Computationally Sound Analysis of Protocols using Bilinear Pairings *

Laurent Mazaré

LSV, ENS Cachan & CNRS
mazare@lsv.ens-cachan.fr

Abstract. In this paper, we introduce a symbolic model to analyse protocols that use a bilinear pairing between two cyclic groups. This model consists in an extension of the Abadi-Rogaway logic and we prove that the logic is still computationally sound: symbolic indistinguishability implies computational indistinguishability provided that the Bilinear Decisional Diffie-Hellman assumption holds and that the encryption scheme is IND-CPA secure. We illustrate our results on classical protocols using bilinear pairing like Joux tripartite Diffie-Hellman protocol or the TAK-2 and TAK-3 protocols.
Keywords: Security, Formal Methods, Dolev-Yao Model, Computational Soundness, Bilinear Pairing

1 Introduction

Recently bilinear pairings such as Weil pairing or Tate pairing on elliptic and hyperelliptic curves have been used to build several cryptographic protocols. One of the first usable pairing-based protocol has been designed by Joux in [Jou00] where a key exchange protocol based on pairing is proposed. This protocol allows three participants to build a shared secret key in a singe round. However this protocol was only designed to be secure in the passive setting and is subject to man-in-the-middle attacks. Several key exchange protocols that extends this original protocol were developed, either to ensure some form of authentication [ARP03] or to extend it to a group setting [BDS03]. Pairings were also used as a robust building block for other cryptographic primitives such as identity based encryption schemes or signature schemes [DBS04].

Our contribution. In this paper, we propose an extension of the symbolic model from Dolev and Yao [DY83] for protocols using bilinear pairing and symmetric encryption. To the best of our knowledge, this is the first time pairings are considered in a Dolev-Yao like model. Moreover we prove that our symbolic model is sound in the computational setting: if there are no attacks in the symbolic setting, then attacks in the computational setting have only a negligible probability of success. This is done by extending the Abadi-Rogaway logic from [AR00] to symbolic terms using pairings. We use classical cryptographic assumptions from

* Work partly supported by the ARA SSIA Formacrypt.

the standard model to prove soundness: the symmetric encryption scheme has to satisfy indistinguishability against chosen-plaintext attacks (IND-CPA) and the bilinear mapping has to satisfy the bilinear decisional Diffie-Hellman assumption (BDDH). Under these assumptions, our soundness result can be used to prove computational security of protocols such as Joux tripartite Diffie-Hellman protocol [Jou00] or the TAK-2 and TAK-3 protocols from Al-Riyami and Paterson [ARP03].

We stick to this passive setting of [AR00]. This setting is restrictive compared to results for active adversaries. However this restriction can be partially removed because as shown by Katz and Yung in [KY03], it is possible to automatically transform a (key agreement) protocol that is secure in the passive setting into a protocol that is secure in the active setting. Thus in order to design a protocol that is secure in the active case, one can write a protocol that is secure against passive adversaries in the symbolic setting. Our result can be used to prove that this protocol is secure against passive adversaries in the computational setting. Then the Katz and Yung compiler can be used to generate a protocol that is secure against active adversaries in the computational setting.

Related work. This result follows the line of a recent trend in bridging the gap which separates the symbolic and computational views of cryptography. This work started with [AR00,AJ01] where only passive adversaries are considered.

Further works focussed on extending this result by considering the active setting and by adding cryptographic primitives. The active setting has been explored through a very rich and generic framework by Backes et al. in [BPW03] and subsequent papers. The work of Canetti and Herzog [CH04] uses a similar model but add the notion of universal composability. Micciancio and Warinschi later proposed another soundness result for the active case in [MW04b]. They consider a less general framework but in their model automatic verification of protocols in the symbolic model is possible through existing tools. This model was later extended in [CW05,JLM05] in order to remove some of the original limitations and to consider digital signatures.

In the passive setting, numerous cryptographic primitives have been studied. For example, exclusive or and ciphers have been considered in [BCK05], low entropy passwords which are subject to guessing attacks are studied in [ABW06], in [GvR06] probabilistic hash functions are used to prove soundness of symbolic hashes and in [ABHS05] a stronger variant of semantic security is used to consider symmetric encryption schemes in presence of key cycles. However we are not aware of any computational soundness result involving pairing-based protocols.

The concept and difficulties of considering pairings are close to those introduced by modular exponentiation. But computational soundness for this primitive has only been considered in a few works. In [GS05], a logic is used to verify protocols that use modular exponentiation and digital signature. However only two-party protocols are handled. Herzog presents in [Her04] an abstract model for Diffie-Hellman key exchange protocols but in this work the adversary is extended with the capability of applying arbitrary polynomial time functions.

Outline of this paper. The next section recalls the necessary definition for bilinear pairings and introduces BDDH security. Section 3 details our symbolic model: terms, deductibility and equivalence are defined in this setting. Then section 4 precises our computational setting by giving concrete semantics to symbolic terms. Our main soundness result is given in section 5: symbolic indistinguishability implies computational indistinguishability for secure cryptographic primitives. Section 6 illustrates this soundness result on some simple protocols using bilinear pairings. Finally a short conclusion is drawn in section 7.

2 Preliminaries for Pairing

In this section, we briefly recall the basics of bilinear pairings. The formal definition is given in section 4. Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two cyclic groups of same prime order q. Let g_1 be a generator of $\mathbb{G}_1$. We use multiplicative notations for both groups. A mapping e from $\mathbb{G}_1 \times \mathbb{G}_1$ to $\mathbb{G}_2$ is called a *cryptographic bilinear map* if it satisfies the three following properties.

- **Bilinearity**: $e(g_1^x, g_1^y) = e(g_1, g_1)^{xy}$ for any x, y in $\mathbb{Z}_q$.
- **Non-degeneracy**: $e(g_1, g_1)$ is a generator of $\mathbb{G}_2$ which is also denoted by g_2, i.e. $g_2 \neq 1_{\mathbb{G}_2}$.
- **Computable**: there exists an efficient algorithm to compute $e(u, v)$ for any u and v in $\mathbb{G}_1$.

