Regular Lossy Functions and Applications in Leakage-Resilient Cryptography

Yu Chen[1] Baodong Qin[2] Haiyang Xue[1]

[1]SKLOIS, IIE, Chinese Academy of Sciences
[2]Xi'an University of Posts and Telecommunication

CT-RSA 2018

April 20th, 2018

Outline

Outline

Lossy Trapdoor Functions

Lossy object *indistinguishable* from original

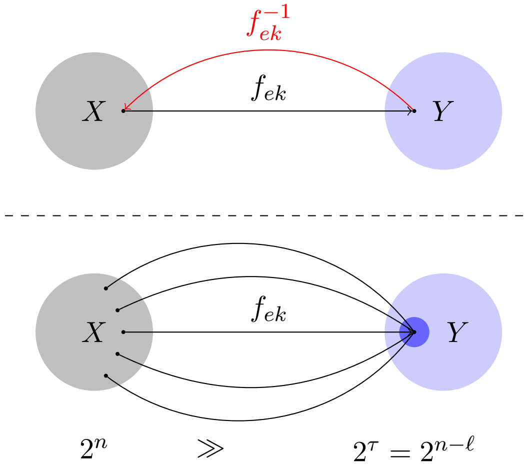
STOC 2008 Peikert and Waters: Lossy Trapdoor Functions and Their Applications

Lossy TDFs

injective
$\mathsf{Gen}(\lambda) \to (ek, td)$

$\approx_c$

$\mathsf{Gen}(\lambda) \to (ek, \perp)$
lossy



Extension of LTFs: ABO LTFs

- $\mathsf{Gen}(\lambda, b^*)$ has extra input: branch $b^* \in B$.

$$\mathsf{Gen}(\lambda, b^*) \to (ek, td)$$

$$f_{ek,b_1}(\cdot) \quad f_{ek,b_2}(\cdot) \quad \cdots \quad f_{ek,b^*}(\cdot) \quad f_{ek,b_i}(\cdot)$$

b^* is hidden from ek

$$f_{ek,b}(\cdot) = \begin{cases} \text{lossy} & b = b^* \\ \text{injective and invertible} & b \neq b^* \end{cases}$$

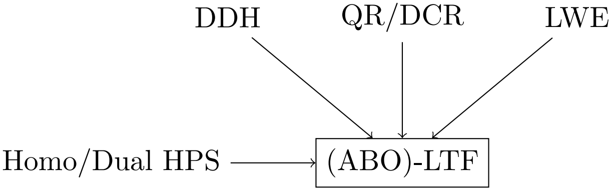
$$\text{LTFs} \Leftrightarrow \text{ABO LTFs}$$

$\boxed{\text{(ABO)-LTF}}$

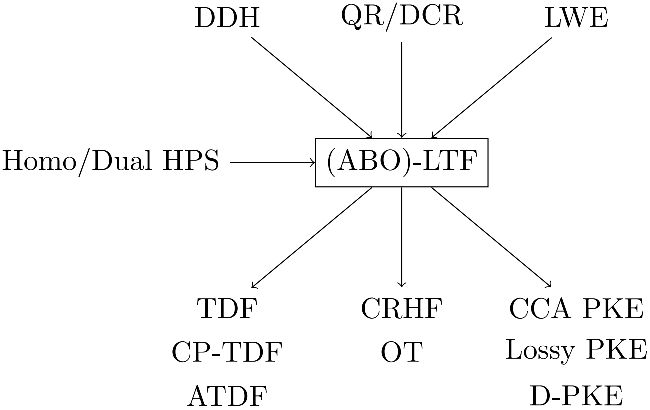
Constructions and Applications

$$\begin{tikzcd} \text{DDH} \arrow[dr] & \text{QR/DCR} \arrow[d] & \text{LWE} \arrow[dl] \\ & \boxed{\text{(ABO)-LTF}} & \end{tikzcd}$$

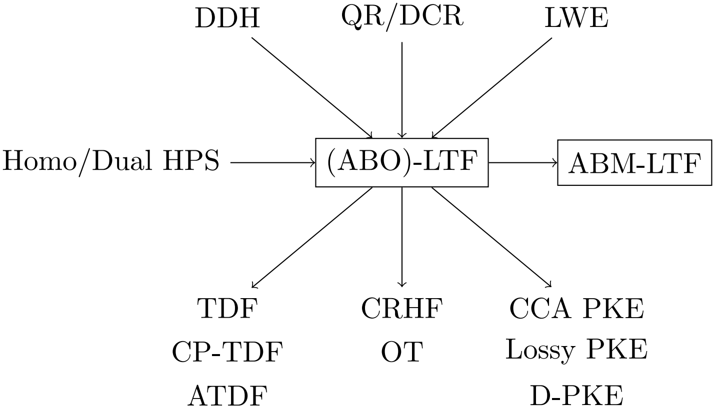
Constructions and Applications



Constructions and Applications



Constructions and Applications



Motivations

In all applications of LTF:

- normal mode: injective+trapdoor fulfill functionality
- lossy mode: establish security

Motivations

In all applications of LTF:

- normal mode: injective+trapdoor fulfill functionality
- lossy mode: establish security

However, the full power of LTF is

- expensive: large key size/high computation cost
- overkill: some applications (e.g., injective OWF, CRHF) do not require a trapdoor, but only normal $\approx_c$ lossy

A central goal in cryptography is to base cryptosystems on primitives that are as weak as possible.

- Peikert and Waters conjectured “the weaker notion LF could be achieved more simply and efficiently than LTF”.
- They left the investigation of this question as an interesting problem.

A central goal in cryptography is to base cryptosystems on primitives that are as weak as possible.

- Peikert and Waters conjectured “the weaker notion LF could be achieved more simply and efficiently than LTF”.
- They left the investigation of this question as an interesting problem.

We are motivated to consider the following problems:

How to realize LF efficiently?
Are there any other applications of LF?
Can we further weaken the notion of LF?

Outline

1. Backgrounds
2. Regular Lossy Functions
3. Constructions of ABO RLFs
   - Concrete Construction
   - Generic Construction
4. Applications of RLFs
   - Leakage-Resilient OWFs
   - Leakage-Resilient MAC
   - Leakage-Resilient CCA-secure KEM

A Simple But Important Observation

When trapdoor is not required for normal mode, the injective property may also be unnecessary.

A Simple But Important Observation

When trapdoor is not required for normal mode, the injective property may also be unnecessary.

This observation leads to our further relaxation of LFs

Regular Lossy Functions

A Simple But Important Observation

When trapdoor is not required for normal mode, the injective property may also be unnecessary.

This observation leads to our further relaxation of LFs

Regular Lossy Functions

Intuition: the output should preserves much *min-entropy* of input

- In RLFs, functions of normal mode could also be lossy, but has to lose in a regular manner.

A Simple But Important Observation

When trapdoor is not required for normal mode, the injective property may also be unnecessary.

This observation leads to our further relaxation of LFs

Regular Lossy Functions

Intuition: the output should preserves much *min-entropy* of input

- In RLFs, functions of normal mode could also be lossy, but has to lose in a regular manner.

Definition 1

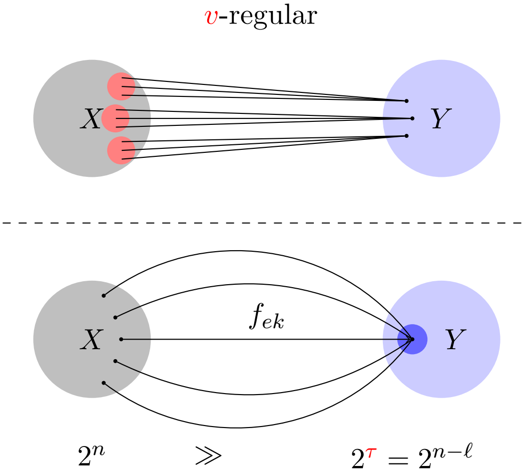
f is v-to-1 (or v-regular) if $\max_y |f^{-1}(y)| \leq v$.

