Automated Security Proof for Symmetric Encryption Modes

Martin Gagné, Pascal Lafourcade, Yassine Lakhnech, Reihaneh Safavi-Naini

Automated Security Proof for Symmetric Encryption Modes. *

Martin Gagné[2], Pascal Lafourcade[1], Yassine Lakhnech[1], and Reihaneh Safavi-Naini[2]

[1] Université Grenoble 1, CNRS,Verimag, FRANCE
[2] Department of Computer Science, University of Calgary, Canada

Abstract. We presents a compositional Hoare logic for proving semantic security of modes of operation for symmetric key block ciphers. We propose a simple programming language to specify encryption modes and an assertion language that allows to state invariants and axioms and rules to establish such invariants. The assertion language consists of few atomic predicates. We were able to use our method to verify semantic security of several encryption modes including Cipher Block Chaining (CBC), Cipher Feedback mode (CFB), Output Feedback (OFB), and Counter mode (CTR).

1 Introduction

A block cipher algorithm (e.g. AES, Blowfish, DES, Serpent and Twofish) is a symmetric key algorithm that takes a fixed size input message block and produces a fixed size output block. A mode of operation is a method of using a block cipher on an arbitrary length message. Important modes of operation are Electronic Code Book (ECB), Cipher Block Chaining (CBC), Cipher FeedBack mode (CFB), Output FeedBack (OFB), and Counter mode (CTR). Modes of operations have different applications and provide different levels of security and efficiency. An important question when a mode of operation is used for encryption is the level of security that the mode provides, assuming the underlying block cipher is secure. The answer to this question is not straightforward. For example if one uses the naive ECB mode with a "secure" block cipher, then the encryption scheme obtained is not even IND-CPA secure. Others, like CBC or CTR, will provide confidentiality only if the initial vector (IV) is chosen adequately.

Recent years have seen an explosion of new modes of operation for block cipher (IACBC, IAPM [19], XCB [23], TMAC [18, 20], HCTR [5], HCH [7], EMU [15], EMU* [12], PEP [6], OMAC [16, 17], TET [13], CMC [14], GCM [24], EAX [4], XEX [25], TAE, TCH, TBC [22, 28] to name only a few). These new modes of operation often offer improved security guarantees, or additional security features. They also tend to be more complex than the traditional modes of operations, and arguments for proving their security can similarly become much

* This work was supported by ANR SeSur SCALP, SFINCS, AVOTE and iCORE.

more complicated – sometimes so complicated that flaws in the security proofs could go unnoticed for years.

Proofs generated by automated verification tools can provide us with an independent argument for the security of modes of operation, thereby increasing our confidence in the security of cryptographic protocols. While the rules used by the prover must also be proven by humans, and are therefore also susceptible to error, they tend to be much simpler than the protocols they will be used to check, which ensures that mistakes are far less likely to go unnoticed. In this paper, we take a first step towards building an automated prover for modes of operation, and show how to automatically generate proofs for many traditional block cipher modes of operation.

Contributions: We propose a compositional Hoare logic for proving semantic security of modes of operation for symmetric key block ciphers. We notice that many modes use a small set of operations such as xor, concatenation, and selection of random values. We introduce a simple programming language to specify encryption modes and an assertion language that allows to state invariants and axioms and rules to establish such invariants. The assertion language requires only four predicates: one that allows us to express that the value of a variable is indistinguishable from a random value when given the values of a set of variables, one that states that an expression has not been yet submitted to the block cipher, and two bookkeeping predicates that allow us to keep track of 'fresh' random values and counters. Transforming the Hoare logic into an (incomplete) automated verification procedure is quite standard. Indeed, we can interpret the logic as a set of rules that tell us how to propagate the invariants backwards. Using our method, an automated prover could verify semantic security of several encryption modes including CBC, CFB, CTR and OFB. Of course our system does not prove ECB mode, because ECB is not semantically secure.

Related Work: Security of symmetric encryption modes have been studied for a long time by the cryptographers. In [1] the authors presented different concrete security notions for symmetric encryption in a concrete security framework. For instance, they give a security analysis of CBC mode. In [2] a security analysis of the encryption mode CBC-MAC [21]. In [26] they propose a new encryption mode called OCB for efficient authenticated encryption and provide a security analysis of this new mode. Many other works present proofs of encryption modes.

Other works try to encode security of symmetric encryption modes as a non-interference property for programs with deterministic encryption. For example, [9] presents a computationally sound type system with exact security bounds for such programs. This type system has been applied to verify some symmetric encryption modes. The logic presented in this paper can be used to give a more structured soundness proof for the proposed type system. Moreover, we believe that our logic is more expressive and can be more easily adapted to more encryption modes.

A first important feature of our method is that it is not based on a global reasoning and global program transformation as it is the case for the game-based approach [3, 27].

In [8], the authors proposed an automatic method for proving semantic security for asymmetric generic encryption schemes. Our work continues that line of work. We extend the input language and axioms of the Hoare logic of [8] in order to capture symmetric encryption modes.

OUTLINE: In Section 2 we introduce the material for describing the encryption modes. In Section 3, we present our Hoare Logic for analyzing the semantic security of encryption modes described with the grammar given in the previous section. Finally before concluding in the last section, we apply our method to some examples in Section 4.

2 Definitions

2.1 Notation and Conventions

For simplicity, over this paper, we assume that all variables range over large domains, whose cardinality is exponential in the security parameter η. We also assume that all programs have length polynomial in η.

A block cipher is a function $\mathcal{E} : \{0,1\}^k \times \{0,1\}^\eta \to \{0,1\}^\eta$ such that for each $K \in \{0,1\}^k$, $\mathcal{E}(K,\cdot)$ is a permutation. It takes as input a k-bit key and an η-bit message block, and returns an η-bit string. We often denote by $\mathcal{E}(x)$ the application of the block cipher to the message block x. We omit the key used every time to simplify the notation, but it is understood that a key was selected at random at the beginning of the experiment and remains the same throughout.

For a mode of operation M, we denote by $\mathcal{E}_M$ the encryption function described by M using block cipher $\mathcal{E}$.

For a probability distribution $\mathcal{D}$, we denote by $x \xleftarrow{\$} \mathcal{D}$ the operation of sampling a value x according to distribution $\mathcal{D}$. If S is a finite set, we denote by $x \xleftarrow{\$} S$ the operation of sampling x uniformly at random among the values in S.

