

Zero-Knowledge Interactive Proofs

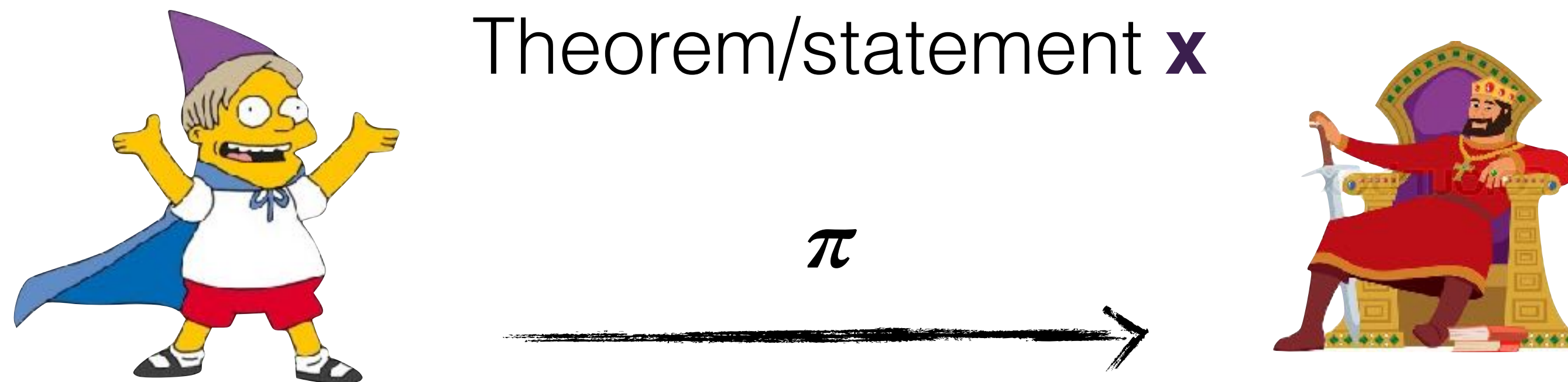
Michele Ciampi



THE UNIVERSITY
of EDINBURGH

Two parties for a proof

- Merlin (prover) has unbounded resources
- Arthur (verifier) has limited resources



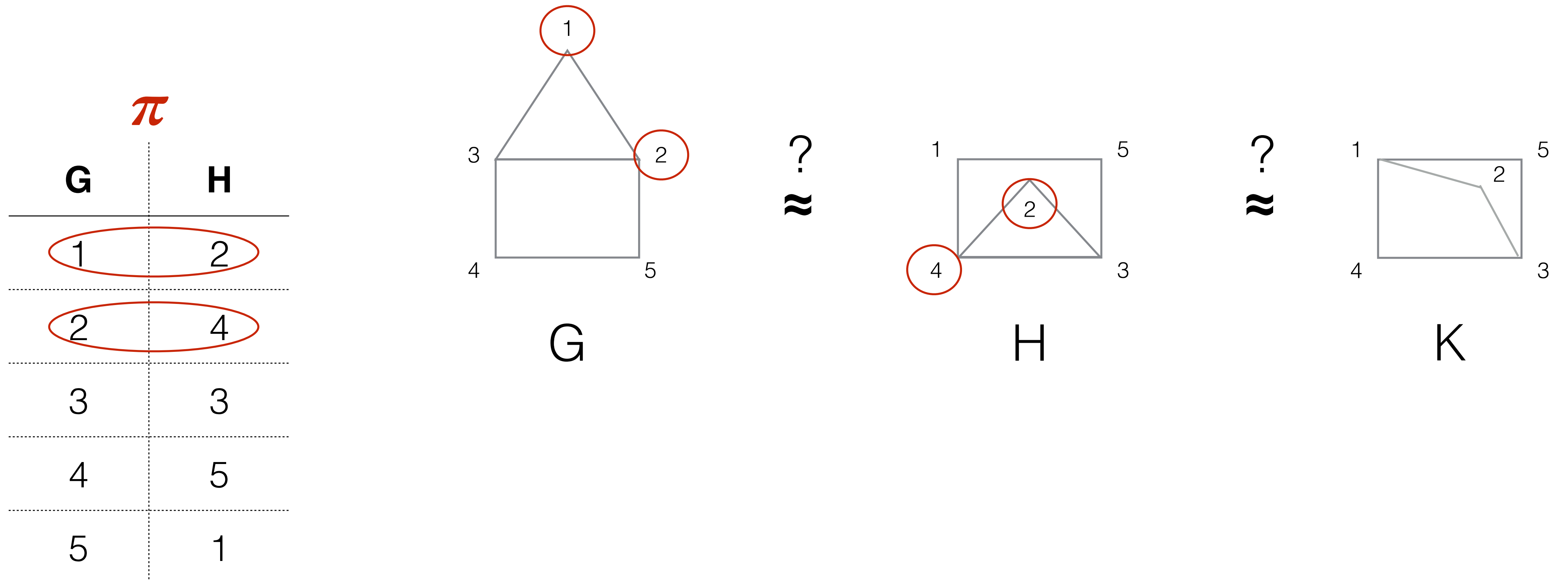
The proof is efficient: x is an NP statement and π is its certificate/witness/proof

Graph Isomorphism

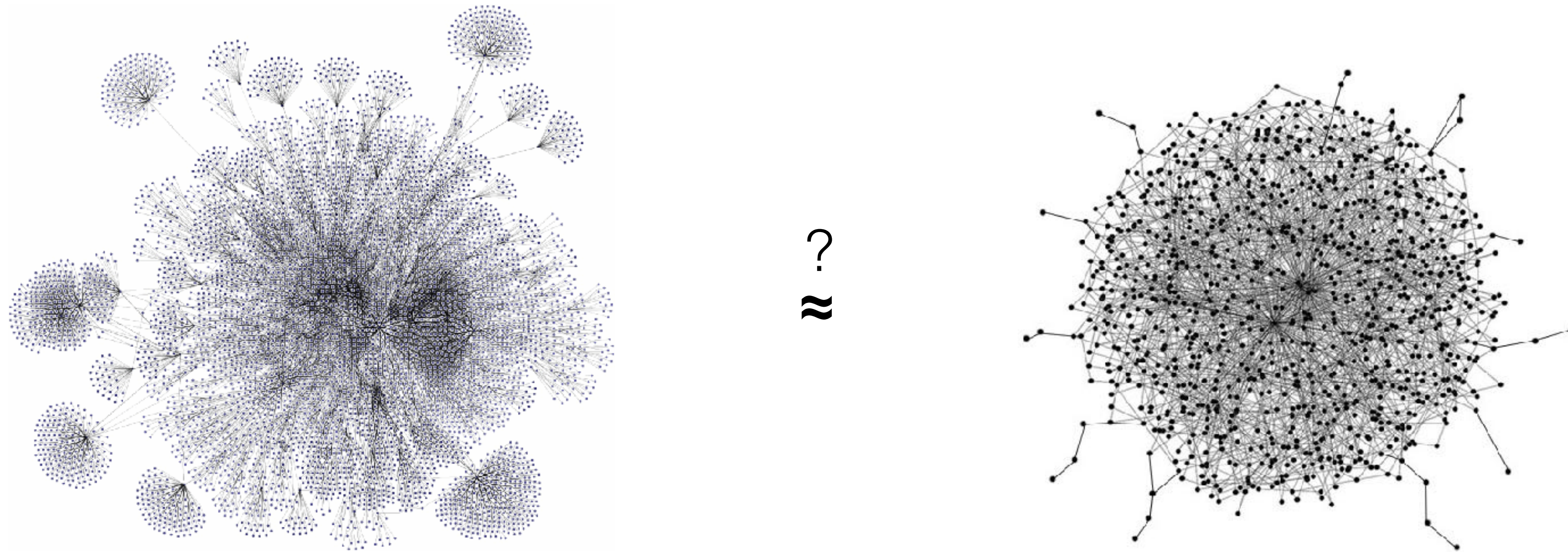
An isomorphism of graphs **G** and **H** is a **bijection** (permutation) π between the vertex sets of **G** and **H**

$$\pi: V(\mathbf{G}) \longrightarrow V(\mathbf{H})$$

such that any two vertices u and v of **G** are adjacent in **G** if and only if $\pi(u)$ and $\pi(v)$ are adjacent in **H**.



