Verifiable Security of Boneh-Franklin Identity-Based Encryption

Federico Olmedo
Gilles Barthe Santiago Zanella Béguelin

IMDEA Software Institute, Madrid, Spain

5^{th} International Conference on Provable Security
2011.10.17

Identity-Based Encryption (IBE)

Problem of standard **PKE**:

key management is involved and troublesome

Identity-Based Encryption (IBE)

Problem of standard **PKE**:

key management is involved and troublesome

Proposed solution by Shamir:

to use recipient's ID as public key

Identity-Based Encryption (**IBE**)

Problem of standard **PKE**:

key management is involved and troublesome

Proposed solution by Shamir:

to use recipient's ID as public key

Alice

Bob

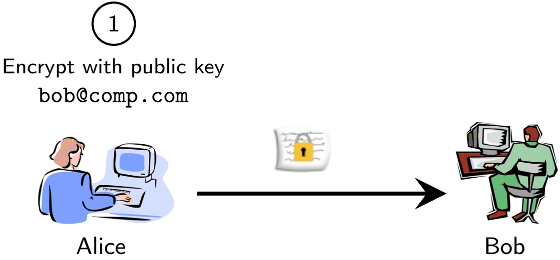
Identity-Based Encryption (**IBE**)

Problem of standard **PKE**:

key management is involved and troublesome

Proposed solution by Shamir:

to use recipient's ID as public key



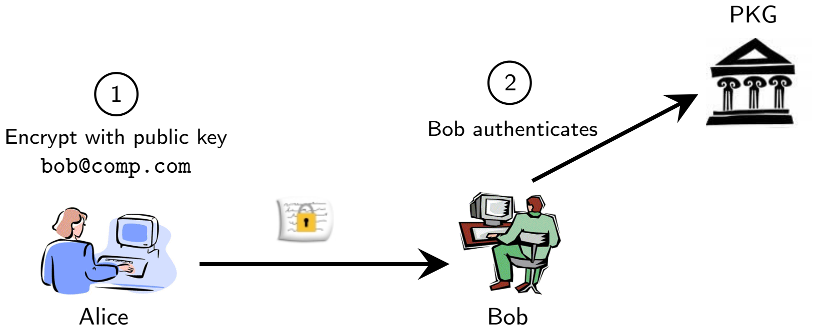
Identity-Based Encryption (**IBE**)

Problem of standard **PKE**:

key management is involved and troublesome

Proposed solution by Shamir:

to use recipient's ID as public key



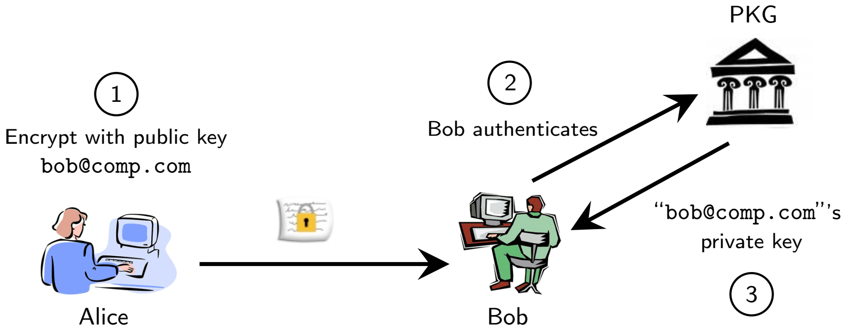
Identity-Based Encryption (IBE)

Problem of standard **PKE**:

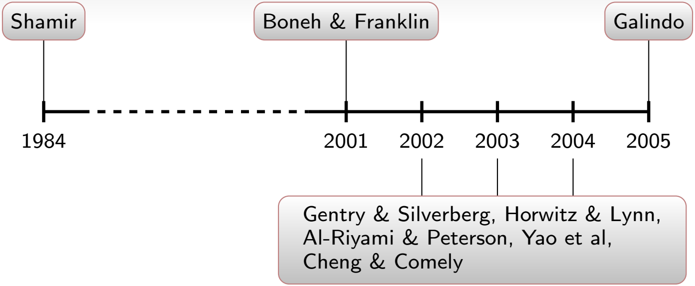
key management is involved and troublesome

Proposed solution by Shamir:

to use recipient's ID as public key



Should we rely on **IBE** schemes?



1984: Conception of identity-based cryptography

2001: First practical provably-secure **IBE** scheme.

