Identity based encryption (IBE)

Jiageng Chen

Boneh, D., & Franklin, M. (2001, August). Identity-based encryption from the Weil pairing. In *Annual international cryptology conference* (pp. 213-229). Springer, Berlin, Heidelberg.

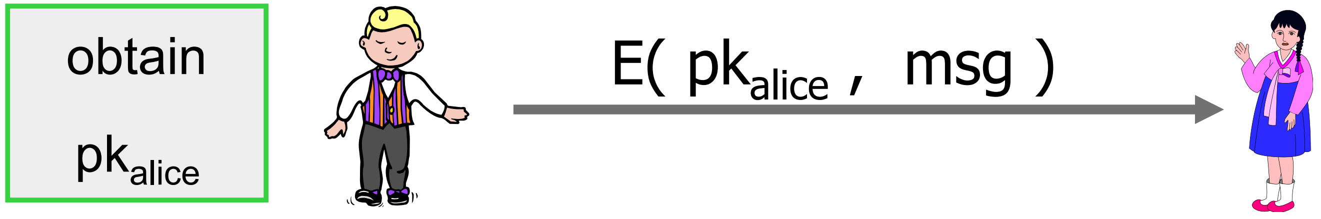
Recall: Pub-Key Encryption (PKE)

<u>PKE Three algorithms</u> : (G, E, D)

$G(\lambda) \rightarrow (pk, sk)$ outputs pub-key and secret-key

$E(pk, m) \rightarrow c$ encrypt m using pub-key pk

$D(sk, c) \rightarrow m$ decrypt c using sk



Example: ElGamal encryption

- $G(\lambda)$: $(G, g, q) \leftarrow \mathrm{GenGroup}(\lambda)$

  $sk := (\alpha \leftarrow F_p)$; $pk := (h \leftarrow g^{\alpha})$

- $E(pk, m \in G)$: $s \leftarrow Z_q$ and do $c \leftarrow (g^s, \; m \cdot h^{\mathbf{s}})$

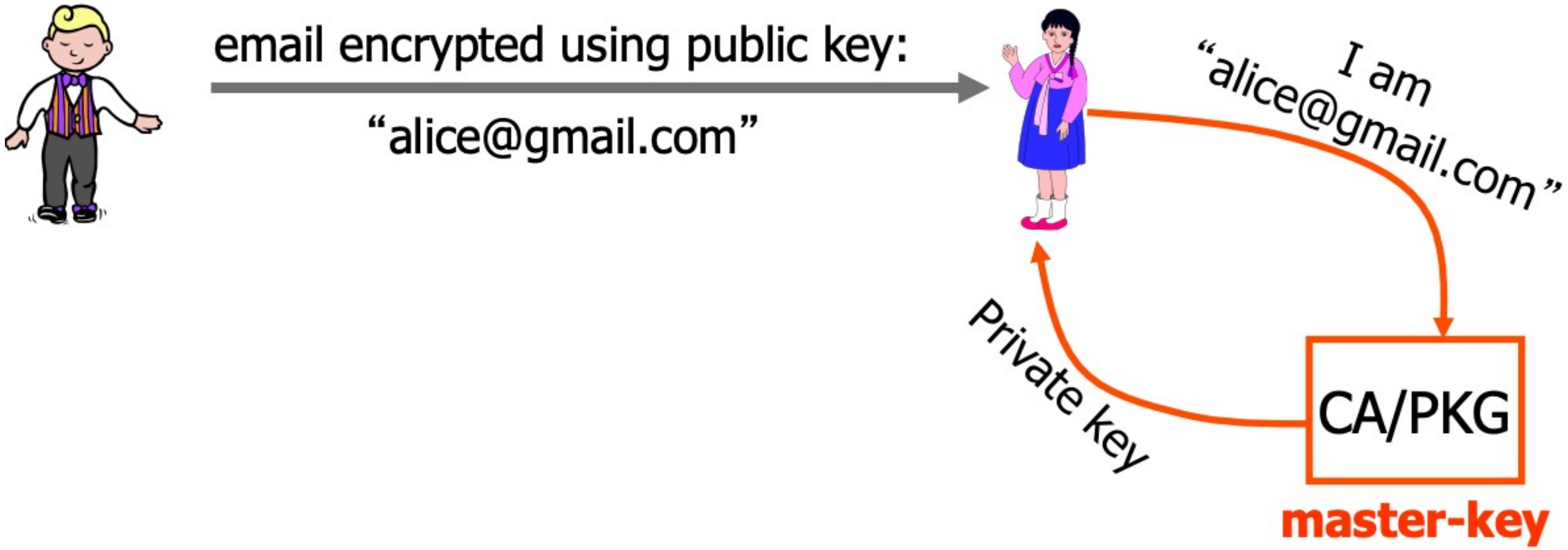
- $D(sk=\alpha, \; c=(c_1,c_2))$: observe $c_1^{\alpha} = (g^s)^{\alpha} = h^s$

