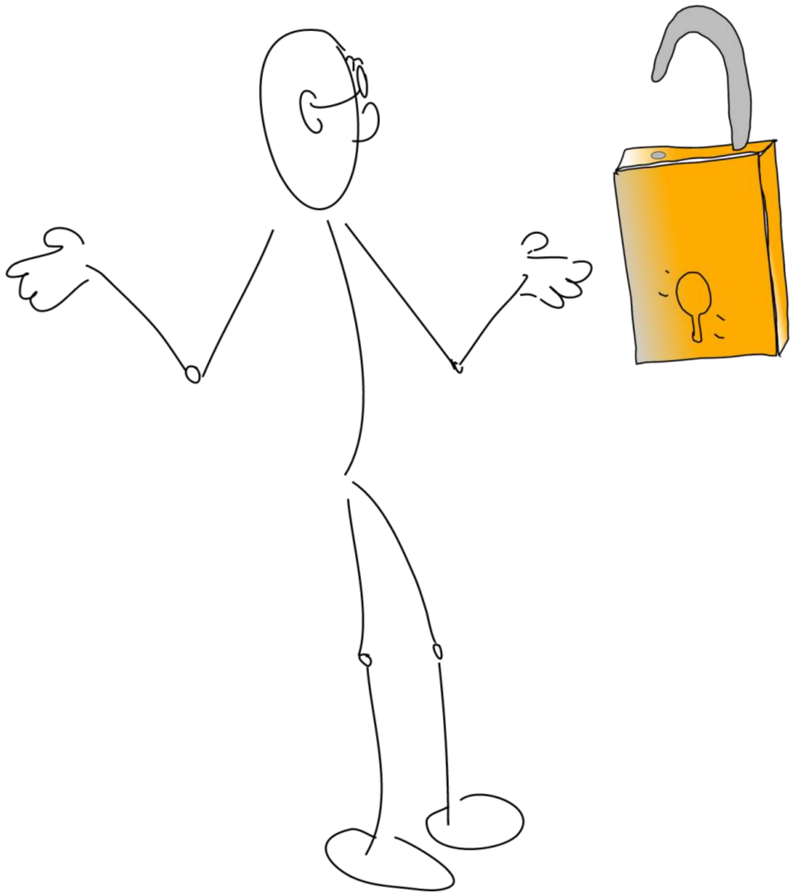
Salvaging Indifferentiability in a Multi-stage Setting

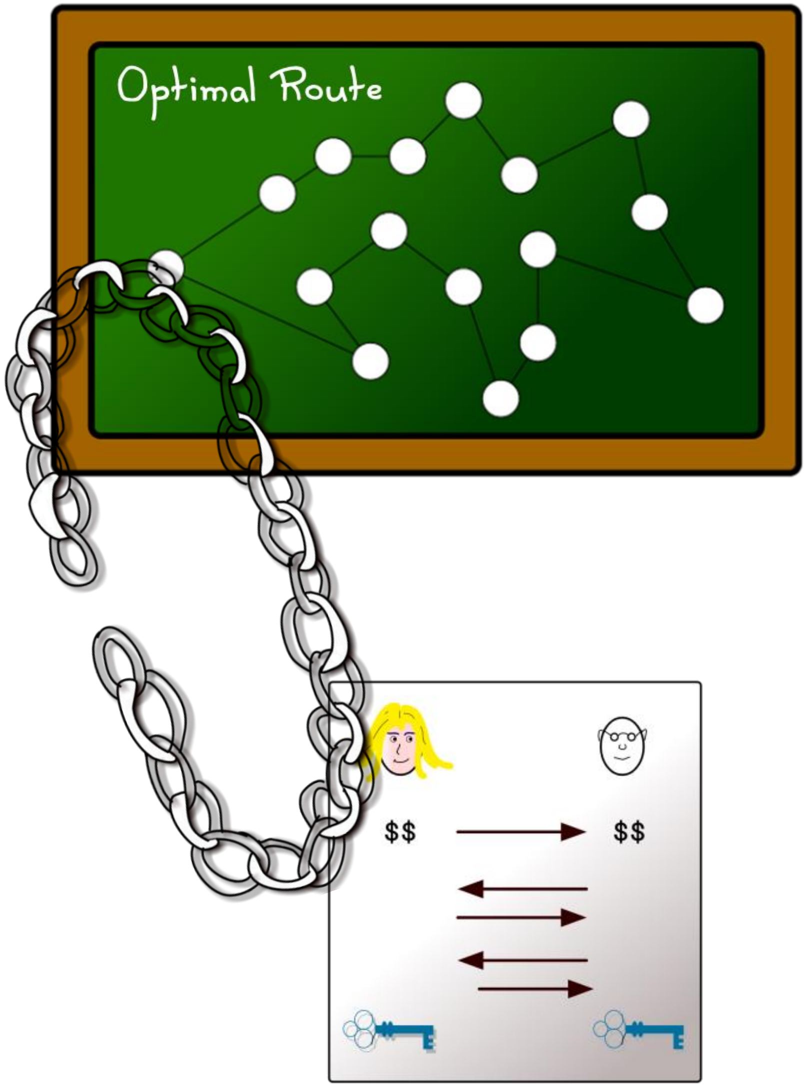
TECHNISCHE UNIVERSITÄT DARMSTADT

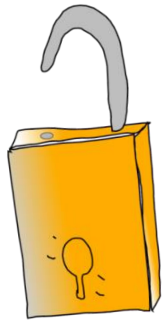
Cryptoplexity
Cryptography & Complexity Theory
Technische Universität Darmstadt
www.cryptoplexity.de

