Lecture 3 - Collision Resistance

6.1600 - fall 2023

Corrigan-Gibbs & Zeldovich

MIT

Collision-Resistant Hash Functions

Plan

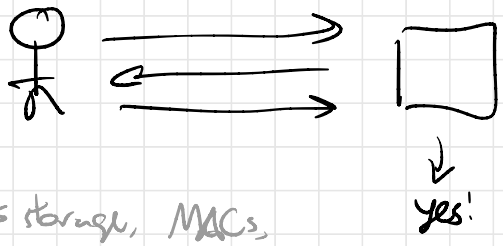
* Intuition & motivation
* Defn of CRHF
* Constructions
* Attacks
* If time: applications.

Logistics

* Lab 0 code & Lab 0 theory due tomorrow
* Lab 1 out on Friday.

Last time...

authenticating PEOPLE



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

Today...

authenticating FILES / CODE / DATA

Main new tool:

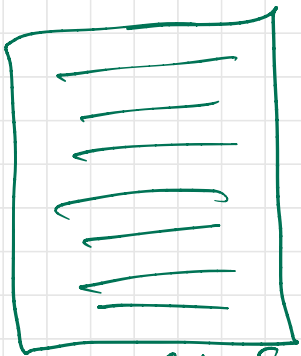
Collision-resistant hash functions (CRHF)

Intuition behind CRHF....

Compressing!

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

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

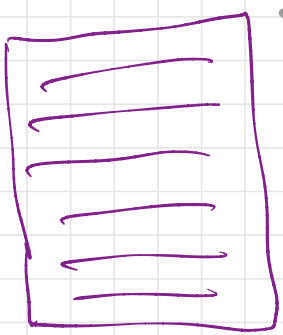


Big file f
(gigabytes or more)

"unique" 256-bit hash

$h = H(f)$

Like a short fingerprint of the big file f.



file f'

different hash

$h' = H(f')$

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

There are many properties we can demand of a hash fn...

$$H: \{0,1\}^{512} \to \{0,1\}^{256}$$

$$H: X \to Y$$

One-way Given $y = H(x)$, for $x \xleftarrow{R} X$,
"hard to find" x' s.t. $H(x') = y$.

Second preimage resistant

Given $x \xleftarrow{R} X$, hard to find x' s.t.
$H(x) = H(x')$.

Collision resistant

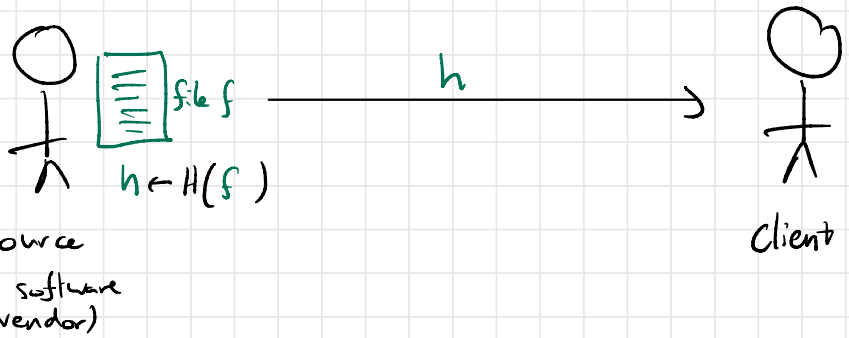
Hard to find (x, x') s.t. $H(x) = H(x')$.

Stronger security prop.

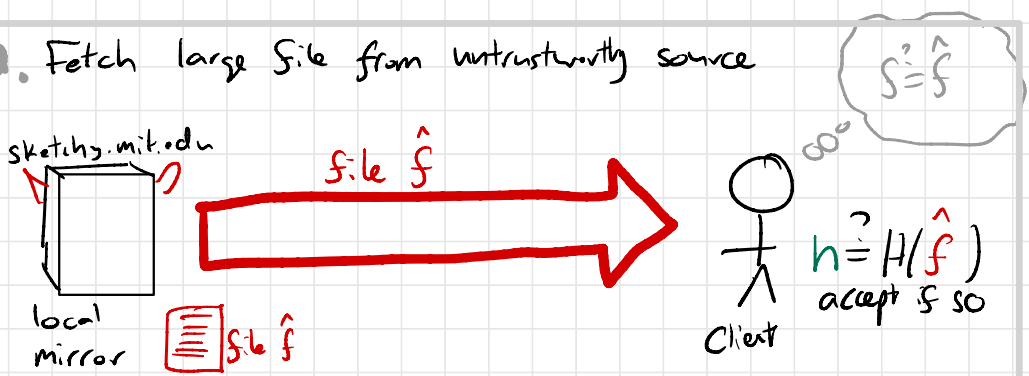
Standard crypto hash fns satisfy all of those.

