Lecture 3 - Collision Resistance

6.5060 - Fall 2021

C-G, Devadas, Kalai, Zeldovich

MIT

Collision - Resistant Hash Functions

Logistics

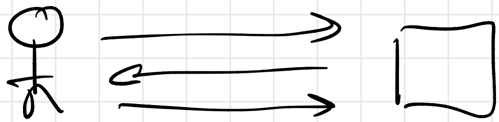
* Lab 0 code & Lab 0 theory due tomorrow 10pm ET via Gradescope
  ↳ Latex for written parts
* Lab 1 out on Friday
* Interrupt any time! Hard to see faces w/ masks... use eyebrows?

Plan

* Intuition & motivation
* Defn of CRHF
* Constructions
* More applications

Last time...

authenticating PEOPLE



Passwords, Pass storage, MACs, biometrics, ...

yes!

Today...

authenticating FILES / CODE / DATA

Main new tool:

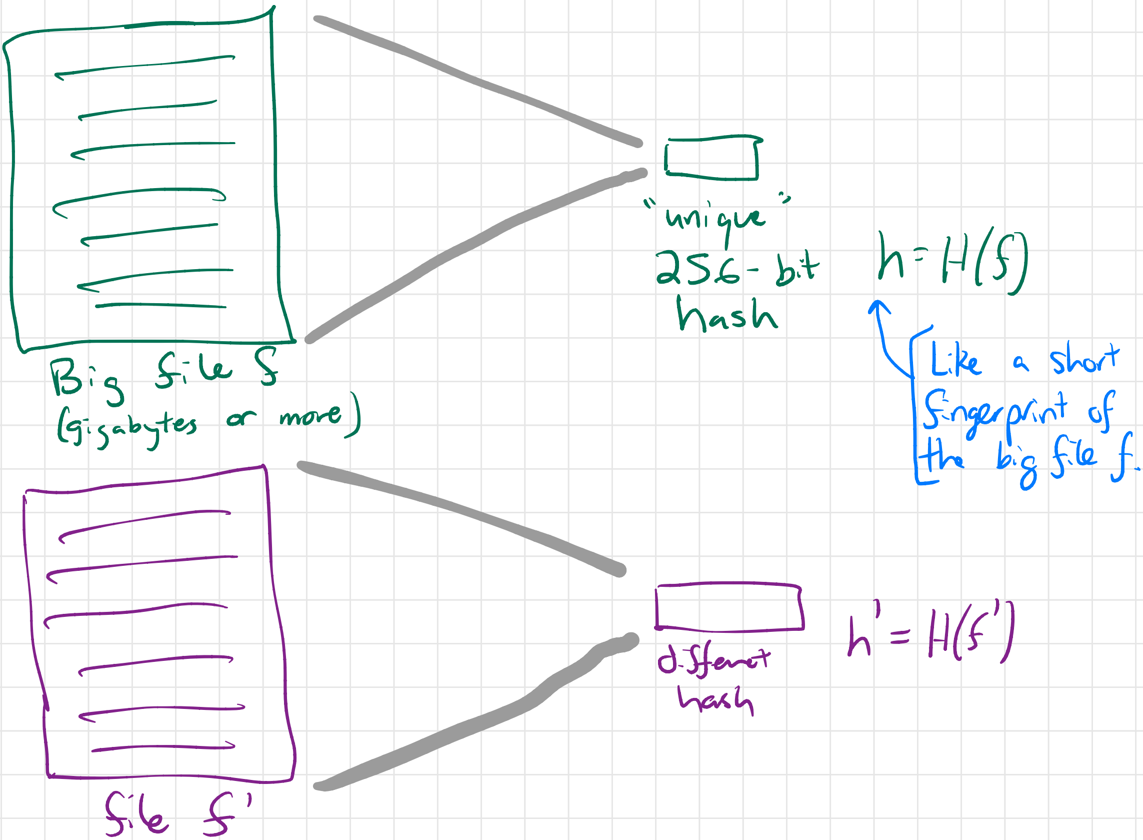
Collision-resistant hash functions (CRHF)

Intuition behind CRHF....

compressing!

