Weak Key Authenticity and the Computational Completeness of Formal Encryption

Omer Horvitz[1] and Virgil Gligor[2]

[1] Department of Computer Science,
University of Maryland, College Park MD 20742
horvitz@cs.umd.edu
[2] Department of Electrical and Computer Engineering,
University of Maryland, College Park MD 20742
gligor@eng.umd.edu

Abstract. A significant effort has recently been made to rigorously relate the formal treatment of cryptography with the computational one. A first substantial step in this direction was taken by Abadi and Rogaway [AR02]. Considering a formal language that treats symmetric encryption, [AR02] show that an associated formal semantics is sound with respect to an associated computational semantics, under a particular, sufficient, condition on the computational encryption scheme. In this paper, we give a necessary and sufficient condition for completeness, tightly characterizing this aspect of the exposition. Our condition involves the ability to distinguish a ciphertext and the key it was encrypted with, from a ciphertext and a random key. It is shown to be strictly weaker than a previously suggested condition for completeness (confusion-freedom of Micciancio and Warinschi [MW02]), and should be of independent interest.

Keywords. Cryptography, Encryption, Authentication, Formal Reasoning, Completeness, Weak Key Authenticity.

1 Introduction

Modern cryptography has been investigated from both a formal and a computational perspective. Of the former, a typical treatment features a formal language, in which statements, representing cryptographic entities and operations, can be made. Their security properties are usually stated outside the language, captured in operations that manipulate the formal statements, or expressed with additional formal constructs. Of the latter, a typical treatment uses algorithms on strings of bits to model cryptographic operations. Security properties are defined in terms of probability and computational complexity of successful attacks.

Recently, an effort has been made to relate the two approaches, traditionally considered separately and mostly by different communities. A successful attempt holds the promise of bringing the strengths of one treatment to the other. From one direction, it is expected to quantify and highlight implicit assumptions of formal semantics. In addition, it should confirm and increase the relevance of formal proofs to concrete computational instantiations. From the other direction, the establishment of such connections may allow the application of high level formal reasoning mechanisms to the computational domain.

A first step in this direction was taken by Abadi and Rogaway [AR02]. Focusing on symmetric encryption, their work is based on a formal language that includes constructs to represent bits, keys and an encryption operation. Two semantics are defined for the language. In the first, an expression in the language is associated with a syntactic counterpart, the *pattern*, which mirrors the expression up to parts that should look unintelligible to a viewer (informally, parts that are encrypted with keys that are not recoverable from the expression). Expressions are said to be equivalent in this setting if their patterns are equal (up to key renaming). This constitutes a formal semantics. In the second definition, an expression is associated with an ensemble of distributions on strings, obtained by instantiating its encryption construct with a concrete computational encryption scheme with different security parameters. Two expressions are said to be indistinguishable in this setting if their associated ensembles are computationally indistinguishable. This constitutes a computational semantics. Under this framework, [AR02] give a soundness result: they show that under specific, sufficient, conditions on the computational encryption scheme, equivalence of expressions in the formal semantics implies their indistinguishability in the computational semantics.

Our Results. In this paper, we tightly characterize the completeness aspect of this exposition. We identify a *necessary* and *sufficient* condition on the computational encryption scheme under which indistinguishability in the computational setting implies equivalence in the formal one. For any two expressions, our condition involves the admittance of an efficient test that distinguishes a ciphertext and the key it was encrypted with, from a ciphertext and some random key, with a noticeable (i.e., non-negligible) probability, when the plaintexts are drawn from the ensembles associated with those expressions. An encryption scheme that satisfies this requirement is said to *admit weak key-authenticity tests for expressions*. The result is obtained using a new proof technique, featuring a fixpoint characterization of formal equivalence.

In the literature [MW02,AJ01], the notion of *confusion-freedom* was previously proposed as sufficient for completeness. Informally, a confusion-free encryption scheme is one in which the decryption of a ciphertext with a wrong key fails with almost certainty. The above-mentioned work suggests the use of a full-fledged *authenticated encryption scheme* [BN00,KY00] to achieve this notion. We compare confusion-freedom with a strengthened version of the notion of the admittance of weak key-authenticity tests for expressions, that involves the admittance of a single, all-purpose, weak key-authenticity test, defined purely in computational terms. Such test is referred to as a *weak key-authenticity test*. We show that the requirement that an encryption scheme admits a weak key-authenticity test is *strictly weaker* than the requirement that it be confusion-free (and certainly weaker than the requirement that it be an authenticated encryption scheme). To that effect, we present a simple encryption scheme that admits a weak key-authenticity test but is not confusion-free. The scheme thus matches our completeness criterion, but not that of [MW02]. Furthermore, it meets the soundness criterion of [AR02].

The notion of the weak key authenticity should be of independent interest. As a primitive, it relates to the absence of a weak version of key-anonymity [BB01] and which-key revealing [AR02] properties. It would be interesting to investigate ways of meeting it, other than the ones presented here, and explore its practical uses.

The paper proceeds as follows. In section 2, we revisit the formal treatment of symmetric encryption of [AR02], and give a fixpoint characterization of the "reachable parts" of expressions. In section 3, we discuss the computational treatment of symmetric encryption, and revisit the computational semantics for expressions of [AR02]. In section 4, we give our main completeness result for schemes that admit weak key-authenticity tests for expressions. In section 5, we present the strengthened version of the test and compare it with other cryptographic notions; in particular, we show that the admittance of a weak key-authenticity test is a weaker property of an encryption scheme than it being confusion-free. The proof demonstrates a method of achieving the admittance of a weak key-authenticity test.

2 Formal Treatment of Symmetric Encryption, Formal Semantics for Expressions

In this section, we revisit the formal treatment of symmetric encryption of [AR02]. That treatment consists of a formal language and a formal semantics. Our goal is to recast the definitions of [AR02] in terms that pertain closely to the tree structure of expressions in the language. In addition, we provide an alternative, fixpoint characterization of the "reachable parts" of expressions, that plays an important role in the proof of our completeness result.

2.1 A Formal Language for Symmetric Encryption

Let **Bits** be the set $\{0, 1\}$. Let **Keys** be a fixed, non-empty set of symbols, disjoint from **Bits**. The elements of **Bits** and **Keys** are referred to as bits and keys, respectively. Following the work of [AR02], we let our formal language, denoted by **Exp**, be a set of *expressions*, defined inductively as follows:

1. bits and keys are *expressions*. They are referred to as *atomic expressions*, or simply *atoms*.
2. (a) If M and N are *expressions*, then so is (M, N). We say that (M, N) is *directly derived* from M and N; it is *non-atomic*.
   (b) If M is an *expression* and K is a key, then $\{M\}_K$ is an *expression*. We say that $\{M\}_K$ is is *directly derived* from M; it is *non-atomic* too.

Parts 2a and 2b of the above definition are called *derivation rules*. Informally, (M, N) represents the pairing of expressions M and N; $\{M\}_K$ represents the encryption of expression M with key K.

Expressions are strings of symbols. The *length* of an expression E is the number of symbols it is comprised of (count '$\}_K$' as a single symbol), and is denoted by $|E|$. We use $E_1 = E_2$ to denote that the expressions E_1, E_2 are identical as strings of symbols.

It is important to note that every non-atomic expression can be associated with a unique rule and a unique set of expressions from which it is directly derived. Expressions in **Exp** are consequently said to be *uniquely readable*. The converse holds too. Formally, two non-atomic expressions $E_1, E_2 \in$ **Exp** are identical as strings of symbols iff either

- $E_1 = (M_1, N_1)$, $E_2 = (M_2, N_2)$ and $M_1 = M_2$, $N_1 = N_2$; or
- $E_1 = \{M_1\}_{K_1}$, $E_2 = \{M_2\}_{K_2}$ and $M_1 = M_2$, $K_1 = K_2$.

For a proof, see the full version of this paper [HG03].

