New Key Recovery in WAKE Protocol

Chong Hee Kim and Pil Joong Lee

Pohang University of Science & Technology (POSTECH), Korea
chhkim@oberon.postech.ac.kr, pj1@postech.ac.kr
<http://ist.postech.ac.kr/>

Abstract. Wireless authentication and key establishment (WAKE) protocols are essential in mobile communications. The key recovery mechanism may also be required in mobile communication systems for the investigation of serious crimes and for national security. Recently some protocols that give key recovery in the WAKE protocol have been published. In this paper we propose new key recovery protocols that improve previous protocols in security and efficiency.

1 Introduction

Mobile communications comprises one of the fastest growing sectors of the IT industry. Current second generation services such as cellular and PCS are world-wide used. The next generation of mobile communication systems, such as UMTS (Universal Mobile Telecommunications System) [1] and IMT-2000 (International Mobile Telecommunications-2000) [2,3] will provide a wider spectrum of services, including high quality multimedia and Internet services.

The security features for mobile communication systems include: confidentiality on the air interface (which is much more vulnerable to eavesdropping than a wired interface), anonymity of the user and authentication of the user to network administration in order to prevent fraudulent use of the system [11]. As the number of network operators (NOs) or value-added service providers (VASPs, e.g. a bank in electronic commerce) grows, a user may want to ensure that he is connected to the NO/VASP that he trusts. Further, users and NOs/VASPs may want to establish a shared session key to encrypt the subsequent message exchange. This is achieved through the successful execution of wireless authentication and key establishment (WAKE) protocol between the user and the NOs/VASPs.

As mobile communications increase, the key recovery mechanism may be required in mobile communication system for the investigation of serious crime and national security. Key recovery, although still a matter of debate, is expected to be deployed in the UMTS security architecture to satisfy government requirements concerning lawful interception [4].

In this paper, we briefly review the background on key recovery in Section 2. This is followed by previous key recovery systems in WAKE protocol of ASPeCT project [11] in Section 3. New key recovery systems will be described in Section 4. Finally Section 5 contains concluding remarks.

2 Background on Key Recovery

The history of key recovery started in April 1993, with the proposal by the U.S. government of the Escrow Encryption Standard [5], EES, also known as the CLIPPER project. Afterwards, many key recovery schemes have been proposed.

To protect user privacy, the confidentiality of data is needed. For this, key recovery (KR) seems useless, but there are some scenarios where key recovery may be needed [6,7]:

- when the decryption key has been lost or the user is not present to provide the key
- where commercial organizations want to monitor their encrypted traffic without alerting the communicating parties; to check that employees are not violating an organization's policies, for example
- when a national government wants to decrypt intercepted data for the investigation of serious crimes or for national security reasons.

In this paper we use following terminology, as in [6]:

Key Recovery Agent (KRA). A trusted third party that performs KR in response to an authorized request.

Key Recovery Information (KRI). An aggregate of data that is needed by the KRA in order to complete a KR request, e.g. a session key encrypted under the KRA's public key.

Key Recovery Requester (KRR). An authorized entity that requests KR from the KRA. The KRR would usually be a LEA (Law Enforcement Agency) in possession of a valid warrant.

Interception Agent. An entity that acts in response to an authorized request for interception of a target identity by filtering out the communications traffic corresponding to the target identity. This function would usually be performed by NOs [8,9]

In [10], two different enforceability levels are defined in key recovery:

Level 1. At this level, the enforceability mechanism ensures that no user of the system (including VASP) can succeed in circumventing the KR mechanism unilaterally, i.e. without the cooperation of the other communication party.

Level 2. At this level, enforceability mechanisms ensure that no user of the system (including VASP) can succeed in circumventing the KR mechanism, even with the cooperation of the other communicating party.