A hash fn $H: \{0,1\}^* \longrightarrow \{0,1\}^{256}$

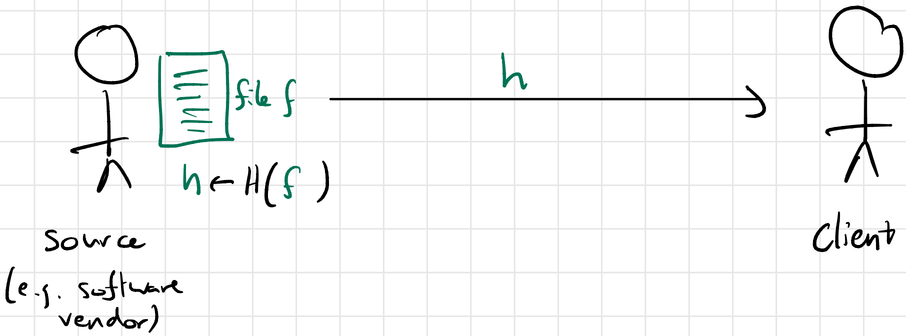
↳ In practice SHA2/SHA256, SHA3/Keccak, ... (Broken: MD5, SHA1)



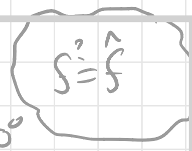
Security goal: It is "computationally infeasible" to find two distinct files that have the same hash value (a "collision")

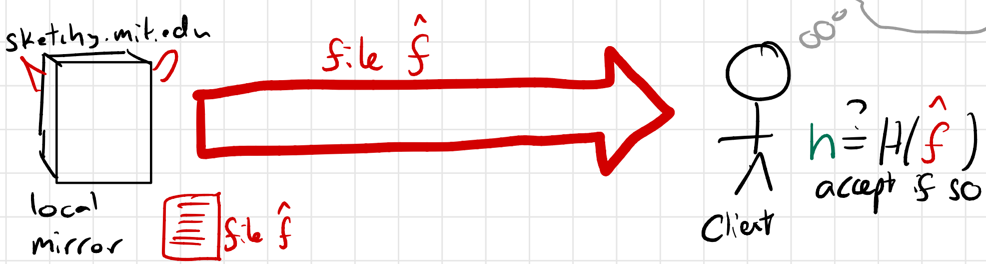
Application I: Secure mirroring

1. Get hash from trustworthy source



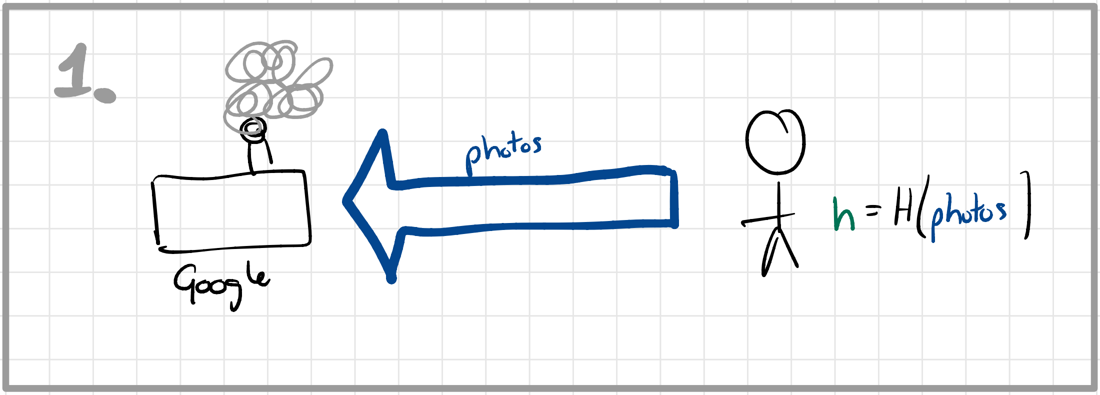
2. Fetch large file from untrustworthy source

