Computational Soundness of Coinductive Symbolic Security under Active Attacks

Mohammad Hajiabadi, Bruce M. Kapron

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

Abstract. In Eurocrypt 2010, Miccinacio initiated an investigation of cryptographically sound, symbolic security analysis with respect to *coinductive* adversarial knowledge, and demonstrated that under an adversarially *passive* model, certain security criteria (e.g. *indistinguishability*) 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 answering this question in the affirmative with respect to an extension of a *trace-based security model* (proposed by Micciancio and Warinschi in 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* which can be achieved under standard notions of encryption security, which may be of independent interest.

Keywords: Computational soundness, adaptive corruptions, coinduction, key-cyclic security, trace-based protocol security, active adversaries, standard security

1 Introduction

Provable security [32], since its introduction in the early 1980s, has provided a rigorous foundation for the security analysis of cryptographic schemes.Typically, proving that 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 [15,16,50]. 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* [30,18,1] greatly simplify security analysis by using idealized abstractions of cryptographic primitives and limiting adversarial computation, even allowing for automated verification (e.g. [43]). 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 [3], attempts to give computational justification 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 [32,48] 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 [24,4] 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 [12] 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 computational soundness results are restricted in their assumptions, which include excluding *key cycles* altogether in the presence of *passive* adversaries [3,2,33], posing certain encryption orderings in the presence of *passive-but-adaptive* adversaries [40,41], and disallowing symmetric encryption in the presence of *active* adversaries [42,27,9,23].

As a resolution to the problems created by key cycles, Micciancio [39] 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, [39] 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, [39] 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 [39]; 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* [42], 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 decryption 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. 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, 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[1]? Several negative results [24,4] 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 [39] 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

[1] Our definition of computational secrecy is close to the idea of *key usability*, developed in [29,49], for defining alternate, composition-amenable security criteria for key-exchange protocols.

underlying encryption ordering, and then use a hybrid argument in the style of [39] 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. Here, the key graph is the (random) multigraph G_k which has a node for every key in the game, and an edge $v_i \to v_j$ 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 [46]'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 corrupt 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 [46] seem to extend, by its mere developed techniques, to an authenticated setting with *nested encryptions*, they crucially rely on the acyclicity assumption 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 [46]. Also, allowing both nested encryption and decryption queries creates a new complication; namely, to simulate a CI adversary $\mathcal{A}^{CI}$ by a CCA2 adversary $\mathcal{A}^{cca}$, *nested encryption* 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 [46] immediately extend if decryption queries are also allowed.

Applications. Our reduction result implies that for a protocol Π (which may contain symmetric keys and nonces as atomic messages) and a *trace-expressible* security property $\mathcal{P}$, 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 co-inductive symbolic adversary may produce a trace not satisfying the underlying symbolic property. We observe that all protocols in the Clark-Jacob library [25], 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. [42,27,9]). Using our techniques, we show that the *Wide-Mouthed Frog authentication protocol*, which is not analyzable under the cryptographic library of [7] 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 might 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* [17,19]. 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. [8] 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, [6] 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 [20,47]. [7] shows that *secure realization* of *ideal* symmetric encryption (in the sense of *reactive simulatability*) is possible in their cryptographic library [9] 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 [36], by extending the framework of [23] 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. (See, e.g., [15,16] for related definitions.) Thus the aforementioned frameworks do not consider the above two attack scenarios. (See, e.g., [5] for the analysis of the *Otway-Rees* protocol under [7].) We remark the commitment problem was known long before in the setting of *adaptively-secure multiparty computation*, with initial solutions given in [22,21]. (See also [45] for a limitation of *non-committing encryption*.)

The results of [26] 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 security assumptions. Although [26] allows symmetric encryption, it imposes the same restrictions as [7,36]. Security under adaptive corruptions is also discussed in [35,38], but only considering the restricted case where the adversary may only see encryptions of his own generated messages.

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 [37,44], logics which axiomatize computational indistinguishability [34,11] and first-order logics augmented with axioms characterizing specific security properties [10]. 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 [13,14]. 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 real-valued) 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*, $\{\diamond\}_\circ$, 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 $(.)^{-1} : \mathcal{K}^{pub} \cup \mathcal{K}^{privsym} \to \mathcal{K}^{priv}$, 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:

$$Exp ::= \mathcal{K}^{privasym} \mid Plain \mid Cipher \mid (Exp, Exp) \tag{1}$$

$$Plain ::= ID \mid \mathcal{X} \mid \mathcal{K}^{pub} \mid \mathcal{K}^{privsym} \mid (Plain, Plain)$$

$$Cipher ::= \{Plain\}_{k \in \mathcal{K}^{pub} \cup \mathcal{K}^{privsym}} \mid \{Cipher\}_{k \in \mathcal{K}^{pub} \cup \mathcal{K}^{privsym}} \mid (Cipher, Cipher)$$

Coniductive Modeling of Adversarial Knowledge. We take a coinductive approach to modeling adversarial attacks. To model *coinductive adversarial knowledge* [39], 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 with respect to $\subseteq$.

The Tarski-Knaster Theorem implies that 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

$$fix(F) = \bigcap_{S : F(S) \subseteq S} S \tag{2}$$

$$FIX(F) = \bigcup_{S : S \subseteq F(S)} S \tag{3}$$

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:

$$fix(\mathcal{F}_e) = \bigcap_{\mathcal{F}_e(S) = S} S = \bigcup_{i \geq 1} \mathcal{F}_e^i(\emptyset) \tag{4}$$

$$FIX(\mathcal{F}_e) = \bigcup_{S = \mathcal{F}_e(S)} S = \bigcap_{i \geq 1} \mathcal{F}_e^i(\mathcal{K}_H^{priv}) \tag{5}$$

We show (5); the proof for (4) follows by a dual argument. The first equality for $FIX(\mathcal{F}_e)$ follows from (3) 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).)

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 essentially inductive, an equivalent, coinductive definition is also possible; however, we adopt the inductive one as it is more natural. Now e_1 is *coinductively recoverable* from e if $e_1 \in closure_c(e)$. Note that, if $e_1 \in closure_c(e)$ and $k^{sym} \notin closure_c(e)$, the last rule does not allow us to deduce that $\{e_1\}_{k^{sym}} \in closure_c(e)$. 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 that 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 $k_1 \rightarrow^e k_2^{-1}$, 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[39]. The converse of this, however, does not hold true (e.g. consider $\{\{k_1^{-1}\}_{k_1}\}_{k_2}$); it is possible some keys occur in certain key cycles but remain coinductively irrecoverable. 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 *concrete 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; we omit the formal definition here.

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 [42] 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 protocol Π as a tuple of messages, we denote its ith component (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 protocol 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-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 *session number*, *sn*, to their session and gives the adversary this number 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 the message is sent to the responder of session *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 $\mathbf{K}_{i,sn}^{sym} = \{k_{i,sn,j}^{sym} \mid j \in \mathbb{N}\} \subseteq \mathcal{K}_H^{privsym}$, $\mathbf{X}_{i,sn} = \{x_{i,sn,j} \mid j \in \mathbb{N}\} \subseteq \mathcal{X}_H$. The adversary may use his own basic symbols to build new messages; we denote the adversary's symmetric keys and nonces, respectively, by $\mathbf{K}_A^{sym} = \{k_{A,j}^{sym} \mid j \in \mathbb{N}\} \subseteq \mathcal{K}_A^{privsym}$, $\mathbf{X}_A = \{x_{A,j} \mid j \in \mathbb{N}\} \subseteq \mathcal{X}_A$. We let Exp_{basic} be the union of all $\mathbf{X}_A, \mathbf{K}_A^{sym}$, $\mathbf{K}_{i,sn}^{sym}$'s, $\mathbf{X}_{i,sn}$'s.

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

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

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 that 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 of asymmetric/symmetric schemes $\mathcal{E}_p = (\mathcal{E}_{asy}, \mathcal{E}_{sym})$, with w.l.o.g. a shared security parameter 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 becomes a participant of, 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 $C_\eta = NS \cup sup\left[Gen_{sym}(\eta)\right] \cup sup\left[Gen_{asy_1}(\eta)\right]$, a *computational state* of the protocol is characterized 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 just 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, and the evolved computational states 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 computational mapping. We say that a computational 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 a mapping τ such that $FT \prec_\tau CT$.

Computational soundness theorems relate the random computational traces to (coinductive) symbolic traces by characterizing cryptographic assumptions under which it is guaranteed that a random computational trace is (almost always) an encoding of a (coinductive) symbolic trace.

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

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

4 Active Adversaries and the Creation of Key Cycles

We present an example to show how a protocol run in an active setting may lead to creation of key cycles, although the protocol does not produce them by itself. (i.e. when run in a passive setting.) Then we illustrate an inherent limitation of inductive methods in providing *sound* security proofs, when encryption is realized under standard notions of encryption security.

Example 1. Consider the following protocol over A, B, C with public keys k_A, k_B, k_C:

$$A \to B : \ (\{k_1, I_C\}_{k_B}, \{k_2, I_C\}_{k_B}, \{I_A\}_{k_B})$$
$$B \to C : \ (\{k_1\}_{k_C}, \{k_2\}_{k_1})$$

Suppose u_1, u_2, u_3, and u_4 are four protocol users with the respective public key k_{u_i} for u_i. Suppose u_1 starts a session of the protocol by sending $(\{k_1, I_{u_3}\}_{k_{u_2}}, \{k_2, I_{u_3}\}_{k_{u_2}}, \{I_{u_1}\}_{k_{u_2}})$ to u_2. As a result, u_2 sends $(\{k_1\}_{k_{u_3}}, \{k_2\}_{k_1})$ to u_3. The adversary at this point, having received all messages, can start a new session with u_2, claiming himself to be u_4, by sending $(\{k_2, I_{u_3}\}_{k_{u_2}}, \{k_1, I_{u_3}\}_{k_{u_2}}, \{I_{u_4}\}_{k_{u_2}})$. Subsequently, u_2 sends $(\{k_2\}_{k_{u_3}}, \{k_1\}_{k_2})$ to u_3. Now the adversary has obtained both $\{k_2\}_{k_1}$ and $\{k_1\}_{k_2}$, which form an encryption cycle.

Given the attack sketched above, if the underlying encryption scheme is not 2-*circular* secure, the adversary may obtain significant information about the bitstring values of k_1 and k_2, leading him to compute new messages, not otherwise computable, depending on them. Now let's look at an inductive, formal modeling of the attack: after the adversary receives $\{k_2\}_{k_1}$ and $\{k_1\}_{k_2}$, it is assumed that he still cannot deduce k_1 or k_2, and as a result, he cannot use these keys to construct new messages. Thus, it turns out that a set of "feasible" concrete traces (which may lead to a successful attack) may not be matched with any abstract inductive formal trace. In other words, inductive traces are not "expressive enough" to abstractly represent all actual concrete traces which may occur in an actual, concrete execution of a protocol. In contrast, the above attack scenario can abstractly be represented as a valid coinductive Dolev-Yao trace, and, hence, the coinductive approach seems more justifiable for sound formal analysis of protocols.

We remark that although the above example may appear rather unrealistic, key cycles are quite likely to arise in realistic situations, e.g., in *key-distribution* protocols where multiple keys are sent during a single run of a protocol may enable this possibility. Of course, if a protocol is shown to not produce key-cycles when run in the presence of an active adversary (e.g. if it does not transmit secret keys at all), then the inductive and coinductive approaches essentially give rise to the same abstract sets of formal traces, and there is semantically no difference between these two approach. However, the problem of deciding whether a protocol produces key cycles in the active-attack setting is known to be *undecidable*, and thus for a large class of protocols we may not be able to decide whether the protocol leads to key-cycle creation. Even, under a bounded number of sessions, the problem is still *NP-complete* [28]. Finally we note that the *used-order* restriction imposed in [7] to allow symmetric encryption in the *simulatable cryptographic library* of [9] is meant, as one purpose, to rule out key-cycle production.

5 Computational Realization of Coinductive Methods

We describe a joint notion of security for asymmetric/symmetric encryption schemes which provably provides soundness for coinductive Dolev-Yao 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 would like to formalize what it means for a piece of data (nonce or symmetric-key element) to remain *secure* (i.e. unknown) in both the formal and computational settings. Under the formal approach, one would typically say that the secrecy of a piece of data is retained if it cannot be deduced by applying Dolev-Yao deduction rules. For a nonce element X, for instance, if X is not formally deducible, it means that 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 the Dolev-Yao adversary is not able to deduce a symmetric-key element, 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. As a notation, for $S = \{(s_1^1, s_1^2), (s_2^1, s_2^2), \ldots (s_n^1, s_n^2)\}$, we define $S^i \triangleq \{s_1^i, s_2^i, \ldots, s_n^i\}$ for $i \in \{1, 2\}$.

5.1 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 a pair of asymmetric/symmetric encryption schemes whose joint security is to be defined, w.l.o.g., with respect to the shared *security parameter* 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 will simply 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 think of τ as a mapping which is 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 $\mathcal{H}$. 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 public/secret key symbols $\{(k_i, k_i^{-1})\}_{1 \le i \le n} \in \mathcal{K}_H^{pub} \times \mathcal{K}_H^{privasym}$, symmetric-key symbols $\{k_i^{sym}\}_{1 \le i \le n} \in \mathcal{K}_H^{privsym}$, and nonce symbols $\{x_i\}_{1 \le i \le n} \in \mathcal{X}_H$, 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 $1 \leq i \leq n$. We initialize $eval\text{-}exp = \emptyset$. 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 any number of queries of the following types:

1. *Corruption*: $\mathcal{A}$ may corrupt a $\mathcal{B}$'s key by issuing $corrupt(s)$, 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 $eval\text{-}exp$.
2. *Encryption*: $\mathcal{A}$ may issue a query $encrypt(e, x)$, 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 $eval\text{-}exp$. Here e may contain both the challenger's and adversary's basic symbols.
3. *Nonce revelation*: $\mathcal{A}$ may issue a query $reveal(x_i)$, and in response receives nc_i, and (x_i, nc_i) is added to $eval\text{-}exp$.
4. *Decryption*: $\mathcal{A}$ may issue $decrypt(c, s')$, 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 $(\{e\}_{k_p}, c_p) \in eval\text{-}exp$ 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 $closure_c(eval\text{-}exp^1)$, 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 $challenge(s)$, where $s \in \{x_1, \ldots, x_n\}$ (nonce challenge), or of the form $challenge(s, bs)$, 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 $s \in closure_c(eval\text{-}exp^1)$, 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 $Enc_{sym}(bs, ck_j)$ 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 random variable corresponding to $\mathcal{A}$'s output when the secret bit is b, his *CI-advantage* is (below $\mathcal{E}_p$ refers to the pair of schemes):

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

Definition 2. *A pair of encryption schemes $\mathcal{E}_p = (\mathcal{E}_{sym}, \mathcal{E}_{asy})$ provides joint security under the CI game (shortly, is CI-secure) if for every adversary $\mathcal{A}$, $Adv_{\mathcal{E}_p,\mathcal{A}}^{CI}(\eta)$ 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 as follows: 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 [24] 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 that $\mathcal{A}$ 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 are trying to prove generic implication results from *standard* notions of encryption security, we have to restrict the set of keys for which we want to prove computational secrecy results (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 that $\mathcal{A}$ challenges in the **guessing** phase may have previously participated 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 additionally 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 the following conditions hold: (1) $Dec_{sym}(c, ck_i) \neq \perp$, and (2) there does not exist $(\{e\}_{k_j}, c') \in eval\text{-}exp$ such that $\{e\}_{k_j}$ has a subexp $\{e'\}_{k_i}$ encrypted in $\{e\}_{k_j}$ only under keys whose decryption keys are in $closure_c(eval\text{-}exp^1)$, 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 [33], which was defined for the passive setting.

5.2 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 $eval\text{-}exp^1$. To this end, he outputs (e, E), where e is an expression (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. $e \notin closure_c(eval\text{-}exp^1)$; 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. The adversary's *CNM-advantage* is:

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

Definition 3. *A pair $\mathcal{E}_p = (\mathcal{E}_{asy}, \mathcal{E}_{sym})$ provides joint security under the CNM game (shortly, is CNM-secure) if for every adversary $\mathcal{A}$, $Adv_{\mathcal{E}_p,\eta}^{CNM}(\mathcal{A})$ 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 $T = closure_c(eval\text{-}exp^1)$ 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 [42]. We give a full proof in Appendix A. $\square$

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 third occurrence of k_p^{sym} is $\{k_4, k_5\}$. We call a node 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. For $v_i \in V_\mathcal{A}$ we define $indeg(v_i)$ to be the maximum a for which we have an incoming edge with label a to v_i; this specifies the number of times its associated key 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 encryption) without key 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 by the adversary. 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 $2 \leq i \leq p$, $v_{y_{i-1}} \to^{a_i} v_{y_i} \in E_\mathcal{A}$, 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 a single node *coinductively continuable* if its associated path of length zero is so.

Lemma 1. *At any point, any irrecoverable node in $G(\mathcal{A})$ 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 Appendix B. $\square$

Definition 4. *We say that $\mathcal{E}_p = (\mathcal{E}_{asy}, \mathcal{E}_{sym})$ provides l-CI security if $Adv^{CI}_{\mathcal{E},\mathcal{A}}(\eta)$ 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. We will give a more detailed outline of the proof in Appendix C.1, and a full proof in Appendix B. $\square$

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 that 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 that 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 Appendix D. $\square$

6 Applications of Soundness

In this section we show how our computational soundness theorem can be used to reason about security of protocols, and exemplify the approach with a representative protocol. We first begin with some background.

A large class of security requirements (e.g. *authentication*) can be expressed in terms of *trace properties*, i.e., predicates over individual execution traces. To formalize this class of security requirements, we first provide some standard definitions from [42]. We let $Ftrace$ denote the set of all symbolic traces and $Ctrace$ denote the set of all computational traces. A formal (trace-based) security property, $\mathcal{P}_f$, is formalized by an associated set $S(\mathcal{P}_f) \subseteq Ftrace$, and we say that protocol Π is *coinductively secure* with respect to $\mathcal{P}_f$ (or *coinductively achieves* $\mathcal{P}_f$) if for all coinductively legitimate adversaries $\mathcal{A}_{\mathbf{F}}$ it holds that $\mathcal{FT}(\mathcal{A}_{\mathbf{F}}, \Pi) \in S(\mathcal{P}_f)$. A computational (trace-based) security property, $\mathcal{P}_c$, is formalized again by an associated set $S(\mathcal{P}_c) \subseteq Ctrace$, and we say $\Pi_{\mathcal{E}_p}$ computationally achieves $\mathcal{P}_c$ if for all PPT adversaries $\mathcal{A}_c$ it holds that: (Here $\Pi_{\mathcal{E}_p}$ represents the computational implementation of Π under the pair of encryption schemes $\mathcal{E}_p$.)

$$\Pr_{\mathcal{R}_A, \mathcal{R}_H}\left[\mathcal{CT}(\mathcal{A}_c, \mathcal{R}_A, \mathcal{R}_H, \Pi_{\mathcal{E}_p}) \in S(\mathcal{P}_c)\right] \geq 1 - negl(\eta).$$

For $\mathcal{P}_f$ and $\mathcal{P}_c$ we say that $\mathcal{P}_c$ *computationally realizes* $\mathcal{P}_f$ if for all $FT \in S(\mathcal{P}_f)$ and $CT \in Ctrace$, if $FT \prec CT$, then $CT \in S(\mathcal{P}_c)$.

We provide some more definitions. For formal adversary $\mathcal{A}_{\mathbf{F}} = \langle (fq_1, fr_1), \ldots, (fq_n, fr_n) \rangle$ and protocol Π we let $Sym(\mathcal{A}_{\mathbf{F}})$ denote the set of honest symmetric-key symbols occurring in $\{fr_1, \ldots, fr_n\}$. (We refer to any symbol in $\mathcal{K}_H^{privsym}$ as an honest symmetric-key symbol.) We define $IrrecSym(\mathcal{A}_{\mathbf{F}}) \stackrel{\triangle}{=} Sym(\mathcal{A}_{\mathbf{F}}) - closure_c(fr_1, \ldots, fr_n)$, and, using fr to denote $(fr_1, \ldots, fr_n)$, we define $Depth(\Pi, \mathcal{A}_{\mathbf{F}})$ to be the maximum d for which we have an encryption chain $k_{i_1}^{sym} \xrightarrow{fr} k_{i_2}^{sym} \xrightarrow{fr} \ldots \xrightarrow{fr} k_{i_{d+1}}^{sym}$ for $\{k_{i_1}^{sym}, \ldots, k_{i_d}^{sym}\} \subseteq IrrecSym(\mathcal{A}_{\mathbf{F}})$. We call any encryption chain satisfying the above property an *irrecoverable chain*. Finally we say that Π

is *constantly depth-bounded* (*bounded*, for short) if there exists a constant d such that for all coinductive adversaries $\mathcal{A}_{\mathbf{F}}$, it holds that $Depth(\Pi, \mathcal{A}_{\mathbf{F}}) \leq d$.

From Theorem 3 and by slightly modifying the proof of Theorem 1, we can arrive at the following Theorem:

Theorem 4. *Suppose $\mathcal{E}_p = (\mathcal{E}_{asy}, \mathcal{E}_{sym})$, where $\mathcal{E}_{asy}$ is IND-CCA2 secure, and $\mathcal{E}_{sym}$ is both IND-CCA2 and INT-CTXT secure. If Π is a bounded protocol which coinductively achieves $\mathcal{P}_f$ and $\mathcal{P}_c$ is the computational realization of $\mathcal{P}_p$, then $\Pi_{\mathcal{E}_p}$ computationally achieves $\mathcal{P}_c$.*

Proof. First by slightly changing the proof of Theorem 1 one can show that (a) if Π is l-depth bounded, then any l-**CNM** secure $\mathcal{E}_p$ provides soundness for Π, and (b) l-**ACI** security $\Rightarrow$ l-**CNM** security. Now from (a), (b) and Theorem 3, the result follows. $\square$

We now provide some general statements about bounded protocols. We first define two more types of security protocols. The class of protocols described in Section (3) are two-party protocols, involving both asymmetric and symmetric encryptions. We call this class of protocols *hybrid* two-party protocols. Another class of two-party protocols, which we call *symmetric two-party* protocols, consists of protocols which use symmetric encryption only, requiring that the two participating parties have previously established a secret key between themselves. Beyond two-party protocols, another widely-used type of protocol (especially) for authentication and key-exchange protocols is the *three-party* one, where two parties, I, R, with the aid of a trusted, incorruptible server, S, with whom each I and R shares a *long-term secret key*, attempt to accomplish a security task. We also call these protocols *server-based* protocols. (We use both terms "server-based" and "three-party" interchangeably.) Analogously to hybrid two-party protocols, one may define *valid* symmetric two-party and three-party protocols. Below, we give the definition for the three-party case; the definition for the symmetric two-party case is similar. Assuming $IDs = \{I, R, S\}$, $nonces = \{X_1, X_2, \dots\}$, $shared\text{-}keys = \{K_{IS}, K_{RS}\}$, and $sym\text{-}keys = \{K_1, K_2, \dots\}$, a server-based protocol is $\Pi = (\langle M_1, P_1 \to Q_1\rangle, \langle M_2, P_2 \to Q_2\rangle, \dots)$, where M_i's, again, are constructed from the above sets using concatenation and symmetric encryption, and $\{P_1, Q_1, P_2, Q_2, \dots\} \subseteq IDs$. Here, $\{K_{IS}, K_{RS}\}$ represent the two long-term secret keys that I and R share with S, and $\langle M_i, P_i \to Q_i\rangle$ states that P_i sends Q_i message M_i. We now define a *valid* server-based protocol to be the one in which K_{IS}, K_{RS} are never sent as plaintexts, and that every message received by each party is completely decipherable by the party. To formalize the latter requirement, we first define $K^{(S)} = \{K_{IS}, K_{RS}\}$, $K^{(I)} = \{K_{IS}\}$, and $K^{(R)} = \{K_{RS}\}$; now for Π introduced above and all $i \geq 1$, if M_i has a subexpression $\{M\}_K$, then K must be *inductively* recoverable from the expression formed by concatenating all elements from the following set:

$$K^{(Q_i)} \cup \{M_j \ : \ j \leq i \wedge Q_j = Q_i\}.$$

Similarly to hybrid protocols, one may also define *boundedness* for three-party and symmetric two-party protocols, and restate a theorem similar to Theorem 4 for them. (We note, however, that since for these two latter types of protocols we assume the only primitive used is symmetric encryption, the new theorems will only speak about an IND-CCA2+INT-CTXT-secure symmetric encryption scheme.)

6.1 Boundedness for the Wide Mouthed-Frog Authentication Protocol

The *Wide Mouthed-Frog* authentication protocol (WMF protocol for short) [18] is a three-party protocol described as follows:

$$\Pi_{wmf} = (\langle M_1, I \to S\rangle, \langle M_2, S \to R\rangle)$$

$$M_1 = (I, \{T_I, R, k_1^{sym}\}_{K_{IS}})$$
$$M_2 = \{T_S, I, k_1^{sym}\}_{K_{RS}}$$

Here, T_I and T_s denote *timestamps* generated, respectively, by I and R to ensure freshness of messages. (Although our symbolic protocol model does not consider timestamps, this does not affect what we are going to present below.)

Now suppose the WMF protocol is succeeded by I sending 0 (or any other fixed message) encrypted under k_1^{sym} to R. (i.e. $(\{0\}_{k_1^{sym}}, I \to R)$); we call this new extension the *extended WMF* (*EWMF* for short) protocol. For the EWMF protocol, as pointed out in [7], the so-called *used-order* property (which is a necessary assumption for most simulation-based soundness settings [7,36]) may be violated. The used-order property states that if a symmetric key k_i is used for encryption for the first time at some point T (and is unknown to the adversary at this point), *any* subsequent occurrence of k_i as a plaintext should *only* be encrypted under keys which were first used for encryption before T. For the EWMF protocol the used-order property may be violated as follows: (1) I sends S the message $(I, \{T_I, R, k_1^{sym}\}_{K_{IS}})$ and before S sending R the message $\{T_S, I, k_1^{sym}\}_{K_{RS}}$, I sends $\{0\}_{k_1^{sym}}$ to R, violating the used-order property.

We now show that the EWMF protocol is a bounded protocol, satisfying our soundness restriction. We should mention here that we are not claiming that the WMF protocol is secure (as it is known to be not); rather we want to provide some general statements to show how reasoning about our soundness restriction can be made, and to discuss its applicability.

We require the following definitions. We call a three-party protocol Π *type-1* if K_{IS} and K_{RS} are the only keys used for encryption in Π. (For example, the WMF protocol is type-1.) We now show that every type-1 three party protocol is bounded, and that if any type-1 three party protocol Π_1 is run concurrently with a bounded, symmetric two-party protocol Π_2, then the boundedness property is retained for their concurrent execution. (i.e. the length of the longest irrecoverable chain that may be created when Π_1 and Π_2 are executed concurrently is at most longer by a *constant* than the longest irrecoverable chain that may be produced when any of Π_1 and Π_2 is run individually.) Denoting by $\Pi_1||\Pi_2$ the system that corresponds to the concurrent execution of Π_1 and Π_2 and by $u_1, \ldots, u_n$ the set of users, the setup model of concurrent execution assumes the existence of a trusted server S who shares a unique long-term secret key with each user (this is the setup inherited from Π_1), and further requires that any u_i and u_j who want to run an instance of Π_2 together, should have already set up a secret key between themselves. (This secret key can be provided by, possibly, a run of Π_1, or the setup model may additionally assume that every two users also have a *private communication* link between themselves. Our proposition below holds under any of these two latter assumptions.)

Proposition 1. *If Π_1 is a type-1 three party protocol, and Π_2 is a symmetric two-party protocol, then*

1. *for any formal adversary $\mathcal{A}_F$, it holds $Depth(\Pi_1, \mathcal{A}_F) \leq 1$, and*
2. *for any formal adversary $\mathcal{A}_F$, it holds that $Depth(\Pi_1||\Pi_2, \mathcal{A}_F) \leq Depth(\Pi_2) + 1$.*

Proof. **Proof of (1):** For any set of users $\{u_1, \ldots, u_n\}$ of Π_1 and any formal adversary $\mathcal{A}_F$, we show that the length of the longest irrecoverable chain is at most one. For $1 \leq i \leq n$ we denote u_i's shared long-term secret key with s, the server, by k_{u_is}. Assume $\mathcal{A}_F$'s computation produces the following sequence of queries/replies: $\mathcal{A}_F = \langle (q_1, r_1), \ldots, (q_m, r_m) \rangle$, and, letting r denote $(r_1, \ldots, r_m)$, assume, to the contrary, that $k_1 \xrightarrow{r} k_2 \xrightarrow{r} k_3$, for some $k_1, k_2, k_3 \notin closure_c(r)$. We define $K_u = \{k_{u_is} : 1 \leq i \leq n\}$ and $K_u^{irr} = K_u - closure_c(r)$. First note that since $\mathcal{A}_F$ is coinductively legitimate, for every $1 \leq i \leq n$, we have $q_i \in closure_c(r_1, \ldots, r_{i-1})$. Now we show that (a) $k_1 \xrightarrow{r} k_2$ and $k_1, k_2 \notin closure_c(r)$ imply that $k_1 \in K_u^{irr}$ (i.e. k_1 cannot be a symmetric key outside K_u^{irr}; for example it is not a fresh symmetric key that may have been generated during the protocol execution.), (b) similarly to (a), $k_2 \xrightarrow{r} k_3$ and $k_2, k_3 \notin closure_c(r)$ imply that $k_2 \in K_u^{irr}$, and (c) For no $k_i, k_j \in K_u^{irr}$ it holds that $k_i \xrightarrow{r} k_j$. Note that (a), (b), and (c) conclude the proof of this part of the proposition. The proofs of (a) and (b) are entirely similar, and the proof for (c) follows using the same ideas used for (a) and (b); so we only show (a). Since $k_1 \xrightarrow{r} k_2$, for some $1 \leq w \leq m$, we have $k_1 \xrightarrow{r_w} k_2$, implying that either $q_w = new\text{-}session(i, j)$, or $q_w = send(sn_w, e_w, R_w)$, for $R_w \in \{I, R, S\}$. Now when $\mathcal{A}_F$ makes query $q_w = new\text{-}session(i, j)$, if u_i is already corrupted, then it must be that all encryption keys used to form r_w are already known by the adversary, contradicting the fact that $k_1, k_2 \notin closure_c(r)$. On the other hand, if u_i is not corrupted, then u_i follows the specification of the protocol, implying that the *only* encryption key used in r_w must be $k_{u_i s}$ (this follows from the fact that Π_1 is type-1 and u_i is honest), implying that $k_1 \in K_u^{irr}$. With the same reasoning one can also conclude that for the case $q_w = send(sn_w, e_w, R_w)$ it should also hold that all encryptions used in r_w are under a key which is in K_u^{irr}. This completes the proof of part (1).

Proof of (2): The proof of this part also uses the same ideas discussed above, so we just give a proof outline. Suppose $\mathcal{A}_F = \langle (q_1, r_1), \ldots, (q_m, r_m) \rangle$ (Here each q_i is a query made to either Π_1 or Π_2.) To prove this part, one needs to first show that no key in K_u^{irr} appear as a plaintext in any r_i's (or otherwise, $\mathcal{A}_F$ is not coinductively valid); and then deduce that for any irrecoverable chain $P = k_{i_1} \xrightarrow{r} k_{i_2} \xrightarrow{r} \ldots \xrightarrow{r} k_{i_p}$, either P or $k_{i_2} \xrightarrow{r} \ldots \xrightarrow{r} k_{i_p}$ can be created by an adversary $\mathcal{A}'_F$ who is only run in an environment with only Π_2 executing.The proof then follows. $\square$

Now from the above proposition it follows that the the depth of the EWMF protocol is one. This is because, every execution under the EWMF protocol may already be created under a concurrent execution of the WMF protocol (which is a type-1 three party protocol) and the simple symmetric two party protocol $\Pi_2 = (\{0\}_{K_{IR}}, I \to R)$.

7 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 called *boundedness* (Appendix 6), which requires that no symbolic execution of the underlying protocol produce a coinductively-irrecoverable encryption chain of nonconstant length. We observed that almost all protocols from [25] (when run in *isolation*) admit (at most) 2-boundedness. (All of them are bounded.) We also provided 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* [29] in the case of secret keys).

For simplicity, we have assumed that if a user is corrupted, the adversary receives *only* her long-lived key and her past generated secret keys/nonces, but *not* her past *random coins.* In Appendix E we give some 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 realization 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 in the introduction, defining the extension in an entirely left-or-right indistinguishability sense, as in [8], would entail inherent limitations; for example, if a left-or-right encryption query is made under ck, then ck may not be corrupted afterward.

Finally it would be interesting to improve the bounds imposed by our soundness theorem (and those of [46]), and investigate its extension to more general cryptographic frameworks supporting *compositional reasoning* [9,23].

References

1. Martín Abadi and Andrew D. Gordon. A calculus for cryptographic protocols: the spi calculus. In *Proceedings of the 4th ACM conference on Computer and communications security*, CCS '97, pages 36–47, New York, NY, USA, 1997. ACM. 1
2. Martín Abadi and Jan Jürjens. Formal eavesdropping and its computational interpretation. In Naoki Kobayashi and Benjamin C. Pierce, editors, *TACS*, volume 2215 of *Lecture Notes in Computer Science*, pages 82–94. Springer, 2001. 1
3. 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. 1
4. Tolga Acar, Mira Belenkiy, Mihir Bellare, and David Cash. Cryptographic agility and its relation to circular encryption. In Gilbert [31], pages 403–422. 1
5. Michael Backes. Real-or-random key secrecy of the otway-rees protocol via a symbolic security proof. *Electr. Notes Theor. Comput. Sci.*, 155:111–145, 2006. 1
6. 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. 1
7. Michael Backes and Birgit Pfitzmann. Symmetric encryption in a simulatable dolev-yao style cryptographic library. In *CSFW*, pages 204–218. IEEE Computer Society, 2004. 1, 4, 6.1
8. 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. 1, 7
9. 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. 1, 4, 7
10. 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. 1
11. 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. 1
12. 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. 1
13. 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. 1
14. Mihir Bellare and Chanathip Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. *J. Cryptology*, 21(4):469–491, 2008. 1

15. 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. 1
16. 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. 1
17. 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. 1
18. Michael Burrows, Martin Abadi, and Roger Needham. A logic of authentication. *ACM Trans. Comput. Syst.*, 8:18–36, February 1990. 1, 6.1
19. 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. 1
20. Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *FOCS*, pages 136–145. IEEE Computer Society, 2001. 1
21. Ran Canetti, Cynthia Dwork, Moni Naor, and Rafail Ostrovsky. Deniable encryption. In Burton S. Kaliski Jr., editor, *CRYPTO*, volume 1294 of *Lecture Notes in Computer Science*, pages 90–104. Springer, 1997. 1
22. 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. 1
23. 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. 1, 7
24. 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. 1, 5.1
25. John Clark and Jeremy Jacob. A survey of authentication protocol literature. Technical report, 1997. 1, 7
26. 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. 1
27. Véronique Cortier and Bogdan Warinschi. Computationally sound, automated proofs for security protocols. In Shmuel Sagiv, editor, *ESOP*, volume 3444 of *Lecture Notes in Computer Science*, pages 157–171. Springer, 2005. 1
28. Véronique Cortier and Eugen Zalinescu. Deciding key cycles for security protocols. In Miki Hermann and Andrei Voronkov, editors, *LPAR*, volume 4246 of *Lecture Notes in Computer Science*, pages 317–331. Springer, 2006. 4
29. 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. 1, 7
30. 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. 1
31. 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. 4, 39
32. Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984. 1
33. Jonathan Herzog. A computational interpretation of dolev-yao adversaries. *Theor. Comput. Sci.*, 340(1):57–81, 2005. 1, 5.1
34. Russell Impagliazzo and Bruce M. Kapron. Logics for reasoning about cryptographic constructions. *J. Comput. Syst. Sci.*, 72(2):286–320, 2006. 1
35. Steve Kremer, Graham Steel, and Bogdan Warinschi. Security for key management interfaces. In *CSF*, pages 266–280. IEEE Computer Society, 2011. 1
36. 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. 1, 6.1
37. 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. 1
38. Laurent Mazaré and Bogdan Warinschi. Separating trace mapping and reactive simulatability soundness: The case of adaptive corruption. In Pierpaolo Degano and Luca Viganò, editors, *ARSPA-WITS*, volume 5511 of *Lecture Notes in Computer Science*, pages 193–210. Springer, 2009. 1
39. Daniele Micciancio. Computational soundness, co-induction, and encryption cycles. In Gilbert [31], pages 362–380. 1, 2, 2
40. 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. 1

41. Daniele Micciancio and Saurabh Panjwani. Corrupting one vs. corrupting many: The case of broadcast and multicast encryption. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *ICALP (2)*, volume 4052 of *Lecture Notes in Computer Science*, pages 70–82. Springer, 2006. 1
42. 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. 1, 3, 5.2, 6, A
43. John C. Mitchell, Mark Mitchell, and Ulrich Stern. Automated analysis of cryptographic protocols using mur-phi. In *IEEE Symposium on Security and Privacy*, pages 141–151. IEEE Computer Society, 1997. 1
44. 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. 1
45. Jesper Buus Nielsen. Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In Moti Yung, editor, *CRYPTO*, volume 2442 of *Lecture Notes in Computer Science*, pages 111–126. Springer, 2002. 1
46. 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. 1, 7, C.1, C.1, C.5
47. 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. 1
48. 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. 1
49. Arnab Roy, Anupam Datta, Ante Derek, and John C. Mitchell. Inductive trace properties for computational security. *Journal of Computer Security*, 18(6):1035–1073, 2010. 1
50. Bogdan Warinschi. A computational analysis of the needham-schröeder-(lowe) protocol. In *CSFW*, pages 248–. IEEE Computer Society, 2003. 1

A Proof of Theorem 1

Proof of Theorem 1 (Part 1): Suppose $\mathcal{A}_{cs}$ is an adversary attacking $\mathcal{E}_p = (\mathcal{E}_{asy}, \mathcal{E}_{sym})$ in the sense of "soundness" and winning with non-negligible probability; namely, for whom there exists a protocol Π such that Definition 1 does not hold. From $\mathcal{A}_{cs}$ we construct $\mathcal{A}_{cnm}$, who attacks the scheme in the sense of *CNM* and wins, again, with non-negligible probability.

First we make an assumption about our protocol execution models. We assume that in reply to a query $\textit{new-session}(i,j)$, in which party u_i is already corrupted, *only* a session (with its uniquely assigned number) is opened between u_i and u_j, but no message is given to the adversary. This is without loss of generality since u_i is already corrupted and the adversary can generate the opening message by himself. Similarly we assume that no reply is given to a query $\textit{send}(sn, m, X)$, for $X \in \{I, R\}$, if the user taking role X in session sn is already corrupted.

Our proof proceeds in two parts. The first part of the proof is similar to that of [42]. Namely, for every $\mathcal{A}_{cs}$ and associated random coins $\mathcal{R}_A$ and $\mathcal{R}_H$, if $CT \stackrel{\triangle}{=} \mathcal{CT}(\mathcal{A}_{cs}, \mathcal{R}_A, \mathcal{R}_H, \Pi_{\mathcal{E}_p}) = \langle (cq_1, cr_1), (cq_2, cr_2), \ldots \rangle$ (recall the corresponding notation from Section 3) we show that there *exists* a symbolic adversary $\mathcal{A}_f$ and a computational mapping τ such that $FT \stackrel{\triangle}{=} \mathcal{FT}(\mathcal{A}_f) = \langle (fq_1, fr_1), (fq_2, fr_2), \ldots \rangle$ is mapped to CT under τ. We should remark that the constructed $\mathcal{A}_f$ may or may not be coinductively-valid; for this part we just want to show the *existence* of $\mathcal{A}_f$ and τ, and for this we assume that we are already given $\mathcal{R}_A$ and $\mathcal{R}_H$.

In the second part of the proof, we show that if the constructed $\mathcal{A}_f$ is indeed coinductively invalid with a non-negligible probability (note that if $\mathcal{A}_f$ is coinductively invalid only with a negligible probability then $\mathcal{A}_{cs}$ is "soundness-respecting"), then from $\mathcal{A}_{cs}$ we can build $\mathcal{A}_{cnm}$ in such a way that he first guesses s, the point in which soundness is violated (i.e. the first point in which $fq_s \notin closure_c(fr_1, \ldots, fr_{s-1})$), and for every query cq_j (for $j < s$) of $\mathcal{A}_{cs}$, $\mathcal{A}_{cnm}$ properly constructs cr_j by first building fq_j, then fr_j and finally constructing cr_j from fr_j with the aid of his CNM-*encryption* oracle (and his *decryption* oracle). Finally when $\mathcal{A}_{cs}$ outputs cq_s, $\mathcal{A}_{cnm}$

properly builds fq_s (the underlying expression) and outputs (fq_s, cq_s). We provide the details below.

