The Link Level Security Implementation of Bluetooth Technology

By

Zheng Sun
School of Computer Science
McGill University
Montréal, Québec
Canada

August 2001

A Thesis submitted to the
Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of
Master in Science

Abstract

Bluetooth technology is a de facto global standard for a short-range, low-power and low-cost form of wireless communication. New as it is, the technology is anticipated to play a very important role in an increasingly mobile world.

Like almost any other modern communication technology, Bluetooth must be sufficiently secure before it can be widely accepted. This is particularly challenging for Bluetooth due to the fact that it uses radio as the physical communication media. The nature of microwave links, as opposed to fixed cable connections, means that Bluetooth is especially vulnerable to attacks. However, through the use of some well-known and highly-effective cryptographic algorithms, Bluetooth is able to achieve the three most important security goals: secrecy, authenticity and integrity.

The focus of the thesis is how Bluetooth translates the theoretical cryptographic techniques into effective real life implementations. Security measures in Bluetooth are taken at both the application level and the link level. In the thesis, the author places the emphasis on the link level measures. Various security areas are discussed, such as random number generation, key management, encryption, authentication, etc. The limitations and weaknesses of Bluetooth security implementations are also discussed, and some suggestions for future improvements are offered.

Résumé

La technologie de Bluetooth est une norme globale de facto pour une forme de transmission sans fil à courte portée, à basse puissance et à prix réduit. Toute nouvelle qu'elle soit, on prévoit que cette technologie jouera un rôle très important dans un monde de plus en plus mobile.

Comme presque n'importe quelle autre technologie moderne de transmission, Bluetooth doit être suffisamment sécure avant qu'elle puisse être largement acceptée. C'est un défi particulièrement difficile pour la technologie Bluetooth étant donné qu'elle utilise la radio comme medium de transmission physique. La nature des liens par micro-ondes, par opposition aux connections par câbles fixes, signifie que Bluetooth est particulièrement vulnérable aux attaques. Cependant, par l'utilisation de quelques algorithmes cryptographiques bien connus et très efficaces, Bluetooth peut réaliser les trois buts de sécurité les plus importants: le secret, l'authenticité et l'intégrité.

Le centre de la thèse consiste en comment Bluetooth traduit les techniques cryptographiques théoriques en réalisations efficaces dans la réalité. Des mesures de sécurité dans Bluetooth sont prises au niveau des applications ainsi qu'au niveau des liens. Dans la thèse, l'auteur met l'accent sur les mesures de niveau des liens. De diverses zones de sécurité sont discutées, comme la génération de nombres aléatoires, la gestion de clés, le chiffrement, l'authentification, etc. Les limitations et les faiblesses des réalisations de sécurité de Bluetooth sont également discutées, et quelques suggestions pour de futures améliorations sont offertes.

Acknowledgements

First and foremost, I would like to express my gratitude to my supervisor, Professor Gerald Ratzer. His constant guidance and encouragement have helped me enormously along the way. Without his supervision, it would have been much more difficult for me to complete the thesis, if it was ever possible. One thing that impresses me the most about Professor Ratzer is that while he never gives up his supervisor role, he allows the maximum amount of academic freedom to his students. Whenever possible, he encourages the students to think on their own, work on their own and discover on their own. Because of this, his students are never confined in any way. Instead, they are always free to explore any academic area in which they become interested. I personally feel this is the best way to supervise.

Also, I would like to thank my parents. They have been the most important people in my life, and they always will be. All my achievements can be credited to them, directly or indirectly. I know there will be no obstacle in my life that I cannot overpass, because their emotional support will be with me forever.

Last but not the least, I would like to thank my fellow students at McGill University and my colleagues at Ericsson Research Canada who helped me in the process of writing the thesis. I will remember them for a long time to come.

Contents

List of Figures

List of Tables

1. Introduction

1.1 Birth of Bluetooth Technology:

In 1994 Ericsson Mobile Communications decided to investigate the feasibility of a low-power, low-cost radio interface between mobile phones and their accessories. The idea was that a small radio built into both the cellular telephone and the laptop would replace the cumbersome cable used today to connect the two devices. A year later the engineering work began and the true potential of the technology began to crystallize. Beyond unleashing devices by replacing cables, the radio technology showed possibilities to become a universal bridge to existing data networks, a peripheral interface, and a mechanism to form small private ad hoc groups of connected devices away from fixed network infrastructures. This technology is known today as Bluetooth technology.

1.2 Why Bluetooth?

Harald Bluetooth was a Viking king who ruled Denmark between 940 and 981. One of his greatest skills was to make people talk to each other, and during his rule Denmark and Norway were Christianized and united. Since most of the original Ericsson researchers have Scandinavian backgrounds, and they believe that their new technology can enable people to communicate to each other more freely, like the old Viking king could, thus they named the technology as Bluetooth [1].

1.3 What Exactly is Bluetooth?

Bluetooth is a de facto global standard that [2]:

- Eliminates wires and cables between both stationary and mobile devices
- Facilitates both data and voice communication
- Offers the possibility of ad hoc networks and delivers easy synchronicity between all personal electronic devices.

The Bluetooth wireless technology comprises hardware, software and interoperability requirements. It has been adopted not only by all major players in the telecom, computer and home entertainment industry, but also in such diverse areas as the automotive industry and health care, automation and toys, etc. – almost all sectors of the economy. It is expected that before long Bluetooth technology will be built into hundreds of millions of electronic devices all over the world.

1.4 Why Use Bluetooth Technology?

Mobility among people has constantly grown and wireless technologies for voice and data have evolved rapidly during the past years. Countless electronic devices for home, personal and business use have been presented to the market during recent years but no widespread technology has addressed the needs of connecting personal devices in Personal Area Networks (PAN). The demand for a system that could easily connect devices for transfer of data and voice over short distances – without cables – grew stronger [2].

Bluetooth wireless technology fills this important communication need with its ability to communicate both voice and data wirelessly, using a standard low-power, low-cost technology which can be integrated in all devices and thus enable total mobility. To illustrate the benefits and power of Bluetooth technology, four usage models are presented here [3]:

- *Ultimate Headset:*

  The headset allows you to use your mobile phone even if the phone is in a briefcase, so you can always keep your hands free for more important tasks when you are at the office or in your car.

- *Automatic Synchronization:*

  Automatic synchronization of calendars, address books, e-mails, etc. is a long-awaited feature. As soon as you enter your office, the calendar on

your phone or PDA will be automatically updated to be consistent with the one in your desktop PC, or vice versa. Phone numbers and addresses will always be correct in all your portable devices without docking through cables or infrared.

- *Internet Bridge:*

Wherever you are, Bluetooth wireless technology lets you surf the Internet without any cable connections. You can access the net wirelessly using either a portable computer or a cellular phone.

- *Three-in-One Phone*

At home your phone can function as a cordless phone (fixed-line charge). When you are on the move, your phone can function as a cellular phone (cellular charge). And when your phone comes within range of another cellular phone with built-in Bluetooth technology it can function as a walkie-talkie (no telephony charge).

1.5 What is the Bluetooth SIG?

In February 1998 the Special Interest Group (SIG) was formed. Today the Bluetooth SIG includes promoter companies Ericsson, IBM, Intel, 3Com, Lucent, Microsoft, Motorola, Nokia and Toshiba, and thousands of Adopter/Associate member companies [3].

The assignment of the SIG originally was to monitor the technical development of short range radio and create an open global standard, thus preventing the technology from becoming the property of a single company. This work resulted in the release of the first Bluetooth specification in July 1999. The further development of the specification is still one of the main issues for the SIG, other important tasks are interoperability requirements, frequency harmonization and promotion of the technology.

The SIG has developed the Bluetooth Qualification Program which guarantees global interoperability between devices regardless of the vendor and regardless of the country in which they are used. Full interoperability between

different devices from various manufacturers, provided the devices share the same profile, is an essential element that will ensure the success of Bluetooth. During the test procedure that all devices must pass, it must be verified that they meet all requirements regarding: radio link quality, lower layer protocols, security strength, profiles and information to end-users, etc. All qualified devices are certified by the SIG.

1.6 Security Features of Bluetooth:

Like almost any other modern communication technology, Bluetooth must be sufficiently secure before it can be widely accepted. This is particularly challenging for Bluetooth. The nature of microwave links, as opposed to fixed cable connections, means that Bluetooth is especially vulnerable to attacks. However, through the use of some well-known and highly-effective cryptography algorithms, Bluetooth is able to achieve the three most important security goals: secrecy, authenticity and integrity.

Secrecy refers to denial of access to information by unauthorized individuals; authenticity refers to validating the source of a message, i.e. that it was transmitted by a properly identified sender; integrity refers to the assurance that a message was not modified accidentally or deliberately in transit, by replacement, insertion or deletion [4].

The focus of the thesis is how Bluetooth translates the theoretical cryptography techniques into effective real life implementations. Security measures are taken at both the application level and the link level. In this paper, the author places the emphasis on the link level measures. Various security areas are discussed, such as random number generation, key management, encryption, authentication, etc. The limitations and weaknesses of Bluetooth security implementations are also discussed, and some suggestions for future improvements are offered.

1.7 Organization of the Thesis:

The thesis is organized as follows:

1. **Introduction:** In this section, the author introduces the Bluetooth technology as a de facto global standard for a short-range, low-power and low-cost form of wireless communication. The various actual and potential applications of Bluetooth are also briefly discussed in order to demonstrate the important role the technology can play in an increasingly mobile world. Then the author points out that it is particularly challenging to build sufficient security features into Bluetooth due to the physical nature of microwaves. It becomes, therefore, the interest of the thesis to focus on the security implementations in Bluetooth, especially the security measures taken at the link level since they are unique to Bluetooth technology.
2. **Survey of Bluetooth Technology:** Various essential elements of Bluetooth technology are concisely examined in this section. Radio, as the physical link between Bluetooth devices, forms the foundation of the technology. Important wireless concepts such as ISM band, frequency hopping and spread spectrum are explained. One of the biggest strengths of Bluetooth is its ability to set up ad hoc networks in an extremely flexible and efficient fashion. Piconets and scatternets are new and unique ways to achieve this kind of dynamic network architecture. Bluetooth hardware architecture includes the analog part, which transmits radio waves, and the digital part, which processes the signals. Bluetooth software architecture is basically a protocol stack. Different layers of the stack serve different functions. This kind of the architecture makes the design more modular and enables the inter-operability between Bluetooth devices from various vendors. The OSI network model and Bluetooth protocol stack makes an interesting comparison.
3. **Basic Concepts of Cryptography:** The science of cryptography forms the foundation of most modern information security measures. This section provides an overview of the development of cryptography, then presents some

basic but important concepts such as encryption, authentication, keys, stream ciphering, challenge-response protocol, random number generator, Diffie-Hellman algorithm, etc. As shown in section 4 and 5, the security implementation of Bluetooth uses these concepts extensively.

4. **Service Level Security Implementation in Bluetooth:** Security measures are provided at both the application level and the link level in Bluetooth. While the focus of the paper is the security at the link level, section 4 examines the measures taken at the application level in order to complete the whole picture. The general architecture of the Bluetooth security at the service level is presented. It shows that different layers of the Bluetooth protocol stack can all interact with the security manager, which is the central component of the security implementation. This kind of architecture provides a framework with a high flexibility and a high usability.
5. **Link Level Security Implementation in Bluetooth:** This section is by far the most important part of the entire paper. It explains in great details how the security is implemented at the Bluetooth link level. It first introduces the pseudo-random number generator, which is one of the four most important security entities. Then, the section discusses thoroughly how the different types of keys are generated, initialized, negotiated and exchanged. Key management is critical in any cryptographic system. Using these keys, Bluetooth performs important security functions such as encryption and authentication. Last but not the least, the various key-generating functions are presented and studied in the end.
6. **Evaluation and Conclusion:** The last section of the paper evaluates the overall security of Bluetooth. It reveals several known problems with the link level security and the consequences that these problems can cause. It offers suggestions for future improvements. In the end, the author concludes that the current security features of Bluetooth are sufficient for its intended normal use. However, if Bluetooth is to become a high end wireless standard, as it is evolving towards now, it must strengthen its overall security.

2. Survey of Bluetooth Technology

2.1 Radio:

The Bluetooth specification defines a short (around 10m) or optionally a medium range (around 100m) radio link capable of voice or data transmission to a maximum capacity of 720 kbps per channel.

Radio frequency operation is in the unlicensed industrial, scientific and medical (ISM) band at 2.4 to 2.4835 GHz, using a spread spectrum, frequency hopping, full-duplex signal at up to 1600 hops/sec [5].

The ISM band is occupied by a plethora of other radio frequency emitters which contribute significantly to the noise floor. These emitters range from wireless applications such as short-range proprietary techniques (car security, cordless headphones, etc.) and WLAN (Wireless Local Area Network) to random noise generators such as microwave ovens and sodium vapour street lamps. As a result, the 2.4 GHz band is not a terribly stable or reliable medium. However, the band's worldwide availability ensures widespread acceptance of Bluetooth. To cope with the hostile environment presented by the ISM band, Bluetooth specifically employs several techniques: frequency hopping, adaptive power control, and short data packets [6].

The mechanism of frequency hopping is used as a means for secure communication and robust communication. It was invented in the U.S. by an Austrian-born actress Hedy Lamarr during World War II. Lamarr became intrigued with radio-controlled missiles and the problem of how easy it was to jam the guidance signal. She realized that if the signal could be made to jump from one frequency to another very quickly – like changing stations on a radio – and both the sender and receiver changed in the same order at the same time, then the signal could never be blocked without knowing exactly how and when the frequency changes. Although the frequency-hopping idea could not be implemented due to technology limitations at that time, it eventually became the

basis for cellular communication [7]. Bluetooth also takes advantage of this technique.

It is always possible for a radio channel to become temporarily blocked by an interference source, and this is quite likely in the busy ISM band. Even though Bluetooth provides a re-transmission scheme for lost data packets, it is altogether more efficient and robust to retransmit the data on a new channel, which is unlikely to also be blocked [8]. Indeed, the algorithm employed to calculate the hop sequence ensures maximum distance between adjacent hop channels in the sequence. Also several active Bluetooth piconets may be within range of each other; with each piconet hopping independently with a pseudo random sequence based on each piconet's identity/access code, collisions will be minimized. This is important as all Bluetooth devices only have 79 channels in which to operate. In an office or public environment, the number of active devices can very quickly reach the limit, and a low correlation between non-communicating pairs is essential.