The structure of an expression can be represented naturally in the form of a tree. A *derivation tree* T_E for an expression E is defined inductively as follows:

1. If E is atomic, then T_E consists of a single node, the *root*, labelled by E.
2. If E is non-atomic, then T_E consists of a single node, the *root*, labelled by E, and an ordered list of *trees* for the expressions from which E is directly derived; the sets of nodes of these trees are disjoint, and none contains the root of T_E. If $E = (M, N)$, we say that T_M and T_N are the *left* and *right subtrees* of T_E, respectively. The roots of T_M and T_N are said to be the *left* and *right children* of the root of T_E, respectively. Similarly, if $E = \{M\}_K$ then T_M is said to be the *subtree* of T_E; the root of T_M is said to be the *child* of the root of T_E.

Informally, the notion of a derivation tree resembles that of the standard parse tree; the two relate in that a node in a derivation tree is labelled with the *yield* of the corresponding node in the parse tree. We let $|T_E|$ denote the cardinality of the set of nodes of T_E.

We mention two properties of expressions and their derivation trees that are relevant to our treatment. First, two expressions are identical as strings of symbols iff their respective derivation trees are identical; to see this, apply the unique readability property of expressions and its converse inductively to the structure of the derivation trees. Second, if $|E| = n$, then T_E consists of at most n nodes; this can be shown by induction on the length of an expression.

2.2 Formal Semantics for Expressions

In defining a formal semantics for **Exp**, we seek to capture a notion of privacy, intuitively associated with the encryption operation. In particular, we would like to express our understanding that parts of expressions, representing encryptions with keys that are not recoverable from the text, are unintelligible (or unreachable) to a viewer. We would also like to capture our understanding that expressions, differing only in their unintelligible parts, "look the same" to a viewer. Just as in [AR02], we do so by mapping each expression to a syntactic counterpart, the *pattern*, which mirrors only its reachable parts. We then define equivalence of expressions in terms of their respective patterns. We state our definitions in terms of functions on derivation trees of expressions and patterns, rather than in the form of procedures on expressions and patterns, as is done in [AR02].

Let T_E be the derivation tree of an expression E, let V be its node set and $r \in V$ its root. A set $U \subseteq V$ is said to *contain the reachable nodes* of T_E if:

1. $r \in U$.
2. For all $u \in U$,
   (a) if u is labelled with an expression of the form (M, N), then both the children of u in T_E (labelled M and N) are in U.
   (b) if u is labelled with an expression of the form $\{M\}_K$, and there exists a $u' \in U$ labelled K, then the child of u in T_E (labelled M) is in U.

For $E \in \mathbf{Exp}$ of length n, T_E consists of at most n nodes, of which there are at most 2^n subsets. It follows that the number of sets containing the set of reachable nodes of T_E is finite. Let R be the intersection of all those sets. It is easy to show that R itself contains the set of reachable nodes of T_E; it is minimal in the sense that it is contained in all such sets. We call R the *set of reachable nodes* of T_E. Informally, reachable nodes correspond to parts of an expression that should be intelligible to a viewer.

Let T_E be a derivation tree with a root r and a set of reachable nodes R. The graph induced by T_E on R must be a tree rooted at r, and not a forest (otherwise, let R' be the set of nodes in the connected component that contains r; R' is a set that contains the set of reachable nodes in T_E, contradicting the minimality of R). We call this tree the *tree of reachable nodes*, and use T_E^R to denote it.

The definition of a pattern extends that of an expression with the addition of an atomic symbol, $\Box$. Informally, $\Box$ will appear in parts of a pattern that correspond to unintelligible parts of the associated expression.

Let **Pat** be the set of *patterns*, defined inductively as follows:

1. bits, keys and the symbol $\Box$ are *(atomic) patterns*.
2. (a) If M and N are *patterns*, then (M, N) is a *(non-atomic) pattern*.
   (b) If M is a *pattern* and K is a key, then $\{M\}_K$ is a *(non-atomic) pattern*.

As with expressions, we associate a pattern P with a derivation tree T_P. Two patterns are identical as strings of symbols iff their respective derivation trees are identical.

To map expressions to patterns via their respective derivation trees, we will need an appropriate notion of tree isomorphism. Let T_1, T_2 be finite, rooted, ordered trees with node sets V_1, V_2 and roots $r_1 \in V_1$, $r_2 \in V_2$, respectively. T_1, T_2 are said to be *isomorphic as rooted, ordered trees* if there exists a bijection $\varphi : V_1 \rightarrow V_2$ such that:

1. $\varphi(r_1) = r_2$.
2. For all $v \in V_1$, $(u_1, \ldots, u_k)$ are the children of v in T_1 *iff* $(\varphi(u_1), \ldots, \varphi(u_k))$ are the children of $\varphi(v)$ in T_2.

φ is said to be an *isomorphism* of T_1, T_2 *as rooted, ordered trees.*

Let T_E be the derivation tree of an expression E, V_E its node set, $R \subseteq V_E$ its set of reachable nodes. Let T_P be the derivation tree of a pattern P, V_P its node set. We say that *expression E has a pattern P* if there exists a $\varphi : R \to V_P$ such that:

1. φ is an isomorphism of T_E^R, T_P as rooted, ordered trees.
2. For all $v \in R$,
   (a) if v is labelled with a bit, then $\varphi(v)$ is labelled with an identical bit.
   (b) if v is labelled with a key, then $\varphi(v)$ is labelled with an identical key.
   (c) if v is labelled (M, N), then $\varphi(v)$ is labelled (M', N').
   (d) if v is labelled $\{M\}_K$ and there exists a $u \in R$ labelled K, then $\varphi(v)$ is labelled $\{M'\}_K$.
   (e) if v is labelled $\{M\}_K$ and there does not exist a $u \in R$ labelled K, then $\varphi(v)$ is labelled $\Box$.

The corresponding definition of [AR02] amounts to a walk of T_E^R and T_P that enforces the above constraints.

We note that the pattern P associated with each expression E is unique. To see this, notice that the uniqueness of T_E implies a unique set of reachable nodes R, which implies a unique T_E^R, which is mapped to a unique T_P, which, in turn, guarantees a unique P. The converse is not true, however; every pattern has infinitely many expressions that are mapped to it.

We proceed with the notion of expression equivalence. Informally, we require that the derivation trees of patterns corresponding to equivalent expressions be isomorphic up to key renaming. For $i \in \{1, 2\}$, let P_i be the pattern of expression E_i, with a derivation tree T_{P_i} over V_{P_i}. We say that E_1 is *equivalent* to E_2, and write $E_1 \cong E_2$, iff there exists a $\varphi : V_{P_1} \to V_{P_2}$ and a permutation σ on **Keys** such that:

1. φ is an isomorphism of T_{P_1}, T_{P_2} as rooted, ordered trees.
2. For all $v \in V_{P_1}$,
   (a) if v is labelled with a bit, then $\varphi(v)$ is labelled with an identical bit.
   (b) if v is labelled K, then $\varphi(v)$ is labelled with $\sigma(K)$.
   (c) if v is labelled (M, N), then $\varphi(v)$ is labelled (M', N').
   (d) if v is labelled $\{M\}_K$, then $\varphi(v)$ is labelled $\{M'\}_{\sigma(K)}$.
   (e) if v is labelled $\Box$, then $\varphi(v)$ is labelled $\Box$.

Composing the above definitions, we obtain the following property of the equivalence relation.

