Adaptive Soundness of Static Equivalence[⋆]

Steve Kremer and Laurent Mazaré

LSV, ENS Cachan & CNRS & INRIA Futurs
{kremer|mazare}@lsv.ens-cachan.fr

Abstract. We define a framework to reason about implementations of equational theories in the presence of an adaptive adversary. We particularly focus on soundess of static equivalence. We illustrate our framework on several equational theories: symmetric encryption, XOR, modular exponentiation and also joint theories of encryption and modular exponentiation. This last example relies on a combination result for reusing proofs for the separate theories. Finally, we define a model for symbolic analysis of dynamic group key exchange protocols, and show its computational soundness.

1 Introduction

It is well-known that even simple security protocols are extremely error-prone. This is mainly due to the fact that they are executed in a hostile environment, *e.g.*, the Internet. The need for rigorous proofs was recognized very early and two distinct, competing approaches have been developed. The symbolic approach considers an abstract model, where messages and cryptographic primitives are modeled by a term algebra. The adversary manipulates terms according to a pre-defined set of rules, typically an inference system. The computational approach considers a more detailed execution model. Protocol messages are modeled as bitstrings and cryptographic primitives are polynomial-time algorithms. The adversary is an arbitrary probabilistic polynomial-time Turing machine. Security of a protocol is measured as the adversary's success probability.

Proofs in the symbolic model can be (partially) automated, but it is not clear whether this abstract model captures all possible attacks. Proofs in the computational model provide stronger security guarantees but are generally harder and difficult to automate. A recent trend tries to get the best of both worlds: an abstract model with strong computational guarantees. In their seminal paper, Abadi and Rogaway [4] have shown a first such *soundness result* for symmetric encryption in the presence of a passive attacker.

Recently, Baudet *et al.* [9] presented a general framework for reasoning about sound implementations of equational theories. Instead of a fixed set of cryptographic primitives, they allow a specification by the means of an equational theory. The formal indistinguishability relation they consider is static equivalence, a well-established security notion coming from cryptographic pi calculi [3] whose verification can often be automated [2, 10]. Studying soundness of equational theories is motivated by the numerous recent works on extending the classical Dolev-Yao result with equations which are intended to capture algebraic properties of cryptographic primitives (see [17] for a survey

[⋆] Work partly supported by ARA SSIA Formacrypt and ARTIST2 Network of Excellence.

). Showing a soundness result for an equational theory proves that indeed “enough” equations have been considered in the symbolic model.

In this paper we consider the question of soundness of static equivalence in the presence of an *adaptive adversary*, rather than a purely passive one. This extends the work by Baudet *et al.* in a similar way as the work of Micciancio and Panjwani [25] extended the work of Abadi and Rogaway [4]. An adaptive adversary is allowed to choose the messages whose implementation he will be given. The choice of the messages can hence depend on previously observed distributions. We illustrate the usefullness of such a model on dynamic group key exchange protocols.

More precisely the contributions of our paper are as follows. We define the notion of adaptive soundness of static equivalence in a general framework. The definition is parameterized by the equational theory and the concrete algebra implementing the symbolic model. Our notion is strictly stronger than the purely passive soundness from [9]. We also develop a combination based proof technique: it allows us to reuse soundness results of two disjoint abstract signatures and conclude soundness of the joint signature. While the conditions under which such a combination works are of course restrictive they nevertheless match cases of practical interest. We give adaptive soundness results for several theories: symmetric encryption provided that the encryption scheme respects a length-conceiling IND-CPA security notion (this is similar to the main result in [25]), exclusive or (XOR), modular exponentiation in an Abelian group provided that the *Decisional Diffie-Hellman* (DDH) assumption is verified. Finally, we use our combination technique to derive adaptive soundness for the joint theory of encryption and modular exponentiation. We believe these are the first adaptive soundness results for modular exponentiation. Their importance is motivated by real-life protocols such as SSL/TLS that rely on Diffie-Hellman key exchange and thus use modular exponentiation.

To illustrate the usefulness of adaptive adversaries we define a symbolic model for dynamic group key exchange (DKE) protocols. A DKE protocol is a suite of protocols which allows three actions: exchange of an initial key between a group of users, joining and leaving the group. A typical example of DKE is the AKE1 protocol [12]. In our symbolic model we assume static corruption, as it was the case in [25], and allow the adversary to schedule these subprotocol and decide which users initially exchange the key, join, respectively leave the group. We use our adaptive soundness result to show that this symbolic model is sound with respect to a corresponding computational model.

Related work. As discussed above this paper generalizes both work by Baudet *et al.* [9] and Micciancio and Panjwani [25]. Abadi *et al.* [1] also use the framework of [9] to show soundness of an equational theory useful for reasoning about offline guessing attacks modeled in terms of static equivalence. In [8], Bana *et al.* argue that the notion of static equivalence is too coarse and not sound for many interesting equational theories. As an example they show that the DDH assumption is not sufficient to imply soundness of static equivalence. They introduce a general notion of *formal indistinguishability relation*. In this paper we prefer to stick to static equivalence which has the advantage of being a well-established, tool-supported equivalence relation. We address the problems highlighted in [8] by proving soundness for a restricted set of *well-formed* frames (in the same vein Abadi and Rogaway used restrictions to forbid key cycles). Regarding the theory of XOR, Backes and Pfitzmann [6] have shown an impossibility result in the reactive simulatability framework with active attackers and a soundness result for passive attackers. Note that we use the same model as [9] and restrict ourselves to the XOR of pure random values and not arbitrary payloads. While this is a restriction, it may nevertheless be useful for computing keys as the XOR of two random values when combining XOR and encryption.

There have also been numerous works considering an active adversary using approaches. Without being exhaustive this work includes reactive simulatability providing universally composable results in [7, 15], soundness results (but not universal composability) for an automated tool are presented in [18], cryptographically sound type systems [24], a *Protocol Composition Logic* in [19] and an automatic tool that aims at directly generating cryptographic proofs via sequences of games in [11]. However these works stick to classical cryptographic primitives: digital signatures, symmetric and asymmetric encryption. We are not aware of any general results for equational theories in the active case. Considering an active adversary is technically more involved although incomparable to an adaptive adversary. The case of a both active and adaptive adversary is a challenging problem and a topic of active research.

Because of lack of space, proofs are omitted. They are available in [23].

2 Abstract and computational algebras

We introduce our model, which is the same up to some minor changes as in [9].

2.1 Abstract algebras

Our abstract models—called *abstract algebras*—consist of term algebras defined over a many-sorted first-order signature and equipped with equational theories.

Specifically, a *signature* $(\mathcal{S}, \mathcal{F})$ is made of a set of *sorts* $\mathcal{S} = \{s, s_1 \ldots\}$ and a set of *symbols* $\mathcal{F} = \{f, f_1 \ldots\}$ together with arities of the form $\mathrm{ar}(f) = s_1 \times \ldots \times s_k \to s$, $k \geq 0$. Symbols that take $k = 0$ arguments are called *constants*; their arity is simply written s. We fix a set of *names* $\mathcal{N} = \{a, b \ldots\}$ and a set of *variables* $\mathcal{X} = \{x, y \ldots\}$. Names and variables are given with sorts and an infinite number of names and variables are available for each sort. The set of *terms of sort* s is defined inductively by

$$\begin{array}{lll} t ::= & & \text{term of sort } s \\ & \mid\ x & \text{variable } x \text{ of sort s} \\ & \mid\ a & \text{name } a \text{ of sort s} \\ & \mid\ f(t_1, \ldots, t_k) & \text{application of symbol } f \in \mathcal{F} \end{array}$$

where for the last case, we further require that t_i is of sort s_i and $\mathrm{ar}(f) = s_1 \times \ldots \times s_k \to s$. We also allow subsorts: if s_2 is a subsort of s_1 we allow a term of sort s_2 whenever a term of sort s_1 is expected. We write $sort(t)$ for the sort of term t. We define $root(t)$ to be f if $t = f(t_1, \ldots, t_n)$ and t otherwise, *i.e.* if t is either a name or a variable. We write $\mathrm{var}(t)$ and $\mathrm{names}(t)$ for the set of variables and names occurring in t, respectively. A term t is *ground* or *closed* if $\mathrm{var}(t) = \emptyset$.

