Security Enhancement of Pairing and Authentication Process of Bluetooth

Md. Ariful Alam and Mohammad Ibrahim Khan

Department of Computer Science and Engineering
Chittagong University of Engineering and Technology
Chittagong-4349, Bangladesh

Summary
Security is a major concern in wireless communication. Like other wireless technologies Bluetooth is susceptible to different types of security threats. In this paper, we analyze Man-in-the-Middle attack on Bluetooth Secure Simple Pairing. Furthermore, we propose a modification to the Secure Simple Pairing to enhance the security of pairing and authentication process of Bluetooth.

Key words:
Bluetooth, Man-in-the-middle attack, Authentication, Secure Simple Pairing, Diffie-Hellman cryptography, Vernam Cipher

1. Introduction

Bluetooth is an open standard for short range radio frequency (RF) communication. It operates at 2.4 GHz frequency in the free ISM-band (Industrial, Scientific, and Medical) by using frequency hopping. Bluetooth devices that communicate with each other form a piconet. The device that initiates a connection is the piconet master. One piconet can have maximum of seven active slave devices and one master device.

Because Bluetooth is a wireless communication system, there is always a possibility that its transmissions could be deliberately jammed or intercepted, or false/altered information could be passed to the piconet devices. To provide protection for the piconet, the system can establish security at several protocol levels. Bluetooth has built-in security measures at the link level.

Our work mainly concentrates on the Man-In-The-Middle (MITM) attack. By principle, without any verification of the public keys, MITM attacks are generally possible against any message sent by using public-key technology.

In the next section of this paper, we will illustrate the Man-In-The-Middle attack on Secure Simple Pairing (SSP). Furthermore, a modification to the SSP has been proposed with a view to preventing Man-In-The-Middle attack thereby enhancing the security level of Bluetooth.

Manuscript received June 5, 2010
Manuscript revised June 20, 2010

2. Overview of Bluetooth Secure Simple Pairing

Bluetooth version 2.1+EDR adds a new specification for the pairing procedure, namely, Secure Simple Pairing [1]. Its main goal is to improve the security of pairing by providing protection against passive eavesdropping and MITM attacks. Instead of using (often short) passkeys as the only source of entropy for building the link keys, Secure Simple Pairing employs Elliptic Curve Diffie-Hellman public-key cryptography [2]. To construct the link key, devices use public-private key pairs, a number of nonces, and Bluetooth addresses of the devices. Passive eavesdropping is effectively thwarted by the Secure Simple Pairing, as running an exhaustive search on a private key with approximately 95 bits of entropy is currently considered to be infeasible in short time.

In order to provide protection against MITM attacks, Secure Simple Pairing either uses an Out-Of-Band (OOB) channel(e.g., Near Field Communication), or asks for the user's help: for example, when both devices have displays and keyboards, the user is asked to compare two six-digit numbers. Such a comparison can be also thought as an OOB channel which is not controlled by the MITM. If the values used in the pairing process have been tampered with by the MITM, the six-digit integrity checksums will differ with the probability of 0.999999.

Secure Simple Pairing uses four association models:

i. Out-of-Band
ii. Numeric comparison
iii. Passkey Entry
iv. Just Works

The Passkey Entry model is used in the cases when one device has input capability, but no screen that can display six digits. A six-digit checksum is shown to the user on the device that has output capability, and the user is asked to enter it on the device with input capability. The Passkey Entry model is also used if both devices have input, but no output capabilities. In this case the user chooses a 6-digit checksum and enters it in both

devices. Finally, if at least one of the devices has neither input nor output capability and an OOB cannot be used, the Just Works association model is used. In this model the user is not asked to perform any operations on numbers; instead, the device may simply ask the user to accept the connection.

The choice of the association model depending on the device capabilities is shown in Table 1. DisplayYesNo indicates that the device has a display and at least two buttons that are mapped to "yes" and "no": using the buttons the user can either accept the connection or decline it. Other notation in the table is self-explanatory.