Regular Lossy Functions

normal
$\mathsf{Gen}(\lambda) \to ek$

$\approx_c$

$\mathsf{Gen}(\lambda) \to ek$
lossy



- When $\upsilon = 1$, RLFs specialize to standard LFs

Remarks

Why we choose regularity but not image size to capture normal mode?

Remarks

Why we choose regularity but not image size to capture normal mode?

- image size is a *global* characterization, which only suffices to give the lower bound of $\tilde{\mathsf{H}}_\infty(x|f(x))$ by the chain rule.

Remarks

Why we choose regularity but not image size to capture normal mode?

- image size is a *global* characterization, which only suffices to give the lower bound of $\tilde{\mathsf{H}}_\infty(x|f(x))$ by the chain rule.
- In contrast, regularity is a *local* characterization, which suffices to give the lower bound of $\mathsf{H}_\infty(f(x))$.

Remarks

Why we choose regularity but not image size to capture normal mode?

- image size is a *global* characterization, which only suffices to give the lower bound of $\tilde{\mathsf{H}}_\infty(x|f(x))$ by the chain rule.
- In contrast, regularity is a *local* characterization, which suffices to give the lower bound of $\mathsf{H}_\infty(f(x))$.

The following technical lemma establishes the relation between the min-entropy of x and $f(x)$:

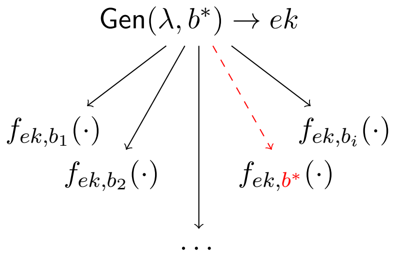
Lemma 2

Let f be a v-to-1 function and x be a random variable over the domain:

$$\mathsf{H}_\infty(f(x)) \geq \mathsf{H}_\infty(x) - \log v$$

All-But-One Regular Lossy Functions

- $\mathsf{Gen}(\lambda, b^*)$ has an extra input: branch $b^* \in B$.



b^* is hidden from ek

$$f_{ek,b}(\cdot) = \begin{cases} \text{lossy} & b = b^* \\ \text{regular} & b \neq b^* \end{cases}$$

$$\text{RLF} \Leftrightarrow \text{ABO-RLF}$$

Outline

Concrete Construction from the DDH Assumption

Matrix approach for ABO-LTFs $f_{ek,b}(x) \rightarrow y$ due to Peikert and Waters

Concrete Construction from the DDH Assumption

Matrix approach for ABO-LTFs $f_{ek,b}(x) \to y$ due to Peikert and Waters

$x \in \mathbb{Z}_2^n$

Concrete Construction from the DDH Assumption

Matrix approach for ABO-LTFs $f_{ek,b}(x) \to y$ due to Peikert and Waters

$$\mathsf{Gen}(\lambda, b^*) \to ek$$

$x \in \mathbb{Z}_2^n$

Concrete Construction from the DDH Assumption

Matrix approach for ABO-LTFs $f_{ek,b}(x) \to y$ due to Peikert and Waters

$$\mathsf{Gen}(\lambda, b^*) \to ek$$

$$\mathsf{GenConceal}(n,m) = g^{\mathbf{V}}$$

$$x \in \mathbb{Z}_2^n \qquad \begin{pmatrix} g^{r_1 s_1} & g^{r_1 s_2} & \dots & g^{r_1 s_m} \\ g^{r_2 s_1} & g^{r_2 s_2} & \dots & g^{r_2 s_m} \\ \vdots & \vdots & \vdots & \vdots \\ g^{r_n s_1} & g^{r_n s_2} & \dots & g^{r_n s_m} \end{pmatrix}$$

Concrete Construction from the DDH Assumption

Matrix approach for ABO-LTFs $f_{ek,b}(x) \to y$ due to Peikert and Waters

$$\mathsf{Gen}(\lambda, b^*) \to ek$$

$$\mathsf{GenConceal}(n, m) = g^{\mathbf{V}}$$

$$x \in \mathbb{Z}_2^n \qquad \begin{pmatrix} g^{r_1 s_1} & g^{r_1 s_2} & \dots & g^{r_1 s_m} \\ g^{r_2 s_1} & g^{r_2 s_2} & \dots & g^{r_2 s_m} \\ \vdots & \vdots & \vdots & \vdots \\ g^{r_n s_1} & g^{r_n s_2} & \dots & g^{r_n s_m} \end{pmatrix}$$

$$\text{DDH} \Rightarrow \approx_c U_{\mathbb{G}^{n \times m}}$$

Concrete Construction from the DDH Assumption

Matrix approach for ABO-LTFs $f_{ek,b}(x) \to y$ due to Peikert and Waters

$$\mathsf{Gen}(\lambda, b^*) \to ek$$

$$\mathsf{GenConceal}(n,m) = g^{\mathbf{V}}$$

$$x \in \mathbb{Z}_2^n \qquad \begin{pmatrix} g^{r_1 s_1} & g^{r_1 s_2} & \dots & g^{r_1 s_m} \\ g^{r_2 s_1} & g^{r_2 s_2} & \dots & g^{r_2 s_m} \\ \vdots & \vdots & \vdots & \vdots \\ g^{r_n s_1} & g^{r_n s_2} & \dots & g^{r_n s_m} \end{pmatrix} - b^*(\mathbf{e}_1, \dots, \mathbf{e}_m)$$

$$\text{DDH} \Rightarrow \approx_c U_{\mathbb{G}^{n\times m}}$$

Concrete Construction from the DDH Assumption

Matrix approach for ABO-LTFs $f_{ek,b}(x) \to y$ due to Peikert and Waters

$$\mathsf{Gen}(\lambda, b^*) \to ek$$

$$\mathsf{GenConceal}(n, m) = g^{\mathbf{V}}$$

$$x \in \mathbb{Z}_2^n \times \begin{pmatrix} g^{r_1 s_1} & g^{r_1 s_2} & \dots & g^{r_1 s_m} \\ g^{r_2 s_1} & g^{r_2 s_2} & \dots & g^{r_2 s_m} \\ \vdots & \vdots & \vdots & \vdots \\ g^{r_n s_1} & g^{r_n s_2} & \dots & g^{r_n s_m} \end{pmatrix} - b^*(\mathbf{e}_1, \dots, \mathbf{e}_m) + b(\mathbf{e}_1, \dots, \mathbf{e}_m) \to y \in \mathbb{G}^m$$

$$\text{DDH} \Rightarrow \approx_c U_{\mathbb{G}^{n \times m}}$$

Concrete Construction from the DDH Assumption

Matrix approach for ABO-LTFs $f_{ek,b}(x) \to y$ due to Peikert and Waters

$$\mathsf{Gen}(\lambda, b^*) \to ek$$

$$\mathsf{GenConceal}(n,m) = g^{\mathbf{V}}$$

$$x \in \mathbb{Z}_2^n \;\times\; \begin{pmatrix} g^{r_1 s_1} & g^{r_1 s_2} & \dots & g^{r_1 s_m} \\ g^{r_2 s_1} & g^{r_2 s_2} & \dots & g^{r_2 s_m} \\ \vdots & \vdots & \vdots & \vdots \\ g^{r_n s_1} & g^{r_n s_2} & \dots & g^{r_n s_m} \end{pmatrix} - b^*(\mathbf{e}_1, \dots, \mathbf{e}_m) + b(\mathbf{e}_1, \dots, \mathbf{e}_m) \to y \in \mathbb{G}^m$$

$$\text{DDH} \Rightarrow \approx_c U_{\mathbb{G}^{n\times m}}$$

To ensure invertible property

- input space is restricted to $\mathbb{Z}_2^n$ (a.k.a. $\{0,1\}^n$)
- column dimension $m = n + 1$

