Sound and Complete Computational Interpretation of Symbolic Hashes in the Standard Model

Flavio D. Garcia and Peter van Rossum

Institute for Computing and Information Sciences, Radboud University Nijmegen, The Netherlands.

Abstract

This paper provides one more step towards bridging the gap between the formal and computational approaches to the verification of cryptographic protocols. We extend the well-known Abadi-Rogaway logic with probabilistic hashes and give a precise semantic interpretation to it using Canetti's oracle hashes. These are probabilistic polynomial-time hashes that hide all partial information. We show that, under appropriate conditions on the encryption scheme, this interpretation is computationally sound and complete. This can be used to port security results from the formal world to the computational world when considering passive adversaries. We also give an explicit example showing that oracle hashing is not strong enough to obtain such a result for active adversaries.

1 Introduction

The analysis of security protocols is being carried out mainly by means of two different techniques. On the one hand, there is the logic approach, which sees messages as algebraic objects defined using some formal language. In this view, cryptographic operations are symbolic operations which are unbreakable. Attackers are typically modeled as so-called Dolev-Yao attackers [DY83], having total control over the network, having no computational limitations, and being only (but absolutely) incapable of breaking cryptographic operations. This view is appealing, because it is relatively easy to use and captures most mistakes commonly made in security protocols.

On the other hand, there is the complexity-based approach. Here messages are bitstrings and cryptographic operations are functions on bit strings

Email addresses: `flaviog@cs.ru.nl` (Flavio D. Garcia), `petervr@cs.ru.nl` (Peter van Rossum).

satisfying certain security properties [Yao82,GM84,Gol01]. Common security notions like secrecy, authenticity, and integrity are formulated in terms of the probability that someone can mount a successful attack. An attacker here is a resource bounded probabilistic algorithm, limited by running time and/or memory. The complexity based methods are more general and more realistic, but also more complex to use.

In the last few years much research has been done to relate these two perspectives [AR02,AJ01,MW04a,MW04b,Her05,GGvR08]. Such a relation takes the form of a function mapping symbolic messages m to (distributions over) bitstrings $[\![m]\!]$. This map then should relate messages that are observationally equivalent in the symbolic world (meaning that a Dolev-Yao attacker can see no difference between them) to indistinguishable distributions over bitstrings (meaning that a computationally bounded adversary can only with negligible probability distinguish the distributions). Such a map allows one to use formal methods, possibly even automated, to reason about security properties of protocols and have those reasonings be valid also in the computational world.

The work carried out in the literature on relating these two perspectives mainly deals with symmetric encryption [AR02,MW04a] and public key encryption [MW04b,Her05]. Several extensions have been proposed dealing, for instance, with key cycles [ABHS05]; partial information leakage [ABS05]; active instead of passive adversaries [MW04b] and more realistic security notions [AW05]. Other extensions add new primitives to the logic such as bilinear pairings [Maz07], modular exponentiation [BLMW07], and there are also frameworks dealing with generic equational theories [BCK05,ABW06,KM07].

Micciancio and Warinschi [MW04a] briefly but explicitly question if this logical approach can be extended to, among other things, collision resistant hashes. Backes, Pfitzmann, and Waidner [BPW06] show that in their universal composability framework [PW00], which deals with active adversaries, a sound interpretation of hashes cannot exist, but that it is possible to give a sound interpretation of formal hashes in the universal composability framework using random oracles. Similar results, also in the random oracle model, have been recently shown by Janvier et al. [JLM07] for passive adversaries and by Cortier et al. [CKKW06] for active adversaries. Random oracles are often used in the literature to model hash functions, although they are also often criticized for being unsound in the standard model; there exist examples of protocols that are secure in the random oracle model but provably insecure for every possible instantiation with a real function [CGH04].

The problem with hashes is that in the symbolic world $h(m)$ and $h(m')$ are indistinguishable for a Dolev-Yao attacker if the attacker does not know m or m'. In the computational world, however, the normal security definition — it must be computationally infeasible to compute any pre-image of a hash value or a hash collision [RS04] — does not guarantee that the hash function hides all partial information about the message; hence there is no guarantee that their interpretation as bitstrings $[\![h(m)]\!]$ and $[\![h(m')]\!]$ are computationally indistinguishable.

A possible solution to this can be found in the work of Canetti and others [Can97a,CMR98] on perfectly one-way functions (a.k.a. oracle hashing). These are computable probabilistic hash functions that hide all partial information of their input (see Section 3.3 for the definition and an example).

Our contribution. We propose an extension to the commonly used Abadi-Rogaway logic of symbolic messages introducing a *probabilistic hash operator* $\mathrm{h}^r(m)$ in the logic, next to the probabilistic symmetric encryption operator $\{\!|m|\!\}_k^r$. Just as the original logic introduces a $\square$-operator to put in place of undecryptable ciphertext (for us $\square^r$, since we also deal with repetitions of ciphertexts), we introduce a $\boxtimes^r$-operator to put in place of the hash of an unknown message. In the computational world, we interpret h as a perfectly one-way function $\mathcal{H}$. We prove that if the encryption algorithm $\mathcal{E}$ is type-0 secure, the resulting interpretation is sound, extending the result from [AR02]. The same result holds if $\mathcal{E}$ is IND-CPA. This result previously appeared, in abbreviated form, as [GvR06]. Furthermore, assuming that the encryption scheme is confusion-free, we prove that the interpretation is complete as well, extending the result from [MW04a].

For these results, the adversaries under consideration are passive: they do not try to actively modify messages or insert messages into the network. We show that although an oracle hash scheme allows us to port security results from the Dolev-Yao setting to the computational setting with respect to passive adversaries, it does not do so with respect to active adversaries. The reason is the same as for type-0 or IND-CPA encryption schemes: in such an encryption scheme it might be possible for an attacker to modify an encryption $\mathcal{E}(\kappa, \mu)$ of μ to the encryption $\mathcal{E}(\kappa, f(\mu))$ of another message that has some relation to μ; similarly, in an oracle hash scheme it might be possible for an attacker to modify the hash of an unknown message μ to the hash of another unknown message that has some relation to μ. Concretely, we construct an explicit oracle hash scheme in which an adversary can create the hash value $\mathcal{H}(\mu'\mu)$ given only $\mathcal{H}(\mu)$ and μ'. We stress that passive adversaries are the best that one can hope for as the standard security definitions for hash functions or even hash schemes are not resilient against active adversaries.

Overview. Section 2 introduces the message algebra, including the probabilistic encryption and probabilistic hash operators. It also defines the observational equivalence relation on messages. Section 3 then introduces the computational world, giving the security definitions for encryption and hashes. In Section 4 the semantic interpretation $[\![-]\!]$ is defined and Section 5 proves the soundness of this interpretation and Section 6 completeness. Section 7 discusses the limitation to passive adversaries. Finally, Section 8 discusses further research directions.

2 The Symbolic Setting

This section describes the message space and the observational equivalence extending the well-known Abadi-Rogaway logic [AR02] of symbolic messages with hashes. These messages are used to describe cryptographic protocols and the observational equivalence tells whether or not two protocol runs are indistinguishable for a global eavesdropper. Here a protocol run is simply the concatenation of all the messages exchanged in the run.

Definition 2.1 **Key** is an infinite set of *key symbols*, **Nonce** an infinite set of *nonce symbols*, **Const** a finite set of *constant symbols* including 0 and 1, and **Random** an infinite set of *randomness labels*. Keys are denoted by $k, k', \ldots$, nonces by $n, n', \ldots$, constants by $c, c', \ldots$, and randomness labels by $r, r', \ldots$. There is one special key called $k_\Box$ and for every randomness label r there is a special nonce called $n^r_\boxtimes$. Using these building blocks, *messages* are constructed using symbolic encryption, hashing, and pairing operations:

$$\mathbf{Msg} \ni m := c \mid k \mid n \mid \{\!|m|\!\}^r_k \mid \mathrm{h}^r(m) \mid \langle m, m\rangle \mid \Box^r \mid \boxtimes^r .$$

Here k and n do not range over all keys/nonces, but only over the non-special ones. Special symbols ($\Box^r$ and $\boxtimes^r$) are used to indicate undecryptable ciphertexts or hash values of unknown messages. When interpreting messages as (ensembles of distributions over) bitstrings, we will treat $\Box^r$ as if it were $\{\!|0|\!\}^r_{k_\Box}$ and $\boxtimes^r$ as if it were $\mathrm{h}^r(n^r_\boxtimes)$.

A message of the form $\{\!|m|\!\}^r_k$ is called an *encryption* and the set of all such messages is denoted by **Enc**. Similarly, messages of the form $\mathrm{h}^r(m)$ are called *hash values* and the set of all these messages is denoted by **Hash**. Finally **Box** denotes the set of all messages of the form $\Box^r$ or $\boxtimes^r$. The set of all messages that involve a "random choice" at their "top level", i.e., **Key**$\cup$**Nonce**$\cup$**Enc**$\cup$**Hash**$\cup$**Box**, is denoted by **RndMsg**. The randomness labels model the fact that our computational encryption scheme is probabilistic: $\langle\{\!|m|\!\}^r_k, \{\!|m|\!\}^r_k\rangle$ models a repetition (forwarding) of the same ciphertext, whereas $\langle\{\!|m|\!\}^r_k, \{\!|m|\!\}^{r'}_k\rangle$ models a re-encryption with the same key. Note that our computational hash scheme, oracle hashing, is also probabilistic; hence, also the symbolic hash function is equipped with randomness labels.

The *closure* of a set U of messages is the set of all messages that can be constructed from U using tupling, detupling, and decryption. It represents the useful information an adversary could deduce knowing U. Note that, due to one-wayness of a hash function, knowing $h^r(m)$ does not provide an adversary with any information about m. Similarly, and modeling the randomization of the encryption and hash schemes, building new encryptions or hash values does not give any useful information to the adversary. We also remark that pairs are useful to the adversary for checking if the argument m of a certain hash value $\mathrm{h}^r(m)$ can be constructed from other messages he already knows.

In contrast, freshly generated keys, nonces, (randomized) encryptions or (randomized) hash values are of no use to the adversary as they will not appear in m.

Definition 2.2 (Closure) Let U be a set of messages. The *closure* of U, denoted by $\overline{U}$, is the smallest set of messages satisfying:
(1) $\mathbf{Const} \subseteq \overline{U}$;
(2) $U \subseteq \overline{U}$;
(3) $m, m' \in \overline{U} \implies \langle m, m' \rangle \in \overline{U}$;
(4) $\{\!|m|\!\}_k^r, k \in \overline{U} \implies m \in \overline{U}$;
(5) $\langle m, m' \rangle \in \overline{U} \implies m, m' \in \overline{U}$.
For the singleton set $\{m\}$, we write $\overline{m}$ instead of $\overline{\{m\}}$.

The function *pattern*: $\mathbf{Msg} \to \mathbf{Msg}$ is a straightforward extension of the same function in Abadi-Rogaway [AR02] which takes a message m and reduces it to a pattern. Intuitively, $pattern(m)$ is the pattern that an attacker sees in a message m and messages with the same pattern (up to renaming) look the same to her. Encryptions with keys the attacker cannot learn (i.e., not in $\overline{m}$) are replaced by $\square$ and hash values of unknown messages by $\boxtimes$. Since we allow repetition of ciphertexts and hash values, we have to distinguish between unknown, but equal, ciphertexts $\langle \square^r, \square^r \rangle$ and unknown, but distinct, ciphertexts $\langle \square^r, \square^{r'} \rangle$ and likewise for hashes.

Definition 2.3 The function *pattern*: $\mathbf{Msg} \to \mathbf{Msg}$ is defined by

$$pattern(m) = pattern(m, \overline{m})$$

where we overload *pattern* by defining *pattern*: $\mathbf{Msg} \times \mathcal{P}(\mathbf{Msg}) \to \mathbf{Msg}$ as

$$\begin{aligned}
pattern(\langle m_1, m_2 \rangle, U) &= \langle pattern(m_1, U), pattern(m_2, U) \rangle \\
pattern(\{\!|m|\!\}_k^r, U) &= \begin{cases} \{\!|pattern(m, U)|\!\}_k^r, & \text{if } k \in U; \\ \square^{\mathcal{R}(\{\!|m|\!\}_k^r)}, & \text{otherwise.} \end{cases} \\
pattern(\mathrm{h}^r(m), U) &= \begin{cases} \mathrm{h}^r(pattern(m, U)), & \text{if } m \in U; \\ \boxtimes^{\mathcal{R}(\mathrm{h}^r(m))}, & \text{otherwise.} \end{cases} \\
pattern(m, U) &= m \quad \text{in any other case.}
\end{aligned}$$

Here $\mathcal{R}$: $\mathbf{Enc} \cup \mathbf{Hash} \hookrightarrow \mathbf{Random}$ is an injective function that takes an encryption or a hash value and outputs a tag that identifies its randomness.

The tagging function $\mathcal{R}$ is needed to make sure that the function *pattern* is injective: distinct undecryptable messages should be replaced by distinct boxes and similarly for hashes. Note that instead of using $\mathcal{R}$ one could also tacitly assume that randomness labels r used in distinct contexts, such as $\{\!|m|\!\}_k^r$ and $\{\!|m'|\!\}_k^{r'}$ with $m \neq m'$, are always distinct.

Example 2.4 Consider the message

$$m = \langle \{\!\!|\, \{\!\!|1|\!\!\}^{r'}_{k'}, \mathrm{h}^{\tilde{r}}(n) \,|\!\!\}^{r}_{k}, \mathrm{h}^{\hat{r}}(k), k \rangle .$$

Then $pattern(m) = \langle \{\!\!|\ \Box^s\ ,\ \boxtimes^t\ |\!\!\}^{r}_{k}, \mathrm{h}^{\hat{r}}(k), k \rangle$, because k', n are not in $\overline{m}$,

where $t = \mathcal{R}(\mathrm{h}^{\tilde{r}}(n))$ and $s = \mathcal{R}(\{\!\!|1|\!\!\}^{r'}_{k'})$.

Definition 2.5 (Renaming) Two messages m and m' are said to be *equivalent up to renaming*, notation $m \approx m'$, if there is a type preserving permutation σ of $\mathbf{Key} \cup \mathbf{Nonce} \cup \mathbf{Box} \cup \mathbf{Random}$ such that $m = m'\sigma$. Here $m'\sigma$ denotes simultaneous substitution of x by $\sigma(x)$ in m', for all $x \in \mathbf{Key} \cup \mathbf{Nonce} \cup \mathbf{Box} \cup \mathbf{Random}$.

Definition 2.6 (Observational equivalence) Two messages m and m' are said to be *observationally equivalent*, denoted by $m \cong m'$, if $pattern(m) \approx pattern(m')$.

In the computational world, security definitions for encryption schemes do not guarantee security when encryption cycles occur. In the symbolic world, however, this poses no problem. Therefore, as in the original setting in [AR02], for the mapping from the symbolic world to the computational world to be sound, we have to forbid encryption cycles in the symbolic world.

