Lengths may break privacy – or how to check for equivalences with length *

Vincent Cheval[1], Véronique Cortier[2], and Antoine Plet[2]

[1] LSV, ENS Cachan & CNRS & INRIA, France
[2] LORIA, CNRS, France

Abstract. Security protocols have been successfully analyzed using symbolic models, where messages are represented by terms and protocols by processes. Privacy properties like anonymity or untraceability are typically expressed as equivalence between processes. While some decision procedures have been proposed for automatically deciding process equivalence, all existing approaches abstract away the information an attacker may get when observing the length of messages.
In this paper, we study process equivalence with length tests. We first show that, in the static case, almost all existing decidability results (for static equivalence) can be extended to cope with length tests. In the active case, we prove decidability of trace equivalence with length tests, for a bounded number of sessions and for standard primitives. Our result relies on a previous decidability result from Cheval *et al* [15] (without length tests). Our procedure has been implemented and we have discovered a new flaw against privacy in the biometric passport protocol.

1 Introduction

Privacy is an important concern in our today's life where many documents and transactions are digital. For example, we are usually carrying RFIDs cards (for ground transportation, access to office buildings, for opening modern cars, etc.). Due to these cards, malicious users may (attempt to) track us or learn more about us. For instance, the biometric passport contains a chip that stores sensitive information such as birth date, nationality, picture, fingerprints, and also iris characteristics. In order to protect passport holders privacy, the application (or protocol) deployed on biometric passports is designed to achieve authentication without revealing any information to a third party (data is sent encrypted). However, it is well known that designing security protocols is error prone. For example, it was possible to track French citizens due to an additional error message introduced in French passports [5]. Symbolic models have been very successful for analyzing security protocols. Several automatic tools have been designed such as ProVerif [10], Avispa [6], *etc.*. They are very effective to detect flaws or prove

* This work has been partially supported by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC grant agreement no 258865, project ProSecure and project JCJC VIP n^o 11 JS02 006 01

security of many real-case studies (e.g JFK [2], OAuth2.0 [7], *etc.*). However, these tools are, for most of them, dedicated to accessibility properties. While data secrecy or authentication can be easily expressed as accessibility properties, privacy properties are instead stated as indistinguishability (or equivalence) properties: Alice remains anonymous if an attacker cannot distinguish a session with Alice as participant from a session with Bob as participant. The literature on how to decide equivalence of security protocols is much less prolific than for accessibility. Some procedures have been proposed [8,15,12,11] for some classes of cryptographic primitives, not all procedures being guaranteed to terminate. However, none of these results take into account the fact that an attacker can always observe the length of a message. For example, even if k is a secret key, the cyphertext $\{n\}_k$ corresponding to the encryption of a random number n by the key k can always be distinguished from the cyphertext $\{n, n\}_k$ corresponding to the encryption of a random number n repeated twice by the key k. This is simply due to the fact that $\{n, n\}_k$ is longer than $\{n\}_k$. These two messages would be considered as indistinguishable in all previous mentioned symbolic approaches. The fact that encryption reveals the length of the underling plaintext is a well-identified issue in cryptography. Therefore and not surprisingly, introducing a length function becomes necessary in symbolic models when proving that symbolic process equivalence implies cryptographic indistinguishability [16].

Our contributions. In this paper, we consider an equivalence notion that takes into account the information leaked by the length of a message. More precisely, we equip the term algebra T with a length function $\ell : T \mapsto \mathbb{R}^+$ that associates a non negative real number to any term and we let the attacker compare the length of any two messages he can construct. As usual, the properties of the cryptographic primitives are modeled through an equational theory. For example, the equation $\mathsf{sdec}(\mathsf{senc}(m, k), k)$ models the fact that decrypting with a key k a message m (symmetrically) encrypted by k yields the message m in clear. The goal of our paper is to study the decidability of equivalence with length tests.

The simplest case is the so-called *static case*, where an attacker can only observe protocol executions. Two sequences of messages are *statically equivalent* if an attacker cannot see the difference between them. For example, the two messages $\{0\}_k$ and $\{1\}_k$ are distinct but cannot be distinguished by an attacker unless he knows the key k. We show how most existing decidability results for static equivalence can be extended to length tests. We simply require the length function to be homomorphic, that is, the length $\ell(M)$ of a term $M = f(M_1, \ldots, M_k)$ is a function of f and the lengths of $M_1, \ldots, M_k$. We show that whenever static equivalence is decidable for some equational theory E then static equivalence remains decidable when adding length tests. The result requires a simple hypothesis called SET-stability that is satisfied by most equational theories that have been showed decidable for static equivalence. As an application, we deduce decidability of static equivalence for many primitives, including symmetric and asymmetric encryption, signatures, hash, blind signatures, exclusive or, *etc.*

The *active case*, where an attacker can freely interact with the protocol, is of course more involved. Even without the introduction of a length function, there are very few decidability results [17,15]. Starting from the decision procedure developed in [15], we show how to deal with length functions for the standard cryptographic primitives (symmetric and asymmetric encryption, signatures, hash, and concatenation). Like for the static case, our result is actually very modular. In order to check whether two protocols P and Q are in trace equivalence with length tests, it is sufficient to first run the procedure of [15], checking whether two protocols P and Q are in trace equivalence without length tests. It is then sufficient to check for equalities of the polynomials we derive from the processes that appear in the final states of the procedure of [15]. As such, we provide a decision procedure for the two following problems: (1) Given two processes P and Q and a length function ℓ, are P and Q in trace equivalence with length tests (w.r.t. the length function ℓ)? (2) Given two processes P and Q, does there exist a length function ℓ such that P and Q in not trace equivalence with length tests (w.r.t. the length function ℓ)? From a practical point of view, this amounts into deciding whether there exists an implementation of the primitives (that would meet some particular length property) such that an attacker could distinguish between P and Q, leading to a privacy attack. We have implemented our decision procedure for trace equivalence with length tests as an extension of the APTE tool developped for [15]. As an application, we study the biometric passport [1] and discover a new flaw. We show that an attacker can break privacy by observing messages of different lengths depending on which passport is used, therefore discovering who between Alice or Bob is currently using her/his passport.

Related work. Existing decision procedures for trace equivalence do not consider length tests. [15] shows that trace equivalence is decidable for finitely many sessions and for a fixed term algebra (encryption, signatures, hash, ...). A procedure for a more flexible term algebra is provided in [12] but is not guaranteed to terminate. Building on [8], it has been shown that trace equivalence can be decided for any convergent subterm equational theories, for protocols with no else branches [17]. The tool ProVerif [10,11] is also able to check for equivalence but is again not guaranteed to terminate (and prove an equivalence that is sometimes too strong). One of the only symbolic models that introduce a length function is the model developed in [16] for proving that symbolic process equivalence implies cryptographic indistinguishability. However, [16] does not discuss any decision procedure for process equivalence.

2 Preliminaries

A key ingredient of formal models for security protocols is the representation of messages by *terms*. This section is devoted to the definitions of terms and two key notions of knowledge for the attacker: deduction and static equivalence.

2.1 Terms

Given a *signature* $\mathcal{F}$ (*i.e.* a finite set of function symbols, with a given arity), an infinite set of *names* $\mathcal{N}$, and an infinite set of variables $\mathcal{X}$, the set of terms

$T(\mathcal{F}, \mathcal{N}, \mathcal{X})$ is defined as the union of names $\mathcal{N}$, variables $\mathcal{X}$, and function symbols of $\mathcal{F}$ applied to other terms. A term is said to be ground if it contains no variable. $\tilde{n}$ denotes a set of names. The set of names of a term M is denoted by $\mathit{fnames}(M)$. Substitutions are replacement of variables by terms and are denoted by $\theta = \{M_1/x_1, \ldots, M_k/x_k\}$. The application of a substitution θ to a term M is defined as usual and is denoted $M\theta$. A *context* C is a term with holes. Given terms $M_1, \ldots, M_k$, the term $C[M_1, \ldots, M_k]$ may be denoted $C[\tilde{M_i}]$.

Example 1. A signature for modelling the standard cryptographic primitives (symmetric and asymmetric encryption, concatenation, signatures, and hash) is $\mathcal{F}_{\mathsf{stand}} = \mathcal{F}_c \cup \mathcal{F}_d$ where $\mathcal{F}_c$ and $\mathcal{F}_d$ are defined as follows (the second argument being the arity):

$$\mathcal{F}_c = \{\mathsf{senc}/2,\ \mathsf{aenc}/2,\ \mathsf{pk}/1,\ \mathsf{sign}/2,\ \mathsf{vk}/1,\ \langle\,\rangle/2,\ \mathsf{h}/1\}$$
$$\mathcal{F}_d = \{\mathsf{sdec}/2,\ \mathsf{adec}/2,\ \mathsf{check}/2,\ \mathsf{proj}_1/1,\ \mathsf{proj}_2/1\}.$$

The function aenc (resp. senc) represents asymmetric encryption (resp. symmetric encryption) with corresponding decryption function adec (resp. sdec) and public key pk. Concatenation is represented by $\langle\,\rangle$ with associated projectors proj_1 and proj_2. Signature is modeled by the function sign with corresponding validity check check and verification key vk. h represents the hash function.

The properties of the cryptographic primitives (e.g. decrypting an encrypted message yields the message in clear) are expressed through equations. Formally, we equip the term algebra with an *equational theory*, that is, an equivalence relation on terms which is closed under substitutions for variables and names. We write $M =_E N$ when the terms M and N are equivalent modulo E. Equational theories can be used to specify a large variety of cryptographic primitives, from the standard cryptographic primitives of Example 1 to exclusive or (XOR), blind signatures, homomorphic encryption, trapdoor-commitment or Diffie-Hellman. We provide below a theory for the standard primitives and for XOR. More examples of equational theories can be found in [3,4].

Example 2. Continuing Example 1, the equational theory E_{stand} for the standard primitives is defined by the equations:

$$\mathsf{sdec}(\mathsf{senc}(x, y), y) = x \qquad (1)$$
$$\mathsf{adec}(\mathsf{aenc}(x, \mathsf{pk}(y)), y) = x \qquad (2)$$
$$\mathsf{check}(\mathsf{sign}(x, y), \mathsf{vk}(y)) = x \qquad (3)$$
$$\mathsf{proj}_1(\langle x, y\rangle) = x \qquad (4)$$
$$\mathsf{proj}_2(\langle x, y\rangle) = y \qquad (5)$$

Equation 1 models that decrypting an encrypted message $\mathsf{senc}(m, k)$ with the right key k yields the message m in clear. Equation 2 is the asymmetric analog of Equation 1. Similarly, Equations 4 and 5 model the first and second projections for concatenation. There are various ways for modeling signature. Here, Equation 3 models actually two properties. First, the validity of a signature $\mathsf{sign}(m, k)$ given the verification key $\mathsf{vk}(k)$ can be checked by applying the test function check. Second, the underlying message m under signature can be retrieved (as it is often the case in symbolic models). This is because we assume that a signature $\mathsf{sign}(m, k)$, which represents the digital signature itself, is always sent together with the corresponding message m.

Example 3. The theory of XOR $E_\oplus$, is based on the signature $\Sigma = \{\oplus/2, 0/0\}$ and the equations:

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

The two left equations model the fact that the function $\oplus$ is *associative* and *commutative*. The right equations model the fact that XORing twice the same element yields the neutral element 0.

A function symbol $+$ is said to be AC (associative and commutative) if it satisfies the two equations $(x + y) + z = x + (y + z)$ and $x + y = y + x$. For example, the symbol $\oplus$ is an AC-symbol of the theory $E_\oplus$. Given an equational theory E, we write $M =_{AC} N$ if M and N are equal modulo the associativity and commutativity of their AC-symbols.

2.2 Deduction and static equivalence

During protocol executions, the attacker learns sequences of messages $M_1, \ldots, M_k$ where some names are initially unknown to him. This is modeled by defining a *frame* ϕ to be an expression of the form

$$\phi = \nu\tilde{n}\{M_1/x_1, \ldots, M_k/x_k\}$$

where $\tilde{n}$ is a set of names (representing the secret names) and the M_i are terms. A frame is *ground* is all its terms are ground. The *domain* of the frame ϕ is $\mathrm{dom}(\phi) = \{x_1, \ldots, x_n\}$.

The first basic notion when modeling the attacker is the notion of *deduction*. It captures what an attacker can built from a frame ϕ. Intuitively, the attacker knows all the terms of ϕ and can apply any function to them.

Definition 1 (deduction). *Given an equational theory E and a frame $\phi = \nu\tilde{n}\sigma$, a ground term N is* deducible *from ϕ, denoted $\phi \vdash N$, if there is a* free *term M (i.e. $fnames(M) \cap \tilde{n} = \emptyset$), such that $M\sigma =_E N$. The term M is called a* recipe *of N.*

Example 4. Consider $\phi_1 = \nu n, k, k'\{k/x_1, \mathsf{senc}(\langle n, n\rangle, k)/x_2, \mathsf{senc}(n, k')/x_3\}$. Then $\phi_1 \vdash k$, $\phi_1 \vdash n$, but $\phi_1 \not\vdash k'$. A recipe for k is x_1 while a recipe for n is $\mathsf{proj}_1(\mathsf{sdec}(x_2, x_1))$. Another possible recipe of n is $\mathsf{proj}_2(\mathsf{sdec}(x_2, x_1))$.

As mentioned in the introduction, the confidentiality of a vote v or the anonymity of an agent a cannot be defined as the non deducibility of v or a. Indeed, both are in general public values and are thus always deducible. Instead, the standard approach consists in defining privacy based on an indistinguishability notion: an execution with a should indistinguishable from an execution with b. Indistinguishability of sequences of terms is formally defined as static equivalence.

