

Towards Language-Based Cryptographic Proofs

Santiago Zanella Béguelin
Gilles Barthe Benjamin Grégoire Romain Janvier



INRIA Sophia Antipolis
INRIA-Microsoft Research Joint Centre

2007.04.19

Increasing complexity in cryptographic proofs

+

Unmanageable numbers of them appearing in articles

+

No one willing to verify boring, repetitive, handmade proofs

Subtle errors in supposedly peer-reviewed cryptographic proofs

Increasing complexity in cryptographic proofs

+

Unmanageable numbers of them appearing in articles

+

No one willing to verify boring, repetitive, handmade proofs

Subtle errors in supposedly peer-reviewed cryptographic proofs

Increasing complexity in cryptographic proofs

+

Unmanageable numbers of them appearing in articles

+

No one willing to verify boring, repetitive, handmade proofs

Subtle errors in supposedly peer-reviewed cryptographic proofs

Increasing complexity in cryptographic proofs

+

Unmanageable numbers of them appearing in articles

+

No one willing to verify boring, repetitive, handmade proofs

Subtle errors in supposedly peer-reviewed 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.
- *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
- *Security proofs in cryptography may be organized as sequences of games [...] this can be a useful tool in taming the complexity of security proofs that might otherwise become so messy, complicated, and subtle as to be nearly impossible to verify*
V. Shoup

- M. Bellare and P. Rogaway. *Code-based game-playing proofs and the security of triple encryption*. EuroCrypt 2006.
- V. Shoup. *Sequences of games: a tool for taming complexity in security proofs*. Cryptology ePrint Archive, 2004/332.
- S. Halevi. *A plausible approach to computer-aided cryptographic proofs*. Cryptology ePrint Archive, 2005/181.

The general idea

- Describe security using a game played between a challenger and an adversary. May be encoded as a program in a probabilistic programming language,
- Pick an initial game, transform it stepwise preserving (up to a negligible factor) or increasing the winning probability of the adversary,
- Bound this probability in the final game.
- Argue that the bound also holds for the initial game
- For all this, rely on a well-defined set of hypotheses (e.g. Decisional Diffie-Hellman) and properties of primitives (Ideal-cipher, one-way function)

The general idea

- Describe security using a game played between a challenger and an adversary. May be encoded as a program in a probabilistic programming language,
- Pick an initial game, transform it stepwise preserving (up to a negligible factor) or increasing the winning probability of the adversary,
- Bound this probability in the final game.
- Argue that the bound also holds for the initial game
- For all this, rely on a well-defined set of hypotheses (e.g. Decisional Diffie-Hellman) and properties of primitives (Ideal-cipher, one-way function)

Caveat: Game-playing doesn't substitute probabilistic reasoning but supplements it.

Our objective is to build a certified tool for checking game-playing proofs, on top of a general purpose proof assistant (Coq)

- The tool provides independently checkable certificates that justify transitions between games
- Security goals, properties and hypotheses are explicit. The latter can be taken from a standard library.
- The “mundane” and “innovative” parts of the proofs can be justified formally in a unified formalism.

Disclaimer: we are (currently) not interested in

- Discovering the sequence of games,
- user interface

A probabilistic WHILE programming language

$$\begin{aligned} \mathcal{C} \ni c &::= \text{skip} \\ &| x \leftarrow e \\ &| x \stackrel{\$}{\leftarrow} T \\ &| \text{while } e \text{ do } c \\ &| \text{if } e \text{ then } c_1 \text{ else } c_2 \\ &| c_1 ; c_2 \\ &| x \leftarrow p(e_1, e_2, \dots) \end{aligned}$$

Values may be natural numbers, booleans, bitstrings, etc.

Probabilistic programs may be interpreted as distribution transformers.

Our semantics maps a command c and an initial state σ to the expected value operator over the distribution of states where the execution c halts starting from σ

$$\llbracket \cdot \rrbracket : \mathcal{C} \rightarrow \mathcal{S} \rightarrow (\mathcal{S} \rightarrow [0, 1]) \rightarrow [0, 1]$$

Intuitively,

$$\llbracket c \rrbracket \sigma f = \sum_{\sigma' \in \mathcal{S}} f(\sigma') \Pr[\langle c, \sigma \rangle \downarrow \sigma']$$

Instead of defining the semantic function directly, we rely on a frame-base small-step semantics.

We define $\mathcal{D}_A = (A \rightarrow [0, 1]) \rightarrow [0, 1]$.

$\llbracket \cdot \rrbracket_1 : \mathcal{C} \rightarrow \mathcal{S} \rightarrow \mathcal{D}_{\mathcal{S}}$ is the frame-based small-step semantics

$\llbracket \cdot \rrbracket_n : \mathcal{C} \rightarrow \mathcal{S} \rightarrow \mathcal{D}_{\mathcal{S}}$ is the n -unfold of $\llbracket \cdot \rrbracket_1$

$\llbracket \cdot \rrbracket : \mathcal{C} \rightarrow \mathcal{M} \rightarrow \mathcal{D}_{\mathcal{M}}$ is defined as the LUB of $\llbracket \cdot \rrbracket_n$, measuring the function on memories of final configurations reachable in at most n steps.

$$\llbracket c \rrbracket \mu f = \text{lub} (\lambda n \cdot \llbracket c \rrbracket_n \mu f!)$$

Where $f! \sigma'$ takes the value of f on the memory of σ' if it is a final configuration and 0 otherwise.

The least upper bound is guaranteed to exist and corresponds to the limit when restricted to monotone sequences.

Since $\llbracket c \rrbracket_n \mu !f$ is increasing, the semantics is well defined.

