COS 533: Advanced Cryptography — Princeton University
Lecture 12 (3/18, recorded 3/12)
Lecturer: Mark Zhandry — Scribe: Alex Valtchanov

Notes for Lecture 12

1 Last Time

Last class, we introduced Identity-Based Encryption (IBE).

Definition 1 *Identity-Based Encryption (IBE)*

Syntax:

$$\begin{aligned}(mpk, msk) &\leftarrow \texttt{Gen}(1^\lambda)\\ c &\leftarrow \texttt{Enc}(mpk, id, m)\\ sk_{id} &\leftarrow \texttt{Extract}(msk, id)\\ m &\leftarrow \texttt{Dec}(sk_{id}, c)\end{aligned}$$

Correctness:

$\forall\ (mpk, msk) \leftarrow \texttt{Gen}(1^\lambda), id, m, sk_{id} \leftarrow \texttt{Extract}(msk, id)$

$$Pr[\texttt{Dec}(sk_{id}, \texttt{Enc}(mpk, id, m)) = m] = 1$$

Security:

Let the adversary be denoted $\mathcal{A}$ and the challenger be denoted $\mathcal{C}$. The IBE security game is defined as follows:

1. $\mathcal{C}$ runs $(mpk, msk) \leftarrow \texttt{Gen}(1^\lambda)$ and sends mpk to $\mathcal{A}$.
2. $\mathcal{A}$ is allowed queries to `Extract` in which $\mathcal{A}$ sends an identity id to $\mathcal{C}$ and $\mathcal{C}$ responds with $sk_{id} \leftarrow \texttt{Extract}(msk, id)$.
3. $\mathcal{A}$ makes a challenge query by sending a new identity $id^* \notin \{id_j\}$ and two messages m_0, m_1 to $\mathcal{C}$. $\mathcal{C}$ responds with $c^* \leftarrow \texttt{Enc}(mpk, id^*, m_b)$.

4. After the challenge query, $\mathcal{A}$ can continue to make queries to `Extract` with any identity $id \neq id^*$.

$\mathcal{A}$ wins if it can guess the bit b. Security is defined as usual, where the IBE scheme is secure if, for all PPT $\mathcal{A}$, the advantage of $\mathcal{A}$ in guessing b in the IBE security experiment is negligible.

2 Identity-Based Encryption from Pairings

Let $\mathbb{G}$ be a group with order p and generator g. Assume a pairing $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_2$. Assume a hash function $H : \{0,1\}^* \to \mathbb{G}$ that maps identities to random-looking group elements.[1] Our goal is to construct an IBE scheme $(\texttt{Gen}, \texttt{Extract}, \texttt{Enc}, \texttt{Dec})$. We define the scheme as follows:

- $\texttt{Gen}(1^\lambda)$: Choose a random $a \leftarrow \mathbb{Z}_p$ and set $msk = a, mpk = g^a$.
- $\texttt{Extract}(msk, id)$: Output $sk_{id} = H(id)^a$.
- $\texttt{Enc}(mpk, id, m)$: Choose a random $b \leftarrow \mathbb{Z}_p$ and output $(g^b, e(H(id), g^a)^b \times m)$.[2]
- $\texttt{Dec}(sk_{id}, (c, d))$: Output $e(sk_{id}, c)^{-1} \times d$.

Note that

$$\begin{aligned} e(H(id), g^a)^b &= e(H(id), g)^{ab} \\ &= e(H(id)^a, g^b) \\ &= e(sk_{id}, g^b) \end{aligned}$$

Furthermore, $\texttt{Dec}(sk_{id}, \texttt{Enc}(mpk, id, m))$ is given by

$$\begin{aligned} e(sk_{id}, c)^{-1} \times d &= e(sk_{id}, g^b)^{-1} \times e(H(id), g^a)^b \times m \\ &= e(H(id)^a, g^b)^{-1} \times e(H(id), g^a)^b \times m \\ &= e(H(id), g^a)^{-b} \times e(H(id), g^a)^b \times m \\ &= m \end{aligned}$$

[1] We will not formally specify the requirements for $\mathbb{G}$, but assume that $H(id)$ is a group element for which computing the discrete log of is hard.

