Guessing Attacks and the Computational Soundness of Static Equivalence*

Mathieu Baudet[1] Bogdan Warinschi[2] Martín Abadi[3,4]

[1] ANSSI, Paris, France
[2] University of Bristol, United Kingdom
[3] University of California, Santa Cruz, USA
[4] Microsoft Research, Silicon Valley, USA

Abstract

The indistinguishability of two pieces of data (or two lists of pieces of data) can be represented formally in terms of a relation called static equivalence. Static equivalence depends on an underlying equational theory. The choice of an inappropriate equational theory can lead to overly pessimistic or overly optimistic notions of indistinguishability, and in turn to security criteria that require protection against impossible attacks or—worse yet—that ignore feasible ones. In this paper, we define and justify an equational theory for standard, fundamental cryptographic operations. This equational theory yields a notion of static equivalence that implies computational indistinguishability. Static equivalence remains liberal enough for use in applications. In particular, we develop and analyze a principled formal account of guessing attacks in terms of static equivalence.

1 Introduction

In the study of security, it is frequent to reason about whether two pieces of data can be distinguished by an observer. For example, the pieces of data might be two encrypted messages, and the observer an attacker that attempts to learn something about the underlying cleartexts by analyzing the encrypted messages. The two encrypted messages are indistinguishable if, no matter how the attacker operates on them, it cannot discern any meaningful difference. The encrypted messages may however be different—for instance, they may look like different random numbers.

Formally, indistinguishability can be represented in terms of a relation called static equivalence [3]. Roughly, two terms (and, more generally, two lists of terms) are statically equivalent when they satisfy all the same equations. This relation is essentially a special case of the observational equivalence relation of process calculi. It is simpler than observational equivalence in that it does not allow for continued interaction between a system and an observer: the observer gets data once and then conducts experiments on its own. Nevertheless, observational equivalence can be reduced to a combination of static equivalence and usual bisimulation requirements [3, 4, 18].

Static equivalence depends on an underlying equational theory. The choice of an inappropriate equational theory can lead to overly pessimistic or optimistic notions of indistinguishability, and in turn to security criteria that require protection against impossible attacks or—worse yet—that ignore feasible ones.

In this paper, we define an equational theory for standard, fundamental cryptographic operations, and we justify and apply the resulting concept of static equivalence. These operations include various flavors of encryption and decryption. Static equivalence in this theory implies computational indistinguishability. In other words, if the formal notion of static equivalence indicates that two pieces of data are indistinguishable, then no computationally feasible experiment can tell those two pieces of data apart. (This property is a

*This work has been presented in preliminary form at a conference [1].

soundness theorem. Although it is less important, we have also explored completeness; we discuss it only briefly in this paper.) Our notion of computational feasibility is based on the sorts of assumptions typically employed in complexity-theoretic cryptography. It includes certain assumptions on the security properties of the cryptographic operations; those assumptions appear reasonable and fairly standard, but so do others, and picking satisfactory ones is somewhat delicate.

While static equivalence is conservative enough to exclude feasible attacks, it also remains liberal enough for use in applications. In particular, we develop a formal account of guessing attacks (e.g., [26, 27, 15, 35, 34]) in terms of static equivalence. Since guessing attacks constitute a significant threat against protocols that rely on passwords and other weak secrets, the recent literature contains several studies of guessing attacks, with both formal and computational approaches (e.g., [30, 22, 21, 20, 8, 12, 19, 25, 28, 24]). Formal approaches are attractive because of their relative simplicity, which often enables automation. On the other hand, formal approaches are rather varied and sometimes ad hoc. Fortunately, it has been suggested that a formulation of guessing attacks could be based on static equivalence [20, 23]. We believe that this idea has a number of virtues. It leads to a crisp definition, it is fairly independent of specific choices of cryptographic operations, and it extends nicely to general process calculi. To date, however, this idea has not been worked out fully, in the setting of an appropriate equational theory. We aim to address this gap.

A related, frequent shortcoming of formal analyses is the lack of computational justifications. This lack allows the possibility that a protocol is safe against attacks formally, but that a feasible attack exists nonetheless. An active line of recent research aims to address such shortcomings, by defining and proving computational soundness results for formal methods (e.g., [5, 7, 32, 29]). The results of this paper belong in this line of research. We prove a computational soundness theorem for our choice of equational theory with respect to an implementation based on standard cryptographic primitives. This theorem not only enables symbolic verification that yields computational guarantees, but also offers further justification for our choice of equational theory.

Related work. More specifically, our results build on some of our previous work and expand it [6, 9]. Next we discuss in more detail the relation with this previous work. Other less-closely related work is discussed above and throughout the paper.

Our previous work includes a study of static equivalence [9]. The theories considered there do not include the one that we define in this paper (in part because those theories do not model probabilistic encryption functions, nor encryption under weak keys) and have not provided a satisfactory account of guessing attacks. They are nonetheless an important piece of the context of the present paper.

Our previous work also presents an approach towards computationally sound security analysis of password-based protocols [6]. That approach relies on an ad hoc extension of the concept of patterns [5], rather than on static equivalence. Patterns are symbolic representations of the information gleaned from expressions by passive adversaries; secure use of passwords in expressions is identified via a careful analysis of their corresponding patterns. Informally, this analysis ensures that the expressions do not contain redundancy that would help an adversary in verifying a password guess. Unfortunately, the analysis is highly dependent on the syntax of the language and on a specific choice of cryptographic primitives. Any extension of the syntax requires adjustment of the pattern-based security notion (and of the corresponding soundness proofs). For example, although extending the previous work to the case of multiple passwords seems possible, such a step requires new definitions and analysis.

The approach presented in this paper is more uniform. In particular, the concept of static equivalence is independent from the syntax of the language. Thus, the theory that we develop applies to both expressions that use a single password and to those that use multiple passwords. Accordingly, we require somewhat stronger assumptions on password-based encryption. Our previous work shows that weaker assumptions suffice in special cases. Interestingly, it may also indicate how to prove completeness results. Specifically, existing techniques for achieving completeness of formalisms based on patterns [31] appear to extend to password-based encryption. It should thereby be possible to obtain completeness results at least for a fragment of the language defined in this paper.

Structure of the paper. The next section, Section 2, presents a formal model: it defines sorted terms, an equational theory for them, and the corresponding notion of static equivalence. Section 3 interprets the syntax of the formal model in a computational universe; it includes cryptographic assumptions. Section 4 establishes the computational soundness of static equivalence for the equational theory. Section 5 applies our results to the study of guessing attacks. Section 6 concludes. An appendix contains additional details.

2 Abstract Model

We represent cryptographic messages in an abstract way, by terms over a many-sorted signature, equipped with an equational theory. The equational theory that we use extends well-established ones for encryption with carefully chosen equations for password-based encryption. Importantly, our choices enable a soundness theorem with respect to an implementation based on standard cryptographic primitives. We comment on some specific choices of equations later in the paper.

2.1 Sorts and Terms

The set of *sorts (or types)* that we consider is defined by the following grammar:

τ	::=		
	|	$SKey$	symmetric keys
	|	$EKey$	(public) encryption keys
	|	$DKey$	(private) decryption keys
	|	$Data$	passwords and other data
	|	$Coins$	coins for encryption
	|	$Pair[\tau_1, \tau_2]$	pairs of messages
	|	$SCipher[\tau]$	symmetric encryptions of messages of type τ
	|	$ACipher[\tau]$	asymmetric encryptions of messages of type τ

The set of *(well-sorted) terms*, written S, T, U, V, ..., is built from an infinite number of variables x, y, ... and names a, b, n, r, k, sk, pk, ... for each sort, with the following function symbols:

enc_τ	:	$\tau \times Data \to \tau$	encryption under data
dec_τ	:	$\tau \times Data \to \tau$	decryption with data
penc_τ	:	$\tau \times EKey \times Coins \to ACipher[\tau]$	public-key encryption
pdec_τ	:	$ACipher[\tau] \times DKey \to \tau$	private-key decryption
pub	:	$DKey \to EKey$	public-key extraction
$\mathsf{pdec\_success}_\tau$	:	$ACipher[\tau] \times DKey \to Data$	domain predicate for private-key decryption
senc_τ	:	$\tau \times SKey \times Coins \to SCipher[\tau]$	symmetric encryption
sdec_τ	:	$SCipher[\tau] \times SKey \to \tau$	symmetric decryption
$\mathsf{sdec\_success}_\tau$	:	$SCipher[\tau] \times SKey \to Data$	domain predicate for symmetric decryption
$\mathsf{pair}_{\tau_1,\tau_2}$	:	$\tau_1 \times \tau_2 \to Pair[\tau_1, \tau_2]$	pairing
$\mathsf{fst}_{\tau_1,\tau_2}$	:	$Pair[\tau_1, \tau_2] \to \tau_1$	first projection
$\mathsf{snd}_{\tau_1,\tau_2}$	:	$Pair[\tau_1, \tau_2] \to \tau_2$	second projection
$0, 1$	:	$Data$	boolean constants
$\mathsf{w}, \mathsf{c}_0, \mathsf{c}_1 \ldots$	:	$Data$	additional data constants

Encryption and decryption symbols may not be available for all sorts τ. We let T_{penc} be the set of types τ for which the symbols penc_τ, pdec_τ, and $\mathsf{pdec\_success}_\tau$ are available, and define T_{senc} and T_{enc} analogously. We assume that pairs are not encrypted under data values, that is, $T_{\mathsf{enc}} \cap \{Pair[\tau_1, \tau_2]\}_{\tau_1,\tau_2} = \emptyset$. This choice is motivated by typical implementations of the pairing operation that use special delimiters to make the

operation general and easily invertible. Encryptions of such pairs under data values immediately lead to off-line attacks.

Our function symbols represent encryption and decryption functions and auxiliary operations. The first two functions (enc_τ and dec_τ) are to be used with data values as keys; the data values may be the constant symbols of the grammar, which may represent the passwords in a dictionary. (In contrast, fresh names may represent strong keys; the scoping rules justify the respective uses of constant symbols and names.) The fact that enc_τ does not take a parameter of type *Coins* relates to the difficulties with probabilistic password-based encryption [6]. Moreover, the language provides no direct way for the attacker to check that a value results from applying enc_τ with a particular key. Such properties are essential for thwarting guessing attacks in practice (for example, in the EKE protocol [15]). The remaining functions are fairly standard; they include functions for public-key and symmetric encryption (penc_τ and senc_τ), which are probabilistic in the sense that they take a parameter of type *Coins*.

We often omit type annotations on function symbols. For instance, provided that S, T, and U have type *Data*, we may write $\mathsf{pair}(\mathsf{enc}(S,T),U)$ instead of $\mathsf{pair}_{Data,Data}(\mathsf{enc}_{Data}(S,T),U)$. In addition, we sometimes use the abbreviations $\{S\}_T$ for $\mathsf{enc}(S,T)$, $\{S\}^r_{\mathsf{pub}(sk)}$ for $\mathsf{penc}(S,\mathsf{pub}(sk),r)$, and $\{S\}^r_k$ for $\mathsf{senc}(S,k,r)$.

We write $\mathrm{var}(T)$ and $\mathrm{names}(T)$ for the sets of variables and names that occur in a term T. We extend the notation to tuples and sets of terms. A term T is *ground* or *closed* when $\mathrm{var}(T) = \emptyset$. We write $\sigma = \{x_1 \mapsto T_1, \ldots, x_n \mapsto T_n\}$ for a substitution, and let $\mathrm{dom}(\sigma) = \{x_1, \ldots, x_n\}$, $\mathrm{var}(\sigma) = \mathrm{var}(T_1, \ldots, T_n)$, and $\mathrm{names}(\sigma) = \mathrm{names}(T_1, \ldots, T_n)$. A substitution σ is *ground* or *closed* when $\mathrm{var}(\sigma) = \emptyset$. We consider only well-sorted substitutions (that is, for each i, $T_i = x_i\sigma$ has the same sort as x_i).

2.2 Equational Theory

We model the semantics of the cryptographic primitives by equipping terms with an equational theory, that is, a reflexive, symmetric, transitive relation, stable by (well-sorted) substitutions of terms for variables and (in this case) for names, and stable by application of contexts. Specifically, we consider the equational theory $=_E$ generated by the following equations:

$$\begin{aligned}
\mathsf{dec}_\tau(\mathsf{enc}_\tau(x,y),y) &= x & \qquad \mathsf{enc}_\tau(\mathsf{dec}_\tau(x,y),y) &= x \\
\mathsf{pdec}_\tau(\mathsf{penc}_\tau(x,\mathsf{pub}(y),z),y) &= x & \qquad \mathsf{pdec_success}_\tau(\mathsf{penc}_\tau(x,\mathsf{pub}(y),z),y) &= 1 \\
\mathsf{sdec}_\tau(\mathsf{senc}_\tau(x,y,z),y) &= x & \qquad \mathsf{sdec_success}_\tau(\mathsf{senc}_\tau(x,y,z),y) &= 1 \\
\mathsf{fst}_{\tau_1,\tau_2}(\mathsf{pair}_{\tau_1,\tau_2}(x,y)) &= x & \qquad \mathsf{snd}_{\tau_1,\tau_2}(\mathsf{pair}_{\tau_1,\tau_2}(x,y)) &= y \\
\mathsf{pair}_{\tau_1,\tau_2}(\mathsf{fst}_{\tau_1,\tau_2}(x),\mathsf{snd}_{\tau_1,\tau_2}(x)) &= x
\end{aligned}$$

where the symbols x, y, and z represent variables of the appropriate sorts. Most of the equations are fairly standard. The symbols $\mathsf{pdec\_success}_\tau$ and $\mathsf{sdec\_success}_\tau$, and the corresponding equations, are used for checking the success of symmetric and asymmetric decryptions. Indeed, such cryptographic operations may visibly fail when executed with the wrong decryption key. The only surprise may be the inclusion of $\mathsf{enc}_\tau(\mathsf{dec}_\tau(x,y),y) = x$, without which an attacker that sees x and guesses y might confirm whether x is a ciphertext encrypted under y by decrypting x with y, reencrypting with y, and comparing the result to x; the equation implies that the comparison always succeeds, whether the guess was correct or not. So, for instance, $\mathsf{enc}_\tau(n,\mathsf{c}_0)$ and $\mathsf{enc}_\tau(n,\mathsf{c}_1)$ are shown to be indistinguishable when n is a fresh name of sort τ. Such consequences of the equation are important for the security of protocols that rely on weak secrets. Moreover, the equation holds in many reasonable implementations, in particular those based on keyed permutations.

When oriented from left to right, the equations above form a convergent rewriting system that we call $\mathcal{R}$. Note that in our case $\mathcal{R}$ is infinite, because each of the equations is parameterized by one or more types.

2.3 Frames

Frames represent sets of messages available to an observer (for example, because they were sent over a public network) [3]. More precisely, a *frame* is an expression $\varphi = \nu\tilde{n}.\{x_1 = T_1, \ldots, x_n = T_n\}$ where $\tilde{n}$ is a set of *restricted* names, and each T_i is a closed term of the same sort as x_i. For simplicity, we require (without loss

of generality) that every name in use be restricted, that is, $\tilde{n} = \text{names}(T_1, \ldots, T_n)$. A name k may still be disclosed explicitly, for instance by a dedicated mapping $x_i = k$. Therefore, we tend to omit the binders $\nu\tilde{n}$, and identify a frame φ with its underlying substitution $\{x_1 \mapsto T_1, \ldots, x_n \mapsto T_n\}$.

As an example, we give the frame associated to an execution of the Encrypted Key Exchange (EKE) protocol [15]. We revisit this example in Sections 4.2 and 5. The flows of the protocol between parties A and B that share a given password w_{AB} are as follows.

1. A generates an asymmetric key (pk, sk) and sends $\{pk\}_{\mathsf{w}_{AB}}$ to B.
2. B decrypts this message using w_{AB}. Then B generates a symmetric key k_1 and sends $\{\{k_1\}^{r_1}_{pk}\}_{\mathsf{w}_{AB}}$ to A.
3. At this point the parties share the key k_1, and check if the protocol was executed as expected. For this purpose, A generates a random nonce n_A and sends $\{n_A\}^{r_2}_{k_1}$ to B.
4. Upon receiving this message, B obtains n_A, generates a new nonce n_B, and sends $\{\mathsf{pair}(n_A, n_B)\}^{r_3}_{k_1}$ to A.
5. A decrypts this message and checks that the first component of the resulting pair is n_A. If so, it obtains n_B, sends $\{n_B\}^{r_4}_{k_1}$ to B, and terminates successfully.
6. Finally, B decrypts this last message, verifies that it contains the nonce n_B it previously sent to A, and if so, it terminates successfully.

The frame associated with one instance of these flows is:

$$\varphi = \nu pk, k_1, n_A, n_B, r_1, r_2, r_3, r_4.\ \{x_1 = \{pk\}_{\mathsf{w}_{AB}},\ x_2 = \{\{k_1\}^{r_1}_{pk}\}_{\mathsf{w}_{AB}},\ x_3 = \{n_A\}^{r_2}_{k_1},\ x_4 = \{\mathsf{pair}(n_A, n_B)\}^{r_3}_{k_1},\ x_5 = \{n_B\}^{r_4}_{k_1}\ \}$$

where pk is a name of sort *EKey*, k_1 is a name of sort *SKey*, n_A and n_B are names of sort *Data*, and r_1, r_2, r_3, and r_4 are names of sort *Coins*.

2.4 Static Equivalence of Frames

Two frames φ_1 and φ_2 such that $\text{dom}(\varphi_1) = \text{dom}(\varphi_2)$ are *statically equivalent* (written $\varphi_1 \approx_E \varphi_2$) if, for every pair of terms (M, N) such that $\text{var}(M, N) \subseteq \text{dom}(\varphi_1)$ and $\text{names}(M, N) \cap \text{names}(\varphi_1, \varphi_2) = \emptyset$, it holds that $M\varphi_1 =_E N\varphi_1$ if and only if $M\varphi_2 =_E N\varphi_2$. Proving static equivalence may not be easy. Fortunately, efficient methods exist in many cases (e.g., [2, 17]). In particular, static equivalence is decidable in polynomial time for unsorted convergent subterm theories [2]; we expect that this result carries over to sorted convergent subterm theories such as $=_E$. The proof of our main result (Section 4) provides an alternative decision procedure for the class of frames and the equational theory under consideration.

We close this section with a few examples of equivalences and inequivalences under the theory E:

$$\{x = \{0\}^r_k\} \quad \approx_E \quad \{x = \{1\}^r_k\} \tag{1}$$

$$\{x = \{0\}^r_k,\ \ y = \{0\}^{r'}_k\} \quad \approx_E \quad \{x = \{1\}^r_k,\ \ y = \{0\}^{r'}_{k'}\} \tag{2}$$

$$\{x = \{n\}_{\mathsf{w}},\ \ y = \{m\}_{\mathsf{w}}\} \quad \approx_E \quad \{x = a_1,\ \ y = a_2\} \tag{3}$$

$$\{x = \{\{n\}_{\mathsf{w}}\}_{\mathsf{w}},\ \ y = \{m\}_{\mathsf{w}}\} \quad \approx_E \quad \{x = a_1,\ \ y = a_2\} \tag{4}$$

$$\{x = \{\{0\}^{r_1}_{\mathsf{pub}(sk)}\}_{\mathsf{w}},\ \ y = \{0\}^{r_2}_{\mathsf{pub}(sk)}\} \quad \approx_E \quad \{x = a_1,\ \ y = a_2\} \tag{5}$$

$$\{x = \{\{0\}^{r_1}_{\mathsf{pub}(sk)}\}_{\mathsf{w}},\ \ y = \{0\}^{r_1}_{\mathsf{pub}(sk)}\} \quad \approx_E \quad \{x = \{a_1\}_{\mathsf{w}},\ \ y = a_1\} \tag{6}$$

$$\{x = \{\{n\}^{r_1}_k\}_{\mathsf{w}},\ \ y = k\} \quad \not\approx_E \quad \{x = a_1,\ \ y = k\} \tag{7}$$

Examples (1) and (2) are simple examples about symmetric encryptions under strong keys, illustrating that those encryptions hide plaintexts and also equalities of plaintexts or keys across encryptions. Examples (3) and (4) illustrate that encryptions of fresh names under a constant w (intuitively, under a weak secret) can look like fresh names. The values of x and y are two such encryptions—and the former is in fact a double encryption in example (4)—with unrelated underlying names. Example (5) resembles example (4); it illustrates that an encryption of a public-key ciphertext $\{0\}^{r_1}_{\mathsf{pub}(sk)}$ under w can look like a fresh name. In examples (3)–(5), the plaintexts being encrypted are not otherwise available to the observer, though somewhat related plaintexts may be (as the values of the variable y). Example (6) treats a case in which the observer also obtains the plaintext being encrypted, through y; in that case, the observer can see a relation between the value of x and the value of y, namely that the former is an encryption of the latter under w. Example (7) indicates that the observer that is given k can distinguish $\{\{n\}^{r_1}_k\}_{\mathsf{w}}$ from a fresh name; intuitively, after decrypting with w, the adversary can tell if what it sees is a ciphertext under k or not, since the success of shared-key decryption is detectable. Formally, this distinction corresponds to the discriminating test $\mathsf{sdec\_success}(\mathsf{dec}(x,\mathsf{w}),y) =^? 1$ which is valid on the left-hand frame (modulo E) but not on the right-hand one. A decision procedure to justify the other equivalences is detailed in Subsection 4.2 or may be found in previous work [2].

3 Implementation

In this section we interpret the syntax of the formal model in a computational universe. We also discuss cryptographic assumptions on which the implementation relies.

3.1 Interpreting the Syntax

Next we detail the mapping from terms to distribution ensembles over bit-strings. We write η for a (global) security parameter, typically related to the sizes of keys in cryptographic schemes.

3.1.1 Encryption schemes

The mapping uses a public-key encryption scheme $\Pi^p = (\mathcal{K}^p, \mathcal{E}^p, \mathcal{D}^p)$ and a symmetric encryption scheme $\Pi^s = (\mathcal{K}^s, \mathcal{E}^s, \mathcal{D}^s)$. It also uses a symmetric, deterministic, type-preserving encryption scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$. (The definition of type preserving is given below.) In each of these triples, the first component is a key-generation algorithm, the second an encryption algorithm, and the third a decryption algorithm. Each of these algorithms must be probabilistic polynomial time (PPTIME). In the case of public-key encryption Π^p, we assume that there exists a PPTIME algorithm $\mathcal{K}^{\mathsf{pub}}$ for mapping private keys to corresponding public keys.

For each value η of the security parameter, we write $k \xleftarrow{R} \mathcal{K}_\eta$ (respectively $k \xleftarrow{R} \mathcal{K}^s_\eta$) to indicate that k is the output of the key generation algorithm $\mathcal{K}$ (respectively $\mathcal{K}^s$) on input 1^η. Similarly, we write $(pk, sk) \xleftarrow{R} \mathcal{K}^p_\eta$ to indicate that (pk, sk) are encryption/decryption keys produced by the key generation algorithm of Π^p on input 1^η.

As usual, the encryption functions $\mathcal{E}^p$ and $\mathcal{E}^s$ are randomized; we write $\mathcal{E}^p(m,k,r)$ and $\mathcal{E}^s(m,k,r)$ for public-key and symmetric encryptions, respectively, of message m under encryption key k with random coins r. We write $c \xleftarrow{R} \mathcal{E}^p(m,k)$ and $c \xleftarrow{R} \mathcal{E}^s(m,k)$ for the corresponding encryption processes, using fresh random coins.

We assume that the set of keys for Π is of the form $\{0,1\}^{\alpha_1(\eta)}$, and that the set of coins for Π^s and Π^p is $\{0,1\}^{\alpha_2(\eta)}$, where the functions $\alpha_1(\eta)$ and $\alpha_2(\eta)$ are polynomially bounded and at least linearly increasing. We say that Π is *type-preserving* when, for every $\tau \in T_{\mathsf{enc}}$, encryption and decryption by Π map $[\![\tau]\!]_\eta$—the set of bit-strings that corresponding to the type τ—to itself.

3.1.2 Sorts, functions, and random drawings

For each value of the security parameter η, the concrete meaning of sorts and terms is characterized (much as in [9]) by:

- for each sort τ, a carrier set $[\![\tau]\!]_\eta$;
- for each function symbol $f : \tau_1 \times \ldots \times \tau_n \to \tau$, a function $[\![f]\!]_\eta : [\![\tau_1]\!]_\eta \times \ldots \times [\![\tau_n]\!]_\eta \to [\![\tau]\!]_\eta$;
- for each sort τ, a procedure written $e \xleftarrow{R} [\![\tau]\!]_\eta$ for drawing a random element e from $[\![\tau]\!]_\eta$, according to a distribution written $(\xleftarrow{R} [\![\tau]\!]_\eta)$.

