A group is a set $G$ along with a binary op. $\cdot$ for which the following conditions hold:

1. **Closure:**
   \[ \forall g, h \in G, \quad g \cdot h \in G \]

2. **Existence of Identity:**
   \[ \exists e \in G \quad \forall g \in G, \quad e \cdot g = g = g \cdot e \]

3. **Existence of Inverses:**
   \[ \forall g \in G, \quad \exists h \in G \quad g \cdot h = h \cdot g = e \]

4. **Associativity:**
   \[ \forall g_1, g_2, g_3 \in G, \quad (g_1 \cdot g_2) \cdot g_3 = g_1 \cdot (g_2 \cdot g_3) \]

When $G$ has a finite number of elements, we denote it as $|G| \geq$ order of the group.
We call a group, an Abelian group if

**Commutativity:**

\[ \forall g, h \in G \implies gh = hg \]

\((\mathbb{Z}_n, \cdot \mod n)\) is an Abelian group under multiplication modulo \(n\)

\[ \mathbb{Z}_n^* = \{ a \mid a \in \{1, \ldots, n-1\} \land \gcd(a, n) = 1 \} \]

**Proof closure:**

If \( a \in \mathbb{Z}_n^* \) and \( b \in \mathbb{Z}_n^* \),

\[ a \cdot b \mod n \in \mathbb{Z}_n^* \]

\[ \gcd(a, n) = 1 \land \gcd(b, n) = 1 \implies \gcd(ab, n) = 1 \]

\[ \implies \gcd(a \cdot b \mod n, n) = 1 \]

**Proof of identity element:**

\( 1 \) is the identity element

**Proof of commutativity:**

\( a \cdot b = b \cdot a \mod n \), so multiplication mod \( n \) is commutative
Proof of associativity:

\[ a \cdot (b \cdot c) = (a \cdot b) \cdot c \pmod{n} \]

Proof of existence of inverse:

\( \forall a \in \mathbb{Z}_n^+, \exists a^{-1} \text{ such that} \)

\[ a \cdot a^{-1} = 1 \pmod{n} \]

If \( \gcd(a, n) = 1 \), \( \exists a^{-1} \text{ such that} \)

\[ a \cdot a^{-1} = 1 \pmod{n} \]

\[ a \cdot x = 1 \pmod{n} \]

\( \Rightarrow \) if \( \gcd(a, n) = 1 \) then

\[ 2 \cdot x \cdot y \text{ such that } a \cdot x + n \cdot y = 1 \]

\[ \Rightarrow a \cdot x = 1 - n \cdot y \]

\[ \Rightarrow a \cdot x = 1 \pmod{n} \]
Let $a, b \in \mathbb{F}_s$ and $4a^3 + 27b^2 \neq 0$

A non-singular elliptic curve is the set $E$ of solutions $(x, y) \in \mathbb{F}_s^2$ to the equation

$$y^2 = x^3 + ax + b$$

plus special point $O$ called the point at infinity

Example: $y^2 = x^3 - 4x$
Define binary op \( + \) to make \((E,+)\) an Abelian group. \((O)\) is the identity element.

Suppose \( P, Q \in E \) where \( P = (x_1, y_1) \), \( Q = (x_2, y_2) \).

Cases:
1) \( x_1 \neq x_2 \)
2) \( x_1 = x_2 \) \& \( y_1 = -y_2 \)
3) \( x_1 = x_2 \) \& \( y_1 = y_2 \)

\[ P + Q = R \text{ where } R = (x_3, y_3) \]

\[ s + \begin{cases} x_3 = \frac{x_1^2 - x_1 - x_2}{y_2 - y_1} \quad y_3 = \lambda (x_1 - x_3) - y_1 \\ \lambda = \frac{y_2 - y_1}{x_2 - x_1} \end{cases} \]

\( (x_1, y_1) + (x_2, y_2) = y = \lambda x + c \)

\( (\lambda x + c)^2 = x^3 + ax + b \)

