G22.3033-003 Introduction to Cryptography	October 18, 2001

Lecture 7

Lecturer: Yevgeniy Dodis — *Scribe: Nelly Fazio*

Last time we proved that if there exists a secure PKE $\mathcal{E}$ to encrypt one bit, then there also exists a secure PKE $\mathcal{E}'$ to encrypt as many bits as we want. In this lecture, we will go further in this direction, searching for the minimal assumption that implies the existence of a indistinguishable (IND) PKE.

Afterwards, we will temporarily leave the construction of our theoretical framework, and we will move to analyzing efficient and practical encryption schemes, based on less general assumptions. In particular, we will present the *ElGamal* PKE, whose security can be stated in term of two different (but related) assumptions: the *Computational Diffie-Hellman* (CDH) Assumption, and the *Decisional Diffie-Hellman* (DDH) Assumption.

Finally, we will criticize our definition of indistinguishable PKE, both from a syntactic and semantic point of view, and we will fix those problems, thus obtaining the notion of indistinguishable PKE against chosen-plaintext attack (CPA).

1 Minimal Assumption For Indistinguishable Encryption

Up to now, we have introduced a few mathematical entities in our theoretical framework, and a significant part of our work has been in realize their connections. For the sake of clarity, in fig. 1 we sketch the cryptographic primitives we have encountered so far, and how they are related to each other: an arrow from a primitive to another, means that the existence of the first implies the existence of the second.

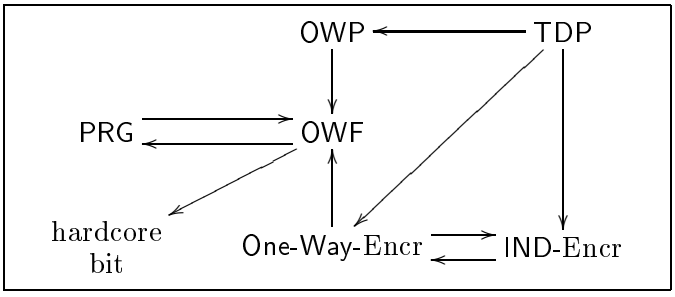

Figure 1: Relations between Cryptographic Primitives

It is worth noticing that the arrow from One-Way-Encryption to IND-Encryption does not mean that each PKE which is One-Way secure is also PKE secure (which is not the case!). Instead, it means that the existence of a One-Way secure PKE implies the existence of an IND PKE as proved in the following theorem.

Theorem 1
If there exists a One-Way *secure* PKE $\mathcal{E} = (G, E, D)$ *then there exists an* IND PKE $\mathcal{E}' = (G, E', D')$.

Proof: The proof is an application of the One-Time Pad Lemma. By the hypothesis, we have that $\mathcal{E}$ is a One-Way secure PKE, i.e. the encryption function E is a OWF. Using the Goldreich-Levin theorem, we know that a random parity of the input bits of x is a hardcore predicate of E:

$$(E(x), r, (x \cdot r \bmod 2)) \approx (E(x), r, b), \quad \text{where } x, r \leftarrow_r M_k, b \leftarrow_r \{0,1\} \qquad (\dagger)$$

Let us define a one-bit PKE $\mathcal{E}' = (G, E', D')$, where to encrypt one bit b' using the randomness (x, r), E' runs E as follows:

$$E'(b'; x, r) = (E(x), r, ((x \cdot r) \bmod 2) \oplus b')$$

The key-generation algorithm for $\mathcal{E}'$ is exactly the same as for $\mathcal{E}$, while the decryption algorithm D' is the straightforward inverse of E':

$$D'(y, r, c) = (((D(y) \cdot r) \bmod 2) \oplus c).$$

To prove that $\mathcal{E}'$ is an IND PKE, we apply the One-Time Pad Lemma, choosing the distributions $X = (E(x), r)$ and $Y = (x \cdot r) \bmod 2$. Therefore, the (†) can be rewritten as $(X, Y) \approx (X, b)$. Thus, by the One-Time Pad Lemma, it holds that $(X, Y \oplus 0) \approx (X, Y \oplus 1)$, that is:

$$E'(0; x, r) = (E(x), r, ((x \cdot r) \bmod 2) \oplus 0) \approx (E(x), r, ((x \cdot r) \bmod 2) \oplus 1) = E'(1; x, r).$$

