# Lecture 7.2 — Hash Proof Systems and CCA-Security

We are going to build a CCA-secure public-key encryption scheme without random oracles, with security based on the DDH assumption. While the resulting scheme appears more complex than those we studied in the random-oracle model, it is actually fairly efficient and in particular it usually has shorter ciphertexts in practice for comparable security levels.

## 1 Hash-proof Systems

### 1.1 Definitions

We introduce the (somewhat abstract) concept of a hash-proof system. In particular, we need to consider first the concept of an NP-language $L$. This is a language $L$ where each element $x \in L$ has a witness $w$ that it belongs to the language. In particular, such a language comes with an efficiently computable relation $R_L$ such that $R_L(x, w)$ is true if and only if $w$ is a witness for $x$. We write, for convenience, $(x, w) \in L$ to mean that $x \in L$ and $w$ is a witness. The following are some cryptographic examples:

- The language $L$ where $N = P \cdot Q$ is an integer which is the product of two primes. (A witness would be $w = (P, Q)$, noting that primality can be checked in polynomial time.)

- Below, our examples will be focused around the language $L^*$ we assume we are given a group $G$ with prime order $q$, and efficient group operation in it. Note that this means that all elements in $G$ (except the unit element) are generators. Let $g_1 \neq g_2$ be two such generators. Then, let
$$L^* = \{(g_1^r, g_2^r) \,:\, r \in \mathbb{Z}_q\} \ .$$

**Definition 1.1** (Hash-proof system)**.** A *hash-proof system* (HPS) for an NP-language $L$ (with a relation $R_L$) is a triple of efficient algorithms $\mathsf{HPS} = (\mathsf{Kg}, \mathcal{P}, \mathcal{V})$ such that:

- $\mathsf{Kg}$ is the *key generation* algorithm, and on input $1^\lambda$, it outputs a secret-key / public-key pair $(\mathsf{pk}, \mathsf{sk})$.

- $\mathcal{P}$ is the (deterministic) prover algorithm, and on input $\mathsf{pk}$, as well as $x \in L$ with a witness $w$, and outputs a *proof* $\pi \in \{0,1\}^n$ (for some $n$, polynomial in $\lambda$)

- $\mathcal{V}$ is the (deterministic) verifier algorithm: It takes as input $\mathsf{sk}$ and some string $x$ (not necessarily $x \in L$), and outputs a string $\widetilde{\pi}$.

The HPS needs to satisfy correctness. Namely, for all $(x, w) \in L$, we have that

$$\mathsf{Pr}\left[(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{Kg}(1^\lambda),\ \pi \leftarrow \mathcal{P}(\mathsf{pk}, x, w),\ \widetilde{\pi} \leftarrow \mathcal{V}(\mathsf{sk}, x) :\ \widetilde{\pi} = \pi\right] = 1 \ .$$

$\Diamond$

### 1.2 Security of HPSs: 1-Universality

Intuitively, what we want is that if $x \notin L$, then there exists no way for an adversary to come up with some "fake proof" $\pi$ such that $\mathcal{V}(\mathsf{sk}, x) = \pi$. We will ask for something more concrete which easily implies this, and which we call 1-universality.

**Definition 1.2** (1-universality)**.** We say that $\mathsf{HPS} = (\mathsf{Kg}, \mathcal{P}, \mathcal{V})$ for an NP-language $L$ is 1-*universal* if the following the distributions of the following pairs are identical for all $x \notin L$:

- $(\mathsf{pk}, \widetilde{\pi})$, where $(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{Kg}(1^\lambda)$ and $\widetilde{\pi} \xleftarrow{\$} \mathcal{V}(\mathsf{sk}, x)$.
- $(\mathsf{pk}, z)$, where $(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{Kg}(1^\lambda)$ and $z \xleftarrow{\$} \{0,1\}^n$.

