CMSC 858K — Advanced Topics in Cryptography	February 24, 2004

Lecture 9

Lecturer: Jonathan Katz

Scribe(s): Julie Staub, Avi Dalal, Abheek Anand, Gelareh Taban

1 Introduction

In previous lectures, we constructed public-key encryption schemes which were provably secure against non-adaptive chosen-ciphertext (CCA-1) attacks, and also adaptive chosen-ciphertext (CCA-2) attacks. However, both constructions used generic non-interactive zero-knowledge proof systems which — although poly-time — are not very efficient (as we will see later in the course). Therefore, the constructions are not very practical.

In 1998, Cramer and Shoup proposed an encryption scheme [1] which was provably secure against adaptive chosen-ciphertext attacks and was also practical. The proof of security relies on the hardness of the Decisional Diffie-Hellman (DDH) problem in some underlying group.

In this lecture, we will first review the Decisional Diffie-Hellman assumption and the El-Gamal cryptosystem. Then we will modify the El-Gamal encryption scheme to construct a scheme secure against non-adaptive chosen-ciphertext attacks. This will be a step toward the full Cramer-Shoup cryptosystem, which we will cover in later lectures.

2 Background

The Cramer-Shoup cryptosystem relies on the DDH assumption in some finite group. In Lecture 4, we defined the Discrete Logarithm (DL) problem and DDH problem; we review them here. Let $\mathcal{G}$ be a finite cyclic group of prime order q, and let $g \in \mathcal{G}$ be a generator. Given $h \in \mathcal{G}$, the discrete logarithm problem requires us to compute $x \in \mathbb{Z}_q$ such that $g^x = h$. We denote this (unique) x by $\log_g h$. In particular groups $\mathcal{G}$ and for q large, it is assumed *hard* to compute x (this was formalized in Lecture 4).

A stronger assumption is the Decisional Diffie-Hellman (DDH) assumption. Here, given $\mathcal{G}$, a generator g of $\mathcal{G}$, and three elements $a, b, c \in \mathcal{G}$, we are asked (informally) to decide whether there exist x, y such that $a = g^x, b = g^y$ and $c = g^{xy}$. More formally, the DDH assumption states that the following two distributions are computationally indistinguishable:

- $\{\mathcal{G}, g, g^x, g^y, g^{xy}\}$
- $\{\mathcal{G}, g, g^x, g^y, g^z\}$

where g is a generator of $\mathcal{G}$ and x, y, z are chosen at random from $\mathbb{Z}_q$. (Again, see Lecture 4 for more formal definitions.)

Clearly, the DDH assumption implies the DL assumption. In fact, it appears to be considerably stronger. In particular, there are groups where DDH is false, but DL is still believed to hold. For example[1], let $\mathcal{G} = \mathbb{Z}_p^*$ for a prime p. On the one hand, the DL problem is believed to be hard in this group. Yet given g^a, g^b (for generator g) one can easily deduce the Legendre symbol of g^{ab} (which we denote by $\mathcal{L}(g^{ab})$). This observation gives an immediate method for distinguishing $\{\mathcal{G}, g, g^x, g^y, g^{xy}\}$ and $\{\mathcal{G}, g, g^x, g^y, g^z\}$ with non-negligible probability; namely, guess "DDH tuple" iff $\mathcal{L}(g^z) = \mathcal{L}(g^{xy})$.

A group in which DDH is assumed to hold is the following: Let $p = 2q+1$ where p, q are both prime. Let $\mathcal{G}$ be the subgroup of quadratic residues in $\mathbb{Z}_p^*$. Then $\mathcal{G}$ is a cyclic group of prime order q in which the DDH assumption is believed to hold.

