Attribute-based group key establishment

Rainer Steinwandt and Adriana Suárez Corona

[1] Department of Mathematical Sciences, Florida Atlantic University, Boca Raton, FL 33431, USA, email: `rsteinwa@fau.edu`
[2] Departamento de Matemáticas, Universidad de Oviedo, 33007 Oviedo, Spain, email: `adriana@orion.ciencias.uniovi.es`

Abstract. Motivated by the problem of establishing a session key among parties based on the possession of certain credentials only, we discuss a notion of *attribute-based key establishment*. A number of new issues arise in this setting that are not present in the usual settings of group key establishment where unique user identities are assumed to be publicly available.
After detailing the security model, we give a two-round solution in the random oracle model. As main technical tool we introduce a notion of attribute-based signcryption, which may be of independent interest. We show that the type of signcryption needed can be realized through the encrypt-then-sign paradigm. Further, we discuss additional guarantees of the proposed protocol, that can be interpreted in terms of deniability and privacy.

Keywords: group key establishment, attribute-based cryptography, signcryption

AMS classification: 94A60

1 Introduction

In the context of group key establishment, protocol participants are typically modeled as Turing machines $U_1, \ldots, U_n$, and a unique identifier for each protocol participant is assumed to be publicly known. This identifier is usually identified with U_i and used to specify with whom a key is to be established. It can also be used to impose a virtual connection topology among participants, e. g., the construction of Burmester and Desdmedt in [8] arranges parties in a circle with neighborhood relations being determined by an ordering on the set of identifiers.

In this paper we consider a scenario where participants in a group key establishment aim at obtaining a common session key with partners having certain attributes, disregarding individual identities. This can, for instance, mean that a key is to be established with members of a department that have the right to negotiate agreements of a certain value. In a two-party setting it could mean that a member from the sales department wants to establish a key with anyone in human resources who is entitled to deal with healthcare issues, and the representative in human resources establishes keys only with any representative of a department committee. The essential point is that we do not distinguish between individual user identities, but each participant specifies the attributes she expects her partners to have and the session key should be available to users that meet all imposed conditions. Another scenario where attribute-based group key establishment seems interesting is a project in an enterprise (or crossing enterprise boundaries), where project team members need—read and/or write—access to data relevant for the task at hand. In such a scenario, a common key could be established among all members possessing the necessary attributes to work on a particular project, without resorting to individual user identities.

Shifting the focus from individual identities to the possession of attributes, privacy questions naturally arise: depending on the application context, it may be a design goal that users do not have to reveal which exact set of attributes they possess, but only the fact that they possess a qualified set. Consequently, treating a user's set of attributes as a substitute for a public identifier can be problematic. In the protocol below we address this problem through (i) a form of privacy reminiscent of attribute-based encryption with hidden ciphertext policy [17] and attribute-hiding predicate encryption [5, 13], and (ii) through a form of deniability reminiscent of deniable group key establishment [3].

Organization of the paper. For a general introduction to the topic of key establishment, we refer to the book [6] by Boyd and Mathuria. Throughout, we formalize our setting of attribute-based key establishment in Section 2 by adapting the group key establishment model in [4] (which in turn builds on [7, 14]) appropriately; the replacement of unique identifiers for protocol participants with attribute sets raises some technical issues that are to be addressed here. As a technical tool, in Section 3 we start by defining an attribute-based variant of signcryption, a tool which might be of independent interest. We will then use this tool to devise a two-round solution in the random oracle model, based on an attribute-based signcryption scheme.

Related work. As prior work on attribute-based key establishment, Wang et al.'s results in [22, 21, 20] can be mentioned. These three papers address a two-party scenario and suggest solutions for such a setting with [21] and without random oracles [22, 20], respectively. After submission of the original manuscript of our paper in November 2009, further work related to attribute-based key establishment has been made available, evidencing a wider interest in this topic. In particular, Camenisch et al. [9] discuss *Credential-Authenticated Key Exchange* (CAKE), where a two-party key exchange is conditioned on the compatibility of credentials held by the involved parties. Unlike the approach taken below, on the technical side, Camenisch et al. build on Canetti's *Universal Composability* (UC) framework [10]. Gorantla et al. [12] suggest a notion of *attribute-based authenticated key exchange*, with the security being captured in an "oracle based" security model, similar to the model employed below. While the essential working horse in our approach is *attribute-based signcryption*, a main technical tool in [12] is a type of key encapsulation mechanism (KEM), to which Gorantla et al. refer as *encapsulation-policy attribute based KEM.* Their paper presents a construction for deriving a secure attribute-based authenticated key exchange from such a key encapsulation mechanism, assuming the latter fulfills an appropriate security guarantee. In [2], Birkett and Stebila consider *predicate-based key authenticated exchange* between two parties. Similarly as in [12] and below, an "oracle based" security model is used. The authors of [2] show how to achieve both *credential privacy* and *session key security* by combinining a suitable *predicate-based signature scheme* with a Diffie-Hellman key exchange.

2 Security model

By ℓ we denote the security parameter, and by $\mathcal{U} \subseteq \{0,1\}^{\mathrm{O}(1)}$ a non-empty constant-size *universe of attributes.*

2.1 Communication model and adversarial capabilities

Participants and initialization. The set of potential protocol participants in an attribute-based group key establishment are probabilistic polynomial time (ppt) Turing machines labeled with subsets of $\mathcal{U}$, and in slight abuse of terminology we will speak of a "protocol participant U", identifying a Turing machine with its unique label. Analogously as in attribute-based encryption, an identifier $U \in 2^{\mathcal{U}}$ represents any user having exactly the attributes contained in U; we *do not distinguish among users possessing identical attributes.* During a trusted initialization phase, a master key mk is chosen and used to derive the public system parameters pk as well as secret (attribute) keys ak_U for each $U \in 2^{\mathcal{U}}$. The secret key ak_U is stored by protocol participant U as long-term secret.

Each protocol participant U may execute a polynomial number of protocol instances in parallel, and we will refer to instance s of protocol participant $U \in 2^{\mathcal{U}}$ as Π_U^s ($s \in \mathbb{N}$). Each such instance has associated seven variables: $\mathtt{used}_U^s$, $\mathtt{state}_U^s$, $\mathtt{term}_U^s$, $\mathtt{sid}_U^s$, $\mathtt{pid}_U^s$, $\mathtt{acc}_U^s$ and $\mathtt{sk}_U^s$:

$\mathtt{used}_U^s$ indicates if the instance is or has been used for a protocol run. The $\mathtt{used}_U^s$ flag can only be set through a protocol message received by the instance due to a call to the Send-oracle;

$\mathtt{term}_U^s$ shows if the execution has terminated;

$\mathtt{state}_U^s$ keeps the state information during the protocol execution;

$\mathtt{acc}_U^s$ indicates if the protocol instance was succesful, i. e, if the the session key has been accepted by U;

$\mathtt{sk}_U^s$ stores the session key once it is accepted by the instance Π_U^s . Before acceptance, it stores a distinguished NULL value.

$\mathtt{sid}_U^s$ denotes a (non-secret) session identifier that can serve as identifier for the session key sk_U^s ;

$\mathtt{pid}_U^s$ stores the possible sets of attributes a user U aims at establishing a key with, i. e., $\mathtt{pid}_U^s \subseteq 2^{\mathcal{U}}$ such that $U \in \mathtt{pid}_U^s$;

Remark 1. Note that the role of $\mathtt{pid}_U^s$ differs from "ordinary" authenticated key establishment: we interpret $\mathtt{pid}_U^s$ as *access structure* specifying the qualified sets of attributes, which in turn may be regarded as representing acceptable communciation partners. In particular, for a successful protocol execution we will not impose that *all* $U' \in \mathtt{pid}_U^s$ participate—but only $U' \in \mathtt{pid}_U^s$ may obtain the established session key. In a threshold-based setting, $\mathtt{pid}_U^s$ could consist of all subsets of $\mathcal{U}$ with cardinality greater than some threshold.

Communication network. We assume that arbitrary point-to-point connections among parties are available. As connections are under adversarial control (cf. the adversarial model below), the network is non-private and fully asynchronous. In particular, when *broadcasting* a message, this means that the adversary can create a situation where the protocol participants receive in fact different messages or only a subset of the participants receives the message.

