Zero Knowledge Proofs

Introduction to Zero Knowledge Interactive Proofs

Dan Boneh, **Shafi Goldwasser,** Dawn Song, Justin Thaler, Yupeng Zhang





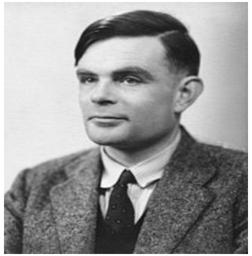
Stanford University

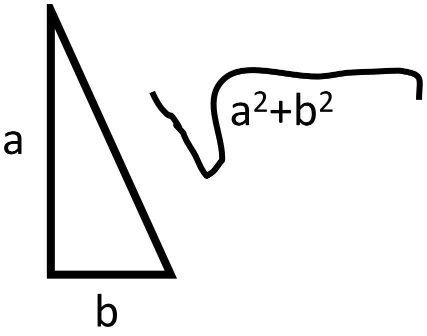
Berkeley UNIVERSITY OF CALIFORNIA

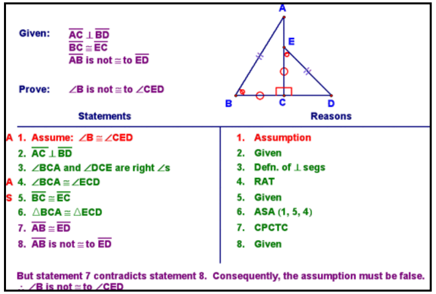
GEORGETOWN UNIVERSITY

TEXAS A&M UNIVERSITY

Classical Proofs







Given: $\overline{AC} \perp \overline{BD}$, $\overline{BC} \cong \overline{EC}$, $\overline{AB}$ is not $\cong$ to $\overline{ED}$

Prove: $\angle B$ is not $\cong$ to $\angle CED$

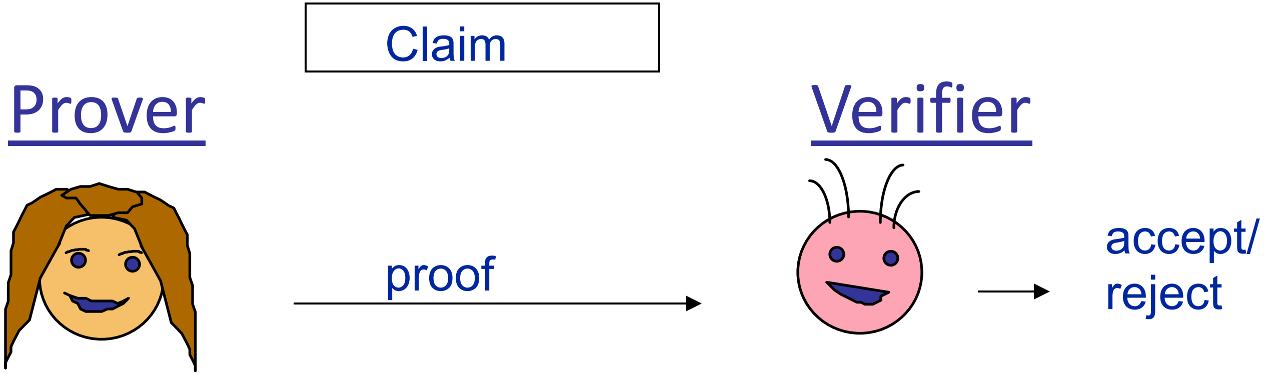
	Statements	Reasons
A	1. Assume: $\angle B \cong \angle CED$	1. Assumption
	2. $\overline{AC} \perp \overline{BD}$	2. Given
	3. $\angle BCA$ and $\angle DCE$ are right $\angle$s	3. Defn. of $\perp$ segs
A	4. $\angle BCA \cong \angle ECD$	4. RAT
S	5. $\overline{BC} \cong \overline{EC}$	5. Given
	6. $\triangle BCA \cong \triangle ECD$	6. ASA (1, 5, 4)
	7. $\overline{AB} \cong \overline{ED}$	7. CPCTC
	8. $\overline{AB}$ is not $\cong$ to $\overline{ED}$	8. Given

But statement 7 contradicts statement 8. Consequently, the assumption must be false.
$\therefore$ $\angle B$ is not $\cong$ to $\angle CED$

. . .

Prime-Number Thm

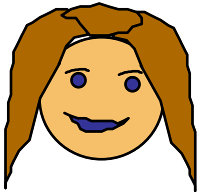
Proofs



Efficiently Verifiable Proofs (NP-proofs)

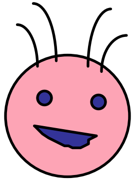
Claim

Prover



short proof

Verifier



accept/
reject

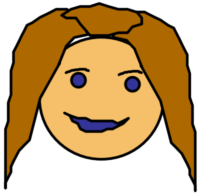
Works Hard

Polynomial Time

Efficiently Verifiable Proofs (NP-proofs)

Claim x

Prover P

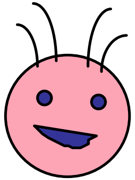


Unbounded

w

$|w|$ = polynomial in $|x|$

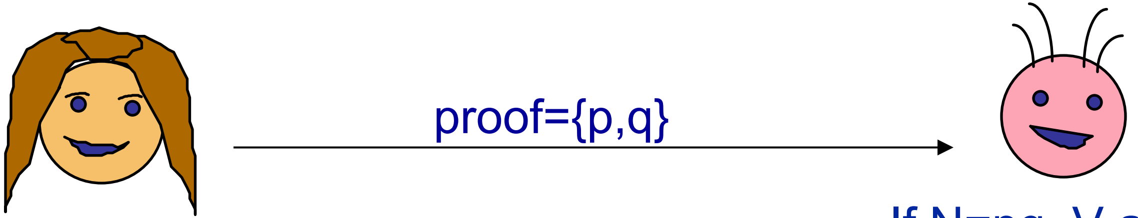
Verifier V



Accepts x If
V(x,w)=1
Else reject

V takes time Polynomial **in |x|**

Claim: N is a product of 2 large primes

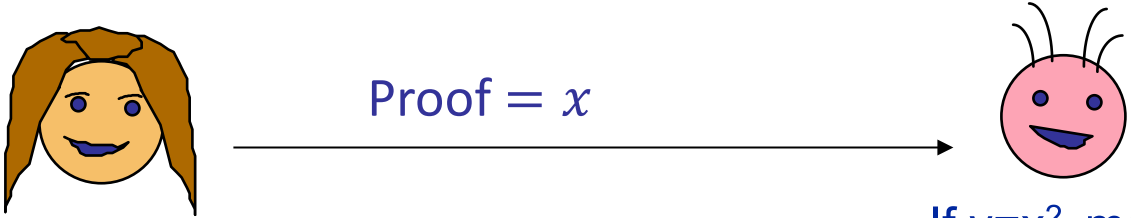


If N=pq, V accepts
Else V rejects

After interaction, V knows:

1) N is product of 2 primes
2) The two primes p and q

Claim: y is a quadratic residue mod N
(i.e $\exists x \ in \ Z_N^*$ s.t. y=x^2 mod N)

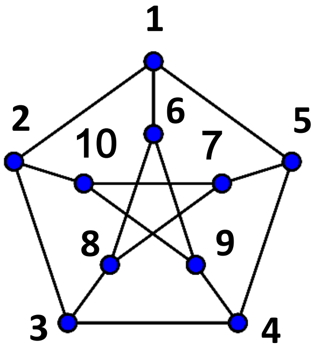


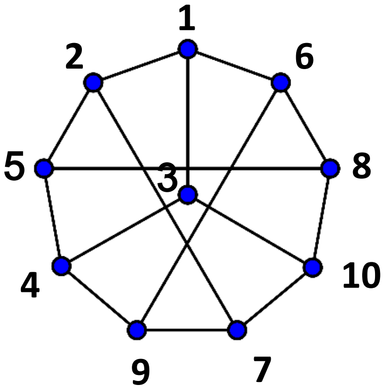
If y=x^2 mod N, V accepts
Else V rejects

After interaction, V knows:

1. y is a quadratic residue mod
2. Square root of y (hard problem equivalent to factoring N)

Claim: the two graphs are isomorphic

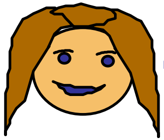




After interaction, V knows:

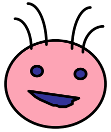
1) G_0 is isomorphic to G_1

2) The isomorphism $\boldsymbol{\pi}$



$\boldsymbol{\pi}: [N] \rightarrow [N]$,

the isomorphism

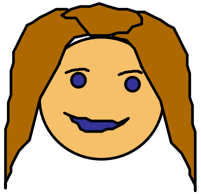


Accept if $\forall i, j$:
$\big(\pi(i), \pi(j)\big) \in E_1$ iff $(i, j) \in E_0$.