The El-Gamal cryptosystem. In Lecture 4, we introduced the El-Gamal encryption scheme and proved that it was semantically secure under the DDH assumption; it may be useful to review that proof before continuing. We recall the scheme now (here, g is a generator of a group $\mathcal{G}$):

$$\begin{array}{ll} \mathsf{KeyGen}(1^k): & x \leftarrow \mathbb{Z}_q \\ & y = g^x \\ & PK = \langle g, y \rangle \\ & SK = \langle x \rangle \end{array}$$

$$\begin{array}{ll} \mathcal{E}_{PK}(m): & r \leftarrow \{0,1\}^k \\ & \text{output } \langle g^r, y^r \cdot m \rangle \end{array}$$

$$\begin{array}{ll} \mathcal{D}_{SK}(u,v): & \text{output } \frac{v}{u^x} \end{array}$$

Correctness: Assuming an honest execution of the protocol, we have

$$\frac{v}{u^x} = \frac{y^r \cdot m}{(g^r)^x} = \frac{(g^x)^r \cdot m}{(g^r)^x} = m.$$

3 Modifying El-Gamal

To build up to the Cramer-Shoup scheme, we first modify the El-Gamal encryption scheme and prove that the modified scheme is semantically secure under the DDH assumption. Although we achieve the same result, we introduce the modified scheme because the proof technique used to prove the modified scheme secure is different than that used to prove security of the El Gamal scheme in Lecture 4. The same sort of techniques will later be used to analyze the Cramer-Shoup scheme.

Consider the following scheme, where g_1, g_2 are two randomly-chosen generators in $\mathcal{G}$:

$$\begin{array}{ll} \mathsf{KeyGen}(1^k): & x, y \leftarrow \mathbb{Z}_q \\ & h = g_1^x g_2^y \\ & PK = \langle g_1, g_2, h \rangle \\ & SK = \langle x, y \rangle \end{array}$$

[1]Note that in this example, the order of $\mathcal{G}$ is not prime. However, all groups we use in our constructions will have prime order.

$\mathcal{E}_{PK}(m)$: $\quad r \leftarrow \mathbb{Z}_q$
output $\langle g_1^r,\, g_2^r,\, h^r \cdot m \rangle$

$\mathcal{D}_{SK}(u, v, e)$: $\quad$ output $\frac{e}{u^x v^y}$

Correctness: Assuming an honest execution of the protocol, we have

$$\frac{e}{u^x v^y} = \frac{h^r m}{(g_1^r)^x (g_2^r)^y} = \frac{h^r m}{(g_1^x g_2^y)^r} = m.$$

Theorem 1 *The above encryption scheme is semantically secure, assuming the DDH assumption in $\mathcal{G}$.*

Proof We prove security of this scheme by a reduction to the DDH problem. Suppose a PPT algorithm A can break the semantic security of the modified scheme. We then construct a PPT adversary $\hat{A}$ that can break the DDH problem by distinguishing a DDH tuple from a random tuple. Thus by contradiction, the security of the new scheme is proved.

The input to algorithm $\hat{A}$ is (g_1, g_2, g_3, g_4), which is either a DDH tuple or a random tuple. The algorithm $\hat{A}$ runs the following experiment.

$\underline{\hat{A}(g_1, g_2, g_3, g_4)}$
$x, y \leftarrow \mathbb{Z}_q$
$h = g_1^x g_2^y$
$PK = \langle g_1, g_2, h \rangle$
$(m_0, m_1) \leftarrow A(PK)$
$b \leftarrow \{0, 1\}$
$C = \langle g_3,\, g_4,\, g_3^x g_4^y \cdot m_b \rangle$
$b' \leftarrow A(PK, C)$
if $b = b'$, then guess "DDH tuple"
else guess "random tuple"

Claim 2 *If adversary $\hat{A}$ gets a DDH tuple, then A's view of the game is the same as in an execution of the real encryption scheme.*

Assume $\hat{A}$ gets a DH tuple. Then there exist α, $r \in \mathbb{Z}_q$, such that:

$$\langle g_1, g_2 = g_1^\alpha, g_3 = g_1^r, g_4 = g_1^{\alpha r} = g_2^r \rangle.$$

Therefore, the constructed public key and ciphertext have the following forms:

$$PK = \langle g_1,\, g_2,\, h = g_1^x g_2^y \rangle \text{ and } C = \langle g_1^r,\, g_2^r,\, (g_1^r)^x\, (g_2^r)^y \cdot m_b \rangle = \langle g_1^r,\, g_2^r,\, h^r \cdot m_b \rangle\,.$$

