Verification of anonymity properties in e-voting protocols

Stefan Ciobaca
Supervisors: Stéphanie Delaune, Steve Kremer

Project team SECSI,
Laboratoire Spécification et Vérification,
ENS Cachan & CNRS

August 24, 2008

The general context

The language of this thesis is English because learning French is still work in progress for me.

In this thesis, we have approached two problems related to the automatic verification of anonymity-like properties in the passive case.

The first problem concerns the security of the composition of two protocols under a shared secret. We ask ourselves under which circumstances is the parallel composition of two independently secure protocols secure, assuming the protocols share a secret.

This question is important, because a good answer allows a verifier to construct modular proofs of security. Moreover, shared secrets between different network services are fairly common (for example, an user has the same password for email and for online banking). It is acknowledged that, in general, composition is an important problem [1, 2, 3], but to our knowledge, we are the first to ask this question.

After answering the first problem, we approach the construction of decision procedures for anonymity-like properties. We study the passive case, where such properties can be expressed in terms of the power of the attacker to distinguish between two sequences of messages.

This problem is important because hand-verifying security properties is error-prone. Decision procedures have already been proposed in the literature for various cryptographic primitives [4, 5, 6]. However, none of these approaches works for deciding anonymity-like properties if the cryptographic primitive of trapdoor commitment is used. We are currently actively studying this problem and we present some work in progress that includes an algorithm for it.

The proposed contribution

For the composition problem, we prove that in the passive case and when the two protocols properly use probabilistic cryptographic primitives, the parallel composition of two independently secure cryptographic protocols which share a secret is secure.

For the second problem studied, we present a generic semi-decision procedure, we describe intuitively why it works, and state some conjectures about its correctness. We show that the procedure terminates for the case of trapdoor commitment.

The arguments in favor of its validity

In the case of the parallel composition of two protocols, we provide a theorem that states our result. For the theorem to hold, we demand that the messages exchanged by the participants are well-formed. If the participants improperly use the probabilistic cryptographic primitives, the result no longer holds.

For the semi-decision procedure, we intuitively describe why it is correct and formally state some conjectures that still need to be proved. We show that, for the case of trapdoor commitment, the procedure terminates. Also, we have created a prototype implementation of the algorithms involved. It is worth noting that our semi-decision procedure terminates for many other classes of cryptographic primitives. For example, our algorithm is a generalization of the results in [4].

The conclusions and the perspectives

We have proven a composition result and presented a work-in-progress semi-decision procedure for anonymity-like properties in the passive case.

Although in the proof for the composition result we restrict ourselves to symmetric encryption, we can generalize the result for different types of probabilistic cryptographic primitives, like asymmetric encryption and probabilistic hash functions.

The semi-decision procedure is implementable for a large class of cryptographic primitives and it terminates on interesting examples. Obviously, we still need to formalize the proof of its correctness. It would be interesting in characterizing the class of cryptographic primitives for which it terminates.

Both of the problems need to be studied in the active case as well.

Another good direction for further study is the treatment of associative-commutative operators in the semi-decision procedure.

1 Introduction

A cryptographic protocol is a small set of rules followed by agents on a network to ensure secure communication. Cryptographic protocols are ubiquitous in our society. They are widely deployed on the Internet, to ensure secrecy and authenticity.

In the formal (also called logical) model, we assume that the encryption primitives are ideal (i.e. the attacker does not know anything about encrypted data, unless he knows the key). We also assume the attacker has full control of the network: he can intercept, modify, duplicate, combine, delay, create etc. any messages.

Dolev and Yao [7] were the first to realize that the security of cryptographic protocols must also be analyzed from a logical point of view (i.e. assuming that the *logic* behind the protocols is attacked, not the cryptographic primitives such as encryption).

1.1 Anonymity

Many models for checking security properties of cryptographic protocols are *trace-based.* In trace-based models, the possible *executions* of a protocol are described by *traces* and the security properties are expressed in terms of reachability: a certain data is secret iff it is not reachable in any trace.

An important application of security protocols is e-voting, where anonymity is often a desirable property. Anonymity means that an attacker cannot link a certain voter to his vote. However, anonymity cannot be expressed as a trace property.

Consider the example of a protocol where an agent A wants to start a communication session with another agent B. To start with, A sends his identity to B, encrypting it with the public key of the receiver:

$$A \to B : aenc(A, pub(B))$$

In the above notation, $pub(B)$ denotes the public-key of B and $aenc$ is the (asymmetric) encryption function. In the above protocol, the identity of A is a secret (not reachable by any trace), because we assume that the encryption cannot be broken and that the private key of B remains private. However, a third party can still identify A by creating messages of the form $aenc(I, pub(B))$ for all possible identities I and comparing them against the message transmitted by A. Therefore, the anonymity of A is not observed.

There are many possible formal definitions of anonymity, but it is usually expressed in terms of the possibility of an attacker to distinguish between two different variants of the protocol. In the passive case (when the attacker only has the power to listen to messages, but not to create, modify of delete messages), anonymity can be expressed in terms of the power of the intruder to distinguish between two sequences of messages.

1.2 Contributions

In this thesis, we concentrate on verifying anonymity-like properties of protocols in the passive case. The passive case is important for a number of reasons:

- firstly, it is a prerequisite to solving the active case, where the intruder is given more power

- secondly, the cost of actively attacking a network might be prohibitive (for example, it might require very advanced hardware), so we only need to analyze the passive case

We study anonymity properties in the formalism of the applied-pi calculus [8]. In the applied-pi calculus, a sequence of messages exchanged over the network is organized in a *frame*. The power of an intruder to distinguish between two frames is formalized by the notion of *static equivalence*. In section 2, we formally define all of the notions we use.

The contributions of this thesis are divided into two parts.

- *Parallel composition under a shared secret*

  The first part, which constitutes section 3, consists of a *composition result*.

  In this part, we study the security guarantees of the parallel composition of two protocols which share a secret. This situation happens when, for example, an user chooses the same password for both an email service and for an online banking service.

  In the passive case and under certain reasonable assumptions, we prove that if the two secure protocols make use of probabilistic encryption primitives, their parallel composition under a shared secret is also secure, assuming the security guarantees are expressed as the static equivalence of some frames.

  This composition result may help with the automatic verification by allowing a verifier to split its work in smaller problems, increasing its efficiency and allowing to create more modular proves of security.

- *Decidability of static equivalence*

  The second part, constituting section 4, consists of a work-in-progress algorithm for *static equivalence*.

  Static equivalence is important because security properties like anonymity can be expressed formally in terms of the static equivalence of two frames. There are some decidability results for static equivalence [4, 5, 6], but none of them treat the case where trapdoor commitment is used as a cryptographic primitive.

  Trapdoor commitment can be used in e-voting protocols to help provide coercion resistance. Motivated by this cryptographic primitive, we develop a generic semi-decision procedure for checking static equivalence in the case where the equational theory that axiomizes the cryptographic primitives can be expressed as a convergent rewriting system.

Section 6 contains the conclusions and possible directions for future work.

2 Preliminaries

2.1 Messages

Messages sent over the network are represented as terms over some *signature* which associates to each functional symbol its *arity*, over a countably infinite set of *names* $\mathcal{N}$ (intuitively representing keys, nonces and other data) and over a countably infinite set of *variables* $\mathcal{X}$ disjoint from $\mathcal{N}$.

If $\mathcal{X}_0 \subseteq \mathcal{X}$, $\mathcal{N}_0 \subseteq \mathcal{N}$ and Σ_0 is some signature, we use the notation $\mathcal{T}(\Sigma_0, \mathcal{N}_0, \mathcal{X}_0)$ to denote the set of terms over the signature Σ_0, over the set of names $\mathcal{N}_0$ and over the set of variables $\mathcal{X}_0$.

A term is ground if it does not contain variables. The set of ground terms over the signature Σ_0 is $\mathcal{T}(\Sigma_0, \mathcal{N}, \emptyset)$. If the signature Σ_0 is obvious from the context, we write $\mathcal{T}(\mathcal{N}_0, \mathcal{X}_0)$ instead of $\mathcal{T}(\Sigma_0, \mathcal{N}_0, \emptyset)$ and $\mathcal{T}(\mathcal{N}_0)$ instead of $\mathcal{T}(\mathcal{N}_0, \emptyset)$.

Example Consider the signature:

$$\Sigma_{senc} = \{pair/2, fst/1, snd/1, senc/3, sdec/2\} \cup \{c_i/0\}_{0 \leq i \leq N}$$

The function symbols *senc* and *sdec* represent the probabilistic symmetric encryption and respectively decryption algorithm: $senc(a, b, c)$, also written $senc^a(b, c)$ for convenience, represents the encryption of the data b by key c and randomness a. The function symbols *pair*, *fst* and *snd* are are used to group and ungroup terms. The arity 0 function symbols c_i are any number of constants.

2.2 Equational theories

Syntactic equality of terms does not suffice to study a cryptographic protocol. For example, assuming 0 is a constant in Σ_{senc} and k a name in $\mathcal{N}$, the term 0 is not syntactically equal to $sdec(senc^r(0, k), k)$, even though they should intuitively represent the same data.

