Modeling Random Oracles under Unpredictable Queries

Pooya Farshim[1] Arno Mittelbach[2]

[1]ENS, CNRS & INRIA, PSL Research University, Paris, France
[2]TU Darmstadt, Germany

23rd Fast Software Encryption
Nordrhein-Westfalen

The Random-Oracle Model (ROM)

- Random oracles (ROs) model ideal hash functions [BR93].
  In the RO model:

  All parties have oracle access to a
  uniformly chosen **random function**.

- ROs enable the security proofs of a wide range of practical and strongly secure cryptosystems:

The Random-Oracle Model (ROM)

- Random oracles (ROs) model ideal hash functions [BR93].
  In the RO model:

  All parties have oracle access to a
  uniformly chosen **random function**.

- ROs enable the security proofs of a wide range of practical and strongly secure cryptosystems: encryption & signature schemes, key exchange, disk encryption, . . .

The Random-Oracle Model (ROM)

- Random oracles (ROs) model ideal hash functions [BR93].
  In the RO model:

  All parties have oracle access to a
  uniformly chosen **random function**.

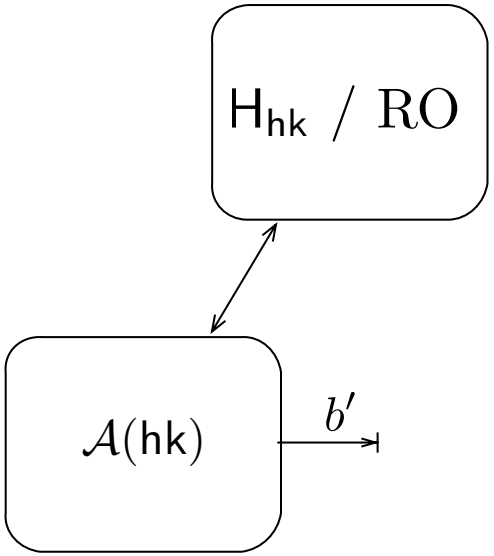
- ROs enable the security proofs of a wide range of practical and strongly secure cryptosystems: encryption & signature schemes, key exchange, disk encryption, ...
- Reliance on ROM, although practical, is also debatable:
  - There are **uninstantiable** ROM schemes [CGH98]:

    $\exists\, \mathsf{Enc}^{\mathcal{O}}$: $\mathsf{Enc}^{\mathcal{RO}}$ is secure but $\mathsf{Enc}^{\mathsf{H}}$ is insecure for **any** H.

  - Lack of a **definition** formalizing "RO-like behavior."

(Very) Naïve Attempt at Modeling ROs

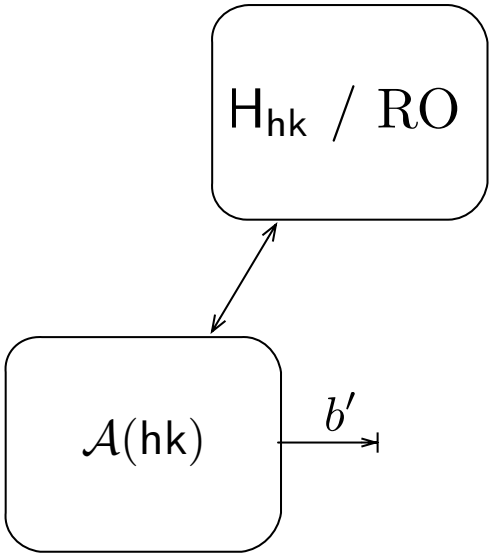
Call a hash function “IND-RO” if, over a random choice of hk:



$$\mathbf{Adv}^{\text{ind-ro}}_{\mathsf{H},\mathcal{A}}(\lambda) := 2 \cdot \Pr\left[b' = b\right] - 1$$

(Very) Naïve Attempt at Modeling ROs

Call a hash function "IND-RO" if, over a random choice of hk:



$$\mathbf{Adv}^{\text{ind-ro}}_{\mathsf{H},\mathcal{A}}(\lambda) := 2 \cdot \Pr\left[b' = b\right] - 1$$

Clearly uninstantiable:

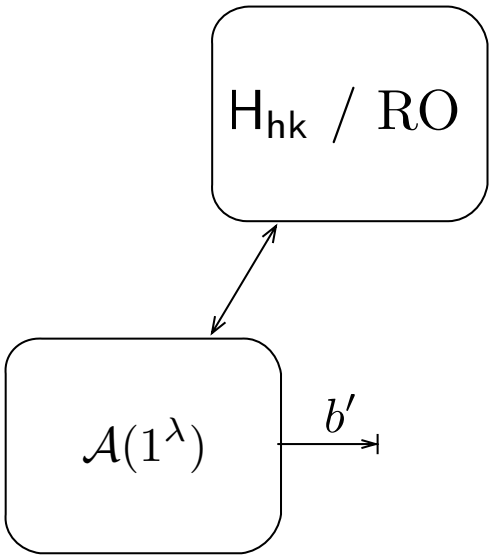
$\mathcal{A}(\mathsf{hk})$: compute $\mathsf{H}_{\mathsf{hk}}(0)$ and compare to the oracle's answer.

But observe:

The attack works because the "full" input $(\mathsf{hk}, 0)$ is known to $\mathcal{A}$.

Can We Fix the Naïve Model?

Let's hide hk.

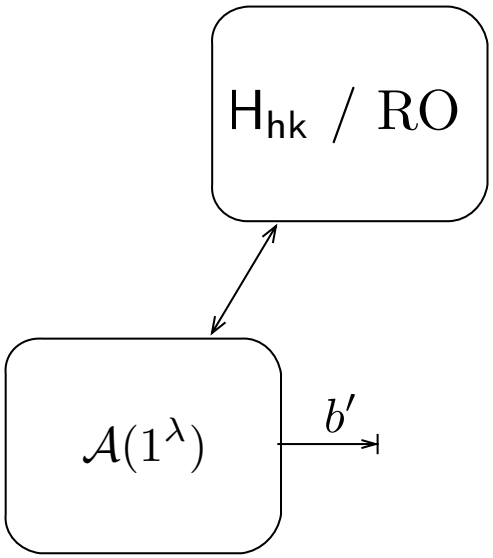


$$\mathbf{Adv}^{\mathrm{prf}}_{\mathsf{H},\mathcal{A}}(\lambda) := 2 \cdot \Pr\left[b' = b\right] - 1$$

We obtain PRF security: Not so useful in the context of hashing as hk is publicly available.

Can We Fix the Naïve Model?

Let's hide hk.



$$\mathbf{Adv}^{\mathrm{prf}}_{\mathsf{H},\mathcal{A}}(\lambda) := 2 \cdot \Pr\left[b' = b\right] - 1$$

We obtain PRF security: Not so useful in the context of hashing as hk is publicly available.

First idea:

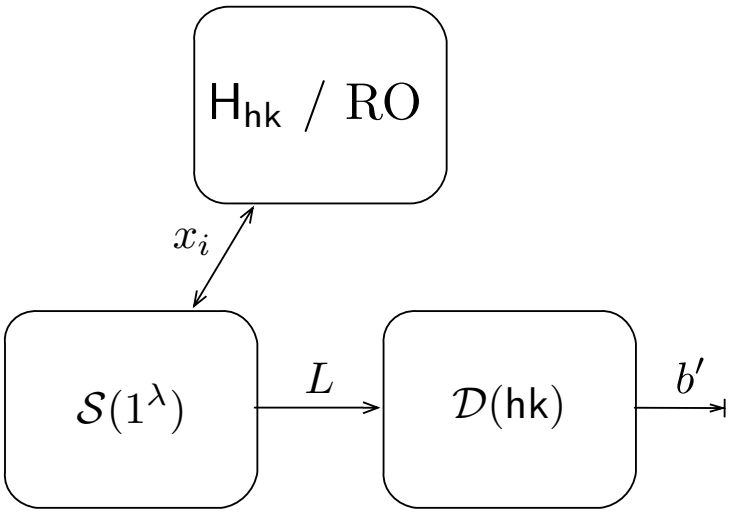
Split $\mathcal{A}$: one part gets hk and the other gets oracle access.

