Beyond Provable Security: Verifiable IND-CCA Security of OAEP

Gilles Barthe[1] Benjamin Grégoire[2]
Yassine Lakhnech[3] Santiago Zanella Beguelin[1]

[1] IMDEA Software, Madrid, Spain

[2] INRIA Sophia Antipolis - Méditerranée, France

[2] 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 verfication, 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

$\mathcal{C}$	::=	skip	nop
	|	$\mathcal{C}$; $\mathcal{C}$	sequence
	|	$\mathcal{V} \leftarrow \mathcal{E}$	assignment
	|	$\mathcal{V} \xleftarrow{\$} T$	random sampling
	|	if $\mathcal{E}$ then $\mathcal{C}$ else $\mathcal{C}$	conditional
	|	while $\mathcal{E}$ do $\mathcal{C}$	while loop
	|	$\mathcal{V} \leftarrow \mathcal{P}(\mathcal{E}, \ldots, \mathcal{E})$	procedure call
	|	$\mathcal{V} \leftarrow \mathcal{A}(\mathcal{E}, \ldots, \mathcal{E})$	adversary call

The semantics of the language is instrumented with cost

$$[\![\cdot]\!] : \mathcal{C} \to (\mathcal{S} \times \mathbb{N}) \to (\mathcal{S} \times \mathbb{N} \to [0, 1]) \to [0, 1]$$

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

$\vDash \mathsf{G}_1 \simeq_O^I \mathsf{G}_2$ iff for all memories m_1 and m_2:

- IF $m_1 =_I m_2$, i.e. m_1 and m_2 coincide on input variables I,
- THEN $[\![\mathsf{G}_1]\!]\, m_1$ and $[\![\mathsf{G}_2]\!]\, m_2$ coincide on output variables O

Assume $\vDash \mathsf{G}_1 \simeq_O^I \mathsf{G}_2$.

- IF $m_1 =_I m_2$ and $A =_O A$ (A only depends on O),
- THEN $\Pr_{\mathsf{G}_1, m_1}[A] = \Pr_{\mathsf{G}_2, m_2}[A]$

Reasoning about program equivalence

- Verified library of program transformations

$$\frac{T(\mathsf{G}_1, \mathsf{G}_2, I, O) = (\mathsf{G}_1', \mathsf{G}_2', I', O') \qquad \vDash \mathsf{G}_1' \simeq_{O'}^{I'} \mathsf{G}_2'}{\vDash \mathsf{G}_1 \simeq_{O}^{I} \mathsf{G}_2}$$

  for

  - common compiler optimizations
  - interprocedural motion of random assignments

- Automated information flow analysis: find I such that

$$\vDash \mathsf{G} \simeq_{O}^{I} \mathsf{G}$$

- Equality of distributions from algebraic equalities

$$\vDash x \xleftarrow{\$} \{0,1\}^k; y \leftarrow x \oplus z \simeq_{\{x,y,z\}}^{\{z\}} y \xleftarrow{\$} \{0,1\}^k; x \leftarrow y \oplus z$$