First Part. We show how to construct $\mathcal{A}_f$ from $\mathcal{A}_{cs}$ under $\mathcal{R}_A$ and $\mathcal{R}_H$. The construction is summarized as follows: for every cq_i we first fully decrypt cq_i (using $\mathcal{R}_A$ and $\mathcal{R}_H$ and previously sent messages); accordingly update $F(.,.,.)$; from that we update $f(.,.,.)$; and finally from the updated f we construct fq_i. (Recall that $F(.,.,.)$ and $f(.,.,.)$ are the components of, respectively, a computational and a formal protocol state.) Once fq_i is built, fr_i is obtained uniquely. Now we describe the details.

First, for every $cq_i = corrupt(\cdot)$, we will have $fq_i = cq_i$. Next for $cq_i = new\text{-}session(j, k)$, we will have $fq_i = cq_i$, and if u_j is already corrupted then no reply should be given to the adversary (as described above); otherwise, cr_i will only have basic values generated by an honest party which can uniquely be determined by $\mathcal{R}_H$, and so we can update F accordingly. After F is updated and assuming that the new session is given number sn, if, for $Q \in BS(\Pi)$, $F(I, Q, sn)$ is updated to value c, we also update $f(I, Q, sn)$ to a fresh honest party's symbol s_H (of the appropriate type) and set $\tau(s_H) = c$. For the final case, if $cq_i = send(sn_i, cm_i, X)$, where $X \in \{I, R\}$, we will set $fq_i = send(sn_i, fm_i, X)$, where fm_i is an expression constructed by first parsing cm_i according to the syntax given in the protocol (note that here cm_i may have injected adversary values as well, but since we assumed that all encrypted parts of a party's received message should be decipherable using the party's current knowledge, cm_i has to be fully and *uniquely* decipherable using the partially constructed $F(\cdot,\cdot,\cdot)$ and cm_i itself), and then updating F accordingly. Now if, for $Q \in BS(\Pi)$, $F(X, Q, sn)$ is updated to value c, c may possibly be an adversarially-generated value, and, hence, to update $f(X, Q, sn)$ we proceed as follows: if for some s we already have $\tau(s) = c$ we set $f(X, Q, sn) = s$, otherwise, we introduce a fresh adversary symbol q_A (of the proper type) and set $f(X, Q, sn) = q_A$, $\tau(q_A) = c$. (Note that if c is not in the range of τ, this means that c is not a basic value of any reply the adversary has received from previous *new-session* and *send* queries, implying that c should be a value that the adversary has injected in.) From the updated f we construct fm_i, fq_i, and fr_i. Now it can easily be verified that CT is a computational interpretation of FT under τ.

Second Part. Now suppose the probability that $\mathcal{A}_f$ formed above is coinductively invalid is non-negligible; namely, there exists a *smallest* s (which we assume is a priori known as it can be guessed with a non-negligible probability) where $fq_s \notin closure_c(fr_1, \ldots, fr_{s-1})$[2]. Now $\mathcal{A}_{cnm}$ is going to simulate $\mathcal{A}_{cs}$ as follows: for every query cq_i of $\mathcal{A}_{cs}$, $\mathcal{A}_{cnm}$ is going to construct its symbolic query fq_i (possibly with the aid of his decryption oracle), compute fr_i the symbolic reply to fq_i (note that this is possible given fq_i, the protocol specification, and the past symbolic execution), call his oracles to receive the computational evaluation of fr_i, and return it to $\mathcal{A}_{cs}$. Now when $\mathcal{A}_{cs}$ outputs $cq_s = send(sn_s, cm_s, i_s)$ (note that cq_s has to be in this form if it's the first query invalidating the coinductive closure assumption), $\mathcal{A}_{cnm}$ constructs fm_s the symbolic counterpart of cm_s, and outputs (fm_s, cm_s) in his CNM game. More details come below.

First some simplifying assumptions. We represent any CNM adversary's *corruption* and *encryption* queries using a single type of query $Eval(e)$, which requests the computational interpretation of e. For example, $corrupt(k_i^{-1})$ is like asking $Eval(k_i^{-1})$, or $encrypt(e, k_j)$ is like $Eval(\{e\}_{k_j})$. Also, we let the input of $Eval$ be any arbitrary expression. Although the *encryption* query in CNM does not (explicitly) give us the computational image of, say, a paired expression $e = (e_1, e_2)$, any adversary can manually do this by completely un-pairing e, requesting the computational images of all produced encrypted expressions, and successively pairing them all back.

[2] Here we are actually abusing notation since fq_s is a query not an expression. We define $fq_s \in closure_c(fr_1, \ldots, fr_{s-1})$ if fq_s is either a *new-session* or *corrupt* query, or $fq_s = send(sn_s, fm_s, X)$ and $fm_s \in closure_c(fr_1, \ldots, fr_{s-1})$.

Proceeding with the proof of the claim, we assume for $\mathcal{A}_{cs}$, without loss of generality, that the set of protocol users is $\{u_1, \ldots, u_n\}$, where $n = poly_1(\eta)$, and the number of to-be-created sessions is upper-bounded by $p = poly_2(\eta)$, where the first created session is assigned number 1, the second 2, and so on. To construct $\mathcal{A}_{cnm}$, first we explain how the **setup** phase of the CNM game is initialized (i.e. what challenger's basic symbols are): the set S of challenger's basic symbols contains (k_i, k_i^{-1}), for every $1 \le i \le n$, where (k_i, k_i^{-1}) is going to serve as a symbolic counterpart of u_i's pair of long-lived public/private keys. Also for every $1 \le i \le n$ and $1 \le sn \le p$, S contains $S_{i,sn} = \{x_{i,sn,1}, \ldots, k^{sym}_{i,sn,1}, \ldots\}$, whose elements are going to serve as symbolic counterparts of nonces and symmetric keys that u_i is going to use in session sn, if she becomes a participant in that session. (Since this latter is a priori unknown, we designated a larger set of challenger's basic symbols at the beginning.) Now we explain below how $\mathcal{A}_{cnm}$ simulates $\mathcal{A}_{cs}$, as informally sketched above:

- For a query $cq_i = new\text{-}session(i, j)$ of $\mathcal{A}_{cs}$, we set $fq_i = cq_i$, and if this newly opened session receives number sn, $\mathcal{A}_{cnm}$ asks his oracle the query $Eval(fr_i)$ (and gives the result as cr_i to $\mathcal{A}_{cs}$), where fr_i is built from $S_{i,sn}$, following the "syntax" specified in the protocol.
- For $cq_i = corrupt(i)$, we again set $fq_i = cq_i$ and $\mathcal{A}_{cnm}$ will reply to cq_i by issuing a sequence of *Eval* queries. (We have a sequence because, as we mentioned earlier, when the adversary corrupts a party during the protocol execution, the adversary receives the whole internal state of that party, including, especially, all symmetric keys/nonces that party has generated in each session. So here, fr_i will be an expression consisting of all of the parties basic symbols generated so far, and cr_i, the reply to cq_i will be the computational interpretation of fr_i.)
- For $cq_i = send(sn_i, cm_i, R)$, first note that cm_i may have $\mathcal{A}_{cs}$'s injected values as well. In order for $\mathcal{A}_{cnm}$ to be able to build fm_i (and consequently fq_i), $\mathcal{A}_{cnm}$ has to be able to (i) obtain all the basic values that were injected by $\mathcal{A}_{cs}$ (and for every such value $\mathcal{A}_{cnm}$ will introduce, if he has not already introduced, a new basic symbol along with the extracted value as the computational image of the symbol in his CNM game), and (ii) for every other constitutuent basic value of cm_i, $\mathcal{A}_{cnm}$ has to be able to determine the basic symbol corresponding to that value. (For example, if $\mathcal{A}_{cnm}$ encounters a ciphertext c that he knows is encrypted under public key pk_i, but whose private key is unknown, if he has already obtained c under an *encrypt* query $encrypt(e, k_i)$, he knows the expression corresponding to c and can use this to build up fm_i accordingly.)
  First we show (i); namely, how $\mathcal{A}_{cnm}$ extracts all $\mathcal{A}_{cs}$'s injected values (i.e. those whose symbolic counterparts would be adversarial symbols in the process of constructing $\mathcal{A}_f$; see Part (1) of the proof). For this assume that $\mathcal{A}_{cnm}$ has extracted all $\mathcal{A}_{cs}$'s inserted values up to this query; now to acquire all $\mathcal{A}_{cs}$'s newly inserted values (symmetric keys or nonces) in cm_i, since by assumption $fm_i \in closure_c(fr_1, \ldots, fr_{i-1})$—Note that $\mathcal{A}_{cnm}$ still does not know fm_i; this latter is a condition which will hold if $\mathcal{A}_{cnm}$'s guessing was correct— it must be the case that any newly-inserted adversarial symbol s in fm_i is encrypted only under public keys or under keys whose decryption keys fall into the coinductive knowledge set of the adversary. Any other case implies that $fm_i \notin closure_c(fr_1, \ldots, fr_{i-1})$. As a result $\mathcal{A}_{cnm}$ can acquire c, the computational value of s in cm_i through a sequence of decryption operations, each of which is either performed under a key already known by $\mathcal{A}_{cnm}$ (in case the encountered encryption key is an adversarial key), or through calling the *decryption* oracle (in case the encountered encryption key is the public key of a user)[3]. Now for every

[3] We remark that here we are implicitly assuming that, under a given τ and two e_1 and e_2 with different underlying parse trees (here, for example we consider $\{0\}_{k_1}$ and $\{0\}_{k_2}$ to have different parse tress), we have $sup\,[[\![e_1]\!]_\tau] \cap sup\,[[\![e_2]\!]_\tau] = \emptyset$; this ensures that whenever the *decryption* oracle is called down along the path toward s, it will decrypt for $\mathcal{A}_{cnm}$. This assumption can easily be guaranteed by "tagging" each bitstring with its type (e.g. basic, pair, encryption), and moreover by having each party tag each of its produced ciphertexts

newly recovered value c injected by$\mathcal{A}_{cs}$, $\mathcal{A}_{cnm}$ will introduce a new basic symbol and assign its computational image to c in his CNM game. Next, we show how (ii) can be done. After all of $\mathcal{A}_{cs}$'s injected values are recovered (i.e. after (i) is done), $\mathcal{A}_{cnm}$ parses cm_i according to the syntax prescribed by the protocol, unpairs concatenated strings, and for every obtained ciphertext ct, if ct is encrypted under an adversarial key (note that $\mathcal{A}_{cnm}$ is able to tell if the obtained ciphertext was encrypted under an $\mathcal{A}_{cs}$'s injected key, since he already knows the position of all those keys in the expression), $\mathcal{A}_{cnm}$ already knows its decryption key and can decipher ct by himself. If, otherwise, ct is encrypted under an honest key (whose decryption key is not known to $\mathcal{A}_{cnm}$), $\mathcal{A}_{cnm}$ asks his oracle to decrypt ct (under the appropriate decryption key whose symbolic key can be determined by the syntax of the protocol); now if the oracle decrypts successfully, $\mathcal{A}_{cnm}$ continues the above process with the resulting string, otherwise, if the decryption fails, this means that ct is coinductively visible in a ciphertext he has obtained from his *Eval* queries and, hence, $\mathcal{A}_{cnm}$ already knows its corresponding symbolic expression.

Now finally when $\mathcal{A}_{cs}$ makes the query $cq_s = send(sn_s, cm_s, i_s)$, if all guesses made so far have been correct (which is the case with a non-negligible probability) and if $\mathcal{A}_{cnm}$ successfully manages to construct fm_s from cm_s, then (fm_s, cm_s) will allow $\mathcal{A}_{cnm}$ to win in his game. But how $\mathcal{A}_{cnm}$ can construct fm_s from cm_s? Here we present the simplest way (and arguably the most inefficient way) which will work with a non-negligible probability: since $\mathcal{A}_{cnm}$ knows the syntactic structure of cm_s (as provided in the protocol) and since fm_s will have at most a constant number of symbols, $\mathcal{A}_{cnm}$ can guess all constituent elements of fm_s and build it up this way. This completes the proof. $\square$

Proof of Theorem 1 (Part 2): Suppose $\mathcal{A}^{cnm}$ attacks $\mathcal{E}_p = (\mathcal{E}_{sym}, \mathcal{E}_{asy})$ under the CNM game and wins with a non-negligible probability; from $\mathcal{A}^{cnm}$ we construct $\mathcal{A}^{aci}$ who attacks the ACI security of $\mathcal{E}_p$. Without loss of generality, we assume that when $\mathcal{A}^{cnm}$ outputs (e, E), it holds that $e = \{e_1\}_k$, for some e_1, where $k^{-1} \notin closure_c(eval\text{-}exp^1)$. (This is without loss of generality because for every $k^{-1} \in closure_c(eval\text{-}exp^1)$, even if the adversary does not have the computational value of k^{-1}—for example if he has obtained a computational image of $\{k^{-1}\}_k$, which does not necessarily give out the underlying value of k^{-1}—he can obtain the computational value of k^{-1} by corrupting k^{-1}; note that this *corruption* does not change the symbolic knowledge of the adversary.) Now, $\mathcal{A}^{aci}$ simulates $\mathcal{A}^{cnm}$, answering $\mathcal{A}^{cnm}$'s oracle queries using his oracle, and waits for $\mathcal{A}^{cnm}$ to output his "guess". It is clear that the **interaction** phase of $\mathcal{A}^{cnm}$ can be perfectly simulated by $\mathcal{A}^{aci}$ since they both make exactly the same queries. Now when $\mathcal{A}^{cnm}$ outputs his guess $(\{e\}_{k_i}, E)$, since $\{e\}_{k_i} \notin closure_c(eval\text{-}exp^1)$, there exists $s \sqsubseteq \{e\}_{k_i}$, such that s is either a symmetric-key/nonce, or an encrypted expression $\{e_j\}_{k_j^{sym}}$ (where $k_j^{sym} \notin closure_c(eval\text{-}exp^1)$), and that the following conditions hold: (a) $s \notin closure_c(eval\text{-}exp^1)$, and (b) for any subexp $\{e'\}_{k'}$ of $\{e\}_{k_i}$ in which s occurs, $\{e'\}_{k'} \notin closure_c(eval\text{-}exp^1)$. Informally, Condition (b) ensures that $\mathcal{A}^{aci}$ can extract the corresponding computational value of s (from E)

by a label, in such a way that ciphertexts encrypted under the same key receive the same label. This, now, rules out the possibility that $\mathcal{A}_{cnm}$, at some point during his simulation, call his *encryption* oracle to obtain c, an image of $\{\{x\}_{k_1}\}_{k_2}$ (to give it to $\mathcal{A}_{ci}$), and later on $\mathcal{A}_{ci}$ inject c in a context where a value of $\{\{x_1\}_{k_3}\}_{k_2}$ is expected. (If this happens, the simulating $\mathcal{A}_{cnm}$ will then have to to call his *decryption* oracle to decrypt c successively under k_2 and k_3's decryption values to obtain the underlying value of x, which here will be an adversarial value.) This issue, which makes our simulation fail (because $\mathcal{A}_{cnm}$ is not permitted to make $decrypt(c, k_2^{-1})$), is resolved by the use of tagging described above. (i.e. c will be immediately rejected by the respective user if it is used in the context described above.) We note that this tagging assumption is common to almost all computational soundness results, and the reason is that symbolic models are typically assumed to be *free*, meaning that every expression has a unique parse tree. (i.e. Injecting $\{\{x\}_{k_1}\}_{k_2}$ in a context where something like $\{\{x_1\}_{k_3}\}_{k_2}$ is expected would produce an error.)

through successive decryption operations, down along the path leading to s, while the condition $s \notin closure_c(eval\text{-}exp^1)$ allows $\mathcal{A}^{aci}$ to attack the CI security of $\mathcal{E}_p$ in case s is a symmetric key or a nonce, and ciphertext integrity in case s is an encrypted expression. Below we formally show how $\mathcal{A}^{aci}$ will attack the CI security of $\mathcal{E}_p$ if s is a symmetric key or a nonce; the attack against ciphertext integrity in case that s is an encrypted expression follows similarly. At first, $\mathcal{A}^{aci}$ issues the *decryption* query $decrypt(E, k_i^{-1})$ to decrypt E; this works because $\{e\}_{k_i} \notin closure_c(eval\text{-}exp^1)$[4]. After receiving the decrypted string, $\mathcal{A}^{aci}$ proceeds to move toward s, separating emerging "pairs" as he goes down along the path, and if he encounters a ciphertext E_h being the computational image of, say, $\{e_h\}_{k_h}$, in case he already does not know the decryption-key value of k_h, he again invokes his *decryption* oracle to obtain $Dec(E_h, k_h^{-1})$, which according to item (b) above will be decrypted by the oracle because $\{e_h\}_{k_h} \notin closure_c(eval\text{-}exp^1)$. If at any point $\mathcal{A}^{aci}$ fails during this computation (e.g. receiving a plaintext from the *decryption* oracle which is not of expected format, meaning that the original "guess" of $\mathcal{A}^{cnm}$ was incorrect), he outputs a random guess and terminates. (In more detail, we assume $\mathcal{A}^{aci}$ has designated a special nonce symbol x of the challenger from the beginning—where x does not participate in any query—and here he simply chooses x as his "challenge symbol" and chooses his output bit uniformly at random.) Finally, if $\mathcal{A}^{aci}$ succeeded in obtaining the computational value of s, he can easily win in the game by choosing s as the challenge symbol; we omit the details. This completes the proof. □

B Proof of Lemma 1 and Theorem 2

B.1 Proof of Lemma 1

First note that if an asymmetric decryption key k_p^{-1} is coinductively irrecoverable (irrecoverable for short), then v_p^{asy} is, trivially, coinductively continuable (continuable for short) in the key graph (this is because $indeg(v_p^{asy}) = 0$). Thus we just need to show that if a symmetric decryption key, k_p^{sym}, is irrecoverable (i.e. $k_p^{sym} \notin closure_c(eval\text{-}exp^1)$), then v_p^{sym} is continuable.

We first give some definitions. For the key graph $G(\mathcal{A}) = (V, E)$, we let $V_{rec} \subseteq V$ denote the set of recoverable nodes of V, and V_{corr} denote the set of *corrupt nodes* of V. (A node is called corrupt if its associated decryption key has been input to a **corruption** query.) We define $prev\text{-}nodes(v_k, d_k)$ to be the set containing all v_j's where $v_j \xrightarrow{d_k} v_k \in E$. We call $S \subseteq V$ *coinductively closed* (closed for short) if for every $v_i \in S$, there exists d_i such that $prev\text{-}nodes(v_i, d_i) \subseteq S \cup V_{corr}$. If $S_1 \subseteq S_2$, we say that S_1 is closed under S_2, if if for every $v_i \in S_1$, there exists d_i such that $prev\text{-}nodes(v_i, d_i) \subseteq S_2 \cup V_{corr}$.

Next, we present some easily verifiable statements about the key graph.

1. If $v_i \in V_{rec}$ and $v_i \notin V_{corr}$, then it must be that $indeg(v_i) > 0$.
2. If for $v_i \in V$ and label d_i we have $prev\text{-}nodes(v_i, d_i) \subseteq V_{rec}$, then it must be that $v_i \in V_{rec}$.
3. If S is closed, then $S \subseteq V_{rec}$. The reason for this is that, if we denote by S_k the set of decryption keys associated to S and define $fs = (eval\text{-}exp^1)$, then we have $S_k \subseteq \mathcal{F}_{fs}(S_k)$, and hence by Equation (3), we have $S_k \subseteq FIX(\mathcal{F}_{fs})$.
4. The converse of Item (3): If $S \subseteq V_{rec}$ then there exists S_1 such that $S \subseteq S_1$ and that S_1 is closed. This follows from the basic coinductive properties described in Section 2.
5. If $v_i \notin V_{rec}$, then v_i is not the member of any closed set. This is a trivial implication of Item (3).
6. If $S_3 = S_1 \cup S_2$ and both S_1 and S_2 are closed under S_3, then S_3 is a closed set. (We call this property the *union property*.)

[4] It should be noted for this to hold, we require the tagging mechanism as explained above.

We now show that if v_p^{sym} is not coinductively continuable, then there exists a closed set S_p such that $v_p^{sym} \in S_p$, concluding from Item (3) that v_p^{sym} is recoverable. It suffices to prove this for the case $indeg(v_p^{sym}) > 0$ (because, otherwise, non-continuability of v_p^{sym} immediately implies that $v_p^{sym} \in V_{rec}$).

Statement: If $indeg(v_p^{sym}) > 0$ and v_p^{sym} is non-continuable, then there exists a closed set S_p such that $v_p^{sym} \in S_p$.

We prove the above statement by induction over the length of the longest path ending at v_p^{sym}, which is denoted by $lendpath(v_p^{sym})$.

Base case: If $lendpath(v_p^{sym}) = 1$, non-continuability of v_p^{sym} implies that there exists d_p such that for all $v_i \in \textit{prev-nodes}(v_p^{sym}, d_p)$, it holds that either $indeg(v_i) = 0$ and $v_i \in V_{corr}$, or $indeg(v_i) > 1$ and v_i has only incoming edges from v_p^{sym}. Thus if we define $S_p = \{v_p^{sym}\} \cup \textit{prev-nodes}(v_p^{sym}, d_p)$, we have that S_p is closed and $v_p^{sym} \in S_p$.

Now suppose for some $l > 0$ and all v_i where $lendpath(v_i) \leq l$ the above statement holds; we prove it for when $lendpath(v_p^{sym}) = l + 1$. Assume $v_p^{sym} \notin V_{rec}(G)$ (because otherwise by Item 4 we are done). To construct S_p, we first add v_p^{sym} to S_p. Next we build graph G' as follows: G' is obtained from G by removing v_p^{sym} (and, hence, all edges incident on it). Also the recoverable nodes of G' are obtained as follows: We label every node labeled with *recoverable* in G as recoverable in G' as well, and if for $v_h \in V(G')$ (which has not received "recoverable" label yet), it holds that for some d_h, all nodes which have d_h-labeled edges to v_h in G' are labeled recoverable, we label v_h recoverable too. (Note that this last operation may create new recoverable nodes, since v_p^{sym} is removed from G'.) Intuitively G' corresponds to the key graph created when all adversary's *encryption* queries where k_p^{sym} appears as an encryption key is replaced with a new query which uses an adversarial key instead of k_p^{sym}.

Now, to perform the inductive step, since v_p^{sym} is non-continuable, there exists d_p such that for all $v_i \in \textit{prev-nodes}_G(v_p^{sym}, d_p)$, $v_i \xrightarrow{d_p} v_p^{sym}$ is non-continuable. Now for $v_i \xrightarrow{d_p} v_p^{sym}$, if $v_i \in V_{rec}(G)$, then, by Item (4), v_i is an element of a closed set S_i and we add all elements of S_i to S_p. Otherwise, if $v_i \notin V_{rec}(G)$, it should be either: (a) $v_i \notin V_{rec}(G')$ and $v_i \notin V_{rec}(G)$, or (b) $v_i \notin V_{rec}(G)$ and $v_i \in V_{rec}(G')$.

We first show that (a) is a contradiction to the assumption that $v_i \xrightarrow{d_p} v_p^{sym}$ is non-continuable. Since $lendpath_{G'}(v_i) \leq l$ and $v_i \notin V_{rec}(G')$, by the induction hypothesis, v_i is coinductively continuable in G'. This implies that $v_i \xrightarrow{d_p} v_p^{sym}$ is also coinductively continuable in G. (For this we just need to verify that for every coinductively continuable path $v_{r_1} \to^{a_2} v_{r_2} \to^{a_3} \cdots \to^{a_m} v_{r_m}$ in G', there does *not* exist v_{r_i}, for $1 \leq i \leq m$, such that $\textit{prev-nodes}(v_{r_i}, h) = \{v_p\}$, for $h \leq indeg(v_{r_i})$. This is the case because otherwise $v_{r_i} \in V_{rec}(G')$.) For (b) (from item (4)), we obtain that v_i is a member of a closed set S_i in G'; we add all elements of S_i to S_p. Now we can easily see that in this case, S_i is closed under the (thus-far) constructed S_p in G; this is because we already have $v_p^{sym} \in S_p$. Finally it holds that v_p^{sym} is also closed under S_p; this is because all elements of $\textit{prev-nodes}_G(v_p^{sym}, d_p)$ have already been added to S_p. Now from the union property we conclude that S_p is closed in G and this completes the proof. □

B.2 Proof of Theorem 2

We first give some definitions. Under a fixed computational encoding τ and a pair of schemes $\mathcal{E}_p$, if $c \in [\![e]\!]_\tau^{\mathcal{E}_p}$ and $e_1 \sqsubseteq e$ and c_1 is the corresponding computational image of e_1 in e, we call c_1 a *constituent* (or a *constituting bitstring*) *of* c *with respect to* e, τ *and* $\mathcal{E}_p$. When e, τ and $\mathcal{E}_p$ are clear from the context, we simply say that c_1 *is a constituent of* c. Also when $\mathcal{E}_p$ is clear from the context, we simply write $[\![e]\!]_\tau$ instead of $[\![e]\!]_\tau^{\mathcal{E}_p}$. Note that again, given e, τ, and $\mathcal{E}_p$, one can efficiently decide if a given bitstring is a constituent of c. We call $e \in Exp$ *type-1* if there exists a subexp $\{e'\}_k$ of e such that k is an honest encryption key (i.e. $k \in \mathcal{K}_H^{pub} \cup \mathcal{K}_H^{privsym}$) and that k^{-1} does not occur in e.

For simplicity, we slightly modify the CI game and prove Theorem 2 first for this modified version of the game. Then we show that the proof for this modified game easily extends for the original CI game (Subsection C.11). The modified game is defined *only* over an asymmetric encryption scheme $\mathcal{E}$ (there is no symmetric encryption scheme involved) and, hence, in the **setup** phase, only $\{(pk_i, sk_i\}_{1\leq i\leq n}$ and $\{nc_i\}_{1\leq i\leq n}$ are sampled (and their corresponding symbols $\{(k_1, k_1^{-1})\}_{1\leq i\leq n}$ and $\{x_i\}_{1\leq i\leq n}$ are introduced), while in the **interaction** phase, in a query $encrypt(e, k_i)$, e may now contain k_i^{-1}'s as well (as opposed to the the original definition wherein decryption asymmetric keys cannot be part of plaintexts). Henceforth, unless otherwise stated, when we say CI security we mean security under this modified version of the game.

We now define a variant of this modified CI game, by introducing changes to the **interaction** phase, as follows:

- *Restricting adversary's decryption queries:* We add the following extra condition: a *decryption* query (c, s), for a private-key symbol s, is invalid (and hence answered by $\perp$) if there exists $(\{e_j\}_{k_j}, bs_j) \in eval\text{-}exp$ such that $\{e_j\}_{k_j}$ has a type-1 subexp $\{e_p\}_{k_p}$, for $k_p \in \mathcal{K}_H^{pub}$ (i.e. an honest public key), where c is the corresponding computational image of $\{e_p\}_{k_p}$ in bs_j.
- *Allowing a new type of query:* We supply a new type of oracle query, called a *subexp-testing* query, which allows the adversary to test whether his queried bitstring is a "certain" constituent of a ciphertext he has received under an *encryption* query. The *subexp-testing* query takes as input four arguments, $(\{e\}_{k_i}, bs_i, \{e'\}_{k_j}, bs_j)$, where $(\{e\}_{k_i}, bs_i) \in eval\text{-}exp$, $k_j \in \mathcal{K}_H^{pub}$ and $\{e'\}_{k_j}$ is a type-1 subexp of $\{e\}_{k_i}$. (Wlog, we assume that all the adversary's *subexp-testing* queries satisfy these two conditions.) The output of this query is 1 if bs_j is the constituting bitstring of bs_i corresponding to $\{e'\}_{k_j}$, and 0 otherwise.

The rest of the game is as in the CI game. We call this new variant the *weak, subexp-testing*, or *WSTCI* game. As for the CI game, we may define the advantage of an adversary $\mathcal{A}$, denoted by $Adv_{\mathcal{E},\mathcal{A}}^{\mathrm{WSTCI}}(\eta)$, when run under the WSTCI game, and give the following definition.

Definition 5. *An asymmetric encryption scheme $\mathcal{E}$ provides l-WSTCI security if for every $\mathcal{A}$ for whom the diameter of the hidden subgraph of $G(\mathcal{A})$ is upperbounded by l it holds that $Adv_{\mathcal{E},\mathcal{A}}^{WSTCI}(\eta)$ is negligible.*

To prove Theorem 2 we first show that IND-CCA2 security implies l-WSTCI security, and then prove that l-WSTCI+IND-CCA2 security implies l-CI security. It will become clear later why we imposed an additional decryption restriction to the WSTCI game. The proof of Theorem 2 follows from the following two lemmas:

Lemma 2. *If $\mathcal{E} = (Gen, Enc, Dec)$ provides IND-CCA2 security, then it is l-WSTCI secure for every constant l.*

Lemma 3. *For every IND-CCA2 secure encryption scheme $\mathcal{E}$, if $\mathcal{E}$ provides l-WSTCI security, it also provides l-*CI *security.*

Informally speaking, for Lemma 3 it suffices to prove that the probability that any adversary ever produces a ciphertext which is legitimate under the CI game but illegitimate under the WSTCI game (due to the added decryption restriction) is negligible. We show this with the aid of the supplied *subexp-testing* oracle in the WSTCI game. It will become clear later why we need to do the proof first for WSTCI security. (Informally speaking, this is, in part, due to the issue related to *decryption* queries, which we briefly alluded to in the introduction.) In the next two sections, we give the proofs of the above two lemmas.

C Proof of Lemma 2

C.0.1 Notation We introduce some notation and terminology that will be used in the remainder of the paper. Given $L = \{(e_1, E_1), \ldots, (e_n, E_n)\}$, where $e_1, \ldots, e_n$ are expressions and $E_i \leftarrow [\![e_i]\!]_\tau^{\mathcal{E}}$, for $1 \le i \le n$ (and arbitrary τ and $\mathcal{E}$), we say that e *is coinductively visible in* L (more precisely, e *is coinductively visible in* $(e_1, \ldots, e_n)$,) if $e \sqsubseteq e_j$ for some $1 \le j \le n$ and $k^{-1} \in closure_c(e_1, \ldots, e_n)$ for *every* k which *encrypts* e in e_j. Also we call $E \in \{0,1\}^*$ *coinductively visible in* L if for some j, E is a *constituent* of E_j (with respect to e_j, τ and $\mathcal{E}$) and its corresponding expression is coinductively visible in L. Finally, we call (E, i) a *coinductively visible pair* in L if for some j, E is a *constituent* of E_j (with respect to e_j, τ and $\mathcal{E}$) and its corresponding expression is of the form $\{.\}_{k_i}$ and is coinductively visible in L. For example, under this terminology, one may restate the *decryption* condition of the CI game (Subsection 5.1) as follows: $decrypt(c, k_i^{-1})$ is a valid *decryption* query if (c, i) is *not* coinductively visible in *eval-exp*.

As explained earlier, under the CI (and other related) games we associate an increasing number to each occurrence of each challenger's private key in the sequence of the adversary's *encryption* queries. For convenience, we make the following notational convention about the format of the adversary's inputs to *encryption* queries: the oth occurrence of a *challenger*'s key k_j^{-1} in the sequence of *encryption* queries is represented by $k_{j,o}^{-1}$. (Recall that, according to the assumption made earlier, the challenger's private keys are now asymmetric decryption keys, $k_1^{-1}, \ldots, k_n^{-1}$.) For example, if the two first *encryption* queries involve encrypting k_j^{-1} under, respectively, k_{i_1} and k_{i_2}, under the new representation they are denoted $encrypt(k_{j,1}^{-1}, k_{i_1})$ and $encrypt(k_{j,2}^{-1}, k_{i_2})$. This convention will greatly simplify our presentation later.

We continue with some more notations. We call e_1 a *proper* subexp of e_2 if $e_1 \sqsubseteq e_2$ and $e_1 \neq e_2$. If $e_1 \sqsubseteq e$ and $e_2 \sqsubseteq e$ we call e_1 and e_2 *disjoint* if it holds that $e_1 \not\sqsubseteq e_2$ and that $e_2 \not\sqsubseteq e_1$. For $e \in Exp$ and symbol s occurring in e only once, if s is encrypted under $k_1, \ldots, k_r$ in e, pictorially being of the form $\left\{ \{\{(s, \ldots)\}_{k_1}, \ldots\}_{k_2} \ddots \right\}_{k_r}$, then we call k_i the ith-*innermost* (also ith-*closest*) key encrypting s in e. (We required s to occur only once in e to avoid confusion regarding to what occurrence of s we are referring.) Note that $k_1, \ldots, k_r$ above are not necessarily distinct, so we always speak about the *multiset* of keys which encrypt a symbol s in e. Recalling the definition of the key graph $G(\mathcal{A})$, sometimes it is helpful to work with a more elaborate key graph, denoted by $G_{\text{elab}}(\mathcal{A})$, whose *set of nodes* is the same as that of $G(\mathcal{A})$ (except that we do not have any v_i^{sym} nodes, and v_j^{asy} nodes are now denoted v_j), and whose *edges* are of the form $v_i \xrightarrow{o,d} v_j$, indicating that k_i is the dth-closest key encrypting $k_{j,o}^{-1}$ in the sequence of *encryption* queries. For $e \in Exp$, we define the *(maximum) encryption depth* of e as follows: $enc\text{-}depth(s) = 0$ for every basic symbol s, and $enc\text{-}depth((e_1, e_2)) = \max(enc\text{-}depth(e_1), enc\text{-}depth(e_2))$, and finally $enc\text{-}depth(\{e\}_k) = enc\text{-}depth(e) + 1$. Suppose k^{-1} occurs in e only once and is encrypted in e under (possibly) multiple instances of encryption under k_1 (as a result of nested encryption); Then the statement, "$\{e_1\}_{k_1}$ *is the smallest subexp of* e *with* e_1 *containing* k^{-1}", means that $k^{-1} \sqsubseteq \{e_1\}_{k_1} \sqsubseteq e$, and that e_1 does not contain a proper subexp $\{e_2\}_{k_1}$ with k^{-1} occurring in e_2. We say that $v_r \xrightarrow{a_{r+1}} v_{r+1} \xrightarrow{a_{r+2}} \ldots \xrightarrow{a_p} v_p$ is *strongly, coinductively continuable* [5] if for all i's,

[5] Here the nodes having consecutive indices (i.e. being v_r, v_{r+1}, $\ldots$, v_p) is coincident and is not a requirement of the definition.

where $r \leq i \leq p$, it holds that $v_i \xrightarrow{a_{i+1}} v_{i+1} \xrightarrow{a_{i+2}} \ldots \xrightarrow{a_p} v_p$ is a coinductively continuable path. If $\mathcal{P} = v_i \xrightarrow{a_{i+1}} v_{i+1} \xrightarrow{a_{i+2}} \ldots \xrightarrow{a_{i+k}} v_{i+k}$ is a coinductively continuable path in $G(\mathcal{A})$, where $indeg(v_i) > 0$ and $a_i \leq indeg(v_i)$, then we define $\Re_{\text{cont.}}(\xrightarrow{a_i} \mathcal{P})$ to be the *multiset* containing all nodes v_r such that (a) k_r encrypts k^{-1}_{i,a_i} (i.e. k_r encrypts the a_ith occurrence of k_i^{-1} in the sequence of *encryption* queries), and (b) $v_r \xrightarrow{a_i} \mathcal{P}$ is coinductively continuable. Defined formally, for all $d \geq 1$, if $v_r \xrightarrow{a_i,d} v_i$ is an edge in $G_{elab}(\mathcal{A})$ and $v_r \xrightarrow{a_i} \mathcal{P}$ is coinductively continuable, then we add v_r to $\Re_{\text{cont.}}(\xrightarrow{a_i} \mathcal{P})$. For simplicity, in the last definition, we also associate d as a *tag* to v_r in $\Re_{\text{cont.}}(\xrightarrow{a_i} \mathcal{P})$, and this way, we define *the ith-lowest node among $\Re_{cont.}(\xrightarrow{a_i} \mathcal{P})$* to be the node with the ith-smallest tag in this multiset. We stress here that $\Re_{\text{cont.}}(\xrightarrow{a_i} \mathcal{P})$ is a multiset, containing possibly multiple instances of a single key but with different associated tags. Moreover, by abusing the notation, we call k_h *the ith-lowest key among $\Re_{cont.}(\xrightarrow{a_i} \mathcal{P})$* if v_h is the ith-lowest node among $\Re_{\text{cont.}}(\xrightarrow{a_i} \mathcal{P})$. Finally if $\mathcal{P}$ is not a coinductively continuable path in $G(\mathcal{A})$ we define $\Re_{\text{cont.}}(\xrightarrow{a_i} \mathcal{P}) = \emptyset$, for all $a_i \geq 1$.

Proceeding with the Proof of Lemma 2. Suppose $\mathcal{A}_{wstci}$ is an adversary attacking the asymmetric encryption scheme $\mathcal{E} = (Gen, Enc, Dec)$ in the sense of l-WSTCI. From $\mathcal{A}_{wstci}$, we will construct another adversary, $\mathcal{A}_{cca}$, who attacks the scheme in the sense of IND-CCA2, and whose CCA2 advantage is polynomially related to WSTCI-advantage of $\mathcal{A}_{wstci}$. First, since $\mathcal{A}_{cca}$ works under an IND-CCA2 game, we assume that he is provided with a *left-or-right encryption oracle*, $O^{\mathcal{E}}_{pk,b}$, which on input (bs_0, bs_1) returns $c \leftarrow Enc_{pk}(bs_b)$, and with a *decryption oracle*, $O^{\mathcal{E}}_{sk}$, which on input c' returns $Dec(c', sk)$, provided that c' has not been previously produced by the encryption oracle. *We fix l and $\mathcal{E}$* as well as $\mathcal{A}_{cca}$'s associated CCA2 oracles, $O^{\mathcal{E}}_{pk,b}$ and $O^{\mathcal{E}}_{sk}$, *throughout this section.*

We present a general picture of how our reduction is developed. For an adversary $\mathcal{A}$ playing in the WSTCI game against a *challenger* $\mathcal{B}$, we refer to the *strategy* prescribed under the standard WSTCI game for $\mathcal{B}$ to reply to $\mathcal{A}$'s oracle queries when the *challenge bit* is b as $wstci^b$. The difference between $wstci^0$ and $estci^1$ lies only in the way they respond to a *challenge* query, under one of which $\mathcal{B}$ outputs the real reply and under the other he outputs the random reply. Under $wstci^b$ (for both $b = 0$ and $b = 1$), for every private-key symbol k_i^{-1} of the challenger, $\mathcal{B}$ uses a fixed bitstring value sk_i (in constructing her replies to $\mathcal{A}$), whose public-key value is known to $\mathcal{A}$, and whose value, sk_i, is given to $\mathcal{A}$ should it be corrupted. During our *reduction procedure*, we will work with different *strategies* that may be employed by the challenger to answer to adversary's queries. Broadly speaking, under these *strategies*, replies given to some of the adversary's *encryption* queries may be *faked*, in the sense that in computing the ciphertexts to be returned, certain private keys (of the challenger) may be evaluated under *fake* values. For example, for a private key symbol k_i^{-1} of the challenger, the challenger may replace certain (or all) occurrences of k_i^{-1} under a fake value fsk_i, which may be completely *independent* of pk_i, the value that the adversary has for k_i. Thus in the discussion below, by *strategy* we mean a procedure adopted by a *challenger* to play against an *adversary* in the game.