Modeling ROs via Split Adversaries

Call the two components of $\mathcal{A}$ the **source** $\mathcal{S}$ and the **distinguisher** $\mathcal{D}$:

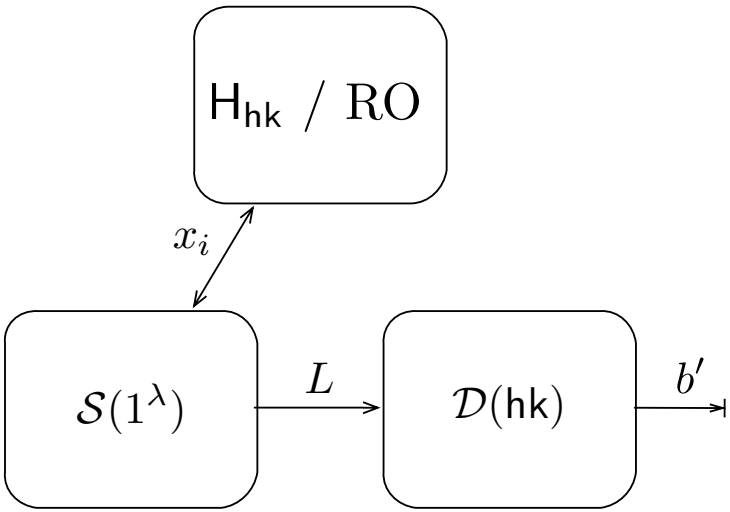
Modeling ROs via Split Adversaries

Call the two components of $\mathcal{A}$ the **source** $\mathcal{S}$ and the **distinguisher** $\mathcal{D}$:



Modeling ROs via Split Adversaries

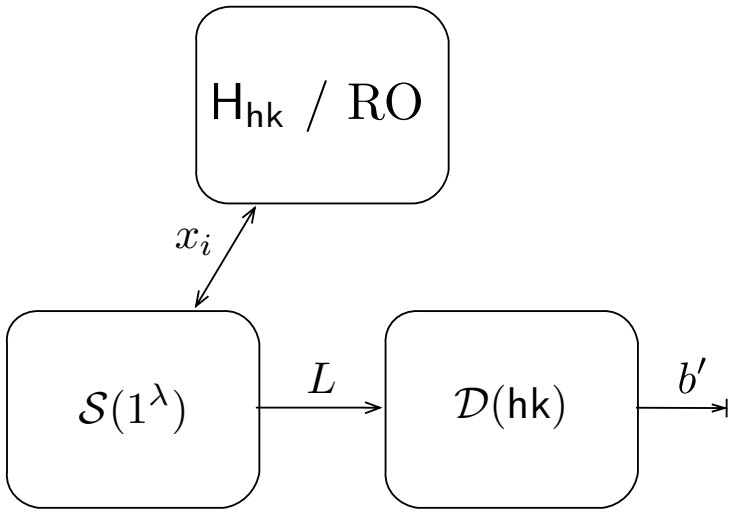
Call the two components of $\mathcal{A}$ the **source** $\mathcal{S}$ and the **distinguisher** $\mathcal{D}$:



$$\mathbf{Adv}^{\mathrm{uce}}_{\mathsf{H},\mathcal{S},\mathcal{D}}(\lambda) := 2 \cdot \Pr\left[b' = b\right] - 1$$

Modeling ROs via Split Adversaries

Call the two components of $\mathcal{A}$ the **source** $\mathcal{S}$ and the **distinguisher** $\mathcal{D}$:



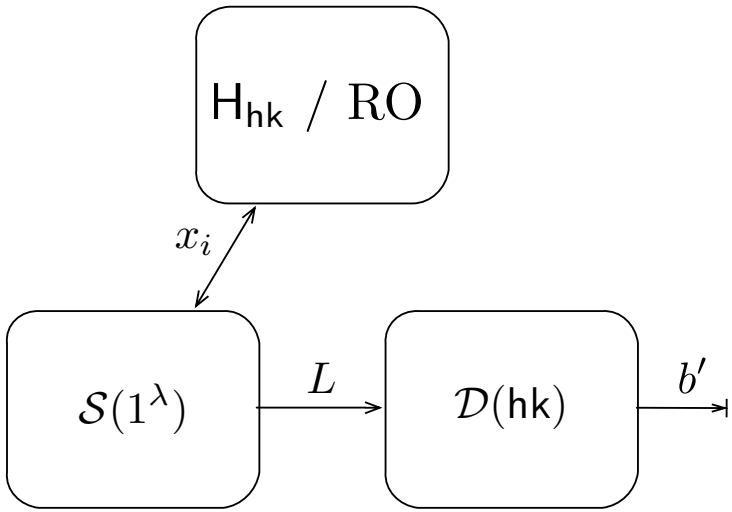
$$\mathbf{Adv}^{\mathrm{uce}}_{\mathsf{H},\mathcal{S},\mathcal{D}}(\lambda) := 2 \cdot \Pr\left[b' = b\right] - 1$$

Still uninstantiable:

$\mathcal{S}$ leaks oracle's response on 0 via L, and
$\mathcal{D}(\mathsf{hk})$ checks where it's coming from.

Modeling ROs via Split Adversaries

Call the two components of $\mathcal{A}$ the **source** $\mathcal{S}$ and the **distinguisher** $\mathcal{D}$:



$$\mathbf{Adv}^{\mathrm{uce}}_{\mathsf{H},\mathcal{S},\mathcal{D}}(\lambda) := 2 \cdot \Pr\left[b' = b\right] - 1$$

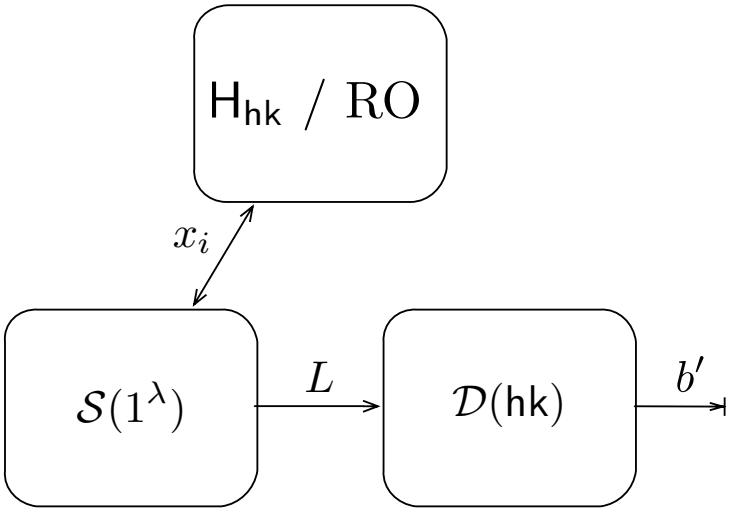
Still uninstantiable:

$\mathcal{S}$ leaks oracle's response on 0 via L, and
$\mathcal{D}(\mathsf{hk})$ checks where it's coming from.

Second idea:

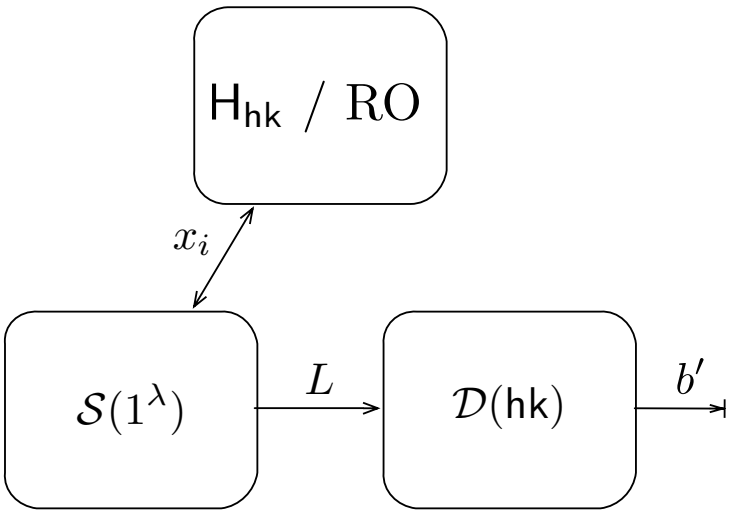
Restrict L: it must not leak any of $\mathcal{S}$'s queries.

