The Fifth IFIP–TC6 International Conference on Mobile and Wireless Communications Networks, pages 59–62.
(october 27 - 29, 2003, Singapore).
Revision – September 28th, 2004.

Mutual Authentication and Group Key Agreement for Low-Power Mobile Devices

Emmanuel Bresson[1]*, Olivier Chevassut[2], Abdelilah Essiari[2], and David Pointcheval[3]

[1] Cryptology department – CELAR, 35170 Bruz, France
Emmanuel.Bresson@polytechnique.org
[2] Lawrence Berkeley National Lab. – Berkeley, CA, USA †
{OChevassut,AEssiari}@lbl.gov
[3] École normale supérieure – Paris, France
David.Pointcheval@ens.fr

Abstract. Wireless networking has the power to fit the Internet with wings, however, it will not take off until the security technological hurdles have been overcome. In this paper we propose a *very efficient* and *provably-secure* group key agreement well suited for unbalanced networks consisting of devices with strict power consumption restrictions and wireless gateways with less stringent restrictions. Our method meets practicability, simplicity, and strong notions of security.

1 Introduction

Wireless technology has become more pervasive as Internet electronic and commerce transactions on mobile devices have gained in popularity. Institutions and industries are hankering for the power and flexibility of wireless networks, but many are postponing rollouts in strategic areas until they are convinced that their systems are not at risk. The security technologies currently deployed to protect the Internet against attacks are not fully applicable to the wireless Internet since the traditional Internet does not typically place constraints on available power consumption or bandwidth. The nodes in a wireless network are usually mobile and have computation bandwidth capabilities that place severe restrictions when designing cryptographic protocols. Storage limitation on the other hand is becoming less of an issue as many add-on memory cards are now widely available.

In the present paper we have focused on computing applications involving clusters of mobile devices [3, 9, 11]. The Wired Equivalent Privacy (WEP) protocol, which is part of the IEEE 802.11 standard, specifies how to protect the traffic between mobile devices and access points (i.e. gateways) using *pre-established* session keys without specifying how the keys are established. This lack of proper key-establishment scheme has opened the door to various attacks [4]. Our contribution in this paper is a *provably-secure* authenticated group key-exchange scheme based on public-key cryptography that can complement the WEP protocol. Schemes based on symmetric cryptography have obvious performance advantages over public-key cryptography, but suffer from a complex key management; they require trust in the entire network as a device moves from one domain to another. Other schemes based on public-key technology trade less computation for more communication rounds, but are still too costly to be practical for wireless networks that involve low-power computing devices [1, 2, 5, 6].

Our key-exchange scheme allows a cluster of mobiles and one wireless gateway to dynamically agree on a session key. It shifts the computational burden to the gateway and provides mobile devices with the ability to perform most of the public-key cryptographic computations off-line.

* Work done while at ENS, Paris.
† The authors are supported by the Director, Office of Science, Office of Advanced Scientific Computing Research, Mathematical Information and Computing Sciences Division, of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098. This document is report LBNL-52542. Disclaimer available at `http://www-library.lbl.gov/disclaimer`.

This scheme can furthermore be combined with a group Diffie-Hellman key exchange scheme [5] to cover wireless environments involving more than one gateway [8]. A mobile device would only perform one public-cryptographic computation as it moves from one wireless domain to the other.

The paper is organized as follows. In Section 3 we present a scheme that achieves "implicit" authentication and in Section 5 we discuss enhancements to achieve "explicit" authentication.

2 Modeling Unbalanced Wireless Networks

Wireless Nodes. The wireless-communication system we model is a set $\mathcal{C}$, of N wireless-capable mobile devices (also called *clients*), and a wireless gateway (also called *server* or base station). We assume the clients and the server do not deviate from the algorithm they are expected to execute. We consider a nonempty subset of $\mathcal{C}$ which we call the *wireless client group* $\mathcal{G}_c$ that consists of the clients communicating with the server. The server S has the special role of adding and removing clients from the group $\mathcal{G}_c$. (In practice, this server covers an entire wireless region called a cell or domain and, thus, it never leaves, hence its special role.) Each mobile U, as well as the base station, also holds a long-lived key LL_U which is a pair of matching public/private keys.