If a user in the KR system sends a bogus value instead of KRI, the KR cannot succeed and the session key will be unrecoverable. Here, the level 1 condition is not satisfied. In order to satisfy the level 1 condition, the KR system needs additional verifiable information to prove validity of KRI before KR occurs. However, KR system cannot satisfy the level 2 condition because of super-encryption [10]. Therefore, we do not consider level 2 condition here.

3 Previous Protocols

In this section we review the ASPeCT WAKE protocol [11,12], the KR enhanced ASPeCT protocol [7], and the modified KR enhanced ASPeCT protocol [6]. We mainly use notations and definition in [6] to describe these protocols.

3.1 ASPeCT WAKE Protocol

The most well-known public key based WAKE protocol for UMTS is proposed by the ASPeCT project, which is responsible for the research and development of security technologies to be used in the UMTS system. The ASPeCT protocol [11] is proposed for user-to-NO/VASP interfaces to achieve the following goals:

1. mutual explicit authentication of A and B;
2. agreement between A and B on a secret session key K_{AB} with mutual implicit key authentication;
3. mutual key confirmation between A and B;
4. mutual assurance of key freshness (mutual key control);
5. non-repudiation of origin by A for relevant data sent from A to B;
6. confidentiality of relevant data sent by A to B.

Two protocols have been designed for these purposes (B- and C- variant) [11,12]. Their main difference is the existence in the C-variant of an on-line TTP of user A. For brevity we describe the B-variant and mainly consider key recovery in B-variant, but the same solutions can be applied to C-variant.

The message flows of the ASPeCT protocol are shown in Figure 1, where all the charging related data field are omitted for simplicity. The detailed description can be found in [11,12]. The notations used in this, and subsequent protocol descriptions, are shown below.

A	the identity of the user
B	the identity of the VASP
TTP_A	the identity of the TTP that user A trusts
g	a generator of a finite group
r_A	a random nonce chosen by user A

r_B	a random nonce chosen by user B
K_{AB}	a secret session key established between A and B
K_A^{-1}	A 's private signature key
A_{Cert}	public key certificate of the user A that is signed by TTP_A
B_{Cert}	public key certificate of the user B that is signed by TTP_A
b	the private key component of the public-private key-agreement key pair of the B
g^b	the public key component of the public-private key-agreement key pair of the B
$\{m\}_{K_A^{-1}}$	the message m signed by the user with his private signature key K_A^{-1}
$\{m\}_{K_{AB}}$	the symmetric encryption of a message m using the session key K_{AB}
h, h_1, h_2, h_3	one-way hash functions

A : user, B : VASP, TTP_A : TTP of A , $K_{AB} = h_1(r_B, g^{br_A})$

1. $A \rightarrow B: g^{r_A}, TTP_A$
2. $A \leftarrow B: r_B, h_2(K_{AB}, r_B, B), B_{Cert}$
3. $A \rightarrow B: \{\{h_3(g^{r_A}, g^b, r_B, B)\}_{K_A^{-1}}, A_{Cert}\}_{K_{AB}}$

Fig.1 The ASPeCT protocol

A generates a random number r_A , computes g^{r_A} , and sends it to B with the identity TTP_A . On receipt of the first token, B generates a random number r_B and computes a session key $K_{AB} = h_1(r_B, (g^{r_A})^b)$. B then sends Token 2 to A . On the receipt of the second token, A computes the key $K_{AB} = h_1(r_B, (g^b)^{r_A})$ and compares the hashed value $h_2(K_{AB}, r_B, B)$ with the one received. If the check succeeds, A generates the signature (shown in Figure 1) and sends Token 3 to B .

3.2 KR Enhanced ASPeCT Protocol

Rantos and Mitchell proposed a KR enhanced ASPeCT protocol [7]. Their strategy was to modify the already designed and well-studied ASPeCT WAKE protocol [11,12] without any overhead at the user end and without introducing any vulnerability.

