From Sigma-Protocols to Zero-Knowledge in the Plain Model and Beyond

Michele Ciampi

THE UNIVERSITY OF EDINBURGH

Sigma protocols

- Completeness

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

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

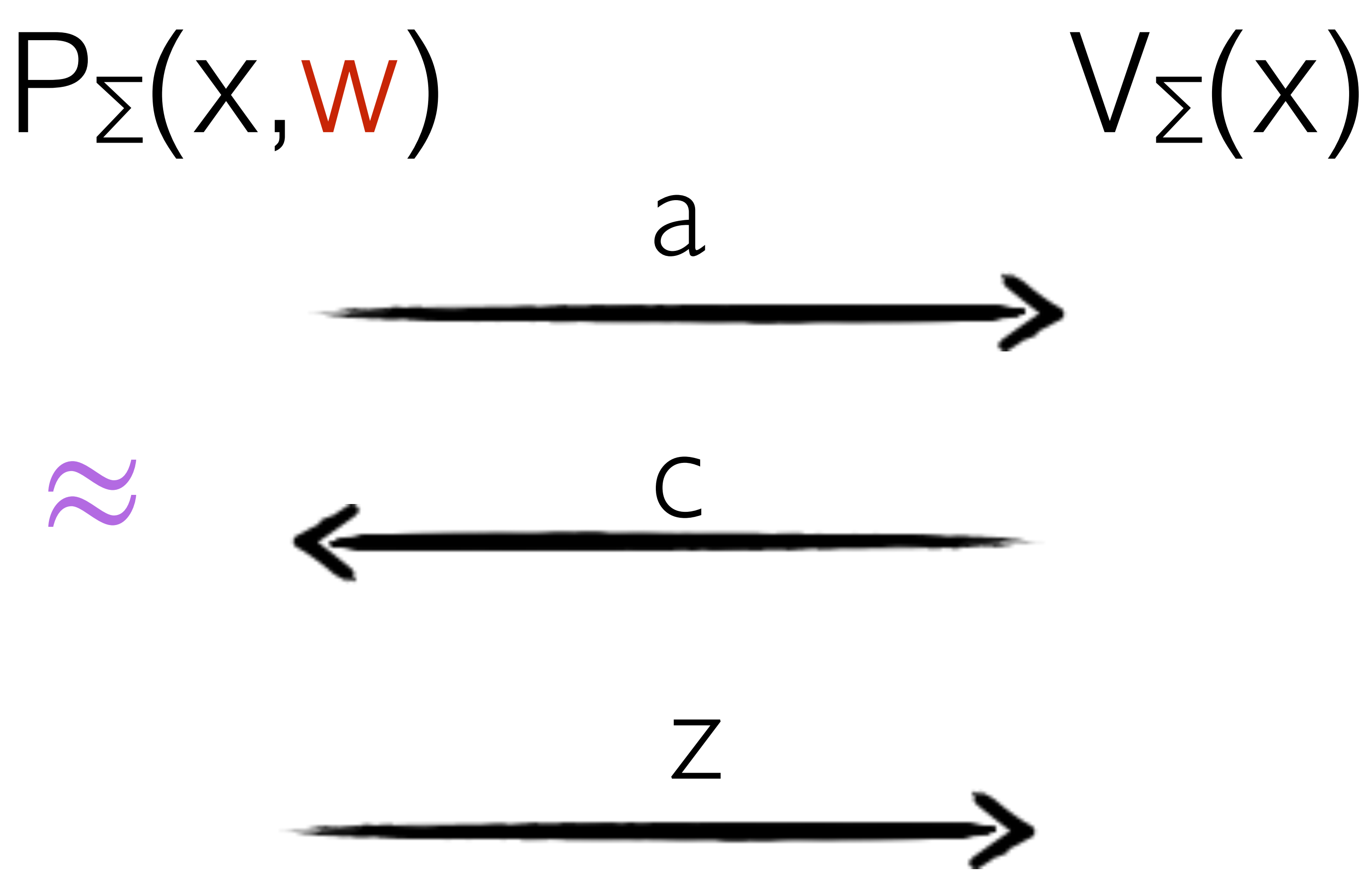
Computational
- Special Soundness

a'

c'

z'

Thm: x

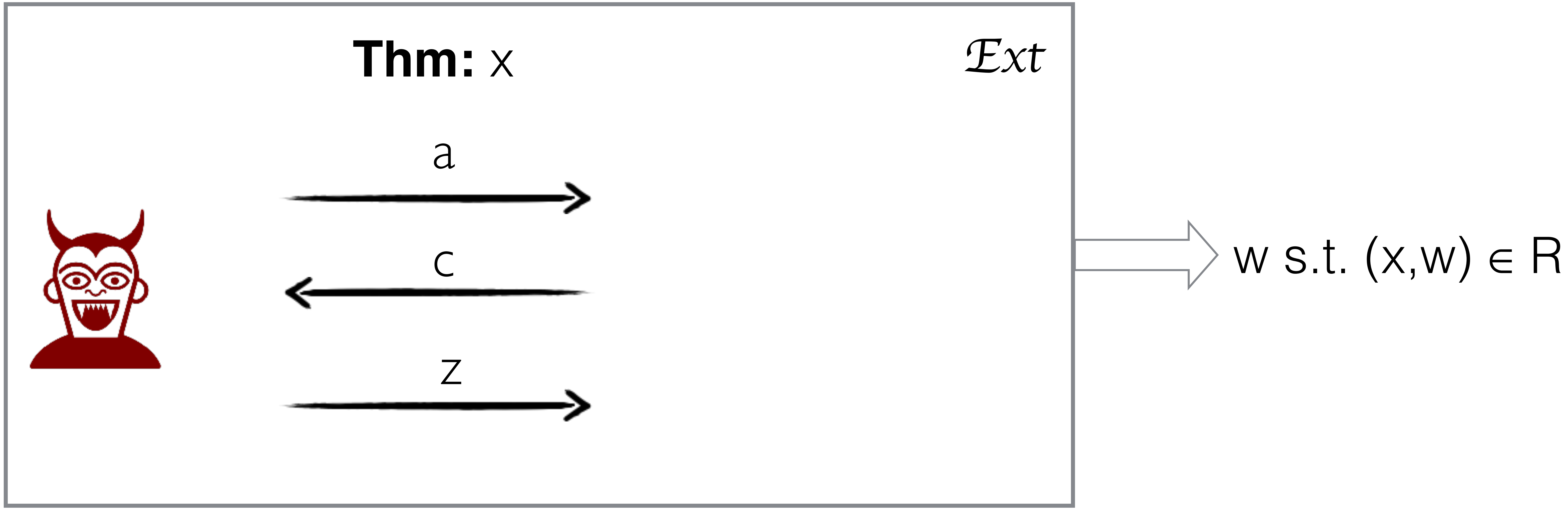


x, (**a** c z)

x, (**a** c' z')

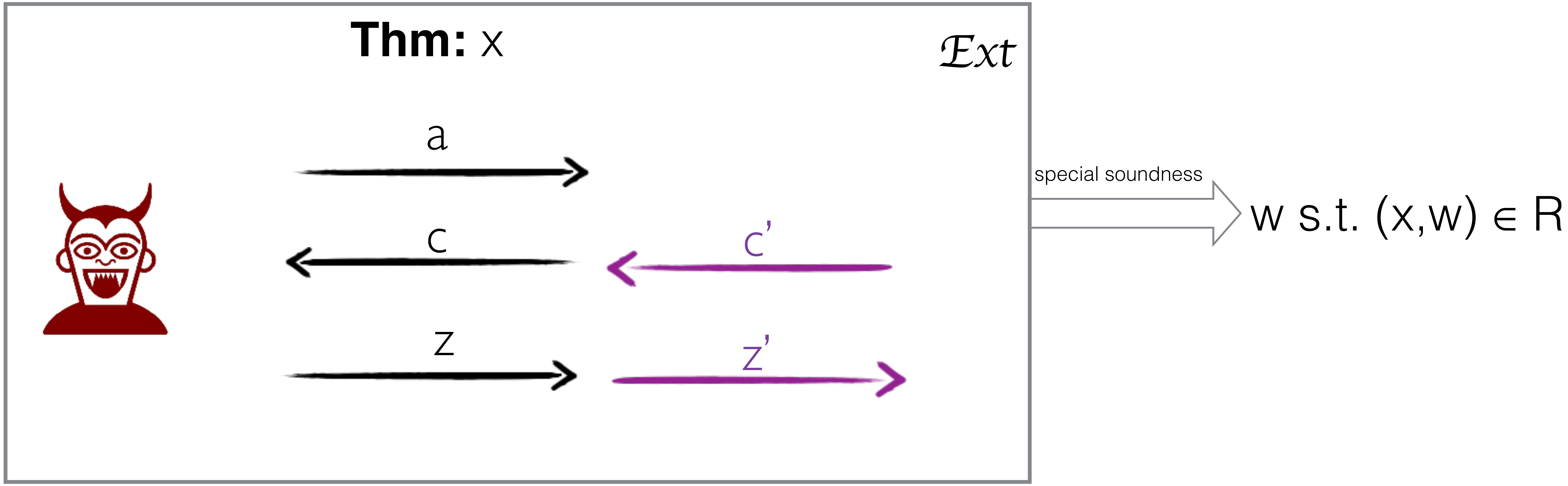
c ≠ c' ⇒ w: (**x**,w)∈ R

Proof of Knowledge



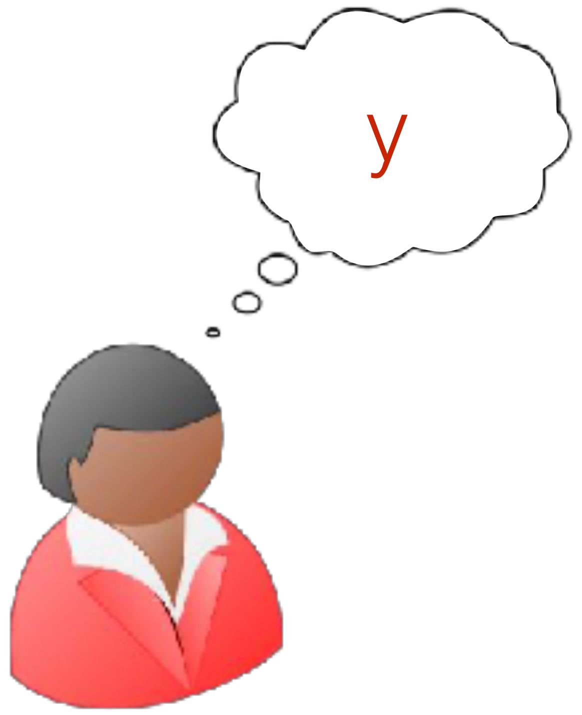
If the transcript is accepted with more than some probability p>k, then the extractor returns the witness in the expected time 1/(p-k) where k is the knowledge error

Special-soundness [D10] —> Proof of Knowledge



If the transcript is accepted with more than some probability p>k, then the extractor returns the witness in the expected time 1/(p-k) where k is the knowledge error

Schnorr protocol



$x = g^y$

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

$a = g^r$ →

← c

$z = r + cy$ →

Accept iff $g^z = a x^c$

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

Special-soundness

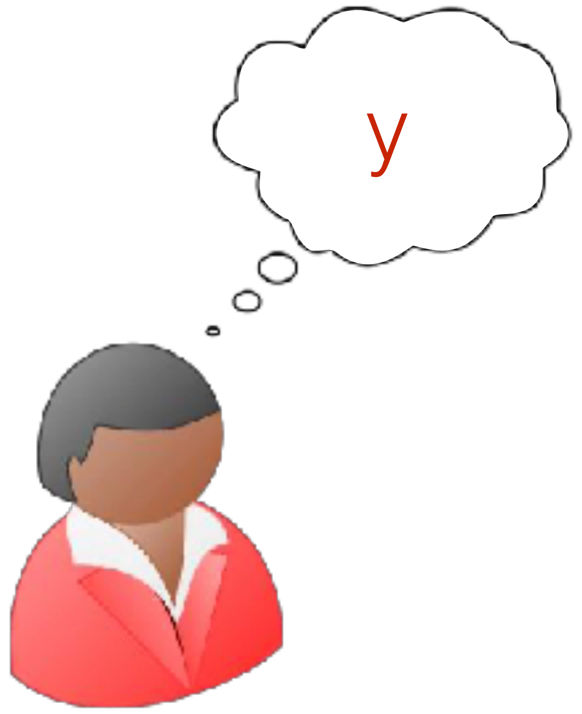
a →

← c $\quad$ ← c'

z → $\quad$ z' →

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

Schnorr protocol



$x = g^y$

$a = g^r$ →

← c

$z = r + cy$ →

Accept iff $g^z = a x^c$

HVZK

$\mathcal{HVZK}_{sim}$

$c \Leftarrow Z_q$

$z \Leftarrow Z_q$

$a = g^z / x^c$

a →

← c

z →

Sigma Protocol for Diffie-Hellman tuples

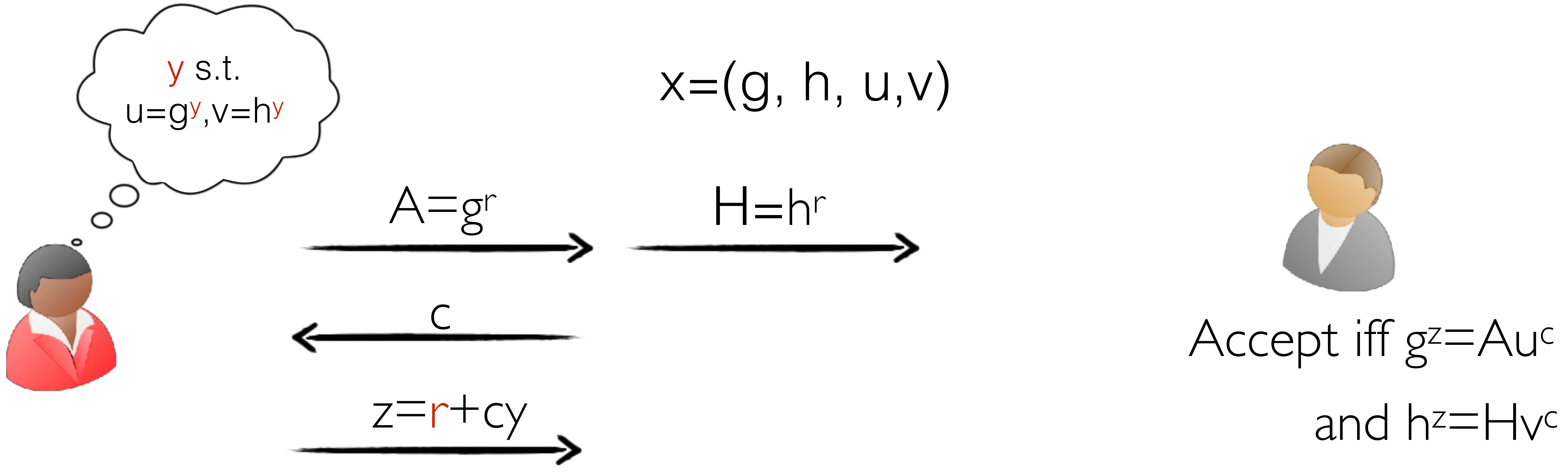
Let G be a group of order q, with generators g and h

$x=(g, h, u,v)$ Is a DH tuple if $u=g^{y}, v=h^{y}$

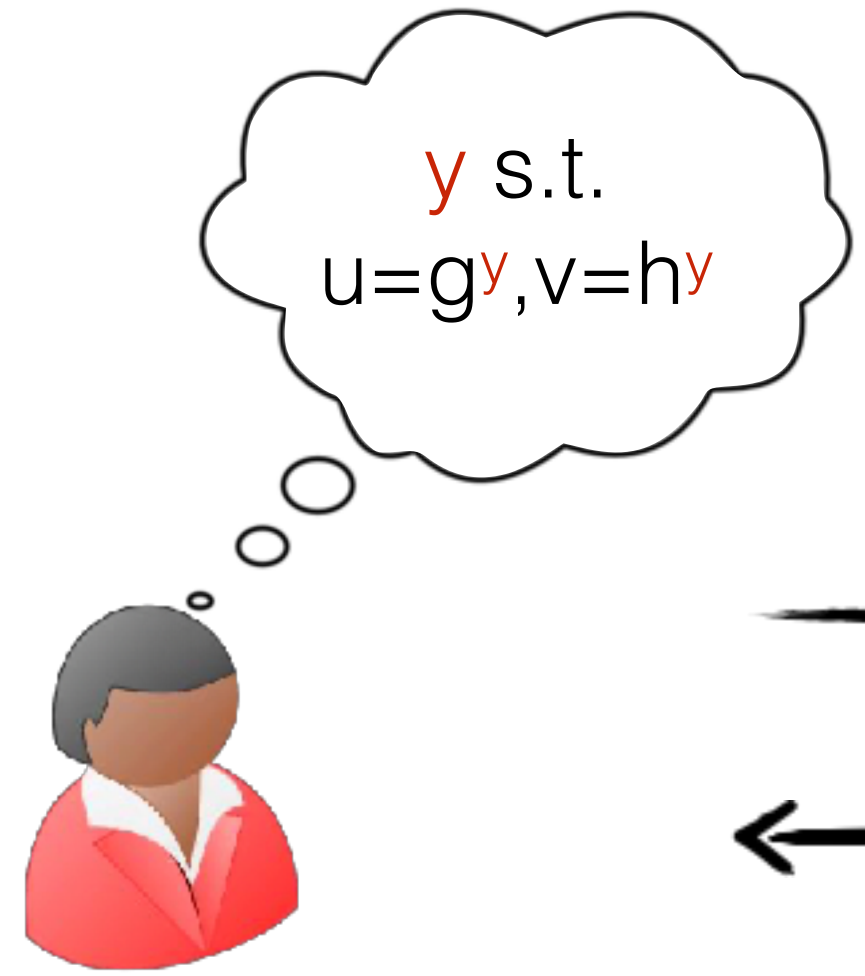
$b \leftarrow \{0,1\}$
if $b=0$ then
  $T=(g, h, u=g^{y}, v=h^{y})$
else
  $T=(g, h, u=g^{y}, v=h^{w})$ with $y \neq w$