Graph Isomorphism



The problem belongs to NP

We do not know if it is in P: best known algorithm is
quasi-polynomial time

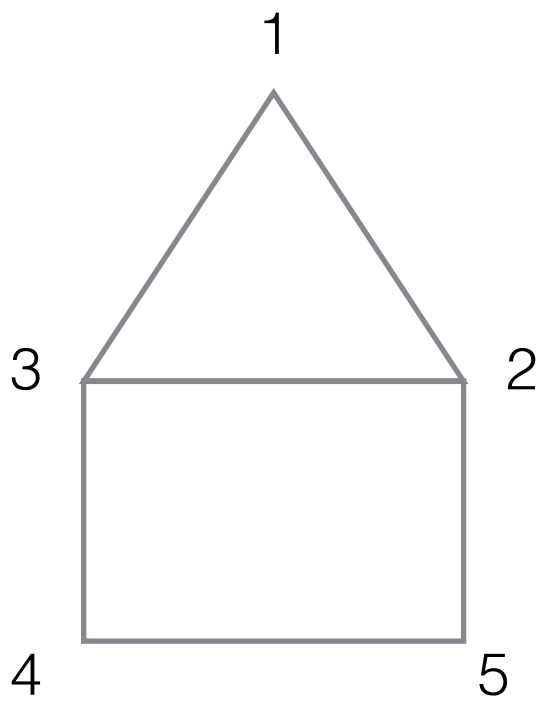
Graph Isomorphism

Witness

π

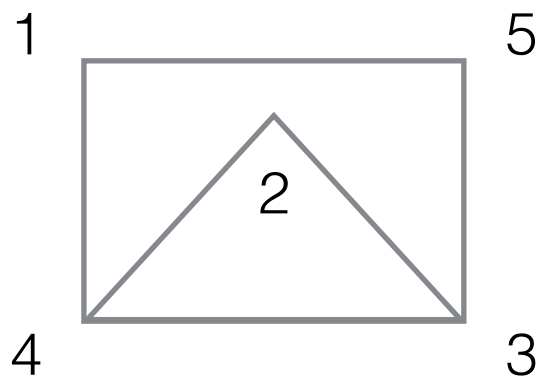
G	H
1	2
2	4
3	3
4	5
5	1

Thm



G

$\approx$



H



$\pi: 1 \rightarrow 2, 4 \rightarrow 1, \dots$



Interactive Proofs

- Suppose now that I want to prove that two graphs are **not isomorphic** or that an equation has no solutions.
- Introduced by Goldwasser, Micali and Rackoff
 - A proof is described as a game between a *prover* and a *verifier*
 - The theorem is true if and only if the prover wins the game always.
 - If the theorem is false then the prover loses the game with 50% probability



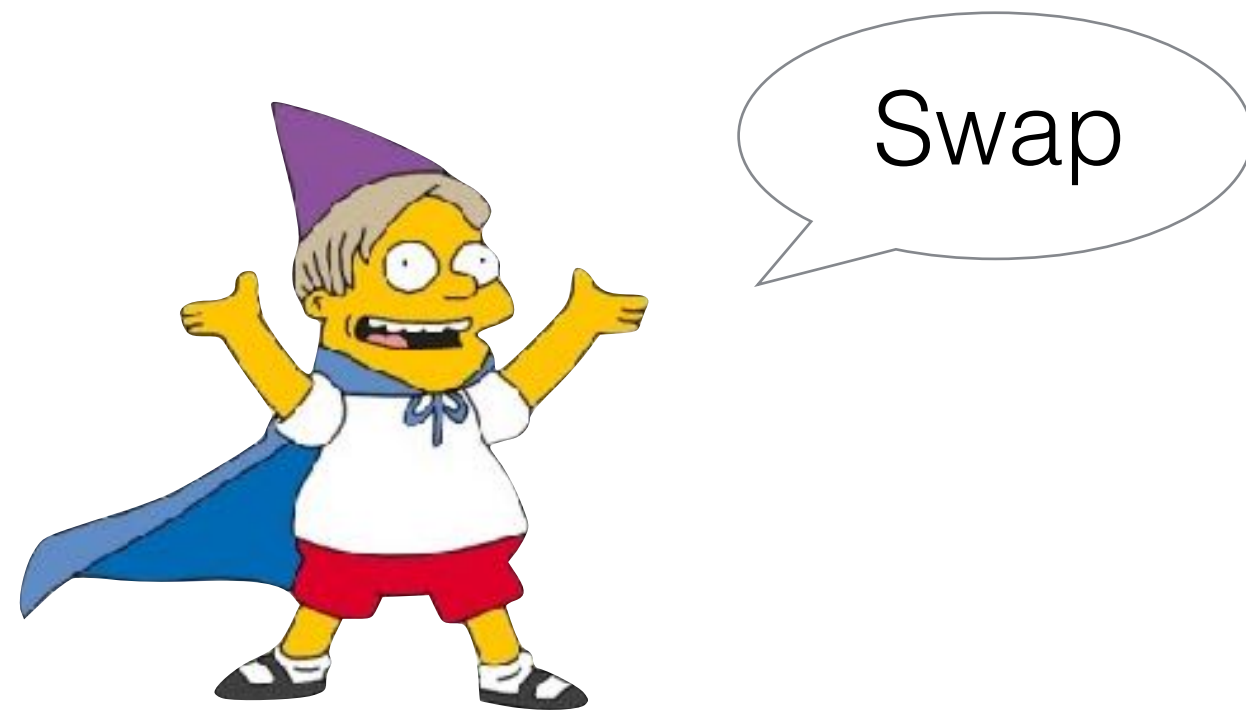
Prover (Merlin)



Verifier (Arthur)

Interactive Proofs

A simple example first



Prover



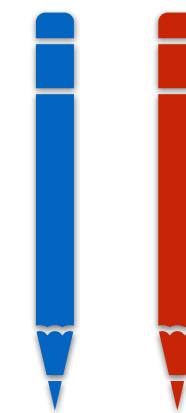
Verifier

Interactive Proofs

A simple example first



Prover



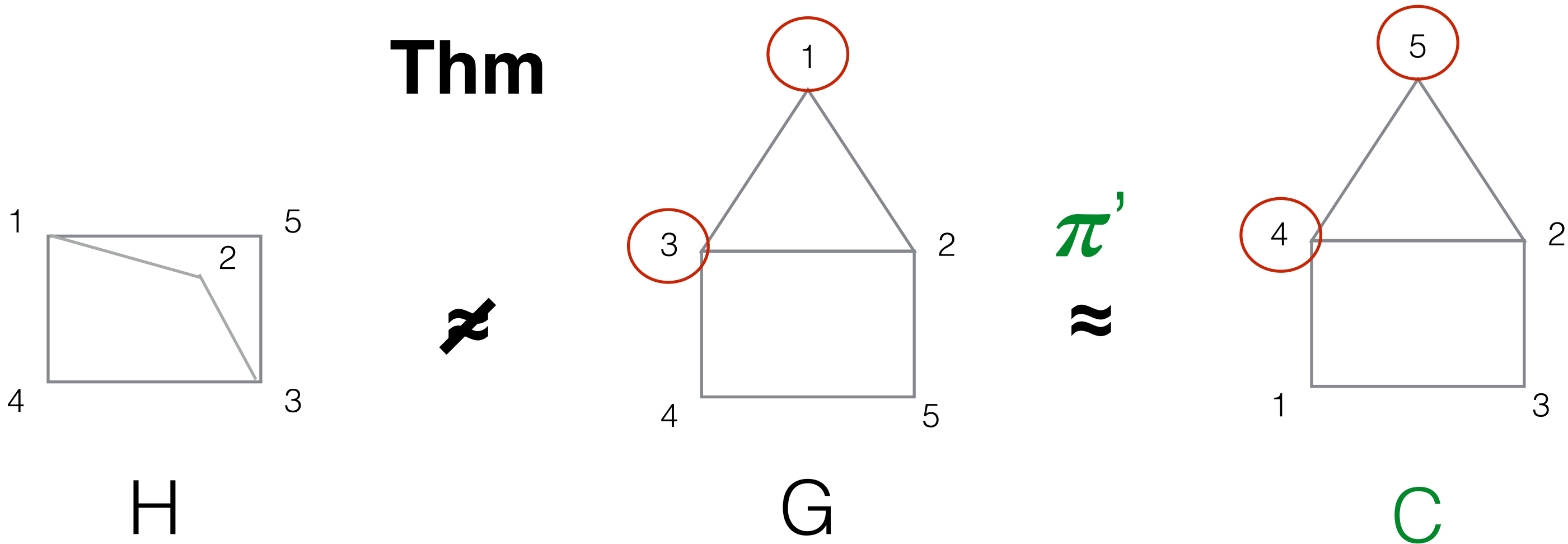
Verifier

If the pencils are both red, then the prover convinces the verifier with a 50% probability

We can repeat the proof many times to make this probability small

Graph Non-Isomorphism

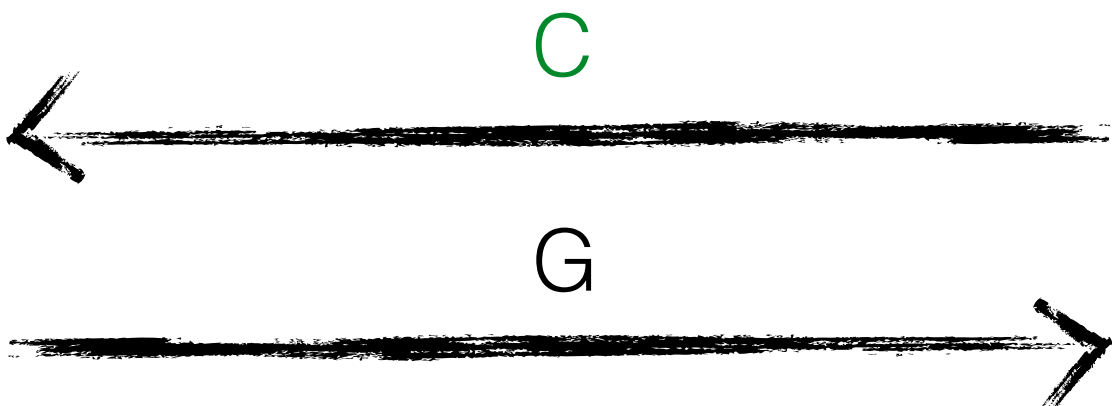
Thm



C cannot be isomorphic to H (due to transitivity)



