Hash Proof Systems and Password Protocols

I – Hash Proof Systems

David Pointcheval

CNRS, Ecole normale supérieure/PSL & INRIA

8th BIU Winter School – Key Exchange
February 2018

Hard Subset Membership

NP Language $\mathcal{L} \subseteq \mathcal{X}$: $(\exists \mathcal{R}$ polynomial relation$)$ $(x \in \mathcal{L} \subseteq \mathcal{X} \iff \exists w,\ \mathcal{R}(x, w) = 1)$

Distinguisher between distributions:

$$\mathbf{Adv}^{\mathcal{L},\mathcal{X}}(\mathcal{D}) = \Pr[\mathcal{D}(x) = 1 \mid x \stackrel{\$}{\leftarrow} \mathcal{L}] - \Pr[\mathcal{D}(x) = 1 \mid x \stackrel{\$}{\leftarrow} \mathcal{X}\backslash\mathcal{L}]$$

Hard Subset Membership for $\mathcal{L} \subseteq \mathcal{X}$: $\forall \mathcal{D}$ polynomial, $\mathbf{Adv}^{\mathcal{L},\mathcal{X}}(\mathcal{D})$ negligible

Example (Decisional Diffie Hellman Problem)

$$\mathbb{G} = \langle g \rangle = \langle h \rangle \qquad \mathcal{X} = \{(G = g^r, H = h^s) \mid r, s \stackrel{\$}{\leftarrow} \mathbb{Z}_q\} = \mathbb{G} \times \mathbb{G}$$

$$\mathcal{L} = \{(G = g^r, H = h^r) \mid r \stackrel{\$}{\leftarrow} \mathbb{Z}_q\} = \langle (g, h) \rangle$$

Proof of Membership

For an NP-Language $\mathcal{L} \subseteq \mathcal{X}$

- defined by a polynomial relation $\mathcal{R}$, such that $x \in \mathcal{L} \iff \exists w,\ \mathcal{R}(x, w) = 1$
- with Hard Subset Membership

A proof system between a prover $\mathcal{P}$ and a verifier $\mathcal{V}$ is

- Correct: for any $x \in \mathcal{L}$, with a witness w such that $\mathcal{R}(x, w) = 1$
  $\mathcal{P}(x, w)$ is accepted by $\mathcal{V}$ with overwhelming probability
- Sound: for any $x \in \mathcal{X}\backslash\mathcal{L}$ (without any witness)
  any $\mathcal{P}^*(x)$ is accepted by $\mathcal{V}$ with negligible probability
- Zero-Knowledge: a simulator $\mathcal{S}$ can generate indistinguishable transcripts to $\mathcal{V}$ for any $x \in \mathcal{L}$, without witness (for any $x \in \mathcal{X}$, under the Hard Subset Membership)
- Simulation-Sound: sound for a new $x \in \mathcal{X}\backslash\mathcal{L}$, after the view of simulated transcripts

Smooth Projective Hash Functions (SPHFs)

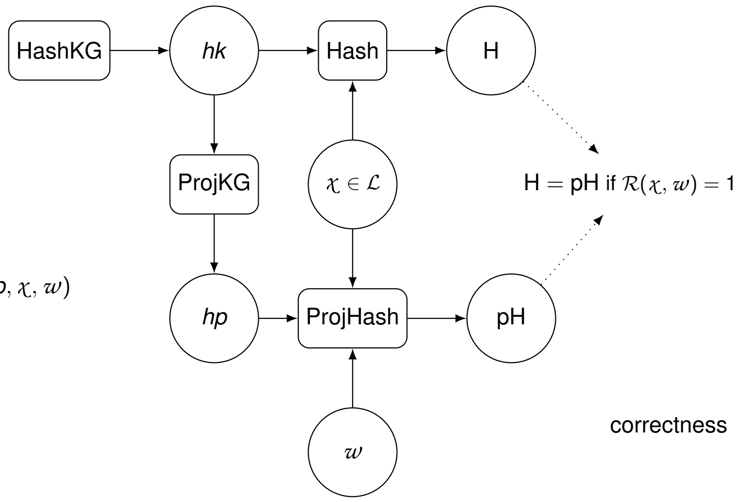
[Cramer-Shoup – Eurocrypt '02]

$hk \xleftarrow{\$} \mathsf{HashKG}()$

$\mathsf{H} \leftarrow \mathsf{Hash}(hk, \chi)$

$hp \leftarrow \mathsf{ProjKG}(hk)$

$\mathsf{pH} \leftarrow \mathsf{ProjHash}(hp, \chi, w)$



SPHF on Diffie-Hellman Pairs

[Cramer-Shoup – Crypto '98]

Let $\mathbb{G} = \langle g \rangle = \langle h \rangle$ of prime order q

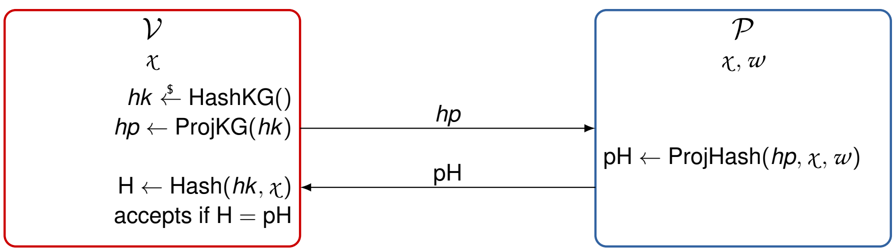
$$\mathcal{X} = \{(G = g^r, H = h^s) \mid r, s \xleftarrow{\$} \mathbb{Z}_q\} = \mathbb{G} \times \mathbb{G}$$
$$\mathcal{L} = \{(G = g^r, H = h^r) \mid r \xleftarrow{\$} \mathbb{Z}_q\} = \langle (g, h) \rangle$$

SPHF for Diffie-Hellman Pairs

$$hk \leftarrow (\alpha, \beta) \xleftarrow{\$} \mathbb{Z}_q^2 \qquad hp \leftarrow g^\alpha h^\beta = \mathsf{ProjKG}(hk)$$
$$\mathsf{H} \leftarrow G^\alpha H^\beta = \mathsf{Hash}(hk, \chi = (G, H)) \qquad \mathsf{pH} \leftarrow hp^r = \mathsf{ProjHash}(hp, \chi, w = r)$$

- Correctness: $\mathsf{H} = (g^r)^\alpha (h^r)^\beta = (g^\alpha)^r (h^\beta)^r = hp^r = \mathsf{pH}$
- Smoothness: $\mathsf{H} = (g^r)^\alpha (h^s)^\beta = (g^\alpha)^r (h^\beta)^s = hp^r (h^{s-r})^\beta$ (no information about β)

Proof of Membership



- Correctness: from the correctness of the SPHF
- Soundness: from the smoothness of the SPHF
- Honest-Verifier Zero-Knowledge