Now, turning into $\mathcal{A}_{wstci}$ introduced above, "$\mathcal{A}_{wstci}$ breaking $\mathcal{E}$ in the sense of WSTCI" indeed means that $\mathcal{A}_{wstci}$ is effectively able to tell the two *strategies*, $WSTCI^0$ and $WSTCI^1$, of a *challenger* apart. Now to do the reduction, our to-be-constructed $\mathcal{A}_{cca}$ will simulate $\mathcal{A}_{wstci}$ by responding to $\mathcal{A}_{wstci}$'s oracle queries under a (to be explained) feasible *strategy* (which may create fake replies sometimes as well) in such a way that at the end, $\mathcal{A}_{wstci}$'s *output* somehow helps $\mathcal{A}_{cca}$ to determine (with a polynomially close advantage) that under what *CCA2-world* he was operating. More technically, $\mathcal{A}_{cca}$ develops an *strategy*, $\mathbf{Strgy}_0$, for replying to $\mathcal{A}_{wstci}$'s oracle queries, and $\mathbf{Strgy}_0$, in turn, depending upon under what *CCA2-world* $\mathcal{A}_{cca}$ is running, results in one of the two *induced strategies* $\mathbf{Strgy}_0^0$ and $\mathbf{Strgy}_0^1$, for which it provably holds that, if an event *Bad* does not occur (which will be the case with a non-negligible probability), then the advantage of $\mathcal{A}_{wstci}$ of differentiating between $\mathbf{Strgy}_0^0$ and $\mathbf{Strgy}_0^1$ is polynomially close to that of differentiating between $WSTCI^0$ and $WSTCI^1$. Now since, by assumption, $\mathcal{A}_{wstci}$ is able to effectively tell $WSTCI^0$ and $WSTCI^1$ apart, so is he able to tell $\mathbf{Strgy}_0^0$ and $\mathbf{Strgy}_0^1$ effectively apart, giving $\mathcal{A}_{cca}$ a way to tell effectively under what CCA2 world he was operating. To complete the simulation, in case event *Bad* occurs (which is a failed situation), $\mathcal{A}_{cca}$ will simply flip a coin to guess his CCA2 challenge bit. Otherwise, he will determine his CCA2 challenge bit with the aid of $\mathcal{A}_{wstci}$.

After developing $\mathbf{Strgy}_0$, to show that the simulation is fulfilled correctly, we have to verify the following requirements: (a) *Efficiency*: $\mathbf{Strgy}_0$ is *efficiently implementable* (e.g. showing that at no point of the simulation $\mathcal{A}_{cca}$ will need to query his CCA2 *decryption* oracle on invalid ciphertexts[6]; the whole simulation works in polynomial time; etc.), and (b) *Correctness*: Rigorously showing that the two advantage functions described above are polynomially related.

In Subsection C.2 we will formalize $\mathbf{Strgy}_0$, the *strategy* used by $\mathcal{A}_{cca}$ to simulate and reply to $\mathcal{A}_{wstci}$'s queries during the l-WSTCI game. In the developed algorithm (and throughout this section), we assume that $P(\eta)$ is a polynomial upperbounding the running time of $\mathcal{A}_{wstci}$. Actually, P suffices to only upperbound the following two parameters: the *maximum* number of times that a challenger's private-key symbol occurs in $\mathcal{A}_{wstci}$'s encryption queries, and the *maximum* encryption-depth of an expression given to the encryption oracle. For the sake of readability we adopt the following convention in the remainder of the proof.

Convention 1 *We will write P and n instead of $P(\eta)$ and $n(\eta)$. (Recall that n is the number of public/private key pairs sampled by the challenger.)*

We first provide an informal overview of the *reduction* in the next subsection.

C.1 Overview of the Reduction

We first try to explain the idea of the reduction for a simple case. The case where $l = 1$ (i.e. the diameter of the coinductively-hidden subgraph is 1) is not a good representative example, since for $l = 1$ one can easily show that no cycle can exist in the underlying hidden subgraph, and so this case may easily be handled only using the techniques of [46]. To demonstrate our techniques, we explain the idea for a simple case $l' = 1$, where l' is a new parameter denoting the length of the longest path which ends in the challenge key. To simplify the discussion even further, let's first consider a simplified version of the WSTCI game; the one in which (i) there is *no* decryption query (and we are interested in making our reduction to the IND-CPA security of the scheme), (ii) the adversary may only *challenge* a key in the **guessing** phase (i.e. no nonce *challenge*), and (iii) $l' = 1$: the length of the longest path ending in the *challenge* key is at most one. (The diameter of the whole hidden subgraph could be any large number, even non-constant.)[7] We will try to first make our arguments for the simpler setting of acyclic single encryptions (as in the setting of [46]), and then show what challenges arise if we want to extend them to the nested cyclic encryption setting, and how we can resolve them.

Assume that $\mathcal{A}$ is an adversary able to win under the restricted version of the WSTCI game described above. That is, $\mathcal{A}$ is able to distinguish between encryptions of real/random messages under $\tau(k_{ch})$ (after observing *eval-exp*), where k_{ch} is the (*a priori unknown*) *challenge*

[6] Actually this is one reason why we needed to prove the simulation first for the more "decryption-limited" WSTCI security.

[7] This actually shows that our result holds for a stronger setting in which no restriction is placed on the diameter of the hidden subgraph, only assuming that the length of the longest path ending in the challenge key is at most constant.

key (specified by $\mathcal{A}$ himself in the **guessing** phase), and τ is the underlying computational mapping. We wish to reduce $\mathcal{A}$'s described ability to a *CPA attack*. First the immediately arisen difficultly toward this is clear: since k_{ch}^{-1} may have previously occurred as a plaintext (i.e. in an expression in *eval-exp*), it is not immediate to translate $\mathcal{A}$'s ability to distinguish between encryptions under $\tau(k_{ch})$ into a CPA attack. But now imagine a *new* world w' under which *all* instances of k_{ch}^{-1} occurring in encryption queries (i.e. $k_{ch,o}^{-1}$ for every $o \geq 1$, notationally describing) are replaced with a *fake* random value instead of the actual value $\tau(k_{ch}^{-1})$. That is, w' is the *world* in which first k_{ch}, the key which is going to be challenged in the **guessing** phase is guessed, and subsequently all instances of k_{ch}^{-1} in $\mathcal{A}$'s *encryption queries* are replaced with a fake value. (Note that w' may *not* be successfully constructed, because our **guessing** may fail; but to simplify the analysis, let's for a moment assume that we *a priori* know what key is going to be challenged in the **guessing** phase.) First it is easy to see that the ability of *any* adversary to distinguish between encryptions of real/random messages under $\tau(k_{ch})$ in w' would now easily translate into a feasible CPA attack. Now, denoting by w the actual world under which all queries of the adversary are supposed to be replied to (i.e. no fake value for k_{ch}^{-1}), we consider two possibilities:

(a) The two worlds w and w' are computationally indistinguishable.
(b) There exists an adversary, $\mathcal{A}'$, who can distinguish between w and w'.

Now if (a) is the case, this means that $\mathcal{A}$, in particular, is not able to tell the two worlds w and w' apart. This in turn implies that $\mathcal{A}$'s *advantage* of distinguishing between encryptions under $\tau(k_{ch})$ in w should be negligibly close to that in w'. (If this does not hold—i.e., the advantage of $\mathcal{A}$ (in the sense of WSTCI) induced under w is non-negligibly different from that under w'—it is easy to see how one can modify $\mathcal{A}$ to obtain $\mathcal{B}$ who can differentiate between w and w'.) Therefore, from our earlier assumption that $\mathcal{A}$ has a non-negligible advantage under w, we obtain that $\mathcal{A}$ has a non-negligible advantage under w' as well, giving rise to a CPA attack as described above. Now what if (b) holds; namely, there exists $\mathcal{A}'$ who has a *non-negligible* chance of telling w and w' apart? If this is the case, then $\mathcal{A}'$ should have a *non-negligible* chance of telling w_i and w_{i+1}, for some i, apart, where w_k denotes a world in which all the first k occurrences of k_{ch}^{-1} are "faked" and the rest are replaced with real values.

Now here is the part that the allowance of both *nested encryptions* and *key cycles* makes the situation somewhat onerous. If we were only dealing with *single encryptions without key cycles* as the Panjwani's framework [46] (i.e. each encryption query is of the form $encrypt(k_i, k_j)$ with no cycle creation), one would simply reduce $\mathcal{A}'$'s ability to distinguish between w_i and w_{i+1} to a CPA-attack as follows. Suppose the CPA-encryption oracle is parameterized over (pk, sk) where pk is publicly known and sk is secret. Now the reduction is as follows: guess $k_{g_{i+1}}$, the *single key* which encrypts the $i+1$st occurrence of k_{ch}^{-1}, associate sk with $k_{g_{i+1}}^{-1}$, generate two values sk_{ch} and fsk_{ch} for k_{ch}^{-1}, replace the first ith occurrences of k_{ch}^{-1} with fsk_{ch} (a fake value), and for the $i+1$'st occurrence, return the output of the *left-or-right CPA encryption oracle* on (sk_{ch}, fsk_{ch}), and finally replace any subsequent occurrences of k_{ch}^{-1} with sk_{ch}. Now the assumptions of *key acyclicity* and *single encryption*, together, ensure that the reduction is fulfilled correctly; namely, $k_{g_{i+1}}^{-1}$, to which sk is associated, *does not* itself occur as a plaintext in any encryption query. The reason is, otherwise, $k_{g_{i+1}}^{-1}$ should be encrypted either under itself, or under k_{ch}, or under some key other than these two; now, the two former cases imply the existence of a cycle in the graph (a contradiction to the acyclicity assumption), and the latter implies that $l' > 1$, reaching a contradiction again.

In the case of *cyclic, nested encryptions* (as in our setting), however, the above appealing reduction may fail for the following reason: there may *legitimately* be a self-loop on $k_{g_{i+1}}^{-1}$ (more precisely, on $v_{g_{i+1}}$ in the underlying key graph), or there *may* be an edge from v_{ch} back to $v_{g_{i+1}}$.

(i.e. $k_{g_{i+1}}^{-1}$ may be encrypted under k_{ch} at some point.) Note that any of these possibilities *may* happen while $k_{g_{i+1}}^{-1}$ (as well as k_{ch}^{-1}) *remains coinductively irrecoverable* and l' does *not* exceed 1—for example, the encryption query containing the $i{+}1$st occurrence of k_{ch}^{-1} might be *encrypt* (e, t), for $e = \left\{ \ldots \{k_{ch}^{-1}\}_{k_{g_{i+1}}} \ddots \right\}_{k_p}$, and the adversary might at some other point (earlier or later) ask the query *encrypt* (e', t'), for $e' = \left\{ \left\{ k_{g_{i+1}}^{-1} \right\}_{k_{ch}} \right\}_{k_{g_{i+1}}}$. Note that in this example, l' does not necessarily change by the latter query, and also $k_{g_{i+1}}^{-1}$ may still be coinductively irrecoverable. (For example, if k_p^{-1} is coinductively irrecoverable.) Now obviously, in such a case, the reduction described above (which works for the *single encryption* case) fails. (This is because the reduction requires $k_{g_{i+1}}^{-1}$ *not* to occur as a plaintext itself.) In summary, the above discussion shows that the mere assumption that the guessed $k_{g_{i+1}}^{-1}$ be coinductively irrecoverable does *not* suffice, and one has to add extra conditions to resolve the above obstacle.

To remedy the above complication, we have to, inescapably, further assume that $k_{g_{i+1}}^{-1}$ does *not* occur as a plaintext, in addition to being *coinductively irrecoverable*. But, how can we guarantee that for the $i + 1$'st occurrence of k_{ch}^{-1} (for any i) such a choice for $k_{g_{i+1}}^{-1}$ indeed exists? Now this is the point we resort to our developed notion of *coinductive continuability*: Since k_{ch}^{-1} is coinductively irrecoverable, by Lemma 1, its associated node v_{ch} in the underlying key graph is coinductively continuable, where for the case $l' = 1$, this means that, for every $1 \leq j \leq indeg(v_{ch})$, there exists an edge $v_{g_j} \xrightarrow{j} v_{ch}$ such that $indeg(v_{g_j}) = 0$. (Put differently, for every j, there exists a coinductively irrecoverable $k_{g_j}^{-1}$ such that $k_{g_j}^{-1}$ does *not* occur as a plaintext and that k_{g_j} encrypts $k_{ch,j}^{-1}$.) This solves the complication arisen above: namely, to reason about the indistinguishability of w_i and w_{i+1} by reduction, one suffices to guess $k_{g_{i+1}}^{-1}$ in such a way that it satisfies the condition that $v_{g_{i+1}} \in \Re_{\text{cont.}}(\xrightarrow{i+1} v_{ch})$, and then proceed as in the *single encryption* case.

Note that the above lines of reasoning consists of applying reduction in two phases: (1) the adversary has no significant advantage of distinguishing between encryptions of real/random messages (under key $\tau(k_{ch})$) when his *view* is set according to w', and (2) no adversary is able to differentiate between w and w' with a *non-negligible* probability. In trying to extend the described idea to the case $l' > 1$, we can again similarly define worlds w and w', and work out Step (1) above as easily as for the case $l = 1$. However, for Part (2), once trying to, using a left-or-right CPA-encryption oracle, create one of the w_i or w_{i+1} worlds, the guessed $k_{g_{i+1}}^{-1}$, may now in turn be encrypted under some other keys (and those keys may again be encrypted under some other keys, depending on the value of l'), and, hence, Part (2) may not be accomplished in "one-shot" as before. Informally speaking, what more is involved for this general case is that, after $k_{g_{i+1}}^{-1}$ is guessed, toward reasoning that w_i and w_{i+1} are indistinguishable, we first prove the indistinguishability of two *new intermediate* worlds w_i^0 and w_{i+1}^0 defined as follows:

Under both w_i^0 and w_{i+1}^0 we replace *all* occurrences of $k_{g_{i+1}}^{-1}$ with a *fake* value, the first ith occurrences of k_{ch}^{-1} also with a *fake* value, the $i + 1$'st occurrence of k_{ch}^{-1} with the real value for w_i^0 and the fake value for w_{i+1}^0, and finally any subsequent occurrence of k_{ch}^{-1} with the real value.

Now from the formalizations of w_i^0 and w_{i+1}^0 it is clear they can be simulated and produced by a *left-or-right* encryption oracle. (And hence the ability of any adversary to distinguish between w_i^0 and w_{i+1}^0 will reduce to a CPA attack.) Now having shown w_i^0 and w_{i+1}^0 are indistinguishable, we move a step backward, and reason about the indistinguishability of w_i^0 (resp. w_{i+1}^0) from a *new* world w_i^1 (resp. w_{i+1}^1), defined similarly to w_i^0 (resp. w_{i+1}^0) except that all occurrences of $k_{g_{i+1}}^{-1}$ receive the *real* value. Comparing w_i^0 to w_i^1 (analogously w_{i+1}^0 to w_{i+1}^1), they both "behave" identically with respect to k_{ch}^{-1} (i.e. they both agree on the occurrences of k_{ch}^{-1} that they give a *fake* value to), while for $k_{g_{i+1}}^{-1}$, one replaces *all* its occurrences with a *fake* value, and the other with the *real* value. Now to argue about the indistinguishability of w_i^0 and w_i^1, we have to proceed as we did for w and w', with the difference that the "focused" key (the key whose all occurrences receive the *fake* value under one world and the *real* value under the other world) is now $k_{g_{i+1}}^{-1}$ (instead of k_{ch}^{-1} which was for w/w'), and with the *replying strategy* for k_{ch}^{-1} being identical for both w_i^0/w_i^1. (i.e., They both give a *fake* value to the exact same occurrences of k_{ch}^{-1}.)

The above *successive world-creation* approach will be very difficult to handle when $l' > 1$, as it involves "manually" introducing many intermediate worlds. Therefore, as in [46], we carry out the whole simulation within a single algorithmic procedure, with letting the intermediate worlds be created as the result of the random coins used in the procedure. We again first describe the idea for the case of *single, acyclic encryption* when $l' = 1$, and for the *absence* of *decryption* queries. (i.e., We focus on "reducing the adversary's ability to distinguish between w and w' to a CPA attack".) Then we describe some of the challenges that occur when trying to extend it to the case of *nested, cyclic encryption*, and show how to circumvent them. Here is how the general reduction works: we pick $k_{u_1}, k_{u_2} \leftarrow \{k_1, \ldots, k_n\}$, $on_2 \leftarrow \{1, \ldots, P\}$, and call our guessing *successful* (or event *success* occurs) if $k_{u_2} = k_{ch}$ (i.e. the *challenge* key) and k_{u_1} encrypts k_{u_2,on_2}^{-1}. (We again remark that in the case of *single encryption*, any occurrence of any decryption key is encrypted under a single key.) Proceeding with the reduction, we associate the public key value parameterizing the LOR CPA-encryption oracle with k_{u_1}, replace k_{u_2,o_1}^{-1} with fsk_{u_2} (a *fake* value), and k_{u_2,o_2}^{-1} with sk_{u_2} (a *real* value), for all $o_1 < on_2$ and $o_2 > on_2$. Finally, for k_{u_2,on_2}^{-1}, which should be encrypted under k_{u_1} according to our guessing, we reply to its associated encryption query with the output of the LOR encryption oracle on (sk_{u_2}, fsk_{u_2}). As usual, we return the outcome of a fair coin flip if our guessing fails. Now we can verify the following:

(a) for all $i > 1$, the *view* of the adversary when $(b = 0 \wedge on_2 = i \wedge success)$ is *identically distributed* to that when $(b = 1 \wedge on_2 = i - 1 \wedge success)$,
(b) the view of the adversary when $(b = 0 \wedge on_2 = 1 \wedge success)$ (resp. $(b = 1 \wedge on_2 = indeg(v_{u_2}) \wedge success)$) is identically distributed to that under w (resp. w'), and
(c) the probability that $(on_2 = i \wedge success)$ occurs is $1/Pn^2$ (and hence independent of i).

The last item may be less trivial to see why holds true; it derives from the fact that k_{u_1}, k_{u_2}, and on_2 are all picked *independently* and *uniformly* at random. If we refer to any distribution other than those two appearing in Item (b) (i.e., the two which correspond to w and w') as an *intermediate* distribution, now (a) and (c) together imply that all *intermediate* distributions created under $b = 0$ will, informally speaking, *cancel out* all those created under $b = 1$, and (b) yields that the resulting *reduction factor* is $1/Pn^2$ (in reducing an adversary's ability to distinguish w/w' to a CPA attack).

If we want to extend the above idea to the case of *nested, cyclic encryptions*, from the earlier discussion, we know that the additional condition $v_{u_1} \in \Re_{\text{cont.}}(\xrightarrow{on_2} v_{ch})$ should also be satisfied. Now we can easily verify that assertions (a) and (b) above still hold true; however, assertion (c) may fail to hold because of the reason described next. In order for event *success* to occur, it should hold $v_{u_1} \in \Re_{\text{cont.}}(\xrightarrow{on_2} v_{ch})$, whose occurrence depends on the size of *multiset* $\Re_{\text{cont.}}(\xrightarrow{on_2} v_{ch})$, and hence *is dependent* on the particular value of on_2. As a result, we *cannot* argue anymore that all intermediate distributions will "cancel out" each other, as *was* the case for the single encryption case. Put differently, the probability that $v_{u_1} \in \Re_{\text{cont.}}(\xrightarrow{on_2} v_{ch})$, is not *uniform* over the possible choices of on_2. To make it *uniform*, we guess two more parameters $ed_2 \leftarrow \{1, \ldots, P\}$ and $ed'_2 \leftarrow \{1, \ldots, ed_2\}$, and now call $(v_{u_1}, on_2, ed_2, ed'_2, v_{u_2})$ a *successful* guess, if $v_{ch} = v_{u_2}$, $ed_2 = \left|\Re_{\text{cont.}}(\xrightarrow{on_2} v_{ch})\right|$, and v_{u_1} is *the* ed'_2*th-lowest node among* $\Re_{cont.}(\xrightarrow{on_2} v_{ch})$. (We refer the reader to the related definitions given earlier.) Now it is easy to verify that the probability that $(on_2 = i \wedge success)$ holds is

$$\frac{1}{P} \cdot \frac{1}{P} \cdot \frac{1}{n} \cdot \left[\overbrace{(\frac{1}{n} \cdot \frac{1}{ed_2}) + \cdots + (\frac{1}{n} \cdot \frac{1}{ed_2})}^{ed_2 \text{ factors}} \right] = \frac{1}{P^2 n^2},$$

and hence is *independent* of i now. Now all assertions (a), (b), and (c) above hold true, as desired.

C.1.1 WSTCI and Restricted Decryption Queries.

Now we explain why we added a new *decryption* condition to the WSTCI game. Suppose one wants to extend the idea presented above (for the *cyclic nested encryptions*) to the case where *decryption queries* are also allowed. Assume, for the chosen parameters described above, it holds that event *success* occurs, $ed'_2 < ed_2$, and the query containing the on_2th occurrence of $k_{u_2}^{-1}$ is $encrypt(e, k_h)$. Now since event *success* occurs and ed'_2 is *strictly* less than ed_2, it must be the case that $\{e\}_{k_h}$ has a subexp $e_1 = \left\{ \{k_{u_2,on_2}^{-1}\}_{\ddots} \right\}_{k_{u_1}}$, which is encrypted in $\{e\}_{k_h}$ under at least one k_p such that k_p^{-1} is *coinductively irrecoverable*. Our *simulation* above is such that E, the *computational image* of e_1 is produced by the *LOR encryption oracle*, and from that, E_1, a *computational image* of $\{e\}_{k_h}$ is constructed and given to the adversary. Now in the *absence* of the new *decryption* condition added to the WSTCI game, E would *be* a valid *decryption* query for $\mathcal{A}_{ci}$ (i.e. the adversary who plays under the CI game which does not have the decryption restriction), while it is *invalid* for the *simulating CCA2-adversary*, causing our simulation to go wrong. That is, $\mathcal{A}_{ci}$ would *legitimately* be allowed to ask for decryption of E under any key, while E would be the *challenge ciphertext* for the CCA2-adversary. Thus, to eliminate this issue from our simulation, we first do the reduction for the more *decryption*-restricted WSTCI game, and then we will prove that the probability that any adversary can indeed produce such a ciphertext for decryption is negligible (Lemma 3).

Now for the general case $l > 1$, we randomly guess, a priori, a "strongly, coinductively continuable path" $v_{u_s} \to v_{u_{s+1}} \to \cdots \to v_{u_l}$, with associated parameters

$$\{(on_{s+1}, ed_{s+1}, ed'_{s+1}), \ldots, (on_l, ed_l, ed'_l)\},$$

and, for every $s + 1 \leq i \leq l$, expect to have

(a) $k_{u_{i-1}}$ encrypts k_{u_i,on_i}^{-1},
(b) $ed_i = \left|\Re_{\text{cont.}}(\xrightarrow{on_i} v_{u_i} \xrightarrow{on_{i+1}} \ldots \xrightarrow{on_l} v_{u_l})\right|$,
(c) $v_{u_{i-1}}$ is the ed'_ith lowest node among $\Re_{\text{cont.}}(\xrightarrow{on_i} v_{u_i} \xrightarrow{on_{i+1}} \ldots \xrightarrow{on_l} v_{u_l})$.

Note that item (c) automatically derives from item (a).

C.2 Algorithm ($\mathbf{Strgy_0}$ strategy)

We now formalize $Strgy_0$, the strategy used by the simulating adversary $\mathcal{A}_{cca}$ to reply to $\mathcal{A}_{wstci}$'s queries. To make the presentation simpler, we first show how the proof works for a special case of the WSTCI game where only a secret-key challenge is allowed. Then we will explain how the proof works for the case where only a nonce challenge is allowed. (The proof of this latter case proceeds very similarly to that of the first case with a few slight modifications.) Note that security with respect to both of these two sub-games implies security with respect to the original WSTCI game. (This is because, if $\mathcal{A}_{wstci}$ wins in the WSTCI game with a non-negligible probability, then one can construct another adversary who wins in one of these two sub-games.)

Assumption 1 *$\mathcal{A}_{wstci}$ makes only a secret key challenge in his* ***guessing*** *phase. Moreover, if k_i^{-1} is challenged by $\mathcal{A}_{wstci}$, then it holds $indeg(v_i) > 0$. (We call k_i^{-1} the* challenge key.*)*

For the second assumption above note that if with a non-negligible probability it holds that $indeg(v_i) = 0$, then an IND-CCA2 attack readily follows.

We recall some of the assumptions made earlier. We assume $\mathcal{A}_{cca}$'s LOR-encryption oracle is parameterized over pk, and its decryption oracle is parameterized over sk, where $(pk, sk) \leftarrow Gen(1^\eta)$. First we make some without-loss-of-generality assumptions. We assume that $\mathcal{A}_{wstci}$ does not make any *nonce-revelation* query (i.e. a query of the form $reveal(x_i)$); For any WSTCI-adversary who makes such a query there exists another adversary, who at the beginning of the **interaction** phase corrupts a specially-selected decryption key k_h^{-1} of the challenger, and for any $reveal(x_i)$ query, instead makes an *encryption* query $encrypt(x_i, k_h)$. Also we assume that if $\mathcal{A}_{wstci}$ makes a *subexp-testing* query $subexp\text{-}testing(\{e\}_{k_i}, bs_i, \{e'\}_{k_j}, bs_j)$ it holds that $(\{e\}_{k_i}, bs_i) \in eval\text{-}exp$ and that $\{e'\}_{k_j}$ is a type-1 subexp of $\{e\}_{k_i}$. Next, we assume that if the adversary makes a query $decrypt(c, k_i^{-1})$, then $k_i^{-1} \notin closure_c(eval\text{-}exp^1)$; note that for $k_j^{-1} \in closure_c(eval\text{-}exp^1)$, even if the adversary does not know the underlying bitstring value of k_j^{-1} (for example if he has obtained a computational image of $\{k_j^{-1}\}_{k_j}$, which does not necessarily reveal the value of k_i^{-1},) he can obtain it by issuing a query $corrupt(k_j^{-1})$, which is easy to see that *does* not change the adversarial symbolic knowledge. Also whenever we write $k_i, k_j, k_p, \ldots$ we mean honest party's public keys. Finally for simplicity we write an *encryption* query $encrypt(e, k_i)$ as $encrypt(e, i)$ (and use a similar convention for *corrupt* and *decryption* queries).

$\mathcal{A}_{cca}$ is organized in two parts: the *SETUP* phase (the first phase) and the *SIMULATION* phase (the second phase). (We used capital letters for SETUP so that it is not confused with the setup phase of the CI (and the other related) game.)

SIMULATION PHASE

We introduce two *computational mappings*, τ_H^0 and τ_H^1, and two corresponding *key-renaming symbolic operators*, R_0 and R_1, as follows:

For $1 \le i \le n$, assign $\tau_H^0(x_i) = nc_i$, and if $i \neq u_s$, then $\tau_H^0(k_i) = pk_i$. Finally $\tau_H^0(k_{u_s}) = pk$.

For all $t \ge 1$, assign $\tau_H^0(k_{u_s,t}^{-1}) = fsk_{u_s}$, and $\tau_H^0(k_{h,t}^{-1}) = sk_h$, where $k_h^{-1} \notin \{k_{u_s}^{-1}, k_{u_{s+1}}^{-1} \ldots k_{u_l}^{-1}\}$. For $i \in \{s+1, s+2, \ldots, l\}$, assign $\tau_H^0(k_{u_i,t_1}^{-1}) = fsk_{u_i}$ and $\tau_H^0(k_{u_i,t_2}^{-1}) = sk_{u_i}$ for all $t_1 < on_i$ and $t_2 > on_i$.

For $i \in \{s+2, \ldots, l\}$, if $b_{i-1} = b_i$, assign $\tau_H^0(k_{u_i,on_i}^{-1}) = sk_{u_i}$; otherwise assign $\tau_H^0(k_{u_i,on_i}^{-1}) = fsk_{u_i}$. Finally let $\tau_H^0(k_{u_{s+1},on_{s+1}}^{-1}) = sk_{u_{s+1}}$. Now we define $\tau_H^1(s) = \tau_H^0(s)$ if $s \neq k_{u_{s+1},on_{s+1}}^{-1}$, and $\tau_H^1(k_{u_{s+1},on_{s+1}}^{-1}) = fsk_{u_{s+1}}$

For all $i \neq u_s$, $o \ge 1$ and $q \in \{0,1\}$, set $R_q[k_{i,o}^{-1}] = k_i^{-1}$ if $\tau_H^q(k_{i,o}^{-1}) = sk_i$, and $R_q(k_{i,o}^{-1}) = k'^{-1}_i$, if $\tau_H^q(k_{i,o}^{-1}) = fsk_i$. Also for all $q \in \{0,1\}$ and $o \ge 1$ set $R_q[k_{u_s,o}^{-1}] = k'^{-1}_{u_s}$.

Run $\mathcal{A}_{wstci}$; if $\mathcal{A}_{wstci}$ makes a query $encrypt(e, i)$, answer it as follows:

if $s = l$ or $k_{u_{s+1},on_{s+1}}^{-1}$ does not occur in e **then**

Algorithm 1 SETUP PHASE

1: Choose l numbers $u_0, \ldots, u_{l-1}$ from $\{0, 1, \ldots, n\}$ according to the following probability distribution:
$\Pr[u_i = 0] = \frac{1}{1+2nP^2}$
$\Pr[u_i = k] = \frac{2P^2}{1+2nP^2}$
for all $1 \leq k \leq n$ and $0 \leq i \leq l-1$.

2: Choose $u_l \leftarrow \{1, \ldots, n\}$.

3: Let s be *the smallest index* for which u_s is non-zero in the sequence $(u_0, \ldots, u_l)$. \\\\ In the simulation phase, pk, the public key parameterizing the *LOR encryption oracle*, will be associated with k_{u_s}, and all instances of $k_{u_s}^{-1}$ will be *faked*.

4: Sample $l - s$ bit values $b_{s+1}, \ldots, b_l \leftarrow \{0, 1\}$, and also set $b_{l+1} = 1$.

5: For each $u_i \in \{u_{s+1}, u_{s+2}, \ldots, u_l\}$, sample three parameters (on_i, ed_i, ed'_i) as follows:
$on_i, ed_i \leftarrow \{1, \ldots, P\}$
$ed'_i \leftarrow \{1, \ldots, ed_i\}$.

6: Generate $n-1$ pairs of public/private keys $(pk_1, sk_1), \ldots, (pk_{u_s-1}, sk_{u_s-1}), (pk_{u_s+1}, sk_{u_s+1}), \ldots, (pk_n, sk_n)$, and $l - s + 1$ private-key values $fsk_{u_s}, fsk_{u_{s+1}} \ldots fsk_{u_l}$ by running the key-generation algorithm (here fsk_{u_r} is going to be used a fake value for $k_{u_r}^{-1}$ in answering to some queries of the adversary).

7: Sample n nonce values $nc_1, \ldots, nc_n$ independently and uniformly at random from the *nonce space*.

8: return $E \leftarrow [\![\{e\}_{k_i}]\!]_{\tau_H^0}$ and add $(\{e\}_{k_i}, E)$ to *eval-exp*. (or, equivalently, sample E as $E \leftarrow [\![\{e\}_{k_i}]\!]_{\tau_H^1}$; the underlying distributions are identical.)
9: **else if** k_{u_s} does not encrypt $k_{u_{s+1}, on_{s+1}}^{-1}$ in $\{e\}_{k_i}$ **then**
10: output a random bit (chosen uniformly) and **HALT**. (In this situation, our *guessing* has failed.)
11: **else**
12: Let $\{e_1\}_{k_{u_s}}$ *be the smallest subexp of* $\{e\}_{k_i}$ *in which* e_1 *contains* $k_{u_{s+1}, on_{s+1}}^{-1}$. Now sample $E_{real} \leftarrow [\![e_1]\!]_{\tau_H^0}$ and $E_{fake} \leftarrow [\![e_1]\!]_{\tau_H^1}$, and let $e_{chal} \stackrel{\triangle}{=} e_1$. Construct E' as follows:
13: **if** $b_{s+1} = 0$ **then**
14: $E' = \mathcal{O}_{pk,b}^{\mathcal{E}}(E_{real}, E_{fake})$.
15: **else**
16: $E' = \mathcal{O}_{pk,b}^{\mathcal{E}}(E_{fake}, E_{real})$
17: **end if**
18: Using E' as the computational value of $\{e_1\}_{k_{u_s}}$, construct, E, a computational value for $\{e\}_{k_i}$ based on τ_H^0-evaluation (or τ_H^1-evaluation, equivalently), and return E to $\mathcal{A}_{wstci}$ and add $(\{e\}_{k_i}, E)$ to *eval-exp*.
19: **end if**

20: If $\mathcal{A}_{wstci}$ makes a *corruption* query $corrupt(i)$ do the following:
21: **if** $i \in \{u_s, u_{s+1}, \ldots u_l\}$ **then**
22: output a uniformly-chosen random bit and **HALT**. (This is again a situation in which the *guessing* fails. The guessing requires that *all* $k_{u_s}^{-1}, \ldots, k_{u_l}^{-1}$ remain *coinductively irrecoverable.*)
23: **else**
24: return sk_i and add (k_i^{-1}, sk_i) to *eval-exp*.
25: **end if**

26: **if** $\mathcal{A}_{wstci}$ makes the *challenge* query $challenge(k_j^{-1}, bs)$ **then**
27: **if** $k_j^{-1} \neq k_{u_l}^{-1}$ **then**
28: output a uniformly-chosen random bit and **HALT**. (This is again a failing situation; the guessing requires that $k_{u_l}^{-1}$ be the *challenge* key.)

else if $s < l$ **then**
  if $b_l = b_{l+1}$ **then**
    Return $C \leftarrow Enc(bs, pk_j)$
  else
    Choose $r \leftarrow \{0,1\}^{|bs|}$ and return $C \leftarrow Enc(r, pk_j)$.
  end if
else
  Return $\mathcal{O}^{\mathcal{E}}_{pk,b}(r, bs)$, where $r \leftarrow \{0,1\}^{|bs|}$
end if
end if

if $\mathcal{A}_{wstci}$ makes a *decryption* query $decrypt(c, i)$ **then**
  if any of the following events occur
  (i) (c, i) is coinductively visible in *eval-exp* (See the corresponding definition in Paragraph C.0.1);
  (ii) for some $(\{e\}_{k_j}, bs) \in$ *eval-exp*, where $k^{-1}_{u_{s+1},on_{s+1}}$ does not occur in $\{e\}_{k_j}$, it holds that $\{e\}_{k_j}$ has a subexp $\{e'\}_{k_p}$ where $R_0[\{e'\}_{k_p}]$ (or equivalently $R_1[\{e'\}_{k_p}]$) is type-1, and that c is the corresponding image of $\{e'\}_{k_p}$ in bs (see Subsection C.3 for the notation $R_0[\{e'\}_{k_p}]$);
  (iii) there exists $(\{e\}_{k_j}, bs) \in$ *eval-exp*, such that $k^{-1}_{u_{s+1},on_{s+1}}$ occurs in $\{e\}_{k_j}$ (i.e. $\{e_{chal}\}_{k_{u_s}}$ is a subexp of $\{e\}_{k_j}$), and that for some subexp $\{e'\}_{k_p}$ of $\{e\}_{k_j}$, it holds that c is the corresponding computational image of $\{e'\}_{k_p}$ in bs and that $\{e_{chal}\}_{k_{u_s}} \sqsubseteq \{e'\}_{k_p}$;
  (iv) there exists $(\{e\}_{k_j}, bs) \in$ *eval-exp*, such that $k^{-1}_{u_{s+1},on_{s+1}}$ occurs in $\{e\}_{k_j}$ and that for some subexp $\{e'\}_{k_p}$ of $\{e\}_{k_j}$, it holds that c is the concrete value of $\{e'\}_{k_p}$ in bs, $\{e'\}_{k_p}$ is disjoint from $\{e_{chal}\}_{k_{u_s}}$ and $R_0[\{e'\}_{k_p}]$ (or equivalently $R_1[\{e'\}_{k_p}]$) is type-1;
  (v) e_{chal} has a subexp $\{e'\}_{k_p}$ such that $R_0[\{e'\}_{k_p}]$ is type-1 and that c is the computational image of $\{e'\}_{k_p}$ in E_{real};
  (vi) e_{chal} has a subexp e' such that $R_1[\{e'\}_{k_p}]$ is type-1 and that c is the computational image of $\{e'\}_{k_p}$ in E_{fake}.
  then
    return $\perp$ (i.e. the *decryption* query is invalid)
  else
    if $i \neq u_s$ **then**
      return $Dec(c, sk_i)$
    else
      return $O^{\mathcal{E}}_{sk}(c)$
    end if
  end if
end if

if $\mathcal{A}_{wstci}$ makes a *subexp-testing* query $subexp(\{e\}_{k_i}, bs_i, \{e'\}_{k_j}, bs_j)$ **then**
  if $k^{-1}_{u_{s+1},on_{s+1}}$ does not occur in $\{e\}_{k_i}$ **then**
    if bs_j is the concrete value of $\{e'\}_{k_j}$ in bs_i **then**
      return 1
    else
      return 0
    end if
  else

if $k^{-1}_{u_{s+1},on_{s+1}}$ occurs in $\{e\}_{k_i}$ and either $\{e'\}_{k_j}$ has $\{e_{chal}\}_{k_{u_s}}$ as a subexp, or is disjoint from $\{e_{chal}\}_{k_{u_s}}$ **then**
  if bs_j is the concrete value of $\{e'\}_{k_j}$ in bs_i **then**
    return 1
  else
    return 0
  end if
else
  if bs_j is the concrete value of $\{e'\}_{k_j}$ in E_{real} or in E_{fake} **then**
    return 1
  else
    return 0
  end if
end if
end if
end if

If at any point during the simulation, any of the following events occurs, **HALT** and return a random bit (these spell out situations in which the simulation fails).

– For $s \le i \ne j \le l$, it holds that $u_i = u_j$
– For $s+1 \le i \le l$, $k_{u_{i-1}}$ does not encrypt $k^{-1}_{u_i,on_i}$

When $\mathcal{A}_{wstci}$ outputs his guess, if all the following conditions hold, output the adversary's guess; otherwise, return a bit uniformly at random.

– $v_{u_s} \xrightarrow{on_{s+1}} v_{u_{s+1}} \xrightarrow{on_{s+2}} \dots \xrightarrow{on_l} v_{u_l}$ is a strongly, coinductively continuable path.
– For every $s+1 \le j \le l$, it should hold that $\left|\Re_{\text{cont.}}(\xrightarrow{on_j} v_{u_j} \xrightarrow{on_{j+1}} v_{u_{j+1}} \xrightarrow{on_{j+2}} \dots \xrightarrow{on_l} v_{u_l})\right| = ed_j$. Moreover, $k_{u_{j-1}}$ is the ed'_jth lowest key among $\Re_{\text{cont.}}(\xrightarrow{on_j} v_{u_j} \xrightarrow{on_{j+1}} v_{u_{j+1}} \xrightarrow{on_{j+2}} \dots \xrightarrow{on_l} v_{u_l})$.

In the sequel, we verify the two required properties described earlier, namely *Efficiency* and *Correctness*, about $\mathbf{Strgy}_0$. First we provide a brief explanation about the presented algorithms.

C.3 Explanation

For all $1 \le i \le n$ we assume that k'_i is a fresh symbolic key and different from all k_i's and other k'_j's. The key-renaming operators R_0 and R_1 correspond to the two computational mappings τ^0_H and τ^1_H, in that, for $q \in \{0,1\}$, when τ^q_H gives a fake value to $k^{-1}_{i,o}$ (i.e. a value independent of pk_i, which is already given to the adversary as the public key value for k_i), R_q maps $k^{-1}_{i,o}$ to k'^{-1}_i. We assume that these two key-renaming operators, and all others that will be introduced henceforth, map any adversarial key to itself. We also use $R_q[e]$ to mean the expression obtained from e by replacing every k^{-1}_i with $R_q[k^{-1}_i]$. The only difference between the two worlds $\mathbf{Strgy}^0_0$ and $\mathbf{Strgy}^1_0$ is that one of them gives a fake value to $k^{-1}_{u_{s+1},on_{s+1}}$ and the other one gives the real value to it. (Recall that $\mathbf{Strgy}^b_0$ is the world created from $\mathbf{Strgy}_0$ when the CCA2-challenge bit is b.)

Roughly speaking, the distribution of replies given to an adversary $\mathcal{A}$'s *encryption*, *corruption* and *decryption* queries under $\mathbf{Strgy}_0$ is "close" to the distribution of replies that $\mathcal{A}$ would have received under the standard WSTCI game, for one of $h = 0$ or $h = 1$, when every $\mathcal{A}$'s query $encrypt(e, k_i)$ was replaced with $encrypt(R_h[e], k_i)$. (The point that to which of these two distributions $\mathbf{Strgy}_0$ corresponds to depends on the values of b_{s+1} and the CCA2-challenge bit

b; namely, if $b = b_{s+1}$ then it corresponds to $h = 0$ and otherwise it corresponds to $h = 1$. See line 13.) Let's call this statement (which we will formalize in subsequent sections) the *basic* statement. With this intuition in mind, we now briefly explain the ideas behind the decryption restrictions spelled out in lines proceeding 39.

