Computer-aided cryptographic proofs and designs

Gilles Barthe

IMDEA Software Institute, Madrid, Spain

Benjamin Grégoire (INRIA)
Juan Manuel Crespo (IMDEA)
Francois Dupressoir (IMDEA)
César Kunz (IMDEA)
Yassine Lakhnech (U. Grenoble)
Benedikt Schmidt (IMDEA)
Pierre-Yves Strub (IMDEA)
Santiago Zanella (MSR)

- Cryptography is a small but important part of security
- Proofs are a small but important part of cryptography
- Hard to get right
- Often iterate over extended period ($\geq$10 years)

CertiCrypt project (2006-)

- Cryptographic proofs as deductive program verification
- Tools for high-assurance cryptographic proofs
- Applying PL and verification techniques to cryptography (program synthesis, certifying compilers,. . .)

- Cryptography is a small but important part of security
- Proofs are a small but important part of cryptography
- Hard to get right
- Often iterate over extended period ($\geq$10 years)

CertiCrypt project (2006-)

- Cryptographic proofs as deductive program verification
- Tools for high-assurance cryptographic proofs
- Applying PL and verification techniques to cryptography (program synthesis, certifying compilers,. . .)

The two views of cryptography

Computational cryptography (Goldwasser and Micali, 1982)

- Strong ties with complexity theory
- Feasible adversary breaks scheme with small probability
- Design of secure and efficient primitives and protocols
- Complex and manual proofs

The two views of cryptography

Computational cryptography (Goldwasser and Micali, 1982)

- Strong ties with complexity theory
- Feasible adversary breaks scheme with small probability
- Design of secure and efficient primitives and protocols
- Complex and manual proofs

Symbolic cryptography (Dolev and Yao, 1983)

- Assume perfect cryptography
- Adversary cannot win
- Discovery of logical bugs in protocols
- Strong ties with verification
- Automated proofs

Reconciling the two views

Computational soundness (Abadi and Rogaway, 2000)

- Security in symbolic model implies computational security
- . . . under non-standard assumptions on primitives
- Symbolic tools deliver asymptotic guarantees
- . . . but no end-to-end security

Reconciling the two views

Computational soundness (Abadi and Rogaway, 2000)

- Security in symbolic model implies computational security
- . . . under non-standard assumptions on primitives
- Symbolic tools deliver asymptotic guarantees
- . . . but no end-to-end security

Computational proof systems

- Prove concrete security, under standard assumptions, directly in the computational model
- Logics of probability and indistinguishability. Influenced by cryptographic practice, esp. game-playing proofs (Shoup 2004, Bellare & Rogaway, 2004, Halevi, 2005)
- Symbolic techniques may be used for guiding proof search

Challenges and agenda

1. Anatomy of cryptographic proofs
2. (Semantics-based) formalization
3. Proof checkers
4. Automated proof search
5. Synthesis

- The talk focuses on public-key encryption
- Applies to digital signatures, modes of operation, hash functions, zero-knowledge proofs, authenticated key exchange protocols, multi-party computation...

Anatomy of a cryptographic proof

Example: Bellare & Rogaway encryption

Game IND-CPA :
$(sk, pk) \leftarrow \mathcal{KG}(\,)$;
$(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(pk)$;
$b \xleftarrow{\$} \{0,1\}$;
$c^* \leftarrow \mathrm{Enc}(pk, m_b)$;
$b' \leftarrow \mathcal{A}_2(pk, c^*, \sigma)$;
return $b = b'$

$\mathrm{Enc}_{pk}(m)$:
$r \xleftarrow{\$} \{0,1\}^\ell$;
$s \leftarrow G(r) \oplus m$;
$y \leftarrow f_{pk}(r) \,\|\, s$;
return y

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

Game OW :
$(sk, pk) \leftarrow \mathcal{KG}()$;
$y \xleftarrow{\$} \{0,1\}^\ell$;
$y' \leftarrow \mathcal{I}(f_{pk}(y))$;
return $y = y'$

For every IND-CPA adversary $\mathcal{A}$ making at most q_G queries to G, there exists an inverter $\mathcal{I}$ against OW such that

$$\left| \Pr_{\text{IND-CPA}}\left[b = b'\right] - \frac{1}{2} \right| \leq q_G \, \mathbf{Succ}_f^{\mathrm{OW}}(\mathcal{I})$$

Failure event

Conditionally semantics-preserving transformation

Game G_0 :