Outline

Outline

Outline

Cramer-Shoup SPHFs

Smooth Projective Hash Functions

$$hk \stackrel{\$}{\leftarrow} \mathsf{HashKG}() \qquad hp \leftarrow \mathsf{ProjKG}(hk)$$
$$\mathsf{H} \leftarrow \mathsf{Hash}(hk, \chi) \qquad \mathsf{pH} \leftarrow \mathsf{ProjHash}(hp, \chi, w)$$

Hash and ProjHash onto the set Π

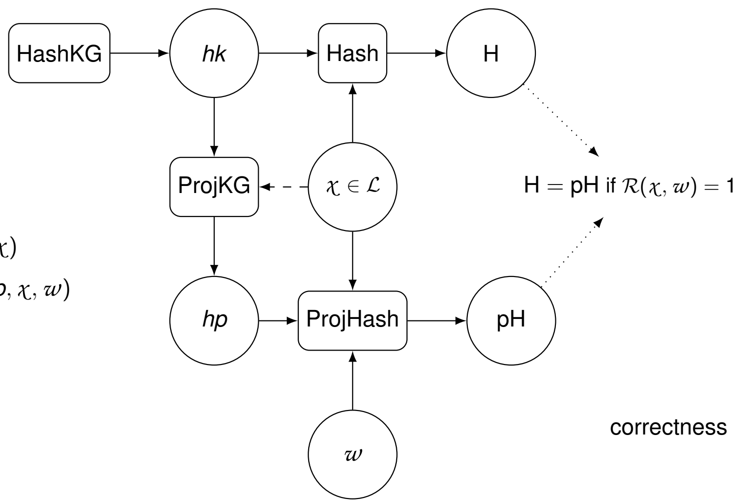
- Correctness: $\forall \chi \in \mathcal{L}, \forall w$ such that $\mathcal{R}(\chi, w) = 1$
  $\forall hk \leftarrow \mathsf{HashKG}(), hp \leftarrow \mathsf{ProjKG}(hk) : \mathsf{Hash}(hk, \chi) = \mathsf{ProjHash}(hp, \chi, w)$
- Smoothness: $\forall \chi \in \mathcal{X} \backslash \mathcal{L}$
  with the probability space $hk \stackrel{\$}{\leftarrow} \mathsf{HashKG}(), hp \leftarrow \mathsf{ProjKG}(hk)$

$$\{(hp, \mathsf{H}) \mid \mathsf{H} \leftarrow \mathsf{Hash}(hk, \chi)\} \approx \{(hp, \mathsf{H}) \mid \mathsf{H} \stackrel{\$}{\leftarrow} \Pi\}$$

Gennaro-Lindell SPHFs

[Gennaro-Lindell – Eurocrypt '03]

$hk \stackrel{\$}{\leftarrow} \mathsf{HashKG}()$
$\mathsf{H} \leftarrow \mathsf{Hash}(hk, \chi)$
$hp \leftarrow \mathsf{ProjKG}(hk, \chi)$
$\mathsf{pH} \leftarrow \mathsf{ProjHash}(hp, \chi, w)$



correctness

Gennaro-Lindell SPHFs

[Gennaro-Lindell – Eurocrypt '03]

Smooth Projective Hash Functions

$$hk \stackrel{\$}{\leftarrow} \mathsf{HashKG}() \qquad hp \leftarrow \mathsf{ProjKG}(hk, \chi)$$
$$\mathsf{H} \leftarrow \mathsf{Hash}(hk, \chi) \qquad \mathsf{pH} \leftarrow \mathsf{ProjHash}(hp, \chi, w)$$

Hash and ProjHash onto the set Π

- Correctness: $\forall \chi \in \mathcal{L}, \forall w$ such that $\mathcal{R}(\chi, w) = 1$
  $\forall hk \leftarrow \mathsf{HashKG}(), hp \leftarrow \mathsf{ProjKG}(hk, \chi) : \mathsf{Hash}(hk, \chi) = \mathsf{ProjHash}(hp, \chi, w)$
- Smoothness: $\forall \chi \in \mathcal{X} \backslash \mathcal{L}$
  with the probability space $hk \stackrel{\$}{\leftarrow} \mathsf{HashKG}(), hp \leftarrow \mathsf{ProjKG}(hk, \chi)$

$$\{(hp, \mathsf{H}) \mid \mathsf{H} \leftarrow \mathsf{Hash}(hk, \chi)\} \approx \{(hp, \mathsf{H}) \mid \mathsf{H} \stackrel{\$}{\leftarrow} \Pi\}$$

Proof of Membership

If the statement χ is known from the beginning by both parties

$\mathcal{V}$		$\mathcal{P}$
χ		χ, w
$hk \stackrel{\$}{\leftarrow} \mathsf{HashKG}()$		
$hp \leftarrow \mathsf{ProjKG}(hk, \chi)$	$\xrightarrow{hp}$	
		$\mathsf{pH} \leftarrow \mathsf{ProjHash}(hp, \chi, w)$
$\mathsf{H} \leftarrow \mathsf{Hash}(hk, \chi)$	$\xleftarrow{\mathsf{pH}}$	
accepts if $\mathsf{H} = \mathsf{pH}$		

GL-SPHFs are enough for the Proof of Membership

Proof of Membership

CS-SPHFs not enough for Adaptive Statements

$\mathcal{V}$		$\mathcal{P}$
		χ, w
$hk \stackrel{\$}{\leftarrow} \mathsf{HashKG}()$		
$hp \leftarrow \mathsf{ProjKG}(hk)$	$\xrightarrow{hp}$	
		$\mathsf{pH} \leftarrow \mathsf{ProjHash}(hp, \chi, w)$
$\mathsf{H} \leftarrow \mathsf{Hash}(hk, \chi)$	$\xleftarrow{\chi, \mathsf{pH}}$	
accepts if $\mathsf{H} = \mathsf{pH}$		

The adversarial prover could choose χ according to *hp*

Adaptive Smoothness

CS-Smoothness

$\forall \chi \in \mathcal{X} \backslash \mathcal{L}$, with the probability space $hk \stackrel{\$}{\leftarrow} \mathsf{HashKG}()$, $hp \leftarrow \mathsf{ProjKG}(hk)$

$$\{(hp, \mathsf{H}) \mid \mathsf{H} \leftarrow \mathsf{Hash}(hk, \chi)\} \approx \{(hp, \mathsf{H}) \mid \mathsf{H} \stackrel{\$}{\leftarrow} \Pi\}$$

- When χ is fixed, *hk* is randomly chosen
- If perfect indistinguishability for every word: no weak word
- If statistical indistinguishability only: weak words exist (can be found)

Let $hk' = (hk, x)$, for $x \stackrel{\$}{\leftarrow} \mathcal{X} \backslash \mathcal{L}$, $hp' = (hp, (x, h = \mathsf{Hash}(hk, x))$,
$\mathsf{Hash}'(hk', \chi) = \mathsf{Hash}(hk, \chi)$ and $\mathsf{ProjHash}'(hp', \chi, w) = \mathsf{ProjHash}(hp, \chi, w)$
The new SPHF can still be CS-Smooth, but the adversarial prover can cheat on x

Katz-Vaikuntanathan SPHFs

[Katz-Vaikuntanathan – TCC '11]