$\xrightarrow{T}$

Sigma Protocol for Diffie-Hellman tuples



Sigma Protocol for Diffie-Hellman tuples



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

$A=g^r$ → $H=h^r$ →

← c

$z=r+cy$ →

Accept iff $g^z=Au^c$

and $h^z=Hv^c$

$\mathcal{HVZK}_{sim}$

$c \Leftarrow Z_q$

$z \Leftarrow Z_q$

$A=g^z/u^c$

$H=h^z/v^c$

$a=(A,H)$ →

← c

z →

HVZK

Sigma Protocol for Diffie-Hellman tuples

y s.t.
$u=g^y, v=h^y$

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

$A=g^r$ → $H=h^r$ →

← c

$z=r+cy$ →

Accept iff $g^z=Au^c$

and $h^z=Hv^c$

Special-soundness

Exactly the same as the one for the Dlog protocol

OR-Composition

$\Sigma_0 = (P_{\Sigma_0}, V_{\Sigma_0})$

x_0 or x_1

$\Sigma_1 = (P_{\Sigma_1}, V_{\Sigma_1})$

a_0 →

← c_0

z_0 →

$\mathcal{HVZK}^0_{sim}(x_0) \longrightarrow a_0, c_0, z_0$

$\mathcal{HVZK}^1_{sim}(x_1) \longrightarrow a_1, c_1, z_1$

a_1 →

← c_1

z_1 →

w_0

$\mathcal{HVZK}^1_{sim}(x_1) \longrightarrow a_1, c_1, z_1$

$a_0 \longleftarrow P_{\Sigma_0}(x_0, w_0)$

a_0 → a_1 →

$c_0 \longleftarrow c \oplus c_1$

$z_0 \longleftarrow P_{\Sigma_0}(x_0, w_0, c_0)$

← c

c_0, z_0 → c_1, z_1 →

$V_{\Sigma_0}(x_0, a_0, c_0, z_0) = 1$

and

$V_{\Sigma_1}(x_1, a_1, c_1, z_1) = 1$

and

$c = c_0 \oplus c_1$

OR-Composition

x_0 or x_1

Special Soundness

a_0 → a_1 →

← c ← c'

c_0, z_0 → c_1, z_1 → c'_0, z'_0 → c'_1, z'_1 →

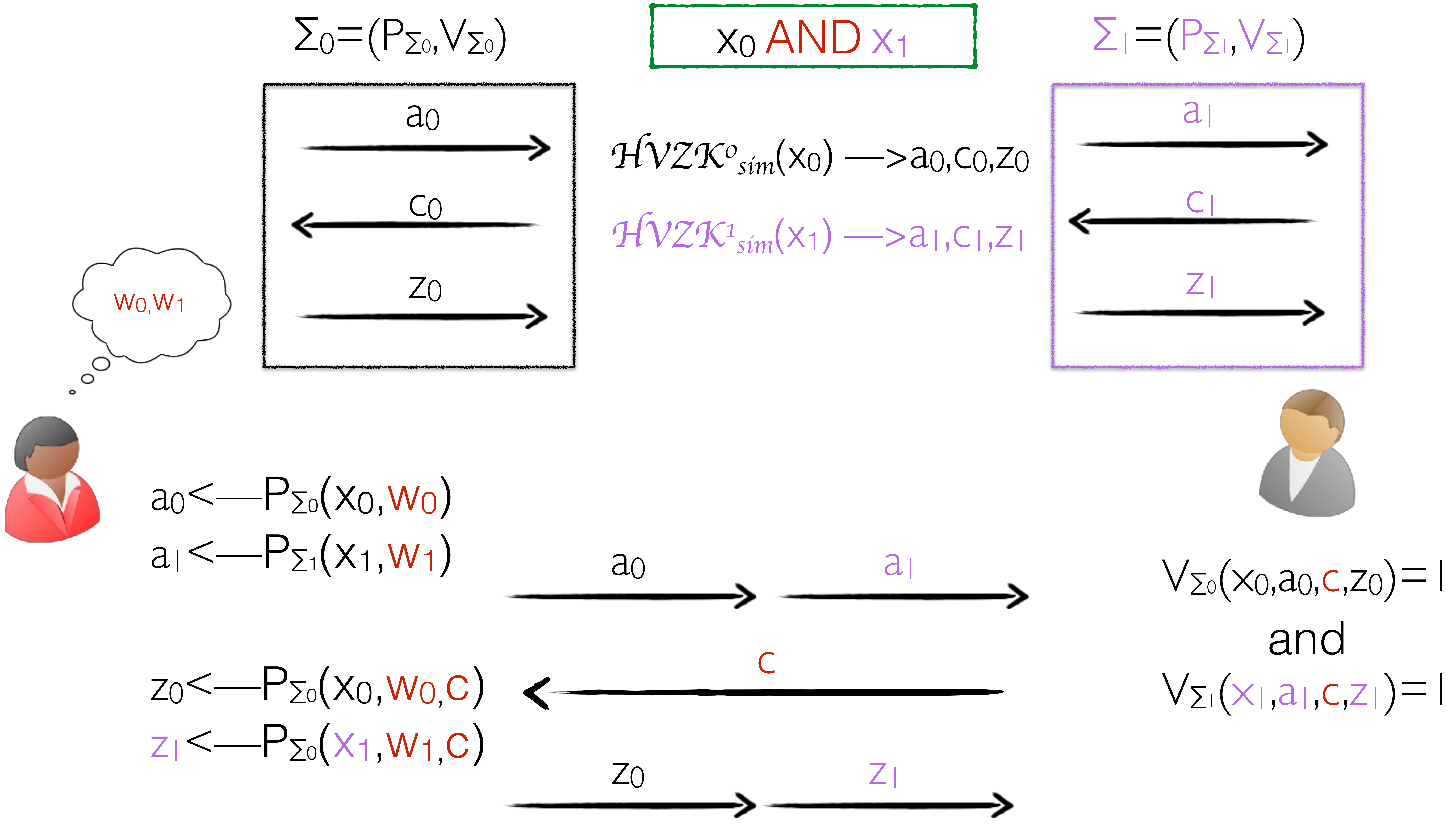
$V_{\Sigma_0}(x_0,a_0,c_0,z_0)=1$ and $V_{\Sigma_1}(x_1,a_1,c_1,z_1)=1$ and $c=c_0 \oplus c_1$

$V_{\Sigma_0}(x_0,a_0,c'_0,z'_0)=1$ and $V_{\Sigma_1}(x_1,a_1,c'_1,z'_1)=1$ and $c'=c'_0 \oplus c'_1$

$c \neq c'$ ⇒ $c_0 \neq c'_0$ or $c_1 \neq c'_1$

e.g. $c_0 \neq c'_0$ — by s-soundness of Σ_0 ⇒ w_0

AND-Composition



AND-Composition

x_0 AND x_1

Special Soundness

$a_0 \leftarrow P_{\Sigma_0}(x_0, w_0)$

$a_1 \leftarrow P_{\Sigma_1}(x_1, w_1)$

$a_0 \longrightarrow$ $a_1 \longrightarrow$

$\longleftarrow c$ $\longleftarrow c'$

$z_0 \leftarrow P_{\Sigma_0}(x_0, w_0, c)$

$z_1 \leftarrow P_{\Sigma_0}(x_1, w_1, c)$

$z_0 \longrightarrow$ $z_1 \longrightarrow$ $z'_0 \longrightarrow$ $z'_1 \longrightarrow$

$V_{\Sigma_0}(x_0, a_0, c, z_0) = 1$ $V_{\Sigma_0}(x_0, a_0, c', z'_0) = 1$

and and

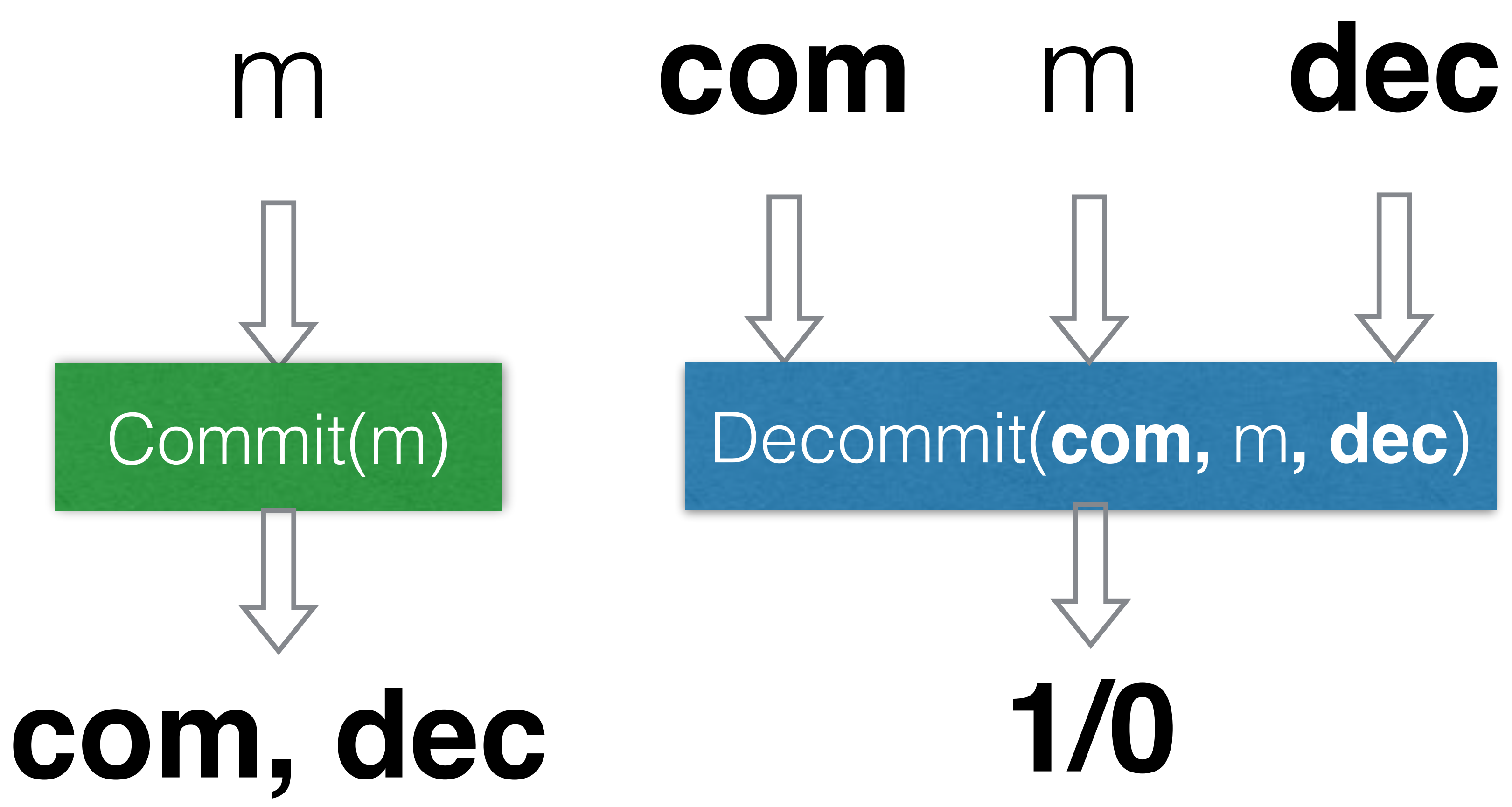
$V_{\Sigma_1}(x_1, a_1, c, z_1) = 1$ $V_{\Sigma_1}(x_1, a_1, c', z'_1) = 1$

$c \neq c'$
and
s-soundness of
Σ_0 and Σ_1

w_0, w_1

Commitments from Sigma-Protocols

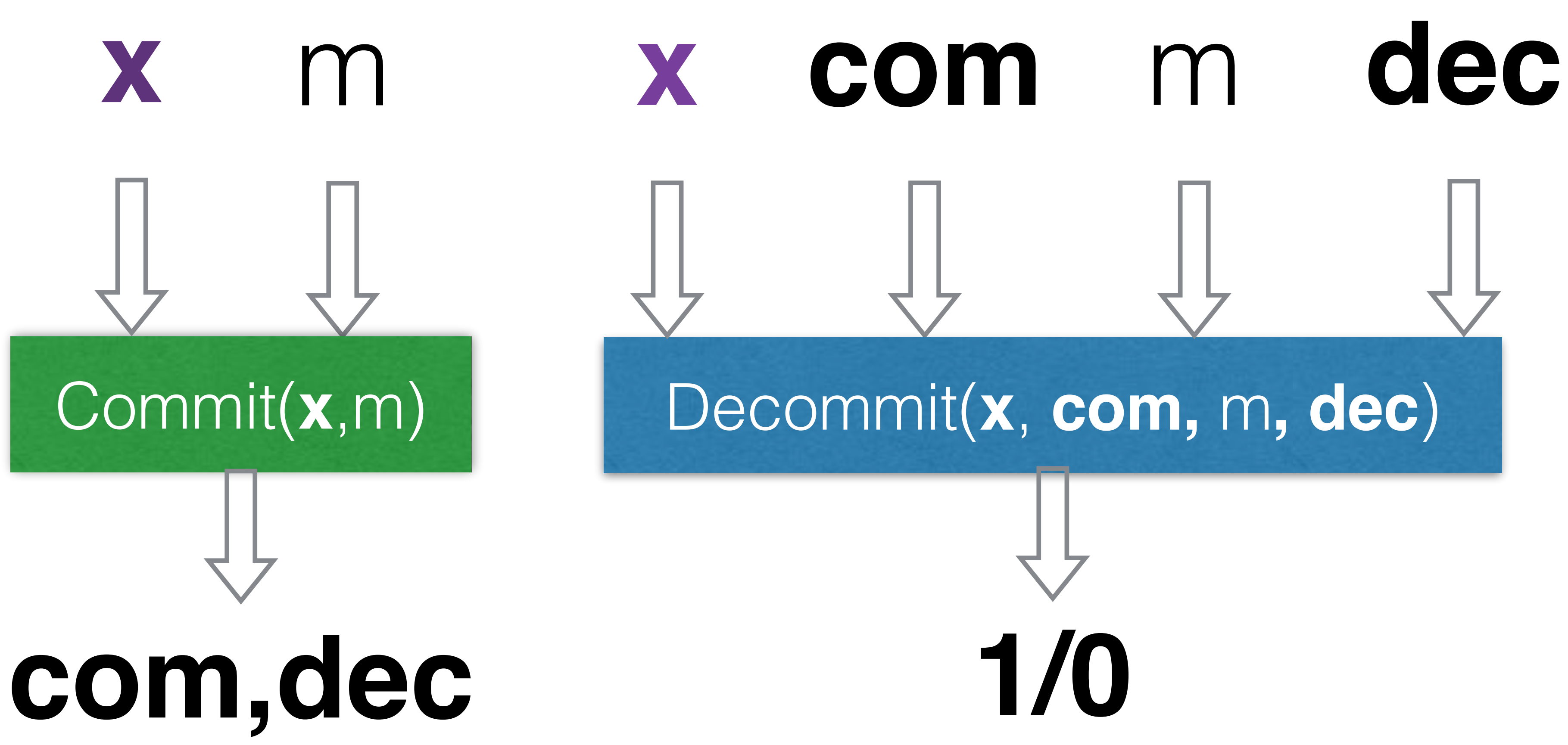
Commitment scheme



- Hiding
- Binding

  $\nexists$ **dec'**, m', with $m \neq m$ s.t.
  Decommit(**com,** m**, dec**)=1 and
  Decommit(**com,** m'**, dec'**)=1

Instance-dependent commitment scheme
NP-Language L

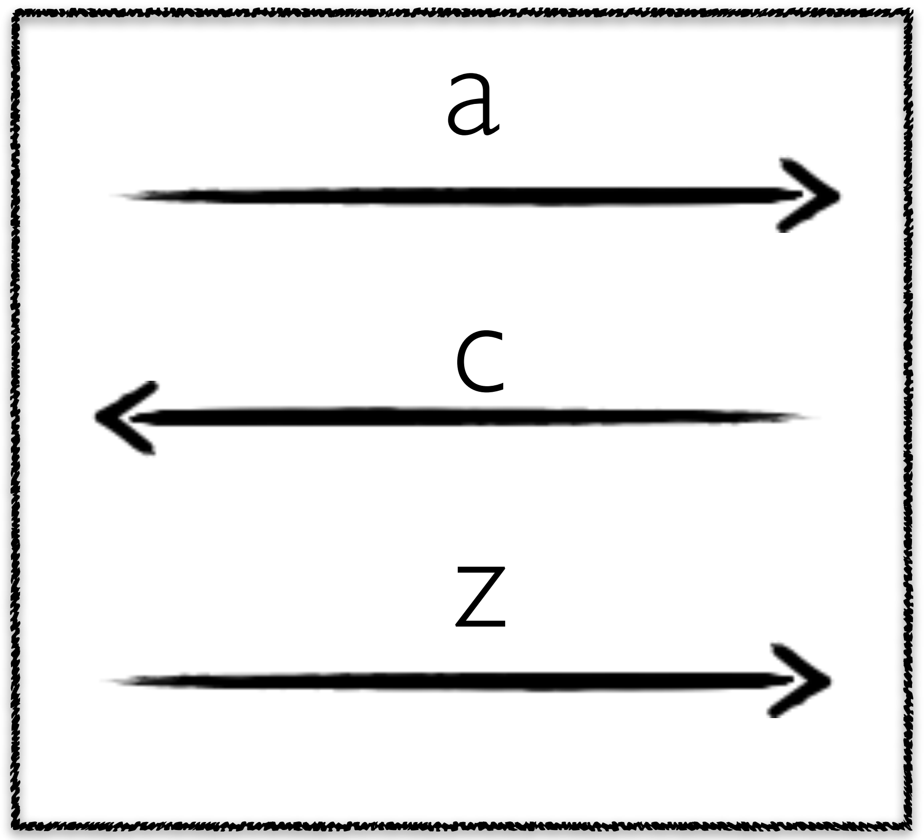


- if **x** $\in$ L Hiding
- If **x** $\notin$ L Binding

  $\nexists$ **dec'**, m', with $m \neq m$ s.t.
  Decommit(**x**, **com,** m**, dec**)=1 and
  Decommit(**x**, **com,** m'**, dec'**)=1

Commitments from Sigma-Protocols

$\Sigma = (P_\Sigma, V_\Sigma)$



$\mathcal{SHVZK}_{sim}(\mathbf{x}, c) \longrightarrow a, z$

x m

$\mathcal{SHVZK}_{sim}(\mathbf{x}, m) \longrightarrow a, z$

com <— a

dec <— z

com, dec

x **com** m **dec**

$V_\Sigma(\mathbf{x}, \mathbf{com}, m, \mathbf{dec}) \longrightarrow \mathbf{b}$

b

Binding (x ∉ L)

$V_\Sigma(\mathbf{x}, \mathbf{com}, m, \mathbf{dec}) \longrightarrow 1$

$V_\Sigma(\mathbf{x}, \mathbf{com}, m', \mathbf{dec'}) \longrightarrow 1$

$m' \neq m$

s-soundness of Σ ⇒ w: witness for **x**

Commitments from Sigma-Protocols

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

$x \in L$

x m_b

$\mathcal{SHVZK}_{sim}(x, m_b) \longrightarrow a, z$

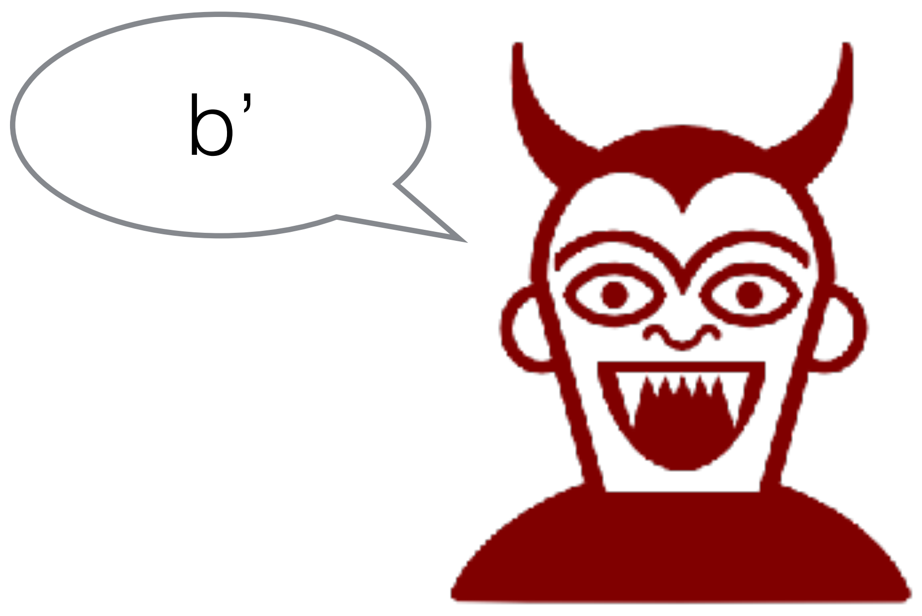
com $\leftarrow a$

dec $\leftarrow z$

com, dec

m_0, m_1 ←

com →



By contradiction b=b'