$L_G \leftarrow \emptyset;\ L_G^{\mathcal{A}} \leftarrow [\,];$
$(sk, pk) \leftarrow \mathcal{KG}();$
$(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$r \xleftarrow{\$} \{0,1\}^{\ell};$
$g \leftarrow G(r);$
$s \leftarrow g \oplus m_b;$
$c^* \leftarrow f_{pk}(r) \,\|\, s;$
$b' \leftarrow \mathcal{A}_2(pk, c^*, \sigma);$

Game G_1 :

$L_G \leftarrow \emptyset;\ L_G^{\mathcal{A}} \leftarrow [\,];$
$(sk, pk) \leftarrow \mathcal{KG}();$
$(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$r \xleftarrow{\$} \{0,1\}^{\ell};$
$g \xleftarrow{\$} \{0,1\}^{k};$
$s \leftarrow g \oplus m_b;$
$c^* \leftarrow f_{pk}(r) \,\|\, s;$
$b' \leftarrow \mathcal{A}_2(pk, c^*, \sigma);$

Failure event

Conditionally semantics-preserving transformation

Game G_0 :

$L_G \leftarrow \emptyset;\ L_G^{\mathcal{A}} \leftarrow [\,];$
$(sk, pk) \leftarrow \mathcal{KG}();$
$(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$r \xleftarrow{\$} \{0,1\}^{\ell};$
$g \leftarrow G(r);$
$s \leftarrow g \oplus m_b;$
$c^* \leftarrow f_{pk}(r) \,\|\, s;$
$b' \leftarrow \mathcal{A}_2(pk, c^*, \sigma);$

Game G_1 :

$L_G \leftarrow \emptyset;\ L_G^{\mathcal{A}} \leftarrow [\,];$
$(sk, pk) \leftarrow \mathcal{KG}();$
$(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$r \xleftarrow{\$} \{0,1\}^{\ell};$
$g \xleftarrow{\$} \{0,1\}^{k};$
$s \leftarrow g \oplus m_b;$
$c^* \leftarrow f_{pk}(r) \,\|\, s;$
$b' \leftarrow \mathcal{A}_2(pk, c^*, \sigma);$

The games are equivalent until the adversary queries G with r

$$\left|\Pr_{\mathsf{G}_0}\left[b = b'\right] - \Pr_{\mathsf{G}_1}\left[b = b'\right]\right| \leq \Pr_{\mathsf{G}_1}\left[r \in L_G^{\mathcal{A}}\right]$$

Failure event

Conditionally semantics-preserving transformation

Game G_0 :

$L_G \leftarrow \emptyset;\ L_G^{\mathcal{A}} \leftarrow [\,];$
$(sk, pk) \leftarrow \mathcal{KG}();$
$(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$r \xleftarrow{\$} \{0,1\}^{\ell};$
$g \leftarrow G(r);$
$s \leftarrow g \oplus m_b;$
$c^* \leftarrow f_{pk}(r) \,\|\, s;$
$b' \leftarrow \mathcal{A}_2(pk, c^*, \sigma);$

Game G_1 :

$L_G \leftarrow \emptyset;\ L_G^{\mathcal{A}} \leftarrow [\,];$
$(sk, pk) \leftarrow \mathcal{KG}();$
$(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$r \xleftarrow{\$} \{0,1\}^{\ell};$
$g \xleftarrow{\$} \{0,1\}^{k};$
$s \leftarrow g \oplus m_b;$
$c^* \leftarrow f_{pk}(r) \,\|\, s;$
$b' \leftarrow \mathcal{A}_2(pk, c^*, \sigma);$

The games are equivalent until the adversary queries G with r

$$\left|\Pr_{\text{IND-CPA}}\left[b = b'\right] - \Pr_{\mathsf{G}_1}\left[b = b'\right]\right| \leq \Pr_{\mathsf{G}_1}\left[r \in L_G^{\mathcal{A}}\right]$$

Bridging step

Semantics-preserving transformation

Game G_1 :

$L_G \leftarrow \emptyset;\ L_G^{\mathcal{A}} \leftarrow [\,];$
$(sk, pk) \leftarrow \mathcal{KG}();$
$(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$r \xleftarrow{\$} \{0,1\}^{\ell};$
$g \xleftarrow{\$} \{0,1\}^{k};$
$s \leftarrow g \oplus m_b;$
$c^* \leftarrow f_{pk}(r) \,\|\, s;$
$b' \leftarrow \mathcal{A}_2(pk, c^*, \sigma);$

