Symbolic Security Criteria for Blockwise Adaptive Secure Modes of Encryption

Catherine Meadows

Naval Research Laboratory (USA
Washington, DC 20375
Email: catherine.meadows@nrl.navy.mil

Abstract. Symbolic methods for reasoning about the security of cryptographic systems have for some time concentrated mainly on protocols. More recently, however, we see a rising interest in the use of symbolic methods to reason about the security of algorithms as well, especially algorithms that are built by combining well-defined primitives. For this kind of application two things are generally required: the ability to reason about term algebras obeying equational theories at the symbolic level, and the ability to prove computational soundness and completeness of the symbolic model. It is often challenging to provide both these capabilities, especially for an adaptive adversary that can perform chosen plaintext or ciphertext attacks. In this paper we derive sound and complete symbolic criteria for computational security against adaptive chosen plaintext attacks of a class of modes of encryption. These apply to any scheduling policy used to send the cipher text, ranging from the messagewise schedule, in which ciphertext blocks are sent to the adversary only after all the plaintext blocks have been received, to the blockwise schedule, in which ciphertext blocks are sent as soon as they are computed. We also discuss how this approach could extended to larger classes of modes, and how could it be applied to the automatic synthesis of cryptosystems.

1 Introduction

Symbolic methods for the analysis of cryptographic systems have been increasing in popularity, and they have proven to be useful for addressing a number of different cryptographic problems. One use of such a symbolic approach is to provide computational guarantees; that is, security on the symbolic level should guarantee security at the computational level. This line of research started in the analysis of cryptographic protocols, but has proved particularly effective in the analysis of cryptographic algorithms, in particular algorithms built by combining different cryptographic primitives, which can often be represented symbolically.

However, a problem arises when trying to combine symbolic algebras satisfying equational theories with more powerful adversarys. Early on, researchers were able to obtain soundness and completness results for the powerful Dolev-Yao adversary for free symbolic algebras. However, once the algebra satisfies some equational theory, the problem becomes harder. Indeed most results on

computational soundness and completeness in the face of *adaptive* adversarys put limitations on the adversary's adaptive capabilities, (e.g [17, 10]) allowing the adversary to choose adaptively from a previously chosen set of values, but not to input unrestricted plaintext.

There have, however, been some approaches that have provided symbolic conditions for security against more robust adaptive adversarys. For example, Malezomoff et al. [19] and Hoang et al. [14] derive security conditions for different types of modes of encryption that hold for a *messagewise adaptive adversary*. A mode of encryption is a process that is applied to block ciphers, which can encrypt only one block at a time, to produce encryptions of messages that are more than one block long. A messagewise adaptive adversary has access to an oracle that encrypts messages supplied by the adversary, but does not see any blocks of the encrypted message until it has submitted all the plaintext blocks. The conditions of [14, 19] can be stated and verified in a symbolic algebra that has both an exclusive-or operator and an encryption operator. This makes it possible to check the conditions symbolically.

In this paper symbolic guarantees symbolic criteria for security against adaptive chosen plaintext attacks against a class of modes of encryption using exclusive-or and block ciphers that are both sound and complete with respect to the computational model. Secondly, these criteria hold no matter what schedule is used to send the ciphertext blocks to the adversary. In particular they can be used both for the messagewise adaptive adversary and the *blockwise adaptive adversary* [7, 15], that has access to an oracle that sends a ciphertext block immediately after receiving a plaintext block. Blockwise adaptive security is strictly stronger than message adaptive security; for example cipher block chaining is known to be message adaptive but not blockwise adaptive; see [7, 15]. In order to reason about blockwise adaptive security, we make use of a methodology based on the Baader-Schulz combination procedure [2] for combining unifiers for disjoint equational theories. We believe this methodology can be extended to other adaptive computational soundness and completeness problems as well.

The proof strategy we use is the following. We consider three different models of computation: a symbolic model with equational theories, the computational algebra of Baudet et al. [5], and the probabilistic polynomial time Turing machine (PPT) model used by cryptographers. In the symbolic model the adversary can compute and send only symbolic expressions. In the computational algebra model, the adversary has the computational power of a PPT Turing machine, but will send only messages that can be expressed via the computational algebra, which is a computational image of the symbolic algebra. In the PPT model we assume that the adversary may compute and send any message computable by a PPT algorithm. We define a security property for R-wise adaptive security based on IND$-CPA that we call IND$-RCPA, where R is any deterministic schedule without choice points used to send encrypted blocks to the adversary. We show that, given certain conditions on the mode, IND$-RCPA security holds if and only if the adversary is unable to force a mode of encryption oracle to force two terms from a certain class of encrypted terms to be equal, thus reducing

the problem of finding an attack to a unification problem. However, it is a unification problem with constraints, since the adversary can only use information it itself has created or that it has already received from the oracle to construct the unifier. The main result is the identification of a symbolic condition on the unification problem that holds if and only if no unifier can be computed with non-negligible probability by a PPT adversary.

The key problem is determining whether or not it is possible to construct a unifier that satisfies these constraints. We show that a unifier does so if and only if it satisfies a PPT analogue of the *linear constant restriction* defined for symbolic algebras. We divide our constrained unification problem into two, as in Baader-Schulz. However, instead of dividing the problem into two problems in the component equational theories, we divide into it into two problems that describe the programs executed respectively by the adversary and the oracle, as in Carmer and Rosulek [10]. We then show that there is a solution to the adversary's problem in all three models if and only if the unification problem has a solution realizing these constraints. We are able to provide characterizations of the problems with realizable constraints, thus giving our result.

The paper is organized as follows. In Section 2 we discuss background and related work. In Section 3 we present standard results and definitions for symbolic algebras that we use. In Section 4 we present symbolic and computational algebra models of cryptographic operations and show how they relate to symbolic and PPT adversaries. In Section 5 we present known results on soundness and completeness that we use. In Section 6 we present the mode of encryption problem as a program (the MOE program) executed by the adversary and an oracle. We give a computational definition of IND$-RCPA for MOE programs and show that it is equivalent to a collision-freeness property for a certain class of modes. In Section 7 we derive our symbolic conditions for security of MOE programs. In Section 8 we show that a MOE program is secure against a PPT adversary if and only it is secure against a symbolic adversary. Finally, in Section 9 we conclude the paper and discuss potential improvements and future applications of the methodology that we have developed.

2 Background and Related Work

Probably the earliest work on using symbolic methods for cryptographic proofs of security protocols in the face of adaptive adversarys is that of Backes, Pfitzmann, and Waidner beginning in [3], in which the underlying encryption primitives are defined in a cryptographic library, and the adversary is a full Dolev-Yao adversary that interacts with principals over a network it completely controls. In later work Micciancio and Panjwani [20] prove a computational soundness and completeness theorem for adaptive adversaries that applies to any CPA encryption scheme. However these works used the free equational theory, and so were not useful for algorithms constructed using cryptographic primitives, since this generally requires the modeling of non-trivial equational theories.

More recent work has addressed computational soundness and completeness of symbolic security in different equational theories, but in general this work has only led to results for weaker, e.g. passive adversaries. One notable exception is the work of Kremer and Mazaré [17], which addresses the soundness of static equivalence, a symbolic concept used to express indistinguishability properties, in the face of an adaptive adversary. However, this does not allow arbitrary input from the adversary; instead the adversary is restricted to adaptively choosing from a set of protocol executions with which it interacts passively. This is because the soundness results require that the input to the protocol obey certain typing restrictions, which an adversary will not necessarily follow.

There has also been a substantial body of work that focuses explicitly on modes of encryption. Gagné et. al [13] have developed a Hoare logic for proving semantic security of block cipher modes of encryption, and a program implementing the logic than can be used to automatically prove their security. However, their work concentrates on heuristically driven theorem proving techniques rather than evaluating symbolic security conditions. We also note the work of Bard [4], who considers circumstances under which security for modes of encryption can be reduced to a collision-freeness property. Although [4] does not address symbolic security, our approach to deriving collision-freeness criteria security owes much to it. The work of Malezomoff et al. [19] and Hoang et al. [14] is probably the closest to that in this paper. They prove messagewise adaptive chosen plaintext security of modes of encryption block ciphers [19] and authenticated modes of encryption using block ciphers with tweaks [14] by defining a set of symbolic conditions checked on automatically generated modes. These conditions are proved complete; however no guarantee is given that all secure modes satisfy them. Our own results on symbolic conditions can be thought of as an extension of the criteria [19] to sound and complete criteria for security against an R -wise chosen plaintext adaptive adversary.

Another approach to which ours is closely related is Carmer and Rosulek's Linicrypt model [10], which is used to reason about algorithms that involve hash functions or block ciphers, and additive group operations, but in which the adversary is unable to compute the hash functions/block ciphers itself. This has applications to a number of types of algorithms, including block cipher modes of encryption, one-time lightweight signature schemes, and garbled circuits. Both our work and [10] follow the procedure of dividing the symbolic program into an adversarial program and an oracle program.

Although the general Linicrypt model is designed to allow adversarial input, the work in [10] limits itself to *input-free* Linicrypt programs; however the adversary is allowed to choose among a fixed set of input-free Linicrypt programs to run, similar to the case of [17]. In this paper we do not attempt to achieve the same breadth as in Linicrypt, concentrating instead on reasoning about a powerful adversary in the context of a specific problem. However we believe the scope of our results can be extended in future work.

3 Symbolic Preliminaries

A *signature* is a finite set of function symbols Σ of different arities. An *order-sorted signature* is a signature together with a finite poset of sorts $(\mathsf{S}, \leq)$, such that each function symbol and each argument of a function symbol is assigned a sort. We assume an S-sorted family $\mathcal{X} = \{\mathcal{X}_\mathsf{s}\}_{\mathsf{s}\in\mathsf{S}}$ of disjoint countably infinite variable sets $\mathcal{X}_\mathsf{s}$. We write $\mathcal{T}_\Sigma(\mathcal{X})$ and $\mathcal{T}_\Sigma$ for the corresponding set of all possible terms and ground terms. $\mathcal{T}_\Sigma(\mathcal{X})$ and $\mathcal{T}_\Sigma$ are referred to as *term algebras*.

