# Verifiable Security of Boneh-Franklin Identity-Based Encryption*

Gilles Barthe, Federico Olmedo, and Santiago Zanella Béguelin

{Gilles.Barthe,Federico.Olmedo,Santiago.Zanella}@imdea.org
IMDEA Software Institute, Madrid, Spain

**Abstract.** Identity-based encryption (IBE) allows one party to send ciphered messages to another using an arbitrary identity string as an encryption key. Since IBE does not require prior generation and distribution of keys, it greatly simplifies key management in public-key cryptography. Although the concept of IBE was introduced by Shamir in 1981, constructing a practical IBE scheme remained an open problem for years. The first satisfactory solution was proposed by Boneh and Franklin in 2001 and constitutes one of the most prominent applications of pairing-based cryptography. We present a game-based machine-checked reduction of the security of the Boneh-Franklin IBE scheme to the Bilinear Diffie-Hellman assumption, and analyze its tightness by providing an exact security bound. Our proof simplifies and clarifies the original proof by Boneh and Franklin and can be automatically verified by running a trusted checker.

**Keywords**: Bilinear Diffie-Hellman problem, Boneh-Franklin scheme, CertiCrypt, iddentity-based encryption, pairing-based cryptography, verifiable security.

## 1 Introduction

Identity-based cryptography is an approach to public-key cryptography in which public keys can be arbitrary identity strings associated to users, e.g. their email addresses. Identity-based cryptography significantly reduces the cost and complexity of managing a public-key infrastructure because, in contrast to standard public-key systems, it does not require prior distribution and generation of keys. Although the concept of identity-based cryptography was introduced by Shamir in 1984 [14] and identity-based signature schemes are relatively easy to construct, a solution to the problem of building a practical identity-based encryption scheme eluded cryptographers for years. The first satisfactory solution to this problem was proposed by Boneh and Franklin in 2001 [7] using the Weil pairing,

* Partially funded by European Project FP7-256980 NESSoS, French project ANR SESUR-012 SCALP, Spanish project TIN2009-14599 DESAFIOS 10, and Madrid Regional project S2009TIC-1465 PROMETIDOS.

and constitutes one of the most prominent applications of pairing-based cryptography. Boneh and Franklin proved that this scheme is secure against chosen-ciphertext attacks in the Random Oracle Model (ROM) under the Bilinear Diffie-Hellman assumption. The proof proceeds in two stages: first, an identity-based scheme BasicIdent is introduced and proved secure against chosen-plaintext attacks; second, the BasicIdent scheme is transformed into a scheme that is secure against chosen-ciphertext attacks by applying a variant of the Fujisaki-Okamoto transformation [11]. A flaw in the second part of this proof was discovered and fixed by Galindo [12]. Although, fortunately, in this case the fix did not require to modify the scheme or the underlying assumption, this shows that some degree of wariness is needed when evaluating provable security arguments.

Boneh and Boyen [6] and Waters [16] subsequently proposed other provably-secure IBE schemes that admit reductions in the standard model; Bellare and Ristenpart [5] improve on the security bound of Waters' scheme by removing artificial abort steps from the proof. Over the last decade, more sophisticated schemes have emerged, such as hierarchical [13] and anonymous [8] IBE schemes. As the security proofs for such schemes are getting more and more involved, it becomes increasingly difficult to assess the correctness of the mathematical arguments, or the tightness of the concrete security bounds.

Verifiable security [1, 2] is an emerging approach to provable security that advocates using state-of-the-art tools to build fully formalized, independently verifiable proofs of security of cryptographic systems. This approach has been applied to prominent cryptographic constructions, including proofs of chosen-ciphertext security of OAEP encryption [2] and unforgeability of FDH signatures [17]. In this paper we follow this approach and use CertiCrypt [1] to build a machine-checked game-based proof of the security of Boneh-Franklin BasicIdent scheme. Our main contributions are the following: 1) We extend the CertiCrypt framework with primitive operations for bilinear maps and mechanisms to reason about their algebraic properties; 2) We formalize a game-based proof of the security of Boneh-Franklin BasicIdent scheme that is simpler than the original one; 3) We analyze the tightness of the reduction and obtain an exact security bound that coincides with the one in the original proof. To the best of our knowledge, the proof presented here constitutes the first machine-checked proof of a pairing-based cryptographic scheme, and paves the way to formally analyze the provable security of other pairing-based constructions.

## 2 An Introduction to CertiCrypt

CertiCrypt [1] is a framework for building and verifying game-based proofs of cryptographic systems that adopts a code-based view of games. CertiCrypt is built on top of the general-purpose proof assistant Coq [15], that has been used effectively for verifying intricate results from mathematics and computer science.

The core of CertiCrypt is a formalization of the probabilistic programming language used to represent games; the syntax of games is defined as follows:

| | | | |
|---|---|---|---|
| $\mathcal{I} ::= \mathcal{V} \leftarrow \mathcal{E}$ | deterministic assignment | $\mathcal{C} ::= \mathsf{skip}$ | nop |
| $\mid\ \mathcal{V} \stackrel{\$}{\leftarrow} \mathcal{DE}$ | random assignment | $\mid\ \mathcal{I};\ \mathcal{C}$ | sequence |
| $\mid\ \mathsf{if}\ \mathcal{E}\ \mathsf{then}\ \mathcal{C}\ \mathsf{else}\ \mathcal{C}$ | conditional | | |
| $\mid\ \mathsf{while}\ \mathcal{E}\ \mathsf{do}\ \mathcal{C}$ | while loop | | |
| $\mid\ \mathcal{V} \leftarrow \mathcal{P}(\mathcal{E}, \ldots, \mathcal{E})$ | procedure call | | |

where $\mathcal{V}$ is a set of variables, $\mathcal{E}$ a set of expressions, $\mathcal{DE}$ is a set of expressions that represent distributions from which values can be sampled in random assignments, and $\mathcal{P}$ is a set of procedures that includes oracles and adversaries. Adversaries are formalized as procedures with unknown code; the only requirement is that adversaries execute in probabilistic polynomial-time and comply with an interface that specifies a read/write access policy to global variables. The semantics of a game $G$ is given by a function $\llbracket G \rrbracket : \mathcal{M} \to \mathcal{D}(\mathcal{M})$ which yields for any initial memory $m$, mapping program variables to values, the (sub-)distribution of final memories resulting from executing $G$ starting from $m$. We denote by $\Pr[G, m : A]$ the probability of event $A$ occurring after executing game $G$ in an initial memory $m$.

In order to formalize security proofs, CertiCrypt provides support for most common reasoning patterns used in game-based proofs. In particular, CertiCrypt supports program optimizations that are commonly used in bridging steps in game-based proofs, such as game simplifications like expression propagation, procedure call inlining, code motion, and dead code elimination. More importantly, CertiCrypt provides a mechanization of the Fundamental Lemma of Game-Playing (see Appendix A), that allows to bound the difference in the probability of an event in two different games by the probability of a designated failure event. This allows to analyze simulation-based reductions that are not tight by bounding the gap by the probability of failure of the simulation.

Following a foundational approach to verification, the soundness of all the above reasoning mechanisms is verified formally in the Coq proof assistant. This is done using a relational Hoare logic, which manipulates judgments of the form

$$\models G_1 \sim G_2 : \Psi \Rightarrow \Phi$$

where $G_1, G_2$ are games and $\Psi, \Phi$ are binary relations over program memories. The above judgment is valid if for any initial memories $m_1$ and $m_2$ satisfying the pre-condition $m_1\ \Psi\ m_2$, the distributions $\llbracket \mathsf{G}_1 \rrbracket\ m_1$ and $\llbracket \mathsf{G}_2 \rrbracket\ m_2$ are related by the lifting of relation $\Phi$ to distributions. We refer the reader to [1, 3] for an appropriate definition of lifting and a more thorough description of the logic. Relational Hoare Logic subsumes observational equivalence, which is obtained by restricting pre- and post-conditions in judgments to equality relations on subsets of program variables.

Relational logic can be used to prove claims about the probability of events in games by using the following rules:

$$\frac{m_1 \; \Psi \; m_2 \qquad \models \mathsf{G}_1 \sim \mathsf{G}_2 : \Psi \Rightarrow \Phi \qquad \Phi \rightarrow (A\langle 1\rangle \leftrightarrow B\langle 2\rangle)}{\Pr\left[\mathsf{G}_1, m_1 : A\right] = \Pr\left[\mathsf{G}_2, m_2 : B\right]}$$

$$\frac{m_1 \; \Psi \; m_2 \qquad \models \mathsf{G}_1 \sim \mathsf{G}_2 : \Psi \Rightarrow \Phi \qquad \Phi \rightarrow (A\langle 1\rangle \rightarrow B\langle 2\rangle)}{\Pr\left[\mathsf{G}_1, m_1 : A\right] \leq \Pr\left[\mathsf{G}_2, m_2 : B\right]}$$

We represent relations on states as first-order formulae over tagged program variables; we use the tags $\langle 1\rangle$ and $\langle 2\rangle$ to distinguish between the value of a variable or formula in the left and right-hand side program, respectively, and $=_X$ to denote the binary relation that relates memories that coincide on variables in set $X$.