Unbounded



$G \leftarrow$

$\pi' \leftarrow$ Random permutations

$C \leftarrow \pi'(G)$



Poly

π'	
G	C
1	5
2	2
3	4
4	1
5	3

Interactive Proofs (formal definition)

Definition 4.2.6 (Generalized Interactive Proof): Let $c, s : \mathbb{N} \rightarrow \mathbb{R}$ be functions satisfying $c(n) > s(n) + \frac{1}{p(n)}$ for some polynomial $p(\cdot)$. An interactive pair (P, V) is called a (generalized) interactive proof system for the language L , with **completeness bound** $c(\cdot)$ and **soundness bound** $s(\cdot)$, if

- (modified) completeness: for every $x \in L$,

$$\Pr[\langle P, V \rangle(x) = 1] \geq c(|x|)$$

- (modified) soundness: for every $x \notin L$ and every interactive machine B ,

$$\Pr[\langle B, V \rangle(x) = 1] \leq s(|x|)$$

In the previous example $c(|x|)=1$ and $s(|x|)=1/2$

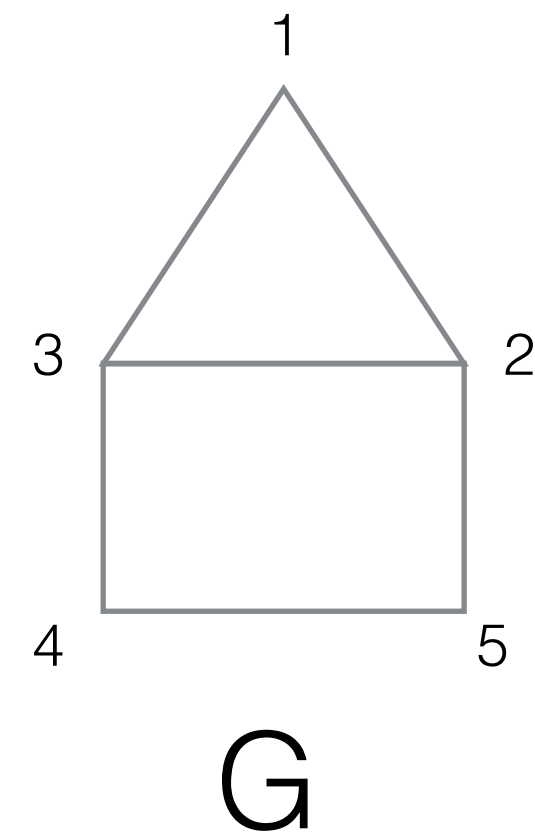
Zero-Knowledge (ZK)

Witness

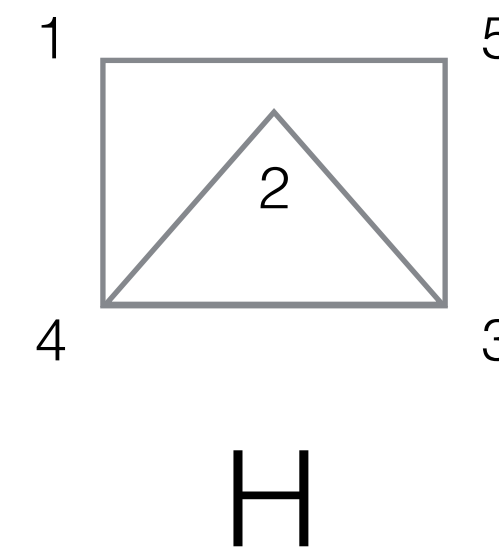
Thm

π

G	H
1	2
2	4
3	3
4	5
5	1



$\approx$



$\pi: 1 \rightarrow 2, 2 \rightarrow 4, \dots$



- How much knowledge is transmitted to the verifier?
- We would like to transmit only one bit: 1 if the theorem is true and 0 otherwise.
- E.g. For the case of graph isomorphism, the prover does not want to disclose the witness

ZK for Graph Isomorphism

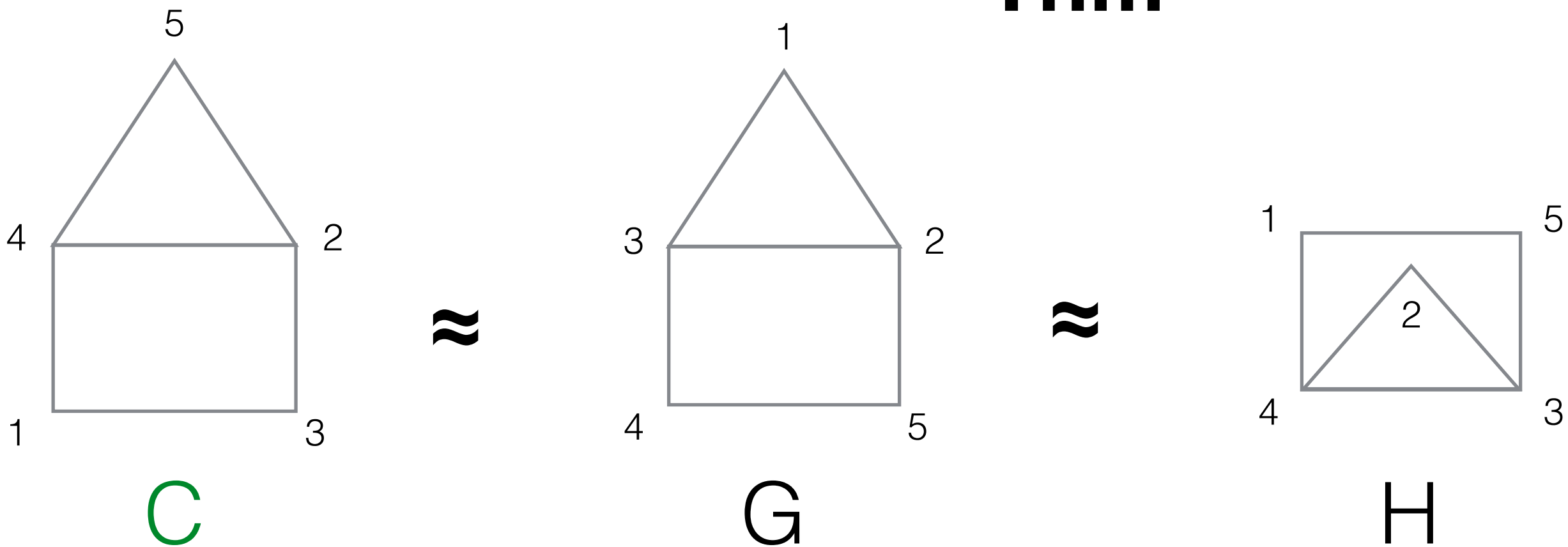
Witness

π		π'	
G	H	G	C
1	2	1	5
2	4	2	2
3	3	3	4
4	5	4	1
5	1	5	3

$C \leftarrow \pi'(G)$



Thm



ok, $G \approx C$

$G \leftarrow$

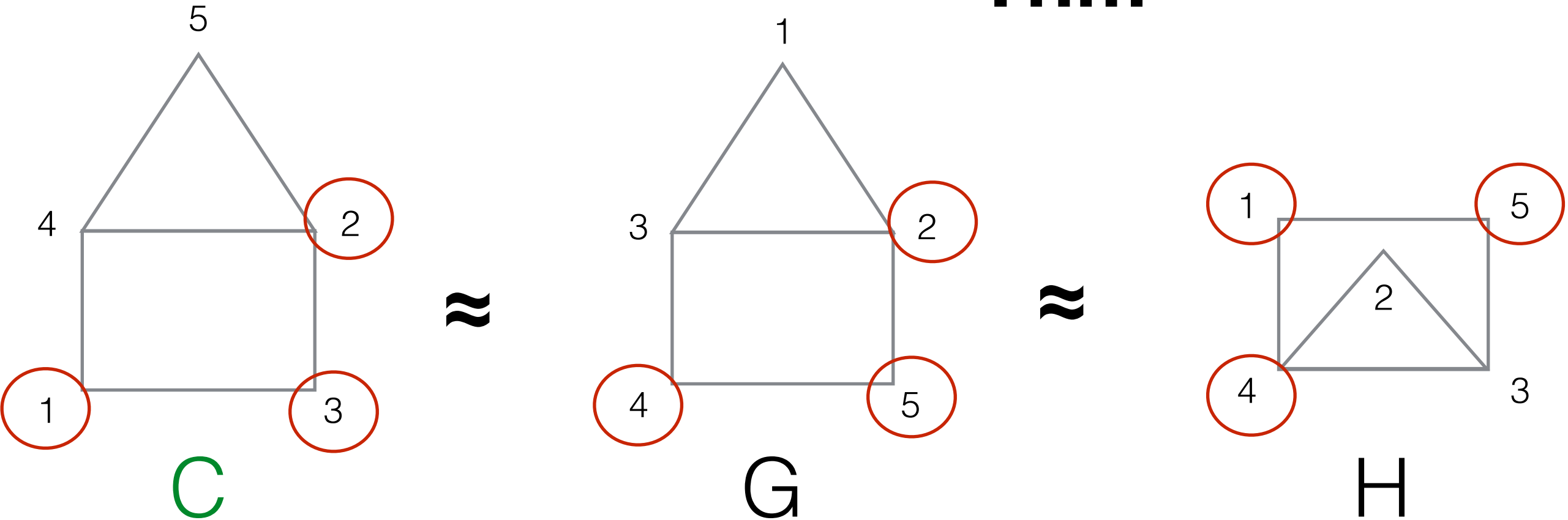


ZK for Graph Isomorphism

Witness

π		π'		π''	
H	G	G	C	C	H
2	1	1	5	1	
4	2	2	2	2	
3	3	3	4	3	
5	4	4	1	4	
1	5	5	3	5	

$C \leftarrow \pi'(G)$



Thm

ok, $H \approx C$

$H \leftarrow$



If the graphs are non-isomorphic then the prover convinces the verifier with a 50% probability