Definition 2.7 (Sub-message) Define the relation sub-message, notation $\preceq$, as the smallest reflexive and transitive relation on **Msg** satisfying
(1) $m_1 \preceq \langle m_1, m_2 \rangle$
(2) $m_2 \preceq \langle m_1, m_2 \rangle$
(3) $m \preceq \{\!\!|m|\!\!\}^{r}_{k}$
(4) $m \preceq \mathrm{h}^{r}(m)$.

Definition 2.8 (Acyclicity) Let m be a message and k, k' two keys. The key k is said to *encrypt k' in m* if m has a sub-message of the form $\{\!\!|m'|\!\!\}^{r}_{k}$ with k' being a sub-message of m'. A message is said to be *acyclic* if there is no sequence $k_1, k_2, \ldots, k_n, k_{n+1} = k_1$ of keys such that k_i encrypts k_{i+1} in m for all $i \in \{1, \ldots, n\}$.

3 The Computational Setting

This section gives a brief overview of some of the concepts used in the complexity theoretic approach to security protocols. Much of this is standard; the reader is referred to [GB01,BDJR97] for a thorough treatment of the basic concepts, to [AR02] for the notion of *type-0 security* for cryptographic schemes (see Section 3.2 below), and to [Can97a] for the notion of *oracle hashing* (see Section 3.3 below).

In the computational world, messages are elements of $\mathbf{Str} := \{0,1\}^*$. Cryptographic algorithms and adversaries are probabilistic polynomial-time algorithms. When analyzing cryptographic primitives, it is customary to consider probabilistic algorithms that take an element in $\mathbf{Param} := \{1\}^*$ as input, whose length scales with the security parameter. By making the security parameter large enough, the system should become arbitrarily hard to break.

This idea is formalized in the security notions of the cryptographic operations. The basic one, which is what is used to define the notion of semantically equivalent messages, is that of *computational indistinguishability* of probability ensembles over $\mathbf{Str}$. Here a *probability ensemble over* $\mathbf{Str}$ is a sequence $\{A_\eta\}_{\eta\in\mathbb{N}}$ of probability distributions over $\mathbf{Str}$ indexed by the security parameter.

Definition 3.1 (Computational indistinguishability) Two probability ensembles $\{X_\eta\}_{\eta\in\mathbb{N}}$ and $\{Y_\eta\}_{\eta\in\mathbb{N}}$ are said to be *computationally indistinguishable*, notation $\{X_\eta\}_\eta \equiv \{Y_\eta\}_\eta$, if for every probabilistic polynomial-time algorithm D,

$$\mathbb{P}[x \leftarrow X_\eta; D(1^\eta, x) = 1] - \mathbb{P}[x \leftarrow Y_\eta; D(1^\eta, x) = 1]$$

is a negligible function of η.

Recall that a function $f\colon \mathbb{N} \to \mathbb{N}$ is called *negligible* if for all positive polynomials p, $f(\eta) \leq 1/p(\eta)$ for large enough η (i.e., if for all positive polynomials p, there exists an $\eta_0 \in \mathbb{N}$ such that for all $\eta \in \mathbb{N}$ with $\eta > \eta_0$, $f(\eta) \leq 1/p(\eta)$).

After a brief interlude on probabilistic polynomial-time algorithms in Section 3.1, we give the formal definition of an encryption scheme and its security notion in Section 3.2 and of oracle hashing in Section 3.3.

3.1 Probabilistic Algorithms

In Definition 3.1, the notion of probabilistic polynomial-time algorithm was already used. Because we explicitly use two different views of these algorithms and in order to fix notation, we give a more precise definition.

Definition 3.2 Coins is the set $\{0,1\}^\omega$, the set of all infinite sequences of 0's and 1's. We equip Coins with the probability distribution obtained by flipping a fair coin for each element in the sequence. To be precise, $\text{Coins} = \langle\{0,1\}^\omega, \mathcal{F}, \mathbb{P}\rangle$. Here $\mathcal{F}$ is the σ-field on $\{0,1\}^\omega$ generated by the *cones* $\{\alpha\beta \mid \beta \in \{0,1\}^\omega\}$, $\alpha \in \{0,1\}^*$ and $\mathbb{P}\colon \mathcal{F} \to [0,1]$ is the unique measure given on the cones by $\mathbb{P}(\{\alpha\beta \mid \beta \in \{0,1\}^\omega\}) = 2^{-|\alpha|}$.

Definition 3.3 The result of running a probabilistic algorithm A on an input $x \in \mathbf{Str}$ is a probability distribution $A(x)$ over $\mathbf{Str}$. When we need to explicitly write the randomness used while running A, we write $A(x, \rho)$ with $\rho \in \text{Coins}$. Using this notation, $\mathbb{P}[A(x) = y] = \mathbb{P}[\rho \leftarrow \text{Coins}; A(x,\rho) = y]$. When confusion is unlikely, we will also denote the support of this probability distribution, $\{y \in \mathbf{Str} | \mathbb{P}[\rho \leftarrow \text{Coins}; A(x,\rho) = y)] > 0\}$, by $A(x)$.

Now suppose that A runs in polynomial time p. Then running A on x cannot use more than $p(|x|)$ coin flips. Letting $\mathrm{Coins}_{p(|x|)}$ denote the uniform probability distribution on $\{0,1\}^{p(|x|)}$, we can also write $\mathbb{P}[A(x) = y] = \mathbb{P}[\rho \leftarrow \mathrm{Coins}_{p(|x|)}; A(x,\rho) = y]$.

3.2 Encryption Scheme

For each security parameter $\eta \in \mathbb{N}$ we let $\mathbf{Plaintext}_\eta \subseteq \mathbf{Str}$ be a non-empty set of *plaintexts*, satisfying that for each $\eta \in \mathbb{N}$: $\mathbf{Plaintext}_\eta \subseteq \mathbf{Plaintext}_{\eta+1}$ as in Goldwasser and Bellare [GB01]. Let us define $\mathbf{Plaintext} = \bigcup_\eta \mathbf{Plaintext}_\eta$. There is a set $\mathbf{Keys} \subseteq \mathbf{Str}$ of *keys* and also a set $\mathbf{Ciphertext} \subseteq \mathbf{Str}$ of *ciphertexts*. Furthermore, there is a special bitstring $\perp$ not appearing in $\mathbf{Plaintext}$ or $\mathbf{Ciphertext}$. An *encryption scheme* Π consists of three algorithms:

(1) a (probabilistic) key generation algorithm $\mathcal{K}\colon \mathbf{Param} \to \mathbf{Keys}$ that outputs, given a unary sequence of length η, a randomly chosen element of $\mathbf{Keys}$;
(2) a (probabilistic) encryption algorithm $\mathcal{E}\colon \mathbf{Keys} \times \mathbf{Str} \to \mathbf{Ciphertext} \cup \{\perp\}$ that returns, given a key and a bitstring, an element from $\mathbf{Ciphertext}$ or $\perp$;
(3) a (deterministic) decryption algorithm $\mathcal{D}\colon \mathbf{Keys} \times \mathbf{Str} \to \mathbf{Plaintext} \cup \{\perp\}$ that returns, given a key and a ciphertext, an element from $\mathbf{Plaintext}$ or $\perp$.

These algorithms must satisfy that the decryption (with the correct key) of a ciphertext returns the original plaintext. The element $\perp$ is used to indicate failure of en- or decryption, although, before hand, there is no requirement that decrypting with the wrong keys yields $\perp$. Now we define type-0 security of an encryption scheme as in [AR02], which is a variant of the standard semantic security definition, enhanced with some extra properties. In particular a type-0 secure encryption scheme is which-key concealing, repetition concealing and length hiding. We refer to the original paper for motivation and explanations on how to achieve such an encryption scheme. The notion of type-0 security makes slightly unrealistic assumptions on the encryption scheme. However our result on hashes does not significantly depend on the specific security notion for the encryption scheme. As in [MP05,Her05,Ban04], it is possible to replace type-0 security by the standard notion of IND-CPA or IND-CCA by adapting the definition of *pattern*. For simplicity of the exposition, throughout this paper we adopt the former security notion.

Definition 3.4 An *adversary (for type-0 security)* is a probabilistic polynomial-time algorithm $A^{\mathcal{F}(-),\mathcal{G}(-)}\colon \mathbf{Param} \to \{0,1\}$ having access to two probabilistic oracles $\mathcal{F},\mathcal{G}\colon \mathbf{Str} \to \mathbf{Str}$. The *advantage* of such an adversary is the function $\mathrm{Adv}_A \colon \mathbb{N} \to \mathbb{R}$ defined by

$$\mathrm{Adv}_A(\eta) = \mathbb{P}[\kappa, \kappa' \leftarrow \mathcal{K}(1^\eta); A^{\mathcal{E}(\kappa,-),\mathcal{E}(\kappa',-)}(1^\eta) = 1] - \mathbb{P}[\kappa \leftarrow \mathcal{K}(1^\eta); A^{\mathcal{E}(\kappa,0),\mathcal{E}(\kappa,0)}(1^\eta) = 1].$$

Here the probabilities are taken over the choice of κ and κ' by the key generation algorithm, over the internal coins used by the oracles, and over the internal choices of A. An encryption scheme $\langle \mathcal{K}, \mathcal{E}, \mathcal{D} \rangle$ is called *type-0 secure* if for all polynomial-time adversaries A as above, the advantage Adv_A is a negligible function of η. enough η, $\mathrm{Adv}_A(\eta) \leq \frac{1}{p(\eta)}$.

In the sequel we need an extra assumption on the encryption scheme, namely that the ciphertexts are well-spread as a function of the coins tosses of $\mathcal{E}$. It means that for *all* plaintexts μ and *all* keys κ, no ciphertext is exceptionally likely to occur as the encryption of μ under κ. Note that this does not follow from, nor implies type-0 security. Also note that every encryption scheme running in cipher block chaining mode automatically has this property: the initial vector provides the required randomness.

Definition 3.5 (Well-spread) An encryption scheme $\langle \mathcal{K}, \mathcal{E}, \mathcal{D} \rangle$ is said to be *well-spread* if

$$\sup_{x \in \mathbf{Ciphertext}, \kappa \in \mathcal{K}(1^\eta), \mu \in \mathbf{Plaintext}_\eta} \mathbb{P}[\mathcal{E}(\kappa, \mu) = x]$$

is a negligible function of η.

For completeness it is needed that the decryption algorithm returns *reject* whenever it is called with a key that was not used to encrypt the message in the first place. The special bitstring $\perp$ is used to indicate failure of decryption. This property is called *confusion freeness*. See [MW04a], where the completeness for the original Abadi-Rogaway logic is proven.

Definition 3.6 (Confusion freeness) Let $\Pi = \langle \mathcal{K}, \mathcal{E}, \mathcal{D} \rangle$ be an encryption scheme indexed by the security parameter η. Π is said to be *confusion free* if for all bitstrings μ the probability

$$\mathbb{P}[\kappa_1, \kappa_2 \leftarrow \mathcal{K}(\eta) : \mathcal{D}_{\kappa_1}(\mathcal{E}_{\kappa_2}(\mu)) \neq \perp]$$

is a negligible function of η.

3.3 Oracle Hashing

The underlying secrecy assumptions behind formal or Dolev-Yao hashes [DY83] are very strong. It is assumed that given a hash value $f(x)$, it is not possible for an adversary to learn any information about the pre-image x. In the literature this idealization is often modeled with the random oracle [BR93]. Such a primitive is not computable and therefore it is also an idealization. Practical hash functions like SHA or MD5 are very useful cryptographic primitives but have no proven security guaranties. Moreover, under the traditional security notions (one-wayness), a function that reveals half of its input might still be secure. In addition, any deterministic hash function f leaks partial information about x, namely $f(x)$ itself. Throughout this paper we consider a new primitive introduced by Canetti [Can97a] called *oracle hashing*, that mimics what semantic security is for encryption schemes. This hash function is probabilistic and therefore it needs a verification function, just as in a signature scheme. A *hash scheme* consists of two algorithms $\mathcal{H}$ and $\mathcal{V}$. The probabilistic algorithm $\mathcal{H}\colon \mathbf{Param} \times \mathbf{Str} \to \mathbf{Str}$ takes a unary sequence and a message and outputs a hash value; the verification algorithm $\mathcal{V}\colon \mathbf{Str} \times \mathbf{Str} \to \{0,1\}$ that given two messages x and c correctly decides whether c is a hash of x or not. As an example we reproduce here a hash scheme proposed in the original paper. Let p be a large (i.e., scaling with η) safe prime (i.e., a prime of the form $2q+1$ where q is also prime). Take $\mathcal{H}(x) = \langle r^2 \mod p, r^{2\cdot f(x)} \mod p\rangle$, where r is a randomly chosen element in $\mathbb{Z}_p^*$ and f is any collision resistant hash function. The verification algorithm $\mathcal{V}(x, \langle a, b\rangle)$ just checks whether $b = a^{f(x)} \mod p$.

We consider two security notions for such a hash scheme. The first one, proposed by Canetti [Can97a] and later revisited in [CMR98], *oracle indistinguishability*, guarantees that an adversary can gain no information at all about a bitstring, given its hash value (or rather, with sufficiently small probability). The second one, more standard, is an appropriate form of collision resistance. It guarantees that an adversary cannot (or rather, again, with sufficiently small probability) compute two distinct messages that successfully pass the verification test with the same hash value.

Definition 3.7 (Oracle indistinguishability) A hash scheme $\langle\mathcal{H},\mathcal{V}\rangle$ is said to be *oracle indistinguishable* if for every family of probabilistic polynomial-time predicates $\{D_\eta\colon \mathbf{Str} \to \{0,1\}\}_{\eta\in\mathbb{N}}$ and every positive polynomial p there is a polynomial size family $\{L_\eta\}_{\eta\in\mathbb{N}}$ of subsets of $\mathbf{Str}$ such that for all large enough η and all $x, y \in \mathbf{Str} \setminus L_\eta$:

$$\mathbb{P}[D_\eta(\mathcal{H}(1^\eta, x)) = 1] - \mathbb{P}[D_\eta(\mathcal{H}(1^\eta, y)) = 1] < \frac{1}{p(\eta)}.$$

Here the probabilities are taken over the choices made by $\mathcal{H}$ and the choices made by D_η. This definition is the non-uniform [Gol01] version of oracle indistinguishability proposed by Canetti [Can97a] as it is used in the proofs in Appendix B of the full version [Can97b].

Definition 3.8 (Collision resistance) A hash scheme $\langle\mathcal{H},\mathcal{V}\rangle$ is said to be *collision resistant* if for every probabilistic polynomial-time adversary A, the probability

$$\mathbb{P}[\langle c, x, y\rangle \leftarrow A(1^\eta); x \neq y \wedge \mathcal{V}(x, c) = \mathcal{V}(y, c) = 1]$$

is a negligible function of η.