Efficiently Verifiable Proofs (NP-Languages)

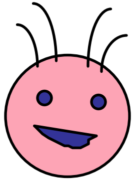
Claim x

Prover P



w

Verifier V



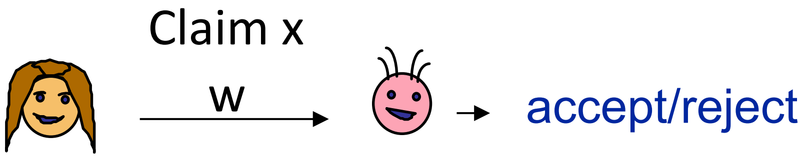
Accepts x If
V(x,w)=1
Else reject

Works Hard

V Polynomial time

Def: A language L is a set of binary strings x.

Efficiently Verifiable Proofs (NP-Languages)

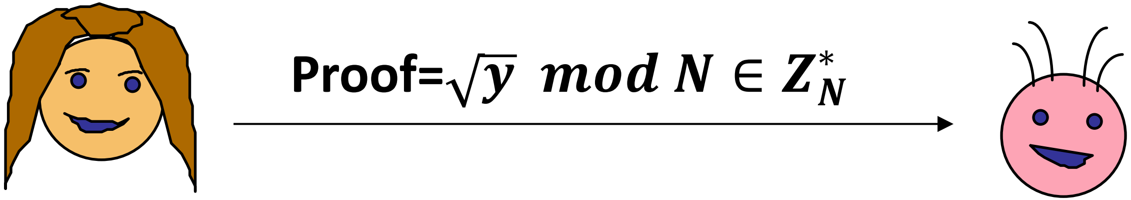


Def: $\mathcal{L}$ is an **NP**-language (or NP-decision problem), if there is a **poly (|x|) time** verifier V where

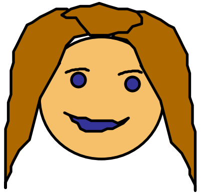
- **Completeness [True claims have (short) proofs].**
  if $\mathrm{x} \in \mathcal{L}$, there is a **poly(|x|)-long** witness $\mathrm{w} \in \{0,1\}^*$ s.t. $V(x, w) = 1$.
- **Soundness [False theorems have no proofs}.**
  if $\mathrm{x} \notin \mathcal{L}$, there is no witness. That is, for all $\mathrm{w} \in \{0,1\}^*$, $V(x, w) = 0$.

1982-1985: Is there any other way?

Theorem: y is a quadratic residue mod N



Zero Knowledge Proofs: Yes



Main Idea:

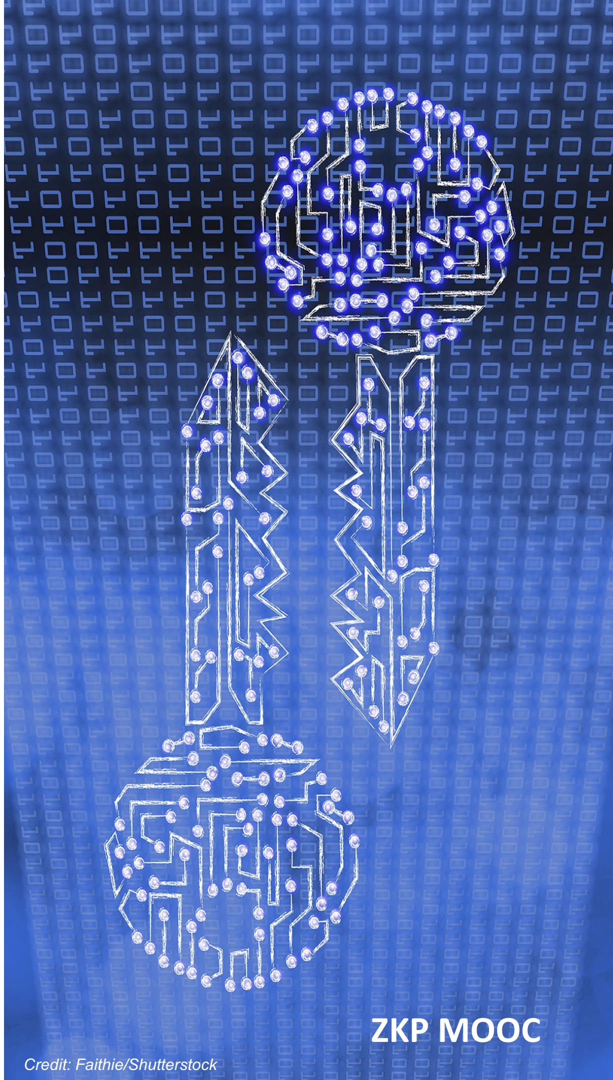
Prove that

I **could** prove it

If I felt like it

Micali Goldwasser Rackoff

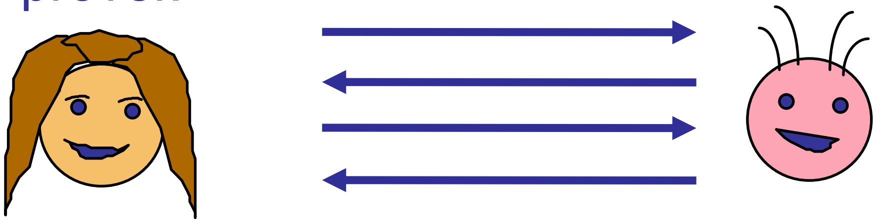
Zero Knowledge Interactive Proofs



Two New Ingredients

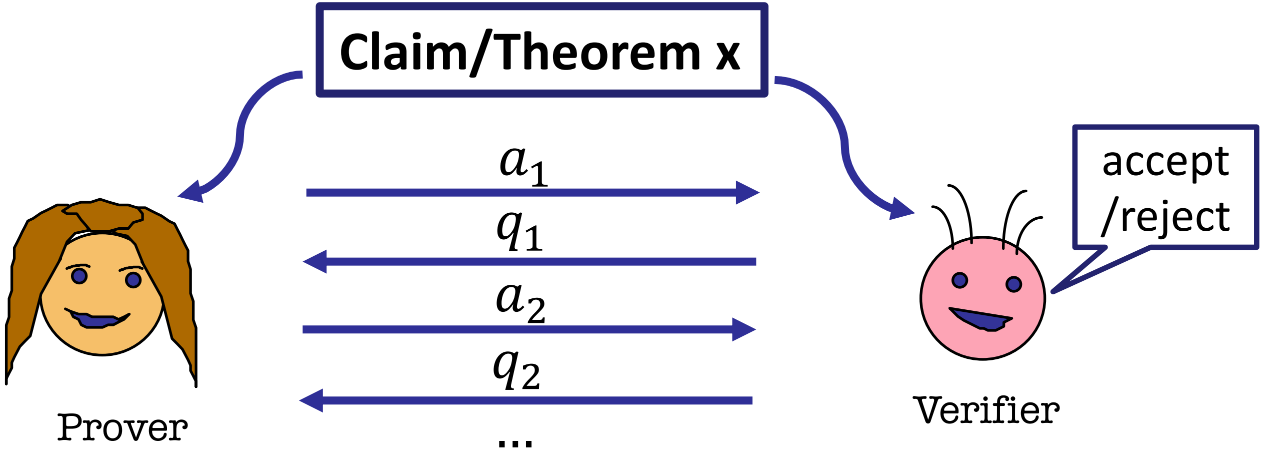
Interactive and Probabilistic Proofs

Interaction: rather than passively "reading" proof, verifier engages in a non-trivial interaction with the prover.



