Elliptic Curve Cryptography Based Wireless Authentication Protocol

Liu Yongliang[1], Wen Gao[1], Hongxun Yao[1], and Xinghua Yu[2]
(Corresponding author: Liu Yongliang)
School of Computer Science and Technology, Harbin Institute of Technology, Harbin[1]
8/F, Bld. A, Technology Fortune Center, No.8, XueQing Rd., HaiDian District, Beijing, 100085, P.R. China
(Email: liuyongliangs@hotmail.com)
School of Telecommunication Engineering, Beijing University of Posts and Telecommunications[2]

(Received Dec. 18, 2005; revised and accepted Feb. 16, 2006)

Abstract

Recently, Aydos et al. proposed an ECC-based wireless authentication protocol. Because their protocol is based on ECC, the protocol has significant advantage including lower computational burden, lower communication bandwidth and storage requirements. However, Mangipudi et al showed that the protocol is vulnerable to the man-in-the-middle attack from the attacker within the system and proposed a user authentication protocol to prevent the attack. This paper further shows that Aydos et al.'s protocol is vulnerable to man-in-the-middle attack from any attacker not restricted on the inside attacker. Then, a forging certificate attack on Mangipudi et al's protocol is presented. Next, the reasons that Aydos et al's protocol and Mangipudi et al's protocol suffer the attacks are analyzed. Finally, we propose a novel ECC-based wireless authentication protocol and analyze the security of our protocol.

Keywords: Elliptic curve cryptography, key exchange scheme, mutual authentication, security, wireless communication

1 Introduction

With rapid development of communication technology, wireless technology has been widely used in different areas. A problem coming with that is the security of wireless communication system which has drawn more and more attention. Compared with fixed network, wireless communication system is more vulnerable to eavesdropping, intercepting, and unauthorized access. In order to obtain reliable security for wireless communication system, user and system server need to authenticate mutually and establish the session key for subsequent communication.Since the limitation of user's equipment source, e.g., low power capability and small storage, it has caused more challenges to design an effective Mutual Authentication and Key Establishment Protocol (MAKEP) for wireless communication system. Traditional public key based MAKEP, including Station-to-Station protocol, have provided robust solution to security and authentication in general communication system. However, this kind of protocol has high computational complexity and it will take agency quite a little time to execute the protocol. Meanwhile, the agency's storage room must fulfill a high requirement due to the larger size of key and certificate. Thus the traditional public key based protocol is not suitable for wireless communication system. In private key based MAKEP, two communication parties need to share a long-term key or to have the trusted third party participate when establishing session key. In the fist case, to communicate with different parties, each party has to keep a series of different keys. In the latter one, the trusted third party must participate every time when the protocol is executed. Thus these protocols are also not suitable for wireless communication system.

Recently, Aydos et al. proposed an ECC-based wireless authentication protocol [2], and discussed the realization of it in [3]. This protocol has been paid quite a number of attention [4, 6, 7, 11, 12, 13, 14, 15]. This protocol uses elliptic curve digital signature algorithm and Diffie-Hellman key exchange scheme to provide mutual authentication and establish session key. The use of ECC results that this protocol has remarkable advantages to protocols based on RSA or digital signature algorithm (DSA) in terms of performance while providing the same secure level. These advantages include lower computational burden (meaning faster speed), lower bandwidth requirement and lower storage requirement. Unfortunately, there are some serious security deficiencies in this protocol. Sun et al. have proved that this protocol can't provide forward security and known-key security, as well as the mutual identity authentication [8]. Mangipudi et al. have proved

that this protocol is vulnerable to man-in-the-middle attack from attacker within system and proposed a user authentication protocol to prevent man-in-the-middle attack. This paper further shows that Aydos et al.'s protocol is vulnerable to man-in-the-middle attack from any attacker, not restricts to internal attackers. Then, we analyze the protocol proposed by Mangipudi et al and prove that the attacker can pass identity authentication by forging digital certificate. Next, the reasons why both protocols suffer from the attacks are analyzed. Finally, we propose a new ECC-based wireless authentication protocol and analyze the security of our protocol.

2 Theoretic Backgrounds

In this section, we briefly introduce the theoretic background of the protocol proposed by Aydos et al.: Elliptic Curve Diffie-Hellman Key Exchange Scheme (ECDH) and Elliptic Curve Digital Signature Algorithm (ECDSA).

2.1 ECDH

Let $P(x, y)$ be a point with order of n on elliptic curve E which is defined over finite field $GF(p)$, the ECDH can be described as follows:

1) S generates a random number $d_{\mathrm{S}} \in [2, n-1]$, calculates $D_{\mathrm{S}} = d_{\mathrm{S}}P$, and sends D_{S} to T.

2) T generates a random number $d_{\mathrm{T}} \in [2, n-1]$, calculates $D_{\mathrm{T}} = d_{\mathrm{T}}P$, and sends D_{T} to S.

3) S calculates key $sk_{\mathrm{S}} = d_{\mathrm{S}}D_{\mathrm{T}}$ and T calculates key $sk_{\mathrm{T}} = d_{\mathrm{T}}D_{\mathrm{S}}$.

Since $d_{\mathrm{S}}D_{\mathrm{T}} = d_{\mathrm{S}}d_{\mathrm{T}}P = d_{\mathrm{T}}d_{\mathrm{S}}P = d_{\mathrm{T}}D_{\mathrm{S}}$, thus $sk_S = sk_T$.

2.2 ECDSA

The ECDSA is composed of generate key pair, generate signature, and verify signature.

- Generate key pair:
  1) Generate a random number $d \in [2, n-1]$.
  2) Calculate $Q = dP$.
  3) Public key is Q, private key is d.
- Generate signature:
  In order to sign a message m, entity S operates as follows:
  1) Generate a random number $k \in [2, n-1]$.
  2) Calculate kP.
  3) Calculate $r = kP.x \bmod n$, where $kP.x$ denotes x coordinate of kP. If $r = 0$, return (1).
  4) Calculate $s = k^{-1}(h(m)+dr) \bmod n$, where $h(\cdot)$ is a secure hash function. If $s = 0$, return (1). The signature on message m is (r, s).
- Verify signature:
  1) Check $r \in [1, n-1]$, $s \in [1, n-1]$.
  2) Calculate $c = s^{-1} \bmod n$.
  3) Calculate $u_1 = h(m)c \bmod n$ and $u_2 = rc \bmod n$.
  4) Calculate $R = u_1P + u_2Q$.
  5) Calculate $v = R.x \bmod n$.
  6) Accept signature if and only if $r = v$.

3 Aydos et al.'s Protocol

Aydos et al. proposed ECC-based wireless authentication protocol in [2]. This protocol can be described as follows.

