Identity based and CCA-secure encryption

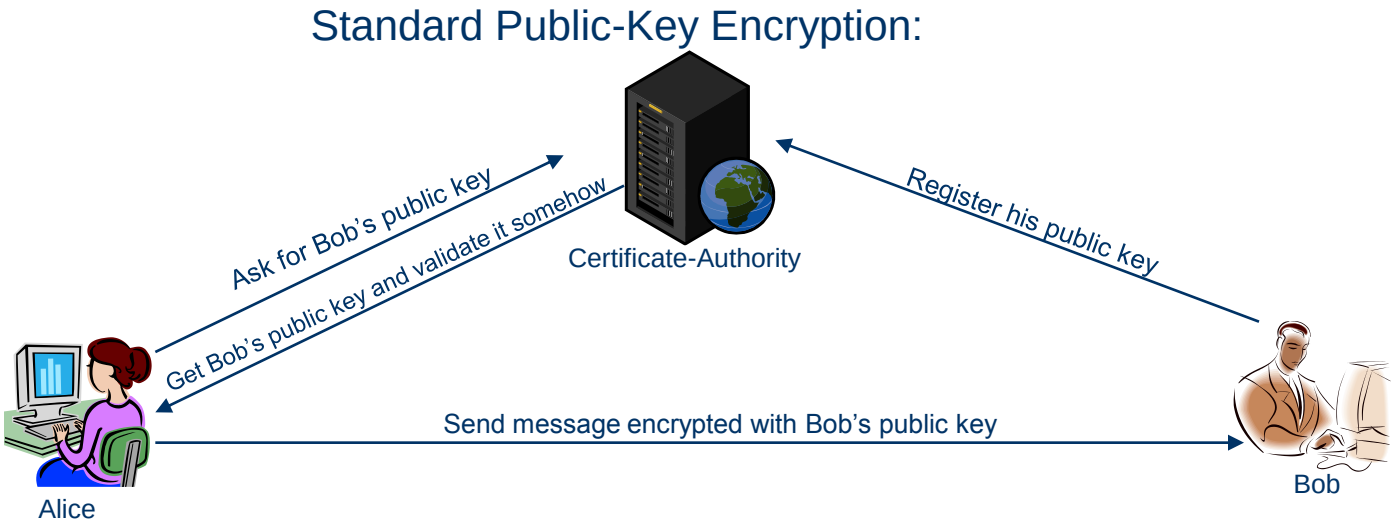
By Ilia Lotosh

Based on [BF'03], [BCHK '07]

Agenda

- Definition of ID-based encryption
- Possible applications
- CCA-secure encryption based on IBE
- Boneh-Franklin construction
- Possible implementations of BF constructions
- Boneh-Franklin IBE scheme

Definition of ID-based encryption

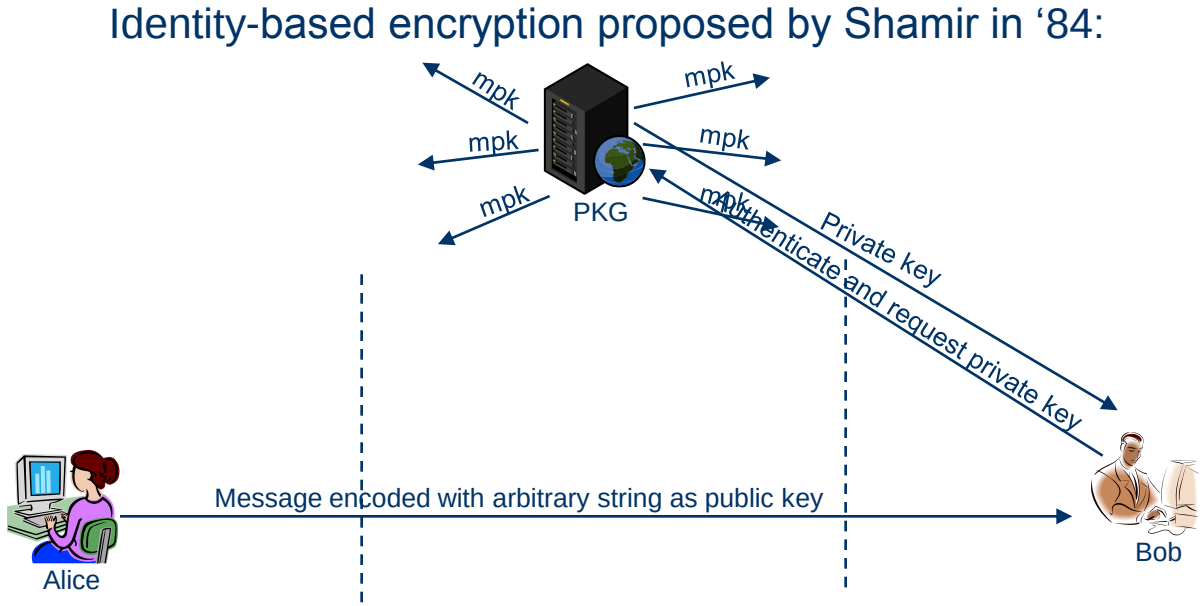


Problems with this approach:

- There is a need in central certificate-authority that will provide public key associated with Bob
- Alice needs a way to validate Bob's certificate to make sure message is being sent to Bob
- The system is tightly-coupled: messages can be sent only after Bob registers his public key, and Alice has to know about this before sending the message

Definition of ID-based encryption

Identity-based encryption proposed by Shamir in ‘84:



- Messages can be encoded with any public key
- There is a central authority that generates private keys for public keys
- Sender’s and receiver’s actions are independent and can be done in any order
- Authorization against PKG is done like with regular CA

Formal definition

IBE scheme consists of 4 randomized algorithms:

- **Setup:** Takes a security parameter *k* and returns **mpk** and **msk**. The parameters include a description of a finite message space *M*, and a description of a finite ciphertext space *C*.
- **Extract:** Takes as input **mpk**, **msk**, and an arbitrary $\mathrm{ID} \in \{0,1\}^*$, and returns a private key SK_{ID}. Here ID is an arbitrary string that will be used as a public key, and SK_{ID} is the corresponding private decryption key.
- **Encrypt:** Takes as input **mpk, ID** and $m \in M$. It returns a ciphertext $c \in C$.
- **Decrypt:** Takes as input **mpk,** $c \in C$, and a private key SK_{ID}. It returns $m \in M$

These algorithm must satisfy the standard consistency constraint, namely when SK_{ID} is the private key generated by algorithm **Extract** when it is given ID as the public key, then

$$\forall m \in M : \mathrm{Decrypt}(\mathrm{mpk}, c, \mathrm{SK_{ID}}) = m \text{ where } c = \mathrm{Encrypt}(\mathrm{mpk}, \mathrm{ID}, m)$$

Security notions – IND-ID-CPA

IBE scheme is semantically secure against an adaptive chosen plaintext attack if no poly-bounded adversary A has non-negligible advantage against Challenger in the following IND-ID-CPA game:

Setup: The challenger takes a security parameter k and runs the **Setup** algorithm. It gives the adversary the resulting **mpk**. It keeps the **msk** for itself.

Phase 1: The adversary issues queries $q_1, q_2, \ldots, q_m$ where q_i is extraction query:

- Extraction query $<ID_i>$. The challenger responds the private key d_i corresponding to the public key $<ID_i>$.

Challenge: The adversary outputs two equal length messages $m_0, m_1 \in M$ and an identity ID that did not appear in any extraction query. The challenger picks a random **b** and sets C = **Encrypt**(params, ID, M_b). It sends C as a challenge to the adversary

Phase 2: Adversary issues more queries as in phase 1 (but not about the challenge)

Guess: Adversary outputs a guess **b'** and with the game if **b=b'**

Security notions – selective IBE

Even weaker security notion can be obtained if we require adversary to choose ID he wants to compromise **before** seeing public system parameters generated by the challenger. Selective IBE IND-ID-CPA game will be the following:

ID Selection: The adversary chooses identity ID and passes it to the challenger

Setup: The challenger takes a security parameter k and runs the **Setup** algorithm. It gives the adversary the resulting **params**. It keeps the **master-key** for itself.

Phase 1: The adversary issues queries q_1, q_2,…,q_m where q_i is extraction query (not on ID):

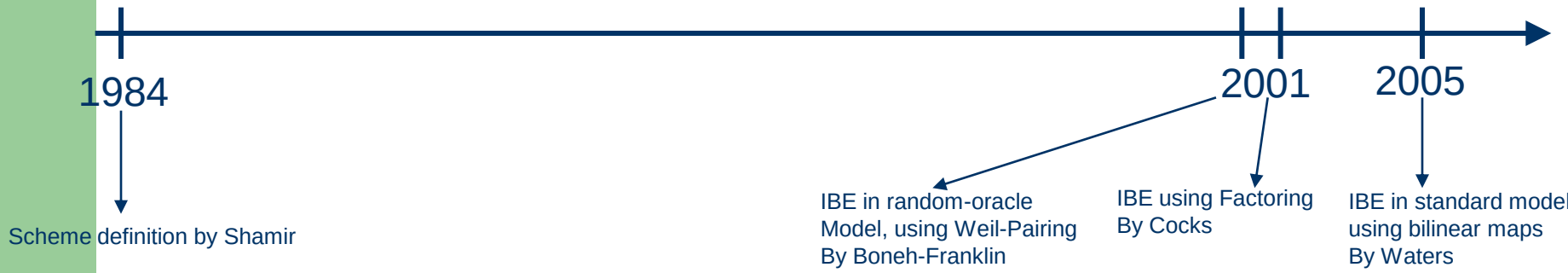
- Extraction query <ID_i>. The challenger responds the private key d_i corresponding to the public key <ID_i>.

Challenge: The adversary outputs two equal length messages $m_0, m_1 \in M$. The challenger picks a random **b** and sets C = **Encrypt**(params, ID, M_b). It sends C as a challenge to the adversary

Phase 2: Adversary issues more queries as in phase 1 (but not about the challenge)

Guess: Adversary outputs a guess **b'** and with the game if **b=b'**

A little bit of history



Possible applications

- First and trivial – to overcome PKE scheme problems we've discussed

Possible applications

In addition, ability to use an arbitrary string as public key allows following usages:

- Revocation of Public Keys
  - Keys of form bob@company.com || year
  - There is a corporate PKG which will give Bob private key valid for a year
- Managing user credentials
  - Keys of form bob@company.com ||year||clearance
  - Bob will be able to read messages only if he has appropriate clearance on the specified date
- Delegation to a laptop
  - Bob knows private master-key and creates temporary private keys to be used on his laptop during vacation
- Delegation of duties
  - Suppose Bob has several assistants for different subjects, then he can create different private keys for each subject, and having master key will allow him to read all the mail.

Possible applications

- Finally, identity based encryption can be used to construct CCA2-secure encryption.

We will see such construction now.

Recall – CCA2 security

CCA2 or adaptive chosen ciphertext attack security means that there is no poly-time adversary A that can win IND-CCA game with probability non-negligibly greater than half. The IND-CCA game is defined as follows:

Setup: The challenger takes a security parameter k and runs the **GEN** algorithm. It gives the adversary the resulting **public key**. It keeps the **private key** for itself.

