Automated Proofs for Asymmetric Encryption: First results in the random oracle model

J. Courant, M. Daubignard, C. Ene, P. Lafourcade and Y. Lakhnech *
Université Grenoble 1, CNRS, VERIMAG

June 9, 2008

Abstract

Chosen-ciphertext security is by now a standard security property for asymmetric encryption. Many generic constructions for building secure cryptosystems from primitives with lower level of security have been proposed. Providing security proofs has also become standard practice. There is, however, a lack of automated verification procedures that analyse such cryptosystems and provide security proofs. This paper presents an automated procedure for analysing generic asymmetric encryption schemes in the random oracle model. It has been applied to several examples of encryption schemes.

1 Introduction

Our day-to-day lives increasingly depend upon information and our ability to manipulate it securely. This requires solutions based on cryptographic systems (primitives and protocols). In 1976, Diffie and Hellman invented public-key cryptography, coined the notion of one-way functions and discussed the relationship between cryptography and complexity theory. Shortly after, the first cryptosystem with a reductionist security proof appeared (Rabin 1979). The next breakthrough towards formal proofs of security was the adoption of computational theory for the purpose of rigorously defining the security of cryptographic schemes. In this framework, a system is *provably secure* if there is a polynomial-time reduction proof from a hard problem to an attack against the security of the system. The provable security framework has been later refined into *the exact (also called concrete) security framework* where better estimates of the computational complexity of attacks is achieved. Provable cryptography has become a very active field of research and public-key cryptography is probably one of the most active topics (An objective analysis of what it is about and what it is not about can be found in [12].) Yet, there is a problem with cryptographic proofs, as expressed by S. Halevi in [17], A. Dent [14], J. Stern et al [23] and others. While research in the

*Grenoble, email:name@imag.fr This work has been partially supported by the ANR project SCALP, SFINCS, AVOTE

field of provable cryptography has achieved tremendous progress towards rigorously defining the functionalities and requirements of many cryptosystems, little has been done for developing computer-aided proof methods or more generally for investigating a proof theory for cryptosystems as it exists for imperative programs, concurrent systems, reactive systems, etc...

In this paper, we present an automated proof method for analyzing generic asymmetric encryption schemes in the random oracle model (ROM). Generic encryption schemes aim at transforming schemes with weak security properties, such as one-wayness, into schemes with stronger security properties, specifically security against chosen ciphertext attacks. Examples of generic encryption schemes are [11, 25, 5, 6, 19, 18, 22, 20]. The paper contains two main parts. The first one presents a compositional Hoare logic for proving IND-CPA-security. The second part presents a method for proving plaintext awareness (PA). Hence, altogether we have a proof method for IND-CCA security, that applies for instance to the constructions in [5, 18, 19]. An 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 [7, 21]. Indeed, both approaches can be considered complementary as the Hoare logic-based one can be considered as aiming at characterizing by means of predicates the set of contexts in which the game transformations can be applied safely.

Our automated verification method has been implemented in CAML [1] and applied to several examples.

Related work: We restrict our discussion to work aiming at providing computational proofs for cryptosystems. In particular, this excludes symbolic verification (including ours). We mentioned above the game-based approach [7, 21, 17]. B. Blanchet and D. Pointcheval developed a dedicated tool, CryptoVerif, that supports security proofs within the game-based approach [8, 9]. From the theoretical point of view, the main differences in our approaches are the following. CryptoVerif is based on observational equivalence. The equivalence relation induces rewriting rules applicable in contexts that satisfy some properties. Invariants provable in our Hoare logic can be considered as logical representations of these contexts. Moreover, as we are working with invariants, that is we follow a state-based approach, we need to prove results that link our invariants to game-based properties such as indistinguishability (cf. Proposition 1 and 3). G. Barthe and S. Tarento were among the first to provide machine-checked proofs of cryptographic schemes without relying on the perfect cryptography hypothesis. They have provided formal models of the Generic Model and the Random Oracle Model in the Coq proof assistant, and used this formalisation to prove hardness of the discrete logarithm [1], security of signed ElGamal encryption against interactive attacks [3], and of Schnorr signatures against forgery attacks [24]. They are currently working on formalizing the game-based approach in Coq [2]. Another interesting work to mention is the Hoare-style proof system proposed by R. Corin and J. Den Hartog for game-based cryptographic proofs [10]. Yet, there is no computer-assistance for the developed logic. In [13], Datta et al. present a computationally sound compositional logic for key exchange protocols. There is, however, no proof assistance provided for this logic neither.

[1] The implementation can be downloaded at *http://www-verimag.imag.fr/ lakhnech/checker.ml*

Outline: In Section 2, we introduce notations used for defining our programming language and generic asymmetric encryption schemes. In Section 3, we present our method for proving IND-CPA security. In Section 4 we introduce a criterion to prove plaintext awareness. Finally, in Section 5 we conclude.

2 Definitions

We are interested in analysing generic schemes for asymmetric encryption assuming ideal hash functions. That is, we are working in the *random oracle model* [15, 5]. Using standard notations, we write $H \stackrel{r}{\leftarrow} \Omega$ to denote that H is randomly chosen from the set of functions with appropriate domain. By abuse of notation, for a list $\vec{H} = H_1, \cdots, H_n$ of hash functions, we write $\vec{H} \stackrel{r}{\leftarrow} \Omega$ instead of the sequence $H_1 \stackrel{r}{\leftarrow} \Omega \cdots, H_n \stackrel{r}{\leftarrow} \Omega$. We also fix a finite set Π of trapdoor permutations and a finite set $\mathcal{H} = \{H_1, \cdots, H_n\}$ of hash functions and $\mathcal{O} = \Pi \cup \mathcal{H}$. We assume an arbitrary but fixed ordering on Π and $\mathcal{H}$; just to be able to switch between set-based and vector-based notation. A *distribution ensemble* is a countable sequence of distributions $\{X_\eta\}_{\eta \in \mathbb{N}}$. We only consider distribution ensembles that can be constructed in polynomial-time by probabilistic algorithms that have oracle access to $\mathcal{O}$.

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:

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

We insist, above, that for each hash function H, the probabilities are also taken over the set of functions with the appropriate type. Let $\mathsf{Adv}(\eta, X, X') = \sup_{\mathcal{A}}(\mathsf{Adv}(\mathcal{A}, \eta, X, X'))$ be the maximal advantage taken over all probabilistic polynomial-time algorithms. Then, two distribution ensembles X and X' are called *indistinguishable*, denoted by $X \sim X'$, if $\mathsf{Adv}(\eta, X, X')$ is negligible as a function of η. In other words, for any polynomial-time (in η) probabilistic algorithm $\mathcal{A}$, $\mathsf{Adv}(\mathcal{A}, \eta, X, X')$ is negligible as a function of η. We insist that all security notions we are going to use are in the ROM, where all algorithms, including adversaries, are equipped with oracle access to the hash functions.