Examples of cryptographic bilinear maps includes modified Weil pairing [BF01] and Tate pairing [BKLS02]: $\mathbb{G}_1$ is a group of points on an elliptic curve and $\mathbb{G}_2$ is a multiplicative subgroup of a finite field. The traditional notation for group $\mathbb{G}_1$ originates from elliptic curve groups and thus is additive however we stick to multiplicative notations in order to simplify our symbolic model of section 3.

The classical decisional security assumption for groups with pairing is the Bilinear Decisional Diffie-Hellman (BDDH) assumption. This assumption states it is difficult for an adversary that has access to three elements of $\mathbb{G}_1$, g_1^x, g_1^y and g_1^z to distinguish g_2^{xyz} from a randomly sampled element g_2^r of $\mathbb{G}_2$.

A simple key exchange protocol has been proposed by Joux in [Jou00]. This protocol is an extension of the classical Diffie-Hellman key exchange for three participants. Let A, B and C be the three participants. Each of them randomly samples a value in $\mathbb{Z}_q$ (denoted by x for A, by y for B and by z for C). Then the three following messages are exchanged:

$$(1)\ A \rightarrow B, C : g_1^x \qquad (2)\ B \rightarrow A, C : g_1^y \qquad (3)\ C \rightarrow A, B : g_1^z$$

The shared secret key is g_2^{xyz}, it is easy to check that A, B and C can compute this key by using the bilinear map e on the two messages that they have received. This protocol is safe in the passive setting provided that the BDDH assumption holds. No form of authentication is provided in this protocol, so it is trivially subject to man-in-the-middle attacks.

In the following sections, our objective is to provide a symbolic model for protocols using bilinear maps and to give a computational justification of this

model. We stick to the passive setting but as noted earlier this is not a real restriction thanks to the Katz and Yung compiler [KY03]. As usual the computational setting is parameterized by a security parameter η which can be thought of as the length of keys. Adversaries are probabilistic polynomial-time (in η) Turing machines. By convention, the adversaries considered in this paper are given access implicitly to as many fresh random coins as needed, as well as the complexity parameter η.

3 The Symbolic Setting

In this section, we introduce the symbolic view of cryptography: messages are represented as algebraic terms, the adversary's capabilities are defined by an entailment relation $\vdash$ and an observational equivalence $\cong$. This equivalence is an extension of the well-known Abadi-Rogaway logic to terms using symmetric encryption and pairing. The main difference with the original logic is that we introduce generator g_1 for the first group ($\mathbb{G}_1$), and generator g_2 for the second group ($\mathbb{G}_2$) as long as an infinite set of names representing exponents.

3.1 Terms and Deductibility

Let **Keys** and **Exponents** be two countable disjoint sets of symbols for keys and exponents. A power-free 3-monomial is a product of three *distinct* exponents and a power-free 3-polynomial is a linear combination of monomials using coefficients in $\mathbb{Z}$ (with no constant coefficient), hence $2x_1x_2x_3 + x_3x_4x_5$ is a power-free 3-polynomial but $x_1^2x_2$ and $x_1x_2x_3+1$ are not. We let **Poly** be the set of power-free 3-polynomials with variables in **Exponents** and coefficients in $\mathbb{Z}$. With a slight abuse of notation, we often refer to power-free 3-monomials as monomials and to power-free 3-polynomials as polynomials.

Let k, x and p be meta-variables over **Keys**, **Exponents** and **Poly** respectively. Polynomials can be exponentiated and the set **T** of terms is built using symbolic encryption and concatenation of keys, exponents and exponentiations:

$$msg ::= (msg, msg) \mid \{msg\}_{key} \mid x \mid key \mid g_1^x$$
$$key ::= k \mid g_2^p$$

Term (t_1, t_2) represents the pair composed of terms t_1 and t_2, $\{t\}_k$ represents (symmetric) encryption of term t using key k. $\{t\}_{g_2^p}$ represents encryption of term t using a key derived from g_2^p (there is an implicit application of a deterministic key extraction algorithm Kex which is detailed in the computational semantics), in this case g_2^p is said to occur at a *key position*. g_1^x and g_2^p represent modular exponentiation of g_1 (generator of the first group) and g_2 (generator of the second group) to the power of an exponent x in the first case and a polynomial p in the second case. An exponent x can be used exponentiated using g_1 or g_2 but can also be used as plaintext.

For any term t, $\mathsf{pol}\,(t)$ designates the set of polynomials that appear in t and $\mathsf{mon}\,(t)$ designates the set of monomials used by polynomials in $\mathsf{pol}\,(t)$.

Equality between polynomials is considered modulo the classical equational theory: associativity and commutativity for addition and multiplication, distributivity of multiplication over addition.

First we define a deduction relation $E \vdash t$ where E is a finite set of terms and t is a term. The intuitive meaning of $E \vdash t$ is that t can be deduced from E. The deductibility relation is an extension of the classical Dolev-Yao inference system [DY83]:

$$\frac{t \in E}{E \vdash t} \quad \frac{E \vdash (t_1, t_2)}{E \vdash t_1} \quad \frac{E \vdash (t_1, t_2)}{E \vdash t_2} \quad \frac{E \vdash \{t\}_k \quad E \vdash k}{E \vdash t}$$

Note that we did not consider composition rules like if t_1 and t_2 are deductible then (t_1, t_2) is also deductible. Indeed these rules are not necessary as deduction is only used to check whether some key can be deduced from a term. As keys are atomic, it is sufficient to consider the four previous rules. We add four new deduction rules in order to handle pairing. The three first rules correspond to the three possible ways to obtain an exponentiation g_2^{xyz} using the cryptographic bilinear map:

$$\frac{E \vdash x \quad E \vdash g_1^y \quad E \vdash g_1^z}{E \vdash g_2^{xyz}} \quad \frac{E \vdash x \quad E \vdash y \quad E \vdash g_1^z}{E \vdash g_2^{xyz}} \quad \frac{E \vdash x \quad E \vdash y \quad E \vdash z}{E \vdash g_2^{xyz}}$$

Note that these three rules correspond to "real" capacities of the adversary in the computational setting. In the first case, an adversary knowing g_1^y and g_1^z can use the bilinear map to produce g_2^{yz}. As he also knows x he can exponentiate g_2^{yz} to obtain g_2^{xyz}. In the second case, the adversary knows y so he can produce g_1^y and act as in the first case. Finally, the third case is also similar, the adversary can compute g_1^z and act as in the second case.