We can repeat the proof many times to make this probability small

Zero Knowledge

- The notion of zero knowledge requires the existence of a simulator **S** that:
 - knows **only** that the theorem is true
 - is efficient
 - generates a transcript that is distributed similarly* to the real one (when the verifier is honest)
 - has black-box access to the adversary

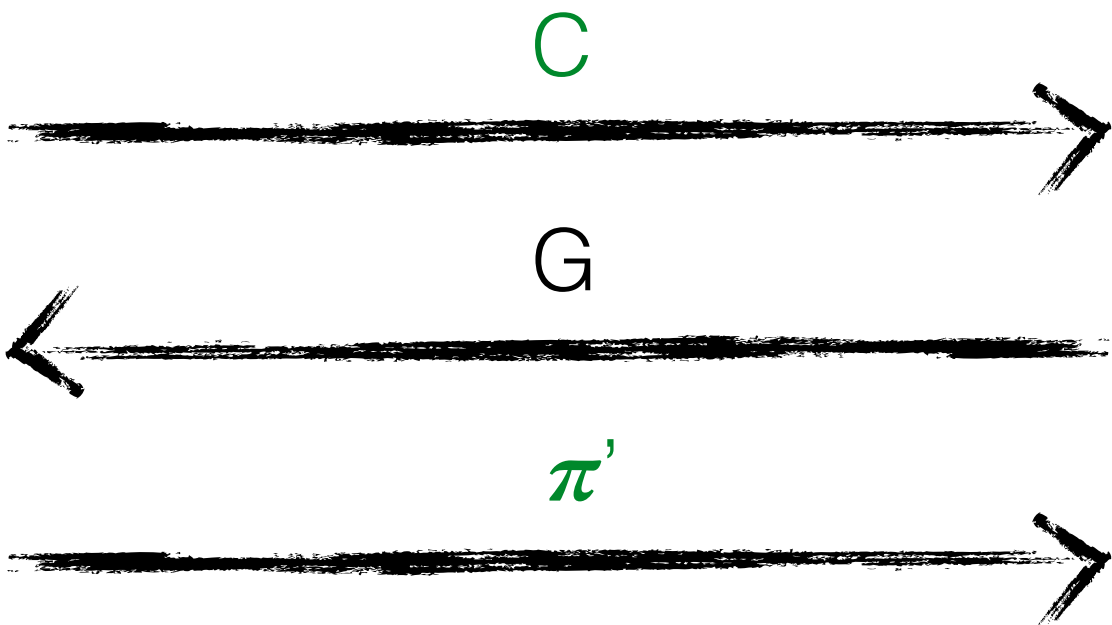
Honest-Verifier ZK for Graph Isomorphism

π

H	G
2	1
4	2
3	3
5	4
1	5



Thm $G \approx H$



ok, $G \approx C$

$G \leftarrow$ —



Sím

Thm $G \approx H$

$G \leftarrow$ —

π^*



$C \leftarrow \pi^*(G)$

Sigma protocols

- Completeness

Computational

- Honest Verifier Zero-Knowledge $\mathcal{HVZK}_{Sim}(x) \Rightarrow$

Special Honest Verifier Zero-Knowledge $\mathcal{SHVZK}_{Sim}(x, c) \Rightarrow a', z'$

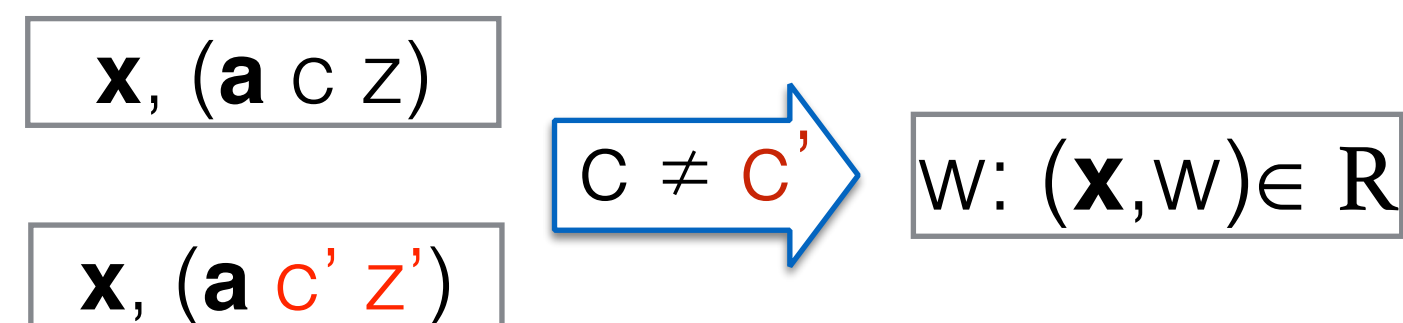
Computational

- Special Soundness

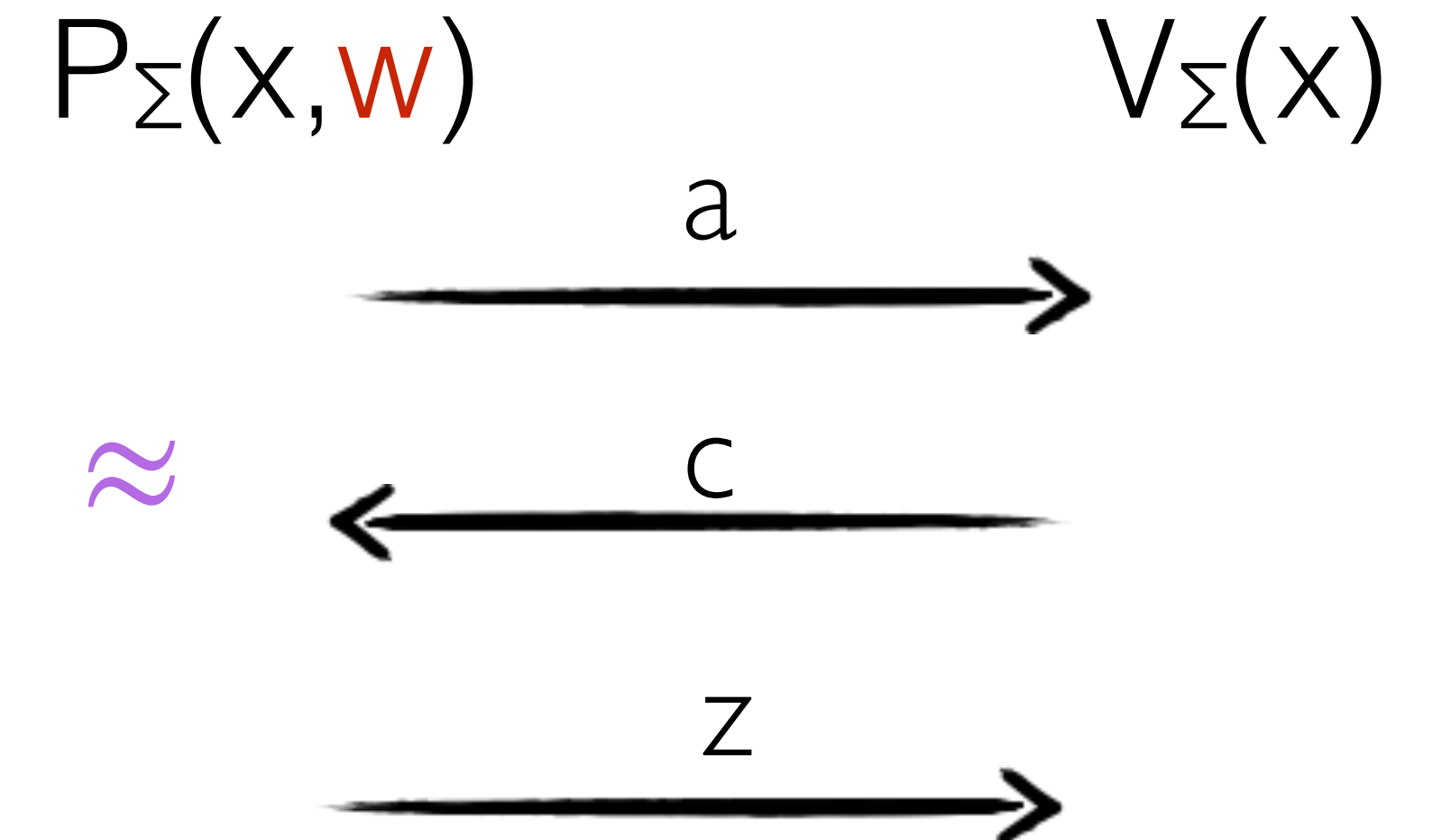
a'

c'