Randomness: verifier is randomized (tosses coins as a primitive operation), and can err in accept/reject with small probability

Interactive Proof Model

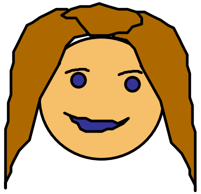


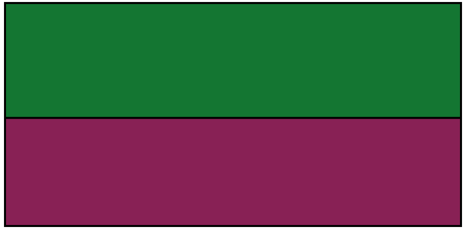
Comp. Unbounded

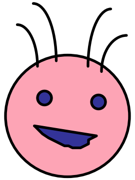
Probabilistic
Polynomial-time (PPT)

Here is the idea:
How to prove colors are different to a **blind verifier**

Claim: This page contains 2 colors







Toss coin to decide if to flip page over or not
Heads flip, Tails don't

Sends resulting page

If page is flipped
Set coin'=heads
Else coin'=tails

I guess you tossed coin'

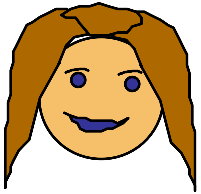
If coin $\neq$ coin',
reject, else accept

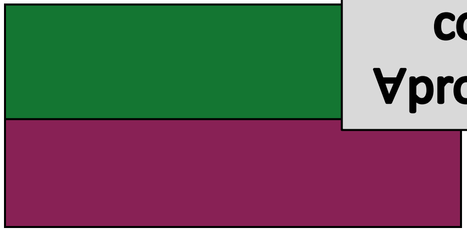
Here is the idea:
How to prove colors are differ

Claim: This page contains 2 c

- If there are 2 colors, then Verifier will accept
- If there is a single color, ∀provers $Prob_{coins}$(Verifier accept)$\leq$ 1/2
- If repeat i=1..k times and V accept if $coin_i' = coin_i$ every repetition,

∀provers $Prob_{coins}$(Verifier accept)$\leq 1/2^k$





Toss coin to decide if to flip page over or not
Heads flip, Tails don't

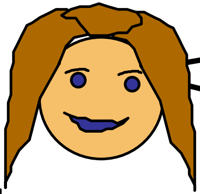
Sends resulting page

If page is flipped
Set coin'=heads
Else coin'=tails

I guess you tossed coin'

If coin $\neq$ coin',
reject, else accept

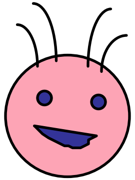
Interactive Proof for QR$= \{(N, y): \exists x \ s.t. \ y = x^2 \ mod \ N\}$



Choose random $1 \leq r \leq N$ s.t. gcd(r,N)=1

Send s=r^2 mod n and say

- If I gave you square roots of both s and sy mod N you would be convinced that the claim is true (but also know $\sqrt{y}$ mod N)
- Instead, I will give you a square root of either s or of sy mod N but you get to choose which!



Flip a b=  to choose

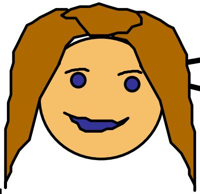
If b=1: send z=r

If b=0: send z=r$\sqrt{y}$ mod N

Accepts only if z^2=sy^{1-b} mod N

Interactive Proof for QR= {

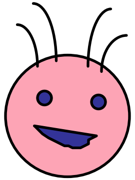
- **Completeness: If Claim is true, then Verifier will accept**
- **Soundness: If Claim is false, ∀provers $\text{Prob}_{\text{coins}}$(Verifier accept)≤ 1/2**
- **Prover only needs to know $\mathbf{x} = \sqrt{y}$**



Choose random $1 \leq r \leq N$ s.t. gcd(r,N)=1

Sends s=r^2 mod n and says

- If I gave you square roots … be convinced that the claim is true (but also know $\sqrt{y}$ mod N)
- Instead, I will give you a a square rot of s or of sy mod N but you get to choose which!



Flip a b=  to choose

Accepts only if z^2=sy^{1-b} mod N

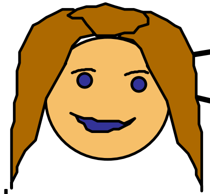
If b=1: send z=r

If b=0: send z=r$\sqrt{y}$ mod N

Interactive Proof for Q

- **Completeness: If Claim is true, then Verifier will accept**
- **Soundness: If Claim is false, $\forall$provers $\text{Prob}_{\text{coins}}$(Verifier accept)$\leq$ $(1/2)^{100}$**
- **Prover only needs to know $\mathbf{x = \sqrt{y}}$**

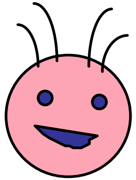
Repeat 100 times



Send $s=r^2$ mod n and s

- If I gave you square
  convinced that the claim is true (but also know $\sqrt{y}$ mod N)
- Instead, I will give you a a square rot of s or of sy mod N but you get to choose which!
- The fact that I COULD (in principle) do both, should convince you

Choose random $1 \leq r \leq N$ s.t. gcd(r,N)=1



Flip a b=  to choose

Accepts only if $z^2 = sy^{1-b}$ mod N

If b=1: send z=r
If b=0: send z=r$\sqrt{y}$ mod N

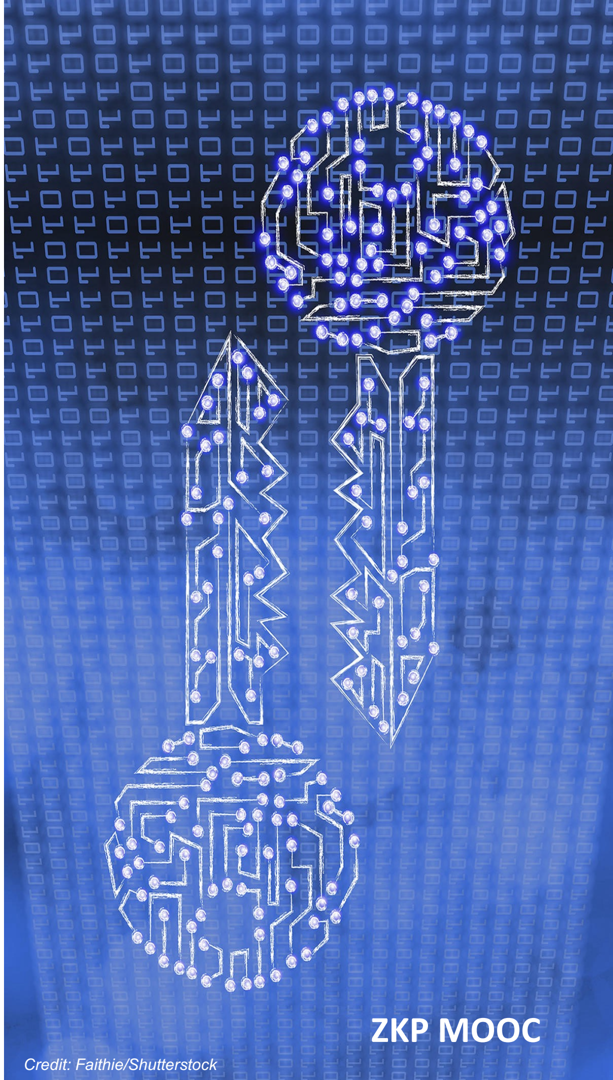
What Made it possible?

- The statement to be proven has **many possible proofs** of which the prover chooses one ***at random***.