Application I: Secure mirroring

1. Get hash from trustworthy source



2. Fetch large file from untrustworthy source



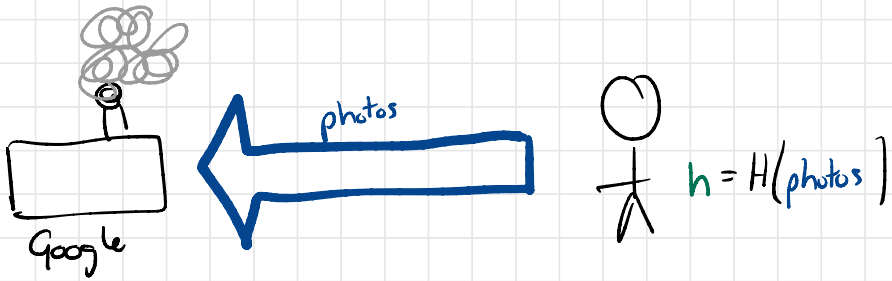
If Hash is CRHF, then sketchy mirror will not be able to find a file $\hat{f} \neq f$ that client will accept.

Used in subresource integrity (SRI)!

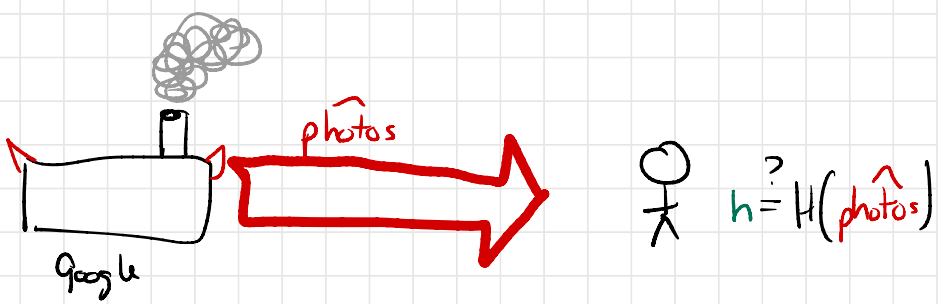
```
<script src="sketchy.mit.edu/code.js"
        integrity="sha256-oqAB....." >
```

Application II: Outsourced File Storage

1.



2.

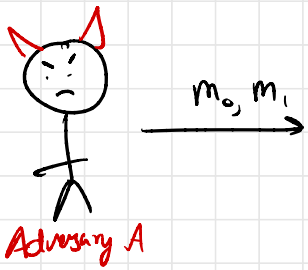


If hash is CRHF, then Google can't trick you into accepting incorrect photos/files.

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

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

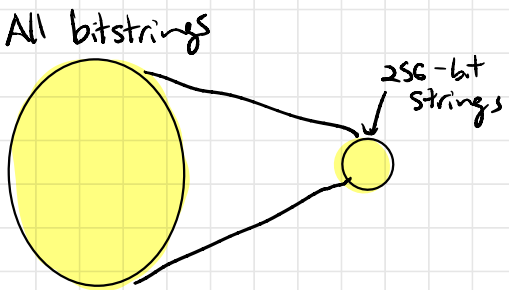
Adversary's goal in breaking CRHF



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

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

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

In theory: λ = "security parameter" (≈ key length)

"efficient" = randomized alg runs in time $\text{poly}(\lambda)$

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

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

In practice: $\lambda = 128, 256, 384$

"efficient" algorithm ≈ "fast enough"

"negl" ≈ 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^{43}	ops/sec on PS5
2^{58}	ops/sec on Fugaku supercomputer ($\approx \$1$ billion)
2^{68}	hashes/sec computed by Bitcoin miners
2^{92}	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

Probability

2^{-1}	fair coin lands heads
2^{-8}	tax returns audited by IRS
2^{-19}	dealt a royal flush in poker game
2^{-28}	probability of winning Mega Millions jackpot
2^{-56}	probability of winning Mega Millions twice in a row
2^{-128}	... a billion billion times less likely than that.

