A Secure Protocol for Bluetooth Piconets Using Elliptic Curve Cryptography*

Min-Shiang Hwang† Cheng-Chi Lee‡ Ji-Zhe Lee† Chao-Chen Yang†

Department of Management Information System†
National Chung Hsing University
250 Kuo Kuang Road, Taichung, 402 Taiwan, R.O.C.
Email: mshwang@nchu.edu.tw

Department of Computer Science‡
National Chung Hsing University
250 Kuo Kuang Road, 402 Taichung, Taiwan, R.O.C.
and Department of Computer & Communication Engineering
Taichung Healthcare and Management University
No. 500, Lioufeng Raod, Wufeng Shiang, Taichung, Taiwan 413, R.O.C.

October 31, 2012

*This research was partially supported by the National Science Council, Taiwan, R.O.C., under contract no.: NSC91-2213-E-324-003.

†Responsible for correspondence: Prof. Min-Shiang Hwang.

A Secure Protocol for Bluetooth Piconets Using Elliptic Curve Cryptography

Abstract

In this article, the authors shall propose a new method for the implementation of secure Bluetooth piconets. Two requirements for the systems must be considered, i.e., privacy and authentication. Privacy ensures that an eavesdropper cannot intercept conversations between two slaves in piconets. Authentication ensures that service is not obtained fraudulently in order to avoid charge for usage. Additionally, a new key distribution scheme is designed for practical implementation in low-cost and low-power Bluetooth piconets. The proposed method employs elliptic curve cryptography for the use in the Bluetooth network. We have proper solutions to Bluetooth devices registration and Bluetooth piconets establishment. Furthermore, compared with Seo and Lee's protocol, the proposed scheme has a lower computation cost.

Keywords: Authentication, Bluetooth, Cryptosystems, Piconet, Security.

1 Introduction

Bluetooth is a wireless short-range communication technology. The goal is to replace linking wires with electronic devices. In fact, the main concept of Bluetooth is to control correlative pieces of equipment with a cell-phone in order to form a personal area network. The effective transmission range between Bluetooth devices is $10 \sim 100$ meters. The communication bandwidth is 2.4 GHz on the ISM (Industrial Scientific Medical) band. This band is the global, public, and non-statutory channel for radio transmission. The highest one-way transmission rate is 721 Kbps. Voices and other forms of data can be transmitted synchronously in Bluetooth.

To avoid the electromagnetic wave jamming problem, Bluetooth systems use the FHSS (Frequency Hopping Spread Spectrum) technology and divide the band into several hop channels, transmitting data among these hop channels pseudo-randomly. Such a method can reduce the electromagnetic jamming to its minimum. Bluetooth devices connect with each other in a point-to-point manner or point to multi-points. The basic network type is called the piconet. The originated piconet device is called the Master, and other devices linked to the Master are named Slaves. There is only one Master in a piconet; it connects to seven slaves top at the same time. Several piconets can be linked to one another to form a large-scale Bluetooth network called a scatternet. Relying on this connecting way, Bluetooth networks can engage with LAN, WAN or the global Internet [12, 13].

The Bluetooth security standards are provided in the Bluetooth system profile specification. In many norm profiles of this specification, the Generic access profile defines three Bluetooth security modes [1, 11].

1. Non-security: Devices will never initiate any security procedure. Supporting authentication is optional for devices. This mode is the lowest security grade.

2. Service-level-enforced security: This mode will take security measures when the logical link control adaptation protocol [11] is setting up. It is the medium security grade.

3. Link-level-enforced security: This mode will execute security operations when transmitting messages in the link manager protocol [11]. It is the highest security grade.

In recent decade, the demand for wireless communication access is proliferate. Protection of the connected resources for wireless communications becomes critical to prevent unauthorized accessing, intercepting and masquerading [2, 3]. The Bluetooth is a new and popular wireless communication technology. But the security weakness of the Bluetooth is needed to improve [6]. Designing a secure protocol to suit the Bluetooth Piconets is the research purpose of this paper. In the new scheme, we use the efficient elliptic curve cryptography (ECC) in order to adapt our system to the Bluetooth network. The 160-bit order of a prime point on the elliptic curve can equal the security level of RSA 1024-bit modulus [8]. Generally, the security of RSA cryptosystem is based on the hard problem in factoring two large primes, $n = p \times q$. The high security of RSA is generally set the length of n to 1024 bits. So ECC can be computed more rapidly on Bluetooth devices for wireless communication because its modulus is only 160 bits and its security level is equal to the RSA 1024-bit modulus [8].

In the next section, we shall review the ECC. Then, we shall offer our new protocol in Section 3. The security analysis will be discussed in Section 4, followed by the performance comparison, where the computation cost of our method will be compared with that of Seo and Lee's [10] scheme. Finally, conclusions are given in Section 6.

2 Review of Elliptic Curve Cryptography

The proposed scheme is based on the elliptic curve cryptography (ECC). Its security is based on the elliptic curve discrete logarithm problem (ECDLP). It is an efficient cryptographic system, suiting the highly changeable Bluetooth network. We explain ECDLP as follows [8, 9, 15]:

1. ECDLP: The problem is similar to the discrete logarithm problem (DLP) [5]. Let g denotes a generator and p is a large prime number. Consider the following equation $J = g^j \bmod p$. If we know g, j and p, it is ease to compute J. However, if we know g, J and p, it is very difficult to calculate j. The calculating j is called DLP. Next, ECDLP is that let q

is a large prime, the elliptic curve formula E is $y^2 = x^3 + ax + b(\mathrm{mod} q)$, $a, b \in GF(q)$, and $4a^3 + 27b^2 \neq 0(\mathrm{mod} q)$. Suppose we decide on one point $G \in E(Z_p)$ in E, as the public generation point, and q is its order. Then we choose a random number s_i such that $p_i = s_i \times G = (x_i, y_i)$. Then, s_i can be taken as a secret key of the elliptic curve cryptographic system, and p_i is the public key. Solving s_i is right ECDLP if p_i and G are known. To solve j and s_i, it is still an open problem.

