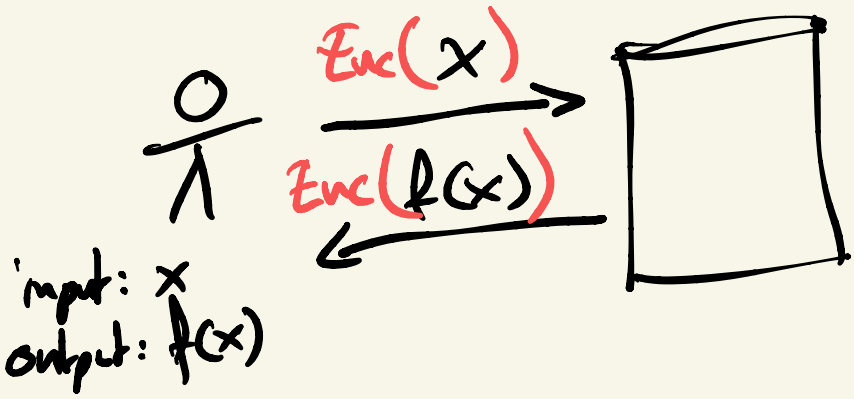
PETS lectures 2-3-4

Commitment Schemes

Florian Tramèr

Recap: FHE



Key = p (large odd integer)

$Enc(m) = pq + 2r + m$, $r \ll p$, $r \in_R [0, 2^{\lambda})$

$Dec(c) = (c \bmod p) \bmod 2$

Homomorphisms

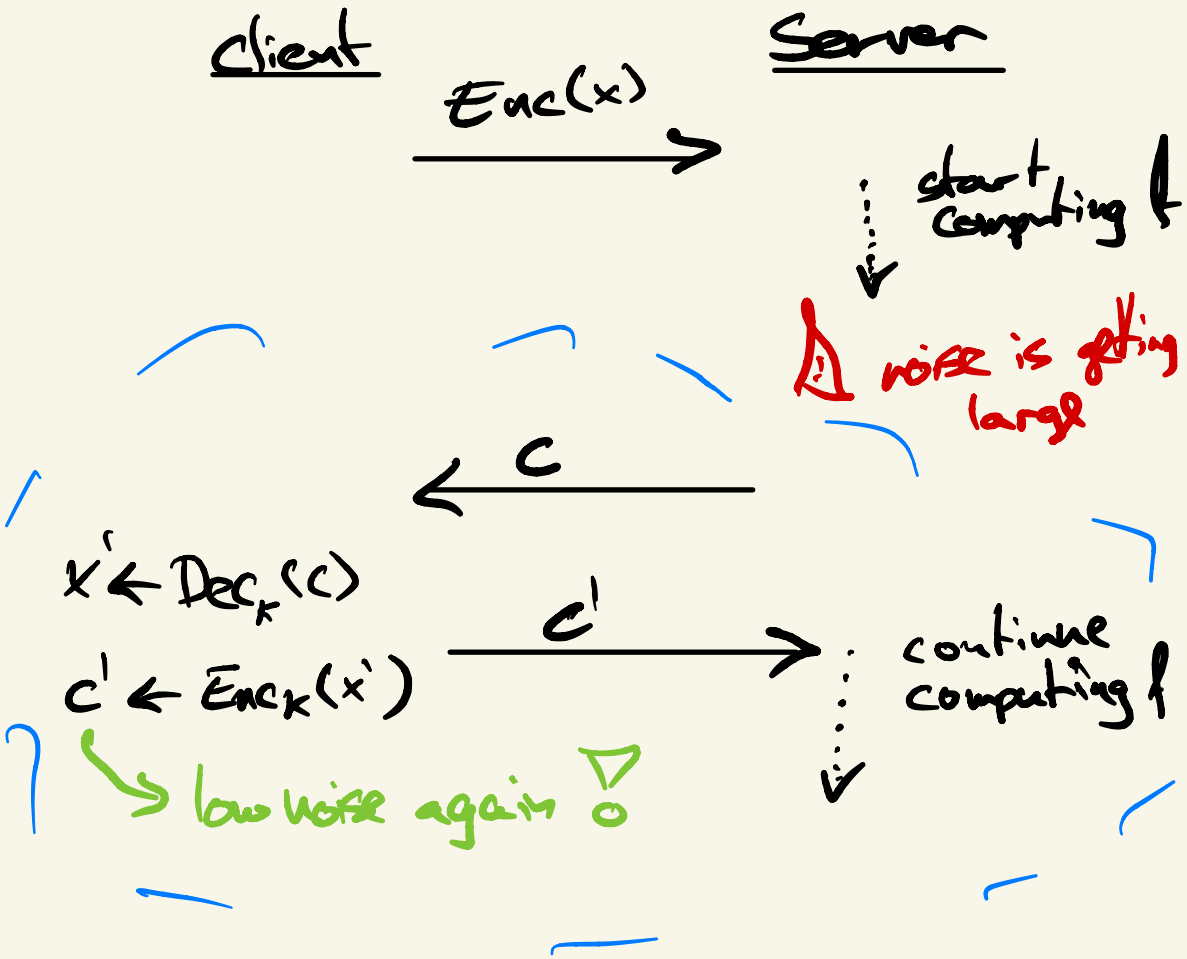
$Enc(m_1) + Enc(m_2) = pq' + 2(r_1 + r_2) + (m_1 + m_2)$

