Adaptive Security of Symbolic Encryption*

Daniele Micciancio[1] and Saurabh Panjwani[1]

University of California, San Diego
La Jolla, CA 92093, USA,
{daniele, panjwani}@cs.ucsd.edu

Abstract. We prove a computational soundness theorem for the symbolic analysis of cryptographic protocols which extends an analogous theorem of Abadi and Rogaway (J. of Cryptology 15(2):103–127, 2002) to a scenario where the adversary gets to see the encryption of a sequence of adaptively chosen symbolic expressions. The extension of the theorem of Abadi and Rogaway to such an adaptive scenario is nontrivial, and raises issues related to the classic problem of selective decommitment, which do not appear in the original formulation of the theorem.

Although the theorem of Abadi and Rogaway applies only to passive adversaries, our extension to adaptive attacks makes it substantially stronger, and powerful enough to analyze the security of cryptographic protocols of practical interest. We exemplify the use of our soundness theorem in the analysis of group key distribution protocols like those that arise in multicast and broadcast applications. Specifically, we provide cryptographic definitions of security for multicast key distribution protocols both in the symbolic as well as the computational framework and use our theorem to prove soundness of the symbolic definition.

Keywords: Symbolic encryption, adaptive adversaries, soundness theorem, formal methods for security protocols.

1 Introduction

Traditionally, security protocols have been designed and analyzed using two competing approaches: the symbolic one, and the computational one. The symbolic approach is characterized by an abstract (adversarial) execution model, where cryptographic operations and objects are treated as an abstract data type, not only when used by honest protocol participants, but also when used by malicious players attacking the system. This allows for simple proofs of security, typically based on syntactic properties of the messages exchanged during the execution of the protocol. The computational approach is based on a more detailed execution

* Research supported in part by NSF grants 0313241 and 0430595. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

model that accounts for a much wider class of adversaries attacking the system, namely arbitrary probabilistic polynomial time bounded adversaries that do not necessarily respect the cryptographic abstractions used by the protocol. The stronger security guarantees offered by the computational approach come at a substantial price in complexity: proofs of security in this framework typically involve subtle probabilistic arguments, complicated running time analysis, and the ubiquitous use of computational assumptions, like the intractability of factoring large integers.

Recently, there has been a lot of interest in combining the two approaches, with the generic goal of coming up with abstract models that allow computationally sound symbolic security analysis, i.e., a method to translate symbolic security proofs into precise computational statements about the security of concrete protocol executions in the computational framework.

Our work follows a line of research initiated by Abadi and Rogaway in [2], where a simple language of encrypted expressions is defined, together with a computationally sound symbolic semantics. Technically, [2] introduces a mapping from expressions to *patterns* that characterize the information leaked by the expressions when evaluated using a computationally secure encryption scheme. The structure of the soundness result of [2] is rather simple: an adversary attacking the system produces a symbolic expression and subsequently receives the computational evaluation of either the expression or its pattern. The soundness theorem of [2] states that if the expression satisfy certain syntactic restrictions (namely, it does not contain encryption cycles) then the adversary cannot efficiently determine if it received the evaluation of the expression or its pattern.

The result of [2] is an interesting first step demonstrating the feasibility of computationally sound symbolic security analysis. The class of encrypted expressions considered in [2] is fairly general, and allows to describe the messages transmitted in many practical protocols. However, the result itself is too simple for direct application to security analysis of protocols. Intuitively, the scenario considered in [2] involves a party sending a single message to another party over an authenticated channel, and a passive adversary monitoring the channel. In practice, security protocols involve the exchange of several messages, among two or more parties, and in different directions. Moreover, messages may depend on the computational interpretation of previously chosen messages and/or external inputs that are not known at the beginning of the protocol.

Our Results. In this paper, we consider an extension of [2] wherein the adversary produces a *sequence* of expressions, which are subsequently evaluated according to a common key assignment. If all the expressions in the sequence were specified at the same time, then this wouldn't be any different from the original soundness theorem of [2], as all the expressions in the sequence could be concatenated into a single expression. What makes our extension interesting and nontrivial is the fact that the sequence is adversarially specified in an *adaptive* way, so that each expression in the sequence may depend on the computational evaluation of the previous ones. The ability to specify the expressions adaptively allows the adversary to generate probability distributions that do not correspond to any fixed sequence of expressions, and immediately raises issues related to the classic problem of selective decommitment [9] and adaptive corruption [5]. (See Section 3 for details.) In order to avoid these problems we introduce some syntactical restrictions on the expressions, beside the acyclicity condition already considered by [2]. Informally, the syntactic restrictions postulate that each key is used in two stages: a key distribution stage during which the key can be used as a message, and a key deployment stage during which the key is used to encrypt other messages. Using these syntactic restrictions, we are able to bypass the selective decommitment problems, and prove a soundness result for symbolic encryption with adaptively chosen expressions.

Our soundness result allows to analyze a generic class of protocols that involve communication between multiple parties over an authenticated network.[1] The execution model for these protocols involves an adversary that observes all messages sent over the network and can adaptively change the execution flow of the protocol (e.g., through interaction with the execution environment), but is not allowed to modify or delete any of the messages sent or received by the legitimate parties.

Our soundness result for adaptively chosen encrypted expressions substantially increases the expressive power of the soundness theorem of [2], making it powerful enough to analyze practical cryptographic protocols. We exemplify the use of our soundness theorem in the analysis of group key distribution protocols, like those used in multicast applications [23, 6, 22]. In the multicast key distribution problem, a data source wants to broadcast information to a dynamically changing group of parties, in such a way that at any given point in time only current group members can decipher the transmitted messages. The problem is typically solved by establishing a secret key, known to all and only the "current" group members. Each time a user joins or leaves the group, a group controller broadcasts some messages which are used by the new set of members to update the group key.

We give formal definitions of security for multicast key distribution protocols, in both the computational framework and the symbolic framework, and show that if a protocol is secure in the symbolic setting, then the (implementation of the) protocol is also secure in the computational setting (provided the messages used in the protocol conform with the syntactic restrictions of our soundness theorem). Most multicast key distribution protocols we are aware of (e.g., [23, 6]) satisfy these restrictions and can thus be proven secure (against powerful computational adversaries) using the symbolic definition. To the best of our knowledge, formal definitions for security of these protocols have not been discussed in the literature prior to this work nor has there been any attempt to relate security analysis of these protocols in the symbolic framework (as is done implicitly in many papers) to computational security guarantees on their implementations. Our soundness theorem is an important building block in that direction.

[1] Authenticated channels are a widely used model in cryptographic protocol design, and can be implemented on top of non-authenticated networks using standard techniques, like message authentication codes and digital signatures.

Related work. Several improvements and refinements have followed the original work of Abadi and Rogaway [2], but they are mostly orthogonal to our results. In [19], Micciancio and Warinschi prove a converse of the soundness theorem, showing that if a sufficiently strong encryption scheme is used, then a computationally bounded adversary can recover all and only the information captured by the symbolic patterns. The result is further refined by Gligor and Horvitz [10], who give an exact characterization of the computational requirements on the encryption scheme under which the completeness theorem of [19] holds true.

An extension of the soundness result of [2] to multiple message/player settings, is presented by Abadi and Jürjens in [1]. This result considers an arbitrary set of parties exchanging several messages over an authenticated network over time. However, the protocol specification language used in [1] only allows to describe protocols in which all messages transmitted during the protocol execution can be uniquely determined before the execution of the protocol begins. In other words, the result of [1] does not account for scenarios where the messages are chosen adaptively, and from a technical point of view, it is much closer to [2] than to our work.

