On Bluetooth™ security

Nikos Mavrogiannopoulos

December 16, 2005

Abstract

In this paper we discuss several aspects of the security of Bluetooth. Apart from security, several privacy issues linked to the use of Bluetooth protocols are also discussed. The focus is on the lower layer protocols, called the core Bluetooth protocols, thus security aspects of application profiles are left out of the scope of this document. We give a summary of the networking protocols of Bluetooth and try to describe in detail the security protocols. Are these protocol secure enough? Known attacks to the protocols and several privacy issues suggest not. The individual algorithms such as the stream algorithm and also the protocols they operate under seem to be vulnerable to multiple kind of attacks. As such the Bluetooth protocols alone should not be used to ensure authenticity or privacy.

Contents

Chapter 1

The technology

1.1 Introduction

Bluetooth is a technology that enables all kind of electronic devices to communicate with each other. It is a wireless protocol and is usually used for short distance communications, about 10 to 100 meters. It is more lightweight than other comparable protocols, such as IEEE's 802.11 and is optimized for low power consumption.

It was originally developed by Ericsson, though after 1999 it is being developed by a company consortium called the Bluetooth Special Interest Group or simply SIG. Today several versions of the Bluetooth specification exist with the latest being Bluetooth 2.0, whilst most of the products in the market are still Bluetooth 1.2 or 1.1 compliant. This protocol is also being standardized by IEEE as the 802.15.1 protocol.

The Bluetooth protocol is being used by numerous mobile phone devices as a cheap connection method with nearby devices, by printers and other home appliances. It can be seen as the wireless equivalent of the USB protocol. Eventhough this protocol is not used for high-stake transactions, its popularity and widespread usage, triggers us into looking at its security offerings.

1.2 Roadmap

In the rest of this document we will give a basic description of the protocols involved and list any known possible weakness. In chapter 2 we summarize the basic networking functions available in Bluetooth, focusing more on the lower level protocols. In chapter 3 we discuss the security features of Bluetooth, the known attacks and some privacy issues emerging from the technology usage. A summary of the issues is found in the last chapter.

1.3 The technology

1.3.1 Frequency range

Bluetooth is a wireless protocol that operates on the unlicensed[1] 2.4GHz band. Since version 1.2 it uses an adaptive frequency hopping algorithm to avoid service interruption due to other equipment using the same frequencies, and also to avoid causing interference to other equipment as well.

Frequency range	Channels
2.400 – 2.4835 GHz	$f = 2402 + k$, $k = 0, \ldots, 78$ MHz

Table 1.1: Bluetooth frequency range and channels

As we can see in Table 1.1 there are 79 frequencies in the 2.4GHz band that Bluetooth may use for its hopping algorithm. This hopping algorithm does not add any security on the link, since the hopping sequence is broadcasted in clear at the initiation of a connection.

1.3.2 Distance covered

Not all Bluetooth devices have the same signal strength nor can cover the same distance. Most of the devices have a freedom in selecting their output power level. The Bluetooth specification sorts devices based on their power class which is summarized in Table 1.2.

Power class	Minimum output power	Maximum output power	Distance covered
Class 1	1 mW	100 mW	up to 100 meters
Class 2	0.25 mW	2.5 mW	to 10 meters
Class 3	1 mW	1mW	up to 1 meter

Table 1.2: Bluetooth power classes

[1] in most European countries

Chapter 2

Networking

2.1 Introduction

All Bluetooth devices hold a unique address called the Bluetooth Device Address. This is a 48 bit number assigned at production time to the device by the manufacturer and cannot be altered. This is very similar to the ethernet unique MAC addresses. These device addresses are usually represented in hexadecimal colon separated format such as `00:0f:fa:ad:ea:f0`. The importance of these addresses in networking is substantial since they are used for device identification in a network.

2.2 Layers

The lowest Bluetooth core protocol layers are shown in Figure 2.1.

L2CAP layer	responsible for managing the ordering of submission of PDU fragments to the baseband and scheduling
Link Manager Protocol layer	responsible for all aspects of a Bluetooth connection, such as power control, roles, encryption etc.
Link Controller layer	responsible for the encoding and decoding of Bluetooth packets from the data payload and parameters related to the physical channel, logical transport and logical link
Radio layer	responsible for the actual transmitting and receiving of packets of information on the physical channel

Figure 2.1: Bluetooth core protocol stack

The most important transport layers available in the Bluetooth link controller are summarized below

- **SCO:** This is the Synchronous Connection Oriented transport, which is a point to point channel between a master and a slave. It has a constant data rate whilst no retransmission is available. It is typically used for voice connections.
- **eSCO:** The extended Synchronous Connection Oriented transport, which is a point to point channel between a master and a slave. It offers some extensions over the SCO that include a limited retransmission.
- **ASB:** The Active Slave Broadcast transport carries L2CAP traffic to the devices in the piconet that are connected to the channel of ASB. This is uni-directional communication between the master and the slaved and there is no acknowledgment of packet receipt.
- **ACL:** The Asynchronous Connection Oriented transport carries the Link Manager and the L2CAP control and data. It is a bidirectional point to point protocol.

2.3 Network structure

Since in wireless networks devices do not really share a physical link, such as a common cable, some other way of joining a network has to be used. In Bluetooth networked devices are in a common piconet channel. That means that they share a common clock and a common frequency hopping sequence. The common clock is the clock of a device called the master, which is the same device that provides the hopping sequence. All the other devices are called slaves. Devices can be members in several piconets and in that case they are called as being part of a scatternet as shown in figure Figure 2.2. Routing between such piconets is not part of the Bluetooth core protocols and thus left for upper layer protocols.