Lenstra
Kleinjung
Thome

Slightly misleading in that attacker can play lotto's quickly

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]

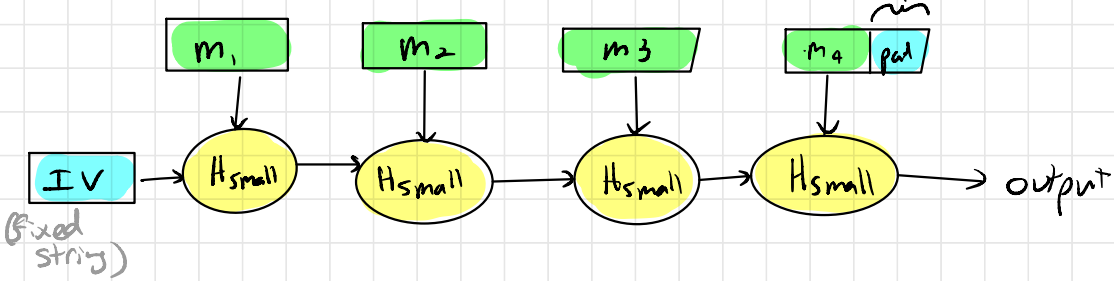
2. Use H_{small} to construct $H: \{0,1\}^* \to \{0,1\}^\lambda$

"Merkle-Damgard"

↳ H is CRHF if H_{small} is
(No need for extra assumption)

actually limited in standard

100 000 <len>



Why padding block?

* Make input a multiple of block size.
* Avoid certain precomputation attacks.

Why IV?

* W/o IV and w/o length ⇒ Insecure!
* H_{small} can be weaker than CRHF and still H can be CRHF (insurance against bad H_{small})

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

  $\Rightarrow$ SHA1 deprecated

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

↳ versus 2^n for brute-force search

"Birthday Paradox"

If you throw $\sqrt{N}$ balls independently & uniformly at random into N bins, you will have 2 balls in 1 bin w.p. $\approx \frac{1}{4}$.

⇒ To find collision, hash $2^{n/2}$ random strings. By an argument involving R.P.P. / Birthday, ∃ colliding pair often

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

↳ In practice, we use SHA256 (or SHA3)

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

openssl speed sha256

Domain Separation

Given one-input CRHF $H: \{0,1\}^* \to \{0,1\}^{256}$,

often want to build two-input CRHF:

$$H_2(x, y).$$

BAD IDEA:

$$H_2(x,y) := H(x \,\|\, \text{"string const"} \,\|\, y)$$

Notice: $H_2(\text{"key"}, \text{"value"})$

$= H_2(\text{"ke"}, \text{"yvalue"})$

→ Even though H is CRHF, H_2 is not!

action=create & user = nickol

Same hash value! [action=create&user = action delete user nickola:

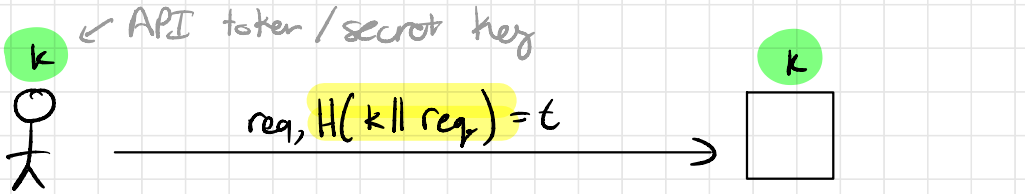
[a=ctioncreateuser & action=delete & user = nickola:

Flickr and Amazon EC2 APIs were vulnerable to this attack!

Better idea: Unambiguous encoding (e.g. (length, val, length, val, ...))

Beware! A CRHF is only a CRHF.

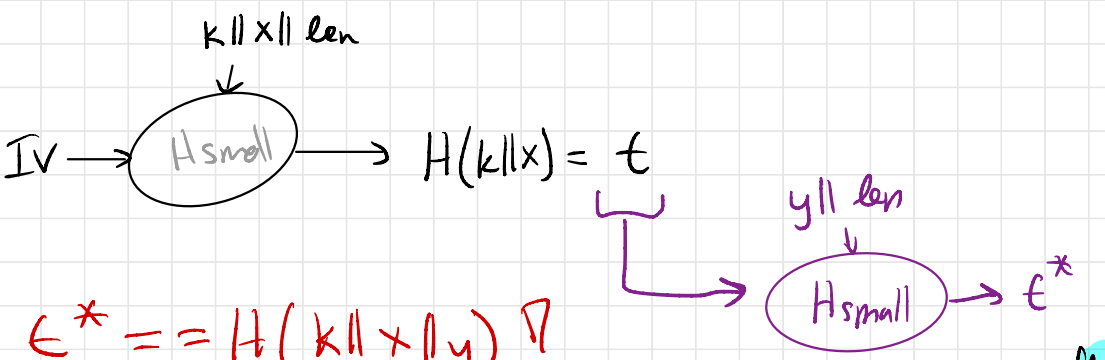
A bad authentication scheme



Implicitly assumes that given $H(k\|x)$, it's "hard to find" $H(k\|x\|y)$

$k, r \xleftarrow{r} \{0,1\}^{128}$

Not implied by CR! In fact, false for SHA2.