□

2 Efficient Encryption

So far, we have dealt with general constructions for PKE, based on the weakest possible assumptions; anyway, those constructions, although polynomial, are still inefficient and of no practical use. Now, we study a specific, efficient example of PKE, the *ElGamal* cryptosystem, based on the hardness of the *Diffie-Hellman* Problem.

2.1 The Diffie-Hellman Problem

In their seminal paper on Public-Key Cryptography, W. Diffie and M. Hellman proposed the first *Key-Exchange Scheme*, i.e. a scheme to enable two parties, Alice and Bob, to exchange a jointly-selected shared key k to be used in a subsequent, secure communication (for example, as the shared key in a One-Time Pad cryptosystem). The scheme can be sketched as follows:

- Alice and Bob choose a prime number p, and a generator g of the cyclic group $\mathbb{Z}_p^*$;
- Alice chooses (and keeps secret) an exponent $a \in \mathbb{Z}_{p-1}$, then computes $\alpha = g^a \bmod p$, and finally sends α to Bob;

- Bob chooses (and keeps secret) an exponent $b \in \mathbb{Z}_{p-1}$, then computes $\beta = g^b \bmod p$, and finally sends β to Alice;
- as soon as Bob receives the message α from Alice, he computes $k_B = \alpha^b \bmod p$;
- as soon as Alice receives the message β from Bob, she computes $k_A = \beta^a \bmod p$.

Clearly, at the end of this protocol, Alice and Bob share the secret key $k = g^{ab} \bmod p$:

$$k_A \equiv \beta^a \equiv (g^b)^a \equiv (g^a)^b \equiv \alpha^b \equiv k_B \pmod p$$

Notice that, while Alice can easily compute k (since she knows a), and Bob can easily compute k (since he knows b), an eavesdropper Eve has to face the much more difficult problem of computing g^{ab} given knowledge $(p, g, g^a \bmod p, g^b \bmod p)$: we will refer to this problem as the *Diffie-Hellman Problem* (DHP).

INPUT: p, g generator of $\mathbb{Z}_p^*$, $(g^a \bmod p)$ and $(g^b \bmod p)$
OUTPUT: $g^{ab} \bmod p$

But how can Eve succeed in her task? Of course, if Eve were able to compute the discrete logarithms in $\mathbb{Z}_p^*$, then she could compute $g^{ab} \bmod p$, by first extracting a from $g^a \bmod p$, and then raising $g^b \bmod p$ to the ath power. It follows that Diffie-Hellman problem is at most as difficult as the more general Discrete Logarithm problem; anyway, this does not imply the equivalence of the two problems: it could certainly be the case that computing discrete logarithms is hard, while solving the Diffie-Hellman problem is possible.

Problems.
It is worth pointing out that the Diffie-Hellman Key-Exchange Scheme, as presented here, suffers from two problems:

- A first problem is that there is no way for Bob to know who is the sender of a message. This fact allows Eve to mount the so called *person-in-the-middle* attack:
  - Alice chooses (and keeps secret) an exponent $a \in \mathbb{Z}_{p-1}$, then computes $g^a \bmod p$, and sends it to Bob;
  - Eve intercepts $g^a \bmod p$ (thus preventing Bob from getting it), and sends $g^{a'} \bmod p$ to Bob, for a random a' of her choice;
  - Bob chooses (and keeps secret) an exponent $b \in \mathbb{Z}_{p-1}$, then computes $g^b \bmod p$, and sends it to Alice;
  - Eve intercepts $g^b \bmod p$ (thus preventing Alice from getting it), and sends $g^{b'} \bmod p$ to Alice, for a random b' of her choice;
  - when Bob receives the message $g^{a'} \bmod p$, he (erroneously) assumes that it comes from Alice, and thus he sets $k_B = (g^{a'})^b \bmod p = g^{a'b} \bmod p$;
  - when Alice receives the message $g^{b'} \bmod p$, she (erroneously) assumes that it comes from Bob, and thus she sets $k_A = (g^{b'})^a \bmod p = g^{ab'} \bmod p$;

