Efficient Public-Key Cryptography with Bounded Leakage and Tamper Resilience

Antonio Faonio[1] Daniele Venturi[2]

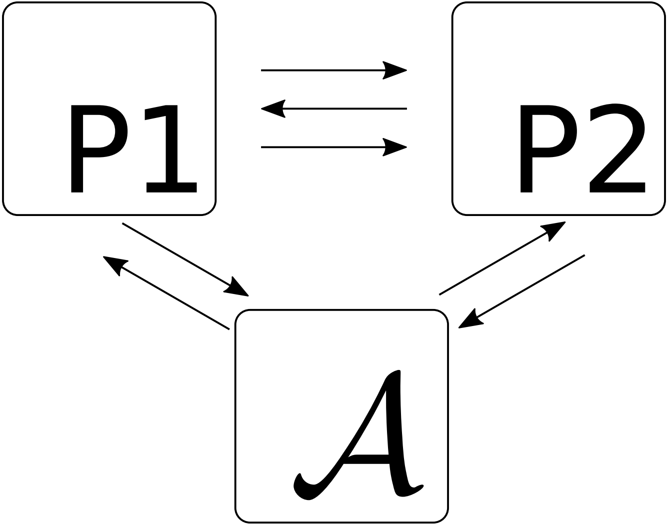
Department of Computer Science, Aarhus University, Aarhus, Denmark

Department of Information Engineering and Computer Science, University of Trento, Trento, Italy