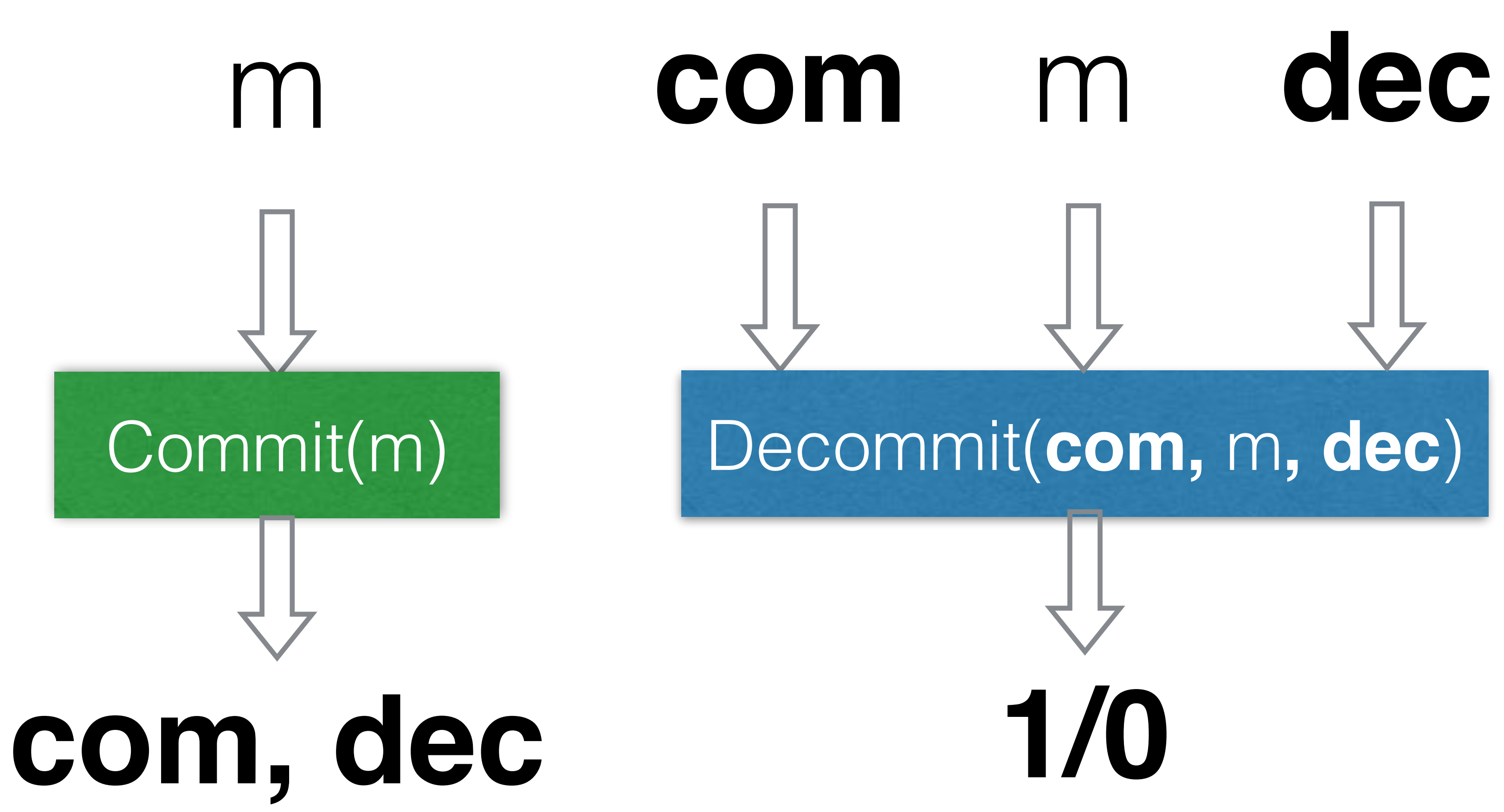
$$\mathcal{SHVZK}_{sim}(x, m_0) \longrightarrow \begin{matrix} a_0 \\ z_0 \end{matrix} \equiv \begin{matrix} a \leftarrow P_\Sigma(x, w) \\ z \leftarrow P_\Sigma(x, w, m_0) \end{matrix} \equiv \begin{matrix} a_1 \\ z_1 \end{matrix} \leftarrow \mathcal{SHVZK}_{sim}(x, m_1)$$

So far

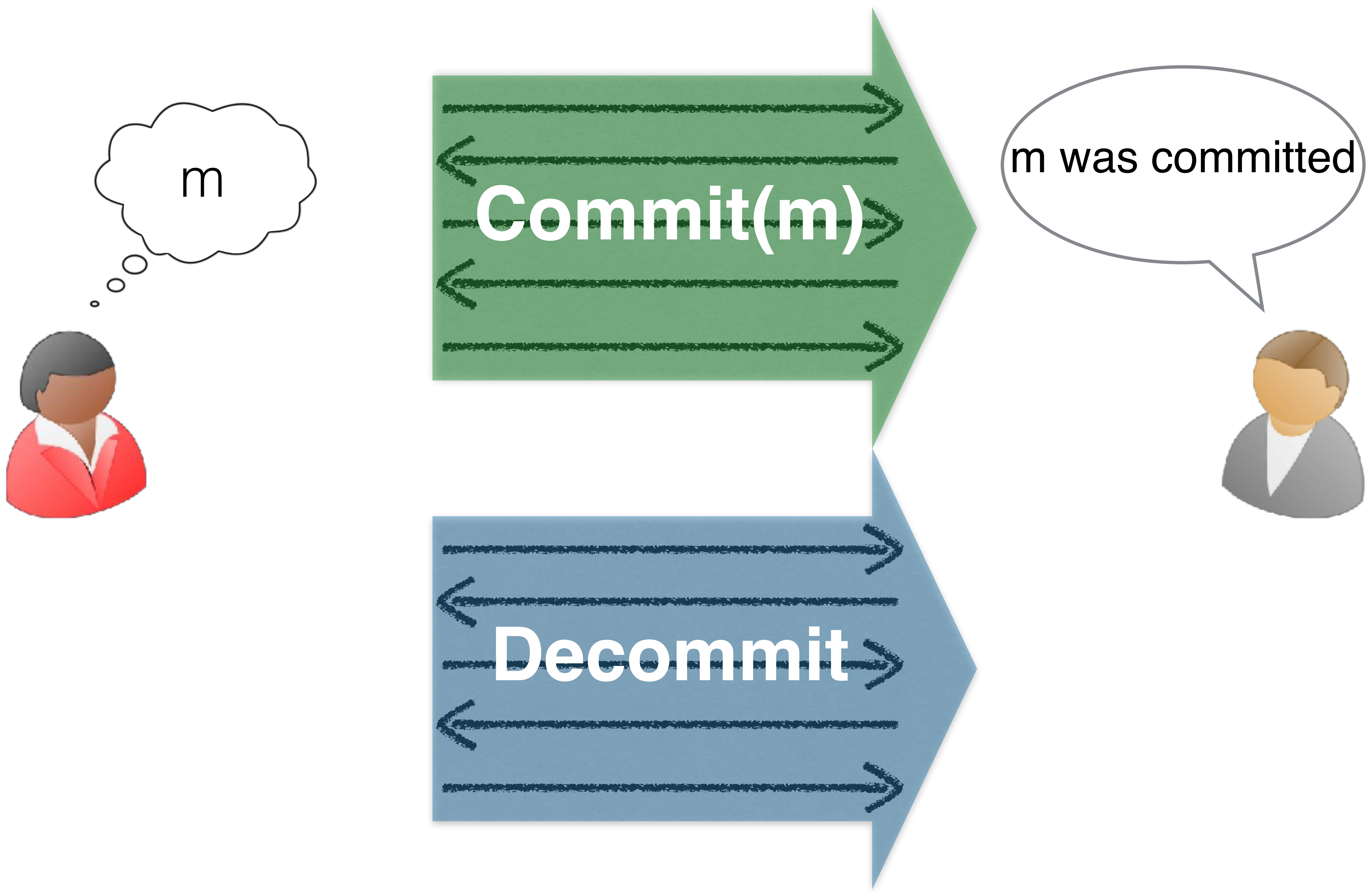
- Sigma protocols for some fixed languages
- Practical efficiency
- Only HVZK
- Can we have a sigma protocol for all NP?
- How do we get security against malicious verifiers?

Commitments

Non-interactive



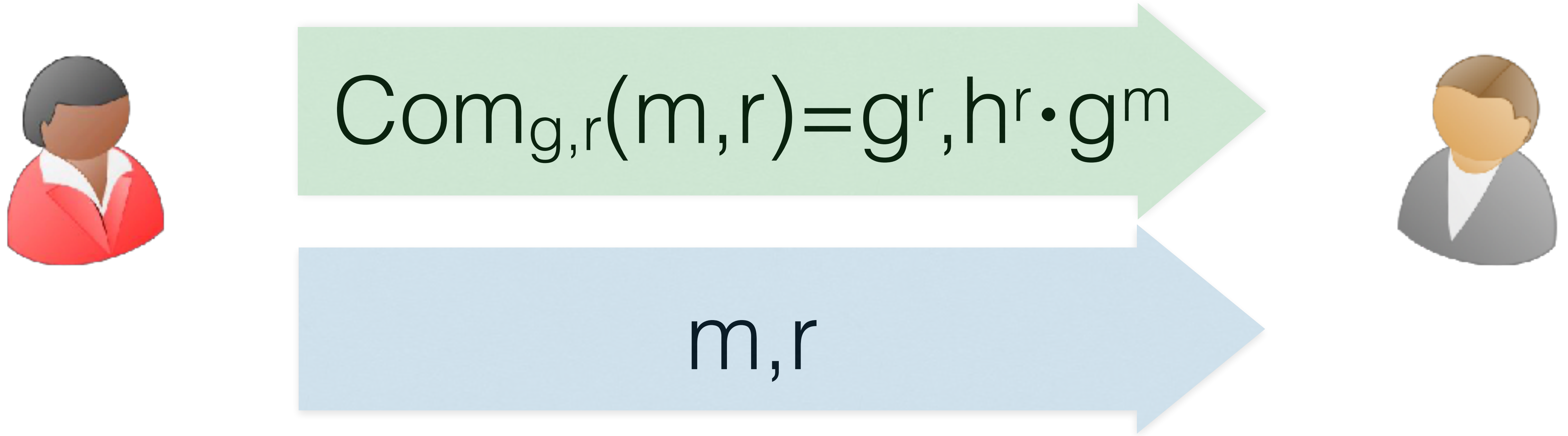
Interactive



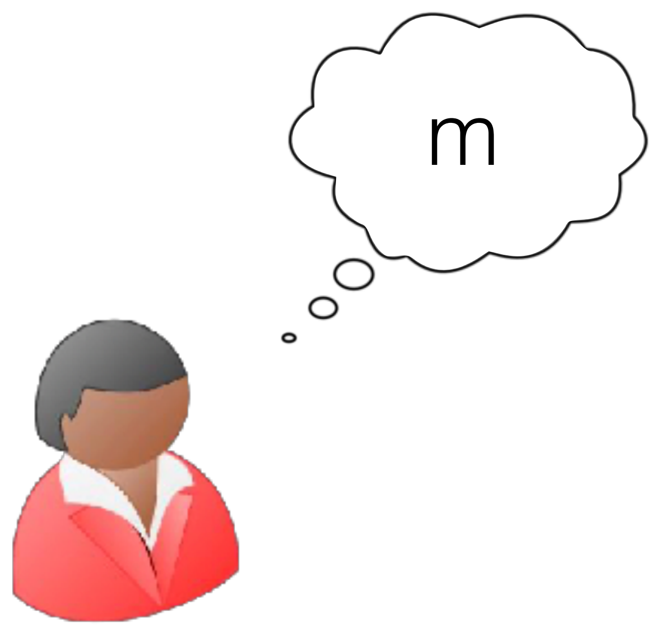
- (computational statistical) Hiding
- (computational statistical) Binding

Statistically binding commitments

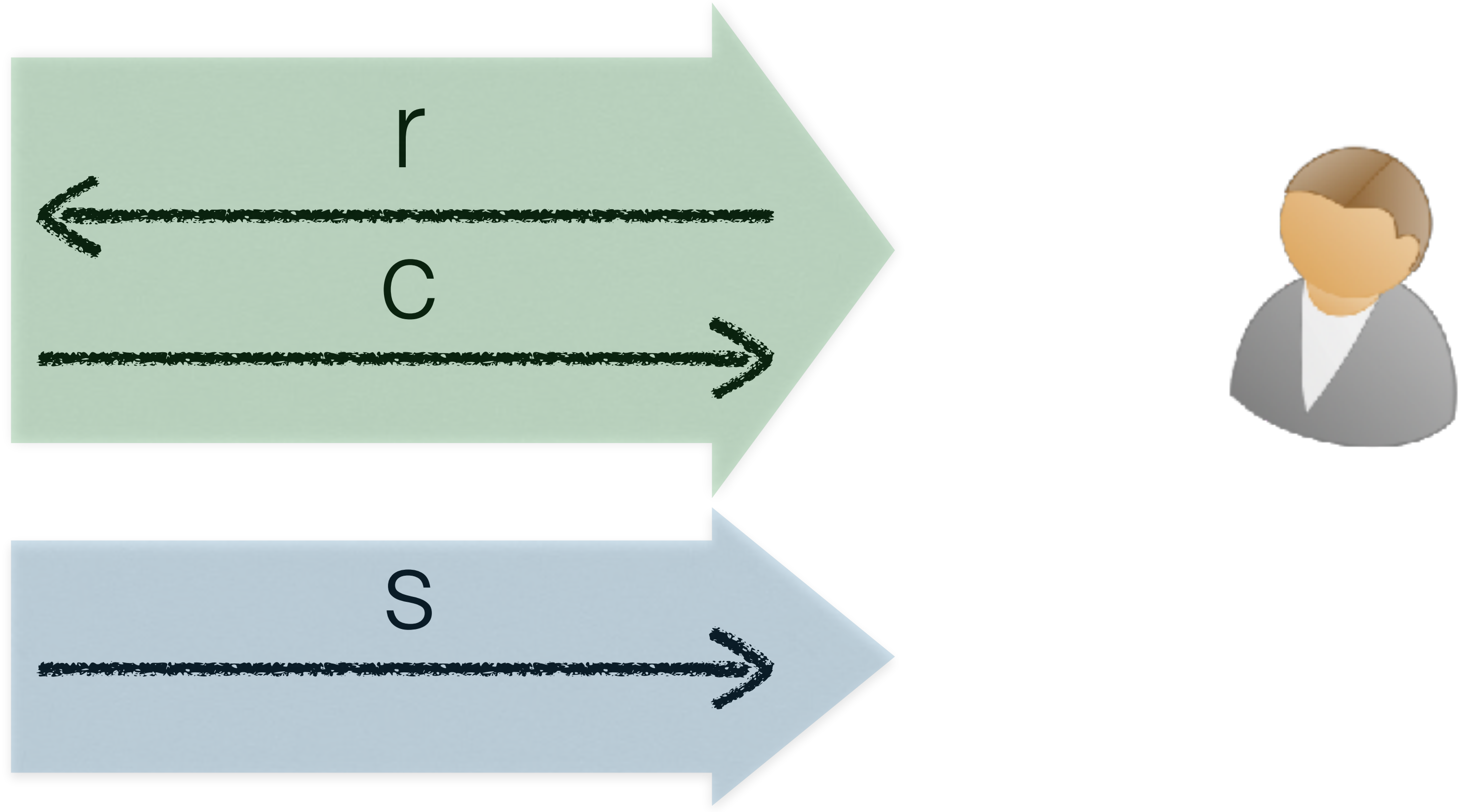
El-Gamal



From PRGs (OWFs)



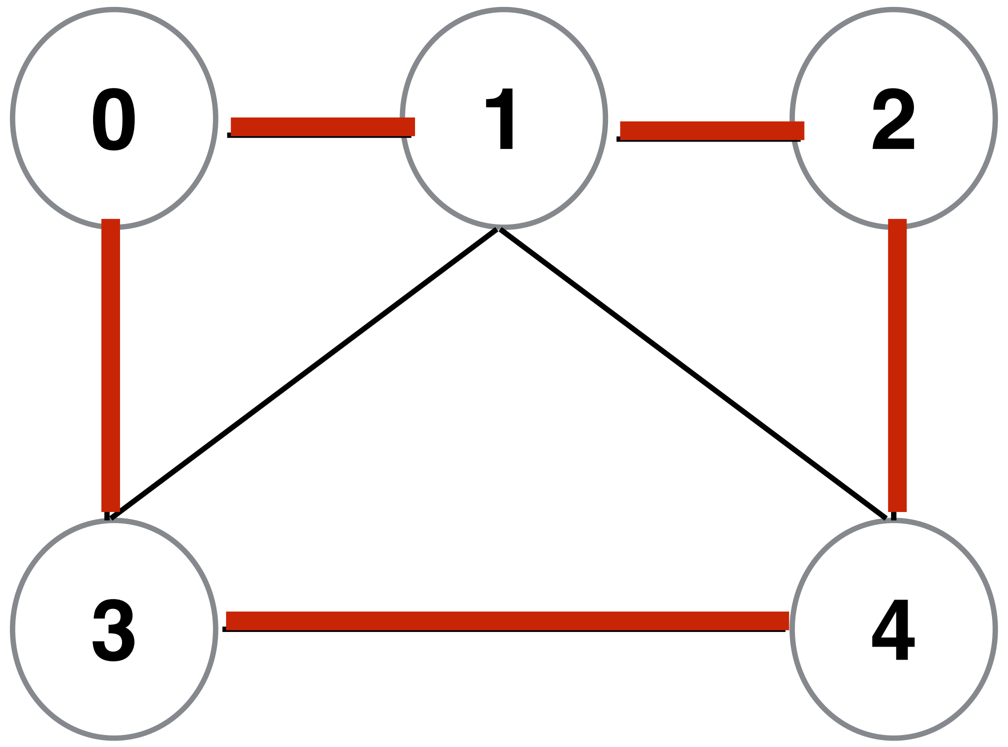
$$c=\begin{cases} G(s) & m=0 \\ G(s)\oplus r & m=1 \end{cases}$$



if $G(s)=c$ then 0
if $G(s)\oplus r=c$ then 1

Hamiltonian graphs

G



(0,1,2,4,3,0)

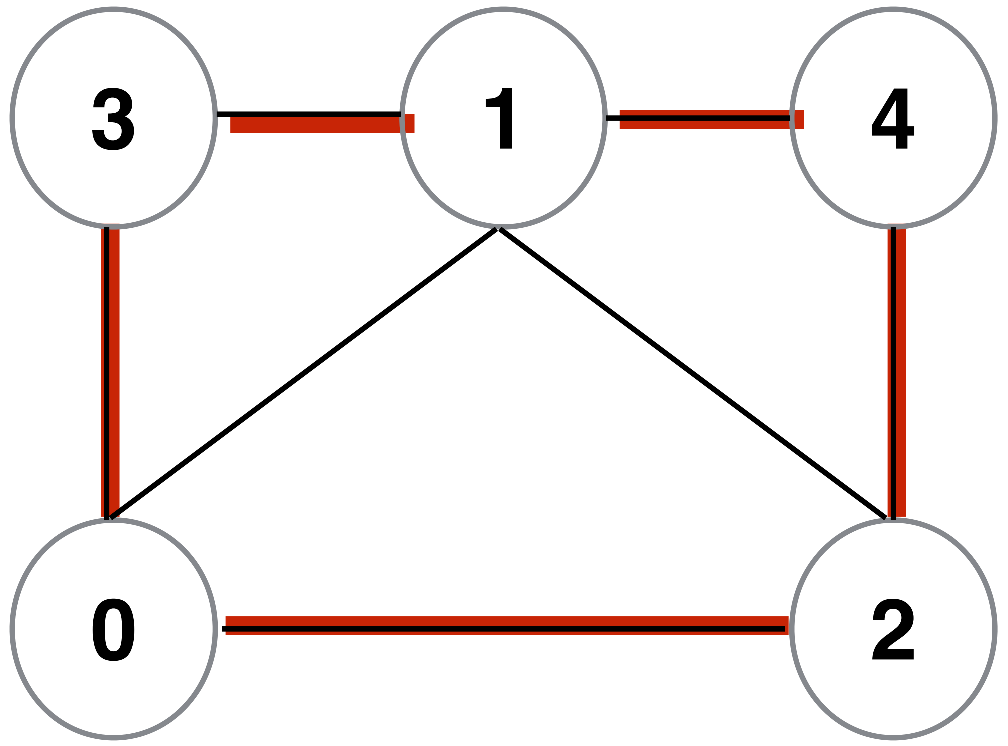
	0	**1**	**2**	**3**	**4**
0	1	1	0	1	0
1	1	1	1	1	1
2	0	1	1	0	1
3	1	1	0	1	1
4	0	0	1	1	1