- Eve can easily compute both $k_A = (g^a)^{b'} \bmod p = g^{ab'} \bmod p$ and $k_B = (g^b)^{a'} \bmod p = g^{a'b} \bmod p$.

As a consequence of the attack, at the end of this run of the protocol, Alice and Bob happily enjoy their "secure" connection, exchanging their love messages and saying nasty things about Eve, but ... $k_A \neq k_B$, and so actually they are talking trough Eve, who can not only learn everything, but also manipulate the communication.

To avoid this attack, a keep point is to add *Authentication* to the basic Key-Exchange Scheme, i.e. provide some mechanisms that enables the receipient of a message to be sure about the identity of the sender. This is typically achieved by means of *Digital Signature*, which will be discussed later.

- Another problem is that Eve can learn the quadratic character[1] of $g^{ab} \bmod p$. Recall that the Legendre's symbol of $g^a \bmod p$ (which is efficiently computable), reveals the LSB of a; similarly, the Legendre's symbol of $g^b \bmod p$ tell us the LSB of b. From these two pieces of information, Eve can recover the LSB of ab, and thus she can determine whether g^{ab} is in QR_p or not. In other words, the scheme leaks some information about the key.

  It is common practice to fix this problem via an *ad hoc* solution, that consists in imposing a specific structure on p. Let us assume that $p = 2q + 1$, with p and q both primes (such a prime p is called *strong prime*). Then consider the subgroup G of $\mathbb{Z}_p^*$ made up by all the quadratic residues modulo p: in other words $G \doteq QR_p$. It is known that the order of this subgroup is $\frac{p-1}{2} = q$, i.e. QR_p has prime order. Moreover, if h is a generator of $\mathbb{Z}_p^*$, then $g = h^2 \bmod p$ is a generator of G. Finally, $(G, \cdot)$ is isomorphic to $(\mathbb{Z}_q, +)$, since $g^a \cdot g^b \equiv g^{(a+b) \bmod q} \pmod p$. Indeed, by definition of *generator* and of *order* of a group, it holds that $g^q \equiv 1 \pmod p$. Now, writing $a+b = c \cdot q + (a+b) \bmod q$ for some c, we can make the following considerations:

$$g^a \cdot g^b \equiv g^{(a+b)} \equiv g^{c \cdot q} \cdot g^{(a+b) \bmod q} \equiv \left(g^q\right)^c \cdot g^{(a+b) \bmod q} \equiv 1^c \cdot g^{(a+b) \bmod q} \pmod p$$

  Finally:

$$g^a \cdot g^b \equiv g^{(a+b) \bmod q} \pmod p \ .$$

  We can now modify the Diffie-Hellman Key-Exchange protocol, so that the random values a and b chosen by Alice and Bob are drawn from $\mathbb{Z}_q$ (instead of $\mathbb{Z}_{p-1}$), and the whole computation is performed in G (instead of $\mathbb{Z}_p^*$).

  The reason why these changes are effective is that now it is always the case that $g^a \bmod p$, $g^b \bmod p$ and $g^{ab} \bmod p$ are quadratic residues, and so what Eve can learn trough the considerations above (namely, the quadratic character of $g^{ab} \bmod p$), is already publicly know.

  However, there is no guarantee that this is enough in order to prevent the leakage of *any* partial information about $g^{ab} \bmod p$, although it is believed to be true, and it constitutes the core of the Decisional Diffie-Hellman Assumption.

