Beyond Provable Security: Verifiable IND-CCA Security of OAEP

Gilles Barthe\textsuperscript{1}  Benjamin Grégoire\textsuperscript{2}
Yassine Lakhnech\textsuperscript{3}  Santiago Zanella Beguelin\textsuperscript{1}

\textsuperscript{1} IMDEA Software, Madrid, Spain
\textsuperscript{2} INRIA Sophia Antipolis - Méditerranée, France
\textsuperscript{3} VERIMAG, CNRS Université Joseph Fourier, Grenoble, France
Computer-aided security proofs

Something is wrong with cryptographic proofs:

- In our opinion, many proofs in cryptography have become essentially unverifiable. Our field may be approaching a crisis of rigor. M. Bellare and P. Rogaway, 2006.
- Do we have a problem with cryptographic proofs? Yes, we do [...] We generate more proofs than we carefully verify (and as a consequence some of our published proofs are incorrect). S. Halevi, 2005

Computer-aided proofs

- Provide high guarantees of mathematical correctness
- Have been used successfully for hardware design, program verification, compiler verification, correct-by-construction operating systems, certified program analysers...
- Can be used successfully for provable cryptography
CertiCrypt: machine-checking provable security

A framework for checking code-based game-based concrete security proofs in a general purpose proof assistant

- Security goals, properties and hypotheses are explicit
- All proof steps are conducted in a unified formalism
- The tool provides independently checkable certificates

CertiCrypt has been used for proving:

- indistinguishability of encryption schemes
- unforgeability of signature schemes
- zero-knowledge protocols
- indifferentiability from random oracles
Architecture

The code-based view

- Game = Probabilistic program
- Game transformation = Program transformation
- Game-based proof = Program verification

- Framework for defining games
  - Mathematical libraries: group, fields...
  - Semantics and complexity of probabilistic programs
  - Adversarial model and formalization of security definitions

- Tools to reason about games
  - Semantics-preserving program transformations
  - Observational equivalence and relational logic
  - Game-based lemmas, e.g. failure events
**PWHILE: a probabilistic programming language**

\[
C ::= \text{skip} \quad \text{nop} \\
| C; C \quad \text{sequence} \\
| V ← E \quad \text{assignment} \\
| V ← T \quad \text{random sampling} \\
| \text{if } E \text{ then } C \text{ else } C \quad \text{conditional} \\
| \text{while } E \text{ do } C \quad \text{while loop} \\
| V ← P(E, \ldots, E) \quad \text{procedure call} \\
| V ← A(E, \ldots, E) \quad \text{adversary call}
\]

The semantics of the language is instrumented with cost

\[
\llbracket \cdot \rrbracket : C \rightarrow (S \times \mathbb{N}) \rightarrow (S \times \mathbb{N} \rightarrow [0, 1]) \rightarrow [0, 1]
\]

to capture PPT computations
Program equivalence

All game-based reasoning is justified relative to established notions of program correctness:

- Observational equivalence
- Relational Hoare Logic

Observational equivalence

\[ \models G_1 \simeq_O G_2 \text{ iff for all memories } m_1 \text{ and } m_2:\]

- IF \( m_1 =_I m_2 \), i.e. \( m_1 \) and \( m_2 \) coincide on input variables \( I \),
- THEN \( \llbracket G_1 \rrbracket m_1 \) and \( \llbracket G_2 \rrbracket m_2 \) coincide on output variables \( O \)

Assume \( \models G_1 \simeq_O G_2 \).

- IF \( m_1 =_I m_2 \) and \( A =_O A \) (\( A \) only depends on \( O \)),
- THEN \( \Pr_{G_1,m_1}[A] = \Pr_{G_2,m_2}[A] \)
Reasoning about program equivalence

- Verified library of program transformations

\[
T(G_1, G_2, I, O) = (G'_1, G'_2, I', O') \models G'_1 \sim_{O'} G'_2
\]
\[
\models G_1 \sim_{O} G_2
\]

for
- common compiler optimizations
- interprocedural motion of random assignments

- Automated information flow analysis: find \( I \) such that

\[
\models G \sim_{O} G
\]

- Equality of distributions from algebraic equalities

\[
\models x \leftarrow \{0, 1\}^k; y \leftarrow x \oplus z \sim_{\{x,y,z\}} \{z\};
\]
\[
y \leftarrow \{0, 1\}^k; x \leftarrow y \oplus z
\]
Beyond program equivalence: failure events

- Fundamental Lemma: if two games $G_1$ and $G_2$ are identical up to some failure event $bad$ then,

$$|\Pr_{G_1,m[A]} - \Pr_{G_2,m[A]}| \leq \max(\Pr_{G_1,m[bad]}, \Pr_{G_2,m[bad]})$$

