Identity Based Encryption

Rana Barua

Indian Statistical Institute
Kolkata

JU-ISI Workshop on

Many Facets of Cryptology

October 14-15, 2011

Public Key Cryptosystem(PKC)

- In a PKC setting each user has two keys: **a private key** and a **public key** that is published in a public directory
- If Bob wishes to send an encrypted message to Alice, he encrypts the message using Alice's **public key**.
- Alice, on receiving the encrypted message, decrypts it using her **private secret key**

Public Key Cryptosystem: Practical Issues

- Eve may pose to Bob as Alice and submit a public key to Bob. Thus the first issue is of trust. This is solved by having a certifying authority called CA
- Certificates may have a time limit or may be compromised. So need new certificate.
- Management of certificates is complex and cumbersome

Identity Based Encryption(IBE)

In order to overcome the shortcomings of PKE schemes, ID-based encryption was proposed by Adi Shamir in 1984. He was however only able to give an instantiation of identity-based signatures. Identity-based encryption remained an open problem for many years.
The Boneh/Franklin's pairing-based encryption scheme(also by Sakai, Ohgishi & Kashahara) and Cocks's encryption scheme based on quadratic residues both solved the IBE problem in 2001.

Identity Based Encryption : IBE

- IBE is a type of public key encryption scheme where the public key of a user can be any arbitrary string e.g. an e-mail id.
- When Alice wants to send a message to Bob; she encrypts it using the e-mail id of Bob as the public key.
- There is no need for Alice to go to the TA or CA to verify the public key of Bob.

IBE

An IBE consists of four algorithms

- **Setup:** This algorithm is run by the **PKG** one time for creating the whole IBE environment. The **master key** is kept secret and used to derive users' private keys, while the system parameters are made public. It accepts a security parameter k and outputs:
  A set $\mathcal{PP}$ of system parameters, a master key MSK.
- **Extract**: This algorithm is run by the PKG when a user requests his private key. It takes as input $\mathcal{PP}, MSK$ and an identity $ID \in \{0,1\}^*$ and returns the private key d_{ID} for user ID.
- **Encrypt:** Takes $\mathcal{PP}$, a message $m \in \mathcal{M}$ and $ID \in \{0,1\}^*$ and outputs the encryption $c \in \mathcal{C}$.
- **Decrypt:** Accepts $d_{ID}, \mathcal{PP}$ and $c \in \mathcal{C}$ and returns $m \in \mathcal{M}$

Admissible Bilinear Map

Let G_1 and G_2 be two groups of order q for some large prime q. The Boneh-Franklin IBE system makes use of a **bilinear map** $\hat{e} : G_1 \times G_1 \rightarrow G_2$ between these two groups. The map satisfy the following properties:

- **Bilinear**:

  $$\hat{e}(aP; bQ) = \hat{e}(P; Q)^{ab}$$

  for all $P; Q \in G_1$ and all $a; b \in Z$.
- **Non-degenerate**: The map does not send all pairs in $G_1 \times G_1$ to the identity in G_2.
- **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.

Boneh-Franklin IBE : BasicIdent

- **Setup**: Let P be a generator of G_1. Pick a random $s \in Z_q^*$ and set $P_{pub} = sP$. Choose cryptographic hash functions $H_1 : \{0,1\}^* \rightarrow G_1^*, H_2 : G_2 \rightarrow \{0,1\}^n$. The **master secret** is s and the public parameters are $PP =< P, P_{pub}, H_1, H_2 >$.
- **Key-Gen**: Given an identity $ID \in \{0,1\}^*$, compute $Q_{ID} = H_1(ID)$ and the private key is $d_{ID} = sQ_{ID}$.
- **Encrypt**: To encrypt $M \in \{0,1\}^n$ to ID compute $Q_{ID} = H_1(ID)$, choose a random $r \in Z_q^*$ and set the ciphertext: $C =< rP, M \oplus H_2(\hat{e}(Q_{ID}, P_{pub})^r) >$
- **Decrypt**: To decrypt $C =< U, V >$ using d_{ID} compute $V \oplus H_2(\hat{e}(d_{ID}, U)) = M$

Correctness

Note that Alice is able to decrypt using her secret key d_{ID} since

$$\hat{e}(d_{ID}, U) = \hat{e}(sQ_{ID}, rP) = \hat{e}(Q_{ID}, P)^{sr}$$
$$= \hat{e}(Q_{ID}, sP)^{r} = \hat{e}(Q_{ID}, P_{pub})^{r}.$$

Thus Alice can remove the "mask" from V and recover M.