NP-Complete

Every L ∈ NP is poly-time reducible to HAM

If we have a protocol with property **p** for the language *HAM* then we have a protocol with the property **p** for every language L ∈ NP

H



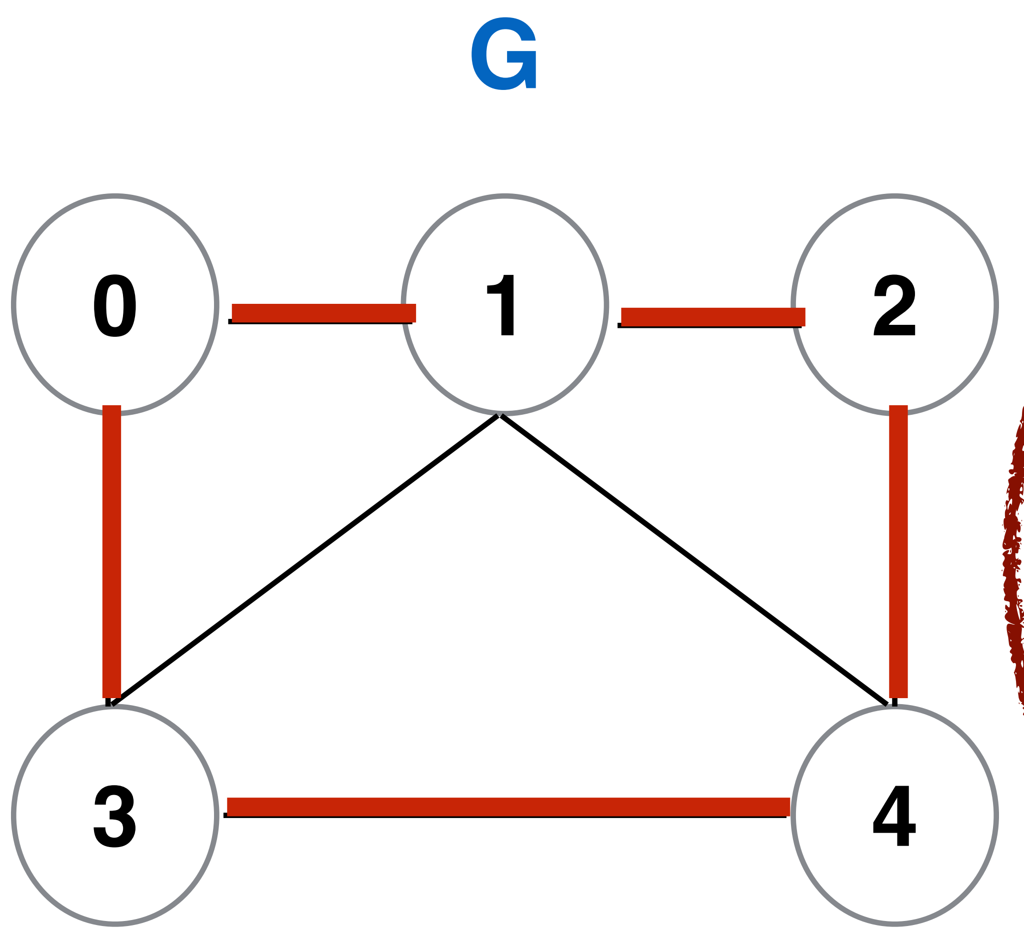
(0,3,1,4,2,0)

	0	**1**	**2**	**3**	**4**
0	1	1	1	1	0
1	1	1	1	1	1
2	1	1	1	0	1
3	1	1	0	0	0
4	0	1	1	0	1

π

G	**H**
0	3
1	1
2	4
3	0
4	2

Sigma Protocol for HAM



(0,1,2,4,3,0)

π

G	H
0	3
1	1
2	4
3	0
4	2

G

	0	1	2	3	4
0	1	1	0	1	0
1	1	1	1	1	1
2	0	1	1	0	1
3	1	1	0	1	1
4	0	0	1	1	1

Stm: **G** is Hamiltonian

π

H

	0	1	2	3	4
0	1	1	1	1	0
1	1	1	1	1	1
2	1	1	1	0	1
3	1	1	0	0	0
4	0	1	1	0	1

H

	0	1	2	3	4
0	Com	Com	Com	Com	Com
1	Com	Com	Com	Com	Com
2	Com	Com	Com	Com	Com
3	Com	Com	Com	Com	Com
4	Com	Com	Com	Com	Com

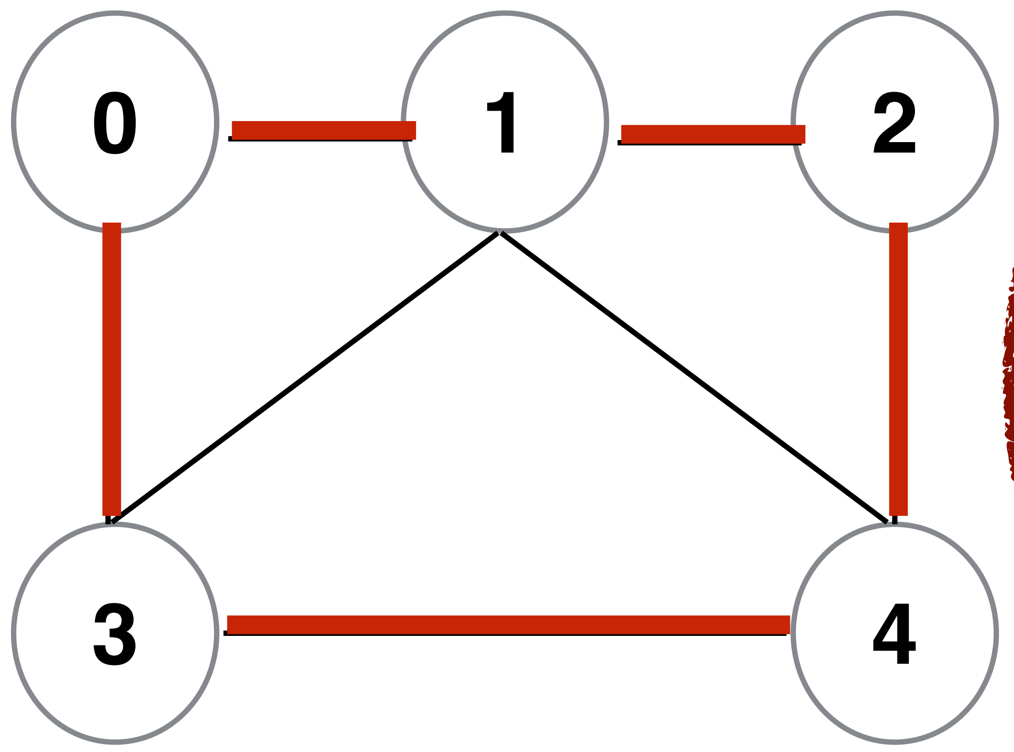
0

	0	1	2	3	4
0	1	1	1	1	0
1	1	1	1	1	1
2	1	1	1	0	1
3	1	1	0	0	0
4	0	1	1	0	1

G	H
0	3
1	1
2	4
3	0
4	2

Sigma Protocol for HAM

G



(0,1,2,4,3,0)

π

G	H
0	3
1	1
2	4
3	0
4	2

	0	1	2	3	4
0	1	1	0	1	0
1	1	1	1	1	1
2	0	1	1	0	1
3	1	1	0	1	1
4	0	0	1	1	1

Stm: **G** is Hamiltonian

H

	0	1	2	3	4
0	Com	Com	Com	Com	Com
1	Com	Com	Com	Com	Com
2	Com	Com	Com	Com	Com
3	Com	Com	Com	Com	Com
4	Com	Com	Com	Com	Com

1

	0	1	2	3	4
0	Com	Com	Com	1	Com
1	Com	Com	Com	Com	1
2	1	Com	Com	Com	Com
3	Com	1	Com	Com	Com
4	Com	Com	1	Com	Com

Special Soundness

Stm: **G** is Hamiltonian

	0	1	2	3	4
0	1	1	0	1	0
1	1	1	1	1	1
2	0	1	1	0	1
3	1	1	0	1	1
4	0	0	1	1	1

(0,1,2,4,3,0)

H

	0	1	2	3	4
0	Com	Com	Com	Com	Com
1	Com	Com	Com	Com	Com
2	Com	Com	Com	Com	Com
3	Com	Com	Com	Com	Com
4	Com	Com	Com	Com	Com

Cycle in **H**

(0,3,1,4,2,0)

0

1

	0	1	2	3	4
0	1	1	1	1	0
1	1	1	1	1	1
2	1	1	1	0	1
3	1	1	0	0	0
4	0	1	1	0	1

G	H
0	3
1	1
2	4
3	0
4	2

	0	1	2	3	4
0	Com	Com	Com	1	Com
1	Com	Com	Com	Com	1
2	1	Com	Com	Com	Com
3	Com	1	Com	Com	Com
4	Com	Com	1	Com	Com

Cycle in **G**

(3,0,1,2,4,3)

It relies on the **binding** of the commitment

Special Honest Verifier Zero-Knowledge (b=1)

Stm: **G** is Hamiltonian

	0	1	2	3	4
0	1	1	0	1	0
1	1	1	1	1	1
2	0	1	1	0	1
3	1	1	0	1	1
4	0	0	1	1	1

	0	1	2	3	4
0	Com	Com	Com	Com	Com
1	Com	Com	Com	Com	Com
2	Com	Com	Com	Com	Com
3	Com	Com	Com	Com	Com
4	Com	Com	Com	Com	Com

1

	0	1	2	3	4
0	Com	Com	Com	1	Com
1	Com	Com	Com	Com	1
2	1	Com	Com	Com	Com
3	Com	1	Com	Com	Com
4	Com	Com	1	Com	Com

It relies on the **hiding** of the commitment

Special Honest Verifier Zero-Knowledge (b=0)

Stm: **G** is Hamiltonian

	0	1	2	3	4
0	1	1	0	1	0
1	1	1	1	1	1
2	0	1	1	0	1
3	1	1	0	1	1
4	0	0	1	1	1

H

	0	1	2	3	4
0	Com	Com	Com	Com	Com
1	Com	Com	Com	Com	Com
2	Com	Com	Com	Com	Com
3	Com	Com	Com	Com	Com
4	Com	Com	Com	Com	Com

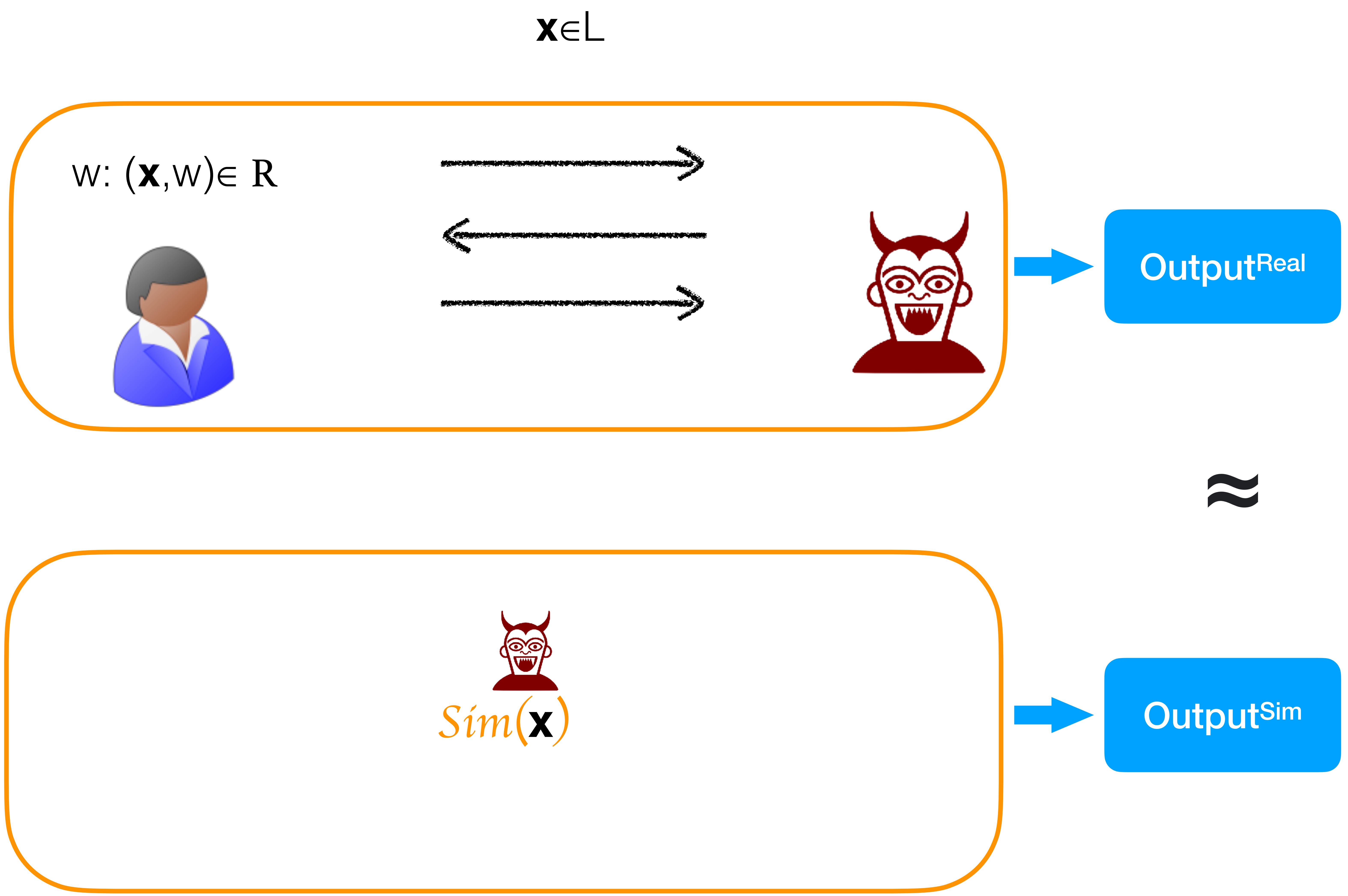
0

G	**H**
0	3
1	1
2	4
3	0
4	2

	0	1	2	3	4
0	1	1	1	1	0
1	1	1	1	1	1
2	1	1	1	0	1
3	1	1	0	0	0
4	0	1	1	0	1

G	**H**
0	3
1	1
2	4
3	0
4	2

Zero-Knowledge against arbitrary verifiers



Completeness

Soundness

Zero-knowledge

Zero-Knowledge against arbitrary verifiers

Sim(**x**)

- Sample a random bit b
- SHVZK(x,b)->a,c,z
- If b=b*
- If b≠b*

a →

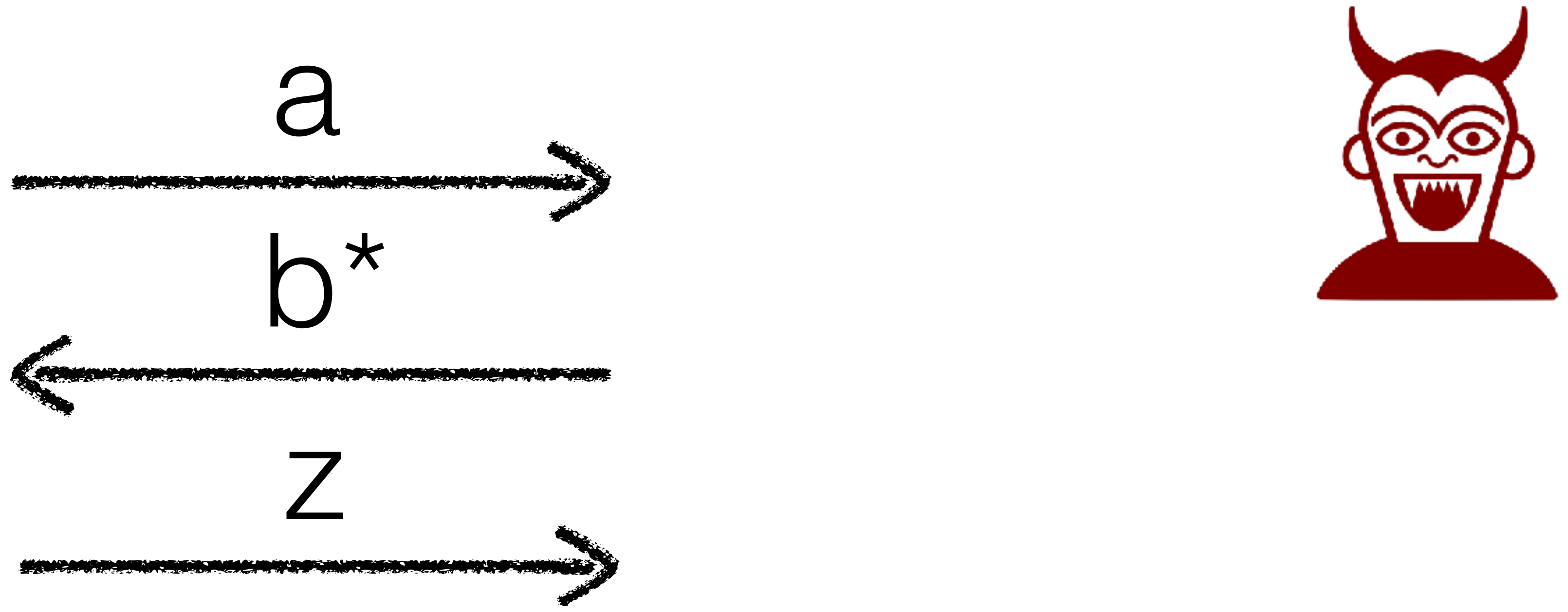
← b*

z →

Zero-Knowledge against arbitrary verifiers

Sim(**x**)

- Sample a random bit b
- SHVZK(x,b)->a,c,z
- If b=b*
- If b≠b*



The simulator succeeds in 2 expected number of rewinds

If we use the Sigma protocol for HAM, we have a 3-round ZK protocol for all NP [Blum86]

- Computational ZK if the commitments are statistically binding (one additional round is needed if we want to rely on OWFs)
- Statistical ZK if the commitments are statistically hiding

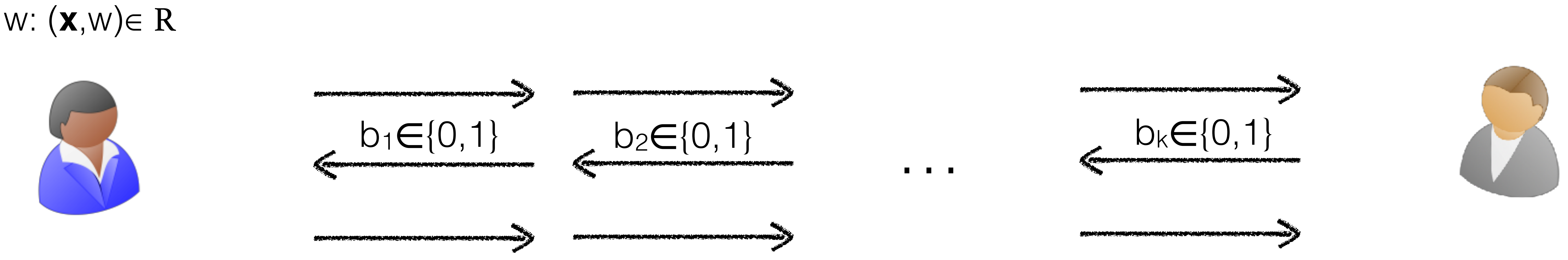
Are we happy with this protocol?

