COS433/Math 473: Cryptography

Mark Zhandry

Princeton University

Spring 2017

Previously…

Encryption

+

Authentication

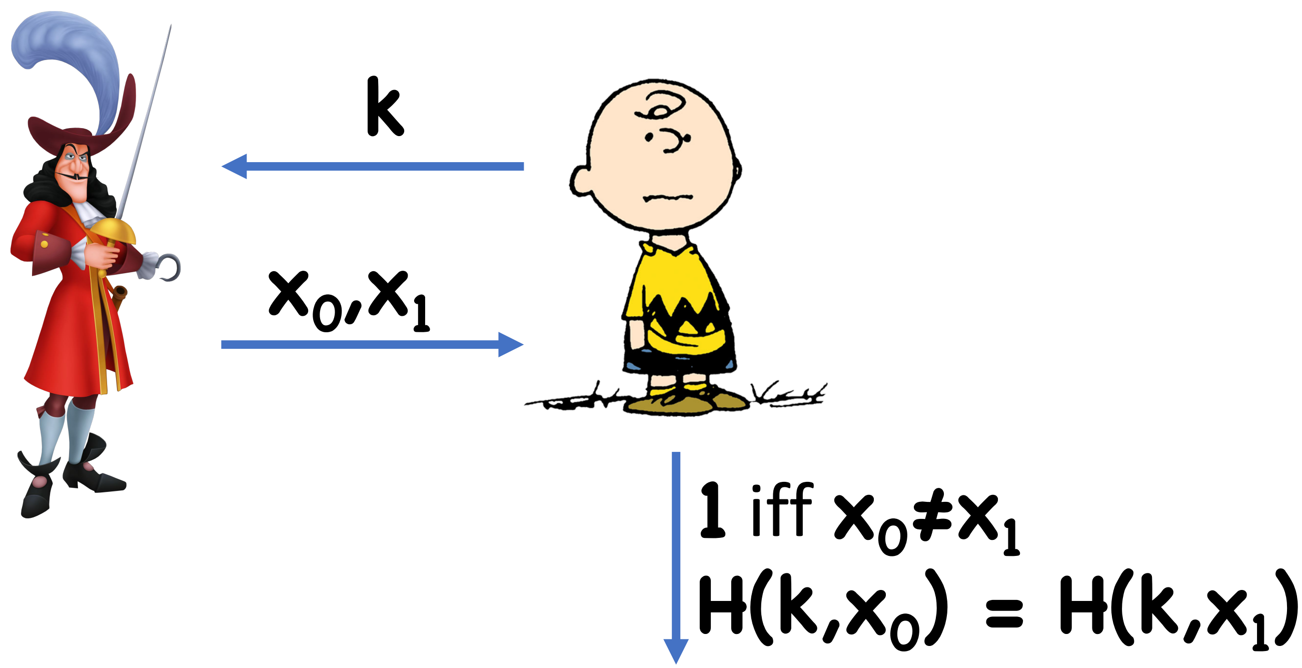
=

Authenticated Encryption

Collision Resistance

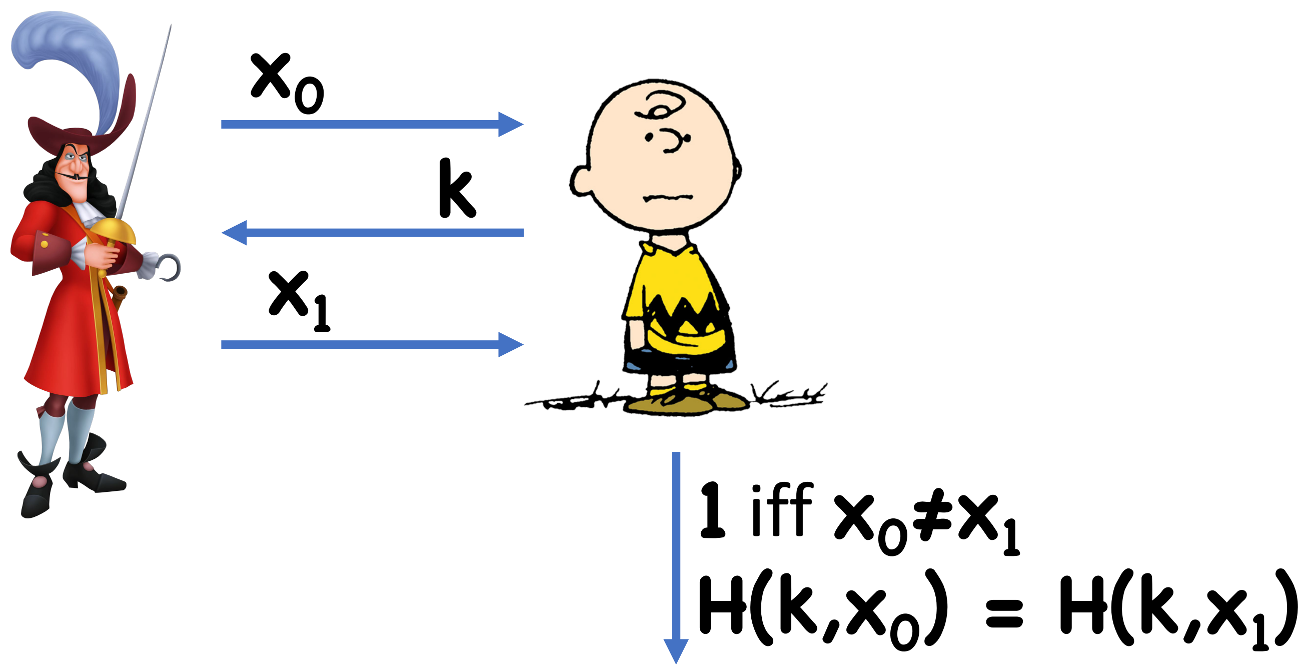
Security Notions for Hashing

Collision resistance as a game:



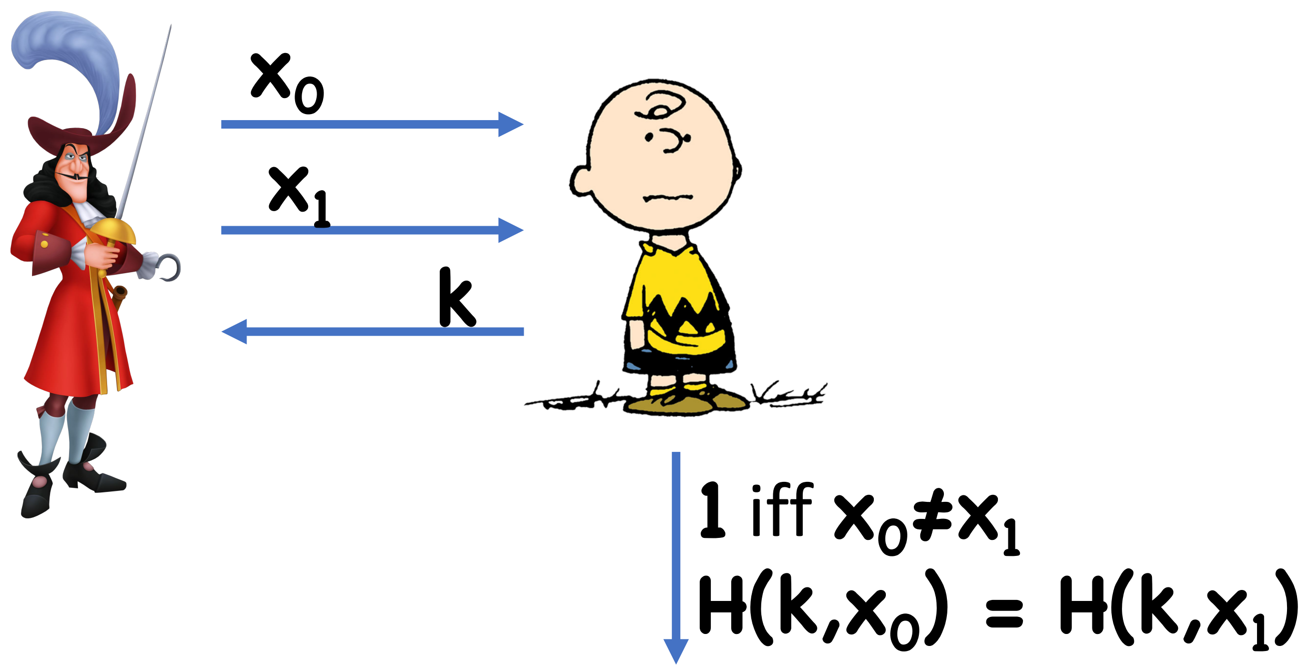
Security Notions for Hashing

2nd Preimage Resistance (or target collision resistance):