Security Against Chosen Ciphertext Attack

The IND-ID-CCA security for an IBE is defined in terms of the following game between a challenger and an adversary $\mathcal{A}$. The adversary is allowed to place two types of oracle queries decryption queries to a decryption oracle $\mathcal{O}_d$ and key-extraction queries to a key-extraction oracle $\mathcal{O}_k$

- **Set-Up**. The challenger takes input a security parameter 1^k and runs the Setup algorithm of the IBE. It provides $\mathcal{A}$ with the system parameters PP while keeping the master key *msk* to itself.

Security Against Chosen Ciphertext Attack

- **Phase 1**: Adversary $\mathcal{A}$ makes a finite number of queries where each query is one of the two types:
  - key-extraction query (ID): This query is placed to the key-extraction oracle $\mathcal{O}_k$. It generates a private key d_{ID} of ID and returns it to $\mathcal{A}$.
  - decryption query (ID,C): This query is placed to the decryption oracle $\mathcal{O}_d$. It returns the resulting plaintext or **null** if the ciphertext cannot be decrypted.

  $\mathcal{A}$ is allowed to make these queries adaptively, i.e., any query may depend on the previous queries as well as their answers.
- Challenge: When $\mathcal{A}$ decides that Phase 1 is over, it fixes an identity ID^* and two equal length messagesM_0, M_1 under the (obvious) constraint that it has not asked for the private key of ID^*. The challenger chooses uniformly at random a bit $b \in \{0, 1\}$ and obtains a ciphertext C^* corresponding to M_b , It returns C^* as the challenge ciphertext to $\mathcal{A}$.

Security Against Chosen Ciphertext Attack

- Phase 2: $\mathcal{A}$ now issues additional queries just like Phase 1, with the (obvious) restriction that it cannot place a decryption query for the decryption of C^* under ID^* nor a key-extraction query for the private key of ID^*. The challenger responds as in Phase 1.
- Guess: $\mathcal{A}$ outputs a guess $\bar{b}$ of b . The **advantage** of the adversary $\mathcal{A}$ in attacking the IBE scheme H is defined as:

$$Adv_{\mathcal{A}} = |Pr[(b = \bar{b})] - 1/2|.$$

An IBE scheme is said to be IND-ID-CCA secure if for any (poly-time) adversary $\mathcal{A}$ that makes at most polynomial private key queries and at most polynomial decryption queries, $Adv_{\mathcal{A}}$ is negligible.

Security Against Chosen Plaintext Attack: IND-ID-CPA

• Security reduction of IBE protocols available in the literature generally concentrate on proving security in a weaker model. This is called security against chosen plaintext attack –: IND-ID- CPA security.

• The corresponding game is similar to the game defined above, except that the adversary is **not** allowed access to the decryption oracle $\mathcal{O}_d$.

• The adversary is allowed to place adaptive private key extraction queries to the key-extraction oracle $\mathcal{O}_k$ and everything else remains the same

Security of BasicIdent

BasicIdent is IND-ID-CPA secure provided

- H_1, H_2 are regarded as random functions
- Bilinear Diffie-Hellman problem is hard

Bilinear Diffie Hellman problem: (BDH):

- **Instance**: Given (P, aP, bP, cP), where a, b, c are random.
- **Task**: compute $\hat{e}(P, P)^{abc}$

FullIdent: IND-ID-CCA secure scheme

An IBE with chosen ciphertext security called FullIdent is obtained by applying the so called Fujisaki- Okamoto transformation to BasicIdent

- **Setup** Define 2 additional hash $H_3 : \{0,1\}^n \times \{0,1\}^n \rightarrow Z_p^*$; $H_4 : \{0,1\}^n \rightarrow \{0,1\}^n$. Same as the Setup of BasicIdent.
- **Key Generation**: Same as BasicIdent.
- **Encryption**: To encrypt M, compute $Q_{ID} = H_1(ID)$, choose random $\sigma \in \{0,1\}^n$ and set $r = H_3(\sigma, M)$. Ciphertext is $C =< rP, \sigma \oplus H_2(\hat{e}(Q_{ID}, P_{pub})^r), M \oplus H_4(\sigma) >$
- **Decryption:** To decrypt $C =< U, V, W >$, compute $V \oplus H_2(\hat{e}(d_{ID}, U)) = \sigma$ and then $W \oplus H_4(\sigma) = M$. Finally, set $r = H_3(\sigma, M)$ and verify whether $U = rP$. If not, reject; otherwise output M.

