6.875J/18.425J Cryptography and Cryptanalysis	September 26, 2001
Lecture 6	
Instructor: Anna Lysyanskaya	*Scribed by: Emina Torlak*

1 Recap

1.1 The Goldwasser-Micali Cryptosystem

1.1.1 For one-bit messages

Key Generation G Alice chooses two random k-bit primes, p and q. Let $n = pq$. Alice also chooses a random $x \in QNR(n)$. I.e., x is a random non-square with Jacobi symbol 1. Alice's public key PK is the tuple (n, x), while her secret key SK is n's factorization (p, q).

Encryption E To encrypt a 0, Bob sends Alice a random square modulo n. To encrypt a 1, Bob sends Alice a random non-square with Jacobi symbol 1. More precisely, $c \leftarrow E(n, x, b)$ is computed as follows: $(r \leftarrow \mathbb{Z}_n^*; c \leftarrow r^2 x^b \bmod n)$.

Decryption D To decrypt, Alice determines whether her received value c is a quadratic residue modulo n. More precisely, Alice tests whether $\left(\frac{c}{p}\right) = \left(\frac{c}{q}\right) = 1$. In Lecture 3, we saw an efficient algorithm for doing this.

1.1.2 For l-bit messages

Key Generation G : same as in the binary message case: the public key consists of an RSA modulus n and a quadratic residue a. The secret key is the factorization of n.

Encryption E : Break up the message into bits, denoted $m = b_1 \circ b_2 \circ \ldots \circ b_l$. Encrypt m bit-by-bit: $c = r_1^2 a^{b_1}, \ldots, r_l^2 a^{b_l}$.

Decryption D : same as in the binary message case, for each bit.

1.2 Definition of GM-security for l-bit messages

Definition 1 *A cryptosystem* (G, E, D) *for message space* $\{0,1\}^l$ *is* GM-secure *if*

$$\begin{aligned}
&\forall PPTF\{Eve_k\}\forall m_0, m_1 \in \{0,1\}^l \exists f(k) \in negl \exists K \text{ such that for } k > K : \\
&\quad \Pr[\ (PK, SK) \leftarrow G(1^k); \\
&\qquad b \leftarrow \{0,1\}; \\
&\qquad c \leftarrow E(PK, m_b); \\
&\qquad b' \leftarrow Eve(PK, c) \quad : \quad b = b'] \leq 1/2 + f(k)
\end{aligned}$$

2 GM-Security for bits implies GM-security for strings

Theorem 2 *Suppose a cryptosystem (G', E', D') is GM-secure for one-bit messages. Then the following one-to-many transformation produces a cryptosystem (G, E, D) that is secure for l-bit messages:*

Key Generation G *: Same as G'.*

Encryption E *: Use E' for each bit of m, individually.*

Decryption D *: Using D', decrypt each individual bit.*

Proof Suppose, for contradiction, that (G, E, D) is not secure. We wish to show that this implies that (G', E', D') is not secure either.

What does it mean for (G, E, D) not to be GM-secure? It means that we can find an adversary $\{A_k\}$ which, for some polynomial $p(k)$ and infinitely many k, can distinguish between messages m_0 and m_1 with probability non-negligibly better than random guessing. More precisely, if (G, E, D) is not GM-secure, then

$$\Pr[(PK, SK) \leftarrow G(1^k); b \leftarrow \{0,1\};$$
$$c \leftarrow E(PK, m_{k,b}); b' \leftarrow A_k(PK, c, m_0, m_1) \quad : \quad b = b'] \geq 1/2 + 1/p(k)$$

Now, let us define the $l + 1$ *hybrid* messages as follows: for $0 \leq i \leq l$, m^i is obtained by taking the first i bits of m_1 and concatenating them with the last $l - i$ bits of m_0. Note that $m^0 \equiv m_0$ and $m^l \equiv m_1$. For example,