2. ECC: In public key cryptosystem such as RSA, the pair of keys is secret key and public key. Due to the public key cryptosystem, users can encrypt/decrypt the messages over insecure networks. Assume that two parties, A and B, want to communicate in privacy. A can encrypt the messages using the public key of B and then sends it to B. After receiving these encrypted messages, B can decrypt it using B's secret key. Without knowing the B's secret key, no one can decrypt the encrypted messages. In the same way, B can do it in inverse. If A or B wants to prove a context is signed by himself/herself, he/she can sign the context using A's or B's secret key. Other party can verify the correctness of the digital signature using A's or B's public key. Next, we show ECC in Figure 1. The symbol $(A \rightarrow B : M)$ denotes A sends M to B.

   (a) A → B: $(lG, M + lp_B)$ – The sender A wants to encrypt the message M and deliver it to the receiver B. So A generates the value l and computes $l \cdot G$, and then encrypts the message M by computing $M + lp_B$. A transmits the message pair $(lG, M + lp_B)$ to B.

   (b) B receives the message pair from A and then decrypts it. B multiplies his/her secret key s_B by the value $l \cdot G$, and then B subtracts the said value from $M + lp_B$. The plaintext M is decrypted at last.

To solve l, s_A, and s_B, it is also infeasible to solve ECDLP.

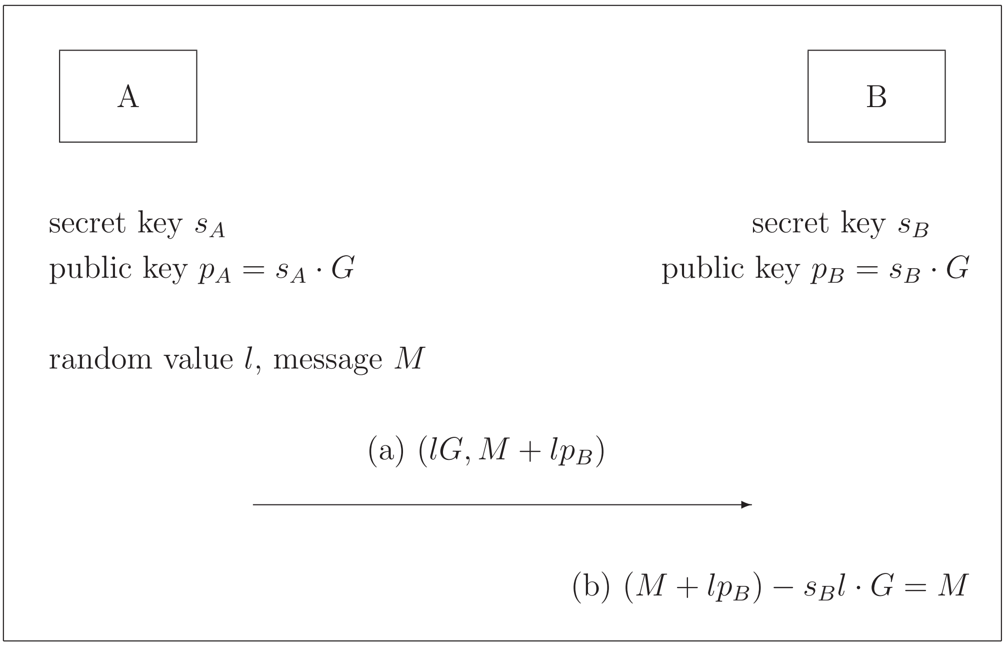
A
B
secret key s_A
public key $p_A = s_A \cdot G$
secret key s_B
public key $p_B = s_B \cdot G$
random value l, message M
(a) $(lG, M + lp_B)$
(b) $(M + lp_B) - s_B l \cdot G = M$

Figure 1: An elliptic curve cryptosystem

3. Elliptic curve Diffie-Hellman key exchange: Diffie-Hellman key exchange is the common problem on secure session key generation [4]. The Diffie-Hellman key exchange allows two parties communicate each other in a public communication with the agreed session key. Its security is based on solving DLP. Assume that Alice and Bob to agree on a session key over insecure networks. The parameters g and p are public. Then, they do the following steps to agree on a session key.

    - Alice randomly chooses a large number a and sends Bob $A = g^a \bmod p$.
    - In the meantime, Bob also randomly chooses a large number b and sends Alice $B = g^b \bmod p$.
    - After that, Alice and Bob can calculate their session key $K = B^a \bmod p = A^b \bmod p = g^{ab} \bmod p$.

    Without knowing a and b, no one has the ability to eavesdrop from the Alice-Bob channel. It is also a DLP to obtain a and b from knowing

K and B. The same thing happens to the elliptic curve. For example, suppose that the sender A wants to negotiate the session key with the receiver B. First, A and B produce their random integers s_A and s_B, and then they multiply their integers by the generation point G on the elliptic curve respectively. Next, A and B exchange the product with each other. The shared session key is then calculated by multiplying the product by the random integer of A or B. The session key is $SK = s_A \cdot s_B \cdot G$.

4. Elliptic curve digital signature algorithm: Assume the signer A signs the message M with his/her secret key s_A. Then, the verifier B uses A's public key p_A to verify M. Two phases are described as Figure 2.