$$\mathsf{Gen}(\lambda, b^*) \to ek$$

$$\mathsf{GenConceal}(n,m) = g^{\mathbf{V}}$$

$$x \in \mathbb{Z}_p^n \times \begin{pmatrix} g^{r_1 s_1} & g^{r_1 s_2} & \dots & g^{r_1 s_m} \\ g^{r_2 s_1} & g^{r_2 s_2} & \dots & g^{r_2 s_m} \\ \vdots & \vdots & \vdots & \vdots \\ g^{r_n s_1} & g^{r_n s_2} & \dots & g^{r_n s_m} \end{pmatrix} - b^*(\mathbf{e}_1, \dots, \mathbf{e}_m) + b(\mathbf{e}_1, \dots, \mathbf{e}_m) \to y \in \mathbb{G}^m$$

$$\text{DDH} \Rightarrow \approx_c U_{\mathbb{G}^{n \times m}}$$

(ABO)-RLFs do not require invertible or even injective

$$\mathsf{Gen}(\lambda, b^*) \to ek$$

$$\mathsf{GenConceal}(n,m) = g^{\mathbf{V}} \qquad m \ll n$$

$$x \in \mathbb{Z}_p^n \gg \mathbb{Z}_2^n \quad \times \quad \begin{pmatrix} g^{r_1 s_1} & g^{r_1 s_2} & \dots & g^{r_1 s_m} \\ g^{r_2 s_1} & g^{r_2 s_2} & \dots & g^{r_2 s_m} \\ \vdots & \vdots & \vdots & \vdots \\ g^{r_n s_1} & g^{r_n s_2} & \dots & g^{r_n s_m} \end{pmatrix} - b^*(\mathbf{e}_1, \dots, \mathbf{e}_m) + b(\mathbf{e}_1, \dots, \mathbf{e}_m) \to y \in \mathbb{G}^m$$

$$\mathrm{DDH} \Rightarrow \approx_c U_{\mathbb{G}^{n\times m}}$$

(ABO)-RLFs do not require invertible or even injective

$$\mathsf{Gen}(\lambda, b^*) \to ek$$

$$\mathsf{GenConceal}(n,m) = g^{\mathbf{V}} \qquad m \ll n$$

$$x \in \mathbb{Z}_p^n \times \begin{pmatrix} g^{r_1 s_1} & g^{r_1 s_2} & \dots & g^{r_1 s_m} \\ g^{r_2 s_1} & g^{r_2 s_2} & \dots & g^{r_2 s_m} \\ \vdots & \vdots & \vdots & \vdots \\ g^{r_n s_1} & g^{r_n s_2} & \dots & g^{r_n s_m} \end{pmatrix} - b^*(\mathbf{e}_1, \dots, \mathbf{e}_m) + b(\mathbf{e}_1, \dots, \mathbf{e}_m) \to y \in \mathbb{G}^m$$

$$\gg \mathbb{Z}_2^n$$

$$\text{DDH} \Rightarrow \approx_c U_{\mathbb{G}^{n\times m}}$$

Lemma 3

The above construction constitutes $(p^{n-m}, \log p)$-ABO-RLF.

- $\forall b \neq b^*$, $\mathrm{rank}(\mathbf{Y} + b\mathbf{I}') = m$ and $\#$(solution space) for every $y \in \mathbb{G}^m$ is p^{n-m}.
- $b = b^*$, $\mathrm{rank}(\mathbf{Y} + b\mathbf{I}') = 1$ and thus the image size is at most p.
- Pseudorandomness of $\mathbf{C} = g^{\mathbf{V}} \Rightarrow$ hidden lossy branch

Summary and Comparison

Our DDH construction applies to extended DDH $\rightsquigarrow$ generalize DDH, QR, DCR

- We have a more efficient and direct DCR-based construction

ABO-LTF/RLF	Assump.	Input	Lossiness	Key	Efficiency
ABO-LTF[PW08]	DDH	2^n	$n - \log p$	$nm\lvert\mathbb{G}\rvert$	nm Add
ABO-RLF	DDH	p^n	$(n-1)\log p$	$nm\lvert\mathbb{G}\rvert$	nm (Exp+Add)
ABO-LTF[FGK+13]	DCR	N^2	$\log N$	$\lvert\mathbb{Z}_{N^3}^*\rvert$	1 Exp
ABO-LF	$N^2/4$	DCR	$\log N$	$\lvert\mathbb{Z}_{N^2}^*\rvert$	1 Exp

Generic Construction from HPS

Wee (Eurocrypt 2012): dual HPS $\Rightarrow$ LTF

- dual HPS: HPS satisfing strong property
- No efficient ABO construction is known

Generic Construction from HPS

Wee (Eurocrypt 2012): dual HPS $\Rightarrow$ LTF

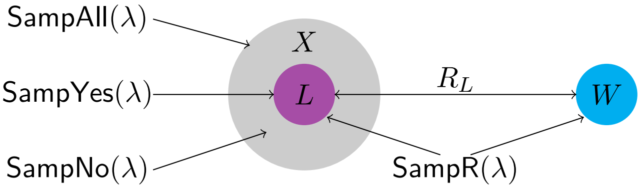
- dual HPS: HPS satisfing strong property
- No efficient ABO construction is known

We show $\boxed{\text{HPS} \Rightarrow \text{ABO-RLF}}$

- exploit algebra property of the underlying SMP

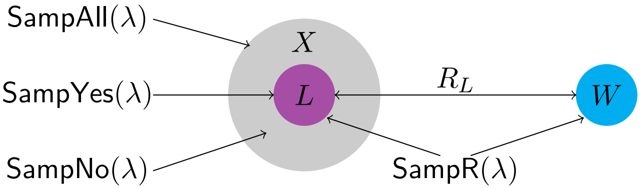
(Algebra) Subset Membership Problem

Task: distinguish $U_X \approx_c U_L$ Solution: $\{0,1\}$



(Algebra) Subset Membership Problem

Task: distinguish $U_X \approx_c U_L$ Solution: $\{0, 1\}$

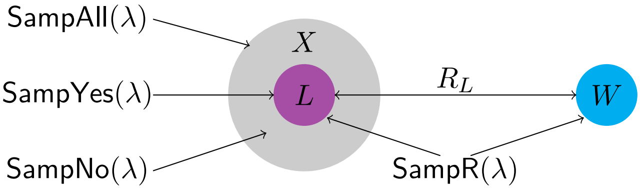


Algebra SMP (mild & natural)

- X forms an Abelian group, L forms a subgroup of X
- The quotient group $H = X/L$ is cyclic with order $p = |X|/|L|$

(Algebra) Subset Membership Problem

Task: distinguish $U_X \approx_c U_L$ Solution: $\{0, 1\}$



Algebra SMP (mild & natural)

- X forms an Abelian group, L forms a subgroup of X
- The quotient group $H = X/L$ is cyclic with order $p = |X|/|L|$

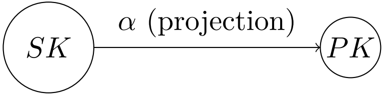
Algebraic properties $\Rightarrow$ two useful facts

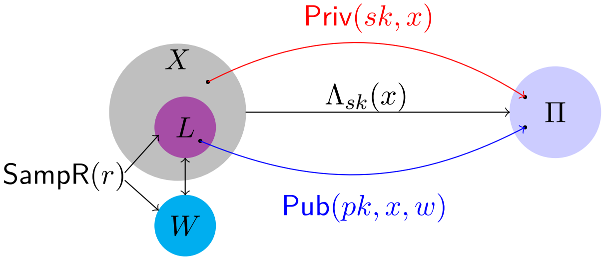
1. Let $\bar{a} = aL$ for some $a \in X \backslash L$ be a generator of H, the co-sets $(aL, 2aL, \ldots, (p-1)aL, paL = L)$ constitute a partition of X.
2. For each $x \in L$, $ia + x \notin L$ for $1 \leq i < p$

