Simulation extractability of zkSNARKs

or

What Makes Fiat–Shamir zkSNARKs (Updatable) Simulation Extractable?

Chaya Ganesh Hamidreza Khoshakhlagh Markulf Kohlweiss Anca Nitulescu **Michał Zając**

Agenda

Our **contribution**

NIZKs definitions

The **problem** with SRS

Notions of **soundness** and what they give

Simulation extractability

How to **build an extractor** for zkSNARKs?

Contribution

Contribution

Define **updatable simulation extractability** for NIZKs

Contribution

Define **updatable simulation extractability** for NIZKs

Analyze three properties **needed** for the NIZK to be simulation extractable

Contribution

Define **updatable simulation extractability** for NIZKs

Analyze three properties **needed** for the NIZK to be simulation extractable

Show that a wide class (Plonk, Marlin, Lunar, Sonic, Vampire) of **zkSNARKs** is simulation-extractable **out of the box**

no changes in the protocol needed $\mapsto$ **no efficiency loss**!

Contribution

Define **updatable simulation extractability** for NIZKs

Analyze three properties **needed** for the NIZK to be simulation extractable

Show that a wide class (Plonk, Marlin, Lunar, Sonic, Vampire) of **zkSNARKs** is simulation-extractable **out of the box**

no changes in the protocol needed $\mapsto$ **no efficiency loss**!

Snarky signature in the **updatable** setting

zkSNARKs properties

zkSNARKs properties

$$\mathcal{P}(srs, x, w) \xrightarrow{\pi} \mathcal{V}(srs, x)$$

$$R(x, w)$$

zkSNARKs properties

$$\mathscr{P}(srs, x, w) \xrightarrow{\pi} \mathscr{V}(srs, x)$$

$$R(x, w)$$

Completeness

Honest verifier always accepts a proof from an honest prover $\mathscr{P}$

zkSNARKs properties

$$\mathscr{P}(srs, x, w) \xrightarrow{\pi} \mathscr{V}(srs, x)$$

$$R(x, w)$$

Completeness
Honest verifier always accepts a proof from an honest prover $\mathscr{P}$

Knowledge soundness
If verifier $\mathscr{V}$ accepts a proof for a statement x then the prover $\mathscr{A}(p, srs; r)$ **knows** w such that $R(x, w)$

$$\mathscr{E}_{\mathscr{A}}(p, srs, r) = w$$

zkSNARKs properties

$$\mathcal{P}(srs, x, w) \xrightarrow{\pi} \mathcal{V}(srs, x)$$

$$R(x, w)$$

Completeness
Honest verifier always accepts a proof from an honest prover $\mathcal{P}$

Knowledge soundness
If verifier $\mathcal{V}$ accepts a proof for a statement x then the prover $\mathcal{A}(p, srs; r)$ **knows** w such that $R(x, w)$

$$\mathcal{E}_{\mathcal{A}}(p, srs, r) = w$$

Zero knowledge
Verifiers learns nothing besides the validity of the statement

$$\mathcal{S}(p, srs, td, x) \approx \mathcal{P}(p, srs, x, w)$$

zkSNARKs properties

$$\mathcal{P}(srs, x, w) \xrightarrow{\pi} \mathcal{V}(srs, x)$$

$$R(x, w)$$

Completeness
Honest verifier always accepts a proof from an honest prover $\mathcal{P}$

Knowledge soundness
If verifier $\mathcal{V}$ accepts a proof for a statement x then the prover $\mathcal{A}(p, srs; r)$ **knows** w such that $R(x, w)$

$$\mathcal{E}_{\mathcal{A}}(p, srs, r) = w$$

Zero knowledge
Verifiers learns nothing besides the validity of the statement

$$\mathcal{S}(p, srs, td, x) \approx \mathcal{P}(p, srs, x, w)$$

Succinctness
The proof π is short

$$|\pi| = O(\log(|x| + |w|))$$

All the fuss with the SRS

$$td, srs \leftarrow \mathsf{KGen}(p)$$

All the fuss with the SRS

$$td, srs \leftarrow \mathsf{KGen}(p)$$

Trusted party
Party that creates the SRS can easily provide valid proofs for false statements

$$\mathcal{S}(p, srs, td, x)$$

All the fuss with the SRS

$$td, srs \leftarrow \mathsf{KGen}(p)$$

Trusted party
Party that creates the SRS can easily provide valid proofs for false statements

$$\mathcal{S}(p, srs, td, x)$$

Real life instantiation
MPC ceremonies to generate the SRS

All the fuss with the SRS

$$td, srs \leftarrow \mathsf{KGen}(p)$$

Trusted party
Party that creates the SRS can easily provide valid proofs for false statements

$$\mathcal{S}(p, srs, td, x)$$

Real life instantiation
MPC ceremonies to generate the SRS

Updatable SRS
Everybody can take an existing SRS srs and **update** it to get an SRS srs'
One honest update is enough for the proof system to be sound