- Security (IND-CPA) based on the DDH assumption:

  (g, h, g^s, h^s) indist. from (g, h, g^s, g^{rand})

Identity based encryption

- IBE: PKE system where PK is an arbitrary string
  - e.g. e-mail address, phone number, ip address



IBE in practice

Bob encrypts message with pub-key:

$$\underbrace{\text{“alice@hotmail} \ \| \ \text{role=accounting}}_{\text{policy-based encryption}} \ \| \ \underbrace{\text{time=week-num”}}_{\text{short-lived keys}}$$

Aug. 2011: “… Voltage SecureMail … with over one billion secure business emails sent annually and over 50 million worldwide users.”

Four algorithms

Four algorithms : (S,K,E,D)

$S(\lambda) \to (pp, mk)$	output params, pp, and master-key, mk
$K(mk, ID) \to d_{ID}$	outputs private key, d_{ID} , for ID
$E(pp, ID, m) \to c$	encrypt m using pub-key ID (and pp)
$D(d_{ID}, c) \to m$	decrypt c using d_{ID}

IBE "compresses" exponentially many pk's into a short pp

BasicIdent(IBE)

$params$

$params = \{q, G_1, G_2, \hat{e}, n, P, P_{pub}, H_1, H_2\}$

Bob:

$M \leftarrow \{0,1\}^n$

$r \overset{R}{\leftarrow} Z_q^*$

$U = rP$

$Q_{ID} = H_1(\text{alice@gmail.com})$

$g = \hat{e}(Q_{id}, P_{pub})$

$V = M \oplus H_2(g^r)$

$C = (U, V)$

Alice:

id=alice@gmail.com

$M = V \oplus H_2(\hat{e}(d_{id}, U))$

Note:

$H_2(\hat{e}(d_{id}, U))$
$= H_2(\hat{e}(sQ_{id}, rP))$
$= H_2(\hat{e}(Q_{id}, P)^{sr})$
$= H_2(\hat{e}(Q_{id}, sP)^r)$
$= H_2\left(\hat{e}\left(Q_{id}, P_{pub}\right)^r\right)$
$= H_2(g^r)$

$params$

id

Private key of Alice d_{id}

TA

$\hat{e}: G_1 \times G_1 \rightarrow G_2$

Generator $P \in G_1$

$msk = s \overset{R}{\leftarrow} Z_q^*$

$P_{pub} = sP$

$H_1: \{0,1\}^* \rightarrow G_1^*$

$H_2: G_2 \rightarrow \{0,1\}^n$

$Q_{id} = H_1(\text{id})$

$d_{id} = sQ_{id}$

BasicIdent (IBE)

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

Step 1: Run $\mathcal{G}$ on input k to generate a prime q, two groups $\mathbb{G}_1, \mathbb{G}_2$ of order q, and an admissible bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$. Choose a random generator $P \in \mathbb{G}_1$.

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

Step 3: Choose a cryptographic hash function $H_1 : \{0,1\}^* \to \mathbb{G}_1^*$. Choose a cryptographic hash function $H_2 : \mathbb{G}_2 \to \{0,1\}^n$ for some n. The security analysis will view H_1, H_2 as random oracles.

