CS 7810 Graduate Cryptography November 20, 2017

Lecture 21: CCA2 Security for the Cramer-Shoup Cryptosystem

Lecturer: Daniel Wichs *Scribe: Jack Doerner*

1 Topics Covered

2. Definitions
3. Remembering Cramer-Shoup
4. Proof of IND-CCA1 Security and Disproof of IND-CCA2 Security
5. Revising Cramer-Shoup
6. IND-CCA2 Security

2 Definitions

Definition (Chosen Ciphertext Attack)**.**

A Chosen Ciphertext Attack (CCA or CCA1) is a security game wherein an adversary with oracle access to a decryption function attempts to defeat the security of the encryption scheme to which that function belongs. In essence, the adversary may choose polynomially many arbitrary ciphertexts, and receive the plaintexts to which those ciphertexts decrypt under a specific key. The adversary is static, and therefore it may no longer access the decryption oracle once it has challenged the encryption scheme and received a reply. There are two notions of security defined under this game: ciphertext indistinguishability (IND-CCA Security) and ciphertext non-malleability (NM-CCA Security). Note that IND-CCA2 implies IND-CCA, and NM-CCA2 implies NM-CCA. Furthermore, NM-CCA implies IND-CCA, but the converse is not true.

Definition (Adaptive Chosen Ciphertext Attack)**.**

An Adaptive Chosen Ciphertext Attack (CCA2) is a security game wherein an adversary with oracle access to a decryption function attempts to defeat the security of the enryption scheme to which that function belongs. In essence, the adversary may choose polynomially many arbitrary ciphertexts, and receive the plaintexts to which those ciphertexts decrypt under a specific key. Because the adversary is adaptive, it may continue acccessing this oracle even after challenging the game and receiving a response. There are two notions of security defined under this game: ciphertext indistinguishability (IND-CCA2 Security) and ciphertext non-malleability (NM-CCA2 Security). Note that NM-CCA2 implies IND-

CCA2 and vice versa.

3 Remembering Cramer-Shoup

First, recall the Cramer-Shoup cryptosystem, which has its basis in the notion of a Hash-Proof system.

3.1 Hash-Proof Systems

Note that what follows suffices for Cramer-Shoup, but is not a fully general definition. Suppose there exists some public description of a group G of order q (such that q is represented in n bits) generated by the element g. For some $h = g^\beta$ such that $\beta \neq 0 \in \mathbb{Z}_q$ we define a language L over (G, q, g, h):

$$L = \left\{(g^r, h^r) : r \in \mathbb{Z}_q\right\}$$

We also define a complement language:

$$\bar{L} = \left\{(g^{r_1}, h^{r_2}) : r_1, r_2 \in \mathbb{Z}_q, r_1 \neq r_2\right\}$$

We can construct a Hash-Proof system as a tuple of algorithms $\left(\text{Gen}, H_{\mathsf{pk}}, H_{\mathsf{sk}}\right)$. The generator drawns $x, y \leftarrow \mathbb{Z}_q$ and then returns $\mathsf{pk} = g^x h^y, \mathsf{sk} = (x, y)$, and the secret and public-key hash functions work in the following way:

$$H_{\mathsf{pk}}\left((c_1, c_2), r\right) = \mathsf{pk}^r$$
$$H_{\mathsf{sk}}\left((c_1, c_2)\right) = c_1^x \cdot c_2^y$$

This scheme has the following properties, to which we will later refer by number

1. $U_L \stackrel{c}{\equiv} U_{\bar{L}}$; that is, the uniform distribution over the language L is computationally indistinguishable from the uniform distribution over its complement.
2. $(c_1, c_2) \in L \Rightarrow H_{\mathsf{pk}}\left((c_1, c_2), r\right) = H_{\mathsf{sk}}\left(c_1, c_2\right)$ where r is the witness for (c_1, c_2); i.e. correctness.
3. $\forall(\mathsf{pk}, \mathsf{sk}) \leftarrow \text{Gen}(1^n) \forall c = (c_1, c_2) \in \bar{L}, H_{\mathsf{sk}}(c) \equiv U_G$; that is, for elements not in the language L, the output of the secret-key hash function is statistically equivalent to the uniform distribution over the group G.