Table 1: Device capabilities and simple pairing association models

Device 1	***Device 2***	***Association Model***
DisplayYesNo	DisplayYesNo	Numeric Comparison
	DisplayOnly	Numeric Comparison
	KeyboardOnly	Passkey Entry
	NoInNoOut	Just Works
DisplayOnly	DisplayOnly	Numeric Comparison
	KeyboardOnly	Passkey Entry
	NoInNoOut	Just Works
KeyboardOnly	KeyboardOnly	Passkey Entry
	NoInNoOut	Just Works
NoInNoOut	NoInNoOut	Just Works

Secure Simple Pairing is comprised of six phases:

1) Capabilities exchange: The devices that have never met before or want to perform re-pairing for some reason, first exchange their IO (Input/Output) capabilities (see Table 1) to determine the proper association model to be used.
2) Public key exchange: The devices generate their public private key pairs and send the public keys to each other. They also compute the Diffie-Hellman key.
3) Authentication stage 1: The protocol that is run at this stage depends on the association model. One of the goals of this stage is to ensure that there is no MITM in the communication between the devices. This is achieved by using a series of nonces, commitments to the nonces, and a final check of integrity checksums performed either through the OOB channel or with the help of user.
4) Authentication stage 2: The devices complete the exchange of values (public keys and nonces) and verify the integrity of them.
5) Authentication stage 2: The devices complete the exchange of values (public keys and nonces) and verify the integrity of them.
6) Link key calculation: The parties compute the link key using their Bluetooth addresses, the previously exchanged values and the Diffie-Hellman key constructed in phase 2.

