Forward Secrecy and Its Application to Future Mobile Communications Security*

DongGook Park [1, 2], Colin Boyd [2] and Sang-Jae Moon[3] **

[1] Korea Telecom, Access Network Laboratory,
17 WooMyeon-Dong, SeoCho-Gu, 137-792, Seoul, Korea
park@isrc.qut.edu.au
[2] Queensland University of Technology, Information Security Research Centre,
2 George Street, GPO Box 2434, Brisbane, Queensland 4001, Australia
boyd@fit.qut.edu.au
[3] Kyungpook National University, School of Electronic & Electrical Engineering,
1370, SanKyuk-Dong, Pook-Gu, Taegu, 702-701, Korea
sjmoon@ee.knu.ac.kr

Abstract. Perfect forward secrecy, one of the possible security features provided by key establishment protocols, concerns dependency of a session key upon long-term secret keys (symmetric or asymmetric). The feature promises that even if a long-term private key is disclosed to any adversary, the session keys established in the protocol runs using the long-term key would not be compromised. The importance of this kind of belief may differ greatly among application environments, in terms of both communication types and different communicating entities. We describe two generic prototypes of protocols which bring forward secrecy to security protocols. We note that future generation mobile communication environment will be filled with diverse types of communication users and data. The security protocol in a prominent future mobile system, UMTS, was originally designed without any consideration of perfect forward secrecy. We consider modified protocols to provide this property.

1 Introduction

The use of session keys allows different sessions to be independently secure so that if one session key becomes compromised then this should not affect any other session key. Long term keys (symmetric or asymmetric) are used to establish session keys and so must be protected much more securely than session keys. Although it may be

* This research is part of the co-operative project "Security Technologies in Wireless Communications" between Queensland University of Technology and Korea Telecom

** This work was done while this author was a visiting research fellow at the Information Security Research Centre, Queensland University of Technology, Brisbane, Australia.

H. Imai, Y. Zheng (Eds.): PKC 2000, LNCS 1751, pp. 433-445, 2000.

an unlikely event, the consequences of the compromise of a long-term key should be considered.

A protocol is said to provide *forward secrecy* if the compromise of long-term keys does not compromise past session keys that have been established before the compromise of the long-term key [3]. The idea of forward secrecy seems to have been coined by Günther [4] in connection with an identity based protocol he proposed. In fact Günther used the term *perfect* forward secrecy; however since the word 'perfect' has connotations with unconditional security which are not relevant here, we will use the simpler term in common with a number of other authors. It should be noted that there seems to be a disagreement in the definition of forward secrecy because we can find a literature where forward secrecy is intended to mean that a secret encryption key used in a session must be securely discarded after the session to prevent an adversary from obtaining the encryption key in any way and eavesdropping any future sessions protected by the same encryption key [7]. In this paper, however, we use only the former definition of forward secrecy, which appears more generally agreed one.

We also note that there is a somewhat similar concept called forward *security* to address another significance of losing long-term private keys [9]. A signature scheme with forward security protects users from the threat of *signature forgery* in case their signature keys have been compromised. The basic idea to implement forward security is to update the signature key itself frequently to reduce the risk of key exposure. This may contributes also to the forward *secrecy* when it is the case that the signature key is used for authentication and key establishment as well, because the limited longevity of the signature key reduces the risk of relevant session key compromise down to the lifetime of the signature key. Still, however, forward security is not a sufficient condition for forward secrecy considering that the disclosure of the signature key would compromise any session keys computed using the signature key. In other words, if we confine our focus on a particular long-term private key (however long it lives), then it is only forward secrecy that protects the relevant session keys from the compromise of the long-term private key. With this in mind, we argue that the essential characteristic of forward secrecy is orthogonal to that of forward security. It should be noticed, however, that there seems to be rather loose distinction, which reflects, as we have already described, the fact that forward security may be regarded as a weak alternative to forward secrecy in a practical sense [2]. In this paper, however, we confine our discussion only to the forward secrecy in distinction to forward security.