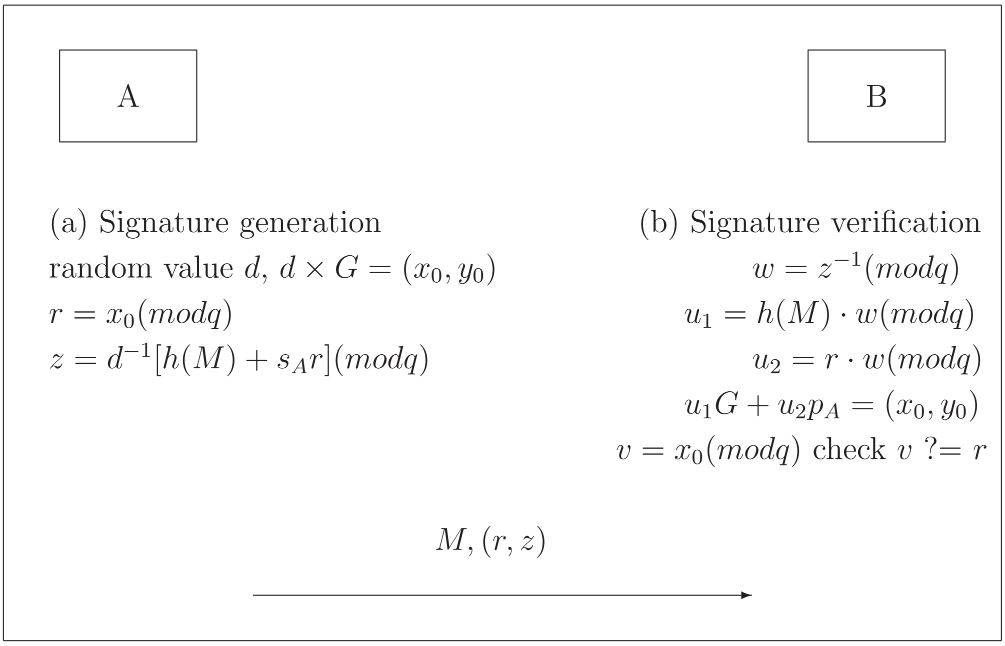
A
B
(a) Signature generation
random value d, $d \times G = (x_0, y_0)$
$r = x_0(modq)$
$z = d^{-1}[h(M) + s_A r](modq)$
(b) Signature verification
$w = z^{-1}(modq)$
$u_1 = h(M) \cdot w(modq)$
$u_2 = r \cdot w(modq)$
$u_1 G + u_2 p_A = (x_0, y_0)$
$v = x_0(modq)$ check v ?= r
$M, (r, z)$

Figure 2: An elliptic curve digital signature

(a) Signature generation: A generates a random number d such that $d \times G(\text{mod}q) = (x_0, y_0)$. Let $r = x_0(\text{mod}q)$, and $z = d^{-1}\{h(M) + s_A r\}(\text{mod}q)$, where $h(\cdot)$ is a one-way hash function. At last, A sends the signed information (r, z) and the message M to B.

(b) Signature verification: B receives the foregoing data and computes

$$\begin{aligned} w &= z^{-1} \bmod q, \\ u_1 &= h(M)w \bmod q, \\ &= h(M)z^{-1} \bmod q, \\ u_2 &= rw \bmod q, \\ &= rz^{-1} \bmod q. \end{aligned}$$

Then B computes $u_1G + u_2p_A \bmod q$ as follows:

$$\begin{aligned} u_1G + u_2p_A \bmod q &= h(M)z^{-1}G + rz^{-1}p_A \bmod q \\ &= h(M)z^{-1}G + rz^{-1}s_AG \bmod q \\ &= Gz^{-1}(h(M) + rs_A) \bmod q \\ &= Gd(h(M) + s_Ar)^{-1}(h(M) + rs_A) \bmod q \\ &= dG \bmod q \\ &= (x_0, y_0). \end{aligned}$$

and makes $v = x_0 (\mathrm{mod} q)$. If $v = r$, then the verification is passed.

We will show our scheme in the next section. The security of our protocol is the same as that of the above-mentioned theory based on ECDLP which is difficult to solve.

3 The Proposed Scheme

Our scheme is divided into two phases and described in the next two subsections. The first phase is the Bluetooth devices registration phase, where the devices must be registered with the fair certificate authority before leaving the factory. When Bluetooth devices have obtained the ECC public keys and secret keys, they enter the second phase. The main goal of the second phase is to construct the Bluetooth piconet. This stage includes the mutual authentication and session key generation of Bluetooth devices. Because Bluetooth

Table 1: The system parameters

Symbol	Meaning
p	a large prime
q	a factor of the large prime p
G	a generation point on the elliptic curve with order q
I_i	the identity information of i
P_i	the public key of i
s_i	the secret key of i
w_i	the witness of i
T_i	the time stamp of i
$X(\cdot)$	a function outputs the X-axis value of an elliptic curve point
$EC_{P_i}(\cdot)$	an elliptic curve encryption function using P_i
$ECDS_{s_i}(\cdot)$	an elliptic curve digital signature function using s_i
$h(\cdot)$	a public one-way hash function

devices must check the identities of one another before building up the piconet, the piconet managing device (Master) can command the linked devices (Slaves) via the same hop channels. For the sake of forwarding data to Slaves privately, Master must agree on the session key with each Slave. So Master and each Slave will hold different shared session keys. Before the illustration of our scheme, let's check out the system parameters in Table 1.

3.1 Bluetooth Devices Registration Phase

Before the first time Bluetooth devices (BD) are used, they must be registered with the certificate authority (CA). Assumed that a trusted third Certification Authority (CA) provides a public key certification service and issues a certificate to all users. The CA should be a trusted third organization. The CA plays the same role as the Government in the certificate issuing process which checks the person's identification to assure the identity of the unit. When a unit wants to get a certificate, he/she generates his/her own key pair, gives the public key as well as some proofs of his/her identification to CA. Bluetooth devices can get their public keys and secret keys after the registration. The figure and the depiction of this phase are as Figure 3 below. The symbol

$(A \rightarrow B : M)$ denotes A sends M to B.