To overcome this problem, we equip the signature Σ_{senc} with an equational theory, that is, with an equivalence relation that is closed under substitution of terms for variables.

In this paper, we always suppose that an equational theory E is given by a binary relation $\sim_E$, consisting of finite set of tuples (u, v), where $u \in \mathcal{T}(\emptyset, \mathcal{X})$ and $v \in \mathcal{T}(\emptyset, \mathcal{X})$. Then equality modulo E, written $=_E$ is the smallest equivalence relation that contains $\sim_E$ and is closed under substitutions of terms for variables and application of contexts. Notice that such equational theories are always closed by substitution of arbitrary terms for names.

To the signature Σ_{senc}, we associate the equation theory E_{senc} induced by the following binary relation $\sim_{E_{senc}}$:

$$\begin{aligned} &sdec(senc^z(x, y), y) = x \\ &fst(pair(x, y)) = x \\ &snd(pair(x, y)) = y \end{aligned}$$

The term rewriting system R_{senc} obtained by orienting the equations $\sim_{E_{senc}}$ from left to right is convergent. For any convergent term rewriting system R, we associate to each term

U its unique normal form $U{\downarrow_R}$. If R is obvious from the context, we also write $U{\downarrow}$ instead of $U{\downarrow_R}$.

Example. If $T = sdec(senc^n(m,k),k)$ is a term over Σ_{senc}, with $n,m,k \in \mathcal{N}$, then its normal form with respect to R_{senc} is $T{\downarrow} = m$.

We say that a convergent term rewriting system R implements an equational theory E iff, for any terms U and V, $U =_E V$ iff $U{\downarrow_R} = V{\downarrow_R}$.

2.3 Substitutions

A *substitution* σ is a partial function from $\mathcal{N} \cup \mathcal{X}$ to $\mathcal{T}(\mathcal{N},\mathcal{X})$. We use the following notation to explicitly write a substitution: $\sigma = \{T_1/u_1, \dots, T_k/u_k\}$, if $dom(\sigma) = \{u_1,\dots,u_k\} \subseteq \mathcal{X} \cup \mathcal{N}$ and $\sigma(u_i) = T_i$.

Applying a substitution σ to a term T yields the term $T\sigma$, which is obtained from T by replacing every occurrence of a name or variable $u \in dom(\sigma)$ by $\sigma(u)$. We extend application of substitutions to sets and tuples in the following way: $\{T_1,\dots,T_k\}\sigma = \{T_1\sigma,\dots,T_k\sigma\}$ and $(T_1,\dots,T_k)\sigma = (T_1\sigma,\dots,T_k\sigma)$.

If $dom(\sigma) \subseteq \mathcal{X}$, we call σ a *variable substitution.*

A renaming of names is a bijective substitution from a subset of $\mathcal{N}$ to a subset of $\mathcal{N}$. A renaming of variables is defined similarly.

2.4 Frames

The sequence of messages observed by an attacker in the applied π-calculus is represented as a *frame*. A frame is nothing more than that sequence of ground terms, together with the set of names that were freshly generated (names which the attacker did not previously know).

Formally, a frame $\phi = \nu\tilde{n}.\sigma$ consists of a set $\tilde{n}$ of restricted (freshly generated) names and a variable substitution $\sigma = \{M_1/x_1, M_2/x_2, \dots, M_n/x_n\}$, where $M_i \in \mathcal{T}(\mathcal{N})$. The variables in $dom(\phi) = dom(\sigma) = \{x_1,x_2,\dots,x_n\}$ enable the attacker to refer to each term M_i. If $\tilde{n}$ contains only one element n, we also write $\nu n.\sigma$ instead of $\nu\tilde{n}.\sigma$.

If $\phi = \nu\tilde{n}.\{M_1/x_1, M_2/x_2, \dots, M_n/x_n\}$ is a frame and σ' is a substitution, then the application of the substitution σ' to ϕ is the frame $\phi\sigma' = \nu\tilde{n}\sigma'.\{M_1\sigma'/x_1,\dots,M_n\sigma'/x_n\}$.

Given a term $T \in \mathcal{T}(\Sigma,\mathcal{N},\mathcal{X})$, we denote by $fn(T)$ the set of names that appear in T. By $vars(T)$ we denote the set of variables that appear in T. We extend $vars$ to tuples, sets and multiple arguments: $vars(\{U_1,\dots,U_k\}) = vars((U_1,\dots,U_k)) = vars(U_1,\dots,U_k) = vars(U_1) \cup \dots vars(U_k)$.

Given a frame $\phi = \nu\tilde{n}.\{M_1/x_1, M_2/x_2, \dots, M_n/x_n\}$, we let $names(\phi) = \tilde{n} \cup names(M_1) \cup \dots \cup names(M_n)$, $bn(\phi) = \tilde{n}$ and $fn(\phi) = names(\phi) \setminus bn(\phi)$,

We extend the function fn to sets, tuples, and multiple arguments in a natural way: $fn((U_1,\dots,U_k)) = fn(\{U_1,\dots,U_k\}) = fn(U_1,\dots,U_k) = fn(U_1) \cup \dots \cup fn(U_k)$. The functions $names$ and bn are extended in a similar way.

Definition 1. We say that two frames $\phi = \nu\tilde{n}.\sigma$ and $\phi' = \nu\tilde{n'}.\sigma'$ are *equal up to alpha renaming*, and write $\phi =_\alpha \phi'$, iff the bound names $\tilde{n}$ and $\tilde{n'}$ can be α-renamed such that the two resulting frames are equal.

If $\phi = \nu\tilde{n}.\sigma$ is a frame, then we denote by $\nu\tilde{m}.\phi$ the frame $\nu(\tilde{n} \cup \tilde{m}).\phi$. If $\phi' = \nu\tilde{n'}.\sigma'$ is another frame with $dom(\phi) \cap dom(\phi') = \emptyset$, we denote by $\phi \mid \phi'$ the parallel composition of ϕ and ϕ', defined to be any frame of the form $\nu(\tilde{n_\alpha} \cup \tilde{n'_\alpha}).(\sigma_\alpha \cup \sigma'_\alpha)$, with $\tilde{n_\alpha} \cap \tilde{n'_\alpha} = \emptyset$, $\phi =_\alpha \nu\tilde{n_\alpha}.\sigma_\alpha$ and $\phi' =_\alpha \nu\tilde{n'_\alpha}.\sigma'_\alpha$.

2.5 Deduction

A weak notion of the secrecy of some data is non-deducibility.

Definition 2. We say that a term T is deducible from a frame $\phi = \nu\tilde{n}.\sigma$ under and equational theory E, and write $\phi \vdash_E T$, iff there exists a term $T' \in \mathcal{T}(\mathcal{N}\backslash\tilde{n}, dom(\sigma))$ such that $T'\sigma =_E T$ (in this case, T' is called a recipe of T in ϕ).

Example. In the frame $\nu\{n, r, k\}.\{senc^r(n, k)/x_1, k/x_2\}$, r is an example of a term that is not deducible. However n and $senc^k(k, k)$ are examples of terms which are deducible, the corresponding recipes being $sdec(x_1, x_2)$ and $senc^{x_2}(x_2, x_2)$.

If R is a convergent term rewriting system that implements E, we also write $\phi \vdash_R T$ instead of $\phi \vdash_E T$ and if $\phi = \nu\tilde{n}.\{M_1/x_1, \ldots, M_n/x_n\}$, we let $\phi{\downarrow_R} = \nu\tilde{n}.\{M_1{\downarrow_R}/x_1, \ldots, M_n{\downarrow_R}/x_n\}$. If R is obvious from the context, we also write $\phi{\downarrow}$ instead of $\phi{\downarrow_R}$.

2.6 Static equivalence

A stronger form of secrecy is to demand that the attacker is not able to distinguish a real frame from the same frame, but where the secret data is replaced by a new random name.

The impossibility of the intruder to distinguish between two frames is formalized by the notion of static equivalence.

Example Assume $\phi = \nu\{r, k\}.\{senc^r(n, k)/x_1, k/x_2\}$ and $\phi' = \nu\{r, k\}.\{senc^r(k, k)/x_1, k/x_2\}$ are two frames over Σ_{senc} and assume 0 is a constant.

Then the intruder can differentiate between the two frames by determining if $sdec(x_1, x_2)$ is equal modulo the equational theory to n. This test holds in ϕ but not in ϕ'. Notice that the test contains only function symbols ($sdec$), free names (n) and variables from $dom(\phi) = dom(\phi')$. In particular, the test does not make use of bounded names (r, k), which the intruder does not know, because they were freshly generated. We say that ϕ is not statically equivalent to ϕ'.

Let us consider $\phi_0 = \nu\{k, r\}.\{senc^r(n, k)/x_1\}$ and $\phi'_0 = \nu\{r, k\}.\{senc^r(k, k)/x_1\}$ to be variants of the frames above where the key k remains secret. Then there is no test that the intruder can perform to distinguish between ϕ_0 and ϕ'_0, therefore ϕ_0 is statically equivalent to ϕ'_0.

Now we are ready to formally the notion of static equivalence.