The first condition reflects the decryption restriction from the CI game; namely $decrypt(c, i)$ is invalid, if there exists $(\{e\}_{k_j}, E_j) \in eval\text{-}exp$ such that $\{e\}_{k_j}$ has a subexp $\{e'\}_{k_i}$, which is encrypted in $\{e\}_{k_j}$ only under keys whose decryption keys are in $closure_c(eval\text{-}exp^1)$ and that c corresponds to the computational image of $\{e'\}_{k_i}$ in E_j. We remark here that if $\{e'\}_{k_i}$ is "coinductively reachable" in $\{e\}_{k_j}$ (using $eval\text{-}exp^1$), then for both $w = 0$ and $w = 1$, $R_w[\{e'\}_{k_i}]$ *is* coinductively reachable in $R_w[\{e\}_{k_j}]$. (This is because the key-renaming operators R_0 and R_1 only "rename" $k_{u_s}^{-1}, \ldots, k_{u_l}^{-1}$, all of which are required to remain coinductively irrecoverable by the guesswork—otherwise the guessing fails—implying that, in case that the guesswork does not fail, the sets of coinductively irrecoverable keys from $eval\text{-}exp^1$, $R_0[eval\text{-}exp^1]$ and $R_1[eval\text{-}exp^1]$ are all equivalent.) Hence the first decryption condition, informally speaking, "agrees" with the basic statement described above. Moreover it is easy to see that this condition is efficiently verifiable by the simulating adversary $\mathcal{A}_{cca}$. (This is because $\mathcal{A}_{cca}$ has the computational values of all private key symbols which are coinductively recoverable.)

The remaining items reflect the decryption restriction imposed under the WSTCI game (they were separated for the reason that we give below); namely $decrypt(c, i)$ is invalid, if there exists $(\{e\}_{k_j}, E_j) \in eval\text{-}exp$ such that $\{e\}_{k_j}$ has a type-1 subexp $\{e'\}_{k_i}$, where c is the corresponding computational image of $\{e'\}_{k_i}$ in E_j.

For Item (ii), assuming $(\{e\}_{k_j}, bs) \in eval\text{-}exp$, note that if $k_{u_{s+1}, on_{s+1}}^{-1}$ does not occur in $\{e\}_{k_j}$, then (1) *all* constituting bitstrings of bs is already known by $\mathcal{A}_{cca}$ (since the LOR encryption oracle is never called for constructing bs , as $k_{u_{s+1}, on_{s+1}}^{-1} \not\sqsubseteq \{e\}_{k_j}$), and (2) for any $e'' \sqsubseteq \{e\}_{k_j}$, it holds $R_0[e''] = R_1[e'']$. Thus this item can be efficiently implemented by $\mathcal{A}_{cca}$.

For Item (iii), again assuming $(\{e\}_{k_j}, bs) \in eval\text{-}exp$, if $k_{u_{s+1}, on_{s+1}}^{-1}$ occurs in $\{e\}_{k_j}$ and also $\{e_{chal}\}_{k_{u_s}} \sqsubseteq \{e'\}_{k_p} \sqsubseteq \{e\}_{k_j}$, then it holds that (1) both $R_0[\{e'\}_{k_p}]$ and $R_1[\{e'\}_{k_p}]$ are type-1 (this is because $\{e_{chal}\}_{k_{u_s}} \sqsubseteq \{e'\}_{k_p}$ and $R_0[k_{u_s,o}^{-1}] = R_1[k_{u_s,o}^{-1}] = k'^{-1}_{u_s}$, for all o's; namely, all occurrences of $k_{u_s}^{-1}$ as a plaintext receive a fake value), and (2) $\mathcal{A}_{cca}$ can efficiently check if a given c is the corresponding computational image of $\{e'\}_{k_p}$ in bs. (This is because $\{e_{chal}\}_{k_{u_s}} \sqsubseteq \{e'\}_{k_p}$ and the computational image of $\{e'\}_{k_p}$ is constructed from the computational image of $\{e_{chal}\}_{k_{u_s}}$, whose computational image is obtained through the LRO-encryption oracle; thus, $\mathcal{A}_{cca}$ knows the corresponding computational image of $\{e'\}_{k_p}$.) Now (1) implies that any c which corresponds to the computational image of such an $\{e'\}_{k_p}$ is an illegitimate ciphertext for decryption (according to the WSTCI decryption condition). In particular, E', which is returned by the LOR-encryption oracle (Line 13), will be an illegitimate ciphertext for decryption. (This is actually the main reason that we added a new decryption condition to the WSTCI game. That is, under the original CI game, E' may possibly be a legitimate ciphertext and if at any point during the simulation the decryption of E' is requested, then the simulation fails.)

For Item (iv), again assuming $(\{e\}_{k_j}, bs) \in eval\text{-}exp$, if $k_{u_{s+1}, on_{s+1}}^{-1}$ occurs in $\{e\}_{k_j}$ and also $\{e'\}_{k_p} \sqsubseteq \{e\}_{k_j}$ and $\{e'\}_{k_p}$ is disjoint from $\{e_{chal}\}_{k_{u_s}}$, then quite analogously to Item (ii), we can verify that we have (1) $R_0[\{e'\}_{k_p}] = R_1[\{e'\}_{k_p}]$, and (2) the computational image of $\{e'\}_{k_p}$ in bs is known by $\mathcal{A}_{cca}$.

Finally for Items (v) and (vi)—which describe the only remaining case; namely, ciphertexts that are constituents of the underlying plaintext of the challenge ciphertext E' and which should not be decrypted due to the WSTCI-decryption condition—since $\mathcal{A}_{cca}$ does not know the value of the underlying CCA2-challenge-bit, $\mathcal{A}_{cca}$ cannot determine, with certainty, whether a given ciphertext is the constituent of the underlying plaintext of E', and so he cannot determine if the ciphertext is illegitimate due to corresponding to the image of a type-1, subexp of e_{chal}. Therefore for this case, we exclude a larger set of ciphertexts; namely, if e_{chal} has a subexp $\{e'\}_{k_p}$ such that it holds that either $R_0[\{e'\}_{k_p}]$ is type-1 and c is the corresponding computational image of $\{e'\}_{k_p}$ in E_{real}, or $R_1[\{e'\}_{k_p}]$ is type-1 and c is the corresponding computational image of $\{e'\}_{k_p}$ in E_{fake}, then we determine c as an illegitimate ciphertext. Note that, this fact may cause the distribution of replies given to an adversary $\mathcal{A}$'s *encryption*, *corruption* and *decryption* queries under $\mathbf{Strgy}_0$ not to be completely identical to the distribution of replies that $\mathcal{A}$ would have received under the standard WSTCI game, for any $h = 0$ or $h = 1$, when every $\mathcal{A}$'s query $encrypt(e, k_i)$ is replaced with $encrypt(R_h[e], k_i)$. (i.e. The basic statement explained above may not "perfectly" hold.) However we will prove in subsequent subsections that this these two distributions are computationally identical. (Intuitively, the reason for this is that if we assume that E' is the encryption of, say, E_{real}, then E_{fake} does *not* participate in the replies given to adversary's queries, and in some sense, it is identically distributed as a ciphertext freshly sampled from the same distribution, that is $[\![e_1]\!]_{\tau_H^1}$. This means that the chances of any adversary to ever produce a ciphertext c that is the corresponding computational image of $\{e'\}_{k_p}$ in E_{fake} where $R_1[\{e'\}_{k_p}]$ is type-1 is "as good as" the chances of an adversary who can do this based on a freshly generated bitstring from $[\![e_1]\!]_{\tau_H^1}$, which we will prove is negligible.

C.4 Efficiency

Based on the previous Subsection C.3, it is fairly easy to verify that the whole simulation works in polynomial time. All checks related to *decryption* queries (i.e. starting at line 39) can be performed in polynomial time, and also the CCA2-challenge ciphertext (as explained in the previous subsection) will never be asked for decryption. It is also easy to verify that the other parts of the algorithm can be efficiency performed, and hence the whole simulation is efficiently implementable.

C.5 Correctness

We assume the following in the remainder of the paper.

Convention 2 *In all the theorems, lemmas, and propositions given henceforth, unless otherwise stated, we assume any used asymmetric or symmetric encryption scheme provides IND-CCA2 security. Therefore, we do not explicitly mention this assumption henceforth.*

First, we denote $\mathcal{A}_{wstci}$'s and $\mathcal{A}_{cca}$'s advantages in attacking the encryption scheme, in the sense of $l-$WSTCI and IND-CCA2, respectively, by:

$$\begin{aligned}
\Delta_{\mathrm{cca}}(\eta) &\stackrel{\triangle}{=} |\Pr[\mathcal{A}_{cca}(\eta) = 1 \mid b = 0] - \Pr[\mathcal{A}_{cca}(\eta) = 1 \mid b = 1]| \\
\Delta_{\mathrm{wstci}}(\eta) &\stackrel{\triangle}{=} \left|\Pr\left[\mathcal{A}_{wstci}^{\mathrm{WSTCI}}(\eta) = 1 \mid b = 0\right] - \Pr\left[\mathcal{A}_{wstci}^{\mathrm{WSTCI}}(\eta) = 1 \mid b = 1\right]\right| \qquad (6)
\end{aligned}$$

For the sake of readability, we omitted the mentions of $\mathcal{E}$, the encryption scheme for which WSTCI security is violated, and b in the above equality assignments, and wrote $\mathcal{A}_{wstci}^{\mathrm{WSTCI}}$ instead of $\mathcal{A}_{wstci}^{\mathrm{WSTCI}_b^{\mathcal{E}}}$. Also sometimes when it is necessary to differentiate between several challenge bits, we add as a subscript to them the name of the game they belong to. (e.g. b_{wstci} may be used to denote the challenge bit under the standard WSTCI game.) we denoted the challenge bit of the WSTCI game by b_{wstci} to make it distinct from b, the CCA2-challenge bit that $\mathcal{A}_{cca}$ is challenged to guess. We fix $\mathcal{E}$ in the remainder of this paper.

Our goal is to show the following:

Goal 1 $\Delta_{wstci}(\eta) > negl(\eta) \Rightarrow \Delta_{cca}(\eta) > negl(\eta)$

We borrow some notations from [46]. We let Bad denote the event that the simulation process of $\mathcal{A}_{cca}$ fails; namely, Bad occurs if any of the following events occur:

- $v_{u_s} \xrightarrow{on_{s+1}} v_{u_{s+1}} \xrightarrow{on_{s+2}} \ldots \xrightarrow{on_l} v_{u_l}$ is *not* a strongly, coinductively continuable path;
- For some $s+1 \leq j \leq l$, it holds that $\left|\Re_{\text{cont.}}(\xrightarrow{on_j} v_{u_j} \xrightarrow{on_{j+1}} v_{u_{j+1}} \xrightarrow{on_{j+2}} \ldots \xrightarrow{on_l} v_{u_l})\right| \neq ed_j$;
- For some $s+1 \leq j \leq l$, $k_{u_{j-1}}$ is *not* the ed'_jth lowest key among $\Re_{\text{cont.}}(\xrightarrow{on_j} v_{u_j} \xrightarrow{on_{j+1}} v_{u_{j+1}} \xrightarrow{on_{j+2}} \ldots \xrightarrow{on_l} v_{u_l})$; or
- $\mathcal{A}_{wstci}$ makes a *challenge* query $challenge(k_j^{-1}, bs)$ and $u_l \neq j$.

The first step of our proof is similar to that of [46]. Namely, we show that the probability Bad occurs given $b = 0$ is at most negligibly different from that given $b = 1$.

Observation 1 *We have*

$$|\Pr[Bad \mid b=0] - \Pr[Bad \mid b=1]| = negl(\eta).$$

It is not hard to see why Observation 1 is true. Given that $\mathcal{E}$ is IND-CCA2 secure (see Convention 2), if the statement of the observation fails to hold, then one can easily construct a second adversary $\mathcal{A}''$ who simulates $\mathcal{A}_{cca}$ and who outputs 1 exactly when the event Bad occurs. (If $\mathcal{A}_{cca}$ finishes his computation without Bad having occurred, $\mathcal{A}''$ outputs a uniformly-selected random bit.) Now it is easy to verify that $\mathcal{A}''$ violates the IND-CCA2 security assumption of $\mathcal{E}$.

Also from the very description of the SIMULATION process, one can easily verify the following statement.

Observation 2

$$\Pr[\mathcal{A}_{cca}(\eta) = 1 \mid Bad \wedge b = 0] \;=\; \Pr[\mathcal{A}_{cca}(\eta) = 1 \mid Bad \wedge b = 1] \;=\; \frac{1}{2}$$

The above observation holds because $\mathcal{A}_{cca}$ simply outputs a uniformly-selected random bit whenever Bad occurs.

We write $\overline{Bad}$ to denote the *complement* of Bad. Given Observations (1) and (2), we expand $\Delta_{\text{cca}}(\eta)$ as follows.

$$\begin{aligned}
\Delta_{\text{cca}}(\eta) &= \big|\Pr[\mathcal{A}_{cca}(\eta) = 1 \mid b = 0] - \Pr[\mathcal{A}_{cca}(\eta) = 1 \mid b = 1]\big| \\
&= \big|\Pr[\mathcal{A}_{cca}(\eta) = 1 \ \wedge \ Bad \mid b = 0] + \Pr[\mathcal{A}_{cca}(\eta) = 1 \ \wedge \ \overline{Bad} \mid b = 0] \\
&\quad - \Pr[\mathcal{A}_{cca}(\eta) = 1 \ \wedge \ Bad \mid b = 1] - \Pr[\mathcal{A}_{cca}(\eta) = 1 \ \wedge \ \overline{Bad} \mid b = 1]\big| \\
&= \big|(\Pr[\mathcal{A}_{cca}(\eta) = 1 \ \wedge \ \overline{Bad} \mid b = 0] - \Pr[\mathcal{A}_{cca}(\eta) = 1 \ \wedge \ \overline{Bad} \mid b = 1]|) + \\
&\quad + (\Pr[\mathcal{A}_{cca}(\eta) = 1 \mid Bad \wedge b = 0] \ \Pr[Bad \mid b = 0] - \Pr[\mathcal{A}_{cca}(\eta) = 1 \mid Bad \wedge b = 1] \ \Pr[Bad \mid b = 1])\big| \\
&= \big|\Pr[\mathcal{A}_{cca}(\eta) = 1 \ \wedge \ \overline{Bad} \mid b = 0] - \Pr[\mathcal{A}_{cca}(\eta) = 1 \ \wedge \ \overline{Bad} \mid b = 1] + \frac{1}{2} \ negl(\eta)\big| \\
&= \frac{1}{2} \ negl(\eta) + \frac{1}{2} \ \big|\Pr[\mathcal{A}_{cca}(\eta) = 1 \ \wedge \ \overline{Bad} \ \wedge \ b = 0] - \Pr[\mathcal{A}_{cca}(\eta) = 1 \ \wedge \ \overline{Bad} \ \wedge \ b = 1]\big|
\end{aligned}$$

Defining

$$\Delta_{\text{cca-sim}} \triangleq \big|\Pr\left[\mathcal{A}_{cca}(\eta) = 1 \ \wedge \ \overline{Bad} \ \wedge \ b = 0\right] - \Pr\left[\mathcal{A}_{cca}(\eta) = 1 \ \wedge \ \overline{Bad} \ \wedge \ b = 1\right]\big|, \qquad (7)$$

to prove the statement of Goal 1, it suffices to prove the statement of the following goal.

Goal 2 $\Delta_{wstci}(\eta) > negl(\eta)$*, then* $\Delta_{cca\text{-}sim} > negl(\eta)$

For $m \in \mathbb{N}$ we define $[m] \stackrel{\triangle}{=} \{1, \ldots, m\}$. We introduce a few more events. Let $\Psi^{(d_l, d_{l-1}, \ldots, d_{j+1})}_{(w_l, w_{l-1}, \ldots, w_{j+1})}$ denote the event that the following conditions are all satisfied:

(a) $u_0 = u_1 = \ldots u_{j-1} = 0$, $u_j \neq 0$ (i.e. $s = j$), and for $j+1 \leq i \leq l$, it holds that $indeg(v_{u_i}) = d_i$ and $on_i = w_i$,

(b) $v_{u_s} \xrightarrow{on_{s+1}} v_{u_{s+1}} \xrightarrow{on_{s+2}} \ldots \xrightarrow{on_l} v_{u_l}$ is a strongly, coinductively continuable path, and for $j+1 \leq i \leq l$, we have $\left| \Re_{cont}[\xrightarrow{on_i} v_{u_i} \xrightarrow{on_{i+1}} v_{u_{i+1}} \xrightarrow{on_{i+2}} \ldots, \xrightarrow{on_l} v_{u_l}] \right| = ed_i$, and $k_{u_{i-1}}$ is the ed_i''th lowest key among $\Re_{cont}[\xrightarrow{on_i} v_{u_i} \xrightarrow{on_{i+1}} v_{u_{i+1}} \xrightarrow{on_{i+2}} \ldots, \xrightarrow{on_l} v_{u_l}]$,

(c) if $\mathcal{A}_{wstci}$ makes a *challenge* query $challenge(k_j^{-1}, bs)$, then $j = u_l$.

Also we define $\Psi{-}d_l$ to be the event that the following conditions are satisfied:

(a) $u_0 = \cdots = u_{l-1} = 0$ and $indeg(v_{u_l}) = d_l$ (note that it always holds $u_l \neq 0$ and, by Assumption 1, $indeg(v_{u_l}) > 0$ given v_{u_l} is the challenge node),

(b) if $\mathcal{A}_{wstci}$ makes a *challenge* query $challenge(k_j^{-1}, bs)$, then $j = u_l$. (Note that this by itself implies that v_{u_l} is coinductively irrecoverable.)

We continue to introduce a few more types of events. We define $\Psi{-}max^{(d_l, d_{l-1}, \ldots, d_{j+1})}_{(w_l, w_{l-1}, \ldots, w_{j+1})}$ to be the event that $\Psi^{(d_l, d_{l-1}, \ldots, d_{j+1})}_{(w_l, w_{l-1}, \ldots, w_{j+1})}$ occurs and $indeg(v_{u_j}) = 0$. Dually, we define $\Psi{-}\overline{max}^{(d_l, d_{l-1}, \ldots, d_{j+1})}_{(w_l, w_{l-1}, \ldots, w_{j+1})}$ to be the event that $\Psi^{(d_l, d_{l-1}, \ldots, d_{j+1})}_{(w_l, w_{l-1}, \ldots, w_{j+1})}$ occurs and $indeg(v_{u_j}) \neq 0$. Also, for $b_l', b_{l-1}', \ldots, b_{j+1}' \in \{0, 1\}$, we let $\Psi^{(d_l, d_{l-1}, \ldots, d_{j+1})}_{(w_l, w_{l-1}, \ldots, w_{j+1})}[b_l', b_{l-1}', \ldots, b_{j+1}']$ denote the event that, (a) $\Psi^{(d_l, d_{l-1}, \ldots, d_{j+1})}_{(w_l, w_{l-1}, \ldots, w_{j+1})}$ happens, and (b) for every $j+1 \leq i \leq l$, it holds $b_i = b_i'$. (Recall that, b_i is the random bit value sampled in SETUP phase.)

We now expand $\Delta_{\text{cca-sim}}$ in terms of Ψ's as follows:

$$\begin{aligned}
\Delta_{\text{cca-sim}} &= \left| \Pr[\mathcal{A}_{cca}(\eta) = 1 \ \wedge \ \overline{Bad} \ \wedge \ b = 0] - \Pr[\mathcal{A}_{cca}(\eta) = 1 \ \wedge \ \overline{Bad} \ \wedge \ b = 1] \right| \\
&= \Bigg| \sum_{1 \leq k \leq l} \sum_{b_l', \ldots, b_k' \in \{0,1\}} \sum_{\substack{d_l \in [P] \\ w_l \in [d_l]}} \cdots \sum_{\substack{d_k \in [P] \\ w_k \in [d_l]}} \Big[\Pr[\mathcal{A}_{cca}(\eta) = 1 \ \wedge \ \Psi^{(d_l, \ldots, d_k)}_{(w_l, \ldots, w_k)}[b_l', \ldots, b_k'] \ \wedge \ b = 0] \\
&\qquad\qquad - \Pr[\mathcal{A}_{cca}(\eta) = 1 \ \wedge \ \Psi^{(d_l, \ldots, d_k)}_{(w_l, \ldots, w_k)}[b_l', \ldots, b_k'] \ \wedge \ b = 1] \Big] \\
&\quad + \sum_{d_l \in [P]} \Big[\Pr[\mathcal{A}_{cca}(\eta) = 1 \ \wedge \ \Psi - d_l \ \wedge \ b = 0] \\
&\qquad\qquad - \Pr[\mathcal{A}_{cca}(\eta) = 1 \ \wedge \ \Psi - d_l \ \wedge \ b = 1] \Big] \Bigg|
\end{aligned}$$

Recalling the definition of *strategy* explained earlier, if **ST** is an *strategy* used by a challenger to answer to $\mathcal{A}_{wstci}$'s queries, then as a simplifying notation we write $\Pr_{\mathbf{ST}}[E]$ to denote the probability that when $\mathcal{A}_{wstci}$ operates under **ST**, event E occurs. For example, $\Pr_{\mathbf{Strgy}_0}[indeg(v_{u_l}) > 3]$ denotes the probability that when $\mathcal{A}_{cca}$ uses strategy $\mathbf{Strgy}_0$ to answer to $\mathcal{A}_{wstci}$'s queries, it holds that $indeg(v_{u_l}) > 3$. Moreover, as mentioned earlier, we use **wstci** to refer to the *strategy* prescribed under the standard WSTCI game, and, for example, we may write $\Pr_{\mathbf{wstci}}[\mathcal{A}_{wstci}(\eta) = 1 \mid b = 0]$ instead of $\Pr\left[\mathcal{A}^{\text{WSTCI}}_{wstci}(\eta) = 1 \mid b = 0\right]$. (See Equation 6.)

For two events E_1 and E_2 we write $E_1 \equiv E_2$ if it holds that $\Pr_{\mathbf{Strgy}_0}[\mathcal{A}_{wstci}(\eta) = 1 \wedge E_1] = \Pr_{\mathbf{Strgy}_0}[\mathcal{A}_{wstci}(\eta) = 1 \wedge E_2]$. Also, we write $E_1 \cong E_2$ if it holds that $|\Pr_{\mathbf{Strgy}_0}[\mathcal{A}_{wstci}(\eta) = 1 \wedge E_1] - \Pr_{\mathbf{Strgy}_0}[\mathcal{A}_{wstci}(\eta) = 1 \wedge E_2]| = negl(\eta)$.

We define the following events:

$$\theta_0^{\text{wstci}} = \bigvee_{i=1}^{l} \bigvee_{\substack{d_l \in [P] \\ \vdots \\ d_i \in [P]}} \Psi\text{−}max_{(1,\ldots,1)}^{(d_l,\ldots,d_i)} [b_l = 0;\ \ \ldots;\ b_i = 0] \wedge b = 0$$

$$\theta_1^{\text{wstci}} = \bigvee_{i=1}^{l} \bigvee_{\substack{d_l \in [P] \\ \vdots \\ d_i \in [P]}} \Psi\text{−}max_{(1,\ldots,1)}^{(d_l,\ldots,d_i)} [b_l = 1;\ \ \ldots;\ b_i = 1] \wedge b = 1.$$

It is not hard to intuitively see that the view of $\mathcal{A}_{wstci}$ under $\mathbf{Strgy}_0$ and when θ_0 (resp. θ_1) occurs is quite similar to that under the WSTCI game when the challenger bit, b_{wstci}, is 0 (resp. is 1). We will formally show this in Corollary 1.

For any $n > 0$ and $H_1 = (h_1, \ldots, h_n) \in \mathbb{N}^n$ and $H_2 = (h'_1, \ldots, h'_n) \in \mathbb{N}^n$, we say $H_1 \leq H_2$ if for all $1 \leq i \leq n$ it holds that $h_i \leq h'_i$.

Lemma 4. *For all $1 \leq j \leq l$, $B_{j+1} = (b'_l, \ldots, b'_{j+1}) \in \{0,1\}^{l-j}$, $D_j = (d_l, d_{l-1}, \ldots, d_j) \in [P]^{l+1-j}$, and $W_j = (w_l, \ldots, w_{j+1}, w_j) \leq (d_l, \ldots, d_{j+1}, d_j - 1)$, if we define*

$$A^0_{total,j,B_{j+1},D_j,W_j} = \bigvee_{k=1}^{j-1} \bigvee_{\substack{d_{j-1} \in [P] \\ \vdots \\ d_{j-k} \in [P]}} \Psi\text{−}max_{(w_l,\ldots,w_{j+1},w_j,1,\ldots,1)}^{(d_l,\ldots,d_{j+1},d_j,d_{j-1},\ldots,d_{j-k})} [b'_l, \ldots, b'_{j+1}, 1, 0, \ldots, b_{j-k} = 0]\ \wedge\ b = 0$$
$$\bigvee \Psi\text{−}max_{(w_l,\ldots,w_{j+1},w_j)}^{(d_l,\ldots,d_{j+1},d_j)} [b'_l, \ldots, b'_{j+1}, 1]\ \wedge\ b = 0$$

$$A^1_{total,j,B_{j+1},D_j,W_j} = \bigvee_{k=1}^{j-1} \bigvee_{\substack{d_{j-1} \in [P] \\ \vdots \\ d_{j-k} \in [P]}} \Psi\text{−}max_{(w_l,\ldots,w_{j+1},w_j+1,1,\ldots,1)}^{(d_l,\ldots,d_{j+1},d_j,d_{j-1},\ldots,d_{j-k})} [b'_l, \ldots, b'_{j+1}, 1, 1, \ldots, b_{j-k} = 1] \wedge b = 1$$
$$\bigvee \Psi\text{−}max_{(w_l,\ldots,w_{j+1},w_j+1)}^{(d_l,\ldots,d_{j+1},d_j)} [b'_l, \ldots, b'_{j+1}, 1]\ \wedge\ b = 1$$

$$A'^0_{total,j,B_{j+1},D_j,W_j} = \bigvee_{k=1}^{j-1} \bigvee_{\substack{d_{j-1} \in [P] \\ \vdots \\ d_{j-k} \in [P]}} \Psi\text{−}max_{(w_l,\ldots,w_{j+1},w_j+1,1,\ldots,1)}^{(d_l,\ldots,d_{j+1},d_j,d_{j-1},\ldots,d_{j-k})} [b'_l, \ldots, b'_{j+1}, 0, 0, \ldots, b_{j-k} = 0] \wedge b = 0$$
$$\bigvee \Psi\text{−}max_{(w_l,\ldots,w_{j+1},w_j+1)}^{(d_l,\ldots,d_{j+1},d_j)} [b'_l, \ldots, b'_{j+1}, 0]\ \wedge\ b = 0$$

$$A'^1_{total,j,B_{j+1},D_j,W_j} = \bigvee_{k=1}^{j-1} \bigvee_{\substack{d_{j-1} \in [P] \\ \vdots \\ d_{j-k} \in [P]}} \Psi\text{−}max_{(w_l,\ldots,w_{j+1},w_j,1,\ldots,1)}^{(d_l,\ldots,d_{j+1},d_j,d_{j-1},\ldots,d_{j-k})} [b'_l, \ldots, b'_{j+1}, 0, 1, \ldots, b_{j-k} = 1] \wedge b = 1$$
$$\bigvee \Psi\text{−}max_{(w_l,\ldots,w_{j+1},w_j)}^{(d_l,\ldots,d_{j+1},d_j)} [b'_l, \ldots, b'_{j+1}, 0]\ \wedge\ b = 1$$

then, we have:

1.
$$A^{0}_{total,j,B_{j+1},D_j,W_j} \cong A^{1}_{total,j,B_{j+1},D_j,W_j}$$
2.
$$A'^{0}_{total,j,B_{j+1},D_j,W_j} \cong A'^{1}_{total,j,B_{j+1},D_j,W_j}$$

Lemma 5. *For all $2 \leq j \leq l$, $B_j = (b'_l, \ldots, b'_j) \in \{0,1\}^{l+1-j}$, $D_j = (d_l, d_{l-1}, \ldots, d_j) \in [P]^{l+1-j}$, and $W_j = (w_l, \ldots, w_{j+1}, w_j) \leq D_j$, if we define*

$$B^{0}_{total,j,B_j,D_j,W_j} = \mathit{\Psi}{-}\overline{max}^{(d_l,\ldots,d_j)}_{(w_l,\ldots,w_j)}[b'_l, \ldots, b'_j] \wedge b = 0$$

$$B^{1}_{total,j,B_j,D_j,W_j} = \bigvee_{k=2}^{j-1} \bigvee_{\substack{d_{j-1}\in[P] \\ \vdots \\ d_{j-k}\in[P]}} \mathit{\Psi}{-}max^{(d_l,\ldots,d_j,d_{j-1},d_{j-2},\ldots,d_{j-k})}_{(w_l,\ldots,w_j,d_{j-1},1\ldots,1)}[b'_l, \ldots, b'_j, b_{j-1} = 0, b_{j-2} = 1, \ldots, b_{j-k} = 1] \wedge b = 1$$
$$\bigvee_{d_{j-1}\in[P]} \mathit{\Psi}{-}max^{(d_l,\ldots,d_j,d_{j-1})}_{(w_l,\ldots,w_j,d_{j-1})}[b'_l, \ldots, b'_j, 0] \wedge b = 1$$

$$C^{0}_{total,l+1} = \bigvee_{d'_l\in[P]} \left(\mathit{\Psi} - d'_l\right) \wedge (b = 0)$$

$$C^{1}_{total,l+1} = \bigvee_{k=1}^{l-1} \bigvee_{\substack{d'_l\in[P] \\ \vdots \\ d'_{l-k}\in[P]}} \mathit{\Psi}{-}max^{(d'_l,d'_{l-1}\ldots,d'_{l-k})}_{(d'_l,1,\ldots,1)}[b_l = 0, b_{l-1} = 1, \ldots b_{l-k} = 1] \wedge b = 1$$
$$\bigvee_{d'_l\in[P]} \mathit{\Psi}{-}max^{(d'_l)}_{(d'_l)}[b_l = 0] \wedge b = 1$$

$$B'^{0}_{total,j,B_j,D_j,W_j} = \bigvee_{k=2}^{j-1} \bigvee_{\substack{d_{j-1}\in[P] \\ \vdots \\ d_{j-k}\in[P]}} \mathit{\Psi}{-}max^{(d_l,\ldots,d_j,d_{j-1},d_{j-2},\ldots,d_{j-k})}_{(w_l,\ldots,w_j,d_{j-1},1\ldots,1)}[b'_l, \ldots, b'_j, b_{j-1} = 1, b_{j-2} = 0, \ldots, b_{j-k} = 0] \wedge b = 0$$
$$\bigvee_{d_{j-1}\in[P]} \mathit{\Psi}{-}max^{(d_l,\ldots,d_j,d_{j-1})}_{(w_l,\ldots,w_j,d_{j-1})}[b'_l, \ldots, b'_j, 1] \wedge b = 0$$

$$B'^{1}_{total,j,B_j,D_j,W_j} = \mathit{\Psi}{-}\overline{max}^{(d_l,\ldots,d_j)}_{(w_l,\ldots,w_j)}[b'_l, \ldots, b'_j] \wedge b = 1$$

$$C'^{0}_{total,l+1} = \bigvee_{k=1}^{l-1} \bigvee_{\substack{d'_l\in[P] \\ \vdots \\ d'_{l-k}\in[P]}} \mathit{\Psi}{-}max^{(d'_l,d'_{l-1}\ldots,d'_{l-k})}_{(d'_l,1,\ldots,1)}[b_l = 1, b_{l-1} = 0, \ldots b_{l-k} = 0] \wedge b = 0$$
$$\bigvee_{d'_l\in[P]} \mathit{\Psi}{-}max^{(d'_l)}_{(d'_l)}[b_l = 1] \wedge b = 0$$

$$C'^{1}_{total,l+1} = \bigvee_{d'_l\in[P]} \left(\mathit{\Psi} - d'_l\right) \wedge (b = 1)$$

then we have:

1.
$$B^{0}_{total,j,B_j,D_j,W_j} \cong B^{1}_{total,j,B_j,D_j,W_j}$$

2.
$$B'^{0}_{total,j,B_j,D_j,W_j} \cong B'^{1}_{total,j,B_j,D_j,W_j}$$

3.
$$C^{0}_{total,l+1} \cong C^{1}_{total,l+1,\emptyset,\emptyset,\emptyset}$$

4.
$$C'^{0}_{total,l+1} \cong C'^{1}_{total,l+1,\emptyset,\emptyset,\emptyset}.$$

Now for $h \in \{0,1\}$ we define

$$\begin{aligned}
A^{h}_{total,j} &= \bigvee_{B_{j+1}\in\{0,1\}^{l-j}} \bigvee_{\substack{D_j=(d_l,\ldots,d_{j+1},d_j)\\ d_l\in[P]\\ \vdots \\ d_{j+1}\in[P]\\ d_j\in[P]}} \bigvee_{\substack{W_j=(w_l,\ldots,w_j)\\ w_l\in[d_l]\\ \vdots \\ w_{j+1}\in[d_{j+1}]\\ w_j\in[d_j-1]}} A^{h}_{total,j,B_{j+1},D_j,W_j} \\
A'^{h}_{total,j} &= \bigvee_{B_{j+1}\in\{0,1\}^{l-j}} \bigvee_{\substack{D_j=(d_l,\ldots,d_{j+1},d_j)\\ d_l\in[P]\\ \vdots \\ d_{j+1}\in[P]\\ d_j\in[P]}} \bigvee_{\substack{W_j=(w_l,\ldots,w_j)\\ w_l\in[d_l]\\ \vdots \\ w_{j+1}\in[d_{j+1}]\\ w_j\in[d_j-1]}} A'^{h}_{total,j,B_{j+1},D_j,W_j} \\
B^{h}_{total,j} &= \bigvee_{B_j\in\{0,1\}^{l+1-j}} \bigvee_{D_j\in[P]^{l+1-j}} \bigvee_{W_j\le D_j} B^{h}_{total,j,B_j,D_j,W_j} \\
B'^{h}_{total,j} &= \bigvee_{B_j\in\{0,1\}^{l+1-j}} \bigvee_{D_j\in[P]^{l+1-j}} \bigvee_{W_j\le D_j} B'^{h}_{total,j,B_j,D_j,W_j}.
\end{aligned}$$

Also we define

$$\begin{aligned}
\theta_0 &= \left(\bigvee_{j=1}^{l}\left[A^{0}_{total,j} \vee A'^{0}_{total,j}\right]\right) \vee \left(\bigvee_{j=2}^{l}\left[B^{0}_{total,j} \vee B'^{0}_{total,j}\right]\right) \vee \left(C^{0}_{total,l+1} \vee C'^{0}_{total,l+1}\right), \\
\theta_1 &= \left(\bigvee_{j=1}^{l}\left[A^{1}_{total,j} \vee A'^{1}_{total,j}\right]\right) \vee \left(\bigvee_{j=2}^{l}\left[B^{1}_{total,j} \vee B'^{1}_{total,j}\right]\right) \vee \left(C^{1}_{total,l+1} \vee C'^{1}_{total,l+1}\right).
\end{aligned}$$

Now we can concisely write the results of Lemmas 4 and 5 as follows:

Summary of Lemmas 4 and 5.

$$|\Pr[\mathcal{A}_{cca}(\eta) = 1 \wedge \theta_0] - \Pr[\mathcal{A}_{cca}(\eta) = 1 \wedge \theta_1]| = negl(\eta) \tag{8}$$

Note that, the assumption that "l is constant" is necessary to have Equation (8) derive from Lemmas 4 and 5. Under this assumption and the fact that the sum of polynomially-many negligible functions is negligible, the implication holds by a union bound.

Note that, in fact Lemmas 4 and 5 imply the following:

$$\left|\Pr_{\mathbf{Strgy}_0}[\mathcal{A}_{wstci}(\eta) = 1 \wedge \theta_0] - \Pr_{\mathbf{Strgy}_0}[\mathcal{A}_{wstci}(\eta) = 1 \wedge \theta_1]\right| = negl(\eta),$$

but since we have $\Pr_{\mathbf{Strgy}_0}[\mathcal{A}_{wstci}(\eta)=1 \wedge \theta_h] = \Pr[\mathcal{A}_{cca}(\eta)=1 \wedge \theta_h]$, for $h \in \{0,1\}$, Equation (8) follows. This latter follows from the fact that $\theta_h \wedge Bad = \emptyset$ and that we have

$$\Pr_{\mathbf{Strgy}_0}\left[\mathcal{A}_{wstci}(\eta)=1 \wedge \theta_h \wedge \overline{Bad}\right] = \Pr\left[\mathcal{A}_{cca}(\eta)=1 \wedge \theta_h \wedge \overline{Bad}\right].$$

The following corollary completes the proof of Lemma 2:

Corollary 1. *We have*

1.

$$\begin{aligned}\Delta_{\textit{cca-sim}} = \Big| & (\Pr[\mathcal{A}_{cca}(\eta)=1 \wedge \theta_0] - \Pr[\mathcal{A}_{cca}(\eta)=1 \wedge \theta_1]) \\ & + \left(\Pr\left[\mathcal{A}_{cca}(\eta)=1 \wedge \theta_0^{wstci}\right] - \Pr\left[\mathcal{A}_{cca}(\eta)=1 \wedge \theta_1^{wstci}\right]\right)\Big|\end{aligned}$$

2.

$$\Pr_{\textit{Strgy}_0}\left[\theta_0^{wstci}\right] = (\frac{1}{1+2nP^2})^{l+1} + negl(\eta)$$
$$\Pr_{\textit{Strgy}_0}\left[\theta_1^{wstci}\right] = (\frac{1}{1+2nP^2})^{l+1} + negl(\eta)$$

3.

$$\left|\Pr_{WSTCI}[\mathcal{A}_{wstci}(\eta)=1 \mid b=0] - \Pr_{\textit{Strgy}_0}\left[\mathcal{A}_{wstci}(\eta)=1 \mid \theta_0^{wstci}\right]\right| = negl(\eta)$$
$$\left|\Pr_{WSTCI}[\mathcal{A}_{wstci}(\eta)=1 \mid b=1] - \Pr_{\textit{Strgy}_0}\left[\mathcal{A}_{wstci}(\eta)=1 \mid \theta_1^{wstci}\right]\right| = negl(\eta)$$

4.

$$\Pr\left[\mathcal{A}_{cca}(\eta)=1 \mid \theta_0^{wstci}\right] = \Pr_{\textit{Strgy}_0}\left[\mathcal{A}_{wstci}(\eta)=1 \mid \theta_0^{wstci}\right]$$
$$\Pr\left[\mathcal{A}_{cca}(\eta)=1 \mid \theta_1^{wstci}\right] = \Pr_{\textit{Strgy}_0}\left[\mathcal{A}_{wstci}(\eta)=1 \mid \theta_1^{wstci}\right].$$

5.

$$\Delta_{wstci}(\eta) > negl(\eta) \Rightarrow \Delta_{\textit{cca-sim}}(\eta) > negl(\eta)$$

Item 5 of above Corollary is exactly what we need to prove for Lemma 2 (i.e. Goal 2). In the remainder of this section, we give the proof of both of the above lemmas and corollary.

C.6 Proof of Lemma 4

The proofs of both parts of the lemma are entirely similar, so we prove part (1) of the lemma. Let $j, b'_l, \ldots, b'_{j+1}, d_l, \ldots, d_{j+1}, d_j, w_l, \ldots, w_{j+1}, w_j$ be as the statement of the lemma and be fixed. Also, for the sake of better readability, since j is fixed, we will drop subscripts j, B_{j+1}, D_j and W_j from the associated events described in this part of the lemma and write, for example, A^0_{total}. Therefore, we have

$$A^0_{total} = \bigvee_{k=1}^{j-1} \bigvee_{\substack{d_{j-1}\in[P]\\ \vdots \\ d_{j-k}\in[P]}} \Psi\text{−}max^{(d_l,\ldots,d_{j+1},d_j,d_{j-1},\ldots,d_{j-k})}_{(w_l,\ldots,w_{j+1},w_j,1,\ldots,1)}[b'_l,\ldots,b'_{j+1},1,0,\ldots,b_{j-k}=0] \ \wedge \ b=0$$

$$\bigvee \Psi\text{−}max^{(d_l,\ldots,d_{j+1},d_j)}_{(w_l,\ldots,w_{j+1},w_j)}[b'_l,\ldots,b'_{j+1},1] \ \wedge \ b=0$$

$$A^1_{total} = \bigvee_{k=1}^{j-1} \bigvee_{\substack{d_{j-1}\in[P]\\ \vdots \\ d_{j-k}\in[P]}} \Psi\text{−}max^{(d_l,\ldots,d_{j+1},d_j,d_{j-1},\ldots,d_{j-k})}_{(w_l,\ldots,w_{j+1},w_j+1,1,\ldots,1)}[b'_l,\ldots,b'_{j+1},1,1,\ldots,b_{j-k}=1] \wedge b=1$$

$$\bigvee \Psi\text{−}max^{(d_l,\ldots,d_{j+1},d_j)}_{(w_l,\ldots,w_{j+1},w_j+1)}[b'_l,\ldots,b'_{j+1},1] \ \wedge \ b=1$$

Now defining $\Delta_0 = \Pr_{\mathbf{Strgy}_0}[A^0_{total}]$ and $\Delta_1 = \Pr_{\mathbf{Strgy}_0}[A^1_{total}]$, we will prove the following two claims.

