Some Unification Problems Arising From Symbolic Cryptography

Catherine Meadows

Center for High Assurance Computer Systems, Code 5540
Naval Research Laboratory
Washington, DC, 20375
catherine.meadows@nrl.navy.mil

Symbolic methods for the analysis of cryptographic protocols have been increasing in popularity, and they have proven to be useful for addressing a number of different cryptographic problems. Such symbolic methods often rely on unification, in particular unification modulo equational theories that arise from equational properties of the cryptographic algorithms involved. This has led to a synergistic interaction between cryptographic protocol analysis and equational unification, resulting in new unification techniques such as variant unification, new problem areas such as asymmetric unification, and a new focus on equational theories that describe the operations of cryptographic algorithms.

Security researchers have lately begun exploring the application of symbolic techniques at a lower level of abstraction: to the verification of cryptosytems themselves. One particular area of interest is the automatic generation of cryptographic algorithms, in which cryptosystems are generated using a fixed set of operations that can be combined in multiple ways. Symbolic methods can then be used to evaluate the automatically generated cryptosystems. In many cases, the symbolic criteria are shown to guarantee cryptographic security, as in [5, 3, 2]. However, we note that, the more complex the equational theory, and the more powerful the adversary, the harder it becomes to guarantee cryptographic soundness and completeness of symbolic security properties. Our own work has concentrated on theories that, while relatively simple, are widely used in cryptography, and on proving security against a powerful adversary capable of *adaptive chosen plaintext attacks*, in which an adversary can use information from previous interactions with the cryptosystem to provide new input to it.

One simple theory that is applicable to a wide class of algorithms is the combination of an Abelian group theory (of which the exclusive-or theory is a special case), and a free unary operator f, along with a finite set of free constants. The operator f stands for a keyed function where the key is not known by the adversary, e.g. a keyed hash function or a block cipher.

A *block cipher* is a cipher that, given a plaintext block of a fixed length m, computes a cipher-text block of fixed length n. Block ciphers are one of the basic building blocks used in many cryptographic systems. However, when used naively, by breaking up a message into blocks and encrypting each separately, they leak information, since identical input produces identical output. Thus an observer of the ciphertext can tell when two plaintext blocks are the same. As a result they generally they are combined using several simple operations that randomize the input into encrypted blocks. Procedures for doing this are called *modes of encryption*. In order to be efficient, modes of encryption must use simple operations, and this makes them more amenable to symbolic analysis. In particular Malozemoff et al.[5] and Hoang et al.[3] have found easily checkable symbolic conditions on modes constructed using block ciphers and exclusive-or that guarantee IND$-CPA security. IND$-CPA security is defined as indistinguishability of the output of modes of encryption from random by a *messagewise adaptive adversary* that, while not being able to compute the encryption function itself, plays a game in which it interacts multiple times with one of two oracles. One is an encryption oracle that, after receiving all the

blocks of a plaintext message outputs the corresponding ciphertext message. The other, given the same input from the adversary, outputs the corresponding number or random blocks. At the end of the game, the adversary must guess which oracle it was interacting with. It wins if it guesses correctly. The mode is secure if the probability of the adversary's winning differs from random by a negligible function of the security parameter η, that is it is less than $1/c^{-\eta}$, where c is a constant and η is the block length in bits.

In our own work we extend [3, 5] by 1) providing a symbolic criterion that not only guarantees security when satisfied but demonstrates an attack when not, and 2) applying the criterion to a stronger version of indistinguishability IND$-BCPA in which the game is played by a *blockwise adaptive adversary* that receives each ciphertext block from the oracle immediately after submitting a plaintext block. IND$-BCPA is known to be strictly stronger than IND$-CPA. For example, the cipher block chaining mode (see below) is IND$-CPA-secure but not IND$-BCPA-secure.

The interesting point from a unification perspective is that both IND$-CPA and IND$-BCPA hold if and only if the adversary cannot force the oracle to return two identical plaintext blocks with nonnegligible probability. When this is represented at the symbolic level it becomes a unification problem: can the adversary produce plaintext that unifies two blocks returned by the oracle? Note that this is a unification problem with constraints, however. When computing a unifier, the adversary cannot apply the encryption function itself. The only encrypted terms it can use are the encrypted blocks returned by the adversary.