Adversarial capabilities. The adversary $\mathcal{A}$ is modeled as ppt time Turing machine and considered to be active: $\mathcal{A}$ has full control of the communication network and may delay, eavesdrop, suppress, alter and insert messages at will. To make the adversarys capabilities explicit, the subsequently listed oracles are used and can be invoked by $\mathcal{A}$:

$\mathtt{Send}(U, s, M)$ This oracle serves two purposes:

- If $\mathtt{used}_U^s = \text{TRUE}$, the message M is sent to the instance Π_U^s. If Π_U^s sends a message in the protocol right after receiving M, then the $\mathtt{Send}$ oracle returns this message to the adversary.
- If $\mathtt{used}_U^s = \text{FALSE}$, the message M has to be of the form $M = (B, b)$, where $B \subseteq 2^U$ is an access structure and $b \in \{\text{INIT}, \overline{\text{INIT}}\}$ is a role flag. In this way the adversary can initialize a protocol run among principals U' such that each $U' \in B$. The flag b allows to designate a protocol initiator whose computations may differ from those of other protocol participants. After such a query, Π_U^s's $\mathtt{pid}_U^s$-value is initialized to B, the $\mathtt{used}_U^s$-flag is set and Π_U^s processes the first step of a protocol execution. This means that in this session, U aims at establishing a common key with at least one principal $U' \in B \setminus \{U\}$.

$\mathtt{Reveal}(U, s)$ yields the session key $\mathtt{sk}_U^s$ provided that it is defined, i. e., if $\mathtt{acc}_U^s = \text{TRUE}$ and $\mathtt{sk}_U^s \neq \text{NULL}$. Otherwise the distinguished NULL-value is returned.

$\mathtt{Corrupt}(U)$ reveals the long-term secret key ak_U of U to the adversary. Given a concrete protocol run, involving instances Π_U^s, we say that user U is *honest* if and only if no query of the form $\mathtt{Corrupt}(U)$ has been made by the adversary.

$\mathtt{Test}(U, s)$ Only one query of this form is allowed for the adversary $\mathcal{A}$. Provided that sk_U^s is defined, (i. e., $\mathtt{acc}_U^s = \text{TRUE}$ and $\mathtt{sk}_U^s \neq \text{NULL}$), $\mathcal{A}$ can execute this oracle query at any time when being activated. A test bit $t \in \{0, 1\}$ is chosen uniformly at random and if $t = 0$ then the session key sk_U^s is returned. If $t = 1$, then a uniformly chosen random session key is returned.

2.2 Protocol goals

Correctness. This property expresses that in the absence of adversarial interference, the protocol will establish a common key along with a matching identifier:

Definition 1 (Correctness). *An attribute-based group key establishment is* correct *if on honest delivery of all messages and all users being honest, a single protocol execution among users $\mathcal{V} \subseteq 2^{\mathcal{U}}$ involves instances $\Pi_U^{s_U}$ $(U \in \mathcal{V})$ such that with overwhelming probability all of the following hold:*

- *all users in $\mathcal{V}$ accept, i. e.,* $\mathtt{acc}_U^{s_U} = \text{TRUE}$ *for all $U \in \mathcal{V}$;*
- *all users in $\mathcal{V}$ obtain the same session identifier, i. e.,* $\mathtt{sid}_U^{s_U}$ *is identical for all $U \in \mathcal{V}$;*
- *all users in $\mathcal{V}$ accept the same session key, i. e.,* $\mathtt{sk}_U^{s_U}$ *is identical and* $\neq$NULL *for all $U \in \mathcal{V}$;*
- *all communication partners are specified as desired communication partner, i. e., $\mathcal{V} \subseteq \mathtt{pid}_U^{s_U}$ for all $U \in \mathcal{V}$.*

Correctness alone is a rather weak guarantee, as it refers to a scenario where no attack takes place. For instance, the last condition ensures that every protocol participant is aware that the users in $\mathcal{V}$ may know the session key, but no statement is made about the session key being known to users in $\mathcal{U} \setminus \mathcal{V}$—actually, broadcasting the session key to all users is not ruled out by correctness. To formalize security guarantees, we use the following terminology.

Partnering and freshness We have to specify under which circumstances a Test-query may be executed and under which circumstances a correct guess of the adversary constitutes a viable attack. To do so, we fix the following notions of partnering and freshness.

Definition 2 (Partnering). *We say that two instances Π_U^s and $\Pi_{U'}^{s'}$ are* partnered *if* $\mathtt{sid}_U^s = \mathtt{sid}_{U'}^{s'}$, $\mathtt{acc}_U^s = \mathtt{acc}_{U'}^{s'} = \text{TRUE}$, *$U \in \mathtt{pid}_{U'}^{s'}$ and $U' \in \mathtt{pid}_U^s$.*

The notion of partnering is mainly a technical tool, but crucial for capturing the the intuition of a secure key establishment adequately. An adversary is restricted to attacking *fresh* instances, and for an instance to be fresh, in particular no partnered instance must have been queried to the Reveal oracle:

Definition 3 (Freshness). *An instance Π_U^s is said to be* fresh *if none of the following events has occurred:*

- *For some $U' \in \mathtt{pid}_U^s$ a* $\mathtt{Corrupt}(U')$ *query was executed before a query of the form* $\mathtt{Send}(U'', s'', *)$ *has taken place where $U'' \in \mathtt{pid}_U^s$.*
- *The adversary $\mathcal{A}$ queried* $\mathtt{Reveal}(U', s')$ *with Π_U^s and $\Pi_{U'}^{s'}$ being partnered.*

With the above terminology we can capture (semantic) security of an attribute-based key establishment protocol P in the usual way. For an adversary $\mathcal{A}$ attacking an attribute-based key establishment protocol P, we define an advantage function $\mathsf{Adv}_{\mathcal{A}} = \mathsf{Adv}_{\mathcal{A}}(\ell)$ by setting $\mathsf{Adv}_{\mathcal{A}} := |\mathsf{Succ}_{\mathcal{A}}^{\text{sem}} - 1/2|$, where $\mathsf{Succ}_{\mathcal{A}}^{\text{sem}}$ is the probability that the adversary queries the Test oracle on a fresh instance Π_U^s and guesses correctly the test bit t used by the Test oracle.

Definition 4 (Semantic security). *We call an attribute-based group key establishment* secure *if for any ppt adversary $\mathcal{A}$ the function $\mathsf{Adv}_{\mathcal{A}} = \mathsf{Adv}_{\mathcal{A}}(\ell)$ is negligible.*

Remark 2. According to our freshness definition, an adversary is allowed to corrupt all remaining honest parties right before quering `Test` without violating freshness. Thus the above definition of semantic security implies *forward security* in the usual sense: even after having access to all longterm secrets of users, session keys remain indistinguishable from random keys.

In addition to these standard security goals, we adapt the notion of integrity from [4], which can be seen as a correctness guarantee in the presence of an active adversary:

Definition 5 (Integrity). *We say that a correct attribute-based group key establishment fulfills* integrity *if with overwhelming probability all instances of honest parties U, U' that have accepted with the same session identifier $\mathtt{sid}_U^s = \mathtt{sid}_{U'}^{s'}$ hold an identical session key $\mathtt{sk}_U^s = \mathtt{sk}_{U'}^{s'}$, and we have $U \in \mathtt{pid}_{U'}^{s'}$ and $U' \in \mathtt{pid}_U^s$.*

Another possible protocol goal for an attribute-based key establishment is to reveal not more information about the identity of participating users than actually needed: if a user U specifies a particular access structure in a $\mathtt{pid}_U^s$-value, there is no immediate need to reveal which *particular* qualified subset of attributes is used by a communication partner. For instance, if U wants to be sure that its communication partner posseses at least the attributes $u_1, u_2 \in \mathcal{U}$, U does not have to know which other attributes a communication partner has in addition to u_1 and u_2. In this paper we do not offer a formalization of such a guarantee, but in Section 4 will discuss our proposed protocol from this point of view. There, we will also address the question of *deniability* for our protocol: to what extent it is possible to provide convincing evidence to a third party about the involvement of a particular $U \in 2^{\mathcal{U}}$ in a protocol execution.

3 A protocol for attribute-based key establishment

For describing the suggested protocol, an attribute-based variant of signcryption turns out to be helpful. As we are not aware of a discussion of this primitive in the literature, in the next section we give a formalization, the pertinent security definitions and show how concrete instances can be obtained through sequential composition of attribute-based encryption and attribute-based signature schemes.