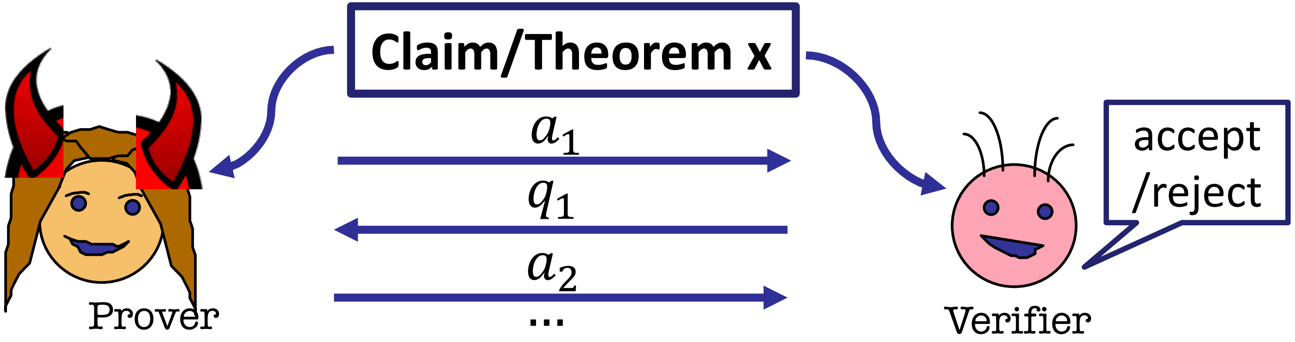
- Each such proof is made up of exactly 2 parts: seeing either part on its own gives the verifier no knowledge; seeing both parts imply 100% correctness.

- Verifier chooses **at random** which of the two parts of the proof he wants the prover to give him. The ability of the prover to provide either part, convinces the verifier

Definitions : of Zero Knowledge Interactive Proofs



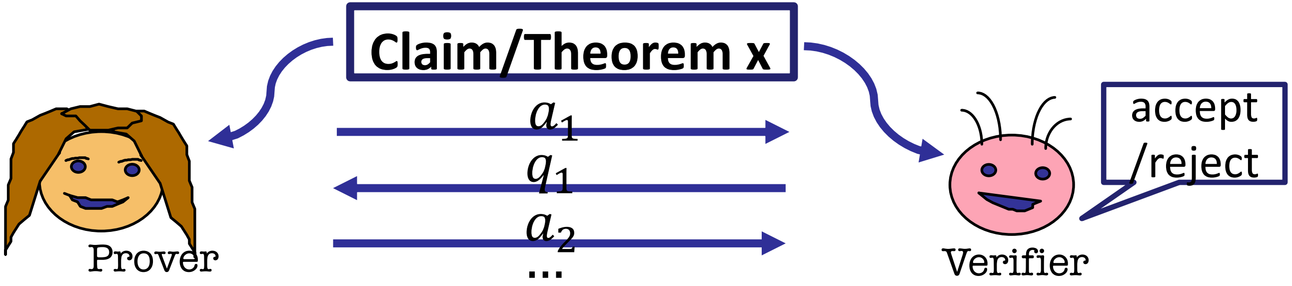
Interactive Proofs for a Language $\mathcal{L}$



Def: (P, V) is an interactive proof for L, if V is probabilistic poly (|x|) time &

- **Completeness**: If $x \in \mathcal{L}$, V always accepts.
- **Soundness:** If $x \notin \mathcal{L}$, for all cheating prover strategy, V will not accept except with negligible probability.

Interactive Proofs: Notation

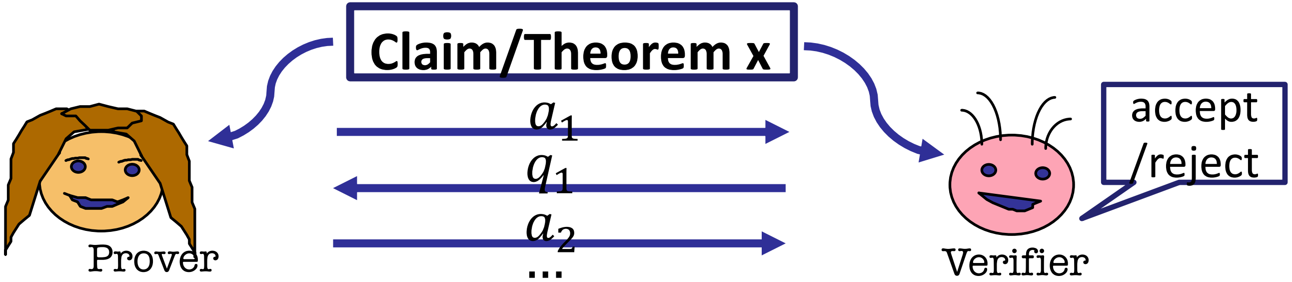


<u>Def</u>: (P, V) is an interactive proof for L, if V *is* probabilistic poly (|x|) and

- **Completeness: If** $\mathrm{x} \in \mathcal{L}$, $Pr[(P,V)(x) = accept] = 1$.
- **Soundness: If** $\mathrm{x} \notin \mathcal{L}$, **for every** $\boldsymbol{P}^*$, $\Pr[(P^*,V)(x) = accept] = negl(|x|)]$

where $\mathrm{negl}(\lambda) < \frac{1}{\mathrm{polynomial}(\lambda)}$ for all polynomial functions

Interactive Proofs: Notation



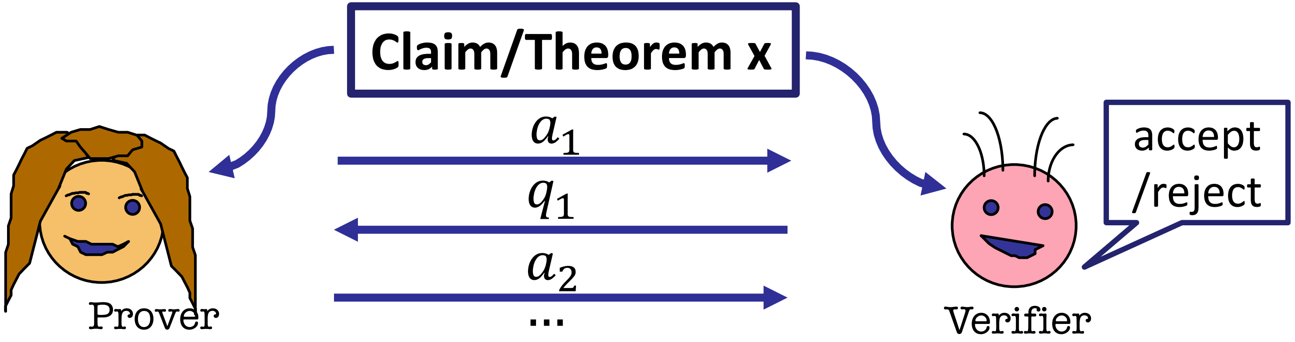
Def

- (
- $|)]$

whe

This is what a proof ultimately is!

Interactive Proofs for a Language $\mathcal{L}$: Notation

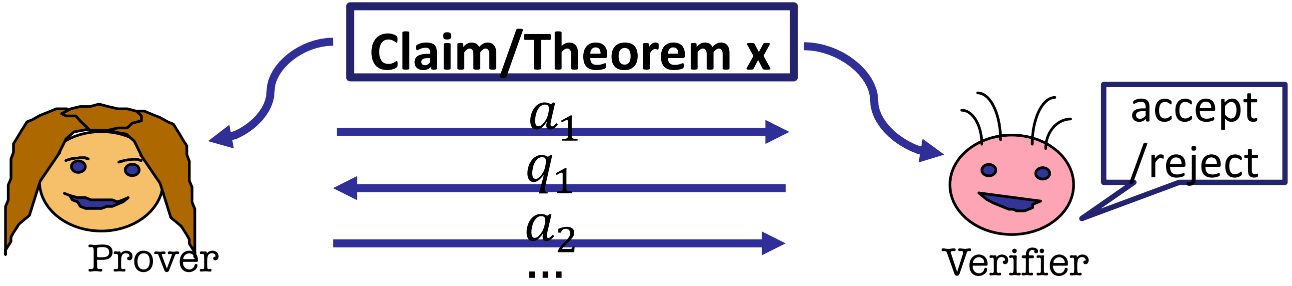


<u>Def</u>: (P, V) is an interactive proof for L, if V *is* probabilistic poly (|x|) and

- **Completeness: If** $\mathrm{x} \in \mathcal{L}$, $\Pr[(\mathrm{P}, V)(x) = accept] \geq c$
- **Soundness: If** $\mathrm{x} \notin \mathcal{L}$, **for every** $\boldsymbol{P}^*$, $\Pr[(P^*, V)(x) = accept] \leq s$

Equivalent as long as $\boldsymbol{c} - \mathbf{s} \geq 1/\mathrm{poly}(|x|)$

The class of Interactive Proofs (IP)



<u>Def</u>: class of languages **IP =**

{L for which there is an interactive proof}

What is zero-knowledge?