[1] The quadratic character $\chi(a)$ for an element $a \in \mathbb{Z}_p^*$ is defined as: $\chi(a) = \begin{cases} 0 & \text{if } a \in QR_p \\ 1 & \text{if } a \notin QR_p \end{cases}$

3 The Diffie-Hellman Assumption

The DHP presented in the previous section is a clear, well-stated mathematical problem; since the first time it was posed, there has been a great effort in attempting to solve this problem, but, as of today, the best known way to solve it is through a reduction to the Discrete Logarithm Problem. For this reason, it is believed to be intractable (when working in a group G of prime order); this is formalized by the following assumption:

Definition 2 (Computational Diffie-Hellman (CDH) Assumption)
$\forall$ PPT *algorithm* A

$$\Pr[A(g^a, g^b) = g^{ab} \mid p = 2q+1, |p| = k, p, q \textit{ primes}, a, b \leftarrow_r \mathbb{Z}_q, g \textit{ generator of } G] = \mathsf{negl}(k)$$

(where it is omitted that A also knows the public modulo p and the generator g.)

In other words, the CDH assumption states that "it is hard to *completely* compute $g^{ab} \bmod p$". But this is not enough: consider, for example, a scenario where Alice and Bob, after engaging in the Key-Exchange Protocol, need to use only a small part of the shared key. Based on the CDH assumption, it wouldn't be 'safe' to take just a portion of g^{ab}, since the possibility that Eve is able to learn that portion of the shared key is not ruled out by the CDH assumption.

Therefore, it is often convenient to consider the much stronger assumption below:

Definition 3 (Decisional Diffie-Hellman (DDH) Assumption)
$\forall$ PPT *algorithm* A

$$\Big| \Pr[A(g^a, g^b, g^{ab}) = 1 \mid p = 2q+1, |p| = k, p, q \textit{ primes}, a, b \leftarrow_r \mathbb{Z}_q, g \textit{ generator of } G] - \Pr[A(g^a, g^b, g^c) = 1 \mid p = 2q+1, |p| = k, p, q \textit{ primes}, a, b, c \leftarrow_r \mathbb{Z}_q, g \textit{ generator of } G] \Big| = \mathsf{negl}(k)$$

(where, again, it is omitted that A also knows the public modulo p and the generator g.)

The above definition is equivalent to the conjecture that "it is infeasible to distinguish between g^{ab} and g^c", i.e.

$$(g^a, g^b, g^{ab}) \approx (g^a, g^b, g^c), \quad \text{where } a, b, c \leftarrow_r \mathbb{Z}_q$$

llk Interestingly, the relation between the CDH and the DDH resembles the relation between the notion of One-Way Security and of Polynomial Indistinguishability Security: again, in one case it is hard to *completely* compute "something", and in the other case "something" is (polynomially) indistinguishable from random. We will see that this is more than a coincidence when treating the security of the ElGamal Cryptosystem.

4 The ElGamal Cryptosystem

Now, we describe the ElGamal Cryptosystem, which exploits the hardness of the DHP to achieve a reasonable level of security quite efficiently.

1. On input 1^k, the *key-generation algorithm* G chooses a strong prime $p = 2q + 1$ along with a generator h of $\mathbb{Z}_p^*$, and sets $g = h^2 \bmod p$. Afterwards, G takes a random element $x \in \mathbb{Z}_q$ and sets $y = g^x \bmod p$. Finally, G outputs (PK, SK, M_k), where $PK = (p, g, y)$, $SK = x$, and the message space is $M_k = G = QR_p$.

2. Given a message $m \in M_k$ and some randomness r, the *encryption algorithm* E_{PK} outputs the ciphertext $(s, t) = (g^r, y^r \cdot m)$, where $PK = (p, g, y)$.[2] Notice that, since $y = g^x$, the ciphertext is actually of the form $(g^r, g^{xr} \cdot m)$, although, of course, this form is "hidden', and Eve can only see (s, t).

3. In order to decrypt a ciphertext (s, t), given the private key $SK = x$, the decryption algorithm D first recovers the quantity $s^x = g^{rx} = y^r$, and then 'simplifies' it out from t, computing $t \cdot (s^x)^{-1} = t \cdot (y^r)^{-1} = y^r \cdot m \cdot (y^r)^{-1} = m$.

