POWER AWARE PUBLIC KEY AUTHENTICATION FOR BLUETOOTH

By

BALDEEP ANAND

A THESIS PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE

UNIVERSITY OF FLORIDA

2003

To my parents and Niki

ACKNOWLEDGMENTS

I would like to express my gratitude to my advisor, Dr. Abdelsalam (Sumi) Helal, for his encouragement and motivation all throughout my work. I also thank Dr. Michael P. Frank and Dr. Douglas Dankel for serving on my thesis committee.

I would also like to thank Brent Miller at IBM Corporation and Sami Kibria at TRLabs for some valuable discussions.

Finally, I would like to thank my friends at the Harris Lab for their support and great company.

TABLE OF CONTENTS

page

ACKNOWLEDGMENTS iv

LIST OF TABLES vii

LIST OF FIGURES viii

ABSTRACT ix

CHAPTER

1 INTRODUCTION 1

1.1 Problem Statement 2
1.2 Organization Of The Thesis 3

2 BLUETOOTH WIRELESS TECHNOLOGY 4

2.1 Bluetooth 4
2.2 The Protocol Stack 6
2.3 Bluetooth Profiles 8

3 SECURITY BASICS 11

3.1 Security Goals 11
3.2 Cryptography 11
3.2.1 Symmetric Key Cryptography 12
3.2.2 Public Key Cryptography 13
3.2.3 Building Blocks Of Public Key Cryptography 13

4 EXISTING BLUETOOTH SECURITY 16

4.1 Bluetooth Security Features 16
4.1.1 Pairing and Bonding 17
4.1.2 Key Management Procedures 19
4.2 Analysis Of Bluetooth Security Model 24

5 POWER AWARE PUBLIC KEY AUTHENTICATION FOR BLUETOOTH 26

5.1 Protocol Overview 26

5.1.2 Key Distribution 27
5.2.2 Authentication 28
5.2.3 Encryption 28
5.2.4 Access control 29
5.3 Protocol Details 29
5.4 Fitting Our Protocol Into Bluetooth 30
5.5 Implementation Details 33
5.5.1 Signature Algorithm 33
5.5.2 Key Distribution 34
5.6 Performance Evaluation 34

6 CONCLUSION 36

LIST OF REFERENCES 38

BIOGRAPHICAL SKETCH 40

LIST OF TABLES

Table page

5-1 Notation used in the protocol....27

5-2 Protocol steps....29

5-3 LMP messages used during connection establishment....31

5-4 Additional LMP messages added for PKI authentication. 31

5-5 Time required by different signature algorithms....34

5-6 Code size of different modules....35

5-7 Time taken for different security operations 35

LIST OF FIGURES

Figure page

2-1 Bluetooth protocol stack 7

2-2 Bluetooth profile structure 9

3-1 Encryption and decryption 12

4-1 Pairing 18

4-2 Bonding 18

4-3 Unit key generation 19

4-4 Generation of initialization key. 20

4-5 Authentication using initialization key. 21

4-6 Link key generation 22

4-7 Generation of combination key on device A. A similar procedure takes place on device B where LK_RAND A is received from A. 22

4-8 Generation of encryption key 23

5-1 Fitting our protocol into Bluetooth 32

Abstract of Thesis Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Master of Science

POWER AWARE PUBLIC KEY AUTHENTICATION FOR BLUETOOTH

By

Baldeep Anand

May, 2003.

Chair: Abdelsalam (Sumi) Helal.
Major Department: Computer and Information Science and Engineering.

Although Bluetooth has been touted as an emerging technology for mobile commerce (m-Commerce) applications, its limited security model poses a significant hindrance to its utility in such applications. Current Bluetooth security architecture is sufficient for home networking applications but certainly not for credit card transactions. We propose a public key security protocol that would strengthen Bluetooth security and gear it towards m-commerce applications.

After analyzing the weaknesses of the existing Bluetooth security architecture, we present our protocol and discuss the effects on performance and discuss the trade-offs of using public key authentication on the resource constrained devices on which Bluetooth is intended to be used. The power aware public key authentication protocol has been implemented on Axis's openBT Bluetooth protocol stack on Linux platform.

CHAPTER 1
INTRODUCTION

In what is being called the second half of the information age, we are seeing that the computing tool is shifting from a crunching tool to a connecting tool. Mobile and wireless devices are gaining popularity as they provide unrestricted movement and the ability to access the network from almost anywhere in the world. In the near future, it is expected that millions of users will have access to on-line distributed systems through mobile and wireless devices.

Bluetooth is the new emerging technology for wireless communication. It was developed by the Bluetooth Special Interest Group (SIG), formed in May, 1998. Bluetooth is a way of connecting machines to each other without using cables or any other physical medium. It uses radio waves to transfer information and can be used to connect almost any device to any other device. Bluetooth wireless technology is set to revolutionize the personal connectivity market by providing freedom from wired connections. It is a specification for a small form-factor, low-cost radio solution providing links among mobile computers, mobile phones and other portable and handheld devices, and connectivity to the Internet. With Bluetooth, it is feasible to have several interesting usage scenarios like the following:

1. Electronic identity validation by simply walking through a doorway with your phone in your pocket.
2. Using a multipurpose PDA to purchase electronic tokens or tickets which are "punched" as you drive or walk by a sensor.

3. Doctors automatically know the identity of emergency care patients who carry Bluetooth-enabled identification cards. Because Bluetooth chips are always on and seeking other Bluetooth devices, a doctor can call up a patient's information in a central database by toting another Bluetooth-enabled device.

1.1 Problem Statement

Today we have a continuously growing market for mobile and wireless communication. A key requirement of this new world of computing will be the ability to access data securely. As it is hard to restrict access to network resources physically, security becomes mandatory in such networks. The dynamic and unpredictable nature of the network and use of wireless links which are susceptible to link attacks ranging from passive eavesdropping to active impersonation, message replay and message distortion make the security requirements of these networks more stringent and harder to implement than in traditional wired networks.

Bluetooth is an emerging technology for wireless communications. But, limited security of Bluetooth has deterred its widespread deployment. For example, patients do not want sensitive information readily available on Bluetooth-enabled patient identification cards. One of the problems of implementing security on the resource constrained devices on which Bluetooth runs, is the computational overhead caused by the existing security protocols.

In this thesis we examine the existing security mechanisms in Bluetooth, identify weaknesses and propose a new protocol for secure communications. The proposed protocol focuses on incorporating public key cryptography for authentication. At the same time we make use of secret key cryptography (which is computationally less expensive) to accommodate the limited battery power and computational capability

resource constrained devices. We also try to reduce the cost of increased security by running some security operations in parallel.

1.2 Organization Of The Thesis

Chapter 2 provides a background on the Bluetooth wireless technology. Chapter 3 discusses general purpose security goals and explains most of the security related terms used in the thesis. Existing Bluetooth security and its weaknesses are discussed in Chapter 4. Chapter 5 discusses our power aware public key security protocol in detail and provides performance evaluation. Chapter 6 provides the conclusion and suggests future work.

CHAPTER 2
BLUETOOTH WIRELESS TECHNOLOGY

This chapter provides an introduction to the Bluetooth Wireless technology. It also details the different layers of the Bluetooth protocol stack. It also defines several terms used throughout this thesis.

2.1 Bluetooth

With the increase in the number of devices and peripherals connected to a computer, numerous cables can be seen in the proximity of computers. As a new technology for wireless connectivity, Bluetooth offers a solution for removing long cables trailing all over the place to connect devices. By using a radio-based link it connects these devices together without a single cable, providing greater freedom to roam. People using Bluetooth will no longer need to connect, install, enable or configure computer peripherals. Bluetooth makes it possible to use any compatible portable and stationary communication device without an inch of cable.

Bluetooth is a low-cost, low-power, short-range radio technology, originally developed by the Bluetooth Special Interest Group (SIG) as a cable replacement to connect devices such as mobile phone handsets, headsets and portable computers. The Bluetooth SIG, comprising of leaders in the telecommunications, computing, and network industries, is driving development of the technology and bringing it to market. Bluetooth has evolved over time and is on the way to become an industry standard for short-range wireless communications.