The fourth rule handles linear relations between polynomials.

$$\frac{E \vdash g_2^p \quad E \vdash g_2^q}{E \vdash g_2^{\lambda p+q}} \lambda \in \mathbb{Z}$$

In the computational world an adversary can multiply two group elements g_2^p and g_2^q in order to get g_2^{p+q}. He can also exponentiate a group element g_2^p and obtain $g_2^{\lambda p}$. Thus it is feasible for the adversary to produce $g_2^{\lambda p+q}$ from g_2^p and g_2^q. After adding these new deductions, the deductibility relation is still decidable.

Proposition 1. *Let t be a term and E be a finite set of terms. Then deductibility of t from E is decidable.*

3.2 Equivalence

Well-formed Terms. Equivalence is only defined for terms that make a correct use of the bilinear pairing. Such terms are called *well-formed* terms. Formally a term t is *well-formed* if for any distinct monomials m and m' in $\mathsf{mon}(t)$, m and m' do not have any common exponent.

Patterns. Patterns are used to characterize the information that can be extracted from a well-formed term. These patterns are close to those introduced in [AR00,MP05] but are extended in order to handle modular exponentiation. We introduce a new symbol $\Box$ representing a ciphertext that the adversary cannot decrypt. Moreover we consider that the encryption scheme is not necessarily key-concealing hence it could be possible for an adversary to tell that two ciphertexts have been produced using the same key. Let t be a term and K be a finite set of keys and elements of the second group g_2^p, then the pattern of t using K, $\mathsf{pat}\,(t, K)$ is inductively defined by:

$$\begin{array}{lll}
\mathsf{pat}\,((t_1, t_2), K) & = \big(\mathsf{pat}\,(t_1, K), \mathsf{pat}\,(t_2, K)\big) & \\
\mathsf{pat}\,(\{t'\}_{key}, K) & = \{\mathsf{pat}\,(t', K)\}_{key} & \text{if } key \in K \\
\mathsf{pat}\,(\{t'\}_{key}, K) & = \{\Box\}_{key} & \text{if } key \notin K \\
\mathsf{pat}\,(a, K) & = a & \text{for } a \text{ in } x,\ k,\ g_1^x \text{ and } g_2^p
\end{array}$$

The set K is used to store keys that are known by the adversary, for that purpose we use a function $K(t)$ that associates to each term t the set of keys k and exponentiations g_2^p which are subterms of t and deductible from t. Note that we consider k to be a subterm of $\{m\}_k$ and g_2^p to be a subterm of $\{m\}_{g_2^p}$.

$$K(t) = \{k | t \vdash k\} \cup \{g_2^p | t \vdash g_2^p \wedge g_2^p \text{ is a subterm of } t\}$$

We say that two well-formed terms t_1 and t_2 are *equivalent*, $t_1 \equiv t_2$, if they have the same pattern: $t_1 \equiv t_2$ if and only if $\mathsf{pat}\,(t_1, K(t_1)) = \mathsf{pat}\,(t_2, K(t_2))$. Intuitively patterns hide information that are encrypted with secure keys. Hence two terms have the same pattern if the information that can be extracted are the same, so it is impossible to distinguish these two terms.

Equivalence up to renaming. We allow renaming of keys in a similar way as [AR00] but renaming of polynomials is slightly more complex and is realized through linear relation preserving bijections. Let us illustrate this on the two following examples.

- Let t_1 be the term $(x_1, x_2, g_1^{x_3}, g_2^{x_4x_5x_6}, g_2^{x_1x_2x_3+x_4x_5x_6})$ and t_2 be the term $(x_1, x_2, g_1^{x_3}, g_2^{x_4x_5x_6}, g_2^{x_7x_8x_9})$. A renaming from polynomials of t_2 to polynomials of t_1 could be

  $$\{x_7x_8x_9 \mapsto x_1x_2x_3 + x_4x_5x_6\}$$

  However this renaming does not correctly preserve linear relations. In term t_1, $g_2^{x_1x_2x_3+x_4x_5x_6}$ can be obtained by multiplying $g_2^{x_4x_5x_6}$ with $g_2^{x_1x_2x_3}$ (which is obtained by applying the bilinear map to $g_1^{x_2}$ and $g_1^{x_3}$ and raising the result to the power x_1). In term t_2, $g_2^{x_7x_8x_9}$ cannot be obtained in a similar way.
- Let t_1 be the term $(g_2^{x_4x_5x_6}, g_2^{x_1x_2x_3+x_4x_5x_6})$ and t_2 be the term $(g_2^{x_4x_5x_6}, g_2^{x_7x_8x_9})$. The associated renaming is also:

  $$\{x_7x_8x_9 \mapsto x_1x_2x_3 + x_4x_5x_6\}$$

This correctly preserves linear relations as $g_2^{x_1x_2x_3+x_4x_5x_6}$ cannot be obtained from other parts of t_1 ($x_1x_2x_3+x_4x_5x_6$ is not involved in any linear relations) and $g_2^{x_7x_8x_9}$ cannot be obtained from other parts of t_2.