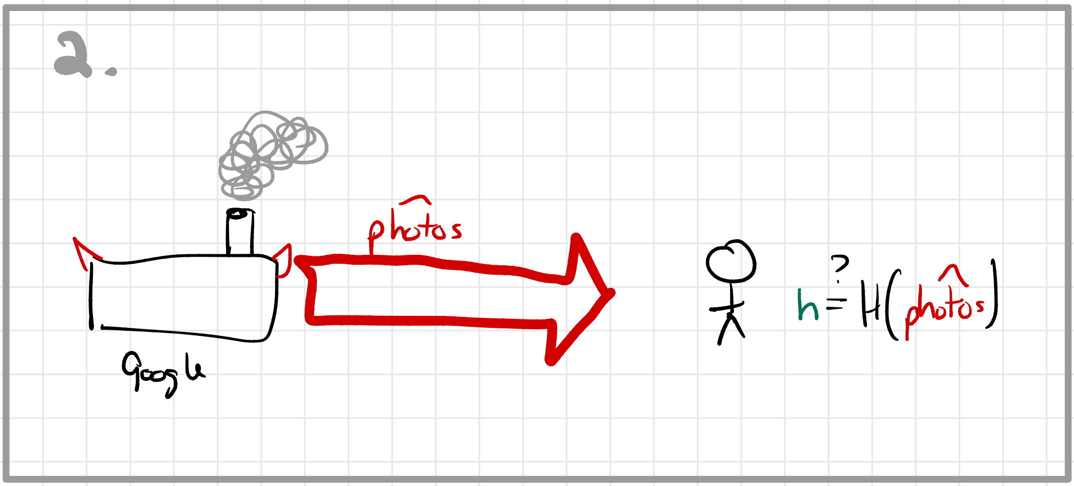




If Hash is CRHF, then sketchy mirror will not be able to find a file $\hat{f} \neq f$ that client will accept.

Application II: Outsourced File Storage



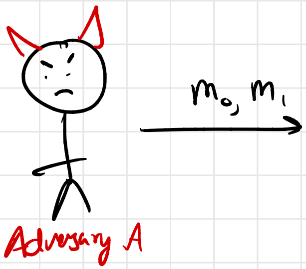


If hash is CRHF, then Google can't trick you into accepting incorrect photos/files.

More generally, CRHFs let you authenticate a LONG message by authenticating only a SHORT string.

We will see more applications...
"Hash and Sign", ...

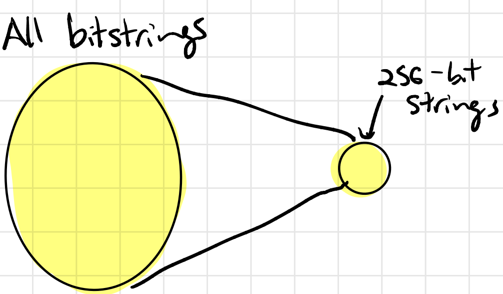
Adversary's goal in breaking CRHF:



Distinct msgs s.t.

$$H(m_0) = H(m_1)$$

Observe: There are lots of collisions!



Mega pigeonhole principle!

Jamming infinitely many pigeons in finite holes

If CRHF is good/secure, these collisions will be "hard to find."

↳ How do we formalize this?

Definition: Collision-Resistant Hash Function

A function $H: \{0,1\}^* \to \{0,1\}^\lambda$ is collision resistant if for all "efficient" adversaries A

Adv is randomized

$$\Pr\left[\begin{matrix} m_0 \neq m_1 \\ H(m_0) = H(m_1) \end{matrix} : (m_0, m_1) \leftarrow A() \right] \leq \text{"negligible"}$$

(To be useful, H must also be efficiently computable.)