Case 2: \( (x_1, y_1) + (x_1, -y_1) = O \)
Case 3: \((x_1, y_1) + (x_1', y_1') = (x_3, y_2)\)

\[
x_3 = \lambda^2 - 2x_1 \\
y_3 = \lambda (x_1 - x_3) - y_1 \\
\lambda = \frac{3x_1^2 + 9}{2y_1}
\]

1) Addition is closed on the set \(E\)

2) \(\Pi\) is commutative

3) \(0\) is an identity element

4) Every point on \(E\) has an inverse. 
\(p = (x_1, y) \in E\) \(p^{-1} = (x_1, -y) \in E\)

5) Associativity is satisfied.
Let $p > 3$ be prime. $E \subset \mathbb{F}_p$ is the set of solutions $\{ (x,y) \in \mathbb{F}_p \mid y^2 = x^3 + ax + b \pmod{p} \}$, where $a, b \in \mathbb{F}_p$ and $4a^3 + 27b^2 \neq 0 \pmod{p}$, plus a special point $\mathcal{O}$ called the point at infinity.

\[ P = (x_1, y_1), \quad Q = (x_2, y_2) \]

If $x_1 = x_2$ and $y_1 = -y_2$, then

\[ P + Q = \mathcal{O} \]

Else

\[ x_3 = \lambda^2 - x_1 - x_2 \pmod{p}, \quad y_3 = \lambda (x_1 - x_3) - y_1 \pmod{p} \]

\[ \lambda = \begin{cases} \frac{y_2 - y_1}{x_2 - x_1} \pmod{p} & \text{if } P \neq Q \\ \left(3x_1^2 + a\right)^{-1} \pmod{p} & \text{if } P = Q \end{cases} \]
Example 2: $y^2 = x^3 + x + 6$ over $\mathbb{F}_{11}$

<table>
<thead>
<tr>
<th>$x$</th>
<th>$x^3 + x + 6 \mod 11$</th>
<th>$QR$?</th>
<th>$y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>6</td>
<td>NO</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>8</td>
<td>NO</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>YES</td>
<td>$(4, 7)$</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>YES</td>
<td>5/6</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
<td>NO</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>YES</td>
<td>2/9</td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td>NO</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>4</td>
<td>YES</td>
<td>2/9</td>
</tr>
<tr>
<td>8</td>
<td>9</td>
<td>YES</td>
<td>3/8</td>
</tr>
<tr>
<td>9</td>
<td>7</td>
<td>NO</td>
<td></td>
</tr>
</tbody>
</table>
The order of $E$ (including 0) is 13. Any group of prime order is cyclic. This means you could represent each element of $E$ as $k \alpha$ where $\alpha \in E$ for some generator of the group $G$.

$2 \alpha = (2,7)$

$2 \alpha = (2,7) + (2,7) = (5,2)$

$3 \alpha = (8,3) = (2*2) + (2,7) + (2,7) = (5,2) + (2,7)$

$4 \alpha = (10,2)$

$5 \alpha = (3,6)$

$6 \alpha = (7,8)$, $7 \alpha = (9,2)$

$8 \alpha = (3,5)$, $9 \alpha = (0,8)$

$10 \alpha = (8,8)$
\[ 11 \alpha = (5, 9) \quad 12 \alpha = (2, 4) \quad 0. \alpha = 0 \]

I can represent \((7, 2)\) as \(7 \alpha\)

ElGamal over \(\mathbb{Z}_p\)
- \(a\) is secret key
- \(g^a \mod p\) is publicly
- \(g\) is generator

\[ E(m) = (g^b, M \cdot (g^a)^b) = c \]

\[ D(c) = U \cdot (U^a)^{-1} = (M \cdot g^{ab}) \cdot ((g^a)^{-1}) = M \]

Analog of ElGamal on \(EC\)

Given \(E\) over \(\mathbb{Z}_p\), choose random \(s \in \mathbb{Z}_p\)

Publish \(sB\) where \(B\) is a generator of \(E\)
given \( m \), embed \( m \) to \( E \), let's call it \( P_m \)

\[
E(P_m) = \langle kB, \quad P_m + k(sb) \rangle = C
\]

random

\[
D(C) = D(kB, C_2) = C_2 - s(kB)
\]

\[
= P_m + k(csB) - s(kB)
\]

\[
= P_m
\]

An elliptic curve \( E \) defined over \( \mathbb{F}_p \) will have roughly \( p \) points.

