Computational Soundness of Coinductive Symbolic Security under Active Attacks

Mohammad Hajiabadi, Bruce M. Kapron

Dept. of Computer Science, University of Victoria, Victoria, CANADA V8W 3P6
{mhaji,bmkapron}@cs.uvic.ca

Abstract. In Eurocrypt 2010, Miccinacio initiated an investigation of cryptographically sound, symbolic security analysis with respect to *coinductive* adversarial knowledge, and showed that under an adversarially *passive* model, certain security criteria may be given a computationally sound symbolic characterization, without the assumption of *key acyclicity*. Left open in his work was the fundamental question of "the viability of extending the coinductive approach to prove computational soundness results in the presence of *active* adversaries." In this paper we make some initial steps toward this goal with respect to an extension of a *trace-based security model* (Micciancio and Warinschi, TCC 2004) including asymmetric and symmetric encryption; in particular we prove that a random *computational trace* can be *soundly abstracted* by a *coinductive symbolic trace* with overwhelming probability, provided that both the underlying encryption schemes provide IND-CCA2 security (plus ciphertext integrity for the symmetric scheme), and that the *diameter* of the underlying *coinductively-hidden subgraph* is constant in every symbolic trace. This result holds *even if* the protocol allows arbitrarily nested applications of symmetric/asymmetric encryption, unrestricted transmission of symmetric keys, and adversaries who *adaptively corrupt users*, along with other forms of active attack.
As part of our proof, we formulate a game-based definition of encryption security allowing *adaptive corruptions of keys* and certain forms of *adaptive key-dependent plaintext attack*, along with other common forms of CCA2 attack. We prove that (with assumptions similar to above) security under this game is implied by IND-CCA2 security. This also characterizes a *provably benign* form of *cyclic encryption* implied by standard security definitions, which may be of independent interest.

Keywords: Computational soundness, adaptive corruptions, coinduction, circular security, trace-based protocol security, active adversaries

1 Introduction

Provable security, since its introduction in the early 1980s, has provided a rigorous foundation for the security analysis of cryptographic schemes.Typically, proving a cryptographic construction meets a given security goal within the provable security framework requires: (1) formally defining the security goal in

terms of what comprises a violation of the goal and what is assumed about the computational power of the adversary, and (2) giving a feasible method which transforms any attack against the construction to an attack against one of its underlying primitives [12, 13]. This methodology provides strong security assurances against resource-bounded attackers, which is a fairly realistic assumption in real-world applications. However, doing computational security analysis, even for small-sized protocols, can be a gruelingly tedious task, and normally a small change in the protocol necessitates a new security proof. On the other hand, *formal* (logic-based) *methods* [24, 15] greatly simplify security analysis using idealized abstractions of cryptographic primitives and limiting adversarial computation, even allowing for automated verification. While formal methods may help designers identify subtle flaws in their schemes, they do not necessarily provide guarantees of computational security. At the very least, a formally verified scheme may be computationally insecure if realized under "insufficiently strong" primitives (e.g. using *malleable* encryption in the case of active attacks). Motivated by the mismatch between these two approaches, a large body of work, starting from [1], attempts to give computational justifications for formal security proofs, in the form of *computational soundness* theorems. Generally speaking, a formal system for security proofs is computationally sound if whenever a scheme is proved secure in the system, it is guaranteed to also be secure in an appropriate computational security framework.

Background. Standard notions of secure encryption [26, 38] ensure privacy of plaintexts chosen independently from the underlying secret key(s). It has long been known that a key encrypted under itself may no longer remain secret, and recent results [20, 2] show that indeed for all $k \geq 1$, k-circular security is not implied by standard security. Moreover, currently known techniques for standard security fall short when trying to prove non-trivial security statements against more *adaptive* adversaries. As an example, assume in the standard multiple-key-based indistinguishability game [9] over keys $ck_1, \ldots, ck_n$, the adversary is additionally allowed to obtain the (nested) encryption of any ck_i under $\{ck_1, \ldots, ck_{i-1}\}$, giving rise to an acyclic *encryption ordering* between keys. One can use a standard hybrid argument to show that security in this setting is no stronger than standard security. However, this simple hybrid argument fails in the case that the (acyclic) encryption ordering is *a priori* unknown and formed adaptively by the adversary. (The naive approach of guessing the underlying ordering also trivially yields an exponential reduction factor.) In contrast, conventional Dolev-Yao style security analysis models adversarial knowledge *inductively* in an *all-or-nothing* fashion (i.e. the adversary either knows a secret piece of data, or it does not have any information about it). As a result, adversarial power is limited, essentially treating uniformly all symbolic ciphertexts whose encryption keys are *underivable* under so-called Dolev-Yao deduction rules. Consequently, Dolev-Yao models typically assume no difference between two symbolic encryptions $\{k\}_k$ and $\{k_1\}_k$. Also, the "adaptive problem" described above seems to not be a challenge within these models. For these reasons, most existing soundness results are restricted in their assumptions, which include excluding *key cycles*

altogether in the case of *passive* adversaries [1, 28], posing certain encryption orderings in the case of *passive-but-adaptive* adversaries [33], and disallowing symmetric encryption in the case of *active* adversaries [34, 6, 19].
As a resolution to the problems created by key cycles, Micciancio [32] proposes a *coinductive* method for modeling symbolic security, and obtains computational soundness in the setting of *message indistinguishability* for passive adversaries, while allowing key cycles and assuming only *semantic security* for the underlying encryption function. Coinductive symbolic security corresponds to a *greatest-fixedpoint*-based definition of adversarial knowledge, as opposed to the *least-fixedpoint*-based definition adopted by conventional inductive methods. From a cryptographic perspective, [32] implicitly characterizes a *provably benign* form of *circular encryption*, in particular the equivalence of standard security to secure encryption under a variant of the multiple-key-based game described above in which the adversary may obtain the (single or nested) encryption of any ck_i under arbitrary keys, provided at least one of them is in $\{ck_1, \ldots, ck_{i-1}\}$, resulting in a (possibly) cyclic encryption ordering. To obtain soundness, [32] shows that for an *a priori* known sequence of exchanged symbolic messages (which is the case in the passive setting), one may order all *coinductively irrecoverable* keys from this sequence as $k_1, \ldots, k_m$, such that each occurrence of k_i is encrypted under at least one of $\{k_1, \ldots, k_{i-1}\}$.

Our Results. In this paper we investigate the question left open in [32]; namely, whether a coinductive approach provides similar soundness guarantees when applied in the setting of *active* adversaries. We consider a symbolic/computational *trace-based execution model* [34], including asymmetric and symmetric encryption. In contrast to previous work, we allow *symmetric keys* to be *freely* included in protocol messages, symmetric and asymmetric encryptions to be arbitrarily *nested*, and adversaries to *adaptively corrupt users*, along with other forms of active attack. We first pose the following central question: to what extent can *any* encryption scheme with standard security withstand stronger types of attack including *adaptive corruptions of keys* and *key-dependent/circular encryption*? To formalize this, consider the following game over symmetric/asymmetric encryption schemes $\mathcal{E}^s = (G^s, E^s, D^s)$, $\mathcal{E}^a = (G^a, E^a, D^a)$, $\{ck_i\}_{1 \leq i \leq n} \leftarrow G^s(1^\eta)$, and $\{(pk_i, sk_i)\}_{1 \leq i \leq n} \leftarrow G^a(1^\eta)$, in which the adversary is allowed to adaptively corrupt keys (symmetric and asymmetric), obtain decryptions of *permissible* ciphertexts, and issue key-dependent encryption queries of the form $E^s(f(ck_1, \ldots, ck_n), ck_j)$ or $E^a(f(ck_1, \ldots, ck_n), pk_j)$, where f is any arbitrary composition of *constant*, *pairing*, *projection* ($P_i(ck_1, \ldots, ck_n) = ck_i$), and *encryption* ($E^a_{pk_i}(\cdot)$, $E^s_{ck_i}(\cdot)$) functions. We remark asymmetric decryption keys may not be used to form key-dependent messages, reflecting our assumption that such keys are not sent as plaintexts in protocol messages[1]. This function family allows one to describe encryption queries symbolically (e.g. $E^s(E^s(ck_1, ck_2), ck_1)$ is denoted $\{\{k_1\}_{k_2}\}_{k_1}$), and hence symbolically keep track of adversarial knowledge. Now we ask: if $\mathcal{E}^a$ and $\mathcal{E}^s$ provide IND-CCA2 security *only*, can we prove,

[1] Relaxing this requirement does not add to the technical difficulty of the proofs. We assumed this requirement as it seems to be the case for most protocols in practice.

at the end of the game, certain keys still maintain computational *secrecy*, in the sense they can securely be used in an encryption-based indistinguishability game[2]? Several negative results [20, 2] show certain key cycles may compromise the secrecy of their component keys, but on the positive side this problem (in a generic sense involving circular encryption) has not been considered much. Motivating the discussion, the results of [32] in the context of the above game (but where only symmetric encryption is used,) imply if all queries are made at once (i.e. *nonadaptively*), then any ck_i, whose symbolic key k_i remains *coinductively irrecoverable* (*irrecoverable* for short), even if used in key cycles, maintains computational secrecy. Along these lines, we call $(\mathcal{E}^a, \mathcal{E}^s)$ *CI secure* if after the *adaptive* execution of the above game all keys whose symbolic keys remain irrecoverable maintain computational secrecy. We also consider *ACI security*, an extension of CI security which adds *ciphertext integrity* and obtain the following

Theorem (informal). If $(\mathcal{E}^a, \mathcal{E}^s)$ is ACI secure, it provides soundness for coinductive traces.