The message space is $\mathcal{M} = \{0,1\}^n$. The ciphertext space is $\mathcal{C} = \mathbb{G}_1^* \times \{0,1\}^n$. The system parameters are params $= \langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, P, P_{pub}, H_1, H_2 \rangle$. The master-key is $s \in \mathbb{Z}_q^*$.

Extract: For a given string $\mathsf{ID} \in \{0,1\}^*$ the algorithm does: (1) computes $Q_{\mathsf{ID}} = H_1(\mathsf{ID}) \in \mathbb{G}_1^*$, and (2) sets the private key d_{ID} to be $d_{\mathsf{ID}} = sQ_{\mathsf{ID}}$ where s is the master key.

Encrypt: To encrypt $M \in \mathcal{M}$ under the public key ID do the following: (1) compute $Q_{\mathsf{ID}} = H_1(\mathsf{ID}) \in \mathbb{G}_1^*$, (2) choose a random $r \in \mathbb{Z}_q^*$, and (3) set the ciphertext to be

$$C = \langle rP, \ \ M \oplus H_2(g_{\mathsf{ID}}^r) \rangle \quad \text{where} \quad g_{\mathsf{ID}} = \hat{e}(Q_{\mathsf{ID}}, P_{pub}) \in \mathbb{G}_2^*$$

Decrypt: Let $C = \langle U, V \rangle \in \mathcal{C}$ be a ciphertext encrypted using the public key ID. To decrypt C using the private key $d_{\mathsf{ID}} \in \mathbb{G}_1^*$ compute:

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

Theorem

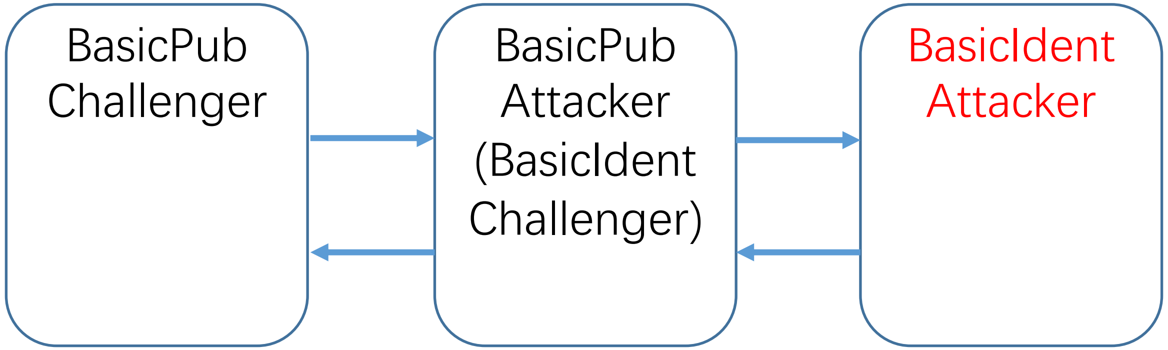
- Suppose H_1 and H_2 are random oracles. The BasicIdent is a semantically secure identity based encryption (IND-ID-CPA) assuming BDH is hard in groups generated by G.

BDH Assumption. Let $\mathcal{G}$ be a BDH parameter generator. We say that an algorithm $\mathcal{A}$ has advantage $\epsilon(k)$ in solving the BDH problem for $\mathcal{G}$ if for sufficiently large k:

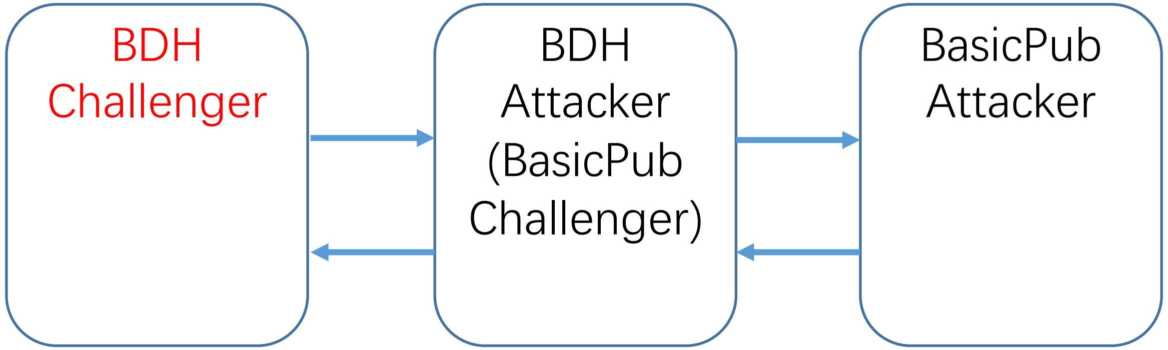
$$\mathrm{Adv}_{\mathcal{G},\mathcal{A}}(k) = \Pr\left[\mathcal{A}(q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, P, aP, bP, cP) = \hat{e}(P,P)^{abc} \;\middle|\; \begin{array}{l} \langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}\rangle \leftarrow \mathcal{G}(1^k), \\ P \leftarrow \mathbb{G}_1^*,\ a, b, c \leftarrow \mathbb{Z}_q^* \end{array}\right] \geq \epsilon(k)$$

Proof reductions

Lemma 1. *Let H_1 be a random oracle from $\{0,1\}^*$ to $\mathbb{G}_1^*$. Let $\mathcal{A}$ be an* IND-ID-CPA *adversary that has advantage $\epsilon(k)$ against* BasicIdent. *Suppose $\mathcal{A}$ makes at most $q_E > 0$ private key extraction queries. Then there is a* IND-CPA *adversary $\mathcal{B}$ that has advantage at least $\epsilon(k)/e(1+q_E)$ against* BasicPub.



Lemma 2. *Let H_2 be a random oracle from $\mathbb{G}_2$ to $\{0,1\}^n$. Let $\mathcal{A}$ be an* IND-CPA *adversary that has advantage $\epsilon(k)$ against* BasicPub. *Suppose $\mathcal{A}$ makes a total of $q_{H_2} > 0$ queries to H_2. Then there is an algorithm $\mathcal{B}$ that solves the BDH problem for $\mathcal{G}$ with advantage at least $2\epsilon(k)/q_{H_2}$.*



BasicPub (PKE)

$params = \{q, G_1, G_2, \hat{e}, n, P, P_{pub}, Q_{id}, H_2\}$

Bob:

$M \leftarrow \{0,1\}^n$

$r \overset{R}{\leftarrow} Z_q^*$

$U = rP$

$g = \hat{e}(Q_{id}, P_{pub})$

$V = M \oplus H_2(g^r)$

$C = (U, V)$

Alice:

$\hat{e}: G_1 \times G_1 \rightarrow G_2$

Generator $P \in G_1$

$s \overset{R}{\leftarrow} Z_q^*$

$P_{pub} = sP$

$H_2: G_2 \rightarrow \{0,1\}^n$

$Q_{id} \overset{R}{\leftarrow} G_1^*$

Private key: $d_{id} = sQ_{id}$

$M = V \oplus H_2(\hat{e}(d_{id}, U))$

Note:

$H_2(\hat{e}(d_{id}, U))$

$= H_2(\hat{e}(sQ_{id}, rP))$

$= H_2(\hat{e}(Q_{id}, P)^{sr})$

$= H_2(\hat{e}(Q_{id}, sP)^r)$

$= H_2\left(\hat{e}\left(Q_{id}, P_{pub}\right)^r\right)$

$= H_2(g^r)$

BasicPub (PKE)

BasicPub is described by three algorithms: keygen, encrypt, decrypt.

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

Step 1: Run $\mathcal{G}$ on input k to generate two prime order groups $\mathbb{G}_1, \mathbb{G}_2$ and a bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$. Let q be the order of $\mathbb{G}_1, \mathbb{G}_2$. Choose a random generator $P \in \mathbb{G}_1$.

Step 2: Pick a random $s \in \mathbb{Z}_q^*$ and set $P_{pub} = sP$. Pick a random $Q_{\mathsf{ID}} \in \mathbb{G}_1^*$.

Step 3: Choose a cryptographic hash function $H_2 : \mathbb{G}_2 \to \{0,1\}^n$ for some n.