Hierarchical identity-based encryption (HIBE)

• Hierarchical identity-based encryption (HIBE) is an extension of IBE. Here identities are represented as vectors. In h-HIBE any identity ID is a tuple $(ID_1, ..., ID_t)$ where $1 \leq t \leq h$.
• Like IBE, here also the PKG has public parameters PP and a master key *msk*. For all identities at the first level the private key is generated by the PKG using *msk*. For identities at the second level on-wards, the private key can be generated by the PKG or by any of the ancestors of the identity.
• For example, the private key d_{ID} of ID can be generated by an entity whose identity is a prefix of ID and who has obtained the corresponding private key.

Motivation for HIBE

- The generation of private key can be a computationally intensive. The identity of an entity must be authenticated before issuing a private key and the private key needs to be transmitted securely.
- HIBE reduces the workload of the PKG by
  - delegating the task of private key generation and
  - authentication of identity and secure transmission of private key to its lower levels.
- HIBE has many interesting applications: forward secure encryption schemes and broadcast encryption schemes.

HIBE

- **Set-Up**: Takes input a security parameter 1^k and returns public parameters PP and the master secret key *msk*. PP is public while *msk* is known only to PKG. In case, there is some maximal level h of the HIBE, then that is also made public.
- **Key-Gen**: Takes as input an identity tuple $ID = (ID_1, ..., ID_j)$, and the private key $d_{ID}|j-1$ for the identity $(ID_1, ..., ID_{j-1})$ and returns a private key d_{ID}
- **Encrypt**: Takes as input PP, an identity ID and a message M and outputs a ciphertext C.
- **Decrypt**: This algorithm takes as input the public parameters PP, an identity ID, a ciphertext C and a private key d_{ID} and returns the message or **null** if the ciphertext is not valid.

Gentry-Silverberg's BasicHIBE

- **Set-up**: Let P be a generator of G_1. Pick $x_0 \in Z_p^*$ and set $P_{pub} = x_0 P$. Choose hash functions $H_1 : \{0,1\}^* \to G_1$ and $H_2 : G_2 \to \{0,1\}^n$. $PP =< P, P_{pub}, H_1, H_2 >$, while the master secret is x_0.
- **Key-Gen**: Given $ID = (ID_1, ..., ID_j)$ PKG chooses $x_1, ..., x_{j-1} \in Z_p^*$; computes $d_i = x_i P$ for $1 \leq i \leq j-1$; $d_j = \sum_{i=1}^{j} x_{i-1} Q_i$, where $Q_i = H_1(ID_1, ..., ID_i), 1 \leq i \leq j$. It gives $d_{ID} = (d_1, ..., d_j)$ to ID.
  A private key for ID can also be generated by its parent. Let $ID|j-1 = (ID_1, ..., ID_{j-1})$ be the parent of ID, i.e., $ID|j-1$ is one level up. Let private key of $ID|j-1$ be $d_{ID|j-1} = (d'_1, ..., d'_{j-1})$. Then d_{ID} is formed by $ID|j-1$ as follows: compute $Q_{ID} = H_1(ID_1, ..., ID_j)$, choose $x_{j-1} \in Z_p^*$; set $d_j = d'_{j-1} + x_{j-1} Q_{ID}$ and $d_i = d'_i$ for $1 \leq i \leq j-2$ and $d_{j-1} = x_{j-1} P$. The private key, $d_{ID} = (d_1, ..., d_j)$ is given to ID.

BasicHIBE

- **Encrypt:** To encrypt M under the identity $ID = (ID_1, ..., ID_j)$, compute $Q_i = H_1(ID_1, ..., ID_i)$ for $1 \leq i \leq j$. Choose a random $r \in Z_p^*$ and set the ciphertext $C =< rP, rQ_2, ..., rQ_j, M \oplus H_2(\hat{e}(P_{pub}, Q_1)^r) >$
- **Decrypt**: Given $C = (U_0, U_2, ..., U_j, V_i)$ and $d_{ID} =< d_1, ..., d_j >$, compute

$$V \oplus H_2 \left(\frac{\hat{e}(U_0, d_j)}{\prod_{i=2}^{j} \hat{e}(d_{i-1}, U_i)} \right) = M$$

Fuzzy IBE

- One common feature of all previous Identity-Based Encryption systems is that they view identities as a string of characters. And another feature of all previous IBE is that a sender used to encrypt the message for a **particular recipient** not for a group of recipients. Suppose Alice want to share a secret data for a group of people but she doesn't know the exact identities of all who should be able to access the data,rather she may only have a way to describe them in terms of descriptive attributes or credentials. So in this scenario previous IBE won't work.