Definition 2 (static equivalence). *Two frames $\phi_1 = \nu\tilde{n_1}\sigma_1$ and $\phi_2 = \nu\tilde{n_2}\sigma_2$ are* statically equivalent*, denoted $\phi_1 \sim \phi_2$, if and only if for all terms M, N such that $(fn(M) \cup fn(N)) \cap (\tilde{n_1} \cup \tilde{n_2}) = \emptyset$,*

$$(M\sigma_1 =_E N\sigma_1) \Leftrightarrow (M\sigma_2 =_E N\sigma_2).$$

Example 5. Let $\phi_2 = \nu n, n', k, k'\{k/x_1, \mathsf{senc}(\langle n', n\rangle, k)/x_2, \mathsf{senc}(n, k')/x_3\}$ and $\phi_3 = \nu n, k, k'\{k/x_1, \mathsf{senc}(\langle n, n\rangle, k)/x_2, \mathsf{senc}(\langle n, n\rangle, k')/x_3\}$. ϕ_1 is defined in Example 4. Then $\phi_1 \not\sim \phi_2$ since $\mathsf{proj}_1(\mathsf{sdec}(x_2, x_1)) = \mathsf{proj}_2(\mathsf{sdec}(x_2, x_1))$ is true in ϕ_1 but not in ϕ_2. Intuitively, an attacker may distinguish between ϕ_1 and ϕ_2 by decrypting the second message and noticing that the two components are equal for ϕ_1 while they differ for ϕ_2. Conversely, we have $\phi_1 \sim \phi_3$.

2.3 Rewrite systems

To decide deduction and static equivalence, it is often convenient to reason with a rewrite system instead of an equational theory. A *rewrite system* $\mathcal{R}$ is a set of rewrite rules $l \to r$ (where l and r are terms) that is closed by substitution and context. Formally a term u can be rewritten in v, denoted by $u \to_{\mathcal{R}} v$ if there exists $l \to r \in \mathcal{R}$, a substitution θ, and a position p of u such that $u|_p = l\theta$ and $v = u[r\theta]_p$. The transitive and reflexive cloture of $\to_{\mathcal{R}}$ is denoted $\to^*_{\mathcal{R}}$. We write $\to$ instead of $\to_{\mathcal{R}}$ when $\mathcal{R}$ is clear from the context.

Definition 3 (convergent). *A rewrite system $\mathcal{R}$ is* convergent *if it is:*

- terminating*: there is no infinite sequence $u_1 \to u_2 \to \cdots \to u_n \to \cdots$*
- confluent*: for every terms u, u_1, u_2 such that $u \to u_1$ and $u \to u_2$, there exists v such that $u_1 \to^* v$ and $u_2 \to^* v$.*

For a convergent rewrite system, a term t has a unique normal form *$t\!\downarrow$ such that $t \to^* t\!\downarrow$ and $t\!\downarrow$ has no successor.*

An equational theory E is convergent *if there exists a finite convergent rewrite system $\mathcal{R}$ such that for any two terms u, v, we have $u =_E v$ if and only if $u\!\downarrow = v\!\downarrow$.*

For example, the theory E_{stand} defined in Example 2 is convergent. Its associated finite convergent rewrite system is obtained by orienting the equations from left to right. Conversely, the theory $E_{\oplus}$ defined in Example 3 is not convergent due the equations of associativity and commutativity. Since many equational theories modeling cryptographic primitives do have associative and commutative symbols, we define *rewriting modulo AC* as $M \to_{AC} N$ if there is a term M' such that $M =_{AC} M'$ and $M' \to N$. AC-convergence can then be defined similarly to convergence.

Definition 4 (AC-convergent). *A rewrite system $\mathcal{R}$ is* AC-convergent *if it is:*

- AC-terminating*: there is no infinite sequence $u_1 \to_{AC} \cdots \to_{AC} u_n \to_{AC} \cdots$*
- AC-confluent*: for every terms u, u_1, u_2 such that $u \to_{AC} u_1$ and $u \to_{AC} u_2$, there exists v such that $u_1 \to^*_{AC} v$ and $u_2 \to^*_{AC} v$.*

For a AC-convergent rewrite system, a term t has a unique set of normal forms $t\downarrow_{AC}= \{t' \mid t \to^* t'$ *and* t' *has no successor*$\}$. *For any* $u, v \in t\downarrow_{AC}$, $u =_{AC} v$.

An equational theory E is AC-convergent *if there exists a finite AC-convergent rewrite system $\mathcal{R}$ such that for any two terms u, v, we have $u =_E v$ if and only if $u\downarrow_{AC}= v\downarrow_{AC}$.*

For example, the theory $E_\oplus$ defined Example 3 is AC-convergent. Its associated finite AC-convergent rewrite system is obtained by orienting the two right equations from left to right. Of course, any convergent theory is AC-convergent. Most, if not all, equational theories for cryptographic primitives are convergent or at least AC-convergent. So in what follows, we only consider AC-convergent theories.

3 Length equivalence - static case

While many decidability results have been provided for deduction and static equivalence, for various theories, none of them study the leak induced by the length of messages. In this section, we provide a definition for length functions and we study how to extend existing decidability results to length functions.

3.1 Length function

A *length function* is simply a function $\ell : T(\mathcal{F}, \mathcal{N}, \mathcal{X}) \to \mathbb{R}^+$ that associates non-negative real numbers to terms. A meaningful length function should associate the same length to terms that are equal modulo the equational theory. Since we consider AC-convergent theories, we assume that the length of a term t is evaluated by an auxiliary function applied once t is in normal form. Moreover, the size of a term $f(M_1, \ldots, M_k)$ is typically a function that depends on f and the length of $M_1 \ldots, M_k$. This class of length functions is called *normalized length functions*.

Definition 5 (Normalized length function). *Let $T(\mathcal{F}, \mathcal{N}, \mathcal{X})$ be a term algebra and E be an AC-convergent equational theory. A length function ℓ is a* normalized length function *if there exists a function $\ell_{\mathsf{aux}} : T(\mathcal{F}, \mathcal{N}, \mathcal{X}) \to \mathbb{R}^+$ (called auxiliary length function) such that the following properties hold:*

1. *ℓ_{aux} is a morphism, that is, for every function symbol f of arity k, there exists a function $\ell_f : {\mathbb{R}^+}^k \to \mathbb{R}^+$ s.t. for all terms $M_1, \ldots, M_k$ $\ell_{\mathsf{aux}}(f(M_1, \ldots, M_k)) = \ell_f(\ell_{\mathsf{aux}}(M_1), \ldots, \ell_{\mathsf{aux}}(M_k))$*
2. *ℓ_{aux} is stable modulo AC: $\ell_{\mathsf{aux}}(M) = \ell_{\mathsf{aux}}(N)$ for all M, N s.t. $M =_{AC} N$.*
3. *ℓ_{aux} decreases with rewriting: $\ell_{\mathsf{aux}}(M) \geq \ell_{\mathsf{aux}}(N)$ for all M, N s.t. $M \to_{AC} N$.*
4. *ℓ coincides with ℓ_{aux} on normal forms: $\ell(M) = \ell_{\mathsf{aux}}(M\downarrow_{AC})$ where $\ell_{\mathsf{aux}}(M\downarrow_{AC})$ is defined to be $\ell_{\mathsf{aux}}(N)$ for any $N \in M\downarrow_{AC}$.*
5. *For any $r \in \mathbb{R}^+$, the set $\{n \in \mathcal{N} \mid \ell(n) = r\}$ is either infinite or empty. A name should not be particularized by its length.*

Note that item 5 implies in particular that $\ell_{\mathsf{aux}}(M) = \ell_{\mathsf{aux}}(M\sigma)$ for any σ that replaces the names of M by names of equal length (i.e. such that $\ell_{\mathsf{aux}}(\sigma(n) = \ell_{\mathsf{aux}}(n))$. Indeed, the length should not depend of the choice of names.

Example 6. A natural length function for the standard primitives defined in Example 2 is ℓ_{stand} induced by the following auxiliary length function ℓ_{aux}:

$$\begin{aligned}
\ell_{\mathsf{aux}}(n) &= 1 && n \in \mathcal{N}\\
\ell_{\mathsf{aux}}(\mathsf{senc}(u,v)) &= \ell_{\mathsf{aux}}(u) + \ell_{\mathsf{aux}}(v)\\
\ell_{\mathsf{aux}}(\langle u,v\rangle) &= 1 + \ell_{\mathsf{aux}}(u) + \ell_{\mathsf{aux}}(v)\\
\ell_{\mathsf{aux}}(\mathsf{aenc}(u,v)) &= 2 + \ell_{\mathsf{aux}}(u) + \ell_{\mathsf{aux}}(v)\\
\ell_{\mathsf{aux}}(\mathsf{sign}(u,v)) &= 3 + \ell_{\mathsf{aux}}(u) + \ell_{\mathsf{aux}}(v)\\
\ell_{\mathsf{aux}}(f(u,v)) &= 1 + \ell_{\mathsf{aux}}(u) + \ell_{\mathsf{aux}}(v) && f \in \{\mathsf{sdec}, \mathsf{adec}, \mathsf{check}\}\\
\ell_{\mathsf{aux}}(f(u)) &= 1 + \ell_{\mathsf{aux}}(u) && f \in \{\mathsf{proj}_1, \mathsf{proj}_2\}
\end{aligned}$$

Then the length of a term M is simply the auxiliary length of its normal form: $\ell(M) = \ell_{\mathsf{aux}}(M\downarrow)$ and ℓ is a normalized length function. Note that the constants 1, 2, 3 are rather arbitrary and ℓ would be a normalized length function for any other choice. The choice of the exact parameters typically depends on the implementation of the primitives.

Example 7. A length function for XOR is $\ell_{\oplus}$, induced by the auxiliary function ℓ_{aux} defined by $\ell_{\mathsf{aux}}(n) = 1$ for n name, $\ell_{\mathsf{aux}}(0) = 0$, and $\ell_{\mathsf{aux}}(u \oplus v) = \max(\ell_{\mathsf{aux}}(u), \ell_{\mathsf{aux}}(v))$. Then $\ell_{\oplus}$ is again a normalized length function.

An attacker may compare the length of messages, which gives him additional power. For example, the frames ϕ_1 and ϕ_3 (defined in Example 5) are statically equivalent. However, in reality, an attacker would notice that the third messages are of different length. In particular, $\ell_{\mathsf{stand}}(\mathsf{senc}(n, k')) = 2$ while $\ell_{\mathsf{stand}}(\mathsf{senc}(\langle n, n\rangle, k')) = 4$ (where ℓ_{stand} has been defined in Example 6).

We extend the notion of static equivalence to take into account the ability of an attacker to check for equality of lengths.

Definition 6 (static equivalence w.r.t. length). *Two frames $\phi_1 = \nu\tilde{n}_1\sigma_1$ and $\phi_2 = \nu\tilde{n}_2\sigma_2$ are* statically equivalent w.r.t. the length function ℓ, *denoted $\phi_1 \sim^{\ell} \phi_2$, if ϕ_1 and ϕ are statically equivalent ($\phi_1 \sim \phi_2$) and for all terms M, N such that $(fn(M) \cup fn(N)) \cap (\tilde{n_1} \cup \tilde{n_2}) = \emptyset$,*

$$(\ell(M\sigma_1) =_E \ell(N\sigma_1)) \Leftrightarrow (\ell(M\sigma_2) =_E \ell(N\sigma_2)).$$

3.2 Decidability

Ideally, we would like to inherit any decidability result that exists for the usual static equivalence $\sim$. We actually need to look deeper in how decidability results are obtained for $\sim$. In many approaches (e.g. [3,9]), decidability of static equivalence is obtained by computing from a frame ϕ, an upper set that symbolically describes the set of all deducible subterms. Here, we generalize this property into SET-stability.

Definition 7 (SET-stable). *An equational theory E is* SET-stable *if for any frame $\phi = \nu\tilde{n}\{M_1/x_1, \ldots, M_k/x_k\}$ there exists a set $\mathsf{SET}(\phi)$ such that:*

- $M_1, \ldots, M_k \in \mathsf{SET}(\phi)$,
- $\forall M \in \mathsf{SET}(\phi), \phi \vdash M$,
- *for any finite set of names $\tilde{n'} \supseteq \tilde{n}$, for every context C_1 such that $fn(C_1) \cap \tilde{n'} = \emptyset$, for all $N_i^1 \in \mathsf{SET}(\phi)$, for all $T \in (C_1[\tilde{N_i^1}]\downarrow)$, there exist a context C_2 such that $fn(C_2) \cap \tilde{n'} = \emptyset$ and terms $N_i^2 \in \mathsf{SET}(\phi)$ such that $T =_{AC} C_2[\tilde{N_i^2}]$.*

We say that E is efficiently SET-stable *if there is an algorithm that computes the set $\mathsf{SET}(\phi)$ being given a frame ϕ and that computes a recipe ζ_M for any $M \in \mathsf{SET}(\phi)$.*

We are now ready to state our first main theorem.

Theorem 1. *Let E be an efficiently SET-stable equational theory and ℓ be a normalized length function. If $\sim_E$ is decidable then $\sim_E^\ell$ is decidable.*

Sketch of proof The algorithm for checking for $\sim_E^\ell$ works as follows. Given two frames $\phi_1 = \nu\tilde{n}\sigma_1$ and $\phi_2 = \nu\tilde{n}\sigma_2$,

- check whether $\phi_1 \sim_E \phi_2$
- compute $\mathsf{SET}(\phi_1)$ and $\mathsf{SET}(\phi_2)$;
- for any $M \in \mathsf{SET}(\phi_1)$, compute its corresponding recipe ζ_M and check whether $\ell(\zeta_M\sigma_2) = \ell(M)$;
- symmetrically, for any $M \in \mathsf{SET}(\phi_2)$, compute its corresponding recipe ζ_M and check whether $\ell(\zeta_M\sigma_1) = \ell(M)$;
- return true if all checks succeeded and false otherwise.

The algorithm returns true if $\phi_1 \sim_E^\ell \phi_2$. Indeed, for any $M \in \mathsf{SET}(\phi_1)$, $\ell(\zeta_M\sigma_1) = \ell(M) = \ell(M_0)$ where M_0 is length-preserving renaming of $M\downarrow$ with free names only. $\ell(\zeta_M\sigma_1) = \ell(M_0\sigma_1)$ implies $\ell(\zeta_M\sigma_2) = \ell(M_0\sigma_2) = \ell(M_0) = \ell(M)$.