It has long been known that protocols based on the Diffie-Hellman key agreement protocol [12] will usually provide forward secrecy. This is because the long-term keys are normally used only to authenticate messages and not to encrypt them. This property is widely regarded as a useful extra security feature of Diffie-Hellman based protocols, since most other protocols do not possess it. In the next section, however, we will see that forward secrecy does not require any particular type of cryptosystem such as Diffie-Hellman. Considering the significance of forward secrecy, it is rather surprising that this security feature has almost never been given a proper effort to understand it.

Unlike many other goals of security protocols, forward secrecy may have to be treated more practically. Its significance in the real applications dramatically varies through both angles of communication types and user types. In the communications between a private user and a public commercial entity, it is more the user than the commercial entity that is concerned about confidentiality for the past communications, and hence is more concerned about forward secrecy. On the other hand, forward secrecy usually requires some additional computations of asymmetric key cryptography, and hence might be a quite expensive cryptographic service in some types of communications, for example, voice communications or message broadcasting in some value added services.

In the next section two prototype constructions are presented for protocols providing forward secrecy. The first is based on the Diffie-Hellman protocol, while the second can be used with any chosen asymmetric encryption scheme. Section 3 then discusses the notion of partial forward secrecy and presents prototype constructions. In sections 4 and 5 we examine a prominent proposed protocol for third generation mobile communications, show that it does not provide forward secrecy, and show how it may be modified to do so.

Now, we summarize below the notation used in this paper.

$\mathbf{A}$, $\mathbf{B}$: the identities of the two peer principals involved in a particular session

$\mathbf{g}$: a generator of a finite group

$\mathbf{r_A}$, $\mathbf{r_B}$: random nonces chosen by the principals, $\mathbf{A}$ and $\mathbf{B}$, respectively

$\mathbf{K_{AB}}$: a secret session key established between two principals, $\mathbf{A}$ and $\mathbf{B}$

$\mathbf{b}$: the private key component of the public-private key-agreement key pair of the principal, $\mathbf{B}$

$\mathbf{g^b}$: the public key component of the public-private key-agreement key pair of the principal, $\mathbf{B}$

$\{\mathbf{m}\}_{\mathbf{K_A}^{-1}}$: the message $\mathbf{m}$ signed by the user with his/her private signature key $\mathbf{K_A}^{-1}$

$\{\mathbf{m}\}_{\mathbf{K_{AB}}}$: the symmetric encryption of a message $\mathbf{m}$ using the session key $\mathbf{K_{AB}}$

$\mathbf{h}$: a common hash function agreed between the two principals, $\mathbf{A}$ and $\mathbf{B}$

$\mathbf{e_A}$, $\mathbf{n_A}$: a temporary asymmetric public key pair of $\mathbf{A}$ in the RSA context

To make some description clear, we also use novel but easy-to-understand notation, which we want to help capture some abstract property with regard to forward secrecy, as follows.

APriKey, **APubKey**: a *long-term certified* authentic asymmetric key pair (private key, **APriKey** and public key, **APubKey**) of a principal **A**

APriKeyX, **APubKeyX**: a *temporary uncertified* asymmetric key pair (private key, **APriKeyX** and public key, **APubKeyX**) of a principal **A**

APubKey{m}: a message **m** encrypted under the *long-term authentic* public key, **APubKey** of **A** for data confidentiality. The message **m** can only be retrieved by the owner of the **APriKey** (i.e., the principal **A**)

APubKeyX{m}: a message **m** encrypted under the *temporary* public key of **A** for data confidentiality. The message **m** can only be retrieved by the owner of the **APriKeyX** (i.e., the principal **A**)

2 Two Prototypes for Forward Secrecy

Almost all authors discuss only Diffie-Hellman based key establishment protocols as examples providing forward secrecy and there seems to be an implicit assumption that these are the only possible examples. Consequently, there seems to be a tendency in protocol design that only Diffie-Hellman type key agreement functions are considered for forward secrecy implementation [1]. In this section, two prototype constructions are presented, one based on the special algebraic properties of Diffie-Hellman key exchange, and the other which can work with any asymmetric encryption scheme. The second prototype addresses the question: *is there any other cryptographic algorithm than Diffie-Hellman satisfying forward secrecy?* We do not try to answer this question directly. Instead, we enlarge our scope of the search for forward secrecy to cover cryptographic *protocols*. Then, the answer to the above question is: “Yes. As many as the number of different asymmetric cryptographic algorithms.”