We write $Var(t)$ and $Sym(t)$ for the set of variables (respectively function symbols) present in a term t. A *position* is either Λ or a sequence of positive integers $i_1.\ldots.i_k$. If t is a term and p is a position, we say that the subterm of t at p, denoted by $t|_p$, is t if $p = \Lambda$, and, if $p = q.r$ and $t|_q = f(s_1, \ldots, s_r, \ldots, s_k)$, then $t|_p = s_r$, and $t[u]_p$ is the result of replacing $t|_p$ by u in t. If $t|_p$ is defined, we say p is a *position of* t, and if $t|_p$ is (respectively, is not) a variable, we say that p is a variable (respectively, non-variable) position of t. The set of positions of a term t is written $Pos(t)$, and the set of non-variable positions $Pos_\Sigma(t)$. Thus, if $t = f(x, g(x, y))$, then $Pos(t) = \{\Lambda, 1, 2, 2.1, 2.2\}$, $t|_\Lambda = f(x, g(x, y))$, $t|_1 = x$, $t|_2 = g(x, y)$, $t|_{2.1} = x$, and $t|_{2.2} = y$. In addition, $t[z]_2 = f(x, z)$.

A *Σ-equation* is a pair $t = t'$. A set E of Σ-equations induces a congruence relation $=_E$ on terms $t, t' \in \mathcal{T}_\Sigma(\mathcal{X})$, so that $t =_E t'$ if and only if t can be made equal to t' via applications of equations from E. An *equational theory* is pair (Σ, E), where Σ is an order-sorted signature and E a set of Σ-equations. We will refer to a term algebra $\mathcal{T}_\Sigma(\mathcal{X})$ together with an equational theory (Σ, E) as $(\mathcal{T}_\Sigma(\mathcal{X}), E)$. Continuing our example above, suppose that $\Sigma = \{f/2, g/2, 0/0\}$, and $E = \{f(x, g(x, y)) = 0\}$. Then $g(f(g(z, z), g((g(z, z), w)), v) =_E g(0, v)$, by setting $x = g(z, z)$ and $y = v$ in the equation $f(x, g(x, y)) = 0$.

A *substitution* σ is a sort-preserving mapping from $\mathcal{X}$ to $\mathcal{T}_\Sigma(\mathcal{X})$ that is the identity on all but a finite subset of $\mathcal{X}$ known as the *domain* of σ. Substitutions are homomorphically extended to $\mathcal{T}_\Sigma(\mathcal{X})$. Application of σ to a term t is denoted by σt, and its restriction to a set of variables V is $\sigma|_V$. The composition of two substitutions is $(\sigma\theta)X = \sigma(\theta X)$ for $X \in \mathcal{X}$. A substitution σ is an *E-unifier* of a system of equations $S = \{\ldots, s_i =^? t_i. \ldots\}$, if $\sigma s_i =_E \sigma t_i$ for every $s_i =^? t_j \in S$. Thus, consider the equation $S = \{g((g(z_1, z_2), g(z_3, w)), v) =^? 0\}$ in the algebra (Σ, E) described in the previous paragraph. Then $\sigma : z_2 \mapsto z_1, \sigma : z_3 \mapsto g(z_1, z_1)$, is a unifier of S modulo E.

We will be interested in the algebra consisting of a number of *free symbols* that obey no equational theory, plus an exclusive-or operator $\oplus$ and a null operator 0. The $\oplus$ operator is associative and commutative and satisfies $X \oplus 0 = 0$ and $X \oplus X = 0$. Equality of two terms modulo this theory is equivalent to equality under the theory $(R_\oplus \uplus AC)$ where AC is the associative and commutative rules for $\oplus$, and $R_\oplus$ is a set of *rewrite rules*, $\{X \oplus 0 \to X,\ X \oplus X \to\ 0\}$ oriented from left to right. A rewrite rule $\ell \to r$ is applied to a term t by finding a position p of t such that $t|_p = \sigma\ell$ modulo AC for some substitution σ, and replacing t by $t[\sigma r]_p$. Thus $0 \oplus a \oplus b$ can be reduced to $a \oplus b$ by noting that $0 \oplus a \oplus b = (0 \oplus a) \oplus b = \sigma\ell \oplus b$ modulo AC, where ℓ is the left-hand side of $X \oplus 0 \to X$, and $\sigma X = a$. In addition, every term t reduces after a finite

number of steps to a *normal form* $\downarrow_{R_\oplus,AC} t$ to which no further rewrite rules can be applied, and this normal form is unique up to AC-equivalence.

4 Symbolic and Computational Models of Cryptographic Operations

In this section we present three different ways of modeling cryptographic operations: the first using symbolic algebras, the second using computational algebras that implement symbolic operations as cryptographic ones, and the third the standard computational model used in cryptography. We also give relevant soundness and completeness results from the literature.

4.1 Symbolic Algebra Representation of Cryptographic Operations

A *symbolic cryptographic algebra* is a sorted term algebra $(\mathcal{T}_\Sigma(\mathcal{X}), E)$ augmented with a quantifier ν. A bound variable $\nu.X$ denotes a random variable in the cryptographic theory. No substitutions may be made to bound variables. Thus, a bound variable behaves identically to a free constant, except that there are a countably infinite number of them. We say that a term is *closed* if all its variables are bound. Thus, $\nu X.\nu Y.f(g(X,Y))$ is closed, while $\nu X.f(g(X,Y))$ is not. We let $FVar(t)$ denote the free variables of t.

Such bound variables are often referred to as *names*, a terminology that comes from the applied pi calculus [1], but often used independently of it. We will refer to them as *random names*, to avoid confusion with identifiers of principals.

We will be interested in security properties that can be expressed in terms of sequences of messages created and/or observed by an adversary. Such sequences are expressed as *frames* in the symbolic algebra. These are defined below.

Definition 1. *Let $(\mathcal{T}_\Sigma(\mathcal{X}), E)$ be a term algebra with quantifier ν. A* frame *is a set of unquantified variables $\{x_1, \ldots, x_n\}$ together with a substitution ϕ on $\mathcal{T}_\Sigma(\mathcal{X})$ with domain $x_1, \ldots, x_n$. We will refer to a frame simply by ϕ when we can avoid confusion. and often refer to ϕ as $[\phi x_1, \ldots, \phi x_n]$,*

A frame represents a sequence of messages sent and received by participants in a protocol. Thus, a principal sends a message at step i by computing a substitution to X_i. For example, suppose that A sends a message $\nu r.r$ to B, B returns $\nu s.(r \oplus s)$, and A returns s. Then the corresponding frame is $\phi[X_1, X_2, X_3] = \nu r.\nu s.[r, r \oplus s, s]$,

There is generally more than one way of computing a substitution. For example, in the exchange above, B computes s by choosing a random nonce. A however computes s indirectly by first computing r, waiting until she receives $r \oplus s$, and computing $r \oplus r \oplus s =_{R_\oplus,AC} s$. We make this precise via the definition of substitution programs below.

Definition 2. *Let $(\mathcal{T}_\Sigma(\mathcal{X}), E)$ be a term algebra, and $t_1, \ldots, t_n$ be elements of $\mathcal{T}_\Sigma(\mathcal{X})$. A* substitution *program $\Theta : x_1, \ldots, x_n \mapsto t_1, \ldots, t_n$, is a program such that, if $\Theta x = t$, then $\Theta(x|_{p.r})$ is computed before $\Theta(x|_p)$.*

A substitution program is *computable by a principal* if it can be produced by applying operations available to that principal to terms it has previously received or produced. Thus, in the exchange between A and B given right before Definition 2, s is not computable by A until it has received $r \oplus s$. We will make this more formal later on when we introduce Mode of Encryption (MOE) programs.

4.2 Computational Algebra Semantics

In this section we present the concrete computational algebra semantics for symbolic cryptographic algebras from [17].

Definition 3. *Given a cryptographic term algebra* $(\mathcal{T}_\Sigma(\mathcal{X}), E)$ *with sorts* S*, the associated* computational algebra *is an algebra* $A(\mathcal{X})$ *consisting of*

- *for each sort* s*, a non-empty set of bitstrings* $[\![s]\!]_A$*, such that if* $\mathsf{s}_2 \leq \mathsf{s}_1$*, then* $[\![s_1]\!] \subseteq [\![s_2]\!]$*;*
- *for each* $f \in \Sigma$ *of sort* s *with arity* k *and arguments of sort* s_1 *through* s_k*, a function* $[\![f]\!]_A : [\![s_1]\!]_A \times \ldots \times [\![s_k]\!]_A \to [\![s]\!]_A$*.*
- *an effective procedure to draw random elements from* $[\![s]\!]_A$*, denoted by* $X \xleftarrow{A} [\![s]\!]_A$*;*
- *A countable set of sorted variables* $\mathcal{X}$*.*

We will refer to $A(\mathcal{X})$ simply as a computational algebra when we can avoid confusion.

Substitutions and substitution programs are defined for computational algebras analogously to the case of symbolic algebras. Gven a substitution program $\Theta = \{x_1 \mapsto t_1, \ldots x_n \mapsto t_n\}$, a concrete program $[\![\theta]\!]_A$ is computed as follows:

1. For each random name a of sort s appearing in $t_1, \ldots, t_n$, draw a value $\hat{a} \xleftarrow{R} [\![s]\!]_A$.
2. For each constant c appearing in $t_1, \ldots, t_n$, $\hat{c} = c$.
3. For $r = f(r_1, \ldots, r_k)$ appearing in $t_1, \ldots, t_n$, $\hat{r} = f(\hat{r_1}, \ldots, \hat{r_k})$.
4. For each unbound variable of sort s $\hat{X} = X$, and is also of sort s.
5. Return the value $\hat{\Theta} = \{x_1 \mapsto \hat{t_1}, \ldots, x_n \mapsto \hat{t_n}\}$.

We call $\hat{\Theta}$ the *output* of Θ.

We next introduce constructs that will be needed to reason about asymptotic security. We consider *families* of computational algebras (A_η) indexed by a complexity parameter η. The *concrete semantics* of a closed frame ϕ is a family of distributions over concrete frames $[\![\phi]\!]_{A_\eta}$. We will use the terminology $[\![\phi]\!]_\eta$ when we can avoid confusion, and define $\hat{\phi}_\eta$ to be the output of $[\![\phi]\!]_\eta$.

