Characterizing Deterministic-Prover Zero Knowledge

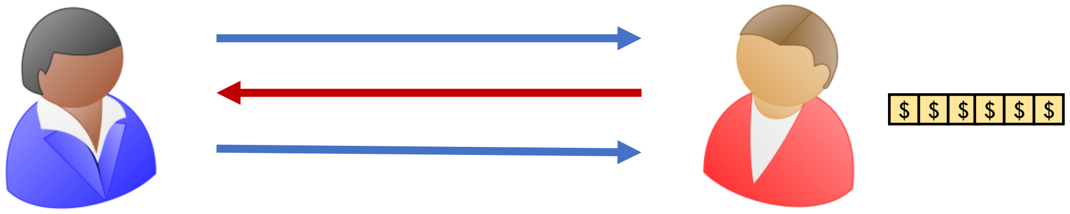
Nir Bitansky

Tel Aviv University

Arka Rai Choudhuri

Johns Hopkins University

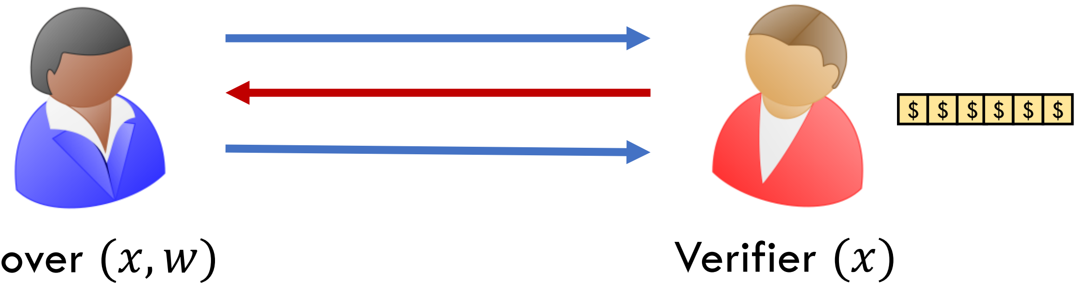
Zero Knowledge [Goldwasser-Micali-Rackoff’85]



Prover (x, w)

Verifier (x)

Zero Knowledge [Goldwasser-Micali-Rackoff’85]

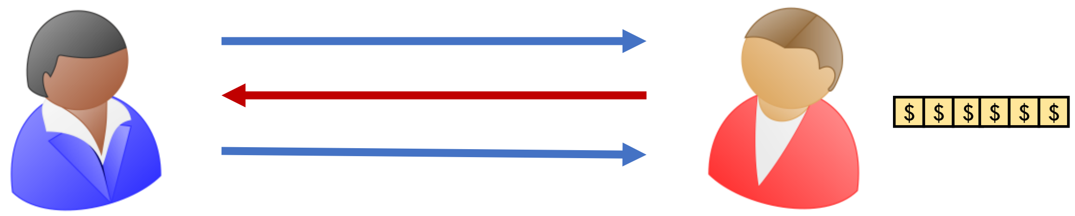


Completeness: $\forall x \in \mathcal{L}$, verifier accepts.

(Computational) Soundness

Zero Knowledge

Zero Knowledge [Goldwasser-Micali-Rackoff’85]



Prover (x, w)

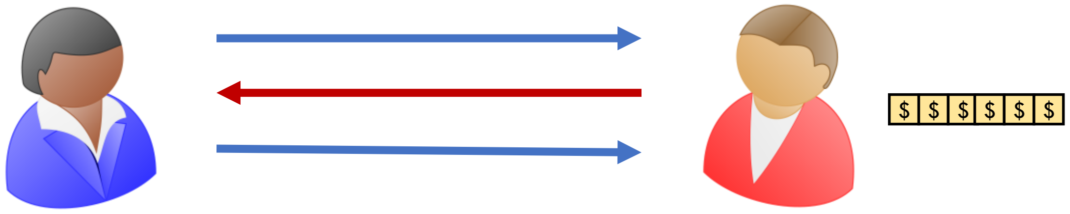
Verifier (x)

Completeness

(Computational) Soundness: $\forall x \notin \mathcal{L}$, no PPT prover  can make the verifier accept.

Zero Knowledge

Zero Knowledge [Goldwasser-Micali-Rackoff'85]



Prover (x, w)

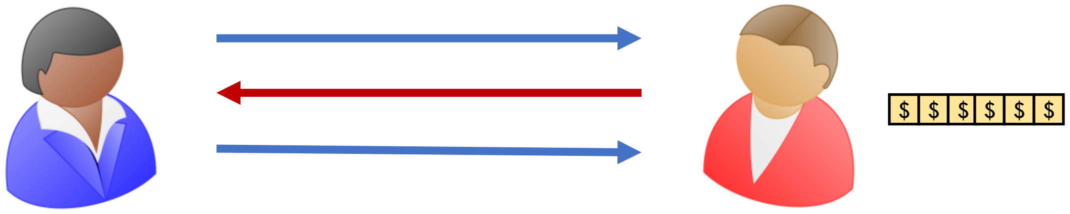
Verifier (x)

Completeness

(Computational) Soundness

Zero Knowledge: ∀ Verifiers ∃ Simulator

Zero Knowledge [Goldwasser-Micali-Rackoff’85]

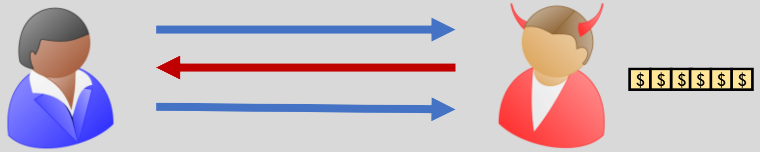


Prover (x, w) Verifier (x)

Completeness

(Computational) Soundness

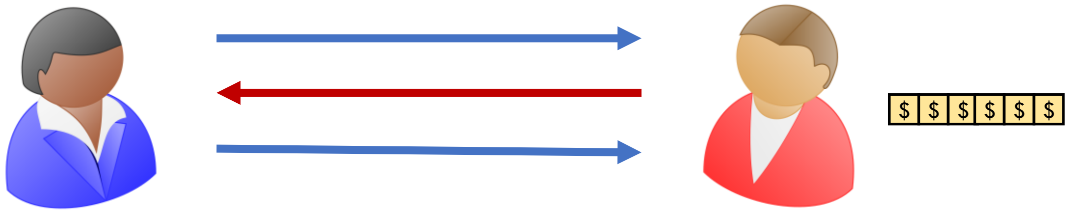
Zero Knowledge: ∀ Verifiers ∃ Simulator



Prover (x, w)

Verifier’s view in an execution with the prover

Zero Knowledge [Goldwasser-Micali-Rackoff’85]

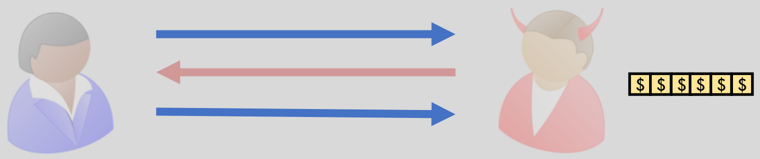


Prover (x, w) Verifier (x)

Completeness

(Computational) Soundness

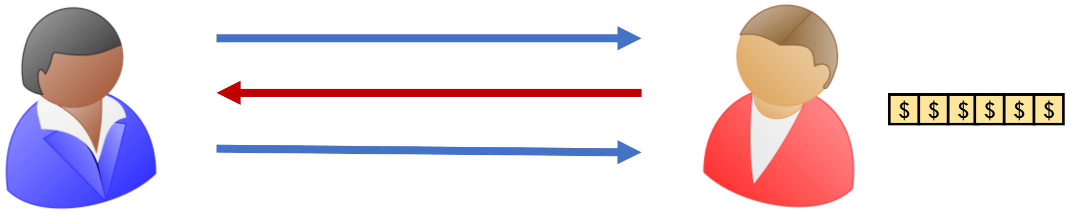
Zero Knowledge: ∀ Verifiers ∃ Simulator



Prover (x, w)

Verifier’s view in an execution with the prover

Zero Knowledge [Goldwasser-Micali-Rackoff'85]



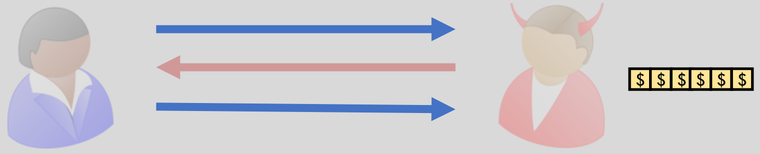
Prover (x, w)

Verifier (x)

Completeness

(Computational) Soundness

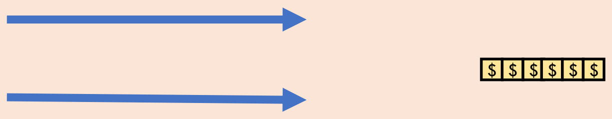
Zero Knowledge: ∀ Verifiers ∃ Simulator



Prover (x, w)

Verifier's view in an execution with the prover

$\approx$



Simulator's output on input x

Many Flavors of Zero-Knowledge (ZK)

$\forall$ Verifier

$\exists$ Simulator

View $\approx$ (x)

GMR ZK

Many Flavors of Zero-Knowledge (ZK)

∀ Verifier

∃ Simulator

View $\approx$ (x)

GMR ZK

∀ Verifier

∃ Simulator

∀ aux-IP $z \in \{0,1\}^*$

View $\approx$ (x, z)

Auxiliary-input ZK

Many Flavors of Zero-Knowledge (ZK)

∀ Verifier

∃ Simulator

View $\approx$ (x)

GMR ZK

∀ Verifier

∃ Simulator

∀ aux-IP $z \in \{0,1\}^*$

View $\approx$ (x, z)

Auxiliary-input ZK

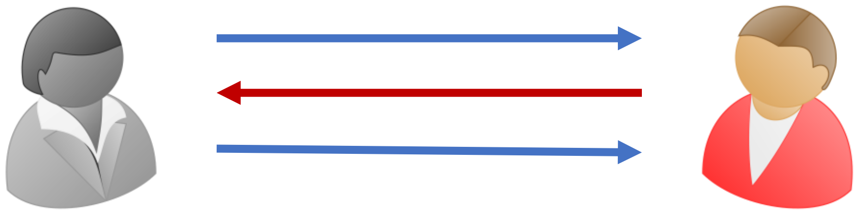
∃ Simulator

∀ Verifier

View $\approx$ (x)

Black-box ZK

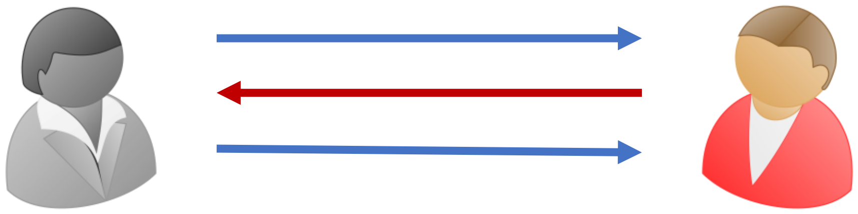
Deterministic Prover Zero Knowledge (DPZK)



Deterministic Prover (x, w)

Verifier (x)

Deterministic Prover Zero Knowledge (DPZK)



Deterministic Prover (x, w)

Verifier (x)

Is prover randomness essential for zero knowledge?