Remark 3. In our protocol only uniformly at random chosen bitstrings are encrypted, and one could consider the use of an attribute-based variant of a signcryption key encapsulation mechanism—possibly building on the discussion in [15], where Li et al. consider an identity-based variant of such a primitive. In [11] an attribute-based variant of key encapsulation is discussed by Fang et al., but not much work seems to be available on connecting attribute-based cryptography and key encapsulation mechanisms.

3.1 Attribute-based signcryption

Our definition and security model for attribute-based signcryption is modeled after the discussion of standard signcryption by An et al. in [1]. To formalize attribute-based encryption and attribute-based signing, we build on the work by Sahai and Waters [18] and Shahandashti and Safavi-Naini [19] respectively.

Definition 6 (Attribute-based encryption). *An* attribute-based encryption scheme *is a tuple of polynomial time algorithms* (Setup, Gen, Enc, Dec)*:*

Setup *is probabilistic and run by a trusted authority: on input the security parameter 1^ℓ and a universe of attributes $\mathcal{U}$, a master secret key mk and public system parameters pm are generated. The public parameters include a description of the message space $\mathcal{M}$.*

Gen *is probabilistic and run by a trusted authority: on input the master secret key mk and a set of attributes U belonging to a user, a secret key dk_U for these attributes is generated.*

Enc *is probabilistic and run by a user who wants to send a plaintext message m to a user with a set of attributes in the access structure $\mathbb{A}$: on input $m \in \mathcal{M}$ and $\mathbb{A} \subseteq 2^{\mathcal{U}}$, this algorithm generates a ciphertext c.*

Dec *is a deterministic algorithm run by a user with a set of attributes $U \subseteq \mathcal{U}$. On input c and dk_U, this algorithm outputs the underlying plaintext m, if c is a valid encryption of m and U is contained in the access structure $\mathbb{A}$ specified in the computation of c. Otherwise an error symbol $\perp$ is returned.*

For our purposes, where only uniformly at random chosen plaintexts are encrypted, a rather basic security guarantee will be sufficient:

Definition 7 (One-Wayness for attribute-based encryption). *For a ppt adversary $\mathcal{A}$, denote by $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{OW\text{-}CPA}}$ the probability that $\mathcal{A}$ wins the game described in Figure 1. We refer to an attribute-based encryption scheme as* OW-CPA secure in the selective access structure model, *if $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{OW\text{-}CPA}} = \mathsf{Adv}_{\mathcal{A}}^{\mathsf{OW\text{-}CPA}}(\ell)$ is negligible for all ppt adversaries $\mathcal{A}$.*

Example 1. For access structures describing qualified subsets through a threshold, we can employ Sahai and Waters' pairing-based construction in [18] to achieve security in the sense of Definition 7.

A natural approach to derive an attribute-based signcryption scheme as needed for our key establishment protocol, is to compose an OW-CPA secure attribute-based encryption scheme with an existentially unforgeable attribute-based signature scheme:

Definition 8 (Attribute-based signature). *An* attribute-based signature scheme *is a tuple of polynomial time algorithms* (Setup, Gen, Sig, Ver)*:*

Setup *is probabilistic and run by a trusted authority: on input the security parameter 1^ℓ and a universe of attributes $\mathcal{U}$, a master secret key mk and public system parameters pm are generated. The public parameters include a description of the message space $\mathcal{M}$.*

Init phase Given the security parameter 1^ℓ, the adversary $\mathcal{A}$ outputs:
- a non-empty set $\mathcal{U}$, the universe of attributes;
- a non-empty access structure $\mathbb{A} \subseteq 2^{\mathcal{U}}$ that it wants to be challenged upon.

Setup phase The challenger runs Setup and hands the public parameters to $\mathcal{A}$.

Query phase 1 The adversary is allowed to ask (adaptively) queries for
- private decryption keys dk_U for attribute sets $U \subseteq \mathcal{U}$ subject to the restriction $U \notin \mathbb{A}$.

Challenge phase The challenger picks a message m uniformly at random from the message space and hands the resulting ciphertext $\mathsf{Enc}(m, \mathbb{A})$ to the adversary $\mathcal{A}$.[1]

Query phase 2 Identical to Phase 1.

Guess phase The adversary outputs a guess m' for m and wins if and only if $m = m'$.

[1] We assume that all plaintext messages $m \in \mathcal{M}$ have the same length.

Fig. 1. OW-CPA: one-wayness of an attribute-based encryption scheme in the selective access structure model

Gen *is probabilistic and run by a trusted authority: on input the master secret key* mk *and a set of attributes* U *belonging to a user, a secret key* sk_U *for these attributes is generated.*

Sig *is probabilistic and run by a user who wants to sign a message* m *with his secret key* sk_U*: on input* $m \in \mathcal{M}$ *and* sk_U*, this algorithms generates a signature* σ*.*

Ver *is deterministic and run by a user who wants to verify if a signature has been created by a user with a set of attributes in the verification access structure* $\mathbb{B}$*: on input a message* m*, a signature* σ *and an access structure* $\mathbb{B} \subseteq 2^{\mathcal{U}}$*, this algorithm outputs* TRUE *if* σ *is a valid signature for* m *under* sk_U *for some* $U \in \mathbb{B}$*. Otherwise the algorithm outputs* FALSE*.*

Definition 9 (Existential unforgeability for attribute-based signing). *For a ppt adversary* $\mathcal{A}$*, denote by* $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{UF\text{-}CMAA}}$ *the probability that* $\mathcal{A}$ *wins the game described in Figure 2. An attribute-based signature scheme is* secure in the sense of UF-CMAA*, if the advantage* $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{UF\text{-}CMAA}} = \mathsf{Adv}_{\mathcal{A}}^{\mathsf{UF\text{-}CMAA}}(\ell)$ *is negligible for all ppt adversaries* $\mathcal{A}$*.*

Example 2. For access structures describing qualified subsets through a threshold, we can employ Shahandashti and Safavi-Naini's pairing-based construction in [19] to achieve security in the sense of Definition 9.

Given the above terminology, the following definition of an attribute-based signcryption scheme seems a natural one, and below we will argue that a generic way to obtain such a signcryption scheme is provided by an attribute-based variant of the encrypt-then-sign paradigm.

Definition 10 (Attribute-based signcryption). *An* attribute-based signcryption scheme *is a tuple* (Setup, Gen, Signcrypt, Unsigncrypt) *of polynomial time algorithms:*

Init phase Given the security parameter 1^ℓ, the adversary $\mathcal{A}$ outputs:
- a non-empty set $\mathcal{U}$, the universe of attributes.

Setup phase The challenger runs Setup and hands the public parameters to $\mathcal{A}$.

Query phase The adversary is allowed to ask (adaptively) queries for:
- private keys for attribute sets U;
- signatures of a signer with attribute set U on a message m.

Forgery phase The adversary outputs a tuple $(\mu, \sigma, \mathbb{A})$, where μ is a message and $\mathbb{A}$ is an access structure. The adversary wins if and only if $\mathsf{Ver}(\mu, \sigma, \mathbb{A}) = \text{TRUE}$ and the following restrictions on the queries in the challenge phase hold:
- all attribute sets U asked in private key queries satisfy $U \notin \mathbb{A}$;
- all inputs of signature queries satisfy $m \neq \mu$ or $U \notin \mathbb{A}$.

Fig. 2. UF-CMAA security: unforgeability under chosen message and attribute attacks

Setup *is probabilistic and run by a trusted authority: on input the security parameter 1^ℓ and a universe of attributes $\mathcal{U}$, a master secret key mk and public system parameters pm are generated. The public parameters include a description of the message space $\mathcal{M}$.*

Gen *is probabilistic and run by a trusted authority: on input the master secret key mk and a set of attributes U belonging to a user, a secret key ak_U for these attributes is generated.*

Signcrypt *is probabilistic and run by a user who wants to send a plaintext message m authenticated with his secret key ak_U for the set of attributes U to a user with a set of attributes in the access structure $\mathbb{A}$: on input $m \in \mathcal{M}$, ak_U and $\mathbb{A} \subseteq 2^{\mathcal{U}}$, this algorithm generates a signcryption s.*

Unsigncrypt *is deterministic and run by a user with a set of attributes U' and expecting a message that is authenticated with a set of attributes in the verification access structure $\mathbb{B}$: on input s, $ak_{U'}$ and $\mathbb{B}$, this algorithm outputs the underlying plaintext m, if s is a valid signcryption authenticated by some $U \in \mathbb{B}$ and such that U' is contained in the access structure $\mathbb{A}$ specified in the computation of s. Otherwise, an error symbol $\perp$ is returned.*

We impose the obvious correctness condition:

$$\mathsf{Unsigncrypt}(\mathsf{Signcrypt}(m, ak_U, \mathbb{A}), ak_{U'}, \mathbb{B}) = m$$

for all $U \in \mathbb{B}$ and $U' \in \mathbb{A}$.