Consider the example of Cipher Block Chaining, one of the oldest modes of encryption. It is defined as follows, where P_i is the i'th plaintext block, C_i is the i'th ciphertext block, IV is a randomly generated block known as the *initialization vector*, and $\oplus$ is bitwise exclusive-or.

$$C_0 = IV, C_1 = f(IV \oplus P1), \ldots, C_k = f(C_{k-1} \oplus P_k)$$

To demonstrate the fact that it is not IND$-BCPA, consider the following sequence of messages, where the x_i are variables standing for plaintext submitted by the adversary, and the other terms are computed by an oracle executing the mode of encryption algorithm.

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

The adversary can cause the two ciphertext blocks to be the same if it sets $x_2 = x_1 \oplus iv \oplus f(x_1 \oplus iv)$. Thus, the adversary is able to unify $f(x_1 \oplus iv)$ and $f(x_2 \oplus f(x_1 \oplus iv))$. However, if the adversary had only been able to see the two ciphertext blocks and iv after it had sent the two plaintext blocks, it would not have been able to perform this unification, since it would not has have seen iv, or $f(x_1 \oplus iv)$ at the time it computed x_2.

In [6], we explore the problem of symbolic criteria for security of modes of encryption against blockwise adaptive adversaries, and reduce it to a linear algebra problem with constraints via a variant of the Baader-Schulz [1] combination procedure. Baader-Schulz works by dividing a unification problem in the joint theory into two problems, one in each component theory. The problems are solved separately and then combined. In our case the two problems are the $\oplus$ theory for the adversary, and a solved form problem in the full theory for the oracle. The procedure and proofs sketched in this abstract apply to the symbolic adversary only, but we have also constructed a soundness and completeness proof for the probabilistic polynomial time adversary used in cryptography.

Let $(\mathcal{T}_\Sigma(\mathcal{X}), E)$ be the term algebra such that $\Sigma = \{f/1, \oplus/2, 0/1, r_1/0, \ldots, r_k/0\}$, and let E be the equational theory defined by $(R \uplus AC)$, where $R = \{x \oplus 0 = x, x \oplus x = 0, x \oplus x \oplus y = y\}$. In the following, all unification problems are defined over $(\mathcal{T}_\Sigma(\mathcal{X}), E)$.

Given a mode of encryption, we begin with a sequence of the form $[q_0, q_1, \ldots, q_m]$ describing the interaction of the adversary with the oracle encrypting messages using that mode, where each q_i is a either a variable (standing for a plaintext block input by the adversary), or $q_i \in (\mathcal{T}_\Sigma(\mathcal{X}), E)$ is rooted in f (standing for a ciphertext block returned by the oracle), and the only variables appearing in non-variable q_i are variable terms q_j such that $j < i$. We call the such a sequence a *mode of encryption (MOE) frame* and the collection of all possible MOE frames generated by a given mode a MOE program. Pick two f-rooted terms s_1 and s_2 from the MOE frame, and let $S = \{s_1 =^? s_2\}$ be the corresponding unification problem. In the following, we use the notation z^t to stand for a variable that replaces a term t.

Step 1: We convert S into a derived system S_2 as follows. For any term $t \in (\mathcal{T}_\Sigma(\mathcal{X}), E)$, we define pure(t) recursively as follows:

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

where z is a variable, r is a constant, and g is a variable, constant, 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 constant, add $z^r = r$ to S_2.
3. Add $z^{s_1} =^? z^{s_2}$ to S_2.

We characterize the variables of S_2 as follows. If x is a variable standing for a term created by the adversary, we call it an *adversarial variable*. If $y = y^{f(t)}$ or y^c, where c is a constant, 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. If z is an f-variable or $\oplus$-variable, we call it an *oracle variable*

Step 2: We define the relation $<_\mathcal{O}$ on the variables of S_2 to be the relation defined 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 the adversarial and f-variables of S_2 by 1) $y^{f(t)} <_\mathcal{A} x$ if x is an adversarial variable, and $f(t)$ is 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.

It is possible to show that the transitive closure of $<_\mathcal{O} \cup <_\mathcal{A}$ is a suborder of a total order. Thus, in particular, it contains no cycles.

