Tutorial on
Identity-Based Cryptography

Dr. Abhijit Das
Associate Professor
Department of Computer Science and Engineering
Indian Institute of Technology Kharagpur
http://cse.iitkgp.ac.in/~abhij/

June 29, 2017
Short Term Course on Introduction to Cryptography
Department of Mathematics, IIT Kharagpur
Public keys are used for encryption and digital signature verification.

Private keys are used for decryption and digital signature generation.

Public keys are accessible to all parties.

Private keys are to be kept secret.

How to associate entities with their respective public keys?

An attacker may present a harmful key as the public key of a victim.

Before using a public key, one should verify that the key belongs to the claimed party.
There is a trusted Certification Authority (CA).
CA issues public-key certificates to parties.
A certificate contains a public key, some identifying information of the party to whom the key belongs, a period of validity.
The certificate is digitally signed by the CA.
Key compromise and/or malicious activities may lead to revocation of certificates.
The CA maintains a list of revoked certificates.
Alice wants to send an encrypted message to Bob.
Alice obtains Bob’s public-key certificate.
Alice verifies the signature of the CA on the certificate.
Alice confirms that Bob’s identity is stored in the certificate.
Alice checks the validity of the certificate.
Alice ensures that the certificate does not reside in the revocation list maintained by the CA.
Alice then uses Bob’s public key for encryption.
Identity-Based Cryptography: A Viable Substitute

Problems of Public-Key Certificates

- A trusted CA is needed.
- Every certificate validation requires contact with the CA for the verification key and for the revocation list.

Identity-Based Public Keys

- Alice’s identity (like e-mail ID) is used as her public key.
- No contact with the CA is necessary to validate public keys.
- A trusted authority is still needed: Private-Key Generator (PKG) or Key-Generation Center (KGC).
- Each party should meet the PKG privately once (registration phase).
- **Limitation:** Revocation of public keys may be difficult.
Historical Remarks

- Shamir (Crypto 1984) introduces the concept of identity-based encryption (IBE) and signature (IBS). He gives a concrete realization of an IBS scheme.
- In early 2000s, bilinear pairing maps are used for concrete realizations of IBE schemes.
- Boneh and Franklin (Crypto 2001) propose an IBE scheme. Its security is proved in the random-oracle model.
- Boneh and Boyen (EuroCrypt 2004) propose an IBE scheme whose security is proved without random oracles.
- Joux (ANTS 2004) proposes a pairing-based three-party key-agreement protocol.
A Failed Attempt

- Let $H$ map public identities to unique odd integers.
- In order to generate an RSA key pair, Bob (the recipient) takes $e = H(ID_{Bob})$.
- Bob keeps on generating random primes $p, q$ until $\gcd(p - 1, e) = \gcd(q - 1, e) = 1$.
- Bob publishes $e$ and $n = pq$.
- Bob computes $d \equiv e^{-1} \pmod{\phi(n)}$ (private key).

- The public key of Bob is the pair $(e, n)$.
- An attacker can generate $n$ as Bob does.
- A certificate is needed to validate $n$. 
Let $G_1, G_2, G_3$ be groups of finite order $r$ (usually prime).
$G_1, G_2$ are additive, and $G_3$ multiplicative.

A **bilinear pairing map** $e : G_1 \times G_2 \to G_3$ satisfies:

- $e(P_1 + P_2, Q) = e(P_1, Q)e(P_2, Q)$ and $e(P, Q_1 + Q_2) = e(P, Q_1)e(P, Q_2)$ for all $P, P_1, P_2 \in G_1$ and $Q, Q_1, Q_2 \in G_2$.

- $e(aP, bQ) = e(P, Q)^{ab}$ for all $P \in G_1$, $Q \in G_2$, and $a, b \in \mathbb{Z}$.

- $e$ is non-degenerate, that is, $e(P, Q)$ is not the identity of $G_3$ for some $P, Q$.

- $e$ is efficiently computable.

Example: Weil or reduced Tate pairing over elliptic curves. $G_1, G_2$ are elliptic-curve groups, $G_3$ is a subgroup of the multiplicative group of a finite field.