More precisely

\[
p + 1 - 2\sqrt{p} \leq |E| \leq p + 1 + 2\sqrt{p}
\]
First \( 1 \leq n \leq M \) choose \( k \) s.t.
\[ k, n \leq p \]

for \( (f=1; f<k; f++) \)
\[ m^1 = m^k + 5 \]
\[ y^1 = m^{12} + am^{12} + 6 \mod p \]
if \( y^1 \) is QR then
return \((x, \sqrt{y}) \mod p\)

goal: Create PK with following four algorithms to realize IBE

\{ params, mkey \} \leftarrow setup(k) \( \{ k \} \)

\# description of message space
\# \# Space
mkey will be kept secret.
Extract:
\[ d \leftarrow \text{Extract}(\text{params}, \text{Mkey}, \text{ID}) \]
\[ \text{any string} \]
\[ \text{private key for ID} \quad \text{ID} \in \mathbb{Z}_3^* \]

Encrypt:
\[ c \leftarrow \text{Enc}(\text{ID}, \text{params}, m) \]
\[ c \in \mathbb{C} \]
\[ m \in \mathbb{M} \]

Decrypt:
\[ m \leftarrow \text{Dec}(\text{params}, c, d) \]
\[ \text{private key for ID} \]
\[ \forall m \in \mathbb{M} \quad \text{Dec}(\text{params}, \text{Enc}(\text{params}, \text{ID}, m), d) = m \]

security game ?
setup: the challenger takes a security parameter \( k \) and runs setup algorithm. Keeps \( Mkey \) secret, sends params to adversary.

Phase 1: Adversary issues \( q_i \in \mathbb{Z}^m \) each \( q_i \) is answered as follows.

If \( q_i = \langle ID_i \rangle \)
\[ d_i \leftarrow \text{Extract}(\text{params}, Mkey, ID_i) \]
else if \( q_i = \langle ID_i, e_i \rangle \)
\[ m_e \leftarrow \text{Decrypt}(\text{params}, e_i, d_i) \]

Challenge: Adversary creates a challenge \( \langle m_0, m_1, ID \rangle \) where \( m_0, m_1 \in M \)
ID has not been queried before.
Challenger responds with
\[ b \leftarrow \mathbb{Z}_{poly(1)} \]
\( \leftarrow \text{Enc}(\text{params}, ID, \mathsf{m}_0) \)

**Phase 2:** Adv. issues \( q_{m+1}, \ldots, q_n \) such that each \( q_i \) is one of the following:
- \( \langle ID_i \rangle \) where \( ID_i \in \mathcal{ID} \)
- \( \langle ID_i, c_f \rangle \) where \( \langle ID_i, c_f \rangle \notin \langle ID, c \rangle \)

**Guess:** Adv. guesses \( b' \) & wins if \( b' = 0 \)

Adv.'s advantage in attacking \( E \) is as the following function:

\[
\text{Adv.}(E) = \left| \Pr[b' = 0] - \frac{1}{2} \right|
\]

We say that IBE system \( E \) is semantically secure against an adaptive chosen ciphertext attack if for any poly-time IND-ID-CCA adversary \( A \)
the function $\text{Adv}_E(b)$ is negligible.

$E$ is called IND-\text{I0-CCA} secure.

Main components of IPE scheme we discuss. We will use bilinear maps.

Let $G_1, G_2$ groups of order $q$ where $q$ is a large prime number.

$\mathbb{G}: G_1 \times G_1 \rightarrow G_2$, $\mathbb{G}$ must satisfy following:

1) Bilinear: $\mathbb{G}(aP, bQ) = \mathbb{G}(P, Q)^{ab}$ for all $P, Q \in G_1, a, b \in \mathbb{Z}$

2) Non-degenerate: $\mathbb{G}$ does not map all pairs in $G_1 \times G_1$ to identity in $G_2$.