2002-2005: Used as building block for many other protocols

2005: Security proof is flawed (but can be patched)

Verifiable security paradigm

Use formal methods to build certified security proofs of cryptographic systems

- Gives strong evidence of correctness of security arguments
- Enables *automation* in proofs
- Demonstrated *applicability* and *effectiveness*

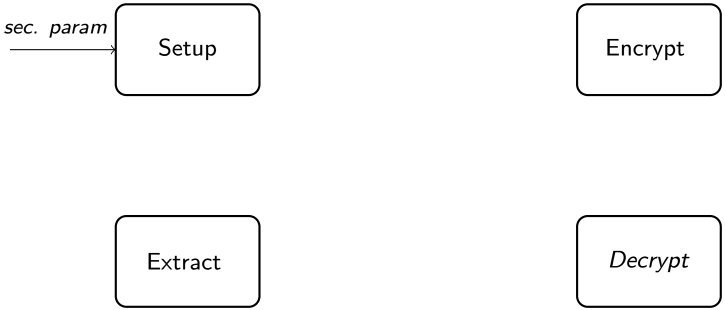
Outline

1. The provably-secure `BasicIdent` scheme
2. CertiCrypt framework
3. Machine-checked proof of `BasicIdent` security
4. Summary and perspectives

An *identity-based encryption scheme* is specified by four polynomial algorithms:

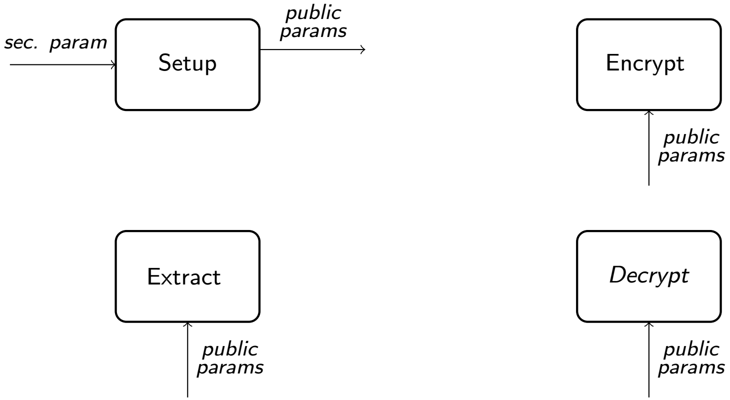
Setup	Encrypt
Extract	*Decrypt*

An *identity-based encryption scheme* is specified by four polynomial algorithms:

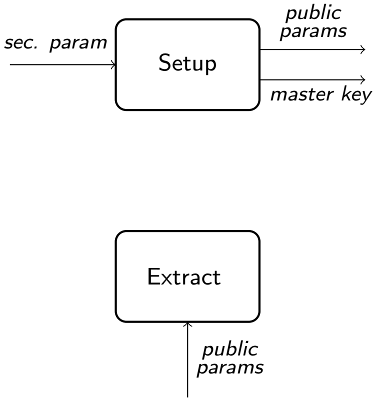


An IBE Scheme

An *identity-based encryption scheme* is specified by four polynomial algorithms:

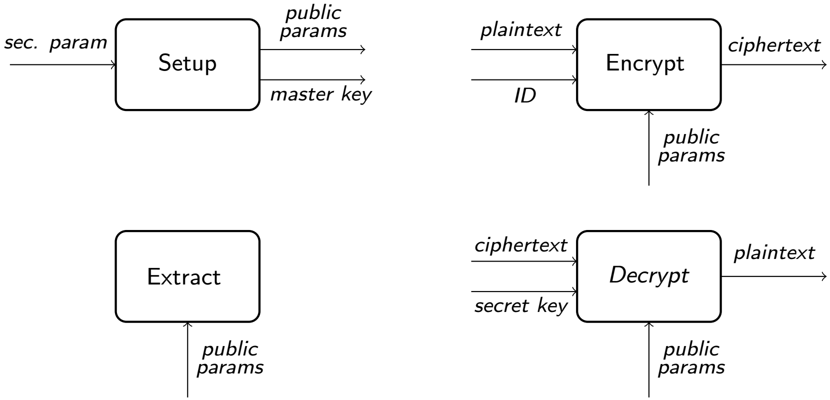


An *identity-based encryption scheme* is specified by four polynomial algorithms:



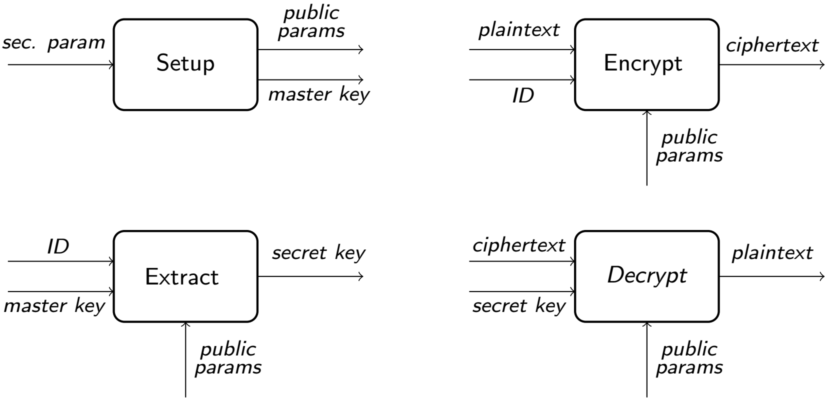
An IBE Scheme

An *identity-based encryption scheme* is specified by four polynomial algorithms:



An IBE Scheme

An *identity-based encryption scheme* is specified by four polynomial algorithms:



Boneh-Franklin's recipe

1. Extend the notions of IND-CPA and IND-CCA to **IBE** schemes
2. Build an IND-CPA-secure **IBE** scheme `BasicIdent`
3. Apply a variant of Fujisaki-Okamoto transformation to turn `BasicIdent` into an IND-CCA-secure **IBE** scheme

The BasicIdent scheme (definition)

Consider

- $\mathbb{G}_1$ and $\mathbb{G}_2$, two cyclic groups of prime order q,
- $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$, an efficiently computable bilinear map

$$\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$$
$$\langle P \rangle = \mathbb{G}_1 \implies \langle \hat{e}(P, P) \rangle = \mathbb{G}_2$$

- Two hash functions

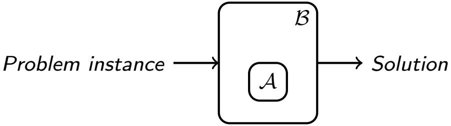
$$\mathcal{H}_1 : \{0,1\}^\star \to \mathbb{G}_1^+$$
$$\mathcal{H}_2 : \mathbb{G}_2 \to \{0,1\}^n$$

The `BasicIdent` **IBE**-scheme is defined as

$$\begin{array}{lcl}
\mathsf{Setup}(k) & : & P \xleftarrow{\$} \mathbb{G}_1^+;\ mk \xleftarrow{\$} \mathbb{Z}_q^+;\ P_{pub} \leftarrow mk \cdot P;\ \text{return } ((P, P_{pub}), mk) \\
\mathsf{Extract}(mk, ID) & : & Q_{ID} \leftarrow \mathcal{H}_1(ID);\ \text{return } mk \cdot Q_{ID} \\
\mathsf{Encrypt}(ID, m) & : & Q_{ID} \leftarrow \mathcal{H}_1(ID);\ c \xleftarrow{\$} \mathbb{Z}_q^+;\ m' \leftarrow \mathcal{H}_2(e(Q_{ID}, P_{pub})^c); \\
 & & \text{return } (c \cdot P, m \oplus m') \\
\mathsf{Decrypt}(sk, (u, v)) & : & \text{return } v \oplus \mathcal{H}_2(\hat{e}(sk, u))
\end{array}$$

The `BasicIdent` scheme (security proof)

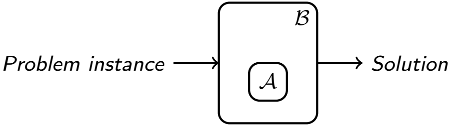
- Proof by reduction (in the random oracle model)
  - Define security goal (and adversarial model)
  - Consider a computational assumption
  - Reduce the security of the scheme to the intractability assumption.



$$\Pr\left[\begin{array}{l}\mathcal{A}\text{ breaks}\\\text{the scheme}\end{array}\right] \leq \mathscr{F}\left(\Pr\left[\begin{array}{l}\mathcal{B}\text{ solves the}\\\text{hard problem}\end{array}\right]\right)$$

- Proof by reduction (in the random oracle model)
  - Define security goal (and adversarial model)

    ➥ **Indistinguishability under Chosen Plaintext Attack**
    Strengthened notion of ***PKE*** *IND-CPA for* ***IBE***
  - Consider a computational assumption
  - Reduce the security of the scheme to the intractability assumption.