Fuzzy IBE

- In 2005, Sahai and Waters first proposed a new type of IBE scheme called `Fuzzy Identity-Based Encryption` in which, we view identities as a *set of descriptive attributes*. In a `Fuzzy Identity-Based Encryption` scheme, a user with the secret key for the identity ω is able to decrypt a ciphertext encrypted with the identity ω' if and only if ω and ω' are within a certain distance of each other as judged by some metric. Therefore, this system allows for a certain amount of error-tolerance in the identities

Fuzzy IBE

- One application of fuzzy IBE is an Identity-Based Encryption system that uses biometric identities. That is, we can view a user's biometric, for example an iris scan, as the user's identity described by several attributes and then encrypt to the user using his biometric identity. Since biometric measurements are noisy, we cannot use existing IBE systems. However, the error-tolerance property of `Fuzzy-IBE` allows for a private key (derived from a measurement of a biometric) to decrypt a ciphertext encrypted with a slightly different measurement of the same biometric.

Attribute Based Encryption (ABE)

- In another direction, `Fuzzy-IBE` can be used for an application that they called `Attribute-Based Encryption`. In this application, a party will wish to encrypt a document to all users that have a certain set of attributes. For example, in a computer science department, the chairperson might want to encrypt a document to all of its systems faculty on a hiring committee. In this case it would encrypt to the identity $\{hiring\ committee, faculty, systems\}$. Any user who has an identity that contains all of these attributes could decrypt the document.
- In other words, a sender encrypts a message with a functionality or policy over a set attributes and any user who has sufficient attributes or credentials to satisfy the corresponding functionality or policy, can decrypt the message.

Attribute Based Encryption (ABE)

There are two type Attribute-Based Encryption(ABE).

- First one is called **Key-policy ABE** and,
- the other **Ciphertext-policy ABE**(CP-ABE).
- In Key-policy ABE(KP-ABE) scheme, ciphertext is associated with a set of attributes and user's key is associated with a policy.
- In CP-ABE, ciphertext is associated with a policy and the user's secret key is associated with a set of attributes.

Goyal, Panday, Sahai and Waters (2006) introduced the general Key-Policy ABE where encrypting data are shared at a fine-grained level.

Access Structures

Definition (Access Structure)

Let $\{P_1, P_2, ..., P_n\}$ be a set of parties. A collection $\mathbb{A} \subset 2^{\{P_1, P_2, ..., P_n\}}$ is monotone if $\forall B, C$: if $B \in \mathbb{A}$ and $B \subset C$ then $C \in \mathcal{A}$. An access structure (respectively, monotone access structure) is a collection (respectively, monotone collection) $\mathbb{A}$ of non-empty subsets of $\{P_1, P_2, ..., P_n\}$, i.e., $\mathbb{A} \subset 2^{\{P_1, P_2, ..., P_n\}} \setminus \{\emptyset\}$. The members of $\mathbb{A}$ are called the authorized sets, and the sets not in $\mathbb{A}$ are called the unauthorized sets.

Definition (Satisfying an Access Structure)

Let $\mathbb{A}$ be a monotone access structure over the set parties $\{P_1, P_2, ..., P_n\}$. A set of attributes S satisfies the access structure $\mathbb{A}$ if $S \in \mathbb{A}$.

Key Policy-ABE

A KP-ABE scheme consists of four algorithms.

- **Setup:**This is a randomized algorithm that takes a security parameter as input. It outputs the public parameters **PP** and a master key **MSK**.
- **Encryption:** This is a randomized algorithm that takes as input a message M, a set of attributes γ, and the public parameters **PP**. It outputs the ciphertext E.
- **Key Generation** This is a randomized algorithm that takes as input an access structure $\mathcal{A}$, the master key **MSK** and the public parameters **PP**. It outputs a decryption key D.
- **Decryption** This algorithm takes as input the ciphertext E encrypted under the set γ of attributes, the decryption key D for access control structure $\mathcal{A}$ and the public parameters **PP**. It outputs the message M if $\gamma \in \mathcal{A}$.

CP-ABE

Goyal et al. left an open the problem of constructing `Ciphertext-policy ABE`. In 2007, Bethencourt, Sahai and Waters came up with a `Ciphertext-policy ABE` construction. In both the schemes, the policy is taken to be an access structure, more precisely an access tree. If a user's set of attributes(or policy) satisfies the policy(or set of attributes) of ciphertext, then only the user can decrypt the ciphertext.