As an example, consider the term algebra $(\mathcal{T}_\Sigma(\mathcal{X}), (\oplus, AC)$ where $\Sigma = \{\oplus/2, 0/0\}$. Consider the term $\nu x.\nu y.(x \oplus y \oplus 0)$. The term in the concrete semantics is a program that, for each value of the security parameter η, chooses two length η bitstrings uniformly at random and returns the the bitwise exclusive-or of these and 0^η.

4.3 PPT Functions

In this section we develop a theory of PPT functions in a way analogous to the theory of computational algebras.

We consider programs P computable by a probabilistic Turing machine with an input tape x, a random input tape r, and one output tape. We denote this program by $\nu r_1 \dots \nu r_n . P(x_1, \dots, x_n, r_1, \dots, r_m)$, where the x_i are logical variables standing for strings taken from x, and the r_i are random names standing for strings taken from r. When we can avoid confusion, we denote by P by $P(x_1, \dots, x_n)$. Consider a family of probabilistic programs $\{P_\eta(x_1, \dots, x_n) | \eta \in \mathbb{Z}^+\}$ such that the time to compute P_η is a bounded by a polynomial function of η. We say that $P = \{P_\eta(x_1, \dots, x_n) | \eta \in \mathbb{Z}^+\}$ is a PPT function of $x_1, \dots, x_n$. Given a countable set of variables $\mathcal{X}$ we define $\mathcal{P}(\mathcal{X})$ to be the set of PPT functions defined on variables from $\mathcal{X}$. Terms from a computational algebra A may be embedded in the set of PPT functions in the obvious way. As in the other models, we say that P is closed if all variables are bound.

We are now ready to define PPT analogues of substitution and unification.

Definition 4. *Let $x_1, \dots, x_n$ be variables in $\mathcal{X}$. A* PPT substitution program *Θ on $x_1, \dots, x_n$ is a map that sends each x_i to a PPT function Q_i and is the identity on all other variables. We let $\widehat{\Theta}_\eta x$ denote an output of the program Θx.*

We now define unification as follows:

Definition 5. *A* PPT unification problem *$S = \{\dots, s_i =^? t_i, \dots\}$ is a system of equations where s_i and t_i are PPT functions. We say that a substitution program Θ* unifies *S with probability p_η if $P(\widehat{\Theta}_\eta \widehat{s_i}_\eta = \widehat{\Theta}_\eta \widehat{t_i}_\eta) = p_\eta$.*

For example, let $S = \{x =^? c\}$, be a unification problem, where c_η is a function that returns a uniformly random string of bets of length η. Let d_η be a function that first samples c_η and then performs one of two actions. With probability $1/2$ it returns $\widehat{c}_\eta$; otherwise it ignores $\widehat{c}_\eta$ and returns 0^η. Thus $\widehat{\Theta x}_\eta = \widehat{d}_\eta$ is a solution to S with probability $p_\eta = 2^{-1} + 2^{-\eta-1}$. That is, it is the sum of the probability 2^{-1} that d_η returns $\widehat{c}_\eta$ and the probability $2^{-\eta-1}$ that $\widehat{c}_\eta$ is ignored but is equal to 0.

We will often need to convert unification problems to equivalent ones. We define equivalence in the PPT model below.

Definition 6. *Let S and S' be two PPT unification problems. We say that they are* equivalent *if they have the same free variables and, for all substitution programs Θ with domain $FVar(S)$, $\widehat{\Theta}_\eta$ is a unifier of $\widehat{S}_\eta$ if and only if $\widehat{(\Theta)}_\eta$ is a unifier of $\widehat{S'}_\eta$.*

Finally, we note that in this section we have introduced different notations to describe the different models. This us useful when we need to distinguish between them, but it can become cumbersome in the cases in which we prove the same result for two or more of the theories in the same theorem. Thus, when we can avoid confusion, we will use the symbolic notation and simply refer to the relevant model (symbolic, computational algebra, or PPT).

5 Computational Soundness and Completeness Results for Theories of Interest

In this section we consider theories for block ciphers and exclusive-or, and prove relevant results on soundness and completeness.

We make use of the following definition [5].

Definition 7. *Let $(\mathcal{T}, E)$ be a cryptographic term algebra, and (A_η) a family of computational algebras. We say $(\mathcal{T}, E)$ is*

- $=_E$-sound *if for every pair of closed terms t_1, $t_2 \in \mathcal{T}$ of the same sort, $t_1 =_E t_2$ implies that $P[e_1, e_2 \xleftarrow{R} [\![t_1, t_2]\!]_{A_\eta} : e_1 =_{A_\eta} e_2]$ is overwhelming.*
- $=_E$-faithful *if for every pair of closed terms t_1, $t_2 \in \mathcal{T}$ of the same sort, $t_1 \neq_E t_2$ implies that $P[e_1, e_2 \xleftarrow{R} [\![t_1, t_2]\!]_{A_\eta} : e_1 =_{A_\eta} e_2]$ is negligible.*

5.1 Block Ciphers

The symbolic model in this case consists consists of a free term algebra $(T_{\Sigma\cup\{f\}}, \emptyset)$, where all symbols and their arguments belong to a single sort block and f is a unary function symbol.

In the computational model, we let $f = F_k$ where F is a strong pseudorandom permutation, as defined in [16].

Definition 8. *Let F be a function $\{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$, and denote the function $F(k, \_)$ as F_k. Let $k \leftarrow \{0,1\}^\eta$ be chosen uniformly at random, and G be chosen uniformly at random from the set of permutations on η-bit strings. We say that F is a* strong pseudorandom permutation *if F_k is a permutation of $\{0,1\}^\eta$, and for all probabilistic polynomial-time adversaries $\mathcal{A}$, the expression*

$$|Pr[\mathcal{A}^{F_k(\cdot), F_k^{-1}(\cdot)}(1^\eta) - Pr[\mathcal{A}^{G(\cdot), G^{-1}(\cdot)}(1^\eta) = 1]|$$

is a negligible function of η.

Suppose that a computational algebra semantics is defined for $(\mathcal{T}_\Sigma(\mathcal{X}), E)$. Then the computational algebra semantics for $(\mathcal{T}_{\Sigma\cup\{f\}}(\mathcal{X}), E)$ is defined by mapping f to F_k for some k chosen uniformly at random.

Lemma 1 *Suppose that $(\mathcal{T}_\Sigma, E)$ is $=_E$ sound (respectively, $=$ E-faithful). Then so is $(T_{\Sigma,f}, E)$.*

Proof. The proof follows from the fact that $f(t) =_E f(s)$ if and only if $t =_E s$.

5.2 Exclusive-Or

In this section we consider the $\oplus$ algebra $(T_\Sigma, E_\oplus)$, where Σ consists of the symbols $\oplus/2, 0/0$ and $E = R_\oplus \uplus AC$. There is a single sort, block. In the computational algebra A_η, $\oplus$ is mapped to the bitwise exclusive-or operation on η-length bitstrings, and 0 is mapped to the bitstring 0^η.

Lemma 2 *The $\oplus$ algebra is $=_{E_\oplus}$-faithful and $=_{E_\oplus}$-sound.*

Proof. In Appendix.

6 Mode of Encryption (MOE) Programs

In this section we formalize modes of encryption as programs executed by an adversary and an oracle and define security properties of MOE programs.

6.1 Definition of MOE Program

A MOE program is a program jointly executed by an adversary and an oracle in which the adversary sends messages and the oracle uses them to instantiate variables in MOE terms that represent encrypted blocks. In the symbolic and computational algebra models, the messages sent by the adversary are terms in $\mathcal{T}_\Sigma(\mathcal{X})$ that can be computed from terms the adversary has already received and from random names generated by the adversary. In the PPT case the adversary's messages can be any term computable by a probabilistic polynomial time Turing machine from the messages the adversary has already received.

We give definitions of MOE programs in the symbolic model. Definitions in the computational algebra and PPT models can be straightforwardly derived from these. We describe the oracle's program as follows. The oracle can perform two types of actions: it can receive a message from the adversary, or it can compute and send a MOE term.

Definition 9. *The* MOE program actions *engaged in by the oracle are the following, where the random name c is a session identifier, identifying blocks that belong to the same message.*

1. *Rcv_START(c). The oracle receives a START message from the adversary, indicating that it is going to start sending a sequence of plaintext blocks.*
2. *Rcv_STOP(c). The oracle receives a STOP message from the adversary, indicating that it has stopped sending plaintext blocks.*
3. *Rcv_BLOCK(c, x). The oracle receives a block to be encrypted from the adversary. This is represented by a logical variable x, which will be instantiated to the message sent by the adversary as the program executes.*
4. *Send(c, m). The oracle sends a MOE term $m \in \mathcal{T}_\Sigma(\mathcal{X})$ whose unbound variables are variables x such that Rcv_BLOCK(c, x) occurs before Send(c, m).*

The oracle's choice of which it does and the MOE term it sends is entirely determined by the type of messages it has received and the value of c. We call the specification of the point at which $\mathrm{Send}(c, m)$ occurs the *schedule* of the program, denoted by R. We have already described two schedules: the message-wise schedule, in which ciphertext blocks are not sent until all plaintext blocks have been received (that is, after the $\mathrm{Rcv\_STOP}(c)$ action), and the blockwise schedule, in which a ciphertext block as sent as soon as it is computed.

The oracle program can also return more than one block per plaintext block from the adversary. However, we assume that the oracle, like the adversary, can only preform PPT computations. Thus, the number of blocks returned by the oracle, and the number of computations required to compute those blocks, should be bounded by a polynomial function of the security parameter η.

A MOE program is executed by the adversary as follows. The adversary first sends a START(c) message. At any later point, after that adversary has received all messages expected from the oracle, it can send another message, which can be another START(c') using a fresh value c', or, if it has previously sent START(c) but has not sent STOP(c), it can send a BLOCK(c, x) message or a STOP(c) message. Thus the adversary is able to interleave the encryption of several different messages by interleaving messages labeled with different unique identifiers. The adversary can also terminate the program at any time.

We now define MOE frames.

