Verifiable Security of Boneh-Franklin Identity-Based Encryption

Federico Olmedo
Gilles Barthe Santiago Zanella Béguelin

IMDEA Software Institute, Madrid, Spain

5th 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*
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*

1. Encrypt with public key
   
   \[ \text{bob@comp.com} \]
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

1. Encrypt with public key
   bob@comp.com

2. Bob authenticates

Alice

Bob

PKG
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*

1. Encrypt with public key
   bob@comp.com

2. Bob authenticates
   "bob@comp.com"’s private key

3. "bob@comp.com"’s private key

Verifiable Security of Boneh-Franklin, Identity-Based Encryption
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)
Improving the security argument

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 IBE Scheme

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:

- **Setup**
- **Encrypt**
- **Extract**
- **Decrypt**
An identity-based encryption scheme is specified by four polynomial algorithms:

- **Setup**
  - Input: security parameter
  - Output: master key, public parameters

- **Encrypt**
  - Input: public parameters
  - Output: ciphertext, public parameters

- **Extract**
  - Input: public parameters
  - Output: secret key, plaintext

- **Decrypt**
  - Input: public parameters
  - Output: master key, ciphertext, secret key
An identity-based encryption scheme is specified by four polynomial algorithms:

- **Setup**: Takes a secret parameter and outputs public parameters.
- **Encrypt**: Takes public parameters and a master key to generate ciphertext.
- **Extract**: Takes public parameters and outputs a secret key.
- **Decrypt**: Takes public parameters and a ciphertext to produce plaintext.
An identity-based encryption scheme is specified by four polynomial algorithms:
An identity-based encryption scheme is specified by four polynomial algorithms:

- **Setup**: Takes the secret parameters as input and outputs the public parameters and master key.
- **Encrypt**: Takes the plaintext and identity as input and outputs the ciphertext.
- **Extract**: Takes the identity and master key as input and outputs the secret key.
- **Decrypt**: Takes the ciphertext, secret key, and public parameters as input and outputs the plaintext.
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

- $G_1$ and $G_2$, two cyclic groups of prime order $q$,
- $\hat{e} : G_1 \times G_1 \to G_2$, an efficiently computable bilinear map
  
  \[
  \hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab} \\
  \langle P \rangle = G_1 \implies \langle \hat{e}(P, P) \rangle = G_2
  \]

- Two hash functions
  
  $H_1 : \{0, 1\}^* \to G_1^+$
  
  $H_2 : G_2 \to \{0, 1\}^n$

The BasicIdent IBE-scheme is defined as

- **Setup**($k$) : $P \leftarrow G_1^+$; $mk \leftarrow \mathbb{Z}_q^+$; $P_{pub} \leftarrow mk \cdot P$; return $((P, P_{pub}), mk)$
- **Extract**($mk, ID$) : $Q_{ID} \leftarrow H_1(ID)$; return $mk \cdot Q_{ID}$
- **Encrypt**($ID, m$) : $Q_{ID} \leftarrow H_1(ID)$; $c \leftarrow \mathbb{Z}_q^+$; $m' \leftarrow H_2(e(Q_{ID}, P_{pub})^c)$; return $(c \cdot P, m \oplus m')$
- **Decrypt**($sk, (u, v)$) : return $v \oplus H_2(\hat{e}(sk, u))$
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.

\[
\Pr \left[ A \text{ breaks the scheme} \right] \leq \mathcal{R} \left( \Pr \left[ B \text{ solves the hard problem} \right] \right)
\]
The BasicIdent scheme (security proof)

- 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.

\[
\Pr \left[ A \text{ breaks the scheme} \right] \leq \mathcal{F} \left( \Pr \left[ B \text{ solves the hard problem} \right] \right)
\]
The BasicIdent scheme (security proof)

- 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.

\[
\Pr \left[ \mathcal{A} \text{ breaks the scheme} \right] \leq \mathcal{F} \left( \Pr \left[ \mathcal{B} \text{ solves the hard problem} \right] \right)
\]
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.

![Diagram](image)

\[
\Pr \left[ A \text{ breaks the scheme} \right] \leq \mathcal{F} \left( \Pr \left[ B \text{ solves the hard problem} \right] \right)
\]

\[
\text{Adv}_{\text{IND-ID-CPA}}^A \leq \text{Adv}_{\text{BDH}}^B \frac{\exp(1) q_{H_2} (1+q_{E x})}{2}
\]
Tidying the proof up

The game-playing technique

Security Goal

\[
\text{Game } G_0 \\
\vdots \\
\vdots \leftarrow A(\text{)} \\
\vdots \\
\Pr_{G_0}[S_0] \leq f_1(Pr_{G_1}[S_1]) \leq \cdots \leq f_n(Pr_{G_n}[S_n])
\]

Reduction

\[
\text{Game } G_1 \\
\vdots \\
\vdots \\
\vdots \leftarrow B(\text{)} \\
\vdots
\]
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

\[ C ::= \begin{align*} & \text{skip} & \text{nop} \\ & C; C & \text{sequence} \\ & V \leftarrow E & \text{assignment} \\ & V \leftarrow D & \text{random sampling} \\ & \text{if } E \text{ then } C \text{ else } C & \text{conditional} \\ & \text{while } E \text{ do } C & \text{while loop} \\ & V \leftarrow \mathcal{P}(E, \ldots, E) & \text{procedure call} \end{align*} \]

