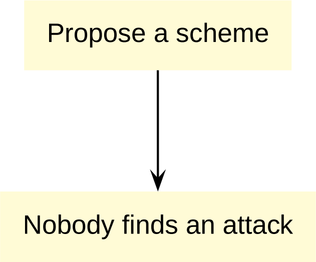
The next 700 cryptosystems

Gilles Barthe

IMDEA Software Institute, Madrid, Spain

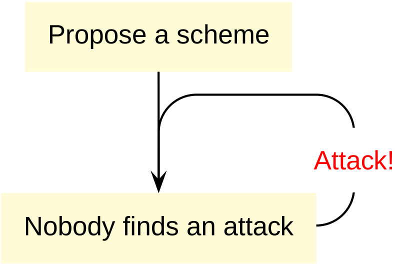
Based on joint work with:
Benjamin Grégoire (INRIA Sophia Antipolis, France)
César Kunz (U. Politecnica Madrid/IMDEA, Spain)
Yassine Lakhnech (U. de Grenoble/CNRS, France)
Santiago Zanella Béguelin (MSR Cambridge, UK)

Historical perspective



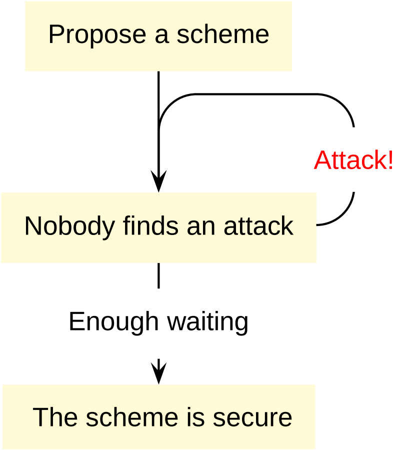
Before 1980

Historical perspective



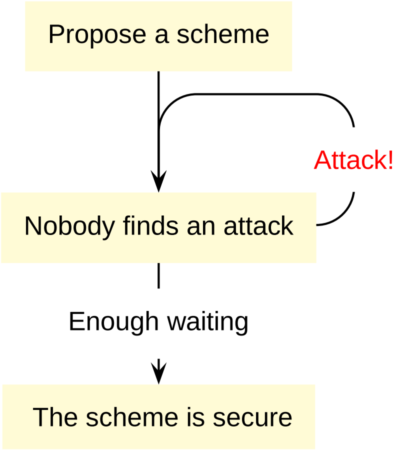
Before 1980

Historical perspective



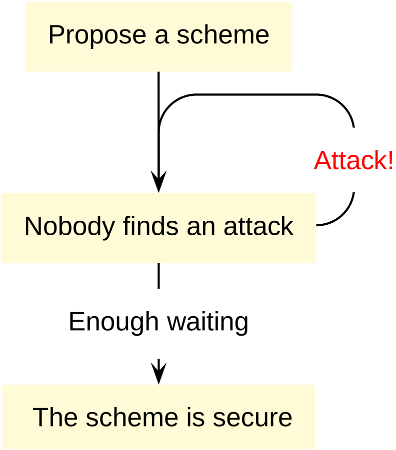
Before 1980

Historical perspective



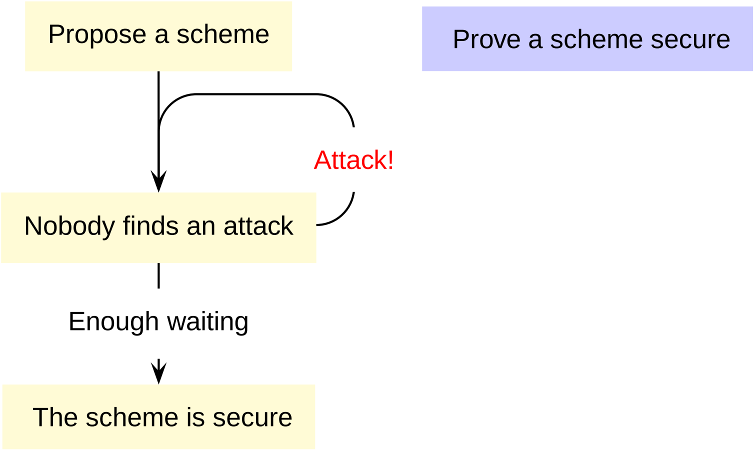
How much time is *enough*?