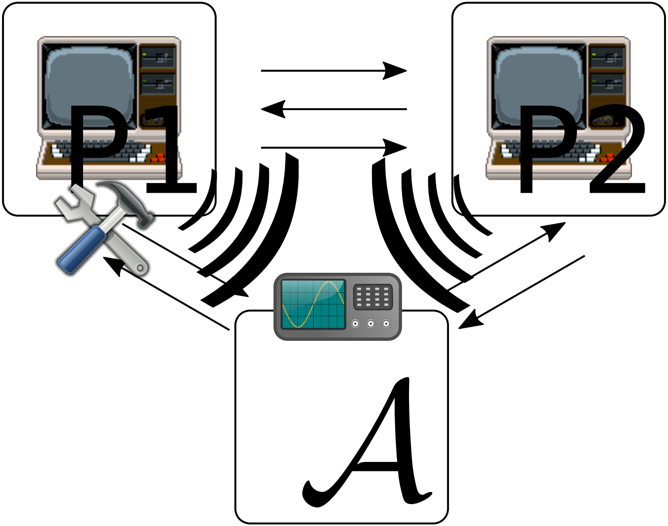
December 8, 2016

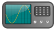
AARHUS UNIVERSITET

(Provable Secure) Crypto before Physical Attacks

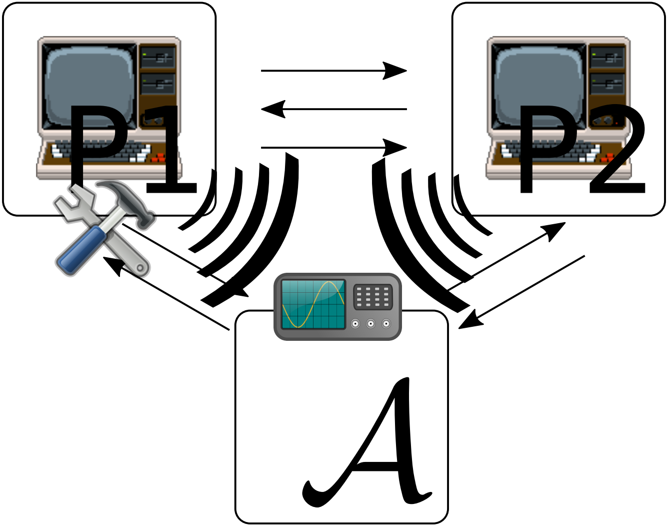


Crypto with Physical Attacks



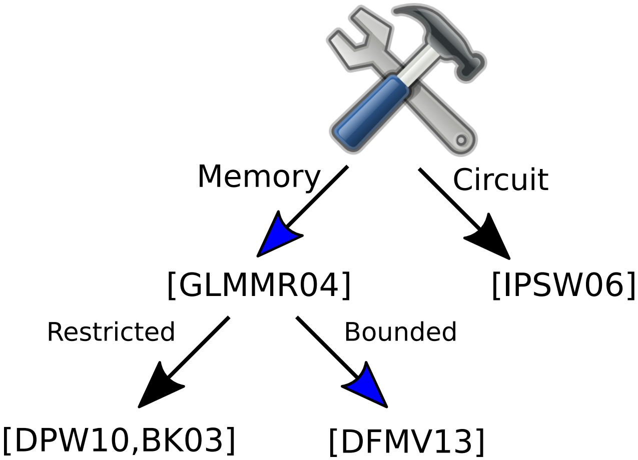
 Leak Attacks [Koc96],

Crypto with Physical Attacks

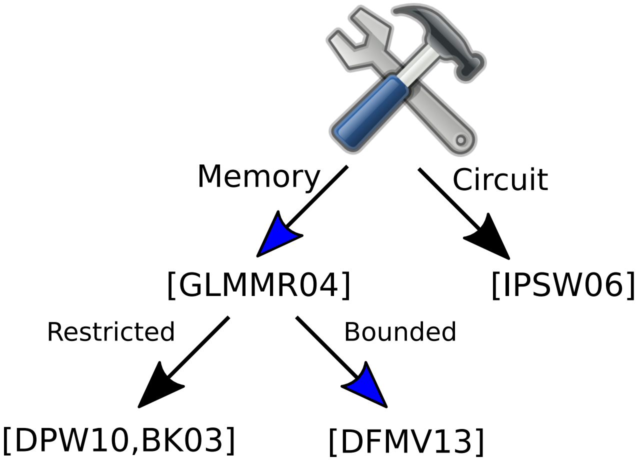


Leak Attacks [Koc96], Tampering Attacks [BDL97]

(Minimal) Related Works



(Minimal) Related Works



- Definitions of Bounded-Tamper (and Leakage) Resilience,
- Identification Scheme and Signatures (ROM),
- CCA-Secure PKE.

Our Contributions

- BTL Signature Scheme.

  Example. The Imp. result of [GLMMR03] does not hold.

Our Contributions

- BTL Signature Scheme.

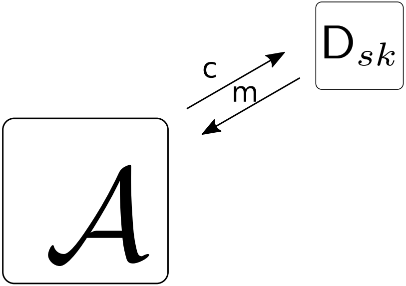
  Example. The Imp. result of [GLMMR03] does not hold.

- BLT CCA Public Key Encryption.
  Naor-Yung paradigm, what about Cramer-Shoup?

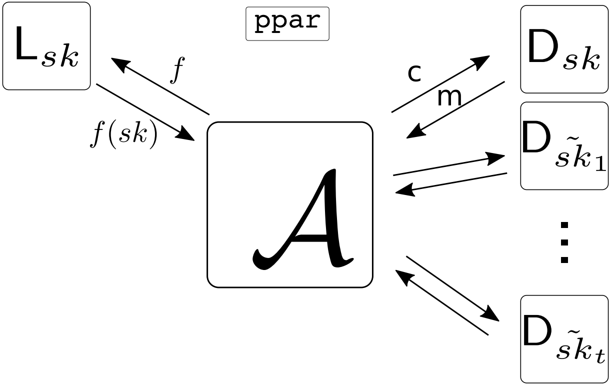
Section 2

BLT-CCA PKE

(t, ℓ)-BLT IND-CCA PKE:



(t, ℓ)-BLT IND-CCA PKE:



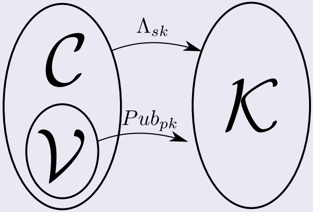
- $\mathcal{A}$ leaks before challenge ℓ bits;
- $\mathcal{A}$ instantiates before challenge t oracles