Theorem 2.1. *For $i \in \{1, 2\}$, let E_i be an expression with a derivation tree T_{E_i}, a set of reachable nodes R_{E_i} and an induced tree of reachable nodes $T_{E_i}^R$. Then $E_1 \cong E_2$ iff there exist a $\varphi : R_{E_1} \to R_{E_2}$ and a permutation σ on **Keys** such that:*

1. *φ is an isomorphism of $T^R_{E_1}$, $T^R_{E_2}$ as rooted, ordered trees.*
2. *For all $v \in R_{E_1}$,*
   (a) *if v is labelled with a bit, then $\varphi(v)$ is labelled with an identical bit.*
   (b) *if v is labelled K, then $\varphi(v)$ is labelled with $\sigma(K)$.*
   (c) *if v is labelled (M, N), then $\varphi(v)$ is labelled (M', N').*
   (d) *if v is labelled $\{M\}_K$ and there exists a $u \in R_{E_1}$ labelled K, then $\varphi(v)$ is labelled $\{M'\}_{\sigma(K)}$ and $\varphi(u)$ is labelled $\sigma(K)$.*
   (e) *if v is labelled $\{M\}_K$ and there does not exist a $u \in R_{E_1}$ labelled K, then $\varphi(v)$ is labelled $\{M'\}_{K'}$ and there does not exist a $u' \in R_{E_2}$ labelled K'.*

The proof, mostly technical, appears in the full version of this paper [HG03].

We conclude with a brief discussion of some ramifications of the formal semantics we have seen in this section. We observe that under the above definitions, the encryption operator:

– *"Preserves privacy"*, as seen in the equivalence $\{0\}_K \cong \{1\}_K$. Informally, a ciphertext conceals the underlying plaintext.
– *"Conceals plaintext repetitions"*, as seen in the equivalence $(\{0\}_K, \{0\}_K) \cong (\{0\}_K, \{1\}_K)$. Informally, an adversary, given two ciphertexts, cannot tell whether their underlying plaintexts are identical or not.
– *"Conceals key repetitions"*, as seen in the equivalence $(\{0\}_{K_1}, \{1\}_{K_1}) \cong (\{0\}_{K_7}, \{1\}_{K_8})$. Informally, an adversary, given two ciphertexts, cannot tell whether they were generated with the same encryption key or not.
– *"Conceals plaintext length"*, as seen in the equivalence $\{0\}_K \cong \{(0, (1, 0))\}_K$. Informally, the ciphertext conceals the length of the underlying plaintext.

The definitions of semantics can be modified to accommodate relaxations of the above properties. For example, the semantics can be made sensitive to different plaintext lengths, by introducing an atomic pattern symbol $\Box_n$ for each size n and modifying the definition of equivalence appropriately. We stress that the results of [AR02] and ours can be modified to tolerate such changes.

2.3 A Fixpoint Characterization of the Set of Reachable Nodes

The set of reachable nodes was defined in the previous section in set-intersection terms. Here, we give an alternative characterization that plays an important role in the proof of our completeness result. We show that for an expression E, the set of reachable nodes of T_E is the least fixpoint of an associated operator, O_E. In addition, we show that this fixpoint can be achieved by an iterative application of the operator, no more than a polynomial (in the size of E) number of times. The reader is referred to the full version of this paper [HG03] for a full account.

Let S be a finite set, and let 2^S be the set of all subsets of S. A set $A \subseteq 2^S$ is said to be a *fixpoint* of $O : 2^S \to 2^S$ if $O(A) = A$; A is said to be the *least fixpoint* of O, and is denoted $\mathrm{lfp}(O)$, if A is a fixpoint of O and for all fixpoints B of O, $A \subseteq B$. The *powers* of O are defined as follows:

$$
\begin{aligned}
O^0 &= \emptyset \\
O^i &= O(O^{i-1}) \qquad \text{for all } i \in \mathbb{N}^+
\end{aligned}
$$

Consider an expression E with a derivation tree T_E over a set of nodes V_E with a root $r_E \in V_E$. Let $O_E : 2^{V_E} \to 2^{V_E}$ be defined as follows:

$$O_E(A) = \left\{ u \in V_E \;\middle|\; \begin{array}{l} \text{either:} \\ \text{(a) } u = r_E\text{; or} \\ \text{(b) } \exists v \in A \text{ labelled } (M, N) \text{ with a left child } u \text{ in } T_E \text{ (labelled } M\text{); or} \\ \text{(c) } \exists v \in A \text{ labelled } (M, N) \text{ with a right child } u \text{ in } T_E \text{ (labelled } N\text{); or} \\ \text{(d) } \exists v \in A \text{ labelled } \{M\}_K \text{ with a child } u \text{ in } T_E \text{ (labelled } M\text{)} \\ \quad\;\; \text{and } \exists w \in A \text{ labelled } K. \end{array} \right\}$$

We prove the following:

Theorem 2.2 (A Fixpoint Characterization of the Set of Reachable Nodes [HG03]). *Let E be an expression of length n, T_E its derivation tree over V_E, $R_E \subseteq V_E$ the set of reachable nodes. Then there exists an $i \in \mathbb{N}$, $0 \le i \le n$, such that for all $j \ge i$, $O_E^j = \mathrm{lfp}(O_E) = R_E$.*

3 Computational Treatment of Symmetric Encryption, Computational Semantics for Expressions

In this section, we describe a computational treatment of symmetric encryption: we define an encryption scheme, discuss a relevant notion of security, and review methods of achieving such a notion under standard assumptions. The discussion is similar to the one in [AR02], and may be skipped without significant damage. We then use a computational encryption scheme to define a semantics for the language of expressions of subsection 2.1, recasting the corresponding definition of [AR02] in terms of the derivation trees of expressions.

3.1 Computational Treatment of Symmetric Encryption

Let $\{0,1\}^*$denote the set of all finite binary strings and let $|x|$ denote the length of $x \in \{0,1\}^*$.

An *encryption scheme* $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ with a security parameter $\eta \in \mathbb{N}$ consists of three polynomial-time algorithms, as follows:

- $\mathcal{K}$, the *key generation algorithm*, is a probabilistic algorithm that takes a security parameter $\eta \in \mathbb{N}$ (provided in unary—denoted by 1^η) and returns a *key* $k \in \{0,1\}^*$. We write $k \xleftarrow{R} \mathcal{K}(1^\eta)$, thinking of k as being drawn from the probability distribution induced by $\mathcal{K}(1^\eta)$ on $\{0,1\}^*$. When used as a set, we let $\mathcal{K}(1^\eta)$ denote the support of that distribution.
- $\mathcal{E}$, the *encryption algorithm*, is a probabilistic algorithm that takes a key $k \in \mathcal{K}(1^\eta)$ for some $\eta \in \mathbb{N}$ and a *plaintext* $x \in \{0,1\}^*$ and returns a *ciphertext* $c \in \{0,1\}^* \cup \{\bot\}$. As before, we write $c \xleftarrow{R} \mathcal{E}_k(x)$, thinking of c as being drawn from the probability distribution induced by $\mathcal{E}_k(x)$ on $\{0,1\}^*$. When used as a set, we let $\mathcal{E}_k(x)$ denote the support of that distribution.

It is common for encryption schemes to restrict the set of strings they are willing to encrypt; having the encryption algorithm return $\perp$ is intended to capture such restrictions. We make two requirements. First, we insist that for a given $\eta \in \mathbb{N}$, a plaintext $x \in \{0,1\}^*$ is either *restricted* or not, that is, for all $k \in \mathcal{K}(1^\eta)$, $\mathcal{E}_k(x) = \{\perp\}$ or for all $k \in \mathcal{K}(1^\eta)$, $\mathcal{E}_k(x) \not\ni \perp$. We use $\mathrm{Plain}_{\Pi[\eta]}$ to denote the set of unrestricted plaintexts, for any $\eta \in \mathbb{N}$. Second, we require that for any $\eta \in \mathbb{N}$, if $x \in \{0,1\}^*$ is not restricted, then all $x' \in \{0,1\}^*$ of the same length are unrestricted.
In addition, we insist that the length of a ciphertext $c \in \mathcal{E}_k(x)$ depend only on η and $|x|$ when $k \in \mathcal{K}(1^\eta)$, for any x and η.