CertiCrypt inherits two essential features from the Coq proof assistant. First, since Coq is a general-purpose proof assistant, CertiCrypt is modular and extensible and can be used to reason about arbitrary mathematical constructions. In particular the language of expressions that games manipulate can be extended by the user. We take advantage of this characteristic and extend the language of expressions with values denoting elements of groups and a primitive operator that denotes a bilinear map; we enrich the simplification mechanism of CertiCrypt to compute normal forms of expressions involving this operator. Second, since any Coq proof can be automatically verified using a small and trustworthy type checker, and the reasoning principles that are supported by CertiCrypt are proved sound with respect to the semantics of games, the correctness of a machine-checked proof can be reduced to a small trusted base. This trusted base includes the security statement and the formalization of the semantics of games, but not the proof itself, which is verified by the Coq type checker.

## 3 Preliminaries

### 3.1 Bilinear maps and Bilinear Diffie-Hellman Assumption

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two cyclic groups of prime order $q$. In the remainder, we use additive notation for $\mathbb{G}_1$ and multiplicative notation for $\mathbb{G}_2$. Moreover, we let $\mathbb{G}_1^+ = \mathbb{G}_1 \setminus \{0\}$, and $\mathbb{Z}_q^+ = \{1, .., q-1\}$. An *admissible bilinear map* is a polynomially computable function $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ satisfying the following two conditions:

**Bilinearity:** for any $P, Q \in \mathbb{G}_1$ and $a, b \in \mathbb{Z}$, $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$;
**Non-degeneracy:** for any generator $P$ of $\mathbb{G}_1$, $\hat{e}(P, P)$ is a generator of $\mathbb{G}_2$.

The Bilinear Diffie-Hellman (BDH) problem is a variant of the computational Diffie-Hellman problem: given a quadruple of uniformly chosen values $(P, aP, bP, cP)$ the goal is to compute $\hat{e}(P, P)^{abc}$. The BDH assumption on a family of groups equipped with an admissible bilinear map can be formalized in terms of the following probabilistic game:

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

We define the advantage of an algorithm $\mathcal{B}$ in solving $\mathsf{BDH}$ as

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

The $\mathsf{BDH}$ assumption holds if the advantage of every probabilistic polynomial-time procedure $\mathcal{B}$ is a negligible function of a security parameter that determines the order of the groups in the family.

### 3.2 Identity-Based Encryption

In a typical setting, an IBE scheme involves a trusted third party, the Private Key Generator (PKG). The PKG generates the scheme public parameters and a master private key. On request of users, the PKG derives from the master key the private decryption key associated to a public identity by running an extraction algorithm. More formally, an IBE scheme is defined as follows.

**Definition 1 (Identity-Based Encryption scheme).** *An identity-based encryption scheme is specified by a quadruple of algorithms* $(\mathsf{Setup}, \mathcal{EX}, \mathcal{E}, \mathcal{D})$*:*

**Setup:** *Given a security parameter* $\eta$*, the* $\mathsf{Setup}$ *algorithm generates the public parameters of the scheme and a master private key;*

**Extract:** *Given a master key* $mk$ *and a public identity* $id \in \{0,1\}^\star$*,* $\mathcal{EX}(mk, id)$ *computes the corresponding decryption key* $sk$*;*

**Encrypt:** *Given a public identity* $id$ *and a message* $m$*,* $\mathcal{E}(id, m)$ *computes a ciphertext* $c$ *corresponding to the encryption of* $m$ *under* $id$*;*

**Decrypt:** *Given a private decryption key* $sk$ *and ciphertext* $c$*,* $\mathcal{D}(sk, c)$ *returns either the plaintext corresponding to the decryption of* $c$*, if it is a valid ciphertext, or a distinguished value* $\perp$ *otherwise.*

**Definition 2 (BasicIdent scheme).** *Let* $\mathbb{G}_1$*,* $\mathbb{G}_2$ *be two (families of) cyclic groups of prime order* $q$ *equipped with a bilinear map* $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$*, and let* $\mathcal{H}_1 : \{0,1\}^\star \to \mathbb{G}_1^+$*,* $\mathcal{H}_2 : \mathbb{G}_2 \to \{0,1\}^n$ *be two hash functions for some* $n \in \mathbb{N}$*.* *BasicIdent is defined by the following algorithms:*

$$\begin{array}{lcl}
\mathsf{Setup}(\eta) & : & P \xleftarrow{\$} \mathbb{G}_1^+;\ a \xleftarrow{\$} \mathbb{Z}_q^+;\ P_{pub} \leftarrow aP;\ \mathsf{return}\ ((P, P_{pub}), a) \\
\mathcal{EX}(a, id) & : & Q_{id} \leftarrow \mathcal{H}_1(id);\ \mathsf{return}\ aQ_{id} \\
\mathcal{E}(id, m) & : & Q_{id} \leftarrow \mathcal{H}_1(id);\ c \xleftarrow{\$} \mathbb{Z}_q^+;\ m' \leftarrow \mathcal{H}_2(e(Q_{id}, P_{pub})^c); \\
 & & \mathsf{return}\ (cP, m \oplus m') \\
\mathcal{D}(sk, (u, v)) & : & \mathsf{return}\ v \oplus \mathcal{H}_2(\hat{e}(sk, u))
\end{array}$$

**Definition 3 (Semantic security against chosen-plaintext attacks).** *The semantic security of an IBE scheme against chosen-plaintext attacks is defined by means of the following probabilistic experiment parametrized by an adversary $\mathcal{A}$:*

```
Game $\mathsf{G}_{\textsf{IND-ID-CPA}}$ :
  $(\mathsf{params}, mk) \leftarrow \mathsf{Setup}(\eta)$;
  $(m_0, m_1, id_{\mathcal{A}}) \leftarrow \mathcal{A}_1(\mathsf{params})$;
  $b \xleftarrow{\$} \{0,1\}$;
  $c \leftarrow \mathcal{E}(id_{\mathcal{A}}, m_b)$;
  $b_{\mathcal{A}} \leftarrow \mathcal{A}_2(c)$
```

*The two phases of the adversary $\mathcal{A}$ are modelled by two procedures $\mathcal{A}_1$ and $\mathcal{A}_2$ that can communicate through shared variables and have oracle access to a private-key extraction oracle (but not to a decryption oracle). In the first phase, $\mathcal{A}_1$ chooses two plaintexts and a challenge identity $id_{\mathcal{A}}$, while in the second phase $\mathcal{A}_2$ outputs a guess $b_{\mathcal{A}}$ for $b$. During the second phase of the experiment $\mathcal{A}_2$ is not allowed to query $id_{\mathcal{A}}$ to the extraction oracle. The IND-ID-CPA-advantage of $\mathcal{A}$ is defined as*

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

*An IBE scheme is semantically secure if every probabilistic polynomial-time adversary $\mathcal{A}$ has only a negligible advantage.*

## 4 Security of the Boneh-Franklin BasicIdent scheme

We prove that BasicIdent is semantically secure against chosen-plaintext attacks in the Random Oracle Model under the Bilinear-Diffie Hellman assumption on the underlying map $\hat{e}(\cdot,\cdot)$. The formal security statement is specified in terms of the IND-ID-CPA experiment instantiated to the BasicIdent scheme and appears at the bottom of Figure 1. It takes the form of an implication, whose premise fixes the class of adversaries considered. Specifically, the statement considers any well-formed IND-ID-CPA adversary $\mathcal{A}$ that makes at most $q_{\mathcal{H}_1}$ queries to oracle $\mathcal{H}_1$, at most $q_{\mathcal{H}_2}$ queries to oracle $\mathcal{H}_2$, and exactly $q_{\mathcal{EX}}$ queries to oracle $\mathcal{EX}$, and that does not query the $\mathcal{EX}$ oracle with the identity $id_{\mathcal{A}}$ it chooses to attack. An adversary $\mathcal{A}$ is well-formed if it does not read or write any global variables besides its own. The conclusion of the statement upper bounds the advantage of the adversary $\mathcal{A}$ in terms of the advantage of an algorithm $\mathcal{B}$ in solving the BDH problem. The code of an algorithm $\mathcal{B}$ that uses $\mathcal{A}$ as a subroutine and achieves the bound in the statement is given in the next section. Theorem 1 summarizes in simpler terms the result that we prove, which coincides with the one given in [7, Theorem 4.1].