4 Interpretation

Section 2 describes a setting where messages are symbolic terms generated by some grammar. In Section 3 messages are bitstrings and operations are given by probabilistic algorithms operating on bitstrings. This section shows how to map symbolic messages to (distributions over) bitstrings. This interpretation is very much standard. We refer to [AR02,AJ01,MW04a] for a thorough explanation. In particular this section introduces notation that allows us to assign, beforehand, some of the random coin flips used for the computation of the interpretation of a message. This notation becomes useful throughout the soundness proof.

Tagged representation. Throughout this paper we assume that it is always possible to recover the type information of a message from its bitstring representation. This can be easily achieved by adding the necessary type tags to the bitstring representation. We will abstract from this representation by overloading the notation. We use Greek letters for bitstrings and μ represents a bitstring of a generic type. We write $\mu_1\mu_2$ for a pair of bitstrings (in [AR02] this would be written as $\langle(\mu_1, \mu_2), \text{“pair”}\rangle$); ϵ for a ciphertext; κ for a key; ψ for a hash value; ν for a nonce and ς for a constant.

Definition 4.1 For every message m we define the set $\mathrm{R}(m) \subseteq \mathbf{Msg}$ of *random messages in m* as follows:

$$\begin{aligned} \mathrm{R}(c) &= \emptyset & \mathrm{R}(\{\!|m|\!\}_k^r) &= \mathrm{R}(m) \cup \{k, \{\!|m|\!\}_k^r\} \\ \mathrm{R}(n) &= \{n\} & \mathrm{R}(\mathrm{h}^r(m)) &= \mathrm{R}(m) \cup \{\mathrm{h}^r(m)\} \\ \mathrm{R}(k) &= \{k\} & \mathrm{R}(\langle m_1, m_2\rangle) &= \mathrm{R}(m_1) \cup \mathrm{R}(m_2) \\ \mathrm{R}(\Box^r) &= \{k_\Box, \Box^r\} & \mathrm{R}(\boxtimes^r) &= \{n_\boxtimes^r, \boxtimes^r\}. \end{aligned}$$

Note that $\mathrm{R}(m)$ is nearly equal to the set of all sub-messages of m that are in **RndMsg**; the only difference is that $\mathrm{R}(m)$ also may contain the special key $k_\Box$ or special nonces $n_\boxtimes^r$. When interpreting a message m as (ensembles of distributions over) bitstrings (Definition 4.3 below), we will first choose a sequence of coin flips for all elements of $\mathrm{R}(m)$ and use these sequences as source of randomness for the appropriate interpretation algorithms.

Definition 4.2 For every finite subset X of **RndMsg** we define $\mathrm{Coins}(X)$ as $\{\tau \mid \tau \colon X \to \mathrm{Coins}\}$. We equip $\mathrm{Coins}(X)$ with the induced product probability distribution. Furthermore, for every message m we write $\mathrm{Coins}(m)$ instead of $\mathrm{Coins}(\mathrm{R}(m))$.

An element of τ of $\mathrm{Coins}(m)$ gives, for every sub-message m' of m that requires random choices when interpreting this sub-message as a bitstring, an infinite sequence $\tau(m')$ of coin flips that will be used to resolve the randomness.

Now we are ready to give semantic to our message algebra. We use $\mathcal{E}$ to interpret encryptions, $\mathcal{K}$ to interpret key symbols, and $\mathcal{H}$ to interpret hashes. We let $\mathcal{C}\colon \mathbf{Const} \to \mathbf{Str}$ be a function that (deterministically) assigns a constant bit string to each constant identifier. We let $\mathcal{N}\colon \mathbf{Param} \to \mathbf{Str}$ be the nonce generation function that, given a unary sequence of length η, chooses uniformly and randomly a bitstring from $\{0,1\}^\eta$.

Definition 4.3 For a message m, a value of the security parameter $\eta \in \mathbb{N}$, a finite set U of messages containing $\mathrm{R}(m)$, and a $\tau \in \mathrm{Coins}(U)$, we can (deterministically) create a bitstring $\llbracket m \rrbracket^\tau_\eta \in \mathbf{Str}$ as follows:

$$\begin{aligned}
\llbracket c \rrbracket^\tau_\eta &= \mathcal{C}(c) & \llbracket \{\!| m |\!\}^r_k \rrbracket^\tau_\eta &= \mathcal{E}(\llbracket k \rrbracket^\tau_\eta, \llbracket m \rrbracket^\tau_\eta, \tau(\{\!| m |\!\}^r_k))\\
\llbracket k \rrbracket^\tau_\eta &= \mathcal{K}(1^\eta, \tau(k)) & \llbracket \mathrm{h}^r(m) \rrbracket^\tau_\eta &= \mathcal{H}(1^\eta, \llbracket m \rrbracket^\tau_\eta, \tau(\mathrm{h}^r(m)))\\
\llbracket n \rrbracket^\tau_\eta &= \mathcal{N}(1^\eta, \tau(n)) & \llbracket \Box^r \rrbracket^\tau_\eta &= \mathcal{E}(\llbracket k_\Box \rrbracket^\tau_\eta, \mathcal{C}(0), \tau(\Box^r))\\
\llbracket \langle m_1, m_2 \rangle \rrbracket^\tau_\eta &= \llbracket m_1 \rrbracket^\tau_\eta \llbracket m_2 \rrbracket^\tau_\eta & \llbracket \boxtimes^r \rrbracket^\tau_\eta &= \mathcal{H}(1^\eta, \llbracket n^r_\boxtimes \rrbracket^\tau_\eta, \tau(\boxtimes^r)).
\end{aligned}$$

Note that $\llbracket m \rrbracket^\tau_\eta = \llbracket m \rrbracket^{\tau|_{\mathrm{R}(m)}}_\eta$. For a fixed message m and $\eta \in \mathbb{N}$, choosing τ from the probability distribution $\mathrm{Coins}(\mathrm{R}(m))$ creates a probability distribution $\llbracket m \rrbracket_\eta$ over $\mathbf{Str}$:

$$\llbracket m \rrbracket_\eta := [\tau \leftarrow \mathrm{Coins}(m); \llbracket m \rrbracket^\tau_\eta].$$

Note that although the codomain of $\tau \in \mathrm{Coins}(m)$ is Coins, the set of *infinite* bitstrings, when interpreting a fixed message m at a fixed value of the security parameter η, only a predetermined *finite* initial segment of each sequence of coin flips will be used by $\mathcal{K}$, $\mathcal{N}$, $\mathcal{E}$, and $\mathcal{H}$ (cf. Definition 3.3). In fact, because these four algorithms run in polynomial time, for every message m there exists a polynomial p_m such that every call to one of these four algorithms uses at most the first $p_m(\eta)$ elements of τ. Now define $\mathrm{Coins}_\eta(m) = \{\tau \mid \tau\colon \mathrm{R}(m) \to \{0,1\}^{p_m(\eta)}\}$ and equip this with the uniform probability distribution. Then we can also write when

$$\llbracket m \rrbracket_\eta = [\tau \leftarrow \mathrm{Coins}_\eta(m); \llbracket m \rrbracket^\tau_\eta].$$

Furthermore, letting η range over $\mathbb{N}$ creates an ensemble of probability distributions $\llbracket m \rrbracket$ over $\mathbf{Str}$, namely $\llbracket m \rrbracket := \{\llbracket m \rrbracket_\eta\}_{\eta\in\mathbb{N}}$.

4.1 Partial interpretation

For technical reasons throughout the soundness proof, we need to compute the interpretation of a symbolic message while part of its randomness has already been chosen. This section introduces straightforward notation to do so.

Definition 4.4 For every message m and m' we define the set $\mathrm{R}(m, m') \subseteq$ **RndMsg** of *random messages in m relative to m′* as follows:
if $m = m'$, then $\mathrm{R}(m, m') = \emptyset$, otherwise

$$\begin{aligned} \mathrm{R}(c, m') &= \emptyset & \mathrm{R}(\{\!|m|\!\}_k^r, m') &= \mathrm{R}(m, m') \cup \{k, \{\!|m|\!\}_k^r\} \\ \mathrm{R}(n, m') &= \{n\} & \mathrm{R}(\mathrm{h}^r(m), m') &= \mathrm{R}(m, m') \cup \{\mathrm{h}^r(m)\} \\ \mathrm{R}(k, m') &= \{k\} & \mathrm{R}(\langle m_1, m_2\rangle, m') &= \mathrm{R}(m_1, m') \cup \mathrm{R}(m_2, m') \\ \mathrm{R}(\square^r, m') &= \{k_\square, \square^r\} & \mathrm{R}(\boxtimes^r, m') &= \{n_\boxtimes^r, \boxtimes^r\}. \end{aligned}$$

Note that $\mathrm{R}(m, m')$ is the set of all random messages in m except those that *only* occur as a sub-message of m'.

Example 4.5 Let m be the message $\langle k, \{\!|0|\!\}_k^r, \mathrm{h}^{r'}(\{\!|0|\!\}_k^r, n), n'\rangle$ and let $\tilde{m}$ be the message inside the hash: $\langle \{\!|0|\!\}_k^r, n\rangle$. Then the randomness in m is $\mathrm{R}(m) = \{k, \{\!|0|\!\}_k^r, \mathrm{h}^{r'}(\{\!|0|\!\}_k^r, n), n', n\}$, the randomness inside the hash is $\mathrm{R}(\tilde{m}) = \{\{\!|0|\!\}_k^r, k, n\}$, and the randomness that occurs only outside the hash is $\mathrm{R}(m, \mathrm{h}^{r'}(\tilde{m})) = \mathrm{R}(m) \setminus \{\mathrm{h}^{r'}(\tilde{m}), n\}$. The randomness that is shared between the inside of the hash and the outside of the hash is $\mathrm{R}(m, \mathrm{h}^{r'}(\tilde{m})) \cap \mathrm{R}(\tilde{m}) = \{k, \{\!|0|\!\}_k^r\}$.

We will need a way of interpreting a message as a bitstring when the interpretation of certain sub-messages has already been chosen in some other way.

Definition 4.6 Let $\eta \in \mathbb{N}$, let e be a function from $\mathrm{Dom}(e) \subseteq$ **Msg** to **Str** and let $\tau \in \mathrm{Coins}_\eta(U \setminus \mathrm{Dom}(e))$ with U a finite set of messages containing $\mathrm{R}(m)$. We interpret a message m using e whenever possible. Otherwise, we use the coin flips assigned by τ to generate an interpretation. If $m \in \mathrm{Dom}(e)$, then $[\![m]\!]_\eta^{e,\tau} = e(m)$, else

$$\begin{aligned} [\![c]\!]_\eta^{e,\tau} &= \mathcal{C}(c) & [\![\{\!|m|\!\}_k^r]\!]_\eta^{e,\tau} &= \mathcal{E}([\![k]\!]_\eta^\tau, [\![m]\!]_\eta^{e,\tau}, \tau(\{\!|m|\!\}_k^r)) \\ [\![k]\!]_\eta^{e,\tau} &= \mathcal{K}(1^\eta, \tau(k)) & [\![\mathrm{h}^r(m)]\!]_\eta^{e,\tau} &= \mathcal{H}(1^\eta, [\![m]\!]_\eta^{e,\tau}, \tau(\mathrm{h}^r(m))) \\ [\![n]\!]_\eta^{e,\tau} &= \mathcal{N}(1^\eta, \tau(n)) & [\![\square^r]\!]_\eta^{e,\tau} &= \mathcal{E}([\![k_\square]\!]_\eta^{e,\tau}, \mathcal{C}(0), \tau(\square^r)) \\ [\![\langle m_1, m_2\rangle]\!]_\eta^{e,\tau} &= [\![m_1]\!]_\eta^{e,\tau}[\![m_2]\!]_\eta^{e,\tau} & [\![\boxtimes^r]\!]_\eta^{e,\tau} &= \mathcal{H}(1^\eta, [\![n_\boxtimes^r]\!]_\eta^{e,\tau}, \tau(\boxtimes^r)). \end{aligned}$$

We also need a way of pre-specifying some of the random choices to be made when interpreting a message.

Definition 4.7 Let $\eta \in \mathbb{N}$ and let $\tau \in \mathrm{Coins}_\eta(U)$ for some finite set of messages U. Then for every message m, the distribution $[\![m]\!]_\eta^\tau$ is obtained by randomly choosing coins for the remaining randomness labels in m. Formally,

$$[\![m]\!]_\eta^\tau := [\tau' \leftarrow \mathrm{Coins}_\eta(\mathrm{R}(m) \setminus U); [\![m]\!]_\eta^{\tau \cup \tau'}],$$

where $\tau \cup \tau' \in \mathrm{Coins}_\eta(m)$ denotes the function which agrees with τ on $U \cap \mathrm{R}(m)$ and with τ' on $\mathrm{R}(m) \setminus U$.

This can also be combined with the previous way of preselecting a part of the interpretation. For a function e from a set $\mathrm{Dom}(e) \subseteq \mathbf{Msg}$ to $\mathbf{Str}$ and $\tau \in \mathrm{Coins}_\eta(U)$ as above, we define

$$[\![m]\!]^{e,\tau}_\eta := [\tau' \leftarrow \mathrm{Coins}_\eta(\mathrm{R}(m) \setminus U); [\![m]\!]^{e,\tau\cup\tau'}_\eta].$$

5 Soundness

This section shows that the interpretation proposed in the previous section is computationally sound. Throughout this section we assume that the encryption scheme $\langle \mathcal{K}, \mathcal{E}, \mathcal{D} \rangle$ is type-0 secure (or IND-CCA with *pattern* modified as in [Her05,MP05]) and well-spread, and that the probabilistic hash scheme $\langle \mathcal{H}, \mathcal{V} \rangle$ is oracle indistinguishable and collision resistant.

In order to isolate our contribution, we split the function *pattern* in two parts: one replacing the encryptions and one replacing the hashes. We define the function *encpat* as in Abadi-Rogaway [AR02] which takes a message m and reduces it to a pattern. This function does not replace hashes.

Definition 5.1 The function *encpat*: $\mathbf{Msg} \to \mathbf{Msg}$ is defined as follows

$$encpat(m) = encpat(m, \overline{m})$$

where *encpat*: $\mathbf{Msg} \times \mathcal{P}(\mathbf{Msg}) \to \mathbf{Msg}$ is defined by

$$\begin{aligned}
encpat(\langle m_1, m_2 \rangle, U) &= \langle encpat(m_1, U), encpat(m_2, U) \rangle \\
encpat(\{\![m]\!\}^r_k, U) &= \begin{cases} \{\![encpat(m, U)]\!\}^r_k, & \text{if } k \in U; \\ \square^{\mathcal{R}(\{\![m]\!\}^r_k)}, & \text{otherwise.} \end{cases} \\
encpat(\mathrm{h}^r(m), U) &= \mathrm{h}^r(encpat(m, U)) \\
encpat(m, U) &= m \quad \text{in any other case.}
\end{aligned}$$

Now we define the function *hashpat* which takes a message m and reduces all hashes of unknown (not in $\overline{m}$) sub-messages, to $\boxtimes$. This function does not replace encryptions.