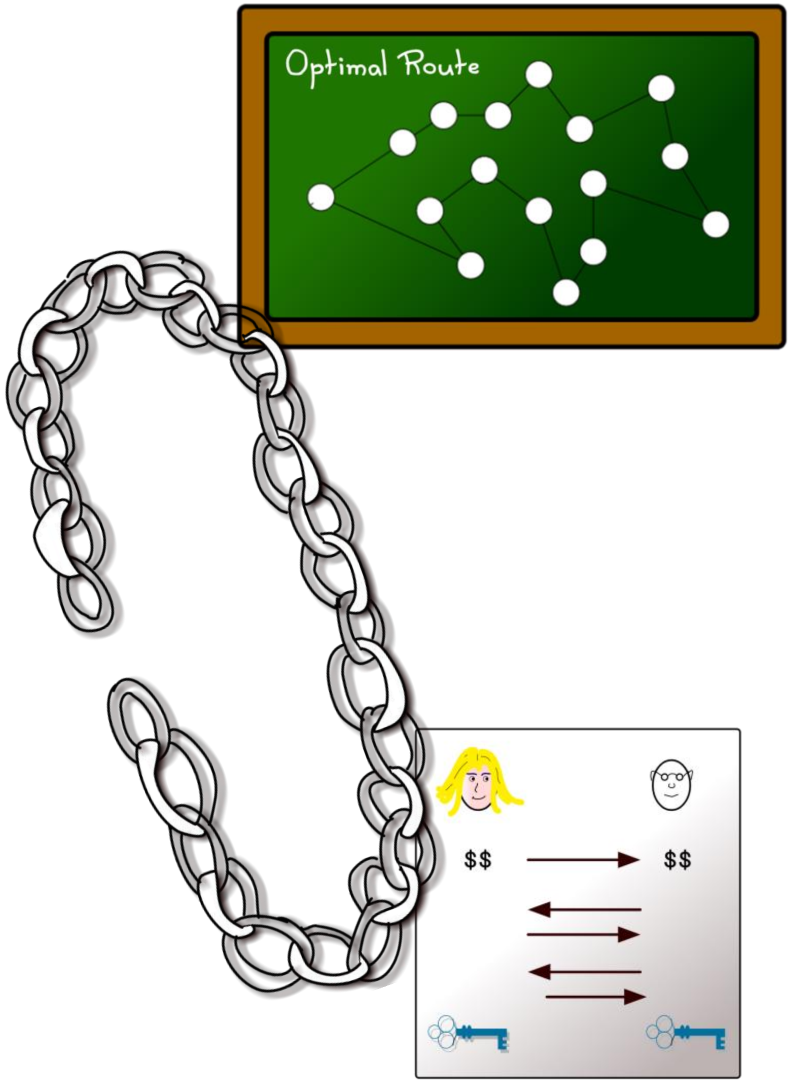
EUROCRYPT 2014, May 15th

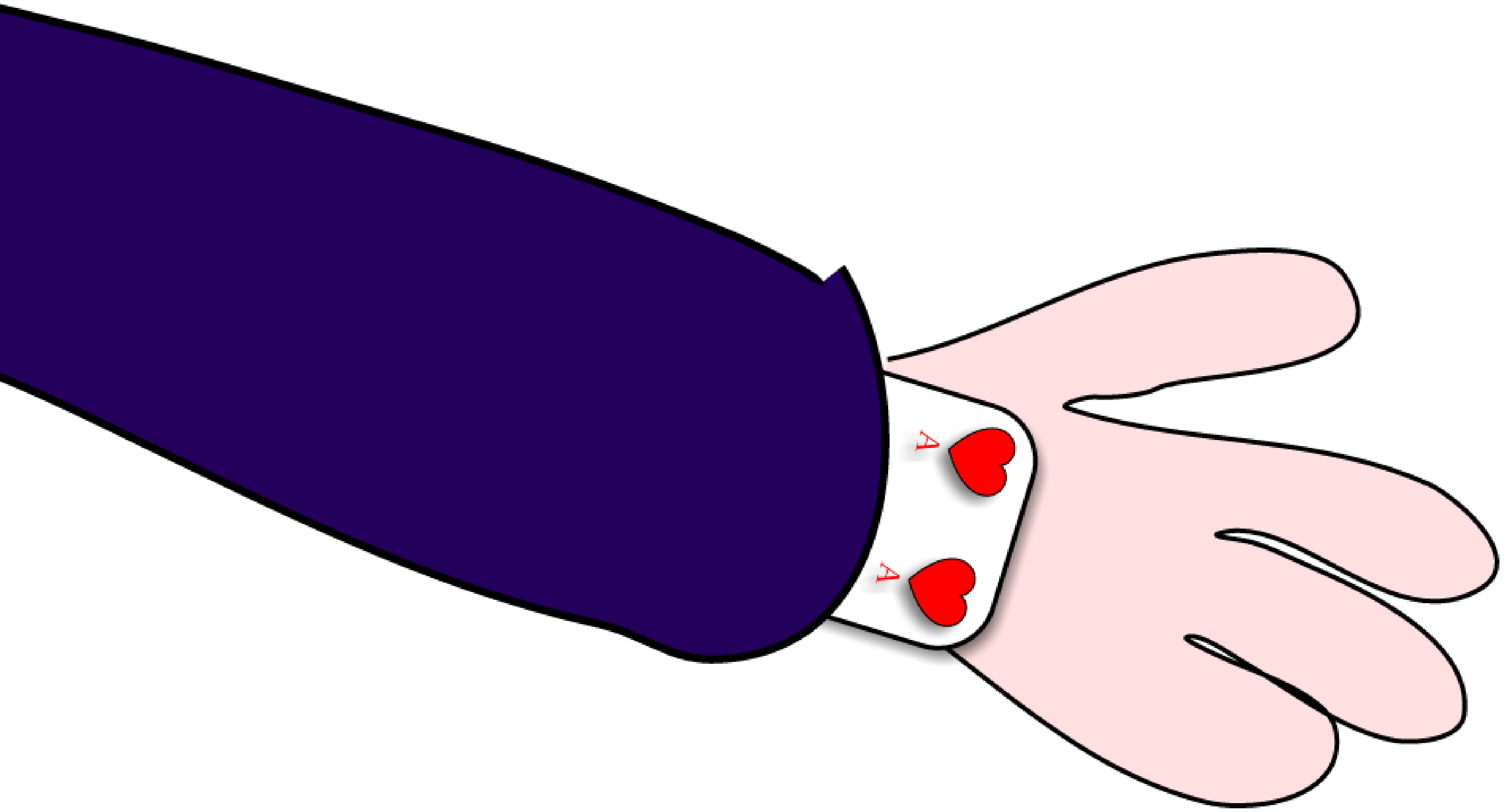
Arno Mittelbach

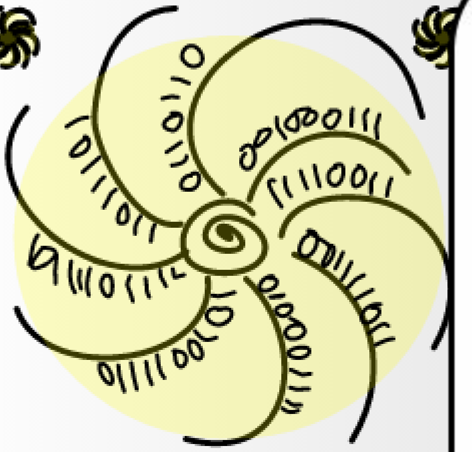



Optimal Route
$$
$$




Optimal Route
$$
$$


A
A

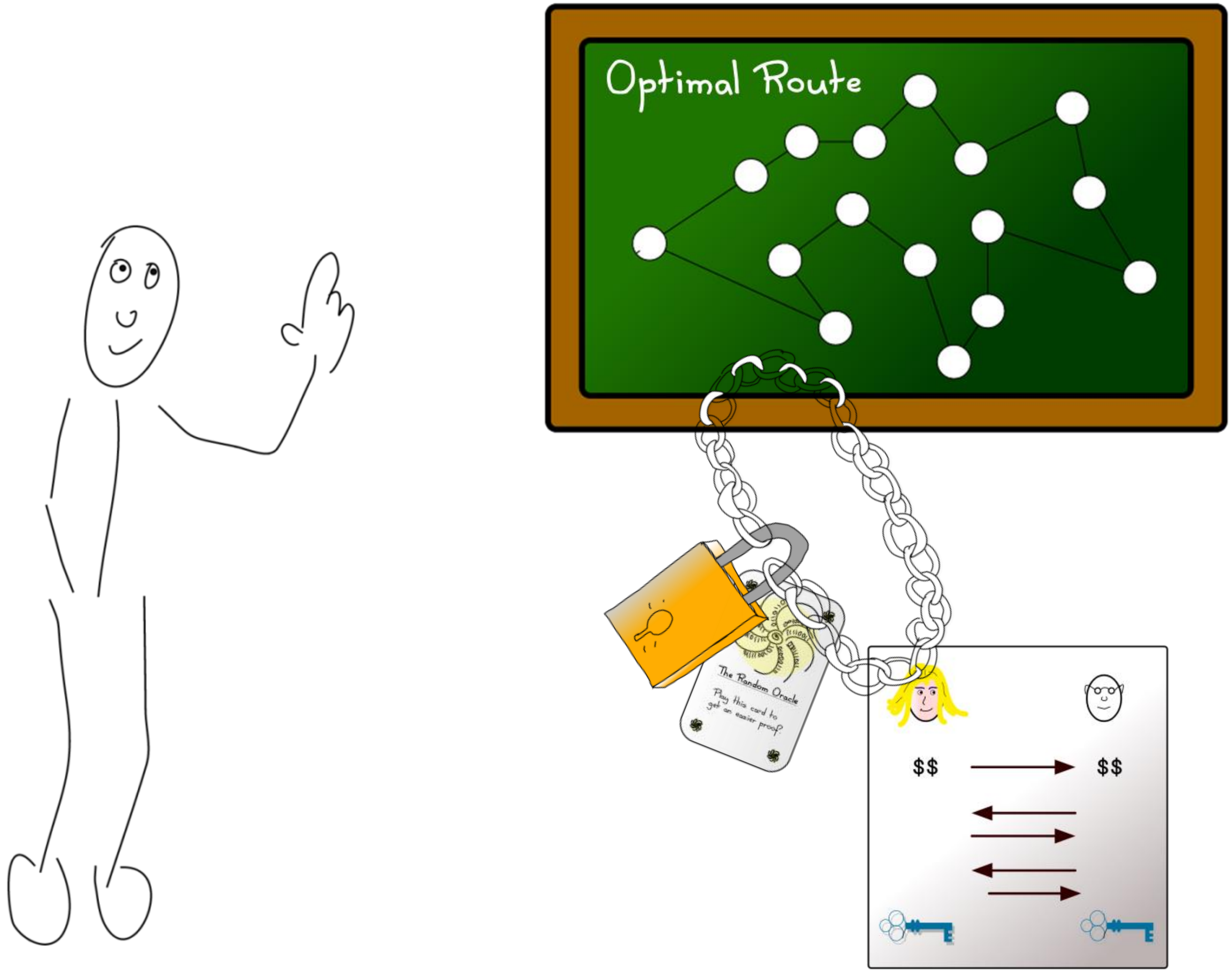


The Random Oracle

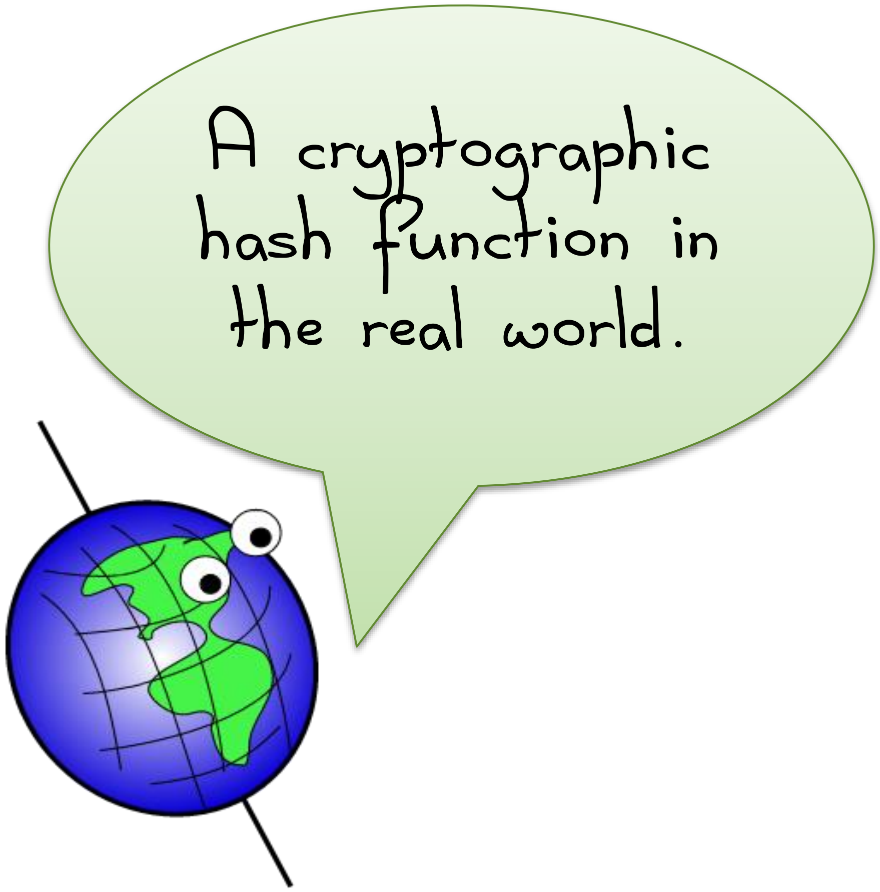
Play this card to get an easier proof.

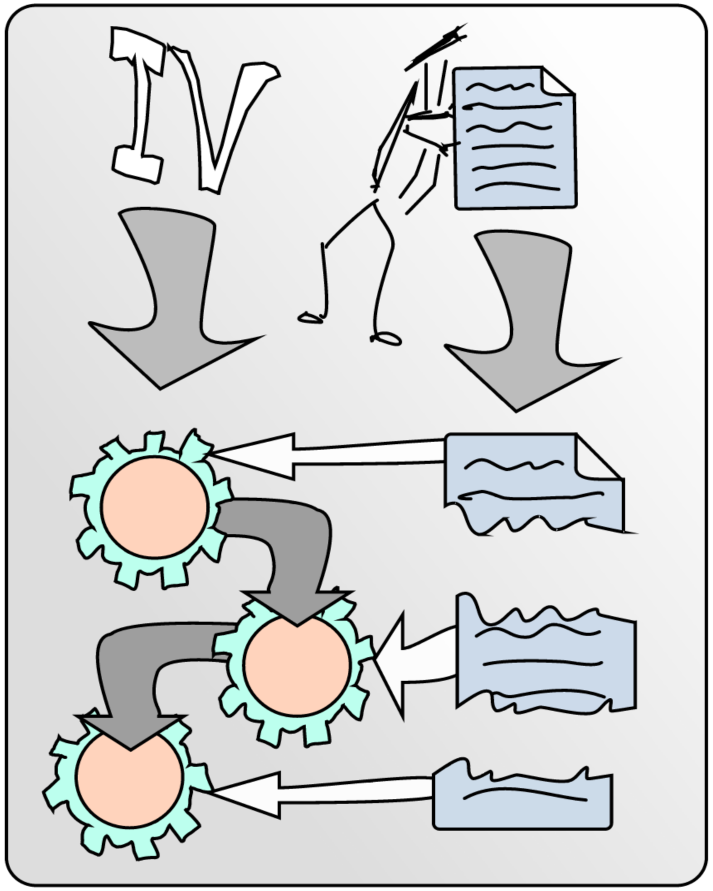
The Random Oracle

The Random Oracle allows you to replace all hash functions within your proof with a completely random function.
In the real world use a cryptographic hash construction.


Optimal Route
The Random Oracle
Play this card to get an easier proof.
$$
$$

The Real World





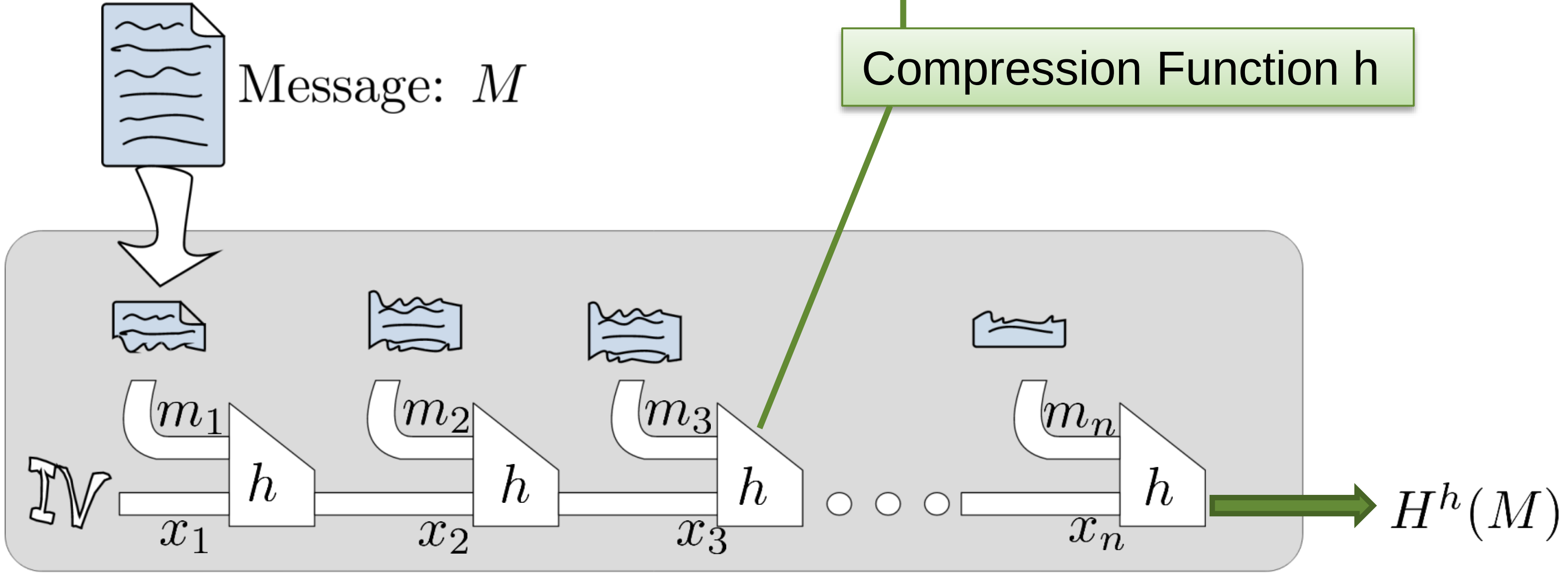
(Iterative) Hash Function Design

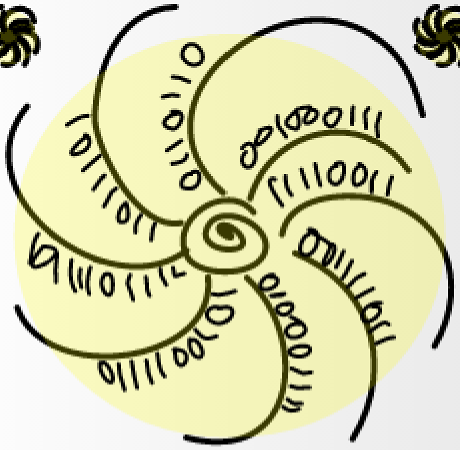
[Merkle-Damgård]

$$H^h : \{0,1\}^* \to \{0,1\}^n$$

$$h : \{0,1\}^d \times \{0,1\}^k \to \{0,1\}^s$$

Compression Function h





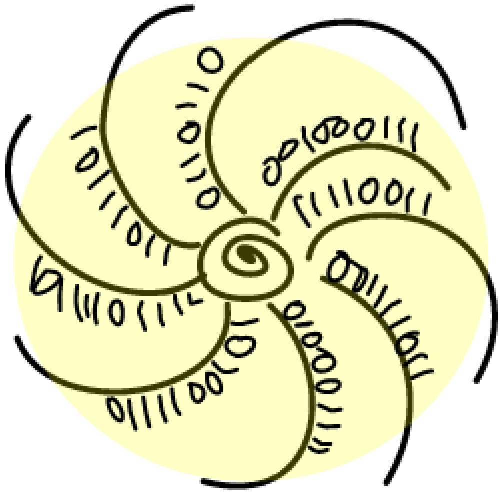
The Random Oracle

Play this card to get an easier proof.

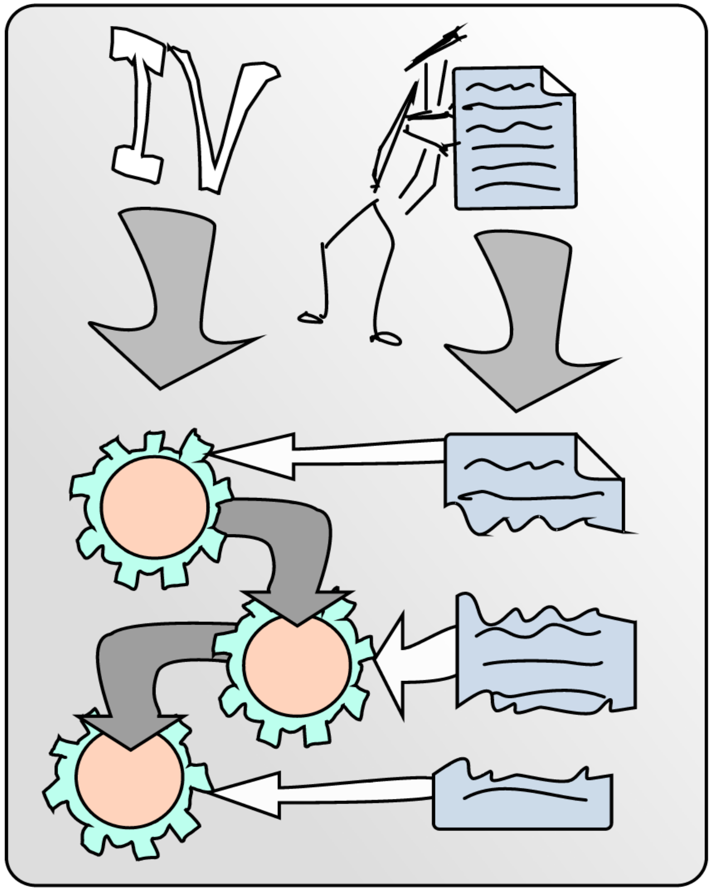
The Random Oracle

The Random Oracle allows you to replace all hash functions within your proof with a completely random function.
In the real world use a cryptographic hash construction.

Problem



$\approx$

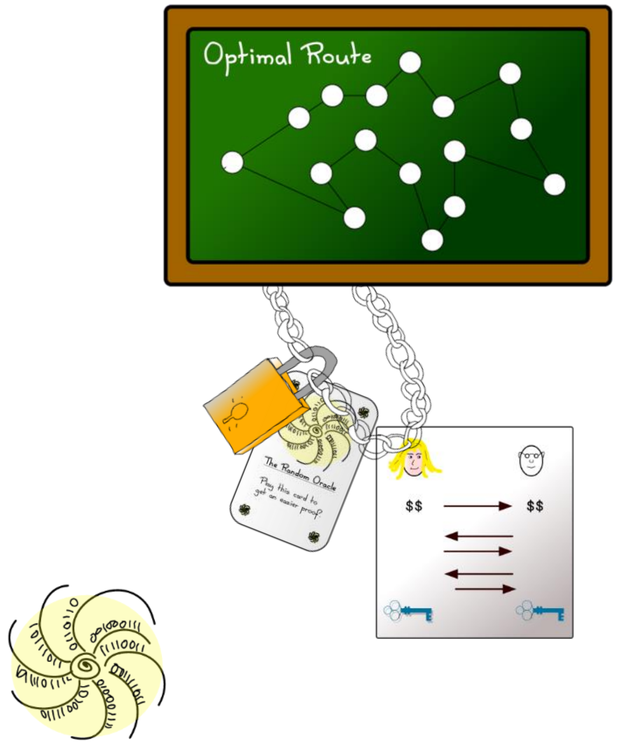


How to gain more confidence in our scheme?