Thus, the distribution of the public key and the ciphertext correspond exactly to A's view in the real world. (It should be noted that this occurs even though $\hat{A}$ does not know nor use the value of r.) $\square$

Note that the claim implies:

$$\Pr[\hat{A} \text{ outputs "DDH tuple"} \mid \text{DDH tuple}] = \Pr[b' = b \mid A \text{ attacks real encryption scheme}].$$

Claim 3 *If adversary $\hat{A}$ gets a random tuple, then (with all but negligible probability) even an all-powerful A has no information about the bit b chosen by $\hat{A}$. In other words, b is information-theoretically hidden from A with all but negligible probability.*

An immediate corollary is that the probability that A correctly guesses b must be negligibly close to $1/2$ in this case (note that this holds even if A is all powerful). We continue with the proof of the claim.

Assume $\hat{A}$ gets a random tuple. Then there exist α, r, β chosen uniformly from $\mathbb{Z}_q$ such that (g_1, g_2, g_3, g_4) have the following form:

$$\langle g_1, g_2 = g_1^{\alpha}, g_3 = g_1^{r}, g_4 = g_1^{\beta} \rangle.$$

Note that with all but negligible probability, $\beta \neq \alpha r \pmod q$ and $\alpha \neq 0$. This is because, for example, $\beta = \alpha r$ with probability $1/q$, and q is exponentially large. From now on, we simply assume that these hold. Re-writing, this means that there exist $r, r' \in \mathbb{Z}_q$ with $r \neq r'$ such that $g_3 = g_1^r$ and $g_4 = g_2^{r'}$. We now look at A's information about x and y. Given the public key $PK = \langle g_1, g_2, h \rangle$, note that there are exactly q possible pairs (x, y) that could have been chosen by A. This is because h satisfies $h = g_1^x g_2^y$, and hence x and y satisfy

$$\log_{g_1} h = x + (\log_{g_1} g_2) \cdot y = x + \alpha y\,. \tag{1}$$

Now, for every $x \in \mathbb{Z}_q$ there is a unique $y \in \mathbb{Z}_q$ satisfying the above equation (and similarly for y). (We use the fact here that $\alpha \neq 0$.) In particular, then, there are exactly q solutions to the above equation and furthermore each of these possibilities are equally likely from the point of view of A.

Now, consider the term $g_3^x g_4^y$. We will be interested in the probability that $g_3^x g_4^y = \mu$, where μ is an arbitrary group element. In order for this to occur, we must have $\log_{g_1} \mu = \log_{g_1}(g_3^x g_4^y)$; i.e.:

$$\begin{aligned} &\log_{g_1} \mu \\ &\quad= \quad x \cdot \log_{g_1} g_3 + y \cdot \log_{g_1} g_4 \\ &\quad= \quad r \cdot x + r'\alpha \cdot y\,. \end{aligned} \tag{2}$$

Let $z_1 = \log_{g_1} h$ and $z_2 = \log_{g_1} \mu$. Then Equations (1) and (2) form a system of linear equations in x and y (over $\mathbb{Z}_q$) given by $\mathbf{B}\vec{x} = \vec{z}$, where

$$\mathbf{B} = \begin{pmatrix} 1 & \alpha \\ r & r'\alpha \end{pmatrix}, \quad \vec{x} = [x \;\; y]^T, \quad \vec{z} = [z_1 \;\; z_2]^T.$$

Assuming $r' \neq r$ and $\alpha \neq 0$ (see above), the matrix B above has rank 2 and therefore the above system of equations always has a (unique) solution in x, y. But since μ was an arbitrary group element, this means that *each possible value μ is possible* and moreover, *each value of μ is equally likely.* In other words, what we are saying is the following: given g_1, g_2, g_3, g_4, and $h = g_1^x g_2^y$ for x and y chosen uniformly at random from $\mathbb{Z}_q$ (and assuming $\log_{g_1} g_3 \neq \log_{g_2} g_4$), *even an all-powerful algorithm* cannot predict the value of $g_3^x g_4^y$ with probability better than $1/q$. (again, this is because all values of $g_3^x g_4^y$ are equally likely).