Communication using the Bluetooth protocol stack involves discovering devices that are in range, establishing connection, performing service discovery and then accessing those services. Bluetooth devices operate at 2.4 GHz unlicensed ISM band using a Frequency Hopping Spread Sequence (FHSS). A Bluetooth Chip is designed to transmit and receive information by hopping 1600 times per second on 79 different frequencies from 2402 MHZ to 2480 MHZ in a pseudo-random fashion. It uses Guassian Frequency Shift key modulation with a maximum data rate of 721Kbits/sec and maximum range of 100 meters. With Bluetooth, it is possible to have up to 3 high quality voice channels simultaneously [Bray and Sturman 2000]. A radio using Bluetooth consumes much lesser power than a modern mobile phone. Bluetooth limits the radio microchip's output power exactly to that actually required. If the receiving radio indicates that it is only a few meters away, the transmitter immediately modifies its signal strength to suit the exact range.

Bluetooth can be used to form ad hoc networks. Such a group of Bluetooth devices, connected together is called a Piconet. Two or more independent and non-synchronized piconets communicating with each other form a Scatternet. Bluetooth devices operate in one of the two modes, as a master or a slave. A device that initiates an action or requests a service on a piconet is called the Master device. The master’s clock and hopping sequence are used to synchronize all other devices in the piconet. A device that is not a master is called slave. A device cannot be a master in two piconets. There can be at most 7 slaves within a piconet. Blueooth allows both time-critical data communication such as that required for voice or audio, as well as high-speed, time insensitive packet data communication. For this two different links are defined between any two devices. These

are SCO (Synchronous Connection Oriented) links for voice communication and ACL (Asynchronous Connectionless) links for data communication.

2.2 The Protocol Stack

Bluetooth is an open specification for wireless communication of data and voice. As shown in figure 2-1, the Bluetooth protocol stack is a multilayered architecture divided in two main categories:

1. The Core specification that describes how the technology works, and
2. The Profiles specification that describes how the technology is used to fulfill a desired function for a Bluetooth device.

Radio. The Bluetooth radio layer defines the radio transmitter characteristics like power level, modulation parameters and tolerance, and receiver characteristics like sensitivity level, interference performance, out-of-band blocking, inter-modulation characteristics and receiver signal strength. It is also responsible for frequency band and channel agreement.

BaseBand. BaseBand is the lowest communication layer and is responsible for actual physical channel establishment. The physical channel consists of a random hopping sequence through 79 different radio frequencies. Packets are sent on the physical channel, where each packet is sent on a different hopping frequency. The baseband is responsible for low-level link control information like acknowledgement, flow control and payload characterization. The Bluetooth baseband supports both synchronous connection-oriented and asynchronous connection-less links.

Link manager. The link manager (LM) is responsible for link setup, control and configuration. Security procedures for encryption and authentication are also handled on this layer. The LM communicates with other remote LMs using the Link Manager

protocol (LMP). It converts the commands given by the Bluetooth host into operations at baseband level.

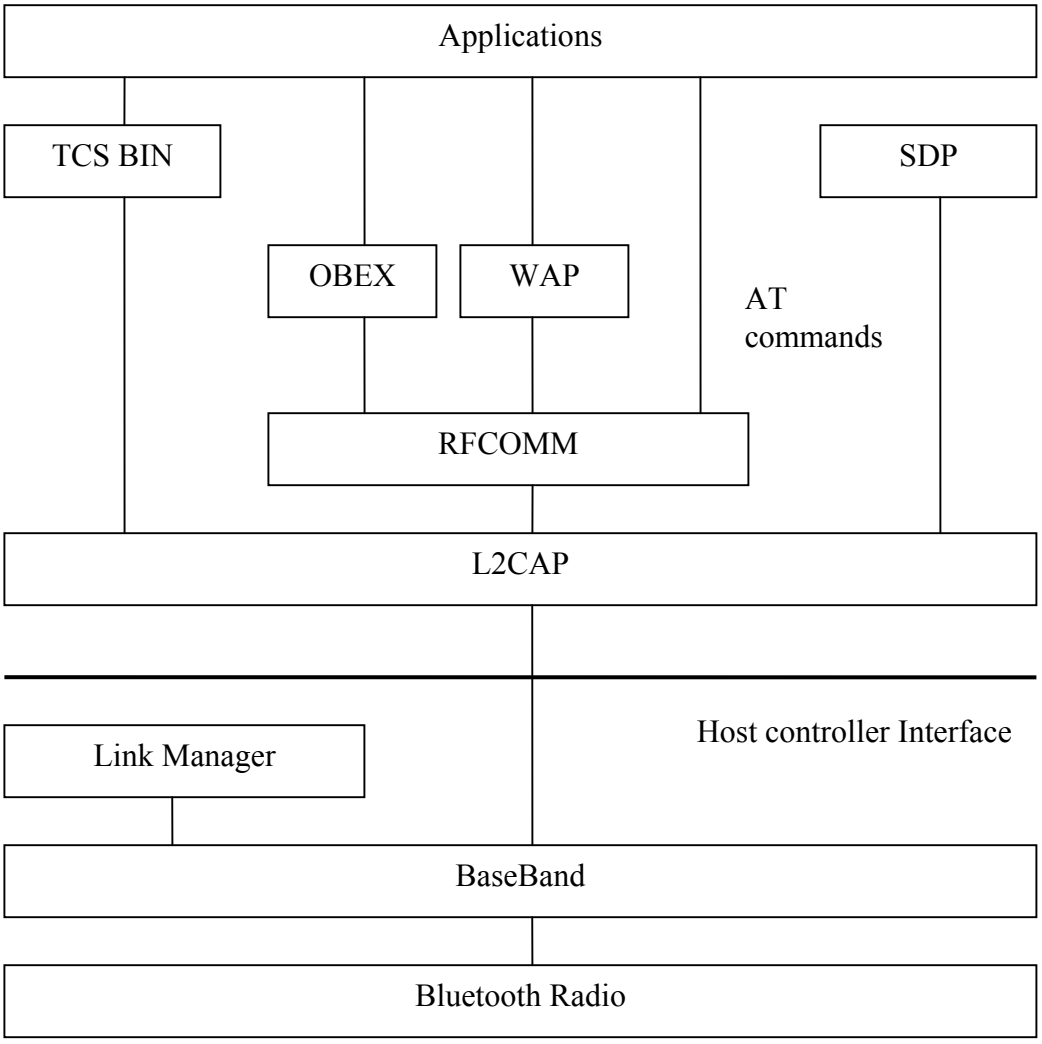


Figure 2-1 Bluetooth protocol stack

Host controller interface. Most of the Bluetooth systems have baseband and Link Manager on one processor and higher layers like L2CAP, RFCOMM and SDP and applications on a separate host processor. The host controller interface (HCI) provides an

interface between the higher and the lower layers. This enables the applications to interact with the hardware without knowledge of the hardware implementation details.

Logical link control and adaptation layer. The logical link control and adaptation layer, L2CAP, provides protocol multiplexing, segmentation and reassembly, and group abstractions capabilities to connection-oriented and connection-less data services at upper layers. L2CAP permits higher level protocols and applications to transmit and receive L2CAP data packets up to 64 kilobytes in length. L2CAP defined for only asynchronous connection-less links.

TCS BIN and AT commands. These are telephony control protocols that allow services such as modems and fax to run over Bluetooth.

RFCOMM. The RFCOMM protocol provides serial port emulation over the L2CAP. It is based on a subset of the ETSI standard, TS 07.10. It relies on the Bluetooth baseband to deliver reliable in-sequence byte-stream. It supports both direct communications between devices, acting as endpoints, and device-modem-device communication. The RFCOMM also has a built in scheme for null modem emulation.