Definition 5.2 Define the function *hashpat*: $\mathbf{Msg} \to \mathbf{Msg}$ as

$$hashpat(m) = hashpat(m, \overline{m})$$

where *hashpat*: $\mathbf{Msg} \times \mathcal{P}(\mathbf{Msg}) \to \mathbf{Msg}$ is defined by

$$\begin{aligned}
hashpat(\langle m_1, m_2 \rangle, U) &= \langle hashpat(m_1, U), hashpat(m_2, U) \rangle \\
hashpat(\{\![m]\!\}^r_k, U) &= \{\![hashpat(m, U)]\!\}^r_k \\
hashpat(\mathrm{h}^r(m), U) &= \begin{cases} \mathrm{h}^r(hashpat(m, U)), & \text{if } m \in U; \\ \boxtimes^{\mathcal{R}(\mathrm{h}^r(m))}, & \text{otherwise.} \end{cases} \\
hashpat(m, U) &= m \quad \text{in any other case.}
\end{aligned}$$

Lemma 5.3 *pattern* $\approx$ *encpat* $\circ$ *hashpat.*

Proof. A straightforward induction on m shows that $\overline{m} = \overline{\mathit{hashpat}(m)}$. Using this, another straightforward induction on m shows that $\mathit{pattern}(m) \approx \mathit{encpat}(\mathit{hashpat}(m))$. $\square$

Theorem 5.4 (Abadi-Rogaway) *Let m be an acyclic message. Suppose that for every sub-message $\mathrm{h}^r(\tilde{m})$ of m, $\tilde{m} \in \overline{m}$. Then $[\![m]\!] \equiv [\![\mathit{encpat}(m)]\!]$.*

Proof. The proof follows just like in Abadi-Rogaway [AR02]. Interpreting hashes here is straightforward because their argument is always known, by assumption. We refer the reader to the original paper for a full proof. $\square$

Before starting the proof of the soundness theorem, we give a sketch for an easy case and point out where the technical problems in the general case are. Consider two messages $h(n,0)n'$ and $h(n,1)n'$. These are observationally equivalent and we have to prove that $[\![h(n,0),n']\!]$ and $[\![h(n,1),n']\!]$ are computationally indistinguishable. The way to prove this is standard: we assume that there exists a probabilistic polynomial-time adversary A that can distinguish these two ensembles of probability distribution and use it to build an adversary D that break the oracle indistinguishability of the hash scheme $\langle \mathcal{H}, \mathcal{V} \rangle$. What D has to do is clear: it receives a hash value α which is either $\mathcal{H}(1^\eta, \nu 0)$ or $\mathcal{H}(1^\eta, \nu 1)$, and has to guess which one it is. It generates a nonce ν' and calls A on $\alpha\nu'$. Since A can successfully distinguish $\mathcal{H}(1^\eta, \nu 0)\nu'$ from $\mathcal{H}(1^\eta, \nu 1)\nu'$, this enables D to break oracle indistinguishability.

A problem occurs in the case of the messages $h(n,n')n'$ and $h(n,1)n'$. Receiving a hash value α which is either $\mathcal{H}(1^\eta, \nu\nu')$ or $\mathcal{H}(1^\eta, \nu 1)$, D cannot just generate a nonce ν'' and call A on $\alpha\nu''$: the distribution of $\mathcal{H}(1^\eta, \nu\nu')\nu'$ is not equal to that of $\mathcal{H}(1^\eta, \nu\nu')\nu''$. The technical solution is to provide the adversary D with access to ν'; this is enough to prove that oracle indistinguishability can then be broken. What is still needed is that the inside of the hash is still "random enough" even if part of it is revealed; in this particular case this means that revealing ν', the inside of that hash is still hidden to D.

We now first prove, in general, that it is safe to reveal some of the randomness inside the hash. More accurately, if you pre-specify some, but not all, of the sequences of coins to be chosen when interpreting a message m, then no single bitstring x is exceptionally likely to occur as the interpretation of m. After this lemma, we can prove the soundness result.

Lemma 5.5 *Let m be a message, $U \subsetneq \mathrm{R}(m)$. Let p be a positive polynomial. Then, for large enough η*

$$\forall \tau \in \mathrm{Coins}_\eta(U). \forall x \in \mathbf{Str} : \mathbb{P}[\alpha \leftarrow [\![m]\!]^\tau_\eta; \alpha = x] < \frac{1}{p(\eta)}.$$

Proof. The proof follows by induction on the structure of m.

- Consider the case $m = \langle m_1, m_2 \rangle$. We get by induction hypothesis that the statement holds either for m_1 or m_2, which suffices for the proof given that concatenating a bitstring might just lower the probability of hitting a particular element.
- The cases $m = \{\!|m_1|\!\}_k^r$ and $m = \Box^r$ are trivial due to well-spreadness (Definition 3.5).
- The cases $\boxtimes^r$ and $m = \mathrm{h}^r(m_1)$ follow from collision resistance (Definition 3.8).
- The case $m = c$ does not occur since U must be a proper subset of $\mathrm{R}(c) = \emptyset$.
- If m is a nonce n, then $U = \emptyset$ since it must be a proper subset of $\mathrm{R}(n) = \{n\}$. Then $\mathbb{P}[\alpha \leftarrow [\![n]\!]_\eta^\tau; \alpha = x] = \frac{1}{2^\eta} < \frac{1}{p(\eta)}$.
- If m is a key k, then again $U = \emptyset$. Suppose that for infinitely many η there is a $\tau \in \mathrm{Coins}_\eta(U)$ and an $x \in \mathbf{Str}$ for which

$$\mathbb{P}[\alpha \leftarrow [\![k]\!]_\eta^\tau; \alpha = x] = \mathbb{P}[\alpha \leftarrow \mathcal{K}(1^\eta); \alpha = x] \geq \frac{1}{p(\eta)}. \tag{1}$$

Now we build an adversary $A^{\mathcal{F}(-),\mathcal{G}(-)} \colon \mathbf{Param} \to \{0,1\}$ that breaks type-0 security.

```
algorithm  $A^{\mathcal{F}(-),\mathcal{G}(-)}(1^\eta)$:
  $\nu \leftarrow \mathcal{N}(1^\eta)$
  $\epsilon \leftarrow \mathcal{F}(\nu)$
  $\kappa \leftarrow \mathcal{K}(1^\eta)$
  if $\mathcal{D}(\kappa, \epsilon) = \nu$ return 1
  else return 0
```

This adversary generates a random nonce ν and gives it to the oracle $\mathcal{F}$ to encrypt. The adversary tries to guess if the oracle was instantiated with $\mathcal{E}(k, -)$ or with $\mathcal{E}(k, 0)$ by simply randomly generating a key itself and trying to decrypt. We will show that the probability that the oracle and the adversary choose the same key is non-negligible and hence the probability that this adversary guesses correctly is also non-negligible. Omitting $\mathcal{G}$ as it is not used by A, we get

$$\begin{aligned}
&\mathrm{Adv}_A(\eta)\\
&\quad = \mathbb{P}[\kappa \leftarrow \mathcal{K}(1^\eta); \nu \leftarrow \mathcal{N}(1^\eta); \epsilon \leftarrow \mathcal{E}(\kappa, \nu); \kappa' \leftarrow \mathcal{K}(1^\eta); \mathcal{D}(\kappa', \epsilon) = \nu]\\
&\qquad - \mathbb{P}[\kappa \leftarrow \mathcal{K}(1^\eta); \nu \leftarrow \mathcal{N}(1^\eta); \epsilon \leftarrow \mathcal{E}(\kappa, 0); \kappa' \leftarrow \mathcal{K}(1^\eta); \mathcal{D}(\kappa', \epsilon) = \nu]\\
&\quad \geq \mathbb{P}[\kappa, \kappa' \leftarrow \mathcal{K}(1^\eta); \kappa = \kappa']\\
&\qquad - \sum_{y \in \{0,1\}^\eta} \mathbb{P}[\nu \leftarrow \mathcal{N}(1^\eta); y = \nu]\\
&\qquad\qquad \cdot \mathbb{P}[\kappa, \kappa' \leftarrow \mathcal{K}(1^\eta); \epsilon \leftarrow \mathcal{E}(\kappa, 0); \mathcal{D}(\kappa', \epsilon) = y]\\
&\qquad \text{(because it is always possible to decrypt with the proper key)}\\
&\quad \geq \frac{1}{p(\eta)^2} - 2^{-\eta} \sum_{y \in \{0,1\}^\eta} \mathbb{P}[\kappa, \kappa' \leftarrow \mathcal{K}(1^\eta); \epsilon \leftarrow \mathcal{E}(\kappa, 0); \mathcal{D}(\kappa', \epsilon) = y]
\end{aligned}$$

(bounding the first term by the probability of getting x two times)

$$\geq \frac{1}{p(\eta)^2} - 2^{-\eta} \sum_{\kappa_0,\kappa'_0,\epsilon_0} \Bigg(\mathbb{P}[\kappa,\kappa' \leftarrow \mathcal{K}(1^\eta); \kappa = \kappa_0; \kappa' = \kappa'_0; \mathcal{E}(\kappa,0) = \epsilon_0] \cdot \sum_{y\in\{0,1\}^\eta} \mathbb{P}[\mathcal{D}(\kappa'_0,\epsilon_0) = y] \Bigg)$$

$$\geq \frac{1}{p(\eta)^2} - 2^{-\eta} \geq \frac{1}{p(\eta)^3} \quad \text{(for large enough } \eta\text{)}. \qquad \square$$

Theorem 5.6 *Let m be a message with a sub-message of the form $\mathrm{h}^r(\tilde{m})$. Assume that $\tilde{m} \notin \overline{m}$. Take $m' := m[\mathrm{h}^r(\tilde{m}) := \boxtimes^s]$, where $s = \mathcal{R}(\mathrm{h}^r(\tilde{m}))$. Then $[\![m]\!] \equiv [\![m']\!]$.*

Proof. Assume that $[\![m]\!] \not\equiv [\![m']\!]$, say A: **Param** $\times$ **Str** $\to \{0,1\}$ is a probabilistic polynomial-time adversary and p a positive polynomial such that

$$\frac{1}{p(\eta)} \leq \mathbb{P}[\mu \leftarrow [\![m]\!]_\eta; A(1^\eta,\mu) = 1] - \mathbb{P}[\mu \leftarrow [\![m']\!]_\eta; A(1^\eta,\mu) = 1] \qquad (2)$$

for infinitely many $\eta \in \mathbb{N}$. We will use this to build a distinguisher as in Definition 3.7 that breaks oracle indistinguishability of $\langle \mathcal{H}, \mathcal{V} \rangle$. $\square$

Let $\eta \in \mathbb{N}$, abbreviate $\mathrm{R}(m,\tilde{m}) \cap \mathrm{R}(\tilde{m})$ to U and let $\tau \in \mathrm{Coins}_\eta(U)$. Note that τ chooses coin flips for the shared randomness between the inside and the outside of the hash. Then define a probabilistic polynomial-time algorithm $D^\tau_\eta\colon \{0,1\}^* \to \{0,1\}$ as follows.

```
algorithm  $D^\tau_\eta(\alpha)$:
  $\mu \leftarrow [\![m]\!]^{\{\mathrm{h}^r(\tilde{m})\mapsto\alpha\},\tau}_\eta$
  $\beta \leftarrow A(\eta,\mu)$
  return $\beta$
```

This algorithm tries to guess if a given bitstring α was drawn from $[\![\mathrm{h}^r(\tilde{m})]\!]^\tau_\eta$ or from $[\![\boxtimes^s]\!]^\tau_\eta = [\![\mathrm{h}^s(n^s_\boxtimes)]\!]^\tau_\eta$. It does so by computing an interpretation for m as follows. The sub-message $\mathrm{h}^r(\tilde{m})$ is interpreted as α; the randomness that is shared between the inside of the hash ($\tilde{m}$) and the rest of the message is resolved using hard-coded sequences of coin flips τ. It then uses the adversary A to guess if the resulting interpretation was drawn from $[\![m]\!]_\eta$ (in which case it guesses that α was drawn from $[\![\mathrm{h}^r(\tilde{m})]\!]_\eta$) or from $[\![m']\!]_\eta$ (in which case it guesses that α was drawn from $[\![\boxtimes^s]\!]_\eta$).

Note that for every η and τ we have a different algorithm D^τ_η that has hardcoded coin flips for the shared randomness. However we do not have a single algorithm taking η and α as arguments since such an algorithm would need an infinite sequence of hardcoded coin flips.

Now consider one of the infinitely many values of η for which (2) holds.

Using D_η^τ we can rephrase (2) as follows:

$$\begin{aligned}
\frac{1}{p(\eta)} &\leq \mathbb{P}[\tau \leftarrow \mathrm{Coins}_\eta(U), \alpha \leftarrow [\![\mathrm{h}^r(\tilde{m})]\!]_\eta^\tau; D_\eta^\tau(\alpha) = 1] - \\
&\qquad\qquad \mathbb{P}[\tau \leftarrow \mathrm{Coins}_\eta(U), \alpha \leftarrow [\![\boxtimes^s]\!]_\eta^\tau; D_\eta^\tau(\alpha) = 1] \\
&= \sum_{\tau \in \mathrm{Coins}_\eta(U)} \Big(\mathbb{P}[\alpha \leftarrow [\![\mathrm{h}^r(\tilde{m})]\!]_\eta^\tau; D_\eta^\tau(\alpha) = 1] - \\
&\qquad\qquad \mathbb{P}[\alpha \leftarrow [\![\boxtimes^s]\!]_\eta^\tau; D_\eta^\tau(\alpha) = 1]\Big) \cdot \mathbb{P}[T \leftarrow \mathrm{Coins}_\eta(U); T = \tau] \\
&= \sum_{\tau \in \mathrm{Coins}_\eta(U)} \Big(\mathbb{P}[\alpha \leftarrow [\![\tilde{m}]\!]_\eta^\tau; D_\eta^\tau(\mathcal{H}(1^\eta, \alpha)) = 1] - \\
&\qquad\qquad \mathbb{P}[\alpha \leftarrow [\![n_\boxtimes^s]\!]_\eta^\tau; D_\eta^\tau(\mathcal{H}(1^\eta, \alpha)) = 1]\Big) \cdot \mathbb{P}[T \leftarrow \mathrm{Coins}_\eta(U); T = \tau] \\
&= \frac{1}{|\mathrm{Coins}_\eta(U)|} \sum_{\tau \in \mathrm{Coins}_\eta(U)} \Big(\mathbb{P}[\alpha \leftarrow [\![\tilde{m}]\!]_\eta^\tau; D_\eta^\tau(\mathcal{H}(1^\eta, \alpha)) = 1] - \\
&\qquad\qquad \mathbb{P}[\alpha \leftarrow [\![n_\boxtimes^s]\!]_\eta^\tau; D_\eta^\tau(\mathcal{H}(1^\eta, \alpha)) = 1]\Big).
\end{aligned}$$

Note that τ selects the randomness that is shared between the inside of the hash and the outside of the hash; when α is drawn from $[\![\tilde{m}]\!]_\eta^\tau$ the randomness that appears only inside the hash is chosen (and the assumption on $\tilde{m}$ means that there is really something to choose); $\mathcal{H}$ chooses the randomness for taking the hash; and D_η^τ itself resolves the randomness that appears only outside the hash. This means that there must be a particular value of τ, say $\bar{\tau}_\eta$, such that

$$\frac{1}{p(\eta)} \leq \mathbb{P}[\alpha \leftarrow [\![\tilde{m}]\!]_\eta^{\bar{\tau}_\eta}; D_\eta^{\bar{\tau}_\eta}(\mathcal{H}(1^\eta, \alpha)) = 1] - \mathbb{P}[\alpha \leftarrow [\![n_\boxtimes^s]\!]_\eta^{\bar{\tau}_\eta}; D_\eta^{\bar{\tau}_\eta}(\mathcal{H}(1^\eta, \alpha)) = 1]. \quad (3)$$

Gathering all $D_\eta^{\bar{\tau}_\eta}$ together for the various values of η, let D be the non-uniform adversary $\{D_\eta^{\bar{\tau}_\eta}\}_{\eta \in \mathbb{N}}$. Note that we have not actually defined $D_\eta^{\bar{\tau}_\eta}$ for all η, but only for those (infinitely many) for which (2) actually holds. What D does for the other values of η is irrelevant.