Claim 1 $|\Delta_0 - \Delta_1| = negl(\eta)$

Claim 2 *If $\Delta_0 > negl(\eta)$ and $\Delta_1 > negl(\eta)$, then*

$$|\Pr_{\boldsymbol{Strgy}_0}[A_{wstci}(\eta)=1 \mid A^0_{total}] - \Pr_{\boldsymbol{Strgy}_0}[A_{wstci}(\eta)=1 \mid A^1_{total}]| = negl(\eta)$$

We will separately prove the above two claims. Note that Claims 1 and 2 together complete the proof of this part of the lemma. This is because we need to prove that:

$$\left|\Delta_0 \times \Pr_{\mathbf{Strgy}_0}[A_{wstci}(\eta)=1 \mid A^0_{total}] - \Delta_1 \times \Pr_{\mathbf{Strgy}_0}[A_{wstci}(\eta)=1 \mid A^1_{total}]\right| = negl(\eta), \quad (9)$$

and if, say, $\Delta_0 = negl(\eta)$, then by Claim 1, we obtain $\Delta_1 = negl(\eta)$, and the above equality then obviously holds. On the other hand, if both $\Delta_0 > negl(\eta)$ and $\Delta_1 > negl(\eta)$, then Claims 1 and 2 imply equation 9. We omit the details.

First we establish some notations that will be used in the proofs of both claims above. For $1 \le k \le j-1$, $D = (d_{j-1},\ldots,d_{j-k}) \in \{[P]\}^k$, we define the following events:

$$\begin{aligned}
A^0_{k,D} &= \Psi\text{−}max^{(d_l,\ldots,d_{j+1},d_j,d_{j-1},\ldots,d_{j-k})}_{(w_l,\ldots,w_{j+1},w_j,1,\ldots,1)}[b_l=b'_l,\ldots,b_{j+1}=b'_{j+1},b_j=1,b_{j-1}=0,\ldots,b_{j-k}=0] \wedge b=0\\
A^0_{0,\emptyset} &= \Psi\text{−}max^{(d_l,\ldots,d_{j+1},d_j)}_{(w_l,\ldots,w_{j+1},w_j)}[b_l=b'_l,\ldots,b_{j+1}=b'_{j+1},b_j=1] \wedge b=0\\
A^1_{k,D} &= \Psi\text{−}max^{(d_l,\ldots,d_{j+1},d_j,d_{j-1},\ldots,d_{j-k})}_{(w_l,\ldots,w_{j+1},w_j+1,1,\ldots,1)}[b_l=b'_l,\ldots,b_{j+1}=b'_{j+1},b_j=1,b_{j-1}=1,\ldots,b_{j-k}=1] \wedge b=1\\
A^1_{0,\emptyset} &= \Psi\text{−}max^{(d_l,\ldots,d_{j+1},d_j)}_{(w_l,\ldots,w_{j+1},w_j+1)}[b_l=b'_l,\ldots,b_{j+1}=b'_{j+1},b_j=1] \wedge b=1\\
E^0_{k,D} &= \Psi\text{−}max^{(d_l,\ldots,d_{j+1},d_j,d_{j-1},\ldots,d_{j-k})}_{(w_l,\ldots,w_{j+1},w_j,1,\ldots,1)}\\
E^0_{0,\emptyset} &= \Psi\text{−}max^{(d_l,\ldots,d_{j+1},d_j)}_{(w_l,\ldots,w_{j+1},w_j)}\\
E^1_{k,D} &= \Psi\text{−}max^{(d_l,\ldots,d_{j+1},d_j,d_{j-1},\ldots,d_{j-k})}_{(w_l,\ldots,w_{j+1},w_j+1,1,\ldots,1)}\\
E^1_{0,\emptyset} &= \Psi\text{−}max^{(d_l,\ldots,d_{j+1},d_j)}_{(w_l,\ldots,w_{j+1},w_j+1)}
\end{aligned}$$

We also define

$$E^0_{total} = \bigvee_{k=0}^{j-1} \bigvee_{D\in[P]^k} E^0_{k,D}$$

$$E^1_{total} = \bigvee_{k=0}^{j-1} \bigvee_{D\in[P]^k} E^1_{k,D}$$

Note that we have:

$$\Pr_{\mathbf{Strgy}_0}[A^0_{k,D}] = \frac{1}{2}^{l-j+k+2} \cdot \Pr_{\mathbf{Strgy}_0}[E^0_{k,D} \mid b_l = b'_l;\ \dots;\ b_{j+1} = b'_{j+1};\ b_j = 1;\ b_{j-1} = 0;\ \dots;\ b_{j-k} = 0;\ b = 0]$$

$$\Pr_{\mathbf{Strgy}_0}[A^1_{k,D}] = \frac{1}{2}^{l-j+k+2} \cdot \Pr_{\mathbf{Strgy}_0}[E^1_{k,D} \mid b_l = b'_l;\ \dots;\ b_{j+1} = b'_{j+1};\ b_j = 1;\ b_{j-1} = 1;\ \dots;\ b_{j-k} = 1;\ b = 1]$$

We now expand Δ_0 and Δ_1 in terms of the probability of events $E^0_{k,D}$'s and $E^1_{k,D}$'s as follows:

$$\Delta_0 = \sum_{k=0}^{j-1} \sum_{D\in\{[P]\}^k} (\frac{1}{2})^{l-j+k+2} \Pr_{\mathbf{Strgy}_0}[E^0_{k,D} \mid b_l = b'_l;\ \dots;\ b_{j+1} = b'_{j+1};\ b_j = 1;\ b_{j-1} = 0;\ \dots;\ b_{j-k} = 0;\ b = 0] \tag{10}$$

$$\Delta_1 = \sum_{k=0}^{j-1} \sum_{D\in\{[P]\}^k} (\frac{1}{2})^{l-j+k+2} \Pr_{\mathbf{Strgy}_0}[E^1_{k,D} \mid b_l = b'_l;\ \dots;\ b_{j+1} = b'_{j+1};\ b_j = 1;\ b_{j-1} = 1;\ \dots;\ b_{j-k} = 1;\ b = 1] \tag{11}$$

C.6.1 Proof of Claim 1 We introduce a new replying strategy, denoted by $\mathbf{Strgy}_j$, which is a modified version of $\mathbf{Strgy}_0$ for responding to adversary's queries. (Henceforth we may also refer to an strategy as a game, and use both names interchangeably.) In the SETUP phase:

- Variables $u_0, \dots, u_l$ are sampled in the same manner as in $\mathbf{Strgy}_0$, and denoting by s the smallest index where $u_s \neq 0$, if $s \geq j$, then the execution is terminated (a failed situation). Otherwise, for each $s < t \leq l$ we sample three parameters (on_t, ed_t, ed'_t), again, in exactly the same manner as in $\mathbf{Strgy}_0$ (under $\mathbf{Strgy}_j$ this sequence of triples of random values play no role in responses made to adversary's queries; they are included in the game as we are to formulate some events depending on them).
- We generate n pairs of public/private key pairs $(pk_1, sk_1), \dots, (pk_n, sk_n)$, fake private keys $fsk_{u_s}, fsk_{u_{s+1}} \dots, fsk_{u_l}$, and n nonces $nc_1, \dots, nc_n$, all independently at random from the same distributions used in $\mathbf{Strgy}_0$. (We stress here that, unlike $\mathbf{Strgy}_0$, under $\mathbf{Strgy}_j$ we sample a pair of public/private keys for all key pairs.)

In the Simulation phase: (we first define $b'_j = 1$)

- An *encryption* query $encrypt(e, i)$ is answered by $E \leftarrow [\![e]\!]_{\tau_0}$, where τ_0 is defined as follows: $\tau_o(x_i) = nc_i$, $\tau_0(k_i) = pk_i$ for $1 \leq i \leq n$. Moreover, if $k_i^{-1} \notin \{k^{-1}_{u_j}, k^{-1}_{u_{j+1}}, \dots, k^{-1}_{u_l}\}$, we assign $\tau_0(k^{-1}_{i,o}) = sk_i$, for all o's, and for $k^{-1}_{u_j}$, we assign $\tau_0(k^{-1}_{u_{j,o_1}}) = fsk_{u_j}$ and $\tau_0(k^{-1}_{u_{j,o_2}}) = sk_{u_j}$, for

all $1 \le o_1 \le w_j$ and $o_2 \ge w_j+1$. Finally, for all $t \in \{j+1,\ldots,l\}$, we define $\tau_0(k^{-1}_{u_t,o_3}) = fsk_{u_t}$ and $\tau_0(k^{-1}_{u_t,o_4}) = sk_{u_t}$, for all $o_3 < w_t$ and $o_4 > w_t$, and if $b'_t = b'_{t-1}$, then $\tau_0(k^{-1}_{u_t,w_t}) = sk_{u_t}$, and, $\tau_0(k^{-1}_{u_t,w_t}) = fsk_{u_t}$, otherwise.
- A *corruption* query $corrupt(i)$ is handled by returning sk_i.
- For *decryption* queries, we first define a key-renaming function, R, as follows: for all $k^{-1}_{i,o}$ we have $R[k^{-1}_{i,o}] = k^{-1}_i$ if $\tau_0(k^{-1}_{i,o}) = sk_i$, and $R[k^{-1}_{i,o}] = k'^{-1}_i$ if $\tau_0(k^{-1}_{i,o}) = fsk_i$. Now a query $decrypt(c,i)$ is answered by $Dec(c, sk_i)$, if none of the following events occur, and it is answered by $\perp$, otherwise: (a) (c,i) is coinductively visible in *eval-exp*, and (b) for some $(\{e\}_{k_h}, bs) \in$ *eval-exp* it holds that there exists a subexp $\{e'\}_{k_p}$ of $\{e\}_{k_h}$ such that $R[\{e'\}_{k_p}]$ is type-1 and that c is the computational image of $\{e'\}_{k_p}$ in bs.
- A *subexp-testing* query $subexp(\{e\}_{k_i}, bs_i, \{e'\}_{k_j}, bs_j)$ is answered by 1 if bs_j is the computational image of $\{e'\}_{k_j}$ in bs_i, and answered by 0, otherwise.

For $\mathbf{Strgy}_j$, we may also define events of the form $\Psi^{(d'_l,d'_{l-1},\ldots,d'_{r+1})}_{(w'_l,w'_{l-1},\ldots,w'_{r+1})}$. We claim that

Lemma 6. *For every $D = [P]^k$, it holds that:*

$$| \Pr_{\textbf{\textit{Strgy}}_0}[E^0_{k,D} \mid b_l = b'_l;\ \ldots;\ b_{j+1} = b'_{j+1};\ b_j = 1;\ b_{j-1} = 0;\ \ldots;\ b_{j-k} = 0;\ b = 0] - \Pr_{\textbf{\textit{Strgy}}_j}[E^0_{k,D}]| = negl(\eta)$$

$$| \Pr_{\textbf{\textit{Strgy}}_0}[E^1_{k,D} \mid b_l = b'_l;\ \ldots;\ b_{j+1} = b'_{j+1};\ b_j = 1;\ b_{j-1} = 1;\ \ldots;\ b_{j-k} = 1;\ b = 1] - \Pr_{\textbf{\textit{Strgy}}_j}[E^1_{k,D}]| = negl(\eta)$$

Lemma 7.

$$\sum_{k=0}^{j-1} \sum_{D\in[P]^k} (\frac{1}{2})^{l-j+k+2} \Pr_{\textbf{\textit{Strgy}}_j}[E^0_{k,D}] = \sum_{k=0}^{j-1} \sum_{D\in[P]^k} (\frac{1}{2})^{l-j+k+2} \Pr_{\textbf{\textit{Strgy}}_j}[E^1_{k,D}]$$

Note that Lemma 6 and Lemma 7, together, complete the proof of Claim 1. Now we give the proofs of both lemmas:

C.6.1.1 ***Proof of Lemma 6*** We prove the first part of the lemma; the proof of the second part follows similarly. We introduce another security game (i.e. replying strategy) $\mathbf{Strgy}'_0$, as follows. First in the SETUP phase, we do the following steps:

- we first generate the sequence $\{u_i\}_{0\le i\le l}$ in the same manner as that in $\mathbf{Strgy}_0$, and denoting by s the smallest index where $u_s \ne 0$, if $s \ge j$ then the execution is terminated (failed situation). To sample b_i's, we assign $b_r = b'_r$ for all $j+1 \le r \le l$, $b_j = 1$, and assign $b_i = 0$ for all $s \le i \le j-1$. (Note that, differently from $\mathbf{Strgy}_0$, under this game we introduce and assign a bit value to b_s as well.)
- We sample $\{(on_r, ed_r, ed'_r)\}_{s+1\le t\le l}$, and $\{pk_i, sk_i\}_{1\le i\le n}$, and $\{nc_i\}_{1\le i\le n}$, and $\{fsk_{u_i}\}_{s\le i\le l}$, all independently at random from the same distributions as used in $\mathbf{Strgy}_0$. (Again note that we sample (pk_{u_s}, sk_{u_s}).)

  In the SIMULATION phase:

- we respond to an *encryption* query, $encrypt(e,i)$, by $E \leftarrow enc(\llbracket e\rrbracket_{\tau_0}, pk_i)$, where τ_0 is defined as follows: For $1 \le i \le n$, assign $\tau_0(k_i) = pk_i$ and $\tau_0(x_i) = nc_i$. For all $t \ge 1$, assign $\tau_0(k^{-1}_{u_s,t}) = fsk_{u_s}$, and $\tau_0(k^{-1}_{h,t}) = sk_h$, where $k^{-1}_h \notin \{k^{-1}_{u_s}, k^{-1}_{u_{s+1}} \ldots k^{-1}_{u_l}\}$. For $i \in \{s+1, s+2, \ldots, l\}$, assign $\tau_0(k^{-1}_{u_i,t_1}) = fsk_{u_i}$ and $\tau_0(k^{-1}_{u_i,t_2}) = sk_{u_i}$ for all $t_1 < on_i$ and $t_2 > on_i$, and if $b_{i-1} = b_i$, assign $\tau_0(k^{-1}_{u_i,on_i}) = sk_{u_i}$, and $\tau_0(k^{-1}_{u_i,on_i}) = fsk_{u_i}$, otherwise.

– A *corruption* query $corrupt(i)$ is handled by returning sk_i.
– For *decryption* queries, we again similarly define a key-renaming function, R_0, as follows: for all $k_{i,o}^{-1}$ we have $R_0[k_{i,o}^{-1}] = k_i^{-1}$ if $\tau_0(k_{i,o}^{-1}) = sk_i$, and $R_0[k_{i,o}^{-1}] = {k'}_i^{-1}$ if $\tau_0(k_{i,o}^{-1}) = fsk_i$. Now a query $decrypt(c, i)$ is answered by $Dec(c, sk_i)$ if none of the following events occurs, and by $\perp$, otherwise: (a) (c, i) is coinductively visible in *eval-exp*, or (b) for some $(\{e\}_{k_h}, bs) \in$ *eval-exp* it holds that there exists a subexp $\{e'\}_{k_p}$ of $\{e\}_{k_h}$ such that $R_0[\{e'\}_{k_p}]$ is type-1 and that c is the computational image of $\{e'\}_{k_p}$ in bs.
– For a *subexp-testing* query $subexp(\{e\}_{k_i}, bs_i, \{e'\}_{k_j}, bs_j)$, it proceeds as $\mathbf{Strgy}_j$; namely, it is answered by 1 if bs_j is the computational image of $\{e'\}_{k_j}$ in bs_i, and answered by 0, otherwise.

Now Lemma 6 derives from the following two claims.

Claim 3

$$\Pr_{\boldsymbol{Strgy}_j}[E^0_{k,D}] = \Pr_{\boldsymbol{Strgy}'_0}[E^0_{k,D}]$$

Claim 4

$$|\Pr_{\boldsymbol{Strgy}_0}[E^0_{k,D} \mid b_l = b'_l;\ \ldots;\ b_{j+1} = b'_{j+1};\ b_j = 1;\ b_{j-1} = 0;\ \ldots;\ b_{j-k} = 0;\ b = 0] - \Pr_{\boldsymbol{Strgy}'_0}[E^0_{k,D}]\,| = negl(\eta)$$

We separately provide the proof of each of the above two claims.

Proof of claim 3: First recall that

$$E^0_{k,D} \;=\; \Psi{-}max^{(d_l,\ldots,d_{j+1},d_j,d_{j-1},\ldots,d_{j-k})}_{(w_l,\ldots,w_{j+1},w_j,1,\ldots,1)},\ \text{where } D = (d_{j-1}, \ldots, d_{j-k})$$

Let E be the following event:

$$\begin{aligned} E ::= \;& (u_0 = \cdots = u_{j-k-3} = u_{j-k-2} = 0) \wedge (u_{j-k-1} \neq 0) \\ & \wedge (on_{j-k} = on_{j-k+1} = \cdots = on_{j-1} = 1) \wedge (on_j = w_j) \wedge (on_{j+1} = w_{j+1}) \wedge \cdots \wedge (on_l = w_l) \end{aligned}$$

Now we can write

$$\Pr_{\mathbf{Strgy}'_0}[E^0_{k,D}] = \Pr_{\mathbf{Strgy}'_0}[E^0_{k,D} \mid E] \cdot \Pr_{\mathbf{Strgy}'_0}[E] \tag{12}$$

$$\Pr_{\mathbf{Strgy}_j}[E^0_{k,D}] = \Pr_{\mathbf{Strgy}_j}[E^0_{k,D} \mid E] \cdot \Pr_{\mathbf{Strgy}_j}[E] \tag{13}$$

The above two equalities follow from the fact that $\Pr_{\mathbf{Strgy}'_0}[E^0_{k,D} \mid \overline{E}] = \Pr_{\mathbf{Strgy}_j}[E^0_{k,D} \mid \overline{E}] = 0$. Moreover it is easy to see that $\Pr_{\mathbf{Strgy}'_0}[E] = \Pr_{\mathbf{Strgy}_j}[E]$; this is because occurrence/non-occurrence of E is determined right after the execution of the SETUP phase (it is in fact independent of how SIMULATION phase proceeds); thus, it occurs with equal probabilities in both games. Finally it is not hard to verify that, provided that E occurs, an adversary has identical view when run under $\mathbf{Strgy}'_0$ or $\mathbf{Strgy}_j$, implying that $\Pr_{\mathbf{Strgy}'_0}[E^0_{k,D} \mid E] = \Pr_{\mathbf{Strgy}_j}[E^0_{k,D} \mid E]$, and this completes the proof of the claim. □

Proof of Claim 4: We need to prove

$$|\Pr_{\mathbf{Strgy}_0}[E^0_{k,D} \mid b_l = b'_l;\ \ldots;\ b_{j+1} = b'_{j+1};\ b_j = 1;\ b_{j-1} = 0;\ \ldots;\ b_{j-k} = 0;\ b = 0] - \Pr_{\mathbf{Strgy}'_0}[E^0_{k,D}]\,| = negl(\eta).$$

We define

$$p_1 = \Pr_{\mathbf{Strgy}_0}[E^0_{k,D} \mid b_l = b'_l;\ \ldots;\ b_{j+1} = b'_{j+1};\ b_j = 1;\ b_{j-1} = 0;\ \ldots;\ b_{j-k} = 0;\ b = 0].$$

We first add a "failing condition" to the SIMULATION phase of $\mathbf{Strgy}'_0$, making it closer to that of $\mathbf{Strgy}_0$, and prove that the affected change does not modify the probability of $E^0_{k,D}$'s occurrence (under $\mathbf{Strgy}'_0$). In the new version of $\mathbf{Strgy}'_0$, we build τ_0 and answer to adversary's queries similarly as before, with the only addition that in the sequence of adversary's *encryption* queries, if it happens that k_{u_s} does not encrypt $k^{-1}_{u_{s+1},on_{s+1}}$, then we halt the execution; otherwise if $encrypt(e,i)$ is the query for which $k^{-1}_{u_{s+1},on_{s+1}}$ occurs in e, assuming that $\{e_1\}_{u_{k_s}}$ is the smallest subexp of e with e_1 containing $k^{-1}_{u_{s+1},on_{s+1}}$, we introduce $e_{chal} = e_1$ and let $M \leftarrow [\![e_1]\!]_{\tau_1}$, where $\tau_1(s) = \tau_0(s)$ if $s \neq k^{-1}_{u_{s+1},on_{s+1}}$ and $\tau_1(k^{-1}_{u_{s+1},on_{s+1}}) = q$, where $q = \{fsk_{u_{s+1}}, sk_{u_{s+1}}\} - \tau_0[k^{-1}_{u_{s+1},on_{s+1}}]$. We then continue replying to adversary's queries exactly the same as before. (We note that the value of M and expression e_{chal}, in this game, are not used in the process of answering to adversary's queries; we have introduced them to later define an event based on them.) Also, for simplicity, we divide the execution of the game into two phases; the first phase, called the *preceding phase*, ends as soon as the adversary issues and receives the reply to the *encryption* query $encrypt(e,i)$, where $k^{-1}_{u_{s+1},on_{s+1}}$ occurs in e, and the second phase, called *succeeding phase*, begins (if the game does not halt due to the above condition) right after the completion of the preceding phase. It is fairly easy to see that the probability of $E^0_{k,D}$'s occurrence under the two versions of $\mathbf{Strgy}'_0$ is equal; thus in the rest of the proof, by $\mathbf{Strgy}'_0$ we refer to the new version of the game.

We now introduce another game, $\mathbf{Strgy}''_0$ by slightly changing $\mathbf{Strgy}'_0$, and prove that the probability of $E^0_{k,D}$ occurring under $\mathbf{Strgy}''_0$ is equal to p_1. The SETUP phase of $\mathbf{Strgy}''_0$ is performed exactly the same as that of $\mathbf{Strgy}'_0$, and in the SIMULATION phase, we introduce τ_0 and answer to adversary's *encryption* queries exactly like that of $\mathbf{Strgy}'_0$. Just for *decryption* and *subexp-testing* queries, when the game proceeds to the succeeding phase (i.e. after τ_1 is introduced and $M \to [\![e_{chal}]\!]_{\tau_1}$ is sampled), we do the following:

- *decryption* queries: we add another restriction to the set of invalid *decryption* queries (i.e. those replied with $\perp$). For this, we first define a key-renaming function, R_1 (which corresponds to τ_1), as follows: for all $k^{-1}_{i,o}$ we have $R_1[k^{-1}_{i,o}] = k^{-1}_i$ if $\tau_1(k^{-1}_{i,o}) = sk_i$, and $R_1[k^{-1}_{i,o}] = k'^{-1}_i$ if $\tau_1(k^{-1}_{i,o}) = fsk_i$. Now for a *decryption* query $decrypt(c,i)$ if it holds that e_{chal} has a subexp $\{e'\}_{k_p}$ such that $R_1[\{e'\}_{k_p}]$ is type-1 and that c is the computational image of $\{e'\}_{k_p}$ in M, then the query is answered by $\perp$.
- *subexp-testing* queries: We add one more condition to those of $\mathbf{Strgy}'_0$ for **subexp-queries**. If in the succeeding phase for $subexp(\{e\}_{k_i}, bs_i, \{e'\}_{k_j}, bs_j)$ it holds that $\{e'\}_{k_j}$ is a subexp of e_{chal} and that bs_j is the corresponding computational image of $\{e'\}_{k_j}$ in M, then we return 1.

The rest of the game is exactly the same as that of $\mathbf{Strgy}'_0$.

Now from the very description of $\mathbf{Strgy}''_0$ (especially considering how b_i's are assigned values under $\mathbf{Strgy}''_0$), the following is immediately derived:

$$p_1 = \Pr_{\mathbf{Strgy}''_0}[E^0_{k,D}]$$

Thus it remains to prove that the difference of probabilities of $E^0_{k,D}$'s occurrence under $\mathbf{Strgy}'_0$ and $\mathbf{Strgy}''_0$ is negligible. To show this, we let *Failed-Dec* describe the event that, in the succeeding phase of the game, either a *decryption* query is answered by $\perp$ due to the newly added condition, or a *subexp-testing* query is answered by 1 due to the newly added condition. That is, *Failed-Dec* occurs if any of the following events occur:

(a) For a *decryption* query $decrypt(c,i)$ in the succeeding phase, it holds that c is the computational image of $\{e'\}_{k_p}$ in M, where $\{e'\}_{k_p} \sqsubseteq e_{chal}$ $R_1[\{e'\}_{k_p}]$ is type-1;

(b) For a *subexp-testing* query $subexp(\{e\}_{k_i}, bs_i, \{e'\}_{k_j}, c)$ in the succeeding phase it holds that c is the computational image of $\{e'\}_{k_j}$ in M, where $\{e'\}_{k_j} \sqsubseteq e_{chal}$.

Note that *Failed-Dec* is well-defined under both $\mathbf{Strgy}'_0$ and $\mathbf{Strgy}''_0$, and, moreover,

$$\Pr_{\mathbf{Strgy}'_0}[\overline{Failed-Dec}] = \Pr_{\mathbf{Strgy}''_0}[\overline{Failed-Dec}].$$

Also from the very descriptions of $\mathbf{Strgy}'_0$ and $\mathbf{Strgy}''_0$, it immediately follows that:

$$\Pr_{\mathbf{Strgy}'_0}[E^0_{k,D} \mid \overline{Failed-Dec}] = Pr_{\mathbf{Strgy}''_0}[E^0_{k,D} \mid \overline{Failed-Dec}]$$

Therefore, to complete the proof of Claim (4), it is sufficient to show that $\Pr_{\mathbf{Strgy}'_0}[\textit{Failed-Dec}]$ is negligible. We show this in the following lemma.

Lemma 8.

$$\Pr_{\textbf{\textit{Strgy}}'_0}[\textit{Failed-Dec}] = negl(\eta)$$

Proof of Lemma 8: Suppose, to the contrary, that there exists an adversary $\mathcal{A}_1$ who interacts with the challenger under $\mathbf{Strgy}'_0$, and for whom *Failed-Dec* occurs. We first define two random variables which wil help to simplify Failed-Dec. We let C_i denote the random variable that corresponds to the ith ciphertext that the adversary issues for decryption in the *succeeding* phase. Also by C'_i we denote the *fourth* argument of the ith *subexp-testing* query issued in the succeeding phase. (i.e. In $subexp(\{e\}_{k_i}, bs_i, \{e'\}_{k_j}, bs_j)$ the fourth argument is bs_j.) First we can easily see that both random variables in each of the pairs (M, C_i) and (M, C'_i) are independent; namely, if $m_1, m_2 \in [\![e_{chal}]\!]_{\tau_1}$, then $\Pr[C_i = c \mid M = m_1] = \Pr[C_i = c \mid M = m_2]$ and $\Pr[C'_i = c \mid M = m_1] = \Pr[C'_i = c \mid M = m_2]$. This is because M is kept secret (i.e. M takes part in no replies given to adversary's queries), and, therefore, the random value of M does not affect the view of the adversary.

To conclude the proof of this lemma, if such an adversary $\mathcal{A}_1$ exists, then there exists an adversary $\mathcal{A}_2$ who *wins* with a non-negligible probability in the following experiment. However, the next proposition will show the computational impossibility of this fact. The experiment, called **exper-1**, runs in the following steps: (a) The adversary specifies a public keys symbol k; (b) (pk, sk) is sampled by running $Gen(\eta)$, and a computational mapping τ is initialized by $(\tau(k), \tau(k^{-1})) = (pk, sk)$, and the computational values for other symbols under τ are chosen by the adversary; (c) the adversary introduces an expression $\{e\}_k$, where $k^{-1} \not\sqsubseteq e$; (d) the challenger generates $E \leftarrow [\![\{e\}_k]\!]_\tau$ and keeps E secret from the adversary; and (e) the adversary (*given access to* τ) wins if he correctly guesses E.

Before presenting the proposition, we briefly explain how $\mathcal{A}_2$ is constructed from $\mathcal{A}_1$. Recalling the random variable M and computational mapping τ_1 defined earlier, the main observation here is that, since M takes no part in any replies given to adversary's queries, for any subexp e_w of e_{chal}, if M_w is the corresponding computational image of e_w in M, then the chances of any adversary to correctly guess M_w during the game is the same as his chances of correctly guessing the value of E_w, where E_w is freshly generated from $[\![e_w]\!]_{\tau_1}$. This in particular implies that for any query $decrypt(c, i)$ issued in the succeeding phase, the probability that c hits the corresponding computational image of e_w in M is equal to the probability that c hits E_w. Using this observation we explain how $\mathcal{A}_2$ works. Suppose $\mathcal{A}_1$ in the succeeding phase produces $decrypt(c, i)$, where, with a nonnegligible probability, it holds that c is the computational image of $\{e'\}_{k_p}$ in M, where $\{e'\}_{k_p} \sqsubseteq e_{chal}$ and $R_1[\{e'\}_{k_p}]$ is type-1. (The other case where *Failed-Dec* occurs due to a *subexp-testing* query is proved using a similar argument.) Since $R_1[\{e'\}_{k_p}]$ is type-1, it has a subexpression $\{e''\}_{k_q} \sqsubseteq \{e'\}_{k_p}$, where $k_q^{-1} \not\sqsubseteq R_1[\{e''\}_{k_q}]$. Now $\mathcal{A}_2$ plays as a challenger under $\mathbf{Strgy}'_0$ with the following slight modifications: (a) $\mathcal{A}_2$ first guesses k_q and specifies it as the public key in **Exper1**, and if (pk, sk) are the values sampled for (k_q, k_q^{-1}), he uses pk as the value for k_q and sk as the real value for k_q^{-1} in $\mathbf{Strgy}'_0$, and accordingly he initializes τ_0 and τ_1. Finally based on τ_0 and τ_1, he initializes his **Exper-1**-computational-mapping τ; (b) $\mathcal{A}_2$ no longer samples M; and (c) $\mathcal{A}_2$ guesses the decryption query number corresponding to $decrypt(c, i)$ which makes *Failed-Dec* occur. Now using τ_1, he decrypts c to to obtain c'', the (possible) corresponding computational image of $\{e''\}_{k_q}$ in c. Now from the observation given above and from the assumption that *Failed-Dec* occurs with a non-negligible probability, it follows that $(\{e''\}_{k_q}, c'')$ makes $\mathcal{A}_2$ wins in his **Exper1** experiment with a non-negligible probability.

Proposition 1 *Suppose $k^{-1} \not\sqsubseteq e$ and τ is a computational mapping, where $(\tau(k), \tau(k^{-1}) \leftarrow Gen(\eta)$. For other symbols, τ samples their computational values in an arbitrary, efficient manner. If $\mathcal{E}$ is IND-CCA2 secure and $E \leftarrow [\![\{e\}_k]\!]^{\mathcal{E}}_{\tau}$, then for every adversary $\mathcal{A}$ having access to τ, the probability that $\mathcal{A}$ guesses E is negligible.*

Proof of Proposition 1: Define $p_{max}(\eta) = \max_{c \in S} \Pr[E = c]$, for $S = \{0, 1\}^{|[\![\{e\}_k]\!]^{\mathcal{E}}_{\tau}|}$, where the probability is computed over the random coins used to form τ and to sample E from $[\![\{e\}_k]\!]^{\mathcal{E}}_{\tau}$. First we can show that since the encryption scheme is IND-CCA2 secure, p_{max} should be a negligible function. (If this does not hold, one can show that there exists an adversary $\mathcal{B}$ who can distinguish between encryptions under $\tau(k)$ of two messages $m_1 \leftarrow [\![e]\!]^{\mathcal{E}}_{\tau}$ and an arbitrary message $m_2 \notin [\![e]\!]^{\mathcal{E}}_{\tau}$. Note that this immediately enables an IND-CPA attack since $k^{-1} \not\sqsubseteq e$.) Now let the random variable E_1 be defined as $E_1 = out(\mathcal{A}^{\tau}(\eta))$, where $out(\mathcal{A}^{\tau}(\eta))$ denotes the output of $\mathcal{A}$, having access to τ, and given η as input. Note that, since E is not given to the adversary, for $c_1, c_2 \in S$, we have $\Pr[E_1 = c \mid E = c_1] = \Pr[E_1 = c \mid E = c_2]$, which implies that $\Pr[E = c_1 \wedge E_1 = c_2] = \Pr[E = c_1]. \Pr[E_1 = c_2]$. Now the probability of success is:

$$\sum_{c \in S} \Pr[E = c \wedge E_1 = c] = \sum_{c \in S} \Pr[E = c]. \Pr[E_1 = c] \leq p_{max} \sum_{c \in S} \Pr[E_1 = c] = p_{max}$$

and this completes the proof. □

Now from the above proposition, the proof of Lemma 8, and consequently Claim 4, follows. □

□

Now Claim 3 and Claim 4 imply the result of lemma 6.

□

C.6.1.2 ***Proof of Lemma 7:*** We need to show that

$$\sum_{k=0}^{j-1} \sum_{D \in [P]^k} (\frac{1}{2})^{l-j+k+2} \Pr_{\mathbf{Strgy}_j}[E^0_{k,D}] = \sum_{k=0}^{j-1} \sum_{D \in [P]^k} (\frac{1}{2})^{l-j+k+2} \Pr_{\mathbf{Strgy}_j}[E^1_{k,D}].$$

Recall that for $0 \leq k \leq j-1$

$$\begin{aligned} E^0_{k,D} &= \Psi\text{-}max^{(d_l,\ldots,d_{j+1},d_j,d_{j-1},\ldots,d_{j-k})}_{(w_l,\ldots,w_{j+1},w_j,1,\ldots,1)}, \text{ where } D = (d_{j-1}, \ldots, d_{j-k}) \\ E^1_{k,D} &= \Psi\text{-}max^{(d_l,\ldots,d_{j+1},d_j,d_{j-1},\ldots,d_{j-k})}_{(w_l,\ldots,w_{j+1},w_j+1,1,\ldots,1)}, \text{ where } D = (d_{j-1}, \ldots, d_{j-k}). \end{aligned}$$

We first establish some notation. Let $G(\mathcal{A}_{wstci})$ be the key-graph after the execution of $\mathbf{Strgy}_j$ and assume that $\mathcal{P} = v_{p_r} \xrightarrow{a_{r-1}} v_{p_{r-1}} \xrightarrow{a_{r-2}} \ldots \xrightarrow{a_1} v_{p_1}$ is a coinductively, continuable path in $G(\mathcal{A}_{wstci})$. For $a_r \leq indeg(v_{p_r})$ and $i \leq \left|\Re_{\text{cont.}}(\xrightarrow{a_r} \mathcal{P}])\right|$, we define $prev(\xrightarrow{a_r} \mathcal{P}, i)$ to be the ith-deepest key in $\Re_{\text{cont.}}(\xrightarrow{a_r} \mathcal{P})$. In case that $\mathcal{P}$ is not a coinductively, continuable path, we define $\Re_{\text{cont.}}(\xrightarrow{a_r} \mathcal{P})$, for every $a_1 \geq 1$.

Let E_{basis} be the event that after the execution of $\mathbf{Strgy}_j$ all the following conditions hold:

1. $E^0_{basis} ::= \forall i \in \{j, \ldots, l\}, indeg(v_{u_i}) = d_i$,
2. $E^1_{basis} ::= \forall i \in \{j, \ldots, l-1\}, on_{i+1} = w_{i+1}$,
3. $E^2_{basis} ::= \forall i \in \{j, \ldots, l-1\}, ed_{i+1} = \Re_{\text{cont.}}(\xrightarrow{w_{i+1}} v_{u_{i+1}} \xrightarrow{w_{i+2}} v_{u_{i+2}} \xrightarrow{w_{i+3}} \ldots \xrightarrow{w_l} v_{u_l})$,
4. $E^3_{basis} ::= \forall i \in \{j, \ldots, l\}, v_{u_i} = prev\left(\xrightarrow{w_{i+1}} v_{u_{i+1}} \xrightarrow{w_{i+2}} v_{u_{i+2}} \xrightarrow{w_{i+3}} \ldots \xrightarrow{w_l} v_{u_l}, ed'_{i+1}\right)$,
5. E^4_{basis} ::= if $\mathcal{A}_{wstci}$ makes a *challenge* query $challenge(k_j^{-1}, bs)$ (i.e. private-key challenge), then $j = u_l$.

That is, $E_{basis} = E^0_{basis} \wedge \cdots \wedge E^4_{basis}$. Informally speaking, E_{basis} is the "largest" event which is a subset of both $E^0_{k,D}$ and $E^1_{k,D}$. For $b \in \{0,1\}$, letting

$$total_b = \sum_{k=0}^{j-1} \sum_{D \in [P]^k} (\frac{1}{2})^{l-j+k+2} \Pr_{\mathbf{Strgy}_j}[E^b_{k,D}],$$

we prove that:

$$total_0 = total_1 = (\frac{1}{2})^{l-j+1} \cdot (\frac{1}{1+2nP^2})^{j} \cdot \Pr_{\mathbf{Strgy}_j}[E_{basis}] \tag{14}$$

To prove (14), we show that

$$total_0 = (\frac{1}{2})^{l-j+1} \cdot (\frac{1}{1+2nP^2})^{j} \cdot \Pr_{\mathbf{Strgy}_j}[E_{basis}], \tag{15}$$

and the equality for $total_1$ is also obtained similarly, and this concludes the proof.

Let $\mathcal{P}_c = v_{u_j} \xrightarrow{w_{j+1}} v_{u_{j+1}} \xrightarrow{w_{j+2}} \ldots \xrightarrow{w_l} v_{u_l}$, which is a coinductively continuable path if E_{basis} occurs. (Subscript "c" stands as an initial for "common".) We define random variable V_0 as follows: after the execution of $\mathbf{Strgy}_j$, if $E_{\mathbf{basis}}$ does not occur, then $V_0 = -1$, otherwise, V_0 is determined as follows: pick a node, v_{i_0}, uniformly at random from $\Re_{\text{cont.}}(\xrightarrow{w_j} \mathcal{P}_c)$ and if $indeg(v_{i_0}) = 0$ then $V_0 = 0$, otherwise, let $c = 0$, and:

while $indeg(v_{i_c} > 0)$ **do**
  Pick $v_{i_{c+1}}$ uniformly at random from $\Re_{\text{cont.}}(\xrightarrow{1} v_{i_c} \xrightarrow{1} \ldots \xrightarrow{1} v_{i_0} \xrightarrow{w_j} \mathcal{P}_c)$
  $c \rightarrow c+1$
end while
$V_0 = c$

Now letting

$$sum_k = \sum_{D \in [P]^k} \Pr_{\mathbf{Strgy}_j}[E^0_{k,D}],$$

we prove that for all $0 \leq k \leq j-1$, we have

$$sum_k = 2^{k+1} \cdot (\frac{1}{1+2nP^2})^j \cdot \Pr_{\mathbf{Strgy}_j}[E_{basis}] \cdot \Pr_{\mathbf{Strgy}_j}[V_0 = k \mid E_{basis}] \tag{16}$$

From this, (15) follows. (This is because, from the assumption that the diameter of the hidden subgraph is at most l, we obtain that $V_0 = -1$ or $0 \leq V_0 \leq j-1$.) Thus we focus on proving (16).