KV-Smoothness

$\forall f$ onto $\mathcal{X}\backslash\mathcal{L}$, with the probability space $hk \stackrel{\$}{\leftarrow} \mathsf{HashKG}()$, $hp \leftarrow \mathsf{ProjKG}(hk)$

$$\{(hp, \mathsf{H}) \mid \mathsf{H} \leftarrow \mathsf{Hash}(hk, f(hp))\} \approx \{(hp, \mathsf{H}) \mid \mathsf{H} \stackrel{\$}{\leftarrow} \Pi\}$$

There is no deterministic way to extract a wrong word from hp

Outline

Matrix Formalism: Correctness

[Benhamouda-Blazy-Chevalier-P.-Vergnaud – Crypto '13]

$$\mathcal{L} = \langle \begin{pmatrix} g \\ h \end{pmatrix} \rangle \subseteq \mathbb{G}^2$$

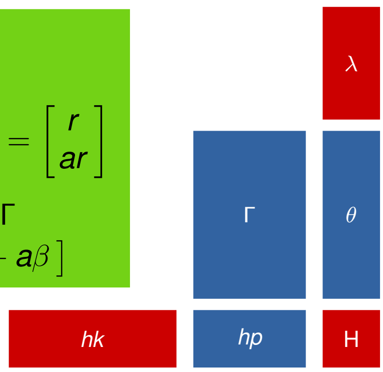
$$\chi = \begin{pmatrix} g^r \\ h^r \end{pmatrix} = \begin{pmatrix} g \\ h \end{pmatrix} \bullet r$$

$$hp = g^\alpha \times h^\beta = (\alpha\ \beta) \bullet \begin{pmatrix} g \\ h \end{pmatrix}$$

$h = g^a$ $\quad \Gamma = \begin{bmatrix} 1 \\ a \end{bmatrix}$

$\lambda = \begin{bmatrix} r \end{bmatrix}$ $\quad \theta = \Gamma \cdot \lambda = \begin{bmatrix} r \\ ar \end{bmatrix}$

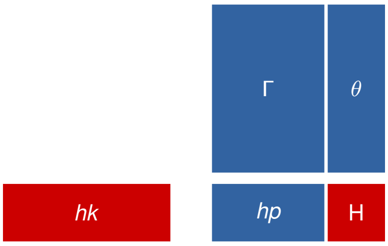
$hk = \begin{bmatrix} \alpha\ \beta \end{bmatrix}$ $\quad hp = hk \cdot \Gamma = \begin{bmatrix} \alpha + a\beta \end{bmatrix}$



$$\mathsf{H} = hk \bullet \chi = (\alpha\ \beta) \bullet \begin{pmatrix} g^r \\ h^r \end{pmatrix} = (\alpha\ \beta) \bullet \begin{pmatrix} g \\ h \end{pmatrix} \bullet r = g^\alpha h^\beta \bullet r = hp \bullet w = \mathsf{pH}$$

$$\mathsf{H} \equiv \begin{bmatrix} \alpha\ \beta \end{bmatrix} \cdot \begin{bmatrix} r \\ ar \end{bmatrix} = r(\alpha + a\beta) = \begin{bmatrix} \alpha + a\beta \end{bmatrix} \cdot \begin{bmatrix} r \end{bmatrix} \equiv \mathsf{pH}$$

Matrix Formalism: Smoothness



- $\theta \in \langle \Gamma \rangle$: H fully determined by hp

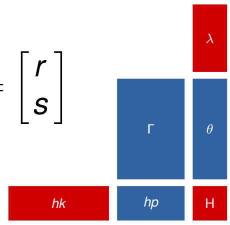
  $\theta = \Gamma \cdot \lambda : \mathrm{H} = hk \cdot \Gamma \cdot \lambda = hp \cdot \lambda = \mathrm{pH}$

- $\theta \notin \langle \Gamma \rangle$: H independent of hp
  - Key hk is randomly chosen
  - $\mathrm{H} = hk \cdot \theta$ while $hp = hk \cdot \Gamma$

Application: DDH and DLin Languages

$$\begin{aligned} \theta &= \Gamma \cdot \lambda \\ hp &= hk \cdot \Gamma \\ \mathrm{H} &= hk \cdot \theta \\ \mathrm{pH} &= hp \cdot \lambda \end{aligned}$$

- DDH: $\{\chi = (g^r, h^r)\}$ with $h = g^a \Longrightarrow \Gamma = \begin{bmatrix} 1 \\ a \end{bmatrix}, \lambda = \begin{bmatrix} r \end{bmatrix}$
  - $hk = \begin{bmatrix} \alpha\ \beta \end{bmatrix} \xleftarrow{\$} \mathbb{Z}_q^2 \Rightarrow hp = \begin{bmatrix} \alpha + a\beta \end{bmatrix} \Rightarrow g^\alpha h^\beta$
  - $(u = g^x, v = g^y) \to \theta = \begin{bmatrix} x \\ y \end{bmatrix} \Rightarrow \mathrm{H} = \begin{bmatrix} \alpha x + \beta y \end{bmatrix} \Rightarrow u^\alpha v^\beta$
  - $(g^r, h^r) \to \theta = \begin{bmatrix} r \\ ar \end{bmatrix} \Rightarrow \mathrm{pH} = \begin{bmatrix} \alpha r + \beta a r \end{bmatrix} \Rightarrow hp^r$
- DLin: $\{\chi = (g^r, h^s, f^{r+s})\}$, with $h = g^a, f = g^b \Longrightarrow \Gamma = \begin{bmatrix} 1 & 0 \\ 0 & a \\ b & b \end{bmatrix}, \lambda = \begin{bmatrix} r \\ s \end{bmatrix}$
  - $hk = \begin{bmatrix} \alpha & \beta & \gamma \end{bmatrix} \xleftarrow{\$} \mathbb{Z}_q^3 \Rightarrow hp = \begin{bmatrix} \alpha + \gamma b\ \ a\beta + \gamma b \end{bmatrix} \Rightarrow (g^\alpha f^\gamma, h^\beta f^\gamma)$
  - $(u = g^x, v = g^y, w = g^z) \to \theta = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \Rightarrow \mathrm{H} = \begin{bmatrix} \alpha x + \beta y + \gamma z \end{bmatrix} \Rightarrow u^\alpha v^\beta w^\gamma$
  - $(g^r, h^s, f^{r+s}) \to \theta = \begin{bmatrix} r \\ ar \\ b(r+s) \end{bmatrix} \Rightarrow \mathrm{pH} = \begin{bmatrix} \alpha r + \beta a r + \gamma b(r+s) \end{bmatrix} \Rightarrow hp_1^r hp_2^s$



Outline

Outline

Public-Key Encryption

[Cramer-Shoup – Crypto '98]