Definition 3. We say that two terms M and N from $\mathcal{T}(\mathcal{N}, \mathcal{X})$ are equal in the frame ϕ modulo an equational theory E, and we write $(M =_E N)\phi$, iff $\phi =_\alpha \nu\tilde{n}.\sigma$ for some set of names $\tilde{n}$ and some substitution σ such that $(fn(M) \cup fn(N)) \cap \tilde{n} = \emptyset$ and such that $M\sigma =_E N\sigma$.

Definition 4. We say that two frames $\phi_1 = \nu\tilde{n}_1.\sigma_1$ and $\phi_2 = \nu\tilde{n}_2.\sigma_2$ are statically equivalent modulo an equational theory E, and we write $\phi_1 \approx_E \phi_2$ iff $dom(\sigma_1) = dom(\sigma_2)$ and, for any two terms M and N, $(M =_E N)\phi_1 \iff (M =_E N)\phi_2$.

If R is a convergent term rewriting system that implements E, we sometimes use the notation $\phi_1 \approx_R \phi_2$ instead of $\phi_1 \approx_E \phi_2$.

Example. Let $\phi_1 = \nu\{r, n, k\}\{senc^r(0, k)/x, r/y\}$ and $\phi_2 = \nu\{r, n, k\}\{senc^r(1, k)/x, r/y\}$. These two frames are statically equivalent, intuitively because the key k is secret. However, as soon as k is *published*, the equivalence no longer holds: $\phi'_1 = \nu\{r, n, k\}\{senc^r(0, k)/x, r/y, k/z\}$ and $\phi'_2 = \nu\{r, n, k\}\{senc^r(1, k)/x, r/y, k/z\}$ are not statically equivalent, because the term $sdec(x, z)$ is equal modulo the equational theory to 0 in the first frame but not in the second.

3 Composition under a shared secret in the passive case

The security of the composition of security protocols is an important problem, because protocols do not exist in isolation: usually several protocols run on the same network.

We are interested in analyzing the security of protocols that share a common secret k. For example, k might be a key that is used by a user to communicate with two different network services.

A similar question was posed and answered in [3], where the authors investigate the security of the parallel composition of security protocols under a weak secret in a model which is close to the applied-π calculus.

They prove that in the passive case, security does compose under a weak shared secret. In the active case, they give a disarming counter-example, but provide a syntactic transformation that helps preserve the security of the composition.

In the following, we analyze the parallel composition of two security protocols under a general secret k.

We concentrate here on the passive case and we suppose in what follows that the security property we have in mind can be described as the static equivalence of two frames (we have already shown how strong secrecy can be described in terms of static equivalence).

We show that, under certain reasonable conditions, the composition of two independently secure protocols that only share the secret k and make use of probabilistic cryptographic primitives, is secure in the passive case in the formal model of the applied π-calculus.

3.1 Static equivalence is not in general preserved by composition under shared secrets

If two cryptographic protocols are secure independently, it is not in general true that their composition under a shared secret k is still secure. Consider the case of a toy e-voting protocol, where the vote 'yes' or 'no' (represented by the constants 1 and 0) is encrypted using a deterministic encryption method with the secret key k. The eavesdropper will observe one of the two frames:

$\phi_1 = \nu k.\{senc(0,k)/x_1\}$

$\phi_2 = \nu k.\{senc(1,k)/x_1\}$

Because the two frames are statically equivalent, it is not possible for the eavesdropper to distinguish between them. Therefore, the value of the vote is a strong secret.

Composition counter-example 1

Suppose that another protocol running in parallel is badly written and for some reason sends the value of k in clear-text. The attacker will then observe the frame:

$\phi = \nu k.\{k/x_2\}$

But now the attacker can distinguish between the two possible values of the vote by checking if $0 =_E sdec(x_1, x_2)$. So to have any chance of being able to obtain a composition result under a shared key k, we must demand that k is not deducible from ϕ.

Composition counter-example 2

But this is not always sufficient. Assume that the second protocol does not send k in clear, but instead, it always sends through the network a vote of 0, encrypted with the key k. An eavesdropper will therefore always observe the frame:

$\phi = \nu k.\{senc(0,k)/x_2\}$

Independently, the two protocols are secure. However, a passive attacker observing both protocols at the same time will be able to find out the value of the vote in the first protocol by comparing x_1 with x_2.

Technically, even if $\nu k.\phi'_1 \approx \nu k.\phi'_2$, it is not in general true that $\nu k.(\phi'_1 \mid \phi') \approx \nu k.(\phi'_2 \mid \phi')$.

Composition counter-example 3

Another interesting example of why static equivalence is not preserved in general by composition under a shared secret is the case where, for some reason, the secret k in sent over the network wrapped by a hash function h:

$\phi'_1 = \{h(k)/x_1\}$

$\phi'_2 = \nu n.\{h(n)/x_1\}$

The frames $\nu k.\phi'_1$ and $\nu k.\phi'_2$ are statically equivalent, because both essentially consist of a bound name under a hash function. However, if another protocol generates the frame $\nu k.\phi'$, where:

$\phi' = \{h(k)/x_2\}$

the attacker can distinguish between $\nu k.(\phi'_1 \mid \phi')$ and $\nu k.(\phi'_2 \mid \phi')$ by testing if $x_1 =_E x_2$.

In fact, if k is allowed to appear directly under a deterministic cryptographic operator, we can create small variations of the above counterexamples.

3.2 Probabilistic cryptographic primitives

We have presented some examples in which parallel composition of frames under a shared secret does not preserve static equivalence. Indeed, as soon as the equational theory contains a deterministic cryptographic operator we have some disarming counter-examples.

Therefore, in this section, we chose to study the composition problem for a class of equational theories that model probabilistic cryptographic primitives. To continue, we fix the signature Σ to be:

$$\Sigma = \{pair/2, fst/1, snd/1, senc/3, sdec/2\} \cup \{(c_i)_{1 \le i \le N}/0\}$$

In the above signature, *pair* denotes the pairing of two terms, while *fst* and *snd* extract the first and respectively the second element from a pair.

The function symbol *senc* denotes probabilistic symmetric encryption. The encryption of V with the key W and the randomness U is denoted by the term $senc(U,V,W)$. To emphasize the randomness, we also write $senc^U(V,W) = senc(U,V,W)$. Decryption is performed by *sdec*, $sdec(U,V)$ denoting the decryption of U with the key V.

c_i are any number $N \ge 0$ of constants (arity 0 function symbols).

We fix the equational theory E:

$$sdec(senc^r(x,y),y) = x$$
$$fst(pair(x,y)) = x$$
$$snd(pair(x,y)) = y$$

It is easy to see that the rewriting system R obtained by orienting the above equations from left to right is convergent and that it implements E.

In the rest of this section, we will assume we work with the signature Σ, with the equational theory E and with the rewriting system R.

3.3 Constructor frames

We call *senc*, *pair* and c_i constructors, while *sdec*, *fst* and *snd* are destructors. Typically, in the normal run of a protocol, agents will not apply destructor symbols over some term which does not begin with the corresponding constructor symbol.

In the case of our equational theory, we suppose the honest agents will not attempt to decrypt an encryption for which they do not know the key, and will not attempt to use *fst* or *snd* except on messages which begin by *pair*.

For example, it would not make sense for an honest agent to create the terms $sdec(0,1)$, $sdec(senc(n,k),k')$ or $fst(k)$, where n, k and k' are distinct names and 0 and 1 are constants. Under such an assumption, the normal forms of the terms exchanged during the protocol will only contain constructor symbols.

Definition 5. The frames which are in normal form and in which terms do not contain destructor symbols are called *constructor* frames.

3.4 Probabilistic frames

In the case of probabilistic cryptographic primitives, the terms that represent the randomness given as an input to the cryptographic primitives cannot be no-matter what. For example, if all terms that begin by *senc* would be of the form $senc^0(\_,\_)$, where 0 is a constant, this would effectively render the encryption algorithm deterministic.

We formally define the function *rand* which associates to a term the set of its subterms which appear as randomness:

$rand(senc^R(U,V)) = \{R\} \cup rand(R) \cup rand(U) \cup rand(V)$

$rand(function(T_1,\dots,T_l)) = rand(T_1) \cup \dots \cup rand(T_l)$, where *function* is a function symbol from $\Sigma \setminus \{senc\}$ and l is its arity.

We naturally extend the above function to frames, such that:

$rand(\nu\tilde{n}.\{T_1/x_1,\dots,T_l/x_l\}) = rand(T_1) \cup \dots \cup rand(T_l)$.

Definition 6. A frame ϕ is called *probabilistic* if all terms $T \in rand(\phi)$ are names which are not deducible from ϕ.

3.5 Well-formed frames

We are now ready to state the hypothesis on the frames for which we prove the composition result:

Definition 7. A frame ϕ is well-formed with respect to k iff the following three conditions hold:

- $\nu k.\phi \nvdash k$
- ϕ is probabilistic[1]

[1] it is worth noting that the frame $\nu k.\phi$ will not contain k in random position

- ϕ is a constructor frame

3.6 Composition of deducibility

Before we can show a composition result for static equivalence, we must give a composition result for deducibility.