Historical perspective



1 year? 5 years? 10 years?

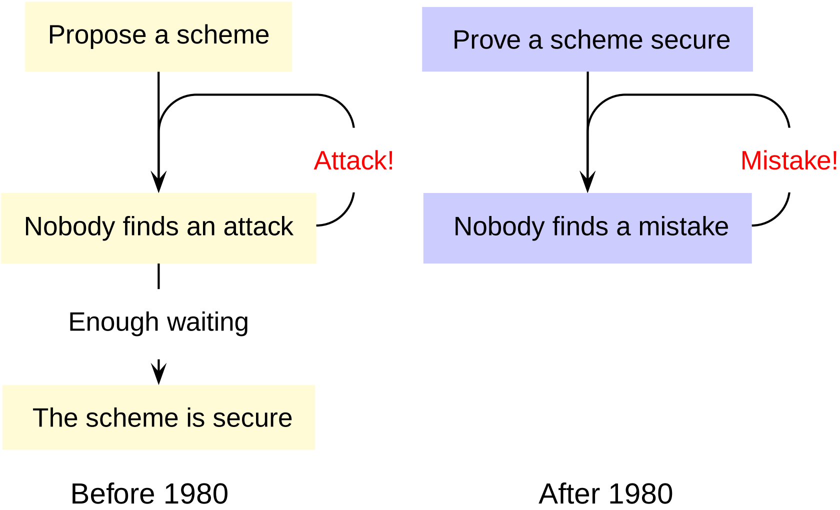
Historical perspective



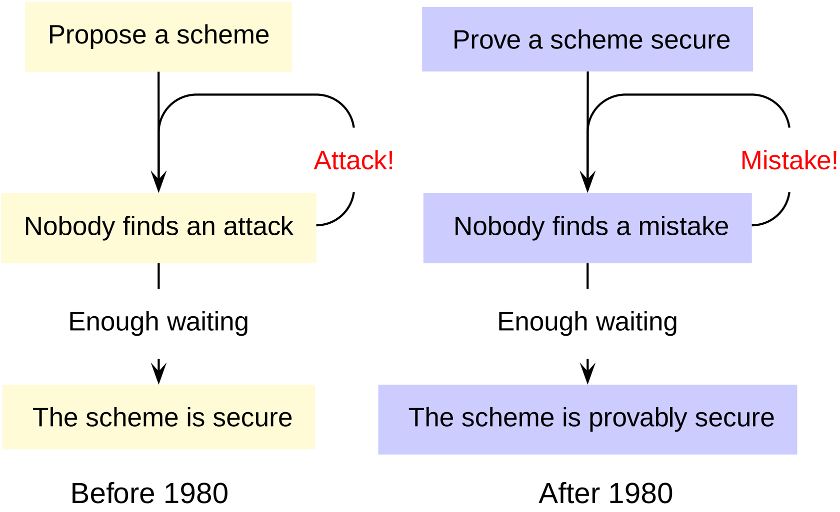
Before 1980

After 1980

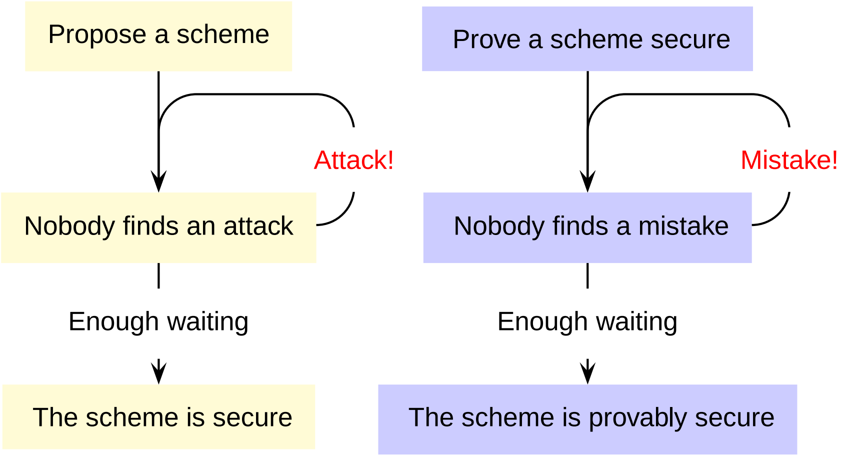
Historical perspective



Historical perspective

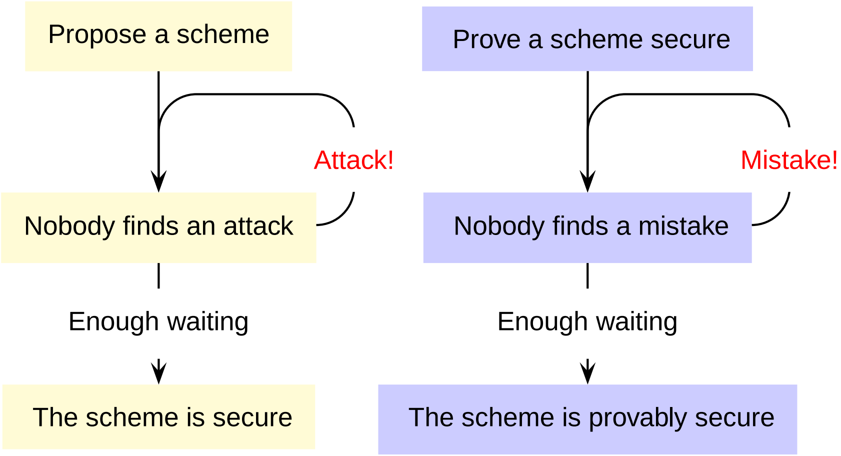


Historical perspective



How much time is *enough*?

Historical perspective



1 week? 1 year? 5 years? 10 years? 20 years?

A famous 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 $\mathrm{Dec}_{sk}(c)$:

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

A famous example: RSA-OAEP

Oracle $\mathrm{Enc}_{pk}(m)$:

$r \stackrel{\$}{\leftarrow} \{0,1\}^{k_0}$;
$s \leftarrow G(r) \oplus (m \| 0^{k_1})$;
$t \leftarrow H(s) \oplus r$;
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 $\perp$

Game IND-CCA2 :