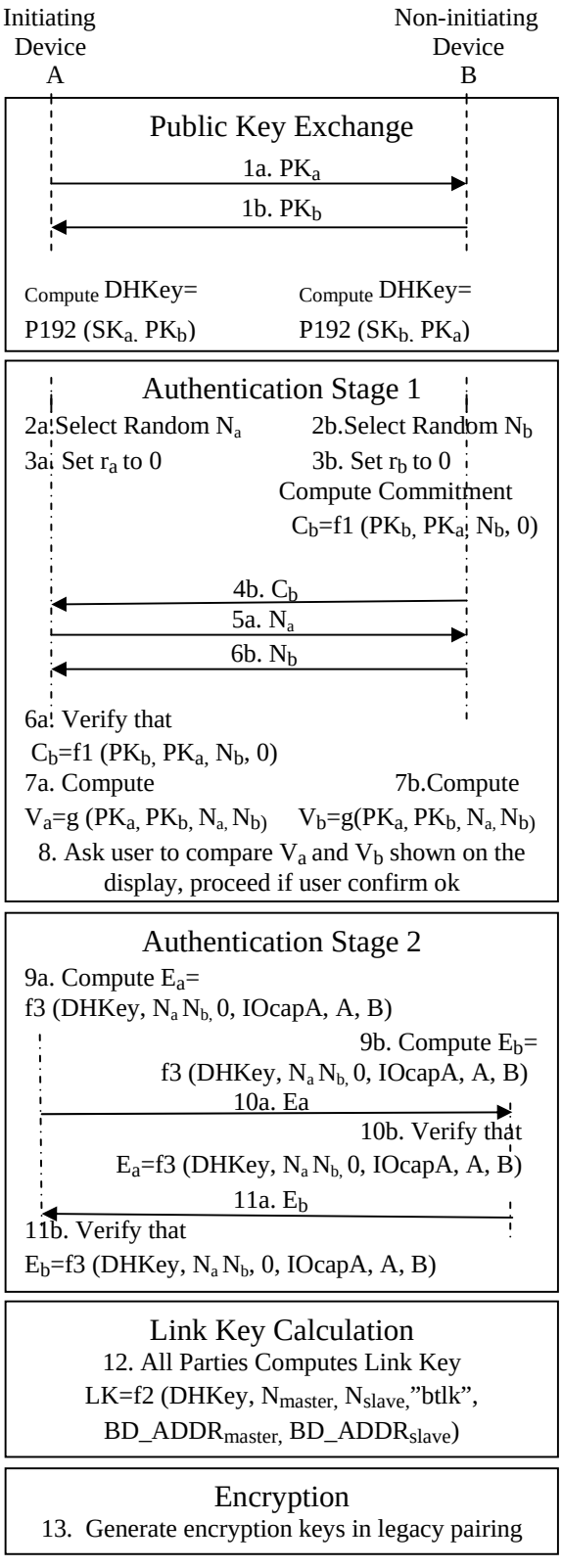


Fig. 1 Secure Simple P airing with Numeric Association Model

7) Authentication stage 2: The devices complete the exchange of values (public keys and nonces) and verify the integrity of them.
8) Link key calculation: The parties compute the link key using their Bluetooth addresses, the previously exchanged values and the Diffie-Hellman key constructed in phase 2.
9) LMP authentication and encryption: Encryption keys are generated in this phase, which is the same as the final steps of pairing in Bluetooth 2.0+EDR and earlier.

The contents of messages sent during the Secure Simple Pairing are outlined in Fig. 1, and used notations are explained in Table 2.

3. Man-In-The-Middle attack on Secure Simple Pairing

Althouh it was expected that Secure Simple Pairing would be able to prevent the Man-In-The-Middle (MITM) attack, unfortunately, it failed to meet this goal. It is possible to impose MITM attack on SSP [3]-[6]. In the attack we exploit the fact that the devices must exchange the information about their IO capabilities during the first phase of the Secure Simple Pairing.
The exchange is done over an unauthenticated channel, and an attacker that controls this channel can therefore modify the information about capabilities and force the devices to use the association model of his choice.

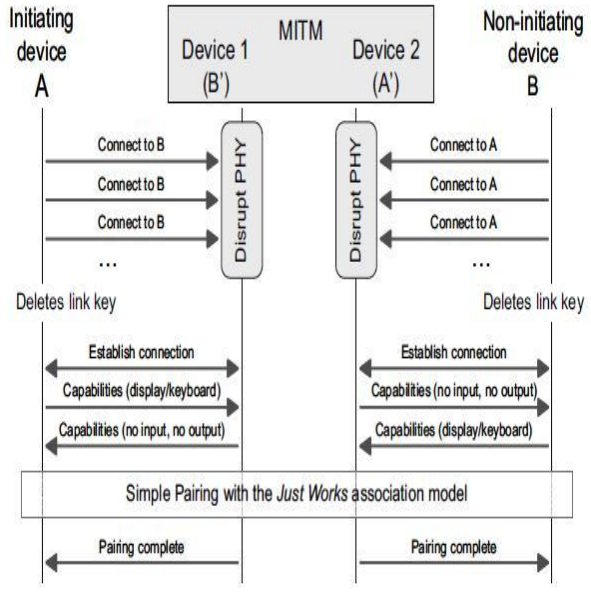


Fig.2 Main idea of the attack.

Devices are forced to use the Just Works association model, which does not provide protection against the MITM attack. The MITM uses two separate Bluetooth devices with adjustable BD ADDRs for the attack. Such devices are readily available on the market. The MITM clones the BD ADDRs and user-friendly names (1–248 bytes long user-defined strings describing the Bluetooth devices) of the victim devices, in order to impersonate them more plausibly.
The main idea of the attack is depicted in Fig. 2. In what follows, we describe three scenarios for the attack.
In the first scenario, the MITM first disrupts (jams) the PHY (physical layer) by hopping along with the victim devices and sending random data in every timeslot. Another possibility is to jam the entire 2.4 GHz band altogether by using a wideband signal. This way, the MITM shuts down all piconets within the range of susceptibility and there is no need to use a Bluetooth chipset to generate hopping patterns. Finally, a frustrated user thinks that something is wrong with his Bluetooth devices and deletes previously stored link keys. After that the user initiates a new pairing process by using Secure Simple Pairing, and the MITM can forge messages exchanged during the IO capabilities exchange phase. When the Just Works association model has been forced into use, the attack continues as illustrated in Fig. 3.