- \( x \leftarrow d \): sample the value of \( x \) according to distribution \( d \)
- The language of expressions (\( E \)) and distribution expressions (\( D \)) admits user-defined extensions
Inside CertiCrypt (standard tools)

Observational equivalence

\[ \models c_1 \sim^I_O c_2 \]

Example

\[ \models x \leftarrow \{0, 1\}^k; y \leftarrow x \oplus z \sim^{\{z\}}_{\{x,y,z\}} y \leftarrow \{0, 1\}^k; x \leftarrow y \oplus z \]

- Useful to relate probabilities

\[
\begin{align*}
\text{fv}(A) \subseteq O & \models c_1 \sim^I_O c_2 \quad m_1 \equiv m_2 \\
\Pr [c_1, m_1 : A] &= \Pr [c_2, m_2 : A]
\end{align*}
\]
Inside CertiCrypt (crypto-specific tool)

Fundamental lemma of game-playing

\[
\begin{align*}
\text{Game } G_1 \\
&\ldots \\
&\text{bad } \leftarrow \text{true}; c_1 \\
&\ldots \\
\text{Game } G_2 \\
&\ldots \\
&\text{bad } \leftarrow \text{true}; c_2 \\
&\ldots 
\end{align*}
\]

Two identical up to \text{bad} games

Lemma

If \( G_1 \) and \( G_2 \) are identical up to \text{bad}, then

\[
|\Pr [ G_1, m : A ] - \Pr [ G_2, m : A ]| \leq \max \{ \Pr [ G_1, m : \text{bad} ], \Pr [ G_2, m : \text{bad} ] \}
\]
We extended CertiCrypt with:

- Types and operators for the groups $G_1, G_2$
- An operator for a bilinear map $\hat{e} : G_1 \times G_1 \rightarrow G_2$
- Simplification rules for computing normal forms of applications of the bilinear map $\hat{e}$
- An instruction for sampling from Bernoulli distributions
Our proof in CertiCrypt

Formalizing the security goal:

\[
\text{Game } G_{\text{IND-ID-CPA}} : \\
(\text{params}, \text{mk}) \leftarrow \text{Setup}(k); \\
(m_0, m_1, \text{ID}_A) \leftarrow \mathcal{A}_1(\text{params}); \\
b \leftarrow \{0, 1\}; \\
c \leftarrow \text{Encrypt}(\text{ID}_A, m_b); \\
b_A \leftarrow \mathcal{A}_2(c)
\]

- 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 $\text{ID}_A$ to the extraction oracle.

\[
\text{Adv}^A_{\text{IND-ID-CPA}} \overset{\text{def}}{=} \left| \Pr_{G_{\text{IND-ID-CPA}}} [b = b_A] - \frac{1}{2} \right|
\]
Our proof in CertiCrypt

Formalizing the assumptions

- The Bilinear Diffie-Hellman assumption

**Game** $G_{BDH}^B$

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

**Adv** $B_{BDH}$ \( \overset{\text{def}}{=} \Pr_{G_{BDH}^B} [z = \hat{e}(P, P)^{abc}] \)

\( \forall B \cdot \text{PPT}(B) \implies \text{negl}(\text{Adv}_{BDH}^B) \)

- The random oracle model

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

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

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

- if $r \notin \text{dom}(L_2)$ then
- $m \leftarrow \{0, 1\}^n$;
- $L_2(r) \leftarrow m$
- return $L_2(r)$
Building the reduction...

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

$(\text{parm}, mk) \leftarrow \text{Setup}(k);$  
$(m_0, m_1, ID_A) \leftarrow A_1(\text{parm});$  
$b \leftarrow \{0, 1\};$  
$c \leftarrow \text{Encrypt}(ID_A, m_b);$  
$b_A \leftarrow A_2(c)$

Game $G_{\text{BDH}}^B$:

$P \leftarrow G_1^+$;  
$a, b, c \leftarrow \mathbb{Z}_q^+$;  
$z \leftarrow B(P, a \cdot P, b \cdot P, c \cdot P)$

$\text{Adv}_{\text{IND-ID-CPA}}^A \leq \cdots \leq \text{Adv}_{\text{BDH}}^B \frac{\exp(1) q \cdot \mathcal{H}_2}{2} \left(1 + q \epsilon x\right)$

- 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 $G_1$: formalize Brier et al. work about indifferentiability of hash functions into elliptic curves.
Final remarks

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:

\[
\lbrack c \in C \rbrack : \mathcal{M} \rightarrow \mathcal{D}(\mathcal{M})
\]

We use Paulin’s measure monad to represent distributions:

\[
\mathcal{D}(A) \overset{\text{def}}{=} (A \rightarrow [0, 1]) \rightarrow [0, 1]
\]

For instance

\[
\lbrack x \gets \{\text{true, false}\} \rbrack m = \lambda f \cdot \left(\frac{1}{2} f(m[x/\text{true}]) + \frac{1}{2} f(m[x/\text{false}])\right)
\]