Special case: $G_1 = G_2 = G$. Example: Distorted Weil or Tate pairing on supersingular curves.
Let $G$ be an additive group of prime order $r$.

**Computational Diffie–Hellman Problem (CDHP):** Given $P, aP, bP \in G$, compute $abP$.

**Decisional Diffie–Hellman Problem (DDHP):** Given $P, aP, bP, zP \in G$, decide whether $z \equiv ab \pmod{r}$.

If $e : G \times G \to G_3$ is a bilinear pairing map, the DDHP is easy: Check whether $e(aP, bP) = e(P, zP)$.

The CDHP is not known to be aided by $e$.

$G$ is called a **gap Diffie–Hellman (GDH)** group.

**External Diffie–Hellman Assumption (XDH):** Presence of bilinear pairing maps $e : G_1 \times G_2 \to G_3$ does not make DDHP easy in $G_1$ or $G_2$ (different groups).
Bilinear Diffie–Hellman Problems

- Let $e : G \times G \rightarrow G_3$ be a bilinear pairing map.
- **Decisional Bilinear Diffie–Hellman Problem (DBDHP):** Given $P, aP, bP, cP, zP \in G$, decide whether $z \equiv abc \pmod{r}$ (that is, $e(P, P)^z = e(P, P)^{abc}$).
- **Bilinear Diffie–Hellman Assumption:** The BDHP and DBDHP are computationally infeasible for suitably chosen groups even in the presence of efficiently computable bilinear pairing maps.
- DLP in $G$ should be difficult (as $e(aP, bP)^c = e(P, P)^{abc}$).
- DHP in $G$ should be difficult (as $e(abP, cP) = e(P, P)^{abc}$).
Set-up Phase

The PKG/KGC/TA sets up the following parameters.

- Groups $G, G_3$ of prime order $r$.
- A bilinear pairing map $e : G \times G \rightarrow G_3$.
- A generator $P$ of $G$.
- A hash function $H$ to map public identities (like e-mail addresses) to elements of $G$.
- PKG’s master secret key $s \in_U \mathbb{Z}_r$.
- PKG’s public key $P_{PKG} = sP$. 
Registration (Key-Extraction) Phase

- A user $Usr$ meets the PKG securely.
- The PKG hashes the public identity of $Usr$ to generate $P_{Usr} = H(ID_{Usr}) \in G$.
- The PKG delivers $D_{Usr} = sP_{Usr} \in G$ to $Usr$.

Notes

- Anybody can compute the hashed public identity $P_U$.
- Computation of $D_{Usr}$ is equivalent to solving DHP in $G$ ($P_{Usr} = uP$, $P_{PKG} = sP$, and $D_{Usr} = usP$). This is assumed to be intractable.
- Alice and Bob securely registers with the PKG to get $D_{Alice}$ and $D_{Bob}$.
- Anybody can compute $P_{Alice}$ and $P_{Bob}$. 
Key Agreement (Non-interactive)

- Alice computes Bob’s hashed identity $P_{Bob}$.
- Alice computes $S_{Alice} = e(D_{Alice}, P_{Bob})$.
- Bob computes Alice’s hashed identity $P_{Alice}$.
- Bob computes $S_{Bob} = e(P_{Alice}, D_{Bob})$.
- $S_{Alice} = e(D_{Alice}, P_{Bob}) = e(sP_{Alice}, P_{Bob}) = e(P_{Alice}, P_{Bob})^s = e(P_{Alice}, sP_{Bob}) = e(P_{Alice}, D_{Bob})$ is the shared secret.

Security (Based on BDHP)

- Let $P_{Alice} = aP$ and $P_{Bob} = bP$. We have $P_{PKG} = sP$.
- $P, aP, bP, sP$ are known to any attacker.
- The shared secret is $e(P_{Alice}, P_{Bob})^s = e(P, P)^{abs}$. 
Joux Three-Party Key Agreement

- Not an identity-based protocol.
- Alice, Bob, and Carol respectively generate $a, b, c \in U \mathbb{Z}_r$.
- Alice sends $aP$ to Bob and Carol.
- Bob sends $bP$ to Alice and Carol.
- Carol sends $cP$ to Alice and Bob.
- Alice computes $e(bP, cP)^a = e(P, P)^{abc}$.
- Bob computes $e(aP, cP)^b = e(P, P)^{abc}$.
- Carol computes $e(aP, bP)^c = e(P, P)^{abc}$.
- Man-in-the-middle attack possible.
Boneh–Franklin IBE

Set-up Phase

The PKG/KGC/TA sets up the following parameters.