– $\mathcal{D}$, the *decryption algorithm*, is a deterministic algorithm that takes a key $k \in \mathcal{K}(1^\eta)$ for some $\eta \in \mathbb{N}$ and a ciphertext $c \in \{0,1\}^*$ and returns some $x \in \{0,1\}^* \cup \{\perp\}$. We write $x \leftarrow \mathcal{D}_k(c)$.
Having the decryption algorithm output $\perp$ is intended to reflect a rejection of the given ciphertext.

We require that Π be *correct*; that is, for all $\eta \in \mathbb{N}$, for all $k \in \mathcal{K}(1^\eta)$ and for all $x \in \mathrm{Plain}_{\Pi[\eta]}$, $\mathcal{D}_k(\mathcal{E}_k(x)) = x$.

A Notion of Security. We consider a variation of the standard notion of indistinguishability under chosen-plaintext attacks (*IND-CPA* security, for short) of [GM84,BD97]. Informally, the strengthened version "conceals key repetitions" and "conceals message lengths", as discussed in subsection 2.2. This is necessary for a soundness result (see [AR02] for additional motivation).

Recall that a function $\epsilon : \mathbb{N} \to \mathbb{R}$ is *negligible* if for every constant $c \in \mathbb{N}$ there exists an η_c such that for all $\eta > \eta_c$, $\epsilon(\eta) \le \eta^{-c}$.

Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme, $\eta \in \mathbb{N}$ a security parameter and A an adversary with access to two oracles (denoted $A^{(\cdot),(\cdot)}$). Define:

$$\begin{aligned}\mathrm{Adv}^0_{\Pi[\eta]}(A) = &\Pr[k, k' \xleftarrow{R} \mathcal{K}(1^\eta) : A^{\mathcal{E}_k(\cdot),\mathcal{E}_{k'}(\cdot)}(1^\eta) = 1] \\ &- \Pr[k \xleftarrow{R} \mathcal{K}(1^\eta) : A^{\mathcal{E}_k(0),\mathcal{E}_k(0)}(1^\eta) = 1],\end{aligned}$$

where $\mathcal{E}_k(\cdot)$ is an oracle that returns $c \xleftarrow{R} \mathcal{E}_k(m)$ on input m, and $\mathcal{E}_k(0)$ is an oracle that returns $c \xleftarrow{R} \mathcal{E}_k(0)$ on input m. We say that Π is *Type-0/IND-CPA, Key-repetition Concealing, Length Concealing secure* [AR02] if for every probabilistic, polynomial-time adversary A, $\mathrm{Adv}^0_{\Pi[\eta]}(A)$ is negligible (as a function of η).

Pseudorandom Function Families and Achieving Type-0 Security. Given a set S, let $x \xleftarrow{R} S$ denote the sampling of x from S endowed with a uniform distribution.

Let $\eta \in \mathbb{N}$, l, L be polynomials, $\mathrm{Func}^{l(\eta)\to L(\eta)}$ the set of all functions from $\{0,1\}^{l(n)}$ to $\{0,1\}^{L(n)}$, $F \subseteq \mathrm{Func}^{l(\eta)\to L(\eta)}$ a family of functions indexed by $\{0,1\}^\eta$, and A an adversary with access to an oracle (denoted $A^{(\cdot)}$). Define:

$$\begin{aligned}\mathrm{Adv}^{\mathrm{prf}}_{F[\eta]}(A) = &\Pr[k \xleftarrow{R} \{0,1\}^\eta : A^{F_k(\cdot)}(1^\eta) = 1] \\ &- \Pr[f \xleftarrow{R} \mathrm{Func}^{l(\eta)\to L(\eta)} : A^{f(\cdot)}(1^\eta) = 1],\end{aligned}$$

where $F_k(\cdot)$ is an oracle that returns $F_k(x)$ on input x, and $f(\cdot)$ is an oracle that returns $f(x)$ on input x. We say that F is *pseudorandom [GG86]* if for every probabilistic, polynomial time adversary A, $\mathrm{Adv}^{\mathrm{prf}}_{F[\eta]}(A)$ is negligible (as a function of η).

Pseudorandom function families are commonly used in computational cryptography as building blocks for encryption schemes, as in the CBC and CTR modes. In [BD97], it is shown that these modes are IND-CPA secure when using an underlying pseudorandom family of functions. In [AR02], the authors describe how these results extend to achieve Type-0 security. See the mentioned references for more details.

3.2 Computational Semantics for Expressions

In this section, we define a computational semantics for the language of expressions of section 2.1. We first associate an expression with an ensemble of distributions over $\{0,1\}^*$, resulting each from the "instantiation" of the expression with a concrete computational encryption scheme with a particular security parameter. We then define expression indistinguishability in terms of the indistinguishability of associated ensembles.

Let E be an expression, T_E its derivation tree over V_E, Keys_E the set of key symbols appearing in E (that is, atomic keys and keys from derivations of the form $\{\cdot\}_K$). Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme with a security parameter $\eta \in \mathbb{N}$. For $x_1, \ldots, x_k \in \{0,1\}^*$ and a tag t from some finite, fixed set of tags, let $\langle x_1, \ldots, x_k, t\rangle$ denote an (arbitrary, fixed, unambiguous, polynomial-time) encoding of $x_1, \ldots, x_k, t$ as a string over $\{0,1\}^*$. Define the following procedure:

$\mathbf{Sample}_{\boldsymbol{\Pi}[\eta]}(\mathbf{E})$

1. For each $K \in \mathrm{Keys}_E$, let $\tau(K) \xleftarrow{R} \mathcal{K}(1^\eta)$.
2. Assign a *sampling label* to each $v \in V_E$, inductively, as follows:
   (a) If v is labelled with a bit b, let its *sampling label* be $\langle b, \text{"bit"}\rangle$.
   (b) If v is labelled with a key K, let its *sampling label* be $\langle \tau(K), \text{"key"}\rangle$.
   (c) If v ia labelled (M, N), its left child in T_E has a sampling label m and its right child in T_E has a sampling label n, then let the *sampling label* of v be $\langle m, n, \text{"pair"}\rangle$ if $m, n \neq \perp$, $\perp$ otherwise.
   (d) If v is labelled $\{M\}_K$ and its child in T_E has a sampling label m, then let the *sampling label* of v be $\langle \mathcal{E}_{\tau(K)}(m), \text{"ciphertext"}\rangle$ if $m \neq \perp$, $\perp$ otherwise.
3. Output the sampling label of the root of T_E.

Let $[\![E]\!]_{\Pi(\eta)}$ denote the probability distribution induced by $\mathrm{Sample}_{\Pi[\eta]}(E)$ on $\{0,1\}^* \cup \perp$; let $[\![E]\!]_{\Pi}$ denote the ensemble $\left\{[\![E]\!]_{\Pi(\eta)}\right\}_{\eta\in\mathbb{N}}$.

We write $x \xleftarrow{R} D$ to indicate that x is sampled from a distribution D. To make our forthcoming definitions robust, we require that Π is such that for every expression E, there exists an $\eta_E \in \mathbb{N}$ such that for all $\eta \geq \eta_E$ and $e \xleftarrow{R} [\![E]\!]_{\Pi(\eta)}$, $e \in \mathrm{Plain}_{\Pi[\eta]}$.

For $i \in \{1,2\}$, let $D_i = \{D_i(\eta)\}_{\eta \in \mathbb{N}}$ be probability distribution ensembles, A an algorithm. Define:

$$\mathrm{Adv}^{\mathrm{ind}}_{D_1(\eta),D_2(\eta)}(A) = \Pr[x \xleftarrow{R} D_1(\eta) : A(1^\eta, x) = 1] \\ - \Pr[x \xleftarrow{R} D_2(\eta) : A(1^\eta, x) = 1].$$

We say that D_1, D_2 are *indistinguishable*, and write $D_1 \approx D_2$, if for every probabilistic, polynomial time algorithm A, $\mathrm{Adv}^{\mathrm{ind}}_{D_1(\eta),D_2(\eta)}(A)$ is negligible (as a function of η).