Since $g_3^x g_4^y$ is distributed uniformly in the group (from the point of view of A), it essentially acts like a "one-time pad" and thus A has no information (in an information-theoretic sense) about which message was encrypted, and hence no information about the value of b. This implies the claim. $\square$

The above claim implies:

$$\Pr[\hat{A} \text{ outputs "DDH tuple"} \mid \text{random tuple}] = 1/2 \pm negl(k).$$

Thus, the advantage of $\hat{A}$ is negligibly close to:

$$\left|\Pr[b = b' \mid A \text{ attacks real scheme}] - 1/2\right|.$$

Since we know that the advantage of $\hat{A}$ must be negligible, this implies that the probability that A correctly guess the value of b must be negligibly close to 1/2. But this exactly means that the encryption scheme is semantically secure, as desired. ∎

4 The Cramer-Shoup-"Lite" Cryptosystem

We next define the Cramer-Shoup "lite" encryption scheme. This is a step toward the full Cramer-Shoup scheme, but is only secure against *non-adaptive* chosen-ciphertext attacks. The scheme is defined as follows (g_1, g_2 are randomly-chosen generators of group $\mathcal{G}$):

$\mathsf{KeyGen}(1^k)$:	$x, y, a, b \leftarrow \mathbb{Z}_q$ $h = g_1^x g_2^y$ $c = g_1^a g_2^b$ $PK = \langle g_1, g_2, h, c \rangle$ $SK = \langle x, y, a, b \rangle$
$\mathcal{E}_{PK}(m)$:	$r \leftarrow \mathbb{Z}_q$ output $\langle g_1^r,\, g_2^r,\, h^r \cdot m,\, c^r \rangle$
$\mathcal{D}_{SK}(u, v, e, w)$:	// Verify w has the correct form if $(w = u^a v^b)$, then output $\frac{e}{u^x v^y}$ else output $\perp$

Correctness: If the ciphertext is computed honestly, the validity check succeeds since

$$w = c^r = \left(g_1^a g_2^b\right)^r = (g_1^r)^a\,(g_2^r)^b = u^a v^b$$

and the message is then recovered as

$$\frac{e}{u^x v^y} = \frac{h^r m}{(g_1^r)^x (g_2^r)^y} = \frac{h^r m}{(g_1^x g_2^y)^r} = m.$$

We now prove the security of the scheme.

Theorem 4 *Under the DDH Assumption, the above encryption scheme is secure against non-adaptive chosen-ciphertext attack.*

Proof The proof is very similar to the proof of the previous theorem. As there, assume we are given a PPT algorithm A attacking the above encryption scheme. We construct algorithm $\hat{A}$ trying to distinguish DDH tuples from random tuples. As in the previous proof, we will argue that if the tuple given to $\hat{A}$ is a DDH tuple, then the view of A is identical to its view when attaching the above encryption scheme. On the other hand, if the tuple given to $\hat{A}$ is a random tuple, then A will have no information about the message that is encrypted in an information-theoretic sense. The difference here is that we will be considering the more difficult case of CCA-1 security, and we must show that the queries made to the decryption oracle by A will not reveal anything. With this in mind, let us begin a formal proof.

Given some adversary A attacking the above encryption scheme via a non-adaptive chosen-ciphertext attack, we construct an adversary $\hat{A}$ as follows:

$$\begin{array}{l}
\underline{\hat{A}(g_1, g_2, g_3, g_4)} \\
x, y, a, b \leftarrow \mathbb{Z}_q \\
h = g_1^x g_2^y;\ c = g_1^a g_2^b \\
PK = \langle g_1, g_2, h, c \rangle \\
SK = \langle x, y, a, b \rangle \\
(m_0, m_1) \leftarrow A^{\mathcal{D}_{SK}(\cdot)}(PK) \\
b \leftarrow \{0, 1\} \\
C = \langle g_3,\ g_4,\ g_3^x g_4^y \cdot m_b,\ g_3^a g_4^b \rangle \\
b' \leftarrow A(PK, C) \\
\text{if } b = b', \text{ then guess “DDH tuple”} \\
\text{else guess “random tuple”}
\end{array}$$

Claim 5 *If $\hat{A}$ gets a DDH tuple, then A's view of the game is the same as in an execution of the real Cramer-Shoup-lite encryption scheme.*

A corollary of this claim is that

$$\Pr[\hat{A} \text{ outputs “DDH tuple”} \mid \text{DDH tuple}] = \Pr[b' = b \mid A \text{ attacks real scheme}].$$

We now prove the claim.