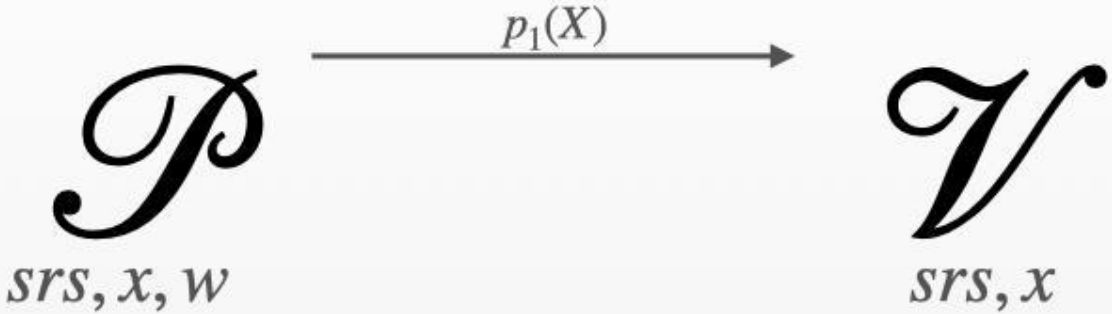
$$\mathsf{Update}(srs) \rightarrow srs', \pi_{upd}$$

How are zkSNARK-s built?

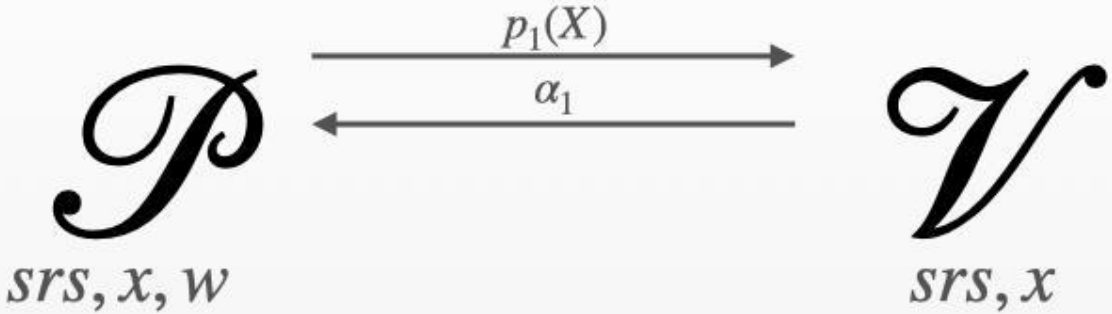
srs, x, w

srs, x

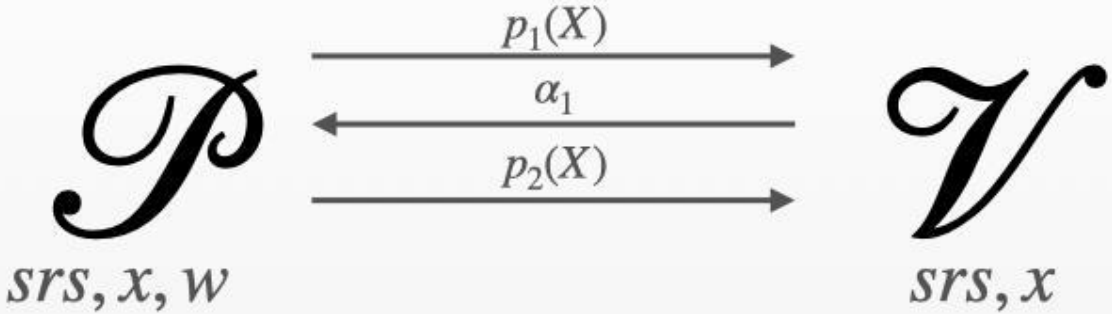
How are zkSNARK-s built?



