U.C. Berkeley — CS276: Cryptography Handout N17
Luca Trevisan March 17, 2009

Notes for Lecture 17

Scribed by Matt Finifter, posted April 8, 2009

Summary

Today we begin to talk about public-key cryptography, starting from public-key encryption.

We define the public-key analog of the weakest form of security we studied in the private-key setting: message-indistinguishability for one encryption. Because of the public-key setting, in which everybody, including the adversary, has the ability to encrypt messages, this is already equivalent to CPA security.

We then describe the El Gamal cryptosystem, which is message-indistinguishable (and hence CPA-secure) under the plausible *Decision Diffie-Hellman* assumption.

1 Public-Key Cryptography

So far, we have studied the setting in which two parties, Alice and Bob, share a secret key K and use it to communicate securely over an unreliable channel. In many cases, it is not difficult for the two parties to create and share the secret key; for example, when we connect a laptop to a wireless router, we can enter the same password both into the router and into the laptop, and, before we begin to do online banking, our bank can send us a password in the physical mail, and so on.

In many other situations, however, the insecure channel is the only communication device available to the parties, so it is not possible to share a secret key in advance. A general problem of private-key cryptography is also that, in a large network, the number of required secret keys grows with the *square* of the size of the network.

In public-key cryptography, every party generates two keys: a secret key SK and a public key PK. The secret key is known only to the party who generated it, while the public key is known to everybody.

(For public-key cryptosystems to work, it is important that everybody is aware of, or has secure access to, everybody else's public key. A mechanism for the secure exchange of public keys is called a *Public Key Infrastructure* (PKI). In a network model in which adversaries are *passive*, meaning that they only eavesdrop on communication, the parties can just send each other's public keys over the network. In a network that has *active* adversaries, who can inject their own packets and drop other users' packets, creating a public-key infrastructure is a very difficult problem, to which we may return when we talk about network protocols. For now we assume that either the adversary is passive or that a PKI is in place.)

As in the private-key setting, we will be concerned with two problems: *privacy*, that is the communication of data so that an eavesdropper can gain no information about it, and *authentication*, which guarantees to the recipient the identity of the sender. The first task is solved by public-key *encryption* and the second task is solved by *signature* schemes.

2 Public Key Encryption

A public-key encryption scheme is defined by three efficient algorithms (G, E, D) such that

- G takes no input and outputs a pair of keys (PK, SK)
- E, on input a public key PK and a plaintext message m outputs a ciphertext $E(PK, M)$.

  (Typically, E is a probabilistic procedure.)
- D, on input a secret key SK and ciphertext C, decodes C. We require that for every message m

$$\mathop{\mathbb{P}}_{\substack{(PK,SK)=G() \\ \text{randomness of } E}} [D(SK, E(PK, m)) = m] = 1$$

A basic definition of security is message-indistinguishability for one encryption.

Definition 1 *We say that a public-key encryption scheme (G, E, D) is (t, ϵ) message-indistinguishable if for every algorithm A of complexity $\leq t$ and for every two messages m_1, m_2,*

$$\left| \mathop{\mathbb{P}}_{\substack{(PK,SK)=G() \\ \textit{randomness of } E}} [A(PK, E(PK, m_1)) = 1] \right.$$

$$\left. - \underset{\substack{(PK,SK)=G() \\ \textit{randomness of } E}}{\mathbb{P}} [A(PK, E(PK, m_2)) = 1] \right| \leq \epsilon$$

(From now on, we will not explicitly state the dependance of probabilities on the internal coin tosses of E, although it should always be assumed.)

Exercise 1 *Formalize the notion of CPA-security for public-key encryption. Show that if (G, E, D) is (t, ϵ) message indistinguishable, and $E(\cdot, \cdot)$ is computable in time $\leq r$, then (G, E, D) is also $(t/r, \epsilon)$ CPA-secure.*

3 The Decision Diffie-Hellman Assumption

Fix a prime p and consider the group $\mathbb{Z}_p^*$, which consists of the elements in the set $\{1, \ldots, p-1\}$, along with the operator of multiplication mod p. This is a group because it includes 1, the operation is associative and commutative, and every element in $\{1, \ldots, p-1\}$ has a multiplicative inverse mod$p$ if p is prime.