Given two distribution ensembles $X = \{X_\eta\}_{\eta\in\mathbb{N}}$ and $X' = \{X'_\eta\}_{\eta\in\mathbb{N}}$, an algorithm $\mathcal{A}$ and $\eta \in \mathbb{N}$, we define the *advantage* of $\mathcal{A}$ in distinguishing X_η and X'_η as the following quantity:

$$\mathsf{Adv}(\mathcal{A},\eta,X,X') = \mathsf{Pr}[x \xleftarrow{\$} X_\eta : \mathcal{A}(x) = 1] - \mathsf{Pr}[x \xleftarrow{\$} X'_\eta : \mathcal{A}(x) = 1].$$

Two distribution ensembles X and X' are called *indistinguishable*, denoted by $X \sim X'$, if $\mathsf{Adv}(\mathcal{A},\eta,X,X')$ is negligible as a function of η for every probabilistic polynomial-time algorithm $\mathcal{A}$.

2.2 Grammar

We introduce our language for defining a generic encryption mode. The commands are given by the grammar of Figure 1, where:

- $x \xleftarrow{\$} \mathcal{U}$ denotes uniform sampling of a value and assigning it to x.

- $x := \mathcal{E}(y)$ denotes application of the block cipher $\mathcal{E}$ to the value of y and assigning the result to x.
- Similarly for $x := \mathcal{E}^{-1}(y)$, where $\mathcal{E}^{-1}$ denotes the inverse function of $\mathcal{E}$.
- $x := y \oplus z$ denotes application of the exclusive-or operator to the values of y and z and assigning the result to x.
- $x := y||z$ represents the concatenation of the values of y and z.
- $x := y[n, m]$ assigns to x the bits at positions between n and m in the bit-string value of y. I.e., for a bit-string $bs = b_1 \dots b_k$, where the b_i's are bits, $bs[n, m]$ denotes the bits-string $b_n \dots b_m$[1]. Then, $x := y[n, m]$ assigns $bs[n, m]$ to x, where bs is the value of y. Here, n and m are polynomials in the security parameter η.
- $x := y + 1$ increments by one the value of y and assigns the result to x. The operation is carried modulo 2^η.
- $c_1; c_2$ is the sequential composition of c_1 and c_2.

$$\begin{aligned} \mathrm{c} ::= {} & x \xleftarrow{\$} \mathcal{U} \mid x := \mathcal{E}(y) \mid x := \mathcal{E}^{-1}(y) \mid x := y \oplus z \mid x := y \| z \mid x := y[n, m] \mid \\ & \mid x := y + 1 \mid c_1; c_2 \end{aligned}$$

Fig. 1. Language grammar

2.3 Generic Encryption Mode

We can now formally define a mode of encryption.

Definition 1 (Generic Encryption Mode). *A generic encryption mode M is represented by $\mathcal{E}_M(m_1|\dots|m_i, c_0|\dots|c_i)$: **var** $\boldsymbol{x_i}; \mathsf{c}_i$, where $\boldsymbol{x_i}$ is the set of variables used in c_i, all commands of c_i are built using the grammar described in Figure 1, each m_j is a message blocks, and each c_j is a cipher block, both of size n according to the input length of the block cipher $\mathcal{E}$.*

We add the additional block c_0 to the ciphertext because encryption modes are usually generate ciphertexts longer than the message. In all examples in this paper, c_0 will be the initialization vector (IV). The definition can easily be extended for encryption modes that also add one or more blocks at the end.

In Figure 2, we present the famous encryption mode $\mathcal{E}_{CBC}$ for a message of three blocks.

2.4 Semantics

In addition to the variables in Var,[2] we consider a variable $\mathcal{T}_E$ that records the values on which $\mathcal{E}$ was computed and cannot be accessed by the adversary.

[1] Notice that $bs[n, m] = \epsilon$, when $m < n$ and $bs[n, m] = bs[n, k]$, when $m > k$

[2] We denote by Var the complete set of variables in the program, whereas **var** denotes the set of variables in the program that are not input or output variables.

$\mathcal{E}_{CBC}(m_1|m_2|m_3, IV|c_1|c_2|c_3)$:
var z_1, z_2, z_3;
$IV \xleftarrow{\$} \mathcal{U}$;
$z_1 := IV \oplus m_1$;
$c_1 := \mathcal{E}(z_1)$;
$z_2 := c_1 \oplus m_2$;
$c_2 := \mathcal{E}(z_2)$;
$z_3 := c_2 \oplus m_3$;
$c_3 := \mathcal{E}(z_3)$;

Fig. 2. Description of $\mathcal{E}_{CBC}$

Thus, we consider states that assign bit-strings to the variables in Var and lists of pairs of bit-strings to $\mathcal{T}_E$. Given a state S, $S(\mathcal{T}_E).\mathsf{dom}$ and $S(\mathcal{T}_E).\mathsf{res}$ denote the lists obtained by projecting each pair in $S(\mathcal{T}_E)$ to its first and second element respectively.

The state also contains two sets of variables, F and C, which are used for bookkeeping purposes. The set F contains the variables with values that were sampled at random or obtained as a result of the computation of the block cipher, and have not yet been operated on. Those values are called *fresh* random values. The set C contains the variables whose value are the most recent increment of a counter that started at a fresh random value.

A program takes as input a *configuration* $(S, \mathcal{E})$ and yields a distribution on configurations. A configuration is composed of a state S, a block cipher $\mathcal{E}$. Let $\Gamma_{\mathcal{E}}$ denote the set of configurations and $\textsc{Dist}(\Gamma_{\mathcal{E}})$ the set of distributions on configurations. The semantics is given in Table 1. In the table, $\delta(x)$ denotes the Dirac measure, i.e. $\mathsf{Pr}[x] = 1$ and $\mathcal{T}_E \mapsto S(\mathcal{T}_E) \cdot (x, y)$ denotes the addition of element (x, y) to $\mathcal{T}_E$. Notice that the semantic function of commands can be lifted in the usual way to a function from $\textsc{Dist}(\Gamma_{\mathcal{E}})$ to $\textsc{Dist}(\Gamma_{\mathcal{E}})$. That is, let $\phi : \Gamma_{\mathcal{E}} \to \textsc{Dist}(\Gamma_{\mathcal{E}})$ be a function. Then, ϕ defines a unique function $\phi^* : \textsc{Dist}(\Gamma_{\mathcal{E}}) \to \textsc{Dist}(\Gamma_{\mathcal{E}})$ obtained by point-wise application of ϕ. By abuse of notation we also denote the lifted semantics by $[\![\mathsf{c}]\!]$.

A notational convention. It is easy to see that commands never alter $\mathcal{E}$. Therefore, we can, without ambiguity, write $S' \xleftarrow{\$} [\![c]\!](S, \mathcal{E})$ instead of $(S', \mathcal{E}) \xleftarrow{\$} [\![c]\!](S, \mathcal{E})$.