Phase 1: The adversary issues queries q_1, q_2,…,q_m where q_i:

- Decryption query <C_i>. The challenger responds by decrypting C_i using the private key. It sends resulting plaintext to the adversary.

Challenge: The adversary outputs two equal length messages $m_0, m_1 \in M$ that did not appear in any decryption query. The challenger picks a random **b** and sets C = **Encrypt**(*private-key*, M_b). It sends C as a challenge to the adversary

Phase 2: Adversary issues more queries as in phase 1 (but not about the challenge)

Guess: Adversary outputs a guess **b'** and with the game if **b=b'**

Constructing CCA-secure scheme

We will construct now a public-key encryption scheme that is based on

- IBE scheme which is selective-ID secure against chosen-plaintext attacks
- One-time signature which is strongly unforgeable (which means that an adversary should not be able to forge a new signature even on a previously-signed message). Example of such scheme:

Lamport scheme:

Let f be a one-way-function. Then to sign a message of n bits, do:

The signing key is $2n$ random elements in the domain of f:

$X = \{x_{10}, x_{11}\}, \{x_{20}, x_{21}\}, \ldots \{x_{n0}, x_{n1}\}$

The public verification key is the images of X under f :

$Y = \{y_{10}, y_{11}\}, \{y_{20}, y_{21}\}, \ldots \{y_{n0}, y_{n1}\}, \quad \text{where } \forall i, j : y_{i,j} = f(x_{i,j})$

To sing a message $m = m_0 m_1 \ldots m_n$ output the n values $x_{1m_1}, x_{2m_2}, \ldots, x_{nm_n}$

To verify a signature $x_{1m_1}, x_{2m_2}, \ldots, x_{nm_n}$ on message m, with public key Y,

verify that for each $i, y_{im_i} = f(x_{im_i})$

Constructing CCA-secure scheme

The construction is by Canetti, Halevi and Katz, and goes as following:

Let $\prod'$=(Setup,Extract,Encode',Decode') be an IBE scheme and Sig=(G, Sign, Verify) be a one-time signature scheme.
Our public encryption scheme $\prod$=(Gen,Encode,Decode) will work as follows:

Gen on 1^k

Runs Setup(1^k) to obtain (PK, msk). The public key is PK and the secret key is msk.

Encryption

To encrypt message m using public key PK, the sender first runs G(1^k) to obtain verification key vk and signing key sk.
The sender then computes $c \leftarrow Encode'(PK, vk, m)$ (i.e. sender uses vk as an identity) and $\sigma \leftarrow Sign(sk, c)$.
The final ciphertext is (vk, c, σ).

Decryption

To decrypt ciphertext (vk, c, σ) using secret key msk, the receiver first checks whether Verify$(vk, c, \sigma) \stackrel{?}{=} 1$.
If not, the receiver simply outputs $\perp$. Otherwise, the receiver computes $SK_{vk} \leftarrow Extract(msk, vk)$ and outputs $Decode'(SK_{vk}, vk, c) = m$.
It is clear that this scheme is indeed a correct public-key encryption scheme.

Proof of CCA2-security

Intuition for the proof (informal):

Let (vk^*, c^*, σ^*) be the challenge ciphertext. It is clear that without any decryption oracle queries, the plaintext corresponding to the ciphertext remains "hidden" to the adversary; this is so because c^* is output by Π' which is CPA-secure (and the additional components of the ciphertext provide no additional help).

Decryption oracle queries can't further help the adversary. On one hand, if the adversary submits to the oracle a ciphertext (vk', c', σ') that is different from the challenge ciphertext but with $vk' = vk^*$ then the decryption oracle will reply with $\perp$ since the adversary is unable to forge new, valid signatures with respect to vk. On the other hand, if $vk' \neq vk^*$ then the decryption query will not help the adversary since the eventual decryption using Decrypt' will be done with respect to a different "identity" vk'.

Proof of CCA2-security

Formal proof:

Assume we are given a poly-time adversary A attacking $\prod$ in an adaptive chosen-ciphertext attack. Say a ciphertext (vk, c, σ) is valid if $Verify(vk, c, \sigma) = 1$. Let (vk^*, c^*, σ^*) denote the challenge ciphertext received by A during a particular run of the game, and let Forge denote the event that A submits a valid ciphertext (vk^*, c, σ). We prove the following claims:

Claim 1: $\Pr_{A,\Pi}^{PKE}[Forge]$ is negligible.

Claim 2: $|\Pr_{A,\Pi}^{PKE}[Success \wedge \overline{Forge}] + \frac{1}{2}\Pr_{A,\Pi}^{PKE}[Forge] - \frac{1}{2}|$ is negligible.

Now from these two claims we get:

$$\left|\Pr_{A,\Pi}^{PKE}[Success] - \frac{1}{2}\right|$$

$$\leq |\Pr_{A,\Pi}^{PKE}[Success \wedge Forge] - \frac{1}{2}\Pr_{A,\Pi}^{PKE}[Forge]| + |\Pr_{A,\Pi}^{PKE}[Success \wedge \overline{Forge}] + \frac{1}{2}\Pr_{A,\Pi}^{PKE}[Forge] - \frac{1}{2}|$$

$$\leq \Pr_{A,\Pi}^{PKE}[Forge] + |\Pr_{A,\Pi}^{PKE}[Success \wedge \overline{Forge}] + \frac{1}{2}\Pr_{A,\Pi}^{PKE}[Forge] - \frac{1}{2}|$$

which is negligible given the stated claims.

Proof of CCA2-security

Proof of claim 1:

We construct a poly-time forger F who forges a signature with respect to signature scheme Sig with probability exactly $\Pr^{PKE}_{A,\Pi}[Forge]$. Security of Sig implies the claim.