Let $\mathcal{L} \subseteq \mathcal{X}$ be a hard subset membership

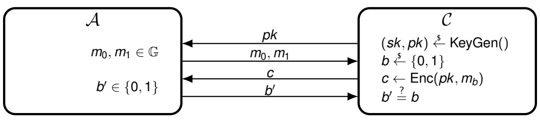
- with an SPHF onto a group $(\mathbb{G}, +)$
- with efficient uniform generation of elements in $\mathcal{L}$ with witnesses

$$
\begin{aligned}
\mathsf{KeyGen}() : &\quad sk \xleftarrow{\$} \mathsf{HashKG}() \text{ and } pk \leftarrow \mathsf{ProjKG}(sk) \\
\mathsf{Enc}(pk, m \in \mathbb{G}) : &\quad \chi \xleftarrow{\$} \mathcal{L} \text{ with witness } \mathit{w} \\
&\quad c \leftarrow (\chi, e = \mathsf{ProjHash}(pk, \chi, \mathit{w}) + m) \\
\mathsf{Dec}(sk, c \in \mathcal{X} \times \mathbb{G}) : &\quad m \leftarrow e - \mathsf{Hash}(sk, \chi)
\end{aligned}
$$

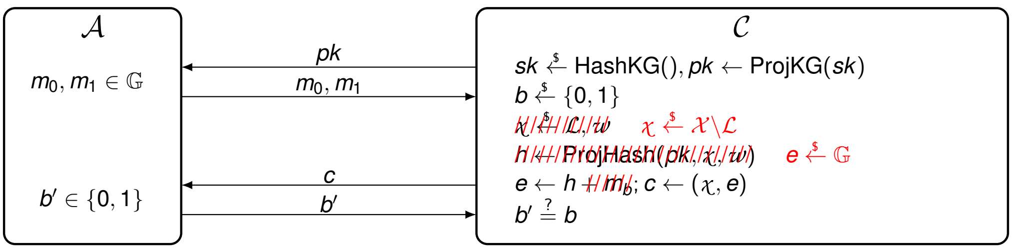
IND-CPA Encryption

- Correctness: since $\mathsf{ProjHash}(pk, \chi, \mathit{w}) = \mathsf{Hash}(sk, \chi)$
- IND-CPA security: from Hash Subset Membership and Smoothness

IND-CPA Security



$$
\begin{aligned}
\mathsf{KeyGen}() : &\quad sk \xleftarrow{\$} \mathsf{HashKG}(), pk \leftarrow \mathsf{ProjKG}(sk) \\
\mathsf{Enc}(pk, m) : &\quad \chi \xleftarrow{\$} \mathcal{L} \text{ with witness } \mathit{w} \\
&\quad c \leftarrow (\chi, e = \mathsf{ProjHash}(pk, \chi, \mathit{w}) + m) \\
\mathsf{Dec}(sk, c) : &\quad m \leftarrow e - \mathsf{Hash}(sk, \chi)
\end{aligned}
$$



Correctness + Hard Subset Membership + Smoothness $\Longrightarrow \Pr[b' = b] = \frac{1}{2}$

DH-based Public-Key Encryption

Let $\mathbb{G} = \langle g \rangle = \langle h \rangle$ of prime order q

$$\mathcal{X} = \{(G = g^r, H = h^s) \,|\, r, s \xleftarrow{\$} \mathbb{Z}_q\} = \mathbb{G} \times \mathbb{G}$$
$$\mathcal{L} = \{(G = g^r, H = h^r) \,|\, r \xleftarrow{\$} \mathbb{Z}_q\} = \langle (g, h) \rangle$$

$$\begin{aligned} \mathsf{KeyGen}() : &\quad sk = (\alpha, \beta) \xleftarrow{\$} \mathbb{Z}_q^2 \quad pk \leftarrow g^\alpha h^\beta \\ \mathsf{Enc}(pk, m) : &\quad r \xleftarrow{\$} \mathbb{Z}_q \qquad \chi \leftarrow (u_1 = g^r, u_2 = h^r) \quad e \leftarrow pk^r \times m \\ \mathsf{Dec}(sk, c = (u_1, u_2, e)) : &\quad m \leftarrow e/(u_1^\alpha u_2^\beta) \end{aligned}$$

IND-CCA Security

$$\begin{aligned} \mathsf{KeyGen}() : &\quad sk \xleftarrow{\$} \mathsf{HashKG}(), pk \leftarrow \mathsf{ProjKG}(sk) \\ \mathsf{Enc}(pk, m) : &\quad \chi \xleftarrow{\$} \mathcal{L} \text{ with witness } w, c \leftarrow (\chi, e = \mathsf{ProjHash}(pk, \chi, w) + m) \\ \mathsf{Dec}(sk, c) : &\quad m \leftarrow e - \mathsf{Hash}(sk, \chi) \end{aligned}$$

The decryption procedure does not leak any information about sk if $\chi \in \mathcal{L}$ but it might leak when $\chi \in \mathcal{X} \backslash \mathcal{L}$: what about adding a second SPHF?

$$\begin{aligned} \mathsf{KeyGen}() : &\quad hk \xleftarrow{\$} \mathsf{HashKG}(), hp \leftarrow \mathsf{ProjKG}(hk) \\ &\quad hk' \xleftarrow{\$} \mathsf{HashKG}(), hp' \leftarrow \mathsf{ProjKG}(hk') \\ &\quad sk \leftarrow (hk, hk'), pk \leftarrow (hp, hp') \\ \mathsf{Enc}(pk, m) : &\quad \chi \xleftarrow{\$} \mathcal{L} \text{ with witness } w \\ &\quad c \leftarrow (\chi, e = \mathsf{ProjHash}(hp, \chi, w) + m, v = \mathsf{ProjHash}(hp', \chi, w)) \\ \mathsf{Dec}(sk, c) : &\quad v \stackrel{?}{=} \mathsf{Hash}(hk', \chi), m \leftarrow e - \mathsf{Hash}(hk, \chi) \end{aligned}$$

IND-CCA Security: First Attempt

$\mathcal{A}$		$\mathcal{C}$
		$hk, hk' \xleftarrow{\$} \mathsf{HashKG}(), hp \leftarrow \mathsf{ProjKG}(hk), hp' \leftarrow \mathsf{ProjKG}(hk')$
	$\xleftarrow{pk = (hp, hp')}$	
$m_0, m_1 \in \mathbb{G}$	$\xrightarrow{m_0, m_1}$	$b \xleftarrow{\$} \{0, 1\}$
		$\cancel{\chi \xleftarrow{\$} \mathcal{L}, w}$ $\quad \chi \xleftarrow{\$} \mathcal{X} \backslash \mathcal{L}$
		$\cancel{h \leftarrow \mathsf{ProjHash}(hp, \chi, w)}$ $\quad h \leftarrow \mathsf{Hash}(hk, \chi)$
		$\cancel{v \leftarrow \mathsf{ProjHash}(hp', \chi, w)}$ $\quad v \leftarrow \mathsf{Hash}(hk', \chi)$
	$\xleftarrow{c}$	$e = h + m_b; c \leftarrow (\chi, e, v)$
$b' \in \{0, 1\}$	$\xrightarrow{b'}$	$b' \stackrel{?}{=} b$