We take a rather over-simplified approach to ease the capture of the very basic property which enables forward secrecy. Any other security goals such as entity authentication and key authenticity will be entirely omitted in the following descriptions. We believe that by taking this approach we can understand the mechanism of forward secrecy more clearly. Furthermore, these building blocks of forward secrecy mechanisms without any extra property would be more effectively integrated into authentication and key establishment protocols.

We first investigate what kind of property in Diffie-Hellman type protocols present us forward secrecy. Two principals A and B select random secrets r_A and r_B, compute g^{r_A} and g^{r_B} respectively, and exchange them over an unsecured channel.

$$
\begin{aligned}
&1.\ A \rightarrow B:\ g^{r_A}\\
&2.\ A \leftarrow B:\ g^{r_B}\\
&A:\ K_{AB} = (g^{r_B})^{r_A}\\
&B:\ K_{AB} = (g^{r_A})^{r_B}
\end{aligned}
$$

Fig. 1. Basic Diffie-Hellman key agreement

This basic protocol for key agreement is integrated into more sophisticated protocols which use long-term asymmetric keys of principals to provide entity authentication. The famous STS (Station-to-Station) protocol [13] is such an example (Fig. 2).

In the STS protocol, the session key establishment is exactly in the same form as that of basic Diffie-Hellman protocol, and does not depend on the long term asymmetric keys of A and B. Therefore a future possible disclosure of the long-term keys of A, B or both does not lead to the compromise of the session key. It should be noted that there are a number of other methods to incorporate authentication into basic Diffie-Hellman and which also provide forward secrecy. Some examples may be found in the IEEE P1363 draft standard [8].

1. A → B: g^{r_A}

2. A ← B: g^{r_B}, $\{\{g^{r_B}, g^{r_A}\}_{K_B^{-1}}\}_{K_{AB}}$

3. A → B: $\{\{g^{r_A}, g^{r_B}\}_{K_A^{-1}}, A\}_{K_{AB}}$

A, B: $K_{AB} = g^{r_A r_B}$

Fig. 2. Station-to-Station (STS) protocol

Now, we try to describe the very general property of Diffie-Hellman protocol in a quite abstract way. Using the abstract notation shown in Section 1, we can re-describe the Diffie-Hellman protocol as follows, which we call the first prototype for forward secrecy (Fig. 3).

1. A → B: APubKeyX

2. A ← B: BPubKeyX

A: K_{AB} = F(BPubKeyX, APriKeyX)
B: K_{AB} = F(APubKeyX, BPriKeyX)

where F is a common key agreement function with the following property:

F(BpubKeyX, APriKeyX) = F(APubKeyX, BPriKeyX)

Fig. 3. An abstract level description of basic Diffie-Hellman key agreement

The public components, **APubKeyX** and **BPubKeyX** of the temporary uncertified asymmetric key pairs of both principals are transmitted in clear, and no long-term key is involved in the calculation of the session key. Only the principals **A** and **B** which are in possession of secret private components of the short-term asymmetric key pairs, i.e. **APriKeyX** and **BPriKeyX**, respectively, can derive the true authentic session key K_{AB}. Another rather trivial requirement for forward secrecy is that the short-term secrets of both parties should be securely discarded after the completion of the corresponding session.

Now, we consider another alternative protocol for forward secrecy, which is also described using the same abstract level notation (Fig. 4).

1. A → B: APubKeyX

2. A ← B: APubKeyX$\{r_B\}$

A, B: K_{AB} = h(APubKeyX, r_B) or alternatively
$K_{AB} = r_B$ when key *transport* scheme is preferred.