A malicious prover can cheat with 1/2 probability

Our Goal

- Computational zero-knowledge
- Constant round (1 round maybe)
- Negligible soundness error
- Minimal assumptions

Reduce the soundness error of the sigma-protocol



- Repeat the protocol in parallel k times in parallel
- A corrupted prover cannot guess the challenge in advance

How do we simulate?

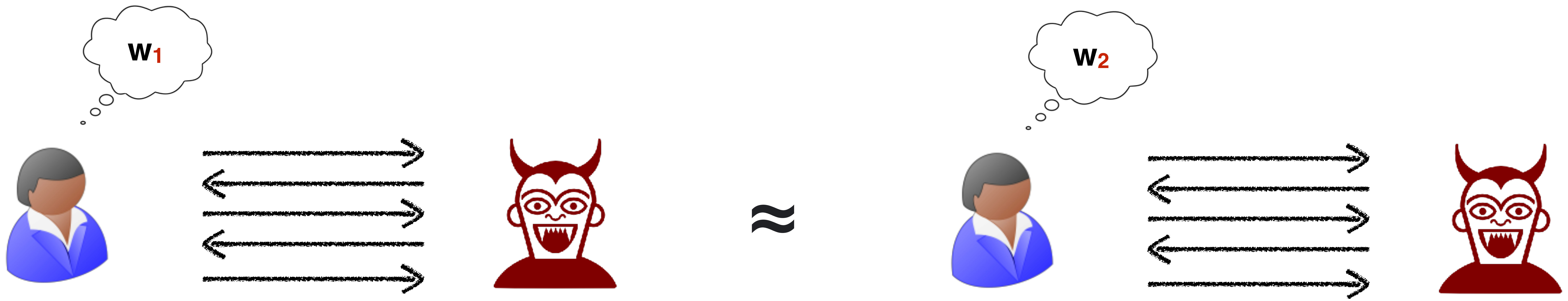
- In general, we cannot have a ZK 3-round protocol unless the polynomial hierarchy collapses*
- We can achieve a weaker notion of ZK, which we will use as a tool for our final, optimal round protocol

Witness Indistinguishability

Witness Indistinguishability

The interaction between the prover and the verifier does not reveal which of the NP witnesses for $x \in L$ was used in the proof

For every w_1, w_2 such that $(x, w_1) \in Rel$ and $(x, w_2) \in Rel$



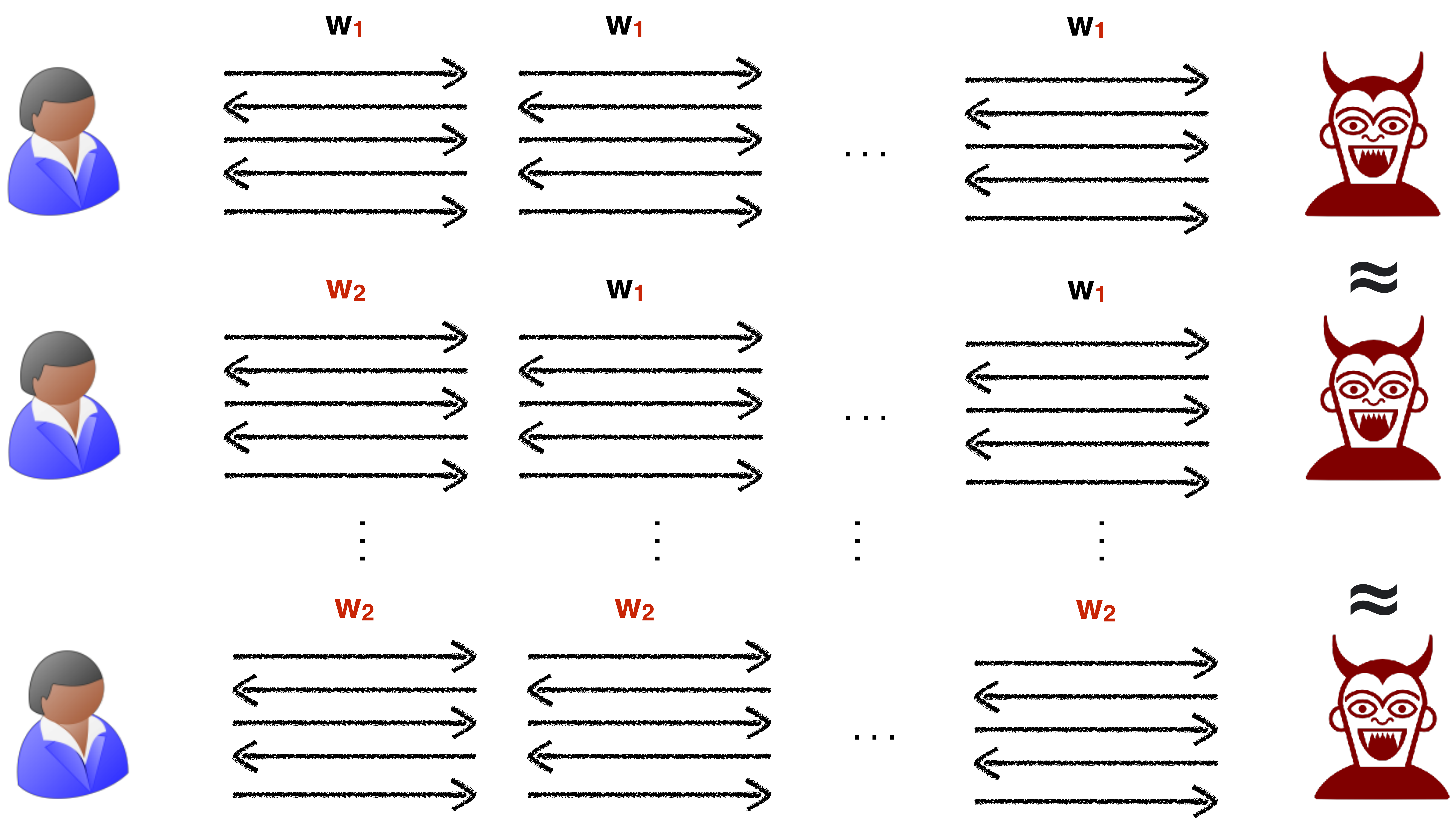
- $L \in NP$ can have many different relations. The relation specifies what I am hiding
- Trivial if there is only one witness
- In the security game, the witnesses are public
- Every ZK proof/argument is also WI
- WI is closed under parallel/concurrent composition

Every ZK proof/argument is also WI

For every w_1, w_2 such that $(\mathbf{x}, w_1) \in Rel$ and $(\mathbf{x}, w_2) \in Rel$

WI is closed under parallel composition

For every w_1, w_2 such that $(\mathbf{x}, w_1) \in Rel$ and $(\mathbf{x}, w_2) \in Rel$



Observations and Corollaries

Every zero-knowledge protocol is WI

+

A sigma-protocol with 1-bit challenge is zero-knowledge

+

HAM is a sigma-protocol with 1-bit challenge based on the existence of statistically binding non-interactive commitment scheme

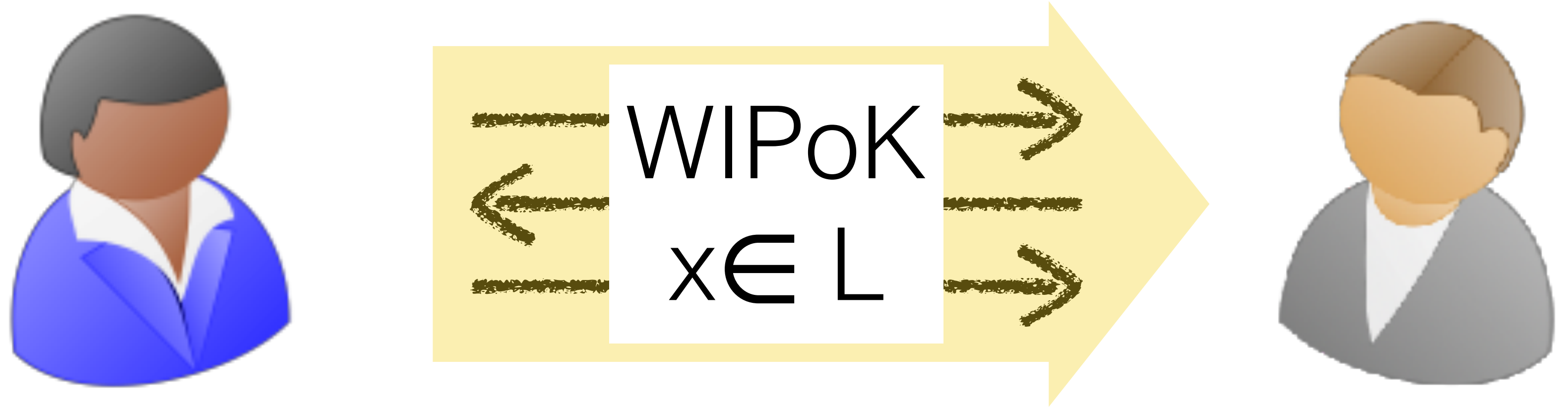
+

Amplify the soundness of the WI via parallel repetition

+

Sigma-Protocols are PoK

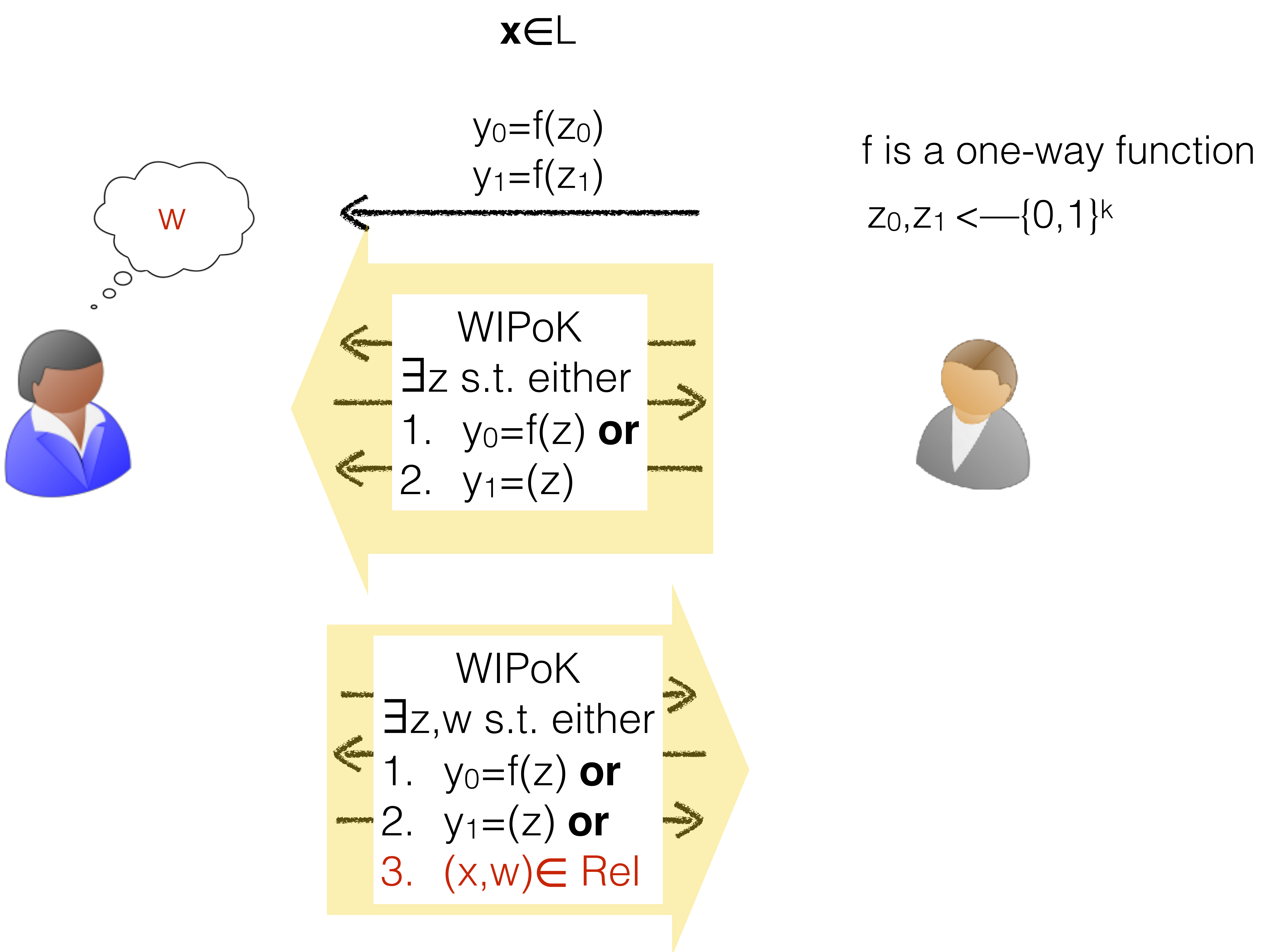
=



Theorem

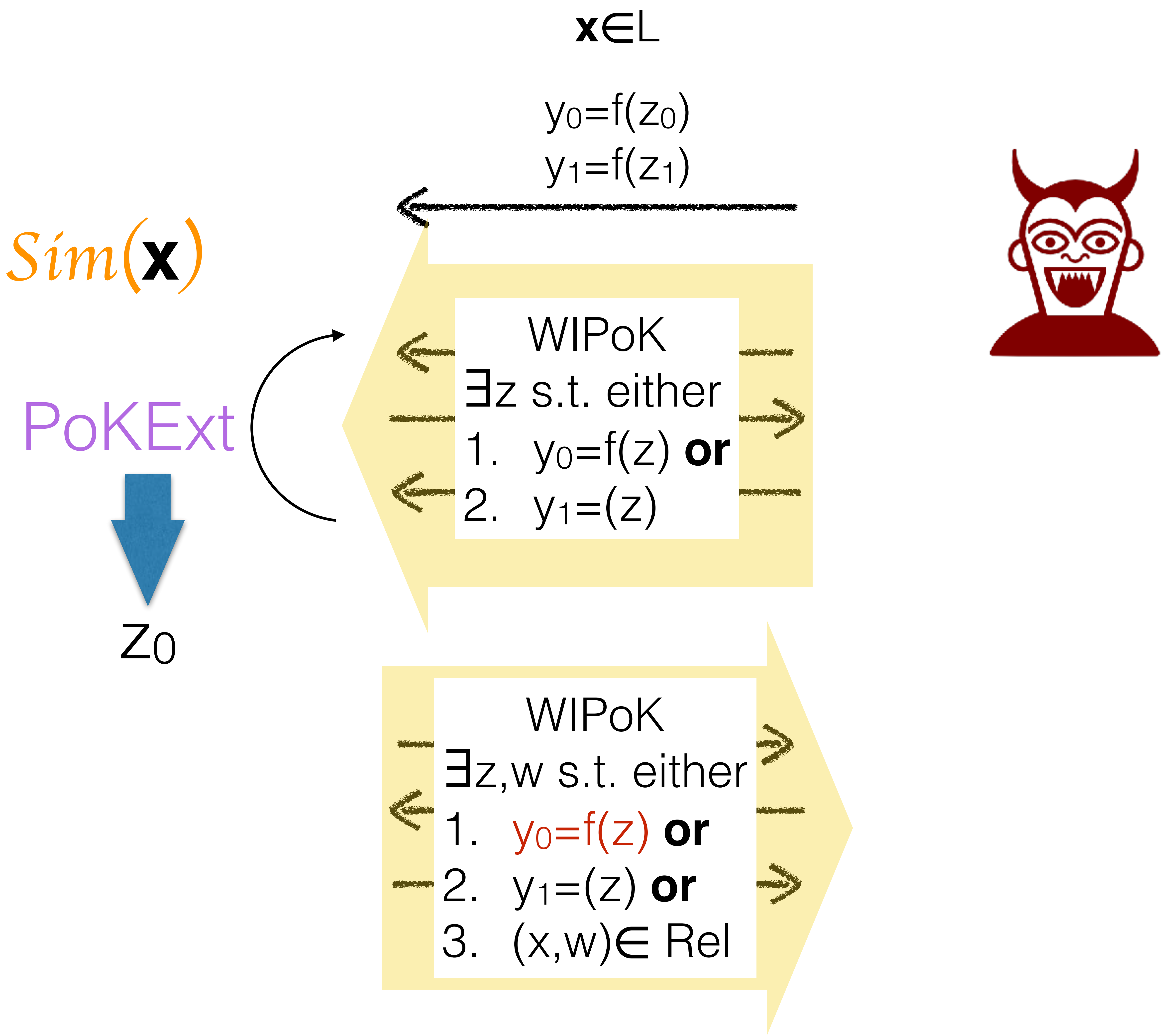
Assuming non-interactive statistically binding commitments every $L \in NP$ has a 3-round witness-indistinguishable proof-of-knowledge (WIPoK) with negligible soundness error

Constant round zero-knowledge argument for NP [FS90,FLS90]



Constant round zero-knowledge argument for NP [FS90,FLS90]

Zero-Knowledge



the PoK property guarantees that the extraction is successful in *expected polynomial time*

WI guarantees that the adversary does not distinguishes between the real proof and the simulated proof

Constant round zero-knowledge argument for NP [FS90,FLS90]

Soundness

x∉L

$y_0=f(z_0)$
$y_1=f(z_1)$

f is a one-way function

z_0,z_1 <—{0,1}^{k}

WIPoK
∃z s.t. either
1. $y_0=f(z)$ **or**
2. $y_1=f(z)$

Do the WIPoK using z_0

WIPoK
∃z,w s.t. either
1. $y_0=f(z)$ **or**
2. $y_1=f(z)$ **or**
3. ~~(x,w)∈Rel~~

Could we extract z_1?

