21
More obfuscation, exotic encryptions

Fully homomorphic encryption is an extremely powerful notion, but it does not allow us to obtain fine control over the access to information. With the public key you can do all sorts of computation on the encrypted data, but you still do not learn it, while with the private key you learn everything. But in many situations we want *fine grained access control*: some people should get access to some of the information for some of the time. This makes the "all or nothing" nature of traditional encryptions problematic. While one could still implement such access control by interacting with the holder(s) of the secret key, this is not always possible.

The most general notion of an encryption scheme allowing fine control is known as *functional encryption*, as was described in the previous lecture. This can be viewed as an object dual to Fully Homomorphic Encryption, and incomparable to it. For every function f, we can construct an *f-restricted decryption key* d_f that allows recovery of $f(m)$ from an encryption of m but not anything else.

In this lecture we will focus on a weaker notion known as *identity based encryption* (*IBE*). Unlike the case of full fledged functional encryption, there are fairly efficient constructions known for IBE.

21.1 SLOWER, WEAKER, LESS SECURER

In a sense, functional encryption or IBE is all about selective *leaking* of information. That is, in some sense we want to modify an encryption scheme so that it actually is "less secure" in some very precise sense, so that it would be possible to learn something about the plaintext even without knowing the (full) decryption key.

There is actually a history of cryptographic technique meant to support such operations. Perhaps the "mother" of all such "quasi encryption" schemes is the modular exponentiation operation $x \mapsto g^x$ for some discrete group $\mathbb{G}$. The map $x \mapsto g^x$ is not exactly an encryption of x- for one thing, we don't know how to decrypt it. Also, as a deterministic map, it cannot be semantically secure. Nevertheless, if x

is random, or even high entropy, in groups such as cyclic subgroup of a multiplicative group modulo some prime, we don't know how to recover x from g^x. However, given $g^{x_1}, \ldots, g^{x_k}$ and $a_1, \ldots, a_k$ we can find out if $\sum a_i x_i = 0$, and this can be quite useful in many applications.

More generally, even in the private key setting, people have studied encryption schemes such as

- **Deterministic encryption** : an encryption scheme that maps x to $E(x)$ in a deterministic way. This cannot be semantically secure in general but can be good enough if the message x has high enough entropy or doesn't repeat and allows to check if two encryptions encrypt the same object. (We can also do this by publishing a hash of x under some secret salt.)

- **Order preserving encryption**: is an encryption scheme mapping numbers in some range $\{1, \ldots, N\}$ to ciphertexts so that given $E(x)$ and $E(y)$ one can efficiently compare whether $x < y$. This is quite problematic for security. For example, given $poly(t)$ random such encryptions you can more or less know where they lie in the interval up to $(1 \pm 1/t)$ multiplicative factor..

- **Searchable encryption**: is a generalization of deterministic encryption that allows some more sophisticated searchers (such as not only exact match).

Some of these constructions can be quite efficient. In particular the system CryptDB developed by Popa et al uses these kinds of encryptions to automatically turn a SQL database into one that works on encrypted data and still supports the required queries. However, the issue of how dangerous the "leakage" can be is somewhat subtle. See this paper and blog post claiming weaknesses in practical use cases for CryptDB, as well as this response by the CryptDB authors.

While the constructions of IBE and functional encryption often use maps such as $x \mapsto g^x$ as subroutines, they offer a stronger control over the leakage in the sense that, in the absence of publishing a (restricted) decryption key, we always get at least CPA security.

21.2 HOW TO GET IBE FROM PAIRING BASED ASSUMPTIONS.

The standard exponentiation mapping $x \mapsto g^x$ allows us to compute *linear functions in the exponent*. That is, given any linear map L of the form $L(x_1, \ldots, x_k) = \sum a_i x_i$, we can efficiently compute the map $g^{x_1}, \ldots, g^{x_k} \mapsto g^{L(x_1, \ldots, x_k)}$. But can we do more? In particular, can we compute *quadratic* functions? This is an issue, as even computing the map $g^x, g^y \mapsto g^{xy}$ is exactly the Diffie Hellman problem that is considered hard in many of the groups we are interested in.

Pairing based cryptography begins with the observation that in some elliptic curve groups we can use a map based on the so called Weil or Tate pairings. The idea is that we have an efficiently computable isomorphism from a group $\mathbb{G}_1$ to a group $\mathbb{G}_2$ mapping g to $\hat{g}$ such that we can efficiently map the elements g^x and g^y to the element $\varphi(g^x, g^y) = \hat{g}^{xy}$. This in particular means that given $g^{x_1}, \ldots, g^{x_k}$ we can compute $\hat{g}^{Q(x_1,\ldots,x_k)}$ for every quadratic Q. Note that we cannot repeat this to compute, say, degree 4 functions in the exponent, since we don't know how to invert the map φ.