Specifically, the carrier set $[\![\tau]\!]_\eta$ of a type τ is defined inductively:

$$\begin{array}{rcl}
[\![SKey]\!]_\eta & = & \text{“}SKey\text{”} \,\|\, \{\text{symmetric keys for } \Pi^s(\eta)\} \\
[\![EKey]\!]_\eta & = & \text{“}EKey\text{”} \,\|\, \{\text{public keys for } \Pi^p(\eta)\} \\
[\![DKey]\!]_\eta & = & \text{“}DKey\text{”} \,\|\, \{\text{private keys for } \Pi^p(\eta)\} \\
[\![Data]\!]_\eta & = & \text{“}Data\text{”} \,\|\, \{0,1\}^{\alpha_1(\eta)} \\
[\![Coins]\!]_\eta & = & \text{“}Coins\text{”} \,\|\, \{0,1\}^{\alpha_2(\eta)} \\
[\![Pair[\tau_1,\tau_2]]\!]_\eta & = & \text{“}Pair\text{”} \,\|\, [\![\tau_1]\!]_\eta \,\|\, [\![\tau_2]\!]_\eta \\
[\![SCipher[\tau]]\!]_\eta & = & \text{“}SCipher\text{”} \,\|\, \tau \,\|\, \{\text{images of } [\![\tau]\!]_\eta \text{ by } \Pi^s(\eta)\} \\
[\![ACipher[\tau]]\!]_\eta & = & \text{“}ACipher\text{”} \,\|\, \tau \,\|\, \{\text{images of } [\![\tau]\!]_\eta \text{ by } \Pi^p(\eta)\}
\end{array}$$

where $\|$ denotes the concatenation of bit-strings (applied by extension on sets of bit-strings), and we assume an encoding of identifiers for types τ into bit-strings.

The meaning of function symbols is as follows:

- Symbols $\mathsf{pair}_{\tau_1,\tau_2}$, $\mathsf{fst}_{\tau_1,\tau_2}$, and $\mathsf{snd}_{\tau_1,\tau_2}$ are implemented on bit-strings by tagged concatenation and projections, as one might expect.
- Constants w, c_0, c_1, ... are mapped to arbitrary PPTIME-computable sequences of bit-strings of length $\alpha_1(\eta)$, prefixed with the tag “*Data*”; 0 and 1 are mapped respectively to “*Data*” $\| 0^{\alpha_1(\eta)}$ and “*Data*” $\| 1^{\alpha_1(\eta)}$.
- For every $\tau \in T_{\mathsf{penc}}$, the implementations of penc_τ, pdec_τ, and $\mathsf{pdec\_success}_\tau$ are defined by:

$$[\![\mathsf{penc}_\tau]\!]_\eta(m, \text{“}EKey\text{”}||pk, \text{“}Coins\text{”}||r) = \text{“}ACipher\text{”}\|\tau\|\mathcal{E}^p_\tau(m, pk, r)$$

$$[\![\mathsf{pdec}_\tau]\!]_\eta(m, \text{“}DKey\text{”}||sk) = \begin{cases} \mathcal{D}^p(c, sk) & \text{if } m = \text{“}ACipher\text{”}\|\tau\|c \text{ and the} \\ & \text{decryption } \mathcal{D}^p(c, sk) \text{ succeeds} \\ c_{\eta,\tau} & \text{otherwise} \end{cases}$$

$$[\![\mathsf{pdec\_success}_\tau]\!]_\eta(m, \text{“}DKey\text{”}||sk) = \text{“}Data\text{”} \,\| \begin{cases} 1^{\alpha_1(\eta)} & \text{if } m = \text{“}ACipher\text{”}\|\tau\|c \text{ and the decryption } \mathcal{D}^p(c, sk) \text{ succeeds} \\ 0^{\alpha_1(\eta)} & \text{otherwise} \end{cases}$$

  where $c_{\eta,\tau}$ denotes some arbitrarily fixed constant in $[\![\tau]\!]_\eta$. The implementations of senc_τ, sdec_τ, and $\mathsf{sdec\_success}_\tau$, for $\tau \in T_{\mathsf{senc}}$, are defined similarly.
- For every $\tau \in T_{\mathsf{enc}}$, the implementations of enc_τ and dec_τ are defined by:

$$\begin{array}{rcl}
[\![\mathsf{enc}_\tau]\!]_\eta(m, \text{“}Data\text{”}\|k) & = & \mathcal{E}(m, k) \\
[\![\mathsf{dec}_\tau]\!]_\eta(c, \text{“}Data\text{”}\|k) & = & \mathcal{D}(c, k)
\end{array}$$

  We assume that $\mathcal{E}(\cdot, k)$ and $\mathcal{D}(\cdot, k)$ are inverse bijections from $[\![\tau]\!]_\eta$ to itself. In particular, tags are left unchanged by these functions.

- Finally, the symbol pub is mapped to the function $\mathcal{K}^{\mathsf{pub}}$ for extracting public keys from private keys.

The distribution $(\xleftarrow{R} [\![\tau]\!]_\eta)$ is defined inductively on τ as follows. (This description omits the obvious tags.)

- If τ is one of *SKey*, *EKey*, and *DKey*, then $(\xleftarrow{R} [\![\tau]\!]_\eta)$ is the output distribution of, respectively, the algorithms $\mathcal{K}^s$, $\mathit{fst}(\mathcal{K}^p)$, and $\mathit{snd}(\mathcal{K}^p)$, on input η.
- If τ is *Data* or *Coins*, then $(\xleftarrow{R} [\![\tau]\!]_\eta)$ is the uniform distribution over $[\![\tau]\!]_\eta$.
- If $\tau = \mathit{Pair}[\tau_1, \tau_2]$, then $(\xleftarrow{R} [\![\tau]\!]_\eta)$ is the output distribution of the algorithm that samples two random elements according to $(\xleftarrow{R} [\![\tau_1]\!]_\eta)$ and $(\xleftarrow{R} [\![\tau_2]\!]_\eta)$, then outputs their concatenation.
- If τ is $\mathit{SCipher}[\tau]$ or $\mathit{ACipher}[\tau]$, then $(\xleftarrow{R} [\![\tau]\!]_\eta)$ is the output distribution of an algorithm that outputs an encryption of a plaintext sampled according to $(\xleftarrow{R} [\![\tau]\!]_\eta)$ under a fresh encryption key of the appropriate kind.

3.1.3 Interpreting terms and frames

Given η, we associate with each frame $\varphi = \nu\widetilde{n}.\{x_1 = T_1, \ldots, x_n = T_n\}$ a distribution $[\![\varphi]\!]_\eta$ defined by the following procedure for computing a sample $\phi \xleftarrow{R} [\![\varphi]\!]_\eta$:

1. for each name of sort τ that occurs in φ, draw a value $\widehat{a} \xleftarrow{R} [\![\tau]\!]_\eta$;
2. compute the value $\widehat{T_i}$ of each closed term T_i, recursively:
$$\text{for every function symbol } f, \quad \widehat{f(S_1, \ldots, S_n)} = [\![f]\!]_\eta(\widehat{S_1}, \ldots, \widehat{S_1})$$
3. let the resulting *concrete frame* be $\phi = \{x_1 = \widehat{T_1}, \ldots, x_n = \widehat{T_n}\}$.

We define the notation $[\![\_]\!]_\eta$ for closed terms and tuples of closed terms similarly. We write $[\![\varphi]\!]_{\eta,\, a_1 \mapsto e_1,\, \ldots,\, a_n \mapsto e_n}$ so as to specify the values for names, and $[\![\varphi]\!]_{\eta,\, \mathsf{c}_1 \mapsto e_1,\, \ldots,\, \mathsf{c}_n \mapsto e_n}$ so as to specify the values of the constants c_1, $\ldots$, c_n. We write $[\![\varphi]\!]$ for the *ensemble* (family of distributions) $([\![\varphi]\!]_\eta)_\eta$. We identify a single-valued (Dirac) distribution with its unique value.

Note that, by construction, the sampling of values for any (fixed) term or frame according to $[\![\_]\!]_\eta$ is computable in PPTIME.

3.1.4 Indistinguishability

Two ensembles $D^1 = (D^1_\eta)_\eta$ and $D^2 = (D^2_\eta)_\eta$ are *indistinguishable* (written $D^1 \approx D^2$) when, for every PPTIME adversary A, the function

$$\mathrm{Adv}_A(\eta) = \Pr\left[e \xleftarrow{R} D^1_\eta : A(e) = 1\right] - \Pr\left[e \xleftarrow{R} D^2_\eta : A(e) = 1\right]$$

is negligible (that is, asymptotically smaller than any inverse polynomial).

3.2 Cryptographic Assumptions

We use symmetric and asymmetric encryption schemes that satisfy a notion of security related to type-0 and type-1 security [5]. Essentially, we require that for each type τ, the encryption function restricted to elements of $[\![\tau]\!]$ reveal no information about the key used for encryption and hide all partial information about underlying plaintexts—except for their belonging to the carrier set $[\![\tau]\!]$.

Definition 1. Let $\Pi^s = (\mathcal{K}^s, \mathcal{E}^s, \mathcal{D}^s)$ be a symmetric encryption scheme. For each η and type $\tau \in T_{\mathsf{senc}}$, we consider the following experiment, with a two-stage PPTIME adversary $A = (A_1, A_2)$:

- a key k is generated via $k \xleftarrow{R} \mathcal{K}^s(\eta)$;
- A_1 is provided access to an oracle $\mathcal{E}^s(\cdot, k)$, that is, A_1 may submit messages m to the oracle and receives in return corresponding encryptions $\mathcal{E}^s(m, k)$;
- then A_1 outputs a challenge message $m^* \in [\![\tau]\!]_\eta$ together with some state information st;
- a bit $b \xleftarrow{R} \{0,1\}$ is selected at random; if $b = 0$, we let c be a (tagged) encryption of m^* under k, that is, $c \xleftarrow{R}$ "*SCipher*"$\|\tau\|\mathcal{E}^s(m^*, k)$; otherwise, we let c be a (tagged) encryption of a random element of τ under a random key, that is, $c \xleftarrow{R} [\![\mathit{SCipher}[\tau]]\!]_\eta$;
- A_2 is given c and st, and outputs a bit b'.

The adversary A is successful if $b' = b$. The advantage of A is defined by $\mathrm{Adv}^{\tau}_{\Pi^s, A}(\eta) = \Pr[A \text{ is successful}] - \frac{1}{2}$. We say that Π^s is T_{senc}-*secure* if for all PPTIME adversaries A and all $\tau \in T_{\mathsf{senc}}$, the function $\mathrm{Adv}^{\tau}_{\Pi^s, A}(\cdot)$ is negligible.

Definition 2. Let $\Pi^p = (\mathcal{K}^p, \mathcal{E}^p, \mathcal{D}^p)$ be an asymmetric encryption scheme. For each η and type $\tau \in T_{\mathsf{penc}}$, we consider the following experiment, with a two-stage PPTIME adversary $A = (A_1, A_2)$:

- a pair of encryption/decryption keys (pk, sk) is generated via $(pk, sk) \xleftarrow{R} \mathcal{K}^p(\eta)$, and A_1 is given pk;
- A_1 outputs a challenge message $m^* \in [\![\tau]\!]_\eta$ together with some state information st;
- a bit $b \xleftarrow{R} \{0,1\}$ is selected at random; if $b = 0$, we let c be a (tagged) encryption of m^* under pk, that is, $c \xleftarrow{R}$ "*ACipher*"$\|\tau\|\mathcal{E}^p(m^*, pk)$; otherwise, we let c be a (tagged) encryption of a random element of τ under a random public key, that is, $c \xleftarrow{R} [\![\mathit{ACipher}[\tau]]\!]_\eta$;
- A_2 is given c and st, and outputs a bit b'.

The adversary $A = (A_1, A_2)$ is successful if $b' = b$. The advantage of A is defined by $\mathrm{Adv}^{\tau}_{\Pi^p, A}(\eta) = \Pr[A \text{ is successful}] - \frac{1}{2}$. We say that Π^p is T_{penc}-*secure* if for all PPTIME adversaries A and all $\tau \in T_{\mathsf{penc}}$, the function $\mathrm{Adv}^{\tau}_{\Pi^p, A}(\cdot)$ is negligible.

Our notion of security for encryption schemes that use data values (such as passwords) as keys is less standard—and there is not yet a standard notion in the area:

Definition 3. Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a symmetric, deterministic, type-preserving encryption scheme such that the set of keys is $\{0,1\}^{\alpha_1(\eta)}$ for each η.

1. **Real-or-Random security (T_{enc}-RoR):** For each η and type $\tau \in T_{\mathsf{enc}}$, we consider the following experiment, with a two-stage PPTIME adversary $A = (A_1, A_2)$:
   - a key k is generated via $k \xleftarrow{R} \mathcal{K}(\eta)$;
   - A_1 is provided access to an oracle $\mathcal{E}(\cdot, k)$, that is, A_1 may submit (tagged) messages m to the oracle and receives in return corresponding (tagged) encryptions $\mathcal{E}(m, k)$;
   - then A_1 submits a challenge message $m^* \in [\![\tau]\!]_\eta$ and some state information st;
   - a bit $b \xleftarrow{R} \{0,1\}$ is selected at random; if $b = 0$, we let c be the (tagged) encryption of m^* under k, that is, $c = \mathcal{E}(m^*, k)$; otherwise, we let $c \xleftarrow{R} [\![\tau]\!]_\eta$ be a random element from $[\![\tau]\!]_\eta$;
   - A_2 is given c and st, and outputs a bit b'.

The adversary A is successful if $b' = b$ *and* the challenge message m^* is different from all the messages m submitted by A to the encryption oracle. The advantage of A is $\mathrm{Adv}^{\tau}_{\mathrm{RoR},\Pi,A}(\eta) = \Pr[A \text{ is successful}] - \frac{1}{2}$. We say that Π is T_{enc}-*RoR secure* if for all PPTIME adversaries A and all $\tau \in T_{\mathsf{enc}}$, the function $\mathrm{Adv}^{\tau}_{\mathrm{RoR},\Pi,A}(\cdot)$ is negligible.

2. **Encryption under passwords or other data values (T_{enc}-Pwd):** For each η and type $\tau \in T_{\mathsf{enc}}$, we consider the following experiment, with a two-stage PPTIME adversary $A = (A_1, A_2)$:

   - A_1 outputs a key $k \in \{0,1\}^{\alpha_1(\eta)}$ and some state information st;
   - a bit $b \xleftarrow{R} \{0,1\}$ is selected at random; if $b = 0$, we let c be the (tagged) encryption of some random element under k, that is, $m \xleftarrow{R} [\![\tau]\!]_\eta$ and $c = \mathcal{E}(m,k)$; otherwise, we let $c \xleftarrow{R} [\![\tau]\!]_\eta$ be a random element from $[\![\tau]\!]_\eta$;
   - A_2 is given c and st, and outputs a bit b'.

   The adversary A is successful if $b' = b$. The advantage of A is defined by $\mathrm{Adv}^{\tau}_{\mathrm{Pwd},\Pi,A}(\eta) = \Pr[A \text{ is successful}] - \frac{1}{2}$. We say that Π is a T_{enc}-*Pwd secure* if for all PPTIME adversaries A and all $\tau \in T_{\mathsf{enc}}$, the function $\mathrm{Adv}^{\tau}_{\mathrm{Pwd},\Pi,A}(\cdot)$ is negligible.

Finally, Π is T_{enc}-*secure* if it is both T_{enc}-RoR and T_{enc}-Pwd secure.

Condition 1 (T_{enc}-RoR security) is a variant of IND-P1-C0 security [33, 9]. We require it because we allow enc to be used as a first-class encryption algorithm, that is, with strong keys (not just passwords). Without this condition, our main result remains true on frames which use only constants as keys for enc (much as in [6]). Condition 2 (T_{enc}-Pwd security) addresses the security of passwords (or other data) when used as keys. Intuitively, it states that the encryption of a random value must be distributed like the value. A related previous condition [6] allows a possibly different distribution for the encryptions of random values and the values themselves. This difference is mostly due to the fact that we authorize multiple layers of encryptions with passwords (see example (4)).

Finally, an implementation with (Π^s, Π^p, Π) is $(T_{\mathsf{senc}}, T_{\mathsf{penc}}, T_{\mathsf{enc}})$-*secure* (or simply *secure*) if the three schemes Π^s, Π^p, and Π are, respectively, T_{senc}-secure, T_{penc}-secure, and T_{enc}-secure.

3.3 Example Implementation

In this section we outline a $(T_{\mathsf{senc}}, T_{\mathsf{penc}}, T_{\mathsf{enc}})$-*secure* implementation, based on standard cryptographic tools. Importantly, all encryption functions that we define map equal-length plaintexts to equal-length ciphertexts. It follows that, for each type τ, the set $[\![\tau]\!]$ contains bit-strings of equal length. We set $\alpha_1(\eta) = 6\eta$. We leave α_2 unspecified since it is determined by the specifics of our encryption functions.

3.3.1 Symmetric encryption

An important component of our construction is a symmetric encryption scheme that hides all information on the encryption key, all information about the plaintext (except possibly its length), and for which the ciphertexts of any message look uniformly random. We use a formalization of this requirement that appears in prior work [11]. For a symmetric encryption scheme $\Pi^s = (\mathcal{K}^s, \mathcal{E}^s, \mathcal{D}^s)$ and an adversary A, we consider the function:

$$\begin{aligned}\mathrm{Adv}^{\mathrm{rand}}_{A,\Pi^s}(\eta) \quad = \quad & \Pr[k \xleftarrow{R} \mathcal{K}^s(\eta) \ : \ A^{\mathcal{E}^s(\cdot,k)}(\eta) = 1] - \\ & \Pr[k \xleftarrow{R} \mathcal{K}^s(\eta) \ : \ A^{\$^{|\mathcal{E}(\cdot,k)|}}(\eta) = 1]\end{aligned}$$

where by $\$^{|\mathcal{E}(\cdot,k)|}$ we denote an oracle that on input m computes an encryption $c \xleftarrow{R} \mathcal{E}^s(m,k)$ and returns a random string of length $|c|$. A scheme is deemed secure if the function $\mathrm{Adv}^{\mathrm{rand}}_{A,\Pi^s}(\eta)$ is negligible for all

PPTIME adversaries A. Bellare et al. prove that the CBC encryption mode of secure block ciphers achieves this notion of security [11].

Using that for all $\tau \in T_{\mathsf{senc}}$ the set $[\![\tau]\!]$ contains only bit-strings of equal length, it is easy to show that a scheme secure according to $\mathrm{Adv}^{\mathrm{rand}}_{\Pi^s,A}$ is also T_{senc}-secure.

3.3.2 Asymmetric encryption

In the construction of our asymmetric encryption scheme we use a sequence of groups $\mathcal{G} = (\mathcal{G}_\eta)_\eta$ for which the Decisional Diffie-Hellman assumption holds. Moreover, we make the standard assumptions that each group comes with a generator. We omit the security parameter, and write generically g for each of the generators. Finally, we assume that deciding group membership is easy and that for each security parameter η, group elements in $\mathcal{G}_\eta$ have bit representations of equal length. The asymmetric encryption scheme that we construct is a hybrid encryption scheme between the El-Gamal encryption scheme based on $\mathcal{G}$ and the symmetric encryption scheme Π^s described above. Our construction assumes that the keys of Π^s for security parameter η are elements of the group $\mathcal{G}_\eta$. We define the asymmetric encryption scheme $\Pi^p = (\mathcal{K}^p, \mathcal{E}^p, \mathcal{D}^p)$ as follows.

- For security parameter η, the key generation algorithm selects a random element x of $\{0, 1, \ldots, |\mathcal{G}_\eta|-1\}$; the public key is set to g^x and its corresponding private key is set to x.
- An encryption of message m under public key $X = g^x$ is computed as follows. First, a key for the symmetric encryption scheme Π^s is generated via $k \xleftarrow{R} \mathcal{K}^s(\eta)$ and the key is used for obtaining an encryption c of m, that is, $c \xleftarrow{R} \mathcal{E}^s(m, k)$. The final ciphertext is obtained by concatenating an El-Gamal encryption of k with c (formally, $\mathcal{E}^p(m, X) = (g^r, X^r \cdot k, c)$ where $r \xleftarrow{R} \{0, 1, \ldots, |\mathcal{G}_\eta| - 1\}$).

The choice of the El-Gamal encryption for the asymmetric part of our hybrid construction is not arbitrary. It is known that El-Gamal ciphertexts hide all information on the encryption key [10], and the property immediately extends to the hybrid construction. Also, it is a well-known result that the resulting hybrid encryption scheme is semantically secure [13] and for each $\tau \in T_{\mathsf{penc}}$ the set of plaintexts have equal length, it immediately follows that the resulting hybrid encryption scheme is T_{penc}-secure.

3.3.3 Encryption under a data value

We assume the existence of two families of pseudorandom functions F and G.

Family $F = (F_\eta)_\eta$ is such that, for each security parameter, F_η is a keyed permutation on $\mathcal{G}_\eta$, that is, $F_\eta : \{0,1\}^\eta \times \mathcal{G}_\eta \to \mathcal{G}_\eta$, where $\mathcal{G} = (\mathcal{G}_\eta)_\eta$ is the sequence of groups used in the construction of Π^p. Recall that the pseudorandomness assumption on the function ensemble F essentially requires that an adversary with access to a function cannot tell if the function is $F_k(\cdot)$ for a randomly chosen key $k \in Keys$ or it is a truly random permutation on the domain of F. Formally, for any PPTIME adversary A with oracle access to a function, the quantity

$$\begin{aligned}\mathrm{Adv}^{\mathrm{prp}}_{A,F}(\eta) \quad = \quad & \Pr[k \xleftarrow{R} Keys \ : \ A^{F_\eta(k,\cdot)}(\eta) = 1] - \\ & \Pr[f \xleftarrow{R} \mathrm{Perm}(\mathcal{G}_\eta) \ : \ A^{f(\cdot)}(\eta) = 1]\end{aligned}$$

is negligible, where $\mathrm{Perm}(\mathcal{G}_\eta)$ is the set of all permutations on $\mathcal{G}_\eta$.

Family $G = (G_\eta)_\eta$ is such that, for each η, $G_\eta : \{0,1\}^\eta \times \{0,1\}^* \to \{0,1\}^*$ is a keyed length-preserving permutation on $\{0,1\}^*$. Black and Rogaway [16] and Bellare and Rogaway[14] study the existence of families of function as above.

Our construction combines five encryption algorithms, one for each of the sets $S_1 = [\![SKey]\!]$, $S_2 = [\![EKey]\!]$, $S_3 = [\![Data]\!]$, $S_4 = \bigcup_\tau [\![SCipher[\tau]]\!]$, and $S_5 = \bigcup_\tau [\![ACipher[\tau]]\!]$. We define encryption schemes $\Pi_i = (\mathcal{K}_i, \mathcal{E}_i, \mathcal{D}_i)$ for $i = 1..5$. For each $i \in \{1, 2, 3, 4\}$, the key generation algorithm $\mathcal{K}_i$ selects a random string in $\{0,1\}^\eta$. Algorithm $\mathcal{E}_5$ selects two such random strings. We define the encryption functions as follows:

- $\mathcal{E}_1 : [\![Data]\!] \times \{0,1\}^\eta \to [\![Data]\!]$ is defined by

$$\mathcal{E}_1(\text{“}Data\text{”}||s,k) = \text{“}Data\text{”}||G_\eta(k,s)$$

- $\mathcal{E}_2 : [\![SKey]\!] \times \{0,1\}^\eta \to [\![SKey]\!]$ is defined by

$$\mathcal{E}_2(\text{“}SKey\text{”}||s,k) = \text{“}SKey\text{”}||G_\eta(k,s)$$

- $\mathcal{E}_3 : [\![EKey]\!] \times \{0,1\}^\eta \to [\![EKey]\!]$ is defined by

$$\mathcal{E}_3(\text{“}EKey\text{”}||X,k) = \text{“}EKey\text{”}||F_\eta(k,X)$$

- $\mathcal{E}_4 : [\![SCipher[\tau]]\!] \times \{0,1\}^\eta \to [\![SCipher[\tau]]\!]$ is defined by

$$\mathcal{E}_4(\text{“}SCipher\text{”}||\tau||c,k) = \text{“}SCipher\text{”}||\tau||G_\eta(k,c)$$

- $\mathcal{E}_5 : [\![ACipher[\tau]]\!] \times (\{0,1\}^\eta \times \{0,1\}^\eta) \to [\![ACipher[\tau]]\!]$ is defined by

$$\mathcal{E}_5(\text{“}ACipher\text{”}||\tau||(R,Y,c),(k,k')) = \text{“}ACipher\text{”}||\tau||(F_\eta(k,R),F_\eta(k,Y),G_\eta(k',c))$$

The corresponding decryption functions are obvious.

Finally we combine the functions above into a symmetric, deterministic, type-preserving encryption scheme $\Pi = (\mathcal{K},\mathcal{E},\mathcal{D})$:

- The key generation algorithm selects one random key for each of the encryption functions defined above: $\mathcal{K}(\eta) = \text{“}Data\text{”}||(k_1,k_2,k_3,k_4,k_5)$ with $k_i \xleftarrow{R} \mathcal{K}_i(\eta)$. Here we write $(k_1,k_2,\ldots,k_5)$ for the string of length $5\cdot\eta$ obtained by concatenating the strings k_1,$k_2\ldots,k_5$.
- The encryption algorithm essentially encrypts elements of S_i using encryption function $\mathcal{E}_i$. Formally, $\mathcal{E}(m,\text{“}Data\text{”}||(k_1,k_2,k_3,k_4,k_5,k_6))$ outputs $\mathcal{E}_i(m,k_i)$ if $m \in S_i$.
- The decryption function examines the tag of a ciphertext, then proceeds accordingly in the obvious way.

A basic observation about our construction is that the encryption functions $\mathcal{E}_i$ are pseudorandom bijective functions on their respective domains and their domains are disjoint. A standard hybrid argument shows that each encryption function $\mathcal{E}_i$ satisfies T_{enc}-RoR where the queries to the oracle of A are mandated to be elements of S_i. Since in the construction of $\mathcal{E}$ we use independent keys for each of the five algorithms that define it, it follows by a simulation argument that $\mathcal{E}$ is T_{enc}-RoR secure.

To argue that $\mathcal{E}$ is also T_{enc}-Pwd secure, we show that each encryption function $\mathcal{E}_i$ satisfies this condition. T_{enc}-Pwd security follows straightforwardly.