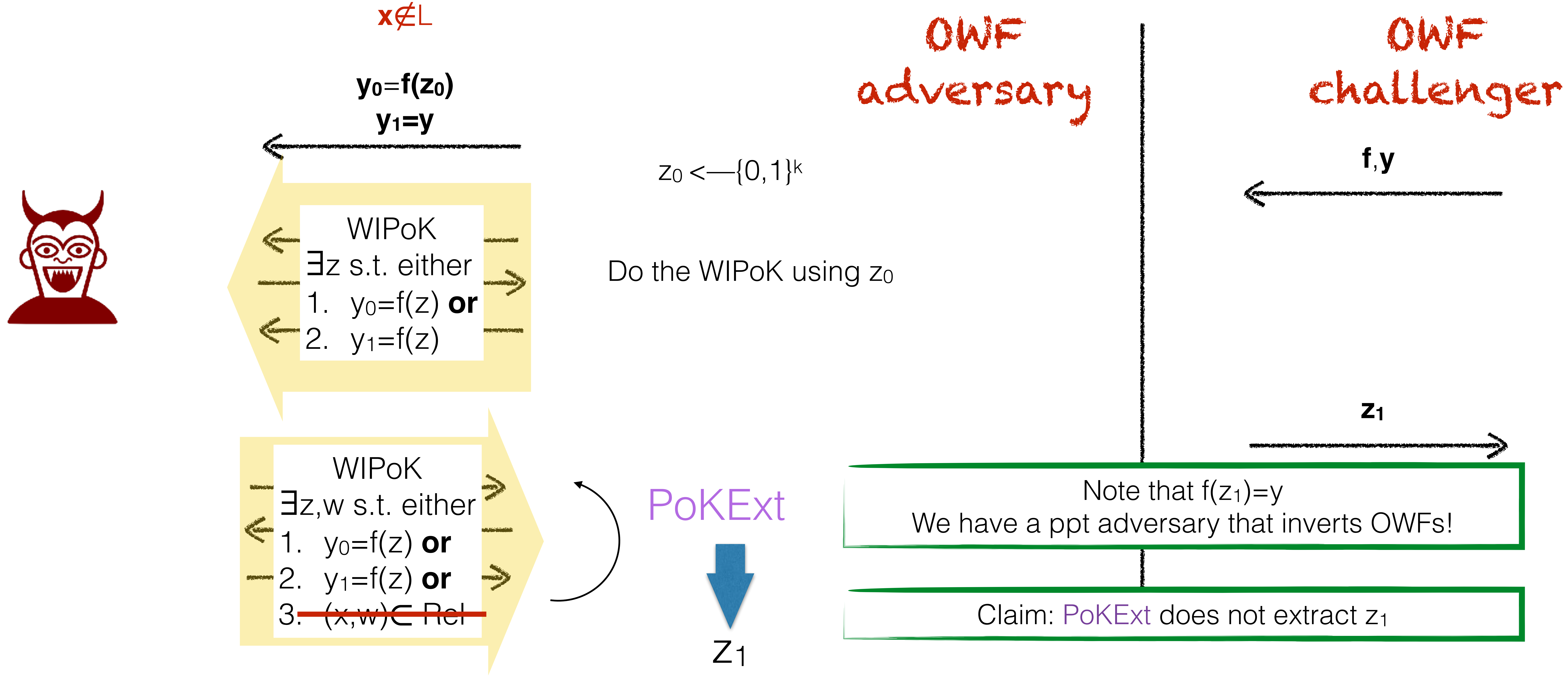
PoKExt

z_1

Assume this happens, then we have an efficient algorithm to compute the pre-image of y_1

Constant round zero-knowledge argument for NP [FS90,FLS90]

Soundness



Constant round zero-knowledge argument for NP [FS90,FLS90]

Soundness

Claim: If we use z_b to complete the first WIPoK then PoKExt does not extract z_{1-b}

$x \notin L$

$y_0=f(z_0)$
$y_1=f(z_1)$

f is a one-way function

$z_0,z_1 \leftarrow \{0,1\}^k$

WIPoK
∃z s.t. either
1. $y_0=f(z)$ **or**
2. $y_1=f(z)$

Do the WIPoK using z_1

WIPoK
∃z,w s.t. either
1. $y_0=f(z)$ **or**
2. $y_1=f(z)$
3. ~~$(x,w) \in Rel$~~

PoKExt

z_0

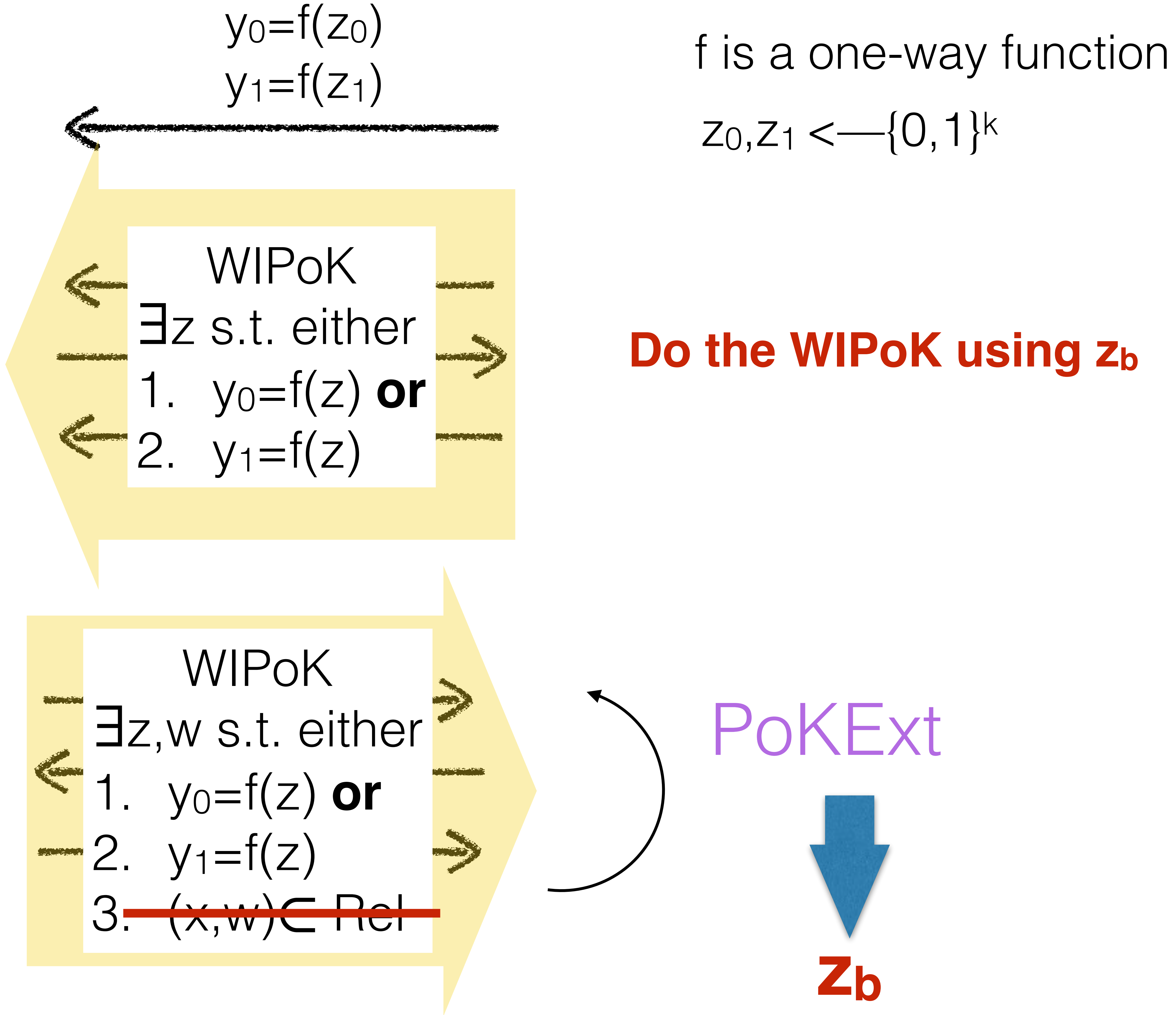
Can the extracted value be z_0?

No, for the same arguments as before

Constant round zero-knowledge argument for NP [FS90,FLS90]

Soundness

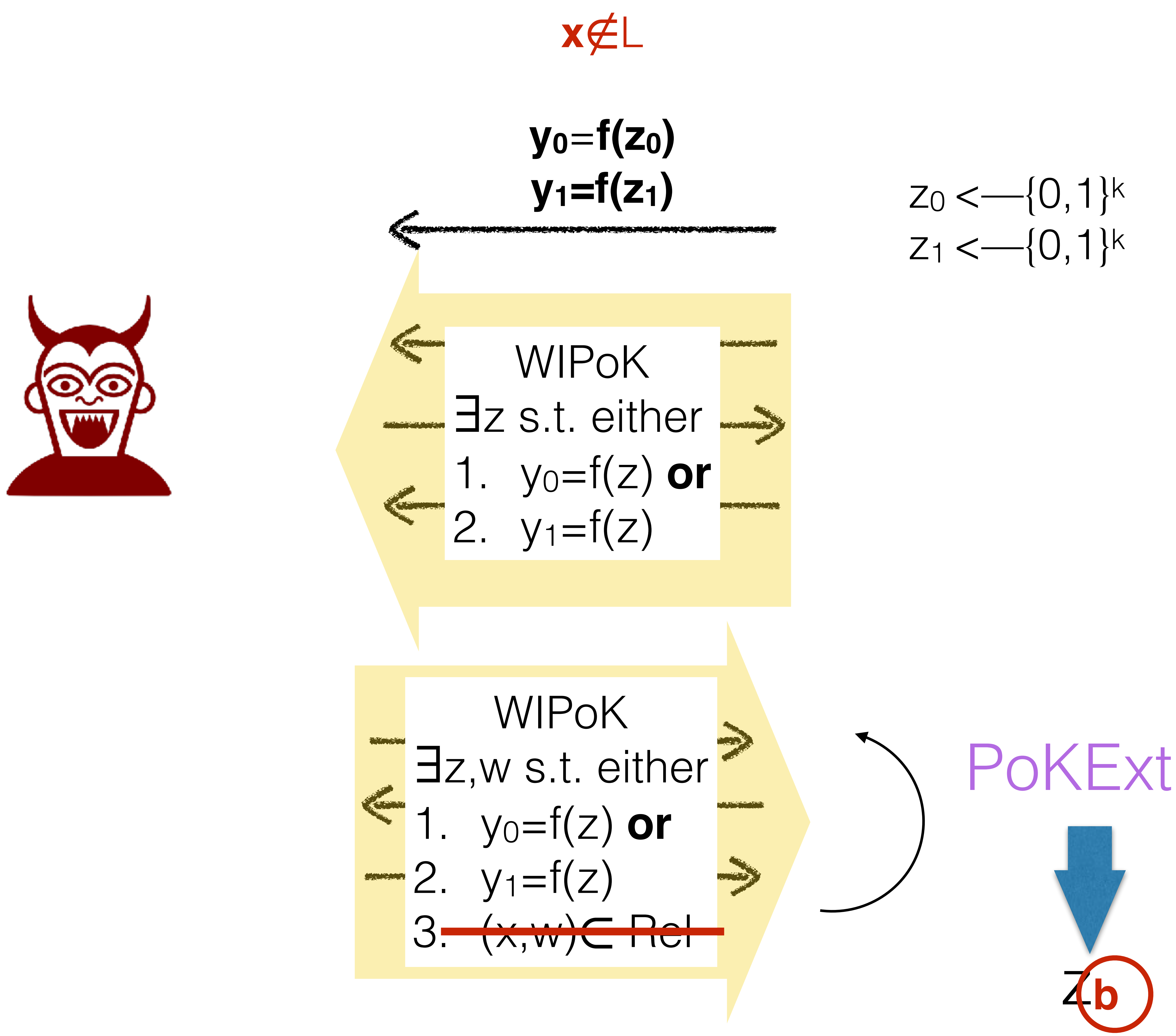
Claim: If we use z_b to complete the first WIPoK then PoKExt does not extract z_{1-b}

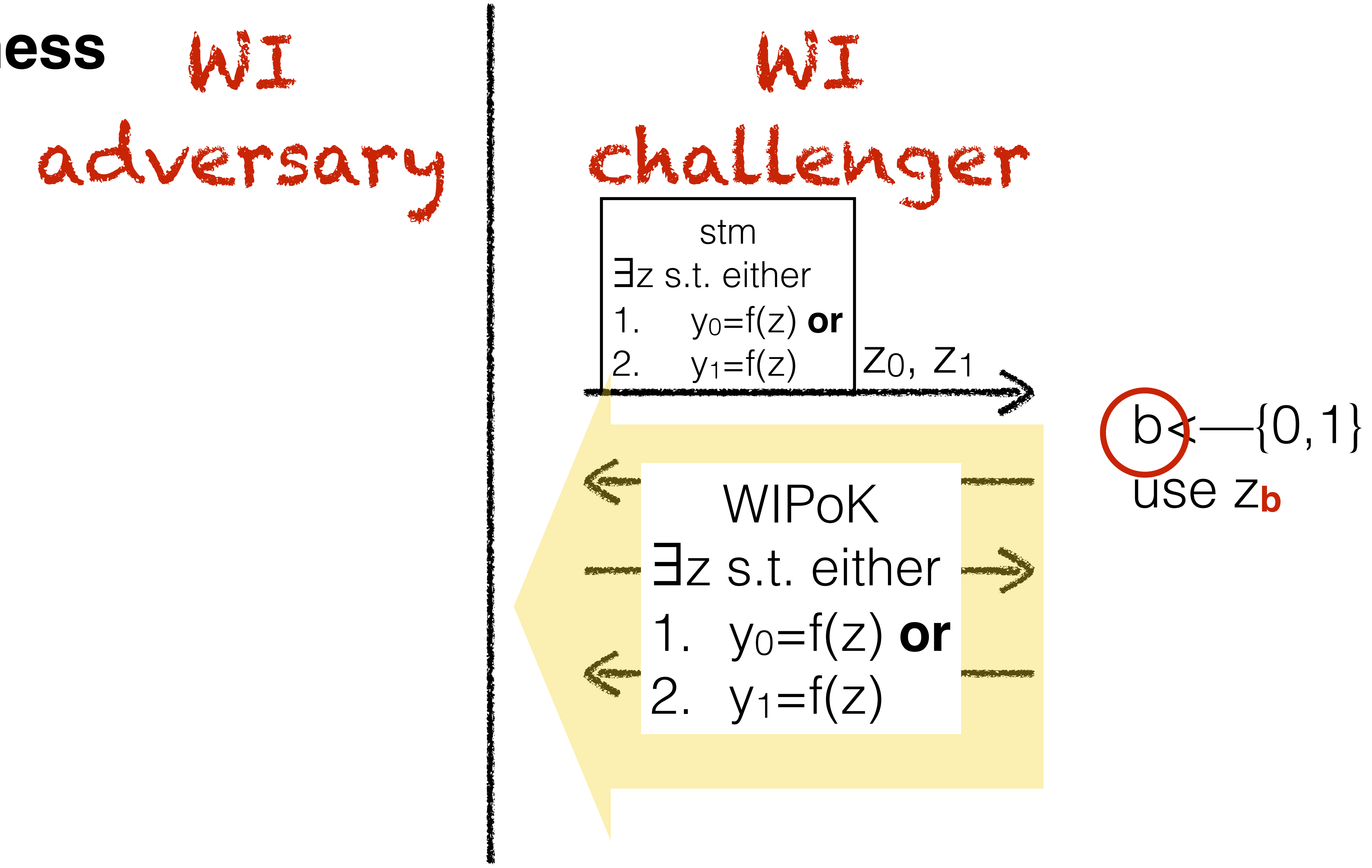


If this happens, we have a reduction to the WI property of the first WIPoK

Constant round zero-knowledge argument for NP [FS90,FLS90]

Soundness



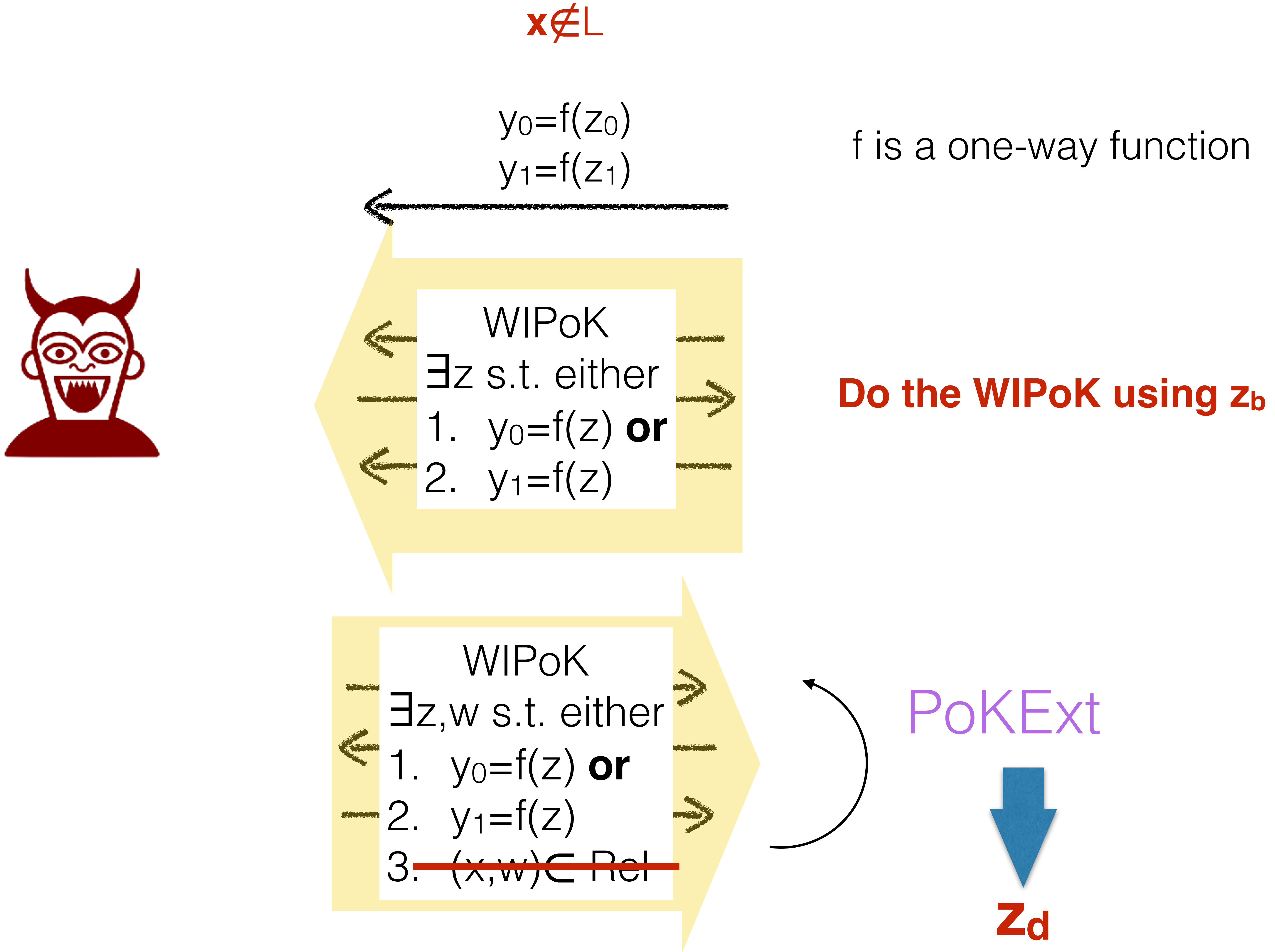


We have an adversary that guesses correctly what witness has been used to compute the first WIPoK

Constant round zero-knowledge argument for NP [FS90,FLS90]

