A Privacy-preserving ID-based Group Key Agreement Scheme applied in VPAN

Yoni De Mulder, Karel Wouters, and Bart Preneel

Katholieke Universiteit Leuven
Dept. Electrical Engineering-ESAT/SCD/IBBT-COSIC
Kasteelpark Arenberg 10, 3001 Heverlee, Belgium
{yoni.demulder,karel.wouters,bart.preneel}@esat.kuleuven.be

Abstract. In 2008, Wan et al. presented an anonymous ID-based group key agreement scheme for wireless networks, for which they claim that it ensures anonymity and unlinkability of the group members, as well as forward and backward secrecy of the group session key. In this paper, we show that forward and backward secrecy do not hold for the protocol. We propose a correction that introduces a shielding factor that protects each member's input to the group key. we also introduce a new feature that assures the correctness of the key as computed by all group members. This results in an increased computation cost, due to extra public key operations, and a similar communication cost. We also show in which practical setting the protocol can be deployed.

Keywords. Privacy, Group key agreement, ID-based cryptography

1 Introduction

In this paper, we propose an improved version of the anonymous ID-based group key agreement scheme for wireless networks by Wan *et al.* [1], and present a scenario in which such a scheme can be used. The original scheme claims to provide group member anonymity from outside eavesdroppers, and forward and backward group key secrecy from leaving resp. joining group members. We show that these claims are incorrect and propose an improved protocol. In Sect. 2, we introduce ID-based cryptography, the original protocol by Wan *et al.* and its vulnerabilities. The main problem with the protocol is that most of the shares of the previously agreed group key remain unaltered in the computation of the new group key, such that joining/leaving members can reconstruct the old/new group key. The improvement, presented in Sect. 3 involves the introduction of a session ID to protect the previously established group key shares. In Sect. 4 we analyse the performance loss, and indicate why the improvements fix the original protocol. Finally, we show how the protocol can be used in the practical setting of a Virtual Private Ad Hoc Network (VPAN), and we conclude with ideas for future work.

The setting in which we operate is as follows: a dynamic set of devices wants to establish a shared secret group key in a privacy-preserving way. 'Dynamic' means that devices can join or leave, albeit at a slow pace (a couple of devices per hour). Encryption with an authenticated group key will ensure confidentiality, but we also require the following:

- **Anonymity:** a group member remains anonymous for an outsider;
- **Unlinkability:** an outsider should be unable to link a group member across sessions with different group keys;
- **Backward and forward secrecy:** only the members that were involved in the group key generation should be able to construct the group key. Being part of the group in the past (forward secrecy) or the future (backward secrecy) leaks no information on the 'current' group key.
- **Perfect forward secrecy:** in case of master key compromise, all previous group keys should remain secret.

2 ID-based Protocol by Wan *et al.*

The group key agreement scheme of Wan *et al.* in [1] is based on *ID-based public-key cryptography* in which the public key of a user is derived from its identity. Identity Based Encryption (IBE) was proposed by Shamir [2] in 1984, and one of the first practical IBE schemes [3, 4] is based on bilinear maps. In IBE, a trusted server S, called a Private Key Generator (PKG), provides private keys for all participating group members using a randomly chosen master secret s and the identity of each user, after checking his ID. The focus on this paper is on how to fix the scheme of Wan *et al.*, but many similar schemes have been proposed; one example of similar work can be found in [5], which also contains some references to other schemes, and a survey [6] of key agreement protocols. The mathematical setup for the protocol in [1] can be summarised as follows: the PKG selects two cyclic groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of order q for some large prime q, and a bilinear mapping $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2 : (P_1, P_2) \mapsto Q$. The PKG determines the generator $P \in \mathbb{G}_1$, a master secret $s \in_R \mathbb{Z}_q \setminus \{0\}$, and a public value $P_{\text{pub}} = sP$.
The public parameters $\langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, H_1, P, P_{\text{pub}} \rangle$ are distributed to all users in the system, where H_1 is a hash function, $H_1 : \{0,1\}^* \to \mathbb{G}_1$, used to embed identity values in $\mathbb{G}_1$: For a user with identity U_i, the PKG generates the private key $\text{PrK}_i = sH_1(U_i)$, for which the corresponding public key is the user's identity $\text{PuK}_i = U_i$. In the remainder of the text, $u_i = H_1(U_i)$.
The building blocks of the anonymous ID-based group key agreement protocol presented by Wan *et al.*, which is based on [7, 8], are: group initialisation, the join protocol and the leave protocol.

Initialisation Protocol
The initialisation protocol is executed when an initiator U_1 wants to have a private group session with a set of users $\{U_2, \ldots, U_n\}$.
Round 1: $U_1 \to U_i : [\text{E}_{\text{PuK}_i}(\mathcal{L}, \text{Sig}_{\text{PrK}_1}(\mathcal{L})),\ r_1 P]$, where $r_1 \in_R \mathbb{Z}_q$ and $\mathcal{L} = U_1 \| \ldots \| U_n \| Nym_1 \| \ldots \| Nym_n$, the concatenation of the identities U_i and the related pseudonyms Nym_i. $\mathcal{L}$ is signed with U_1's private key, and encrypted with the recipient's (U_i) public key.