Similarly as for ordinary signcryption, we consider two security requirements for attribute-based signcryption and formalize these requirements separately. The first security requirement refers to confidentiality:

Definition 11 (One-wayness for attribute-based signcryption). *For a ppt adversary $\mathcal{A}$, denote by $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{OWS\text{-}CPA}}$ the probability that $\mathcal{A}$ wins the game described in Figure 3. An attribute-based signcryption scheme is* OWS-CPA secure in the selective access structure model, *if $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{OWS\text{-}CPA}} = \mathsf{Adv}_{\mathcal{A}}^{\mathsf{OWS\text{-}CPA}}(\ell)$ is negligible for all ppt adversaries $\mathcal{A}$*

Init phase Given the security parameter 1^ℓ, the adversary $\mathcal{A}$ outputs:

- a non-empty set $\mathcal{U}$, the universe of attributes;
- a non-empty access structure $\mathbb{A} \subseteq 2^{\mathcal{U}}$ and an attribute set $U' \in 2^{\mathcal{U}}$ that it wants to be challenged upon.

Setup phase The challenger runs Setup and hands the public parameters to $\mathcal{A}$.

Query phase 1 The adversary is allowed to ask (adaptively) queries for

- private keys for attribute sets $U \subseteq \mathcal{U}$ subject to the restriction $U \notin \mathbb{A}$.
- signcryptions $s_m := \mathsf{Signcrypt}(m, ak_{U'}, \mathbb{A})$ with m being chosen uniformly at random by the challenger. Both m and s_m are returned to the adversary.[2]

Challenge phase The challenger picks uniformly at random a plaintext message m and signcrypts m using $ak_{U'}$ and $\mathbb{A}$.[2] The resulting signcryption $s := \mathsf{Signcrypt}(m, ak_{U'}, \mathbb{A})$ is handed to $\mathcal{A}$.

Query phase 2 Identical to Query phase 1.

Guess phase The adversary outputs a guess m' for the plaintext m underlying the signcryption s and wins if and only if $m = m'$.

[2] We assume that all plaintext messages $m \in \mathcal{M}$ have the same length.

Fig. 3. OWS-CPA: one-wayness of an attribute-based signcryption scheme in the selective access structure model

Similarly, we can capture the desired authenticity guarantee of an attribute-based signcryption scheme:

Definition 12 (Existential unforgeability for attribute-based signcryption). *For a ppt adversary $\mathcal{A}$, denote by $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{UFS\text{-}CMAA}}$ the probability that $\mathcal{A}$ wins the game described in Figure 4. An attribute-based signcryption scheme is* secure in the sense of $\mathsf{UFS\text{-}CMAA}$, *if the advantage $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{UFS\text{-}CMAA}} = \mathsf{Adv}_{\mathcal{A}}^{\mathsf{UFS\text{-}CMAA}}(\ell)$ is negligible for all ppt adversaries $\mathcal{A}$.*

Discussing the problem of dedicated constructions for attribute-based signcryption is outside the scope of this paper, but the following proposition gives a generic way to obtain a signcryption scheme as used in our protocol through a composition of suitable signature and encryption schemes. In particular, for a threshold setting we can build on the schemes of Sahai/Waters [18] and Shahandashti/Safavi-Naini [19].

Definition 13 (Attribute-based encrypt-then-sign).
Let $\mathcal{E} = (\mathsf{Setup}_{\mathcal{E}}, \mathsf{Gen}_{\mathcal{E}}, \mathsf{Enc}, \mathsf{Dec})$ be an attribute-based encryption scheme and $\mathcal{S} = (\mathsf{Setup}_{\mathcal{S}}, \mathsf{Gen}_{\mathcal{S}}, \mathsf{Sig}, \mathsf{Ver})$ be an attribute-based signature scheme. Then we define the encrypt-then-sign *($\mathcal{E}$t$\mathcal{S}$) signcryption scheme as follows:*

Setup *runs, on input the security parameter 1^ℓ and a universe of attributes $\mathcal{U}$, both $\mathsf{Setup}_{\mathcal{E}}(1^\ell,\mathcal{U})$ and $\mathsf{Setup}_{\mathcal{S}}(1^\ell,\mathcal{U})$, resulting in two key pairs $(mk_{\mathcal{E}}, pm_{\mathcal{E}})$ and $(mk_{\mathcal{S}}, pm_{\mathcal{S}})$. The returned master key is the pair $mk := (mk_{\mathcal{E}}, mk_{\mathcal{S}})$ and the public parameters are $pm := (pm_{\mathcal{E}}, pm_{\mathcal{S}})$.*

Init phase Given the security parameter 1^ℓ, the adversary $\mathcal{A}$ outputs:
- a non-empty set $\mathcal{U}$, the universe of attributes.

Setup phase The challenger runs Setup and hands the public parameters to $\mathcal{A}$.
Query phase The adversary is allowed to ask (adaptively) queries for:
- private keys for attribute sets U;
- signcryptions of a signer with attribute set U on a message m with an access structure $\mathbb{D}$

Forgery phase The adversary outputs a tuple $(\mu, s, U', \mathbb{A})$, where μ is a message, U' is a set of attributes, the secret key of which can be used to unsigncrypt and $\mathbb{A}$ is a verification access structure. The adversary wins if and only if s is a valid signcryption of μ that can be unsigncrypted with $ak_{U'}$ and $\mathbb{A}$, and the following restrictions on the queries in the challange phase hold:
- all attribute sets U asked in private key queries satisfy $U \notin \mathbb{A}$;
- all inputs of signcryption queries satisfy $m \neq \mu$ or $U \notin \mathbb{A}$.

Fig. 4. UFS-CMAA: existential unforgeability of an attribute-based signcryption scheme

Gen *runs, on input an attribute set* $U \in 2^{\mathcal{U}}$*, both* $\mathsf{Gen}_{\mathcal{E}}$ *and* $\mathsf{Gen}_{\mathcal{S}}$ *and combines the resulting secret keys* dk_U *and* sk_U *to the secret key* $ak_U := (dk_U, sk_U)$ *for the attribute set* U*.*

$\mathsf{Signcrypt}$ *receives a message* m*, a secret key* $ak_U = (dk_U, sk_U)$ *and an access structure* $\mathbb{A}$ *as input. The returned value is* $\mathsf{Signcrypt}(m, ak_U, \mathbb{A}) := (c, V, \mathsf{Sig}(c\|V, sk_U))$ *where* $c := \mathsf{Enc}(m, \mathbb{A})$ *and* $V \in \mathbb{A}$ *arbitrary.*

$\mathsf{Unsigncrypt}$ *receives a signcryption* (c, V, σ)*, a secret key* $ak_{U'} = (dk_{U'}, sk_{U'})$ *for an attribute set* U' *and a verification access structure* $\mathbb{B}$ *as input. The returned value is*

$$\mathsf{Unsigncrypt}(m, ak_{U'}, \mathbb{B}) := \begin{cases} \mathsf{Dec}(c, dk_{U'}) & , \textit{if } \mathsf{Ver}(c\|V, \sigma, \mathbb{B}) = \text{TRUE} \\ \bot & , \textit{otherwise} \end{cases} .$$

The following theorem says that $\mathcal{E}\mathrm{t}\mathcal{S}$ inherits security guarantees from the comprising component schemes.

Theorem 1. *Let* $\mathcal{S}$ *be an attribute-based signature scheme that is secure in the sense of* UF-CMAA*, and let* $\mathcal{E}$ *be an attribute-based encryption scheme that is secure in the sense of* OWS-CPA*. Then* $\mathcal{E}\mathrm{t}\mathcal{S}$ *is secure in the sense of both* OWS-CMAA *and* OWS-CPA*.*

Proof. We prove the two security guarantees for $\mathcal{E}\mathrm{t}\mathcal{S}$ separately.