4. In attacking the cryptosystem, the adversary Eve knows the public key $PK = (p, g, y = g^x)$ and a ciphertext $(s, t) = (g^r, y^r \cdot m)$, corresponding to m: overall, Eve's knowledge can be represented as $(g^r, g^x, g^{xr} \cdot m)$.

Comments.

- Both the encryption and the decryption algorithms of this scheme are fairly efficient, since, in each invocation, at most two modular exponentiations are required.

- Each ciphertext produced by the encryption algorithm is long exactly twice as much as the plaintext. This space overhead is induced by the presence of some randomness in the encryption algorithm — which is unavoidable in any PKE that 'aims' to fulfil a notion of security that is stronger than One-Way security: as we noticed in previous lectures, a share of non-determinism is necessary to be able to send more than once the same message, in such a way that Eve cannot recognize that two different ciphertexts are related to each other.

- As it is defined, the ElGamal Scheme allows us to encrypt only messages that are elements of the group G. Usually, the messages we want to encrypt are not numbers, although they can easily be mapped onto numbers in some specific range: but how can we force such numbers to be quadratic residues? We need a technique to embed numbers in a certain range into G. First notice that p being a strong prime implies $p \equiv 3 \pmod 4$. Recall that $p = 2q+1$ where q is prime: thus, q is odd, i.e. $q = 2k+1$, and substituting we obtain $p = 4k + 3$, or alternatively $p \equiv 3 \pmod 4$. So, for any $m \in \mathbb{Z}_p^*$, only one among m and $-m$ is a quadratic residuos. Now, suppose that the range onto which messages are mapped is $[0, \ldots, q-1]$. Let us consider m and $-m$: given that $m < q = \frac{p-1}{2}$, it follows that $-m \geq \frac{p-1}{2} = q$, so that no message is mapped onto $-m$. Since one and only one among m and $-m$ is a quadratic residuos, we can choose that one, and use it for the encryption of the original message.

- One important feature presented by the ElGamal PKE is that it is "blindable", i.e. given an encryption $(s, t) = (g^r, y^r \cdot m)$ for an unknown message m, it is possible to

[2] Where omitted, the computation is assumed modulo p.

obtain an encryption for the message $m \cdot u$, for any $u \in G$. The way this is achieved is by picking a random $r' \in \mathbb{Z}_q$ and multiplying s by $g^{r'}$ and t by $y^{r'} \cdot u$. To see why it works, consider:

$$(s \cdot g^{r'}, t \cdot y^{r'} \cdot u) = (g^r \cdot g^{r'}, y^r \cdot m \cdot y^{r'} \cdot u) = (g^{r+r'}, y^{r+r'} \cdot (m \cdot u))$$

This property, by itself, its neither good nor bad: it depends upon the scenario at hand. Consider, for example, the case of a centralized authority C that is able to decrypt messages for others, but should not be allowed to learn the content of those messages. At first glance, this may seem hopeless, but the "blindable" property comes in help: instead of submitting the actual ciphertext, one can choose a random message u and send to C a "blinded" ciphertext for $m \cdot u$. When C decrypts, it cannot understand the content (since the randomness of u makes it random!), but the other party can recover m by simplifying out u.

On the other hand, in the setting of an auction, the "blindable" property would allow a participant to "cheat" by doubling the price offered by another participant, even when the auction bets are being encrypted with the ElGamal PKE.

- The way this scheme works is somehow similar to the One-Time Pad scheme: the message m is "padded" with the mask y^r, and this mask is "encoded" in the ciphertext in such a way that only the holder of the secret key x is able to recover it from the first part of the ciphertext. Actually, the similarity goes further, and in discussing the security of the ElGamal Scheme, we will see that it is possible to use a generalized version of the One-Time Pad Lemma.