Round 2: $U_i \rightarrow U_{i-1}, U_{i+1} : [Nym_i, r_iP]$: The users $U_i, (i \neq 1)$ decrypt the message from Round 1, retrieve their pseudonym Nym_i, which is sent, together with $r_iP,\ r_i \in_R \mathbb{Z}_q$ to the users U_{i-1} and U_{i+1}:
Round 3: $U_i \rightarrow * : [Nym_i, X_i = \frac{k_i}{k_{i-1}}]$: Each U_i broadcasts his Nym_i with a key share $X_i = k_i/k_{i-1}$, depending on his private key $\mathrm{PrK}_i = su_i$, the random number r_i and the points $r_{i-1}P$ and $r_{i+1}P$ from Round 2:
$k_i = h(\hat{e}(u_{i+1}, \mathrm{PrK}_i) \| r_i r_{i+1} P),\ k_{i-1} = h(\hat{e}(u_{i-1}, \mathrm{PrK}_i) \| r_i r_{i-1} P)$.[1] The bilinear property of the mapping $\hat{e}$ ensures the consistency of the subkeys k_i: $\hat{e}(u_{i+1}, \mathrm{PrK}_i) = \hat{e}(u_{i+1}, su_i) = \hat{e}(u_i, su_{i+1}) = \hat{e}(u_i, \mathrm{PrK}_{i+1})$.
Group key K generation: Each user U_i receives all $X_j, (j \neq i)$, and computes[2] the 'subkeys' k_{i+1},k_{i+2},...,k_{i+n-1}, from his own subkey k_i:
$k_{i+1} = k_i X_{i+1}, k_{i+2} = k_{i+1} X_{i+2}, \ldots\ k_{i+n-1} = k_{i-1} = k_{i+n-2} X_{i+n-1} = k_{i-2} X_{i-1}$.
Then U_i verifies $k_{i+n-1} X_{i+n} = k_i$, and forms the group key $K = H(k_1 \| k_2 \| \ldots \| k_n)$.[3]
Finally, each user $U_i, (i \neq 1)$ sends $H(K \| U_1 \| U_2 \| \ldots \| U_n)$ to the initiator U_1, who checks the consistency of the group key K.

Join Protocol

When the join protocol is executed to add a new user U_{n+1} to the group, the group key is updated to ensure backward secrecy.
Round 1: U_1 generates Nym_{n+1} for U_{n+1} and initiates the protocol:

$$U_1 \rightarrow U_n : \mathrm{E}_{\mathrm{PuK}_n}(\mathcal{L}_1 \| \mathrm{Sig}_{\mathrm{PrK}_1}(\mathcal{L}_1)), \quad \mathcal{L}_1 = U_{n+1} \| Nym_{n+1} \tag{1}$$

$$U_1 \rightarrow U_{n+1} : \mathrm{E}_{\mathrm{PuK}_{n+1}}(\mathcal{L}_2 \| \mathrm{Sig}_{\mathrm{PrK}_1}(\mathcal{L}_2)), \tag{2}$$

$$\mathcal{L}_2 = U_1 \| Nym_1 \| r_1 P \| U_n \| Nym_n \| r_n P \| \mathcal{L}_1\ , \tag{3}$$

Round 2: U_{n+1} obtains Nym_{n+1}, chooses $r_{n+1} \in_R \mathbb{Z}_q$ and computes two subkeys $k_{n+1} = h(\hat{e}(u_1, \mathrm{PrK}_{n+1}) \| r_{n+1} r_1 P)$ and $k'_n = h(\hat{e}(u_n, \mathrm{PrK}_{n+1})) \| r_{n+1} r_n P)$. U_{n+1} then sends his information to U_1 and U_n:

$$U_{n+1} \rightarrow U_1, U_n : Nym_{n+1}, r_{n+1}P, X_{n+1}, \quad \text{where } X_{n+1} = k_{n+1}/k'_n. \tag{4}$$

Round 3: U_1 and U_n compute k_{n+1} and k'_n respectively:
$k_{n+1} = h(\hat{e}(u_{n+1}, \mathrm{PrK}_1) \| r_1 r_{n+1} P),\ k'_n = h(\hat{e}(u_{n+1}, \mathrm{PrK}_n) \| r_n r_{n+1} P)$. Then they compute the new X-values $X'_1 = k_1/k_{n+1}$ and $X'_n = k'_n/k_{n-1}$, and distribute them to the new group:

$$U_n \rightarrow U_1 : X'_n\ ,$$

$$U_1 \rightarrow U_{n+1} : \mathrm{E}_{\mathrm{PuK}_{n+1}}(\mathcal{N}_1 \| \mathrm{Sig}_{\mathrm{PrK}_1}(\mathcal{N}_1)), \quad \mathcal{N}_1 = X'_1 \| X_2 \| \ldots \| X_{n-1} \| X'_n\ ,$$

$$U_1 \rightarrow * : E_K(\mathcal{N}_2 \| \mathrm{Sig}_{\mathrm{PrK}_1}(\mathcal{N}_2)), \quad \mathcal{N}_2 = X'_1 \| X_{n+1} \| X'_n\ .$$

Group key K update: Every group member (including U_{n+1}) can now compute all the subkeys $k_i,\ i = 1, \ldots, n+1$ with the altered k'_n and the new k_{n+1} with the following sequence of calculations[4] (illustrated for user U_{n-1}):

[1] $h : \mathbb{G}_2 \times \mathbb{G}_1 \rightarrow \{0,1\}^k$ is a hash function with security parameter k
[2] Subscript numbers are considered modulo n.
[3] $H : \{0,1\}^* \rightarrow \{0,1\}^k$ is a hash function
[4] Note that all subscript numbers are considered modulo $n+1$.