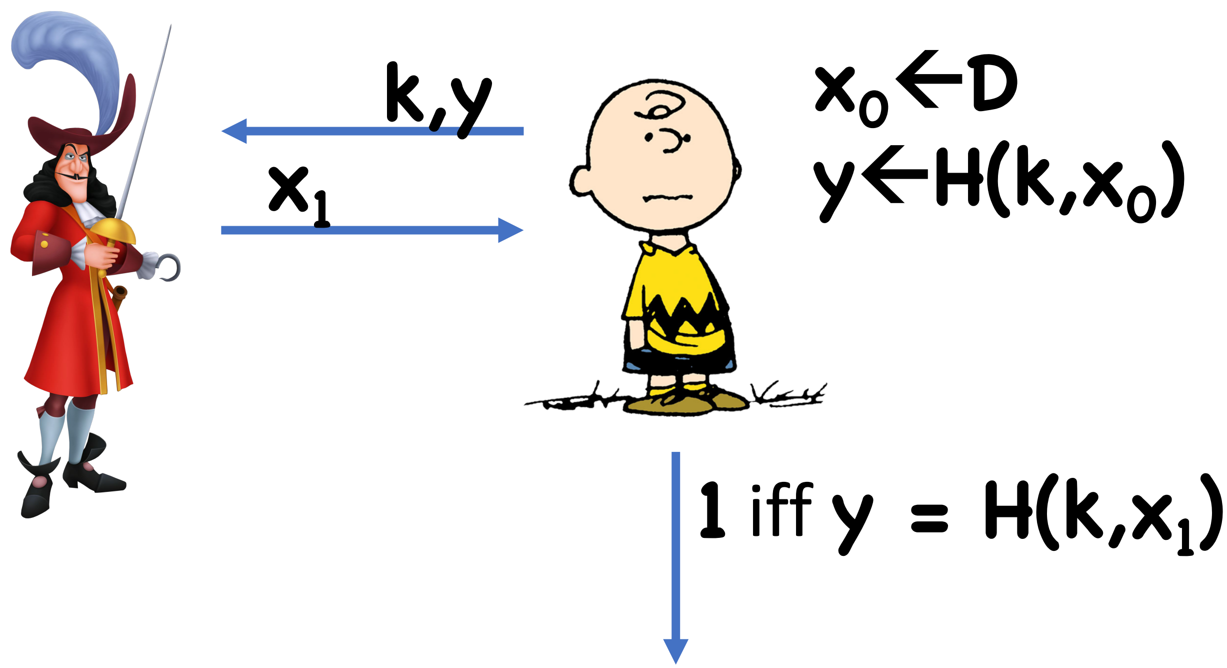
Security Notions for Hashing

2-Universal:



Security Notions for Hashing

One-wayness (or pre-image resistance):



Implications

Collision Resistance

⇓

2^{nd} Pre-image Resistance

⇓

One-wayness

Random Oracle Model

Pretend **H** is a truly random function

Everyone can query **H** on inputs of their choice
- Any protocol using **H**
- The adversary (since he knows the key)

A query to **H** has a time cost of 1

Today

Commitment Schemes

Start: number-theoretic constructions of symmetric key primitives

Remember Galileo

- Galileo observed the rings of Saturn, but mistook them for two moons

- Galileo wanted extra time for verification, but not to get scooped
- Circulates anagram
  SMAISMRMILMEPOETALEUMIBUNENUGTTAUIRAS
- When ready, tell everyone the solution:
  altissimum planetam tergeminum observavi

("I have observed the highest planet tri-form")

Commitment Scheme

Different than encryption

- No need for a decryption procedure
- No secret key
- But still need secrecy (“hiding”)
- Should only be one possible opening (“binding”)
- Sometimes other properties needed as well

(Non-interactive) Commitment Syntax

Message space **M**
Ciphertext Space **C**
(suppressing security parameter)

Com(m; r): outputs a commitment **c** to **m**

Commitments with Setup

Message space **M**
Ciphertext Space **C**
(suppressing security parameter)