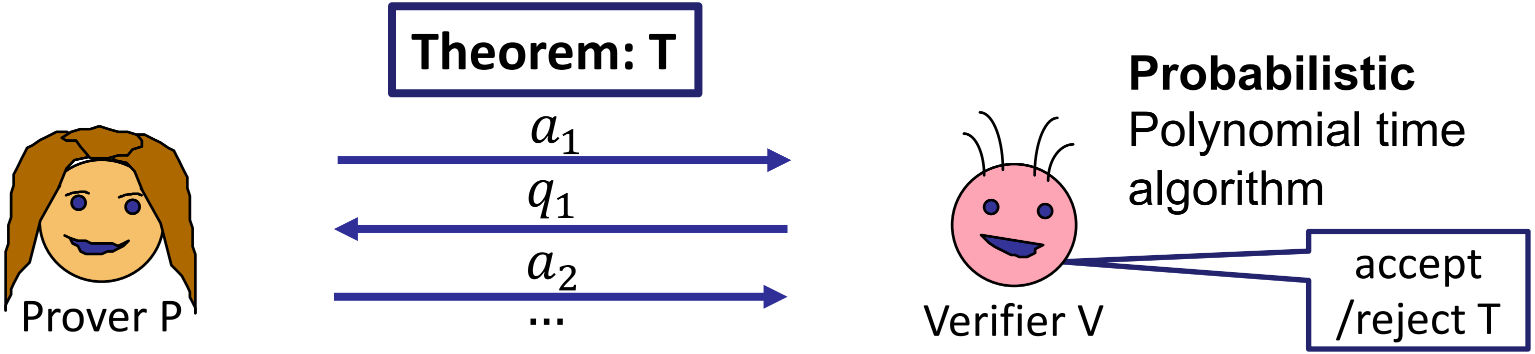
for every verifier

For true Statements,

What the verifier can compute
after the interaction =
What the verifier could have computed
before interaction

How do we capture this mathematically?

The Verifier's View

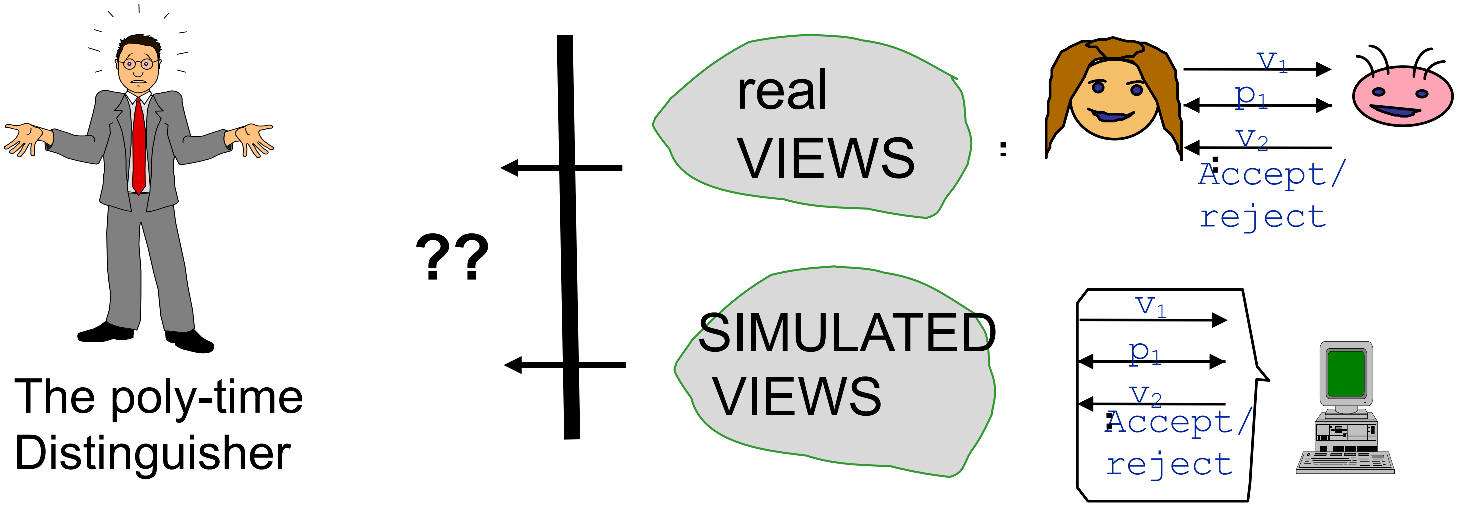


- After interactive proof, V "learned":
  - T is true (or $x \in \mathcal{L}$)
  - A **view** of interaction (= transcript + coins V tossed)

Def: $\textbf{view}_V(P, V)[x] =$ $\{(q_1, a_1, q_2, a_2, \ldots, \text{coins of V})\}$.
(probability distribution over coins of V and P)

The Simulation Paradigm

V's view gives him nothing new, if he could have simulated it its own s.t `simulated view' and `real-view' are computationally-Indistinguishable

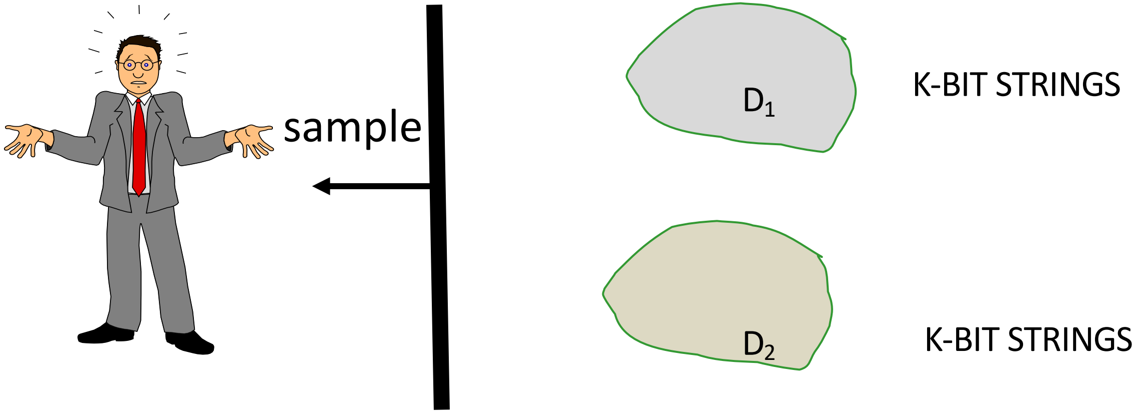


When Theorem is true

Computational Indistinguishability

If no "distinguisher" can tell apart two different probability distributions they are "effectively the same".

Probabilistic
Poly Time (k)
Distinguisher D



For all distinguisher algorithms D, even after receiving a polynomial number of samples from D_b, Prob[D guesses b] <1/2+negl(k)

Zero Knowledge: Definition

An Interactive Protocol (P,V) is zero-knowledge for a language L if there exists a **PPT** algorithm Sim (a simulator) such that **for every $x \in L$**, the following two probability distributions are **poly-time** indistinguishable:

1. $view_V(P, V)[x] =$ $\{(q_1,a_1,q_2,a_2,\ldots,\text{coins of V})\}$
(over coins of V and P)
2. $Sim(x)$

Def: (P,V) is a zero-knowledge interactive protocol if it is *complete, sound and zero-knowledge*

Zero Knowledge: Definition

An Interactive Protocol (P,V) is zero-knowledge for a language L if there exists a **PPT** algorithm Sim (a simulator) such that **for every $x \in L$**, the following two probability distributions are **poly-time** indistinguishable:

Allow simulator S
Expected
Poly-time

1. $view_V(P,V)[x,1^\lambda] =$ $\{(q_1,a_1,q_2,a_2,\ldots,\text{coins of V})\}$ (over coins of V and P)