The papers [20, 14] present two different extensions of the framework of [2] that allow for active attacks, i.e., adversaries that have a total control of the communication network and may drop, alter, or inject messages in the network. Both works are based on encryption schemes satisfying the stronger notion of security against chosen ciphertext attack (CCA [21, 8]). Our results hold for any encryption scheme secure against chosen plaintext attack (CPA [11]). Moreover, the results of [20, 14] have a qualitatively different and somehow more complex formulation than the results presented in this paper. In [20] Micciancio and Warinschi consider trace properties[2] of both symbolic and computational executions of cryptographic protocols and relate the two models by proving that (if a CCA secure encryption scheme is used) any protocol that satisfies a trace property in all its symbolic executions, also satisfies the corresponding trace property in computational executions with overwhelming probability. We remark that the results in [20] are incomparable with those presented in this paper as trace properties do not allow to readily model indistinguishability properties as considered in this paper. Laud's result [14] is quite different from the other soundness results considered so far. Rather than considering a computational and a symbolic execution models, and relating the two, [14] only considers a computational execution model and a set of symbolic program transformations, and proves that the symbolic transformations are computationally sound in the sense that they preserve computational secrecy properties when both the original and transformed program are executed in the computational model.

Other approaches to the problem of computationally sound symbolic analysis are exemplified by [4, 3, 15, 13]. In [4, 3] Backes, Pfitzmann and Waidner present

[2] These are a class of properties, extensively used in the formal verification of distributed protocols, which can be represented as the allowable sequences of internal states (or external actions) performed by the honest protocol participants.

an implementation of Dolev-Yao style terms achieving a simulation based security definition within a general computational framework. It is important to notice that while [4, 3] allow to formulate and prove computational security properties of protocols built using their library, their results do not apply to protocols that make direct use of encryption schemes satisfying standard security notions, like CPA or CCA indistinguishability. We remark that [3] relies on syntactic restrictions on the use of symmetric encryption similar to those used in this paper. Our work shows that the difficulties encountered in [3] in trying to lift these restrictions are not specific to their universally composable security framework, but arise already in much simpler scenarios as those considered in this paper. In [15] Lincoln, Mitchell, Mitchell and Scedrov present a probabilistic process calculus that can be used to analyze computational security properties of cryptographic protocols. All these works are both substantially more complex and powerful than (though, technically incomparable to) the line of work initiated by [2], as they allow to describe arbitrary probabilistic polynomial time computations. The work of Impagliazzo and Kapron [13] approaches the problem of computationally sound symbolic analysis from still another side. They present an axiomatic system with limited forms of recursion that can be used to carry out proofs of the type used in the analysis of basic cryptographic constructions without the explicit use of nested quantifiers and asymptotic notation. An interesting question is whether the soundness theorem proved in this paper can be proved within the logic of [13].

Organization. After giving some basic definitions in Section 2, we present our soundness theorem in Section 3. The proof of the soundness theorem is given in Section 5, after describing an application to multicast key distribution in Section 4. Section 6 concludes with a discussion of future work and open problems.

2 Preliminaries

Let **Keys** and **Const** be two sets of symbols called *keys* and *constants* respectively. We can assume that both sets are finite, and have size bounded by a polynomial in the security parameter. For a given value of the security parameter, let **Keys** $:= \{K_1, \cdots, K_n\}$ be the set of keys. We define a language, **Exp**, of expressions, called *basic expressions*, that is generated using the following syntactic rules:

$$M \to (M, M) | \{M\}_{K_1} | \{M\}_{K_2} | \cdots | \{M\}_{K_n}$$
$$M \to \text{Each symbol in } \mathbf{Keys} \cup \mathbf{Const}$$

The rule $M \to (M, M)$ symbolizes a pairing operation while $M \to \{M\}_{K_i}$ for any K_i symbolizes encryption under K_i. Sequences of expressions can be converted into a single expression using the paring operation in the obvious way, e.g., the sequence $(M[1], \ldots, M[q])$ can be represented by the expression $(M[1], (M[2], \ldots, (M[q-1], M[q]) \ldots))$. For any sequence $(M[1], \ldots, M[q])$ and indexes $1 \leq i \leq j \leq q$, we use notation $M[i..j]$ to denote the subsequence $(M[i], M[i+1], \ldots, M[j])$.

For any $M \in \mathbf{Exp}$, a key that occurs as a plaintext in some sub-expression of M is referred to as a *message key* while one that is used to encrypt some sub-expression is called an *encryption key.* We denote the set of all message (resp. encryption) keys of M by $\mathbf{MsgKeys}(M)$ (resp. $\mathbf{EncKeys}(\mathbf{M})$). We say that a key K_i *encrypts* K_j (or K_j is *encrypted under* K_i) in M, denoted $K_i \rightarrow_M K_j$, if M contains a sub-expression $\{M'\}_{K_i}$ such that $K_j \in \mathbf{MsgKeys}(M')$. As in [2], we call a key *recoverable* in M if it is in $\mathbf{MsgKeys}(M)$ and occurs in it either unencrypted or encrypted under keys that are, in turn, recoverable in it. The set of all recoverable keys of M is denoted $\mathbf{RecKeys}(M)$. The set of all unrecoverable encryption keys in M, i.e. the set $\mathbf{EncKeys}(M) \backslash \mathbf{RecKeys}(M)$, is denoted $\mathbf{UEncKeys}(M)$. As an example, if $M = ((K_1, \{K_2\}_{K_1}), \{K_4\}_{K_3})$, then $\mathbf{EncKeys}(M) = \{K_1, K_3\}; \mathbf{MsgKeys}(M) = \{K_1, K_2, K_4\}; \mathbf{RecKeys}(M) = \{K_1, K_2\}$ and $\mathbf{UEncKeys}(M) = \{K_3\}$.

Formal Semantics. The information that can be extracted from an expression using known keys and the decryption algorithm can be represented by a syntactic object, called *pattern.* We use a definition of patterns recently proposed in [16] which characterizes encryption schemes satisfying the standard notion of semantic security under chosen plaintext attack [11]. (These patterns are slightly different from those used in [2, 19, 10], which correspond to encryption schemes satisfying a variant of semantic security. A definition similar to ours was also used in [12].) We define the *structure* of an expression $M \in \mathbf{Exp}$, denoted $\mathtt{struct}(M)$, as the expression obtained by substituting all message keys in M by a symbol K', all encryption keys in it by K and all constants by a symbol $\mathbf{c}$, where $K, K' \notin \mathbf{Keys}$ and $\mathbf{c} \notin \mathbf{Const}$ are all fresh symbols. For example, $\mathtt{struct}(\{\mathbf{0}\}_{K_2}, \{(K_2, \{K_1\}_{K_2})\}_{K_3}) = (\{\mathbf{c}\}_K, \{(K', \{K'\}_K)\}_K)$.

Definition 1. *For any $M \in \mathbf{Exp}$, the pattern of M given a set of keys T, denoted $\mathtt{pat}(M, T)$, is an expression defined recursively as follows:*

- *If $M \in \mathbf{Keys} \cup \mathbf{Const}$, then $\mathtt{pat}(M, T) = M$.*
- *If $M = (M_1, M_2)$, then $\mathtt{pat}(M, T) = (\mathtt{pat}(M_1, T), \mathtt{pat}(M_2, T))$.*
- *If $M = \{M'\}_{K_i}$ and $K_i \in T$, then $\mathtt{pat}(M, T) = \{\mathtt{pat}(M', T)\}_{K_i}$.*
- *If $M = \{M'\}_{K_i}$ and $K_i \notin T$, then $\mathtt{pat}(M, T) = \{\mathtt{struct}(M')\}_{K_i}$.*

The pattern of M, denoted $\mathtt{pattern}(M)$, is defined as $\mathtt{pat}(M, \mathbf{RecKeys}(M))$.

This definition of patterns captures the intuitive idea that given a bitstring interpretation of any expression encrypted under say K_i, an adversary can learn everything about the expression if he knows K_i, but can learn nothing more than its structure if he does not know K_i.