Substitutions are written $\sigma = \{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$ with domain $\mathrm{dom}(\sigma) = \{x_1, \ldots, x_n\}$. We only consider *well-sorted, cycle-free* substitutions. Such a σ is *closed* if all of the t_i are closed. We let $\mathrm{var}(\sigma) = \bigcup_i \mathrm{var}(t_i)$, $\mathrm{names}(\sigma) = \bigcup_i \mathrm{names}(t_i)$, and extend the notations $\mathrm{var}(.)$ and $\mathrm{names}(.)$ to tuples and sets of terms and substitutions in the obvious way. The application of a substitution σ to a term t is written $\sigma(t) = t\sigma$ and is defined in the usual way. As usual the set of positions $pos(t)$ of a term t is defined inductively as $pos(c) = pos(a) = pos(x) = \{\epsilon\}$ where $\mathrm{ar}(c) = s$ and $pos(f(t_1, \ldots, t_n)) = \{\epsilon\} \cup \bigcup_{1 \leq i \leq n} i \cdot pos(t_i)$. If p is a position of t then expression $t|_p$ denotes the subterm of t at the position p, *i.e.*, $t|_\epsilon = t$ and $f(t_1, \ldots, t_n)|_{i \cdot p} = t_i|_p$.

Symbols in $\mathcal{F}$ are intended to model cryptographic primitives, whereas names in $\mathcal{N}$ are used to model secrets, *e.g.*, keys. The abstract semantics of symbols is described by an equational theory E, *i.e*, an equivalence relation (also written $=_E$) which is stable by application of contexts and substitutions of variables. For instance, symmetric encryption can be modeled by the classical theory $E_{\mathsf{enc}} = \{\mathrm{dec}(\mathrm{enc}(x, y), y) = x\}$.

2.2 Deducibility and static equivalence

We use frames [3, 2] to represent sequences of messages observed by an attacker. Formally, a *frame* is an expression $\varphi = \nu\tilde{a}.\{x_1 = t_1, \ldots, x_n = t_n\}$ where $\tilde{a}$ is a set of *bound (or restricted) names*, and for each i, t_i is a closed term of the same sort as x_i. For simplicity, we only consider frames $\varphi = \nu\tilde{a}.\{x_1 = t_1, \ldots, x_n = t_n\}$ which restrict every name in use, *i.e.*, $\tilde{a} = \mathrm{names}(t_1, \ldots, t_n)$. A name a may still be disclosed explicitly by adding a mapping $x_a = a$ to the frame. Thus we assimilate such frames φ to their *underlying substitutions* $\sigma = \{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$ also denoted $\{x_i \mapsto t_i\}_{1 \leq i \leq n}$.

Definition 1 (Deducibility). *A (closed) term t is* deducible *from a frame φ in an equational theory E, written $\varphi \vdash_E t$, iff there exists a term M such that* $\mathrm{var}(M) \subseteq \mathrm{dom}(\varphi)$, $\mathrm{names}(M) \cap \mathrm{names}(\varphi) = \emptyset$, *and $M\varphi =_E t$.*

For simplicity we only consider deducibility problems $\varphi \vdash_E t$ such that $\mathrm{names}(t) \subseteq \mathrm{names}(\varphi)$. For instance, let $\varphi_1 = \{x_1 \mapsto \mathrm{enc}(k_1, k_2), x_2 \mapsto \mathrm{enc}(k_4, k_3),\ x_3 \mapsto k_3\}$: under the the theory E_{enc} name k_4 is deducible from φ_1 since $\mathrm{dec}(x_2, x_3)\varphi_1 =_{E_{\mathsf{enc}}} k_4$ but neither are k_1 nor k_2. As also argued in [2] deducibility is not always sufficient to account for the knowledge of an attacker. For instance, it lacks partial information on secrets. That is why another classical notion in formal methods is *static equivalence*.

Definition 2 (Static equivalence). *Two frames φ_1 and φ_2 are* statically equivalent *in a theory E, written $\varphi_1 \approx_E \varphi_2$, iff* $\mathrm{dom}(\varphi_1) = \mathrm{dom}(\varphi_2)$, *and for all terms M and N such that* $\mathrm{var}(M, N) \subseteq \mathrm{dom}(\varphi_1)$ *and* $\mathrm{names}(M, N) \cap \mathrm{names}(\varphi_1, \varphi_2) = \emptyset$, $M\varphi_1 =_E N\varphi_1$ *is equivalent to $M\varphi_2 =_E N\varphi_2$.*

For instance, let 0 and 1 be two constants (which are known by the attacker). Then $\{x \mapsto \mathrm{enc}(0, k)\} \approx_{E_{\mathsf{enc}}} \{x \mapsto \mathrm{enc}(1, k)\}$. However $\varphi = \{x \mapsto \mathrm{enc}(0, k),\ y \mapsto k\}$ and $\varphi' = \{x \mapsto \mathrm{enc}(1, k),\ y \mapsto k\}$ are not statically equivalent for E_{enc}: let $M = \mathrm{dec}(x, y)$ and $N = 0$. M and N use only variables defined by φ and φ' and do not use any names. Moreover $M\varphi =_{E_{\mathsf{enc}}} N\varphi$ but $M\varphi \neq_{E_{\mathsf{enc}}} N\varphi$. The test $M \stackrel{?}{=} N$ distinguishes φ from φ'.

2.3 Concrete semantics

We now give terms and frames a concrete semantics, parameterized by an implementation of the primitives. Provided a set of sorts $\mathcal{S}$ and a set of symbols $\mathcal{F}$ as above, a *($\mathcal{S},\mathcal{F}$)-computational algebra* A consists of

- a non-empty set of bit-strings $[\![s]\!]_A \subseteq \{0,1\}^*$ for each sort $s \in \mathcal{S}$; moreover, if s_2 is a subsort of s_1 we require that $[\![s_2]\!]_A \subseteq [\![s_1]\!]_A$;
- a computable function $[\![f]\!]_A : [\![s_1]\!]_A \times \ldots \times [\![s_k]\!]_A \to [\![s]\!]_A$ for each $f \in \mathcal{F}$ with $\mathrm{ar}(f) = s_1 \times \ldots \times s_k \to s$;
- an effective procedure to draw random elements from $[\![s]\!]_A$, denoted $x \stackrel{R}{\longleftarrow} [\![s]\!]_A$.

Assume a fixed ($\mathcal{S},\mathcal{F}$)-computational algebra A. We associate to each frame $\varphi = \{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$ a distribution $\psi = [\![\varphi]\!]_A$, of which the drawings $\widehat{\psi} \stackrel{R}{\longleftarrow} \psi$ are computed as follows:

1. for each name a of sort s appearing in $t_1, \ldots, t_n$, draw a value $\widehat{a} \stackrel{R}{\longleftarrow} [\![s]\!]_A$;
2. for each x_i ($1 \leq i \leq n$) of sort s_i, compute $\widehat{t_i} \in [\![s_i]\!]_A$ recursively on the structure of terms: $f(\widehat{t'_1, \ldots, t'_m}) = [\![f]\!]_A(\widehat{t'_1}, \ldots, \widehat{t'_m})$;
3. return the value $\widehat{\psi} = \{x_1 \mapsto \widehat{t_1}, \ldots, x_n \mapsto \widehat{t_n}\}$.