We prove that, for frames which are well-formed with respect to k, we cannot deduce more messages from $\nu k.(\phi_1 \mid \phi_2)$ than we can from $\nu k.\phi_1$ and $\nu k.\phi_2$, in a sense formalized by the following theorem:

Theorem 1. *Let $\phi_i =_\alpha \nu\tilde{n_i}.\sigma_i$, with $i \in \{1,2\}$ be two well-formed frames with respect to k and such that $\tilde{n_1}$, $\tilde{n_2}$ and $\{k\}$ are pairwise disjoint sets and $dom(\sigma_1) \cap dom(\sigma_2) = \emptyset$. If $\nu k.(\phi_1 \mid \phi_2) \vdash T$ and $fn(T) \cap \tilde{n_i} = \emptyset$, then $\nu k.\phi_{\bar{i}} \vdash T$ (where $\bar{i}$ is short for $3-i$; if $\phi_i = \phi_1$ then $\phi_{\bar{i}} = \phi_2$ and vice-versa).*

Proof. Let $\sigma = \sigma_1 \cup \sigma_2$.

It is sufficient to prove that for any recipe $M \in T(N \setminus (\tilde{n_1} \cup \tilde{n_2} \cup \{k\}), dom(\sigma))$, if $M\sigma\downarrow$ does not contain names from $\tilde{n_i}$, $M\sigma$ is deducible from $\phi_{\bar{i}}$.

We prove by induction on the size of M that $M\sigma\downarrow \cap \tilde{n_i} = \emptyset$ implies that $M\sigma$ is deducible from $\phi_{\bar{i}}$. Indeed, M must have one of the following three forms[2]:

- $M = d_1(d_2(\dots d_n(nam[,T_n])\dots[,T_2])[,T_1])$, for some $n \geq 0$, destructors d_i, name $nam \in N \setminus (\tilde{n_1} \cup \tilde{n_2} \cup \{k\})$ and $T_j \in T(N \setminus (\tilde{n_1} \cup \tilde{n_2} \cup \{k\}), dom(\sigma))$. The macro $[,T_i]$ should be read as "$,T_i$" if $d_i = sdec$ (T_i is then the decryption key) and as "" if $d_i \in \{fst, snd\}$ (because fst and snd only take one argument, a corresponding T_i is not necessary).

  Note that if $n = 0$, we obtain a base case of the induction where the size of M is 1 and M is a free name.

  Because all rewrite rules are of the form $c(d(\vec{U}),\vec{V}) \to U_k$, for some constructor c and destructor d, $M\sigma\downarrow = d_1(d_2(\dots d_n(nam[,T_n\sigma\downarrow])\dots[,T_2\sigma\downarrow])[,T_1\sigma\downarrow])$. As $M\sigma\downarrow$ does not contain names from $\tilde{n_i}$, its subterms $T_i\sigma\downarrow$ do not contain names from $\tilde{n_i}$ either. Also, the terms $T_i\sigma\downarrow$ are deducible from $\nu k.(\phi_1 \mid \phi_2)$ using the recipes T_i and therefore, by the induction hypothesis, they are deducible from $\nu k.\phi_{\bar{i}}$ by some recipes $T'_i \in T(N \setminus (\tilde{n_{\bar{i}}} \cup \{k\}), dom(\sigma_{\bar{i}}))$. Then, $M\sigma\downarrow$ is deducible from $\nu k.\phi_{\bar{i}}$ by the recipe $M' = d_1(d_2(\dots d_n(nam[,T'_n])\dots[,T'_2])[,T'_1]) \in T(N \setminus (\tilde{n_{\bar{i}}} \cup \{k\}), dom(\sigma_{\bar{i}}))$.

- $M = d_1(d_2(\dots d_n(c(U_1,\dots,U_{ar(c)})[,T_n])\dots[,T_2])[,T_1])$, for some $n \geq 0$, destructors d_i, constructor c of arity $ar(c)$ and $T_j \in T(N \setminus (\tilde{n_1} \cup \tilde{n_2} \cup \{k\}), dom(\sigma))$

  Note that if $n = 0$ and $ar(c) = 0$ this case corresponds to the base case of the induction where the size of M is 1 and M is a constant.

  We distinguish two cases.

  - If $n \geq 1$ and $(d_n(c(U_1,\dots,U_{ar(c)})[,T_n]))\sigma \to_R U_l\sigma$ by a head reduction for some $l \in \{1,\dots,ar(c)\}$ then $M\sigma\downarrow =_E d_1(d_2(\dots U_l\sigma\dots[,T_2\sigma])[,T_1\sigma])$ and therefore $M' = d_1(d_2(\dots U_l\dots[,T_2])[,T_1])$ is a recipe for $M\sigma\downarrow$ in $\nu k.(\phi_1 \mid \phi_2)$. But M' is smaller than M and we can apply the induction hypothesis to conclude.

[2]the base cases of the induction, where the size of M is 1, are particular cases of the three more general cases

- If $n = 0$ or $(d_n(c(U_1, \ldots, U_{ar(c)})[, T_n]))\sigma \not\to_R U_l\sigma$ for some $l \in \{1, \ldots, n\}$ by any head reduction, then it holds that $M\sigma \downarrow = d_1(d_2(\ldots d_n(c(U_1\sigma \downarrow, \ldots, U_{ar(c)}\sigma \downarrow)[, T_n\sigma \downarrow]) \ldots [, T_2\sigma \downarrow])[, T_1\sigma \downarrow])$. The terms T_j and U_k are recipes in $\nu k.(\phi_1 \mid \phi_2)$ which are smaller than M and we can apply the induction hypothesis to obtain recipes T'_j and U'_k in $\nu k.\phi_{\bar{i}}$ for the terms $T_j\sigma \downarrow$ and $U_k\sigma \downarrow$. Then $M\sigma \downarrow$ can be deduced from $\nu k.\phi_{\bar{i}}$ by the recipe $d_1(d_2(\ldots d_n(c(U'_1, \ldots, U'_{ar(c)})[, T'_n]) \ldots [, T'_2])[, T'_1])$.

- $M = d_1(d_2(\ldots d_n(x[, T_n]) \ldots [, T_2])[, T_1])$, for some $n \geq 0$, destructors d_i, variable $x \in dom(\sigma)$ and $T_j \in T(N \setminus (\tilde{n_1} \cup \tilde{n_2} \cup \{k\}), dom(\sigma))$

  Note that if $n = 0$ this case corresponds to the base case of the induction where the size of M is 1 and M is a variable.

  - if $M\sigma \downarrow = d_1(U[, T_1\sigma \downarrow])$ for some term U in normal form then U (and $T_1\sigma \downarrow$) cannot contain names from n_i. But then, by applying the induction hypothesis, we can obtain recipes U' (and T'_1) in $\nu k.\phi_{\bar{i}}$ for U (and $T_1\sigma \downarrow$) and a recipe $M' = d_1(U'[, T'_1])$ for $M\sigma$.
  - if $M\sigma \downarrow$ does not begin by d_1, it means that all of the destructors d_i succeeded and that for each $1 \leq j \leq n$, $d_j(\ldots d_n(x[, T_n]) \ldots [, T_j])$ is a subterm of x. Also, for all j such that $d_j = sdec$, $T_j\sigma \downarrow$ must have been the correct encryption key for a subterm of x and thus $T_j\sigma \downarrow$ is a subterm of $x\sigma$.
    1. If $x \in dom(\sigma_{\bar{i}})$ then the $T_j\sigma \downarrow$ for which $d_j = sdec$ are subterms of the frame $\phi_{\bar{i}}$ and they cannot contain names from n_i. We can thus apply the induction hypothesis on these terms and obtain recipes T'_j in $\phi_{\bar{i}}$ for them. But then, we immediately obtain a recipe $M' = d_1(d_2(\ldots d_n(x[, T'_n]) \ldots [, T'_2])[, T'_1])$ in $\phi_{\bar{i}}$ for $M\sigma$.
    2. If $x \in dom(\sigma_i)$ then $M\sigma \downarrow$ is deducible from $\nu k.\phi_i$ (by a reasoning identical modulo renaming to the previous point). By hypothesis, $M\sigma \downarrow$ does not contain names from n_i and therefore cannot contain encryptions [3]. That means $M\sigma \downarrow$ only contains *pair*, free names and k. Then again, k cannot appear in clear in $M\sigma \downarrow$ because it would break the secrecy assumption of k in $\nu k.\phi_i$. This means $M\sigma \downarrow$ only contains free names and *pair* and is therefore trivially deducible.

We have shown that for any recipe $M \in T(N \setminus (\tilde{n_1} \cup \tilde{n_2} \cup \{k\}), dom(\sigma))$, if $M\sigma \downarrow$ does not contain names from n_i, then $M\sigma \downarrow$ is deducible from $\phi_{\bar{i}}$.