3.1 Terminal and Server Initialization

When terminal (T) wants to obtain a certificate, T generates a random number $d_{\mathrm{T}} \in [2, n-1]$, calculates $Q_{\mathrm{T}} = d_{\mathrm{T}}P$, then sends public key Q_{T} to Certificate Authority (CA) through a secure channel. After receiving the message, CA arranges a unique identity I_{T} and an expiration time t_{T}. Then CA generates a random number k_{T}, calculates $k_{\mathrm{T}}P$, $r_{\mathrm{T}} = k_{\mathrm{T}}P.x \bmod n$, $e_{\mathrm{T}} = h(Q_{\mathrm{T}}.x, I_{\mathrm{T}}, t_{\mathrm{T}})$ and $s_{\mathrm{T}} = k_{\mathrm{T}}^{-1}(e_{\mathrm{T}} + d_{\mathrm{CA}}r_{\mathrm{T}}) \bmod n$, where d_{CA} is the private key of CA, (r_{T}, s_{T}) is CA' signature on Q_{T}. CA sends I_{T}, t_{T}, e_{T}, (r_{T}, s_{T}), and its public key Q_{CA} to T through the secure channel. Repeat the same process, server(S) can obtain its certificate I_{S}, t_{S}, e_{S}, (r_{S}, s_{S}), and CA' public key Q_{CA}.

3.2 Mutual Authentication and Key Agreement

Mutual authentication and key agreement between S and T need to be completed in real time. This process is depicted in Figure 1.

In the protocol, T and S first exchange public key Q_{T} and Q_{S}, calculate $Q_{\mathrm{k}} = d_{\mathrm{T}}Q_{\mathrm{S}}$ and $Q_{\mathrm{k}} = d_{\mathrm{S}}Q_{\mathrm{T}}$ respectively. $Q_{\mathrm{k}}.x$ is used as the agreed key. Then, T and S encrypt their own certificate with the agreed key and exchange encrypted certificate respectively. After verifying each other's certificate is valid, T and S calculate session key $sk = Q_{\mathrm{k}}.x + g$.

Compared with RSA-based protocol or DSA-based key agreement protocol, Aydos et al's protocol has significant superiorities in terms of speed, storage requirement and bandwidth requirement. However, this protocol is vulnerable to man-in-the-middle attack. This will be described in the next section.

4 Man-in-the-middle Attacks

4.1 Attack from User within System

Mangipudi et al. proposed a man-in-the-middle attack aimed at Aydos et al.'s protocol, this attack requires that

T		S
	$\xleftarrow{Q_{\mathrm{S}}}$	
	$\xrightarrow{Q_{\mathrm{T}}}$	
Calculate Q_{k}		Calculate Q_{k}
		choose $g \in [2, n-1]$
		$m_{\mathrm{S}} = (e_{\mathrm{S}}, (r_{\mathrm{S}}, s_{\mathrm{S}}), t_{\mathrm{S}}, g)$
		$C_0 = \mathrm{E}(Q_{\mathrm{k}}.x, m_{\mathrm{S}})$
	$\xleftarrow{C_0}$	
Decrypt C_0		
$m_{\mathrm{T}} = (e_{\mathrm{T}}, (r_{\mathrm{T}}, s_{\mathrm{T}}), t_{\mathrm{T}}, g)$		
$C_1 = E(Q_{\mathrm{k}}.x, m_{\mathrm{T}})$		
	$\xrightarrow{C_1}$	
		Decrypt C_0
Check t_{S}		Check t_{T} and g
Calculate $c = s_{\mathrm{S}}^{-1}$		Calculate $c = s_{\mathrm{T}}^{-1}$
Calculate $u_1 = ce_{\mathrm{S}}$		Calculate $u_1 = ce_{\mathrm{T}}$
Calculate $u_2 = cr_{\mathrm{S}}$		Calculate $u_2 = cr_{\mathrm{T}}$
$R = u_1 P + u_2 Q_{\mathrm{CA}}$		$R = u_1 P + u_2 Q_{\mathrm{CA}}$
Verify $r_{\mathrm{S}} = R.x \bmod n$		Verify $r_{\mathrm{T}} = R.x \bmod n$
Calculate $s_{\mathrm{k}} = Q_{\mathrm{k}}.x + g$		Calculate $s_{\mathrm{k}} = Q_{\mathrm{k}}.x + g$

Figure 1: Aydos et al's protocol

attacker M must be a user within system, must have a valid certificate (e_{M}, (r_{M}, s_{M}), t_{M}). This attack is described as follows (see Figure 2).

1) S sends message Q_{S} to T.

2) M intercepts message Q_{S}, sends message Q_{M} to T impersonating S, and calculates $Q_{\mathrm{SM}} = d_{\mathrm{M}} Q_{\mathrm{S}} = d_{\mathrm{M}} d_{\mathrm{S}} P$.

3) T sends message Q_{T} to S, and calculates $Q_{\mathrm{TM}} = d_{\mathrm{T}} Q_{\mathrm{M}} = d_{\mathrm{T}} d_{\mathrm{M}} P$. $Q_{\mathrm{TM}} \cdot x$ is used as the agreed key.

4) M intercepts message Q_{T}, sends message Q_{M} to S impersonating T, and calculates $Q_{\mathrm{TM}} = d_{\mathrm{M}} Q_{\mathrm{T}} = d_{\mathrm{T}} d_{\mathrm{M}} P$.

5) S calculates $Q_{\mathrm{SM}} = d_{\mathrm{S}} Q_{\mathrm{M}} = d_{\mathrm{S}} d_{\mathrm{M}} P$. $Q_{\mathrm{SM}}.x$ is used as the agreed key.

6) S generates random number g, calculates $C_0 = E(Q_{\mathrm{SM}}.x, m_{\mathrm{S}})$, sends C_0 to T, where $m_{\mathrm{S}} = (e_{\mathrm{S}}, (r_{\mathrm{S}}, s_{\mathrm{S}}), t_{\mathrm{S}}, g)$.

7) M intercepts C_0, decrypts C_0 with $Q_{\mathrm{SM}}.x$ to obtain m_{S}. M checks the validity of t_{S}, calculates $c = s_{\mathrm{S}}^{-1}$, $u_1 = ce_{\mathrm{S}}$, $u_2 = cr_{\mathrm{S}}$, $R = u_1 P + u_2 Q_{\mathrm{CA}}$, checks whether $r_{\mathrm{S}} = R.x(\mathrm{mod} n)$ holds. Then, M calculates $C_0' = E(Q_{\mathrm{TM}}.x, m_{\mathrm{M}})$, sends C_0' to T, where $m_{\mathrm{M}} = (e_{\mathrm{M}}, (r_{\mathrm{M}}, s_{\mathrm{M}}), t_{\mathrm{M}}, g)$.