The converse implication is more involved and makes use of the properties of the sets $\mathsf{SET}(\phi_1)$ and $\mathsf{SET}(\phi_2)$. □

Applying Theorem 1 we can deduce the decidability of $\sim_E^\ell$ for any theory E described in [3], e.g. theories for the standard primitives, for XOR, for pure AC, for blind signatures, homomorphic encryption, addition, *etc.* More generally, we can infer decidability for any *locally stable* theories, as defined in [3]. Intuitively, locally-stability is similar to SET-stability except that only small contexts are considered. Locally-stability is easier to check than SET-stability and has been shown to imply SET-stability in [3].

Corollary 1. *Let E be a locally-stable equational theory as defined in [3]. Let ℓ be a normalized length function. If $\sim_E$ is decidable then $\sim_E^\ell$ is decidable.*

4 Length equivalence - active case

We now study length equivalence in the active case, that is when an attacker may fully interact with the protocol under study. We first define our process algebra, in the spirit of the applied-pi calculus [4].

4.1 Syntax

We consider $\mathcal{F}_d$ as defined in Example 1 and $\mathcal{F}'_c \supseteq \mathcal{F}_c$. We let $\mathcal{F}'_c$ contain more primitives than $\mathcal{F}_c$, to allow for constants or free primitives such as mac. We consider the fixed equational theory E_{stand} as defined in Example 2. Orienting the equations of E_{stand} from left to right yields a convergent rewrite system.

The *constructor terms*, resp. *ground constructor terms*, are those in $\mathcal{T}(\mathcal{F}'_c, \mathcal{N} \cup \mathcal{X})$, resp. in $\mathcal{T}(\mathcal{F}'_c, \mathcal{N})$. A ground term u is called a *message*, denoted $\mathsf{Message}(u)$, if $v{\downarrow}$ is a constructor term for all $v \in st(u)$. For instance, the terms $\mathsf{sdec}(a, b)$, $\mathsf{proj}_1(\langle a, \mathsf{sdec}(a, b)\rangle)$, and $\mathsf{proj}_1(a)$ are not messages. Intuitively, we view terms as modus operandi to compute bitstrings where we use the call-by-value evaluation strategy.

The grammar of our *plain processes* is defined as follows:

$$P, Q := 0 \mid (P \,|\, Q) \mid P + Q \mid \mathsf{in}(u, x).P \mid \mathsf{out}(u, v).P \mid \mathsf{if}\ u_1 = u_2\ \mathsf{then}\ P\ \mathsf{else}\ Q$$

where u_1, u_2, u, v are terms, and x is a variable of $\mathcal{X}$. Our calculus contains parallel composition $P \mid Q$, choice $P+Q$, tests, input $\mathsf{in}(u, x).P$, and output $\mathsf{out}(u, v)$. Since we do not consider restriction, private names can simply be specified before hand so there is no need for name restriction. Trivial else branches may be omitted.

Definition 8 (process). *A* process *is a triple* $(\mathcal{E}; \mathcal{P}; \Phi)$ *where:*

- $\mathcal{E}$ *is a set of names that represents the private names of* $\mathcal{P}$*;*
- Φ *is a ground frame with domain included in* $\mathcal{AX}$*. It represents the messages available to the attacker;*
- $\mathcal{P}$ *is a multiset of* closed plain processes.

4.2 Semantics

The semantics for processes is defined as usual. Due to space limitations, we only provide two illustrative rules (see [13] or the appendix for the full definition).

$$(\mathcal{E}; \{\mathsf{in}(u, x).Q\} \uplus \mathcal{P}; \Phi) \xrightarrow{in(N,M)} (\mathcal{E}; \{Q\{x \mapsto t\}\} \uplus \mathcal{P}; \Phi) \qquad (\textsc{In}_c)$$
$$\text{if } M\Phi = t,\ fvars(M, N) \subseteq \mathrm{dom}(\Phi),\ fnames(M, N) \cap \mathcal{E} = \emptyset$$
$$N\Phi{\downarrow} = u{\downarrow},\ \mathsf{Message}(M\Phi),\ \mathsf{Message}(N\Phi), \text{ and } \mathsf{Message}(u)$$

$$(\mathcal{E}; \{\mathsf{out}(u, t).Q\} \uplus \mathcal{P}; \Phi) \xrightarrow{\nu ax_n.out(M, ax_n)} (\mathcal{E}; \{Q\} \uplus \mathcal{P}; \Phi \cup \{ax_n \rhd t\}) \ (\textsc{Out}_c)$$
$$\text{if } M\Phi{\downarrow} = u{\downarrow},\ \mathsf{Message}(u),\ fvars(M) \subseteq \mathrm{dom}(\Phi),\ fnames(M) \cap \mathcal{E} = \emptyset$$
$$\mathsf{Message}(M\Phi),\ \mathsf{Message}(t) \text{ and } ax_n \in \mathcal{AX},\ n = |\Phi| + 1$$

where u, v, t are ground terms, and x is a variable. The $\xrightarrow{w}$ relation is then defined as usual as the reflexive and transitive closure of $\rightarrow$, where w is the concatenation of all non silent actions.

The set of traces of a process $A = (\mathcal{E}; \mathcal{P}_1; \Phi_1)$ is the set of the possible sequences of actions together with the resulting frame.

$$\mathsf{trace}(A) = \{(s, \nu\mathcal{E}.\Phi_2) \mid (\mathcal{E}; \mathcal{P}_1; \Phi_1) \stackrel{s}{\Rightarrow} (\mathcal{E}; \mathcal{P}_2; \Phi_2) \text{ for some } \mathcal{P}_2, \Phi_2\}$$

4.3 Equivalence

Some terms such as $\mathsf{sdec}(\langle a, b\rangle, k)$ or $\mathsf{sdec}(\mathsf{senc}(a, k'), k)$ do not corresponding to actual messages since the corresponding computation would typically fail and return an error message. It would not make sense to compare the length of such decoy messages. We therefore adapt the notion of static equivalence in order to compare only lengths of terms that correspond to actual messages.

Definition 9. *Let $\mathcal{E}$ a set of private names. Let Φ and Φ' two frames. We say that $\nu\mathcal{E}.\Phi$ and $\nu\mathcal{E}.\Phi'$ are statically equivalent w.r.t. a length function ℓ, written $\nu\mathcal{E}.\Phi \sim_c^\ell \nu\mathcal{E}.\Phi'$, when* $\mathrm{dom}(\Phi) = \mathrm{dom}(\Phi')$ *and when for all terms M, N such that $fvars(M, N) \subseteq \mathrm{dom}(\Phi)$ and $fnames(M, N) \cap \mathcal{E} = \emptyset$, we have:*

- $\mathsf{Message}(M\Phi)$ *if and only if* $\mathsf{Message}(M\Phi')$
- *if* $\mathsf{Message}(M\Phi)$ *and* $\mathsf{Message}(N\Phi)$ *then*
  - $M\Phi\downarrow = N\Phi\downarrow$ *if and only* $M\Phi'\downarrow = N\Phi'\downarrow$*; and*
  - $\ell(M\Phi\downarrow) = \ell(N\Phi\downarrow)$ *if and only if* $\ell(M\Phi'\downarrow) = \ell(N\Phi'\downarrow)$.

Two processes A and B are in trace equivalence if any sequence of actions of A can be matched by the same sequence of actions in B such that the resulting frames are statically equivalent.

Definition 10 (trace equivalence w.r.t. length $\approx^\ell$). *Let A and B be processes with the same set of private names $\mathcal{E}$. $A \sqsubseteq^\ell B$ if for every $(s, \nu\mathcal{E}.\Phi) \in \mathit{trace}_c(A)$, there exists $(s, \nu\mathcal{E}.\Phi') \in \mathit{trace}(B)$ such that $\nu\mathcal{E}.\Phi \sim_c^\ell \nu\mathcal{E}.\Phi'$.*

Two closed processes A and B are trace equivalent w.r.t. the length function *ℓ, denoted by $A \approx^\ell B$, if $A \sqsubseteq^\ell B$ and $B \sqsubseteq^\ell A$.*

The length functions associated to standard primitives usually follow a simple pattern (see e.g. Example 6). We focus on *linear* length functions, that have been proved sound w.r.t. symbolic models [16]. A linear function is a function ℓ such that for any $f \in \mathcal{F}'_c$, $\ell(f(t_1, \ldots, t_n)) = l_f(\ell(t_1), \ldots, \ell(t_n))$ where $l_f(x_1, \ldots, x_n) = \beta^f + \sum_{i=1}^n \alpha_i^f x_i$ for some $\alpha_1^f, \ldots, \alpha_n^f, \beta^f \in \mathbb{R}^+$. Moreover, we assume that hashed messages are of fixed size: $\ell(\mathsf{h}(t)) = \ell(n)$ for any term t and name n. Finally, we assume that the size of a pairing is at least the size of its arguments. Our second main contribution is a decision procedure for trace equivalence w.r.t. length.

Theorem 2. *Let ℓ be a linear length function. The problem of trace equivalence w.r.t. ℓ is decidable.*

Even if two processes are in trace equivalence for some length function, they may not be in trace equivalence for another one. Choosing the appropriate length function may be tricky since the "right" parameters depend on the implementation of the primitives. We can actually decide a stronger problem: the existence of a length function that would compromise trace equivalence.

Theorem 3. *The following problem is decidable:*

Entry: *two closed processes A and B*
Question: *does there exist a linear length function ℓ such that $A \not\approx^\ell B$?*

For both theorems, the decision procedure builds upon the decision procedure developed in [15] for trace equivalence (without length). Given two closed processes A and B, our procedure roughly works as follows.
1. We first apply the procedure of [15] to A and B.
2. If $A \not\approx B$ (A and B are not in trace equivalence) then clearly $A \not\approx^{\ell} B$ for any length function ℓ.
3. Otherwise, if $A \approx B$, we look deeper at the output of the procedure of [15]. It ends up with two trees (one for each process), which leaves are sets of "constraint systems" $\mathcal{C}$ that define a parametrized frame $\Phi(\mathcal{C})$. We can associate polynomials to each frame as follows. Given a term u with parameters $param(u)$, we define its *associated polynomial* $P_u \in \mathbb{Z}[param(u)]$ by $P_n = \ell(n)$ for n a name, $P_x = x$ for x a parameter and $P_{f(u_1,\dots,u_k)} = \ell_f(P_{u_1}, \dots, P_{u_k})$ otherwise.

Then the sequence of polynomials associated to a frame $\Phi = \{\xi_1 \rhd u_1, \dots, \xi_n \rhd u_n\}$ is $P_\Phi = P_{u_1}, \dots, P_{u_n}$.

We can show that $A \approx^{\ell} B$ if and only if, for any set Σ_1 of constraint system that appears as leaf in the tree associated to A, its corresponding set Σ_2 of constraint system in the tree associated to B is such that

$$\{P_{\Phi(\mathcal{C})} \mid \mathcal{C} \in \Sigma_1\} = \{P_{\Phi(\mathcal{C})} \mid \mathcal{C} \in \Sigma_2\}.$$

Therefore, checking for trace equivalence for a particular linear length function ℓ (Theorem 2) amounts into checking for equality of sets of polynomials. Checking whether there exists a linear length function ℓ such that an attacker can distinguish between A and B (Theorem 3) amounts into checking for equality of sets of parametrized polynomials, which in turn amounts again into checking for equality of polynomials (since the coefficients of the parametrized polynomials are also polynomials).

Our procedure could be easily extended to non linear length functions, provided that we can solve the corresponding algebraic problem, that is equality of the zeros of the P_u's, when they are not polynomials anymore.

5 Passport

The biometric passport contains an RFID chip that stores sensitive authentication information such as birth date, nationality, picture, fingerprints, and also iris characteristics. The International Civil Aviation Organisation (ICAO) standard specifies the communication protocols that are used to access these information [1]. We have discovered a new attack on anonymity, as soon as the size of the pictures may vary from one user to another one.

5.1 Description of the Passive Authentication protocol

According to the ICAO standard, a reader (*e.g.* officer at the border) and a passport first establishes key sessions (denoted *ksenc* and *ksmac*) through the Basic Access Control protocol. Once such keys are successfully established, the

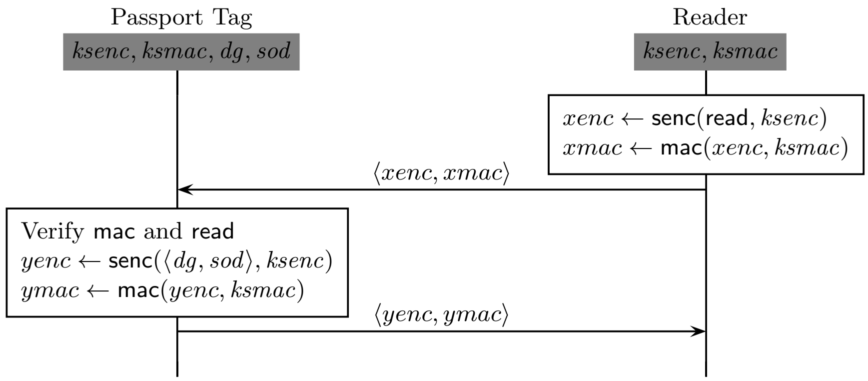

Fig. 1. Passive Authentication protocol (PA)

Passive Authentication protocol (Figure 1) is executed along with other protocols. It establishes a secure communication between the reader and the passport, which sends the (sensitive) authentication information such as the name, date of birth, nationality, and pictures. This information is organised in data groups (dg_1 to dg_{19}). In particular, dg_5 contains the JPEG picture of the passport's holder. The standard specifies that JPEG pictures are of size 0 to 99999 bytes.