$Enc(m_1) \cdot Enc(m_2) \simeq pq' + 2r_1 r_2 + m_1 m_2$

⚠ noise grows with each op

Bootstrapping

Strawman:



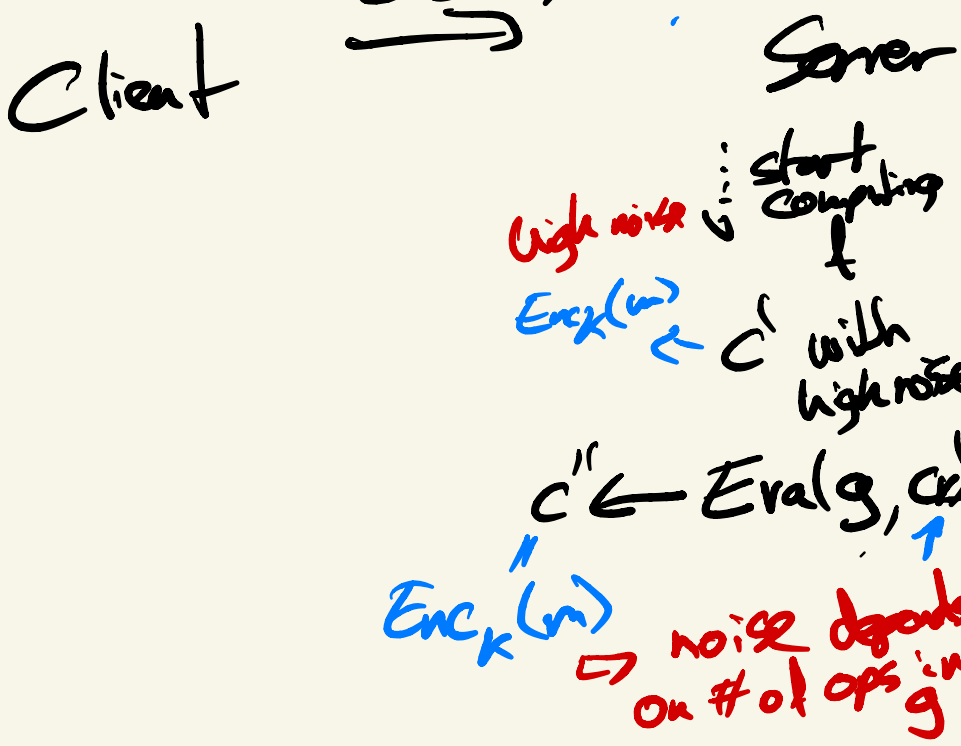
Idea: make the server run $Dec(c)$ homomorphically