Function $\mathcal{E}_1$ clearly satisfies this property: for any valid k the function $\mathcal{E}(\cdot,k)$ is a permutation on $[\![Data]\!]$ and therefore distributions $\mathcal{E}_1(\xleftarrow{R} [\![Data]\!],k)$ and $(\xleftarrow{R} [\![Data]\!])$ are identical. The same argument applies for functions $\mathcal{E}_2$ and $\mathcal{E}_3$. Concerning $\mathcal{E}_4$, we use the fact that the distribution $(\xleftarrow{R} SCipher[\tau])$, once the tags are removed, is indistinguishable from the uniform distribution on bit-strings of appropriate length [11]; therefore, for an arbitrary k, the distributions $\mathcal{E}_4(\xleftarrow{R} [\![Data]\!],k)$ and $(\xleftarrow{R} [\![Data]\!])$ are indistinguishable. The case of $\mathcal{E}_5$ follows from a similar argument: using the definition of the El-Gamal encryption and the property of symmetric ciphers mentioned above, it follows that the distribution $(\xleftarrow{R} ACipher[\tau])$ (omitting the tags) is indistinguishable from the uniform distribution over $\mathcal{G}_\eta \times \mathcal{G}_\eta \times \{0,1\}^\beta$ where β is the size of elements in $[\![\tau]\!]_\eta$.

4 Soundness of Static Equivalence

In this section we present our main soundness result. We first identify some important conditions under which our main result holds, state the theorem, and then give its proof.

4.1 Main Theorem

As usual (following [5]), our soundness result requires a hypothesis that excludes encryption cycles, and also some other well-formedness conditions. A *key position* in an expression is a position that corresponds to the argument U of a subterm of the form $\mathsf{pub}(U)$, or to the second argument V of a subterm $\mathsf{enc}_\tau(U,V)$, $\mathsf{penc}_\tau(U,V,W)$, or $\mathsf{senc}_\tau(U,V,W)$. An *encryption cycle* of a frame φ is a sequence of names k_0, k_1, …, k_n of sort *Data*, *DKey*, and *SKey* such that $k_n = k_0$ and

> for each $0 \leq i \leq n-1$, there exists a subterm of φ of the form $\mathsf{enc}_\tau(U,V)$, $\mathsf{penc}_\tau(U,V,W)$, or $\mathsf{senc}_\tau(U,V,W)$ such that k_i is a subterm of U *not* in key position and k_{i+1} is a subterm of V.

For instance, the frame $\varphi_1 = \{x = \{sk_1\}^{r_1}_{k_2},\ y = \{k_2\}^{r_2}_{\mathsf{pub}(sk_1)}\}$ has an encryption cycle, while $\varphi_2 = \{x = \{\mathsf{pub}(sk_1)\}^{r_1}_{k_2},\ y = \{k_2\}^{r_2}_{\mathsf{pub}(sk_1)}\}$ does not. (Note that sk_1 occurs in φ_2 only in key positions.)

A frame φ is *well-formed* if it satisfies the following conditions:

(i) φ is $\mathcal{R}$-reduced, that is, in normal form with respect to the rewriting system $\mathcal{R}$;

(ii) φ does not contain the symbols dec, pdec, sdec, pdec_success, sdec_success, fst, and snd;

(iii) terms in key position in φ are of the following forms, depending on their sort:

- sorts *DKey* and *SKey*: names,
- sort *EKey*: names and terms of the form $\mathsf{pub}(a)$,
- sort *Data*: names and constants;

(iv) terms of type *Coins* may only be names, and appear as the third argument of an encryption; moreover, if such a name appears twice in φ then the encryption terms in which it appears are identical;

(v) φ has no encryption cycles;

(vi) for every subterm of φ of the form $\mathsf{enc}(T,k)$ where k is a name, T contains none of the constants w, c_0, c_1, …, and T has no subterm of the form $\mathsf{enc}(S,0)$ or $\mathsf{enc}(S,1)$.

Condition (ii) indicates that we focus on the indistinguishability of expressions built from constructors; it does not preclude using other functions in the observations that may distinguish frames. Condition (iii) says that keys are atomic terms for symmetric encryptions, and terms of the form $\mathsf{pub}(a)$ for public-key encryptions. Similarly, condition (iv) says that coins are names and are used only for encryptions, with different coins in each encryption. Condition (v) is the acyclicity requirement. Finally, condition (vi) restricts the occurrences of constants within plaintexts for deterministic encryption under strong keys (represented by names). For instance, this condition excludes the frame $\nu k.\{x_1 = \mathsf{enc}(\mathsf{c}_1,k),\ x_2 = \mathsf{enc}(\mathsf{c}_2,k)\}$, which is equivalent to $\nu a_1, a_2.\{x_1 = a_1,\ x_2 = a_2\}$ formally but not computationally if c_1 and c_2 happen to have the same bit-string implementations. More generally, when T_1 and T_2 are two terms such that $T_1 \neq_E T_2$, the encryptions $\mathsf{enc}(T_1,k)$ and $\mathsf{enc}(T_2,k)$ may behave like distinct fresh names formally but not computationally, unless the bit-string values of T_1 and T_2 collide with negligible probability.

We obtain:

Theorem 1 ($\approx_E$-soundness)**.** *Let φ_1 and φ_2 be two well-formed frames such that $\varphi_1 \approx_E \varphi_2$. In any secure implementation, $[\![\varphi_1]\!] \approx [\![\varphi_2]\!]$.*

An obvious question is whether the converse completeness result holds, in other words whether computational indistinguishability of well-formed frames implies their static equivalence. Although we do not have a definitive answer to this question, we believe that existing techniques for achieving completeness should apply in our setting (see the introduction).

4.2 Proof of the Main Theorem

The remainder of this section is devoted to the proof of Theorem 1. The proof is based on a set of transformation rules on equations between frames. The rules preserve both the symbolic and the computational meaning of equations (in a sense made precise below). Interestingly, the rules provide a decision procedure for static equivalence between well-formed frames.

4.2.1 Overview of the proof

Let us call an *equation* any expression ε of the form $\bot$, $\top$, or $\varphi_1 \approx^? \varphi_2$ where φ_1 and φ_2 are two frames.

In the sequel, we consider an algorithm defined by a set of transformation rules over equations. Let $\Longrightarrow$ be the corresponding transformation relation. Starting from an initial equation $\varepsilon = (\varphi_1 \approx^? \varphi_2)$ between well-formed frames, we expect:

- Each run of the algorithm should yield a finite derivation $\varepsilon \Longrightarrow \varepsilon_1 \Longrightarrow \ldots \Longrightarrow \bot$ in case of a *negative answer*, or a finite derivation $\varepsilon \Longrightarrow \varepsilon_1 \Longrightarrow \ldots \Longrightarrow \top$ for a *positive answer*. This property is called *progress and termination*.
- The rules should be *effective*, that is, applicable by a Turing machine.
- The rules should be *formally sound and complete*: the answer is positive iff the two frames are statically equivalent, that is, $\varphi_1 \approx_E \varphi_2$.
- The rules should be *computationally sound*: if the answer is positive then the two frames are indistinguishable, that is, $[\![\varphi_1]\!] \approx [\![\varphi_2]\!]$.

These properties clearly entail Theorem 1: if $\varphi_1 \approx_E \varphi_2$ and the two frames φ_1 and φ_2 are well-formed, then by progress, termination, and formal completeness, the algorithm must return a positive answer; thus, by computational soundness, we have $[\![\varphi_1]\!] \approx [\![\varphi_2]\!]$.

Furthermore, since the transformation rules are also effective and formally sound, we obtain a decision procedure for static equivalence between well-formed frames.

In the remainder of this section we describe the transformation rules, and show how to apply them in several illustrative examples. In Appendix A we establish that $\Longrightarrow$ satisfies the properties listed above.

4.2.2 Notations and definitions

We introduce some useful notations and auxiliary definitions.

A closed term T is *deducible* [2] from a frame φ (written $\varphi \vdash_E T$) if there exists a term M with $\mathrm{var}(M) \subseteq \mathrm{dom}(\varphi)$ and $\mathrm{names}(M) \cap \mathrm{names}(\varphi, T) = \emptyset$ such that $M\varphi =_E T$. Note that the names of T, just as those of φ, are not allowed to occur in M. (In this respect, the definition is slightly different from that of our original presentation [1].)

Given a term U, n signs $\ell_1 \ldots \ell_n \in \{+1, -1\}$ ($n \geq 0$), and n terms $V_1 \ldots V_n$ of sort *Data*, we define the expression $\{U\}_{V_1^{\ell_1} \cdot \ldots \cdot V_n^{\ell_n}}$ recursively by

$$\begin{aligned} \{U\}_{\Lambda} &= U \\ \{U\}_{\vec{W} \cdot V^{+1}} &= \mathsf{enc}(\{U\}_{\vec{W}}, V) \\ \{U\}_{\vec{W} \cdot V^{-1}} &= \mathsf{dec}(\{U\}_{\vec{W}}, V) \end{aligned}$$

where Λ represents the empty word, and $\cdot$ is an associative symbol. Below, we identify V and V^{+1}. If $\vec{V}$ is the expression $\vec{V} = V_1^{\ell_1} \cdot \ldots \cdot V_n^{\ell_n}$, we write $\vec{V}^{-1} = V_n^{-\ell_n} \cdot \ldots \cdot V_1^{-\ell_1}$ for the *inverse* of $\vec{V}$. We extend notations from terms to such elements as expected, so for instance $\vec{V}\sigma = (V_1\sigma)^{\ell_1} \cdot \ldots \cdot (V_n\sigma)^{\ell_n}$ and $\mathrm{var}(\vec{V}) = \bigcup_i \mathrm{var}(V_i)$.

Let $\mathcal{W} = \bigcup_\tau \{w_1^\tau, w_2^\tau \ldots\}$ be a dedicated set of variables, called *parameters*, where each w_i^τ has sort τ. A *plain n-ary context* is a term C such that $\mathrm{names}(C) = \emptyset$ and $\mathrm{var}(C)$ is included in $\{w_1^{\tau_1}, \ldots, w_n^{\tau_n}\}$ for

some $\tau_1, \ldots, \tau_n$. If $T_1, \ldots, T_n$ are terms of sorts $\tau_1 \ldots \tau_n$, respectively, we write $C[T_1, \ldots, T_n]$ for $C\{w_1^{\tau_1} \mapsto T_1, \ldots, w_n^{\tau_n} \mapsto T_n\}$. We extend the vectorial notation above to contexts:

$$\begin{aligned}\vec{C}[T_1 \ldots T_m] &= (C_1^{\ell_1} \cdot \ldots \cdot C_n^{\ell_n})[T_1 \ldots T_m] \\ &= (C_1[T_1 \ldots T_m])^{\ell_1} \cdot \ldots \cdot (C_n[T_1 \ldots T_m])^{\ell_n}\end{aligned}$$

We write $\sigma_1\sigma_2$ for the usual *composition* of substitutions: for all x, $x\sigma_1\sigma_2 = (x\sigma_1)\sigma_2$. The application of a substitution σ to a frame $\varphi = \{x_1 = T_1, \ldots, x_n = T_n\}$ is defined by $\varphi\sigma = \{x_1 = T_1\sigma, \ldots, x_n = T_n\sigma\}$. (Note that φ is not identified to a substitution in this respect.)

We write $\varphi|_D$ for the *restriction* of a frame to a domain D, and $\varphi_1 \uplus \varphi_2$ for the *union* of two frames φ_1, φ_2 with disjoint domains. The *image* of a frame $\varphi = \{x_1 = T_1, \ldots, x_n = T_n\}$ is defined as $\mathrm{im}(\varphi) = \{T_1, \ldots, T_n\}$.

A term is *maximal* in a set of terms $\mathcal{S}$ if it is the subterm of no other term in $\mathcal{S}$.

We also write $\{T \mapsto T'\}$ for the (syntactic, breadth-first) replacement of T by T'. Namely, if $U = f(U_1, \ldots, U_n)$ is a term (and similarly for frames), the expression $U\{T \mapsto T'\}$ equals T' if $U = T$, and $f(U_1\{T \mapsto T'\}, \ldots, U_n\{T \mapsto T'\})$ otherwise.

Finally, we write $T\downarrow_{\mathcal{R}}$ for the $\mathcal{R}$-normal form of a term T. (Recall that $\mathcal{R}$ is the rewriting system obtained by orienting the equations under consideration.)

4.2.3 Decision procedure

We describe the transformation rules used to simplify equations step by step. Those are presented in Tables 1 and 2. For the purpose of defining the procedure, we see $\approx^?$ as a commutative symbol.

In every rule, the word "fresh" refers to a variable or a name of the appropriate sort, not occurring in the (same side of the) source equation. Below, we make precise the notion of standard frames and the relation $\approx_E^{\mathrm{std}}$ mentioned in the last rule of Table 2.

The procedure itself consists in applying the rules in the following order:

1. a maximal sequence of rule **Undecipherable Encryption**;
2. a maximal sequence of rule **Split Names**;
3. a maximal sequence of **Analysis** rules, that is either **Pair Analysis**, **Redundancy Analysis- 1,2**, or **Encryption Analysis-1,2**;
4. a maximal sequence of rule **Standardize**; and finally
5. a maximal sequence of rule **Solve**.

Rule **Undecipherable Encryption** substitutes a fresh name successively for each encryption term T in which a non-deducible key is used. Note that all the occurrences of T are replaced at once. (Since the frames are well-formed, and in particular $\mathcal{R}$-reduced, a syntactic replacement is sufficient in this respect.) Rule **Split Names** is a technical transformation that suppresses names of sort $Pair[\tau_1, \tau_2]$. (The same remark applies.) Rule **Pair Analysis** splits pairs of terms into separate components in each frame. The two rules **Redundancy Analysis-1,2** detect when a component of a frame is deducible from the other components. If the corresponding component is deducible in the same way in the other frame, then both components are removed (rule **1**), otherwise this gives a counter-example to static equivalence (rule **2**). The two rules **Encryption Analysis-1,2** are meant to detect the success of a symmetric or asymmetric decryption over a deducible subcomponent, possibly under several layers of deterministic encryption. Intuitively, if the same relation holds on both frames, then the components of interest are replaced on both sides by the underlying plaintexts; otherwise, we deduce non-equivalence between the two frames. In **Redundancy Analysis-1,2** and **Encryption Analysis-1,2**, the phrase "as ε" in the hypothesis simply gives the name ε to the equation in the hypothesis so that it can be used in side conditions. The last two rules **Standardize** and **Solve** conclude the procedure by relying on the notion of *standard frames*, defined as follows:

$$\frac{\varphi_1 \approx^? \varphi_2}{\varphi_1\{T \mapsto n\} \approx^? \varphi_2}$$

Undecipherable Encryption
where φ_1 is well-formed; T is a maximal subterm of φ_1 of the form $\mathsf{penc}(U, k', r)$ for a given name k' (respectively $\mathsf{pub}(k)$, $\mathsf{senc}(U, k, r)$, or $\mathsf{enc}(U, k)$, for a given name k whose every occurrence in φ_1 is in key position); and n is a fresh name.

$$\frac{\varphi_1 \approx^? \varphi_2}{\varphi_1\{n \mapsto \mathsf{pair}(n_1, n_2)\} \approx^? \varphi_2}$$

Split Names
where the sort of the name n is $Pair[\tau_1, \tau_2]$; and n_1, n_2 are fresh names.

$$\frac{\varphi_1 \uplus \{x = \mathsf{pair}(U_1, V_1)\} \approx^? \varphi_2 \uplus \{x = \mathsf{pair}(U_2, V_2)\}}{\begin{array}{c}\varphi_1 \uplus \{y = U_1,\, z = V_1\} \\ \approx^? \varphi_2 \uplus \{y = U_2,\, z = V_2)\}\end{array}}$$

Pair Analysis
where y, z are fresh variables.

$$\frac{\varphi_1 \uplus \{x = U\} \approx^? \varphi_2 \uplus \{x = W\} \quad \text{as } \varepsilon}{\varphi_1 \approx^? \varphi_2}$$

Redundancy Analysis-1
if there exist M such that
$$\begin{cases} \mathrm{var}(M) \subseteq \mathrm{dom}(\varphi_1) \\ \mathrm{names}(M) \cap \mathrm{names}(\varepsilon) = \emptyset \\ M\varphi_1 =_E U \\ W =_E M\varphi_2 \end{cases}$$

$$\frac{\varphi_1 \uplus \{x = U\} \approx^? \varphi_2 \uplus \{x = W\} \quad \text{as } \varepsilon}{\bot}$$

Redundancy Analysis-2
if there exist M such that
$$\begin{cases} \mathrm{var}(M) \subseteq \mathrm{dom}(\varphi_1) \\ \mathrm{names}(M) \cap \mathrm{names}(\varepsilon) = \emptyset \\ M\varphi_1 =_E U \\ W \neq_E M\varphi_2 \end{cases}$$

Table 1: Transformation rules

$$\frac{\varphi_1 \uplus \{x = \{U\}_{\vec{V}}\} \approx^? \varphi_2 \uplus \{x = W\} \quad \text{as } \varepsilon}{\varphi_1 \uplus \{y = S_1\} \approx^? \varphi_2 \uplus \{y = S_2\}}$$

Encryption Analysis-1
$U = \mathsf{penc}(S_1, \mathsf{pub}(T_1), r_1)$ (or respectively $U = \mathsf{senc}(S_1, T_1, r_1)$);
r_1 does not appear elsewhere in the same side of ε;
there exist $\vec{M}$ and N such that

$$\begin{cases} \mathrm{var}(\vec{M}, N) \subseteq \mathrm{dom}(\varphi_1) \\ \mathrm{names}(\vec{M}, N) \cap \mathrm{names}(\varepsilon) = \emptyset \\ \vec{M}\varphi_1 =_E \vec{V} \\ N\varphi_1 =_E T_1 \\ \{W\}_{\vec{M}^{-1}\varphi_2}\downarrow_{\mathcal{R}} \\ \quad = \mathsf{penc}(S_2, \mathsf{pub}(N\varphi_2)\downarrow_{\mathcal{R}}, r_2) \\ \text{(respectively} \\ \quad = \mathsf{senc}(S_2, N\varphi_2\downarrow_{\mathcal{R}}, r_2)\ ) \end{cases}$$

r_2 does not appear elsewhere in the same side of ε; y is a fresh variable.

$$\frac{\varphi_1 \uplus \{x = \{U\}_{\vec{V}}\} \approx^? \varphi_2 \uplus \{x = W\} \quad \text{as } \varepsilon}{\bot}$$

Encryption Analysis-2
$U = \mathsf{penc}(S_1, \mathsf{pub}(T_1), r_1)$ (or respectively $U = \mathsf{senc}(S_1, T_1, r_1)$);
there exist $\vec{M}$ and N such that

$$\begin{cases} \mathrm{var}(\vec{M}, N) \subseteq \mathrm{dom}(\varphi_1) \\ \mathrm{names}(\vec{M}, N) \cap \mathrm{names}(\varepsilon) = \emptyset \\ \vec{M}\varphi_1 =_E \vec{V} \\ N\varphi_1 =_E T_1 \\ \mathsf{pdec\_success}(\{W\}_{\vec{M}^{-1}}, N)\varphi_2\downarrow_{\mathcal{R}} \\ \quad \neq 1 \\ \text{(respectively} \\ \mathsf{sdec\_success}(\{W\}_{\vec{M}^{-1}}, N)\varphi_2\downarrow_{\mathcal{R}} \\ \quad \neq 1\ ) \end{cases}$$

$$\frac{\varphi_1 \uplus \{x = \{a\}_{\vec{C}[a_1,\ldots,a_n]}\} \approx^? \varphi_2}{\varphi_1\{a \mapsto \{a'\}_{\vec{C}[a_1,\ldots,a_n]^{-1}}\}\downarrow_{\mathcal{R}} \uplus \{x = a'\} \approx^? \varphi_2}$$

Standardize
where $a_1, \ldots, a_n \in \mathrm{im}(\varphi_1)$; $\vec{C} \neq \Lambda$; $a \notin \mathrm{im}(\varphi_1)$; a' is a fresh name.

$$\frac{\varphi_1 \approx^? \varphi_2}{\varepsilon'}$$

Solve
where φ_1 and φ_2 are standard; if $\varphi_1 \approx_E^{\mathrm{std}} \varphi_2$ then $\varepsilon' = \top$, otherwise $\varepsilon' = \bot$.

Table 2: Transformation rules (cont'd)

Definition 4. A frame φ is *standard* if $\mathrm{names}(\varphi) \subseteq \mathrm{im}(\varphi)$, that is, φ is of the form

$$\varphi = \{x_1 = a_1, \ldots, x_m = a_m, y_1 = C_1[a_1 \ldots a_m], \ldots, y_n = C_n[a_1 \ldots a_m]\}$$

where the names $a_1, \ldots a_m$ are pairwise distinct.

Given two standard frames

$$\varphi_i = \{x_1^i = a_1^i, \ldots, x_{m_i}^i = a_{m_i}^i, y_1^i = C_1^i[a_1^i \ldots a_{m_i}^i], \ldots, y_{n_i}^i = C_{n_i}^i[a_1^i \ldots a_{m_i}^i]\},$$

($i \in \{1, 2\}$), we write $\varphi_1 \approx_E^{\mathrm{std}} \varphi_2$ when

$$\begin{aligned} \text{for all } j \in \{1 \ldots n_1\}, \quad y_j^1\,\varphi_2 &=_E C_j^1[x_1^1 \ldots x_{m_1}^1]\varphi_2, \\ \text{and for all } k \in \{1 \ldots n_2\}, \quad y_k^2\,\varphi_1 &=_E C_k^2[x_1^2 \ldots x_{m_2}^2]\varphi_1. \end{aligned}$$

Notably, the following proposition due to Baudet *et al.* [9] implies that the relations $\approx^{\mathrm{std}}$ and $\approx_E$ coincide on standard frames. We use $=^?$ as a commutative symbol to build (formal) equations between terms.

Proposition 2. *Let φ be a frame of the form*

$$\varphi = \{x_1 = C_1[a_1, \ldots, a_m], \ldots, x_n = C_n[a_1, \ldots, a_m]\}$$

where $a_1, \ldots, a_m$ are pairwise distinct names deducible from φ. For each $1 \leq i \leq m$, let M_i be a term such that $\mathrm{var}(M_i) \subseteq \mathrm{dom}(\varphi)$, $\mathrm{names}(M_i) \cap \mathrm{names}(\varphi)$, and $M_i\varphi =_E a_i$.

Then every equation $M =^? N$ which holds in φ, that is, satisfying

$$\mathrm{var}(M, N) \subseteq \mathrm{dom}(\varphi),\ \mathrm{names}(M, N) \cap \mathrm{names}(\varphi) = \emptyset \text{ and } M\varphi =_E N\varphi,$$

is a logical consequence of E augmented with the equations $x_j =^? C_j[M_1 \ldots M_m]$ for $1 \leq j \leq n$.

Here "logical consequence" refers to the usual first-order equational logics where the variables in $\mathrm{dom}(\varphi)$ are seen as (free) constants. The proof of this proposition relies on the stability of E under substitutions of names.

4.2.4 Examples

We illustrate the workings of the above procedure via a few examples.

Example 1. Recall the frame that corresponds to the EKE protocol (Section 2.3):

$$\begin{aligned} \varphi = \nu pk, k_1, n_A, n_B, r_1, r_2, r_3, r_4.\ &\{x_1 = \{pk\}_{\mathsf{w}_{AB}},\ x_2 = \{\{k_1\}_{pk}^{r_1}\}_{\mathsf{w}_{AB}},\ x_3 = \{n_A\}_{k_1}^{r_2}, \\ &x_4 = \{\mathsf{pair}(n_A,\, n_B)\}_{k_1}^{r_3},\ x_5 = \{n_B\}_{k_1}^{r_4}\ \} \end{aligned}$$

For $i \in \{0, 1\}$, let $\varphi_i = \varphi\{\mathsf{w}_{AB} \mapsto \mathsf{c}_i\}$. We check that $\varphi_0 \approx_E \varphi_1$.

Starting from the equation $\varepsilon_1 = (\varphi_0 \approx^? \varphi_1)$, we apply **Undecipherable Encryption** on $T = \{k_1\}_{pk}^{r_1}$ on both sides and obtain (omitting the binders):

$$\begin{aligned} \{x_1 = \{pk\}_{\mathsf{c}_0},\ x_2 = \{n_0\}_{\mathsf{c}_0},\ x_3 = \{n_A\}_{k_1}^{r_2}, &\\ x_4 = \{\mathsf{pair}(n_A,\, n_B)\}_{k_1}^{r_3},\ x_5 = \{n_B\}_{k_1}^{r_4}\ \} &\\ \approx^? \{x_1 = \{pk\}_{\mathsf{c}_1},\ x_2 = \{n_0'\}_{\mathsf{c}_1},\ x_3 = \{n_A\}_{k_1}^{r_2}, &\\ x_4 = \{\mathsf{pair}(n_A,\, n_B)\}_{k_1}^{r_3},\ x_5 = \{n_B\}_{k_1}^{r_4}\ \} & \end{aligned}$$

Then, by applying the same rule repeatedly on ciphertexts under k_1, we obtain:

$$\{x_1 = \{pk\}_{\mathsf{c}_0},\ x_2 = \{n_0\}_{\mathsf{c}_0},\ x_3 = n_1,\ x_4 = n_2,\ x_5 = n_3\ \} \\ \approx^? \{x_1 = \{pk\}_{\mathsf{c}_1},\ x_2 = \{n_0'\}_{\mathsf{c}_1},\ x_3 = n_1',\ x_4 = n_2',\ x_5 = n_3'\ \}$$

As none of the **Analysis** rules applies in this case, rule **Standardize** on pk, n_0, and n_0' yields:

$$\{x_1 = pk',\ x_2 = n_0'',\ x_3 = n_1,\ x_4 = n_2,\ x_5 = n_3\ \} \\ \approx^? \{x_1 = pk'',\ x_2 = n_0''',\ x_3 = n_1',\ x_4 = n_2',\ x_5 = n_3'\ \}$$

Rule **Solve** then concludes with a positive answer. In this simple example, the two final frames turn out to be equal modulo renaming. This equality might not always hold, as shown by the example below.