[2] Interpreting $e(H(id), g^a)$ as a public key for id, encryption is somewhat analogous to that of ElGamal encryption in which Bob sends the ciphertext $(g^b, g^{ab} \times m)$ (a and g^a are Alice's secret and public keys, respectively). Note that both ElGamal and this construction assume messages are group elements (in this case, of $\mathbb{G}_2$).

We can interpret $e(H(id), g^a)$ as a public key for the user with identity id that anyone can compute for themselves in order to encrypt a message to id. On the other hand, only the holder of the secret key sk_{id} can decrypt such a message as shown above.

Earlier we discussed that IBE, intuitively, compresses a set of public keys into a single master public key. In this construction, we have a set of public keys $e(H(id), g^a)$ that contain the hash of the identity id paired with the master public key g^a. These public keys are compressed into a single master key by giving out g^a and letting everyone compute $e(H(id), g^a)$ for themselves using the pairing.

Unfortunately, the assumption of a collision-resistant hash function is not enough to prove security. Instead, security is proved in the *Random Oracle model*.

Definition 2 *(Random Oracle model) A hash function H is instead assumed to be a truly random function that the adversary can only access through a polynomial number of queries.*

The Random Oracle model reflects an ideal hash function. In particular, the only information available to an adversary is simply the evaluation of the function on a polynomial number of input points of their choice.

Let $\mathcal{A}$ be an adversary for the pairing-based IBE scheme. In the Random Oracle model, for any id, $H(id)$ is a random element g^c for some c. Note that g and g^a are public. Prior to the challenge query, the adversary $\mathcal{A}$ chooses an identity id^* and can compute $H(id^*) = g^c$. After sending id^* (and two messages m_0, m_1), $\mathcal{A}$ receives a ciphertext $(g^b, e(H(id^*), g^a)^b \times m)$ (where m is either m_0 or m_1). Note that

$$\begin{aligned} e(H(id^*), g^a)^b \times m &= e(g^c, g^a)^b \times m \\ &= e(g,g)^{abc} \times m \end{aligned}$$

Hence, the adversary sees

$$\texttt{View}(\mathcal{A}) = (g, g^a, g^b, g^c, e(g,g)^{abc} \times m)$$

Finally, recall the *Bilinear Decisional Diffie-Hellman (BDDH) assumption*, which states that

$$(g, g^a, g^b, g^c, e(g,g)^{abc}) \approx_c (g, g^a, g^b, g^c, e(g,g)^d)$$

Where d is random. Assuming BDDH[3], $e(g,g)^{abc}$ is computationally indistinguishable from a random element of $\mathbb{G}_2$. Therefore, $e(g,g)^{abc} \times m_0$ is computationally indistinguishable from $e(g,g)^{abc} \times m_1$.

In our analysis of $\texttt{View}(\mathcal{A})$, we ignored the adversary's access to `Extract` queries. We will omit this detail that may be assigned in a future homework.

[3]It is widely believed that BDDH holds on pairing-based groups.

Question from Student:

How much stronger of an assumption is the Random Oracle model than a Collision-Resistant hash function?

There is much debate in the Crypto community regarding Random Oracles. Proofs in the Random Oracle model are referred to as *heuristics*, since Random Oracles do not reflect the reality of a concretely-coded hash function. In practice, it turns out that this heuristic is very important since most practical schemes make use of Random Oracles for security. To this end, practitioners generally are fond of the Random Oracle model.

On the other hand, theoreticians are possibly more split on this topic and would generally say that it would be much better to not rely on the Random Oracle heuristic. In fact, there are counterexamples of schemes, though contrived, that are secure in the Random Oracle model, but no matter how the hash function is instantiated, end up being broken. Hence, there is no hope for a generic theorem that allows any Random Oracle to be replaced by a Collision-Resistant hash function (or perhaps something even stronger).

For most non-contrived schemes, however, it appears to be the case there there are not such attacks and the Random Oracle model in some ways reflects the state-of-the-art in terms of attacking such schemes.

3 What is Known for Constructing IBE

IBE can be constructed from each of the following assumptions:

- *Pairings without the Random Oracle model*

  Unfortunately, this scheme is more complicated and less efficient than the scheme presented above.