Spread spectrum is a technique whereby a modulated wave form is modulated (spread) a second time in such a way as to generate an expanded-bandwidth wideband signal that does not significantly interfere with other signals [9]. Bandwidth expansion is achieved by a second modulation means, a means that is independent of the information message. In spread spectrum modulation, a signal's power is spread over a larger band of frequencies. This results in a more robust signal that is less susceptible to interference from similar radio-based systems, since they too are spreading their signals, but with different spreading algorithms. Spread spectrum has two modes of operation: frequency hopping and direct sequencing. Bluetooth technology uses the frequency-hopping mode of spread spectrum.

The operating band of 83.5 MHz (between 2.4 GHz and 2.4835 GHz) is divided into 1 MHz spaced channels, each signaling data at 1M symbols per second so as to obtain the maximum available channel bandwidth. With the chosen modulation scheme of GFSK (Gaussian Frequency Shift Keying), this equates to 1 Mb/s. A binary 1 gives rise to a positive frequency deviation from the

nominal carrier frequency, while a binary 0 gives rise to a negative frequency deviation.

The signal hops among 79 frequencies at 1 MHz intervals to give a high degree of interference immunity. Radio frequency output is specified as 0 dBm (1mW) in the 10m–range version and –30 to + 20 dBm (100mW) in the longer range version [8].

When producing the radio specification, a high emphasis was put on making a design enabling single-chip implementation in CMOS (complementary metal oxide semiconductor) circuits, thereby reducing cost, power consumption and the chip size required for implementation in mobile devices.

Both voice and data can be transmitted via the radio. Up to three simultaneous synchronous voice channels, or, a channel which simultaneously supports asynchronous data and synchronous voice can be used. Each voice channel supports a 64 kb/s synchronous voice channel in both directions. The asynchronous data channel can support maximal 723.2 kb/s asymmetric (and still up to 57.6 kb/s in the return direction), or 433.9 kb/s symmetric [10].

2.2 Network Architecture:

Bluetooth units that come within range of each other can set up ad hoc point-to-point and/or point-to-multipoint connections. Units can dynamically be added or disconnected to the network. Two or more Bluetooth units that share a channel form a piconet.

Several piconets can be established and linked together in ad hoc scatternets to allow communication and data exchange in flexible configurations. If several other piconets are within range they each work independently and each have access to the full bandwidth. Each piconet is established by a different frequency hopping channel. All users participating on the same piconet are synchronized to this channel [11]. Unlike infrared devices, Bluetooth units are not limited to line-of-sight communication.

To regulate traffic on the channel, one of the participating units becomes a master of the piconet, while all other units become slaves. With the current

Bluetooth specification up to seven slaves can actively communicate with one another.

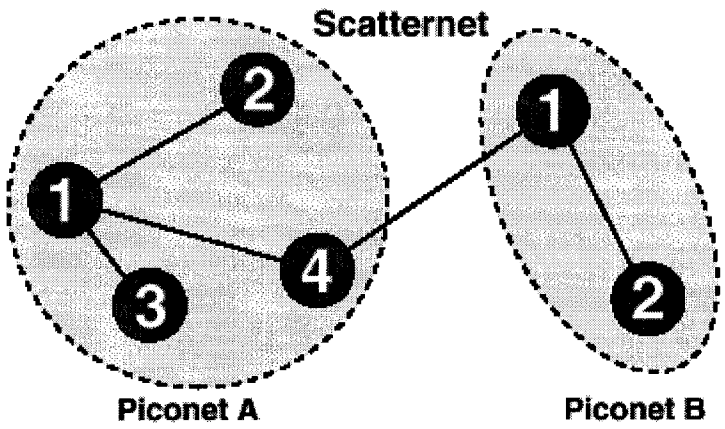

Figure 2.1 Scatternet, ad-hoc combinations in point-to-multipoints and point-to-point scatternet[3].

2.3 Hardware Architecture:

The Bluetooth hardware consists of an analog radio part and a digital part – the host controller. The host controller has a hardware digital signal processing part called the link controller (LC), a CPU core and interfaces to the host environment [12].

The link controller consists of hardware that performs baseband processing and physical layer protocols such as ARQ-protocol and FEC coding. The function of the link controller includes asynchronous transfers, synchronous transfers, audio coding and encryption.

The CPU core allows the Bluetooth module to handle inquiries and filter page requests without involving the host device. The host controller can be programmed to answer certain page messages and authenticate remote links.

The link manager (LM) software runs on the CPU core. The link manager discovers other link managers and communicates with them. Through the link manager protocol (LMP), the link manager is able to perform its service provider role and use the services of the underlying link controller.

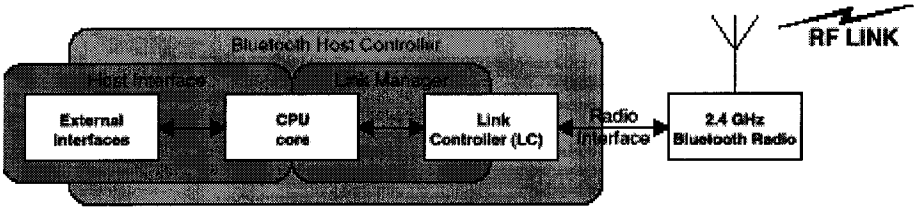

Figure 2.2 Bluetooth Hardware Architecture [12].

2.4 The Protocol Stack:

A key feature of the Bluetooth technology is that it aims to allow devices from lots of different manufacturers to work with one another. To this end, Bluetooth does not just define a radio system, it also defines a software stack to enable applications to find other Bluetooth devices in the area, discover what services they offer, and use those services. The Bluetooth protocols are marked with blue color in Figure 2.3.

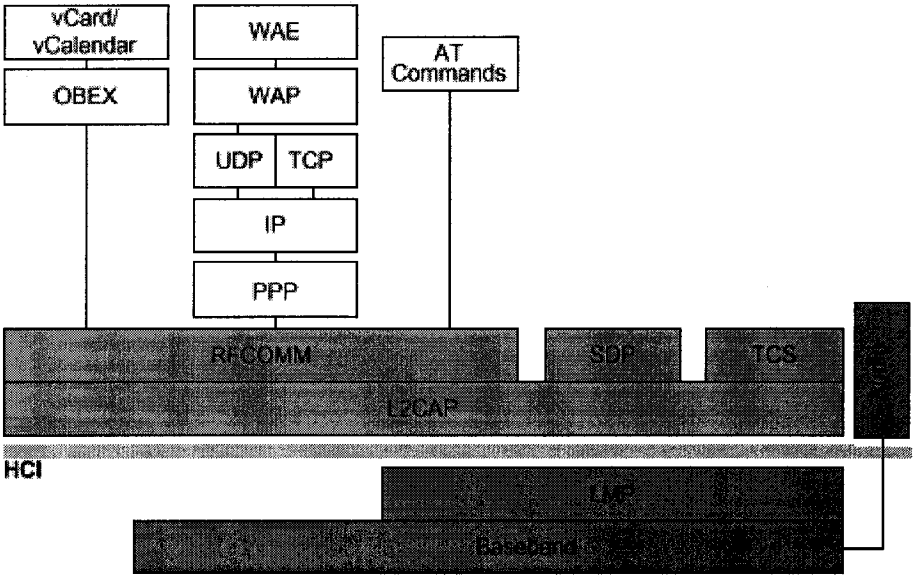

Figure 2.3 Bluetooth Software Architecture [12].

The different Bluetooth protocol layers serve the following functions [6]:

- The Baseband and Link Controller controls the physical links via the radio, assembling packets and controlling frequency hopping.
- The Link Manager Protocol (LMP) controls and configures links to other devices. The Host Controller Interface (HCI) handles communications between a separate host and a Bluetooth module.
- Logical Link Control and Adaptation Protocol (L2CAP) multiplexes data from higher layers, and converts between different packet sizes.
- RFCOMM emulates a serial port connection.
- Service Discovery Protocol (SDP) discovers other Bluetooth devices within the range and establishes connections with them.
- Telephony Control Protocol Specification (TCS) provides telephony services.
- Audio provides services to carry audio signals.

As illustrated in Figure 2.3, other higher, non-Bluetooth communication protocols such as OBEX (Object Exchange), TCP/IP and WAP can be built on top of the Bluetooth protocol stack.

Figure 2.4 shows the familiar Open System Interconnection (OSI) standard reference model for communications protocol stacks. Although Bluetooth does not exactly match the model, it is a useful exercise to relate the different parts of the Bluetooth stack to the various parts of the model. Since the reference model is an ideal, well-partitioned stack, the comparison serves to highlight the division of responsibility in the Bluetooth stack.

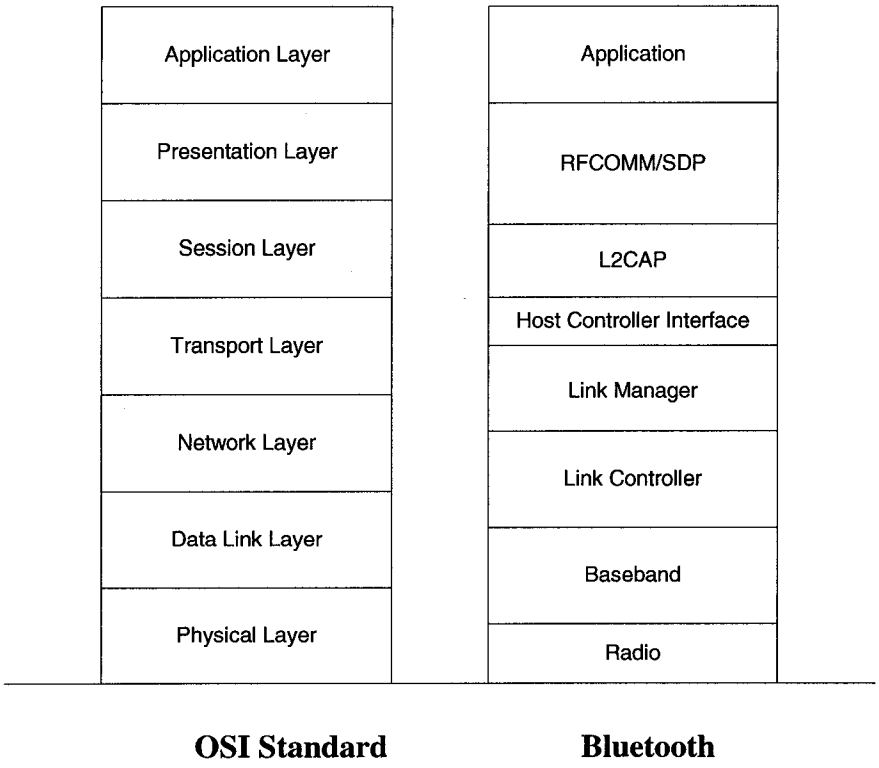

Figure 2.4 Comparing OSI reference model and Bluetooth protocol stack

The Physical Layer is responsible for the electrical interface to the communications media, including modulation and channel coding. It thus covers the radio and part of the baseband.

The Data Link Layer is responsible for transmission, framing and error control over a particular link, and as such, overlaps the link controller task and the control end of the baseband, including error checking and correction.

Starting from the Network Layer, the one-to-one correspondence between the OSI standard layer and the Bluetooth stack ceases to exist. The Network Layer is responsible for data transfer across the network, independent of the media and specific topology of the network. This encompasses the higher end of the link controller, setting up and maintaining multiple links, and also covers most of the Link Manager (LM) task. The Transport Layer is responsible for the reliability and multiplexing of data transfer across the network to the level provided by the application, and thus overlaps at the high end of the LM and covers the Host Controller Interface (HCI), which provides the actual data transport mechanisms.

The Session Layer provides the management and data flow control services, which are covered by L2CAP and the lower ends of RFCOMM/SDP. The Presentation Layer provides a common representation for Application Layer data by adding service structure to the units of data, which is the main task of RFCOMM/SDP. Finally, the Application Layer is responsible for managing communications between host applications.

3. Basic Concepts of Cryptography

Almost all of the modern digital information security measures depend heavily on the science of cryptography. Bluetooth is no exception. In order for the reader to better understand the security implementations of Bluetooth at both the service level and the link level, the author will introduce and explain some basic concepts of cryptography.

3.1 The Science of Cryptography:

Cryptography is the study of mathematical techniques related to aspects of information security such as confidentiality, data integrity, entity authentication, and data origin authentication [4].

Less formally, the fundamental objective of cryptography is to enable two people, usually referred to as Alice and Bob, to communicate over an insecure channel in such a way that an opponent, Oscar, cannot understand what is being said. This channel could be a telephone line, a computer network, or, in the case of Bluetooth, a radio channel. The information that Alice wants to send to Bob, the "plaintext", can be English text, numerical data, or something else. Alice encrypts the plaintext, using a predetermined key, and sends the resulting ciphertext over the channel. Oscar, upon seeing the ciphertext in the channel by eavesdropping, cannot determine what the plaintext is; but Bob, who knows the encryption key, can decrypt the ciphertext and reconstruct the plaintext [13].

Cryptography has a long and fascinating history. The Egyptians started to make limited use of cryptography as early as 4,000 years ago. In the twentieth century, cryptography played a crucial role in the outcome of both world wars. It was heavily employed by the military, diplomatic service and government as a tool to protect national secrets. In the 1960s the proliferation of computers and communication systems brought demand for means to protect information in digital form and to provide security services. DES (Data Encryption Standard),

which was invented by IBM in the early 1970s and later adopted by the U.S. government in 1977, is arguably the most well-known cryptographic mechanism in history [13]. It remains the standard means for securing electronic commerce for many financial institutions around the world. Recently, the U.S. government has been seeking a newer and improved standard, AES (Advanced Encryption Standard), to replace DES. Many different encryption algorithms are competing against each other to become the AES. The algorithm employed by Bluetooth, SAFER+, is one of those leading contenders [14].

The most striking development in the history of cryptography came in 1976 when Diffie and Hellman published *New Directions in Cryptography* [15]. This paper introduced the revolutionary concept of public-key cryptography and also provided a new and ingenious method for key exchange, the security of which is based on the intractability of the discrete logarithm problem. Although Diffie and Hellman had no practical realization of a public-key encryption scheme at the time, the idea was clear and it generated extensive interest and activity in the cryptographic community. In 1978 Rivest, Shamir and Adleman discovered the first practical public-key encryption and signature scheme, now referred to as RSA [16]. The RSA scheme is based on another hard mathematical problem, the intractability of factoring large integers. This application of a hard mathematical problem to cryptography revitalized efforts to find more efficient methods to factor. The 1980s saw major advances in this area but none of which rendered the RSA system insecure. Another class of powerful and practical public-key scheme was found by ElGamal in 1985 [17]. These are also based on the discrete logarithm problem.