Define: $g(k) = Dec_k(c)$

We want to homomorphically evaluate g

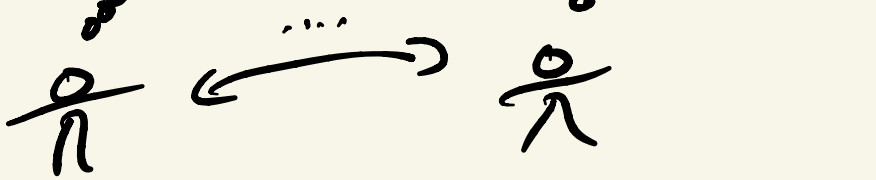
We need an encryption of the key

$c_k \leftarrow Enc_k(k)$



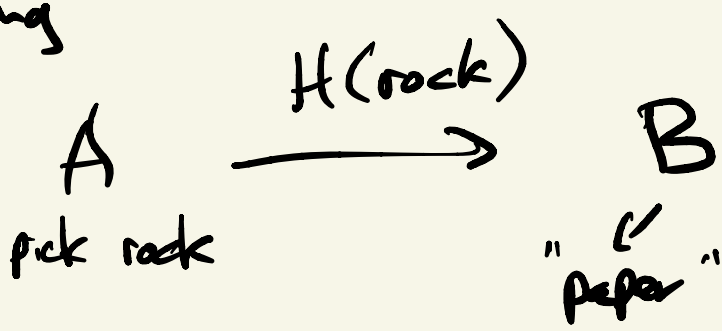
Commitment schemes

A game:



Failed examples

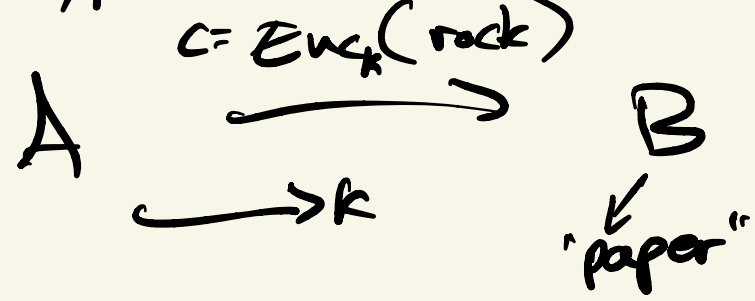
- Hashing



issue: this is not hiding

B can just compute H(rock) H(paper) H(scissors) and check which one A sent

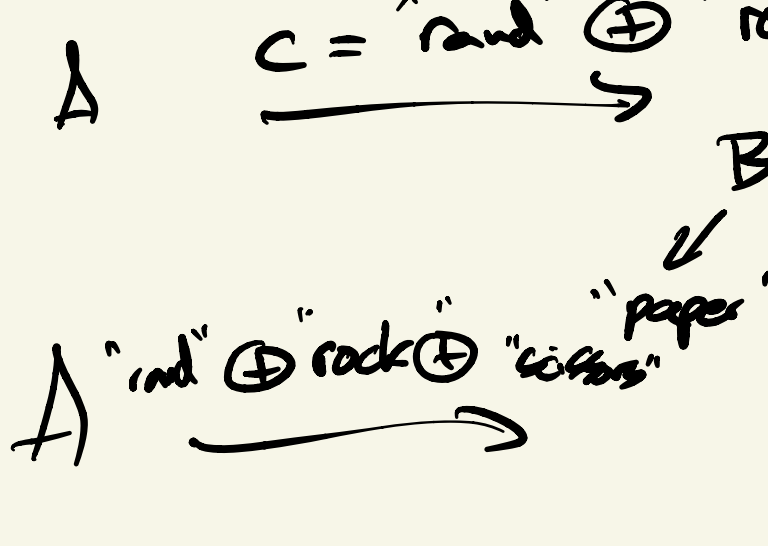
- Encrypt



Issue: A might find k' such that $Dec_{k'}(c) =$ scissors

This is not binding

Ex: one-time pad



Define commitment schemes:

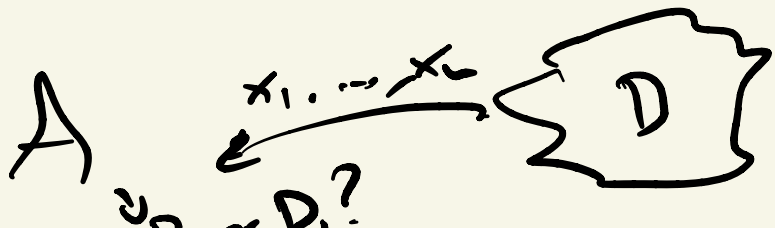
Definition: Indistinguishability

let D_0, D_1 be distributions

- statistically indistinguishable

$$D_0 \equiv D_1$$

$\Rightarrow D_0, D_1$ are impossible to distinguish who even for a computationally unbounded adversary with access to poly(λ) samples from D_0 or D_1



Ex: $D_0 = Unif([0, 2^\lambda])$

$D_1 = Unif([0, 2^\lambda + 1])$

$negl(\lambda)$ means smaller than $\frac{1}{\lambda^c}$

for all constants c

in general, think $e^{-\lambda}$

D_0 and D_1 are computationally indistinguishable

$D_0 \overset{c}{\approx} D_1$ if no poly-time adv can distinguish them with non-negligible prob.

Define two "worlds":

$$W_0 = \{A(x)=1 : x \leftarrow D_0\}$$

$$W_1 = \{A(x)=1 : x \leftarrow D_1\}$$

Def $Adv[A] = |Pr[W_0] - Pr[W_1]|$

$D_0 \overset{c}{\approx} D_1 \iff Adv(A) = negl(\lambda)$ for all poly-time A

Commitment schemes

- $Commit : M \times R \to C$

  $Commit(m, r) = c$

Properties:

- Hiding: $\forall m_0, m_1 \in M$

  $\{ commit(m_0, r) : r \xleftarrow{\$} R \}$

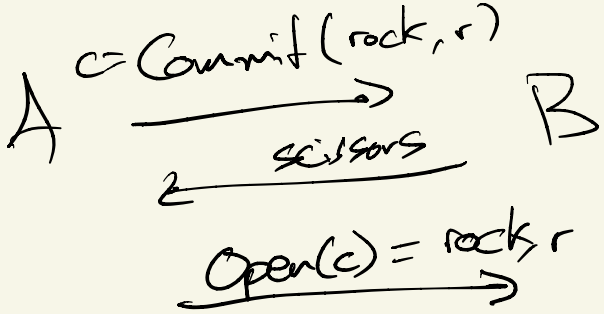
  $\overset{c}{\equiv}$ $\{ commit(m_1, r) : r \xleftarrow{\$} R \}$

- **Binding**

No (PPT) adversary A can produce $m_0, m_1 \in M$, $r_0, r_1 \in R$ s.t.

$$Commit(m_0, r_0) = Commit(m_1, r_1)$$

Terminology:



A simple commitment scheme

$$\text{Commit}(m, r) = H(m, r)$$

How to prove this is secure?