We will now show that D breaks the oracle indistinguishability of $\langle \mathcal{H}, \mathcal{V} \rangle$. For this, let $L = \{L_\eta\}_{\eta \in \mathbb{N}}$ be a polynomial size family of subsets of **Str**. We have to show that for infinitely many values of η, there are $x, y \in \mathbf{Str} \setminus L_\eta$ such that D meaningfully distinguishes between $\mathcal{H}(1^\eta, x)$ and $\mathcal{H}(1^\eta, y)$.

Once again, take one of the infinitely many values of η for which (2) holds. Continuing from (3), we get

$$\begin{aligned}
\frac{1}{p(\eta)} &\leq \sum_{\alpha \in [\![\tilde{m}]\!]_\eta^{\bar{\tau}_\eta}} \mathbb{P}[D_\eta^{\bar{\tau}_\eta}(\mathcal{H}(1^\eta, \alpha)) = 1] \cdot \mathbb{P}[[\![\tilde{m}]\!]_\eta^{\bar{\tau}_\eta} = \alpha] \\
&\qquad - \sum_{\beta \in [\![n_\boxtimes^s]\!]_\eta^{\bar{\tau}_\eta}} \mathbb{P}[D_\eta^{\bar{\tau}_\eta}(\mathcal{H}(1^\eta, \beta)) = 1] \cdot \mathbb{P}[[\![n_\boxtimes^s]\!]_\eta^{\bar{\tau}_\eta} = \beta] \\
&= \sum_{\substack{\alpha \in [\![\tilde{m}]\!]_\eta^{\bar{\tau}_\eta} \\ \beta \in [\![n_\boxtimes^s]\!]_\eta^{\bar{\tau}_\eta}}} \Big[\mathbb{P}[D_\eta^{\bar{\tau}_\eta}(\mathcal{H}(1^\eta, \alpha)) = 1] \cdot \mathbb{P}[[\![\tilde{m}]\!]_\eta^{\bar{\tau}_\eta} = \alpha] \cdot \mathbb{P}[[\![n_\boxtimes^s]\!]_\eta^{\bar{\tau}_\eta} = \beta] \\
&\qquad - \mathbb{P}[D_\eta^{\bar{\tau}_\eta}(\mathcal{H}(1^\eta, \beta)) = 1] \cdot \mathbb{P}[[\![\tilde{m}]\!]_\eta^{\bar{\tau}_\eta} = \alpha] \cdot \mathbb{P}[[\![n_\boxtimes^s]\!]_\eta^{\bar{\tau}_\eta} = \beta]
\end{aligned}$$

$$\left(\text{since} \sum_{\beta\in[\![n^s_\boxtimes]\!]_\eta^{\bar\tau_\eta}} \mathbb{P}[[\![n^s_\boxtimes]\!]_\eta^{\bar\tau_\eta}=\beta]=1 \text{ and } \sum_{\alpha\in[\![\tilde m]\!]_\eta^{\bar\tau_\eta}} \mathbb{P}[[\![\tilde m]\!]_\eta^{\bar\tau_\eta}=\alpha]=1\right)$$

$$\begin{aligned}
= &\sum_{\substack{\alpha\in[\![\tilde m]\!]_\eta^{\bar\tau_\eta}\\ \beta\in[\![n^s_\boxtimes]\!]_\eta^{\bar\tau_\eta}\cap L_\eta}} \Big[\big(\mathbb{P}[D_\eta^{\bar\tau_\eta}(\mathcal{H}(1^\eta,\alpha))=1]-\mathbb{P}[D_\eta^{\bar\tau_\eta}(\mathcal{H}(1^\eta,\beta))=1]\big)\\
&\qquad\cdot\mathbb{P}[[\![\tilde m]\!]_\eta^{\bar\tau_\eta}=\alpha]\cdot\mathbb{P}[[\![n^s_\boxtimes]\!]_\eta^{\bar\tau_\eta}=\beta]\Big]\\
&+\sum_{\substack{\alpha\in[\![\tilde m]\!]_\eta^{\bar\tau_\eta}\cap L_\eta\\ \beta\in[\![n^s_\boxtimes]\!]_\eta^{\bar\tau_\eta}\setminus L_\eta}} \Big[\big(\mathbb{P}[D_\eta^{\bar\tau_\eta}(\mathcal{H}(1^\eta,\alpha))=1]-\mathbb{P}[D_\eta^{\bar\tau_\eta}(\mathcal{H}(1^\eta,\beta))=1]\big)\\
&\qquad\cdot\mathbb{P}[[\![\tilde m]\!]_\eta^{\bar\tau_\eta}=\alpha]\cdot\mathbb{P}[[\![n^s_\boxtimes]\!]_\eta^{\bar\tau_\eta}=\beta]\Big]\\
&+\sum_{\substack{\alpha\in[\![\tilde m]\!]_\eta^{\bar\tau_\eta}\setminus L_\eta\\ \beta\in[\![n^s_\boxtimes]\!]_\eta^{\bar\tau_\eta}\setminus L_\eta}} \Big[\big(\mathbb{P}[D_\eta^{\bar\tau_\eta}(\mathcal{H}(1^\eta,\alpha))=1]-\mathbb{P}[D_\eta^{\bar\tau_\eta}(\mathcal{H}(1^\eta,\beta))=1]\big)\\
&\qquad\cdot\mathbb{P}[[\![\tilde m]\!]_\eta^{\bar\tau_\eta}=\alpha]\cdot\mathbb{P}[[\![n^s_\boxtimes]\!]_\eta^{\bar\tau_\eta}=\beta]\Big]\\
&\qquad(\text{splitting cases on } \in L_\eta \text{ and } \notin L_\eta)\\
\le &\sum_{\substack{\alpha\in[\![\tilde m]\!]_\eta^{\bar\tau_\eta}\\ \beta\in[\![n^s_\boxtimes]\!]_\eta^{\bar\tau_\eta}\cap L_\eta}} \mathbb{P}[D_\eta^{\bar\tau_\eta}(\mathcal{H}(1^\eta,\alpha))=1]\cdot\mathbb{P}[[\![\tilde m]\!]_\eta^{\bar\tau_\eta}=\alpha]\cdot\mathbb{P}[[\![n^s_\boxtimes]\!]_\eta^{\bar\tau_\eta}=\beta]\\
&+\sum_{\substack{\alpha\in[\![\tilde m]\!]_\eta^{\bar\tau_\eta}\cap L_\eta\\ \beta\in[\![n^s_\boxtimes]\!]_\eta^{\bar\tau_\eta}\setminus L_\eta}} \mathbb{P}[D_\eta^{\bar\tau_\eta}(\mathcal{H}(1^\eta,\alpha))=1]\cdot\mathbb{P}[[\![\tilde m]\!]_\eta^{\bar\tau_\eta}=\alpha]\cdot\mathbb{P}[[\![n^s_\boxtimes]\!]_\eta^{\bar\tau_\eta}=\beta]\\
&+\sum_{\substack{\alpha\in[\![\tilde m]\!]_\eta^{\bar\tau_\eta}\setminus L_\eta\\ \beta\in[\![n^s_\boxtimes]\!]_\eta^{\bar\tau_\eta}\setminus L_\eta}} \Big[\big(\mathbb{P}[D_\eta^{\bar\tau_\eta}(\mathcal{H}(1^\eta,\alpha))=1]-\mathbb{P}[D_\eta^{\bar\tau_\eta}(\mathcal{H}(1^\eta,\beta))=1]\big)\\
&\qquad\cdot\mathbb{P}[[\![\tilde m]\!]_\eta^{\bar\tau_\eta}=\alpha]\cdot\mathbb{P}[[\![n^s_\boxtimes]\!]_\eta^{\bar\tau_\eta}=\beta]\Big]\\
&\qquad(\text{since } \mathbb{P}[D_\eta^{\bar\tau_\eta}(\mathcal{H}(1^\eta,\beta))=1]\ge 0)\\
\le &\sum_{\beta\in[\![n^s_\boxtimes]\!]_\eta^{\bar\tau_\eta}\cap L_\eta} \mathbb{P}[[\![n^s_\boxtimes]\!]_\eta^{\bar\tau_\eta}=\beta]+\sum_{\alpha\in[\![\tilde m]\!]_\eta^{\bar\tau_\eta}\cap L_\eta}\mathbb{P}[[\![\tilde m]\!]_\eta^{\bar\tau_\eta}=\alpha]\\
&+\sum_{\substack{\alpha\in[\![\tilde m]\!]_\eta^{\bar\tau_\eta}\setminus L_\eta\\ \beta\in[\![n^s_\boxtimes]\!]_\eta^{\bar\tau_\eta}\setminus L_\eta}} \Big[\big(\mathbb{P}[D_\eta^{\bar\tau_\eta}(\mathcal{H}(1^\eta,\alpha))=1]-\mathbb{P}[D_\eta^{\bar\tau_\eta}(\mathcal{H}(1^\eta,\beta))=1]\big)\\
&\qquad\cdot\mathbb{P}[[\![\tilde m]\!]_\eta^{\bar\tau_\eta}=\alpha]\cdot\mathbb{P}[[\![n^s_\boxtimes]\!]_\eta^{\bar\tau_\eta}=\beta]\Big]\\
&(\text{since}\,\mathbb{P}[D_\eta^{\bar\tau_\eta}(\mathcal{H}(1^\eta,\alpha))=1]\le 1, \sum_{\alpha\in[\![m]\!]_\eta^{\bar\tau_\eta}}\mathbb{P}[[\![\tilde m]\!]_\eta^{\bar\tau_\eta}=\alpha]=1 \text{ and } \sum_{\beta\in[\![n^s_\boxtimes]\!]_\eta^{\bar\tau_\eta}\setminus L_\eta}\mathbb{P}[[\![n^s_\boxtimes]\!]_\eta^{\bar\tau_\eta}=\beta]\le 1)\\
\le &\frac{1}{2p(\eta)}+\sum_{\substack{\alpha\in[\![\tilde m]\!]_\eta^{\bar\tau_\eta}\setminus L_\eta\\ \beta\in[\![n^s_\boxtimes]\!]_\eta^{\bar\tau_\eta}\setminus L_\eta}} \Big[\Big(\mathbb{P}[D_\eta^{\bar\tau_\eta}(\mathcal{H}(1^\eta,\alpha))=1]-\mathbb{P}[D_\eta^{\bar\tau_\eta}(\mathcal{H}(1^\eta,\beta))=1]\Big)\\
&\qquad\cdot\mathbb{P}[[\![\tilde m]\!]_\eta^{\bar\tau_\eta}=\alpha]\cdot\mathbb{P}[[\![n^s_\boxtimes]\!]_\eta^{\bar\tau_\eta}=\beta]\Big]. \qquad (4)\\
&(\text{by Lemma 5.5 (2x) using the polynomial } 4p(\eta)|L_\eta|, \text{ provided that } \eta \text{ is large})
\end{aligned}$$

Now suppose that for all $\alpha\in[\![\tilde m]\!]_\eta^{\bar\tau_\eta}\setminus L_\eta$ and all $\beta\in[\![n^s_\boxtimes]\!]_\eta^{\bar\tau_\eta}\setminus L_\eta$ we have

$$\mathbb{P}[D_\eta^{\bar\tau_\eta}(\mathcal{H}(1^\eta,\alpha))=1]-\mathbb{P}[D_\eta^{\bar\tau_\eta}(\mathcal{H}(1^\eta,\beta))=1]<\frac{1}{2p(\eta)}.$$

Then, continuing from (4), we get a contradiction:

$$\begin{aligned}
\frac{1}{p(\eta)} &< \frac{1}{2p(\eta)} + \sum_{\substack{\alpha \in [\![\tilde{m}]\!]_\eta^{\bar{\tau}_\eta} \setminus L_\eta \\ \beta \in [\![n^s_\boxtimes]\!]_\eta^{\bar{\tau}_\eta} \setminus L_\eta}} \frac{1}{2p(\eta)} \cdot \mathbb{P}[[\![\tilde{m}]\!]_\eta^{\bar{\tau}_\eta} = \alpha] \cdot \mathbb{P}[[\![n^s_\boxtimes]\!]_\eta^{\bar{\tau}_\eta} = \beta] \\
&= \frac{1}{2p(\eta)} + \frac{1}{2p(\eta)} \sum_{\substack{\alpha \in [\![\tilde{m}]\!]_\eta^{\bar{\tau}_\eta} \setminus L_\eta \\ \beta \in [\![n^s_\boxtimes]\!]_\eta^{\bar{\tau}_\eta} \setminus L_\eta}} \mathbb{P}[[\![\tilde{m}]\!]_\eta^{\bar{\tau}_\eta} = \alpha] \cdot \mathbb{P}[[\![n^s_\boxtimes]\!]_\eta^{\bar{\tau}_\eta} = \beta] \\
&\leq \frac{1}{2p(\eta)} + \frac{1}{2p(\eta)} .
\end{aligned}$$