**Theorem 1 (IND-ID-CPA security of BasicIdent).** *Let $\mathcal{A}$ be an adversary against the IND-ID-CPA security of BasicIdent. Suppose $\mathcal{A}$ executes within time $t_{\mathcal{A}}$ and makes at most $q_{\mathcal{H}_1} > 0$ queries to $\mathcal{H}_1$, $q_{\mathcal{H}_2} > 0$ queries to $\mathcal{H}_2$, and exactly*

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

```
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 $id \notin$ dom$(\boldsymbol{L_1})$ then
   $R \overset{\$}{\leftarrow} \mathbb{G}_1^+$;
   $\boldsymbol{L_1}(id) \leftarrow R$
 return $\boldsymbol{L_1}(id)$

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

$$\forall \mathcal{A}.\ \mathsf{WF}(\mathcal{A}) \wedge \Pr\left[\mathsf{G}_{\mathsf{IND\text{-}ID\text{-}CPA}} : id_{\mathcal{A}} \notin \boldsymbol{L_3} \wedge |\boldsymbol{L_1}| \leq q_{\mathcal{H}_1} \wedge |\boldsymbol{L_2}| \leq q_{\mathcal{H}_2} \wedge |\boldsymbol{L_3}| = q_{\mathcal{EX}}\right] = 1$$
$$\implies \exists \mathcal{B}.\ \mathbf{Adv}^{\mathcal{B}}_{\mathsf{BDH}} \geq \mathbf{Adv}^{\mathcal{A}}_{\mathsf{IND\text{-}ID\text{-}CPA}} \ \frac{2 q_{\mathcal{EX}}^{q_{\mathcal{EX}}}}{q_{\mathcal{H}_2}(1+q_{\mathcal{EX}})^{1+q_{\mathcal{EX}}}}$$

**Fig. 1.** Formal statement of the IND-ID-CPA security of BasicIdent

*$q_{\mathcal{EX}} > 0$ queries to the extraction oracle $\mathcal{EX}$. Then, there exists an algorithm $\mathcal{B}$ that executes within time $t_{\mathcal{B}} = O(t_{\mathcal{A}})$ such that*

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

The proof is organized as a sequence of games (the sequence is given as input to CertiCrypt); an outline is given in Figures 2-4. In the figure, each game is shown alongside the code of the oracles made available to adversary $\mathcal{A}$ and global variables are typeset in boldface. Fragments of code displayed inside a box appear only in the game whose name is surrounded by the matching box.

The initial game of the sequence is the game $\mathsf{G}_{\mathsf{IND\text{-}ID\text{-}CPA}}$ appearing in Figure 1. In the first transition from game $\mathsf{G}_{\mathsf{IND\text{-}ID\text{-}CPA}}$ to game $\mathsf{G}_1$, we inline the encryption of the challenge ciphertext and extend the state of oracle $\mathcal{H}_1$ by instrumenting its code with a list $\boldsymbol{J}$ that keeps track of the order of queries. In addition, for each of the $q_{\mathcal{H}_1}$ possible queries to $\mathcal{H}_1$, we toss a coin and store the result in a list $\boldsymbol{T}$. The coins are sampled independently following a Bernoulli distribution $\mathsf{true} \oplus_p \mathsf{false}$, that assigns true with probability $p$ and false with probability $1-p$. We prove that games $\mathsf{G}_{\mathsf{IND\text{-}ID\text{-}CPA}}$ and $\mathsf{G}_1$ are observationally equivalent with respect to $d$ and $d_A$, and thus:

$$\Pr\left[\mathsf{G}_{\mathsf{IND\text{-}ID\text{-}CPA}} : d = d_{\mathcal{A}}\right] = \Pr\left[\mathsf{G}_1 : d = d_{\mathcal{A}}\right] \tag{1}$$

Consider the following event:

$$\mathsf{Guessed} \ \stackrel{\text{def}}{=}\ \boldsymbol{T}[\boldsymbol{J}(id_{\mathcal{A}})] \wedge \forall id \in \boldsymbol{L}_3.\ \neg \boldsymbol{T}[\boldsymbol{J}(id)]$$

Since the events $d = d_{\mathcal{A}}$ and Guessed are trivially independent, we have that

$$\Pr\left[\mathsf{G}_1 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed}\right] = \Pr\left[\mathsf{G}_1 : d = d_{\mathcal{A}}\right]\ \Pr\left[\mathsf{G}_1 : \mathsf{Guessed}\right]$$

Furthermore, a straightforward calculation gives

$$\Pr\left[\mathsf{G}_1 : \mathsf{Guessed}\right] = p(1-p)^{q_{\mathcal{EX}}} \tag{2}$$

```
$\textbf{Game } \mathsf{G}_1$ :
  $\boldsymbol{L_1}, \boldsymbol{L_2}, \boldsymbol{L_3}, \boldsymbol{J} \leftarrow$ nil;
  $\boldsymbol{a} \overset{\$}{\leftarrow} \mathbb{Z}_q^+$; $\boldsymbol{P} \overset{\$}{\leftarrow} \mathbb{G}_1^+$; $\boldsymbol{P_{pub}} \leftarrow \boldsymbol{aP}$;
  $(m_0, m_1, id_{\mathcal{A}}) \leftarrow \mathcal{A}_1(\boldsymbol{P}, \boldsymbol{P_{pub}})$;
  $d \overset{\$}{\leftarrow} \{0,1\}$;
  $Q_{\mathcal{A}} \leftarrow \mathcal{H}_1(id_{\mathcal{A}})$;
  $c \overset{\$}{\leftarrow} \mathbb{Z}_q^+$;
  $m' \leftarrow \mathcal{H}_2(\hat{e}(Q_{\mathcal{A}}, \boldsymbol{P_{pub}})^c)$;
  $y \leftarrow (c\boldsymbol{P}, m_d \oplus m')$;
  $d_{\mathcal{A}} \leftarrow \mathcal{A}_2(y)$;
  Coins
```

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

```
$\textbf{Oracle } \mathcal{H}_1(id)$ :
  if $id \notin$ dom$(\boldsymbol{L_1})$ then
    $\boldsymbol{J}(id) \leftarrow |\boldsymbol{L_1}|$;
    $R \overset{\$}{\leftarrow} \mathbb{G}_1^+$;
    $\boldsymbol{L_1}(id) \leftarrow R$
  return $\boldsymbol{L_1}(id)$

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

$$\left|\Pr\left[\mathsf{G}_1 : d = d_{\mathcal{A}}\right] - \tfrac{1}{2}\right| p(1-p)^{q_{\mathcal{EX}}} = \left|\Pr\left[\mathsf{G}_2 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed}\right] - \tfrac{1}{2}\Pr\left[\mathsf{G}_2 : \mathsf{Guessed}\right]\right|$$

```
$\textbf{Game } \dbox{\mathsf{G}_2}\ \boxed{\mathsf{G}_3}$ :
  Coins;
  $\boldsymbol{L_1}, \boldsymbol{L_2}, \boldsymbol{L_3}, \boldsymbol{V}, \boldsymbol{J} \leftarrow$ nil;
  $\boldsymbol{a}, \boldsymbol{b} \overset{\$}{\leftarrow} \mathbb{Z}_q^+$; $\boldsymbol{P} \overset{\$}{\leftarrow} \mathbb{G}_1^+$;
  $\boldsymbol{P_{pub}} \leftarrow \boldsymbol{aP}$;
  $(m_0, m_1, id_{\mathcal{A}}) \leftarrow \mathcal{A}_1(\boldsymbol{P}, \boldsymbol{P_{pub}})$;
  $d \overset{\$}{\leftarrow} \{0,1\}$;
  $Q_{\mathcal{A}} \leftarrow \mathcal{H}_1(id_{\mathcal{A}})$; $c \overset{\$}{\leftarrow} \mathbb{Z}_q^+$;
  $\dbox{\begin{array}{l} m' \leftarrow \mathcal{H}_2(\hat{e}(Q_{\mathcal{A}}, \boldsymbol{P_{pub}})^c); \\ y \leftarrow (c\boldsymbol{P}, m_d \oplus m'); \end{array}}$
  $\boxed{\begin{array}{l} v' \leftarrow \boldsymbol{V}(id_{\mathcal{A}})^{-1} c \bmod q; \\ m' \leftarrow \mathcal{H}_2(\hat{e}(Q_{\mathcal{A}}, \boldsymbol{P_{pub}})^{v'}); \\ y \leftarrow (v'\boldsymbol{P}, m_d \oplus m'); \end{array}}$
  $d_{\mathcal{A}} \leftarrow \mathcal{A}_2(y)$
```

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

```
$\textbf{Oracle } \mathcal{H}_1(id)$ :
  if $id \notin$ dom$(\boldsymbol{L_1})$ then
    $\boldsymbol{J}(id) \leftarrow |\boldsymbol{L_1}|$;
    $v \overset{\$}{\leftarrow} \mathbb{Z}_q^+$; $\boldsymbol{V}(id) \leftarrow v$;
    if $\boldsymbol{T}[|\boldsymbol{L_1}|]$ then
      $\boldsymbol{L_1}(id) \leftarrow \boldsymbol{b}v\boldsymbol{P}$
    else
      $\boldsymbol{L_1}(id) \leftarrow v\boldsymbol{P}$
  return $\boldsymbol{L_1}(id)$

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

$$\left|\Pr\left[\mathsf{G}_2 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed}\right] - \tfrac{1}{2}\Pr\left[\mathsf{G}_2 : \mathsf{Guessed}\right]\right| = \left|\Pr\left[\mathsf{G}_4 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed}\right] - \tfrac{1}{2}\Pr\left[\mathsf{G}_4 : \mathsf{Guessed}\right]\right|$$