The search for new public-key schemes, improvements to existing cryptographic mechanisms, and proofs of security continues at a rapid pace. Various standards and infrastructures involving cryptography are being put in place. Security products are being developed to address the security needs for an information intensive society.

3.2 Encryption Scheme and Key Pair:

An encryption scheme E is a mathematical function that uses the plaintext M and the encryption key K_e as inputs to produce the ciphertext C. C is transmitted to the intended recipient. With C and K_d as inputs, K_d being the corresponding decryption key to K_e, the recipient can use the decryption scheme D to reconstruct the original plaintext M.

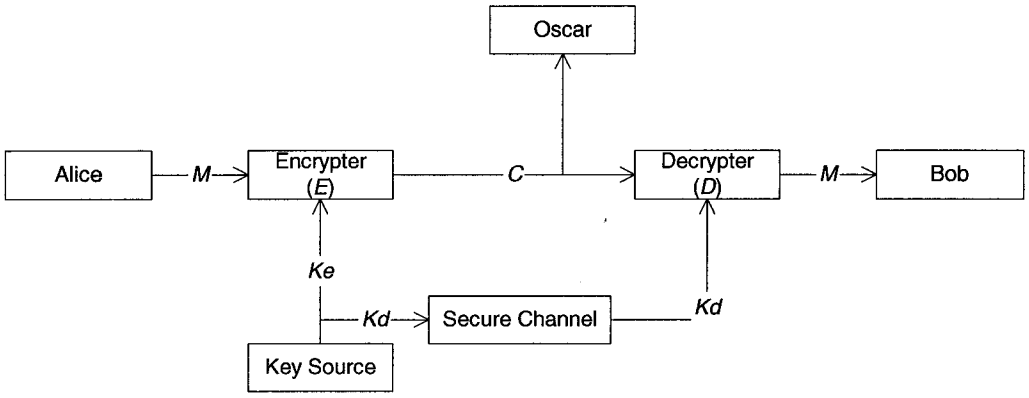

Figure 3.1 Encryption scheme and Key Pair.

E must be a bijection if the process is to be reversed by D and a unique plaintext message M recovered by each distinct ciphertext C. (Note: a function is a bijection if no two values in the domain map to the same value in the codomain, and for every element in the codomain there is some element in the domain that maps to it.)

One may ask why a key pair K_e and K_d are necessary, i.e. why not just choose one encryption function and its corresponding decryption function? The answer is that having key pairs means that if some particular encryption and/or decryption transformation is revealed then one does not have to redesign the entire scheme, but simply change the key pair. It is sound cryptographic practice to change the key pairs frequently. As a physical analog, consider an ordinary resettable combination lock. The structure of the lock is available to anyone who wishes to purchase one but the combination is chosen and set by the owner. If the

owner suspects that the combination has been revealed he can easily reset it without replacing the physical mechanism.

3.3 Stream Ciphering:

Bluetooth security implementation employs the stream ciphering technique extensively. The basic idea of stream ciphering is to generate a keystream $z = z_1z_2\ldots$, and use it to encrypt a plaintext string $x = x_1x_2\ldots$ according to the rule

$$y = y_1y_2\ldots = e_{z1}(x_1)e_{z2}(x_2)\ldots$$

A stream cipher operates as follows. Suppose K is the key and $x_1x_2\ldots$ is the plaintext string. The function f_i is used to generate z_i (the ith element of the keystream), where f_i is a function of the key, K, and the first i-1 plaintext characters:

$$z_i = f_i\,(K, x_1, \ldots, x_{i-1}).$$

The keystream element z_i is used to encrypt x_i, yielding $y_i = e_{zi}(x_i)$. So, to encrypt the plaintext string $x_1x_2\ldots$, we should successively compute

$$z_1, y_1, z_2, y_2, \ldots$$

Decrypting the ciphertext string $y_1y_2\ldots$ can be accomplished by successively computing

$$z_1, x_1, z_2, x_2, \ldots$$

The keystream used in stream ciphering can be produced efficiently in hardware using a linear feedback shift register, or LFSR [13].

3.4 Authentication and Challenge-Response Protocol:

Authentication refers to the technique that allows one party (the verifier) to gain assurance that the identity of another party (the claimant) is as declared, thereby preventing impersonation. The most common way to achieve this goal is through the challenge-response protocol.

The idea of challenge-response protocol is that the claimant "proves" its identity to the verifier by demonstrating knowledge of a secret known to be associated with the genuine claimant, without revealing the secret itself to the verifier during the protocol. This is done by providing a response to a time-variant challenge, where the response depends on both the claimant's secret and the verifier's challenge. The challenge is typically a number chosen by the verifier randomly and secretly at the outset of the protocol. The response from one execution of the identification protocol should not provide an adversary with useful information for a subsequent identification, as subsequent challenges will differ [13].

The challenge-response authentication protocols can be based on symmetric-key techniques, public-key techniques, and zero-knowledge concepts. Bluetooth uses the symmetric-key challenge-response protocol for authentication purpose. It will be discussed in details in section 5.4.

3.5 Diffie-Hellman Key Agreement:

As illustrated in Figure 3.1, in a private-key system, the secret key must be exchanged in a secure channel. However, in practice, this secure channel is hard to establish. In fact, key exchange and key agreement are one of the biggest problems in cryptography. Many Bluetooth devices use a well-known algorithm called Diffie-Hellman to exchange keys.

The specific details of the Diffie-Hellman key exchange scheme are out of the scope of this paper. For more details, interested readers can refer to [15]. All

the reader needs to know is that the security of Diffie-Hellman scheme relies on the assumption that discrete logarithm problems are intractable.

Note that there is no mathematical proof so far that discrete logarithm problems cannot be solved in polynomial times, however, for all practical purposes, this kind of problems are deemed extremely hard to solve, i.e., it is assumed to be NP-complete. Similar kind of assumption applies to the factorization of large integers problem. Most modern cryptography methods rely on either one of these two assumptions. If one day someone can find a way to solve either the discrete logarithm problem or the large integer factorization problem in polynomial times, he or she will probably cause the world to shake.

3.6 Random Number Generator and PRBG:

The security of many cryptographic systems depends on the generation of unpredictable quantities. For example, the private keys K_e and K_d used in Figure 3.1 can be random numbers. The term "random" means that the probability of any particular value being selected must be sufficiently small to preclude an adversary from gaining advantage through optimizing a search strategy based on such probability. For example, suppose K_d used in Figure 3.1 is 56-bit long. If K_d was selected using a true random generator, Oscar the adversary would on average have to try 2^{55} possible keys before guessing the correct key K. If, on the other hand, a key K were selected by first choosing a 16-bit random secret *s*, and then expanding it into a 56-bit key K using a complicated but publicly known function *f*, the adversary would on average only need to try 2^{15} possible keys (obtained by running every possible value for s through the function *f*).

A true random bit generator requires a naturally occurring source of randomness, for example, coins tossing [4]. However, this kind of physical process is usually time-consuming and expensive, so in practice it is common to use a pseudo-random bit generator (PRBG). A PRBG starts with a short random bit-string that serves as a "seed" and expands it into a much longer "random-looking" bit-string. Thus a PRBG reduces the amount of random bits that are required in an application. The pseudo-random number generated by a PRBG

must be sufficiently secure, that is, it should be impossible in a polynomial-time to distinguish a string of pseudo-random bits from a string of true random bits [13]. Bluetooth uses a pseudo-random number generator for its security implementations.

4. Service Level Security Implementation in Bluetooth

In Bluetooth technology, three security levels are defined [18]:

- Security Mode 1: non-secure
- Security Mode 2: service level enforced security
- Security Mode 3: link level enforced security

In security mode 1 a device will not initiate any security - this is the non-secure mode. The essential difference between security mode 2 and security mode 3 is that in security mode 2 the Bluetooth device initiates security procedures after the channel is established (at the higher layers), while in security mode 3 the Bluetooth device initiates security procedures before the channel is established (at the lower layers).

The research interest of this paper is primarily on the link level enforced security, which will be examined in great details in section 5. This section, however, will concisely discuss the security measures taken at the service level in order to complete the picture.

4.1 Security Mode 2 Settings:

Two possibilities exist for a device's access to services. It is either considered as a "trusted device" or an "untrusted device". The trusted device has unrestricted access to all services. The untrusted device does not have fixed relationships and its access to services is limited [19].

For services, three security levels are defined: [20]

- Services that require authorization and authentication. Automatic access is only granted to trusted devices. Other devices need a manual authorization.

- Services that require authentication only, authorization is not necessary.
- Services that are open to all devices. Neither authorization nor authentication is required, no access approval is required before service access is granted.

Note: one possible refinement is to set the trust level of a device specifically for a service or a group of services. The interaction with the remote device does not exclude the implementation of such refined access policies.

4.2 Flexibility and Usability Requirements:

For a successful security system, flexibility and usability are top priorities.

It should be possible to grant access to some services without providing access to other services: for example, on a cellular phone, service discovery records should be generally accessible, whereas dialup networking should only be available for specific devices.

Also, as a general principle, user intervention for access to services should be avoided as much as possible. It should only be needed to allow devices limited access to services or to set up a trusted relationship with devices allowing unlimited access to services [18].

4.3 Implementation and Bluetooth-specific Matters:

The security architecture discussed in section 4 accounts for security at and above L2CAP. Below L2CAP security mode 3 (link-level security) is used. There are also proprietary securities used in some of the adopted standards (e.g. PPP) which exist above the L2CAP layer.

It is important to remember that this security architecture allows different protocols to enforce the security policies. For example, L2CAP will enforce the Bluetooth security policy for cordless telephony, RFCOMM will enforce the

Bluetooth security policy for dialup networking, and OBEX will use its own security policy for file transfer and synchronization.

The enforcement policy for authentication, authorization or encryption might be different for the client and for the server. Thus the security level of peer entities running an application needs not be symmetric. A server may conceivably and quite probably have a higher security level requirement than that of a client [18].

4.4 Security Architecture:

Figure 4.1 gives the general architecture of the Bluetooth security.

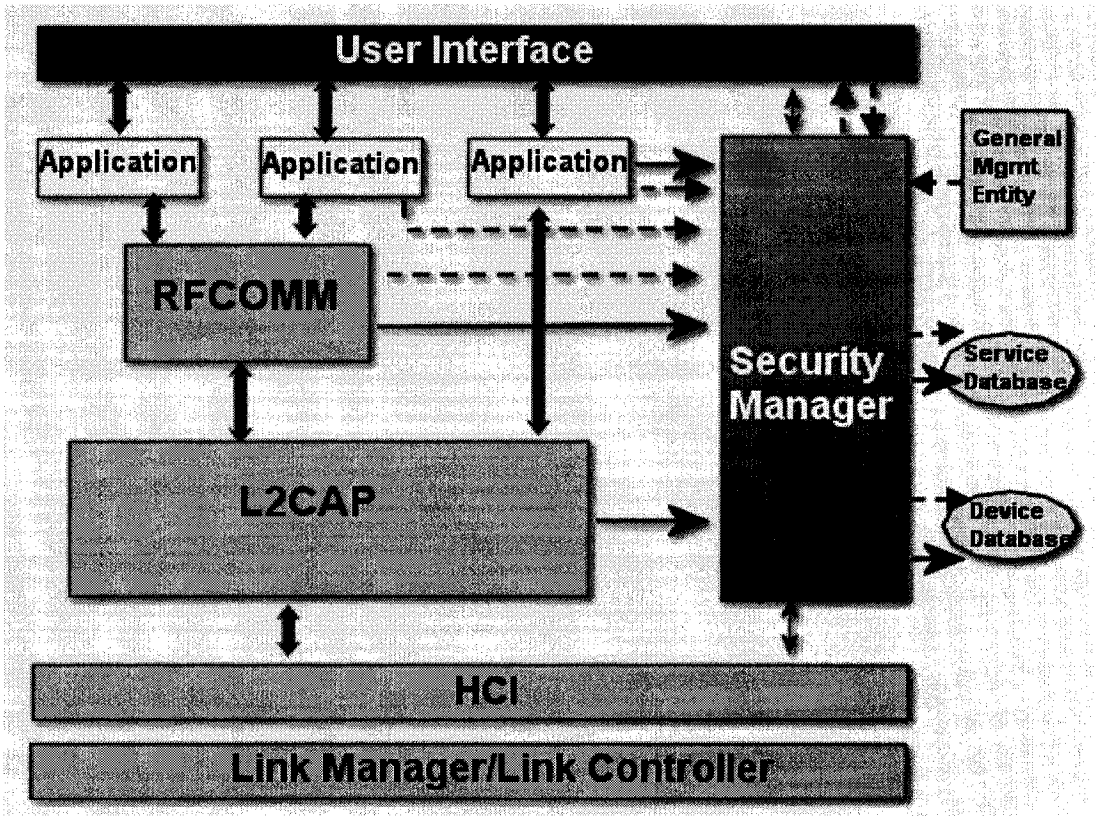

Figure 4.1 Bluetooth security architecture [18]

As can be seen, the key component is the security manager. It has several key tasks, including:

- Store security-related information on services & devices
- Answer access requests by protocol implementations or applications (either access granted or access refused)
- Enforce authentication and/or encryption before connecting to the application.
- Initiate or process input from the device user to set-up trusted relationships on device level.
- Initiate pairing and query PIN entry by the user. Note: PIN entry might also be done by an application.

The security architecture presented here provides a very flexible security framework. This framework dictates when to involve a user (e.g., to provide a PIN) and what actions the underlying Bluetooth protocol layers take to support the desired security check-ups. Within this framework, a number of realizations of the presented architecture can be instantiated, some of them simpler and some of them more advanced, without moving outside the scope of the architecture.

The approach with a centralized security manager allows easy implementation of flexible access policies, because the interfaces to protocols and other entities are kept simple and are limited to query/response and registration procedures. The policies for access control are encapsulated in the security manager. Therefore, implementation of more complex policies would not affect implementation of other parts.

4.5 Security Implementation Within the Architecture:

The three device trust levels are handled as the following:

- **Trusted Device**: The device has been previously authenticated, a link key is stored and the device is marked as "trusted" in the device database.
- **Untrusted Device**: The device has been previously authenticated, a link key is stored, but the device is not marked as "trusted" in the device database.