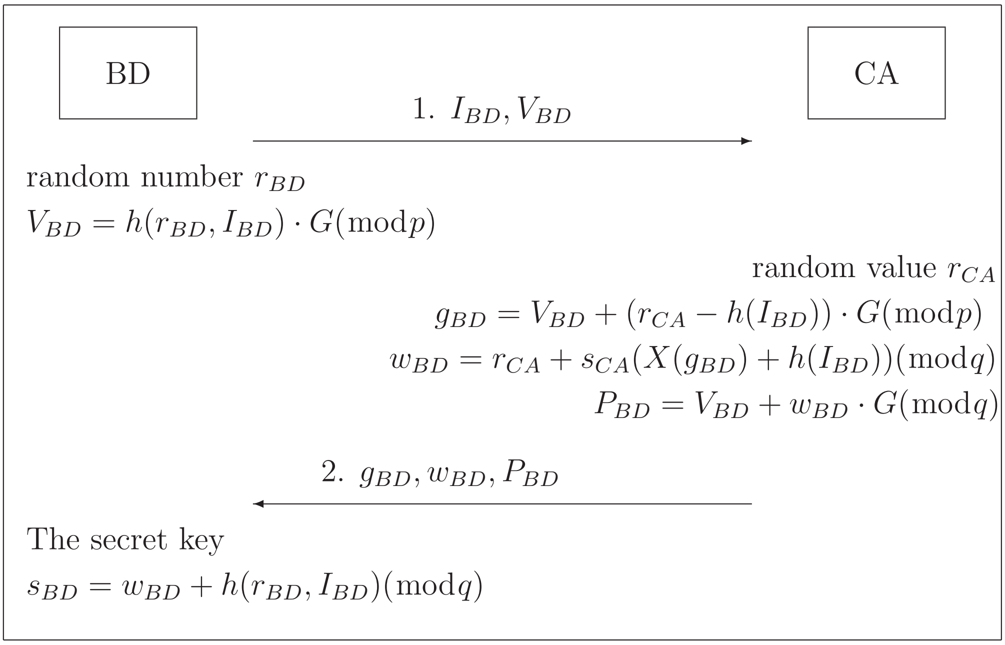
BD
CA
1. I_{BD}, V_{BD}
random number r_{BD}
$V_{BD} = h(r_{BD}, I_{BD}) \cdot G(\text{mod}p)$
random value r_{CA}
$g_{BD} = V_{BD} + (r_{CA} - h(I_{BD})) \cdot G(\text{mod}p)$
$w_{BD} = r_{CA} + s_{CA}(X(g_{BD}) + h(I_{BD}))(\text{mod}q)$
$P_{BD} = V_{BD} + w_{BD} \cdot G(\text{mod}q)$
2. g_{BD}, w_{BD}, P_{BD}
The secret key
$s_{BD} = w_{BD} + h(r_{BD}, I_{BD})(\text{mod}q)$

Figure 3: The Bluetooth devices registration phase of the proposed scheme

1. BD → CA: (I_{BD}, V_{BD}) – BD generates a random number r_{BD} and calculates $V_{BD} = h(r_{BD}, I_{BD}) \cdot G(\text{mod}p)$, and then BD sends it's identity information I_{BD} and V_{BD} to CA. After receiving these values from BD, CA keeps them to compute relative parameters.

2. CA → BD: (g_{BD}, w_{BD}, P_{BD}) – CA generates a random value r_{CA}. Then CA computes $g_{BD} = V_{BD} + (r_{CA} - h(I_{BD})) \cdot G(\text{mod}p)$, BD's public key witness $w_{BD} = r_{CA} + s_{CA}(X(g_{BD}) + h(I_{BD}))(\text{mod}q)$, and BD's public key $P_{BD} = V_{BD} + w_{BD} \cdot G(\text{mod}q)$. Here, CA can compute $h(I_{BD})$ because CA receives I_{BD} from BD in Step 1 and $h(\cdot)$ is a public one-way hash function. CA can also compute w_{BD} because CA knows r_{CA}, g_{BD}, I_{BD}, and his/her secret key s_{CA}. It means that CA signs the w_{BD} using his/her secret key s_{CA}. Afterwards, BD can verify the (g_{BD}, w_{BD}, P_{BD}) which are computed by CA using CA's public key P_{CA}. At last CA sends these parameters to BD and records them. BD can calculate its secret

key: $s_{BD} = w_{BD} + h(r_{BD}, I_{BD})(\bmod q)$ after receiving the messages from CA. Then BD can verify the validity of these parameters by the formula below:

$$P_{BD} \overset{?}{=} g_{BD} + h(I_{BD})G + [X(g_{BD}) + h(I_{BD})] \cdot P_{CA}. \tag{1}$$

The above equation is correct. We explain it as follows. Due to $g_{BD} = V_{BD} + (r_{CA} - h(I_{BD}))G(\text{mod} p)$, we can rewrite it to $g_{BD} + h(I_{BD})G = V_{BD} + r_{CA}G(\bmod p) = h(r_{BD}, I_{BD})G + r_{CA}G(\bmod p)$. And knowing $P_{CA} = s_{CA}G$, We can rewrite Equation 1 as follows.

$$\begin{aligned} P_{BD} &= g_{BD} + h(I_{BD})G + [X(g_{BD}) + h(I_{BD})] \cdot P_{CA} \\ &= h(r_{BD}, I_{BD})G + r_{CA}G + [X(g_{BD}) + h(I_{BD})] \cdot s_{CA}G \\ &= [r_{CA} + s_{CA}(X(g_{BD}) + h(I_{BD}))]G + h(r_{BD}, I_{BD})G \\ &= w_{BD}G + h(r_{BD}, I_{BD})G \\ &= (w_{BD} + h(r_{BD}, I_{BD}))G \\ &= s_{BD} \cdot G. \end{aligned}$$

After Bluetooth units register in the certificate authority, these units can build the connection one another. Once BDs have secret key s_i and public key P_i, they can do mutual authentication and agree a common session key to encrypt/decrypt transmitted messages.