Service discovery protocol. The service discovery protocol (SDP) provides a means for applications to discover available service and determine their characteristics. SDP is of significant importance in ad hoc networks formed by Bluetooth devices where network topologies and available services are not known in advance. SDP does not provide any means of accessing these discovered services.

2.3 Bluetooth Profiles

The Bluetooth standard is supposed to be used by a wide range of manufacturers. The areas where the Bluetooth standard could be implemented are unlimited. To ensure that all devices using the technique are compatible with each other, standard schemes for

communicating are needed in the main areas. The standard schemes are called profiles and specify messages, security procedures, idle procedures, and establishment procedures used between the devices. The Bluetooth profiles describe how to use the Bluetooth protocols in an interoperable way.

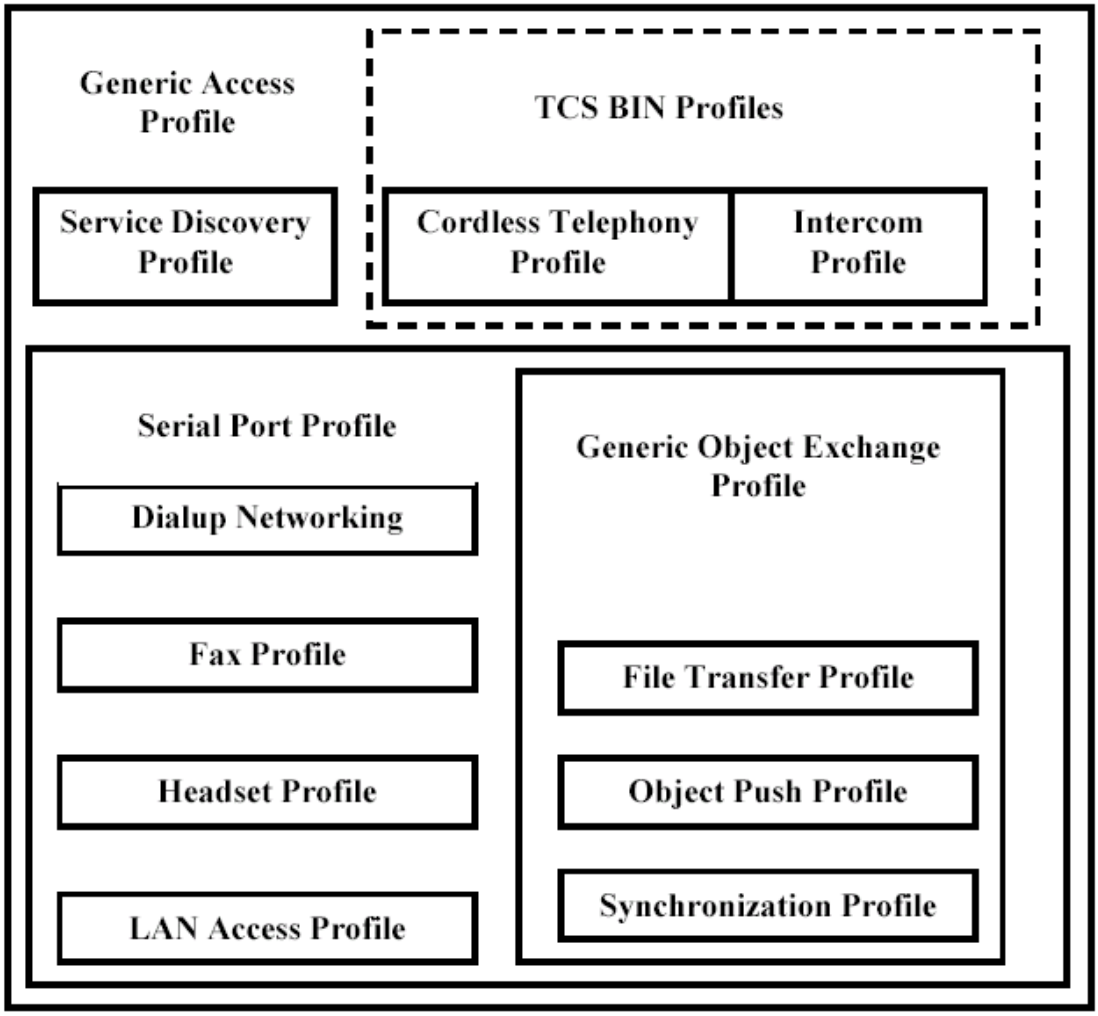


Figure 2-2 Bluetooth profile structure

The Bluetooth SIG has identified various usage models, each of which has a supporting profile that defines the protocol and features for the particular usage model. Some of the Bluetooth profiles are the intercom profile, cordless telephony profile,

headset profile, dialup networking profile, fax profile, LAN access profile, file transfer profile, object push profile and synchronization profile, as shown in figure 2-2. The Bluetooth profile specification describes the abovementioned profiles in detail.

CHAPTER 3
SECURITY BASICS

This chapter provides a background on goals of a general security system and a brief description most of the terms related to security that have been used in this document.

3.1 Security Goals

The main goals of a general purpose security system [Stallings 1999] are:

1. Authentication--Authentication enables a node to ensure the identity of the peer node with which it is communicating. Without authentication, an adversary could masquerade a node, thus gaining unauthorized access to a resource and sensitive information and interfere with the operation of other nodes.

2. Availability--Availability ensures the survivability of network services despite denial of service attacks and network failures.

3. Confidentiality--Confidentiality ensures that certain information is never disclosed to unauthorized entities.

4. Integrity--Integrity guarantees that a message being transferred is never corrupted. A message could be corrupted because of benign failures, such as radio propagation impairment, or because of malicious attacks on the network.

5. Non-repudiation--Non-repudiation ensures that the origin of a message cannot deny having sent the message. It is useful for detection and isolation of compromised nodes. When a node A receives an erroneous message from a node B, it allows A to accuse B using this message and to convince other nodes that B is compromised.

3.2 Cryptography

The abovementioned goals are achieved by using the art and science of cryptography. As shown in figure 3-1, encryption is used to transform information (also known as plaintext) into an unintelligible form (known as cipher text) by using a reversible translation known as decryption.

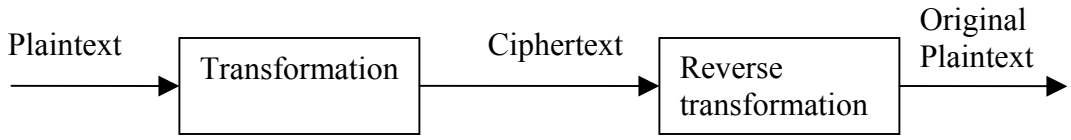


Figure 3-1 Encryption and decryption

Encryption and decryption are based on some secret information shared by the communicating parties. This could be a secret key or a secret algorithm used for encryption/decryption. A general model for encryption/decryption involves the following steps:

1. Development of an algorithm for transformation and reverse transformation.
2. Generation of the secret information, also known as Key Generation.
3. Development of methods used for distribution of the secret information, also known as Key Distribution, and
4. Development of a protocol that makes use of the algorithm and the secret information to achieve a security goal.

Based on the number of keys used, cryptographic schemes are broadly classified into two categories:

1. Symmetric-key or secret-key cryptography, and
2. Public key cryptography.

3.2.1 Symmetric Key Cryptography

In symmetric key cryptography, a single key is used for both encryption and decryption. The security of this system depends on the strength of the key (i.e. key generation algorithm and key length). As the same key is used by the sender and the receiver, key distribution is a major issue in symmetric-key cryptography. It also

necessitates storage of one key for every pair of communicating parties. Also, if the key is shared by more than two people, there is no way to establish the authenticity of the originator. However, the advantage of using this scheme is that it is simple and considerably faster than public key system in terms of required computation time for encryption/decryption.