UF-CMAA **security:** Let $\mathcal{A}'$ be a forger for the $\mathcal{E}t\mathcal{S}$ signcryption scheme. We use $\mathcal{A}'$ to construct a forger $\mathcal{A}$ for the signature scheme $\mathcal{S}$ such that $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{UF\text{-}CMAA}} = \mathsf{Adv}_{\mathcal{A}'}^{\mathsf{UFS\text{-}CMAA}}$. The public parameters $pm = (pm_{\mathcal{E}}, pm_{\mathcal{S}})$ for $\mathcal{A}'$ can be provided by $\mathcal{A}$ by using its own public parameters $pm_{\mathcal{S}}$ and by running $\mathsf{Setup}_{\mathcal{E}}$ to obtain $pm_{\mathcal{E}}$. Note that $\mathcal{A}$ also knows the master key $mk_{\mathcal{E}}$ corresponding to $pm_{\mathcal{E}}$. To reply to signcryption and key extraction queries, $\mathcal{A}$ can proceed as follows.

Private key queries To extract the secret key $ak_U = (dk_U, sk_U)$ for an attribute set U, $\mathcal{A}$ queries its own key extraction oracle to obtain sk_U and runs $\mathsf{Gen}_{\mathcal{E}}$ with input $mk_{\mathcal{E}}$ and U to obtain a decryption key dk_U.

Signcryption queries If $\mathcal{A}'$ queries for a signcryption on a message m with attribute set U and access structure $\mathbb{D}$, $\mathcal{A}$ computes the ciphertext $c := \mathsf{Enc}(m, \mathbb{D})$ and queries its signing oracle for a signature σ on $c\|V$ with attribute set U, where $V \in \mathbb{D}$ is chosen arbitrarily. Then (c, V, σ) is a valid reply to the signcryption query of $\mathcal{A}'$.

Suppose $\mathcal{A}'$ produces a successful forgery $(\mu, (c, V, \sigma), U', \mathbb{A})$ for $\mathcal{E}\mathrm{t}\mathcal{S}$, as specified in the UFS-CMAA game in Figure 4. Then $\mathcal{A}$ outputs the tuple $(c\|V, \sigma, \mathbb{A})$ as forgery for the signature scheme $\mathcal{S}$. We have to argue why this is indeed a forgery meeting the requirements of the UF-CMAA game in Figure 2:

- By definition of $\mathcal{E}\mathrm{t}\mathcal{S}$'s Unsigncrypt algorithm, we have $\mathsf{Ver}(c\|V, \sigma, \mathbb{A}) =$ TRUE.
- Private key queries: as $(\mu, (c, V, \sigma), U', \mathbb{A})$ is a successful forgery for $\mathcal{E}\mathrm{t}\mathcal{S}$, all queried attribute sets U are such that $U \notin \mathbb{A}$.
- Signature queries: for a valid forgery, all signcryption queries $(m, U, \mathbb{D})$ of $\mathcal{A}'$ satisfy $m \neq \mu$ or $U \notin \mathbb{A}$.

  $m \neq \mu$: suppose that $\mathcal{A}$ has submitted $c\|V$ to its signing oracle earlier. Then $c = \mathsf{Enc}(m, \mathbb{D})$ for some access structure $\mathbb{D}$ such that $V \in \mathbb{D}$. As Dec is deterministic, this implies $\mathsf{Dec}(c, dk_V) = m$ and c cannot be a valid encryption of $\mu \neq m$ under an access structure containing V. Consequently, $\mathcal{A}$ has never sent $c\|V$ to its signing oracle.

  $U \notin \mathbb{A}$: then the signature query $(c\|V, U)$ satisfies $U \notin \mathbb{A}$, and $\mathcal{A}$'s forgery is valid.

Summarizing, we have $\mathsf{Adv}^{\mathsf{UF\text{-}CMAA}}_{\mathcal{A}} = \mathsf{Adv}^{\mathsf{UFS\text{-}CMAA}}_{\mathcal{A}'}$ as desired.

OWS-CPA **security:** Let $\mathcal{A}'$ be an adversary in the OWS-CPA game for the $\mathcal{E}\mathrm{t}\mathcal{S}$. We use $\mathcal{A}'$ to construct an adversary $\mathcal{A}$ winning the OW-CPA game for the encryption scheme $\mathcal{E}$ with $\mathsf{Adv}^{\mathsf{OW\text{-}CPA}}_{\mathcal{A}'} = \mathsf{Adv}^{\mathsf{OWS\text{-}CPA}}_{\mathcal{A}'}$ is non-negligible.

For this, $\mathcal{A}$ outputs the same set of attributes $\mathcal{U}$ and the same access structure $\mathbb{A}$ as output by $\mathcal{A}'$ in the init phase. The public parameters $pm = (pm_{\mathcal{E}}, pm_{\mathcal{S}})$ for $\mathcal{A}'$ can be provided by $\mathcal{A}$ by using its own public parameters $pm_{\mathcal{E}}$ and by running $\mathsf{Setup}_{\mathcal{S}}$ to obtain $pm_{\mathcal{S}}$. Note that $\mathcal{A}$ also knows the master key $mk_{\mathcal{S}}$ corresponding to $pm_{\mathcal{S}}$. To reply to signcryption and key extraction queries, $\mathcal{A}$ can proceed as follows.

Private key queries To extract the secret key $ak_U = (dk_U, sk_U)$ for an attribute set U, $\mathcal{A}$ queries its own key extraction oracle to obtain dk_U and runs $\mathsf{Gen}_{\mathcal{S}}$ with input $mk_{\mathcal{S}}$ and U to obtain a signing key sk_U.

Signcryption queries Whenever $\mathcal{A}'$ requests a signcryption with attribute set U' and access structure $\mathbb{A}$, $\mathcal{A}$ computes the ciphertext $c := \mathsf{Enc}(m, \mathbb{A})$ with a uniformly at random chosen m, and in particular can return the plaintext m to $\mathcal{A}'$ as needed. The signcryption returned to $\mathcal{A}'$ is obtained as $(c, V, \mathsf{Sig}(c\|V, sk'_U))$ with $V \in \mathbb{A}$ arbitrary and U' being the identity specified by $\mathcal{A}'$ in the first part of the OWS-CPA game—$\mathcal{A}$ can compute sk'_U as $sk_{U'} = \mathsf{Gen}_{\mathcal{S}}(mk_{\mathcal{S}}, U')$.

In the challenge phase, $\mathcal{A}$ hands $(c, V, \mathsf{Sig}(c\|V, sk'_U))$ with $V \in \mathbb{A}$ arbitrary to $\mathcal{A}'$, where c is $\mathcal{A}$'s OW-CPA challenge ciphertext. The value returned by $\mathcal{A}$ is the plaintext returned by $\mathcal{A}'$. Obviously $\mathcal{A}$ wins the OW-CPA game if and only if $\mathcal{A}'$ returns the correct plaintext underlying $\mathcal{A}$'s OW-CPA challenge, and we have

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{OW\text{-}CPA}} = \mathsf{Adv}_{\mathcal{A}'}^{\mathsf{OWS\text{-}CPA}}.$$

$\square$

3.2 A two-round protocol

Given an attribute-based signcryption scheme (Setup, Gen, Signcrypt, Unsigncrypt) and a random oracle $H(\cdot)$, Figure 5 describes a two-round protocol for attribute-based key establishment. To simplify readability, we do not explicitly mention the instance number of protocol instances Π_U^s and refer, e g., to the session key simply as $\mathtt{sid}_U$ (instead of $\mathtt{sid}_U^s$).

Round 1:

Computation Each user U chooses $k_U \in \{0,1\}^\ell$ and $x_U \in \{1, \ldots, \mathrm{ord}(g)\}$ at random and computes $y_U := g^{x_U}$. In addition, the initiator U_{init} chooses $r \in \{0,1\}^\ell$ at random and computes $c := \mathsf{Signcrypt}(k_{U_{\mathsf{init}}}, ak_{U_{\mathsf{init}}}, \mathtt{pid}_{U_{\mathsf{init}}})$.