- Groups $G, G_3$ of prime order $r$.
- A bilinear pairing map $e : G \times G \rightarrow G_3$.
- A generator $P$ of $G$.
- An encoding function $H_1$ to map public identities (like e-mail addresses) to elements of $G$.
- A function $H_2 : G_3 \rightarrow \{0, 1\}^n$ ($n$ is the message length).
- PKG’s master secret key $s \in U \mathbb{Z}_r$.
- PKG’s public key $P_{PKG} = sP$. 
BF IBE (Contd)

Registration (Key-Extraction) Phase

- A user $Usr$ meets the PKG securely.
- The PKG encodes the public identity of $Usr$ to generate $P_{Usr} = H_1(ID_{Usr}) \in G$.
- The PKG delivers $D_{Usr} = sP_{Usr} \in G$ to $Usr$.

Notes

- Anybody can compute the encoded public identity $P_{Usr}$.
- Computation of $D_{Usr}$ is equivalent to solving the DHP in $G$. This is assumed to be intractable.
- Bob (the recipient) securely meets the PKG to get $D_{Bob}$.
- Anybody can compute $P_{Bob}$.
Encryption

Alice wants to send $M \in \{0, 1\}^n$ to Bob.

- Alice computes $P_{Bob} = H_1(ID_{Bob})$.
- Alice computes $g = e(P_{Bob}, P_{PKG}) \in G_3$.
- Alice chooses a random $a \in U \mathbb{Z}_r^*$.
- Alice computes $U = aP$ and $V = M \oplus H_2(g^a)$.
- A ciphertext for $M$ is the pair $(U, V) \in G \times \{0, 1\}^n$.

**Note:** $H_2(g^a)$ acts as a mask to hide $M$. 
BF IBE (Contd)

Decryption

- Bob recovers $M$ from $(U, V)$ as $M = V \oplus H_2(e(D_{Bob}, U))$.

Correctness

- Let $P_{Bob} = bP$.
- $g^a = e(P_{Bob}, P_{PKG})^a = e(bP, sP)^a = e(P, P)^{abs}$.
- $e(D_{Bob}, U) = e(sP_{Bob}, aP) = e(sbP, aP) = e(P, P)^{abs}$.

Textbook Security

- Malice knows $aP = U$, $bP = P_{Bob}$, and $sP = P_{PKG}$.
- His ability of computing the mask is equivalent to solving an instance of the BDHP.
Insecurity against Active Attacks

- Malice wants to get $M$ corresponding to $(U, V)$.
- Malice gets assistance from Bob’s decryption box.
- The decryption box decrypts any ciphertext except $(U, V)$.
- The decryption box may refuse to answer if decryption results in the message $M$.
- Malice queries with $U' = U$ and $V' = W \oplus V$ for some $W \in U \{0, 1\}^n \setminus \{0^n\}$ chosen by Malice.
- $(U', V') \neq (U, V)$ encrypts $M' = M \oplus W$.
- For random $W$, $M'$ is a random $n$-bit string.
- The decryption box returns $M'$.
- Malice computes $M = M' \oplus W$. 
IND-CPA (Semantic) Security

The IND-CPA Game

- Malice chooses messages $m_0, m_1$ of the same bit length.
- Malice sends $m_0, m_1$ to the victim’s encryption oracle $\mathcal{O}$.
- $\mathcal{O}$ chooses a bit $b \in \{0, 1\}$, and encrypts $m_b$.
- The ciphertext $c^*$ of $m_b$ is sent to Malice as the challenge.
- Malice outputs a bit $b'$. Malice wins if and only if $b' = b$.