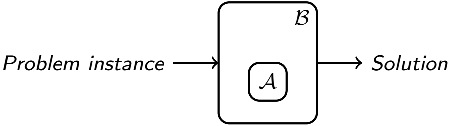


$$\Pr\left[\begin{array}{l}\mathcal{A}\text{ breaks}\\ \text{the scheme}\end{array}\right] \leq \mathscr{F}\left(\Pr\left[\begin{array}{l}\mathcal{B}\text{ solves the}\\ \text{hard problem}\end{array}\right]\right)$$

- Proof by reduction (in the random oracle model)
  - Define security goal (and adversarial model)

    ➥ **Indistinguishability under Chosen Plaintext Attack**
    *Strengthened notion of **PKE** IND-CPA for **IBE***
  - Consider a computational assumption

    ➥ **Bilinear Diffie-Hellman assumption**
    It is hard to compute $\hat{e}(P, P)^{abc}$ given a random tuple $(P, a \cdot P, b \cdot P, c \cdot P)$.
  - Reduce the security of the scheme to the intractability assumption.

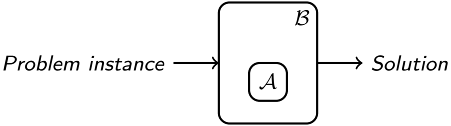


$$\Pr\left[\begin{array}{l}\mathcal{A}\text{ breaks}\\ \text{the scheme}\end{array}\right] \leq \mathscr{F}\left(\Pr\left[\begin{array}{l}\mathcal{B}\text{ solves the}\\ \text{hard problem}\end{array}\right]\right)$$

- Proof by reduction (in the random oracle model)
  - Define security goal (and adversarial model)

    ➥ **Indistinguishability under Chosen Plaintext Attack**
    *Strengthened notion of **PKE** IND-CPA for **IBE***
  - Consider a computational assumption

    ➥ **Bilinear Diffie-Hellman assumption**
    It is hard to compute $\hat{e}(P,P)^{abc}$ given a random tuple $(P, a\cdot P, b\cdot P, c\cdot P)$.
  - Reduce the security of the scheme to the intractability assumption.



$$\Pr\left[\begin{array}{l}\mathcal{A}\text{ breaks}\\ \text{the scheme}\end{array}\right] \leq \mathscr{F}\left(\Pr\left[\begin{array}{l}\mathcal{B}\text{ solves the}\\ \text{hard problem}\end{array}\right]\right)$$

➥ $$\mathbf{Adv}^{\mathcal{A}}_{\text{IND-ID-CPA}} \leq \mathbf{Adv}^{\mathcal{B}}_{\text{BDH}} \; \frac{\exp(1)\; q_{\mathcal{H}_2}\; (1+q_{\mathcal{EX}})}{2}$$

The game-playing technique

Security Goal			*Reduction*
Game G_0	**Game** G_1		**Game** G_n
$\ldots$	$\ldots$		$\ldots$
$\ldots \leftarrow \mathcal{A}(\,)$	$\ldots$	$\ldots$	$\ldots \leftarrow \mathcal{B}(\,)$
$\ldots$	$\ldots$		$\ldots$

$$\Pr_{\mathsf{G_0}}[S_0] \leq f_1\left(\Pr_{\mathsf{G_1}}[S_1]\right) \leq \cdots \leq f_n\left(\Pr_{\mathsf{G_n}}[S_n]\right)$$

CertiCrypt: machine-checked crypto proofs

Certified framework for building and verifying crypto proofs in the Coq proof assistant

- Combination of programming language techniques and cryptographic-specific tools
- Game-based methodology, natural to cryptographers
- Several case studies:
  - Encryption schemes: ElGamal, Hashed ElGamal, OAEP
  - Signature schemes: FDH, BLS
  - Zero-Knowledge protocols: Schnorr, Okamoto, Diffie-Hellman, Fiat-Shamir

Inside CertiCrypt (language syntax)

Language-based proofs

Formalize security definitions, assumptions and games using a probabilistic programming language.

pWhile: a probabilistic programming language

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

- $x \xleftarrow{\$} d$: sample the value of x according to distribution d
- The language of expressions ($\mathcal{E}$) and distribution expressions ($\mathcal{D}$) admits user-defined extensions