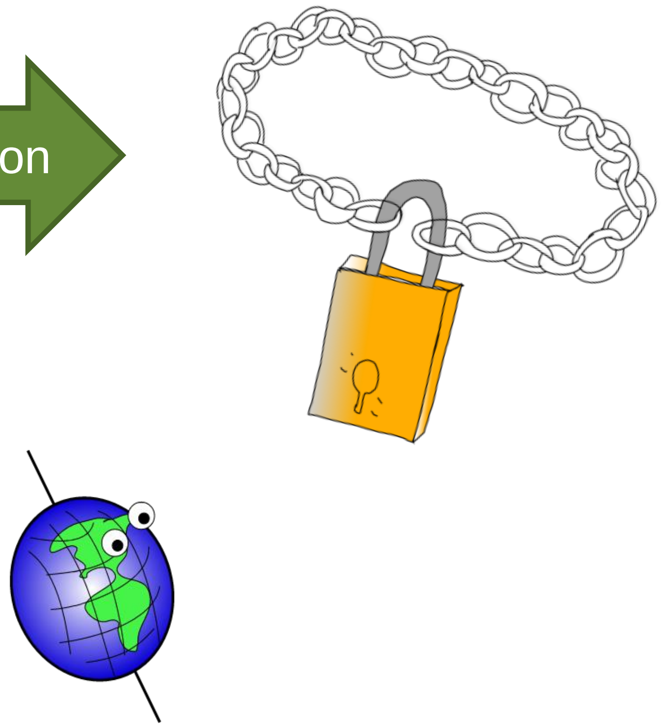
Minimizing Assumptions

Proof in the Random Oracle Model

Proof in the Real World



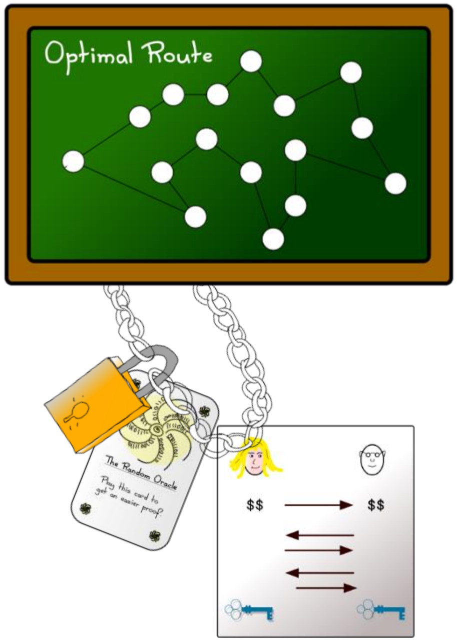
Transformation



Minimizing Assumptions

Proof in the Random Oracle Model

Proof in **something closer to Real World**



Transformation

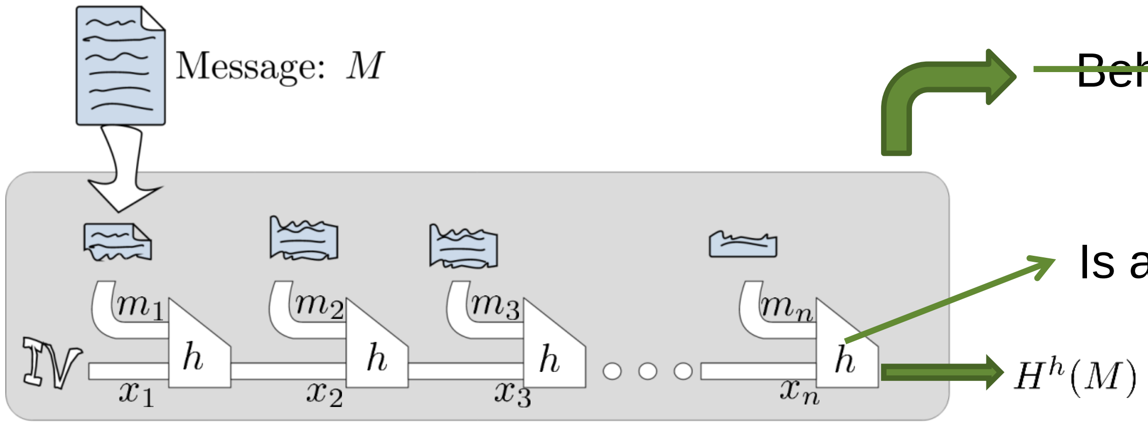
Goal

- Prove that domain extension (i.e. iteration scheme) cannot be attacked.

Assumption:
Random Oracle
$R : \{0,1\}^* \rightarrow \{0,1\}^n$

Transformation

Assumption:
Ideal Compression Function
$h : \{0,1\}^d \times \{0,1\}^k \rightarrow \{0,1\}^s$

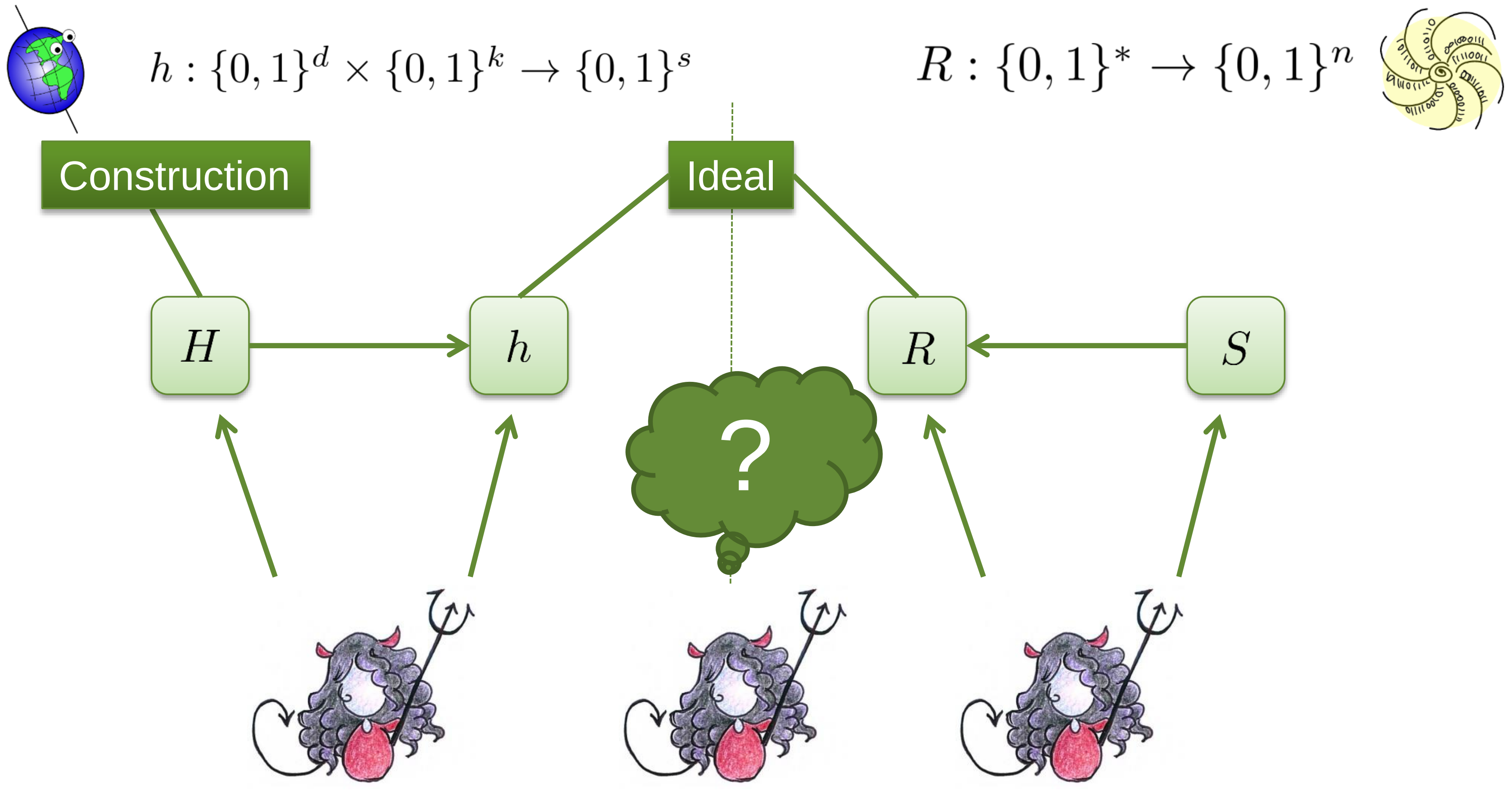


~~Behaves like a random oracle~~

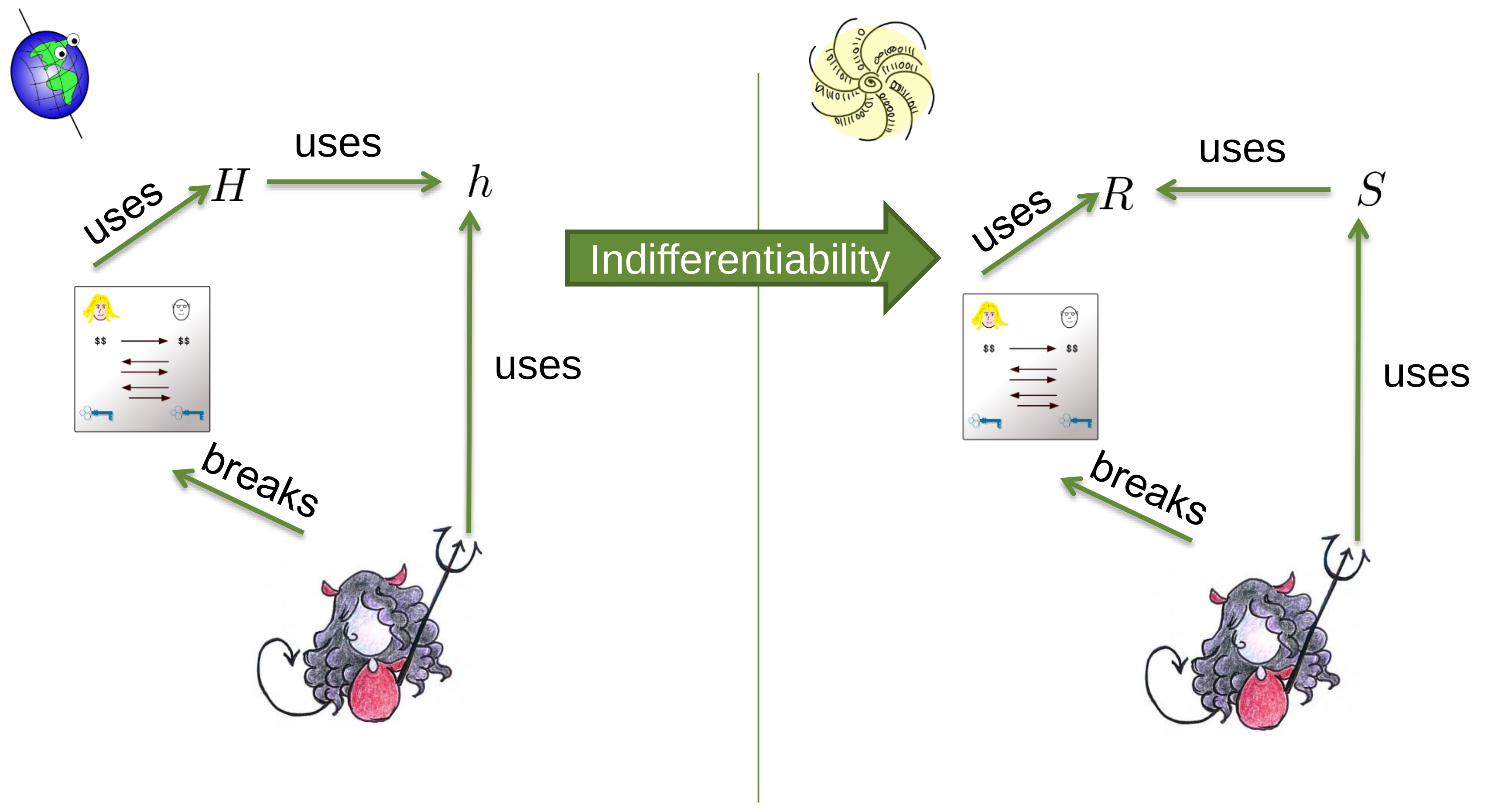
Is an ideal compression function.

Indifferentiability

[MRH04, CDMP05]


$h:\{0,1\}^d \times \{0,1\}^k \to \{0,1\}^s$
$R:\{0,1\}^* \to \{0,1\}^n$
Construction
Ideal
H
h
R
S
?

Indifferentiability



Indifferentiability

- **Reduce security** of scheme G using indifferentiable **hash construction H** (with **ideal compression function h**) to scheme G using **random oracle R**.

Indifferentiable
Hash Constructions

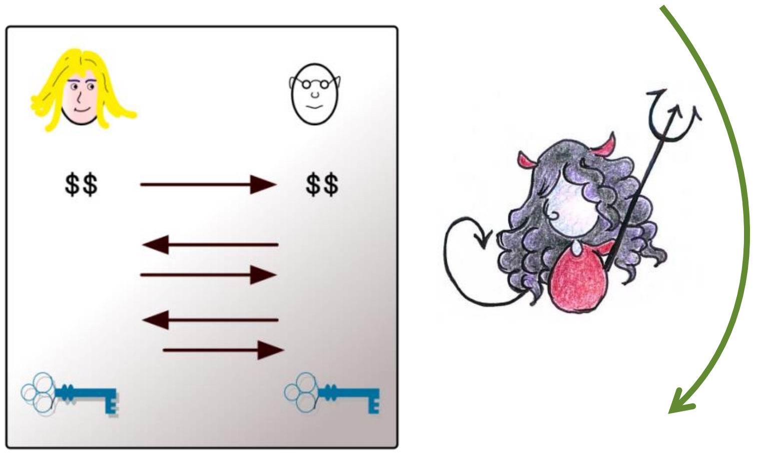
- Chop-MD
- HMAC
- NMAC
- Prefix-Free MD
- …

Enter: EUROCRYPT 2011

Indifferentiability: Not in Multi-Stage Settings

[RSS11]

Indifferentiability only works in Single-Stage Settings.