The Passive Authentication protocol works as follows. (1) The reader sends an authentication query, sending a pre-defined public value read, encrypted by the session key $ksenc$ and MACed by the session MAC key $ksmac$. This ensures that the request comes from a legitimate reader. (2) The passport sends back the authentication information dg (from dg_1 to dg_{19}) together with a certificate $sod \stackrel{\mathsf{def}}{=} \mathsf{sign}(dg, sk_{DS})$, encrypted under the encryption key $ksenc$ and MACed under $ksmac$. The certificate sod ensures the validity of the information.

5.2 Formal specification of the protocol

The formal specification of the Passive Authentication protocol is displayed in Figure 2. The process $PA(dg, \ell)$ represents a session of the passive authentication protocol, where *Pass* and *Reader* represent respectively the Passport Tag and the Reader. The key $ksenc$ and $ksmac$ are fresh names shared only by *Pass* and *Reader* since they are session keys previously established by the Basic Access Control protocol.

5.3 Unlinkability

The ICAO standard specifies that biometric passport must ensure *unlinkability*, *i.e.* must ensure that a user may make multiple uses of a service or a resource

$$
\begin{aligned}
\mathit{Pass}(dg, \mathit{ksenc}, \mathit{ksmac}) \stackrel{\text{def}}{=}\ & \mathsf{in}(c, x). \\
& \mathsf{if}\ \mathsf{mac}(\mathsf{proj}_1(x), \mathit{ksmac}) = \mathsf{proj}_2(x)\ \mathsf{then} \\
& \quad \mathsf{if}\ \mathsf{proj}_1(x) = \mathsf{senc}(\mathsf{read}, \mathit{ksenc})\ \mathsf{then} \\
& \quad\quad \mathsf{let}\ y = \mathsf{senc}(\langle dg, \mathsf{sign}(dg, sk_{DS})\rangle, \mathit{ksenc})\ \mathsf{in} \\
& \quad\quad \mathsf{out}(c, \langle y, \mathsf{mac}(y, \mathit{ksmac})\rangle) \\
& \quad \mathsf{else}\ \mathsf{out}(c, \mathit{Error}) \\
& \mathsf{else}\ \mathsf{out}(c, \mathit{Error}) \\
\mathit{Reader}(\mathit{ksenc}, \mathit{ksmac}) \stackrel{\text{def}}{=}\ & \mathsf{let}\ \mathit{xenc} = \mathsf{senc}(\mathsf{read}, \mathit{ksenc})\ \mathsf{in} \\
& \mathsf{out}(c, \langle \mathit{xenc}, \mathsf{mac}(\mathit{xenc}, \mathit{ksmac})\rangle).\mathsf{in}(c, x). \\
& \mathsf{if}\ \mathsf{mac}(\mathsf{proj}_1(x), \mathit{ksmac}) = \mathsf{proj}_2(x)\ \mathsf{then} \\
& \quad \mathsf{let}\ y = \mathsf{sdec}(\mathsf{proj}_1(x), \mathit{ksenc})\ \mathsf{in} \\
& \quad \mathsf{if}\ \mathsf{check}(\mathsf{proj}_2(y), \mathsf{vk}(sk_{DS})) = \mathsf{proj}_1(y)\ \mathsf{then}\ 0
\end{aligned}
$$

$$PA(dg) \stackrel{\text{def}}{=} \nu \mathit{ksenc}.\nu \mathit{ksmac}.(\mathit{Pass}(dg, \mathit{ksenc}, \mathit{ksmac}) \mid \mathit{Reader}(\mathit{ksenc}, \mathit{ksmac}))$$

Fig. 2. Formal specification of the Passive Authentication Protocol.

without others being able to link these uses together. The unlinkability of the Passive Authentication protocol can be formalised by the following equivalence:

$$\nu sk_{DS}.(PA(dg_1) \mid PA(dg_1) \approx^{\ell} \nu sk_{DS}.(PA(dg_1) \mid PA(dg_2))$$

where dg_1, dg_2 are the respective data groups of two passport. Intuitively, a user is unlinkable if an attacker cannot distinguish two sessions where the same user is present from two sessions where two different users are present.

Attack. Intuitively, the attack works as follows. We assume that the attacker first listens to an honest session between a reader and a passport A under attack. It therefore learns the size of the encryption of the data groups. Now, listening to any session between a reader and a passport B, it can compare the size of the encryption of the data groups. with the previous one. If they differ, A cannot be present, that is $B \neq A$. If they are equal, then B is likely to be A. How likely depends on the variability of the length and the size of the group of passport holders the attacker wish to distinguish from. Formally, this attack shows that $\nu sk_{DS}.(PA(dg_1) \mid PA(dg_1) \not\approx^{\ell} \nu sk_{DS}.(PA(dg_1) \mid PA(dg_2))$.

Impact. Our attack is very simple: a small device placed near a reader may very quickly decides whether A is present or not, simply listening to the messages received by the reader. [5] also describes an attack against unlinkability. It is based on the Basic Access Control protocol and relies on the fact that different error codes were used in the implementation of the French passports. The attack is dedicated to French passports and has now been fixed. Another attack demonstrated by A. Laurie consists in brute-forcing the document numbers of the passport (which normally requires to open and read the first page of the passport). Once the document numbers are known, anyone can access the data groups. In contrast, our attack does not require any access to these numbers and is inherent to the variability of the size of identifying objects such as pictures.

Fixes. The only simple fix is to ensure that data groups are of fixed size, typically by padding and/or restricting the range of size of data groups. However, this would result in heavier exchanges. Alternatively, a solution is to add padding of random size (which size varies at each transaction). The attacker would still gain some information on the probable user's identity but with smaller probability.

5.4 Implementation of the decision procedure

We have implemented our decision procedure in the active case (for the standard primitives) as an extension of the APTE tool [14]. Thanks to our tool, we can prove our fix (with padding) secure. Consider two data groups dg'_1, dg'_2 of the same length ($\ell(dg'_1) = \ell(dg'_2)$). Using APTE, we show that padding ensures unlinkability, that is, $\nu sk_{DS}.(PA(dg'_1) \mid PA(dg'_1) \approx^\ell \nu sk_{DS}.(PA(dg'_1) \mid PA(dg'_2))$. We can also show that our attack relies solely on the ability to compare lengths. Indeed, using APTE again, we can show that PA guarantees unlinkability for trace equivalence without length, that is $\nu sk_{DS}.(PA(dg_1) \mid PA(dg_1) \approx \nu sk_{DS}.(PA(dg_1) \mid PA(dg_2))$.

The following table summarises our findings using APTE on a 2.4 Ghz Intel Core 2 Duo. The input file used can be found in [14].

	Unlinkability	Time
PA w.r.t. $\approx$	true	4.42 sec
PA w.r.t. $\approx^\ell$	false	0.01 sec
PA with padding w.r.t. $\approx$	true	4.44 sec
PA with padding w.r.t. $\approx^\ell$	true	4.36 sec

6 Conclusion

We have proposed the first decision procedure for behavioral equivalence in presence of a length function. This allows e.g. to check for privacy properties more accurately. In the passive case, we have shown how to extend existing decidability results to a length function, for large classes of equational theories. In the active case, we provide a decision procedure for the standard primitives. Its implementation is an extension of the APTE tool [14]. As an application, we have discovered a new privacy flaw in the biometric passport. As future work, we plan to implement our attack and test it on several passports.

In this paper, we have focused on linear length functions since linear length functions can be realized for standard primitives and proved sound w.r.t. a cryptographic model [16]. We plan to investigate other families of length functions that are relevant for cryptographic primitives. In case some of these functions are not linear, we may need to revisit our procedure.

Protocols may sometimes perform length tests as well, for example, an agent may check that some data does not exceed a certain length. We believe that our procedure can be adapted in case length tests appear in the control flow of the protocols. It would require to extend the constraint systems used in the procedure in order to store constraints on the length. Adapting the decision procedure to solve these additional constraints might be challenging and raise difficult termination problems.

Our length function may also be used to capture other kind of leakages such as computation time or power consumption. To detect such side-channel attacks, we would need to model the "length" (or computation time / power consumption) of tests performed in the protocol. We plan to study whether our procedure can be extended to the case where protocols not only leak the length of output terms but also the "length" of performed tests.

References

1. Machine readable travel document. Technical Report 9303, International Civil Aviation Organization, 2008.
2. M. Abadi, B. Blanchet, and C. Fournet. Just fast keying in the pi calculus. *ACM Transactions on Information and System Security (TISSEC)*, 10(3):1–59, 2007.
3. M. Abadi and V. Cortier. Deciding knowledge in security protocols under equational theories. *Theoretical Computer Science*, 387(1-2):2–32, 2006.
4. M. Abadi and C. Fournet. Mobile values, new names, and secure communication. In *28th ACM Symp. on Principles of Programming Languages (POPL'01)*, 2001.
5. M. Arapinis, T. Chothia, E. Ritter, and M. Ryan. Analysing unlinkability and anonymity using the applied pi calculus. In *23rd IEEE Computer Security Foundations Symposium (CSF'10)*, 2010.
6. A. Armando et al. The AVISPA Tool for the automated validation of internet security protocols and applications. In *17th Int. Conference on Computer Aided Verification (CAV'05)*, volume 3576 of *LNCS*, pages 281–285. Springer, 2005.
7. C. Bansal, K. Bhargavan, and S. Maffeis. Discovering concrete attacks on website authorization by formal analysis. In *25th IEEE Computer Security Foundations Symposium (CSF 2012)*, 2012.
8. M. Baudet. Deciding security of protocols against off-line guessing attacks. In *12th Conference on Computer and Communications Security (CCS'05)*, 2005.
9. M. Berrima, N. Ben Rajeb, and V. Cortier. Deciding knowledge in security protocols under some e-voting theories. *Theoretical Informatics and Applications (RAIRO-ITA)*, 45:269–299, 2011.
10. B. Blanchet. An Efficient Cryptographic Protocol Verifier Based on Prolog Rules. In *14th Computer Security Foundations Workshop (CSFW'01)*, 2001.
11. B. Blanchet, M. Abadi, and C. Fournet. Automated verification of selected equivalences for security protocols. *Journal of Logic and Algebraic Programming*, 75(1):3–51, 2008.
12. R. Chadha, Ş. Ciobâcă, and S. Kremer. Automated verification of equivalence properties of cryptographic protocols. In *21th European Symposium on Programming (ESOP'12)*, 2012.
13. V. Cheval. *Automatic verification of cryptographic protocols: privacy-type properties*. Phd thesis, ENS Cachan, France, 2012.
14. V. Cheval. APTE (Algorithm for Proving Trace Equivalence), 2013. `http://projects.lsv.ens-cachan.fr/APTE/`.
15. V. Cheval, H. Comon-Lundh, and S. Delaune. Trace equivalence decision: Negative tests and non-determinism. In *18th ACM Conference on Computer and Communications Security (CCS'11)*, 2011.
16. H. Comon-Lundh and V. Cortier. Computational soundness of observational equivalence. In *15th Conf. on Computer and Communications Security (CCS'08)*, 2008.
17. V. Cortier and S. Delaune. A method for proving observational equivalence. In *22nd IEEE Computer Security Foundations Symposium (CSF'09)*, 2009.

$$(\mathcal{E}; \{\mathsf{if}\ u = v\ \mathsf{then}\ Q_1\ \mathsf{else}\ Q_2\} \uplus \mathcal{P}; \Phi) \xrightarrow{\tau}_i (\mathcal{E}; \{Q_1\} \uplus \mathcal{P}; \Phi) \quad (\textsc{Then}_c)$$
$$\text{if } u{\downarrow} = v{\downarrow}, \mathsf{Message}(u) \text{ and } \mathsf{Message}(v)$$

$$(\mathcal{E}; \{\mathsf{if}\ u = v\ \mathsf{then}\ Q_1\ \mathsf{else}\ Q_2\} \uplus \mathcal{P}; \Phi) \xrightarrow{\tau}_i (\mathcal{E}; \{Q_2\} \uplus \mathcal{P}; \Phi) \quad (\textsc{Else}_c)$$
$$\text{if } u{\downarrow} \neq v{\downarrow} \text{ or } \neg\mathsf{Message}(u) \text{ or } \neg\mathsf{Message}(v)$$

$$(\mathcal{E}; \{\mathsf{out}(u,t).Q_1; \mathsf{in}(v,x).Q_2\} \uplus \mathcal{P}; \Phi) \xrightarrow{\tau}_i (\mathcal{E}; \{Q_1; Q_2\{x \mapsto t\}\} \uplus \mathcal{P}; \Phi) \quad (\textsc{Comm}_c)$$
$$\text{if } \mathsf{Message}(u), \mathsf{Message}(v), \mathsf{Message}(t) \text{ and } u{\downarrow} = v{\downarrow}$$

$$(\mathcal{E}; \{\mathsf{in}(u,x).Q\} \uplus \mathcal{P}; \Phi) \xrightarrow{in(N,M)}_i (\mathcal{E}; \{Q\{x \mapsto t\}\} \uplus \mathcal{P}; \Phi) \quad (\textsc{In}_c)$$
$$\text{if } M\Phi = t,\ \mathit{fvars}(M,N) \subseteq \mathrm{dom}(\Phi),\ \mathit{fnames}(M,N) \cap \mathcal{E} = \emptyset$$
$$N\Phi{\downarrow} = u{\downarrow},\ \mathsf{Message}(M\Phi),\ \mathsf{Message}(N\Phi), \text{ and } \mathsf{Message}(u)$$

$$(\mathcal{E}; \{\mathsf{out}(u,t).Q\} \uplus \mathcal{P}; \Phi) \xrightarrow{\nu ax_n.out(M,ax_n)}_i (\mathcal{E}; \{Q\} \uplus \mathcal{P}; \Phi \cup \{ax_n \rhd t\}) \quad (\textsc{Out}_c)$$
$$\text{if } M\Phi{\downarrow} = u{\downarrow},\ \mathsf{Message}(u),\ \mathit{fvars}(M) \subseteq \mathrm{dom}(\Phi),\ \mathit{fnames}(M) \cap \mathcal{E} = \emptyset$$
$$\mathsf{Message}(M\Phi),\ \mathsf{Message}(t) \text{ and } ax_n \in \mathcal{AX},\ n = |\Phi| + 1$$

$$(\mathcal{E}; \{P_1 \mid P_2\} \uplus \mathcal{P}; \Phi) \xrightarrow{\tau}_i (\mathcal{E}; \{P_1; P_2\} \uplus \mathcal{P}; \Phi) \quad (\textsc{Par}_c)$$

$$(\mathcal{E}; \{P_1 + P_2\} \uplus \mathcal{P}; \Phi) \xrightarrow{\tau}_i (\mathcal{E}; \{P_1\} \uplus \mathcal{P}; \Phi) \quad (\textsc{Choice}_c\text{-}1)$$

$$(\mathcal{E}; \{P_1 + P_2\} \uplus \mathcal{P}; \Phi) \xrightarrow{\tau}_i (\mathcal{E}; \{P_2\} \uplus \mathcal{P}; \Phi) \quad (\textsc{Choice}_c\text{-}2)$$

where u, v, t are ground terms, and x is a variable.