Observational equivalence

$$\models c_1 \simeq^I_O c_2$$

Example

$$\models x \xleftarrow{\$} \{0,1\}^k; y \leftarrow x \oplus z \simeq^{\{z\}}_{\{x,y,z\}} y \xleftarrow{\$} \{0,1\}^k; x \leftarrow y \oplus z$$

- Useful to relate probabilities

$$\frac{\mathrm{fv}(A) \subseteq O \qquad \models c_1 \simeq^I_O c_2 \qquad m_1 =_I m_2}{\Pr[c_1, m_1 : A] = \Pr[c_2, m_2 : A]}$$

Inside CertiCrypt (crypto-specific tool)

Fundamental lemma of game-playing

Game G_1

...

bad $\leftarrow$ true; c_1

...

Game G_2

...

bad $\leftarrow$ true; c_2

...

Two identical up to **bad** games

Lemma

If G_1 *and* G_2 *are identical up to* **bad**, *then*

$$|\Pr[G_1, m : A] - \Pr[G_2, m : A]| \leq \max\{\Pr[G_1, m : \mathbf{bad}], \Pr[G_2, m : \mathbf{bad}]\}$$

Our proof in CertiCrypt

We extended CertiCrypt with:

- Types and operators for the groups $\mathbb{G}_1, \mathbb{G}_2$
- An operator for a bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$
- Simplification rules for computing normal forms of applications of the bilinear map $\hat{e}$
- An instruction for sampling from Bernoulli distributions

Formalizing the security goal:

$$
\begin{array}{l}
\textbf{Game } \mathsf{G}_{\mathsf{IND\text{-}ID\text{-}CPA}}: \\
\quad (params, mk) \leftarrow \mathsf{Setup}(k); \\
\quad (m_0, m_1, ID_{\mathcal{A}}) \leftarrow \mathcal{A}_1(params); \\
\quad b \xleftarrow{\$} \{0,1\}; \\
\quad c \leftarrow \mathsf{Encrypt}(ID_{\mathcal{A}}, m_b); \\
\quad b_{\mathcal{A}} \leftarrow \mathcal{A}_2(c)
\end{array}
$$

- The adversary is modeled by two procedures (of unknown code) $\mathcal{A}_1$ and $\mathcal{A}_2$ that communicate through shared variables
- $\mathcal{A}_1$ and $\mathcal{A}_2$ have oracle access to the extraction algorithm and to both random oracles
- Neither $\mathcal{A}_1$ nor $\mathcal{A}_2$ is allowed to query the challenge $ID_{\mathcal{A}}$ to the extraction oracle.

$$\mathbf{Adv}^{\mathcal{A}}_{\mathsf{IND\text{-}ID\text{-}CPA}} \stackrel{\text{def}}{=} \left| \Pr_{\mathsf{G}_{\mathsf{IND\text{-}ID\text{-}CPA}}}[b = b_{\mathcal{A}}] - \frac{1}{2} \right|$$

Formalizing the assumptions

- The Bilinear Diffie-Hellman assumption

Game $\mathsf{G}^{\mathcal{B}}_{\mathsf{BDH}}$:

$P \xleftarrow{\$} \mathbb{G}_1^+;\ a, b, c \xleftarrow{\$} \mathbb{Z}_q^+;$
$z \leftarrow \mathcal{B}(P, a\cdot P, b\cdot P, c\cdot P)$

$$\mathbf{Adv}^{\mathcal{B}}_{\mathsf{BDH}} \stackrel{\text{def}}{=} \Pr_{\mathsf{G}^{\mathcal{B}}_{\mathsf{BDH}}}\left[z = \hat{e}(P, P)^{abc}\right]$$

$$\forall \mathcal{B} \bullet \mathsf{PPT}(\mathcal{B}) \implies \mathsf{negl}(\mathbf{Adv}^{\mathcal{B}}_{\mathsf{BDH}})$$

- The random oracle model

Oracle $\mathcal{H}_1(ID)$:

if $ID \notin \mathrm{dom}(L_1)$ then
  $R \xleftarrow{\$} \mathbb{G}_1^+;$
  $L_1(ID) \leftarrow R$
return $L_1(ID)$

Oracle $\mathcal{H}_2(r)$:

if $r \notin \mathrm{dom}(L_2)$ then
  $m \xleftarrow{\$} \{0,1\}^n;$
  $L_2(r) \leftarrow m$
return $L_2(r)$

Building the reduction...