3.2 The Cramer-Shoup Cryptosystem

Assume the existence of a public key $\mathsf{pk} = (\mathsf{pk}_1, \mathsf{pk}_2)$ where pk_1 and pk_2 are hash-proof public keys, and a corresponding secret key $\mathsf{sk} = (\mathsf{sk}_1, \mathsf{sk}_2)$. The Cramer-Shoup cryptosystem consists of the algorithms (Enc, Dec) which work as follows:

Algorithm 1. $\text{Enc}_{\mathsf{pk}}(m)$**:**

1. $r \leftarrow \mathbb{Z}_q$
2. $c \coloneqq (g^r, h^r) \in L$

3. $h_1 := H_{\mathsf{pk}_1}(c, r) \cdot m$
4. $h_2 := H_{\mathsf{pk}_2}(c, r)$
5. output (c, h_1, h_2)

Algorithm 2. $\mathrm{Dec}_{\mathsf{sk}}\big((c, h_1, h_2)\big)$**:**

1. if $h_2 = H_{\mathsf{sk}_2}(c)$ output $\big(h_1 / H_{\mathsf{sk}_1}(c)\big)$
2. otherwise output $\perp$

3.3 The CCA1/2 Indistinguishability Game

The game $\mathrm{CCAGame}^b$ for indistinguishability of public key encryption is played by an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$ in the following way:

1. $\mathcal{C}$ runs $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathrm{Gen}(1^n)$ and sends pk to $\mathcal{A}$
2. $\mathcal{A}$ may send polynomially many ciphertexts ct_i to $\mathcal{C}$, where each has an index i, and receive after each a message m_i which is the decryption of ct_i under sk.
3. $\mathcal{A}$ sends a pair of challenge messages, m_0^*, m_1^* to $\mathcal{C}$, and recieves $\mathsf{ct}^* \leftarrow \mathrm{Enc}_{\mathsf{pk}}(m_b^*)$ in return.
4. If the parties are playing the CCA2 game, $\mathcal{A}$ may again send polynomially many ciphertexts ct_i to $\mathcal{C}$, and receive after each a message m_i which is the decryption of ct_i under sk. However, $\mathcal{A}$ is forbidden to query the challenge ciphertext ct^*. If the parties are playing the CCA1 game, then this step is skipped.
5. $\mathcal{A}$ attemps to determine whether it is playing the game with $b = 0$ or $b = 1$. It sends its guess for b to $\mathcal{C}$, and wins if it guesses correctly.

4 Proof of IND-CCA1 Security and Disproof of IND-CCA2 Security

4.1 Proof of IND-CCA1 Security

IND-CCA1 Security for the Cramer-Shoup cryptosystem is proven by showing that $\mathrm{CCA1Game}^0 \stackrel{c}{\equiv} \mathrm{CCA1Game}^1$, i.e. the adversary's view of the game with $b = 0$ is indistinguishable from its view of the game with $b = 1$, via a series of hybrids:

1. $\mathcal{H}_0^b$ is the ordinary $\mathrm{CCA1Game}^b$.
2. $\mathcal{H}_1^b$: When computing the challenge ciphertext ct^*, instead of outputting

$$(c, H_{\mathsf{pk}_1}(c, r) \cdot m_b^*, H_{\mathsf{pk}_2}(c, r))$$

as would normally be the case for Cramer-Shoup, $\mathcal{C}$ outputs

$$(c, H_{\mathsf{sk}_1}(c) \cdot m_b^*, H_{\mathsf{sk}_2}(c))$$

Due to Property 2 (correctness) of Hash-Proof Systems, $\mathcal{H}_0^b \equiv \mathcal{H}_1^b$, but note that $\mathcal{C}$ no longer uses the witness r in computing the encryption of m_b^*.

3. $\mathcal{H}_2^b$: When computing the challenge ciphertext ct^*, instead of drawing $c \leftarrow L$ as would normally be the case for Cramer-Shoup, $\mathcal{C}$ chooses $c \leftarrow \bar{L}$. Due to Property 1 of Hash-Proof Systems, $\mathcal{H}_1^b \stackrel{c}{\equiv} \mathcal{H}_2^b$.