In theory:

λ = "security parameter" ($\approx$ key length)

"efficient" = randomized alg runs in time poly(λ)

"negl" $= O\left(\frac{1}{\lambda^c}\right) \; \forall c \in \mathbb{N}$

(e.g. $\frac{1}{2^\lambda}, \frac{1}{2^{\sqrt{\lambda}}}, \frac{1}{\lambda^{\log \lambda}}, \ldots$)

In practice:

$\lambda = 128, 256, 384$

"efficient" adversary $\simeq$ runs in time $\leq 2^{128}$

"negl" $\simeq$ prob $\leq 2^{-128}$

In practice, aim to defend against advs running in time $< 2^{128}$.

Time

2^{30} ops/sec on your laptop

2^{58} ops/sec on Fugaku supercomputer ($\approx$ \$1 billion)

2^{66} hashes/sec computed by Bitcoin miners

2^{90} hashes/year " " "

2^{114} hashes requires enough energy to boil all water on the planet

2^{140} hashes requires one year of sun's energy

Lenstra Kleinjung Thomé

Probability

2^{-1} fair coin lands heads

2^{-8} tax returns audited by IRS

2^{-13} probability that randomly sampled MIT grad is Nobel prize winner

2^{-19} struck by lightning in next year

2^{-28} probability of winning Mega Millions jackpot (now \$700m)

2^{-69} probability of all happening ..... (assuming independence)

2^{-128} ..... a billion billion times less likely than that.

How to construct CRHFs.

Two steps:

1. Construct CRHF $H_{small}: \{0,1\}^{2\lambda} \to \{0,1\}^{\lambda}$

↳ More art than science.
Come up with candidate, try to break it using known techniques, assume it's CRHF

↳ Current standard is SHA256, designed by NSA, published 2001

↳ Can also build from number theory (factoring, etc)
...but too slow

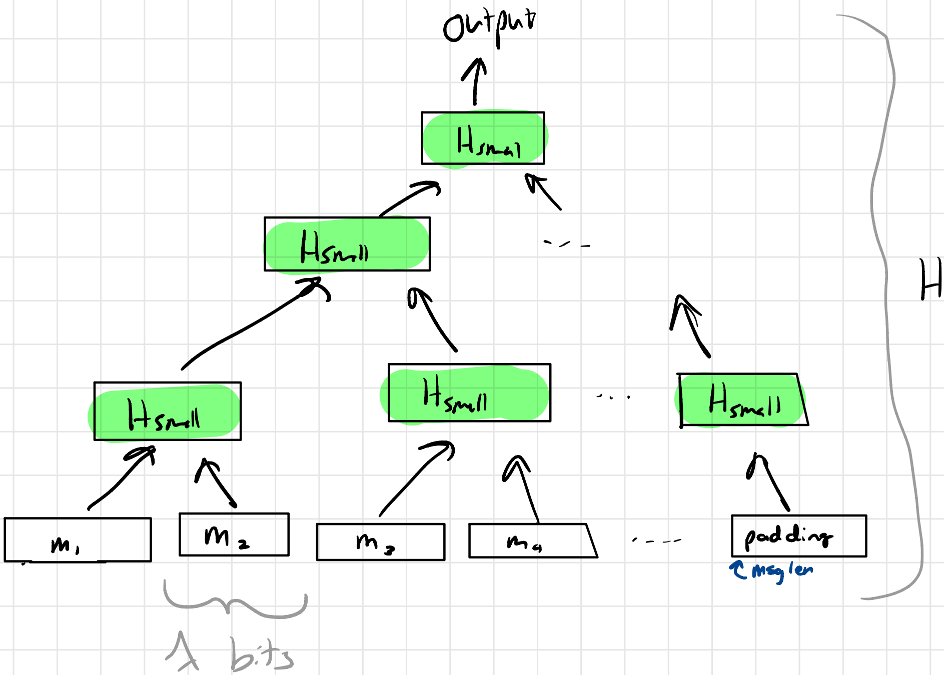
[Aside: If P=NP, CRHFs don't exist.
So security of CRHFs relies on unproven assumptions... P≠NP & more]

2. Use H_{small} to construct $H : \{0,1\}^* \to \{0,1\}^\lambda$

"Merkle-Damgard"