Fig. 3. Semantics

A Semantics

The semantics for processes is given in Figure 3. Since we consider arbitrary terms for channels, we need to check whether the channel is known by the attacker or not (see rules In and Out). Moreover, we check that all terms that have to be evaluated during the execution are messages.

Let $\mathcal{A}_c$ be the alphabet of actions α used in transitions $\xrightarrow{\alpha}_i$. For every $w \in \mathcal{A}_c^*$ the relation $\xrightarrow{w}_c$ is defined as usual: $P \xrightarrow{\alpha.w'}_c Q$ if there exists P' such that $P \xrightarrow{\alpha}_i P'$ and $P' \xrightarrow{w'}_c Q$. For $s \in (\mathcal{A}_c \smallsetminus \{\tau\})^*$, the relation $\stackrel{s}{\Rightarrow}_c$ is defined by: $A \stackrel{s}{\Rightarrow}_c B$ if, and only if there exists $w \in \mathcal{A}_c^*$ such that $A \xrightarrow{w}_c B$ and s is obtained by erasing all occurrences of τ.

B Decidability of static equivalence for length

B.1 Locally stable

Definition 11 (Size). *The size c_E of an equationnal theory E defined by the equations $\bigcup_{i=1}^{k}\{M_i =_E N_i\}$ is given by $c_E = \max_{1 \leq i \leq k}(|M_i|, |N_i|, ar(\mathcal{F}) + 1)$.*

Definition 12 (Local stability). *An AC-convergent equationnal theory is locally stable if and only if for every closed frame $\phi = \nu\tilde{n}\{M_1/x_1, \ldots, M_k/x_k\}$ in normal form, there exists a finite (computable) set $\mathsf{SAT}(\phi)$, closed modulo AC, such that*

1. $\forall i, M_i \in \mathsf{SAT}(\phi)$ *and* $\forall n \in fnames(\phi), n \in \mathsf{SAT}(\phi)$
2. *if* $N_1, \ldots, N_k$ *are terms such that* $\forall i, N_i \in \mathsf{SAT}(\phi)$ *and* $f(N_1, \ldots, N_k)$ *is a subterm of* $\mathsf{SAT}(\phi)$*, then* $f(N_1, \ldots, N_k) \in \mathsf{SAT}(\phi)$*,*
3. *if* $C[S_1, \ldots, S_l]{\to}M$ *and the reduction occurs in head, where* C *is a context such that* $|C| \leq c_E$ *and* $fnames(C) \cap \tilde{n} = \emptyset$*, and where* $S_i \in sum_\oplus(\mathsf{SAT}(\phi), \tilde{n})$ *for some AC symbol* $\oplus$*, then there exist a context* C'*, a term* M'*, and* $S'_1, \ldots, S'_k \in sum_\oplus(\mathsf{SAT}(\phi), \tilde{n})$ *such that* $|C'| \leq c_E^2$*,* $fnames(C') \cap \tilde{n} = \emptyset$*, and* $M \to^*_{AC} M' =_{AC} C'[S'_1, \ldots, S'_k]$*,*
4. *if* $M \in \mathsf{SAT}(\phi)$*, then* $\phi \vdash M$*.*

where $sum_\oplus(S, \tilde{n})$ *is the set of linear combinations (with* $\oplus$*) built on terms in* S *and names in* $\mathcal{N}\backslash\tilde{n}$*.*

B.2 Proof of Theorem 1

This section is devoted to the proof of Theorem 1, that is, the soundness and completeness of the algorithm provided in Section 3.2. In the remaining of this section, we assume given a SET-stable equational theory E.

Let us first remark that length equality is stable by application of context.

Lemma 1. *If* C *is a context and* $\tilde{M_i^1}, \tilde{M_i^2}$ *are terms such that* $\forall i, \ell_{\mathsf{aux}}(M_i^1) = \ell_{\mathsf{aux}}(M_i^2)$ *then* $\ell_{\mathsf{aux}}(C[\tilde{M_i^1}]) = \ell_{\mathsf{aux}}(C[\tilde{M_i^2}])$.

Proof. We prove recursively on C that $\forall i, \ell_{\mathsf{aux}}(M_i^1) = \ell_{\mathsf{aux}}(M_i^2)$ implies that $\ell_{\mathsf{aux}}(C[\tilde{M_i^1}]) = \ell_{\mathsf{aux}}(C[\tilde{M_i^2}])$.

1. if $|C| = 1$, then either C is a hole and $\exists k$ such that $C[\tilde{M_i}] = M_k$ hence

$$\ell_{\mathsf{aux}}(C[\tilde{M_i^1}]) = \ell_{\mathsf{aux}}(M_k^1) = \ell_{\mathsf{aux}}(M_k^2) = \ell_{\mathsf{aux}}(C[\tilde{M_i^2}])$$

   or C is a constant hence $C[\tilde{M_i^1}] = C[\tilde{M_i^2}]$.
2. if $|C| > 1$, then there exists f such that $C[\tilde{M_i}] = f(C_1[\tilde{M_i}], \ldots, C_k[\tilde{M_i}])$. So $\ell_{\mathsf{aux}}(C_j[\tilde{M_i^1}]) = \ell_{\mathsf{aux}}(C_j[\tilde{M_i^2}])$ holds hence, using item 1 of Definition 5:

$$\begin{aligned}\ell_{\mathsf{aux}}(C[\tilde{M_i^1}]) &= \ell_{\mathsf{aux}}(f(C_1[\tilde{M_i^1}], \ldots, C_k[\tilde{M_i^1}])) \\ &= l_f(\ell_{\mathsf{aux}}(C_1[\tilde{M_i^1}]), \ldots, \ell_{\mathsf{aux}}(C_k[\tilde{M_i^1}])) \\ \ell_{\mathsf{aux}}(C[\tilde{M_i^2}]) &= \ell_{\mathsf{aux}}(f(C_1[\tilde{M_i^2}], \ldots, C_k[\tilde{M_i^2}])) \\ &= l_f(\ell_{\mathsf{aux}}(C_1[\tilde{M_i^2}]), \ldots, \ell_{\mathsf{aux}}(C_k[\tilde{M_i^2}]))\end{aligned}$$

   so $\ell_{\mathsf{aux}}(C[\tilde{M_i^1}]) = \ell_{\mathsf{aux}}(C[\tilde{M_i^2}])$, which concludes the proof.

We now introduce some additional notations.

Definition 13. *Given a frame $\phi_1 = \nu\tilde{n}\sigma_1$, we define the small set of length tests to be*

$$\mathsf{Eq}_{\mathsf{small}}(\phi_1) = \{(\zeta_M, M) \mid M \in SET(\phi_1)\}$$

where ζ_M is a recipe for M (ie fnames$(\zeta_M) \cap \tilde{n} = \emptyset$ and $\zeta_M\sigma_1 =_E M$).

We define $\phi_2 \models \mathsf{Eq}_{\mathsf{small}}(\phi_1)$ to hold if for all $(\zeta_M, M) \in \mathsf{SET}(\phi_1)$, $\ell(\zeta_M\phi_2) = \ell(M)$ (with a possible renaming in ϕ_2).

The algorithm presented in Section 3.2 amounts in checking $\phi_2 \models \mathsf{Eq}_{\mathsf{small}}(\phi_1)$ and $\phi_1 \models \mathsf{Eq}_{\mathsf{small}}(\phi_2)$. We have already seen that $\phi_1 \sim_E^\ell \phi_2$ implies $\phi_2 \models \mathsf{Eq}_{\mathsf{small}}(\phi_1)$ and $\phi_1 \models \mathsf{Eq}_{\mathsf{small}}(\phi_2)$. Let us show that, conversely, $\phi_1 \sim_E \phi_2$, $\phi_2 \models \mathsf{Eq}_{\mathsf{small}}(\phi_1)$, and $\phi_1 \models \mathsf{Eq}_{\mathsf{small}}(\phi_2)$ implies $\phi_1 \sim_E^\ell \phi_2$. It is a direct consequence of the following lemma.

Lemma 2. *Let $\phi_1 = \nu\tilde{n}\sigma_1$ and $\phi_2 = \nu\tilde{n}\sigma_2$ be two frames such that $\phi_1 \sim \phi_2$. If $\phi_1 \models \mathsf{Eq}_{\mathsf{small}}(\phi_2)$ then for any term M such that fnames$(M) \cap \tilde{n_1} = \emptyset$. we have $\ell(M\sigma_1) \leq \ell(M\sigma_2)$.*

Proof. Let $\phi_1 = \nu\tilde{n}\sigma_1$ and $\phi_2 = \nu\tilde{n}\sigma_2$ be two frames such that $\phi_1 \sim \phi_2$. Assume $\phi_1 \models \mathsf{Eq}_{\mathsf{small}}(\phi_2)$ and let M be a term such that *fnames*$(M) \cap \tilde{n_1} = \emptyset$.

Let $U \in M\sigma_2{\downarrow}$. Since $M\sigma_2$ is a context over terms in $\mathsf{SET}(\phi_2)$ and applying Definition 7, there exist a context C and terms $N_1, \ldots, N_k \in \mathsf{SET}(\phi_2)$ such that *fnames*$(C) \cap \tilde{n} = \emptyset$ and

$$U =_{AC} C[\tilde{N_i}]$$

Thus $\ell(M\sigma_2) = \ell_{\mathsf{aux}}(U) = \ell_{\mathsf{aux}}(C[\tilde{N_i}])$. Moreover, $\phi_1 \sim \phi_2$ thus $(M =_E C\tilde{\zeta_{N_i}})\phi_2$ implies $(M =_E C\tilde{\zeta_{N_i}})\phi_1$. Let $V_i \in (\zeta_{N_i}\sigma_1){\downarrow}$. We have

$$\ell(M\sigma_1) = \ell(C\tilde{\zeta_{N_i}})\sigma_1) = \ell(C[\tilde{V_i}])$$

Since $\phi_1 \models \mathsf{Eq}_{\mathsf{small}}(\phi_2)$, we have $\ell(V_i) = \ell(\zeta_{N_i}\sigma_1) = \ell(\zeta_{N_i}\sigma_2) = \ell(N_i)$ thus $\ell_{\mathsf{aux}}(V_i) = \ell_{\mathsf{aux}}(N_i)$. Applying Lemma 1, we deduce $\ell_{\mathsf{aux}}(C[\tilde{V_i}]) = \ell_{\mathsf{aux}}(C[\tilde{N_i}])$. The item 3 of Definition 5 implies that $\ell(C[\tilde{V_i}]) \leq \ell_{\mathsf{aux}}(C[\tilde{V_i}])$. Thus we have

$$\ell(M\sigma_1) = \ell(C[\tilde{V_i}]) \leq \ell_{\mathsf{aux}}(C[\tilde{V_i}]) = \ell_{\mathsf{aux}}(C[\tilde{N_i}]) = \ell(M\sigma_2)$$

which concludes the proof.

C Decidability of trace equivalence for length

C.1 Symbolic framework

As mentioned in the paper, our procedure is based for on the procedure of [15] that decide the symbolic equivalence between sets of constraint system without considering a length function. One particularity of the constraint system, obtained by applying the procedure [15], is the generation of an extended frame defined below.

We call *recipe*, usually denoted ξ, the terms in $\mathcal{T}(\mathcal{F}, \mathcal{N} \cup \mathcal{X}^2 \cup \mathcal{AX})$. We say that a recipe ξ is *closed* (or *ground*) if $\xi \in \mathcal{T}(\mathcal{F}, \mathcal{N} \cup \mathcal{AX})$. Given a recipe $\xi \in \mathcal{T}(\mathcal{F}, \mathcal{X}^2 \cup \mathcal{AX})$, we denote $param(\xi)$ the set of parameter in ξ, *i.e.* $vars(\xi) \cap \mathcal{AX}$. To avoid confusion, we denote the set of first order variables $\mathcal{X}$ by $\mathcal{X}^1$.

Definition 14. *We define an extended frame as a sequence* $\{\xi_1 \rhd u_1, \dots, \xi_n \rhd u_n\}$ *where* $\xi_j \in \mathcal{T}(\mathcal{F}, \mathcal{X}^2 \cup \mathcal{AX})$ *and* $u_j \in \mathcal{T}(\mathcal{F}'_c, \mathcal{X}^1 \cup \mathcal{N})$ *for all* $j \in \{1, \dots, n\}$.

Intuitively, the extended frame in a simplified constraint system contains all possible deducible messages obtained by applying destructors like $\mathsf{sdec}, \mathsf{adec}, \dots$.

Example 8. Consider the frame $\Phi = \{a/ax_1, \mathsf{senc}(b, a)/ax_2\}$ of some constraint system. From the procedure [15], Φ would be extended into $\{ax_1 \rhd a, ax_2 \rhd \mathsf{senc}(b, a), \mathsf{sdec}(ax_2, ax_1) \rhd b\}$. Note that $\mathsf{sdec}(ax_2, ax_1)$ represent the recipe need for the intruder to deduce b from Φ.