Limitations of DPZK [Golreich-Oren’94]

∀ Verifier
∃ Simulator

View ≈ (x)

GMR ZK

∀ Verifier
∃ Simulator
∀ aux-IP $z \in \{0,1\}^*$

View ≈ (x, z)

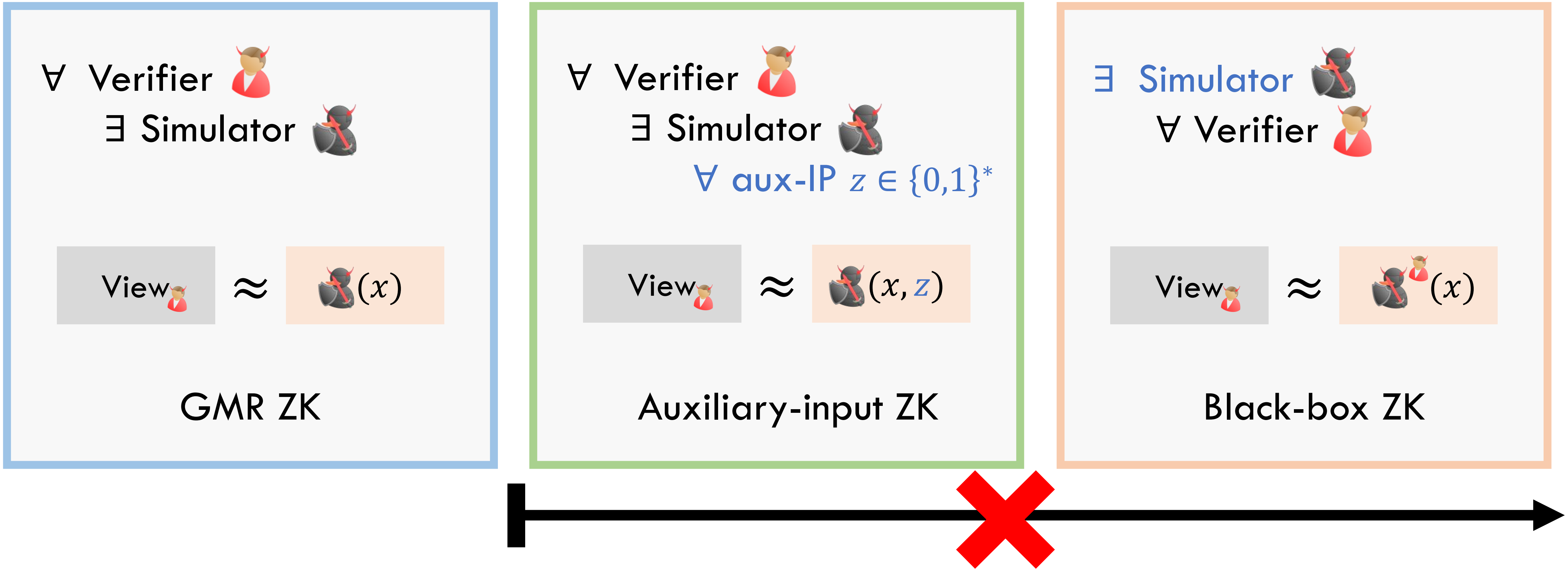
Auxiliary-input ZK

∃ Simulator
∀ Verifier

View ≈ (x)

Black-box ZK

Limitations of DPZK [Golreich-Oren'94]



$\forall$ Verifier

$\exists$ Simulator

View $\approx$ (x)

GMR ZK

$\forall$ Verifier

$\exists$ Simulator

$\forall$ aux-IP $z \in \{0,1\}^*$

View $\approx$ (x, z)

Auxiliary-input ZK

$\exists$ Simulator

$\forall$ Verifier

View $\approx$ (x)

Black-box ZK

Impossible for non-trivial languages.

Prior Work

[Faonio-Nielsen-Venturi'17]

Witness encryption for $\mathcal{L} \Longrightarrow$ Honest-verifier DPZK for $\mathcal{L}$

Hash proof system for $\mathcal{L} \Longrightarrow$ Honest-verifier DPZK proofs for $\mathcal{L}$

Prior Work

[Faonio-Nielsen-Venturi'17]

Witness encryption for $\mathcal{L} \Longrightarrow$ Honest-verifier DPZK for $\mathcal{L}$

Hash proof system for $\mathcal{L} \Longrightarrow$ Honest-verifier DPZK proofs for $\mathcal{L}$

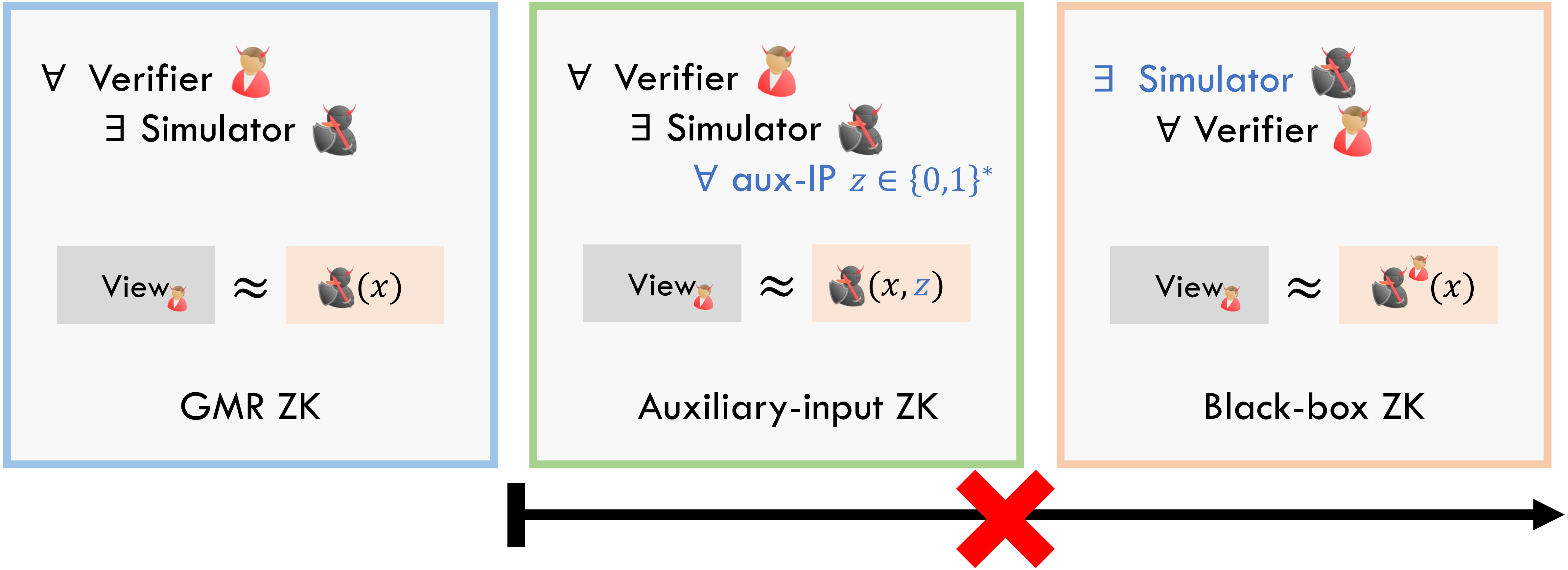
[Dahari-Lindell'20]

Doubly enhanced injective OWFs $\Longrightarrow$ Honest-verifier DPZK proofs for NP

Inefficient honest prover.

Malicious-verifier DPZK for languages that have an entropy guarantee from witnesses.

Our Results



∀ Verifier

∃ Simulator

View $\approx$ (x)

GMR ZK

∀ Verifier

∃ Simulator

∀ aux-IP $z \in \{0,1\}^*$

View $\approx$ (x, z)

Auxiliary-input ZK

∃ Simulator

∀ Verifier

View $\approx$ (x)

Black-box ZK

Impossible for non-trivial languages.

Our Results

$\forall$ Verifier
$\exists$ Simulator

View $\approx$ (x)

GMR ZK

$\forall$ Verifier
$\exists$ Simulator
$\forall$ aux-IP $z \in \{0,1\}^b$

View $\approx$ (x, z)

b-Bounded auxiliary-input ZK

$\forall$ Verifier
$\exists$ Simulator
$\forall$ aux-IP $z \in \{0,1\}^*$

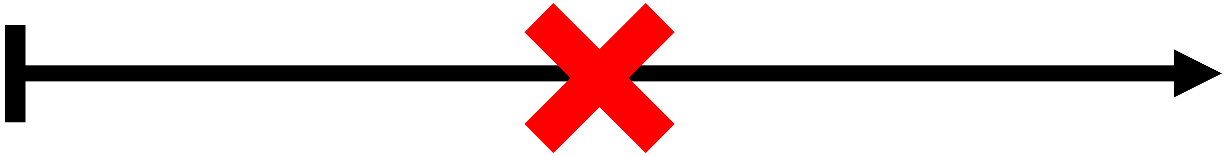
View $\approx$ (x, z)

Auxiliary-input ZK

$\exists$ Simulator
$\forall$ Verifier

View $\approx$ (x)

Black-box ZK



Impossible for non-trivial languages.