Abstract Interface. We define the basic structure of a group key agreement scheme for unbalanced wireless networks. The scheme GKE consists of four algorithms:

– The *key generation algorithm* $\mathsf{GKE.KGen}(1^\ell)$ is a probabilistic algorithm which on input a security parameter 1^ℓ, provides each client U_i in $\mathcal{C}$ and the base station with a long-lived key.
– The *setup algorithm* $\mathsf{GKE.Setup}(\mathcal{J})$ is an interactive protocol which on input of a set of clients $\mathcal{J}$, sets the wireless client group to be $\mathcal{G}_c = \mathcal{J}$ and provides each client U in $\mathcal{G}_c$ with a secret value sk shared with the base station.
– The *join algorithm* $\mathsf{GKE.Join}(\mathcal{J})$ is an interactive protocol which on input of a set of clients $\mathcal{J}$, updates the wireless client group $\mathcal{G}_c$ to be $\mathcal{G}_c \cup \mathcal{J}$, and provides each client U in $\mathcal{G}_c$ with a new shared secret value sk.
– The *remove algorithm* $\mathsf{GKE.Remove}(\mathcal{J})$ is an interactive protocol which on input of a subset $\mathcal{J}$ of the wireless client group $\mathcal{G}_c$, updates the latter to be $\mathcal{G}_c \backslash \mathcal{J}$, and provides each client U in $\mathcal{G}_c$ with a new shared secret value sk.

3 Key Agreement

This section provides a method accommodating group key agreement to mobiles lacking the computational resources to perform multiple on-line computation in a finite cyclic group (such as modular exponentiation), but with enough computational resources to perform symmetric-cryptographic operations.

The session-key space SK associated to this method is $\{0,1\}^\ell$ equipped with a uniform distribution, where ℓ is a security parameter. Arithmetic is in a finite cyclic group $\mathbb{G} = \langle g \rangle$ of ℓ-bit prime order q. Both g and q are publicly known. There are also three hash functions $\mathcal{H} : \{0,1\}^\star \rightarrow \{0,1\}^\ell$, $\mathcal{H}_0 : \{0,1\}^\star \rightarrow \{0,1\}^{\ell_0}$, where ℓ_0 needs not be equal to ℓ, and $\mathcal{H}_1 : \{0,1\}^{\ell_1} \times \mathbb{G} \rightarrow \{0,1\}^{\ell_0}$, where ℓ_1 is the maximal bit-length of a counter c used to prevent replay attacks.

We consider a signature scheme $\mathsf{SIGN} = (\mathsf{SIGN.KGen}, \mathsf{SIGN.Sig}, \mathsf{SIGN.Ver})$. Each client U_i holds a pair of signing private/public key (SK_i, PK_i) which are the output of the key generation