Next we ask if CI security may be based on IND-CCA2 security. Note that the CI attack model is ostensibly much stronger than the CCA2 one, allowing a CI adversary to adaptively corrupt keys and obtain circularly-encrypted ciphertexts. A naive reduction attempt would be to *a priori* guess all keys which remain irrecoverable during the game, together with their underlying encryption ordering, and then use a hybrid argument in the style of [32] to do the reduction. Such an idea clearly yields an infeasible reduction factor. Instead, we prove that if the *diameter* of the *coinductively-hidden subgraph* of the resulting *key graph* is constant, then CI security is implied by IND-CCA2 security. (It will soon be informally described why our reduction is dependent on this parameter.) Here, the key graph is the (random) multigraph G_k which has a node for every key in the game, and an edge $v_i \rightarrow v_j$ if v_i's associated key encrypts v_j's in an encryption query (e.g. the encryption query $\{\{k_1\}_{k_2}\}_{k_1}$ creates one self-loop and one normal edge,) and by "coinductively hidden subgraph" we mean the *induced subgraph* of G_k on *irrecoverable nodes* (nodes whose associated keys remain irrecoverable). We remark that as long as the above condition holds, the adversary may corrupt any number of keys, and create arbitrary key cycles and arbitrarily-long paths in the whole key graph.

Theorem (informal). If $\mathcal{E}^a$ and $\mathcal{E}^s$ are both IND-CCA2 secure, then for every adversary $\mathcal{A}$ where the diameter of the coinductively hidden subgraph of $G_k(\mathcal{A})$ is constant (i.e. independent of the security parameter), $\mathcal{A}$ has a negligible advantage in the CI game for $(\mathcal{E}^a, \mathcal{E}^s)$. Moreover, if $\mathcal{E}^s$ is also INT-CTXT secure, $\mathcal{A}$ has a negligible advantage in the ACI game.

The starting point of our proof is [36]'s positive results on security against adaptive corruptions (in an *authenticated* channel setting), showing that security in a setting over $\mathcal{E}^s$ and $\{ck_i\}_{1 \leq i \leq n} \leftarrow G^s(1^\eta)$, in which $\mathcal{A}$ may adaptively cor-

[2] Our definition of computational secrecy is close to the idea of *key usability*, developed in [23], for defining alternate, composition-amenable security criteria for key-exchange protocols.

rupt keys, and obtain *single encryptions* $E^s(ck_i, ck_j)$, for $1 \leq i, j \leq n$, subject to key *acyclicity*, is obtained via a reduction to the semantic security of $\mathcal{E}^s$, with a factor of $O(n^l)$ where l is the diameter of the resulting key graph. Although the results of [36] seem to extend, by its mere developed techniques, to an authenticated setting with *nested encryptions*, they crucially rely on acyclicity and break down if this latter is relaxed. Allowing *cyclic nested encryptions*, irrecoverable nodes may have self-loops or oppositely-directed edges between themselves (encryption queries $\{\{k_1\}_{k_2}\}_{k_3}$ and $\{k_2\}_{k_1}$ create such edges, while k_1, k_2 remain irrecoverable), and we still need to prove their computational secrecy. Central to our proof is a new notion of *coinductive continuability*, which for every irrecoverable node characterizes a special set of paths ending in that node, satisfying a property which enables a path-based reduction proof in the style of [36]. (Our reduction is based on guessing random coinductiely continuable paths with certain properties, making it depend on the diameter.) Also, allowing both nested encryptions and decryption queries creates a new complication; namely, to simulate a CI adversary $\mathcal{A}^{CI}$ by a CCA2 adversary $\mathcal{A}^{cca}$, nested encryptions may make an $\mathcal{A}^{cca}$'s *challenge ciphertext* a "legitimate" ciphertext for $\mathcal{A}^{CI}$ (e.g., when the ciphertext corresponding to $\{k_1\}_{k_2}$ in $\{\{k_1\}_{k_2}\}_{k_3}$ is created under $\mathcal{A}^{cca}$'s left-or-right oracle and k_3 remains irrecoverable), and if $\mathcal{A}^{CI}$ makes such a decryption query, our simulation fails. A large part of our proof, thus, involves showing $\mathcal{A}^{CI}$ may produce such ciphertexts only with negligible probability. Such a complication does not arise if one only deals with single encryptions, and in fact, the results of [36] immediately extend if decryption queries are also allowed.

Applications. Our reduction result implies for a protocol Π (which may contain symmetric keys and nonces as atomic messages) and a *trace-expressible* security property $\mathcal{P}$ (here, loosely speaking, by a trace we mean a sequence of states created during an execution of a protocol as a result of adversarial/honest-parties' actions. Formal definitions are given in Section 3), if the following two symbolic assertions hold, then the (CCA2, CCA2+CTXT)-based implementation of Π provably achieves $\mathcal{P}$ (in an *insecure* channel setting) with strong security guarantees against adaptive corruptions: (a) No symbolic coinductive adversary may create a trace containing an arbitrarily-long encryption chain (in the sense described above), and (b) Π is coinductively secure; namely, no coinductive symbolic adversary may produce a trace not satisfying the underlying symbolic property. We observe that all protocols in the Clark-Jacob library [21], in which the only primitives used are asymmetric/symmetric encryption, satisfy our soundness restriction (item (a) above), making it applicable to them. A number of these protocols are asymmetric encryption-based, and analyzable under previous soundness theorems (e.g. [34, 6]). Using our techniques, we show that [27] the *Wide-Mouthed Frog authentication protocol*, which is not analyzable under the cryptographic library of [4] due to the classic *commitment problem* prevalent in simulation-based approaches, satisfies our soundness restriction. This advocates for the use of coinduction as a strong tool in yielding provably-sound security proofs, while circumventing issues involved with using induction-based methods.

Why not KDM security? It may be asked why we bother to investigate soundness of coinductive methods, when there are constructions in the standard model for secure encryption under *key-dependent messaging* [14, 16]. We note that security against adaptive corruptions is a necessary requirement for any encryption scheme used in a protocol which is run in an environment with adversarially adaptive corruptions. In such situations, once a key is corrupted, the security of the protocol will depend on the preservation of secrecy for keys which are not trivially corrupted. Even in the idealized *static corruption* model, a key may dynamically be revealed by the exploitation of potential weaknesses of a protocol (e.g., consider a situation where the adversary gets to alter a communicated message by replacing an "honest" key with his own key, making an honest party then encrypt a secret key under the adversarial key.) To the best of our knowledge, there are no provable constructions of KDM-secure encryption in the standard model which also provide security against adaptive corruptions. Backes et al. [5] consider a limited case in which security is defined only in a left-or-right indistinguishability sense, not addressing the above problem. In subsequent work, [3] considers the problem in its full generality as described above, but their construction is in the random-oracle model. Moreover, they do not consider the question of whether generic constructions from KDM-secure encryption schemes exist (in the standard model) which also provide security against adaptive corruptions.

Related Work. Obtaining sound abstract security proofs for protocols involving symmetric encryption has also been considered following the ideal/real simulation paradigms of [17, 37]. [4] shows that *secure realization* of *ideal* symmetric encryption (in the sense of *reactive simulatability*) is possible in their cryptographic library [6] if the *commitment problem* does not occur (i.e. any honest party's key, after it is used for encryption, never becomes "known" to the adversary), and the *used-order property* is satisfied. (i.e. Deployed keys admit an *a priori* encryption ordering.) The authors of [30], by extending the framework of [19] to allow symmetric encryption, show if a key-exchange protocol satisfies their symbolic criteria and if the above conditions hold, the protocol securely realizes a *key-exchange functionality* in the sense of *universal composability*. We comment the commitment problem may intrinsically occur as a direct result of security formalizations; adaptive corruptions, for instance, trivially enable this possibility. Also, the requirement that "a session-key loss in a key-exchange protocol should not affect the secrecy of other session keys" is formalized by allowing the adversary to adaptively learn session keys, leading, possibly, to the commitment problem. Thus the aforementioned frameworks do not consider the above two attack scenarios. We remark the commitment problem was known long before in the setting of *adaptively-secure multiparty computation*, with initial solutions given in [18]. The results of [22] are aimed at indistinguishability-based security properties (e.g., secrecy requirements for key-exchange protocols), by showing that *observational equivalence* between two processes implies computational indistinguishability under standard cryptographic assumptions. Although [22] allows symmetric encryption, it imposes the same restrictions as [4, 30].

A very different approach which in principle supports reasoning about situations which include key-cyclic encryption and adaptive corruption for both symmetric and asymmetric encryption as well as other primitives is the use of what might be called *general-purpose security logics*. Here we include probabilistic process calculi [31, 35], logics which axiomatize computational indistinguishability [29, 8] and first-order logics augmented with axioms characterizing specific security properties [7]. The tradeoff involved in taking a more generic approach is the loss of structure in proofs, potentially undermining some of the benefits of the formal approach.