In their protocol, each entity A and B registers with a KRA, KRA_A and KRA_B in their respective domain. The same TTP is assumed to act both as the certification authority (CA) and the KRA for each entity. Two different KR schemes are proposed that are applicable to both B- and C-variants of the

ASPeCT protocol. For brevity, we describe the B-variant protocol with KR. The application to C-variant ASPeCT protocol is straightforward. The first KR scheme is described in Figure 2.

A : user, B : VASP, TTP_A : TTP of A , $K_{AB} = h_1(r_B, g^{br_A})$
 A : $r_A = f(w_A, s_A)$, $L = (g^{x_A})^{r_A}$

1. $A \rightarrow B$: $g^{r_A}, s_A, \{A\}_L, TTP_A$
2. $A \leftarrow B$: $r_B, h_2(K_{AB}, r_B, B), B_{Cert}$
3. $A \rightarrow B$: $\{\{h_3(g^{r_A}, g^b, r_B, B)\}_{K_A^{-1}}, A_{Cert}\}_{K_{AB}}$

Fig.2 KR enhanced B-variant ASPeCT protocol

The ASPeCT protocol is given a key recovery capability by slightly modifying the way that A 's random nonce, r_A is generated. In the ASPeCT protocol, r_A is chosen at random by A prior to the start of protocol. But in the KR enhanced protocol, r_A is computed as:

$$r_A = f(w_A, s_A)$$

where f is a one-way function, s_A is a one-time random seed, and w_A is a secret value shared between A and KRA_A . The user A registers w_A with his KRA_A during the key recovery registration phase. In order for the KRA_A to be able to compute the value s_A , the user A must send s_A and his own identity A encrypted under $L = (g^{x_A})^{r_A}$, where g^{x_A} is the KRA_A 's public key-agreement key.

In the key recovery phase of A 's domain, intercepted values include s_A , $\{A\}_L$, r_B , and g^b . With these values, KRA_A decrypt $\{A\}_L$ and, from A 's identity, obtain the corresponding secret value w_A , which, in turn, can be used to recompute K_{AB} . In B 's domain, however, the procedure is slightly different. In key recovery registration phase, B escrows his private key b to KRA_B . Thus KRA_B can recover K_{AB} when presented with r_B and g^{r_A} . Other fields from the protocol also must be submitted to KRA to check that the request is within the scope of the warrant. However, we omit these fields for brevity.

To increase flexibility, Rantos and Mitchell [7] proposed that w_A could be a temporary secret, computed as:

$$w_A = f^*(w_A^*, TT)$$

Where f^* is a second one-way function, w_A^* is a long-term secret shared between KRA_A and A , and TT is a time stamp.

The second KR scheme does not require any shared secret between A and KRA_A . The only difference is that the first token is changed as:

1. $A \rightarrow B: g^{r_A}, \{A, r_A\}_L, TTP_A$

where r_A is a random number generated by A . KRA_A can compute K_{AB} directly by decrypting $\{A, r_A\}_L$.

3.3 Modified KR Enhanced ASPeCT Protocol

Nieto et al. identified security flaws with the KR enhanced ASPeCT protocol [7], and proposed a modified KR enhanced ASPeCT protocol [6]. They pointed out that the inclusion of $\{A\}_L$ in Token 1 of the KR enhanced ASPeCT protocol was unnecessary (see Figure 2). In the KR enhanced ASPeCT protocol, A 's identity allows KRA_A to obtain the corresponding secret key w_A . In order to preserve the anonymity of the user, the identity is further encrypted under L . However, the identification of the encrypted data is performed before the actual submission of the KRI to the KRA. Therefore, Nieto et al. dropped the unnecessary $\{A\}_L$ from Token 1 in their modified protocol.