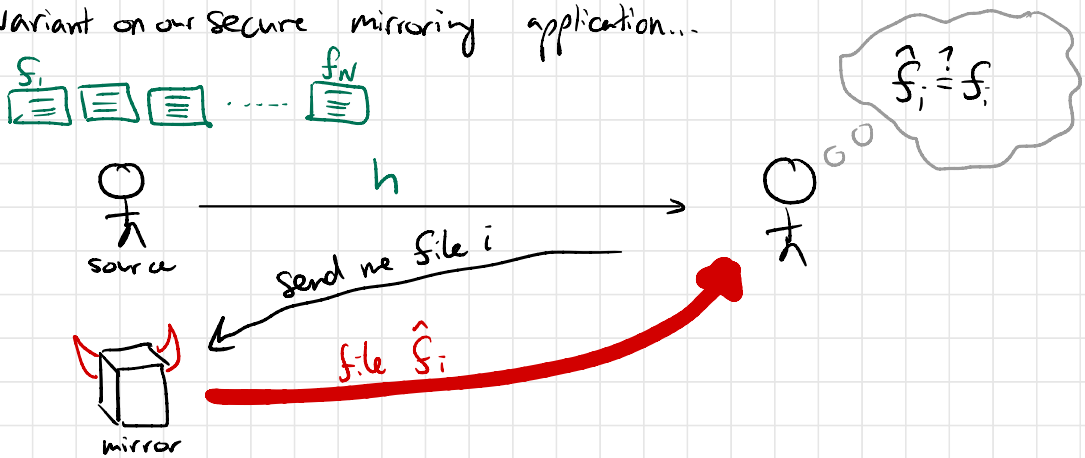


$t^* == H(k\|x\|y)$!!

We will discuss the Right Way to do this (MAC) later on.

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

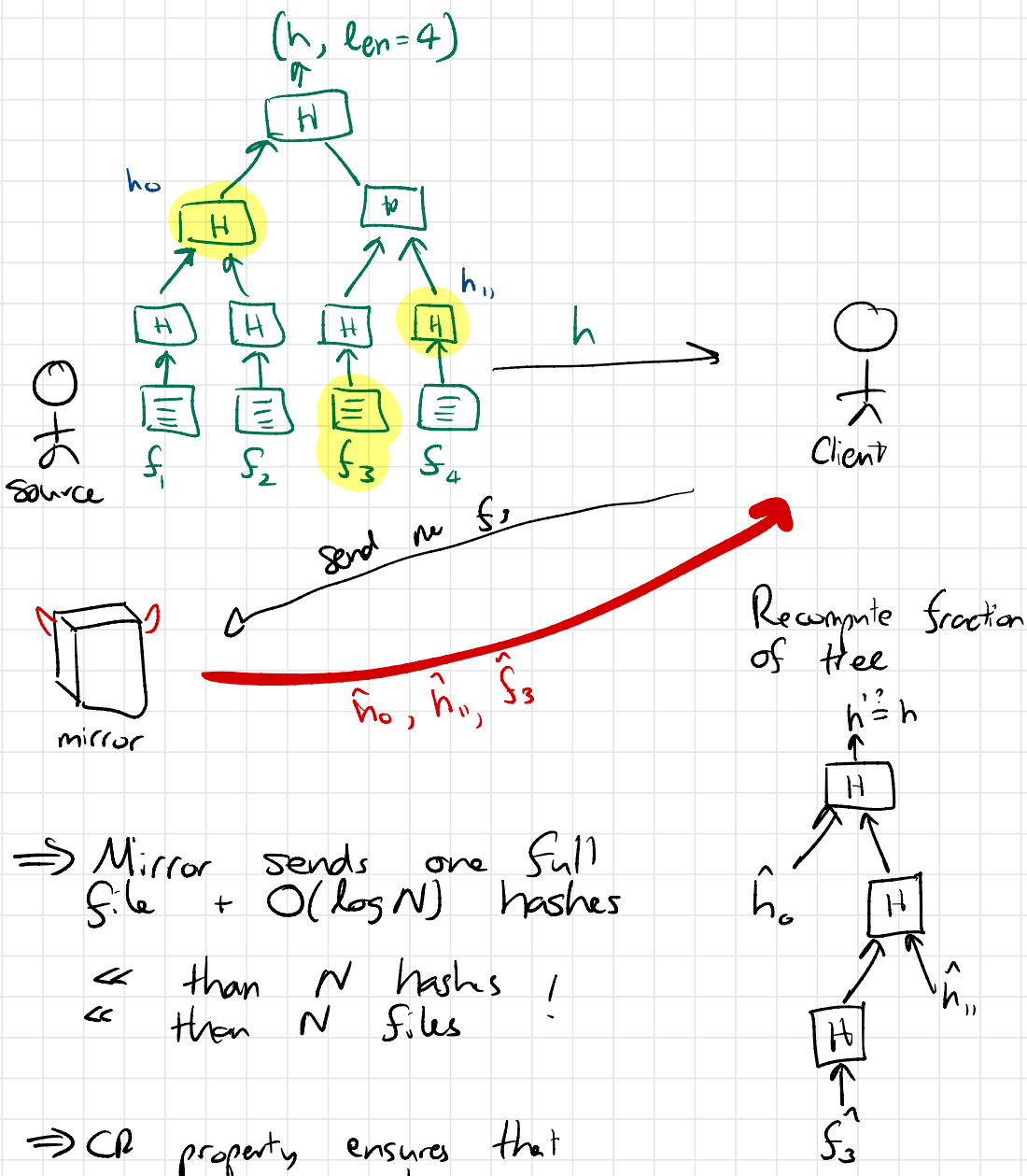
A variant on our secure mirroring application...