Setup(): Outputs a key **k**
Com(k, m; r): outputs a commitment **c** to **m**

Using Commitments

Commit Stage

$r \leftarrow R$

$c \leftarrow Com(m;r)$

m

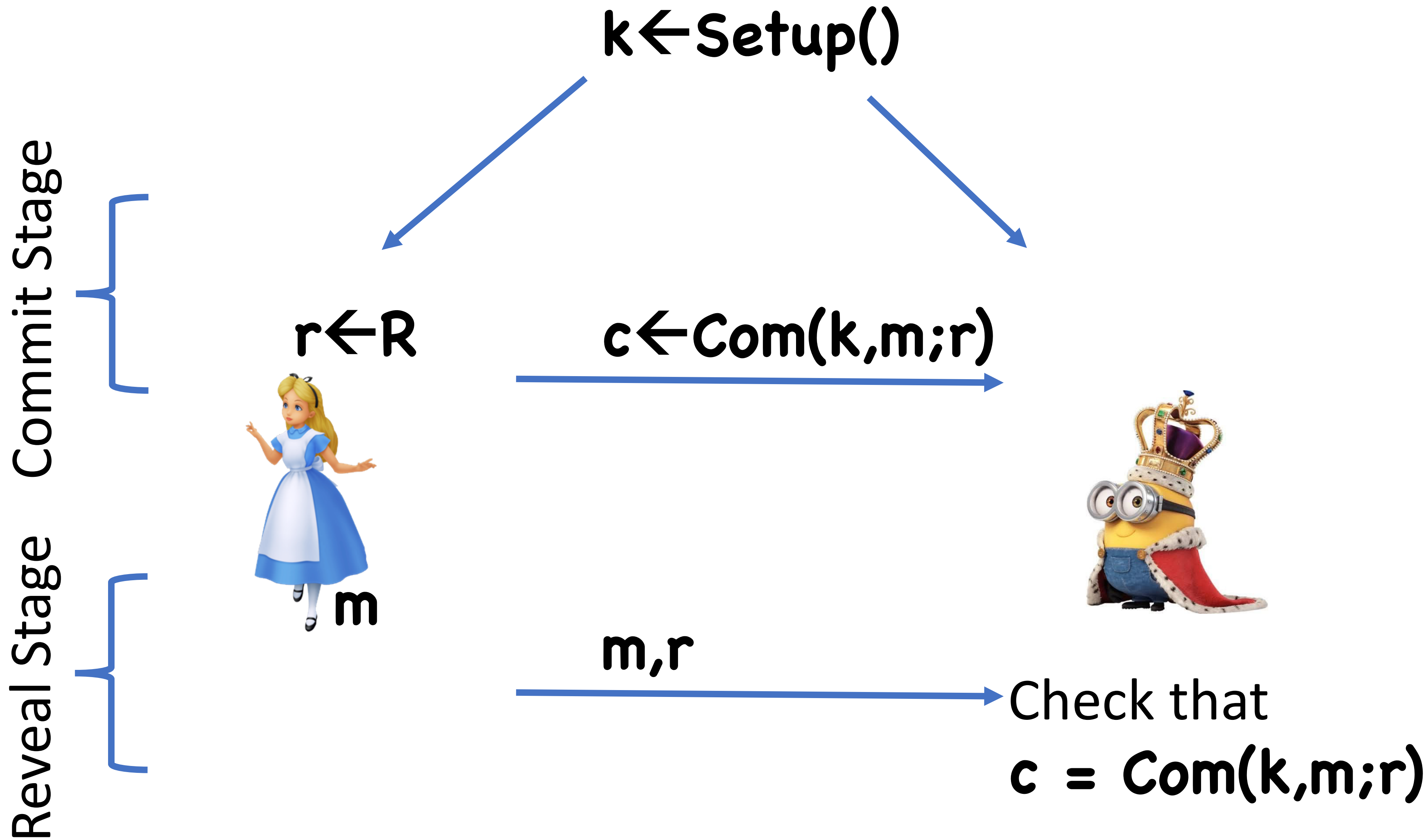
Reveal Stage

m,r

Check that

$c = Com(m;r)$

Using Commitments (with setup)



Security Properties

Hiding: **c** should hide **m**

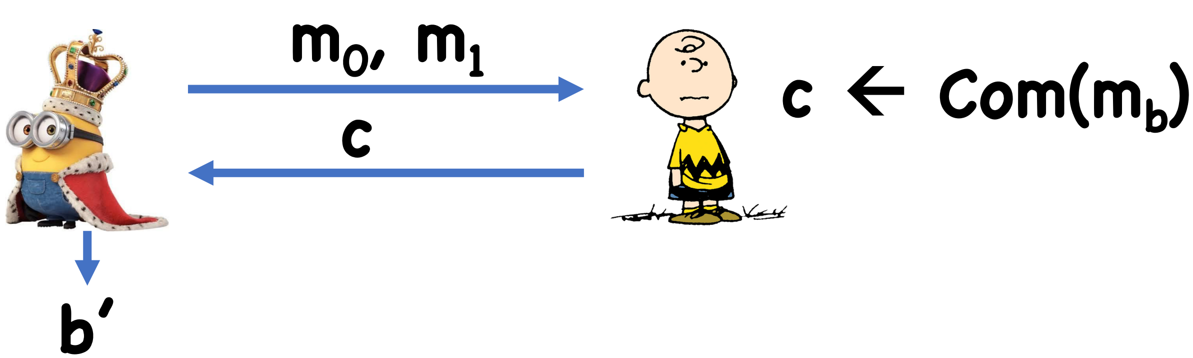
- Perfect hiding: for any $\mathbf{m_0}$, $\mathbf{m_1}$,

$$\mathrm{Com}(m_0) \overset{d}{=} \mathrm{Com}(m_1)$$

- Statistical hiding: for any $\mathbf{m_0}$, $\mathbf{m_1}$,

$$\Delta(\ \mathrm{Com}(m_0), \mathrm{Com}(m_1)\ ) < \mathrm{negl}$$

- Computational hiding:



Security Properties (with Setup)

Hiding: **c** should hide **m**

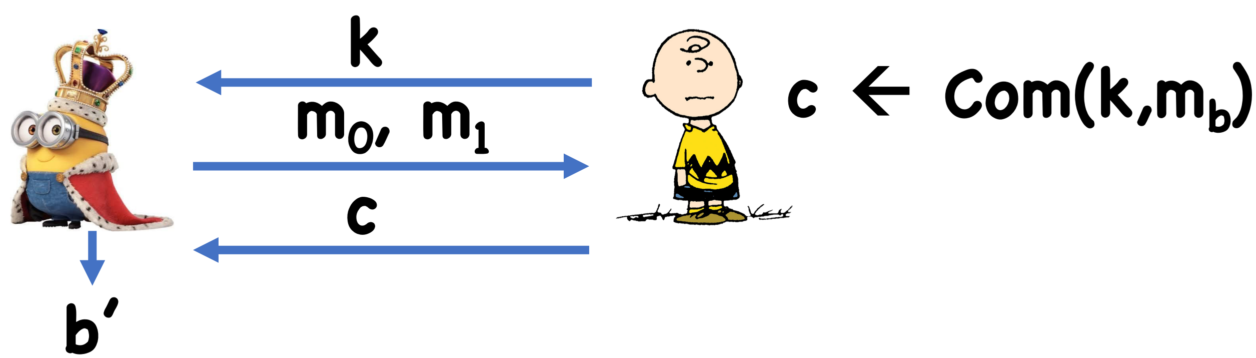
- Perfect hiding: for any m_0, m_1,

$$k, Com(k, m_0) \stackrel{d}{=} k, Com(k, m_1)$$

- Statistical hiding: for any m_0, m_1,

$$\Delta(\ [k, Com(k, m_0)],\ [k, Com(k, m_1)]\ ) < negl$$

- Computational hiding:



Security Properties

Binding: Impossible to change committed value

- Perfect binding: For any **c**, $\exists$ at most a single **m** such that **c = Com(m;r)** for some **r**

- Computational binding: no PPT adversary can find $(m_0,r_0),(m_1,r_1)$ such that $Com(m_0;r_0)=Com(m_1;r_1)$

Security Properties (with Setup)

Binding: Impossible to change committed value

- Perfect binding: For any **k,c**, $\exists$ at most a single **m** such that **c = Com(k,m;r)** for some **r**

- Statistical binding: except with negligible prob over **k**, for any **c**, $\exists$ at most a single **m** such that **c = Com(k,m;r)** for some **r**

- Computational binding: no PPT adversary, given **k←Setup()**, can find $(m_0,r_0),(m_1,r_1)$ such that $Com(k,m_0;r_0)=Com(k,m_1;r_1)$

Who Runs Setup()

Trusted third party (TTP)?