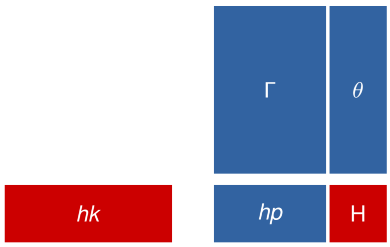
$$\begin{aligned} \mathsf{Dec}(sk, (\chi', e', v')) : &\quad v'' \leftarrow \mathsf{Hash}(hk', \chi) \qquad (\chi', e', v') \neq (\chi, e, v) \Longrightarrow (\chi', e') \neq (\chi, e) \\ &\quad v'' \neq v' : \text{ reject} \qquad \Longrightarrow \chi' \in \mathcal{L} : \text{Simulation-Soundness (?)} \\ &\quad m' \leftarrow e' - \mathsf{Hash}(hk, \chi') \end{aligned}$$

Correctness + Correctness + Hard Subset Membership

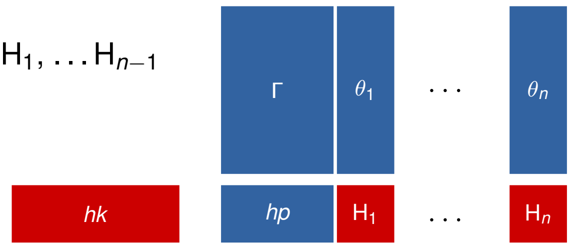
Outline

Soundness: Smoothness

- Soundness: if $\chi \notin \mathcal{L}$
  H must be independent from *hp*

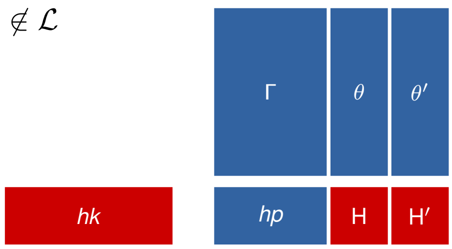


- Simulation-Soundness: if $\chi_n \notin \mathcal{L}$
  H_n must be independent from *hp*, $\mathsf{H}_1, \ldots \mathsf{H}_{n-1}$
  even if $\chi_1, \ldots, \chi_{n-1} \notin \mathcal{L}$

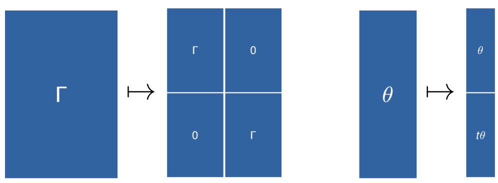


One-Time Simulation-Soundness: 2-Smoothness

- One-Time Simulation-Soundness: if $\chi' \notin \mathcal{L}$
  H′ must be independent from *hp*, H
  even if $\chi \notin \mathcal{L}$

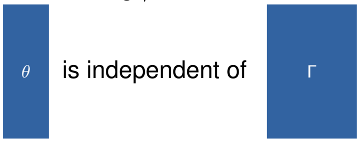


- Tag-Based SPHF: for a word χ and a tag t

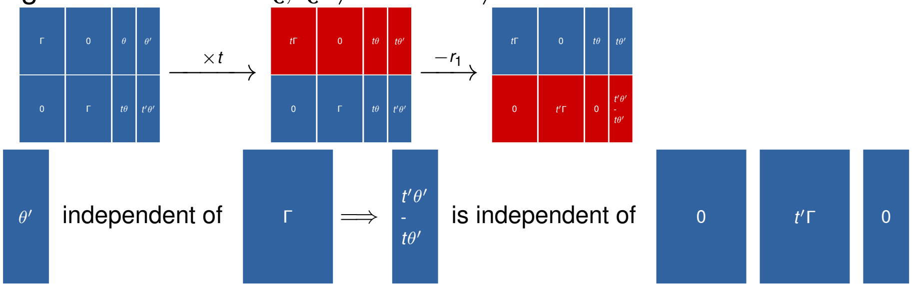


One-Time Simulation-Soundness

- SPHF: if $\chi \notin \mathcal{L}$



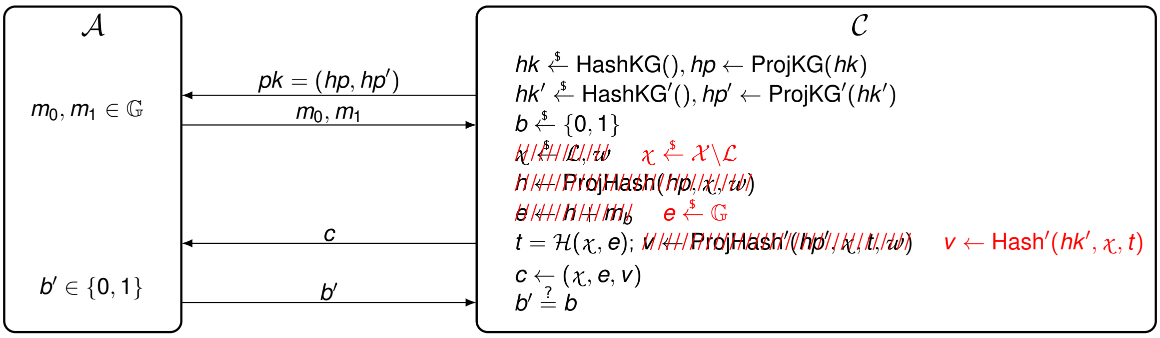
- Tag-Based SPHF: if $\chi, \chi' \notin \mathcal{L}$ and $t' \neq t$



2-Smooth Projective Hash Function

- SPHF:
  - $hk \stackrel{\$}{\leftarrow} \mathsf{HashKG}()$
  - $hp \leftarrow \mathsf{ProjKG}(hk)$
  - $\mathsf{H} \leftarrow \mathsf{Hash}(hk, \chi)$
  - $\mathsf{pH} \leftarrow \mathsf{ProjHash}(hp, \chi, w)$
- 2-SPHF:
  - $hk' = \mathsf{HashKG}'() \leftarrow (hk_1, hk_2)$, $hk_1, hk_2 \stackrel{\$}{\leftarrow} \mathsf{HashKG}()$
  - $hp' = \mathsf{ProjKG}'(hk') \leftarrow (hp_1, hp_2)$, $hp_1 \leftarrow \mathsf{ProjKG}(hk_1)$, $hp_2 \leftarrow \mathsf{ProjKG}(hk_2)$
  - $\mathsf{H}' = \mathsf{Hash}'(hk', \chi, t) \leftarrow \mathsf{Hash}(hk_1, \chi) + t \times \mathsf{Hash}(hk_2, \chi)$
  - $\mathsf{pH}' = \mathsf{ProjHash}'(hp', \chi, t, w) \leftarrow \mathsf{ProjHash}(hp_1, \chi, w) + t \times \mathsf{ProjHash}(hp_2, \chi, w)$

IND-CCA Security: Second Attempt



$\mathsf{Dec}(sk, (\chi', e', v'))$: $v'' \leftarrow \mathsf{Hash}'(hk', \chi', t')$ $(\chi', e', v') \neq (\chi, e, v) \Longrightarrow (\chi', e') \neq (\chi, e)$

$v'' \neq v'$: reject $\Longrightarrow \chi' \in \mathcal{L}$: OT Simulation-Soundness