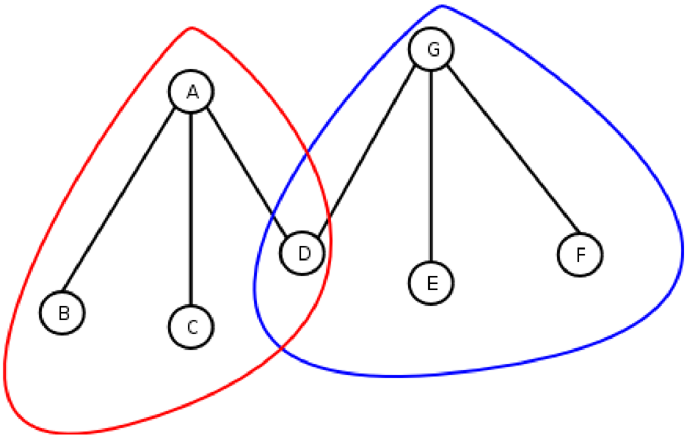

Figure 2.2: A scatternet consisting of two piconets

2.4 Networking functions

Some low level networking functions of Bluetooth are Inquiry and Paging. Inquiry is the procedure of discovering for nearby devices. This is done by sending inquiry requests. Bluetooth devices that are available to be found listen for these inquiries and send responces. The physical channel used for these requests and responces is a separate one.

Paging is the procedure of a device connecting to another one. This is a targeted procedure which means that only one device will respond to this request. For this request also a separate physical channel is used to listen for the requests.

2.5 Networking protocols

Bluetooth provides two networking protocols to programmers that resemble the Internet UDP and TCP protocols. These are the L2CAP and the RFCOMM protocols.

2.5.1 L2CAP

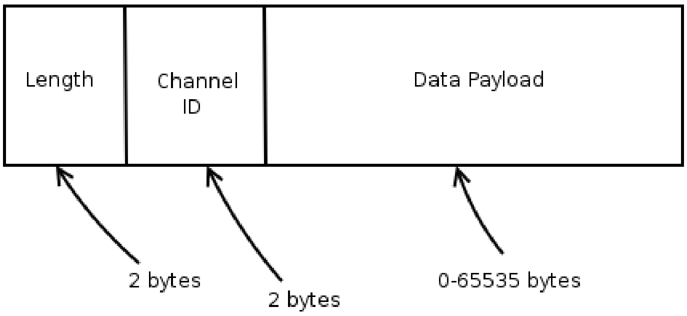

Figure 2.3: L2CAP PDU format

L2CAP is a high level protocol that provides a connection oriented and connectionless messaging to upper layer protocols. Its features are connection flow control, error detection and segmentation and reassembly of messages. It is built around the concept of channels, a notion similar to the TCP ports. Any L2CAP channel is described by a number between the range 1-65535. L2CAP can operate in several modes, such as basic, flow control mode and retransmission mode. All the modes deliver unreliable communication similar to UDP except for the retransmission mode. The L2CAP PDU format for a connection-oriented channel is shown in Figure 2.3. In retransmission mode a more complex format is used and also supervisory frames are introduced to handle for acknowledge packets. This is shown in Figure 2.4.

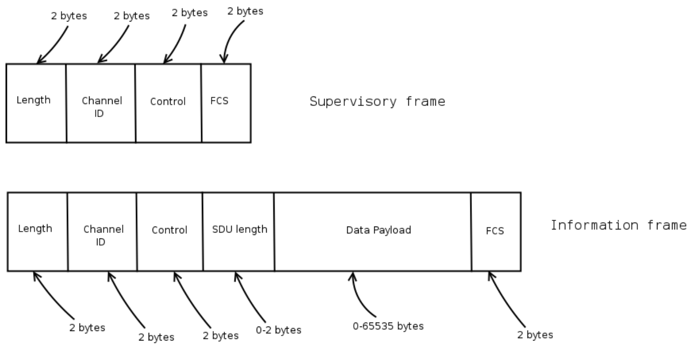

Figure 2.4: L2CAP PDU format

2.5.2 RFCOMM

Before retransmission L2CAP mode was introduced the only way to use a reliable network mode such as TCP was to use the RFCOMM channels. This protocol is built over the L2CAP protocol and offers an emulation for a serial cable. It was intended as a wireless replacement for RS-232 serial communication applications and included the control signals. It offers 20 connection channels, as opposed to 65535 of L2CAP and this made tricky the allocation and usage of the RFCOMM channels. Despite being a serial communication emulator, it is very often used as a reliable transport layer.

Chapter 3

Security

3.1 Point to point networking

Since Bluetooth offers reliable and unreliable transport layers any existing security protocol, such as TLS[1] and IPSec[2], can be used over them. However the specifications (see [3] and [4]) define also some security protocols to be included in all Bluetooth equipment. Such features were considered mandatory because of the inherited insecurity of the wireless networks. This includes authentication, which is mandatory and optionally encryption. These protocols operate on the Link Manager layer, that is below the transport layer protocols, such as L2CAP or RFCOMM.

These methods are used in the context of a connection procedure, called pairing. In that process a secure link is created between two devices. The security modes available for pairing are

- **mode 1:** non secure mode;
- **mode 2:** non secure mode until a channel establishment has been initiated;
- **mode 3:** link level enforced security; The security process is executed before the link is established.