Fig. 4. An abstract level description of an alternative protocol for forward secrecy

This second prototype for forward secrecy is based on *confidentiality of a random nonce* r_B chosen by the principal **B**. Here, the temporary public key **APubKeyX** is used for temporary encryption of r_B. On receipt of this encrypted r_B, the principal **A** can decrypt and recover r_B using **APriKeyX** as the decryption key. This ephemeral characteristic of the encryption of a random secret is the source of forward secrecy.

The critical difference between the previous prototype derived from Diffie-Hellman scheme and this one is that the former depends on the special feature of a key agreement function **F** with an elegant symmetric property, whereas the latter relies upon confidentiality using a temporary public key. Therefore, the first prototype can be implemented only by a cryptosystem which satisfies the requirement of the key agreement function **F**. Presently, only Diffie-Hellman key exchange in various suitable groups is known to be such a system.

On the other hand, the second prototype can easily be applied to *any asymmetric cryptosystem*, which does not have to be Diffie-Hellman system. The random nonce r_A for temporary encryption from **A** to **B** may also be used as a challenge value for **A** to authenticate **B**. The response value then has to be an indication that **B** has used its own secret private key to generate it.

It is interesting to note that the second prototype can be deployed into both key *agreement* and *transport* schemes, unlike the first one which allows only key agreement.

We consider two instance protocols which are easily derived from the second prototype. The following protocols are an application of the prototype to discrete log based cryptosystem (Fig. 5)and RSA cryptosystem (Fig. 6) respectively.

1. A → B: g^{r_A}
2. A ← B: $g^{r_A r_B}$

A: $K_{AB} = h(g^{r_A}, g^{r_B}) = h(g^{r_A}, (g^{r_A r_B})^{r_A^{-1}})$

B: $K_{AB} = h(g^{r_A}, g^{r_B})$

Fig. 5. Discrete log cryptography based implementation of the second prototype

1. A → B: e_A, n_A
2. A ← B: $(r_B)^{e_A} \bmod n_A$

A, B: $K_{AB} = h(e_A, r_B)$

Fig. 6. RSA cryptography based implementation of the second prototype

Even though the example protocol shown in Fig. 5 is very similar to the original Diffie-Hellman protocol, it should be noticed that this is just an example of the prototype, and any different kind of public key mechanisms can be adopted to provide forward secrecy. Of course, as with the prototype protocols, these examples must be used together with authentication in order to achieve a *secure* protocol. It is worth noting that the authentication required can be achieved through either symmetric or asymmetric cryptography, so it should not be assumed that forward secrecy requires asymmetric long-term keys. Indeed, several password based protocols [5], [11], in which the long-term key is a (short) shared secret, or a value derived from it, have been proposed and provide forward secrecy. Whilst it may be of questionable value to consider the efficiency of incomplete protocols, it is also interesting to note that the example based on RSA can be more efficient than that based on discrete log for entity B if the value e_A is chosen to be a small value such as is often recommended. However, we must note that a new temporary RSA key pair must be generated by A for each session.

In fact, we have noticed that basically the same concept of forward secrecy based upon RSA had already been described by Wiener [10]. His description, however, still only concerns the implementation of forward secrecy in terms of a particular cryptographic *algorithm* like RSA rather than *protocols* using any asymmetric cryptography. This is likely to lead to a rather misleading conclusion that RSA (or even the second prototype itself) is always a more expensive way to achieve forward secrecy than Diffie-Hellman (or the first prototype). Furthermore, Wiener seems to ignore the possibility that the use of ephemeral encryption for forward secrecy (i.e., the second prototype above) can be just as efficient as the first prototype even in the case of discrete-log cryptosystems. As will be made clear later in this paper, however, both prototypes may require the same cost for forward secrecy to be implemented into key establishment protocols. The choice of the suitable prototype for forward secrecy may be made according to the application scenario rather than the particular cryptographic primitives used in the protocols.