Definition 10. *Let G be a trace of a MOE program. We let G' be the seqence obtained by removing all START and STOP messages. We define the* MOE frame *associated to G, or ϕ_G, to be the frame defined by $\phi X_i \mapsto t$ if the i-th action of G' is Rcv_BLOCK(t) or Send(t).*

Example 1. We introduce a running example. In CBC mode the MOE trace of the encryption of a two-block message in the presence of a a blockwise-adaptive adversary that sends its first message after receiving the IV is

$$\nu c \nu r.[\text{Rcv\_START}(c),\ \text{Send}(c, r), \text{Rcv\_BLOCK}(c, x_1),$$
$$\text{Send}(c, f(x_1 \oplus r)), \text{Rcv\_BLOCK}(c, x_2),\quad \text{Send}(c, f(x_2 \oplus f(x_1 \oplus r))), \text{Rcv\_STOP}(c)]$$

The corresponding frame is $\nu r[r, x_1, f(x_1 \oplus \nu r), x_2, f(x_2 \oplus f(x_1 \oplus r))]$.

We now describe the term algebra we will use.

Definition 11. *Let $\Sigma = \{0/0, \oplus/2, f\}$ where $\{f\}$ stands for block cipher evaluation with F_k, and let $E = R_\oplus \uplus AC$. We call $(\mathcal{T}_\Sigma(\mathcal{X}), E)$ the* $\oplus$-Mode of Encryption ($\oplus$-MOE) algebra*. We call MOE programs defined in the $\oplus$-MOE algebra* $\oplus$-MOE programs*.*

Example 2. We give two examples of modes of encryption defined using the $\oplus$-MOE algebra. Let x_i be a variable standing for the i'th block of plaintext, and let C_i stand for the i'th block of ciphertext. Then *Cipher Block Chaining (CBC)* is defined by $C_0 = \nu.r$ and $C_i = f(C_{i-1} \oplus x_i)$ for $i > 0$, and *Cipher Feedback Mode (CFB)* is defined by $C_0 = \nu.r$ and $C_i = f(C_{i-1}) \oplus x_i$ for $i > 0$.

Proposition 1. *The $\oplus$-MOE algebra is is $=_E$-faithful and $=_E$-sound.*

Proof. From the $=_E$-faithfulness and $=_E$ soundness of the exclusive-or theory, and Lemma 1.

6.2 Cryptographic Security Definitions

In the following, we make the assumption that the function F is a strong pseudorandom function, and let $f = F_k$ for randomly chosen k. We assume that plaintext blocks and ciphertext blocks and keys are of length η, where η is the security parameter.

Definition 12. *A MOE program is* admissable *if*

1. *The oracle computes one new f-rooted term or one new random name νr for each new block of ciphertext. We call such a term appearing in a ciphertext block the* fresh term *of that block.*
2. *Each block computed by the oracle is of the form h or $h \oplus t$, where h is the fresh term, the adversary is able to compute t, and h is not used in the computation of t.*

Example 3. We note that if each block of ciphertext produced by a MOE program Π is either h_i or $h_i \oplus P_i$ where h_i is a fresh term and P_i is a plaintext block, then Π is admissable. Thus CBC and CFB are both admissable modes of encryption expressible in the $\oplus$-MOE algebra.

Some other admissable modes of encryption in the $\oplus$-MOE algebra are OFB [11], PCBC [11], and infinite garble extension [9].

We now define a notion of R-chosen-plaintext adaptive security for a MOE program with policy R that is analogous to the Rogaway's IND$-CPA security [21]: but uses an adversarial model similar to Joux's et al.'s BACPA security [15].

Definition 13. *We define a* MOE game *over a MOE program Π with schedule R, security parameter η, and strong pseudorandom function F as follows. The environment chooses k uniformly at random, and lets $f = F_k$¿*

The adversary interacts with one of two oracles, $\Pi(\cdot)$ and $\$(\cdot)$ by submitting START, STOP and BLOCK messages and receiving responses from the oracle (if expected) as if interacting with the MOE program Π. The $\Pi(\cdot)$ oracle responds according to the MOE program Π, while the $\$(\cdot)$ oracle returns randomly generated streams of the same length and returned at the same time as those returned by the $\Pi(\cdot)$ oracle.

At any point the adversary may halt the game guess which oracle it is interacting with. The output of the game is 1 if it has guessed correctly. We say that Π is IND$-RCPA *secure if the following is negligible in η for all probabilistic polynomial time algorithms $\mathcal{A}$:*

$$\left| Pr_{k \leftarrow \{0,1\}^{n_\eta}} \left[\mathcal{A}^{\Pi(\cdot)} = 1 \right] - Pr \left[\mathcal{A}^{\$(\cdot)} = 1 \right] \right|$$

We next define a security condition equivalent to IND$-RCPA that is defined in terms of a unification problem for admissable MOE programs.

Definition 14. *Consider a game identical to that in Definition 13, except that its output is 1 if any two blocks returned by the oracle are have identical fresh terms. We say that Π is* IND$-R-UNIF *secure if the following is negligible in η for all probabilistic polynomial time algorithms $\mathcal{A}'$:*

$$\left| Pr_{f \leftarrow \mathcal{F}_{n_e ta}} \left[{\mathcal{A}'}^{\Pi(\cdot)} = 1 \right] \right|$$

Proposition 2. *An admissible MOE program Π is* IND$-RCPA *secure if and only if it* IND$-R-UNIF *secure.*

Proof. See the Appendix.

This result makes comparing the real and symbolic game much easier than is normally the case. Instead of trying to prove indistinguishability directly, we need only determine whether or not the adversary can compute a non-negligible unifier of two terms sent by the oracle. This means we can avoid such issues as soundness and completeness of properties such as static equivalence [1], and consider only $=_E$ soundness and $=_E$-faithfulness.

We now define computable substitutions on MOE frames.

Definition 15. *Let ϕ be a MOE frame from a MOE game $\mathcal{G}$. We say that a PPT (respectively, computational algebra) substitution program Θ is* computable *where x_i is the variable standing for the i'the plaintext block sent by the adversary, can be computed by the adversary using information it has received before sending x_i. We say that a symbolic unifier Θ is a* computable symbolic unification program *if for each such x_i, Θx_i can be computed using a combination of 1) random names that do not appear in the MOE frame, 2) messages that were sent by the oracle previously to the sending of x_i, 3) applications of any operator deemed to be computable by the adversary, in particular $\oplus$.*

Example 4. Consider the frame from Example 1: $\nu r[r, x_1, f(x_1 \oplus r), x_2, f(x_2 \oplus f(x_1 \oplus r))]$. We note that $\Theta_1 : x_1 \mapsto r, x_2 \mapsto f(x_1 \oplus r)$ is computable, and thus cipher block chaining is not IND$-RCPA-secure when R is the blockwise schedule. But $\Theta_2 : x_1 \mapsto f(x_1 \oplus r)$ is not computable, since the adversary has not received $f(x_1 \oplus r)$ by the time it sends x_1.

Lemma 3 *An $\oplus$-MOE frame substitution is computable in the computational algebra model if and only if it is computable in the symbolic model.*

Proof. This is a direct consequence of the $=_E$-soundness and faithfulness of the $\oplus$ algebra and the definition of computability in both models.

7 Computing Unifiers for $\oplus$-MOE Programs

The unification programs we produce must be computable by the adversary. This means it must be possible to produce a unifier without the use of f-rooted terms or oracle-produced random names that it has not yet seen. In order to determine whether this is possible we adapt the Baader-Schulz unification procedure [2]. This is a procedure for solving a unification problem for a combination of disjoint theories E_1 and E_2 for which unification algorithms are already known. In the first part of the algorithm, the original problem is divided into an E_1 problem and an E_2 problem. There is a unifier for the original problem if and only there are solutions for the E_1 and E_2 problems that satisfy separate constraints that

obey certain consistency criteria. These constraints are closely related to the constraints on our unification programs.

Our strategy will be as follows. We wish to find a symbolic criterion for the existence of a computable unifier arising from an $\oplus$-MOE frame. We use a variant of the Baader-Schulz procedure to show that the unification problem can be used to derive two unification problems, one for the adversary, and one for the oracle, where the oracle problem is in solved form, and the adversarial problem is defined over the exclusive-or theory. We show that the PPT unifier must satisfy a condition similar to the *linear constant restriction* used in Baader-Schulz. We then use this result to derive a symbolic condition on the unification problem that is asymptotically sound and complete with respect to IND$-R-UNIF security.

7.1 Computing Equations for $\oplus$-MOE Unification

Let $S(s_1, s_2) = \{s_1 =^? s_2\}$ (denoted as S when we can avoid confusion) where s_1 and s_2 are sent by the oracle in a MOE frame ϕ. We begin by converting S into a *derived problem* $S_2(s_1, s_2)$ so that each equation in S_2 is either a pure equation in the exclusive-or theory or describes a computation done by the oracle. This is done by replacing terms with new variables. In the following, we use the notation y^t to denote a variable that replaces a term t.

For any term $t \in \mathcal{T}_\Sigma(\mathcal{X})$, we define pure(t) recursively as follows:

$$\text{pure}(x) = x; \text{pure}(r) = z^r; \text{pure}(f(t)) = z^{f(t)}; \text{pure}(g \oplus t) = \text{pure}(g) \oplus \text{pure}(t)$$

where x is a variable, r is a random name, and g is a variable, random name, or f-rooted term. We now define S_2 as follows:

1. If $t = f(t')$ is an f-rooted subterm of s_1 or s_2, add $z^{f(t')} =^? f(z^{t'})$ to S_2 and also add $z^{t'} =^? \text{pure}(t')$ to S_2 if t' is $\oplus$-rooted.
2. If r is a random name, add $z^r = r$ to S_2.
3. Add $z^{s_1} =^? z^{s_1}$ to S_2.

Example 5. Consider the $\oplus$-MOE frame $\nu r_1.[r_1, x_1, f(x_1 \oplus r_1), x_2, f(x_2 \oplus f(x_1 \oplus r_1))]$, and the problem and suppose we want to see if the adversary can solve $S(f(x_1 \oplus r_1), f(x_2 \oplus f(x_1 \oplus r_1)))$. We convert it to the problem S_2 of the form

$$\begin{array}{ll} y^{r_1} =^? r_1; & w^{x_1 \oplus r_1} =^? x_1 \oplus y^{r_1}; \\ y^{f(x_1 \oplus r_1)} =^? f(w^{f(x_1 \oplus r_1)}) & w^{x_2 \oplus f(x_1 \oplus r_1)} =^? x_2 \oplus y^{f(x_1 \oplus r_1)} \\ y^{f(x_1 \oplus r_1)} =^? y^{f(x_2 \oplus f(x_1 \oplus r_1))} & \end{array}$$

We characterize the variables of S_2 as follows. If x is a variable that originally appeared in S, then it stands for a term created by the adversary, and we call it an *adversarial variable*. If $y = y^{f(t)}$ or y^c, where c is a random name, we call y an *f-variable*. If $w = w^{t_1 \oplus \ldots \oplus t_n}$ where $n > 1$, we call w an $\oplus$ variable. We will use the convention that the letter x stands for an adversarial variable, y stands for an f-variable, w stands for an $\oplus$-variable, and z may stand for any variable.