Nieto et al. also pointed out that the mutual authentication property, which was originally provided in the ASPeCT protocol, was not provided in the KR enhanced ASPeCT protocol. They explained how an impersonation attack can be mounted in the KR enhanced ASPeCT protocol in the case w_A is a temporary secret. Further, user A cannot be sure whether the protocol tokens are being exchanged with B or KRA_B , since both B and KRA_B know b . Therefore, in their modified protocol the escrow of b is avoided and the protocol token is changed slightly to protect the impersonation attack. They also enforced the KR protocol by providing a *public KRI validation function* in the domain A . According to their definition, a *public KRI validation function* is a validation function that can be executed to verify the integrity of KRI by anyone using only public available information.

The modified KR enhanced protocol is described in Figure 3.

A : user, B : VASP, TTP_A : TTP of A , $K_{AB} = h_1(r_B, g^{br_A})$
 A : $s_A = (w_A h(g^{r_A}) + r_A) \bmod q$, B : $r_B = f_B(w_B, s_B)$

1. $A \rightarrow B: g^{r_A}, TTP_A$
2. $A \leftarrow B: r_B \oplus g^{br_A}, h_2(K_{AB}, r_B, B), \{s_B\}_{K_{AB}}, B_{Cert}$
3. $A \rightarrow B: \{\{h_3(g^{r_A}, g^b, r_B, B)\}_{K_A^{-1}}, A_{Cert}\}_{K_{AB}}, s_A, s_B$

Fig.3 Modified KR enhanced B-variant ASPeCT protocol

In the description of the protocol, we use the following notation: p a large prime, q a prime with $q|(p-1)$, and g an element in the multiplicative group Z_p^* of order q .

KRI Generation Phase. The user A generates w_A , $1 \leq w_A \leq q - 1$, and shares with KRA_A . The value $\phi_A = g^{w_A}$ is made publicly available. A selects a random integer r_A , $1 \leq r_A \leq q - 1$, and compute $u_A = g^{r_A}$ and calculates s_A as:

$$s_A = (w_A h(u_A) + r_A) \bmod q$$

B generates a random number $r_B = f_B(w_B, s_B)$, where f_B is a one-way function, w_B is a secret value shared with KRA_B , and s_B is a random number.

KR Phase. In A 's domain, KRA_A first computes r_A as:

$$r_A = s_A - w_A h(u_A) \bmod q,$$

Next, KRA_A compute $K_{AB} = h_1(r_B, (g^b)^{r_A})$.

In B 's domain, KRA_B computes g^{br_A} by first calculating

$$r_B = f_B(w_B, s_B)$$

and therefore

$$g^{br_A} = (r_B \oplus g^{br_A}) \oplus r_B.$$

Next, KRA_B computes $K_{AB} = h_1(r_B, g^{br_A})$.

Public KRI Validation Phase in A 's Domain. Given the public data u_A, s_A , and A , a monitoring third party V can check the integrity of the KRI fields generated by A , as follows:

- obtain authentic public value ϕ_A
- compute $c' = h(u_A) \bmod q$
- V resolve the validation process as successful if and only if $g^{s_A} = \phi_A^{c'} u_A$

Problems on Modified KR Enhanced ASPeCT Protocol. In Figure 3, s_A and s_B are withheld until the last token exchange in order to prevent impersonation attack. This insertion of s_A, s_B , and $\{s_B\}_{K_{AB}}$ may occure transimission and computation overhead. Due to the different manner in which B authenticates to A , a *public KRI validation property* cannot be applied in B 's domain in the modified KR enhanced ASPeCT protocol. If r_B and s_B are generated in a similar manner to A , whoever knows r_A can restore w_B , which is secret between B and KRA_B . For example, A knows r_A and he can obtain r_B from the token $r_B \oplus g^{br_A}$. Next, from the following equation w_B can be restored:

$$s_B = (w_B h(g^{r_B}) + r_B) \bmod q.$$

When designing KR schemes, it is common practice to distribute shares among multiple KRAs. However, the KR enhanced ASPeCT protocol specifies only one KRA. Nieto et al. showed simple (but not efficient) example of how to allow multiple KRAs using the modified KR enhanced ASPeCT protocol that doesn't give public KRI validation property.