Universal Computational Extractors (UCEs) [BHK13]

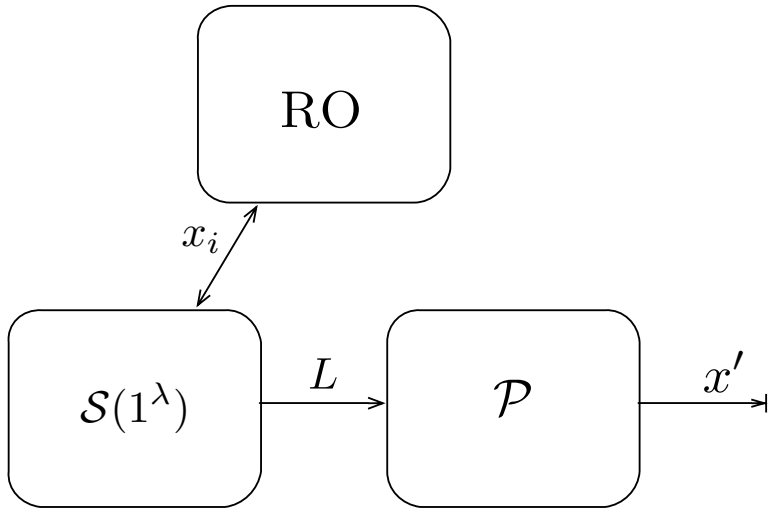


$$\mathbf{Adv}^{\mathrm{uce}}_{\mathsf{H},\mathcal{S},\mathcal{D}}(\lambda) := 2 \cdot \Pr\left[b = b'\right] - 1$$

Universal Computational Extractors (UCEs) [BHK13]

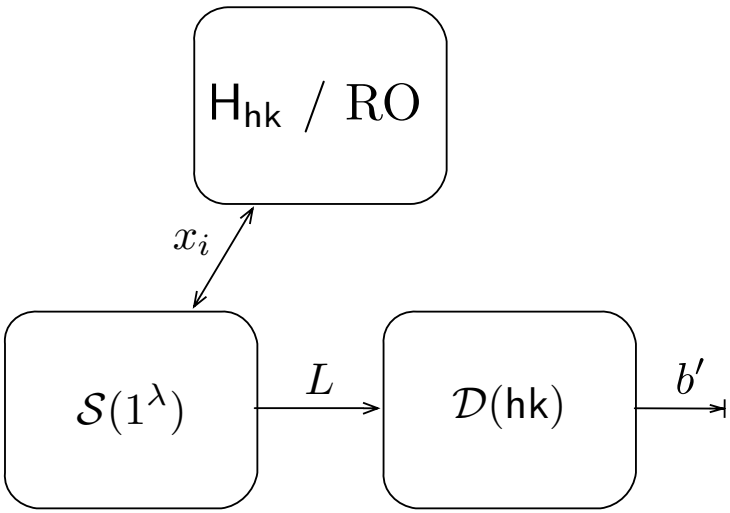


$$\mathbf{Adv}^{\mathrm{uce}}_{\mathsf{H},\mathcal{S},\mathcal{D}}(\lambda) := 2 \cdot \Pr\left[b = b'\right] - 1$$

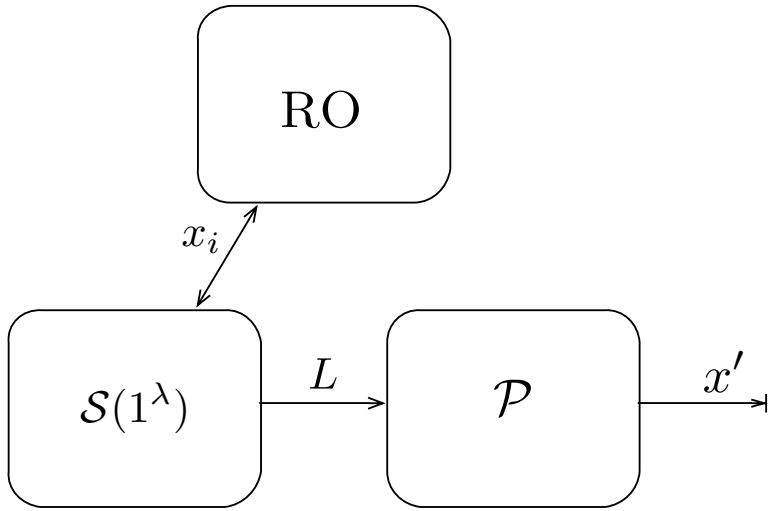


$$\mathbf{Adv}^{\mathrm{pred}}_{\mathcal{S},\mathcal{P}}(\lambda) := \Pr\left[x' \in \{x_1, \ldots, x_n\}\right]$$

Universal Computational Extractors (UCEs) [BHK13]



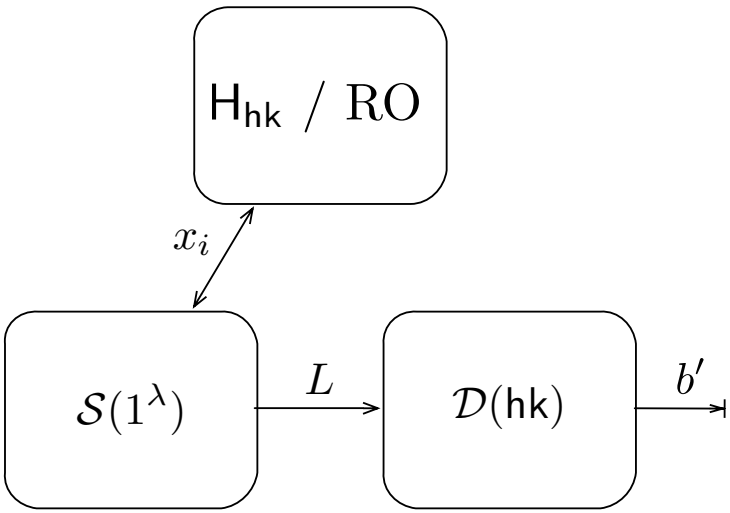
$$\mathbf{Adv}^{\mathrm{uce}}_{\mathsf{H},\mathcal{S},\mathcal{D}}(\lambda) := 2 \cdot \Pr\left[b = b'\right] - 1$$

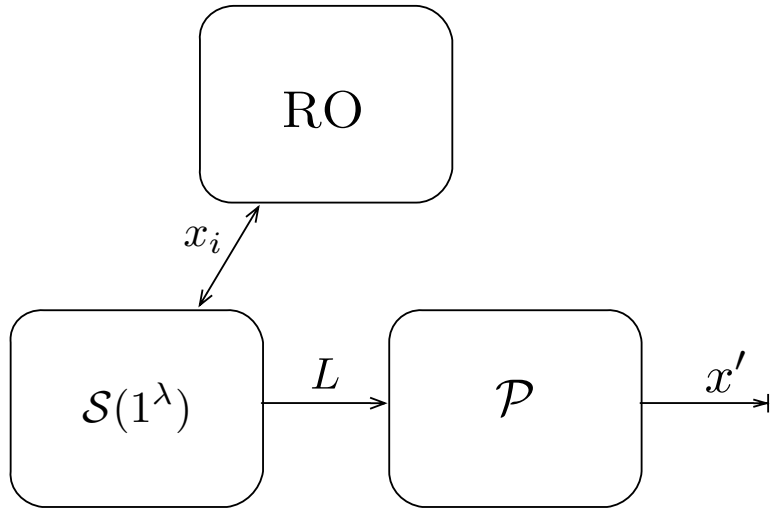


$$\mathbf{Adv}^{\mathrm{pred}}_{\mathcal{S},\mathcal{P}}(\lambda) := \Pr\left[x' \in \{x_1, \ldots, x_n\}\right]$$

Say $\mathcal{S}$ is **unpredictable** if $\mathbf{Adv}^{\mathrm{pred}}_{\mathcal{S},\mathcal{P}}(\lambda)$ is negl. for all efficient $\mathcal{P}$.