- *Computational Diffie-Hellman (CDH) on plain groups (no pairings)*

  This was a big theoretical breakthrough from a few years ago and is highly non-trivial. The construction employs *Non-black box* use of the group. In our applications, we do not need to know the code of the group or pairing (for example, we can simply multiply or exponentiate group elements through the interface of the group). In contrast, this result uses the actual code of the group description to do interesting things.

  This result is interesting as it illustrates how to do novel things by making non-black box use of the underlying building blocks of the group. However, non-black box techniques are usually very inefficient. When using a non-black box technique, the group is first converted into a boolean circuit, and then all operations are performed gate-by-gate.

- *Factoring*

  The above result result actually implies IBE from factoring. It turns out that a group can be built from factoring where CDH is hard, but the construction is still impractical for the same reasons as above.

- *Learning With Errors (LWE)*

  This can be done by building off of the PKE scheme from LWE we saw in class. The construction requires some effort.

- *Quadratic Residuosity*

  Definition 3 *(Quadratic Residuosity) Given integers x and N, the Quadratic Residuosity problem is to decide if x is a quadratic reside modulo N, that is, if $\exists y : x \equiv y^2 \pmod{N}$.*

  The Quadratic Residuosity problem is not known to be as hard as factoring. The only known attack is to factor N and decide quadratic residuosity modulo each of the factors.

4 Constructions from IBE

4.1 IBE $\Longrightarrow$ Signatures

From a feasibility perspective, signatures can be made from One-Way Functions (OWFs)– the simplest object in Cryptography. However, building signatures from IBE can lead to very efficient signatures. As we will see, using the pairing-based IBE construction from above results in the shortest practical signatures known.

Let $(\texttt{Gen}, \texttt{Extract}, \texttt{Enc}, \texttt{Dec})$ be an IBE scheme. Our goal is to construct a signature scheme $(\texttt{Gen}', \texttt{Sign}, \texttt{Ver})$. We define the scheme as follows:

- $\texttt{Gen}'(1^\lambda) = \texttt{Gen}(1^\lambda)$. In particular the secret key for signing $sk = msk$ and the public key for signing $pk = mpk$.

- $\texttt{Sign}(sk, x)$: Output $\sigma = sk_x \leftarrow \texttt{Extract}(msk, x)$. Here, we interpret the message x as an identity. Notice that the secret key for user x constitutes a signature on x.

- $\texttt{Ver}(pk, x, \sigma)$: Choose a random message m from the message space of the IBE scheme. Let $c \leftarrow \texttt{Enc}(pk, x, m)$ and compute $m' \leftarrow \texttt{Dec}(\sigma, c)$. Output 1 iff $m = m'$.[4]

For security, we provide a proof sketch. Notice that by interpreting messages of the signature scheme as identities of the IBE scheme (and consequently signing by extracting a secret key), the security experiment for the proposed construction lines up exactly with the IBE experiment except for the challenge query.

Let $\mathcal{A}$ be an adversary for the proposed signature scheme. To construct an adversary for the IBE scheme, simply forward all queries from $\mathcal{A}$ to the IBE challenger. $\mathcal{A}$ will produce a forgery (x^*, σ^*). Take this forgery to produce a challenge query (x^*, m_0^*, m_1^*) for random messages m_0^*, m_1^* that the challenger will respond to with a ciphertext $c^* \leftarrow \texttt{Enc}(mpk, x^*, m_b^*)$. To distinguish between the two cases, try to decrypt c^* with σ^*. If σ^* was indeed a valid signature on x^*, then σ^* is a valid secret key sk_{x^*} that can decrypt c^*.

4.1.1 Application: Short signatures from pairing-based IBE

Now, we will apply the above signature scheme to the pairing-based IBE scheme from Section 2. We have the following signature scheme:

- $\texttt{Gen}(1^\lambda)$: Set $pk = g^a, sk = a$.
- $\texttt{Sign}(sk, x)$: Ouput $H(x)^a$.
- $\texttt{Ver}(pk, x, \sigma)$: Check $e(\sigma, g) \overset{?}{=} e(H(x), pk)$.