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

\[
\Pr [c, m : A] \overset{\text{def}}{=} \lbrack c \rbrack m 1_A
\]
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**

<table>
<thead>
<tr>
<th>Game CPA</th>
<th>Oracle $\mathcal{E}(ID)$</th>
<th>Oracle $\mathcal{H}_1(ID)$</th>
<th>Oracle $\mathcal{H}_2(r)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$L_1, L_2, L_3 \leftarrow \text{nil}$; $P \leftarrow G_1^+$; $a \leftarrow \mathbb{Z}<em>q^+$; $P</em>{\text{pub}} \leftarrow aP$; $(m_0, m_1, \text{ID}<em>A) \leftarrow A_1(P, P</em>{\text{pub}})$; $d \leftarrow {0, 1}$; $y \leftarrow \mathcal{E}(\text{ID}_A, m_d)$; $d_A \leftarrow A_2(y)$</td>
<td>if $ID \notin L_3$ then $L_3 \leftarrow ID :: L_3$; $Q \leftarrow \mathcal{H}_1(ID)$; return $aQ$</td>
<td>if $ID \notin \text{dom}(L_1)$ then $R \leftarrow G_1^+$; $L_1(id) \leftarrow R$; return $L_1(ID)$</td>
<td>if $r \notin \text{dom}(L_2)$ then $m \leftarrow {0, 1}^n$; $L_2(r) \leftarrow m$; return $L_2(r)$</td>
</tr>
<tr>
<td>$P \leftarrow G_1^+$; $a, b, c \leftarrow \mathbb{Z}_q^+$; $z \leftarrow B(P, aP, bP, cP)$; $B(P_0, P_1, P_2, P_3)$: $L_1, L_2, L_3, V, T \leftarrow \text{nil}$; while $</td>
<td>T</td>
<td>&lt; q_{\mathcal{H}<em>1}$ do $t \leftarrow \text{true } \oplus_p \text{false}$; $T \leftarrow t :: T$; $P \leftarrow P_0$; $P</em>{\text{pub}} \leftarrow P_1$; $P' \leftarrow P_2$; $(m_0, m_1, \text{ID}<em>A) \leftarrow A_1(P, P</em>{\text{pub}})$; $Q_A \leftarrow \mathcal{H}_1(ID_A)$; $v' \leftarrow V(ID_A)^{-1}$; $R \leftarrow {0, 1}^n$; $y \leftarrow (v'P_3, R)$; $d_A \leftarrow A_2(y)$; $i \leftarrow [1..</td>
<td>L_2</td>
</tr>
</tbody>
</table>

**Game BDH**:

- $P \leftarrow G_1^+$; $a, b, c \leftarrow \mathbb{Z}_q^+$; $z \leftarrow B(P, aP, bP, cP)$; $B(P_0, P_1, P_2, P_3)$:
  - $L_1, L_2, L_3, V, T \leftarrow \text{nil}$;
  - while $|T| < q_{\mathcal{H}_1}$ do $t \leftarrow \text{true } \oplus_p \text{false}$; $T \leftarrow t :: T$; $P \leftarrow P_0$; $P_{\text{pub}} \leftarrow P_1$; $P' \leftarrow P_2$; $(m_0, m_1, \text{ID}_A) \leftarrow A_1(P, P_{\text{pub}})$; $Q_A \leftarrow \mathcal{H}_1(ID_A)$; $v' \leftarrow V(ID_A)^{-1}$; $R \leftarrow \{0, 1\}^n$; $y \leftarrow (v'P_3, R)$; $d_A \leftarrow A_2(y)$; $i \leftarrow [1..|L_2|];$ return $\text{fst}(L_2[i])$

**Oracle $\mathcal{E}(ID)$**:

- if $ID \notin L_3$ then $L_3 \leftarrow ID :: L_3$;
- $Q \leftarrow \mathcal{H}_1(ID)$;
- return $aQ$

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

- if $ID \notin \text{dom}(L_1)$ then $R \leftarrow G_1^+$;
- $L_1(id) \leftarrow R$;
- return $L_1(ID)$

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

- if $r \notin \text{dom}(L_2)$ then $m \leftarrow \{0, 1\}^n$;
- $L_2(r) \leftarrow m$;
- return $L_2(r)$
An **IBE** scheme is *IND-ID-CPA-secure* iff

$$\forall \mathcal{A} \cdot \text{PPT}(\mathcal{A}) \implies \left| \Pr[b = b'] - \frac{1}{2} \right| \text{ is negligible}$$
Semantic security of an **IBE** scheme

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

\[ \forall A \cdot \text{PPT}(A) \implies \left| \Pr[b = b'] - \frac{1}{2} \right| \text{ is negligible} \]
Semantic security of an IBE scheme

An IBE scheme is \textit{IND-ID-CPA-secure} iff

\[
\forall A \cdot \text{PPT}(A) \implies \Pr[b = b'] - \frac{1}{2} \text{ is negligible}
\]
An IBE scheme is **IND-ID-CPA-secure** iff

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