Let E_1, E_2 be expressions. We say that E_1, E_2 are *indistinguishable*, and write $E_1 \overset{\Pi}{\approx} E_2$, iff $[\![E_1]\!]_\Pi \approx [\![E_2]\!]_\Pi$.

4 Weak Key-Authenticity Tests for Expressions, Semantic Completeness

The soundness result of Abadi and Rogaway states that for acyclic expressions[3] E_1, E_2 and a Type-0 encryption scheme Π, $E_1 \cong E_2$ implies $E_1 \overset{\Pi}{\approx} E_2$. Here, we give a *necessary* and *sufficient* condition for completeness, tightly characterizing this aspect of the exposition. For any two acyclic expressions, the condition involves the admittance of an efficient test that distinguishes a ciphertext and the key it was encrypted with, from a ciphertext and some random key, with a noticeable probability, when the plaintexts are drawn from the ensembles associated with those expressions. Formally:

Definition 4.1 (Weak Key-Authenticity Test for Expressions). Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme with a security parameter $\eta \in \mathbb{N}$, let E_1, E_2 be acyclic expressions, A an algorithm. Define:

$$\mathrm{Adv}^{\text{wka-exp}}_{\Pi[\eta],E_1,E_2}(A) \\ = \Pr[e \xleftarrow{R} [\![E_1]\!]_{\Pi(\eta)}; k \xleftarrow{R} \mathcal{K}(1^\eta); c \xleftarrow{R} \mathcal{E}_k(e) : A(1^\eta, c, k) = 1] \\ - \Pr[e \xleftarrow{R} [\![E_2]\!]_{\Pi(\eta)}; k, k' \xleftarrow{R} \mathcal{K}(1^\eta); c \xleftarrow{R} \mathcal{E}_k(e) : A(1^\eta, c, k') = 1].$$

We say that Π *admits a weak key-authenticity test for* E_1, E_2 (*WKA-EXP-*(E_1, E_2) *test*, for short), if there exists a probabilistic, polynomial-time algorithm A such that $\mathrm{Adv}^{\text{wka-exp}}_{\Pi[\eta],E_1,E_2}(A)$ is non-negligible (as a function of η).

We say that Π *admits weak key-authenticity tests for expressions* (*WKA-EXP tests*, for short), if for all acyclic expressions E_1 and E_2, Π admits a weak key-authenticity test for E_1, E_2.

Our main result is the following:

[3] Expressions that do not contain "encryption cycles"; see [AR02] for a formal definition.

Theorem 4.2 (The admittance of WKA-EXP tests is necessary and sufficient for completeness). *Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme. Then for all acyclic expressions E_1 and E_2, $E_1 \overset{\Pi}{\approx} E_2$ implies $E_1 \cong E_2$* iff *Π admits weak key-authenticity tests for expressions.*

We begin by proving the necessity part. Let E_1, E_2 be two acyclic expressions. Consider the expressions $M_1 = (\{E_1\}_K, K)$, $M_2 = (\{E_2\}_K, K')$ (without loss of generality, assume K does not occur in E_1, E_2). $M_1 \not\cong M_2$, so by the completeness assumption $M_1 \overset{\Pi}{\not\approx} M_2$. Let B be such that $\mathrm{Adv}^{\mathrm{ind}}_{[\![M_1]\!]_{\Pi(\eta)}, [\![M_2]\!]_{\Pi(\eta)}}(B)$ is non-negligible. We use B to construct a WKA-EXP-(E_1, E_2) test A for Π. Define: $A(1^\eta, c, k) \overset{\mathrm{def}}{=} B(1^\eta, \langle\langle c, \text{“ciphertext”}\rangle, \langle k, \text{“key”}\rangle, \text{“pair”}\rangle)$. Now:

$$
\begin{aligned}
&\mathrm{Adv}^{\text{wka-exp}}_{\Pi[\eta], E_1, E_2}(A) \\
&\quad = \Pr[e \xleftarrow{R} [\![E_1]\!]_{\Pi(\eta)}; k \xleftarrow{R} \mathcal{K}(1^\eta); c \xleftarrow{R} \mathcal{E}_k(e) : A(1^\eta, c, k) = 1] \\
&\qquad - \Pr[e \xleftarrow{R} [\![E_2]\!]_{\Pi(\eta)}; k, k' \xleftarrow{R} \mathcal{K}(1^\eta); c \xleftarrow{R} \mathcal{E}_k(e) : A(1^\eta, c, k') = 1] \\
&\quad = \Pr\left[\begin{array}{l} e \xleftarrow{R} [\![E_1]\!]_{\Pi(\eta)}; \\ k \xleftarrow{R} \mathcal{K}(1^\eta); \\ c \xleftarrow{R} \mathcal{E}_k(e) \end{array} : B\left(1^\eta, \begin{array}{c} \langle\langle c, \text{“ciphertext”}\rangle, \\ \langle k, \text{“key”}\rangle, \text{“pair”}\rangle \end{array}\right) = 1\right] \\
&\qquad - \Pr\left[\begin{array}{l} e \xleftarrow{R} [\![E_2]\!]_{\Pi(\eta)}; \\ k, k' \xleftarrow{R} \mathcal{K}(1^\eta); \\ c \xleftarrow{R} \mathcal{E}_k(e) \end{array} : B\left(1^\eta, \begin{array}{c} \langle\langle c, \text{“ciphertext”}\rangle, \\ \langle k', \text{“key”}\rangle, \text{“pair”}\rangle \end{array}\right) = 1\right] \\
&\quad = \Pr[e \xleftarrow{R} [\![(\{E_1\}_K, K)]\!]_{\Pi(\eta)} : B(1^\eta, e) = 1] \\
&\qquad - \Pr[e \xleftarrow{R} [\![(\{E_2\}_K, K')]\!]_{\Pi(\eta)} : B(1^\eta, e) = 1] \\
&\quad = \Pr[e \xleftarrow{R} [\![M_1]\!]_{\Pi(\eta)} : B(1^\eta, e) = 1] - \Pr[e \xleftarrow{R} [\![M_2]\!]_{\Pi(\eta)} : B(1^\eta, e) = 1] \\
&\quad = \mathrm{Adv}^{\mathrm{ind}}_{[\![M_1]\!]_{\Pi(\eta)}, [\![M_2]\!]_{\Pi(\eta)}}(B),
\end{aligned}
$$

where the second equality is due to the definition of A, and the third due to the definition of $\mathrm{Sample}_{\Pi[\eta]}$. It follows that A is a weak key-authenticity test for E_1, E_2, as required. This completes the necessity part of the proof.

Next, we sketch the sufficiency part; a complete proof appears in [HG03]. Assume $E_1 \not\cong E_2$. To show that $[\![E_1]\!]_{\Pi(\eta)} \not\approx [\![E_2]\!]_{\Pi(\eta)}$, we consider an algorithm that simultaneously parses its input e, a sample from either $[\![E_1]\!]_{\Pi(\eta)}$ or $[\![E_2]\!]_{\Pi(\eta)}$, and expressions E_1, E_2, attempting to construct φ, σ that bear witness to the equivalence of the expressions. By the assumption, this attempt is bound to fail. We show that upon failure, the algorithm has enough parsed information to predict the origin of the sample with a non-negligible probability of success. In some cases, the prediction depends on an application of a weak key-authenticity test for (particular, fixed) expressions to the amassed information.