7.2 Introducing Partitions and Splitting S_2

Following the Baader-Schultz approach, we now introduce partitions of variables, where, if Θ is a substitution program with domain D, then $P \subset D \times D$ is the *partition enforced by* Θ if $(z_1, z_2) \in P$ if and only if $\Theta z_1 = \Theta z_2$. The idea is that each symbolic unifier must enforce *some* partition of the variables, even if it is only the empty one. Thus, if we can show if that no unifier exists for *any* partition, then no unifier exists at all. We now show how to introduce a partition of $FVar(S_2)$ and how it can be used to split S_2 into two problems, one for the adversary, and one for the oracle.

We define the oracle's problem first.

Definition 16. *Given a problem $S(s_1 =^? s_2)$ arising from an $\oplus$-MOE frame ϕ, and the derived problem S_2, we define the* oracle's substitution program arising from S *to be $S_2 \setminus \{y_1^s =^? y_2^s\}$.*

We now define the adversary's problem.

Definition 17. *Suppose that P is a partition of the variables of S_2. We let I_P to be the set of equations $z^{t_1} =^? z^{t_2}$ such that $(z^{t_1}, z^{t_2}) \in P$. We let $S_{\mathcal{A},P}$ be the set of all equations $\downarrow_{R_\oplus, AC}(\overbrace{z_i} \oplus \overbrace{z_j}) =^? 0$ such that $z_i =^? z_j \in I_P$, where $\overbrace{z} = z$ if z is an adversarial or f-variable, and $\overbrace{z} = pure(t)$ if $z = w^t$ where w^t is an $\oplus$-variable.*

Lemma 4 *$S_{\mathcal{A},P} \cup S_{\mathcal{O}}$ is equivalent to $I_P \cup S_{\mathcal{O}}$. Moreover, $FVar(S_{\mathcal{A},P})$ contains only adversarial and f-variables, and $Sym(S_{\mathcal{A},P}) \subseteq \{\oplus, 0\}$.*

Proof. The first statement follows from the fact that for any t_1, t_2 such that $z^{t_1} =^? z^{t_2} \in I_P$, the equation $(\overbrace{z^{t_1}} \oplus \overbrace{z^{t_2}}) =^? 0$ may be transformed into $z^{t_1} \oplus z^{t_2}$ and vice versa by use of $S_{\mathcal{O}}$. The rest follows from the construction of $S_{\mathcal{A},P}$.

Example 6. Consider the derived problem S_2 obtained in Example 5. We dentify $w^{x_1 \oplus r_1}$ and $w^{x_2 \oplus f(x_1 \oplus r_1)}$, which is indeed necessary in order to solve $y^{f(x_1 \oplus r_1)} =^? y^{f(x_2 \oplus f(x_1 \oplus r_1))}$, thus obtaining $I_P = \{y^{f(x_1 \oplus r_1)} =^? y^{f(x_2 \oplus f(x_1 \oplus r_1))}, w^{x_1 \oplus r_1} =^? w^{x_2 \oplus f(x_1 \oplus r_1)}\}$. We have $S_{\mathcal{O}} =$

$$
\begin{array}{rr}
y^{r_1} =^? r_1 & w^{x_1 \oplus r_1} =^? x_1 \oplus y^{r_1} \\
y^{f(x_1 \oplus r_1)} =^? f(w^{f(x_1 \oplus r_1)}) & w^{x_2 \oplus f(x_1 \oplus r_1)} =^? x_2 \oplus y^{f(x_1 \oplus r_1)} \\
y^{f(x_1 \oplus r_1)} =^? y^{f(x_2 \oplus f(x_1 \oplus r_1)} & x_1 \oplus y^{r_1} =^? x_2 \oplus y^{f(x_2 \oplus f(x_1 \oplus r_1))}
\end{array}
$$

We then use I_P and $S_{\mathcal{O}}$ to obtain $S_{\mathcal{A},P} =$

$$\{y^{f(x_1 \oplus r_1)} =^? y^{f(x_2 \oplus f(x_1 \oplus r_1)}, x_1 \oplus y^{r_1} =^? x_2 \oplus y^{f(x_2 \oplus f(x_1 \oplus r_1))}\}$$

7.3 Computing the Partial Order Restriction

Any computable substitution program on the variables in S_2 is constrained by the properties of the oracle and the adversary. First, the program is constrained by the oracle's program, which is fixed by the definition of the frame. Secondly, the only f-rooted terms and random names computed by the oracle that the adversary can use are those it has previously seen. We also assume that it the adversary sends x_i before x_j, then for any σ computed by the adversary, σx_i is computed before σx_j. This is actually not strictly true. As long as it has not sent either term yet, it could even compute σx_j first and use this to compute σx_i, However, this is equivalent to the adversary computing some program Q first and then using the output of Q to compute first σx_i and then σx_j. Thus, without loss of generality we may assume that σx_i is computed before σx_j.

We begin by defining the two partial orders below.

Definition 18. *We define the relation $z_1 <_{\mathcal{O}} z_2$ on the variables of S_2 by 1) $x <_{\mathcal{O}} z^t$ if the adversarial variable x is a subterm of t, and 2) for any two oracle variables z^{t_1} and z^{t_1}, $z^{t_1} <_{\mathcal{O}} z^{t_2}$ if t_1 is a subterm of t_2.*

We define the relation $<_{\mathcal{A}}$ on variables of S_2 by 1) $y^t <_{\mathcal{A}} x$ if x is an adversarial variable, and t is the fresh term in a term sent by the oracle to the adversary before x is sent by the adversary to the oracle, and 2) for any two adversarial variables x_i and x_j, $x_i <_{\mathcal{A}} x_j$ if and only if x_i is sent by the adversary to the oracle before x_j.

Example 7. Consider the $\oplus$-MOE frame from Example 1:

$$[r_1, x_1, f(x_1 \oplus r_1), x_2, f(x_2 \oplus f(x_1 \oplus r_1))]$$

with derived problem S_2 as in Example 5. Then for $S_{\mathcal{A}}$ we have

$$y^{r_1} <_{\mathcal{A}} x_1,\ y^{r_1} <_{\mathcal{A}} x_2,\ y^{f(x_1 \oplus r_1)} <_{\mathcal{A}} x_2,\ x_1 <_{\mathcal{A}} x_2$$

and for $S_{\mathcal{O}}$, $<_{\mathcal{O}}$ is the transitive closure of the following:

$$\begin{aligned} y^{r_1} &<_{\mathcal{O}} w^{x_1 \oplus r_1} & x_1 &<_{\mathcal{O}} w^{x_1 \oplus r_1} \\ x_2 &<_{\mathcal{O}} w^{x_2 \oplus f(x_1 \oplus r_1)} & w^{x_1 \oplus r_1} &<_{\mathcal{O}} y^{f(x_1 \oplus r_1)} \\ y^{f(x_1 \oplus r_1)} &<_{\mathcal{O}} w^{x_2 \oplus f(x_1 \oplus r_1)} & w^{x_2 \oplus f(x_1 \oplus r_1)} &<_{\mathcal{O}} y^{f(x_2 \oplus f(x_1 \oplus r_1)} \end{aligned}$$

Our plan will be to solve $S_{\mathcal{A},P}$, and, if a unifier can be found that satisfies the constraint imposed by $<_{\mathcal{A},P}$, combine it with $S_{\mathcal{O}}$ to obtain a unifier for S_2. However, we will need to guarantee that when these two separate unifiers are combined, the result satisfies both sets of constraints. We do this by showing that the combination of $<_{\mathcal{A},P}$ and $<_{\mathcal{O}}$ is itself a partial order on the variables of S_2, so that the combined unifier contains no cycles.

7.4 Computing a Solved Form for $S_{\mathcal{A},P}$

Here we describe a canonical form for $S_{\mathcal{A},P}$, and show how to compute it. This is used to determine if an $\oplus$-MOE unification problem has a solution or not.

Definition 19. *Let S be an $\oplus$-MOE unification problem with derived problem S_2, and let P be a partition of $FVar(S_2)$.For each y-variable $y^{t_i} \in S_2$, we define $E^{\mathcal{O},P}(y^{t_i})$ to be the set of all f-variables y^{t_j} such that $(y^{t_i}, y^{t_j}) \in P$. We then define a* choosing function π *on the f-variables of S_2 to be*

1. *If $E^{\mathcal{O},P}(y^{t_i})$ contains an f-variable y^{t_j} where t_j is the fresh term in a term sent by the oracle to the adversary, let πy^{t_i} be the fresh term in the earliest term sent by the oracle such that its fresh term is in $E^{\mathcal{O},P}(y^{t_i})$.*
2. *Else, pick an arbitrary $y^{t_j} \in E^{\mathcal{O},P}(y^{t_i})$ to be $\pi(y)$ for all $y \in E^{\mathcal{O},P}(y^{t_i})$.*

We then let $\pi S_{\mathcal{A},P}$ be the unification problem obtained by replacing each variable y^{t_i} appearing in $S_{\mathcal{A},P}$ with πy^{t_i}

We note that $\pi S_{\mathcal{A},P} \cup S_{\mathcal{O}}$ is equivalent to $S_{\mathcal{A},P} \cup S_{\mathcal{O}}$.