It is a theorem which we will not prove that $\mathbb{Z}_p^*$ is a *cyclic group*, that is, there exists a $g \in \{1, \ldots, p-1\}$ such that $\{g^1, g^2, \ldots, g^{p-1}\}$ is the set of all elements in the group. That is, each power of g generates a different element in the group, and all elements are generated. We call g a *generator*.

Now, pick a prime p, and assume we have a generator g of $\mathbb{Z}_p^*$. Consider the function that maps an element x to $g^x \mod p$. This function is a bijection, and its inverse is called the discrete log. That is, given $y \in \{1, \ldots, p-1\}$, there is a unique x such that $g^x \mod p = y$. x is the discrete logarithm of $y \mod p$.

It is believed that the discrete log problem is hard to solve. No one knows how to compute it efficiently (without a quantum algorithm). $\mathbb{Z}_p^*$ is but one family of groups for which the above discussion applies. In fact, our discussion generalizes to any cyclic group. Since the points on the elliptic curve (along with the addition operator) form a cyclic group, this generalization also captures elliptic curve cryptography.

While computing the discrete log is believed to be hard, modular exponentiation is efficient in $\mathbb{Z}_p^*$ using binary exponentiation, even when p is very large. In fact, in any group for which multiplication is efficient, exponentiation is also efficient, because binary exponentiation uses only $O(\log n)$ multiplications.

For the construction of the public-key cryptosystem that we will present, we will actually need an assumption slightly stronger than the assumption of the hardness of the discrete log problem. (As we shall see in a later class, this assumption is false for $\mathbb{Z}_p^*$, but it is believed to be true in other related groups.)

Definition 2 (Decision Diffie-Hellman Assumption) *A distribution $\mathcal{D}$ over triples $(\mathbb{G}, g, q)$, where $\mathbb{G}$ is a cyclic group of q elements and g is a generator, satisfies the (t, ϵ) Decision Diffie-Hellman Assumption if for every algorithm A of complexity $\leq t$ we have*

$$\left| \underset{\substack{(\mathbb{G},g,q)\sim\mathcal{D} \\ x,y,z\sim\{0,\ldots,q-1\}}}{\mathbb{P}} [A(\mathbb{G}, g, q, g^x, g^y, g^z) = 1] - \underset{\substack{(\mathbb{G},g,q)\sim\mathcal{D} \\ x,y\sim\{0,\ldots,q-1\}}}{\mathbb{P}} [A(\mathbb{G}, g, q, g^x, g^y, g^{xy}) = 1] \right| \leq \epsilon$$

Note that the El Gamal assumption may be plausibly satisfied even by a *fixed* group $\mathbb{G}$ and a fixed generator g.

4 El Gamal Encryption

The El Gamal encryption scheme works as follows. Let $\mathcal{D}$ be a distribution over $(\mathbb{G}, g, q)$ that satisfies the Decision Diffie-Hellman assumption:

- $G()$ samples $(\mathbb{G}, g, q)$, and picks a random number $x \in \{0, \ldots, q-1\}$.
  - $PK = (\mathbb{G}, g, q, g^x)$
  - $SK = (\mathbb{G}, g, q, x)$
- $E((\mathbb{G}, g, q, a), m)$:
  - pick at random $r \in \{0, \ldots, q-1\}$
  - output $(g^r, a^r \cdot m)$
- $D((\mathbb{G}, g, q, x), (c_1, c_2))$
  - Compute $b := c_1^x$
  - Find the multiplicative inverse b' of b
  - output $b' \cdot c_2$

The decryption algorithm works as follows. c_1 is g^r (as returned by E), so $b = g^{rx}$. c_2, as returned by E, is $a^r \cdot m$, where a is g^x. This means that $c_2 = g^{rx} \cdot m$. We see that $c_2 = b \cdot m$, which is why multiplying c_2 by b^{-1} correctly yields m.