Two well-formed terms t_1 and t_2 are *equivalent up to renaming*, $t_1 \cong t_2$ if they are equivalent up to some renaming of keys and monomials.

$$\begin{aligned} t_1 \cong t_2 \text{ if and only if } & \exists \sigma_1 \text{ a permutation of } \mathbf{Keys} \\ & \exists \sigma_2 \text{ a linear relation preserving renaming from } t_2 \text{ to } t_1 \\ & \text{such that } t_1 \equiv t_2\sigma_1\sigma_2 \end{aligned}$$

We first define the set $dm(t)$ of *deductible monomials* from t, i.e. monomials that can be obtained using the bilinear map operation (this is a slight abuse of notation as the monomial itself can be not deductible its exponentiation using g_2 is deductible). A monomial $x_1x_2x_3$ from $\mathsf{mon}(t)$ is in $dm(t)$ if one of the following conditions hold:

- Either x_1, x_2 and x_3 are deductible from t,
- or x_1, x_2 and $g_1^{x_3}$ are deductible from t,
- or x_1, $g_1^{x_2}$ and $g_1^{x_3}$ are deductible from t.

Let t_2 and t_1 be two terms, a renaming σ from t_2 to t_1 is *linear relation preserving* if the same linear relations are verified between polynomials from t_2 and their image using σ. However monomials from $dm(t_2)$ cannot be renamed as they are linked to other parts of term t_2 due to the bilinear pairing. Formally, σ has to verify the following condition:

$$\forall p_1, ..., p_n \in \mathsf{pol}(t_2),\ \forall a_1, ..., a_n \in \mathbb{Z}, \quad \forall m_1, ..., m_{n'} \in dm(t_2),\ \forall b_1, ..., b_{n'} \in \mathbb{Z},$$
$$\sum_{i=1}^{n} a_i p_i = \sum_{j=1}^{n'} b_j m_j \Leftrightarrow \sum_{i=1}^{n} a_i (p_i\sigma) = \sum_{j=1}^{n'} b_j m_j$$

In this definition of equivalence, we have not considered renaming of **Exponents** to preserve simplicity but this can easily be added. Using this new definition, an interesting result is the decidability of equivalence up to renaming.

Proposition 2. *Let t_1 and t_2 be two well-formed terms. Equivalence up to renaming of t_1 and t_2 is decidable.*

In our symbolic model, we have two important restrictions: elements of the first group have the form g_1^x where x is an exponent and elements of the second group have the form g_2^p where p is a linear combination of monomials of order 3. The first restriction is useful as the Decisional Diffie-Hellman assumption is always false on the first group, knowing g_1^x, g_1^y it is easy to distinguish g_1^{xy} from a random group element by using the cryptographic bilinear map. The second restriction illustrates correct use of pairing. Pairing allows a safe use of keys using monomials of order 3, however this might not be the case for monomials of order different from 3.

3.3 Examples

Here we give some examples that illustrate the choices we made when defining the equivalence. These choices are motivated by the possibilities of adversaries in the computational setting. Unlike [AR00], our symbolic model does not include symbolic constants like 0 or 1 as plaintexts. However it is possible to encode these constants using for example two key names k_0 and k_1. Then 1 denotes k_1 and 0 denotes k_0 and instead of looking at equivalence between t and t', we look at equivalence between (k_0, k_1, t) and (k_0, k_1, t').

1. $\{0\}_k \cong \{1\}_k$. This example shows that symmetric encryption perfectly hides its plaintext.
2. $(\{0\}_k, \{0\}_k) \cong (\{0\}_k, \{1\}_k)$. Symmetric encryption also hides equalities among the underlying plaintexts. To achieve this, encryption has to be probabilistic. As modular exponentiation is deterministic, we cannot ask modular exponentiation to hide such relations.
3. $(g_1^{x_1}, g_1^{x_2}, g_1^{x_3}, g_2^{x_1x_2x_3}) \cong (g_1^{x_1}, g_1^{x_2}, g_1^{x_3}, g_2^{x'_1x'_2x'_3})$. This example illustrates security of the Joux protocol [Jou00] against passive adversaries. The adversary observes the unfolding of the protocol where three exponentiations are exchanged. These exponentiations allows the three participants to build a shared secret key $g_2^{x_1x_2x_3}$. Then the adversary cannot distinguish the shared key from a randomly sampled element of the second group $g_2^{x'_1x'_2x'_3}$ (as the order of the group is prime, $g_2^{x'_1x'_2x'_3}$ has a uniform distribution over elements of the second group).
   Moreover the symbolic setting can be used to verify that each participant is able to compute the shared key. For example the first participant generates exponent x_1 and receives $g_1^{x_2}$ and $g_1^{x_3}$ from the second and third participants. Using this knowledge, he is able to compute the shared secret key as $x_1, g_1^{x_2}, g_1^{x_3} \vdash g_2^{x_1x_2x_3}$.
4. $(g_1^{x_1}, g_1^{x_2}, g_1^{x_3}, \{0\}_{g_2^{x_1x_2x_3}}) \cong (g_1^{x_1}, g_1^{x_2}, g_1^{x_3}, \{1\}_{g_2^{x_1x_2x_3}})$. This example combines the Joux protocol with an exchange of secret information using the shared key. Thus in this example symmetric encryption and bilinear pairing are used simultaneously.

4 The Computational Setting

In this section, we formalize the mapping between symbolic terms and distributions of bit-strings. This mapping depends on the algorithm used to implement the two cryptographic primitives used in the symbolic setting: symmetric encryption and pairing.