Such values $\phi = \{x_1 = e_1, \ldots, x_n = e_n\}$ with $e_i \in [\![s_i]\!]_A$ are called *concrete frames*. We extend the notation $[\![.]\!]_A$ to (tuples of) closed terms in the obvious way. We also generalize the notation to terms with variables, by specifying the concrete values for all of them: $[\![.]\!]_{A,\{x_1=e_1,\ldots,x_n=e_n\}}$.

In the rest of the paper we focus on asymptotic notions of cryptographic security and consider families of computational algebra (A_η) indexed by a complexity parameter $\eta \geq 0$. (This parameter η might be thought of as the size of keys and other secret values.) The *concrete semantics* of a frame φ is a family of distributions over concrete frames $([\![\varphi]\!]_{A_\eta})$. We only consider families of computational algebras (A_η) such that each required operation on algebras is feasible by a (uniform, probabilistic) polynomial-time algorithm in the complexity parameter η. This ensures that the concrete semantics of terms and frames is efficiently computable (in the same sense).

Families of distributions (*ensembles*) over concrete frames benefit from the usual notion of cryptographic indistinguishability. Intuitively, two families of distributions (ψ_η) and (ψ'_η) are *indistinguishable*, written $(\psi_\eta) \approx (\psi'_\eta)$, if and only if no probabilistic polynomial-time adversary $\mathcal{A}$ can guess whether he is given a sample from ψ_η or ψ'_η with a probability significantly greater than $\frac{1}{2}$. Formally, we ask the *advantage* of $\mathcal{A}$,

$$\mathrm{Adv}^{\mathrm{IND}}_{\mathcal{A}}(\psi_\eta, \psi'_\eta) = \left| \mathbb{P}[\widehat{\psi} \stackrel{R}{\longleftarrow} \psi_\eta \ : \ \mathcal{A}(\widehat{\psi}) = 1] - \mathbb{P}[\widehat{\psi} \stackrel{R}{\longleftarrow} \psi'_\eta \ : \ \mathcal{A}(\widehat{\psi}) = 1] \right|$$

to be a *negligible* function of η, that is, to remain eventually smaller than any η^{-n} ($n > 0$) for sufficiently large η. By convention, the adversaries are given access implicitly to as many fresh random coins as needed, as well as the complexity parameter η.

3 Adaptive soundness

In this section, we recall the original notion of soundness for static equivalence which considers a passive adversary [9] and then extend it to an adaptive adversary. We show relations between the classical soundness and our new adaptive soundness and also provide a combination result which allows us, under some hypotheses, to prove adaptive soundness of computational algebras (A_η) from adaptive soundness of parts of (A_η).

3.1 Soundness definitions

Definition 3 ($\approx_E$-soundness). *Let E be an equational theory. A family of computational algebras (A_η) is* $\approx_E$-sound *iff for every frames φ_1, φ_2 with the same domain, $\varphi_1 \approx_E \varphi_2$ implies that $([\![\varphi_1]\!]_{A_\eta}) \approx ([\![\varphi_2]\!]_{A_\eta})$,*

Similarly, Baudet *et al.* [9] define soundness for $=_E$ and $\vdash_E$. We here concentrate on soundness of static equivalence. As shown in [9], for many theories soundness of static equivalence implies all of the other notions. Baudet *et al.* also introduce a strong notion of soundness that holds without restriction on the computational power of adversaries.

Definition 4 (Unconditional $\approx_E$-soundness). *Let E be an equational theory. A family of computational algebras (A_η) is* unconditionally $\approx_E$-sound *iff for every frames φ_1, φ_2 with the same domain, $\varphi_1 \approx_E \varphi_2$ implies $([\![\varphi_1]\!]_{A_\eta}) = ([\![\varphi_2]\!]_{A_\eta})$.*

Unconditional soundness stipulates that for any pairs of equivalent frames, the related distributions are equal. Hence even an adversary which is not polynomially bounded cannot distinguish these two distributions.

3.2 Adaptive security

We extend soundness of static equivalence to the adaptive setting from [25] where the adversary observes the computational value of a sequence of adaptively chosen terms.

The adaptive setting is formalized through the following cryptographic game. Let (A_η) be a family of computational algebras and $\mathcal{A}$ be an adversary. $\mathcal{A}$ has access to a left-right evaluation oracle $\mathcal{O}_{LR}$ which given a pair of symbolic terms (t_0, t_1) outputs either the implementation of t_0 or of t_1. This oracle depends on a selection bit b and uses a local store to record values generated for the different names (these values are used when processing further queries). With a slight abuse of notation, we omit this store and write: $\mathcal{O}^b_{LR,A_\eta}(t_0, t_1) = [\![t_b]\!]_{A_\eta}$. Adversary $\mathcal{A}$ plays an indistinguishability game with the objective of finding the value of b. Formally the advantage of $\mathcal{A}$ is defined by:

$$\mathrm{Adv}^{\mathsf{ADPT}}_{\mathcal{A},A_\eta}(\eta) = \left|\mathbb{P}\left[\mathcal{A}^{\mathcal{O}^1_{LR,A_\eta}} = 1\right] - \mathbb{P}\left[\mathcal{A}^{\mathcal{O}^0_{LR,A_\eta}} = 1\right]\right|$$

Without further restrictions on the queries of the adversary, having a non-negligible advantage is easy in most cases. For example the adversary could submit a pair $(0, 1)$ to his oracle. We therefore require the adversary to be *legal*.

Definition 5 (Adaptive soundness). *An adversary $\mathcal{A}$ is* legal *if for any sequence of queries $(t_0^i, t_1^i)_{1\leq i\leq n}$ made by $\mathcal{A}$ to its left-right oracle, queries are statically equivalent: $\{x_1 \mapsto t_0^1, \ldots, x_n \mapsto t_0^n\} \approx_E \{x_1 \mapsto t_1^1, \ldots, x_n \mapsto t_1^n\}$. A family of computational algebras (A_η) is*

- $\approx_E$-ad-sound *iff $\mathrm{Adv}^{\mathsf{ADPT}}_{\mathcal{A},A_\eta}(\eta)$ is negligible for any probabilistic polynomial-time legal adversary $\mathcal{A}$.*
- unconditionally $\approx_E$-ad-sound *iff $\mathrm{Adv}^{\mathsf{ADPT}}_{\mathcal{A},A_\eta}(\eta)$ is 0 for any legal adversary $\mathcal{A}$.*

Adaptive soundness implies the original soundness notion for static equivalence.

Proposition 1. *Let (A_η) be a family of computational algebras. If A_η is $\approx_E$-ad-sound then A_η is also $\approx_E$-sound but the converse is false in general.*

Interestingly, for unconditional soundness, the adaptive and non-adaptive case coincide.

Proposition 2. *Let (A_η) be a family of computational algebras. A_η is unconditionally $\approx_E$-ad-sound iff A_η is unconditionally $\approx_E$-sound.*

3.3 Combination result

Our objective here is to provide a combination result of the form: let Σ_1 and Σ_2 be two signatures that do not share any symbol. If A_η^1 is $\approx_{E_1}$-ad-sound and A_η^2 is $\approx_{E_2}$-ad-sound, then the combination of A_η^1 and A_η^2 denoted $A_\eta^1 \times A_\eta^2$ is $\approx_{E_1\cup E_2}$-ad-sound. However this is false in general. Therefore, we provide restrictions under which combination is possible: we consider disjoint signatures as well as layered signatures.