Each device can be configured in a different security mode. Some examples of pairing procedures are shown in Figure 3.1 and Figure 3.2 in pages 8 and 9. Also a feature of the Bluetooth protocol is that multiple devices can be paired together thus share common authentication and encryption keys. In this document we will not discuss this feature.

3.1.1 Authentication

Authentication is the process in which somebody verifies that a second party is the one it claims to be. In the Bluetooth world the authentication between two

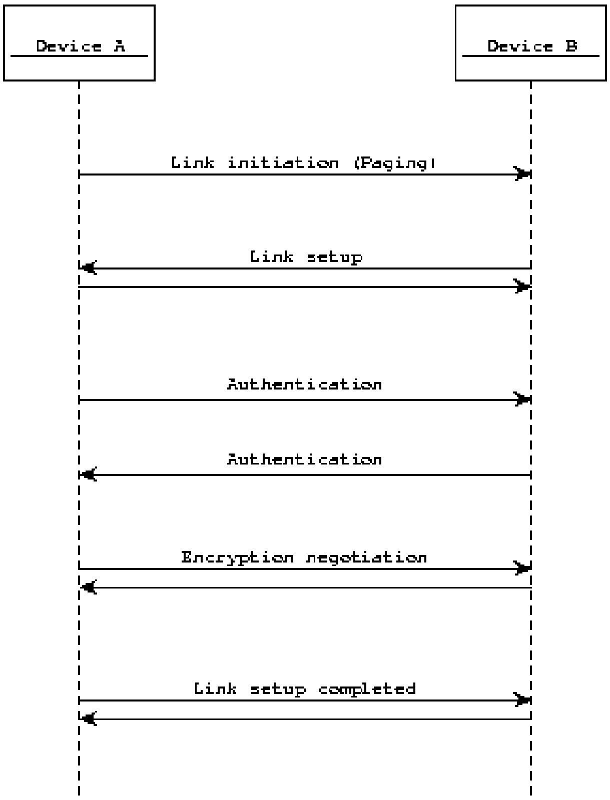

Figure 3.1: A pairing example where both devices are in security mode 3

devices covers only the knowledge of a common secret key (called PIN) and the knowledge of the device addresses[1]. In the context of [7] we could say that the form of this authentication protocol is an agreement protocol.

The authentication process is a simple challenge and responce protocol as depicted in Figure 3.3 in page 10 and involves the following steps:

1. generation of an initialization key
2. generation of an authentication key[2]
3. authentication

Initialization key

The initialization key is created during the first pairing attempt and is there to protect the transfer of the initialization parameters. It is also used used

[1] Eventhough the addresses are not verified using a third party, one can be assured that the device he speaks to used the given address to advertise itself. Thus attacks like man in the middle between two distinct devices that are not willing to talk with are avoided.

[2] This is called the link key in the Bluetooth specification.

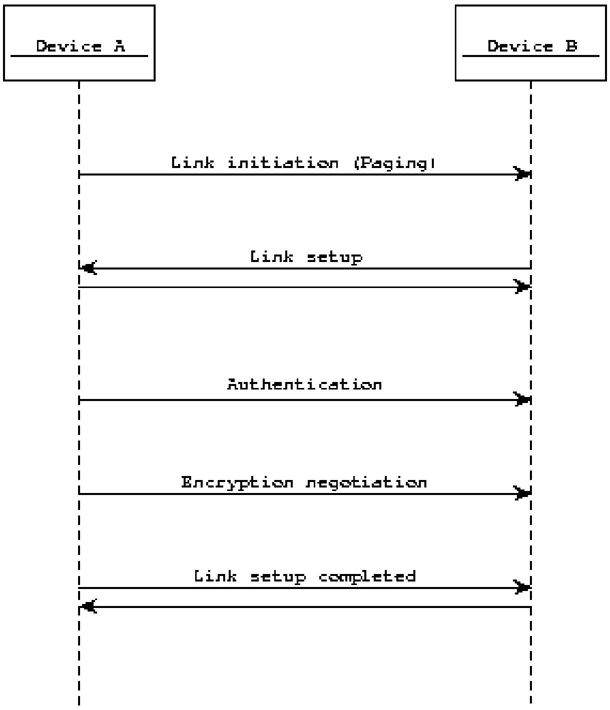

Figure 3.2: Pairing with device A in security mode 3 and B in mode 1

to generate the authentication key. This key is generated using the following formula

$$K_{init} = E_{22}^{*}(BD\_ADDR, PIN, length(PIN), IN\_RAND)$$

where PIN is a user provided sequence of bytes and IN_RAND is an 128 bit random number exchanged during the pairing initiation and BD_ADDR is the address of the device that received the IN_RAND value. E_{22} is a cryptographic hash functions and the star in E_{22}^{*} indicates a difference between this function and the actual one defined in [3]. In more detail the K_{init} is calculated as

$$\begin{aligned} K_{init} = E_{22}(PIN', RAND, L') = \\ \text{SAFER}^{+}_{oneway}(expand_{128}(PIN'), IN\_RAND[0..14] \cup (IN\_RAND[15] \oplus L'), \\ PIN' = (PIN \cup BD\_ADDR), \text{up to 16 bytes}, \\ L' = min\{16, length(PIN) + 6\} \end{aligned}$$

The algorithm SAFER$^{+}_{oneway}$ is a version of the SAFER+[10] encryption algorithm, modified in a way to make it non invertible[3]. The first parameter is an 128 bit key and the second is the 128 bit input data.

The $expand_{128}$ algorithm just copies its input as many times are required to produce an 128 bit output. If the output is larger then the output is truncated at the first 128 bits.