$m' \leftarrow e' - \mathsf{Hash}(hk, \chi')$

Correctness + Hard Subset Membership + Smoothness $\Longrightarrow \Pr[b' = b] = \frac{1}{2}$

DH-based Public-Key Encryption

[Cramer-Shoup – Crypto '98]

$$\mathcal{X} = \{(G = g_1^r, H = g_2^s) \mid r, s \xleftarrow{\$} \mathbb{Z}_q\} \qquad \mathcal{L} = \{(G = g_1^r, H = g_2^r) \mid r \xleftarrow{\$} \mathbb{Z}_q\}$$

$$\begin{aligned}
\mathsf{KeyGen}(): \quad & sk = (hk = (\alpha, \beta), hk'_1 = (x_1, x_2), hk'_2 = (y_1, y_2)) \xleftarrow{\$} \mathbb{Z}_q^6 \\
& pk = (hp \leftarrow g_1^\alpha g_2^\beta, hp'_1 \leftarrow g_1^{x_1} g_2^{x_2}, hp'_2 \leftarrow g_1^{y_1} g_2^{y_2}) \\
\mathsf{Enc}(pk, m): \quad & r \xleftarrow{\$} \mathbb{Z}_q; u_1 = g_1^r, u_2 = g_2^r; e \leftarrow hp^r \times m \\
& v = (hp'_1 \times {hp'_2}^t)^r, \text{ with } t \leftarrow \mathcal{H}(u_1, u_2, e) \\
\mathsf{Dec}(sk, (u_1, u_2, e, v)): \quad & v \stackrel{?}{=} u_1^{x_1} u_2^{y_1} \times (u_2^{x_2} u_2^{y_2})^t, \text{ with } t \leftarrow \mathcal{H}(u_1, u_2, e) : m = e / u_1^\alpha u_2^\beta
\end{aligned}$$

Cramer-Shoup CCA Encryption Scheme

$$\begin{aligned}
\mathsf{KeyGen}(): \quad & sk = (z, x_1, x_2, y_1, y_2) \xleftarrow{\$} \mathbb{Z}_q^5 \\
& pk = (h \leftarrow g_1^z, c \leftarrow g_1^{x_1} g_2^{x_2}, d \leftarrow g_1^{y_1} g_2^{y_2}) \\
\mathsf{Enc}(pk, m): \quad & r \xleftarrow{\$} \mathbb{Z}_q; u_1 = g_1^r, u_2 = g_2^r; e \leftarrow h^r \times m \\
& v = (c \times d^t)^r, \text{ with } t \leftarrow \mathcal{H}(u_1, u_2, e) \\
\mathsf{Dec}(sk, (u_1, u_2, e, v)): \quad & v \stackrel{?}{=} u_1^{x_1 + t x_2} u_2^{y_1 + t y_2}, \text{ with } t \leftarrow \mathcal{H}(u_1, u_2, e) : m = e / u_1^z
\end{aligned}$$

Outline

Outline

DH-based Languages

- DH-tuples for (g,h): $\mathcal{L} = \{(g^r, h^r)\} \subseteq \{(g^x, g^y)\} = \mathcal{X}$ with $h = g^a$

$$\Gamma = \begin{bmatrix} 1 \\ a \end{bmatrix} \quad \lambda = \begin{bmatrix} r \end{bmatrix} \quad \theta = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} r \\ ar \end{bmatrix}$$

- ElGamal ciphertext of m: $c = (u = g^r, e = h^r m) \Longrightarrow (u, e/m) \in \mathcal{L}$
- Valid Cramer-Shoup ciphertext:

$$c = (u_1 = g_1^r, u_2 = g_2^r, e = h^r m, v = (cd^t)^r) \text{ with } t = \mathcal{H}(u_1, u_2, e)$$

If $g_2 = g_1^s$, $h = g_1^a$, $c = g_1^\alpha$, $d = g_1^\beta$ and $c = (u_1 = g_1^{r_1}, u_2 = g_1^{r_2}, e = g_1^y, v = g_1^z)$
c is a valid CS ciphertext iff (u_1, u_2, v) is an r-th power of (g_1, g_2, cd^t)

$$\Gamma = \begin{bmatrix} 1 \\ s \\ \alpha + t\beta \end{bmatrix} \quad \lambda = \begin{bmatrix} r \end{bmatrix} \quad \theta = \begin{bmatrix} r_1 \\ r_2 \\ z \end{bmatrix} = \begin{bmatrix} r \\ sr \\ (\alpha + t\beta)r \end{bmatrix} \text{ for } t = \mathcal{H}(u_1, u_2, e)$$

Cramer-Shoup Ciphertext Languages

$$c = (u_1 = g_1^r, u_2 = g_2^r, e = h^r m, v = (cd^t)^r) \text{ with } t = \mathcal{H}(u_1, u_2, e)$$

If $g_2 = g_1^s$, $h = g_1^a$, $c = g_1^\alpha$, $d = g_1^\beta$ and $c = (u_1 = g_1^{r_1}, u_2 = g_1^{r_2}, e = g_1^y m, v = g_1^z)$

- c is a valid CS ciphertext iff (u_1, u_2, v) is an r-th power of (g_1, g_2, cd^t)

$$\Gamma = \begin{bmatrix} 1 \\ s \\ \boxed{\alpha + t\beta} \end{bmatrix} \quad \lambda = \begin{bmatrix} r \end{bmatrix} \quad \theta = \begin{bmatrix} r_1 \\ r_2 \\ z \end{bmatrix} = \begin{bmatrix} r \\ sr \\ (\alpha + t\beta)r \end{bmatrix} \text{ for } t = \mathcal{H}(u_1, u_2, e)$$

- c is a CS ciphertext of m iff $(u_1, u_2, e/m, v)$ is an r-th power of (g_1, g_2, h, cd^t)

$$\Gamma = \begin{bmatrix} 1 \\ s \\ a \\ \boxed{\alpha + t\beta} \end{bmatrix} \quad \lambda = \begin{bmatrix} r \end{bmatrix} \quad \theta = \begin{bmatrix} r_1 \\ r_2 \\ y \\ z \end{bmatrix} = \begin{bmatrix} r \\ sr \\ ar \\ (\alpha + t\beta)r \end{bmatrix} \text{ for } t = \mathcal{H}(u_1, u_2, e)$$

Adaptive Statement

c is a valid CS ciphertext $\quad \Gamma = \begin{bmatrix} 1 \\ s \\ \alpha + t\beta \end{bmatrix}$

$\Gamma = \begin{bmatrix} 1 \\ s \\ a \\ \alpha + t\beta \end{bmatrix}$ $\quad c$ is a CS ciphertext of m

$$\begin{array}{rcl} \theta & = & \Gamma \cdot \lambda \\ hp & = & hk \cdot \Gamma \\ \mathrm{H} & = & hk \cdot \theta \\ \mathrm{pH} & = & hp \cdot \lambda \end{array}$$

Γ depends on $t = \mathcal{H}(u_1, u_2, e)$
$\Longrightarrow \Gamma$ depends on c
$\Longrightarrow hp$ depends on c

These are GL-SPHFs only!