Universal Computational Extractors (UCEs) [BHK13]





$$\mathbf{Adv}^{\mathrm{uce}}_{\mathsf{H},\mathcal{S},\mathcal{D}}(\lambda) := 2 \cdot \Pr\left[b = b'\right] - 1 \qquad \mathbf{Adv}^{\mathrm{pred}}_{\mathcal{S},\mathcal{P}}(\lambda) := \Pr\left[x' \in \{x_1, \ldots, x_n\}\right]$$

Say $\mathcal{S}$ is **unpredictable** if $\mathbf{Adv}^{\mathrm{pred}}_{\mathcal{S},\mathcal{P}}(\lambda)$ is negl. for all efficient $\mathcal{P}$.

Say H is **UCE secure** if $\mathbf{Adv}^{\mathrm{uce}}_{\mathsf{H},\mathcal{S},\mathcal{D}}(\lambda)$ is negl. for any **unpredictable** $\mathcal{S}$.

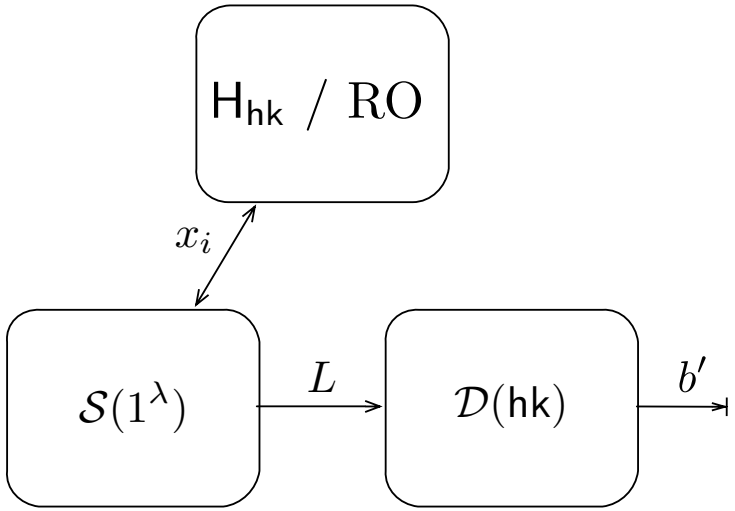
Applications of UCE [BHK13]

UCE-secure hash functions can instantiate the RO in:

- Deterministic public-key encryption (D-PKE)
- Message-locked encryption (MLE)
- Selective related-key and key-dependent message security
- Point-function obfuscation
- Proofs of storage
- Poly-many hard-core bits
- OAEP, garbling schemes, . . .

UCEs model many RO-like properties.

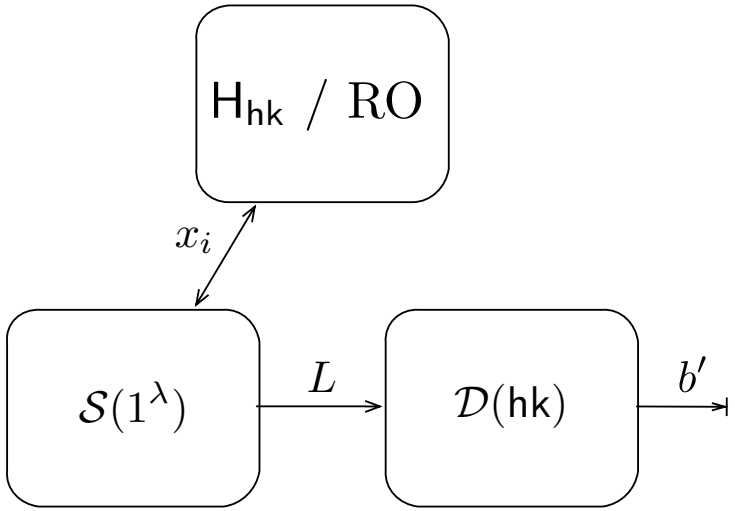
Shortcomings of UCEs



Three inter-related drawbacks:

- UCEs are not instantiable with unkeyed hash functions
- Queries are independent of the hash key
- The model does not allow for adaptive queries

Shortcomings of UCEs



Three inter-related drawbacks:

- UCEs are not instantiable with unkeyed hash functions
- Queries are independent of the hash key
- The model does not allow for adaptive queries

Can we overcome these?

Goal

Note also that:

Conceptually UCEs avoid the CGH attack by restricting queries to **high entropy** ones.

Goal

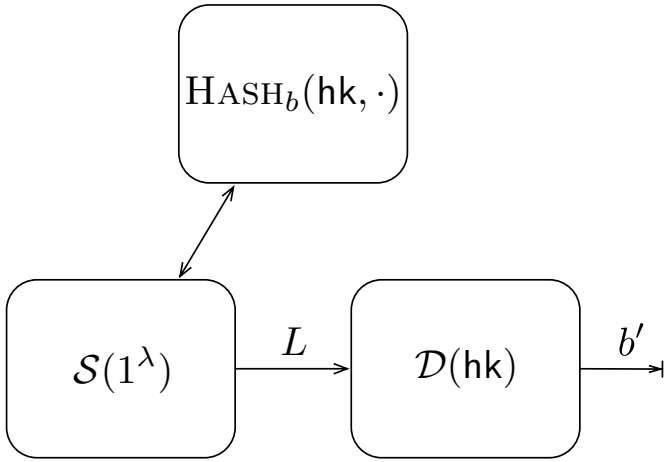
Note also that:

Conceptually UCEs avoid the CGH attack by
restricting queries to **high entropy** ones.

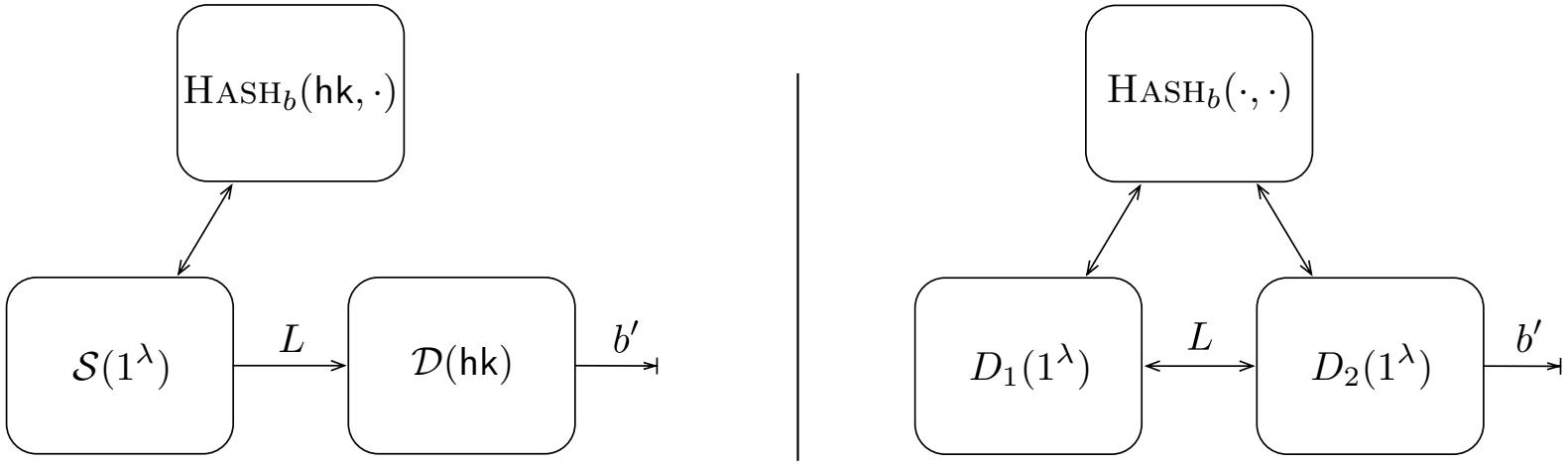
To what extent can we build on this view of UCEs to formulate

A more general framework modeling
a wider class of RO-like properties for hash functions.