2.1 A simple programming language for encryption and decryption oracles

We introduce a notation (a simple programming language) in which the encryption and decryption oracles are specified. The motivation for fixing a notation is obvious: it is mandatory for developing an automatic verification procedure. Let Var be an arbitrary finite non-empty set of variables. Then, our programming language is built according to the following BNF described in Figure 1, where for a bit-string $bs = b_1 \cdots b_k$ (b_i are bits), $bs[n,m] = b_n \cdots b_m$[2], and $\mathcal{N}$ is the name of the oracle, $\mathtt{c}$ its body and x and y are the input and output variable respectively. Commands are standard, where $x \stackrel{r}{\leftarrow} \mathcal{U}$

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

means that the value of x is randomly sampled following the uniform distribution on the appropriate domain, $\oplus$ is the bitwise-xor operation and $||$ is the string concatenation.

$$\begin{array}{rrl} \text{Command} \quad \mathsf{c} & ::= & x \stackrel{r}{\leftarrow} \mathcal{U} \mid x := f(y) \mid x := f^{-1}(y) \mid x := H(y) \mid x := y[n,m] \\ & & \mid x := y \oplus z \mid x := y||z \mid \mathsf{if}\ x = y\ \mathsf{then\ c1\ else\ c2\ fi} \mid \mathsf{c;c} \\ \text{Oracle} \quad \mathcal{O} & ::= & \mathcal{N}(x,y) : \mathsf{c} \end{array}$$

Figure 1: **Language grammar.**

Example 1 *The following command encodes the encryption scheme proposed by Bellare and Rogaway in [5] (shortly $\mathcal{E}(in_e;r) = f(r)||in_e \oplus G(r)||H(in_e||r)$):*
$\mathcal{E}(in_e, out_e):\ \ r \stackrel{r}{\leftarrow} \{0,1\}^{\eta_0};\ a := f(r);\ g := G(r);\ b := in_e \oplus g;\ s := in_e||r;$
$c := H(s);\ u := a||b||c;\ out_e := u;$ *where, $f \in \Pi$ and $G, H \in \mathcal{H}$.*

Semantics: In addition to variables in Var, we consider variables $\mathbb{T}_{H_1}, \dots, \mathbb{T}_{H_n}$. Variable $\mathbb{T}_{H_i}$ records the queries to the hash function H_i and *can not be accessed by the adversary*. Thus, we consider states that assign bit-strings to the variables in Var and lists of pairs of bit-strings to $\mathbb{T}_{H_i}$.

A *state* associates a value in $\{0,1\}^*$ to each variable in Var and a list of pairs of values to $\mathbb{T}_H$. For simplicity of the presentation, we assume that all variables range over large domains, whose cardinalities are exponential in the security parameter η. Given a state S, $S(\mathbb{T}_H).\mathsf{dom}$, respectively $S(\mathbb{T}_H).\mathsf{res}$, denotes the list obtained by projecting each pair in $S(\mathbb{T}_H)$ to its first, respectively second, element. A program takes as input a *configuration* $(S, \vec{H}, (f, f^{-1}))$ and yields a distribution on configurations. A configuration is composed of a state S, a vector of hash functions $(H_1, \cdots, H_n)$ and a pair (f, f^{-1}) of a trapdoor permutation and its inverse. Let Γ denote the set of configurations and $\mathrm{DIST}(\Gamma)$ the set of distributions on configurations. The semantics is given in Figure 2, where $\delta(x)$ denotes the Dirac measure, i.e. $\mathsf{Pr}(x) = 1$. Notice that the semantic function of commands can be lifted in the usual way to a function from $\mathrm{DIST}(\Gamma)$ to $\mathrm{DIST}(\Gamma)$. By abuse of notation we also denote the lifted semantics by $[\![\mathsf{c}]\!]$.

A notational convention: It is easy to prove that commands preserve the values of $\vec{H}$ and (f, f^{-1}). Therefore, we can, without ambiguity, write $S' \stackrel{r}{\leftarrow} [\![c]\!](S, \vec{H}, (f, f^{-1}))$ instead of $(S', \vec{H}, (f, f^{-1})) \stackrel{r}{\leftarrow} [\![c]\!](S, \vec{H}, (f, f^{-1}))$. According to our semantics, commands have as denotations functions that transform distributions on configurations to distributions on configurations. However, only distributions that are constructible are of interest. Their set is denoted by $\mathrm{DIST}(\Gamma, \vec{H}, \mathbb{F})$ and is defined as the set of distributions of the form: $[(f, f^{-1}) \stackrel{r}{\leftarrow} \mathbb{F}(1^\eta); \vec{H} \stackrel{r}{\leftarrow} \Omega; S \stackrel{r}{\leftarrow} A^{\vec{H}, f, f^{-1}}() : (S, \vec{H}, f, f^{-1})]$, where A is an probabilistic polynomial-time algorithm accessing f, f^{-1} and $\vec{H}$ and which records its queries to hashing oracles into the $\mathbb{T}_H$'s in S, belongs to $\mathrm{DIST}(\Gamma, \vec{H}, \mathbb{F})$.

2.2 Asymmetric Encryption

We are interested in generic constructions that convert any trapdoor permutation scheme into a public-key encryption scheme. More specifically, our aim is to provide an automatic verification method for generic encryption schemes. We also adapt IND-CPA and IND-CCA security notions to our setting.

$$
\begin{aligned}
&[\![x \xleftarrow{r} \mathcal{U}]\!](S,\vec{H},(f,f^{-1})) = [u \xleftarrow{r} \mathcal{U} : (S\{x \mapsto u\},\vec{H},(f,f^{-1}))]\\
&[\![x := f(y)]\!](S,\vec{H},(f,f^{-1})) = \delta(S\{x \mapsto f(S(y))\},\vec{H},(f,f^{-1}))\\
&[\![x := f^{-1}(y)]\!](S,\vec{H},(f,f^{-1})) = \delta(S\{x \mapsto f^{-1}(S(y))\},\vec{H},(f,f^{-1}))\\
&[\![x := y[n,m]]\!](S,\vec{H},(f,f^{-1})) = \delta(S\{x \mapsto S(y)[n,m]\},\vec{H},(f,f^{-1}))\\
&[\![x := H(y)]\!](S,\vec{H},(f,f^{-1})) =\\
&\quad\begin{cases}
\delta(S\{x \mapsto v\},\vec{H},(f,f^{-1})) & ;\text{if } (S(y),v) \in \mathbb{T}_H\\
\delta(S\{x \mapsto v, \mathbb{T}_H \mapsto S(\mathbb{T}_H)\cdot(S(y),v)\},\vec{H},(f,f^{-1}))\,; & \\
\qquad\qquad \text{if } (S(y),v) \notin \mathbb{T}_H \text{ and } v = \vec{H}(H)(S(y)) &
\end{cases}
\end{aligned}
$$

$$
\begin{aligned}
&[\![x := y \oplus z]\!](S,\vec{H},(f,f^{-1})) = \delta(S\{x \mapsto S(y) \oplus S(z)\},\vec{H},(f,f^{-1}))\\
&[\![x := y||z]\!](S,\vec{H},(f,f^{-1})) = \delta(S\{x \mapsto S(y)||S(z)\},\vec{H},(f,f^{-1}))\\
&[\![c_1;c_2]\!] = [\![c_2]\!] \circ [\![c_1]\!]
\end{aligned}
$$

$$
[\![\text{if } x \text{ then } \mathsf{c}_1 \text{ else } \mathsf{c}_2 \text{ fi}]\!](S,\vec{H},(f,f^{-1})) = \begin{cases} [\![\mathsf{c}_1]\!](S,\vec{H},(f,f^{-1})) & \text{if } S(x) = 1\\ [\![\mathsf{c}_2]\!](S,\vec{H},(f,f^{-1})) & \text{otherwise}\end{cases}
$$

$$
[\![\mathcal{N}(v)]\!](S,\vec{H},(f,f^{-1})) = [\![c]\!](S\{x \mapsto v\},\vec{H},(f,f^{-1})), \text{ where } c \text{ is the body of } \mathcal{N}.
$$