8) T decrypts C_0' with $Q_{\mathrm{TM}}.x$ to obtain m_{M}, checks the validity of t_{M}, calculates $c = s_{\mathrm{M}}^{-1}$, $u_1 = ce_{\mathrm{M}}$, $u_2 = cr_{\mathrm{M}}$, $R = u_1 P + u_2 Q_{\mathrm{CA}}$, checks whether $r_{\mathrm{M}} = R.x(\mathrm{mod} n)$ holds. Then, T calculates $C_1 = E(Q_{\mathrm{TM}}.x, m_{\mathrm{T}})$, sends C_1 to S, where $m_{\mathrm{T}} = (e_{\mathrm{T}}, (r_{\mathrm{T}}, s_{\mathrm{T}}), t_{\mathrm{T}}, g)$. Finally, T calculates $sk_{\mathrm{TM}} = Q_{\mathrm{TM}}.x + g$. sk_{TM} is used as the session key.

9) M intercepts C_1, decrypts C_1 with $Q_{\mathrm{TM}}.x$ to obtain m_{T}. M checks the validity of t_{T} and g, calculates $c = s_{\mathrm{T}} - 1$, $u_1 = ce_{\mathrm{T}}$, $u_2 = cr_{\mathrm{T}}$, $R = u_1 P + u_2 Q_{\mathrm{CA}}$, checks whether $r_{\mathrm{T}} = R.x(\mathrm{mod} n)$ holds. Then, M calculates $C_1' = E(Q_{\mathrm{SM}}.x, m_{\mathrm{M}})$, send $C_1'$to S, where $m_{\mathrm{M}} = (e_{\mathrm{M}}, (r_{\mathrm{M}}, s_{\mathrm{M}}), t_{\mathrm{M}}, g)$.

10) S decrypts $C_1'$with $Q_{\mathrm{SM}}.x$ to obtain m_{M}, checks the validity of t_{M} and g, calculates $c = s_{\mathrm{M}}^{-1}$, $u_1 = ce_{\mathrm{M}}$, $u_2 = cr_{\mathrm{M}}$, $R = u_1 P + u_2 Q_{\mathrm{CA}}$, checks whether $r_{\mathrm{M}} = R.x(\mathrm{mod} n)$ holds. Then, S calculates $sk_{\mathrm{SM}} = Q_{\mathrm{SM}}.x + g$. sk_{SM} is used as the session key.

The man-in-the-middle attack from user within system, which is described above, surely threatens protocol to a degree. In order to launch this attack, attacker has to send his own certificate to both communication parties. However, in many cases, malicious attacker doesn't want to leave such "evidence". In fact, the attacker needs not to send his own certificate to communication parties in order to launch the man-in-the-middle attack. We demonstrate this assertion by providing an attack example in next subsection.

4.2 Attack from any Attacker

Before launching attack, the attacker M needs to forge a public key "certificate": generates a random number d_{M}, calculates $Q_{\mathrm{M}} = d_{\mathrm{M}} P$. When T and S execute proto-

T		M		S
	$\xleftarrow{Q_{\mathrm{M}}}$		$\xleftarrow{Q_{\mathrm{S}}}$	
	$\xrightarrow{Q_{\mathrm{T}}}$		$\xrightarrow{Q_{\mathrm{M}}}$	
Calculate Q_{TM}		Calculate Q_{SM}		Calculate Q_{SM}
		Calculate Q_{TM}		Calculate C_0
			$\xleftarrow{C_0}$	
		Decrypt C_0		
		Calculate C_0'		
	$\xleftarrow{C_0'}$			
Decrypt C_0'				
Verify M's certificate				
Calculate C_1				
	$\xrightarrow{C_1}$			
		Decrypt C_1		
		Calculate C_1'		
			$\xrightarrow{C_1'}$	
				Decrypt C_1'
				Verify M's certificate
Calculate sk_{TM}		Calculate sk_{TM}; sk_{SM}		Calculate sk_{SM}

Figure 2: Man-in-the-middle attack from user within system

col, M can launch the man-in-the-middle attack which is described as follows and depicted in Figure 3.

1) S sends message Q_{S} to T.

2) M intercepts message Q_{S}, sends message Q_{M} to T impersonating S, and calculates $Q_{\mathrm{SM}} = d_{\mathrm{M}}Q_{\mathrm{S}} = d_{\mathrm{M}}d_{\mathrm{S}}P$.

3) T sends message Q_{T} to S, and calculates $Q_{\mathrm{TM}} = d_{\mathrm{T}}Q_{\mathrm{M}} = d_{\mathrm{T}}d_{\mathrm{M}}P$. $Q_{\mathrm{TM}} \cdot x$ is used as the agreed key.

4) M intercepts message Q_{T}, sends message Q_{T} to S impersonating T, and calculates $Q_{\mathrm{TM}} = d_{\mathrm{M}}Q_{\mathrm{T}} = d_{\mathrm{T}}d_{\mathrm{M}}P$.

5) S calculates $Q_{\mathrm{SM}} = d_{\mathrm{S}}Q_{\mathrm{M}} = d_{\mathrm{S}}d_{\mathrm{M}}P$. $Q_{\mathrm{SM}}.x$ is used as the agreed key.

6) S generates a random number g, calculates $C_0 = E(Q_{\mathrm{SM}}.x, m_{\mathrm{S}})$, sends C_0 to T, where $m_{\mathrm{S}} = (e_{\mathrm{S}}, (r_{\mathrm{S}}, s_{\mathrm{S}}), t_{\mathrm{S}}, g)$.

7) M intercepts C_0, decrypts C_0 with $Q_{\mathrm{SM}}.x$ to obtain m_{S}. M checks the validity of t_{S}, calculates $c = s_{\mathrm{S}}^{-1}$, $u_1 = ce_{\mathrm{S}}$, $u_2 = cr_{\mathrm{S}}$, $R = u_1P + u_2Q_{\mathrm{CA}}$, checks whether $r_{\mathrm{S}} = R.x(\mathrm{mod}n)$ holds. Then, M calculates $C_0' = E(Q_{\mathrm{TM}}.x, m_{\mathrm{S}})$, sends C_0' to T.