Soundness

f is a one-way function



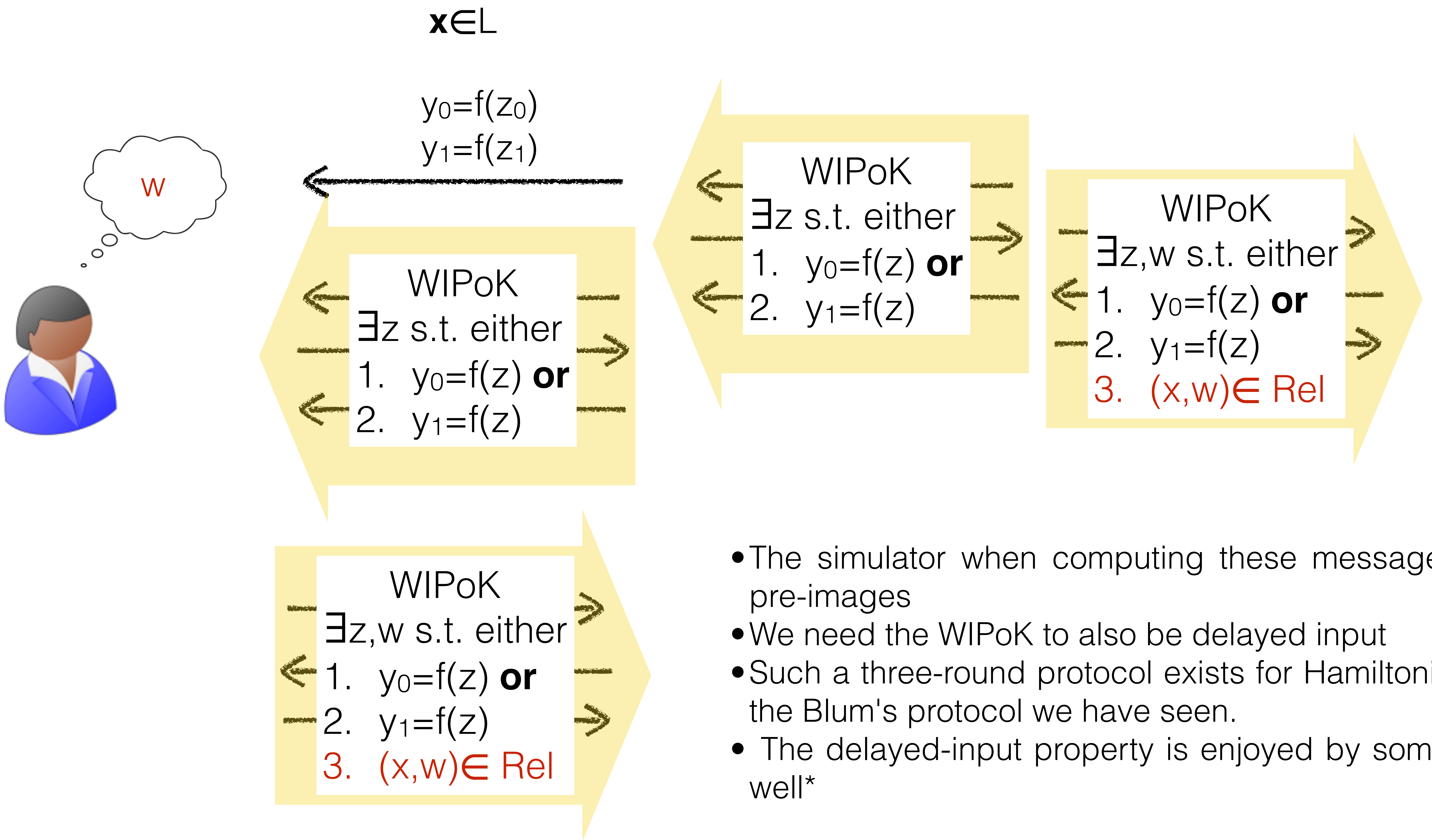
Claim: If we use z_b to complete the first WIPoK then PoKExt does not extract z_{1-b}

Claim: If we use z_b to complete the first WIPoK then PoKExt does not extract z_b

Hence it must be that we extract the witness for x —>**x**∈L

Let's squeeze it into four rounds

Soundness

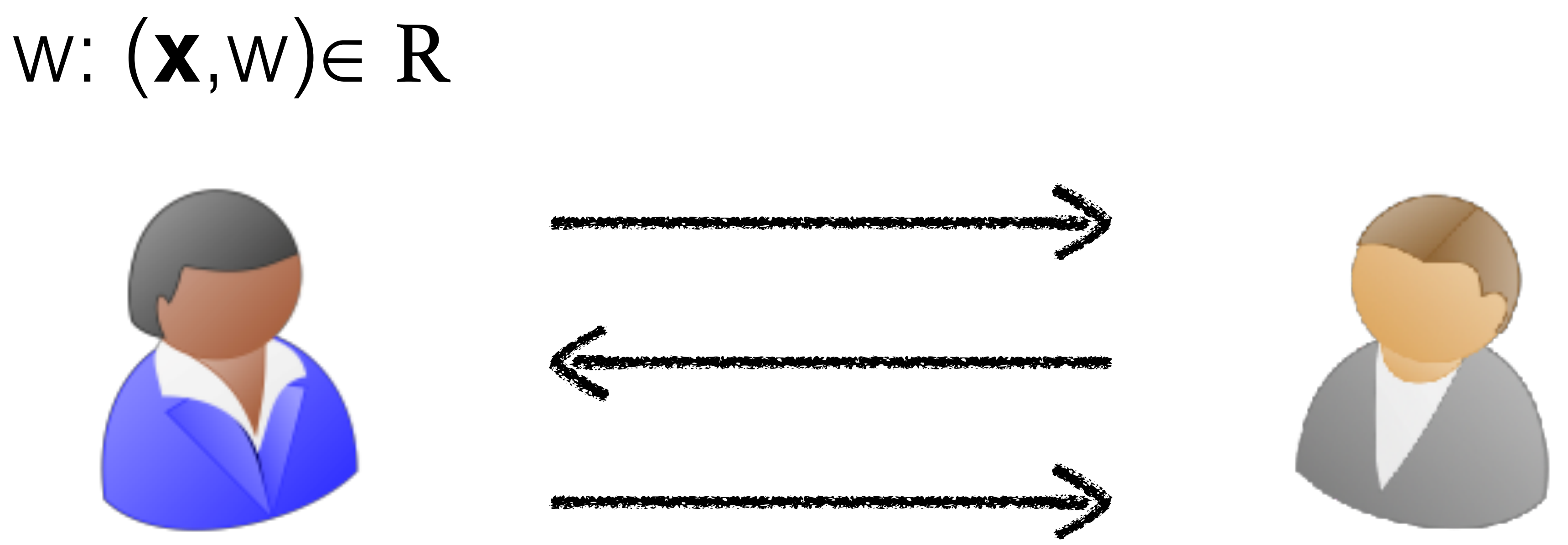


- The simulator when computing these messages does not know any of the pre-images
- We need the WIPoK to also be delayed input
- Such a three-round protocol exists for Hamiltonicity [**LS90**] and it is similar to the Blum's protocol we have seen.
- The delayed-input property is enjoyed by some efficient sigma-protocols as well*

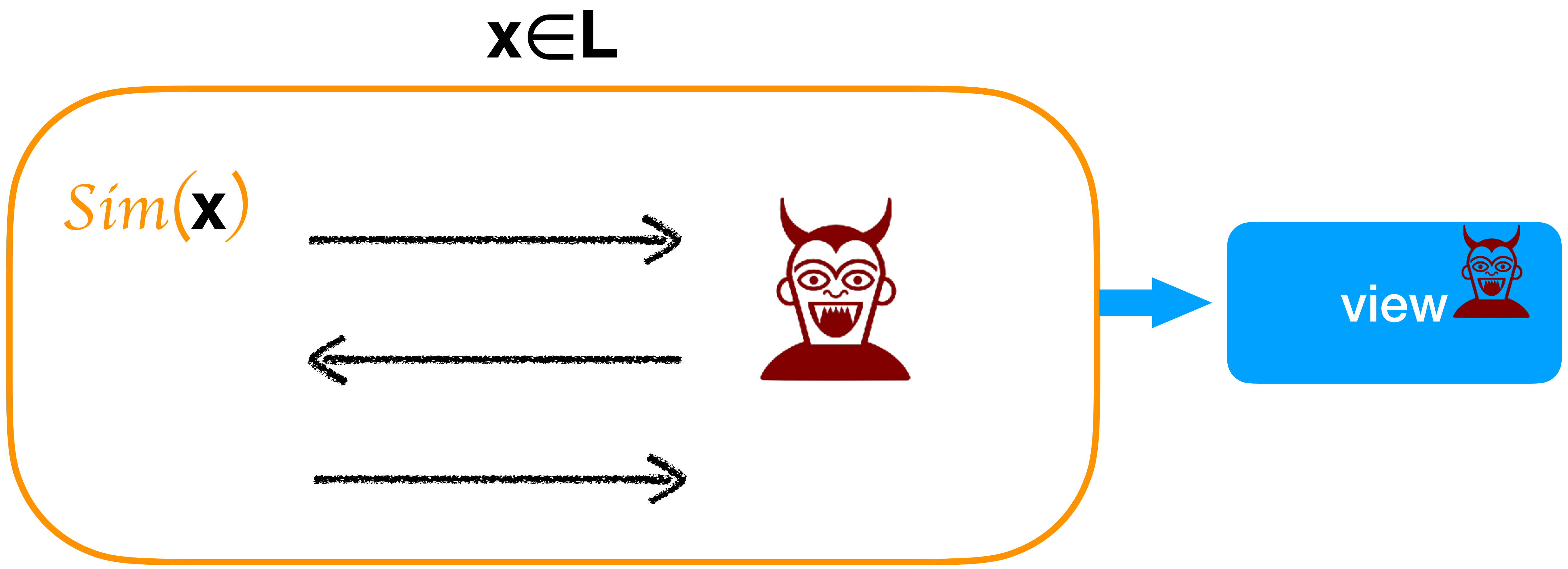
So far

- **ZK** implies **WI**
- **WI** composes (concurrently)
- The four-round computational zero-knowledge argument of knowledge for Hamiltonian graphs
- **NP** $\subseteq$ **CZK** will be in four rounds, assuming statistically binding commitments.
- **NP** $\subseteq$ **SZK** in four rounds assuming statistically hiding commitments
- Can we do better than 4 rounds?

Impossibility for languages outside BPP

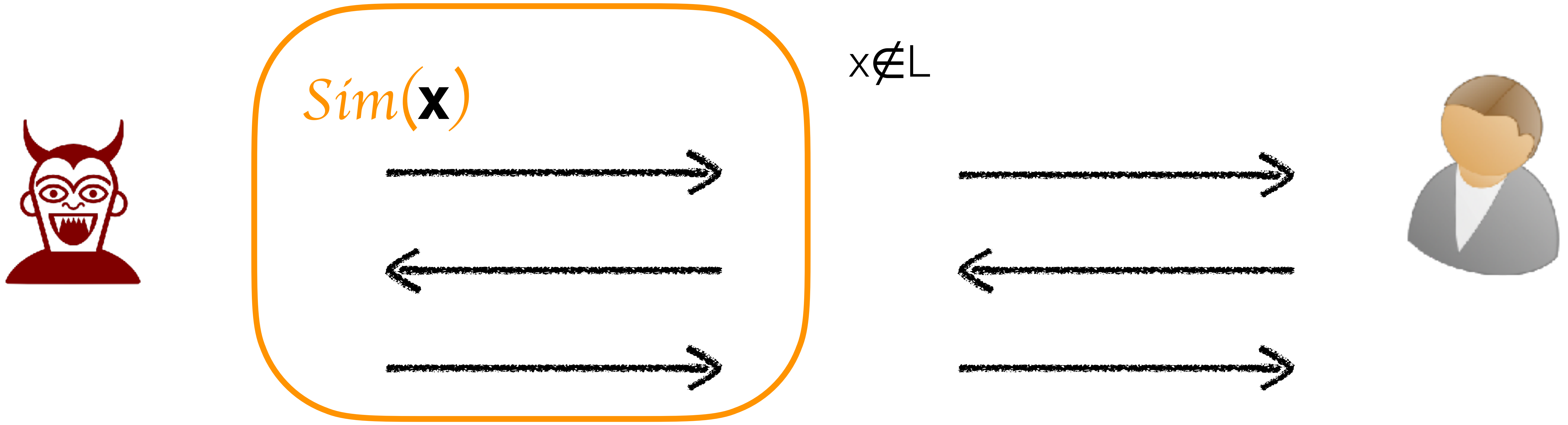


Zero-Knowledge and negligible soundness error



What happens if I run the simulator with x∉L

If we assume that it is difficult to decide whether x∉L or x∈L then the simulator must work in the same way

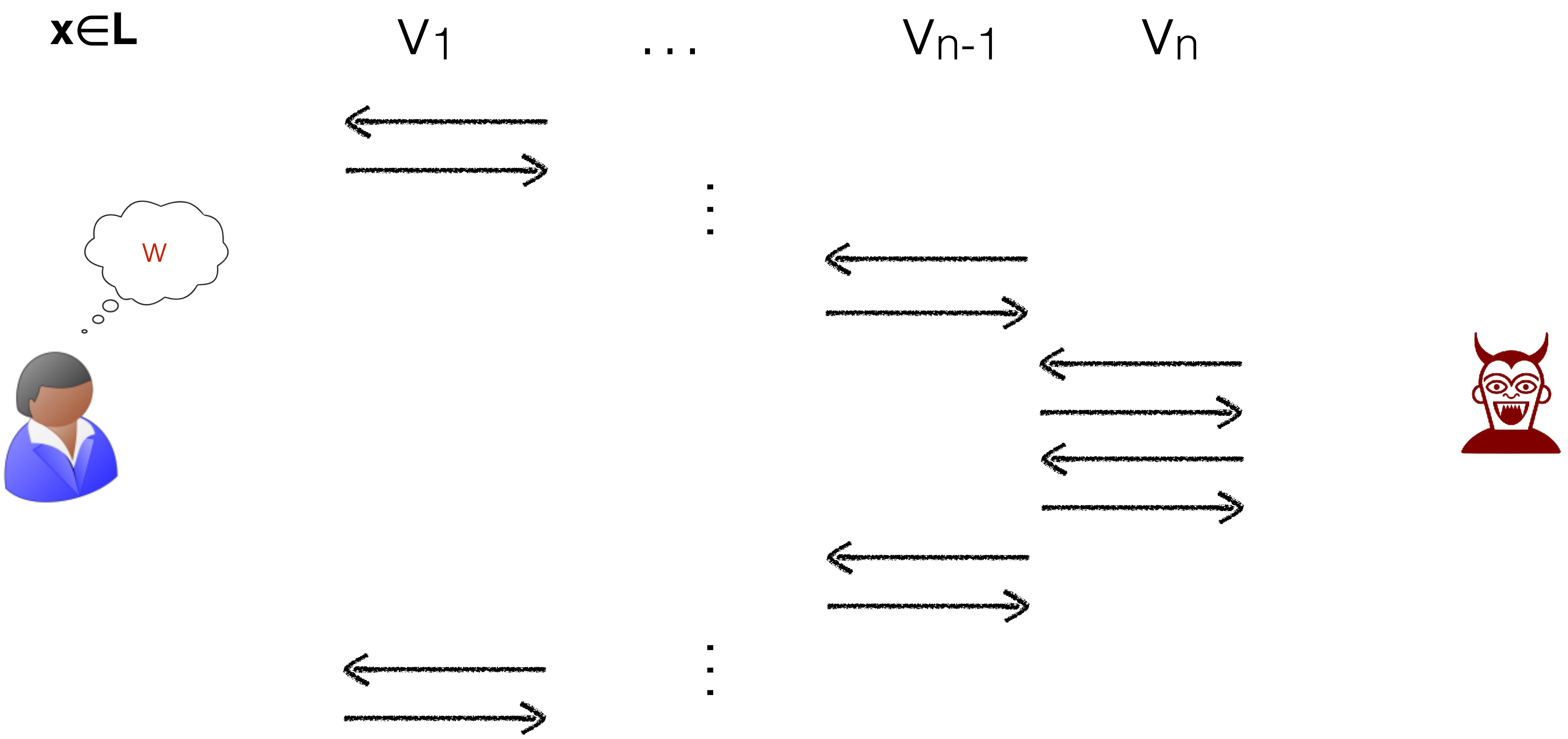


For non-trivial languages and with BB simulation 4-round is the best we can do

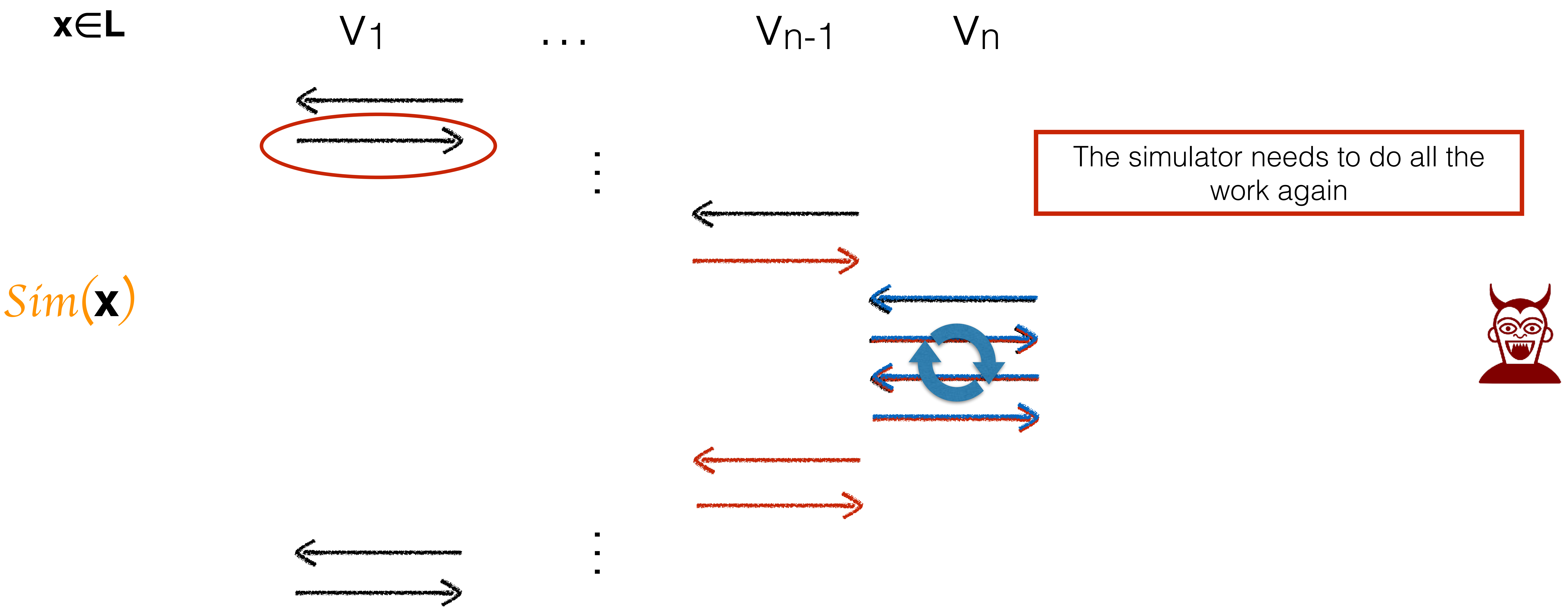
About compositoin

- The standalone setting for zero-knowledge.
- We made one attempt at parallel composition and it failed
- Can we design a constant round protocol that can be run in concurrency?
  - The schedule of the messages is arbitrary (maliciously chosen) [**DNS98**]

Concurrent composition



Concurrent composition



How many steps does the simulation of concurrent executions take?

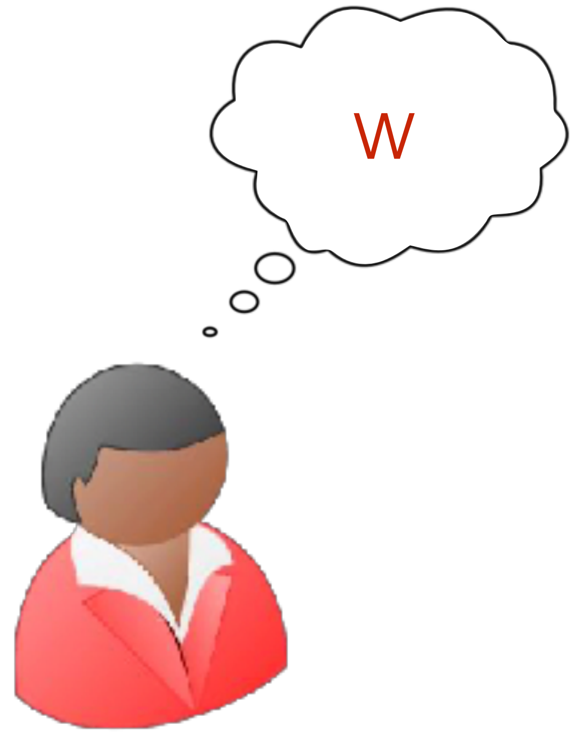
About compositioin

- [**DNS98**] Concurrent composition of constant round protocols becomes possible in the timing model
- [**D00**] If we assume trusted setup, then every language in NP has a constant round zero-knowledge protocol
- [**KPR98,CKPR01**] Only languages in BPP have BB concurrent ZK with *o(log n/log log n)* rounds
- [**KP01,PRS02**] Every language in NP has a concurrent ZK protocol with ω(log n) rounds.
- If the number of sessions is known apriori then constant round protocols are possible

Summary

- Sigma-Protocol
- Every language in NP has a sigma-protocol
- Boost security from HVZK to zero-knowledge
- The best possible round complexity is 4 round
- Can we circumvent the 3-round impossibility and design an efficient non-interactive argument?