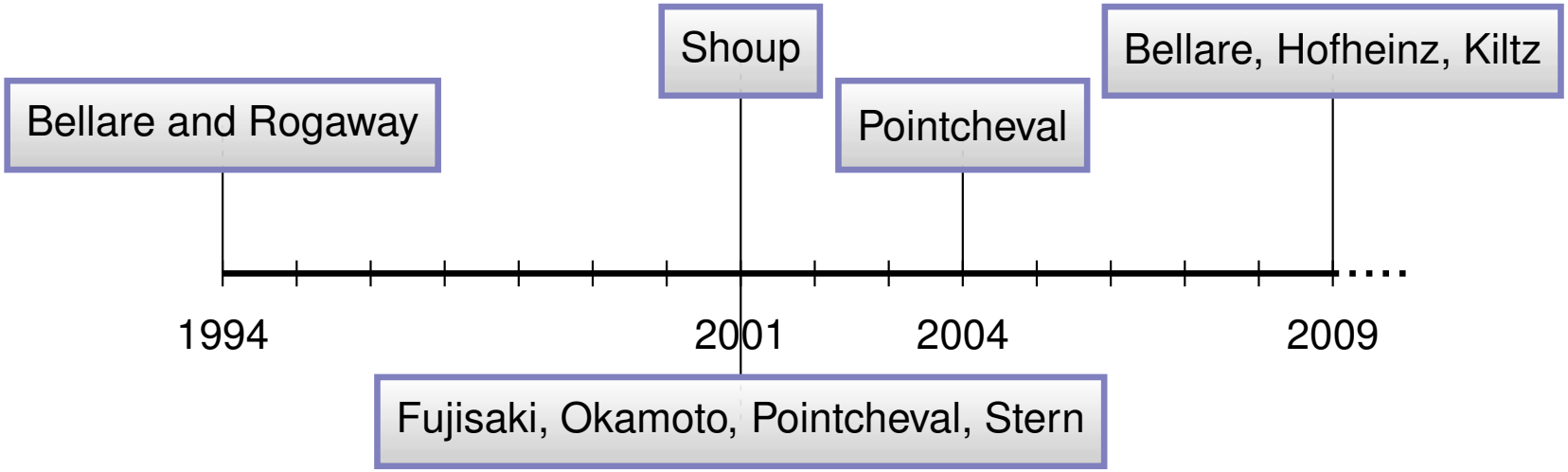
Beyond program equivalence: failure events

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

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

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

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 :

$(pk, sk) \leftarrow \mathcal{KG}(\eta)$;
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk)$;
$b \xleftarrow{\$} \{0,1\}$;
$c^* \leftarrow \mathrm{Enc}(m_b)$;
$\overline{b} \leftarrow \mathcal{A}_2(c^*)$

Game PD-OW :

$(pk, sk) \leftarrow \mathcal{KG}_f(\eta)$;
$s \xleftarrow{\$} \{0,1\}^{n+k_1}$;
$t \xleftarrow{\$} \{0,1\}^{k_0}$;
$\overline{s} \leftarrow \mathcal{I}(f(pk, s \| t))$

Security statement

$\forall \mathcal{A}, \exists \mathcal{I},$

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

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

Exact IND-CCA security of OAEP: formal statement

Game IND-CCA2 :
$L_G, L_H, L_{\mathrm{Dec}} \leftarrow d$;
$(pk, sk) \leftarrow \mathcal{KG}(\eta)$;
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk)$;
$b \xleftarrow{\$} \{0,1\}$;
$c^* \leftarrow \mathrm{Enc}(m_b)$;
$c^*{}_{def} \leftarrow \mathrm{true}$;
$\overline{b} \leftarrow \mathcal{A}_2(c^*)$

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

Oracle $H(r)$: ...

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

Security statement

$$\forall \mathcal{A}, \exists \mathcal{I},\ WF(\mathcal{A}) \wedge$$

$$\Pr\left[\text{IND-CCA2} : \begin{array}{l} |L_G| \leq q_G + q_{\mathrm{Dec}} \wedge |L_H| \leq q_{\mathrm{H}} \wedge |L_{\mathrm{Dec}}| \leq q_{\mathrm{Dec}} \\ \wedge\ (\mathrm{true}, c^*) \notin L_{\mathrm{Dec}} \end{array}\right] = 1$$

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

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

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

Trusting verifiable security

- You only need to trust:
  - the checker
    - foundational formalism, studied by logicians for $\geq$30 years
      $\Longrightarrow$ rock solid
  - part of CertiCrypt infrastructure
    - probabilities, programming language semantics
      $\Longrightarrow$ well understood
  - the statement
    - for OAEP, about 100 lines
      $\Longrightarrow$ 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