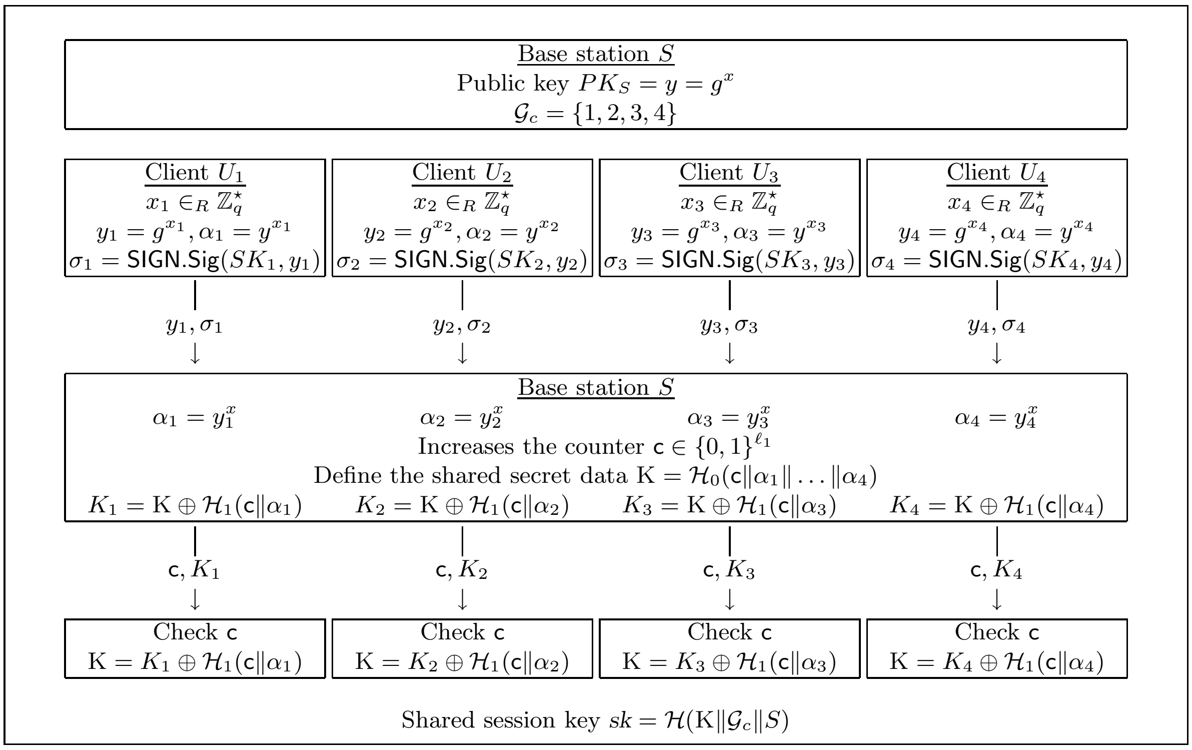
Base station S
Public key $PK_S = y = g^x$
$\mathcal{G}_c = \{1, 2, 3, 4\}$
Client U_1
$x_1 \in_R \mathbb{Z}_q^\star$
$y_1 = g^{x_1}, \alpha_1 = y^{x_1}$
$\sigma_1 = \mathsf{SIGN.Sig}(SK_1, y_1)$
Client U_2
$x_2 \in_R \mathbb{Z}_q^\star$
$y_2 = g^{x_2}, \alpha_2 = y^{x_2}$
$\sigma_2 = \mathsf{SIGN.Sig}(SK_2, y_2)$
Client U_3
$x_3 \in_R \mathbb{Z}_q^\star$
$y_3 = g^{x_3}, \alpha_3 = y^{x_3}$
$\sigma_3 = \mathsf{SIGN.Sig}(SK_3, y_3)$
Client U_4
$x_4 \in_R \mathbb{Z}_q^\star$
$y_4 = g^{x_4}, \alpha_4 = y^{x_4}$
$\sigma_4 = \mathsf{SIGN.Sig}(SK_4, y_4)$
y_1, σ_1
y_2, σ_2
y_3, σ_3
y_4, σ_4
Base station S
$\alpha_1 = y_1^x$
$\alpha_2 = y_2^x$
$\alpha_3 = y_3^x$
$\alpha_4 = y_4^x$
Increases the counter $\mathsf{c} \in \{0,1\}^{\ell_1}$
Define the shared secret data $\mathrm{K} = \mathcal{H}_0(\mathsf{c}\|\alpha_1\| \ldots \|\alpha_4)$
$K_1 = \mathrm{K} \oplus \mathcal{H}_1(\mathsf{c}\|\alpha_1)$
$K_2 = \mathrm{K} \oplus \mathcal{H}_1(\mathsf{c}\|\alpha_2)$
$K_3 = \mathrm{K} \oplus \mathcal{H}_1(\mathsf{c}\|\alpha_3)$
$K_4 = \mathrm{K} \oplus \mathcal{H}_1(\mathsf{c}\|\alpha_4)$
c, K_1
c, K_2
c, K_3
c, K_4
Check c
$\mathrm{K} = K_1 \oplus \mathcal{H}_1(\mathsf{c}\|\alpha_1)$
Check c
$\mathrm{K} = K_2 \oplus \mathcal{H}_1(\mathsf{c}\|\alpha_2)$
Check c
$\mathrm{K} = K_3 \oplus \mathcal{H}_1(\mathsf{c}\|\alpha_3)$
Check c
$\mathrm{K} = K_4 \oplus \mathcal{H}_1(\mathsf{c}\|\alpha_4)$
Shared session key $sk = \mathcal{H}(\mathrm{K}\|\mathcal{G}_c\|S)$