Hash Proof System

- $L \subset X$ — language defined by R_L where SMP holds.
- HPS equips $L \subset X$ with Gen, Priv, Pub.

$$\mathsf{Gen}(\lambda) \to (pk, sk)$$
$$\text{s.t. } \alpha(sk) = pk$$



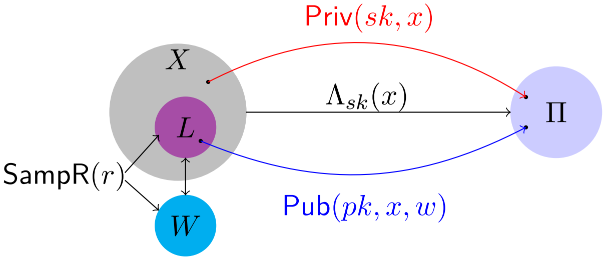


Hash Proof System

- $L \subset X$ — language defined by R_L where SMP holds.
- HPS equips $L \subset X$ with Gen, Priv, Pub.

$$\mathsf{Gen}(\lambda) \to (pk, sk)$$
$$\text{s.t. } \alpha(sk) = pk$$

$$SK \xrightarrow{\alpha \text{ (projection)}} PK$$



Projective: $\forall x \in L$, $\Lambda_{sk}(x)$ is uniquely determined by x and $pk \leftarrow \alpha(sk)$.

ABO-RLF from HPS for ASMP

Let aL be a generator for $H = X/L$, we build ABO-RLF from HPS for ASMP as below:

- $\mathsf{Gen}(\lambda, b^*)$: $(x, w) \leftarrow \mathsf{SampYes}(\lambda)$, output $ek = -b^*a + x$
- $f_{ek,b}(sk)$: output $\alpha(sk)||\Lambda_{sk}(ek + ba)$

ABO-RLF from HPS for ASMP

Let aL be a generator for $H = X/L$, we build ABO-RLF from HPS for ASMP as below:

- $\mathsf{Gen}(\lambda, b^*)$: $(x, w) \leftarrow \mathsf{SampYes}(\lambda)$, output $ek = -b^*a + x$
- $f_{ek,b}(sk)$: output $\alpha(sk)||\Lambda_{sk}(ek + ba)$

Lemma 4

Assume $g_x(sk) := \alpha(sk)||\Lambda_{sk}(x)$ is v-regular for any $x \notin L$. The above construction is $(v, \log|\mathrm{Img}\alpha|)$-ABO-RLF under ASMP.

ABO-RLF from HPS for ASMP

Let aL be a generator for $H = X/L$, we build ABO-RLF from HPS for ASMP as below:

- $\mathsf{Gen}(\lambda, b^*)$: $(x, w) \leftarrow \mathsf{SampYes}(\lambda)$, output $ek = -b^*a + x$
- $f_{ek,b}(sk)$: output $\alpha(sk)||\Lambda_{sk}(ek + ba)$

Lemma 4

Assume $g_x(sk) := \alpha(sk)||\Lambda_{sk}(x)$ is v-regular for any $x \notin L$. The above construction is $(v, \log|\mathrm{Img}\alpha|)$-ABO-RLF under ASMP.

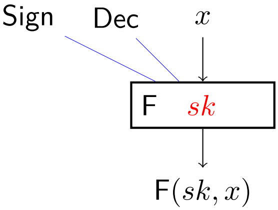
- $ek + ba = x + (b - b^*)a \notin L$ if $b \neq b^* \Rightarrow v$-regular
- $ek + ba = x + (b - b^*)a \in L$ if $b = b^* \Rightarrow$ lossy by the projective property
- ASMP $\Rightarrow$ Hidden lossy branch. For any $b_0^*, b_1^* \in \mathbb{Z}_p$:

$$(-b_0^*a + x) \approx_c (b_0^*a + u) \equiv (b_1^*a + u) \approx_c (b_1^*a + x)$$

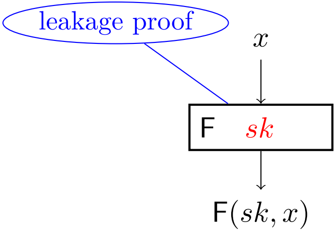
where $u \xleftarrow{\mathrm{R}} X$.

Outline

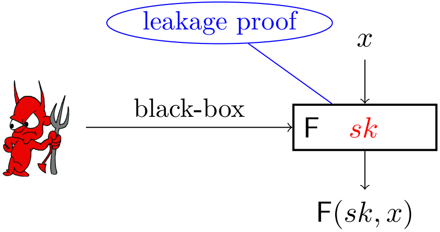
Leakage-Resilient Cryptography



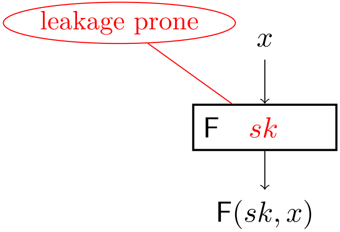
Leakage-Resilient Cryptography



Leakage-Resilient Cryptography



Leakage-Resilient Cryptography



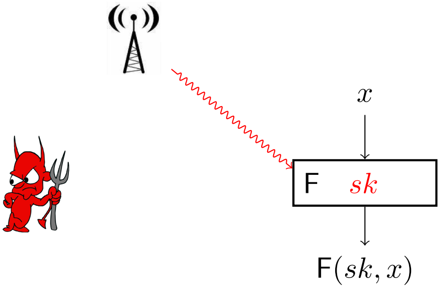
Leakage-Resilient Cryptography

leakage attacks (since 1996) invalidate this idealized assumption

$$\begin{tikzcd} x \arrow[d] \\ \boxed{\mathsf{F} \quad sk} \arrow[d] \\ \mathsf{F}(sk, x) \end{tikzcd}$$

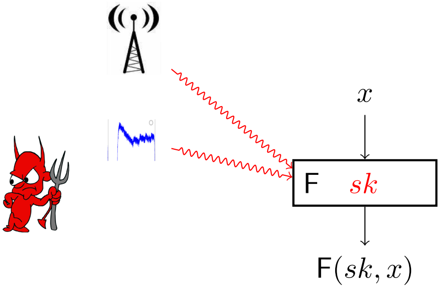
Leakage-Resilient Cryptography

leakage attacks (since 1996) invalidate this idealized assumption



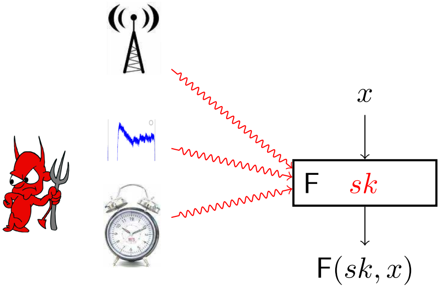
Leakage-Resilient Cryptography

leakage attacks (since 1996) invalidate this idealized assumption



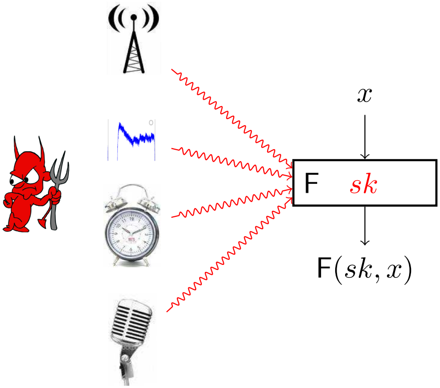
Leakage-Resilient Cryptography

leakage attacks (since 1996) invalidate this idealized assumption



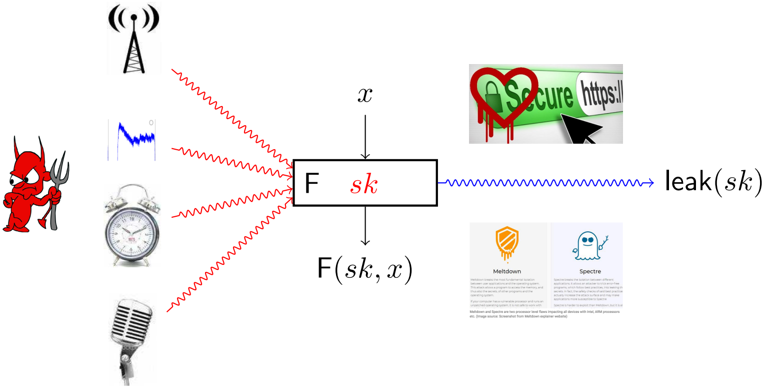
Leakage-Resilient Cryptography

leakage attacks (since 1996) invalidate this idealized assumption



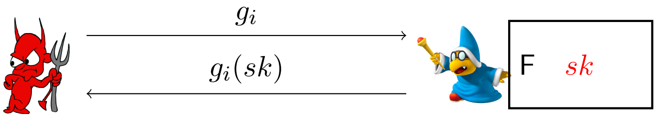
Leakage-Resilient Cryptography

leakage attacks (since 1996) invalidate this idealized assumption



Bounded Leakage Model

In this work, we focus on a simple yet general leakage model called Bounded Leakage Model