Game G_2 :

$L_G \leftarrow \emptyset;\ L_G^{\mathcal{A}} \leftarrow [\,];$
$(sk, pk) \leftarrow \mathcal{KG}();$
$(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$r \xleftarrow{\$} \{0,1\}^{\ell};$
$s \xleftarrow{\$} \{0,1\}^{k};$
$g \leftarrow s \oplus m_b;$
$c^* \leftarrow f_{pk}(r) \,\|\, s;$
$b' \leftarrow \mathcal{A}_2(pk, c^*, \sigma);$

Bridging step

Semantics-preserving transformation

Game G_1 :

$L_G \leftarrow \emptyset;\ L_G^{\mathcal{A}} \leftarrow [\,];$
$(sk, pk) \leftarrow \mathcal{KG}();$
$(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$r \xleftarrow{\$} \{0,1\}^{\ell};$
$g \xleftarrow{\$} \{0,1\}^{k};$
$s \leftarrow g \oplus m_b;$
$c^* \leftarrow f_{pk}(r) \,\|\, s;$
$b' \leftarrow \mathcal{A}_2(pk, c^*, \sigma);$

Game G_2 :

$L_G \leftarrow \emptyset;\ L_G^{\mathcal{A}} \leftarrow [\,];$
$(sk, pk) \leftarrow \mathcal{KG}();$
$(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$r \xleftarrow{\$} \{0,1\}^{\ell};$
$s \xleftarrow{\$} \{0,1\}^{k};$
$g \leftarrow s \oplus m_b;$
$c^* \leftarrow f_{pk}(r) \,\|\, s;$
$b' \leftarrow \mathcal{A}_2(pk, c^*, \sigma);$

Games are equivalent and c^* is independent from b, hence

$$\left|\Pr_{\text{IND-CPA}}\left[b = b'\right] - \Pr_{\mathrm{G}_2}\left[b = b'\right]\right| \leq \Pr_{\mathrm{G}_2}\left[r \in L_G^{\mathcal{A}}\right]$$

Bridging step

Semantics-preserving transformation

Game G_1 :

$L_G \leftarrow \emptyset;\ L_G^{\mathcal{A}} \leftarrow [\,];$
$(sk, pk) \leftarrow \mathcal{KG}();$
$(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$r \xleftarrow{\$} \{0,1\}^{\ell};$
$g \xleftarrow{\$} \{0,1\}^{k};$
$s \leftarrow g \oplus m_b;$
$c^* \leftarrow f_{pk}(r) \,\|\, s;$
$b' \leftarrow \mathcal{A}_2(pk, c^*, \sigma);$

Game G_2 :

$L_G \leftarrow \emptyset;\ L_G^{\mathcal{A}} \leftarrow [\,];$
$(sk, pk) \leftarrow \mathcal{KG}();$
$(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$r \xleftarrow{\$} \{0,1\}^{\ell};$
$s \xleftarrow{\$} \{0,1\}^{k};$
$g \leftarrow s \oplus m_b;$
$c^* \leftarrow f_{pk}(r) \,\|\, s;$
$b' \leftarrow \mathcal{A}_2(pk, c^*, \sigma);$

Games are equivalent and c^* is independent from b, hence

$$\left| \Pr_{\text{IND-CPA}}\left[b = b'\right] - \frac{1}{2} \right| \leq \Pr_{\mathsf{G}_2}\left[r \in L_G^{\mathcal{A}}\right]$$

Reduction

Existence of simulation

Game G_2 :

$L_G \leftarrow \emptyset;\ L_G^{\mathcal{A}} \leftarrow [\,];$
$(sk, pk) \leftarrow \mathcal{KG}();$
$(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(pk);$
$r \xleftarrow{\$} \{0,1\}^{\ell};$
$s \xleftarrow{\$} \{0,1\}^{k};$
$c^* \leftarrow f_{pk}(r) \,\|\, s;$
$b' \leftarrow \mathcal{A}_2(pk, c^*, \sigma);$

Game OW :

$(sk, pk) \leftarrow \mathcal{KG}();$
$y \xleftarrow{\$} \{0,1\}^{\ell};$
$y' \leftarrow \mathcal{I}(f_{pk}(y));$
return $y = y'$

Adversary $\mathcal{I}(x)$:

$L_G \leftarrow \emptyset;\ L_G^{\mathcal{A}} \leftarrow [\,];$
$(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(pk);$
$s \xleftarrow{\$} \{0,1\}^{k};\ y \leftarrow x \,\|\, s;$
$b' \leftarrow \mathcal{A}_2(pk, y, \sigma);$
$i \xleftarrow{\$} [1, |L_G^{\mathcal{A}}|];$
return $L_G^{\mathcal{A}}[i];$

Reduction

Existence of simulation

Game G_2 :
$L_G \leftarrow \emptyset;\ L_G^{\mathcal{A}} \leftarrow [\,];$
$(sk, pk) \leftarrow \mathcal{KG}();$
$(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(pk);$
$r \xleftarrow{\$} \{0,1\}^{\ell};$
$s \xleftarrow{\$} \{0,1\}^{k};$
$c^* \leftarrow f_{pk}(r) \,\|\, s;$
$b' \leftarrow \mathcal{A}_2(pk, c^*, \sigma);$

Game OW :
$(sk, pk) \leftarrow \mathcal{KG}();$
$y \xleftarrow{\$} \{0,1\}^{\ell};$
$y' \leftarrow \mathcal{I}(f_{pk}(y));$
return $y = y'$

Adversary $\mathcal{I}(x)$:
$L_G \leftarrow \emptyset;\ L_G^{\mathcal{A}} \leftarrow [\,];$
$(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(pk);$
$s \xleftarrow{\$} \{0,1\}^{k};\ y \leftarrow x \,\|\, s;$
$b' \leftarrow \mathcal{A}_2(pk, y, \sigma);$
$i \xleftarrow{\$} [1, |L_G^{\mathcal{A}}|];$
return $L_G^{\mathcal{A}}[i];$

The inverter wins if $r \in L_G^{\mathcal{A}}$ and it guesses i correctly.

$$\left| \Pr_{\text{IND-CPA}}\left[b = b'\right] - \frac{1}{2} \right| \leq q_G \, \mathbf{Succ}_f^{\text{OW}}(\mathcal{I})$$

Formalization

Everything is a probabilistic program!

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

- $\mathcal{M}$ = memories. Map variables to values
- $\mathcal{D}(\mathcal{M})$ = discrete sub-distributions on memories

$$[\![c]\!] : \mathcal{M} \to \mathcal{D}(\mathcal{M})$$

Verification

Relational Hoare Logic: $c_1 \sim c_2 : P \Rightarrow Q$ iff

$$\forall m_1, m_2.\ (m_1, m_2) \vDash P \rightarrow (\llbracket c_1 \rrbracket\ m_1, \llbracket c_2 \rrbracket\ m_2) \vDash Q^\sharp$$

Deriving probability claims

- Fail event: if $\vDash c_1 \sim c_2 : P \Rightarrow \neg F\langle 2 \rangle \rightarrow Q_1\langle 1 \rangle \leftrightarrow Q_2\langle 2 \rangle$ then

$$(m_1, m_2) \vDash P \Longrightarrow |\Pr_{c_1,m_1}[Q_1] - \Pr_{c_2,m_2}[Q_2]| \leq \Pr_{c_2,m_2}[F]$$

- Bridging steps: if $\vDash c_1 \sim c_2 : P \Rightarrow =$ then for all events Q

$$(m_1, m_2) \vDash P \Longrightarrow \Pr_{c_1,m_1}[Q] = \Pr_{c_2,m_2}[Q]$$

- Reductions: if $\vDash c_1 \sim c_2 : P \Rightarrow Q_1\langle 1 \rangle \rightarrow Q_2\langle 2 \rangle$, then

$$(m_1, m_2) \vDash P \Longrightarrow \Pr_{c_1,m_1}[Q_1] \leq \Pr_{c_2,m_2}[Q_2]$$

Instrumentation

Deduction rules for valid pRHL judgments

Assignment

$$\frac{}{\vDash x \leftarrow e \sim x' \leftarrow e' : Q\{x\langle 1\rangle := e\langle 1\rangle, x'\langle 2\rangle := e'\langle 2\rangle\} \Rightarrow Q}$$

$$\frac{}{\vDash x \leftarrow e \sim \mathsf{skip} : Q\{x\langle 1\rangle := e\langle 1\rangle\} \Rightarrow Q}$$