3.2 Bluetooth Piconet Establishment Phase

BDs can be linked to one another and develop the piconet after being registered with CA. Several piconets can also be connected up to form a scatternet. That is to say, scatternet is a large-scale Bluetooth network that is several piconets can be linked to one another. Before the piconet is set up, Bluetooth devices must authenticate each other. Then the piconet Master can verify the identities of the Slaves and begin to negotiate the session key with each Slave. So, Master and each Slave can transmit data confidentially by the shared session key in the piconet. This phase has two parts as follows:

1. Mutual authentication: Bluetooth Master and Slave must verify each other to form the piconet. The Figure 4 and the descriptions are as below:

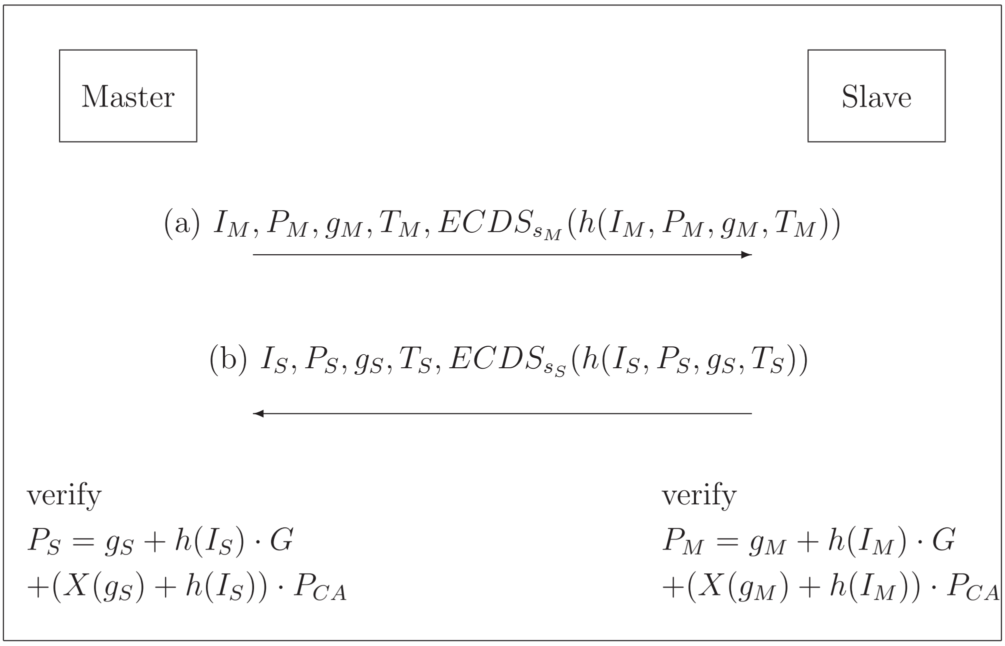
Master
Slave
(a) $I_M, P_M, g_M, T_M, ECDS_{s_M}(h(I_M, P_M, g_M, T_M))$
(b) $I_S, P_S, g_S, T_S, ECDS_{s_S}(h(I_S, P_S, g_S, T_S))$
verify
$P_S = g_S + h(I_S) \cdot G$
$+(X(g_S) + h(I_S)) \cdot P_{CA}$
verify
$P_M = g_M + h(I_M) \cdot G$
$+(X(g_M) + h(I_M)) \cdot P_{CA}$

Figure 4: The mutual authentication

   (a) Master → Slave: $(I_M, P_M, g_M, T_M, ECDS_{s_M}(h(I_M, P_M, g_M, T_M)))$ – Master inputs its identity information, public key and time stamp to the hash function. Then Master gets the output value from the hash function to make the elliptic curve digital signature. To compute $ECDS_{s_M}(h(I_M, P_M, g_M, T_M))$, Master should use the elliptic curve digital signature algorithm introduced in Section 2 applying his/her secret key s_M. Master sends the above values eventually. Slave verifies the validity of these messages after receiving them.

   (b) Slave → Master: $(I_S, P_S, g_S, T_S, ECDS_{s_S}(h(I_S, P_S, g_S, T_S)))$ – Slave also takes the same steps Master does. Master verifies the received parameters.

(c) Verification: Master and Slave can enter the verification phase after exchanging the above-mentioned values. The verification equation is expressed as below. Master verifies the first equation using CA's public key P_{CA}, and Slave checks the second one using CA's public key P_{CA} as follows:

$$\begin{cases} P_S = g_S + h(I_S) \cdot G + (X(g_S) + h(I_S)) \cdot P_{CA} \\ P_M = g_M + h(I_M) \cdot G + (X(g_M) + h(I_M)) \cdot P_{CA} \end{cases}$$

2. Session key generation: This part is for the mutual authentication between Master and Slave. Bluetooth Master will agree on the session key with each Slave in the piconet. Master will produce different session keys for different Slaves. The process is depicted in the Figure 5 as follows:

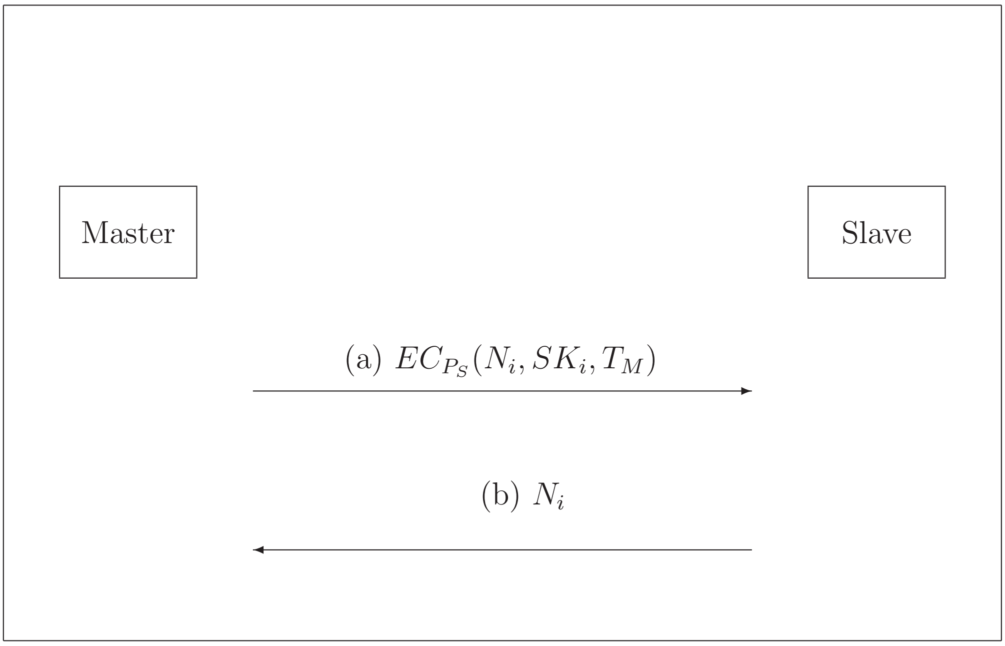

Figure 5: The session key generation

(a) Master → Slave: $EC_{P_S}(N_i, SK_i, T_M)$ – Master delivers a randomly chosen nonce N_i, the session key SK_i, and a time stamp T_M to Slave. Master inputs these values to the elliptic curve encryption function with Slave's public key and then sends the encrypted data to Slave. Slave verifies the validity of the received information in the end. So Slave can get the session key from Master to transmit information privately.