How do we make non-interactive proofs?



$\mathbf{x}$

$\mathcal{O}$

$a \leftarrow P_\Sigma(\mathbf{x})$

$c \leftarrow \mathcal{O}(a,\mathbf{x})$

$z \leftarrow P_\Sigma(\mathbf{x},w,c)$

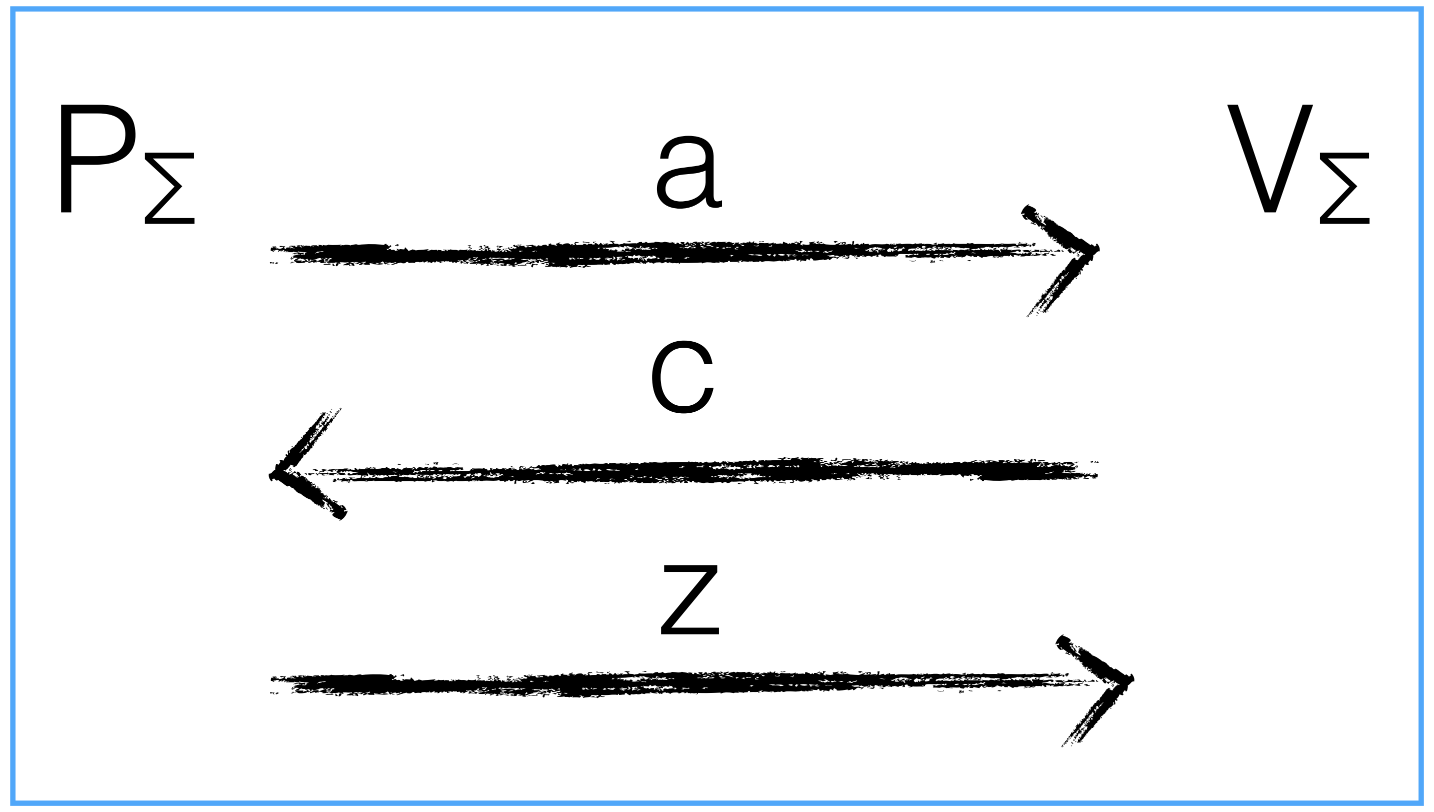
a,z →

$c \leftarrow \mathcal{O}(a,\mathbf{x})$

$V_\Sigma(a,c,z)=1$

- Fiat-Shamir transform
- in practice $\mathcal{O}$ is a hash function (e.g.SHA2)

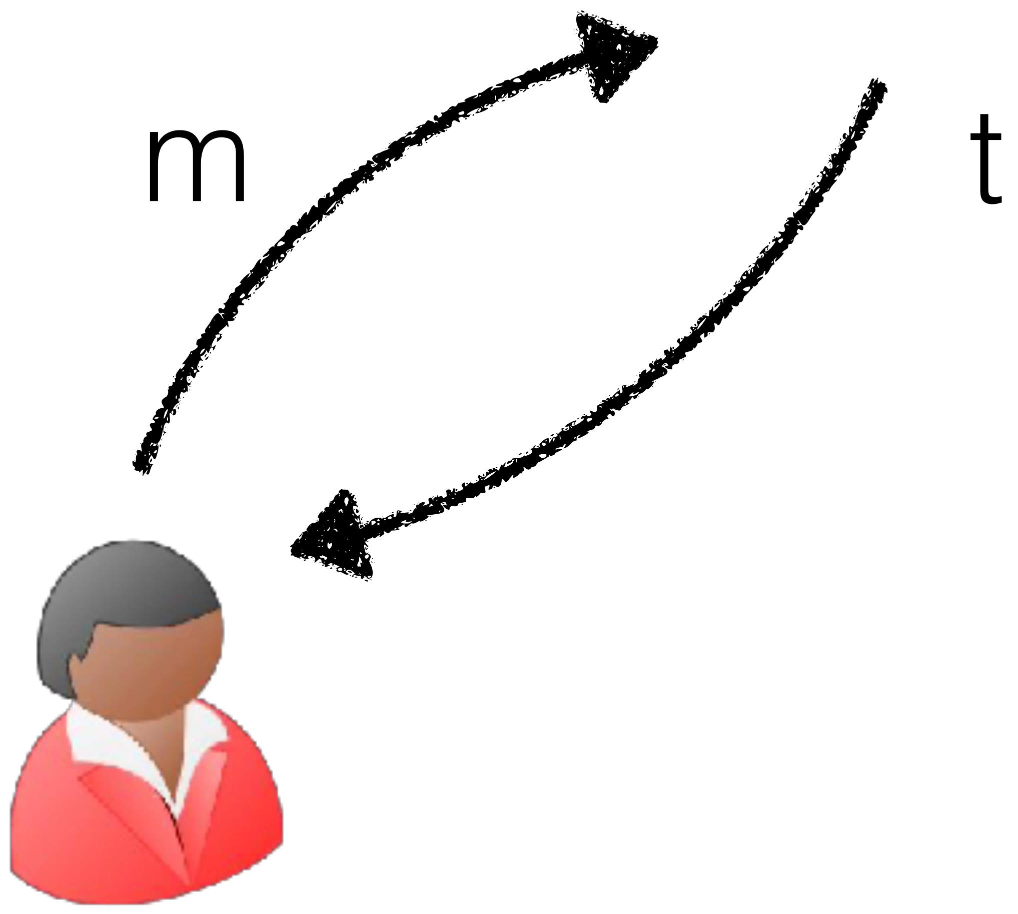
- Adds very little overhead to the starting sigma-protocol
- Used in practice for identification scheme, signatures, SNARKS, …



The Random Oracle Model [**BR93**]

O

- Given a query m, s.t. $(m, t) \in$ History for some t, then return t.
- Given a query m .s.t $(m, \cdot) \notin$ History then pick a random $t \leftarrow \{0,1\}^n$, add (m,t) to History and return t

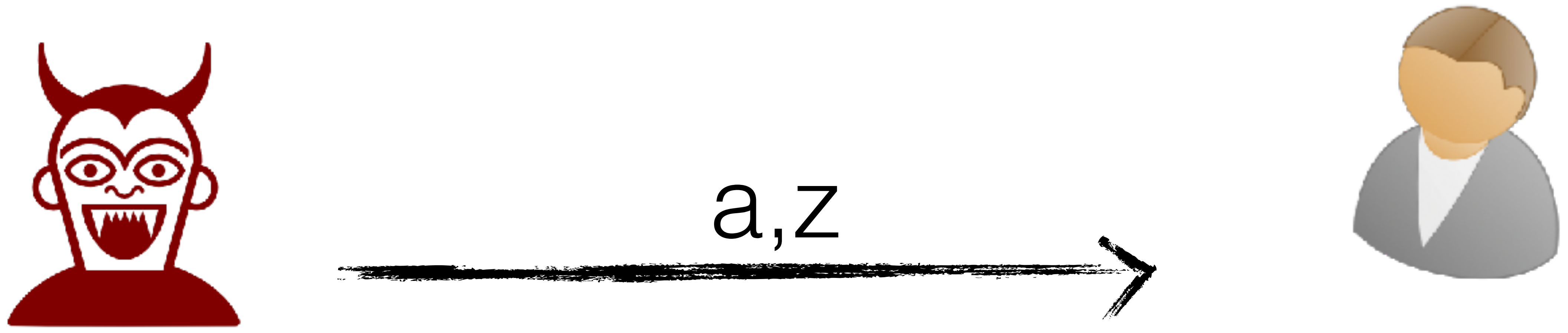


- It is an ideal functionality and nobody has its description
- Can only be treated like a black-box
- Security holds with high probability over the choice of O
- The reduction can control the RO

Soundness of Fiat-Shamir

$\mathbf{x} \notin L$

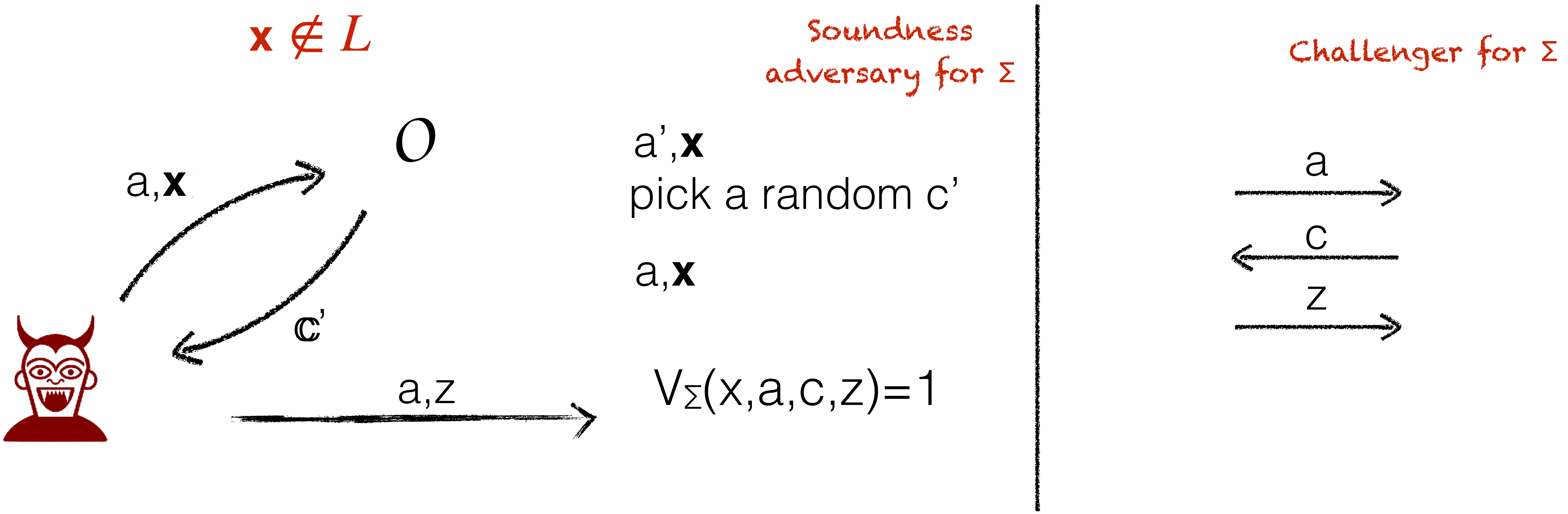
O



$c \leftarrow O(a, \mathbf{x})$

$V_{\Sigma}(a,c,z)=1$

Soundness of Fiat-Shamir



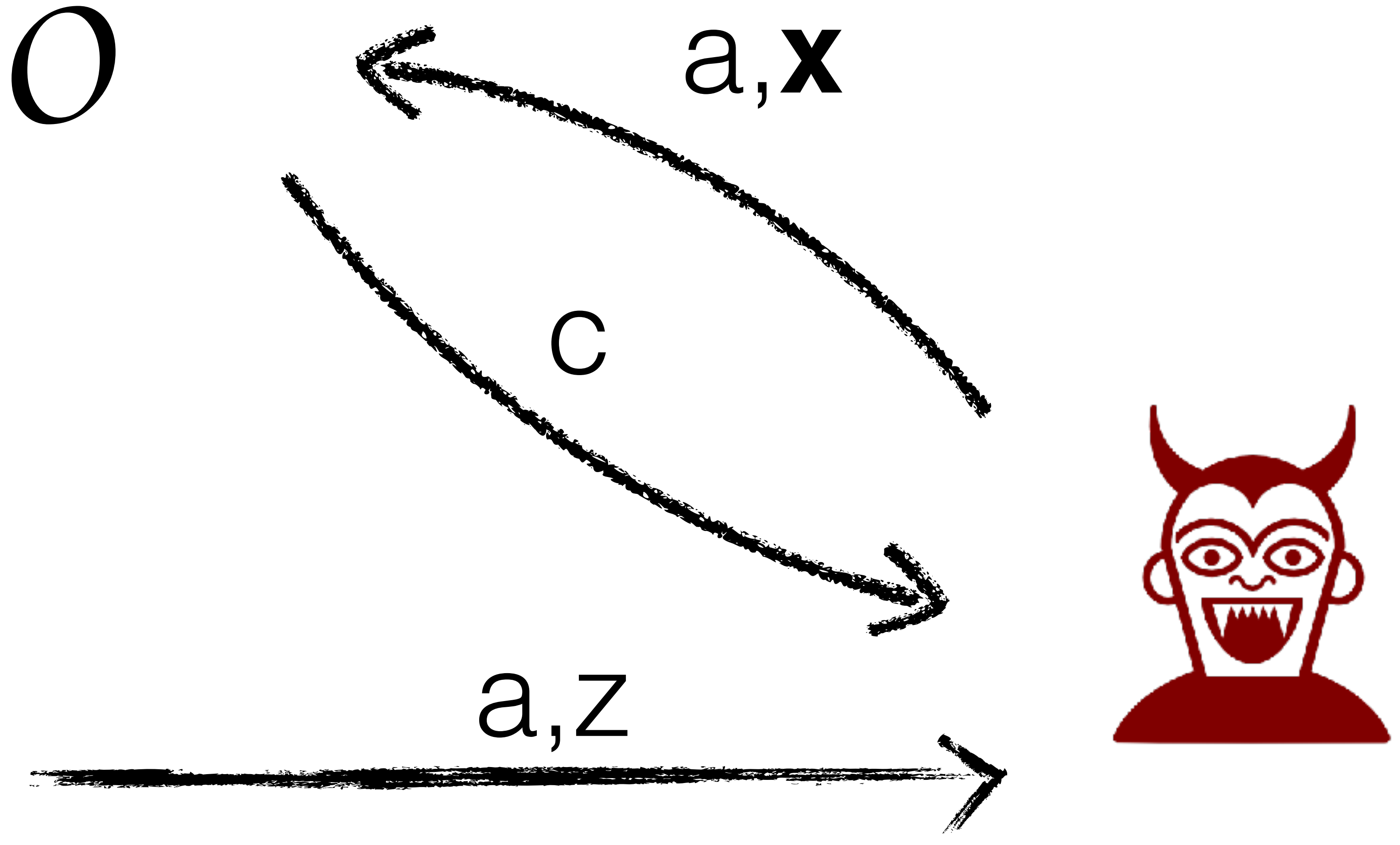
We have turned a successful adversary for the soundness of the FS-transform into an adversary that breaks the soundness of the sigma-protocol

Formally proving this requires a more involved analysis based on the forking lemma

Zero-Knowledge of Fiat-Shamir

x∈L

Sim(**x**)



- $c \leftarrow \{0,1\}^n$
- SHVZK(**x**,c)->a,c,z

- Various ways to define zero-knowledge
- A programmable hash function suffices (like a CRS)
- Is this still *zero knowledge*?

A bit more discussion on the RO model

- Hash functions are far from being random functions (PRFs?)
- [**CGH98**] Exist protocols secure in the RO model but broken when replacing the RO with *any* hash function

Optimistic view

- Counterexamples have very specific characteristics
- Better to have proof than no proof at all
- Good heuristic
- Recent results show that the FS transform if the RO is replaced with a special type of hash function and a special type of sigma-protocols is used*[**HMR08,CCH+19**]

Pessimistic view

- Basing security on assumptions we do not understand is undesirable

Summary and Conclusions

- It works with constant round public coin protocols with negligible soundness error (tight)
- It prevents malleability attacks (a stronger form of zero-knowledge, but assuming a quite strong setup).
- Setup is needed if we want to circumvent the 4-round impossibility
  - Weaker notions still exist that do not require setup (witness hiding, weak zero-knowledge, …)
- Setup is needed for full composition
- The plain model provides a pure form of zero-knowledge
- Pick your tool, depending on your application: you do not always need the strongest possible protection

References

- [D10] On Sigma-Protocols. Ivan Damgaard. https://www.cs.au.dk/~ivan/Sigma.pdf
- [DNS98] Cynthia Dwork, Moni Naor, Amit Sahai. Concurrent Zero-Knowledge
- [D00] Ivan Damgaard. Efficient Concurrent Zero-Knowledge in the Auxiliary String Model
- [CKPR01] Ran Canetti, Joe Kilian, Erez Petrank, Alon Rosen. Black-Box Concurrent Zero-Knowledge Requires (Almost) Logarithmically Many Rounds.
- [KPR98] Joe Kilian, Erez Petrank, Charles Rackoff. Lower Bounds for Zero Knowledge on the Internet.
- [KP01] Joe Kilian, Erez Petrank. Concurrent and resettable zero-knowledge in poly-loalgorithm rounds. STOC 2001.
- [FS90] Uriel Feige, Adi Shamir. Witness Indistinguishable and Witness Hiding Protocols. STOC 1990
- [FLS90] Uriel Feige, Dror Lapidot, and Adi Shamir. Multiple non-interactive zero knowledge proofs based on a single random string (extended abstract).
- [CGH98] Ran Canetti, Oded Goldreich, Shai Halevi: The Random Oracle Methodology, Revisited.
- [CCH+19] Ran Canetti, Yilei Chen, Justin Holmgren, Alex Lombardi, Guy N. Rothblum, Ron D. Rothblum, Daniel Wichs. Fiat-Shamir: from practice to theory. STOC 2019.
- [HMR08] Shai Halevi, Steven Myers, and Charles Rackoff, On seed-incompressible functions

Thank you