Random assignment

$$\frac{f : A \to B \text{ is 1-1}}{\vDash x \xleftarrow{\$} A \sim x \xleftarrow{\$} B : (\forall v, Q\{x\langle 1\rangle := v, x\langle 2\rangle := f\ v\}) \Rightarrow Q}$$

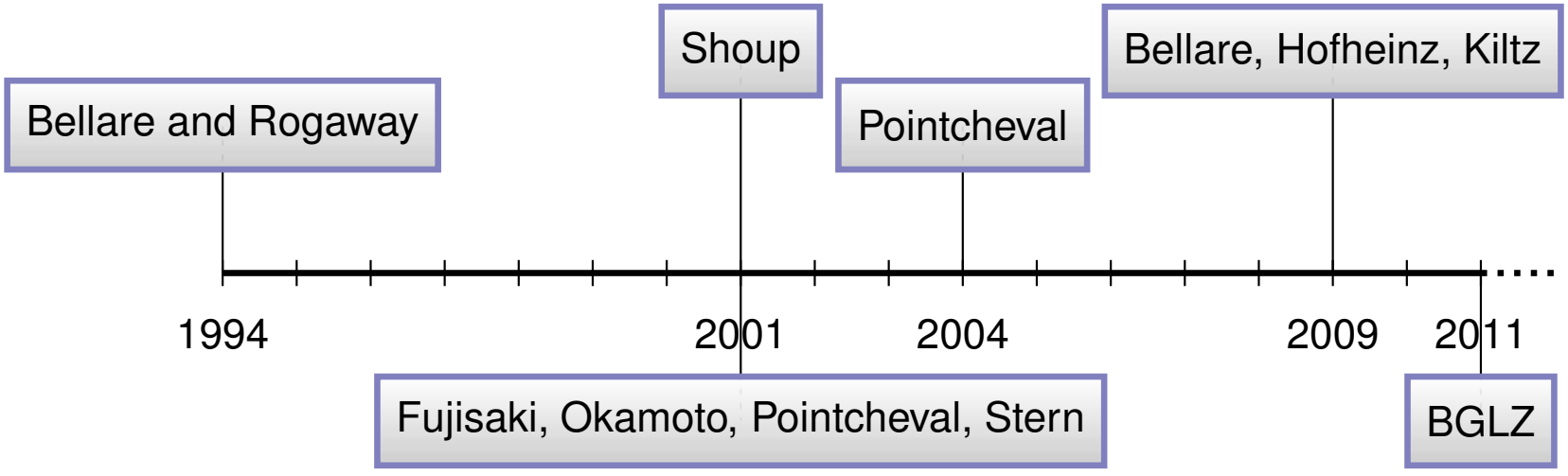
- VC generation or interactive proofs; plus SMT and ATP
- General product program constructions for loops
- Inference of relational invariants (loops, adversaries)

A worked out example: RSA-OAEP

Oracle $\mathrm{Enc}_{pk}(m)$:
$r \xleftarrow{\$} \{0,1\}^{k_0}$;
$s \leftarrow G(r) \oplus (m \| 0^{k_1})$;
$t \leftarrow H(s) \oplus r$;
return $f_{pk}(s \| t)$

Oracle $\mathrm{Dec}_{sk}(c)$:
$(s,t) \leftarrow f_{sk}^{-1}(c)$; $r \leftarrow t \oplus H(s)$;
if $[s \oplus G(r)]_{k_1} = 0^{k_1}$ then return $[s \oplus G(r)]^n$
else return $\perp$