Specifically, the algorithm computes the powers of the operator $O_{E_1,E_2,e}$, defined in Fig. 1 (where $S = V_{E_1} \times V_{E_2} \times \{0,1\}^*$), as long as they satisfy the predicate TEST, also of Fig. 1. Let $i \in \mathbb{N}$. Let $V^i_{E_1} = \left\{v_1 \,\middle|\, (v_1, \cdot, \cdot) \in O^i_{E_1,E_2,e}\right\}$, $V^i_{E_2} = \left\{v_2 \,\middle|\, (\cdot, v_2, \cdot) \in O^i_{E_1,E_2,e}\right\}$. Let $j \in \{1, 2\}$. Let $T^{V^i_{E_j}}_{E_j}$ denote the subtree

$$O_{E_1,E_2,e}(A) =$$

$$\left\{ \begin{array}{c|l} & \text{either:} \\ & \text{(a) } u_1 = r_{E_1},\ u_2 = r_{E_2},\ y = e\text{; or} \\ & \text{(b) } \exists (v_1, v_2, x) \in A \text{ such that:} \\ & \quad v_1 \text{ is labelled } (M, N) \text{ and has a left child } u_1 \text{ in } T_{E_1}, \\ & \quad v_2 \text{ is labelled } (M', N') \text{ and has a left child } u_2 \text{ in } T_{E_2} \\ & \quad \text{and } x \text{ is of the form } \langle y, z, \text{“pair”} \rangle\text{; or} \\ & \text{(c) } \exists (v_1, v_2, x) \in A \text{ such that:} \\ & \quad v_1 \text{ is labelled } (M, N) \text{ and has a right child } u_1 \text{ in } T_{E_1}, \\ (u_1, u_2, y) & \quad v_2 \text{ is labelled } (M', N') \text{ and has a right child } u_2 \text{ in } T_{E_2} \\ \in S & \quad \text{and } x \text{ is of the form } \langle y, z, \text{“pair”} \rangle\text{; or} \\ & \text{(d) } \exists (v_1, v_2, x) \in A \text{ and } \exists (w_1, w_2, z) \in A \text{ such that:} \\ & \quad v_1 \text{ is labelled } \{M\}_K \text{ and has a child } u_1 \text{ in } T_{E_1}, \\ & \quad v_2 \text{ is labelled } \{M'\}_{K'} \text{ and has a child } u_2 \text{ in } T_{E_2}, \\ & \quad x \text{ is of the form } \langle c, \text{“ciphertext”} \rangle, \\ & \quad w_1 \text{ is labelled } K, \\ & \quad w_2 \text{ is labelled } K', \\ & \quad z \text{ is of the form } \langle k, \text{“key”} \rangle \\ & \quad \text{and } y = \mathcal{D}_k(c). \end{array} \right\}$$

$$\text{TEST}(A) =$$

$$\begin{cases} \textit{true} & \text{if for all } (v_1, v_2, x) \in A\text{, either:} \\ & \text{(a) } v_1 \text{ is labelled with } b \in \textbf{Bits} \text{ and } v_2 \text{ is labelled } b\text{; or} \\ & \text{(b) } v_1 \text{ is labelled } K,\ v_2 \text{ is labelled } K' \text{ and for all } (u_1, u_2, y) \in A, \\ & \quad u_1 \text{ is labelled } K \text{ iff } u_2 \text{ is labelled } K'\text{; or} \\ & \text{(c) } v_1 \text{ is labelled } (M, N) \text{ and } v_2 \text{ is labelled } (M', N')\text{; or} \\ & \text{(d) } v_1 \text{ is labelled } \{M\}_K,\ v_2 \text{ is labelled } \{M'\}_{K'} \text{ and for all} \\ & \quad (u_1, u_2, y) \in A,\ u_1 \text{ is labelled } K \text{ iff } u_2 \text{ is labelled } K'. \\ \textit{false} & \text{otherwise.} \end{cases}$$

Fig. 1. Definitions of $O_{E_1,E_2,e} : 2^S \to 2^S$, TEST : $2^S \to \{\text{true}, \text{false}\}$.

induced by $V^i_{E_j}$ on T_{E_j}. Let O_{E_j} be the operator from the fixpoint characterizations of the set of reachable nodes of T_{E_j} (see Theorem 2.2). We show that as long as TEST$(O^i_{E_1,E_2,e})$ holds,

– $V^{i+1}_{E_1} = O^{i+1}_{E_1}$ and $V^{i+1}_{E_2} = O^{i+1}_{E_2}$;
– there exist φ, σ consistent with the requirements of Theorem 2.1 when restricted to $T^{V^i_{E_1}}_{E_1}$, $T^{V^i_{E_2}}_{E_2}$, $V^i_{E_1}$, and $V^i_{E_2}$ (instead of $T^R_{E_1}$, $T^R_{E_2}$, R_{E_1}, and R_{E_2}).

If TEST does not fail by the $\max(|E_1|, |E_2|)$'s power of $O_{E_1,E_2,e}$, V_{E_1}, V_{E_2} achieve the sets of reachable nodes of T_{E_1}, T_{E_2}, respectively, by the first point above, and so $E_1 \cong E_2$ by the second point, contradicting our assumption. We conclude that TEST must fail on some lower power of $O_{E_1,E_2,e}$; let $i^* \in \mathbb{N}$ be the lowest such power.

We use $O^{i^*}_{E_1,E_2,e}$ to make a prediction, based on the reason TEST fails. Here, we illustrate a case that calls for the use of a weak key-authenticity test for expressions. Assume TEST fails because there exist $(v_1, v_2, x), (u_1, u_2, y) \in O^{i^*}_{E_1,E_2,e}$ such that v_1 is labelled $\{M\}_K$, u_1 is labelled K, v_2 is labelled $\{M'\}_{K'}$, and u_2 is labelled K''. An inductive argument on the powers of our operator shows that x, y are the sampling labels of either v_1, u_1, respectively, or v_2, u_2, respectively, depending on the origin of e. Let $x = \langle c, \text{“ciphertext”}\rangle, y = \langle k, \text{“key”}\rangle$. In the first case, c is an encryption of a sample from $[\![M]\!]_{\Pi(\eta)}$ with the key k; in the second case, c is an encryption of a sample from $[\![M']\!]_{\Pi(\eta)}$ with some key, and k is a random key. The WKA-EXP-(M,M') test on c and k distinguishes these cases with a noticeable probability of success.

Finally, we show that the procedure is efficient. This completes the sketch of the sufficiency part of the proof.

5 How the Notion of Weak Key-Authenticity Relates to Other Cryptographic Notions

In this section, we strengthen the notion of the admittance of weak key authenticity tests for expressions. We consider the admittance of a *single*, all-purpose test, hereby referred to as the *weak key-authenticity test*, that distinguishes any ciphertext and the key it was encrypted with from any ciphertext and a random key, with a non-negligible probability; the test is defined in terms that are *independent* of the formal language of the preceding sections. We compare the strengthened version with the notions of confusion-freedom and authenticated encryption, previously discussed in the literature in the context of the completeness result [MW02,AJ01]. Specifically, we show that the requirement that an encryption scheme admits a weak key-authenticity test is *strictly weaker* than the requirement that it be confusion-free, as defined in the above references (which, in turn, is enough to show it is strictly weaker than authenticated encryption as well). To that effect, we present an encryption scheme that admits a weak key-authenticity test but *is not* confusion-free. The scheme we present is also Type-0. It therefore satisfies the soundness criteria of [AR02], our completeness criteria, but not the previous completeness criteria of [MW02]. The notions we present and the methods used to achieve the admittance of a weak key-authenticity test should be of independent interest.

Informally, confusion-freedom captures the ability of a decryption algorithm to distinguish a ciphertext and the key it was encrypted with from a ciphertext and a random key with almost full certainty. In contrast, the weak key-authenticity test is required to distinguish the two with merely a noticeable probability. We will separate the notions in a strong sense, pertaining directly to the gap in their required distinguishing certainties (as opposed to pertaining to the placement of the distinguisher—inside or outside the decryption algorithm).

First, we give formal definitions for the notions at hand.

Confusion-freedom is defined as it appears in the completeness result of [MW02]; our proofs can be modified to accommodate the version of [AJ01] too.