Table 2 : Protocol notation

Term	***Definition***
PKx	Public key of device X
SKx	Private key of device X
DHKey	Diffie-Hellman Key
Nx	Nonce generated by device X
rx	Random number generated by device X; equal to 0 in numeric comparison model
Cx	Commitment value from device X
f1	One way function used to compute commitmant value
f2	One way function used to compute the link key
f3	One way function used to compute check value
g	One way function used to compute numeric check value
IOcapX	Input/output capabilities of device
BD_ADDR	48-bit Bluetooth device address
fcrypt	Cryptographic Function Used to Compute Symmetric Key

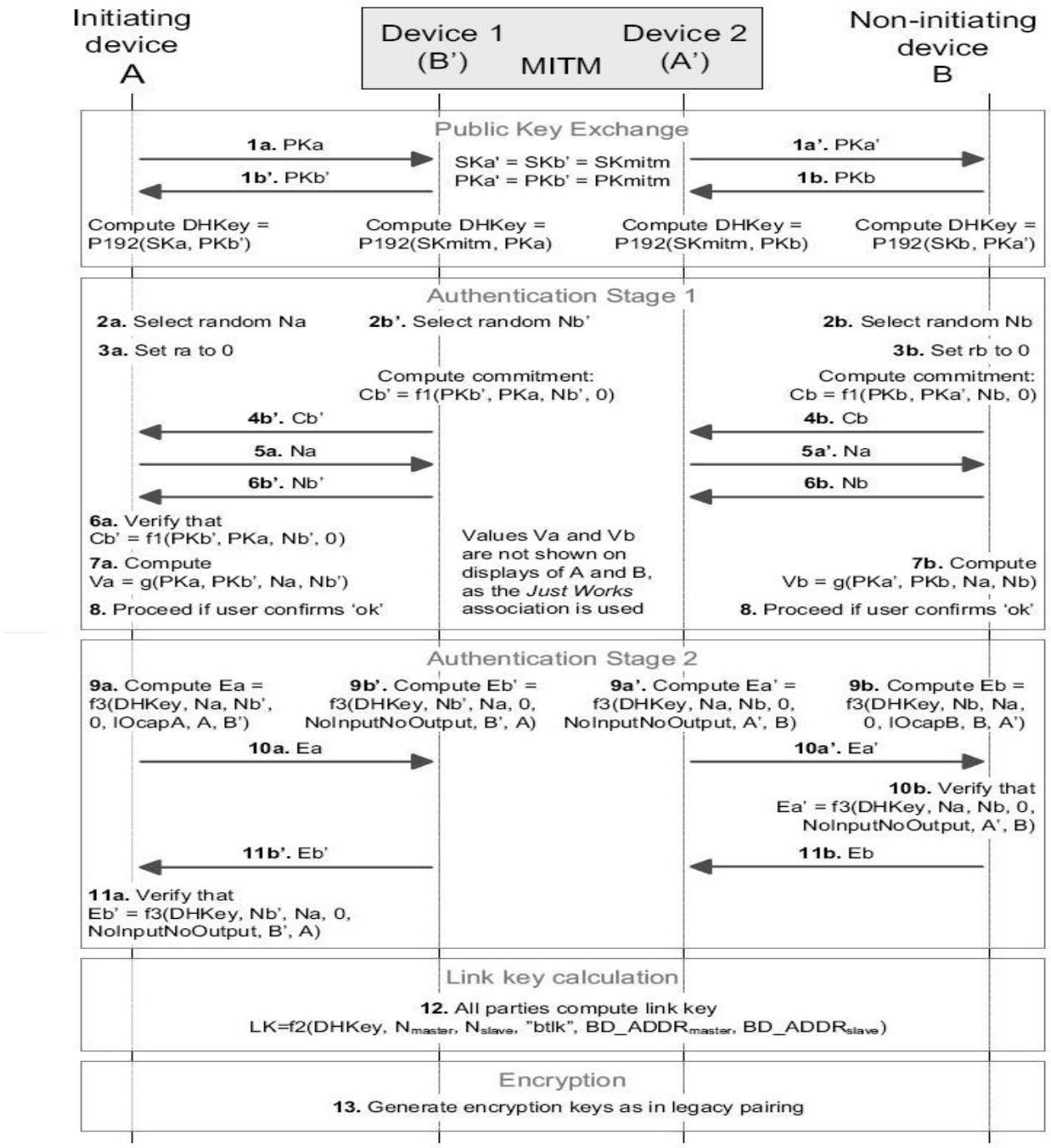


Fig 3: Pairing Details of Man-In-The-Middle Attack on Secure Simple Pairing

It is worth noting that in this first scenario two victim devices have already performed the initial pairing (including the capabilities exchange). Therefore, link keys are saved on the devices for use in subsequent connections, i.e. the victim devices normally use Secure Simple Pairing without capabilities exchange.