4 New Key Recovery in WAKE Protocol

In this section, we propose some improved KR enhanced WAKE protocols which improve on the previous protocols described in Section 3. First, we improved Nieto et al.'s protocol [6] by adding the public KRI validation property in domain B . Secondly, we consider multiple KRAs. The use of multiple KRAs is more practical and increases the user's acceptability on the KR system over that of only one KRA. Finally, we considered key recovery on the more general WAKE protocol with the property of *perfect forward secrecy*.

4.1 Public Verifiable Key Recovery in Both A 's and B 's Domains

As already seen in Section 3.3, Nieto et al.'s protocol [6] doesn't provide public KRI validation property in B 's domain. However if we slightly modify the protocol, we can provide this property. In Nieto et al.'s protocol (see Figure 3), user A knows r_B from the Token 2. This eventually makes the public KRI validation in B 's domain to be impossible. Therefore if we modify the session key to $K_{AB} = h_1(g^{br_A+r_B})$ and use g^{r_B} instead of r_B , we can resolve this problem. The proposed protocol (we call this type 1) is described in Figure 4.

A : user, B : VASP, TTP_A : TTP of A , $K_{AB} = h_1(g^{br_A+r_B})$
 A : $s_A = (w_A h(g^{r_A}) + r_A) \bmod q$, B : $s_B = (w_B h(g^{r_B}) + r_B) \bmod q$

1. $A \rightarrow B$: g^{r_A}, TTP_A
2. $A \leftarrow B$: $g^{r_B}, w_B \oplus r_B \oplus g^{br_A}, h_2(K_{AB}, g^{r_B}, B), \{s_B\}_{K_{AB}}, B_{Cert}$
3. $A \rightarrow B$: $\{\{h_3(g^{r_A}, g^b, g^{r_B}, B)\}_{K_A^{-1}}, A_{Cert}\}_{K_{AB}}, s_A, s_B$

Fig.4 Proposed KR enhanced protocol - type 1

In this proposed protocol, the modification of session key, K_{AB} , does not break any property that WAKE protocol should have (see Section 3.1). The user A generate session key K_{AB} by first calculating

$$(g^b)^{r_A}$$

and therefore

$$K_{AB} = h_1(g^{br_A} g^{r_B}).$$

The user B generate session key K_{AB} by first calculating

$$(g^{r_A})^b$$

and therefore

$$K_{AB} = h_1(g^{r_A b} g^{r_B}).$$

KRI Generation Phase. Each user A and B generates w_A and w_B , and shares with KRA_A and KRA_B respectively. The value $\phi_A = g^{w_A}$ and $\phi_B = g^{w_B}$ are made publicly available. The user A selects a random integer, r_A , compute $u_A = g^{r_A}$, and calculates s_A as:

$$s_A = (w_A h(u_A) + r_A) \bmod q$$

Similarly the user B generates r_B , u_B , and s_B .

KR Phase. In A 's domain, KRA_A computes firstly r_A as:

$$r_A = s_A - w_A h(u_A) \bmod q,$$

KRA_A computes $(g^b)^{r_A}$ and computes $K_{AB} = h_1(g^{br_A} g^{r_B})$.

In B 's domain, KRA_B computes firstly r_B as:

$$r_B = s_B - w_B h(u_B) \bmod q,$$

KRA_B computes g^{br_A} from $w_B \oplus r_B \oplus g^{br_A}$. Then KRA_B computes $K_{AB} = h_1(g^{r_B} g^{br_A})$.

Public KRI Validation Phase. Given the public data u_A , s_A , and A , a monitoring third party V can check the integrity of the KRI fields generated by A , by doing the following:

- obtain authentic public value ϕ_A
- compute $c' = h(u_A) \bmod q$
- V resolve the validation process as successful if and only if $g^{s_A} = \phi_A^{c'} u_A$

In B 's case, the same procedure is applied.

