Public-key Encryption December 19, 2012

Lecture 8

Lecturer: Dominique Schröder

1 Towards the Cramer-Shoup Encryption Scheme

1.1 Preliminaries

In this lecture, we are making the first step towards the efficient CCA2 secure encryption scheme that has been suggested by Cramer and Shoup [1]. Our construction and proof follows [2]. All schemes are secure under the DDH assumption:

Definition 1 The DDH problem is hard relative to $\mathcal{G}$ if for all PPT algorithms $\mathcal{A}$ there exists a negligible function $\varepsilon(\lambda)$ such that

$$|\operatorname{Prob}[\mathcal{A}(\mathbb{G}, q, g, g^x, g^y, g^z) = 1] - \operatorname{Prob}[\mathcal{A}(\mathbb{G}, q, g, g^x, g^y, g^{xy}) = 1]| \leq \varepsilon(\lambda)$$

where the probability is taken over the random coins of $\mathcal{G}$ and over the random choices of $x, y, z \in \mathbb{Z}_q$. ♢

Recall a variant of the El Gamal encryption scheme that has been discussed in Problem Set 5:

$\underline{\mathsf{Gen}(1^\lambda)}$
- $(\mathbb{G}, q, g_1, g_2) \leftarrow \mathcal{G}(1^\lambda)$
- $x, y \leftarrow \mathbb{Z}_p$
- $h \leftarrow g_1^x g_2^y$
- $dk := (\mathbb{G}, q, g_1, g_2, x, y)$
- $ek := (\mathbb{G}, q, g_1, g_2, h)$
- return (dk, ek)

$\underline{\mathsf{Enc}(ek, m)}$
- Parse ek as $(\mathbb{G}, q, g, h)$
- $r \leftarrow \mathbb{Z}_q$
- $C := (g_1^r, g_2^r, h^r \cdot m)$
- return C

$\underline{\mathsf{Dec}(dk, C)}$
- parse dk as $(\mathbb{G}, q, g_1, g_2, x, y)$
- parse C as (u, v, e)
- output $\frac{e}{u^x \cdot v^y}$

1.2 A Lite Version

We modify the scheme even further in order to obtain an encryption scheme that is CCA1 secure, i.e., secure against *non*-adaptive chosen-ciphertext attacks. The modified encryption scheme $\mathsf{PKE}_{\mathsf{cca1}} = (\mathsf{Gen}_{\mathsf{cca1}}, \mathsf{Enc}_{\mathsf{cca1}}, \mathsf{Dec}_{\mathsf{cca1}})$ is defined as follows (where we assume that $(\mathbb{G}, q)$ are system parameters, which are known to everybody):

$\underline{\mathsf{Gen}_{\mathsf{cca1}}(1^\lambda)}$

$(\mathbb{G}, q, g_1, g_2) \leftarrow \mathcal{G}(1^\lambda)$
$x, y \leftarrow \mathbb{Z}_p$
$a, b \leftarrow \mathbb{Z}_p$
$h \leftarrow g_1^x g_2^y$
$c \leftarrow g_1^a g_2^b$
$dk := (g_1, g_2, x, y, a, b)$
$ek := (g_1, g_2, h, c)$
return (dk, ek)

$\underline{\mathsf{Enc}_{\mathsf{cca1}}(ek, m)}$

parse ek as (g, h, c)
$r \leftarrow \mathbb{Z}_q$
$C := (g_1^r, g_2^r, h^r \cdot m, c^r)$
return C

$\underline{\mathsf{Dec}_{\mathsf{cca1}}(dk, C)}$

parse dk as (g_1, g_2, x, y, a, b)
parse C as (u, v, e, w)
if $w = u^a \cdot v^b$
    output $\frac{e}{u^x \cdot v^y}$
    else $\perp$

The scheme is complete, because for every honestly generated key and chiphertext the validity check holds

$$w = c^r = (g_1^a \cdot g_2^b)^r = (g_1^a)^r \cdot (g_2^b)^r = (g_1^r)^a \cdot (g_2^r)^b = u^a \cdot v^b.$$

Now, if the validity check holds, then the decryption algorithm returns the message:

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

Theorem 1 *If the DDH assumption holds relative to $\mathcal{G}$, then the public-key encryption scheme $\mathsf{PKE}_{\mathsf{cca1}} = (\mathsf{Gen}_{\mathsf{cca1}}, \mathsf{Enc}_{\mathsf{cca1}}, \mathsf{Dec}_{\mathsf{cca1}})$ has indistinguishable encryption under non-adaptively chosen ciphertext attacks.*

Proof The proof is similar to the one from the problem set, with the difference that we have to show that the access to the decryption oracle in the first stage of the game, does not help to distinguish two ciphertexts.

More formally, assume towards contradiction that there exists an efficient adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ that predicts the bit b in the $\mathsf{IND\text{-}CCA1}^{\mathsf{PKE}}_{\mathcal{A}}(\lambda)$ game with probability $1/2 + \delta(\lambda)$, where δ is non-negligible. Then, we show how to construct a distinguisher D against the DDH assumption. Recall that the distinguisher against the DDH assumption gets as input a tuple (g, g^x, g^y, g^z) and it has to decide if z is a random element, or if $z = xy$. By real we denote the event that $z = xy$ and fake is the event that z is a random element.