(for $\ell + t \leqslant |sk| - \omega(\log k)$)

The Scheme of [QL13]: Building Blocks

The Scheme of [QL13]: Building Blocks

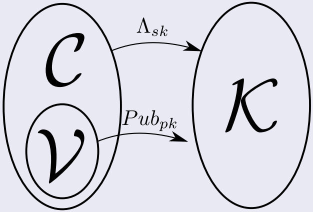
ϵ-Hash Proof System



- Complete: For $c \in \mathcal{V}$, $Pub_{pk}(c, w) = \Lambda_{sk}(c)$.
- Sound: For $c \in \mathcal{C} \setminus \mathcal{V}$,any $pk = \mu(sk)$: $\widetilde{\mathbb{H}}_\infty(K := \Lambda_{sk}(c)|pk) \geqslant -\log \epsilon$
- Set Membership Problem.

The Scheme of [QL13]: Building Blocks

ϵ-Hash Proof System



- Complete: For $c \in \mathcal{V}$,
  $Pub_{pk}(c, w) = \Lambda_{sk}(c)$.
- Sound: For $c \in \mathcal{C} \setminus \mathcal{V}$,any $pk = \mu(sk)$:
  $\widetilde{\mathbb{H}}_\infty(K := \Lambda_{sk}(c)|pk) \geqslant -\log \epsilon$
- Set Membership Problem.

δ-extractor

$\widetilde{\mathbb{H}}_\infty(\mathbf{X}|\mathbf{Z}) \geqslant \delta$, we have $(\mathbf{Z}, \mathbf{S}, \mathrm{Ext}(\mathbf{X}, \mathbf{S})) \approx (\mathbf{Z}, \mathbf{S}, \mathbf{U})$

The Scheme of [QL13]: Building Blocks, Pt.2

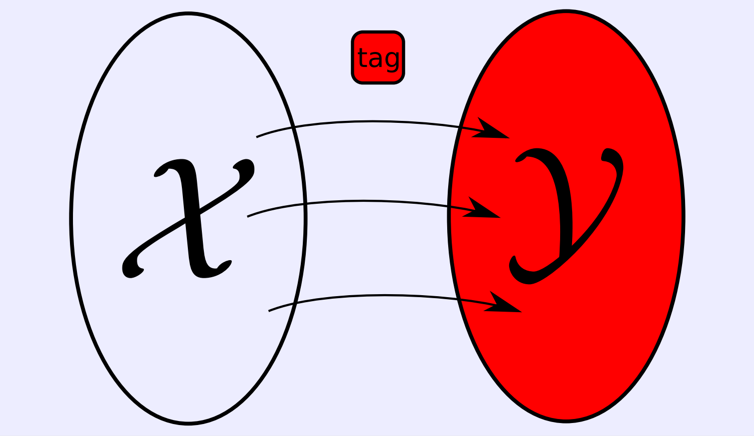
ℓ-(OT-)Lossy Filter

$$\mathsf{LF}_\phi : \mathcal{T} \times \mathcal{X} \to \mathcal{Y}$$

The Scheme of [QL13]: Building Blocks, Pt.2

ℓ-(OT-)Lossy Filter

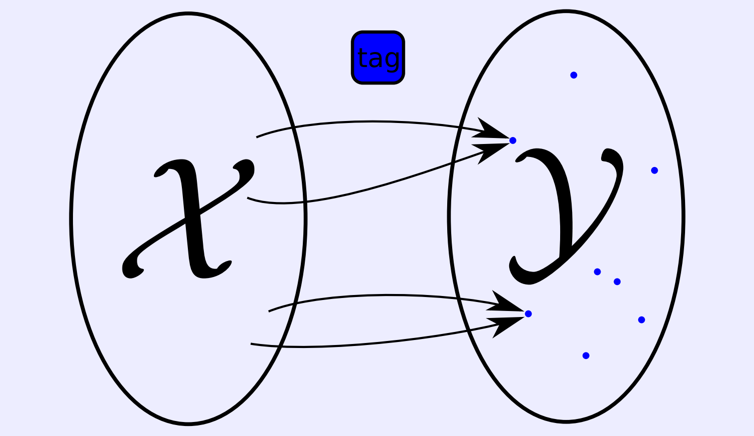
$$\mathsf{LF}_\phi : \mathcal{T} \times \mathcal{X} \to \mathcal{Y}$$