4.1 Encryption Scheme

We recall the standard definition for symmetric encryption schemes. A symmetric encryption scheme $\mathcal{SE}$ is defined by three algorithms $\mathcal{KG}$, $\mathcal{E}$ and $\mathcal{D}$. The key

generation algorithm $\mathcal{KG}$ 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 and a key k and returns the encryption of s using k. The decryption algorithm $\mathcal{D}$ takes as input a bit-string c representing a ciphertext and a key k and outputs the corresponding plaintext. Given $k \leftarrow \mathcal{KG}(\eta)$, we have that for any bit-string s, if $c \leftarrow \mathcal{E}(k, s)$ then it is required that $\mathcal{D}(c) = s$.

In order to characterize security of a symmetric encryption scheme, we use the classical IND-CPA (indistinguishability against chosen plaintext attacks) notion [GM82].

IND-CPA *security.* In this paper we use schemes that satisfy length-concealing semantic security [1]. 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:

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

Encryption scheme $\mathcal{SE}$ is IND-CPA secure if the advantage of any adversary $\mathcal{A}$ is negligible in η. The difference with the standard notion of semantic security is that we require that the scheme hides 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.

4.2 Pairing

A *bilinear pairing instance generator* is defined as a probabilistic polynomial-time algorithm IG which given a security parameter η outputs a tuple $(q, \mathbb{G}_1, \mathbb{G}_2, g_1, e)$ composed by two groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of prime order q, a generator g_1 of $\mathbb{G}_1$ and a cryptographic bilinear map e between $\mathbb{G}_1$ and $\mathbb{G}_2$. A generator g_2 of group $\mathbb{G}_2$ is obtained by applying e to (g_1, g_1).

BDDH *security.* An instance generator IG satisfies the *Bilinear Decisional Diffie-Hellman assumption*, BDDH, iff for any probabilistic polynomial-time adversary $\mathcal{A}$, the advantage of $\mathcal{A}$ against BDDH, $\mathrm{Adv}^{\mathsf{BDDH}}_{\mathcal{A},IG}$, defined above is negligible in η.

$$\begin{aligned}\mathrm{Adv}^{\mathsf{BDDH}}_{\mathcal{A},IG}(\eta) = \mathbb{P}\left[\begin{array}{c}(q, \mathbb{G}_1, \mathbb{G}_2, g_1, e) \leftarrow IG(\eta) \\ x, y, z \leftarrow \mathbb{Z}_q\end{array}, \ \mathcal{A}(g_1, g_1^x, g_1^y, g_1^z, g_2^{xyz}) = 1\right] \\ -\mathbb{P}\left[\begin{array}{c}(q, \mathbb{G}_1, \mathbb{G}_2, g_1, e) \leftarrow IG(\eta) \\ x, y, z, r \leftarrow \mathbb{Z}_q\end{array}, \ \mathcal{A}(g_1, g_1^x, g_1^y, g_1^z, g_2^r) = 1\right]\end{aligned}$$

This means that an adversary that is given g_1^x, g_1^y and g_1^z can only make the difference between g_2^{xyz} and a random group element with negligible probability.

[1] Such schemes can only exist if the maximum length of plaintexts is bounded, however we do not take this into account in this paper.

4.3 Computational Semantics of Terms

Computational semantics depend on a symmetric encryption scheme $\mathcal{SE} = (\mathcal{KG}, \mathcal{E}, \mathcal{D})$ and of an instance generator IG. In order to transform elements of the second group into keys usable for $\mathcal{SE}$, we assume the existence of a key extractor [CFGP05] algorithm Kex (usually a universal hash function used with the entropy smoothing theorem). We suppose that the distribution of keys generated by $\mathcal{KG}$ is equal to the distribution obtained by applying Kex to a random element of $\mathbb{G}_2$ (generated as second group by IG). We associate to each symbolic term t a distribution of bit-strings $[\![t]\!]_{\mathcal{SE},IG}$ that depends on the security parameter η. This distribution is defined by the following random algorithm:

1. Algorithm IG is used to generate two paired groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of order q and of generators $\widehat{g_1}$ and $\widehat{g_2}$. For each key k from t, a value $\widehat{k}$ is randomly drawn using $\mathcal{KG}$. For each exponent x, a value $\widehat{x}$ is randomly sampled in $\mathbb{Z}_q$ equipped with the uniform distribution.
2. Then the bit-string evaluation of term t is computed recursively on the structure of t:
   - If t is a pair (t_1, t_2), the algorithm is applied recursively on t_1 holding bs_1 and on t_2 holding bs_2. The output of the algorithm is the concatenation of bs_1 and bs_2.
   - If t is an encryption $\{t'\}_k$, the algorithm is applied recursively on t' holding bs' and on k holding bs_k. The output of the algorithm is $\mathcal{E}(bs', bs_k)$.
   - If t is an encryption $\{t'\}_{g_2^p}$, the algorithm is applied recursively on t' holding bs' and on g_2^p holding bs_k. The output of the algorithm is $\mathcal{E}\left(bs', \mathsf{Kex}(bs_k)\right)$.
   - If t is an exponentiation g_b^p for b in $[0, 1]$ (p is an exponent if $b = 0$), then the algorithm computes the value n of p in $\mathbb{Z}_q$, and the exponentiation of $\widehat{g_b}$ to the power of n is returned.
   - If t is a key k or an exponent x, then the value $\widehat{t}$ is returned.

5 Soundness of the Symbolic Model

In this section we prove that the extension of the Abadi-Rogaway logic given in section 3 is computationally sound when implemented using an IND-CPA encryption scheme and using an instance generator satisfying BDDH: if two terms are equivalent up to renaming in the symbolic setting, their evaluations (given by the computational semantics of section 4) are computationally indistinguishable.