5 Security of the ElGamal Cryptosystem

We now examine the security of the ElGamal Cryptosystem, proving the following theorems.

Theorem 4 *Under the* CDH *assumption, the ElGamal cryptosystem is a One-Way secure* PKE.

Proof: For the sake of contradiction, let us assume that the ElGamal cryptosystem is not a One-Way secure PKE:
$\exists$ PPT algorithm A such that:

$$\Pr[A(g^x, g^r, g^{xr} \cdot m) = m \mid p = 2q+1, |p| = k, p, q \text{ primes}, x \leftarrow_r \mathbb{Z}_q, g \text{ generator of } G] = \epsilon$$

for some non-negligible ϵ.

Using a reduction approach, we now construct an algorithm A' which, given black-box access to the algorithm A, can efficiently solve the DHP with non-negligible probability, contradicting the hypothesis. On input (g^x, g^r), for random x and r in $\mathbb{Z}_q$, A' chooses $c \leftarrow_r \mathbb{Z}_q$ and runs $A(g^x, g^r, g^c)$: with probability ϵ, A will decrypt the 'ciphertext' g^c and thus will output $\tilde{m} = g^c \cdot (g^{xr})^{-1}$. At this point, A' computes $g^c \cdot \tilde{m}^{-1}$ as its guess for g^{xr}. Therefore, with probability roughly ϵ, A' will successfully solve the Diffie-Hellman Problem. $\square$

Before moving to the analysis of security under the DDH assumption, it is useful to introduce the following straightforward generalization of the One-Time Pad Lemma:

Lemma 5 (Generalized One-Time Pad Lemma) *Let $(G, \cdot)$ be a group, and R denote the uniform distribution over G. For all the distributions X, Y (not necessarily independent) over G, if $(X, Y) \approx (X, R)$, then for all $m_0, m_1 \in G$, we have $(X, Y \cdot m_0) \approx (X, Y \cdot m_1)$.*

Proof: Just substitute $\cdot$ for $\oplus$ in the proof of the One-Time Pad Lemma (see Lecture 6). $\square$

Theorem 6 *Under the* DDH *assumption, the ElGamal cryptosystem is an* IND *secure* PKE.

Proof: We want to apply the Generalized One-Time Pad Lemma, choosing the distributions $X = (g^r, g^x)$ and $Y = g^{xr}$. From the DDH assumption, we have that $(g^r, g^x, g^{rx}) \approx (g^r, g^x, g^z)$, which can be restated as $(X, Y) \approx (X, R)$, where R is the uniform distribution over the group $G = QR_p$. Now, by the Generalized One-Time Pad Lemma, we can conclude that for all m_0, m_1 in G, it holds that $(X, Y \cdot m_0) \approx (X, Y \cdot m_1)$, i.e.:

$$(\forall m_0, m_1 \in G) \quad (g^r, g^x, g^{xr} \cdot m_0) \approx (g^r, g^x, g^{xr} \cdot m_1)$$

$\square$

Comments. Quite interestingly, based on two different assumptions, it is possible to show that the *same* construction fulfil two different notions of security. Moreover, the assumption made and the security obtained are well-coupled: assuming that it is difficult to completely solve the DHP leads to the difficulty of completely breaking the ElGamal PKE; assuming that it is infeasible to learn anything useful about the solution of the DHP leads to the difficulty of learning anything useful about messages encrypted with the ElGamal PKE.

6 Towards A Better Definitions Of Security

The notion of Indistinguishable Security we have used so far is not well stated: in fact, it is slightly incorrect, since it suffers from both a syntactic criticism and a semantic one.