Just as in [2], we say that two expressions $M_1, M_2 \in \mathbf{Exp}$ are *equivalent*, denoted $M_1 \simeq M_2$, if their patterns, when viewed as strings of symbols, are identical up to renaming of their keys. That is, M_1 and M_2 are equivalent if there exists an injective map, ϕ, from the keys in M_1 to the keys in M_2 such that when every key K_i (other than a structure key) in $\mathtt{pattern}(M_1)$ is substituted with $\phi(K_i)$, the resulting expression is identical to $\texttt{pattern}(M_2)$. For example, if we consider the expressions

$$\begin{aligned}
M_1 &= ((K_1, \{\mathbf{0}\}_{K_2}), \{K_2, \{K_1\}_{K_2}\}_{K_3}) \\
M_2 &= ((K_1, \{\mathbf{1}\}_{K_6}), \{K_5, \{K_6\}_{K_1}\}_{K_2}) \\
M_3 &= ((K_1, \{K_5\}_{K_2}), \{K_5, \{K_6\}_{K_1}\}_{K_2})
\end{aligned}$$

we have $M_1 \simeq M_2$ but $M_1 \not\simeq M_3$.

Computational Semantics. We define computational semantics of all expressions, including all basic expressions and their respective patterns, using a single procedure. (We denote the set of all such expressions by $\mathbf{Exp'}$ below). For any symmetric encryption scheme, $\Pi = \{\mathcal{K}, \mathcal{E}, \mathcal{D}\}$, let $\mathcal{K}^m(\eta)$ denote the random variable corresponding to a vector of m keys sampled independently using the key generation algorithm $\mathcal{K}$, giving it security parameter η as input. The procedure for defining computational semantics of expressions takes the security parameter η as input and works in two steps.

1. We generate a key vector τ from the distribution $\mathcal{K}^{n+2}(\eta)$ (where $n = |\mathbf{Keys}|$) and map all keys in $\mathbf{Keys} \cup \{K', K\}$ to elements in this vector. Specifically, for all $i \in \{1, \cdots, n\}$, $\tau[i]$ corresponds to K_i, $\tau[n+1]$ to K' and $\tau[n+2]$ to K.
2. In the second step, we look at expressions in $\mathbf{Exp'}$ and for each expression M we define the *bitstring interpretation of* M given τ as a random variable, $[\![M]\!]_{\Pi,\tau}$, in the following recursive manner:
   - If $M \in \mathbf{Const} \cup \{\mathbf{c}\}$, then $[\![M]\!]_{\Pi,\tau}$ is the bitstring representation of M, using some standard encoding.
   - If $M = K_i \in \mathbf{Keys}$, then $[\![M]\!]_{\Pi,\tau} \equiv \tau[i]$. If $M = K'$, then $[\![M]\!]_{\Pi,\tau} \equiv \tau[n+1]$.
   - If $M = (M_1, M_2)$ for some $M_1, M_2 \in \mathbf{Exp'}$, then $[\![M]\!]_{\Pi,\tau}$ is the random variable corresponding to $([\![M_1]\!]_{\Pi,\tau}, [\![M_2]\!]_{\Pi,\tau})$ (we use some standard efficiently computable and invertible encoding for the pairing operation).
   - If $M = \{M'\}_{K_i}$ for some $M' \in \mathbf{Exp'}$ and $K_i \in \mathbf{Keys}$, then $[\![M]\!]_{\Pi,\tau}$ is the random variable corresponding to $\mathcal{E}_{\tau[i]}([\![M']\!]_{\Pi,\tau})$. If $M = \{M'\}_K$ for some $M' \in \mathbf{Exp'}$, then $[\![M]\!]_{\Pi,\tau}$ is the random variable corresponding to $\mathcal{E}_{\tau[n+2]}([\![M']\!]_{\Pi,\tau})$.

Security of Encryption. We consider encryption schemes that are semantically secure against chosen plaintext attacks. For any symmetric encryption scheme $\Pi = \{\mathcal{K}, \mathcal{E}, \mathcal{D}\}$, a left-right oracle, $\mathcal{LR}_{\Pi,b}$ for Π is a program that first generates a key k using the key generating algorithm $\mathcal{K}$ and then for each query, (m_0, m_1) (m_0 and m_1 being bitstrings of equal length), given to it replies with the ciphertext $\mathcal{E}_k(m_b)$. Π is called `ind-cpa` secure if for any probabilistic polynomial-time distinguisher D the following quantity

$$\mathbf{Adv}_{\Pi}^{ind-cpa}(D, \eta) = |\Pr[D^{\mathcal{LR}_{\Pi,0}(\eta)}(\eta) = 1] - \Pr[D^{\mathcal{LR}_{\Pi,1}(\eta)}(\eta) = 1]|$$

is a negligible function of η, i.e., it is less than $1/\eta^c$ for any $c > 0$ and sufficiently large η.

3 Soundness

We consider a setting in which an adversary gets to see the computational evaluation of a sequence of (adaptively chosen) expressions. We want to model the fact that the adversary does not learn anything about the expressions, beside whatever information can be deduced from their patterns.

We formalize the problem using a cryptographic experiment as follows. Fix a symmetric encryption scheme $\Pi = \{\mathcal{K}, \mathcal{E}, \mathcal{D}\}$. Let A be any probabilistic polynomial-time machine that issues queries consisting of pairs[3] of basic expressions, the ith query of A being denoted by $(M_0[i], M_1[i])$. The experiment runs in one of two worlds, decided based on a bit b sampled uniformly at random in the beginning. After selecting b, the adversary is executed (given some security parameter as input) and the queries of the adversary are answered using an oracle, $\mathcal{O}_{\Pi,b}$, parameterized by the encryption scheme Π and the bit b. This oracle first selects a random key vector τ using the key generation algorithm of Π, $\mathcal{K}$, and for each query $(M_0[j], M_1[j])$, replies with a sample from the distribution $[\![M_b[j]]\!]_{\Pi,\tau}$, i.e. the bitstring interpretation of the bth expression in the query (with respect to Π and τ). A concise description of the oracle and the experiment appears below.

Oracle $\mathcal{O}_{\Pi,b}(\eta)$	$\mathbf{Expt}_{\Pi}^{adpt}(A)$
Let $\tau \xleftarrow{\$} \mathcal{K}^{n+2}(\eta)$ For the jth query received, $(M_0[j], M_1[j])$, reply with a sample from $[\![M_b[j]]\!]_{\Pi,\tau}$	Let $b \xleftarrow{\$} \{0,1\}$. Fix η. Run $A^{\mathcal{O}_{\Pi,b}(\eta)}(\eta)$

The goal of the adversary is to guess the value of b with probability better than random, and under the constraint that the two sequences of expressions queried to the oracle have the same pattern. More specifically, let q denote the number of queries made by A in any execution of the experiment, and let $M_b = M_b[1..q]$ be the sequence of expressions encrypted by the oracle $\mathcal{O}_{\Pi,b}(\eta)$. (Without loss of generality, q can be assumed to be a fixed polynomial in the security parameter, e.g., a polynomial upper bound on the running time of A.) We require that $M_0 \simeq M_1$. For technical reasons, in order to prove our soundness theorem we need to introduce some additional restrictions on the syntax of M_0 and M_1.

Definition 2. *A sequence of basic expressions, $M_b[1], M_b[2], \cdots, M_b[q]$, is called* legal *if it satisfies the following two properties:*

1. *The expressions M_0 and M_1 contain no encryption cycles.*[4]

[3] As standard in cryptography, we use distinguishability of pairs of messages to model leakage of partial information about those messages.

[4] An expression M is said to contain an encryption cycle if there exist keys $K_{i_1}, K_{i_2}, \cdots K_{i_m}$ such that $K_{i_1} \to_M K_{i_2} \to_M \cdots K_{i_{m-1}} \to_M K_{i_m} \to_M K_{i_1}$. Examples of such expressions are $\{K_1\}_{K_1}$ and $(\{K_1\}_{K_2}, \{(K_3, \{K_2\}_{K_3})\}_{K_1})$.