Analysis of the Protocol. The user A does not know the real value of r_B . Therefore, a public KRI validation in B 's domain as well as A 's domain can be achieved without any computational overhead at the user's end. In B 's end, additional one modular exponentiation, g^{r_B} , is required. However, this can be precomputed, and usually the computation power of the server is expected to be great. Therefore, this is unlikely to pose serious problems.

4.2 Multiple KRAs

If we use multiple KRAs, the KR protocol can be more efficient in transmission bandwidth. In the proposed protocol (we call this type 2), the user B shares private key-agreement key, b , with multiple KRAs and then the session key, K_{AB} , can be recovered without revealing the value of b in KR phase. Before we describe the proposed protocol, we review verifiable secret sharing (VSS) scheme [13].

VSS Scheme The secret $s \in Z_q$ is distributed to n trusted entities, $T_i, 1 \leq i \leq n$. The dealer selects $(k-1)$ th polynomial $f(x) = f_0 + f_1x + \dots + f_{k-1}x^{k-1}$ that satisfies $f(0) = s$ and computes $s_i = f(x_i)$, where x_i is public number associated with T_i . The s_i and $(g^{f_1}, \dots, g^{f_{k-1}})$ are sent to each T_i , where g is a generator in Z_q . The secret s can be recovered when (for notational convenience the first) k trusted entities are merged:

$$s = \sum_{i=1}^k a_i s_i$$

$$a_i = \prod_{h \neq i} \frac{x_h}{x_h - x_i}$$

Each trusted entity, T_i , could also verify that the received s_i is a valid share. For brevity we omit the validation process of s_i . For a detailed description, see [13].

A : user, B : VASP, TTP_A : TTP of A , $K_{AB} = h_1(r_B, g^{br_A})$
 A : $s_A = (w_A h(g^{r_A}) + r_A) \bmod q$

1. $A \rightarrow B$: g^{r_A}, TTP_A
2. $A \leftarrow B$: $r_B, h_2(K_{AB}, r_B, B), B_{Cert}$
3. $A \rightarrow B$: $\{\{h_3(g^{r_A}, g^b, r_B, B)\}_{K_A^{-1}}, A_{Cert}\}_{K_{AB}}, s_A$

Fig.5 Proposed KR enhanced protocol - type 2

The proposed protocol is described in Figure 5. It is same with ASPeCT protocol without sending s_A in the third token. We consider multiple KRA's only in B's domain for brevity. In the case of multiple KRA's in A's domain, the secret, w_A , which is shared between A and only one KRA in the protocol, has to be shared with multiple KRA's by using a publicly verifiable secret sharing scheme.

KRI Generation Phase. User A generates w_A , $1 \leq w_A \leq q-1$, and shares with KRA_A . The value $\phi_A = g^{w_A}$ is made publicly available. A selects a random integer r_A , $1 \leq r_A \leq q-1$, and compute $u_A = g^{r_A}$ and calculates s_A as:

$$s_A = (w_A h(u_A) + r_A) \bmod q$$

User B shares his private key-agreement key b with KRA_B 's using VSS scheme. In practice, any VSS scheme can be used if the session key K_{AB} can be recovered without revealing the value of b in KR phase.

KR and Public KRI Validation Phase. In B 's side, each KRA_B gives $(g^{r_A})^{a_i b_i}$ and then the value of $g^{r_A b}$ is recovered as:

$$\begin{aligned} g^{r_A b} &= g^{r_A a_1 b_1} g^{r_A a_2 b_2} \dots g^{r_A a_k b_k} \\ &= g^{r_A (a_1 b_1 + a_2 b_2 + \dots + a_k b_k)} \end{aligned}$$

Because the real value of b is not recovered in KR phase, A can be sure only B knows b , i.e. authentication of B to A can be achieved. Further, public KRI validation can be achieved if we use public verifiable secret sharing (PVSS) scheme instead of VSS scheme.