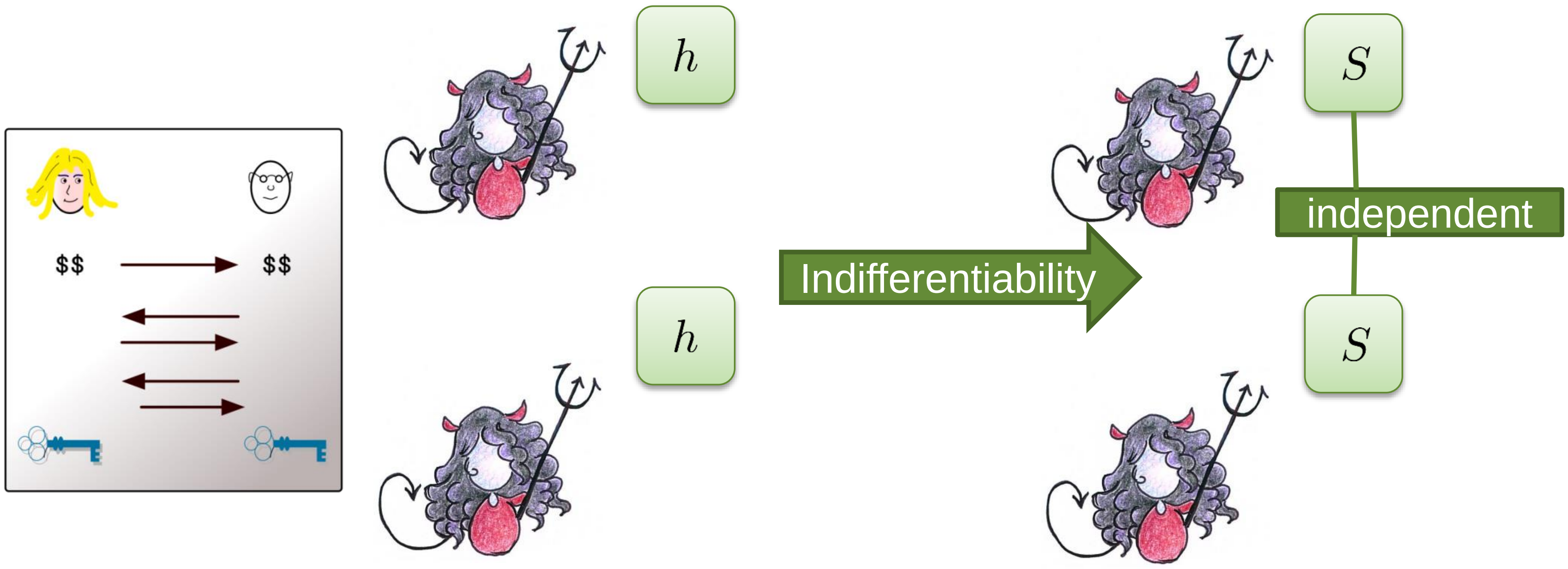


Restricted Communication

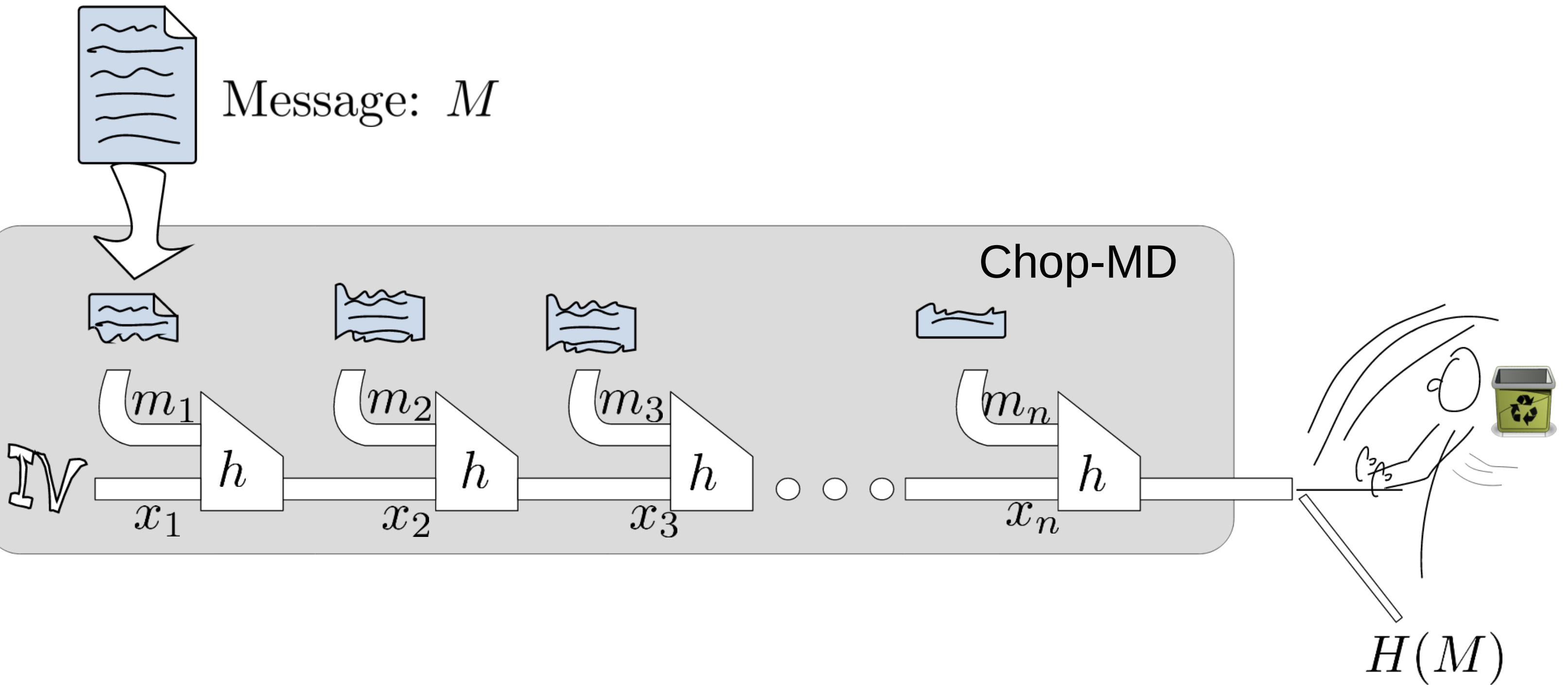
Indifferentiability: Not in Multi-Stage Settings

[RSS11]

Indifferentiability only works in Single-Stage Settings.



The Problem in a Nutshell



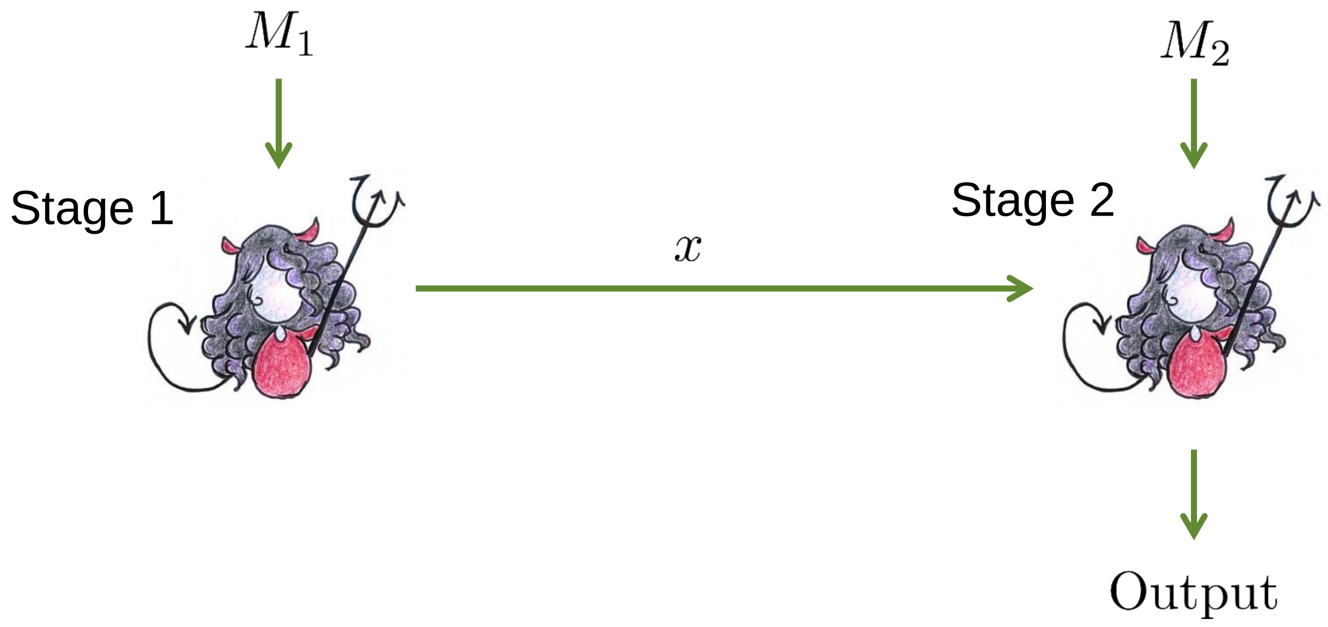
[CDMP05] Chop-MD is indifferentiable

The Problem in a Nutshell

Setting: Choose messages M_1, M_2 uniformly at random

Task: Jointly compute hash value of $M_1 \| M_2$

Restriction: $x \ll M_1$

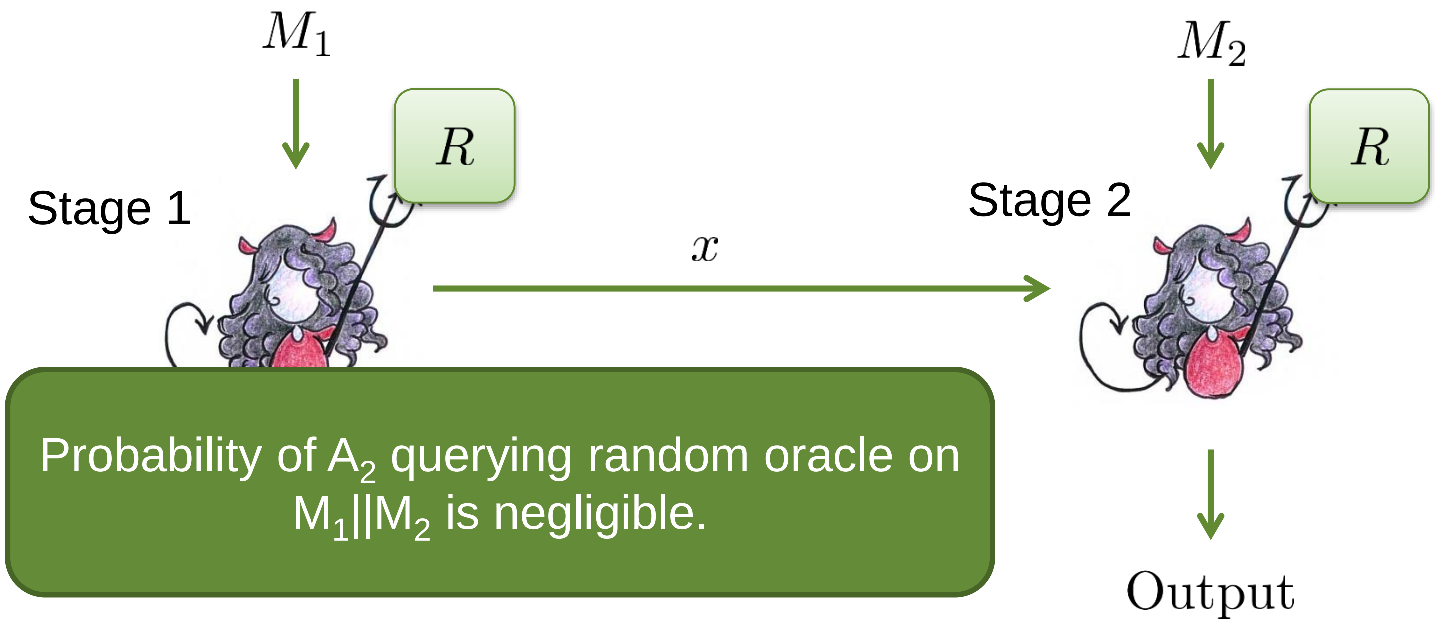


The Problem in a Nutshell

Setting: Choose messages M_1, M_2 uniformly at random

Task: Jointly compute ~~hash value~~ of $M_1 \| M_2$
Random Oracle value

Restriction: $x \ll M_1$



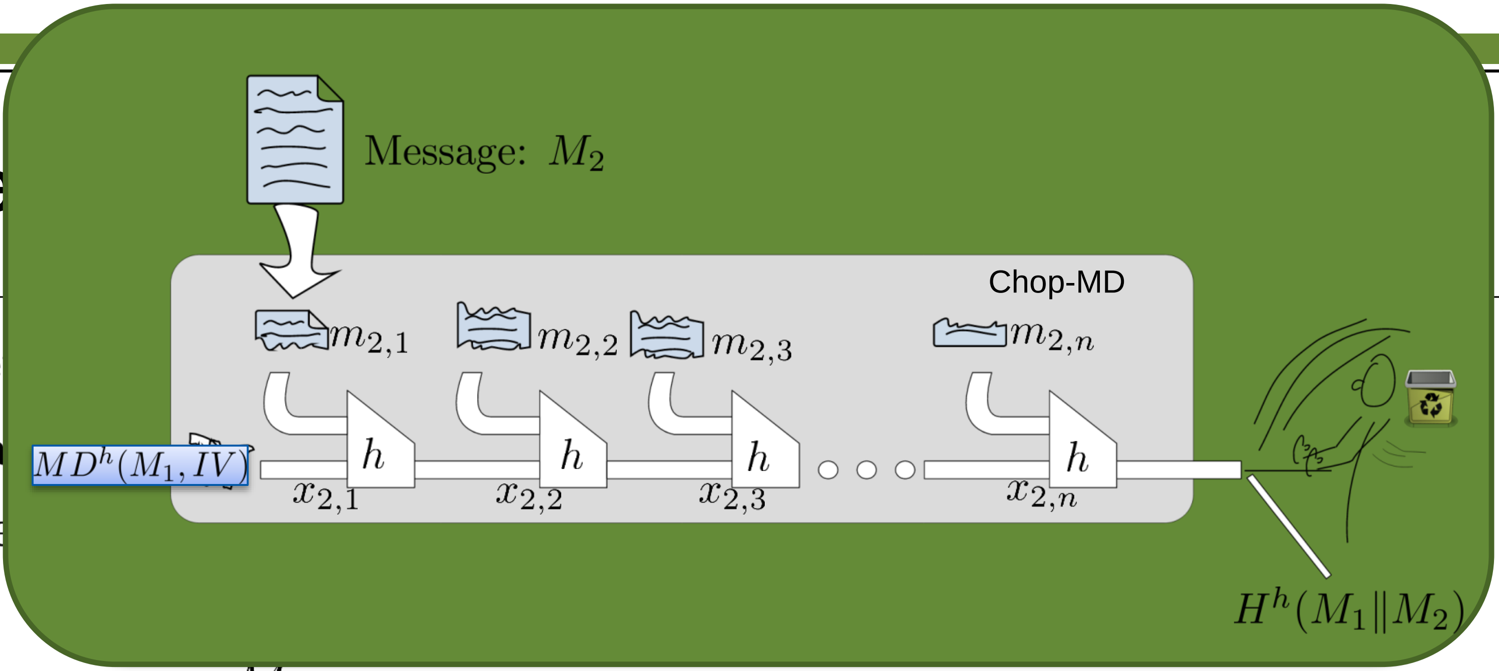
Probability of A_2 querying random oracle on $M_1 \| M_2$ is negligible.