Here, we are only interested in distributions that can be constructed in polynomial time. We denote their set by $\textsc{Dist}(\Gamma, \mathcal{F})$, where $\mathcal{F}$ is a family of block ciphers, and is defined as the set of distributions of the form:

$$[\mathcal{E} \xleftarrow{\$} \mathcal{F}(1^\eta); S \xleftarrow{\$} [\![p]\!](I, \mathcal{E}) : (S, \mathcal{E})]$$

where p is a program with a polynomial number of commands, and I is the "initial" state, in which all variables are undefined and all lists and sets are empty.

$$[\![x \xleftarrow{\$} \mathcal{U}]\!](S,\mathcal{E}) = [u \xleftarrow{\$} \mathcal{U} : (S\{x \mapsto u, \mathsf{F} \mapsto \mathsf{F} \cup \{x\}, \mathsf{C} \mapsto \mathsf{C} \setminus \{x\}\}, \mathcal{E})]$$
$$[\![x := \mathcal{E}(y)]\!](S,\mathcal{E}) =$$
$$\begin{cases} \delta(S\{x \mapsto v, F \mapsto F \cup \{x\} \setminus \{y\}, \mathsf{C} \mapsto \mathsf{C} \setminus \{x\}\}, \mathcal{E}) \text{ if } (S(y), v) \in S(\mathcal{T}_E) \\ \delta(S\{x \mapsto v, F \mapsto F \cup \{x\} \setminus \{y\}, \mathsf{C} \mapsto \mathsf{C} \setminus \{x\}, \mathcal{T}_E \mapsto S(\mathcal{T}_E) \cdot (S(y), v)\}, \mathcal{E}) \\ \qquad\qquad \text{if } (S(y), v) \notin S(\mathcal{T}_E) \text{ and } v = \mathcal{E}(S(y)) \end{cases}$$
$$[\![x := \mathcal{E}^{-1}(y)]\!](S,\mathcal{E}) = \delta(S\{x \mapsto \mathcal{E}^{-1}(S(y)), F \mapsto F \setminus \{x, y\}, \mathsf{C} \mapsto \mathsf{C} \setminus \{x\}\}, \mathcal{E})$$
$$[\![x := y \oplus z]\!](S,\mathcal{E}) = \delta(S\{x \mapsto S(y) \oplus S(z), F \mapsto F \setminus \{x, y, z\}, \mathsf{C} \mapsto \mathsf{C} \setminus \{x\}\}, \mathcal{E})$$
$$[\![x := y||z]\!](S,\mathcal{E}) = \delta(S\{x \mapsto S(y)||S(z), F \mapsto F \setminus \{x, y, z\}, \mathsf{C} \mapsto \mathsf{C} \setminus \{x\}\}, \mathcal{E})$$
$$[\![x := y[n, m]]\!](S,\mathcal{E}) = \delta(S\{x \mapsto S(y)[n, m], F \mapsto F \setminus \{x, y\}, \mathsf{C} \mapsto \mathsf{C} \setminus \{x\}\}, \mathcal{E})$$
$$[\![x := y + 1]\!](S,\mathcal{E}) =$$
$$\begin{cases} \delta(S\{x \mapsto S(y) + 1, C \mapsto C \cup \{x\} \setminus \{y\}, F \mapsto F \setminus \{x, y\}\}, \mathcal{E}) \text{ if } y \in S(F) \text{ or } y \in S(C) \\ \delta(S\{x \mapsto S(y) + 1, F \mapsto F \setminus \{x, y\}, \mathsf{C} \mapsto \mathsf{C} \setminus \{x\}\}, \mathcal{E}) \text{ otherwise} \end{cases}$$
$$[\![c_1; c_2]\!] = [\![c_2]\!] \circ [\![c_1]\!]$$

Table 1. The semantics of the programming language

2.5 Security Model

Ideal Cipher Model

We prove the modes of encryption secure in the ideal cipher model. That is, we assume that the block cipher is a pseudo-random function.[3] This is a standard assumption for proving the security of any block-cipher-based scheme.

The advantage of an algorithm $\mathcal{A}$ against a family of pseudo-random function is defined as follows.

Definition 2. *Let $P : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be a family of functions and let $\mathcal{A}$ be an algorithm that takes an oracle and returns a bit. The* prf-advantage *of $\mathcal{A}$ is defined as follows.*

$$\mathsf{Adv}^{prf}_{\mathcal{A},P} = Pr[K \xleftarrow{\$} \{0,1\}^k; \mathcal{A}^{P(K,\cdot)} = 1] - Pr[R \xleftarrow{\$} \Phi_n; \mathcal{A}^{R(\cdot)} = 1]$$

where Φ_n is the set of all functions from $\{0,1\}^n$ to $\{0,1\}^n$.

The security of a symmetric mode of operation is usually proven by first showing that the mode of operation would be secure if $\mathcal{E}$ was a random function in Φ_n. As a result, an adversary $\mathcal{A}$ against the encryption scheme can be transformed into an adversary $\mathcal{B}$ against the block cipher (as a pseudo-random function) with a similar running time, such that $\mathcal{B}$'s prf-advantage is similar to $\mathcal{A}$'s advantage in breaking the encryption scheme.

Encryption Security

Semantic security for a mode of encryption is defined as follows.

[3] While block ciphers are really families of permutations, it is well known that pseudo-random permutations are indistinguishable from pseudo-random functions if the block size is large enough.

Definition 3. *Let* $\mathcal{E}_M(m_1|\ldots|m_i, c_0|\ldots|c_i) : \mathbf{var}\ \boldsymbol{x_i}; \mathsf{c}_i$ *be a generic encryption mode.* $A = (A_1, A_2)$ *be an adversary and* $X \in \textsc{Dist}(\Gamma, \mathcal{E})$*. For* $\eta \in \mathbb{N}$*, let*

$$\begin{aligned}
&\mathsf{Adv}^{ind-CPA}_{A,M}(\eta, X) \\
&\quad = 2 * \mathsf{Pr}[(S, \mathcal{E}) \xleftarrow{\$} X; \\
&\qquad (x_0, x_1, p, s) \xleftarrow{\$} A_1^{\mathcal{O}_1}(\eta); b \xleftarrow{\$} \{0,1\}; \\
&\qquad S' \xleftarrow{\$} [\![\mathsf{c}_p]\!](S\{m_1|\ldots|m_p \mapsto x_b\}, \mathcal{E}) : \\
&\qquad A_2^{\mathcal{O}_2}(x_0, x_1, s, S'(c_0|\ldots|c_p)) = b] - 1
\end{aligned}$$

where $\mathcal{O}_1 = \mathcal{O}_2$ *are oracles that take a pair* (m, j) *as input, where* m *is a string and* j *is the block length of* m*, and answers using the* j^{th} *algorithm in* $\mathcal{E}_M$*.* A_1 *outputs* x_0, x_1 *such that* $|x_0| = |x_1|$ *and are composed of* p *blocks. The mode of operation* M *is semantically (IND-CPA) secure if* $\mathsf{Adv}^{ind-CPA}_{A,M}(\eta, X)$ *is negligible for any constructible distribution ensemble* X *and polynomial-time adversary* A*.*

It is important to note that in this definition, an adversary against the scheme is only given oracle access to the encryption mode $\mathcal{E}_M$, and *not* to the block cipher $\mathcal{E}$ itself.