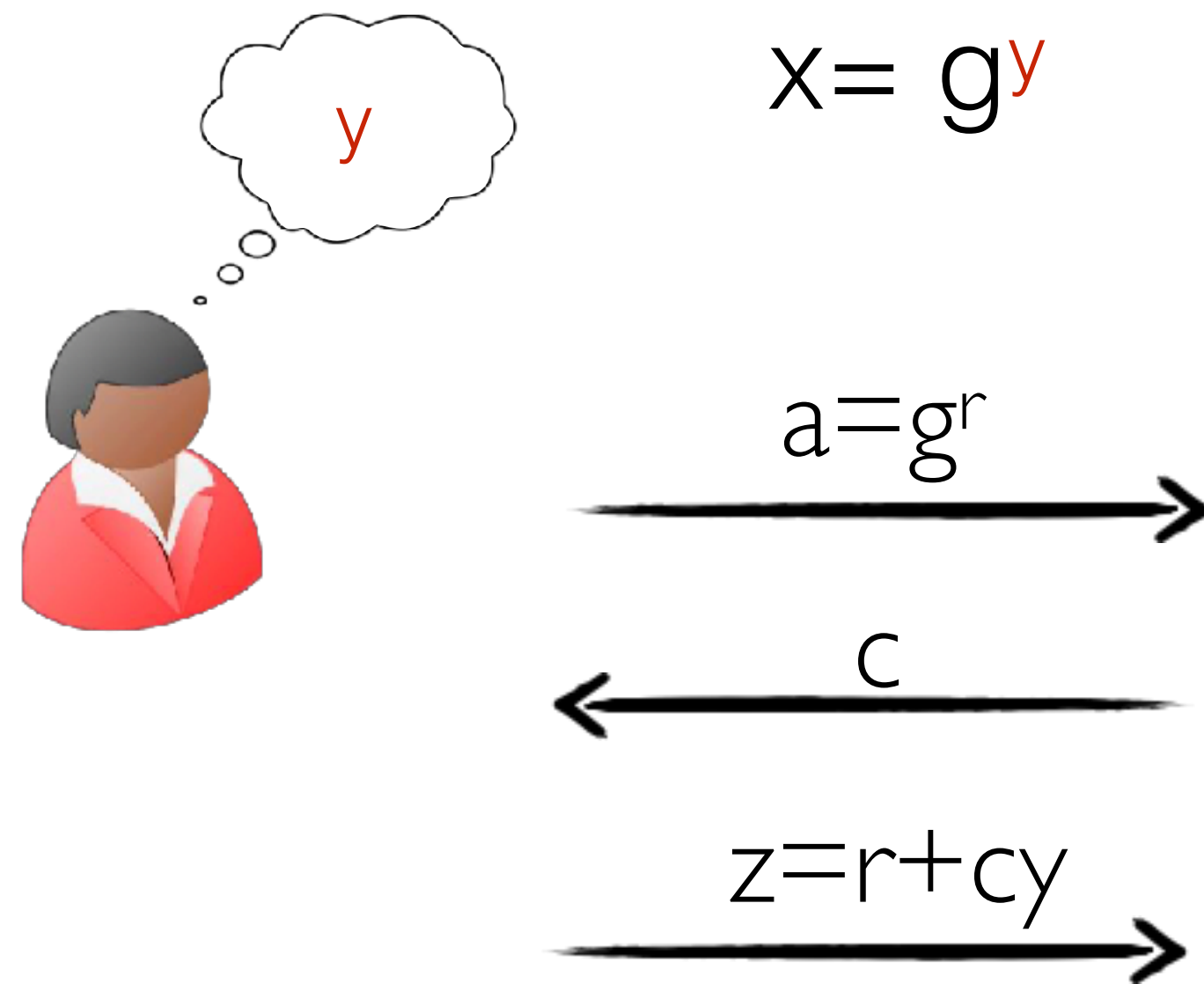
z'



Thm: x



Schnorr protocol

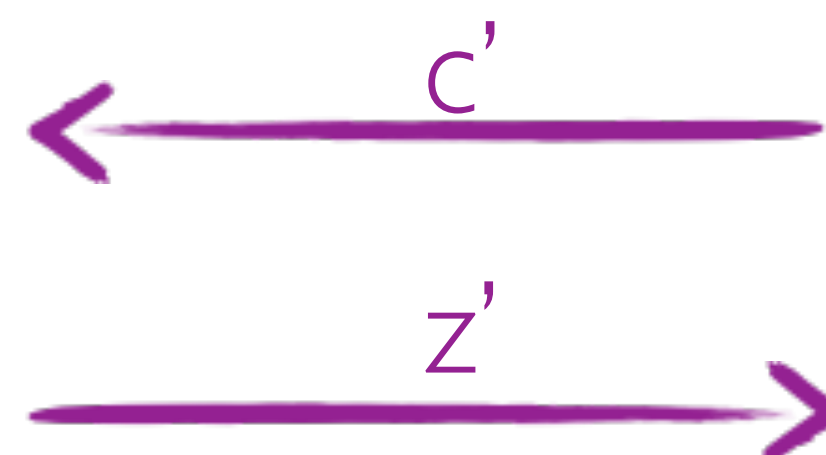
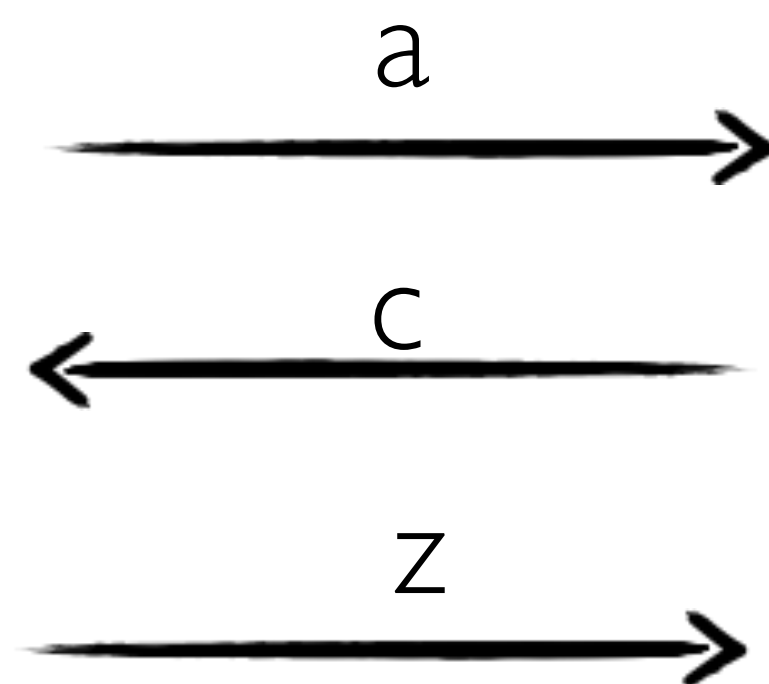


Let G be a group of order q ,
with generator g

Accept iff $g^z = ax^c$

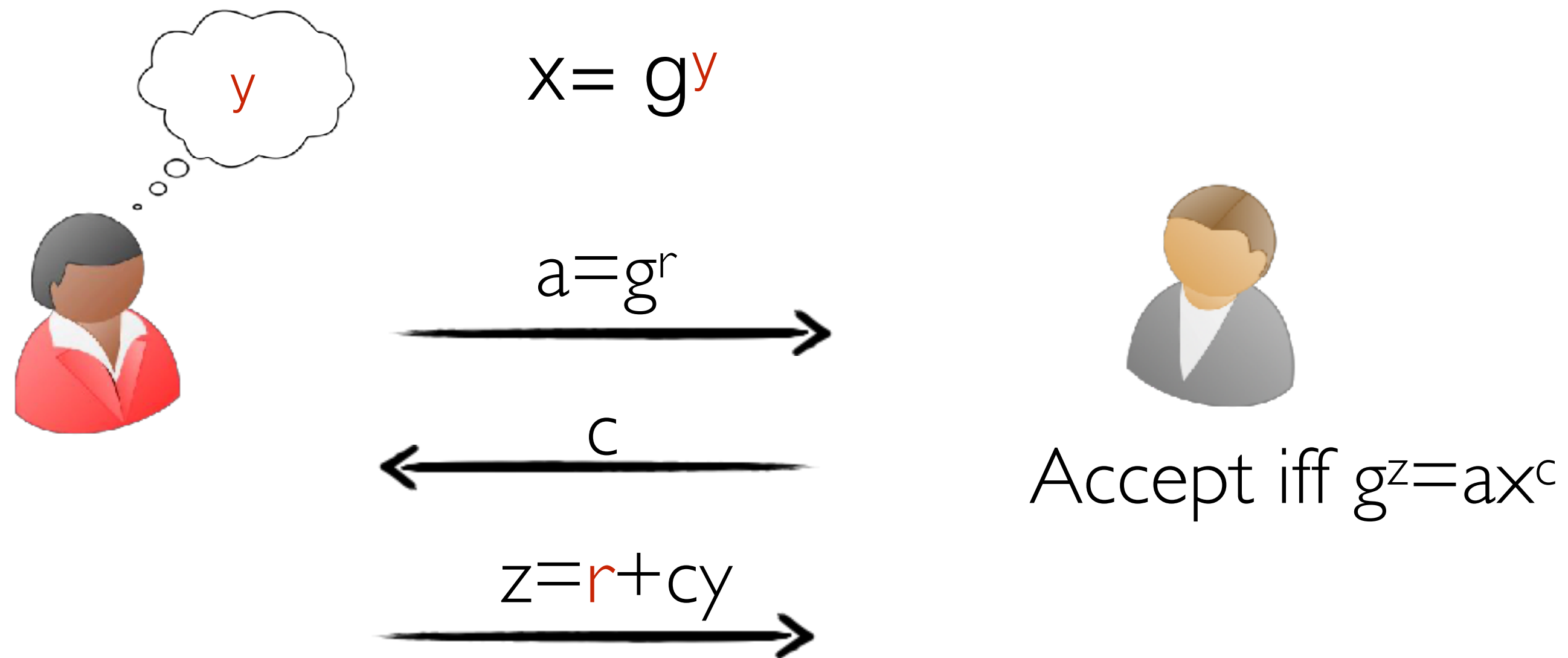
$$g^z = g^{r+cy} \quad ax^c = g^r g^{yc} = g^{r+cy}$$