Broadcast Each U except U_{init} broadcasts $k_U\|y_U$. The initiator U_{init} broadcasts $s\|y_{U_{\mathsf{init}}}\|H(r)\|\mathtt{pid}_{U_{\mathsf{init}}}$.

Round 2:

Computation Each user U unsigncrypts c using the secret key ak_U and verification access structure $\mathtt{pid}_U$. If this yields the error symbol $\perp$ or $\mathtt{pid}_{U_{\mathsf{init}}} \not\subseteq \mathtt{pid}_U$ or $U \notin \mathtt{pid}_{U_{\mathsf{init}}}$, then U aborts.
Otherwise $k_{U_{\mathsf{init}}} := \mathsf{Unsigncrypt}(c, ak_U, \mathtt{pid}_U)$, and U orders the received $k_{U'}$-values, including $k_{U_{\mathsf{init}}}$, lexicographically[3]. Thus, U can index the $k_{U'}$s as $k_0 < \cdots < k_{n-1}$ and label users and y-values from Round 1 according to k_i as U_i and y_i. To simplify notation, we assume w. l. o. g. that $k_0 = k_{U_{\mathsf{init}}}$. Taking indices mod n, each U_i computes the values $t_i^{\mathrm{L}} := H(y_{i-1}^{x_i}\|k_0)$, $t_i^{\mathrm{R}} := H(y_{i+1}^{x_i}\|k_0)$ and $X_i := t_i^{\mathrm{L}} \oplus t_i^{\mathrm{R}}$. The initiator U_0 computes additionally $e := k_0 \oplus r \oplus t_0^{\mathrm{R}}$.

Broadcast Each U_i broadcasts (X_i, i) and U_0 broadcasts additionally e.

Check Each U_i checks if $X_0 \oplus \cdots \oplus X_{n-1} = 0$, obtains $t_0^{\mathrm{R}} = t_i^{\mathrm{L}} \oplus X_0 \oplus \bigoplus_{j=i}^{n-1} X_j$, computes r and checks if the commitment $H(r)$ from Round 1 is correct. If any check fails, the protocol is aborted.

Key derivation Each participant U_i computes the session key

$$\mathtt{sk}_{U_i} = H(r\|k_0\|k_1\| \cdots \|k_{n-1}\|\mathtt{pid}_{U_0}\|0)$$

and the session identifier $\mathtt{sid}_{U_i} = H(r\|k_0\|k_1\| \cdots \|k_{n-1}\|\mathtt{pid}_{U_0}\|1)$.

[3]If the k_i-values are not pairwise different, U aborts the protocol.

Fig. 5. Attribute based group key establishment in two rounds

It is worth noting that the computations performed by the protocol initiator deviate slightly from those performed by the other parties. In particular, the protocol initiator U_0 is the only party running the $\mathsf{Signcrypt}$ algorithm—all other protocol participants apply $\mathsf{Unsigncrypt}$ instead. The following result identifies the protocol as a secure attribute-based key establishment—provided the underlying attribute-based signcryption scheme offers appropriate guarantees and the Computational Diffie-Hellman (CDH) assumption holds.

Theorem 2. *Suppose that the CDH assumption holds for the group generated by g, $H(\cdot)$ is a random oracle, and the attribute-based signcryption scheme used in Figure 5 is secure in the sense of* OWS-CPA *and* UFS-CMAA*. Then the protocol in Figure 5 is a correct attribute-based key establishment that is secure in the sense of Definition 4 and fulfills integrity in the sense of Definition 5.*

Proof. Correctness is obvious, and we can restrict to showing security and integrity. For this, let q_s and q_ro be polynomial upper bounds for the number of the adversary $\mathcal{A}$'s queries to the `Send` respectively the random oracle. We begin by defining four events that occur throughout the proof, and we give negligible upper bounds for the probabilities of these events to occur.

Collision is the event that the random oracle produces a collision. A `Send` query causes at most 5 random oracle calls. Thus, the total number of random oracle queries is bounded by $5q_\mathrm{s} + q_\mathrm{ro}$ and the probability that a collision of the random oracle occurs is

$$P(\mathsf{Collision}) \leq \frac{(5q_\mathrm{s} + q_\mathrm{ro})^2}{2^\ell},$$

which is negligible in ℓ.

Decrypt is the event that the adversary $\mathcal{A}$ succeeds in recovering a random message k_{U_init} from a signcryption c with secret key ak_{U_init} and access structure $\mathtt{pid}_{U_\mathsf{init}}$, without having queried $\mathsf{Corrupt}(U)$ for any $U \in \mathtt{pid}_{U_\mathsf{init}}$ and without having queried Reveal for the respective instance of U_init. An adversary $\mathcal{A}$ that can reach Decrypt can be used to construct an adversary $\mathcal{C}$ violating the OWS-CPA security of the signcryption scheme: $\mathcal{C}$ guesses the access structure $\mathtt{pid}_{U_\mathsf{init}}$, the attribute set U_init as well as the respective instance of U_init uniformly at random. As $\mathcal{U}$ has constant size, this guess is correct with probability $\geq 1/p$ for some polynomial $p = p(\ell)$. If any of the guessed values is incorrect, then $\mathcal{C}$ aborts. In case of everything being guessed correctly, $\mathtt{pid}_{U_\mathsf{init}}$ and U_init form the access structure and the set of attributes that $\mathcal{C}$ has to specify in the Init phase of the OWS-CPA game. All of $\mathcal{A}$'s oracle queries can be simulated in the obvious way by $\mathcal{C}$, and we obtain

$$\mathsf{Adv}_{\mathcal{C}}^{\mathsf{OWS\text{-}CPA}} \geq \frac{1}{p} \cdot P(\mathsf{Decrypt}).$$

Thus, the event Decrypt occurs with negligible probability only.

Forge is the event that $\mathcal{A}$ succeeds in forging a signcryption c of a message $k_{U_{\mathsf{init}}}$ for attribute set U_{init} and access structure $\mathtt{pid}_{U_{\mathsf{init}}}$ without having queried $\mathsf{Corrupt}(U_{\mathsf{init}})$ and where $k_{U_{\mathsf{init}}}$ was not output by any of U_{init}'s instances. An adversary $\mathcal{A}$ that can reach Forge can be used for forging a signcryption: the tuple $(k_{U_{\mathsf{init}}}, s, U_{\mathsf{init}}, \{U_{\mathsf{init}}\})$ would constitute a valid forgery, since $s = \mathsf{Signcrypt}(k_{U_{\mathsf{init}}}, ak_{U_{\mathsf{init}}}, \mathtt{pid}_{U_{\mathsf{init}}})$, so it can be unsigncrypted successfully with with the secret key of U_{init} and the verification access structure $\{U_{\mathsf{init}}\}$. Moreover, there has not been any private key query of U_{init} (no $\mathsf{Corrupt}(U_{\mathsf{init}})$) nor a signcryption query of $(k_{U_{\mathsf{init}}}, U_{\mathsf{init}}, \mathtt{pid}_{U_{\mathsf{init}}})$ ($k_{U_{\mathsf{init}}}$ was not output by any of U_{init}'s instances). Thus, using $\mathcal{A}$ as a black box we obtain an attacker $\mathcal{B}$ defeating the existential unforgeability of the underlying signcryption scheme with advantage

$$\mathsf{Adv}_{\mathcal{B}}^{\mathsf{UFS\text{-}CMAA}} \geq P(\mathsf{Forge}).$$

By assumption $\mathsf{Adv}_{\mathcal{B}}^{\mathsf{UFS\text{-}CMAA}}$ is negligible, and we see that Forge occurs with negligible probability only.

Repeat is the event that an uncorrupted participant chooses a nonce k_i or r that was previously used by an oracle of some party. There are at most q_{s} used instances that may have chosen a nonce k_i or r, and thus the event Repeat occurs with probability

$$P(\mathsf{Repeat}) \leq 4 \cdot \frac{q_{\mathrm{s}}^2}{2^{\ell}},$$

which again is negligible in ℓ.