[3] Done by adding the input of the round 1 to the input of round 3 (see [3] p. 779).

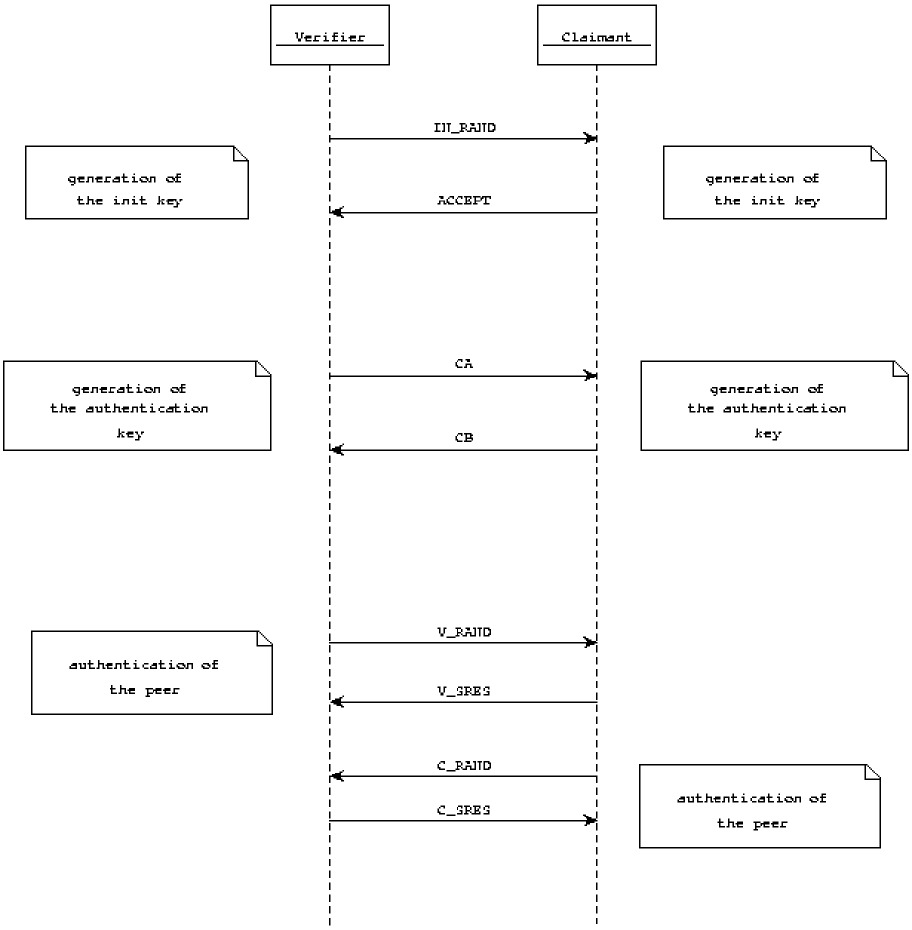
Verifier
Claimant
IN_RAND
generation of the init key
ACCEPT
generation of the init key
CA
generation of the authentication key
CB
generation of the authentication key
V_RAND
authentication of the peer
V_SRES
C_RAND
C_SRES
authentication of the peer

Figure 3.3: A two-way authentication process

Note that the initialization key is only generated at the first pairing attempt. On subsequent attempts the initialization key is the same as the previous authentication key.

Authentication key

In order to perform the authentication step of the protocol a common secret key is required for the parties. This is the role of the Authentication key. This key is generated during the pairing process, however some older versions of the Bluetooth protocols supported a permanently stored key option. Those permanent keys, called unit keys, are now deprecated, thus we will not discuss them further. When only two parties are involved the authentication key is called a combination key and is generated using a key exchange algorithm as shown in Figure 3.3. After this exchange algorithm is completed the two devices share a common authentication key.

Here we explain the algorithm in detail.

C_A: `Verifier` $\xrightarrow{C_A}$ `Claimant`
The verifier device sends this messsage to the claimant device that consists of

$$\begin{aligned} V\_LK &= E_{21}(V\_RAND, V\_ADDR) \\ C_A &= V\_LK \oplus K_{init} \end{aligned}$$

where V_RAND is a random number known to verifier only, V_ADDR is verifier's address.

C_B: `Claimant` $\xrightarrow{C_B}$ `Verifier`
After the receipt of C_A message, the claimant responds with this message. This consists of

$$\begin{aligned} C\_LK &= E_{21}(C\_RAND, C\_ADDR) \\ C_B &= C\_LK \oplus K_{init} \end{aligned}$$

where C_RAND is a random number known to claimant only, C_ADDR is claimant's address.

After this exchange of the messages both parties can calculate a common authentication key as:

Verifier device

$$\begin{aligned} C\_RAND &= C_B \oplus K_{init} \\ KEY &= V\_LK \oplus E_{21}(C\_RAND, C\_ADDR) \end{aligned}$$

Claimant device

$$\begin{aligned} V\_RAND &= C_A \oplus K_{init} \\ KEY &= C\_LK \oplus E_{21}(V\_RAND, V\_ADDR) \end{aligned}$$

E_{21} is a cryptographic hash function, similar to E_{22} as discussed in 3.1.1 and is defined to be

$$\begin{aligned} E_{21}(RAND, ADDRESS) = \\ \text{SAFER}^{+}_{oneway}((RAND[0\ldots14] \cup (RAND[15] \oplus 6)), expand_{128}(ADDRESS)) \end{aligned}$$

Authentication

Here we will discuss the actual authentication method as shown in Figure 3.3 on page 10 and Figure 3.4. This authentication method can be executed twice in a pairing procedure, in order to authenticate both parties. In this algorithm the knowledge of the common authentication key and the right addresses are the basis of authentication. The knowledge of the common authentication key indirectly implies that parties have used the same PIN to produce it.