- **Unknown Device**: No security information is available for this device. This is also an untrusted device.

This information is stored in the device database table maintained in the security manager [18].

There are also three security levels for the services:

- **Authorization Required**: Access is granted to an untrusted or unknown device only after an authorization procedure. The authorization procedure always requires authentication to verify that the remote device is the genuine one. For a trusted device, however, access is automatically granted without any authorization.
- **Authentication Required**: Before connecting to the application, the remote device must be authenticated.
- **Encryption Required**: The link must be changed to the encrypted mode before access to the service is possible.

This information is stored in the service database of the security manager [18].

5. Link Level Security Implementation in Bluetooth

Security measures are provided at both the application layer and the link layer for usage protection and information confidentiality. In each Bluetooth unit, the authentication and encryption routines are implemented the same way. This makes it possible to establish secure communication channels between different Bluetooth units.

Four different entities are used for maintaining security at the link level: BD_ADDR (48 bits), private authentication user key (128 bits), private encryption user key (configurable between 8 bits and 128 bits) and RAND (128 bits) [21].

BD_ADDR is the unique Bluetooth device address. It is public information. A human user can easily obtain it via an MMI, or another Bluetooth unit can obtain it by a simple inquiry.

The authentication key size is always 128 bits. The encryption key size, however, is configurable. It can be as short as 8 bits, or as long as 128 bits. A major reason for the configurable encryption key size has to do with the regulations in different countries. While there are few governmental regulations against the strength of authentication, in some countries there are requirements imposed upon the strength of encryption. Making the encryption key size configurable can help manufacturers and vendors meet these requirements. Also, the simplest way to combat increasing computing power on the attacker's side is to increase the effective key size, this is another reason why the encryption key size can be configured. Note that the encryption key size of a specific Bluetooth unit can only be preset in the factory, and the end user cannot change it.

The secret keys are derived during initialization and are never disclosed further. Normally, the encryption key is derived from the authentication key during the authentication process. However, the encryption key is entirely different from the authentication key (even though the latter is used when creating the former, as is described in section 5.5.4). Each time encryption is activated, a

new encryption key shall be generated. Thus, the lifetime of the encryption key does not necessarily correspond to the lifetime of the authentication key. More likely than not, the authentication key will be more static in its nature than the encryption key. To underline the fundamental importance of the authentication key to a specific Bluetooth link, it will often be referred to as the link key. Once a link is established between two or more Bluetooth devices, the particular application running on the Bluetooth devices will decide if and when to change the authentication key.

The RAND is a random number that is derived from a random or pseudo-random process in the Bluetooth unit. It is not a static parameter, and it will change for each new transaction. Many security functions need to use random numbers. For instance, the challenge-response scheme, the generation of authentication keys, the generation of encryption keys, etc, all of them need random numbers as part of the inputs. For this reason, each Bluetooth unit has a random number generator.

5.1 The RAND:

Ideally, a true random generator based on some physical process with inherent randomness is used. Examples of such processes are thermal noise from a semiconductor or resistor and the frequency instability of a free running oscillator. In reality, a software-based solution with a pseudo-random generator is usually preferred due to its ease of implementation. For all practical purposes, it is quite difficult to differentiate a true random number and a pseudo-random number [13].

Within Bluetooth, the requirements placed on the random numbers used are that they be non-repeating and randomly generated [5]. The expression "non-repeating" means that it shall be highly unlikely that the value should repeat itself within the lifetime of the authentication key. For example, a non-repeating value could be the output of a counter that is unlikely to repeat during the lifetime of the authentication key, or a date/time stamp. The expression "randomly generated"

means that it shall not be possible to predict its value with a chance that is significantly larger than 0 (e.g., greater than $1/2^L$ for a key length of L bits).

5.2 Key Management:

Once preset in the factory, the encryption key size of a specific Bluetooth unit cannot be changed by the end user. In order to prevent the user from overriding the permitted key size, the Bluetooth baseband processing does not accept an encryption key given from higher software layers. There are specific procedures implemented at the baseband level to create new encryption keys, as well as to modify existing link keys.

5.2.1 Types of keys

The link keys can be either semi-permanent or temporary. A semi-permanent link key is stored in non-volatile memory and may be used after the current session is terminated [21]. A session is defined as the time interval for which the unit is a member of a particular piconet. Thus, the session terminates when the unit disconnects from the piconet. Consequently, once a semi-permanent link key is defined, it may be used in the authentication of several subsequent connections between the Bluetooth units sharing it.

A temporary key can only be used during the current session. In a point-to-multipoint broadcasting situation, a common encryption key is useful. To achieve this, a special link key can temporarily replace the current link key.

In order to accommodate for different types of applications, four types of link keys have been defined [20]:

- the combination key K_{AB}
- the unit key K_A
- the temporary key K_{master}
- the initialization key K_{init}

In addition to these keys there is an encryption key, denoted K_c. This key is derived from the current link key.

Functionally speaking, K_{AB} and K_A are indistinguishable; the only difference is how they are generated. The unit key K_A is generated in, and therefore dependent on, a single unit A. The combination key K_{AB} is dynamic. It is derived from information in both units A and B, and is therefore always dependent on two units. The unit key is generated once at installation of the Bluetooth unit, therefore, it is rarely changed. The combination key is derived for each new combination of two Bluetooth units.

When the master wants to broadcast a message to several slaves simultaneously, K_{master} will be used to replace the original link key. After the broadcasting, K_{master} will be discarded, and the original link key will be used again.

The initialization key, K_{init}, is used as the link key during the initialization process before any combination or unit keys have been defined and exchanged yet. The initialization key protects the transfer of initialization parameters. The key is derived from a random number, an L-octet PIN code, and a BD_ADDR. Other than during the initialization phase, K_{init} is also used when a link key is lost and a new link key must be established.

When two Bluetooth units are to be "bonded", meaning that secure communication channels are to be established between them, they must first share the same PIN. The PIN can be a fixed number that is built into the Bluetooth unit by the manufacturer, or it can be selected arbitrarily by the user and entered in the units that are to be "bonded" [6]. The latter procedure is used when both units have an MMI, for example a phone and a laptop. Entering a PIN in both units is more secure than using a fixed PIN in one of the units, and should be used whenever possible. Zero is the default value if the PIN has not been defined.

The PIN code can be chosen to be any length from 1 to 16 octets. Obviously, the longer the PIN, the stronger the security. However, a long PIN will be inconvenient for the end user to enter and remember if the PIN is not a built-in code. The user needs to make a trade-off between security and convenience depending on the situation. For the long PIN's, it is usually envisioned that the units exchange PIN codes not through mechanical (i.e. human) interaction, but

rather through means supported by software at the application layer. For example, this can be a Diffie-Hellman key agreement [15], where the exchanged key is passed on to the K_{init} generation process in both units, just as in the case of shorter PIN code.

5.2.2 Generation and initialization of keys

During the initialization phase, the link keys must be generated and distributed among the Bluetooth units. This is necessary because the authentication process that will take place later needs this information.

All initialization procedures consist of the following steps:

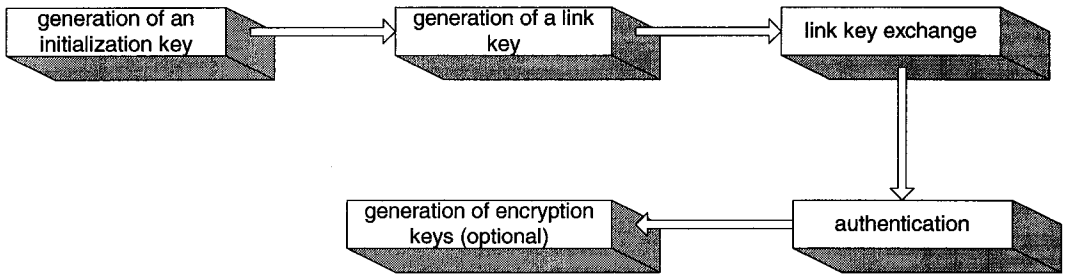

Figure 5.1 Steps in the initialization procedure.

The units basically have two choices after the initialization process: they can either proceed to communicate, or the link they just established can be disconnected. If encryption is implemented, the E_0 algorithm will be used with the proper encryption key derived from the current link key (For more details on E_0 algorithm, see section 5.3).

In a particular session between unit A and unit B, one and only one link key is needed, no matter how many connections there are between these devices, i.e. the initialization process does not have to be repeated for each new connection. A new encryption key derived from that particular link key will be created next time encryption is activated [5]. If for some reason the link key is lost, then the link manager shall automatically start a new initialization procedure.

5.2.2.1 Generating an initialization key

K_{init} is temporarily used during the initialization phase. This key is derived by the algorithm E_{22}. The inputs include a BD_ADDR, a PIN code, the length of the PIN (in octets), and a random number IN_RAND. The 128-bit output from E_{22} will be used for the key exchange during the generation of a link key. The algorithm E_{22} is illustrated in Figure 5.11. When the units have performed the link key exchange, the initialization key shall be discarded.

The PIN code used can be either a fixed one, which means it is built in by the manufacturer and cannot be changed by the user, or it can be a variable one, which means the user can change the PIN via a user interface on the device. The reason some devices have fixed PIN code is that they would not need a costly and bulky user interface to enter security information [6]. The Ultimate Headset mentioned in section 1.4 is a perfect example.

When the initialization key is generated, the PIN is augmented with the BD_ADDR. If one unit has a fixed PIN the BD_ADDR of the other unit is used. If both units have a variable PIN the BD_ADDR of the device that received IN_RAND is used. If both units have a fixed PIN they cannot be paired [5]. Since the maximum length of the PIN used in the algorithm cannot exceed 16 octets, it is possible that not all octets of BD_ADDR will be used. This procedure ensures that K_{init} depends on the identity of the unit with a variable PIN. A fraudulent Bluetooth unit may try to test a large number of PINs by each time claiming another BD_ADDR. It is the application's responsibility to take countermeasures against this threat. If the device address is kept fixed, the waiting interval until next try that is permitted is increased exponentially [20] (See section 5.4.1 for more details).

5.2.2.2 Authentication

Mutual authentication is achieved by first performing the authentication procedure in one direction and immediately followed by performing the authentication procedure in the opposite direction. The authentication procedure is

carried out as described in section 5.4. Note that during each authentication, a new AU_RAND_A is issued. The claimant/verifier status is determined by the link manager.

As a side effect of a successful authentication procedure an auxiliary parameter, the Authenticated Ciphering Offset (ACO), will be computed. The ACO is used for ciphering key generation as described in section 5.2.2.5.

5.2.2.3 Generating a unit key

The unit key K_A is generated in the Bluetooth unit by the E_{21} algorithm as described in section 5.5.3. It is generated when the Bluetooth unit is for the first time in operation, then the unit key is stored in non-volatile memory and almost never changed during the lifetime of the device. If after initialization the unit key is changed, the previously initialized units will possess a wrong link key. At initialization, the application has to determine which of the two parties will provide the unit key as link key [5]. Typically, this will be the unit with restricted memory capabilities, since this unit only has to remember its own unit key. The unit key is transferred to the other party and then stored as the link key for that particular party. So, for example in Figure 5.2, the unit key of unit A, K_A , is being used as the link key for the connection A-B; unit A sends the unit key K_A to unit B; unit B will store K_A as the link key K_{BA} . For another initialization, for example with unit C, unit A will reuse its unit key, whereas unit C stores it as K_{CA}.

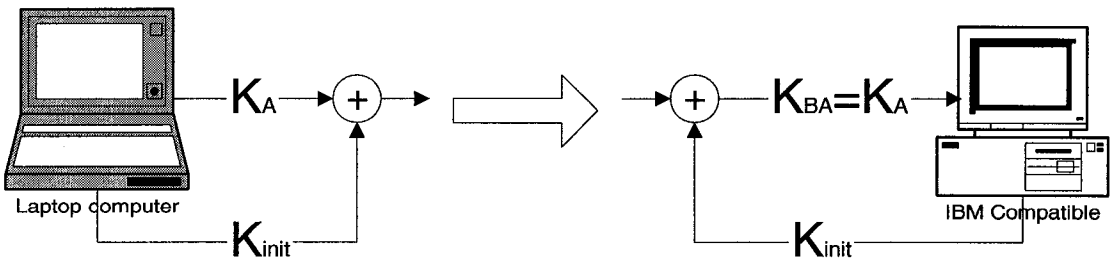

Figure 5.2: Exchange of the unit key. K_{init} is used to mask the unit key K_A before transmission. After the K_A has been successfully transmitted, K_{init} will be discarded.

5.2.2.4 Generating a combination key

The combination key is the combination of two numbers generated in unit A and B, respectively. First, each unit generates a random number, say LK_RAND_A and LK_RAND_B. Then, utilizing E_{21} with the random number and the own BD_ADDR, the two random numbers

$$LK\_K_A = E_{21}(LK\_RAND_A, BD\_ADDR_A), \quad \text{EQ1}$$

and

$$LK\_K_B = E_{21}(LK\_RAND_B, BD\_ADDR_B), \quad \text{EQ2}$$

are created in unit A and unit B, respectively (For more details on E_{21}, refer to section 5.5.3). These numbers constitute the units' contribution to the combination key that is to be created. Then, the two random numbers LK_RAND_A and LK_RAND_B are exchanged securely by XORing with the current link key, say K. Thus, unit A sends $K \oplus LK\_RAND_A$ to unit B, while unit B sends $K \oplus LK\_RAND_B$ to unit A. Clearly, if this is done during the initialization phase the link key $K = K_{init}$. When the random numbers LK_RAND_A and LK_RAND_B have been mutually exchanged, each unit recalculates the other unit's contribution to the combination key. This is possible since each unit knows the Bluetooth device address of the other unit. Thus, unit A calculates EQ2 and unit B calculates EQ1. After this, both units combine the two numbers to generate the 128-bit link key. The combining operation is a simple bitwise modulo-2 addition (i.e. XOR). The result is stored in unit A as the link key K_{AB} and in unit B as the link key K_{BA}. When both units have derived the new combination key, a mutual authentication procedure shall be initiated to confirm the success of the transaction. The old link key shall be discarded after a successful exchange of a new combination key [5]. The message flow between master and slave and the principle for creating the combination key is depicted in Figure 5.3.