Our method verifies the security of an encryption scheme by proving that the ciphertext is indistinguishable from random bits. It is a classical result that this implies semantic security.

3 Proving Semantic Security

In this section, we present our Hoare logic for proving semantic (IND-CPA) security for generic encryption mode defined with our language. We prove that our logic is sound although not complete. Our logic can be used to annotate each command of our programming language with a set of invariants that hold at each point of the program for any execution.

3.1 Assertion Language

We consider new predicates in order to consider properties of symmetric encryption modes. We use a Hoare Logic based on the following invariants:

$$\begin{aligned}
&\varphi ::= \mathsf{true} \mid \varphi \wedge \varphi \mid \psi \\
&\psi ::= \mathsf{Indis}(\nu x; V) \mid F(x) \mid \mathsf{E}(\mathcal{E}, e) \mid \mathit{Rcounter}(e),
\end{aligned}$$

where $V \subseteq \mathsf{Var}$ and e is an expression constructible out of the variables used in the program and the grammar presented in Section 2. Intuitively:

Indis$(\nu x; V)$**:** means that any adversary has negligible probability to distinguish whether he is given results of computations performed using the value of x or a random value, when he is given the values of the variables in V.

E$(\mathcal{E}, e)$**:** means that the probability that the value $\mathcal{E}(e)$ has already been computed is negligible.

$F(e)$**:** means e is a fresh random value.
$RCounter(e)$**:** means that e is the most recent value of a counter that started at a fresh random value.

More formally, for each invariant ψ, we define that a distribution X satisfies ψ, denoted $X \models \psi$ as follows:

- $X \models \mathsf{true}$.
- $X \models \varphi \wedge \varphi'$ iff $X \models \varphi$ and $X \models \varphi'$.
- $X \models \mathsf{Indis}(\nu x; V)$ iff $[(S, \mathcal{E}) \xleftarrow{\$} X : (S(x, V), \mathcal{E})] \sim [(S, \mathcal{E}) \xleftarrow{\$} X; u \xleftarrow{\$} \mathcal{U}; S' = S\{x \mapsto u\} : (S'(x, V), \mathcal{E})]$
- $X \models \mathsf{E}(\mathcal{E}, e)$ iff $\mathsf{Pr}[(S, \mathcal{E}) \xleftarrow{\$} X : S(e) \in S(\mathcal{T}_E).\mathsf{dom}]$ is negligible.
- $X \models F(e)$ iff $\mathsf{Pr}[(S, E) \xleftarrow{\$} X : e \in S(\mathsf{F})] = 1$.
- $X \models RCounter(e)$ iff $\mathsf{Pr}[(S, E) \xleftarrow{\$} X : e \in S(\mathsf{C})] = 1$.

3.2 Hoare Logic Rules

We present a set of rules of the form $\{\varphi\}\mathsf{c}\{\varphi'\}$, meaning that execution of command c in any distribution that satisfies φ leads to a distribution that satisfies φ'. Using Hoare logic terminology, this means that the triple $\{\varphi\}\mathsf{c}\{\varphi'\}$ is valid. We group rules together according to their corresponding commands. We do not provide rules for the commands $x := \mathcal{E}^{-1}(y)$ or $x := y[n, m]$ since those commands are only used during decryption.

Notation: For a set V, we write V, x as a shorthand for $V \cup \{x\}$, $V - x$ as a shorthand for $V \setminus \{x\}$, and $\mathsf{Indis}(\nu x)$ as a shorthand for $\mathsf{Indis}(\nu x; \mathsf{Var})$.

Random Assignment:

- (R1) $\{true\}\ x \xleftarrow{\$} \mathcal{U}\ \{F(x) \wedge \mathsf{Indis}(\nu x) \wedge \mathsf{E}(\mathcal{E}, x)\}$
- (R2) $\{\mathsf{Indis}(\nu y; V)\}\ x \xleftarrow{\$} \mathcal{U}\ \{\mathsf{Indis}(\nu y; V, x)\}$

Increment:

- (I1) $\{F(y)\}\ x := y + 1\ \{RCounter(x) \wedge \mathsf{E}(\mathcal{E}, x)\}$
- (I2) $\{RCounter(y)\}\ x := y + 1\ \{RCounter(x) \wedge \mathsf{E}(E, x)\}$
- (I3) $\{\mathsf{Indis}(\nu z; V)\}\ x := y + 1\ \{\mathsf{Indis}(\nu z; V - x)\}$ if $z \neq x, y$ and $y \notin V$

Xor operator:

- (X1) $\{\mathsf{Indis}(\nu y; V, y, z)\}x := y \oplus z\{\mathsf{Indis}(\nu x; V, x, z)\}$ where $x, y, z \notin V$,
- (X2) $\{\mathsf{Indis}(\nu y; V, x)\}x := y \oplus z\{\mathsf{Indis}(\nu y; V)\}$ where $x \notin V$,
- (X3) $\{\mathsf{Indis}(\nu t; V, y, z)\}\ x := y \oplus z\ \{\mathsf{Indis}(\nu t; V, x, y, z)\}$ if $t \neq x, y, z$ and $x, y, z \notin V$
- (X4) $\{F(y)\}\ x := y \oplus z\ \{\mathsf{E}(\mathcal{E}, x)\}$ if $y \neq z$

Concatenation:

- (C1) $\{\mathsf{Indis}(\nu y; V, y, z)\} \wedge \{\mathsf{Indis}(\nu z; V, y, z)\}\ x := y \| z\ \{\mathsf{Indis}(\nu x; V, x)\}$ if $y, z \notin V$

– (C2) $\{\mathsf{Indis}(\nu t; V, y, z)\}\ x := y \| z\ \{\mathsf{Indis}(\nu t; V, x, y, z)\}$ if $t \neq x, y, z$

Block cipher:

– (B1) $\{\mathsf{E}(\mathcal{E}, y)\}\ x := \mathcal{E}(y)\ \{F(x) \wedge \mathsf{Indis}(\nu x) \wedge \mathsf{E}(\mathcal{E}, x)\}$
– (B2) $\{\mathsf{E}(\mathcal{E}, y) \wedge \mathsf{Indis}(\nu z; V)\}\ x := \mathcal{E}(y)\ \{\mathsf{Indis}(\nu z; V)\}$ provided $z \neq x$
– (B3) $\{\mathsf{E}(\mathcal{E}, y) \wedge \mathit{Rcounter}(z)\}\ x := \mathcal{E}(z)\ \{\mathit{Rcounter}(z)\}$ provided $z \neq x$
– (B4) $\{\mathsf{E}(\mathcal{E}, y) \wedge \mathsf{E}(\mathcal{E}, z)\}\ x := \mathcal{E}(y)\ \{\mathsf{E}(\mathcal{E}, z)\}$ provided $z \neq x, y$
– (B5) $\{\mathsf{E}(\mathcal{E}, y) \wedge F(z)\}\ x := \mathcal{E}(y)\ \{F(z)\}$ provided $z \neq x, y$

Finally, we add a few rules whose purpose is to preserve invariants that are unaffected by the command.