The Scheme of [QL13]: Building Blocks, Pt.2

ℓ-(OT-)Lossy Filter

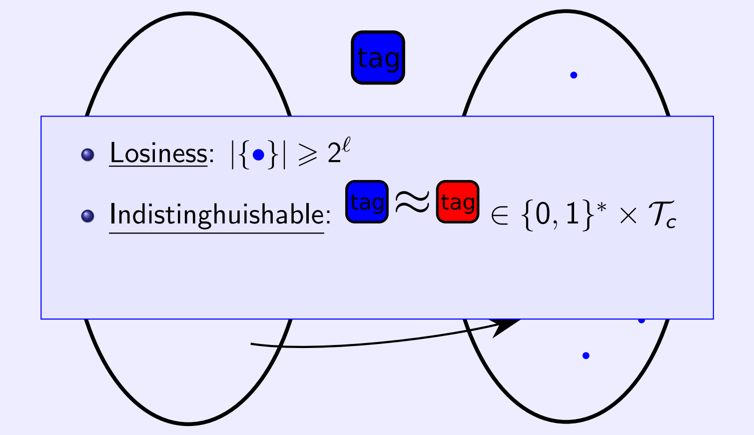
$$\mathsf{LF}_\phi : \mathcal{T} \times \mathcal{X} \to \mathcal{Y}$$



The Scheme of [QL13]: Building Blocks, Pt.2

ℓ-(OT-)Lossy Filter

$$\mathsf{LF}_{\phi} : \mathcal{T} \times \mathcal{X} \to \mathcal{Y}$$

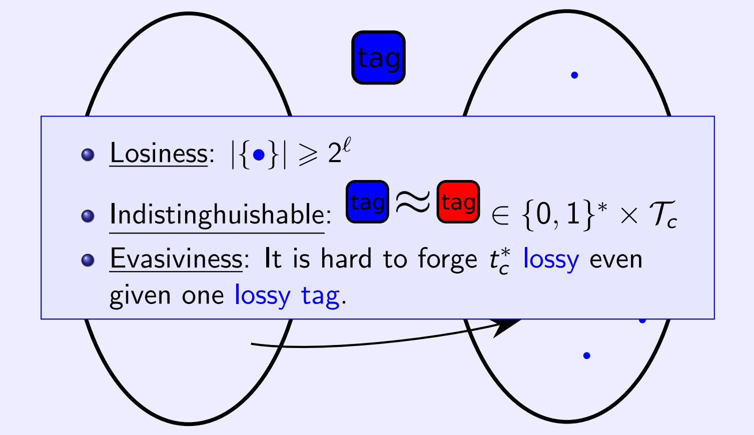


- Losiness: $|\{\bullet\}| \geqslant 2^{\ell}$
- Indistinghuishable: $\boxed{\text{tag}} \approx \boxed{\text{tag}} \in \{0,1\}^* \times \mathcal{T}_c$