Example 2. This example concerns a variant of the EKE protocol where symmetric encryption is used instead of public-key encryption. Accordingly, the first message that A sends to B is $\{k_0\}_{\mathsf{w}_{AB}}$, and the message that B sends to A next is $\{\{k_1\}_{k_0}^{r_1}\}_{\mathsf{w}_{AB}}$. The reminder of the protocol remains unchanged. The frame associated to one session of the protocol is:

$$\varphi = \nu k_0, k_1, n_A, n_B, r_1, r_2, r_3, r_4. \\ \{x_1 = \{k_0\}_{\mathsf{w}_{AB}},\ x_2 = \{\{k_1\}_{k_0}^{r_1}\}_{\mathsf{w}_{AB}},\ x_3 = \{n_A\}_{k_1}^{r_2}, \\ x_4 = \{\mathsf{pair}(n_A,\ n_B)\}_{k_1}^{r_3},\ x_5 = \{n_B\}_{k_1}^{r_4}\ \}$$

For $i \in \{0, 1\}$, let $\varphi_i = \varphi\{\mathsf{w}_{AB} \mapsto \mathsf{c}_i\}$. Our decision procedure concludes that $\varphi_0 \not\approx_E \varphi_1$.

The frame is already simplified with respect to **Undecipherable Encryption** and **Split Names**. We apply **Encryption Analysis-2** on x_2 to obtain the answer $\bot$: indeed $x_2\varphi_0 = \{\mathsf{senc}(S_1, T_1, r_1)\}_{\mathsf{c}_0}$ where $T_1 =_E \mathsf{dec}(x_1, \mathsf{c}_0)\varphi_0$, but

$$\mathsf{sdec\_success}(\mathsf{dec}(x_2, \mathsf{c}_0), \mathsf{dec}(x_1, \mathsf{c}_0))\varphi_1 \neq_E 1$$

Example 3. Let ε_3 be the following equation:

$$\{x_1 = \{\{n\}_k\}_k,\ x_2 = \{n\}_k,\ x_3 = \{k\}_{\mathsf{c}_0},\ x_4 = \{k\}_{\mathsf{c}_1}\ \} \\ \approx^? \{x_1 = \{n\}_k,\ x_2 = n,\ x_3 = \{k\}_{\mathsf{c}_0},\ x_4 = \{k\}_{\mathsf{c}_1}\ \}$$

The first applicable rule is **Standardize**. Applying this rule on $x_2 = \{n\}_k$ in the left-hand side and then on $x_3 = \{k\}_{\mathsf{c}_0}$ on both sides leads to two frames equal modulo renaming:

$$\{x_1 = \{n'\}_{\mathsf{dec}(k',\mathsf{c}_0)},\ x_2 = n',\ x_3 = k',\ x_4 = \{\mathsf{dec}(k', \mathsf{c}_0)\}_{\mathsf{c}_1} \\ \approx^? \{x_1 = \{n\}_{\mathsf{dec}(k',\mathsf{c}_0)},\ x_2 = n,\ x_3 = k',\ x_4 = \{\mathsf{dec}(k', \mathsf{c}_0)\}_{\mathsf{c}_1}\ \}$$

that is (in vector notation):

$$\{x_1 = \{n'\}_{\{k'\}_{\mathsf{c}_0^{-1}}},\ x_2 = n',\ x_3 = k',\ x_4 = \{k'\}_{\mathsf{c}_0^{-1}.\mathsf{c}_1}\ \} \\ \approx^? \{x_1 = \{n\}_{\{k'\}_{\mathsf{c}_0^{-1}}},\ x_2 = n,\ x_3 = k',\ x_4 = \{k'\}_{\mathsf{c}_0^{-1}.\mathsf{c}_1}\ \}$$

But, choosing different applications of the rule, we may obtain as well:

$$\{x_1 = n',\ x_2 = \{n'\}_{(\{k'\}_{\mathsf{c}_0^{-1}})^{-1}},\ x_3 = k',\ x_4 = \{k'\}_{\mathsf{c}_0^{-1}.\mathsf{c}_1}\ \} \\ \approx^? \{x_1 = \{n\}_{\{k''\}_{\mathsf{c}_1^{-1}}},\ x_2 = n,\ x_3 = \{k''\}_{\mathsf{c}_1^{-1}.\mathsf{c}_0},\ x_4 = k''\ \}$$

Here, rule **Solve** still applies whereas the two frames are not equal modulo renaming.

Example 4. This example illustrates the role of the analysis rules. Let ε_4 be the following equation:

$$\{x_1 = \{k_1\}_{\mathsf{c}_0},\ x_2 = \{\mathsf{pair}(k_2, k_3)\}^{r_0}_{k_1},\ x_3 = \{\{k_3\}_{\mathsf{c}_1}\}^{r_1}_{k_2}\ \} \\ \approx^{?} \{x_1 = \{k_1\}_{\mathsf{c}_0},\ x_2 = \{\mathsf{pair}(k_2, k_3)\}^{r_0}_{k_1},\ x_3 = \{\{k_4\}_{\mathsf{c}_1}\}^{r_1}_{k_2}\ \}$$

Let $N = \mathsf{sdec}(x_2, \mathsf{dec}(x_1, \mathsf{c}_0))$. Since the decryption of x_3 by $\mathsf{fst}(N)$ succeeds, and r_1 occurs nowhere else in each side, rule **Encryption Analysis-1** applies twice:

$$\{x_1 = \{k_1\}_{\mathsf{c}_0},\ x_2 = \{\mathsf{pair}(k_2, k_3)\}^{r_0}_{k_1},\ y_1 = \{k_3\}_{\mathsf{c}_1}\ \} \\ \approx^{?} \{x_1 = \{k_1\}_{\mathsf{c}_0},\ x_2 = \{\mathsf{pair}(k_2, k_3)\}^{r_0}_{k_1},\ y_1 = \{k_4\}_{\mathsf{c}_1}\ \}$$

Then, we apply successively rules **Encryption Analysis-1** and **Pair Analysis** on x_2:

$$\{x_1 = \{k_1\}_{\mathsf{c}_0},\ y_1 = \{k_3\}_{\mathsf{c}_1},\ y_2 = k_2,\ y_3 = k_3\} \\ \approx^{?} \{x_1 = \{k_1\}_{\mathsf{c}_0},\ x_3 = \{k_4\}_{\mathsf{c}_1}, y_2 = k_2,\ y_3 = k_3\ \}$$

followed by **Standardize** on x_1, and on k_4 for the right-hand side:

$$\{x_1 = k'_1,\ y_1 = \{k_3\}_{\mathsf{c}_1},\ y_2 = k_2,\ y_3 = k_3\} \\ \approx^{?} \{x_1 = k'_1,\ y_1 = k'_4,\ y_2 = k_2,\ y_3 = k_3\ \}$$

Finally, rule **Solve** concludes $\bot$.

Example 5. Our last example illustrates the use of the rules **Redundancy Analysis-1,2**. Let ε_5 be the following equation:

$$\{x_1 = k_1,\ x_2 = k_2,\ x_3 = \{k_3\}^{r_0}_{k_1},\ x_4 = \{\{k_3\}^{r_0}_{k_1}\}_{k_2}\ \} \\ \approx^{?} \{x_1 = k_1,\ x_2 = k_2,\ x_3 = \{k_3\}^{r_0}_{k_1},\ x_4 = \{\{k_3\}^{r_1}_{k_1}\}_{k_2}\ \}$$

Replacing $x_3 = \{k_3\}^{r_0}_{k_1}$ on both sides with $y_1 = k_3$ would make the procedure conclude $\top$ incorrectly. This shows that we cannot relax the condition of rule **Encryption Analysis-1** concerning the occurrences of coins.

Instead, we notice that the equation $x_3 = \mathsf{pdec}(x_4, x_2)$ holds on the left-hand side but not on the right-hand side. Thus, rule **Redundancy Analysis-2** applies to conclude $\bot$.

5 Application to Security against Guessing Attacks

Weak secrets such as PINs and passwords sometimes serve as encryption keys. Their safe use is challenging because of the possibility of guessing attacks, in which data that depends on a weak secret allows an attacker to check guesses of the values of the weak secret. For example, if a message contains a fixed cleartext Hello, and it is encrypted under a password pwd drawn from a small dictionary, then an attacker that sees the message can try to decrypt it with all values in the dictionary until one yields the cleartext Hello, thus discovering a probable value for the password. The attacker may mount this attack off-line, avoiding detection. The attack is made possible by the fact that, given the data available to the attacker, pwd can be distinguished from another value pwd': $\mathsf{enc}_{\mathsf{string}}(\mathsf{Hello}, \mathsf{pwd}) \not\approx_E \mathsf{enc}_{\mathsf{string}}(\mathsf{Hello}, \mathsf{pwd'})$. Conversely, immunity to such guessing attacks can be formulated as a static equivalence between two frames, one that corresponds to what is actually available to the attacker and the other to a variant in which the weak secrets are replaced with fresh keys or with arbitrary other keys [20, 23].

We believe that, as suggested in the introduction, the treatment of guessing attacks in terms of static equivalence is attractive in several respects. This section shows that this treatment can be computationally sound. In comparison with the only previous computational justification for a formal criterion against

guessing attacks [6], the present results have several strengths. First, they apply to a criterion formulated in terms of standard notions, rather than an ad hoc criterion. Consequently, they fit into a standard analysis method which can also deal with other properties and other kinds of attacks. In addition, they are more general, in that they immediately apply to scenarios with multiple weak secrets. Finally, it is satisfying that these results follow from theorems of somewhat broader interest.

In our formalism, modeling a password as a constant w of sort *Data*, we say that the password is not revealed by a frame φ if $\varphi\{\mathsf{w} \mapsto \mathsf{c}_0\} \approx_E \varphi\{\mathsf{w} \mapsto \mathsf{c}_1\}$. The substitutions $\{\mathsf{w} \mapsto \mathsf{c}_0\}$ and $\{\mathsf{w} \mapsto \mathsf{c}_1\}$ correspond to instantiations of the password with distinct values; each of the frames represents what an attacker may obtain in the course of a protocol execution and then analyze off-line. Similarly, for n passwords represented by a sequence $\mathsf{w}_1, \ldots, \mathsf{w}_n$ of constants of sort *Data*, we say that they are not revealed by a frame φ if $\varphi\{\mathsf{w}_1 \mapsto \mathsf{c}_1, \ldots, \mathsf{w}_n \mapsto c_n\} \approx_E \varphi\{\mathsf{w}_1 \mapsto \mathsf{c}_0, \ldots, \mathsf{w}_n \mapsto c_0\}$, where $\mathsf{c}_0, \mathsf{c}_1, \ldots, \mathsf{c}_n$ are $n+1$ fresh, distinct constants of sort *Data*. By transitivity, it follows that $\varphi\{\mathsf{w}_1 \mapsto \mathsf{c}_1, \ldots, \mathsf{w}_n \mapsto c_n\} \approx_E \varphi\{\mathsf{w}_1 \mapsto \mathsf{c}'_1, \ldots, \mathsf{w}_n \mapsto c'_n\}$, where $\mathsf{c}_1, \mathsf{c}'_1, \ldots, \mathsf{c}_n, \mathsf{c}'_n$ are $2n$ constants of sort *Data*.

On the other hand, computationally, a password is hidden if it is protected from PPTIME adversaries. Such an adversary should not be able to distinguish two interpretations of the frame with different values for the password. More precisely, the following definition gives a criterion for computational hiding of a sequence of passwords:

Definition 5. Let φ be a well-formed frame, let $\mathsf{w}_1, \ldots, \mathsf{w}_n$ be n constants of sort *Data*. We say that $\mathsf{w}_1, \ldots, \mathsf{w}_n$ are computationally hidden in φ if for all (not necessarily pairwise distinct) PPTIME-computable sequences of bit-strings $\kappa_1 \ldots \kappa_n$, $\kappa'_1 \ldots \kappa'_n$ with $\kappa_i(\eta), \kappa'_i(\eta) \in \{0,1\}^{\alpha_1(\eta)}$,

$$[\![\varphi]\!]_{\eta, \mathsf{w}_1 \mapsto \kappa_1(\eta), \ldots, \mathsf{w}_n \mapsto \kappa_n(\eta)} \approx [\![\varphi]\!]_{\eta, \mathsf{w}_1 \mapsto \kappa'_1(\eta), \ldots, \mathsf{w}_n \mapsto \kappa'_n(\eta)}$$

We obtain:

Corollary 3 (Single password)**.** *Assume a secure implementation. Let φ be a well-formed frame, let w be a constant of sort Data, and let c_0, c_1 be two fresh, distinct constants of sort Data. If $\varphi\{\mathsf{w} \mapsto \mathsf{c}_0\} \approx_E \varphi\{\mathsf{w} \mapsto \mathsf{c}_1\}$ then w is computationally hidden in φ.*

Corollary 4 (Multiple passwords)**.** *Assume a secure implementation. Let φ be a well-formed frame, let $\mathsf{w}_1, \ldots, \mathsf{w}_n$ be n constants of sort Data, and let $\mathsf{c}_0, \mathsf{c}_1, \ldots, \mathsf{c}_n$ be $n+1$ fresh, distinct constants of sort Data. If $\varphi\{\mathsf{w}_1 \mapsto \mathsf{c}_1, \ldots, \mathsf{w}_n \mapsto c_n\} \approx_E \varphi\{\mathsf{w}_1 \mapsto \mathsf{c}_0, \ldots, \mathsf{w}_n \mapsto c_0\}$ then $\mathsf{w}_1, \ldots, \mathsf{w}_n$ are computationally hidden in φ.*

For example, for the frame φ associated to the EKE protocol we show in Section 4.2 that it does not reveal the password w_{AB}: $\varphi\{\mathsf{w}_{AB} \mapsto \mathsf{c}_0\} \approx_E \varphi\{\mathsf{w}_{AB} \mapsto \mathsf{c}_1\}$. Corollary 3 implies that the frame also hides the password computationally.

6 Conclusion

In this paper we investigate the computational foundations of a formal notion of data indistinguishability, static equivalence. We define a particular equational theory for which we can obtain a computational soundness result. Although they are largely based on ideas common in previous work, neither the equational theory nor our computational assumptions are straightforward. The main difficulties that we address relate to encryption under data values. Correspondingly, we obtain a soundness result for a formal criterion of protection against guessing attacks on those data values.

A direction for further work is the generalization of our results to other cryptographic primitives. For instance, certain password-based protocols make a sophisticated use of exponentiation, which we do not include in our equational theory. Yet other primitives, such as digital signatures, are important for trace properties and for process equivalences (more so than for static equivalences). We hope that, perhaps with these extensions, the present work may serve as a component of an eventual computational justification of process equivalences.

Acknowledgments. We thank Steve Kremer for helpful comments. This research was partly carried out while Mathieu Baudet was a doctoral student at the Laboratoire Spécification et Vérification (ENS Cachan, INRIA Futurs, CNRS) and while he was visiting the University of California at Santa Cruz and Microsoft Research Silicon Valley; and while Bogdan Warinschi was at Stanford University and at Loria, Université Henri Poincaré and INRIA project Cassis. It was partly supported by the National Science Foundation under Grants CCR-0204162, CCR-0208800, CCF-0524078, and ITR-0430594, and by the ARA SSIA Formacrypt and ACI Jeunes Chercheurs JC9005.

References

[1] M. Abadi, M. Baudet, and B. Warinschi. Guessing attacks and the computational soundness of static equivalence. In *Proc. 9th International Conference on Foundations of Software Science and Computation Structures (FoSSaCS'06)*, volume 3921 of *LNCS*, pages 398–412, 2006.

[2] M. Abadi and V. Cortier. Deciding knowledge in security protocols under equational theories. *Theoretical Computer Science*, 367(1-2):2–32, 2006.

[3] M. Abadi and C. Fournet. Mobile values, new names, and secure communication. In *Proc. 28th ACM Symposium on Principles of Programming Languages (POPL'01)*, pages 104–115, 2001.

[4] M. Abadi and A. D. Gordon. A bisimulation method for cryptographic protocols. *Nordic Journal of Computing*, 5(4):267–303, 1998.

[5] M. Abadi and P. Rogaway. Reconciling two views of cryptography (The computational soundness of formal encryption). *Journal of Cryptology*, 15(2):103–127, 2002.

[6] M. Abadi and B. Warinschi. Password-based encryption analyzed. In *Proc. 32nd International Colloquium on Automata, Languages and Programming (ICALP'05)*, volume 3580 of *LNCS*, pages 664–676. Springer, 2005.

[7] M. Backes, B. Pfitzmann, and M. Waidner. A composable cryptographic library with nested operations. In *Proc. 10th ACM Conference on Computer and Communications Security (CCS'03)*, pages 220–330, 2003.

[8] M. Baudet. Deciding security of protocols against off-line guessing attacks. In *Proc. 12th ACM Conference on Computer and Communications Security (CCS'05)*, pages 16–25, 2005.

[9] M. Baudet, V. Cortier, and S. Kremer. Computationally sound implementations of equational theories against passive adversaries. *Information and Computation*, 207(4):496–520, 2009.

[10] M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval. Key-privacy in public-key encryption. In *Advances in Cryptology – ASIACRYPT'01*, volume 2248 of *LNCS*, pages 566–582. Springer, 2001.

[11] M. Bellare, A. Desai, E. Jakopii, and P. Rogaway. A concrete security treatement of symmetric encryption: Analysis of the DES modes of operation. In *Proc. 38th Symposium on Foundations of Computer Science (FOCS'97)*, pages 394–403, 1997.

[12] M. Bellare, D. Pointcheval, and P. Rogaway. Authenticated key exchange secure against dictionary attacks. In *Advances in Cryptology – EUROCRYPT'00*, volume 1807 of *LNCS*, pages 139–155. Springer, 2000.

[13] M. Bellare and P. Rogaway. Introduction to modern cryptography. Available at: http://www.cs.ucsd.edu/users/mihir/cse207/classnotes.html.

[14] M. Bellare and P. Rogaway. On the construction of variable-input-length ciphers. In *Proc. 6th Workshop on Fast Software Encryption (FSE'99)*, volume 1636 of *LNCS*, pages 321–344. Springer, 1999.

[15] S. M. Bellovin and M. Merritt. Encrypted key exchange: Password-based protocols secure against dictionary attacks. In *Proc. 1992 IEEE Symposium on Security and Privacy (SSP'92)*, pages 72–84, 1992.

[16] J. Black and P. Rogaway. Ciphers with arbitrary finite domains. In *Proc. Cryptographer's Track at RSA Conference (CT-RSA'02)*, volume 2271 of *LNCS*, pages 114–130. Springer, 2002.

[17] 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, February-March 2008.

[18] M. Boreale, R. De Nicola, and R. Pugliese. Proof techniques for cryptographic processes. In *Proc. 14th IEEE Symposium on Logic in Computer Science (LICS'99)*, pages 157–166, 1999.

[19] V. Boyko, P. MacKenzie, and S. Patel. Provably secure password-authenticated key exchange using Diffie-Hellman. In *Advances in Cryptology – EUROCRYPT'00*, volume 1807 of *LNCS*, pages 156–171. Springer, 2000.

[20] R. Corin, J. M. Doumen, and S. Etalle. Analysing password protocol security against off-line dictionary attacks. Technical report TR-CTIT-03-52, Centre for Telematics and Information Technology, Univ. of Twente, The Netherlands, 2003.

[21] R. Corin, S. Malladi, J. Alves-Foss, and S. Etalle. Guess what? Here is a new tool that finds some new guessing attacks (extended abstract). In *IFIP WG 1.7 and ACM SIGPLAN Workshop on Issues in the Theory of Security (WITS'03)*, pages 62–71, 2003.

[22] S. Delaune and F. Jacquemard. A theory of dictionary attacks and its complexity. In *Proc. 17th IEEE Computer Security Foundations Workshop (CSFW'04)*, pages 2–15, 2004.

[23] C. Fournet. Private communication, 2002.

[24] R. Gennaro and Y. Lindell. A framework for password-based authenticated key exchange. In *Advances in Cryptology – EUROCRYPT'03*, volume 2656 of *LNCS*, pages 524–543. Springer, 2003.

[25] O. Goldreich and Y. Lindell. Session key generation using human passwords only. In *Advances in Cryptology – CRYPTO'01*, volume 2139 of *LNCS*, pages 403–432. Springer, 2001.

[26] L. Gong. Verifiable-text attacks in cryptographic protocols. In *Proc. 9th IEEE Conference on Computer Communications (INFOCOM'90)*, pages 686–693, 1990.

[27] L. Gong, T. M. A. Lomas, R. M. Needham, and J. H. Saltzer. Protecting poorly chosen secrets from guessing attacks. *IEEE Journal on Selected Areas in Communications*, 11(5):648–656, 1993.

[28] J. Katz, R. Ostrovsky, and M. Yung. Practical password-authenticated key exchange provably secure under standard assumptions. In *Advances in Cryptology – EUROCRYPT'01*, volume 2045 of *LNCS*, pages 475–494. Springer, 2001.

[29] P. Laud. Symmetric encryption in automatic analyses for confidentiality against active adversaries. In *Proc. 2004 IEEE Symposium on Security and Privacy (SSP'04)*, pages 71–85, 2004.

[30] G. Lowe. Analysing protocols subject to guessing attacks. *Journal of Computer Security*, 12(1):83–98, 2004.

[31] D. Micciancio and B. Warinschi. Completeness theorems for the Abadi-Rogaway logic of encrypted expressions. *Journal of Computer Security*, 12(1):99–129, 2004.

[32] D. Micciancio and B. Warinschi. Soundness of formal encryption in the presence of active adversaries. In *Proc. Theory of Cryptography Conference (TCC'04)*, volume 2951 of *LNCS*, pages 133–151. Springer, 2004.

[33] D. H. Phan and D. Pointcheval. About the security of ciphers (semantic security and pseudo-random permutations). In *Proc. Selected Areas in Cryptography (SAC'04)*, volume 3357 of *LNCS*, pages 185–200. Springer, 2004.

[34] M. Steiner, G. Tsudik, and M. Waidner. Refinement and extension of encrypted key exchange. *ACM SIGOPS Oper. Syst. Rev.*, 29(3):22–30, 1995.

[35] G. Tsudik and E. V. Herreweghen. Some remarks on protecting weak secrets and poorly-chosen keys from guessing attacks. In *Proc. 12th IEEE Symposium on Reliable Distributed Systems (SRDS'93)*, pages 136–142, 1993.

Appendix

The appendix contains additional details of the proofs.

A Properties of the Transformation Rules

In this section we prove the properties of the transformation relation $\Longrightarrow$ defined and used in Section 4.

We say that an equation ε is *formally true* if either $\varepsilon = \top$ or $\varepsilon = (\varphi_1 \approx^? \varphi_2)$ and the two frames are statically equivalent in E, that is, $\varphi_1 \approx_E \varphi_2$. It is *computationally true* if either $\varepsilon = \top$ or $\varepsilon = (\varphi_1 \approx^? \varphi_2)$ and the two frames are indistinguishable, that is, $[\![\varphi_1]\!] \approx [\![\varphi_2]\!]$. Finally, an equation ε is *reachable* iff there exists a finite derivation $\varepsilon_0 \Longrightarrow \ldots \Longrightarrow \varepsilon$ that originates from an equation $\varepsilon_0 = \varphi_1 \approx_E \varphi_2$ between two well-formed frames.

The expected properties of the transformation relation can be accounted for, in a local way, as follows.

- **Termination.** If ε is reachable, then there exists no infinite sequence of transformations $\varepsilon = \varepsilon_0 \Longrightarrow \varepsilon_1 \Longrightarrow \ldots \Longrightarrow \varepsilon_n \ldots$
- **Progress.** If ε is reachable and $\varepsilon \notin \{\top, \bot\}$, then there exists ε' such that $\varepsilon \Longrightarrow \varepsilon'$.
- **Effectiveness.** The above equation ε' is effectively computable from ε.
- **Formal soundness and completeness.** If $\varepsilon \Longrightarrow \varepsilon'$, then ε is formally true iff ε' is.
- **Computational soundness.** Assume a secure implementation. If $\varepsilon \Longrightarrow \varepsilon'$ and ε' is computationally true, then so is ε.

A.1 Termination and Progress

Recall that the procedure consists of the following sequences of rule applications:

1. a maximal sequence of applications of rule **Undecipherable Encryption**;
2. a maximal sequence of applications of rule **Split Names**;
3. a maximal sequence of applications of **Analysis** rules, that is either **Pair Analysis**, **Redundancy Analysis- 1,2**, or **Encryption Analysis-1,2**;
4. a maximal sequence of applications of rule **Standardize**; and finally
5. a maximal sequence of applications of rule **Solve**.

First, we check that

- every rule except the last two (**Standardize** and **Solve**) preserves the well-formedness of frames;
- each of the sequences of rules above terminates on well-formed frames (respectively, on every frame for the last two rules).