8) T decrypts C_0' with $Q_{\mathrm{TM}}.x$ to obtain m_{S}, checks the validity of t_{S}, calculates $c = s_{\mathrm{S}}^{-1}$, $u_1 = ce_{\mathrm{S}}$, $u_2 = cr_{\mathrm{S}}$, $R = u_1P + u_2Q_{\mathrm{CA}}$, checks whether $r_{\mathrm{S}} = R.x(\mathrm{mod}\, n)$ holds. Then, T calculates $C_1 = E(Q_{\mathrm{TM}}.x, m_{\mathrm{T}})$, sends C_1 to S, where $m_{\mathrm{T}} = (e_{\mathrm{T}}, (r_{\mathrm{T}}, s_{\mathrm{T}}), t_{\mathrm{T}}, g)$. Finally, T calculates $K_{\mathrm{TM}} = Q_{\mathrm{TM}}.x + g$. K_{TM} is as the session key.

9) M intercepts C_1, decrypts C_1 with $Q_{\mathrm{TM}}.x$ to obtain m_{T}. M checks the validity of t_{T} and g, calculates $c = s_{\mathrm{T}}^{-1}$, $u_1 = ce_{\mathrm{T}}$, $u_2 = cr_{\mathrm{T}}$, $R = u_1P + u_2Q_{\mathrm{CA}}$, checks whether $r_{\mathrm{T}} = R.x(\mathrm{mod}n)$ holds. Then, M calculates $C_1' = E(Q_{\mathrm{SM}}.x, m_{\mathrm{T}})$, and sends $C_1'$to S.

10) S decrypts C_1' with $Q_{\mathrm{SM}}.x$ to obtain m_{T}, checks the validity of t_{T} and g, calculates $c = s_{\mathrm{T}}^{-1}$, $u_1 = ce_{\mathrm{T}}$, $u_2 = cr_{\mathrm{T}}$, $R = u_1P + u_2Q_{\mathrm{CA}}$, checks whether $r_{\mathrm{T}} = R.x(\mathrm{mod}n)$ holds. Then, S calculates $sk_{\mathrm{SM}} = Q_{\mathrm{SM}}.x + g$. $sk_{\mathrm{SM}}$is as the session key.

The result of the attack described above is that $T(S)$ thinks that the received messages come from $S(T)$ and he shares the same session key K_{TM} (K_{SM}) with $S(T)$. But actually the session key K_{TM} is shared between T and M, and session key K_{SM} is shared between S and M. During the valid period of these two session keys, M can use these two keys to freely eavesdrop the communication between T and S, modify communication messages, or impersonate one party to the other.

5 Mangipudi et al.'s Protocol

Mangipudi et al. proposed a User Authentication Protocol (UAP), which is a variant of Aydos et al.'s protocol. In this protocol, user needs to authenticate himself to server. Similar to Aydos et al.'s protocol, Mangipudi et al.'s protocol is divided into two phases: initialization phase and user authentication phase.

T		M		S
	$\xleftarrow{Q_{\mathrm{M}}}$		$\xleftarrow{Q_{\mathrm{S}}}$	
	$\xrightarrow{Q_{\mathrm{T}}}$		$\xrightarrow{Q_{\mathrm{M}}}$	
Calculate Q_{TM}		Calculate Q_{SM}		Calculate Q_{SM}
		Calculate Q_{TM}		Calculate C_0
			$\xleftarrow{C_0}$	
		Decrypt C_0		
		Calculate C_0'		
	$\xleftarrow{C_0'}$			
Decrypt C_0'				
Verify S's certificate				
Calculate C_1				
	$\xrightarrow{C_1}$			
		Decrypt C_1		
		Calculate C_1'		
			$\xrightarrow{C_1'}$	
				Decrypt C_1'
				Verify T's certificate
Calculate sk_{TM}		Calculate sk_{TM}; sk_{SM}		Calculate sk_{SM}

Figure 3: Man-in-the-middle attack from any attacker

5.1 Initialization Phase

Server Initialization Phase: The differences in server initialization phase between Mangipudi et al.' protocol and Aydos et al.'s protocol are: (1) Server needs to calculate the private key d_S's inversion d_S^{-1}. (2) CA only arranges an identity I_S and an expiration time t_S (not certificate) for server, sends $Q_{C,A}$, I_S , and t_S to server, and stores server's public key Q_S and expiration time t_S.

User Initialization Phase: The differences in user initialization phase between Mangipudi et al.' protocol and Aydos et al.'s protocol is: CA sends server's public key and expiration time, not its own public key, to user.

5.2 User Authentication Phase

The process is as follows and is depicted in Figure 4. User generates a random number $g_{\mathrm{T}} \in [2, n-1]$, calculates $Q_{\mathrm{R}} = g_{\mathrm{T}}Q_{\mathrm{S}} = g_{\mathrm{T}}d_{\mathrm{S}}P$, sends Q_{R} to server, and calculates $Q_{\mathrm{TS}} = g_{\mathrm{T}}P$. $g_{\mathrm{TS}} = Q_{\mathrm{TS}}.x$ is used as the agreed key. After receiving message, server calculates $d_{\mathrm{S}}^{-1}Q_{\mathrm{R}}$ to obtain the agreed key g_{ST}. Then, server encrypts the message $m_{\mathrm{S}}$with agreed key to obtain $C_0 = E(g_{\mathrm{ST}}; m_{\mathrm{S}})$, sends C_0 to user, where $m_{\mathrm{S}} = (g_{\mathrm{TS}}, g_{\mathrm{S}}, t_{\mathrm{S}})$, g_{S} is a random number generated by server, $gS \in [2, n-1]$. User decrypts C_0 with agreed key g_{TS} to obtain m_{S}, checks the validity of g_{TS} and t_{S}. Then, user encrypts message m_{T} with g_{TS} to obtain $C_1 = E(g_{\mathrm{TS}}; m_{\mathrm{T}})$, where $m_{\mathrm{T}} = (e_{\mathrm{T}}, (r_{\mathrm{T}}, s_{\mathrm{T}})$, t_{T}, $g_{\mathrm{S}})$, sends C_1 to server, and calculates session key $sk_{\mathrm{TS}} = h(g_{\mathrm{TS}}, g_{\mathrm{S}})$. Server decrypts C_1, checks the validity of g_{S} and t_{T}, verifies user's certificate. Then, server calculates session key $sk_{\mathrm{ST}} = h(g_{\mathrm{ST}}, g_{\mathrm{S}})$.

6 Attack to Mangipudi et al.'s Protocol

Mangipudi et al.'s protocol is robust to man-in-the-middle attack. However, this protocol is vulnerable to the forging certificate attack launched by attacker after he has forged the certificate. To make it easy to be understood, we present a simple way of forging certificate here. Let $e_{\mathrm{M}} = 0$, $r_{\mathrm{M}} = Q_{\mathrm{CA}}.x$, $s_{\mathrm{M}} = r_{\mathrm{M}}$, M uses $(e_{\mathrm{M}}, (r_{\mathrm{M}}, s_{\mathrm{M}}))$ as the forging certificate, and forges an expiration time t_{M} for this certificate. Then, M can use forging certificate to impersonate a legal user to server. This attack is described as follows and is depicted in Figure 5.