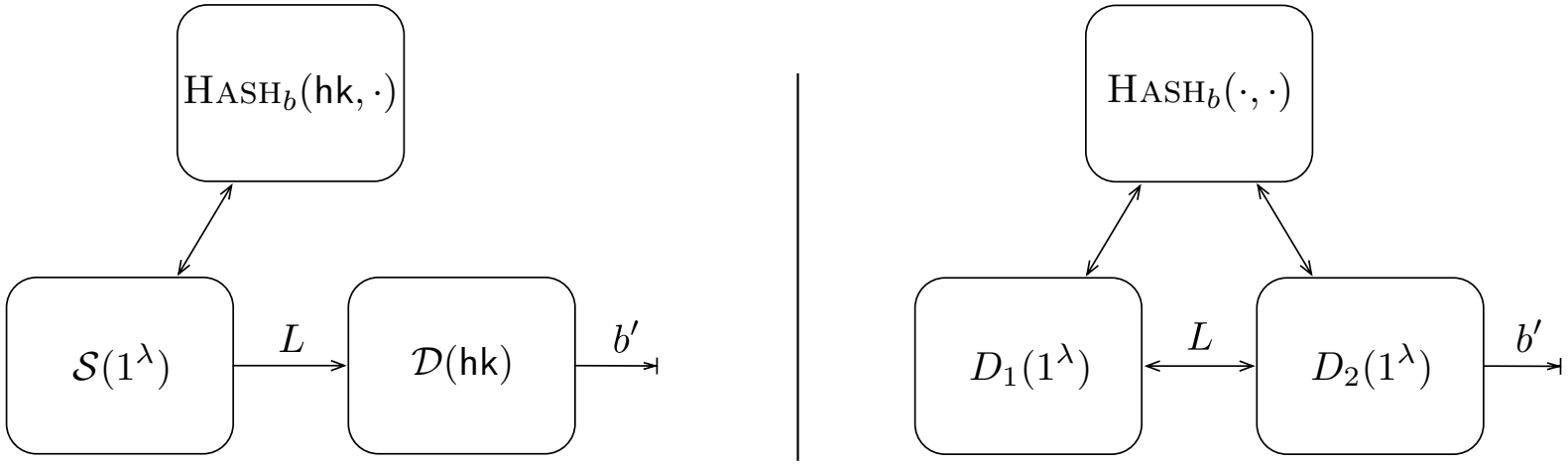
Interactive Computational Extractors (ICEs)



Interactive Computational Extractors (ICEs)

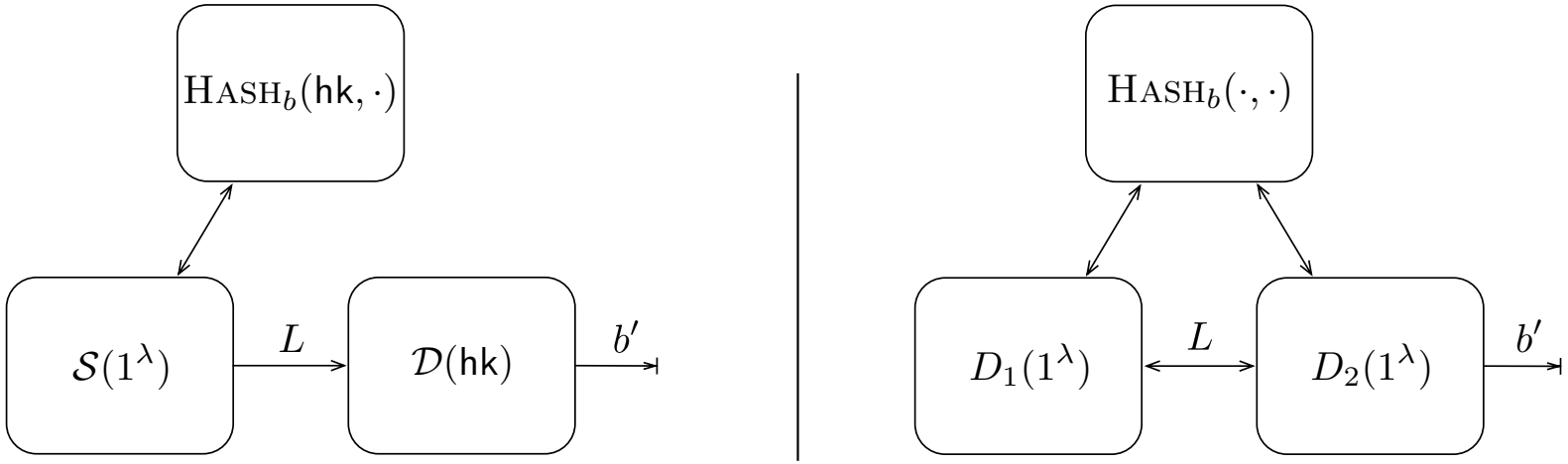


Interactive Computational Extractors (ICEs)



Unpredictability: Wrt. an RO, no $\mathcal{P}$ can predict a query of D_1 or D_2:

Interactive Computational Extractors (ICEs)



Unpredictability: Wrt. an RO, no $\mathcal{P}$ can predict a query of D_1 or D_2:

$$x \leftarrow\!\!\$\ \mathcal{P}(\mathsf{View}(D_i)) \quad \text{st.} \quad x \in \mathsf{Qrys}(D_1) \cup \mathsf{Qrys}(D_2)\ ,$$

where

$$\begin{aligned}\mathsf{View}(D_i) &:= \text{Everything that } D_i \text{ sees} \\ &= \mathsf{Coins}(D_i) + \mathsf{InLeakage}(D_i) + \text{Hash values}\ .\end{aligned}$$

Example Application: RKA Security

Recall the Black–Rogaway–Shrimpton [BRS03] encryption scheme:

$$\mathsf{Enc}^{\mathsf{H}}(K, M; R) := \big(R, \mathsf{H}(K|R) \oplus M\big)$$

Example Application: RKA Security

Recall the Black–Rogaway–Shrimpton [BRS03] encryption scheme:

$$\mathsf{Enc}^{\mathsf{H}}(K, M; R) := \big(R, \mathsf{H}(K|R) \oplus M\big)$$

This was shown to be KDM secure in the ROM, where

$$\text{Adversary gets to see: } \mathsf{Enc}(K, f(K)) \ .$$

Example Application: RKA Security

Recall the Black–Rogaway–Shrimpton [BRS03] encryption scheme:

$$\mathsf{Enc}^{\mathsf{H}}(K, M; R) := \big(R, \mathsf{H}(K|R) \oplus M\big)$$

This was shown to be KDM secure in the ROM, where

$$\text{Adversary gets to see: } \mathsf{Enc}(K, f(K)) \ .$$

We establish its RKA (and KDM) security without ROs, where

$$\text{Adversary gets to see: } \mathsf{Enc}(f(K), M) \ .$$

Example Application: RKA Security

Recall the Black–Rogaway–Shrimpton [BRS03] encryption scheme:

$$\mathsf{Enc}^{\mathsf{H}}(K, M; R) := \big(R, \mathsf{H}(K|R) \oplus M\big)$$

This was shown to be KDM secure in the ROM, where

$$\text{Adversary gets to see: } \mathsf{Enc}(K, f(K)) \ .$$

We establish its RKA (and KDM) security without ROs, where

$$\text{Adversary gets to see: } \mathsf{Enc}(f(K), M) \ .$$

Theorem

The BRS scheme is RKA secure against **split** functions:

$$f : K_1|K_2 \mapsto f_1(K_1)|f_2(K_2)$$

if H is ICE secure. (As we'll see, this implies RKA security in ROM.)