It is an obvious question to ask whether there are other protocols providing forward secrecy that do not fit into the two prototype constructions discussed in this section. It seems that forward secrecy can be provided only through use of a *one-way function*, so that later the session key cannot be recovered because this function cannot be inverted. Because of its special algebraic properties, exponentiation modulo a prime is a suitable one-way function. Naturally other similar functions, such as multiplication in an elliptic curve group can equally be used, and these provide natural generalisations of the original Diffie-Hellman key exchange protocol. Therefore any other one-way function with suitable algebraic properties may be candidates for alternative protocols providing forward secrecy, although there are no such functions currently available to our knowledge. The second prototype makes use of the trapdoor one-way function underlying any asymmetric cryptosystem. In this case the function cannot be inverted once the trapdoor is deleted. Therefore, although we offer no further evidence, we conjecture that forward secrecy can only be provided by protocols which either have similar algebraic properties as modular exponentiation (prototype one) or use trapdoor one-way functions (prototype two).

3 Two Prototypes for *Partial* Forward Secrecy

It seems that we do not need to insist only upon *complete* forward secrecy in some application environments. Instead, we may consider a sort of *imperfect* or *partial* forward secrecy if it is more computationally effective. There may be, for example, a particular communication type where only one of two peer entities is really concerned about confidentiality of the past data and/or the other entity's long-term private key is more likely to be compromised. In this situation, it may be reasonable to consider protection against the long-term key compromise of only one principal of two peers, which we may call *partial* forward secrecy as opposed to the usual *complete* forward secrecy.

The same reasoning for prototypes of forward secrecy can be directly applied to identify the prototypes for partial forward secrecy. The two prototypes shown in Fig. 7 and Fig. 8 are partial forward secrecy analogues of the previous ones.

1. A ← B: BPubKeyX

A: K_{AB} **= F(BPubKeyX, APriKey)**
B: K_{AB} **= F(APubKey, BPriKeyX)**

where **F** is a common key agreement function with the following property:

F(BPubKeyX, APriKey) = F(APubKey, BPriKeyX)

Fig. 7. The first prototype for partial forward secrecy

1. A ← B: APubKey{r_B**}**

A, B: K_{AB} **= h(m,** r_B**)** or alternatively K_{AB} = r_B when key *transport* scheme is preferred.

Notation :

m: an optional key input data to the hash function (**m** should be agreed previously between **A** and **B** in the more broad context of the relevant whole key establishment protocol which includes this forward secrecy scheme)

Fig. 8. The second prototype for partial forward secrecy

Here in both prototypes, we assume that **B** has an authentic copy of **A**'s long-term public key, **APubKey**. The only difference from *complete* versions is that **A**'s short-term private key is replaced with its long-term authentic public key. The computation procedures of the session key, K_{AB} take **A**'s long-term public key, but not that of the principal **B**. Therefore, the compromise of **B**'s long-term private key alone does not cause the compromise of the session key. It should be noticed, however, that these prototypes, including both partial and complete cases, do not guarantee any other security goals except forward secrecy. It is the whole integration of elements in a protocol that provides the ultimate security goals including entity authentication.

4 Future Mobile Communications and Forward Secrecy

Wireless mobile communications are notorious with so-called *usage fraud* and *eavesdropping* because of its radio media characteristics. Although rather limited, user authentication and data encryption is provided over the air interface of the current generation of digital cellular systems such as Global System for Mobile communications

(GSM). Their security technologies came from conventional symmetric cryptography. It is expected that future generation mobile communication systems such as Universal Mobile Telecommunications System (UMTS) will include asymmetric cryptography to enlarge their security features. The ASPeCT project for research and development of security technologies to be used in UMTS, has proposed a public key based authentication and key establishment protocol with both air interface between the user and the network, and the user-to-Value Added Service Provider (VASP) interface in mind [6]. User to VASP communications will be much like the present wireline based internet communication using web browsers. According to the increase in the number and diversity of VASPs, the trust relations between communicating parties will be dramatically complicated and the deployment of the required public key infrastructure would be a great challenge.

