

Public Key Protocols for Wireless Communications^{*}

Colin Boyd¹ and Dong-Gook Park^{1,2}

¹ Information Security Research Centre
Queensland University of Technology
Brisbane Q4001 Australia
`boyd@fit.qut.edu.au`

² Wireless Communications Research Laboratory
Korea Telecom
`park@isrc.qut.edu.au`

Abstract. Protocols for authentication and key establishment have special requirements in a wireless environment. In the next generation of wireless systems it is likely that public key based protocols will be employed. There are a number of important design decisions to be made in choosing an appropriate protocol. In this paper the design requirements are reviewed and some recently proposed public key protocols for wireless communications are examined. A new public key protocol is also proposed.

1 Introduction

Security requirements in the emerging third generation of wireless communications will be considerably more comprehensive than those in the current second generation digital systems. Cryptographic protocols and algorithms need to be used in order to satisfy these requirements. Probably the most critical security interface is that between the user and network, characterised by the radio connection. The security of this interface is paramount in preventing fraudulent access to network resources and in preserving user privacy.

Once a shared secret key is established between the two entities across an interface, digitally encoded speech and control information can be protected by symmetric encryption and integrity mechanisms. Although the design of appropriate bulk cryptographic algorithms is an area worthy of study, in this paper we will focus only on the protocols for establishing such symmetric session keys, including the problem of how to authenticate the entities involved.

Current second generation authentication protocols [11] rely on shared long-term keys between users and their home networks to establish session keys. This means that the home authentication centre needs to be on-line at the time of call setup (some solutions allow a set of session keys to be established together to alleviate this requirement). Consequently the home authentication centre needs

^{*} Research sponsored by Korea Telecom

to provide a high level of reliability and availability which is expensive to ensure. Since third generation systems are expected to be more widely distributed, this problem will increase. A solution is to use asymmetric (public key) cryptography which does not require an on-line server. This solution was not available to second generation designers due to the computational limitations of handheld devices at the time, but is now a possibility. Since it is likely that asymmetric cryptography will be required for other purposes, specifically for digital signatures in mobile electronic commerce, protocols using asymmetric cryptography seem to be a logical choice in the third generation.

The purpose of this paper is to examine various aspects of the use of public key based protocols for key establishment and authentication in third generation wireless systems. In the next section we consider the requirements for these protocols and compare the advantages of the two types of protocols known as key transport and key agreement protocols. We then review some recent proposals and point out some problems which have been missed by various authors. In particular, we present a fatal error in a recent proposal of Park [13] and question the design of protocols proposed in the ASPeCT project [10]. We then make a proposal of our own for a suitable protocol and finally compare this proposal with the ASPeCT.

2 Requirements for Mobile Protocols

The European ASPeCT project [6, 10] has been active in proposing public key based protocols for third generation wireless systems. In section 3.3 we will examine a protocol proposed from that project, but for now we look at the requirements they have identified. Horn and Preneel [6] proposed six goals for authentication protocols between mobile entities and the fixed network (which may include value added service providers (VASPs)). These requirements are as follows, with our own comments.

HP1 Mutual authentication of user and network. This is a widely accepted requirement, since one of the weaknesses in second generation systems is the lack of network side authentication. However, it should be mentioned that the definition of what entity authentication means is not universally accepted [5], and therefore this requirement may be redundant once goals concerning key establishment have been achieved.

HP2 Agreement between user and network on a secret session key with mutual implicit key authentication. Naturally both user and network must agree on what is the session key to be used. Mutual implicit key authentication means that both users must be sure which other entities may know the session key. This is a standard requirement for all key establishment protocols.

HP3 Mutual key confirmation. This is another property whose definition is not universally agreed upon. The general idea is to ensure that the other entity really does possess the same session key. However, different sources

disagree on whether the other entity needs to be one that has been identified [12, 8].

HP4 Mutual assurance of key freshness (mutual key control). It is usual to insist that the session key is a fresh one to prevent attacks based on replay of compromised keys from previous sessions. On the other hand mutual key control usually refers to a slightly different property, namely the inability of one party to force the choice of a specific session key value (a precise definition is difficult to decide upon). We would question the need for this latter requirement and will discuss it further in our comparison of key agreement and key transport protocols below.