3.2.2 Public Key Cryptography

Public key cryptography was developed to address the problems with symmetric key cryptography. It uses a pair of keys, one for encryption and the other for decryption. This pair of keys is called a public key-private key pair. The public key is made public by using some key distribution mechanism like publishing. The private key is kept secret and is known only to the owner of the key pair. Although the keys are mathematically related to each other, it is computationally infeasible to determine the private key knowing the public key. Anyone can encrypt a message using someone's public key. However, only the specified person can decrypt it because only he knows the private key. A problem with public key systems is that the power needed to encrypt and decrypt a message is much higher than symmetric-key cryptography. However, careful selection of protocols and protocol optimizations can reduce this overhead within tolerable limits.

3.2.3 Building Blocks Of Public Key Cryptography

Several techniques are used for achieve the security goals in public key cryptography. Some of them are described in the following section.

Hashing. Hashing is the process of mapping a variable-length data block or message into a fixed length value called a hash code or message digest. As it is based on all bits of the message, the hash code provides an error-detection capability. A change in any bit of the message causes a change in its hash code. Hence, it is a means of verifying message

integrity. Also, it is impossible to determine the message by knowing its hash code. Hence, the best hashing functions are many-to-one and one way.

Digital signatures. Digital Signature is an authentication mechanism that attaches a code to the message that acts as a signature thereby, enabling the receiver to guarantee the source and integrity of the message. This code acts as a signature and is analogous to the handwritten signature. The purpose of a digital signature on any kind of data is to verify the identity of the person who digitally signed the code and to verify the integrity of the data. Digital signature is one of the most important developments from the work on public key cryptography. The complementary operation to signature is Verification i.e. it is the process of making sure that the signature is signed by the claimant. Some of the requirements of a digital signature are:

1. It should be message dependent.
2. It should use some information unique to the sender. This prevents forgery and denial.
3. It should be computationally infeasible to forge a digital signature.

A simple scheme (as used in the RSA approach) of creating digital signature over a message is to create a hash of the message and encrypt this hash using the sender's private key. This signed hash when appended to the original message can be used to prove the authenticity of the source and the integrity of the message. The digital signature algorithm also uses a hash function along with a set of globally known parameters to compute the digital signature over a message and relies on the difficulty of computing discrete logarithms. Another digital signature scheme known as ESIGN from NTT Japan is based on the intractability of factoring the product of a large prime number and the square of another large prime number. It is at least as secure as and considerably faster

than either RSA or DSA, with similar key and signature lengths [Okamoto and Kobayashi 1999].

Public key certificates. Public key certificates are a means of distributing public keys. A public key certificate contains information about a subject, including its public key, and a signature by a trusted authority, often called the Certificate authority (CA). Hence, a receiver can obtain the public key and other information about the subject by verifying the subject's public key. It is assumed that both the sender and the receiver have access to trusted authority's public key. The following are the requirements on public key certificates:

1. It is possible for anybody to read the certificate to read the subject's name and public key.
2. It is possible for anybody (who, knows CA's public key) to verify that the certificate originated from the CA.
3. Only the CA can create and make changes to the certificate.

CHAPTER 4
EXISTING BLUETOOTH SECURITY

This chapter describes the existing Bluetooth security architecture. It gives an introduction to the different security modes used in Bluetooth. Then it analyzes this security architecture for weaknesses. This chapter provides the motivation behind our work.

4.1 Bluetooth Security Features

Bluetooth promises to open up and link different networks by linking ubiquitous devices such as PDAs and cell phones to different types of hardware platforms thus bringing about pervasive connectivity. However, Security concerns are slowing down the mass adoption of this wireless technology.

Bluetooth employs frequency hopping, a practice of skipping around the radio band 1600 times each second. This improves clarity and also reduces casual eavesdropping by allowing only synchronized devices to be able to communicate. This provides a certain level of security by obscurity. Bluetooth also supports security features at the link level. Authentication and encryption are based on a secret link key that is shared by a pair of devices and is generated by using a pairing procedure when the devices communicate for the first time. Bluetooth enabled devices can operate in one of three different security modes [Muller 1999]:

1. Mode 1 - The most insecure mode in which the Bluetooth device does not initiate any security procedures and allows other Bluetooth devices to establish connections with it.

2. Mode 2: Service level enforced security (i.e., security is enforced after an L2CAP link is established). It allows enforcement of security policies and fine-grained application layer controls on top of the lower layer protocols. Each service is free to implement its own security procedures.

3. Mode 3: Link level enforced security (i.e., security is enforced before the establishment of the link). The link level functions are defined in the Bluetooth Baseband and the Link Manager Protocol Specifications.

There are several levels of trust for Bluetooth devices and services. Devices have the following three levels:

1. Trusted Device: One with which a trust establishment has been done and has access to all services.

2. Untrusted Device: One that has a restricted access to services.

3. Unknown Device: Also an untrusted device.

Services also use three modes of trust, which are combination of authentication and authorization, as follows:

1. Services that require authentication and authorization: Trusted devices have access automatically. Untrusted devices need authorization.

2. Services that require only authentication: No Authorization required.

3. Services accessible by all devices: Neither authentication nor authorization is required.

4.1.1 Pairing and Bonding

Security procedures in Bluetooth communication are established by what is known as pairing and bonding. Two devices are bonded if they share a common link key for communication. The Bonding procedure involves establishing a link to create and exchange a common link key. Pairing is the collective link level procedure for authentication and key generation, as shown in figure 4-1.

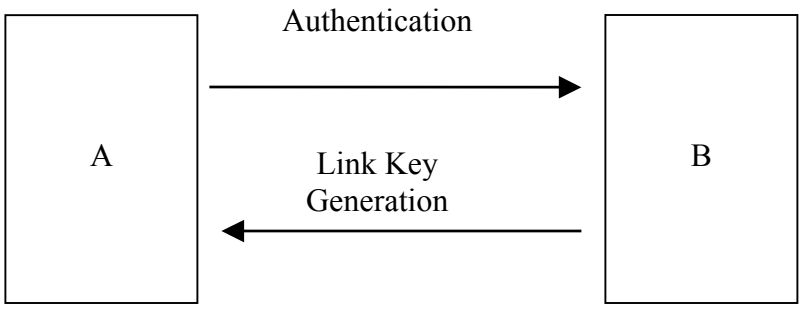


Figure 4-1 Pairing

Bonding can either be general bonding or dedicated bonding. In dedicated bonding the devices perform only link key generation whereas general bonding would involve the intervention of higher layers to initialize their security parameters.

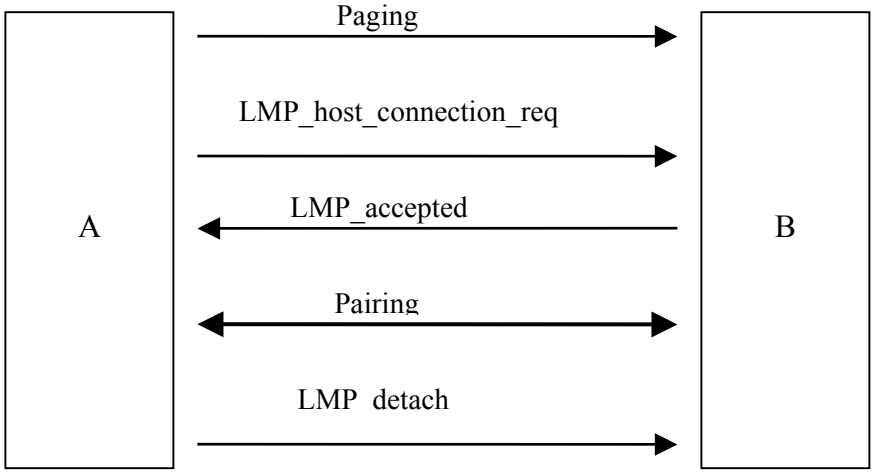


Figure 4-2 Bonding

At the user level, Bluetooth bonding is used to collectively refer to pairing and bonding procedures. As shown in figure 4-2, A sends a connection request to B, then they

undergo the pairing process, and then this link is torn down after the link key has been exchanged. Nodes A and B are then said to be bonded.