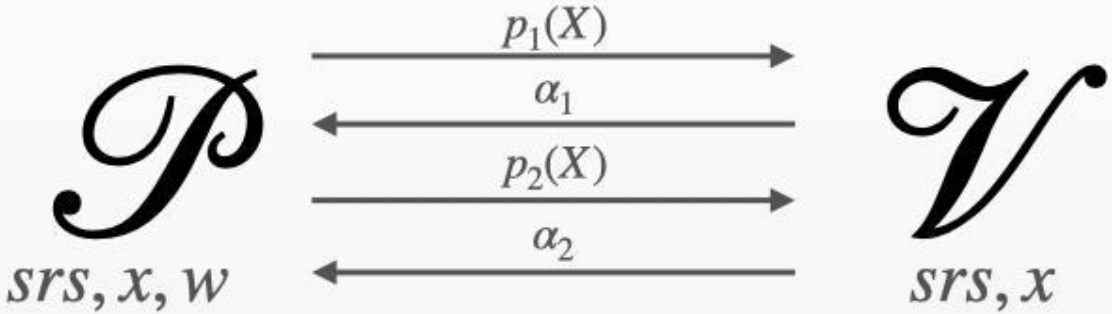
How are zkSNARK-s built?



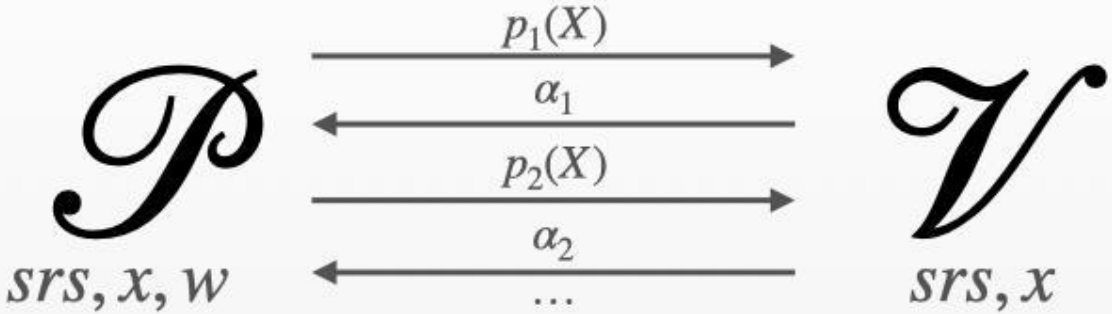
How are zkSNARK-s built?



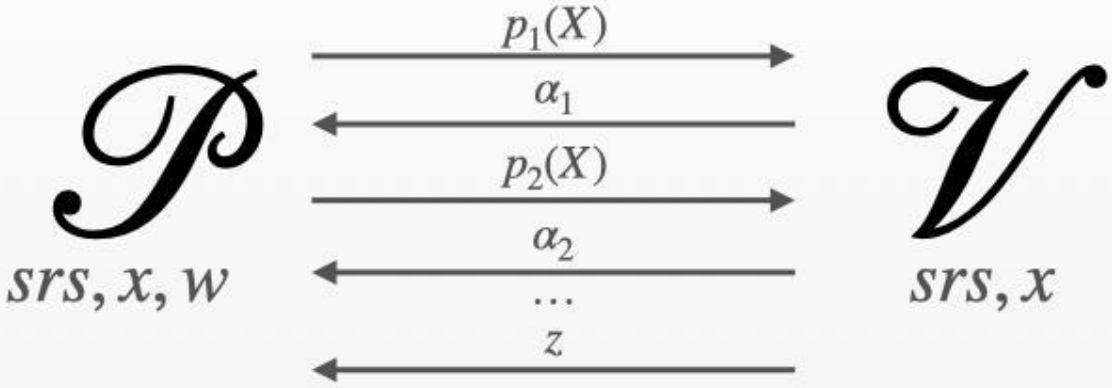
How are zkSNARK-s built?