Figure 2: **The semantics of the programming language**

Definition 1 *A* generic encryption scheme *is a triple* $(\mathbb{F}, \mathcal{E}(in_e, out_e) : \mathsf{c}, \mathcal{D}(in_d, out_d) : \mathsf{c}')$*:*
1. $\mathbb{F}$ *is a* trapdoor permutation generator *that on input* η *generates an* η*-bit string trapdoor permutation* (f, f^{-1})
2. $\mathcal{E}(in_e, out_e) : \mathsf{c}$ *and* $\mathcal{D}(in_d, out_d) : \mathsf{c}'$ *are oracle declarations.* □

Definition 2 *Let* $GE = (\mathbb{F}, \mathcal{E}(in_e, out_e) : \mathsf{c}, \mathcal{D}(in_d, out_d) : \mathsf{c}')$ *be a generic encryption scheme. Let* $A = (A_1, A_2)$ *be an adversary and* $X \in \mathrm{DIST}(\Gamma, \vec{H}, \mathbb{F})$*. For* $\alpha \in \{cpa, cca\}$ *and* $\eta \in \mathbb{N}$*, let*

$$
\begin{aligned}
\mathsf{Adv}^{ind-\alpha}_{A,GE}(\eta, X) = 2 * \mathsf{Pr}[&(S,\vec{H},(f,f^{-1})) \xleftarrow{r} X; (x_0,x_1,s) \xleftarrow{r} A_1^{O_1}(f); b \xleftarrow{r} \{0,1\};\\
&S' \xleftarrow{r} [\![\mathcal{E}(x_b)]\!](S,\vec{H},(f,f^{-1})) : A_2^{O_2}(x_0,x_1,s,S'(out_e)) = b] - 1
\end{aligned}
$$

where if $\alpha = cpa$ *then* $O_1 = O_2 = \vec{H}$ *and if* $\alpha = cca$ *then* $O_1 = O_2 = \vec{H} \cup \{\mathcal{D}\}$*. We insist, above, that* A_1 *outputs* x_0, x_1 *such that* $|x_0| = |x_1|$ *and that in the case of CCA,* A_2 *does not ask its oracle* $\mathcal{D}$ *to decrypt* $S'(y)$*. We say that GE is IND-*α *secure if* $\mathsf{Adv}^{ind-\alpha}_{A,GE}(\eta, S)$ *is negligible for any state S and polynomial-time adversary A.* □

3 Verification of IND-CPA security

In this section, we present an effective procedure to verify IND-CPA security. The procedure may fail to prove a secure encryption scheme but never declares correct an insecure one. Thus, we sacrifice completeness for soundness, a situation very frequent in verification[3]. We insist that our procedure does not fail for any of the numerous

[3] We conjecture that the IND-CPA verification problem of schemes described in our language is undecidable.

constructions we tried.

We are aiming at developing a procedure that allows us to prove properties, i.e. invariants, of the encryption oracle. More precisely, the procedure annotates each control point of the encryption command with a set of predicates that hold at that point for any execution except with negligible probability. Given an encryption oracle $\mathcal{E}(\text{in}_e, \text{out}_e) : \mathsf{c}$ we want to prove that at the final control point, we have an invariant that tells us that the value of out_e is indistinguishable from a random value. As we will show, this implies IND-CPA security.

A few words now concerning how we present the verification procedure. First, we present the invariant properties we are interested in the assertion language. Then, we present a set of rules of the form $\{\varphi\}c\{\varphi'\}$ 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 $\{\varphi\}c\{\varphi'\}$ is valid.

3.1 The Assertion Language

Our assertion language is defined by the following grammar, where ψ defines the set of atomic assertions: $\psi ::= \mathsf{Indis}(\nu x; V_1; V_2) \mid \mathsf{WS}(x; V) \mid \mathsf{H}(H, e)$ and $\varphi ::= \mathsf{true} \mid \psi \mid \varphi \wedge \varphi$ where $V_1, V_2 \subseteq \mathsf{Var}$ and e is either a variable in Var or an expression $x||y$ with $x, y \in \mathsf{Var}$.

Intuitively, $\mathsf{Indis}(\nu x; V_1; V_2)$ is satisfied by a distribution on configurations, if any adversary has negligible probability to distinguish whether he is given the value of x or a random value, even when he is additionally given the values of the variables in V_1 and the image by the one-way permutation of those in V_2. The assertion $\mathsf{WS}(x; V)$ is satisfied by a distribution, if any adversary has negligible probability to compute the value of x, even when he is given the values of the variables in V. Finally, $\mathsf{H}(H, e)$ is satisfied, when the value of e has not been submitted to the hash oracle H.

Notations: We use $\mathsf{Indis}(\nu x; V)$ instead of $\mathsf{Indis}(\nu x; V; \emptyset)$ and $\mathsf{Indis}(\nu x)$ instead of $\mathsf{Indis}(\nu x; \mathsf{Var})$. We also write V, x instead of $V \cup \{x\}$ and even x, y instead of $\{x, y\}$.

Formally, the meaning of the assertion language is defined by a satisfaction relation $X \models \varphi$, which tells us when a distribution on configurations X satisfies the assertion φ. In order to define the satisfaction relation $X \models \varphi$, we need to generalize indistinguishability as follows. Let X be a family of distributions in $\textsc{Dist}(\Gamma)$ and V_1 and V_2 be sets of variables in Var. By $D(X, V_1, V_2)$ we denote the following distribution family (on tuples of bit-strings):

$D(X, V_1, V_2)_\eta = [(S, \vec{H}, (f, f^{-1})) \xleftarrow{r} X : (S(V_1), f(S(V_2)), \vec{H}, f)]$

Here $S(V)$ is the pointwise application of S to the elements of V and $f(S(V_2))$ is the pointwise application of f to the elements of $S(V_2)$. We say that X and X' are $V_1; V_2$-*indistinguishable*, denoted by $X \sim_{V_1;V_2} X'$, if $D(X, V_1, V_2) \sim D(X', V_1, V_2)$.

Example 2 *Let S_0 be any state and let H_1 be a hash function. Recall that we are working in the ROM. Consider the following distributions:*

$X_\eta = [\beta; S := S_0\{x \mapsto u, y \mapsto H_1(u)\} : (S, \vec{H}, (f, f^{-1}))]$ *and* $X'_\eta = [\beta; u' \xleftarrow{r} \{0,1\}^{n(\eta)}; S := S_0\{x \mapsto u, y \mapsto H_1(u')\} : (S, \vec{H}, (f, f^{-1}))]$*, where* $\beta = \vec{H} \xleftarrow{r} \Omega; (f, f^{-1}) \xleftarrow{r} \mathbb{F}(1^\eta); u \xleftarrow{r} \{0,1\}^{n(\eta)}$*. Then, we have $X \sim_{\{y\};\{x\}} X'$ but we do not have $X \sim_{\{y,x\}} X'$.* □

The satisfaction relation $X \models \psi$ is defined as follows:
1. $X \models \mathsf{true}$, $X \models \varphi \wedge \varphi'$ iff $X \models \varphi$ and $X \models \varphi'$.