$k'_n = k_{n-1}X'_n;\ k_{n+1} = k'_n X_{n+1};\ k_{n+2} = k_1 = k_{n+1}X'_1;\ k_{2n} = k_{n-1} = k_{n-2}X_{n-1}$.
The updated group key $K' = H(k_1\|k_2\|\ldots\|k'_n\|k_{n+1})$, is formed and checked in the same way as in the initialisation protocol.

Leave Protocol

This protocol ensures forward secrecy when a group member U_i leaves.

Round 1: U_1 assigns new pseudonyms to U_{i-1} and U_{i+1}, using the current group key K:

$$U_1 \to U_{i-1}, U_{i+1} : E_K(\mathcal{L}\|\mathrm{Sig}_{\mathrm{PrK}_1}(\mathcal{L})), \qquad (5)$$
$$\mathcal{L} = U_i\|Nym_i\|U_{i-1}\|Nym'_{i-1}\|U_{i+1}\|Nym'_{i+1}\ .$$

Round 2: U_{i-1} and U_{i+1} verify the signature of U_1 and exchange new parameters $r'_{i-1}P$ and $r'_{i+1}P$ using their new pseudonyms:

$$U_{i-1} \to U_{i+1} : Nym'_{i-1}, r'_{i-1}P, \qquad U_{i+1} \to U_{i-1} : Nym'_{i+1}, r'_{i+1}P\ .$$

Round 3: U_{i-1} and U_{i+1} recompute their (equal) subkeys k'_{i-1} and k'_i:

$$U_{i-1} : k'_{i-1} = h(\hat{e}(u_{i+1}, \mathrm{PrK}_{i-1})\|r'_{i-1}r'_{i+1}P)\ ,$$
$$U_{i+1} : k'_i = h(\hat{e}(u_{i-1}, \mathrm{PrK}_{i+1})\|r'_{i+1}r'_{i-1}P)\ .$$

Next, the updated X-values are computed and distributed:

$$U_{i-1} \to U_1 : X'_{i-1} = \frac{k'_{i-1}}{k_{i-2}}, \quad U_{i+1} \to U_1 : X'_{i+1} = \frac{k_{i+1}}{k'_i} = \frac{k_{i+1}}{k'_{i-1}},$$
$$U_1 \to * : E_K(\mathcal{N}\|\mathrm{Sig}_{\mathrm{PrK}_1}(\mathcal{N})), \quad \mathcal{N} = U_i\|U_{i-1}\|U_{i+1}\|X'_{i-1}\|X'_{i+1}. \qquad (6)$$

Group key K update: The remaining $n-1$ group members can now compute the updated group key $K' = H(k_1\|k_2\|\ldots\|k'_{i-1}\|k_{i+1}\|\ldots\|k_n)$, which is checked in the same way as in the initialisation protocol.

Security Properties

Forward and backward secrecy are not met in the described protocols, contrary to the claims in [1]. In the adversary model, we assume a global, active attacker who is capable of eavesdropping, injecting, modifying or dropping messages within the network at will.

Forward secrecy: In the leave protocol described above, forward secrecy is not guaranteed. In round 3 of the protocol, the leaving member U_i can obtain the values X'_{i-1} and X'_{i+1} in two ways: they are sent unprotected to U_1 AND they are broadcasted under the old group key K in (6). U_i already knows k_i $(i = 1, \ldots, n)$, from the old group key, and only needs to recover the updated subkey $k'_{i-1} = k'_i$, to get the new group key $K' = H(k_1\|\ldots\|k_{i-2}\|\mathbf{k'_{i-1}}\|k_{i+1}\|\ldots\|k_n)$, which can be done from X'_{i-1} and X'_{i+1}:

$$k'_{i-1} = k_{i-2}X'_{i-1}, \quad k'_i = k_{i+1}X'_{i+1}.$$

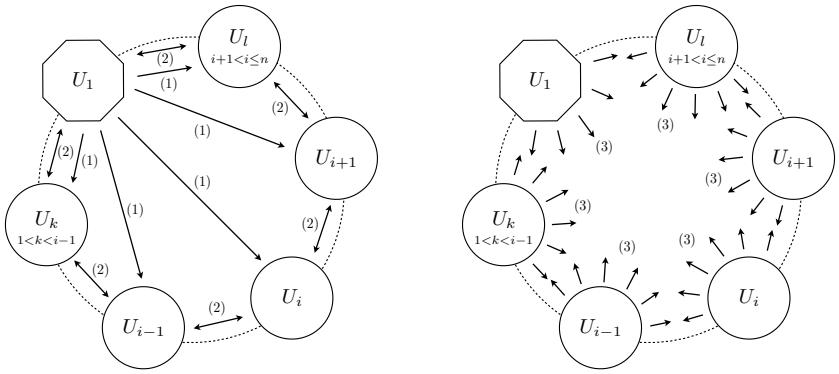
Backward secrecy: Backward secrecy is not ensured by the group member join protocol. At the end of this protocol, user U_{n+1} computes the new group

key $K' = H(k_1\| \ldots \|k_{n-1}\|k'_n\|k_{n+1})$ knowing all subkeys k_i. To be able to compute K, only k_n is missing, which can be computed from $X_1 = k_1/k_n$ or $X_n = k_n/k_{n-1}$, sent around unencrypted in the previous session, which U_{n+1} could have monitored.