1) M generates a random number $g_{\mathrm{M}} \in [2, n-1]$, calculates $Q_{\mathrm{R}} = g_{\mathrm{M}}Q_{\mathrm{S}}$, sends Q_{R} to server, and calculates $Q_{\mathrm{MS}} = g_{\mathrm{M}}P$. $g_{\mathrm{MS}} = Q_{\mathrm{MS}}.x$ is as the agreed key.

2) Server calculates $Q_{\mathrm{SM}} = d_{\mathrm{S}}^{-1}Q_{\mathrm{R}} = g_{\mathrm{M}}P$. $Q_{\mathrm{SM}}.x$ is as the agreed key. Then, server calculates $C_0 = E(g_{\mathrm{SM}}, m_{\mathrm{S}})$, sends C_0 to M, where $m_{\mathrm{S}} = (g_{\mathrm{SM}}, g_{\mathrm{S}}, t_{\mathrm{S}})$, g_{S} is random number generated by server.

3) M decrypts C_0 with the agreed key Q_{SM} to obtain m_{S}, checks the validity of g_{SM} and t_{S}. Then, M encrypts message $m_{\mathrm{M}} = (e_{\mathrm{M}}, (r_{\mathrm{M}}, s_{\mathrm{M}}), t_{\mathrm{M}}, g_{\mathrm{S}})$ with g_{SM} to obtain $C_1 = E(g_{\mathrm{MS}}; m_{\mathrm{M}})$, sends C_1 to server, and calculates session key $sk_{\mathrm{MS}} = h(g_{\mathrm{MS}}, g_{\mathrm{S}})$.

4) Sever decrypts C_1, check the validity of g_{S} and t_{M}, calculates $c = s_{\mathrm{M}}^{-1}$, $u_1 = ce_{\mathrm{M}} = 0$, $u_2 = cr_{\mathrm{M}} = 1$,$R = u_1P + u_2Q_{\mathrm{CA}} = Q_{\mathrm{CA}}$, $v = R.x = Q_{\mathrm{CA}}.x$, verify

T		S
Choose $g_{\mathrm{T}} \in [2, n-1]$		Choose $g_{\mathrm{S}} \in [2, n-1]$
Calculate $Q_{\mathrm{R}} = g_{\mathrm{T}} Q_{\mathrm{S}} = g_{\mathrm{T}} d_{\mathrm{S}} P$		
Calculate $Q_{\mathrm{TS}} = g_{\mathrm{T}} P$		
$g_{\mathrm{TS}} = Q_{\mathrm{TS}}.x$	$\xrightarrow{Q_{\mathrm{R}}}$	
		Calculate $Q_{\mathrm{ST}} = d_{\mathrm{S}}^{-1} Q_{\mathrm{R}}$
		$g_{\mathrm{ST}} = Q_{\mathrm{ST}}.x$
		$C_0 = E(g_{\mathrm{ST}}; g_{\mathrm{ST}}, g_{\mathrm{S}}, t_{\mathrm{S}})$
	$\xleftarrow{C_0}$	
Decrypt C_0		
$C_1 = E(g_{\mathrm{TS}}, (e_{\mathrm{T}}, (r_{\mathrm{T}}, s_{\mathrm{T}}), t_{\mathrm{T}}, g_{\mathrm{S}}))$		
	$\xrightarrow{C_1}$	
		Decrypt C_1
		Check the validity of $g_{\mathrm{S}}$and t_{T}
		Calculate $c = s_{\mathrm{T}}^{-1}$
		Calculate $u_1 = ce_{\mathrm{T}}$
		Calculate $u_2 = cr_{\mathrm{T}}$
		Calculate $R = u_1 P + u_2 Q_{\mathrm{CA}}$
		$v = R.x (\mathrm{mod} n)$
		Verify $r_{\mathrm{T}} = v$
Calculate $sk_{\mathrm{TS}} = h(g_{\mathrm{TS}}, g_{\mathrm{S}})$		Calculate $sk_{\mathrm{ST}} = h(g_{\mathrm{ST}}, g_{\mathrm{S}})$

Figure 4: Mangipudi et al.'s user authentication protocol

M		S
Calculate $Q_{\mathrm{MS}} = g_{\mathrm{M}} P$		
	$\xrightarrow{Q_{\mathrm{R}}}$	
		Calculate $Q_{\mathrm{SM}} = d_{\mathrm{S}}^{-1} Q_{\mathrm{R}}$
$g_{\mathrm{MS}} = Q_{\mathrm{MS}}.x$		$g_{\mathrm{SM}} = Q_{\mathrm{SM}}.x$
		$C_0 = E(g_{\mathrm{SM}}; m_{\mathrm{S}})$
	$\xleftarrow{C_0}$	
Decrypt C_0		$C_0 = E(g_{\mathrm{ST}}; g_{\mathrm{ST}}, g_{\mathrm{S}}, t_{\mathrm{S}})$
$C_1 = E(g_{\mathrm{MS}}, m_{\mathrm{M}})$		
	$\xrightarrow{C_1}$	
		Decrypt C_1
		$c = s_{\mathrm{M}}^{-1}$
		$u_1 = ce_{\mathrm{M}}$
		$u_2 = cr_{\mathrm{M}}$
		$R = u_1 P + u_2 Q_{\mathrm{CA}}$
		Verify $r_{\mathrm{M}} = R.x$ holds
sk_{MS}		sk_{SM}

Figure 5: Mutual authentications and key agreement

whether the equation $r_{\mathrm{M}} = v$ holds. According to the manner in which M forges certificate, $r_{\mathrm{M}} = v$ holds. Thus, M has passed the identity authentication to server. Finally, server calculates session key $sk_{\mathrm{SM}} = h(g_{\mathrm{SM}}, g_{\mathrm{S}})$.

Through the way of forging certificate described above, M can successfully pass server's authentication. This is surely a very serious threat to Mangipudi's protocol. What worth being paid attention is that only checking whether $e = 0$ is not sufficient in order to prevent forging certificate attack. This is because the manner of forging certificate described above is just one of the ways to forge certificate, some other way to forge certificate can be devised in which the value of e can be not equal to zero.

7 Analyzes

In this section, some security deficiencies of Aydos et al.'s protocol and Mangipudi et al.'s protocol, as well as the reason why the two protocols suffer from attacks, will be analyzed.