Basic Notation: For a review of the standard notions of encryption security, we refer to [10, 11]. If D is a probability distribution, then $x \leftarrow D$ denotes choosing an element according to D, and if S is a set, $x \leftarrow S$ denotes choosing an element uniformly at random from S. For a probability distribution D, $sup[D]$ denotes the *support set* of D, and we write $x \in D$ to mean $x \in sup\,[D]$. We call a function *negligible* if it grows more slowly than the inverse of any polynomial function. For ease of notation, we use $negl(\cdot)$ to refer to any negligible function.

2 Preliminaries

A Formal Language for Cryptographic Expressions. *Expressions* are built from four infinite sets of *basic symbols* – *identifiers*, ID, *public-key symbols*, $\mathcal{K}^{pub}$, *private-key symbols* $\mathcal{K}^{priv}$, and *nonces*, $\mathcal{X}$ – using *encryption*, $\{\diamond\}_\diamond$, and *concatenation*, $(\cdot,\cdot)$, operators for building *compound* messages. We further partition $\mathcal{K}^{priv}$ into *asymmetric private keys*, $\mathcal{K}^{privasym}$, and *symmetric private keys* $\mathcal{K}^{privsym}$. We fix a bijective *key-inverse* operation $(.)^{-1} : \mathcal{K}^{pub} \cup \mathcal{K}^{privsym} \rightarrow \mathcal{K}^{priv}$, which induces the *identity* function on subdomain $\mathcal{K}^{privsym}$.

Whenever it is essential to distinguish between the adversary's and honest parties' basic symbols, we add a subscript A or H to basic symbols, and for every set S defined above, we further define $S = S_H \cup S_A$ (e.g. $\mathcal{K}_H^{priv}$ and $\mathcal{K}_A^{priv}$). Moreover, whenever it is necessary to distinguish between symmetric and asymmetric private-key symbols, we add a superscript sym to symmetric ones. (e.g. we have $(k_1^{sym})^{-1} = k_1^{sym}$.) The set of *formal expressions*, Exp, is:

$$
\begin{aligned}
Exp &::= Plain \mid Cipher \mid (Exp, Exp)\\
Plain &::= ID \mid \mathcal{X} \mid \mathcal{K}^{pub} \mid \mathcal{K}^{privsym}\\
Cipher &::= \{Plain\}_{k \in \mathcal{K}^{pub} \cup \mathcal{K}^{privsym}} \mid \{Cipher\}_{k \in \mathcal{K}^{pub} \cup \mathcal{K}^{privsym}}
\end{aligned}
$$

Coinductive Modeling of Adversarial Knowledge. We take a coinductive approach to modeling adversarial attacks. To model *coinductive adversarial knowledge* [32], we define a *key-recovery function*, $\mathcal{F}$, which specifies given $e \in Exp$ and $T \subseteq \mathcal{K}_H^{priv}$, what keys can be deduced by "single-round" applications of Dolev-Yao rules. Defined naturally, $\mathcal{F}_s(T) = s \cap \mathcal{K}_H^{priv}$ for a basic symbol s, $\mathcal{F}_{(e_1,e_2)}(T) = \mathcal{F}_{e_1}(T) \cup \mathcal{F}_{e_2}(T)$, and $\mathcal{F}_{\{e\}_k}(T) = \mathcal{F}_e(T)$ if $k^{-1} \in T \cup \mathcal{K}_A^{priv}$ and $\mathcal{F}_{\{e\}_k}(T) = \emptyset$, otherwise. T is a *fixedpoint* of $\mathcal{F}_e$ if $\mathcal{F}_e(T) = T$, and is the *greatest* (resp. *least*) fixedpoint if T is the greatest (resp. least) solution of $\mathcal{F}_e(X) = X$

(according to $\subseteq$ ordering). Now T is *coinductively* (resp. *inductively*) defined by $\mathcal{F}_e$ if T is the greatest (resp. least) fixedpoint of $\mathcal{F}_e$. It is easy to see that $\mathcal{F}_e$ is a *monotone* function (i.e., $S_1 \subseteq S_2 \Rightarrow \mathcal{F}_e(S_1) \subseteq \mathcal{F}_e(S_2)$.

The Tarski-Knaster Theorem implies for every monotone function $F : \wp(D) \to \wp(D)$, where D is some set and $\wp(D)$ is its *powerset*, the least fixedpoint, $fix(F)$, and greatest fixedpoint, $FIX(F)$, of F exist and are obtained as follows

$$fix(F) = \bigcap_{S:F(S)\subseteq S} S \qquad (1) \qquad\qquad FIX(F) = \bigcup_{S:S\subseteq F(S)} S \qquad (2)$$

Note that if $T \subseteq \mathcal{F}_e(T)$, then $cl(T) \stackrel{\triangle}{=} \cup_{i\geq 1}\mathcal{F}_e^i(T)$ is a fixedpoint, for which $T \subseteq cl(T)$, where $\mathcal{F}_e^i(T)$ denotes i successive applications of $\mathcal{F}_e$ on T. The latter follows from monotonicity of $\mathcal{F}_e$, and the former follows observing that $\mathcal{F}_e^k(T) = \mathcal{F}_e^{k+1}(T)$ for sufficiently large k's. (This is because the number of keys in e is finite.) Thus the following equivalent formulations follow:

$$fix(\mathcal{F}_e) = \bigcap_{\mathcal{F}_e(S)=S} S \;=\; \bigcup_{i\geq 1}\mathcal{F}_e^i(\emptyset) \quad (3) \qquad FIX(\mathcal{F}_e) = \bigcup_{S=\mathcal{F}_e(S)} S \;=\; \bigcap_{i\geq 1}\mathcal{F}_e^i(\mathcal{K}_H^{priv}) \quad (4)$$

We show (4); the proof for (3) follows by a dual argument. The first equality for $FIX(\mathcal{F}_e)$ follows from (2) and the argument presented above. The second equality follows from the following three observations: (a) $\bigcap_{i\geq 1}\mathcal{F}_e^i(\mathcal{K}_H^{priv})$ is a fixedpoint of $\mathcal{F}_e$, (b) if T is a fixedpoint of $\mathcal{F}_e$, then $T = \bigcap_{i\geq 1}\mathcal{F}_e^i(T)$, and (c) by monotonicity, $\bigcap_{i\geq 1}\mathcal{F}_e^i(T) \subseteq \bigcap_{i\geq 1}\mathcal{F}_e^i(\mathcal{K}_H^{priv})$. Now the set of *coinductively recoverable keys* of e is the set coinductively defined by $\mathcal{F}_e$. For example for $e = k^{-1}, \{\{k_1^{sym}\}_{k_2^{sym}}, \{k_2^{sym}\}_{k_1^{sym}}\}_k$, its coinductively recoverable keys are $\{k^{-1}, k_1^{sym}, k_2^{sym}\}$. (As a convention, we omit parentheses in expressions and write e_1, e_2 for (e_1, e_2).) See [32] for more examples.

We define the *coinductive closure set* of $e \in Exp$, denoted $closure_c(e)$, to be the *smallest* set satisfying: (i) $closure_c(e)$ contains e, $FIX(\mathcal{F}_e)$, ID, $\mathcal{K}^{pub}$, and all the adversary's basic symbols, (ii) if $(e_1, e_2) \in closure_c(e)$ then $e_1, e_2 \in closure_c(e)$, (iii) if e' and e'' are both in $closure_c(e)$, so is (e', e''), (iv) if $\{m\}_k \in closure_c(e)$ and $k^{-1} \in closure_c(e)$ then $m \in closure_c(e)$, and (v) if $m \in closure_c(e)$ and $k \in closure_c(e)$ then $\{m\}_k \in closure_c(e)$. Although the above definition is a hybrid of inductive and coinductive definitions, an equivalent, (fully) coinductive definition is also possible; however, we adopt the above one as it is more natural. Now e_1 is *coinductively recoverable* from e if $e_1 \in closure_c(e)$. Note, if $e_1 \in closure_c(e)$ but $k^{sym} \notin closure_c(e)$, Rule (v) does not allow us to deduce $\{e_1\}_{k^{sym}} \in closure_c(e)$. This models the idealized symbolic assumption that if the adversary does not know an honest party's symmetric key, he cannot produce a ciphertext which decrypts to a meaningful plaintext under that key. To support this assumption in our computational model, we will assume the symmetric encryption scheme provides *ciphertext integrity*.

We say e' is a *subexp* of (or *occurs* in) e, denoted $e' \sqsubseteq e$, if $e = e'$, or $e = (e_1, e_2)$ and $e' \sqsubseteq e_1$ or $e' \sqsubseteq e_2$, or $e = \{e_1\}_k$ and $e' \sqsubseteq e_1$. We say k_1 encrypts k_2^{-1} in e, denoted $k_1 \to^e k_2^{-1}$, if for some $\{e_1\}_{k_1}$ which occurs in e, $k_2^{-1} \sqsubseteq e_1$. An

expression is *key cyclic* if it contains a *key cycle*, that is a sequence $k_0, k_1, \ldots k_{i-1}$ such that $k_j \rightarrow k^{-1}_{(j+1 \mod i)}$ for all $j \geq 0$, and is called *key acyclic* if it is not key cyclic. It is known the inductive and coinductive definitions coincide for key-acyclic expressions[32]. The converse of this, however, does not hold true; it is possible some keys occur in certain key cycles but remain coinductively irrecoverable (e.g. consider $\{\{k_1^{-1}\}_{k_1}\}_{k_2}$). In fact, we will prove it is exactly such keys that remain "secure" under concrete implementations.