CP-ABE

An ciphertext-policy attribute based encryption scheme consists of four fundamental algorithms: Setup, Encrypt, KeyGen, and Decrypt. In addition, we allow for the option of a fifth algorithm Delegate

CP-ABE

- **Setup**. The setup algorithm takes a security parameter. It outputs the public parameters PP and a master key **MSK**.
- **Encrypt**$(PP, M, \mathbb{A})$. The encryption algorithm takes as input the public parameters **PP**, a message M, and an access structure $\mathbb{A}$ over the universe of attributes. The algorithm will encrypt M and produce a ciphertext $\mathcal{C}$ such that only a user that possesses a set of attributes that satisfies the access structure will be able to decrypt the message. We will assume that the ciphertext implicitly contains $\mathbb{A}$.

- **Delegate**$(SK, \tilde{S})$. The delegate algorithm takes as input a secret key SK for some set of attributes S and a set $\tilde{S} \subset S$. It output a secret key $\tilde{SK}$ for the set of attributes $\tilde{S}$.

Security Model for ***CP-ABE***

- **Setup**. The challenger runs the Setup algorithm and gives the public parameters, PP to the adversary.
- **Phase 1**. The adversary makes repeated private keys corresponding to sets of attributes $S_1, ..., S_{q_1}$.
- **Challenge**. The adversary submits two equal length messages M_0 and M_1 . In addition the adversary gives a challenge access structure $\mathbb{A}$ such that none of the sets $S_1, ..., S_{q_1}$ from **Phase 1** satisfy the access structure. The challenger flips a random coin b, and encrypts M_b under $\mathbb{A}$. The ciphertext CT is given to the adversary.

Security Model for ***CP-ABE***

- **Phase 2**. Phase 1 is repeated with the restriction that none of sets of attributes $S_{q_{1+1}}, ..., S_q$ satisfy the access structure corresponding to the challenge.
- **Guess**. The adversary outputs a guess b of b.

The advantage of an adversary $\mathcal{A}$ in this game is defined as $Pr[b = b]1/2$.
Definition An ciphertext-policy attribute-based encryption scheme is secure if all polynomial time adversaries have at most a negligible advantage in the above game.

Fully secure CP-ABE

- Previous constructions of ABE were only proven to be selectively secure in mostly generic bilinear group model. Recently (Eurocrypt2010) A.Lewko, Okamoto, Sahai, Takashima and Waters proposed a fully secure ABE scheme. They construct their system in composite order bilinear groups, where the order is a product of three primes and they proved the security of their scheme in a standard model under three static assumptions.

Fully secure CP-ABE Scheme

- **Setup**$(\lambda; U) \rightarrow PK; MSK$ The setup algorithm chooses a bilinear group G of order $N = p_1 p_2 p_3$ (3 distinct primes). We let G_{p_i} denote the subgroup of order p_i in G. It then chooses random exponents $\alpha; a \in Z_N$, and a random group element $g \in G_{p_1}$. For each attribute $i \in U$, it chooses a random value $s_i \in Z_N$. The public parameters PK are $N; g; g^a; e(g; g)^\alpha; T_i = g^{s_i} \forall i$. The master secret key MSK is α and a generator X_3 of G_{p_3} .
- **KeyGen**$(MSK; S; PK) \rightarrow SK$ The key generation algorithm chooses a random $t \in Z_N$, and random elements $R_0; R_0'; R_i \in G_{p_3}$. The secret key is:

$$S; K = g^\alpha g^{at} R_0; L = g^t R_0'; K_i = T_i^t R_i \forall i \in S :$$

Fully secure CP-ABE Scheme

- **Encrypt**$((A;\rho);PK;M) \rightarrow CT$ A is an $\ell \times n$ matrix and ρ is map from each row A_x of A to an attribute $\rho(x)$. The encryption algorithm chooses a random vector $v \in Z_N^n$, denoted $v = (s; v_2; :::; v_n)$. For each row A_x of A, it chooses a random $r_x \in Z_N$. The ciphertext is (we also include $(A;\rho)$ in the ciphertext, though we do not write it below):

$$C = Me(g;g)^{\alpha s}; C' = g^s;$$

$$C_x = g^{aA_x.v} T_{\rho(x)}^{-r_x}; D_x = g^{r_x} \forall x :$$

Fully secure CP-ABE

- **Decrypt**$(CT; PK; SK) \rightarrow M$ The decryption algorithm computes constants $w_x \in Z_N$ such that $\Sigma_{\rho(x)\in S} w_x A_x = (1; 0; :::; 0)$. It then computes:

$$e(C'; K)/\Pi_{\rho(x)\in S}(e(C_x; L)e(D_x; K_{\rho(x)}))^{w_x} = e(g; g)^{\alpha s}$$

  Then M can be recovered as $C/e(g; g)^{\alpha s}$.