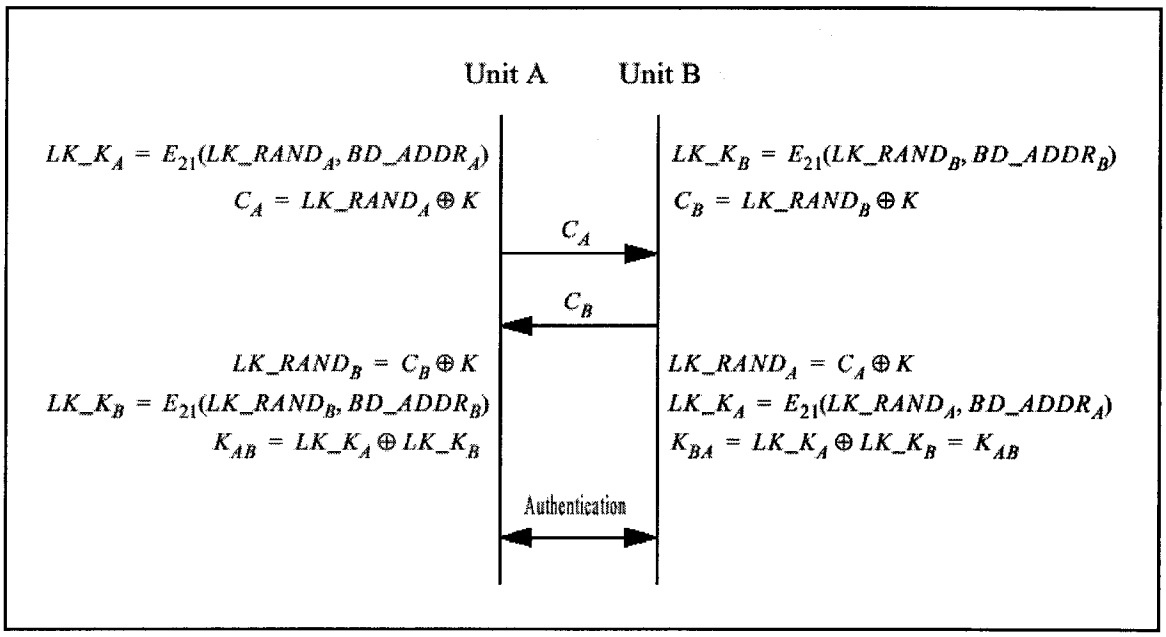

Figure 5.3: Generating and exchanging the combination key [5].

5.2.2.5 Generating the Encryption Key

The encryption key, K_C, is derived by algorithm E_3. The details of the key generating function E_3 can be found in section 5.5.4. The inputs include the current link key, a 96-bit Ciphering OFfset number (COF), and a 128-bit random number. The output is a 128-bit number.

The COF is determined in one of two ways. If the current link key is a master key, then COF = BD_ADDR ∪ BD_ADDR. Otherwise the value of COF is set to the value of ACO as computed during the authentication procedure. (Note: x ∪ y denotes the concatenation of the octet strings x and y).

There is an explicit call of E_3 when the link manager activates encryption. Consequently, the encryption key is automatically changed each time the unit enters the encryption mode.

5.2.2.6 Point-to-multipoint configuration

In a piconet, the master may talk to each slave using a separate encryption key. If the application requires more than one slave to listen to the same payload, each slave must be addressed individually. This may cause unwanted capacity loss for the piconet. Moreover, a Bluetooth unit (slave) is not capable of switching

between two or more encryption keys in real time. Thus, the master cannot use different encryption keys for broadcast messages and individually addressed traffic. As a solution to this problem, the slave units may use a common link key and, hence, indirectly also use a common encryption key [5]. The generation and exchange of the common link key are controlled by the master. For many applications, this common link key is only of temporary interest and it is denoted as K_{master}.

Section 5.2.2.8 describes the routine to protect the transfer of necessary parameters. After the confirmation of successful reception in each slave, the master shall issue a command to the slaves to temporarily discard their current link keys and start using the master key. To activate encryption, the master also generates and distributes a common EN_RAND to all participating slaves. Using this random number and the newly derived master key, each slave generates a new encryption key.

Note that the master must negotiate what encryption key length to use individually with each slave who wants to use the master key. In case the master already has negotiated with some of these slaves, it has knowledge of what sizes can be accepted. Clearly, there might be situations where the permitted key lengths of some units are incompatible. In that case, the master must have the limiting unit excluded from the group.

When all slaves have received the necessary data, the master can communicate information on the piconet securely using the encryption key derived from the new temporary link key [5]. Obviously, each slave in possession of the master key can eavesdrop on all encrypted traffic, not only the traffic intended for itself. If so desired, the master can tell all participants to fall back to their old link keys simultaneously.

5.2.2.7 Modifying Link Keys

The ability to modify the link keys is important in some circumstances. There are two types of link keys, K_A and K_{AB}, and the ways to modify them are different.

A link key based on a unit key can be changed, but not very easily. The unit key is created once during the first use. Changing the unit key is a less desirable alternative, as several units may share the unit key as the link key (think of a printer whose unit key is distributed among all users using the printer's unit key as link key). Changing the unit key will require re-initialization of all units trying to connect. In certain cases, this might be desirable; for example, to deny access to previously allowed units.

If the key change concerns combination keys, then the procedure is rather straightforward. The change procedure is identical to the procedure illustrated in Figure 5.3, using the current value of the combination key as link key. This procedure can be carried out at any time after the authentication and encryption start. In fact, since the combination key corresponds to a single link, it can be modified each time this link is established. This will improve the security of the system since the old keys lose their validity after each session.

5.2.2.8 Generating a master key

To generate a master key, first, the master Bluetooth unit creates a new link key from two 128-bit random numbers, RAND1 and RAND2. This is done by

$$K_{master} = E_{22}(RAND1, RAND2, 16) \qquad EQ3$$

This key is a 128-bit random number. The reason to use the output of E_{22} and not directly choose a random number as the key is to avoid possible problems with degraded randomness due to a poor implementation of the random number generator within the Bluetooth unit [5].

Then, a third random number, say RAND, is transmitted to the slave. Using E_{22} with the current link key and RAND as inputs, both the master and slave compute a 128-bit overlay. The master sends the bitwise XOR of the overlay and the new link key to the slave. The slave, who knows the overlay, recalculates K_{master}. To confirm the success of this transaction, the units shall perform a mutual authentication procedure using the new link key. The ACO values from the involved authentications should not replace the current existing

ACO as this ACO is needed to recompute a ciphering key when the master wants to fall back to the previous link (non-temporary) key.

When so required – and potentially long after the actual distribution of the master key – the master activates encryption by a link manager command. Before doing that, the master must ensure that all slaves receive the same random number, say, EN_RAND, since the encryption key is derived through the means of E_3 individually in all participating units. Then, each slave computes a new encryption key,

$$K_c = E_3(K_{master}, EN\_RAND, COF) \qquad EQ4$$

where the value of COF is derived from the master's BD_ADDR (COF = BD_ADDR $\cup$ BD_ADDR). The details on the encryption key generating function can be found in section 5.5.4. The principle of the message flow between the master and slave when generating the master key is depicted in Figure 5.4. Note that in this case the ACO produced during the authentication is not used when computing the ciphering key.

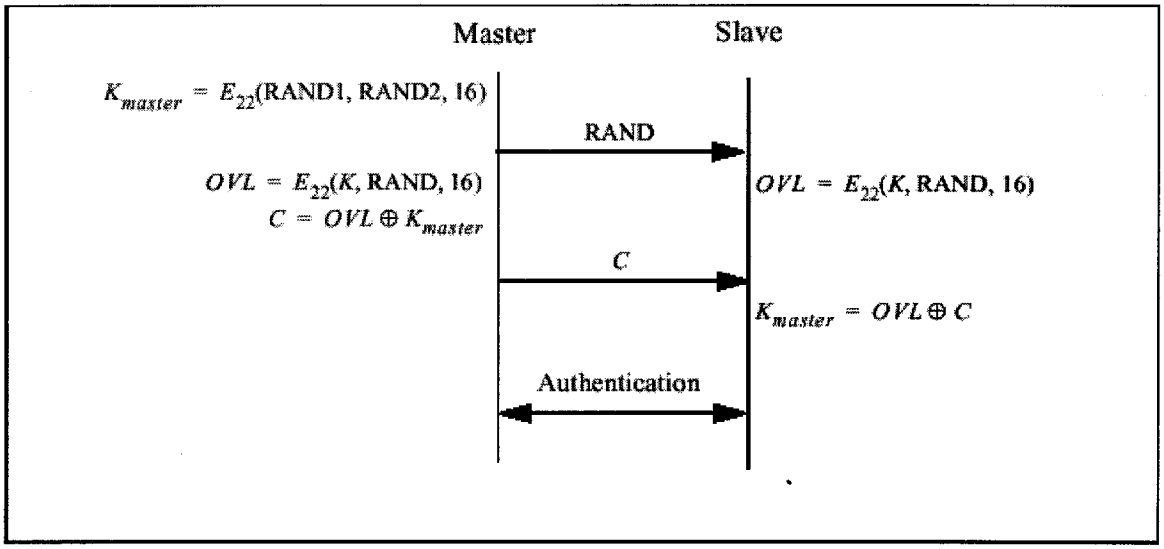

Figure 5.4: Master link key distribution and computation of the corresponding encryption key [5].

5.3 Encryption:

The encryption of the payloads is carried out with a stream cipher called E_0 that is re-synchronized for every payload. The overall principle is shown in Figure 5.5.

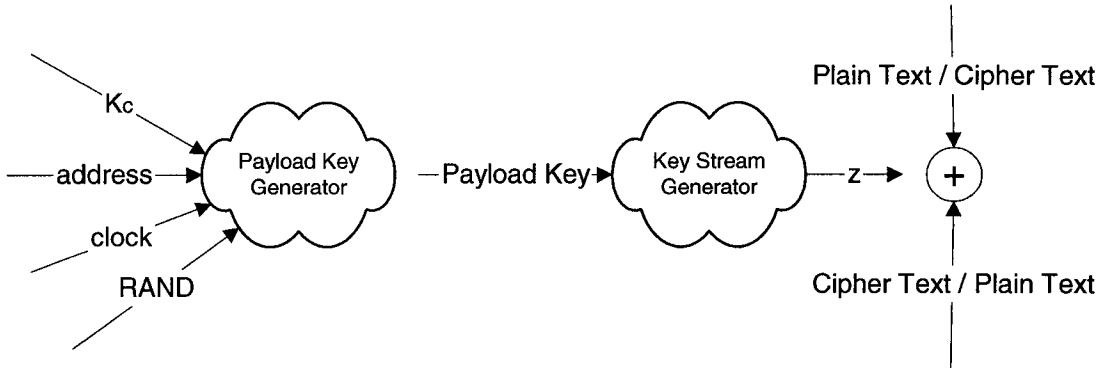

Figure 5.5: Three parts of E_0.

The stream cipher system E_0 consists of three parts. The first part performs the initialization, i.e. generation of the payload key; the second part generates the key stream bits; and the third part performs the encryption and decryption. The payload generator is very simple – it merely combines the input bits in an appropriate order and shift them into the four LFSRs used in the key stream generator. The main part of the cipher system is the second, as it also will be used for the initialization. The key stream bits are generated by a method derived from the summation stream cipher generator attributable to Massey and Rueppel [22]. The method has been thoroughly investigated, and there exists good estimates of its strength with respect to presently known methods for cryptanalysis. Although the summation generator has weaknesses that can be used in so-called correlation attacks, the high re-synchronization frequency will disrupt such attacks.

5.3.1 Encryption key size negotiation

Before generating the encryption key, the involved units must negotiate to decide what key size to actually use. Each Bluetooth device implementing the baseband specification needs a parameter defining the maximal allowed key length, L_{max}, $1 \leq L_{max} \leq 16$ (number of octets in the key). For each application, a number L_{min}, is defined indicating the smallest acceptable key size for that particular application.

The master sends a suggested value, $L^{(M)}_{sug}$, to the slave. Initially, the suggested value is set to $L^{(S)}_{min} \leq L^{(M)}_{sug} \leq L^{(M)}_{max}$. If so, and the slave supports the suggested length, the slave acknowledges and this value will be the length of the encryption key for this link. However, if both conditions are not fulfilled, the slave sends a new proposal, $L^{(S)}_{sug} \leq L^{(M)}_{sug}$ to the master. This value should be the largest among all supported lengths less than the previous master suggestion. Then, the master performs the corresponding test on the slave suggestion. This procedure is repeated until a key length agreement is reached, or, one unit aborts the negotiation. An abortion may be caused by lack of support for L_{sug} and all smaller key lengths, or if $L_{sug} < L_{min}$ in one of the units. In case of abortion Bluetooth link encryption cannot be employed.

The possibility of a failure in setting up a secure link is an unavoidable consequence of letting the application decide whether to accept or reject a suggested key size [5]. However, this is a necessary precaution. Otherwise a fraudulent unit could enforce a weak protection on a link by claiming a small maximum key size.

5.3.2 Encryption modes

If a slave has a semi-permanent link key (i.e. a combination key or a unit key), it can only accept encryption on slots individually addressed to itself (and, of course, in the reverse direction to the master). In particular, it will assume that broadcast messages are not encrypted. The possible traffic modes are described in table 5.1.

Broadcast Traffic	No Encryption	No Encryption
Individually Addressed Traffic	No Encryption	Encryption, semi-permanent link key

Table 5.1: Possible traffic modes if the slave does not have a master key.

Encrypting a broadcast message becomes possible with the introduction of the master key. If the slave has received a master key, there are three possible combinations as defined in table 5.2. In this case, all units in the piconet uses a common link key, K_{master}. Since the master uses encryption keys derived from this link key for all secure traffic on the piconet, it is possible to avoid ambiguity in the participating slaves on which encryption key to use. Also in this case the default mode is that broadcast messages are not encrypted. A specific link manager command is required to activate encryption – both for broadcast and for individually addressed traffic.

Broadcast Traffic	No Encryption	No Encryption	Encryption, Kmaster
Individually Addressed Traffic	No Encryption	Encryption, Kmaster	Encryption, Kmaster

Table 5.2: Possible encryption if the slave has a master key.

The master can issue a link manager command to the slaves telling them to fall back to their previous semi-permanent link key. Then, regardless of the previous mode they were in, they will end up in the first row of table 5.2; i.e. no encryption.

5.3.3 Encryption Concept

For the encryption routine, a stream cipher algorithm is used. The data stream to be sent over the air interface is XORed with the key stream. The payload is ciphered after the CRC bits are appended, but prior to the FEC encoding.