F is defined as follows: given input 1^k and verification key vk^*, F first runs Setup(1^k) to obtain (PK, msk), and then runs $A(1^k, PK)$. Note that F can answer any decryption queries of A. If A happens to submit a valid ciphertext (vk^*, c, σ) to its decryption oracle before requesting the challenge ciphertext, then F simple outputs the forgery (c, σ) and stops. Otherwise, when A outputs messages m_0, m_1, forger F proceeds as follows: chooses a random bit b, computes $c^* \leftarrow Encrypt'(vk^*, m_b)$ and obtains from its signing oracle a signature σ^* on the message c^*. Finally, F hands (vk^*, c^*, σ^*) to A. If A submits a valid ciphertext (vk^*, c, σ) to its decryption oracle, note that we must have $(c, \sigma) \neq (c^*, \sigma^*)$. In this case, F simply outputs (c, σ) as its forgery. It is easy to see that F's success probability is exactly $\Pr^{PKE}_{A,\Pi}[Forge]$.

Proof of CCA2-security

Proof of claim 2:

We use A to construct a poly-time adversary A' which attacks the IBE scheme $\prod'$ in selective IND-ID-CPA game.

Define adversary A' as follows:

1. $A'(1^k)$ runs $G(1^k)$ to generate (vk^*, sk^*), and outputs the "target" identity $ID^* = vk^*$
2. A' is given a master public key PK. Adversary A', in turn, runs $A(1^k, PK)$.
3. When A makes a decryption oracle query $Decode(vk, c, \sigma)$, adversary A' proceeds as follows:
   (a) If $vk = vk^*$ then A' checks whether $Verify(vk^*, c, \sigma) = 1$. If so, A' aborts and outputs a random bit. Otherwise, it simply responds with $\perp$.
   (b) If $vk \neq vk^*$ and $Verify(vk, c, \sigma) = 0$ then A' responds with $\perp$.
   (c) If $vk \neq vk^*$ and $Verify(vk, c, \sigma) = 1$ then A' makes the oracle query $Extract(msk, vk)$ to obtain SK_{vk}. It then computes $m \leftarrow Decode(SK_{vk}, vk, c)$ and responds with m.
4. At some point A outputs two messages m_0, m_1. These messages are output by A' as well. In return, A' is given a challenge ciphertext c^*; adversary A' then computes $\sigma^* \leftarrow Sign(sk^*, c^*)$ and returns (vk^*, c^*, σ^*) to A.
5. A may continue to make decryption queries and these are answered as before
6. Finally, A outputs a guess b'; this same guess is output by A'.

Proof of CCA2-security

Proof of claim 2, continued:

Note that A' represents a legal adversarial strategy for attacking Π'; in particular, A' never requests the secret key corresponding to the "target" identity vk^*. Furthermore, A' provides a perfect simulation for A until event Forge occurs (in such event A' outputs a random bit). And thus:

$$\left|\Pr^{IBE}_{A',\Pi'}[Success]-\frac{1}{2}\right| = \left|\Pr^{PKE}_{A,\Pi}[Success \wedge \overline{Forge}]+\frac{1}{2}\Pr^{PKE}_{A,\Pi}[Success \wedge Forge]-\frac{1}{2}\right|$$

And the left side of the above is negligible by the assumed security of $\prod'$.

■

Boneh-Franklin construction

Bilinear maps

Let G_1 and G_2 be two groups of order q. We say that a map $\hat{e}: G_1 \times G_1 \rightarrow G_2$ between these two groups is bilinear if it satisfies the following properties:

1. Bilinear: for all $P, Q \in G_1$ and $a, b \in$ □ $\hat{e}(P^a, Q^b) = \hat{e}(P, Q)^{ab}$
2. Non-degenerate: The map does not send all pairs in $G_1 \times G_1$ to the identity in G_2.
3. Computable: There is an efficient algorithm to compute $\hat{e}(P, Q)$ for any $P, Q \in G_1$.

A bilinear map satisfying the three properties above is said to be an *admissible* bilinear map. The existence of such a map has two direct implications to these groups, we will see them next.

Bilinear maps – MOV reduction

- Named after Menezes, Okamoto and Vanstone
- Shows that the discrete log problem in G_1 is no harder than the discrete log problem in G_2.

Let $P, Q \in G_1$, where both P, Q have order q. We wish to find $\alpha \in \mathbb{Z}$ such that $Q = P^{\alpha}$.

Let $g = \hat{e}(P, P)$ and $h = \hat{e}(Q, P) \overset{\text{bilinearity}}{\Rightarrow} h = g^{\alpha}$

By non-degeneracy of $\hat{e}$ both g, h have order q in G_2

Hence, we reduced the discrete log problem in G_1 to a discrete log problem in G_2.

Bilinear maps – DDH is easy

The Decision Diffie-Hellman problem in G_1 is to distinguish between the distributions $(P, P^a, P^b, P^{ab})$and (P, P^a, P^b, P^c) where a,b,c are random in $Z_q\backslash\{0\}$ and P is random in $G_1\backslash\{0\}$:

$$\text{Given } P, P^a, P^b, P^c \in G_1 \text{ we have:}$$

$$c = ab \bmod q \Leftrightarrow \hat{e}(P, P^c) = \hat{e}(P^a, P^b).$$

Bilinear Diffie-Hellman Problem

BDH problem:

Let G_1, G_2 be two groups of prime order q. Let $\hat{e}: G_1 \times G_1 \to G_2$ be an admissible bilinear map and let P be a generator of G_1. The BDH problem in $(G_1, G_2, \hat{e})$ is:

Given (P, P^a, P^b, P^c) for some $a, b, c \in \mathbb{Z}_q^*$ compute $W = \hat{e}(P,P)^{abc} \in G_2$. An algorithm A has advantage ε in solving BDH in $(G_1, G_2, \hat{e})$ if $\Pr[A(P, P^a, P^b, P^c) = \hat{e}(P,P)^{abc}] \geq \varepsilon$