Example Application: RKA Security

Recall the Black–Rogaway–Shrimpton [BRS03] encryption scheme:

$$\mathsf{Enc}^{\mathsf{H}}(K, M; R) := \big(R, \mathsf{H}(K|R) \oplus M\big)$$

This was shown to be KDM secure in the ROM, where

$$\text{Adversary gets to see: } \mathsf{Enc}(K, f(K)) \ .$$

We establish its RKA (and KDM) security without ROs, where

$$\text{Adversary gets to see: } \mathsf{Enc}(f(K), M) \ .$$

Theorem

The BRS scheme is RKA secure against **split** functions:

$$f : K_1|K_2 \mapsto f_1(K_1)|f_2(K_2)$$

if H is ICE secure. (As we'll see, this implies RKA security in ROM.)

Full RKA: Via a new ICE notion (see upcoming full version).

Other Applications

- All applications of UCEs:
  - Non-adaptive RKA/KDM security
  - Point function obfuscation
  - Message-locked encryption, . . .

Other Applications

- All applications of UCEs:
  - Non-adaptive RKA/KDM security
  - Point function obfuscation
  - Message-locked encryption, . . .
- Semi-adaptive split KDM security of the BRS scheme
- Correlated-input hashing
- Foundational primitives:
  - (weak) PRFs,
  - Randomness extractors,
  - One-way security for polynomial regularity, . . .

Feasibility I: VIL-ROM

Why consider this question at all?

Feasibility I: VIL-ROM

Why consider this question at all?

- No generic attacks: the ICE model is structurally sound
- Enables a layered approach to security analysis: one first proves security under ICEs, and then applies ROM feasibility.

Feasibility I: VIL-ROM

Why consider this question at all?

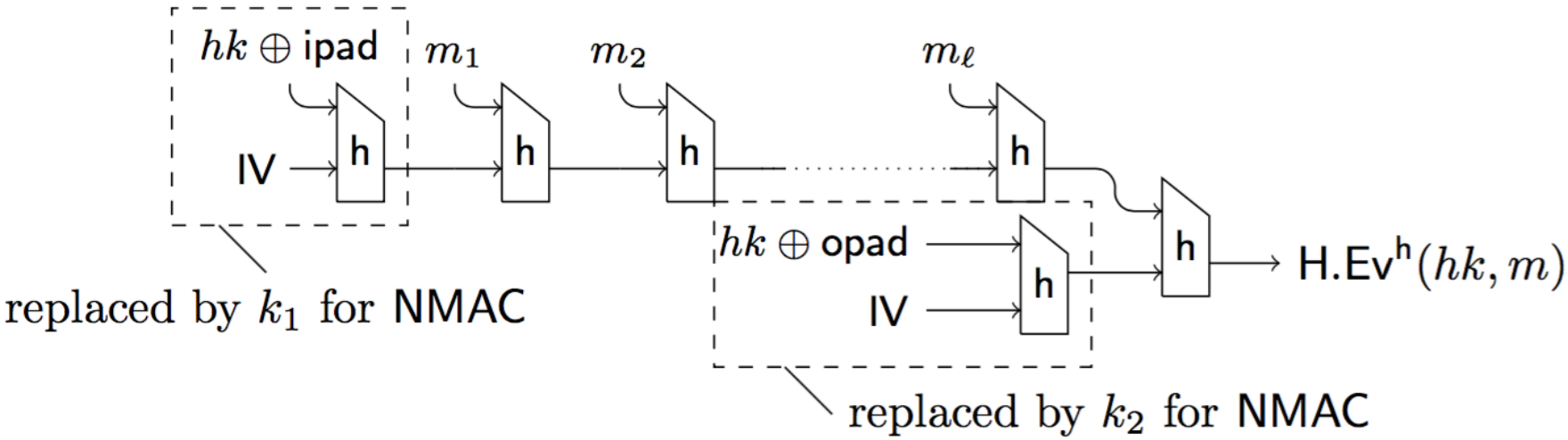
- No generic attacks: the ICE model is structurally sound
- Enables a layered approach to security analysis: one first proves security under ICEs, and then applies ROM feasibility.

Theorem

$\mathsf{H}^{\mathcal{RO}}(\mathsf{hk}, M) := \mathcal{RO}(\mathsf{hk}|M)$ is ICE secure against computationally unpredictable (D_1, D_2).

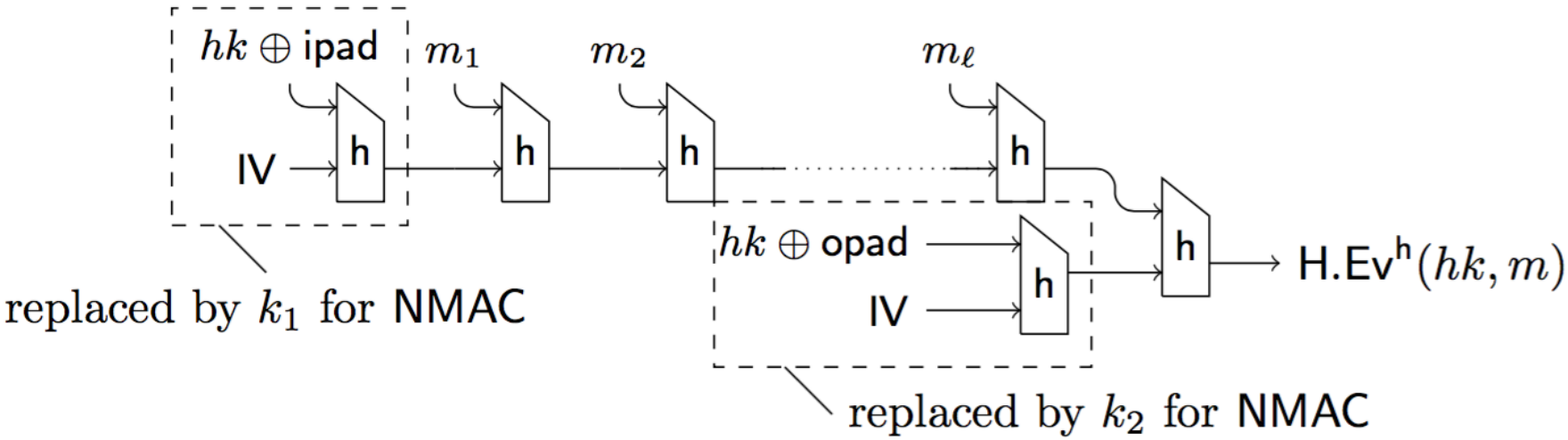
Feasibility II: FIL-ROM

Let's look at HMAC/NMAC:



Feasibility II: FIL-ROM

Let's look at HMAC/NMAC:

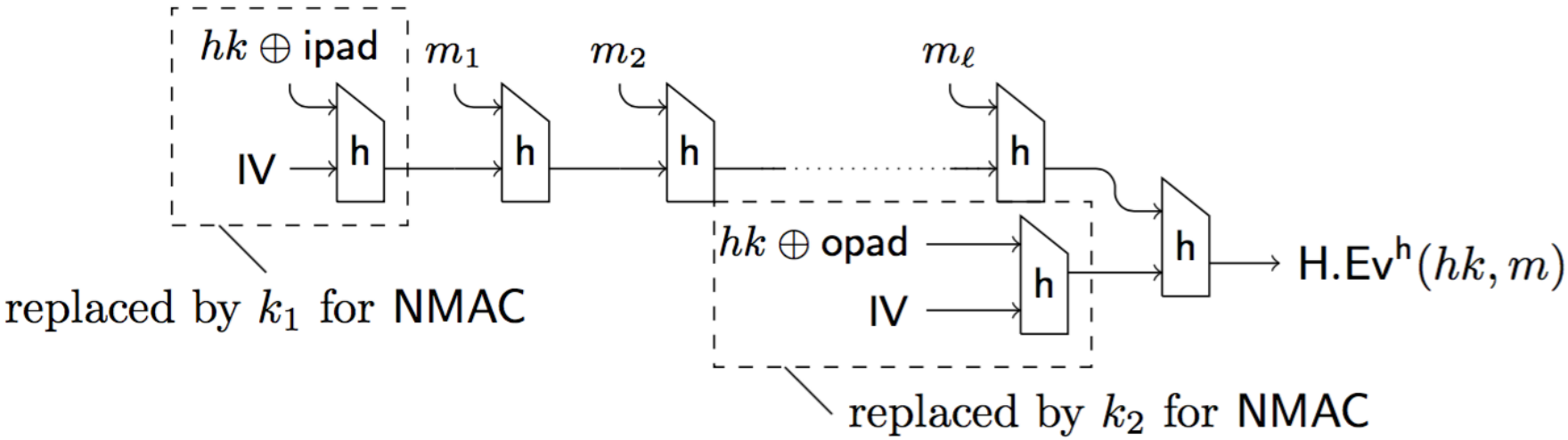


Claim: HMAC is ICE secure in the FIL-ROM.