Computational Interpretation of Cryptographic Expressions. Under a pair of symmetric/asymmetric schemes $\mathcal{E}_p = (\mathcal{E}_{sym}, \mathcal{E}_{asy})$ with parameters (η_{sym}, η_{asy}), an invertible *pairing function*, and a mapping $\tau(\eta_{sym}, \eta_{asy}, \circ)$, which gives a *concrete value* to every basic symbol, every $e \in Exp$ induces a natural probability distribution, denoted $[\![e]\!]_\tau^{\mathcal{E}_p}$, which we call the *computational image of e with respect to $\mathcal{E}_p$ and τ*. If $E \in [\![e]\!]_\tau^{\mathcal{E}_p}$ and $e_1 \sqsubseteq e$, given τ, one may define the underlying value of e_1 in E in a natural way.

3 Symbolic and Computational Trace-Based Protocol Security

We will now introduce a protocol specification language and consider an extension of the model given in [34] for analyzing security protocols in the presence of active adversaries. For simplicity, we consider two-party protocols, and assume that each protocol runs in a constant number of rounds, and admits a symbolic specification. Under these assumptions, a *protocol* can be described as a sequence $\Pi = (M_1^I, M_1^R, M_2^I, M_2^R, \ldots M_r^I, M_r^R)$ of *messages* being sent alternately between two parties: *initiator* and *responder*. (Here having the responder send the last message is arbitrary.) We assume that each party has an associated *long-lived public key* which the other party may use to encrypt messages, and whose matching private key is never sent as a plaintext. The parties, however, may generate fresh symmetric keys, send them (encrypted) to each other, and later on use exchanged keys to encrypt future messages. Messages that we use to specify protocols are built upon four disjoint sets $Ids = \{I, R\}$, $nonces = \{X_1, X_2, \ldots\}$, $pubkeys = \{K_I, K_R\}$, and $symkeys = \{K_1^{sym}, K_2^{sym}, \ldots\}$, using encryption and concatenation for building compound messages, where K_I and K_R denote the parties' respective public keys. We further require protocols be *computationally executable*; in particular, a party should be able to fully decrypt (all encrypted parts of) a message she receives. (Our results seem to easily extend by relaxing this restriction, allowing, e.g., *ciphertext forwarding*, which allows a party to forward a message without decrypting it.) To summarize our assumptions, we call Π *valid* if: (1) for all $1 \leq i \leq r$ and $x \in \{I, R\}$; K_I^{-1} and K_R^{-1} do not occur in M_i^x, and (2) for all $1 \leq i \leq r$, $x \in \{I, R\}$, and $y = \{I, R\} - \{x\}$; if M_i^x has a subexp $\{M\}_K$, then K is *inductively* recoverable from $(K_y, M_1^x, \ldots, M_i^x)$. (We will use a coinductive approach for modeling adversarial attacks, and this condition is solely meant to specify our class of protocols. In particular, since we require parties be able to fully decrypt their received messages, and their roles be computationally executable, such a condition seems necessary.)

So far we have only described the "syntax" of protocols; this should not be confused with the formal execution semantics to be presented below. Treating Π as a tuple of messages, we denote its ith message by Π_i. We denote the set of protocol *users* (*participants*) by $U = \{u_1, u_2, ..., u_n\}$, where any two of whom may initiate an instance of the protocol together, in a manner controlled by an adversary. The adversary is not himself a protocol user, but may dynamically subvert users during the execution. We model adversarial power as an *oracle* with which he is *adaptively* interacting, by making the following types of query:

- $corrupt(i)$: Corrupts user u_i. In response, the long-lived secret key of u_i (and all other u_i's internal information) is given to the adversary.
- $new\text{-}session(i, j)$: Causes u_i and u_j to start a new session, with u_i as the initiator. The oracle assigns a unique *number*, sn, to their session and gives sn to the adversary plus the first message that u_i sends to u_j in this session.
- $send(sn, m_1, I)$: Causes the oracle to send message m_1 to the initiator of session sn and give m_2, the message that the user produces in response, to the adversary. Here m_2 may be a valid message, an *error message* $\perp$ (indicating m_1 was not of the right format), or a *flag message* $*$ indicating that the user has received her last message, finishing her session.
- $send(sn, m_1, R)$: Similar to above, but m_1 is sent to the responder of sn.

We now give formal and computational semantics for protocols. In the formal setting, we denote the long-lived public key of u_i by k_{u_i}, and for each session sn that u_i is a user of, we denote u_i's generated symmetric keys and nonces in sn, respectively, by $\mathbf{K}^{sym}_{i,sn} = \{k^{sym}_{i,sn,j} \mid j \in \mathbb{N}\} \subseteq \mathcal{K}^{privsym}_H$, $\mathbf{X}_{i,sn} = \{x_{i,sn,j} \mid j \in \mathbb{N}\} \subseteq \mathcal{X}_H$. The adversary may use his own basic symbols to build new messages; we denote the adversary's symmetric keys and nonces, respectively, by $\mathbf{K}^{sym}_A = \{k^{sym}_{A,j} \mid j \in \mathbb{N}\} \subseteq \mathcal{K}^{privsym}_A$, $\mathbf{X}_A = \{x_{A,j} \mid j \in \mathbb{N}\} \subseteq \mathcal{X}_A$. We let Exp_{basic} be the union of all $\mathbf{X}_A, \mathbf{K}^{sym}_A$, $\mathbf{K}^{sym}_{i,sn}$'s, $\mathbf{X}_{i,sn}$'s.

The adversary initially knows only his own basic symbols and parties' IDs and public keys. If he corrupts u_i, he receives $k^{-1}_{u_i}$ as well as $\mathbf{K}^{sym}_{i,sn} \cup \mathbf{X}_{i,sn}$, for every sn that u_i has engaged in. A protocol *state* is characterized by the following components:

$$f : \{I, R\} \times BS(\Pi) \times SN \to Exp_{basic} \cup \{\perp\} \quad l : \{I, R\} \times SN \to \Pi_i \cup \{\surd\}$$
$$h : \{I, R\} \times SN \to U \qquad corr\text{-}users \subseteq \{u_1, \ldots, u_n\}$$

Here SN denotes the set of all session numbers, and, recall that, U is the set of all protocol users. Function f represents the symbolic values that the initiator and responder of each session of the protocol give to basic symbols in that session, and $\perp$ means that the party does not yet know the value of the corresponding basic message. Function l denotes the index of the next message in the protocol that the initiator and responder of each session expect to receive, and $\surd$ indicates that the party has finished her respective session. Finally function h indicates what protocol users take the roles of "initiator" and "responder" in each session.

We denote the initial state of the system by FS_0, where $corr\text{-}users = \emptyset$, and l, f, h map all their inputs to null values. An execution of a *formal adversary*,

$\mathcal{A}_F$, can be described as a sequence of queries $E(\mathcal{A}_F) = (q_1, q_2, \dots)$, with corresponding replies $(r_1, r_2, \dots)$. We then call $\mathcal{A}_F$ *coinductively legitimate* if $m \in closure_c(r_1, r_2, \dots, r_{i-1})$ for all i such that $q_i = send(sn, m, \{I, R\})$. Under $\mathcal{A}_F$'s execution, we denote the induced *formal trace* by $\mathcal{FT}(\mathcal{A}_F) = (FS_0, FS_1 \dots)$, where state FS_i is obtained from FS_{i-1} as a result of query q_i.

Under the computational execution, elements of $BS(\Pi) \subseteq Ids \cup nonces \cup pubkeys \cup symkeys$ are replaced with random bitstrings, sampled w.r.t. a pair $\mathcal{E}_p = (\mathcal{E}_{asy}, \mathcal{E}_{sym})$, with w.l.o.g. a shared security parameter 1^η, and the coins tossed by both protocol users and the adversary during the protocol execution. Each (initially honest) u_i, before engaging in the protocol execution, samples her long-lived key pair, $(pk_i, sk_i) \leftarrow Gen_{asy}(\eta)$, and for each session sn that u_i participates in, u_i uses a (polynomially-long) uniformly-selected random string $\mathcal{R}_{i,sn}$ to sample her nonces and symmetric keys in that session, where symmetric keys are sampled according to Gen_{sym}, and nonces chosen uniformly at random from a fixed *nonce space*, $NS = \{0,1\}^{poly(\eta)}$. The adversary, using random string $\mathcal{R}_A$, may choose his nonces and symmetric keys (to, e.g., replace those of corrupted parties, inject in messages on the network, etc.) in any arbitrary efficient manner; he may also initially corrupt a party and choose her public/private key pair in any arbitrary manner (not necessarily following Gen_{asy}).