For $i \in \{0, \ldots, j-1\}$, we define the following events:

$$
\begin{aligned}
A_i^0 &= (s \le i) \wedge (\forall i_1 : i+1 \le i_1 \le j-1, on_{i_1} = 1) \wedge (\forall i_2 \in \{j, \ldots, l\} : on_{i_2} = w_{i_2}) \\
A_i^1 &= \Big(\forall i_1 : i+1 \le i_1 \le j-1, ed_{i_1} = \Big|\Re_{\text{cont.}}(\xrightarrow{1} v_{u_{i_1}} \xrightarrow{1} \ldots \xrightarrow{1} v_{u_{j-1}} \xrightarrow{w_j} v_{u_j} \xrightarrow{w_{j+1}} \ldots \xrightarrow{w_l} v_{u_l})\Big|\Big) \\
&\quad \wedge \Big(\forall i_2 \in \{j, \ldots, l\}, ed_{i_2} = \Big|\Re_{\text{cont.}}(\xrightarrow{w_{i_2}} v_{u_{i_2}} \xrightarrow{w_{i_2+1}} \ldots \xrightarrow{w_l} v_{u_l})\Big|\Big) \\
A_i^2 &= \Big(\forall i_1 : i+1 \le i_1 \le j-1, v_{u_{(i_1-1)}} = prev\Big(\xrightarrow{1} v_{u_{i_1}} \xrightarrow{1} \ldots \xrightarrow{1} v_{u_{j-1}} \xrightarrow{w_j} v_{u_j} \xrightarrow{w_{j+1}} \ldots \xrightarrow{w_l} v_{u_l}, ed'_{i_1}\Big)\Big) \\
&\quad \wedge \Big(\forall i_2 \in \{j, \ldots, l\}, v_{u_{(i_2-1)}} = prev\Big(\xrightarrow{w_{i_2}} v_{u_{i_2}} \xrightarrow{w_{i_2+1}} v_{u_{(i_2+1)}} \xrightarrow{w_{i_2+2}} \ldots \xrightarrow{w_l} v_{u_l}, ed'_{i_2}\Big)\Big)
\end{aligned}
$$

We define $A_i = A_i^0 \wedge A_i^1 \wedge A_i^2 \wedge E^0_{basis}$. We can now expand sum_k as follows:

$$
\begin{aligned}
sum_k &= \Pr_{\mathbf{Strgy}_j}\Big[(u_0 = \cdots = u_{j-k-2} = 0) \wedge A_{j-k-1} \wedge indeg(v_{u_{(j-k-1)}}) = 0\Big] \qquad (17) \\
&= (\frac{1}{1+2nP^2})^{j-k-1} \cdot \Pr_{\mathbf{Strgy}_j}\Big[A_{j-k-1} \wedge indeg(v_{u_{(j-k-1)}}) = 0\Big]
\end{aligned}
$$

The reason that (17) holds is that all u_i's are sampled independently from each other. Thus to prove 16, it is sufficient to prove the following lemma:

Lemma 9. *For all $0 \le k \le j-1$, it holds*

$$
\Pr_{\textbf{\textit{Strgy}}_j}\Big[A_{j-k-1} \wedge indeg(v_{u_{(j-k-1)}}) = 0\Big] = (\frac{2}{1+2nP^2})^{k+1} \cdot \Pr_{\textbf{\textit{Strgy}}_j}[E_{basis}] \cdot \Pr_{\textbf{\textit{Strgy}}_j}[V_0 = k \mid E_{basis}]
$$

Proof of Lemma 9.

For $j-k \le i \le j-1$, we define $A'_i = A_i \wedge (indeg(v_{u_i}) > 0)$, and we also define $A'_{j-k-1} = A_{j-k-1} \wedge \big(indeg(v_{u_{j-k-1}} = 0)\big)$. Thus we want to compute the probability of A'_{j-k-1} under $\mathbf{Strgy}_j$. We expand the probability as follows:

$$
\begin{aligned}
\Pr_{\mathbf{Strgy}_j}\big[A'_{j-k-1}\big] &= \Pr_{\mathbf{Strgy}_j}\big[A'_{j-k-1} \mid A'_{j-k}\big] \cdot \Pr_{\mathbf{Strgy}_j}\big[A'_{j-k} \mid A'_{j-k+1}\big] \qquad (18) \\
&\cdots \Pr_{\mathbf{Strgy}_j}\big[A'_{j-2} \mid A'_{j-1}\big] \cdot \Pr_{\mathbf{Strgy}_j}\big[A'_{j-1} \mid E_{basis}\big] \cdot \Pr_{\mathbf{Strgy}_j}\big[E_{basis}\big]. \qquad (19)
\end{aligned}
$$

For $j-k \le i \le j-2$, we can write:

$$
\begin{aligned}
\Pr_{\mathbf{Strgy}_j}\big[A'_i \mid A'_{i+1}\big] &= \Pr_{\mathbf{Strgy}_j}\big[A_i \wedge indeg(v_{u_i}) > 0 \mid A'_{i+1}\big] \\
&= \Pr_{\mathbf{Strgy}_j}\big[A_i \mid A'_{i+1}\big] \cdot \Pr_{\mathbf{Strgy}_j}\big[indeg(v_{u_i}) > 0 \mid A_i \wedge A'_{i+1}\big] \\
&= \Pr_{\mathbf{Strgy}_j}\big[A_i \mid A'_{i+1}\big] \cdot \Pr_{\mathbf{Strgy}_j}\big[indeg(v_{u_i}) > 0 \mid A_i\big]
\end{aligned}
$$

Similarly, we can show that

$$
\begin{aligned}
\Pr_{\mathbf{Strgy}_j}\big[A'_{j-k-1} \mid A'_{j-k}\big] &= \Pr_{\mathbf{Strgy}_j}\big[A_{j-k-1} \mid A'_{j-k}\big] \cdot \Pr_{\mathbf{Strgy}_j}\big[indeg(v_{u_{j-k-1}}) = 0 \mid A_{j-k-1}\big] \\
\Pr_{\mathbf{Strgy}_j}\big[A'_{j-1} \mid E_{basis}\big] &= \Pr_{\mathbf{Strgy}_j}\big[A_{j-1} \mid E_{basis}\big] \cdot \Pr_{\mathbf{Strgy}_j}\big[indeg(v_{u_{j-1}}) > 0 \mid A_{j-1}\big]
\end{aligned}
$$

Now if we define

$$\begin{aligned}
product_1 = &\Pr_{\mathbf{Strgy}_j}\left[indeg(v_{u_{j-1}}) > 0 \mid A_{j-1}\right] \cdot \Pr_{\mathbf{Strgy}_j}\left[indeg(v_{u_{j-2}}) > 0 \mid A_{j-2}\right] \cdot \\
&\cdots \Pr_{\mathbf{Strgy}_j}\left[indeg(v_{u_{j-k}}) > 0 \mid A_{j-k}\right] \cdot \Pr_{\mathbf{Strgy}_j}\left[indeg(v_{u_{j-k-1}}) = 0 \mid A_{j-k-1}\right] \\
product_2 = &\Pr_{\mathbf{Strgy}_j}\left[A_{j-1} \mid E_{basis}\right] \cdot \Pr_{\mathbf{Strgy}_j}\left[A_{j-2} | \ A'_{j-1}\right] \cdots \Pr_{\mathbf{Strgy}_j}\left[A_{j-k-1} | \ A'_{j-k}\right],
\end{aligned}$$

we have

$$\Pr_{\mathbf{Strgy}_j}\left[A'_{j-k-1}\right] = product_1 \cdot product_2 \cdot \Pr_{\mathbf{Strgy}_j}\left[E_{basis}\right]. \tag{20}$$

For $j - k - 1 \leq h \leq j - 2$, we can write:

$$\begin{aligned}
\Pr_{\mathbf{Strgy}_j}\left[A_h | \ A'_{h+1}\right] &= \Pr_{\mathbf{Strgy}_j}\left[(on_{h+1} = 1) \wedge \left(ed_{h+1} = \left|\Re_{\text{cont.}}(\xrightarrow{1} v_{u_{h+1}} \xrightarrow{1} \ldots \xrightarrow{1} v_{u_{j-1}} \xrightarrow{w_j} \mathcal{P}_c)\right|\right)\right. \\
&\qquad \left. \wedge \left(v_{u_h} = prev(\xrightarrow{1} v_{u_{h+1}} \xrightarrow{1} \ldots \xrightarrow{1} v_{u_{j-1}} \xrightarrow{w_j} \mathcal{P}_c, ed'_{h+1})\right)\right] \\
&= \frac{1}{P} \cdot \frac{1}{P} \cdot \frac{2P^2}{1 + 2nP^2} \\
&= \frac{2}{1 + 2nP^2}.
\end{aligned} \tag{21}$$

Also, similarly, we can show that

$$\Pr_{\mathbf{Strgy}_j}\left[A_{j-1} \mid E_{basis}\right] = \frac{2}{1 + 2nP^2}.$$

For $j - k \leq i \leq j - 2$ we have:

$$\begin{aligned}
&= \Pr_{\mathbf{Strgy}_j}\left[indeg(v_{u_i}) > 0 \mid A_i\right] \\
&= \Pr_{\mathbf{Strgy}_j}\left[V_0 > j - i - 1 \mid V_0 > j - i - 2\right].
\end{aligned} \tag{22}$$

The last equation follows from the very description of V_0 and the fact that under $\mathbf{Strgy}_j$, $u_0, \ldots, u_{j-1}$ does not affect the randomness of replies given to adversary's queries. Similarly we have

$$\begin{aligned}
\Pr_{\mathbf{Strgy}_j}\left[indeg(v_{u_{j-k-1}}) = 0 \mid A_{j-k-1}\right] &= \Pr_{\mathbf{Strgy}_j}\left[V_0 = k \mid V_0 > k - 1\right] \\
\Pr_{\mathbf{Strgy}_j}\left[indeg(v_{u_{j-1}}) > 0 \mid A_{j-1}\right] &= \Pr_{\mathbf{Strgy}_j}\left[V_0 > 0 \mid E_{basis}\right].
\end{aligned}$$

Thus we have

$$\begin{aligned}
product_1 &= \Pr_{\mathbf{Strgy}_j}\left[V_0 = k \mid V_0 > k - 1\right] \cdot \Pr_{\mathbf{Strgy}_j}\left[V_0 > k - 1 \mid V_0 > k - 2\right] \\
&\qquad \cdots \Pr_{\mathbf{Strgy}_j}\left[V_0 > 1 \mid V_0 > 0\right] \cdot \Pr_{\mathbf{Strgy}_j}\left[V_0 > 0 \mid E_{basis}\right] \\
&= \frac{\Pr_{\mathbf{Strgy}_j}\left[V_0 = k\right]}{\Pr_{\mathbf{Strgy}_j}\left[V_0 > k - 1\right]} \cdot \frac{\Pr_{\mathbf{Strgy}_j}\left[V_0 > k - 1\right]}{\Pr_{\mathbf{Strgy}_j}\left[V_0 > k - 2\right]} \cdots \frac{\Pr_{\mathbf{Strgy}_j}\left[V_0 > 1\right]}{\Pr_{\mathbf{Strgy}_j}\left[V_0 > 0\right]} \cdot \frac{\Pr_{\mathbf{Strgy}_j}\left[V_0 > 0\right]}{\Pr_{\mathbf{Strgy}_j}\left[E_{basis}\right]} \\
&= \Pr_{\mathbf{Strgy}_j}\left[V_0 = k \mid E_{basis}\right]
\end{aligned}$$

$product_2 = (\frac{2}{1+2nP^2})^{k+1}.$

Now from equation (20) and the above two equality formulas the proof follows. $\square$

Now the proof of Lemma 7 is complete. $\square$

Now Lemma 6 and Lemma 7 imply the result of Claim 1. $\square$

C.6.2 Proof of Claim 2 We first review some of the notations we fixed earlier.

$$A^0_{total} = \bigvee_{k=1}^{j-1} \bigvee_{\substack{d_{j-1}\in[P]\\ \vdots \\ d_{j-k}\in[P]}} \Psi{-}max^{(d_l,\ldots,d_{j+1},d_j,d_{j-1},\ldots,d_{j-k})}_{(w_l,\ldots,w_{j+1},w_j,1,\ldots,1)}[b'_l,\ldots,b'_{j+1},1,0,\ldots,b_{j-k}=0] \ \wedge\ b=0$$

$$\bigvee \Psi{-}max^{(d_l,\ldots,d_{j+1},d_j)}_{(w_l,\ldots,w_{j+1},w_j)}[b'_l,\ldots,b'_{j+1},1] \ \wedge\ b=0$$

$$A^1_{total} = \bigvee_{k=1}^{j-1} \bigvee_{\substack{d_{j-1}\in[P]\\ \vdots \\ d_{j-k}\in[P]}} \Psi{-}max^{(d_l,\ldots,d_{j+1},d_j,d_{j-1},\ldots,d_{j-k})}_{(w_l,\ldots,w_{j+1},w_j+1,1,\ldots,1)}[b'_l,\ldots,b'_{j+1},1,1,\ldots,b_{j-k}=1] \wedge b=1$$

$$\bigvee \Psi{-}max^{(d_l,\ldots,d_{j+1},d_j)}_{(w_l,\ldots,w_{j+1},w_j+1)}[b'_l,\ldots,b'_{j+1},1] \ \wedge\ b=1$$

$$E^0_{k,D} \ = \ \Psi{-}max^{(d_l,\ldots,d_{j+1},d_j,d_{j-1},\ldots,d_{j-k})}_{(w_l,\ldots,w_{j+1},w_j,1,\ldots,1)}, \text{ where } 1\le k\le j-1 \text{ and } D=(d_{j-1},\ldots,d_{j-k})$$

$$E^0_{0,\emptyset} \ = \ \Psi{-}max^{(d_l,\ldots,d_{j+1},d_j)}_{(w_l,\ldots,w_{j+1},w_j)}$$

$$E^1_{k,D} \ = \ \Psi{-}max^{(d_l,\ldots,d_{j+1},d_j,d_{j-1},\ldots,d_{j-k})}_{(w_l,\ldots,w_{j+1},w_j+1,1,\ldots,1)}, \text{ where } 1\le k\le j-1 \text{ and } D=(d_{j-1},\ldots,d_{j-k})$$

$$E^1_{0,D} \ = \ \Psi{-}max^{(d_l,\ldots,d_{j+1},d_j)}_{(w_l,\ldots,w_{j+1},w_j+1)}$$

$$E^0_{total} = \bigvee_{k=0}^{j-1} \bigvee_{D\in[P]^k} E^0_{k,D}$$

$$E^1_{total} = \bigvee_{k=0}^{j-1} \bigvee_{D\in[P]^k} E^1_{k,D}$$

Letting

$$D(\eta) = |\Pr_{\mathbf{Strgy_0}}[\mathcal{A}_{wstci}(\eta)=1 \mid A^0_{total}] - \Pr_{\mathbf{Strgy_0}}[\mathcal{A}_{wstci}(\eta)=1 \mid A^1_{total}]|$$

we need to prove that if $\Delta_0 = \Pr[A^0_{total}] > negl(\eta)$ and $\Delta_1 = \Pr[A^1_{total}] > negl(\eta)$, then $D(\cdot)$ is a negligible function. To this end, we make the following two claims:

Claim 5 *If* $\Pr[A^0_{total}] > negl(\eta)$ *and* $\Pr[A^1_{total}] > negl(\eta)$*, then*

$$(i) \qquad \left|\Pr_{\boldsymbol{Strgy_0}}\left[\mathcal{A}_{wstci}(\eta)=1 \mid A^0_{total}\right] - \Pr_{\boldsymbol{Strgy_j}}\left[\mathcal{A}_{wstci}(\eta)=1 \mid E^0_{total}\right]\right| = negl(\eta)$$

$$(ii) \qquad \left|\Pr_{\boldsymbol{Strgy_0}}\left[\mathcal{A}_{wstci}(\eta)=1 \mid A^1_{total}\right] - \Pr_{\boldsymbol{Strgy_j}}\left[\mathcal{A}_{wstci}(\eta)=1 \mid E^1_{total}\right]\right| = negl(\eta)$$

Claim 6

$$\Pr_{\boldsymbol{Strgy_j}}\left[\mathcal{A}_{wstci}(\eta)=1 \mid E^0_{total}\right] = \Pr_{\boldsymbol{Strgy_j}}\left[\mathcal{A}_{wstci}(\eta)=1 \mid E^1_{total}\right]$$

Note that from the above two claims, the proof of Claim 2 follows immediately. Thus we focus on proving the above claims.

C.6.2.1 ***Proof of Claim 5*** The proof of both (i) and (ii) are similar, so we prove part (i). We first show that the probability that $\mathcal{A}_{wstci}$ outputs 1 given E^0_{total} occurs in $\mathbf{Strgy}_j$ is equal to the same conditional probability under $\mathbf{Strgy}'_0$. Then we proceed to show that the difference of the values of the said conditional probability under $\mathbf{Strgy}_0$ and $\mathbf{Strgy}'_0$ is negligible, concluding the proof.

Lemma 10. *We have*

(a)

$$\Pr_{\boldsymbol{Strgy}_j}\left[\mathcal{A}_{wstci}(\eta) = 1 \mid E^0_{total}\right] = \Pr_{\boldsymbol{Strgy}'_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \mid E^0_{total}\right]$$

(b) If $\Pr[A^0_{total}] > negl(\eta)$*, then*

$$\left|\Pr_{\boldsymbol{Strgy}_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \mid A^0_{total}\right] - \Pr_{\boldsymbol{Strgy}'_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \mid E^0_{total}\right]\right| = negl(\eta)$$

Now we give the proof of each part of the lemma:

Proof of Part (a) of Lemma 10. The proof of this part almost immediately follows from the very descriptions of $\mathbf{Strgy}_j$ and $\mathbf{Strgy}'_0$, considering that

1. E^0_{total} occurs with the same probability under $\mathbf{Strgy}_j$ and $\mathbf{Strgy}'_0$, and
2. Given E^0_{total} occurs, $\mathcal{A}_{wstci}$ has identical views when run under $\mathbf{Strgy}_j$ and $\mathbf{Strgy}'_0$.

We omit the details.

□

Proof of Part (b) of Lemma 10. We have to show that if $\Pr_{\mathbf{Strgy}_0}[A^0_{total}] > negl(\eta)$, then

$$\left|\Pr_{\mathbf{Strgy}_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \mid A^0_{total}\right] - \Pr_{\mathbf{Strgy}'_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \mid E^0_{total}\right]\right| = negl(\eta)$$

We expand each conditional probability as follows:

$$\begin{aligned}
p_1 &\triangleq \Pr_{\mathbf{Strgy}_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \mid A^0_{total}\right] \\
&= \Pr_{\mathbf{Strgy}_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \wedge \overline{\textit{Failed-Dec}} \mid A^0_{total}\right] + \Pr_{\mathbf{Strgy}_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \wedge \textit{Failed-Dec} \mid A^0_{total}\right] \\
&= \Pr_{\mathbf{Strgy}_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \mid \overline{\textit{Failed-Dec}} \wedge A^0_{total}\right] \cdot \Pr_{\mathbf{Strgy}_0}\left[\overline{\textit{Failed-Dec}} \mid A^0_{total}\right] \\
&\quad + \Pr_{\mathbf{Strgy}_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \wedge \textit{Failed-Dec} \mid A^0_{total}\right].
\end{aligned}$$

Similarly we have

$$\begin{aligned}
p_2 &\triangleq \Pr_{\mathbf{Strgy}'_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \mid E^0_{total}\right] \\
&= \Pr_{\mathbf{Strgy}'_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \mid \overline{\textit{Failed-Dec}} \wedge E^0_{total}\right] \cdot \Pr_{\mathbf{Strgy}'_0}\left[\overline{\textit{Failed-Dec}} \mid E^0_{total}\right] \\
&\quad + \Pr_{\mathbf{Strgy}'_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \wedge \textit{Failed-Dec} \mid E^0_{total}\right].
\end{aligned}$$

We need to show that if $\Pr_{\mathbf{Strgy}_0}[A^0_{total}] > negl(\eta)$, then $|p_1 - p_2|$ is negligible. To do so, we first show that if $\Pr_{\mathbf{Strgy}_0}[A^0_{total}] > negl(\eta)$, then $\Pr_{\mathbf{Strgy}'_0}[E^0_{total}] > negl(\eta)$. Then the proof of $|p_1 - p_2| = negl(\eta)$ is derived from the following proposition. In the following proposition, *Failed-Dec* refers to the event defined in the proof of Claim (4). Although, it was defined for $\mathbf{Strgy}'_0$, it can similarly be defined for $\mathbf{Strgy}_0$ as well.

Proposition 2. *We have*

(i) If $\Pr_{\textit{Strgy}_0}[A^0_{total}] > negl(\eta)$*, then*

$$\Pr_{\textit{Strgy}_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \wedge \textit{Failed-Dec} \mid A^0_{total}\right] = negl(\eta)$$

(ii) If $\Pr_{\textit{Strgy}'_0}[E^0_{total}] > negl(\eta)$*, then*

$$\Pr_{\textit{Strgy}'_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \wedge \textit{Failed-Dec} \mid E^0_{total}\right] = negl(\eta)$$

(iii)

$$\Pr_{\textit{Strgy}_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \mid \overline{\textit{Failed-Dec}} \wedge A^0_{total}\right] = \Pr_{\textit{Strgy}'_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \mid \overline{\textit{Failed-Dec}} \wedge E^0_{total}\right]$$

(iv) If $\Pr_{\textit{Strgy}_0}[A^0_{total}] > negl(\eta)$ *and* $\Pr_{\textit{Strgy}'_0}[E^0_{total}] > negl(\eta)$

$$\left|\Pr_{\textit{Strgy}_0}\left[\overline{\textit{Failed-Dec}} \mid A^0_{total}\right] - \Pr_{\textit{Strgy}'_0}\left[\overline{\textit{Failed-Dec}} \mid E^0_{total}\right]\right| = negl(\eta)$$

As discussed earlier, we first show that if $\Pr_{\mathbf{Strgy}_0}[A^0_{total}] > negl(\eta)$, then $\Pr_{\mathbf{Strgy}'_0}[E^0_{total}] > negl(\eta)$, and then we proceed to prove the above proposition.

Suppose $\Pr_{\mathbf{Strgy}_0}[A^0_{total}] > negl(\eta)$, and, to the contrary, $\Pr_{\mathbf{Strgy}'_0}[E^0_{total}] = negl(\eta)$. From Claim (3), we obtain that $\Pr_{\mathbf{Strgy}_j}[E^0_{total}] = negl(\eta)$, and consequently, the following,

$$\sum_{k=0}^{j-1} \sum_{D \in [P]^k} (\frac{1}{2})^{l-j+k+2} \cdot \Pr_{\mathbf{Strgy}_j}[E^0_{k,D}]$$

is also negligible. (Note that we have $j \leq l$.) Now we are reaching a contradiction; since, by Lemma (6), Equation (10) and the fact that l is constant, we have

$$\left|\Pr_{\mathbf{Strgy}_0}[A^0_{total}] - \sum_{k=0}^{j-1} \sum_{D \in [P]^k} (\frac{1}{2})^{l-j+k+2} \cdot \Pr_{\mathbf{Strgy}_j}[E^0_{k,D}]\right| = negl(\eta),$$

which implies that $\Pr_{\mathbf{Strgy}_0}[A^0_{total}]$ should also be negligible.

Now we proceed to prove Proposition (2).

Proofs of Parts (i) and (ii) of Proposition (2). The proofs of both (i) and (ii) are entirely similar, so we prove (i). Suppose, to the contrary,

$$\Pr_{\mathbf{Strgy}_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \wedge \textit{Failed-Dec} \mid A^0_{total}\right] > negl(\eta).$$

Since we already have $\Pr_{\mathbf{Strgy}_0}[A^0_{total}] > negl(\eta)$, we obtain that

$$\Pr_{\mathbf{Strgy}_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \wedge \textit{Failed-Dec} \wedge A^0_{total}\right] > negl(\eta),$$

and as a result,

$$\Pr_{\mathbf{Strgy}_0}[\textit{Failed-Dec}] > negl(\eta).$$

Now exactly like the proof of Lemma (8), from non-negligibility of the above probability, one can efficiently construct a successful IND-CCA2-attack against the encryption scheme, thereby contradicting the IND-CCA2 security assumption of the scheme. This concludes the proof. □

Proof of Part (iii). The idea of the proof is analogous to that of part (a) of Lemma 10; namely, intuitively, the view of $\mathcal{A}_{wstci}$ under $\mathbf{Strgy}_0$ and provided $\overline{\textit{Failed-Dec}} \wedge A^0_{total}$ occurs is "identical" to his view under $\mathbf{Strgy}'_0$ and provided $\overline{\textit{Failed-Dec}} \wedge E^0_{total}$ occurs. We now formalize this idea below.

First note that all the random coins needed to be tossed by $\mathcal{A}_{cca}$ under $\mathbf{Strgy}_0$ are also tossed under $\mathbf{Strgy}'_0$, while, at most, an additional l number of coin-tossings are needed under $\mathbf{Strgy}_0$ to sample b_i's. (Note that the values of b_i's are fixed under $\mathbf{Strgy}'_0$.) Therefore, without loss of generality, we assume that $\mathcal{A}_{cca}$ when run under $\mathbf{Strgy}_0$ uses l more random bits than when run under $\mathbf{Strgy}'_0$, and that the first l bits of $r_{\mathcal{A}_{cca}}$ under $\mathbf{Strgy}_0$ are used to sample b_i's. Under this assumption, if $(r_{\mathcal{A}_{wstci}}, r_{\mathcal{A}_{cca}})$ is a valid (i.e. of required length) pair of random coins under $\mathbf{Strgy}'_0$, so is $(r_{\mathcal{A}_{wstci}}, r_l || r_{\mathcal{A}_{cca}})$ under $\mathbf{Strgy}_0$, where $r_l \in \{0,1\}^l$. Now we can easily verify that if, for $r_l \in \{0,1\}^l$, the pair $(r_{\mathcal{A}_{wstci}}, r_l || r_{\mathcal{A}_{cca}})$ enforces $(\overline{\textit{Failed-Dec}} \wedge A^0_{total})$ to occur under $\mathbf{Strgy}_0$, then $(r_{\mathcal{A}_{wstci}}, r_{\mathcal{A}_{cca}})$ enforces $(\overline{\textit{Failed-Dec}} \wedge E^0_{total})$ to occur under $\mathbf{Strgy}'_0$; and that if $(r_{\mathcal{A}_{wstci}}, r_l || r_{\mathcal{A}_{cca}})$, under $\mathbf{Strgy}_0$, makes $\mathcal{A}_{wstci}$ output 1 provided that it enforces $(\overline{\textit{Failed-Dec}} \wedge A^0_{total})$ to occur, then $(r_{\mathcal{A}_{wstci}}, r_{\mathcal{A}_{cca}})$ makes $\mathcal{A}_{wstci}$ output 1 under $\mathbf{Strgy}'_0$. The converse of this is also true; namely, if $(r_{\mathcal{A}_{wstci}}, r_{\mathcal{A}_{cca}})$ under $\mathbf{Strgy}'_0$ enforces $(\overline{\textit{Failed-Dec}} \wedge E^0_{total})$ to occur, and in particular $(\overline{\textit{Failed-Dec}} \wedge E^0_{k,D})$ to occur for some $k \in \{0, \ldots, j-1\}$, then $(r_{\mathcal{A}_{wstci}}, r_l || r_{\mathcal{A}_{cca}})$ does also enforce $(\overline{\textit{Failed-Dec}} \wedge A^0_{k,D})$ (and as a result $(\overline{\textit{Failed-Dec}} \wedge A^0_{total})$) to occur under $\mathbf{Strgy}_0$, where r_l is any bitstring of length l which induces the same sampling of b_i's as that enforced under $\mathbf{Strgy}'_0$ by using $(r_{\mathcal{A}_{wstci}}, r_{\mathcal{A}_{cca}})$. (i.e. $b_r = b'_r$ for $j+1 \leq r \leq l$, $b_j = 1$, and $b_i = 0$ for $s \leq i \leq j-1$, where s is the number induced by $(r_{\mathcal{A}_{wstci}}, r_{\mathcal{A}_{cca}})$ under $\mathbf{Strgy}'_0$.) Moreover, if if $(r_{\mathcal{A}_{wstci}}, r_{\mathcal{A}_{cca}})$ under $\mathbf{Strgy}'_0$ makes $\mathcal{A}_{wstci}$ output 1 provided that it enforces $(\overline{\textit{Failed-Dec}} \wedge E^0_{total})$ to occur, then $(r_{\mathcal{A}_{wstci}}, r_l || r_{\mathcal{A}_{cca}})$ makes $\mathcal{A}_{wstci}$ output 1 under $\mathbf{Strgy}_0$, for any $r_l \in \{0,1\}^l$ which enforces the same sampling on b_i's as that enforced under $\mathbf{Strgy}'_0$ by using $(r_{\mathcal{A}_{wstci}}, r_{\mathcal{A}_{cca}})$. The proof immediately follows from the above observations. □

Proof of Part (iv). To prove this part, we show that $\Pr_{\mathbf{Strgy}_0}\left[\textit{Failed-Dec} \mid A^0_{total}\right] = negl(\eta)$ and $\Pr_{\mathbf{Strgy}'_0}\left[\textit{Failed-Dec} \mid E^0_{total}\right] = negl(\eta)$, and this completes the proof. This is because we have:

$$\Pr_{\mathbf{Strgy}_0}\left[\overline{\textit{Failed-Dec}} \mid A^0_{total}\right] = 1 - \Pr_{\mathbf{Strgy}_0}\left[\textit{Failed-Dec} \mid A^0_{total}\right]$$
$$\Pr_{\mathbf{Strgy}'_0}\left[\overline{\textit{Failed-Dec}} \mid E^0_{total}\right] = 1 - \Pr_{\mathbf{Strgy}'_0}\left[\textit{Failed-Dec} \mid E^0_{total}\right].$$

Now we show that $\Pr_{\mathbf{Strgy}_0}\left[\textit{Failed-Dec} \mid A^0_{total}\right] = negl(\eta)$. (The proof for $\Pr_{\mathbf{Strgy}'_0}\left[\textit{Failed-Dec} \mid E^0_{total}\right] = negl(\eta)$ is exactly similar.) Suppose $\Pr_{\mathbf{Strgy}_0}\left[\textit{Failed-Dec} \mid A^0_{total}\right] > negl(\eta)$; since by assumption we have $\Pr_{\mathbf{Strgy}_0}[A^0_{total}] > negl(\eta)$, we obtain $\Pr_{\mathbf{Strgy}_0}\left[\textit{Failed-Dec} \wedge A^0_{total}\right] > negl(\eta)$, and as a result, $\Pr_{\mathbf{Strgy}_0}[\textit{Failed-Dec}] > negl(\eta)$, which is a contradiction, as illustrated in the proof of Part (i). □

Now the proof of Proposition (2), Lemma (10) , and Claim (5) are all complete. □

□

□

C.6.2.2 ***Proof of Claim 6*** We need to prove that

$$\Pr_{\mathbf{Strgy}_j}\left[\mathcal{A}_{wstci}(\eta)=1 \mid E^0_{total}\right]=\Pr_{\mathbf{Strgy}_j}\left[\mathcal{A}_{wstci}(\eta)=1 \mid E^1_{total}\right].$$

First we discuss why intuitively the above equality should be true, and then we rigorously demonstrate its correctness using a probabilistic argument. The intuition behind the above equality is that the view of $\mathcal{A}_{wstci}$ run under $\mathbf{Strgy}_j$ is identical under the occurrence of E^0_{total} or E^1_{total}. To see this, note that provided that E^0_{total} occurs, the distribution of replies given to $\mathcal{A}_{wstci}$'s queries is formed as follows:

1. The first w_j occurrences of $k^{-1}_{u_j}$ are replaced with a fake value (i.e. fsk_{u_j}), and the rest with the real value (i.e. sk_{u_j}). Also for $j+1 \le t \le l$, the first $w_t - b''_t$ occurrences of $k^{-1}_{u_t}$ are replaced with a fake value (i.e. fsk_{u_t}), and the rest with the real value (i.e. sk_{u_t}), where $b''_t = 0$ if $b'_t \neq b'_{t-1}$, and $b''_t = 1$ otherwise.
2. All other symbols (i.e. nonces, public keys, and other private keys) are replaced with their real values.

Note that the above distribution is also what formed as a result of the occurrence of E^1_{total}. In particular, the values of $u_{j-1},\dots,u_s$ have no effect on the distribution of replies given to the adversary, and the values of $u_j,\dots,u_l$ (which do participate in the randomness of replies) are distributed "identically" under both E^0_{total} and E^1_{total}. Now we turn the above reasoning into a rigorous argument. First let's introduce some notation.

$E^0_{basis} = \forall i \in \{j,\dots,l\}, indeg(v_{u_i}) = d_i$

$E^1_{basis} = \forall i \in \{j+1,\dots,l\}, on_i = w_i$

$E^2_{basis} = \forall i \in \{j+1,\dots,l\}, ed_i = \Re_{\text{cont.}}(\xrightarrow{w_i} v_{u_i} \xrightarrow{w_{i+1}} v_{u_{i+1}} \xrightarrow{w_{i+2}} \dots \xrightarrow{w_l} v_{u_l})$

$E^3_{basis} = \forall i \in \{j,\dots,l-1\}, v_{u_i} = prev\left(\xrightarrow{w_{i+1}} v_{u_{i+1}} \xrightarrow{w_{i+2}} v_{u_{i+2}} \xrightarrow{w_{i+3}} \dots \xrightarrow{w_l} v_{u_l}, ed'_{i+1}\right)$

$E^4_{basis} =$ If $\mathcal{A}_{wstci}$ makes a *challenge* query $challenge(k^{-1}_j, bs)$ (i.e. private-key challenge), then $u_l = j$.

We prove that:

$$\Pr_{\mathbf{Strgy}_j}\left[\mathcal{A}_{wstci}(\eta)=1 \mid E^0_{total}\right]=\Pr_{\mathbf{Strgy}_j}\left[\mathcal{A}_{wstci}(\eta)=1 \mid E^1_{total}\right]=\Pr_{\mathbf{Strgy}_j}\left[\mathcal{A}_{wstci}(\eta)=1 \mid E_{basis}\right], \tag{23}$$

where $E_{basis} = E^0_{basis} \wedge \dots \wedge E^4_{basis}$. We show the above equality for E^0_{total} and the proof for E^1_{total} is obtained similarly. We give the following last set of notation:

$$A^0_i = (u_0 = u_1 = \dots = u_{i-1} = 0) \wedge (u_i \neq 0) \wedge (\forall i_1 : i+1 \le i_1 \le j-1, on_{i_1} = 1) \wedge (on_j = w_j)$$

$$\begin{aligned} A^1_i = {} & \left(\forall i_1 : i+1 \le i_1 \le j-1, ed_{i_1} = \Re_{\text{cont.}}(\xrightarrow{1} v_{u_{i_1}} \xrightarrow{1} \dots \xrightarrow{1} v_{u_{j-1}} \xrightarrow{w_j} v_{u_j} \xrightarrow{w_{j+1}} \dots \xrightarrow{w_l} v_{u_l})\right) \\ & \wedge (ed_j = \Re_{\text{cont.}}(\xrightarrow{w_j} v_{u_j} \xrightarrow{w_{j+1}} \dots \xrightarrow{w_l} v_{u_l})) \end{aligned}$$

$$\begin{aligned} A^2_i = {} & \left(\forall i_1 : i+1 \le i_1 \le j-1, v_{u_{(i_1-1)}} = prev\left(\xrightarrow{1} v_{u_{i_1}} \xrightarrow{1} \dots \xrightarrow{1} v_{u_{j-1}} \xrightarrow{w_j} v_{u_j} \xrightarrow{w_{j+1}} \dots \xrightarrow{w_l} v_{u_l}, ed'_{i_1}\right)\right) \\ & \wedge \left(v_{u_{(j-1)}} = prev\left(\xrightarrow{w_j} v_{u_j} \xrightarrow{w_{j+1}} v_{u_{(j+1)}} \xrightarrow{w_{j+2}} \dots \xrightarrow{w_l} v_{u_l}, ed'_j\right)\right). \end{aligned}$$

Letting $A_i = A^0_i \wedge A^1_i \wedge A^2_i$, and defining $E_c = A_0 \vee A_1 \vee \dots \vee A_{j-1}$, we have $E^0_{total} = E_{basis} \wedge E_c$. Now we can write

$$\Pr_{\mathbf{Strgy}_j}\left[\mathcal{A}_{wstci}(\eta)=1 \mid E^0_{total}\right]$$

$$
\begin{aligned}
&= \Pr_{\mathbf{Strgy}_j}[\mathcal{A}_{wstci}(\eta) = 1 \mid E_{basis} \wedge E_c] \\
&= \frac{\Pr_{\mathbf{Strgy}_j}[\mathcal{A}_{wstci}(\eta) = 1 \wedge E_c \mid E_{basis}]}{\Pr_{\mathbf{Strgy}_j}[E_c \mid E_{basis}]} \\
&= \frac{\Pr_{\mathbf{Strgy}_j}[\mathcal{A}_{wstci}(\eta) = 1 \mid E_{basis}] \cdot \Pr_{\mathbf{Strgy}_j}[E_c \mid E_{basis}]}{\Pr_{\mathbf{Strgy}_j}[E_c \mid E_{basis}]} \\
&= \Pr_{\mathbf{Strgy}_j}[\mathcal{A}_{wstci}(\eta) = 1 \mid E_{basis}]
\end{aligned}
\tag{24}
$$

The reason why the equality line (15) holds is that, given that E_{basis} occurs, the two events $\mathcal{A}_{wstci}(\eta) = 1$ and E_c are independent. This is because (as also explained above) variables $\{u_i\}_{s \le i \le j-1}, \{(on_i, ed_i, ed'_i)\}_{s \le i \le j-1}$, ed_j do not affect the randomness of replies given to adversary's queries, which make the two events $\mathcal{A}_{wstci}(\eta) = 1$ and E_c independent. Now the proof of Claim 6 is complete. □

Now Claim 5 and Claim 6 imply Claim 2. □

Claim 1 and Claim 2 completes the proof of Lemma 4. □

C.7 Proof of Lemma 5

The proof of this lemma fairly follows a similar path taken for the proof of Lemma (4). Thus we present a less detailed proof than that of Lemma (4). The proof of all parts of the lemma are entirely similar, so we give the proof for part (1). Let $j, b'_l, \ldots, b'_j, w_l, \ldots, w_j, d_l, \ldots, d_j$ be defined as specified in the statement of the lemma, and also define $b'_{j-1} = 0$. We fix all these variables throughout the proof, and for ease of notation, we write B^h_{total} instead of $B^h_{total,j,B_j,D_j,W_j}$ (for $h \in \{0,1\}$). Now defining $S_0 = \Pr_{\mathbf{Strgy}_0}[B^0_{total}]$ and $S_1 = \Pr_{\mathbf{Strgy}_0}[B^1_{total}]$, we have to show that,

$$
\left| \Pr_{\mathbf{Strgy}_0}\left[B^0_{total}\right] \cdot \Pr_{\mathbf{Strgy}_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \mid B^0_{total}\right] - \Pr_{\mathbf{Strgy}_0}\left[B^1_{total}\right] \cdot \Pr_{\mathbf{Strgy}_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \mid B^1_{total}\right] \right| = negl(\eta).
\tag{25}
$$

To this end, we, similarly as before, prove the following two claims.

Claim 7 $|S_0 - S_1| = negl(\eta)$.

Claim 8 *If* $S_0 > negl(\eta)$ *and* $S_1 > negl(\eta)$, *then*

$$
\Pr_{\mathit{Strgy}_0}[\mathcal{A}_{wstci}(\eta) = 1 \mid B^0_{total}] - \Pr_{\mathit{Strgy}_0}[\mathcal{A}_{wstci}(\eta) = 1 \mid B^1_{total}] = negl(\eta)
$$

.

Note that, again, the above two claims yield (25), because if either S_0 or S_1 is negligible, then according to Claim (7), the other one will also be negligible, and consequently (25) follows.

Thus we focus on proving the above two claims. We give the following notations. For $2 \le k \le j-1$ and $D = (d_{j-1}, \ldots, d_{j-k}) \in [P]^k$, we define the following events:

$$
\begin{aligned}
B^1_{k,D} &= \Psi{-}max^{(d_l,\ldots,d_j,d_{j-1},d_{j-2},\ldots,d_{j-k})}_{(w_l,\ldots,w_j,d_{j-1},1\ldots,1)}[b_l = b'_l, \ldots, b_j = b'_j, b_{j-1} = 0, b_{j-2} = 1, \ldots, b_{j-k} = 1] \wedge b = 1 \\
B^1_{1,\{d_{j-1}\}} &= \Psi{-}max^{(d_l,\ldots,d_j,d_{j-1})}_{(w_l,\ldots,w_j,d_{j-1})}[b_l = b'_l, \ldots, b_j = b'_j, b_{j-1} = 0] \wedge b = 1 \\
F^1_{k,D} &= \Psi{-}max^{(d_l,\ldots,d_j,d_{j-1},d_{j-2},\ldots,d_{j-k})}_{(w_l,\ldots,w_j,d_{j-1},1\ldots,1)} \\
F^1_{1,\{d_{j-1}\}} &= \Psi{-}max^{(d_l,\ldots,d_j,d_{j-1})}_{(w_l,\ldots,w_j,d_{j-1})}
\end{aligned}
$$

$$F^0_{total} \;=\; \Psi\text{--}\overline{max}^{(d_l,\ldots,d_j)}_{(w_l,\ldots,w_j)}$$
$$F^1_{total} = \bigvee_{k=1}^{j-1} \bigvee_{D\in[P]^k} F^1_{k,D}$$

Now can expand S_0 and S_1 as follows:

$$S_0 = (\frac{1}{2})^{l-j+2} \cdot \Pr_{\mathbf{Strgy}_0}\left[F^0_{total} \;\mid\; b_l = b'_l;\; \ldots;\; b_j = b'_j;\; b = 0\right]$$
$$S_1 = \sum_{k=1}^{j-1} \sum_{D\in[P]^k} (\frac{1}{2})^{l-j+k+2} \cdot \Pr_{\mathbf{Strgy}_0}\left[F^1_{k,D} \mid b_l = b'_l;\; \ldots;\; b_j = b'_j;\; b_{j-1} = 0;\; b_{j-2} = 1;\; \ldots;\; b_{j-k} = 1;\; b = 1\right]$$

C.7.1 Proof of Claim 7 We introduce a new security game, $\mathbf{Strgy}'_{j-1}$, as follows: in the SETUP phase:

- Variables $u_0, \ldots, u_l$ are sampled in the same manner as in $\mathbf{Strgy}_0$, and denoting by s the smallest index where $u_s \neq 0$, if $s \geq j$, then the execution is terminated (a failed situation), otherwise, for each $s < t \leq l$ we sample three parameters (on_t, ed_t, ed'_t), again, in exactly the same manner as in $\mathbf{Strgy}_0$. (Again, under $\mathbf{Strgy}'_{j-1}$ this sequence of triples of random values has no effect on the randomness of replies given to adversary's queries; they are included in the game as we are to formulate some events depending on them.)
- We generate n pairs of public/private key pairs $(pk_1, sk_1), \ldots, (pk_n, sk_n)$, fake private keys $fsk_{u_{j-1}}, fsk_{u_j} \ldots, fsk_{u_l}$, and n nonces $nc_1, \ldots, nc_n$, independently at random from the same distributions used in $\mathbf{Strgy}_0$.