Analysis of the Protocol. The proposed protocol has reduced transmission overhead by dropping $\{s_B\}_{K_{AB}}$ in Token 2 and s_B in Token 3. Further, the symmetric encryption/decryption process, which are needed to encrypt s_B in B and decrypting $\{s_B\}_{K_{AB}}$ in A , is not required. Furthermore public verifiability on both A and B 's domain can be achieved. By using multiple KRA 's, it is more practical and applicable.

4.3 Comparison of Protocols

In this section, we compare existing protocols and our proposed protocols. In Table 1, we calculate the number of modular exponentiation in A 's and B 's side. The number within the bracket refers to the number of the required online modular exponentiation when precomputation is allowed.

Table 1. Comparison of protocols

protocol	modular exp.		additional transmission overhead	public KRI validation
	A	B		
ASPeCT	2(1)	1(1)		
KR enhanced	3(1)	1(1)	$s_A, \{A\}_L$ or $\{A, r_A\}_L$	None
Modified KR enhanced	2(1)	1(1)	$\{s_B\}_{K_{AB}}, s_A, s_B$	A
Proposed type 1	2(1)	2(1)	$\{s_B\}_{K_{AB}}, s_A, s_B, g^{r_B}$	A, B
proposed type 2	2(1)	1(1)	s_A	A, B

The proposed protocol type 1 requires one more modular exponentiation in B 's side. However this can be precomputed and usually the computation power of B is expected to be great. So it can be serious problem. The proposed protocol type 2 reduces the transmission overhead remaining same computation load. Also both the proposed protocols provide public KRI validation property in A 's and B 's side.

4.4 Problem of Perfect Forward Secrecy

The original ASPeCT protocol [11] described in Section 3.1 does not provide *perfect forward secrecy*. A protocol is said to have *perfect forward secrecy* if compromise of long-term keys does not compromise past session keys (ch.12, [14]).

In the original ASPeCT protocol, if the long term key, b , is disclosed, then all the previous session keys can be calculated by keeping g^{r_A}, r_B which used in previous protocol. To solve this perfect forward secrecy problem, the generation method of the session key must be changed. We improved the original ASPeCT protocol by changing the session key to $K_{AB} = h(g^{r_A r_B})$ and using g^{r_B} instead of r_B in the protocol.

The modified protocol is shown in Figure 6. The g^{br_A} is inserted in the hashed value $h_2(K_{AB}, g^{r_B}, g^{br_A}, B)$ of Token 2 to provide the authentication of B to A . This protocol is similar to the protocols in [15], although the protocols were developed independently.

A : user, B : VASP, TTP_A : TTP of A , $K_{AB} = h_1(g^{r_A r_B})$

1. $A \rightarrow B: g^{r_A}, TTP_A$
2. $A \leftarrow B: g^{r_B}, h_2(K_{AB}, g^{r_B}, g^{br_A}, B), B_{Cert}$
3. $A \rightarrow B: \{\{h_3(g^{r_A}, g^b, g^{r_B}, B)\}_{K_A^{-1}}, A_{Cert}\}_{K_{AB}}$

Fig.6 Modified ASPeCT protocol with perfect forward secrecy property

Considering the key recovery system in this modified protocol. Each KRA, KRA_A and KRA_B , must know r_A or r_B in order to recover K_{AB} in KR phase. We can achieve this easily by the method used in previous protocols. The resulting protocol is described in Figure 7.