A worked out example: RSA-OAEP



1994 Purported proof of chosen-ciphertext security

2001 1994 proof gives weaker security; desired security holds

- for a modified scheme
- under stronger assumptions

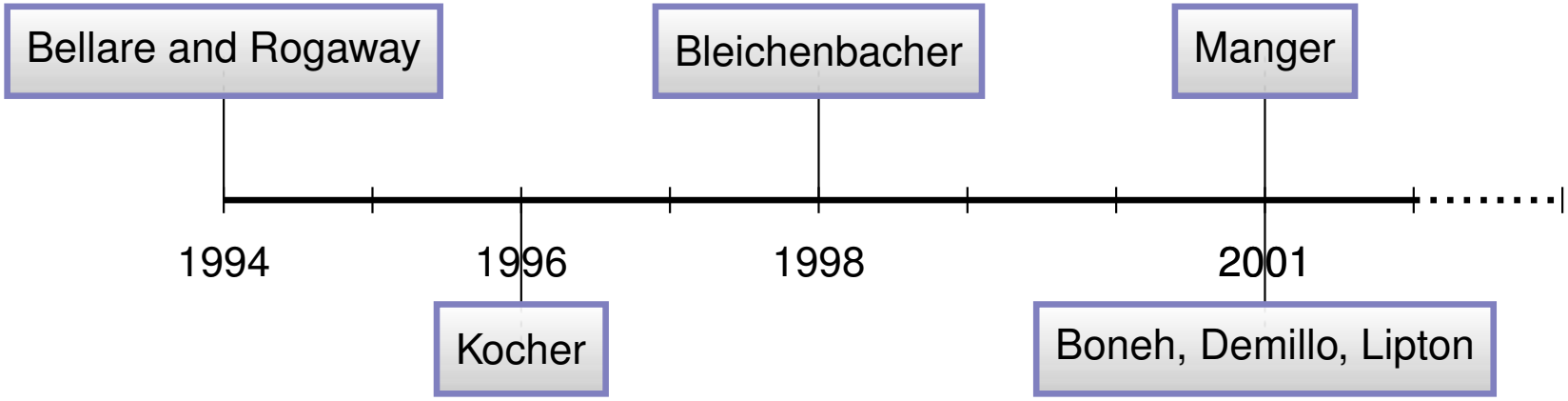
2004 Filled gaps in Fujisaki et al. 2001 proof

2009 Security definition needs to be clarified

2011 Machine-checked proof of algorithm

A worked out example: RSA-OAEP

- Conversion from integers to bitstrings; error messages; . . .
- Attacks and countermeasures against implementations



1996 Timing attack (on RSA)

1998 Padding oracle attack

2001 Fault injection attack

2010 Timing attack (on conversion)

2012 Machine-checked proof of pseudo-implementation

Automation

Goal

- Discover and verify automatically the sequence of games

Solution

- Domain-specific specialization of pRHL
- Symbolic expressions and deducibility
- Hybrid proof rules with computational interpretation
- Naive proof search

$$
\begin{array}{rcll}
\mathcal{E} & ::= & m & \text{input message} \\
 & | & \mathcal{R} & \text{uniform random bitstring} \\
 & | & \mathcal{E} \oplus \mathcal{E} & \text{xor} \\
 & | & \mathcal{E} \parallel \mathcal{E} & \text{concatenation} \\
 & | & H(\mathcal{E}) & \text{hash} \\
 & | & f(\mathcal{E}) & \text{trapdoor permutation} \\
 & | & [\mathcal{E}]^{\ell}_{\ell'} & \text{trapdoor permutation}
\end{array}
$$

Logic for chosen-plaintext security

Deducibility

$$\frac{e \vdash e_1 \quad e \vdash e_2}{e \vdash e_1 \parallel e_2} \qquad \frac{e \vdash e_1 \qquad e \vdash e_2}{e \vdash e_1 \oplus e_2} \qquad \frac{e \vdash e}{e \vdash [e]_n^{\ell}}$$

$$\frac{e \vdash e_1 \qquad \vdash e_1 \doteq e_2}{e \vdash e_2} \qquad \frac{e \vdash e'}{e \vdash H(e')} \qquad \frac{e \vdash e'}{e \vdash f(e')}$$

PadCPA logic

$$\frac{m \notin c^*}{c^* :_{\frac{1}{2}} \text{Guess}} \qquad \frac{e \vdash \vec{r} \qquad f(\vec{r}) \parallel \ldots \vdash c^*}{c^* :_{q_{\mathrm{H}}\ \mathbf{Succ}_f^{\mathrm{OW}}} e \in L_H^{\mathcal{A}}} \qquad \frac{c^* :_p \varphi}{(c^* :_p \varphi)\{e \oplus r/r\}}$$

- Pen-and-paper soundness proof
- Generation to EasyCrypt
- Extends to chosen-ciphertext security (no generation yet)

The next 700 cryptosystems

(adapted from Landin, 1966)

Do the cryptosystems reflect [...] the situations that are being catered for? Or are they accidents of history and personal background that may be obscuring fruitful developments?
[...]
We must think in terms, not of cryptosystems, but of families of cryptosystems. That is to say we must systematize their design so that a new system is a point chosen from a well-mapped space, rather than a laboriously devised construction.