7.1 Security Deficiencies of Aydos et al.'s Protocol

The security deficiencies of Aydos et al.'s protocol can be explained from two aspects: design principle and design details of the protocol.

- Deficiency of design principle:
  The design principle of Aydos et al.'s protocol is that both communication parties exchange public key to establish the agreed key, then exchange certificate to verify each one's identity, and finally calculate session key using both the agreed key and random number. In which, certificates are exchanged in their encrypted form. This not only provides the secrecy of the certificate, but also provides (agreed) key confirmation to both parties. With the assumption that the agreed key won't be disclosed, this de-

sign principle can achieve entity authentication [8]. This is because: (1) Certificate assures authenticity and coupling of entity identity and its public key. (2) Key confirmation makes both two parties believe that each other has the private key corresponding to public key included in certificate. (3) Using nonce random number ensures the freshness of message. Thus, with the assumption described above, this design principle can achieve entity authentication. However, this assumption isn't tenable. Since public key information is changeless, so the agreed key computed from public keys of two parties is constant. Therefore the agreed key may be disclosed (Taking the principle of being prudent into consideration, many protocol designers assume that long term public key (Q_{S} and Q_{T}) is much possible to be disclosed). In the case of that agreed key is possible to be disclosed, the protocol can't achieve entity authentication. For example, the attacker can use Alice's certificate eavesdropped to impersonate Alice to Bob under the assumption that the agreed key between Alice and Bob was disclosed.

- Deficiencies in design details:
  Even under the assumption that agreed key won't be disclosed, some deficiencies in design details of this protocol can still induce the protocol to be unable to achieve entity authentication.

  1) This protocol can not provide entity authentication of server to terminal. The nonce random number generated by server is included in message C_1 which is returned to server by terminal, thus server knows the current message is fresh, this makes it possible for protocol to provide entity authentication of terminal to server. However, the same mechanism is not introduced into the authentication of server to terminal. Thus, the terminal can only obtain the message origination authentication at most, and it can't obtain the entity authentication of server. Since terminal can't make sure whether the message is fresh, the attacker may send old message again to deceive.
  2) There are some problems in the entity authentication because it is impossible for verifier to obtain any identity information of the other party. In protocol, to satisfy the anonymity requirement, both parties only exchange public key and certificate, but don't exchange identity information (They can't learn identity information from certificate). In this case, both communication parties can't know any identity information of the other party. This induces that the protocol is vulnerable to man-in-the-middle attack from user within system, which is proposed by Mangipudi et al.
  3) The verification for public key certificate is unreasonable, and there is no way to verify the association between public key certificate and public key. Usually, a public certificate contains messages such as entity identity, public key, and expiration time, and so on. It also contains CA's signature on these messages. Entity identity and its public key are associated closely by signatures. And the signature illuminates that the public key in certificate is issued to the entity whose identity is included in certificate. In the protocol, CA issues certificate $(e, (r, s), t)$ to entity, where $e = h(Q, I, t)$, (r, s) is CA's signature on e. When verify certificate, since both communication parties haven't exchanged identity information, both of them can't verify the association between e and Q (can't verify whether e is surely obtained by calculating the hash value of the other party's identity, public key and expiration time), and can't make sure whether the public key certificate is issued for the other party's either. This induces that this protocol is vulnerable to the man-in-the-middle attack presented in this paper.

Due to the deficiencies in design principle and design details, besides can't provide entity authentication, Aydos et al.'s protocol can't satisfy some basic security aims either, including explicit key authentication, forward security, known-key security, key control, and so on.

7.2 Security Deficiencies of Mangipudi et al.'s Protocol

Mangipudi et al.'s protocol is the variant of Aydos et al.'s protocol, and it can resist man-in-the-middle attack. However, there are still some vital security deficiencies in this protocol, including deficiencies inherited from Aydos et al.'s protocol and deficiencies in the protocol itself.

Firstly, this protocol doesn't provide user's entity authentication to server. The reasons are similar to (2) and (3) in Section 7.1.2. This allows attacker to forge a certificate and use it to pass server's identity authentication.

Secondly, this protocol doesn't provide forward security because disclosing server's secret key d_{S} or d_{S}^{-1} will lead to all previous secret information being disclosed.

Thirdly, this protocol doesn't provide explicit (session) key authentication. This makes both communication parties can't know whether the other party exactly calculates the session key.

Fourthly, it is difficult to implement the renewal for security of wireless communication system based on this protocol. Once server's secret key is disclosed, all previous communication messages between users and server, including the user's certificate, will be disclosed. In this case, just updating server's secret key isn't sufficient because the attacker can use the disclosed certificate to impersonate authorized user to server. Thus, the system has to be updated entirely. Compared with this protocol,

many protocols such as Station-to-Station protocol only need to update the disclosed long-term secret key. At this point, there is quite a deficiency in this protocol.

8 Our Protocol

8.1 The Description of Our Protocol

In this section, we will propose a novel ECC-based key establishment protocol. Our protocol takes station-to-station protocol as the basic framework. In order to prevent Lowe attack [9], the communication parties' identity information is added into the signature message. Meanwhile, to make it more efficient to execute, this protocol uses one of the most efficient signature schemes, Schnorr signature scheme [10], as digital signature algorithm

Assume that terminal T and server S respectively get certificate C(T) and C(S) from certificate authority. In our protocol, terminal and server execute steps as follows, and it is depicted in Figure 6.

1) T executes precomputation: generates random number $y \in [1, n-1]$, calculates $Y = yP$; generates random number b, calculates $(b_1, b_2) = bPandb' = b_1(\mathrm{mod}n)$.

2) T sends Y to S.

3) S generates a random number $x \in [1, n-1]$, calculates $X = xP$.

4) S calculates $K = yX = yxP$, K is as session key shared by T and S (In practice, session key is generated by a key derivation function). S calculates the signature on (X,Y, I_S): (5) and (6).

5) S generates random number a, calculates $(a_1, a_2) = aP$ and $a' = a_1(\mathrm{mod}n)$. Then S calculates $g = h(a', X, Y, I_S)$. If $g = 0$, executes this step again, otherwise, goes on executing the following protocol.

6) S calculates $v = (a - d_S g)(\mathrm{mod}n)$. (a', v) is as S's signature on message (X, Y, I_S).

7) S uses symmetric encryption algorithm, such as AES, to encrypt C(S) and (a', v) with K. The encryption result is $E_K(C(S), (a', v))$.

8) S sends message X and $E_K(C(S), (a', v))$ to T.

9) T calculates session key $K = yX = xyP$, decrypts $EK(C(S), (a', v))$ with K to get C(S) and (a', v). T verifies certificate C(S). If the verification result is wrong, the protocol will be terminated. Otherwise, the protocol will be continued. T verifies S's signature: (10)-(12).