Fig. 1. An execution of the Setup algorithm with the five devices U_1, U_2, U_3, U_4 and S.

signature scheme algorithm SIGN.KGen. One would probably argue that when dealing with low-power computing mobiles, special low-cost [12] or on-line/off-line [13] signature schemes are required. However, the messages to be signed are in our setting known at pre-computation time and, thus, a mobile does not have to compute anything on-line to send its contribution.

3.1 Algorithms

Note 1. The recent paper [10] pointed out a mistake in our counter management: the counter must always be increasing. We thus initialise it just once.

Key Generation. The algorithm GKE.KGen, on input the set of clients $\mathcal{C}$ and a security parameter ℓ, performs the following steps:

1. Run $\mathsf{SIGN.KGen}(1^\ell)$ for each client U_i in $\mathcal{C}$ to provide each client with a pair (SK_i, PK_i) of signing/verifying keys;
2. Choose $x \in_R \mathbb{Z}_q^\star$ and set the Server's private/public keys to be: $(SK_S, PK_S) = (x, g^x)$. One denotes $y = g^x$;
3. All the parties initialize their own counters $\mathsf{c} = 0$, as bit-strings of length ℓ_1.

Setup. The algorithm GKE.Setup, on input a set of client-devices $\mathcal{J}$, performs the following steps (see also Figure 1):

1. Set the wireless client group $\mathcal{G}_c$ to be the input set $\mathcal{J}$.
2. Each client $U_i \in \mathcal{G}_c$ chooses at random a value $x_i \in \mathbb{Z}_q$ and *precomputes* $y_i = g^{x_i}$, $\alpha_i = PK_S^{x_i} = y^{x_i}$ as well as a signature σ_i of y_i, under the private key SK_i.
3. Each client U_i sends (y_i, σ_i) to S.
4. For each $i \in \mathcal{G}_c$, the server S checks the signature σ_i using PK_i, and if they are all correct, computes the values $\alpha_i = y_i^x$.

5. The server S increases the counter c, and computes the shared secret value:

$$\mathrm{K} = \mathcal{H}_0(\mathsf{c}\|\{\alpha_i\}_{i\in\mathcal{G}_c})$$

   and sends to each client U_i the values c and $K_i = \mathrm{K} \oplus \mathcal{H}_1(\mathsf{c}\|\alpha_i)$.
6. Each client U_i (and S) first checks that the new counter is greater than the old one, and recovers the shared secret value K and the session key *sk* as described below, and accepts:

$$\mathrm{K} = K_i \oplus \mathcal{H}_1(\mathsf{c}\|\alpha_i) \quad \text{and} \quad sk = \mathcal{H}(\mathrm{K}\|\mathcal{G}_c\|S).$$

Remove. The algorithm $\mathsf{GKE.Remove}$, on input the set $\mathcal{J}$ of disappearing client-devices, performs the following steps:

1. Update the wireless client group $\mathcal{G}_c = \mathcal{G}_c \backslash \mathcal{J}$.
2. The server S operates as in the Setup phase. It increases the counter c and computes the shared secret value $\mathrm{K} = \mathcal{H}_0(\mathsf{c}\|\{\alpha_i\}_{i\in\mathcal{G}_c})$.
3. Then it sends to each client $U_i \in \mathcal{G}_c$ the values c and $K_i = \mathrm{K} \oplus \mathcal{H}_1(\mathsf{c}\|\alpha_i)$.
4. Each client $U_i \in \mathcal{G}_c$ already holds the value $\alpha_i = g^{xx_i}$, and the old counter value. So it first checks that the new counter is greater than the old one, and simply recovers the secret shared value K and the session key *sk* as described below, and accepts:

$$\mathrm{K} = K_i \oplus \mathcal{H}_1(\mathsf{c}\|\alpha_i) \quad \text{and} \quad sk = \mathcal{H}(\mathrm{K}\|\mathcal{G}_c\|S).$$

Join. The algorithm $\mathsf{GKE.Join}$, on input the set of appearing clients $\mathcal{J}$, performs similar steps to the $\mathsf{GKE.Setup}$.