$\mathtt{A}$: user, $\mathtt{B}$: network or VASP

1. $\mathtt{A} \rightarrow \mathtt{B}: g^{r_A}$
2. $\mathtt{A} \leftarrow \mathtt{B}: r_B,\ h2(K_{AB},\ r_B,\ B)$
3. $\mathtt{A} \rightarrow \mathtt{B}: \{\{h3(g^{r_A},\ g^b,\ r_B,\ B,\ K_{AB})\}_{K_A^{-1}},\ A\ \}_{K_{AB}}$

$\mathtt{A},\ \mathtt{B}: K_{AB} = h1(r_B,\ g^{br_A})$

Notation :

$h1,\ h2,\ h3$: one-way hash functions specified for ASPeCT protocol

Fig. 9. ASPeCT protocol for authentication and key establishment in UMTS

The ASPeCT protocol is basically the same as the Station-to-Station protocol in that both protocols use the same challenge-response mechanism, i.e., **A** and **B** challenge each other with random nonces (g^{r_A} and r_B in this protocol) exchanged in clear and calculate responses using private keys (K_A^{-1} and b respectively in this protocol). In this protocol, however, it should be noted that the second message in the protocol does not contain any signature by **B**. The third message includes the signature by **A** like STS protocol to accommodate non-repudiation requirement for user-network or user-VASP applications.

Significant effort was expended to make this protocol satisfy a demanding set of security goals and computation efficiency at the same time. Through a saving of the non-repudiation property of **B** to **A**, the resultant computational load is significantly lower than that of STS protocol. One additional feature omitted from this protocol is forward secrecy which is supported in STS protocol. This difference comes from the session key generation of the two protocols. The ASPeCT protocol uses a similar but subtly different method from the Diffie-Hellman key computation. STS protocol complies with the original Diffie-Hellman form $g^{r_A r_B}$ where two terms of the exponents are the random nonces generated within two principals. Instead, ASPeCT protocol uses only one nonce r_A and the private key b of **B**. In this way, the protocol saves some public key based computations and succeeds in turning on-line exponentiation within the user terminal **A** into off-line because **A** may take the advantage of the pre-knowledge of **B**'s public key g^b in most cases. For the sake of simplicity, we only refer to [6] for detailed security analysis of the protocol.

If the long term private key **b** of the VASP (or network) is compromised and all the protocol transcripts for a particular session are recorded (for the knowledge of g^{r_A} and r_B) by an attacker, the session key for the session is easily disclosed to the attacker, and so no forward secrecy is provided in this protocol. Of course, the disclosure of the private key of the mobile side alone does not lead to the disclosure of the session key (so partial forward secrecy is satisfied), but in that case, the real problem is more authentication rather than forward secrecy. Moreover, *forward secrecy concerns the user who cannot ensure that the private key of the VASP would not be compromised.*

It is surprising that the ASPeCT analysis of protocols has not considered the issue of forward secrecy at all. The likelihood of compromise of a long-term key is difficult to assess. With regard to a trusted network base station this probability may be regarded as sufficiently low. However, when it concerns any VASP, users may not have confidence that long term keys are sufficiently secure. It should be noted that if the long-term key of a VASP becomes compromised then all transactions made with that VASP during the lifetime of that key may be available to an eavesdropper, unless forward secrecy has been provided. Long term keys may have lifetimes of several months, so such an attack could be very attractive to the attacker.

5 Modification of UMTS Security Protocol for Forward Secrecy

We propose two alternatives that are modified from the ASPeCT protocol for UMTS. They require more modulo exponentiations but guarantee forward secrecy. The first protocol is derived from the first prototype (Diffie-Hellman) described above in the paper.