Syntactic Criticism A first issue with this definition of security is that it requires that, for each pair of messages (m_0, m_1), their encryptions are indistinguishable from each other: $\boxed{(\forall m_0, m_1 \in M_k)}$, $\forall$ PPT algorithm A

$$\left| \Pr[A(c, PK) = b \mid \boxed{(SK, PK, M_k) \leftarrow_r G(1^k)}, b \leftarrow_r \{0,1\}, c \leftarrow_r E_{PK}(m_b)] - \frac{1}{2} \right| \leq \mathsf{negl}(k)$$

In other words, we are quantifying on the message space M_k, even before the random choice of (SK, PK, M_k)! In the cases in which the message space M_k is fixed, or just varies as a function of the security parameter k (e.g. when $M_k = \{0,1\}$ or $M_k = \{0,1\}^{p(k)}$), this can be fixed simply "moving M_k out" of the output of the key-generation algorithm $G(1^k)$. But in other interesting cases, like the RSA and the ElGamal PKE, the message space M_k

is indeed somehow related to the public key being used, so that it doesn't yet make sense to choose a pair of messages before knowing the actual public key PK being selected.

This is clearly just a syntactic problem, so that it doesn't affect too much the overall correctness of what we have proved until now.

Semantic Criticism A semantic problem with the notion of indistinguishable security is that it just states that any PPT adversary must have a negligible advantage in distinguishing the encryptions of any pair of messages (m_0, m_1), when the public key is *randomly* selected. Therefore, it does not cast away the chance that, given knowledge of the public key, an adversary may be able to come up with a pair of messages such that it can indeed distinguish the associated ciphertexts.

To overcome this problem, we could be tempted to modify the definition in such a way that first we fix the public key PK and the private key SK, and then we quantify overall the possible pair of messages. But in this case we are asking too much, since the existence of "bad" pairs of messages is unavoidable. To see why, consider an adversary that, in distinguishing between the encryptions of (m_0, m_1), always assume that the second message is the private key SK, and try to decrypt the encryption it has being given accordingly. Clearly, the advantage of this adversary is negligible whenever the second message is not the secret key; however, in the very specific, pathological case in which the pair is of the form (m_0, SK), its advantage will be 1.

After all, what we really wants from our definition of security is that, even if the adversary already knows PK, it should be infeasible for it to find two messages for which it has non negligible advantage.

7 Indistinguishable Security Against Chosen Plaintext Attack

Both the criticisms stated above stem from the the fact that the messages m_0, m_1 are quantified "outside" the probability which constitutes the advantage of the adversary:

- from a syntactic point of view, this is bad since the message space M_k is chosen "inside" the probability;
- from a semantic point of view, this is bad since we want to allow the adversary to choose the pair of messages after seeing the PK, and thus the probability should be taken over all the pairs of messages that it can efficiently find.

We can fix both this problems with the following definition of:

Definition 7 (Indistinguishable Security Against Chosen Plaintext Attack)
A PKE $\mathcal{E} = (G, E, D)$ *is* IND *(against a chosen plaintext attack* CPA*) if,*
$\forall$ PPT *algorithm* A

$$\left| \Pr\left[b = \tilde{b} \;\middle|\; \begin{array}{c} (PK, SK, M_k) \leftarrow_r G(1^k), (m_0, m_1, \alpha) \leftarrow A(PK, M_k, \text{'find'}) \\ b \leftarrow_r \{0,1\}, c \leftarrow E_{PK}(m_b), \tilde{b} \leftarrow A(c, \alpha, \text{'guess'}) \end{array} \right] - \frac{1}{2} \right| = \mathsf{negl}(k)$$

Let's take a closer look at what is going on! This notion of security can be viewed as the following game between us and the adversary:

1. we run the key-generation algorithm, obtaining (PK, SK, M_k);
2. we give the public key PK and the message space M_k to the adversary and ask it to 'find' a pair of messages in M_k for which it believes it can distinguish encryptions under the key PK;
3. the adversary outputs a pair of messages (m_0, m_1) of its choice, along with its final "state", i.e. some kind of summary of the considerations and computations that led it to choose this specific pair of messages;
4. we choose which one of two message we want encrypt, and give it the corresponding ciphertext c;
5. we ask the adversary to tell which message we encrypted, allowing it to "remember" the reasons for which it chosen the pair (m_0, m_1).

We win the game (i.e. the considered PKE is IND secure against CPA) if the adversary's advantage (defined as usual as its probability of success) is negligible.