  In the Simulation phase:

- An *encryption* query $encrypt(e, i)$ is answered by $E \leftarrow [\![e]\!]_{\tau_0}$, where τ_0 is defined as follows: $\tau_o(x_i) = nc_i$, $\tau_0(k_i) = pk_i$ for $1 \leq i \leq n$. Moreover, if $k_i^{-1} \notin \{k^{-1}_{u_{j-1}}, k^{-1}_{u_j}, \ldots, k^{-1}_{u_l}\}$, we assign $\tau_0(k^{-1}_{i,o}) = sk_i$, for all o's, and for $k^{-1}_{u_{(j-1)}}$, we assign $\tau_0(k^{-1}_{u_{(j-1)},o_1}) = fsk_{u_j}$, for all o_1's. Finally, for all $t \in \{j, \ldots, l\}$, we define $\tau_0(k^{-1}_{u_t,o_3}) = fsk_{u_t}$ and $\tau_0(k^{-1}_{u_t,o_4}) = sk_{u_t}$, for all $o_3 < w_t$ and $o_4 > w_t$, and if $b'_t = b'_{t-1}$, then $\tau_0(k^{-1}_{u_t,w_t}) = sk_{u_t}$, and, $\tau_0(k^{-1}_{u_t,w_t}) = fsk_{u_t}$, otherwise. (Recall that we set $b'_{j-1} = 0$.)
- A *corruption* query $corrupt(i)$ is handled by returning sk_i.
- For *decryption* queries, we first define a key-renaming function, R, as follows: for all $k^{-1}_{i,o}$ we have $R[k^{-1}_{i,o}] = k^{-1}_i$ if $\tau_0(k^{-1}_{i,o}) = sk_i$, and $R[k^{-1}_{i,o}] = {k'}^{-1}_i$ if $\tau_0(k^{-1}_{i,o}) = fsk_i$. Now a query $decrypt(c, i)$ is answered by $Dec(c, sk_i)$, if none of the following events occur, and it is answered by $\perp$, otherwise: (a) (c, i) is coinductively visible in *eval-exp*, and (b) for some $(\{e\}_{k_h}, bs) \in$ *eval-exp* it holds that there exists a subexp $\{e'\}_{k_p}$ of $\{e\}_{k_h}$ such that $R[\{e'\}_{k_p}]$ is type-1 and that c is the computational image of $\{e'\}_{k_p}$ in bs.
- A *subexp-testing* query $subexp(\{e\}_{k_i}, bs_i, \{e'\}_{k_j}, bs_j)$ is answered by 1 if bs_j is the computational image of $\{e'\}_{k_j}$ in bs_i, and answered by 0, otherwise.

Similarly to Lemmas 6 and 7, we can give the following two Lemmas.

Lemma 11. *For every $D = [P]^k$, it holds that:*

$$\left|\Pr_{\mathit{Strgy}_0}\left[F^0_{total} \;\mid\; b_l = b'_l;\; \ldots;\; b_j = b'_j;\; b = 0\right] - \Pr_{\mathit{Strgy}'_{j-1}}\left[F^0_{total}\right]\right| = negl(\eta)$$

$$\left| \Pr_{\textbf{\textit{Strgy}}_0} \left[F^1_{k,D} \mid b_l = b'_l;\ \ldots;\ b_j = b'_j;\ b_{j-1} = 0;\ b_{j-2} = 1;\ \ldots;\ b_{j-k} = 1;\ b = 1\right] - \Pr_{\textbf{\textit{Strgy}}'_{j-1}} [F^1_{k,D}] \right| = negl(\eta)$$

Lemma 12.

$$(\frac{1}{2})^{l-j+2} \cdot \Pr_{\textbf{\textit{Strgy}}'_{j-1}} \left[F^0_{total}\right] = \sum_{k=1}^{j-1} \sum_{D \in \{[P]\}^k} (\frac{1}{2})^{l-j+k+2} \cdot \Pr_{\textbf{\textit{Strgy}}'_{j-1}} \left[F^1_{k,D}\right]$$

The above two Lemmas yield Claim (7). (Again here we are using the assumption that l is constant.) The proofs of the above lemmas also follow an essentially similar line of reasoning as those of Lemmas (6) and (7); therefore, we will omit most of the details.

C.7.1.1 ***Proof Sketch for Lemma (11).*** We give the outline of the proof for the second part; the proof for the first part is obtained similarly. We proceed similarly to the proof of Lemma 6. We first define an intermediate game, $\mathbf{Strgy}''_0$, which similarly to $\mathbf{Strgy}'_0$, is a special case of $\mathbf{Strgy}_0$ by just (deterministically) setting the values for b_i's to b'_i's. Namely, after sampling the sequence $\{u_i\}_{0 \leq i \leq l}$, and letting s denote the smallest index for which it holds $u_s \neq 0$, if $s \geq j$, then the execution is terminated; otherwise we assign $b_i = b'_i$ for $j \leq i \leq l$, and $b_{j-1} = 0$, and finally $b_r = 1$ for $s \leq r \leq j-2$. The rest of the game is exactly similar to that of $\mathbf{Strgy}'_0$. Now analogously to Claims 3 and 4, we can prove the following two results, completing the proof of Lemma 11.

$$\Pr_{\mathbf{Strgy}'_{j-1}} \left[F^1_{k,D}\right] = \Pr_{\mathbf{Strgy}''_0} \left[F^1_{k,D}\right]$$

$$\left| \Pr_{\mathbf{Strgy}''_0} \left[F^1_{k,D}\right] - \Pr_{\mathbf{Strgy}_0} \left[F^1_{k,D} \mid b_l = b'_l;\ \ldots;\ b_j = b'_j;\ b_{j-1} = 0;\ b_{j-2} = 1;\ \ldots;\ b_{j-k} = 1;\ b = 1\right] \right| = negl(\eta).$$

The proofs of both equality formulas above are obtained by the same reasoning used to prove Claims 3 and 4; therefore, we omit the details. □

C.7.1.2 ***Proof Sketch for Lemma 12.*** Defining

$$tot_0 = (\frac{1}{2})^{l-j+2} \cdot \Pr_{\mathbf{Strgy}'_{j-1}} \left[F^0_{total}\right]$$
$$tot_1 = \sum_{k=1}^{j-1} \sum_{D \in [P]^k} (\frac{1}{2})^{l-j+k+2} \cdot \Pr_{\mathbf{Strgy}'_{j-1}} \left[F^1_{k,D}\right],$$

we prove that

$$tot_0 = tot_1 = (\frac{1}{2})^{l-j+2} \cdot (\frac{1}{1+2nP^2})^{j-1} \cdot \Pr_{\mathbf{Strgy}'_{j-1}} [F_{basis}], \tag{26}$$

$$\tag{27}$$

where F_{basis} is the intersection of the following four events (i.e. $F_{basis} = F^0_{basis} \wedge \cdots \wedge F^4_{basis}$):

1. $F^0_{basis} ::= (\forall i \in \{j, \ldots, l\}, indeg(v_{u_i}) = d_i) \wedge \Big(indeg(v_{u_{(j-1)}}) > 0\Big)$,
2. $F^1_{basis} ::= \forall i \in \{j, \ldots, l\}, on_i = w_i$,
3. $F^2_{basis} ::= \forall i \in \{j, \ldots, l\}, ed_i = \Re_{\text{cont.}}(\xrightarrow{w_i} v_{u_i} \xrightarrow{w_{i+1}} v_{u_{i+1}} \xrightarrow{w_{i+3}} \ldots \xrightarrow{w_l} v_{u_l})$,
4. $F^3_{basis} ::= \forall i \in \{j-1, j, \ldots, l\}, v_{u_i} = prev\left(\xrightarrow{w_{i+1}} v_{u_{i+1}} \xrightarrow{w_{i+2}} v_{u_{i+2}} \xrightarrow{w_{i+3}} \ldots \xrightarrow{w_l} v_{u_l}, ed'_{i+1}\right)$.

5. F^4_{basis} ::= If $\mathcal{A}_{wstci}$ makes a *challenge* query $challenge(k_j^{-1}, bs)$ (i.e. private-key challenge), then $u_l = j$.

Proving (26) for tot_0 is rather straightforward. For this, note that we have:

$$\begin{aligned} tot_0 &= (\frac{1}{2})^{l-j+2} \cdot \Pr_{\mathbf{Strgy}'_{j-1}}\left[F^0_{total}\right] \\ &= (\frac{1}{2})^{l-j+2} \cdot \Pr_{\mathbf{Strgy}'_{j-1}}\left[F_{basis} \wedge (u_{j-2} = u_{j-3} = \cdots = u_0 = 0)\right] \\ &= (\frac{1}{2})^{l-j+2} \cdot \Pr_{\mathbf{Strgy}'_{j-1}}\left[F_{basis}\right] \cdot \Pr_{\mathbf{Strgy}'_{j-1}}\left[u_{j-2} = u_{j-3} = \cdots = u_0 = 0\right] \\ &= (\frac{1}{2})^{l-j+2} \cdot (\frac{1}{1+2nP^2})^{j-1} \cdot \Pr_{\mathbf{Strgy}'_{j-1}}\left[F_{basis}\right]. \end{aligned}$$

Showing (26) for tot_1 is relatively more involved, but essentially follows a similar line of probabilistic reasoning to that of the proof of Lemma 7 for $total_0$. Now we briefly explain how the proof works.

First, we let $\mathcal{P}_c = v_{u_{j-1}} \xrightarrow{w_j} v_{u_j} \xrightarrow{w_{j+1}} \dots \xrightarrow{w_l} v_{u_l}$, which is a non-maximal, coinductively-continuable path if F_{basis} occurs. Now, analogously to V_0 defined in the proof of Lemma 7, we define the random variable V_1 which takes values in $\{-1, 1, 2, \dots, j-1\}$ with the following probability distribution: after the execution of $\mathbf{Strgy}'_{j-1}$, if F_{basis} does not occur, then $V_1 = -1$; otherwise, the probability that $V_1 = 1$ is the probability that $indeg(v_{i_1}) = 0$, where v_{i_1} is picked uniformly at random from $\Re_{\text{cont.}}(\xrightarrow{indeg(v_{u_{j-1}})} \mathcal{P}_c)$. For $r > 1$, the probability that $V_1 = r$ is defined to be the probability that $q = r$ after the execution of the following:

```
1: q ← 1
2: while indeg(v_{i_q} > 0) do
3:    Pick v_{i_{q+1}} uniformly at random from ℜ_cont.(→^1 v_{i_q} →^1 … →^1 v_{i_1} →^{indeg(v_{u_{j-1}})} P_c)
4:    q ← q + 1
5: end while
```

Recall that

$$F^1_{k,D} \;\; = \;\; \Psi{-}max^{(d_l,\dots,d_j,d_{j-1},d_{j-2},\dots,d_{j-k})}_{(w_l,\dots,w_j,d_{j-1},1\dots,1)}.$$

Now, defining

$$sum_k = \sum_{D \in [P]^k} \Pr_{\mathbf{Strgy}'_{j-1}}\left[F^1_{k,D}\right],$$

to prove (26) we show that

$$sum_k = 2^k \cdot (\frac{1}{1+2nP^2})^{j-1} \cdot \Pr_{\mathbf{Strgy}'_{j-1}}\left[F_{basis}\right] \cdot \Pr_{\mathbf{Strgy}'_{j-1}}\left[V_1 = k \mid F_{basis}\right], \tag{28}$$

and this completes the proof. (This is because, the assumption that the diameter of the hidden subgraph is at most l implies that $V_1 = -1$ or $1 \leq V_1 \leq j-1$.) To this end, for $i \in \{1, \dots, j-2\}$, we introduce the following random variables:

$$A^0_i = (s \leq i) \wedge (\forall i_1 : i+1 \leq i_1 \leq j-2, on_{i_1} = 1) \wedge \big(on_{j-1} = indeg(v_{u_{j-1}})\big) \wedge (\forall i_2 \in \{j, \dots, l\} : on_{i_2} = w_{i_2})$$

$$A_i^1 = \left(\forall i_1 : i+1 \le i_1 \le j-2, ed_{i_1} = \Re_{\text{cont.}}\left(\xrightarrow{1} v_{u_{i_1}} \xrightarrow{1} \dots \xrightarrow{1} v_{u_{j-2}} \xrightarrow{indeg(v_{u_{j-1}})} \mathcal{P}_c\right)\right)$$
$$\wedge \left(ed_{j-1} = \Re_{\text{cont.}}\left(\xrightarrow{indeg(v_{u_{j-1}})} \mathcal{P}_c\right)\right)$$
$$\wedge \left(\forall i_2 \in \{j,\dots,l\}, ed_{i_2} = \Re_{\text{cont.}}\left(\xrightarrow{w_{i_2}} v_{u_{i_2}} \xrightarrow{w_{i_2+1}} \dots \xrightarrow{w_l} v_{u_l}\right)\right)$$
$$A_i^2 = \left(\forall i_1 : i+1 \le i_1 \le j-2, v_{u_{(i_1-1)}} = prev\left(\xrightarrow{1} v_{u_{i_1}} \xrightarrow{1} \dots \xrightarrow{1} v_{u_{j-2}} \xrightarrow{indeg(v_{u_{j-1}})} \mathcal{P}_c, ed'_{i_1}\right)\right)$$
$$\wedge \left(v_{u_{j-2}} = prev\left(\xrightarrow{indeg(v_{u_{j-1}})} \mathcal{P}_c, ed'_{j-1}\right)\right)$$
$$\wedge \left(\forall i_2 \in \{j,\dots,l\}, v_{u_{(i_2-1)}} = prev\left(\xrightarrow{w_{i_2}} v_{u_{i_2}} \xrightarrow{w_{i_2+1}} v_{u_{(i_2+1)}} \xrightarrow{w_{i_2+2}} \dots \xrightarrow{w_l} v_{u_l}, ed'_{i_2}\right)\right)$$

We define $A_i = A_i^0 \wedge A_i^1 \wedge A_i^2 \wedge F^0_{basis}$, and for $j-k \le i \le j-2$, we define $A'_i = A_i \wedge (indeg(v_{u_i}) > 0)$, and finally $A'_{j-k-1} = A_{j-k-1} \wedge \left(indeg(v_{u_{j-k-1}} = 0)\right)$. Similarly to the proof of Claim 4, one can see that we can write sum_k as follows:

$$\begin{aligned} sum_k &= \Pr_{\mathbf{Strgy}'_{j-1}}\left[(u_0 = \dots = u_{j-k-2} = 0) \wedge A'_{j-k-1}\right] \\ &= (\frac{1}{1+2nP^2})^{j-k-1} \cdot \Pr_{\mathbf{Strgy}'_{j-1}}\left[A'_{j-k-1}\right] \end{aligned}$$

Thus to prove (28), we just need to prove the following lemma:

Lemma 13. *For all $0 \le k \le j-1$, it holds*

$$\Pr_{\mathit{Strgy}'_{j-1}}\left[A'_{j-k-1}\right] = (\frac{2}{1+2nP^2})^k \cdot \Pr_{\mathit{Strgy}'_{j-1}}[F_{basis}] \cdot \Pr_{\mathit{Strgy}'_{j-1}}[V_1 = k \mid F_{basis}]$$

Proof of Lemma 13.

Note that we have

$$A'_{j-k-1} \subseteq A'_{j-k} \subseteq \dots \subseteq A'_{j-2} \subseteq F_{basis}.$$

Based on this, we expand the probability as follows:

$$\begin{aligned} \Pr_{\mathbf{Strgy}'_{j-1}}\left[A'_{j-k-1}\right] = &\Pr_{\mathbf{Strgy}'_{j-1}}\left[A'_{j-k-1} \mid A'_{j-k}\right] \cdot \Pr_{\mathbf{Strgy}'_{j-1}}\left[A'_{j-k} \mid A'_{j-k+1}\right] \\ &\cdots \Pr_{\mathbf{Strgy}'_{j-1}}\left[A'_{j-3} \mid A'_{j-2}\right] \cdot \Pr_{\mathbf{Strgy}'_{j-1}}\left[A'_{j-2} \mid F_{basis}\right] \cdot \Pr_{\mathbf{Strgy}'_{j-1}}[F_{basis}]. \end{aligned}$$

For $j-k \le i \le j-3$, we can write:

$$\begin{aligned} \Pr_{\mathbf{Strgy}'_{j-1}}\left[A'_i \mid A'_{i+1}\right] &= \Pr_{\mathbf{Strgy}'_{j-1}}\left[A_i \wedge indeg(v_{u_i}) > 0 \mid A'_{i+1}\right] \\ &= \Pr_{\mathbf{Strgy}'_{j-1}}\left[A_i \mid A'_{i+1}\right] \cdot \Pr_{\mathbf{Strgy}'_{j-1}}\left[indeg(v_{u_i}) > 0 \mid A_i \wedge A'_{i+1}\right] \\ &= \Pr_{\mathbf{Strgy}'_{j-1}}\left[A_i \mid A'_{i+1}\right] \cdot \Pr_{\mathbf{Strgy}'_{j-1}}[indeg(v_{u_i}) > 0 \mid A_i]. \end{aligned}$$

The last equality follows from the fact that $A_i \subseteq A'_{i+1}$. Similarly, we can show that

$$\Pr_{\mathbf{Strgy}'_{j-1}}\left[A'_{j-k-1} \mid A'_{j-k}\right] = \Pr_{\mathbf{Strgy}'_{j-1}}\left[A_{j-k-1} \mid A'_{j-k}\right] \cdot \Pr_{\mathbf{Strgy}'_{j-1}}\left[indeg(v_{u_{j-k-1}}) = 0 \mid A_{j-k-1}\right]$$

$$\Pr_{\mathbf{Strgy}'_{j-1}}\left[A'_{j-2} \mid F_{basis}\right] = \Pr_{\mathbf{Strgy}'_{j-1}}\left[A_{j-2} \mid F_{basis}\right] \cdot \Pr_{\mathbf{Strgy}'_{j-1}}\left[indeg(v_{u_{j-2}}) > 0 \mid A_{j-2}\right]$$

Now if we define

$$\begin{aligned} product_1 = & \Pr_{\mathbf{Strgy}'_{j-1}}\left[indeg(v_{u_{j-2}}) > 0 \mid A_{j-2}\right] \cdot \Pr_{\mathbf{Strgy}'_{j-1}}\left[indeg(v_{u_{j-3}}) > 0 \mid A_{j-3}\right] \cdot \\ & \cdots \Pr_{\mathbf{Strgy}'_{j-1}}\left[indeg(v_{u_{j-k}}) > 0 \mid A_{j-k}\right] \cdot \Pr_{\mathbf{Strgy}'_{j-1}}\left[indeg(v_{u_{j-k-1}}) = 0 \mid A_{j-k-1}\right] \\ product_2 = & \Pr_{\mathbf{Strgy}'_{j-1}}\left[A_{j-2} \mid F_{basis}\right] \cdot \Pr_{\mathbf{Strgy}'_{j-1}}\left[A_{j-3} \mid A'_{j-2}\right] \cdots \Pr_{\mathbf{Strgy}'_{j-1}}\left[A_{j-k-1} \mid A'_{j-k}\right], \end{aligned}$$

we have

$$\Pr_{\mathbf{Strgy}'_{j-1}}\left[A'_{j-k-1}\right] = product_1 \cdot product_2 \cdot \Pr_{\mathbf{Strgy}'_{j-1}}\left[F_{basis}\right]. \tag{29}$$

Now similarly to the proof of Lemma 9 one can derive the following equality formulas:

$$\Pr_{\mathbf{Strgy}'_{j-1}}\left[A_h \mid A'_{h+1}\right] = \frac{2}{1+2nP^2}, \text{for } j-k-1 \le h \le j-3 \tag{30}$$

$$\Pr_{\mathbf{Strgy}'_{j-1}}\left[A_{j-2} \mid F_{basis}\right] = \frac{2}{1+2nP^2} \tag{31}$$

$$\Pr_{\mathbf{Strgy}'_{j-1}}\left[indeg(v_{u_i}) > 0 \mid A_i\right] = \Pr_{\mathbf{Strgy}'_{j-1}}\left[V_1 > j-i-1 \mid V_1 > j-i-2\right], \text{for } j-k \le i \le j-3 \tag{32}$$

$$\Pr_{\mathbf{Strgy}'_{j-1}}\left[indeg(v_{u_{j-k-1}}) = 0 \mid A_{j-k-1}\right] = \Pr_{\mathbf{Strgy}'_{j-1}}\left[V_1 = k \mid V_1 > k-1\right] \tag{33}$$

$$\Pr_{\mathbf{Strgy}'_{j-1}}\left[indeg(v_{u_{j-2}}) > 0 \mid A_{j-2}\right] = \Pr_{\mathbf{Strgy}'_{j-1}}\left[V_1 > 1 \mid F_{basis}\right] \tag{34}$$

As a result, we can simplify $product_1$ and $product_2$ as follows, completing the proof of Lemma 13:

$$\begin{aligned} product_1 &= \Pr_{\mathbf{Strgy}'_{j-1}}\left[V_1 = k \mid F_{basis}\right] \\ product_2 &= \left(\frac{2}{1+2nP^2}\right)^k \end{aligned}$$

□

Now the proofs of Lemma 12 and Claim 7 are complete.

□

□

C.7.2 Proof of Claim 8 The idea of the proof is exactly like that of Claim 2, therefore we omit most of the details. Letting

$$D'(\eta) = \left| \Pr_{\mathbf{Strgy}_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \mid B^0_{total}\right] - \Pr_{\mathbf{Strgy}_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \mid B^1_{total}\right] \right|,$$

we need to prove that if $S_0 = \Pr[B^0_{total}] > negl(\eta)$ and $S_1 = \Pr[B^1_{total}] > negl(\eta)$, then $D'(\cdot)$ is a negligible function. To this end, we give the following lemma:

Lemma 14. *We have:*

1. If $\Pr[B^0_{total}] > negl(\eta)$ *and* $\Pr[B^1_{total}] > negl(\eta)$*, then*

$$(i) \qquad \left| \Pr_{\textit{\textbf{Strgy}}_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \mid B^0_{total}\right] - \Pr_{\textit{\textbf{Strgy}}'_{j-1}}\left[\mathcal{A}_{wstci}(\eta) = 1 \mid F^0_{total}\right] \right| = negl(\eta)$$

$$(ii) \qquad \left| \Pr_{\textit{\textbf{Strgy}}_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \mid B^1_{total}\right] - \Pr_{\textit{\textbf{Strgy}}'_{j-1}}\left[\mathcal{A}_{wstci}(\eta) = 1 \mid F^1_{total}\right] \right| = negl(\eta)$$

2.

$$\Pr_{\textit{\textbf{Strgy}}'_{j-1}}\left[\mathcal{A}_{wstci}(\eta) = 1 \mid F^0_{total}\right] = \Pr_{\textit{\textbf{Strgy}}'_{j-1}}\left[\mathcal{A}_{wstci}(\eta) = 1 \mid F^1_{total}\right]$$

Note that the above lemma immediately implies that D' is negligible. Thus we turn our attention to proving the above lemma. The idea of the proof for part (2) of the above lemma is essentially similar to that of Claim 6; namely, using the same argument presented there, one can also show here for game $\mathbf{Strgy}'_{j-1}$ that the view of $\mathcal{A}_{wstci}$ is identically distributed under the occurrences of F^0_{total} and F^1_{total}, yielding the desired equality. We omit the details.

For part (1), similarly to the proof of Claim 5, the proof of both (i) and (ii) follow the same line of reasoning and, hence, we present an overview of the proof for (i). To prove (i), we can show that

(a)

$$\Pr_{\mathbf{Strgy}'_{j-1}}\left[\mathcal{A}_{wstci}(\eta) = 1 \mid F^0_{total}\right] = \Pr_{\mathbf{Strgy}''_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \mid F^0_{total}\right]$$

(b) If $\Pr[B^0_{total}] > negl(\eta)$, then

$$\left| \Pr_{\mathbf{Strgy}_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \mid B^0_{total}\right] - \Pr_{\mathbf{Strgy}''_0}\left[\mathcal{A}_{wstci}(\eta) = 1 \mid F^0_{total}\right] \right| = negl(\eta),$$

where $\mathbf{Strgy}''_0$ is the game defined in the proof of Lemma 11. Now the proofs for (a) and (b) above, again, proceed in a very similar manner to those of, respectively, parts (a) and (b) of Lemma 10. This concludes the proofs of Lemma 14 and Claim 8. □

□

Now Claim 7 and Claim 8 imply the result of Lemma 5. □

C.8 Proof of Corollary 1

Proof of Part 1. From Equation (7) we have

$$\varDelta_{\text{cca-sim}} = \left|\Pr\left[\mathcal{A}_{cca}(\eta) = 1 \ \wedge \ \overline{Bad} \ \wedge \ b = 0\right] - \Pr\left[\mathcal{A}_{cca}(\eta) = 1 \ \wedge \ \overline{Bad} \ \wedge \ b = 1\right]\right|.$$

To prove Part 1, it suffices to show that

$$\left(\overline{Bad} \wedge b = 0\right) = \left(\theta_0 \vee \theta_0^{\text{wstci}}\right) \tag{35}$$

$$\left(\overline{Bad} \wedge b = 1\right) = \left(\theta_1 \vee \theta_1^{\text{wstci}}\right) \tag{36}$$

We give the proof for (35), and the proof for (36) is obtained similarly. If E is one of the events $\overline{Bad}$, θ_0, or θ_0^{wstci}, we define $\mathcal{S}(E)$ to be the set of all events OR'ed together to form E, where each of such events is either of the form $\psi\text{-max}^{(.,\dots,.)}_{(.,\dots,.)}[.,\dots,.]$ or of the form $\psi\text{-}\overline{max}^{(.,\dots,.)}_{(.,\dots,.)}[.,\dots,.]$.

(Henceforth, we refer to the former as ψ-max-type and to the latter as ψ-$\overline{max}$-type of events.) To show the result, it suffices to show that

$$E \in \mathcal{S}(\overline{Bad}) \Leftrightarrow (E \wedge b = 0) \in \mathcal{S}(\theta_0) \cup \mathcal{S}(\theta_0^{\text{wstci}})$$

The ($\Leftarrow$) direction is obvious; if $(E \wedge b = 0) \in \mathcal{S}(\theta_0) \cup \mathcal{S}(\theta_0^{\text{wstci}})$, then obviously $E \in \overline{Bad}$; this follows from the very definitions of $\mathcal{S}(\theta_0)$ and $\mathcal{S}(\theta_0^{\text{wstci}})$.

The other direction is less trivial and requires careful reasoning. Suppose $E \in \overline{Bad}$; we have the following cases:

(i) E is a ψ-$\overline{max}$-type event; namely, there exists $i \in \{2, \ldots, l\}$, and $d_l, \ldots, d_i \in [P]$, and $w_l \in [d_l], \ldots, w_i \in [d_i]$, and $b'_l, \ldots, b'_i \in \{0, 1\}$ such that $E = \psi\text{-}\overline{max}^{(d_l,\ldots,d_i)}_{(w_l,\ldots,w_i)}[b'_l, \ldots, b'_i]$; then obviously $(E \wedge b = 0) \in \mathcal{S}(B^0_{total,i})$, and consequently, $(E \wedge b = 0) \in \mathcal{S}(\theta_0)$. Moreover if $E = \psi - d_l$ then $(E \wedge b = 0) \in S(C^0_{total,l+1})$.

(ii) $E = \psi\text{-}max^{(d_l,\ldots,d_{i+1},d_i)}_{(w_l,\ldots,w_{i+1},w_i)}[b'_l, \ldots, b'_{i+1}, 1]$, where $1 \le i \le l$, and $d_l, \ldots, d_i \in [P]$, and $w_l \in [d_l], \ldots, w_i \in [d_i]$, and $b'_l, \ldots, b'_{i+1} \in \{0, 1\}$. Now it is easy to see that if $w_i < d_i$, then $(E \wedge b = 0) \in \mathcal{S}(A^0_{total,i})$. Otherwise, we have two cases: (a) $w_i = d_i$ and $i < l$: in this case $(E \wedge b = 0) \in \mathcal{S}(B'^0_{total,i+1})$, or (b) $w_i = d_i$ and $i = l$: in this case $(E \wedge b = 0) \in \mathcal{S}(C'^0_{total,l+1})$. Thus, in either case, we have $(E \wedge b = 0) \in \mathcal{S}(\theta_0)$.

(iii) $E = \psi\text{-}max^{(d_l,\ldots,d_{i+1},d_i)}_{(w_l,\ldots,w_{i+1},w_i)}[b'_l, \ldots, b'_{i+1}, 0]$, where $b'_l, \ldots, b'_{i+1} \in \{0, 1\}$ and $1 \le i \le l$, and $d_l, \ldots, d_i \in [P]$, and $w_l \in [d_l], \ldots, w_{i+1} \in [d_{i+1}]$ and $2 \le w_i \le d_i$. In this case, we have $(E \wedge b = 0) \in \mathcal{S}(A'^0_{total,i})$, and again, $(E \wedge b = 0) \in \mathcal{S}(\theta_0)$.

(iv) $E = \psi\text{-}max^{(d_l,\ldots,d_{i+1},d_i)}_{(w_l,\ldots,w_{i+1},1)}[b'_l, \ldots, b'_{i+1}, 0]$, where $b'_l, \ldots, b'_{i+1} \in \{0, 1\}$ and $1 \le i \le l$, and $d_l, \ldots, d_i \in [P]$, and $w_l \in [d_l], \ldots, w_{i+1} \in [d_{i+1}]$. Now if for all $i + 1 \le k \le l$, both $w_k = 1$ and $b'_k = 0$ hold, then $(E \wedge b = 0) \in \mathcal{S}(\theta_0^{\text{wstci}})$. Otherwise, let $k_1 \in \{i + 1, \ldots, l\}$ be the smallest number for which either $w_{k_1} > 1$ or $b'_{k_1} = 1$ holds (they could both happen). It follows from the definition that for all $k_2 \in \{i, \ldots, k_1 - 1\}$ both $w_{k_2} = 1$ and $b'_{k_2} = 0$ hold. Now if $b'_{k_1} = 1$, then there are two cases: either $w_{k_1} < d_{k_1}$ in which case we have $(E \wedge b = 0) \in \mathcal{S}(A^0_{total,k_1})$, or $w_{k_1} = d_{k_1}$ in which case we have $(E \wedge b = 0) \in \mathcal{S}(B'^0_{total,k_1+1})$ if $k_1 < l$, and $(E \wedge b = 0) \in \mathcal{S}(C'^0_{total,l+1})$ if $k_1 = l$. Otherwise if $b'_{k_1} = 0$, then we must have $w_{k_1} > 1$, in which case we will obtain $(E \wedge b = 0) \in \mathcal{S}(A'^0_{total,k_1})$. Thus, we always have $(E \wedge b = 0) \in \mathcal{S}(\theta_0^{\text{wstci}}) \cup \mathcal{S}(\theta_0)$.

Now the proof is complete.

□

<u>Proof of Part 2.</u> Recalling that

$$\theta_0^{\text{wstci}} = \bigvee_{i=1}^{l} \bigvee_{\substack{d_l \in [P] \\ \vdots \\ d_i \in [P]}} \Psi{-}max^{(d_l,\ldots,d_i)}_{(1,\ldots,1)}\,[b_l = 0;\ \ldots;\ b_i = 0] \wedge b = 0$$

$$\theta_1^{\text{wstci}} = \bigvee_{i=1}^{l} \bigvee_{\substack{d_l \in [P] \\ \vdots \\ d_i \in [P]}} \Psi{-}max^{(d_l,\ldots,d_i)}_{(1,\ldots,1)}\,[b_l = 1;\ \ldots;\ b_i = 1] \wedge b = 1,$$

we have to show that

$$\Pr_{\mathbf{Strgy}_0}\left[\theta_0^{\text{wstci}}\right] = (\frac{1}{1+2nP^2})^{l+1} + negl(\eta) \tag{37}$$

$$\Pr_{\mathbf{Strgy}_0}\left[\theta_1^{\text{wstci}}\right] = (\frac{1}{1+2nP^2})^{l+1} + negl(\eta). \tag{38}$$

We prove (37); the proof for (38) is also obtained similarly. For $1 \le k \le l$ and $D = (d_l, \ldots, d_k) \in [P]^{l-k+1}$ we define

$$\theta_0^{k,D} = \Psi\text{-}max_{(1,\ldots,1)}^{(d_l,\ldots,d_k)}.$$

Now to prove (37), it suffices to show that

$$\sum_{k=1}^{l}\sum_{D\in[P]^{l-k+1}}(\frac{1}{2})^{l-k+2}\cdot\Pr_{\mathbf{Strgy}_0}\left[\theta_0^{k,D}\mid b_l=0;\ \ldots;\ b_k=0;\ b=0\right] = (\frac{1}{1+2nP^2})^{l+1} + negl(\eta). \tag{39}$$

We define a new security game, $\mathbf{Strgy}_{ci,0}$, which executes exactly as in the WSTCI game when the challenge bit is 0. Namely, under $\mathbf{Strgy}_{ci,0}$, we randomly sample $\{(pk_i, sk_i)\}_{1\le i\le n}$ and $\{nc_i\}_{1\le i\le n}$, and proceed in the same way as that of the WSTCI game when the challenge bit is 0. Moreover, to discuss events $\theta_0^{k,D}$'s under $\mathbf{Strgy}_{ci,0}$, we additionally sample the following parameters as before (note that none of these coming parameters take part in the randomness of the replies given to adversary's queries): variables $u_0, \ldots, u_l$ are sampled in the same way as that of $\mathbf{Strgy}_0$, and letting s denote the smallest index where $u_s \neq 0$, for each $s < t \le l$ we sample three parameters (on_t, ed_t, ed'_t) from the same distributions used in $\mathbf{Strgy}_0$. Now to prove (39), we show that:

(A)

$$\left|\Pr_{\mathbf{Strgy}_0}\left[\theta_0^{k,D}\mid b_l=0;\ \ldots;\ b_k=0;\ b=0\right] - \Pr_{\mathbf{Strgy}_{ci,0}}\left[\theta_0^{k,D}\right]\right| = negl(\eta), \tag{40}$$

(B)

$$\sum_{k=1}^{l}\sum_{D\in[P]^{l-k+1}}(\frac{1}{2})^{l-k+2}\cdot\Pr_{\mathbf{Strgy}_{ci,0}}\left[\theta_0^{k,D}\right] = (\frac{1}{1+2nP^2})^{l+1}. \tag{41}$$

The proof of (A) follows using the same arguments as those used in Lemma 6 (or Lemma 11). Namely to argue that the difference of probability values is negligible, one has to show that the probability that $Failed\text{-}Dec$ occurs under $\mathbf{Strgy}_{ci,0}$ is negligible. We omit the details.

To prove (B), we first define L_{first}, a random variable which (intuitively) corresponds to the length of a random path selected uniformly from the set of paths which are strongly coinductively continuable and end in the challenge node. More formally, L_{first} takes values in $\{1, \ldots, l\}$ according to the following distribution: after the execution of $\mathbf{Strgy}_{ci,0}$, denoting by v_{ch} the random variable corresponding to the challenge node, the probability that $L_{\text{first}} = 1$ is the probability that $indeg(v_{i_1}) = 0$, where v_{i_1} is picked uniformly at random from $\Re_{\text{cont.}}(\xrightarrow{1} v_{\text{ch}})$. For $r > 1$, the probability that $L_{\text{first}} = r$ is defined to be the probability that $q = r$ after the execution of the following:

1: $q \leftarrow 1$
2: **while** $indeg(v_{i_q} > 0)$ **do**

Pick $v_{i_{q+1}}$ uniformly at random from $\Re_{\text{cont.}}(\xrightarrow{1} v_{i_q} \xrightarrow{1} \ldots \xrightarrow{1} v_{i_1} \xrightarrow{1} v_{\text{ch}})$
$q \leftarrow q+1$
end while

We now prove that

$$sum_k \overset{\triangle}{=} \sum_{D \in [P]^{l-k+1}} \Pr_{\mathbf{Strgy}_{ci,0}} \left[\theta_0^{k,D}\right] \qquad \text{(for } 1 \leq k \leq l)$$
$$= (\frac{2}{1+2nP^2})^{l-k+2} \cdot (\frac{1}{1+2nP^2})^{k-1} \cdot \Pr\left[L_{\text{first}} = l-k+1\right] \tag{42}$$

Now it is easy to see that the above equality formula yields the result claimed in (B). The proof for (42) follows along the same line of reasoning as that of Lemma 7; namely, first for $0 \leq i \leq l-1$, we define the following random variables:

$$A_i^0 = (s \leq i) \wedge (\forall i_2 \in \{i+1, \ldots, l\} : on_{i_2} = 1) \wedge (v_{u_l} = v_{\text{ch}})$$
$$A_i^1 = \left(\forall i_2 \in \{i+1, \ldots, l\}, ed_{i_2} = \Re_{\text{cont.}}\left(\xrightarrow{1} v_{u_{i_2}} \xrightarrow{1} v_{u_{i_2+1}} \xrightarrow{1} \ldots \xrightarrow{1} v_{u_l}\right)\right)$$
$$A_i^2 = \left(\forall i_2 \in \{i+1, \ldots, l\}, v_{u_{(i_2-1)}} = prev\left(\xrightarrow{1} v_{u_{i_2}} \xrightarrow{1} v_{u_{(i_2+1)}} \xrightarrow{1} \ldots \xrightarrow{1} v_{u_l}, ed'_{i_2}\right)\right)$$

Now if we define $A_i \overset{\triangle}{=} A_i^0 \wedge A_i^1 \wedge A_i^2 \wedge indeg(v_{u_i} = 0)$, similarly to the proof of Lemma 9, one can see that it holds:

$$sum_k = \Pr_{\mathbf{Strgy}_{ci,0}} \left[(u_0 = \cdots = u_{k-2} = 0) \wedge A_{k-1}\right]$$
$$= (\frac{1}{1+2nP^2})^{k-1} \cdot \Pr_{\mathbf{Strgy}_{ci,0}} \left[A_{k-1}\right], \tag{43}$$
$$\Pr_{\mathbf{Strgy}_{ci,0}} \left[A_{k-1}\right] = (\frac{2}{1+2nP^2})^{l-k+2} \cdot \Pr\left[L_{\text{first}} = l-k+1\right].$$

We omit the details as to how the above equalities follow, as they are highly similar to those of the proof of Lemma 9. From the above two equalities, the claimed equality of (42) follows, and the proof is complete.