$\Rightarrow$ If $P$ is generator of $G_1$ then $\mathbb{G}(P, P)$ is a generator of $G_2$. 
3) computable: $\hat{e}(P, Q)$ is efficiently computable for any $P, Q \in E$

In practice, $E_1$ will be $E / F_p$
$E_2$ will be $F_{p^2}$

Notes on this bilinear maps

D.L. in $E_1$ is no harder than D.L. in $E_2$

D.L.: $aQ, Q$ it is hard to find $a$

\[ g \in \mathbb{F}_p, g \mod p \]

Hard to compute $a$

For $Q = \alpha P$, given $P$, we want to find $\alpha \in \mathbb{F}_q$
Let $g = \hat{e}(P,P)$ and $h = \hat{e}(Q,P)$. Then

$$h = g^x \quad \hat{e}(Q,P) = \hat{e}(P,P)^x \quad h = g^x$$

Given $g^a$, $g^b$, it is hard to compute $g^{ab}$. Computationally

D. H. assumption

$\langle g^a, g^b, g^{ab} \rangle, \langle g^a, g^b, g^c \rangle$

D. H. Key

Exchange

The Bilinear Diff. Hell Assumption (BDH)

Let $G_1, G_2$ be two groups of prime order $q$. Let $\hat{e} : G_1 \times G_1 \to G_2$ bilinear map.
Let \( p \) be a generator of \( G_1 \). BOH problem is defined as given \( \langle p, ap, bp, cp \rangle \) for some \( a, b, c \in \mathbb{Z}_q \), compute \( w = e(p, p)^{abc} \in G_2 \).

Alg. A has advantage \( \varepsilon \) in solving BOH over \( (G_1, G_2, e) \):

\[
\text{if } p \in A(\langle p, ap, bp, cp \rangle) = e(p, p)^{abc} \gtrsim \varepsilon
\]

**Note:** If you can solve CDH in \( G_1 \) or \( G_2 \), then you can solve BOH.

Assume CDH is easy in \( G_1 \); given \( ap, bp \), I can compute \( abp \) then I can compute \( abc \):

\[
w = e(abp, cp) = e(p, p)
\]
Basic Identity Based Encryption

set up: given \( k \in \mathbb{Z}^+ \)

1) run \( \mathcal{K}(k) \) to get \( (P, q, G_1, G_2, \hat{e}) \)

2) \( \text{Pub} = s \cdot P \quad s \leftarrow \mathcal{R} \mathbb{Z}_q^* \)

3) Choose \( H_1 : \Sigma_0, \mathbb{Z} \rightarrow G_1 \)
   \( H_2 : G_2 \rightarrow \Sigma_0, \mathbb{Z} \)

\( M = \Sigma_0, \mathbb{Z} \), \( C = G_1 \times \Sigma_0, \mathbb{Z} \)

params = \( \langle q, G_1, G_2, \hat{e}, P, \mathcal{H}_1, \mathcal{H}_2 \rangle \)

Extract: for any \( \Pi_0 \in \Sigma_0, \mathbb{Z} \)

1) \( \mathcal{Q}_{\Pi_0} = \mathcal{H}_1(\Pi_0) \quad (\mathcal{Q}_{\Pi_0} \in G_1) \)

2) \( \mathcal{d}_{\Pi_0} \leftarrow s \cdot \mathcal{Q}_{\Pi_0} \quad \text{Mikey} \)
\text{Encrypt}(\text{params}, m, IO) \\
\begin{align*}
1) & \quad q_{\text{IO}} \leq H_1(\text{CID}) \\
2) & \quad r \leq 2^k \\
3) & \quad \left( g_{\text{IO}} \right)^r \leq e\left( q_{\text{IO}}, \rho_{\text{pub}} \right) \leq S_2
\end{align*}

\text{Decrypt}(\text{params}, c, d_{\text{IO}}) \\
\begin{align*}
\left( c \right) & = \left< \left( \rho, M \oplus H_2\left( g_{\text{IO}}^r \right) \right) \right> \\
\left( c \right) & = \left< \left( \rho, M \oplus H_2\left( e\left( d_{\text{IO}}, U \right) \right) \right) \right>
\end{align*}