4.1.2 Key Management Procedures

Several keys are used throughout the authentication and encryption process. Key generation and initialization are done as follows [Lamm, Falauto, Estrada and Gadiyaram 2001]:

Generation of Unit key: The unit key, K is normally generated once in the device lifetime when it first boots up. As shown in figure 4-3, it is generated using the BD_ADDR (a 48-bit unique device address) and RAND (a random number), and is stored in the device's permanent memory.

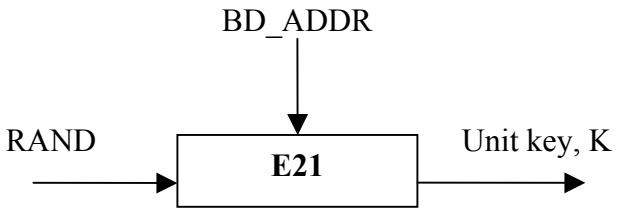


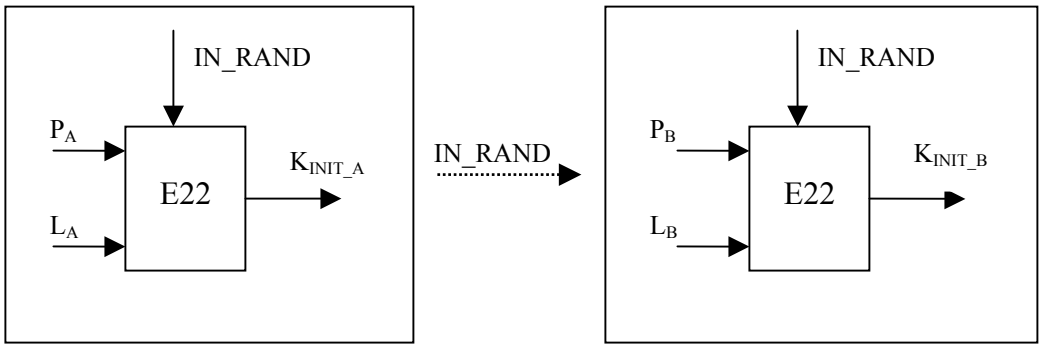
Figure 4-3 Unit key generation

Generation of initialization key (used as a temporary link key): This key is generated in both the devices trying to communicate. As shown in figure 4-4, this key is generated using a random number IN_RAND (which is passed by one device to the other in clear), the PIN and the PIN length.

Kinit = E22 (IN_RAND, P, L)

where, E22 is the algorithm used for the generation of the initialization key Kinit,

P is the PIN number and L is the PIN length.



Device A **Device B**

Figure 4-4 Generation of initialization key.

Authentication using initialization key in each device: Suppose device A wants to communicate with device B. As shown in figure 4-5, the following takes place:

1. A performs an inquiry scan and obtains B's BD_ADDR.
2. A sends a random number (challenge) to B (AU_RAND).
3. Both A and B compute SRES and SRES' using the E1 algorithm with BD_ADDR, KINIT, and AU_RAND as the parameters.
4. B sends SRES' to A.
5. If SRES = SRES', then the authentication procedure is successful and the devices can proceed with further communication.

Successful Authentication depends on whether the two devices share the same initialization key or not, which in turn depends on whether they share the same PIN or not.

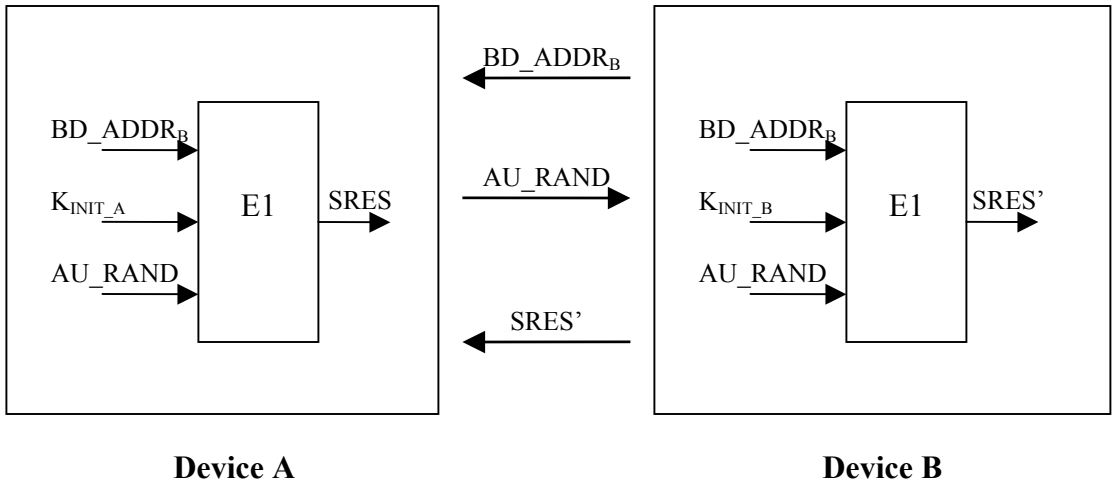


Figure 4-5 Authentication using initialization key.

Generation of link key in each device: The link key to be decided upon to be used in future communication by the two devices could be either the Unit Key of either device or a Combination Key that is a function of parameters of both the devices (hence unique for each pair).

Link key exchange: If the Link Key has to be a Unit Key then it is sent to the other device by XORing with the initialization key, as shown in figure 4-6.

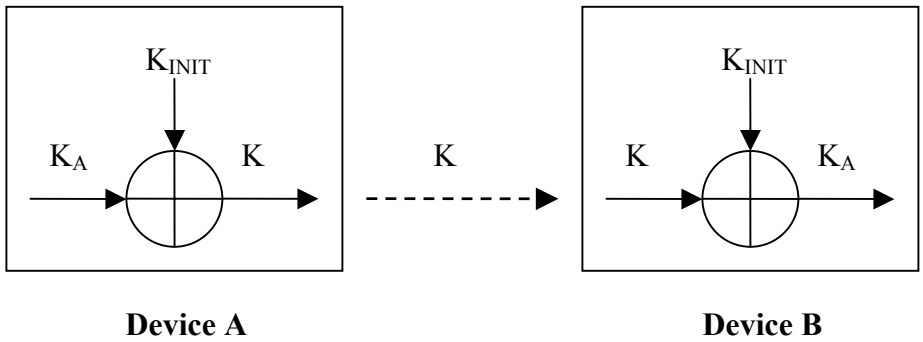


Figure 4-6 Link key generation

If the Link Key has to be a Combination Key, both sides generate a random number (LK_RAND) and calculate a temporary key (LK_K) using the E21 algorithm with the random number and their BD_ADDR as the parameters. Each device also computes the other device's key using a random number (passed by the other device in clear) and BD_ADDR of the other device. Then both devices compute the combination key by XORing its temporary key with the other device's key, as shown in figure 4-7. After this the old initialization key is discarded and a new authentication phase with this combination key is started.