- Game-playing cryptographic proofs try to bound the winning probability of an adversary often by proving indistinguishability between a scheme and an ideal version of it. Observational equivalence is key for this kind of proofs
- Our definition of observational equivalence satisfies congruence properties that allow to relate two programs under different contexts.
- Although our definition is semantical, we derive syntactic criteria for deciding observational equivalence and prove them correct wrt the semantical definition.

Definition (Indistinguishable functions)

We say that two functions $f, g : \mathcal{M} \rightarrow A$ are indistinguishable wrt a relation $R \subseteq \mathcal{M} \times \mathcal{M}$ and denote it as $f \simeq_R g$ iff

$$\forall (\mu_1, \mu_2) \in R \cdot f \mu_1 = g \mu_2$$

Definition (Observational equivalence)

Let $P, Q \subseteq \mathcal{M} \times \mathcal{M}$ be PER over memories, we say that c_1 is observational equivalent to c_2 wrt to the input relation P and the output relation Q and denote it $c_1 \simeq_Q c_2$ iff,

$$\forall (\mu_1, \mu_2) \in P; f, g \in \mathcal{M} \rightarrow [0, 1]. \\ f \simeq_Q g \Rightarrow \llbracket c_1 \rrbracket \mu_1 f = \llbracket c_2 \rrbracket \mu_2 g$$

Observational equivalence properties

$$\frac{c_1 P \simeq_Q c_2}{c_2 P \simeq_Q c_1} \text{ sym}$$

$$\frac{c_1 P \simeq_Q c_2 \quad c_2 P \simeq_Q c_2}{c_1 P \simeq_Q c_3} \text{ trans}$$

$$\frac{c_1 P \simeq_Q c_2 \quad P' \subseteq P}{c_1 P' \simeq_Q c_2} \text{ str}$$

$$\frac{c_1 P \simeq_Q c_2 \quad Q \subseteq Q'}{c_1 P \simeq_{Q'} c_2} \text{ weak}$$

$$\frac{c_1 P \simeq_Q c'_1 \quad c_2 Q \simeq_R c'_2}{c_1; c_2 P \simeq_R c'_1; c'_2} \text{ seq}$$

$$\frac{c_1 P|_e \simeq_Q c'_1 \quad c_2 P|_{\neg e} \simeq_Q c'_2 \quad \llbracket e \rrbracket \simeq_P \llbracket e' \rrbracket}{\text{if } e \text{ then } c_1 \text{ else } c_2 P \simeq_Q \text{if } e' \text{ then } c'_1 \text{ else } c'_2} \text{ cond}$$

...

- Algebraic manipulations

- substitute $s_1 \stackrel{\$}{\leftarrow} \{0, 1\}^n; s_2 \leftarrow s_1 \oplus t$ for

- $s_2 \stackrel{\$}{\leftarrow} \{0, 1\}^n; s_1 \leftarrow s_2 \oplus t$

- substitute $h_1 \leftarrow g^{u_1}; h_2 \leftarrow h_1^{u_2}$ for $h_1 \leftarrow g^{u_1}; h_2 \leftarrow g^{u_1 u_2}$

- Code motion

- Constant propagation

- Dead-code elimination

- Inlining of procedure calls

- Failure events:

- Introduction & elimination

- Equivalent-until-failure games

- Derandomization

- replace $x \stackrel{\$}{\leftarrow} t; c$ with $x \leftarrow v; c$ where v maximizes over t the probability of a failure event

IND-CPA security of a asymmetric encryption scheme

Definition (Asymmetric encryption scheme)

A triple of algorithms $(\mathcal{K}, \mathcal{E}, \mathcal{D})$

\mathcal{K}_η	: Coins \rightarrow Key * Key	Key generation
\mathcal{E}	: Key \times Plaintext \times Coins \rightarrow Ciphertext	Encryption
\mathcal{D}	: Key \times Ciphertext \rightarrow Plaintext	Decryption

where $\forall(pk, sk) = \mathcal{K}_\eta(r), m, \phi = \mathcal{E}(pk, m) \Rightarrow m = \mathcal{D}(sk, \phi)$

A game-playing proof of IND-CPA for an asymmetric encryption scheme begins with a game like

$$\begin{aligned}(pk, sk) &\leftarrow \mathcal{K}_\eta; (m_0, m_1) \leftarrow A_1(pk); \\ b &\stackrel{\$}{\leftarrow} \{0, 1\}; \phi \leftarrow \mathcal{E}(pk, m_b); \\ \hat{b} &\leftarrow A_2(m_0, m_1, pk, \phi)\end{aligned}$$

If the probability of the event $\hat{b} = b$ after the execution can be bound by a *negligible* function of η , the game is IND-CPA secure.

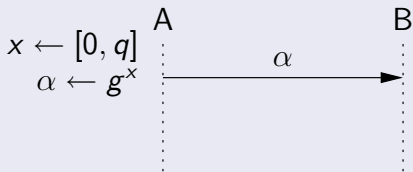
Let G be a cyclic group of prime order q generated by γ

The scheme

ElGamal encryption scheme

Let G be a cyclic group of prime order q generated by γ

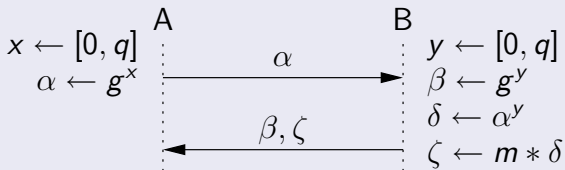
The scheme



ElGamal encryption scheme

Let G be a cyclic group of prime order q generated by γ

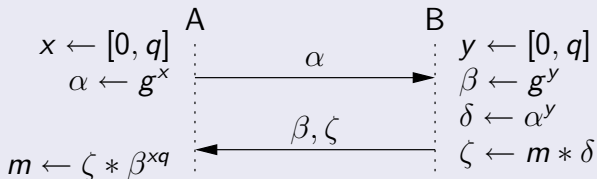
The scheme



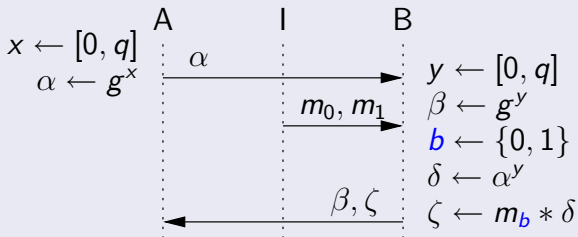
ElGamal encryption scheme

Let G be a cyclic group of prime order q generated by γ

The scheme



The Adversary chooses the secret messages



The Adversary must guess the value of b .