Definition 6 (Disjoint signatures). *Let $\Sigma_1 = (\mathcal{S}_1, \mathcal{F}_1)$ and $\Sigma_2 = (\mathcal{S}_2, \mathcal{F}_2)$ be two signatures. We say that Σ_1 and Σ_2 are disjoint iff $\mathcal{F}_1 \cap \mathcal{F}_2 = \emptyset$ and $\mathcal{S}_1 \cap \mathcal{S}_2 = \emptyset$.*

We denote by $\Sigma_1 \cup \Sigma_2 = (\mathcal{S}_1 \cup \mathcal{S}_2, \mathcal{F}_1 \cup \mathcal{F}_2)$ the union of the signature Σ_1 with Σ_2.

Definition 7 (Signature combination, layered signatures). *Let $\Sigma_1 = (\mathcal{S}_1, \mathcal{F}_1)$ and $\Sigma_2 = (\mathcal{S}_2, \mathcal{F}_2)$ be two disjoint signatures. A subsort relation S is a* signature combination *for Σ_1 and Σ_2 if $\mathsf{S} \subseteq \mathcal{S}_2 \times \mathcal{S}_1$. Then $\Sigma = \Sigma_1 \cup \Sigma_2$ is a $(\Sigma_1, \Sigma_2)_{\mathsf{S}}$-layered signature.*

Intuitively, if a signature is layered then a constructor of $\mathcal{F}_1$ never occurs under a constructor of $\mathcal{F}_2$ and S defines which sorts of Σ_2 can be used as subsort of Σ_1. Given a $(\Sigma_1, \Sigma_2)_{\mathsf{S}}$-layered signature Σ and a term t over Σ we define the set of Σ_1 positions of t, $pos_{\Sigma_1}(t) = \{p \mid p \in pos(t), sort(t|_p) \in \mathcal{S}_1\}$ and the set of Σ_2 minimal positions of t, $pos^\star_{\Sigma_2}(t) = \{p \mid p \in pos(t), sort(t|_p) \in \mathcal{S}_2, p = p' \cdot i \Rightarrow sort(t|_{p'}) \notin \mathcal{S}_2\}$.

As an example consider a theory with symmetric encryption and a pseudo-random generator. Signature Σ_1 contains a sort Data and two symbols enc and dec, both of arity $\mathrm{Data} \times \mathrm{Data} \to \mathrm{Data}$. Signature Σ_2 contains one sort Rand and a symbol prg (a pseudo-random generator) of arity $\mathrm{Rand} \to \mathrm{Rand}$. The signature combination S contains a single element $(\mathrm{Rand}, \mathrm{Data})$: elements of sort Rand can be used as keys or as plaintext. Σ_1 and Σ_2 are disjoint, and $\Sigma = \Sigma_1 \cup \Sigma_2$ is $(\Sigma_1, \Sigma_2)_{\mathsf{S}}$-layered. Given the term $t = \mathrm{enc}(\mathrm{enc}(\mathrm{prg}(r), k), \mathrm{prg}(\mathrm{prg}(r')))$ where $k \in \mathrm{Data}$ and $r, r' \in \mathrm{Rand}$, t is indeed a valid term of Σ. However, the term $t' = \mathrm{prg}(\mathrm{enc}(r, k))$ is not a term of Σ as it is not well sorted. We have that $pos_{\Sigma_1}(t) = \{\epsilon, 1, 12\}$ and $pos^{\star}_{\Sigma_2} = \{11, 2\}$.

Definition 8 (Hybrid functions). *Let Σ_1, Σ_2 be two disjoint signatures and* S *a signature combination such that $\Sigma = \Sigma_1 \cup \Sigma_2$ is $(\Sigma_1, \Sigma_2)_{\mathsf{S}}$-layered. Let E_1, E_2 be equational theories over Σ_1 and Σ_2 respectively. A (E_1, E_2)-*hybrid function *for a set F of pairs of frames is a function σ from lists of terms over Σ to terms over Σ such that:*

- *for any frame φ occurring in F, $\varphi \approx_{E_1} \sigma(\varphi)$ where we naturally extended σ over frames by $\sigma(\{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}) = \{x_1 \mapsto \sigma([t_1]), \ldots, x_n \mapsto \sigma([t_1 \ldots t_n])\}$;*
- *for any $(\varphi, \varphi') \in F$, if $\varphi \approx_{E_1 \cup E_2} \varphi'$ then let $\varphi = \{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$ and $\varphi' = \{x_1 \mapsto u_1, \ldots, x_n \mapsto u_n\}$. We have that for all i in $[1, n]$,*
  - *$pos_{\Sigma_1}(\sigma([t_1 \ldots t_i])) = pos_{\Sigma_1}(\sigma([u_1 \ldots u_i])) = P$ and for any $p \in P$*
  $$root(\sigma([t_1 \ldots t_i])|_p) = root(\sigma([u_1 \ldots u_i])|_p)$$
  - *$pos^{\star}_{\Sigma_2}(\sigma([t_1 \ldots t_i])) = pos^{\star}_{\Sigma_2}(\sigma([u_1 \ldots u_i])) = Q$ and we have that*
  $$\{x_q \mapsto \sigma([t_1 \ldots t_i])|_q\}_{q \in Q} \approx_{E_2} \{x_q \mapsto \sigma([u_1 \ldots u_i])|_q\}_{q \in Q}$$

Moreover σ has to be computable in polynomial time (in its input).

Adaptive soundness may not hold on all frames, but only on a subset of well-formed frames, *e.g.*, when considering encryption one typically discards all frames that contain key cycles. Therefore we say that an abstract algebra A_η is $\approx_E$*-ad-sound for a set F of pair of frames* if the advantage $\mathrm{Adv}^{\mathsf{ADPT}}_{\mathcal{A}, A_\eta}(\eta)$ of any polynomial-time legal adversary $\mathcal{A}$, whose sequence of queries $(t^i_0, t^i_1)_i$ verifies that the pair $(\{x_i \mapsto t^i_0\}_i, \{x_i \mapsto t^i_1\}_i)$ is in F, is negligible. We typically show soundness for the set of all pairs of "well-formed" frames (the notion of well-formed frames depends on the particular equational theory).

Proposition 3 (Combination). *Let Σ_1 and Σ_2 be two disjoint signatures and* S *be a signature combination for Σ_1 and Σ_2. Let E_1 and E_2 be equational theories over Σ_1 and Σ_2 respectively. We consider a family of computational algebras (A^1_η) for Σ_1 and another family (A^2_η) for Σ_2 respecting* S, i.e. $(s_2, s_1) \in \mathsf{S}$ *implies that* $[\![s_2]\!]_{A^2_\eta} \subseteq [\![s_1]\!]_{A^1_\eta}$.

Let F be a set of pair of frames over $\Sigma_1 \cup \Sigma_2$ and σ be a (E_1, E_2)-hybrid function for F. If $A^1_\eta \times A^2_\eta$ is $\approx_{E_1}$-ad-sound for $G = \{(\varphi, \sigma(\varphi)) \mid \varphi$ occurs in $F\}$ and A^2_η is $\approx_{E_2}$-ad-sound for frames on Σ_2, then $A^1_\eta \times A^2_\eta$ is $\approx_{E_1 \cup E_2}$-ad-sound for F.

The idea of the proof is that if an adversary $\mathcal{A}$ against $E_1 \cup E_2$-ad-soundness queries his oracle with a pair of frames (φ, φ') in F then it is possible to build an adversary $\mathcal{B}_1$ against E_1-ad-soundness who submits $(\varphi, \sigma(\varphi))$ to his oracle, an adversary $\mathcal{B}_2$ against E_2-ad-soundness who submits $(\sigma(\varphi), \sigma(\varphi'))$ and an adversary $\mathcal{B}_3$ against E_1-ad-soundness who submits $(\sigma(\varphi'), \varphi')$ such that the advantages of $\mathcal{A}$, $\mathcal{B}_1$, $\mathcal{B}_2$ and $\mathcal{B}_3$ are related. This combination result will be useful in Section 4 when combining encryption with modular exponentiation.