2. $X \models \mathsf{Indis}(\nu x; V_1; V_2)$ iff $X \sim_{V_1;V_2} [u \xleftarrow{r} \mathcal{U}; (S, \vec{H}, (f, f^{-1})) \xleftarrow{r} X : (S\{x \mapsto u\}, \vec{H}, (f, f^{-1}))]$.
3. $X \models \mathsf{WS}(x; V)$ iff $\mathsf{Pr}[(S, \vec{H}, (f, f^{-1})) \xleftarrow{r} X : A(S(V)) = S(x)]$ is negligible, for any polynomial-time adversary A.
4. $X \models \mathsf{H}(H, e)$ iff $\mathsf{Pr}[(S, \vec{H}, (f, f^{-1})) \xleftarrow{r} X : S(e) \in S(\mathbb{T}_H).\mathsf{dom}]$ is negligible.

The relation between our Hoare triples and public key security is established by the following proposition that states that, if the value of out_e is indistinguishable from a random value then *GE* is IND-CPA.

Proposition 1 *Let GE be a generic encryption scheme. If* $\{\mathit{true}\}c\{\mathit{Indis}(\nu out_e; out_e, in_e)\}$ *is valid then GE is IND-CPA secure.*

3.2 The Hoare Logic

In this section we present our Hoare logic for IND-CPA security. We begin with a set of preservation axioms that tell us when an invariant established at the control point before a command can be transferred to the control point after the commands. Then, for each command, we present a set of specific axioms.

3.2.1 Generic preservation rules:

We assume $z \neq x$ and c is $x \xleftarrow{r} \mathcal{U}$, $x := y_1 || y_2$, $x = y \oplus t$, $x := f(y)$ or $x := H(y)$.

Lemma 1 *The following axioms are sound, provided* $x \notin V_1 \cup V_2$ *and* $x \notin V$ *for (G2):*
(G1) $\{\mathit{Indis}(\nu z; V_1; V_2)\}\ c\ \{\mathit{Indis}(\nu z; V_1; V_2)\}$
(G2) $\{\mathit{WS}(z; V)\}\ c\ \{\mathit{WS}(z; V)\}$
(G3) $\{\mathit{H}(H', e)\}\ c\ \{\mathit{H}(H', e)\}$, *if* $x \notin \mathit{var}(e) \wedge H' \neq H$

3.2.2 Random Assignment:

Consider the command $\mathsf{c} \equiv x \xleftarrow{r} \mathcal{U}$.

Lemma 2 *The following axioms are sound:*
(R1) $\{\mathit{true}\}\ c\ \{\mathit{Indis}(\nu x)\}$
(R2) $\{\mathit{true}\}\ c\ \{\mathit{H}(H, e)\}$ *if* $x \in \mathit{var}(e)$

Lemma 3 *The following preservation axioms, where we assume* $x \neq y$ [4]*, are sound:*
(R3) $\{\mathit{Indis}(\nu y; V_1; V_2)\}c\{\mathit{Indis}(\nu y; V_1, x; V_2)\}$
(R4) $\{\mathit{WS}(y; V)\}c\{\mathit{WS}(y; V, x)\}$

3.2.3 Hash Function:

Lemma 4 *The following basic axioms are sound, for* $x \neq y$*:*
(H1) $\{\mathit{WS}(y; V) \wedge \mathit{H}(H, y)\}x := H(y)\{\mathit{Indis}(\nu x; V, x)\}$
(H2) $\{\mathit{H}(H, y)\}\ x := H(y)\{\mathit{H}(H', e)\}$ *if* $x \in \mathit{var}(e)$
(H3) $\{\mathit{Indis}(\nu y; V; V', y) \wedge \mathit{H}(H, y)\}x := H(y)\{\mathit{Indis}(\nu x; V, x; V', y)\}$ *if* $y \notin V$

[4] By $x = y$ we mean syntactic equality.

The following preservation axioms are sound provided $x \neq y$ and $z \neq x$:
(H4) $\{\mathsf{WS}(y;V) \wedge \mathsf{WS}(z;V) \wedge \mathsf{H}(H,y)\}x := H(y)\{\mathsf{WS}(z;V,x)\}$
(H5) $\{\mathsf{H}(H,e) \wedge \mathsf{WS}(z;y)\}x := H(y)\{\mathsf{H}(H,e)\}$, if $z \in \mathsf{var}(e) \wedge x \notin \mathsf{var}(e)$
(H6) $\{\mathsf{Indis}(\nu y;V_1;V_2,y) \wedge \mathsf{H}(H,y)\}x := H(y)\{\mathsf{Indis}(\nu y;V_1,x;V_2,y)\}$, if $y \notin V_1$
(H7) $\{\mathsf{Indis}(\nu z;V_1,z;V_2) \wedge \mathsf{WS}(y;V_1 \cup V_2,z) \wedge \mathsf{H}(H,y)\}x := H(y)\{\mathsf{Indis}(\nu z;V_1,z,x;V_2)\}$

3.2.4 One-way Function:

Lemma 5 *The following axiom is sound:*
(O1) $\{\mathit{Indis}(\nu y;V;y)\}\ x := f(y)\ \{\mathit{WS}(y;V,x)\}$, *if* $(y \notin V \cup \{x\})$

Lemma 6 *The following axioms are sound for $z \neq x$:*
(O2) $\{\mathit{Indis}(\nu z;V_1,z;V_2,y)\}\ x := f(y)\ \{\mathit{Indis}(\nu z;V_1,z,x;V_2)\}$, *if* $z \neq y$ (O3) $\{\mathit{WS}(z;V) \wedge \mathit{Indis}(\nu y;V,z;y)\}\ x := f(y)\ \{\mathit{WS}(z;V,x)\}$ *For one-way permutations, we additionally have the following axiom:*
(P1)$\{\mathit{Indis}(\nu y;V_1;V_2,y)\}\ x := f(y)\ \{\mathit{Indis}(\nu x;V_1,x;V_2)\}$, *if* $y \notin V_1 \cup V_2$

3.2.5 The Xor operator

In the following axioms, we assume $y \neq z$.