HP5 Non-repudiation of origin for relevant user data. This ensures that users can provide a verifiable commitment to purchase items from a VASP. It will almost certainly be provided by use of digital signatures.

HP6 Confidentiality of relevant data. This requirement particularly applies to user identities and is special to the mobile environment, because users may require confidentiality of their location and movements. This requirement means that user certificates may not be sent in cleartext over the radio interface.

As well as these requirements, another factor in design of authentication protocols for the wireless environment is the limited power of the mobile handset. Even though we must assume that handsets in the third generation will be capable of public key cryptographic operations, the limited size, power and storage capabilities still apply. Horn and Preneel [6] therefore state as a design principle that as much possible, computational and storage requirements should be shifted from the mobile to the network side.

2.1 Key agreement or key transport?

Key establishment protocols can be broadly classified as either *key transport* protocols or *key agreement* protocols. Key transport occurs when one entity in the protocol chooses the session key unilaterally and sends it encrypted to the other entity (or entities). In contrast, both entities contribute to the session key in a key agreement protocol. The vast majority of key agreement protocols are based in some fashion on the protocol called Diffie-Hellman key exchange [3]. Although there are other kinds of key agreement protocol, we will limit the discussion in this section to those based on Diffie-Helman. The majority of recent authors proposing key establishment protocols for wireless communications have proposed such a key agreement scheme. Here we consider the arguments for and against such a decision.

In an application such as wireless communications, some cryptographic properties required in general network security protocols are more questionable. For example, the necessity of key freshness may be questioned if the authentication of the involved entities gives sufficient assurance and they are not expected to use an old (or compromised) session key. Usually key input data from one principal in a key agreement protocol has a nonce-like random property, and the

inclusion of this random number in the calculation of key value gives the principal the assurance that the resultant key value is fresh. However, key freshness seems to be intermingled with the freshness of the cryptographic message for entity authentication when people talk about the freshness property, because the freshness property of the key input data contributes to the authenticity of the cryptographic message.

Many proposed authentication protocols [1] have adopted key agreement instead of key transport. When using Diffie-Hellman style key agreement, however, this seems somewhat extravagant when compared to key transport, because the relationship between the user and the service provider is quite asymmetrical. An individual user of mobile services is literally a private entity while the other entity, the service provider, is a public entity (even though it may be owned by a private person). Hence, the necessity of confidentiality of communication cannot be taken the same for these two different kind of entities.

If a key transport scheme (or a key agreement scheme based on hashing) is adopted instead, significant savings of computational load can be acquired. This will be demonstrated with an example protocol later. Therefore we should consider what are the possible arguments in favour of key agreement protocols to justify their use. We particularly consider protocols based on Diffie-Hellman. Some of the possible arguments are as follows.

Key control Requirement HP4 states that neither party should be able to decide the value of the session key. We first observe that since either party can choose to give away the session key to a third party, the reason for this requirement cannot be one of trust. Therefore we conclude that it must be one of *key quality*. This arises because either the mobile or the network may be regarded as not sufficiently competent to choose the session key. On the other hand, many protocols, including key agreement protocols based on Diffie-Hellman, derive the session key as the output of a one-way hash function and we note that this technique can be used both in key transport and in key agreement not based on Diffie-Hellman. In summary, we believe that the requirements for key control can be satisfied by every type of protocol.

No encryption required Because of the complex export rules in different countries, it has often been proposed that protocols which do not include explicit encryption steps will be easier to export. Since Diffie-Hellman style key agreement requires only signatures, and not encryption, this may be an argument in its favour. We regard this argument as slightly suspect since it is clear that such protocols can be, and are intended to be, used for subsequent encryption using the session key.

Forward secrecy Diffie-Hellman style protocols have the attractive property that if the long-term private key of any user becomes compromised, this does not allow previous session keys to be found by an attacker. This property is not shared by key transport protocols, including key agreement protocols which rely on encryption of one of the user shares. Even though compromise of a long-term key is an unlikely scenario in the mobile environment, we

regard this as one useful advantage of using key agreement, even though it may be the only one.