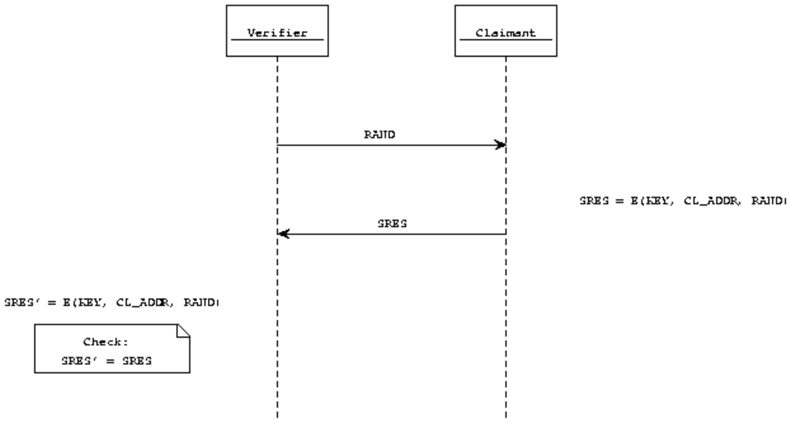

Figure 3.4: The challenge response authentication

RAND: `Verifier` $\xrightarrow{RAND}$ `Claimant`
The verifier device sends a challenge message to the claimant device that consists of an 128 bit random number.

SRES: `Claimant` $\xrightarrow{SRES}$ `Verifier`
After the receipt of $RAND$ message, the claimant responds with this message. This consists of $SRES$, that is calculated as the first 32 bits of $E_1(KEY, ADDR, RAND)$, where KEY is the authentication key and $ADDR$ is the claimant's device address. The verifier then calculates $SRES' = E_1(KEY, ADDR, RAND)$ and if the output matches the received $SRES$ the authentication is complete. In that case the rest 96 bits of the output of E_1 are assigned the name ACO and stored.

The E_1 algorithm is based on the SAFER+ block algorithm and is shown below.

$$E_1(KEY, ADDR, RAND) =$$
$$\text{SAFER}^{+}_{oneway}(\underline{KEY}, expand_{128}(ADDR) + (\text{SAFER}^{+}(KEY, RAND) \oplus RAND))$$
$$\underline{KEY} = \text{linear transformation of the KEY}$$

The *linear transformation* of the key is (for 128 bits keys) performed in byte level and is:

$$\begin{array}{ll}
\underline{K[0]} = K[0] + 233 \mod 256 & \underline{K[1]} = K[1] \oplus 229 \\
\underline{K[2]} = K[2] + 223 \mod 256 & \underline{K[3]} = K[3] \oplus 193 \\
\underline{K[4]} = K[4] + 179 \mod 256 & \underline{K[5]} = K[5] \oplus 167 \\
\underline{K[6]} = K[6] + 149 \mod 256 & \underline{K[7]} = K[7] \oplus 131 \\
\underline{K[8]} = K[8] \oplus 233 & \underline{K[9]} = K[9] + 229 \mod 256 \\
\underline{K[10]} = K[10] \oplus 223 & \underline{K[11]} = K[11] + 193 \mod 256 \\
\underline{K[12]} = K[12] \oplus 179 & \underline{K[13]} = K[13] + 167 \mod 256 \\
\underline{K[14]} = K[14] \oplus 149 & \underline{K[15]} = K[15] + 131 \mod 256
\end{array}$$

3.1.2 Encryption

Encryption is a separate process that starts after authentication is successfully finished. For encryption a different key is used, called the *encryption key* and it allows for sizes from 8 to 128 bits. The fact that the encryption key is not fixed is political rather than technical. Since devices are constructed in countries with different laws about encryption of data, it was allowed in the specification for devices to negotiate the encryption key size (see [3] p. 749). The process of enabling encryption consists of the steps:

- encryption negotiation;
- generation of the encryption key;
- encryption of all traffic using this key.

The encryption applies only to the payload of the Bluetooth packets (see Figure 3.8 on page 16). The headers are never encrypted.

Encryption negotiation

The process of negotiating encryption is shown in Figure 3.5 and explained further in this section. Initially the initiator device sends an encryption mode request message to the peer device. The encryption mode can be either enable encryption or not. If usage of encryption is requested a negotiation of the encryption size follows. This negotiation may occur multiple times until an acceptable by both sides key size is negotiated. The last phase includes the sending of a random number by the initiator device in order for both peers to calculate the encryption key. After this key is calculated encryption is enabled.

It should be noted that this process is clearly vulnerable to man in the middle attacks, thus both devices must terminate the negotiation if the negotiated values are considered to be weak or insecure.

Encryption key

The encryption key is generated using the current authentication key, an 128 bit random number exchanged during the encryption negotiation and the 96 bit value of ACO as generated in the authentication process. To generate the key the E_3 algorithm is used that is defined as:

$$K_{encr} = E_3(KEY, ACO, RAND) =$$

$$\text{SAFER}^{+}_{oneway}(\underline{KEY}, expand_{128}(ACO) + (\text{SAFER}^{+}(KEY, RAND) \oplus RAND))$$

$$\underline{KEY} = \text{linear transformation of the KEY}$$

The expansion algorithm and the transformation of the key are the same as in 3.1.1.