Step 4: The public key is $\langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, P, P_{pub}, Q_{\mathsf{ID}}, H_2 \rangle$. The private key is $d_{\mathsf{ID}} = sQ_{\mathsf{ID}} \in \mathbb{G}_1^*$.

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

$$C = \langle rP, \;\; M \oplus H_2(g^r) \rangle \quad \text{where} \quad g = \hat{e}(Q_{\mathsf{ID}}, P_{pub}) \in \mathbb{G}_2^*$$

decrypt: Let $C = \langle U, V \rangle$ be a ciphertext created using the public key $\langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, P, P_{pub}, Q_{\mathsf{ID}}, H_2 \rangle$.
To decrypt C using the private key $d_{\mathsf{ID}} \in \mathbb{G}_1^*$ compute:

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

Lemma 1

$params1 = \{q, G_1, G_2, \hat{e}, n, P, P_{pub}, Q_{ID}, H_2\} \rightarrow params2 = \{q, G_1, G_2, \hat{e}, n, P, P_{pub}, H_1, H_2\}$

BasicPub Chal

$Q_{ID} \xleftarrow{R} Q_1^*, s \xleftarrow{R} Z_q^*$

$P_{pub} = sP, d_{ID} = sQ_{ID}$

$\xrightarrow{params1}$

B (BasicIdent simulator)

Maintain list $(ID, Q, b, coin)$

(1) H₁ query

$\xrightarrow{params2}$

$\xleftarrow{ID_i}$

If ID_i in list: return Q_i

else $coin_i \xleftarrow{R} \{0,1\}, b \xleftarrow{R} Z_q^*$

$\mathrm{coin_i} = 0: Q_i = bP$

$\mathrm{coin_i} = 1: Q_i = bQ_{ID}$

Add $ID, Q_i, b_i, coin_i$ to the list

$\xrightarrow{Q_i}$

(2) Private key query

$\xleftarrow{ID_i}$

If $coin_i = 1$, fail

else $d_i = b_i P_{pub}$

$\xrightarrow{d_i}$

(3) Challenge phase

$\xleftarrow{ID_{ch}, M_0, M_1}$

$H_1\ query(ID_{ch})$

$\xleftarrow{M_0, M_1}$

If $coin_{ch} = 0, fail$

else let $C' = (b^{-1}U, V)$

$\xrightarrow{C'}$

$\xrightarrow{C = (U, V)}$

$\xleftarrow{ID_j\ \text{query}}$

(4) ID_j query like (1)

$\xrightarrow{Q_j, d_j}$

A(BasicIdent Attacker)

$d_i = sQ_i = sb_iP = b_iP_{pub}$

$\downarrow \hat{b}$

$b \xleftarrow{R} \{0,1\}, r \xleftarrow{R} Z_q^*$

$U = rP$

$g = \hat{e}(Q_{ID}, P_{pub})$

$V = M_b \oplus H_2(g^r)$

If $coin_i = 1, B\ fail$
Because
$$d_i = sQ_i = sb_iQ_{ID} = b_id_{ID}$$
But B does not know d_{ID}