Each packet payload is ciphered separately. The cipher algorithm E_0 uses the master Bluetooth address, 26 bits of the master real-time clock (CLK_{26-1}) and

the encryption key K_c as input, see Figure 5.6 (where it is assumed that unit A is the master).

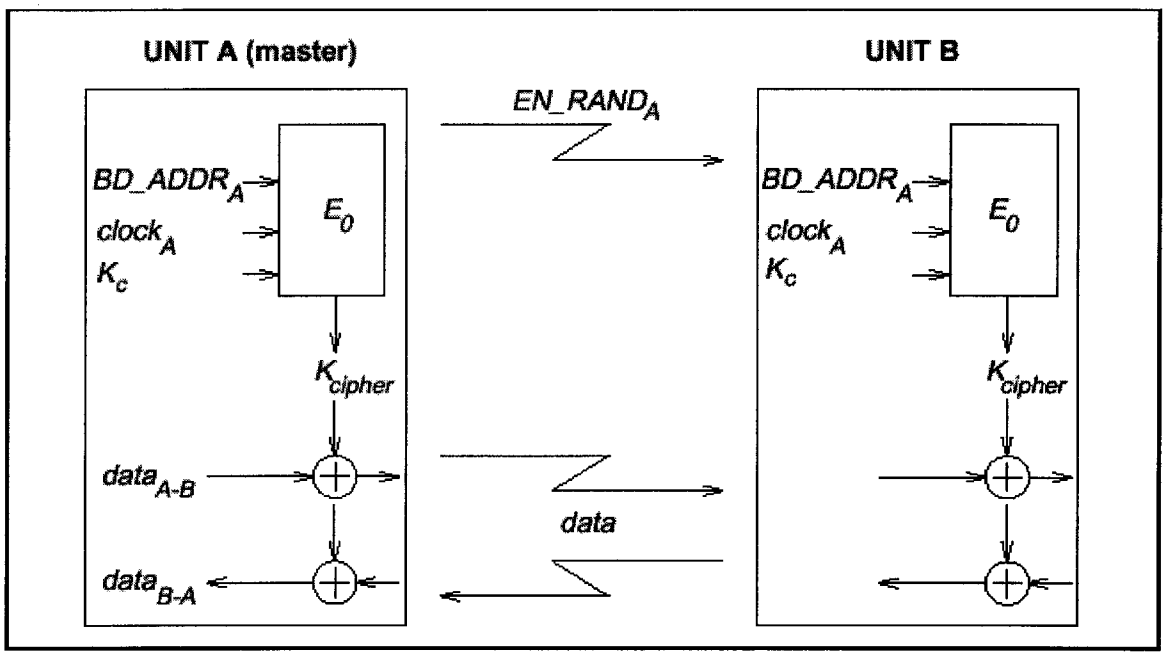

Figure 5.6: Functional description of the encryption procedure [5].

The real-time clock is incremented for each slot. The E_0 algorithm is re-initialized at the start of each new packet (i.e. for Master-to-Slave as well as for Slave-to-Master transmission). By using CLK_{26-1} at least one bit is changed between two transmissions. Thus, a new key stream is generated after each re-initialization. For packets covering more than a single slot, the Bluetooth clock as found in the first slot is being used for the entire packet.

The encryption key K_c is derived from the current link key, COF, and EN_RAND_A (see section 5.5.4). EN_RAND_A is a random number that is issued by the master before entering encryption mode. EN_RAND_A is publicly known since it is transmitted as plain text over the air.

The encryption algorithm E_0 generates a binary key stream, K_{cipher}, which is modulo-2 added to the data to be encrypted. The cipher is symmetric; decryption is performed in exactly the same way using the same key as used for encryption.

5.4 Authentication:

The entity authentication used in Bluetooth uses a challenge-response scheme in which a claimant's knowledge of a secret key is checked through a 2-move protocol using symmetric secret keys. The latter implies that a correct claimant/verifier pair shares the same secret key, for example K. In the challenge-response scheme the verifier challenges the claimant to authenticate a random input (the challenge), denoted by AU_RAND_A, with an authenticated code, denoted by E_1, and return the result SRES to the verifier, see Figure 5.7. This figure shows also that in Bluetooth the input to E_1 consists of the tuple AU_RAND_A and the Bluetooth device address (BD_ADDR) of the claimant. The secret key shared by units A and B is the current link key.

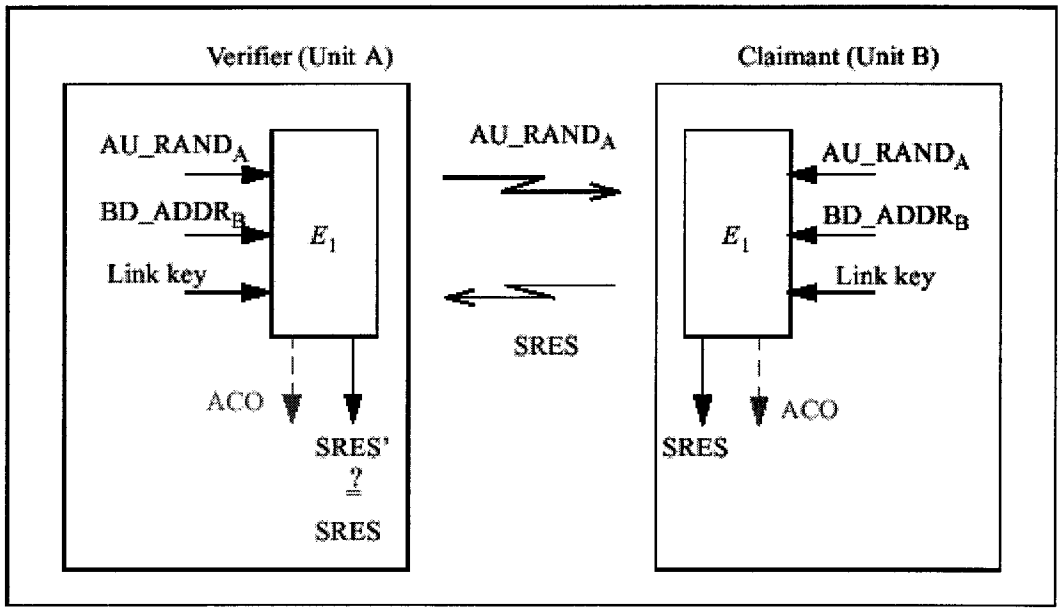

Figure 5.7: Challenge-response for the Bluetooth [5].

The challenge-response scheme for symmetric keys used in the Bluetooth is depicted in Figure 5.8.

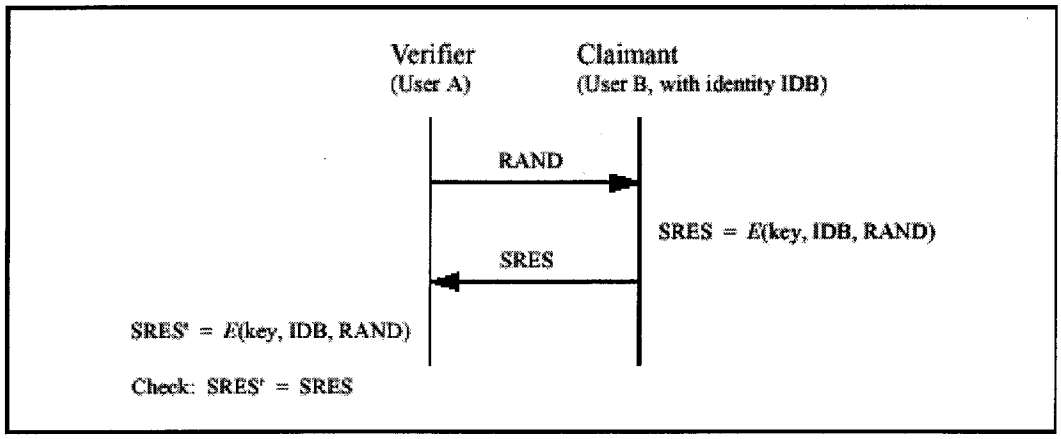

Figure 5.8: Challenge-response for symmetric key systems [5].

In the Bluetooth, the verifier is not necessarily the master. The application indicates who has to be authenticated by whom. Certain applications only require a one-way authentication. However, in some peer-to-peer communications, one might prefer a mutual authentication in which each unit is subsequently the challenger (verifier) in two authentication procedures. The link manager coordinates the indicated authentication preference by the application to determine in which direction the authentication has to take place. For mutual authentication with units of Figure 5.7, after unit A has successfully authenticated unit B, unit B could authenticate unit A by sending a AU_RAND_B (different from the AU_RAND_A that unit A issued) to unit A, and deriving the SRES and SRES' from the new AU_RAND_B, the address of unit A, and the link key K_{AB}.

If an authentication is successful the value of ACO as produced by E_1 should be retained.

5.4.1 Repeated attempts

When the authentication attempt fails, a certain waiting interval must pass before the verifier will initiate a new authentication attempt to the same claimant, or before it will respond to an authentication attempt initiated by a unit claiming the same identity as the suspicious unit. For each subsequent authentication failure with the same Bluetooth address, the waiting interval shall be increased exponentially. That is, after each failure, the waiting interval before a new attempt

can be made, for example, twice as long as the waiting interval prior to the previous attempt. The waiting interval shall be limited to a maximum. The maximum waiting interval depends on the implementation. The waiting time shall exponentially decrease to a minimum when no new failed attempts are being made during a certain time period. This procedure prevents an intruder to repeat the authentication procedure with a large number of different keys.

To make the system somewhat less vulnerable to denial-of-service attacks, the Bluetooth unit should keep a list of individual waiting intervals for each unit it has established contact with. Clearly, the size of this list must be restricted only to contain N units with which the most recent contact has been made. The number N can vary for different units depending on available memory size and user environment.

5.5 The Authentication and Key-Generating Functions:

This section describes the algorithmic means for supporting the Bluetooth security requirements on authentication and key generation.

5.5.1 The authentication function E_1

Bluetooth authentication function employs the SAFER+ algorithm. This algorithm is an enhanced version of an existing 64-bit block cipher SAFER-SK128, and it is freely available [23]. In the sequel the block cipher will be denoted as the function A_r which maps under a 128-bit key, a 128-bit input to a 128-bit output, i.e.

$$\begin{aligned} A_r&: \{0,1\}^{128} \times \{0,1\}^{128} \rightarrow \{0,1\}^{128} \\ &(k \times x) \mapsto t \end{aligned} \qquad \text{EQ5}$$

The details of A_r are given in the next section. The authentication function E_1 is constructed using A_r as follows

$$\begin{aligned} E_1:\ &\{0,1\}^{128} \times \{0,1\}^{128} \times \{0,1\}^{48} \rightarrow\ \{0,1\}^{32} \times \{0,1\}^{96} \\ &(K, RAND, address) \mapsto (SRES, ACO), \end{aligned} \qquad \text{EQ6}$$

where SRES = *Hash* (K, RAND, address, 6) [0, …, 3], where *Hash* is a keyed hash function defined as,

$$\textit{Hash}: \{0,1\}^{128} \times \{0,1\}^{128} \times \{0,1\}^{8\times L} \times \{6,12\} \rightarrow \{0,1\}^{128} \quad \text{EQ7}$$
$$(K, I_1, I_2, L) \mid\rightarrow A'_r([\acute{K}], [E(I_2, L) +16 (A_r(K, I_1) \oplus_{16} I_1)]),$$

(Note: The operator +16 denotes bytewise addition mod 256 of the 16 octets, and the operator $\oplus_{16}$ denotes bytewise XORing of the 16 octets).

and where

$$E: \{0,1\}^{8\times L} \times \{6,12\} \rightarrow \{0,1\}^{8\times 16} \quad \text{EQ8}$$
$$(X[0,\ldots,L\text{-}1], L) \mid\rightarrow (X[i(\text{mod } L)] \text{ for } i = 0\ldots15)$$

is an expansion of the L octet word X into a 128-bit word. Thus we see that we have to evaluate the function A_r twice for each evaluation of E_1. The key $\acute{K}$ for the second use of A_r (actually A'_r) is offsetted from K as follows

$\acute{K}[0] = (K[0] + 233) \text{ mod } 256$, $\acute{K}[1] = K[1] \oplus 229$,
$\acute{K}[2] = (K[2] + 233) \text{ mod } 256$, $\acute{K}[3] = K[3] \oplus 193$,
$\acute{K}[4] = (K[4] + 179) \text{ mod } 256$, $\acute{K}[5] = K[5] \oplus 167$,
$\acute{K}[6] = (K[6] + 149) \text{ mod } 256$, $\acute{K}[7] = K[7] \oplus 131$,
$\acute{K}[8] = K[8] \oplus 233$, $\acute{K}[9] = (K[9] + 229) \text{ mod } 256$,
$\acute{K}[10] = K[10] \oplus 233$, $\acute{K}[11] = (K[11] + 193) \text{ mod } 256$,
$\acute{K}[12] = K[12] \oplus 179$, $\acute{K}[13] = (K[13] + 167) \text{ mod } 256$,
$\acute{K}[14] = K[14] \oplus 149$, $\acute{K}[15] = (K[15] + 131) \text{ mod } 256$.