4 Adaptively sound theories

We now present adaptive soundness results for several equational theories.We consider probabilistic symmetric encryption and try to be as close as possible to the models from [4] and from [25]. We assume that the implementation of the symmetric encryption scheme is semantically secure [22] and use a relevant formal theory.

Symbolic model. Our symbolic model consists of the set of sorts $\mathcal{S} = \{\mathrm{Data}\}$, an infinite number of names for sort Data called keys and the function symbols:

$$\begin{array}{rlrl} \mathrm{enc}, \mathrm{dec} : & \mathrm{Data} \times \mathrm{Data} \to \mathrm{Data} & \mathrm{samekey} : & \mathrm{Data} \times \mathrm{Data} \to \mathrm{Data} \\ \mathrm{pair} : & \mathrm{Data} \times \mathrm{Data} \to \mathrm{Data} & \mathrm{tenc}, \mathrm{tpair} : & \mathrm{Data} \to \mathrm{Data} \\ \pi_l, \pi_r : & \mathrm{Data} \to \mathrm{Data} & 0, 1 : & \mathrm{Data} \end{array}$$

We consider the equational theory E_{sym} generated by:

$$\begin{array}{rr} \mathrm{dec}(\mathrm{enc}(x, y), y) = x & \pi_l(\mathrm{pair}(x, y)) = x \\ \pi_r(\mathrm{pair}(x, y)) = y & \mathrm{samekey}(\mathrm{enc}(x, y), \mathrm{enc}(z, y)) = 1 \\ \mathrm{tenc}(\mathrm{enc}(x, y)) = 1 & \mathrm{tpair}(\mathrm{pair}(x, y)) = 1 \end{array}$$

Intuitively, the function symbols $\mathrm{tenc}, \mathrm{tpair}$ are type testers. The meaning of the remaining symbols should be clear. As usual $\mathrm{enc}(t, k)$ is also written $\{t\}_k$ and $\mathrm{pair}(t, t')$ is also written (t, t'). A name k is used at a key position in a term t if there exists a sub-term $\mathrm{enc}(t', k)$ of t. Else k is used at a plaintext position.

Well-formed frames and adversaries. The importance of key cycles was already described in [4]. In general IND-CPA is not sufficient to prove any soundness result in the presence of key cycles. Thus, as in numerous previous work, we forbid the formal terms to contain such cycles. Let $\prec$ be a total order among keys. A *frame* φ *is acyclic for* $\prec$ if for any subterm $\{t\}_k$ of φ, if k' occurs in t then $k' \prec k$. (Another possibility to handle key cycles is to consider stronger computational requirements like Key Dependent Message – KDM – security as done in [5].) Moreover as noted in [25], selective decommitment [21] can be a problem. The classical solution to this problem is to require keys to be sent *before* being used to encrypt a message or they must never appear as a plaintext. A *frame* $\varphi = \{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$ *is well-formed for* $\prec$ if

- φ is acyclic for $\prec$;
- the terms t_i only use symbols enc, pair, 0 and 1, and only names are used at key positions;
- if k is used as plaintext in t_i, then k cannot be used at a key position in t_j for $j < i$.

An *adversary is well-formed for* $\prec$ if the sequence of queries $(t_0^i, t_1^i)_{1 \leq i \leq n}$ that he makes to his oracle yields two well-formed frames $\{x_1 \mapsto t_0^1, \ldots, x_n \mapsto t_0^n\}$ and $\{x_1 \mapsto t_1^1, \ldots, x_n \mapsto t_1^n\}$ for $\prec$.

Concrete model. A symmetric encryption scheme $\mathcal{SE}$ is defined by three algorithms $\mathcal{KG}$, $\mathcal{E}$ and $\mathcal{D}$. The key generation algorithm takes as input the security parameter η and outputs a key k. The encryption algorithm $\mathcal{E}$ is randomized. It takes as input a bit-string s, a key k and returns the encryption of s using k. The decryption algorithm $\mathcal{D}$ takes as input a bit-string c (a ciphertext), a key k and outputs the corresponding plaintext. Given $k \leftarrow \mathcal{KG}(\eta)$, for any bit-string s, if $c \leftarrow \mathcal{E}(k, s)$ then $\mathcal{D}(c) = s$.

The family of computational algebras (A_η) giving the concrete semantics depends on a symmetric encryption scheme $\mathcal{SE} = (\mathcal{KG}, \mathcal{E}, \mathcal{D})$. The concrete domain $[\![\mathrm{Data}]\!]_{A_\eta}$ contains all the possible bit-strings and is equipped with the distribution induced by $\mathcal{KG}$. Interpretation for constants 0 and 1 are respectively bit-strings 0^η and 1^η. The enc and dec function are respectively interpreted using algorithm $\mathcal{E}$ and $\mathcal{D}$. We assume the existence in the concrete model of a concatenation operation which is used to interpret the pair symbol. The corresponding left and right projections implement π_l and π_r. Finally, as we are only interested in well-formed frames, we do not provide any computational interpretation for tenc, tpair and $\mathrm{samekey}$.

Semantic security. In this paper we use schemes that satisfy length-concealing semantic security. The definition that we recall below uses a left-right encryption oracle $LR^b_{\mathcal{SE}}$. This oracle first generates a key k using $\mathcal{KG}$. Then it answers queries of the form (bs_0, bs_1), where bs_0 and bs_1 are bit-strings. The oracle returns ciphertext $\mathcal{E}(bs_b, k)$. The goal of the adversary $\mathcal{A}$ is to guess the value of bit b. His advantage is defined as:

$$\mathrm{Adv}^{cpa}_{\mathcal{SE},\mathcal{A}}(\eta) = \left|\mathbb{P}\left[\mathcal{A}^{LR^1_{\mathcal{SE}}} = 1\right] - \mathbb{P}\left[\mathcal{A}^{LR^0_{\mathcal{SE}}} = 1\right]\right|$$

Encryption scheme $\mathcal{SE}$ is IND-CPA secure if the advantage of any adversary $\mathcal{A}$ is negligible in η. The difference with standard semantic security is that we require the scheme to hide the length of the plaintext (and therefore we do not restrict bs_0 and bs_1 to have equal length). By abuse of notation we call the resulting scheme also IND-CPA secure.

Proposition 4. *Let $\prec$ be a total order among keys. In the remainder of this proposition we only consider well-formed adversaries for $\prec$. Let (A_η) be a family of computational algebras based on a symmetric encryption scheme $\mathcal{SE}$. (A_η) is $\approx_{E_{\mathsf{sym}}}$-ad-sound if $\mathcal{SE}$ is* IND-CPA *but the converse is false.*

4.1 Exclusive OR

We study the adaptive soundness problem for the usual theory and implementation of the Exclusive Or (XOR) in the same model as given in [9]. The symbolic model $\Sigma_\oplus$ consists of a single sort $\mathrm{Data}_\oplus$, an infinite number of names, the infix symbol $\oplus$: $\mathrm{Data}_\oplus \times \mathrm{Data}_\oplus \to \mathrm{Data}_\oplus$ and two constants $0_\oplus, 1_\oplus$: $\mathrm{Data}_\oplus$. Terms are equipped with the equational theory $E_\oplus$ generated by:

$$(x \oplus y) \oplus z = x \oplus (y \oplus z) \qquad x \oplus y = y \oplus x \qquad x \oplus x = 0_\oplus \qquad x \oplus 0_\oplus = x$$

As an implementation, we define the computational algebras A_η: the concrete domain $[\![\mathrm{Data}_\oplus]\!]_{A_\eta}$ is $\{0,1\}^\eta$ equipped with the uniform distribution; $\oplus$ is interpreted by the usual XOR function over $\{0,1\}^\eta$, $[\![0_\oplus]\!]_{A_\eta} = 0^\eta$, $[\![1_\oplus]\!]_{A_\eta} = 1^\eta$. This implementation of XOR enjoys unconditional adaptive soundness with respect to $\approx_{E_\oplus}$.