BDH parameter Generator:

A randomized algorithm G is a BDH parameter generator if:

1) G takes security parameter $k \in \mathbb{Z}$

2) G runs in time polynomial in k

3) G outputs prime number q, description of two groups G_1, G_2 of order q and the description of admissible bilinear map $\hat{e}: G_1 \times G_1 \to G_2$

BDH assumption:

Let G be a BDH parameter generator. We say that an algorithm A has advantage $\varepsilon(k)$ in solving BDH problem for G if:

$$Adv_{G,A}(k) = \Pr[A(q, G_1, G_2, \hat{e}, P, P^a, P^b, P^c) = \hat{e}(P,P)^{abc} \mid \begin{matrix}(q, G_1, G_2, \hat{e}) \leftarrow G(1^k), \\ P \leftarrow G_1^*, a, b, c \leftarrow \mathbb{Z}_q^*\end{matrix}] \geq \varepsilon(k)$$

We say that G satisfies the BDH assumption if for any randomized polynomial time (in k) algorithm A we have that $Adv_{G,A}(k)$ is a negliglible function.

Possible construction for generator satisfying BDH assumption

Elliptic curves

A curve defined by the equation $y^2 = x^3 + ax + b$ over some field. We will talk about elliptic curve E defined by the equation $y^2 = x^3 + 1$ over field F_p where p is a prime satisfying $p = 2 \bmod 3$. Let $E(\mathrm{F}_{p^r})$ denote the group of points on E defined over F_{p^r}.

Some facts from number theory regarding *E*

Fact 1: $x^3 + 1$ is a permutation on $\mathrm{F}_p \Rightarrow E(\mathrm{F}_p)$ contains $p + 1$ points.

Let O denote a point at infinity, let $P \in E(\mathrm{F}_p)$ be a point of order q and let G_1 be the subgroup generated by P.

Fact 2: For any $y_0 \in \mathrm{F}_p$ there is a unique point (x_0, y_0) on $E(\mathrm{F}_p)$, namely $x_0 = (y_0^2 - 1)^{1/3} \in \mathrm{F}_p$. Hence, if (x, y) is a random non-zero point on $E(\mathrm{F}_p)$ then y is uniform in F_p.

Fact 3: Let $1 \neq \zeta \in \mathrm{F}_{p^2}$ be a solution of $x^3 - 1 = 0$ in F_{p^2}. Then the map $\phi(x, y) = (\zeta x, y)$ is an automorphism of the group of points on E. Note that for any point $Q = (x, y) \in E(\mathrm{F}_p)$ we have that $\phi(Q) \in E(\mathrm{F}_{p^2})$, but $\phi(Q) \notin E(\mathrm{F}_p)$. Hence, $\mathrm{Q} \in E(\mathrm{F}_p)$ is linearly independent of $\phi(Q) \in E(\mathrm{F}_{p^2})$.

Fact 4: Since the points $P \in \mathrm{G}_1$ and $\phi(P)$ are linearly independent they generate a group isomorphic to $\mathbb{Z}_q \times \mathbb{Z}_q$.

We denote this group of points by $E[q]$.

Possible construction for generator satisfying BDH assumption

Some basic concepts

In the following we let P and Q be arbitrary points in $E(\mathrm{F}_{p^2})$:

Divisors A divisor is a formal sum of points on the curve $E(\mathrm{F}_{p^2})$. We write divisors as $\mathrm{A} = \sum_P a_P(P)$ where $a_P \in$ □ and $P \in E(\mathrm{F}_{p^2})$. We will only consider divisors $\mathrm{A} = \sum_P a_P(P)$ where $\sum_P a_P = 0$.

Functions A function f on the curve $E(\mathrm{F}_{p^2})$ can be viewed as a rational function $f(x, y) \in E(\mathrm{F}_{p^2})$. For any point $P = (x, y) \in E(\mathrm{F}_{p^2})$ we define $f(P) = f(x, y)$.

Divisors of functions Let f be a function on the curve $E(\mathrm{F}_{p^2})$. We define its divisor, denoted by (f), as $(f) = \sum_P ord_P(f) \cdot (P)$. Here $ord_P(f)$ is the order of the zero that f has at point P.

Principal divisors Let A be a divisor. If there exists a function f such that $(f) = \mathrm{A}$ then we say that A is a principal divisor. We know that a divisor $\mathrm{A} = \sum_P a_P(P)$ is principal if and only if $\sum_P a_P = 0$ and $\sum_P a_P P = O$. Furthemore, given a principal divisor A there exists a unique function f such that $(f) = \mathrm{A}$.

Equivalence of divisors We say that two divisors A,B are equivalent is their difference A-B is a principal divisor. We know that any divisor $\mathrm{A} = \sum_P a_P(P)$ (with $\sum_P a_P = 0$) is equivalent to a divisor of the form $\mathrm{A}' = (Q) - (O)$.

Notation Given a function f and a divisor $\mathrm{A} = \sum_P a_P(P)$ we define $f(\mathrm{A})$ as $f(\mathrm{A}) = \prod_P f(P)^{a_P}$.

Possible construction for generator satisfying BDH assumption

Weil pairing

We will define now the Weil pairing of two points $P, Q \in E[n]$. Let A_P be some divisor equivalent to the divisor $(P)-(O)$. We know that nA_P is a principal divisor (it is equivalent to $n(P)-n(O)$). Hence, there exists a function f_p such that $(f_P) = nA_P$. Define A_Q and f_Q analogously.