3 Improved Protocol

Forward and backward secrecy can be ensured by the following improved leave/join protocols. Our improvement, partially based on ideas of Jung [9], makes use of a session ID, denoted as SID. This random string, unique for each new group session is newly generated and distributed by U_1 in each join/leave protocol. The SID will blind all subkeys k, such that each member will affect the updated group key. Below we describe the difference to the original protocol, for the building blocks described in Sect. 2.

3.1 Initialisation Protocol



(1) $U_1 \to U_i : \mathrm{E}_{\mathrm{PuK}_i}(\mathcal{L}\|\mathrm{Sig}_{\mathrm{PrK}_1}(\mathcal{L})), r_1 P$,
$\mathcal{L} = U_1\| \ldots \|U_n\|Nym_1\| \ldots \|Nym_n\|\mathrm{SID}$
(2) $U_i \to U_{i-1}, U_{i+1} : Nym_i, r_i P$

(3) $U_i, (i = 1, \ldots, n) \to * : Nym_i, X_i$,
$X_i = H(k_i, \mathrm{SID}) \oplus H(k_{i-1}, \mathrm{SID})$

(a) Round-step 1 & 2 (b) Round-step 3

Fig. 1. Initialisation Protocol

Round 1: U_1 generates a SID and adds it to the encrypted and signed message, sent to each user U_i, in which $\mathcal{L} = U_1\| \ldots \|U_n\|Nym_1\| \ldots \|Nym_n\|\mathrm{SID}$:

$$U_1 \to U_i : \mathrm{E}_{\mathrm{PuK}_i}(\mathcal{L}\|\mathrm{Sig}_{\mathrm{PrK}_1}(\mathcal{L})), r_1 P\,. \tag{7}$$

Round 2: No changes to the original protocol.
Round 3: Each U_i calculates k_i and k_{i-1} as in the original. X_i is different:

$$X_i = H(k_i\|\mathrm{SID}) \oplus H(k_{i-1}\|\mathrm{SID})\,.$$

X_i now also depends on SID, and will be updated with each change of the group. As before Nym_i, X_i is broadcasted to all other users.

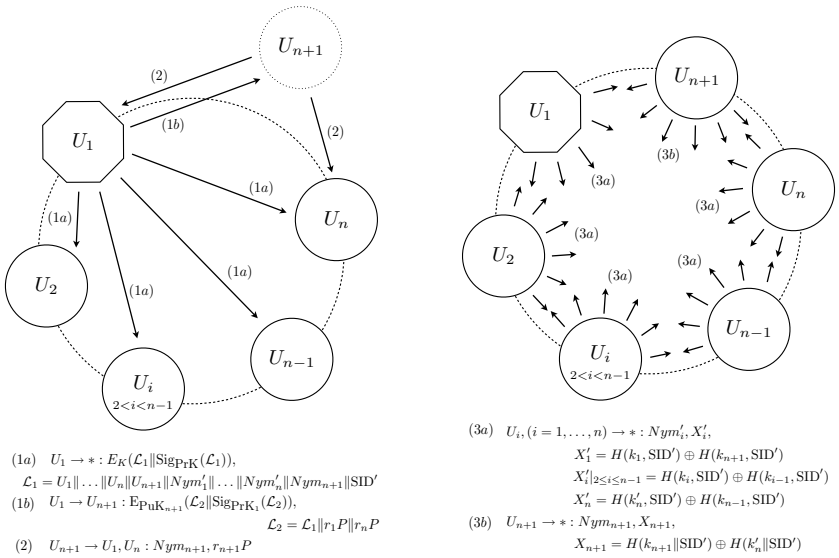
Group key K generation: Each U_i executes a series of calculations:

$$H(k_{i+1}, \mathrm{SID}) = H(k_i, \mathrm{SID}) \oplus X_{i+1}\,, H(k_{i+2}, \mathrm{SID}) = H(k_{i+1}, \mathrm{SID}) \oplus X_{i+2}\,,$$
$$\ldots$$
$$H(k_{i+n-1}, \mathrm{SID}) = H(k_{i+n-2}, \mathrm{SID}) \oplus X_{i+n-1}\,.$$

At the end, U_i verifies if $H(k_{i+n}, \mathrm{SID}) = H(k_{i+n-1}, \mathrm{SID}) \oplus X_{i+n} = H(k_i, \mathrm{SID})$, and the group key K is formed: $K = H(H(k_1, \mathrm{SID}) \| H(k_2, \mathrm{SID}) \| \ldots \| H(k_n, \mathrm{SID}))\,,$

Group key K consistency verification: In the original scheme, each $U_i, (i \neq 1)$ sends the *same* check value to U_1, which means that U_1 cannot verify its origin. In our version of the protocol, the confirmation message can only be generated by a legitimate U_i: $U_i \rightarrow U_1 : E_K(U_i \| \mathrm{Sig}_{\mathrm{PrK}_i}(K))$.