4. $\mathcal{H}_3^b$: When $\mathcal{C}$ receives a decryption query for a ciphertext $\mathsf{ct}_i = (c, h_1, h_2)$, rather than rejecting the query if $h_2 \neq H_{\mathsf{sk}_2}(c)$, it rejects if $c \notin L$. Note that this requires $\mathcal{C}$ to break discrete logarithms, and that in $\mathcal{H}_2^b$, if $\mathcal{A}$ chooses $c \notin L$ and manages to guess $H_{\mathsf{sk}_2}(c)$, then it will receive a (nonsense) decryption, whereas in $\mathcal{H}_3^b$ its query will be rejected because $c \notin L$, which would allow it to distinguish the two hybrids. Due to Property 3 of Hash-Proof Systems, even a computationally-unbounded $\mathcal{A}$ has a negligible probability of guessing correctly. Therefore $\mathcal{H}_2^b \stackrel{s}{\equiv} \mathcal{H}_3^b$.

5. $\mathcal{H}_4^b$: When computing the challenge ciphertext $\mathsf{ct}^* = (c, h_1, h_2)$, rather than computing $h_1 := H_{\mathsf{sk}_i}(c) \cdot m_b^*$, $\mathcal{C}$ draws $h_1 \leftarrow U_G$. Due to Property 3 of Hash-Proof Systems, $\mathcal{H}_3^b \equiv \mathcal{H}_4^b$. Finally it is trivially true that $\mathcal{H}_4^0 \equiv \mathcal{H}_4^1$, which implies that $\mathcal{H}_0^0 \stackrel{c}{\equiv} \mathcal{H}_0^1$

4.2 . . . and Disproof of CCA2 Security

Intuitively, Cramer-Shoup is not secure against Adaptive Chosen Ciphertext attacks because the ciphertexts produced are malleable. Consequently, an adversary can re-randomize a ciphertext and then ask for a decryption of that ciphertext. Since the re-randomized ciphertext is not equal to the original, it will not trigger the condition forbidding a query to the original, but because it decrypts to the same value as the original, the adversary can use it to disginguish between CCA2Game^0 and CCA2Game^1.

Specifically, suppose $\mathcal{A}$ receives

$$\mathsf{ct}^* = (c = (g^r, h^r), \mathsf{pk}_1^r \cdot m_b, \mathsf{pk}_2^r)$$

It can then choose $r' \leftarrow \mathbb{Z}_q$ and calculate

$$\mathsf{ct}^{*\prime} := \left(c = \left(g^{r+r'}, h^{r+r'}\right), \mathsf{pk}_1^{r+r'} \cdot m_b, \mathsf{pk}_2^{r+r'}\right)$$

which is also a valid encryption of m_b

The flaw in the IND-CCA1 proof that prevents it from showing IND-CCA2 can be found in $\mathcal{H}_3^b$. As previously stated, $\mathcal{H}_3^b \stackrel{s}{\equiv} \mathcal{H}_2^b$ because even a computationally unbounded adersary is negligibly likely to guess an accepting value of $H_{\mathsf{sk}_1}(c)$ given $c \leftarrow \bar{L}$. However, by giving the adversary ct^*, we give it enough information to calculate such a value with probability 1 via above method. Thus, in the CCA2Game, $\mathcal{H}_3^b$ and $\mathcal{H}_2^b$ are easily distinguished.

In order do address this flaw, we need to strengthen Property 3 of Hash-Proof Systems. Specifically, we need a property of the following form:

3. $\forall(\mathsf{pk},\mathsf{sk}) \leftarrow \text{Gen}(1^n) \forall c = (c_1, c_2), c' = (c'_1, c'_2) \in \bar{L}$ s.t. $c \neq c'$, $\left(H_{\mathsf{sk}}(c), H_{\mathsf{sk}}(c')\right) \stackrel{c}{\equiv} U_{G^2}$; that is, for any pair of elements not in the language L, the joint distribution of outputs of the secret-key hash function on those elements is computationally indistinguishable from the uniform distribution over the two elements of group G.

5 Revising Cramer-Shoup

We will now revise the Cramer-Shoup Cryptosystem to achieve IND-CCA2 Security. We begin by constructing a stronger Hash-Proof System that satisfies our revised Property 3. We will refer to this scheme as HPS2, whereas the original is HPS1.