Definition 5.1 (Confusion-Freedom). Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme, $\eta \in \mathbb{N}$ a security parameter, and $D[\eta] = \{D_1[\eta], \ldots, D_l[\eta]\}$ a series of finite sets of distributions. For $1 \leq i \leq l$, define:

$$\mathrm{Adv}^{\mathrm{cf}}_{\Pi[\eta],D[\eta],i} = \Pr[k, k' \xleftarrow{R} \mathcal{K}(1^\eta); x \xleftarrow{R} D_i[\eta] : \mathcal{D}_{k'}(\mathcal{E}_k(x)) \neq \perp].$$

We say that Π is *confusion-free* (*CF* for short) if for any $1 \leq i \leq l$, $\mathrm{Adv}^{\mathrm{cf}}_{\Pi[\eta],D[\eta],i}$ is negligible (as a function of η).

Next, we define two auxiliary notions that will enable us to focus on the the above-mentioned gap. These will provide a "middle ground" for comparing the WKA-EXP test with CF.

Definition 5.2 (Strong Key-Authenticity Test, Weak Key-Authenticity Test). Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme, $\eta \in \mathbb{N}$ a security parameter. Let $\mathcal{P}_1, \mathcal{P}_2$ (hereby referred to as *plaintext generators*) be probabilistic algorithms that take a security parameter η (provided in unary), and for sufficiently large η always return a $x \in \mathrm{Plain}_{\Pi[\eta]}$; we write $x \xleftarrow{R} \mathcal{P}_j(1^\eta)$ for $j \in \{1, 2\}$, thinking of x as being drawn from the probability distribution induced by $\mathcal{P}_j(1^\eta)$ on $\{0,1\}^*$. Let A be an algorithm. Define:

$$\begin{aligned}
&\mathrm{Adv}^{\mathrm{tst}}_{\Pi[\eta],\mathcal{P}_1[\eta],\mathcal{P}_2[\eta]}(A) \\
&\quad = \Pr[x \xleftarrow{R} \mathcal{P}_1(1^\eta); k \xleftarrow{R} \mathcal{K}(1^\eta); c \xleftarrow{R} \mathcal{E}_k(x) : A(1^\eta, c, k) = 1] \\
&\quad\quad - \Pr[x \xleftarrow{R} \mathcal{P}_2(1^\eta); k, k' \xleftarrow{R} \mathcal{K}(1^\eta); c \xleftarrow{R} \mathcal{E}_k(x) : A(1^\eta, c, k') = 1],
\end{aligned}$$

where $\mathrm{tst} \in \{\mathrm{ska}, \mathrm{wka}\}$. We say that Π admits a *strong* (resp., *weak*) *key-authenticity test*, *SKA* (resp., *WKA*) for short, if there exists a probabilistic, polynomial-time algorithm A such that for all probabilistic, polynomial-time algorithms $\mathcal{P}_1, \mathcal{P}_2$, $\mathrm{Adv}^{\mathrm{ska}}_{\Pi[\eta],\mathcal{P}_1[\eta],\mathcal{P}_2[\eta]}(A)$ (resp., $\mathrm{Adv}^{\mathrm{wka}}_{\Pi[\eta],\mathcal{P}_1[\eta],\mathcal{P}_2[\eta]}(A)$) is negligibly close to 1 (resp., is non-negligible) as a function of η.

As for the definition of *integrity of plaintext* security (*INT-PTXT* for short), a flavor of authenticated encryption, we refer the reader to [BN00,KY00] and to [MW02].

The following diagram depicts relationships between our notions of interest.

$$\begin{array}{c}\text{INT-}\\ \text{PTXT}\end{array} \longrightarrow \text{CF} \longrightarrow \begin{array}{c}\text{Admittance}\\ \text{of a}\\ \text{SKA test}\end{array} \begin{array}{c}\longrightarrow\\ \not\longleftarrow\end{array} \begin{array}{c}\text{Admittance}\\ \text{of a}\\ \text{WKA test}\end{array} \longrightarrow \begin{array}{c}\text{Admittance}\\ \text{of}\\ \text{WKA-EXP tests}\end{array}$$

In the above, $A \longrightarrow B$ means that an encryption scheme that meets notion A must also meet notion B; we call such a relationship an *implication.* $A \not\longrightarrow B$ means that an encryption scheme that meets notion A does not necessarily meet notion B; we call such a relationship a *separation.*

The implications in the diagram are mostly straightforward (see [HG03]). The rest of the section is devoted to the separation of WKA from SKA. To that end, we show an encryption scheme that admits a WKA test but does not admit an SKA test. We use a standard construction based on a pseudorandom function family, with an added "weak redundancy". To simplify the exposition, we use a single, constant bit as redundancy; refer to the end of the section for a generalization.

Let F be a pseudorandom family of functions with a security parameter $\eta \in \mathbb{N}$, key domain $\{0,1\}^\eta$, domain $\{0,1\}^{l(\eta)}$ and range $\{0,1\}^{L(\eta)}$ (where l, L are polynomials in η); let ϵ be a negligible function such that $\mathrm{Adv}^{\mathrm{prf}}_{F[\eta]}(A) \leq \epsilon(\eta)$ for any probabilistic, polynomial-time algorithm A. We use $x_1 x_2 \cdots x_m$ to denote the individual bits of a string $x \in \{0,1\}^m$. We use $\circ$ to denote the concatenation operator on strings of bits, $\oplus$ to denote the bitwise XOR operator on strings of bits of equal length.

Define an encryption scheme $\Pi^* = (\mathcal{K}^*, \mathcal{E}^*, \mathcal{D}^*)$ with a security parameter $\eta \in \mathbb{N}$ as follows:

$\mathcal{K}^*(1^\eta)$	$\mathcal{E}^*_k(x = x_1 x_2 \cdots x_{L(\eta)-1})$	$\mathcal{D}^*_k(\langle y = y_1 y_2 \cdots y_{L(\eta)}, r\rangle)$
$k \xleftarrow{R} \{0,1\}^\eta$;	$r \xleftarrow{R} \{0,1\}^{l(\eta)}$;	$x' \leftarrow y \oplus F_k(r)$;
Output k.	$y \leftarrow (x \circ 1) \oplus F_k(r)$;	Output $x'_1 x'_2 \cdots x'_{L(\eta)-1}$.
	Output $\langle y, r\rangle$.	

Note that $\mathrm{Plain}_{\Pi^*[\eta]} = \{0,1\}^{L(\eta)-1}$. Also note that $\mathcal{E}^*$ and $\mathcal{D}^*$ can deduce η from k ($\eta = |k|$).

Π^* can easily be shown to be IND-CPA secure based on the pseudorandomness of F. For a proof, see [GG86], or simply think of Π^* as a degenerate version of the randomized CTR mode, and rely on [BD97]. Using the results of [AR02], it can further be shown to be Type-0.

We have that:

Theorem 5.3. *Π^* admits a WKA test.*

To see this, consider an algorithm that takes as input $\langle y, r\rangle$ and k, computes $y \oplus F_k(r)$ and outputs 1 iff the last bit of the outcome is 1. The algorithm is a WKA test for Π^* by a simple reduction to the pseudorandomness of F (see [HG03]). In addition, we have that:

Theorem 5.4. *Π^* does not admit an SKA test.*

Proof. Let A be a probabilistic algorithm that runs in time t, a function of the size of its input. Let $A(a_1, a_2, \ldots; w)$ denote the outcome of running A on inputs $a_1, a_2, \ldots$ and randomness w. Note that the length of w is bounded by t.