The two initial games

For any polynomial adversary $(\mathcal{A}, \mathcal{A}')$, we want to prove that:

Game El Gamal⁰:

```
x ←$- q; α ← gx;
(m0, m1) ← A((α));
y ←$- q; β ← gy;
δ ← αy;
ζ ← δ * m0;
d ← A'((α, β, ζ))
```

Game El Gamal¹:

```
(g, q)
 $\underset{d}{\sim}$ 
x ←$- q; α ← gx;
(m0, m1) ← A((α));
y ←$- q; β ← gy;
δ ← αy;
ζ ← δ * m1;
d ← A'((α, β, ζ))
```

The order of the group must be exponential in the security parameter.

The proof as a sequence of games

$$\begin{aligned}x &\leftarrow \$-q; \alpha \leftarrow g^{\wedge}x; \\(m_0, m_1) &\leftarrow \mathcal{A}((\alpha)) ; \\y &\leftarrow \$-q; \beta \leftarrow g^{\wedge}y; \\\delta &\leftarrow \alpha^{\wedge}y; \\\zeta &\leftarrow \delta * m_0; \\d &\leftarrow \mathcal{A}'((\alpha, \beta, \zeta))\end{aligned}$$
$$\begin{array}{c} (g, q) \\ \sim \\ d \end{array}$$
$$\begin{aligned}x &\leftarrow \$-q; \alpha \leftarrow g^{\wedge}x; \\(m_0, m_1) &\leftarrow \mathcal{A}((\alpha)) ; \\y &\leftarrow \$-q; \beta \leftarrow g^{\wedge}y; \\\delta &\leftarrow \alpha^{\wedge}y; \\\zeta &\leftarrow \delta * m_1; \\d &\leftarrow \mathcal{A}'((\alpha, \beta, \zeta))\end{aligned}$$

The proof as a sequence of games

$$\begin{aligned}x &<\$- q; \alpha \leftarrow g^x; \\(m_0, m_1) &\leftarrow \mathcal{A}((\alpha)) ; \\y &<\$- q; \beta \leftarrow g^y; \\ \delta &\leftarrow \alpha^y; \\ \zeta &\leftarrow \delta * m_0; \\ d &\leftarrow \mathcal{A}'((\alpha, \beta, \zeta))\end{aligned}$$
$$\simeq$$
$$\begin{aligned}x &<\$- q; \alpha \leftarrow g^x; \\(m_0, m_1) &\leftarrow \mathcal{A}((\alpha)) ; \\y &<\$- q; \beta \leftarrow g^y; \\z &<\$- q; \delta \leftarrow g^z; \\ \zeta &\leftarrow \delta * m_0; \\ d &\leftarrow \mathcal{A}'((\alpha, \beta, \zeta))\end{aligned}$$
$$\begin{aligned}x &<\$- q; \alpha \leftarrow g^x; \\(m_0, m_1) &\leftarrow \mathcal{A}((\alpha)) ; \\y &<\$- q; \beta \leftarrow g^y; \\ \delta &\leftarrow \alpha^y; \\ \zeta &\leftarrow \delta * m_1; \\ d &\leftarrow \mathcal{A}'((\alpha, \beta, \zeta))\end{aligned}$$
$$\simeq$$
$$\begin{aligned}x &<\$- q; \alpha \leftarrow g^x; \\(m_0, m_1) &\leftarrow \mathcal{A}((\alpha)) ; \\y &<\$- q; \beta \leftarrow g^y; \\z &<\$- q; \delta \leftarrow g^y; \\ \zeta &\leftarrow \delta * m_1; \\ d &\leftarrow \mathcal{A}'((\alpha, \beta, \zeta))\end{aligned}$$