Game $G_{\text{IND-ID-CPA}}$:

$$
\begin{array}{l}
(parm, mk) \leftarrow \text{Setup}(k); \\
(m_0, m_1, ID_{\mathcal{A}}) \leftarrow \mathcal{A}_1(parm); \\
b \stackrel{\$}{\leftarrow} \{0,1\}; \\
c \leftarrow \text{Encrypt}(ID_{\mathcal{A}}, m_b); \\
b_{\mathcal{A}} \leftarrow \mathcal{A}_2(c)
\end{array}
$$

...

Game $G^{\mathcal{B}}_{\text{BDH}}$:

$$
\begin{array}{l}
P \stackrel{\$}{\leftarrow} \mathbb{G}_1^+;\ a, b, c \stackrel{\$}{\leftarrow} \mathbb{Z}_q^+; \\
z \leftarrow \mathcal{B}(P, a \cdot P, b \cdot P, c \cdot P)
\end{array}
$$

$$
\mathbf{Adv}^{\mathcal{A}}_{\text{IND-ID-CPA}} \leq \cdots \leq \mathbf{Adv}^{\mathcal{B}}_{\text{BDH}} \frac{\exp(1)\ q_{\mathcal{H}_2}\ (1+q_{\mathcal{EX}})}{2}
$$

- Seven intermediate games
- Lazy sampling, fundamental lemma, Coron's technique
- Same bound as Boneh & Franklin proof

Our proof in CertiCrypt

- Our reduction is direct in contrast to Boneh-Franklin proof that goes through an intermediate IND-CPA-secure (non-IBE) encryption scheme
- Used a simpler argument instead of an inductive argument in Boneh-Franklin's proof that we could not reproduce
- 5000 lines of Coq script
- Built in 3 man-months (but automatically verifiable in 10 minutes)

Contributions

- Presented a machine-checked reduction of the security of the `BasicIdent` **IBE** scheme to the Bilinear Diffie-Hellman assumption
- Demonstrated that CertiCrypt can be extended to deal with complex security proofs of cryptographic schemes

Perspectives

- Formalize Fujisaki-Okamoto meta-result.
- Eliminate RO assumption on $\mathbb{G}_1$: formalize Brier *et al* work about indifferentiability of hash functions into elliptic curves.

Questions?

Get CertiCrypt (and EasyCrypt) from:
`http://certicrypt.gforge.inria.fr`

Inside CertiCrypt (language semantics)

Programs map an initial memory to a distribution of final memories:

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

We use Paulin's measure monad to represent distributions:

$$\mathcal{D}(A) \quad \stackrel{\text{def}}{=} \quad (A \to [0,1]) \to [0,1]$$

For instance

$$[\![x \stackrel{\$}{\leftarrow} \{\mathsf{true}, \mathsf{false}\}]\!] \ m = \lambda f \cdot \left(\frac{1}{2} f(m[x/\mathsf{true}]) + \frac{1}{2} f(m[x/\mathsf{false}]) \right)$$

To compute probabilities, just measure the characteristic function of the event:

$$\Pr[c, m : A] \stackrel{\text{def}}{=} [\![c]\!] \ m \ \mathbb{1}_A$$

What does it take to trust a proof in CertiCrypt

- You need to
  - trust the type checker of Coq
  - trust the definition of the language semantics
  - make sure the security statement and the computational assumption (a few lines in Coq) are what you expect it to be
- You don't need to
  - understand or even read the proof
  - trust proof tactics, program transformations
  - trust program logics, wp-calculus
  - be an expert in Coq

Our proof in CertiCrypt I

Game CPA :
$\boldsymbol{L_1}, \boldsymbol{L_2}, \boldsymbol{L_3} \leftarrow \text{nil};$
$\boldsymbol{P} \xleftarrow{\$} \mathbb{G}_1^+;\ \boldsymbol{a} \xleftarrow{\$} \mathbb{Z}_q^+;$
$\boldsymbol{P_{pub}} \leftarrow \boldsymbol{aP};$
$(m_0, m_1, ID_\mathcal{A}) \leftarrow \mathcal{A}_1(\boldsymbol{P}, \boldsymbol{P_{pub}});$
$d \xleftarrow{\$} \{0,1\};$
$y \leftarrow \mathcal{E}(ID_\mathcal{A}, m_d);$
$d_\mathcal{A} \leftarrow \mathcal{A}_2(y)$