Letting $C_\eta = NS \cup sup\,[Gen_{sym}(\eta)] \cup sup\left[Gen_{asy_1}(\eta)\right]$, a *computational state* of the protocol is given by $(F, L, H, Corr\text{-}Users)$, where $L, H, Corr\text{-}Users$ are defined analogously to their formal counterparts, and F is also defined similarly to f by replacing Exp_b with C_η. The adversary interacts with a *computational oracle* by issuing the four types of queries explained above, where the input/output of queries are probabilistic, depending on $\mathcal{R}_A$ and $\mathcal{R}_H$. (Here $\mathcal{R}_H$ is the concatenation of all random coins used by honest parties.) Among oracle queries, we only explain the effect of a **corruption** query (the others are fairly straightforward): if the adversary corrupts u_i, he is given (pk_i, sk_i), and for every session sn in which u_i takes the role $X \in \{I, R\}$, the adversary is given $F(X, bs, sn)$, for every $bs \in BS(\Pi)$. Finally, under fixed $\mathcal{R}_H$ and $\mathcal{R}_A$, the induced *computational trace* is deterministic and denoted by $\mathcal{CT}(\mathcal{A}, \mathcal{R}_A, \mathcal{R}_H, \Pi_{\mathcal{E}_p})$.
Let $FT = \langle (f_1, l_1, h_1, corr\text{-}users_1), (f_2, h_2, l_2, corr\text{-}users_2), \dots \rangle$ be a formal trace and let $\tau : Exp_{basic} \to \mathcal{C}_\eta$ be a concrete mapping. We say a concrete trace

$$CT = \langle (F_1, L_1, H_1, Corr\text{-}Users_1), (F_2, L_2, H_2, Corr\text{-}Users_2), \dots \rangle$$

is an *encoding* of FT under τ, written $FT \prec_\tau CT$, if $l_i = L_i$, $h_i = H_i$, $Corr\text{-}Users = corr\text{-}users$ and $F_i = \tau f_i$, for all $i \geq 1$. We say CT is the *computational image* of FT, written $FT \prec CT$, if there exists τ such that $FT \prec_\tau CT$.

We are now ready give the computational soundness definition.

Definition 1. *A pair $\mathcal{E}_p = (\mathcal{E}_{asy}, \mathcal{E}_{sym})$ provides a computationally-sound interpretation of symbolic encryption with respect to coinductive Dolev-Yao traces (shortly, provides soundness) if for all valid protocols Π, adversaries $\mathcal{A}_c$, we have*

$$\Pr_{\mathcal{R}_A, \mathcal{R}_H}[\exists \{coind\text{-}legit\ \mathcal{A}_F\} : \mathcal{FT}(A_F) \prec \mathcal{CT}(\mathcal{A}_c, \mathcal{R}_A, \mathcal{R}_H, \Pi_{\mathcal{E}_p})] \geq 1 - negl(\eta)$$

4 Computational Realization of Coinductive Methods

We describe a joint notion of security for asymmetric/symmetric encryption which provides soundness for coinductive symbolic traces. We then explore how this notion may be achieved under standard complexity-theoretic assumptions.

We begin with some motivation. Consider a single run of a protocol against a passive adversary, in which the whole sequence of exchanged messages is known *a priori*. We wish to formalize what it means for a piece of data (nonce or symmetric key) to remain *secure* in both the formal and computational settings. Under the formal approach, one would typically say the secrecy of a piece of data is retained if it cannot be deduced by applying Dolev-Yao rules. For a nonce X, for instance, if X is not formally deducible, it means all occurrences of X are encrypted under keys which cannot be obtained by a Dolev-Yao adversary. Thus, under the concrete instantiation, after the adversary has received the computational representations of the exchanged messages, the random nonce value underlying X should still be as computationally random as a freshly-generated random nonce, provided the encryption scheme is sufficiently strong. However, for the case of symmetric keys the situation is quite different: even if a symmetric key is not Dolev-Yao-style deducible, the key may leak significant information when it comes to a concrete implementation. For instance, a symmetric-key value may lose its original randomness if used for encryption. (i.e. The adversary will be able to tell it apart from a fixed key, causing it to not be as "random" as a freshly generated key.) Thus the definition of secrecy for symmetric keys in the computational model turns out to be more delicate.

Our ultimate goal is to establish a close correspondence between coinductive Dolev-Yao adversaries and computational adversaries, by showing that a computational adversary essentially cannot do anything (in terms of mounting successful attacks) which cannot already be performed by a simple Dolev-Yao adversary. We capture the essence of active-attack scenario within a *cryptographic game*, played between an adversary and a challenger, in which the adversary is faced with a number of unknown keys (both asymmetric and symmetric) and nonces, generated by the challenger, and his goal is to infer "non-trivial" information from the challenger's secret data, by exploiting active attacks such as corrupting arbitrary keys of the challenger, getting her to encrypt messages which depend on her own secret data, and getting her to decrypt "permissible" ciphertexts. Our goal is to show that, under sufficiently strong security requirements, the computational adversary cannot learn non-trivial information from a piece of data (nonce or private key) that cannot already be obtained by a coinductive Dolev-Yao adversary. The key point in our security definition is to formalize the idea of "computational secrecy" for private keys. As it is probably clear from the above discussion, "requiring the adversary not be able to distinguish the private key (used in the game) from a freshly generated key" would not work. We formalize it in the following standard way: a private key retains its computational secrecy if the adversary is unable to distinguish between the encryptions of real/random messages under that key. We will be able to show that security in our game provides computational soundness.

Our security notion is formalized via the following game which we call the *coinductive, key-dependent indistinguishability* game, or the *CI game* for short. Below, for $S = \{(s_1^1, s_1^2), \ldots (s_n^1, s_n^2)\}$, we define $S^i \triangleq \{s_1^i, \ldots, s_n^i\}$ for $i \in \{1, 2\}$.

4.1 Coinductive, Key-Dependent Indistinguishability (CI) Game

Assume $\mathcal{E}_{asy} = (Gen_{asy}, Enc_{asy}, Dec_{asy})$ and $\mathcal{E}_{sym} = (Gen_{sym}, Enc_{sym}, Dec_{sym})$ are asymmetric/symmetric encryption schemes whose joint security is to be defined, w.l.o.g., w.r.t. the shared *security parameter* 1^η. The game is played between an adversary, $\mathcal{A}$, and a challenger, $\mathcal{B}$, and is parameterized over a publicly-known, poly-bounded integer function $n(\eta)$ (we write n for $n(\eta)$). Suppose $\tau_B(\cdot)$ and $\tau_A(\cdot)$ are (dynamically growing) mappings which give bitstring values to, respectively, the basic symbols of $\mathcal{B}$ and $\mathcal{A}$ (we will see shortly what those symbols are), and let τ be a mapping defined to be τ_B on the domain of $\mathcal{B}$'s symbols and τ_A on $\mathcal{A}$'s. Here τ_A is publicly known, while access to τ_B and τ is restricted to $\mathcal{B}$. The game proceeds in three phases: **setup**, **interaction**, and **guessing**.

In the **setup** phase, $\mathcal{B}$ first picks $b \leftarrow \{0,1\}$, generates $\{(pk_i, sk_i)\}_{1\le i\le n} \leftarrow Gen_{asy}(\eta)$, symmetric keys $\{ck_i\}_{1\le i\le n} \leftarrow Gen_{sym}(\eta)$, and nonces $\{nc_i\}_{1\le i\le n} \leftarrow \{0,1\}^{q(\eta)}$ (for some poly q), makes $\{pk_i\}_{1\le i\le n}$ public, and keeps the rest secret. We introduce $\{(k_i, k_i^{-1})\}_{1\le i\le n} \in \mathcal{K}_H^{pub} \times \mathcal{K}_H^{privasym}$, and $\{k_i^{sym}\}_{1\le i\le n} \in \mathcal{K}_H^{privsym}$, and $\{x_i\}_{1\le i\le n} \in \mathcal{X}_H$, and assign $\tau_B(k_i) = pk_i$, $\tau_B(k_i^{-1}) = sk_i$, $\tau_B(k_i^{sym}) = ck_i$ and $\tau_B(x_i) = nc_i$, for $1 \le i \le n$. We initialize $\mathit{eval\text{-}exp} = \emptyset$. During the **interaction** phase, $\mathcal{A}$ may dynamically update τ_A, mapping his newly-created basic symbols to arbitrary values. In the **interaction** phase $\mathcal{A}$ *adaptively* interacts with $\mathcal{B}$ by issuing queries of the following types:

1. *Corruption*: $\mathcal{A}$ may corrupt a $\mathcal{B}$'s key by issuing $\mathit{corrupt}(s)$, where $s \in \{k_1^{-1}, \ldots, k_n^{-1}, k_1^{sym}, \ldots, k_n^{sym}\}$. In response $\mathcal{A}$ receives $\tau(s)$, and $(s, \tau(s))$ is added to *eval-exp*.
2. *Encryption*: $\mathcal{A}$ may issue a query $\mathit{encrypt}(e, x)$, where $x \in \{k_1, \ldots, k_n, k_1^{sym}, \ldots, k_n^{sym}\}$, and e may not have any k_i^{-1}'s as a subexp. In response, $\mathcal{A}$ is given $c \leftarrow [\![\{e\}_x]\!]_\tau$ and $(\{e\}_x, c)$ is added to *eval-exp*. Here e may contain both the challenger's and adversary's basic symbols.
3. *Decryption*: $\mathcal{A}$ may issue $\mathit{decrypt}(c, s')$, where $s' \in \{k_1^{-1}, \ldots, k_n^{-1}, k_1^{sym}, \ldots, k_n^{sym}\}$. In response $\mathcal{A}$ receives $Dec_{asy}(c, sk_i)$ if $s' = k_i^{-1}$ and $Dec_{sym}(c, ck_i)$ if $s' = k_i^{sym}$, *unless* there exists $(\{e\}_{k_p}, c_p) \in \mathit{eval\text{-}exp}$ such that $\{e\}_{k_p}$ has a subexp $\{e'\}_s$ (where $s' = s^{-1}$) which in $\{e\}_{k_p}$ is encrypted only under keys whose decryption keys are in $\mathit{closure}_c(\mathit{eval\text{-}exp}^1)$, and that c corresponds to the computational image of $\{e'\}_s$ in c_p. In this case the answer is $\perp$.