The proof as a sequence of games

$$\begin{aligned}x &\leftarrow \$-q; \alpha \leftarrow g^{\wedge}x; \\(m_0, m_1) &\leftarrow \mathcal{A}((\alpha)) ; \\y &\leftarrow \$-q; \beta \leftarrow g^{\wedge}y; \\ \delta &\leftarrow \alpha^{\wedge}y; \\ \zeta &\leftarrow \delta * m_0; \\d &\leftarrow \mathcal{A}'((\alpha, \beta, \zeta))\end{aligned}$$
 \simeq
$$\begin{aligned}x &\leftarrow \$-q; \alpha \leftarrow g^{\wedge}x; \\(m_0, m_1) &\leftarrow \mathcal{A}((\alpha)) ; \\y &\leftarrow \$-q; \beta \leftarrow g^{\wedge}y; \\z &\leftarrow \$-q; \delta \leftarrow g^{\wedge}z; \\ \zeta &\leftarrow \delta * m_0; \\d &\leftarrow \mathcal{A}'((\alpha, \beta, \zeta))\end{aligned}$$
 \approx
$$\begin{aligned}x &\leftarrow \$-q; \alpha \leftarrow g^{\wedge}x; \\(m_0, m_1) &\leftarrow \mathcal{A}((\alpha)) ; \\y &\leftarrow \$-q; \beta \leftarrow g^{\wedge}y; \\z &\leftarrow \$-q; \zeta \leftarrow g^{\wedge}y; \\d &\leftarrow \mathcal{A}'((\alpha, \beta, \zeta))\end{aligned}$$
$$\begin{aligned}x &\leftarrow \$-q; \alpha \leftarrow g^{\wedge}x; \\(m_0, m_1) &\leftarrow \mathcal{A}((\alpha)) ; \\y &\leftarrow \$-q; \beta \leftarrow g^{\wedge}y; \\ \delta &\leftarrow \alpha^{\wedge}y; \\ \zeta &\leftarrow \delta * m_1; \\d &\leftarrow \mathcal{A}'((\alpha, \beta, \zeta))\end{aligned}$$
 \simeq
$$\begin{aligned}x &\leftarrow \$-q; \alpha \leftarrow g^{\wedge}x; \\(m_0, m_1) &\leftarrow \mathcal{A}((\alpha)) ; \\y &\leftarrow \$-q; \beta \leftarrow g^{\wedge}y; \\z &\leftarrow \$-q; \delta \leftarrow g^{\wedge}y; \\ \zeta &\leftarrow \delta * m_1; \\d &\leftarrow \mathcal{A}'((\alpha, \beta, \zeta))\end{aligned}$$
 \approx

The proof as a sequence of games

$$\begin{array}{ccc} \begin{array}{l} x \leftarrow \$-q; \alpha \leftarrow g^x; \\ (m_0, m_1) \leftarrow \mathcal{A}(\alpha) ; \\ y \leftarrow \$-q; \beta \leftarrow g^y; \\ \delta \leftarrow \alpha^y; \\ \zeta \leftarrow \delta * m_0; \\ d \leftarrow \mathcal{A}'(\alpha, \beta, \zeta) \end{array} & \begin{array}{c} (g, q) \\ \approx \\ d \end{array} & \begin{array}{l} x \leftarrow \$-q; \alpha \leftarrow g^x; \\ (m_0, m_1) \leftarrow \mathcal{A}(\alpha) ; \\ y \leftarrow \$-q; \beta \leftarrow g^y; \\ \delta \leftarrow \alpha^y; \\ \zeta \leftarrow \delta * m_1; \\ d \leftarrow \mathcal{A}'(\alpha, \beta, \zeta) \end{array} \\ \approx & & \approx \end{array}$$

$$\begin{array}{ccc} \begin{array}{l} x \leftarrow \$-q; \alpha \leftarrow g^x; \\ (m_0, m_1) \leftarrow \mathcal{A}(\alpha) ; \\ y \leftarrow \$-q; \beta \leftarrow g^y; \\ \mathbf{z} \leftarrow \$-q; \delta \leftarrow g^z; \\ \zeta \leftarrow \delta * m_0; \\ d \leftarrow \mathcal{A}'(\alpha, \beta, \zeta) \end{array} & & \begin{array}{l} x \leftarrow \$-q; \alpha \leftarrow g^x; \\ (m_0, m_1) \leftarrow \mathcal{A}(\alpha) ; \\ y \leftarrow \$-q; \beta \leftarrow g^y; \\ \mathbf{z} \leftarrow \$-q; \delta \leftarrow g^y; \\ \zeta \leftarrow \delta * m_1; \\ d \leftarrow \mathcal{A}'(\alpha, \beta, \zeta) \end{array} \\ \approx & & \approx \end{array}$$

$$\begin{array}{l} x \leftarrow \$-q; \alpha \leftarrow g^x; \\ (m_0, m_1) \leftarrow \mathcal{A}(\alpha) ; \\ y \leftarrow \$-q; \beta \leftarrow g^y; \\ \mathbf{z} \leftarrow \$-q; \zeta \leftarrow g^y; \\ d \leftarrow \mathcal{A}'(\alpha, \beta, \zeta) \end{array}$$

The Diffie-Hellman hypothesis

For any polynomial adversary \mathcal{B} ,

Game ddh^0 :

$x \leftarrow \mathbb{Z}_q; \alpha \leftarrow g^x;$

$y \leftarrow \mathbb{Z}_q; \beta \leftarrow g^y;$

$\delta \leftarrow \alpha^y;$

$d \leftarrow \mathcal{B}((\alpha, \beta, \delta))$

Game ddh^1 :

$x \leftarrow \mathbb{Z}_q; \alpha \leftarrow g^x;$

$y \leftarrow \mathbb{Z}_q; \beta \leftarrow g^y;$

$z \leftarrow \mathbb{Z}_q; \delta \leftarrow g^z;$

$d \leftarrow \mathcal{B}((\alpha, \beta, \delta))$

(g, q)
 $\stackrel{\sim}{d}$

The Diffie-Hellman reduction

Game El Gamal⁰:

$x \leftarrow \mathbb{Z}_q; \alpha \leftarrow g^x;$
 $(m_0, m_1) \leftarrow \mathcal{A}((\alpha));$
 $y \leftarrow \mathbb{Z}_q; \beta \leftarrow g^y;$
 $\delta \leftarrow \alpha^y;$
 $\zeta \leftarrow \delta * m_0;$
 $d \leftarrow \mathcal{A}'((\alpha, \beta, \zeta))$

\approx

Game ddh⁰:

$x \leftarrow \mathbb{Z}_q; \alpha \leftarrow g^x;$
 $y \leftarrow \mathbb{Z}_q; \beta \leftarrow g^y;$
 $\delta \leftarrow \alpha^y;$
 $d \leftarrow \mathcal{B}((\alpha, \beta, \delta))$

Game El Gamal¹:

(g, q)
 $\stackrel{\approx}{\sim}_d$
 $x \leftarrow \mathbb{Z}_q; \alpha \leftarrow g^x;$
 $(m_0, m_1) \leftarrow \mathcal{A}((\alpha));$
 $y \leftarrow \mathbb{Z}_q; \beta \leftarrow g^y;$
 $z \leftarrow \mathbb{Z}_q; \delta \leftarrow g^z;$
 $\zeta \leftarrow \delta * m_0;$
 $d \leftarrow \mathcal{A}'((\alpha, \beta, \zeta))$

\approx

Game ddh¹:

(g, q)
 $\stackrel{\approx}{\sim}_d$
 $x \leftarrow \mathbb{Z}_q; \alpha \leftarrow g^x;$
 $y \leftarrow \mathbb{Z}_q; \beta \leftarrow g^y;$
 $z \leftarrow \mathbb{Z}_q; \delta \leftarrow g^z;$
 $d \leftarrow \mathcal{B}((\alpha, \beta, \delta))$

Adversary $\mathcal{B}(\alpha', \beta', \delta') \rightarrow d'$
 $(m_0, m_1) \leftarrow \mathcal{A}((\alpha'));$
 $\zeta \leftarrow \delta' * m_0;$
 $d \leftarrow \mathcal{A}'((\alpha', \beta', \zeta))$

The Diffie-Hellman reduction

Game El Gamal⁰:

$x \leftarrow \mathbb{Z}_q; \alpha \leftarrow g^x;$
 $(m_0, m_1) \leftarrow \mathcal{A}((\alpha));$
 $y \leftarrow \mathbb{Z}_q; \beta \leftarrow g^y;$
 $\delta \leftarrow \alpha^y;$
 $\zeta \leftarrow \delta * m_0;$
 $d \leftarrow \mathcal{A}'((\alpha, \beta, \zeta))$

\approx

Game ddh⁰:

$x \leftarrow \mathbb{Z}_q; \alpha \leftarrow g^x;$
 $y \leftarrow \mathbb{Z}_q; \beta \leftarrow g^y;$
 $\delta \leftarrow \alpha^y;$
 $d \leftarrow \mathcal{B}((\alpha, \beta, \delta))$

Game El Gamal¹:

$x \leftarrow \mathbb{Z}_q; \alpha \leftarrow g^x;$
 $(m_0, m_1) \leftarrow \mathcal{A}((\alpha));$
 $y \leftarrow \mathbb{Z}_q; \beta \leftarrow g^y;$
 $z \leftarrow \mathbb{Z}_q; \delta \leftarrow g^z;$
 $\zeta \leftarrow \delta * m_0;$
 $d \leftarrow \mathcal{A}'((\alpha, \beta, \zeta))$

\approx

Game ddh¹:

$x \leftarrow \mathbb{Z}_q; \alpha \leftarrow g^x;$
 $y \leftarrow \mathbb{Z}_q; \beta \leftarrow g^y;$
 $z \leftarrow \mathbb{Z}_q; \delta \leftarrow g^z;$
 $d \leftarrow \mathcal{B}((\alpha, \beta, \delta))$

(g, q)
 \xrightarrow{d}

(g, q)
 \xrightarrow{d}

Adversary $\mathcal{B}(\alpha', \beta', \delta') \rightarrow d'$
 $(m_0, m_1) \leftarrow \mathcal{A}((\alpha'));$
 $\zeta \leftarrow \delta' * m_0;$
 $d \leftarrow \mathcal{A}'((\alpha', \beta', \zeta))$

The Diffie-Hellman reduction

Game El Gamal⁰:

$x \leftarrow \$-q; \alpha \leftarrow g^x;$
 $(m_0, m_1) \leftarrow \mathcal{A}((\alpha));$
 $y \leftarrow \$-q; \beta \leftarrow g^y;$
 $\delta \leftarrow \alpha^y;$
 $\zeta \leftarrow \delta * m_0;$
 $d \leftarrow \mathcal{A}'((\alpha, \beta, \zeta))$

\approx

Game ddh⁰:

$x \leftarrow \$-q; \alpha \leftarrow g^x;$
 $y \leftarrow \$-q; \beta \leftarrow g^y;$
 $\delta \leftarrow \alpha^y;$
 $d \leftarrow \mathcal{B}((\alpha, \beta, \delta))$

Game El Gamal¹:

$x \leftarrow \$-q; \alpha \leftarrow g^x;$
 $(m_0, m_1) \leftarrow \mathcal{A}((\alpha));$
 $y \leftarrow \$-q; \beta \leftarrow g^y;$
 $z \leftarrow \$-q; \delta \leftarrow g^z;$
 $\zeta \leftarrow \delta * m_0;$
 $d \leftarrow \mathcal{A}'((\alpha, \beta, \zeta))$

\approx

Game ddh¹:

$x \leftarrow \$-q; \alpha \leftarrow g^x;$
 $y \leftarrow \$-q; \beta \leftarrow g^y;$
 $z \leftarrow \$-q; \delta \leftarrow g^z;$
 $d \leftarrow \mathcal{B}((\alpha, \beta, \delta))$

(g, q)
 \xrightarrow{d}

(g, q)
 \xrightarrow{d}

Adversary $\mathcal{B}(\alpha', \beta', \delta') \rightarrow d'$
 $(m_0, m_1) \leftarrow \mathcal{A}((\alpha'));$
 $\zeta \leftarrow \delta' * m_0;$
 $d \leftarrow \mathcal{A}'((\alpha', \beta', \zeta))$

The Diffie-Hellman reduction

Game El Gamal⁰:

$x \leftarrow \mathbb{Z}_q; \alpha \leftarrow g^x;$
 $(m_0, m_1) \leftarrow \mathcal{A}((\alpha));$
 $y \leftarrow \mathbb{Z}_q; \beta \leftarrow g^y;$
 $\delta \leftarrow \alpha^y;$
 $\zeta \leftarrow \delta * m_0;$
 $d \leftarrow \mathcal{A}'((\alpha, \beta, \zeta))$

\approx

Game ddh⁰:

$x \leftarrow \mathbb{Z}_q; \alpha \leftarrow g^x;$
 $y \leftarrow \mathbb{Z}_q; \beta \leftarrow g^y;$
 $\delta \leftarrow \alpha^y;$
 $d \leftarrow \mathcal{B}((\alpha, \beta, \delta))$

Game El Gamal¹:

$x \leftarrow \mathbb{Z}_q; \alpha \leftarrow g^x;$
 $(m_0, m_1) \leftarrow \mathcal{A}((\alpha));$
 $y \leftarrow \mathbb{Z}_q; \beta \leftarrow g^y;$
 $z \leftarrow \mathbb{Z}_q; \delta \leftarrow g^z;$
 $\zeta \leftarrow \delta * m_0;$
 $d \leftarrow \mathcal{A}'((\alpha, \beta, \zeta))$

\approx

Game ddh¹:

$x \leftarrow \mathbb{Z}_q; \alpha \leftarrow g^x;$
 $y \leftarrow \mathbb{Z}_q; \beta \leftarrow g^y;$
 $z \leftarrow \mathbb{Z}_q; \delta \leftarrow g^z;$
 $d \leftarrow \mathcal{B}((\alpha, \beta, \delta))$

(g, q)
 \xrightarrow{d}

(g, q)
 \xrightarrow{d}

Adversary $\mathcal{B}(\alpha', \beta', \delta') \rightarrow d'$
 $(m_0, m_1) \leftarrow \mathcal{A}((\alpha'));$
 $\zeta \leftarrow \delta' * m_0;$
 $d \leftarrow \mathcal{A}'((\alpha', \beta', \zeta))$

The Diffie-Hellman reduction

Game El Gamal⁰:

$x \leftarrow \mathbb{Z}_q; \alpha \leftarrow g^x;$
 $(m_0, m_1) \leftarrow \mathcal{A}((\alpha));$
 $y \leftarrow \mathbb{Z}_q; \beta \leftarrow g^y;$
 $\delta \leftarrow \alpha^y;$
 $\zeta \leftarrow \delta * m_0;$
 $d \leftarrow \mathcal{A}'((\alpha, \beta, \zeta))$

\approx

Game ddh⁰:

$x \leftarrow \mathbb{Z}_q; \alpha \leftarrow g^x;$
 $y \leftarrow \mathbb{Z}_q; \beta \leftarrow g^y;$
 $\delta \leftarrow \alpha^y;$
 $d \leftarrow \mathcal{B}((\alpha, \beta, \delta))$

Game El Gamal¹:

(g, q)
 $\stackrel{\approx}{d}$
 $x \leftarrow \mathbb{Z}_q; \alpha \leftarrow g^x;$
 $(m_0, m_1) \leftarrow \mathcal{A}((\alpha));$
 $y \leftarrow \mathbb{Z}_q; \beta \leftarrow g^y;$
 $z \leftarrow \mathbb{Z}_q; \delta \leftarrow g^z;$
 $\zeta \leftarrow \delta * m_0;$
 $d \leftarrow \mathcal{A}'((\alpha, \beta, \zeta))$

\approx

Game ddh¹:

(g, q)
 $\stackrel{\approx}{d}$
 $x \leftarrow \mathbb{Z}_q; \alpha \leftarrow g^x;$
 $y \leftarrow \mathbb{Z}_q; \beta \leftarrow g^y;$
 $z \leftarrow \mathbb{Z}_q; \delta \leftarrow g^z;$
 $d \leftarrow \mathcal{B}((\alpha, \beta, \delta))$

Adversary $\mathcal{B}(\alpha', \beta', \delta') \rightarrow d'$

$(m_0, m_1) \leftarrow \mathcal{A}((\alpha'));$
 $\zeta \leftarrow \delta' * m_0;$
 $d \leftarrow \mathcal{A}'((\alpha', \beta', \zeta))$

The first step of the reduction

Game El Gamal⁰() \rightarrow d

```
x <$- q;  $\alpha$  <- g^x;  
(m0,m1) <-  $\mathcal{A}((\alpha))$ ;  
y <$- q;  $\beta$  <- g^y;  
 $\delta$  <-  $\alpha^y$ ;  
 $\zeta$  <-  $\delta * m0$ ;  
d <-  $\mathcal{A}'((\alpha, \beta, \zeta))$ 
```

$(g, q) \simeq_d$

Game ddh⁰() \rightarrow d

```
x <$- q;  $\alpha$  <- g^x;  
y <$- q;  $\beta$  <- g^y;  
 $\delta$  <-  $\alpha^y$ ;  
d <-  $\mathcal{B}((\alpha, \beta, \delta))$ 
```

Adversary $\mathcal{B}(\alpha', \beta', \delta') \rightarrow d'$

```
(m0,m1) <-  $\mathcal{A}((\alpha'))$ ;  
 $\zeta$  <-  $\delta' * m0$ ;  
d' <-  $\mathcal{A}'((\alpha', \beta', \zeta))$ 
```

The first step of the reduction

Game El Gamal⁰() \rightarrow d

```
x <$- q;  $\alpha$  <- g^x;  
(m0, m1) <-  $\mathcal{A}((\alpha))$ ;  
y <$- q;  $\beta$  <- g^y;  
 $\delta$  <-  $\alpha^y$ ;  
 $\zeta$  <-  $\delta * m0$ ;  
d <-  $\mathcal{A}'((\alpha, \beta, \zeta))$ 
```