2. *No unrecoverable encryption key in $M_b[1..i]$ occurs as a message key in $M_b[j]$ for any $j > i$. That is, for all $i < j \leq q$*

$$\mathbf{UEncKeys}(M_b[1..i]) \cap \mathbf{MsgKeys}(M_b[j]) = \emptyset$$

For example, if $M_b[1] = \{K_i\}_{K_j}$ then it is illegal to have $M_b[2] = \{K_j\}_{K_l}$ or even $M_b[2] = K_j$.

The first requirement is standard in cryptography, and was already used in [2]. The second requirement is also very natural and it informally states that each key is used in two stages: a key distribution stage where the key is used as a message, and a subsequent deployment stage where the key is used to encrypt other messages and keys. This is the way keys are used in many cryptographic protocols, e.g., the key distribution protocols of [23, 6]. Intuitively, the reason we introduce this requirement is that if a key is first used to encrypt messages, and then revealed, the symbolic patterns of previously received messages may change and, consequently, our proof breaks down. At a more technical level, in the absence of the second requirement, an adversary can play the following game: first issue the expressions $\{M_1\}_{K_1}, \ldots, \{M_l\}_{K_l}$, and then, after getting the corresponding ciphertexts, ask for a randomly chosen set of keys $\{k_{i_1}, \ldots, k_{i_m}\}$ $(i_1, \cdots, i_m \in \{1, \cdots, l\})$ by issuing the expression $(K_{i_1}, \ldots, K_{i_m})$. The question of whether security of the ciphertexts (other than those that can be decrypted using the revealed keys) is ensured in this game is the classic problem of selective decryption for which no answer is known to date [9].

An adversary in $\mathbf{Expt}_{\Pi}^{adpt}$ is called *legal* if the queries issued by it are such that both $M_0[1], \cdots, M_0[q]$ and $M_1[1], \cdots, M_1[q]$ are legal sequences and $M_0 \simeq M_1$. The advantage of A in the experiment, denoted $\mathbf{Adv}_{\Pi}^{adpt}(A, \eta)$, is defined as the following quantity:

$$\mathbf{Adv}_{\Pi}^{adpt}(A, \eta) = |\mathbf{P}[A^{\mathcal{O}_{\Pi,0}(\eta)}(\eta) = 1] - \mathbf{P}[A^{\mathcal{O}_{\Pi,1}(\eta)}(\eta) = 1]|$$

where the probabilities are taken based on the randomness used by A and $\mathcal{O}_{\Pi,b}$.

We now state our soundness theorem:

Theorem 1. *If Π is an* `ind-cpa` *secure encryption scheme, then for any legal adversary A, $\mathbf{Adv}_{\Pi}^{adpt}(A, \eta)$ is a negligible function of η.*

We provide an overview of the proof of the soundness theorem in Section 5 but before doing that, we discuss an application to multicast key distribution.

4 Application to Secure Multicast

In this section, we present an example to illustrate how our soundness theorem can be used in the analysis of real cryptographic protocols. Our example is the *multicast key distribution problem* in which a large set of users communicates using a multicast (or broadcast) channel and at any time some of these users, called "group members", share a secret key which is known only to them and not to the rest of the users. The group members change dynamically and in order to maintain the secrecy property of the group key over time, a central authority, called the *group center*, broadcasts messages to enable the members to update the key whenever a new member joins or an old member leaves the group. In other words, the center "rekeys" the group whenever its composition changes. The goal is to ensure that at any point in time, the non-members are unable to compute the group key, even if several of them collude together and share all their information in an attempt to do so.

This problem arises in many practical scenarios and has been studied extensively by the cryptography as well as computer networks communities. (See for example [23, 6, 7, 22, 18].) However, there seems to have been very little attempt towards formulating a sound cryptographic model for the problem and proving security of any of the proposed solutions using standard cryptographic techniques. Although some works implicitly use a Dolev-Yao like framework in arguing for security of multicast key distribution protocols, it is not clear how such analysis relates to actual security of the protocols. Our soundness theorem provides a useful tool in relating proofs of security for these protocols in the formal framework to security proofs in the standard computational framework used in cryptography.

4.1 Security in the Computational Framework

We model a multicast key distribution protocol as a set of three programs $\Gamma = \{\mathcal{I}, \mathcal{C}, \mathcal{U}\}$ where $\mathcal{I}$ is an initialization program, $\mathcal{C}$ is the group center's program used to compute the rekey messages and $\mathcal{U}$ is the program run by the group members $U = \{u_1, \cdots, u_N\}$.

These programs work as follows. $\mathcal{I}$ takes the security parameter η as input and outputs the initial state of the center, $s_0^{\mathcal{C}}$, the initial states of all users $s_0^1, s_0^2, \cdots, s_0^N$ and the initial group membership $G_0 \subset U$. (Typically, $G_0 = \emptyset$.) The center's program, $\mathcal{C}$, takes as input η, the current state $s_t^{\mathcal{C}}$ and a command $\mathtt{com}_t$ and returns a message m_t (the rekey message at time t) and the updated state of the center $s_{t+1}^{\mathcal{C}}$. Each command $\mathtt{com}_t$ given to the center is either of the form $add(u_i)$ (which adds a new user to the group) or of the form $del(u_i)$ (which removes an existing member from the group). The users' program, $\mathcal{U}$, takes as input η, a user index i ($\leq N$), the previous state s_{t-1}^i of user u_i, and the current rekey message m_t, and outputs a string k_t^i and the updated state s_t^i of u_i. For correctness, we require that at every time instant $t \geq 0$, k_t^i be identical for every member u_i in the current group G_t. This value is called the group key at time t and is denoted k_t.

Security Definition: The security of multicast key distribution is modelled using an adversary that controls a subset of corrupted users and adaptively issues commands to change membership of the group. The adversary's goal is to gain information about the group key when none of the corrupted users are part of the group. Formally, for any protocol $\Gamma = \{\mathcal{I}, \mathcal{C}, \mathcal{U}\}$, we consider the following experiment, which we denote by $\mathbf{Expt}_{\Gamma}^{gkd}$. First, $\mathcal{I}$ is used to generate the initial states of the group center and all users, and the adversary $\mathcal{A}$ is given a set of corrupted users $B \subset U$, together with their initial states $\{s_0^i | u_i \in B\}$. The adversary then issues a sequence of t commands $\texttt{com}_1, \cdots, \texttt{com}_t$ and for each command $\texttt{com}_{t'}$, it is given the corresponding rekey message $m_{t'}$, computed according to program $\mathcal{C}$ and the group center's initial state produced by $\mathcal{I}$. (At any point in time, the users in B may or may not be in the group). Let $k_1, \cdots, k_t$ be the group keys at times $1, \cdots, t$ as computed by the honest group members. Let also $T \subseteq \{1, \cdots, t\}$ be the set of time instants when none of the corrupted users are in the group, and let $\bar{k}_T = \{k_i : i \in T\}$ be the corresponding keys. The security requirement is that the keys in $\bar{k}_T$ are pseudorandom. More precisely, let $\bar{k}'_T$ be a set of $|T|$ uniformly and independently chosen keys, and let b be a random bit. At the end of the experiment, the adversary is given either $\bar{k}_T$ or $\bar{k}'_T$ (depending on whether $b = 0$ or $b = 1$, respectively) and her goal is to correctly guess the value of b.[5]

Let $p_{\mathcal{A}}(B, b)$ be the probability that $\mathcal{A}$ outputs 1 in $\mathbf{Expt}_{\Gamma}^{gkd}$ when the corrupted set of users is B (here probabilities are taken based on the random choices of $\mathcal{A}, \mathcal{I}$ and $\mathcal{C}$). The advantage function of $\mathcal{A}$ in the experiment is defined as:

$$\mathbf{Adv}_{\Gamma}^{gkd}(\mathcal{A}, B, \eta) = |p_{\mathcal{A}}(B, 0) - p_{\mathcal{A}}(B, 1)|$$

Definition 3. *A multicast key distribution protocol Γ is secure if for all probabilistic polynomial-time adversaries $\mathcal{A}$ and all sets $B \subseteq U$, $\mathbf{Adv}_{\Gamma}^{gkd}(\mathcal{A}, B, \eta)$ is a negligible function of η.*

We remark that the definition above allows the adversary to change the group membership in an adaptive way, but does not permit *adaptive corruption* of the users, i.e. the set of corrupted users must be chosen before the protocol starts executing.