Generic preservation rules:

Assume that $z \neq x, w, v$ and c is either $x \xleftarrow{\$} \mathcal{U}$, $x := w \| v$, $x := w \oplus v$, or $x := w + 1$:

– (G1) $\{\mathsf{Indis}(\nu z; V)\}\ \mathsf{c}\ \{\mathsf{Indis}(\nu z; V)\}$ provided $w, v \in V$
– (G2) $\{\mathsf{E}(\mathcal{E}, z)\}\ \mathsf{c}\ \{\mathsf{E}(\mathcal{E}, z)\}$
– (G3) $\{\mathit{RCounter}(z)\}\ \mathsf{c}\ \{\mathit{RCounter}(z)\}$
– (G4) $\{F(z)\}\ \mathsf{c}\ \{F(z)\}$

3.3 Proof Sketches

Due to space restrictions, we cannot present formal proofs of all our rules here. We present quick sketches instead to give the reader some intuition as to why each rule holds. The complete proofs are available in our full manuscript [11].

Rules for random assignment.

In rule (R1), $F(x)$ simply follows from the definition of $F(\cdot)$, and $\mathsf{Indis}(\nu x)$ should be obvious since x has just been sampled at random, independently of all other values. Also, since the block cipher has been computed only on a polynomial number of values, out of an exponential domain, the probability that x has been queried to the block cipher is clearly negligible. Rule (R2) is easily proven using the fact that, at this point, x is independent from all other values in the program.

Rules for increment.

For rules (I1) and (I2) the behavior of $\mathit{RCounter}(\cdot)$ easily follows from its definition. Note that since we have either $F(y)$ or $\mathit{RCounter}(y)$, y (and x) were obtained by repeatedly applying $+1$ to a random value r, i.e. $x = r + k$ for some number k. Since $\mathcal{E}$ was computed only on a polynomial number of values, the probability of being less than k away from one of those values is negligible, therefore the probability that x has been queried to the block cipher is negligible. In (I3), if $\mathsf{Indis}(\nu z; V)$ holds, then clearly $\mathsf{Indis}(\nu z; V - x)$ holds as well, and the values in $V - x$ are unchanged by the command.

Rules for Xor.

Rules (X1) and (X2) are proven by considering y as a one-time pad applied to z. As a result, one of x or y will be indistinguishable from random provided that the other is not known. For (X3), one simply notes that x is easy to construct from y and z, so if t is indistinguishable from random given y and z, then it is also indistinguishable from random given x, y and z. For rule (X4), since y is fresh, it is still independent from all other values, from z in particular. It then follows that x has the same distribution as y and is independent from all values except y and therefore, the probability that it has been queried to $\mathcal{E}$ is negligible for the same reason that y is.

Rules for concatenation.

Rules (C1) and (C2) follow simply from the observation that the concatenation of two independent random strings is a random string.

Rules for block cipher.

To prove (B1), in the Ideal Cipher Model, $\mathcal{E}$ is sampled at random among all possible functions $\{0,1\}^\eta \to \{0,1\}^\eta$. Since y has never been queried to the block cipher, $x := \mathcal{E}(y)$ is indistinguishable from an independent random value, and so possess the same invariants as if $x \xleftarrow{\$} \mathcal{U}$ had been executed. Rules (B2) to (B5) simply preserve invariants that are unaffected by the computation of the block cipher on a value that has never been queried before.

Generic preservation rules.

The conditions for applying those rules, particularly $z \neq x, w, v$ were designed specifically so that the command would have no effect on the invariant. The invariant is therefore preserved.

As a result of all this, we have the following:

Proposition 1. *In the Ideal Cipher Model, the Hoare triples given in the previous rules are valid.*

As a result, our method can be used to prove the semantic security of an encryption mode by proving that, from the adversary's point of view, the ciphertexts are indistinguishable from random bits.

Proposition 2. *Let $\mathcal{E}_M(m_1|\ldots|m_i, c_0|\ldots|c_i)$: **var** $\boldsymbol{x_i}$; c_i be a generic encryption mode describe with our language, and let $IO = \{m_1, \ldots, m_i, c_0, \ldots, c_i\}$. If $\{\mathsf{true}\}\mathsf{c}_i \bigwedge_{k=0}^{i} \{\mathsf{Indis}(\nu c_k; IO)\}$ is valid for every i, then $\mathcal{E}_M$ is IND-CPA secure in the Ideal Cipher Model.*

We conclude with the following, which states that our method of proving security of encryption modes is sound in the standard model.

Proposition 3. *Let $\mathcal{E}_M$ be an encryption mode proven secure in the Ideal Cipher Model using the method of Proposition 2. If there exists a standard model algorithm A such that $\mathsf{Adv}_{A,M}^{ind-CPA}(\eta, X)$ is non-negligible, then there exists an algorithm B such that $\mathsf{Adv}_{B,\mathcal{E}}^{prf}$ is non-negligible.*

4 Examples

In this section we apply our method to the traditional encryption modes (CBC), (CFB), (OFB) and (CTR) in respectively Figure 3, 4, 5 and 6. For simplicity, we consider messages consisting of only 3 blocks. The reader can easily be convinced that the same invariant propagation holds for any finite number of blocks. In order to prove IND-CPA security of these encryption schemes we have to prove that $c_0 = IV, c_1, c_2, c_3$ are indistinguishable from random bitstrings when given $m_1, m_2, m_3, c_0, c_1, c_2$ and c_3. Of course our method fails in analyzing ECB encryption mode and the "counter" version of CBC, which are two insecure operation modes.

CBC & CFB : In Figure 3 and 4, we describe the application of our set of rules on CBC and CFB examples. The analysis of these two encryption modes are similar.

OFB : The order of the commands in our description of OFB may seem strange, but it is not without reason. The variable z_{i+1} must be computed before c_i because no rule can preserve the invariant $\mathsf{E}(\mathcal{E}, z_i)$ through the computation of c_i.

CTR : This scheme is the only one of the four encryption modes we have studied that uses the increment command. The analysis is presented in Figure 6. We can see how the *RCounter* invariant is used for proving the IND-CPA security of this mode.