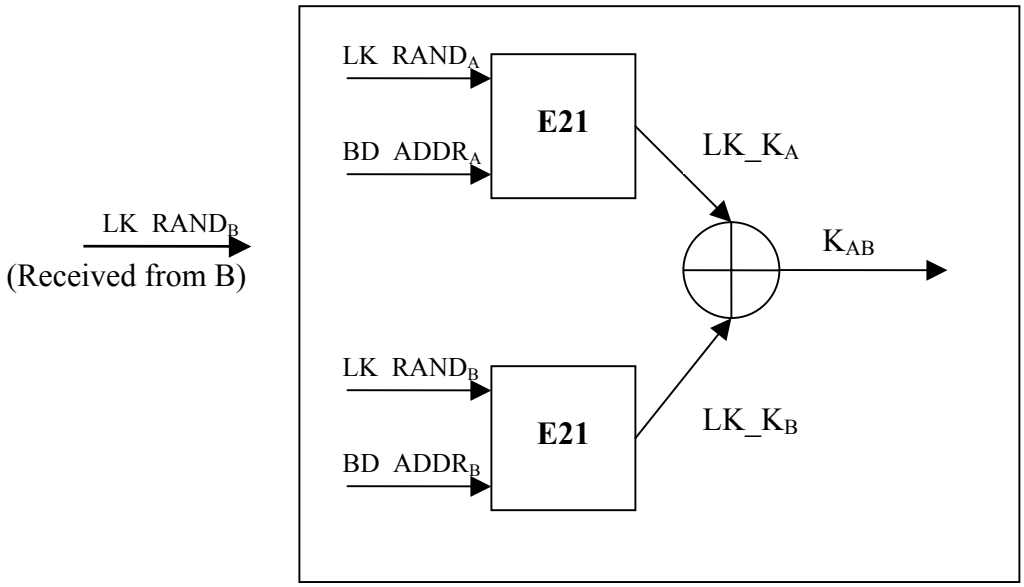


Figure 4-7 Generation of combination key on device A. A similar procedure takes place on device B where LK_RAND A is received from A.

Authentication using link key: This authentication procedure is the same as the initial authentication (that uses initialization key). The only difference is that now the shared link key is used in place of the initialization key.

Generation of encryption key in each device: If encryption is required, then the encryption key, Kc is generated using the E3 algorithm with parameters being the Link key, a ciphering offset (COF) and a random number (EN_RAND), as shown in figure 4-8.

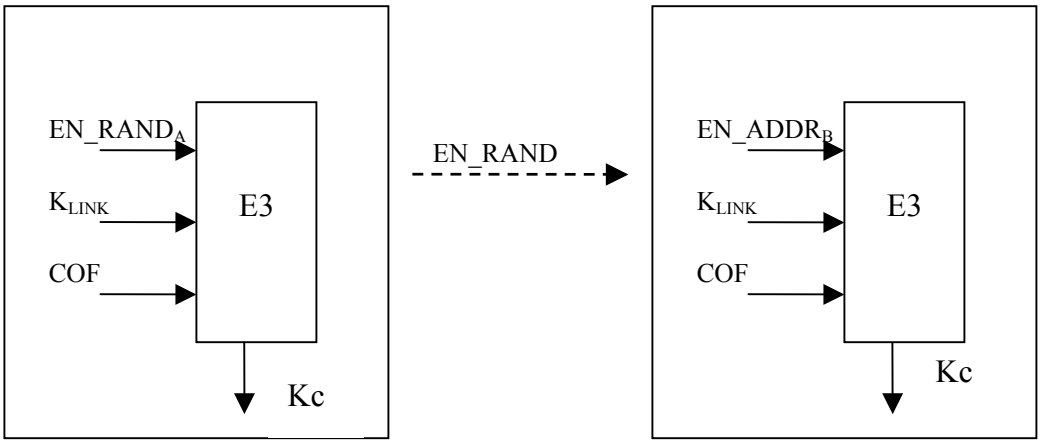


Figure 4-8 Generation of encryption key

Encrypted communication: The encrypted communication is carried out by performing encryption on the data using a cipher key Kcipher that is generated using the E0 algorithm with the parameters being the encryption key, BD_ADDR and the clock. This key is then XORed with the data for encryption purposes.

4.2 Analysis Of Bluetooth Security Model

Bluetooth security model is a simple shared secret device level security model that has several drawbacks and weaknesses and can be attacked in several ways.

PIN Code Attack: Bluetooth security relies heavily on a shared PIN code that is used to generate initialization key and subsequent link keys. This PIN code is usually a 4-digit number and most of the times it is "0000". Breaking a 4-digit PIN is not very difficult and since, it is the only secret piece of information shared by the two devices, a third party knowing the PIN can break into all the subsequent communication without much difficulty.

Man-in-the-Middle Attack: This attack also known as impersonation can by done by spoofing in the following ways:

- **Spoofing due to non-secret link key**: Device A communicates with device B using B's unit key as the link key. Then device A communicates with device C and gives B's unit key to C. After this the device C can communicate with B using this key by pretending to be any device that has previously communicated with B using this link key.
- **Spoofing the BD_ADDR**: Since the Bluetooth address of a device can easily be obtained by a general inquiry access routine, a device can perform an inquiry scan on another device and spoof its BD_ADDR to establish subsequent connections with other devices.

Exhausting the Battery: This attack (basically, a form of denial of service attack) is possible if a device is overwhelmed with connection requests and is not able to respond to legitimate requests.

No user level security defined: While Bluetooth claims to provide authentication, its authentication does not determine the identity of a subject (person), but only verifies that the nodes on a connection have the same link key that was exchanged in a pairing process. So, only device level authentication is performed, not user level.

Authentication misrepresented: The Bluetooth security procedure misrepresents authentication as the process of verification that the two parties share the same link key [Hall 2001]. It does not actually verify the identity of the other device.

The lack of user level security limits the use of Bluetooth to services that do not need user authentication. As Bluetooth devices become omnipresent, the access to services provided by them to public users has to be controlled. Consider a scenario with a Bluetooth enabled printer in a public laboratory. To have an account of the number of pages printed by each user, it is insufficient to use device level security. A computer X, can be used by different users at different times. Hence, user level security is important.

Moreover, Bluetooth security is based on a shared secret that is distributed to the communicating parties offline. In ad hoc networking scenarios where two devices communicate with into each other for the first time ever, this security mechanism cannot be used. The devices do not have any shared secret. Also, the shared secret solution needs to store a unique secret for each pair of communicating devices. This is clearly a big overhead.

Finally, key generation has to be done during the authentication process whenever a link key is not already present for a communicating device. This takes considerable time during the bonding process.

CHAPTER 5
POWER AWARE PUBLIC KEY AUTHENTICATION FOR BLUETOOTH

This chapter describes our Power Aware Public key authentication protocol in detail. It also describes how this protocol has been implemented on an existing bluetooth protocol stack.

5.1 Protocol Overview

We address the issue of user level authentication by proposing the use of conventional Public key cryptography. The main challenge is to make use of public key algorithms that are computationally less intensive so that they do not drain out the battery power or cause significant time overhead. Another challenge is to design a protocol that makes the cryptographic operations proceed in parallel on both ends to reduce response time penalty due to the increased security. Finally, the aim is to reduce public key cryptography as much as possible and make use of conventional (secret key) encryption. We tackle these challenges using the following approaches:

1. Key Distribution - by using public key certificates.
2. Authentication - by using a challenge-response scheme and digital signatures.
3. Encryption - by using session key (secret key cryptography).
4. Access control - by using levels in public key certificates.

Our approach enables us to easily implement user level security at link level. Although, if a public key-private key pair per device is used, the same technique can be used to implement device level security. After the session (link) key has been exchanged, the existing protocols in the Bluetooth Stack can be used for encryption over a link.

Table 5-1 Notation used in the protocol.

Pui	public key of user/device i.
Pri	Private key of user/device i.
X→ Y : a || b || c	X sends to Y, a message containing a, followed by b, followed by c.
Signature(x)	A message x signed by the sender using its private key.
K[a]	A message a encrypted with K. eg. PUa[m] denotes a message m encrypted with a's public key.

5.1.2 Key Distribution

We use public key certificates for Key distribution. These certificates are issued by a Certification authority and it is assumed that each user carry his/her own certificate. These can be downloaded to the device offline. The main contents of a public key certificate are:

1. Signature Algorithm specifier.
2. Issuer Name - Name of the Certification Authority (CA) issued the certificate.
3. Period of Validity - First and last dates on which the certificate can be used.
4. Subject Name - Name of the user to whom the certificate has been issued.
5. Subject's Public Key.
6. Level of Certificate.
7. Signature - Hash of fields 1-6 encrypted with CA's private key.

The level of certificate can be any information that would be used by services for access control. It could be an attribute of the subject that enables it to access some attribute-based services. For example, the attribute "frequent flyer" can provide a traveler

access to some privileged services that the airline offers. The only cryptographic tasks are hashing and signing the certificate. These operations are done rarely and often offline. Therefore, certificate creation is not a major overhead in our system.