Example 8. Consider the problem $S_{\mathcal{A},P}$ from Example 7. In this case we have $E^{\mathcal{O},P}(y^{f(x_1 \oplus r_1)}) = \{y^{f(x_1 \oplus r_1)}, y^{x_2, f(x_1 \oplus r_1)}\}$, and $E^{\mathcal{O},P}(y^{r_1}) = \{y^{r_1}\}$. By default, $\pi y^{r_1} = y^{r_1}$. For $E^{\mathcal{O},P}(y^{f(x_1 \oplus r_1)})$ both variables represent terms sent by the oracle to the adversary, but the term $f(x_1 \oplus r_1)$ is sent first, so $\pi y^{f(x_1 \oplus r_1)} = \pi y^{f(x_2 \oplus f(x_1 \oplus r_1))} = y^{f(x_1 \oplus r_1)}$. The unification problem $\pi S_{\mathcal{A},P}$ consists of the equations $x_1 \oplus y^{r_1} =^? x_2 \oplus y^{f(x_1 \oplus r_1)}$, and $y^{f(x_1 \oplus y^{r_1})} =^? y^{f(x_2 \oplus y^{f(x_1 \oplus r_1)})}$, so $\pi S_{\mathcal{A},P} = \{x_1 \oplus y^{r_1} =^? x_2 \oplus y^{f(x_1 \oplus r_1)}, 0 =^? 0\}$.

We now show how, given a partition P, and a problem $S_{\mathcal{A},P}$, we can use any choosing function π on P to transform $S_{\mathcal{A},P}$ into a solved $\oplus$ unification problem we call its $\oplus$-*MOE solved form*. This will be used to characterize the cases in which the original problem $S(s_1, s_2)$ is or is not solvable by the adversary.

Definition 20. *Given $\pi S_{\mathcal{A},P}$ where π is a choosing function, we define the* $\oplus$-MOE solved form *of $\mathcal{M}(\pi S_{\mathcal{A},P})$ as follows. First, we let ${\pi S_{\mathcal{A},P}}'$ be the system of equations obtained by first writing each equation of $\pi S_{\mathcal{A},P}$ in the form*

$$(\sum_{j=0}^{k-1} \oplus \alpha_{i,j} x_{k-j}) =^? (\sum_{j=1}^{m} \oplus \beta_{i,j} y^{t_j})$$

where $x_1, \ldots, x_m$ are the adversarial variables in $S_{\mathcal{A},P}$ in the order in which they are sent by the adversary, and $y^{t_1} \ldots, y^{t_m}$ are the y-variables of $\pi S_{\mathcal{A},P}$, and then converting the problem to reduced row echelon form in the x-variables. Thus each equation in ${\pi S_{\mathcal{A},P}}'$ is in one of the following forms:

1. *$0 =^? 0$;*
2. *For each adversarial term x_i, at most one equation of the form $x_i \oplus (\sum_{j=1}^{i-1} \oplus \alpha'_j x_{i-j}) =^? \sum_{j=1}^{n} \oplus \beta'_{i,j} y^{t_j}$ or ;*

3. $\sum_{j=1}^{n} \oplus \beta'_{i,j} y^{t_j} =^? 0$ *where at least one* $\beta'_{i,j} = 1$.

In the $\oplus$*-MOE solved form* $\mathcal{M}(\pi S_{\mathcal{A},P})$*, each equation containing an adversarial variable is to the form* $x_i =^? (\sum_{j=1}^{i-1} \oplus \alpha'_j x_{i-j}) \oplus (\sum_{j=1}^{n} \oplus \beta'_{i,j} y_j)$*, and the tautological equations* $0 =^? 0$ *are removed.*

Example 9. Consider the problem $x_1 \oplus y^{r_1} =^? x_2 \oplus y^{f(x_1 \oplus r_1)}$ from Example 8. The MOE solved form of this problem is $x_2 =^? x_1 \oplus y^{r_1} \oplus y^{f(x_1 \oplus r_1)}$

Definition 21. *We say that a problem* $\pi S_{\mathcal{A},P}$ *is* well-ordered *if its* $\oplus$*-MOE solved form consists of equations of the form* $x_i =^? (\sum_{j=1}^{i-1} \oplus \alpha'_j x_{i-j}) \oplus (\sum_{j=1}^{n} \oplus \beta'_{i,j} y_j) \in \mathcal{M}(\pi S_{\mathcal{A},P})$ *such that* $\beta'_{i,j} \neq 0$ *implies that* $y_j <_{\mathcal{A},P} x_i$*. We say that* $S_{\mathcal{A},P}$ *is well-ordered if there is a choosing function* π *such that* $\pi S_{\mathcal{A},P}$ *is well-ordered.*

Lemma 5 *If there exists a choosing function* π *on* P *such that* $\pi S_{\mathcal{A},P}$ *is well-ordered, then so is* $\pi' S_{\mathcal{A},P}$ *for any choosing function* π' *on* P*.*

Proof. This follows directly from the fact that choosing functions can differ only for f-variables y such that there is no member y^t of y's equivalence class such that t is sent by the oracle to the adversary..

Lemma 5 means that we do not need to check all choosing functions π to see if there is a well-ordered $\pi S_{\mathcal{A},P}$; it is enough to pick an arbitrary one. Thus we may assume that for each partition P, there is a single choosing function π_P.

We now show how to construct a computable unifier for an $\oplus$-MOE problem S, given that there is a partition P of the variables of S_2 such $S_{\mathcal{A},P}$ is well-ordered.

Lemma 6 *The order* $<_{\mathcal{A},\mathcal{O}}$ *on the variables of* S_2 *defined as the transitive closure of* $<_{\mathcal{A}} \cup <_{\mathcal{O}})$ *is suborder of a total order* $<_{\ell}$*.*

Proof. In the Appendix.

Definition 22. *Given an* $\oplus$*-MOE unification problem* S *with derived problem* S_2 *with partition* P *of its variables such that* $S_{\mathcal{A},P}$ *is well-ordered, compute* Ξ *on each non-adversarial variable* z^t *ascending in the* $<_{\ell}$*-order, where* $\Xi z^t = t$*. Then compute* $\Theta_{S,P}$ *on each adversarial variable* x *as follows, ascending in the* ℓ*-order.*

1. *If* $z = x_i$*, where* $x_i =^? (\sum_{j=1}^{i-1} \oplus \alpha'_j x_{i-j}) \oplus (\sum_{j=1}^{n} \oplus \beta'_{i,j} \Xi y^{t_j}) \in \mathcal{M}(\pi_P S_{\mathcal{A},P})$*, then* $\Theta_{S,P} z = ((\sum_{j=1}^{i-1} \oplus \alpha'_j x_{i-j}) \oplus (\sum_{j=1}^{n} \oplus \beta'_{i,j} \Xi y^{t_j}))$*;*
2. *Else, if* $z = x_i$*, where there is no* $x_i =^? (\sum_{j=1}^{i-1} \oplus \alpha'_j x_{i-j}) \oplus \sum_{j=1}^{n} \oplus \beta'_{i,j} y^{t_j} \in \mathcal{M}(\pi_P S_{\mathcal{A},P})$*, then* $\Theta_{S,P} x_i = 0$*.*

Proposition 3. *Suppose that for a given* $\oplus$*-MOE problem* S *and partition* P*, the problem* $S_{\mathcal{A},P}$ *is well-ordered. Then* $\Theta_{S,P}$ *is a computable solution program in all three models.*

Proof. In the Appendix.

8 Completeness of the Well-Orderedness Criterion

We are not interested only in the behavior of an adversary with respect to a given frame, unification problem, and partition, but *all* computable $\oplus$-MOE frames, problems, and problems arising from a given $\oplus$-MOE program. Given that the number of partitions of a set asymptotically approaches an exponential function of its size (see for example [8]), we need to rule out the case in which the probability of the adversary succeeding overall is non-negligible, but the probability of its succeeding with respect to any given frame is negligible. For this we make use of a result of Bellare's [6] on negligible functions, which says that any (infinite) set of negligible functions is bounded by a negligible function.

We first prove completeness for a single $\oplus$-MOE frame.

Proposition 4. *In the PPT and computational algebra models, given an admissable $\oplus$-MOE program G, let ϕ be an $\oplus$-MOE frame and let Θ be a computable substitution program whose domain is $FVar(\phi)$. Suppose that the probability that there is an $\oplus$-MOE unification problem $S(s_1, s_2)$ derived from ϕ such that $\widehat{\Theta s_1}_\eta = \widehat{\Theta s_1}_\eta$ is non-negligible. Then there is an $\oplus$-MOE unification problem $S(s'_1, s'_2)$ derived from ϕ and a partition P of the variables of $S_2(s'_1, s'_2)$ such that $S_{\mathcal{A},P}(s_1, s_2)$ is well-ordered. Likewise, in the symbolic model, if Θ is a computable unifier of $S(s_1, s_2)$, then there is a partition P of the variables of $S_2(s_1, s_2)$ such that $S_{\mathcal{A},P}(s_1, s_2)$ is well-ordered.*

Proof. In the Appendix.

We now give the full completeness result.

Theorem 7. *An admissable $\oplus$-MOE program is* IND$-RCPA*-secure in all three models and only if there is no $\oplus$-MOE frame ϕ with unification problem $S(s_1 =^? s_2)$, with a partition P of the variables of $S_2(s_1, s_2)$ such that $S(s_1, s_2)_{\mathcal{A},P}$ is well-ordered.*

Proof. In the Appendix.

Example 10. We now show that two modes of encryption are IND$-RCPA secure: CFB with the blockwise schedule, and CBC with a delayed schedule in which the i'th cipher text is sent after the $i+1$st plaintext. The latter is due to Fouque et al. [12], and known as *delayed CBC* (DCBC).

In the case of CFB, let h_i and h_j where $i < j$ be the first pair of fresh terms that the adversary is able to make equal with non-negligible probability. We consider the case $h_i = f(h_{i-1} \oplus x_{i-1}$ and $h_j = f(h_{j-1} \oplus x_{j-1}$. Thus the adversary needs to solve the equation $x_{j-1} =^? x_{i-1} \oplus y^{h_{j-1}} \oplus y^{h_{i-1}}$. But $x_{j-1} \not\leq y^{j-1}$, so this is only possible h_ℓ and h_j are unified by the adversary. But this means that h_ℓ and h_i are unified by the adversary, thus contradicting our assumption. In the case of DCBC, under the same assumptions, the adversary needs to solve $h_{i-1} \oplus x_i =^? h_{j_1} \oplus x_j$ before it has seen h_{j-1}. This is impossible by the similar argument to that for CFB.

:

9 Conclusion

In this paper we have identified a symbolic condition that is equivalent to a computational notion of security of a class of modes of encryption: IND$-RCPA. Here we consider some open problems and ways this work could be extended.