$(g, q) \simeq_{(g, q, \alpha)}$

$(g, q, \alpha) \simeq_d$

Game ddh⁰() \rightarrow d

```
x <$- q;  $\alpha$  <- g^x;  
y <$- q;  $\beta$  <- g^y;  
 $\delta$  <-  $\alpha^y$ ;  
d <-  $\mathcal{B}((\alpha, \beta, \delta))$ 
```

Adversary $\mathcal{B}(\alpha', \beta', \delta') \rightarrow d'$

```
(m0, m1) <-  $\mathcal{A}((\alpha'))$ ;  
 $\zeta$  <-  $\delta' * m0$ ;  
d' <-  $\mathcal{A}'((\alpha', \beta', \zeta))$ 
```

The first step of the reduction

Game El Gamal⁰() \rightarrow d

x \leftarrow \mathbb{Z}_q ; $\alpha \leftarrow g^x$;

(m0, m1) \leftarrow $\mathcal{A}(\alpha)$;

y \leftarrow \mathbb{Z}_q ; $\beta \leftarrow g^y$;

$\delta \leftarrow \alpha^y$;

$\zeta \leftarrow \delta * m0$;

d \leftarrow $\mathcal{A}'(\alpha, \beta, \zeta)$

$(g, q, \alpha) \simeq_d$

Game ddh⁰() \rightarrow d

x \leftarrow \mathbb{Z}_q ; $\alpha \leftarrow g^x$;

y \leftarrow \mathbb{Z}_q ; $\beta \leftarrow g^y$;

$\delta \leftarrow \alpha^y$;

d \leftarrow $\mathcal{B}(\alpha, \beta, \delta)$

Adversary $\mathcal{B}(\alpha', \beta', \delta') \rightarrow d'$

(m0, m1) \leftarrow $\mathcal{A}(\alpha')$;

$\zeta \leftarrow \delta' * m0$;

d' \leftarrow $\mathcal{A}'(\alpha', \beta', \zeta)$

The first step of the reduction

Game El Gamal⁰() \rightarrow d

$x \leftarrow \$-q; \alpha \leftarrow g^x;$

$y \leftarrow \$-q; \beta \leftarrow g^y;$

$\delta \leftarrow \alpha^y;$

$(m_0, m_1) \leftarrow \mathcal{A}(\alpha);$

$\zeta \leftarrow \delta * m_0;$

$d \leftarrow \mathcal{A}'(\alpha, \beta, \zeta)$

$(g, q, \alpha) \simeq_{(g, q, \alpha, \beta, \delta)}$

$(g, q, \alpha, \beta, \delta) \simeq_d$

Game ddh⁰() \rightarrow d

$x \leftarrow \$-q; \alpha \leftarrow g^x;$

$y \leftarrow \$-q; \beta \leftarrow g^y;$

$\delta \leftarrow \alpha^y;$

$d \leftarrow \mathcal{B}(\alpha, \beta, \delta)$

Adversary $\mathcal{B}(\alpha', \beta', \delta') \rightarrow d'$

$(m_0, m_1) \leftarrow \mathcal{A}(\alpha');$

$\zeta \leftarrow \delta' * m_0;$

$d' \leftarrow \mathcal{A}'(\alpha', \beta', \zeta)$

The Diffie-Hellman reduction (2)

Game El Gamal⁰:

$x \leftarrow \mathbb{Z}_q; \alpha \leftarrow g^x;$
 $(m_0, m_1) \leftarrow \mathcal{A}((\alpha));$
 $y \leftarrow \mathbb{Z}_q; \beta \leftarrow g^y;$
 $\delta \leftarrow \alpha^y;$
 $\zeta \leftarrow \delta * m_0;$
 $d \leftarrow \mathcal{A}'((\alpha, \beta, \zeta))$

\approx

Game ddh⁰:

$x \leftarrow \mathbb{Z}_q; \alpha \leftarrow g^x;$
 $y \leftarrow \mathbb{Z}_q; \beta \leftarrow g^y;$
 $\delta \leftarrow \alpha^y;$
 $d \leftarrow \mathcal{B}((\alpha, \beta, \delta))$

Game El Gamal¹:

(g, q)
 $\stackrel{\approx}{\sim}_d$
 $x \leftarrow \mathbb{Z}_q; \alpha \leftarrow g^x;$
 $(m_0, m_1) \leftarrow \mathcal{A}((\alpha));$
 $y \leftarrow \mathbb{Z}_q; \beta \leftarrow g^y;$
 $z \leftarrow \mathbb{Z}_q; \delta \leftarrow g^z;$
 $\zeta \leftarrow \delta * m_0;$
 $d \leftarrow \mathcal{A}'((\alpha, \beta, \zeta))$

\approx

Game ddh¹:

(g, q)
 $\stackrel{\approx}{\sim}_d$
 $x \leftarrow \mathbb{Z}_q; \alpha \leftarrow g^x;$
 $y \leftarrow \mathbb{Z}_q; \beta \leftarrow g^y;$
 $z \leftarrow \mathbb{Z}_q; \delta \leftarrow g^z;$
 $d \leftarrow \mathcal{B}((\alpha, \beta, \delta))$

Adversary $\mathcal{B}(\alpha', \beta', \delta') \rightarrow d'$

$(m_0, m_1) \leftarrow \mathcal{A}((\alpha'));$
 $\zeta \leftarrow \delta' * m_0;$
 $d \leftarrow \mathcal{A}'((\alpha', \beta', \zeta))$

The last step of reduction

Game El Gamal₁⁰() \rightarrow d

```
x <$- q;  $\alpha$  <- g^x;  
(m0, m1) <-  $\mathcal{A}((\alpha))$ ;  
y <$- q;  $\beta$  <- g^y;  
z <$- q;  $\delta$  <- g^z;  
 $\zeta$  <-  $\delta * m0$ ;  
d <-  $\mathcal{A}'((\alpha, \beta, \zeta))$ 
```