Notes

- Encryption must be randomized.
- A random guess of Malice succeeds with probability $1/2$.
- Malice succeeds with probability $1/2 + \varepsilon$ ($\varepsilon$ is advantage).
- If $\varepsilon$ is less than one over all polynomial expressions in the security parameter, the scheme in IND-CPA secure.
IND-CCA Security

- Malice has access to the victim’s decryption oracle $O$.
- Malice sends indifferent chosen ciphertexts for decryption before the IND-CPA game.
- Malice sends adaptive chosen ciphertexts for decryption after the IND-CPA game.
- Query on $c^*$ cannot be made after the challenge is posed.
- CCA1: Decryption assistance stops after the challenge.
- CCA2: Decryption assistance continues after the challenge.
- The cryptanalysis training before and/or after the challenge is supposed to help Malice in winning.
- CCA2 is the accepted standard model of the adversary.
In an IBE scheme, there are registration requests.
Malice has access to the registration oracle \( R \).
Malice can make queries to \( R \) before and after the challenge.
Bob is the targeted victim (\( c^* \) is generated by Bob’s encryption oracle).
Malice may never ask \( R \) to reveal Bob’s private key.
Malice may ask \( R \) to reveal Bob’s public key (or can compute the public key himself).
A **random oracle** is a function $H$ from $\{0, 1\}^*$ to a finite set $D$.  
- $H$ is deterministic.  
- For each input $\alpha \in \{0, 1\}^*$, $H(\alpha)$ is a uniformly random element of $D$.  
- $H$ is efficiently computable.

**In theory:** Random oracles do not exist.

**In practice**

- $H$ can be treated as a random oracle if its output cannot be distinguished from truly random output by any probabilistic polynomial-time algorithm.  
- Cryptographic hash functions are used as random oracles.
Security Proof in the Random-Oracle Model (ROM)

In Real Life

- Malice can compute all hash functions himself.
- Malice can access encryption/decryption/registration oracles.

In ROM Proofs

- Malice communicates only with Ronald.
- Ronald has no access to the victim’s/PKG’s private keys.
- Ronald has full control over hash computations.
- Malice has to contact Ronald if he wants to hash anything.
- By manipulating hash values, Ronald reliably simulates encryption/decryption/registration queries.
- If the simulation is reliable, Malice unleashes his cryptanalytic prowess to win the game.
Ronald maintains a table $T$ of $(\alpha, H(\alpha))$ values. Initially, $T$ is empty. Whenever some $H(Q)$ needs to be returned, Ronald searches for $Q$ in $T$. If the search is successful, the second stored component is returned. If the search is unsuccessful, Ronald chooses a uniformly random $\gamma \in D$, stores $(Q, \gamma)$ in $T$, and returns $\gamma$. The attack runs for polynomial time, so the size of $T$ never grows beyond polynomial. Searching in $T$ is efficient. Sometimes additional information is stored in entries of $T$. 

**Identity-Based Cryptography**
Abhijit Das
IND-ID-CPA Proof of BF IBE in the ROM

- $H_1, H_2$ are treated as hash functions (random oracles).
- Step 1: Infeasibility of BDHP in $G$ implies IND-CPA security.
- Step 2: IND-CPA security implies IND-ID-CPA security.
- If there is an IND-ID-CPA adversary $A$ for BF IBE, then there is an IND-CPA adversary $B$ for BF IBE.
- If there is an IND-CPA adversary $B$ for BF IBE, then Ronald can reliably solve the BDHP in $G$.
- Let the advantage of $A$ be $\varepsilon$.
- Let the number of $H_1$ and $H_2$ queries be $q_{H_1}$ and $q_{H_2}$.
- Then, the advantage of $B$ is $\frac{\varepsilon}{e(1+q_{H_1})}$, and the advantage of Ronald in solving the BDHP is $\frac{2\varepsilon}{e(1+q_{H_1})q_{H_2}}$. 
IND-CPA Security Implies IND-ID-CPA Security