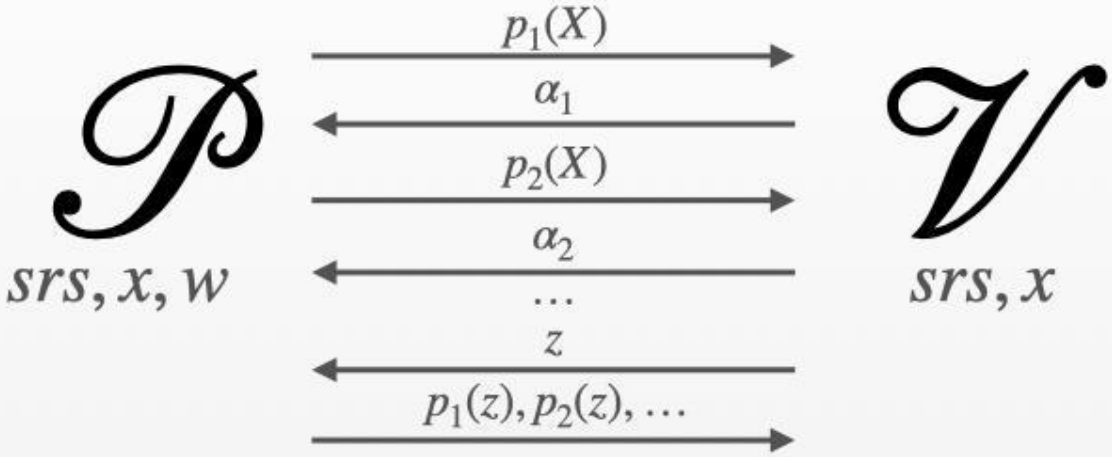
How are zkSNARK-s built?



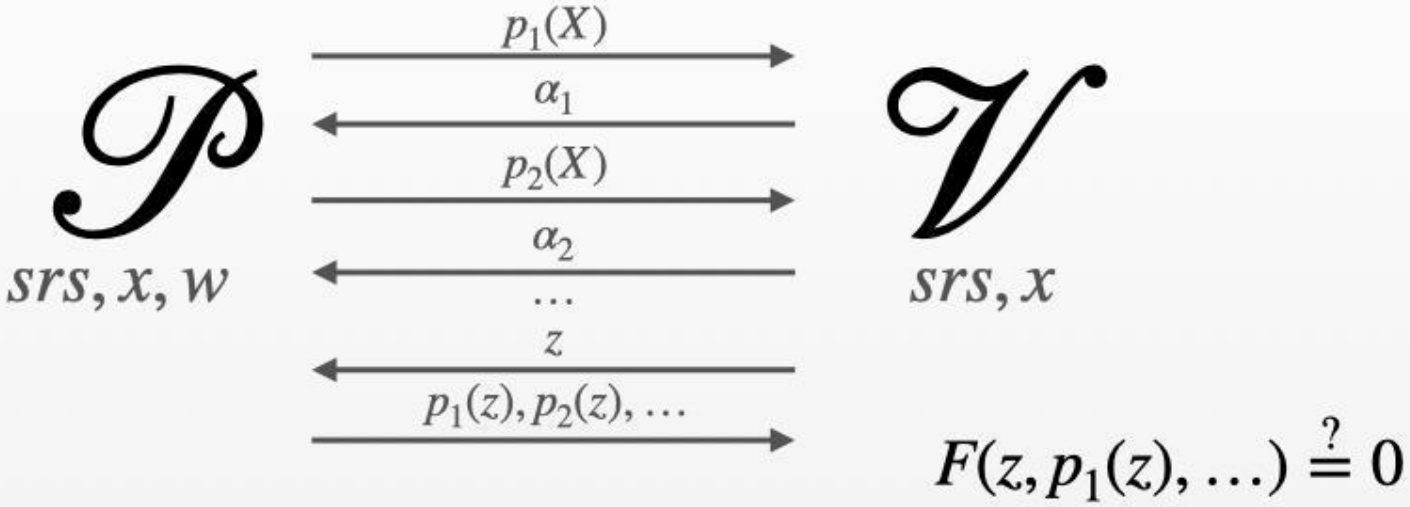
How are zkSNARK-s built?



How are zkSNARK-s built?