Special-soundness



$$\begin{cases} z = r + cy \\ z' = r + c'y \end{cases} \xrightarrow{c \neq c'} y$$

Schnorr protocol



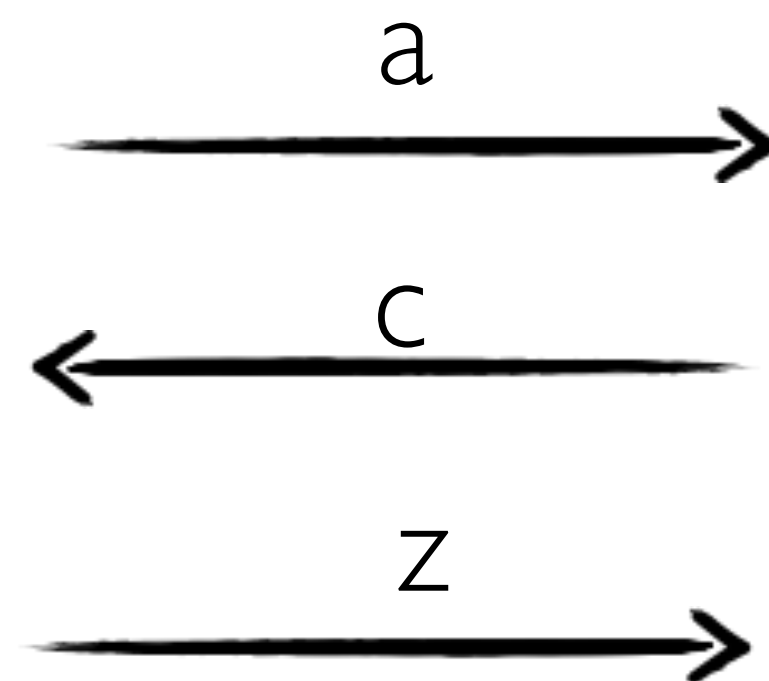
HVZK

$\mathcal{HVZK}_{sim}$

$$c \leftarrow \mathbb{Z}_q$$

$$z \leftarrow \mathbb{Z}_q$$

$$a = g^z / x^c$$



Sigma Protocol for Diffie-Hellman tuples

$x=(g, h, u,v)$

Is a DH tuple if

$u=g^y, v=h^y$

Let G be a group of order q ,
with generators g and h

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

if $b=0$ then

$T=(g, h, u=g^y, v=h^y)$

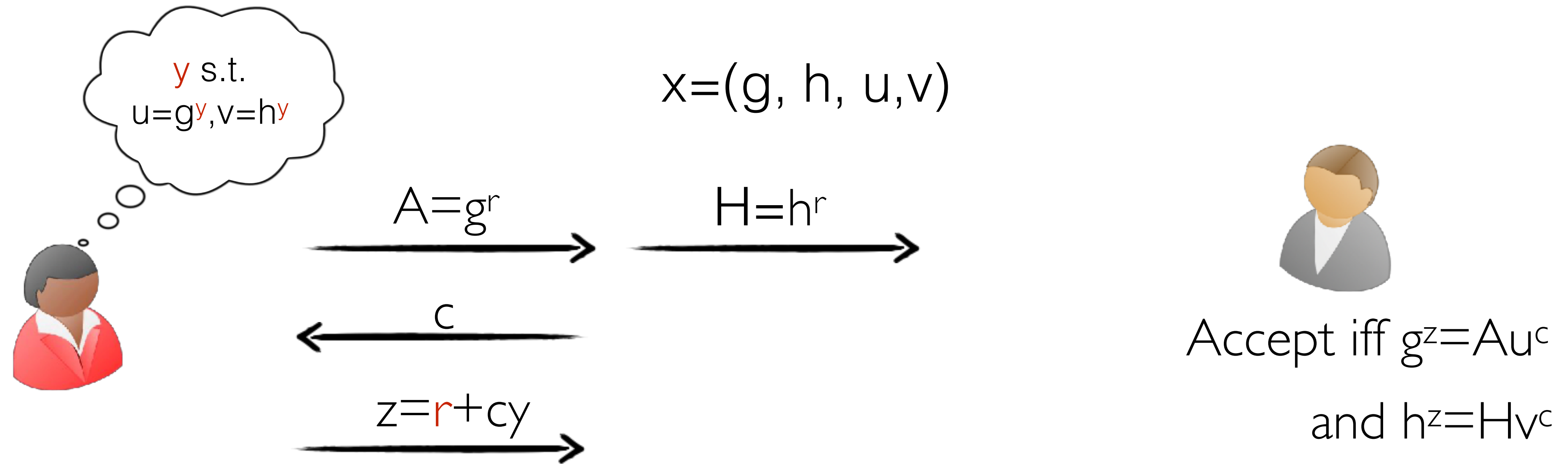
else

$T=(g, h, u=g^y, v=h^w)$ with $y \neq w$

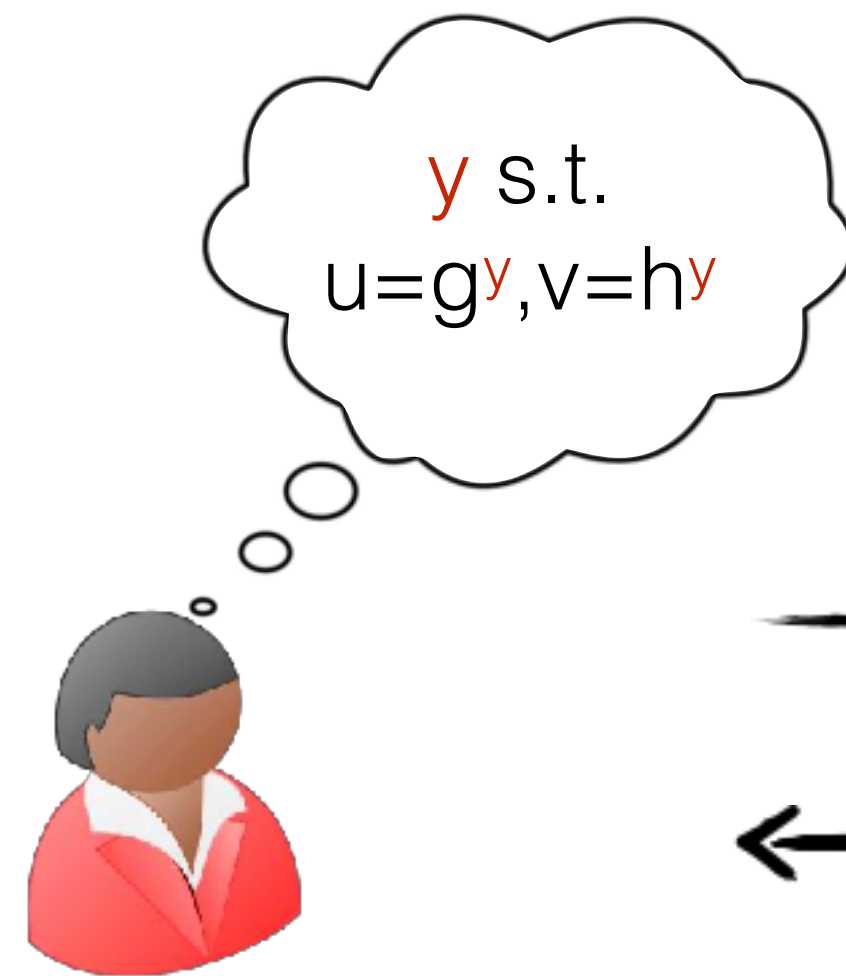
$\xrightarrow{T}$



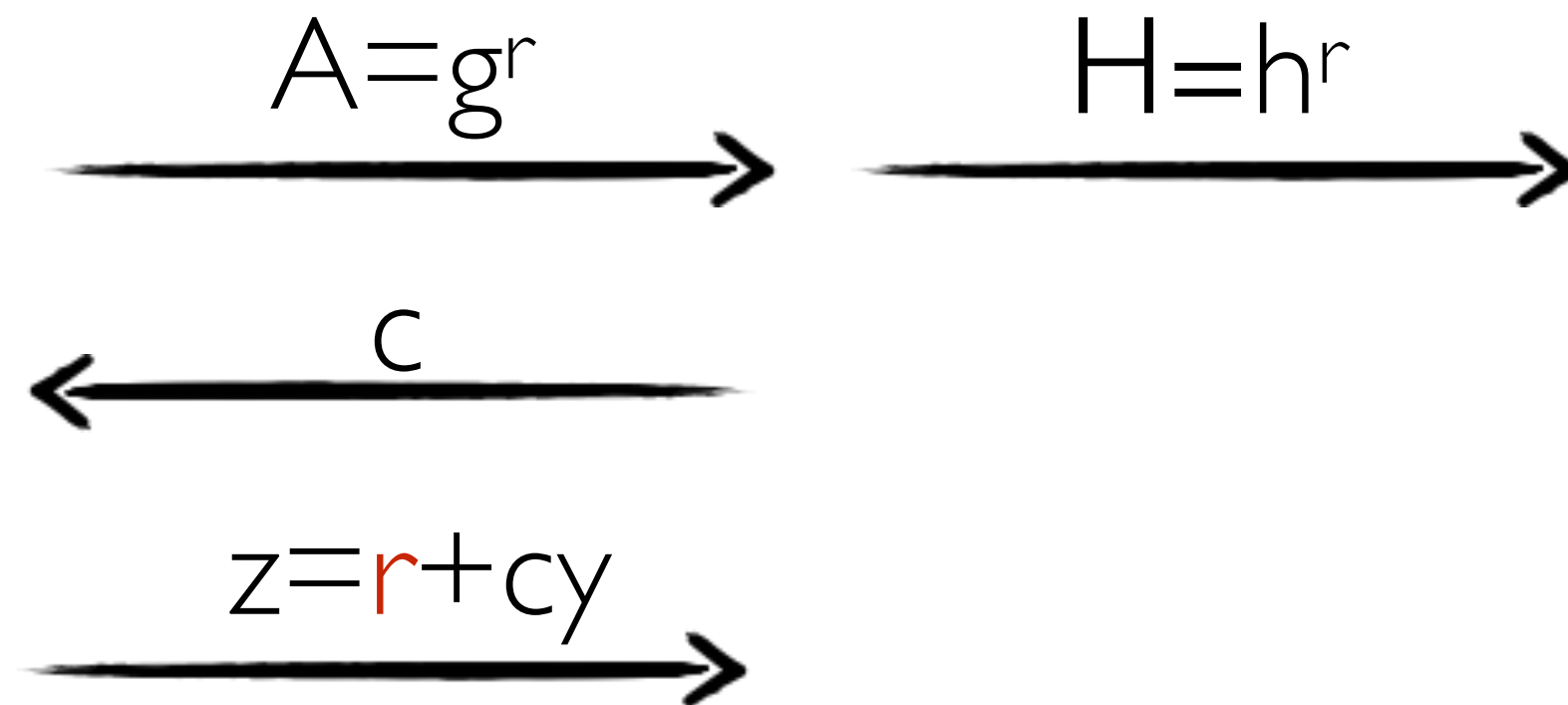
Sigma Protocol for Diffie-Hellman tuples