3.2 Join Protocol



(1a) $U_1 \rightarrow * : E_K(\mathcal{L}_1 \| \mathrm{Sig}_{\mathrm{PrK}}(\mathcal{L}_1))$,
$\mathcal{L}_1 = U_1 \| \ldots \| U_n \| U_{n+1} \| Nym'_1 \| \ldots \| Nym'_n \| Nym_{n+1} \| \mathrm{SID}'$

(1b) $U_1 \rightarrow U_{n+1} : \mathrm{E}_{\mathrm{PuK}_{n+1}}(\mathcal{L}_2 \| \mathrm{Sig}_{\mathrm{PrK}_1}(\mathcal{L}_2))$,
$\mathcal{L}_2 = \mathcal{L}_1 \| r_1 P \| r_n P$

(2) $U_{n+1} \rightarrow U_1, U_n : Nym_{n+1}, r_{n+1} P$

(a) Round-step 1 & 2

(3a) $U_i, (i = 1, \ldots, n) \rightarrow * : Nym'_i, X'_i$,
$X'_1 = H(k_1, \mathrm{SID}') \oplus H(k_{n+1}, \mathrm{SID}')$
$X'_i|_{2 \leq i \leq n-1} = H(k_i, \mathrm{SID}') \oplus H(k_{i-1}, \mathrm{SID}')$
$X'_n = H(k'_n, \mathrm{SID}') \oplus H(k_{n-1}, \mathrm{SID}')$

(3b) $U_{n+1} \rightarrow * : Nym_{n+1}, X_{n+1}$,
$X_{n+1} = H(k_{n+1} \| \mathrm{SID}') \oplus H(k'_n \| \mathrm{SID}')$

(b) Round-step 3

Fig. 2. Join Protocol

Round 1: U_1 generates a new session ID, denoted as SID'. Next U_1 informs all members $U_i, (i = 1, \ldots, n)$ and U_{n+1} about U_{n+1}'s joining, and assigns new pseudonyms Nym'_i to all U_i and a pseudonym Nym_{n+1} to U_{n+1}[5] :

$$U_1 \rightarrow * : E_K(\mathcal{L}_1 \| \mathrm{Sig}_{\mathrm{PrK}_1}(\mathcal{L}_1)),$$
$$\mathcal{L}_1 = U_1 \| \ldots \| U_n \| U_{n+1} \| Nym'_1 \| \ldots \| Nym'_n \| Nym_{n+1} \| \mathrm{SID}' \quad , \quad (8)$$

[5] Note that the encryption algorithms in (8) and (9) are different.

$$U_1 \rightarrow U_{n+1} : \mathrm{E}_{\mathrm{PuK}_{n+1}}(\mathcal{L}_2 \| \mathrm{Sig}_{\mathrm{PrK}_1}(\mathcal{L}_2)), \quad \mathcal{L}_2 = \mathcal{L}_1 \| r_1 P \| r_n P \ , \tag{9}$$

Round 2: After decryption, U_{n+1} obtains the pseudonyms of all $U_i, (i = 1, \ldots, n+1)$, chooses $r_{n+1} \in_R \mathbb{Z}_q$ and computes k_{n+1} and k'_n as in the original protocol. He also computes his own $X_{n+1} = H(k_{n+1} \| \mathrm{SID}') \oplus H(k'_n \| \mathrm{SID}')$ and sends $Nym_{n+1}, r_{n+1}P$ to U_1 and U_n.

Round 3: Upon reception of $r_{n+1}P$, U_1 and U_n can compute subkeys k_{n+1} and k'_n as before and recompute their X-values:

$$U_1 : X'_1 = H(k_1, \mathrm{SID}') \oplus H(k_{n+1}, \mathrm{SID}') \ ,$$
$$U_n : X'_n = H(k'_n, \mathrm{SID}') \oplus H(k_{n-1}, \mathrm{SID}') \ .$$

The other users U_i need to update their X-value as well:

$$U_i, (i = 2, \ldots, n-1) : X'_i = H(k_i, \mathrm{SID}') \oplus H(k_{i-1}, \mathrm{SID}') \ .$$

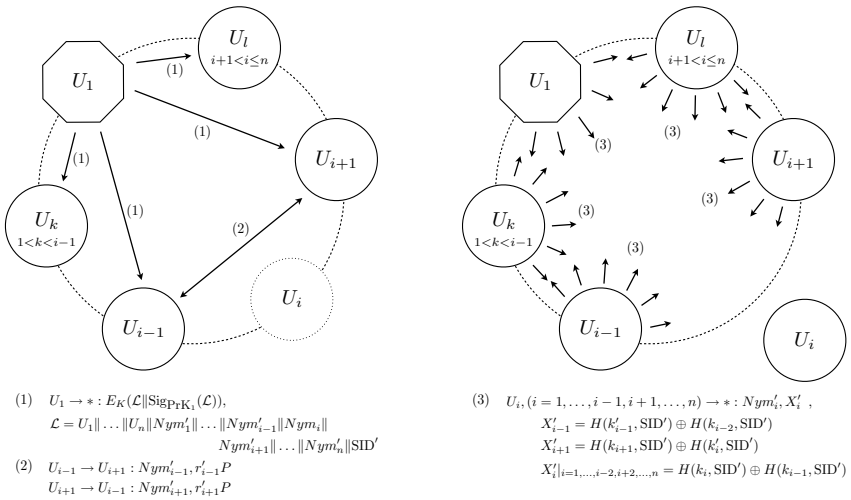
Finally, all group members broadcast their new X-values to all other users along with their new pseudonym:

$$U_i, (i = 1, \ldots, n) \rightarrow * : Nym'_i, X'_i \ , \quad U_{n+1} \rightarrow * : Nym_{n+1}, X_{n+1} \ .$$

Group key K update: Each user can now compute every $H(k_i \| \mathrm{SID}')$, using the X-values, and compute and verify the new group key K' [6]:

$$K' = H(H(k_1, \mathrm{SID}') \| \ldots \| H(k'_n, \mathrm{SID}') \| H(k_{n+1}, \mathrm{SID}')) \ .$$

3.3 Leave Protocol



(a) Round-step 1 & 2 (b) Round-step 3

Fig. 3. Leave Protocol

[6] The consistency verification is identical to the one in the initialisation protocol.