Certainly the public key created by $\hat{A}$ is exactly identical to the public key seen by A in a real execution of the encryption scheme. In fact, the secret key held by $\hat{A}$ is also identical to that used in a real execution of the encryption scheme, and thus the decryption queries of A are answered exactly as they would be in a real execution of the encryption scheme. The only thing left to examine is the ciphertext. But if the input to $\mathcal{A}$ is a DDH tuple, then we can write $g_3 = g_1^r$ and $g_4 = g_2^r$ where r is uniformly distributed in $\mathbb{Z}_q$. But then simple algebra shows that the ciphertext is distributed identically to the challenge ciphertext in a real execution of the encryption scheme (details left to the reader). □

Claim 6 *If $\hat{A}$ gets a random tuple, then (with all but negligible probability) A has no information about the bit b chosen by $\hat{A}$. We remark that this holds in an information-theoretic sense, for all-powerful A, as long as A can only make polynomially-many queries to the decryption oracle.*

Before proving the claim, we show how this claim completes the proof of the theorem. The claim implies that the probability that A correctly guesses b is negligibly close to $1/2$ and therefore $\Pr[\hat{A}$ outputs "DDH tuple" | random tuple] is negligibly close to $1/2$ as well. Thus, the advantage of $\hat{A}$ is negligibly close to:

$$\left|\Pr[b = b' \mid A \text{ attacks real scheme}] - 1/2\right|.$$

Since the DDH assumption implies that the advantage of $\hat{A}$ is negligible, this implies that the probability that A correctly guesses the value of b when attacking the real scheme is negligibly close to $1/2$. But this is exactly the definition of CCA-1 security.

We return to the proof of the claim. The proof is similar to the analogous claim proven previously, in that we argue that A's information about x and y will not be enough to determine which message was encrypted. But we have to be a little more careful here because A can now potentially learn additional information about x and y from the decryption oracle queries it makes.

Let (g_1, g_2, g_3, g_4) be a random tuple. As before, we may write these as:

$$\langle g_1, g_2 = g_2^{\alpha}, g_3 = g_1^{r}, g_4 = g_2^{r'} \rangle,$$

where with all but negligible probability $\alpha \neq 0$ and $r \neq r'$ (and we assume this from now on). From PK, adversary A learns that $h = g_1^x g_2^y$ and this constrains x, y according to:

$$\log_{g_1} h = x + (\log_{g_1} g_2) \cdot y = x + \alpha y \tag{3}$$

exactly as before.

We now consider what additional information A learns about x, y from its queries to the decryption oracle. When A makes a decryption oracle query (μ, ν, e, w) there are two cases: either there exists an r'' such that $\mu = g_1^{r''}$ and $\nu = g_2^{r''}$ (and hence this ciphertext is "legal"), or not. We call queries of the latter form "illegal". We first show that A only learns additional information about x, y if it makes an illegal query which is not rejected. But we next show that (with all but negligible probability) all A's illegal queries are rejected. Putting this together will imply that A does not learn additional information about x, y with all but negligible probability.

Claim 7 *A gets additional information about x, y only if it submits a decryption query (μ, ν, e, w) such that:*

1. *$\log_{g_1} \mu \neq \log_{g_2} \nu$ (i.e., an illegal query), and*
2. *$\mathcal{D}_{SK}(\cdot)$ does not return $\perp$.*

To see this, first suppose that $\mathcal{D}_{SK}(\cdot)$ returns $\perp$. The only time this happens is when the decryption routine rejects because w is not of the correct form. But this check only involves a and b, and hence cannot reveal any information about x, y.

Next suppose that A submits a query for which $\log_{g_1} \mu = \log_{g_2} \nu = r''$ for some arbitrary r''. In this case, based on the output m of the decryption oracle, A learns that $m = \frac{e}{\mu^x \nu^y}$.