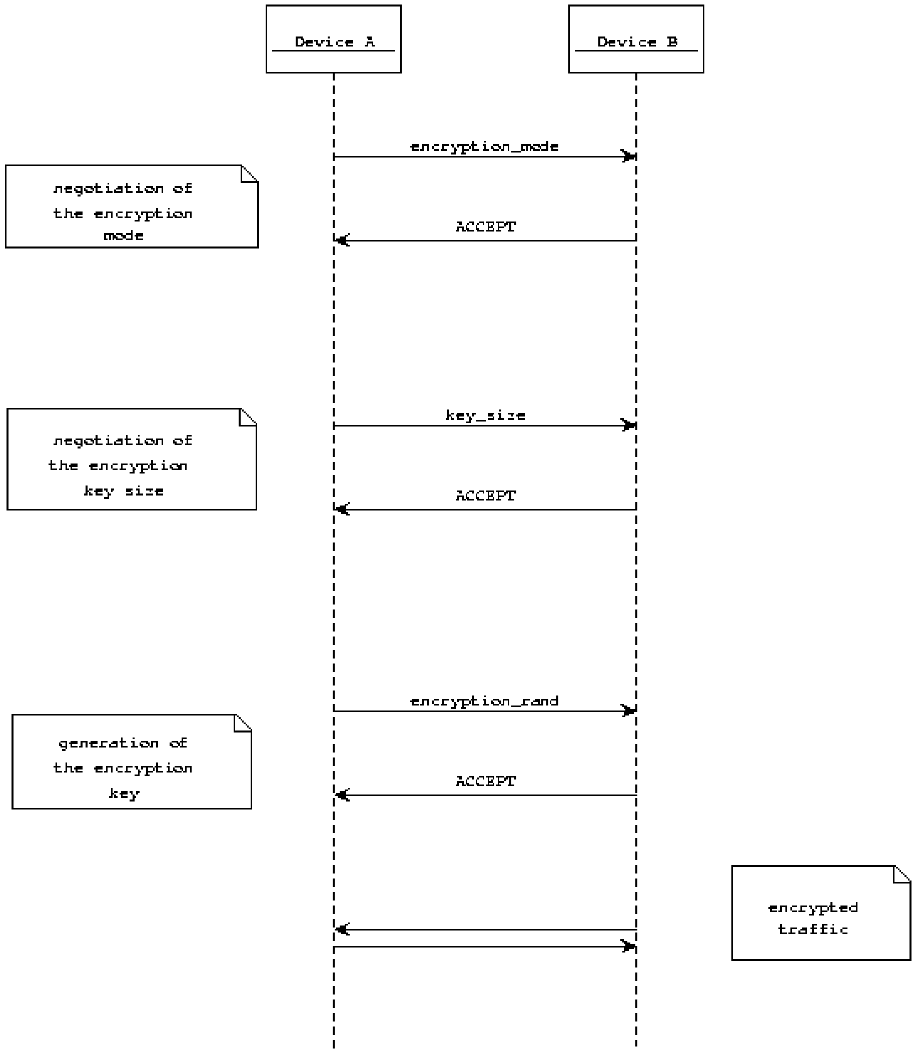
Device A
Device B
encryption_mode
negotiation of the encryption mode
ACCEPT
key_size
negotiation of the encryption key size
ACCEPT
encryption_rand
generation of the encryption key
ACCEPT
encrypted traffic

Figure 3.5: Encryption negotiation

Encryption algorithm

Due to the nature of the hardware used to perform encryption, no block cipher is used as an encryption algorithm in Bluetooth. For efficiency reasons and power consumption a stream cipher called E_0, based on Linear Feedback Shift Registers (LFSR) was used. Four shift registers are used in the algorithm an a non-linear part that combines their output as shown in Figure 3.6. The plaintext is then combine with the output key stream using an exclusive or.

The operational mode of the algorithm is quite peculiar. The stream cipher is initialized on every new packet to be encrypted with the following data

- the encryption key;

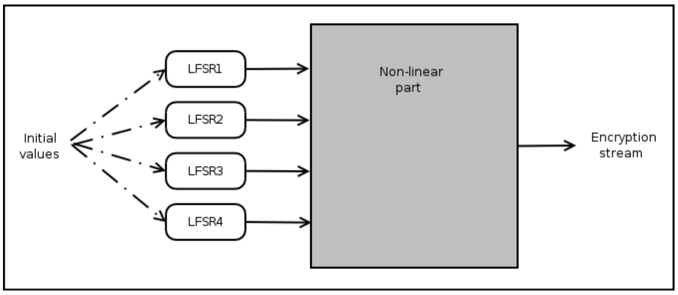

Figure 3.6: Encryption algorithm

- the master device address;
- the 26 bits of the master clock.

Although it will not be discussed in detail the cipher initialization phase includes an encryption key reduction to whatever bits it has been negotiated and setting the cipher's initial encryption state.