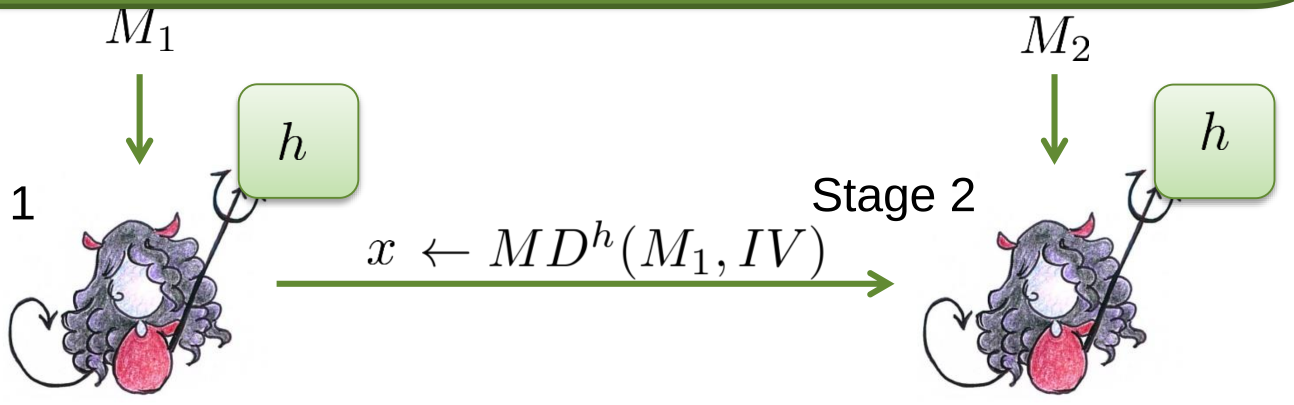
The

Se

Ta

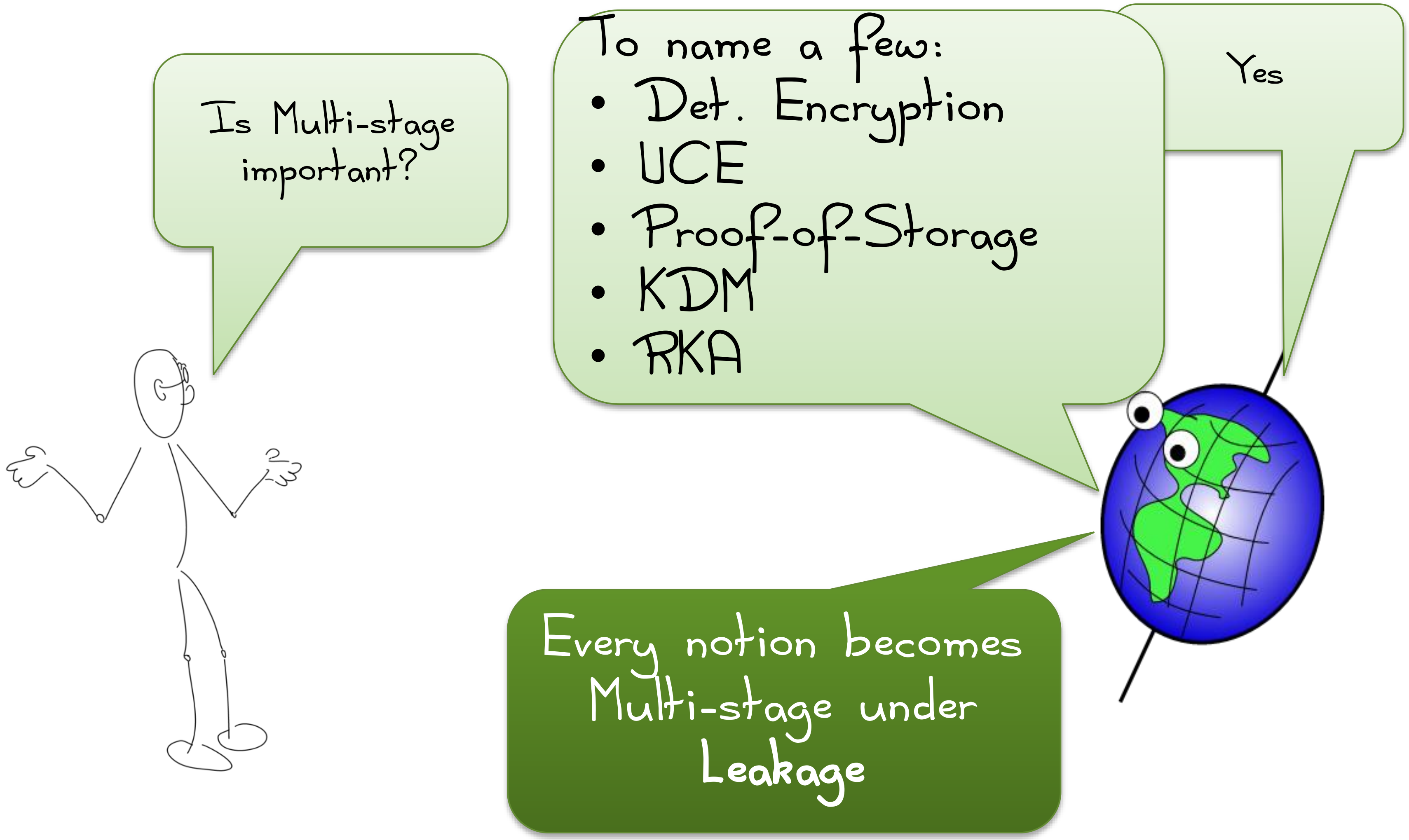
Re





$$\text{Output}: MD^h(M_2, x)_{\downarrow} = H^h(M_1\|M_2)$$

„Plain“ Indifferentiability is **not sufficient** to achieve composition in **multi-stage** settings. [RSS11]

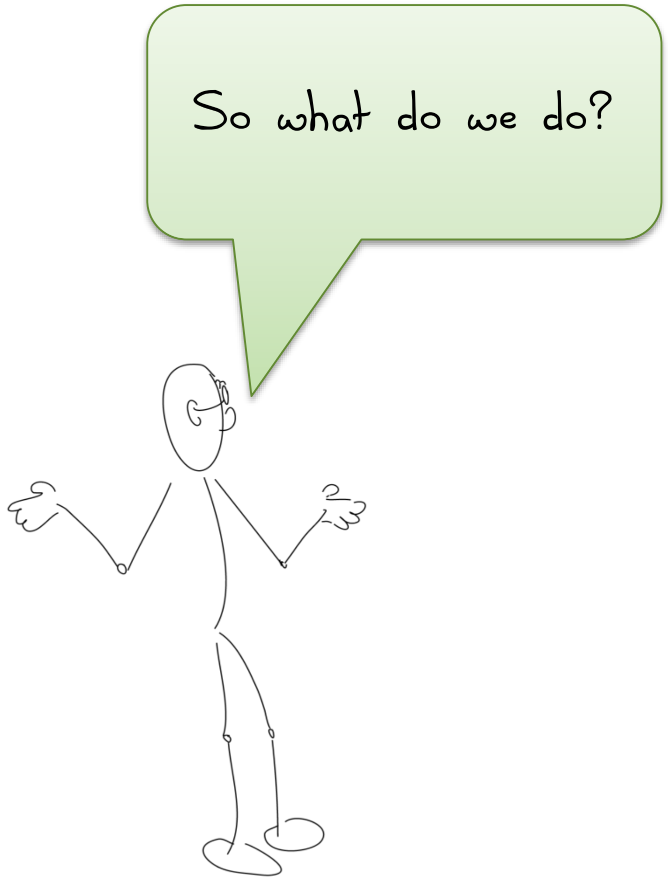

Is Multi-stage important?
Yes
To name a few:
• Det. Encryption
• UCE
• Proof-of-Storage
• KDM
• RKA
Every notion becomes Multi-stage under Leakage

Can we strengthen indifferentiability ?

Yes, but …

- Impossible for domain extenders (iterated hash constructions) [DGHM13,LAMP12,BBM13]
- Even single-reset is impossible [BBM13]

[This Paper]



1
- Formalize iterated hash constructions

2
- Formalize Problem

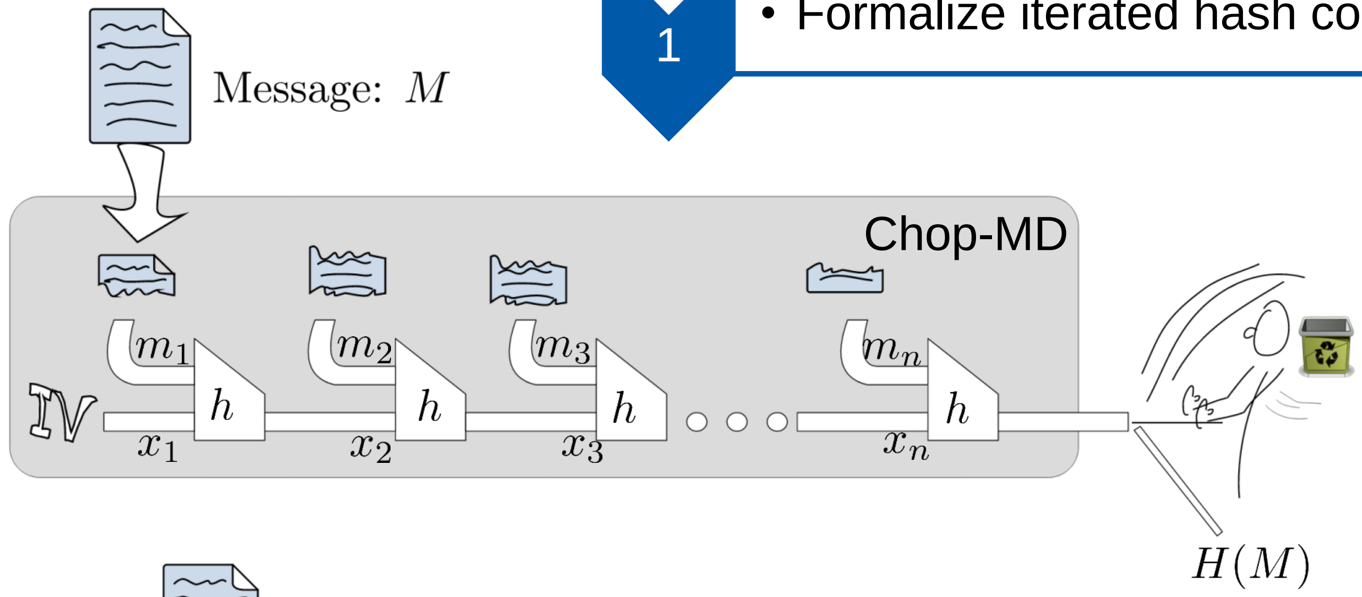
3
- Formalize joint property on game and hash constructions