The Weil pairing of P and Q is defined as:

$$e(P,Q) = \frac{f_P(A_Q)}{f_Q(A_P)}$$

It's clear that this map is bilinear, since: $e(P_1+P_2,Q) = e(P_1,Q)\cdot e(P_2,Q)$ and $e(P,Q_1+Q_2) = e(P,Q_1)\cdot e(P,Q_2)$

But, it's degenerate, since for all $P \in E[n]$ we have $e(P,P) = 1$.

Possible construction for generator satisfying BDH assumption

Modified Weil pairing

To overcome the problem of degeneracy we modify Weil pairing.

Modified Weil pairing $\hat{e}: G_1 \times G_1 \to G_2$ is defined as follows:

$$\hat{e}(P,Q) = e(P,\phi(Q))$$

Recall:

Let $1 \neq \zeta \in F_{p^2}$ be a solution of $x^3 - 1 = 0$ in F_{p^2}. Then the map $\phi(x,y) = (\zeta x, y)$ is an automorphism of the group of points on E. Note that for any point $Q = (x,y) \in E(F_p)$ we have that $\phi(Q) \in E(F_{p^2})$, but $\phi(Q) \notin E(F_p)$. Hence, $Q \in E(F_p)$ is linearly independent of $\phi(Q) \in E(F_{p^2})$.

G_1 – Subgroup of points in $E(F_p)$ generated by the point P of order q

G_2 – Subgroup of $F_{p^2}^*$ of order q

Possible construction for generator satisfying BDH assumption

Modified Weil pairing

To overcome the problem of degeneracy we modify Weil pairing.

Modified Weil pairing $\hat{e}: G_1 \times G_1 \to G_2$ is defined as follows:

$$\hat{e}(P,Q) = e(P,\phi(Q))$$

Modified Weil pairing satisfy the following properties:

1. Bilinear (follows from bilinearity of Weil pairing)

2. Non-degenerate: Obvious

3. Computable: There is an efficient algorithm to compute the value of the map

Generator built basing on this map is believed to satisfy BDH assumption asymptotically. However, there is still the question of what values of p and q can be used in practice to make the BDH problem sufficiently hard.

Boneh-Franklin IBE scheme

Let G be some BDH parameter generator (for example the one we saw before).

Setup: Given a security parameter $k \in \mathbb{Z}^+$, the algorithm works as follows:

Step 1: Run G on input k to generate a prime q, two groups G_1, G_2 of order q, and an admissible map $\hat{e}: G_1 \times G_1 \to G_2$. Choose a random generator $P \in G_1$.

Step 2: Pick a random $s \in \mathbb{Z}_q^*$ and set $P_{pub} = P^s$.

Step 3: Choose a cryptographic hash function $H_1: \{0,1\}^* \to G_1^*$. Choose a cryptographic hash function $H_2: G_2 \to \{0,1\}^n$ for some n.

The message space is $M = \{0,1\}^n$. The ciphertext space is $C = G_1^* \times \{0,1\}^n$. The system parameters are $\mathbf{mpk} = (q, G_1, G_2, \hat{e}, n, P, P_{pub}, H_1, H_2)$. The **msk** is $s \in \mathbb{Z}_q^*$.

Boneh-Franklin IBE scheme

Extract: For a given string $ID \in \{0,1\}^*$ the algorithm does:

1) Computes $Q_{ID} = H_1(ID) \in \mathrm{G}_1^*$

2) Sets the private key d_{ID} to be $d_{ID} = Q_{ID}^s$, where s is the master key.

Boneh-Franklin IBE scheme

Encrypt: To encrypt $m \in M$ under the public key ID do the following:

(1) compute $Q_{ID} = H_1(ID) \in \mathrm{G}_1^*$

(2) choose a random $r \in \mathbb{Z}_q^*$

(3) set the ciphertext to be

$C = (P^r, m \oplus H_2(g_{ID}^r))$ where $g_{ID} = \hat{e}(Q_{ID}, P_{pub}) \in \mathrm{G}_2^*$

Boneh-Franklin IBE scheme

Decrypt: Let $c = (U, V) \in C$ be a ciphertext encypted using the public key *ID*.

To decrypt c using the private key $d_{ID} \in \mathrm{G}_1^*$ compute:

$$V \oplus H_2(\hat{e}(d_{ID}, U)) = m$$

Boneh-Franklin IBE scheme

Consistency:

1. During encryption *m* is bitwise xored with the hash of: g_{ID}^{r}
2. During decryption *V* is bitwise xored with the hash of: $\hat{e}(d_{ID}, U)$

These masks used during encryption and decryption are the same since:

$$\hat{e}(d_{ID}, U) = \hat{e}(Q_{ID}^{s}, P^{r}) = \hat{e}(Q_{ID}, P)^{sr} = \hat{e}(Q_{ID}, P_{pub})^{r} = g_{ID}^{r}$$

BF IBE scheme security

Selective IND-ID-CPA security under standard model

We will prove now that presented scheme is selective IND-ID-CPA secure in the standard model.

Reminder: In selective IND-ID-CPA game the adversary first tells challenger which ID he wants to be challenged on, then he receives public setup and is allowed to issue key extraction queries

BF IBE scheme security

Selective IND-ID-CPA security under standard model

Decisional BDH: To distinguish between $(P, P^{\alpha}, P^{\beta}, P^{\gamma}, P^{\alpha\beta\gamma})$ and $(P, P^{a}, P^{\beta}, P^{\gamma}, P^{\mu})$ which is equal to distinguish between $(P, P^{\alpha}, P^{\beta}, P^{\gamma}, \hat{e}(P)^{\alpha\beta\gamma})$ and $(P, P^{\alpha}, P^{\beta}, P^{\gamma}, r)$ for random $r \in \mathrm{G}_2$

Theorem: If there exists a poly-time adversary A that gains advantage ε in selective IND-ID-CPA game, then there exists a poly-time adversary B that solves Decisional BDH with probability ε.

