1 Review

Here’s another way to think about the encryption/decryption operations in Boneh-Franklin:

\[ e(H(ID), g^y) = e(H(ID)^y, g^s) \]

The left side corresponds to computing the blinding factor during encryption, and the right side corresponds to computing the blinding factor during decryption. During encryption, \( s \) is known, but we can only make use of \( y \) through \( g^y \). During decryption, we no longer know \( s \), but we are able to compute the same blinding factor by pairing \( g^s \) and \( H(ID)^y \), namely, \( e(H(ID), g)^y \).

To finish off, we show the success probability of breaking decisional-BDH, given an IBE attacker with advantage \( \epsilon \).

\[
Pr[\text{success}] = Pr[\text{success}|\text{abort}] Pr[\text{abort}] + Pr[\text{success}|\text{not abort}] Pr[\text{not abort}]
\]

\[
= \frac{1}{2} \left( 1 - \frac{1}{Q_H} \right) + Pr[\text{abort}] \left( \frac{1}{2} Pr[\text{success}|\text{abort} \land \gamma = 0] + \frac{1}{2} Pr[\text{success}|\text{abort} \land \gamma = 1] \right)
\]

\[
= \frac{1}{2} \left( 1 - \frac{1}{Q_H} \right) + \frac{1}{Q_H} \left( \frac{1}{2} \left( \frac{1}{2} + \epsilon \right) + \frac{1}{2} \cdot \frac{1}{2} \right)
\]

\[
= \frac{1}{2} + \frac{\epsilon}{2Q_H}
\]

Where \( Q_H \) is the number of key generation queries the attacker makes. Note that when we have guessed the wrong ID the attacker tries to attack, or when \( \gamma = 1 \) (the attacker is not given a valid encryption), the probability of success is set to \( 1/2 \).

2 Aside: Random Oracles

Why are random oracles used in cryptography? Clearly they do not correspond to anything in real life. One reason they are useful is that they are stepping stones towards creating systems that do not make use of random oracles (as we will see, the next IBE system does not invoke them). Often, proofs in the random oracle model are simpler than proofs without them. Lastly, many systems used in practice (e.g., SSL, PGP) are actually shown to be secure only in the random oracle model. When considering proofs of security, one should always be aware of the risks in using random oracles.
3 Boneh-Boyen IBE [BB04]

The first step towards IBE without random oracles was given by Dan Boneh and Xavier Boyen in 2004. The security game they use, called Selective-ID, has the following modification: the attacker specifies beforehand which ID she plans to attack. Thus, before seeing the IBE’s public parameters or the keys for any IDs, the attacker announces which ID she will use when requesting the encryption of $M_0$ or $M_1$.

3.1 Algorithms

- **Setup($\lambda$)**
  - randomly pick $g, h \in G$, and $a, b \in \mathbb{Z}_p$
  - the public parameters are: $g, h, u = g^a, e(g, g)^{ab}$ (and the hash function $f$)
  - let the hash function $f : \mathbb{Z}_p \rightarrow G$ be defined as $f(ID) = u^{ID} \cdot h$
  - the master secret key is $g^{ab}$

- **KeyGen($ID, MSK$)**
  - randomly pick $r \in \mathbb{Z}_p$
  - $K_{ID} = (K_1, K_2) = (g^{ab} \cdot f(ID)^r, g^r)$
  - think of this as encrypting the MSK using the secret $f(ID)^r$

- **Encryption($PP, M, ID$)**
  - randomly pick $s \in \mathbb{Z}_p$
  - $CT = (C_0, C_1, C_2) = (M \cdot (e(g, g)^{ab})^s, g^s, f(ID)^s)$

- **Decryption($K_{ID}, CT$)**
  - $M = C_0 \cdot e(K_1, C_1)^{-1} \cdot e(K_2, C_2)$
  - This requires a bit of explanation. Note that in order to recover the plaintext $M$, one must be able to compute the blind $e(g, g)^{abs}$. Consider the computation of $e(K_1, C_1)$:

$$
e(K_1, C_1) = e(g^{ab} \cdot f(ID)^r, g^s)$$
$$= e(g^{ab}, g^s) \cdot e(f(ID)^r, g^s)$$
$$= e(g, g)^{abs} \cdot e(f(ID), g)^{rs}$$

This gives us the blind... if we could compute $e(f(ID), g)^{rs}$. However, that is not hard because $e(K_2, C_2) = e(g^r, f(ID)^s) = e(g, f(ID))^{rs}$. 
3.2 Randomness necessary for KeyGen?