- Let $\mathcal{A}$ be a PPT IND-ID-CPA adversary.
- Ronald interacts with $\mathcal{A}$ and $\mathcal{O}$.
- System parameters $G, G_3, r, e, P, P_{PKG}, n, H_2$ are public.
- The master secret $s$ is fixed, but not known to $\mathcal{A}$, Ronald, or $\mathcal{O}$.
- Bob is the targeted victim decided by $\mathcal{A}$.
- A registration query to get $D_{Bob}$ cannot be made by $\mathcal{A}$.
- A query to get $P_{Bob} = H_1(ID_{Bob})$ is allowed. $\mathcal{A}$ cannot know $P_{Bob}$ without making this query.
- $H_1$ is a random oracle to $\mathcal{A}$.
- The encryption oracle $\mathcal{O}$ uses actual hash values. $P^{(\mathcal{O})}_{Bob}$ and $D^{(\mathcal{O})}_{Bob}$ are the actual (not simulated) keys of Bob. Both Ronald and $\mathcal{O}$ knows how to compute $P^{(\mathcal{O})}_{Usr}$ for any $Usr$. 
The Reduction Mechanism

A

\[ M_0, M_1 \]

C**

Hash and Key Extraction Queries

B

\[ M_0, M_1 \]

C* = E(M_b)

\[ b' \]

O

Identity-Based Cryptography

Abhijit Das
Handling $H_1$ Queries

**Key Extraction:** $P_{Usr} = H_1(ID_{Usr})$, $D_{Usr} = sP_{Usr}$.

**Encryption:** $U = aP$, $g = e(P_{Bob}, P_{PKG})$, $V = M \oplus H_2(g^a)$.

**Decryption:** $M = V \oplus H_2(e(D_{Bob}, U))$.

- $H_1$ hashes public ID’s to public keys.
- Public keys are needed for key extraction and encryption.
- Ronald does not know $s$. Let $P_{Usr} = tP$ (where $Usr \neq Bob$).
  Then, $D_{Usr} = sP_{Usr} = stP = t(sP) = tP_{PKG}$.
- If $Usr = Bob$, $D_{Bob}$ is not needed. Let $P_{Bob} = tP_{Bob}^{(0)}$, and $C^* = (U^*, V^*)$. Then, $e(D_{Bob}^{(0)}, U^*) = e(t^{-1}D_{Bob}, U^*) = e(D_{Bob}, t^{-1}U^*)$. So if $C^* = (U^*, V^*)$ is an actual encryption of $M_b$ done by $\theta$, then $C^{**} = (t^{-1}U^*, V^*)$ is an encryption of $M_b$ simulated by Ronald.
- When a query $H_1(ID_{Usr})$ comes, Ronald need not know whether $Usr$ is the targeted victim.
Handling $H_1$ Queries (Contd)

- Ronald maintains an $H_1$-table of $(ID_{Usr}, P_{Usr}, t, c)$ entries.
- Suppose that a query $H_1(ID_{Usr})$ comes.
- If $ID_{Usr}$ resides in the $H_1$-table, the corresponding $P_{Usr}$ is returned.
- Otherwise, Ronald tosses a coin to get $c$ such that $\Pr[c = 0] = \delta \approx 1$.
- If $c = 0$, Ronald assumes $ID \neq Bob$. He chooses random $t \in \mathbb{Z}_r^*$, computes $P_{Usr} = tP$, stores $(ID_{Usr}, P_{Usr}, t, 0)$ in his $H_1$-table, and returns $P_{Usr}$.
- If $c = 1$, Ronald assumes $ID = Bob$. He chooses random $t \in \mathbb{Z}_r^*$, computes $P_{Usr} = tP^{(\phi)}$, stores $(ID_{Usr}, P_{Usr}, t, 1)$ in his $H_1$-table, and returns $P_{Usr}$.
Handling Key-Extraction Queries

- \( \mathcal{A} \) asks Ronald to supply the private key \( D_{Usr} \) of \( Usr \).
- Ronald searches for \( ID_{Usr} \) in his \( H_1 \)-table.
- If the search fails, Ronald initiates an internal query for computing \( H_1(ID_{Usr}) \) (he may force \( c = 0 \) in this query).
- If the \( H_1 \)-table contains an entry \( (ID_{Usr}, P_{Usr}, t, c) \) with \( c = 1 \), Ronald aborts.
- Finally, suppose that the \( H_1 \)-table contains an entry \( (ID_{Usr}, P_{Usr}, t, c) \) with \( c = 0 \). Ronald computes and returns \( D_{Usr} = tP_{PKG} \).
- Ronald successfully handles a key-extraction query with probability \( \delta \).
Handling the IND-CPA Game