5 Conclusion

We proposed an automatic method for proving the semantic security of symmetric encryption modes. We introduced a small programming language in order to describe these modes. We construct a Hoare logic to make assertions about variables and propagate the assertions with the execution of the commands in the language. If the program which represents an encryption mode satisfies some invariants at the end of our automatic analysis then we conclude that the encryption mode is IND-CPA secure.

Future work: An obvious extension to our work would be to add a loop construct to our grammar. This would remove the necessity of having a different program for each message length within a mode of operation. We are also considering an extension of our work to prove CCA security of encryption modes using approaches such as the one proposed in [10] or the method proposed in [8]. Another more complex and challenging direction is to propose an extended version of our Hoare Logic in order to be able to analyze "modern" encryption modes which use more complex mathematical operation or primitives, or to try to use our method to prove security properties of other block-cipher based construction, such as unforgeability for block-cipher based MACs, or collision-resistance for block-cipher based hash functions.

$\mathcal{E}_{CBC}(m_1|m_2|m_3, IV|c_1|c_2|c_3)$
var IV, z_1, z_2, z_3;

$IV \xleftarrow{\$} \mathcal{U}$;	$\{\mathsf{Indis}(\nu IV; \mathsf{Var}) \wedge F(IV) \wedge \mathsf{E}(\mathcal{E}, IV)\}$	(R1)
$z_1 := IV \oplus m_1$;	$\{\mathsf{Indis}(\nu IV; \mathsf{Var} - z_1) \wedge \mathsf{E}(\mathcal{E}, z_1)\}$	(X2)(X4)
$c_1 := \mathcal{E}(z_1)$;	$\{\mathsf{Indis}(\nu IV; \mathsf{Var} - z_1)$	(B2)
	$\wedge\ \mathsf{Indis}(\nu c_1; \mathsf{Var}) \wedge F(c_1)\}$	(B1)
$z_2 := c_1 \oplus m_2$;	$\{\mathsf{Indis}(\nu IV; \mathsf{Var} - z_1)$	(G1)
	$\wedge\ \mathsf{Indis}(\nu c_1; \mathsf{Var} - z_2) \wedge \mathsf{E}(\mathcal{E}, z_2)\}$	(X2)(X4)
$c_2 := \mathcal{E}(z_2)$;	$\{\mathsf{Indis}(\nu IV; \mathsf{Var} - z_1) \wedge \mathsf{Indis}(\nu c_1; \mathsf{Var} - z_2)$	(B2)
	$\wedge\ \mathsf{Indis}(\nu c_2; \mathsf{Var}) \wedge F(c_2)\}$	(B1)
$z_3 := c_2 \oplus m_3$;	$\{\mathsf{Indis}(\nu IV; \mathsf{Var} - z_1) \wedge \mathsf{Indis}(\nu c_1; \mathsf{Var} - z_2)$	(G1)
	$\wedge\ \mathsf{Indis}(\nu c_2; \mathsf{Var} - z_3) \wedge \mathsf{E}(\mathcal{E}, z_3)\}$	(X2)(X4)
$c_3 := \mathcal{E}(z_3)$;	$\{\mathsf{Indis}(\nu IV; \mathsf{Var} - z_1) \wedge \mathsf{Indis}(\nu c_1; \mathsf{Var} - z_2)$	(B2)
	$\wedge\ \mathsf{Indis}(\nu c_2; \mathsf{Var} - z_3) \wedge Indis(\nu c_3; \mathsf{Var})\}$	(B1)

Fig. 3. Analysis of CBC encryption mode

$\mathcal{E}_{CFB}(m_1|m_2|m_3, IV|c_1|c_2|c_3)$
var IV, z_1, z_2, z_3;

$IV \xleftarrow{\$} \mathcal{U}$;	$\{\mathsf{Indis}(\nu IV) \wedge F(IV) \wedge E(\mathcal{E}, IV)\}$	(R1)
$z_1 := \mathcal{E}(IV)$;	$\{\mathsf{Indis}(\nu IV) \wedge \mathsf{Indis}(\nu z_1) \wedge F(z_1)\}$	(B1)(B2)
$c_1 := z_1 \oplus m_1$;	$\{\mathsf{Indis}(\nu IV) \wedge \mathsf{Indis}(\nu c_1; \mathsf{Var} - z_1) \wedge \mathsf{E}(\mathcal{E}, c_1)\}$	(G1)(X1)(X4)
$z_2 := \mathcal{E}(c_1)$;	$\{\mathsf{Indis}(\nu IV) \wedge \mathsf{Indis}(\nu c_1; \mathsf{Var} - z_1) \wedge F(z_2)\}$	(B1)(B2)
$c_2 := z_2 \oplus m_2$;	$\{\mathsf{Indis}(\nu IV) \wedge \mathsf{Indis}(\nu c_1; \mathsf{Var} - z_1)$	(G1)
	$\wedge\ \mathsf{Indis}(\nu c_2; \mathsf{Var} - z_2) \wedge \mathsf{E}(\mathcal{E}, c_2)\}$	(X1) (X4)
$z_3 := \mathcal{E}(c_2)$;	$\{\mathsf{Indis}(\nu IV) \wedge \mathsf{Indis}(\nu c_1; \mathsf{Var} - z_1)$	(B2)
	$\wedge\ \mathsf{Indis}(\nu c_2; \mathsf{Var} - z_2) \wedge F(z_3)\}$	(B1)
$c_3 := z_3 \oplus m_3$;	$\{\mathsf{Indis}(\nu IV) \wedge \mathsf{Indis}(\nu c_1; \mathsf{Var} - z_1)$	(G1)
	$\wedge\ \mathsf{Indis}(\nu c_2; \mathsf{Var} - z_2)$	(X1)
	$\wedge\ \mathsf{Indis}(\nu c_3; \mathsf{Var} - z_3)\}$	

Fig. 4. Analysis of CFB encryption mode

$\mathcal{E}_{OFB}(m_1|m_2|m_3, IV|c_1|c_2|c_3)$
var IV, z_1, z_2, z_3;