Acyclicity Restrictions. The importance of key cycles was already described in [AR00]. In general IND-CPA is not sufficient to prove any soundness result in presence of key cycles, thus as in numerous previous work we forbid the symbolic terms to contain such cycles. (Another possibility to handle key cycles is to consider stronger computational requirements like Key Dependent Message – KDM – security as done in [ABHS05].) For any well-formed term t, let $kp(t)$ be the set of polynomials p such that g_2^p occurs at a key position in t and g_2^p is not deductible from t. Let $dm'(t)$ be the set of monomials $x_1x_2x_3$ such that:

– x_1, x_2 and x_3 occur as plaintexts in t.
– x_1 and x_2 occur as plaintexts in t and $g_1^{x_3}$ also appears in t.
– x_1 occurs as plaintext in t and $g_1^{x_2}$ and $g_1^{x_3}$ also appear in t.

A well-formed term t is *acyclic* if the two following restrictions are verified.

– For any p in $kp(t)$, p is linearly independent from any other polynomials from $\mathsf{pol}\,(t)$ and from monomials from $dm'(t)$, i.e. if $\mathsf{pol}\,(t) = \{p, p_1, ..., p_n\}$ and $dm'(t) = \{m_1, ..., m_{n'}\}$ then there does not exist any integers a, a_1 to a_n and b_1 to $b_{n'}$ such that $a \neq 0$ and:

$$a.p = \sum_{i=1}^{n} a_i p_i + \sum_{j=1}^{n'} b_j m_j$$

– Let t' be term t where each g_2^p for p in $kp(t)$ has been replaced by a fresh key name k. Then there exists an order $\prec$ among keys such that for any subterm $\{t'\}_k$ of t, k' can only occur in t' if $k' \prec k$.

5.1 Soundness Result

Indistinguishable Distributions. Before giving our soundness result, we recall the usual notion of indistinguishable distributions. Intuitively, two distributions D_1 and D_2 are computationally indistinguishable if for any adversary $\mathcal{A}$, the probability for $\mathcal{A}$ to detect the difference between a randomly sampled element of D_1 and a randomly sampled element of D_2 is negligible.

Definition 1. *Let D_1 and D_2 be two distributions (that depend on η). The advantage of an adversary $\mathcal{A}$ in distinguishing D_1 and D_2 is defined by:*

$$\mathrm{Adv}_{\mathcal{A}}^{D_1,D_2} = \mathbb{P}\left[x \leftarrow D_1(\eta)\ ;\ \mathcal{A}(x) = 1\right] - \mathbb{P}\left[x \leftarrow D_2(\eta)\ ;\ \mathcal{A}(x) = 1\right]$$

Distributions D_1 and D_2 are computationally indistinguishable*, $D_1 \approx D_2$, if the advantage for any adversary $\mathcal{A}$ in distinguishing D_1 and D_2 is negligible.*

Then our main soundness result states that distributions related to equivalent terms are computationally indistinguishable.

Proposition 3. *Let t_0 and t_1 be two acyclic well-formed terms, such that $t_0 \cong t_1$. Let $\mathcal{SE}$ be a symmetric encryption scheme that is secure for* IND-CPA *and IG be an instance generator satisfying* BDDH*, then $[\![t_0]\!]_{\mathcal{SE},IG} \approx [\![t_1]\!]_{\mathcal{SE},IG}$.*

This result states soundness of symbolic equivalence in the computational world. However, the reciprocal (i.e. completeness) is in general false. There are two main problems that prevent completeness. First, the symmetric encryption scheme may allow decryption with the wrong key and output a random bit-string in that case. Then the distributions related to terms $(\{x\}_k, k)$ and $(\{x\}_k, k')$ can be computationally indistinguishable, even though these two terms do not have the same pattern. This can be solved by adding a confusion freeness hypothesis

for the symmetric encryption scheme [MW04a,AJ01]. The second problem is that the symmetric encryption scheme can satisfy key concealing (this is ensured by type 0 security in [AR00]). Then the distributions related to terms $(\{0\}_k, \{0\}_{k'})$ and $(\{0\}_k, \{0\}_k)$ are computationally indistinguishable but these terms are not equivalent even with renaming. To solve this, one can either ask the encryption scheme to be key revealing or modify the pattern definition in order to hide the key name (but the encryption scheme has to be key concealing in order to prove soundness).

The previous proposition considers the case of equivalence and is typically used to verify security of key-exchange protocols. In the next proposition, we are interested in completeness for deductibility. We prove even more than completeness: if t is deductible from E then there exists an algorithm which is able to build an evaluation of t from an evaluation of E with probability 1. This result can be used to verify that a key-agreement protocol can really be implemented in the computational setting: we first check that the shared key is deductible from the knowledge of any participants in the symbolic setting, then applying the following proposition tells us that there exists an efficient algorithm to obtain the shared key from the participant knowledge in the computational setting.

Proposition 4. *Let E be a finite set of terms t_1 to t_n and t be a term that does not use any encryption (e.g. a modular exponentiation). If $E \vdash t$ then there exists a polynomial-time algorithm A such that $A\,(\llbracket (t_1, ..., t_n) \rrbracket_{\mathcal{SE},IG})$ outputs the evaluation of t using values for exponents and keys that have been generated to compute $\llbracket (t_1, ..., t_n) \rrbracket_{\mathcal{SE},IG}$, i.e.:*

$$(bs, bs') \leftarrow \llbracket ((t_1, ..., t_n), t) \rrbracket_{\mathcal{SE},IG} \ : \ A(bs) = bs'$$

Note that it is not necessary for terms to be well-formed or acyclic in this proposition.

6 Examples of Application

Now we illustrate how proposition 3 can be used to prove a key-exchange protocol secure in the computational world.

Our notion of security is strong secrecy of the shared key in the passive setting: the adversary gets to observe messages exchanged between the participants and has to distinguish the shared key from a random group element. In the symbolic world, let us suppose that the exchanged terms where t_1 to t_n and that the shared key is g_2^p, then security in the symbolic setting holds if:

$$(t_1, ..., t_n, g_2^p) \approx (t_1, ..., t_n, g_2^{r_1 r_2 r_3})$$

Where r_1, r_2 and r_3 are three fresh exponent names. It is then possible to apply proposition 3 in order to prove security in the computational setting.