A complete description of the protocols can be found in the full version of this paper [7].

3.2 Efficiency

The protocol presented in this paper is very efficient, since almost everything can be precomputed off-line for the clients, while achieving a strong level of security. The amount of memory available on the clients may provide a trade-off:

– by storing many distinct triples $(y_i, \sigma_i, \alpha_i)$ one increases the security level, but one hashing and one XOR have to be performed on-line;
– by storing many $\mathcal{H}_1(\mathsf{c}\|\alpha_i)$, for each $(y_i, \sigma_i, \alpha_i)$, for several values of the counter, one increases efficiency, since only one XOR has to be performed on-line.

4 Security Analysis

4.1 Assumptions

The security analysis of our protocol is based on the security of the underlying signature scheme that is used to authenticate messages, and on the computational Diffie-Hellman problem.

The security of the signature scheme is measured as the probability that an adversary, having access to an oracle that can sign arbitrary messages (chosen-message attack, or cma), produces a new, valid *(message, signature)* pair. Such a probability is denoted $\mathsf{Succ}^{\mathsf{cma}}_{\mathsf{SIGN}}(t, Q)$ where t is the adversary's working time and Q is the number of signing oracle queries.

The intractability of the Computational Diffie-Hellman (CDH) problem in the group $\mathbb{G}$ is measured as the probability that an attacker, on input two random elements g^a and g^b in $\mathbb{G}$, can compute g^{ab}. Such a probability is denoted $\mathsf{Succ}^{\mathsf{cdh}}_{\mathbb{G}}(t)$, where t is the adversary's working time.

4.2 Our result

The security of our protocol is measured as the probability that an adversary can get some (partial) information on the key. This probability is denoted $\mathsf{Adv}_P^{\mathsf{ake}}$ and depends on the number of messages sent on the network.

Security Theorem. Let $\mathcal{A}$ be an adversary against the Authenticated Key Exchange (AKE) security of our protocol P, making at most q_s active requests, and asking at most q_H queries to the hash oracles ($\mathcal{H}_0$ and $\mathcal{H}_1$). Let N denote the total number of low-power devices. Then we have:

$$\mathsf{Adv}_P^{\mathsf{ake}}(\mathcal{A}) \leq 2N \cdot \mathsf{Succ}_{\mathsf{SIGN}}^{\mathsf{cma}}(t, q_s) + 2q_s q_H \cdot \mathsf{Succ}_{\mathbb{G}}^{\mathsf{cdh}}(t).$$

The above theorem shows that the security of our protocol is based on the intractability of the well-studied computational Diffie-Hellman problem (CDH) and on the security of the signature scheme (CMA) to prevent existential forgeries under adaptive chosen message attacks.

Sketch of Proof. Unless the adversary breaks the signature scheme for one among the N clients (so, $N \cdot \mathsf{Succ}_{\mathsf{SIGN}}^{\mathsf{cma}}(\cdot)$), any valid message is output by a regular node, and can be simulated. We then note that the simulated flows can be derived from a CDH problem (g^a, g^b), while the hash values that would involve g^{ab} are answered randomly. The adversary breaks the scheme if it is able to discover which hash value corresponds to a given flow (so, $q_s q_H \cdot \mathsf{Succ}_{\mathbb{G}}^{\mathsf{cdh}}(\cdot)$). The full proof can be found in the full version [7].

Note 2. Granted the increasing property of the counter, the replay attack proposed in [10] is now prevented.

5 Mutual Authentication and (Partial) Forward Secrecy

Mutual authentication ensures each player that all other parties did actually compute the same key. Our protocol can be modified to achieve this goal. The natural modification, wherein each player sends to all the other ones an “authenticator”, requires that each low-power device computes N hashings and sends one flow to the server S. This computational overhead is tolerable only if N does not get too large, but for larger values of N this overhead can also be kept to a minimum by performing mutual authentication through the server. Each client authenticates to the server which then in turn authenticates to each client only after all clients have been authenticated. This approach has the attractive advantage of being not only provably secure, in the random-oracle model, but to also add little overhead to the original protocol.