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

A famous 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 $\perp$

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

Oracle $H(x)$:
if $x \notin \mathrm{dom}(L_H)$ then $L_H[x] \xleftarrow{\$} \{0,1\}^{k_0}$;
return $L_H[x]$

Game IND-CCA2 :
$(sk, pk) \leftarrow \mathcal{KG}(\ )$;
$(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(pk)$;
$b \xleftarrow{\$} \{0,1\}$;
$c^* \leftarrow \mathrm{Enc}(pk, m_b)$;
$b' \leftarrow \mathcal{A}_2(pk, c^*, \sigma)$;
return $b = b'$

Game POW :
$(sk, pk) \leftarrow \mathcal{KG}()$;
$y \xleftarrow{\$} \{0,1\}^{n+k_1}$;
$z \xleftarrow{\$} \{0,1\}^{k_0}$;
$y' \leftarrow \mathcal{I}(f_{pk}(y \| z))$;
return $y = y'$

A famous example: RSA-OAEP

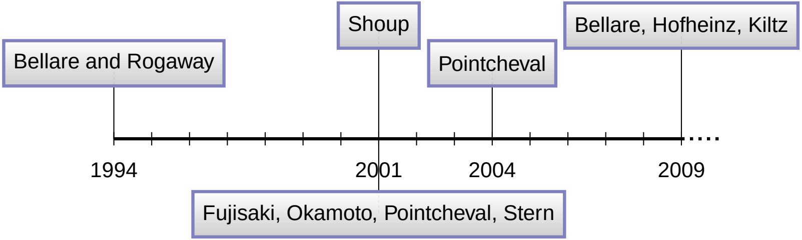
For every IND-CCA2 adversary $\mathcal{A}$ there exists an inverter $\mathcal{I}$ s.t.