We use these extended frame in our procedure to decide the symbolic equivalence w.r.t. length. Given a term u (with variables), we define its *associated polynomial* $P_u \in \mathbb{Z}[vars(u)]$ as follows:

$$\begin{aligned} P_n &= \ell(n) && \text{if } n \text{ is a name} \\ P_x &= \ell(x) && \text{if } x \text{ is a variable} \\ P_{f(u_1,\dots,u_k)} &= \ell_f(P_{u_1}, \dots, P_{u_k}) \end{aligned}$$

We extend this definition to frames: given a frame $\Phi = \{\xi_1 \rhd u_1, \dots, \xi_n \rhd u_n\}$, its associated polynomial is $P_\Phi = P_{u_1}, \dots, P_{u_n}$.

Proposition 1. *Assume* $u\sigma$ *to be a term in normal form and let* $x_1, \dots, x_n$ *be the variables of* u. *Let* P_u *is the associated polynomial of* u. *Then*

$$\ell(u\sigma) = P_u(\ell(x_1\sigma), \dots, \ell(x_n\sigma))$$

This follows immediately from the definition of P_u and the length function.

Definition 15. *A* constraint system *is a triple* $(\Phi; D; Eq)$*:*

- Φ *is an extended frame* $\{\xi_1 \rhd u_1, \dots, \xi_n \rhd u_n\}$ *where* u_i *are constructor terms and* ξ_i *are recipes.*
- D *is a set of deducible constraints of the form* $X, i \overset{?}{\vdash} x$ *with* $i \in \mathbb{N}$, $X \in \mathcal{X}^2$, $x \in \mathcal{X}^1$.
- Eq *is a set of inequations of the form* $t \overset{?}{\neq} t'$ *where* t, t' *are constructors terms that do not contain names.*

Given a set D of constraints, we denote by $vars^1(D)$ (resp. $vars^2(D)$) the first order (resp. second order) variables of D, that is $vars^1(D) = fvars(D) \cap \mathcal{X}^1$ (resp. $vars^2(D) = fvars(D) \cap \mathcal{X}^2$). We also assume the following conditions are satisfied on a constraint system:

1. for every $x \in vars^1(D)$, there exists a unique X such that $(X, i \stackrel{?}{\vdash} x) \in D$, and each variable X occurs at most once in D.
2. $vars^1(\mathcal{C}) \subseteq vars^1(D)$
3. for every $1 \leq k \leq n$, for every $x \in vars^1(t_k)$, if $ax_j \in vars^2 \xi_k$ then there exists $(X, i \stackrel{?}{\vdash} x) \in D$ such that $i < j$.

Given an extended frame $\Phi = \{\xi_1, i_1 \rhd t_1, \ldots, \xi_n, i_n \rhd t_n\}$ and a recipe ξ, we say that ξ is build from Φ if there exists a context $C[\_1, \ldots, \_m]$ containing only constructor function symbol and $j_1, \ldots, j_m \in \{1, \ldots, n\}$ such that $\xi = C[\xi_{j_1}, \ldots, \xi_{j_m}]$. Moreover, we denote $\xi\Phi$ the term $C[t_{j_1}, \ldots, t_{j_m}]$.

Definition 16 (solution). *A* solution *of a constraint system $\mathcal{C} = (\Phi; D; Eq)$ is a pair of substitutions (σ, θ) such that σ is a mapping from $vars^1(\mathcal{C})$ to $\mathcal{T}(\mathcal{F}'_c, \mathcal{N})$, θ is a mapping from $vars^2(\mathcal{C})$ to $\mathcal{T}(\mathcal{F}, \mathcal{AX})$, and:*

1. *for all $(X, k \stackrel{?}{\vdash} x) \in D$, $X\theta$ is built from $\Phi\theta$, $(X\theta)(\Phi\theta\sigma) = x\sigma$, and $param(X\theta) \subseteq \{ax_1, \ldots, ax_k\}$;*
2. *for all $t \stackrel{?}{\neq} t'$, $t\sigma \neq t'\sigma$*

The substitution σ is called the first-order solution *of $\mathcal{C}$ associated to θ, called* second-order solution *of $\mathcal{C}$. The set of solutions of a constraint system $\mathcal{C}$ is denoted $\mathsf{Sol}(\mathcal{C})$.*

Definition 17 (static equivalence). *Let Φ and Φ' two closed extended frames having the same structure. We say that Φ and Φ' are in static equivalence if for all ξ, ξ' built from Φ (thus also built from Φ'), $\xi\Phi = \xi'\Phi$ if and only if $\xi\Phi' = \xi'\Phi'$. Moreover, we say that Φ and Φ' are in static equivalent w.r.t. length if Φ and Φ' are statically equivalent and for all ξ, ξ' built from Φ, $\ell(\xi\Phi) = \ell(\xi'\Phi)$ if and only if $\ell(\xi\Phi') = \ell(\xi'\Phi')$.*

Definition 18 (symbolic equivalence). *Let Σ and Σ' be two sets of onstraint systems that contain constraint systems having the same structure. We say that Σ and Σ' are in symbolic equivalence, denoted by $\Sigma \approx^c_s \Sigma'$, if for all $\mathcal{C} \in \Sigma$, for all $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$, there exists $\mathcal{C}' \in \Sigma'$ and a substitution σ' such that $(\sigma', \theta) \in \mathsf{Sol}(\mathcal{C}')$ and $\Phi\theta\sigma \sim_c \Phi'\theta\sigma'$ (and conversely) where $\mathcal{C} = (\Phi; D; Eq)$ and $\mathcal{C}' = (\Phi'; D'; Eq')$.*

We say that Σ and Σ' are in symbolic equivalence w.r.t. length, denoted by $\Sigma \approx^c_{s\ell} \Sigma'$, if for all $\mathcal{C} \in \Sigma$, for all $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$, there exists $\mathcal{C}' \in \Sigma'$ and a substitution σ' such that $(\sigma', \theta) \in \mathsf{Sol}(\mathcal{C}')$ and $\Phi\theta\sigma \sim^\ell_c \Phi'\theta\sigma'$ (and conversely) where $\mathcal{C} = (\Phi; D; Eq)$ and $\mathcal{C}' = (\Phi'; D'; Eq')$.

C.2 Reduction of trace equivalence to symbolic equivalence

The algorithm of [15] is explained in detail in [13]. In particular, [13] contains a reduction result of trace equivalence to symbolic equivalence w.r.t. to a predicate that satisfies a stability property under replacement of name by successive application of the function h.

Let $\mathcal{E}$ be a finite set of names denoted $\{b_1, \ldots, b_n\}$. Let N be a positive integer. Let $a \in \mathcal{N}$. We denote $\sigma_{\mathcal{E},N,a}$ the substitution defined such that for all $i \in \mathbb{N}^+$, $b_i\sigma_{\mathcal{E},N,a} = \mathsf{h}^{i\times N}(a)$.

Property 1. Let $\mathcal{E}$ be a finite set of names. For all u, v closed constructor terms, there exists $N \in \mathbb{N}$ such that for all $N' > N$, $\ell(u) = \ell(v)$ if and only if $\ell(u\sigma_{\mathcal{E},N,a}) = \ell(v\sigma_{\mathcal{E},N,a})$.

Intuitively, this property is used in [13] to replace every public name introduced by the intruder by successive applications of h that preserves the length of messages.

Theorem 4 (derived from [13, Theorem 8.4]). *Consider a length function that satisfies Property 1. Given an algorithm for deciding the symbolic equivalence w.r.t. length between two sets of constraint systems Σ and Σ' that contain constraint systems having the same structure and such that there exists $a \in \mathcal{N}$ such that for all $(\Phi; D; Eq) \in \Sigma \cup \Sigma'$, $(ax_1 \rhd a) \in \Phi$, and for all $\mathcal{C}, \mathcal{C}' \in \Sigma \cup \Sigma'$, $\mathcal{C} \approx_s^c \mathcal{C}'$, $D = D'$ and $Eq = Eq'$ where $\mathcal{C} = (\Phi; D; Eq)$ and $\mathcal{C}' = (\Phi'; D'; Eq')$, we can derive an algorithm for deciding trace equivalence w.r.t. length between two bounded processes.*

C.3 Specific solutions of a constraint system

In this section, we assume $a \in \mathcal{N}$. Let $\mathsf{H}(t)$ be the height of the term t.

Definition 19. *Let $\xi \in \mathcal{T}(\mathcal{F}'_c, \{ax_1\})$ and $a \in \mathcal{N}$. We define $\mathsf{Rec}^2_n(\xi) \in \mathcal{T}(\mathcal{F}'_c, \{ax_1\})$ inductively as follow:*

- $\mathsf{Rec}^2_0(\xi) = ax_1$
- $\mathsf{Rec}^2_n(\xi) = \langle \mathsf{Rec}^2_{n-1}(\xi), \xi \rangle$

Similarly, let $u \in \mathcal{T}(\mathcal{F}'_c, \{a\}$. We define $\mathsf{Rec}^1_n(u) \in \mathcal{T}(\mathcal{F}'_c, \{a\})$ inductively as follow:

- $\mathsf{Rec}^1_0(u) = a$
- $\mathsf{Rec}^1_n(u) = \langle \mathsf{Rec}^1_{n-1}(u), u \rangle$

Lemma 3. *Let Φ an extended frame such that $(ax_1 \rhd a) \in \Phi$. Let $\xi \in \mathcal{T}(\mathcal{F}'_c, \{ax_1\}$. Let $n \in \mathbb{N}$. Then $\mathsf{Rec}^2_n(\xi)\Phi = \mathsf{Rec}^1_n(\xi\Phi)$.*

Lemma 4. *Let Φ an extended frame such that $(ax_1 \rhd a) \in \Phi$. Let $\xi \in \mathcal{T}(\mathcal{F}'_c, \{ax_1\}$. Let $n \in \mathbb{N}$ such that $n > 0$. Then*

$$\mathsf{H}(\mathsf{Rec}^2_n(\xi)) = \mathsf{H}(\mathsf{Rec}^1_n(\xi\Phi)) = n + \mathsf{H}(\xi)$$

Proof. We prove the result by induction on n.

Base case $n = 1$: In such a case, we have $\mathsf{Rec}^2_1(\xi) = \langle ax_1, \xi \rangle$. Since the $\xi \in \mathcal{T}(\mathcal{F}'_c, \{ax_1\}$, we deduce that $\mathsf{H}(\xi) > 0$ hence the result holds.

Inductive step $n > 1$: Otherwise, $\mathsf{Rec}^2_n(\xi) = \langle \mathsf{Rec}^2_{n-1}(\xi), \xi \rangle$. Thus, we have $\mathsf{H}(\mathsf{Rec}^2_n(\xi)) = 1 + \max(\mathsf{H}(\mathsf{Rec}^2_{n-1}(\xi)); \mathsf{H}(\xi)) = 1 + \max(n - 1 + \mathsf{H}(\xi); \mathsf{H}(\xi))$. Since $n > 0$, we deduce that $\mathsf{H}(\xi) = 1 + n - 1 + \mathsf{H}(\xi) = n + \mathsf{H}(\xi)$. □

Lemma 5. *Let $n, n' \in \mathbb{N}$ with $n, n' > 0$. Let $u, u' \in \mathcal{T}(\mathcal{F}'_c, \{a\})$. We have:*

$$\mathsf{Rec}^1_n(u) = \mathsf{Rec}^1_{n'}(u') \text{ implies } u = u' \text{ and } n = n'$$

Proof. By hypothesis, we have $n, n' > 0$. Hence by definition of $\mathsf{Rec}^1_n(u)$ and $\mathsf{Rec}^1_{n'}(u')$, $\mathsf{Rec}^1_n(u) = \mathsf{Rec}^1_{n'}(u')$ implies that $\mathsf{Rec}^1_{n-1}(u) = \mathsf{Rec}^1_{n'-1}(u')$ and $u = u'$ which implies that $\mathsf{H}(\mathsf{Rec}^1_{n-1}(u)) = \mathsf{H}(\mathsf{Rec}^1_{n'-1}(u))$. Hence by Lemma 4, we deduce $n - 1 + \mathsf{H}(u) = n' - 1 + \mathsf{H}(u)$ and so $n = n'$. $\square$

Lemma 6. *Let $x_0 \in \mathcal{X}^1$ and $u \in \mathcal{T}(\mathcal{F}'_c, \mathcal{X}^1)$ such that $x_0 \notin vars^1(u)$. Assume that $vars^1(u) = \{x_1, \ldots, x_k\}$. At last, let $N = \mathsf{H}(u)$. Let σ such that:*

- *for all $i \in \{0, \ldots, k\}$, $x_i\sigma = \mathsf{Rec}^1_{n_i}(\mathsf{Rec}^1_{m_i}(a))$ for some $n_i, m_i > N$*
- *for all $i, i' \in \{0, \ldots, k\}$, $i \neq i'$ implies $m_i \neq m'_i$.*
- *for all $i, i' \in \{0, \ldots, k\}$, $n_i > m_{i'}$.*

We have that $x_0\sigma \neq u\sigma$.

Proof. To prove this result, we distinguish two cases:

Case 1, $|u| = 1$: It implies that $u \in \mathcal{X}^1$. Assume that $u = x_\ell$. By hypothesis, we know that $x_0 \notin vars^1(u)$ hence $0 \neq \ell$ and so $m_\ell \neq m_0$. Thanks to Lemma 5, $m_\ell \neq m_0$ implies that $\mathsf{Rec}^1_{m_0}(a) \neq \mathsf{Rec}^1_{m_\ell}(a)$ which implies again thanks to Lemma 5 that $\mathsf{Rec}^1_{n_0}(\mathsf{Rec}^1_{m_0}(a)) \neq \mathsf{Rec}^1_{n_\ell}(\mathsf{Rec}^1_{m_\ell}(a))$. Hence we conclude that $x_0\sigma \neq x_\ell\sigma$.

Case 2, $|u| > 1$: Otherwise, we denote $u = \mathsf{g}(u_1, \ldots, u_n)$. If $\mathsf{g} \neq \langle\rangle$ then by definition of $\mathsf{Rec}^1_{n_0}(\mathsf{Rec}^1_{m_0}(a))$ and since $n_0 > 0$, we have that $x_0\sigma \neq u\sigma$.

