1 Outline

Last Time: We constructed IBE using Random Oracles.

This Time: We’ll construct IBE without Random Oracles, and we’ll construct Attribute-Based Encryption.

Key Property: With high probability over \( A \in \mathbb{Z}_q^{n \times m} \), for every trapdoor \( S \) of \( A \), the following two distributions are statistically equivalent:

1. \( \{(r, u) : u \leftarrow \mathbb{Z}_q^n, r \leftarrow \text{GPV-Sample}(A, S, u, \sigma)\} \).
2. \( \{(r, u) : r \leftarrow \mathbb{D}_m, u = Ar \in \mathbb{Z}_q^n\} \).

2 IBE without Random Oracles

For each \( x \in \{0, 1\}^d \), we take a matrix \( B_x \leftarrow \mathbb{Z}_q^{n \times m} \). \( B_0 = A \), generated with trapdoor \( S \). For nonempty \( x \), \( A_x = [A|B_{x_1}| \cdots |B_{x_d}] \in \mathbb{Z}_q^{n \times (d+1)} \); this is the “tree construction” referred to last time.

Claim: Given \( A, S, \) and \( A_x = [A|B] \), we can get a trapdoor \( S_x \) for \( A_x \) which reveals nothing about \( S \). (This is what we need for IBE-Extract, as in last class)

Proof: Let the dimensions of \( A_x \) be \( n \times m' \). Choose \( m' \) linearly independent vectors \( t_i \) from \( \mathbb{D}_m, u_i = -B t_i \). Choose \( t_i \leftarrow \text{GPV-Sample}(A, S, u_i, \sigma) \), and let \( s_i' = \begin{pmatrix} t_i \\ t_i' \end{pmatrix} \). Then output \( S_x \) whose columns are the \( s_i' \). One can check that this is secure by some combination of the Key Property, the Leftover Hash Lemma, and LWE.

3 Attribute-Based Encryption

ABE is like IBE, but based on an attribute: the private key should be an encryption of a circuit that evaluates to 1 on that attribute, so you can give different people the ability to decrypt different things.

Needs four methods:

1. ABE-Setup(1^\lambda, 1^\ell) outputs (MPK, MSK).
2. ABE-Encrypt(MPK, att, msg) outputs \( ct_{att} \), contains att.
3. ABE-KeyGen(MPK, MSK, c) outputs \( sk_c \), contains c.
4. ABE-Decrypt\((ct_{\text{att}}, sk_c)\) outputs \(msg\).

Security is based on the following game: the Adversary sends an attribute and gets the master’s public key for it; then can send circuits that evaluate to 0 on that attribute and get their secret keys, then sends two messages and guesses which one was encrypted.

Correctness requires that for every attribute \(att\) and circuit \(C\) s.t. \(C(\text{att}) = 1\) and every message \(msg\), 
\[
\text{ABE-Decrypt}(\text{ABE-Encrypt}(\text{MPK}, \text{att}, msg), \text{ABE-KGen}(\text{MPK}, MSK, C)) = msg.
\]

1. ABE-Setup: for \(i \in \{1, \ldots, \ell\}\) and \(b \in \{0, 1\}\), choose \(A_{i,b} \leftarrow \mathbb{Z}_q^{n \times m}\) with trapdoor \(S_{i,b}\). Also choose \(A_{\text{out}} \leftarrow \mathbb{Z}_q^{n \times m}\). Then \((\text{MPK}, \text{MSK}) = ((\{A_{i,b}\}, A_{\text{out}}), \{S_{i,b}\})\).

2. ABE-Encrypt: choose \(s \leftarrow \mathbb{Z}_q^n\) and \(e_1, \ldots, e_\ell, e_{\text{out}} \leftarrow D_{\mathbb{Z}_q^m, \sigma}\). Let \(v_i = A^t_{i,\text{att}} + e_i\) and \(v_{\text{out}} = A^t_{\text{out}} s + e_{\text{out}} + (\frac{q}{2})msg\). Then \(ct_{\text{att}} = (\text{att}, v_1, \ldots, v_\ell, v_{\text{out}})\). Note that of the \(v\), the only one that has anything about the message is \(v_{\text{out}}\), but they're all LWE instances.

3. ABE-KeyGen: for \(i \in \{\ell + 1, \ldots, |C|\}\) and \(b \in \{0, 1\}\), choose \(A_{i,b} \leftarrow \mathbb{Z}_q^{n \times m}\) with trapdoor \(S_{i,b}\), except that \(A_{|C|,1} = A_{\text{out}}\). That is, we choose a matrix and trapdoor for every possible value of every non-input wire of the circuit. Also, for every gate \(G_k \in C\) and \(a, b \in \{0, 1\}\), compute a short \(R^k_{a,b} \in \mathbb{Z}_q^{2m \times m}\) s.t. \([A_{i,a}]R = A_{k,G_k(a,b)}\). Output \(sk_C = (C, \{R^k_{a,b}\}_{a,b \in \{0,1\}, G_k \in C})\).

4. ABE-Decrypt: for intuition, we use the matrices \(R^k_{a,b}\) to compute encryptions of the outputs of gates from encryptions of their inputs, and chase those through the circuit. Specifically, if we have \((A_{i,a}, v_i)\) and \((A_{j,b}, v_j), R^k_{a,b}, \) and \(A_{k,G_k(a,b)}\), we need \(v_k = A^t_{k,G_k(a,b)} s + e_k\). Note that 
\[
(R^k_{a,b})^t \begin{pmatrix} v_i \\ v_j \end{pmatrix} = (R^k_{a,b})^t \begin{pmatrix} A^t_{i,a} \\ A^t_{j,b} \end{pmatrix} s + (R^k_{a,b})^t \begin{pmatrix} e_i \\ e_j \end{pmatrix},
\]
which differs by only a short vector from \(A^t_{k,G_k(a,b)} s\).

[The details of this are unfinished in these notes.]