5.2.2 Authentication

Once the certificates have been exchanged among the communicating parties, they need to verify the certificates and initiate an authentication protocol. Authentication is based on a challenge-response scheme and digital signatures. For example, A sends to B,

A → B: message m | Signature (m)

where, Signature (m) is the digital signature that can only be created by A (by encrypting m with its private key) and can only be verified by decrypting it with A's public key. When the signature is verified, B can be sure that the message came from A because only A knows his private key. In response to A's message, B sends to A,

B→ A: f (m) | Signature (f (m))

where, f (m) is a function of the received message m and is known to both the parties. This way A can authenticate B. In the above scheme, A can generate some messages mi in advance and can pre-compute Signature (mi). This saves time during authentication.

5.2.3 Encryption

Encryption is done using a session key that can be exchanged between A and B after they have authenticated each other. To exchange the session key, A sends to B

A → B: PU_b [Ks]

where, PUb is B's public key and Ks is the session key to be used for conventional encryption. Now, A and B can communicate with each other by encrypting messages using this session key.

5.2.4 Access control

This is the only security operation done above the link layer. Using the level of certificate attribute in the user's certificate, services can grant, deny or limit access to its features.

5.3 Protocol Details

The following table describes the protocol used for authentication and session (link) key exchange between two devices.

Table 5-2 Protocol steps

A	Message contents	B
1. Send to B	Request, Nonce n1, Ca, Signature (n1)	1. Receive request
2. Receive B's certificate	B's Certificate, Cb	2. Send to A
3.1 Verify B's certificate (obtain B's public key) 3.2 Generate a session key Ks (or select from a pool of pre-generated keys) 3.3 Encrypt Ks with B's public key.		3.1 Verify A's certificate (obtain A's public key) 3.2 Verify A's signature (authenticate A) 3.3 Calculate f(n1) 3.4 Sign f(n1)
4. Receive response from A and verify Signature over signed sent nonce.	Signed f(n1)	4. Send to A
5. Send to B	Encrypted Ks	5. Receive from A

		6. Decrypt Ks and use it as link key.
	<Encrypted Communication>	

From a careful examination of the protocol in table 5-2, we notice that A can verify B's certificate without authenticating B. This enables it to obtain B's public key in advance. This key is later used to verify B's signature and encrypt a session key. Therefore, B sends its certificate as soon as it receives the request from A. A verifies B's certificate, generates a session key Ks (or picks one from a pre-generated pool of keys to enhance performance) and encrypts it with B's public key. In the mean time, B verifies A's certificate and signature and sends it its own signature along with the response. After A verifies B's signature, it sends the encrypted session key to B and they can begin encrypted communication using Ks. As B's certificate is public, this step of verifying B's signature acts as an early detection of spoofing. This way we have parallelized some of the computationally intensive tasks at both ends. The above protocol requires one certificate verification, one signature verification, one signature generation and one decryption at B, and, one certificate verification, one encryption and one signature verification at A. The signature generation at A (for signing the initial nonce n1) can be done ahead of time. All operations at A except for the signature verification can be overlapped with the operations at B.

5.4 Fitting Our Protocol Into Bluetooth

The Link Manager (LM) handles the link level procedures for configuring security. In the following, we show how our protocol can be incorporated into the link layer. The LM

protocol (LMP) messages start with LMP_. Some of the LMP messages Bluetooth uses to create and configure a connection are given in table 5-3.

Table 5-3 LMP messages used during connection establishment

Message	Description
LMP_host_connect_req	Indicates a connection request from one device to the other.
LMP_accepted / LMP_not_accepted	Indicates whether or not the request was accepted.
LMP_setup_complete	Indicates that the setup is complete.

In addition to the above, table 5-4 gives the messages that we have for PKI authentication.

Table 5-4 Additional LMP messages added for PKI authentication.

Message	Description
LMP_pki_authenticate	Indicating a request to the connecting device to initiate public key authentication.
LMP_pki_authentication_rsp	A positive or negative response to LMP_pki_authenticate. This also initiates exchange of a series of messages as required by the authentication protocol.
LMP_pki_session_key	Indicating exchange of the session key used for encrypted communication.

As shown in figure 5-1, when A wishes to establish a secure connection with B, the following steps take place:

1. A sends LMP_connect_req to B whose parameters. B receives this request and sends back LMP_pki_authenticate asking A to authenticate itself and sends its own certificate, Cb to A.

2. If A can authenticate itself, it sends a positive LMP_pki_authentication_rsp along with a Nonce n1, its certificate Ca and a signature over n1 using A's private key. Otherwise it sends a negative LMP_pki_authentication_rsp.

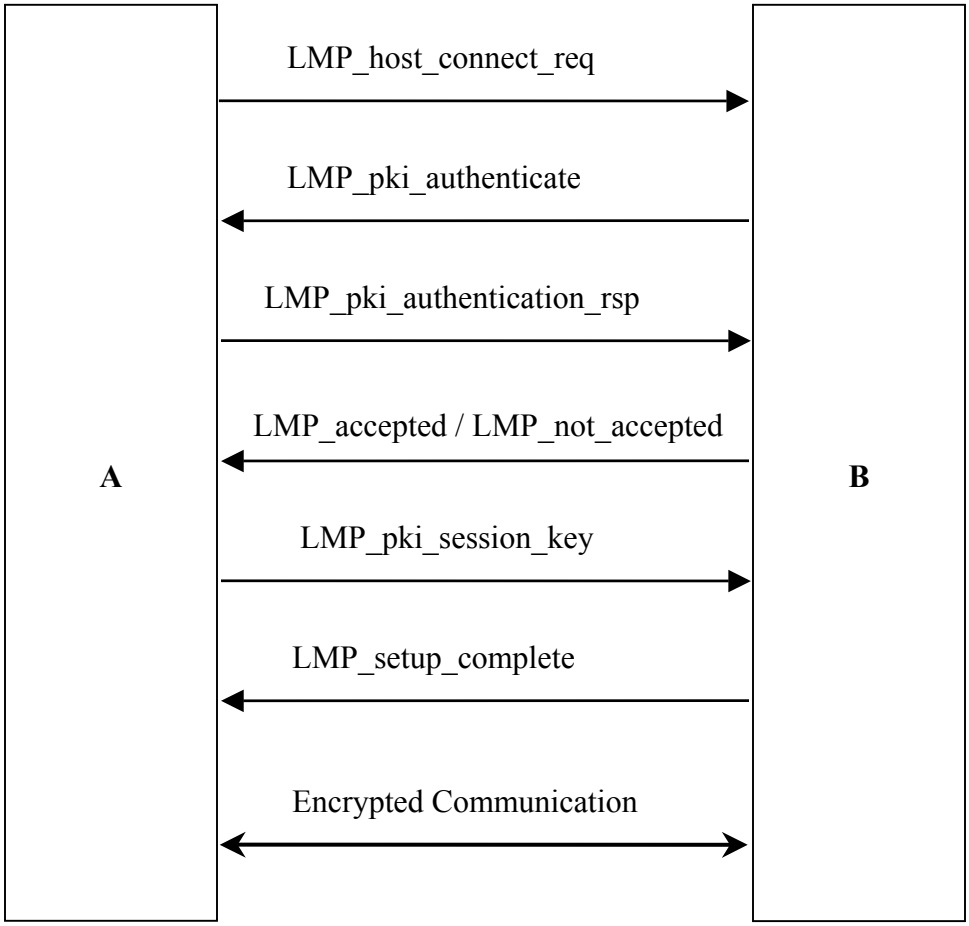


Figure 5-1 Fitting our protocol into Bluetooth

3. B verifies A's certificate, verifies A's signature (authenticate A), calculates a function f(n1) of the nonce n1 and signs it. Now if A has been authenticated successfully, B sends an LMP_accepted to A along with signed f(n1). If A has not been authenticated successfully, B sends an LMP_not_accepted to A and tears down the connection.