We are going to use a family of hash functions $\mathrm{H} = \{H_k\}$ that satisfy the following properties:

1. $H_k : \{0,1\}^* \rightarrow \mathrm{G}_1^*$
2. $\forall x \in \{0,1\}^*, y \in \mathrm{G}_1 \exists! k$ s.t. $H_k(x) = y$
3. Such k is easy to find

BF IBE scheme security

Selective IND-ID-CPA security under standard model

Algorithm B:

1. Gets $q, \mathrm{G}_1, \mathrm{G}_2, \hat{e}$ and $(P, Q = P^{\alpha}, U = P^{\beta}, R = P^{\gamma}, r \in \mathrm{G}_2)$

   B has to answer 1 if $\hat{e}(P,P)^{\alpha\beta\gamma} = r$ and 0 otherwise
2. B starts to execute A and on a first step receives ID
3. B chooses random $s \in \mathbb{Z}_q^*$, and finds s^{-1}

   B chooses hash function $H_1 \in \mathrm{H}$ such that $H_1(ID) = P^{s^{-1}\beta}$, and another hash function $H_2 : \mathrm{G}_2 \to \{0,1\}^n$ for some n without any restriction.
4. B provides A with public setup $(q, \mathrm{G}_1, \mathrm{G}_2, \hat{e}, n, Q, Q^s, H_1, H_2)$

   So master-key is s and public key is $P^{s\alpha}$
5. B answers A's extraction queries in a standard way
6. When A ready for a challenge it gives B two messages m_0, m_1

   B chooses bit b at random and gives A $C = (R, m_b \oplus H_2(r))$
7. B answers 1 if A was correct

BF IBE scheme security

Selective IND-ID-CPA security under standard model

Analysis of algorithm B:

- Algorithm B runs the same time as A
- In the last stage:
  - If $\hat{e}(P,P)^{\alpha\beta\gamma} = r$ then $H_2(r) = H_2(g_{ID}^{\gamma})$, since $g_{ID} = \hat{e}(P^{s^{-1}\beta}, P^{s\alpha})$ and thus $(P^{\gamma}, H_2(r) \oplus m_b)$ is a valid encryption of m_b, and hence $\Pr[A \text{ answers correctly}] \geq \varepsilon$
  - Otherwise, $H_2(r)$ is a random uniform string and thus $H_2(r) \oplus m_b$ is a random uniform string, and hence $\Pr[A \text{ answers correctly}] \leq \frac{1}{2}$
  - Thus B answers correctly with probability at least ε

BF IBE scheme security

IND-ID-CPA security under random oracle model

Now we will see how to show that BF IBE scheme is IND-ID-CPA secure in random oracle model.

Reminder: In random oracle model cryptographic hash functions are replaced by truly random functions. Our benefit in this model is that we can build our random oracle on the fly according to adversary actions

BF IBE scheme security

IND-ID-CPA security under random oracle model

First, we will show a reduction from BF IBE scheme to the following public-key scheme, called BasicPub, this scheme is defined by the following algorithms:

keygen: Given a security parameter $k \in \mathbb{Z}^+$, the algorithm works as follows:

Step 1: Generate two prime order groups G_1, G_2 and a bilinear map $\hat{e}: G_1 \times G_1 \to G_2$. Let q be the order of G_1, G_2. Choose a random generator $P \in G_1$.

Step 2: Pick a random $k \in \mathbb{Z}_q^*$ and set $P_{pub} = P^s$. Pick a random $Q_{id} \in G_1^*$

Step 3: Choose a cryptographic hash function $H_2 : G_2 \to \{0,1\}^n$ for some n

Step 4: The public key is $(q, G_1, G_2, \hat{e}, n, P, P_{pub}, Q_{id}, H_2)$ the private key is $d_{id} = Q_{id}^s$

encrypt: To encrypt $m \in \{0,1\}^n$ choose a random $r \in \mathbb{Z}_q^*$ and set the cipthertext to be:

$C = (P^r, m \oplus H_2(g^r))$ where $g = \hat{e}(Q_{id}, P_{pub}) \in G_2^*$

decrypt: Let $C = (U, V)$ be a ciphertext created using the public key above. To decrypt C using a private key d_{id} compute: $V \oplus H_2(\hat{e}(d_{id}, U)) = m$

BF IBE scheme security

IND-ID-CPA security under random oracle model

Now, if there is a polytime adversary A that wins IND-ID-CPA game against BF IBE scheme with non-negligible, then there is a polytime adversary B that wins IND-CPA game against BasicPub, with non-negligible probability.

Proof:

Algorithm B is given public key $K_{pub} = (q, \mathrm{G}_1, \mathrm{G}_2, \hat{e}, n, P, P_{pub}, Q_{id}, H_2)$.

setup

B gives A system parameters $(q, \mathrm{G}_1, \mathrm{G}_2, \hat{e}, n, P, P_{pub}, H_1, H_2)$. H_1 is a random oracle controlled by B in the following way:

B maintains a list of tuples (ID_j, Q_j, b_j, c_j) we call it H_1^{list}, the list is initially empty. When asked on ID_i :

1. If the query ID_i already appears on the H_1^{list} in a tuple (ID_i, Q_i, b_i, c_i) then B responds with $H_1(ID_i) = Q_i$
2. Otherwise B generates a random $coin \in \{0,1\}$ so that $\Pr[coin = 0] = \delta$
3. B picks a random $b \in \mathbb{Z}_q^*$. If $coin = 0$ $Q_i = P^b$, otherwise $Q_i = Q_{id}^b$
4. B adds the tuple (ID_i, Q_i, b_i, c_i) to the list and responds with Q_i

BF IBE scheme security

IND-ID-CPA security under random oracle model

key-extraction

Let ID_i be a private key extraction query issued by algorithm *A*. Algorithm *B* do the following:

1. Obtain Q_i using the previous algorithm, let (ID_i, Q_i, b_i, c_i) be corresponding tuple
2. If $coin_i = 1$ then B reports failure and terminates.
3. Otherwise we know that $Q_i = P^{b_i}$. Define $d_i = P_{pub}^{b_i}$. Observe that $d_i = P^{b_i s} = Q_i^s$, and therefore d_i is the private key associated to the public key ID_i. Give d_i to algorithm *A*.