Alice?

- Must ensure that Alice cannot devise **k** for which she can break binding
- If binding holds, can actually devise scheme Com' without setup

Bob?

- Must ensure Bob cannot devise **k** for which he can break hiding

Honest-but Curious vs Malicious

Honest-but Curious receiver: runs **Setup** as expected, tries to learn committed message

Malicious receiver: can generate **k** however he wants, tries to learn message

Anagrams as Commitment Schemes

Com(m) = sort characters of message

Problems?

- Not hiding: “Jupiter has four moons” vs “Jupiter has five moons”

- Not binding: Kepler recodes Galileo’s anagram to conclude Mars has two moons

Anagrams as Commitment Schemes

Com(m) = add random superfluous text, then sort characters of message

Might still not be hiding

- Need to guarantee, for example that expected number of each letter in output is independent of input string

Still not binding…

Other Bad Commitments

Com(m) = m

- Has binding, but no hiding

Com(m;r) = m⊕r

- Has hiding, but no binding

Can a commitment scheme be both statistically hiding and statistically binding?

A Simple Commitment Scheme

Let **H** be a hash function

Com(m;r) = H(m || r)

Binding?

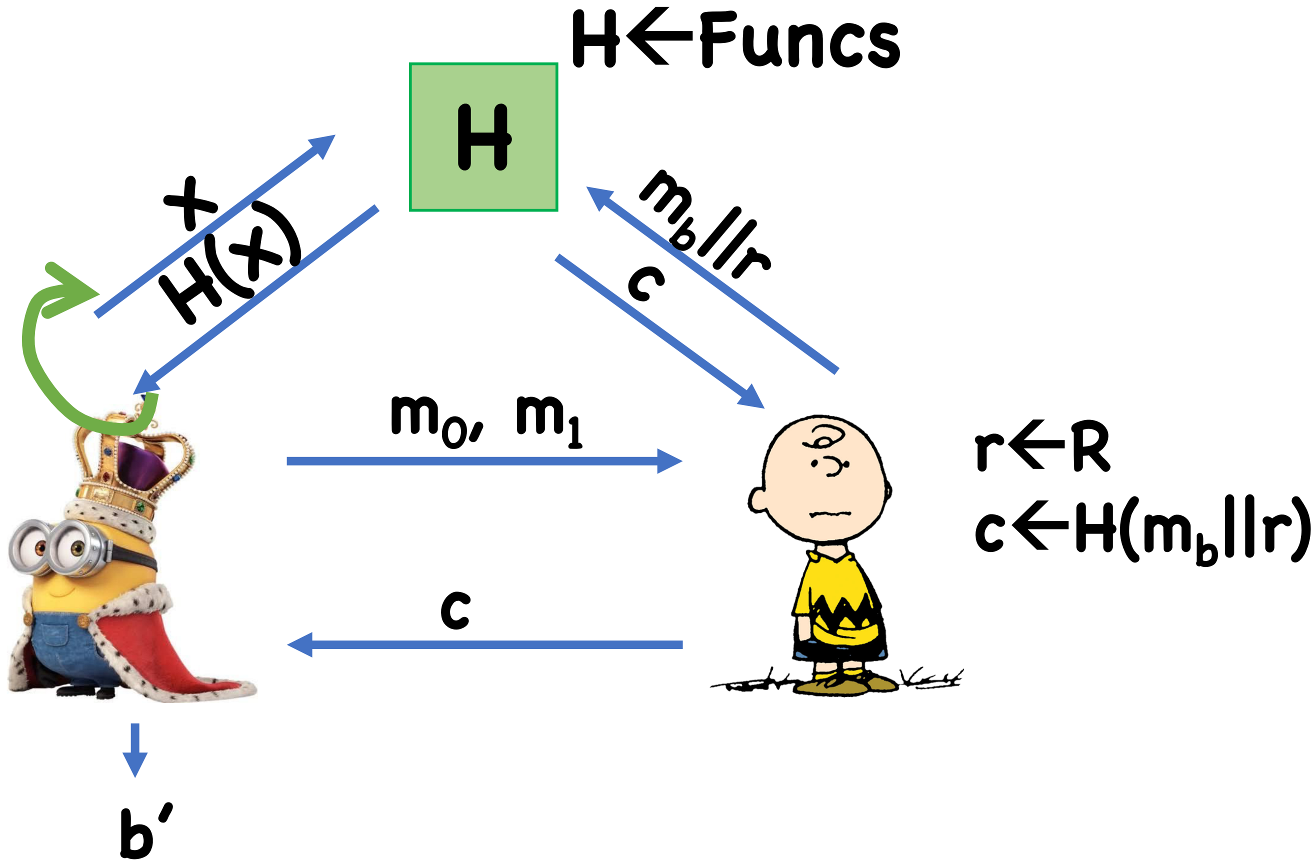
Hiding?