$$\begin{aligned}\text{Adv}_{\text{IND-CCA2}(\mathcal{A})} &= \left|\Pr_{\text{IND-CCA2}}[b = b'] - \tfrac{1}{2}\right| \\ &\leq \textbf{Succ}_f^{\text{POW}}(\mathcal{I}) + \frac{3q_D q_G + q_D^2 + 4q_D + q_G}{2^{k_0}} + \frac{2q_D}{2^{k_1}}\end{aligned}$$

where

$$\textbf{Succ}_f^{\text{POW}} = \Pr_{\text{POW}}\left[y = y'\right]$$

A famous example: 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 improved bounds from 2004 proof

2012 Improved bound from 2010 proof

Provable security as program verification!

CertiCrypt/EasyCrypt project, 2006-

High-assurance cryptographic proofs

- based on rigorous methods from programming languages
  - program verification
  - compiler verification
- with machine support building upon off-the-shelf tools

Code-based game-playing proofs

(Bellare & Rogaway 2004, Halevi 2005)

Games as probabilistic programs

$$
\begin{array}{rcll}
\mathcal{C} & ::= & \mathcal{V} \leftarrow \mathcal{E} & \text{assignment} \\
& | & \mathcal{V} \xleftarrow{\$} \mathcal{D} & \text{random sampling} \\
& | & \mathcal{C};\ \mathcal{C} & \text{sequence} \\
& | & \text{if } \mathcal{E} \text{ then } \mathcal{C} \text{ else } \mathcal{C} & \text{conditional} \\
& | & \text{while } \mathcal{E} \text{ do } \mathcal{C} & \text{while loop} \\
& | & \mathcal{V} \leftarrow \mathcal{P}(\mathcal{E}, \ldots, \mathcal{E}) & \text{procedure call}
\end{array}
$$

- For cryptographers: rigorous notation for games
- In our work: rigorous justification of game-based proofs

$$
\begin{array}{rcll}
\mathcal{E} & ::= & \mathcal{E} \oplus \mathcal{E} & \text{xor} \\
& | & \mathcal{E} \parallel \mathcal{E} & \text{concatenation} \\
& & \ldots &
\end{array}
$$

The game-playing approach

(Shoup 2004, Bellare & Rogaway 2004, Halevi, 2005)

For every feasible adversary $\mathcal{A}$ against scheme **S** (wrt goal **G**) **there exists** a feasible adversary $\mathcal{B}$ against assumption **H** st

$$\Pr_{\mathsf{G}_a}[\mathcal{A} \text{ breaks } \mathbf{S}] \leq h(\Pr_{\mathsf{G}_h}[\mathcal{B} \text{ breaks } \mathbf{H}])$$

The game-playing approach

(Shoup 2004, Bellare & Rogaway 2004, Halevi, 2005)

For every feasible adversary $\mathcal{A}$ against scheme **S** (wrt goal **G**) **there exists** a feasible adversary $\mathcal{B}$ against assumption **H** st

$$\Pr_{\mathsf{G}_a}[\mathcal{A} \text{ breaks } \mathbf{S}] \leq h(\Pr_{\mathsf{G}_h}[\mathcal{B} \text{ breaks } \mathbf{H}])$$

Game G_a :	**Game** G_1 :		**Game** G_h :
$\ldots$	$\ldots$		$\ldots$
$\ldots \leftarrow \mathcal{A}(\ldots)$;	$\ldots$	$\cdots$	$\ldots \leftarrow \mathcal{B}(\ldots)$;
$\ldots$	$\ldots$		$\ldots$

$$\Pr_{\mathsf{G}_a}[\mathcal{A} \text{ breaks } \mathbf{S}] \quad \leq \quad h_1(\Pr_{\mathsf{G}_1}[E_1]) \leq \ldots \leq h(\Pr_{\mathsf{G}_h}[\mathcal{B} \text{ breaks } \mathbf{H}])$$

pRHL: a Relational Hoare Logic for pWHILE

(after Benton 2004)