Lemma 7 *The following axiom is sound:*
(X1) $\{\mathit{Indis}(\nu y;V_1,y,z;V_2)\}\ x := y \oplus z\ \{\mathit{Indis}(\nu x;V_1,x,z;V_2)\}$, *if* $y \notin V_1 \cup V_2$

Lemma 8 *The following axioms are sound provided $t \neq x,y,z$.*
(X2) $\{\mathit{Indis}(\nu t;V_1,y,z;V_2)\}\ x := y \oplus z\ \{\mathit{Indis}(\nu t;V_1,x,y,z;V_2)\}$
(X3) $\{\mathit{WS}(t;V,y,z)\}\ x := y \oplus z\ \{\mathit{WS}(t;V,y,z,x)\}$

3.2.6 Concatenation:

Lemma 9 *The following axioms are sound:*
(C1) $\{\mathit{WS}(y;V)\}\ x := y||z\ \{\mathit{WS}(x;V)\}$, *if* $x \notin V$. *A dual axiom applies for z.*
(C2) $\{\mathit{Indis}(\nu y;V_1,y,z;V_2) \wedge \mathit{Indis}(\nu z;V_1,y,z;V_2)\}\ x := y||z\ \{\mathit{Indis}(\nu x;V_1;V_2)\}$, *if* $y,z \notin V_1 \cup V_2$
(C3) $\{\mathit{H}(H,y||z)\}\ x := y||z\ \{\mathit{H}(H,x)\}$
(C4) $\{\mathit{Indis}(\nu t;V_1,y,z;V_2)\}\ x := y||z\ \{\mathit{Indis}(\nu t;V_1,x,y,z;V_2)\}$, *if* $t \neq x,y,z$
(C5) $\{\mathit{WS}(t;V,y,z)\}\ x := y||z\ \{\mathit{WS}(t;V,y,z,x)\}$, *if* $t \neq x,y,z$

In addition to the axioms above, we have the usual Sequential composition and Consequence rules of the Hoare logic. In order to apply the Consequence rule, we use entailment (logic implication) between assertions as in Lemma 10.

Lemma 10 *Let X be a distribution ensemble in* $\mathrm{DIST}(\Gamma,\vec{H},\mathbb{F})$.

1. *If* $X \models \mathit{Indis}(\nu x;V_1;V_2)$, $V_1' \subseteq V_1$ *and* $V_2' \subseteq V_1 \cup V_2$ *then* $X \models \mathit{Indis}(\nu x;V_1';V_2')$.
2. *If* $X \models \mathit{WS}(x;V')$ *and* $V \subseteq V'$ *then* $X \models \mathit{WS}(x;V)$.
3. *If* $X \models \mathit{Indis}(\nu x;V_1;V_2 \cup \{x\})$ *and* $V \subseteq V_1$ *then* $X \models \mathit{WS}(x;V)$.

The soundness of the Hoare Logic follows by induction from the soundness of each axiom and soundness of the Consequence and Sequential composition rules.

Proposition 2 *The Hoare triples given in Section 3.2 are valid.*

3.3 Extensions

In this section, we show how our Hoare logic, and hence our verification procedure, can be adapted to deal with on one hand injective partially trapdoor one-way functions and on the other hand OW-PCA (probabilistic) functions. The first extension is motivated by Pointcheval's construction in [19] and second by the Rapid Enhanced-security Asymmetric Cryptosystem Transform (REACT) [18]. For obvious reasons, we cannot recall the definitions of the security of these functions; we explain them informally. The first observation we have to make is that Proposition 1 is too demanding in case we do not assume trapdoor permutations. Therefore, we introduce a new predicate $\mathsf{Indis}_f(\nu x;V_1;V_2)$ whose meaning is as follows: $X \models \mathsf{Indis}_f(\nu x;V_1;V_2)$ iff $X \sim_{V_1;V_2} [u \xleftarrow{r} \mathcal{U};(S,\vec{H},(f,f^{-1})) \xleftarrow{r} X : (S\{x \mapsto f(u)\},\vec{H},(f,f^{-1}))]$. Notice that $\mathsf{Indis}_f(\nu x;V_1;V_2)$ is equivalent to $\mathsf{Indis}(\nu x;V_1;V_2)$, when f is a bijection. Now, let out_e, the output of the encryption oracle, have the form $a_1||\cdots||a_n$ with $a_i = f_i(x_i)$ (f_i can be the identity function). Then, we can prove the following:

Proposition 3 *Let $GE = (\mathbb{F},\mathcal{E}(in_e,out_e) : \mathsf{c},\mathcal{D}(in_d,out_d) : \mathsf{c}')$ be a generic encryption scheme. If $\{true\}\mathsf{c}\{\bigwedge_{i=1}^{n} \mathsf{Indis}_{f_i}(\nu a_i;a_1,\ldots,a_n,in_e)\}$ is valid then GE is IND-CPA.*

Now, we introduce a new axiom for $\mathsf{Indis}_f(\nu x;V_1;V_2)$ that replaces axiom (P1) in case the one-way function f is not a permutation:

$(P1')$ $\{\mathsf{Indis}(\nu y;V_1;V_2,y)\}\ x := f(y)\ \{\mathsf{Indis}_f(\nu x;V_1,x;V_2)\}$ if $y \notin V_1 \cup V_2$

Injective partially trapdoor one-way functions: In contrast to the previous section, we do not assume f to be a permutation. On the other hand, we demand a stronger property than one-wayness. Let $f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$ be a function and let $f^{-1} : \mathcal{Z} \to \mathcal{X}$ be such that $\forall z \in \mathsf{dom}(f^{-1})\exists y \in \mathcal{Y},\ z = f(f^{-1}(z),y)$. Here f^{-1} is a partial function. The function f is said *partially one-way*, if for any given $z = f(x,y)$, it is computationally impossible to compute a corresponding x. In order to deal with the fact that f is now partially one-way, we add the following axioms, where we assume $x,y \notin V \cup \{z\}$ and where we identify f and $(x,y) \mapsto f(x||y)$:

(PO1) $\{\mathsf{Indis}(\nu x;V,x,y) \wedge \mathsf{Indis}(\nu y;V,x,y)\}\ z := f(x||y)\ \{\mathsf{WS}(x;V,z)\}$ }.

(PO2) $\{\mathsf{Indis}(\nu x;V,x,y) \wedge \mathsf{WS}(y;V,x) \wedge \mathsf{H}(H,y)\}\ z := f(x||H(y))\ \{\mathsf{WS}(x;V,z)\}$ }.

The intuition behind (PO1) and (PO2) is that f guarantees one-way secrecy of the x-part of $x||y$. For example, we verify Pointcheval's transformer in [19] in Appendix **??**.

OW-PCA: Some constructions such as REACT are based on probabilistic one-way functions that are difficult to invert even when the adversary has access to a plaintext checking oracle (PC), which on input a pair (m,c), answers whether c encrypts m. In order to deal with OW-PCA functions, we need to strengthen the meaning of our predicates allowing the adversary to access to the additional plaintext checking oracle. For instance, the definition of $\mathsf{WS}(x;V)$ becomes: $X \models \mathsf{WS}(x;V)$ iff $\Pr[(S,\vec{H},(f,f^{-1})) \xleftarrow{r} X : A^{PCA}(S(V)) = S(x)]$ is negligible, for any adversary A. Now, we have to revisit Lemma 10 and the axioms that introduce $\mathsf{WS}(x;V)$ in the postcondition. It is, however, easy to check that they are valid.

These extensions allow us to prove the security of PKC and REACT in the extended version of this paper.