10) T checks a', $v \in [1, n-1]$.

11) T calculates $g = h(a', X, Y, I_S)$, $t_1 = v(\mathrm{mod}n)$, $t_2 = g(\mathrm{mod}n)$ and $(x_1, x_2) = t_1 P + t_2 Q_S$.

12) T checks whether $a' = x1(\mathrm{mod}\ n)$ holds. If it doesn't hold, the protocol will be terminated. Otherwise, the protocol will be continued. T calculates signature on (Y, X, IT, IS): (13)-(14).

13) T calculates $g = h(b', Y, X, I_T, I_S)$.

14) T calculates $u = (b - d_T g) \bmod n$. (b', u) is used as T's signature on (Y, X, I_T, I_S).

15) T uses symmetry encryption algorithm to encrypt C(T) and (b', u) with K, sends $E_K(C(T), (b', u))$ to S.

16) S decrypts $E_K(C(T), (b', u))$ with K and gets C(T) and (b', u).

17) S checks the validity of C(T). If C(T) is a valid certificate, the protocol will be continued. Otherwise, the protocol will be terminated. S verifies T's signature: (18)-(20).

18) S checks b', $u \in [1, n-1]$.

19) S calculates $f = h(b', Y, X, I_T, I_S)$, $t_1 = u(\mathrm{mod}n)$, $t_2 = f(\mathrm{mod}n)$, and $(x_1, x_2) = t_1 P + t_2 Q_S$.

20) S checks whether $b' = x_1(\mathrm{mod}n)$ holds.

8.2 Security Analysis of our Protocol

- Entity authentication and explicit key authentication:
  In our protocol, T first sends message Y to S. Once receiving the message, S calculates session key K. Under the assumption that it is difficult to calculate elliptic curve discrete logarithm problem, S believes that only the party, who created the message, can calculate the shared session key exactly. After that, S calculates the signature (a',v) on message (X, Y, I_S), encrypts certificate C(S) and signature (a',v) with session key K, and sends X and $E_K(C(S), (a', v))$ to T. T calculates session key K, and decrypts the received message with K. After verifying certificate and signature, T makes sure that S is legal intended server, no other than S can calculate the session key shared with it, and S has calculated session key exactly. Therefore, our protocol has provided S's entity authentication and explicit key authentication to T. Then, T calculates the signature (b', u) on (Y,X, I_T, I_S), encrypts certificate C(T) and signature (b', u) with session key K, and sends $E_K(C(T), (b', u))$ to S. S decrypts the received message with session key K, and gets C(T) and (b', u). After verifying certificate and signature, S is sure that T is a legal terminal, only T can calculate the session key shared with it, and T has calculated session key exactly. Therefore, our protocol has provided T's entity authentication and explicit key authentication to S. In a word, our protocol has provided entity authentication and definite key authentication to both of communication parties.

S		T
		Precomputation
		(1)' Choose random number y, $b \in [1, n-1]$
		(2)' Calculate $Y = yP$, $(b_1, b_2) = bP$, $b' = b_1 (\mathrm{mod} n)$
	$\underleftarrow{Y}$	
(1)' Choose random number $x \in [1, n-1]$		
(2)' Calculate $X = xP$		
(3)' Calculate $K = yX = xyP$ Calculate signature		
(4)' Choose random number a		
(5)' Calculate $(a_1, a_2) = aP$, $a = a_1 (\mathrm{mod} n)$		
(6)' Calculate $g = h(a, X, Y, I_S)$, if $g = 0$, go to (4)'.		
(7)' Calculate $v = (a - dSg) (\mathrm{mod} n)$, if $v = 0$, go to (4)'.		
(8)' The signature on (X, Y, I_S) is (a, v)		
(9)' Encrypt certificate C(S) and (a, v) with K		
Get $E_K(C(S), (a, v))$.		
	$\underrightarrow{X, E_K(C(S), (a, v))}$	
		(3)' Calculate $K = xY = xyP$
		(4)' Decrypt $E_K(C(S), (a, v))$ to get C(S) and (a, v)
		(5)' Validate certificate C(S) Validate signature
		(6)' Check a, $v[1, n-1]$
		(7)' Calculate $g = h(a, X, Y, I_S)$
		(8)' Calculate$t_1 = v \bmod n$, $t_2 = g \bmod n$ and $(x_1, x_2) = t_1 P + t_2 Q_S$
		(9)' Check whether $a = x_1 \bmod n$ holds
		(10)' Calculate the signature (b, u) on (Y, X, I_T, I_S)
		(11)' Encrypt certificate C(T) and (b, u) with K
	$\underleftarrow{E_K(C(T), (b, u))}$	
(10)' Decrypt $E_K(C(T), (b, u))$		
(11)' Verify both certificate and signature		

Figure 6: Our protocol

- Known key security:
  We show that our protocol possesses the known key security property from two aspects.

  1) In our protocol, session key is calculated from the random numbers generated by the two communication parties every time. If random number generator or algorithm is good enough, then the probability that the same session keys are generated in two runs of our protocol is neglectable. Therefore, from this point, it is impossible to learn other keys when knowing several session keys.
  2) Our protocol's security depends on the difficulty of calculating elliptic curve discrete logarithm problem. Knowing several session keys is not helpful to decrease the difficulty of calculating elliptic curve discrete logarithm problem. Therefore, even though attacker has known several session keys, he can't know other session keys.

- Forward secret:
  In our protocol, session key is calculated from the random numbers generated by the two communication parties every time. Therefore, the disclosure of secret information won't lead to the disclosure of previous session keys.

- Key compromise impersonation:
  In our protocol, the secret information of both communication parties is bounded with certificate. Therefore, using its secret information or other secret information it gets through certain manner, the attacker can't impersonate other party besides the owners of secret information. That is to say, our protocol can prevent key compromise impersonation.

- Unknown key share:
  In our protocol, both parties' public key and secret key are unique. What's more, messages in this protocol contain the identities of both communication parties, and the protocol has provided explicit key authentication. Therefore, our protocol is robust to the unknown key share attack based on public key substitution and based on duplicate-signature key selection [17].

- Key control:
  In our protocol, session key is calculated from the random numbers generated by both parties, and both random numbers have coequal effect on generation of session key. Therefore, any party can't force the session key to be a value chosen beforehand.

- Terminal's anonymity:
  In our protocol, terminal T's public key certificate is sent to server S in the encryption form. This has prevented attacker from eavesdropping terminal identity information. Therefore, our protocol has provided terminal's anonymity.

All the characters described above have ensured that our protocol satisfies the basic security requirement and is robust to the existing attacks.