→ binding : collision resistance

→ hiding : not a standard property of hash functions

Random oracle model

- suppose your scheme uses a hash function H
- when proving security, assume H is a random function from X to Y
  - ↳ for each x, sample $y = H(x)$ uniformly from Y
- A random function ≠ a "randomized" function

Prove that $H(m,r)$ is a secure commitment scheme in the ROM:

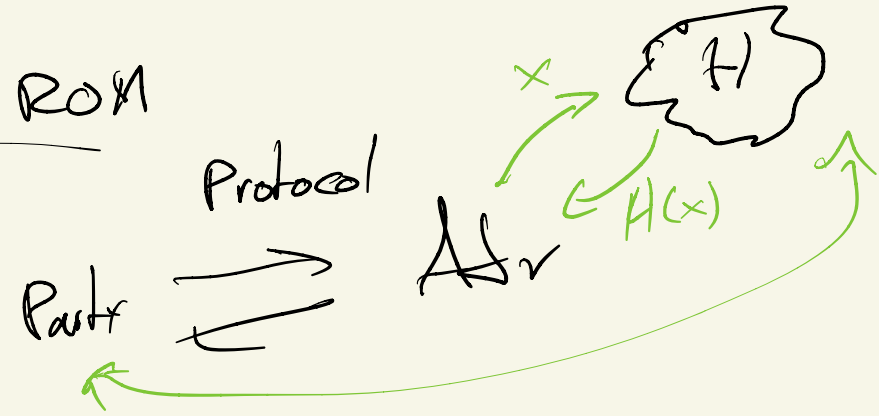
* binding: a random function is collision resistant

  $O(\sqrt{|Y|})$ queries to find a collision

* statistically hiding: if H is a random function then $H(m,r)$ is uniform in Y unless the Adv queries H on (m,r) then have no information about m

the adv has to make exp. # of queries to H

Proofs in ROM



Commitment without ROM (Pedersen)

- let $\mathbb{G}$ be a group of prime order q with g, h generators of $\mathbb{G}$ and $h = g^x \bmod q$ for x unknown

- $\text{Commit} : \mathbb{Z}_q \times \mathbb{Z}_q \to \mathbb{G}$

$$\text{Commit}(m, r) = g^m h^r$$

- this scheme is homomorphic $\forall$

$$\text{Commit}(m_0, r_0) \cdot \text{Commit}(m_1, r_1)$$

$$= g^{m_0+m_1} \cdot h^{r_0+r_1} = \text{Commit}(m_0+m_1, r_0+r_1)$$

We can compute linear functions over commitm.

Proof of security

- Hiding: Pedersen is statistically hiding

- $\text{Commit}(m, r) = g^m h^r = g^{m+\alpha x}$ $\quad h = g^x$

Given a commitment $C = g^\alpha$

then for any message m, there exists one randomness r s.t

$$\text{Commit}(m, r) = C$$

namely $r = (\alpha - m)/x$

- Computational binding assuming hardness of discrete log

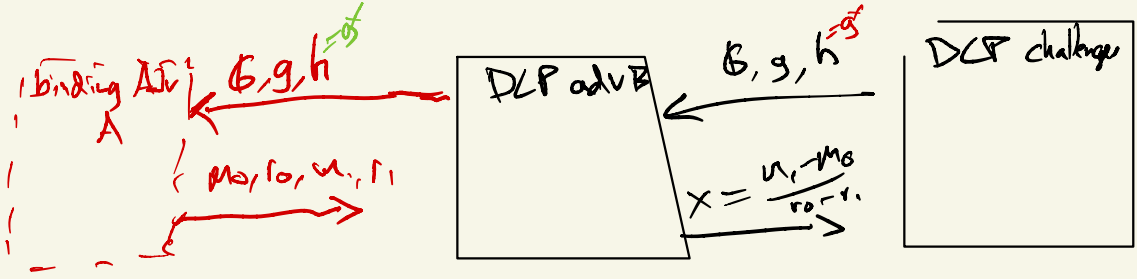
Def: Discrete log problem

- let $\mathbb{G}$ be a group of prime order q
- let g be a generator of $\mathbb{G}$
- Given $g^x \in \mathbb{G}$ for $x \xleftarrow{\$} \mathbb{Z}_q$ find x.