Theorem 3 *Suppose $\mathcal{D}$ is a distribution that satisfies the (t, ϵ) Decision Diffie-Hellman assumption and that it is possible to perform multiplication in time $\leq r$ in the groups $\mathbb{G}$ occurring in $\mathcal{D}$.*

Then the El Gamal cryptosystem is $(t - r, 2\epsilon)$ message-indistinguishable.

Proof: Let A be an algorithm of complexity $\leq t - r$ and fix any two messages m_1, m_2. We want to prove

$$|\,\mathbb{P}[A(\mathbb{G}, g, q, g^x, g^r, g^{xr} \cdot m_1) = 1] - \mathbb{P}[A(\mathbb{G}, g, q, g^x, g^r, g^{xr} \cdot m_2) = 1]| \leq 2\epsilon$$

(From now, we shall not write the dependency on $\mathbb{G}, g, q$.)

We utilize a variant of the encryption algorithm that uses a random group element g^y (instead of g^{xr}) as the multiplier for m.[1]

$$|\,\mathbb{P}[A(g^x, g^r, g^{xr} \cdot m_1) = 1] - \mathbb{P}[A(g^x, g^r, g^{xr} \cdot m_2) = 1]| \tag{1}$$

$$\leq |\,\mathbb{P}[A(\mathbb{G}, g, q, g^x, g^r, g^{xr} \cdot m_1) = 1] - \mathbb{P}[A(g^x, g^r, g^y \cdot m_1) = 1]| \tag{2}$$

$$+|\,\mathbb{P}[A(g^x, g^r, g^y \cdot m_1) = 1] - \mathbb{P}[A(g^x, g^r, g^y \cdot m_2) = 1]| \tag{3}$$

$$+|\,\mathbb{P}[A(g^x, g^r, g^{xr} \cdot m_2) = 1] - \mathbb{P}[A(g^x, g^r, g^y \cdot m_2) = 1]| \tag{4}$$

Each of the expressions in (2) and (4) is $\leq \epsilon$ due to the (t, ϵ) Decision Diffie-Hellman Assumption. There is an extra factor of m_1 or m_2, respectively, but the D.D.H. still holds in this case. Informally, multiplying a group element that looks random by a fixed element yields another random-looking element. We can formalize this as follows:

We claim that if G, g, q satisfies the (t, ϵ) Decision Diffie-Hellman Assumption, and r is an upper bound to the time it takes to compute products in G, then for all group elements m and for all algorithms A of complexity $\leq t - r$

$$|\,\mathbb{P}[A(g^x, g^y, g^{xy} \cdot m) = 1] - \mathbb{P}[A(g^x, g^y, g^z \cdot m) = 1]| \leq \epsilon$$

[1]This would not actually function as an encryption algorithm, but we can still consider it, as the construction is well-defined.

To prove this claim, suppose to the contrary that there exists an algorithm A of complexity $\leq t - r$ and a group element m such that the above difference is $> \epsilon$.

Let A' be an algorithm that on input (G, g, q, a, b, c) outputs $A(G, g, q, a, b, c \cdot m)$. Then A' has complexity $\leq t$ and

$$\begin{aligned} & |\,\mathbb{P}[A'(g^x, g^y, g^{xy}) = 1] - \mathbb{P}[A'(g^x, g^y, g^z) = 1]| \\ = \;& |\,\mathbb{P}[A(g^x, g^y, g^{xy} \cdot m) = 1] - \mathbb{P}[A(g^x, g^y, g^z \cdot m) = 1]| \\ > \;& \epsilon \end{aligned}$$

which contradicts the (t, ϵ) Decision Diffie-Hellman Assumption.

Next, we consider (3). This is an instance of "perfect security," since distinguishing between m_1 and m_2 requires distinguishing two completely random elements. (Again, we use the fact that multiplying a random element by a fixed element yields a random element.) Thus, the expression in line (3) is equal to 0.

This means that (1) is at most 2ϵ. $\square$