5.1 A New Hash-Proof Construction

Assume that there exists some Collision Resistant Hash Function (CRHF) $H_{\mathsf{CR}}(\cdot)$. We construct a new Hash-Proof System (HPS2) as a tuple of algorithms $\left(\text{Gen}, H'_{\mathsf{pk}}, H'_{\mathsf{sk}}\right)$ such that the generator drawns $x_1, y_1, x_2, y_2 \leftarrow \mathbb{Z}_q$ and then returns $\mathsf{pk} = \left(f_1 = g^{x_1}h^{y_1}, f_2 = g^{x_2}h^{y_2}\right), \mathsf{sk} = (x_1, y_1, x_2, y_2)$, and the secret and public-key hash functions work in the following way:

$$H'_{\mathsf{pk}}\left(c = (c_1, c_2), r, \alpha\right) = f_1^r f_2^{H_{\mathsf{CR}}(c\|\alpha)\cdot r}$$
$$H'_{\mathsf{sk}}\left(c = (c_1, c_2), \alpha\right) = c_1^{x_1 + H_{\mathsf{CR}}(c\|\alpha)\cdot x_2} \cdot c_2^{y_1 + H_{\mathsf{CR}}(c\|\alpha)\cdot y_2}$$

5.2 Revised Cramer-Shoup Cryptosystem

Assume the existence of a public key $\mathsf{pk} = (\mathsf{pk}_1, \mathsf{pk}_2)$ such that $(\mathsf{pk}_1, \mathsf{sk}_1)$ are a keypair for HPS1, H, and $(\mathsf{pk}_2, \mathsf{sk}_2)$ are a keypair for HPS2, H'. The Revised Cramer-Shoup cryptosystem consists of the algorithms $(\text{Enc}', \text{Dec}')$ which work in the following way:

Algorithm 3. $\text{Enc}'_{\mathsf{pk}}(m)$**:**

1. $r \leftarrow \mathbb{Z}_q$
2. $c \coloneqq (g^r, h^r) \in L$
3. $h_1 \coloneqq H_{\mathsf{pk}_1}(c, r) \cdot m$
4. $h_2 \coloneqq H'_{\mathsf{pk}_2}(c, r, h_1)$
5. output (c, h_1, h_2)

Algorithm 4. $\text{Dec}'_{\mathsf{sk}}\left((c, h_1, h_2)\right)$**:**

1. if $h_2 = H'_{\mathsf{sk}_2}(c, h_1)$ output $\left(h_1 / H_{\mathsf{sk}_1}(c)\right)$
2. otherwise output $\perp$

6 IND-CCA2 Security

6.1 Revisiting the Attack

Consider the attack from earlier, wherein an adversary mangles a ciphertext to generate a new ciphertext that decrypts to the same value. Under our revised encryption scheme, if the adversary alters c or m, then the value of h_1 changes, as, therefore, does $H_{\mathsf{CR}}(c\|h_1)$ (if the latter value does not, then the collision resistance of H_{CR} has been violated). The adversary now has

$$h_2 = f_1^r f_2^{r \cdot H_{\mathsf{CR}}(c\|h_1)}$$

and wishes to calculate

$$h_2' = f_1^{r+r'} f_2^{(r+r') \cdot H_{\mathsf{CR}}\left(c\|h_1'\right)}$$

To do this, it must calculate

$$f_2^{-r \cdot H_{\mathsf{CR}}(c\|h_1)}$$

which implies breaking discrete logarithm to find r. Thus, this attack is now impossible for any computationally bounded adversary.

6.2 Proof of Strengthened Property 3 for HPS2

Theorem 1. *Over all* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathrm{Gen}(1^n)$ *and for all probabilistic polynomial time adversaries* $\mathcal{A}$ *and any* $(c, c', \alpha, \alpha') \leftarrow \mathcal{A}(1^n)$ *such that* $c = (c_1, c_2), c' = (c_1', c_2') \in \bar{L}$ *and* $c \neq c'$, $\left(H'_{\mathsf{sk}}(c, \alpha), H'_{\mathsf{sk}}(c', \alpha')\right) \stackrel{c}{\equiv} U_{G^2}$