Therefore, there must be an $\alpha \in [\![\tilde{m}]\!]_\eta^{\bar{\tau}_\eta} \setminus L_\eta$ and a $\beta \in [\![n^s_\boxtimes]\!]_\eta^{\bar{\tau}_\eta} \setminus L_\eta$ such that

$$\frac{1}{2p(\eta)} \leq \mathbb{P}[D_\eta^{\bar{\tau}_\eta}(\mathcal{H}(1^\eta, \alpha)) = 1] - \mathbb{P}[D_\eta^{\bar{\tau}_\eta}(\mathcal{H}(1^\eta, \beta)) = 1].$$

Hence D breaks oracle indistinguishability, contradicting the assumption on $\langle \mathcal{H}, \mathcal{V} \rangle$. □

Theorem 5.7 (Soundness) *Let m and m' be acyclic messages. Then $m \cong m' \implies [\![m]\!] \equiv [\![m']\!]$.*

Proof. The assumption that $m \cong m'$ means that $\mathit{pattern}(m) \approx \mathit{pattern}(m')$. Therefore we get $[\![\mathit{pattern}(m)]\!] = [\![\mathit{pattern}(m')]\!]$. Next, by Lemma 5.3 we get $[\![\mathit{encpat} \circ \mathit{hashpat}(m)]\!] = [\![\mathit{encpat} \circ \mathit{hashpat}(m')]\!]$. Now, by applying Theorem 5.4 two times, we obtain $[\![\mathit{hashpat}(m)]\!] \equiv [\![\mathit{hashpat}(m')]\!]$. Finally, we start with m and repeatedly apply Theorem 5.6, replacing one hash at a time by $\boxtimes$, and arrive at $\mathit{hashpat}(m)$. This shows that $[\![m]\!] \equiv [\![\mathit{hashpat}(m)]\!]$ and similarly $[\![m']\!] \equiv [\![\mathit{hashpat}(m')]\!]$. Therefore $[\![m]\!] \equiv [\![m']\!]$. □

6 Completeness

Although soundness results allow us to port proofs of *secrecy* properties from the symbolic world to the computational world, it does not permit to port, for instance, *authenticity* and *integrity* results. For example, consider a protocol in which an agent A chooses a nonce n and commits to this nonce by sending $h(n)$ to another agent B. Later in the protocol, A will reveal the nonce n by sending n itself to B. Security in this setting means that A cannot change her choice after sending $h(n)$. In the symbolic world, this is guaranteed by the fact that the message $h(n)n$ (the concatenation of the relevant messages in the protocol run) is observationally distinct from $h(n)n'$, with $n' \neq n$. We would like to be able to conclude from this symbolic property that $[\![h(n)n]\!]$ is computationally distinct from $[\![h(n)n']\!]$, since that is needed to guarantee the security in the computational world.

What is needed here is completeness: computational equivalence of $[\![m]\!]$ and $[\![m']\!]$ should imply observational equivalence of m and m'. For the original Abadi-Rogaway logic, completeness under appropriate conditions on the encryption scheme was proven by Micciancio and Warinschi [MW04a].

This section now shows that the interpretation proposed in the Section 4 is complete. Throughout this section we assume that the encryption scheme $\langle \mathcal{K}, \mathcal{E}, \mathcal{D} \rangle$ is type-0 secure, well-spread, and confusion free, and that the probabilistic hash scheme $\langle \mathcal{H}, \mathcal{V} \rangle$ is collision resistant and oracle indistinguishable.

Throughout the completeness proof we follow the steps of Micciancio–Warinschi and their notation when possible. We recall here some of their results as they are used in our proof.

In the original Abadi-Rogaway logic, the useful information for an adversary is determined by the set of keys she can learn. We define the function *arecover* and its computational counterpart *brecover* as in [MW04a]. There they are called *recoverable* and *getkeys*. These functions extract the set of keys observable by an adversary from a symbolic message and a bitstring respectively.

Definition 6.1 The function *arecover*: $\mathbf{Msg} \to \mathcal{P}(\mathbf{Key})$ is defined by

$$arecover(m) = arecover(m, |m|)$$

where *arecover*: $\mathbf{Msg} \times \mathbb{N} \to \mathcal{P}(\mathbf{Key})$ is given by

$$\begin{aligned} arecover(m, 0) &= \emptyset \\ arecover(m, d+1) &= \mathrm{F}_{\mathrm{kr}}(m, arecover(m, d)) \end{aligned}$$

and $\mathrm{F}_{\mathrm{kr}}\colon \mathbf{Msg} \times \mathcal{P}(\mathbf{Key}) \to \mathcal{P}(\mathbf{Key})$ is given by

$$\begin{aligned} \mathrm{F}_{\mathrm{kr}}(\langle m_1, m_2 \rangle, U) &= \mathrm{F}_{\mathrm{kr}}(m_1, U) \cup \mathrm{F}_{\mathrm{kr}}(m_2, U) \\ \mathrm{F}_{\mathrm{kr}}(k, U) &= \{k\} \cup U \\ \mathrm{F}_{\mathrm{kr}}(\{\!|m|\!\}_k^r, U) &= \mathrm{F}_{\mathrm{kr}}(m, U), && \text{if } k \in U; \\ \mathrm{F}_{\mathrm{kr}}(m, U) &= U, && \text{in any other case.} \end{aligned}$$

The function *brecover*: $\mathbf{Str} \to \mathcal{P}(\mathbf{Keys})$ is defined by

```
algorithm brecover(μ) :
Gets all the keys in the bitstring μ
with high probability.
  $U' := \emptyset$
  do:
    $U := U'$
    $U' := \mathrm{C}_{\mathrm{kr}}(\mu, U)$
  until $U = U'$
  return $U$
```

where $\mathrm{C_{kr}}\colon \mathbf{Msg} \times \mathcal{P}(\mathbf{Keys}) \to \mathcal{P}(\mathbf{Keys})$ is defined by

$$\begin{aligned}
\mathrm{C_{kr}}(\kappa, U) &= \{\kappa\} \cup U \\
\mathrm{C_{kr}}(\mu_1\mu_2, U) &= \mathrm{C_{kr}}(\mu_1, U) \cup \mathrm{C_{kr}}(\mu_2, U) \\
\mathrm{C_{kr}}(\epsilon, U) &= \mathrm{C_{kr}}(\mu, U), \text{if } \exists!\kappa \in U \text{ s.t. } \mathcal{D}(\epsilon, \kappa) = \mu \neq \bot; \\
\mathrm{C_{kr}}(\mu, U) &= U, \quad \text{otherwise.}
\end{aligned}$$

Note that *brecover* can be computed in polynomial time: we can assume without loss of generality that the size of the output of the decryption algorithm is smaller than the input ciphertext (since the encryption scheme is randomized). Then $\mathrm{C_{kr}}$ recurses a linear number of times and every iteration of the until loop of *brecover* adds at least one key to U' and therefore the number of iterations is bounded by the maximum number of keys in μ.

The following lemma also from [MW04a] shows the relation between these two functions.

Lemma 6.2 *Let $\Pi = \langle \mathcal{K}, \mathcal{E}, \mathcal{D} \rangle$ be a confusion free encryption scheme and let $m \in \mathbf{Msg}$. Then*

$$\mathbb{P}[\tau \leftarrow \mathrm{Coins}_\eta(m); \mathit{brecover}([\![m]\!]_\eta^\tau) \neq [\![\mathit{arecover}(m)]\!]_\eta^\tau]$$

is a negligible function of η.

Proof. We refer the reader to the original paper for a complete proof of this lemma. The hashes that appear in our logic have no influence at all. □

In our extended logic, due to the hashes, it is not true any more that the only useful information for an adversary is the set of keys. Any message an adversary can learn might be the pre-image of a certain hash value. Therefore, we need to be able to compute the closure (up to a certain size) of a given message or bitstring. For this reason the closure operators defined below are closed (up to a certain size) under pairing but not under encryption or hashing.

Definition 6.3 The function *aclosure*: $\mathbf{Msg} \times \mathbb{N} \to \mathcal{P}(\mathbf{Msg})$ computes the messages in the symbolic closure of a message, up to a certain size d:

$$\mathit{aclosure}(m, d) = \mathit{aclosure}(m, d, \mathit{arecover}(m))$$

where *aclosure*: $\mathbf{Msg} \times \mathbb{N} \times \mathcal{P}(\mathbf{Key}) \to \mathcal{P}(\mathbf{Msg})$

$$\mathit{aclosure}(m, d, U) = \mathit{asynth}(\mathit{aanalz}(m, U), d).$$

Here the function *aanalz*: $\mathbf{Msg} \times \mathcal{P}(\mathbf{Msg}) \to \mathcal{P}(\mathbf{Msg})$, defined below, splits a message in all its meaningful subterms, using the keys in U, when possible, for decrypting.

$$\mathit{aanalz}(\langle m_1, m_2 \rangle, U) = \mathit{aanalz}(m_1, U) \cup \mathit{aanalz}(m_2, U)$$

$$\begin{aligned}
aanalz(\{\!|m|\!\}_k^r, U) &= \{\{\!|m|\!\}_k^r\} \cup aanalz(m, U), && \text{if } k \in U;\\
aanalz(m, U) &= \{m\}, && \text{in any other case.}
\end{aligned}$$

The function *asynth*: $\mathcal{P}(\mathbf{Msg}) \times \mathbb{N} \to \mathcal{P}(\mathbf{Msg})$ generates all possible vectors of messages in U of size up to d:

```
algorithm $asynth(U, d)$ :
  $U_1 = U$
  for $i = 2$ to $d$
    $U_i := \emptyset$
    for each $m \in U$
      for each $v \in U_{i-1}$
        $U_i := U_i \cup \{\langle m, v\rangle\}$
  return $U_d$
```

```
algorithm $bsynth(U, d)$ :
  $U_1 = U$
  for $i = 2$ to $d$
    $U_i := \emptyset$
    for each $\mu \in U$
      for each $\omega \in U_{i-1}$
        $U_i := U_i \cup \{\mu\omega\}$
  return $U_d$
```

Next, the functions *bclosure*: $\mathbf{Str} \times \mathbb{N} \to \mathcal{P}(\mathbf{Str})$, *banalz*: $\mathbf{Str} \times \mathcal{P}(\mathbf{Str}) \to \mathcal{P}(\mathbf{Str})$ and *bsynth*: $\mathcal{P}(\mathbf{Str}) \times \mathbb{N} \to \mathcal{P}(\mathbf{Str})$ are the computational counterparts of *aclosure*, *aanalz* and *asynth* respectively:

$$bclosure(\mu, d) = bclosure(\mu, d, brecover(\mu))$$

where *bclosure*: $\mathbf{Str} \times \mathbb{N} \times \mathcal{P}(\mathbf{Keys}) \to \mathcal{P}(\mathbf{Str})$ is defined by

$$bclosure(\mu, d, U) = bsynth(banalz(\mu, U), d)$$

and

$$\begin{aligned}
banalz(\mu_1\mu_2, U) &= banalz(\mu_1, U) \cup banalz(\mu_2, U)\\
banalz(\epsilon, U) &= \{\epsilon\} \cup banalz(\mu, U), && \text{if } \exists!\kappa \in U s.t. \mathcal{D}(\epsilon, \kappa) = \mu \neq \bot;\\
banalz(\mu, U) &= \{\mu\}, && \text{in any other case.}
\end{aligned}$$

Note that for a fixed $d \in \mathbb{N}$, $bclosure(\mu, d)$ can be computed in a time that is polynomial in $|\mu|$.

Now we show that the proposed functions behave similarly with high probability.

Lemma 6.4 *Let $m \in \mathbf{Msg}$, $d \in \mathbb{N}$ and $T \subseteq \mathbf{Key}$. Then the probability*

$$\mathbb{P}\left[\tau \leftarrow \mathrm{Coins}_\eta(m); bclosure([\![m]\!]_\eta^\tau, d, [\![T]\!]_\eta^\tau) \neq [\![aclosure(m, d, T)]\!]_\eta^\tau\right]$$

is a negligible function of η.

Proof. We prove by induction on the structure of m that the probability

$$\mathbb{P}\left[\tau \leftarrow \mathrm{Coins}_\eta(m); banalz([\![m]\!]_\eta^\tau, [\![T]\!]_\eta^\tau) \neq [\![aanalz(m, T)]\!]_\eta^\tau\right]$$

is a negligible function of η. The original statement follows from this, using that *asynth* and *bsynth* have similar behavior.

The only non-trivial case is that of $m = \{\!|m_1|\!\}^r_k$.

• If $k \in T$, then $[\![k]\!]^\tau_\eta \in [\![T]\!]^\tau_\eta$. Next

$$\begin{aligned}
&\mathbb{P}[\tau \leftarrow \mathrm{Coins}_\eta(m); \mathit{banalz}([\![m]\!]^\tau_\eta, [\![T]\!]^\tau_\eta) = [\![\mathit{aanalz}(m,T)]\!]^\tau_\eta] \\
&\quad = \mathbb{P}[\tau \leftarrow \mathrm{Coins}_\eta(m); \mathit{banalz}([\![m]\!]^\tau_\eta, [\![T]\!]^\tau_\eta) = [\![\{m\} \cup \mathit{aanalz}(m_1,T)]\!]^\tau_\eta] \\
&\quad \geq \mathbb{P}[\tau \leftarrow \mathrm{Coins}_\eta(m); [\![m]\!]^\tau_\eta \cup \mathit{banalz}([\![m_1]\!]^\tau_\eta, [\![T]\!]^\tau_\eta) = [\![\{m\} \cup \mathit{aanalz}(m_1,T)]\!]^\tau_\eta \\
&\qquad\qquad \wedge \forall \kappa \in [\![T \setminus k]\!]^\tau_\eta : \mathcal{D}([\![m]\!]^\tau_\eta, \kappa) = \bot] \\
&\quad \geq \mathbb{P}[\tau \leftarrow \mathrm{Coins}_\eta(m); \mathit{banalz}([\![m_1]\!]^\tau_\eta, [\![T]\!]^\tau_\eta) = [\![\mathit{aanalz}(m_1,T)]\!]^\tau_\eta \\
&\qquad\qquad \wedge \forall \kappa \in [\![T \setminus k]\!]^\tau_\eta : \mathcal{D}([\![m]\!]^\tau_\eta, \kappa) = \bot] \\
&\quad \geq 1 - (\mathbb{P}[\tau \leftarrow \mathrm{Coins}_\eta(m); \mathit{banalz}([\![m_1]\!]^\tau_\eta, [\![T]\!]^\tau_\eta) \neq [\![\mathit{aanalz}(m_1,T)]\!]^\tau_\eta \\
&\qquad\qquad \vee \exists \kappa \in [\![T \setminus k]\!]^\tau_\eta : \mathcal{D}([\![m]\!]^\tau_\eta, \kappa) \neq \bot]) \\
&\quad \geq 1 - (\mathbb{P}[\tau \leftarrow \mathrm{Coins}_\eta(m); \mathit{banalz}([\![m_1]\!]^\tau_\eta, [\![T]\!]^\tau_\eta) \neq [\![\mathit{aanalz}(m_1,T)]\!]^\tau_\eta] \\
&\qquad\qquad + \mathbb{P}[\tau \leftarrow \mathrm{Coins}_\eta(m); \exists \kappa \in [\![T \setminus k]\!]^\tau_\eta : \mathcal{D}([\![m]\!]^\tau_\eta, \kappa) \neq \bot]) \\
&\quad \geq 1 - (\varepsilon_1(\eta) + \sum_{\kappa \in [\![T \setminus k]\!]^\tau_\eta} \mathbb{P}[\tau \leftarrow \mathrm{Coins}_\eta(m); \mathcal{D}([\![m]\!]^\tau_\eta, \kappa) \neq \bot]) \\
&\quad \geq 1 - (\varepsilon_1(\eta) + \varepsilon_2(\eta) \cdot (|T| - 1)),
\end{aligned}$$

where $\varepsilon_1, \varepsilon_2$ are the negligible functions from the induction hypothesis and confusion freeness respectively.