The above system is secure in $\text{IND-IND-CPA}$.

\[ e\left( d_{\text{IO}}, U \right) = e\left( sQ_{\text{IO}}, \rho \right) = e\left( q_{\text{IO}}, \rho \right)^{\text{s.r}} = e\left( q_{\text{IO}}, \rho_{\text{pub}} \right)^r = g_{\text{IO}}^r \]

\[ U = M \oplus H_2\left( g_{\text{IO}}^r \right) \]

\[ U \oplus H_2\left( e\left( d_{\text{IO}}, U \right) \right) = U \oplus H_2\left( g_{\text{IO}}^r \right) = M \]
we can use Fujisaki – Okamoto schema to convert Basic Iden to IBE secure against IND-\textit{CCA}

\[ E_{pk}^{hy} (M) = \langle E_{pk} (\sigma, H_3 (\sigma M)), H_4 (\sigma) \oplus m \rangle \]

\[ E_{pk} (M_r (r)) \text{ is one of } M \text{ using random bits of } r \]

Different Models for searching data privacy

- protect data vs protect access patterns
- searching on private-key encrypted data

user encrypts the data

encrypted data & additional data structures
are stored in the cloud

- searching public-key encrypted data
  - next class.

- PIR is urdes which data has been
  accessed. PIR requires linear time
  search time

\[ \Delta = \{ w_1, \ldots, w_D \} \parallel \text{set of all possible words} \]

- \( D_i \subseteq 2^\Delta \parallel \text{Document} \)

- \( D = \{ D_1, \ldots, D_N \} \subseteq 2^\Delta \parallel \text{Document set} \)

- \( \text{id}(D_i) \) is the identifier of Document \( D \).

- given query \( w \), \( D(w) \) is the sorted
  list of IDs for Docs that contain \( w \).

- \( (D(w_1), \ldots, D(w_n)) \) is the access
  pattern of the client
$x \in A(y)$ output of an algorithm

\[ \text{concatenation} \]

$v : N \rightarrow N$ negligible if

$\text{poly} - p(k)$ for sufficiently large $k$

$v(k) < \frac{1}{p(k)}$

---

**Model**

- **single-user:**
  
  Owner has $O = \{O_1, \ldots, O_n\}$

  sends encrypted $O$ to servers.

  and query encrypted $O$ stored on $S$.

- **multi-user:**

  \[
  \begin{cases}
  \text{owner } O, \text{ server } S \\
  \text{set of users } N, \quad G \subseteq N
  \end{cases}
  \]

  where $G$ is allowed to do search on encrypted $O$.

  \[ \not\text{ discussed in paper for more details} \]
S is honest but curious.

construction uses symm. key enc. scheme

pseudo-random permutation (e.g., AES)
pseudo-random function (e.g., Hash function)

searchable Symm. Enc. Scheme

\[ K \leftarrow \text{Keygen}(k) \quad \text{run by user} \]

\[ K \text{ is in poly.-time with respect to } k \]

\[ I \leftarrow \text{BuildIndex}(K, D) \quad \text{trapdoor value for search.} \]

\[ T_w \leftarrow \text{Trapdoor}(K, w) \]

\[ D(w) \leftarrow \text{Search}(T_w, I) \quad \text{run by server} \]
Security definition idea:

For any two adversarially consistent
urstrag with equal length and trace,
for adversary can distinguish the
view of one from the other
with non-negligibly better than
random.

This def. assume revealing search
pattern is ok.

\[ S \subseteq \text{ that is adaptive security} \]

\[ \text{Key Gen}(1^k) \]
\[
\begin{cases} 
  s \leq 2^0.13^k \\
  K \leq s \\
  \text{return } K \\
\end{cases}
\]
Build Index (K, D) a set of key-words in D

Build \( \Delta_i \) from \( D_i \)

Build \( O(w_i) \) for each \( w_i \in \Delta_i \)

For each \( w_i \in \Delta_i \)

\[
\text{for } 1 \leq s \leq |O(w_i)|
\]

// T is look-up
\[
T \left[ \prod_{s=1}^{w_i} (\xi_s) \right] = c(D_i, s)
\]

// Table of size \( m \)

// \( \Pi \) is pseudo-random permutation

Let \( m' = \sum_{w_i \in \Delta_i} |O(w_i)| \)

Let \( m = \text{size of largest document in } D \times n \)

If \( m > m' \)

for remaining \((m-m')\) entries

put random \( c(D) \) for exactly
max entries.