$m_0 \equiv$	m^0	0	0	1	0	1	1	0	0
	m^1	0	0	1	0	1	1	0	0
	m^2	0	1	1	0	1	1	0	0
	m^3	0	1	1	0	1	1	0	0
	m^4	0	1	1	0	1	1	0	0
	m^5	0	1	1	0	1	1	0	0
	m^6	0	1	1	0	1	0	0	0
	m^7	0	1	1	0	1	0	1	0
$m_1 \equiv$	m^8	0	1	1	0	1	0	1	1

Claim 3 *If there exists an adversary $\{A_k\}$ which can distinguish between m_0 and m_1 with probability polynomially better than 1/2, then for some $0 \leq i < l$,*

$$\Pr[(PK, SK) \leftarrow G(1^k); b \leftarrow \{0,1\};$$
$$c \leftarrow E(PK, m^{i+b}); b' \leftarrow A_k(PK, c, m_0, m_1) \quad : \quad b = b'] \geq 1/2 + 1/(l * p(k))$$

Moreover, given A_k, m_0 and m_1, one can efficiently determine i.

Proof of 3: First, let us show that i exists. Let P_i denote the following probability:

$$\begin{aligned}\Pr[(PK, SK) \leftarrow G(1^k); b \leftarrow \{0,1\};\\ c \leftarrow E(PK, m^{i+b}); b' \leftarrow A_k(PK, c, m_0, m_1) \quad : \quad b' = 0]\end{aligned}$$

We will prove the existence of i using the following chain of implications: $P_0 - P_l \geq 2/p(k) \Rightarrow \exists i$ such that $P_i - P_{i+1} \geq 2/(l * p(k)) \Rightarrow$ The first part of Claim 3 is true.

1. $P_0 - P_l \geq 2/p(k)$

$$\begin{aligned} 1/2 + 1/p(k) \leq \Pr[(PK, SK) \leftarrow G(1^k); b \leftarrow \{0,1\};\\ c \leftarrow E(PK, m_b); b' \leftarrow A_k(PK, c, m_0, m_1) : b = b'] =\\ 1/2 * \Pr[(PK, SK) \leftarrow G(1^k); c \leftarrow E(PK, m_0);\\ b' \leftarrow A_k(PK, c, m_0, m_1) : b' = 0] +\\ 1/2 * \Pr[(PK, SK) \leftarrow G(1^k); c \leftarrow E(PK, m_1);\\ b' \leftarrow A_k(PK, c, m_0, m_1) : b' = 1] =\\ 1/2 * P_0 + 1/2 * (1 - P_l) = & \quad 1/2 + (P_0 - P_l)/2 \end{aligned}$$

   So we get $1/2 + 1/p(k) \leq 1/2 + (P_0 - P_l)/2 \Rightarrow P_0 - P_l \geq 2/p(k)$.

2. $P_0 - P_l \geq 2/p(k) \Rightarrow \exists i$ such that $P_i - P_{i+1} \geq 2/(l * p(k))$

   Let us show the contrapositive. If, for all i, $P_i - P_{i+1} < 2/(l * p(k))$, then

$$P_0 - P_l = \sum_{i=0}^{l-1} P_i - P_{i+1} < \sum_{i=0}^{l-1} 2/(l * p(k)) = 2l/(l * p(k)) = 2/p(k)$$

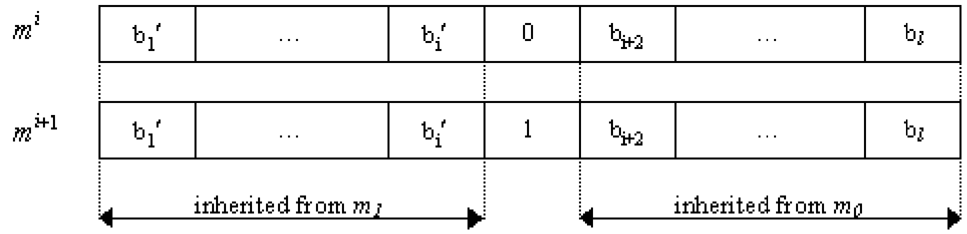
3. $\exists i$ such that $P_i - P_{i+1} \geq 2/(l * p(k)) \Rightarrow$ for that i, $Pr[(PK, SK) \leftarrow G(1^k);$ $b \leftarrow \{0,1\}; c \leftarrow E(PK, m^{i+b}); b' \leftarrow A_k(PK, c, m_0, m_1) : b = b'] \geq 1/2 + 1/(l * p(k))$