Round 1: When U_i wants to leave the current session, U_1 generates a new SID$'$ and informs all remaining group members by broadcasting:

$$U_1 \rightarrow * : E_K(\mathcal{L} \| \mathrm{Sig}_{\mathrm{PrK}_1}(\mathcal{L})),$$
$$\mathcal{L} = U_1 \| \ldots \| U_n \| Nym'_1 \| \ldots \| Nym'_{i-1} \| Nym_i \| Nym'_{i+1} \| \ldots \| Nym'_n \| \mathrm{SID}' \ ,$$

in which new pseudonyms Nym'_i are assigned.
Round 2: Upon reception of their new pseudonym, U_{i-1} and U_{i+1} exchange their new values $r'_{i-1}P$ and $r'_{i+1}P$, as in the original protocol.
Round 3: Then, U_{i-1} and U_{i+1} recompute their subkeys k'_{i-1} and k'_i, again as in the original protocol, and compute their X-values:

$$U_{i-1} : X'_{i-1} = H(k'_{i-1}, \mathrm{SID}') \oplus H(k_{i-2}, \mathrm{SID}') \ ,$$
$$U_{i+1} : X'_{i+1} = H(k_{i+1}, \mathrm{SID}') \oplus H(k'_i, \mathrm{SID}') \ .$$

The other users U_i need to update their X-value as well:

$$U_i, (i = 1, \ldots, i-2, i+2, \ldots, n) : X'_i = H(k_i, \mathrm{SID}') \oplus H(k_{i-1}, \mathrm{SID}') \ .$$

Finally, all remaining group members broadcast their new X-value:

$$U_i, (i = 1, \ldots, i-1, i+1, \ldots, n) \rightarrow * : Nym'_i, X'_i \ .$$

Group key K update: Each user can now compute every $H(k_i \| \mathrm{SID}')$[7], using the X-values, and compute and verify the new group key K' [8]:

$$K' = H(H(k_1, \mathrm{SID}') \| \ldots \| H(k'_{i-1}, \mathrm{SID}') \| H(k_{i+1}, \mathrm{SID}') \| \ldots \| H(k_n, \mathrm{SID}')).$$

4 Security and Performance Analysis

4.1 Security

Here we informally show that our scheme ensures all the requirements, including forward/backward secrecy. In the security analysis below, we assume that the adversary knows the current SID, unless otherwise stated.
Anonymity: The identities of the users U_i are encrypted at all times in the protocols, either by anonymous ID-based encryption using their public keys PuK_i or by symmetric encryption using the group key K. In all protocols, identities are masked by pseudonyms constructed by the initiator U_1. The use of identities and private keys in the calculation of the values X_i is masked by hashing with a random point in $\mathbb{G}_1$; identity information cannot be retrieved without the master secret s or the user's private key PrK_i.
Unlinkability: Anonymity alone only hides the real identities of the group members. To prevent an adversary from tracing a user by his pseudonym, pseudonyms or X-values are never re-used when the group changes and a new group key needs to be established. Therefore, a user U_i is able to join or leave a private group session anonymously.

[7] The equality $k'_{i-1} = k'_i$ is necessary for the correctness of the sequence of calculations.
[8] The consistency verification is identical to the one in the initialisation protocol.

Group key secrecy and perfect forward secrecy: If an attacker knows SID, the group key $K = H(H(k_1, \text{SID}) \| H(k_2, \text{SID}) \| \dots \| H(k_n, \text{SID}))$ remains secure: to retrieve K, he needs access to at least one user's shared subkey k_i, to compute the other subkeys k_j from the broadcasted X-values. The subkey $k_i = h(\hat{e}(u_{i+1}, \text{PrK}_i) \| r_i r_{i+1} P)$ cannot be computed without knowing U_{i+1}, the private key PrK_i and the element $r_i r_{i+1} P$:

- The IDs U_i are encrypted, and only known to the participating members;
- The private key of the users is never shared and can only be computed with the master secret s;
- The element $r_i r_{i+1} P$ cannot be computed from the exchanged $r_i P$ and $r_{i+1} P$ because of the ECDHP assumption [10].

Knowing the identities U_i next to SID, is also insufficient to compute k_i without having knowledge of the master secret s or the private keys PrK_i, since $\hat{e}(u_{i+1}, \text{PrK}_i) = \hat{e}(u_{i+1}, s u_i)$. The argumentation above shows that the group key K cannot be retrieved by the adversary in our threat model.

Even in case when the long-term master secret s is compromised and the identities of the participating members are revealed, group keys from previous stages remain uncompromised, because of the last argument in the list above, such that the protocol is also perfectly forward secret.