□

Proof of Part 3. The proofs of both equalities are entirely similar, and their main proof ideas have been extensively discussed earlier. Thus, here we just sketch an overview of the proof for the first equality. Namely we show

$$\left| \Pr_{\mathbf{wstci}} \left[\mathcal{A}_{wstci}(\eta) = 1 \mid b = 0\right] - \Pr_{\mathbf{Strgy}_0} \left[\mathcal{A}_{wstci}(\eta) = 1 \mid \theta_0^{\text{wstci}}\right] \right| = negl(\eta). \tag{44}$$

Now recalling the notations introduced in the proof of part (2) of this corollary, the following claimed equalitues imply the equality of (44):

$$\Pr_{\mathbf{wstci}} \left[\mathcal{A}_{wstci}(\eta) = 1 \mid b = 0\right] = \Pr_{\mathbf{Strgy}_{ci,0}} \left[\mathcal{A}_{wstci}(\eta) = 1\right], \tag{45}$$
$$\Pr_{\mathbf{Strgy}_{ci,0}} \left[\mathcal{A}_{wstci}(\eta) = 1\right] = \Pr_{\mathbf{Strgy}_{ci,0}} \left[\mathcal{A}_{wstci}(\eta) = 1 \mid \theta_0^{\text{wstci}}\right], \tag{46}$$

$$\left| \Pr_{\mathbf{Strgy}_{ci,0}} \left[\mathcal{A}_{wstci}(\eta) = 1 \mid \theta_0^{\text{wstci}} \right] - \Pr_{\mathbf{Strgy}_0} \left[\mathcal{A}_{wstci}(\eta) = 1 \mid \theta_0^{\text{wstci}} \right] \right| = negl(\eta). \tag{47}$$

Equality (45) trivially holds by the very definition of $\mathbf{Strgy}_{ci,0}$. Equality (46) also holds by considering the fact that the two events $(\mathcal{A}_{wstci}(\eta) = 1)$ and θ_0^{wstci} are independent under $\mathbf{Strgy}_{ci,0}$. (Note that under $\mathbf{Strgy}_{ci,0}$, the values of u_i's, on_i's, ed_i's, ed_i''s do not influence the randomness of replies given to adversary's queries.) Finally, (47) can also be proved using similar arguments to those used in the proof of part (b) Lemma 10; namely, we expand

$$\begin{aligned} p_1 &\triangleq \Pr_{\mathbf{Strgy}_0} \left[\mathcal{A}_{wstci}(\eta) = 1 \mid \theta_0^{\text{wstci}} \right] \\ &= \Pr_{\mathbf{Strgy}_0} \left[\mathcal{A}_{wstci}(\eta) = 1 \mid \overline{\textit{Failed-Dec}} \wedge \theta_0^{\text{wstci}} \right] \cdot \Pr_{\mathbf{Strgy}_0} \left[\overline{\textit{Failed-Dec}} \mid \theta_0^{\text{wstci}} \right] \\ &\quad + \Pr_{\mathbf{Strgy}_0} \left[\mathcal{A}_{wstci}(\eta) = 1 \wedge \textit{Failed-Dec} \mid \theta_0^{\text{wstci}} \right], \end{aligned}$$

$$\begin{aligned} p_2 &\triangleq \Pr_{\mathbf{Strgy}_{ci,0}} \left[\mathcal{A}_{wstci}(\eta) = 1 \mid \theta_0^{\text{wstci}} \right] \\ &= \Pr_{\mathbf{Strgy}_{ci,0}} \left[\mathcal{A}_{wstci}(\eta) = 1 \mid \overline{\textit{Failed-Dec}} \wedge \theta_0^{\text{wstci}} \right] \cdot \Pr_{\mathbf{Strgy}_{ci,0}} \left[\overline{\textit{Failed-Dec}} \mid \theta_0^{\text{wstci}} \right] \\ &\quad + \Pr_{\mathbf{Strgy}_{ci,0}} \left[\mathcal{A}_{wstci}(\eta) = 1 \wedge \textit{Failed-Dec} \mid \theta_0^{\text{wstci}} \right]. \end{aligned}$$

From part (2) of this corollary we have $\Pr_{\mathbf{Strgy}_0} \left[\theta_0^{\text{wstci}} \right] > negl(\eta)$. From the same part, we also have that $\Pr_{\mathbf{Strgy}_{ci,0}} \left[\theta_0^{\text{wstci}} \right] > negl(\eta)$. Also similarly as before, we can prove that $\Pr_{\mathbf{Strgy}_0} [\textit{Failed-Dec}] = negl(\eta)$ and $\Pr_{\mathbf{Strgy}_{ci,0}} [\textit{Failed-Dec}] = negl(\eta)$. Consequently, it follows that:

$$\Pr_{\mathbf{Strgy}_0} \left[\mathcal{A}_{wstci}(\eta) = 1 \wedge \textit{Failed-Dec} \mid \theta_0^{\text{wstci}} \right] = negl(\eta)$$

$$\Pr_{\mathbf{Strgy}_{ci,0}} \left[\mathcal{A}_{wstci}(\eta) = 1 \wedge \textit{Failed-Dec} \mid \theta_0^{\text{wstci}} \right] = negl(\eta)$$

$$\left| \Pr_{\mathbf{Strgy}_0} \left[\overline{\textit{Failed-Dec}} \mid \theta_0^{\text{wstci}} \right] - \Pr_{\mathbf{Strgy}_{ci,0}} \left[\overline{\textit{Failed-Dec}} \mid \theta_0^{\text{wstci}} \right] \right| = negl(\eta).$$

Also using the same techniques described earlier, we can easily verify that

$$\Pr_{\mathbf{Strgy}_0} \left[\mathcal{A}_{wstci}(\eta) = 1 \mid \overline{\textit{Failed-Dec}} \wedge \theta_0^{\text{wstci}} \right] = \Pr_{\mathbf{Strgy}_{ci,0}} \left[\mathcal{A}_{wstci}(\eta) = 1 \mid \overline{\textit{Failed-Dec}} \wedge \theta_0^{\text{wstci}} \right].$$

Now the proof is complete.

□

Proof of Part 4. Follows from the very description of the SIMULATION phase of $\mathcal{A}_{cca}$, considering that for both $h = 0$ and $h = 1$ we have $\theta_h^{wstci} \wedge Bad = \emptyset$.

□

Proof of Part 5. Follows immediately from the previous parts of the corollary. □

C.9 WSTCI Game and Nonce Challenge

The proof that we have given so far is for the case where the adversary makes only a *key-challenge* query in his **guessing** phase. The proof for the other case (i.e. the case where the adversary is allowed to make only a *nonce-challenge* query) is obtained in a very similar manner with the following simple modifications. First in the SETUP phase, besides $u_s, \ldots, u_l$, we also sample $u_{l+1} \leftarrow \{1, \ldots, n\}$, with associated parameters $(on_{l+1}, kd_{l+1}, kd'_{l+1})$, and, further, require the following conditions: (a) the number of keys which encrypt the on_{l+1}'th occurrence of $x_{u_{l+1}}$ and which are coinductively irrecoverable is kd_{l+1} (we call this set of keys A), (b) k_{u_l} is the kd'_{l+1}'th lowest key among A, and (c) $x_{u_{l+1}} = x_{ch}$, where x_{ch} is the nonce challenged by the adversary in the **guessing** phase. We continue to sample the same number of b_i's (from the same distributions) under the new reduction, and we answer to encryption queries in the same way as $\mathbf{Strgy}_0$, with the following modification: for any $o_1 < on_{l+1}$ and $o_2 > on_{l+1}$, we replace x_{u_{l+1},o_1} with $fnc_{u_{l+1}}$ (i.e. fake nonce) and x_{u_{l+1},o_2} with $nc_{u_{l+1}}$ (real nonce), where $x_{i,o}$ denotes the o'th occurrence of x_i. For $x_{u_{l+1},on_{l+1}}$ if $b_{l+1} \neq b_l$ then we replace it with $fnc_{u_{l+1}}$, and, otherwise, e replace it with $nc_{u_{l+1}}$. Finally for *challenge*(x_{ch}), if $x_{u_{l+1}} \neq x_{ch}$, then we halt the execution (a failing situation), and otherwise, we *always* return $nc_{u_{l+1}}$. Now using the approach used to prove Lemmas 4, 5, we can easily verify that the two events which remain at the end of the proofs (i.e. those which correspond to θ^0_{wstci} and θ^1_{wstci}) are the following two events: (1) the one in which all keys/nonces are replaced with their real values (in particular, $nc_{u_{l+1}}$ is used both to replace $x_{u_{l+1}}$'s and as the value returned in response to *challenge*$(x_{u_{l+1}})$), and (2) the one in which all keys and nonces, except $x_{u_{l+1}}$, are replaced with their real values, and for $x_{u_{l+1}}$ the following holds: all occurrences of $x_{u_{l+1}}$ (during the **interaction** phase) are replaced with $fnc_{u_{l+1}}$ and $nc_{u_{l+1}}$ is returned in response to *challenge*$(x_{u_{l+1}})$. We omit the details. This concludes the proof.

C.10 Proof of Lemma 3

We formulate a new variation of the WSTCI game, which we call $WSTCI_1$, as follows: the game again proceeds in three phases: the setup phase, the interaction phase, and the guessing phase. The setup phase of WSTCI_1 is exactly similar to that of the WSTCI game; namely, the values $\{(pk_j, sk_j)\}_{1\leq j\leq n}$, $\{nc_j\}_{1\leq j\leq n}$ and the challenge bit b are sampled; symbols $\{(k_j, k_j^{-1})\}_{1\leq j\leq n}$ and $\{x_j\}_{1\leq j\leq n}$ are introduced; and finally the computational mapping τ is accordingly initialized. In the interaction phase, the adversary first arbitrarily chooses a challenger's private key, say k_i^{-1}, and then begins adaptively interacting with the challenger by issuing the following queries, with the only restriction that at the end of the interaction phase, it must hold that $k_i^{-1} \notin closure_c(eval\text{-}exp^1)$. Before discussing how responses to adversary's queries are made, we define two key-renaming functions R_0 and R_1 as follows: $R_0[k_h^{-1}] = k_h^{-1}$, for all $1 \leq h \leq n$ (i.e. the identity function), and $R_1[k_r^{-1}] = k_r^{-1}$, if $r \neq i$, and $R_1[k_i^{-1}] = {k'}_i^{-1}$. The adversary's queries as usual are:

- Corruption: A *corruption* query *corrupt*(j) is answered by returning sk_j.
- Encryption: In response to an *encryption* query *encrypt*(e, j), the challenger does the following: if $b = 0$, the challenger returns $c \leftarrow [\![\{e\}_{k_j}]\!]_\tau$, and if $b = 1$, he returns $c \leftarrow [\![\{e\}_{k_j}]\!]_{\tau'}$, where $\tau'(s) = \tau(s)$ if $s \neq k_i^{-1}$, and $\tau'(k_i^{-1}) = fsk_i$, with $fsk_i \leftarrow Gen_1(\eta)$ being chosen independently from all values sampled in the setup phase. (Here Gen_1 denotes the second component, that is the secret key, of the key-generation algorithm.)
- Decryption: As in the WSTCI game, a *decryption* query *decrypt*(c, j) is replied to with $\perp$ if any of the following conditions hold, and with $Dec(c, sk_j)$, otherwise: (a) (c, j) is coinductively visible in *eval-exp*; or (b) there exists $(\{e\}_{k_p}, E_p) \in eval\text{-}exp$ such that $\{e\}_{k_p}$ has a subexp $\{e_h\}_{k_h}$ where $R_b\,[\{e_h\}_{k_h}]$ is type-1 and c is the corresponding computational image of $\{e_h\}_{k_h}$ in E_p.

– A Subexp-Testing query is also treated similarly as before: namely $subexp(\{e\}_{k_i}, bs_i, \{e'\}_{k_j}, bs_j)$ is answered by 1 if bs_j is the corresponding computational image of $\{e'\}_{k_j}$ in bs_i, and by 0, otherwise.

Finally, in the guessing phase, the adversary outputs his guess for b. Similarly to Definition 4, we can define l-WSTCT$_1$ security, for any constant l.

Intuitively, an adversary under the WSTCI$_1$ game is challenged to tell two worlds apart: one in which all occurrences of k_i^{-1} receive a fake value, and the other in which all of them receive the real value. By slightly modifying the proof of Lemma 2, one can also prove that CCA2 security implies l-WSTCT$_1$ security, for every constant l. For this, to reduce $\mathcal{A}_{\mathrm{WSTCT}_1}$, attacking the scheme in the sense of l-WSTCT$_1$, to $\mathcal{A}_{cca}$, attacking the scheme in the sense of CCA2, we just need to make the following two adjustments to Algorithm 1 (which is the SETUP phase of $\mathcal{A}_{cca}$): (1) We let u_l be the index of the challenger's key that $\mathcal{A}_{\mathrm{WSTCT}_1}$ decides to challenge in the beginning of the interaction phase and sample $u_0, \ldots, u_{l-1}$ as in Algorithm 1, and (2) b_{l+1} is no longer sampled under this new construction, and, instead, we let $b_l = 1$ and sample $b_{s+1}, \ldots, b_{l-1}$ as that in Algorithm 1. The rest of the construction stays unchanged. With these adjustments, the following statement follows:

$$\mathrm{CCA2} \Rightarrow \mathrm{WSTCT}_1 \tag{48}$$

As the next step, we introduce another game, called the *SEG game*. (Here SEG stands for subexp generation.) The SEG game proceeds in three phases: the setup phase, the interaction phase, and the *generation* phase (as opposed to the guessing phase). The setup phase runs similarly as in the WSTCT$_1$ game, except that under the SEG game no challenge bit is sampled. The interaction phase is also similar to that of $\mathrm{WSTCT}_1^{b=0}$ (i.e. the WSTCI$_1$ game when the WSTCI$_1$-challenge bit is 0), except that *subexp-testing* queries are no longer allowed. Namely, in the interaction phase, the adversary first chooses a challenger's key k_i^{-1}, and starts interacting with the challenger by making only *corruption*, *encryption*, and *decryption* queries, provided that it must always hold that $k_i^{-1} \notin closure_c(eval\text{-}exp^1)$. Finally in the generation phase, the adversary outputs $(\{e\}_{k_i}, E_i, \{e'\}_{k_j}, E_j)$; the output of the game is 1 (the adversary wins) if all the following conditions hold, and 0 otherwise: there exists $(\{e_p\}_{k_p}, E_p) \in eval\text{-}exp$ such that (1) $\{e\}_{k_i}$ is a subexp $\{e_p\}_{k_p}$ and is encrypted in $\{e_p\}_{k_p}$ only under keys whose decryption keys are in $closure_c(eval\text{-}exp^1)$ (i.e. $\{e\}_{k_i}$ is coinductively visible in $eval\text{-}exp$), (2) E_i is the corresponding computational image of $\{e\}_{k_i}$ in E_p, (3) $\{e'\}_{k_j} \sqsubseteq \{e\}_{k_i}$, (4) $\{e'\}_{k_j}$ is type-1, and (5) E_j is the corresponding computational image of $\{e'\}_{k_j}$ in E_p. We say that an encryption scheme provides SEG-security if for every $\mathcal{A}$, his probability of success is negligible.

Finally as the last game, we define a weaker version of the SEG game, which we call *SEG$_w$* game, which is exactly like SEG except that its interaction phase proceeds according to $\mathrm{WSTCT}_1^{b=0}$ (i.e. all occurrences of k_i^{-1} are replaced with the fake value fsk_i). Now it is not hard to prove that if an encryption scheme provides IND-CCA2 security, then it also provides SEG$_w$ security. Before presenting the statements formally, we give some explanation about the SEG (and SEG$_w$) games.

First without loss of generality we assume that for the adversary's output $(\{e\}_{k_i}, E_i, \{e'\}_{k_j}, E_j)$ the conditions (1), (2), (3) and (4) above always hold. Note that this is without loss of generality, because conditions (1), (3) and (4) can be trivially checked by the adversary, and condition (2) can also be checked by the adversary because of the following observation (that was also pointed out previously a few times): if $\{e_i\}_{k_i}$ is encrypted under a key k_v (where by assumption $k_v^{-1} \in closure_c(eval\text{-}exp^1)$) and even if the adversary does not know the computational value of k_v^{-1}, the adversary can obtain it by querying $corrupt(k_v^{-1})$; note that this last query does not change the symbolic knowledge of the adversary.

We now formally show the following statement which was informally stated above.

Lemma 15. *(1) IND-CCA2 $\Rightarrow$ SEG$_w$*
(2) IND-CCA2 $\Rightarrow$ SEG

Proof. Proof of (1) is relatively straightforward; namely, if $\mathcal{A}_{seg_w}$ attacks the scheme in the sense of SEG$_w$ and wins with a non-negligible probability, then one can build $\mathcal{A}_{cca}$ to win in a CCA2-indistinguishability experiment under key pk (whose matching secret key is unknown to the adversary) as follows: if $\mathcal{A}_{seg_w}$ chooses to challenge k_i^{-1}, then we use pk as the value for k_i, and after that, we generate a fake value fsk_i to replace k_i^{-1}'s, and freshly generated values for all other symbols of the challenger, thereby forming the concrete-evaluation function τ. Also, in the beginning, $\mathcal{A}_{cca}$ guesses the *encryption* query number and the "index of the subexps", which $\mathcal{A}_{seg_w}$'s output is going to depend on in the **generation** phase. (To stay in line with the above formalization, we assume that the (a priori unknown) encryption query is $encrypt(e_p, k_p)$ and also assume that $\{e\}_{k_i}$ and $\{e'\}_{k_j}$ are as above. Note that $\{e\}_{k_i}$ need not be guessed, as it is uniquely determinable from $\{e_p\}_{k_p}$ and *eval-exp*.) Now when $\mathcal{A}_{cca}$ has to reply to $encrypt(e_p, k_p)$, he generates $E_0, E_1 \leftarrow [\![\{e'\}_{k_j}]\!]_\tau$, and, based on E_0, E_1, he produces E'_0, E'_1, computational images of e. Now he submits (E'_0, E'_1) to his LOR-encryption-oracle to receive the challenge ciphertext c, and based on c, he properly builds c_p, a computational image of $\{e_p\}_{k_p}$ and returns c_p to $\mathcal{A}_{seg_w}$. Now using τ, he replies to the rest of $\mathcal{A}_{seg_w}$'s queries in its usual way. (No more query will be made to the CCA2-LOR-encryption oracle.) Note that, $\mathcal{A}_{seg_w}$ is *not* permitted to ask for decryption of c under the secret key of the oracle. (since it is an invalid *decryption* query, guaranteed by the assumption that $\{e\}_{k_i}$ is coinductively visible in $\{e_p\}_{k_p}$; actually this is the only reason we included that requirement for valid SEG outputs.) For decryption requests for all other ciphertexts, $\mathcal{A}_{cca}$ can decrypt them either by himself or through his decryption oracle. Now when $\mathcal{A}_{seg_w}$ returns his output in the **generation** phase, $\mathcal{A}_{cca}$ can use this to determine what message was encrypted under his CCA2-oracle. It is obvious that the probabilities of success of the two adversaries are polynomially related. We omit the details.

A similar approach taken for part (1) does not work for part (2) since k_i^{-1}, under the SEG game, has to be replaced under its real value, and, hence, we cannot directly simulate an $\mathcal{A}_{seg}$ adversary using an $\mathcal{A}_{cca}$ adversary. Rather, we do the following: we show that for every $\mathcal{A}_{seg}$ attacking the scheme in the sense of SEG, his probability of success, P_{seg}, is negligibly different from his probability of success, P_{seg_w}, which would have been obtained if run under SEG$_w$, thereby concluding from part (1) that P_{seg} must be negligible. Now why should it be that $|P_{seg} - P_{seg_w}| = negl(\eta)$? Because, otherwise, one can construct an adversary $\mathcal{A}_{wstct_1}$ to break the WSTCT$_1$-security of the scheme, and as a result of (48) its CCA2-security, reaching a contradiction. It is also immediate to see how one can construct $\mathcal{A}_{wstct_1}$ from $\mathcal{A}_{seg}$; namely, when $\mathcal{A}_{seg}$ challenges an adversary's key in his **interaction** phase, $\mathcal{A}_{wstct_1}$ starts by challenging the same key, and answers all $\mathcal{A}_{seg}$'s queries using his own oracle access. Note that $\mathcal{A}_{wstct_1}$ can perfectly simulate and answer to $\mathcal{A}_{seg}$'s oracle queries using his own oracle access, and depending on the secret bit b, it results in one of the two induced distributions that corresponds to $\mathcal{A}_{seg}$'s run either under SEG or under SEG$_w$. Now when $\mathcal{A}_{seg}$ returns his output in the **generation** phase, $\mathcal{A}_{wstct_1}$ queries this output from his, additional, *subexp-testing* oracle, and returns whatever the oracle returns. Now it is easy to see that $\mathcal{A}_{wstct_1}$'s advantage is exactly $|P_{seg} - P_{seg_w}|$, and this completes the proof.

□

Completing the Proof of Lemma 3: Suppose $\mathcal{A}$ provides l-CI-security, but not l-WSTCI-security. Thus it must be the case that, with a non-negligible probability, $\mathcal{A}$ makes a decryption query that is valid under the l-CI-game, but not under the l-WSTCI-game. That is, with a non-negligible probability, at some point, $\mathcal{A}$ produces a *decryption* query $decrypt(E_j, w)$ for which it holds that there exists $(\{e_p\}_{k_p}, E_p) \in eval\text{-}exp$ such that $\{e_p\}_{k_p}$ has a "type-1", "coinductively-invisible" subexp $\{e'\}_{k_j}$, where E_j is the computational image of $\{e'\}_{k_j}$ in E_p. Using this observation, one can construct another adversary, $\mathcal{A}'$, who attacks the scheme in the sense of SEG and wins with a non-negligible probability. The reduction is simple: $\mathcal{A}'$ just needs to guess the *encryption* query number (which makes $(\{e_p\}_{k_p}, E_p)$ as described above), the "index" of the subexp of $\{e_p\}_{k_p}$ that corresponds to $\{e'\}_{k_j}$, and the index of the subexp of $\{e_p\}_{k_p}$ that corresponds to $\{e\}_{k_i}$ (here k_i^{-1} is going to be the key that $\mathcal{A}'$ is going to "challenge" in his **interaction** phase), and the *decryption* query number that corresponds to $decrypt(E_j, w)$. Therefore, the encryption scheme is not l-SEG secure, and, hence by Lemma 15, is not IND-CCA2 secure, reaching a contradiction. □

C.11 Lemmas 2 and 3 with Symmetric Encryption Schemes

The proofs that we gave for Lemmas 2 and 3 easily extend by adding the symmetric encryption scheme (and hence disallowing k_i^{-1}'s, where k_i is a public key, to occur as plaintexts). The reason that we chose to work with these modified versions of the CI and WSTCI games was to do away with the encryption oracles (which encrypts a single plaintext under the oracle's key) which would be otherwise present if we were dealing with private-key encryption also. Nevertheless, all the presented proofs simply extend by observing that in IND-CCA2 definitions for symmetric-key encryption schemes there exists an additional encryption oracle which encrypts plaintexts under the oracle's key. (This can aslo be modeled using a LOR encryption oracle, with the two messages given to the oracle being the same.) Now we briefly explain how the extension is made: in Algorithm 1 (with omitting some details), $k_{u_s}, k_{u_{s+1}}, \ldots, k_{u_l}$ will now be selected from $\{k_1, \ldots, k_n, k_1^{sym}, \ldots, k_n^{sym}\}$. (Note, although symmetric decryption keys do not occur as plaintexts, meaning that the guessed parameters $u_{s+1}, \ldots, u_l$ should definitely be indices of symmetric key symbols (or otherwise the guessing is obviously wrong), we let all u_i's be selected from the space of indices for both symmetric and asymmetric keys. Although this may result in trivially wrong guessings, excluding it does not significantly improve the reduction bounds.)

Now in the SIMULATION phase if k_{u_s} is a public key, no substantial change is needed. Otherwise, if k_{u_s} is a symmetric key, $\mathcal{A}_{cca}$ will again replace all the occurrences of k_{u_s} as a plaintext with a fake value, and, now, makes "single" encryptions under k_{u_s} with the aid of his encryption oracle. The rest of the SIMULATION phase does not need substantial changes.

D Proof of Theorem 3

Suppose $\mathcal{E}_{asy}$ provides IND-CCA2 security, and $\mathcal{E}_{sym}$ provides both IND-CCA2 and INT-CTXT security. We want to prove that $\mathcal{E}_p = (\mathcal{E}_{asy}, \mathcal{E}_{sym})$ provides l-ACI-security. First by Theorem 2 we have that $\mathcal{E}_p$ provides l-CI-security; therefore, we just need to show that $\mathcal{E}_p$ provides security under the ACI game. For the sake of contradiction, suppose there exists $\mathcal{A}_{aci}$ who attacks $\mathcal{E}_p$ under the ACI game and wins with a non-negligible probability; namely, with a non-negligible probability for his outputs (c, i) it holds that (1) $k_i^{sym} \notin closure_c(eval\text{-}exp^1)$, (2) $Dec_{sym}(c, ck_i) \neq \perp$, and (3) (c, i) is *not* coinductively visible in *eval-exp*. (See the corresponding Definition in Paragraph C.0.1.) We first give a couple of claims below.

Claim A: We claim that, under the security assumptions made above about $\mathcal{E}_{asy}$ and $\mathcal{E}_{sym}$, for every adversary $\mathcal{A}$ in the ACI game, the probability that the following event occurs is negligible: $\mathcal{A}$ during his interaction phase makes a query $decrypt(c', s)$, for $s \in \{k_1^{-1}, \ldots, k_n^{-1}, k_1^{sym}, \ldots, k_n^{sym}\}$, such that there exists $(\{e_h\}_{k_h}, c_h) \in eval\text{-}exp$ which has a type-1 subexp $\{e_t\}_{s_1}$, for $s_1 \in$

$\{k_1, \ldots, k_n, k_1^{sym}, \ldots, k_n^{sym}\}$, where $s_1^{-1} \notin closure_c(eval\text{-}exp^1)$, $\{e_t\}_{s_1} \notin clsoure_c(eval\text{-}exp^1)$ and c' is the underlying value of $\{e_t\}_{s_1}$ in c_h. This is because, otherwise, one can modify $\mathcal{A}$ slightly to construct a new adversary who wins with a non-negligible probability under the SEG game, and by Lemma 15, contradicting the fact that $\mathcal{E}_{sym}$ and $\mathcal{E}_{sym}$ provide IND-CCA2 security.

Claim B: We claim that with the assumptions made about $\mathcal{A}_{aci}$ and the assumptions made about $\mathcal{E}_{asy}$ and $\mathcal{E}_{sym}$ above, the probability that the following occurs is non-negligible: for $\mathcal{A}_{aci}$'s output (c, i) all conditions (1), (2) and (3) above hold and moreover it holds that there *does* not exist $(\{e_h\}_{k_h}, c_h) \in eval\text{-}exp$ such that $\{e_h\}_{k_h}$ has a type-1 subexp $\{e_t\}_{s_1}$, for $s_1 \in \{k_1, \ldots, k_n, k_1^{sym}, \ldots, k_n^{sym}\}$, where $s_1^{-1} \notin clsoure_c(eval\text{-}exp^1)$, $\{e_t\}_{s_1} \notin clsoure_c(eval\text{-}exp^1)$ and c' is the underlying value of $\{e_t\}_{s_1}$ in c_h. This claim follows immediately from the previous claim and the assumptions made about $\mathcal{A}_{aci}$.

Now we introduce a new adversary $\mathcal{A}_{wstci1}$ who, using $\mathcal{A}_{aci}$ as a blackbox, attacks $\mathcal{E}_p$ under the WSTCI$_1$ game as follows. $\mathcal{A}_{wstci1}$ first in his **interaction** phase guesses k_i^{sym}, the symmetric key that $\mathcal{A}_{aci}$ is going to challenge in his **guessing** phase, specifies k_i^{sym} as his WSTCI$_1$-*challenge* key, and then replies to $\mathcal{A}_{aci}$'s queries using his own oracle access. (We will explain a bit later why $\mathcal{A}_{wstci1}$ is able to answer to all $\mathcal{A}_{aci}$'s queries.) Once $\mathcal{A}_{aci}$ moves to his **guessing** phase and outputs his ACI-challenge (c, i), first if the key that $\mathcal{A}_{wstci1}$ guessed in his **interaction** does not match the key that $\mathcal{A}_{aci}$ challenges, then $\mathcal{A}_{wstci1}$ returns a bit uniformly at random. Otherwise, $\mathcal{A}_{wstci1}$ calls his *decryption* oracle on (c, k_i^{sym}) and determines his guess for the WSTCI$_1$-challenge-bit as follows: if the *decryption* oracle does not output $\perp$, then $\mathcal{A}_{wstci1}$ outputs 1, otherwise he outputs 0.

Let's first show that why $\mathcal{A}_{wstci1}$ is able to answer to all $\mathcal{A}_{aci}$'s queries. First it is easy all *encryption* and *corrupt* queries of $\mathcal{A}_{aci}$ can be perfectly handled by $\mathcal{A}_{wstci1}$'s oracles. As for *decryption* queries, we claim that for both $b_{wstci1} = 0$ and $b_{wstci1} = 1$ (where b_{wstci1} is the WSTCI$_1$-challenge bit), the probability that $\mathcal{A}_{aci}$ makes a *decryption* query that cannot be answered by $\mathcal{A}_{wstci1}$'s decryption oracle (because of the additional decryption condition presented there) is negligible (or otherwise one of the assumptions made above is violated). If $b_{wstci1} = 0$, this claim immediately follows from Claim (A). If $b_{wstci1} = 1$, things are less clear (because here the occurrences of k_i^{sym} as a plaintext are "faked" and it is not clear how Claim (A) applies to this situation). However, under $b_{wstci1} = 1$ the view of $\mathcal{A}_{aci}$ will be identically distributed as his view under an ACI game in which he replaces all plaintext occurrences k_i^{sym}'s in his *encryption* queries with a new private key symbol $k_i'^{sym}$. Now if under $b_{wstci1} = 1$ the above claim does not hold, one can properly modifies $\mathcal{A}_{aci}$ in such a way that the new $\mathcal{A}_{aci}$ will violate the statement of Claim (A). We omit the details.

Now having shown that the simulation can be done, now from Claim (B) it immediately follows that the probability that $\mathcal{A}_{wstci1}$ outputs 1 when $b_{wstci1} = 0$ is non-negligible. As a result, from Equation (48) we obtain that with a non-negligible probability (c, ck_i) (i.e. the output of $\mathcal{A}_{aci}$) is also a valid *decryption* query under world WSTCI1$_{b=1}$. Now since under WSTCI1$_{b=1}$ all occurrences of k_i^{sym} (i.e. the key that $\mathcal{A}_{wstci1}$ specifies in the beginning of his **interaction** phase) are replaced with fake values, "(c, ck_i) being a valid *decryption* query under WSTCI1$_{b=1}$ implies that for no $(\{e_h\}_{k_h}, c_h) \in eval\text{-}exp$, it holds that $\{e_h\}_{k_h}$ has a subexp $\{e'\}_{k_i^{sym}}$ with c being the underlying value of $\{e'\}_{k_i^{sym}}$ in c_h. (This is because $\{e'\}_{k_i^{sym}}$ is always type-1 under world WSTCI1$_{b=1}$.) Now this enables efficiently performing an INT-CTXT attack. We provide the details below.

We define a new game, the *ACI1* game, as follows: the ACI1 game is similar to the ACI game with the only difference that in the beginning of the **interaction** phase the adversary has to specify k_i^{sym}, the key that he is going to challenge in the **guessing** phase, and this key should remain coinductively irrecoverable during the game. Moreover, during the **interaction** phase all occurrences of k_i^{sym} (as a plaintext) are replaced with a fake value. (i.e. The value used for k_i^{sym} when it appears a plaintext is independent of the value used when it appears as an encryption key.) Finally a playing adversary under the ACI1 game wins if for his output, (c, i), it holds that there exists no $(\{e_j\}_{k_j}, c_j) \in \mathit{eval\text{-}exp}$ such that $\{e_j\}_{k_j}$ has a subexp $\{e'\}_{k_i^{sym}}$ with c being the underlying value of $\{e'\}_{k_i^{sym}}$ in c_j. Now from what proved thus far, we can easily see that $\mathcal{A}_{aci}$ wins under the ACI1 game (i.e. breaks ACI1 security). The following claim concludes the proof of this theorem.

Claim C: If $\mathcal{E}_{sym}$ provides INT-CTXT security, then $(\mathcal{E}_{sym}, \mathcal{E}_{asy})$ provides ACI1-security.

It is fairly easy to see why the above claim is true; the central idea is that all occurrences of k_i^{sym} (the key that is specified in the **interaction** phase of the ACI1 game and for which a fresh ciphertext is produced in the final phase) as a plaintext are given a fake value; this enables making an INT-CTXT attack without having to know the underlying value of the key which parameterizes the CTXT-encryption oracle. To formalize the idea, we briefly show how to reduce the ability of $\mathcal{B}_{aci1}$, who wins under the ACI1-game, to a new adversary, $\mathcal{B}_{ctx}$, who breaks the INT-CTXT security of $\mathcal{E}_{sym}$. Since $\mathcal{B}_{ctxt}$ attacks under the INT-CTXT game, he is provided with an *encryption* oracle, which encrypts messages under an unknown random key ck, and his goal is to produce a fresh ciphertext (not generated by the oracle) which decrypts to a meaningful plaintext under ck. Now the idea of the reduction is clear: $\mathcal{B}_{ctx}$ simulates $\mathcal{B}_{aci1}$, generates random values for all basic symbols (as well as ck_i', a fake value for k_i^{sym}), and replies to $\mathcal{B}_{aci1}$'s queries using his generated values, with the only exception that encryptions under k_i^{sym} are performed by its *encryption* oracle; namely, to compute the bitstring value of an expression like $\{e\}_{k_i^{sym}}$, he first recursively computes E, the bitstring value of e, and then calls his oracle on E to obtain the bitstring value of the expression. (We stress here again that all "plaintext" occurrences of k_i^{sym} are replaced with ck_i'.) Note that *corruptions* and *decryptions* of $\mathcal{A}_{aci1}$ can also be very easily replied to by $\mathcal{B}_{ctx}$. Now when $\mathcal{B}_{aci1}$ outputs his guess (c, i), $\mathcal{B}_{ctxt}$ also outputs c. This completes the proof.

□

E Revealing Random Coins

Our computational trace-based model for protocols analysis (Section 3) assumes that if a user u_i is corrupted by the adversary, the adversary will *only* obtain the long-lived secret key of u_i and symmetric keys/nonces that u_i has generated during the sessions she has engaged in, but not her past random coins. In this section we present some results which allow us to relax this assumption by allowing the adversary to also obtain the random coins of the corrupted users. For this, for every user u_i, we partition the random coins used by u_i to $\mathcal{R}^1_{u_i}$, those used to sample the pair of long-lived keys, symmetric keys and nonces, and $\mathcal{R}^2_{u_i}$, those used to perform encryptions throughout the execution.

E.1 Random Coins for Non-Encryption Operations

We can easily show that our soundness results extend if the computational model is strengthened by allowing the adversary to additionally receive $\mathcal{R}^1_{u_i}$, for every u_i who he corrupts. We call this new computational model $\mathtt{C}_{\mathit{weak\text{-}coins}}$, and call the respective soundness notion (defined along the lines of Definition 1) the *soundness+weak-coins-revelation* notion.

Theorem 5. *Suppose $\mathcal{E}_p = (\mathcal{E}_{asy}, \mathcal{E}_{sym})$, where $\mathcal{E}_{asy}$ is IND-CCA2 secure, and $\mathcal{E}_{sym}$ is both IND-CCA2 and INT-CTXT secure. Then for every PPT adversary $\mathcal{A}$ under the $\mathtt{C}_{\mathit{weak\text{-}coins}}$ model, bounded protocol Π (See Section 3) it holds that*

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

To prove the above theorem, first we can easily show that if the CI, ACI, and CNM games are strengthened such that under a *corrupt*(s) or *reveal*(s) query, the random coins used to sample $\tau(s)$ are also given out, then Theorems 2 and 3 extend for these stronger definitions. The extension is easy; the only thing that needs to be added to Algorithm 1 is that under a *corrupt*(s) or a *reveal*(s) query, $\mathcal{A}_{cca}$ returns the random coins he used in the SETUP phase to sample the corresponding random bitstring. We omit further details. Finally the same poof given for Theorem 1 applies (with very small modifications) to conclude that extended-ACI-security implies extended-CNM-security and this latter implies extended soundness. We omit the details.

E.2 All Random Coins

We show that for every *single-encryption* protocol our soundness extends to still a stronger setting wherein *all* the random coins of a corrupted user are given to the adversary. We call $\Pi = (M_1^I, M_i^R, \ldots, M_r^X)$, for $X \in \{I, R\}$, a *single-encryption protocol* if for all its components M_j^Y it holds that M_j^Y has only *one* layer of encryption. (i.e. If M_j^Y has a subexp $\{M'\}_K$, then M' does not have any encrypted subexp.) We denote this stronger computational model by $\mathtt{C}_{strong\text{-}coins}$ (in which if u_i is corrupted, then $\mathcal{R}^2_{u_i}$ is also given out), and also refer to the respective soundness notion by *soundness+strong-coins-revelation.*

Theorem 6. *Suppose $\mathcal{E}_p = (\mathcal{E}_{asy}, \mathcal{E}_{sym})$, where $\mathcal{E}_{asy}$ is IND-CCA2 secure, and $\mathcal{E}_{sym}$ is both IND-CCA2 and INT-CTXT secure. Then for every PPT adversary $\mathcal{A}$ under the $\mathtt{C}_{strong\text{-}coins}$ model, single bounded protocol Π, it holds that*

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

To show the above theorem, we first define variants of the CI, ACI and CNM games, which are *single-encryption*-based variants of them strengthened by adding a new type of query. For every CI, ACI and CNM game, the new variant of the game requires that, for every query *encrypt*(e, k) of the adversary, e must be an encryption-free expression. (i.e. e does not have a subexp of the form $\{e'\}_{k'}$.) Moreover the new variant allows a new type of query as follows: for $(\{e\}_k, c) \in \textit{eval-exp}$, if R is the random string used by the challenger to obtain c from its corresponding plaintext, the adversary can make a query *reveal-coins*(c) to obtain R, provided the following condition holds: either $k = k_i^{sym}$ (i.e. k is a symmetric key) and $k_i^{sym} \in closure_c(\textit{eval-exp}, \textit{corrupt-keys})$, or $k = k_j^{asy}$ (i.e. k is an asymmetric key) and $e \in closure_c(\textit{eval-exp}, \textit{corrupt-keys})$. For $X \in \{CI, ACI, CNM\}$, we call the respective extended game the *single-encryption, coins-revealing X* (shortly the *SECRX*) game.

We first briefly explain how the proof of the second part of Theorem 1 extends to conclude $SECRCNM \Rightarrow soundness$. In the proof the only part that is added is that when the simulated adversary $\mathcal{A}_{cs}$ corrupts u_i, and assuming that u_i has already sent a ciphertext c (encrypted under random coins R) to, say, (corrupted or uncorrupted) u_j, we need to show how the simulating adversary $\mathcal{A}_{cnm}$ obtains R and gives it to $\mathcal{A}_{cs}$. We describe the idea through a running example: consider two cases: (a) c is the computational image of $\{(x_{i,sn,h}, k^{sym}_{i,sn,h})\}_{k_j}$ (i.e. c is obtained by encrypting the concatenation of the local nonce and symmetric key under u_j's public key); and, (b) c is the computational image of $\{(x_{i,sn,h}, k^{sym}_{i,sn,h})\}_{k^{sym}_{i,sn,w}}$ (i.e. c is obtained by encrypting the concatenation of the local nonce and symmetric key under another local symmetric key. See the proof of Theorem 1 for the meaning of these symbols). Now if (a) is the case, note that after $\mathcal{A}_{cs}$ corrupts u_i, the simulating adversary, $\mathcal{A}_{cnm}$, (to simulate $\mathcal{A}_{cs}$) corrupts $x_{i,sn,h}$ and $k^{sym}_{i,sn,h}$) (among perhaps other corruption queries) to give their computational images to $\mathcal{A}_{cs}$. Now at this point $\mathcal{A}_{cnm}$ under the CNM game is legitimately allowed to obtain random coins used to encrypt c under both cases (a) and (b) above.

Now the proof of the other part of Theorem 1 follows exactly as before. We now briefly explain how the proof of Theorem 2 extends for the SECRCI game. (The extension for Theorem 3 also follows similarly.) For this, suppose for $(\{e\}_k, c) \in \textit{eval-exp}$ the adversary makes $\textit{reveal-coins}(c)$ query. Now since $e \in \textit{closure}_c(\textit{eval-exp}^1, \textit{corrupt-keys})$, in Algorithm 1, the ciphertext c is constructed *entirely* by $\mathcal{A}_{cca}$ adversary himself, without calling his left-or-right encryption query. This is because, in Algorithm 1 for the guessed parameters $u_s, u_{s+1}, \ldots, u_l$, $on_{s+1}, \ldots, on_l$, the (only) subexpression whose computational value is obtained through the left-or-right encryption oracle (and hence its underlying random coins cannot be determined by $\mathcal{A}_{cca}$) is the subexpression that contains $k_{u_{s+1}, on_{s+1}}$ (plus satisfying some more properties). Moreover, the guessing procedure requires that $k^{-1}_{u_{s+1}}$ (and other $k^{-1}_{u_i}$'s) remain coinductively irrecoverable during the game.