KV-SPHFs for Cramer-Shoup Ciphertext Languages

[Benhamouda-Blazy-Chevalier-P.-Vergnaud – Crypto '13]

$$c = (u_1 = g_1^r, u_2 = g_2^r, e = h^r m, v = (cd^t)^r) \text{ with } t = \mathcal{H}(u_1, u_2, e)$$

$$\text{If } g_2 = g_1^s,\ h = g_1^a,\ c = g_1^\alpha,\ d = g_1^\beta \text{ and } c = (u_1 = g_1^{r_1}, u_2 = g_1^{r_2}, e = g_1^y m, v = g_1^z)$$

- Valid CS ciphertext $\Longrightarrow \chi = (u_1, u_1^t, u_2, v)$ for $t = \mathcal{H}(u_1, u_2, e)$

$$\Gamma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ s & 0 \\ \alpha & \beta \end{bmatrix} \quad \lambda = \begin{bmatrix} r \\ tr \end{bmatrix} \quad \theta = \begin{bmatrix} r_1 \\ tr_1 \\ r_2 \\ z \end{bmatrix} = \begin{bmatrix} r \\ tr \\ sr \\ \alpha r + \beta tr \end{bmatrix}$$

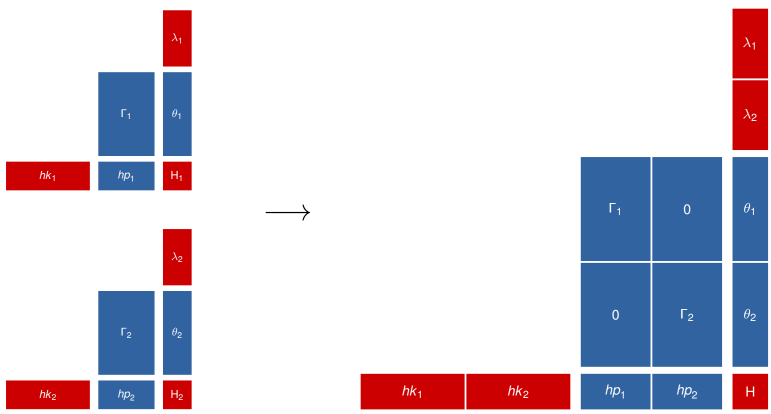
- CS ciphertext of $m \Longrightarrow \chi = (u_1, u_1^t, u_2, e/m, v)$ for $t = \mathcal{H}(u_1, u_2, e)$

$$\Gamma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ s & 0 \\ a & 0 \\ \alpha & \beta \end{bmatrix} \quad \lambda = \begin{bmatrix} r \\ tr \end{bmatrix} \quad \theta = \begin{bmatrix} r_1 \\ tr_1 \\ r_2 \\ y \\ z \end{bmatrix} = \begin{bmatrix} r \\ tr \\ sr \\ ar \\ \alpha r + \beta tr \end{bmatrix}$$

Outline

Conjunctions of Languages

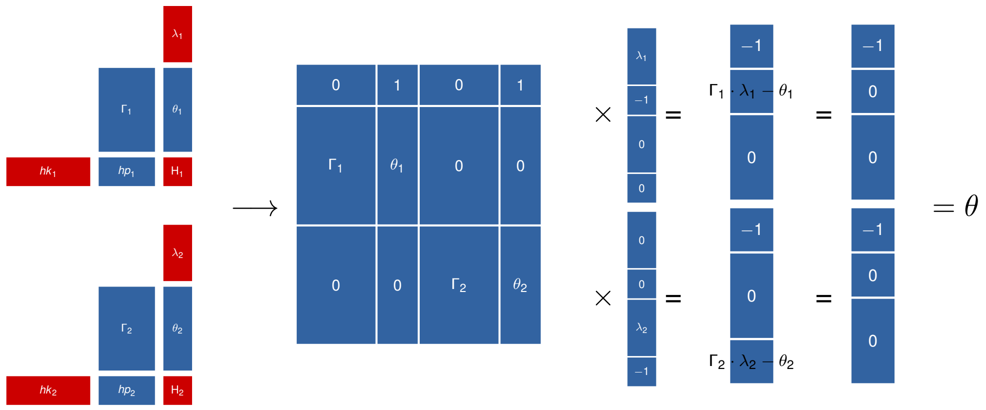
$\mathcal{L}_1 \subseteq \mathcal{X}_1$ and $\mathcal{L}_2 \subseteq \mathcal{X}_2$ $\qquad$ $\mathcal{L}_1 \times \mathcal{L}_2 \subseteq \mathcal{X}_1 \times \mathcal{X}_2$



Disjunctions of Languages

[Abdalla-Benhamouda-P. – Eurocrypt '15]

$\mathcal{L}_1 \subseteq \mathcal{X}_1$ and $\mathcal{L}_2 \subseteq \mathcal{X}_2$ $\qquad (\mathcal{L}_1 \times \mathcal{X}_2) \cup (\mathcal{X}_1 \times \mathcal{L}_2) \subseteq \mathcal{X}_1 \times \mathcal{X}_2$



Disjunctions of DH Languages

$\mathcal{L}_1 = \{(g^r, h_1^r)\}$ and $\mathcal{L}_2 = \{(g^r, h_2^r)\}$, for $h_1 = g^{a_1}$ and $h_2 = g^{a_2}$: $c = (u = g^x, v = g^y)$

$$\Gamma = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & x & 0 & 0 \\ a_1 & y & 0 & 0 \\ 0 & 0 & 1 & x \\ 0 & 0 & a_2 & y \end{bmatrix} = \begin{pmatrix} 1 & g & 1 & g \\ g & u & 1 & 1 \\ h_1 & v & 1 & 1 \\ 1 & 1 & g & u \\ 1 & 1 & h_2 & v \end{pmatrix} \qquad \begin{bmatrix} -1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \theta$$

$$hk = \begin{bmatrix} \alpha & \beta & \gamma & \delta & \epsilon \end{bmatrix} \qquad \begin{pmatrix} g^\beta h_1^\gamma & g^\alpha u^\beta v^\gamma & g^\delta h_2^\epsilon & g^\alpha u^\delta v^\epsilon \end{pmatrix} = hp$$

$c = (u, v) : \mathrm{H} = hk \cdot \theta = \begin{bmatrix} -\alpha \end{bmatrix} \Longrightarrow g^{-\alpha}$

If $c = (g^r, h_1^r)$:

$$\lambda = \begin{bmatrix} r \\ -1 \\ 0 \\ 0 \end{bmatrix}, \mathrm{pH} = hp \cdot \lambda \Longrightarrow g^{-\alpha}(g^r/u)^\beta (h_1^r/v)^\gamma$$

If $c = (g^r, h_2^r)$:

$$\lambda = \begin{bmatrix} 0 \\ 0 \\ r \\ -1 \end{bmatrix}, \mathrm{pH} = hp \cdot \lambda \Longrightarrow g^{-\alpha}(g^r/u)^\delta (h_1^r/v)^\epsilon$$

Outline

Limits of Previous Constructions

$\mathcal{L}_1 \subseteq \mathcal{X}_1$ and $\mathcal{L}_2 \subseteq \mathcal{X}_2$: $\mathcal{L} = (\mathcal{L}_1 \times \mathcal{X}_2) \cup (\mathcal{X}_1 \times \mathcal{L}_2)$

$$\Gamma = \begin{array}{|c|c|c|c|} \hline 0 & 1 & 0 & 1 \\ \hline \Gamma_1 & \theta_1 & 0 & 0 \\ \hline 0 & 0 & \Gamma_2 & \theta_2 \\ \hline \end{array} \implies \Gamma \text{ depends on } \theta_1, \theta_2$$

This is a GL-SPHF!