$\mathsf{TestCorrupt}$ is the event that a participant U_i of a $\mathtt{Test}$ session with fresh instances has been corrupted, and U_i accepted the session key. According to the definition of freshness, U_i was not corrupted yet, when sending its Round 2 message (X_i, i) to the other protocol participants. Consequently, X_i was, with overwhelming probability, computed without knowledge of t_i^{L} and t_i^{R}—for computing the latter either the event $\mathsf{Collision}$ or $\mathsf{Decrypt}$ had to occur. As a consequence the r-value r' recovered by U_i satisfies $H(r') = H(r)$ with negligible probability only. Therefore, with overwhelming probability, U_i aborted the protocol without accepting the session key, and we recognize $P(\mathsf{TestCorrupt})$ as negligible.

Security. To prove security according to Definition 4, we use the usual game hopping technique, letting the adversary $\mathcal{A}$ interact with a simulator. In Game 0, the simulator offers the original protocol environment to $\mathcal{A}$, but subsequently we change the simulator's behavior in several small steps without affecting $\mathcal{A}$'s success probability significantly. Keeping track of the changes between subsequent games, in the last game we will be able to establish a negligible upper bound on $\mathsf{Adv}_{\mathcal{A}}$. We denote the advantange of $\mathcal{A}$ in Game i by $\mathsf{Adv}_{\mathcal{A}}^{\text{Game } i}$.

Game 0: In this game, the simulator faithfully simulates all protocol participants' instances for the adversary $\mathcal{A}$, i. e., the adversary's situation is the same as in the real model:

$$\mathsf{Adv}_{\mathcal{A}}^{\text{Game } 0} = \mathsf{Adv}_{\mathcal{A}}.$$

Game 1: This game is aborted if one of the events Forge, $\mathsf{Collision}$, Repeat or $\mathsf{TestCorrupt}$ occurs. Otherwise the game is identical with Game 0 and the adversary cannot detect the difference. Thus, for adversary $\mathcal{A}$'s advantage we have

$$|\mathsf{Adv}_{\mathcal{A}}^{\text{Game 1}} - \mathsf{Adv}_{\mathcal{A}}^{\text{Game 0}}| \leq P(\mathsf{Forge}) + P(\mathsf{Collision}) + P(\mathsf{Repeat}) + P(\mathsf{TestCorrupt}).$$

Game 2: This game differs from Game 1 in the simulator's response in Round 2. If the simulator has to output the message of an instance $\Pi_{U_i}^{s_i}$ and none of the participants $U_j \in \mathtt{pid}_{U_i}^{s_i}$ is corrupted, then the simulator chooses random values from $\{0,1\}^\ell$ for $t_i^{\mathrm{L}} = t_{i-1}^{\mathrm{R}}$ and $t_i^{\mathrm{R}} = t_{i+1}^{\mathrm{L}}$ instead of querying the random oracle. To keep consistency, the same values have to be used in the neighbored instances subsequently. The adversary can only detect the difference by querying the random oracle with $y_{i-1}^{x_i} \| k_0 = y_i^{x_{i-1}} \| k_0$.
An adversary $\mathcal{A}$ that distinguishes Game 1 and Game 2 can be used as black box to solve a CDH instance: two instances $\Pi_{U_i}^{s_i}$ and $\Pi_{U_j}^{s_j}$ are selected by randomly choosing two different users $U_i, U_j \in 2^{\mathcal{U}}$ plus two numbers $s_i, s_j \in \{1, \ldots, q_{\mathrm{s}}\}$. Game 2 only differs from Game 1, if at least one session is set up of uncorrupted users. To distinguish the games, the adversary has to query the random oracle with at least one Diffie-Hellman key, established between neighbors in a session with uncorrupted participants. These randomly chosen instances will be those neighbored participants with probability at least $1/(2^{|\mathcal{U}|} \cdot q_{\mathrm{s}})^2$.
A given CDH instance (g, g^a, g^b) is then assigned to $\Pi_{U_i}^{s_i}$ and $\Pi_{U_j}^{s_j}$ such that these instances will use $y_i := g^a$ respectively $y_j := g^b$ in Round 1.
If at some point now $\Pi_{U_i}^{s_i}$ and $\Pi_{U_j}^{s_j}$ do not qualify any longer to be neighbored participants in a session with only uncorrupted users, the simulation is aborted.
Then a random index $z \in \{1, \ldots, q_{\mathrm{ro}}\}$ is chosen and the adversary's z-th query to the random oracle is taken for the answer to the CDH challenge. The answer to the CDH challenge is correct if $\mathcal{A}$ distinguished the games with the chosen instances and also the index z was guessed correctly:

$$|\mathsf{Adv}_{\mathcal{A}}^{\text{Game 2}} - \mathsf{Adv}_{\mathcal{A}}^{\text{Game 1}}| \leq \mathsf{Succ}_{(\langle g \rangle, g)}^{\text{CDH}} \cdot q_{\mathrm{ro}} \cdot \left(2^{|\mathcal{U}|} \cdot q_{\mathrm{s}}\right)^2,$$

where $\mathsf{Succ}_{(\langle g \rangle, g)}$ is an upper bound for the success probability of the above algorithm to solve the CDH problem in group generated by g, using generator g. In particular, under the CDH assumption and with $\mathcal{U}$ having constant size, the right-hand side of this inequality is negligible in ℓ.

Game 3: In this game the simulator changes the computation of the session key: having received all messages of Round 2 for an instance $\Pi_{U_i}^{s_i}$, the simulator checks if all $U_j \in \mathtt{pid}_{U_i}^{s_i}$ are uncorrupted and if $\mathtt{Reveal}$ has not been queried with an instance $\Pi_{U_j}^{s_j} \in \mathtt{pid}_{U_i}^{s_i}$. If this is the case the simulator chooses a session key $\mathtt{sk}_{U_i}^{s_i} \in \{0,1\}^\ell$ uniformly at random instead of querying the

random oracle. For consistency, the simulator will assign the same key to all partnered instances. To detect the difference to the previous game, the adversary must query the random oracle for $H(r\|k_0\|\ldots\|k_n\|1)$.
About r only $H(r)$ and $e = k_0 \oplus r \oplus t_0^R$ are known. Thus, the adversary can only guess a random value for r and query the random oracle at most q_{ro} times, or can get the value r if it can invert the signcryption c to get k_0 and can get t_0^R. This results in:

$$|\mathsf{Adv}_{\mathcal{A}}^{\text{Game 3}} - \mathsf{Adv}_{\mathcal{A}}^{\text{Game 2}}| \leq \frac{q_{\mathrm{ro}}}{2^\ell} + P(\mathsf{Decrypt})$$

All participants involved in the `Test` session are uncorrupted, and none the instances involved in the `Test` query have been revaled. Therfore, those instances are affected by the modification just introduced, i. e., they use random session keys. Consequently $\mathsf{Adv}_{\mathcal{A}}^{\text{Game 3}} = 0$.
Putting the probabilities together we recognize the adversary's advantage in the real model as negligible:

$$\mathsf{Adv}_{\mathcal{A}} \leq P(\mathsf{Forge}) + P(\mathsf{Collision}) + P(\mathsf{Repeat}) + \mathsf{Succ}_{(\mathbb{G},g)}^{\mathrm{CDH}} \cdot q_{\mathrm{ro}} \cdot (2^{\mathcal{U}})^2 \cdot q_{\mathrm{s}}^2 + \frac{q_{\mathrm{ro}}}{2^\ell} + P(\mathsf{Decrypt}) + P(\mathsf{TestCorrupt}).$$

Integrity. Let U_i and U_j be any two honest principals whose instances $\Pi_i^{s_i}$ and $\Pi_{U_j}^{s_j}$ accept ($\mathtt{acc}_{U_i}^{s_i} = \mathtt{acc}_{U_i}^{s_i} = \text{TRUE}$) with a matching session identifier $\mathtt{sid}_{U_i}^{s_i} = \mathtt{sid}_{U_j}^{s_j}$.Then with overwhelming probability $r\|k_0\|\ldots\|k_{n-1}\|\mathtt{pid}_{U_0}$ is identical for both users and therewith $\mathtt{sk}_{U_i}^{s_i} = \mathtt{sk}_{U_j}^{s_j}$. In particular, $\Pi_{U_i}^{s_i}$ and $\Pi_{U_j}^{s_j}$ have with overwhelming probability the same value $\mathtt{pid}_{U_0}$. As the tests in Round 2 succeeded, we see that $U_i \in \mathtt{pid}_{U_0}$ and $U_j \in \mathtt{pid}_{U_0}$. Moreover, we have $\mathtt{pid}_{U_0} \subseteq \mathtt{pid}_{U_i} \cap \mathtt{pid}_{U_j}$. Thus $U_i \in \mathtt{pid}_{U_j}$ and $U_j \in \mathtt{pid}_{U_i}$ with overwhelming probability. □