2. $Sim(x,1^\lambda)$

Technicality:
Allows sufficient
Runtime onn small x
λ- security parameter

Def: (P,V) is a zero-knowledge interactive protocol if it is *complete, sound and zero-knowledge*

What if V is NOT HONEST

OLD DEF

An Interactive Protocol (P,V) is **honest-verifier** zero-knowledge for a language L if there exists a PPT simulator Sim such that for every $x \in L$,

$$view_V(P,V)[x] \approx \quad Sim(x, 1^\lambda)$$

REAL DEF

An Interactive Protocol (P,V) is **zero-knowledge** for a language L if **for every PPT V^***, there exists a poly time simulator Sim s.t. for every $x \in L$,

$$view_V(P,V)[x] \approx \quad Sim(x, 1^\lambda)$$

34

Flavors of Zero Knowledge

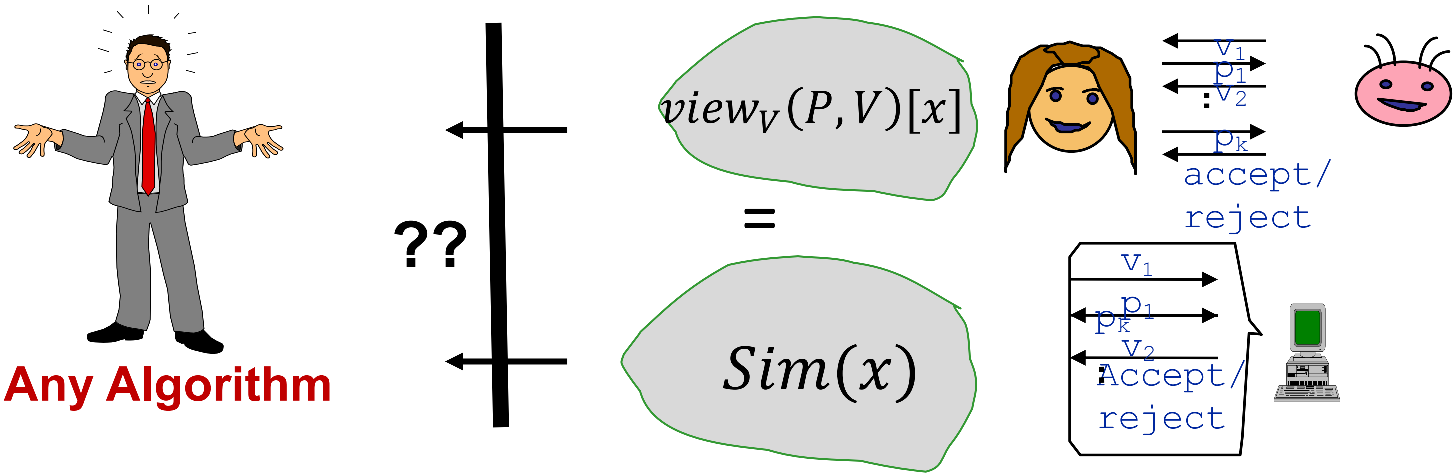
$view_V(P,V)[x]$ $\qquad \approx \qquad$ $Sim(x, 1^\lambda)$

REAL $\qquad\qquad$ SIMULATED

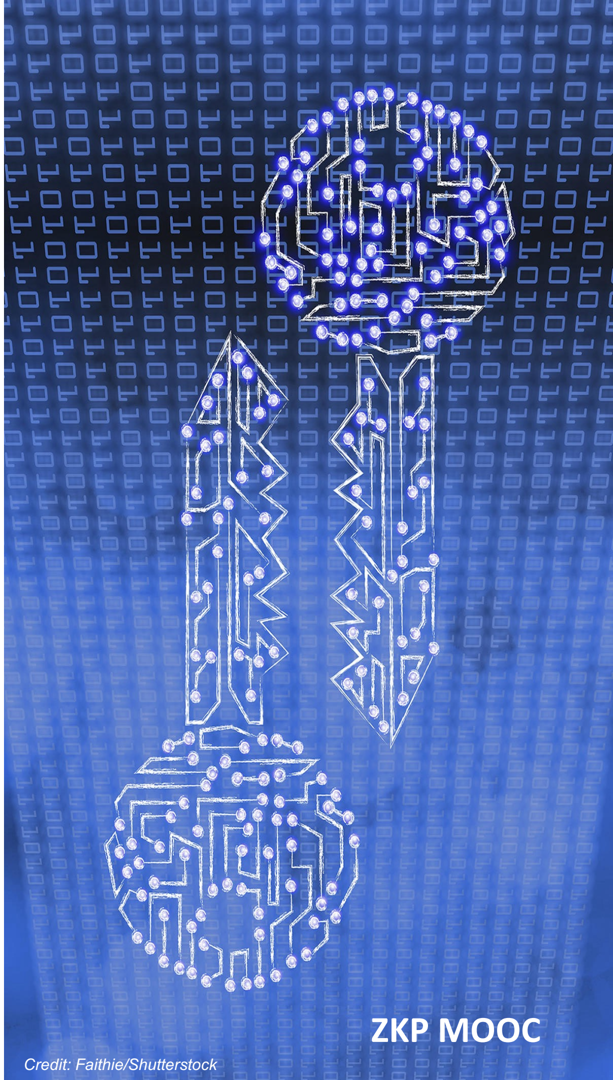
- Computationally indistinguishable distributions = CZK
- Perfectly identical distributions = PZK
- Statistically close distributions = SZK

Special Case: **Perfect** Zero Knowledge

verifier's view can be exactly efficiently simulated
`Simulated views' = `real views'



Working through a Simulation for QR Protocol



Recall the Simulation Paradigm

$view_V(P, V)$:

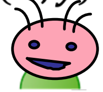
Transcript = (s, b, z),

Coins = b

$s = r^2 \pmod N$ $\longrightarrow$

$b \leftarrow \{0,1\}$ $\longleftarrow$

(N, y)



If b=0: $z = r$ $\longrightarrow$

If b=1: $z = rx$

Check:
$z^2 = sy^b \pmod N$

Recall the Simulation Paradigm

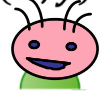
$view_V(P, V)$:
(s, b, z)

sim :
(s, b, z)

$s = r^2 \pmod N$

$b \leftarrow \{0,1\}$

(N, y)

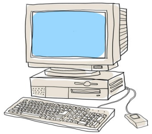


If b=0: $z = r$

If b=1: $z = rx$

Check:
$z^2 = sy^b \pmod N$

PPT "simulator" ***Sim***



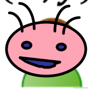
(N, y)

(Honest Verifier) Perfect Zero Knowledge

Claim: The QR protocol is perfect zero knowledge.

$s = r^2 \pmod{N}$

(N, y)



$b \leftarrow \{0,1\}$

If b=0: $z = r$

If b=1: $z = rx$

Check:
$z^2 = sy^b \pmod{N}$

Simulator S works as follows:

1. First pick a random bit b.
2. pick a random $z \in Z_N^*$.
3. compute $\mathrm{s} = z^2/y^b$.
4. output $(\mathrm{s}, \mathrm{b}, \mathrm{z})$.

$view_V(P, V)$:
(s, b, z)

claim: The simulated transcript is identically distributed as the real transcript

Perfect Zero Knowledge: for all V*

Claim: The QR protocol is perfect zero knowledge.