(b) Slave → Master: N_i – Slave finishes the above-mentioned steps and then responds to Master with the nonce. Master can check the nonce to see if Slave has received the data successfully. Then the establishment of the piconet between Master and this Slave is complete.

Bluetooth units can link one another to form the piconet or scatternet safely and rapidly via the above phases. Next we shall analysis the security of our scheme.

4 Security Analysis

The proposed scheme is analyzed as below:

1. No forging and replay attack: Our scheme creates the signature to prevent forging attacks on BDs. Only the legal deliverer can make the elliptic curve digital signature with his/her secret key, which the receiver can verify successfully. Any attacker cannot forge the signature because he/she cannot derive the secret key of the deliverer. The attacker will face ECDLP. This problem is very difficult to solve, so the attacker will fail [8, 14]. On the other hand, our scheme prevents the replay attack with the time stamp. The time stamp can effectively stop any message replaying. Bluetooth Masters and Slaves can check the validity of the identities and public keys via CA's public key. Furthermore, in mutual authentication of Section 3.2, no one can forge a digital signature without knowing s_i and Master or Slave cannot forge and deny transmitted messages I_i, P_i, g_i, T_i because they are signed by their secret key s_i.

2. Attacker cannot get the secret keys of BDs and the session key in the piconet: The attacker cannot derive the secret keys of BDs from the known messages. To gain the session key also means to solve ECDLP. So, the attacker still fails to crack our scheme.

3. CA cannot derive the secret keys of BDs and forge BDs: CA does not know the value $h(r_{BD}, I_{BD})$ because this value is multiplied the elliptic curve generation point. To get this value, means to solve ECDLP introduced in Section 2. So, CA cannot forge the BDs, for the secret keys cannot be derived. Although CA is the trusted third party, it isn't always just. CA is an artificial organization; if someone else attempts to trick the Bluetooth device holder, then the system might be broken. CA can deny his/her signed messages because he/she has two pairs of keys, (s_{CA}, P_{CA}) and (s'_{CA}, P'_{CA}). One pair of keys is used to sign messages and other pair of keys is used to deny his/her signed messages. Most public key cryptosystem need a CA to issue a public key certificate. Therefore, they cannot shirk this shortcoming. This is also our disadvantage.

4. In Bluetooth Devices Registration Phase, an attacker may intercept the values (g_{BD}, w_{BD}, P_{BD}). These values are public. Therefore, an attacker can derive any secret values. However, he/she may try to forge another legal values $(g'_{BD}, w'_{BD}, P'_{BD})$. However, no one knows CA's secret key, he/she cannot forge them. Furthermore, he/she does not know r_{BD}, he/she cannot derive the BD's secret key s_{BD}.

As we have shown, our protocol can resist all the above-mentioned attacks. Owing to the difficulty of solving ECDLP, the attacker cannot break our scheme, even through CA.

5 Performance Comparison

In this section, we shall compare the performance of our scheme with that of Seo and Lee's scheme phase by phase [10]. We briefly review their protocol first, and there are three portions in the follows. Some notations must be defined.

- $E_k(\cdot)$: the symmetric encryption algorithm with key k.
- (R_i, S_i): the elliptic curve digital signature pair of i.

Two subsections of their protocol shall be described next.

5.1 Review of Seo and Lee's protocol

The procedures of the mutual authentication phase of Bluetooth devices is listed as follows. This phase is the first stage of the piconet establishment.

1. Master → Slave: (V_1) – Master produces a random number r_M, then r_M multiplies the elliptic curve generation point G. Taking the result Q_M to link the identity of Master ID_M, and Master can hash ε_M. Last Master encrypts the value $\varepsilon_M || h_M$ and sends the result V_1 to Slave.
2. Slave → Master: (V_2) – Slave also can calculate the V_2 by the same way of Master done, and Slave transmits it to Master.
3. Session key generation: $(SK = r_S \times r_M \times G)$ – Bluetooth units shall get the session key by multiplying random number r of opposition and the generation point G.

Next is that exchange of signature and setting up group key for Stage 2. The notation ↔ between two entities indicates that the left-hand side and right-hand side entity send things to each other.

1. Master → Slave: (V_3) – Master produces a group key GK_S for Slave and makes the elliptic curve digital signature (ECDS) of the group key,

then Master connects these values to α_M. Group key $GK_S = EP \times N_S$, where EP is a random elliptic curve point and N_S is the device number of Slave. Master inputs α_M to the hash function and takes the result to connect itself, then Master encrypts these value by the session key as $V_3 = E_{SK}((\alpha_M||h_M))$. The Master delivers the V_3 to Slave.

2. Slave → Master: (V_4) – Slave also can count the V_4 by the same way of Master done, and Slave transfers it to Master.

3. Master ↔ Slave: Finish – This finish message denotes the end of Stage 2, and the two parties have set up the trust of each other.

Suppose that two Masters which are M_A and M_B belong to each piconet, and one Slave of M_B is called S_B. The following is Stage 3 that S_B wants to request for the connection of M_A. This situation is when M_A and M_B have not authenticated each other yet. If two Masters have authenticated, the Slave of one Master to link the other Master should proceed the above two stages.

1. $S_B \rightarrow M_A$: (ID_{S_B}, SK) – S_B dispatches the identity ID_{S_B}, the session key of S_B and M_B to M_A.