Theorem: $\mathbf{Com(m;r) = H(m||r)}$ has:

- Perfect binding assuming **H** is injective
- Computational binding assuming **H** is collision resistance (implied by RO)

- Computational hiding in the Random Oracle Model

Hiding



Proof of Hiding

Suppose  never queries **H** on $\mathbf{m_b||r}$

Then all query answers and commitment c seen by  are independent uniform strings

-  has no chance of determining **b**

Probability  queries on $\mathbf{m_b||r}$?

- At most $\mathbf{q/|R|}$ = negligible

“Standard Model” Commitments?

Random oracle model proof is heuristic argument for security

Can we prove it under assumptions such as collision resistance, etc?

Single Bit to Many Bit

Let **(Setup,Com)** be a commitment scheme for single bit messages

Let $\textbf{Com'}(k,m;\ r)=(\textbf{Com}(k,m_1;r_1),\ldots,\textbf{Com}(k,m_t;r_t))$

- $m = (m_1,\ldots,m_t),\ m_i \in \{0,1\}$
- $r = (r_1,\ldots,r_t),\ r_i$ are randomness for **Com**

Theorem: If **(Setup,Com)** is perfectly/statistically/computationally binding, then so is **(Setup,Com')**

Theorem: If **(Setup,Com)** is perfectly/statistically/computationally, semi-honest/malicious hiding, then so is **(Setup,Com')**

Binding

Suppose  breaks (say comp) biding of **Com'**

Given **k**, produces $(m_1^0,r_1^0),\dots,(m_t^0,r_t^0),(m_1^1,r_1^1),\dots,(m_t^1,r_t^1)$ such that

- $(m_1^0,\dots,m_t^0) \neq (m_1^1,\dots,m_t^1)$
- $\mathrm{Com}(k,m_i^0;r_i^0) = \mathrm{Com}(k,m_i^1;r_i^1)$ for all **i**

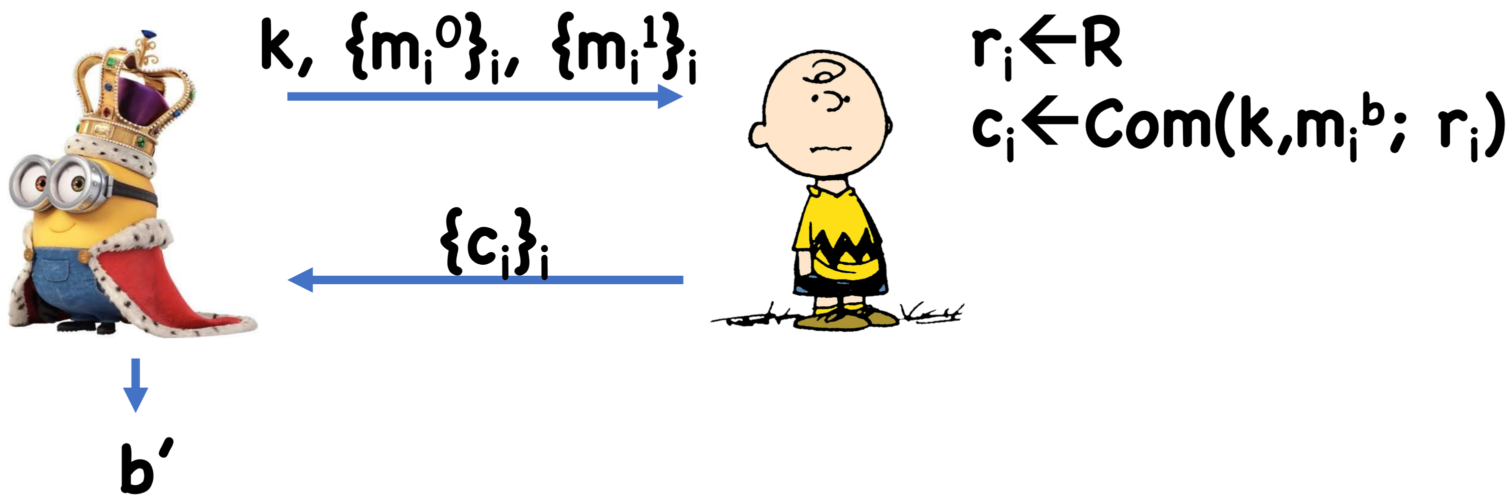
Therefore, $\exists\, i$ such that $m_i^0 \neq m_i^1$ but

$$\mathrm{Com}(k,m_i^0;r_i^0) = \mathrm{Com}(k,m_i^1;r_i^1)$$

$\Rightarrow$ Break binding of **Com**

Hiding

Suppose  breaks (say, computational malicious) hiding



Hiding

Proof by Hybrids

Hybrid **j**:

- For each $i \leq j$, $c_i = \mathrm{Com}(k, m_i^1, r_i)$
- For each $i > j$, $c_i = \mathrm{Com}(k, m_i^0, r_i)$