BF IBE scheme security

IND-ID-CPA security under random oracle model

Challenge

Once algorithm A decides to begin the challenge, it outputs a public key ID_{ch} and two messages, m_0, m_1

1. Algorithm B gives its challenger the messages m_0, m_1. The challenger responds with ciphertext $C = (U, V)$ such that C is the encryption of m_c for a random $c \in \{0,1\}$.
2. Next, B obtains tuple corresponding to ID_{ch}: $(ID_{ch}, Q, b, coin)$. If $coin = 0$ then B reports failure and terminates.
3. We know $coin = 1$ and therefore $Q = Q_{ID}^s$. Recall that when $C = (U, V)$ we have $U \in G_1^*$

   Set $C' = (U^{b^{-1}}, V)$, where b^{-1} is the inverse of $b \bmod q$. Algorithm B responds to A with C'. Note:

   $$\hat{e}(U^{b^{-1}}, d_{ch}) = \hat{e}(U^{b^{-1}}, sQ) = \hat{e}(U, Q^{sb^{-1}}) = \hat{e}(U, Q_{ID}^s) = \hat{e}(U, d_{ID})$$

   Hence, the BF IBE decryption of C' using d_{ch} is the same as the BasicPub decryption of C using d_{ID}

BF IBE scheme security

IND-ID-CPA security under random oracle model

After the challenge being set algorithm A may continue issuing key extraction queries, algorithm B will respond as before.
Eventually, algorithm A will output its answer – b', algorithm B will output the same answer.

If algorithm B does not abort, A's view is identical to its view in the real attack, and thus if A answers correctly – so does B.

B does not abort with non-negligible probability. And thus B wins IND-CPA game against BasicPub scheme with non-negligible probability.

■

BF IBE scheme security

IND-ID-CPA security under random oracle model

As a final step we will show a reduction from BasicPub IND-CPA game to BDH problem, and from this we will have that if BDH is hard then BF IBE is IND-ID-CPA secure.

To show this, let's assume by contradiction that there is a poly-time algorithm A that wins IND-CPA game against BasicPub with non-negligible probability. We will show an algorithm B that solves BDH problem with non-negligible probability.

Algorithm B is given as input the BDH parameters $(q, \mathrm{G}_1, \mathrm{G}_2, \hat{e})$ and a random instance $(P, P^a, P^b, P^c) = (P, P_1, P_2, P_3)$of the BDH problem for these parameters.

Let $D = \hat{e}(P, P)^{abc} \in \mathrm{G}_2$ be the solution to this BDH problem.

BF IBE scheme security

IND-ID-CPA security under random oracle model

setup

Algorithm B creates the BasicPub public key $K_{pub} = (q, \mathrm{G}_1, \mathrm{G}_2, \hat{e}, n, P, P_{pub}, Q_{ID}, H_2)$ by setting $P_{pub} = P_1$ and $Q_{ID} = P_2$. Here H_2 is a random oracle controlled by B as described below. Algorithm B gives A K_{pub}.

The (unknown) private key associated to K_{pub} is $d_{ID} = Q_{ID}^s = P^{ab}$.

At any time algorithm A may issue queries to the random oracle H_2, to respond to them:

B maintains a list of tuples (X_j, H_j) we call it H_2^{list}, the list is initially empty. When asked on X_i :

1. If the query X_i already appears on the H_2^{list} in a tuple (X_i, H_i) then B responds with $H_2(X_i) = H_i$
2. Otherwise B generates just picks a random string $H_i \in \{0,1\}^n$ and adds the tuple (X_i, H_i) to the H_2^{list}. It responds to A with H_i

BF IBE scheme security

IND-ID-CPA security under random oracle model

challenge

Algorithm A outputs two messages m_0, m_1. Algorithms B picks a random string $R \in \{0,1\}^n$ and defines c to be the ciphertext $c = (P_3, R)$. Algorithm B gives c as the challenge to A. Observe that, by definition, the decryption of C is $R \oplus H_2(\hat{e}(P_3, d_{ID})) = R \oplus H_2(D)$.

BF IBE scheme security

IND-ID-CPA security under random oracle model

guess

Algorithm A outputs its guess $c' \in \{0,1\}$. At this point B picks a random tuple (X_j, H_j) from the H_2^{list} and outputs X_j as the solution.

Proof of correctness:

B simulates a real attack environment for A, and thus we expect A to be correct with non-negligible probability (if given correct encryption in last stage). And thus probability of A asking for H_2(D) is not negligible (since otherwise the decryption of C is independent of A's view and thus A can't answer correctly with probability greater than half). So we have D in our list with probability ε, and since it's length is polynomial, picking entry at random will provide correct answer with non-negligible probability.

■

References

- **Identity based encryption from the Weil pairing**
  D. Boneh and M. Franklin
  SIAM J. of Computing, Vol. 32, No. 3, pp. 586-615, 2003.
- **Chosen-Ciphertext Security from Identity-Based Encryption.**
  D. Boneh, R. Canetti, S. Halevi, and J. Katz.
  SIAM J. Comput., 36(5): 1301-1328 (2007)