4 Plaintext awareness

Bellare and Rogaway introduced *plaintext awareness (PA)* in [6][5]. The motivation is to decompose IND-CCA security of an encryption scheme into IND-CPA and PA security. Indeed, a public-key encryption scheme that satisfies IND-CPA (in the ROM) and the original definition of PA is IND-CCA1 (in the ROM). PA has been refined in [4] such that if an encryption scheme is PA and IND-CPA then it is IND-CCA. Intuitively, plaintext awareness means that the decryption oracle can be simulated by a *plaintext extractor* that does not have access to the inverse permutation f^{-1}. Now we introduce a simple analysis that allows us to automatically verify that an encryption scheme is PA in the strong sense [4]. Hence, combined with the results of the previous sections we obtain an analysis that allows to verify IND-CCA security.

We recall the definition of PA-security following the notations and conventions of [4]. Let $GE = (\mathbb{F}, \mathcal{E}(\mathrm{in}_e, \mathrm{out}_e) : \mathsf{c}, \mathcal{D}(\mathrm{in}_d, \mathrm{out}_d) : \mathsf{c}')$ be a generic encryption scheme. An adversary B for plaintext awareness is given the public permutation f, oracle access to the encryption algorithm $\mathcal{E}$ and to the ideal hash functions $\vec{H} = H_1, \cdots, H_n$. His goal is to output a cipher-text that cannot be correctly decrypted by the plaintext extractor. Hence, the success of plaintext extractor K against B in the distribution $X \in \mathrm{DIST}(\Gamma, \vec{H}, \mathbb{F})$ is defined by:

$$\begin{aligned}&\mathsf{Succ}^{\mathsf{pa}}_{K,B,GE}(\eta, X) = \mathsf{Pr}[(S, \vec{H}, (f, f^{-1})) \xleftarrow{r} X; (hH, C, y, S') \xleftarrow{r} B^{\mathcal{E}(),\vec{H}}(f);\\ &\quad S'' \xleftarrow{r} [\![\mathcal{D}(y)]\!](S', \vec{H}, (f, f^{-1})) : y \in C \vee (y \notin C \wedge K(hH, C, y, f) = S''(\mathrm{out}_d))]\end{aligned}$$

Here by $(hH, C, y, S') \xleftarrow{r} B^{\mathcal{E}(),\vec{H}}(f)$ we mean the following. Run B on input f with oracle access to H_i, $i = 1, \cdots, n$ and $\mathcal{E}()$ (which calls f and H_i), recording B's interaction with the hash functions in hH and his interaction with $\mathcal{E}()$ in C. I.e., hH is a list $(hH_1, \cdots, hH_n)$ of lists. Each list $hH_i = ((h_1, v_1), \cdots, (h_{q_i}, v_{q_i}))$ records all B's H_i-oracle queries $h_1, \cdots, h_{q_i}$ and the corresponding answers $v_1, \cdots, v_{q_i}$. The modified state S' is due to calls of the hash functions either by B or the encryption oracle. The list C records the cipher-texts received in reply to $\mathcal{E}$-queries [6]. Finally, y is B's challenge to the plaintext extractor K. Please notice that K wins whenever B outputs a value $y \in C$.

Definition 3 *An encryption scheme $GE = (\mathbb{F}, \mathcal{E}(in_e, out_e) : \mathsf{c}, \mathcal{D}(in_d, out_d) : \mathsf{c}')$ is* PA-secure*, if there is a polynomial-time probabilistic algorithm K such that for every distribution $X \in \mathrm{DIST}(\Gamma, \vec{H}, \mathbb{F})$ and adversary B, $1 - \mathsf{Succ}^{pa}_{K,B,GE}(\eta, X)$ is a negligible function in* η.

The rest of the section is organized as follows. We first introduce a semantic condition on $\mathcal{D}$ that implies the existence of a plaintext extractor. Then, we show how this condition can be checked syntactically on the code of $\mathcal{D}$ (c'). To ease the presentation, we use $\mathcal{E}(x; r_1; \ldots; r_n)$ to note the ciphertext (i.e. the value of out_e) obtained from the plaintext x (i.e. the value of in_e is given by x), using the random seeds $r_1, \ldots r_n$.

In the remainder of this section, we consider an encryption scheme GE that uses the hash functions $\vec{H} = H_1, \cdots, H_n$. We assume that c' has the following form $\mathsf{c}_1; h :=$

[5] While in the original work by Bellare and Rogaway and in subsequent ones, plaintext awareness includes semantic security as a necessary condition, we prefer to separate plaintext extraction and semantic security

[6] This list was not included in the original definition by Bellare and Rogaway. Without it only IND-CCA1 can be proved but not IND-CCA.

$H_1(t)$; if $\mathcal{V}(\vec{x},h) = v$ then $\text{out}_d := m$ else $\text{out}_d :=$ *"error"* fi, where $\vec{x}$ is a vector of variables (possibly empty) and $\mathcal{V}$ is a (deterministic) function (possibly the identity in which case we do not write it) such that for any $\vec{x}$ and v, the probability that $\mathcal{V}(\vec{x},r) = v$ when r is drawn uniformly at random in the right domain is negligible. Furthermore, we require that the hash function H_1 is not called in c_1 and that the encryption algorithm c makes exactly one call to the oracle H_1, (t^*,h^*), and that the value of the variable t after running the decryption oracle is t^*. Consider, for instance, the scheme in [5], $f(r)||\text{in}_e \oplus G(r)||H(\text{in}_e||r)$. Here, t gets assigned the value $\text{in}_e||r$. We call the condition $\mathcal{V}(\vec{x},h) = v$ (or equivalently $\mathcal{V}(\vec{x},H_1(t)) = v$) the "*sanity check*".

It allows us to discriminate valid cipher-text from arbitrary bit-string. We also assume that c_1 does not make calls to H_1 and that decryption behaves correctly with respect to encryption: if y is generated using the encryption algorithm, then the value of t as computed by the decryption oracle coincides with the value used as argument in the call to H_1 by the encryption algorithm.

Example 3 *Bellare and Rogaway [5]:* $\mathcal{D}(in_d = a^*||b^*||v^*, out_d)$:
$r^* := f^{-1}(a^*); g^* := G(r^*); m^* := b^* \oplus g^*; t := m^*||r^*; h := H(t);$
if $h = v^*$ *then* $out_d := m^*$ *else* $out_d :=$ "error" *fi*

A semantic criterion for PA Our semantic criterion for PA-security is composed of three conditions. We begin with an informal presentation of these conditions and how they will enable us to construct a plaintext extractor.

1. The first condition says that there is an algorithm that checks if a given bit-string t^*, that has been submitted to H_1 by B, corresponds to the challenge y. That is, if the tester answers "yes" (1), then t^* matches with the value of t as computed by the decryption oracle; and if it answers "no" (0), then t^* does not satisfy the sanity check.

2. The second condition states that it is easy to compute the plaintext from t^*.

3. The third condition states that for each value of t there is at most one corresponding ciphertext y.

Assume now that these conditions are satisfied. Then, we can construct a plaintext extractor K as follows. Using the algorithm of the first condition, that we call the tester, scan the list hH_1 to find a suitable t^*. If none is found, answer "error". Otherwise, apply the algorithm of the second condition on the found value t^* to extract the plaintext. The third condition ensures that each t^* value corresponds to at most one ciphertext which is necessary to ensure that the extracted plaintext is the correct one. Let us now tackle the formal treatment of these ideas.