The Scheme of [QL13]: Building Blocks, Pt.2

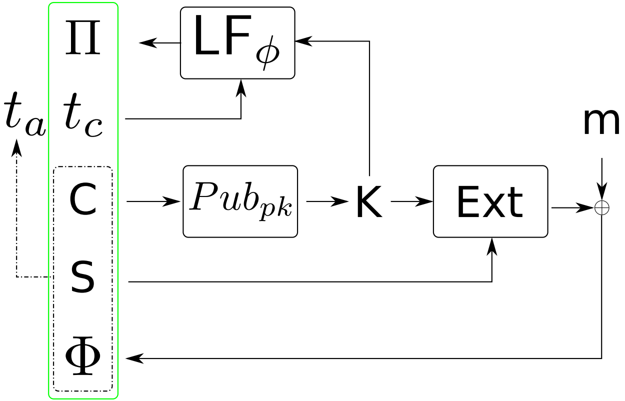
ℓ-(OT-)Lossy Filter

$$\mathsf{LF}_\phi : \mathcal{T} \times \mathcal{X} \to \mathcal{Y}$$

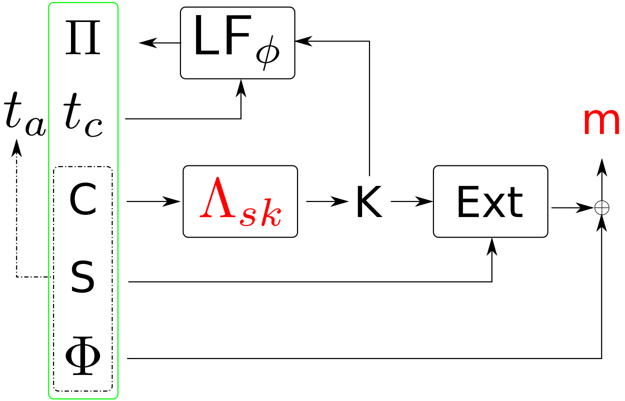


- Losiness: $|\{\bullet\}| \geqslant 2^\ell$
- Indistinghuishable: $\boxed{\text{tag}} \approx \boxed{\text{tag}} \in \{0,1\}^* \times \mathcal{T}_c$
- Evasiviness: It is hard to forge t_c^* lossy even given one lossy tag.

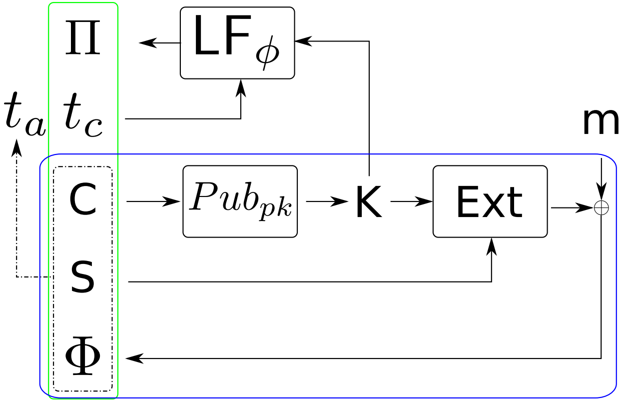
The Scheme of [QL13]:



The Scheme of [QL13]:

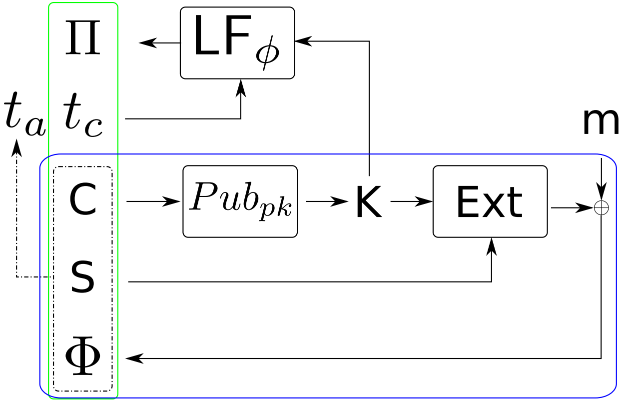


The Scheme of [QL13]:



$$\mathbb{H}_\infty(\mathbf{K}^*|\mathbf{pk}, \mathbf{C}^*, \mathbf{L}) \geqslant -\log \varepsilon - |\mathbf{L}|$$

The Scheme of [QL13]:



$$\mathbb{H}_\infty(\mathbf{K}^*|\mathbf{pk}, \mathbf{C}^*, \mathbf{L}) \geqslant -\log\varepsilon - |\mathbf{L}|$$

$$\mathbb{H}_\infty(\mathbf{K}^*|\mathbf{pk}, \mathbf{C}^*, \mathbf{L}, \mathbf{\Pi}) \geqslant -\log\varepsilon - |\mathbf{L}| - \ell$$

Reduce Tampering to Leakage

$$\boxed{\mathsf{Dec}_{T(sk)}} \approx \boxed{\mathcal{O}_{\boxed{\mathsf{aux}}}}$$

- $aux = L(sk)$
- Interact **unbounded** with $\mathsf{Dec}_{T(sk)}$, while *aux* small and **bounded**.