• If $k \notin T$, then $[\![k]\!]^\tau_\eta \notin [\![T]\!]^\tau_\eta$. Next

$$\begin{aligned}
&\mathbb{P}[\tau \leftarrow \mathrm{Coins}_\eta(m); \mathit{banalz}([\![m]\!]^\tau_\eta, [\![T]\!]^\tau_\eta) = [\![\mathit{aanalz}(m,T)]\!]^\tau_\eta] \\
&\quad = \mathbb{P}[\tau \leftarrow \mathrm{Coins}_\eta(m); \mathit{banalz}([\![m]\!]^\tau_\eta, [\![T]\!]^\tau_\eta) = [\![\{m\}]\!]^\tau_\eta] \\
&\quad \geq \mathbb{P}[\tau \leftarrow \mathrm{Coins}_\eta(m); \mathit{banalz}([\![m]\!]^\tau_\eta, [\![T]\!]^\tau_\eta) = [\![\{m\}]\!]^\tau_\eta \\
&\qquad\qquad \wedge \forall \kappa \in [\![T]\!]^\tau_\eta : \mathcal{D}([\![m]\!]^\tau_\eta, \kappa) = \bot] \\
&\quad = \mathbb{P}[\tau \leftarrow \mathrm{Coins}_\eta(m); [\![m]\!]^\tau_\eta = [\![\{m\}]\!]^\tau_\eta \wedge \forall \kappa \in [\![T]\!]^\tau_\eta : \mathcal{D}([\![m]\!]^\tau_\eta, \kappa) = \bot] \\
&\quad = 1 - \mathbb{P}[\tau \leftarrow \mathrm{Coins}_\eta(m); \exists \kappa \in [\![T]\!]^\tau_\eta : \mathcal{D}([\![m]\!]^\tau_\eta, \kappa) = \bot] \\
&\quad \geq 1 - \sum_{\kappa \in [\![T]\!]^\tau_\eta} \mathbb{P}[\tau \leftarrow \mathrm{Coins}_\eta(m); \mathcal{D}([\![m]\!]^\tau_\eta, \kappa) = \bot] \\
&\quad \geq 1 - \varepsilon(\eta) \cdot |T|,
\end{aligned}$$

where ε is a negligible function due to confusion freeness. □

The following is an extended version of the function *psp* from [MW04a], which is the computational counterpart of *pattern*. This function takes a bitstring as an argument and tries to recover the pattern associated to it. This means that given as input a sample from $[\![m]\!]$, the function outputs (a renaming of) *pattern*(m) with high probability. As in [MW04a] we let f be an arbitrary (but fixed) injective function that associates an identifier (i.e., an element of $\mathbf{Nonce} \cup \mathbf{Key} \cup \mathbf{Const}$) to each bitstring of primitive type (i.e., ν, κ, ς).

Definition 6.5 The function $psp\colon \mathbf{Str} \times \mathcal{P}(\mathbf{Str}) \to \mathbf{Msg}$ is defined by

$$\begin{aligned}
psp(\mu_1\mu_2, U) &= \langle psp(\mu_1, U), psp(\mu_2, U)\rangle \\
psp(\epsilon, U) &= \begin{cases} \{\![psp(\mathcal{D}(\epsilon,\kappa), U)]\!\}^{\mathcal{R}(\epsilon)}_{f(\kappa)}, & \text{if } \exists!\kappa \in U \text{ s.t. } \mathcal{D}(\epsilon,\kappa) \neq \perp; \\ \Box^{\mathcal{R}(\epsilon)}, & \text{otherwise.} \end{cases} \\
psp(\psi, U) &= \begin{cases} \mathrm{h}^{\mathcal{R}(\psi)}(psp(\mu, U)), & \text{if } \exists!\mu \in U \text{ s.t. } \mathcal{V}(\mu,\psi) = 1; \\ \boxtimes^{\mathcal{R}(\psi)}, & \text{otherwise.} \end{cases} \\
psp(\mu, U) &= f(\mu) \qquad \text{in any other case.}
\end{aligned}$$

Theorem 6.6 *Let $m \in \mathbf{Msg}$ and let U be a finite subset of $\mathbf{Msg}$. Then the probability*

$$\mathbb{P}\left[\tau \leftarrow \mathrm{Coins}_\eta(m); psp([\![m]\!]^\tau_\eta, [\![U]\!]^\tau_\eta) \not\approx pattern(m, U)\right]$$

is a negligible function of η.

Proof. The proof follows by induction on the structure of m. We only show here the case $m = \mathrm{h}^r(m_1)$. For the remaining cases, the proof follows similarly to the one in the original Micciancio-Warinschi [MW04a] paper and therefore we refer the reader to it.
• If $m_1 \in U$ then

$$\begin{aligned}
&\mathbb{P}[\tau \leftarrow \mathrm{Coins}_\eta(m); psp([\![m]\!]^\tau_\eta, [\![U]\!]^\tau_\eta) \approx pattern(m, U)] \\
&\quad \geq \mathbb{P}[\tau \leftarrow \mathrm{Coins}_\eta(m); psp([\![m]\!]^\tau_\eta, [\![U]\!]^\tau_\eta) \approx pattern(m, U) \\
&\qquad\qquad \wedge \forall \mu \in [\![U \setminus \{m_1\}]\!]^\tau_\eta : \mathcal{V}(\mu, [\![m]\!]^\tau_\eta) = 0] \\
&\quad = \mathbb{P}[\tau \leftarrow \mathrm{Coins}_\eta(m); psp([\![m_1]\!]^\tau_\eta, [\![U]\!]^\tau_\eta) \approx pattern(m_1, U) \\
&\qquad\qquad \wedge \forall \mu \in [\![U \setminus \{m_1\}]\!]^\tau_\eta : \mathcal{V}(\mu, [\![m]\!]^\tau_\eta) = 0] \\
&\quad = 1 - \mathbb{P}[\tau \leftarrow \mathrm{Coins}_\eta(m); psp([\![m_1]\!]^\tau_\eta, [\![U]\!]^\tau_\eta) \not\approx pattern(m_1, U) \\
&\qquad\qquad \vee \exists \mu \in [\![U \setminus \{m_1\}]\!]^\tau_\eta : \mathcal{V}(\mu, [\![m]\!]^\tau_\eta) = 1] \\
&\quad \geq 1 - \mathbb{P}\left[\tau \leftarrow \mathrm{Coins}_\eta(m); psp([\![m_1]\!]^\tau_\eta, [\![U]\!]^\tau_\eta) \not\approx pattern(m_1, U)\right] \\
&\qquad\qquad + \mathbb{P}\left[\tau \leftarrow \mathrm{Coins}_\eta(m); \exists \mu \in [\![U \setminus \{m_1\}]\!]^\tau_\eta : \mathcal{V}(\mu, [\![m]\!]^\tau_\eta) = 1\right] \\
&\quad \geq 1 - \varepsilon_1(\eta) - \varepsilon_2(\eta)\,,
\end{aligned}$$

where ε_1 is the negligible function from the induction hypothesis and ε_2 is a negligible function from collision resistance, using that an adversary can compute $[\![U \setminus \{m_1\}]\!]^\tau_\eta$.

• If $m_1 \notin U$, then

$$\begin{aligned}&\mathbb{P}[\tau \leftarrow \mathrm{Coins}_\eta(m); \mathit{psp}([\![m]\!]^\tau_\eta, [\![U]\!]^\tau_\eta) \approx \mathit{pattern}(m,U)]\\ &\qquad = \mathbb{P}[\mathit{psp}([\![m]\!]^\tau_\eta, [\![U]\!]^\tau_\eta) \approx \boxtimes^r] = \mathbb{P}[\forall \mu \in [\![U]\!]^\tau_\eta : \mathcal{V}(\mu, [\![m]\!]^\tau_\eta) = 0].\end{aligned}$$

Therefore

$$\begin{aligned}&\mathbb{P}[\tau \leftarrow \mathrm{Coins}_\eta(m); \mathit{psp}([\![m]\!]^\tau_\eta, [\![U]\!]^\tau_\eta) \not\approx \mathit{pattern}(m,U)]\\ &\qquad = \mathbb{P}[\exists \mu \in [\![U]\!]^\tau_\eta : \mathcal{V}(\mu, [\![m]\!]^\tau_\eta) = 1] \le \varepsilon(\eta)\,,\end{aligned}$$

where ε is a negligible function due to collision resistance. □

Lemma 6.7 *Let $m \in \mathbf{Msg}$ and $d \in \mathbb{N}$. Then the probability*

$$\mathbb{P}\left[\mu \leftarrow [\![m]\!]; \mathit{psp}(\mu, \mathit{bclosure}(\mu, d)) \not\approx \mathit{pattern}(m, \mathit{aclosure}(m, d))\right]$$

is a negligible function of η.

Proof.

$$\begin{aligned}
&\mathbb{P}\big[\mu \leftarrow [\![m]\!]; \mathit{psp}(\mu, \mathit{bclosure}(\mu, d)) \approx \mathit{pattern}(m, \mathit{aclosure}(m, d))\big]\\
&\quad = \mathbb{P}\big[\tau \leftarrow \mathrm{Coins}_\eta(m); \mathit{psp}([\![m]\!]^\tau_\eta, \mathit{bclosure}([\![m]\!]^\tau_\eta, d)) \approx \mathit{pattern}(m, \mathit{aclosure}(m, d))\big]\\
&\quad \ge \mathbb{P}\big[\tau \leftarrow \mathrm{Coins}_\eta(m); \mathit{psp}([\![m]\!]^\tau_\eta, \mathit{bclosure}([\![m]\!]^\tau_\eta, d)) \approx \mathit{pattern}(m, \mathit{aclosure}(m, d))\\
&\qquad\qquad \wedge\, \mathit{bclosure}([\![m]\!]^\tau_\eta, d) = [\![\mathit{aclosure}(m, d)]\!]^\tau_\eta\big]\\
&\quad = \mathbb{P}\big[\tau \leftarrow \mathrm{Coins}_\eta(m); \mathit{psp}([\![m]\!]^\tau_\eta, [\![\mathit{aclosure}(m, d)]\!]^\tau_\eta) \approx \mathit{pattern}(m, \mathit{aclosure}(m, d))\\
&\qquad\qquad \wedge\, \mathit{bclosure}([\![m]\!]^\tau_\eta, d) = [\![\mathit{aclosure}(m, d)]\!]^\tau_\eta\big]\\
&\quad \ge 1 - \mathbb{P}\big[\tau \leftarrow \mathrm{Coins}_\eta(m); \mathit{psp}([\![m]\!]^\tau_\eta, [\![\mathit{aclosure}(m, d)]\!]^\tau_\eta) \not\approx \mathit{pattern}(m, \mathit{aclosure}(m, d))\big]\\
&\qquad\qquad - \mathbb{P}\big[\tau \leftarrow \mathrm{Coins}_\eta(m); \mathit{bclosure}([\![m]\!]^\tau_\eta, d) \ne [\![\mathit{aclosure}(m, d)]\!]^\tau_\eta\big]\\
&\quad \ge 1 - \varepsilon_1(\eta) - \mathbb{P}\big[\tau \leftarrow \mathrm{Coins}_\eta(m); \mathit{bclosure}([\![m]\!]^\tau_\eta, d, \mathit{brecover}([\![m]\!]^\tau_\eta))\\
&\qquad\qquad \ne [\![\mathit{aclosure}(m, d, \mathit{arecover}(m))]\!]^\tau_\eta\big]
\end{aligned}$$

(where ε_1 is the negligible function due to Theorem 6.6)

$$\begin{aligned}
&\quad \ge 1 - \varepsilon_1(\eta) - \mathbb{P}\big[\tau \leftarrow \mathrm{Coins}_\eta(m);\\
&\qquad\qquad \mathit{bclosure}([\![m]\!]^\tau_\eta, d, \mathit{brecover}([\![m]\!]^\tau_\eta)) \ne [\![\mathit{aclosure}(m, d, \mathit{arecover}(m))]\!]^\tau_\eta\\
&\qquad\qquad \vee [\![\mathit{arecover}(m)]\!]^\tau_\eta \ne \mathit{brecover}([\![m]\!]^\tau_\eta)\big]\\
&\quad \ge 1 - \varepsilon_1(\eta) - \mathbb{P}\big[\tau \leftarrow \mathrm{Coins}_\eta(m);\\
&\qquad\qquad \mathit{bclosure}([\![m]\!]^\tau_\eta, d, [\![\mathit{arecover}(m)]\!]^\tau_\eta) \ne [\![\mathit{aclosure}(m, d, \mathit{arecover}(m))]\!]^\tau_\eta\\
&\qquad\qquad \vee [\![\mathit{arecover}(m)]\!]^\tau_\eta \ne \mathit{brecover}([\![m]\!]^\tau_\eta)\big]\\
&\quad \ge 1 - \varepsilon_1(\eta) - \mathbb{P}\big[\tau \leftarrow \mathrm{Coins}_\eta(m);\\
&\qquad\qquad \mathit{bclosure}([\![m]\!]^\tau_\eta, d, [\![\mathit{arecover}(m)]\!]^\tau_\eta) \ne [\![\mathit{aclosure}(m, d, \mathit{arecover}(m))]\!]^\tau_\eta\big]\\
&\qquad\qquad - \mathbb{P}\big[\tau \leftarrow \mathrm{Coins}_\eta(m); [\![\mathit{arecover}(m)]\!]^\tau_\eta \ne \mathit{brecover}([\![m]\!]^\tau_\eta)\big]\\
&\quad \ge 1 - \varepsilon_1(\eta) - \varepsilon_2(\eta) - \mathbb{P}\big[\tau \leftarrow \mathrm{Coins}_\eta(m);\\
&\qquad\qquad \mathit{bclosure}([\![m]\!]^\tau_\eta, d, [\![\mathit{arecover}(m)]\!]^\tau_\eta) \ne [\![\mathit{aclosure}(m, d, \mathit{arecover}(m))]\!]^\tau_\eta\big]
\end{aligned}$$

(where ε_2 is the negligible function due to Lemma 6.2)

$$\geq 1 - \varepsilon_1(\eta) - \varepsilon_2(\eta) - \varepsilon_3(\eta)\,,$$

where ε_3 is the negligible function due to Lemma 6.4. □

Theorem 6.8 (Completeness) *Let m_1 and m_2 be acyclic messages. Then $[\![m_1]\!] \equiv [\![m_2]\!] \implies m_1 \cong m_2$.*

Proof. Let us assume that $m_1 \ncong m_2$. Now we show that $[\![m_1]\!] \not\equiv [\![m_2]\!]$ by building a distinguisher D.

```
algorithm $D(\mu)$ :
    $d := \max(|m_1|, |m_2|)$
    if $psp(\mu, bclosure(\mu, d)) \approx pattern(m_1)$
        return 1
    else
        return 0
```

Note that for a randomly chosen $\mu \leftarrow [\![m_1]\!]_\eta$ computing $D(\mu)$ takes, with overwhelming probability, a polynomial amount of time since *psp* runs with overwhelming probability in polynomial time and *bclosure* is of polynomial complexity.