Definition 4 *We say that GE satisfies the* PA-semantic criterion*, if there exist efficient algorithms $\mathcal{T}$ and* Ext *that satisfy the following conditions:*
1. *The* tester $\mathcal{T}$ *takes as input* (hH, C, y, t^*, f) *and returns a value in* $\{0,1\}$. *We require that for any adversary B and any distribution* $X \in \text{DIST}(\Gamma, \vec{H}, \mathbb{F})$,

$$\begin{aligned}1 - Pr[\,&(S,\vec{H},(f,f^{-1})) \xleftarrow{r} X; (hH,C,y,S') \xleftarrow{r} B^{\mathcal{E}(),\vec{H}}(f);\\ &S'' \xleftarrow{r} [\![\mathcal{D}(y)]\!](S',\vec{H},(f,f^{-1})); t^* \xleftarrow{r} hH_1.dom; b \xleftarrow{r} \mathcal{T}(hH,C,y,t^*,f) :\\ &\big(b = 1 \Rightarrow (H_1(t^*) = H_1(S''(t)) \wedge \mathcal{V}(S''(\vec{x}),H_1(t^*)) = S''(v)\big) \wedge\\ &\big(b = 0 \Rightarrow \mathcal{V}(S''(\vec{x}),H_1(t^*)) \neq S''(v)\big)]\ \textit{is negligible.}\end{aligned}$$

2. *For* Ext*, we require that for any adversary B and any distribution* $X \in \textsc{Dist}(\Gamma, \vec{H}, \mathbb{F})$*,*

$$1 - \mathit{Pr}[(S, \vec{H}, (f, f^{-1})) \xleftarrow{r} X; (hH, C, y, S') \xleftarrow{r} B^{\mathcal{E}(), \vec{H}}(f); S'' \xleftarrow{r} [\![\mathcal{D}(y)]\!](S', \vec{H}, (f, f^{-1})) \\ : \mathit{Ext}(hH, C, y, S''(t), f) = S''(out_d)] \text{ is negligible.}$$

3. *Finally, we require that for any adversary B and any distribution* $X \in \textsc{Dist}(\Gamma, \vec{H}, \mathbb{F})$*,*

$$\mathit{Pr}[(S, \vec{H}, (f, f^{-1})) \xleftarrow{r} X; (hH, C, y, y', S') \xleftarrow{r} B^{\mathcal{E}(), \vec{H}}(f); \\ S_1 \xleftarrow{r} [\![\mathcal{D}(y)]\!](S', \vec{H}, (f, f^{-1})); S_2 \xleftarrow{r} [\![\mathcal{D}(y')]\!](S', \vec{H}, (f, f^{-1})) : \\ y \neq y' \wedge S_1(t) = S_2(t) \wedge S_1(out_d) \neq \text{"error"} \wedge S_2(out_d) \neq \text{"error"}] \text{ is negligible.}$$

Of course there are generic encryption schemes for which the conditions above are satisfied under the assumption that $\mathcal{T}$ has access to an extra oracle such as a plaintext checking oracle (PC), or a ciphertext validity-checking oracle, which on input c answers whether c is a valid ciphertext or not (CV). In this case, semantic security of the scheme has to be established under the assumption that f is OW-PCA, respectively OW-CVA. Furthermore, our definition of the PA-semantic criterion makes perfectly sense for constructions that apply to IND-CPA schemes such as Fujisaki and Okamoto's converter [16]. In this case, f has to be considered as the IND-CPA encryption oracle.

Given a tester $\mathcal{T}$ and an algorithm Ext as in Definition 4, we construct a plaintext extractor as follows:

$K^{\mathcal{T}, \mathsf{Ext}}(hH, C, y, f)$: Let $L = \{t^* \mid t^* \in \mathsf{dom}(hH_1)$ such that $\mathcal{T}(hH, C, y, t^*, f) = 1\}$
if $L = \emptyset$ then return "error" else $t^* \xleftarrow{r} L$; return $\mathsf{Ext}(hH, C, y, t^*, f)$

Theorem 1 *Let GE be a generic encryption scheme that satisfies the PA-semantic criterion. Then, GE is PA-secure.*

An easy syntactic check that implies the PA-semantic criterion is as follows.

Definition 5 *A generic encryption scheme GE satisfies the* PA-syntactic criterion*, if the sanity check has the form* $\mathcal{V}(t, h) = v$*, where* $\mathcal{D}$ *is such that h is assigned* $H_1(t)$*, t is assigned* $in_e||r$*,* in_e *is the plaintext and* $\mathcal{E}(in_e; r)$ *is the ciphertext (i.e., r is the random seed of* $\mathcal{E}$*).* □

It is not difficult to see that if *GE* satisfies the PA-syntactic criterion then it also satisfies the PA-semantic with a tester $\mathcal{T}$ as follows (Ext is obvious):

> Look in hH_1 for a bit-string s such that $\mathcal{E}(x^*; r^*) = y$, where y is the challenge and $x^*||r^* = s$.

Here are some examples that satisfy the syntactic criterion (we use $\cdot^*$ to denote the values computed by the decryption oracle):

Example 4
- *Bellare and Rogaway [5]:* $\mathcal{E}(in_e; r) = a||b||c = f(r)||in_e \oplus G(r)||H(in_e||r)$*. The "sanity check" of the decryption algorithm is* $H(m^*||r^*) = c^*$*.*
- *OAEP+ [20]:* $\mathcal{E}(in_e; r) = f(a||b||c)$*, where* $a = in_e \oplus G(r)$*,* $b = H'(in_e||r)$*,* $c = H(s) \oplus r$ *and* $s = in_e \oplus G(r)||H'(in_e||r)$*. The "sanity check" of the decryption algorithm has the form* $H'(m^*||r^*) = b^*$*.*

- *Fujisaki and Okamoto [16]:* $\mathcal{E}(in_e;r) = \mathcal{E}'((in_e||r), H(in_e||r))$, *where* $(\mathcal{K}', \mathcal{E}', \mathcal{D}')$ *is a public encryption scheme (that is CPA). The "sanity check" of the decryption algorithm is:* $\mathcal{E}'(m^*||r^*, H(m^*||r^*)) = in_d$.