Oracle $\mathcal{EX}(ID)$:
if $ID \notin \boldsymbol{L_3}$ then
  $\boldsymbol{L_3} \leftarrow ID :: \boldsymbol{L_3}$
$Q \leftarrow \mathcal{H}_1(ID);$
return $\boldsymbol{a}Q$

Oracle $\mathcal{H}_1(ID)$:
if $ID \notin \text{dom}(\boldsymbol{L_1})$ then
  $R \xleftarrow{\$} \mathbb{G}_1^+;$
  $\boldsymbol{L_1}(id) \leftarrow R$
return $\boldsymbol{L_1}(ID)$

Oracle $\mathcal{H}_2(r)$:
if $r \notin \text{dom}(\boldsymbol{L_2})$ then
  $m \xleftarrow{\$} \{0,1\}^n;$
  $\boldsymbol{L_2}(r) \leftarrow m$
return $\boldsymbol{L_2}(r)$

Game BDH :
$P \xleftarrow{\$} \mathbb{G}_1^+;\ a, b, c \xleftarrow{\$} \mathbb{Z}_q^+;$
$z \leftarrow \mathcal{B}(P, aP, bP, cP)$

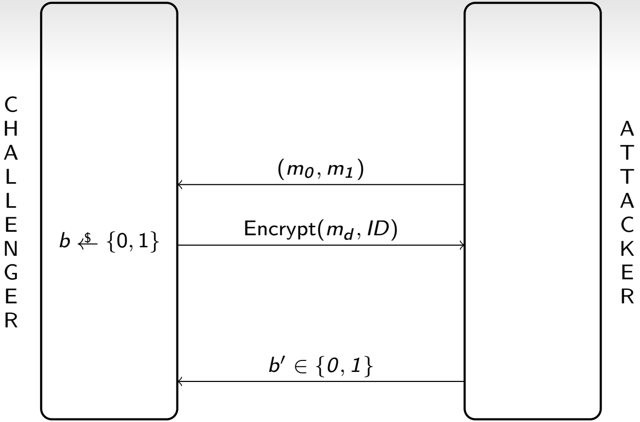
$\mathcal{B}(P_0, P_1, P_2, P_3)$:
$\boldsymbol{L_1}, \boldsymbol{L_2}, \boldsymbol{L_3}, \boldsymbol{V}, \boldsymbol{T} \leftarrow \text{nil};$
while $|\boldsymbol{T}| < q_{\mathcal{H}_1}$ do
  $t \xleftarrow{\$} \text{true} \oplus_p \text{false};\ \boldsymbol{T} \leftarrow t :: \boldsymbol{T}$
$\boldsymbol{P} \leftarrow P_0;\ \boldsymbol{P_{pub}} \leftarrow P_1;\ \boldsymbol{P'} \leftarrow P_2;$
$(m_0, m_1, ID_\mathcal{A}) \leftarrow \mathcal{A}_1(\boldsymbol{P}, \boldsymbol{P_{pub}});$
$Q_\mathcal{A} \leftarrow \mathcal{H}_1(ID_\mathcal{A});\ v' \leftarrow \boldsymbol{V}(ID_\mathcal{A})^{-1};$
$R \xleftarrow{\$} \{0,1\}^n;\ y \leftarrow (v'P_3, R);$
$d_\mathcal{A} \leftarrow \mathcal{A}_2(y);$
$i \xleftarrow{\$} [1..|\boldsymbol{L_2}|];$ return $\text{fst}(\boldsymbol{L_2}[i])$

Oracle $\mathcal{EX}(ID)$:
if $ID \notin \boldsymbol{L_3}$ then
  $\boldsymbol{L_3} \leftarrow ID :: \boldsymbol{L_3}$
$Q \leftarrow \mathcal{H}_1(ID);$
return $\boldsymbol{a}Q$

Oracle $\mathcal{H}_1(ID)$:
if $ID \notin \text{dom}(\boldsymbol{L_1})$ then
  $v \xleftarrow{\$} \mathbb{Z}_q^+;$
  $\boldsymbol{V}(ID) \leftarrow v;$
  if $\boldsymbol{T}[|\boldsymbol{L_1}|]$ then
    $\boldsymbol{L_1}(ID) \leftarrow v\boldsymbol{P'}$
  else
    $\boldsymbol{L_1}(ID) \leftarrow v\boldsymbol{P}$
return $\boldsymbol{L_1}(ID)$