We are also interested in executability of key exchange protocols. A protocol is executable if it is feasible for any participant to compute the shared key from

his knowledge. Let us still suppose that the exchanged terms were t_1 to t_n and that the shared key is g_2^p, moreover let $x_i^1, ..., x_i^k$ be the exponents which are generated by the i^{th} participant. The protocol is executable in the symbolic setting if for any i,

$$t_1, ..., t_n, x_i^1, ..., x_i^1 \vdash g_2^p$$

Executability in the computational world can easily be obtained from here by applying proposition 4.

6.1 Joux Protocol

The Joux protocol has been described in section 2. In an execution of this protocol, three messages are sent, corresponding to terms $g_1^{x_1}$, $g_1^{x_2}$ and $g_1^{x_3}$ then the shared key is $g_2^{x_1x_2x_3}$. Strong secrecy for this key-exchange protocol has been given as an example for our symbolic equivalence notion:

$$(g_1^{x_1}, g_1^{x_2}, g_1^{x_3}, g_2^{x_1x_2x_3}) \cong (g_1^{x_1}, g_1^{x_2}, g_1^{x_3}, g_2^{x'_1x'_2x'_3})$$

Proposition 3 can be applied to show that this protocol is secure in the computational setting if the BDDH assumption holds.

We also verify that this protocol is executable. In the symbolic setting this is the case as we have the following deductibility relation:

$$x_1, g_1^{x_2}, g_1^{x_3} \vdash g_2^{x_1x_2x_3}$$

And similar relation holds when permuting the roles of x_1 and x_2 and of x_1 and x_3. Thus proposition 4 proves that there exists an efficient algorithm in the computational setting which allows each participant to compute his shared secret key.

6.2 TAK-2 and TAK-3 Protocols

The TAK-2 and TAK-3 protocols are two variants of the Joux protocol which were proposed by Al-Riyami and Paterson in [ARP03]. TAK-1 and TAK-2 are tripartite key-exchange protocols which work in the same way, the only difference lies in the shared key. These protocols uses certificates to provide authentication. However as we are only interested in indistinguishability of the shared key, we use a simplified version of the protocol. Let A, B and C be three participants:

$$\begin{array}{lll} (1)\ A & \rightarrow & B, C\ :\ (g_1^{x_1}, g_1^{y_1}) \\ (2)\ B & \rightarrow & A, C\ :\ (g_1^{x_2}, g_1^{y_2}) \\ (3)\ C & \rightarrow & A, B\ :\ (g_1^{x_3}, g_1^{y_3}) \end{array}$$

In TAK-2, the shared key is $g_2^{x_1x_2y_3+x_1y_2x_3+y_1x_2x_3}$. In TAK-3, $g_2^{x_1y_2y_3+y_1x_2y_3+y_1y_2x_3}$ is used as shared key. Our simplified version of the two protocols are quite close as we do not make any difference between short-term secrets (y_1, y_2 and y_3) and long-term secrets (x_1, x_2 and x_3). Thus in our setting it is sufficient to analyse one of the protocol, TAK-2 for example.

Security. In the symbolic setting, strong secrecy of the key generated by the TAK-2 protocol comes from the following equivalence (up to renaming). Note that the two equivalent terms are well-formed and trivially acyclic:

$$\begin{gathered}\left(g_1^{x_1}, g_1^{y_1}, g_1^{x_2}, g_1^{y_2}, g_1^{x_3}, g_1^{y_3}, g_2^{x_1x_2y_3+x_1y_2x_3+y_1x_2x_3}\right) \\ \cong \\ \left(g_1^{x_1}, g_1^{y_1}, g_1^{x_2}, g_1^{y_2}, g_1^{x_3}, g_1^{y_3}, g_2^{x'_1x'_2x'_3}\right)\end{gathered}$$

Hence by using proposition 3, we obtain that in the computational setting an adversary that has access to values for $g_1^{x_1}$, $g_1^{y_1}$, $g_1^{x_2}$, $g_1^{y_2}$, $g_1^{x_3}$ and $g_1^{y_3}$ cannot distinguish the shared key $g_2^{x_1x_2y_3+x_1y_2x_3+y_1x_2x_3}$ from a random group element, so the adversary is not able to obtain a single bit of information on the shared key.

Executability. We also verify executability of the protocol. By symmetry we consider the case of A. A generates two exponents x_1 and y_1 and receives two messages corresponding to terms $(g_1^{x_2}, g_1^{y_2})$ and $(g_1^{x_3}, g_1^{y_3})$. Hence executability in the symbolic setting is a consequence of the following deduction:

$$x_1, y_1, g_1^{x_2}, g_1^{y_2}, g_1^{x_3}, g_1^{y_3} \vdash g_2^{x_1x_2y_3+x_1y_2x_3+y_1x_2x_3}$$

Thus proposition 4 proves that there exists an efficient algorithm in the computational setting which allows participant A to compute his shared secret key. The same thing holds for B and C.

7 Conclusions and Future Work

We have proposed a first symbolic model to analyse cryptographic protocols which use a bilinear pairing. This model can be used to verify security of well-known key-exchange protocols using pairing like Joux protocol or the TAK-2 and TAK-3 protocol. Moreover our symbolic model consists in an extension of Abadi-Rogaway logic which is computationally sound provided that the encryption scheme and the pairing satisfy classical requirements from provable security. A direct consequence of this soundness result is that the Joux, TAK-2 and TAK-3 protocol are also secure in the computational setting.

This paper only consider passive adversaries, an obvious line for future work is to extend the results to deal with active adversaries. Another interesting follow-up would be to investigate completeness of the extended version of Abadi-Rogaway logic as in [MW04a]. However this would require either to tighten the symbolic model or to use stronger versions of the computational requirements IND-CPA and BDDH.