Next we show that $\mathrm{Adv}_D(\eta) = |\mathbb{P}[\mu \leftarrow [\![m_1]\!]_\eta; D(\mu) = 1] - \mathbb{P}[\mu \leftarrow [\![m_2]\!]_\eta; D(\mu) = 1]|$ is not negligible. On the one hand

$$\begin{aligned}\mathbb{P}[\mu \leftarrow [\![m_1]\!]_\eta; D(\mu) = 1] &= \mathbb{P}[\mu \leftarrow [\![m_1]\!]_\eta; psp(\mu, bclosure(\mu, d)) \approx pattern(m_1)]\\ &= 1 - \mathbb{P}[\mu \leftarrow [\![m_1]\!]_\eta; psp(\mu, bclosure(\mu, d)) \not\approx pattern(m_1)]\\ &\geq 1 - \varepsilon_1(\eta)\,,\end{aligned}$$

where ε_1 is the negligible function from Lemma 6.7. Note that $pattern(m_1) = pattern(m_1, aclosure(m_1, |m_1|))$. On the other hand

$$\begin{aligned}\mathbb{P}[\mu \leftarrow [\![m_2]\!]_\eta; D(\mu) = 1] &= \mathbb{P}[\mu \leftarrow [\![m_2]\!]_\eta; psp(\mu, bclosure(\mu, d)) \approx pattern(m_1)]\\ &\leq \mathbb{P}[\mu \leftarrow [\![m_2]\!]_\eta; psp(\mu, bclosure(\mu, d)) \not\approx pattern(m_2)]\\ &\leq \varepsilon_2(\eta)\,,\end{aligned}$$

where ε_2 is the negligible function from Lemma 6.7. Therefore, $\mathrm{Adv}_D(\eta) = 1 - \varepsilon_1(\eta) - \varepsilon_2(\eta)$, which is not negligible. □

7 Active adversaries

We now briefly turn our attention to active adversaries. One could view the passive adversaries we have been considering up to now as being given the transcript of a protocol run and trying to deduce information from that. The soundness and completeness results say that the information an adversary could learn in the computational setting is the same as in the symbolic one. Active adversaries, however, can also try to inject messages in the network while the protocol is running. Hence, to obtain a soundness result, every meaningful message that an adversary could send in the computational world, should also be sendable in the symbolic world. Just as IND-CPA is not strong enough for encryption schemes for this to hold — one needs non-malleability [MW04b,Her05,MP05] — oracle hashing is not strong enough for hashes.

We now show an explicit example of an oracle hash scheme where an adversary is capable of creating more messages in the computational world than in the symbolic world.

Let $p = 2q + 1$ be a large (i.e., scaling with η) safe prime. Consider the oracle hash $\mathcal{H}(x) = \langle r^2, r^{2 \cdot f(x)} \mod p\rangle$ from Section 3.3. Assume that f is homomorphic for arguments up to length 2η, i.e., $f(x+y) = f(x)f(y) \mod q$ when $x + y < 2^\eta$. For instance, one could take $f(x) = g^x \mod q$ when $x < 2^\eta$ and $f(x) = h(x)$ otherwise, assuming that q is also a safe prime, g is a generator of the quadratic residues modulo q, and h is a collision resistant one-way function. For simplicity, ignore the tagged representation (see Section 4) and assume that the representation of the concatenation of two nonces is just $\nu'\nu = 2^\eta \cdot \nu' + \nu$. Then

$$\begin{aligned} \mathcal{H}(\nu'\nu) &= \mathcal{H}(2^\eta \cdot \nu' + \nu) \\ &= (r^2, r^{2 \cdot f(2^\eta \cdot \nu' + \nu)} \mod p) && \text{(for some } r\text{)} \\ &= (r^2, r^{2 \cdot f(\nu')^{2^\eta} f(\nu)} \mod p) && \text{(since } r^{2q} = 1 \mod p\text{)} \\ &= (r^2, (r^{2 \cdot f(\nu)})^{f(\nu')^{2^\eta}} \mod p). \end{aligned}$$

Therefore, in the computational world with this particular oracle hash, an attacker receiving $\mathcal{H}(\nu) = (r^2, r^{2 \cdot f(\nu)})$ is capable of producing $\mathcal{H}(\nu'\nu)$ for a nonce ν' of her own choice. In the symbolic world, however, this is impossible since $h^r(n', n)$ is not in the closure of $\{\mathrm{h}^r(n), n'\}$.

Next, we show a very simple one-way authentication protocol that, implemented with a malleable oracle hash function, results in a broken implementation. In this protocol, principal B authenticates to principal A, with whom he shares a symmetric key k_{AB}. The protocol is the following
(1) $A \rightarrow B : n$
(2) $B \rightarrow A : h(k_{AB}, n)$
Consider a homomorphic implementation $\mathcal{H}$ of h, as before. Now suppose that the attacker sees a protocol run: A sends to B a nonce ν, then B replies $\mathcal{H}(\kappa_{AB}, \nu)$. Later, the attacker is able to answer a new challenge ν' by sending $\mathcal{H}(\kappa_{AB}, \nu) \cdot \mathcal{H}(\nu' - \nu)$ to A. This results in a successful impersonation of B by the attacker.

The conclusion is that oracle hashing is not strong enough to give a perfect correspondence between the symbolic world and the computational world when the adversary is active. Just as for encryption schemes, what would be needed is a concept of *non-malleability* [DDN91] for hashes.

8 Conclusions and Future Work

We have proposed an interpretation for formal hashes that is computationally sound and complete in the standard model. Under standard assumptions on hashes (pre-image and collision resistance), the symbolic world does not perfectly match the computational world. However, our results show that it is still possible to achieve this perfect match, for passive adversaries, using Canetti's oracle hashing. While considering active adversaries, we have shown that the security definition for oracle hashing is not strong enough. It would be interesting to extend the notion of non-malleability for hashes to achieve this perfect match also for active adversaries.

Acknowledgements. We are thankful to David Galindo for providing the reference to [Can97a] and insightful comments.

References

[ABHS05] Pedro Adão, Gergei Bana, Jonathan Herzog, and Andre Scedrov. Soundness of formal encryption in the presence of key-cycles. In Sabrina De Capitani di Vimercati, Paul F. Syverson, and Dieter Gollmann, editors, *Proceedings of the 10th European Symposium on Research in Computer Security (ESORICS'05)*, volume 3679 of *Lecture Notes in Computer Science*, pages 374–396. Springer Verlag, 2005.

[ABS05] Pedro Adão, Gergei Bana, and Andre Scedrov. Computational and information-theoretic soundness and completeness of formal encryption. In *Proceedings of the 18th IEEE Computer Security Foundations Workshop, (CSFW'05)*, pages 170–184. IEEE, 2005.

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

[AJ01] Martín Abadi and Jan Jürjens. Formal eavesdropping and its computational interpretation. In Naoki Kobayashi and Benjamin C. Pierce, editors, *Proceedings of the 4th International Symposium on Theoretical Aspects of Computer Software (TACS'01)*, volume 2215 of *Lecture Notes in Computer Science*, pages 82–94. Springer Verlag, 2001.

[AR02] Martín Abadi and Phillip Rogaway. Reconciling two views of cryptography (the computational soundness of formal encryption). *Journal of Cryptology*, 15(2):103–127, 2002.

[AW05] Martín Abadi and Bogdan Warinschi. Security analysis of cryptographically controlled access to XML documents. In *Proceedings of the 24th ACM Symposium on Principles of Database Systems*, pages 108–117. ACM Press, 2005.

[Ban04] Gergei Bana. *Soundness and Completeness of Formal Logics of Symmetric Encryption*. PhD thesis, University of Pennsylvania, 2004.

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

[BDJR97] Mihir Bellare, Anand Desai, Eron Jokipii, and Philip Rogaway. A concrete security treatment of symmetric encryption. In *Proceedings of the 38th Annual Symposium on Foundations of Computer Science (FOCS'97)*, pages 394–405. IEEE, 1997.

[BLMW07] Emmanuel Bresson, Yassine Lakhnech, Laurent Mazaré, and Bogdan Warinschi. A generalization of DDH with applications to protocol analysis and computational soundness. In A. J. Menezes, editor, *Proceedings the IACR International Conference: Advances in Cryptology (CRYPTO'07)*, Lecture Notes in Computer Science. Springer Verlag, 2007.

[BPW06] Michael Backes, Birgit Pfitzmann, and Michael Waidner. Limits of the BRSIM/UC soundness of dolev-yao models with hashes. In Dieter Gollmann, Jan Meier, and Andrei Sabelfeld, editors, *Proceedings of the 11th European Symposium on Research in Computer Security (ESORICS'06)*, volume 4189 of *Lecture Notes in Computer Science*, pages 404–423. Springer Verlag, 2006.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM Conference on Computer and Communication Security (CCS'93)*, pages 62–73. ACM Press, 1993.

[Can97a] Ran Canetti. Towards realizing random oracles: Hash functions that hide all partial information. In Burt Kaliski, editor, *Advances in Cryptology (CRYPTO'97)*, volume 1294 of *Lecture Notes in Computer Science*, pages 455–469. Springer Verlag, 1997.

[Can97b] Ran Canetti. Towards realizing random oracles: Hash functions that hide all partial information. Cryptology ePrint Archive, Report 1997/007 (http://eprint.iacr.org/1997/007), 1997.

[CGH04] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. *Journal of the ACM*, 51(4):557–594, 2004.

[CKKW06] Véronique Cortier, Steve Kremer, Ralf Küsters, and Bogdan Warinschi. Computationally sound symbolic secrecy in the presence of hash functions. In Naveen Garg and S. Arun-Kumar, editors, *Proceedings of the 26th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'06)*, volume 4337 of *Lecture Notes in Computer Science*, pages 176–187. Springer Verlag, 2006.

[CMR98] Ran Canetti, Danielle Micciancio, and Omer Reingold. Perfectly one-way probabilistic hash functions. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing (STOC'98)*, pages 131–140. ACM Press, 1998.

[DDN91] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing (STOC'91)*, pages 542–552. ACM Press, 1991.

[DY83] Danny Dolev and Andrew C. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(2):198–208, 1983.

[GB01] Shafi Goldwasser and Mihir Bellare. *Lecture Notes on Cryptography*. 2001. http://www-cse.ucsd.edu/~mihir/papers/gb.html.

[GGvR08] David Galindo, Flavio D. Garcia, and Peter van Rossum. Computational soundness of non-malleable commitments. In Liqun Chen, Yi Mu, and Willy Susilo, editors, *4th Information Security Practice and Experience Conference (ISPEC 2008)*, volume 4266 of *Lecture Notes in Computer Science*, pages 361–376. Springer Verlag, 2008.

[GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28:270–299, 1984.

[Gol01] Oded Goldreich. *Foundations of Cryptography*, volume 1. Cambridge University Press, 2001.

[GvR06] Flavio D. Garcia and Peter van Rossum. Sound computational interpretation of symbolic hashes in the standard model. In Hiroshi Yoshiura, Kouichi Sakurai, Kai Rannenberg, Yuko Murayama, and Shinichi Kawamura, editors, *Advances in Information and Computer Security. International Workshop on Security (IWSEC'06)*, volume 4266 of *Lecture Notes in Computer Science*, pages 33–47. Springer Verlag, 2006.

[Her05] Jonathan Herzog. A computational interpretation of Dolev-Yao adversaries. *Theoretical Computer Science*, 340(1):57–81, 2005.

[JLM07] Romain Janvier, Yassine Lakhnech, and Laurent Mazaré. Computational soundness of symbolic analysis for protocols using hash functions. In *Proceedings of the First Workshop in*

Information and Computer Security (ICS'06), volume 186 of *Electronic Notes in Theoretical Computer Science*, pages 121–139, 2007.

[KM07] Steve Kremer and Laurent Mazaré. Adaptive soundness of static equivalence. In Joachim Biskup, editor, *Proceedings of the 12th European Symposium on Research in Computer Security (ESORICS'07)*, Lecture Notes in Computer Science. Springer Verlag, 2007. To appear.

[Maz07] Laurent Mazaré. Computationally sound analysis of protocols using bilinear pairings. In Riccardo Focardi, editor, *Preliminary Proceedings of the 7th International Workshop on Issues in the Theory of Security (WITS'07)*, pages 6–21, 2007.

[MP05] Daniele Micciancio and Saurabh Panjwani. Adaptive security of symbolic encryption. In Joe Kilian, editor, *Proceedings of the 2nd Theory of Cryptography Conference (TCC'05)*, volume 3378 of *Lecture Notes in Computer Science*, pages 169–187. Springer Verlag, 2005.

[MW04a] Daniele Micciancio and Bogdan Warinschi. Completeness theorems of the Abadi-Rogaway logic of encrypted expressions. *Journal of Computer Security*, 12(1):99–129, 2004.

[MW04b] Daniele Micciancio and Bogdan Warinschi. Soundness of formal encryption in the presence of active adversaries. In Moni Naor, editor, *Proceedings of the 1st Theory of Cryptography Conference (TCC'04)*, volume 2951 of *Lecture Notes in Computer Science*, pages 133–151. Springer Verlag, 2004.

[PW00] Birgit Pfitzmann and Michael Waidner. Composition and integrity preservation of secure reactive systems. In *Proceedings of the 7th ACM Conference on Computer and Communication Security (CCS'00)*, pages 245–254, 2000.

[RS04] Phillip Rogaway and Thomas Shrimpton. Cryptographic hash-function basics: Definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance. In Bimal Roy and Willi Meier, editors, *Proceedings of the 11th Fast Software Encryption (FSE'04)*, volume 3017 of *Lecture Notes in Computer Science*, pages 371–388. Springer Verlag, 2004.

[Yao82] Andrew C. Yao. Theory and applications of trapdoor functions. In *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science (FOCS'82)*, pages 80–91, 1982.