Other scenarios, where victim devices have never met before, are easier for the MITM, because in those cases

the first phase of the attack (disrupting the PHY) can be skipped.
There can be two different scenarios for this kind of devices:

1) The victim device (A or B) initiates Secure Simple Pairing: In this scenario, the MITM waits until A or B initiates Secure Simple Pairing. After that, the attack proceeds as illustrated in Fig. 2 and 3.
2) The MITM (A' and B') initiates Secure Simple Pairing. In this scenario, the MITM first initiates Secure Simple Pairing with the victim devices.

After that, the attack proceeds as illustrated in Fig. 2 and 3. Depending on the implementation of the victim devices, it may be possible to perform Secure Simple Pairing without asking the user to accept the connection. Depending on the situation, the MITM can use any of our three described attack scenarios. The applicability of a certain attack scenario obviously depends on the implementation of victim devices. After a successful attack, the MITM can intercept and modify all data exchanged between the victim devices, and even use certain services that victim devices offer.

4. Solution and Improvement

From the scenario shown in the Fig. 3, we have seen that the intruder inject his own public during public key exchange and after performing successive operations in a parallel with the victim devices he becomes successful to compute the link key that will be used later in authentication process. Once the intruder become successful to have the link key, he can repeatedly access the victim devices. This attack can be prevented with a little bit modification in the authentication stage 1 of the Secure Simple Pairing. As the master device (Device A in fig.) verify the commitment value of the slave device (Device B in fig.), it is possible to prevent the attack if we could secure the commitment value computed by the slave device (B). In this existing model we have seen that the intruder pushes his own commitment to overcome the verification by the master device (A).

We can encrypt the commitment value computed by slave based on Elliptic-curve Diffie-Hellman Cryptography [7] which is already employed in this Secure Simple Pairing. And the master will decrypt this commitment value before making verification. We can use symmetric key encryption process. And it is suggested to use 'Vernam Cipher' encryption process rather than DES or AES to encrypt the commitment value. Because of the use of symmetric key encryption as well as Vernam Cipher which required only to performed bitwise Exclusive-OR operation, it will not introduce any traffic overhead in the network. Encryption process is described as follows:

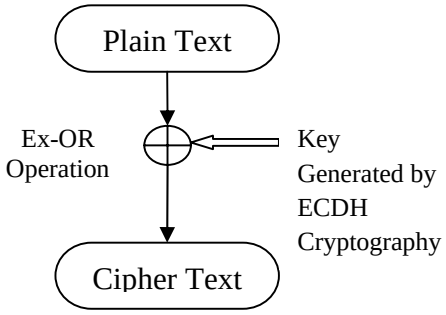
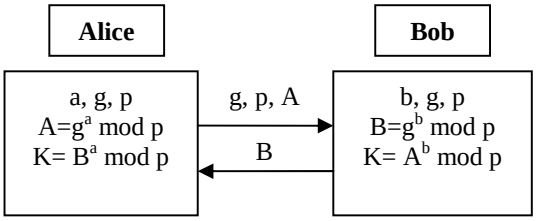


Fig. 4 Encryption Process with DHKey

4.1 Basic of Elliptic Curve Diffie-Hellman Cryptography

The Elliptic Curve Diffie-Hellman (ECDH) key agreement protocol enables two users to create a shared secret agreement. The ECDH protocol relies on two public parameters: p and g. Parameter p is a large prime number, and parameter g is an integer that is less than p. These two parameters are exchanged over a non-secure line. After both Alice and Bob receive the two public parameters, they select private integers. Alice chooses a, and Bob chooses b. These values are referred to as private keys. Alice and Bob then create public keys by using the public parameters and their private keys. Alice uses (g^a) mod p, and Bob uses (g^b) mod p. These are asymmetric keys because they do not match. Alice and Bob exchange these public keys and use them to compute their shared secret agreement. ECDH mathematics guarantees that both Alice and Bob will compute the same shared secret agreement, although they do not know each other's private keys.
ECDH mathematics can be given as follows:



$K=A^b \bmod p = (g^a \bmod p)^b \bmod p = (g^b \bmod p)^a \bmod p = B^a \bmod p$

Fig. 5 ECDH Mathematics

4.2 ECDH Cryptography in Secure Simple Pairing

Simple Pairing uses the FIPS P-192 curve [2]. Elliptic curves are specified by p, a, b and are of the form

$$E: y^2 = x^3 + ax + b \pmod p$$

For each value of b a unique curve can be developed. In NIST P-192:

a = mod (-3, p)

b is defined and its method of generation can be verified by using SHA-114 (with a given seed s and using b^2c= -27 (mod p))

The following parameters are given:

- The prime modulus p, order r, base point x-coordinate Gx, base point y-coordinate Gy
- The integers p and r are given in decimal form; bit strings and field elements are given in hex
  - p=6277101735386680763835789423207666416083908700390324961279
  - r=6277101735386680763835789423176059013767194773182842284081
  - b = 64210519 e59c80e7 0fa7e9ab 72243049 feb8deec c146b9b1
  - Gx = 188da80e b03090f6 7cbf20eb 43a18800 f4ff0afd 82ff1012
  - Gy = 07192b95 ffc8da78 631011ed 6b24cdd5 73f977a1 1e794811

The function P192( ) is defined as follows. Given an integer u, 0 < u < r, and a point V on the curve E, the value P192(u,V) is computed as the x-coordinate of the u-th multiple uV of the point V.

The private keys shall be between 1 and r/2, where r is the Order of the Abelian Group on the elliptic curve (e.g. between 1 and 2192/2).

Both master and slave device compute the DHKey using P192() function that takes private key of own and public key of other device as function arguments. We can use this DHKey to encrypt the commitment value. But unfortunately the intruder can also compute this DHKey as he has pushed his own public key earlier. This problem can be overcome using a simple secret cryptographic function with which two parties can compute a symmetric shared key using DHKey as function argument.

4.3 Modified Secure Simple Pairing

To verify users' authenticity the commitment value can be encrypted. As we have mentioned earlier that using DHKey does not ensure users' authenticity, two parties can use common secret cryptographic function to compute a symmetric key that will be used to encrypt the commitment value. Although the intruder might have the DHKey, he can not compute the symmetric key as he does not know the cryptographic function.

For example we assume that the DHKey computed using P192(SKx,PKx) is 9 and the cryptographic function is

$$f(x)= x^2 + x-5.$$

Then the symmetric key (SymKey) will be 85. The slave device will encrypt its commitment value with this key and master device will decrypt this value before verification. The encryption process now can be depicted as shown in Fig. 6.

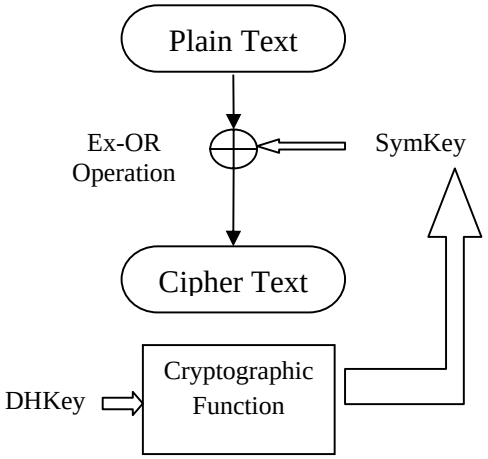


Fig. 6 Encryption process using symmetric key

If fcrypt is the secret cryptographic function then the modified Authentication Stage 1 of the Secure Simple Pairing can be illustrated as shown in Fig. 7.