$$\sum |g_i(sk)| \leq |sk|$$

Leakage-Resilient OWFs

Leakage-Resilient OWFs

$$x^* \xleftarrow{\mathrm{R}} \{0,1\}^n$$
$$y^* \leftarrow f(x^*)$$

 $\xleftarrow{\quad (f, y^*) \quad}$ 

Leakage-Resilient OWFs

$$x^* \xleftarrow{R} \{0,1\}^n$$
$$y^* \leftarrow f(x^*)$$

 $\xleftarrow{\quad (f, y^*) \quad}$ $\xrightarrow{\quad g_i \quad}$ 

Leakage-Resilient OWFs

$$\xleftarrow{\quad (f, y^*) \quad}$$

$$\xrightarrow{\quad g_i \quad}$$

$$\xleftarrow{\quad g_i(x^*) \quad}$$

$$x^* \xleftarrow{\text{R}} \{0,1\}^n$$
$$y^* \leftarrow f(x^*)$$

Leakage-Resilient OWFs

$$\xleftarrow{\quad (f, y^*) \quad}$$

$$\xrightarrow{\quad g_i \quad}$$

$$\xleftarrow{\quad g_i(x^*) \quad}$$

$$\xrightarrow{\quad x \quad}$$

$$x^* \xleftarrow{R} \{0,1\}^n$$

$$y^* \leftarrow f(x^*)$$

$$x =? x^*$$

Leakage-Resilient OWFs

$$\xleftarrow{\quad (f, y^*) \quad}$$

$$\xrightarrow{\quad g_i \quad}$$

$$\xleftarrow{\quad g_i(x^*) \quad}$$

$$\xrightarrow{\quad x \quad}$$

$$x^* \xleftarrow{\mathrm{R}} \{0,1\}^n$$

$$y^* \leftarrow f(x^*)$$

$$x =? x^*$$

Theorem 5

The normal mode of $(1, \tau)$-RLFs (i.e., LFs) over domain $\{0,1\}^n$ constitutes a family of ℓ-leakage-resilient injective OWFs, for any $\ell \leq n - \tau - \omega(\log \lambda)$.

Game 0: real game

1. Setup: $\mathcal{CH}$ generates $f \leftarrow \mathrm{RLF}.\mathsf{GenNormal}(\lambda)$, picks $x^* \xleftarrow{\mathrm{R}} \{0,1\}^n$ and sends $(f, y^* = f(x^*))$ to $\mathcal{A}$.
2. Leakage queries: $\mathcal{A} \hookrightarrow g_i$, $\mathcal{CH}$ responds with $g_i(x^*)$.
3. Invert: $\mathcal{A}$ outputs x and wins if $x = x^*$.

$$\mathsf{Adv}_{\mathcal{A}}(\lambda) = \Pr[S_0]$$

Game 1: same as Game 0 except that:

1. Setup: $\mathcal{CH}$ generates $\boxed{f \leftarrow \mathrm{RLF}.\mathsf{GenLossy}(\lambda)}$.

Security of RLFs $\Rightarrow |\Pr[S_1] - \Pr[S_0]| \leq \mathsf{negl}(\lambda)$

In Game 1, $\tilde{\mathsf{H}}_\infty(x^*|(y^*, leak)) \geq n - \tau - \ell$.

- By the parameter choice, $\tilde{\mathsf{H}}_\infty(x^*|(y^*, leak)) \geq \omega(\log\lambda) \Rightarrow \Pr[S_1] \leq \mathsf{negl}(\lambda)$ even w.r.t. unbounded adversary

Leakage-Resilient MAC

Leakage-Resilient MAC

$$\xleftarrow{\quad pp \quad} \qquad (pp, k) \leftarrow \mathsf{Setup}(\lambda)$$

Leakage-Resilient MAC

$$\xleftarrow{\quad pp \quad} \qquad (pp, k) \leftarrow \mathsf{Setup}(\lambda)$$

$$\xrightarrow{\quad m_i \quad}$$

Leakage-Resilient MAC

$$\xleftarrow{\qquad pp \qquad}$$

$$\xrightarrow{\qquad m_i \qquad}$$

$$\xleftarrow{t_i \leftarrow \mathsf{Tag}(k, m_i)}$$

$(pp, k) \leftarrow \mathsf{Setup}(\lambda)$

Leakage-Resilient MAC

$(pp, k) \leftarrow \mathsf{Setup}(\lambda)$

$\xleftarrow{\quad pp \quad}$

$\xrightarrow{\quad m_i \quad}$

$\xleftarrow{\quad t_i \leftarrow \mathsf{Tag}(k, m_i) \quad}$

$\xrightarrow{\quad g_i \quad}$

Leakage-Resilient MAC

$(pp, k) \leftarrow \mathsf{Setup}(\lambda)$

$\xleftarrow{\quad pp \quad}$

$\xrightarrow{\quad m_i \quad}$

$\xleftarrow{\quad t_i \leftarrow \mathsf{Tag}(k, m_i) \quad}$

$\xrightarrow{\quad g_i \quad}$

$\xleftarrow{\quad g_i(k) \quad}$

Leakage-Resilient MAC

$$(pp, k) \leftarrow \mathsf{Setup}(\lambda)$$

$$\xleftarrow{\quad pp \quad}$$

$$\xrightarrow{\quad m_i \quad}$$

$$\xleftarrow{\quad t_i \leftarrow \mathsf{Tag}(k, m_i) \quad}$$

$$\xrightarrow{\quad g_i \quad}$$

$$\xleftarrow{\quad g_i(k) \quad}$$

$$\xrightarrow{\quad (m^*, t^*) \quad}$$

$$\mathsf{Vefy}(k, m^*, t^*) = 1$$

$$(m^*, t^*) \neq (m_i, t_i)$$

Leakage-Resilient MAC

$$\xleftarrow{\quad pp \quad} \qquad (pp, k) \leftarrow \mathsf{Setup}(\lambda)$$

$$\xrightarrow{\quad m_i \quad}$$

$$\xleftarrow{\quad t_i \leftarrow \mathsf{Tag}(k, m_i) \quad}$$

$$\xrightarrow{\quad g_i \quad}$$

$$\xleftarrow{\quad g_i(k) \quad}$$

$$\xrightarrow{\quad (m^*, t^*) \quad}$$

$$\mathsf{Vefy}(k, m^*, t^*) = 1$$
$$(m^*, t^*) \neq (m_i, t_i)$$

Strong unforgeability can be relaxed in several ways:

Leakage-Resilient MAC

$$(pp, k) \leftarrow \mathsf{Setup}(\lambda)$$

$$\xleftarrow{\quad pp \quad}$$

$$\xrightarrow{\quad m_i \quad}$$

$$\xleftarrow{\quad t_i \leftarrow \mathsf{Tag}(k, m_i) \quad}$$

$$\xrightarrow{\quad g_i \quad}$$

$$\xleftarrow{\quad g_i(k) \quad}$$

$$\xrightarrow{\quad (m^*, t^*) \quad}$$

$$\mathsf{Vefy}(k, m^*, t^*) = 1$$
$$(m^*, t^*) \neq (m_i, t_i)$$

Strong unforgeability can be relaxed in several ways:

- One-time: $\mathcal{A}$ only makes one tag query

Leakage-Resilient MAC

pp ←

m_i →

$t_i \leftarrow \mathsf{Tag}(k, m_i)$ ←

g_i →

$g_i(k)$ ←

(m^*, t^*) →

$(pp, k) \leftarrow \mathsf{Setup}(\lambda)$

$\mathsf{Vefy}(k, m^*, t^*) = 1$
$(m^*, t^*) \neq (m_i, t_i)$

Strong unforgeability can be relaxed in several ways:

- One-time: $\mathcal{A}$ only makes one tag query
- Selective: $\mathcal{A}$ commits the target message before seeing pp

Construction

Ingredient
(v, τ)-ABORLF

Construction

Ingredient
(v, τ)-ABORLF

$$\boxed{\mathsf{KeyGen}} \downarrow$$

$$ek \leftarrow \mathrm{ABORLF}.\mathsf{Gen}(\lambda, 0^d)$$
$$k \xleftarrow{\mathrm{R}} \{0,1\}^n$$

Construction

Ingredient

(v, τ)-ABORLF

KeyGen

$ek \leftarrow \text{ABORLF}.\mathsf{Gen}(\lambda, 0^d)$

$k \xleftarrow{\text{R}} \{0,1\}^n$

m → Tag

$t \leftarrow f_{ek,m}(k)$

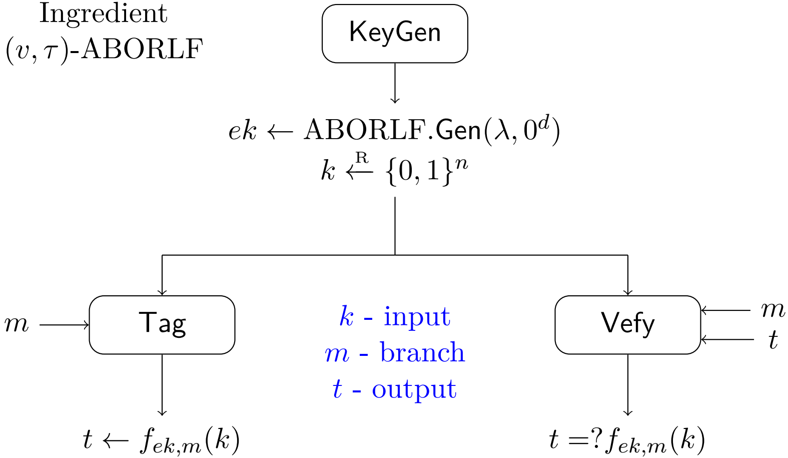
k - input

m - branch

t - output

Construction

Ingredient
(v, τ)-ABORLF



Theorem 6

The above MAC is ℓ-leakage-resilient seletively one-time sUF for any $\ell \leq n - \tau - \log v - \omega(\log \lambda)$.

Game 0: (real game)

1. Setup: $\mathcal{A} \nrightarrow m^*$, $\mathcal{CH}$ generates $ek \leftarrow \text{ABORLF}.\mathsf{Gen}(\lambda, 0^d)$, picks $k \xleftarrow{\text{R}} \{0,1\}^n$, computes $t^* \leftarrow f_{ek,m^*}(k)$ and then sends (ek, t^*) to $\mathcal{A}$.
2. Leakage queries: $\mathcal{A} \nrightarrow g_i$, $\mathcal{CH}$ responds with $g_i(k)$.
3. Forge: $\mathcal{A} \rightarrow (m, t)$ and wins if $m \neq m^* \wedge t = f_{ek,m}(k)$.

$$\mathsf{Adv}_{\mathcal{A}}(\lambda) = \Pr[S_0]$$

Game 1: same as Game 0 except that

1. Setup: $\mathcal{CH}$ generates $\boxed{ek \leftarrow \text{ABORLF}.\mathsf{Gen}(\lambda, m^*)}$.

Hidden lossy branch $\Rightarrow |\Pr[S_1] - \Pr[S_0]| \leq \mathsf{negl}(\lambda)$

In Game 1, $\mathcal{A}$'s view includes $(ek, leak, t^*)$. We have:

$$\begin{aligned}\tilde{\mathsf{H}}_\infty(t|view) &= \tilde{\mathsf{H}}_\infty(t|ek, leak, t^*) \\ &\geq \tilde{\mathsf{H}}_\infty(t|ek) - \ell - \tau \\ &\geq \tilde{\mathsf{H}}_\infty(k|ek) - \log v - \ell - \tau \\ &= n - \log v - \ell - \tau\end{aligned}$$

- By the parameter choice, $\tilde{\mathsf{H}}_\infty(t|view) \geq \omega(\log \lambda) \Rightarrow \Pr[S_1] \leq \mathsf{negl}(\lambda)$ even w.r.t. unbounded adversary.

Leakage-Resilient CCA-secure KEM

Leakage-Resilient CCA-secure KEM

$$\xleftarrow{\quad\quad\quad\quad pk \quad\quad\quad\quad} \qquad (pk, sk) \leftarrow \mathsf{Setup}(\lambda)$$

Leakage-Resilient CCA-secure KEM

$$\xleftarrow{\quad pk \quad}$$

$$\xrightarrow{\quad c_i \quad}$$

$(pk, sk) \leftarrow \mathsf{Setup}(\lambda)$

Leakage-Resilient CCA-secure KEM

$(pk, sk) \leftarrow \mathsf{Setup}(\lambda)$

$$\xleftarrow{\quad pk \quad}$$

$$\xrightarrow{\quad c_i \quad}$$

$$\xleftarrow{\quad k_i \leftarrow \mathsf{Decaps}(sk, c_i) \quad}$$

Leakage-Resilient CCA-secure KEM

$$(pk, sk) \leftarrow \mathsf{Setup}(\lambda)$$

$$\xleftarrow{\quad pk \quad}$$

$$\xrightarrow{\quad c_i \quad}$$

$$\xleftarrow{\quad k_i \leftarrow \mathsf{Decaps}(sk, c_i) \quad}$$

$$\xrightarrow{\quad g_i \quad}$$

Leakage-Resilient CCA-secure KEM

$\xleftarrow{\quad pk \quad}$ $(pk, sk) \leftarrow \mathsf{Setup}(\lambda)$

$\xrightarrow{\quad c_i \quad}$

$\xleftarrow{\quad k_i \leftarrow \mathsf{Decaps}(sk, c_i) \quad}$

$\xrightarrow{\quad g_i \quad}$

$\xleftarrow{\quad g_i(sk) \quad}$

Leakage-Resilient CCA-secure KEM

$$\xleftarrow{\quad pk \quad} \qquad (pk, sk) \leftarrow \mathsf{Setup}(\lambda)$$

$$\xrightarrow{\quad c_i \quad}$$

$$\xleftarrow{\quad k_i \leftarrow \mathsf{Decaps}(sk, c_i) \quad}$$

$$\xrightarrow{\quad g_i \quad}$$

$$\xleftarrow{\quad g_i(sk) \quad}$$

$$\xleftarrow{\quad (c^*, k^*_\beta) \quad}$$

$$(c^*, k^*_0) \leftarrow \mathsf{Encap}(pk)$$

$$k^*_1 \xleftarrow{R} K$$

$$\beta \xleftarrow{R} \{0,1\}$$

Leakage-Resilient CCA-secure KEM

$$\xleftarrow{\quad pk \quad} \qquad (pk, sk) \leftarrow \mathsf{Setup}(\lambda)$$