Proposition 5. *The usual implementation for $E_\oplus$ is unconditionally $\approx_{E_\oplus}$-ad-sound.*

The result follows directly from unconditionally $\approx_{E_\oplus}$-soundness shown in [9] and Proposition 2.

4.2 Modular exponentiation

As a third application, we study soundness of modular exponentiation. The underlying cryptographic assumption is hardness of the *Decisional Diffie-Hellman* (DDH) problem: given g^x and g^y, it is difficult for any feasible computation to distinguish between g^{xy} and g^r, when x, y and r are selected at random. The original Diffie-Hellman protocol [20] has been used as a building block for several key agreement protocols that are widely used in practice (e.g. SSL/TLS and Kerberos V5) as well as for group key exchange protocols such as AKE1 [12] or the Burmester-Desmedt protocol [14].

Symbolic model. The symbolic model consists of sorts G (group elements) and R (ring elements), an infinite number of names for R (but no name for sort G) and the symbols:

$\exp : R \to G$	exponentiation	$+, \cdot : R \times R \to R$	add, mult
$* : G \times G \to G$	mult in $\mathbb{G}$	$- : R \to R$	inverse
	$0_R, 1_R : R$	constants	

We consider the equational theory E_{DH} generated by:

$$\begin{array}{ccc} x + y = y + x & x \cdot y = y \cdot x & (x+y)+z = x+(y+z) \\ x\cdot(y+z) = x\cdot y + x\cdot z & (x\cdot y)\cdot z = x\cdot(y\cdot z) & x + (-x) = 0_R \\ 0_R + x = x & 1_R \cdot x = x & \exp(x) * \exp(y) = \exp(x+y) \end{array}$$

There exists a direct correspondence between terms of sort R and the set of polynomials $\mathbb{Z}[\mathcal{N}_R]$ where $\mathcal{N}_R$ is the set of names of sort R. An integer i simply corresponds to $\underbrace{1_R + \ldots + 1_R}_{i\ times}$ if $i > 0$, to $-(\underbrace{1_R + \ldots + 1_R}_{i\ times})$ if $i < 0$ and to 0_R if $i = 0$. We also write x^n for $\underbrace{x \cdot \ldots \cdot x}_{n\ times}$.

We put two restrictions on formal terms: products have to be *power-free, i.e.,* x^n is forbidden for $n > 1$, and products must not contain more than l elements for some fixed bound l, i.e. $x_1 \cdot \ldots \cdot x_n$ is forbidden for $n > l$. Both restrictions come from the DDH assumption and seem difficult to avoid [13]. Furthermore we are only interested in frames using terms of sort G. Any frame containing only terms of sort G can be rewritten as $\{x_1 \mapsto \exp(p_1), \ldots, x_n \mapsto \exp(p_n)\}$ by orienting the last equation form left to right. For such frames there is an immediate characterization of static equivalence. Two frames are statically equivalent if they satisfy the same linear equations.

Proposition 6. *We have that* $\{x_1 \mapsto \exp(p_1), \ldots, x_n \mapsto \exp(p_n)\} \approx_{E_{\mathsf{DH}}} \{x_1 \mapsto \exp(q_1), \ldots, x_n \mapsto \exp(q_n)\}$ *iff for any sequence of integer* $a_0, a_1, \ldots a_n$ *we have* $a_0 + \sum_{i=1}^n a_i p_i = 0 \Leftrightarrow a_0 + \sum_{i=1}^n a_i q_i = 0$

This characterization can be used to decide static equivalence efficiently.

Concrete model. An Instance Generator IG is a polynomial-time (in η) algorithm that outputs a cyclic group $\mathbb{G}$ (defined by a generator g, an order q and a polynomial-time multiplication algorithm) of prime order q. The family of computational algebras (A_η) depends on an instance generator IG which generates a cyclic group $\mathbb{G}$ of generator g and of order q: the concrete domain $[\![R]\!]_{A_\eta}$ is $\mathbb{Z}_q$ with the uniform distribution. Symbols $+$ and $\cdot$ are the classical addition and multiplication over $\mathbb{Z}_q$, $\exp$ is interpreted as modular exponentiation of g. Constants 0_R and 1_R are interpreted by integers 0 and 1 of $\mathbb{Z}_q$. The domain $[\![G]\!]_{A_\eta}$ contains all bit-string representations of elements of $\mathbb{G}$.

A family of computational algebras satisfies the DDH assumption if its instance generator satisfies the assumption: for every probabilistic polynomial-time adversary $\mathcal{A}$, his advantage $\mathrm{Adv}^{\mathsf{DDH}}_{IG,\mathcal{A}}(\eta) = |\mathbb{P}\left[(g,q) \leftarrow IG(\eta)\ :\ a,b \leftarrow \mathbb{Z}_q\ :\ \mathcal{A}(g^a, g^b, g^{ab}) = 1\right] - \mathbb{P}\left[(g,q) \leftarrow IG(\eta)\ :\ a,b,c \leftarrow \mathbb{Z}_q\ :\ \mathcal{A}(g^a, g^b, g^c) = 1\right]|$ is negligible in η. In the remainder, we generally suppose that for any η there is a unique group given by IG. We show that the DDH assumption is necessary and sufficient to prove adaptive soundness.

Proposition 7. *A family of computational algebras (A_η) is $\approx_{E_{\mathsf{DH}}}$-sound iff (A_η) is $\approx_{E_{\mathsf{DH}}}$-ad-sound iff (A_η) satisfies the* DDH *assumption.*

The proof of this result uses an adaptive variant of DDH called 3DH: it generalizes several previously used variants of DDH. The main difficulty in this proof consists in relating DDH and 3DH. Note that while adaptive soundness and (classical) soundness are not equivalent for symmetric encryption, they coincide in this case.

4.3 Combining encryption with exponentiation

We illustrate our combination result (Proposition 3) by establishing a joint soundness result for symmetric encryption and modular exponentiation.

Symbolic model. We consider an equational theory E containing both E_{DH} and E_{sym}. Let Σ_1 be the signature for symmetric encryption and Σ_2 be the signature for modular exponentiation, then signature $\Sigma = \Sigma_1 \cup \Sigma_2$ is $(\Sigma_1, \Sigma_2)_{\mathsf{S}}$-layered where S contains only one element (G, Data).

Well-formed frames. Let $\prec$ be a total order between keys and exponentiations. A frame φ (on Σ) is well-formed for $\prec$ if:

- φ does not contain any dec, tenc, tpair, π_l, π_r or $*$ symbol, only names and exponentiations are used at key position.
- For any subterm $\exp(p)$ of φ used at a key position, p is linearly independent of other polynomials p' such that $\exp(p')$ is a subterm of φ.
- For any subterm $\{t\}_{t'}$ of φ, if t'' is a subterm of t which is a name of sort Data or an exponentiation then $t'' \prec t'$.