To prove: DLP is hard $\Rightarrow$ Pedersen is binding

Pedersen is not binding $\Rightarrow$ DLP is not hard

Assume an adv for Pedersen and show we can build an adv for DLP



Life advice: to find a dlog, find two different representations of a group element

$$g^{m_0} h^{r_0} = g^{m_1} h^{r_1} \qquad m_0 \neq m_1$$

$$g^{m_0 + x r_0} = g^{m_1 + x r_1}$$

$$m_0 + x r_0 = m_1 + x r_1 \mod q$$

$$x = \frac{m_1 - m_0}{r_0 - r_1} \mod q$$

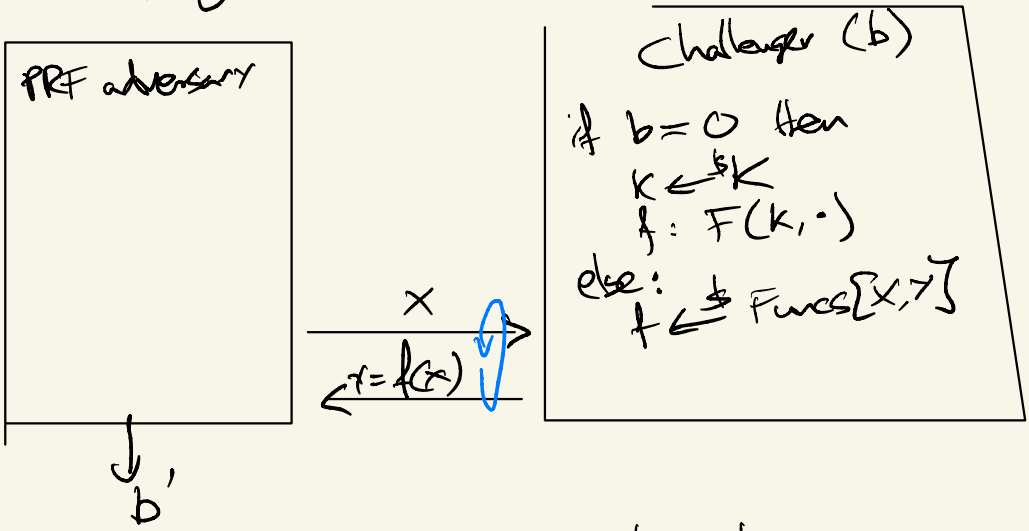
our Adv B succeeds whenever Adv A succeeds ∎

Pseudorandom functions

Def: A PRF $F: K \times X \to Y$ is a deterministic and efficient function

It is secure if indistinguishable from a random function sampled from $Funcs[X, Y]$

A security game:



Adv wins if $b' = b$

$$Adv_{PRF}[A, F] = \left| Pr\left[A^{f(\cdot)} = 1 : f \xleftarrow{\$} Funcs[X,Y]\right] - Pr\left[A^{F(k,\cdot)} : k \xleftarrow{\$} K\right] \right|$$

Examples of PRFs

- $H(k, x)$ is a PRF in the Random Oracle Model

  G in practice rather than SHA(k,x) you would use AES (much faster)

- $F(k, x) = H(x)^k$ where $f: X \to G$

Claim: this PRF is secure in the ROM and assuming that DDH is hard

Def: DDH problem

- Given G of order q, gen g

$D_0 = \{(g^a, g^b, g^{ab}) : a, b \leftarrow \mathbb{Z}_q\}$

$D_1 = \{(g^a, g^b, g^c) : a, b, c \leftarrow \mathbb{Z}_q\}$

$D_0 \approx_s D_1$

Security proof of our PRF

Simplifications: • Adv makes a single PRF query

• we show that

$$Adv_{DDH}(B) \geq \frac{1}{2Q_{RO}} Adv_{PRF}(A) - negl(\lambda)$$

↳ # of RO queries made by A

if PRF is not secure $\Rightarrow$ DDH is not hard