One problem that remains is efficient verification of the symbolic conditions. The existence of a computable unifier can be checked by finding a most general set of unifiers and checking if any has an instance convertible to a well-ordered $\oplus$-MOE normal form. This can be done by checking all the partitions that are refinements of partitions imposed by the unifier, but more efficient solutions may be possible. In addition, we note that we can model the R-UNIF game as a protocol executed between the adversary and oracle. Thus, it may be possible to apply a model-checker for cryptographic protocols to the problem, in which case the symbolic constraints are enforced automatically. In that case our result can be used as a straightforward soundness and completeness guarantee.

The problem remains of verifying our conditions for all possible $\oplus$-MOE frames. This appears to be made more tractable by the fact that for reasons of efficiency modes of encryption are defined recursively. However, the complexity of the security problems for recursively defined protocols using exclusive-or is not that well understood. In [18] Küsters and Truderung consider the decidability of the secrecy problem (related, although not identical, to our security criterion) for recursive protocols using a theory that includes encryption/decryption, cryptographic hashes, concatenation/deconcatenation, and exclusive-or. Secrecy is shown to be undecidable in the bounded session model, and only becomes decidable for $\oplus$-*linear* protocols; however such a restriction is not practical for most modes of encryption.But the undecidability proof makes extensive use of the concatenation operator, as well as the $\oplus$ and hash operators. If the concatenation operator is removed, then the decidability result might change.

We also note that these results may extend, not only to other algorithms in the Linicrypt model, but to other theories, including some that do not satisfy $=_E$-soundness and faithfulness. One example is the increment by one function inc which is used to implement the blockwise adaptive secure counter mode. The inc function, when combined with $\oplus$ is not $=_E$-sound or faithful, since $\mathsf{inc}(r) = r \oplus 1$ with probability $1/2$ for random r. Malozemoff at al. deal with this problem in [19] by restricting themselves to cases where this ambiguity can be avoided, that is a subset of terms for which the theory combining $\oplus$ and inc is E-sound and complete. Looking further, one could think of an analysis with three types of outcomes: when $=_E$-soundness and faithfulness hold, wither provably insecure or provably insecure depending on whether symbolic security holds, and unknown when $=_E$-soundness and faithfulness fail to hold.

In summary, our results open up a new connection between symbolic methods for security of cryptographic algorithms and research in unification theory. Unification theory has proved a powerful tool in the verification of cryptographic protocols, and the interaction between the two has been an impetus to developing new unification techniques for this application. We believe that the same can be true for automatic generation and verification of cryptographic algorithms.

Acknowledgments

The author would like to thank ONR Code 31 for funding this work. She also thanks Jonathan Katz and Christopher Lynch for helpful comments.

References

1. Martín Abadi and Cédric Fournet. Mobile values, new names, and secure communication. In *Conference Record of POPL 2001: The 28th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, London, UK, January 17-19, 2001*, pages 104–115, 2001.
2. Franz Baader and Klaus U. Schulz. Unification in the union of disjoint equational theories: Combining decision procedures. *J. Symb. Comput.*, 21(2):211–243, 1996.
3. Michael Backes and Birgit Pfitzmann. A cryptographically sound security proof of the needham-schroeder-lowe public-key protocol. In *FST TCS 2003: Foundations of Software Technology and Theoretical Computer Science, 23rd Conference, Mumbai, India, December 15-17, 2003, Proceedings*, pages 1–12, 2003.
4. Gregory V. Bard. Blockwise-adaptive chosen-plaintext attack and online modes of encryption. In *Cryptography and Coding, 11th IMA International Conference, Cirencester, UK, December 18-20, 2007, Proceedings*, pages 129–151, 2007.
5. Mathieu Baudet, Véronique Cortier, and Steve Kremer. Computationally sound implementations of equational theories against passive adversaries. In *Automata, Languages and Programming*, pages 652–663. Springer, 2005.
6. Mihir Bellare. A note on negligible functions. *J. Cryptology*, 15(4):271–284, 2002.
7. Mihir Bellare, Tadayoshi Kohno, and Chanathip Namprempre. Authenticated encryption in SSH: provably fixing the SSH binary packet protocol. In *Proceedings of the 9th ACM conference on Computer and communications security*, pages 1–11. ACM, 2002.
8. Daniel Berend and Tamir Tassa. Improved bounds on bell numbers and on moments of sums of random variables. *Probability and Mathematical Statistics*, 30(2):185–205, 2010.
9. Carl Campbell. Design and specification of cryptographic capabilities. *IEEE Computer Society Magazine*, 16(6):15–19, 1978.
10. Brent Carmer and Mike Rosulek. Linicrypt: A model for practical cryptography. In *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part III*, pages 416–445, 2016.
11. Morris Dworkin. Recommendation for Block Cipher Modes of Operation: Methods and Techniques. Technical Report SP 800-38A, National Institute of Standards and Technology, December 2001.
12. Pierre-Alain Fouque, Gwenaëlle Martinet, and Guillaume Poupard. Practical symmetric on-line encryption. In *Fast Software Encryption, 10th International Workshop, FSE 2003, Lund, Sweden, February 24-26, 2003, Revised Papers*, pages 362–375, 2003.
13. Martin Gagné, Pascal Lafourcade, Yassine Lakhnech, and Reihaneh Safavi-Naini. Automated proofs of block cipher modes of operation. *J. Autom. Reasoning*, 56(1):49–94, 2016.

14. Viet Tung Hoang, Jonathan Katz, and Alex J. Malozemoff. Automated analysis and synthesis of authenticated encryption schemes. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-6, 2015*, pages 84–95, 2015.
15. Antoine Joux, Gwenaëlle Martinet, and Frédéric Valette. Blockwise-adaptive attackers revisiting the (in) security of some provably secure encryption modes: CBC, GEM, IACBC. In *Annual International Cryptology Conference*, pages 17–30. Springer, 2002.
16. Jonathan Katz and Yehuda Lindell. *Introduction to modern cryptography*. CRC Press, 2014.
17. Steve Kremer and Laurent Mazaré. Adaptive soundness of static equivalence. In *Computer Security–ESORICS 2007*, pages 610–625. Springer, 2007.
18. Ralf Küsters and Tomasz Truderung. On the automatic analysis of recursive security protocols with XOR. In *STACS 2007, 24th Annual Symposium on Theoretical Aspects of Computer Science, Aachen, Germany, February 22-24, 2007, Proceedings*, pages 646–657, 2007.
19. Alex J Malozemoff, Jonathan Katz, and Matthew D Green. Automated analysis and synthesis of block-cipher modes of operation. In *Computer Security Foundations Symposium (CSF), 2014 IEEE 27th*, pages 140–152. IEEE, 2014.
20. Daniele Micciancio and Saurabh Panjwani. Adaptive security of symbolic encryption. In *Theory of Cryptography*, pages 169–187. Springer, 2005.
21. Phillip Rogaway. Nonce-based symmetric encryption. In *Fast Software Encryption, 11th International Workshop, FSE 2004, Delhi, India, February 5-7, 2004, Revised Papers*, pages 348–359, 2004.

Appendix: Proofs of Results

Proofs for Section 5

Lemma 2 *The $\oplus$ algebra is $=_{E_\oplus}$-faithful and $=_{E_\oplus}$-sound.*

Proof. Baudet at al. [5] prove this result for a slightly larger algebra that extends the $\oplus$ theory to include the symbol 1, which is mapped to the bit string $0^{\eta-1}1$ in the computational algebra. Since the larger algebra is $=_{E_\oplus}$-faithful and $=_{E_\oplus}$-sound, so is the $\oplus$ algebra.

Proofs for Section 6

Proposition 2 *An admissable MOE program Π is* IND$-RCPA *secure if and only if it* IND$-R-UNIF *secure.*

Proof. We prove the "if" direction first. Suppose Π is IND$-R-UNIF secure, that is for any MOE frame ϕ, and any closed substitution Θ to the variables of ϕ, the probability $p_{\Theta\phi}$ that at least one pair of fresh terms in $\widehat{\Theta\phi}_\eta$ being equal is negligible. If the frame is of length n, then there is at least one pair of fresh terms h_i and h_j such that $P(\widehat{\Theta h_i}_\eta = \widehat{\Theta h_l}_\eta) \geq p_\phi/n(n-1)$. Since n is bounded by some polynomial function of η, if p_ϕ is negligible, so is $P(\widehat{t_i}_\eta = \widehat{t_j}_\eta) \geq p_\phi/n(n-1)$. Thus the fresh terms are pairwise independent with overwhelming probability.

Since they are also indistinguishable from random, the sequence $(h_0, h_1, \ldots, h_n)$ is indistinguishable from random as well.

Let $(C_0, C_1, \ldots, C_n)$ be the ciphertext sequence produced by $(h_0, h_1, \ldots, h_n)$. By the assumption of admissability, either $C_i = h_i$, or $C_i = h_i \oplus t_i$, where h_i is not used in the computation of t_i. By the fact that $P(\widehat{\Theta h_{i\eta}} = \widehat{\Theta h_{j\eta}})$ is negligible for $i \neq j$ for any Θh_j, and that, also with overwhelming probability, the adversary has not yet seen $\widehat{\Theta h_{i\eta}}$ at the time of computing any plaintext blocks used to compute t_i, the probability that $\widehat{\Theta h_{j\eta}}$ is used to produce $P(\widehat{\Theta h_{i\eta}})$ is also negligible. Thus, $\{\widehat{\Theta h_{0\eta}}, \ldots, \widehat{\Theta h_{n\eta}}\}$ is indistinguishable from a uniformly random generated sequence of bits that is independent from $\{\widehat{\Theta g_{0\eta}}, \ldots, \widehat{\Theta g_{n\eta}}\}$ where $g_i = 0$ or h_i, depending on how C_i is calculated. Thus $\{\widehat{\Theta C_{0\eta}}, \ldots, \widehat{\Theta h_{C\eta}}\}$ is indistinguishable from random as well.

Conversely, suppose that there is a pair of fresh terms h_i and h_j such that $P(\widehat{\Theta h_{i\eta}} = \widehat{\Theta h_{j\eta}})$ is non-negligible. Since the adversary is able to compute g_i and g_j from C_i and C_j it will be able to see that $P(\widehat{\Theta h_{i\eta}} = \widehat{\Theta h_{j\eta}})$ and thus distinguish $\{\widehat{\Theta C_{0\eta}}, \ldots, \widehat{\Theta C_{n\eta}}\}$ from random with non-negligible probability.