$$\xrightarrow{\quad c_i \quad}$$

$$\xleftarrow{\quad k_i \leftarrow \mathsf{Decaps}(sk, c_i) \quad}$$

$$\xrightarrow{\quad g_i \quad} \qquad (c^*, k_0^*) \leftarrow \mathsf{Encap}(pk)$$

$$\xleftarrow{\quad g_i(sk) \quad} \qquad k_1^* \xleftarrow{\mathrm{R}} K$$

$$\xleftarrow{\quad (c^*, k_\beta^*) \quad} \qquad \beta \xleftarrow{\mathrm{R}} \{0, 1\}$$

$$\xrightarrow{\quad \beta' \quad} \qquad \beta' = \beta$$

Leakage-Resilient CCA-secure KEM

$$\xleftarrow{\quad pk \quad} \qquad (pk, sk) \leftarrow \mathsf{Setup}(\lambda)$$

$$\xrightarrow{\quad c_i \quad}$$

$$\xleftarrow{\quad k_i \leftarrow \mathsf{Decaps}(sk, c_i) \quad}$$

$$\xrightarrow{\quad g_i \quad} \qquad (c^*, k_0^*) \leftarrow \mathsf{Encap}(pk)$$

$$\xleftarrow{\quad g_i(sk) \quad} \qquad k_1^* \xleftarrow{R} K$$

$$\xleftarrow{\quad (c^*, k_\beta^*) \quad} \qquad \beta \xleftarrow{R} \{0,1\}$$

$$\xrightarrow{\quad \beta' \quad} \qquad \beta' = \beta$$

$$|\Pr[\beta' = \beta] - 1/2| \leq \mathsf{negl}(\lambda)$$

Construction

Ingredients

HPS

ABORLF

strong extractor

Construction

Ingredients
HPS
ABORLF
strong extractor

$\boxed{\mathsf{KeyGen}}$

$\downarrow$

$$(pk, sk) \leftarrow \text{HPS}.\mathsf{Gen}(\lambda)$$
$$ek \leftarrow \text{ABORLF}.\mathsf{Gen}(\lambda, 0^{m+d})$$

Construction

Ingredients
HPS
ABORLF
strong extractor

KeyGen

$(pk, sk) \leftarrow \text{HPS}.\mathsf{Gen}(\lambda)$
$ek \leftarrow \text{ABORLF}.\mathsf{Gen}(\lambda, 0^{m+d})$

ek pk

Encaps $\xrightarrow{c = (x, s, t)}$

$(x, w) \leftarrow \mathsf{SampYes}(\lambda)$
$\pi \leftarrow \mathsf{Pub}(pk, x, w)$
$s \xleftarrow{R} \{0,1\}^d$
$t \leftarrow f_{ek, x||s}(\pi)$
$k \leftarrow \mathsf{ext}(\pi, s)$

use π to:
derive k &
authenticate $x||s$

Construction

Ingredients
HPS
ABORLF
strong extractor

KeyGen

$(pk, sk) \leftarrow \mathrm{HPS}.\mathsf{Gen}(\lambda)$
$ek \leftarrow \mathrm{ABORLF}.\mathsf{Gen}(\lambda, 0^{m+d})$

ek | pk → **Encaps**

sk → **Decaps**

Encaps $\xrightarrow{c = (x, s, t)}$ **Decaps**

Encaps:

$(x, w) \leftarrow \mathsf{SampYes}(\lambda)$
$\pi \leftarrow \mathsf{Pub}(pk, x, w)$
$s \xleftarrow{\mathrm{R}} \{0,1\}^d$
$t \leftarrow f_{ek, x||s}(\pi)$
$k \leftarrow \mathsf{ext}(\pi, s)$

use π to:
derive k &
authenticate $x||s$

Decaps:

$\pi \leftarrow \mathsf{Priv}(sk, x)$
$t =? f_{ek, x||s}(\pi)$
$k \leftarrow \mathsf{ext}(\pi, s)$ or $\perp$

Theorem 7

Suppose SMP for $L \subset \{0,1\}^m$ is hard, HPS is ϵ_1-universal$_1$ and $n = \log(1/\epsilon_1)$, ABORLF is (v, τ)-regularly-lossy, ext is $(n - \tau - \ell, \kappa, \epsilon_2)$-strong extractor, then the above KEM is ℓ-LR CCA secure for any $\ell \leq n - \tau - \log v - \omega(\log \lambda)$.

Game 0: (real game)

1. Setup: $\mathcal{CH}$ generates $(pk, sk) \leftarrow \text{HPS}.\mathsf{Gen}(\lambda)$, $ek \leftarrow \text{ABORLF}.\mathsf{Gen}(\lambda, 0^{m+d})$, sends (pk, ek) to $\mathcal{A}$.
2. Leakage queries $\langle g_i \rangle$: $\mathcal{CH}$ responds with $g_i(sk)$.
3. Challenge: $\mathcal{CH}$ picks $\beta \in \{0,1\}$, $s^* \leftarrow \{0,1\}^d$, $(x^*, w^*) \leftarrow \mathsf{SampYes}(\lambda)$, computes $\pi^* \leftarrow \mathsf{Pub}(pk, x^*, w^*)$, $t^* \leftarrow f_{ek, x^*||s^*}(\pi^*)$, $k_0^* \leftarrow \mathsf{ext}(\pi^*, s^*)$, picks $k_1^* \leftarrow \{0,1\}^\kappa$, sends $c^* = (x^*, s^*, t^*)$ and k_β^* to $\mathcal{A}$
4. Decaps queries $\langle c = (x, s, t) \neq c^* \rangle$: $\mathcal{CH}$ computes $\pi \leftarrow \Lambda_{sk}(x)$, output $k \leftarrow \mathsf{ext}(\pi, s)$ if $t = f_{ek, x||s}(\pi)$ and $\perp$ otherwise.

$$\mathsf{Adv}_{\mathcal{A}}(\lambda) = \Pr[S_0] - 1/2$$

Game 1: $\mathcal{CH}$ samples (x^*, w^*) and s^* at Setup.

$$\Pr[S_0] = \Pr[S_1]$$

Game 1: $\mathcal{CH}$ samples (x^*, w^*) and s^* at Setup.

$$\Pr[S_0] = \Pr[S_1]$$

Game 2: $\mathcal{CH}$ generates $ek \leftarrow \text{ABORLF}.\mathsf{Gen}(\lambda, x^*||s^*)$.

$$\text{Hidden lossy branch} \Rightarrow |\Pr[S_2] - \Pr[S_1]| \leq \mathsf{negl}(\lambda)$$

Game 1: $\mathcal{CH}$ samples (x^*, w^*) and s^* at Setup.

$$\Pr[S_0] = \Pr[S_1]$$

Game 2: $\mathcal{CH}$ generates $ek \leftarrow \text{ABORLF}.\mathsf{Gen}(\lambda, x^* || s^*)$.

$$\text{Hidden lossy branch} \Rightarrow |\Pr[S_2] - \Pr[S_1]| \leq \mathsf{negl}(\lambda)$$

Game 3: $\mathcal{CH}$ computes $\pi^* \leftarrow \Lambda_{sk}(x^*)$ via $\mathsf{Priv}(sk, x^*)$.

$$\text{Correctness of HPS} \Rightarrow \Pr[S_3] = \Pr[S_2].$$

Game 1: $\mathcal{CH}$ samples (x^*, w^*) and s^* at Setup.

$$\Pr[S_0] = \Pr[S_1]$$

Game 2: $\mathcal{CH}$ generates $ek \leftarrow \text{ABORLF}.\mathsf{Gen}(\lambda, x^*||s^*)$.

$$\text{Hidden lossy branch} \Rightarrow |\Pr[S_2] - \Pr[S_1]| \leq \mathsf{negl}(\lambda)$$

Game 3: $\mathcal{CH}$ computes $\pi^* \leftarrow \Lambda_{sk}(x^*)$ via $\mathsf{Priv}(sk, x^*)$.