$\Diamond$

A 1-UNIVERSAL HPS FOR $L^*$ We give the following construction of a 1-universal HPS for the language $L^*$ given above:

**Procedure** $\mathsf{Kg}(1^\lambda)$:
$x_1, x_2 \xleftarrow{\$} \mathbb{Z}_q$
$\mathsf{pk} \leftarrow g_1^{x_1} g_2^{x_2}$
$\mathsf{sk} \leftarrow (x_1, x_2)$
**return** $(\mathsf{pk}, \mathsf{sk})$

**Procedure** $\mathcal{P}(\mathsf{pk}, (u_1, u_2), r)$:
$\pi \leftarrow \mathsf{pk}^r$
**return** $\pi$

**Procedure** $\mathcal{V}(\mathsf{sk} = (x_1, x_2), (u_1, u_2))$:
$\pi \leftarrow u_1^{x_1} u_2^{x_2}$
**return** $\pi$

It is easy to verify correctness. To prove 1-universality, note the following. Assume that $(u_1, u_2) \notin L$, i.e., $u_1 = g_1^{r_1}$, $u_2 = g_2^{r_2}$, for $r_1 \neq r_2$. Moreover, note that we can write $g_2 = g_1^\alpha$ for some $\alpha \in \mathbb{Z}_q$, $\alpha \neq 0, 1$. Then, we have

$$\mathcal{V}(\mathsf{sk} = (x_1, x_2), (u_1, u_2)) = u_1^{x_1} u_2^{x_2} = g_1^{r_1 x_1} g_2^{r_2 x_2} = g_1^{r_1 x_1 + \alpha r_2 x_2} \ .$$

Also,

$$\mathsf{pk} = g_1^{x_1} g_2^{x_2} = g_1^{x_1 + \alpha x_2} \ .$$

Now we need to prove that

$$(\mathsf{pk}, \mathcal{V}(\mathsf{sk} = (x_1, x_2), (u_1, u_2))) = \left(g_1^{x_1 + \alpha x_2}, g_1^{r_1 x_1 + \alpha r_2 x_2}\right)$$

is uniform over $G \times G$. To prove this, it is sufficient to argue that $(x_1 + \alpha x_2, r_1 x_1 + \alpha r_2 x_2)$ is uniform over $\mathbb{Z}_q \times \mathbb{Z}_q$. (Recall that here all operations are modulo $q$ wlog) To see this, note that

$$\begin{bmatrix} x_1 + \alpha x_2 \\ r_1 x_1 + \alpha r_2 x_2 \end{bmatrix} = \begin{bmatrix} 1 & \alpha \\ r_1 & r_2 \alpha \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \ ,$$

where remember that $x_1, x_2$ are chosen uniformly. If we can show that the matrix is invertible, then the left-hand side is going to be also invertible, as multiplication with the matrix defines a one-to-one mapping. This can easily be verified by computing the determinant modulo $q$, which is $r_2\alpha - r_1\alpha = \alpha(r_2 - r_1)$. Note that because $\alpha \neq 0$ and $r_1 \neq r_2$, the determinant must be $\neq 0$, which concludes our proof of 1-universality.

### 1.3 Security of HPSs: 2-Universality

In our construction we will need the following strengthening of the above 1-universality property.

**Definition 1.3** (2-universality)**.** We say that $\mathsf{HPS} = (\mathsf{Kg}, \mathcal{P}, \mathcal{V})$ for an NP-language $L$ is 2-*universal* if the distributions of the following pairs are identical for all *distinct* $x, x' \notin L$:

- $(\mathsf{pk}, \widetilde{\pi}, \widetilde{\pi}')$, where $(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{Kg}(1^\lambda)$, $\widetilde{\pi}' \leftarrow \mathcal{V}(\mathsf{sk}, x')$, and $\widetilde{\pi} \leftarrow \mathcal{V}(\mathsf{sk}, x)$
- $(\mathsf{pk}, z, z')$, where $(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{Kg}(1^\lambda)$ and $z, z' \xleftarrow{\$} \{0,1\}^n$.

$\Diamond$

The above definition, requiring *perfect* equality of the distributions, will make the proof below rather simple. In practice, more efficient implementations are obtained by relaxing this requirement to only hold computationally or approximately. We give in the following an example of such an approximate construction, which relies on a computational assumption. We will not prove its security, nor formulate the relaxed security notion it achieves, and only claim everything can be adapted quite easily.