Our Results

∀ Verifier
∃ Simulator

View ≈ (x)

GMR ZK

∀ Verifier
∃ Simulator
∀ aux-IP $z \in \{0,1\}^b$

View ≈ (x, z)

b-Bounded auxiliary-input ZK

∀ Verifier
∃ Simulator
∀ aux-IP $z \in \{0,1\}^*$

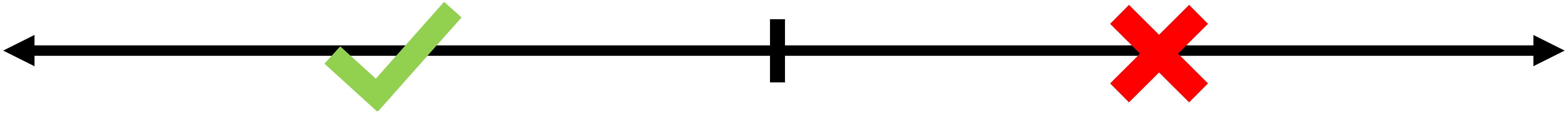
View ≈ (x, z)

Auxiliary-input ZK

∃ Simulator
∀ Verifier

View ≈ (x)

Black-box ZK



Impossible for non-trivial languages.

Our Results

Assuming NIWIs + sub-exponentially secure iO + OWF, there exist two message DPZK arguments for NP ∩ coNP against bounded auxiliary-input verifiers.

Also assuming sub-exponentially secure keyless CRHF, there exist two message DPZK arguments for all of NP against bounded auxiliary-input verifiers.

Our Results

Assuming NIWIs + sub-exponentially secure iO + OWF, there exist two message DPZK arguments for NP ∩ coNP against bounded auxiliary-input verifiers.

Also assuming sub-exponentially secure keyless CRHF, there exist two message DPZK arguments for all of NP against bounded auxiliary-input verifiers.

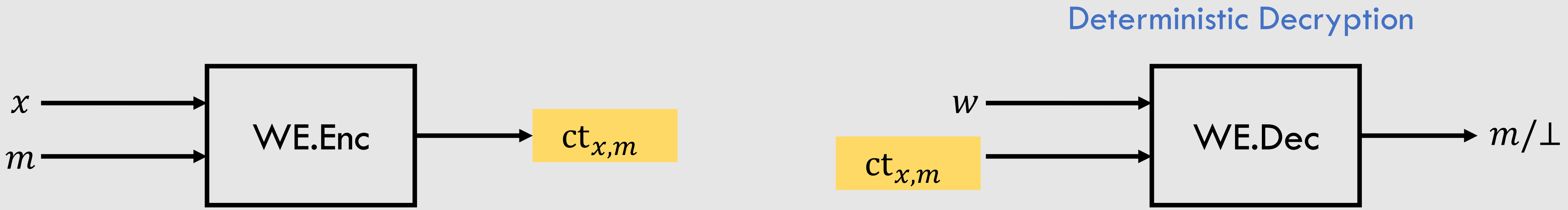
Any DPZK argument for a language $\mathcal{L}$ implies a witness encryption for $\mathcal{L}$.

Two Message DPZK Arguments

Honest Verifier DPZK [Faonio-Nielsen-Venturi’17]

Honest Verifier DPZK [Faonio-Nielsen-Venturi'17]

Witness Encryption for $\mathcal{L}$



Deterministic Decryption

x

m

WE.Enc

$\mathsf{ct}_{x,m}$

w

$\mathsf{ct}_{x,m}$

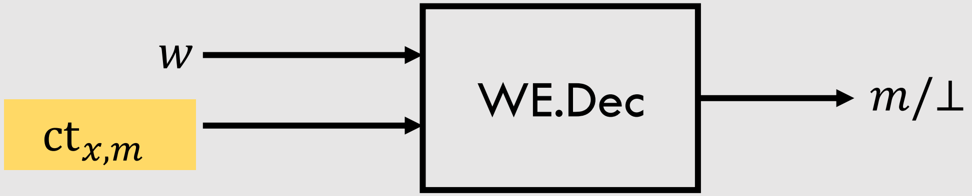
WE.Dec

$m/\perp$

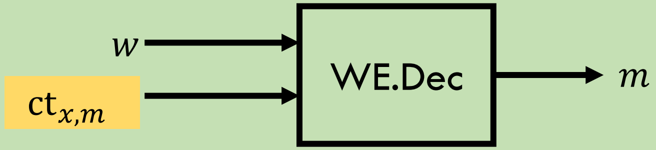
Honest Verifier DPZK [Faonio-Nielsen-Venturi’17]

Witness Encryption for $\mathcal{L}$

Deterministic Decryption



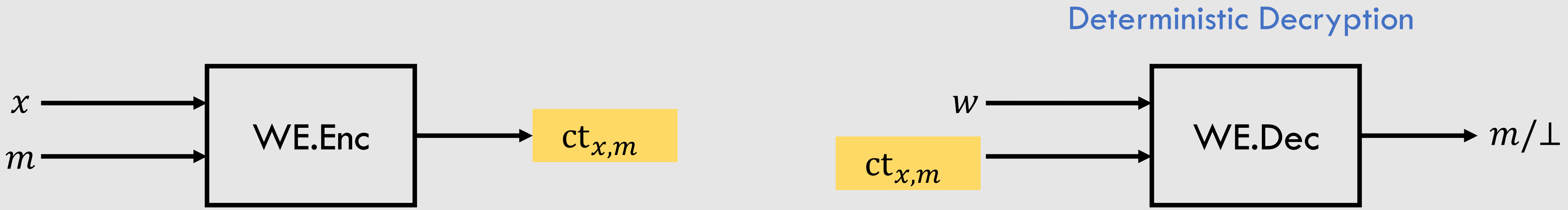
For $(x, w) \in \mathsf{Rel}_{\mathcal{L}}$



Correctness

Honest Verifier DPZK [Faonio-Nielsen-Venturi'17]

Witness Encryption for $\mathcal{L}$

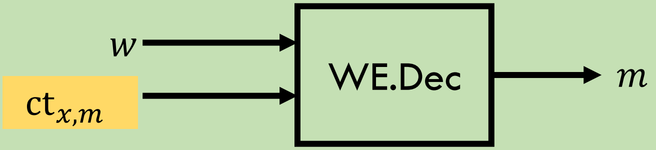


x, m → WE.Enc → $\mathsf{ct}_{x,m}$

Deterministic Decryption

w, $\mathsf{ct}_{x,m}$ → WE.Dec → $m/\perp$

For $(x, w) \in \mathsf{Rel}_{\mathcal{L}}$



w, $\mathsf{ct}_{x,m}$ → WE.Dec → m

Correctness

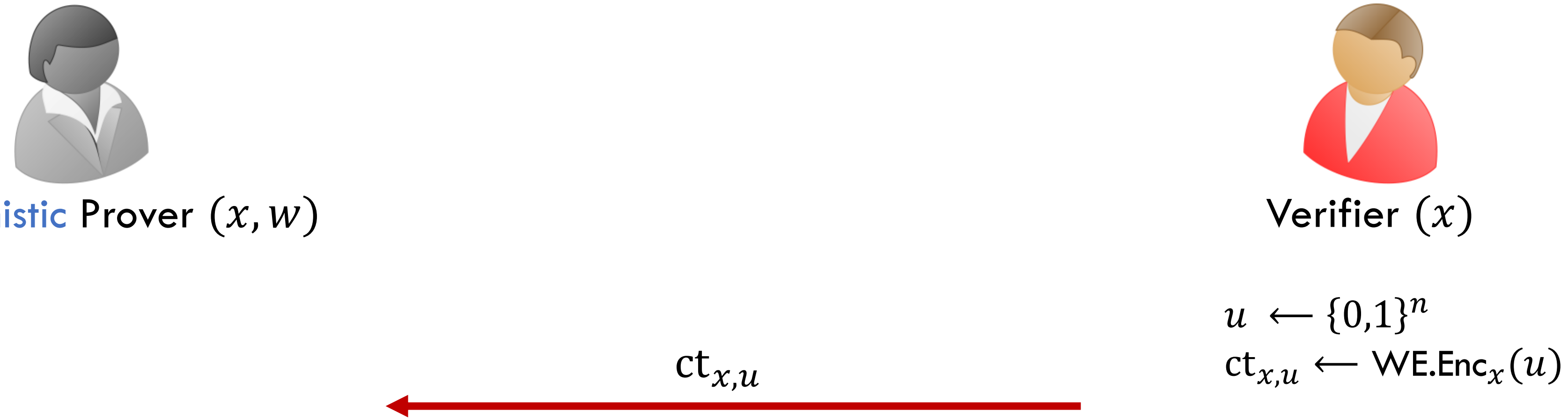
For $x \notin \mathcal{L}$

$\mathsf{ct}_{x,0} \approx \mathsf{ct}_{x,1}$

Security

Honest Verifier DPZK [Faonio-Nielsen-Venturi'17]

Honest Verifier DPZK [Faonio-Nielsen-Venturi'17]



Deterministic **Prover** (x, w)

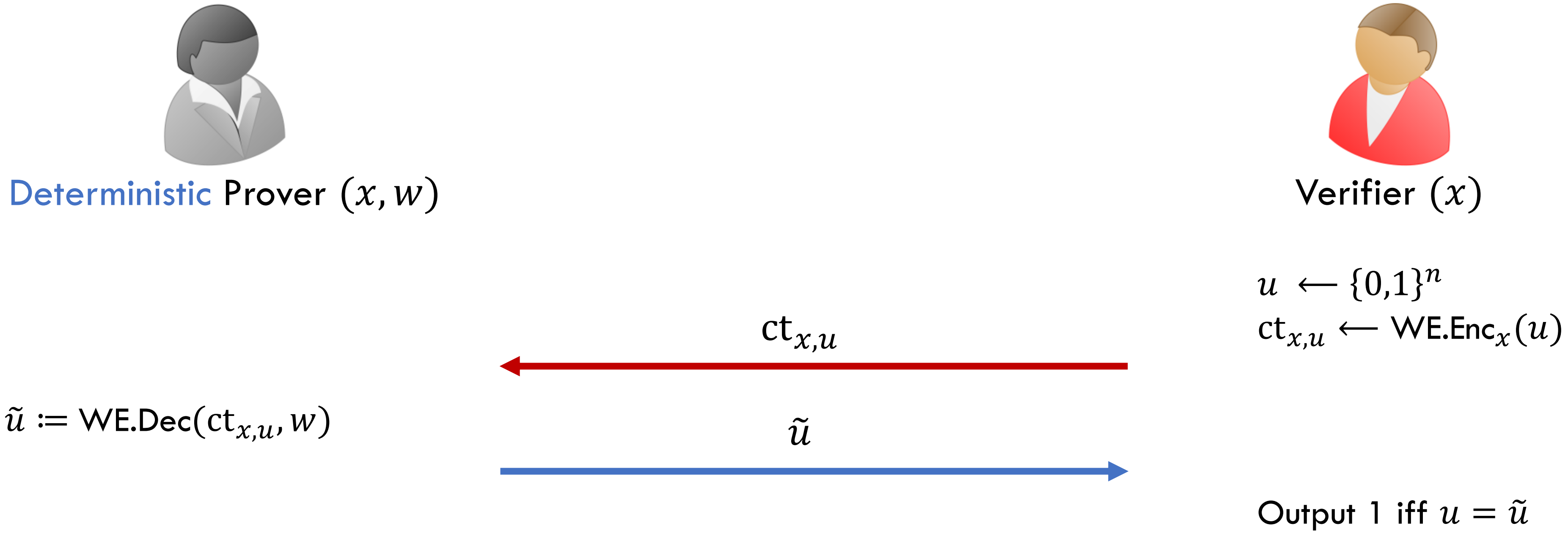
Verifier (x)