$IV \xleftarrow{\$} \mathcal{U}$;	$\{\mathsf{Indis}(\nu IV; \mathsf{Var}) \wedge F(IV) \wedge \mathsf{E}(\mathcal{E}, IV)\}$	(R1)
$z_1 := \mathcal{E}(IV)$;	$\{\mathsf{Indis}(\nu IV; \mathsf{Var}) \wedge \{F(z_1) \wedge \mathsf{E}(\mathcal{E}, z_1) \wedge \mathsf{Indis}(\nu z_1; \mathsf{Var})\}$	(B1)(B2)
$z_2 := \mathcal{E}(z_1)$;	$\{\mathsf{Indis}(\nu IV; \mathsf{Var}) \wedge \mathsf{Indis}(\nu z_1; \mathsf{Var}) \wedge \mathsf{E}(\mathcal{E}, z_2)$	(B1)(B2)
	$\wedge\ F(z_2) \wedge \mathsf{Indis}(\nu z_2; \mathsf{Var})\}$	
$c_1 := m_1 \oplus z_1$;	$\{\mathsf{Indis}(\nu IV; \mathsf{Var}) \wedge \mathsf{Indis}(\nu c_1; \mathsf{Var} - z_1) \wedge \mathsf{E}(\mathcal{E}, z_2)$	(G1)(G2)(X1)
	$\wedge\ F(z_2) \wedge \mathsf{Indis}(\nu z_2; \mathsf{Var})\}\}$	(G4)
$z_3 := \mathcal{E}(z_2)$;	$\{\mathsf{Indis}(\nu IV; \mathsf{Var}) \wedge \mathsf{Indis}(\nu c_1; \mathsf{Var} - z_1) \wedge \mathsf{E}(\mathcal{E}, z_3)$	(B1)(B2)
	$\wedge\ \mathsf{Indis}(\nu z_2; \mathsf{Var}) \wedge F(z_3) \wedge \mathsf{Indis}(\nu z_3; \mathsf{Var})\}$	(B2)
$c_2 := m_2 \oplus z_2$;	$\{\mathsf{Indis}(\nu IV; \mathsf{Var}) \wedge \mathsf{Indis}(\nu c_1; \mathsf{Var} - z_1)\}$	(G1)
	$\wedge\ \mathsf{Indis}(\nu c_2; \mathsf{Var} - z_2) \wedge \mathsf{Indis}(\nu z_3; \mathsf{Var})$	(X1)
$c_3 := m_3 \oplus z_3$;	$\{\mathsf{Indis}(\nu IV; \mathsf{Var}) \wedge \mathsf{Indis}(\nu c_1; \mathsf{Var} - z_1)\}$	(G1)
	$\wedge\ \mathsf{Indis}(\nu c_2; \mathsf{Var} - z_2) \wedge \mathsf{Indis}(\nu c_3; \mathsf{Var} - z_3)$	(X1)

Fig. 5. Analysis of OFB encryption mode

$\mathcal{E}_{CTR}(m_1|m_2|m_3, IV|c_1|c_2|c_3)$
var IV, z_1, z_2, z_3;

$IV \xleftarrow{\$} \mathcal{U}$;	$\{\mathsf{Indis}(\nu IV; \mathsf{Var}) \wedge F(IV) \wedge \mathsf{E}(\mathcal{E}, IV)\}$	(R1)
$ctr1 := IV + 1$;	$\{\mathsf{Indis}(\nu IV; \mathsf{Var} - ctr1)$	(I3)
	$\wedge\ Rcounter(ctr1) \wedge \mathsf{E}(\mathcal{E}, ctr1)\}$	(I1)
$z_1 := \mathcal{E}(ctr1)$;	$\{\mathsf{Indis}(\nu IV; \mathsf{Var} - ctr1) \wedge Rcounter(ctr1)$	(B2)(B3)
	$\wedge\ F(z_1) \wedge \mathsf{E}(\mathcal{E}, z_1) \wedge \mathsf{Indis}(\nu z_1; \mathsf{Var})\}$	(B1)
$c_1 := m_1 \oplus z_1$;	$\{\mathsf{Indis}(\nu IV; \mathsf{Var} - ctr1) \wedge Rcounter(ctr1)$	(G1)(G3)
	$\wedge\ \mathsf{Indis}(\nu c_1; \mathsf{Var} - z_1)\}$	(X1)
$ctr2 := ctr1 + 1$;	$\{\mathsf{Indis}(\nu IV; \mathsf{Var} - ctr1 - ctr2)$	(I3)
	$\wedge\ \mathsf{Indis}(\nu c_1; \mathsf{Var} - z_1)$	(G1)
	$\wedge\ Rcounter(ctr2) \wedge \mathsf{E}(\mathcal{E}, ctr2)\}$	(I2)
$z_2 := \mathcal{E}(ctr2)$;	$\{\mathsf{Indis}(\nu IV; \mathsf{Var} - ctr1 - ctr2)$	(B2)
	$\wedge\ \mathsf{Indis}(\nu c_1; \mathsf{Var} - z_1) \wedge Rcounter(ctr2)$	(B1)
	$\wedge\ F(z_2) \wedge \mathsf{E}(\mathcal{E}, z_2) \wedge \mathsf{Indis}(\nu z_2; \mathsf{Var})\}$	(B3)
$c_2 := m_2 \oplus z_2$;	$\{\mathsf{Indis}(\nu IV; \mathsf{Var} - ctr1 - ctr2)$	(G1)
	$\wedge\ \mathsf{Indis}(\nu c_1; \mathsf{Var} - z_1) \wedge Rcounter(ctr2)$	(G3)
	$\wedge\ \mathsf{Indis}(\nu c_2; \mathsf{Var} - z_2)\}$	(X1)
$ctr3 := ctr2 + 1$;	$\{\mathsf{Indis}(\nu IV; \mathsf{Var} - ctr1 - ctr2 - ctr3)$	(I3)
	$\wedge\ \mathsf{Indis}(\nu c_1; \mathsf{Var} - z_1) \wedge \mathsf{E}(\mathcal{E}, ctr3)$	(I2)
	$\wedge\ \mathsf{Indis}(\nu c_2; \mathsf{Var} - z_2) \wedge Rcounter(ctr3)\}$	(G1)
$z_3 := \mathcal{E}(ctr3)$;	$\{\mathsf{Indis}(\nu IV; \mathsf{Var} - ctr1 - ctr2 - ctr3)$	(B2)
	$\wedge\ \mathsf{Indis}(\nu c_1; \mathsf{Var} - z_1)$	(B1)
	$\wedge\ \mathsf{Indis}(\nu c_2; \mathsf{Var} - z_2) \wedge Rcounter(ctr3)$	(B3)
	$\wedge\ F(z_3) \wedge \mathsf{E}(\mathcal{E}, z_3) \wedge \mathsf{Indis}(\nu z_3; \mathsf{Var})\}$	
$c_3 := m_3 \oplus z_3$;	$\{\mathsf{Indis}(\nu IV; \mathsf{Var} - ctr1 - ctr2 - ctr3)$	(G1)
	$\wedge\ \mathsf{Indis}(\nu c_1; \mathsf{Var} - z_1)$	(X1)
	$\wedge\ \mathsf{Indis}(\nu c_2; \mathsf{Var} - z_2)$	
	$\wedge\ \mathsf{Indis}(\nu c_3; \mathsf{Var} - z_3)\}$	