4.2 Computationally Sound Security in the Dolev-Yao Model

We now define security of multicast key distribution in the Dolev-Yao framework and for this we consider a special class of key distribution protocols that encompasses most of the protocols used in practical applications. Let $\Gamma_F = \{\mathcal{I}_F, \mathcal{C}_F, \mathcal{U}_F\}$ denote a multicast key distribution protocol in the Dolev-Yao framework. The program $\mathcal{I}_F$ works just as $\mathcal{I}$ in the previous definition except that it initializes the state of every user u_i as a fixed symbolic key K_i that is unique to that user and the state of the center as the set of all the unique keys $K_1, K_2, \cdots, K_N$, where N is a bound on the number of users.[6] The program $\mathcal{C}_F$ takes commands of the form $add(u_i)$ and $del(u_i)$ as before but for each command

[5] We remark that this definition can be made stronger by giving to the adversary either the key $k_{t'}$ (if $b = 0$) or a random key $k'_{t'}$ (if $b = 1$) at every time instant t' for which $B \cap G_{t'} = \emptyset$ (instead of giving the set of these keys, $\bar{k}_t$ or $\bar{k}'_t$, at the end of the experiment as is done above). This strengthening does not affect our result in any way and we use the above definition only for the sake of simplicity.

[6] In practice, the group center can store a compact representation of all these keys using a pseudorandom function.

$\mathtt{com}_t$, returns an *expression* M_t (denoting the rekey message for time t). The internal state of $\mathcal{C}_F$ at time t consists of all unique keys, all rekey messages sent till time t and the group composition at time t, G_t. $\mathcal{U}_F$ takes a user index i as input and returns a key, K_t^i, that can be obtained by applying the Dolev-Yao rules on all the rekey messages received till the current time, given the knowledge of the key K_i. K_t^i should also be such that it is not used as an encryption key in any of the rekey messages sent by the group center at any time[7]. Again, for correctness we require that for all time instants t, K_t^i be identical for each group member u_i at time t. We let $\bar{M}_t$ denote the expression $(M_1, M_2, \ldots, M_t)$ and for any set $B \subseteq U$, we let K_B denote the set of unique keys of all users in the set $B \subseteq U$.

Definition 4. *A multicast key distribution protocol Γ_F is secure in the Dolev-Yao framework if for every sequence of commands, $\mathtt{com}_1, \mathtt{com}_2, \cdots, \mathtt{com}_t$ and for every subset $B \subseteq U$, the following holds: Let $K_{t'}$ and $G_{t'}$ be the group key and group member set at time $t' \leq t$, and let T be the set of all t' such that $B \cap G_{t'} = \emptyset$. Then, $((\bar{M}_t, K_T), K_B) \simeq ((\bar{M}_t, K'_T), K_B)$, where $K_T = \{K_{t'} : t' \in T\}$ and K'_T is a set of $|T|$ fresh keys.*

For any protocol Γ_F in the Dolev-Yao framework, the translation of Γ_F in the computational framework with respect to a symmetric encryption scheme Π, is the protocol Γ_F^{Π} which behaves identically to Γ_F with the difference that a key assignment τ is generated for the set of all keys ever used in the protocol execution (using the key generation algorithm for Π) and each symbolic expressions M (a key or a rekey message) used in Γ_F is replaced with the bitstring interpretation of M, $[\![M]\!]_{\Pi,\tau}$. Using our soundness theorem, we can now show the following connection between the above two definitions.

Theorem 2. *Let Γ_F be a multicast key distribution protocol in the Dolev-Yao framework with the property that for any sequence of commands $\mathtt{com}_1, \cdots, \mathtt{com}_t$, the sequence of rekey messages, $M_1, M_2, \cdots, M_t$, returned by the center's program $\mathcal{C}_F$ is a legal sequence. Let Π be any* ind-cpa *secure symmetric encryption scheme. If Γ_F is secure in the Dolev-Yao framework (Definition 4), then Γ_F^{Π} is secure in the computational framework (Definition 3).*

Proof (Sketch): Suppose, towards contradiction, that Γ_F satisfies Definition 4, but Γ_F^{Π} does not satisfy Definition 3. Let $\mathcal{A}$ be a computational adversary and $B \subset U$ a set of initially corrupted users such that $\mathbf{Adv}_{\Gamma_F^{\Pi}}^{gkd}(\mathcal{A}, B, \eta)$ is non-negligible (in η). Given any such choice of $\mathcal{A}$ and B, we can build an adversary $\mathcal{A}'$ that uses $\mathcal{A}$ as a black-box and has non-negligible advantage in the experiment $\mathbf{Expt}_{\Pi}^{adpt}$ defined in our soundness theorem. $\mathcal{A}'$ first queries its oracle on the unique keys of all users in B and invokes $\mathcal{A}$ on input B and the corresponding keys. For any query $\mathtt{com}_{t'}$ of $\mathcal{A}$, $\mathcal{A}'$ uses the program $\mathcal{C}_F$ to determine the rekey message $M_{t'}$ and uses its oracle to determine the computational interpretation

[7] The reason we introduce this requirement is that if a key is used to encrypt a message, then the key is no necessarily pseudorandom anymore, as the encryption scheme may leak partial information about the key.

of the same (which it then returns to $\mathcal{A}$). Finally, $\mathcal{A}'$ queries its oracle on the pair (K_T, K'_T) where $T = \{t' : B \cap G_{t'} = \emptyset\}$ and K'_T is a set of $|T|$ fresh (symbolic) keys and the reply is passed on to $\mathcal{A}$. $\mathcal{A}'$ outputs whatever $\mathcal{A}$ outputs.

Given that the sequence of rekey messages generated in any run of Γ_F is a legal sequence and the fact that no key in K_T is ever used as an encryption key in any of the messages, it follows that the adversary $\mathcal{A}'$ constructed above is a legal adversary. It is easy to see that the advantage of $\mathcal{A}'$ in $\mathbf{Expt}_{\Pi}^{adpt}$ is exactly the same as that of $\mathcal{A}$ in $\mathbf{Expt}_{\Gamma_F^{\Pi}}^{gkd}$ (which means that if the latter is a non-negligible quantity, so is the former). This leads us to a contradiction of our soundness theorem. ▮

We remark that many practical group key distribution schemes (e.g., [23]) satisfy the precondition of Theorem 2 (that requires all sequences of rekey messages generated by the protocol to be legal sequences). Moreover, these protocols can be easily proved secure in the symbolic framework. It follows that their natural implementation is also secure in the computational framework.

5 Proof of the Soundness Theorem

This section provides an overview of the proof of our soundness theorem. More details appear in the full version of the paper [17].

5.1 Defining Orders for Legal Sequences of Expressions

For any acyclic expression $M \in \mathbf{Exp}$, the "encrypts" relation defines a partial order on the keys in M and we consider the restriction of this partial order on just its unrecoverable encryption keys. Any total order on the set of unrecoverable encryption keys of M that is consistent with such a partial order is called a *good order* for M. For example, in the expression

$$M := (((K_1, \{K_2\}_{K_3}), \{(K_6, \{K_1\}_{K_3})\}_{K_2}), \{K_4\}_{K_5})$$

the unrecoverable encryption keys are K_2, K_3 and K_5, and we have $K_3 \rightarrow_M K_2$. This gives us a partial order $K_3 \le K_2$ on $\mathbf{UEncKeys}(M)$ and so the good orders for M are $K_3 \le K_2 \le K_5$, $K_3 \le K_5 \le K_2$ and $K_5 \le K_3 \le K_2$.