$u \longleftarrow \{0,1\}^n$

$\mathsf{ct}_{x,u} \longleftarrow \mathsf{WE.Enc}_x(u)$

$\mathsf{ct}_{x,u}$

Honest Verifier DPZK [Faonio-Nielsen-Venturi'17]



Deterministic **Prover** (x, w)

Verifier (x)

$u \leftarrow \{0,1\}^n$

$\mathrm{ct}_{x,u} \leftarrow \mathsf{WE.Enc}_x(u)$

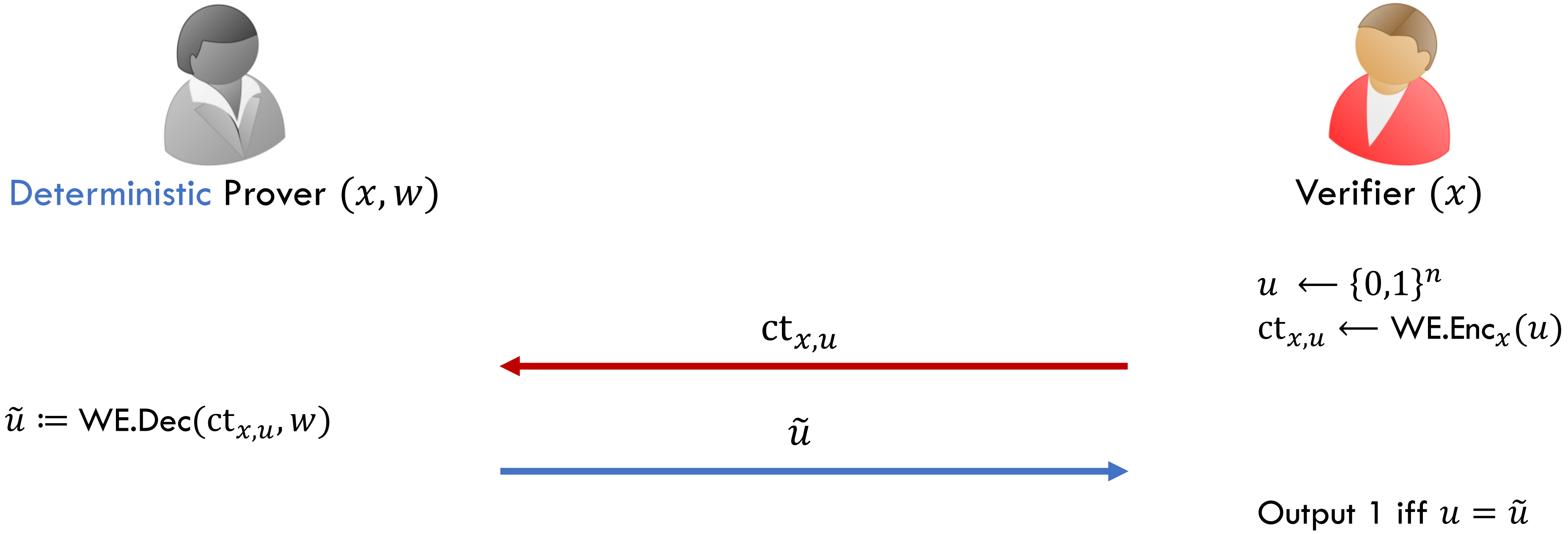
$\mathrm{ct}_{x,u}$

$\tilde{u} := \mathsf{WE.Dec}(\mathrm{ct}_{x,u}, w)$

$\tilde{u}$

Output 1 iff $u = \tilde{u}$

Honest Verifier DPZK [Faonio-Nielsen-Venturi'17]



Deterministic Prover (x, w)

Verifier (x)

$u \leftarrow \{0,1\}^n$
$\mathsf{ct}_{x,u} \leftarrow \mathsf{WE.Enc}_x(u)$

$\mathsf{ct}_{x,u}$

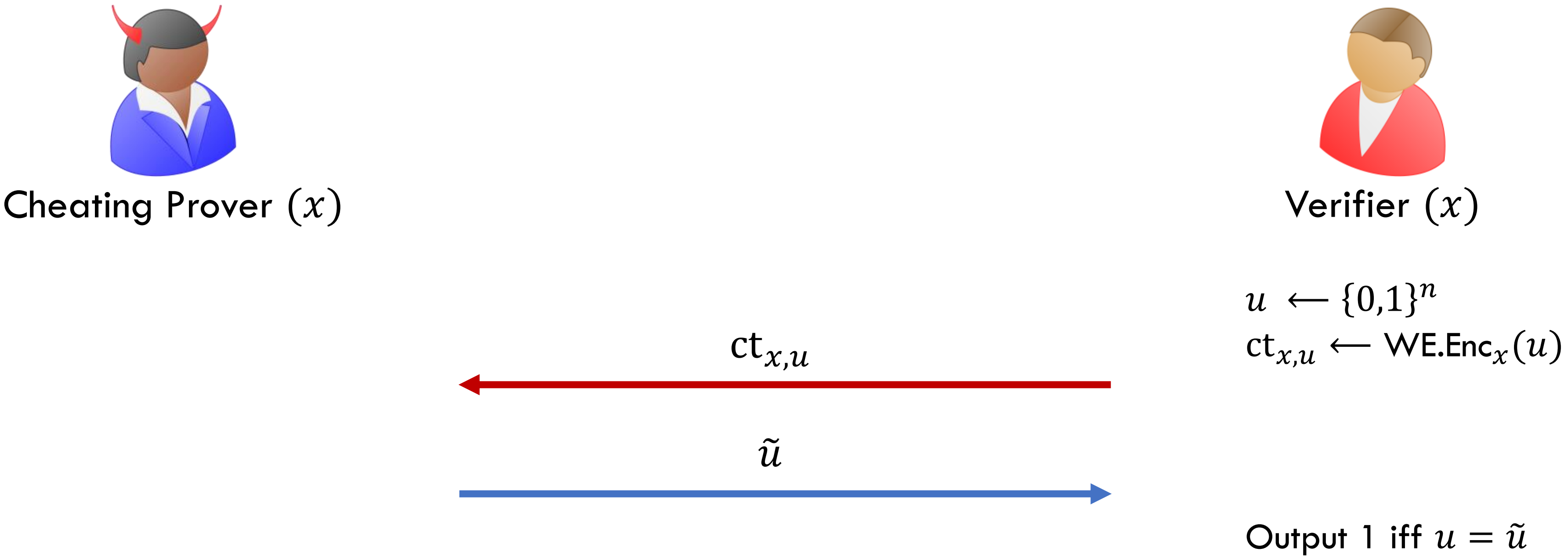
$\tilde{u} := \mathsf{WE.Dec}(\mathsf{ct}_{x,u}, w)$

$\tilde{u}$

Output 1 iff $u = \tilde{u}$

Completeness: From correctness of WE.

Honest Verifier DPZK [Faonio-Nielsen-Venturi'17]



Cheating Prover (x)

Verifier (x)

$u \leftarrow \{0,1\}^n$

$\mathsf{ct}_{x,u} \leftarrow \mathsf{WE.Enc}_x(u)$

$\mathsf{ct}_{x,u}$

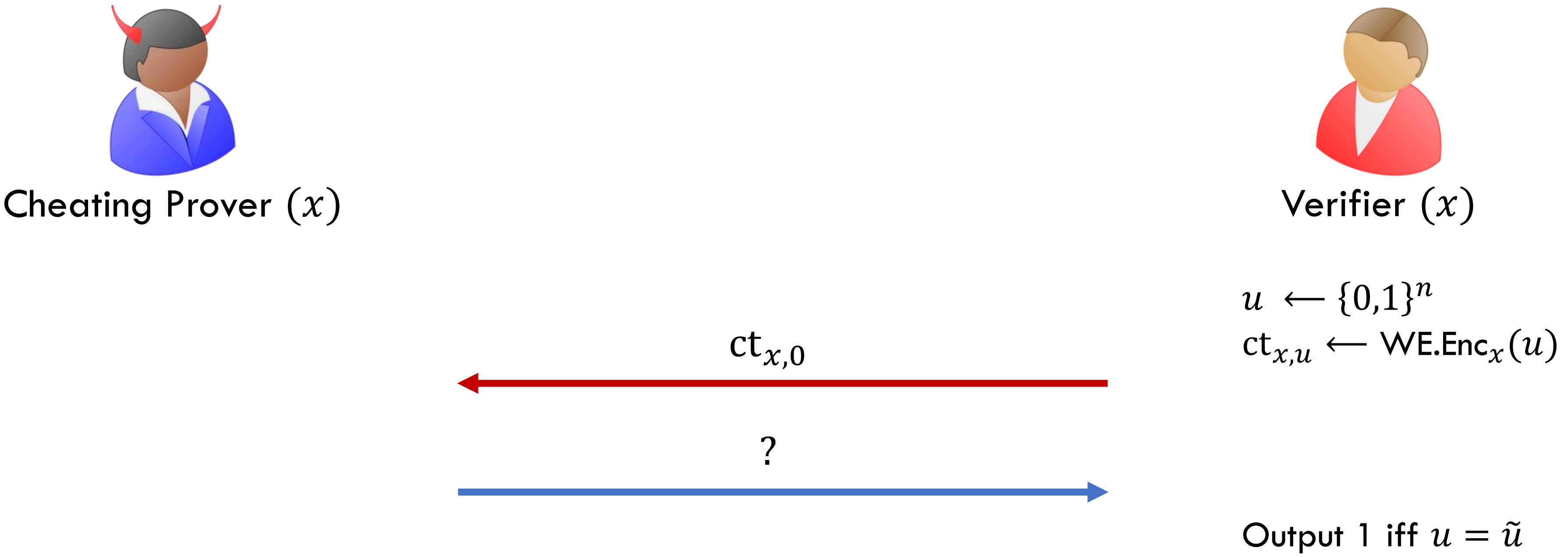
$\tilde{u}$

Output 1 iff $u = \tilde{u}$

Completeness

Soundness: From WE security when $x \notin \mathcal{L}$

Honest Verifier DPZK [Faonio-Nielsen-Venturi’17]