COLLISION-RESISTANT HASH FUNCTIONS. To get our construction, we first need to introduce a security notion for hash functions. In particular, we consider *keyed* hash functions $H : \{0,1\}^\lambda \times \{0,1\}^* \to \{0,1\}^n$, where the first parameter is a public random parameter. We say that such a function is *collision resistant* if for all PPT adversaries $\mathcal{A}$, the following advantage is negligible:

$$\mathsf{Adv}^{\mathsf{cr}}_H(\mathcal{A}) = \mathsf{Pr}\left[s \xleftarrow{\$} \{0,1\}^\lambda,\ (M, M') \xleftarrow{\$} \mathcal{A}(s) :\ M \neq M' \ \wedge\ H_s(M) = H_s(M')\right] .$$

AN (APPROXIMATE) 2-UNIVERSAL HPS. The following construction uses a collision-resistant hash function $H$ to implement the following HPS.

**Procedure** $\mathsf{Kg}(1^\lambda)$:
$s \xleftarrow{\$} \{0,1\}^\lambda$
$x_1, x_2, y_1, y_2 \xleftarrow{\$} \mathbb{Z}_q$
$\mathsf{pk} \leftarrow (s, \mathsf{pk}_1 = g_1^{x_1} g_2^{x_2}, \mathsf{pk}_2 = g_1^{y_1} g_2^{y_2})$
$\mathsf{sk} \leftarrow (s, (x_1, x_2), (y_1, y_2))$
**return** $(\mathsf{pk}, \mathsf{sk})$

**Procedure** $\mathcal{P}(\mathsf{pk}, (u_1, u_2), r)$:
$\alpha \leftarrow H_s(u_1 \parallel u_2)$
$\pi \leftarrow \mathsf{pk}_1^r \mathsf{pk}_2^{\alpha r}$
**return** $\pi$

**Procedure** $\mathcal{V}(\mathsf{sk}, (u_1, u_2))$:
$\alpha \leftarrow H_s(u_1 \parallel u_2)$
$\pi \leftarrow u_1^{x_1 + \alpha y_1} u_2^{x_2 + \alpha y_2}$
**return** $\pi$

ADDING LABELS. A variant of HPS we are going to use below, and that we refer to as *supporting label*, take an additional argument $e \in \{0,1\}^*$ together with $x$ for the prover and the verifier, i.e., we write $\mathcal{P}(\mathsf{pk}, e, x, w)$ and $\mathcal{V}(\mathsf{sk}, e, x)$. We call $e$ a *label*. The correctness and 2-universality requirements remain the same, with addition of the label $e$. In particular:

**Definition 1.4** (2-universality)**.** We say that $\mathsf{HPS} = (\mathsf{Kg}, \mathcal{P}, \mathcal{V})$ supporting labels for an NP-language $L$ is 2-*universal* if the distributions of the following pairs are identical for all $(x, e) \neq (x', e')$, where $x, x' \notin L$ and $e, e' \in \{0,1\}^*$,

- $(\mathsf{pk}, \widetilde{\pi}, \widetilde{\pi}')$, where $(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{Kg}(1^\lambda)$, $\widetilde{\pi}' \leftarrow \mathcal{V}(\mathsf{sk}, e', x')$, and $\widetilde{\pi} \leftarrow \mathcal{V}(\mathsf{sk}, e, x)$
- $(\mathsf{pk}, z, z')$, where $(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{Kg}(1^\lambda)$ and $z, z' \xleftarrow{\$} \{0,1\}^n$.

$\Diamond$

## 2 A CCA-secure PKE from HPSs

HARD LANGUAGES. Assume we are given an NP language $L$ with the following properties:

- It is easy to sample $x \xleftarrow{\$} L$ with a corresponding witness $w$. We denote this as $(x, w) \xleftarrow{\$} L$.
- It is easy to sample $x \xleftarrow{\$} \overline{L}$, where $\overline{L}$ is the complement of $L$.
- The distributions $x \xleftarrow{\$} L$ and $x \xleftarrow{\$} \overline{L}$ are computationally indistinguishable.