From this, we can easily conclude: if T is deducible from $\nu k.(\phi_1 \mid \phi_2)$, then there exists a recipe $M \in T(N \setminus (\tilde{n_1} \cup \tilde{n_2} \cup \{k\}), dom(\sigma))$ such that $T =_E M\sigma$. If T does not contain names from n_i, neither does $T \downarrow = M\sigma \downarrow$. We have shown by induction that all such $M\sigma \downarrow$ are deducible from $\nu k.\phi_{\bar{i}}$ and therefore T is deducible as well. □

3.7 Composition of static equivalence

In this section we show that static equivalence composes under a shared secret k, in the case of frames which are well-formed with respect to k. This is enough to conclude that security properties expressible as static equivalences between frames, compose in the passive case under a shared secret k, assuming the protocols respect the given conditions.

[3]because ϕ_i is a probabilistic frame

Theorem 2. *Let $\phi_1 =_\alpha \nu\tilde{n}_1.\sigma_1$, $\phi_2 =_\alpha \nu\tilde{n}_2.\sigma_2$ and $\phi =_\alpha \nu\tilde{n}.\sigma$ be well-formed frames with respect to k, such that $\tilde{n}$, $\tilde{n}_1$, $\tilde{n}_2$ and $\{k\}$ are pairwise disjoint sets, $(fn(\phi_1) \cup fn(\phi_2)) \cap (\tilde{n}_1 \cup \tilde{n}_2) = \emptyset$, $dom(\sigma_1) = dom(\sigma_2)$ and $dom(\sigma_1) \cap dom(\sigma) = \emptyset$.*

If $\nu k.\phi_1 \approx \nu k.\phi_2$, then $\nu k.(\phi_1 \mid \phi) \approx \nu k.(\phi_2 \mid \phi)$.

Proof. When no name is shared by the frames under parallel composition, it is easy to prove that $\phi_1 \approx \phi_2$ implies $\phi_1 \mid \phi \approx \phi_2 \mid \phi$: supposing by contradiction that there is a test $M \overset{?}{=}_E N$ that distiguishes between $\phi_1 \mid \phi$ and $\phi_2 \mid \phi$ we can easily construct a test $M' \overset{?}{=}_E N'$ that distinguishes between ϕ_1 and ϕ_2 by setting $M' = M\sigma$ and $N' = N\sigma$ if $\phi = \nu\tilde{n}.\sigma$.

The same trick does not work for proving that $\nu k.\phi_1 \approx \nu k.\phi_2$ implies $\nu k.(\phi_1 \mid \phi) \approx \nu k.(\phi_2 \mid \phi)$ because $M' = M\sigma$ and $N' = N\sigma$ might contain k, which is not allowed in a test that distinguishes between $\nu k.\phi_1$ and $\nu k.\phi_2$.

To show that $\nu k.\phi_1 \approx \nu k.\phi_2$ implies $\nu k.(\phi_1 \mid \phi) \approx \nu k.(\phi_2 \mid \phi)$ when ϕ, ϕ_1 and ϕ_2 are well-formed frames with respect to k, we transform $M\sigma$ and $N\sigma$ in a way such that they do not contain k and such that (in)equality modulo the equational theory E is preserved by this transformation for the terms we care about.

The transformation is defined by the function f_ϕ, parameterized by the frame $\phi = \nu\tilde{n}.\sigma$. $f_\phi(T)$ is a copy of term T, where each subterm T' of the form $senc^r(\_, K)$, with $r \in \tilde{n}$ and such that $\phi \nvdash r$ and $\nu k.\phi \nvdash K$ [4], is replaced by a new name $r_{T'}$.

To formally prove the theorem, we need a few lemmas about f_ϕ. In all of the following lemmas, we suppose $\phi = \nu\tilde{n}.\sigma$ and $\phi' = \nu\tilde{n'}.\sigma'$ are well-formed frames with respect to k such that $dom(\sigma)$, $dom(\sigma')$, $\tilde{n}$, $\tilde{n'}$ and $\{k\}$ are pairwise disjoint sets. We also assume that $(fn(\phi) \cup fn(\phi')) \cap (\tilde{n} \cup \tilde{n'}) = \emptyset$.

Lemma 1. *If $M \in T(N \setminus (\tilde{n} \cup \tilde{n'} \cup \{k\}), dom(\sigma_1) \cup dom(\sigma_2))$ is a recipe in $\nu k.(\phi \mid \phi')$, then $f_\phi((M\sigma)\sigma') = f_\phi(M\sigma)\sigma'$.*

Proof. By structural induction on M:

- *Base cases.* If M is a constant or a name, the conclusion is trivial.

  If $M = x_i$ for some $x_i \in dom(\sigma)$ then $M\sigma = x_i\sigma = \sigma(x_i)$. Therefore $M\sigma$ does not contain variables from $dom(\sigma')$ and therefore $(M\sigma)\sigma' = M\sigma$. Applying f_ϕ to both sides of the equation yields $f_\phi((M\sigma)\sigma') = f_\phi(M\sigma)$. Note that $f_\phi(M\sigma)$ doesn't contain variables from $dom(\sigma')$ either and therefore $f_\phi(M\sigma) = f_\phi(M\sigma)\sigma'$. The conclusion results from the last two equalities.

  If $M = x_i$ for some $x_i \in dom(\sigma')$ then $M\sigma = M$. As $M\sigma' = x_i\sigma' = \sigma'(x_i)$ cannot contain names from $\tilde{n}$ [5], $f_\phi(M\sigma') = M\sigma'$. Also, $f_\phi(M)$ is equal to M and by applying σ' to both, we obtain $f_\phi(M)\sigma' = M\sigma'$. From the last two equalities we obtain $f_\phi(M\sigma') = f_\phi(M)\sigma'$. Earlier we have seen that $M\sigma$ is equal to M and we obtain our conclusion by replacing M with $M\sigma$ in the last equation.

- *Inductive case.* If $M = senc^T(V, K)$ and if $(T\sigma)\sigma'$ is a name from $\tilde{n}$, then $(T\sigma)\sigma'$ is deducible from $\nu k.(\phi \mid \phi')$ by the recipe T and, by theorem 1, it is also deducible from $\nu k.\phi$. Therefore, the replacement of *senc* by a fresh name is not performed and we

[4] by theorem 1, K cannot be deduced from $\nu k.(\phi_1 \mid \phi)$ either

[5] $\tilde{n'}$ is disjoint from $\tilde{n}$ by hypothesis

obtain that $f_\phi(M) = senc^{f_\phi(T)}(f_\phi(V), f_\phi(K))$. The conclusion follows trivially from the induction hypothesis.

Similarly, if $M = func(U_1, U_2, \ldots, U_{ar(func)})$, where *func* is a function symbol other than *senc* and $ar(func)$ is its arity, then $f_\phi(M) = func(f_\phi(U_1), f_\phi(U_2), \ldots, f_\phi(U_{ar(func)}))$ and the result is immediate from the induction hypothesis.

□

Lemma 2. *If M is a recipe in the frame $\nu k.(\phi \mid \phi')$ and $\phi = \nu\tilde{n}.\sigma$ then $f_\phi(M\sigma)$ does not contain k as a subterm.*

Proof. We prove by structural induction on M that all the subterms of $f_\phi(M\sigma)$ are deducible from $\nu k.\phi$:

- if M is a constant or a free name, the conclusion is trivial.
- if M is a variable $x \in dom(\sigma)$, $f_\phi(M\sigma) = f_\phi(\sigma(x))$. It is easy to see that all of the term of $f_\phi(\sigma(x))$ are deducible, since f_ϕ replaces the encryptions for which the key is not deducible by a free name.
- if M is a variable $x \in dom(\sigma')$, $f_\phi(M\sigma) = M\sigma = M = x$.
- if M is $func(T_1, \ldots, T_{ar(func)})$, we can apply the induction hypothesis on T_i to obtain that $T_i\sigma$ are deducible, and, as an immediate consequence, $M\sigma$ is deducible.

We have shown that all subterms of $f_\phi(M\sigma)$ are deducible from $\nu k.\phi$. Because ϕ is well-formed with respect to k, $\nu k.\phi \nvdash k$. We conclude that k cannot appear as a subterm of $f_\phi(M\sigma)$. □

Lemma 3. *If M is a recipe in $\nu k.(\phi \mid \phi')$ then $f_\phi((M\sigma)\sigma' \downarrow) = f_\phi((M\sigma)\sigma') \downarrow$*

Proof. By induction on the recipe M:

- if M is a constant or a name, the conclusion is immediate
- if $M = x \in dom(\sigma)$ then $(M\sigma)\sigma' = M\sigma = \sigma(x)$ is already in normal form [6]. Therefore, we only need to show $f_\phi(\sigma(x)) = f_\phi(\sigma(x)) \downarrow$. But this is obviously the case, since we cannot introduce new rewriting possibilities by replacing equal terms with equal names.
- if $M = x \in dom(\sigma')$ then $(M\sigma)\sigma' = M\sigma' = \sigma'(x)$ is already in normal form and this case is similar to the previous.
- if $M = senc^U(V, W)$, then $(M\sigma)\sigma' \downarrow = senc^{(U\sigma)\sigma'\downarrow}((V\sigma)\sigma' \downarrow, (W\sigma)\sigma' \downarrow)$. The term $(U\sigma)\sigma'$ cannot be both a name from $\tilde{n}$ and non-deducible from $\nu k.\phi$[7], and we obtain $f_\phi((M\sigma)\sigma' \downarrow) = senc^{f_\phi((U\sigma)\sigma'\downarrow)}(f_\phi((V\sigma)\sigma' \downarrow), f_\phi((W\sigma)\sigma' \downarrow))$. On the other hand, $f_\phi((M\sigma)\sigma') \downarrow = senc^{f_\phi((U\sigma)\sigma')\downarrow}(f_\phi((V\sigma)\sigma') \downarrow, f_\phi((W\sigma)\sigma') \downarrow$. We apply the induction hypothesis on U, V and W to obtain immediately the conclusion.
- if $M = pair(U, V)$, the reasoning is similar to the previous case