- $A$ sends the ID of a targeted victim Bob, and two messages $M_0, M_1$ of length $n$, to Ronald.
- Ronald searches for $ID_{Bob}$ in his $H_1$-table.
- If the search fails, Ronald initiates an internal query for computing $H_1(ID_{Bob})$ (he may force $c = 1$ in this query).
- If the $H_1$-table contains an entry $(ID_{Bob}, P_{Bob}, t, c)$ with $c = 0$, Ronald aborts.
- Finally, suppose that the $H_1$-table contains an entry $(ID_{Bob}, P_{Bob}, t, c)$ with $c = 1$.
  - Ronald forwards $ID_{Bob}, M_0, M_1$ to $O$.
  - $O$ chooses $b \in U \{0, 1\}$, and returns an actual (not simulated) encryption $C^* = (U^*, V^*)$ of $M_b$ using Bob’s public key.
  - Ronald forwards $C^{**} = (t^{-1}U^*, V^*)$ to $A$.
- Ronald successfully participates in the IND-CPA game with probability $1 - \delta$. 
Let $\mathcal{A}$ have a non-negligible advantage $\epsilon$.

If Ronald does not abort, his simulation is perfect. In this case, he has the same advantage $\epsilon$.

Let $q_{H_1}$ be the number of $H_1$-queries made.

Ronald does not abort with probability $\delta^{q_{H_1}} (1 - \delta)$.

This probability is maximized for $\delta = \frac{q_{H_1}}{q_{H_1} + 1}$.

The maximum is approximately $\frac{1}{e(q_{H_1} + 1)}$.

Ronald’s advantage in winning the IND-CPA game is therefore $\frac{\epsilon}{e(q_{H_1} + 1)}$.

If Bob is known to be the targeted victim at the beginning, all $H_1$ queries can be answered appropriately, and Ronald never aborts (selective-ID or IND-sID security).
Let $B$ be a PPT IND-CPA adversary.

Then, there exists a PPT algorithm $C$ to solve the bilinear Diffie–Hellman problem.

$C$ takes $P, uP, vP, wP$ as inputs, and returns $D = e(P, P)^{uvw}$.

$C$ consists of $B$ and Ronald (no external oracle $O$ now).

All interactions are between $B$ and Ronald.

System parameters $G, G_3, r, e, P, P_{PKG}, n, H_1$ are public.

Bob is the targeted victim from the beginning.

$C$ sets and publicizes $P_{PKG} = uP$ and $P_{Bob} = vP$.

The master secret is therefore $u$.

Bob’s private key $D_{Bob} = uP_{Bob} = uvP$ is unknown.

$H_2$ is now a random oracle to $B$. 

Identity-Based Key Exchange (IBKE)  
Identity-Based Encryption (IBE)  
Identity-Based Signatures (IBS)  

The Reduction Mechanism

\[ D = e(P, P)^{uvw} \]

\[ C = \mathcal{E}(M_b) \]

Hash Queries

\[ M_0, M_1 \]

\[ b' \]
Handling $H_2$ Queries

- Ronald maintains an $H_2$-table of $(Q, W)$ pairs ($W = H_2(Q)$).
- Suppose that a query $H_2(Q)$ comes.
- If some $(Q, W)$ is found in the $H_2$-table, $W$ is returned as $H_2(Q)$.
- Otherwise, Ronald chooses $W \in \mathcal{U}\{0,1\}^n$, stores $(Q, W)$ in his $H_2$-table, and returns $W$.
- Hash queries are not manipulated here.
Handling the IND-CPA Game