A : user, B : VASP, TTP_A : TTP of A , $K_{AB} = h_1(g^{r_A r_B})$
 $A: s_A = (w_A h(g^{r_A}) + r_A) \bmod q$, $B: s_B = (w_B h(g^{r_B}) + r_B) \bmod q$

1. $A \rightarrow B: g^{r_A}, TTP_A$
2. $A \leftarrow B: g^{r_B}, h_2(K_{AB}, g^{r_B}, g^{br_A}, B), \{s_B\}_{k_{AB}}, B_{Cert}$
3. $A \rightarrow B: \{\{h_3(g^{r_A}, g^b, g^{r_B}, B)\}_{K_A^{-1}}, A_{Cert}\}_{K_{AB}}, s_A, s_B$

Fig.7 Proposed KR system on the modified ASPeCT protocol

5 Conclusion

In this paper, we proposed improved KR enhanced WAKE protocols that are both efficient in transmission and have improved security. First, we provided a publicly verifiable key recovery protocol that holds on both sides that communicate each other. Secondly, we improved the efficiency of protocol in the case of multiple key recovery agents. In general, the use of multiple key recovery agents helps increasing the user's acceptability on the key recovery system. Finally, we modified the existing ASPeCT WAKE protocol to add the perfect forward secrecy. We also gave this modified protocol to have key recovery property.

Acknowledgment

The authors wish to thank anonymous referees for their critical comments.

References

1. U. Black, "Third Generation Mobile Systems (TGMSs)", *Second Generation Mobile & Wireless Networks*, Prentice Hall, 1999
2. T. Ojanpera and R. Prasad, "IMT-2000 Applications", *Wideband CDMA for Third Generation Mobile Communication*, T. Ojanpera and R. Prasad (ed.), Artech House Publishers, pp.65-76, , 1998
3. K. Buhanal et al., "IMT-2000: Service Providers Perspective", *IEEE Personal Communications*, August 1997
4. ETSI SMG10, "Universal Mobile Telecommunications System (UMTS): Security Requirement", *Draft UMTS 33.21 version 2.0.0*, Feb., 1999
5. NIST, "Escrow Encryption Standard (EES)", *Federal Information Processing Standard Publication (FIPS PUB)* 185, 1994
6. J. Nieto, D. Park, C. Boyd, and E. Dawson, "Key Recovery in Third Generation Wireless Communication Systems", *Public Key Cryptography-PKC2000*, LNCS 1751, pp.223-237, 2000
7. K. Rantos and C. Mitchell, "Key Recovery in ASPeCT Authentication and Initialization of Payment protocol", *Proceedings of ACTS Mobile Summit*, Sorrento, Italy, June 1999
8. ETSI TC Security, "Specification for Trusted Third Party Services: Part1 Key Management and Key Escrow/Recovery", *DEN/SEC-003001x, Draft Version 1.0 (edition2)*, 11th Nov, 1997
9. ETSI TC-STAG, "Security Techniques Advisory Group (STAG); Definition of User Requirements for Lawful Interception of telecommunications: Requirements of the Law Enforcement Agencies", *ETR 331*, December 1996.
10. B. Pfizmann and M. Waidner, "How to Break Fraud-Detectable Key Recovery", *Operating Systems Review*, 21, 1998, pp.23-28
11. G. Horn and B. Preneel, "Authentication and payment in future mobile systems", *Computer Security - ESORICS'98*, LNCS, 1485, pp.277-293, 1998
12. ACTS AC095, ASPeCT Deliverable D02, "Initial Report on Security Requirements", *AC095/ATEA/W21/DS/P/02/B*, Feb., 1997, Available from <http://www.esat.kuleuven.ac.be/cosic/aspect/>

13. T. Pederson, "Distributed provers with application to undeniable signatures", *Advances in Cryptology-Eurocrypt'91*, LNCS vol. 963, pp.222-242, 1991
14. A. Menezes, P. Oorschot, and S. Vanstone, *Handbook of Applied Cryptography*, 1997
15. DongGook Park, Colin Boyd and Sang-Jae Moon, "Forward Secrecy and Its Application to Future Mobile Communications Security", *PKC2000*, Springer-Verlag, pp.433-445, 2000