Sigma Protocol for Diffie-Hellman tuples



$$x=(g, h, u, v)$$



Accept iff $g^z=Au^c$
 and $h^z=Hv^c$

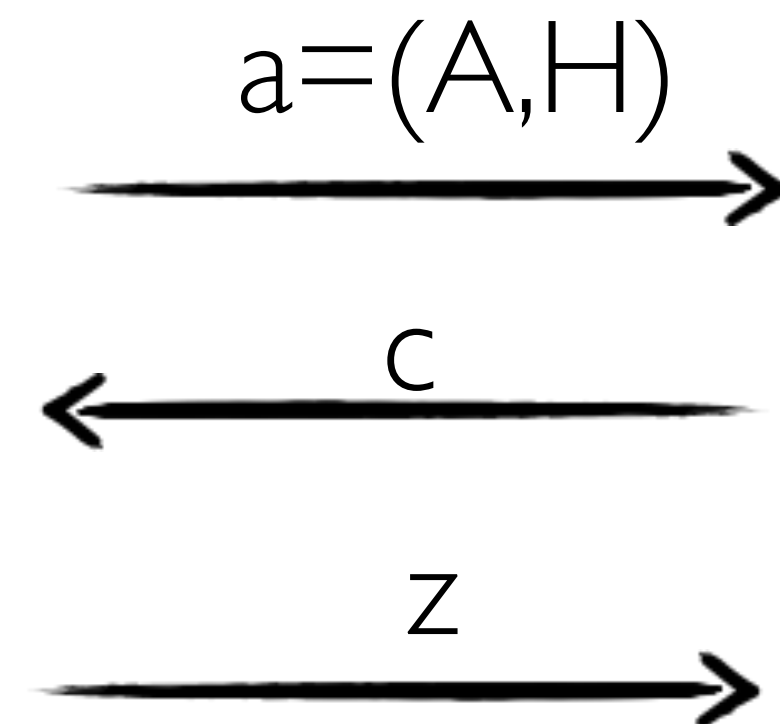
$\mathcal{HVZK}_{sim}$

$$c \leftarrow \mathbb{Z}_q$$

$$z \leftarrow \mathbb{Z}_q$$

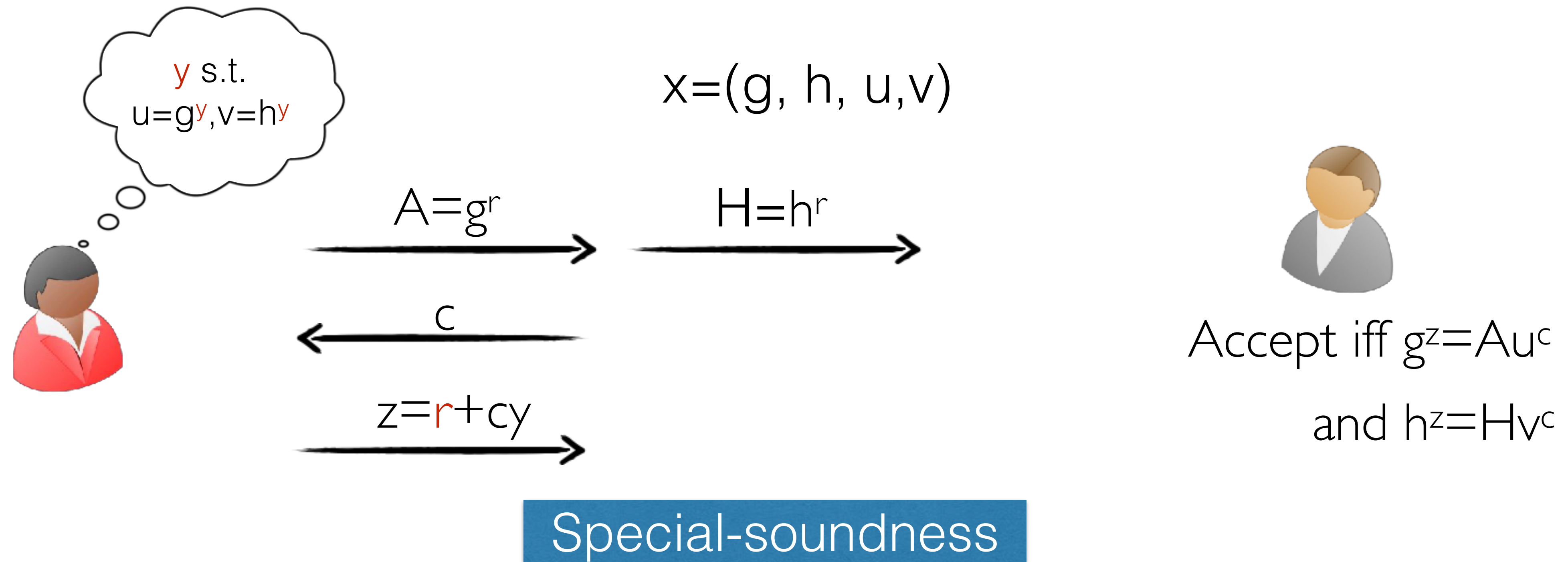
$$A=g^z/u^c$$

$$H=h^z/v^c$$



HVZK

Sigma Protocol for Diffie-Hellman tuples



Exactly the same as the one for the Dlog protocol

Why do we care?

- We know how to construct ZK proofs for any NP-language (with both efficient prover and verifier)
- CCA-encryption scheme
- Multi-party computation
- Identification schemes
- Privacy-preserving blockchains

Identification scheme



Password_{Alice}



Password₁

Password₂

....

Identification scheme



Password_{Alice}



Password₁

Password₂

....