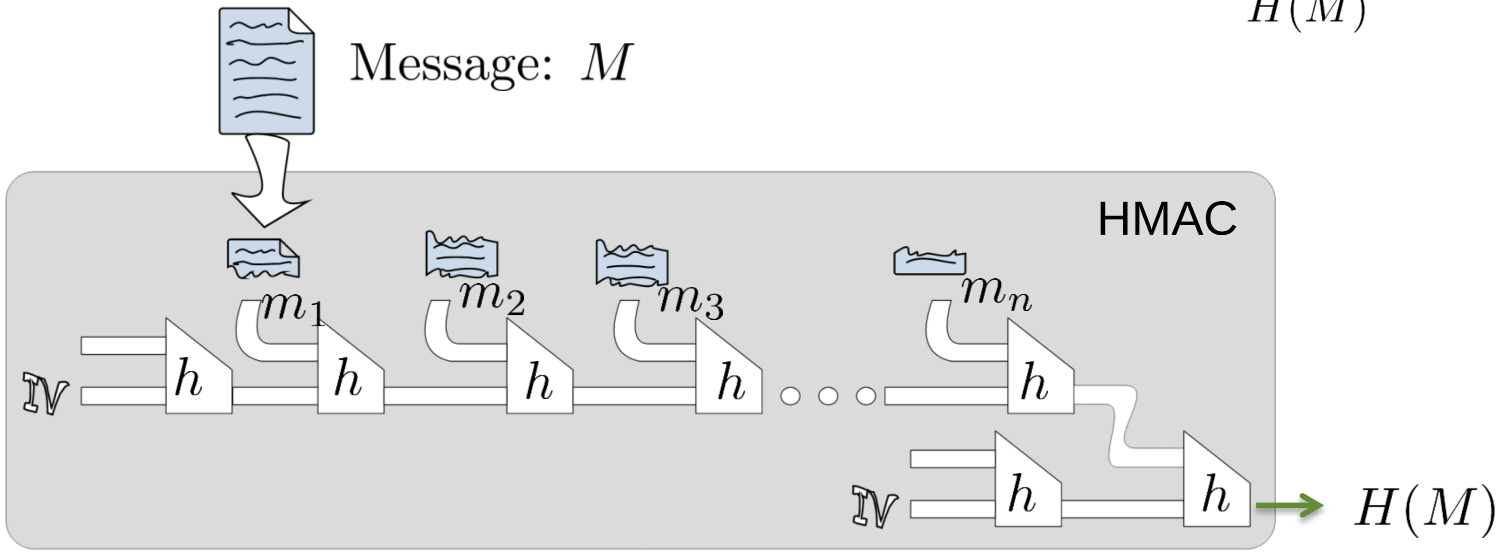
4
- Prove Composition

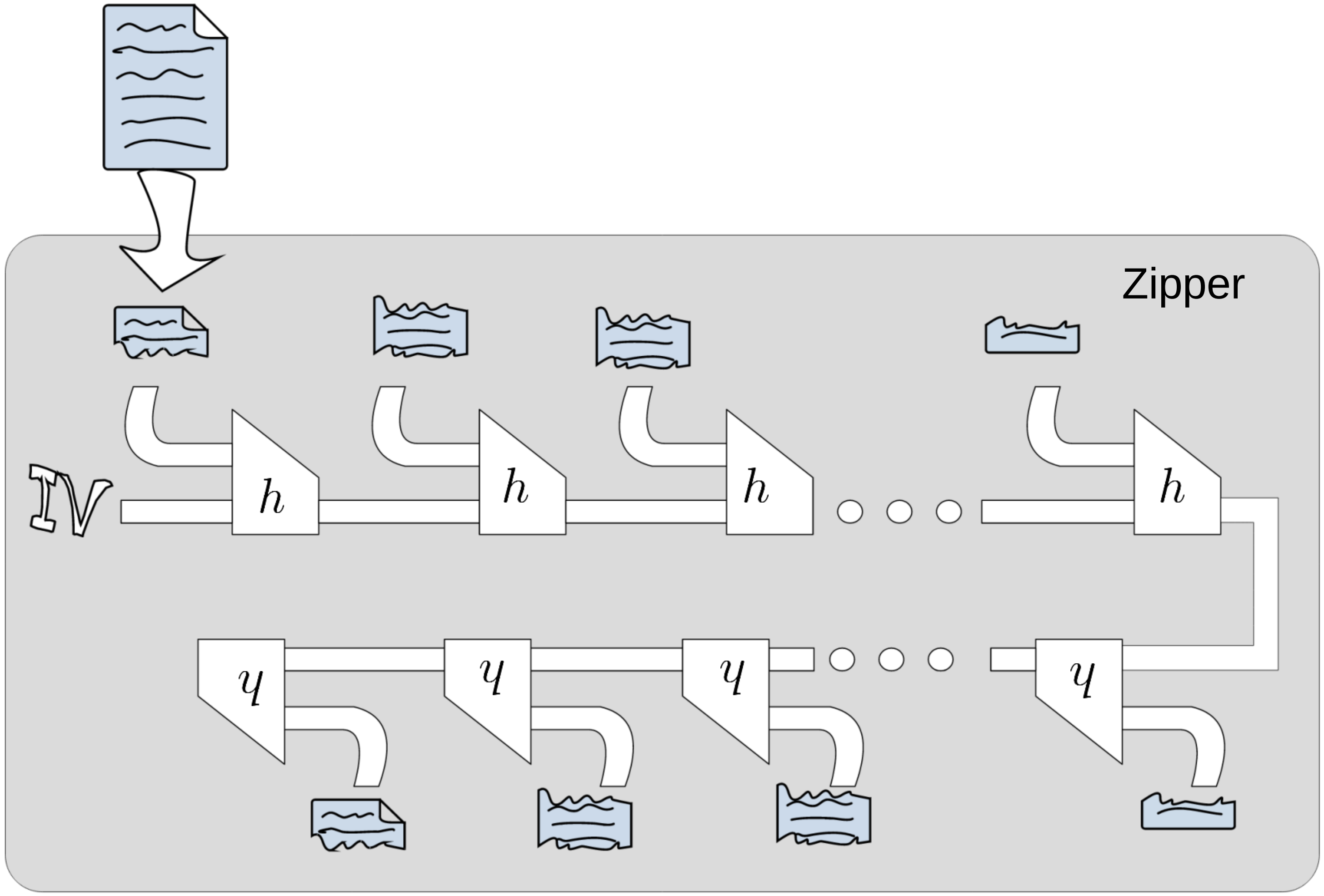
5
- Prove Property for interesting games and hash constructions.

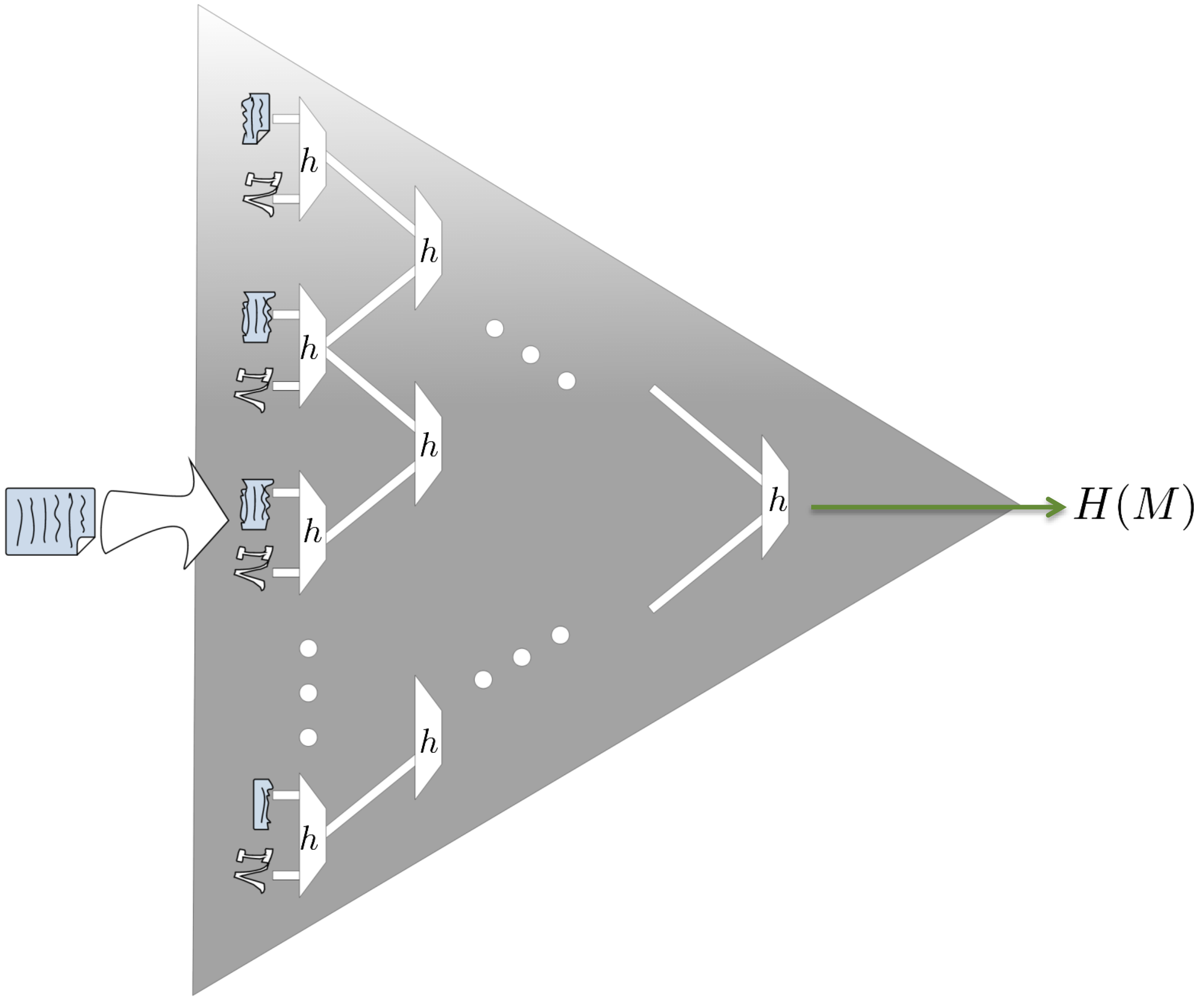
1

- Formalize iterated hash constructions

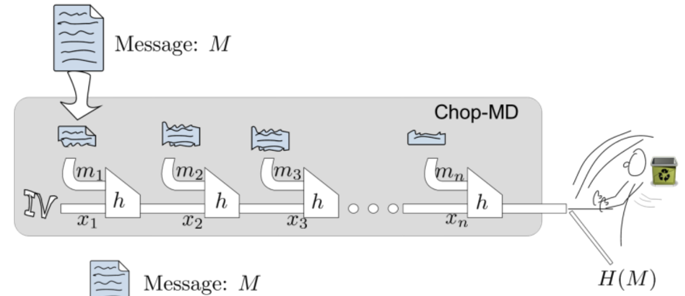


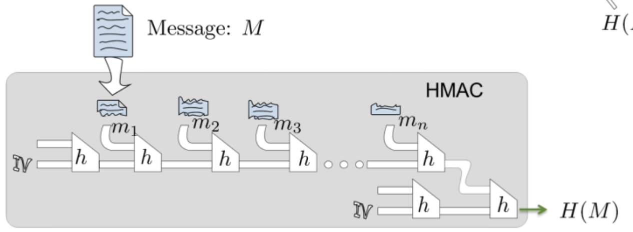


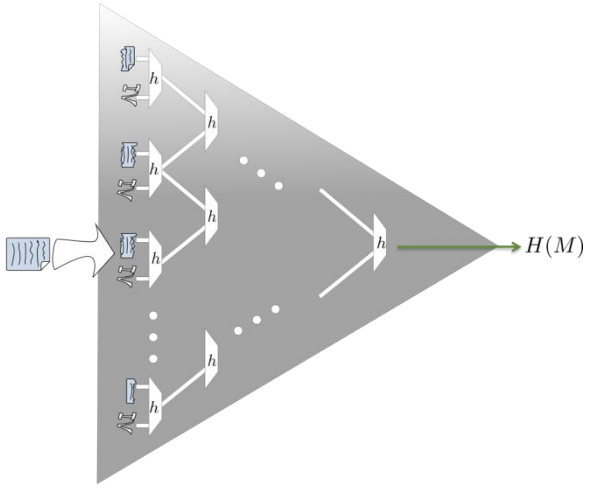

Zipper
IV
h
h
h
h
q
q
q
q


h
h
h
h
h
h
h
h
IV
IV
IV
IV
$H(M)$

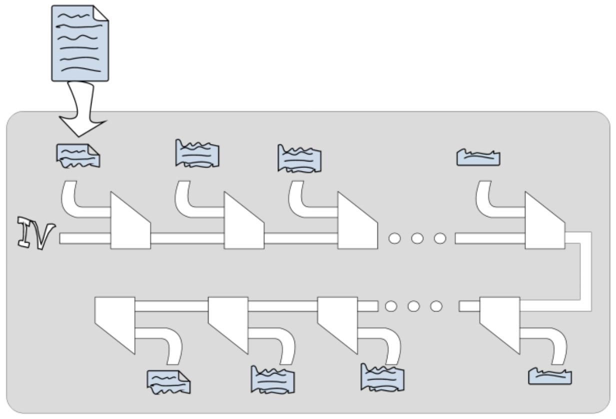
Iterative Hash Functions







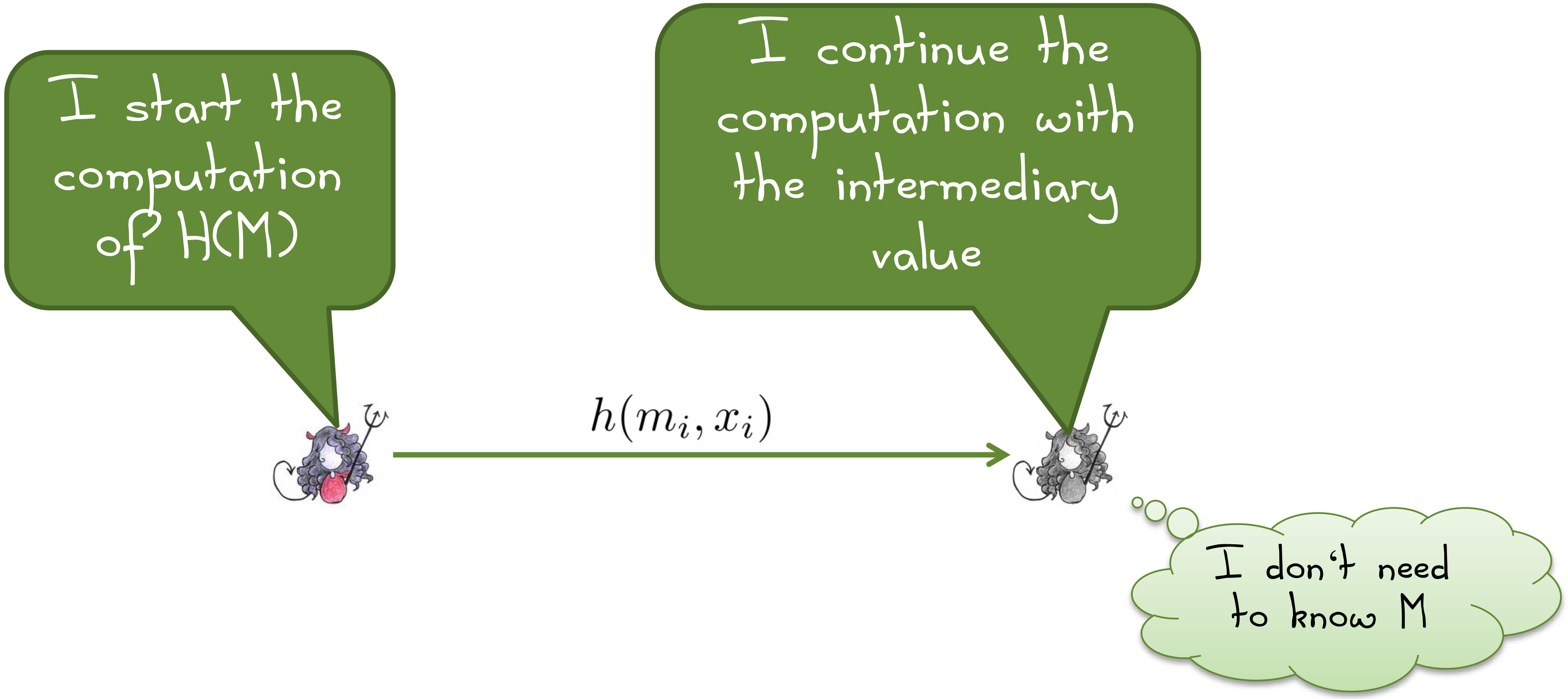
- The output of h is input to next h
- The final output is the output of h plus a simple transformation.
  - Identity
  - Projection
- Constants can be used
- Given M -> one can build an execution graph
- Given a graph -> one can extract M