Here we argue that it is important to choose a fresh random \( r \) for each KeyGen operation. Consider what happens if we use the same \( \tilde{r} \) each time. Suppose we have 2 keys \( K_{ID_{alice}} = (g^{ab}(u^{ID_{alice}} \cdot h)^{\tilde{r}}, g^{\tilde{r}}) \) and \( K_{ID_{bob}} = (g^{ab}(u^{ID_{bob}} \cdot h)^{\tilde{r}}, g^{\tilde{r}}) \). If we divide the first element of Bob’s key into Alice’s key, we get \( u^{(ID_{alice} - ID_{bob}) \cdot \tilde{r}} \). By raising this value to \( 1^{ID_{alice} - ID_{bob}} \), we obtain \( u^{\tilde{r}} \). This can be multiplied into \( K_{ID} \) to obtain a valid key for \( ID + 1 \). (Do you see how to obtain a valid key for any \( ID \)?)

In the next lecture, we show how to rerandomize keys. That is, given a valid key, produce the same valid key under a different randomness without using knowledge of any secret.

3.3 Proof of Security

As with Boneh-Franklin, the proof of security is with respect to decisional-BDH.

**Theorem 3.1 (Boneh-Boyen IBE)** decisional-BDH is hard \( \Rightarrow \) Boneh-Boyen IBE is CPA selective-ID secure

As usual, the proof is by contradiction. We show that an attacker who can break Boneh-Boyen IBE with \( \epsilon \) advantage can be used to distinguish decisional-BDH with non-negligible probability. The reduction (roughly) proceeds as follows:

- Receive the decisional-BDH challenge: \( g, g^a, g^b, g^c, T \), where \( T = e(g, g)^{ab} \) when \( \gamma = 0 \) and \( T = random \) when \( \gamma = 1 \)

- Since this is the selective-ID model, the IBE attacker now announces \( ID^* \), the \( ID \) she wishes to attack.

- The next step is to simulate the Setup phase for the attacker, which requires careful consideration. We need to not only utilize the decisional-BDH parameters, but also \( ID^* \). Otherwise, we aren’t really using the attacker’s power to help us solve the decisional-BDH instance.

Here is one possible setup: \( u = g^a, h = u^{-ID^*}, e(g, g)^{ab} = e(g^a, g^b) \).

Under this setting, applying the hash function to \( ID^* \) yields \( f(ID^*) = u^{ID^*} \cdot u^{-ID^*} = 1 \). This is a problem because had \( g, h, a \) (and thus \( u \)) been chosen randomly as specified by the setup algorithm, it is unlikely that \( ID^* \) hashes to the identity element. Thus, there is no guarantee that our IBE attacker will still work correctly with \( \epsilon \) advantage.

It turns out we just need to add a blind to \( h \). Randomly pick \( y \in \mathbb{Z}_p \), and use the following: \( u = g^a, h = u^{-ID^*} \cdot g^y, e(g, g)^{ab} = e(g^a, g^b) \).

In this case, \( f(ID^*) = g^y \) (which is random), while for \( ID \neq ID^* \) we have \( f(ID) = u^{ID - ID^*} \cdot g^y \).

- The next step to consider is KeyGen. This is not trivial because we need to be able to generate valid keys without knowing \( a \) or \( b \). Consider what happens if we were to use \( r = \frac{-b}{ID - ID^*} \) (we can’t actually compute this directly, because we don’t know \( b \)):

6-3
\[ K_1 = g^{ab} \cdot f(ID)^r \]
\[ = g^{ab} \cdot \left( u^{ID} \cdot u^{-ID^*} \cdot g^y \right)^{ID^{-b}ID^r} \]
\[ = g^{ab} \cdot g^{-ab} \cdot g^{ID^{-b}ID^r} \]
\[ = (g^b)^{ID^{-y}ID^r} \]

\[ K_2 \] can be computed as \( g^r = g^{ID^{-b}ID^r} = (g^b)^{ID^{-1}ID^r} \).

Since we can compute \( ID^{-y}ID^r \) and \( ID^{-1}ID^r \), we are able to generate these keys under the randomness \( r = ID^{-b}ID^r \). However, we have the same problem as in the Setup phase: there is no inherent randomness in our values of \( r! \) The solution is simple: we actually use \( r = ID^{-b}ID^r + \tilde{r} \), where \( \tilde{r} \) is fresh randomness chosen for each \( ID \).

- The last important step is to send proper parameters to the attacker when she requests the encryption of \( M_0 \) or \( M_1 \).

In the next lecture we will finish the proof.

**References**