It turns out that the above language $L^*$ has all of these properties as long as the DDH assumption holds. In fact, we can show:

**Lemma 2.1.** *If the DDH assumption holds, then we cannot distinguish* $(u_1, u_2) \xleftarrow{\$} L$ *from* $(u_1, u_2) \xleftarrow{\$} \overline{L}$ *if* $g_1, g_2 \xleftarrow{\$} G$.

*Proof.* See homework! □

A PKE CONSTRUCTION. We now give the public-key encryption scheme $\mathsf{PKE} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec})$ which uses two HPSs $\mathsf{HPS}_1 = (\mathsf{Kg}_1, \mathcal{P}_1, \mathcal{V}_1)$ and $\mathsf{HPS}_2 = (\mathsf{Kg}_2, \mathcal{P}_2, \mathcal{V}_2)$ for the language $L$, where $\mathsf{HPS}_2$ additionally supports labels. It is defined as follows:

**Procedure** $\mathsf{Kg}(1^\lambda)$:
$(\mathsf{pk}_1, \mathsf{sk}_1) \xleftarrow{\$} \mathsf{Kg}_1(1^\lambda)$
$(\mathsf{pk}_2, \mathsf{sk}_2) \xleftarrow{\$} \mathsf{Kg}_1(1^\lambda)$
**return** $(\mathsf{pk} = (\mathsf{pk}_1, \mathsf{pk}_2), \mathsf{sk} = (\mathsf{sk}_1, \mathsf{sk}_2))$

**Procedure** $\mathsf{Enc}(\mathsf{pk}, m)$:
$(x, w) \xleftarrow{\$} L$
$\pi_1 \leftarrow \mathcal{P}_1(\mathsf{pk}_1, x, w)$
$c \leftarrow m \oplus \pi_1$
$\pi_2 \xleftarrow{\$} \mathcal{P}_2(\mathsf{pk}_2, c, x, w)$
**return** $(x, c, \pi_2)$

**Procedure** $\mathsf{Dec}(\mathsf{sk}, (x, c, \pi_2))$:
$\widetilde{\pi}_1 \leftarrow \mathcal{V}_1(\mathsf{sk}_1, x)$
$\widetilde{\pi}_2 \leftarrow \mathcal{V}_2(\mathsf{sk}_2, c, x)$
**If** $\widetilde{\pi}_2 = \pi_2$ **then**
  **return** $c \oplus \widetilde{\pi}_1$
**return** $\perp$

We note that when instantiated with $L^*$ and the above two HPSs, this is referred to as the Cramer-Shoup cryptosystem.

**Theorem 2.2.** *If* $\mathsf{HPS}_1$ *is* 1*-universal, and* $\mathsf{HPS}_2$ *is* 2*-universal, then* $\mathsf{PKE}$ *is IND-CCA secure.*

*Proof.* Assume that we are given a PPT adversary $\mathcal{A}$. Recall that without loss of generality, $\mathcal{A}$ only queries the $\mathsf{LR}$ oracle *once*. We define the following game $G_0$ which corresponds to the experiment in the CCA security definition. Then, we are going to transition to games $G_1, G_2$, ... etc which are all negligibly close to each other, until we reach a final game where the adversary has no chance to win the game with probability different than $\frac{1}{2}$. (This is similar to the proofs in the random-oracle model we have seen in Lecture 7.1.)

**The game $G_0$:** The game $G_0$ mimics the IND-CCA security experiment.

**Game $G_0$:**
$(\mathsf{pk}_1, \mathsf{sk}_1) \xleftarrow{\$} \mathsf{Kg}_1(1^\lambda)$
$(\mathsf{pk}_2, \mathsf{sk}_2) \xleftarrow{\$} \mathsf{Kg}_1(1^\lambda)$
$b \xleftarrow{\$} \{0,1\}$
$b' \xleftarrow{\$} \mathcal{A}^{\mathsf{LR}_b, \mathsf{D}}(\mathsf{pk}_1, \mathsf{pk}_2)$
**If** $b = b'$ **then return** $1$
**return** $0$