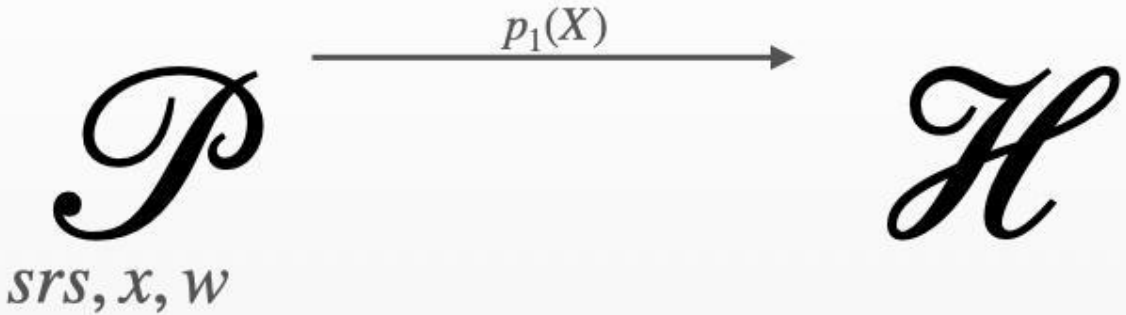
How are zkSNARK-s built?



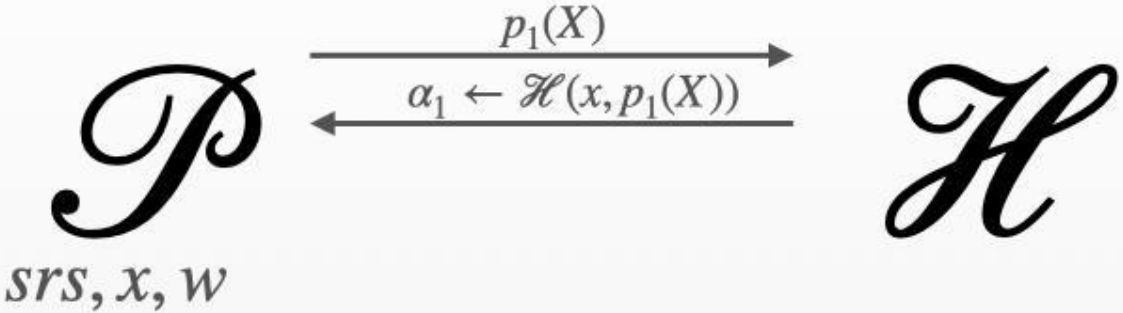
From interactive to non interactive: Fiat–Shamir transformation

srs, x, w

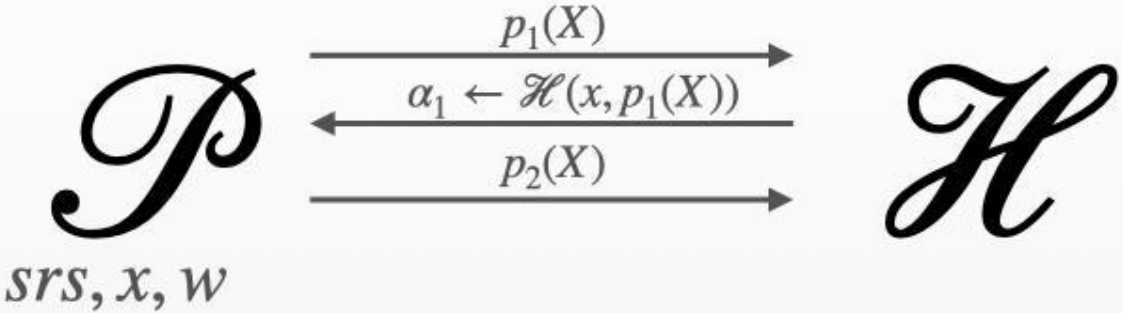
From interactive to non interactive: Fiat–Shamir transformation



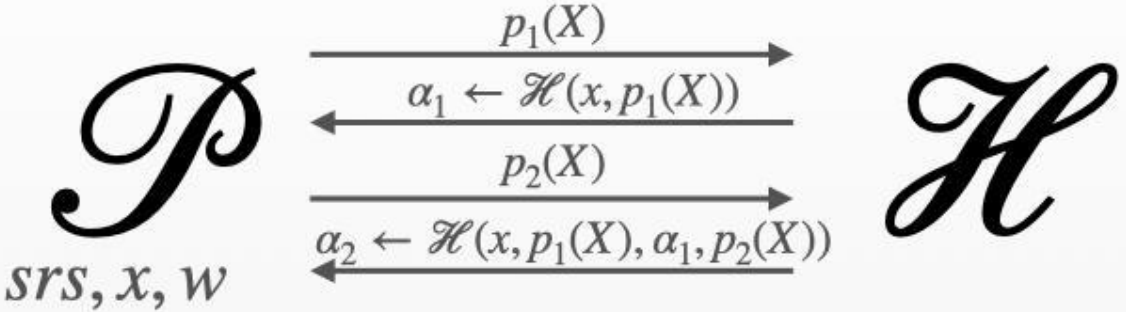
From interactive to non interactive: Fiat–Shamir transformation