$$\boxed{\mathsf{Dec}_{T(sk)}} \approx \boxed{\quad \mathcal{O}_{\boxed{\mathsf{aux}}}}$$

$\boxed{\mathsf{Dec}_{T(sk)}} \approx \boxed{\mathcal{O}_{\text{aux}}}$

Let $\tilde{sk} = T(sk)$, <u>leak</u> $\mu(\tilde{sk})$

$$((C, S, \Phi),\ t_c, \Pi)$$

$\boxed{\mathsf{Dec}_{T(sk)}} \approx \boxed{\mathcal{O}_{\mathsf{aux}}}$

Let $\tilde{sk} = T(sk)$, <u>leak</u> $\mu(\tilde{sk})$

$$((C, S, \Phi),\ t_c, \Pi)$$

$C \in \mathcal{V}$

$(C, \mu(\tilde{sk}))$ fully define K. Execute Decryption.

$\boxed{\mathrm{Dec}_{T(sk)}} \approx \boxed{\mathcal{O}_{\text{aux}}}$

Let $\tilde{sk} = T(sk)$, <u>leak</u> $\mu(\tilde{sk})$

$$((C, S, \Phi),\ t_c, \Pi)$$

$C \in \mathcal{V}$

$(C, \mu(\tilde{sk}))$ fully define K. Execute Decryption.

$C \notin \mathcal{V}$

Depend on $\mathbb{H}_\infty(\Lambda_{\tilde{sk}}(C)|\textbf{View} = v)$.

- If big then output $\perp$;
- If small then <u>leak</u> $\tilde{sk}$ and run $\mathrm{Dec}_{\tilde{sk}}$.

$\boxed{\mathsf{Dec}_{T(sk)}} \approx \boxed{\mathcal{O}_{\mathsf{aux}}}$

Let $\tilde{sk} = T(sk)$, leak $\mu(\tilde{sk})$

$$((C, S, \Phi),\ t_c, \Pi)$$

$C \in \mathcal{V}$

$(C, \mu(\tilde{sk}))$ fully define K. Execute Decryption.

$C \notin \mathcal{V}$

Depend on $\mathbb{H}_\infty(\Lambda_{\tilde{sk}}(C)|\textbf{View} = v)$.

- If big then output $\perp$;
- If small then leak $\tilde{sk}$ and run $\mathsf{Dec}_{\tilde{sk}}$.

Yeah, but what do big and small even mean?

$\boxed{\mathsf{Dec}_{T(sk)}} \approx \boxed{\mathcal{O}_{\text{aux}}}$

Let $\tilde{sk} = T(sk)$, <u>leak</u> $\mu(\tilde{sk})$

$$((C, S, \Phi),\ t_c, \Pi)$$

$C \in \mathcal{V}$

$(C, \mu(\tilde{sk}))$ fully define K. Execute Decryption.

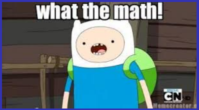
$C \notin \mathcal{V}$

Depend on $\mathbb{H}_\infty(\Lambda_{\tilde{sk}}(C) | \textbf{View} = v)$.

- If big then output $\perp$;
- If small then <u>leak</u> $\tilde{sk}$ and run $\mathsf{Dec}_{\tilde{sk}}$.

Yeah, but what do big and small even mean? I would tell you, if I had time..

Mathemagical!!

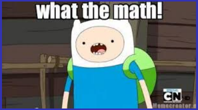


$\beta = s - \log \varepsilon$, $s = \log |SK|$
$\alpha = \log |PK|$

- We pay approx $\alpha + \beta$ bits of leakage for each tampering oracle.

$$t = \frac{s}{\alpha + \beta}$$

Mathemagical!!



$\beta = s - \log \varepsilon$, $s = \log |SK|$
$\alpha = \log |PK|$

- We pay approx $\alpha + \beta$ bits of leakage for each tampering oracle.

$$t = \frac{s}{\alpha + \beta}$$

We can instantiate the HPS using RSI.

Open Problems

- Is the tampering rate $O(1/k)$ inherent?
- A better Hash Proof System?

Open Problems

- Is the tampering rate $O(1/k)$ inherent?
- A better Hash Proof System?

Thank You!