2. $M_A \rightarrow M_B$: (V_{M_A}) – M_A combines the identity ID_{M_A}, the signature pair of ID_{M_A}, and the time stamp T_{M_A} as β_{M_A}. M_A encrypts β_{M_A} and hash value of itself by the session key SK, then M_A forwards the result V_{M_A} to M_B.

3. $M_B \rightarrow M_A$: (V_{M_B}) – M_B makes the ECDS of ID_{S_B} and combines it with the signature pairs of S_B, then M_B attaches T_{M_B} to input the hash function. Last, M_B encrypts the information as $V_{M_B} = EC_{P_{M_A}}(\beta_{M_B}||h_{M_B})$ and sends the result to M_A.

4. $S_B \rightarrow M_A$: (V_{S_B}) – S_B does as the same thing as M_B has done, and S_B also delivers the outcome to M_A.

5. $M_A \rightarrow M_B$ and S_B: (Finish) – M_A sends the event finish message to M_B and S_B.

5.2 The Contrastive Results

The comparison includes two parts: computation cost and communication cost.

1. Computation cost: Given that a 160-bit prime p on the elliptic curve is equal to a 1024-bit modulus n of RSA. We define the notations for the computation cost comparison as follows.

   - T_H: the time for a hash function computation.
   - T_{INV}: the time for finding the inverse.
   - T_{MUL}: the time for a 1024-bit modular multiplication.
   - T_{EXP}: the time for the modular exponentiation with 1024-bits modulus.
   - T_{EC-MUL}: the time for an elliptic curve multiplication with 160-bit multiplier.
   - T_{SYM}: the time for a symmetric key cryptography

   Therefore we know $T_{EXP} \approx 240T_{MUL}, T_{EC-MUL} \approx 29T_{MUL}$, and T_{EXP} is the longest of all the above-mentioned time units. The next is T_{EC-MUL} [8]. The computation cost of the Seo-Lee scheme and that of our scheme can be compared. First, we will see how their mutual authentication and session key generation phases are compared. Then, we shall show the total comparison between the two schemes.

   (a) Part comparison: Here we contrast the Seo-Lee scheme with our scheme in part. This comparison is under the Seo-Lee model. We

Table 2: Mutual authentication and session key agreement

Method	Computation cost
Seo-Lee scheme	$12T_H + 17T_{EC-MUL} + 8T_{MUL} + 4T_{INV} + 2T_{SYM}$
Our scheme	$10T_H + 13T_{EC-MUL} + 8T_{MUL} + 4T_{INV}$

take the Stages 1 and 2 of Seo-Lee scheme to compare the computation cost with ours in mutual authentication. The results are in the tables below.

In Table 2, we can obtain that our scheme spends less elliptic curve multiplication time than the Seo-Lee scheme. The reason is that the Seo-Lee scheme uses elliptic curve public key cryptography and elliptic curve digital signature while ours just uses the latter. That is why we can reduce the computation cost in this stage. However, the elliptic curve multiplication time is the longest time above.

Next, we analyze the Seo-Lee scheme. In Step 1 of Stage 1, Master calculates and sends V_1 to Slave, then Slave verifies this value. So there are $2T_H + 4T_{EC-MUL}$ in this step and so is Step 2. Adding the computation cost of session key generation $2T_{EC-MUL}$, the total cost is $4T_H + 10T_{EC-MUL}$. In Stage 2 of Seo-Lee scheme, owing to the time of ECDS is $2T_H + 3T_{EC-MUL} + 4T_{MUL} + 2T_{INV}$ we can get the time of Step 1 between Master and Slave is $4T_H + 4T_{EC-MUL} + 4T_{MUL} + 2T_{INV} + 1T_{SYM}$. The cost of Step 2 is $4T_H + 3T_{EC-MUL} + 4T_{MUL} + 2T_{INV} + 1T_{SYM}$, so we can gain the total cost is $8T_H + 7T_{EC-MUL} + 8T_{MUL} + 4T_{INV} + 2T_{SYM}$. Therefore the total time of Stages 1 and 2 is as the show of Table 2. Then we dissect the computation cost of our scheme. In Figure 4, we can fetch the cost is $10T_H + 10T_{EC-MUL} + 8T_{MUL} + 4T_{INV}$. Due to the time is $3T_{EC-MUL}$ in Figure 5, so we can obtain the total computation cost as data in Table 2.

Table 3: No authentication between Masters

Method	Computation cost
Seo-Lee scheme	$22T_H + 29T_{EC-MUL} + 16T_{MUL} + 8T_{INV} + 3T_{SYM}$
Our scheme	$20T_H + 26T_{EC-MUL} + 16T_{MUL} + 8T_{INV}$

Table 4: Authentication between Masters

Method	Computation cost
Seo-Lee scheme	$36T_H + 51T_{EC-MUL} + 24T_{MUL} + 12T_{INV} + 6T_{SYM}$
Our scheme	$30T_H + 39T_{EC-MUL} + 24T_{MUL} + 12T_{INV}$

(b) Total comparison: Here is the total computation cost comparison between the Seo-Lee scheme and ours. Because the analysis is under the Seo-Lee model, we have to discuss two situations. One situation is no authentication between Masters, meaning that one slave in a piconet will be connected to another piconet. The Masters of the two piconets have no mutual authentication in this case. The other situation is the opposite case. The Slave of one piconet who wants to link to the other Master has to ask his/her own Master if the two Masters have authenticated each other yet. If no, then it proceeds with the no authentication case; Otherwise, it proceeds with the opposite case. We present comparison results as Tables 3 and 4 below:

Let us see the analysis of Seo-Lee scheme, we can understand the case in Table 3 is composed of steps of all stages. In the above-mentioned words, we can get the computation cost in Stages 1 and 2. Because the cost of Stage 3 is $10T_H + 12T_{EC-MUL} + 8T_{MUL} + 4T_{INV} + 1T_{SYM}$, so we adds the cost $12T_H + 17T_{EC-MUL} + 8T_{MUL} + 4T_{INV} + 2T_{SYM}$ of Stages 1 and 2 then the total cost is the same in Table 3. The opposite situation is that Masters are authenticated each other, and we can get the data of this case by computing the cost of Stages 1 and 2 for three times. So, the cost is $12T_H +$

Table 5: No authentication between Masters

Method	Communication cost
Seo-Lee scheme	$12m$
Our scheme	$8m$

Table 6: Authentication between Masters

Method	Communication cost
Seo-Lee scheme	$18m$
Our scheme	$12m$

$17T_{EC-MUL}+8T_{MUL}+4T_{INV}+2T_{SYM}$ to multiply 3 and the result is as data in Table 4.