$s = r^2 \pmod N$ $\rightarrow$

(N, y)

$b \leftarrow \{0,1\}$ $\leftarrow$

If b=0: $z = r$ $\rightarrow$
If b=1: $z = rx$

Check:
$z^2 = sy^b \pmod N$

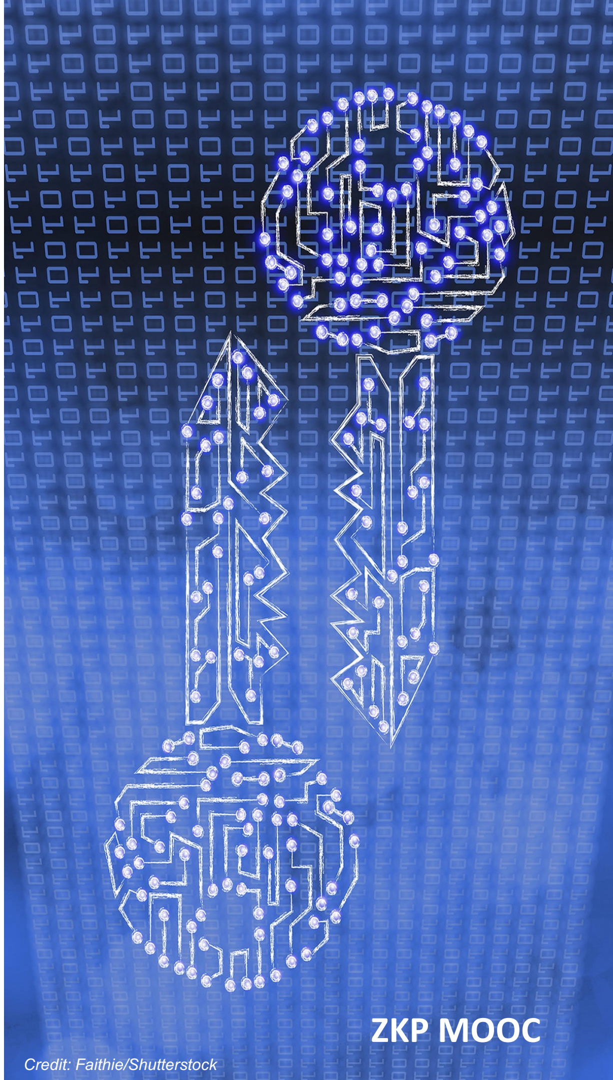
$view_V(P, V)$:
(s, b, z)

Simulator S works as follows:

1. First pick a random bit b.
2. pick a random $z \in Z_N^*$.
3. compute s $= z^2/y^b$.
4. If V*((N,y),s) = b output (s, b, z)
   if not goto 1 and repeat

Claim: Expected number of repetitions is two

ZK proof of Knowledge



Prover seems to have proved more: theorem is correct and that she "knows" a square root mod N

Consider $\mathbf{L_R} = \{\mathbf{x} : \exists \boldsymbol{w\ s.t.\ R(x,w) = accept}\}$ for poly-time relation R.

Def: (P,V) is a proof of knowledge (POK) for L_R if :

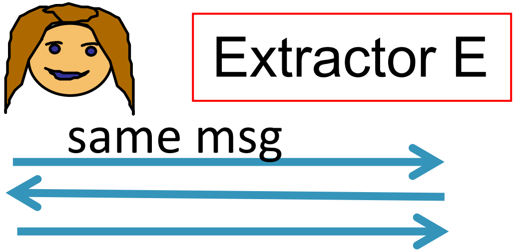
$\exists$ PPT (knowledge) extractor algorithm E s.t. $\forall$x in L,

in expected poly-time $E^P(x)$ outputs w s.t. R(x,w)=accept.

$E^P(x)$ (E may run P repeatedly on the same randomness)

possibly asking different questions in every executions

This is called the rewinding technique



Prover seems to have proved more not only that theorem is correct, but that she “knows” a square root mod N

Consider $\mathbf{L_R = \{x : \exists} \boldsymbol{w\ s.t.\ R(x,w) = accept}\}$ for poly-time relation R.

Def: (P,V) is a proof of knowledge (POK) for L_R if :

∃ PPT (knowledge) extractor algorithm E s.t. ∀x in L,

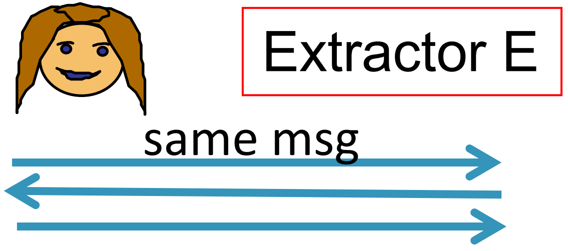
in expected poly-time $E^P(x)$ outputs w s.t. R(x,w)=accept.

[if Prob[(P,V)(x)=accept] > α, then $E^P(x)$ runs in expected poly(|x|,1/α) time]

$E^P(x)$ (may run P repeatedly on the same randomness)

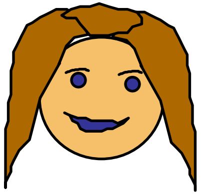
Possibly asking different questions in every executions

This is called the rewinding technique



ZKPOK that Prover knows a square root x of y mod N

Input: (y,N)

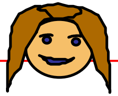


Extractor Algorithm

$s=r^2$ mod N →

← head

r →



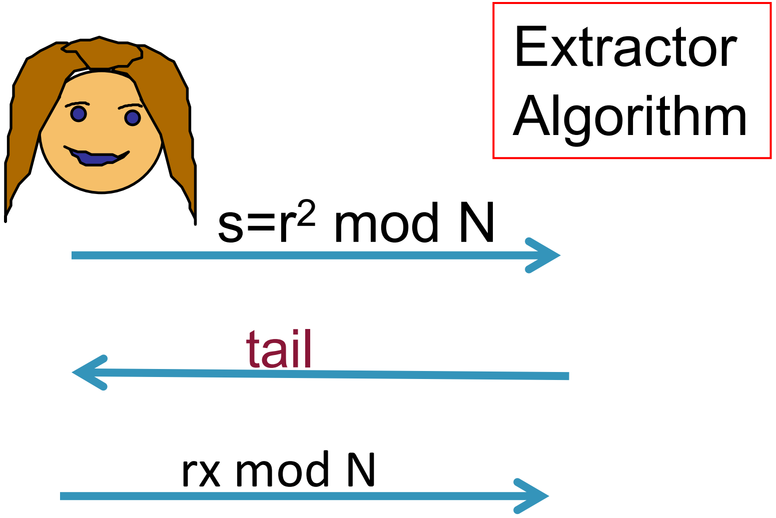
Extractor:

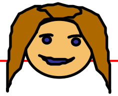
On input (y,N),

1. Run prover & receive s
2. Set verifier message to head; Store r

The Rewinding Method

Input: (y,N)



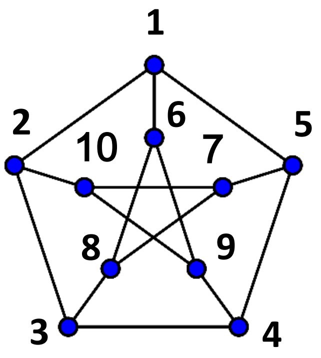


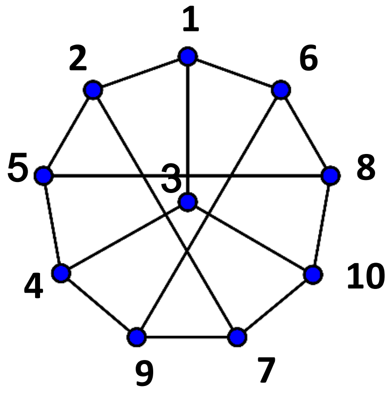
Extractor:

On input (y,N)

1. Run prover & receive s
2. Set verifier message to head; receive and store r
3. Rewind and 2^{nd} time set verifier message to tail receive rx
4. Output rx/r=x mod N

ZK Proof for Graph Isomorphism

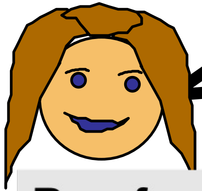




Recall:

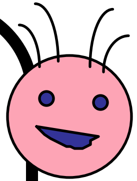
G_0 is isomorphic to G_1
If ∃ isomorphism $\boldsymbol{\pi}: [\boldsymbol{N}] \to [\boldsymbol{N}]$, $\forall i, j$: $\big(\pi(i), \pi(j)\big) \in E_1$ iff $(i, j) \in E_0$.

ZK Interactive Proof for Graph Isomorphism



I will produce a random graph H for which
1: I can give an isomorphism γ_0 from G_0 to H
OR
2: I can give an isomorphism γ_1 from G_1 to H
Thus, $\exists$isomorphism σ from G_0 to G_1

Verifier, please randomly choose if I should demonstrate my ability to do #1 or #2.