Proof. Consider the view of a adversary with access to the output of $H'_{\mathsf{sk}}(c, \alpha)$, $H'_{\mathsf{sk}}(c', \alpha')$ such that $c \neq c'$ and $c\|\alpha$ does not collide with $c'\|\alpha'$ under H_{CR}, but c, c', α, α' are otherwise arbitrarily and adversarially chosen. The adversary also has knowledge of the public key, $\mathsf{pk} = (f_1, f_2)$. We will show that if the four elements of the secret key (x_1, y_1, x_2, y_2) are random and independent (as they will be if they were produced by Gen), then $(f_1, f_2, H'_{\mathsf{sk}}(c, \alpha), H'_{\mathsf{sk}}(c', \alpha'))$ must also be. First, recall what you already know, but this time in matrix form:

$$\log_g \left(\begin{bmatrix} f_1 \\ f_2 \\ H'_{\mathsf{sk}}(c, \alpha) \\ H'_{\mathsf{sk}}(c', \alpha') \end{bmatrix} \right) = \begin{bmatrix} 1 & \beta & 0 & 0 \\ 0 & 0 & 1 & \beta \\ r_1 & r_2\beta & r_1\gamma & r_2\beta\gamma \\ r_1' & r_2'\beta & r_1'\gamma' & r_2'\beta\gamma' \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \end{bmatrix}$$

where

$$\gamma = H_{\mathsf{CR}}\left(c\|\alpha\right)$$
$$\gamma' = H_{\mathsf{CR}}\left(c'\|\alpha'\right)$$

If the matrix (hereafter referred to as A) is invertible, then it is a permutation (i.e. a 1-to-1 function), which implies that if (x_1, y_1, x_2, y_2) are random and independent then

$(f_1, f_2, H'_{\mathsf{sk}}(c, \alpha),\ H'_{\mathsf{sk}}(c', \alpha'))$ are as well. A matrix is invertible if and only if it has a non-zero determinant. The determinant of this matrix is

$$\begin{aligned}\det(A) &= \left(r_1 r'_1 \gamma - r_1 r'_1 \gamma' + r_2 r'_1 \gamma - r_2 r'_1 \gamma' + r_1 r'_2 \gamma - r_1 r'_2 \gamma' + r_2 r'_2 \gamma - r_2 r'_2 \gamma'\right) \beta^2 \\ &= (r_1 - r_2)(r'_1 - r'_2)(\gamma - \gamma')\beta^2\end{aligned}$$

In our theorem statement we have assumed that $c, c' \in \bar{L}$; therefore we know that $r_1 \neq r_2$ and $r'_1 \neq r'_2$. We have also assumed that $\beta \neq 0$. Due to the Collision Resistance property of H_{CR}, no computationally bounded adversary can find two inputs such that $\gamma = \gamma'$. Consequently, for all probabilistic polynomial time adversaries the matrix A will be invertible, which implies that if the secret key is uniformly and independently chosen, then $(f_1, f_2, H'_{\mathsf{sk}}(c, \alpha), H'_{\mathsf{sk}}(c', \alpha'))$ must be uniform and independent. □

6.3 Proof of IND-CCA2 Security for Revised Cramer-Shoup

The proof is exactly the same as the proof of IND-CCA1 security for standard Cramer-Shoup, except for the logic associated with the third hybrid:

4. $\mathcal{H}'^b_3$: When $\mathcal{C}$ receives a decryption query for a ciphertext $\mathsf{ct}_i = (c, h_1, h_2)$, rather than rejecting the query if $h_2 \neq H_{\mathsf{sk}_2}(c, h_1)$, it rejects if $c \notin L$. Note that this requires $\mathcal{C}$ to break discrete logarithms, and that in $\mathcal{H}^b_2$, if $\mathcal{A}$ chooses $c \notin L$ and manages to guess $H_{\mathsf{sk}_2}(c, h_1)$, then it will receive a (nonsense) decryption, whereas in $\mathcal{H}^b_3$ its query will be rejected because $c \notin L$, which would allow it to distinguish the two hybrids. Due to the Strengthened Property 3 of HPS2, no PPT adversary $\mathcal{A}$ can find $H'_{\mathsf{sk}_2}(c, h_1)$, even if given $H'_{\mathsf{sk}_2}(c', h'_1)$ for some other c', h'_1 of its choosing. Therefore $\mathcal{H}^b_2 \stackrel{c}{\equiv} \mathcal{H}'^b_3$.