Taking logarithms of both sides means that A learns the following linear constraint on x and y:

$$\begin{aligned}&\log_{g_1} m \\ &= \log_{g_1} e - (\log_{g_1} \mu)x - (\alpha \log_{g_2} \nu)y \\ &= \log_{g_1} e - r''x - \alpha r''y\end{aligned}$$

(note that e and m are known, so x and y are the only variables here). But this equation is linearly *dependent* on Equation (3). Thus, this does *not* introduce any additional constraint on x, y and hence the adversary has not learned any additional information about x, y. $\square$

Claim 8 *The probability that A submits a decryption query (μ, ν, e, w) for which $\log_{g_1} \mu \neq \log_{g_2} \nu$ but $\mathcal{D}_{SK}(\mu, \nu, e, w) \neq \perp$ is negligible.*

Let $\log_{g_1} \mu = r_1$ and $\log_{g_2} \nu = r_2$. In order for the decryption oracle to not reject, the adversary must "predict" the value of $\mu^a \nu^b$ (so that it can set w equal to this value). We show that it cannot do so with better than negligible probability.

Consider the information the adversary knows about a, b. From the public key, A learns that $c = g_1^a g_2^b$ and this constrains a, b according to:

$$\log_{g_1} c = a + (\log_{g_1} g_2) \cdot b = a + \alpha b\,. \tag{4}$$

The first time A makes an illegal decryption oracle query with $\log_{g_1} \mu \neq \log_{g_2} \nu$, the above equation represents all the information the adversary knows about a, b. Now, let w' be an arbitrary group element. The value of $\mu^a \nu^b$ is equal to w' exactly if:

$$\begin{aligned}&\log_{g_1} w' \\ &= a \log_{g_1} \mu + b \log_{g_1} \nu \\ &= r_1 \cdot a + \alpha r_2 \cdot b\,.\end{aligned} \tag{5}$$

But Equations (4) and (5) (viewed as equations in unknowns a, b over $\mathbb{Z}_q$) are linearly independent and hence have a solution in terms of a, b. Since this is true for *arbitrary* w', this means that any value of w' is possible (in fact, they are all equally likely) and hence A can only predict the correct value of w with probability $1/q$. (Note that this argument is substantially similar to the proof of Claim 3, above.)

Now, the above was true for the *first* illegal decryption query of A. However, each illegal decryption query of A *does* reveal some additional information about a, b. In particular, when an illegal query (μ, ν, e, w) is rejected the adversary learns that $w \neq \mu^a \nu^b$. At best, however, this eliminates one possibility for a, b. From Equation (4 alone, there are q possibilities for (a, b), and each rejected decryption query of A eliminates at most one of these solutions. Thus, at the time of the $(p+1)^{\text{th}}$ decryption query of A, assuming the first p of A's illegal decryption queries were rejected, there are (at least) $q - p$ possible solutions for (a, b). The argument of the previous paragraph now has to be modified to take this into account. But what we see is that eliminating one possibility for (a, b) has the effect of eliminating one possible value of w. So now the probability that A can correctly guess w is $1/(q - p)$.

Assume A makes a total of p decryption queries. Straightforward probability calculations show that the probability that *any* of A's illegal queries are *not* rejected is at most $p/(q-p)$ (in each of p illegal decryption queries, A has at best probability $1/(q-p)$ of the query not being rejected). But since p is polynomial and q is exponential, this is a negligible quantity. $\square$

Putting the above two claims together shows that, with all but negligible probability, A never learns any additional information about x, y beyond that implied by Equation (3). Assuming this is the case, an argument exactly like that given in the proof of Claim 3 shows that $g_1^x g_2^y$ is uniformly distributed in the group (from the point of view of A) and hence A has no information about the value of b. This completes the proof of Claim 6, and thus the proof of the theorem. $\blacksquare$

References

[1] R. Cramer and V. Shoup. A Practical Public-Key Cryptosystem Provably Secure Against Adaptive Chosen Ciphertext Attack. In *Adv. in Cryptology — CRYPTO 1998.*