We now re-interpret the definition of legal sequences of expressions given in Section 3. Recall that for any such sequence, $M[1], M[2], \cdots, M[q]$, the expression $M = M[1..q]$ is acyclic and for any $i < j \le q$, no unrecoverable encryption key in $M[1..i]$ is a message key in $M[j]$. The latter condition implies that for any $i \in \{1, \cdots, q-1\}$ no unrecoverable encryption key in $M[1..i]$ can be recoverable in $M[1..i+1]$, and therefore the sets

$$\mathbf{UEncKeys}(M[1\ldots 1]) \subseteq \mathbf{UEncKeys}(M[1\ldots 2]) \subseteq \cdots \subseteq \mathbf{UEncKeys}(M[1..q])$$

form a monotonically nondecreasing sequences.

This relation enables us to partition the unrecoverable encryption keys of M into q sets such that the ith set in the partition, say $\mathbf{UEnc}_i$, contains the keys that are used as encryption keys in $M[i]$ but in none of $M[1], M[2], \cdots, M[i-1]$, i.e. $\mathbf{UEnc}_i := \mathbf{UEncKeys}(M[1..i]) \setminus \mathbf{UEncKeys}(M[1..i-1])$. The definition of legal sequences implies that for any $i < j \leq q$, no key from $\mathbf{UEnc}_j$ can encrypt any key from $\mathbf{UEnc}_i$ in M. Now, using the fact that M is acyclic, we can find a good order $\leq$ for every $M[1..i]$ such that for any $1 \leq i_1 < j_1 \leq i$, the keys from $\mathbf{UEnc}_{i_1}$ "precede" all the keys from $\mathbf{UEnc}_{j_1}$ in $\leq$ i.e. for all $K_{i'} \in \mathbf{UEnc}_{i_1}$ and $K_{j'} \in \mathbf{UEnc}_{j_1}$, $K_{i'} \leq K_{j'}$. We select the lexicographically first order among all orders having this property and denote it by $\leq_{M[1..i]}$. The order $\leq_{M[1..1]}$ is just the lexicographically first good order defined on $M[1]$ and for each $i < q$, the ordering produced by $\leq_{M[1..i-1]}$ is a prefix of that produced by $\leq_{M[1..i]}$.

As an example, consider the following sequence of expressions:

$$
\begin{aligned}
M[1] &= (K_1, \{K_2\}_{K_3}) \\
M[2] &= \{(K_6, \{K_1\}_{K_3})\}_{K_2} \\
M[3] &= \{K_4\}_{K_5}
\end{aligned}
$$

Observe that this sequence is consistent with our definition of legal sequences. The expressions $M[1..2]$ and $M[1..3]$ are

$$
\begin{aligned}
M[1..2] &= ((K_1, \{K_2\}_{K_3}), \{K_6, \{K_1\}_{K_3}\}_{K_2}) \\
M[1..3] &= (((K_1, \{K_2\}_{K_3}), \{K_6, \{K_1\}_{K_3}\}_{K_2}), \{K_4\}_{K_5})
\end{aligned}
$$

We have $\mathbf{UEncKeys}(M[1..1]) = \{K_3\}$; $\mathbf{UEncKeys}(M[1..2]) = \{K_2, K_3\}$ and $\mathbf{UEncKeys}(M[1..3]) = \{K_2, K_3, K_5\}$ and so $\mathbf{UEnc}_1 = \{K_3\}; \mathbf{UEnc}_2 = \{K_2\}$ and $\mathbf{UEnc}_3 = \{K_5\}$. The lexicographically first good orders for $M[1..2]$ and $M[1..3]$ are given by $K_3 \leq K_2$ and $K_3 \leq K_2 \leq K_5$. These relations are denoted $\leq_{M[1..2]}$ and $\leq_{M[1..3]}$ respectively.

5.2 Defining hybrid oracles

The proof of the soundness theorem uses a hybrid technique. We define a set of $2n + 2$ hybrid oracles (where $n = |\mathbf{Keys}|$)[8] and relate the success probability of any legal adversary in distinguishing between any neighboring pair of these oracles to its success probability in distinguishing between the instances of the oracle $\mathcal{O}_{\Pi,b}$ (viz. $\mathcal{O}_{\Pi,0}$ and $\mathcal{O}_{\Pi,1}$) used in experiment $\mathbf{Expt}_{\Pi}^{adpt}$. We then use this relation to show how any legal adversary with a non-negligible advantage in $\mathbf{Expt}_{\Pi}^{adpt}$ (i.e. a non-negligible success probability in distinguishing between $\mathcal{O}_{\Pi,0}$ and $\mathcal{O}_{\Pi,1}$) can be used to mount a successful attack against the `ind-cpa` security of the underlying encryption scheme Π.

We denote the hybrid oracles by $\mathcal{O}_{\Pi,0}^0, \mathcal{O}_{\Pi,0}^1, \cdots \mathcal{O}_{\Pi,0}^n, \mathcal{O}_{\Pi,1}^n, \mathcal{O}_{\Pi,1}^{n-1}, \cdots, \mathcal{O}_{\Pi,1}^0$. The extreme oracles, $\mathcal{O}_{\Pi,0}^0$ and $\mathcal{O}_{\Pi,1}^0$, correspond to the instantiations of $\mathcal{O}_{\Pi,b}$

[8] Without loss of generality, the number of key symbols that can potentially be used by the adversary in generating queries can be assumed to be a fixed polynomial in the security parameter.

with $b = 0$ and $b = 1$ respectively. The behavior of oracle $\mathcal{O}^0_{\Pi,0}$ is close to that of $\mathcal{O}^1_{\Pi,0}$, the behavior of $\mathcal{O}^1_{\Pi,0}$ is close to that of $\mathcal{O}^2_{\Pi,0}$ and so on up to $\mathcal{O}^n_{\Pi,0}$. Similarly, $\mathcal{O}^n_{\Pi,1}$'s behavior is similar to that of $\mathcal{O}^{n-1}_{\Pi,1}$'s, $\mathcal{O}^{n-1}_{\Pi,1}$'s close to $\mathcal{O}^{n-2}_{\Pi,1}$'s and so on up to $\mathcal{O}^0_{\Pi,1}$. For each $i \in \{1, 2, \cdots, n\}$, the oracle $\mathcal{O}^i_{\Pi,0}$ is defined as follows:

Oracle $\mathcal{O}^i_{\Pi,0}(\eta)$

1. Let $\tau \xleftarrow{\$} \mathcal{K}^{n+2}(\eta)$
2. For the jth query received, $(M_0[j], M_1[j])$, do the following:
   (a) Compute the order $\leq_{M_0[1..j]}$ and let S be the set of those keys in $M_0[j]$ that are among the smallest i keys of $\leq_{M_0[1..j]}$.
   (b) Let M^{new} be the expression obtained by substituting, for all $K_l \in S$, all sub-expressions in $M_0[j]$ of the form $\{M'\}_{K_l}$ with $\{\texttt{struct}(M')\}_{K_l}$.
   (c) Return $[\![M^{new}]\!]_{\Pi,\tau}$.