The **Pairing Diffie Hellman Assumption** is that we can find two such groups $\mathbb{G}_1, \mathbb{G}_2$ and generator g for $\mathbb{G}$ such that there is no efficient algorithm A that on input g^a, g^b, g^c (for random $a, b, c \in \{0, \ldots, |\mathbb{G}| - 1\}$) computes $\hat{g}^{abc}$. That is, while we can compute a quadratic in the exponent, we can't compute a cubic.

We now show an IBE construction due to Boneh and Franklin[1] how we can obtain from the pairing diffie hellman assumption an identity based encryption:

- **Master key generation:** We generate $\mathbb{G}_1, \mathbb{G}_2, g$ as above, choose a at random in $\{0, \ldots, |\mathbb{G}| - 1\}$. The master private key is a and the master public key is $\mathbb{G}_1, \mathbb{G}_2, g, h = g^a$. We let $H : \{0,1\}^* \to \mathbb{G}_1$ and $H' : \mathbb{G}_2 \mapsto \{0,1\}^\ell$ be two hash functions modeled as random oracles.

- **Key distribution:** Given an arbitrary string $id \in \{0,1\}^*$, we generate the decryption key corresponding to id, as $d_{id} = H(id)^a$.

- **Encryption:** To encrypt a message $m \in \{0,1\}^\ell$ given the public paramters and some id id, we choose $c \in \{0, \ldots, |\mathbb{G}| - 1\}$, and output $g^c, H'(id \| \varphi(h, H(id))^c) \oplus m$

- **Decryption:** Given the secret key d_{id} and a ciphertext h', y, we output $H'(id \| \varphi(d_{id}, h')) \oplus x$

Correctness: We claim that $D_{d_{id}}(E_{id}(m)) = m$. Indeed, write $h_{id} = H(id)$ and let $b = \log_g h_{id}$. Then an encryption of m has the form $h' = g^c, H'(id \| \varphi(g^a, h_{id})^c) \oplus m$, and so the second term is equal to $H'(id \| \hat{g}^{abc}) \oplus m$. However, since $d_{id} = h_{id}^a = g^{ab}$, we get that $\varphi(h', d_{id}) = \hat{g}^{abc}$ and hence decryption will recover the message. QED

Security: To prove security we need to first present a *definition* of IBE security. The definition allows the adversary to request keys corresponding to arbitrary identities, as long as it does not ask for keys corresponding to the target identity it wants to attack. There are several variants, including CCA type of security definitions, but we stick to a simple one here:

[1] The construction we show was first published in the CRYPTO 2001 conference. The Weil and Tate pairings were used before for cryptographic attacks, but were used for a positive cryptographic result by Antoine Joux in his 2000 paper getting a three-party Diffie Hellman protocol and then Boneh and Franklin used this to obtain an identity based encryption scheme, answering an open question of Shamir. At approximately the same time as these papers, Sakai, Ohgishi and Kasahara presented a paper in the SCIS 2000 conference in Japan showing an identity-based key exchange protocol from pairing. Also Clifford Cocks (who as we mentioned above in the 1970's invented the RSA scheme at GCHQ before R,S, and A did), also came up in 2001 with a different identity-based encryption scheme using the quadratic residuosity assumption.

Definition: An IBE scheme is said to be CPA secure if every efficient adversary Eve wins the following game with probability at most $1/2 + negl(n)$:

- The keys are generated and Eve gets the master public key.
- For $i = 1, \ldots, T = poly(n)$, Eve chooses an identity $id_i \in \{0,1\}^*$ and gets the key d_{id}.
- Eve chooses an identity $id^* \notin \{id_1, \ldots, id_T\}$ and two messages m_0, m_1.
- We choose $b \leftarrow_R \{0,1\}$ and Eve gets the encryption of m_b with respect to the identity id^*.
- Eve outputs b' and *wins* if $b' = b$.

Theorem: If the pairing Diffie Hellman assumption holds and H, H' are random oracles, then the scheme above is CPA secure.

Proof: Suppose for the sake of contradiction that there exists some time $T = poly(n)$ adversary A that succeeds in the IBE-CPA with probability at least $1/2 + \epsilon$ for some non-negligible ϵ. We assume without loss of generality that whenever A makes a query to the key distribution function with id id or a query to H' with prefix id, it had already previously made the query id to H. (A can be easily modified to have this behavior)

We will build an algorithm B that on input $\mathbb{G}_1, \mathbb{G}_2, g, g^a, g^b, g^c$ will output $\hat{g}^{abc}$ with probability $poly(\epsilon, 1/T)$.

The algorithm B will guess $i_0, j_0 \leftarrow_R \{1, \ldots, T\}$ and simulate A "in its belly" giving it the public key g^a, and act as follows:

- When A makes a query to H with id, then for all but the i_0^{th} queries, B will chooose a random $b_{id} \in \{0, \ldots, |\mathbb{G}|\}$ (as usual we'll assume $|\mathbb{G}|$ is prime), choose $e_{id} = g^{b_{id}}$ and define $H(id) = e_{id}$. Let id_0 be the i_0^{th} query A made to the oracle. We define $H(i_0) = g^b$ (where g^b is the input to B- recall that B does not know b.)