[6]because frames well-formed with respect to k are in normal form

[7]Suppose $(U\sigma)\sigma'$ is a name n_0 from $\tilde{n}$. Then the term n_0 is deducible from $\nu k.(\phi \mid \phi')$ by the recipe U and, by theorem 1, is deducible from $\nu k.\phi$ as well

- if $M = sdec(U, V)$, we distinguish two cases:
  - if $(U\sigma)\sigma' \downarrow = senc^W(T, (V\sigma)\sigma' \downarrow)$ then $(M\sigma)\sigma' \downarrow = T$ and
$$f_\phi((M\sigma)\sigma' \downarrow) = f_\phi(T) \tag{1}$$
    If T is a name in $\tilde{n}$ which is not deducible from ϕ, then $senc^W(T, (V\sigma)\sigma' \downarrow)$ must be a subterm of a term in ϕ. Then, $(V\sigma)\sigma' \downarrow$ is deducible from $\nu k.(\phi \mid \phi')$ by the recipe V and, by theorem 1, from $\nu k.\phi$ as well. Thus, in any case, f_ϕ does not replace this encryption by a new name and therefore $f_\phi((U\sigma)\sigma' \downarrow) = senc^{f_\phi(W)}(f_\phi(T), f_\phi((V\sigma)\sigma' \downarrow))$. By applying the induction hypothesis on U we obtain $f_\phi((U\sigma)\sigma') \downarrow = senc^{f_\phi(W)}(f_\phi(T), f_\phi((V\sigma)\sigma' \downarrow))$
    By definition, $f_\phi((M\sigma)\sigma') = sdec(f_\phi((U\sigma)\sigma'), f_\phi((V\sigma)\sigma'))$ and therefore $f_\phi((M\sigma)\sigma') \downarrow = f_\phi(T)$. Unifying with (1), we obtain our conclusion.
  - if $(U\sigma)\sigma' \downarrow$ does not start with *senc* then can be no head reduction and the result follows immediately from the induction hypothesis
- if $M = fst(U)$ or $M = snd(U)$, the reasoning is similar to the previous case

□

Lemma 4. *f_ϕ is bijective.*

Proof. The inverse function f_ϕ^{-1} simply replaces every name $r_{T'}$ in its argument by T'. It is easy to see that $f_\phi^{-1}(f_\phi(T)) = T$. □

Lemma 5. *If M and N are recipes in $\nu k.(\phi \mid \phi')$ then $(M\sigma)\sigma' =_E (N\sigma)\sigma'$ iff $f_\phi((M\sigma)\sigma') =_E f_\phi((N\sigma)\sigma')$.*

Proof.
$$\begin{array}{cccl} (M\sigma)\sigma' & =_E & (N\sigma)\sigma' & \text{iff} \\ (M\sigma)\sigma' \downarrow & = & (N\sigma)\sigma' \downarrow & \text{iff (by bijectivity of } f_\phi) \\ f_\phi((M\sigma)\sigma' \downarrow) & = & f_\phi((N\sigma)\sigma' \downarrow) & \text{iff (by lemma 3)} \\ f_\phi((M\sigma)\sigma') & =_E & f_\phi((N\sigma)\sigma') & \end{array}$$

□

We now have sufficient lemmas to prove our main result. Suppose by contradiction that $\nu k.(\phi_1 \mid \phi) \not\approx \nu k.(\phi_2 \mid \phi)$. Let $M \overset{?}{=}_E N$ be a test that holds in $\nu k.(\phi_i \mid \phi)$ and not in $\nu k.(\phi_{\bar{i}} \mid \phi)$, for some $i \in \{1, 2\}$ and where $\bar{i} = 3 - i$. Then:

$$(M\sigma)\sigma_i =_E (N\sigma)\sigma_i$$

and

$$(M\sigma)\sigma_{\bar{i}} \neq_E (N\sigma)\sigma_{\bar{i}}$$

From lemma 2, it follows that $f_\phi(M\sigma) \overset{?}{=}_E f_\phi(N\sigma)$ is a valid test in ϕ_1 and ϕ_2.

Let us now prove that $f_\phi(M\sigma)\sigma_i =_E f_\phi(N\sigma)\sigma_i$. By our contradiction hypothesis, we know that $(M\sigma)\sigma_i =_E (N\sigma)\sigma_i$. By applying lemma 5 we obtain that $f_\phi((M\sigma)\sigma_i) =_E f_\phi((N\sigma)\sigma_i)$. By lemma 1 we immediately get $f_\phi(M\sigma)\sigma_i =_E f_\phi(N\sigma)\sigma_i$.

Now we only need to prove that $f_\phi(M\sigma)\sigma_{\bar{i}} \neq_E f_\phi(N\sigma)\sigma_{\bar{i}}$. By applying lemma 5 to the contradiction hypothesis $(M\sigma)\sigma_{\bar{i}} \neq_E (N\sigma)\sigma_{\bar{i}}$ we obtain that $f_\phi((M\sigma)\sigma_{\bar{i}}) \neq_E f_\phi((N\sigma)\sigma_{\bar{i}})$. By lemma 1 we immediately get $f_\phi(M\sigma)\sigma_{\bar{i}} \neq_E f_\phi(N\sigma)\sigma_{\bar{i}}$.

We supposed by contradiction that $\nu k.(\phi_1 \mid \phi) \not\approx \nu k.(\phi_2 \mid \phi)$ and therefore there exists a test $M \overset{?}{=}_E N$ that distinguishes between $\nu k.(\phi_1 \mid \phi)$ and $\nu k.(\phi_2 \mid \phi)$. We have shown that there exists a test $f_\phi(M\sigma) \overset{?}{=}_E f_\phi(N\sigma)$ which distinguishes between $\nu k.\phi_1$ and $\nu k.\phi_2$.

But this contradicts the hypothesis and therefore $\nu k.(\phi_1 \mid \phi) \approx \nu k.(\phi_2 \mid \phi)$. $\square$

4 Decidability results

In this section we study the following two problems:

- Intruder deduction problem

  Input: A frame ϕ and a ground term T.

  Output: $\phi \overset{?}{\vdash}_E T$

- Static equivalence problem

  Input: Two frames ϕ and ϕ'.

  Output: $\phi \overset{?}{\approx}_E \phi'$

It is well-known that, in general, both of the problems are undecidable [4]. It has been shown that even if E can be implemented by a convergent term rewriting system, the intruder deduction problem may be undecidable [4]. Also, as soon as the signature/equational theory contains an encryption operator, the intruder deduction problem is easier than static equivalence [4].

There exist many results in the literature that provide decision procedures for the first problem, taking into consideration various classes of equational theories [9, 4, 10, 11, 5, 6] and many others. However, for static equivalence, there are less results: in [4], the authors propose decision procedures for both problems when there exists a sub-term convergent rewriting system that implements E. Also, in an extension [6] of the first paper, the authors give some general conditions on the equational theory that ensures decidability of both problems. In [5], the authors provide decision procedures for so called monadic equational theories.

None of the above results applies to an equational theory that models a trapdoor bit commitment cryptographic primitive, which could be used is e-voting protocols to provide coercion resistance. Motivated by the example of this equational theory, which we present in detail in subsection 5.2, we give a semi-decision procedure which can be implemented for any convergent term rewriting system R.

In the remainder of this section, we assume R is a convergent term rewriting system that implements the equational theory of interest E.

4.1 Preliminaries

A deduction fact is a tuple $(T, t, \{(X_1, t_1), \dots, (X_k, t_k)\})$, written for convenience as $[T \rhd t \mid X_1 \rhd t_1, \dots, X_k \rhd t_k]$, where $X_i \in \mathcal{X}$ and T, t and t_i are terms from $\mathcal{T}(\mathcal{N}, \mathcal{X})$ such that $vars(t) \subseteq vars(t_1, \dots, t_k)$, $vars(T) \subseteq \{X_1, \dots, X_k\}$ and $\{X_1, \dots, X_k\} \cap vars(t_1, \dots, t_k) = \emptyset$. If $k = 0$, we also write $[T \rhd t]$ instead of $[T \rhd t \mid \emptyset]$. A deduction fact is *solved* if all t_i are variables.

Intuitively, assuming we are in possession of a deduction fact $[T \rhd t \mid X_1 \rhd t_1, \dots, X_k \rhd t_k]$, it means that if terms of the form t_i are deducible by recipes R_i, then a term of the form t is deducible by the recipe $T\{R_i/X_i\}$.