```
$\textbf{Game } \dbox{\mathsf{G}_{3'}}\ \boxed{\mathsf{G}_4}$ :
  Coins;
  $\boldsymbol{L_1}, \boldsymbol{L_2}, \boldsymbol{L_3}, \boldsymbol{V}, \boldsymbol{J} \leftarrow$ nil;
  $\boldsymbol{a}, \boldsymbol{b}, c \overset{\$}{\leftarrow} \mathbb{Z}_q^+$; $\boldsymbol{P} \overset{\$}{\leftarrow} \mathbb{G}_1^+$;
  $\boldsymbol{P_{pub}} \leftarrow \boldsymbol{aP}$;
  $(m_0, m_1, id_{\mathcal{A}}) \leftarrow \mathcal{A}_1(\boldsymbol{P}, \boldsymbol{P_{pub}})$;
  $Q_{\mathcal{A}} \leftarrow \mathcal{H}_1(id_{\mathcal{A}})$;
  $v' \leftarrow \boldsymbol{V}(id_{\mathcal{A}})^{-1} c \bmod q$;
  if $\boldsymbol{T}[\boldsymbol{J}(id_{\mathcal{A}})]$ then
    $m' \leftarrow \mathcal{H}_2(\hat{e}(\boldsymbol{P}, \boldsymbol{P})^{\boldsymbol{abc}})$
  else
    $\mathbf{bad} \leftarrow$ true;
    $\dbox{m' \leftarrow \mathcal{H}_2(\hat{e}(Q_{\mathcal{A}}, \boldsymbol{P_{pub}})^{v'})}$
    $\boxed{m' \leftarrow \mathcal{H}_2(\hat{e}(\boldsymbol{P}, \boldsymbol{P})^{\boldsymbol{abc}})}$
  $d \overset{\$}{\leftarrow} \{0,1\}$; $y \leftarrow (v'\boldsymbol{P}, m_d \oplus m')$;
  $d_{\mathcal{A}} \leftarrow \mathcal{A}_2(y)$
```

```
$\textbf{Oracle } \mathcal{EX}(id)$ :
  if $id \notin \boldsymbol{L_3}$ then
    $\boldsymbol{L_3} \leftarrow id :: \boldsymbol{L_3}$
  $Q \leftarrow \mathcal{H}_1(id)$;
  if $\boldsymbol{T}[\boldsymbol{J}(id)]$ then
    return $\boldsymbol{V}(id)\boldsymbol{P_{pub}}$
  else
    $\mathbf{bad} \leftarrow$ true;
    $\dbox{\text{return } \boldsymbol{a}Q}$
    $\boxed{\text{return } \boldsymbol{V}(id)\boldsymbol{P_{pub}}}$
```

```
$\textbf{Oracle } \mathcal{H}_1(id)$ :
  if $id \notin$ dom$(\boldsymbol{L_1})$ then
    $\boldsymbol{J}(id) \leftarrow |\boldsymbol{L_1}|$;
    $v \overset{\$}{\leftarrow} \mathbb{Z}_q^+$; $\boldsymbol{V}(id) \leftarrow v$;
    if $\boldsymbol{T}[|\boldsymbol{L_1}|]$ then
      $\boldsymbol{L_1}(id) \leftarrow \boldsymbol{b}v\boldsymbol{P}$
    else
      $\boldsymbol{L_1}(id) \leftarrow v\boldsymbol{P}$
  return $\boldsymbol{L_1}(id)$

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

Coins $\overset{\text{def}}{=}$ $\boldsymbol{T} \leftarrow$ nil; while $|\boldsymbol{T}| < q_{\mathcal{H}_1}$ do $(t \overset{\$}{\leftarrow}$ true $\oplus_p$ false; $\boldsymbol{T} \leftarrow t :: \boldsymbol{T})$

**Fig. 2.** Outline of the proof of IND-ID-CPA security of BasicIdent

$\left|\Pr\left[\mathsf{G}_4 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed}\right] - \frac{1}{2}\Pr\left[\mathsf{G}_4 : \mathsf{Guessed}\right]\right| = \left|\Pr\left[\mathsf{G}_5 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed}\right] - \frac{1}{2}\Pr\left[\mathsf{G}_5 : \mathsf{Guessed}\right]\right|$

```
Game $\mathsf{G}_5$ :
  Coins;
  $\boldsymbol{L_1}, \boldsymbol{L_2}, \boldsymbol{L_3}, \boldsymbol{V}, \boldsymbol{J} \leftarrow$ nil;
  $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c} \xleftarrow{\$} \mathbb{Z}_q^+;\ \boldsymbol{P} \xleftarrow{\$} \mathbb{G}_1^+;$
  $\boldsymbol{P_{pub}} \leftarrow \boldsymbol{aP};\ \boldsymbol{m}^\star \xleftarrow{\$} \{0,1\}^n;$
  $(m_0, m_1, id_{\mathcal{A}}) \leftarrow \mathcal{A}_1(\boldsymbol{P}, \boldsymbol{P_{pub}});$
  $Q_{\mathcal{A}} \leftarrow \mathcal{H}_1(id_{\mathcal{A}});$
  $v' \leftarrow \boldsymbol{V}(id_{\mathcal{A}})^{-1}\boldsymbol{c} \bmod q;$
  $m' \leftarrow \mathcal{H}_2(\hat{e}(\boldsymbol{P}, \boldsymbol{P})^{\boldsymbol{abc}});$
  $d \xleftarrow{\$} \{0,1\};\ y \leftarrow (v'\boldsymbol{P}, m_d \oplus m');$
  $d_{\mathcal{A}} \leftarrow \mathcal{A}_2(y)$
```

```
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 $\boldsymbol{V}(id)\boldsymbol{P_{pub}}$
```

```
Oracle $\mathcal{H}_1(id)$ :
  if $id \notin$ dom$(\boldsymbol{L_1})$ then
    $\boldsymbol{J}(id) \leftarrow |\boldsymbol{L_1}|;$
    $v \xleftarrow{\$} \mathbb{Z}_q^+;\ \boldsymbol{V}(id) \leftarrow v;$
    if $\boldsymbol{T}[|\boldsymbol{L_1}|]$ then
      $\boldsymbol{L_1}(id) \leftarrow \boldsymbol{b}v\boldsymbol{P}$
    else
      $\boldsymbol{L_1}(id) \leftarrow v\boldsymbol{P}$
  return $\boldsymbol{L_1}(id)$

Oracle $\mathcal{H}_2(r)$ :
  if $r \notin$ dom$(\boldsymbol{L_2})$ then
    if $r = \hat{e}(\boldsymbol{P}, \boldsymbol{P})^{\boldsymbol{abc}}$ then $m \leftarrow \boldsymbol{m}^\star$
    else $m \xleftarrow{\$} \{0,1\}^n$
    $\boldsymbol{L_2}(r) \leftarrow m$
  else $m \leftarrow \boldsymbol{L_2}(r)$
  return $m$
```

$\left|\Pr\left[\mathsf{G}_5 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed}\right] - \frac{1}{2}\Pr\left[\mathsf{G}_5 : \mathsf{Guessed}\right]\right| = \left|\Pr\left[\mathsf{G}_6 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed}\right] - \frac{1}{2}\Pr\left[\mathsf{G}_6 : \mathsf{Guessed}\right]\right|$

```
Game $\mathsf{G}_{6'}$ $\boxed{\mathsf{G}_6}$ :
  Coins;
  $\boldsymbol{L_1}, \boldsymbol{L_2}, \boldsymbol{L_3}, \boldsymbol{V}, \boldsymbol{J} \leftarrow$ nil;
  $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c} \xleftarrow{\$} \mathbb{Z}_q^+;\ \boldsymbol{P} \xleftarrow{\$} \mathbb{G}_1^+;$
  $\boldsymbol{P_{pub}} \leftarrow \boldsymbol{aP};\ \boldsymbol{m}^\star \xleftarrow{\$} \{0,1\}^n;$
  $(m_0, m_1, id_{\mathcal{A}}) \leftarrow \mathcal{A}_1(\boldsymbol{P}, \boldsymbol{P_{pub}});$
  $Q_{\mathcal{A}} \leftarrow \mathcal{H}_1(id_{\mathcal{A}});$
  $v' \leftarrow \boldsymbol{V}(id_{\mathcal{A}})^{-1}\boldsymbol{c} \bmod q;$
  $d \xleftarrow{\$} \{0,1\};$
  $y \leftarrow (v'\boldsymbol{P}, m_d \oplus \boldsymbol{m}^\star);$
  $d_{\mathcal{A}} \leftarrow \mathcal{A}_2(y)$
```

```
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 $\boldsymbol{V}(id)\boldsymbol{P_{pub}}$
```

```
Oracle $\mathcal{H}_1(id)$ :
  if $id \notin$ dom$(\boldsymbol{L_1})$ then
    $\boldsymbol{J}(id) \leftarrow |\boldsymbol{L_1}|;$
    $v \xleftarrow{\$} \mathbb{Z}_q^+;\ \boldsymbol{V}(id) \leftarrow v;$
    if $\boldsymbol{T}[|\boldsymbol{L_1}|]$ then
      $\boldsymbol{L_1}(id) \leftarrow \boldsymbol{b}v\boldsymbol{P}$
    else
      $\boldsymbol{L_1}(id) \leftarrow v\boldsymbol{P}$
  return $\boldsymbol{L_1}(id)$

Oracle $\mathcal{H}_2(r)$ :
  if $r = \hat{e}(\boldsymbol{P}, \boldsymbol{P})^{\boldsymbol{abc}}$ then
    $\boxed{\textbf{bad} \leftarrow \text{true};}$
    $m \leftarrow \boldsymbol{m}^\star;$
    if $r \notin$ dom$(\boldsymbol{L_2})$ then $\boldsymbol{L_2}(r) \leftarrow \boldsymbol{m}^\star$
  else
    if $r \notin$ dom$(\boldsymbol{L_2})$ then
      $m \xleftarrow{\$} \{0,1\}^n;\ \boldsymbol{L_2}(r) \leftarrow m$
    else $m \leftarrow \boldsymbol{L_2}(r)$
  return $m$
```