↳ H is CRHF if H_{small} is

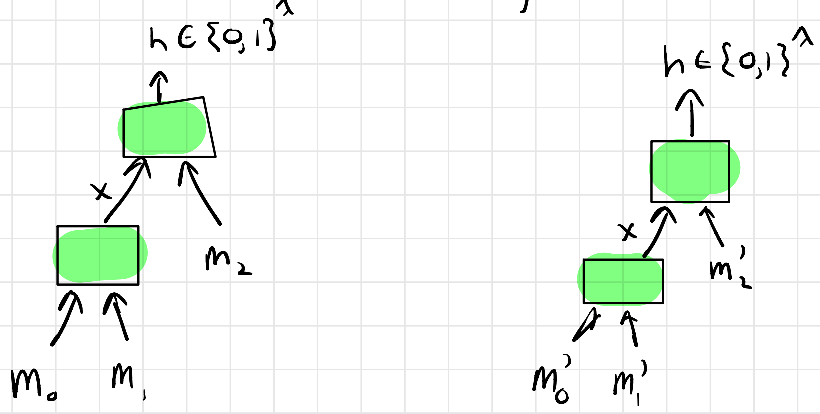
(No need for extra assumption)



* Need to be careful about padding... don't implement yourself!

Why Merkle-Damgard works

Consider hash fn $H_{big}: \{0,1\}^{3\lambda} \to \{0,1\}^{\lambda}$



Claim: If $\exists$ eff adv A that finds collisions in H_{big}, $\exists$ eff adv B that finds collisions in H_{small}

Run $A() \to [(m_0, m_1, m_2), (m_0', m_1', m_2')]$

$(x, m_2) \stackrel{?}{=} (x, m_2')$

NO: $(x, m_2), (x, m_2')$ is a collision for H_{small}!

YES: $(m_0, m_1) \stackrel{?}{=} (m_0', m_1')$

- NO: $(m_0, m_1), (m_0', m_1')$ is a collision for H_{small}!
- YES: Contradiction! $(m_0, m_1, m_2) = (m_0', m_1', m_2')$

Given hash fn with n-bit output, can find collision in time $O(2^{n/2})$.

[versus 2^n for brute-force search]

$\Rightarrow$ If you want adv to do 2^{128} work $\Leftarrow$ to find collision, need to have 256-bit output.

$\rightarrow$ In practice, we use SHA256 (or SHA3)

(on my laptop, get ~1 GB/s)
openssl speed sha256

Historical Note:

* For many years, MD5 (designed by Ron Rivest) was the standard CRHF — 128-bit output
* 2004 Wang et al find collision — time is now $\approx 2^{24}$
* We used to use SHA1 (160-bit output)
* In 2017 researchers at CWI Ams & Google found a collision in SHA1 using 2^{63} hashes

  $\approx$ 100,000x faster than brute force.
* Attack cost $\simeq$ \$100k - \$500k

$\Rightarrow$ SHA1 deprecated

COL. FINDER

(See Bellare textbook appendix)

Given: $H: \{0,1\}^* \to \{0,1\}^n$ [Model H as a random function]

Find: $m_0, m_1 \in \{0,1\}^{2n}$ s.t. $m_0 \neq m_1$, $H(m_0) = H(m_1)$.

Let $T = 2^{n/2}$

Choose distinct $r_1, r_2, r_3, \ldots, r_T \xleftarrow{R} \{0,1\}^{2n}$

Compute $H(r_1), H(r_2), \ldots, H(r_T)$.

↳ Likely to find a collision!

B_i = event that $\nexists$ collision after computing ith hash

$$\Pr[B_i \mid B_{i-1}] = 1 - \frac{i}{2^n}$$

$$\begin{aligned}
\Pr[\text{no collision}] &= \Pr[B_T] \\
&= \Pr[B_T \mid B_{T-1}] \cdot \Pr[B_{T-1}] \\
&\;\;\vdots \\
&= \prod_{i=1}^{T} \Pr[B_i \mid B_{i-1}] \\
&= \prod_{i=1}^{T} \left(1 - \frac{i}{2^n}\right) \\
&\leq \prod_{i=1}^{T} e^{-i/2^n} \\
&\leq \exp\left(-\sum_{i=1}^{T} \frac{i}{2^n}\right) \leq \exp\left(-\Omega\left(T^2/2^n\right)\right)
\end{aligned}$$