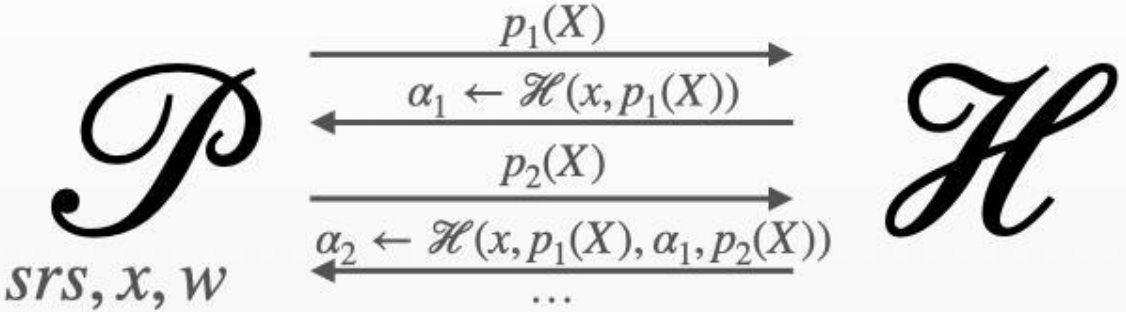
From interactive to non interactive: Fiat–Shamir transformation



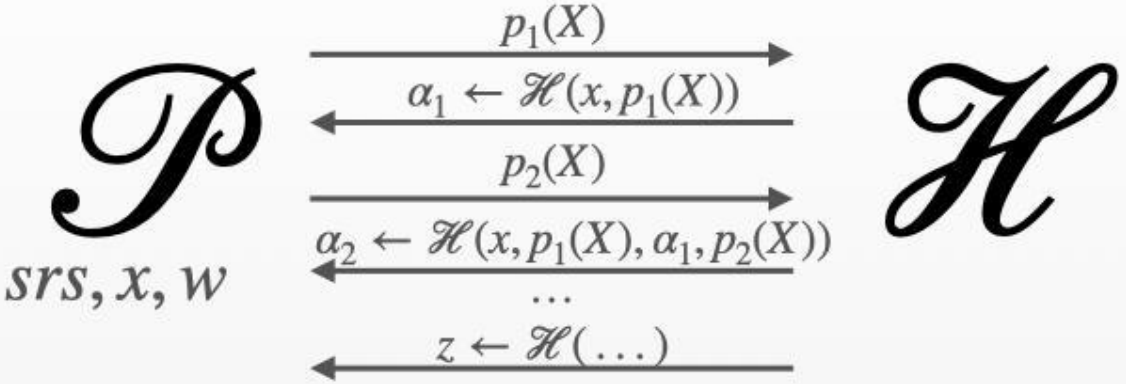
From interactive to non interactive: Fiat–Shamir transformation



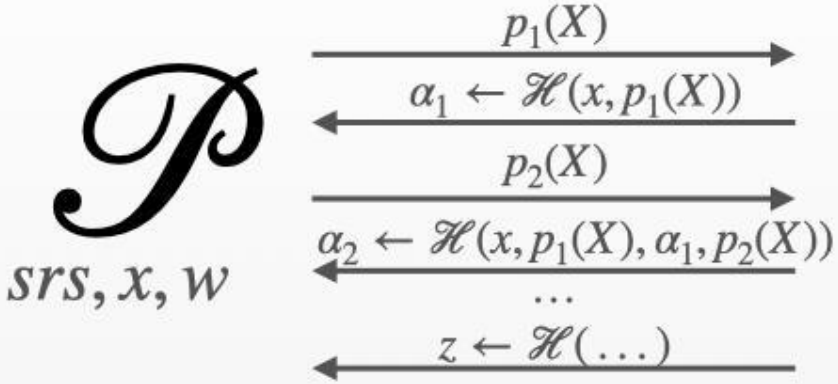
From interactive to non interactive: Fiat–Shamir transformation