Let us now show that this new notion of security is strictly more general than the PK-only security.

Theorem 8 (CPA $\Rightarrow$ PK-only)
If a PKE $\mathcal{E} = (G, E, D)$ *is* CPA *secure, then* $\mathcal{E}$ *is also* PK-*only secure.*

Proof: Let us assume that $\mathcal{E}$ in not PK-only secure, i.e.
$\exists m_0, m_1$, $\exists$ a PPT algorithm A such that

$$\Pr(A(c, PK) = 1 \mid (SK, PK) \leftarrow_r G(1^k), c \leftarrow_r E_{PK}(c)) = \epsilon$$

where ϵ is not-negligible.

Now, we can construct an adversary A' that, using A as a black-box module, breaks the CPA security of $\mathcal{E}$. Recall that, in the CPA notion of security, the adversary acts in two rounds:

find Regardless of the public key PK at hand, A' always chooses the pair of message (m_0, m_1) of the hypothesis, and records in α the public key PK, m_0 and m_1:

$$A'(PK, M_k, \text{'find'}) \rightarrow (m_0, m_1, \alpha)$$

guess When challenged to determine which message was encrypted, A' runs $A(c, PK)$:

$$A'(c, \alpha, \text{'guess'}) \rightarrow A(c, PK)$$

Clearly, this algorithm A' has the same (non-negligible) advantage as A, contradicting the hypothesis that $\mathcal{E}$ was CPA secure. $\square$

Theorem 9 (PK-only $\not\Rightarrow$ CPA)
There exists a PKE $\mathcal{E} = (G, E, D)$ *which is* PK-*only secure, but is not* CPA *secure.*

Proof: To prove the theorem it suffices to come up with a counterexample, namely a specific PKE $\mathcal{E}'$ which is PK-only secure but not CPA secure. To this purpose, consider an arbitrary PKE $\mathcal{E} = (G, E, D)$ which is PK-only secure and modify it as follow:

Key-Generation Algorithm

$$\begin{aligned} &\textbf{set:} && (PK, SK) = G(1^k), r \text{ random} \\ &\textbf{output:} && (PK', SK') = ((PK, r), SK) \end{aligned}$$

Encryption

$$E'(m) = \begin{cases} (0, E(PK')), & \text{if } m = PK' \\ (1, E(m)), & \text{otherwise} \end{cases}$$

Decryption

$$\begin{cases} D'(0, y) = PK' \\ D'(1, y) = D(y) \end{cases}$$

Clearly, $\mathcal{E}'$ is *not* CPA secure, since once the adversary knows the public key PK', it can ask to be challenged on the pair of messages (PK', m_1), and it will always succeed in distinguishing an encryption of PK' (which starts with 0) from an encryption of any other message (which starts with 1).

Nevertheless, $\mathcal{E}'$ is still a PK-only secure PKE, since, in this setting, it is no longer possible to efficiently find a pair of messages (m_0, m_1) whose encryptions are easy-to-distinguish. Indeed, due to the random r present in the public key PK', the probability that the adversary could guess the public key PK' that will be used, is always negligible (no matter how short the public key PK for the PKE $\mathcal{E}$ could be). Besides, it will be difficult for the adversary to find a generic pair (m_0, m_1) which makes its task easy, since such a pair could be also used to tell apart encryptions for the PKE $\mathcal{E}$, contradicting the hypothesis that $\mathcal{E}$ is PK-only secure. □

Remark 10 *Despite the slight differences, and the non-equivalence proved in the above theorem, the two notions of security formalized are quite close. In particular, all the results stated so far (the generalization of a one-bit secure* PKE *to a secure* PKE *for any number of bits, the Blum-Goldwasser system, the analysis of security of the ElGamal cryptosystem) hold as well when we consider the indistinguishable security against* CPA.

This is because the separation between CPA *and* PK-*only security is somewhat artificial, and not of real concern, but anyway, once you know, why not to use the right one?*