Cheating Prover (x)

Verifier (x)

$u \longleftarrow \{0,1\}^n$

$\mathsf{ct}_{x,u} \longleftarrow \mathsf{WE.Enc}_x(u)$

$\mathsf{ct}_{x,0}$

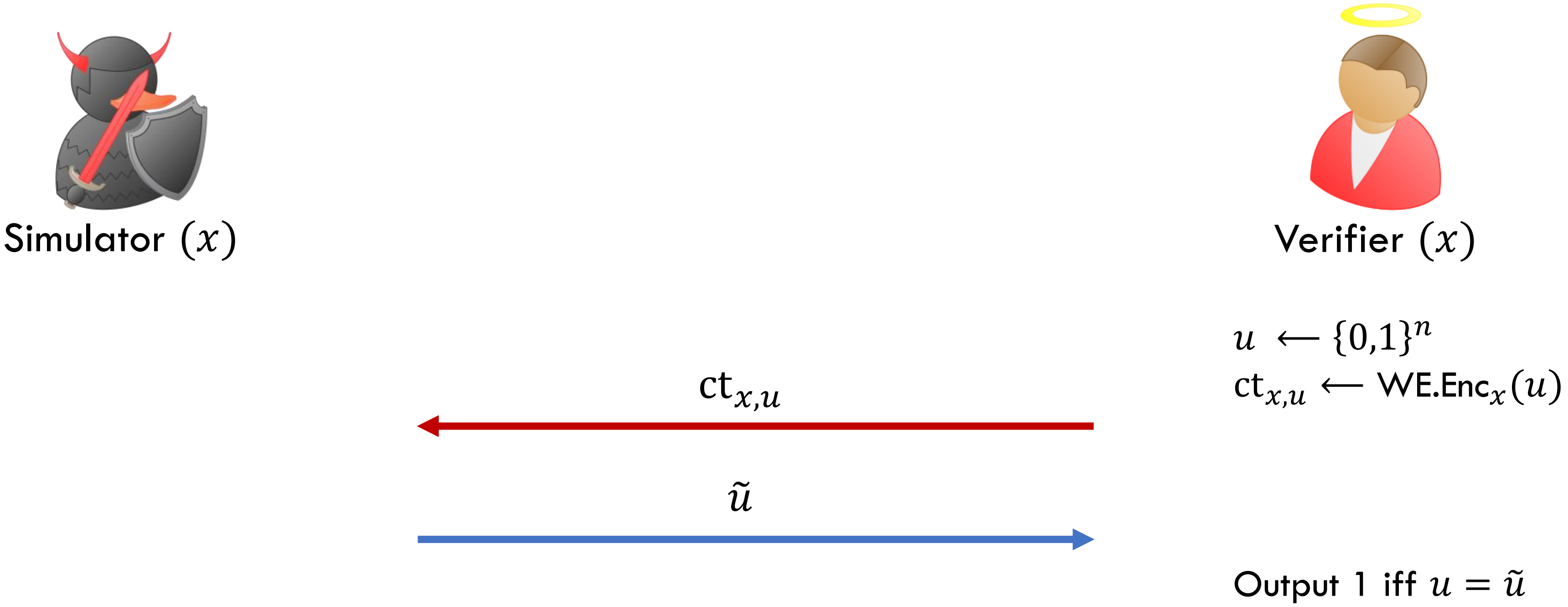
$?$

Output 1 iff $u = \tilde{u}$

Completeness

Soundness: From WE security when $x \notin \mathcal{L}$

Honest Verifier DPZK [Faonio-Nielsen-Venturi’17]



Simulator (x)

Verifier (x)

$u \leftarrow \{0,1\}^n$
$\mathsf{ct}_{x,u} \leftarrow \mathsf{WE.Enc}_x(u)$

$\xleftarrow{\mathsf{ct}_{x,u}}$

$\xrightarrow{\tilde{u}}$

Output 1 iff $u = \tilde{u}$

Completeness

Soundness

Honest Verifier Zero Knowledge: Simulator knows u

Explainable Verifier DPZK

Explainable Verifier

There exist honest verifier coins that explains verifier messages as honest messages.

Unlike related notion of semi-malicious adversaries, these coins may be hard to find.

Explainable Verifier DPZK

Explainable Verifier

There exist honest verifier coins that explains verifier messages as honest messages.

Unlike related notion of semi-malicious adversaries, these coins may be hard to find.

Simulator no longer “knows” the message that an explainable verifier encrypts via the Witness Encryption.

Aux-I/P DPZK for explainable verifiers also ruled out by [Goldreich-Oren’94]

Explainable Verifier DPZK

Explainable Verifier

There exist honest verifier coins that explains verifier messages as honest messages.

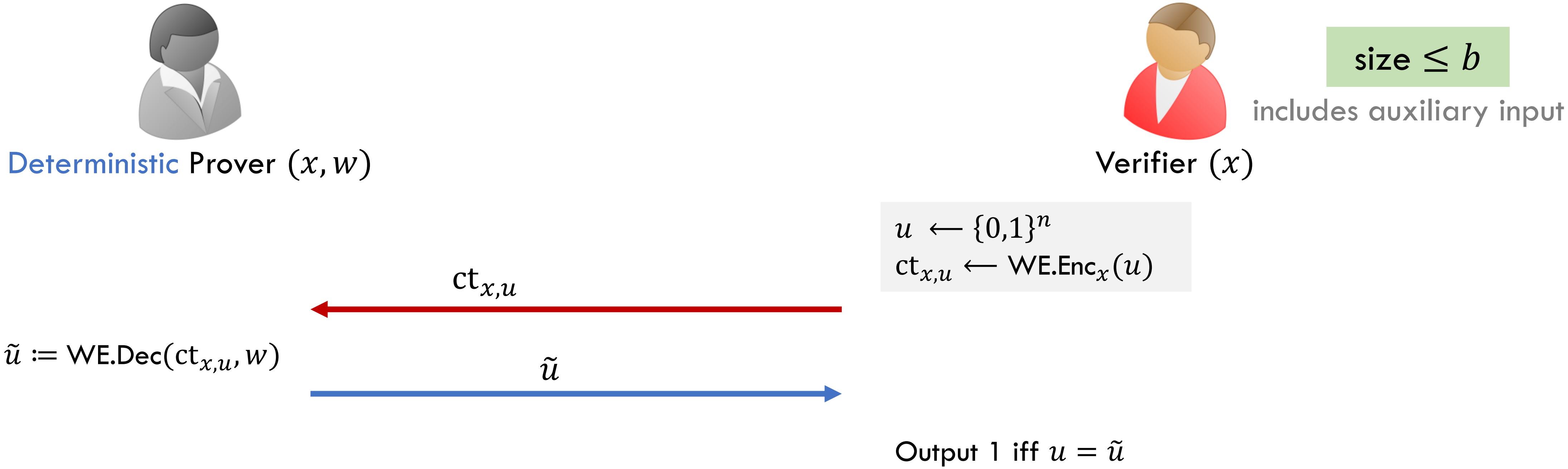
Unlike related notion of semi-malicious adversaries, these coins may be hard to find.

Simulator no longer "knows" the message that an explainable verifier encrypts via the Witness Encryption.

Aux-I/P DPZK for explainable verifiers also ruled out by [Goldreich-Oren'94]

Idea: Use additional trapdoor statement that only the simulator can use.

Explainable Verifier DPZK



Deterministic Prover (x, w)

Verifier (x)

size $\le b$

includes auxiliary input

$u \leftarrow \{0,1\}^n$
$\mathrm{ct}_{x,u} \leftarrow \mathrm{WE.Enc}_x(u)$

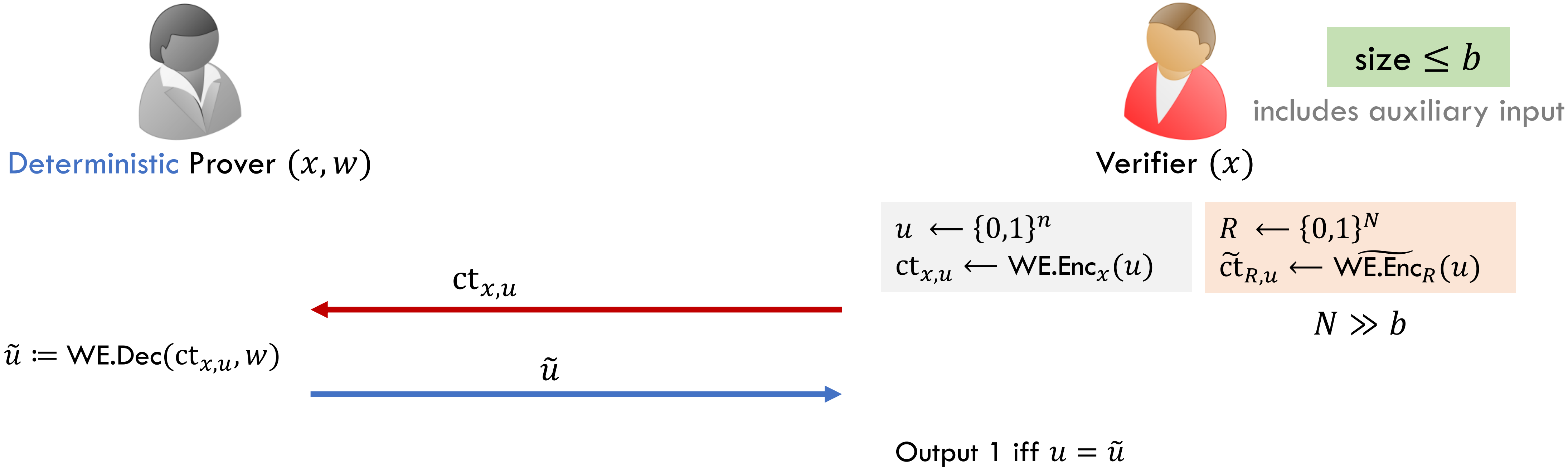
$\mathrm{ct}_{x,u}$

$\tilde{u} := \mathrm{WE.Dec}(\mathrm{ct}_{x,u}, w)$

$\tilde{u}$

Output 1 iff $u = \tilde{u}$

Explainable Verifier DPZK



size $\leq b$

includes auxiliary input

Deterministic Prover (x, w)