Proofs for Section 7

Lemma 6 *The order $<_{\mathcal{A},\mathcal{O}}$ on the variables of S_2 is a suborder of a total order ℓ,.*

Proof. We define ℓ as follows.

Definition 23. *Given an $\oplus$-MOE frame ϕ, we define a ϕ-*position *of a term t to be i.p, where p is a position of t in ϕX_i. We put a total order $<_\ell$ on ϕ-positions, with $p <_\ell q$ if either $p = q.r$, or $p = u.n.v$, $q = u.m.w$, where u is a ϕ-position or $\wedge$, n and m are integers, v and w are positions, and $n < m$. We put a total order $<_\ell$ on the terms of ϕ, where $t <_\ell s$ if the least ϕ-position of t is less than the least ϕ-position of s. Given an $\oplus$-MOE unification problem $S(s_1, s_2)$ from ϕ, we put a total order on the variables of $S_2(s_1, s_2)$ defined by 1) $z^{t_1} <_\ell z^{t_2}$ if $t_1 <_\ell t_2$, 2) $x_i <_\ell z^t$ (resp. $x_i >_\ell z^t$) if $x_i <_\ell t$ (resp. $x_i >_\ell t$), and 3) $x_i <_\ell x_j$ if $i < j$.*

Example 11. Given the frame ϕ in Example 1, the ϕ-positions of x_1 are 2, 3.1, and 5.2.1, the ϕ-positions of r_1 are 1, 3.2, and 5.2.2, and the ϕ-position of $f(x \oplus r_1)$ is 3 and 5.2. Thus $y^{r_1} <_\ell x_1 <_\ell y^{f(x \oplus r_1)}$,

It is straightforward to check that $<_\mathcal{A}$ and $<_\mathcal{O}$ are suborders of $<_\ell$.

Proposition 3 *Suppose that for a given $\oplus$-MOE problem S and partition P, the problem $S_{\mathcal{A},P}$ is well-ordered. Then $\Theta_{S,P}$ is a computable solution program in all three models.*

Proof. Suppose that $\Theta_{S,P}x = t$. Then by construction $FVar(t)$ contains only adversarial variables x' such that $x' <_{\mathcal{A}} x$; Thus $\Theta_{S,P}$ is a well-defined substitution program in all three models.

We next note $\Theta'_{S,P} = \Theta_{S,P}\Xi$ is a unification program for $\mathcal{M}(\pi S_{\mathcal{A},P} \cup S_{\mathcal{O}})$, which is equivalent to $\pi S_{\mathcal{A},P} \cup S_{\mathcal{O}}$, which is equivalent $S_{\mathcal{A},P} \cup S_{\mathcal{O}}$, which contains S_2. Thus $\Theta'_{S,P}$ is a unifier of S_2. Moreover, $\Theta'_{S,P} = \Theta_{S,P}\Xi$, and $\Xi S_2 = S$. Thus $\Theta'_{S,P}$ is a unification program for S.

Finally, we note that, since $S_{\mathcal{A},P}$ is well-ordered, $\Theta_{S,P}$ is also computable.

Proofs for Section 8

To prove Proposition 4, we will need the following definition and lemma.

Definition 24. *Let Θ be a substitution program to the variables in a unification problem Q. Let P be a partition of the variables of Q,and let Θ be a substitution program. In the symbolic model we say that Θ* enforces *P if Θ if $\Theta z =_E \Theta z'$ if and only if $(z, z') \in P$. In the computational algebra and PPT models, we say that Θ* enforces *P* with probability *p_η if*

$$P((\wedge_{(zz')\in P}(\widehat{\Theta}_\eta z = \widehat{\Theta}_\eta z')) \wedge (\wedge_{(zz')\notin P}(\widehat{\Theta}_\eta z \neq \widehat{\Theta}_\eta z'))) = p_\eta$$

Lemma 8 *Given an $\oplus$-MOE frame ϕ with $\oplus$-MOE unification problem $S(s_1, s_2)$, suppose that Θ is a non-negligible unification program for $S_2(s_1, s_2)$ in the PPT model. Suppose furthermore that there is a partition P of the variables of $S_2(s_1, s_2)$ containing (z^{s_1}, z^{s_2}) such that $P(\widehat{\Theta}_\eta$ enforces$P$ is non-negligible. Then $S_{\mathcal{A},P}(s_1, s_2)$ is well-ordered.*

Proof. If Θ enforces P with non-negligible probability, then $\Theta\Xi$ is is a non-negligible unification program for $S_{\mathcal{A},P} \cup S_{\mathcal{O}}$, where Ξ is the unifier of $S_{\mathcal{O}}$. Suppose that $S_{\mathcal{A},P}(s_1, s_2)$ is not well-ordered. Then one of two things can occur. The first is that there is an equation in $\mathcal{M}(\pi S_{\mathcal{A},P}(s_1, s_2))$ of the form $x_\ell =^? (\sum_{i=1}^{\ell-1} \oplus\alpha_i x_i) \oplus (\sum_{i=1}^{n} \oplus\beta_i y^{f(t_i)})$ where there is at least one j such that $\beta_j \neq 0$ and x_ℓ fails to dominate $y^{(t_j)}$ in the $<_{\mathcal{A}}$ order. Thus there is no $y^{f(t)} <_{\mathcal{A}} x_\ell$ such that $P(\widehat{\Theta\Xi}_\eta y^f(t) = \widehat{\Theta\Xi}_\eta y^{f(t_j)})$ is non-negligible. Thus, if

$$\widehat{\Theta\Xi}_\eta x_\ell = (\sum_{i=1}^{\ell-1} \oplus\alpha_i \widehat{\Theta\Xi}_\eta x_i) \oplus (\sum_{i=1}^{n} \oplus\beta_i \widehat{\Theta\Xi}_\eta y^{f(t_i)})$$

with non-negligible probability, then the adversary is able, also with non-negligible probability, to compute the exclusive-or of f-rooted terms it has not yet seen, contradicting our assumption that f is a member of a strong pseudo-random function family.

The other possibility is that there is an equation $\sum_{i=1}^{n} \oplus\beta_i y^{f(t_i)} =^? 0$ such that, with non-negligible probability $\sum_{i=1}^{n} \oplus\beta_i \widehat{\Theta}_\eta y^{f(t_i)} =^? 0$ and $\widehat{\Theta\Xi}_\eta y^{f(t_i)} \neq \widehat{\Theta\Xi}_\eta y^{f(t_j)}$ whenever $\beta_i = \beta_j = 1$. But this contradicts the assumption that f is a member of a strong pseudo-random function family.

We are now ready to prove Proposition 4.

Proposition 4 *In the PPT and computational algebra models, let ϕ be an $\oplus$-MOE frame and let Θ be a computable substitution program whose domain is $FVar(\phi)$. Suppose that the probability that there is an $\oplus$-MOE unification problem $S(s_1, s_2)$ derived from ϕ such that $\widehat{\Theta s_1}_\eta = \widehat{\Theta s_1}_\eta$ is non-negligible. Then there is an $\oplus$-MOE unification problem $S(s'_1, s'_2)$ derived from ϕ and a partition P of the variables of $S_2(s'_1, s'_2)$ such that $S_{\mathcal{A},P}(s_1, s_2)$ is well-ordered. Likewise, in the symbolic model, if Θ is a computable unifier of $S(s_1, s_2)$, then there is a partition P of the variables of $S_2(s_1, s_2)$ such that $S_{\mathcal{A},P}(s_1, s_2)$.*

Proof. The proof for the symbolic model is a direct consequence of Lemma 8. For the PPT and computational algebra models, all we need to note is that the number of $\oplus$-MOE unification problems $S(s_1, s_2)$ and the number of partitions of the variables of $S_2(s_1, s_2)$ remain fixed as the security parameter η increases. Thus, there must be at least one unification problem $S(s'_1, s'_2)$ and partition P of the variables of $S(s'_1, s'_2)$ such Θ enforces P with non-negligible probability. The result then again follows from Lemma 8.

Theorem 7 *An admissable $\oplus$-MOE program is* IND$-RCPA*-secure in all three models and only if there is no $\oplus$-MOE frame ϕ with unification problem $S(s_1 =^? s_2)$, with a partition P of the variables of $S_2(s_1, s_2)$ such that $S(s_1, s_2)_{\mathcal{A},P}$ is well-ordered.*

Proof. We prove the result for the PPT model, of which the computational algebra model is a special case. The proof for the symbolic model follows from its $=_E$ soundness and $=_E$ faithfulness. The "only if" part of the proof follows from Proposition 3. To prove the "if" part, suppose that an $\oplus$-MOE game $\mathcal{G}$ is not IND$-RCPA-secure. We need to show that there is an $\oplus$-MOE frame ϕ and a computable substitution program unifying with non-negligible probability two terms s_1 and s_2 sent by the oracle in ϕ.

The probability of the adversary's winning the game when the security parameter is η is $P_\eta = \sum_\phi p(\phi, \eta) q(\widehat{\Theta[\phi]}_\eta)$, where $p(\phi, \eta)$ is the probability that the adversary executes frame ϕ when the value of the security parameter is η, and $q(\widehat{\Theta[\phi]}_\eta)$ is the probability that the substitution program $\widehat{\Theta[\phi]}_\eta$ unifies two terms sent by the oracle in ϕ. Note that for any value of η, $p_{(\phi,\eta)}$ is zero for all but a finite number of frames ϕ, so P_η is well-defined.

Suppose that for all Θ, $q_{(\Theta,\eta)}$ is a negligible function of η. According to Bellare [6], any (infinite) set of negligible functions is bounded by a negligible function, so there is a negligible function $\mathrm{neg}(\eta) > q_{(\phi,\eta)}$ for all ϕ and η. But then

$$P_\eta = \sum_\phi p_{(\phi,\eta)} q(\widehat{\Theta[\phi]}_\eta) < \mathrm{neg}(\eta)(\sum_\phi p_{(\phi,\eta)}) = \mathrm{neg}(\eta)$$

contradicting its non-negligibility. Thus, by Lemma 4, $S(s_1, s_2)_{\mathcal{A},P}$ is well-ordered for some partition P.