About forward-secrecy, it is clear that as soon as the long-term key x of the server is leaked, all the session keys can be recovered, since all the α_i can easily be computed from the y_i and x. Therefore, no forward-secrecy exists when the server long-term key is revealed. However, the long-term keys of the low-power devices (the signing keys) are used for implicit authentication only, and not for hiding the session key. Therefore, the leakage of clients' long-term keys do not reveal anything about previous session keys. Furthermore, strong (partial) forward-secrecy (where any internal data is revealed in case of corruption, *i.e.* the signing key, but also the x_i, y_i and α_i) is also achieved if the x_i's and α_i's are erased as soon as they are not useful (the client has left from the group). As a consequence, no information about previous session keys can be found in the memory of the low-power devices. We claim these considerations make sense since the server can be reasonably seen as more reliable that the mobile devices.

6 Conclusion

In this paper we presented an efficient key agreement protocol for heterogeneous wireless networks. Our protocol allows a set of heterogeneous mobiles devices to form a secure group and to handle the continuous disappearing and reappearing of mobiles due to communication failures. Our protocol has been proved secure in the random oracle model under the computational Diffie-Hellman assumption.

References

1. N. Asokan and P. Ginzboorg. Key Agreement in Ad-hoc Networks, February 2000. Expanded version of a talk given at the Nordsec '99 workshop.
2. G. Ateniese, M. Steiner, and G. Tsudik. New Multiparty Authentication Services and Key Agreement Protocols. *IEEE J. of Selected Areas in Communications*, Apr. 2000.
3. K. Berket, P. M. Melliar-Smith, and L. E. Moser. The InterGroup Protocols: Scalable Group Communication for the Internet. *3rd Global Internet Mini-Conference*, Nov. 1998.
4. N. Borisov, I. Goldberg, and D. Wagner. Intercepting Mobile Communications: The Insecurity of 802.11. In *ACM MobiCom'01*, 2001.
5. E. Bresson, O. Chevassut, and D. Pointcheval. Dynamic Group Diffie-Hellman Key Exchange under Standard Assumptions. In *Eurocrypt '02*, LNCS 2332, pp. 321–336. Springer, May 2002.
6. E. Bresson, O. Chevassut, and D. Pointcheval. Group Diffie-Hellman Key Exchange Secure Against Dictionary Attacks. In *Asiacrypt '02*, LNCS 2501, pp. 497-514. Springer, Dec 2002.
7. E. Bresson, O. Chevassut, A. Essiari, and D. Pointcheval. Mutual Authentication and Group Key Agreement for Low-Power Mobile Devices. In *5th IEEE MWCN*. IEEE, Oct 2003.
8. G. D. Crescenzo and O. Kornievskaia. Efficient Kerberized Multicast in a Practical Distributed Setting. In *Information Security Conference (ISC'01)*. Oct 2001
9. A. Harbitter and D. A. Menace. The Performance of Public Key-Enabled Kerberos Authentication in Mobile Computing Applications. In *ACM CCS'01*, pp. 78–85.
10. J. Nam, S. Kim, and D. Won. Attacks on Bresson-Chevassut-Essiari-Pointcheval's Group Key Agreement Scheme. In *Cryptology ePrint Archive*, Report 2004/251.
11. T. Phan, L. Huang, and C. Dulan. Challenge: Integrating Mobile Wireless Devices Into the Computational Grid. In *MobiCom '02*, pp. 271–278, 2002.
12. G. Poupard and J. Stern. Security Analysis of a Practical "on the fly" Authentification and Signature Generation. In *Eurocrypt '98*, LNCS 1403, pp. 422–436. Springer, May 1998.
13. A. Shamir and Y. Tauman. Improved Online/Offline Signature Schemes. In *Crypto '01*, LNCS 2139, pp. 355–367. Springer, Aug 2001.