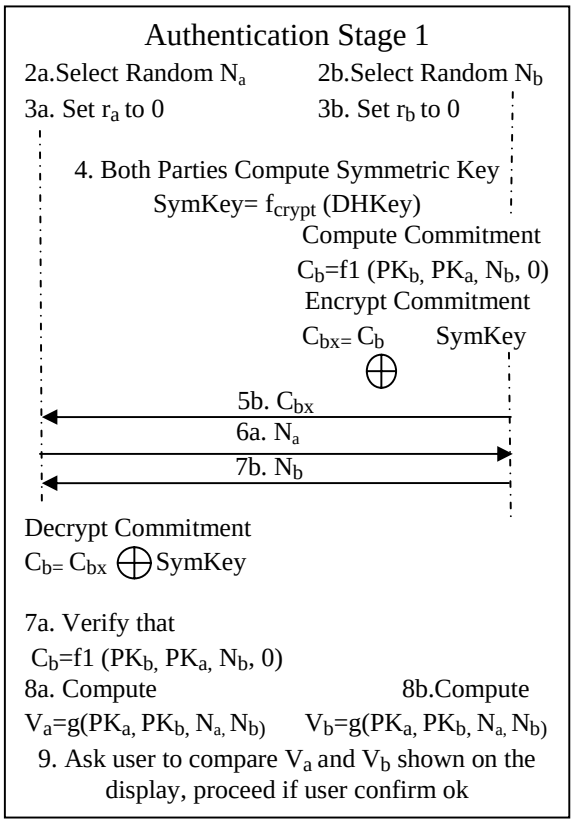


Fig. 7 Modified Authentication Stage 1 of Secure Simple Pairing

5. Conclusion

The emerging popularity of Bluetooth has inspired us to concentrate on its security. In this paper, we have focused on one of the prominent attacks namely, Man-In-The-Middle (MITM) attack that can be imposed even on the Secure Simple Pairing model. We analyze how the MITM attack is made on Secure Simple Pairing. Moreover, based on the Elliptic Curve Diffie-Hellman cryptography we have proposed a modification to the existing Secure Simple Pairing protocol to enhance the security level of pairing and authentication process of Bluetooth. As the modification is made only on one phase, the overhead will be considerably low. After simulation we firmly believe that this modification with very low overhead can eliminate the possibility of Man-In-The-Middle (MITM) attack on Bluetooth thereby enhancing the security level of pairing and authentication of Bluetooth.

References

[1] Konstantin Hypponen, Keijo M.J. Haataja: “Nino man-in-the-middle attack on bluetooth secure simple pairing”; Internet, 2007. ICI 2007. 3rd IEEE/IFIP International Conference in Central Asia on Digital Object Identifier: 10.1109/CANET.2007.4401672 Publication Year: 2007, Page(s): 1 – 5.
http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=4401653

[2] Bluetooth Special Interest Group-"Simple Pairing Whitepaper";
http://www.bluetooth.com/NR/rdonlyres/0A0B3F36-D15F-4470-85A6-F2CCFA26F70F/0/SimplePairing_WP_V10r00.pdf

[3] K.M.J. Haataja, K. Hypponen: “Man-In-The-Middle attacks on Bluetooth: a comparative analysis, a novel attack, and countermeasures”; Communications, Control and Signal Processing, 2008. ISCCSP 2008. 3rd International Symposium on Digital Object Identifier: 10.1109/ISCCSP.2008.4537388
Publication Year: 2008, Page(s): 1096 - 1102
http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=4531369

[4] K.Haataja, P.Toivanen:” Two practical man-in-the-middle attacks on Bluetooth secure simple pairing and countermeasures” ; Wireless Communications, IEEE Transactions on Volume: 9 , Issue: 1 Digital Object Identifier: 10.1109/TWC.2010.01.090935
Publication Year: 2010, Page(s): 384 - 392
http://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=7693

[5] D.Sharmila, R.Neelaveni, K.Kiruba: “Bluetooth Man-In-The-Middle attack based on Secure Simple Pairing using Out Of Band association model”; Control, Automation, Communication and Energy Conservation, 2009. INCACEC 2009. 2009 International Conference on
Publication Year: 2009, Page(s): 1 – 6
http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=5191366

[6] K.Haataja, P.Toivanen: “Practical Man-in-the-Middle Attacks Against Bluetooth Secure Simple Pairing”; Wireless Communications, Networking and Mobile Computing, 2008. WiCOM '08. 4th International Conference on Publication Year: 2008 , Page(s): 1 – 5
http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=4677908

[7] MSDN library-"Overview of the ECDH Algorithm(CNG Example)";
http://msdn.microsoft.com/en-us/library/cc488016.aspx