3 Recent Protocol Proposals

In this section we will consider two recently proposed protocols for wireless communications. In all our descriptions we use the notation A for the mobile user and B for the network. Entities A and B possess private keys x_A and x_B , with public keys y_A and y_B respectively. All computations take place in the integers modulo a large prime p , unless stated otherwise, and there exists some value g which is known to have large order modulo p . The items r_A and r_B are random values chosen by A and B respectively. The notation $\{X\}_K$ denotes encryption with the symmetric key K of the message X .

3.1 Park's protocol

This protocol [13] is a modified version of an earlier protocol designed by Yacobi and Shmueli [16]. The public keys of A and B are $y_A = g^{-x_A}$ and $y_B = g^{-x_B}$ respectively. It is worthwhile to look at the original Yacobi and Shmueli protocol first. (It should be noted that in the original version computations took place using a composite modulus in place of the prime p we use here. This enabled a proof of security to be given for the protocol.) The protocol messages are as follows.

1. $B \rightarrow A : x_B + r_B$
2. $A \rightarrow B : x_A + r_A$

The session key is $K_{AB} = g^{r_A r_B}$. This is calculated by A as $K_{AB} = (g^{x_B + r_B} y_B)^{r_A}$ and by B as $K_{AB} = (g^{x_A + r_A} y_A)^{r_B}$. We will make a comment on the security of this protocol below, but now we turn to Park's protocol [13], whose only difference is a change in the messages sent as follows.

1. $B \rightarrow A : g^{x_B + r_B}$
2. $A \rightarrow B : x_A + r_A$

The session key is again $K_{AB} = g^{r_A r_B}$. This is calculated in the same way as for the Yacobi-Shmueli protocol but now A has already received $g^{x_B + r_B}$ so its computational effort is reduced. In the above description we have omitted two more fields in two messages exchanged. Those additional fields are for certificate exchanges and key confirmation, but they are not necessary for our discussion here.

The asymmetry between the messages is made with the limited computational power of the mobile terminal in mind. In the original Yacobi-Shmueli protocol both parties in the protocol have to perform two modular exponentiations to calculate K_{AB} . In Park's protocol the network side must perform three exponentiations while the mobile side A has to perform only one. In effect the

network side is performing an extra exponentiation to reduce the cost for the mobile. Unfortunately the saving has a fatal effect on the protocol security as we discuss below.

Martin and Mitchell [10] have found an attack on Park's protocol which allows an attacker who obtains a previously used key between A and B to masquerade as B . The first thing we would like to point out is that this attack is equally applicable to the original Yacobi-Shmueli protocol and we now describe Martin and Mitchell's attack in that context¹.

The condition for success of the attack is that the attacker C obtains a session key K_{AB} from an old protocol run, for which the protocol messages have been previously recorded. Thus C has the values $K_{AB} = g^{r_A r_B}$, $x_A + r_A$ and $g^{x_B + r_B}$. The attacker C now starts a new run of the protocol with the entity A in which the message sent by B is a replay of the message $x_B + r_B$ sent by B in the old protocol run. Entity A will send a new message $x_A + r'_A$, but by subtracting the recorded message $x_A + r_A$ from this, C obtains $r'_A - r_A$. It is now straightforward to check that the attacker is able to find the new session key $K'_{AB} = g^{r'_A r_B}$ by the following calculation.

$$K'_{AB} = (g^{x_B + r_B} y_B)^{r'_A - r_A} K_{AB}$$

Notice that in the original symmetrical Yacobi-Shmueli protocol this attack can be mounted in either direction so that C can masquerade as either A or B . However, in Park's protocol, since $x_B + r_B$ is not available to the attacker but only $g^{x_B + r_B}$, the attack can only be mounted by masquerading as B , as shown above.

The ability of an attacker to obtain old session key is a standard assumption in protocol analysis and has been used to break many well known protocols [2]. It may possibly be argued that such an attack is difficult to mount in the context of mobile communications in which a tamper resistant mobile handset is used. We now show that even without this assumption, the Park protocol is totally insecure.