Forward and backward secrecy: Forward and backward secrecy should be ensured for the leaving and joining protocol respectively. Each time the group membership changes, the initiator U_1 introduces a new SID' and the group key is updated. While each user U_i still computes both subkeys k_i and k_{i-1}, he will share $H(k_i \| \text{SID}')$, instead of the unprotected k_i. This is done by broadcasting $X'_i = H(k_i \| \text{SID}') \oplus H(k_{i-1} \| \text{SID}')$. A joining/leaving member is unable to compromise a past/future group key K:

1. the only subkeys k a joining/leaving member knows are his own, which are both newly generated or updated during the join/leave protocol such that the knowledge of the previous SID/updated SID' cannot be used to compute a previous/updated group key;
2. each user's shared contribution and the corresponding X-value are SID-dependent, and are updated and re-broadcasted in the join/leave protocol.

Forward secrecy for the leaving protocol: In round 1, the leaving member U_i gains knowledge of the updated SID' since it is encrypted using the old group key K. This is no problem for the forward secrecy, as U_i can only generate $H(k_i \| \text{SID}')$ and $H(k_{i-1} \| \text{SID}')$; these are exactly the values that are updated by U_{i-1} and U_{i+1} in round 3, and shared through updated X'-values:

$$U_{i-1} : X'_{i-1} = H(k'_{i-1}, \text{SID}') \oplus H(k_{i-2}, \text{SID}') \ ,$$
$$U_{i+1} : X'_{i+1} = H(k_{i+1}, \text{SID}') \oplus H(k'_i, \text{SID}') \ ,$$

where each shared contribution is updated with the new SID'. Because U_i only knows $H(k_{i-1}, \text{SID}'), H(k_{i-2}, \text{SID}), H(k_{i+1}, \text{SID}), H(k_i, \text{SID}')$, he is unable to retrieve any useful information from X'_{i-1} and X'_{i+1} to retrieve the new K'.

Backward secrecy for the joining protocol: In round 2, the joining member U_{n+1} computes both subkeys k_{n+1} and k'_n and shares this through $X_{n+1} = H(k_{n+1}\|\text{SID}') \oplus H(k'_n\|\text{SID}')$. If U_{n+1} knows the previous session ID, i.e. SID, he can compute $H(k_{n+1}\|\text{SID})$ and $H(k'_n\|\text{SID})$. However, this does not give him an advantage in recovering any past group key K since the subkey k_n has been updated. The shared contributions of all other group members are now masked with a new SID$'$, hence knowing $H(k_1\|\text{SID}')$ in combination with $X_1 = H(k_1\|\text{SID}) \oplus H(k_n\|\text{SID})$ does not help to retrieve K.

4.2 Performance

In Tables 1 and 2, we give an overview of the communication and computation cost of the new protocol, compared to the original protocol by Wan *et al.*

Communication cost: the number of broadcast messages in the join and leave protocol is higher: each time the group (of size n) changes, each U_i needs to broadcast his updated share $H(k_i\|\text{SID}')$ to all other users.

Table 1. Communication Cost Comparison

Protocols		Rounds	Messages	Unicast	Broadcast
Wan *et al.*	Initialise	3	$4n$	$3n$	n
	Join	3	7	6	1
	Leave	3	7	6	1
Our protocol	Initialise	3	$4n$	$3n$	n
	Join	3	$n+5$	3	$n+2$
	Leave	3	$n+2$	2	n

Computation cost: the increase in signature generations and verifications is due to (1) the initiator U_1 broadcasts his signed message containing the newly reassigned pseudonyms Nym'_i and updated SID$'$ during the join/leave protocol, and (2) the improved group key consistency verification process at the end of each protocol.

Table 2. Computation Cost Comparison

Protocols		ID-based Encryption	Pairing Computation ($\hat{e}$)	Signature Gen./Verif.		Point Multiplication (in $\mathbb{G}$)
Wan *et al.*	Initialise	$n-1$	$2n$	1	$n-1$	$3n$
	Join	3	4	4	$n+3$	5
	Leave	0	2	2	$n+1$	4
Our protocol	Initialise	$n-1$	$2n$	n	$2n-2$	$3n$
	Join	1	4	$n+2$	$2n+1$	5
	Leave	0	2	$n-1$	$2n-4$	4

5 Application: Virtual Private Ad Hoc Network

As more and more mobile devices interconnect though largescale IP networks, new network architectures become important. Virtual Private Ad Hoc Network (VPAN) is a concept that aims to establish a secure virtual network

on top of the existing insecure IP base network by combining network virtualisation and ad hoc networking techniques. This concept was proposed and introduced in [11] and [12].

Due to geographical distribution of VPAN entities, clusters of entities – *VPAN Nodes* – that are able to connect to each other directly are formed, in which a special node, the *Gateway Node*, has connectivity to an IP-based access network. Within the same VPAN, clusters are interconnected through their Gateway Nodes. The VPAN membership is self-organising: members need to be able to discover each other and form a secure overlay without user intervention. Additionally, ad hoc routing techniques are used for efficient internal routing. A VPAN is identified by a VPAN prefix, which is prepended to the VPAN Node ID for every node in a VPAN, such that one node can be active in multiple VPANs.

Our privacy-preserving ID-based group key agreement scheme described in Sect. 3 can be applied to the VPAN setting to protect the privacy of VPAN cluster nodes to the outside world as well as to obtain a shared group session key within each cluster while supporting dynamic cluster membership. More specifically, VPAN Nodes remain anonymous and an outside eavesdropper is unable to trace or monitor activities of a specific VPAN Node, or to link the same VPAN Node in clusters of different VPANs.