- Judgment:

$$c_1 \sim c_2 : P \Rightarrow Q$$

  where P and Q are relations on memories

- Validity:

$$\models c_1 \sim c_2 : P \Rightarrow Q$$

  iff for all memories m_1 and m_2

$$(m_1, m_2) \models P \to (\llbracket c_1 \rrbracket_{m_1}, \llbracket c_2 \rrbracket_{m_2}) \models Q^\sharp$$

- Lifting $Q^\sharp$ asserts *existence* of maximal flow in flow network (beware of existential quantification)

pRHL captures common patterns of reasoning in crypto proofs

Conditionals

$$\frac{\models c_1 \sim c : P \wedge e\langle 1\rangle \Rightarrow Q \qquad \models c_2 \sim c : P \wedge \neg e\langle 1\rangle \Rightarrow Q}{\models \text{if } e \text{ then } c_1 \text{ else } c_2 \sim c : P \Rightarrow Q}$$

Assignment

$$\frac{}{\models x \leftarrow e \sim \text{nil} : Q\{x\langle 1\rangle := e\langle 1\rangle\} \Rightarrow Q}$$

Random assignment

$$\frac{f \text{ is 1-1 and } Q' \stackrel{\text{def}}{=} \forall v, Q\{x\langle 1\rangle := f\ v, x\langle 2\rangle := v\}}{\models x \stackrel{\$}{\leftarrow} A \sim x \stackrel{\$}{\leftarrow} A : Q' \Rightarrow Q}$$

Adversary calls

$$\frac{\forall \mathcal{O}.\ \models z \leftarrow \mathcal{O}(\vec{w}) \sim z \leftarrow \mathcal{O}(\vec{w}) : Q \wedge =_W \Rightarrow Q \wedge =_{\{z\}}}{\models x \leftarrow \mathcal{A}(\vec{y}) \sim x \leftarrow \mathcal{A}(\vec{y}) : Q \wedge =_Y \Rightarrow Q \wedge =_{\{x\}}}$$

Tool support and examples

CertiCrypt: formally verified Coq libraries

- Optimizations and probabilistic relational Hoare logic
- Verified against operational semantics based on ALEA

EasyCrypt: SMT-based verification tool

- Probabilistic relational Hoare logic
- Verification condition generation + why3 back-end
- Accessible to cryptographers

Examples

- Crypto: public-key encryption, block ciphers, signatures, hash designs, zero-knowledge proofs of knowledge, authenticated key exchange protocols
- Differential privacy: continuous statistics, approximation algorithms, synthetic databases, 2-party computation

A simple example: BR93 encryption

Game IND-CPA :	$\mathsf{Enc}_{pk}(m)$:
$(sk, pk) \leftarrow \mathcal{KG}(\ )$; $(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(pk)$; $b \xleftarrow{\$} \{0,1\}$; $c^* \leftarrow \mathsf{Enc}(pk, m_b)$; $b' \leftarrow \mathcal{A}_2(pk, c^*, \sigma)$; return $b = b'$	$r \xleftarrow{\$} \{0,1\}^\ell$; $s \leftarrow G(r) \oplus m$; $y \leftarrow f_{pk}(r) \,\|\, s$; return y

$G(x)$:
 if $x \notin \mathsf{dom}(L_G)$ 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

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

Step 1: failure event

Game G_0 :
$L_G \leftarrow \emptyset;\ Q_G \leftarrow [\,];$
$(sk, pk) \leftarrow \mathcal{KG}();$
$(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$r \xleftarrow{\$} \{0,1\}^{\ell};$
$g \leftarrow G(r);$
$s \leftarrow g \oplus m_b;$
$c^* \leftarrow f_{pk}(r) \,\|\, s;$
$b' \leftarrow \mathcal{A}_2(pk, c^*, \sigma);$

Game G_1 :
$L_G \leftarrow \emptyset;\ Q_G \leftarrow [\,];$
$(sk, pk) \leftarrow \mathcal{KG}();$
$(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$r \xleftarrow{\$} \{0,1\}^{\ell};$
$g \xleftarrow{\$} \{0,1\}^{k};$
$s \leftarrow g \oplus m_b;$
$c^* \leftarrow f_{pk}(r) \,\|\, s;$
$b' \leftarrow \mathcal{A}_2(pk, c^*, \sigma);$