3.2 New attack on Park's protocol

In the original Yacobi-Shmueli protocol $x_B + r_B$ is sent by B from which any entity can retrieve g^{r_B} . Nevertheless, no entity can form the message $x_B + r_B$ because x_B is known only to B . Now, as for the modified form $g^{x_B + r_B}$ of Park's protocol, any entity, say E , can form $y_B^{-1} g^{r_B} = g^{x_B + r_B}$ from y_B and g^{r_B} with any random value r_B chosen by E . Hence there is no signature effect in the modified form $g^{x_B + r_B}$.

Note that this change returns the authentication of B to A to that of basic Diffie-Hellman type protocol, which means there is no authentication of B to A at all! That is, any entity E can execute this protocol successfully with the

¹ This attack has previously been noted by Yacobi [17]. We thank an anonymous referee for drawing this to our attention.

entity A without the knowledge of the private key of B . This can be seen in the following attack procedure.

1. $E \rightarrow A : g^{x_B+r_B}$
2. $A \rightarrow E : x_A + r_A$

Here r_B is an arbitrarily random value chosen not by B but by E . Entity E can now calculate the session key in exactly the same way as B does in a normal protocol run, since it is not necessary to know the value of x_B to calculate $K_{AB} = (y_A g^{x_A+r_A})^{r_B} = g^{r_A r_B}$. The entity A (mobile terminal) believes the entity B (the network) is prepared to communicate with A and has established the same session key with that of A . The truth, however, is that the attacker has successfully derived the same session key with A . This attack does not require the attacker E to try any ‘play-in-the-middle’ between the mobile and the network. He can initiate this attack procedure at any time that he wants. Additional fields such as key confirmation in the actual Park protocol do not contribute any prevention to this attack because the session key itself is known to the attacker E .

3.3 The ASPeCT protocol

The following description [6, 10] is for the candidate protocol for UMTS authentication and key establishment based on asymmetric cryptography. It is intended as a user to value added service provider (VASP) security protocol as well as user to UMTS network. As well as the entities A representing the user, and B representing the network, a trusted certification authority, CA , is involved to provide certification of entities’ public keys. The public keys of A and B are $y_A = g^{x_A}$ and $y_B = g^{x_B}$ respectively. The other parameters are as follows.

h_1, h_2, h_3 Three hash functions.

$Sig_A\{X\}$ A ’s signature transformation on message X .

$ACert$ A ’s certificate which contains A ’s signature.

$BCert$ B ’s certificate which contains B ’s public key g^b .

chd Charging data

pay Payment data

T_B Time stamp issued by B .

1. $A \rightarrow B : g^{r_A}, CA$
2. $B \rightarrow A : r_B, h_2(K_{AB}, r_B, B), chd, T_B, BCert$
3. $A \rightarrow B : \{Sig_A(h_3(g^{r_A}, g^b, r_B, B), chd, TS, pay), ACert, pay\}_{K_{AB}}$

The session key is calculated by A as $K_{AB} = h_1(r_B, (y_B^{r_A}))$ and by B as $K_{AB} = h_1(r_B, (g^{r_A})^{x_B})$.

This protocol has evolved from several modifications and this description is the most current version available as of August, 1998. Several correctional additions to earlier version have been identified. The most prominent one is the inclusion of B ’s identity in the third message. This inclusion is prudent practice

in cryptographic protocol design, even though it is not easy to find a reasonable motive from relevant attacks. This is because the relationship of A and B is characterised as user to provider (UMTS network or VASP provider).

The inclusion of r_B in the message 2 has no direct contribution to the goal of the protocol. It is included to preclude time-memory trade-off attacks to find the value of K_{AB} . Note that the session key value is included in the hashed message field. Another interesting field is the identity of B in the message 2. Several papers in the literature [6] explain such inclusions as prevention of a sort of source substitution attack which may be possible only by unauthentic registration of B 's public key with the CA . This assumption can be avoided as long as authentic registration of each entity's public key is required for any application based on the public key infrastructure. Since there are well known protocols for achieving this [15], we regard it as reasonable to expect that this should be done at public key registration time. This is important if the public key will be used for other purposes such as for electronic commerce applications which are one of main targets of the ASPeCT protocol.