After making a number of such queries, $\mathcal{A}$ proceeds to the final, **guessing** phase, in which he claims he is able to infer "non-trivial" information about irrecoverable secret data of $\mathcal{B}$. He does so by issuing a *challenge* query, which is either of the form $\mathit{challenge}(s)$, where $s \in \{x_1, \ldots, x_n\}$ (nonce challenge), or of the form $\mathit{challenge}(s, bs)$, where $bs \in \{0,1\}^*$ and $s \in \{k_1^{-1}, \ldots, k_n^{-1}, k_1^{sym}, \ldots, k_n^{sym}\}$ (secret key challenge.) The response to the query is decided as follows: if $s \in \mathit{closure}_c(\mathit{eval\text{-}exp}^1)$, then he is given $\perp$, otherwise:

- if $b = 0$, $\mathcal{A}$ is given nc_j if $s = x_j$, $Enc_{asy}(bs, pk_j)$ if $s = k_j^{-1}$, and otherwise $Enc_{sym}(bs, ck_j)$ if $s = k_j^{sym}$.
- if $b = 1$, $\mathcal{A}$ is given $nc'_j \leftarrow \{0,1\}^{q(\eta)}$ if $s = x_j$, $Enc_{asy}(r, pk_j)$ if $s = k_j^{-1}$, and $Enc_{sym}(r, ck_j)$ if $s = k_j^{sym}$, where $r \leftarrow \{0,1\}^{|bs|}$.

$\mathcal{A}$ finally outputs his guess for b. Denoting by $\mathcal{A}^{\mathrm{CI}_b^{\mathcal{E}_p}}$ the output of $\mathcal{A}$ when the secret bit is b, his *CI-advantage* is (below $\mathcal{E}_p$ refers to the pair of schemes):

$$Adv^{\mathrm{CI}}_{\mathcal{E}_p,\mathcal{A}}(\eta) = \big|\Pr[\mathcal{A}^{\mathrm{CI}_b^{\mathcal{E}_p}}(\eta) = 1 \mid b = 0] - \Pr[\mathcal{A}^{\mathrm{CI}_b^{\mathcal{E}_p}}(\eta) = 1 \mid b = 1]\big|.$$

Definition 2. *A pair of $\mathcal{E}_p = (\mathcal{E}_{sym}, \mathcal{E}_{asy})$ provides joint security under the CI game (shortly, is CI-secure) if for every $\mathcal{A}$, $Adv^{CI}_{\mathcal{E}_p,\mathcal{A}}(\eta)$ is negligible.*

We now explain the restrictions on *challenge* and *decryption* queries. For our discussion, assume that $\mathcal{E} = (Gen, Enc, Dec)$ is a symmetric encryption scheme wherein $Enc(ck, ck)$ leads to computation of ck. (This could happen although $\mathcal{E}$ is secure in any standard sense.) In the absence of the condition for *challenge* queries, $\mathcal{A}$ could simply win the game by doing the following: make two queries $encrypt(k_1^{sym}, k_1^{sym})$ and $encrypt(k_2^{sym}, k_1^{sym})$ to receive, respectively, c_1 and c_2, and then issue the *challenge* query $challenge(k_2^{sym}, 0^n)$; $\mathcal{A}$ may now obtain $\tau(k_1^{sym})$ from c_1 and $\tau(k_2^{sym})$ from c_2, trivially winning the game. Also in the absence of the condition for *decryption* queries, $\mathcal{A}$ could simply win as follows: (1) make two queries $encrypt(k_1^{sym}, k_1^{sym})$ and $encrypt(\{k_2^{sym}\}_{k_3^{sym}}, k_1^{sym})$ to receive, respectively, c_1 and c_2, (2) after computing $\tau(k_1^{sym})$ from c_1, issue the *decryption* query $decrypt(c_3, k_3^{sym})$, where $c_3 = Dec(c_2, \tau(k_1^{sym}))$, and (3) after obtaining $\tau(k_2^{sym})$ issue the *challenge* query $challenge(k_2^{sym}, 0^n)$, trivially winning the game. Finally we remark that the recent results of [20] show that there exists an IND-CCA2-secure symmetric encryption scheme such that ciphertexts $Enc(ck_1, ck_2), \ldots, Enc(ck_{n-1}, ck_n), Enc(ck_n, ck_1)$, for randomly-generated ck_i's, lead to revelation of all $ck_1, \ldots, ck_n$ (a weaker case than k-circular security). Therefore, the above attack methods extend to longer key cycles.

Note, $\mathcal{A}$ may use an *encrypt* query to obtain the encryption of any bitstring. For example, to encrypt m under ck_i, he may introduce a new basic symbol $x_{\mathcal{A}}$, set $\tau_{\mathcal{A}}(x_{\mathcal{A}}) = m$, and then issue $encrypt(x_{\mathcal{A}}, k_i^{sym})$. Also it is possible to define and extend results we present about CI security to a (seemingly) stronger notion in which $\mathcal{A}$ is allowed to make multiple *challenge* queries, possibly making them interleave with the other types of queries. Right now for applications that we consider, CI security suffices. CI security may be thought of as a variant of KDM security with the underlying function family consisting of any arbitrary composition of *constant*, *projection*, *pairing* and *encryption* functions. However, since we aim to prove generic implication results from standard security definitions, we have to restrict the set of keys for which we want to prove computational secrecy (i.e. those which remain coinductively irrecoverable). This differs from KDM security in which one wants to prove computational secrecy for all keys, regardless of what encryption queries were made. Finally we stress that a key $\mathcal{A}$ challenges in the **guessing** phase may have previously occurred in key cycles.

CI security is still insufficient for providing soundness as it does not provide *integrity of ciphertexts*. To account for this, we strengthen it to also provide ciphertext integrity and call the new notion *authenticated* CI (or ACI) security. We say $(\mathcal{E}_{asy}, \mathcal{E}_{sym})$ is *ACI secure* if it is CI secure and further any $\mathcal{A}$ has a negligible chance of winning in the ACI game defined as follows: the **setup** and **interaction** phases proceed exactly as in the CI game, while in the **guessing** phase, $\mathcal{A}$ outputs (c, i) and wins if: (1) $Dec_{sym}(c, ck_i) \neq \perp$, and (2) there does not exist $(\{e\}_{k_j}, c') \in \textit{eval-exp}$ such that $\{e\}_{k_j}$ has a subexp $\{e'\}_{k_i}$ encrypted in $\{e\}_{k_j}$ only under keys whose decryption keys are in $\textit{closure}_c(\textit{eval-exp}^1)$, and that c is the corresponding image of $\{e'\}_{k_i}$ in c'.

As a step toward proving soundness with respect to ACI security, we formulate a new notion which characterizes security requirements capturing the basic Dolev-Yao assumptions made in protocol analysis, and prove that it provides soundness. Our notion, which we call *coinductive, key-dependent non-malleability* (shortly *CNM*) notion, is a generalization of the *Dolev-Yao non-malleability* notion of [28], which was defined for the passive setting.

4.2 Coinductive, Key-Dependent Non-Malleability (CNM) Game

The game is parameterized, again, over $\mathcal{E}_p = (\mathcal{E}_{asy}, \mathcal{E}_{sym})$, a shared security parameter η, and a computational mapping τ, and runs in three phases with the **setup** and **interaction** phases as in the CI game (except that no b is sampled). However, in the **guessing** phase, $\mathcal{A}$ claims he is able to construct the computational image of an expression which is not coinductively constructible from $\textit{eval-exp}^1$. To this end, he outputs (e, E), where $e \in Exp$ (containing, possibly, both the adversary's and challenger's symbols) and $E \in \{0,1\}^*$. The output of the game is 1, written as $\mathrm{CNM}_{\mathcal{E}_p,\eta}(\mathcal{A}) = 1$, if the following two conditions hold:

1. $e \notin \textit{closure}_c(\textit{eval-exp}^1)$; and
2. E is a possible mapping of e under τ and $\mathcal{E}_p$; namely, $E \in [\![e]\!]_\tau^{\mathcal{E}_p}$.

Note condition (2) is efficiently verifiable given access to τ. We define

$$Adv^{\mathrm{CNM}}_{\mathcal{E}_p,\eta}(\mathcal{A}) = \Pr[\mathrm{CNM}_{\mathcal{E}_p,\eta}(\mathcal{A}) = 1].$$

Definition 3. *A pair $\mathcal{E}_p = (\mathcal{E}_{asy}, \mathcal{E}_{sym})$ provides security under the CNM game (shortly, is CNM-secure) if for every adversary $\mathcal{A}$, $Adv^{CNM}_{\mathcal{E}_p,\eta}(\mathcal{A})$ is negligible.*

Theorem 1. *1. CNM security $\Rightarrow$ soundness*
2. ACI security $\Rightarrow$ CNM security.