Proof: HMAC is indifferentiable from RO, and RO is ICE secure. □

Feasibility II: FIL-ROM

Let's look at HMAC/NMAC:

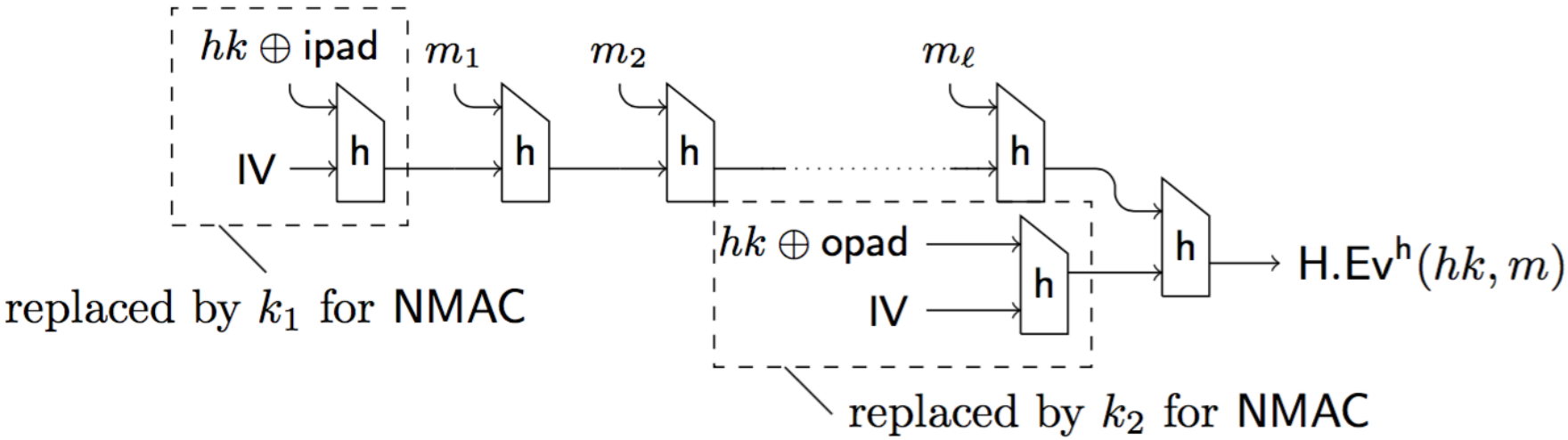


Claim: HMAC is ICE secure in the FIL-ROM.
Proof: HMAC is indifferentiable from RO, and RO is ICE secure. □

Not true! ICE is a multi-staged and indifferentiability can fail in these settings [RSS11].

Feasibility II: FIL-ROM

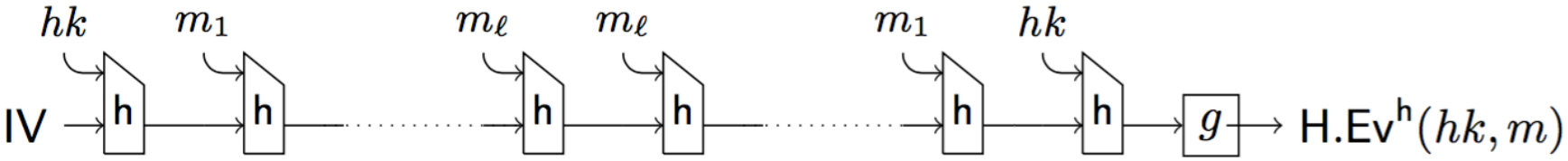
Let's look at HMAC/NMAC:



Claim: HMAC is ICE secure in the FIL-ROM.
Proof: HMAC is indifferentiable from RO, and RO is ICE secure. □

Not true! ICE is a multi-staged and indifferentiability can fail in these settings [RSS11]. Indeed, there are ICE attacks on HMAC via chain completion.

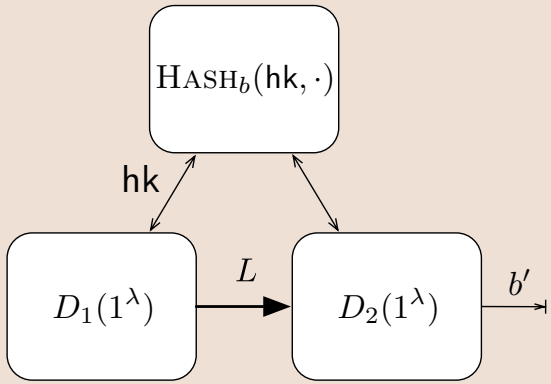
Feasibility II: Zipper Hash



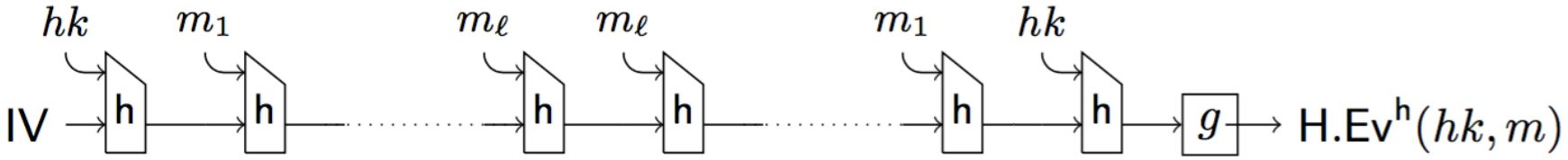
Feasibility II: Zipper Hash

Theorem

The Keyed & Chopped Zipper Hash above is ICE secure against

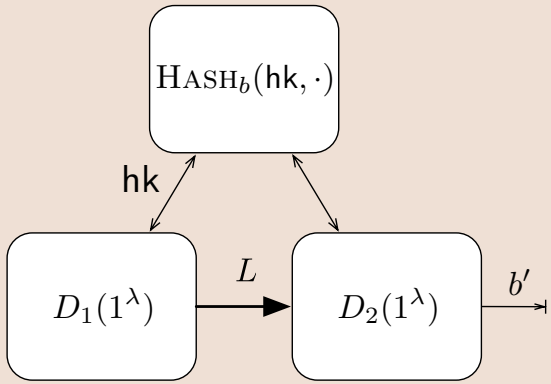


Feasibility II: Zipper Hash



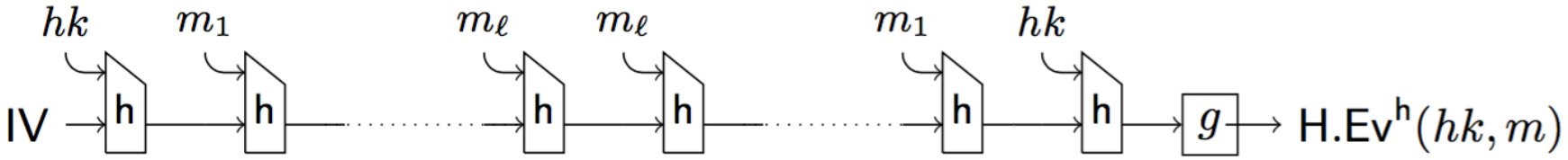
Theorem

The Keyed & Chopped Zipper Hash above is ICE secure against



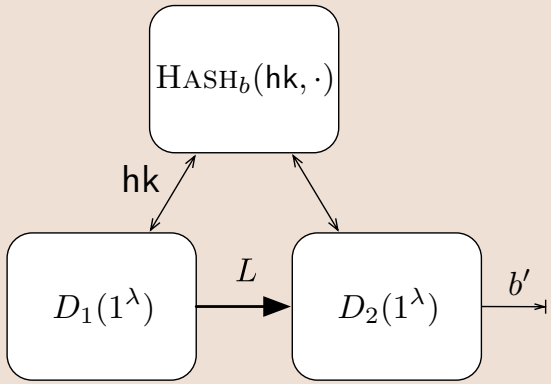
- Sufficient for many applications, including split RKA security.