- Address fields would be set to random value.

\[
\text{return } I \leq T_j
\]

\[
\text{Trap door } (w) \quad \gamma \rightarrow \\
T_w \left( \prod_l (c_{wll}), \quad - \quad \prod_l (c_{wll^n}) \right) \}
\]

\[
\text{return } Tw \gamma
\]

\[
\text{search } C \left( I, Tw \right)
\]

\[
\{ \quad \text{for } 1 \leq i \leq \text{max} \quad \text{retrieve } rd = T \left[ Tw_i \right] \\
\text{return all retrieved } rd \text{s}
\}
\]
Public Key Encryption with Search (PEKS)

Goal:
Bob wants to send message to Alice.

\[ E(\text{msg}), \text{PEKS}(A_{pub}, w) \rightarrow \text{PEKS}(A_{pub}, w) \]

Goal is to enable Alice to generate \( T_w \) (trapdoor) for word \( w \) s.t.

server can learn all the messages with word \( w \) without disclosing anything else.

Formal Definition

1) Keygen(\( S \)) : Takes a security param \( S \) and generates \( A_{pub}, A_{priv} \).

2) \( \text{PEKS}(A_{pub}, w) \) produces searchable enc. \( S \).

3) \( T_w \leftarrow \text{Trapdoor}(A_{priv}, w) \)
w produces a searchable encryption of w.

4) Test (Apub, S, Tw) : given Alice's Apub, searchable enc. \( S = PEKS(Apub, w) \) and Tw outputs yes if w in Tw else outputs no.

---

Security Definition:

1) Challenger runs the key gen(s)
to get Apub, Apriv. and sends Apub to attacker

2) Attacker queries any w of his choice and gets Tw

3) Attacker sends \( \{ w_0, w_1 \} \) such that Tw_0 or Tw_1 never queried.
   challenger chooses \( b \in \mathbb{Z}_2 \)
   and sends Tw_b

4) Attacker asks more queries not related to Tw_0 or Tw_1
5) Attacker predicts $b'$ if $b=b'$

Attacker wins

$$\text{Adv}_A(s) = \left| \Pr \left[ b=b' \right] - \frac{1}{2} \right|$$

we say that PEKS is semantically secure against an adaptive chosen keyword attack if for any P.T. attacker $A$ the $\text{Adv}_A(s)$ is negligible.

---

$\text{PEKS} \Rightarrow \text{IBE}$

Thm.: A non-interactive PEKS that is semantically secure against an adaptive chosen keyword attack implies the existence of a chosen ciphertext secure IBE (IND-IO-CCA).
Proof Idea:

given a PEKS (keygen, PEKS, Trapdoor, Test)
then we construct the IBE system as follows:

1) Run the keygen for PEKS
   get $A_{pub}, A_{priv}$

2) $A_{priv}$ is the master key for the IBE scheme

3) Generate the IBE private key associated with public key $x \in \mathbb{Z}_p^{13*}$
   
   $$d_x = \begin{cases} \text{Trapdoor}(A_{priv}, x \parallel 0), \\ \text{Trapdoor}(A_{priv}, x \parallel 1) \end{cases}$$

4) Encrypt:
   
   Encrypt a bit $b \in \{0,1\}$ using public key $x \in \mathbb{Z}_p^{13*}$
\[ C = \text{PEKS}(A_{pub}, x || b) \]

5) given \( C \), use \( d_x = [d_{x_0}, d_{x_1}] \)

output 0 if \( \text{Test}(A_{pub}, C, d_{x_0}) = \text{Yes} \)

output 1 if \( \text{Test}(A_{pub}, C, d_{x_1}) = \text{Yes} \)

else output 'error'

The above scheme is IND-CCA secure.