9 Conclusions

In this paper, we have proved Aydos et al.'s protocol is vulnerable to man-in-the-middle attacks from any attacker not restricted on attacker within system. Next, we proposed a forging certificate attack to Mangipudi et al.'s protocol. Then, we analyzed the reasons why Aydos et al.'s protocol and Mangipudi et al.'s protocol suffer from attacks and pointed out some other deficiencies in these two protocols. Finally, we propose a new ECC-based wireless authentication protocol and analyze the security of our protocol. The analysis result shows that our proposal satisfies the basic security requirement and is robust to the existing attacks.

Acknowledgments

This work was partially supported by National Natural Science Foundation of China under grant No. 60472043and Microsoft Research Asia (TWC-2006-Project-5). The authors are grateful to the anonymous reviewers for valuable comments. While the authors are grateful to Dr. Shaohui Liu and Dr. Bo Wu for helpful discussions.

References

[1] M. Abadi and R. Needham, "Prudent engineering practice for cryptographic protocols," *IEEE Transactions on Software Engineering*, vol. 22, no. 1, pp. 6-15, 1996.

[2] M. Aydos, B. Sunar, and C. K. Koc, "An elliptic curve cryptography based authentication and key agreement protocol for wireless communication," in *Proceedings of the 2nd International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, pp. 1-12, Dallas, 1998.

[3] M. Aydos, T. Yanik, and C. K. Koc, "High-speed implementation of an ecc-based wireless authentication protocol on an arm microprocessor," *IEE Proceedings - Communication*, vol. 148, no. 5, pp. 273-279, 2001.

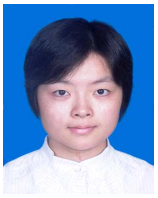
[4] T. Coffey, R. Dojen, and T. Flanagan, "On the automated implementation of modal logics used to verify security protocols," in *Proceedings of the 2003 International Symposium on Information and Communication Technologies*, pp. 321-334, New York, ACM Press, 2003.

[5] W. Diffie, P. Oorschot, and M. Wiener, "Authentication and authenticated key exchanges," *Designs, Codes, and Cryptography*, vol. 2, no. 2, pp. 107-125, 1992.
[6] J. Jiang and C. He, "Novel mutual authentication and key agreement protocol based on NTRU cryptography for wireless communications," *Journal of Zhejiang University (Chinese)*, vol. 6, no. 5, pp. 399-404, 2005.
[7] K. Mangipudi, N. Malneedi, and R. Katti, "Attacks and solutions on Aydos-Savas-Koc's wireless authentication protocol," in *Proceedings of 2004 IEEE Global Telecommunications Conference*, pp. 2229-2234, 2004.
[8] A. Menezes, P. Oorschot, and S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1996.
[9] G. Lowe, "Some new attacks upon security protocols," in *Proceedings of 9th IEEE Computer Security Foundations Workshop*, IEEE Press, pp. 162-169, 1996.
[10] C. P. Schnorr, "Efficient identification and signatures for smart cards," in *Proceedings on Advances in Cryptology (Crypto'89)*, LNCS 1989, pp. 239-252, Springer Verlag, 1989.
[11] R. Schroeppel, C. Beaver, and R. Gonzales, "A low-power design for an elliptic curve digital signature chip," in *Proceedings of the 2002 Workshop on Cryptographic Hardware and Embedded Systems*, pp. 366-380, Berlin, Springer Press, 2003.
[12] A. Sui, Y. Yang, and X. Niu, "Research on the authenticated key agreement protocol based on elliptic curve cryptography," *Journal of Beijing University of Posts and Telecommunications (in Chinese)*, vol. 27, no. 3, pp. 28-32, 2004.
[13] A. F. Sui, L. C. K. Hui, and Y. X. Yang, "Elliptic curve cryptography based Authentication key agreement with pre-shared password," *Journal of Electronics (China)*, vol. 22, no. 3, pp. 268-272, 2005.
[14] H. M. Sun, B. T. Hsieh, and S. M. Tseng, "Cryptanalysis of Aydos et al.'s ECC-based wireless authentication protocol,"in *Proceedings of 2004 IEEE International Conference on E-Technology, E-Commere, and E-Service*, pp. 563-566, 2004.
[15] H. Qiun, Y. Yangn, and Z. Hu, "An improvement of Aydos et al's wireless authentication protocol," *Computer Engineering and Applications (in Chinese)*, vol. 40, no. 31, pp. 3-5, 2004.
[16] S. B. Wilson, and A. Menezes, "Authenticated Diffie-Hellman key agreement protocols," in *Proceedings of the 1998 Selected Areas in Cryptography*, LNCS 1556, pp. 339-361, Springer-Verlag, 1999.
[17] S. B. Wilson and A. Menezes, "Unknown key share attacks on the station-to-station protocol," in *Proceedings of 1999 International Workshop on Practice and Theory in Public Key Cryptography*, LNCS 1560, pp. 154-170, Springer-Verlag, 1999.

Liu Yongliang is a Ph.D. candidate at Harbin Institute of Technology. He received his MS degree in Mathematics from Harbin Institute of Technology, China, 2000. His research interests include: digital watermark, cryptographic protocol, and digital right management. He has published 30 scientific papers.

WEN GAO received his Ph.D. degree in Computer Science, Harbin Institute of Technology, China, 1988 and Ph.D. in Electronics Engineering, University of Tokyo, Japan, 1991. He was a Research Fellow at Institute of Medical Electronics Engineering, the University of Tokyo, in 1992; a Visiting Professor at Robotics Institute, Carnegia Mellon University, in 1993; a Visiting Professor at MIT AI Lab, from May 1994 to December 1995. Now he is a professor of Institute of Computing Technology, Chinese Academy of Sciences. His researchinterests include pattern recognition and arti.cial intelligence, image understanding, data compression, hand gesture recognition, multimodal interface, and computer vision. He has published 7 books and over 260 scientific papers.

Hongxun Yao received her B.S. and M.S. degrees in computer science from Harbin Shipbuilding Engineering Institute, Harbin, China, in 1987 and 1990 respectively, and the Ph.D degree from Harbin Institute of Technology, Harbin, China, in 2003. Her research interests lie in image processing, pattern recognition, multimedia technology and natural human-computer interface and information hiding technology. She has published 4 books and over 60 scientific papers.



Xinghua Yu is an undergraduate in School of Telecommunication Engineering, Beijing University of Posts and Telecommunication. She will receive her B.S. degree in June, 2006, and continue her graduate study in Institute of Computer Technology, Chinese Academy of Sciences. Her research interests lie in telecommunication and digital rights management.