**Fig. 3.** Outline of the proof of IND-ID-CPA security of BasicIdent

$$2\left|\Pr\left[\mathsf{G}_6 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed}\right] - \tfrac{1}{2}\Pr\left[\mathsf{G}_6 : \mathsf{Guessed}\right]\right| \leq \Pr\left[\mathsf{G}_6 : \mathsf{S}\right] = \Pr\left[\mathsf{G}_7 : \mathsf{S}\right]$$

```
Game $\dbox{\mathsf{G}_{7'}}$ $\boxed{\mathsf{G}_7}$ :
 Coins;
 $\boldsymbol{L_1}, \boldsymbol{L_2}, \boldsymbol{L_3}, \boldsymbol{V}, \boldsymbol{J} \leftarrow$ nil;
 $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c} \xleftarrow{\$} \mathbb{Z}_q^+;\ \boldsymbol{P} \xleftarrow{\$} \mathbb{G}_1^+;$
 $\boldsymbol{P_{pub}} \leftarrow \boldsymbol{aP};\ \dbox{\boldsymbol{m}^\star \xleftarrow{\$} \{0,1\}^n;}$
 $(m_0, m_1, id_{\mathcal{A}}) \leftarrow \mathcal{A}_1(\boldsymbol{P}, \boldsymbol{P_{pub}});$
 $Q_{\mathcal{A}} \leftarrow \mathcal{H}_1(id_{\mathcal{A}});$
 $v' \leftarrow \boldsymbol{V}(id_{\mathcal{A}})^{-1}\boldsymbol{c} \bmod q;$
 $d \xleftarrow{\$} \{0,1\};$
 $\dbox{y \leftarrow (v'\boldsymbol{P}, m_d \oplus \boldsymbol{m}^\star);}$
 $\boxed{R \xleftarrow{\$} \{0,1\}^n;\ y \leftarrow (v'\boldsymbol{P}, R);}$
 $d_{\mathcal{A}} \leftarrow \mathcal{A}_2(y)$
```

```
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 $\boldsymbol{V}(id)\boldsymbol{P_{pub}}$
```

```
Oracle $\mathcal{H}_1(id)$ :
 if $id \notin$ dom$(\boldsymbol{L_1})$ then
   $\boldsymbol{J}(id) \leftarrow |\boldsymbol{L_1}|;$
   $v \xleftarrow{\$} \mathbb{Z}_q^+;\ \boldsymbol{V}(id) \leftarrow v;$
   if $\boldsymbol{T}[|\boldsymbol{L_1}|]$ then $\boldsymbol{L_1}(id) \leftarrow \boldsymbol{b}v\boldsymbol{P}$
   else $\boldsymbol{L_1}(id) \leftarrow v\boldsymbol{P}$
 return $\boldsymbol{L_1}(id)$

Oracle $\mathcal{H}_2(r)$ :
 if $r = \hat{e}(\boldsymbol{P}, \boldsymbol{P})^{\boldsymbol{abc}}$ then
   $\mathbf{bad} \leftarrow$ true;
   if $r \notin$ dom$(\boldsymbol{L_2})$ then
     $m \xleftarrow{\$} \{0,1\}^n;\ \boldsymbol{L_2}(r) \leftarrow m$
   else $m \leftarrow \boldsymbol{L_2}(r)$
 else
   if $r \notin$ dom$(\boldsymbol{L_2})$ then
     $m \xleftarrow{\$} \{0,1\}^n;\ \boldsymbol{L_2}(r) \leftarrow m$
   else $m \leftarrow \boldsymbol{L_2}(r)$
 return $m$
```

$$q_{\mathcal{H}_2}^{-1}\Pr\left[\mathsf{G}_7 : \mathsf{S}\right] \leq \Pr\left[\mathsf{G}_{\mathsf{BDH}} : z = \hat{e}(\boldsymbol{P}, \boldsymbol{P})^{\boldsymbol{abc}}\right]$$

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

$\mathcal{B}(P_0, P_1, P_2, P_3)$ :
 Coins;
 $\boldsymbol{L_1}, \boldsymbol{L_2}, \boldsymbol{L_3}, \boldsymbol{V} \leftarrow$ nil;
 $\boldsymbol{P} \leftarrow P_0;\ \boldsymbol{P_{pub}} \leftarrow P_1;\ \boldsymbol{P'} \leftarrow P_2;$
 $(m_0, m_1, id_{\mathcal{A}}) \leftarrow \mathcal{A}_1(\boldsymbol{P}, \boldsymbol{P_{pub}});$
 $Q_{\mathcal{A}} \leftarrow \mathcal{H}_1(id_{\mathcal{A}});$
 $v' \leftarrow \boldsymbol{V}(id_{\mathcal{A}})^{-1};$
 $R \xleftarrow{\$} \{0,1\}^n;\ y \leftarrow (v'P_3, R);$
 $d_{\mathcal{A}} \leftarrow \mathcal{A}_2(y);$
 $i \xleftarrow{\$} [1..|\boldsymbol{L_2}|];$ return 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 $\boldsymbol{V}(id)\boldsymbol{P_{pub}}$
```