A: user, **B**: network or VASP

1. $A \rightarrow B$: g^{r_A}
2. $A \leftarrow B$: g^{r_B}, $h2(K_{AB}, g^{r_B}, B)$
3. $A \rightarrow B$: $\{\{h3(g^{r_A}, g^{b}, g^{r_B}, B, K_{AB})\}_{K_A^{-1}}, A\}_{K_{AB}}$

A, B: $K_{AB} = h(g^{r_A r_B}, g^{b r_A})$

Fig. 10. A modified ASPeCT protocol to provide forward secrecy based on the first prototype

The only difference between this modified version and the original ASPeCT protocol is that g^{r_B} is delivered from **B** to **A** instead of r_B, and $g^{r_A r_B}$ is used as a key input to the hash function instead of r_B. The authentic key establishment in the original protocol comes from the fact that (1) from the viewpoint of **B,** a key input g^{r_A} from **A** is included in the signed message of **A** and the fresh nonce r_B of **B**'s own is used for key computation; (2) from $\mathtt{A}$'s viewpoint, the authentic copy of $\mathtt{B}$'s private key $\mathtt{b}$ and $\mathtt{A}$'s own fresh nonce $\mathtt{r_A}$ are used together to compute the session key; and (3) from any attacker's viewpoint, he cannot compute a key input $\mathtt{g^{br_A}}$ with the knowledge of both $\mathtt{g^b}$ and $\mathtt{g^{r_A}}$.

Now, the replacement of $\mathtt{r_B}$ with $\mathtt{g^{r_B}}$ does not affect the protocol with respect to the above three considerations, and helps both principals establish a secret session key satisfying forward secrecy. Note that the compromise of the long-term private key does not lead to the disclosure of the session key due to the inclusion of a Diffie-Hellman form, $\mathtt{g^{r_A r_B}}$ in the key computation. The computational cost for forward secrecy in the user ($\mathtt{A}$) side is one additional exponentiation to calculate $\mathtt{g^{r_A r_B}}$ (still computationally more effective than the STS protocol), and in the network or VASP side, two additional exponentiations to compute $\mathtt{g^{r_A r_B}}$ and $\mathtt{g^{r_B}}$ (roughly the same computational cost as the STS protocol). The term $\mathtt{g^{br_A}}$ is still required in the key computation because this term plays the essential role in authentication of $\mathtt{B}$ to $\mathtt{A}$.

The second protocol is derived from the second prototype for forward secrecy using confidentiality.

$\mathtt{A}$: user, $\mathtt{B}$: network or VASP

1. $\mathtt{A \rightarrow B:\ g^{r_A}}$
2. $\mathtt{A \leftarrow B:\ g^{r_A r_B},\ h2(K_{AB},\ g^{r_A r_B},\ B)}$
3. $\mathtt{A \rightarrow B:\ \{\{h3(g^{r_A},\ g^{b},\ g^{r_A r_B},\ B,\ K_{AB})\}_{K_A^{-1}},\ A\ \}_{K_{AB}}}$

$\mathtt{A,\ B:\ K_{AB} = h(g^{r_B},\ g^{br_A})}$

Fig. 11. Another modified ASPeCT protocol to provide forward secrecy based on the second prototype

In this variant, $\mathtt{r_B}$ in the second message and the key computation of the original protocol is replaced by $\mathtt{g^{r_A r_B}}$ and $\mathtt{g^{r_B}}$, respectively. This modification does not compromise the protocol with respect to the three considerations as described for the first variant, and enables both principals to establish a secret key satisfying forward secrecy in a different way from the first variant. Retrieval of the term $\mathtt{g^{r_B}}$ in the $\mathtt{A}$ side requires the knowledge of $\mathtt{r_A}$ which can be viewed as a private component of the temporary asymmetric pair ($\mathtt{r_A}$, $\mathtt{g^{r_A}}$). The lifetime of the temporary asymmetric key pairs and $\mathtt{r_B}$ should be at most as long as the corresponding session, and hence $\mathtt{g^{r_B}}$ and the session key is free from the compromise of the long-term private keys of $\mathtt{A}$ and $\mathtt{B}$. In other words, the secure deletion of the $\mathtt{r_A}$ and $\mathtt{r_B}$ in both sides after the session thwarts any effort to retrieve the value of $\mathtt{g^{r_B}}$ because it was delivered to $\mathtt{A}$ after encrypted using the temporary public key $\mathtt{g^{r_A}}$ of $\mathtt{A}$.