**Procedure** $\mathsf{LR}_b(m_0, m_1)$:
$(x^*, w^*) \xleftarrow{\$} L$
$\pi_1^* \leftarrow \mathcal{P}_1(\mathsf{pk}_1, x^*, w^*)$
$c^* \leftarrow m_b \oplus \pi_1^*$
$\pi_2^* \xleftarrow{\$} \mathcal{P}_2(\mathsf{pk}_2, c^*, x^*, w^*)$
**return** $(x^*, c^*, \pi_2^*)$

**Procedure** $\mathsf{D}(x, c, \pi_2)$:
**if** $(x, c, \pi_2) = (x^*, c^*, \pi_2^*)$ **then return** $\perp$
$\widetilde{\pi}_1 \leftarrow \mathcal{V}_1(\mathsf{sk}_1, x)$
$\widetilde{\pi}_2 \leftarrow \mathcal{V}_2(\mathsf{sk}_2, c, x)$
**If** $\widetilde{\pi}_2 = \pi_2$ **then**
  **return** $c \oplus \widetilde{\pi}_1$
**return** $\perp$

Clearly, by definition,

$$\mathsf{Adv}^{\mathsf{ind-cca}}_{\mathsf{PKE}}(\mathcal{A}) = 2 \left| \mathsf{Pr}\left[G_0 \Rightarrow 1\right] - \frac{1}{2} \right| .$$

**The game $G_1$:** Here, we modify the way that the $\mathsf{LR}_b$ query is going to be answered, so that instead of using the provers $\mathcal{P}_1$ and $\mathcal{P}_2$, we use the corresponding verifiers. Note that by definition, the generated proofs are identical, and thus there is no difference between the two experiments, i.e., $\mathsf{Pr}\left[G_0 \Rightarrow 1\right] = \mathsf{Pr}\left[G_1 \Rightarrow 1\right]$. More concretely, the $\mathsf{LR}_b$ oracle now behaves as follows:

**Procedure** $\mathsf{LR}_b(m_0, m_1)$:
$(x^*, w^*) \xleftarrow{\$} L$
$\pi_1^* \leftarrow \mathcal{V}_1(\mathsf{sk}_1, x^*)$
$c^* \leftarrow m_b \oplus \pi_1$
$\pi_2^* \xleftarrow{\$} \mathcal{V}_2(\mathsf{sk}_2, c^*, x^*)$
**return** $(x^*, c^*, \pi_2^*)$

**The game $G_2$:** Here, again, we modify $\mathsf{LR}_b$. In particular, now, instead of sampling $x^*$ from $L$, we are going to sample it from the complement $\overline{L}$. Note that the sampling of $x$ is computationally indistinguishable from the original one by our assumption on $L$, and that the whole game $G_1$ or $G_2$ can be simulated given $x$ sampled from either languages, as the witness for $x$ is not needed (when it is in the language), as we are only using the verifiers. Therefore, $\mathsf{Pr}\left[G_1 \Rightarrow 1\right]$ and $\mathsf{Pr}\left[G_2 \Rightarrow 1\right]$ are negligibly close. Concretely, we now have

**Procedure** $\mathsf{LR}_b(m_0, m_1)$:
$x^* \xleftarrow{\$} \overline{L}$
$\pi_1^* \leftarrow \mathcal{V}_1(\mathsf{sk}_1, x^*)$
$c^* \leftarrow m_b \oplus \pi_1$
$\pi_2^* \xleftarrow{\$} \mathcal{V}_2(\mathsf{sk}_2, c^*, x^*)$
**return** $(x^*, c^*, \pi_2^*)$

**The game $G_3$:** Here, we modify the decryption function only to always return an error (i.e., $\perp$) whenever a ciphertext $(x, c, \pi_2)$ with $x \notin L$ is input to it, i.e.,