Proof (Outline): For (2) if $\mathcal{A}^{cnm}$ is able to output a CNM-valid (e, E), then $e \notin T$, where $T = \textit{closure}_c(\textit{eval-exp}^1)$ implies e has a subexp s such that s is either a nonce/private key, or $s = \{\cdot\}_{k_j^{sym}}$, and that any subexp of e which contains s is not in T. This implies the underlying value of s is recoverable from E (with the aid of the decryption oracle) through successive decryptions down along the path leading to s, which will then enable an attack either against CI security or ciphertext integrity depending on the type of s. The proof for (1) also follows using ideas similar to those of [34]. We give a full proof in [27]. □

For an adversary $\mathcal{A}$ in either of the above games, we define a *labeled key graph*, $G(\mathcal{A}) = (V_{\mathcal{A}}, E_{\mathcal{A}})$, as follows: $V_{\mathcal{A}} = \{v_1^{asy}, \ldots, v_n^{asy}, v_1^{sym}, \ldots, v_n^{sym}\}$, and $v_i^x \xrightarrow{a} v_j^y \in E_{\mathcal{A}}$, for $x, y \in \{asy, sym\}$ and $a \in \mathbb{N}$, if k_i^x encrypts the ath *occurrence* of $(k_j^y)^{-1}$ in the sequence of $\mathcal{A}$'s *encryption* queries. Here ath occurrence refers to an increasing numbering given to each decryption key as it appears in the sequence; for example, if $e_1 = \{k_1^{sym}, k_p^{sym}\}_{k_3}, k_p^{sym}$ and $e_2 = k_2^{sym}, \{k_p^{sym}\}_{k_4}$ and the first two *encryption* queries are $encrypt(e_1, k_3)$ and $encrypt\,(e_2, k_5)$; the set of keys that encrypt the 3rd occurrence of k_p^{sym} is $\{k_4, k_5\}$. We call v_i^x *coinductively irrecoverable* (irrecoverable for short) if $k_i^{x\,-1} \notin closure_c(eval\text{-}exp^1)$, and we refer to the *induced* subgraph on irrecoverable nodes as the *hidden subgraph*. The *diameter* of a graph is the length of the longest path in the graph. We define $indeg(v_i^x)$ to be the maximum a for which we have an incoming edge with label a to v_i^x; this specifies the number of times $k_i^{x\,-1}$ occurs in $\mathcal{A}$'s *encryption* queries. Note, $indeg(v_i^{asy}) = 0$, for every $1 \leq i \leq n$, and also both $G(\mathcal{A})$ and $indeg(v_i)$ are random variables depending on the coins tossed during the game.

If all *encryption* queries were of the form $encrypt(k_i^{sym}, k_j^x)$ (i.e. single encryptions) without cycle creation, then all nodes from which there was a path to an irrecoverable node would also be irrecoverable. However, in the case of nested encryptions with key cycles, the above appealing property no longer holds; namely, an irrecoverable node may occur in certain key cycles, and may have edges from nodes which are recoverable. For example, assuming $e_1 = \{k_1^{sym}\}_{k_2^{sym}}$ and $e_2 = \{k_3^{sym}\}_{k_4^{sym}}$, if $\mathcal{A}$ makes queries $encrypt(e_1, k_5^{sym})$, $encrypt(k_2^{sym}, k_1^{sym})$, $encrypt(e_2, k_6^{sym})$, and $corrupt(k_4^{sym})$, all keys except k_4^{sym} remain irrecoverable, and there exists, for instance, edges in both directions between v_1^{sym} and v_2^{sym} in $G(\mathcal{A})$.

However, in the case of cyclic nested encryption, we will base our hybrid arguments on a provable property, which we call *coinductive continuability*, of irrecoverable nodes. In $G(\mathcal{A})$, we say $v_{y_1}^x \xrightarrow{a_2} v_{y_2}^{sym} \xrightarrow{a_3} \ldots \xrightarrow{a_p} v_{y_p}^{sym}$, for $x \in \{sym, asy\}$, is a *coinductively continuable path* if the following conditions hold: (below for better clarity we drop the superscripts x and sym.)

1. Path validity: For all $2 \leq i \leq p$, $v_{y_{i-1}} \rightarrow^{a_i} v_{y_i} \in E_{\mathcal{A}}$, and if $1 \leq w < h \leq p$ then $v_{y_w} \neq v_{y_h}$,
2. For all $s \in \{k_{y_1}^{x\,-1}, k_{y_2}^{sym}, \ldots, k_{y_p}^{sym}\}$ it holds $s \notin closure_c(eval\text{-}exp^1)$, and
3. either $indeg(v_{y_1}) = 0$ or for *every* $1 \leq a_1 \leq indeg(v_{y_1})$ there exists v_i^w, with $w \in \{asy, sym\}$, such that $v_i^w \xrightarrow{a_1} v_{y_1} \xrightarrow{a_2} \ldots \xrightarrow{a_p} v_{y_p}$ is a coinductively continuable path.

We call v_i^x *coinductively continuable* if its associated path of length zero is so.

Lemma 1. *At any point, any irrecoverable node is coinductively continuable.*

Proof (Outline): We prove this by an induction over the length of the longest path ending in the irrecoverable node. A full proof is given in [27]. □

Definition 4. *We say that $\mathcal{E}_p = (\mathcal{E}_{asy}, \mathcal{E}_{sym})$ provides l-CI security if $Adv_{\mathcal{E},\mathcal{A}}^{CI}(\eta)$ is negligible for every $\mathcal{A}$ for whom the diameter of the hidden subgraph of $G(\mathcal{A})$ is*

always at most l. We say $\mathcal{E}_p$ provides l-ACI security if it is l-CI secure and any $\mathcal{A}$ (under the ACI game) for which the diameter of the resulting hidden subgraph is always at most l has a negligible advantage.

Theorem 2. *If $\mathcal{E}_{asy}$ and $\mathcal{E}_{sym}$ are both IND-CCA2 secure, then $(\mathcal{E}_{asy}, \mathcal{E}_{sym})$ provides l-CI security, for every constant l.*

Proof (Outline): The central idea is to guess a "random", coinductively continuable path, with some associated parameters, which ends in the **challenge** key, give "fake" values to certain private keys occurring as plaintexts, and prove the adversary's advantage under this replying strategy is negligibly different from that under the standard game. A full proof is given in [27]. □

Theorem 3. *If $\mathcal{E}_{asy}$ provides IND-CCA2 security, and $\mathcal{E}_{sym}$ provides both IND-CCA2 and INT-CTXT security, then $(\mathcal{E}_{asy}, \mathcal{E}_{sym})$ provides l-ACI security, for every constant l.*

Proof (Outline): We first show if $\mathcal{A}_{aci}$ is able to output an ACI-valid (c, i), then in a world, W_i, in which occurrences of k_i^{sym} as a plaintext and its occurrences as an encryption key are given two independent values, $\mathcal{A}_{aci}$ should have "the same" probability of producing a valid (c, i), or otherwise a CI-attack can be made. Next, we show if under W_i an adversary $\mathcal{A}$ is able to produce an ACI-valid (c, i) and c is already a plaintext of a ciphertext obtained under an *encryption* query (e.g. $\mathcal{A}$ has called $encrypt(\{x_1\}_{k_i^{sym}}, k_2^{sym})$ to obtain c_2, k_2^{sym} remains coinductively irrecoverably, and $c = Dec_{sym}(c_2, ck_2)$), then a CI attack follows, and otherwise an INT-CTXT attack follows. A full proof is given in [27]. □

5 Conclusion

We investigated soundness of coinductive methods in a protocol model allowing arbitrary composition of symmetric/asymmetric encryption, as well as unrestricted transmission of secret keys. In such situations, an active adversary may selectively influence the encryption ordering between deployed keys, dynamically compromise them (naturally or under his corruption power), and potentially obtain encryption cycles. Any weakness in the underlying encryption schemes in the face of such an adversary may lead to insecure instantiations of protocols. Most previous work on computationally sound symbolic analysis of protocols either does not allow symmetric encryption, or imposes restrictions aimed at avoiding the above possibilities. Our soundness theorem, founded on coinduction, does not assume any such restrictions, while providing strong computational security guarantees against adaptive corruptions. Our results, however, rely on a property of protocols we call *boundedness* (formalized in [27]), which requires that no symbolic execution of the underlying protocol produce a coinductively-irrecoverable encryption chain of nonconstant length. We observe that almost all protocols from [21] (when run in *isolation*) admit (at most) 2-boundedness.

(All of them are bounded.) In [27], we provide statements on how one can reason about boundedness of a protocol, and whether the boundedness property is retained when two (individually bounded) protocols are run concurrently.

While the main focus of this paper is on trace-based security, we believe similar results can also be proved for *key-exchange* (KE) security tasks. A central security requirement for key exchange is the *secrecy* condition, requiring a secret key exchanged by a KE protocol be *indistinguishable* from a freshly generated key. Our CI game is rich enough to encompass common features of a KE attack model, including adaptive corruptions of users and session keys, while guaranteeing that (under stated complexity assumptions) *coinducitve symbolic secrecy* under the game implies computational secrecy (*real-or-random indistinguishability* in the case of nonces and *key usability* [23] in the case of secret keys).