The oracles $\mathcal{O}^i_{\Pi,1}$ (for $i \in [n]$) are defined analogously with the difference that in steps 2(a) and 2(b), $M_0[j]$ gets replaced by $M_1[j]$ and $\leq_{M_0[1..j]}$ by $\leq_{M_1[1..j]}$. The following fact about oracles $\mathcal{O}^n_{\Pi,0}$ and $\mathcal{O}^n_{\Pi,1}$ is easy to deduce:

Lemma 1. *Whenever the queries received by the oracles come from a legal adversary, the two oracles $\mathcal{O}^n_{\Pi,0}$ and $\mathcal{O}^n_{\Pi,1}$ have identical behavior i.e. for any sequence of queries, the distribution of the replies given by one of them is exactly the same as that of the replies given by the other.* ▮

For any $i \in [n]$ and $b \in \{0, 1\}$, we define the advantage of A in distinguishing between oracles $\mathcal{O}^i_{\Pi,b}$ and $\mathcal{O}^{i-1}_{\Pi,b}$, $\mathbf{Adv}^{adpt}_{\Pi,i,b}$, as the following quantity:

$$\mathbf{Adv}^{adpt}_{\Pi,i,b}(A, \eta) = \left| \mathbf{P}[A^{\mathcal{O}^i_{\Pi,b}(\eta)}(\eta) = 1] - \mathbf{P}[A^{\mathcal{O}^{i-1}_{\Pi,b}(\eta)}(\eta) = 1] \right|$$

The following lemma relates these advantages to the advantage of A in $\mathbf{Expt}^{adpt}_{\Pi}$.

Lemma 2. $\sum_{i=1}^{n} \sum_{b \in \{0,1\}} \mathbf{Adv}^{adpt}_{\Pi,i,b}(A, \eta) \geq \mathbf{Adv}^{adpt}_{\Pi}(A, \eta)$ ▮

5.3 The reduction

Given any legal adversary A in experiment $\mathbf{Expt}^{adpt}_{\Pi}$, we construct a distinguisher D attacking the `ind-cpa` security of Π such that the advantage of D in performing an `ind-cpa` attack on Π is related (by a polynomial multiplicative factor) to A's advantage in $\mathbf{Expt}^{adpt}_{\Pi}$. This essentially implies that any successful attack in $\mathbf{Expt}^{adpt}_{\Pi}$ can be effectively translated into an attack on the underlying encryption scheme itself.

Our construction of D will be such that the advantage of D in $\mathbf{Expt}^{ind-cpa}_{\Pi}$ will be $1/poly$ times the *expected* advantage of A in distinguishing between oracles

$\mathcal{O}^{i}_{\Pi,b}$ and $\mathcal{O}^{i-1}_{\Pi,b}$, where i and b are treated as random variables sampled uniformly from $\{1, \cdots, n\}$ and $\{0, 1\}$. More precisely, the construction will be such that

$$\begin{aligned}\mathbf{Adv}^{ind-cpa}_{\Pi}(D,\eta) &= \frac{1}{n}\mathbf{E}_{i \xleftarrow{\$}[n]; b \xleftarrow{\$}\{0,1\}}\left(\mathbf{Adv}^{adpt}_{\Pi,i,b}(A,\eta)\right) \\ &= \frac{1}{n}\sum_{i=1}^{n}\sum_{b\in\{0,1\}}\left(\frac{1}{n}\cdot\frac{1}{2}\cdot\mathbf{Adv}^{adpt}_{\Pi,i,b}(A,\eta)\right)\end{aligned}$$

Now, applying Lemma 2 we get

$$\mathbf{Adv}^{ind-cpa}_{\Pi}(D,\eta) \geq \frac{1}{2n^2}\mathbf{Adv}^{adpt}_{\Pi,i,b}(A,\eta)$$

Thus, a non-negligible advantage of A in experiment $\mathbf{Expt}^{adpt}_{\Pi}$ would imply a non-negligible advantage of D in $\mathbf{Expt}^{ind-cpa}_{\Pi}$ and the theorem would follow immediately from this.

The Construction. The distinguisher D works as follows: it first selects a random number i' in the range $\{1, \cdots, n\}$ and a random bit $b' \in \{0, 1\}$ and then tries to simulate the behavior of the oracle pair $\{\mathcal{O}^{i'}_{\Pi,b'}, \mathcal{O}^{i'-1}_{\Pi,b'}\}$ using its own oracle $\mathcal{LR}_{\Pi,b}$. D runs A inside it and answers A's queries using its simulated setup. To carry out the simulation, it guesses a value i randomly from $\{1, \cdots, n\}$ and hopes that each query, $(M_0[j], M_1[j])$, issued by A would be such that the i'th key in the order $\leq_{M_{b'}[1..j]}$ is K_i (or else the number of unrecoverable encryption keys in $M_{b'}[1..j]$ is smaller than i' and K_i is neither a recoverable key nor an unrecoverable encryption key in $M_{b'}[1..j]$). If D fails in its guess, it gives up and outputs 0. Else, it treats the key used by $\mathcal{LR}_{\Pi,b}$ as corresponding to K_i and answers A's queries in such a way that the behavior of $\mathcal{LR}_{\Pi,b}$ with $b = 0$ (resp. $b = 1$) corresponds to the simulation of the oracle $\mathcal{O}^{i'-1}_{\Pi,b'}$ (resp. $\mathcal{O}^{i'}_{\Pi,b'}$).

Adversary $D^{\mathcal{LR}_{\Pi,b}(\eta)}(\eta)$

1. Let $i' \xleftarrow{\$} \{1, 2, \cdots, n\}$ and $b' \xleftarrow{\$} \{0, 1\}$.
2. Guess a value $i \xleftarrow{\$} \{1, 2, \cdots, n\}$.
3. Generate a key vector τ as follows: $(\tau[1], \ldots, \tau[i-1], \tau[i+1], \ldots, \tau[n+2]) \xleftarrow{\$} \mathcal{K}^{n+1}(\eta)$ (the ith entry in τ is empty and the rest are random keys).
4. Run $A(\eta)$.
5. When A issues the jth query $(M_0[j], M_1[j])$, do the following:
   (a) Compute the order $\leq_{M_{b'}[1..j]}$. Check if either K_i is the i'th key in $\leq_{M_{b'}[1..j]}$; OR $|\mathbf{UEncKeys}(M_{b'}[1..j])| < i'$ and $K_i \notin \mathbf{UEncKeys}(M_{b'}[1..j]) \cup \mathbf{RecKeys}(M_{b'}[1..j])$
   (b) If so, do the following:
      i. Let S be the set of those keys in $M_{b'}[j]$ that are among the *smallest* $(i'-1)$ keys of $\leq_{M_{b'}[1..j]}$.
      ii. Let M^{new} be the expression obtained by substituting, for all $K_l \in S$, all sub-expressions in $M_{b'}[j]$ of the form $\{M'\}_{K_l}$ with $\{\texttt{struct}(M')\}_{K_l}$.

iii. Return $Sample^{\mathcal{LR}_{\Pi,b}}(M^{new}, K_i, \tau)$ to A.

(c) Else, output 0 and halt.

6. Output whatever A outputs.

The crux of the code lies in the subroutine *Sample* (invoked at the end of step 5(b)) which is given below

Procedure $Sample^{\mathcal{LR}_{\Pi,b}}(M, K_i, \tau)$

1. If M is a constant or $\mathbf{c}$, return the corresponding bitstring.
2. If $M = K_l$ (for some $l \in [n]$), return $\tau[l]$. If $M = K'$, return $\tau[n+1]$.
3. If $M = (M_1, M_2)$, return $(Sample(M_1, K_i, \tau), Sample(M_2, K_i, \tau))$.
4. If $M = \{M'\}_{K_l}$ and $l \neq i$, return $\mathcal{E}_{\tau[l]}(Sample(M', K_i, \tau))$. If $M = \{M'\}_K$, return $\mathcal{E}_{\tau[n+2]}(Sample(M', K_i, \tau))$.
5. If $M = \{M'\}_{K_l}$ and $l = i$, return $\mathcal{LR}_{\Pi,b}(Sample(M', K_j, \tau), [\![\texttt{struct}(M')]\!]_{\Pi,\tau})$.