$(g, q) \simeq_d$

Game El Gamal₂() \rightarrow d

```
x <$- q;  $\alpha$  <- g^x;  
(m0, m1) <-  $\mathcal{A}((\alpha))$ ;  
y <$- q;  $\beta$  <- g^y;  
z <$- q;  $\zeta$  <- g^z;  
d <-  $\mathcal{A}'((\alpha, \beta, \zeta))$ 
```

The last step of reduction

Game El Gamal₁⁰() → d

```
x <$- q; α <- g^x;  
(m0, m1) <- A((α));  
y <$- q; β <- g^y;  
z <$- q; δ <- g^z;  
ζ <- δ * m0;  
d <- A'((α, β, ζ))
```

Game El Gamal₂() → d

```
x <$- q; α <- g^x;  
(m0, m1) <- A((α));  
y <$- q; β <- g^y;  
z <$- q; ζ <- g^z;  
d <- A'((α, β, ζ))
```

$(g, q) \simeq_{(g, q, \alpha, m0)}$

$(g, q, \alpha, m0, \beta) \simeq_{(g, q, \alpha, \beta, \zeta)}$

$(g, q, \alpha, \beta, \zeta) \simeq_d$

The proof as a sequence of games

$$\begin{array}{ccc}
 \begin{array}{l}
 x \leftarrow \$-q; \alpha \leftarrow g^x; \\
 (m_0, m_1) \leftarrow \mathcal{A}(\alpha) ; \\
 y \leftarrow \$-q; \beta \leftarrow g^y; \\
 \delta \leftarrow \alpha^y; \\
 \zeta \leftarrow \delta * m_0; \\
 d \leftarrow \mathcal{A}'(\alpha, \beta, \zeta)
 \end{array}
 &
 \begin{array}{c}
 (g, q) \\
 \approx \\
 d
 \end{array}
 &
 \begin{array}{l}
 x \leftarrow \$-q; \alpha \leftarrow g^x; \\
 (m_0, m_1) \leftarrow \mathcal{A}(\alpha) ; \\
 y \leftarrow \$-q; \beta \leftarrow g^y; \\
 \delta \leftarrow \alpha^y; \\
 \zeta \leftarrow \delta * m_1; \\
 d \leftarrow \mathcal{A}'(\alpha, \beta, \zeta)
 \end{array}
 \end{array}$$

$$\begin{array}{ccc}
 \begin{array}{l}
 x \leftarrow \$-q; \alpha \leftarrow g^x; \\
 (m_0, m_1) \leftarrow \mathcal{A}(\alpha) ; \\
 y \leftarrow \$-q; \beta \leftarrow g^y; \\
 z \leftarrow \$-q; \delta \leftarrow g^z; \\
 \zeta \leftarrow \delta * m_0; \\
 d \leftarrow \mathcal{A}'(\alpha, \beta, \zeta)
 \end{array}
 &
 &
 \begin{array}{l}
 x \leftarrow \$-q; \alpha \leftarrow g^x; \\
 (m_0, m_1) \leftarrow \mathcal{A}(\alpha) ; \\
 y \leftarrow \$-q; \beta \leftarrow g^y; \\
 z \leftarrow \$-q; \delta \leftarrow g^y; \\
 \zeta \leftarrow \delta * m_1; \\
 d \leftarrow \mathcal{A}'(\alpha, \beta, \zeta)
 \end{array}
 \end{array}$$

$$\begin{array}{ccc}
 \approx & & \approx \\
 \begin{array}{l}
 x \leftarrow \$-q; \alpha \leftarrow g^x; \\
 (m_0, m_1) \leftarrow \mathcal{A}(\alpha) ; \\
 y \leftarrow \$-q; \beta \leftarrow g^y; \\
 z \leftarrow \$-q; \zeta \leftarrow g^y; \\
 d \leftarrow \mathcal{A}'(\alpha, \beta, \zeta)
 \end{array}
 &
 &
 \end{array}$$

We have applied the same methodology for these games:

Hashed El Gamal⁰:

$x \leftarrow \$-q; \alpha \leftarrow g^x;$

$k \leftarrow \mathcal{KG}((\eta));$

$(m_0, m_1) \leftarrow \mathcal{A}((\alpha, k));$

$y \leftarrow \$-q; \beta \leftarrow g^y;$

$\delta \leftarrow \alpha^y;$

$h \leftarrow \mathcal{H}((\delta, k));$

$v \leftarrow \oplus hm_0;$

$d \leftarrow \mathcal{A}'((\alpha, k, \beta, v))$

Hashed El Gamal¹:

$x \leftarrow \$-q; \alpha \leftarrow g^x;$

$k \leftarrow \mathcal{KG}((\eta));$

$(m_0, m_1) \leftarrow \mathcal{A}((\alpha, k));$

$y \leftarrow \$-q; \beta \leftarrow g^y;$

$\delta \leftarrow \alpha^y;$

$h \leftarrow \mathcal{H}((\delta, k));$

$v \leftarrow \oplus hm_1;$

$d \leftarrow \mathcal{A}'((\alpha, k, \beta, v))$

(g, q, η)
 $\stackrel{\sim}{d}$

So far, formalized in Coq

- Semantics of the probabilistic programming language
- Theory of observational equivalence
- Reflective tactics for dead-code elimination, code-motion, inlining and common subexpression elimination
- Applied to prove ElGamal IND-CPA security

Prospective applications

- Other applications: computational soundness of an information flow type system.
- Other applications besides cryptography: verification of randomized algorithms in general