Verifier (x)

$u \leftarrow \{0,1\}^n$
$\mathsf{ct}_{x,u} \leftarrow \mathsf{WE.Enc}_x(u)$

$R \leftarrow \{0,1\}^N$
$\widetilde{\mathsf{ct}}_{R,u} \leftarrow \widetilde{\mathsf{WE.Enc}}_R(u)$

$N \gg b$

$\mathsf{ct}_{x,u}$

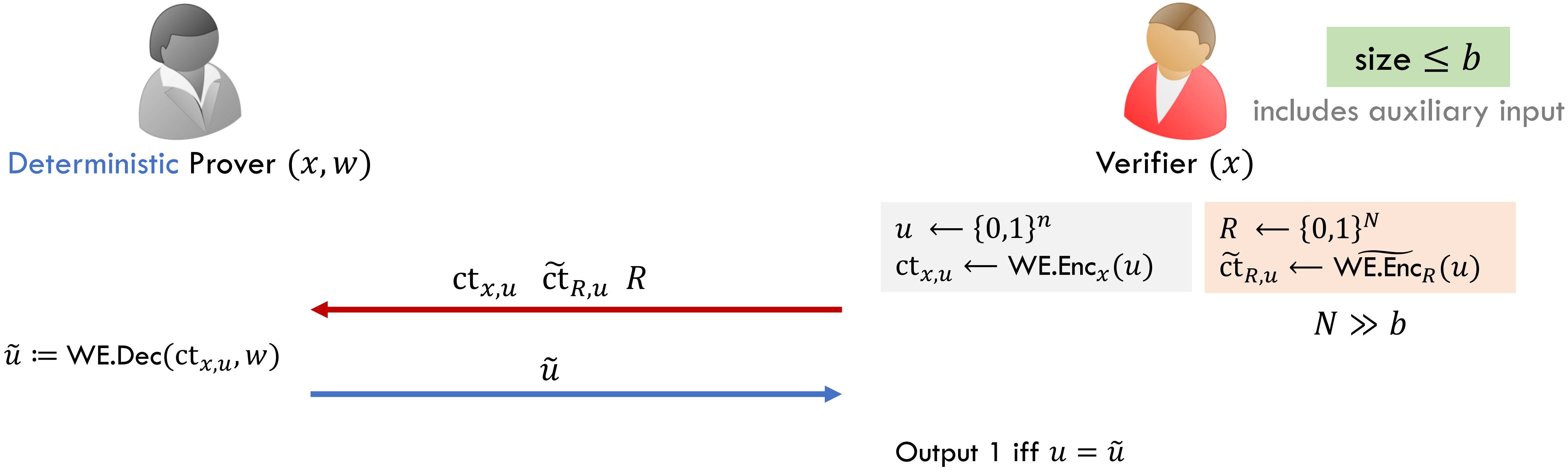
$\tilde{u} \coloneqq \mathsf{WE.Dec}(\mathsf{ct}_{x,u}, w)$

$\tilde{u}$

Output 1 iff $u = \tilde{u}$

$(R, \mathsf{M}) \in \mathsf{Rel}_{\tilde{\mathcal{L}}}$ if
1) M is a Turing Machine that outputs R.
2) Size of M is $b + \lambda$

Explainable Verifier DPZK



Deterministic **Prover** (x, w)

Verifier (x)

size $\leq b$

includes auxiliary input

$u \leftarrow \{0,1\}^n$

$\mathrm{ct}_{x,u} \leftarrow \mathsf{WE.Enc}_x(u)$

$R \leftarrow \{0,1\}^N$

$\widetilde{\mathrm{ct}}_{R,u} \leftarrow \widetilde{\mathsf{WE.Enc}}_R(u)$

$N \gg b$

$\mathrm{ct}_{x,u} \quad \widetilde{\mathrm{ct}}_{R,u} \quad R$

$\tilde{u} := \mathsf{WE.Dec}(\mathrm{ct}_{x,u}, w)$

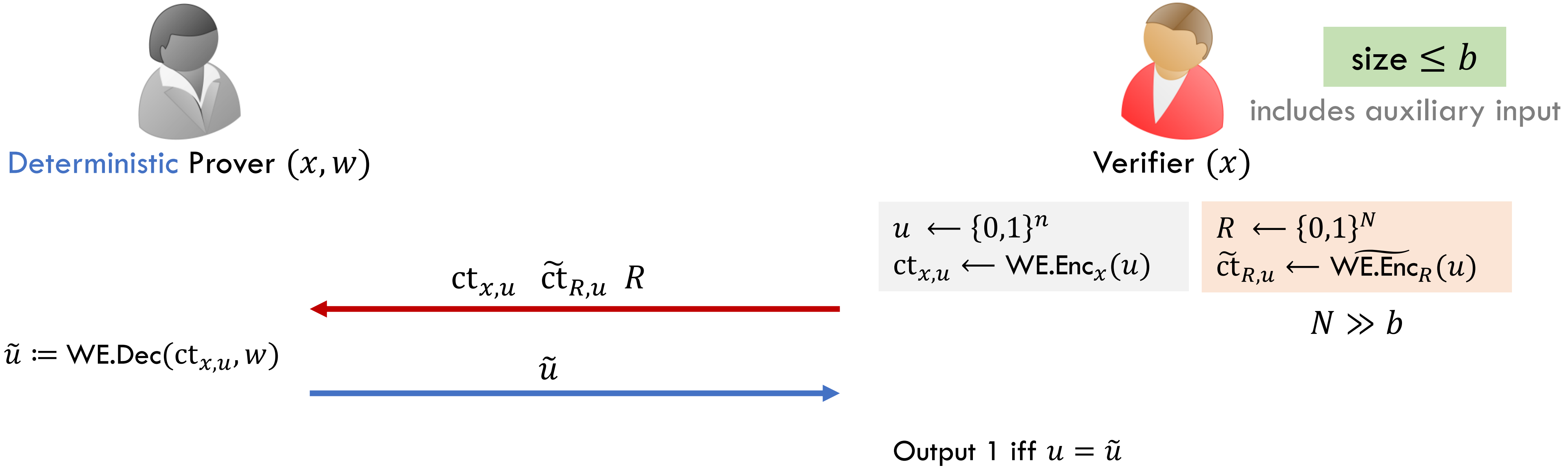
$\tilde{u}$

Output 1 iff $u = \tilde{u}$

$(R, \mathsf{M}) \in \mathsf{Rel}_{\tilde{\mathcal{L}}}$ if

1) M is a Turing Machine that outputs R.
2) Size of M is $b + \lambda$

Explainable Verifier DPZK



Deterministic Prover (x, w)

Verifier (x)

size $\leq b$

includes auxiliary input

$u \leftarrow \{0,1\}^n$
$\mathrm{ct}_{x,u} \leftarrow \mathsf{WE.Enc}_x(u)$

$R \leftarrow \{0,1\}^N$
$\widetilde{\mathrm{ct}}_{R,u} \leftarrow \widetilde{\mathsf{WE.Enc}}_R(u)$

$N \gg b$

$\mathrm{ct}_{x,u} \quad \widetilde{\mathrm{ct}}_{R,u} \quad R$

$\tilde{u} := \mathsf{WE.Dec}(\mathrm{ct}_{x,u}, w)$

$\tilde{u}$

Output 1 iff $u = \tilde{u}$

Completeness: Same as HVZK

$(R, \mathsf{M}) \in \mathsf{Rel}_{\tilde{\mathcal{L}}}$ if

1) M is a Turing Machine that outputs R.
2) Size of M is $b + \lambda$

Explainable Verifier DPZK

Cheating Prover (x, w)

Verifier (x)

size $\leq b$

includes auxiliary input

$u \leftarrow \{0,1\}^n$
$\mathsf{ct}_{x,u} \leftarrow \mathsf{WE.Enc}_x(u)$

$R \leftarrow \{0,1\}^N$
$\widetilde{\mathsf{ct}}_{R,u} \leftarrow \widetilde{\mathsf{WE.Enc}}_R(u)$

$N \gg b$

$\mathsf{ct}_{x,u}$ $\widetilde{\mathsf{ct}}_{R,u}$ R

$\tilde{u} \coloneqq \mathsf{WE.Dec}(\mathsf{ct}_{x,u}, w)$

$\tilde{u}$

Output 1 iff $u = \tilde{u}$

Completeness

Soundness: w.h.p. no short machine exists to satisfy $\mathsf{Rel}_{\tilde{\mathcal{L}}}$

$(R, \mathsf{M}) \in \mathsf{Rel}_{\tilde{\mathcal{L}}}$ if

1) M is a Turing Machine that outputs R.
2) Size of M is $b + \lambda$

Explainable Verifier DPZK

Simulator (x)

Verifier (x)

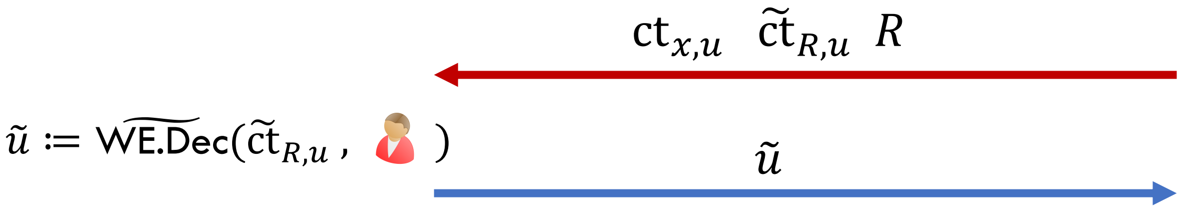
size $\leq b$

includes auxiliary input

$u \longleftarrow \{0,1\}^n$
$\mathsf{ct}_{x,u} \longleftarrow \mathsf{WE.Enc}_x(u)$

$R \longleftarrow \{0,1\}^N$
$\widetilde{\mathsf{ct}}_{R,u} \longleftarrow \widetilde{\mathsf{WE.Enc}}_R(u)$

$N \gg b$



$\mathsf{ct}_{x,u}$ $\widetilde{\mathsf{ct}}_{R,u}$ R

$\tilde{u} \coloneqq \widetilde{\mathsf{WE.Dec}}(\widetilde{\mathsf{ct}}_{R,u},$)

$\tilde{u}$

Output 1 iff $u = \tilde{u}$

Completeness

Soundness

Zero Knowledge: Simulator uses the verifier's code as witness; verifier's randomness simulated by a PRG.