For $\texttt{Ver}(pk, x, \sigma)$, plugging in the pairing-based IBE scheme directly to the signature construction yields

$$\begin{aligned} m &\overset{?}{=} \texttt{Dec}(\sigma, \texttt{Enc}(pk, x, m)) \\ m &\overset{?}{=} \texttt{Dec}(\sigma, (g^b, e(H(x), pk)^b \times m)) \\ m &\overset{?}{=} e(\sigma, g^b)^{-1} \times e(H(x), pk)^b \times m \\ e(\sigma, g^b) &\overset{?}{=} e(H(x), pk)^b \\ e(\sigma, g) &\overset{?}{=} e(H(x), pk) \end{aligned}$$

Note that signatures in this scheme are only a single group element. Without pairing-based IBE signatures, the shortest known signatures are essentially two group elements. In practice, this constant-factor improvement is important.

[4]Note that if $\sigma = sk_x$ is actually a secret key for user x, then we should be able to decrypt an IBE message using σ.

4.2 IBE $\implies$ CCA Secure PKE

Definition 4 *(Chosen Ciphertext Attack (CCA) Security). Let the adversary be denoted $\mathcal{A}$ and the challenger be denoted $\mathcal{C}$. The CCA security game is defined as follows.*[5]

1. $\mathcal{C}$ runs $(pk, sk) \leftarrow \texttt{Gen}(1^\lambda)$ and sends pk to $\mathcal{A}$.
2. $\mathcal{A}$ is allowed queries to $\texttt{Dec}$ in which $\mathcal{A}$ sends a ciphertext c to $\mathcal{C}$ and $\mathcal{C}$ responds with $m \leftarrow \texttt{Dec}(sk, c)$.
3. $\mathcal{A}$ makes a challenge query by sending two messages m_0^*, m_1^* to $\mathcal{C}$ and $\mathcal{C}$ responds with $c^* \leftarrow \texttt{Enc}(pk, m_b^*)$.
4. After the challenge query, $\mathcal{A}$ can continue to make queries to $\texttt{Dec}$ with any ciphertext $c \neq c^*$.[6]

$\mathcal{A}$ wins if it can guess the bit b. Security is defined as usual.

Let $(\texttt{Gen}, \texttt{Extract}, \texttt{Enc}, \texttt{Dec})$ be an IBE scheme and let $(\texttt{Gen}_0, \texttt{Sign}, \texttt{Ver})$ be a one-time *strongly* secure signature scheme.[7] Our goal is to construct a public-key encryption scheme $(\texttt{Gen}', \texttt{Enc}', \texttt{Dec}')$. We define the scheme as follows:

- $\texttt{Gen}'(1^\lambda) = \texttt{Gen}(1^\lambda)$. In particular, $pk = mpk, sk = msk$.
- $\texttt{Enc}'(pk, m)$: Run $(sk_0, pk_0) \leftarrow \texttt{Gen}_0(1^\lambda)$. Interpret pk_0 as an identity of the IBE scheme. Compute $c \leftarrow \texttt{Enc}(pk, id = pk_0, m)$ and $\sigma \leftarrow \texttt{Sign}(sk_0, c)$. Output (pk_0, c, σ).
- $\texttt{Dec}'(sk, (pk_0, c, \sigma))$: Run $\texttt{Ver}(pk_0, c, \sigma)$. If verification fails, abort and output an abort message $\perp$. If verification passes, compute $sk_{pk_0} \leftarrow \texttt{Extract}(sk, pk_0)$ and output $m \leftarrow \texttt{Dec}(sk_{pk_0}, c)$.

In this construction, we sacrifice the identity-based aspect of the IBE scheme, but gain CCA security. We make use of the signature scheme by generating a fresh key-pair each time we want to decrypt (this is why we only need one-time security), and signing each message to the public key of the signature scheme during that particular run of the encryption algorithm. We will see why this is important for CCA security.

[5] This is a definition of CCA security for public-key encryption. We omit CPA queries since anyone can encrypt messages for themselves.

[6] CCA for PKE is very strong notion of security. The adversary is given the ability to encrypt any message and decrypt any ciphertext except c^* (otherwise, the adversary trivially wins). Despite this ability, the adversary still cannot learn anything about the plaintext.

[7] We have seen how to build one-time signature schemes from OWFs, and with some effort, one can achieve strong security. We will take such signature schemes for granted since IBE $\implies$ OWF.