3.4 A Weakness of the ASPeCT Protocol

The most arguable weakness of this protocol is the delay in the identification of the entity A to the point of message 3. The reason for this appears to be to ensure user anonymity to provide requirement HP6. This delay is, in fact, not due to user anonymity itself but due to the type of authentication adopted by this protocol. Usually, the first message is the proper place for A 's identity. In this type of authentication (a typical example is the STS protocol [4]), however, any one of the three component passes cannot provide encryption without using an additional cryptographic operation with the session key because both party's cryptographic operation is executed using their private keys, not their public keys.

An attack scenario exists which may be considered quite reasonable for this protocol. The attacker's role in this procedure is simply to cut off the third message, disabling B from receiving the message. Of course, the attacker cannot derive the session key. However he does succeed in making the user A to sign his payment detail and, nevertheless, the VASP, B can never know that the user A has signed. Considering that this kind of payment transaction failure will inevitably cause users to be upset about the victim provider, this attack may be a good weapon of any ill-willed VASP to frustrate its rival VASPs. Of course this kind of cutting off in the middle is possible in any protocol. However, it should be noted that in this protocol, the entity B (provider) cannot know the identity of the user until it receives the third message.

This possibility of this attack may be considered to be due to the lack of a higher form of key confirmation, in which an entity can gain assurance that the peer entity not only knows the session key but also associates it with the correct communication partner. (This property is sometimes called *mutual belief in the key*.) This property could be added to the list of desirable requirements in any key agreement protocol. A useful comparison can be made with this attack and

that of Lowe [9] on the STS protocol of Diffie, van Oorschot and Wiener [4]. To paraphrase Lowe, we may say that the attack works because “A thought he was running the protocol with B , while B . . . may never have heard of A .” Of course, whether this attack has any consequences depends on what happens next. If A believes that he has been successfully authenticated by B he may expect that subsequent messages sent authenticated with K_{AB} will be properly attributed to B .

4 A New Proposal

Through consideration of the requirements of mobile entities, and in the light of the weaknesses of published protocols, we have devised a new authentication and key establishment protocol. We give two versions, the first of which provides key transport and the second key agreement (the key agreement protocol is not of Diffie-Hellman type.) These protocols are new but can be regarded as related to key transport mechanism 5 and key agreement mechanism 6, as presented in the ISO 11770-3 standard on key establishment protocols using public keys [8].

The notation to describe the protocols is the same as that used above, but in addition use is made of a field COUNT. The purpose of this field is orthogonal to the requirements HP1-HP6, but instead is used to detect cloning fraud in mobile handsets. Use of this field is detailed elsewhere [14] and forms part of an integrated approach to wireless communications security. The notation $Enc_B\{X\}$ denotes encryption of the field X using the public key of B .

1. $A \rightarrow B : Enc_B\{A, K_{AB}, COUNT\}$
2. $B \rightarrow A : \{COUNT, r_B\}_{K_{AB}}$
3. $A \rightarrow B : Sig_A\{B, h(COUNT, K_{AB}, r_B)\}$

In the second example both A and B contribute to the session key, calculated as $K_{AB} = h(r_A, r_B)$ for a suitable hash function h .

1. $A \rightarrow B : Enc_B\{A, r_A, COUNT\}$
2. $B \rightarrow A : r_B, \{COUNT, r_A\}_{K_{AB}}$
3. $A \rightarrow B : Sig_A\{B, h(COUNT, r_B, K_{AB})\}$

It is assumed here as usual that A and B have access to the public keys of each other. As for A 's way to get the public key of the network, it may receive the key value from the system broadcast channel of the network. The network, B can get the public key of the user from the certificate data of A if it is included inside the encrypted information in the first message.

The second message provides key confirmation to A . The third message provides key confirmation (and freshness) for B . It also provides the non-repudiation feature for electronic commerce application. Note that the first message can be pre-calculated off-line by the mobile. The identity of the user A is contained in the first message which is encrypted using B 's public key for user anonymity. Hence, the weakness identified in the ASPeCT protocol does not apply to this

protocol. Furthermore, the inclusion of the identity A in the first message is a prudent practice to prevent any play-in-the-middle attacks.

Martin and Mitchell [10] have pointed out that certain signatures may leak information on the identity of the sender. This occurs if the verification procedure can reveal redundant structure in the signature and does not require the signed message to be known. For our protocol we propose to use an ElGamal or Schnorr [15] type signature. This does not appear to allow such an attack and also has the advantage that almost all of the computation can be done off-line by the mobile. Even more appropriate may be elliptic curve variants of ElGamal signatures, which are currently in the process of standardisation [7]. These promise to allow more efficient computation and storage than those based on prime fields, and so are particularly useful for hand-held devices.