Step 1: failure event

Game G_0 :
$L_G \leftarrow \emptyset;\ Q_G \leftarrow [\,];$
$(sk, pk) \leftarrow \mathcal{KG}();$
$(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$r \xleftarrow{\$} \{0,1\}^\ell;$
$g \leftarrow G(r);$
$s \leftarrow g \oplus m_b;$
$c^* \leftarrow f_{pk}(r) \,\|\, s;$
$b' \leftarrow \mathcal{A}_2(pk, c^*, \sigma);$

Game G_1 :
$L_G \leftarrow \emptyset;\ Q_G \leftarrow [\,];$
$(sk, pk) \leftarrow \mathcal{KG}();$
$(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$r \xleftarrow{\$} \{0,1\}^\ell;$
$g \xleftarrow{\$} \{0,1\}^k;$
$s \leftarrow g \oplus m_b;$
$c^* \leftarrow f_{pk}(r) \,\|\, s;$
$b' \leftarrow \mathcal{A}_2(pk, c^*, \sigma);$

The games are equivalent until the adversary queries G with r

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

Step 2: optimistic sampling

Game G_1 :
$L_G \leftarrow \emptyset;\ Q_G \leftarrow [\,];$
$(sk, pk) \leftarrow \mathcal{KG}();$
$(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$r \xleftarrow{\$} \{0,1\}^\ell;$
$g \leftarrow \{0,1\}^k;$
$s \leftarrow g \oplus m_b;$
$c^* \leftarrow f_{pk}(r) \| s;$
$b' \leftarrow \mathcal{A}_2(pk, c^*, \sigma);$

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

Step 2: optimistic sampling

Game G_1 :
$L_G \leftarrow \emptyset;\ Q_G \leftarrow [\,];$
$(sk, pk) \leftarrow \mathcal{KG}();$
$(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$r \xleftarrow{\$} \{0,1\}^\ell;$
$g \leftarrow \{0,1\}^k;$
$s \leftarrow g \oplus m_b;$
$c^* \leftarrow f_{pk}(r) \| s;$
$b' \leftarrow \mathcal{A}_2(pk, c^*, \sigma);$

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

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

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

Step 3: reduction

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

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

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

Step 3: reduction

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

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

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

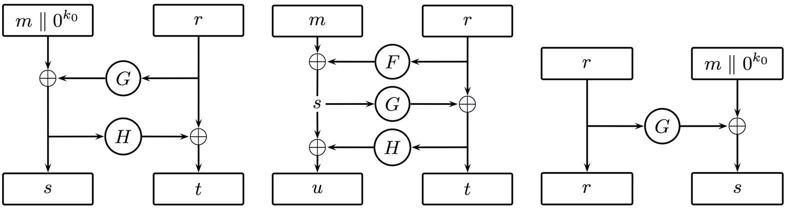
Inverter wins with probability $\frac{1}{q_G}$ if $r \in Q_G$, and 0 otherwise

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

Beyond OAEP

Over 100 variants of OAEP in the literature

- Are they all secure?
- Are there common patterns in proofs?
- Can proofs be automated?



(a) OAEP (b) 3-round OAEP (c) SAEP

The next 700 cryptosystems

(After Landin, 1966)

The question arises, do the idiosyncracies reflect basic logical properties of 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 cryptosystem is a point chosen from a well-mapped space, rather than a laboriously devised construction.