For simplicity we have assumed if a user is corrupted, the adversary obtains *only* her long-lived key and her past generated secret keys/nonces, but *not* her past *random coins*. In [27] we give partial results about this more general case.

As briefly explained in the introduction, current results about KDM security do not seem sufficient for (unrestricted) secure realizations of protocols with *inductive*, symbolic security proofs. It would be interesting to extend (and realize) KDM security definitions to support adaptive corruptions. As pointed out earlier, defining the extension in an entirely left-or-right indistinguishability sense, as in [5], would entail inherent limitations; for example, if a left-or-right encryption query is made under ck, then ck cannot be corrupted afterward.

Finally it would be interesting to improve the bounds imposed by our soundness theorem (and those of [36]), and investigate its extensions to more general cryptographic frameworks supporting *compositional reasoning* [6, 19].

References

1. Martín Abadi and Phillip Rogaway. Reconciling two views of cryptography (the computational soundness of formal encryption). In *Proceedings of the International Conference IFIP on Theoretical Computer Science, Exploring New Frontiers of Theoretical Informatics*, TCS '00, pages 3–22, London, UK, 2000. Springer-Verlag.
2. Tolga Acar, Mira Belenkiy, Mihir Bellare, and David Cash. Cryptographic agility and its relation to circular encryption. In Henri Gilbert, editor, *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 403–422. Springer, 2010.
3. Michael Backes, Markus Dürmuth, and Dominique Unruh. Oaep is secure under key-dependent messages. In Josef Pieprzyk, editor, *ASIACRYPT*, volume 5350 of *Lecture Notes in Computer Science*, pages 506–523. Springer, 2008.
4. Michael Backes and Birgit Pfitzmann. Symmetric encryption in a simulatable dolev-yao style cryptographic library. In *CSFW*, pages 204–218. IEEE Computer Society, 2004.
5. Michael Backes, Birgit Pfitzmann, and Andre Scedrov. Key-dependent message security under active attacks - brsim/uc-soundness of symbolic encryption with key cycles. In *CSF*, pages 112–124. IEEE Computer Society, 2007.
6. Michael Backes, Birgit Pfitzmann, and Michael Waidner. A composable cryptographic library with nested operations. In Sushil Jajodia, Vijayalakshmi Atluri,

and Trent Jaeger, editors, *ACM Conference on Computer and Communications Security*, pages 220–230. ACM, 2003.
7. Gergei Bana and Hubert Comon-Lundh. Towards unconditional soundness: Computationally complete symbolic attacker. In Pierpaolo Degano and Joshua D. Guttman, editors, *POST*, volume 7215 of *Lecture Notes in Computer Science*, pages 189–208. Springer, 2012.
8. Gilles Barthe, Marion Daubignard, Bruce M. Kapron, and Yassine Lakhnech. Computational indistinguishability logic. In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, *ACM Conference on Computer and Communications Security*, pages 375–386. ACM, 2010.
9. Mihir Bellare, Alexandra Boldyreva, and Silvio Micali. Public-key encryption in a multi-user setting: security proofs and improvements. In *Proceedings of the 19th international conference on Theory and application of cryptographic techniques*, EUROCRYPT'00, pages 259–274, Berlin, Heidelberg, 2000. Springer-Verlag.
10. Mihir Bellare, Anand Desai, David Pointcheval, and Phillip Rogaway. Relations among notions of security for public-key encryption schemes. In Hugo Krawczyk, editor, *CRYPTO*, volume 1462 of *Lecture Notes in Computer Science*, pages 26–45. Springer, 1998.
11. Mihir Bellare and Chanathip Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. *J. Cryptology*, 21(4):469–491, 2008.
12. Mihir Bellare and Phillip Rogaway. Entity authentication and key distribution. In Douglas R. Stinson, editor, *CRYPTO*, volume 773 of *Lecture Notes in Computer Science*, pages 232–249. Springer, 1993.
13. Mihir Bellare and Phillip Rogaway. Provably secure session key distribution: the three party case. In Frank Thomson Leighton and Allan Borodin, editors, *STOC*, pages 57–66. ACM, 1995.
14. Dan Boneh, Shai Halevi, Michael Hamburg, and Rafail Ostrovsky. Circular-secure encryption from decision diffie-hellman. In David Wagner, editor, *CRYPTO*, volume 5157 of *Lecture Notes in Computer Science*, pages 108–125. Springer, 2008.
15. Michael Burrows, Martin Abadi, and Roger Needham. A logic of authentication. *ACM Trans. Comput. Syst.*, 8:18–36, February 1990.
16. Jan Camenisch, Nishanth Chandran, and Victor Shoup. A public key encryption scheme secure against key dependent chosen plaintext and adaptive chosen ciphertext attacks. In Antoine Joux, editor, *EUROCRYPT*, volume 5479 of *Lecture Notes in Computer Science*, pages 351–368. Springer, 2009.
17. Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *FOCS*, pages 136–145. IEEE Computer Society, 2001.
18. Ran Canetti, Uriel Feige, Oded Goldreich, and Moni Naor. Adaptively secure multi-party computation. In Gary L. Miller, editor, *STOC*, pages 639–648. ACM, 1996.
19. Ran Canetti and Jonathan Herzog. Universally composable symbolic analysis of mutual authentication and key-exchange protocols. In Shai Halevi and Tal Rabin, editors, *TCC*, volume 3876 of *Lecture Notes in Computer Science*, pages 380–403. Springer, 2006.
20. David Cash, Matthew Green, and Susan Hohenberger. New definitions and separations for circular security. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *Public Key Cryptography*, volume 7293 of *Lecture Notes in Computer Science*, pages 540–557. Springer, 2012.
21. John Clark and Jeremy Jacob. A survey of authentication protocol literature. Technical report, 1997.

22. Hubert Comon-Lundh and Véronique Cortier. Computational soundness of observational equivalence. In Peng Ning, Paul F. Syverson, and Somesh Jha, editors, *ACM Conference on Computer and Communications Security*, pages 109–118. ACM, 2008.
23. Anupam Datta, Ante Derek, John C. Mitchell, and Bogdan Warinschi. Computationally sound compositional logic for key exchange protocols. In *CSFW*, pages 321–334. IEEE Computer Society, 2006.
24. D. Dolev and A. C. Yao. On the security of public key protocols. *Foundations of Computer Science, Annual IEEE Symposium on*, 0:350–357, 1981.
25. Henri Gilbert, editor. *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30 - June 3, 2010. Proceedings*, volume 6110 of *Lecture Notes in Computer Science*. Springer, 2010.
26. Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.
27. Mohammad Hajiabadi and Bruce M. Kapron. Computational soundness of coinductive symbolic security under active attacks. *IACR Cryptology ePrint Archive*, 2012:560, 2012.
28. Jonathan Herzog. A computational interpretation of dolev-yao adversaries. *Theor. Comput. Sci.*, 340(1):57–81, 2005.
29. Russell Impagliazzo and Bruce M. Kapron. Logics for reasoning about cryptographic constructions. *J. Comput. Syst. Sci.*, 72(2):286–320, 2006.
30. Ralf Küsters and Max Tuengerthal. Computational soundness for key exchange protocols with symmetric encryption. In Ehab Al-Shaer, Somesh Jha, and Angelos D. Keromytis, editors, *ACM Conference on Computer and Communications Security*, pages 91–100. ACM, 2009.
31. Patrick Lincoln, John C. Mitchell, Mark Mitchell, and Andre Scedrov. A probabilistic poly-time framework for protocol analysis. In Li Gong and Michael K. Reiter, editors, *ACM Conference on Computer and Communications Security*, pages 112–121. ACM, 1998.
32. Daniele Micciancio. Computational soundness, co-induction, and encryption cycles. In Henri Gilbert, editor, *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 362–380. Springer, 2010.
33. Daniele Micciancio and Saurabh Panjwani. Adaptive security of symbolic encryption. In Joe Kilian, editor, *TCC*, volume 3378 of *Lecture Notes in Computer Science*, pages 169–187. Springer, 2005.
34. Daniele Micciancio and Bogdan Warinschi. Soundness of formal encryption in the presence of active adversaries. In Moni Naor, editor, *TCC*, volume 2951 of *Lecture Notes in Computer Science*, pages 133–151. Springer, 2004.
35. John C. Mitchell, Ajith Ramanathan, Andre Scedrov, and Vanessa Teague. A probabilistic polynomial-time process calculus for the analysis of cryptographic protocols. *Theor. Comput. Sci.*, 353(1-3):118–164, 2006.
36. Saurabh Panjwani. Tackling adaptive corruptions in multicast encryption protocols. In Salil P. Vadhan, editor, *TCC*, volume 4392 of *Lecture Notes in Computer Science*, pages 21–40. Springer, 2007.
37. Birgit Pfitzmann and Michael Waidner. A model for asynchronous reactive systems and its application to secure message transmission. In *IEEE Symposium on Security and Privacy*, pages 184–, 2001.
38. Charles Rackoff and Daniel R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In Joan Feigenbaum, editor, *CRYPTO*, volume 576 of *Lecture Notes in Computer Science*, pages 433–444. Springer, 1991.