$(R, \mathrm{M}) \in \mathsf{Rel}_{\tilde{\mathcal{L}}}$ if
1) M is a Turing Machine that outputs R.
2) Size of M is $b + \lambda$

Explainable Verifier DPZK

$\mathsf{Rel}_{\tilde{\mathcal{L}}}$ is not an NP relation since we do not a priori bound the running time of M.

Efficient Witness Encryption for $\mathsf{Rel}_{\tilde{\mathcal{L}}}$ can be realized assuming indistinguishability obfuscation for Turing Machines.

size $\leq b$

includes auxiliary input

Verifier (x)

$u \leftarrow \{0,1\}^n$
$\mathsf{ct}_{x,u} \leftarrow \mathsf{WE.Enc}_x(u)$

$R \leftarrow \{0,1\}^N$
$\widetilde{\mathsf{ct}}_{R,u} \leftarrow \widetilde{\mathsf{WE.Enc}}_R(u)$

$N \gg b$

Output 1 iff $u = \tilde{u}$

$(R, \mathsf{M}) \in \mathsf{Rel}_{\tilde{\mathcal{L}}}$ if
1) M is a Turing Machine that outputs R.
2) Size of M is $b + \lambda$

Malicious Verifier DPZK

Explainable Verifier DPZK

Malicious Verifier DPZK

Verifier proves honest behavior

Malicious Verifier DPZK

Explainable Verifier DPZK

Malicious Verifier DPZK

Verifier proves honest behavior

$\mathcal{L} \in \mathrm{NP} \cap \mathrm{coNP}$

Verifier proves via NIWI that

1. It behaved honestly; OR
2. $x \notin \mathcal{L}$

Malicious Verifier DPZK

Explainable Verifier DPZK

Malicious Verifier DPZK

Verifier proves honest behavior

$\mathcal{L} \in \mathrm{NP} \cap \mathrm{coNP}$

Verifier proves via NIWI that

1. It behaved honestly; OR
2. $x \notin \mathcal{L}$

$\mathcal{L} \in \mathrm{NP}$

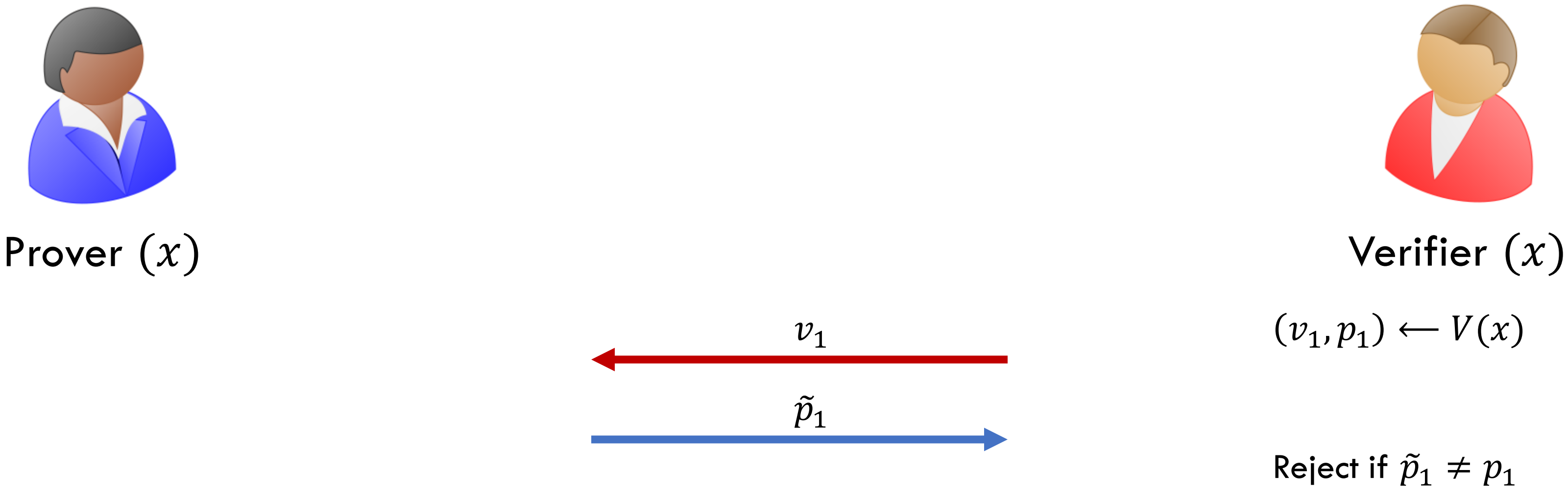
Verifier proves via NIWI that

1. It behaved honestly; OR
2. It has committed to a collision of keyless CRHF.

Necessity of Witness Encryption for DPZK

Predictable Arguments (PA)

[Faonio-Nielsen-Venturi'17]



Prover (x)

Verifier (x)

$(v_1, p_1) \longleftarrow V(x)$

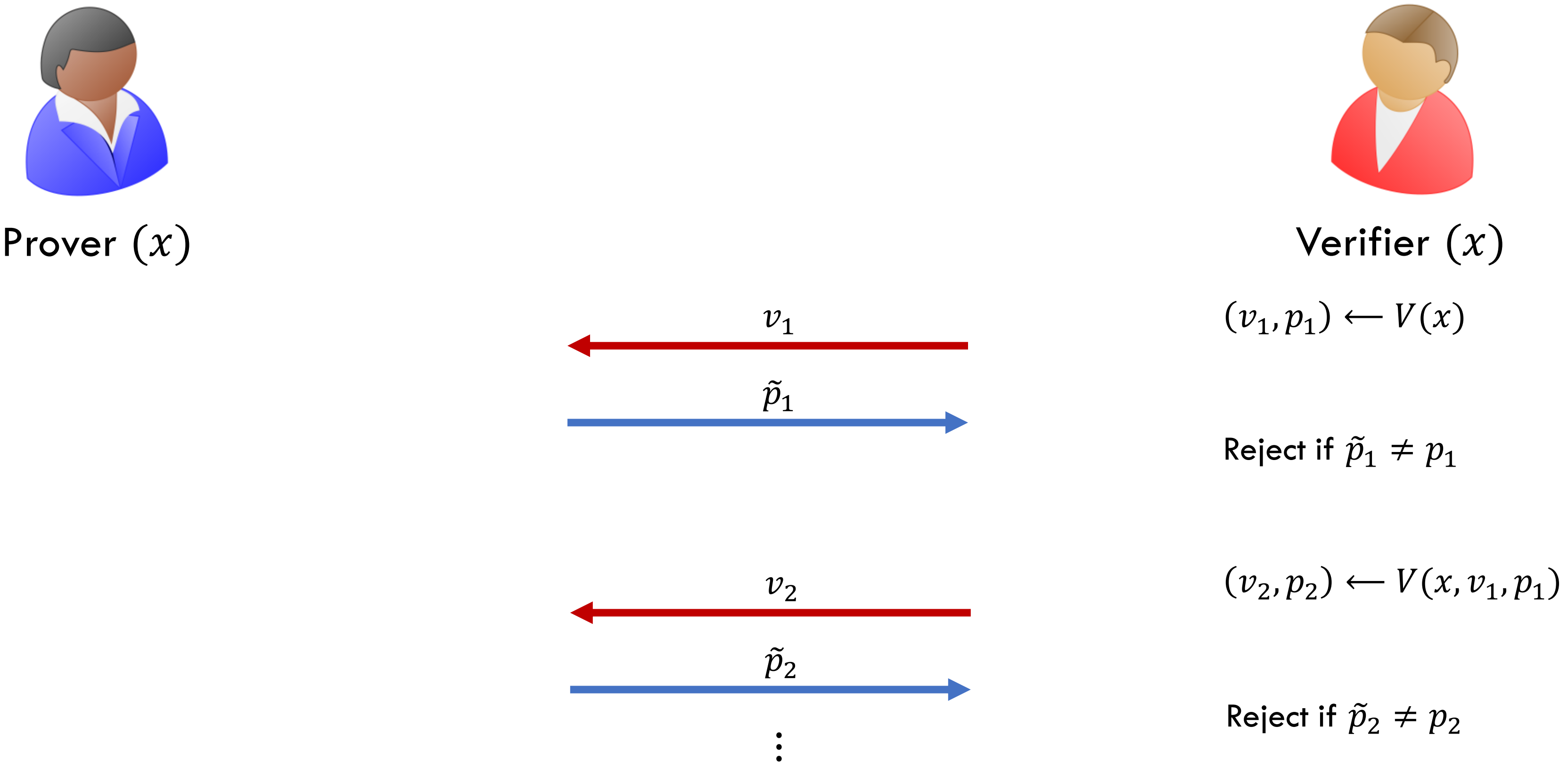
$$\xleftarrow{\quad v_1 \quad}$$

$$\xrightarrow{\quad \tilde{p}_1 \quad}$$

Reject if $\tilde{p}_1 \neq p_1$

Predictable Arguments (PA)

[Faonio-Nielsen-Venturi'17]



Prover (x)

Verifier (x)