Here again, compared to the original ASPeCT protocol, the computational cost for $\mathtt{A}$ is one more modulo exponentiation to retrieve $\mathtt{g^{r_B}}$ from $\mathtt{g^{r_A r_B}}$, and for $\mathtt{B}$, two more exponentiations to generate $\mathtt{g^{r_B}}$ and $\mathtt{g^{r_A r_B}}$, which is exactly the same as the first variant for forward secrecy. This clearly shows that the choice of a particular prototype of forward secrecy does not necessarily lead to more expensive or cheaper implementation.

These more expensive versions do not seem likely to find application in the air interface between the user and the network. For this interface is strictly limited by call set-up delay requirements and the possibility of the network private key compromise would be much lower than in the case of VASPs. On the other hand, user to VASP communication looks like a quite feasible target for the protocols because the key compromise problem of VASP would never be negligible and some transaction delay may be considered not so critical in that kind of communications.

Flexibility, or multilevel features, in future mobile security services cannot be over-emphasized considering multimedia features and the complex trust relationships. The alternative protocols described above have the same basic format as the original ASPeCT protocol and hence can be integrated together into a *multilevel featured set* of security protocols.

6 Conclusion

We have surveyed and identified two *general prototypes* for forward secrecy: the first one depending on a particular property of key agreement functions **F** and the second one exploiting confidentiality by temporary asymmetric key pairs. We have also applied these two prototypes to the ASPeCT protocol for UMTS and proposed two alternative variations which provide forward secrecy. They require one more modulo exponentiation in the user side and even more computational cost in the network or VASP side when compared to the original ASPeCT protocol. Either of the two modified protocols, however, can be integrated together with the original protocol into a more flexible *set* of protocols which enables a selective protocol usage depending on the required security goals.

References

1. A. Aziz, “SKIP Extension for Perfect Forward Secrecy”. Available from `http://www.skip-vpm.org/wetice98/HacknSlash.html`
2. A. Aziz, et al., “Simple Key-Management for Internet Protocols (SKIP)”. Available from `http://www.tik.ee.ethz.ch/~skip/SKIP.html`
3. A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1997, p. 496.
4. C. Günther, “An Identity-based Key-exchange Protocol”, *Advances in Cryptology - Eurocrypt’89*, Springer-Verlag, 1990, pp. 29-37.
5. D. P. Jablon, “Strong Password-Only Authenticated Key Exchange”, *ACM Computer Communications Review*, October, 1996, pp.5-26.
6. G. Horn and B. Preneel, “Authentication and payment in future mobile systems”, *Computer Security - ESORICS'98*, Lecture Notes in Computer Science, 1485, 1998, pp. 277-293.

7. H. Abelson et al., “The Risks of Key Recovery, Key Escrow, and Trusted Third-Party Encryption”, *A Report by an Ad Hoc Group of Cryptographers and Computer Scientists*, 1998. Available from `http://www.cdt.org/crypto/risks98/`
8. IEEE P1363/D9 Standard Specifications For Public Key Cryptography, February. 1999.
9. M. Bellare and S.K. Miner, “A Forward-Secure Digital Signature Scheme”, *Advances in Cryptology - Crypto’99*, Springer-Verlag, 1999.
10. M. Wiener, “Performance Comparison of Public-Key Cryptosystems”, *Cryptobytes*, vol. 1, no. 2, RSA Laboratories, 1998.
11. T. Wu, “The Secure Remote Password Protocol”, *Proceedings of the 1998 Internet Society Network and Distributed Systems Symposium*, pp.97-111. Available from `ftp://srp.stanford.edu/pub/srp/srp.ps`
12. W. Diffie and M. Hellman, “New Directions in Cryptography”, *IEEE Transactions on Information Theory*, Vol. 22, 1976, pp. 644-654.
13. W. Diffie, P. van Oorschot and M. Wiener, “Authentication and Authenticated Key Exchanges”, *Designs, Codes and Cryptography*, 2, 1992, pp. 107-125.