- $B$ sends two messages $M_0, M_1$ of length $n$ to Ronald.
- Ronald takes $U^* = wP$ and $V^* \in \mathcal{U}\{0, 1\}^n$, and sends the challenge ciphertext $C^* = (U^*, V^*)$ as a purported encryption of $M_b$ (for some $b \in \mathcal{U}\{0, 1\}$).
- $P_{PKG} = uP$, $P_{Bob} = vP$, and $U^* = wP$, so the mask before hashing is $e(P_{Bob}, P_{PKG})^w = e(vP, uP)^w = e(P, P)^{uvw} = D$.
- If $H_2(D) = V^* \oplus M_b$, then $C^*$ is a valid ciphertext for $M_b$.
- $B$ makes an $H_2$-query on $D$ in the post-challenge phase with very high probability, so $D$ ends up in Ronald’s $H_2$-table.
- Ronald cannot identify which is the correct $D$ (difficulty of the decisional BDH problem).
- Ronald chooses a random $(Q, W)$ entry from his $H_2$-table, and returns $W$ as $D = e(P, P)^{uvw}$.
Let the advantage of $B$ be $\varepsilon'$ for winning the IND-CPA game.

The actual $D$ is queried (to the random oracle $H_2$) with probability $\geq 2\varepsilon'$.

Let $q_{H_2}$ denote the number of $H_2$ queries.

Since an entry of the $H_2$-table is chosen at random, the advantage of $C$ is $\geq 2\varepsilon'/q_{H_2}$.
The Fujisaki–Okamoto transform converts an IND-CPA secure encryption scheme to an IND-CCA secure scheme.

Two additional hash functions $H_3 : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \mathbb{Z}_r$ and $H_4 : \{0, 1\}^n \rightarrow \{0, 1\}^n$ are used.

**Encryption** of $M \in \{0, 1\}^n$ is $(U, V, W)$.
- Compute $P_{Bob} = H_1(ID_{Bob}) \in G$.
- Choose $\sigma \in U \{0, 1\}^n$, and compute $a = H_3(\sigma, M)$.
- Compute $g = e(P_{Bob}, P_{PKG})$.
- $U = aP$, $V = \sigma \oplus H_2(g^a)$, and $W = M \oplus H_4(\sigma)$.

**Decryption** of $(U, V, W)$:
- Recover $\sigma = V \oplus H_2(e(D_{Bob}, U))$.
- Recover $M = W \oplus H_4(\sigma)$.
- Set $a = H_3(\sigma, M)$. If $U \neq aP$, return *failure*.
- Return $M$. 
A reduction similar to the IND-CPA to IND-ID-CPA security works.

Now, Ronald has to handle decryption queries like $(ID_{Usr}, U, V, W)$.

Ronald locates $(ID_{Usr}, P_{Usr}, t, c)$ in his $H_1$-table. If such an entry does not exist, it is created.

If $c = 0$, Ronald computes the private key $D_{Usr} = tP_{PKG}$, and carries out the decryption himself.

If $c = 1$, Ronald forwards the query $(ID_{Usr}, tU, V, W)$ to the external decryption oracle $O$, and relays the response of $O$ back to $A$.

Each decryption query is perfectly answered by Ronald.
Boneh–Boyen IBE

Setup Phase

- $G$ (additive) and $G_3$ (multiplicative) are groups of prime order $r$. $P$ is a generator of $G$.
- $e : G \times G \rightarrow G_3$ is a bilinear pairing map.
- Master secret key of PKG: two integers $s_1, s_2 \in \mathbb{Z}_r^*$. 
- Public key of PKG: the elements $Y_1 = s_1P$ and $Y_2 = s_2P$ of $G$.

Registration Phase

- Let $P_{Bob} \in \mathbb{Z}_r^*$ be the hashed public identity of Bob.
- The PKG generates $t \in U \mathbb{Z}_r^*$, and computes $D = (P_{Bob} + s_1 + s_2 t)^{-1} P \in G$.
- Bob’s private key is $(t, D)$.
- **Note:** Registration phase is randomized.
Boneh–Boyen IBE (Contd)

Encryption of $M \in G$

- Alice generates $k \in U \mathbb{Z}_r^*$.
- Alice computes $U = kP_{Bob}P + kY_1 \in G$, $V = kY_2 \in G$, and $W = M \times e(P, P)^k \in G_3$.
- The ciphertext is the triple $(U, V, W)$.