4 Further protocol properties

The protocol in the previous section has a number of characteristics, that seem to be worth commenting. We do not formalize these properties here, and consequently these comments should not be taken as provable guarantees, but rather as issues that might deserve further (formal) exploration in future work:

Key agreement. The protocol is contributory in the sense that each party influences the value of the final session key by its input, and no proper subset of protocol participants can enforce a particular predetermined session key: parties U other than the initiator U_{init} have to publish their contribution k_i before learning the random value r, i. e., parties U can actually not fix any bit in the session key. The initiator U_{init} can mount a rushing attack, however: before fixing r, U_{init} knows all inputs to the key derivation. Because of the application of the random oracle in the derivation of the session key, U_{init}'s potential to manipulate the value of the session key is still rather limited and reduces to quering the random oracle with different r-values. If a stronger guarantee is desired, the following approach (see [16, 4]) seems worth being explored: in Round 1, users $U \neq U_{\mathsf{init}}$ broadcast $H(k_i)$ instead of k_i—and these hash value then form the basis to fix an ordering among protocol participants. The actual values $k_1, \ldots, k_{n-1}$ would then be included in the Round 2 messages and checked for consistency with the Round 1 commitments.

Plausible deniability. Protocol transcripts generated by initiator U_{init} alone are indistinguishable from real protocol transcripts: even after revealing all secret keys, including the master keys, U_{init} cannot provide evidence of any other parties' active involvement in a protocol execution, as secret user keys ak_U with $U \neq U_{\mathsf{init}}$ are only used to recover values signcrypted by U_{init}.

Privacy. As just noted, parties $U \neq U_{\mathsf{init}}$ use their secret keys only for recovering values signcrypted by the initiator U_{init}. At no point in the protocol do those parties have to make their attributes explicit; only the fact that U is contained in the access structure $\mathtt{pid}_{U_{\mathsf{init}}}$ used to create the signcryption in Round 1 has to be revealed.

5 Conclusion

In this paper we discussed a notion of attribute-based key establishment and provided a two-round solution, building on an attribute-based signcryption scheme offering a basic form of security. The discussion of attribute-based signcryption might be of independent interest, and, as shown, such a signcryption scheme can be derived from suitable attribute-based signature and encryption schemes, using the encrypt-then-sign paradigm. We think that our discussion raises a number of questions that deserve follow-up work—like the question of dedicated constructions for attribute-based signcryption schemes or the use of a form of attribute-based key-encapsulation with the proposed protocol.

Acknowledgments

The second author acknowledges support of FICYT (project IB-08-147) and Spanish MEC (project MEC-07-MTM2007-67884-C04-01 and FPU grant AP2007-03141, cofinanced by the European Social Fund). Also, we thank the anonymous referees for their comments.

References

1. Jee Hea An, Yevgeniy Dodis, and Tal Rabin. On the Security of Joint Signature and Encryption. In Lars R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 83–107. Springer, 2002.

2. James Birkett and Douglas Stebila. Predicate-Based Key Exchange. Cryptology ePrint Archive, Report 2010/082, February 2010. Available at `http://eprint.iacr.org/2010/082/`.
3. Jens-Matthias Bohli and Rainer Steinwandt. Deniable Group Key Agreement. In Phong Q. Nguyen, editor, *Progress in Cryptology – VIETCRYPT 2006*, volume 4341 of *Lecture Notes in Computer Science*, pages 298–311. Springer, 2006.
4. Jens-Matthias Bohli, María Isabel González Vasco, and Rainer Steinwandt. Secure group key establishment revisited. *International Journal of Information Security*, 6(4):243–254, 2007.
5. Dan Boneh and Brent Waters. Conjunctive, Subset, and Range Queries on Encrypted Data. In Salil P. Vadhan, editor, *Theory of Cryptography – TCC 2007,*, volume 4392 of *Lecture Notes in Computer Science*, pages 535–554. Springer, 2007.
6. Colin Boyd and Anish Mathuria. *Protocols for Authentication and Key Establishment.* Information security and cryptography. Springer, 2003.
7. Emmanuel Bresson, Olivier Chevassut, David Pointcheval, and Jean-Jacques Quisquater. Provably Authenticated Group Diffie-Hellman Key Exchange. In Pierangela Samarati, editor, *Proceedings of the 8th ACM Conference on Computer and Communications Security (CCS-8)*, pages 255–264. ACM, 2001.
8. Mike Burmester and Yvo Desmedt. A secure and scalable Group Key Exchange system. *Information Processing Letters*, 94:137–143, 2005.
9. Jan Camenisch, Nathalie Casati, Thomas Gross, and Victor Shoup. Credential Authenticated Identification and Key Exchange. Cryptology ePrint Archive: Report 2010/055, February 2010. Available at `http://eprint.iacr.org/2010/055/`.
10. Ran Canetti. Universally composable security: A new paradign for cryptographic protocols. Cryptology ePrint Archive, Report 2000/067, December 2005. Available at `http://eprint.iacr.org/2000/067/`.
11. Liming Fang, Jiandong Wang, Yongjun Ren, Jinyue Xia, and Shizhu Bian. Chosen-Ciphertext Secure Multi-authority Fuzzy Identity-Based Key Encapsulation without ROM. In *Proceedings of the 2008 International Conference on Computational Intelligence and Security – Volume 01*, pages 326–330. IEEE Computer Society, 2008.
12. Malakondayya Choudary Gorantla, Colin Boyd, and Juan Manuel González Nieto. Attribute-based Authenticated Key Exchange. Cryptology ePrint Archive: Report 2010/084, February 2010.
13. Jonathan Katz, Amit Sahai, and Brent Waters. Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products. In Nigel P. Smart, editor, *Advances in cryptology – EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 146–162. Springer, 2008.
14. Jonathan Katz and Moti Yung. Scalable Protocols for Authenticated Group Key Exchange. In Dan Boneh, editor, *Advances in Cryptology – CRYPTO'03*, volume 2729 of *Lecture Notes in Computer Science*, pages 110–125. Springer, 2003.
15. Fagen Li, Masaaki Shirase, and Tsuyoshi Takagi. Identity-Based Hybrid Signcryption. In *International Conference on Availability, Reliability and Security 2009, ARES' 09*, pages 534–539. IEEE Computer Society, 2009.
16. Chris J. Mitchell, Mike Ward, and Piers Wilson. Key control in key agreement protocols. *IEE Electronics Letters*, 34(10):980–981, 1998.
17. Takashi Nishide, Kazuki Yoneyama, and Kazuo Ohta. Attribute-Based Encryption with Partially Hidden Encryptor-Specified Access Structures. In Steven M. Bellovin, Rosario Gennaro, Angelos Keromytis, and MotiYung, editors, *Applied Cryptography and Network Security, 6th International Conference, ACNS 2008*, volume 5037 of *Lecture Notes in Computer Science*, pages 111–129. Springer, 2008.

18. Amit Sahai and Brent Waters. Fuzzy Identity-Based Encryption. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 457–473. Springer, 2005.
19. Siamak F. Shahandashti and Reihaneh Safavi-Naini. Threshold Attribute-Based Signatures and Their Application to Anonymous Credential Systems. In Bart Preneel, editor, *Progress in Cryptology – AFRICACRYPT 2009*, volume 5580 of *Lecture Notes in Computer Science*, pages 198–216. Springer, 2009. Full version available as Cryptology ePrint Archive Report 2009/126, `http://eprint.iacr.org/2009/126/`.
20. Hao Wang, Qiu-Liang Xu, and Xiu Fu. Revocable Attribute-based Key Agreement Protocol without Random Oracles. *Journal of Networks*, 4(8):787–794, October 2009.
21. Hao Wang, Qiuliang Xu, and Tao Ban. A Provably Secure Two-Party Attribute-Based Key Agreement Protocol. In *2009 Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, pages 1042–1045, 2009.
22. Hao Wang, QiuLiang Xu, and Xiu Fu. Two-Party Attribute-based Key Agreement Protocol in the Standard Model. In *Proceedings of the 2009 International Symposium on Information Processing (ISIP'09)*, pages 325–328, 2009.