- When A makes a query to the key distribution oracle with id then if $id \neq id_0$ then B will then respond with $d_{id} = (g^a)^{b_{id}}$. If $id = id_0$ then B aborts and fails.

- When A makes a query to the H' oracle with input $id' \| \hat{h}$ then for all but the j_0^{th} query B answers with a random string in $\{0,1\}^\ell$. In the j_0^{th} query, if $id' \neq id_0$ then B stops and fails. Otherwise, it outputs $\hat{h}$.

- B does stops the simulation and fails if we get to the challenge part.

It might seem weird that we stop the simulation before we reach the challenge part, but the correctness of this reduction follows from the following claim:

Claim: In the actual attack game, with probability at least $\epsilon/10$ A will make the query $id_* \| \hat{g}^{abc}$ to the H' oracle, where $H(id_*) = g^b$ and the public key is g^a.

Proof: If A does not make this query then the message in the challenge is XOR'ed by a completely random string and A cannot distinguish between m_0 and m_1 in this case with probability better than $1/2$. QED

Given this claim, to prove the theorem we just need to observe that, assuming it does not fail, B provides answers to A that are identically distributed to the answers A receives in an actual execution of the CPA game, and hence with probability at least $\epsilon/(10T^2)$, B will guess the query i_0 when A queries $H(id_*)$ and set the answer to be g^b, and then guess the query j_0 when A queries $id_* \| \hat{g}^{abc}$ in which case B's output will be correct. QED

21.3 BEYOND PAIRING BASED CRYPTOGRAPHY

Boneh and Silverberg asked the question of whether we could go beyond quadratic polynomials and get schemes that allow us to compute higher degree. The idea is to get a *multilinear map* which would be a set of isomorphic groups $\mathbb{G}_1, \ldots, \mathbb{G}_d$ with generators $g_1, \ldots, g_d$ such that we can map g_i^a and g_j^b to g_{i+j}^{ab}.

This way we would be able to compute any degree d polynomial in the exponent given $g_1^{x_1}, \ldots, g_1^{x_k}$.

We will now show how using such a multilinear map we can get a construction for a witness encryption scheme. We will only show the construction, without talking about the security definition, the assumption, or security reductions.

Given some circuit $C : \{0,1\}^n \rightarrow \{0,1\}$ and some message x we want to "encrypt" x in a way that given w such that $C(w) = 1$ it would be possible to decrypt x, and otherwise it should be hard. It should be noted that the encrypting party itself does not know any such w and indeed (as in the case of the proof of Reimann hypothesis) might not even know if such a w exists. The idea is the following. We use the fact that the `Exact Cover` problem is NP complete to map C into collection of subsets $S_1, \ldots, S_m$ of the universe U (where $m, |U| = poly(|C|, n)$) such that there exists w with $C(w) = 1$ if and only if there exists d sets $S_{i_1}, \ldots, S_{i_d}$ that are a partition of U (i.e., every element in U is covered by exactly one of these sets), and moreover there is an efficient way to map w to such a partition and vice versa. Now, to encrypt the message x we take a degree d instance of multilinear maps $(\mathbb{G}_1, \ldots, \mathbb{G}_d, g_1, \ldots, g_d)$ (with all groups of size p) and choose random $a_1, \ldots, a_{|U|} \leftarrow_R \{0, \ldots, p-1\}$. We then output the ciphertext $g_1^{\prod_{j \in S_1} a_j}, \ldots, g_1^{\prod_{j \in S_m} a_j}, H(g_d^{\prod_{j \in U}^m a_j}) \oplus x$. Now, given a partition

$S_{i_1}, \ldots, S_{i_d}$ of the universe d, we can use the multilinear operations to compute $g_d^{\prod_{j \in U} a_j}$ and recover the message. Intuitively, since the numbers are random, that would be the only way to come up with computing this value, but showing that requires formulating precise security definitions for both multilinear maps and witness encryption and of course a proof.

The first candidate construction for a multilinear map was given by Garg, Gentry and Halevi. It is based on computational questions on lattices and so (perhaps not surprisingly) it involves significant complications due to *noise*. At a very high level, the idea is to use a fully homomorphic encryption scheme that can evaluate polynomials up to some degree d, but release a "hobbled decryption key" that contains just enough information to provide what's known as a *zero test*: check if an encryption is equal to zero. Because of the homomorphic properties, that means that we can check given encryptions of $x_1, \ldots, x_n$ and some degree d polynomial P, whether $P(x_1, \ldots, x_d) = 0$. Moreover, the notion of security this and similar construction satisfy is rather subtle and indeed not fully understood. Constructions of indistinguishability obfuscators are built based on this idea, but are significantly more involved than the construction of a witness encryption. One central tool they use is the observation that FHE reduces the task of obfuscation to essentially obfuscating a decryption circuit, which can often be rather shallow. But beyond that there is significant work to be done to actually carry out the obfuscation.