Decryption of $(U, V, W)$

- $U + tV = k(P_{Bob} + s_1 + s_2 t)P$.
- $e(U + tV, D) = e(k(P_{Bob} + s_1 + s_2 t)P, (P_{Bob} + s_1 + s_2 t)^{-1}P) = e(P, P)^k$.
- $M = W \times e(U + tV, D)^{-1}$. 
Boneh–Boyen IBE: Security

- **q-BDHI Problem**: Given $P, aP, a^2P, a^3P, \ldots, a^qP \in G$, compute $e(P, P)^{a^{-1}} (\mod r)$ (I in BDHI is Inversion).

- **Decisional q-BDHI Problem**: Given $P, aP, a^2P, a^3P, \ldots, a^qP \in G$ and $T \in G_3$, decide whether $T = e(P, P)^{a^{-1}} (\mod r)$.

- **q-BDHI assumption**: These problems are infeasible.

Boneh–Boyen encryption is IND-sID-CPA secure for a pre-selected victim (Bob) if the decisional q-BDHI assumption holds, where $q$ is the maximum number of key-extraction queries allowed.

The proof does not require random oracles.

Using a transform proposed by Canetti et al., the scheme can be made IND-sID-CCA secure.
Shamir’s IBS

Setup Phase

- PKG generates an RSA modulus $n = pq$, and computes $\phi(n) = (p-1)(q-1)$.
- PKG chooses $e \geq 3$ such that $\gcd(e, \phi(n)) = 1$, and computes $d \equiv e^{-1} \pmod{\phi(n)}$.
- PKG fixes a hash function $H : \{0,1\}^* \rightarrow \mathbb{Z}_n$.
- PKG publishes $n, e, H$.
- $p, q, \phi(n), d$ are kept secret.

Registration Phase

- PKG computes Bob’s hashed public identity $P_{Bob} = H(ID_{Bob})$.
- Bob’s private key: $D_{Bob} \equiv P_{Bob}^d \pmod{n}$.
Shamir’s IBS (Contd)

Signature Generation

- Bob chooses $x \in U \mathbb{Z}_n$.
- Bob computes $s \equiv x^e \pmod{n}$ and $t \equiv D_{Bob} \times x^{H(s, M)} \pmod{n}$.
- Bob’s signature on $M$ is the pair $(s, t)$.

Signature Verification

- $t^e \equiv P_{Bob} \times (x^e)^{H(s, M)} \equiv P_{Bob} \times s^{H(s, M)} \pmod{n}$.

Security

- A forger can generate $x, s, H(s, M)$.
- Generating the correct $t$ is equivalent to knowing $D_{Bob}$.
- Getting $D_{Bob}$ from $P_{Bob}$ is the RSA problem.
Sakai–Ohgishi–Kasahara (SOK) IBS

Setup Phase

- $G$ (additive) and $G_3$ (multiplicative) are groups of prime order $r$. $P$ is a generator of $G$.
- $e : G \times G \rightarrow G_3$ is a bilinear pairing map.
- Master secret key of PKG: $s \in U \mathbb{Z}_r^*$.
- Public key of PKG: $P_{PKG} = sP \in G$.
- $H : \{0,1\}^* \rightarrow G$ is a public hash function.

Registration Phase

- Bob’s public key: $P_{Bob} = H(ID_{Bob}) \in G$.
- Bob’s private key: $D_{Bob} = sP_{Bob} \in G$. 
Signature Generation

- Bob chooses $d \in U \mathbb{Z}_r$, and computes $U = dP \in G$.
- Bob also computes $h = H(P_{Bob}, M, U) \in G$ and $V = D_{Bob} + dh \in G$.
- Bob's signature on $M$ is $(U, V)$.

Signature Verification

\[
e(P, V) = e(P, D_{Bob} + dh) \\
= e(P, sP_{Bob} + dh) \\
= e(P, sP_{Bob})e(P, dh) \\
= e(sP, P_{Bob})e(dP, h) \\
= e(P_{PKG}, P_{Bob})e(U, H(P_{Bob}, M, U)).
\]
References

Thank You

Contact

abhij@cse.iitkgp.ernet.in, SadTijihba@gmail.com
http://cse.iitkgp.ac.in/~abhij/