Hybrid **0**: commit to $\{m_i^0\}_i$
Hybrid **t**: commit to $\{m_i^1\}_i$

$\exists$ **j** such that  distinguishes Hyb **j-1** from Hyb **j**
$\Rightarrow$ break hiding of **Com**

Single Bit to Many Bit

Let **(Setup,Com)** be a commitment scheme for single bit messages

Let $\mathbf{Com'(k,m;\ r)=(Com(k,m_1;r_1),\ldots,Com(k,m_t;r_t))}$

- $\mathbf{m = (m_1,\ldots,m_t),\ m_i \in \{0,1\}}$
- $\mathbf{r = (r_1,\ldots,r_t),\ r_i}$ are randomness for **Com**

Therefore, suffices to focus on commitments for single bit messages

Statistically Hiding Commitments

Let **H** be a collision resistant hash function with domain **X={0,1}×R** and range **Z**

Setup(): k←K, output **k**
Com(k, m; r) = H(k, (m,r))

Binding?

Hiding?

Statistically Hiding Commitments

Let **F** be a pairwise independent function family with domain **X={0,1}×R** and range **Y**

Let **H** be a collision resistant hash function with domain **Y** and range **Z**

Setup(): f←F, k←K, output **(f,k)**
Com((f,k), m; r) = H(k, f(m,r))

Theorem: If **|Y|/|X|** is "sufficiently large" and **H** is collision resistant, then **(Setup,Com)** has computational binding

Theorem: If **|X|** is "sufficiently large", then **(Setup,Com)** has statistical hiding

Theorem: If $|Y|/|X|$ is "sufficiently large" and H is collision resistant, then **(Setup,Com)** has computational binding

Proof:

- Suppose $|Y| > |X|^2 \times 2^\lambda$
- For any $x_0 \neq x_1$, $\Pr[f(x_0)=f(x_1)] < 1/(|X|^2 \times 2^\lambda)$
- Union bound:

$$\Pr[\exists\ x_0 \neq x_1 \text{ s.t. } f(x_0)=f(x_1)] < 1/2^\lambda$$

Theorem: If **|X|** is "sufficiently large", then **(Setup,Com)** has statistical hiding

Goal: show **(f, k, H(k, f(0,r)))** is statistically close to **(f, k, H(k, f(1,r)))**

Min-entropy

Definition: Given a distribution $\mathbf{D}$ over a set $\mathbf{X}$, the min-entropy of $\mathbf{D}$, denoted $\mathbf{H_\infty(D)}$, is

$$-\min_x \log_2(\Pr[x \leftarrow D])$$

Examples:

- $H_\infty(\{0,1\}^n) = n$
- $H_\infty($ random $\mathbf{n}$ bit string with parity $\mathbf{0}) = ?$
- $H_\infty($ random $\mathbf{i>0}$ where $\Pr[i] = 2^{-i}) = ?$

Leftover Hash Lemma

Lemma: Let **D** be a distribution on **X**, and **F** a family of pairwise independent functions from **X** to **Y**. Then

$$\Delta(\ (f,\ f(D))\ ,\ (f,\ R)\ ) \le \varepsilon$$ where

- $f \leftarrow F$
- $R \leftarrow Y$
- $\log |Y| \le H_\infty(D) + 2 \log \varepsilon$

"Crooked" Leftover Hash Lemma

Lemma: Let **D** be a distribution on **X**, and **F** a family of pairwise independent functions from **X** to **Y**, and **h** be any function from **Y** to **Z**. Then

$$\Delta(\ (f,\ h(f(D)))\ ,\ (f,\ h(R))\ )\ \leq\ \varepsilon \text{ where}$$

- $f \leftarrow F$
- $R \leftarrow Y$
- $\log |Z| \leq H_\infty(D) + 2 \log \varepsilon - 1$

Theorem: If $|X|$ is "sufficiently large", then $(Setup,Com)$ has statistical hiding

Goal: show $(f,\ k,\ H(k,\ f(0,r))\ )$ is statistically close to $(f,\ k,\ H(k,\ f(1,r))\ )$

Suppose $|Z| = 2^\lambda$

$(0,r)$ has min-entropy $\log |R|$

Set $R = \{0,1\}^{3\lambda}$, $\varepsilon = 2\times 2^{-\lambda}$

Then $\log |Z| \leq H_\infty(D) + 2 \log \varepsilon - 1$

Theorem: If **|X|** is "sufficiently large", then **(Setup,Com)** has statistical hiding

For any **k**,

$$\Delta(\ (f,\ H(k,\ f(0,r)))\ ,\ (f,\ H(k,\ U))\ )\ \leq\ \varepsilon$$

Thus

$$\Delta(\ (f,\ H(k,\ f(0,r)))\ ,\ (f,\ H(k,\ f(1,r)))\ )\ \leq\ 2\varepsilon$$

Therefore

$$\Delta(\ (f,\ k,\ H(k,\ f(0,r))\ )\ ,\ (f,\ k,\ H(k,\ f(1,r))\ )\ \leq\ 2\varepsilon$$