A deduction fact $F = [T \rhd t \mid X_1 \rhd t_1, \dots, X_k \rhd t_k]$ is equivalent to a deduction fact $F' = [T' \rhd t' \mid X'_1 \rhd t'_1, \dots, X'_l \rhd t'_l]$, and written $F \equiv F'$, iff there is a bijection σ between $vars(F)$ and $vars(F')$ such that $F\sigma = F'$ (informally, two facts are equivalent iff they are the same modulo a renaming of their variables). For a set of deduction facts S, we say that a fact $F \in_{\equiv} S$ iff S contains a deduction fact equivalent to F.

For any deduction fact F, we let $[F]$ denote the equivalence class of F, that is, the set of deduction facts equivalent to F; formally, $[F] = \{F' \mid F \equiv F'\}$. If S is a set of deduction facts, we denote by $S_{|\equiv}$ the set $\{[F] \mid F \in S]\}$.

Let T and T' be any terms in $\mathcal{T}(\mathcal{N}, \mathcal{X})$. We denote by $mgu(T, T')$ the set of most general idempotent unifiers of T and T'.

4.2 Saturation algorithm

To any frame $\phi = \nu\tilde{n}.\{M_1/x_1, \ldots, M_n/x_n\}$ and set of names $\mathcal{N}_0$ (we assume in what follows that $bn(\phi) \cap \mathcal{N}_0 = \emptyset$), we associate an initial set of deduction facts $initial(\phi, \mathcal{N}_0)$ as follows:

- for each $i \in \{1, \ldots, n\}$, $[x_i \rhd M_i\!\downarrow] \in initial(\phi, \mathcal{N}_0)$
- for each function symbol f of arity l in the signature Σ, $[f(X_1, \ldots, X_l) \rhd f(y_1, \ldots y_l) \mid X_1 \rhd y_1, \ldots, X_l \rhd y_l] \in initial(\phi, \mathcal{N}_0)$.
- for each name $n \in \mathcal{N}_0$, $[n \rhd n] \in initial(\phi, \mathcal{N}_0)$
- $initial(\phi, \mathcal{N}_0)$ contains only the above terms

Example Let $\phi = \nu\{a, b, c, d\}\{f(a, b, c, d)/x_1, c/x_2, d/x_3\}$. Then $initial(\phi, \{e\})$ is:

$$
\begin{array}{ll}
\{ & [x_1 \rhd f(a, b, c, d)], \\
 & [x_2 \rhd c], \\
 & [x_3 \rhd d], \\
 & [f(X_1, X_2, X_3, X_4) \rhd f(x_1, x_2, x_3, x_4) \mid X_1 \rhd x_1, X_2 \rhd x_2, X_3 \rhd x_3, X_4 \rhd x_4], \\
 & [tdcommit(X_1, X_2, X_3) \rhd tdcommit(x_1, x_2, x_3) \mid X_1 \rhd x_1, X_2 \rhd x_2, X_3 \rhd x_3], \\
 & [open(X_1, X_2) \rhd open(x_1, x_2) \mid X_1 \rhd x_1, X_2 \rhd x_2] \\
 & [e \rhd e]\}
\end{array}
$$

Intuitively, $initial(\phi, \mathcal{N}_0)$ is a representation of the terms that are context over terms in ϕ.

Let $sat(\phi, \mathcal{N}_0)$ be the saturation of the set $initial(\phi, \mathcal{N}_0)$ by the closure rules described in Figure 1. The saturated set of a frame ϕ represents the closure of the $initial(\phi, \mathcal{N}_0)$ set by the rewrite rules of the convergent term rewriting system R.

In the following definition, we formally describe what terms are represented by a set of deduction facts S.

Definition 8. For any term $T \in \mathcal{T}(\mathcal{N})$, we say that T is *generated by a set of deduction facts* S, and write $S \models T$, iff there exists a solved fact $[U \rhd V \mid X_1 \rhd x_1, \ldots, X_k \rhd x_k] \in_{\equiv} S$ with V and T unifiable, $V \notin \mathcal{X}$ and, if $\sigma \in mgu(T, V)$, $S \models \sigma(x)$ for each $x \in dom(\sigma)$.

It is easy to see that the above recursive definition is well-founded, because $\sigma(x)$ is a strict sub-term of T.

The following conjectures provide crucial properties that the set $sat(\phi, \mathcal{N}_0)$ has.

$$\frac{\begin{array}{l}[T \rhd t \mid X_1 \rhd t_1, \ldots, X_k \rhd t_k] \in_{\equiv} S \text{ is a solved fact} \\ l \to r \in R \\ vars(l) \cap vars(t_1, \ldots, t_k) = \emptyset \\ \sigma \in mgu(l, t)\end{array}}{[T \rhd r\sigma \downarrow \mid X_1 \rhd t_1\sigma, \ldots X_k \rhd t_k\sigma] \in_{\equiv} S}\ \textit{Context reduction}$$

$$\frac{\begin{array}{l}F = [T \rhd t \mid X_1 \rhd t_1, \ldots, X_k \rhd t_k] \in_{\equiv} S \text{ is a solved fact} \\ F' = [T' \rhd t' \mid X'_1 \rhd t'_1, \ldots, X'_l \rhd t'_l] \in_{\equiv} S \text{ is an unsolved fact} \\ vars(F) \cap vars(F') = \emptyset \\ t'_1 \notin \mathcal{X} \\ \sigma \in mgu(t, t'_1)\end{array}}{[T'\{T/X'_1\} \rhd t'\sigma \mid X'_2 \rhd t'_2\sigma, \ldots, X'_l \rhd t'_l\sigma, X_1 \rhd t_1\sigma, \ldots, X_k \rhd t_k\sigma] \in_{\equiv} S}\ \textit{Solver}$$

Figure 1: Saturation rules for a set of deduction facts S and convergent term rewriting system R

Conjecture 1. *Let ϕ be a frame, $\mathcal{N}_0$ a set of names with $\mathcal{N}_0 \cap bn(\phi) = \emptyset$ and $F = [T \rhd t \mid X_1 \rhd t_1, \ldots, X_k \rhd t_k] \in sat(\phi, \mathcal{N}_0)$ be a deduction fact.*

If there exists a substitution τ from $vars(t_1, \ldots, t_k)$ to $\mathcal{T}(\mathcal{N})$ such that $t_i\tau$ are deducible from ϕ with recipes R_i, then $t\tau$ is deducible from ϕ with the recipe $T\{R_i/X_i\}$.

Conjecture 2. *Assume $\phi = \nu\tilde{n}.\sigma$ and $\mathcal{N}_0 \cap \tilde{n} = \emptyset$. For any recipe $W \in \mathcal{T}(\mathcal{N}_0, dom(\phi))$, the term $W\sigma\downarrow$ is generated by $sat(\phi, \mathcal{N}_0)$.*

Conjecture 3. *For any ground term T in normal form, $sat(\phi, fn(T) \cup fn(\phi)) \models T$ iff $\phi \vdash T$.*

Lemma 6. *A ground term T in normal form is deducible from $\phi = \nu\tilde{n}.\sigma$ iff there exists a recipe W from $\mathcal{T}(fn(T) \cup fn(\phi), dom(\phi))$ such that $W\sigma =_E T$.*

Proof. If there exists such a recipe, it is obvious that T is deducible. In the other sense, if T is deducible and in normal form, there exists a recipe $W' \in \mathcal{T}(\mathcal{N}, dom(\sigma))$ such that $W'\sigma \to_R^* T$. But our rewriting system is stable by replacement of arbitrary terms for names, so from W' we can obtain the recipe W we need by replacing all names in $fn(W) \setminus (fn(T) \cup fn(\phi))$ by any terms (for example, by a variable in $dom(\phi)$). □

5 Deduction algorithm

The above conjectures state the correctness of algorithm 1.

```
Input: a frame $\phi = \nu\tilde{n}.\sigma$ and a ground term $T$
Output: $\phi \overset{?}{\vdash}_R T$
begin
    $\phi \leftarrow \phi \downarrow_R$;
    $T \leftarrow T \downarrow_R$;
    return $sat(\phi, fn(T) \cup fn(\phi)) \models T$;
end
```

Algorithm 1: The deduction algorithm

The only step in the above procedure which may not terminate is the computation of the set $sat(\phi, fn(T) \cup fn(\phi))$. Indeed, this set may be infinite for certain term rewriting systems R.

5.1 Static equivalence

We note by $Eq_{all}(\phi)$ the set of all tests that are true in ϕ. By $Eq(\phi, \mathcal{N}_0)$, we note the set of all tests that are true in ϕ and that only use names from $\mathcal{N}_0$.

We say that a frame ϕ satisfies a set of tests S, and write $S \models_E \phi$ (or $S \models \phi$, if E is obvious), iff all the tests of S are true in ϕ.

Remark. Two frames ϕ and ϕ' with the same domain are statically equivalent iff $Eq_{all}(\phi) \models \phi'$ and $Eq_{all}(\phi') \models \phi$.