**Procedure** $\mathsf{D}(x, c, \pi_2)$: // $G_3$
**if** $(x, c, \pi_2) = (x^*, c^*, \pi_2^*)$ **then**
**return** $\perp$
**if** $x \notin L$ **then return** $\perp$
$\widetilde{\pi}_1 \leftarrow \mathcal{V}_1(\mathsf{sk}_1, x)$
$\widetilde{\pi}_2 \leftarrow \mathcal{V}_2(\mathsf{sk}_2, c, x)$
**If** $\widetilde{\pi}_2 = \pi_2$ **then**
**return** $c \oplus \widetilde{\pi}_1$
**return** $\perp$

The rest remains unchanged. Note that $G_2$ and $G_3$ behave identically *until* the attacker can produce a query $(x, c, \pi_2)$ such that $x \notin L$ *and* $\mathcal{V}_2(\mathsf{sk}_2, c, x) = \pi_2$. First off, note that it cannot be that $(x, c) = (x^*, c^*)$, because otherwise $\mathcal{V}_2(\mathsf{sk}_2, c, x) = \pi_2$ implies $\pi_2 = \pi_2^*$, and thus $\mathsf{D}$ can only return $\perp$. Therefore, it must be that $(x, c) \neq (x^*, c^*)$. But then, by 2-universality, for any such $(c, x)$, we have that $\mathcal{V}_2(\mathsf{sk}_2, c, x)$ looks uniform given $\mathcal{V}_2(\mathsf{sk}_2, c^*, x^*)$, for any chosen $x$, and the probability that it equals $\pi_2$ is $2^{-n}$, where $n$ is the proof length, and note that $n$ is polynomial in $\lambda$. (Note that the reduction to computational 2-universality is a bit subtle here, because the latter only gives a guarantee for two queries, but here we have more queries. We omit the fairly standard reduction, and rely on some intuitive understanding.) Therefore, with $q_{\mathsf{D}}$ being the number of decryption queries made by $\mathcal{A}$ (which is polynomial), we have

$$|\mathsf{Pr}[G_3 \Rightarrow 1] - \mathsf{Pr}[G_2 \Rightarrow 1]| \leq \frac{q_{\mathsf{D}}}{2^n} ,$$

which is negligible.

**The game $G_4$:** Here, we modify once again $\mathsf{D}$ to use $\mathcal{P}_1$ instead of $\mathcal{V}_1$. In particular, $\mathsf{D}$ will not be efficient any more and will look for a witness $w$ of $x$ – this will not matter, as we will not need to simulate this experiment in some reduction.

**Procedure** $\mathsf{D}(x, c, \pi_2)$: // $G_3$
**if** $(x, c, \pi_2) = (x^*, c^*, \pi_2^*)$ **then**
**return** $\perp$
**if** $x \notin L$ **then return** $\perp$
Find witness $w$ for $x$
$\widetilde{\pi}_1 \leftarrow \mathcal{P}_1(\mathsf{pk}_1, x, w)$
$\widetilde{\pi}_2 \leftarrow \mathcal{V}_2(\mathsf{sk}_2, c, x)$
**If** $\widetilde{\pi}_2 = \pi_2$ **then**
**return** $c \oplus \widetilde{\pi}_1$
**return** $\perp$

Therefore, $\mathsf{Pr}[G_4 \Rightarrow 1] = \mathsf{Pr}[G_3 \Rightarrow 1]$.

**The game $G_5$:** The final game modifies the $\mathsf{LR}$ oracle as follows:

**Procedure** $\mathsf{LR}_b(m_0, m_1)$:
$x^* \xleftarrow{\$} \overline{L}$
$\pi_1^* \xleftarrow{\$} \{0,1\}^n$
$c^* \leftarrow m_b \oplus \pi_1$
$\pi_2^* \xleftarrow{\$} \mathcal{V}_2(\mathsf{sk}_2, c^*, x^*)$
**return** $(x^*, c^*, \pi_2^*)$

Note that by 1-universality, $\mathsf{Pr}[G_4 \Rightarrow 1] = \mathsf{Pr}[G_5 \Rightarrow 1]$. Moreover, in $G_5$, the attacker obtain no information whatsoever about $m_b$ and $b$ (one-time pad!), and thus we also have $\mathsf{Pr}[G_5 \Rightarrow 1] = \frac{1}{2}$. This concludes the proof. □