In comparison with the ASPeCT protocol there are many features in common. In both protocols, although the mobile side is required to perform public key operations, most of these can be performed off-line. It is reasonable to assume that the identity of B is known to A beforehand. This means that in the ASPeCT protocols the value $g^{r_A x_B}$, used in the calculation of K_{AB} , may be performed off-line by A . The same is true of the first message in the proposed protocols. This leaves only hashing, symmetric encryption operations and generation of the signature to be done on-line. Of these, a suitable choice of algorithm allows most of the effort in signature generation to be done off-line too.

The only disadvantage that we have found with our protocols is the lack of forward secrecy, but this is a feature of the ASPeCT protocol too, since compromise of the network's long term key, x_B , allows the session key to be recovered. Therefore we consider that our proposal has all the advantages of the ASPeCT protocol, while avoiding the potential weakness discussed in section 3.4.

5 Conclusion

We have considered the requirements for key establishment and authentication protocols for wireless communications. We have considered some recently proposed protocols and found some weaknesses. Moreover, we have argued that key agreement using Diffie-Hellman style protocols are both unnecessary and computationally expensive. Consequently we have proposed new protocols which avoid this criticism.

At present we have not provided any formal analysis or security proofs for our proposed protocols. We invite criticisms from interested readers and aim to provide stronger arguments for security in the near future.

References

1. C. Boyd and A. Mathuria, "Key Establishment Protocols for Secure Mobile Communications: A Selective Survey", *Information Security and Privacy*, LNCS 1438, Springer-Verlag, 1998, pp.344-355.

2. D. Denning and G. Sacco, "Timestamps in Key Distribution Protocols", *Communications of the ACM*, 34, 1981 pp.533-536.
3. W. Diffie and M. Hellman, "New Directions in Cryptography", *IEEE Transaction on Information Theory*, 22, 1976, pp.644-654.
4. W. Diffie, P. van Oorshot and M. Wiener, "Authentication and Authenticated Key Exchanges", *Designs, Codes and Cryptography*, 2, 1992, pp.107-125.
5. D. Gollman, "What do we Mean by Entity Authentication", *IEEE Symposium on Security and Privacy*, pp.46-54, 1996.
6. G. Horn and B. Preneel, "Authentication and Payment in Future Mobile Systems", Proceedings of ESORICS'98, Springer-Verlag, 1998.
7. IEEE P1363 Standard Specifications for Public Key Cryptography, Draft Version, September 1998.
8. ISO-IEC, *DIS 11770-3 Key Management - Part 3: Mechanisms using Asymmetric Techniques*, 1996.
9. G. Lowe, "Some New Attacks upon Security Protocols", *9th IEEE Computer Security Foundations Workshop*, IEEE Computer Society Press, 1996, pp.162-169.
10. Keith Martin and Chris Mitchell, "Evaluation of Authentication Protocols for Mobile Environment Value-added Services", Draft, 1998. Available on-line as <http://isg.rhnc.ac.uk/cjm/EOAPFM.ZIP>.
11. A. Mehrota, *GSM System Engineering*, Artech House, 1997.
12. A. Menezes, P. van Oorshot, S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1997.
13. C.-S. Park, "On Certificate-Based Security Protocols for Wireless Mobile Communication Systems", *IEEE Network*, September/October 1997, pp.50-55.
14. D.-G. Park and M. Looi, "A Fraud Detection Method and Its Application to Third Generation Wireless Systems", Globecom'98.
15. C. Schnorr, "Efficient Signature Generation by Smart Cards", *Journal of Cryptology*, 4, 1991, pp.161-174.
16. Y. Yacobi and Z. Shmueli, "On Key Distribution Systems", *Advances in Cryptology - Crypto'89*, Springer-Verlag, pp.344-355.
17. Y. Yacobi, "A Key Distribution "Paradox"", *Advances in Cryptology - CRYPTO'90*, Springer-Verlag 1991, pp.268-273.