A data flowchart of the computation of E_1 is depicted in Figure 5.9. E_1 is also used to deliver the parameter ACO (Authenticated Ciphering Offset) that is used in the generation of the ciphering key by E_3. The value of ACO is formed by the octets 4 through 15 of the output of the hash function defined in (EQ 7), i.e.

$$\text{ACO} = \textit{Hash} \text{ (K, RAND, address, 6) [4, …, 15].} \quad \text{EQ9}$$

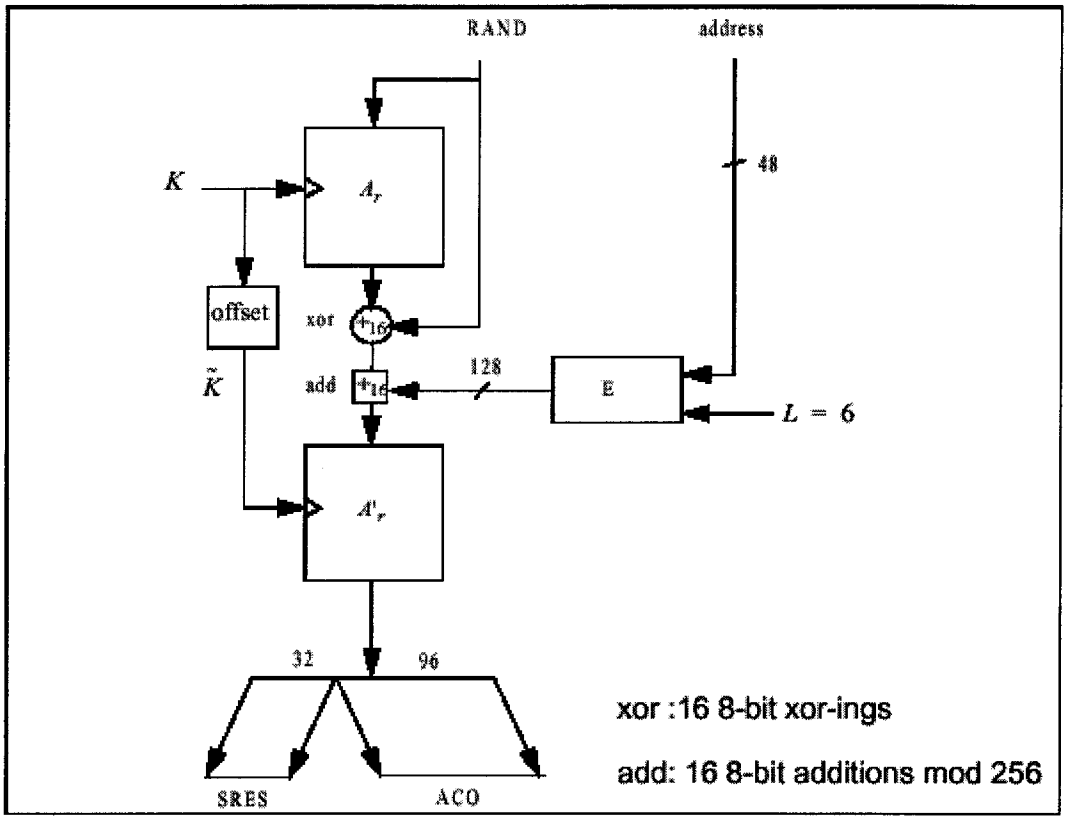

Figure 5.9: Flow of data for the computation of $E_1[5]$.

5.5.2 The functions A_r and A'_r

The function A_r is identical to SAFER+. It consists of a set of 8 layers, (each layer is called a round) and a parallel mechanism for generating the sub keys $K_p[j]$, p = 1, 2, …, 17, the so-called round keys to be used in each round. The function will produce a 128-bit result from a 128-bit "random" input string and a 128-bit "key". Besides the function A_r, a slightly modified version referred to as A'_r is used in which the input of round 1 is added to the input of the 3rd round. This is done to make the modified version non-invertible and prevents the use of A'_r (especially in E_{2x}) as an encryption function. See Figure 5.10 for details.

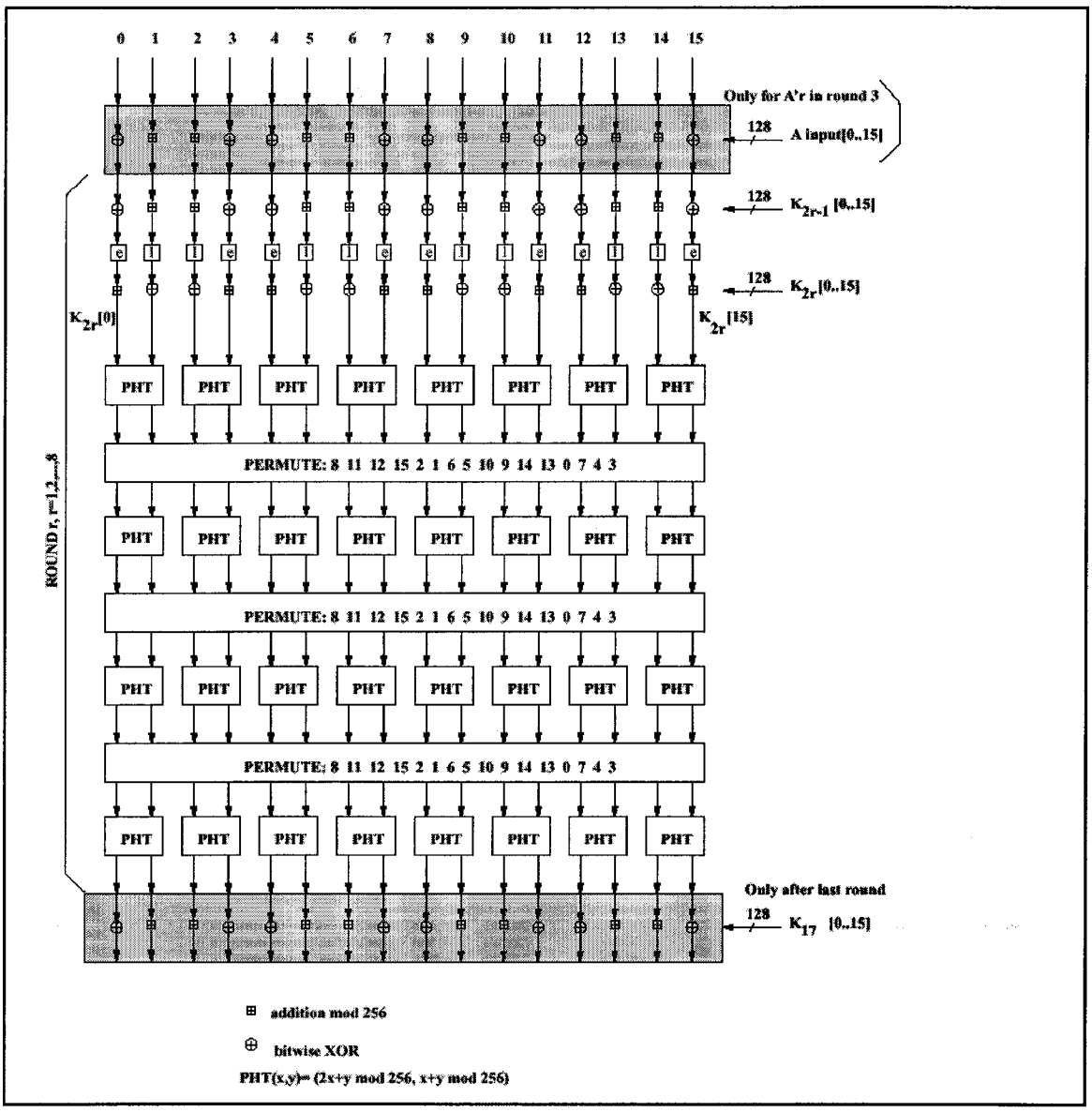

Figure 5.10: One round in A_r and A'_r. The permutation boxes show how input byte indices are mapped onto output byte indices. Thus, position 0 (leftmost) is mapped on position 8, position 1 is mapped on position 11, etc. [5].

5.5.2.1 The round computations

The computations in each round are a composition of encryption with a round key, substitution, encryption with the next round key, and, finally, a Pseudo Hadamard Transform (PHT). The computations in a round are shown in Figure 5.11. The sub keys for round r, r = 1, 2, …, 8 are denoted $K_{2r-1}[j]$, $K_{2r}[j]$, $j = 0, 1,$

.., 15. After the last round $K_{17}[j]$ is applied in a similar fashion as all previous odd numbered keys.

5.5.2.2 The substitution boxes "*e*" and "*l*"

In Figure 5.10 two boxes occur, marked "*e*" and "*l*". These boxes implement the same substitutions as used in SAFER+; i.e. they implement

e, l : $\{0, \ldots, 255\} \rightarrow \{0, \ldots, 255\}$,
e : $i \mapsto (45^i \pmod{257}) \pmod{256}$,
l : i $\mapsto$ j *s.t.* $i = e(j)$.

Their role, as in the SAFER+ algorithm, is to introduce non-linearity.

5.5.2.3 Key Scheduling

In each round, 2 batches of 16 octet-wide keys are needed. These so-called round keys are derived as specified by the key scheduling in SAFER+. Figure 5.11 gives an overview of how the round keys $K_p[j]$ are determined. The bias vectors $B_2, B_3, \ldots, B_{17}$ are computed according to the following equation:

$$p^{[i]} = ((45\text{^} (45^{17p+i+1} \bmod 257) \bmod 257) \bmod 256), \text{ for } i = 0, \ldots, 15 \quad \text{EQ10}$$

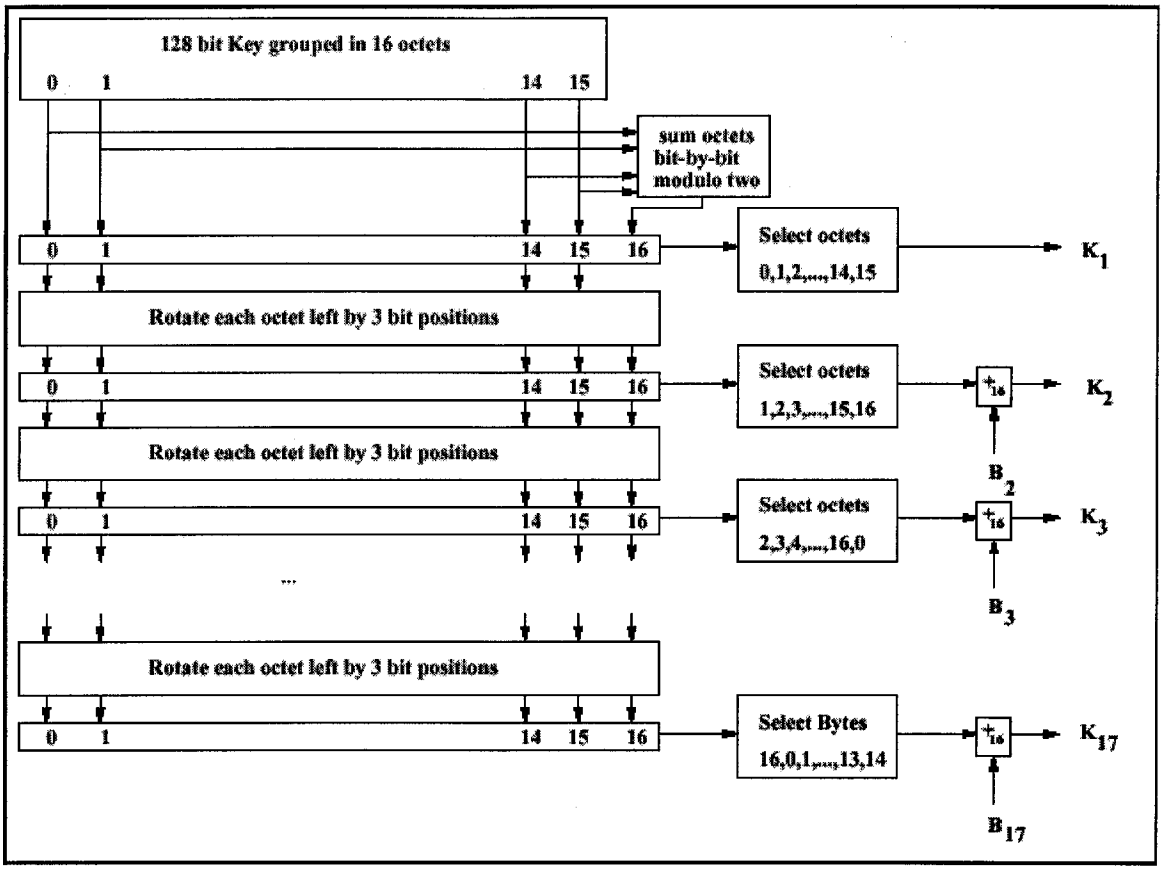

Figure 5.11: Key scheduling in A_r. [5].

5.5.3 E_2-Key generation function for authentication

The key used for authentication is derived through a procedure that is shown in Figure 5.12. The figure shows two different modes of operation for the algorithm. In the first mode, the function E_2 should produce a 128-bit link key K on input of a 128-bit RAND value and a 48-bit address. This mode is utilized when creating unit keys and combination keys. In the second mode the function E_2 should produce, on input of a 128-bit RAND value and an L octet user PIN, a 128-bit link key K. The second mode is used to create the initialization key, or to generate a master key whenever it is needed.

When the initialization key is generated, the PIN is augmented with BD_ADDR, see section 5.2.2.1 for which address to use. The augmentation always starts with the least significant octet of the address immediately following the most significant octet of the PIN. Since the maximum length of the PIN used

in the algorithm cannot exceed 16 octets, it is possible that not all octets of BD_ADDR will be used.

This key-generating algorithm again exploits the cryptographic function. Formally E_2 of mode 1 (denoted E_{21}) can be expressed as

$$E_{21}: \{0,1\}^{128} \times \{0,1\}^{48} \rightarrow \{0,1\}^{128} \quad \text{EQ11}$$
$$(\text{RAND, address}) \mid\rightarrow A'_r (X, Y)$$

where (for mode 1)

$$X = \text{RAND}[0 \ldots 14] \cup (\text{RAND}[15] \oplus 6) \quad \text{EQ12}$$
$$Y = \text{address}[0 \pmod 6] \cup \text{address}[1 \pmod 6] \cup \text{address}[2 \pmod 6] \cup \ldots\ldots \cup \text{address}[14 \pmod 6] \cup \text{address}[15 \pmod 6]$$

Let L be the number of octets in the user PIN. The augmenting is defined by

$$\text{PIN'} = \text{PIN}[0 \ldots L\text{-}1] \cup \text{BD\_ADDR}[0 \ldots \min\{5, 15\text{-}L\}], \quad L<16 \quad \text{EQ13}$$
$$\text{PIN'} = \text{PIN}[0 \ldots L\text{-}1], \quad L=16$$

Then, in mode 2, E_2 (denoted E_{22}) can be expressed as

$$E_{22}: \{0,1\}^{8L'} \times \{0,1\}^{128} \times \{1, 2, \ldots, 16\} \rightarrow \{0,1\}^{128} \quad \text{EQ14}$$
$$(\text{PIN', RAND, L'}) \mid\rightarrow A'_r(X, Y)$$

where

$$X = \text{PIN'}[0 \pmod{L'}] \cup \text{PIN'}[1 \pmod{L'}] \cup \text{PIN'}[2 \pmod{L'}] \cup \ldots\ldots\ldots \cup \text{PIN'}[14 \pmod{L'}] \cup \text{PIN'}[15 \pmod{L'}] \quad \text{EQ15}$$
$$Y = \text{RAND}[0 \ldots 14] \cup (\text{RAND}[15] \oplus L'),$$

And $L' = \min\{16, L+16\}$ is the number of octets in PIN'.