Concrete model. The concrete model is given by the models for symmetric encryption and modular exponentiation. We need to reflect that exponentiations can be used as symmetric keys. The family of computational algebras (A_η) giving the concrete semantics is parameterized by a symmetric encryption scheme $\mathcal{SE}$ and an instance generator

IG. We require that the key generation algorithm of $\mathcal{SE}$ randomly samples an element of $IG(\eta)$. Giving an $\mathsf{IND\text{-}CPA}$ encryption scheme $\mathcal{SE}'$, it is possible to build another $\mathsf{IND\text{-}CPA}$ scheme $\mathcal{SE}$ which indeed uses such a key generation algorithm. This is achieved by using a *key extractor* algorithm Kex [16]. This algorithm (usually a universal hash function used with the entropy smoothing theorem) transforms group elements into valid keys for $\mathcal{SE}'$. The new encryption and decryption algorithms of $\mathcal{SE}$ apply the Kex algorithm to the group element which is used as key. This produces a symmetric key which can be used with the encryption and decryption algorithms of $\mathcal{SE}'$.

The family of computational algebras (A_η) implementing encryption with exponentiation is said *EE-secure* if the encryption scheme $\mathcal{SE}$ is secure against $\mathsf{IND\text{-}CPA}$ and uses a key generation algorithm as described above and IG satisfies the DDH assumption. Soundness is proven by applying Proposition 3.

Proposition 8. *Let $\prec$ be a total order between keys and exponentiations. An EE-secure family of computational algebras (A_η) is $\approx_E$-ad-sound for well-formed frames for $\prec$.*

A similar result is given for symmetric encryption and XOR in the full version [23].

5 Analysis of dynamic group key exchange

Micciancio and Panjwani exemplified their adaptive soundness result from [25] on multicast protocols. We propose another application: *dynamic group key exchange protocols* (DKE) such as the AKE1 protocol [12]. To keep the symbolic security notion as simple as possible we define security for protocols using only modular exponentiation: we consider a subtheory E of E_{DH} (Section 4.2) without $+$, $-$, 1_R and 0_R symbols and the related equations. However our definitions and soundness results can be adapted to other equational theories (e.g. symmetric encryption joint with modular exponentiation).

5.1 Dynamic group protocols

We take a simple model for DKE in the adaptive setting. A DKE protocol is described by four operations which specify the protocol. We suppose that this specification is given by four polynomial-time algorithms $(\mathcal{S}, \mathcal{J}, \mathcal{L}, \mathcal{K})$:

- $\mathcal{S}$ initializes a new group. The algorithm takes as an input a list of users and outputs the internal state s_0 of the protocol as well as a list of formal terms which model the messages that have been exchanged during the setup phase.
- $\mathcal{J}$ and $\mathcal{L}$ take as input the state of the protocol s and a list of users U_1 to U_n (to be respectively added to or suppressed from the group) and output the updated state of the protocol s' as well as a list of formal terms representing message exchanges.
- $\mathcal{K}$ takes as input the state of the group s and outputs a formal term representing the shared key of the group.

The internal state of the protocol can be thought of as the internal state of the four algorithms that describe the protocol.

We partition the set of names of sort R according to the users: n_i^j, $j \in \mathbb{N}$, are the nonces generated by user U_i. We require that the formal term output by $\mathcal{K}$ only uses nonces for users that are currently in the group.

5.2 Security in the symbolic model

In our symbolic setting, the security property is expressed as reachability in a transition system. We represent the states of this transition system as a triple $\langle L, C, T\rangle$ where

- L is the list of users that are currently in the group;
- C is the set of corrupted users;
- T is the list of formal terms sent during the protocol execution.

We suppose that the internal state of the protocol can be recovered from the state $\langle L, C, T\rangle$ and tend to assimilate these two notions of state. We now describe the possible transitions. For convenience, we use set notations for manipulating lists.

1. $\langle \emptyset, C, \emptyset\rangle \xrightarrow{\mathsf{c}(U)} \langle \emptyset, C \cup \{U\}, \emptyset\rangle$: corruption of user U.
2. $\langle \emptyset, C, \emptyset\rangle \xrightarrow{\mathsf{s}(\mathcal{U})} \langle \mathcal{U}, C, T\rangle$: setup of the group given by the list of users $\mathcal{U}$, *i.e.*, $\langle \mathcal{U}, C, T\rangle$ is computed by $\mathcal{S}(\langle \emptyset, C, \emptyset\rangle, \mathcal{U})$.
3. $\langle L, C, T\rangle \xrightarrow{\mathsf{j}(\mathcal{U})} \langle L \cup \mathcal{U}, C, T\rangle \cup T'$: join of users in the list $\mathcal{U}$, *i.e.*, $\langle L \cup \mathcal{U}, C, T \cup T'\rangle$ is computed by $\mathcal{J}(\langle L, C, T\rangle, \mathcal{U})$.
4. $\langle L, C, T\rangle \xrightarrow{\mathsf{l}(\mathcal{U})} \langle L \setminus \mathcal{U}, C, T \cup T'\rangle$: exclusion of the users in the list $\mathcal{U}$, *i.e.*, $\langle L \setminus \mathcal{U}, C, T \cup T'\rangle$ is computed by $\mathcal{L}(\langle L, C, T\rangle, \mathcal{U})$.

To simplify things up, we consider a static corruption model, *i.e.*, corruption transitions only occur at the beginning of the protocol. Then a setup transition is taken followed by leave and join transitions. A DKE protocol is secure if it is impossible for an adversary to get any bit of information on the group key when no corrupted users are in the group.

Definition 9. *We define a DKE protocol to be* symbolically secure *if for any state $\langle L, C, T = \{t_1, ..., t_n\}\rangle$ reachable from $\langle \emptyset, \emptyset, \emptyset\rangle$ and such that $C \cap L = \emptyset$ we have*

$$\{x_1 \mapsto t'_1, ..., x_n \mapsto t'_n, y \mapsto \mathcal{K}(\langle L, C, T\rangle)\} \approx_E \{x_1 \mapsto t'_1, ..., x_n \mapsto t'_n, y \mapsto \exp(r)\}$$

where r is a fresh nonce, $N = \{n_i^j \mid U_i \in C\}$ and t'_i is as t_i but nonces from N have been removed, i.e. if $t = \exp(m_1 \cdot \ldots \cdot m_\ell)$ then $t' = \exp(m'_1 \cdot \ldots \cdot m'_{\ell'})$ where $\{m'_1, \ldots, m'_{\ell'}\} = \{m_1, \ldots, m_\ell\} \setminus N$.

5.3 Security in the concrete model

We use a simplified version of the security model from [12]: some oracles in [12] are not useful anymore in the adaptive setting. Let $(\mathcal{S}, \mathcal{J}, \mathcal{L}, \mathcal{K})$ be a DKE and (A_η) a family of computational algebras. Adversary $\mathcal{A}$ interacts with the group via the following five oracles which store the current state s of the group and use a challenge bit b.

- $\mathsf{Setup}(U_1, ..., U_n)$: initializes the group using $\mathcal{S}(U_1, ..., U_n)$ which produces the new state s and a list of formal terms t_1 to t_m. $\mathcal{A}$ is given $[\![t_i]\!]_{A_\eta}$ for any i in $[1, m]$.
- $\mathsf{Join}(U_1, ..., U_n)$: users U_1 to U_n join the group. $\mathcal{J}(s, U_1, ..., U_n)$ is executed and outputs state s and a list of terms t_1 to t_m. $\mathcal{A}$ is given $[\![t_i]\!]_{A_\eta}$ for any i in $[1, m]$.