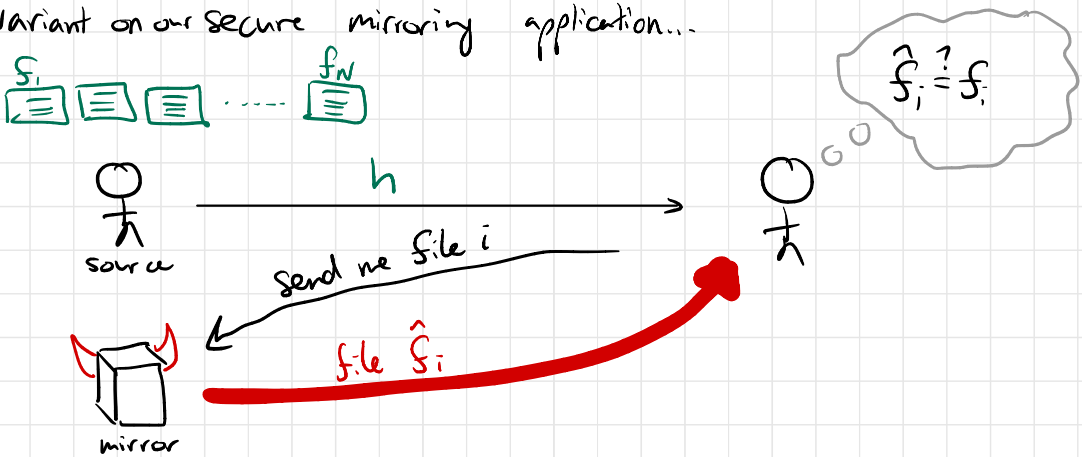
Useful life fact: $1 + x \leq e^x$

$\Pr[\text{collision}] \geq 1 - \text{constant}.$

repeat a few times

Application: Merkle trees (Authenticating many files with a single digest)

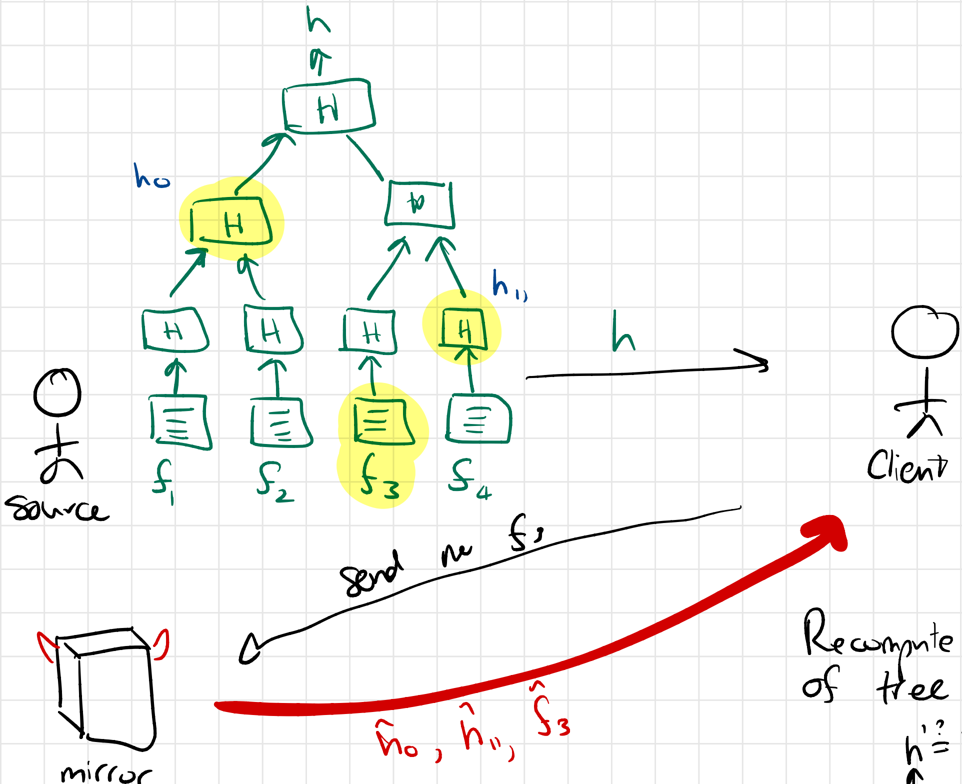
A variant on our secure mirroring application...