The proofs of the two lemmas and the analysis of the distinguisher can be found in the full version of the paper[17].

6 Future Work

We have proved a generalization of the soundness theorem of [2] in which the adversary can issue a sequence of adaptively chosen expressions, rather than a single expression, and demonstrated the usefulness of the theorem in an application to secure multicast key distribution. For simplicity, in this paper we considered a language of expressions that make use of only symmetric encryption operations, but most of the techniques can be easily extended to other cryptographic primitives whose security can be expressed as an indistinguishability property. Examples of such primitives are public key encryption, in which two different keys are used to encrypt and decrypt messages, and pseudorandom number generators, that can be used to expand a key into a sequence of multiple seemingly independent keys. Some of these extensions (e.g., the use of pseudorandom generators) are especially interesting in the context of multicast security protocols as the protocol of [6] (which was shown to be optimal in the Dolev-Yao model in [18]) makes use of these operations.

The proof of our soundness theorem introduces a syntactic restriction (besides the acyclicity condition already used in [2]) about the order in which each key is used as a message or as an encryption key. An interesting question is whether either restriction can be lifted, possibly using a special encryption scheme with additional security properties (a good candidate might be non-committing encryption introduced by Canetti, Feige, Goldreich and Naor in [5]). Although most practical protocols satisfy the syntactic restrictions in our soundness theorem, removing the ordering restriction would allow to model attack scenarios with adaptive corruption of users, where, when the adversary wants to corrupt user i (holding a secret key k_i as its internal state) it can simply issue the expression K_i to learn the value of the key. Currently this is allowed only if K_i has not already been used to encrypt other messages. Designing protocols that are secure against adaptive corruption raises issues similar to the selective decommitment problem discussed in Section 3, and is not easily addressed using the techniques developed in this paper. We leave the investigation of multicast key distribution protocols secure under adaptive corruption of the users to future work.

References

1. M. Abadi and J. Jürjens. Formal eavesdropping and its computational interpretation. In N. Kobayashi and B. Pierce, editors, *Proceedings of the 4th International Symposium on Theoretical Aspects of Computer Software - TACS 2001*, volume 2215 of *Lecture Notes in Computer Science*, pages 82–94, Sendai, Japan, Oct. 2001. Springer-Verlag.
2. M. Abadi and P. Rogaway. Reconciling two views of cryptography (the computational soundness of formal encryption). *Journal of Cryptology*, 15(2):103–127, 2002.
3. M. Backes and B. Pfitzmann. Symmetric encryption in a simulatable Dolev-Yao style cryptographic library. In *Proceedings of the 17th IEEE computer security foundations Workshop*, pages 204–218, Pacific Grove, CA, USA, June 2004. IEEE Computer Society.
4. M. Backes, B. Pfitzmann, and M. Waidner. A Composable Cryptographic Library with Nested Operations. In *Proceedings of the 10th ACM conference on computer and communications security - CCS 2003*, pages 220–230, Washington, DC, USA, Oct. 2003. ACM.
5. R. Canetti, U. Feige, O. Goldreich, and M. Naor. Adaptively Secure Multiparty Computation. In *Proceedings of the twenty-eighth annual ACM symposium on the theory of computing - STOC '96*, pages 639–648, Philadelphia, Pennsylvania, USA, May 1996. ACM.
6. R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor, and B. Pinkas. Multicast security: A taxonomy and some efficient constructions. In *INFOCOM 1999. Proceedings of the Eighteenth Annual Joint conference of the IEEE computer and communications societies*, volume 2, pages 708–716, New York, NY, Mar. 1999. IEEE.
7. R. Canetti, T. Malkin, and K. Nissim. Efficient communication-storage tradeoffs for multicast encryption. In J. Stern, editor, *Advances in Cryptology - EUROCRYPT '99, Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques*, volume 1592 of *Lecture Notes in Computer Science*, Prague, Czech Republic, May 1999. Springer-Verlag.
8. D. Dolev, C. Dwork, and M. Naor. Nonmalleable Cyptography. *SIAM Journal on Computing*, 30(2):391–437, 2000. Preliminary version in STOC 1991.
9. C. Dwork, M. Naor, O. Reingold, and L. Stockmeyer. Magic Functions. *Journal of the ACM*, 50(6):852–921, Nov. 2003.
10. V. Gligor and D. O. Horvitz. Weak Key Authenticity and the Computational Completeness of Formal Encryption. In D. Boneh, editor, *Advances in cryptology - CRYPTO 2003, proceedings of the 23rd annual international cryptology conference*, volume 2729 of *Lecture Notes in Computer Science*, pages 530–547, Santa Barbara, California, USA, Aug. 2003. Springer-Verlag.

11. S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sience*, 28(2):270–299, 1984. Preliminary version in Proc. of STOC 1982.
12. J. C. Herzog. *Computational Soundness for Standard Assumptions of Formal Cryptography.* PhD thesis, Massachusetts Institute of Technology, Boston, USA, 2004.
13. R. Impagliazzo and B. Kapron. Logics for Reasoning about Cryptographic Constructions. In *Proceedings of the 44rd annual symposium on foundations of computer science - FOCS 2003*, pages 372–383, Cambridge, MA, USA, Nov. 2003. IEEE.
14. P. Laud. Symmetric Encryption in Automatic Analyses for Confidentiality against Active Adversaries. In *IEEE symposium on security and Privacy*, pages 71–85, Berkeley, CA, USA, May 2004. IEEE Computer Society.
15. P. D. Lincoln, J. C. Mitchell, M. Mitchell, and A. Scedrov. A probabilistic polytime framework for protocol analysis. In *Proceedings of the fifth ACM conference on computer and communications security - CCS '98*, pages 112–121, San Francisco, California, USA, Nov. 1998. ACM.
16. D. Micciancio. Towards Computationally Sound Symbolic Security Analysis, June 2004. Tutorial. Slides available at http://dimacs.rutgers.edu/Workshops/Protocols/slides/micciancio.pdf.
17. D. Micciancio and S. Panjwani. Adaptive Security of Symbolic Encryption, Nov. 2004. Full version of this paper. Available from http://www-cse.ucsd.edu/users/spanjwan/papers.html.
18. D. Micciancio and S. Panjwani. Optimal communication complexity of generic multicast key distributio. In C. Cachin and J. Camenisch, editors, *Advances in cryptology - EUROCRYPT 2004, proceedings of the internarional conference on the theory and application of cryptographic techniques*, volume 3027 of *Lecture Notes in Computer Science*, Interlaken, Switzerland, May 2004. Springer-Verlag.
19. D. Micciancio and B. Warinschi. Completeness theorems for the abadi-rogaway logic of encrypted expressions. *Journal of Computer Security*, 12(1):99–129, 2004. Preliminary version in WITS 2002.
20. D. Micciancio and B. Warinschi. Soundness of Formal Encryption in the presence of Active Adversaries. In M. Naor, editor, *Theory of cryptography conference - Proceedings of TCC 2004*, volume 2951 of *Lecture Notes in Computer Science*, pages 133–151, Cambridge, MA, USA, Feb. 2004. Springer-Verlag.
21. C. Rackoff and D. R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In J. Feigenbaum, editor, *Advances in Cryptology - CRYPTO '91, Proceedings*, volume 576 of *Lecture Notes in Computer Science*, Santa Barbara, California, USA, Aug. 1991. Springer-Verlag.
22. S. Rafaeli and D. Hutchinson. A survey of key management for secure group communication. *ACM Computing Surveys*, 35(3):309–329, Sept. 2003.
23. C. K. Wong, M. Gouda, and S. S. Lam. Secure group communications using key graphs. *IEEE/ACM Transactions on Networking*, 8(1):16–30, Feb. 2000. Preliminary version in SIGCOMM 1998.