```
Oracle $\mathcal{H}_1(id)$ :
 if $id \notin$ 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 $r \notin$ dom$(\boldsymbol{L_2})$ then
   $m \xleftarrow{\$} \{0,1\}^n;\ \boldsymbol{L_2}(r) \leftarrow m$
 else $m \leftarrow \boldsymbol{L_2}(r)$
return $m$
```

**Fig. 4.** Outline of the proof of IND-ID-CPA security of BasicIdent

In game $\mathsf{G}_2$, we hoist the loop that samples the $q_{\mathcal{H}_1}$ coins in $\boldsymbol{T}$ to the beginning of the game and change the way oracle $\mathcal{H}_1$ answers to queries. To answer the $i$-th hash query, $\mathcal{H}_1$ chooses uniformly a value $v \in \mathbb{Z}_q^+$ and replies according to the $i$-th entry in $\boldsymbol{T}$: if it is $\mathsf{true}$, replies with $\boldsymbol{b}v\boldsymbol{P}$, where $\boldsymbol{b}$ is uniformly chosen at the beginning of the game; otherwise replies with $v\boldsymbol{P}$. Since the value $v$ acts as a one-time pad, in both cases the answers are distributed uniformly and independently from previous queries, and are thus perfectly indistinguishable from those of a random oracle. We prove this by first proving the validity of the following algebraic equivalence:

$$\models R \overset{\$}{\twoheadleftarrow} \mathbb{G}_1^+;\ v \leftarrow (\log R / \log Q) \bmod q \sim v \overset{\$}{\twoheadleftarrow} \mathbb{Z}_q^+;\ R \leftarrow vQ : \Psi \Rightarrow \Phi$$

where

$$\Psi \stackrel{\text{def}}{=} Q\langle 1\rangle = Q\langle 2\rangle \wedge\ \log Q\langle 1\rangle \neq 0 \qquad \Phi \stackrel{\text{def}}{=} =_{\{v,R\}}$$

We then apply this equivalence twice to show that no matter what branch is taken in the conditional in $\mathcal{H}_1$, the value of $\boldsymbol{L_1}(id)$ will be uniformly distributed. In one case, we take $Q = \boldsymbol{bP}$, while in the other we simply take $Q = \boldsymbol{P}$. We conclude that

$$\Pr[\mathsf{G}_1 : \mathsf{Guessed}] = \Pr[\mathsf{G}_2 : \mathsf{Guessed}] \tag{3}$$
$$\Pr[\mathsf{G}_1 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed}] = \Pr[\mathsf{G}_2 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed}] \tag{4}$$

Game $\mathsf{G}_3$ is obtained from game $\mathsf{G}_2$ by padding the random value $c$ used to encrypt $m_d$ with the value $\boldsymbol{V}(id_{\mathcal{A}})^{-1}$. To justify this transformation we prove that the assertion $0 < \boldsymbol{V}(id_{\mathcal{A}}) < q$ holds just before sampling $c$ in $\mathsf{G}_3$, and apply the rule:

$$\models x \overset{\$}{\twoheadleftarrow} \mathbb{Z}_q^+;\ y \leftarrow zx \bmod q \sim y \overset{\$}{\twoheadleftarrow} \mathbb{Z}_q^+;\ x \leftarrow z^{-1}y \bmod q \\ : (=_{\{z\}} \wedge\ 0 < z\langle 2\rangle < q) \implies =_{\{x,y,z\}}$$

to show that the distribution of the challenge ciphertext $y$ is the same in both games. To prove the above assertion, we first show that

$$\forall id \in \mathsf{dom}(\boldsymbol{L_1}).\ 0 < \boldsymbol{V}(id) < q$$

is an invariant of $\mathcal{A}_1$ that it is established right after the initialization of $\boldsymbol{L_1}$ and that after making the oracle call $\mathcal{H}_1(id_{\mathcal{A}})$, the public key $id_{\mathcal{A}}$ necessarily belongs to the domain of $\boldsymbol{L_1}$. Therefore we have

$$\Pr[\mathsf{G}_2 : \mathsf{Guessed}] = \Pr[\mathsf{G}_3 : \mathsf{Guessed}] \tag{5}$$
$$\Pr[\mathsf{G}_2 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed}] = \Pr[\mathsf{G}_3 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed}] \tag{6}$$

In game $\mathsf{G}_4$ we "inject" the challenge $\hat{e}(\boldsymbol{P}, \boldsymbol{P})^{\boldsymbol{abc}}$ into the ciphertext $y$ and we change the simulation of the extraction oracle to eliminate its dependency on the private master key $\boldsymbol{a}$. The former is achieved by replacing the bitstring $m'$ used to pad $m_d$ with $\mathcal{H}_2\left(\hat{e}(\boldsymbol{P}, \boldsymbol{P})^{\boldsymbol{abc}}\right)$, whereas the latter is achieved by replacing the return expression of oracle $\mathcal{H}_1$ with $\boldsymbol{V}(id)\ \boldsymbol{P_{pub}}$.

Observe that if the coin $\boldsymbol{T}[\boldsymbol{J}(id_{\mathcal{A}})]$ used to decide how the hash value $\mathcal{H}_1(id_{\mathcal{A}})$ is computed is $\mathsf{true}$, then $\hat{e}(Q_{\mathcal{A}}, \boldsymbol{P_{pub}})^{v'} = \hat{e}(\boldsymbol{P}, \boldsymbol{P})^{\boldsymbol{abc}}$. Furthermore, if for every extraction query $id$ we have $\boldsymbol{T}[\boldsymbol{J}(id)] = \mathsf{false}$, then $\boldsymbol{a}\mathcal{H}_1(id) = \boldsymbol{V}(id)\boldsymbol{P_{pub}}$. This motivates the introduction of a hybrid game $\mathsf{G}_{3'}$, for which we prove the following invariant:

$$\begin{array}{l}(\boldsymbol{P_{pub}} = \boldsymbol{aP} \wedge\ \boldsymbol{L_3} \subseteq \mathsf{dom}(\boldsymbol{L_1})\ \wedge\ \forall id \in \mathsf{dom}(\boldsymbol{L_1}).\ 0 < \boldsymbol{V}(id) < q)\langle 1\rangle \ \wedge \\ (\forall id \in \mathsf{dom}(\boldsymbol{L_1}).\ \boldsymbol{L_1}(id) = \mathsf{if}\ \boldsymbol{T}[\boldsymbol{J}(id)]\ \mathsf{then}\ \boldsymbol{bV}(id)\boldsymbol{P}\ \mathsf{else}\ \boldsymbol{V}(id)\boldsymbol{P})\ \langle 1\rangle\end{array}$$

from which we can prove that

$$\Pr[\mathsf{G}_3 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed}] = \Pr[\mathsf{G}_{3'} : d = d_{\mathcal{A}} \wedge \mathsf{Guessed}]$$

Now, games $\mathsf{G}_{3'}$ and $\mathsf{G}_4$ differ only on code appearing after the flag **bad** is set and we can apply the Fundamental Lemma to prove that

$$\Pr[\mathsf{G}_{3'} : d = d_{\mathcal{A}} \wedge \mathsf{Guessed} \wedge \neg\mathbf{bad}] = \Pr[\mathsf{G}_4 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed} \wedge \neg\mathbf{bad}]$$

Observe that $\mathsf{Guessed} \Rightarrow \neg\mathbf{bad}$ is a post-condition of both $\mathsf{G}_{3'}$ and $\mathsf{G}_4$, and therefore

$$\Pr[\mathsf{G}_{3'} : d = d_{\mathcal{A}} \wedge \mathsf{Guessed}] = \Pr[\mathsf{G}_4 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed}]$$

Finally by transitivity we have

$$\Pr[\mathsf{G}_3 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed}] = \Pr[\mathsf{G}_4 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed}] \tag{7}$$

and analogously,

$$\Pr[\mathsf{G}_3 : \mathsf{Guessed}] = \Pr[\mathsf{G}_4 : \mathsf{Guessed}] \tag{8}$$

In game $\mathsf{G}_5$ we eagerly sample the hash value $\boldsymbol{m}^\star$ that $\mathcal{H}_2$ gives in response to query $\hat{e}(\boldsymbol{P}, \boldsymbol{P})^{\boldsymbol{abc}}$. To formally justify this transformation we use the logic of swapping statements presented in [4], which constitutes a general technique to reason about inter-procedural code motion and can be readily specialized to deal with this kind of bridging step. The logic yields equations

$$\Pr[\mathsf{G}_4 : \mathsf{Guessed}] = \Pr[\mathsf{G}_5 : \mathsf{Guessed}] \tag{9}$$

$$\Pr[\mathsf{G}_4 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed}] = \Pr[\mathsf{G}_5 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed}] \tag{10}$$

Our goal now is to make explicit that the message used to pad $m_d$ during its encryption is $\boldsymbol{m}^\star$. Note that just inlining in $\mathsf{G}_5$ the call to $\mathcal{H}_2$ made when encrypting the challenge ciphertext would result in the inclusion of the conditional statement

$$\mathsf{if}\ \hat{e}(\boldsymbol{P}, \boldsymbol{P})^{\boldsymbol{abc}} \notin \mathsf{dom}(\boldsymbol{L_2})\ \mathsf{then}\ \boldsymbol{L_2}(\hat{e}(\boldsymbol{P}, \boldsymbol{P})^{\boldsymbol{abc}}) \leftarrow \boldsymbol{m}^\star$$

which depends on $\hat{e}(\boldsymbol{P}, \boldsymbol{P})^{\boldsymbol{abc}}$, while we want to efficiently simulate an environment for $\mathcal{A}$ in terms of only $\boldsymbol{P}$, $\boldsymbol{aP}$, $\boldsymbol{bP}$, and $\boldsymbol{cP}$. We therefore introduce an intermediate game $\mathsf{G}_{6'}$, where the oracle $\mathcal{H}_2$ does not store in its memory the answer to a $\hat{e}(\boldsymbol{P},\boldsymbol{P})^{\boldsymbol{abc}}$ query.

The equivalence between games $\mathsf{G}_5$ and $\mathsf{G}_{6'}$ is proved by inlining the call to oracle $\mathcal{H}_2$ in $\mathsf{G}_5$ and by means of the following relational invariant