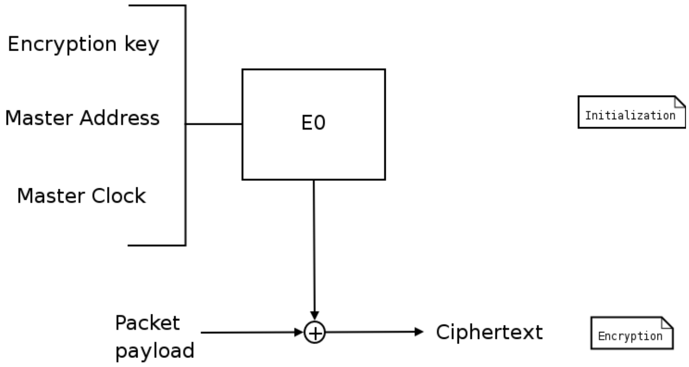

Figure 3.7: The E_0 algorithm operational mode

Encryption process

The cipher is being initialized on every packet transfer (receive or send). In a typical packet, such as the one depicted in Figure 3.8 the payload data – including the header and the CRC code– are encrypted. The stream cipher's operations per packet can been seen at Figure 3.7.

3.1.3 Packet authenticity

After Bluetooth authentication and encryption are enabled all the exchanged packets, such as the one in Figure 3.8, are transferred with an encrypted payload. As we saw before there is no message authentication code or any authentication

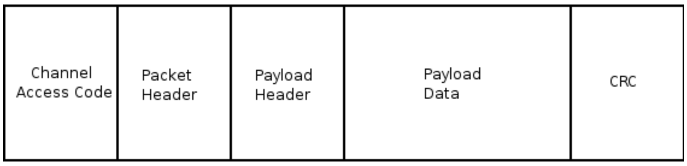

Figure 3.8: A bluetooth packet

payload per packet appended except for the encryption. Thus the only protection is the encrypted 16 bit CRC code. However because of the linearity of the encryption, which is just an exclusive OR over the plaintext, it is very easy to manipulate the CRC code and the ciphertext in order to produce a different output when the transmitted data are known. Thus it should be noted that Bluetooth offers NO packet authenticity[4].

[4] A remark here that can be made is that packets are only valid for the time slot they have been sent. If sent later or if their receipt is delayed then they are considered to be invalid. This makes it difficult to manipulate them, unless this attack is combined with an attack in the time synchronization protocol.

3.2 Multi-point networking

Except for point to point networking Bluetooth allows for a whole piconet's traffic to be encrypted. This is achieved by encrypting all traffic with a common key for all points. In that case case all devices in the piconet can eavesdrop all traffic of the network including traffic not intended for them.

3.2.1 Authentication

The authentication process in the multi-points case is similar to the point to point case except for an extra step. After an authentication key has been generated the master generates a key to be used in the piconet and exchanges it with all the slaves. The procedure is shown in Figure 3.9 and explained below.

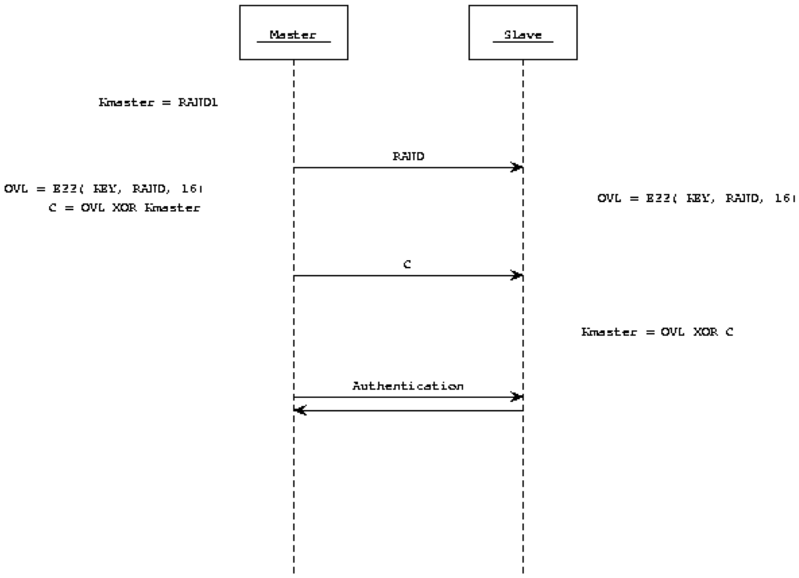

Figure 3.9: Authentication key generation

$RAND$: `Master` $\xrightarrow{RAND}$ `Slave`
After the master has generated a master key, say K_{master}, it also needs to generate a second number called $RAND$ and he sends it to the slave.

C: `Master` $\xrightarrow{C}$ `Slave`
At this point the master generates C as:

$$\begin{aligned} OVL &= E_{22}(KEY_{auth}, RAND, 16) \\ C &= OVL \oplus K_{master} \end{aligned}$$

After the receipt of C message the slave calculates OVL and then the master key as

$$OVL = E_{22}(KEY_{auth}, RAND, 16)$$
$$K_{master} = OVL \oplus C$$

At this point both sides share the same master key. Authentication as in 3.1.1 is repeated in order for devices to confirm the success of this transaction. This process needs to be repeated by the master with all the slaves in the piconet.

3.2.2 Encryption

After a successful authentication encryption can be enabled. In that case the master has to negotiate the appropriate key length with all the slaves and probably drop from the network incompatible slaves. As in the point to point case the encryption negotiation step is performed. The only difference is that the encryption key is calculated using the master key, the random value from the encryption negotiation and the address of the master as

$$K_{encr} = E_3(K_{master}, RAND, BD\_ADDR \cup BD\_ADDR)$$

3.3 Attacks in the protocols

Several attacks have been found on the E_0 stream cipher such as [9], [8] and [6] that reduce the effective key length from 128 bits to 84 bits. These attacks and the lack of integrity (see subsection 3.1.3) suggest that the Bluetooth security protocols should not be used for transactions that require a high level of security.