Step 3: Choose a partition P of the variables of S_2. We let I_P to be the set of equations $z_i =^? z_j$ such that z_i and z_j are in the same equivalence class in P and $z_i \neq z_j$. We let $S_P = S_2 \cup I_P$. Note that I_P is always nonempty, because it contains the equation $y^{s_1} =^? y^{s_2}$. Let $S_{\mathcal{A},P}$ be the set of all equations $\downarrow_{R,AC} (\overbrace{z_i} \oplus \overbrace{z_j}) =^? 0$ such that $z_i =^? z_j \in I_P$, where $\overbrace{z} = z$ if z is an adversarial or f-variable, and $\overbrace{z} = \text{pure}(t)$ if $z = w^t$ where w^t is an $\oplus$-variable. Let $S_\mathcal{O}$ be $S_2 \setminus \{z^{s_1} =^? z^{s_2}\}$. Note that $z^{s_1} =^? z^{s_2} \in S_{\mathcal{A},P}$. Note that $S_{\mathcal{A},P} \cup S_\mathcal{O}$ is equivalent to $S_2 \cup I_P$, that $S_\mathcal{O}$ describe's the oracle's program, and that $Var(S_{\mathcal{A},P})$ consists of adversarial and f-variables, and $Sym(S_{\mathcal{A},P}) = \{\oplus\}$.

.

Step 4: For each f-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} and y^{t_j} are in the same element of the partition 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 sent by the oracle to the adversary, let πy^{t_i} be the earliest such term sent that appears 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} and then reducing the result to normal form.

Step 5: Let $S_P = \pi S_{\mathcal{A},P} \cup S_{\mathcal{O}}$ where π is a choosing function. We define the *MOE solved form* of $\mathcal{M}(\pi S_{\mathcal{A},P})$ as follows. 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 of $\pi S_{\mathcal{A},P}$, and then converting the problem to reduced row echelon form in the x-variables. This means that each equation in $\pi S_{\mathcal{A},P}{}'$ is in one of the following forms:

1. $0 =^? 0$;

2. $x_i \oplus (\sum_{j=1}^{i-1} \oplus \alpha'_j x_{i-j}) =^? \sum_{j=1}^{n} \oplus \beta'_{i,j} y^{t_j}$ for a unique $1 \leq i \leq m$, where at most one such equation exists for each x_i, or ;

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

In the MOE solved form $\mathcal{M}(\pi S_{\mathcal{A},P})$, the equations containing adversarial variables are replaced by the equations $x_i =^? (\sum_{j=1}^{i-1} \oplus \alpha'_j x_{i-j}) \oplus (\sum_{j=1}^{n} \oplus \beta'_{i,j} y_j)$, and any equations $0 =^? 0$ are removed.

We say that the MOE solved form $\mathcal{M}(\pi S_{\mathcal{A},P})$ is *well-ordered* if every equation is of the form $x_i =^? (\sum_{j=1}^{i-1} \oplus \alpha'_j x_{i-j}) \oplus (\sum_{j=1}^{n} \oplus \beta'_{i,j} y_j)$ such that $\beta'_{i,j} = 1$ implies $y_j <_{\mathcal{A}} x_i$.

Theorem. *A mode of encryption is* IND$-BCPA *secure if and only if its MOE program contains no MOE frame ϕ with two terms s_1 and s_2 sent by the oracle in ϕ, such that $\mathcal{M}(\pi S_{\mathcal{A},P})$ is well-ordered. where $S = \{s_1 =^? s_2\}$. Moreover, if $\mathcal{M}(\pi S_{\mathcal{A},P})$ is well-ordered, we can instantiate ϕ to a specification of an attack by applying the following substitution Θ to the adversarial variables x_i in the order which they are sent by the adversary:*

For $i = 1$ to n do

1. *If $x_i =^? \oplus(\sum_{j=1}^{i-1} \oplus \alpha'_j x_{i-j}) \oplus (\sum_{j=1}^{n} \oplus \beta'_{i,j} y^{t_j} \in \mathcal{M}(\pi S_{\mathcal{A},P})$, then $\Theta x_1 = \Theta((\sum_{j=1}^{i-1} \oplus \alpha'_j x_{i-j}) \oplus (\sum_{j=1}^{n} \oplus \beta'_{i,j} t_j))$;*