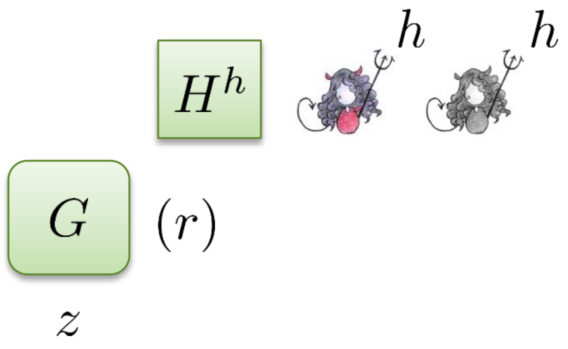
The Problem in Multi-stage Settings

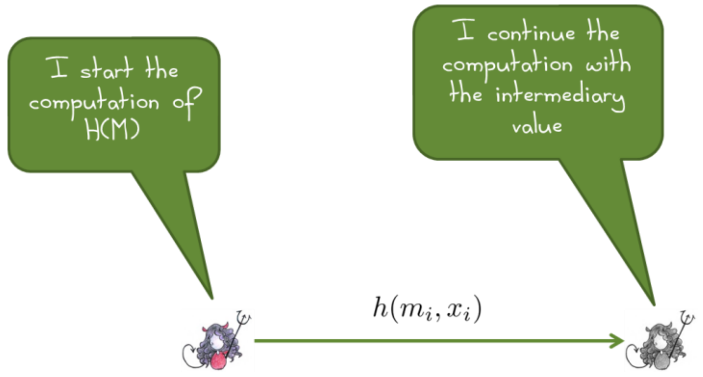
2

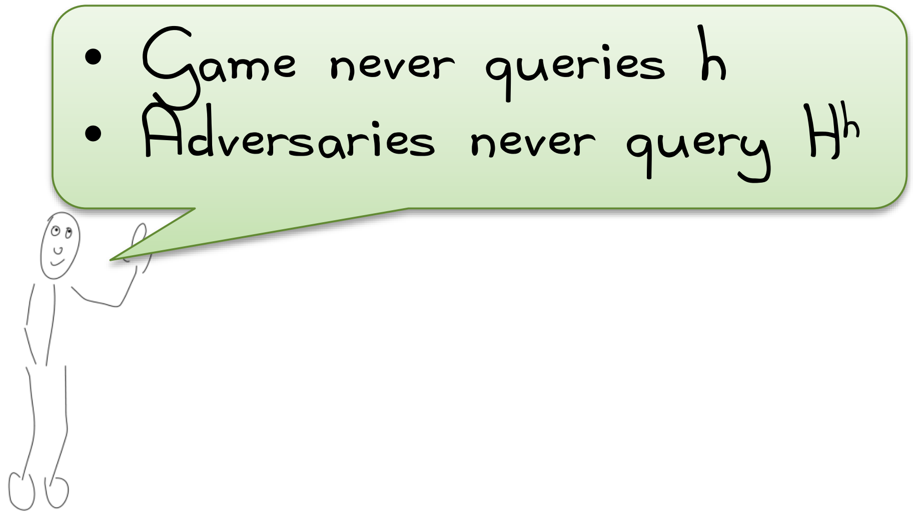
- Formalize Problem


I start the computation of H(M)
I continue the computation with the intermediary value
$h(m_i, x_i)$
I don't need to know M

Formalize Bad Event

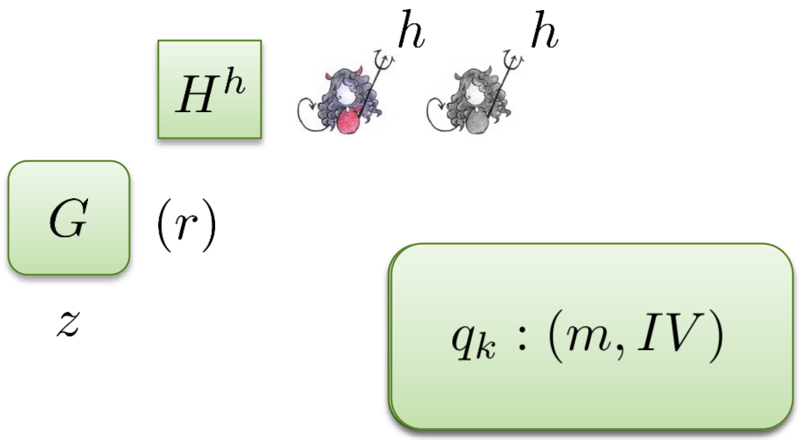


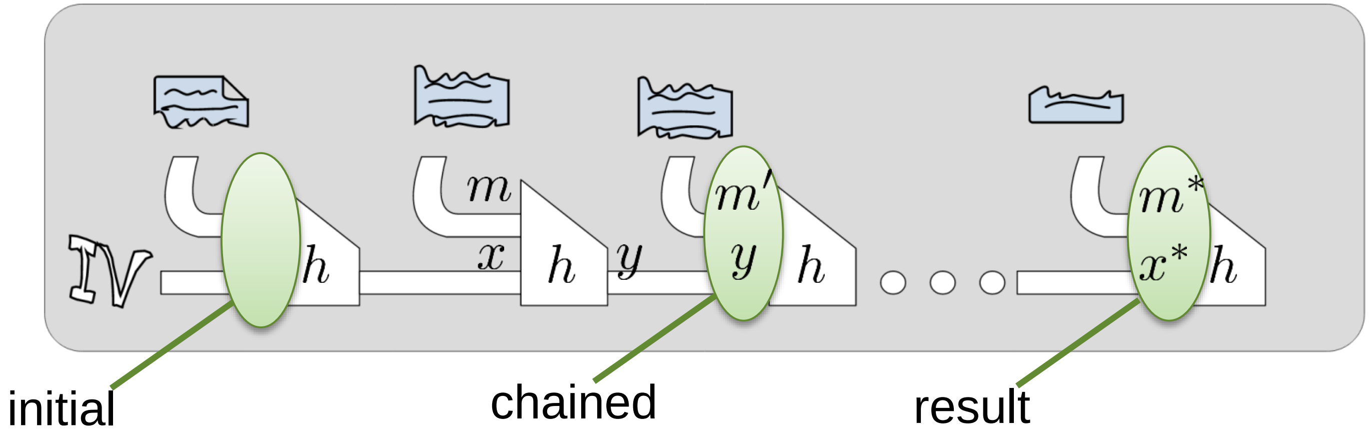

I start the computation of H(M)
I continue the computation with the intermediary value
$h(m_i, x_i)$



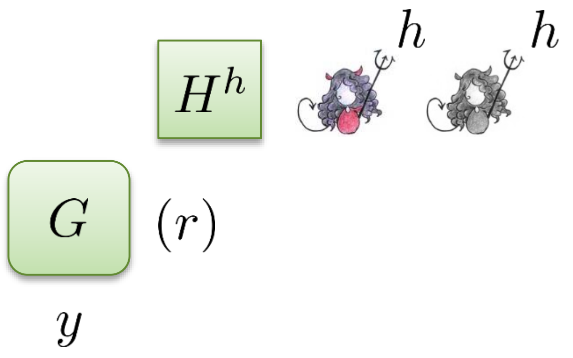
Formalize Bad Event

Fixing r induces a sequence of h-queries by adversaries.





Formalize Bad Event

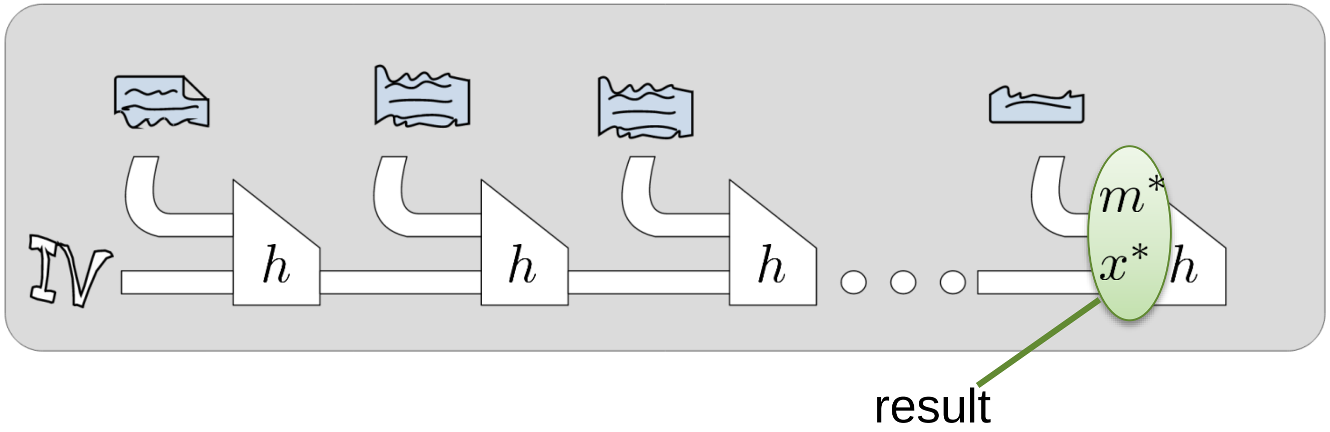


Bad result: (m*,x*) is a bad result query, if

result(m^*,x^*) relative to and

but

$\neg$chained(m^*,x^*) relative to

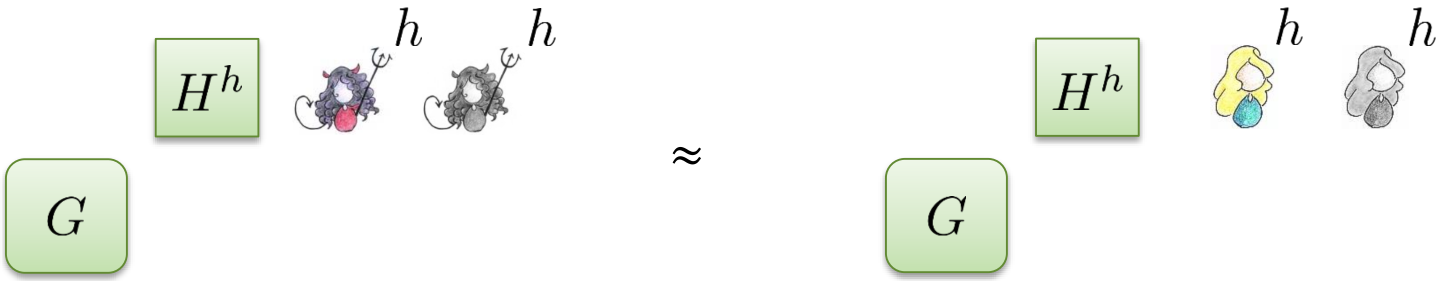


Unsplittability

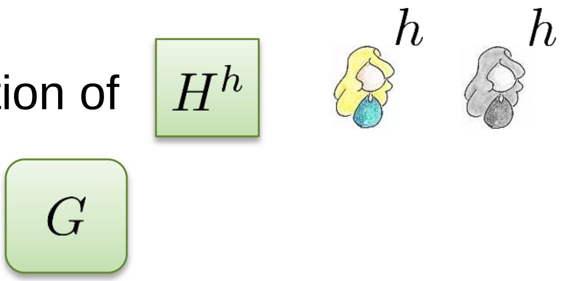
3

- Formalize joint property on game and hash constructions

A game G is UNSPLITTABLE for a hash construction H^h, if for every adversary there exists a simulator (an adapted adversary), such that



And during the execution of



bad result queries occur only with negligible probability.

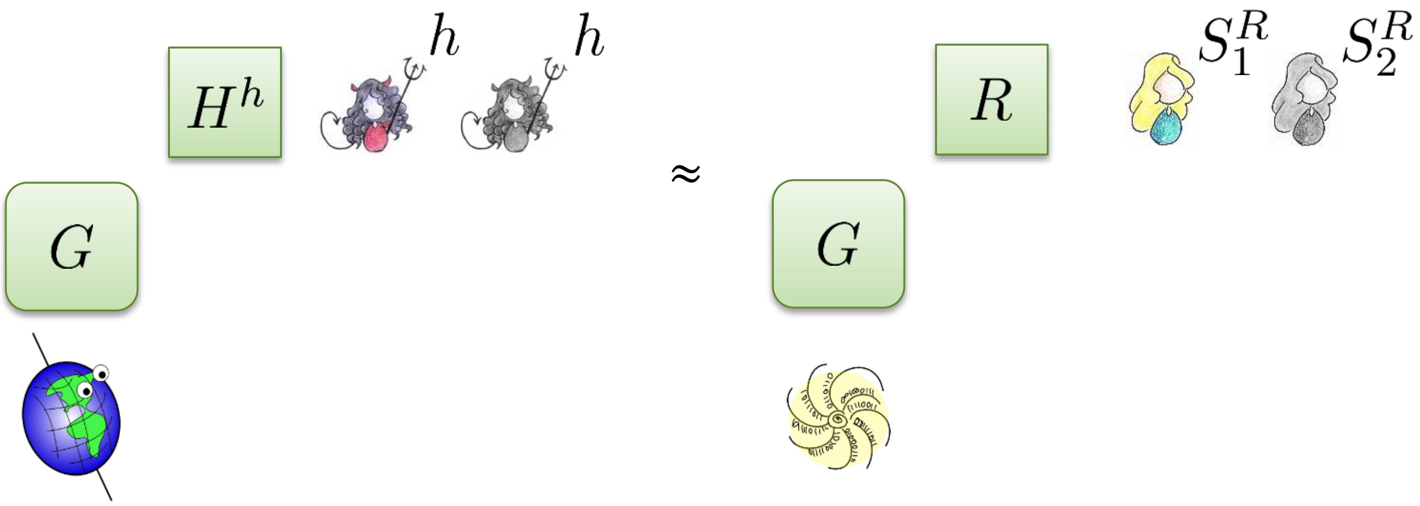
Composition

4

- Prove Composition

- If game is UNSPLITTABLE for hash construction, then a random oracle can be replaced by that hash construction.