$$\text{Correctness of HPS} \Rightarrow \Pr[S_3] = \Pr[S_2].$$

Game 4: $\mathcal{CH}$ samples x^* via SampNo rather than $\mathsf{SampYes}$.

$$\text{SMP} \Rightarrow |\Pr[S_4] - \Pr[S_3]| \leq \mathsf{negl}(\lambda)$$

Game 1: $\mathcal{CH}$ samples (x^*, w^*) and s^* at Setup.

$$\Pr[S_0] = \Pr[S_1]$$

Game 2: $\mathcal{CH}$ generates $ek \leftarrow \mathrm{ABORLF}.\mathsf{Gen}(\lambda, x^*||s^*)$.

$$\text{Hidden lossy branch} \Rightarrow |\Pr[S_2] - \Pr[S_1]| \leq \mathsf{negl}(\lambda)$$

Game 3: $\mathcal{CH}$ computes $\pi^* \leftarrow \Lambda_{sk}(x^*)$ via $\mathsf{Priv}(sk, x^*)$.

$$\text{Correctness of HPS} \Rightarrow \Pr[S_3] = \Pr[S_2].$$

Game 4: $\mathcal{CH}$ samples x^* via SampNo rather than $\mathsf{SampYes}$.

$$\text{SMP} \Rightarrow |\Pr[S_4] - \Pr[S_3]| \leq \mathsf{negl}(\lambda)$$

Game 5: $\mathcal{CH}$ directly rejects $\langle c = (x, s, t)\rangle$ if $x \notin L$. Define E: $\mathcal{A}$ makes an invalid but well-formed decaps queries, i.e., $f_{ek,x||s}(\Lambda_{sk}(x)) = t$ and $x \in L \wedge (x, s, t) \neq (x^*, s^*, t^*)$.

$$|\Pr[S_5] - \Pr[S_4]| \leq \Pr[E]$$

To calculate $\Pr[E]$, it suffice to bound $\tilde{\mathsf{H}}_\infty(t|view)$.

- $view$: $(pk, ek, leak, x^*, s^*, t^*, k^*_\beta)$
- $t = f_{ek,x||s}(\Lambda_{sk}(x))$

We bound $\tilde{\mathsf{H}}_\infty(t|view)$ via $\tilde{\mathsf{H}}_\infty(\Lambda_{sk}(x)|view)$ as below:

- (x^*, s^*) determines a lossy branch $\Rightarrow$ τ only reveal partial info about sk $\Rightarrow$ $\tilde{\mathsf{H}}_\infty(\Lambda_{sk}(x)|view) \geq n - \ell - \tau - \kappa$
- We must have $(x, s) \neq (x^*, s^*)$, which determines a v-regular branch $\Rightarrow$ $\tilde{\mathsf{H}}_\infty(t|view) \geq \tilde{\mathsf{H}}_\infty(\Lambda_{sk}(x)|view) - \log v$

By the parameter choice, $\tilde{\mathsf{H}}_\infty(t|view) \geq \omega(\log \lambda)$, thus we have:

$$\Pr[E] \leq \mathsf{negl}(\lambda)$$

Game 6: $\mathcal{CH}$ samples $\boxed{k_0^* \leftarrow \{0,1\}^\kappa}$ rather than $k_0^* \leftarrow \mathsf{ext}(\Lambda_{sk}(x^*))$. Next, we analysis $\Delta[view_5, view_6]$.

- define $view' = (pk, ek, leak, x^*, s^*, t^*)$, chain rule $\Rightarrow$ $\tilde{\mathsf{H}}_\infty(\Lambda_{sk}(x^*)|view') \geq n - \ell - \tau$
- randomness extractor $\Rightarrow \Delta[(view', k_{5,0}^*), (view', k_{6,0}^*)] \leq \epsilon_2$.
- responses to all decaps queries in Game 5 and 6 are determined by the same function of $(view', k_{5,0}^*)$ and $(view', k_{6,0}^*)$ resp.

$$\Delta[view_5, view_6] \leq \epsilon_2/2 \leq \mathsf{negl}(\lambda)$$

Putting all the above together, $\mathsf{Adv}_{\mathcal{A}}(\lambda) = \mathsf{negl}(\lambda)$.

Significance

Universal$_1$ HPS + ABO-RLF $\Rightarrow$ LR-CCA KEM

- proper parameter choice $\Rightarrow \ell/|sk| = 1 - o(1)$
- HPS $\Rightarrow$ ABO-RLF

Significance

Universal$_1$ HPS + ABO-RLF $\Rightarrow$ LR-CCA KEM

- proper parameter choice $\Rightarrow \ell/|sk| = 1 - o(1)$
- HPS $\Rightarrow$ ABO-RLF

CCA-secure KEM with optimal leakage rate based solely on universal$_1$ HPS

- go beyond the upper bound posed by Dodis et al. (Asiacrypt 2010)

  leakage-rate only approaching 1/6. Unfortunately, it seems that the hash proof system approach to building CCA encryption is inherently limited to leakage-rates below 1/2: this is because the secret-key consists of two components (one for verifying that the ciphertext is well-formed and one for decrypting it) and the proofs break down if either of the components is individually leaked in its entirety.

- extend to identity-based setting as well

Conclusion

$\boxed{\text{(ABO)-RLFs}}$

Conclusion

$$\begin{tikzcd} \text{eDDH} \arrow[dr] & & \text{DCR} \arrow[dl] \\ & \boxed{\text{(ABO)-RLFs}} & \end{tikzcd}$$

Conclusion

$$
\begin{tikzcd}
\text{eDDH} \arrow[dr] & & \text{DCR} \arrow[dl] & & \\
& \boxed{\text{(ABO)-RLFs}} & & & \text{Algebra SMP} \arrow[d] \\
& & & & \text{HPS} \arrow[lllu, start anchor=west]
\end{tikzcd}
$$

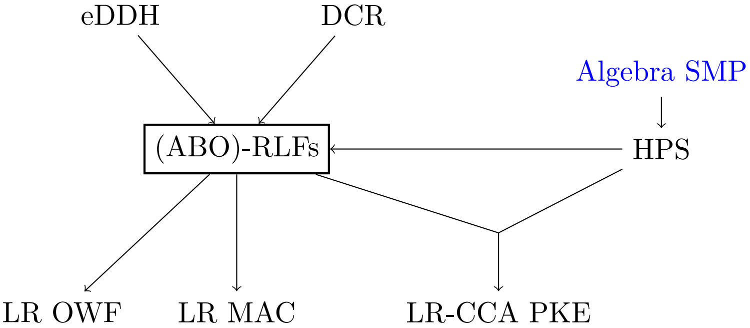
Conclusion

$$
\begin{tikzcd}
& \text{eDDH} \arrow[dr] & & \text{DCR} \arrow[dl] & \text{Algebra SMP} \arrow[d] \\
& & \boxed{\text{(ABO)-RLFs}} \arrow[dll] & & \text{HPS} \arrow[ll] \\
\text{LR OWF} & & & &
\end{tikzcd}
$$

Conclusion

$$
\begin{tikzcd}
 & \text{eDDH} \arrow[ddr] & & \text{DCR} \arrow[ddl] & & \\
 & & & & & \textcolor{blue}{\text{Algebra SMP}} \arrow[d] \\
 & & \boxed{\text{(ABO)-RLFs}} \arrow[dl] \arrow[d] & & & \text{HPS} \arrow[lll] \\
 & \text{LR OWF} & \text{LR MAC} & & &
\end{tikzcd}
$$

Conclusion



Thanks for Your Attention!

Any Questions?

Reference

[FGK+13] David Mandell Freeman, Oded Goldreich, Eike Kiltz, Alon Rosen, and Gil Segev. More constructions of lossy and correlation-secure trapdoor functions. *J. Cryptology*, 26(1):39–74, 2013.

[PW08] Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, STOC 2008*, pages 187–196. ACM, 2008.