2. *Else $\Theta x_i = 0$.*

The idea of the proof in the symbolic model is that, if $\mathcal{M}(\pi S_{\mathcal{A},P})$ is not well-ordered, there is an equation $x_i =^? (\sum_{j=1}^{i-1} \oplus \alpha'_j x_{i-j}) \oplus (\sum_{j=1}^{n} \oplus \beta'_{i,j} y_j)$ and a j such that $\beta'_{i,j} = 1$ and $y_j \not<_{\mathcal{A}} x_i$. Thus, by the construction of the choosing function, in order to compute $\sigma(x_i \oplus \sum_{j=1}^{i-1} \oplus \alpha'_j \sigma x_{i-j}) = \sum_{j=1}^{n} \oplus \beta'_{i,j} \sigma y_j$, the adversary must compute a term $\sigma y_j = \sigma f(t_j)$ that it has not yet seen. Conversely, if $\mathcal{M}(\pi S_{\mathcal{A},P})$ is well-ordered, then Θ is computable by the adversary.

The proof in the cryptographic model is similar to the proof in the symbolic model, except that some complications introduced by the fact that the unifiers are now probabilistic polynomial-time functions need to be dealt with.

As an example, consider the sequence $[r_1, x_1, f(x_1 \oplus r_1), x_2, f(x_2 \oplus f(x_1 \oplus r_1))]$, and suppose we want to see if the adversary can compute a unifier of $f(x_1 \oplus r_1)$ and $f(x_2 \oplus f(x_1 \oplus r_1))$. We first apply Step 1 to convert S to the problem

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

Applying Step 2, we choose to identify $w^{x_1 \oplus r_1}$ and $w^{x_2 \oplus f(x_1 \oplus r_1)}$, which is indeed necessary in order to solve $y^{f(x_1 \oplus r_1)} =^? y^{f(x_2 \oplus f(x_1 \oplus r_1))}$. We obtain

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

Applying Step 3 we obtain the ordering $x_1 <_{\mathcal{A}} x_2$, $y^{r_1} <_{\mathcal{A}} x_1$, $y^{r_1} <_{\mathcal{A}} x_2$, and $y^{f(r_1 \oplus x_1)} <_{\mathcal{A}} x_2$.

We then apply Step 4 to obtain two systems of equations. System (1) describes the oracle program, and System (2) defines the system of equations the adversary must solve.

$$\begin{aligned} y^{r_1} &=^? r_1 & w^{x_1 \oplus r_1} &=^? x_1 \oplus y^{r_1} \\ y^{f(x_1 \oplus r_1)} &=^? f(w^{f(x_1 \oplus r_1)}) & w^{x_2 \oplus f(x_1 \oplus r_1)} &=^? x_2 \oplus y^{f(x_1 \oplus r_1)} \end{aligned} \tag{1}$$

$$y^{f(x_1 \oplus r_1)} =^? y^{f(x_2 \oplus f(x_1 \oplus r_1)} \qquad x_1 \oplus y^{r_1} =^? x_2 \oplus y^{f(x_2 \oplus f(x_1 \oplus r_1))} \tag{2}$$

Applying Step 5, we note that the choosing function is $\pi y^{f(x_2 \oplus f(x_1 \oplus r_1)} = y^{f(x_1 \oplus r_1)}$, and we convert the system to MOE solved form $x_2 =^? x_1 \oplus y^{r_1} \oplus y^{f(x_1 \oplus r_1)}$.

To solve System (2,) we note that $\{x_2 =^? x_1 \oplus y^{r_1} \oplus y^{f(x_1 \oplus r_1)}\}$ is well-ordered, so that $\sigma x_2 = x_1 \oplus y^{r_1} \oplus y^{f(x_1 \oplus r_1)}$ is computable by the adversary. Combining the solutions to the two systems, we obtain $\sigma x_1 = x_1$, $\sigma x_2 = x_1 \oplus y^{r_1} \oplus y^{f(x_1 \oplus r_1)}$, and $\sigma z^t = t$ for each oracle variable z^t. Computing the oracle's substitution to eliminate the variables introduced in Step 1, we obtain