4. If A had received an LMP_pki_authenticate, it will optionally authenticate B. If B is authenticated successfully, it will send an LMP_pki_session_key to B signaling that the authentication is complete and this session key can be used to encrypt further communication. If B cannot be authenticated successfully, then it sends an LMP_disconnect and the connection is broken down.

A and B can now start their communication encrypted using the session key Ks.

5.5 Implementation Details

We have implemented the protocol on the Axis openBT Bluetooth stack on two laptops running Red Hat 7.2 Linux. We are using HCI_EMULATION (i.e., using the stack without Bluetooth hardware). This is being done by connecting the two Linux machines using a null modem cable. The openBT stack implements LMP at the Host Controller Interface (HCI) layer. As a result, we have added HCI events corresponding to the new LMP messages. This requires a change in the hardware because HCI is tightly coupled with the hardware. We have successfully tested our protocol by using HCI_EMULATION. However, hardware implementation of the protocol would considerably reduce the computation time.

5.5.1 Signature Algorithm

The major computationally intensive task in the protocol is message signing and verification. Hence, the choice of signature algorithm considerably affects the performance of the protocol. We use ESIGN digital signature algorithm because it is provably as secure as and more efficient than other signature schemes such as elliptic curve and RSA based schemes. Table 5-5 gives the computation time required for signature and verification by different schemes in terms of the amount of required modular operations in the number of 1024 bit modular multiplications [Okamoto, Fujisaki, Morita 1999].

Table 5-5 Number of operations required by different signature algorithms

Scheme	Signature Operations	Verification Operations
ESIGN	9	5
Elliptic curve based scheme	24	28
RSA based scheme	384	17

5.5.2 Key Distribution

It is assumed that both the communicating devices have the Certification authority's public key. This key can be downloaded to the device offline. This scheme requires only one key to be stored on the device as compared to one link key per device. Also, we have not used certificate chaining. Certificate chaining would require at least one connection outside the Bluetooth subnet. This can be achieved by using LAN access points as hot spots.

5.6 Performance Evaluation

We have implemented the power aware public key authentication protocol with Axis's openBT Bluetooth stack for Linux on Red Hat 7.2. Our test system consists of two IBM Pentium II laptops connected to each other by null modem cable. Hence, we are emulating the Bluetooth link. We have added our modules to the HCI layer because most of the LMP commands are implemented as HCI commands and events.

We have conditionally compiled the stack with and without our protocol. Table 5-6 summarizes the sizes of source files with and without our protocol. The object file(hci.o) size grew up from 58.4 KB to 73 KB as a result of adding the protocol. Table 5-7 summarizes the time taken to perform different security operations during the protocol using 256 bit keys within our setup.

Table 5-6 Code Size of different modules

Code	Size with authentication protocol	Size without authentication protocol
hci.c (HCI source file)	112.8 KB	126.2 KB
ESIGN library (External library)	0 KB	14.9 KB
Miscellaneous Functions	0 KB	12.1 KB

Table 5-7 Time taken for different security operations

Operation	Time taken (in secs)
Certificate Verification	0.010
Signature Verification (over nonce)	0.004
Session key encryption	0.015
Signature over nonce	0.004
Session key Decryption	0.012

The time taken by two devices to connect to each other in the worst case is between 10 and11 seconds. The critical path in our protocol consists of one Certificate Verification, one session key encryption and one signature verification on the device trying to connect, and, one Certificate Verification, one signature verification, one signature generation and one decryption on the device being connected to. Assuming time taken to send and receive messages to be negligible, the total overhead added by the protocol is 0.029 seconds on one device and 0.030 seconds on the other. Considering worst case connection times this is not a big overhead. The protocol can be made more efficient by a custom machine language implementation of the ESIGN library. This would also result in lesser storage space on the device. A hardware implementation would speed up the protocol considerably and make it viable for widespread implementation.

CHAPTER 6
CONCLUSION

This thesis is an attempt to gear Bluetooth wireless technology towards using Mobile Commerce applications and applications that need secure communication on devices that are not as powerful as modern day computers. This is done by making use of less intensive yet secure protocols and overlapping security operations on communicating devices. The work emphasizes the need for more open security architecture than just a weak device level security model that can be used only in limited environments like home networks. As there is a limit to the number of devices to which a Bluetooth device can connect, we have implemented a security protocol at link layer to avoid potential denial of service attacks. At the same time, we make use of user properties like user certificates to incorporate user authentication at link level. This also enables developers to create applications without worrying about the security procedures at the application level.

This new security model can be used as part of a broader specification for Bluetooth security. Vendors should be allowed to have their own choice of security protocols.

We have used the ESIGN library implementation provided by NTT, Japan. The protocol can be made more efficient by a custom machine language implementation of the ESIGN library. This would also result in lesser storage space on the device. A hardware implementation would speed up the protocol considerably. More value can be added by providing access control on the service level by implementing a service level security model.

Issues related to certificate chaining can be incorporated into the protocol. That would involve using computers in the vicinity with connection to LAN as “hot spots”. That would increase the overhead by a significant amount.

LIST OF REFERENCES

Bray J. and Sturman C. 2000. Connect without cables. Prentice Hall Professional, Upper Saddle River, NJ.

Hale D. 2001. Bluetooth leaves room for security. Rapport Technologies, San Jose, CA. Available at http://www.eetimes.com/story/OEG20010226S0080. Accessed on 12/24/2001.

Jacobson M. and Wetzel S. 2001. Security weaknesses in Bluetooth. Lucent Technologies – Bell Labs, Murray Hill, NJ.

Lamm G., Falauto G., Estrada J. and Gadiyaram J. 2001. Bluetooth wireless network security features. Proceedings of the IEEE Workshop on Information Assurance and Security, IEEE press, West Point, NY.

Miller B. and Bisdikian C. 2000. Bluetooth revealed: The insider's guide to an open specification for global wireless communications. Prentice Hall Professional, Upper Saddle River, NJ.

Muller N. 2000. Bluetooth demystified. McGraw-Hill Professional, Two Penn Plaza, NY.

Muller T. 1999. Bluetooth security, Proceedings of Bluetooth '99, London, UK.

Okamoto T. and Kobayashi T. 1999. On the security of EPOC and TSH-ESIGN. NTT Laboratories, Yokosuka-shi, Japan.

Okamoto T., Fujisaki E. and Morita H. 1999a. TSH-ESIGN: Efficient digital signature scheme: Using Trisection Size Hash. NTT Laboratories, Yokosuka-shi, Japan.

Okamoto T., Fujisaki E. and Morita H. 1999b. Self-evaluation of ESIGN. NTT Laboratories, Yokosuka-shi, Japan. Available at http://info.isl.ntt.co.jp/esign/nessie/ESIGN-eval-e.pdf. Accessed on 3/26/2002.

Schneier B. 1996. Applied cryptography. John Wiley & Sons, Ontario, Canada.

Specication of the Bluetooth system, Specication Volume 1, v.1.0B. 1999. Available at www.bluetooth.com/developer/specification/specification.asp. Accessed on 01/10/2002.

Stallings W. 1999. Cryptography and network security: Principles and Practice. Prentice Hall Professional, Upper Saddle River, NJ.

BIOGRAPHICAL SKETCH

Baldeep Anand was born in Indore, India, and completed his undergraduate studies (Bachelor of Engineering) in computer engineering from Shri G.S. Institute of Technology and Science, Indore.

He joined the University of Florida to pursue a master's degree in computer and information science and engineering. He has worked on his master's thesis under Dr. Sumi Helal. He has also worked as a graduate research assistant under Dr. Ronald Dalton in the Nuclear and Radiological Engineering Department.

He has experience working in the industry with Impetus Computing Systems, Indore, on his senior project during Fall 1999-Spring 2000. Currently, He is working with GE Power Systems, Atlanta, as an intern in Fall 2002.