Synthesis

Key steps

- Efficient generation of candidate schemes
- Early detection of insecure and incorrect schemes

  Ex: is randomness extractable without a key? $r \parallel f(m \oplus r)$
- Practical interpretation of secure schemes (for RSA)

Experiments

- Filters are effective ($>$99% of candidates discarded)
- Proof search is effective ($\geq$95% of known schemes)
- Reasonable coverage ($\geq$75%-$\geq$50% definite answers)

OP	GEN	$\neg$ CPA	CPA	$\neg$ CCA	CCA
6	419	244	104	68	1
7	4131	2392	883	537	39
8	41860	24166	7850	4424	436
9	275318	155669	54884	27697	3750

The quest for minimality

Q: How much redundancy is needed to get CCA security?

- OAEP (1994): $f((m\|0) \oplus G(r) \;\|\; r \oplus H((m\|0) \oplus G(r)))$

  not that Optimal; redundancy needed for INDCCA
- SAEP (2001): $f(r \;\|\; (m\|0) \oplus G(r))$

  tighter reduction, also needs redundancy and k big enough
- ZAEP: $f(r \,\|\, G(r) \oplus m)$

  tighter reduction, redundant-free

  $\mathrm{Dec}(c) : r \| t \leftarrow f_{sk}^{-1}(c); g \leftarrow G(r); \text{return } t \oplus g$

The quest for minimality

Q: How much redundancy is needed to get CCA security?

- OAEP (1994): $f((m\|0) \oplus G(r) \;\|\; r \oplus H((m\|0) \oplus G(r)))$

  not that Optimal; redundancy needed for INDCCA
- SAEP (2001): $f(r \;\|\; (m\|0) \oplus G(r))$

  tighter reduction, also needs redundancy and k big enough
- ZAEP: $f(r \,\|\, G(r) \oplus m)$

  tighter reduction, redundant-free

$$\mathrm{Dec}(c) : r \| t \leftarrow f_{sk}^{-1}(c); g \leftarrow G(r); \text{return } t \oplus g$$

The quest for minimality

Q: How much redundancy is needed to get CCA security?

0 (**Z**ero)

- OAEP (1994): $f((m\|0) \oplus G(r) \ \| \ r \oplus H((m\|0) \oplus G(r)))$

  not that Optimal; redundancy needed for INDCCA

- SAEP (2001): $f(r \ \| \ (m\|0) \oplus G(r))$

  tighter reduction, also needs redundancy and k big enough

- ZAEP: $f(r \ \| \ G(r) \oplus m)$

  tighter reduction, redundant-free

  $\mathrm{Dec}(c) : r\|t \leftarrow f_{sk}^{-1}(c); g \leftarrow G(r); \text{return } t \oplus g$

Retrospective

The good

- Deductive program verification with off-the-shelf tools
- Using SMT solvers is critical for scalability
- Hybrid approaches allow automated proofs and exploration

The bad

- No first-class support for program transformations
- Certification in Coq gradually out of synchronization

The ugly

- Instantiation
- Modular proofs

Perspective

Further topics

- Translation validation for cryptographic compilers
- Differential privacy
- Higher-order languages
- Decidability; decision procedures

Future directions

- Unwinding as transaction
- Leakage resilience; fault resilience
- Atlas of classical cryptography; interactive tutorials
- Synthesis of efficient implementations

Conclusion

- Working in program verification? Looking for a fresh topic?
- Cryptography!
- Exciting research at the crossroads of many areas
- Practically relevant and challenging examples
- Fun!

`http://easycrypt.gforge.inria.fr`

Conclusion

- Working in program verification? Looking for a fresh topic?
- Cryptography!
- Exciting research at the crossroads of many areas
- Practically relevant and challenging examples
- Fun!

`http://easycrypt.gforge.inria.fr`

Conclusion

- Working in program verification? Looking for a fresh topic?
- Cryptography!
- Exciting research at the crossroads of many areas
- Practically relevant and challenging examples
- Fun!

`http://easycrypt.gforge.inria.fr`

Conclusion

- Working in program verification? Looking for a fresh topic?
- Cryptography!
- Exciting research at the crossroads of many areas
- Practically relevant and challenging examples
- Fun!

`http://easycrypt.gforge.inria.fr`

Conclusion

- Working in program verification? Looking for a fresh topic?
- Cryptography!
- Exciting research at the crossroads of many areas
- Practically relevant and challenging examples
- Fun!

`http://easycrypt.gforge.inria.fr`