There exists a simulator S_1, S_2 such that

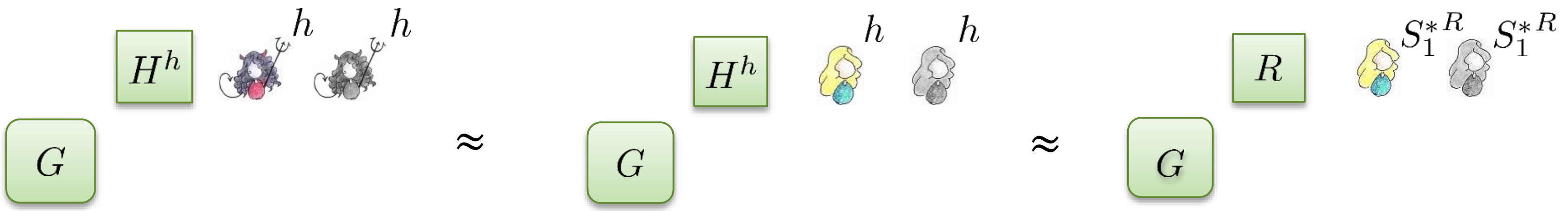


Composition

- If H^h is indifferentiable from a random oracle R there exists a simulator S for single stage settings
- From S build a canonical simulator S^*

$$\begin{aligned} \text{non result queries} &\mapsto \text{random} \\ \text{result queries} &\mapsto \text{consistent with random oracle} \end{aligned}$$

- Derandomize S^* using the random oracle [BG81]



Proof of Storage

5

- Prove Property for interesting games and hash constructions.

- RSS11 give proof-of-storage game as counter-example to general applicability of indifferentiability

- [this paper]: proof-of-storage is UNSPLITTABLE for any **multi-round** hash construction.

Three Two-stage Security Games

5

- Prove Property for interesting games and hash constructions.

CDA

```
$b \leftarrow \{0,1\}$
$(pk, sk) \leftarrow \mathsf{kgen}(1^\lambda)$
$(\mathbf{m_0}, \mathbf{m_1}, \mathbf{r}) \leftarrow \mathcal{A}_1^h(1^\lambda)$
$\mathbf{c} \leftarrow \mathcal{E}^{H^h}(pk, \mathbf{m_b}; \mathbf{r})$
$b' \leftarrow \mathcal{A}_2^h(pk, \mathbf{c})$
return $(b = b')$
```

MLE

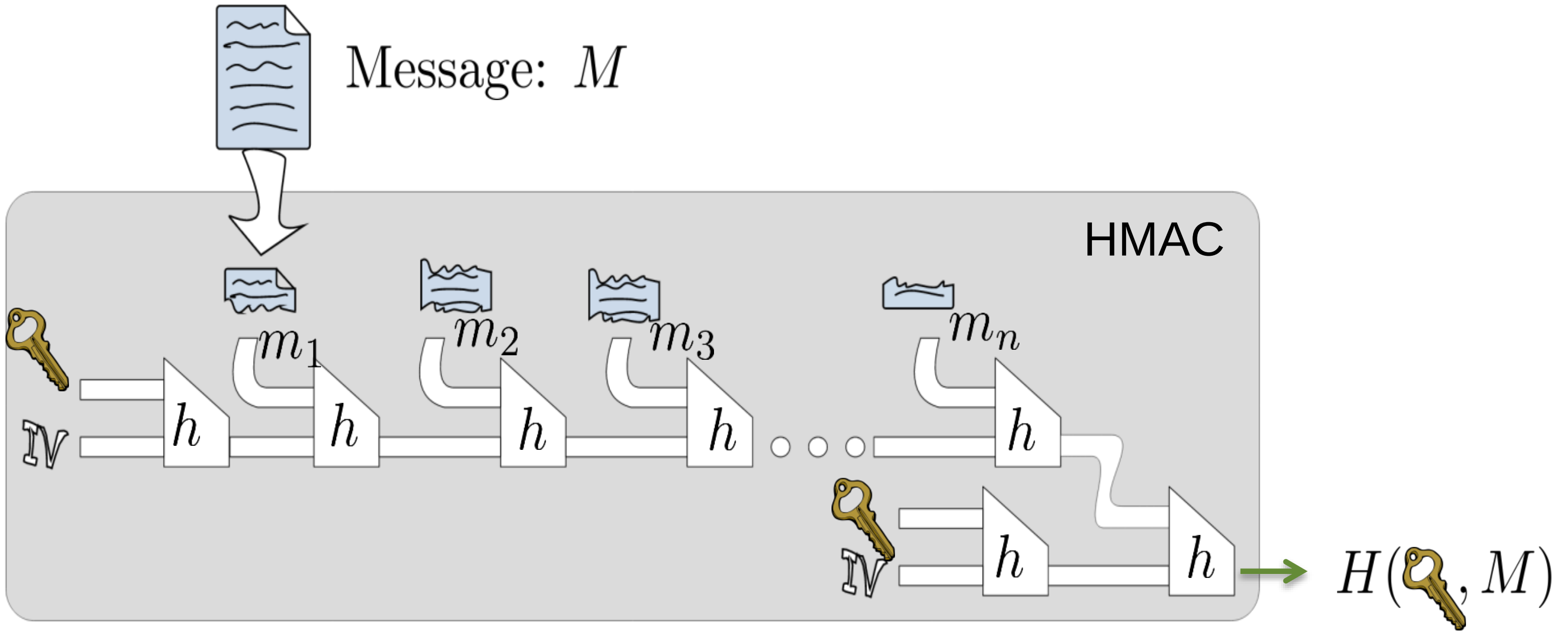
```
$P \leftarrow \mathcal{P}$
$b \leftarrow \{0,1\}$
$(\mathbf{m_0}, \mathbf{m_1}, Z) \leftarrow \mathcal{A}_1^h(1^\lambda)$
$\mathbf{c} \leftarrow \mathcal{E}_P^{H^h}(\mathcal{K}_P(\mathbf{m_b}), \mathbf{m_b})$
$b' \leftarrow \mathcal{A}_2^h(P, \mathbf{c}, Z)$
return $(b = b')$
```

UCE

```
$b \leftarrow \{0,1\};\ hk \leftarrow \mathsf{kgen}(1^\lambda)$
$L \leftarrow \mathcal{S}^{\text{HASH}}(1^\lambda)$
$b' \leftarrow \mathcal{D}(1^\lambda, hk, L)$
return $(b = b')$
```

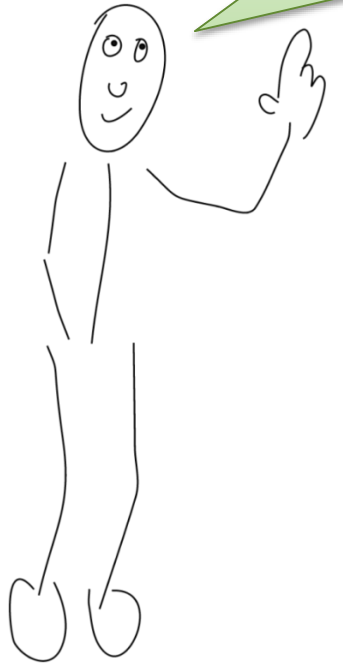
```
$\underline{\text{HASH}(x)}$

if $T[x] = \perp$ then
    if $b = 1$ then
        $T[x] \leftarrow H^h(hk, x)$
    else $T[x] \leftarrow \{0,1\}^\ell$
return $T[x]$
```

Theorem: If only last adversary gets hash keys used by game, then the game is UNSPLITTABLE for key-prefixed hash constructions.



- **Summary**
  - **Unsplittability** allows to use indifferentiability in a multi-stage setting.
  - Interesting games and hash constructions can be shown to be unsplittable.
  - Study of multi-stage games provides insights into hash function design: **keyed**-constructions, **multi-round**
- **Open Problems**
  - **Sufficient conditions** for unsplittability
  - **Ideal ciphers** instead of compression functions
    - Block-cipher based compression functions
    - SHA-3 (Keccak)

Where we are – where to go

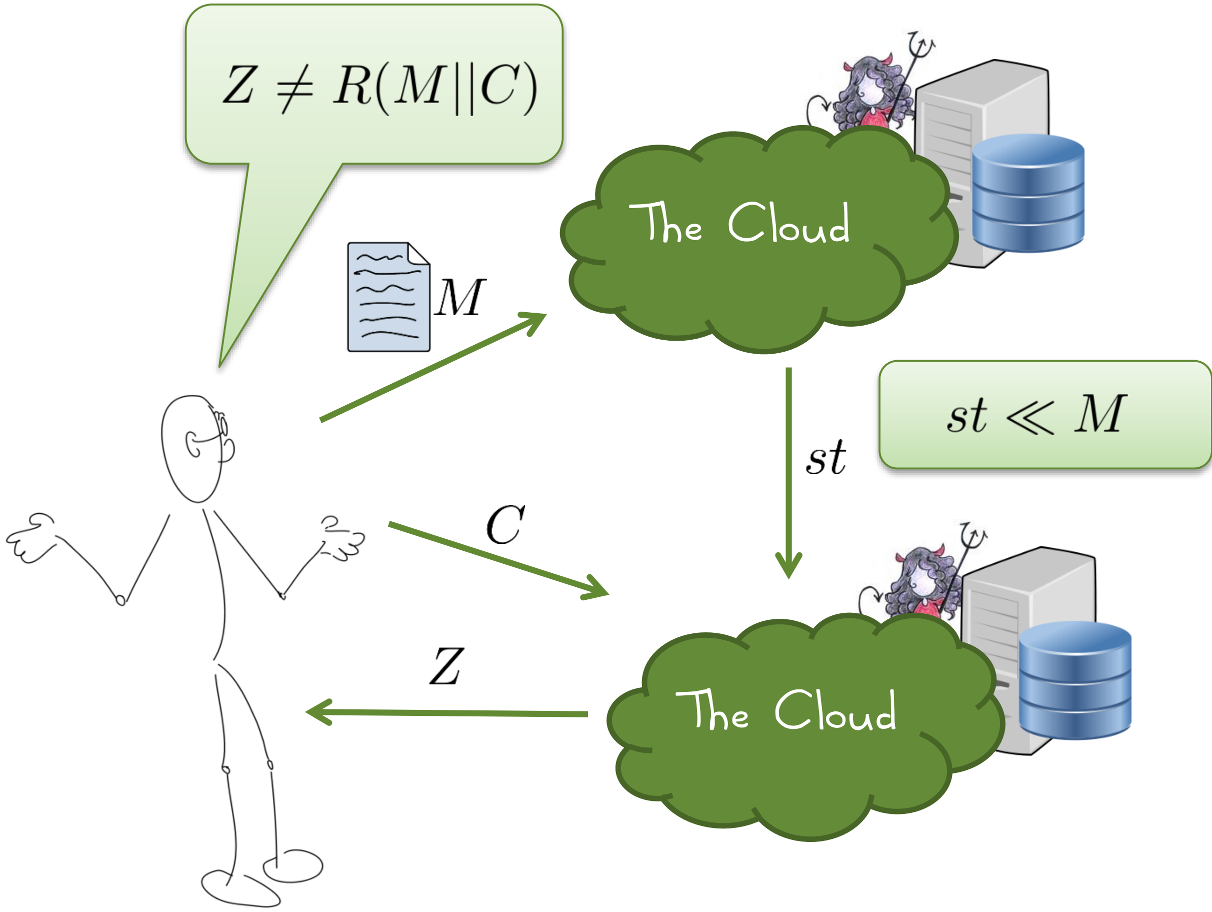


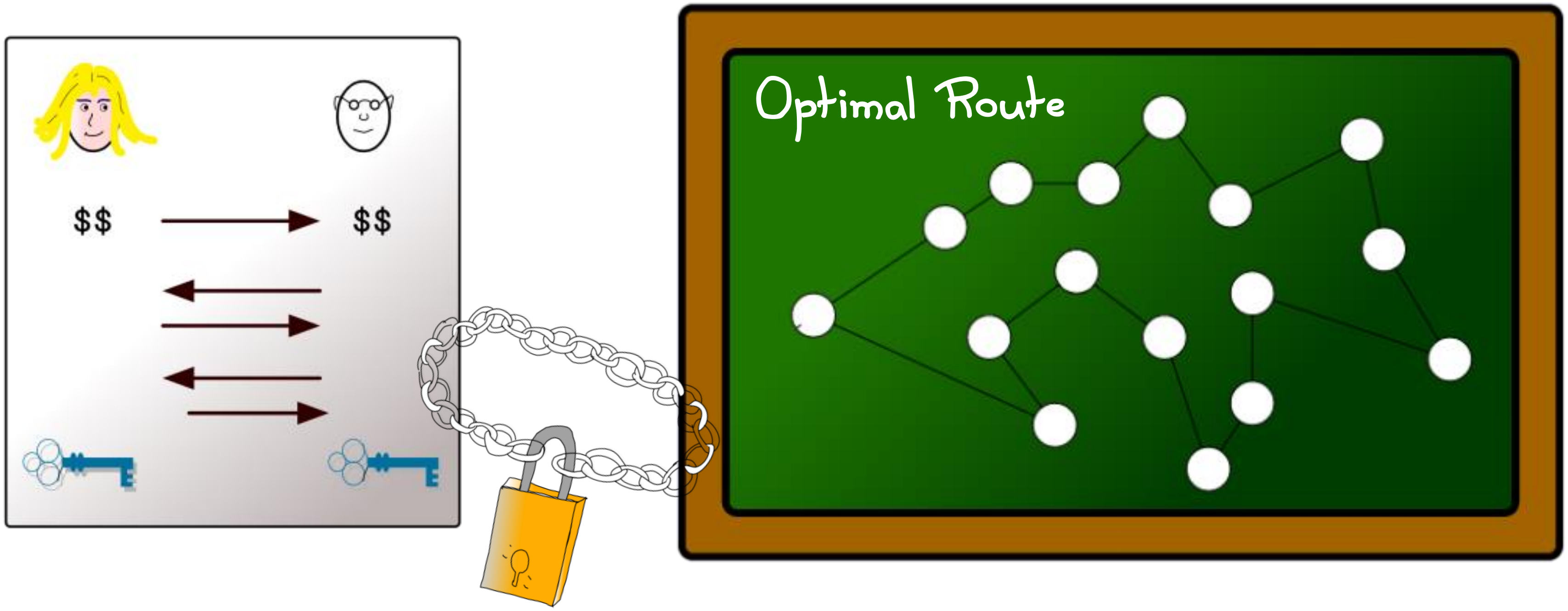
Here are a language and tools to work with indifferentiability in a multi-stage setting.

Of course there is sitll work:
- Ideal Cipher Model
- SHA-3
- Conditions for Unsplittability

Indifferentiability: an Example

```
$M \leftarrow \{0,1\}^p$
$st \leftarrow \mathcal{A}_1^R(M, 1^\lambda)$
**if** $|st| > n$ **then**
    **return** false
$C \leftarrow \{0,1\}^c$
$Z \leftarrow \mathcal{A}_2^R(st, C)$
**return** $(Z = R(M\|C))$
```




$$
$$
Optimal Route