These properties are clear for most of the rules. In the case of **Encryption Analysis**, assume that the two sides of the equation ε are well-formed. Then, given that W is $\mathcal{R}$-reduced, the relation $\{W\}_{\vec{M}^{-1}\varphi_2}\downarrow_{\mathcal{R}} = \mathsf{penc}(S_2, \mathsf{pub}(N\varphi_2)\downarrow_{\mathcal{R}}, r_2)$ is equivalent to $W = \{\mathsf{penc}(S_2, \mathsf{pub}(N\varphi_2)\downarrow_{\mathcal{R}}, r_2)\}_{\vec{M}\varphi_2\downarrow_{\mathcal{R}}}$ (and similarly for symmetric encryption). Hence, S_2 is a subterm of the right-hand side. Concerning the termination of **Standardize**, notice that one application of this rule decreases the size of $\mathrm{names}(\varphi) - \mathrm{im}(\varphi)$, where φ is the side of the equation under consideration.

Concerning progress, we successively prove the following facts:

(1) for every equation $\varphi_1 \approx^? \varphi_2$ reached after sequence 1 (rule **Undecipherable Encryption**), every subterm T of φ_i of sort $\tau \neq \mathit{Coins}$ is deducible from φ_i, and φ_i contains no subterm of the form $\mathsf{penc}(U, k', r)$;

(2) for every equation $\varphi_1 \approx^? \varphi_2$ reached after sequence 2 (rule **Split Names**), φ_1 and φ_2 contain no name of sort $\mathit{Pair}[\tau_1, \tau_2]$;

(3) for every equation $\varphi_1 \approx^? \varphi_2$ reached after sequence 3 (rules **Pair**, **Redundancy** and **Encryption Analysis**), φ_1 and φ_2 contain no symbol pair, penc, or senc;

(4) for every equation $\varphi_1 \approx^? \varphi_2$ reached after sequence 4 (rule **Standardize**), each φ_i is standard.

Given the definition of rule **Solve**, the last point implies that if sequences 1–5 are completed, then the resulting equation is either $\bot$ or $\top$.

We now prove the four points. Throughout the proof, the source equation of a rule is denoted $\varepsilon = \varphi_1^0 \approx^? \varphi_2^0$ whereas the resulting equation is written $\varepsilon' = \varphi_1^1 \approx^? \varphi_2^1$.

(1) First, we verify that the rules of sequences 2–5 do not create non-deducible subterms. (Clearly, they create no terms of the form $\mathsf{penc}(U, k', r)$.) Concerning **Redundancy Analysis-1**, since the suppressed components U and W remain deducible in their respective frames, so do the remaining subterms. Concerning **Encryption Analysis-1**, let T be a subterm of the resulting frame φ_i^1. Thanks to the condition on the coins r_i in φ_i^0, r_i does not occur in T. Since $\vec{V}$ and T are deducible from φ, by Lemmas 12 and 13 of Appendix B, we obtain that T is deducible $\varphi_i^1 = \varphi_i \uplus \{y = S_i\}$. The cases of the other rules are straightforward.

Second, we prove the progress property for rule **Undecipherable Encryption**. Assume an equation $\varphi_1^0 \approx^? \varphi_2^0$ reached just after sequence 1. By contradiction, assume for instance a subterm $\mathsf{penc}(T, k', r)$ or a non-deducible subterm T of sort $\tau \neq \mathit{Coins}$ in φ_1. In the latter case, by Lemma 15 (Appendix B), T appears either in a key position or encrypted under a non-deducible key.

This implies that φ_1^0 contains either a subterm $\mathsf{penc}(T, k', r)$, or a name k (of sort $\tau \neq \mathit{EKey}$) in key position, and not deducible from φ_1. In the former case, rule **Undecipherable Encryption** applies. In the latter case, using the acyclicity condition, let k_0 be a non-deducible name (of sort $\tau \neq \mathit{EKey}$) in key position, maximal for the key-cycle ordering, that is, such that k_0 does not appear encrypted (unless in key position) under any key. Since k_0 is not deducible, by Lemma 15, it must appear only in key position. Rule **Undecipherable Encryption** then applies on a maximal subterm T of the form $\mathsf{pub}(k_0)$, $\mathsf{senc}(U, k_0, r)$, $\mathsf{enc}(U, k_0)$.

(2) It is clear from the definitions that the rules of sequences 3–5 do not create names of sort $\mathit{Pair}[\tau_1, \tau_2]$, and that rule **Split Names** terminates when no such names remain.

(3) Sequences 3–5 do not create symbols pair, penc, and senc. Assume an equation $\varphi_1^0 \approx^? \varphi_2^0$ reached just after sequence 3. By contradiction, assume a maximal subterm U of φ_1^0 (for instance) such that the head symbol of U is pair, penc, or senc. Given the sorting system, since, by well-formedness and (2), the other available symbols are only enc, pub, constants and names, there exists x in $\mathrm{dom}(\varphi_1^0)$ such that $x\varphi_1^0$ is of the form $\{U\}_{\vec{V}}$ (possibly $\vec{V} = \Lambda$). Let $\varphi_1^0 = \varphi_1 \uplus \{x = \{U\}_{\vec{V}}\}$.

If U is a pair, then since $T_{\mathsf{enc}} \cap \{\mathit{Pair}[\tau_1, \tau_2]\}_{\tau_1, \tau_2} = \emptyset$, we have $\vec{V} = \Lambda$. For the same reason, and using (2), the head symbol of $x\varphi_2^0$ is pair, hence rule **Pair Analysis** applies.

Otherwise, by (1) and (2) we may assume for instance that U is of the form $\mathsf{penc}(S_1, \mathsf{pub}(T_1), r_1)$. (The case of symmetric encryption is similar.) By (1), T_1 and $\vec{V}$ must be deducible from φ_1^0. By Lemmas 12 and 13 of Appendix B, this implies that T_1 and $\vec{V}$ are deducible from φ_1 itself. Let $\vec{M}$ and M be such that $\mathrm{var}(\vec{M}, N) \subseteq \mathrm{dom}(\varphi)$, $\mathrm{names}(\vec{M}, N) \cap \mathrm{names}(\varepsilon) = \emptyset$, $\vec{M}\varphi_1 =_E \vec{V}$, and $N\varphi =_E T_1$. (By stability of E under renaming we may exclude a finitely larger set of names in $\vec{M}$ and N.)

We distinguish three cases.

- If U is deducible from φ_1, that is, there exists P such that $\mathrm{var}(P) \subseteq \mathrm{dom}(\varphi)$, $\mathrm{names}(P) \cap \mathrm{names}(\varepsilon) = \emptyset$, and $P\varphi_1 =_E U$, then one of the two rules **Redundancy Analysis-1,2** applies, depending on whether the equation $x =_E \{P\}_{\vec{M}}$ holds in φ_2^0. We obtain a contradiction.
- If the disequality $\mathsf{pdec\_success}(\{x\}_{\vec{M}^{-1}}, N)\varphi_2^0 \neq_E 1$ holds, then rule **Encryption Analysis-2** applies.
- If the two previous cases do not apply, then the term U is not deducible from the frame φ_1 and we have $\mathsf{pdec\_success}(\{x\}_{\vec{M}^{-1}}, N)\varphi_2 =_E 1$. First, we show that r_1 does not occur elsewhere in φ_1^0.

  Indeed, by contradiction, suppose that r_1 occurs elsewhere in φ_1^0. By well-formedness, this means that U is a subterm of φ_1. As U is not deducible from φ_1, by Lemma 15 and property (1), U occurs in φ_1 under an encryption by a key k_1 of sort different from *EKey*, not deducible from φ_1. Besides, by (1), k_1 is deducible from φ_1^0. By Lemma 15, this implies that k_1 occurs in φ_1^0 not in key position. Since k_1 encrypts U, by well-formedness, this occurrence must be in φ_1. (It cannot be in U nor in $\vec{V}$.) Applying the same reasoning to k_1 and so forth, we obtain the existence of an infinite sequence of name $k_1, k_2, \ldots, k_i, \ldots$ of sort different from *EKey*, such that for all $i \geq 1$, k_i encrypts k_{i-1}, occurs in φ_1 not in key position and is not deducible from φ_1. Since $\mathrm{names}(\varphi_1)$ is finite, this contradicts the acyclicity condition.

  Hence, r_1 does not occur elsewhere in φ_1^0. Let $U_2 = \{W\}_{\vec{M}^{-1}\varphi_2}\downarrow_{\mathcal{R}}$. From the equation

  $$\mathsf{pdec\_success}(\{x\}_{\vec{M}^{-1}}, N)\varphi_2 =_E 1$$

  and by well-formedness, we deduce that U_2 is of the form $\mathsf{penc}(S_2, \mathsf{pub}(T_2), r_2)$. Similarly as for U_1, we show that either U_2 is deducible from φ_2^0 (thus rule **Redundancy Analysis** applies), or r_2 occurs once in φ_2^0. In the latter case, rule **Undecipherable Encryption** applies. We obtain a contradiction.

(4) Let $\varphi_1^0 \approx^? \varphi_2^0$ be an equation reached just after sequence 4. By contradiction, assume that, for instance, φ_1^0 is not standard. This means that the set $\mathrm{names}(\varphi_1^0) - \mathrm{im}(\varphi_1^0)$ is not empty.

By (3), φ_1^0 contains only constants, names and symbols enc, dec and pub. Given the sorting system, this entails that for all x in $\mathrm{dom}(\varphi_1^0)$, $x\varphi_1^0$ is of the form

$$x\varphi_1^0 = \begin{cases} & \{c\}_{\vec{C}[a_1 \ldots a_n]} \\ \text{or} & \{a\}_{\vec{C}[a_1 \ldots a_n]} \\ \text{or} & \{\mathsf{pub}(k)\}_{\vec{C}[a_1 \ldots a_n]} \end{cases}$$

where c denotes an arbitrary constant, a is a name, $\vec{C}$ is a (possibly empty) vectorial context made of symbols enc, dec only, and the a_i are names of sort *Data*.

Because of the form of the frame, the saturation procedure described in Corollary 10 (Appendix B) boils down to the following rules:

(i) for every $x \in \mathrm{dom}(\varphi_1^0)$, $x\varphi \in \mathrm{sat}(\varphi_1^0)$; in other words, $\mathrm{im}(\varphi_1^0) \subseteq \mathrm{sat}(\varphi_1^0)$;

(ii) for every $t \in \mathrm{st}(\varphi_1^0)$, if $t = f(t_1, \ldots, t_n)$ and $t_1, \ldots, t_n \in \mathrm{sat}(\varphi_1^0)$, then $t \in \mathrm{sat}(\varphi_1^0)$;

(iii) if $x\varphi_1^0 = \{a\}_{\vec{C}[a_1 \ldots a_n]}$, and $a_1, \ldots, a_n \in \mathrm{sat}(\varphi_1^0)$ then for every (possibly empty) subcontext $\vec{C}'$ of $\vec{C}$ we have $\{a\}_{\vec{C}'[a_1 \ldots a_n]} \in \mathrm{sat}(\varphi_1^0)$.

Note that regarding deducibility of names,

- rule (i) may be restricted to names in $\mathrm{im}(\varphi_1^0)$;
- rule (ii) is useless; and

- rule (iii) may be restricted to $\vec{C}' = \Lambda$, that is, to the conclusion $a \in \mathrm{sat}(\varphi_1^0)$.

Since, by (1), every name in φ_1^0 is deducible, $\mathrm{names}(\varphi_1^0)$ is the smallest set saturated by rule (i) and rule (iii) both restricted to names. In particular, since $\mathrm{names}(\varphi_1^0) - \mathrm{im}(\varphi_1^0) \neq \emptyset$, there exists a name $a \in \mathrm{names}(\varphi_1^0) - \mathrm{im}(\varphi_1^0)$ such that the corresponding derivation using (i) and (iii) is minimal. As $a \notin \mathrm{im}(\varphi_1^0)$, this derivation ends with (iii). Thus there exists a variable x such that $x\varphi_1^0 = \{a\}_{\vec{C}[a_1 \ldots a_n]}$ and $a_1 \ldots a_n \in \mathrm{im}(\varphi_1^0)$. Hence, rule **Standardize** applies.

A.2 Effectiveness

Our main proof does not strictly require the transformation rules to be effective. Yet, we briefly justify this property in order to obtain a decision procedure for the class of equivalences between well-formed frames.

The effectiveness of rules **Undecipherable Encryption**, **Split Names**, **Pair Analysis**, **Standardize** and **Solve** is clear. Concerning rules **Encryption Analysis-1,2**, our proof of progress (above) shows that during the procedure, one of these two rules applies whenever

- $U = \mathsf{penc}(S_1, \mathsf{pub}(T_1), r_1)$ where r_1 does not occur elsewhere in the same side of ε; and
- we have found $\vec{M}$, N such that $\mathrm{var}(\vec{M}, N) \subseteq \mathrm{dom}(\varphi_1)$, $\mathrm{names}(\vec{M}, N) \cap \mathrm{names}(\varepsilon) = \emptyset$, $M\varphi_1 =_E U$ and $N\varphi_1 = T_1$.

Finding such terms $\vec{M}$ and N is done using the classical decision procedure for deducibility (recalled in Corollary 10 of Appendix B). Rules **Redundancy Analysis-1,2** are effective for a similar reason.

A.3 Formal Soundness and Completeness

As above, the source equation of a rule is denoted $\varepsilon = \varphi_1^0 \approx^? \varphi_2^0$ whereas the resulting equation is written $\varepsilon' = \varphi_1^1 \approx^? \varphi_2^1$.

We check that for each rule ε is formally true iff ε' is. In the definition of static equivalence:

$\varphi_1 \approx_E \varphi_2$ iff for all M, N such that $\mathrm{var}(M, N) \subseteq \mathrm{dom}(\varphi_1) = \mathrm{dom}(\varphi_2)$ and $\mathrm{names}(M, N) \cap \mathrm{names}(\varphi_1, \varphi_2)$, we have

$$M\varphi_1 =_E N\varphi_1 \Leftrightarrow M\varphi_2 =_E N\varphi_2,$$

note that we may forbid any additional finite set of names from M and N without loss of generality, by the stability of E by renaming of names.

- **Undecipherable Encryption**. We prove that $\varphi_1 \approx_E \varphi_1^1 = \varphi_1\{T \mapsto n\}$.

  As k appears only in key positions, by Lemma 15, we have $\varphi_1 \not\vdash_E k$.

  Since $\mathcal{R}$ is subterm and convergent, and both T and φ_1 are $\mathcal{R}$-reduced, we obtain $\varphi_1 \approx_E \mathrm{cut}^E_{T,n}(\varphi_1) = \varphi_1^1$ by Proposition 17 of appendix B, with $\mathcal{R}_0$ successively equal to the remaining equations in $\mathcal{R}$. Indeed, for the case $T = \mathsf{penc}(U, \mathsf{pub}(k), r)$, r is not deducible (Lemma 14 of Appendix B), and $\mathsf{pub}(k) =_E \mathsf{pub}(V)$ is equivalent to $k =_E V$ as $\mathcal{R}$ is convergent and no left-hand side of rules in $\mathcal{R}$ has head symbol pub. Similarly, for all V, $k' \neq_E \mathsf{pub}(V)$.

- **Split Names**. We prove that $\varphi_1 \approx_E \varphi_1^1 = \varphi_1\{n \mapsto \mathsf{pair}(n_1, n_2)\downarrow_{\mathcal{R}}\}$.

  Let M, N be two terms with $\mathrm{names}(M, N) \cap \mathrm{names}(\varphi_1, \varphi_1^1) = \emptyset$. Assume $M\varphi_1 =_E N\varphi_1$. Then

$$\begin{aligned} M\varphi_1^1 &=_E (M\varphi_1)\{n \mapsto \mathsf{pair}(n_1, n_2)\} \\ &=_E (N\varphi_1)\{n \mapsto \mathsf{pair}(n_1, n_2)\} =_E N\varphi_1^1 \end{aligned}$$

Conversely, assume $M\varphi_1^1 =_E N\varphi_1^1$. Then

$$\begin{array}{rcl} M\varphi_1 & =_E & (M\varphi_1)\{n_1 \mapsto \mathsf{fst}(n),\, n_2 \mapsto \mathsf{snd}(n)\} \\ & =_E & (N\varphi_1^1)\{n_1 \mapsto \mathsf{fst}(n),\, n_2 \mapsto \mathsf{snd}(n)\} \\ & =_E & N\varphi_1^1 \end{array}$$

- **Pair Analysis**. Assume $\varphi_1^0 \approx_E \varphi_2^0$. Let P, Q be two terms such that $\mathrm{var}(P,Q) \subseteq \mathrm{dom}(\varphi_1^1) = \mathrm{dom}(\varphi_2^1)$, and $\mathrm{names}(P,Q) \cap \mathrm{names}(\varphi_1^1, \varphi_2^1) = \emptyset$. Given that

$$P\varphi_i^1 =_E P\{y \mapsto \mathsf{fst}(x),\, z \mapsto \mathsf{snd}(x)\}\varphi_1^0$$
$$\mathrm{names}(P\{y \mapsto \mathsf{fst}(x),\, z \mapsto \mathsf{snd}(x)\}) \cap \mathrm{names}(\varphi_1^0, \varphi_2^0) = \emptyset$$

and similarly for Q, we have

$$\begin{array}{rc} & P\varphi_1^1 =_E Q\varphi_1^1 \\ \Leftrightarrow & P\{y \mapsto \mathsf{fst}(x),\, z \mapsto \mathsf{snd}(x)\}\varphi_1^0 =_E Q\{y \mapsto \mathsf{fst}(x),\, z \mapsto \mathsf{snd}(x)\}\varphi_1^0 \\ \Leftrightarrow & P\{y \mapsto \mathsf{fst}(x),\, z \mapsto \mathsf{snd}(x)\}\varphi_2^0 =_E Q\{y \mapsto \mathsf{fst}(x),\, z \mapsto \mathsf{snd}(x)\}\varphi_2^0 \\ \Leftrightarrow & P\varphi_2^1 =_E Q\varphi_2^1 \end{array}$$

Assume $\varphi_1^1 \approx_E \varphi_2^1$. Let P, Q be two terms such that $\mathrm{var}(P,Q) \subseteq \mathrm{dom}(\varphi_1^0) = \mathrm{dom}(\varphi_2^0)$, and $\mathrm{names}(P,Q) \cap \mathrm{names}(\varphi_1^0, \varphi_2^0) = \emptyset$. Given that

$$P\varphi_i^0 =_E P\{x \mapsto \mathsf{pair}(y,z)\}\,\varphi_i^1$$
$$\mathrm{names}(P\{x \mapsto \mathsf{pair}(y,z)\}) \cap \mathrm{names}(\varphi_1^1, \varphi_2^1) = \emptyset$$

and similarly for Q, we have

$$\begin{array}{rc} & P\varphi_1^0 =_E Q\varphi_1^0 \\ \Leftrightarrow & P\{x \mapsto \mathsf{pair}(y,z)\}\varphi_1^1 =_E Q\{x \mapsto \mathsf{pair}(y,z)\}\varphi_1^1 \\ \Leftrightarrow & P\{x \mapsto \mathsf{pair}(y,z)\}\varphi_2^1 =_E Q\{x \mapsto \mathsf{pair}(y,z)\}\varphi_2^1 \\ \Leftrightarrow & P\varphi_2^0 =_E Q\varphi_2^0 \end{array}$$

- **Redundancy Analysis-2**. Since the test $x =^? M$ holds in φ_1^0 but does not in φ_2^0, we have indeed $\varphi_1^0 \not\approx_E \varphi_2^0$.

- **Redundancy Analysis-1**. Assume $\varphi_1^0 \approx_E \varphi_2^0$. Let Q_1, Q_2 be two terms such that $\mathrm{var}(Q_1,Q_2) \subseteq \mathrm{dom}(\varphi_1) = \mathrm{dom}(\varphi_2)$, and $\mathrm{names}(Q_1,Q_2) \cap \mathrm{names}(\varphi_1^0, \varphi_2^0) = \emptyset$. We have that

$$\begin{array}{rc} & Q_1\varphi_1 =_E Q_2\varphi_1 \\ \Leftrightarrow & Q_1\varphi_1^0 =_E Q_2\varphi_1^0 \\ \Leftrightarrow & Q_1\varphi_2^0 =_E Q_2\varphi_2^0 \\ \Leftrightarrow & Q_1\varphi_2 =_E Q_2\varphi_2 \end{array}$$

Conversely, assume $\varphi_1 \approx_E \varphi_2$. Let Q_1, Q_2 be two terms such that we have $\mathrm{var}(Q_1,Q_2) \subseteq \mathrm{dom}(\varphi_1) = \mathrm{dom}(\varphi_2)$, and $\mathrm{names}(Q_1,Q_2) \cap \mathrm{names}(\varphi_1^0, \varphi_2^0) = \emptyset$. Given that for $i, j \in \{1,2\}$,

$$Q_j\varphi_i^0 =_E Q_j\{x \mapsto M\}\varphi_i$$
$$\mathrm{names}(Q_j\{x \mapsto M\}) \cap \mathrm{names}(\varphi_1, \varphi_2) = \emptyset$$

we have

$$\begin{array}{cc} & Q_1\varphi_1^0 =_E Q_2\varphi_1^0 \\ \Leftrightarrow & Q_1\{x \mapsto M\}\varphi_1 =_E Q_2\{x \mapsto M\}\varphi_1 \\ \Leftrightarrow & Q_1\{x \mapsto M\}\varphi_2 =_E Q_2\{x \mapsto M\}\varphi_2 \\ & Q_1\varphi_2^0 =_E Q_2\varphi_2^0 \end{array}$$

- **Encryption Analysis-2**. Since the test $\mathsf{pdec\_success}(\{x\}_{\vec{M}^{-1}}, N) =^? 1$ (and respectively the test $\mathsf{sdec\_success}(\{x\}_{\vec{M}^{-1}}, N) =^? 1$) holds in φ_1^0, but does not in φ_2^0, we have indeed $\varphi_1^0 \not\approx_E \varphi_2^0$.

- **Encryption Analysis-1**. We consider only the case of public-key encryption penc since the other case is similar.

  Assume $\varphi_1^0 \approx_E \varphi_2^0$. Let Q_1, Q_2 be two terms such that $\mathrm{var}(Q_1, Q_2) \subseteq \mathrm{dom}(\varphi_1^1) = \mathrm{dom}(\varphi_2^1)$, and $\mathrm{names}(Q_1, Q_2) \cap \mathrm{names}(\varphi_1^0, \varphi_2^0) = \emptyset$. Given that for $i, j \in \{1, 2\}$

$$\begin{array}{c} Q_j\varphi_i^1 =_E Q_j\{y \mapsto \mathsf{pdec}(\{x\}_{\vec{M}^{-1}}, N)\}\,\varphi_i^0 \\ \mathrm{names}(Q_j\{y \mapsto \mathsf{pdec}(\{x\}_{\vec{M}^{-1}}, N)) \cap \mathrm{names}(\varphi_1^0, \varphi_2^0) = \emptyset \end{array}$$

  we have

$$\begin{array}{cc} & Q_1\varphi_1^1 =_E Q_2\varphi_1^1 \\ \Leftrightarrow & Q_1\{y \mapsto \mathsf{pdec}(\{x\}_{\vec{M}^{-1}}, N)\}\,\varphi_1^0 =_E Q_2\{y \mapsto \mathsf{pdec}(\{x\}_{\vec{M}^{-1}}, N)\}\,\varphi_1^0 \\ \Leftrightarrow & Q_1\{y \mapsto \mathsf{pdec}(\{x\}_{\vec{M}^{-1}}, N)\}\,\varphi_2 =_E Q_2\{y \mapsto \mathsf{pdec}(\{x\}_{\vec{M}^{-1}}, N)\}\,\varphi_2 \\ \Leftrightarrow & Q_1\varphi_2^1 =_E Q_2\varphi_2^1 \end{array}$$

  Conversely, assume $\varphi_1^1 \approx_E \varphi_2^1$. Let Q_1, Q_2 be two terms such that $\mathrm{var}(Q_1, Q_2) \subseteq \mathrm{dom}(\varphi_1^0) = \mathrm{dom}(\varphi_2^0)$, and $\mathrm{names}(Q_1, Q_2) \cap \mathrm{names}(\varphi_1^0, \varphi_2^0) = \emptyset$.

  Let r_0 be a fresh name, and ρ_i be the renaming $\rho_i = \{r_i \mapsto r_0\}$. Provided that r_i occurs nowhere else in φ_i^0, we have that for every $i, j \in \{1, 2\}$,

$$\begin{array}{c} Q_j\varphi_i^0\rho_i =_E Q_j\{x \mapsto \{\mathsf{penc}(y, \mathsf{pub}(N), r_0)\}_{\vec{M}}\}\,\varphi_i^1 \\ \mathrm{names}(Q_j\{x \mapsto \{\mathsf{penc}(y, \mathsf{pub}(N), r_0)\}_{\vec{M}}\}) \cap \mathrm{names}(\varphi_1^0, \varphi_2^0) = \emptyset \end{array}$$

  Hence

$$\begin{array}{rcl} Q_1\varphi_1^0 & =_E & Q_2\varphi_1^0 \\ & \Leftrightarrow & \\ Q_1\varphi_1^0\rho_1 & =_E & Q_2\varphi_1^0\rho_1 \\ & \Leftrightarrow & \\ Q_1\{x \mapsto \{\mathsf{penc}(y, \mathsf{pub}(N), r_0)\}_{\vec{M}}\}\,\varphi_1^1 & =_E & Q_2\{x \mapsto \{\mathsf{penc}(y, \mathsf{pub}(N), r_0)\}_{\vec{M}}\}\,\varphi_1^1 \\ & \Leftrightarrow & \\ Q_1\{x \mapsto \{\mathsf{penc}(y, \mathsf{pub}(N), r_0)\}_{\vec{M}}\}\,\varphi_2^1 & =_E & Q_2\{x \mapsto \{\mathsf{penc}(y, \mathsf{pub}(N), r_0)\}_{\vec{M}}\}\,\varphi_2^1 \\ & \Leftrightarrow & \\ Q_1\varphi_2\rho_2 & =_E & Q_2\varphi_2\rho_2 \\ & \Leftrightarrow & \\ Q_1\varphi_2 & =_E & Q_2\varphi_2 \end{array}$$

- **Standardize**. We prove $\varphi_1^0 \approx_E \varphi_1^1$.