Let's now assume that $\mathsf{g} = \mathsf{senc}$ and $x_0\sigma = u\sigma$. In such a case, we would have $\mathsf{Rec}^1_{n_0-1}(\mathsf{Rec}^1_{m_0}(a)) = u_1\sigma$ and $\mathsf{Rec}^1_{m_0}(a) = u_2\sigma$. But $\mathsf{H}(\mathsf{Rec}^1_{m_0}(a)) = m_0 + 1$. Since $\mathsf{H}(u) = N < m_0$, we deduce that there exists $\ell \in \{1, \ldots, k\}$ such that $x_\ell \in vars^1(u_2)$. But we know that $\mathsf{H}(x_k\sigma) = n_k + m_k + 1$ which means that $\mathsf{H}(u_2\sigma) > n_k + m_k + 1$. But by hypothesis we also have that $n_k > m_0$ thus $\mathsf{H}(\mathsf{Rec}^1_{m_0}(a)) < \mathsf{H}(u_2\sigma)$ which contradict the fact that $\mathsf{Rec}^1_{m_0}(a) = u_2\sigma$. $\square$

Definition 20. *Let Σ, Σ' be two sets of constraint systems such that there exists $a \in \mathcal{N}$ such that for all $\mathcal{C}, \mathcal{C}' \in \Sigma \cup \Sigma'$, $D(\mathcal{C}) = D(\mathcal{C}')$, $Eq(\mathcal{C}) = Eq(\mathcal{C}')$ and $(ax_1 \rhd a) \in \Phi(\mathcal{C})$. Let N be maximum height of all terms in the inequations of Σ, Σ'. Let $k = |D(\mathcal{C})|$ for some $\mathcal{C} \in \Sigma$. Let $E_1, \ldots, E_k$ the sets such that for all $i \in \{1, \ldots, k\}$*

$$E_i = \{\mathsf{Rec}^1_n(\mathsf{Rec}^1_{N+i}(a)) \mid n \in \mathbb{N} \wedge n > N + k\}$$

At last, we define $S(\Sigma, \Sigma')$ the set of specific solutions of Σ and Σ' such that such that:

- *$E_1 \times \ldots \times E_k = \{(x_1\sigma, \ldots, x_k\sigma) \mid (\sigma, \theta) \in S(\Sigma, \Sigma')\}$*
- *for all $(\sigma, \theta) \in S(\Sigma, \Sigma')$, for all $(X, i \overset{?}{\vdash} x) \in D(\mathcal{C})$, $x\sigma = \mathsf{Rec}^1_n(\mathsf{Rec}^1_m(a))$ and $X\theta = \mathsf{Rec}^2_n(\mathsf{Rec}^2_m(ax_1))$, for some m,n.*

Lemma 7. *Let Σ, Σ' be two sets of constraint systems such that for all $\mathcal{C}, \mathcal{C}' \in \Sigma \cup \Sigma'$, $D(\mathcal{C}) = D(\mathcal{C}')$ and $Eq(\mathcal{C}) = Eq(\mathcal{C}')$. For all $(\sigma, \theta) \in S(\Sigma, \Sigma')$, for all $\mathcal{C} \in \Sigma \cup \Sigma'$, $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$.*

Proof. Let $(\sigma, \theta) \in S(\Sigma, \Sigma')$. Let $\mathcal{C} \in \Sigma$. We show that $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$ by verifying the two conditions of Definition 16. Assume that $D(\mathcal{C}) = \{X_1, i_1 \overset{?}{\vdash} x_1, \ldots, X_n, i_k \overset{?}{\vdash} x_k\}$.

Let $j \in \{1, \ldots, k\}$. By Definition 20, we know that $X_j\theta = \mathsf{Rec}^2_n(\mathsf{Rec}^2_{N+j}(ax_1))$ where N is the maximum height of all terms in the inequation of $\mathcal{C}$. Hence according to Definition 19, $X_j\theta \in \mathcal{T}(\mathcal{F}'_c, \{ax_1\})$ with $(ax_1 \rhd a) \in \Phi(\mathcal{C})$. We deduce that $X\theta$ is built from $\Phi(\mathcal{C})\theta$. Moreover, thanks to Lemma 3, we know that $X_j\theta(\Phi\theta) = \mathsf{Rec}^1_n(\mathsf{Rec}^1_{N+j}(a))$ thus $X_j\theta(\Phi\theta) = x_j\sigma$. At last $X_j\theta \in \mathcal{T}(\mathcal{F}'_c, \{ax_1\})$ implies that $param(X_j\theta) \subseteq \{ax_1, \ldots, ax_{i_j}\}$.

We now show that for all $(s \overset{?}{\neq} s') \in Eq$, $s\sigma \neq s'\sigma$. Assume that $s\sigma = s'\sigma$. Since Eq is satisfiable, there exists $x \in vars^1(s)$ (resp. $vars^1(s')$) and $t \in st(s')$ (resp. $t \in st(s)$) such that $x\sigma = t\sigma$ and $x \neq t$. Moreover, since $s\sigma = s'\sigma$, we deduce that $x \notin st(t)$. But by definition of a constraint system $x \in vars^1(D(\mathcal{C}))$ hence there exists $j \in \{1, \ldots, k\}$ and $n > N+k$ such that $x\sigma = \mathsf{Rec}^1_n(\mathsf{Rec}^1_{N+j}(a))$. Similarly, if we denote $\{x_{\alpha_1}, \ldots, x_{\alpha_p}\} = vars^1(t)$, we know that there exists $n_{\alpha_1}, \ldots, n_{\alpha_p}$ strictly superior to $N + k$ such that $x_{\alpha_q}\sigma = \mathsf{Rec}^1_{n_{\alpha_q}}(\mathsf{Rec}^1_{N+\alpha_q}(a))$ for all $q \in \{1, \ldots, p\}$. Hence by Lemma 6, we deduce that $x\sigma \neq t\sigma$ which is a contradiction with $s\sigma = s'\sigma$. We conclude that $s\sigma \neq s'\sigma$ and so $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$. □

Lemma 8. *Let $\ell_{\langle\,\rangle}$ the length function of $\langle\,\rangle$ such that $\ell_{\langle\,\rangle}(x_1, x_2) = \alpha_0 + \alpha_1 \times x_1 + \alpha_2 \times x_2$. If $\alpha_1 \geq 1$, $\alpha_2 > 0$ and $\alpha_0 \geq 0$ then for all $k > 0$, for all $i \in \{1, \ldots, k\}$, for all $N > 0$, we have $\{\ell(\mathsf{Rec}^1_n(\mathsf{Rec}^1_{N+i}(a))) \mid n > N + k\}$ is an infinite set.*

Proof. By Definition 19, since $\alpha_1 \geq 1$ and $\alpha_2 > 0$ then for all terms u, for all n, $\ell(\mathsf{Rec}^1_n(u)) < \ell(\mathsf{Rec}^1_{n+1}(u))$. Thus the result holds. □

C.4 Deciding symbolic equivalence w.r.t. length

Proposition 2. *Let $\mathcal{C}_1 = (\Phi_1; D_1; Eq_1)$ and $\mathcal{C}_2 = (\Phi_2; D_2; Eq_2)$ be two simplified constraint systems having same structure. Assume $P_{\Phi_1} = P_{\Phi_2}$. Let $\sigma_1, \sigma_2, \theta$ substitution such that $(\sigma_1, \theta) \in \mathsf{Sol}(\mathcal{C}_1)$, $(\sigma_2, \theta) \in \mathsf{Sol}(\mathcal{C}_2)$. Then for any recipe ξ built from $\Phi_1\theta$, $\ell(\xi\Phi_1\theta\sigma_1) = \ell(\xi\Phi_2\theta\sigma_2)$.*

Proof. Let $(\sigma_1, \theta) \in \mathsf{Sol}(\mathcal{C}_1)$, $(\sigma_2, \theta) \in \mathsf{Sol}(\mathcal{C}_2)$ such that $\Phi_1\theta\sigma_1 \sim_c \Phi_2\theta\sigma_2$. Since $\mathcal{C}_1$ and $\mathcal{C}_2$ have same structure, we can assume that $\Phi_1 = \{\xi_1 \rhd u_1, \ldots, \xi_n \rhd u_n\}$ and $\Phi_2 = \{\xi_1 \rhd v_1, \ldots, \xi_n \rhd v_n\}$.

We first prove by induction that

$$\forall k \in \{1, \ldots, n\}, \ell(u_k\sigma_1) = \ell(v_k\sigma_2)$$

Let H_m be "$\forall k \in \{1,\dots,n\}$, if $param(\xi_k) \subseteq \{ax_1,\dots,ax_m\}$ then $\leq \ell(u_k\sigma_1) = \ell(v_k\sigma_2)$".

Case $m = 1$: Let $k \in \{1,\dots,n\}$ such that $param(\xi_k) \subseteq \{ax_1\}$. By definition of a constraint system, we deduce that u_k is a ground term thus P_{u_k} is a constant. Similarly, P_{v_k} is a constant so $P_{u_k} = P_{v_k}$ implies $\ell(u_k\sigma_1) = \ell(v_k\sigma_2)$. Hence H_1 is true.

Case $m + 1$: Assume H_m is true. Let $k \in \{1,\dots,n\}$ such that $param(\xi_k) \subseteq \{ax_1,\dots,ax_{m+1}\}$. By hypothesis, we know that $P_{u_k} = P_{v_k}$ hence $\ell(u_k\sigma_1) = P_{u_k}(\ell(y_1\sigma_1),\dots,\ell(y_p\sigma_1))$ and $\ell(v_k\sigma_2) = P_{u_k}(\ell(y_1\sigma_2),\dots,\ell(y_p\sigma_2))$ where $y_1,\dots,y_p$ are the variables of u_k and v_k.

Let us show that $\ell(y_j\sigma_1) = \ell(y_j\sigma_2)$ for all $j \in \{1,\dots,p\}$ which will allow us to conclude that $\ell(u_k\sigma_1) = \ell(v_k\sigma_2)$ and so to ensure H_{m+1}. Let $j \in \{1,\dots,p\}$. By definition of a simplified constraint system, $param(\xi_k) \subseteq \{ax_1,\dots,ax_{m+1}\}$ implies that there exists $Y_j, i_j \overset{?}{\vdash} y_j \in D(\mathcal{C}_1) = D(\mathcal{C}_2)$ such that $i_j < m + 1$. By definition of a solution of a simplified constraint system, we know that $Y_j\theta$ is build from $\Phi_1\theta$ hence there exists a constructor context C such that $Y_j\theta = C[\xi_{\alpha_1}\theta,\dots,\xi_{\alpha_p}\theta]$ with $\alpha_1,\dots,\alpha_p \in \{1,\dots,n\}$. Moreover, we know that $Y_j\theta(\Phi_1\theta\sigma_1) = y_j\sigma_1$ and $Y_j\theta(\Phi_2\theta\sigma_2) = y_j\sigma_2$. Hence, we deduce that

$$y_j\sigma_1 = C[u_{\alpha_1}\sigma_1,\dots,u_{\alpha_p}\sigma_1] \qquad y_j\sigma_2 = C[v_{\alpha_1}\sigma_2,\dots,v_{\alpha_p}\sigma_2]$$

So there exists a polynomial P_C such that $\ell(y_j\sigma_1) = P_C(\ell(u_{\alpha_1}\sigma_1),\dots,\ell(u_{\alpha_p}\sigma_1))$, $\ell(y_j\sigma_2) = P_C(\ell(v_{\alpha_1}\sigma_1),\dots,\ell(v_{\alpha_p}\sigma_1))$.

But once again by definition of a solution, $param(Y_j\theta) \subseteq \{ax_1,\dots,ax_{i_j}\}$ with $i_j < m+1$. Hence $param(\xi_{\alpha_q}\theta) \subseteq \{ax_1,\dots,ax_{i_j}\}$ for all $q \in \{1,\dots,p\}$. Thus by the inductive hypothesis H_m, we deduce that for all $q \in \{1,\dots,p\}$, $\ell(u_{\alpha_q}\sigma_1) = \ell(v_{\alpha_q}\sigma_1)$. Hence, it implies that $P_C(\ell(u_{\alpha_1}\sigma_1),\dots,\ell(u_{\alpha_p}\sigma_1)) = P_C(\ell(v_{\alpha_1}\sigma_1),\dots,\ell(v_{\alpha_p}\sigma_1))$ and so $\ell(y_j\sigma_1) = \ell(y_j\sigma_2)$.

Let ξ be a recipe built from $\Phi_1\theta$. Let now show that $\ell(\xi\Phi_1\theta\sigma_1) = \ell(\xi\Phi_2\theta\sigma_2)$. Since ξ is built from $\Phi_1\theta$, there exists a context C with only constructor symbols such that $\xi = C[\xi_{\alpha_1}\theta,\dots,\xi_{\alpha_p}\theta]$ with $\alpha_1,\dots,\alpha_p \in \{1,\dots,n\}$. Moreover, we have that $\xi(\Phi_1\theta)\sigma_1 = C[u_{\alpha_1}\sigma_1,\dots,u_{\alpha_p}\sigma_1]$. Since Φ_1 and Φ_2 have both same structure, then ξ is built from $\Phi_2\theta$ and so $\xi(\Phi_2\theta)\sigma_2 = C[v_{\alpha_1}\sigma_2,\dots,v_{\alpha_p}\sigma_2]$.

So there exists a polynomial P_C such that $\ell(\xi\Phi_1\theta\sigma_1) = P_C(\ell(u_{\alpha_1}\sigma_1),\dots,\ell(u_{\alpha_p}\sigma_1))$ and $\ell(\xi\Phi_2\theta\sigma_2) = P_C(\ell(v_{\alpha_1}\sigma_1),\dots,\ell(v_{\alpha_p}\sigma_1))$. Since $\ell(u_i\sigma_1) = \ell(v_i\sigma_2)$ for all $i \leq n$, we have $\ell(\xi\Phi_1\theta\sigma_1) = \ell(\xi\Phi_2\theta\sigma_2)$.