References

[ABHS05] Pedro Adão, Gergei Bana, Jonathan Herzog, and Andre Scedrov. Soundness of formal encryption in the presence of key-cycles. In *Proceedings of the*

10th European Symposium on Research in Computer Security (ESORICS), volume 3679 of *Lecture Notes in Computer Science*, pages 374–396. Springer-Verlag, 2005.

[ABW06] Martín Abadi, Mathieu Baudet, and Bogdan Warinschi. Guessing attacks and the computational soundness of static equivalence. In *Proceedings of the 9th International Conference on Foundations of Software Science and Computation Structures (FoSSaCS'06)*, volume 3921 of *Lecture Notes in Computer Science*, pages 398–412, Vienna, Austria, 2006. Springer.

[AJ01] M. Abadi and J. Jürgens. Formal eavesdropping and its computational interpretation. In *Proceedings of the Fourth International Symposium on Theoretical Aspects of Computer Software*. Springer, 2001.

[AR00] 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 TCS2000)*, Sendai, Japan, 2000. Springer-Verlag.

[ARP03] Sattam S. Al-Riyami and Kenneth G. Paterson. Tripartite authenticated key agreement protocols from pairings. In *Cryptography and Coding, 9th IMA International Conference, Cirencester, UK, December 16-18, 2003, Proceedings*, volume 2898 of *Lecture Notes in Computer Science*, pages 332–359. Springer, 2003.

[BCK05] Mathieu Baudet, Véronique Cortier, and Steve Kremer. Computationally sound implementations of equational theories against passive adversaries. In *Proceedings of the 32nd International Colloquium on Automata, Languages and Programming (ICALP'05)*, volume 3580 of *Lecture Notes in Computer Science*, pages 652–663. Springer, 2005.

[BDS03] Rana Barua, Ratna Dutta, and Palash Sarkar. Extending joux's protocol to multi party key agreement (extended abstract). In *Proceedings of INDOCRYPT 2003, 4th International Conference on Cryptology in India*, volume 2904 of *Lecture Notes in Computer Science*, pages 205–217. Springer, 2003.

[BF01] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In *Proceedings of CRYPTO 2001, 21st Annual International Cryptology Conference*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer, 2001.

[BKLS02] P. Barreto, H. Kim, B. Lynn, and M. Scott. Efficient algorithms for pairing-based cryptosystcms. In *Proceedings of Crypro 2002*, 2002.

[BPW03] M. Backes, B. Pfitzmann, and M. Waidner. A composable cryptographic library with nested operations. In *Proceedings of the Tenth ACM conference on Computer and Communication Security*, pages 220–230. ACM Press, 2003.

[CFGP05] Olivier Chevassut, Pierre-Alain Fouque, Pierrick Gaudry, and David Pointcheval. Key derivation and randomness extraction. Technical Report 2005/061, Cryptology ePrint Archive, 2005. http://eprint.iacr.org/.

[CH04] R. Canetti and J. Herzog. Universally composable symbolic analysis of cryptographic protocols (the case of encryption-based mutual authentication and key exchange). Cryptology ePrint Archive, Report 2004/334, 2004.

[CW05] V. Cortier and B. Warinschi. Computationally Sound, Automated Proofs for Security Protocols. In *Proceeding of the Fourtennth European Symposium on Programming (ESOP'05)*, pages 157–171. Springer, 2005.

[DBS04] Ratna Dutta, Rana Barua, and Palash Sarkar. Pairing-based cryptographic protocols : A survey. Cryptology ePrint Archive, Report 2004/064, 2004. http://eprint.iacr.org/.

[DY83] D. Dolev and A. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 1983.

[GM82] S. Goldwasser and S. Micali. Probabilistic encryption & how to play mental poker keeping secret all partial information. In *Proceedings of the fourteenth annual ACM symposium on Theory of computing (STOC 1982)*. ACM Press, 1982.

[GS05] P. Gupta and V. Shmatikov. Towards computationally sound symbolic analysis of key exchange protocols. In *Proceedings of the Third ACM Workshop on Formal Methods in Security Engineering: From Specifications to Code*, 2005.

[GvR06] Flavio D. Garcia and Peter van Rossum. Sound computational interpretation of symbolic hashes in the standard model. In *Proceedings of Advances in Information and Computer Security. First International Workshop on Security, IWSEC 2006*, volume 4266 of *Lecture Notes in Computer Science*, pages 33–47. Springer, 2006.

[Her04] Jonathan Herzog. *Computational soundness for standard assumptions of formal cryptography*. PhD thesis, MIT, 2004.

[JLM05] R. Janvier, Y. Lakhnech, and L. Mazaré. Completing the picture: Soundness of formal encryption in the presence of active adversaries. In *Proceedings of the Fourteenth European Symposium on Programming (ESOP'05)*, pages 172–185. Springer, 2005.

[Jou00] Antoine Joux. A one round protocol for tripartite diffie-hellman. In *ANTS-IV: Proceedings of the 4th International Symposium on Algorithmic Number Theory*, pages 385–394, London, UK, 2000. Springer-Verlag.

[KY03] J. Katz and M. Yung. Scalable protocols for authenticated group key exchange. In *Proceedings of the Twenty-Third Annual International Cryptology Conference (CRYPTO 2003)*. Springer, 2003.

[MP05] D. Micciancio and S. Panjwani. Adaptive security of symbolic encryption. In *Proceedings of the Theory of cryptography conference (TCC 2005)*. Springer-Verlag, 2005.

[MW04a] D. Micciancio and B. Warinschi. Completeness theorems for the abadi-rogaway logic of encrypted expressions. *Journal of Computer Security*, 2004. Preliminary version in WITS 2002.

[MW04b] D. Micciancio and B. Warinschi. Soundness of formal encryption in the presence of active adversaries. In *Proceedings of the Theory of Cryptography Conference (TCC 2004)*, pages 133–151. Springer, 2004.