  Let M, N be two terms such that $\mathrm{var}(M,N) \subseteq \mathrm{dom}(\varphi_i^j)$ and $\mathrm{names}(M,N) \cap \mathrm{names}(\varphi_1^0, \varphi_1^1) = \emptyset$. If $M\varphi_1^0 =_E N\varphi_1^0$, then

$$\begin{aligned} M\varphi_1^1 &=_E (M\varphi_1^0)\{a \mapsto \{a'\}_{\vec{C}[a_1,\ldots,a_n]^{-1}}\} \\ &=_E (N\varphi_1^0)\{a \mapsto \{a'\}_{\vec{C}[a_1,\ldots,a_n]^{-1}}\} \\ &=_E N\varphi_1^1 \end{aligned}$$

  Conversely, if $M\varphi_1^1 =_E N\varphi_1^1$, then

$$\begin{aligned} M\varphi_1^0 &=_E (M\varphi_1^1)\{a' \mapsto \{a\}_{\vec{C}[a_1,\ldots,a_n]}\} \\ &=_E (N\varphi_1^1)\{a' \mapsto \{a\}_{\vec{C}[a_1,\ldots,a_n]}\} \\ &=_E N\varphi_1^0 \end{aligned}$$

- **Solve**. The result follows from Proposition 2.

A.4 Computational Soundness

We first prove several simple lemmas. Intuitively, the first one states that a true formal equation is always true concretely.

Lemma 5 (Unconditional soundness of $=_E$)**.** *Let T_1 and T_2 be two closed terms such that $T_1 =_E T_2$. We have that*

$$\Pr\left[e_1, e_2 \leftarrow [\![T_1, T_2]\!]_\eta \ : \ e_1 = e_2\right] = 1$$

Proof. Let $\cong$ be the relation on terms defined by $T_1 \cong T_2$ iff for every η, for every well-sorted concrete mapping ψ with $\mathrm{dom}(\psi) = \mathrm{var}(T_1, T_2)$,

$$\Pr\left[e_1, e_2 \leftarrow [\![T_1, T_2]\!]_{\eta,\psi} \ : \ e_1 = e_2\right] = 1$$

For every generating equation $l = r$ of E, our implementation satisfies $l \cong r$. From its definition, it is also straightforward to check that $\cong$ is a congruence stable by substitution. Therefore, $\cong$ contains the equational theory E. □

As a corollary, we have that $\varphi_1 =_E \varphi_2$ implies that the two family of distributions $[\![\varphi_1]\!]$ and $[\![\varphi_2]\!]$ are identical.

The definition of an implementation (Section 3) ensures that no element in a set $[\![\tau]\!]_\eta$ has probability 0 according to the distribution $(\xleftarrow{R} [\![\tau]\!]_\eta)$. Hence, Lemma 5 can be stated equivalently as follows:

> for every closed terms T_1 and T_2 such that $T_1 =_E T_2$, for every η, for every well-sorted concrete mapping ψ with $\mathrm{dom}(\psi) = \mathrm{names}(T_1, T_2)$, $[\![T_1]\!]_{\eta,\psi} = [\![T_2]\!]_{\eta,\psi}$.

In the next lemma, we justify that the random distributions $(\xleftarrow{R} [\![\tau]\!]_\eta)$ built in Section 3 are collision-free.

Lemma 6. *Assume a secure implementation. Then, for every type τ, the probability of collision for the distribution $(\xleftarrow{R} [\![\tau]\!]_\eta)$ is a negligible function of η.*

Proof. We prove the property by induction on τ.

The cases of $\tau = \mathit{Data}$ and $\tau = \mathit{Coins}$ are clear since $\alpha_1(\eta)$ and $\alpha_2(\eta)$ are at least linearly increasing. The case of $\tau = \mathit{Pair}[\tau_1, \tau_2]$ is clear as well, using the induction hypothesis on τ_1 and τ_2.

For $\tau \in \{\mathit{SKey}, \mathit{DKey}, \mathit{EKey}\}$, if the distribution $(\xleftarrow{R} [\![\tau]\!])$ has non-negligible probability of collision, then an adversary might obtain the secret key of the corresponding experiment using the key generation algorithm, with non-negligible probability.

Finally, assume that $\tau = SCipher[\tau_1]$ (the case of $\tau = ACipher[\tau_1]$ is similar), and that, by contradiction, $(\xleftarrow{R} [\![\tau]\!]_\eta)$ is not collision-free, that is:

$$\Pr\Big[e_1, e_2 \xleftarrow{R} [\![\tau_1]\!]_\eta; k_1, k_2 \xleftarrow{R} [\![SKey]\!]_\eta; r_1, r_2 \xleftarrow{R} [\![Coins]\!]_\eta \ : \\ \mathcal{E}^s(e_1, k_1, r_1) = \mathcal{E}^s(e_2, k_2, r_2)\Big]$$

is not negligible. Using the equational property of decryption, this implies that the following probability is not negligible either:

$$\Pr\Big[e_1, e_2 \xleftarrow{R} [\![\tau_1]\!]_\eta; k_1, k_2 \xleftarrow{R} [\![SKey]\!]_\eta; r_2 \xleftarrow{R} [\![Coins]\!]_\eta \ : \\ e_1 = \mathcal{D}^s(\mathcal{E}^s(e_2, k_2, r_2), k_1)\Big]$$

As the right-hand side of the equality test does not depend on e_1, we deduce in particular that $(\xleftarrow{R} [\![\tau_1]\!]_\eta)$ is not collision-free; this contradicts the induction hypothesis. □

Our last lemma is a reinforcement of the T_{enc}-Pwd criterion of Definition 3.

Lemma 7. *Assume that $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is T_{enc}-Pwd secure.*

Given a parameter η, some elements $k_1 \dots k_n \in \{0,1\}^{\alpha_1(\eta)}$, and $\ell_1 \dots \ell_n \in \{-1, 1\}$, we call vectorial key of size n *any expression $\vec{k} = k_1^{\ell_1} \cdot \ldots \cdot k_n^{\ell_n}$. We write $\mathcal{G}_\eta$ for the set of all vectorial keys.*

Given a bit-string e ($e \in [\![\tau]\!]_\eta$ for some $\tau \in T_{\mathsf{enc}}$), the expression $\mathcal{E}(e, \vec{k})$ stands for the result of the expected sequence of encryptions/decryptions over e using keys k_1, ..., k_n:

$$\begin{aligned} \mathcal{E}(e, \Lambda) &= e \\ \mathcal{E}(e, \vec{k} \cdot k^{+1}) &= \mathcal{E}(\mathcal{E}(e, \vec{k}), k) \\ \mathcal{E}(e, \vec{k} \cdot k^{-1}) &= \mathcal{D}(\mathcal{E}(e, \vec{k}), k) \end{aligned}$$

For every security parameter η, type $\tau \in T_{\mathsf{enc}}$, and positive integer M, consider the following experiment, with a two-stage PPTIME adversary $A = (A_1, A_2)$:

- *A_1 outputs a vectorial key $\vec{k} \in \mathcal{G}_\eta$ of size M and some state information st;*
- *a bit $b \xleftarrow{R} \{0,1\}$ is selected at random; if $b = 0$, we let $m \xleftarrow{R} [\![\tau]\!]_\eta$ and $c = \mathcal{E}(m, \vec{k})$; otherwise, $c \xleftarrow{R} [\![\tau]\!]_\eta$ is a random element from $[\![\tau]\!]_\eta$;*
- *A_2 is given c and st, and outputs a bit b'.*

The adversary A is successful if $b' = b$.

Then the advantage of A, defined by $\mathrm{Adv}^{\tau,M}_{Pwd,\Pi,A}(\eta) = \Pr[A \text{ is successful}] - \frac{1}{2}$, is negligible.

Proof. In the case $M = 0$, for any adversary A, we have $\mathrm{Adv}^{\tau,0}_{\mathrm{Pwd},\Pi,A}(\eta) = 0$.

Next we prove that for any (efficient) adversary $A = (A_1, A_2)$ in the game above for parameter $M+1$, there exist an (efficient) adversary $A' = (A'_1, A'_2)$ for parameter M, and an (efficient) adversary $B = (B_1, B_2)$ in the game of the T_{enc}-Pwd criterion (Definition 3) such that

$$|\,\mathrm{Adv}^{\tau,M+1}_{\mathrm{Pwd},\Pi,A}(\eta) - \mathrm{Adv}^{\tau,M}_{\mathrm{Pwd},\Pi,A'}(\eta)| \le \mathrm{Adv}^{\tau}_{\mathrm{Pwd},\Pi,B}(\eta)$$

(The result follows immediately by induction and the assumption of T_{enc}-Pwd security.)

We obtain A' and B by modifying A as follows. When A_1, the first stage of A, returns $\vec{k} = k_1^{\ell_1} \cdot \ldots \cdot k_{M+1}^{\ell_{M+1}}$, we let A'_1 return $\vec{k}' = k_2^{\ell_2} \cdot \ldots \cdot k_{M+1}^{\ell_{M+1}}$, and B_1 return k_1. In addition, we let A'_2 equal A_2, and define B_2 as follows:

- on input c and st, let $c' = \mathcal{E}(c, \vec{k}')$ if $\ell_1 = +1$, and $c' = \mathcal{E}(c, \vec{k})$ otherwise;
- let $s \in \{-1, +1\}$ be the sign of $\ell_1(\mathrm{Adv}^{\tau,M+1}_{\mathrm{Pwd},\Pi,A}(\eta) - \mathrm{Adv}^{\tau,M}_{\mathrm{Pwd},\Pi,A'}(\eta))$;
- if $s \geq 0$, B_2 returns the same value as A_2 on input c' and st; otherwise, it returns the opposite value.

Using the definitions of the different games, the advantage of B is then written:

$$\begin{aligned}
\mathrm{Adv}^{\tau}_{\mathrm{Pwd},\Pi,B}(\eta) &= \Pr[B \text{ returns } b' = b] - \frac{1}{2} \\
&= \frac{1}{2}(\Pr[B \text{ returns } b' = 0 \,/\, b = 0] - \Pr[B \text{ returns } b' = 0 \,/\, b = 1]) \\
&= \frac{s\ell_1}{2}(\Pr[A \text{ is successful} \,/\, b = 0] - \Pr[A' \text{ is successful} \,/\, b = 0]) \\
&= s\ell_1(\Pr[A \text{ is successful} \,\&\, b = 0] - \Pr[A' \text{ is successful} \,\&\, b = 0]) \\
&= s\ell_1(\Pr[A \text{ is successful}] - \Pr[A' \text{ is successful}]) \\
&= |\,\mathrm{Adv}^{\tau,M+1}_{\mathrm{Pwd},\Pi,A}(\eta) - \mathrm{Adv}^{\tau,M}_{\mathrm{Pwd},\Pi,A'}(\eta)|
\end{aligned}$$

where the sign / introduces conditional probabilities and the variables b and b' in each probability of success refer to the same variables in the corresponding game. □

We now prove the computational soundness of each transformation rule. As before, we write $\varepsilon = \varphi_1^0 \approx^? \varphi_2^0$ for the source equation of a rule and $\varepsilon' = \varphi_1^1 \approx^? \varphi_2^1$ for the resulting equation.

- **Undecipherable Encryption**. For any term T, let $|T|_e$ be the number of distinct subterms of T of the form $\mathsf{pub}(U)$, $\mathsf{enc}(U, V)$, $\mathsf{penc}(U, V, W)$, $\mathsf{senc}(U, V, W)$. The notation is extended to tuples of terms and frames.

  Given a type τ, we write $\mathrm{PPos}(\tau)$ for the set of positions p in τ such that for every proper prefix q of p, the symbol in position q in τ is a symbol $\tau(q) = \mathit{Pair}$. For every term of type τ and $p \in \mathrm{PPos}(\tau)$, we write $\pi_p(T)$ for the term of sort $\tau|_p$ defined by

$$\begin{aligned}
\pi_\Lambda(T) &= T \\
\pi_{1\cdot q}(T) &= \mathsf{fst}(\pi_q(T)) \\
\pi_{2\cdot q}(T) &= \mathsf{snd}(\pi_q(T))
\end{aligned}$$

  We prove the two following properties by mutual induction on a parameter $N \geq 0$:

  - $P(N)$—Within the conditions of rule **Undecipherable Encryption**, that is:
    - φ_1 is a well-formed frame;
    - k is a name whose every occurrence in φ (if any) is in key position;
    - T is a maximal subterm of φ_1 of the form $\mathsf{pub}(k)$, $\mathsf{penc}(U, k', r)$, $\mathsf{senc}(U, k, r)$, $\mathsf{enc}(U, k)$ or $\mathsf{dec}(U, k)$;
    - n is a fresh name;

    and if additionally $|\varphi_1|_e \leq N$, then $[\![\varphi_1]\!] \approx [\![\varphi_1\{T \mapsto n\}]\!]$;
  - $Q(N)$—For every well-formed frame $\varphi = \{x_1 = T_1, x_2 = T_2\}$ with $|\varphi|_e \leq N$, if T_1 and T_2 have the same type τ, contain none of the constants $\mathsf{w}, \mathsf{c}_0, \mathsf{c}_1 \ldots$, and have no subterms of the form $\mathsf{enc}(S, 0)$ or $\mathsf{enc}(S, 1)$, then for every $p \in \mathrm{PPos}(\tau)$, $\pi_p(T_1) \neq_E \pi_p(T_2)$ implies that the quantity

$$\Pr\left[e_1, e_2 \xleftarrow{R} [\![\pi_p(T_1), \pi_p(T_2)]\!]_\eta \;:\; e_1 = e_2\right]$$

    is a negligible function of η.

More precisely, we prove the four statements:

(1) $P(0)$,

(2) $P(N+1) \Leftarrow Q(N)$,

(3) $Q(0)$,

(4) $Q(N+1) \Leftarrow (P(N+1)$ and $Q(N))$.

(1) $P(0)$ is vacuously true.

(2) $P(N+1) \Leftarrow Q(N)$. Let $\varphi_1 = \{x_1 = T_1, \ldots, x_m = T_m\}$, and $\varphi_1' = \varphi_1\{T \mapsto n\}$. We distinguish several cases depending on the different forms of T.

- If $T = \mathsf{pub}(k)$, we have $[\![\varphi_1]\!] = [\![\varphi_1']\!]$ since by assumption k appears only in key position, that is, given the sort system, under the symbol pub.

- If $T = \mathsf{enc}(U, k)$, let τ_0 be the sort of U. Provided an adversary A able to distinguish between $[\![\varphi_1]\!]$ and $[\![\varphi_1']\!]$, we build an adversary B against the T_{enc}-security as follows:

  1. for each name $a \neq k$ of sort τ appearing in φ_1, draw a value $\widehat{a} \xleftarrow{R} [\![s]\!]_\eta$;
  2. for each x_i $(1 \leq i \leq m)$ of sort τ_i, compute $\widehat{T_i} \in [\![\tau_i]\!]_\eta$ recursively as follows:

$$\begin{aligned}
\widehat{\mathsf{enc}_\tau(T, k)} &= \mathcal{E}(\widehat{T}) \text{ if } T \neq U \\
\widehat{\mathsf{enc}_{\tau_0}(U, k)} &= \mathcal{E}^*(\widehat{U}) \\
\widehat{f(T_1, \ldots, T_n)} &= [\![f]\!]_\eta(\widehat{T_1}, \ldots, \widehat{T_n}) \\
&\qquad \text{if } f(T_1, \ldots, T_n) \notin \{\mathsf{enc}(T', k), \mathsf{dec}(T', k)\}
\end{aligned}$$

     where we have written $\mathcal{E}(.)$ for the encryption oracle of the T_{enc}-security game, and $\mathcal{E}^*(\widehat{U})$ for the answer to the challenge query $\widehat{U}$.
  3. submit the concrete frame $\{x_1 = \widehat{T_1}, \ldots, x_n = \widehat{T_n}\}$ to A and return the same answer.

  Note that by maximality T is not a subterm of an encryption with k, thus we may assume that B correctly sends the challenge query last. We also use the fact that k (whose concrete value is not known by B during the experiment) appears only as an encryption key in φ_1.

  The distribution computed by B and submitted to A equals either $[\![\varphi_1]\!]_\eta$ or $[\![\varphi_1']\!]_\eta$ depending on whichever $\mathcal{E}^*(\widehat{U})$ is, respectively, the encryption of $\widehat{U}$ or a random value in $[\![\tau]\!]_\eta$. Thus the probability that B guesses the right answer is the same as A. Nevertheless, B may not meet the extra condition for winning the T_{enc}-security game, that is: not to submit a plaintext previously submitted to the encryption oracle as the challenge plaintext. This happens if there exists a subterm $\mathsf{enc}_{n_0}(U', k)$ such that U and U' have the same type, $U' \neq U$ and $\widehat{U'} = \widehat{U}$.

  Suppose that this is the case. Let $\varphi = \{x = U, y = U'\}$. As U' and $T = \mathsf{enc}_{n_0}(U, k)$ are two subterms of φ_1, and T is not a subterm of U' (by maximality), we have $|\varphi|_e < |\varphi_1|_e = N + 1$. Besides, since φ_1 is well-formed, so is φ, and φ contains no subterm $\mathsf{enc}(T', c)$. Therefore, the induction hypothesis $Q(N)$ implies that the probability for $\widehat{U'} = \widehat{U}$ is negligible.

- If $T = \mathsf{penc}(U, k', r)$, or $T = \mathsf{senc}(U, k, r)$, provided an adversary A able to distinguish between $[\![\varphi_1]\!]$ and $[\![\varphi_1']\!]$, we build an adversary B against the T_{penc}-security (or respectively, the T_{senc}-security) in a similar way as above. We use here the fact that formal coins r appears in at most one encryption term. Note that there is no extra condition to check in these cases (in contrast with the arguments for deterministic encryption).

Before proving (3) and (4), note that $Q(N)$ is unchanged if we restrict the positions p to be the lowest positions in $\mathrm{PPos}(\tau)$, that is, those for which $\tau(p) \neq \mathit{Pair}$. Indeed, if T_1 and T_2 have a type $\mathit{Pair}[\tau_1, \tau_2]$, we have that, in the formal world,

$$T_1 =_E T_2 \text{ iff } \mathsf{fst}(T_1) =_E \mathsf{fst}(T_2) \text{ and } \mathsf{snd}(T_1) =_E \mathsf{snd}(T_2)$$

whereas computationally, $\Pr[e_1, e_2 \leftarrow [\![T_1, T_2]\!]_\eta \ : \ e_1 = e_2]$ is negligible iff

$$\begin{aligned} & \Pr[e_1, e_2 \leftarrow [\![\mathsf{fst}(T_1), \mathsf{fst}(T_2)]\!]_\eta \ : \ e_1 = e_2] \\ \text{and} \quad & \Pr[e_1, e_2 \leftarrow [\![\mathsf{snd}(T_1), \mathsf{snd}(T_2)]\!]_\eta \ : \ e_1 = e_2] \end{aligned}$$

are negligible. Our claim follows by induction.

We now prove (4). (Property (3) is a subcase.) Let $\varphi = \{x_1 = T_1, x_2 = T_2\}$ be a well-formed frame such that $|\varphi|_e \leq N + 1$, T_1, T_2 have a common type τ_0, contain none of the constants $\mathsf{w}, \mathsf{c}_0, \mathsf{c}_1 \ldots$, and have no subterms of the form $\mathsf{enc}(S, 0)$ or $\mathsf{enc}(S, 1)$. Let $p_0 \in \mathrm{PPos}(\tau_0)$ and assume $\pi_{p_0}(T_1) \neq_E \pi_{p_0}(T_2)$.

Let $T_1' = \pi_{p_0}(T_1)\!\downarrow_{\mathcal{R}}$, $T_2' = \pi_{p_0}(T_2)\!\downarrow_{\mathcal{R}}$ and τ be the sort of T_1' and T_2'. By the previous discussion, we may assume that p_0 is maximal in $\mathrm{PPos}(\tau_0)$, that is, τ is not a type $\mathit{Pair}_{\tau_1,\tau_2}$. Therefore τ is one of the following: *SKey*, *EKey*, *DKey*, *Data*, *Coins*, $\mathit{SCipher}[\tau']$, or $\mathit{ACipher}[\tau']$.

Given that T_1' and T_2' are $\mathcal{R}$-reduced, φ is well-formed, T_1' and T_2' (which are subterms of T_1 and T_2 respectively) contain none of the constants $\mathsf{w}, \mathsf{c}_0, \mathsf{c}_1 \ldots$ and have no subterms of the form $\mathsf{enc}(S, 0)$ or $\mathsf{enc}(S, 1)$, and we have $T_{\mathsf{enc}} \cap \{\mathit{Pair}[\tau_1, \tau_2]\}_{\tau_1,\tau_2} = \emptyset$, this entails that T_1' and T_2' are each one of the following form:

(i) projection of name $\pi_p(a)$, (possibly $p = \Lambda$)

(ii) constant 0 or 1,

(iii) public-key $\mathsf{pub}(k)$,

(iv) deterministic ciphertext $\mathsf{enc}(U, k)$,

(v) probabilistic ciphertext, $\mathsf{penc}(U, k', r)$, $\mathsf{penc}(U, \mathsf{pub}(k), r)$, $\mathsf{senc}(U, k, r)$.

We distinguish several cases depending on the form of T_1' and T_2'.

- If T_1' and T_2' are both of the form (i)–(iii), the result is clear by Lemma 6.
- If exactly one of these two terms matches (i)–(iii), say T_1', then we distinguish again two cases:
  - If T_2' is a ciphertext of the form $\mathsf{enc}(U, k)$, $\mathsf{penc}(U, \mathsf{pub}(k), r)$, or $\mathsf{senc}(U, k, r)$, and $T_1' = k$, then the probability
    $$\Pr\left[e_1, e_2 \xleftarrow{R} [\![T_1', T_2']\!]_\eta \ : \ e_1 = e_2\right]$$
    must be negligible, otherwise the security of the encryption scheme is easily broken since the key k can be recovered during the experiment. (We compute the concrete value of T_2' similarly as before, using the fact that k appears only in key position in U.)
  - Otherwise, either
    (i) $T_2' = \mathsf{penc}(U, k', r)$, or
    (ii) T_2' is of the form $\mathsf{enc}(U, k)$, $\mathsf{penc}(U, \mathsf{pub}(k), r)$, or $\mathsf{senc}(U, k, r)$, and either $T_1' = \mathsf{pub}(k)$ or k does not occur in T_1' at all.

    If the probability of collision above is not negligible, then computing T_1' gives a way to predict the value of the ciphertext T_2' during the experiment. Again, this breaks the security of encryption.

– Finally, if both T_1' and T_2' are of the form (iv) or (v), then since φ is well-formed, we may assume for instance that T_1' is of the form $\mathsf{enc}(U,k)$, $\mathsf{penc}(U,\mathsf{pub}(k),r)$, or $\mathsf{senc}(U,k,r)$ (depending on the sort of k), where the key k appears only in key position in T_2' and U. Let T be a maximal subterm of T_1' and T_2' of the form $\mathsf{enc}(U',k)$, $\mathsf{penc}(U',\mathsf{pub}(k),r')$, or $\mathsf{senc}(U',k,r')$.

Let n be a fresh name, and y_1, y_2 two variables all of the same sort as T_1'. Let $\varphi' = \{y_1 = T_1',\ y_2 = T_2'\}$. As φ is well-formed, and given the form of T_1' and T_2', φ' and T satisfy the conditions of $P(N+1)$. We deduce that the probability of collision between T_1' and T_2' is close to that between $T_1'' = T_1'\{T \mapsto n\}$ and $T_2'' = T_2'\{T \mapsto n\}$ except for a negligible difference. We conclude by applying $Q(N)$ on the resulting frame $\varphi'' = \{y_1 = T_1'',\ y_2 = T_2''\}$.

- **Split Names**. We have $[\![\varphi_1^0]\!] = [\![\varphi_1^1]\!]$ by definition of random values of type $Pair[\tau_1,\tau_2]$.

- **Pair Analysis**. Given the semantics of pairs and Lemma 5, it is straightforward to build an (efficient) adversary against $[\![\varphi_1^1]\!] \approx [\![\varphi_2^1]\!]$ provided an (efficient) adversary against $[\![\varphi_1^0]\!] \approx [\![\varphi_2^0]\!]$ (and conversely).

- **Redundancy Analysis-2**. Soundness of this rule is vacuously true.

- **Redundancy Analysis-1**. Given an adversary A against $[\![\varphi_1^0]\!] \approx [\![\varphi_2^0]\!]$, we build an adversary B against $[\![\varphi_1]\!] \approx [\![\varphi_2]\!]$. Indeed, the relation $x\varphi_i^0 =_E M\varphi_i$ holds also concretely by Lemma 5. Given a drawing ϕ of the frame φ_i, B feeds A with $\phi^0 = \phi \cup \{x = [\![\{P\}_{\vec{M}}]\!]_{\eta,\phi}\}$, and returns the same result as A.