$$\begin{aligned} I_{5\to 6'} \stackrel{\text{def}}{=} & \left(\hat{e}(\boldsymbol{P},\boldsymbol{P})^{\boldsymbol{abc}} \in \mathsf{dom}(\boldsymbol{L_2}) \Rightarrow \boldsymbol{L_2}\left(\hat{e}(\boldsymbol{P},\boldsymbol{P})^{\boldsymbol{abc}}\right) = \boldsymbol{m}^\star\right)\langle 1\rangle \wedge \\ & \forall x \neq \hat{e}(\boldsymbol{P},\boldsymbol{P})^{\boldsymbol{abc}}\langle 1\rangle.\ \boldsymbol{L_2}\langle 1\rangle(x) = \boldsymbol{L_2}\langle 2\rangle(x) \end{aligned}$$

The equivalence between games $\mathsf{G}_{6'}$ and $\mathsf{G}_6$ relies on the dual invariant

$$\begin{aligned} I_{6'\to 6} \stackrel{\text{def}}{=} & \left(\hat{e}(\boldsymbol{P},\boldsymbol{P})^{\boldsymbol{abc}} \in \mathsf{dom}(\boldsymbol{L_2}) \Rightarrow \boldsymbol{L_2}\left(\hat{e}(\boldsymbol{P},\boldsymbol{P})^{\boldsymbol{abc}}\right) = \boldsymbol{m}^\star\right)\langle 2\rangle \wedge \\ & \forall x \neq \hat{e}(\boldsymbol{P},\boldsymbol{P})^{\boldsymbol{abc}}\langle 1\rangle.\ \boldsymbol{L_2}\langle 1\rangle(x) = \boldsymbol{L_2}\langle 2\rangle(x) \end{aligned}$$

From these two equivalences we have

$$\Pr\left[\mathsf{G}_5 : \mathsf{Guessed}\right] = \Pr\left[\mathsf{G}_6 : \mathsf{Guessed}\right] \tag{11}$$

$$\Pr\left[\mathsf{G}_5 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed}\right] = \Pr\left[\mathsf{G}_6 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed}\right] \tag{12}$$

Observe that if in game $\mathsf{G}_6$ the value $\hat{e}(\boldsymbol{P},\boldsymbol{P})^{\boldsymbol{abc}}$ is never queried to $\mathcal{H}_2$, then the second component of the challenge $y$ looks completely random to the adversary. This motivates the definition of game $\mathsf{G}_7$, where we also recover the usual implementation of $\mathcal{H}_2$ as a random oracle. To prove this, we introduce an intermediate game $\mathsf{G}_{7'}$ that computes the challenge $y$ given to the adversary as in game $\mathsf{G}_6$, but whose implementation of oracle $\mathcal{H}_2$ is the same as in $\mathsf{G}_7$. This results in two games $\mathsf{G}_6$ and $\mathsf{G}_{7'}$ that are syntactically identical except at program points where the flag **bad** is set. By the Fundamental Lemma we have

$$\begin{aligned} & \Pr\left[\mathsf{G}_6 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed} \wedge \neg\mathsf{S} \wedge \neg\mathbf{bad}\right] \\ & \quad = \Pr\left[\mathsf{G}_{7'} : d = d_{\mathcal{A}} \wedge \mathsf{Guessed} \wedge \neg\mathsf{S} \wedge \neg\mathbf{bad}\right] \end{aligned}$$

where $\mathsf{S}$ is an event defined as

$$\mathsf{S} \stackrel{\text{def}}{=} \hat{e}(\boldsymbol{P},\boldsymbol{P})^{\boldsymbol{abc}} \in \mathsf{dom}(\boldsymbol{L_2})$$

Additionally, we prove that $\neg\mathsf{S} \Rightarrow \neg\mathbf{bad}$ is an invariant of both $\mathsf{G}_6$ and $\mathsf{G}_{7'}$, and thus

$$\Pr\left[\mathsf{G}_6 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed} \wedge \neg\mathsf{S}\right] = \Pr\left[\mathsf{G}_{7'} : d = d_{\mathcal{A}} \wedge \mathsf{Guessed} \wedge \neg\mathsf{S}\right]$$

We next prove that

$$\Pr\left[\mathsf{G}_{7'} : d = d_{\mathcal{A}} \wedge \mathsf{Guessed} \wedge \neg\mathsf{S}\right] = \Pr\left[\mathsf{G}_7 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed} \wedge \neg\mathsf{S}\right]$$

using the following algebraic property of exclusive-or, known as *optimistic sampling*:

$$\begin{aligned} \vDash \boldsymbol{m}^\star \stackrel{\$}{\leftarrow} \{0,1\}^n; R \leftarrow m_d \oplus \boldsymbol{m}^\star \sim R \stackrel{\$}{\leftarrow} \{0,1\}^n; \boldsymbol{m}^\star \leftarrow m_d \oplus R & \\ : =_{\{m_d\}} \implies =_{\{m_d, \boldsymbol{m}^\star, R\}} & \end{aligned}$$

This, together with the previous equation implies

$$\Pr\left[\mathsf{G}_6 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed} \wedge \neg\mathsf{S}\right] = \Pr\left[\mathsf{G}_7 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed} \wedge \neg\mathsf{S}\right] \tag{13}$$

Analogously, we have

$$\Pr\left[\mathsf{G}_6 : \mathsf{Guessed} \wedge \neg\mathsf{S}\right] = \Pr\left[\mathsf{G}_7 : \mathsf{Guessed} \wedge \neg\mathsf{S}\right] \tag{14}$$

$$\Pr\left[\mathsf{G}_6 : \neg\mathsf{S}\right] = \Pr\left[\mathsf{G}_7 : \neg\mathsf{S}\right] \tag{15}$$

In game $\mathsf{G}_7$ the challenge $y$ becomes independent of the random bit $d$. Since the guess $d_{\mathcal{A}}$ of the adversary is now completely independent from $d$, the probability of the guess being correct can be proven to be exactly $1/2$, and hence

$$\Pr\left[\mathsf{G}_7 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed} \wedge \neg\mathsf{S}\right] = \frac{1}{2}\Pr\left[\mathsf{G}_7 : \mathsf{Guessed} \wedge \neg\mathsf{S}\right] \tag{16}$$

The final game $\mathsf{G}_{\mathsf{BDH}}$ constitutes the desired reduction of the security of the scheme to the $\mathsf{BDH}$ assumption. We prove the equivalence between $\mathsf{G}_7$ and $\mathsf{G}_{\mathsf{BDH}}$ by coalescing the branches of the conditional in oracle $\mathcal{H}_2$, inlining the call $z \leftarrow \mathcal{B}(P, aP, bP, cP)$, and removing dead code. This equivalence gives

$$\Pr\left[\mathsf{G}_7 : \hat{e}(\boldsymbol{P}, \boldsymbol{P})^{\boldsymbol{abc}} \in \mathsf{dom}(\boldsymbol{L_2})\right] = \Pr\left[\mathsf{G}_{\mathsf{BDH}} : \hat{e}(P, P)^{abc} \in \mathsf{dom}(\boldsymbol{L_2})\right] \tag{17}$$

To relate the advantage $\mathbf{Adv}^{\mathcal{A}}_{\mathsf{IND\text{-}ID\text{-}CPA}}$ of adversary $\mathcal{A}$ in the initial game with the advantage $\mathbf{Adv}^{\mathcal{B}}_{\mathsf{BDH}}$ of $\mathcal{B}$ in the final game we first claim that

$$\Pr\left[\mathsf{G}_6 : \mathsf{Guessed} \wedge \hat{e}(\boldsymbol{P}, \boldsymbol{P})^{\boldsymbol{abc}} \in \mathsf{dom}(\boldsymbol{L_2})\right] \geq 2\ \mathbf{Adv}^{\mathcal{A}}_{\mathsf{IND\text{-}ID\text{-}CPA}}\ p(1-p)^{q_{\mathcal{EX}}} \tag{18}$$

In Appendix B we show that combining equations (3)–(14) and (16), one gets the inequality

$$\begin{aligned}&\Pr\left[\mathsf{G}_6 : \mathsf{Guessed} \wedge \hat{e}(\boldsymbol{P}, \boldsymbol{P})^{\boldsymbol{abc}} \in \mathsf{dom}(\boldsymbol{L_2})\right]\\ \geq\! &2\left|\Pr\left[\mathsf{G}_1 : d = d_{\mathcal{A}} \wedge \mathsf{Guessed}\right] - \frac{1}{2}\Pr\left[\mathsf{G}_1 : \mathsf{Guessed}\right]\right|\end{aligned}$$

Inequality (18) follows from Equations (1), (2) and the independence of the events $d = d_{\mathcal{A}}$ and $\mathsf{Guessed}$ in $\mathsf{G}_1$.

We conclude from Equations (15) and (17):

$$\begin{aligned}\mathbf{Adv}^{\mathcal{B}}_{\mathsf{BDH}} &= \Pr\left[\mathsf{G}_{\mathsf{BDH}} : z = \hat{e}(P, P)^{abc}\right]\\ &\geq q_{\mathcal{H}_2}^{-1}\Pr\left[\mathsf{G}_{\mathsf{BDH}} : \hat{e}(P, P)^{abc} \in \boldsymbol{L_2} \wedge |\boldsymbol{L_2}| \leq q_{\mathcal{H}_2}\right]\\ &= q_{\mathcal{H}_2}^{-1}\Pr\left[\mathsf{G}_7 : \hat{e}(\boldsymbol{P}, \boldsymbol{P})^{\boldsymbol{abc}} \in \mathsf{dom}(\boldsymbol{L_2})\right]\\ &= q_{\mathcal{H}_2}^{-1}\Pr\left[\mathsf{G}_6 : \hat{e}(\boldsymbol{P}, \boldsymbol{P})^{\boldsymbol{abc}} \in \mathsf{dom}(\boldsymbol{L_2})\right]\\ &\geq q_{\mathcal{H}_2}^{-1}\Pr\left[\mathsf{G}_6 : \mathsf{Guessed} \wedge \hat{e}(\boldsymbol{P}, \boldsymbol{P})^{\boldsymbol{abc}} \in \mathsf{dom}(\boldsymbol{L_2})\right]\\ &\geq 2\ q_{\mathcal{H}_2}^{-1}\ \mathbf{Adv}^{\mathcal{A}}_{\mathsf{IND\text{-}ID\text{-}CPA}}\ p(1-p)^{q_{\mathcal{EX}}}\end{aligned}$$

The bound in the statement of the theorem is obtained by taking

$$p = \frac{1}{1 + q_{\mathcal{EX}}}$$

which maximizes the factor $p(1-p)^{q_{\mathcal{EX}}}$. □

## 5 Conclusion and Future Work

Identity-based cryptography is an active field of public-key cryptography. We have demonstrated that the emerging approach promoted by verifiable security naturally applies to identity-based schemes by building a fully formal, independently verifiable proof of the BasicIdent scheme of Boneh and Franklin. Overall, the formal proof is about 3,500 lines of Coq, while our extension of CertiCrypt required about 1,800 lines. Our proof is more detailed and simpler than the original proof. Since we were not able to reproduce some of the arguments in [7] (e.g. Claim 1), we were compelled to find alternative arguments that resulted in a more compact proof.

