Public-key Encryption

Recommended reading:
KL Sections 11.1, 11.2.1, 11.4.1

In this lecture we will see a fundamentally different type of encryption than what we have encountered so far.

I find it extremely surprising that this different type of encryption can be realized (under reasonable hardness assumptions)!!

The story so far: Symmetric-key Crypto

Before communicating privately, Alice and Bob must generate a shared private key. This requires that they meet beforehand or that they run something like Diffie-Hellman key agreement.

The story today: Public-key Crypto

Alice generates a pair of keys (sk, pk) where sk is the secret decryption key and pk is the public encryption key. For example, pk can be posted on a public website.

Anyone with access to pk can encrypt messages to Alice, but only Alice (who knows sk) can decrypt these messages. This is asymmetric encryption.

First we must define a public-key encryption (PKE) scheme formally.

Def (PKE scheme): A PKE scheme is a tuple of algorithms (Gen, Enc, Dec) such that:

- Gen is a PPT algorithm that on input 1^n outputs a pair of keys (sk, pk).
- Enc is a PPT algorithm that on input a message m and a key k outputs the ciphertext $Enc(k, m)$.

→ Dec is a deterministic polynomial time algorithm that on input a ciphertext c and a key k outputs the decryption $\mathrm{Dec}(k, c)$.

The most basic property we require is correctness:

For any message m,

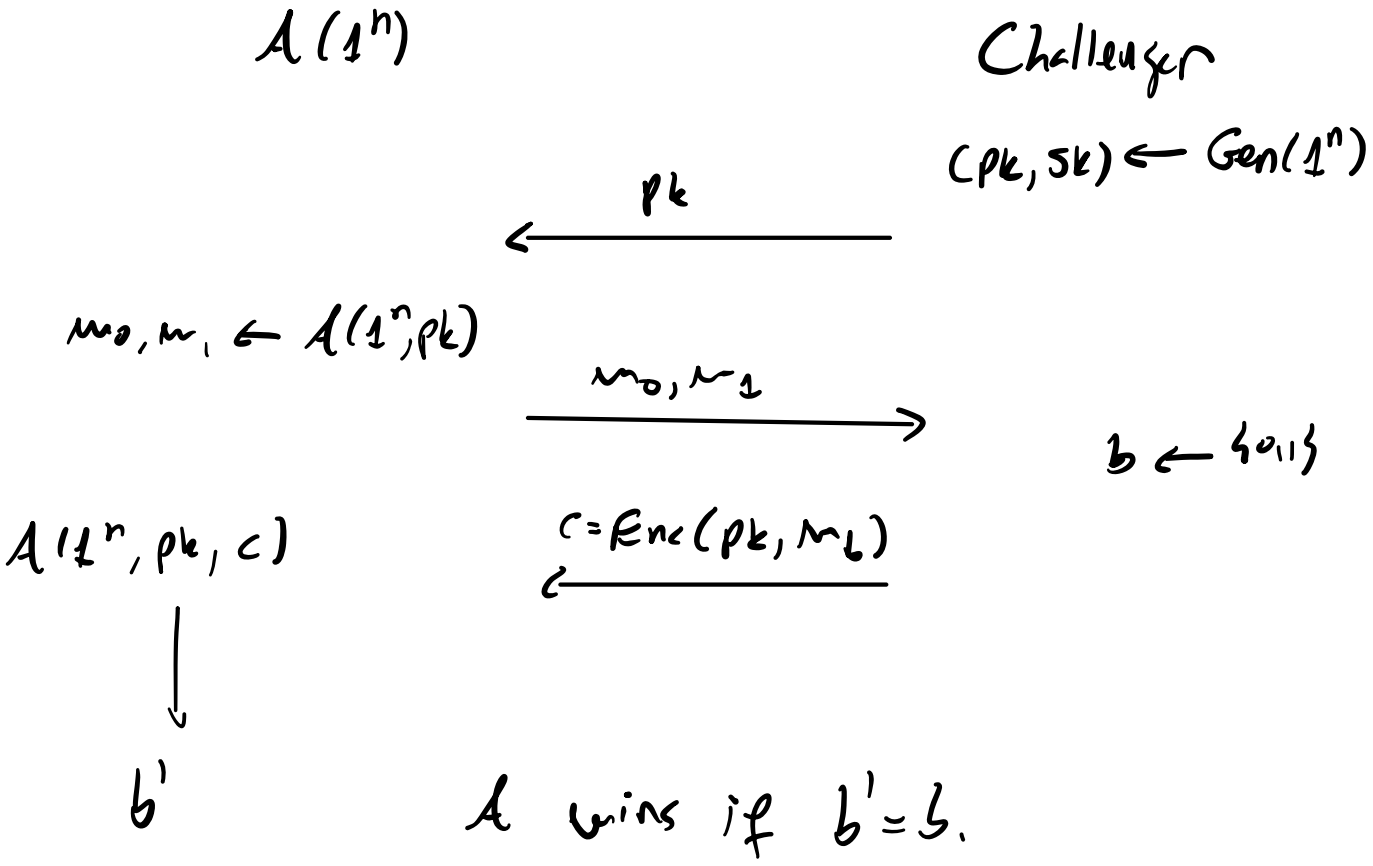
$$\Pr_{(sk, pk) \leftarrow \mathrm{Gen}(1^n)} \left(\mathrm{Dec}(sk, \mathrm{Enc}(pk, m)) = m \right) = 1.$$

Now we need to define an appropriate notion of security for PKE schemes.