The main idea of the reduction is to create an honestly computed ciphertext whenever the reduction's input is a real DDH tuple, and to run $\mathcal{A}_1$ on a ciphertext that information theoretically hides the message otherwise. More formally:

$\underline{D(g_1, g_2, g_3, g_4)}$

$x, y, a, b \leftarrow \mathbb{Z}_p$

$h \leftarrow g_1^x g_2^y$

$c \leftarrow g_1^a g_2^b$

$ek := (g_1, g_2, h, c)$

$dk := (x, y, a, b)$

$(m_0, m_1) \leftarrow \mathcal{A}_0^{\mathcal{D}(dk,\cdot)}(ek)$

$b \leftarrow \{0, 1\}$

$C \leftarrow (g_3, g_4, g_3^x \cdot g_4^y \cdot m_b, g_3^a \cdot g_4^b)$

$b' \leftarrow \mathcal{A}_1(C)$

if $b = b'$ return 1, else 0.

Claim 2 $|\text{Prob}[D \textit{ outputs } 1 \mid \mathsf{real}] - \text{Prob}[D \textit{ outputs } 1 \mid \mathsf{fake}]| \approx 0.$

Proof This claim follows directly from the DDH assumption. ■

We analyze both cases independently. The next claim shows that if (g_1, g_2, g_3, g_4) is a DDH tuple, i.e., the event real happens, then D simulates the CCA1 game perfectly.

Claim 3 $\text{Prob}[D \textit{ outputs } 1 \mid \mathsf{real}] = \text{Prob}\left[\mathsf{IND\text{-}CCA1}_{\mathcal{A}}^{\mathsf{PKE}}(\lambda) = 1\right].$

Proof Both the encryption and the decryption key are identical to the ones in the real scheme. Thus, D can simulate the decryption oracle perfectly and $\mathcal{A}_0$'s view is identical to its view when it is run against the real encryption scheme. Now, we claim that the challenge ciphertext has the same distribution as an honestly generated one, if D's input is a real DDH tuple. To see this observe that there exists a pair (α, r) such that:

$$(g_1, g_2 := g_1^\alpha, g_3 := g_1^r, g_4 := g_1^{\alpha r} = g_2^r).$$

Therefore, we can re-write the public key as

$$ek = (g_1, g_2, h = g_1^x \cdot g_2^y, c = g_1^a \cdot g_2^b)$$

and the ciphertext as

$$C = (g_1^r, g_2^r, (g_1^r)^x (g_2^r)^y \cdot m, (g_1^r)^a (g_2^r)^b) = (g_1^r, g_2^r, (g_1^x g_2^y)^r \cdot m, (g_1^a g_2^b)^r)$$

for some r uniformly distributed in $\mathbb{Z}_q$. ■

The next claim considers the case where D is given a random DDH tuple.

Claim 4 $\text{Prob}[D \textit{ outputs } 1 \mid \mathsf{fake}] = \frac{1}{2}.$

Proof If $D's$ input is a fake DDH tuple, then there exists a tuple (α, r, β) chosen at random such that:

$$(g_1, g_2 := g_1^{\alpha}, g_3 := g_1^{r}, g_4 := g_1^{\beta}).$$

Since (α, r, β) are chosen uniformly at random, it follows that $\beta \neq \alpha r \pmod{q}$ and $\alpha \neq 0$ with all but negligible probability. Thus, in the following we assume that this is the case and we can re-write the equation such that there exists two distinct values $r, r' \in \mathbb{Z}_q$ such that

$$(g_1, g_2 := g_1^{\alpha}, g_3 := g_1^{r}, g_4 := g_2^{r'}).$$

Now, we analyze the probability that D outputs 1. It follows from our construction that D outputs 1 whenever $\mathcal{A}_1$ outputs 1, thus $\text{Prob}[\,D \text{ outputs } 1] = \text{Prob}[\,\mathcal{A}_1 \text{ outputs } 1]$. In the following, we show that $\mathcal{A}_1$ outputs 1 with probability $1/2$ even if $\mathcal{A}$ is a computationally unbounded adversary that queries the decryption oracle polynomially many times. The input of $\mathcal{A}_0$ is a public key $ek = (g_1, g_2, h, c)$, where $h = g_1^x g_2^y$. We argue that the information that $\mathcal{A}$ learns from ek about x, y is not sufficient to learn which message was encrypted. Since $\mathcal{A}$ is computationally unbounded, it can compute discrete logarithms and it learns from the public key that $\log_{g_1} g_2 = \alpha$ and that

$$\log_{g_1} h = x + y\alpha. \tag{1}$$

Since α is fixed, this collapses the space of (x, y) into q possible pairs, one for each value x and y, respectively. In the following, we analyze the amount of information that $\mathcal{A}_0$ learns about (x, y) from its decryption queries. Let (μ, ν, e, w) be an arbitrary query by $\mathcal{A}_0$ to its decryption oracle. For this query, we distinguish between two cases:

Valid ciphertexts: We say that if a ciphertext is valid, then there exists a value r'' such that $\mu = g_1^{r''}$ and $\nu = g_2^{r''}$;

Invalid ciphertexts: otherwise, the ciphertext is invalid.