- $\mathsf{Leave}(U_1, ..., U_n)$: users U_1 to U_n leave the group. $\mathcal{L}(s, U_1, ..., U_n)$ is executed and outputs state s and a list of terms t_1 to t_m. $\mathcal{A}$ is given $[\![t_i]\!]_{A_\eta}$ for any i in $[1, m]$.
- $\mathsf{Corrupt}(U)$: $\mathcal{A}$ corrupts user U; all nonces generated by U are given to $\mathcal{A}$. As $\mathcal{A}$ works in polynomial time, a polynomial number of values is sufficient.
- Test: $\mathcal{A}$ either receives the key of the group (output by $\mathcal{K}(s)$) if $b = 1$ or a random key if $b = 0$. This oracle can only be queried once.

As we consider a static corruption model, queries to the $\mathsf{Corrupt}$ oracle have to be done before all further queries. Then the Setup oracle is called and after that the adversary interleaves queries to the Join and Leave oracles. The adversary makes a final call to the Test oracle. Let $\mathcal{O}_b$ denote the oracles with challenge bit b. The advantage of an adversary $\mathcal{A}$ is given by: $\mathrm{Adv}_{\mathcal{A},A_\eta}^{(\mathcal{S},\mathcal{J},\mathcal{L},\mathcal{K})}(\eta) = \mathbb{P}\left[\mathcal{A}^{\mathcal{O}_1} = 1\right] - \mathbb{P}\left[\mathcal{A}^{\mathcal{O}_0} = 1\right]$. A DKE is *secure in the concrete model* if the advantage of any adversary is negligible in η.

5.4 Soundness result

Our symbolic model for DKE is computationally sound: if a DKE algorithm is secure in the symbolic model, then it is secure in the computational model, provided that static equivalence is adaptively sound (remember that we consider only modular exponentiation hence static equivalence is adaptively sound under DDH).

Proposition 9. *Let (A_η) be a family of computational algebras and $\mathit{\Pi} = (\mathcal{S}, \mathcal{J}, \mathcal{L}, \mathcal{K})$ be a DKE. If (A_η) is $\approx_E$-ad-sound and $\mathit{\Pi}$ is secure in the symbolic model, then $\mathit{\Pi}$ is secure in the concrete model.*

References

1. M. Abadi, M. Baudet, and B. Warinschi. Guessing attacks and the computational soundness of static equivalence. In *Proc. 9th International Conference on Foundations of Software Science and Computation Structures (FoSSaCS'06)*, volume 3921 of *LNCS*, pages 398–412. Springer, 2006.
2. M. Abadi and V. Cortier. Deciding knowledge in security protocols under equational theories. In *Proc. 31st International Colloquium on Automata, Languages and Programming (ICALP'04)*, volume 3142 of *LNCS*, pages 46–58, 2004.
3. M. Abadi and C. Fournet. Mobile values, new names, and secure communications. In *Proc. 28th Annual ACM Symposium on Principles of Programming Languages (POPL'01)*, pages 104–115. ACM Press, 2001.
4. M. Abadi and P. Rogaway. Reconciling two views of cryptography (the computational soundness of formal encryption). In *IFIP International Conference on Theoretical Computer Science (IFIP TCS'00)*, volume 1872 of *LNCS*. Springer, 2000.
5. P. Adão, G. Bana, J. Herzog, and A. Scedrov. Soundness of formal encryption in the presence of key-cycles. In *Proc. 10th European Symposium on Research in Computer Security (ESORICS'05)*, volume 3679 of *LNCS*, pages 374–396. Springer, 2005.
6. M. Backes and B. Pfitzmann. Limits of the cryptographic realization of Dolev-Yao-style XOR. In *Proc. 10th European Symposium on Research in Computer Security (ESORICS'05)*, volume 3679 of *LNCS*, pages 336–354, 2005.

7. M. Backes, B. Pfitzmann, and M. Waidner. A composable cryptographic library with nested operations. In *Proc. 10th ACM Conference on Computer and Communications Security (CCS'03)*, pages 220–230, 2003.
8. G. Bana, P. Mohassel, and T. Stegers. The computational soundness of formal indistinguishability and static equivalence. In *Proc. 11th Asian Computing Science Conference (ASIAN'06)*, LNCS. Springer, 2006. To appear.
9. M. Baudet, V. Cortier, and S. Kremer. Computationally sound implementations of equational theories against passive adversaries. In *Proc. 32nd International Colloquium on Automata, Languages and Programming (ICALP'05)*, volume 3580 of *LNCS*, pages 652–663. Springer, 2005.
10. B. Blanchet. Automatic proof of strong secrecy for security protocols. In *Proc. 25th IEEE Symposium on Security and Privacy (SSP'04)*, pages 86–100, 2004.
11. B. Blanchet. A computationally sound mechanized prover for security protocols. In *Proc. 27th IEEE Symposium on Security and Privacy (SSP'06)*, pages 140–154, 2006.
12. E. Bresson, O. Chevassut, and D. Pointcheval. Provably authenticated group Diffie-Hellman key exchange – the dynamic case. In *Advances in Cryptology - ASIACRYPT '01*, volume 2248 of *LNCS*, pages 290–309. Springer, 2001.
13. E. Bresson, Y. Lakhnech, L. Mazaré, and B. Warinschi. A generalization of DDH with applications to protocol analysis and computational soundness. Submitted, an online version is available at `http://www.lsv.ens-cachan.fr/~mazare/BLMW.pdf`, 2007.
14. M. Burmester and Y. Desmedt. A secure and efficient conference key distribution system (extended abstract). In *Advances in Cryptology - EUROCRYPT'94*, volume 950 of *LNCS*, pages 275–286. Springer, 1994.
15. R. Canetti and J. Herzog. Universally composable symbolic analysis of mutual authentication and key-exchange protocols (extended abstract). In *Proc. 3rd Theory of Cryptography Conference (TCC'06)*, volume 3876 of *LNCS*, pages 380–403. Springer, 2006.
16. O. Chevassut, P.-A. Fouque, P. Gaudry, and D. Pointcheval. Key derivation and randomness extraction. Technical Report 2005/061, Cryptology ePrint Archive, 2005.
17. V. Cortier, S. Delaune, and P. Lafourcade. A Survey of Algebraic Properties Used in Cryptographic Protocols. *Journal of Computer Security*, To appear, 2005.
18. V. Cortier and B. Warinschi. Computationally sound, automated proofs for security protocols. In *European Symposium on Programming (ESOP'05)*, volume 3444 of *LNCS*, pages 157–171, Edinburgh, UK, 2005. Springer.
19. A. Datta, A. Derek, J. C. Mitchell, V. Shmatikov, and M. Turuani. Probabilistic Polynomial-time Semantics for a Protocol Security Logic. In *Proc. 32nd International Colloquium on Automata, Languages and Programming (ICALP'05)*, volume 3580 of *LNCS*, pages 16–29. Springer, 2005.
20. W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, 1976.
21. C. Dwork, M. Naor, O. Reingold, and L. J. Stockmeyer. Magic functions. *J. ACM*, 50(6):852–921, 2003.
22. S. Goldwasser and S. Micali. Probabilistic encryption & how to play mental poker keeping secret all partial information. In *Proc. 14th Annual ACM Symposium on Theory of Computing (STOC'82)*. ACM Press, 1982.
23. S. Kremer and L. Mazaré. Adaptive soundness of static equivalence. Research Report LSV-07-09, Laboratoire Spécification et Vérification, ENS Cachan, France, Feb. 2007. 27 pages.
24. P. Laud. A composable cryptographic library with nested operations. In *Proc. 12th ACM Conference on Computer and Communications Security (CCS'05)*, pages 26–35, 2005.
25. D. Micciancio and S. Panjwani. Adaptive security of symbolic encryption. In *Proc. 2nd Theory of Cryptography Conference (TCC'05)*, volume 3378 of *LNCS*, pages 169–187. Springer, 2005.