- **Encryption Analysis-2**. Soundness of this rule is vacuously true.

- **Encryption Analysis-1**. We consider only the case of public encryption as the other one is similar.

  Given an adversary A against $[\![\varphi_1^0]\!] \approx [\![\varphi_2^0]\!]$, we build an adversary B against $[\![\varphi_1^1]\!] \approx [\![\varphi_2^1]\!]$ as follows. Let r_0 be a fresh name of sort *Coins* and ρ_i be the renaming $\rho_i = \{r_i \mapsto r_0\}$ as before. By Lemma 5, the relation

  $$x\varphi_i^0\rho_i =_E \{\mathsf{penc}(y,\mathsf{pub}(N),r_0)\}_{\vec{M}}\varphi_i^1$$

  (which relies the condition on the coins r_i) holds for the concrete terms as well. Besides, φ_i^0 and $\varphi_i^0\rho_i$ clearly have the same semantics. Given a drawing ϕ^1 of the frame φ_i^1, B feeds A with

  $$\phi^0 = \phi^1 \uplus \{x = [\![\{\mathsf{penc}(y,\mathsf{pub}(N),r_0)\}_{\vec{M}}]\!]_{\eta,\phi^1}\}|_{\mathrm{dom}(\varphi^0)},$$

  and return the same result as A.

- **Standardize**. We prove that $[\![\varphi_1^0]\!] \approx [\![\varphi_1^1]\!]$. Indeed, we have $\varphi_1^1 =_E \varphi_1^0\{a \mapsto \{a'\}_{\vec{C}[a_1\ldots a_n]^{-1}}\}$ where a' is fresh. Let τ_i be the sort of a_i. By Lemma 5, this implies

  $$\begin{aligned}[\![\varphi_1^1]\!]_\eta &= [\![\varphi_1^0\{a \mapsto \{a'\}_{\vec{C}[a_1\ldots a_n]^{-1}}\}]\!]_\eta \\ &= [\![\varphi_1^0]\!]_{\eta,\, e_i \xleftarrow{R} [\![\tau_i]\!]\,;\, a_i \mapsto e_i, a \mapsto [\![\{a'\}_{\vec{C}[a_1\ldots a_n]^{-1}}]\!]_{\eta, a_i \mapsto e_i}}\end{aligned}$$

  From Lemma 7, we deduce that $[\![\varphi_1^0]\!] \approx [\![\varphi_1^1]\!]$.

- **Solve**. We use a notion of ideal semantics inspired by [9]. For each frame $\varphi_i = \{x_1^i = a_1^i, \ldots x_{m_i}^i = a_{m_i}^i, y_1^i = C_1^i[a_1^i \ldots a_{m_i}^i], \ldots y_{n_i}^i = C_{n_1}^i[a_1^i \ldots a_{m_i}^i]\}$, consider the alternative concrete semantics $[\![\varphi_i]\!]_\eta{}'$ defined as follows:

– let τ_j be the sort of x^i and τ'_k the sort of y^i;
– for every concrete frame $\phi = \{x^i_j = e_j, y^i_k = f_k\}_{1\leq j\leq m_i, 1\leq k\leq n_i}$, the probability of sampling ϕ from $[\![\varphi_i]\!]_\eta{}'$ is the probability that for all j, k, $e_j \xleftarrow{R} [\![\tau_i]\!]_\eta$ and $f_k \xleftarrow{R} [\![\tau'_k]\!]_\eta$, conditioned to the fact that (a) for all k, the equation $y^i_k =^? C^i_k[x^i_1 \dots x^i_{m_i}]$ is satisfied concretely, that is, $f_k = [\![C^i_k[x^i_1 \dots x^i_{m_i}]\!]_{\eta, x^i_j \mapsto e_j}$. (If it is not the case, the probability of ϕ is zero).

By a simple reasoning on conditional probabilities, we have $[\![\varphi_i]\!]_\eta{}' = [\![\varphi_i]\!]_\eta$.

Assume that $\varepsilon \neq \bot$, that is: for all $j \in \{1 \dots n_1\}$, $y^1_j\varphi_2 =_E C^1_j[x^1_1 \dots x^1_{m_1}]\varphi_2$, and for all $k \in \{1 \dots n_2\}$, $y^2_k\varphi_1 =_E C^2_k[x^2_1 \dots x^2_{m_2}]\varphi_1$.

Given the form of φ_1 and φ_2, we know by Proposition 2 that the set of equations $\{y^1_j =^? C^1_j[x^1_1 \dots x^1_{m_1}]\}$ and $\{y^2_k =_E C^2_k[x^2_1 \dots x^2_{m_2}]\}$ yield equivalent equational theories (up to E, where the y^i_j are seen as free constants). By Lemma 5, we deduce that conditions (a) in the definitions of $[\![\varphi_1]\!]_\eta{}'$ and $[\![\varphi_2]\!]_\eta{}'$ are logically equivalent. Thus, $[\![\varphi_1]\!]_\eta{}' = [\![\varphi_2]\!]_\eta{}'$.

B Formal Results on Deducibility and Static Equivalence

This section is devoted to proving the formal properties of deducibility and static equivalence used in Appendix A. We first recall the characterization of deducibility due to Abadi and Cortier [2] on convergent subterm systems (see below), and apply it to our specific theory to deduce a number of useful lemmas. Then, we establish several local properties of static equivalence which allow step-by-step reasonings as used in Appendix A. We found that these properties hold for a much broader class of theories than convergent subterm ones. We state them in their full generality with possible future applications in mind. We expect that the techniques that we use could also be helpful in manual proofs of static equivalence, for theories that have not been automated yet or simply cannot be.

Definition and notations. A rewriting system $\mathcal{R}$ is *subterm* if for every rule $l \to r$ in $\mathcal{R}$, either r is a proper subterm of l or r is a $\mathcal{R}$-reduced ground term.

In the sequel, we always assume that $\text{names}(\mathcal{R}) = \emptyset$, so that the resulting equational theory E is stable by substitution of names.

Given a position p in a term T, the expression $T|_p$ denotes the subterm at position p in T; $T[p := T']$ stands for the term obtained by replacing this subterm with T' in T. We write $\text{st}(T)$ for the set of subterms of a term T.

Fact 8. *Let $\mathcal{R}$ be a subterm rewriting system. Then $\mathcal{R}$ is terminating.*

Proof. Let $\mu(T)$ denote the number of positions p in T such that $T|_p$ is not in $\mathcal{R}$-normal-form. We show that $T \to_{\mathcal{R}} T'$ implies $\mu(T) > \mu(T')$.

Indeed, let p be a position of T, $l \to r$ a rule of $\mathcal{R}$ and σ a substitution such that $T|_p = l\sigma$ and $T' = T[r\sigma]_p$. If $r\sigma = r$ is ground and $\mathcal{R}$-reduced, then $\mu(T') \leq \mu(T) - 1$. Otherwise, $r\sigma$ being a proper subterm of $l\sigma$ we have $\mu(T') \leq \mu(T) - 1$ as well. □

B.1 Inductive Characterization of Deducibility

We now characterize our equation-based notion of deducibility in terms of deduction rules (as the one used in many Dolev-Yao models). Most properties are classical so we keep most proofs informal for the sake of brevity.

Proposition 9. *Let E be an equational theory generated by a subterm convergent rewriting system $\mathcal{R}$. Let φ be a frame in $\mathcal{R}$-normal form. Let $\text{sat}(\varphi)$ be the smallest set such that*

(i) for every $x \in \text{dom}(\varphi)$, $x\varphi \in \text{sat}(\varphi)$;

(ii) for every $t \in \mathrm{st}(\varphi)$*, if* $t = f(t_1, \ldots, t_n)$ *and* $t_1, \ldots, t_n \in \mathrm{sat}(\varphi)$*, then* $t \in \mathrm{sat}(\varphi)$*;*

(iii) for every rule $l \to r$ *in* $\mathcal{R}$*, if there exist a plain context* C*, some terms* $t_1, \ldots, t_n \in \mathrm{sat}(\varphi)$*, a substitution* σ*, and some names* $a_1 \ldots a_m \notin \mathrm{names}(\varphi)$ *such that*

- l *is of the form* $l = C[l_1, \ldots, l_n, z_1, \ldots, z_m]$ *where the variables* z_1*, ...,* z_m *do not occur in* $l_1, \ldots, l_m$*,*
- $l\sigma = C[t_1, \ldots, t_n, a_1 \ldots a_m]$ *(that is, for every* i, j*,* $l_i\sigma = t_i$ *and* $z_j\sigma = a_j$*), and finally*
- $r\sigma \in \mathrm{st}(\varphi)$*,*

then $r\sigma \in \mathrm{sat}(\varphi)$*.*

A $\mathcal{R}$*-reduced term* T *is deducible from* φ *iff there exists a plain context* C*, some terms* $t_1, \ldots, t_n \in \mathrm{sat}(\varphi)$*such that* $T = C[t_1, \ldots, t_n]$*.*

A very similar result is stated in the work of Abadi and Cortier [2] concerning static equivalence. Below, we adapt the original proof to make more precise the set of contexts C to be considered. (We also use a slighty more general notion of subterm theory, and allow an infinite, sorted rewriting system.)

Proof of Proposition 9. Terms in $\mathrm{sat}(\varphi)$ are shown deducible by induction on the rules (i)–(iii); this implies the right-to-left implication.

As for the converse, by extending φ with components from $\mathrm{sat}(\varphi)$, we may assume without loss of generality that φ is *saturated*, that is, for every $T \in \mathrm{sat}(\varphi)$, there exists $x \in \mathrm{dom}(\varphi)$ such that $x\varphi = T$.

As T is $\mathcal{R}$-reduced and $\mathcal{R}$ is convergent, $\varphi \vdash_E T$ means that there exists M such that $\mathrm{var}(M) \subseteq \varphi$, $\mathrm{names}(M) \cap \mathrm{names}(\varphi, T) = \emptyset$ and $M\varphi \to_{\mathcal{R}}^* T$.

We prove the existence of a term M_0 that moreover satisfies $M_0\varphi = T$, by induction on $M\varphi$ with respect to the terminating quasi-ordering $\to_{\mathcal{R}}$.

Indeed, assume that $M\varphi \to_{\mathcal{R}}^* T$ and $M\varphi$ is reducible: there exists a position p, a rule $l \to r$ in $\mathcal{R}$, and a substitution σ such that $M\varphi|_p = l\sigma$. By confluence of $\mathcal{R}$, we have $M\varphi \to_{\mathcal{R}} M\varphi[r\sigma]_p \to_{\mathcal{R}}^* T$.

Next, we build a M' such that $M\varphi[r\sigma]_p = M'\varphi$ and conclude the proof by induction. Indeed, as φ is $\mathcal{R}$-reduced and $M\varphi|_p = l\sigma$, p must be the position of a function symbol in M, notably, $M|_p\varphi = l\sigma$.

Considering the greatest common term between $M|_p$ and l, and since $\mathrm{names}(l) = \emptyset$, we find that there exists a plain context C such that

- $M|_p = C[x_1, \ldots, x_p, M_1, \ldots, M_q, N_1, \ldots, N_m]$,
- $l = C[l_1, \ldots, l_p, y_1, \ldots, y_q, z_1, \ldots, z_m]$,

where $y_1, \ldots y_q \in \mathrm{var}(l_1, \ldots, l_p)$, and $z_1, \ldots z_m \notin \mathrm{var}(l_1, \ldots, l_p)$.

The equation $M|_p\varphi = l\sigma$ then implies that

- $l_i\sigma = x_i\varphi$ is in $\mathrm{sat}(\varphi)$ by rule (i) in the definition of $\mathrm{sat}(\varphi)$;
- $y_i\sigma = M_i\varphi \in \mathrm{st}(l_i\sigma)$ (since $y_i \in \mathrm{var}(l_1, \ldots, l_p)$) is in $\mathrm{sat}(\varphi)$ by rule (ii).

By definition, as $\mathcal{R}$ is a subterm rewriting system, r is a proper subterm of l or is ground and $\mathcal{R}$-reduced. Hence $r\sigma$ is either a (proper) subterm of some $l_i\sigma$, or can be written $r\sigma = C'[l_1, \ldots, l_p, y_1, \ldots, y_q, z_1, \ldots, z_m]\sigma$ for some context C' (take $C' = r$ if r is ground and $\mathcal{R}$-reduced).

In the first case, since $z_1, \ldots z_m \notin \mathrm{var}(l_1, \ldots, l_p)$, letting a_1, ..., a_m be fresh names of the appropriate sorts, we have

$$C[l_1\sigma, \ldots, l_p\sigma, y_1\sigma, \ldots, y_q\sigma, a_1, \ldots, a_m] = l\sigma' \to_{\mathcal{R}} r\sigma' = r\sigma \in \mathrm{st}(\mathrm{sat}(\varphi))$$

therefore $r\sigma \in \mathrm{sat}(\varphi)$ by rule (iii). By assumption, there exists x such that $x\varphi = r\sigma$; we let $M' = M[x]_p$.

In the second case, we let

$$M' = C[p := C'[x_1, \ldots, x_p, M_1, \ldots, M_q, N_1, \ldots, N_m]]$$

In both cases, we obtain $M\varphi \to_{\mathcal{R}} M'\varphi$ which concludes our proof by induction. □

Note that, when $\mathcal{R}$ is a convergent subterm system but is infinite, the above characterization does not entail immediately that deducibility is decidable in polynomial time (cf. [2]). Indeed, during the inductive computation of $\mathrm{sat}(\varphi)$, condition (iii) involves a possibly unbounded number of rules $l \to r$.

We focus on the equational theory E defined in the main part of the paper:

$$\begin{array}{rcl@{\qquad}rcl}
\mathsf{dec}_\tau(\mathsf{enc}_\tau(x,y),y) & = & x & \mathsf{enc}_\tau(\mathsf{dec}_\tau(x,y),y) & = & x \\
\mathsf{pdec}_\tau(\mathsf{penc}_\tau(x,\mathsf{pub}(y),z),y) & = & x & \mathsf{pdec_success}_\tau(\mathsf{penc}_\tau(x,\mathsf{pub}(y),z),y) & = & 1 \\
\mathsf{sdec}_\tau(\mathsf{senc}_\tau(x,y,z),y) & = & x & \mathsf{sdec_success}_\tau(\mathsf{senc}_\tau(x,y,z),y) & = & 1 \\
\mathsf{fst}_{\tau_1,\tau_2}(\mathsf{pair}_{\tau_1,\tau_2}(x,y)) & = & x & \mathsf{snd}_{\tau_1,\tau_2}(\mathsf{pair}_{\tau_1,\tau_2}(x,y)) & = & y \\
\mathsf{pair}_{\tau_1,\tau_2}(\mathsf{fst}_{\tau_1,\tau_2}(x),\mathsf{snd}_{\tau_1,\tau_2}(x)) & = & x & & &
\end{array}$$

Applied to this theory, Proposition 9 entails the following characterization.

Corollary 10. *Let φ be a frame in $\mathcal{R}$-normal form. Let $\mathrm{sat}(\varphi)$ be the smallest set such that*

(i) for every $x \in \mathrm{dom}(\varphi)$, $x\varphi \in \mathrm{sat}(\varphi)$;

(ii) for every $t \in \mathrm{st}(\varphi)$, if $t = f(t_1,\ldots,t_n)$ and $t_1,\ldots,t_n \in \mathrm{sat}(\varphi)$, then $t \in \mathrm{sat}(\varphi)$;

(iii) the following rules are satisfied: (omitting type annotations)

$$\begin{array}{rcl}
\mathsf{pair}(t_1,t_2) \in \mathrm{sat}(\varphi) & \Rightarrow & t_1 \in \mathrm{sat}(\varphi) \\
\mathsf{pair}(t_1,t_2) \in \mathrm{sat}(\varphi) & \Rightarrow & t_2 \in \mathrm{sat}(\varphi) \\
\mathsf{fst}(t) \in \mathrm{sat}(\varphi),\ \mathsf{snd}(t) \in \mathrm{sat}(\varphi) \text{ and } t \in \mathrm{st}(\varphi) & \Rightarrow & t \in \mathrm{sat}(\varphi) \\
\mathsf{enc}(t_1,t_2) \in \mathrm{sat}(\varphi) \text{ and } t_2 \in \mathrm{sat}(\varphi) & \Rightarrow & t_1 \in \mathrm{sat}(\varphi) \\
\mathsf{dec}(t_1,t_2) \in \mathrm{sat}(\varphi) \text{ and } t_2 \in \mathrm{sat}(\varphi) & \Rightarrow & t_1 \in \mathrm{sat}(\varphi) \\
\mathsf{penc}(t_1,\mathsf{pub}(t_2),t_3) \in \mathrm{sat}(\varphi) \text{ and } t_2 \in \mathrm{sat}(\varphi) & \Rightarrow & t_1 \in \mathrm{sat}(\varphi) \\
\mathsf{senc}(t_1,t_2,t_3) \in \mathrm{sat}(\varphi) \text{ and } t_2 \in \mathrm{sat}(\varphi) & \Rightarrow & t_1 \in \mathrm{sat}(\varphi)
\end{array}$$

A $\mathcal{R}$-reduced term T is deducible from φ iff there exists a plain context C, some terms $t_1,\ldots,t_n \in \mathrm{sat}(\varphi)$ such that $T = C[t_1,\ldots,t_n]$.

Here we make precise the instantiations of point (iii) of Proposition 9, by case analysis on the rules of $\mathcal{R}$. This essentially yields the classical Dolev-Yao rules for deduction. We have omitted the cases that leave $\mathrm{sat}(\varphi)$ trivially unchanged, such as:

$$t_1, t_2 \in \mathrm{sat}(\varphi) \text{ implies } t_1 \in \mathrm{sat}(\varphi) \text{ by } \mathsf{dec}(\mathsf{enc}(t_1,t_2),t_1) \to_{\mathcal{R}} t_1$$

as well as those redundant with point (ii), for instance:

$$\mathsf{penc}(t_1,\mathsf{pub}(t_2),t_3) \in \mathrm{sat}(\varphi) \text{ and } t_2 \in \mathrm{sat}(\varphi) \text{ implies } 1 \in \mathrm{sat}(\varphi)$$

Note that $\mathrm{sat}(\varphi)$ is still included in $\mathrm{st}(\varphi)$. As Corollary 10 describes $\mathrm{sat}(\varphi)$ by a fixed number of inference rules and the size of $\mathrm{sat}(\varphi)$ is bounded by the DAG-size of the problem, this provides a polynomial-time algorithm to decide deducibility in E (as in [2]). Besides, the algorithm can easily produce a *recipe* in case of a positive answer for $\varphi \vdash^?_E T$, that is, a term M such that $\mathrm{var}(M) \subseteq \varphi$, $\mathrm{names}(M) \cap \mathrm{names}(\varphi) = \emptyset$ and $M\varphi =_E T$. Indeed, the procedure need simply maintain a table associating a recipe to each deducible subterm already found. When a new deducible subterm is discovered, its recipe is deduced from the applied inference rule in the obvious way.

As the main proof of Appendix A concerns mostly well-formed frames, we finally introduce another variant of Proposition 9, specialized to well-formed frames.

Corollary 11. *Let φ be a well-formed frame. Let $\mathrm{sat}'(\varphi)$ be the smallest set such that*

(i) for every $x \in \mathrm{dom}(\varphi)$, $x\varphi \in \mathrm{sat}'(\varphi)$;

(ii) the following rules are satisfied: (omitting type annotations)

$$\begin{aligned}
\mathsf{pair}(t_1,t_2) \in \mathrm{sat}'(\varphi) &\Rightarrow t_1 \in \mathrm{sat}'(\varphi)\\
\mathsf{pair}(t_1,t_2) \in \mathrm{sat}'(\varphi) &\Rightarrow t_2 \in \mathrm{sat}'(\varphi)\\
\mathsf{enc}(t_1,t_2) \in \mathrm{sat}'(\varphi) \text{ and } t_2 \in \mathrm{sat}'(\varphi) &\Rightarrow t_1 \in \mathrm{sat}'(\varphi)\\
\mathsf{penc}(t_1,\mathsf{pub}(t_2),t_3) \in \mathrm{sat}'(\varphi) \text{ and } t_2 \in \mathrm{sat}'(\varphi) &\Rightarrow t_1 \in \mathrm{sat}'(\varphi)\\
\mathsf{senc}(t_1,t_2,t_3) \in \mathrm{sat}'(\varphi) \text{ and } t_2 \in \mathrm{sat}'(\varphi) &\Rightarrow t_1 \in \mathrm{sat}'(\varphi)
\end{aligned}$$

A $\mathcal{R}$-reduced term T is deducible from φ iff there exists a plain context C, some terms $t_1,\ldots,t_n \in \mathrm{sat}'(\varphi)$ such that $T = C[t_1,\ldots,t_n]$.

Indeed, since decryption keys of well-formed frames are always names, the inference rule (ii) of Corollary 10 is never applied to find such keys, so we may postpone this operation to the end. Besides, well-formed frames do not contain dec, fst, and snd symbols, so the corresponding inference rules (third and sixth of previous point (iii)) are useless here. In the end, we obtain a set of inference rules very similar to the ones of [5] and [6].

Some useful lemmas. Using this characterization of deducibility, we now state several technical lemmas used in the main proof of Appendix A.

Given a frame φ and an expression $\vec{V} = V_1^{\ell_1} \cdot \ldots \cdot V_n^{\ell_n}$, we write $\varphi \vdash_E \vec{V}$ iff for every i, $\varphi \vdash_E V_i$.

Lemma 12. *Let $\varphi_0 = \varphi \uplus \{x = \{U\}_{\vec{V}}\}$ be a well-formed frame.*

(1) We have $\varphi_0 \vdash_E \vec{V}$ iff $\varphi \vdash_E \vec{V}$.

(2) Assume $\varphi \vdash_E \vec{V}$. For any term T, $\varphi_0 \vdash_E T$ iff $\varphi \uplus \{x = U\} \vdash_E T$.

Proof. (1) Suppose $\varphi \vdash_E \vec{V}$, that is, there exists $\vec{M}$ such that $\mathrm{var}(\vec{M}) \subseteq \mathrm{dom}(\varphi)$, $\mathrm{names}(M) \cap \mathrm{names}(\varphi, T) = \emptyset$, and $\vec{M}\varphi =_E T$. By renaming names in $\mathrm{names}(\vec{M}) - \mathrm{names}(\varphi, T)$, we may assume without loss of generality that $\mathrm{names}(\vec{M}) \cap \mathrm{names}(\varphi_0) = \emptyset$. Thus, $\vec{M}$ shows that $\varphi_0 \vdash \vec{V}$.

Conversely, assume that $\varphi \nvdash_E \vec{V}$. Let $\vec{V} = a_1^{\ell_1} \cdots a_m^{\ell_m}$ (by well-formedness). There exists a maximal $i \geq 1$ such that $\varphi \nvdash_E a_i$. Using Corollary 11, we have that

$$\mathrm{sat}'(\varphi_0) = \mathrm{sat}'(\varphi) \cup \left\{ \{U\}_{\vec{V}'} \mid \vec{V}' = a_1^{\ell_1} \cdots a_j^{\ell_j}, j \geq i \right\}$$

In particular, since $a_i \notin \mathrm{sat}'(\varphi)$, we have $\varphi_0 \nvdash_E \vec{V}$.

(2) Let $\vec{N}\varphi =_E \vec{V}$ with $\mathrm{var}(\vec{N}) \subseteq \mathrm{dom}(\varphi)$ and $\mathrm{names}(N) \cap \mathrm{names}(\varphi_0, T) = \emptyset$.

Suppose $M\varphi_0 =_E T$ with $\mathrm{var}(M) \subseteq \mathrm{dom}(\varphi_0)$ and $\mathrm{names}(M) \cap \mathrm{names}(\varphi_0, T) = \emptyset$. Then we have $M\{x \mapsto \{x\}_{\vec{N}}\}\varphi =_E T$, thus $\varphi \uplus \{x = U\} \vdash_E T$.

Conversely, suppose $M(\varphi \cup \{x = U\}) =_E T$ with $\mathrm{var}(M) \subseteq \mathrm{dom}(\varphi_0)$ and $\mathrm{names}(M) \cap \mathrm{names}(\varphi_0, T) = \emptyset$. Then we have $M\{x \mapsto \{x\}_{\vec{N}^{-1}}\}\varphi =_E T$, thus $\varphi_0 \vdash_E T$.

□

Lemma 13. *Let $\varphi_0 = \varphi \uplus \{x = \mathsf{penc}(U, \mathsf{pub}(V), r)\}$ be a well-formed frame (respectively $\varphi_0 = \varphi \uplus \{x = \mathsf{senc}(U, V, r)\}$).*

(1) We have $\varphi_0 \vdash_E V$ iff $\varphi \vdash_E V$.

(2) Assume $\varphi \vdash_E V$ and r does not occur in φ. For any term T that does not contain r, $\varphi_0 \vdash_E T$ iff $\varphi \uplus \{y = U\} \vdash_E T$.

Proof. We consider only the case of public-key encryption as that of symmetric encryption is similar.

(1) Similar to Lemma 12.

(2) Let $N\varphi =_E V$ with $\mathrm{var}(N) \subseteq \mathrm{dom}(\varphi)$ and $\mathrm{names}(N) \cap \mathrm{names}(\varphi_0, V) = \emptyset$.

Suppose $M\varphi_0 =_E T$ with $\mathrm{var}(M) \subseteq \mathrm{dom}(\varphi_0)$ and $\mathrm{names}(M) \cap \mathrm{names}(\varphi_0, T) = \emptyset$. Let r' be a fresh name of sort *Coins*. Then we have $M\{x \mapsto \{y\}_N^{r'}\}(\varphi \uplus \{y = U\}) =_E T$, thus $\varphi \uplus \{y = U\} \vdash_E T$.