$\mathcal{L} \neq (\mathcal{L}_1 \times \mathcal{X}_2) + (\mathcal{X}_1 \times \mathcal{L}_2) = \mathcal{X}_1 \times \mathcal{X}_2$ where the sets are identified to vectorial spaces
But $\mathcal{L} = (\mathcal{L}_1 \otimes \mathcal{X}_2) + (\mathcal{X}_1 \otimes \mathcal{L}_2)$

KV Disjunctions

[Abdalla-Benhamouda-P. – Eurocrypt '15]

$$\begin{array}{rcl} \mathcal{L} & = & (\mathcal{L}_1 \otimes \mathcal{X}_2) + (\mathcal{X}_1 \otimes \mathcal{L}_2) \\ \Gamma & = & \begin{array}{|c|c|} \hline \Gamma_1 \otimes \mathrm{Id}_{n_2} & \mathrm{Id}_{n_1} \otimes \Gamma_2 \\ \hline \end{array} \\ \theta & = & \theta_1 \otimes \theta_2 \\ \lambda & = & \begin{array}{|c|} \hline \lambda_1 \otimes \theta_2 \\ \hline 0 \\ \hline \end{array} \text{ if } \chi \in \mathcal{L}_1 \qquad \begin{array}{|c|} \hline 0 \\ \hline \theta_1 \otimes \lambda_2 \\ \hline \end{array} \text{ if } \chi \in \mathcal{L}_2 \end{array}$$

Disjunctions of DH Languages

$\mathcal{L}_1 = \{(g^r, h_1^r)\}$ and $\mathcal{L}_2 = \{(g^r, h_2^r)\}$, for $h_1 = g^{a_1}$ and $h_2 = g^{a_2}$: $c = (u = g^x, v = g^y)$

$$\Gamma_1 = \begin{bmatrix} 1 \\ a_1 \end{bmatrix} \quad \lambda_1 = \begin{bmatrix} r \end{bmatrix} \quad \theta_1 = \begin{bmatrix} x \\ y \end{bmatrix} \qquad \Gamma_2 = \begin{bmatrix} 1 \\ a_2 \end{bmatrix} \quad \lambda_2 = \begin{bmatrix} r \end{bmatrix} \quad \theta_2 = \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\Gamma = \begin{bmatrix} 1 \\ a_1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \Bigg\| \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \otimes \begin{bmatrix} 1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & a_2 & 0 \\ a_1 & 0 & 0 & 1 \\ 0 & a_1 & 0 & a_2 \end{bmatrix} = \begin{pmatrix} g & 1 & g & 1 \\ 1 & g & h_2 & 1 \\ h_1 & 1 & 1 & g \\ 1 & h_1 & 1 & h_2 \end{pmatrix}$$

$$\theta = \begin{bmatrix} x \bullet x \\ x \bullet y \\ y \bullet x \\ y \bullet y \end{bmatrix} = \begin{pmatrix} e(u,u) \\ e(u,v) \\ e(v,u) \\ e(v,v) \end{pmatrix} \qquad \begin{array}{rcl} hk & = & \begin{bmatrix} \alpha & \beta & \gamma & \delta \end{bmatrix} \\ hp & = & \begin{pmatrix} g^\alpha h_1^\gamma & g^\beta h_1^\delta & g^\alpha h_2^\beta & g^\gamma h_2^\delta \end{pmatrix} \end{array}$$

Disjunctions of DH Languages

$\mathcal{L}_1 = \{(g^r, h_1^r)\}$ and $\mathcal{L}_2 = \{(g^r, h_2^r)\}$, for $h_1 = g^{a_1}$ and $h_2 = g^{a_2}$: $c = (u = g^x, v = g^y)$

$$\Gamma_1 = \begin{bmatrix} 1 \\ a_1 \end{bmatrix} \quad \lambda_1 = \begin{bmatrix} r \end{bmatrix} \quad \theta_1 = \begin{bmatrix} x \\ y \end{bmatrix} \qquad \Gamma_2 = \begin{bmatrix} 1 \\ a_2 \end{bmatrix} \quad \lambda_2 = \begin{bmatrix} r \end{bmatrix} \quad \theta_2 = \begin{bmatrix} x \\ y \end{bmatrix}$$

$$hk = \begin{bmatrix} \alpha & \beta & \gamma & \delta \end{bmatrix} \qquad hp = \begin{pmatrix} g^\alpha h_1^\gamma & g^\beta h_1^\delta & g^\alpha h_2^\beta & g^\gamma h_2^\delta \end{pmatrix}$$

$$\mathrm{H} = hk \cdot \theta = \begin{bmatrix} \alpha \bullet x \bullet x + (\beta + \gamma) \bullet x \bullet y + \delta \bullet y \bullet y \end{bmatrix} \Longrightarrow e(u,u)^\alpha e(u,v)^{\beta+\gamma} e(v,v)^\delta$$

For $c = (g^r, h_1^r)$

$$\lambda = \begin{bmatrix} \lambda_1 \otimes \theta_2 \\ 0 \end{bmatrix} = \begin{bmatrix} r \bullet x \\ r \bullet y \\ 0 \\ 0 \end{bmatrix}, \mathrm{pH} = hp \cdot \lambda \Rightarrow \begin{cases} e(g^x, g^r)^\alpha e(g^x, h_1^r)^\gamma \times e(g^y, g^r)^\beta e(g^y, h_1^r)^\delta \\ e(u,u)^\alpha e(u,v)^\gamma \times e(v,u)^\beta e(v,v)^\delta \\ e(u,u)^\alpha \cdot e(u,v)^{\beta+\gamma} \cdot e(v,v)^\delta \end{cases}$$

Outline

Conclusion

SPHFs = Smooth Projective Hash Functions allow

- Honest-Verifier Zero-Knowledge Arguments
  With disjunctions $\Longrightarrow$ Zero-Knowledge Arguments
- And even NIZKs with Simulation-Soundness
- Trapdoor SPHFs, for ZK Arguments [Benhamouda-Blazy-Chevalier-P.-Vergnaud – Crypto '13]
- Implicit ZK, for malicious 2-Party Computations [Benhamouda-Couteau-P.-Wee – Crypto '15]
- Explainable SPHFs, to remove erasures [Abdalla-Benhamouda-P. – PKC '17]

See Fabrice Benhamouda's Thesis: "Diverse modules and zero-knowledge"
for all technical details

Application to Password-Authenticated Key Exchange