In proving CCA security, the main difficulty is handling CCA queries. Let $\mathcal{A}$ be an adversary for the proposed PKE scheme. Suppose that the challenge ciphertext (that is sent to $\mathcal{A}$) is (pk_0^*, c^*, σ^*). Now, consider a CCA query (pk_0, c, σ) from $\mathcal{A}$. From the definition of the CCA security experiment, $(pk_0^*, c^*, \sigma^*) \neq (pk_0, c, \sigma)$. In particular, the tuples must differ in at least one place.

- *Case 1: $pk_0 \neq pk_0^*$.*

  Since the identities are not the same, we can submit an `Extract` query on identity pk_0 to the IBE challenger and get sk_{pk_0}. Then, we can decrypt (pk_0, c, σ) as normal via $\texttt{Dec}(sk_{pk_0}, c)$.

- *Case 2: $pk_0 = pk_0^*, c \neq c^*$.*

  If $\texttt{Ver}(pk_0, c, \sigma) = 1$, (pk_0, c, σ) must constitute a signature forgery since $c \neq c^*$ is a different message that is validated under the same public key $pk_0 = pk_0^*$. Hence, the security of $(\texttt{Gen}_0, \texttt{Sign}, \texttt{Ver})$ would be broken. If $\texttt{Ver}(pk_0, c, \sigma) = 0$, then we can simply respond with $\perp$.

- *Case 3: $pk_0 = pk_0^*, c = c^*, \sigma \neq \sigma^*$.*

  Invoking the *strong* security of $(\texttt{Gen}_0, \texttt{Sign}, \texttt{Ver})$, this case is identical to the case above.

Definition 5 *(Strong One-Time Security). Let the adversary be denoted $\mathcal{A}$ and the challenger be denoted $\mathcal{C}$. The strong one-time signature security game is defined as follows:*

1. $\mathcal{C}$ runs $(pk_0, sk_0) \leftarrow \texttt{Gen}_0(1^\lambda)$ and sends pk_0 to $\mathcal{A}$.
2. $\mathcal{A}$ is allowed one query to `Sign` in which $\mathcal{A}$ sends a message c to $\mathcal{C}$ and $\mathcal{C}$ responds with $\sigma \leftarrow \texttt{Sign}(sk_0, c)$.
3. $\mathcal{A}$ attempts a forgery by sending a message-signature pair (c', σ') .

$\mathcal{A}$ *wins if* $(c', \sigma') \neq (c, \sigma)$ *and* $\texttt{Ver}(pk_0, c', \sigma') = 1$[8]

The one-time signatures we saw in class are also strongly secure if the underlying function is injective, or at the very least collision resistant. Thankfully, from a practical perspective, such OWFs are not hard to instantiate. However, it is possible to build a strongly one-time secure signature scheme from a theory perspective using only the minimal assumption of a general OWF.

[8]The only difference with weak one-time security is that $c' \neq c$ instead of $(c', \sigma') \neq (c, \sigma)$. In particular, $\mathcal{A}$ also wins in the strong security experiment if it can come up with a different signature σ' for the same message $c' = c$.

5 Hierarchical IBE

In plain IBE, the master secret key msk is held by a trusted authority, and it links all the secret keys of all the users. With this paradigm, these are the only two types of users. A natural extension is to introduce a hierarchy where each user of one particular level is viewed as a trusted authority for another set of users at a lower level.

We can imagine the identities of users as a tuple which has length equal to the level of the hierarchy (level 0 is the holder of msk and has identity $\{\}$, level 1 identities are a single id, level 2 identities are a tuple (id, id'), and so on). With this construction, anyone can issue identities to anyone below them, as well as decrypt messages sent to themselves or sent to anyone below them.

We can construct HIBE from the following assumptions (much like plain IBE):

- *Pairings*
- *CDH*
- *Factoring.* This follows from CDH for the same reasons as before.
- *LWE*

With an n-level HIBE scheme, the construction above yields an $(n-1)$-level CCA secure HIBE scheme. This extends the original trade-off of IBE $\implies$ CCA secure PKE in which one “level” of identity-based encryption was lost for a gain in CCA security.

HIBE is a special case of *Attribute-Based Encryption*, which has more complex access policies (we may touch on this topic later in the course). With HIBE, any user can encrypt to any string of identities, and any user whose identity is a prefix of that string will be able to decrypt. To this end, HIBE corresponds to a prefix-based access policy.