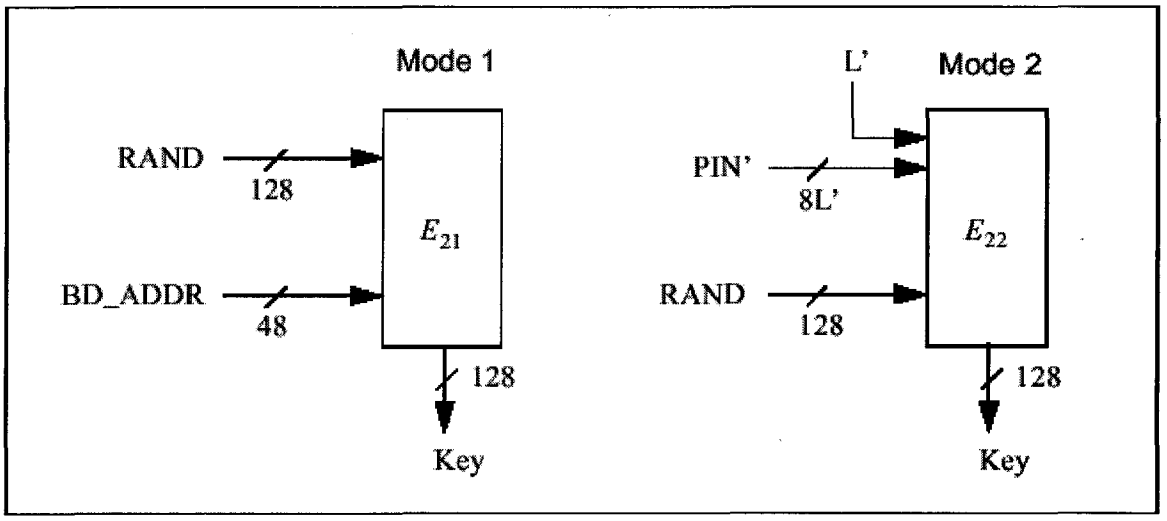

Figure 5.12: Key generating algorithm E_2 and its two modes. Mode 1 is used for unit and combination keys, while mode 2 is used for K_{init} and K_{master} [5].

5.5.4 E_3-Key generation function for encryption

The ciphering key K_c used by E_0 is generated by E_3. The function is constructed using A'_r as follows:

$$E_3: \{0,1\}^{128} \times \{0,1\}^{128} \times \{0,1\}^{96} \rightarrow \{0,1\}^{128} \qquad \text{EQ16}$$
$$(K, RAND, COF) \mapsto \textit{Hash}\,(K, RAND, COF, 12)$$

where *Hash* is the hash function as defined by EQ7. Note that the produced key length is 128 bits. However, before used by E_0, the encryption key K_c will be shortened to the correct encryption key length. A block scheme of E_3 is depicted in Figure 5.13.

The value of COF is determined as specified in section 5.2.2.5.

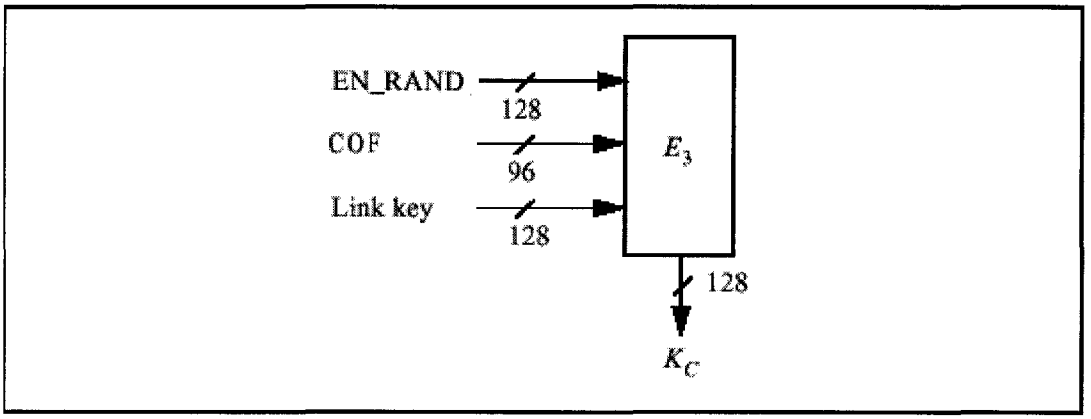

Figure 5.13 Generation of the encryption key [5].

6. Evaluation and Conclusion

As described in section 4 and 5, Bluetooth has a comprehensive set of security features at both the service level and the link level. However, Bluetooth does have some security problems.

6.1 Known Security Problems at the Link Level:

6.1.1 The PIN code problem

A short PIN code, like a four digit one, is likely to weaken the overall security. The PIN code is used in combination with other variables to generate the link key and the encryption key. In fact, it is the only truly secret key generation variable and the only one that is not transmitted over the air. Also, the fact that the same PIN has to be inputted in both Bluetooth devices to initialize a secure connection may cause a security problem. Obviously, some users will eventually become weary of this. They probably will either store the PIN within both devices, or use a number that is easy to remember, like "0000". If this is the case, the overall security of the Bluetooth devices will be put at risk.

6.1.2 Spoofing due to non-secret link key

Another problem arises with the use of the link key. Authentication and encryption are based on the assumption that the link key is the participants' shared secret. All other information used in the procedures is generally public. However, this can lead to a fundamental problem:

1. Assume that devices A and B use A's unit key as their link key.
2. Later on, or at the same time, device C may communicate with device A and also use A's unit key as the link key.
3. B can use A's link key to decrypt the communication between A and C.

Figure 6.1 illustrates the potential security loophole.

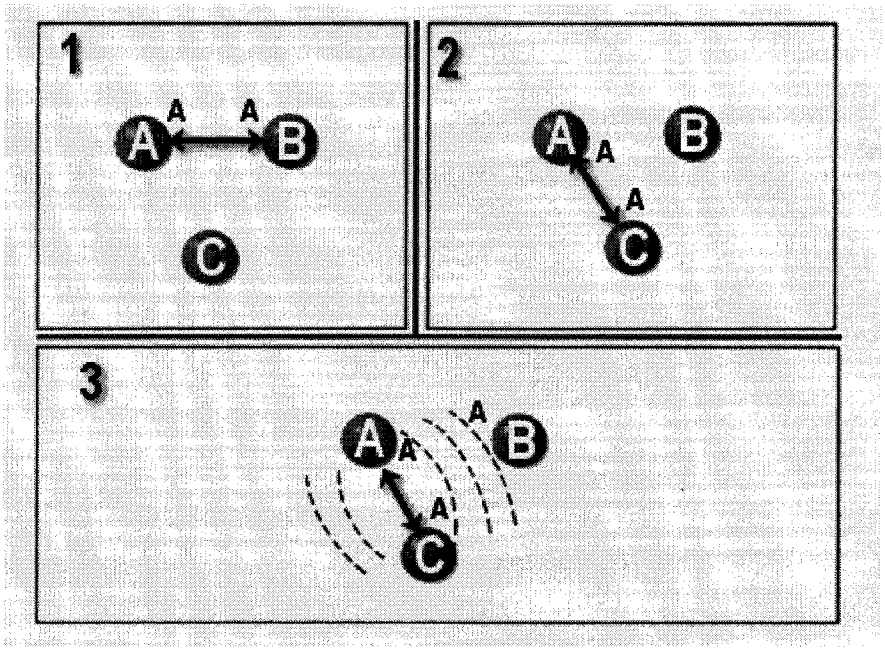

Figure 6.1 Spoofing due to non-secret link key [20].

As shown above, device B, having obtained A's unit key earlier, can use the unit key with a faked BD_ADDR to calculate the encryption key and therefore listen to the traffic. It can also authenticate itself to device A as device C and to device C and device A.

The actual implementation of this attack is not as easy as it sounds, but it has been shown to be possible by the scientists at Bell Laboratories of Lucent Technologies.

6.1.3 Spoofing due to the unique BD_ADDR

The Bluetooth device address is unique to each and every Bluetooth device. However due to its uniqueness it introduces another problem. Once a BD_ADDR is associated with a person, his or her activities can be easily traced and monitored. This violates the principle of privacy.

6.2 Realistic Bluetooth Security Expectations in the Future

All of these problems might lead one to believe that Bluetooth security is not sufficient. However, one must take a major factor into consideration when evaluating the security of Bluetooth: the nature of the data being transmitted across the Bluetooth link.

Bluetooth was never designed with the intention to transmit truly sensitive data. By default, Bluetooth is not a true competitor of WLAN, for which security is paramount, but rather it was intended to form PAN's, for which security is desirable but not essential. After all, who would really want to intercept a normal user's MP3 transfer?

Also, another fact rarely mentioned is that the range for Bluetooth transmission is usually within 10 meters, and in a real environment, the range is likely to be shortened to 5 meters. In this case, the hacker must sit across the room from the user, which is a bit closer than most hackers have been observed.

6.3 Conclusion:

It has been proven again and again that no method of communication is entirely safe. If a malicious entity wants to obtain information there is always the possibility it will succeed. The most one can do is to make it as difficult as possible. Bluetooth in its current state is quite capable to resist most attacks, and its security features are sufficient for its normal intended use. However, its security does have some major problems, thus Bluetooth should not generally be used to transmit highly confidential or sensitive data until these major issues are resolved.

6.4 Possible Future Improvements:

Due to the apparent success and uptake of Bluetooth in recent years, more and more it is being extended to become a higher end wireless standard. It is on its way to

become a real competitor to WLAN. To accommodate this new trend, researchers of Bluetooth can either seek to improve existing security or implement new security:

- Improve existing security:

Users requiring stalwart protection are encouraged to use stronger security mechanisms available in network transport protocols and application programs, i.e. to use security mode 2, the security features on adopted protocols such as PPP, and security applications designed to improve general or specific security.

- Implement new security:

It must be realized that no matter how much more secured transmissions can be made at the service level, Bluetooth still has fundamental problems at the link level , as described earlier in this section. If Bluetooth wants to evolve to become a true competitor of WLAN, its security features must be expanded and strengthened to match those of WLAN.

7. Appendix

7.1 Acronym List:

ACO:	Authenticated Ciphering Offset
ARQ:	Automatic Repeat Request
AU_RAND:	Authenticating Random Input
BD_ADDR:	Bluetooth Device Address
CLK:	Master Clock
CPU:	Central Processing Unit
CRC:	Cyclic Redundancy Check
CMOS:	Complementary Metal Oxide Semiconductor
COF:	Ciphering Offset Number
FEC:	Forward Error Correction
GFSK:	Gaussian Frequency Shift Keying
HCI:	Host Controller Interface
ISM band:	Industrial, Scientific and Medical band
L2CAP:	Logical Link Control and Adaptation Protocol
LC:	Link Controller
LFSR:	Linear Feedback Shift Register
LM:	Link Manager
LMC:	Link Manager Protocol
LSB:	Least Significant Bit
MAC:	Medium Access Control
MMI:	Man Machine Interface
MSB:	Most Significant Bit
OBEX:	Object Exchange
OSI:	Open System Interconnection
PAN:	Personal Area Network
PDA:	Personal Digital Assistant
PRBG:	Pseudo Random Bit Generator
PHT:	Pseudo Hadamard Transform
PIN:	Personal Identification Number
RAND:	Random Number
SDP:	Service Discovery Protocol
SIG:	Special Interest Group
SRES:	Signed Response
TCP/IP:	Transmission Control Protocol / Internet Protocol
TCS:	Telephony Control Protocol Specification
WAP:	Wireless Application Protocol
WLAN:	Wireless Local Area Network
XOR:	Exclusive OR

8. Bibliography

1. King Bluetooth
   http://www.cellular.co.za/bluetooth_king_harald.htm

2. Nokia and Bluetooth: What is Bluetooth?
   http://www.nokia.com/bluetooth/whatis.html.

3. The Official Bluetooth SIG Website
   http://www.bluetooth.com

4. A.Menezes, P. van Oorschot, S. Vanstone, "Handbook of Applied Cryptography", CRC Press, 1996.

5. "Specification of the Bluetooth System", Version 1.1, Volume 1, Bluetooth SIG Publications, February 22nd, 2001.

6. J Bray, C.F. Sturman, "Blutooth – Connect Without Cables", Prentice-Hall Inc., 2001.

7. N. J. Muller, "Bluetooth Demystified", McGraw-Hill Inc., 2001.

8. Bluetooth Tutorial - Radio
   http://www.palowireless.com/bluearticles/radio.asp

9. K. Feher, "Wireless Digital Communications – Modulation & Spread Spectrum Applications", Prentice-Hall Inc., 1995.

10. Wireless Developer Network – An Introduction to Bluetooth
    http://www.wirelessdevnet.com/channels/bluetooth/features/bluetooth.html

11. VBXML.COM: Bluetooth: A Programmer's Prime
    http://www.vbxml.com/wap/articles/bluetooth_overview/stecktext2.asp

12. Ericsson Technology Licensing
    http://www.ericsson.com/bluetooth

13. D. R. Stinson, "Cryptography – Theory and Practice", CRC Press, 1995.

14. NIST Homepage > Computer Security Resource Center > AES
    http://csrc.nist.gov/encryption/aes

15. W Diffie, M.E. Hellman, "New Directions in Cryptography", IEEE Transactions on Information Theory, Vol IT-22, 1976.

16. R. L. Rivest, A. Shamir, L. A. Adleman, "A Method for Obtaining Digital Signatures and Public-key Cryptosystems", Communications of the ACM, Vol.21, 1978.

17. T. ElGamal, "A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms", IEEE Transactions on Information Theory, Vol IT-31, 1985.

18. T. Mueller, "Bluetooth Security Architecture White Paper", Bluetooth SIG Publications, July 15th, 1999.

19. An Overview of Bluetooth Security
http://www.sans.org/infosecFAQ/wireless/bluetooth.htm

20. Bluetooth Security
http://www.palowireless.com/bluearticles/cc1_security1.asp

21. Bluetooth Security Paper by J. T. Vainio
http://www.niksula.cs.hut.fi/~jiitv/bluesec.html

22. J. L. Massey, R. A. Rueppel, "Linear Ciphers and Random Sequence Generators with Multiple Clocks", Proceedings of EUROCRYPT 84, Vol 209, 1984.

23. SAFER+ Downloads
http://www.cylink.com/library2/downloadbody.htm