Oracle $\mathcal{H}_2(r)$:
if $r \notin \text{dom}(\boldsymbol{L_2})$ then
  $m \xleftarrow{\$} \{0,1\}^n;$
  $\boldsymbol{L_2}(r) \leftarrow m$
return $\boldsymbol{L_2}(r)$

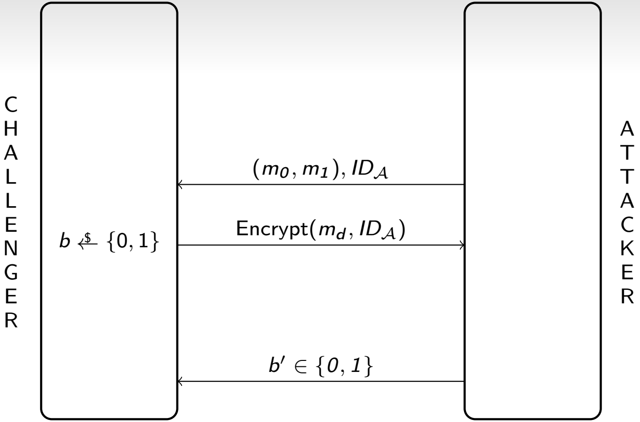
Semantic security of an **IBE** scheme



An **IBE** scheme is *IND-ID-CPA-secure* iff

$$\forall \mathcal{A} \bullet \mathsf{PPT}(\mathcal{A}) \implies \left| \Pr\left[b = b'\right] - \frac{1}{2} \right| \text{ is negligible}$$

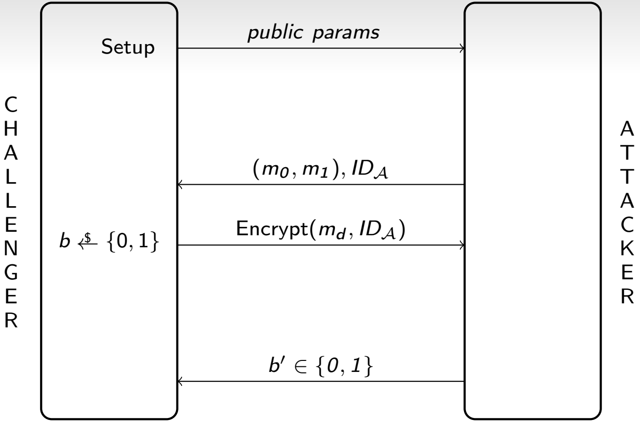
Semantic security of an **IBE** scheme



An **IBE** scheme is *IND-ID-CPA-secure* iff

$$\forall \mathcal{A} \bullet \mathrm{PPT}(\mathcal{A}) \implies \left| \Pr\left[b = b'\right] - \frac{1}{2} \right| \text{ is negligible}$$

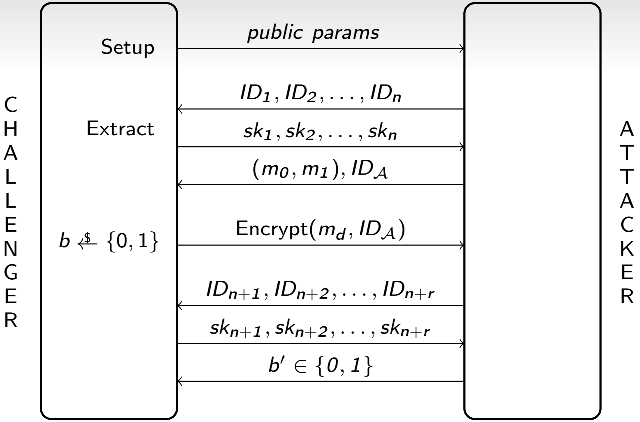
Semantic security of an **IBE** scheme



An **IBE** scheme is *IND-ID-CPA-secure* iff

$$\forall \mathcal{A} \bullet \mathrm{PPT}(\mathcal{A}) \implies \left| \Pr\left[b = b'\right] - \frac{1}{2} \right| \text{ is negligible}$$

Semantic security of an **IBE** scheme



An **IBE** scheme is *IND-ID-CPA-secure* iff

$$\forall \mathcal{A} \bullet \mathrm{PPT}(\mathcal{A}) \wedge \Pr\left[\bigwedge_{i=1}^{m} id_i \neq id_{\mathcal{A}}\right] = 1 \implies \left|\Pr\left[b = b'\right] - \frac{1}{2}\right| \text{ is negligible}$$