Feasibility II: Zipper Hash



Theorem

The Keyed & Chopped Zipper Hash above is ICE secure against



- Sufficient for many applications, including split RKA security.
- Shows multi-pass hash functions can provide extra security over their single-pass counterparts.

What about Full ICE in FIL-ROM?

Consider a message

$$M := \underbrace{[m_1, m_2]}_{M_1} \mid \underbrace{[m_3, m_4]}_{M_2} \mid \cdots \mid \underbrace{[m_{2n-1}, m_{2n}]}_{M_n}$$

What about Full ICE in FIL-ROM?

Consider a message

$$M := \underbrace{[m_1, m_2]}_{M_1} \mid \underbrace{[m_3, m_4]}_{M_2} \mid \cdots \mid \underbrace{[m_{2n-1}, m_{2n}]}_{M_n}$$

Construct all half-block pairs:

$$\begin{aligned}\overline{M} := &[m_1, m_2] \mid [m_1, m_3] \mid \cdots \mid [m_1, m_{2n}] \mid \\ &[m_2, m_3] \mid [m_2, m_4] \mid \cdots \mid [m_2, m_{2n}] \mid \\ &[m_3, m_4] \mid \cdots \mid [m_{2n-1}, m_{2n}]\end{aligned}$$

What about Full ICE in FIL-ROM?

Consider a message

$$M := \underbrace{[m_1, m_2]}_{M_1} \mid \underbrace{[m_3, m_4]}_{M_2} \mid \cdots \mid \underbrace{[m_{2n-1}, m_{2n}]}_{M_n}$$

Construct all half-block pairs:

$$\begin{aligned}\overline{M} :=& [m_1, m_2] \mid [m_1, m_3] \mid \cdots \mid [m_1, m_{2n}] \mid \\ & [m_2, m_3] \mid [m_2, m_4] \mid \cdots \mid [m_2, m_{2n}] \mid \\ & [m_3, m_4] \mid \quad \cdots \quad \mid [m_{2n-1}, m_{2n}]\end{aligned}$$

Now define:

$$\mathsf{MixHash}^{\mathsf{h}}(\mathsf{hk}, M) := \mathsf{HMAC}^{\mathsf{h}}(0, \overline{\mathsf{hk} | M}) \ .$$

What about Full ICE in FIL-ROM?

Consider a message

$$M := \underbrace{[m_1, m_2]}_{M_1} \mid \underbrace{[m_3, m_4]}_{M_2} \mid \cdots \mid \underbrace{[m_{2n-1}, m_{2n}]}_{M_n}$$

Construct all half-block pairs:

$$\begin{aligned}\overline{M} :=& [m_1, m_2] \mid [m_1, m_3] \mid \cdots \mid [m_1, m_{2n}] \mid \\ & [m_2, m_3] \mid [m_2, m_4] \mid \cdots \mid [m_2, m_{2n}] \mid \\ & [m_3, m_4] \mid \quad \cdots \quad \mid [m_{2n-1}, m_{2n}]\end{aligned}$$

Now define:

$$\mathsf{MixHash}^{\mathsf{h}}(\mathsf{hk}, M) := \mathsf{HMAC}^{\mathsf{h}}(0, \overline{\mathsf{hk}|M}) \ .$$

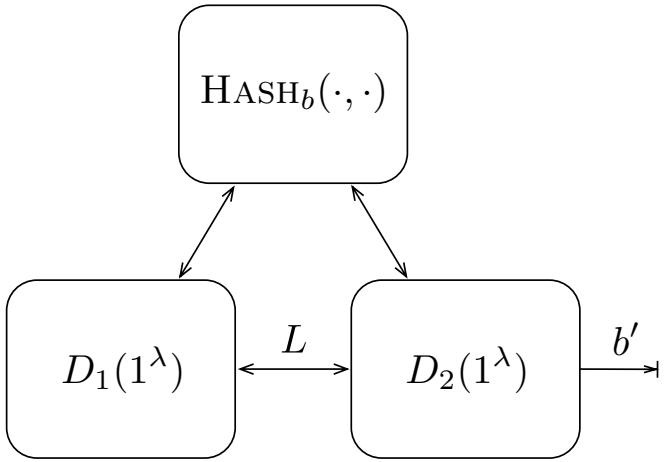
Conjecture

MixHash is fully ICE secure in the FIL-ROM.

Final Thoughts

What this talk was about:

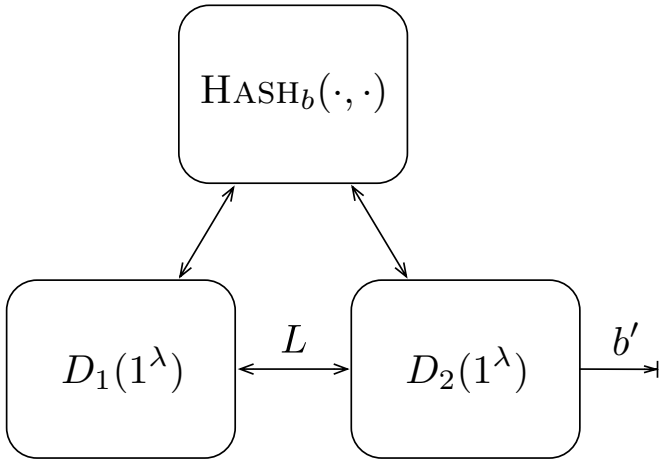
A new security model capturing many RO-like properties.



Final Thoughts

What this talk was about:

A new security model capturing many RO-like properties.



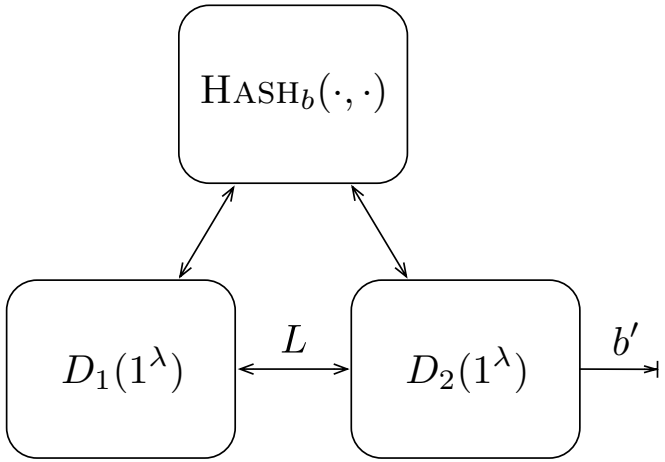
Future directions:

- What's the most general model for RO-like behavior?
- In particular, are there extensions that get us up to full KDM security?
- Weakening assumptions: domain/range extenders for ICEs.
- Are ICEs instantiable in the standard-model?

Final Thoughts

What this talk was about:

A new security model capturing many RO-like properties.



Future directions:

- What's the most general model for RO-like behavior?
- In particular, are there extensions that get us up to full KDM security?
- Weakening assumptions: domain/range extenders for ICEs.
- Are ICEs instantiable in the standard-model?

Thank you.