It is believed that \( (\text{IBE} \Rightarrow \text{PEKS}) \) is not correct.

Construction of a PEKS

Use two groups \( G_1, G_2 \) of a prime order \( p \)

where both \( G_1 \) and \( G_2 \) are multiplicative groups.

and a bilinear map \( e: G_1 \times G_1 \rightarrow G_2 \)

1) \( e \) is efficiently computable
2) \textbf{bi-linear:} for any integers } x, y \in \mathbb{Z}_{p-1} \text{, } \bar{e}(g^x, g^y) = \bar{e}(g, g)^{xy}

3) \textbf{non-degenerate:} if } g \text{ is a generator of } G_1, \text{ then } \bar{e}(g, g) \text{ is a generator of } G_2.

\text{given } H_1 : \mathbb{Z}_p^{13} \rightarrow G_1
\text{ and } \mathbb{Z}_p^{13} \rightarrow \mathbb{Z}_{p'}^{log p}

\text{KeyGen(s)}
\{ 
\text{given } s, \text{ compute } p, G_1, G_2,
\text{ pick } \text{ random } z \in \mathbb{Z}_p^{*},
A_{pub} = g^z = h, A_{priv} = z
\}

\text{PEKS(} A_{pub}, w) 
\{ 
\text{+ } \left\langle \bar{e}(H_1(w), h^r) \text{ for random } r \in \mathbb{Z}_p^{*} \right\rangle
\text{ output } \left[ g^r, H_2(t) \right]
\}
Trapdoor $(A_{\text{pub}}, w) \Rightarrow (H_1(w))^x \in G_1$

Test $(A_{\text{pub}}, S, Tw) \Rightarrow [A, B]$
if $H_2(\hat{e}(Tw, A)) = B$ then output 'yes'
else output 'no'

why it works?

\[ H_1(w) = g^m \]
\[ + \leftarrow \hat{e}(H_1(w), h^r) \]
given
\[ + \leftarrow \hat{e}(g^m, g^x) = \hat{e}(g, g)^{mx} \]
\[ \Rightarrow B = H_2(\hat{e}(g, g)^{mx}) \]

Also consider $\hat{e}(Tw, A) = \hat{e}(H_1(w), g^r)$
\[ = \hat{e}(H_1(w), g)^{xr} \]
\[ = \hat{e}(g^m, g)^{xr} \]
\[ = \hat{e}(g, g)^{mxr} \]
\[ H_2 \left( \hat{e}(Tw, A) \right) = 0 \text{ if } Tw \]

is a trapdoor for \( w \).

It can be proven that above scheme is secure under \( BOH \) assumption in the random oracle model.

A construction that can use any \( PK \), assuming that the searchable keywords are limited. Also we assume that the public key scheme is such that given ciphertext, it is hard to say which \( PK \) this ciphertext is associated with. (source-indistinguishability)
When the keyword family $\Sigma$ is poly size with respect to $s$, it is easy to construct searchable encryption from any $\text{PKE}(\mathbb{G}, \mathcal{E}, \mathcal{D})$ generated Enc $\text{Qe}$.

Key gen:

for each $w \in \Sigma$, run $\mathcal{G}(s)$ and get $\text{PK}_w, \text{Priv}_w$

$A_{\text{pub}} = \{ \text{PK}_w \mid w \in \Sigma \}$

$A_{\text{priv}} = \{ \text{Priv}_w \mid w \in \Sigma \}$

3. $\mathcal{PEKE}(A_{\text{pub}}, w)$

7. Pick random $M \in \{0,1\}^s$

return $(M, \mathcal{E}[\text{PK}_w, M])$
\[ \text{Trapdoor} \left( A_{\text{priv}}^w \right) = \text{priv}^w \]

\[ \text{Test} \left( A_{\text{pub}^w}, S, T_w \right) \]
\[ \Rightarrow [A, B] \]
\[ \text{if } (D_{\text{priv}}^w, B) = D(T_w, B) = A \text{ then output } \text{Yes} \]
\[ \text{else output } \text{No}. \]