$$\sigma x_1 = x_1, \ \sigma x_2 = x_1 \oplus r_1 \oplus f(x_1 \oplus r_2).$$

On the other hand, if we are given the sequence $[x_1, x_2, r_1, f(x_1 \oplus r_1), f(x_2 \oplus f(x_1 \oplus r_1))]$ we find that the adversary cannot compute the unifier of the two terms $f(x_1 \oplus r_1)$ and $f(x_2 \oplus f(x_1 \oplus r_1)$. First, we note that unifying the two terms forces us to set $w^{x_1 \oplus r_1} =^? w^{x_2 \oplus f(x_1 \oplus r_1}$, giving us same equation as before: : $x_2 =^? x_1 \oplus y^{r_1} \oplus y^{f(x_1 \oplus r_1)}$. However, since there are no terms sent by the oracle before x_2 is sent by the adversary, there is no partition of $Var(S_2)$ that can turn this into a well-ordered equation.

We list some open problems below.

1. How do we choose likely candidate pairs of terms to be unified, and rule out unsuitable ones? Are there any shortcuts in specific cases?

2. Can the efficiency of the algorithm be improved, e.g. by intelligent choice of partitions? More generally, are there more efficient unification algorithms guaranteeing the same constraints?

3. We may want to extend these results to modes of encryption that are not restricted to returning f-rooted terms; for example, they could return $\oplus$-sums of f-rooted terms. The problem in that case goes beyond finding unifiers of f-rooted terms to finding substitutions that induce linear relations between values returned by the mode. What is the best way of searching for these?

4. Can this algorithm be applied or extended to the analysis of other types of cryptosystems? In particular, the symbolic Linicrypt model of Carmer and Rosulek [2] is based, similarly to ours, on splitting the security problem into a linear algebra problem solved by the adversary and a set of constraints imposed by the oracle. It has been used in [2] to decide security of a class of encryption algorithms called *garbled circuits*, in which principals compute a joint output without either revealing their input to each other. However, although the Linicrypt model allows for adaptive adversary, the algorithm in [2] assumes a very weak adaptive adversary that can pick adaptively from a set of ciphertexts, but cannot introduce plaintext input itself. Exploring the Linicrypt model may provide further applications for our approach to reasoning about stronger adaptive adversaries.

5 These results may possibly be extended to theories that are not cryptographically sound or complete. One example involves the increment by one function inc which is used to implement secure counter mode, which is known to be secure against blockwise adaptive adversaries. The inc function, when combined with $\oplus$ results in a term algebra which has no unambiguous translation to the computational level, since $\mathsf{inc}(r) = r \oplus 1$ with probability $1/2$ for random r. This problem is dealt with in [5] by restriction to cases where this ambiguity can be avoided. This requires a separation of the two operations so that they do not interact with each other. This introduces another kind of constraint. Can one make use of these constraints to develop symbolic algorithms and modes of encryption for which they can be used to decide security?

6 Looking beyond the unification problem, we consider the complexity of deciding security at the symbolic level. Modes of encryption are generally defined recursively. However, the problem of security for recursively defined protocols using exclusive-or is not that well understood. In [4] 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, which require that, whenever the $\oplus$ function appears, at least one argument should not depend on adversarial input, a restriction that would rule out most most practical modes of encryption. On the other hand, the undecidability proof makes extensive use of the concatenation operator, as well as the $\oplus$ and hash operators. If the concatenation operator is removed, the decidability result might change.

References

[1] 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.

[2] Brent Carmer and Mike Rosulek. Linicrypt: A model for practical cryptography. In *CRYPTO 2016*, pages 416–445, 2016.

[3] Viet Tung Hoang, Jonathan Katz, and Alex J. Malozemoff. Automated analysis and synthesis of authenticated encryption schemes. In *ACM Computer and Communications Security*, pages 84–95, 2015.

[4] Ralf Küsters and Tomasz Truderung. On the automatic analysis of recursive security protocols with XOR. In *STACS 2007, Theoretical Aspects of Computer Science*, pages 646–657, 2007.

[5] 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)*, pages 140–152. IEEE, 2014.

[6] Catherine Meadows. Symbolic security criteria for blockwise adaptive secure modes of encryption. in preperation.