Generation

$\mathcal{E}$	$::=$	m	input message
	$\mid$	0	zero bitstring
	$\mid$	$\mathcal{R}$	uniform random bitstring
	$\mid$	$\mathcal{E} \oplus \mathcal{E}$	xor
	$\mid$	$\mathcal{E} \parallel \mathcal{E}$	concatenation
	$\mid$	$H(\mathcal{E})$	hash
	$\mid$	$f(\mathcal{E})$	trapdoor permutation

Filtering

Eliminate schemes that are not :

- invertible $f(r)$
- IND-CPA
  - is decryption possible without a key? $m \parallel f(r)$
  - is encryption randomized? $f(m)$
  - is randomness extractable without a key? $r \parallel f(m \oplus r)$
- IND-CCA2
  - is encryption malleable? $f(r) \| m \oplus G(r)$

Deducibility relation

$$\frac{e \vdash e_1 \quad e \vdash e_2}{e \vdash e_1 \| e_2}[\text{Conc}] \quad \frac{e \vdash e_1 \quad e \vdash e_2}{e \vdash (e_1 \oplus e_2)\downarrow}[\text{Xor}] \quad \frac{e \vdash e'}{e \vdash H(e')}[\text{H}]$$

$$\frac{e \vdash e_1 \| e_2}{e \vdash e_i}[\text{Proj}_i] \quad \frac{e \vdash e'}{e \vdash f(e')}[\text{f}] \quad \boxed{\frac{e \vdash f(e')}{e \vdash e'}[\text{finv}]}$$

Chosen-plaintext security

Step 1: proof finding

Optimistic sampling	Replace $e \oplus r$, where r is fresh, by r
Permutation	Replace $f(r)$, where r is fresh, by r
Failure event	Replace $H(e)$ by fresh r
Probability	Compute probability of $b = b'$ or $e \in L$

Step 2: proof generation and proof checking

Generation	Output EasyCrypt file
Checking	Independent verification of EasyCrypt file (< 120 s)

Chosen-ciphertext security

If decryption oracle is of the form

$$\begin{array}{l} u_0 \ldots u_n, t_n \leftarrow \mathrm{Extract}(c); \\ \text{for } i \leftarrow n \ldots 1 \text{ do} \quad t_{i-1} \leftarrow u_i \oplus H_i(t_i); \\ \text{if } \mathrm{Test}(\vec{t}, \vec{u}, c) \text{ then } \mathrm{GetMsg}(\vec{t}) \text{ else return } \perp \end{array}$$

and IND-CPA and IND-CPA-like properties then IND-CCA2

Proof intuition

- Plaintext-awareness: infeasible to get valid ciphertext otherwise than by encrypting a known plaintext

  $$\text{IND-CPA + plaintext-awareness} \Longrightarrow \text{IND-CCA2}$$

- Successively modify decryption oracle to reject ciphertexts for which corresponding hash queries have not been made
- Yields plaintext extractor and reduction to IND-CPA
- IND-CPA-like properties provide bounds for failure events

Experiments

- Generated over 100,000 schemes
- Filters leave 4,500 schemes
- Proved IND-CPA security of 3,000 schemes
- Proved IND-CCA2 security of 2,000 schemes

Assessment

- Systematic exploration of design space
- Tens of papers automated
- IND-CCA2 checker fails on redundant-free schemes

ZAEP

(with David Pointcheval)

Two minimal schemes

$$\mathrm{BR93}: \ f(r) \parallel (G(r) \oplus m) \qquad \mathrm{ZAEP}: \ f(r \parallel G(r) \oplus m)$$

ZAEP is redundant-free

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

INDCCA Security of ZAEP for RSA exponent 2 and 3

$$\left| \Pr_{\text{IND-CCA2}}[b = b'] - \frac{1}{2} \right| \leq \mathbf{Succ}_f^{\mathrm{OW}}(\mathcal{I}) + \frac{q_D}{2^n}$$

Based on existence of two efficient algorithms:

- CIE: given $f(r, s_1)$, $f(r, s_2)$ with $s_1 \neq s_2$, returns s_1, s_2 and r
- SIE: given $f(r, s)$ and r returns s

Conclusion

High-assurance cryptographic proofs

- Rigorous proofs using PL techniques (pRHL)
- Independent verification and (in principle) certified proofs

Automated generation of schemes and proofs

- public-key encryption
- zero-knowledge compilers

Directions for future work

- Program logics, decision procedures, relational invariant inference...
- Probabilistic encapsulation, modularity
- Towards implementations
- Synthesis: revisit classical cryptography