Other attacks such as [11] recover a PIN used to secure a connection in a few seconds for PINs of less than 8 characters. This can be done by just an evesdropper since this is an off-line attack. Also in that paper an optimized version of SAFER+ is proposed and used for the attack.

3.4 Privacy

Because Bluetooth is included in devices such as mobile phones that most people carry with them all the time and every device is uniquely identified by its device address, privacy issues are raised.

In a scale smaller than GSM networks people can be traced – for example within a building – and their approximate position could be known in real time. This can be done, for example by having multiple Bluetooth access points that send inquiry requests for new devices each second. Any device that gets into their range could then be detected and identified using the Bluetooth device address. Even if the distance to the device cannot be known with high precision due to many differences in signal strength adjustment by several manufacturers, the address itself could be used to identify a visible device. That is because the address, just like MAC addresses, have a fixed per manufacturer part, that would help distinguish a Nokia phone from an Ericsson one.

For this reason all Bluetooth devices have the option to disable listening and responding to inquiry requests, and this will prevent attacks based on device scanning. However this option does not protect from devices searching for a particular device address[5]. The only protection to this attack is to disable Bluetooth completely on the device.

Other privacy issues can be raised when several Bluetooth equipment is used in a household. In that case everyone within the range of the transmission (typically 100 meters) can eavesdrop or even alter the data exchanged between the available devices. For example a neighbor could eavesdrop on the contacts list while copying it to the mobile phone, or view all the documents sent to the printer and a long list of other possibilities could be added. More information about attacks to the Bluetooth privacy are discussed in [5].

[5] For example, can be done by paging to the device to be traced.

Chapter 4

Summing up

Although there seem to be quite some problems in the security aspects of Bluetooth, such as the encryption and the authentication algorithms, one could justify the weaknesses by the special characteristics of the network. Firstly one should note that a device's range normally is up to 10 meters, with a maximum of 100, thus the threats can be minimized in such a constraint environment by other means. Also the most common uses are for communication between mobile phones, that normally do not require tremendous security features, whilst they do require low power consumption. This is also the reason why an LFSR based cipher was chosen instead of a well studied block cipher such as AES which is considered unbreakable today.

Other reasons that relate to human interface mandated the suggestion of short PINs instead of passphrases for the authentication procedure, that make offline attacks such as [11] possible. Though this can be seen as a weakness, it is also important to consider who the users of these protocols are. No one could market a device that requires to type a 20 digit passphrase just to transfer an image from the mobile to his computer. However the option of using longer PINs is available and despite the name that implies numbers only, they can, and it is recommended to, include any ASCII characters.

However these justifications for the weaknesses are not adequate. Although we did not formulate specific capabilities for an attacker and a typical Bluetooth user's security needs are not high, he should have the option to increase the level of security when required. This is not possible with the current technology, which is inferior say to a well established network security technology such as TLS[1]. So we believe that privacy issues of using Bluetooth will increase as it is becoming a mainstream technology. Currently, when security is required for Bluetooth links and the resources are available[1] the usage of a secure network such as IPSec or TLS over its protocols is recommended.

[1] For example a Bluetooth link when between two computers.

Bibliography

[1] T. Dierks and C. Allen. The tls protocol version 1.0, 1999. Available from http://kaizi.viagenie.qc.ca/ietf/rfc/rfc2411.txt.

[2] R. Thayer et al. Ip security document roadmap, 1998. Available from http://kaizi.viagenie.qc.ca/ietf/rfc/rfc2411.txt.

[3] Bluetooth Special Interest Group. Specification of the bluetooth system: core package version 1.2, 2003. Available from http://www.bluetooth.org.

[4] Bluetooth Special Interest Group. Specification of the bluetooth system: core package version 2.0 + edr, 2004. Available from http://www.bluetooth.org.

[5] Markus Jakobsson and Susanne Wetzel. Security weaknesses in bluetooth. *Lecture Notes in Computer Science*, 2020:176+, 2001. Available from http://www.informatics.indiana.edu/markus/papers/bluetooth.pdf.

[6] O. Levy and A. Wool. A uniform framework for cryptanalysis of the bluetooth e_0 cipher. Cryptology ePrint Archive, Report 2005/107, 2005. Available from http://eprint.iacr.org/2005/107.pdf.

[7] Lowe. A hierarchy of authentication specifications. In *PCSFW: Proceedings of The 10th Computer Security Foundations Workshop*. IEEE Computer Society Press, 1997.

[8] Y. Lu, W. Meier, and S. Vaudenay. The conditional correlation attack: A practical attack on bluetooth encryption. In *Advances in Cryptology - Crypto 2005*. Springer-Verlag, 2005. Available from http://www.iacr.org/conferences/crypto2005/p/16.pdf.

[9] Y. Lu and S. Vaudenay. Cryptanalysis of the bluetooth keystream generator two-level e0. In *Advances in Cryptology - Asiacrypt 2004*. Springer-Verlag, 2004. Available from http://www.iris.re.kr/ac04/data/Asiacrypt2004/11SymmetricKeyCryptanalysis/04_YiLu.pdf.

[10] J. Massey. Nomination of safer+ as candidate algorithm for the advanced encryption standard (aes), 1998. Submission document from Cylink Corporation to N.I.S.T.

[11] Y. Shaked and A. Wool. Cracking the bluetooth pin. In *3rd USENIX/ACM Conf. Mobile Systems, Applications, and Services (MobiSys)*. ACM, 2005. Available from http://www.eng.tau.ac.il/~yash/shaked-wool-mobisys05/index.html.