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



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

≪ than N hashes !
≪ than N files !

⇒ CR property ensures that mirror can't cheat

Used in certificate transparency, ...

The End

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 ∄ collision after computing ith hash

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

$$\Pr[\text{no collision}] = \Pr[B_T]$$

$$= \Pr[B_T \mid B_{T-1}] \cdot \Pr[B_{T-1}]$$

$$\vdots$$

$$= \prod_{i=1}^{T} \Pr[B_i \mid B_{i-1}]$$

$$= \prod_{i=1}^{T} \left(1 - \frac{i}{2^n}\right)$$

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

$$\leq \prod_{i=1}^{T} e^{-i/2^n}$$

$$\leq \exp\left(-\sum_{i=1}^{T} \frac{i}{2^n}\right) \leq \exp\left(-\Omega\left(T^2/2^n\right)\right)$$

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

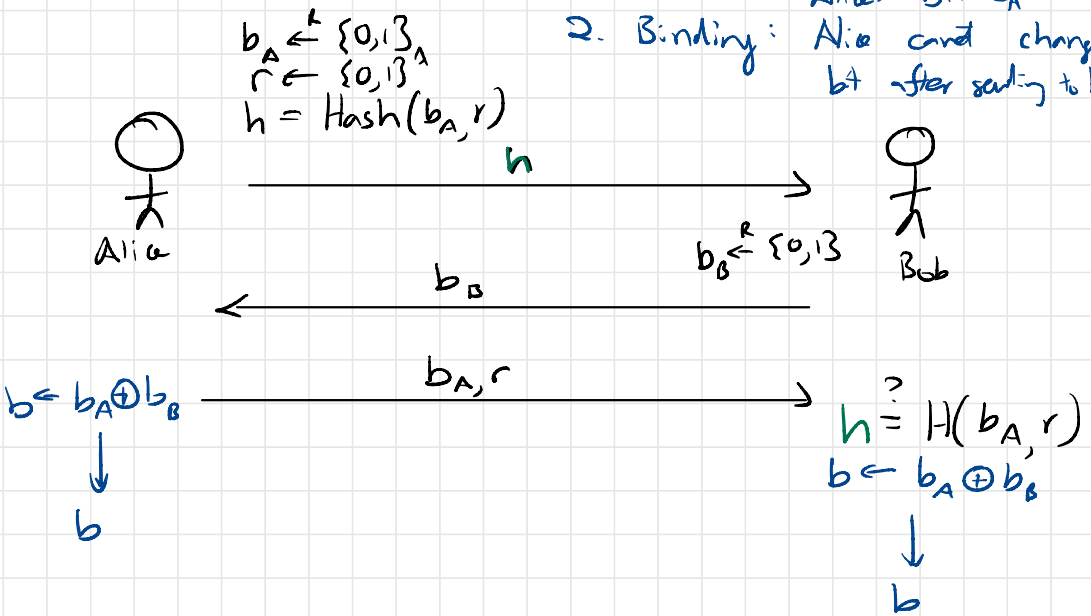
↳ repeat a few times

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"

1. Hiding: $H(m,r)$ "hides" Alice's bit b_A *
2. Binding: Alice can't change bit after sending to Bob



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, ...)