Corollary 2. *Let $\mathcal{C}_1 = (\Phi_1; D_1; Eq_1)$ and $\mathcal{C}_2 = (\Phi_2; D_2; Eq_2)$ be two constraint systems having same structure. Assume $P_{\Phi_1} = P_{\Phi_2}$. Let $\sigma_1, \sigma_2, \theta$ substitution such that $(\sigma_1, \theta) \in \mathsf{Sol}(\mathcal{C}_1)$, $(\sigma_2, \theta) \in \mathsf{Sol}(\mathcal{C}_2)$. Then $\Phi_1\theta\sigma_1 \sim_c \Phi_2\theta\sigma_2$ is equivalent to $\Phi_1\theta\sigma_1 \sim_c^\ell \Phi_2\theta\sigma_2$*

Proof. By definition of the static equivalence, $\Phi_1\theta\sigma_1 \sim_c^\ell \Phi_2\theta\sigma_2$ implies $\Phi_1\theta\sigma_1 \sim_c \Phi_2\theta\sigma_2$. Moreover, let ξ, ξ' two recipes built from $\Phi_1\theta\sigma_1$ such that $\ell(\xi\Phi_1\theta\sigma_1) = \ell(\xi'\Phi_1\theta\sigma_1)$. By Proposition 2, we know that $\ell(\xi\Phi_2\theta\sigma_2) = \ell(\xi\Phi_1\theta\sigma_1) = \ell(\xi'\Phi_1\theta\sigma_1) = \ell(\xi'\Phi_2\theta\sigma_2)$. Hence we conclude that $\Phi_1\theta\sigma_1 \sim_c \Phi_2\theta\sigma_2$ implies $\Phi_1\theta\sigma_1 \sim_c^\ell \Phi_2\theta\sigma_2$.

Lemma 9. *Let Φ_1 and Φ_2 two ground extended frame with same structure. If there exists a name a such that $(ax_1 \rhd a) \in \Phi_1$ and $(ax_1 \rhd a) \in \Phi_2$, then $\Phi_1 \sim_c^\ell \Phi_2$ implies that for all ξ built from Φ_1, $\ell(\xi\Phi_1) = \ell(\xi\Phi_2)$.*

Proof. Let's denote $\Phi_1 = \{\xi_1 \rhd u_1, \dots, \xi_n \rhd u_n\}$ and $\Phi_2 = \{\xi_1 \rhd v_1, \dots, \xi_n \rhd v_n\}$. Let ξ built from Φ_1. There exists C a context with constructor symbol such that $\xi = C[\xi_{\alpha_1}, \dots, \xi_{\alpha_p}]$ with $\alpha_1, \dots, \alpha_p \in \{1, \dots, n\}$. Thus $\xi\Phi_1 = C[u_{\alpha_1}, \dots, u_{\alpha_p}]$. For all $q \in \{1, \dots, p\}$, let ζ_{α_q} be the recipe obtained by replacing every name of u_{α_q} by ax_1. Since $ax_1\Phi_1 \in \mathcal{N}$, we have that $\ell(\xi\Phi_1) = \ell(C[\zeta_{\alpha_1}, \dots, \zeta_{\alpha_p}]\Phi_1)$. By hypothesis, we know that $\Phi \sim_c^\ell \Phi'$ hence $\ell(\xi\Phi_2) = \ell(C[\zeta_{\alpha_1}, \dots, \zeta_{\alpha_p}]\Phi_2)$. But since $ax_1\Phi_2 = ax_1\Phi_1 \in \mathcal{N}$, we deduce that $\ell(C[\zeta_{\alpha_1}, \dots, \zeta_{\alpha_p}]\Phi_1) = \ell(C[\zeta_{\alpha_1}, \dots, \zeta_{\alpha_p}]\Phi_2)$ and so $\ell(\xi\Phi_1) = \ell(\xi\Phi_2)$. ☐

Lemma 10. *For every $m \in \mathbb{N}^*$, for every $P \in \mathbb{R}[X_1, \dots, X_m]$, if there exist some subsets $Y_1, \dots, Y_m \subset \mathbb{R}$, such that for every k, $\#Y_k > d_{max}(P)$ and $\forall y \in Y_1 \times \dots \times Y_m, P(y_1, \dots, y_m) = 0$, then $P = 0$.*

Proof. Let H_m be "for every $P \in \mathbb{R}[X_1, \dots, X_m]$, if there exist some subsets $Y_1, \dots, Y_m \subset \mathbb{R}$, such that for every k, $\#Y_k > d_{max}(P)$ and $\forall y \in Y_1 \times \dots \times Y_m, P(y_1, \dots, y_m) = 0$, then $P = 0$". We prove by recurrence that for every $m \in \mathbb{N}^*$, H_m holds.

Base case $m = 1$: There exist at least $\#Y_1 > deg(P)$ zeros for P so $P = 0$ and H_1 holds.

Inductive step $m+1$: We can see P as a polynomial from $\mathbb{R}[X_1, \dots, X_m][X_{m+1}]$ by writing $P(y_1, \dots, y_{m+1}) = \sum_{k=0}^{r} \beta_k(y_1, \dots, y_m) y_{m+1}^k$ with $\beta_k \in \mathbb{R}[X_1, \dots, X_m]$ and $r < \#Y_{m+1}$. Thus, for every $(y_1, \dots, y_m) \in Y_1 \times \dots \times Y_m$, $Q(X_{m+1}) = P(y_1, \dots, y_m, X_{m+1}) \in \mathbb{R}[X_{m+1}]$. Moreover, $deg(Q) = r < \#Y_{m+1}$ and for every $y_{m+1} \in Y_{m+1}$, $Q(y_n) = 0$ so $Q = 0$. Hence, for every k, for every $(y_1, \dots, y_m) \in Y_1 \times \dots \times Y_m$, $\beta_k(y_1, \dots, y_m) = 0$ so according to H_m, $\beta_k = 0$. Finally, $P = 0$ and $H_m \Rightarrow H_{m+1}$. ☐

Let E be a set of terms. We denote by $\ell(E)$ the set $\{\ell(u) \mid u \in E\}$. By abuse of notations, we write $\sigma \in E_1 \times \cdots \times E_n$ if σ is a substitution of the form $\sigma(x_i) = t_i \in E_i$ for $1 \le i \le n$.

Proposition 3. *Let $E_1, \dots, E_k$ be sets of terms such that $\ell(E_i)$ is infinite. Let $x_1, \dots, x_k$ be variables. Let $\Phi = \{\xi_1 \rhd u_1, \dots, \xi_n \rhd u_n\}$ be an extended frame and $\Phi_i = \{\xi_1 \rhd v_1^i, \dots, \xi_n \rhd v_n^i\}$ be a finite family of extended frames, $1 \le i \le m$ such that $vars(\Phi)$ and $vars(\Phi_i)$ are included in $\{x_1, \dots, x_k\}$. Assume that*

$$\forall \sigma \in E_1 \times \cdots \times E_k \qquad \exists 1 \le i \le m \quad s.t.\ \forall 1 \le j \le n \qquad \ell(u_j\sigma) = \ell(v_j^i\sigma)$$

Then there is $1 \le i \le m$ such that $P_\Phi = P_{\Phi_i}$.

Proof. Since the variables $x_1, \dots, x_k$ are the variables in all the frames, then thanks to Proposition 1, we have that for all $\sigma \in E_1 \times \cdots \times E_k$, there is $1 \le i \le m$ such that

$$P_{u_j}(\ell(x_1\sigma), \dots, \ell(x_k\sigma)) = P_{v_j^i}(\ell(x_1\sigma), \dots, \ell(x_k\sigma))$$

for every $1 \leq j \leq n$ (Property (*)). We wish to show that there is $1 \leq i \leq m$ such that $P_{u_j} - P_{v^i_j} = 0$ for all $1 \leq j \leq n$. Suppose by contradiction that it is not the case, that is, for every $1 \leq i \leq m$, there is j_i such that $P_{u_{j_i}} - P^i_{v_{j_i}} \neq 0$. Consider the polynomial

$$P = \prod_{i=1}^{m} (P_{u_{j_i}} - P_{v^i_{j_i}})$$

Due to Property (*), for any $(z_1, \ldots, z_k) \in \ell(E_1) \times \cdots \times \ell(E_k)$, $P(z_1, \ldots, z_k) = 0$. Thanks to Lemma 10, we deduce that $P = 0$. Now ,since $\mathbb{R}[X_1, \ldots, X_k]$ is an integral domain, there is no zero divisors thus we deduce that there must be an index i such that $P_{u_{j_i}} - P_{v^i_{j_i}} = 0$, contradiction.

We therefore deduce that there is $1 \leq i \leq m$ such that $P_{u_j} - P_{v^i_j} = 0$ for all $1 \leq j \leq n$, that is $P_\Phi = P_{\Phi_i}$.

Theorem 5. *Let Σ, Σ' two sets of constraint systems that contains constraint systems with same structure. Assume that there exists $a \in \mathcal{N}$ such that for all $(\Phi; D; Eq) \in \Sigma \cup \Sigma'$, $(ax_1 \rhd a) \in \Phi$. Moreover, assume that for all $\mathcal{C}, \mathcal{C}' \in \Sigma \cup \Sigma'$, $\mathcal{C} \approx^c_s \mathcal{C}'$, $D = D'$ and $Eq = Eq'$ where $\mathcal{C} = (\Phi; D; Eq)$ and $\mathcal{C}' = (\Phi'; D'; Eq')$. Then $\{P_{\Phi(\mathcal{C})} \mid \mathcal{C} \in \Sigma\} = \{P_{\Phi(\mathcal{C}')} \mid \mathcal{C}' \in \Sigma'\}$ if and only if $\Sigma \approx^c_{s\ell} \Sigma'$.*

Proof. Assume first that $\{P_{\Phi(\mathcal{C})} \mid \mathcal{C} \in \Sigma\} = \{P_{\Phi(\mathcal{C}')} \mid \mathcal{C}' \in \Sigma'\}$. Let us show that $\Sigma \approx^c_{s\ell} \Sigma'$. Let $\mathcal{C}_1 \in \Sigma$. By hypothesis, we know that there is $\mathcal{C}_2 \in \Sigma'$ such that $P_{\Phi(\mathcal{C}_1)} = P_{\Phi(\mathcal{C}_2)}$. Let $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C}_1)$. By hypothesis, $\mathcal{C}_1 \approx^c_s \mathcal{C}_2$ hence there exists σ' such that $(\sigma', \theta) \in \mathsf{Sol}(\mathcal{C}_2)$ and $\Phi(\mathcal{C}_1)\theta\sigma \sim_c \Phi(\mathcal{C}_2)\theta\sigma'$.

We deduce from Corollary 2 that $\Phi(\mathcal{C}_1)\theta\sigma \sim^\ell_c \Phi(\mathcal{C}_2)\theta\sigma'$. Therefore $\mathcal{C}_1 \approx^c_{s\ell} \mathcal{C}_2$ and thus $\Sigma \subseteq_\ell \Sigma'$. Symmetrically, we must have $\Sigma' \subseteq_\ell \Sigma$ and therefore $\Sigma \approx^c_{s\ell} \Sigma'$.

Reciprocally, assume $\Sigma \approx^c_{s\ell} \Sigma'$. Let us show that $\mathcal{P}(\Sigma) \subseteq \mathcal{P}(\Sigma')$ Let $\mathcal{C}_1 \in \Sigma$. By hypothesis, for all $\mathcal{C}_2 \in \Sigma'$, $\{\mathcal{C}_1\} \approx^c_s \{\mathcal{C}_2\}$. Moreover, by Lemma 7, $S(\Sigma, \Sigma') \subseteq \mathsf{Sol}(\mathcal{C}_1)$ and $S(\Sigma, \Sigma') \subseteq \mathsf{Sol}(\mathcal{C}_2)$, for all $\mathcal{C}_2 \in \Sigma'$. Thus for all $(\sigma, \theta) \in S(\Sigma, \Sigma')$, we have $\Phi(\mathcal{C}_1)\theta\sigma \sim_c \nu\Phi(\mathcal{C}_2)\theta\sigma$. Moreover, since $\Sigma \approx^c_{s\ell} \Sigma'$, then for all $(\sigma, \theta) \in S(\Sigma, \Sigma')$, there exists $\mathcal{C}'_2 \in \Sigma'$ such that $\Phi(\mathcal{C}_1)\sigma \sim^\ell_c \Phi(\mathcal{C}'_2)\sigma$.

Let $\Phi(\Sigma') = \{\Phi_1, \ldots, \Phi_k\}$. Let $\Phi(\mathcal{C}_1) = \{ax_{1}, 1 \rhd u_1, \ldots, ax_n, n \rhd u_n\}$ and $\Phi_i = \{ax_1, 1 \rhd v^i_1, \ldots, ax_n, n \rhd v^i_n\}$. If we denote $E_1, \ldots, E_k$ the sets associated to $S(\Sigma, \Sigma')$ (see Definition 20), then for all $\sigma \in E_1 \times \cdots \times E_k$, there is $1 \leq i \leq k$ such that $\Phi(\mathcal{C}_1)\sigma \sim^\ell_c \Phi_i\sigma$, that is $\ell(u_j\sigma) = \ell(v^i_j\sigma)$ for all $1 \leq j \leq n$ thanks to Lemma 9. Since by Lemma 8, $\ell(E_i)$ is infinite, for all $1 \leq i \leq k$, then by applying Proposition 3, we deduce that there is $1 \leq i \leq k$ such that $P_{\Phi(\mathcal{C}_1)} = P_{\Phi_i}$, that is there is $\mathcal{C}_2 \in \Sigma'$ such that $P_{\Phi(\mathcal{C}_1)} = P_{\Phi(\mathcal{C}_2)}$.

This shows that $\mathcal{P}(\Sigma) \subseteq \mathcal{P}(\Sigma')$. Symmetrically, we have $\mathcal{P}(\Sigma') \subseteq \mathcal{P}(\Sigma)$. □

The proof of Theorem 2 and 3 follows from Theorems 4 and 5.