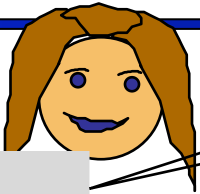


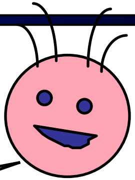
Proof:
$H = \gamma_0(G_0)$,
$H = \gamma_1(G_1)$,
Thus
$G_1 = \gamma_1^{-1}(\gamma_0(G_0))$
Set $\sigma = \gamma_1^{-1} \cdot \gamma_0$

POINT IS: If I can do both,
there exists an isomorphism from G_0 to G_1

REPEAT K
INDEPENDENT TIMES.

Input: (G_0, G_1)





Choose random γ_0 permutation of vertices of G_0. Set $H = \gamma_0(G_0)$

Graph H

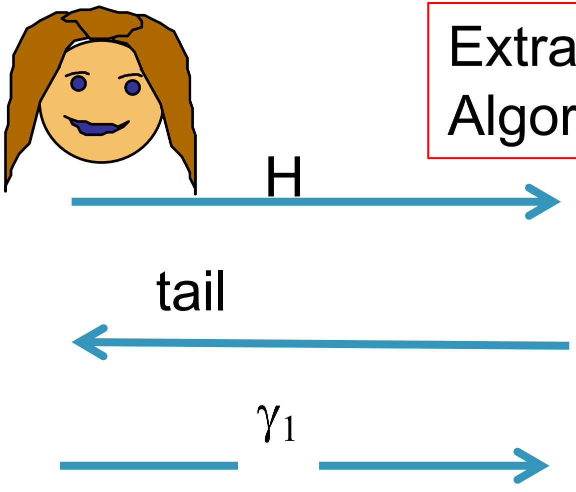
b

Toss coin b

If b=0: send γ_0
If b=1: send $\gamma_0\, \sigma^{-1}$ (where $\sigma(G_0) = G_1$)

Claims: (1) Statement true $\Rightarrow$ can answer correctly for b= 0 and 1
(2) Statement false $\Rightarrow$ $\text{prob}_b(\text{catch a mistake}) \geq 1 - 1/2^k$
(3) Perfect ZK [Exercise]

ZKPOK that **Prover knows an isomorphism** from G_1 to G_2

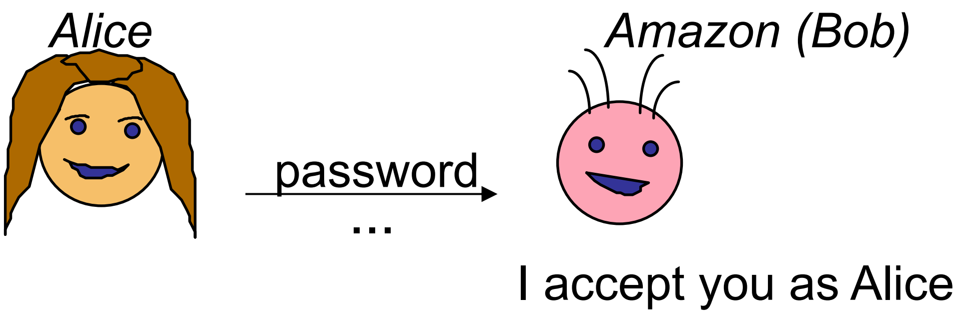


Extractor Algorithm

Extractor :

1) On input H
   set coin=head
   Store γ_0
2) Rewind and 2^{nd} time
   set coin=tail
   Store γ_1
3) Output $\gamma_1^{-1}(\gamma_0)$

The first application: Identity Theft [FS86]



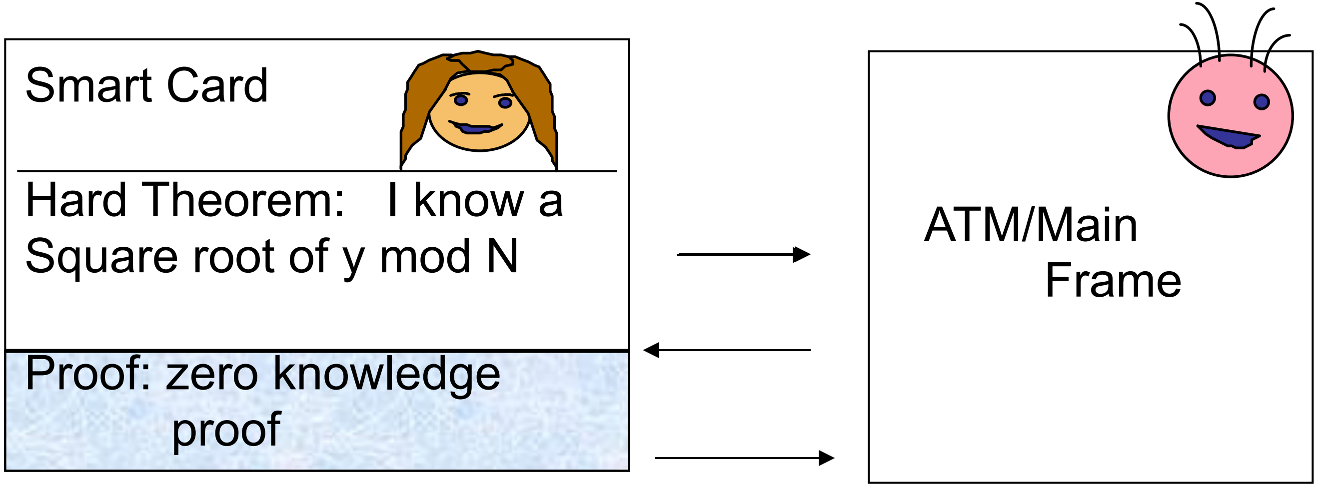
For Settings:

- Alice = Smart Card.
- Over the Net
- Breaking ins at Bob/Amazon are possible

Passwords are no good

Zero Knowledge: Preventing Identity Theft

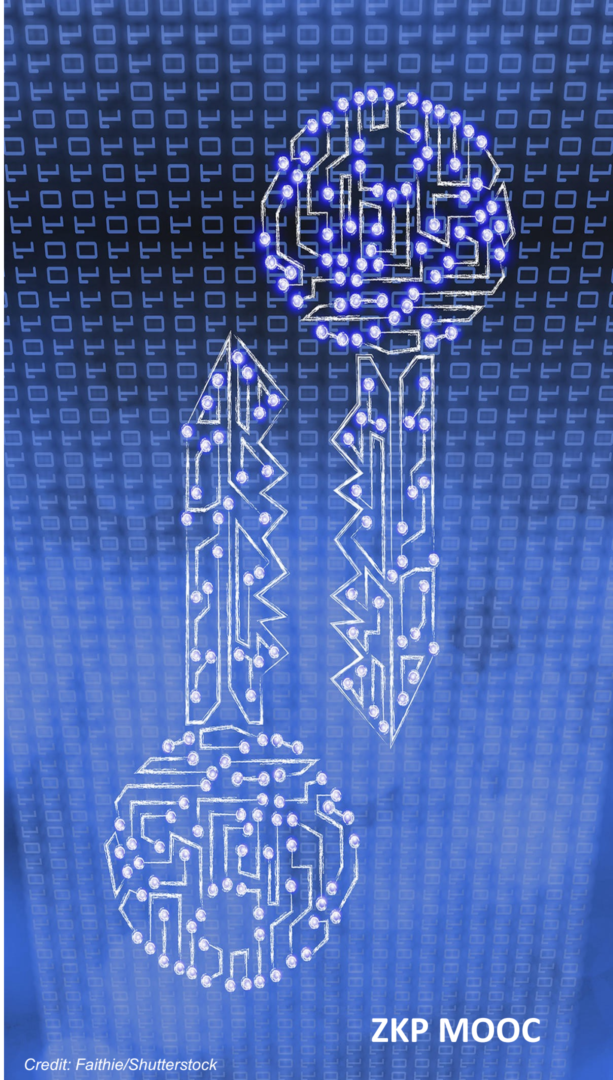
PROVER VERIFIER



To identify itself prover proves a hard theorem.

Interesting examples, one application

But, do all NP Languages have Zero Knowledge Interactive Proofs?



Credit: Faithie/Shutterstock