Conversely, suppose $M(\varphi \uplus \{y = U\}) =_E T$ with $\mathrm{var}(M) \subseteq \mathrm{dom}(\varphi_0)$ and $\mathrm{names}(M) \cap \mathrm{names}(\varphi_0, T) = \emptyset$. Then we have $M\{x \mapsto \mathsf{pdec}(x, N)\}\varphi =_E T$, thus $\varphi_0 \vdash_E T$.

□

We say that a position p is *under an encryption by a non-deducible key* in φ if there exists a variable $x \in \mathrm{dom}(\varphi)$ and a prefix $p' \cdot 1$ of p such that $x\varphi|_{p'}$ is of the form

- $\mathsf{enc}(U, V)$ with $\varphi \nvdash_E V$,
- $\mathsf{senc}(U, V, W)$ with $\varphi \nvdash_E V$, or
- $\mathsf{penc}(U, V', W)$ with $\forall V,\ V' =_E \mathsf{pub}(V) \Rightarrow \varphi \nvdash_E V$.

By $p' \cdot 1$, we mean that p goes into the first argument U of the encryption. Note that "non-deducible key" refers here to the decryption key.

Lemma 14. *Let φ be a well-formed frame, and $r \in \mathrm{names}(\varphi)$ of sort Coins. Then, r is not deducible from φ.*

Proof. Since φ is well-formed, r appears only as a third argument of an encryption symbol. By inspection of the rules of Corollary 11, we have that $r \notin \mathrm{sat}'(\varphi)$, thus r is not deducible. □

Lemma 15. *Let φ be a well-formed frame, and T a subterm of φ sort $\tau \neq$ Coins. Then, T is deducible from φ iff either (i) there exists a symbol f such that $T = f(T_1, \ldots, T_n)$ and all the T_i are deducible, or (ii) T appears in some position p of φ that is not a key position and not under an encryption by a non-deducible key.*

Proof. Thanks to Corollary 11, it is sufficient to verify that (ii) holds iff $T \in \mathrm{sat}'(\varphi)$.

We prove the implication $T \in \mathrm{sat}'(\varphi) \Rightarrow$ (ii) by induction on the proof tree for $T \in \mathrm{sat}'(\varphi)$.

- If point (i) of Corollary 11 applies (axiom rule), then T appears in a root position, which satisfies (ii).
- If a rule $\mathsf{pair}(t_1, t_2) \in \mathrm{sat}'(\varphi) \Rightarrow t_i \in \mathrm{sat}'(\varphi)$ applies, with $t_i = T$, then the induction hypothesis applies on $\mathsf{pair}(t_1, t_2)$ which entails (ii) on T.
- If rule $\mathsf{enc}(t_1, t_2) \in \mathrm{sat}'(\varphi)$ and $t_2 \in \mathrm{sat}'(\varphi) \Rightarrow t_1 \in \mathrm{sat}'(\varphi)$ applies with $t_1 = T$, then the induction hypothesis applies on $\mathsf{enc}(t_1, t_2)$ which entails (ii) on T since t_2 is deducible.
- (The cases of the remaining rules are similar.)

As for the converse implication, we prove it by induction on the position p of T in φ that is given by (ii). Indeed, if p is a root position in φ, the axiom rule (i) of Corollary 11 applies. Suppose $p = p' \cdot i$. Since φ is well-formed, T has sort $\tau \neq$ *Coins*, and (ii) holds, we have that

- the symbol in position p' of φ is either pair, enc, penc, or senc;
- in the last three case, T is the first argument of the encryption and the second argument has a deducible decryption key;
- p' is neither a key position in φ nor under an encryption by a non-deducible key.

Applying the induction hypothesis on T' and the appropriate inference rule of Corollary 11, we obtain that T belongs to $\mathrm{sat}'(\varphi)$. □

B.2 Stepwise Reasoning for Static Equivalence

Assume an arbitrary equational theory E. For every term T_0 of the form $T_0 = f_0(U_1 \dots U_{n_0})$ and every variable x_0 of the same sort, we define a function $\mathrm{cut}^E_{T_0,x_0}$ recursively as follows:

$$\begin{array}{rcl} \mathrm{cut}^E_{T_0,x_0}(x) & = & x \\ \mathrm{cut}^E_{T_0,x_0}(f(T_1 \dots T_n)) & = & \left\{ \begin{array}{ll} x_0 & \text{if } f = f_0 \text{ and } \forall i, T_i =_E U_i, \\ f(\mathrm{cut}^E_{T_0,x_0}(T_1), \dots, \mathrm{cut}^E_{T_0,x_0}(T_n)) & \text{otherwise.} \end{array} \right. \end{array}$$

Thus, $\mathrm{cut}^E_{T_0,x_0}$ has the effect of replacing some subterms S of its argument by x_0. Note that not every subterm S equal to T_0 modulo E is replaced. This is crucial both for the proofs and the applications. If E is the syntactic equality, $\mathrm{cut}^E_{T_0,x_0}$ is simply the replacement that maps T_0 to x_0.

We now introduce a useful "interpolation" lemma.

Lemma 16. *Let Σ_1 and Σ_2 be two disjoint first-order signatures, and $\Sigma = \Sigma_1 \cup \Sigma_2$. Assume two rewriting systems $\mathcal{R}_1$ and $\mathcal{R}_2$ over Σ such that the following conditions hold:*

(i) $\mathcal{R}_1$ is subterm;

(ii) for every rule $l \to r$ in $\mathcal{R}_1$, the head symbol of l is in Σ_1;

(iii) rules in $\mathcal{R}_2$ are made of symbols in Σ_2 only;

(iv) $\mathcal{R}_1 \cup \mathcal{R}_2$ is confluent;

(v) either $\mathcal{R}_1 = \emptyset$, or for every rule $l \to r$ in $\mathcal{R}_2$, $\mathrm{var}(r) \subseteq \mathrm{var}(l)$.

We write E for the equational theory generated by $\mathcal{R}_1$ and $\mathcal{R}_2$, and let $\mathcal{R} = \mathcal{R}_1 \cup \mathcal{R}_2$.

Let $T_0 = f_0(U_1 \dots U_{n_0})$ be a term over Σ such that $f_0 \in \Sigma_1$, and, for every rule $l \to r$ in $\mathcal{R}_1$, if r is not a proper subterm of l (second type of subterm rule), then f_0 does not occur in r.

Let T_1 and T_2 be two terms such that

(a) $T_1 =_E T_2$,

(b) for every subterm $S = g(S_1 \dots S_n)$ of T_1 or T_2, for every rule $l \to r$ in $\mathcal{R}_1$ with $l = g(l_1 \dots l_n)$, for every substitution θ such that $\forall j, S_j \to^*_{\mathcal{R}} l_j\theta$, we have

$$\mathrm{cut}^E_{T_0,x_0}(l\theta) = l\,\mathrm{cut}^E_{T_0,x_0}(\theta),$$

that is, intuitively, $\mathrm{cut}^E_{T_0,x_0}$ does not operate at the positions inside l in $l\theta$.

Then, for any variable x_0 of the appropriate sort, it holds that

$$\mathrm{cut}^E_{T_0,x_0}(T_1) =_E \mathrm{cut}^E_{T_0,x_0}(T_2)$$

Note that in order to establish the equality $\mathrm{cut}^E_{T_0,x_0}(l\theta) = l\,\mathrm{cut}^E_{T_0,x_0}(\theta)$, it is sufficient to prove that for every subterm $t = f_0(t_1 \dots t_{n_0})$ of l, there exists i such that $U_i \neq_E t_i\theta$.

Let us give an example of application before proving Lemma 16.

Example 6. Let E be an arbitrary equational theory over Σ_2, and $\mathcal{R}_2$ a corresponding confluent rewriting system (obtained for instance by orienting the equations of E in both directions). Assume $\Sigma_1 = \{\mathsf{h}\}$ and $\mathcal{R}_1 = \emptyset$, that is, h is a *free symbol* with respect to E. Let $T_0 = \mathsf{h}(U_1 \dots U_n)$. Conditions (i)–(v) are clearly fulfilled. Since condition (b) is vacuous, it holds that, whenever $T_1 =_E T_2$, $\mathrm{cut}^E_{T_0,x_0}(T_1) =_E \mathrm{cut}^E_{T_0,x_0}(T_2)$.

This result is an important (in fact characteristic) property of free symbols. In particular, it implies that for every $U_1 \dots U_n, U'_1 \dots U'_n$, $\mathsf{h}(U_1 \dots U_n) =_E \mathsf{h}(U'_1 \dots U'_n)$ iff $\forall i$, $U_i =_E U'_i$.

Proof. Let us write cut for cut^E. Let $\mathcal{P}(T)$ be the following property:

for every subterm $S = g(S_1 \dots S_n)$ of T, for every rule $l \to r$ in $\mathcal{R}_1$ with $l = g(l_1 \dots l_n)$, for every substitution θ such that $\forall j, S_j \to^*_{\mathcal{R}} l_j\theta$, $\mathrm{cut}_{T_0,x_0}(l\theta) = l\,\mathrm{cut}_{T_0,x_0}(\theta)$.

Thus, condition (b) is equivalent to $\mathcal{P}(T_1)$ and $\mathcal{P}(T_2)$.

Since $T_1 =_E T_2$ and condition (iv) holds, there exists a chain

$$T_1 = T^0 \to^*_{\mathcal{R}_2} \to_{\mathcal{R}_1} \to^*_{\mathcal{R}_2} T^1 \dots \leftarrow^*_{\mathcal{R}_2} \leftarrow_{\mathcal{R}_1} \leftarrow^*_{\mathcal{R}_2} T^m = T_2$$

First, we show the propagation of $\mathcal{P}$ along the chain of equations between T_1 and T_2. More formally, we prove the following facts:

1. $\mathcal{P}(T)$ and $T \to_{\mathcal{R}_1} T'$ imply $\mathcal{P}(T')$;
2. $\mathcal{P}(T)$ and $T \to_{\mathcal{R}_2} T'$ imply $\mathcal{P}(T')$.

If $\mathcal{R}_1 = \emptyset$, then $\mathcal{P}(T)$ is vacuously true, so both implications hold. Otherwise, we proceed as follows:

1. Assume $\mathcal{P}(T)$ and $T \to_{\mathcal{R}_1} T'$. There exists a position p, a rule $l \to r \in \mathcal{R}_1$, and a substitution σ such that $T|_p = l\sigma$ and $T' = T[r\sigma]_p$.

   Assume a subterm $S = g(S_1 \dots S_n)$ of T', a rule $l' = g(l_1 \dots l_n) \to r' \in \mathcal{R}_1$, and a substitution θ such that $\forall i, S_i \to^*_{\mathcal{R}} l_i\theta$.

   We distinguish three cases.

   (a) If S is a subterm of $r\sigma$, then by condition (i), r is a proper subterm of l (otherwise, r is ground and $\mathcal{R}$-reduced and $S = l'\theta$ cannot be a subterm of $r\sigma = r$), hence S is a subterm of $l\sigma$, and we conclude directly by $\mathcal{P}(T)$.

   (b) If S is a subterm of $T'[\_]_p = T[\_]_p$, then $\mathcal{P}(T)$ applies as well.

   (c) Otherwise, S is written $S = T|_{q_1}[r\sigma]_{q_2}$ with $p = q_1q_2$ and $q_1 \neq \Lambda$. Thus, there exists $S'_1 \dots S'_n$ such that $T|_{q_1} = T|_{q_1}[l\sigma]_{q_2} = g(S'_1 \dots S'_n)$ and $\forall i, S'_i \to^*_{\mathcal{R}} S_i \to^*_{\mathcal{R}} l_i\theta$. Thus, we apply $\mathcal{P}(T)$ on $T|_{q_1} = g(S'_1 \dots S'_n)$, $l \to r$ and θ.

2. Assume $\mathcal{P}(T)$ and $T \to_{\mathcal{R}_2} T'$. There exists a position p, a rule $l \to r \in \mathcal{R}_2$, and a substitution σ such that $T|_p = l\sigma$ and $T' = T[r\sigma]_p$.

   Assume a subterm $S = g(S_1 \dots S_n)$ of T', a rule $l' = g(l_1 \dots l_n) \to r' \in \mathcal{R}_1$, and a substitution θ such that $\forall i, S_i \to^*_{\mathcal{R}} l_i\theta$.

   We distinguish three cases.

   (a) If S is a subterm of $r\sigma$, then by condition (ii), since $g \notin \Sigma_2$, S is a subterm of $\mathrm{var}(r)\sigma$. Thus, by condition (v), it is a subterm of $l\sigma$. We conclude by applying $\mathcal{P}(T)$.

   (The last two cases are done as above.)

From the two previous facts and the assumptions $\mathcal{P}(T_1)$ and $\mathcal{P}(T_2)$, we deduce that $\mathcal{P}$ is true on every intermediate term of the chain. Therefore, in order to finish the proof, it is sufficient to prove the following facts:

1. $\mathcal{P}(T)$ and $T \to_{\mathcal{R}_1} T'$ imply $\mathrm{cut}_{T_0,x_0}(T) =_E \mathrm{cut}_{T_0,x_0}(T')$;
2. $T \to_{\mathcal{R}_2} T'$ imply $\mathrm{cut}_{T_0,x_0}(T) =_E \mathrm{cut}_{T_0,x_0}(T')$.

We proceed as follows.

1. Assume $\mathcal{P}(T)$ and $T \to_{\mathcal{R}_1} T'$. There exists a position p, a rule $l \to r \in \mathcal{R}_1$, and a substitution σ such that $T|_p = l\sigma$ and $T' = T[r\sigma]_p$.

   If p is below the positions of the symbol f_0 where cut_{T_0,x_0} operates in T, then by definition of cut, since $T \xrightarrow{p}_{\mathcal{R}_1} T'$, we have

   $$\mathrm{cut}_{T_0,x_0}(T) = \mathrm{cut}_{T_0,x_0}(T').$$

   Otherwise, we have $\mathrm{cut}_{T_0,x_0}(T)|_p = \mathrm{cut}_{T_0,x_0}(T|_p) = \mathrm{cut}_{T_0,x_0}(l\sigma)$. By $\mathcal{P}(T)$, $\mathrm{cut}_{T_0,x_0}(l\sigma) = l\,\mathrm{cut}_{T_0,x_0}(\sigma)$. Similarly, as r is either a subterm of l or a $\mathcal{R}$-reduced ground term not containing f_0,

   $$\begin{array}{rcl} \mathrm{cut}_{T_0,x_0}(T') & = & \mathrm{cut}_{T_0,x_0}(T)[\mathrm{cut}_{T_0,x_0}(r\sigma)]p \\ & = & \mathrm{cut}_{T_0,x_0}(T)[r\,\mathrm{cut}_{T_0,x_0}(\sigma)]_p \end{array}$$

   Hence, $\mathrm{cut}_{T_0,x_0}(T) =_E \mathrm{cut}_{T_0,x_0}(T')$.

2. Assume $T \to_{\mathcal{R}_2} T'$. There exists a position p, a rule $l \to r \in \mathcal{R}_2$, and a substitution σ such that $T|_p = l\sigma$ and $T' = T[r\sigma]_p$.

   Again, if p is below the positions of the symbol f_0 where cut_{T_0,x_0} operates in T, then we have

   $$\mathrm{cut}_{T_0,x_0}(T) = \mathrm{cut}_{T_0,x_0}(T').$$

   Otherwise, $\mathrm{cut}_{T_0,x_0}(T)|_p = \mathrm{cut}_{T_0,x_0}(T|_p) = \mathrm{cut}_{T_0,x_0}(l\sigma)$. By condition (iii), and since $f_0 \notin \Sigma_2$, cut_{T_0,x_0} may not operate at a non-variable position of l in $S = l\sigma$. Thus, $\mathrm{cut}_{T_0,x_0}(l\sigma) = l\,\mathrm{cut}_{T_0,x_0}(\sigma)$.

   Similarly,

   $$\begin{array}{rcl} \mathrm{cut}_{T_0,x_0}(T') & = & \mathrm{cut}_{T_0,x_0}(T)[\mathrm{cut}_{T_0,x_0}(r\sigma)]p \\ & = & \mathrm{cut}_{T_0,x_0}(T)[r\,\mathrm{cut}_{T_0,x_0}(\sigma)]_p \end{array}$$

   Hence, $\mathrm{cut}_{T_0,x_0}(T) =_E \mathrm{cut}_{T_0,x_0}(T')$.

□

Proposition 17. *Let Σ be a first-order signature. Consider the following rewriting systems, terms, and sub-signatures of Σ:*

$$\begin{array}{rcl} \mathcal{R}_1 & = & \{\, \mathsf{dec}(\mathsf{enc}(x,y),y) \to x, \quad \mathsf{enc}(\mathsf{dec}(x,y),y) \to x \,\} \\ T_1 & = & \mathsf{enc}(U_1,U_2) \\ \Sigma_1 & = & \{\mathsf{enc},\mathsf{dec}\} \\ \\ \mathcal{R}_2 & = & \{\, \mathsf{sdec}(\mathsf{senc}(x,y,z),y) \to x, \quad \mathsf{sdec\_success}(\mathsf{senc}(x,y,z),y) \to 1 \,\}, \\ T_2 & = & \mathsf{senc}(U_1,U_2,U_3) \\ \Sigma_2 & = & \{\mathsf{senc},\mathsf{sdec},\mathsf{sdec\_success}\} \\ \\ \mathcal{R}_3 & = & \{\, \mathsf{pdec}(\mathsf{penc}(x,\mathsf{pub}(y),z),y) \to x, \quad \mathsf{pdec\_success}(\mathsf{penc}(x,\mathsf{pub}(y),z),y) \to 1 \,\} \\ T_3 & = & \mathsf{penc}(U_1,U_2,U_3) \\ \Sigma_3 & = & \{\mathsf{penc},\mathsf{pdec},\mathsf{pdec\_success}\} \\ \\ \mathcal{R}_4 & = & \mathcal{R}_3 \\ T_4 & = & \mathsf{pub}(U_2) \\ \Sigma_4 & = & \Sigma_3 \end{array}$$

Notice that for all i, $\mathcal{R}_i$ is subterm and convergent.

Let $i \in \{1,2,3,4\}$. Let $\mathcal{R}_0$ be a rewriting system over $\Sigma_0 = \Sigma - \Sigma_i$ such that

(iii) rules in $\mathcal{R}_0$ are made of symbols in Σ_0 only;

(iv) $\mathcal{R}_i \cup \mathcal{R}_0$ is confluent;

(v) for every rule $l \to r$ in $\mathcal{R}_0$, $\mathrm{var}(r) \subseteq \mathrm{var}(l)$.

Let $\mathcal{R} = \mathcal{R}_i \cup \mathcal{R}_0$ and E be the equational theory associated to $\mathcal{R}$.

We write $\to_{\mathcal{R}_i/\mathcal{R}_0}$ for the relation $\to^*_{\mathcal{R}_0}\to_{\mathcal{R}_i}\to^*_{\mathcal{R}_0}$.

Let φ be a frame in $\mathcal{R}_i/\mathcal{R}_0$-normal form (that is, there exists no φ' such that $\varphi \to^*_{\mathcal{R}_0}\to_{\mathcal{R}_i}\to^*_{\mathcal{R}_0} \varphi'$). Assume that

- *if $i \in \{1, 2, 4\}$, $\varphi \not\vdash_E U_2$;*
- *if $i = 3$,*
  - *there exists $j \in \{1, 2, 3\}$ such that $\varphi \not\vdash_E U_j$, and*
  - *for all V such that $U_2 =_E \mathsf{pub}(V)$, $\varphi \not\vdash_E V$.*

Let a be a fresh name. Then it holds that

$$\varphi \approx_E \mathrm{cut}^E_{T_i,a}(\varphi)$$

Proof. Let $\varphi' = \mathrm{cut}^E_{T_i,a}(\varphi)$. We write $\mathrm{cut}_{T_i,a}$ for $\mathrm{cut}^E_{T_i,a}$.

If $M\varphi' =_E N\varphi'$ with $\mathrm{names}(M, N) \cap \mathrm{names}(\varphi) = \emptyset$, then by stability of E by substitution over names, $M\varphi =_E N\varphi$.

Assume that $M\varphi =_E N\varphi$ with $\mathrm{names}(M, N) \cap \mathrm{names}(\varphi) = \emptyset$. We prove successively,

1. $\mathrm{cut}_{T_i,a}(M\varphi) = M\,\mathrm{cut}_{T_i,a}(\varphi)$ and $\mathrm{cut}_{T_i,a}(N\varphi) = N\,\mathrm{cut}_{T_i,a}(\varphi)$,
2. condition (b) of Lemma 16 is fulfilled on $M\varphi$ and $N\varphi$.

Indeed, using Lemma 16, we then obtain $M\varphi' = \mathrm{cut}_{T_i,a}(M\varphi) = \mathrm{cut}_{T_i,a}(N\varphi) = N\varphi'$.

We consider only the case of M since the one of N is identical. Let $T_i = f_i(U_1, \ldots, U_{n_i})$.

1. If $\mathrm{cut}_{T_i,a}(M\varphi) \neq M\,\mathrm{cut}_{T_i,a}(\varphi)$, then there exists a position p in M such that $M|_p$ is of the form $M|_p = f_i(M_1, \ldots, M_{n_i})$ and $\forall j, U_j =_{E_0} M_j\varphi$. In particular, $\forall j, \varphi \vdash_E U_j$; we obtain a contradiction.
2. Assume that $S = (M\varphi)|_p = g(S_1 \ldots S_{n_i})$ is a subterm of $M\varphi$ at position p and there exists a rule $l = g(l_1 \ldots l_{n_i}) \to r \in \mathcal{R}_i$ and a substitution θ such that $\forall j, S_j \to^*_{\mathcal{R}} l_j\theta$.

   Since φ is in $\mathcal{R}_i/\mathcal{R}_0$-normal form and $S \to^*_{\mathcal{R}} l\theta \to_{\mathcal{R}_i} r\theta$, p is a non-variable position of M. Thus, there exists a subterm $g(M_1 \ldots M_{n_i})$ of M such that $\forall j, M_j\varphi = S_j \to^*_{\mathcal{R}} l_j\theta$.

   We distinguish several cases depending on the values of i and the actual rule $l \to r \in \mathcal{R}_i$.

   (a) If $i = 1$,
   - If $l = \mathsf{dec}(\mathsf{enc}(x, y), y)$, the only subterm to be considered is $t = \mathsf{enc}(x, y)$. $t_2 = y = l_2$ is such that $U_2 \neq_E t_2\theta$. Indeed, otherwise, we deduce $U_2 =_E l_2\theta \leftarrow^*_{\mathcal{R}} M_2\varphi$ which contradicts $M_2 \not\vdash_E U_2$.
   - If $l = \mathsf{enc}(\mathsf{dec}(x, y), y)$, the only subterm to be considered is $t = l$. Similarly as above, $t_2 = y = l_2$ is such that $U_2 \neq_E t_2\theta$.

   (b) If $i = 2$,
   - If $l = \mathsf{sdec}(\mathsf{senc}(x, y, z), y)$, the only subterm to be considered is $t = \mathsf{senc}(x, y, z)$. Again, $t_2 = y = l_2$ is such that $U_2 \neq_E t_2\theta$.
   - The case $l = \mathsf{sdec\_success}(\mathsf{senc}(x, y, z), y)$ is similar.

   (c) If $i = 3$,

- If $l = \mathsf{pdec}(\mathsf{penc}(x, \mathsf{pub}(y), z), y)$, the only subterm to be considered is $t = \mathsf{penc}(x, \mathsf{pub}(y), z)$. $t_2 = \mathsf{pub}(y) = \mathsf{pub}(l_2)$ is such that $U_2 \neq_E t_2\theta$. Otherwise, we deduce $U_2 =_E \mathsf{pub}(l_2\theta)$ and $l_2\theta \leftarrow^*_{\mathcal{R}} M_2\varphi$, which contradicts the assumption $\forall V, U_2 =_E \mathsf{pub}(V) \Rightarrow \varphi \not\vdash_E V$.
- the case $l = \mathsf{pdec\_success}(\mathsf{penc}(x, \mathsf{pub}(y), z), y)$ is similar.

(d) If $i = 4$,

- If $l = \mathsf{pdec}(\mathsf{penc}(x, \mathsf{pub}(y), z), y)$, the only subterm to be considered is $t = \mathsf{pub}(y)$. Similarly as before, $t_1 = y = l_2$ is such that $U_2 \neq_E t_2\theta$.
- The case $l = \mathsf{pdec\_success}(\mathsf{penc}(x, \mathsf{pub}(y), z), y)$ is similar.

□

Example 7. Consider an equational theory E made of the rewriting rules $\mathcal{R}_1$ above and any set of rules $\mathcal{R}_0$ such that the conditions (iii)–(v) are fulfilled. (For instance, the symbols in $\mathcal{R}_0$ and $\mathcal{R}_1$ are disjoint, and $\mathcal{R}_0 \uplus \mathcal{R}_1$ is convergent.)

Using Proposition 17, for every frame

$$\varphi = \varphi_0 \uplus \{x = \mathsf{enc}(U_1, U_2)\}$$

if $\varphi \not\vdash_E U_2$ and $\mathsf{enc}(U_1, U_2)$ is not E-equivalent to a subterm of φ_0, we obtain that

$$\varphi \approx_E \varphi_0 \uplus \{x = n\}$$

for any fresh name n of the appropriate sort.