- Failure Event Lemma (some conditions omitted):
  - IF calls to oracle $\mathcal{O}$ trigger $bad$ with probability less than $\epsilon$
  - AND a maximum of $q$ calls to $\mathcal{O}$ are allowed
  - THEN $\Pr_{G,m[bad]} \leq q \epsilon$
Application: RSA-OAEP

1994  Purported proof of chosen-ciphertext security
2001  Proof establishes a weaker security notion, but desired security can be achieved
   1  ...for a modified scheme, or
   2  ...under stronger assumptions
2004  Filled gaps in Fujisaki et al. 2001 proof
2009  Security definition needs to be clarified
2010  Filled gaps and marginally improved bound in 2004 proof
Exact IND-CCA security of OAEP

**Game IND-CCA2:**
\[(pk, sk) \leftarrow KG(\eta);\]
\[(m_0, m_1) \leftarrow A_1(pk);\]
\[b \leftarrow \{0, 1\};\]
\[c^* \leftarrow \text{Enc}(m_b);\]
\[\overline{b} \leftarrow A_2(c^*)\]

**Game PD-OW:**
\[(pk, sk) \leftarrow KG_f(\eta);\]
\[s \leftarrow \{0, 1\}^{n+k_1};\]
\[t \leftarrow \{0, 1\}^{k_0};\]
\[\overline{s} \leftarrow \mathcal{I}(f(pk, s \parallel t))\]

Security statement
\[
\forall A, \exists I,

2 \left| \Pr_{\text{IND-CCA2}}[\overline{b} = b] - \frac{1}{2} \right| \leq

q_H \Pr_{\text{PD-OW}}[\overline{s} = s] + \frac{3q_{Dec}q_G + q_{Dec}^2}{2^{k_0}} + 4q_{Dec} + q_G + \frac{2q_{Dec}}{2^{k_1}}
\]
Exact IND-CCA security of OAEP: formal statement

**Game** IND-CCA2:

- $L_G, L_H, L_{Dec} \leftarrow d$;
- $(pk, sk) \leftarrow KG(\eta)$;
- $(m_0, m_1) \leftarrow A_1(pk)$;
- $b \leftarrow \{0, 1\}$;
- $c^* \leftarrow Enc(m_b)$;
- $c^*_\text{def} \leftarrow \text{true}$;
- $\overline{b} \leftarrow A_2(c^*)$

**Oracle** $G(r)$:
- if $r \notin \text{dom}(L_G)$ then
- $L_G[r] \leftarrow \{0, 1\}^{n+k_1}$;
- return $L_G[r]$

**Oracle** $H(r)$: ...

**Oracle** $\text{Dec}(c)$:
- $L_{Dec} \leftarrow (c^*_\text{def}, c) :: L_{Dec}$;
- ...

Security statement

$\forall A, \exists I, WF(A) \land$

$$\Pr \left[ \text{IND-CCA2} : |L_G| \leq q_G + q_{Dec} \land |L_H| \leq q_H \land |L_{Dec}| \leq q_{Dec} \land (\text{true}, c^*) \notin L_{Dec} \right] = 1$$

$$\implies 2 \left| \Pr_{\text{IND-CCA2}}[\overline{b} = b] - \frac{1}{2} \right| \leq$$

$$q_H \Pr_{\text{PD-OW}}[\overline{s} = s] + \frac{3q_{Dec}q_G + q_{Dec}^2 + 4q_{Dec} + q_G}{2^{k_0}} + \frac{2q_{Dec}}{2^{k_1}}$$
Proof highlights

Calls to hash oracles are eliminated by successive modifications of the decryption oracle, as in Pointcheval 2004. Main differences:

- Both calls to $G$ are eliminated simultaneously
- Elimination of calls to $H$ requires no more calls to $G$

Justifying eliminations of calls to $G$

- Tag queries to $G$ with origin (adversary vs. decryption oracle), and set a bad flag in Dec when a valid ciphertext is produced with $G(r)$ not queried.
- Shift flag to $G$ oracle. Apply logic of swapping statements to show that values that are uniformly distributed and independent from adversary’s view can be resampled
- Apply logic of failure events
You only need to trust:
- the checker
  - foundational formalism, studied by logicians for \( \geq 30 \) years \( \implies \) rock solid
  - part of CertiCrypt infrastructure
    - probabilities, programming language semantics \( \implies \) well understood
- the statement
  - for OAEP, about 100 lines \( \implies \) manageable

You do not need to trust the proof nor even the proof tools (relational Hoare logic, program transformations, etc), the sequence of games, etc.
Conclusion and perspectives

- Independently verifiable proof of IND-CCA2 security of OAEP
- Computer-aided cryptographic proofs are becoming a reality
- Next step: build highly automated tools accessible to the working cryptographers, using state-of-the-art automated tools