As regards our scheme, we compute the cost in Figures 4 and 5 twice for the case of no authentication between Masters. We similarly calculate the cost in Figures 4 and 5 three times for the case of authentication between Masters. So we can obtain the cost in Figures 4 and 5 is $10T_H+13T_{EC-MUL}+8T_{MUL}+4T_{INV}$ which multiply 2 and 3 as data in Tables 3 and 4, respectively.

From Tables 3 and 4, we discover that our scheme has less computation cost than the Seo-Lee scheme especially on T_{EC-MUL} in the latter case because the authentication procedure of the Seo-Lee scheme takes too much time. Such a slow pace cannot catch up with the fast data exchange in Bluetooth. So, our scheme has better performance than theirs.

2. Communication cost: Before computing the transmission message time, let's define m first. The m represents the forwarding time during the communication. If one sends a message to the other, we would count $1m$ for this situation. The result of comparison is as below.

   We can see the data in Table 5, the forwarding time of Seo-Lee scheme are total 12 times in all stages. As to the total transferring time in

Figures 4 and 5 is 4 times, but we should multiply 2 for the case of no authentication between Masters. So we get the communication cost is $8m$ in our scheme.

In Table 6, we can analyze the data in the same way. Owing to this situation is that authentication between Masters, so we calculate the forwarding time in Stages 1 and 2 three times. Hence the cost of Seo-Lee scheme is $18m$, and our cost in Figures 4 and 5 should multiply 3 to get $12m$.

In Tables 5 and 6, we also have less communication cost, especially in the latter case. This is also because the authentication procedure of the Seo-Lee scheme is so complicated that their protocol consumes more communication cost. So, our scheme proves to be more efficient in terms of communication cost.

We showed the detail analysis in this section. Due to the above-mentioned data, we can prove that our scheme has better performance than Seo-Lee scheme.

6 Conclusions

Our scheme uses the CA registration protocol to enable Bluetooth devices to do mutual authentication. The high reliability perfectly fits variable Bluetooth network infrastructures. The elliptic curve cryptographic system lowers the computation cost for Bluetooth devices. As a result, the same security requirement can be attained more rapidly and effectively when the Bluetooth devices continuously expand the scope of their network.

Compared with Seo and Lee's protocol [10], our scheme consumes less computation cost. At the same time, our communication cost is $4m$ less than Seo's while there is no authentication between Masters, and the cost is $6m$ less than Seo's while authentication is done. Therefore the proposed scheme is more efficient than Seo and Lee's scheme.

References

[1] Jennifer Bray and Charles F Sturman, *Bluetooth-Connect Without Cables.* N.J.: Prentice-Hall, 2001.

[2] Ching-Wen Chen, Ming-Chin Chuang, and Chou-Chen Yang, "An efficient authentication scheme between MANET and WLAN on IPv6 based Internet," *International Journal of Network Security*, vol. 1, no. 1, pp. 12–18, 2005.

[3] Song-Kong Chong, Hsien-Chu Wu, and Min-Shiang Hwang, "A scheme for key management on alternate temporal key hash," *International Journal of Network Security*, vol. 1, no. 1, pp. 7–11, 2005.

[4] W. Diffie and M. Hellman, "New direction in cryptography," *IEEE Transactions on Information Theory*, vol. IT-22, no. 6, pp. 472–492, 1976.

[5] Min-Shiang Hwang, Chin-Chen Chang, and Kuo-Feng Hwang, "An ElGamal-like cryptosystem for enciphering large messages," *IEEE Transactions on Knowledge and Data Engineering*, vol. 14, no. 2, pp. 445–446, 2002.

[6] M. Jacobson and S. Wetzel, "Security weaknesses in Bluetooth," in *RSA Conference 2001*, pp. 179–191, San Francisco, 2001.

[7] Neal Koblitz, "Elliptic curve cryptosystems," *Mathematics of Computation*, vol. 48, pp. 203–209, 1987.

[8] Neal Koblitz, Alfred Menezes, and Scott A. Vanstone, "The state of elliptic curve cryptography," *Designs, Codes and Cryptography*, vol. 9, no. 2/3, pp. 173–193, 2000.

[9] V. Miller, “Use of elliptic curves in cryptography,” in *Advances in Cryptology, CRYPTO’85*, pp. 417–426, 1985.

[10] Dae Hee Seo and Im Yong Lee, “Bluetooth piconet using non-anonymous group key,” in *Proceedings of The 4th International Conference on Advanced Communication Technology*, pp. 883–888, Korea, 2002.

[11] Bluetooth SIG. “Bluetooth security,”, 2001. Bluetooth Specification Version 1.1.

[12] Bluetooth SIG. “Specification of the Bluetooth system-core,”, 2001. Bluetooth Specification Version 1.1.

[13] Bluetooth SIG. “Specification of the Bluetooth system-profiles,”, 2001. Bluetooth Specification Version 1.1.

[14] Scott A. Vanstone, “Elliptic curve cryptosystem- the answer to strong, fast public-key cryptography or securing constrained environments,” *Information Security Technical Report*, vol. 2, no. 2, pp. 78–87, 1997.

[15] Chou-Chen Yang, Ting-Yi Chang, and Min-Shiang Hwang, “A new anonymous conference key distribution system based on the elliptic curve discrete logarithm problem,” *Computer Standards & Interfaces*, vol. 25, no. 2, pp. 141–145, 2003.