The initiator U_1, which should have an IP connection, assumes the role of *Gateway Node* GN, while the remaining users $U_i|_{2\leq i\leq n}$ act as *VPAN Nodes*. Hence a group of users $\{U_i|_{1\leq i\leq n}\}$ is here referred to as a cluster of VPAN members $\{GN, U_i|_{2\leq i\leq n}\}$. To ensure that VPAN Nodes can form new clusters or join an existing cluster, the GN of each VPAN cluster broadcasts periodically the following beacon message: $GN \rightarrow * : GN \| \mathrm{Sig}_{\mathrm{PrK}_{GN}}(GN)$, where GN denotes the Gateway Node's public key.

Initialisation Protocol: cluster initialisation occurs when setting up a new VPAN or forming a new cluster within an existing VPAN. Each VPAN Node U_i receiving GN's beacon, responds with the following encrypted message: $U_i \rightarrow GN : \mathrm{E}_{\mathrm{PuK}_{GN}}(\mathcal{L} \| \mathrm{Sig}_{\mathrm{PrK}_i}(\mathcal{L}))$ with $\mathcal{L} = U_i \| \mathrm{VPAN}_{\mathrm{prefix}}$. At the end, GN has a set of VPAN Nodes $\{U_i|_{2\leq i\leq n}\}$ to form a cluster and to agree upon a shared group session key.

Join Protocol: when a new VPAN Node U_{n+1} wants to join an existing cluster $\{GN, U_i|_{2\leq i\leq n}\}$, he waits for the beacon and responds with the following encrypted message: $U_{n+1} \rightarrow GN : \mathrm{E}_{\mathrm{PuK}_{GN}}(\mathcal{L} \| \mathrm{Sig}_{\mathrm{PrK}_{n+1}}(\mathcal{L}))$ with $\mathcal{L} = U_{n+1} \| \mathrm{VPAN}_{\mathrm{prefix}}$. During the join protocol, the shared group session key is updated to provide backward secrecy.

Leave Protocol: when a VPAN Node U_i wants to leave an existing cluster, the leave protocol is executed to update the group session key and thus to provide forward secrecy.

The protocols themselves remain exactly the same as described in Sect. 3, all initiated by the Gateway Node GN.

6 Conclusion and future work

In this paper, we showed that the key agreement protocol by Wan *et al.* does not offer forward and backward secrecy, contrary to their claims. We adjusted the protocol such that these requirements are met and added an extra safeguard at the end of the protocol. The cost for these improvements is a increased computation cost and a moderately higher communication cost.
Future work includes a thorough investigation of the role of the network in which our protocol will operate; this can be done in a VPAN setting, for which test infrastructures exist already. The designers of the referenced VPAN were involved in this work, and collaboration seems possible.
The cost of running an anonymous routing mechanism in a multihop ad hoc network, as suggested by the original protocol authors, cannot be neglected. Furthermore, privacy-preserving ID-based protocols in which multiple members can join and leave simultaneously and in which groups can merge and split are still to be developed.

Acknowledgements

This work was supported in part by Research Council K.U.Leuven (GOA TENSE), by the IAP Programme P6/26 BCRYPT of the Belgian State (Belgian Science Policy), and by IBBT (Interdisciplinary institute for BroadBand Technology) of the Flemish Government.

References

1. Wan, Z., Ren, K., Lou, W., Preneel, B.: Anonymous id-based group key agreement for wireless networks. In: IEEE WCNC, Network Track. (2008)
2. Shamir, A.: Identity-based cryptosystems and signature schemes. Proceedings of CRYPTO 84 on Advances in cryptology table of contents (1985) 47–53
3. Boneh, D., Franklin, M.: Identity-Based Encryption from the Weil Pairing. SIAM Journal on Computing **32** (2003) 586
4. Sakai, R., Ohgishi, K., Kasahara, M.: Cryptosystems based on pairing. The 2000 Symposium on Cryptography and Information Security (2000) 26–28
5. Dutta, R., Dowling, T.: Secure and efficient group key agreements for cluster based networks. Transactions on Computational Science **4** (2009) 87–116
6. Dutta, R., Barua, R.: Overview of key agreement protocols. Cryptology ePrint Archive, Report 2005/289 (2005) http://eprint.iacr.org/.
7. Burmester, M., Desmedt, Y.: A secure and efficient conference key distribution system (extended abstract). In: EUROCRYPT. (1994) 275–286
8. Smart, N.: Identity-based authenticated key agreement protocol based on Weil pairing. Electronics Letters **38**(13) (2002) 630–632
9. Jung, B.E.: An efficient group key agreement protocol. IEEE Communications Letters **10**(2) (2006) 106–107
10. Brown, D.: Standards for efficient cryptography, SEC 1: elliptic curve cryptography. Technical report, Certicom Research (2009)
11. Hoebeke, J.: Adaptive Ad Hoc Routing and Its Application to Virtual Private Ad Hoc Networks. PhD thesis, Universiteit Gent (2007)
12. Hoebeke, J., Holderbeke, G., Moerman, I., Dhoedt, B., Demeester, P.: Virtual Private Ad Hoc Networking. Wireless Personal Communications **38**(1) (2006) 125–141