Statistically Binding Commitments

Let **G** be a PRG with domain $\{0,1\}^\lambda$, range $\{0,1\}^{3\lambda}$

Setup(): choose and output a random 3λ-bit string **k**

Com(b; r): If **b=0**, output **G(r)**, if **b=1**, output $G(r) \oplus k$

Theorem: **(Setup,Com)** is statistically binding

Theorem: If **G** is a secure PRG, then **(Setup,Com)** has computational hiding

Theorem: If **G** is a secure PRG, then **(Setup,Com)** has computational hiding

Hybrids:

- Hyb 0: $\mathbf{S = Com(0;r) = G(r)}$ where $\mathbf{r \leftarrow \{0,1\}^\lambda}$
- Hyb 1: $\mathbf{S \leftarrow \{0,1\}^{3\lambda}}$
- Hyb 2: $\mathbf{S = S' \oplus k}$, where $\mathbf{S' \leftarrow \{0,1\}^{3\lambda}}$
- Hyb 3: $\mathbf{S = Com(1;r) = G(r) \oplus k}$ where $\mathbf{r \leftarrow \{0,1\}^\lambda}$

Theorem: (Setup,Com) is statistically binding

Proof:

For any r,r', $\Pr[G(r) = G(r') \oplus k] = 2^{-3\lambda}$

By union bound:

$\Pr[\exists\, r,r'$ such that $Com(k,0)=Com(k,1)]$

$= \Pr[\exists\, r,r'$ such that $G(r) = G(r') \oplus k] < 2^{-\lambda}$

Number-theoretic Constructions

So Far…

Two ways to construct cryptographic schemes:

- Use others as building blocks
  - PRGs → Stream ciphers
  - PRFs → PRPs
  - PRFs/PRPs → CPA-secure Encryption
  - …

- From scratch
  - RC4, DES, AES, etc

In either case, ultimately scheme or some building block built from scratch

Cryptographic Assumptions

Security of schemes built from scratch relies solely on our inability to break them

- No security proof
- Perhaps arguments for security

We gain confidence in security over time if we see that nobody can break scheme

Number-theory Constructions

Goal: base security on hard problems of interest to mathematicians

- Wider set of people trying to solve problem
- Longer history

Integer Factorization

Given an integer **N**, factor **N** into its prime factors

Studied for centuries, presumed computationally difficult

- Grade school algorithm: $O(N^{1/2})$
- Much better algorithms:

$$\exp(C (\log n)^{1/3} (\log \log n)^{2/3})$$

- However, all require super-polynomial time

Factoring Assumption: Let $\mathbf{p}$, $\mathbf{q}$ be two random $\boldsymbol{\lambda}$-bit primes, and $\mathbf{N = pq}$. Then any PPT algorithm, given $\mathbf{N}$, has at best a negligible probability of recovering $\mathbf{p}$ and $\mathbf{q}$

One-way Functions From Factoring

P_λ = {λ-bit primes}

$F: P_\lambda^2 \to \{0,1\}^{2\lambda}$
$F(p,q) = p \times q$

Trivial Theorem: If factoring assumption holds, then **F** is one-way

Sampling Random Primes

Prime Number Theorem: A random **λ**-bit number is prime with probability **≈1/λ**

Primality Testing: It is possible in polynomial time to decide if an integer is prime

Fermat Primality Test (randomized, some false positives):

- Choose a random integer $a \in \{0,\ldots,N-1\}$
- Test if $a^N = a \bmod N$
- Repeat many times

Discrete Log

Let **p** be a large integer (maybe prime)

Given $\mathbf{g} \in \mathbb{Z}_p^*$, $\mathbf{a} \in \mathbb{Z}$, easy to compute $\mathbf{g^a \bmod p}$

However, no known efficient ways to recover **a** from **g** and $\mathbf{g^a \bmod p}$

Discrete Log Assumption: Let $\mathbf{p}$ be a $\boldsymbol{\lambda}$-bit integer.

Then the function $\mathbf{(g,a)} \rightarrow \mathbf{(g, g^{a} \bmod p)}$ is one-way, where

- $\mathbf{g} \in \mathbb{Z}_{\mathbf{p}}^{*}$
- $\mathbf{a} \in \mathbb{Z}_{\Phi(\mathbf{p})}$

Generalizing Discrete Log

Let $\mathbf{G}_\lambda$ be multiplicative groups of size $\mathbf{n}_\lambda$

Definition: The discrete log assumption holds on $\{\mathbf{G}_\lambda\}$ if the function $\mathbf{F}:\mathbf{G}_\lambda\times\{0,\ldots,\mathbf{n}_\lambda-1\}\rightarrow\mathbf{G}_\lambda^2$ is one-way, where

$$F(g,a) = (g,g^a)$$

Examples:

- $\mathbf{G} = \mathbb{Z}_p^*$ for a prime $\mathbf{p}$, $\mathbf{n} = \mathbf{p}-1$
- $\mathbf{G} =$ subgroup of $\mathbb{Z}_p^*$ of order $\mathbf{q}$, where $\mathbf{q} \mid \mathbf{p}-1$
- $\mathbf{G} =$ "elliptic curve groups"

Hardness of Discrete Log

Brute force search: **$O(n)$**

Better generic algorithm: **$O(n^{1/2})$**
- Known to be optimal for generic algorithms

Much better algorithms are known for $\mathbb{Z}_p^*$
- Similar running times to integer factorization
- Still super-polynomial