Fig. 6. Analysis of CTR encryption mode

References

1. Mihir Bellare, Anand Desai, Eron Jokipii, and Phillip Rogaway. A concrete security treatment of symmetric encryption. *Foundations of Computer Science, Annual IEEE Symposium on*, 0:394, 1997.
2. Mihir Bellare, Joe Kilian, and Phillip Rogaway. The security of the cipher block chaining message authentication code. *J. Comput. Syst. Sci.*, 61(3):362–399, 2000.
3. Mihir Bellare and Phillip Rogaway. Code-based game-playing proofs and the security of triple encryption. Cryptology ePrint Archive, Report 2004/331, 2004. `http://eprint.iacr.org/`.
4. Mihir Bellare, Phillip Rogaway, and David Wagner. The EAX mode of operation. In *FSE*, pages 389–407, 2004.
5. Debrup Chakraborty and Mridul Nandi. An improved security bound for HCTR. pages 289–302, 2008.
6. Debrup Chakraborty and Palash Sarkar. A new mode of encryption providing a tweakable strong pseudo-random permutation. In Matthew J. B. Robshaw, editor, *FSE*, volume 4047 of *Lecture Notes in Computer Science*, pages 293–309. Springer, 2006.
7. Debrup Chakraborty and Palash Sarkar. HCH: A new tweakable enciphering scheme using the hash-counter-hash approach. *IEEE Transactions on Information Theory*, 54(4):1683–1699, 2008.
8. Judicael Courant, Marion Daubignard, Cristian Ene, Pascal Lafourcade, and Yassine Lahknech. Towards automated proofs for asymmetric encryption schemes in the random oracle model. In *Proceedings of the 15th ACM Conference on Computer and Communications Security, (CCS'08)*, Alexandria, USA, October 2008.
9. Judicaël Courant, Cristian Ene, and Yassine Lakhnech. Computationally sound typing for non-interference: The case of deterministic encryption. In Vikraman Arvind and Sanjiva Prasad, editors, *FSTTCS*, volume 4855 of *Lecture Notes in Computer Science*, pages 364–375. Springer, 2007.
10. Anand Desai. New paradigms for constructing symmetric encryption schemes secure against chosen-ciphertext attack. In *CRYPTO '00: Proceedings of the 20th Annual International Cryptology Conference on Advances in Cryptology*, pages 394–412, London, UK, 2000. Springer-Verlag.
11. Martin Gagné, Pascal Lafourcade, Yassine Lakhnech, and Reihaneh Safavi-Naini. Automated security proof for symmetric encryption modes. Manuscript,2009. Available at `http://pages.cpsc.ucalgary.ca/~mgagne/TR_Asian.pdf`,.
12. Shai Halevi. EME*: Extending EME to handle arbitrary-length messages with associated data. In Anne Canteaut and Kapalee Viswanathan, editors, *Progress in Cryptology - INDOCRYPT 2004, 5th International Conference on Cryptology in India, Chennai, India, December 20-22, 2004, Proceedings*, volume 3348 of *Lecture Notes in Computer Science*, pages 315–327. Springer, 2004.
13. Shai Halevi. Invertible universal hashing and the tet encryption mode. In Alfred Menezes, editor, *CRYPTO*, volume 4622 of *Lecture Notes in Computer Science*, pages 412–429. Springer, 2007.
14. Shai Halevi and Phillip Rogaway. A tweakable enciphering mode. In Dan Boneh, editor, *CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 482–499. Springer, 2003.
15. Shai Halevi and Phillip Rogaway. A parallelizable enciphering mode. In Tatsuaki Okamoto, editor, *CT-RSA*, volume 2964 of *Lecture Notes in Computer Science*, pages 292–304. Springer, 2004.

16. Tetsu Iwata and Kaoru Kurosawa. OMAC: One-key CBC MAC. In Thomas Johansson, editor, *FSE*, volume 2887 of *Lecture Notes in Computer Science*, pages 129–153. Springer, 2003.
17. Tetsu Iwata and Kaoru Kurosawa. On the security of a new variant of OMAC. In Jong In Lim and Dong Hoon Lee, editors, *ICISC*, volume 2971 of *Lecture Notes in Computer Science*, pages 67–78. Springer, 2003.
18. Tetsu Iwata and Kaoru Kurosawa. Stronger security bounds for OMAC, TMAC, and XCBC. In Thomas Johansson and Subhamoy Maitra, editors, *INDOCRYPT*, volume 2904 of *Lecture Notes in Computer Science*, pages 402–415. Springer, 2003.
19. Charanjit S. Jutla. Encryption modes with almost free message integrity. In *EUROCRYPT '01: Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques*, pages 529–544, London, UK, 2001. Springer-Verlag.
20. Kaoru Kurosawa and Tetsu Iwata. TMAC: Two-key CBC MAC. In Marc Joye, editor, *CT-RSA*, volume 2612 of *Lecture Notes in Computer Science*, pages 33–49. Springer, 2003.
21. Éliane Jaulmes, Antoine Joux, and Frédéric Valette. On the security of randomized CBC-MAC beyond the birthday paradox limit - a new construction. In *Fast Software Encryption Ç02, Lecture Notes in Computer Science*, pages 237–251. Springer-Verlag, 2001.
22. Moses Liskov, Ronald L. Rivest, and David Wagner. Tweakable block ciphers. In *CRYPTO '02: Proceedings of the 22nd Annual International Cryptology Conference on Advances in Cryptology*, pages 31–46, London, UK, 2002. Springer-Verlag.
23. David A. McGrew and Scott R. Fluhrer. The security of the extended codebook (XCB) mode of operation, 2007.
24. David A. McGrew and John Viega. The security and performance of the galois/counter mode (GCM) of operation. In *INDOCRYPT*, pages 343–355, 2004.
25. Phillip Rogaway. Efficient instantiations of tweakable blockciphers and refinements to modes OCB and PMAC. In Pil Joong Lee, editor, *ASIACRYPT*, volume 3329 of *Lecture Notes in Computer Science*, pages 16–31. Springer, 2004.
26. Phillip Rogaway, Mihir Bellare, John Black, and Ted Krovetz. OCB: a block-cipher mode of operation for efficient authenticated encryption. In *CCS '01: Proceedings of the 8th ACM conference on Computer and Communications Security*, pages 196–205, New York, NY, USA, 2001. ACM.
27. Victor Shoup. Sequences of games: a tool for taming complexity in security proofs, 2004. http://eprint.iacr.org/2004/332.
28. Peng Wang, Dengguo Feng, and Wenling Wu. On the security of tweakable modes of operation: TBC and TAE. In Jianying Zhou, Javier Lopez, Robert H. Deng, and Feng Bao, editors, *ISC*, volume 3650 of *Lecture Notes in Computer Science*, pages 274–287. Springer, 2005.