A natural follow-up to the work presented here is to formally prove that the application of the Fujisaki-Okamoto transformation to the Boneh-Franklin BasicIdent scheme yields an IND-ID-CCA-secure scheme—this can be done generically for any IND-ID-CPA-secure scheme. Another interesting possibility is to weaken the ROM assumption in the security proof of BasicIdent: when instantiated using e.g. the Weil pairing, the proof assumes the hash function $\mathcal{H}_1$ behaves like a random oracle into an elliptic curve. We could instead assume just the existence of a random oracle into the field over which the elliptic curve is defined, and use it to build a function that is indifferentiable from a random oracle into the elliptic curve as shown by Brier et al. [10], thus recovering the same result under a weaker assumption.

Other research directions include developing mathematical libraries for pairings, such as the Weil pairing or the Tate pairing, and proving the security of other pairing-based systems, such as the Boneh-Boyen [6] and Waters [16] IBE schemes, or the Boneh-Lynn-Shacham signature scheme [9].

## References

1. Barthe, G., Grégoire, B., Heraud, S., Zanella Béguelin, S.: Formal certification of ElGamal encryption. A gentle introduction to CertiCrypt. In: 5th International workshop on Formal Aspects in Security and Trust, FAST 2008. Lecture Notes in Computer Science, vol. 5491, pp. 1–19. Springer, Heidelberg (2009)
2. Barthe, G., Grégoire, B., Lakhnech, Y., Zanella Béguelin, S.: Beyond provable security. Verifiable IND-CCA security of OAEP. In: Topics in Cryptology – CT-RSA 2011. Lecture Notes in Computer Science, vol. 6558, pp. 180–196. Springer, Heidelberg (2011)
3. Barthe, G., Grégoire, B., Zanella Béguelin, S.: Formal certification of code-based cryptographic proofs. In: 36th ACM SIGPLAN-SIGACT symposium on Principles of Programming Languages, POPL 2009. pp. 90–101. ACM, New York (2009)
4. Barthe, G., Grégoire, B., Zanella Béguelin, S.: Programming language techniques for cryptographic proofs. In: 1st International conference on Interactive Theorem Proving, ITP 2010. Lecture Notes in Computer Science, vol. 6172, pp. 115–130. Springer, Heidelberg (2010)
5. Bellare, M., Ristenpart, T.: Simulation without the artificial abort: Simplified proof and improved concrete security for Waters' IBE scheme. In: Advances in Cryptology – EUROCRYPT 2009. Lecture Notes in Computer Science, vol. 5479, pp. 407–424. Springer, Heidelberg (2009)

6. Boneh, D., Boyen, X.: Secure identity based encryption without random oracles. In: Advances in Cryptology — CRYPTO 2004. Lecture Notes in Computer Science, vol. 3152, pp. 197–206. Springer, Heidelberg (2004)
7. Boneh, D., Franklin, M.: Identity-based encryption from the Weil pairing. SIAM J. Comput. 32(3), 586–615 (2003)
8. Boneh, D., Gentry, C., Hamburg, M.: Space-efficient identity based encryption without pairings. In: 48th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2007. pp. 647–657. IEEE Computer Society, Los Alamitos (2007)
9. Boneh, D., Lynn, B., Shacham, H.: Short signatures from the Weil pairing. Journal of Cryptology 17, 297–319 (2004)
10. Brier, E., Coron, J.S., Icart, T., Madore, D., Randriam, H., Tibouchi, M.: Efficient indifferentiable hashing into ordinary elliptic curves. In: Advances in Cryptology – CRYPTO 2010. Lecture Notes in Computer Science, vol. 6223, pp. 237–254. Springer (2010)
11. Fujisaki, E., Okamoto, T.: How to enhance the security of public-key encryption at minimum cost. In: 2nd International workshop on Practice and Theory in Public Key Cryptography, PKC 1999. Lecture Notes in Computer Science, vol. 1560, pp. 634–634. Springer, Heidelberg (1999)
12. Galindo, D.: Boneh-Franklin identity based encryption revisited. In: 32nd International Colloquium on Automata, Languages and Programming, ICALP 2005,. Lecture Notes in Computer Science, vol. 3580, pp. 102–102. Springer, Heidelberg (2005)
13. Horwitz, J., Lynn, B.: Toward hierarchical identity-based encryption. In: Advances in Cryptology – EUROCRYPT 2002. vol. 2332, pp. 466–481. Springer, Heidelberg (2002)
14. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Advances in Cryptology – CRYPTO 1984. Lecture Notes in Computer Science, vol. 196, pp. 47–53. Springer, Heidelberg (1985)
15. The Coq development team: The Coq Proof Assistant Reference Manual Version 8.3. Online – `http://coq.inria.fr` (2010)
16. Waters, B.: Efficient identity-based encryption without random oracles. In: Advances in Cryptology — EUROCRYPT 2005. vol. 3494, pp. 557–557. Springer, Heidelberg (2005)
17. Zanella Béguelin, S., Grégoire, B., Barthe, G., Olmedo, F.: Formally certifying the security of digital signature schemes. In: 30th IEEE symposium on Security and Privacy, S&P 2009. pp. 237–250. IEEE Computer Society, Los Alamitos (2009)

## A Fundamental Lemma of Game-Playing

**Lemma 1 (Fundamental Lemma).** *Let $G_1, G_2$ be two games and let $A, B$, and $F$ be events. If* $\Pr[G_1 : A \wedge \neg F] = \Pr[G_2 : B \wedge \neg F]$*, then*

$$|\Pr[G_1 : A] - \Pr[G_2 : B]| \leq \max(\Pr[G_1 : F], \Pr[G_2 : F])$$

*Proof.*

$$\begin{aligned}
&|\Pr[G_1 : A] - \Pr[G_2 : B]| \\
&\quad = |\Pr[G_1 : A \wedge F] + \Pr[G_1 : A \wedge \neg F] - \Pr[G_2 : B \wedge F] - \Pr[G_2 : B \wedge \neg F]| \\
&\quad = |\Pr[G_1 : A \wedge F] - \Pr[G_2 : B \wedge F]| \\
&\quad \leq \max(\Pr[G_1 : A \wedge F], \Pr[G_2 : B \wedge F]) \\
&\quad \leq \max(\Pr[G_1 : F], \Pr[G_2 : F])
\end{aligned}$$

A syntactic criterion can be applied to discharge the hypothesis of the lemma for the case where $A = B$ and $F = \mathbf{bad}$. The hypothesis can be automatically established by inspecting the code of both games: it holds if their code differs only after program points setting the flag **bad** to true and **bad** is never reset to false afterwards. Note also that if both games terminate with probability 1, then $\Pr[G_1 : \mathbf{bad}] = \Pr[G_2 : \mathbf{bad}]$, and that if, for instance, only game $G_2$ terminates with probability 1, it must be the case that $\Pr[G_1 : \mathbf{bad}] \leq \Pr[G_2 : \mathbf{bad}]$.

## B Derived Equations

From Equations (13), (14) and (16) we can prove the following two inequalities:

$$\begin{aligned}
\Pr[\mathsf{G}_6 : d = d_{\mathcal{A}} \land \mathsf{Guessed}] &\geq \Pr[\mathsf{G}_6 : d = d_{\mathcal{A}} \land \mathsf{Guessed} \land \neg\mathsf{S}] \\
&= \Pr[\mathsf{G}_7 : d = d_{\mathcal{A}} \land \mathsf{Guessed} \land \neg\mathsf{S}] \\
&= \frac{1}{2}\Pr[\mathsf{G}_7 : \mathsf{Guessed} \land \neg\mathsf{S}] \\
&= \frac{1}{2}\Pr[\mathsf{G}_6 : \mathsf{Guessed} \land \neg\mathsf{S}] \\
&= \frac{1}{2}\Pr[\mathsf{G}_6 : \mathsf{Guessed}] - \frac{1}{2}\Pr[\mathsf{G}_6 : \mathsf{Guessed} \land \mathsf{S}]
\end{aligned}$$

and

$$\begin{aligned}
&\Pr[\mathsf{G}_6 : d = d_{\mathcal{A}} \land \mathsf{Guessed}] \\
&\quad = \Pr[\mathsf{G}_6 : d = d_{\mathcal{A}} \land \mathsf{Guessed} \land \mathsf{S}] + \Pr[\mathsf{G}_6 : d = d_{\mathcal{A}} \land \mathsf{Guessed} \land \neg\mathsf{S}] \\
&\quad \leq \Pr[\mathsf{G}_6 : \mathsf{Guessed} \land \mathsf{S}] + \Pr[\mathsf{G}_7 : d = d_{\mathcal{A}} \land \mathsf{Guessed} \land \neg\mathsf{S}] \\
&\quad = \Pr[\mathsf{G}_6 : \mathsf{Guessed} \land \mathsf{S}] + \frac{1}{2}\Pr[\mathsf{G}_7 : \mathsf{Guessed} \land \neg\mathsf{S}] \\
&\quad = \Pr[\mathsf{G}_6 : \mathsf{Guessed} \land \mathsf{S}] + \frac{1}{2}\Pr[\mathsf{G}_6 : \mathsf{Guessed} \land \neg\mathsf{S}] \\
&\quad = \frac{1}{2}\Pr[\mathsf{G}_6 : \mathsf{Guessed}] + \frac{1}{2}\Pr[\mathsf{G}_6 : \mathsf{Guessed} \land \mathsf{S}]
\end{aligned}$$

which together with Equations (3)-(12) imply

$$\Pr[\mathsf{G}_6 : \mathsf{Guessed} \land \mathsf{S}] \geq 2\left|\Pr[\mathsf{G}_1 : d = d_{\mathcal{A}} \land \mathsf{Guessed}] - \frac{1}{2}\Pr[\mathsf{G}_1 : \mathsf{Guessed}]\right|$$