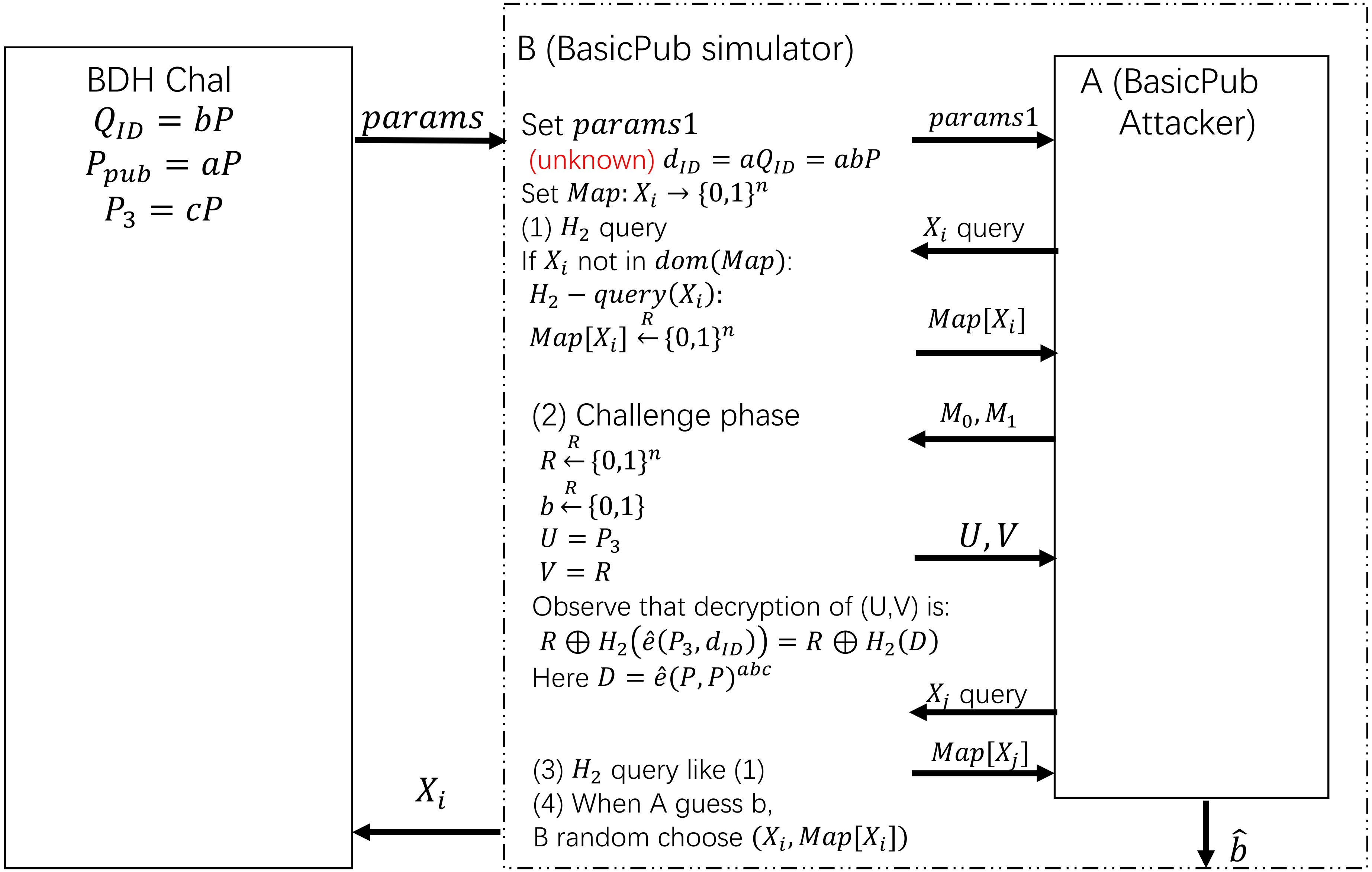
If $coin_{ch} = 1$
The corresponding private key
$$d_{ch} = sQ_{ch}$$
So: $\hat{e}\left({b_{ch}}^{-1}U, d_{ch}\right) =$
$\hat{e}({b_{ch}}^{-1}U, sQ_{ch})$
$= \hat{e}(U, sb^{-1}Q) = \hat{e}(U, sQ_{ID})$
$= \hat{e}(U, d_{ID})$
The decryption in BasicPub with C and d_{ID} is the same as decryption in BasicIdent with d_{ch} and C'.

Assume the probability that B does not abort is δ, the advantage of B is:

$$Adv_{IND-CPA}[B, BasicPub] = \delta Adv_{IND-\mathrm{ID}-CPA}^{RO}[A, BasicIdent]$$

Lemma 2

$params = \{P, aP, bP, cP\} \rightarrow params1 = \{q, G_1, G_2, \hat{e}, n, P, P_{pub}, Q_{ID}, H_2\}$



It can be seen that only A queries D, can A has the advantage to guess the encrypted messages.

So if A output $\hat{b} = b$ iff A queries D, which means that D exists in the H_2 list.