To produce an algorithm for static equivalence, we use $sat(\phi, fn(\phi))$ to generate a finite set of equations $Eq(\phi)$ that characterize the frame ϕ: that is, for any frame phi', ϕ' satisfies the tests in $sat(\phi, fn(\phi))$ iff it satisfies the equations $Eq(\phi)$.

Algorithm 2 describes the construction of the set $Eq(\phi)$.

```
Input: a frame $\phi = \nu\tilde{n}.\sigma$
Output: $Eq(\phi)$, a finite set of equations that characterizes $\phi$
begin
    $\phi \leftarrow \phi\downarrow_R$;
    $S' \leftarrow sat(\phi, fn(\phi))$;
    $Eq(\phi) \leftarrow \emptyset$;
    forall $F \in_{\equiv} S'$ do
        forall $F' \in_{\equiv} S'$ do
            Assume $vars(F) \cap vars(F') = \emptyset$;
            if $mgu(t, t') \neq \emptyset$ then
                let $\sigma$ be any element in $mgu(t, t')$;
                if $\sigma(x)$ is deducible from $\phi$ for all $x \in dom(\sigma)$ then
                    let $R_x$ be a recipe of $\sigma(x)$ in $\phi$, for any $x \in dom(\sigma)$;
                    let $R_x = x$ for any other $x \notin dom(\sigma)$;
                    $Eq(\phi) \leftarrow Eq(\phi) \cup \{(T\{R_{x_i}/X_i\}, T'\{R_{x_i}/X_i\})\}$
                end
            end
        end
    end
end
```

Algorithm 2: Computation of a finite set of equations that characterize a frame

Intuitively, in the above algorithm, for each ground term T in normal form that is deducible, we create tests that check that all of the recipes R_i of T lead to the same term.

The following conjecture states the validity of the above algorithm.

Conjecture 4. *For any frames ϕ and ϕ' it holds that $Eq(\phi, fn(\phi)) \models \phi'$ iff $Eq(\phi) \models \phi'$.*

The following two technical conjecture explain how we can restrict checking static equivalence by only checking the equations that contain a fixed set of free names.

Conjecture 5. *Consider $\Sigma_h = \Sigma \cup \{h/1\}$. We suppose the function symbol h does not appear in any equations (it models a hash function): the equational theory E_h we associate to Σ_h has the same description $\sim_E$ as E.*

If ϕ and ϕ' are frames over the signature Σ, then:
$\phi \approx_E \phi'$ *iff* $\phi \approx_{E_h} \phi'$

Conjecture 6. *If ϕ and ϕ' are two frames over Σ, then $\phi' \models_E Eq_{all}(\phi)$ iff $\phi' \models_{E_h} Eq(\phi, fn(\phi))$.*

5.2 Trapdoor bit commitment

A *bit commitment scheme* is a cryptographic protocol used by two agents A and B to communicate in the following way:

- Agent A wants to send a piece of information x (typically a bit) to agent B, but he does not want to divulge the information to B just yet.
- Instead A sends E, the encryption of x using a key y. B should not be able to find out the value of x from E without knowing y.
- When A is ready to reveal x, he sends to B the key y.

A good bit commitment scheme has the following properties:

- it is computationally infeasable for B to find out the value of x without knowing y.
- it is computationally infeasable for A to compute another key y' such that the decryption of E with y' will yield a value $x' \neq x$.

In a *trapdoor bit commitment scheme*, we relax the second hypothesis, by requiring that:

- if A knows a certain trapdoor z, which is created during the encryption process, it is computationally easy for him to compute another key y' such that the decryption of E with y' will yield any value x' desired by A.
- it is computationally infeasable for any agent not in possesion of the trapdoor information z to compute y' with the property above.

Trapdoor bit commitment schemes have applications in *e-voting* protocols. For example, such a scheme could be used to ensure *coercion-freeness*, by trapdoor commiting the value of the vote. At a later phase, the key for the trapdoor is sent to the *e-voting authority.* If the agent was coerced into voting a certain way, he can simply compute the y' corresponding to his real vote and reveal y' to the *e-voting authority.*

5.2.1 Modelling trapdoor bit commitment

To model a trapdoor bit commitment scheme using an equational theory, we use the following function symbols, defining a signature Σ_{td}:

- $\mathit{tdcommit}(x, y, z)$ represents the trapdoor commitment of the information x using the key y and the trapdoor z.
- $\mathit{open}(v, y)$ represents the opening of the commitment v using the key y.
- $f(x, y, z, x')$ represents the computation of a new key y', such that by opening $\mathit{tdcommit}(x, y, z)$ using y', we will obtain x'.

We associate the following equational theory E_{td} to the above signature:

$$\begin{aligned}
&open(tdcommit(x, y, z), y) = x\\
&tdcommit(x', f(x, y, z, x'), z) = tdcommit(x, y, z)
\end{aligned}$$

The first equation is very easily understood: it simply states that by opening a commitment using the right key we obtain the correct value. The second equation is trickier: it states that commiting to a certain value x, using key y and trapdoor z is *exactly* the same as commiting to another value x' using the newly computed key $f(x, y, z, x')$ and the same trapdoor z.

We associate to the equational theory induced by the above two equations a convergent term rewriting system R_{td}:

$$\begin{aligned}
&open(tdcommit(x, y, z), y) \to x\\
&open(tdcommit(x, y, z), f(x, y, z, x')) \to x'\\
&tdcommit(x', f(x, y, z, x'), z) \to tdcommit(x, y, z)\\
&f(x', f(x, y, z, x'), z, x'') \to f(x, y, z, x'')
\end{aligned}$$

5.2.2 Termination

We prove that for the rewrite system R_{td} the saturated set of deduction facts $sat(\phi, \mathcal{N}_0)$ is finite.

Indeed, we can easily check that all terms t such that a deduction fact of the form $[T \rhd t \mid X_i \rhd t_i]$ is added by the saturation rules are smaller than the terms that are already in the frame.

This ensures that the procedures described in Algorithm 1 and in Algorithm 2 terminate.

5.3 Implementation

We have created a prototype implementation of the saturation algorithm. The open source code will be available from *https://www.lsv.ens-cachan.fr/$\sim$ ciobaca/tdcommit/*.

6 Conclusions and future work

We have analyzed static equivalence, which is a formalism that can be used to model security properties like anonymity and strong secrecy in the passive case.

In section 3, we provide a composition result. We show that in the passive case and under certain reasonable assumptions, the composition of two protocols which are secure independently and which share a secret is secure.

This composition result is important because it allows us to obtain modular proof of security.

In section 4, we present work-in-progress semi-decision procedure for the intruder deduction problem and for the static equivalence problem, when the equational theory modelling the cryptographic primitives can be implemented by a convergent term rewriting system. We show that the semi-decision procedure terminates for an equational theory modelling trapdoor commitment.

We still need to finalize the formal proof of correctness. Also, it would be interesting to characterize the class of equational theories for which the semi-decision procedure presented in section 4 terminates.

In the future, it would be interesting to study the equivalents of the above problems in the active case.

References

[1] Anupam Datta, Ante Derek, John C. Mitchell, and Dusko Pavlovic. Secure protocol composition. In Michael Backes and David A. Basin, editors, *FMSE*, pages 11–23. ACM, 2003.

[2] Nancy A. Durgin, John C. Mitchell, and Dusko Pavlovic. A compositional logic for protocol correctness. In *CSFW*, pages 241–. IEEE Computer Society, 2001.

[3] Stéphanie Delaune, Steve Kremer, and Mark Ryan. Composition of password-based protocols. In *CSF*, pages 239–251. IEEE Computer Society, 2008.

[4] Martín Abadi and Véronique Cortier. Deciding knowledge in security protocols under equational theories. *Theor. Comput. Sci.*, 367(1-2):2–32, 2006.

[5] Véronique Cortier and Stéphanie Delaune. Deciding knowledge in security protocols for monoidal equational theories. In Nachum Dershowitz and Andrei Voronkov, editors, *LPAR*, volume 4790 of *Lecture Notes in Computer Science*, pages 196–210. Springer, 2007.

[6] Martín Abadi and Vronique Cortier. Deciding knowledge in security protocols under (many more) equational theories. In *Proceedings of the 18th IEEE Computer Security Foundations Workshop*, pages 62–76. IEEE, 2005.

[7] Danny Dolev and Andrew Chi-Chih Yao. On the security of public key protocols (extended abstract). In *FOCS*, pages 350–357. IEEE, 1981.

[8] Martín Abadi and Cédric Fournet. Mobile values, new names, and secure communication. In *Proceedings of the 28th ACM Symposium on Principles of Programming Languages*, pages 104–115. ACM, 2001.

[9] Stéphanie Delaune. Easy intruder deduction problems with homomorphisms. *Inf. Process. Lett.*, 97(6):213–218, 2006.

[10] Pascal Lafourcade, Denis Lugiez, and Ralf Treinen. Intruder deduction for the equational theory of abelian groups with distributive encryption. *Inf. Comput.*, 205(4):581–623, 2007.

[11] Pascal Lafourcade. Intruder deduction for the equational theory of exclusive-or with commutative and distributive encryption. *Electr. Notes Theor. Comput. Sci.*, 171(4):37–57, 2007.