$(v_1, p_1) \longleftarrow V(x)$

v_1

$\tilde{p}_1$

Reject if $\tilde{p}_1 \neq p_1$

$(v_2, p_2) \longleftarrow V(x, v_1, p_1)$

v_2

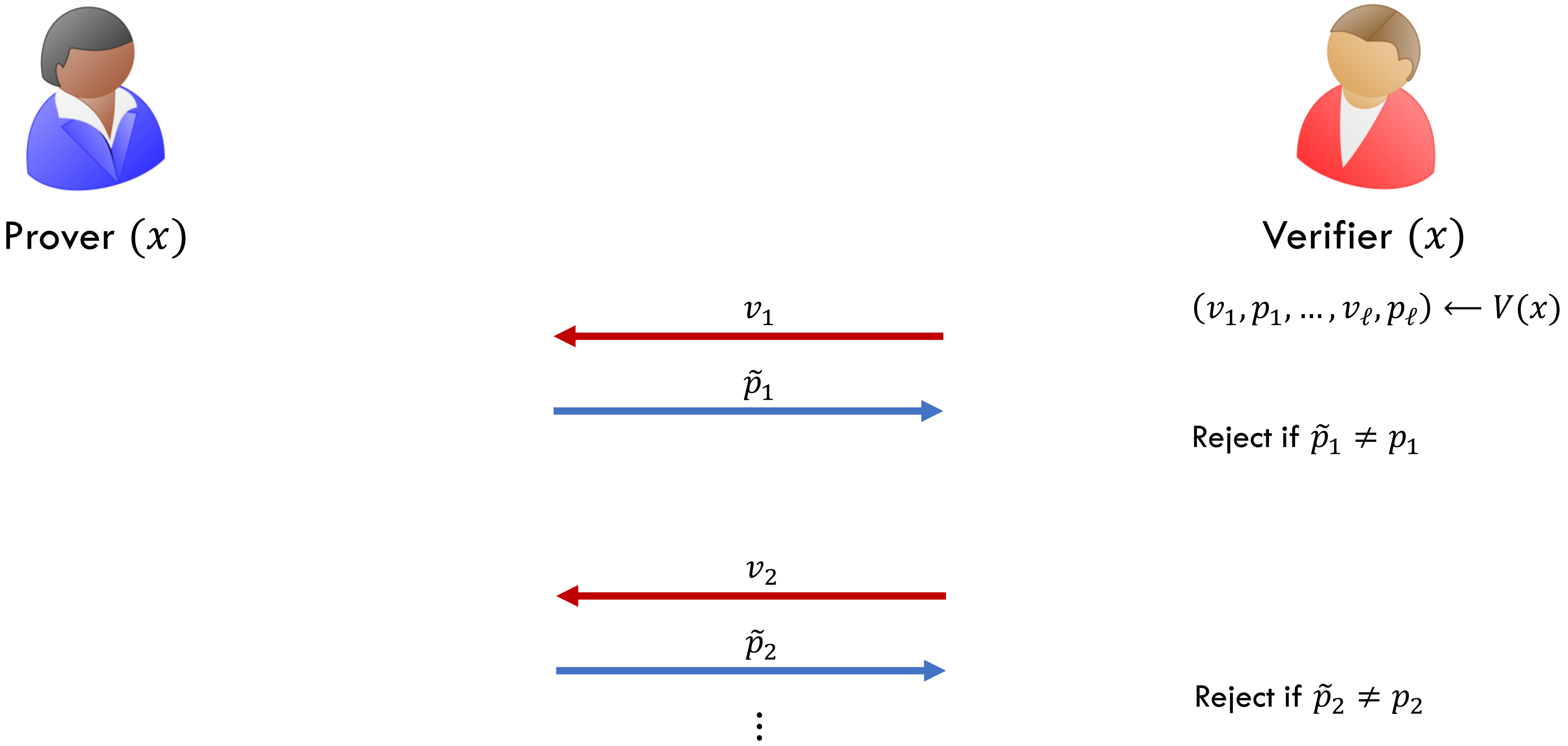
$\tilde{p}_2$

Reject if $\tilde{p}_2 \neq p_2$

$\vdots$

Predictable Arguments (PA)

[Faonio-Nielsen-Venturi'17]



Prover (x)

Verifier (x)

$(v_1, p_1, \dots, v_\ell, p_\ell) \longleftarrow V(x)$

v_1

$\tilde{p}_1$

Reject if $\tilde{p}_1 \neq p_1$

v_2

$\tilde{p}_2$

Reject if $\tilde{p}_2 \neq p_2$

$\vdots$

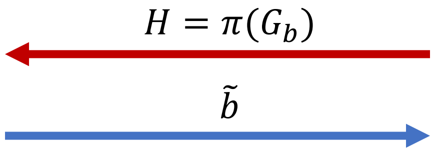
Predictable Arguments (PA)

[Faonio-Nielsen-Venturi'17]

Prover (G_0, G_1)

Verifier (G_0, G_1)

$b \longleftarrow \{0,1\}, \pi \longleftarrow \Pi_n$



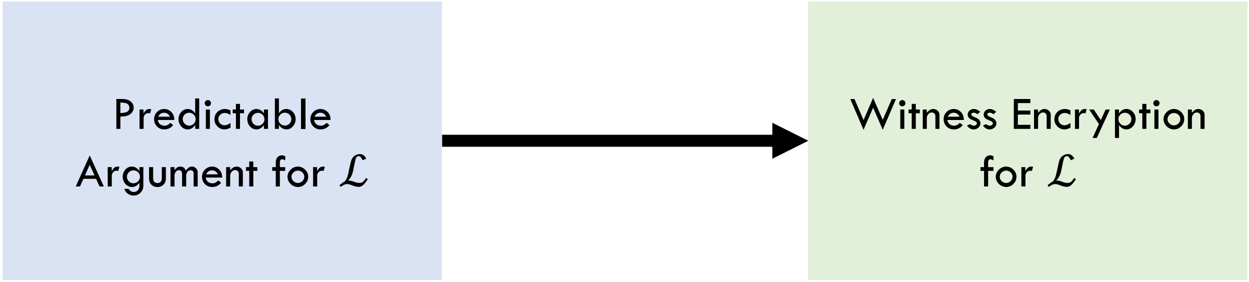
$H = \pi(G_b)$

$\tilde{b}$

Reject if $\tilde{b} \neq b$

Predictable argument for Graph Non-Isomorphism

DPZK to WE

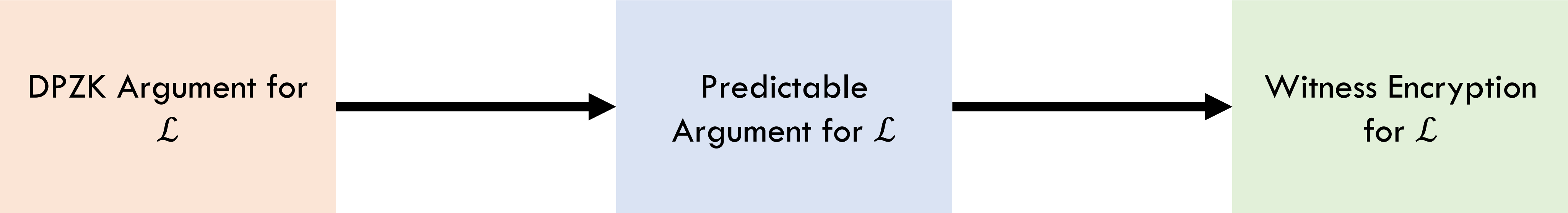


Predictable Argument for $\mathcal{L}$

Witness Encryption for $\mathcal{L}$

[Faonio-Nielsen-Venturi'17]

DPZK to WE



DPZK Argument for $\mathcal{L}$

Predictable Argument for $\mathcal{L}$

Witness Encryption for $\mathcal{L}$

[Faonio-Nielsen-Venturi’17]

DPZK to PA

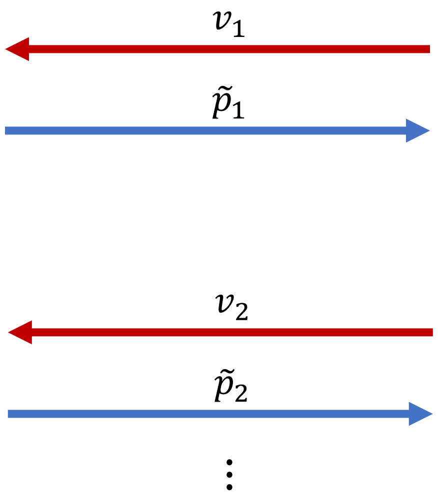
Prover (x, w)

Behaves identically to the DPZK prover

Verifier (x)

$(v_1, p_1, \dots, v_\ell, p_\ell; r) \longleftarrow$  (x)

Verifier rejects if HVZK simulator does not produce accepting transcript



Reject if $\tilde{p}_1 \neq p_1$

Reject if $\tilde{p}_2 \neq p_2$

DPZK to PA

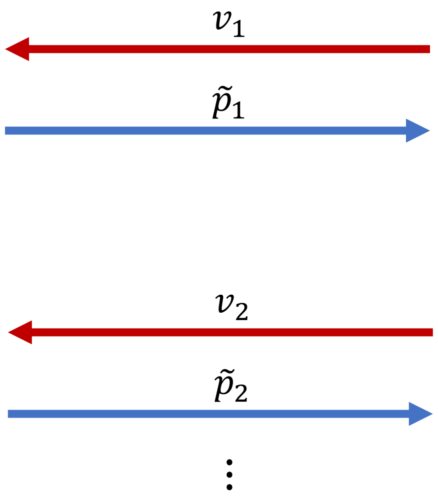
Prover (x, w)

Behaves identically to the DPZK prover

Verifier (x)

$(v_1, p_1, \dots, v_\ell, p_\ell; r) \longleftarrow (x)$

Verifier rejects if HVZK simulator does not produce accepting transcript



Reject if $\tilde{p}_1 \neq p_1$

Reject if $\tilde{p}_2 \neq p_2$

Completeness: From the ZK property, the simulator and (real) DPZK prover generate the same messages.

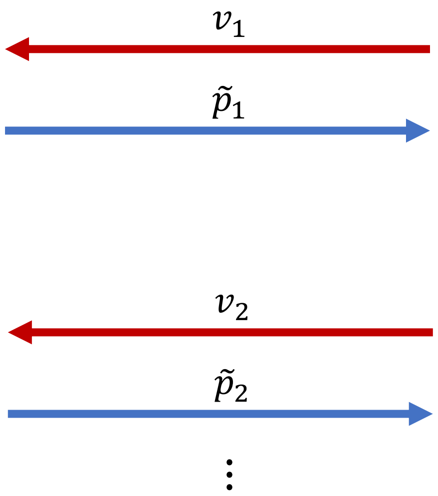
DPZK to PA

Cheating Prover (x)

Verifier (x)

$(v_1, p_1, \ldots, v_\ell, p_\ell; r) \longleftarrow (x)$

Verifier rejects if HVZK simulator does not produce accepting transcript



Reject if $\tilde{p}_1 \neq p_1$

Reject if $\tilde{p}_2 \neq p_2$

Completeness

Soundness: If the verifier does not reject, the cheating prover generates an accepting transcript when $x \notin \mathcal{L}$, breaking soundness of the ZK protocol.*

*implicitly assumed that simulated random coins are pseudorandom when $x \notin \mathcal{L}$.

Other Results

Any DPZK argument for bounded auxiliary input verifiers can be made two message, and laconic in the prover message.

Follows from the transformation on predictable arguments in [Faonio-Nielsen-Venturi'17]. We show that the transformations preserve zero-knowledge.

There exist two message DPZK arguments for all of NP against bounded auxiliary-input verifiers.

Any DPZK argument for a language $\mathcal{L}$ implies a witness encryption for $\mathcal{L}$.

There exist two message DPZK arguments for all of NP against bounded auxiliary-input verifiers.

Any DPZK argument for a language $\mathcal{L}$ implies a witness encryption for $\mathcal{L}$.

Thank you. Questions?

Arka Rai Choudhuri

achoud@cs.jhu.edu

ia.cr/2020/1160