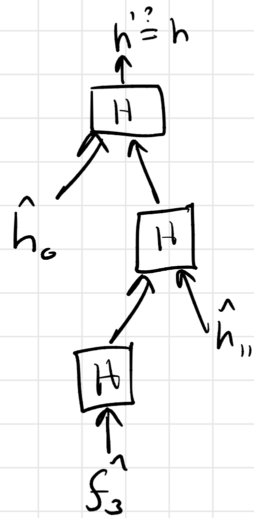
Option: Source sends N hashes
↳ a lot of communication over wide-area net

Option: Client downloads all N files
↳ even more communication!

Better idea: Use the Merkle construction



Recompute fraction of tree



$\Rightarrow$ Mirror sends one full file + $O(\log N)$ hashes

$\ll$ than N hashes !
$\ll$ than N files

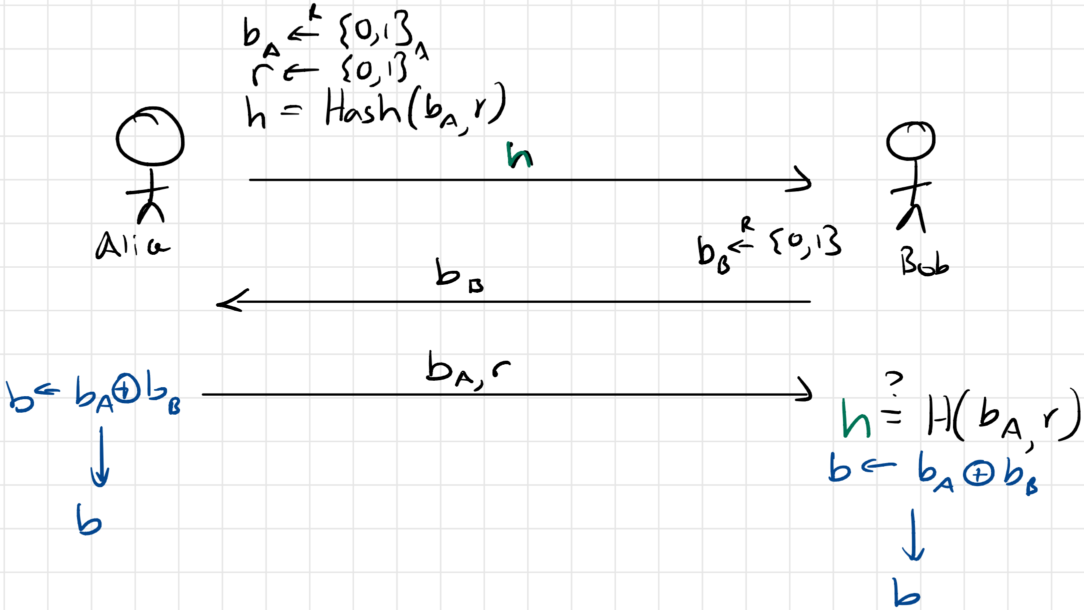
$\Rightarrow$ CR property ensures that mirror can't cheat

Application: Commitments

* "Sealed envelope" with cryptography.
* Just a small tweak to the earlier applications
* Requires a bit more than plain CRHF, but any CRHF can be made suitable ....

[Halevi Micali '96]

"Coin flipping"



Modulo Alice refusing to open, neither party can control bit b.

↳ Distributed randomness used for protocols that require good randomness w/o trustworthy dealer (e.g. lotto, ...)

The End