The PA-semantic criterion applies to the following constructions but not the syntactic one:

Example 5

- *Pointcheval [19]:* $\mathcal{E}(in_e;r;s) = f(r||H(in_e||s))||((in_e||s) \oplus G(r))$, *where* f *is a partially trapdoor one-way injective function. The "sanity check" of the decryption oracle* $\mathcal{D}(a||b)$ *has the form* $f(r^*||H(m^*||s^*)) = a^*$. *The tester looks in hG and hH for* r^* *and* $m^*||s^*$ *such that* $\mathcal{E}(m^*;r^*;s^*) = y$.
- *REACT [18]: This construction applies to any trapdoor one-way function (possibly probabilistic). It is quite similar to the construction in [5]:* $\mathcal{E}(in_e;R;r) = a||b||c = f(R;r)||in_e \oplus G(r)||H(R||in_e||a||b)$, *where* $a = f(R;r)$ *and* $b = in_e \oplus G(R)$. *The "sanity check" of the decryption algorithm is* $H(R^*||m||a^*||b^*) = c$. *For this construction, one can provide a tester* $\mathcal{T}$ *that uses a PCA oracle to check whether a is the encryption of R by* f. *Hence, the PA security of the construction under the assumption of the OW-PCA security of* f. *The tester looks in hH for* $R^*||m||a^*||b^*$ *such that* $c^* = H(R^*||m||a^*||b^*)$ *and* $a^* = f(R^*)$, *which can be checked using the CPA-oracle.*

And now some examples of constructions that do not satisfy the PA-semantic criterion (and hence, not the syntactic one):

Example 6

- *Zheng-Seberry Scheme [25]:* $\mathcal{E}(x;r) = a||b = f(r)||(G(r) \oplus (x||H(x))$. *The third condition of the PA-semantic criterion is not satisfied by this construction. Actually, there is an attack [22] on the IND-CCA security of this scheme that exploits this fact.*
- *OAEP [6]:* $\mathcal{E}(in_e;r) = a = f(s||r \oplus H(s))$, *where* $s = (in_e||0^k) \oplus G(r)$. *Here the third condition is not satisfied.*

5 Conclusion

In this paper we proposed an automatic method to prove IND-CCA security of generic encryption schemes. IND-CPA is proved using a Hoare logic and plaintext awareness using a syntactic criterion. An implementation based on the weakest precondition calculus associated to our logic has been implemented and experimented positively on many examples (cf. *http://www-verimag.imag.fr/ lakhnech/checker.ml*). Finally, it is not difficult to adapt our Hoare logic to allow a security proof in the concrete framework of provable security.

References

[1] G. Barthe, J. Cederquist, and S. Tarento. A Machine-Checked Formalization of the Generic Model and the Random Oracle Model. In *Proceedings of IJCAR'04*, volume 3097 of *LNCS*, pages 385–399, 2004.

[2] Gilles Barthe, Bejamin Grégoire, Romain Janvier, and Santiago Zanella Béguelin. A framework for language-based cryptographic proofs. In *2nd Informal ACM SIGPLAN Workshop on Mechanizing Metatheory*, Oct 2007.

[3] Gilles Barthe and Sabrina Tarento. A machine-checked formalization of the random oracle model. In *Proceedings of TYPES'04*, volume 3839 of *Lecture Notes in Computer Science*, pages 33–49. Springer, 2004.

[4] Mihir Bellare, Anand Desai, David Pointcheval, and Phillip Rogaway. Relations among notions of security for public-key encryption schemes. In *CRYPTO '98: Proceedings of the 18th Annual International Cryptology Conference on Advances in Cryptology*, pages 26–45, London, UK, 1998. Springer-Verlag.

[5] Mihir Bellare and Phillip Rogaway. Random oracles are practical: a paradigm for designing efficient protocols. In *CCS '93: Proceedings of the 1st ACM conference on Computer and communications security*, pages 62–73, New York, USA, November 1993. ACM, ACM.

[6] Mihir Bellare and Phillip Rogaway. Optimal asymmetric encryption. In Alfredo De Santis, editor, *EUROCRYPT*, volume 950 of *Lecture Notes in Computer Science*, pages 92–111. Springer, 1994.

[7] 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/*.

[8] Bruno Blanchet. A computationally sound mechanized prover for security protocols. In *S&P*, pages 140–154. IEEE Computer Society, 2006.

[9] Bruno Blanchet and David Pointcheval. Automated security proofs with sequences of games. In Cynthia Dwork, editor, *CRYPTO*, volume 4117 of *Lecture Notes in Computer Science*, pages 537–554. Springer, 2006.

[10] Ricardo Corin and Jerry den Hartog. A probabilistic hoare-style logic for game-based cryptographic proofs. In *ICALP (2)*, volume 4052 of *Lecture Notes in Computer Science*, pages 252–263. Springer, 2006.

[11] Ivan Damgard. Towards practical public key systems secure against chosen ciphertext attacks. In *CRYPTO '91: Proceedings of the 11th Annual International Cryptology Conference on Advances in Cryptology*, pages 445–456, London, UK, 1992. Springer-Verlag.

[12] Ivan Damgård. A "proof-reading" of some issues in cryptography. In *Automata, Languages and Programming 34th International Colloquium, ICALP 2007.*, volume 4596 of *llncs*, 2007.

[13] Anupam Datta, Ante Derek, John C. Mitchell, and Bogdan Warinschi. Computationally sound compositional logic for key exchange protocols. In *CSFW*, pages 321–334, 2006.

[14] Alexander W. Dent. Fundamental problems in provable security and cryptography. *Phil Trans R Soc A*, 364:3215–3230, 2006.

[15] U. Feige, A. Fiat, and A. Shamir. Zero-knowledge proofs of identity. *J. Cryptol.*, 1(2):77–94, 1988.

[16] Eiichiro Fujisaki and Tatsuaki Okamoto. How to enhance the security of public-key encryption at minimum cost. In *PKC '99: Proceedings of the Second International Workshop on Practice and Theory in Public Key Cryptography*, pages 53–68, London, UK, 1999. Springer-Verlag.

[17] Shai Halevi. A plausible approach to computer-aided cryptographic proofs. *http://theory.lcs.mit.edu/ shaih/pubs.html*, 2005.

[18] Tatsuaki Okamoto and David Pointcheval. React: Rapid enhanced-security asymmetric cryptosystem transform. In *CT-RSA 2001: Proceedings of the 2001 Conference on Topics in Cryptology*, pages 159–175, London, UK, 2001. Springer-Verlag.

[19] David Pointcheval. Chosen-ciphertext security for any one-way cryptosystem. In *PKC '00: Proceedings of the Third International Workshop on Practice and Theory in Public Key Cryptography*, pages 129–146, London, UK, 2000. Springer-Verlag.

[20] Victor Shoup. Oaep reconsidered. *J. Cryptology*, 15(4):223–249, 2002.

[21] Victor Shoup. Sequences of games: a tool for taming complexity in security proofs, 2004. URL: `http://eprint.iacr.org/2004/332`.

[22] David Soldera, Jennifer Seberry, and Chengxin Qu. The analysis of zheng-seberry scheme. In Lynn Margaret Batten and Jennifer Seberry, editors, *ACISP*, volume 2384 of *Lecture Notes in Computer Science*, pages 159–168. Springer, 2002.

[23] Jacques Stern, David Pointcheval, John Malone-Lee, and Nigel P. Smart. Flaws in applying proof methodologies to signature schemes. In Moti Yung, editor, *CRYPTO*, volume 2442 of *Lecture Notes in Computer Science*, pages 93–110. Springer, 2002.

[24] Sabrina Tarento. Machine-checked security proofs of cryptographic signature schemes. In *Computer Security - ESORICS 2005, 10th European Symposium on Research in Computer Security, Milan, Italy, September 12-14, 2005, Proceedings*, volume 3679 of *Lecture Notes in Computer Science*, pages 140–158. Springer, 2005.

[25] Yuliang Zheng and Jennifer Seberry. Immunizing public key cryptosystems against chosen ciphertext attacks. *IEEE Journal on Selected Areas in Communications*, 11(5):715–724, 1993.