Intuition: An adversary that only has access to the public key pk should not be able to distinguish between encryptions of any two messages.

Def (CPA-secure PKE):

We say that (Gen, Enc, Dec) is a CPA-secure PKE scheme if for any PPT adversary A there is a negligible function $\varepsilon(n)$ such that $A(1^n)$ wins the game below with probability at most $\frac{1}{2} + \varepsilon(n)$:



A wins if $b' = b$.

Note that since A knows pk, they can compute encryptions of arbitrary messages of their choice. So, indeed, this definition actually corresponds to CPA-security and not eavesdropping security.

Homework:

- There are no CPA-secure PKE schemes with deterministic encryption.
- There are no information-theoretic CPA-secure PKE schemes.

ElGamal encryption — PKE from DDH

We will now see our first PKE scheme, due to ElGamal in 1985.

Its security is based on a hardness assumption we already know well: DDH.

Note: PKE was first realized by Clifford Cocks at GCHQ in 1973. But this was only declassified in 1997.

ElGamal PKE

→ $Gen(1^n)$: Generate a tuple (G, q, g) where G is a cyclic group of order q and g is a generator for G.
Sample $x \leftarrow \mathbb{Z}_q$ and compute $h = g^x$.
Output
$$pk = (G, q, g, h)$$
$$sk = (G, q, g, x)$$

→ $Enc(pk, m)$: Sample $y \leftarrow \mathbb{Z}_q$ and output the ciphertext $(g^y, h^y \cdot m)$

→ $Dec(sk, c)$: Parse $c = (c_1, c_2)$.
output $m = c_2 \cdot (c_1^{-1})^x$.

First, let's establish the correctness of this scheme:

If $c = \text{Enc}(pk, m)$, then $c_1 = g^y$ & $c_2 = h^y \cdot m$.

Thus,

$$c_2 (c_1^{-1})^x = h^y \cdot m \cdot (g^{-y})^x$$

$$= g^{xy} \cdot m \cdot g^{-yx}$$

$$= g^{xy} \cdot m \cdot g^{-xy}$$

$$= m.$$

We now prove CPA-security under DDH.

Thm: The ElGamal PKE is CPA-secure assuming that DDH holds with respect to Gen.

Proof: Consider the idealized encryption procedure $\widetilde{\text{Enc}}(pk, m)$ that samples $y, z \leftarrow \mathbb{Z}_q$ & outputs the "ciphertext" $(g^y, g^z \cdot m)$.

Let $\widetilde{\text{Gam}}$ denote the CPA-security game above played with $\widetilde{\text{Enc}}$ in place of Enc.

Claim: For any adversary $\mathcal{A}$,

$$\Pr(\mathcal{A} \text{ wins } \widetilde{\text{Gen}}) = 1/2.$$

Proof: Note that $\widetilde{\text{Enc}}(pk, m)$ is independent of m.

Now suppose that there is a PPT adversary $\mathcal{A}$ and a polynomial $p(n)$ such that

$$\Pr(\mathcal{A}(1^n) \text{ wins real Gen}) \geq 1/2 + 1/p(n)$$

for infinitely many n's. Fix any such n.

We will build a PPT distinguisher $\mathcal{D}$ that breaks DDH wrt Gen.

$\mathcal{D}$ receives as input (g^x, g^y, g^z) where $x, y \leftarrow \mathbb{Z}_q$ and either $z \leftarrow \mathbb{Z}_q$ or $z = x \cdot y$.

We want that

$$\left| \Pr_{x,y \leftarrow \mathbb{Z}_q}(\mathcal{D}(G, q, g, g^x, g^y, g^{xy}) = 1) - \Pr_{x,y,z \leftarrow \mathbb{Z}_q}(\mathcal{D}(G, q, g, g^x, g^y, g^z) = 1) \right| \geq \text{non-negl}$$

D proceeds as follows on input (G, q, g, g^x, g^y, g^z):

→ Compute $pk = (G, q, g, g^x)$ and send to $A(1^n)$.

→ If $A(1^n, pk)$ outputs m_0, m_1, sample $b \leftarrow \{0,1\}$ and send $c = (g^y, g^z \cdot m_b)$ to A.

→ If $A(1^n, pk, c)$ outputs b', then output 1 iff $b' = b$.

Note that D outputs 1 exactly when A wins the game. If $z \leftarrow \mathbb{Z}_q$, then the game is $\widetilde{\text{Game}}$. Otherwise, it's the real CPA-security game.

Therefore

$$\left| \Pr_{x,y \leftarrow \mathbb{Z}_q}(D(G,q,g,g^x,g^y,g^{xy}) = 1) - \Pr_{x,y,z \leftarrow \mathbb{Z}_q}(D(G,q,g,g^x,g^y,g^z) = 1) \right|$$

$$= \left| \Pr(A \text{ wins real game}) - \Pr(A \text{ wins } \widetilde{\text{Game}}) \right|$$

$$\geq 1/2 + 1/p(n) - 1/2$$

$$= 1/p(n).$$

This contradicts the assumption that DDH is hard wrt Gen. ∎

A note on the practical use of PKE

PKE schemes are much slower than symmetric encryption schemes. In practice they are used to encrypt and transmit a short random string that is then used to enable fast private two-way communication using fast symmetric encryption schemes.

For more on this, see KL Section 11.3.