Our proof now proceeds with two claims. The first one shows that the distribution of the bit b is independent of $\mathcal{A}$'s view, if the decryption oracle rejects all invalid ciphertexts. This means that the adversary does not learn any information from valid ciphertext queries. The second claim then shows that all invalid ciphertext queries are rejected.

Claim 5 *The adversary A_0 learn additional information about (x, y) only if it queries (μ, ν, e, w) to the decryption oracle such that*

1. $\log_{g_1} \mu \neq \log_{g_2} \nu$ *and*
2. $\mathcal{D}(dk, \cdot)$ *does not return* $\perp$.

Proof If $\mathcal{D}(dk, \cdot)$ returns $\perp$, then this happens whenever $w \neq \mu^a \cdot \nu^b$. Since this check only involves (a, b) the attacker does not learn any information about (x, y).

Now, assume that $\mathcal{A}_0$ submits a valid ciphertext, i.e., there exists a value r'' such that $\log_{g_1} \mu \neq \log_{g_1} \nu = r''$. In this case, the adversary obtains

$$m = \frac{e}{\mu^x \nu^y}$$

from the decryption oracle. Taking logarithms on both sides of the equation yields the information that $\mathcal{A}_0$ learn. In fact, $\mathcal{A}_0$ learns the following linear constraint on x and y

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

where (x, y) are the only variables, because m and e are known. Since equation is linear *dependent* on the equation (1), it does not introduce any additional constraint on (x, y) and thus, the attacker $\mathcal{A}_0$ does not learn any information about (x, y). ■

The next claim shows that with all but negligible probability the adversary cannot submit invalid ciphertexts that pass the consistency check.

Claim 6 *The probability that $\mathcal{A}_0$ submits a query (μ, ν, e, w) such that*

1. $\log_{g_1} \mu \neq \log_{g_2} \nu$ *and*
2. $\mathcal{D}(dk, \cdot)$ *does not return* $\perp$

is negligible.

Proof According to the construction, the decryption oracle only answers queries if $w = \mu^a \cdot \nu^b$. Now, consider the information that $\mathcal{A}_0$ has about (a, b) from the public key. Since $\mathcal{A}$ is unbounded, it can compute the discrete logarithm of $c = g_1^a g_2^b$ which constrains (a, b) according to

$$\log_{g_1} c = a + (\log_{g_1} g_2) \cdot b = a + \alpha b. \tag{2}$$

Now, let $\log_{g_1} \mu = r_1$ and $\log_{g_2} \nu = r_2$ if w' is an arbitrary group element, then the value $\mu^a \cdot \nu^b$ equals to w' if

$$\log_{g_1} w' = a \log_{g_1} \mu + b \log_{g_2} \nu \tag{3}$$

$$= r_1 \cdot a + \alpha r_2 \cdot b \tag{4}$$

Since the equations (2) and (4) are linearly independent in the unknowns a and b over $\mathbb{Z}_q$, they have solutions in terms of a and b. Thus, the value $\mu^a \cdot \nu^b$ is uniformly distributed in $\mathbb{G}$ and $\mathcal{A}_0$ can only guess this value with probability $1/q$. This, however, is only true as long as the decryption oracle does not reject the query. In fact, whenever $\mathcal{D}$ returns $\perp$, then $\mathcal{A}_0$ learns that $w \neq \mu^a \cdot \nu^b$. This, however, eliminates only one from the q possible solutions for (a, b). Thus, if we assume that $\mathcal{A}_0$ made p of these queries (assuming they were all rejected) means that there are still $q - p$ possibilities left, which means that $\mathcal{A}_0$ can guess the value of w with probability $1/(q - p)$. Now, if we assume that $\mathcal{A}_0$ makes a total number of p decryption queries, then the probability that any of $\mathcal{A}$'s invalid ciphertexts are not rejected is at most $p/(q - p)$. Thus, the probability that $\mathcal{A}_0$ submits such a query is negligible because q is exponential and p is polynomially bounded. ■

Now, Claim 4 follows from Claims 5 and 6. ■

Finally, the proof of the theorem follows easily from Claims 3 and 4, because the success probability of $\mathcal{A}$ when giving a real DDH tuple carries over. Thus, if $\mathcal{A}$ win the game in this case with non-negligible probability better than guessing, then we have a distinguisher against the DDH assumption (contradicting Claim 2). ■

References

[1] R. Cramer and V. Shoup, A Practical Public-Key Encryption Cryptosystem Provably Secure Against Adaptive Chosen Ciphertext Attack, In Adv. in Cryptology — CRYPTO 1998. Full version available from http://eprint.iacr.org/1998/006.ps

[2] J. Katz, Lecture Notes *Advanced Cryptography* CMSC 858K, Spring 2004.