$$\begin{aligned} \Pr[(PK, SK) \leftarrow G(1^k); b \leftarrow \{0,1\}; c \leftarrow E(PK, m^{i+b});\\ b' \leftarrow A_k(PK, c, m_0, m_1) : b = b'] =\\ 1/2 * \Pr[(PK, SK) \leftarrow G(1^k); c \leftarrow E(PK, m^i);\\ b' \leftarrow A_k(PK, c, m_0, m_1) : b' = 0] +\\ 1/2 * \Pr[(PK, SK) \leftarrow G(1^k); c \leftarrow E(PK, m^{i+1});\\ b' \leftarrow A_k(PK, c, m_0, m_1) : b' = 1] =\\ 1/2 * P_i + 1/2 * (1 - P_{i+1}) = 1/2 + (P_i - P_{i+1})/2 \geq & \quad 1/2 + 1/(l * p(k)) \end{aligned}$$

Now, we must convince ourselves of the second part of the claim, i.e., the fact that i can be discovered by a polynomial-time algorithm. Here's the algorithm I suggest: run A_k on various hybrids, and estimate P_i for each hybrid. If you run sufficiently many experiments, you can estimate it to within any polynomial fraction, with high probability. Once you've estimated P_i's, just check whether any of them differ. The correctness and running time of this algorithm can be proved using probability theory. ■

Now that we have proved the claim, let's go back to proving the theorem.

First let us make the following observations: (1) the messages m_0 and m_1 must differ in their $(i+1)$st bit, because otherwise m_k^i and m_k^{i+1} would be the same message and so no adversary would be able to tell their encryptions apart; (2) without loss of generality, we can assume that the $(i+1)$st bit of m_0 is 0, but the $(i+1)$st bit of m_1 is 1: they can be re-labelled if this is not the case.



Our reduction R receives as input a public key PK' from the cryptosystem (G', E', D') and a challenge ciphertext c'. Our goal is to determine whether c is an encryption of 0 or an encryption of 1.

Now, we create the challenge ciphertext for our adversary A_k, as follows: the first i blocks of c will be encryptions of the first i bits of m_1. The $i+1$st block will be the input ciphertext c'. The last $l-i-1$ blocks will be the encryptions of the last $l-i-1$ bits of m_0. Note that, in case c' is the encryption of 0, c will be the encryption of m_k^i; otherwise c will be the encryption of m_k^{i+1}. Our reduction will simply output whatever A_k outputs. Let us estimate the success probability of our reduction R_k:

$$\begin{aligned}
\Pr[(PK', SK') \leftarrow G'; b \leftarrow \{0,1\}; c' \leftarrow E'(PK', b); b' \leftarrow R_k(PK', c') : b = b'] &= \\
\Pr[(PK', SK') \leftarrow G'; b \leftarrow \{0,1\}; c' \leftarrow E'(PK', b); & \\
c \leftarrow (E'(PK', m_{k,0}^1), \ldots, E'(PK', m_{k,0}^i), c', E'(PK', m_{k,1}^{i+2}), \ldots, E'(PK', m_{k,1}^l)); & \\
b' \leftarrow A_k(PK', c) : b = b'] &= \\
\Pr[(PK, SK) \leftarrow G; b \leftarrow \{0,1\}; c \leftarrow E(m_k^{i+b}); b' \leftarrow A_k(PK, c) : b = b'] &\geq 1/2 + 1/(l * p(k))
\end{aligned}$$

So, we have proved that GM-security for bits implies GM-security for strings by showing that (G, E, D) is not l-GM-secure $\Rightarrow$ (G', E', D') is not GM-secure. ■

The proof technique we used is called *the hybrid argument.* Its idea is to look at the distributions "in between" the two distributions that we are interested in, and show that if our input distributions are distinguishable, then some "neighbor" distributions are distinguishable too.