Identification scheme

$$x = g^y$$



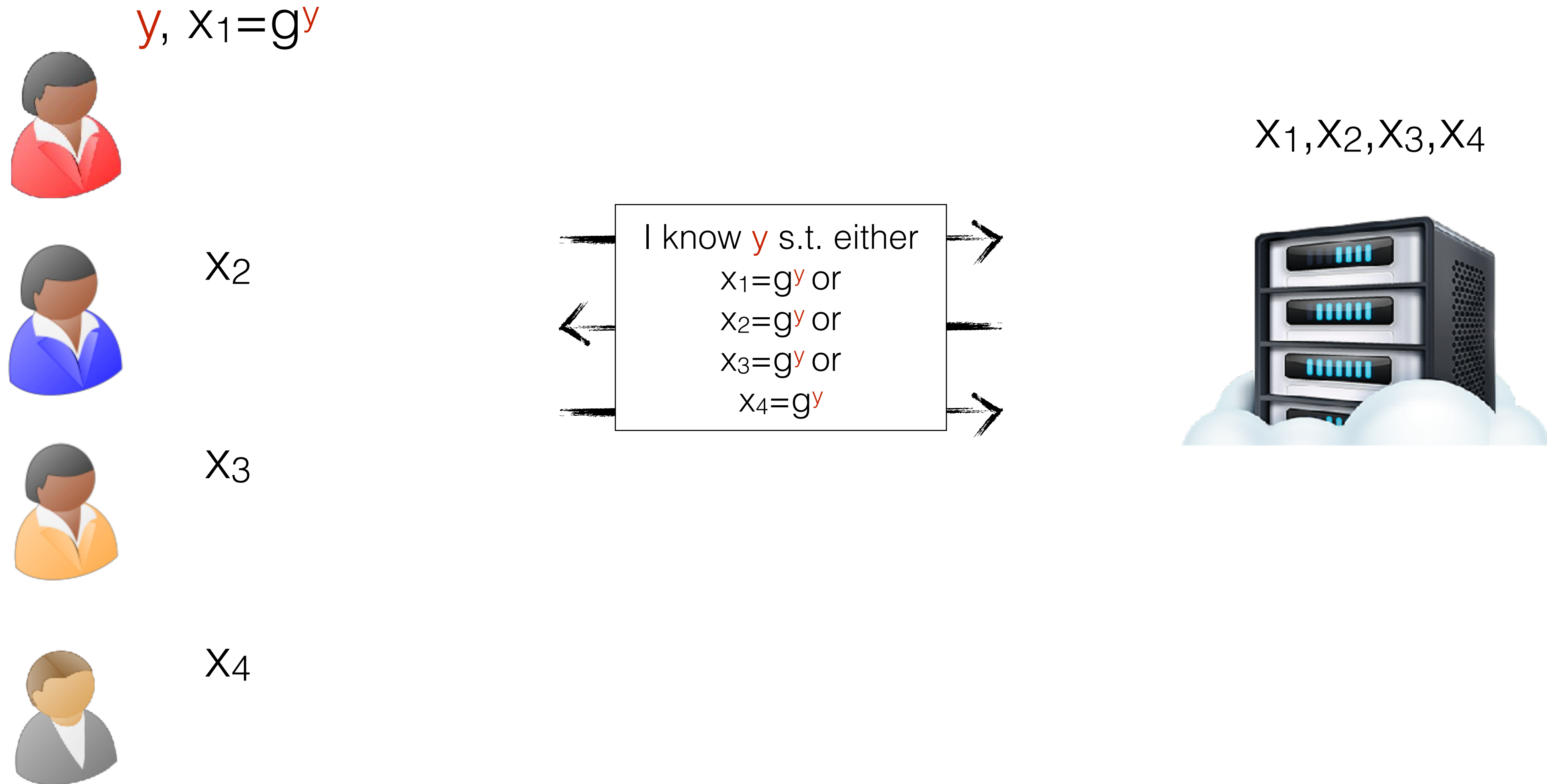
X



X



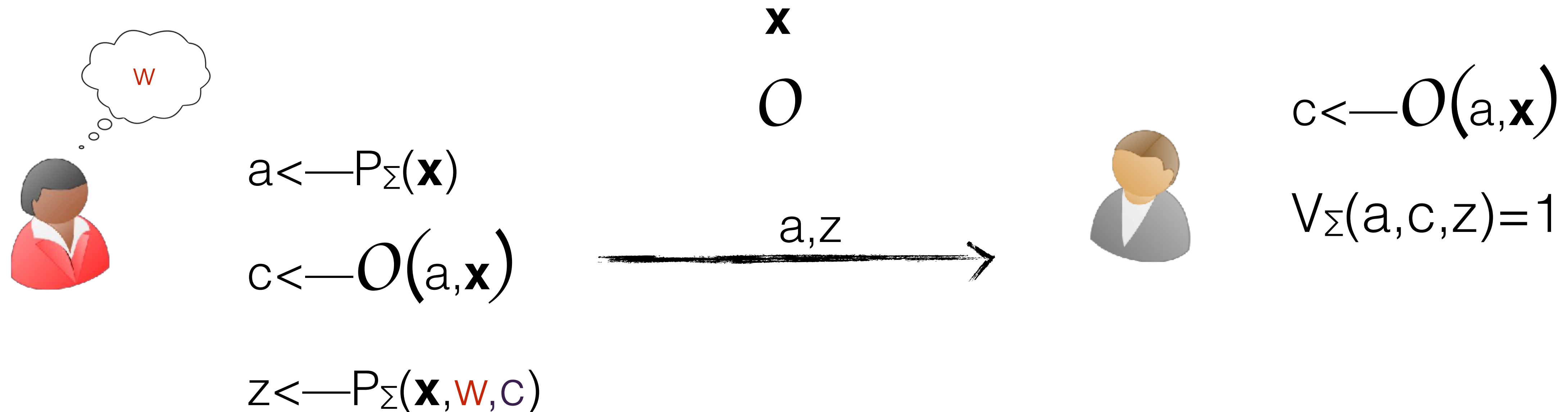
Identification scheme



Summary/Notes

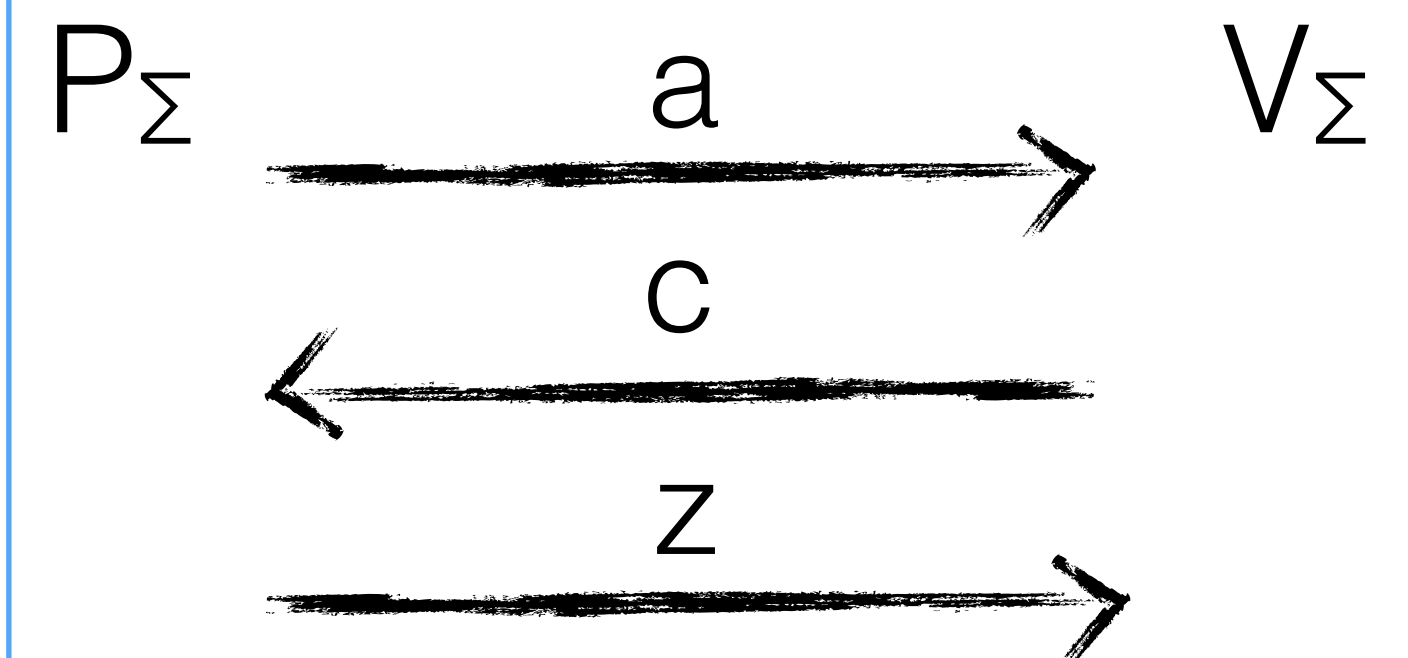
- Sigma-Protocol
- Every language in NP has a sigma-protocol
- Can we circumvent the 3-round impossibility and design an efficient non-interactive argument?

How do we make non-interactive proofs?



- Fiat-Shamir transform
- in practice O is a hash function (e.g. SHA2)

- Adds very little overhead to the starting sigma-protocol
- Used in practice for identification scheme, signatures, SNARKS, ...



Conclusions

- Non-interactive zero-knowledge (NIZK) proofs: length of the proof and verification time dependent on the NP language
 - Known from standard falsifiable assumptions
 - Setup is needed (just RO would suffice)
- SNARKs proofs: length of the proof depends on the security parameter and the verification time is dependent on the instance only
 - Setup is needed (even in the RO model)
 - Based on non-falsifiable assumptions (Knowledge of Exponent Assumptions)

End

References from the book of Goldreich Oded: Foundations of Cryptography: Volume 1, Basic Tools (see the link on learn)

- Sec. 4.2 until (included) Sec. 4.2.2 with no proofs
- Sec. 4.3 until (included) Sec. 4.3.2 with no proofs
- Sec. 4.7 until (included) Definition 4.7.2 with no proofs

More References on Sigma-Protocols: On Sigma-Protocols. Ivan Damgaard. <https://www.cs.au.dk/~ivan/Sigma.pdf>