Let $\mathcal{U}$ be an algorithm that takes $\eta \in \mathbb{N}$ (in unary) as input and outputs a random, uniformly-selected element of $\{0,1\}^{L(\eta)-1}$. We have:

$$
\begin{aligned}
&\mathrm{Adv}^{\mathrm{ska}}_{\Pi^*[\eta],\mathcal{U}[\eta],\mathcal{U}[\eta]}(A) \\
&= \Pr\left[\begin{array}{l} x \xleftarrow{R} \{0,1\}^{L(\eta)-1}; k \xleftarrow{R} \{0,1\}^\eta; r \xleftarrow{R} \{0,1\}^{l(\eta)}; \\ w \xleftarrow{R} \{0,1\}^{t(\eta)}; y \leftarrow (x \circ 1) \oplus F_k(r) \end{array} : A(1^\eta, \langle y, r\rangle, k; w) = 1\right] \\
&\quad - \Pr\left[\begin{array}{l} x \xleftarrow{R} \{0,1\}^{L(\eta)-1}; k, k' \xleftarrow{R} \{0,1\}^\eta; r \xleftarrow{R} \{0,1\}^{l(\eta)}; \\ w \xleftarrow{R} \{0,1\}^{t(\eta)}; y \leftarrow (x \circ 1) \oplus F_k(r) \end{array} : A(1^\eta, \langle y, r\rangle, k'; w) = 1\right],
\end{aligned}
$$

where t is a polynomial in η.

Let S_1 and $A_1 \subseteq S_1$ denote the sample space and event, respectively, depicted by the first term above. Let S_2 and $A_2 \subseteq S_2$ be defined similarly with respect to the second term.

Let $(x_0, k_0, r_0, w_0) \in A_1$. Note that for any $k \in \{0,1\}^\eta$, if there exists an $x \in \{0,1\}^{L(\eta)-1}$ such that $(x \circ 1) \oplus F_k(r_0) = (x_0 \circ 1) \oplus F_{k_0}(r_0)$, then it must be the case that $(x, k, k_0, r_0, w_0) \in A_2$ (because in this case, A, in the second experiment, runs on the same input and randomness as in the first experiment). This happens when $x \circ 1 = (x_0 \circ 1) \oplus F_{k_0}(r_0) \oplus F_k(r_0)$, which must happen for at least $\left(\frac{1}{2} - \epsilon(\eta)\right) \cdot 2^\eta$ of the keys $k \in \{0,1\}^\eta$; otherwise, an adversary that queries its oracle on r_0, XORs the answer with $(x_0 \circ 1)$ and with $F_{k_0}(r_0)$, and outputs 1 if the last bit of the result is different than 1, 0 otherwise—breaks the pseudorandomness of F.

For a given $(x_0, k_0, r_0, w_0) \in A_1$, we've just described a way of counting at least $\left(\frac{1}{2} - \epsilon(\eta)\right) \cdot 2^\eta$ tuples in A_2. We would like to argue that for a distinct $(x_1, k_1, r_1, w_1) \in A_1$, we would be counting *different* tuples in A_2 by employing the same method. This is clear if $k_1 \neq k_0$ or $r_1 \neq r_0$ or $w_1 \neq w_0$. As for the case that $k_1 = k_0, r_1 = r_0, w_1 = w_0$, we would be double-counting a tuple iff

$$(x_0 \circ 1) \oplus F_{k_0}(r_0) \oplus F_k(r_0) = (x_1 \circ 1) \oplus F_{k_1}(r_1) \oplus F_k(r_1) = (x_1 \circ 1) \oplus F_{k_0}(r_0) \oplus F_k(r_0),$$

which happens iff $x_1 = x_0$.

We conclude that $|A_2| \geq \left(\frac{1}{2} - \epsilon(\eta)\right) \cdot 2^\eta \cdot |A_1|$. We also know that $|S_2| = 2^\eta \cdot |S_1|$. Therefore:

$$\mathrm{Adv}^{\mathrm{ska}}_{\Pi^*[\eta],\mathcal{U}[\eta],\mathcal{U}[\eta]}(A) = \frac{|A_1|}{|S_1|} - \frac{|A_2|}{|S_2|} \leq \left(\frac{1}{2} + \epsilon(\eta)\right) \cdot \frac{|A_1|}{|S_1|} \leq \frac{1}{2} + \epsilon(\eta),$$

which is *not* negligibly close to 1. ∎

Finally, we note that our construction can be easily generalized to one that admits a WKA test with an advantage *as small as desired*, as follows. For any $c \in \mathbb{N}^+$, let Π^*_c be a variation on Π^* that adds the bit 1 with probability $\frac{1}{2} + \frac{1}{2^c}$, 0 with probability $\frac{1}{2} - \frac{1}{2^c}$, as redundancy upon encryption (instead of the fixed 1). Our proofs easily extend to show that Π^*_c admits a WKA test with advantage at least $\frac{1}{2^c} - \epsilon(\eta)$.

Acknowledgements. We thank Jonathan Katz for helpful discussions and comments. This work was supported by the Defense Advanced Research Projects Agency and managed by the U.S. Air Force Research Laboratory under contract F30602-00-2-0510; the views and conclusions contained are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of DARPA, U.S. AFRL, or the U.S. Government.

References

[AJ01] M. Abadi, J. Jurgens. Formal Eavesdropping and its Computational Interpretation. In *Proc. of the Fourth International Symposium on Theoretical Aspects of Computer Software (TACS 2001)*, 2001.

[AR02] M. Abadi, P. Rogaway. Reconciling Two Views of Cryptography (the computational soundness of formal encryption). In *Journal of Cryptology*, vol. 15, no. 2, pp. 103–128. (also in *Proc. of the First IFIP International Conference on Theoretical Computer Science*, LNCS vol. 1872, pp. 3–22, Springer Verlag, Berlin, August 2000.)

[BB01] M. Bellare, A. Boldyreva, A. Desai, D. Pointcheval. Key-Privacy in Public-Key Encryption. In *Advances in Cryptology — ASIACRYPT 2001*, LNCS vol. 2248,pp. 566-582, Springer Verlag, 2001.

[BD97] M. Bellare, A. Desai, E. Jokipii, P. Rogaway. A Concrete Security Treatment of Symmetric Encryption: Analysis of the DES Modes of Operation. In *Proceedings of the 38th Annual Symposium on Foundations of Computer Science (FOCS 97)*, 1997.

[BN00] M. Bellare, C. Namprempre. Authenticated Encryption: Relations Among Notions and Analysis of the Generic Composition Paradigm. In *Advances in Cryptology — ASIACRYPT 2000*, LNCS vol. 1976, pp. 541–545, Springer Verlag, 2000.

[GG86] O. Goldreich, S. Goldwasser, S. Micali. How to Construct Random Functions. In *Journal of the ACM*, vol. 33, no. 4, pp. 792–807, 1986.

[GM84] S. Goldwasser, S. Micali. Probabilistic Encryption. In *Journal of Computer and System Sciences*, 28:270-299, April 1984.

[HG03] O. Horvitz, V. Gligor. Weak Key Authenticity and the Computational Completeness of Formal Encryption. Full version available at `http://www.cs.umd.edu/~horvitz`, `http://www.ee.umd.edu/~gligor`

[KY00] J. Katz, M. Yung. Unforgeable Encryption and Chosen Ciphertext Secure Modes of Operation. In *Proceedings of the 7th International Workshop on Fast Software Encryption (FSE 2000)*, LNCS vol. 1978, pp. 284–299, Springer Verlag, 2000.

[Ll87] J. W. Lloyd. *Foundations of Logic Programming.* Second Edition, Springer-Verlag, 1987, section 1.5.

[LN84] J. L. Lassez, V. L. Nguyen, E. A. Sonenberg. Fixpoint Theorems and Semantics: a Folk Tale. In *Information Processing Letters*, vol. 14, no. 3, 1982, pp. 112–116.

[MW02] D. Micciancio, B. Warinschi. Completeness Theorems for the Abadi-Rogaway Language of Encrypted Expressions. In *Journal of Computer Security* (to appear). Also in *Proceedings of the Workshop on Issues in the Theory of Security*, 2002.

[Ta55] A. Tarski. A Lattice-theoretical Fixpoint Theorem and its Applications. In *Pacific Journal of Mathematics*, vol. 5, pp.285–309, 1955.