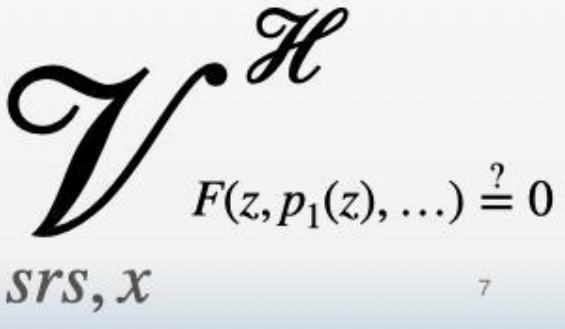


From interactive to non interactive: Fiat–Shamir transformation

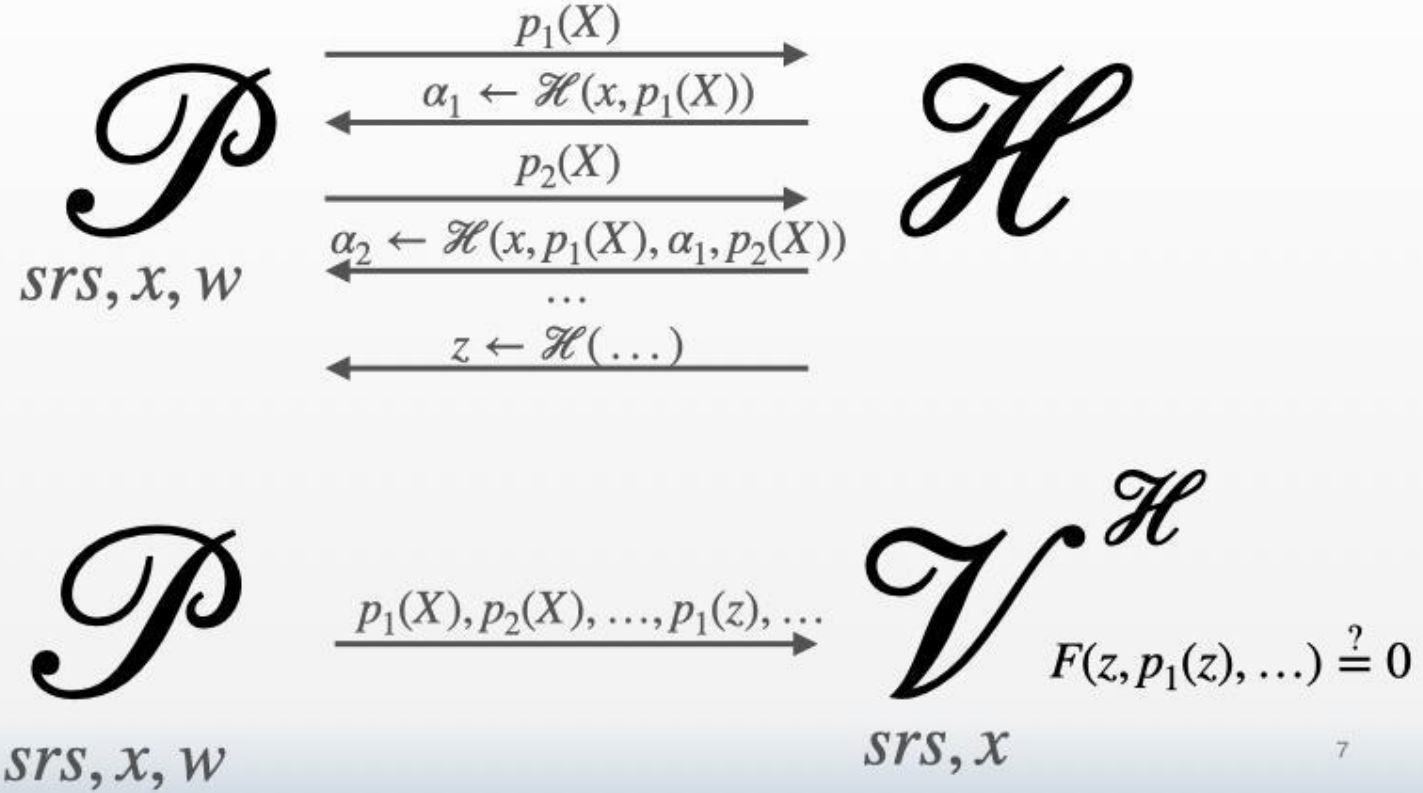


From interactive to non interactive: Fiat–Shamir transformation





From interactive to non interactive: Fiat–Shamir transformation



The importance of non-malleability

The importance of non-malleability

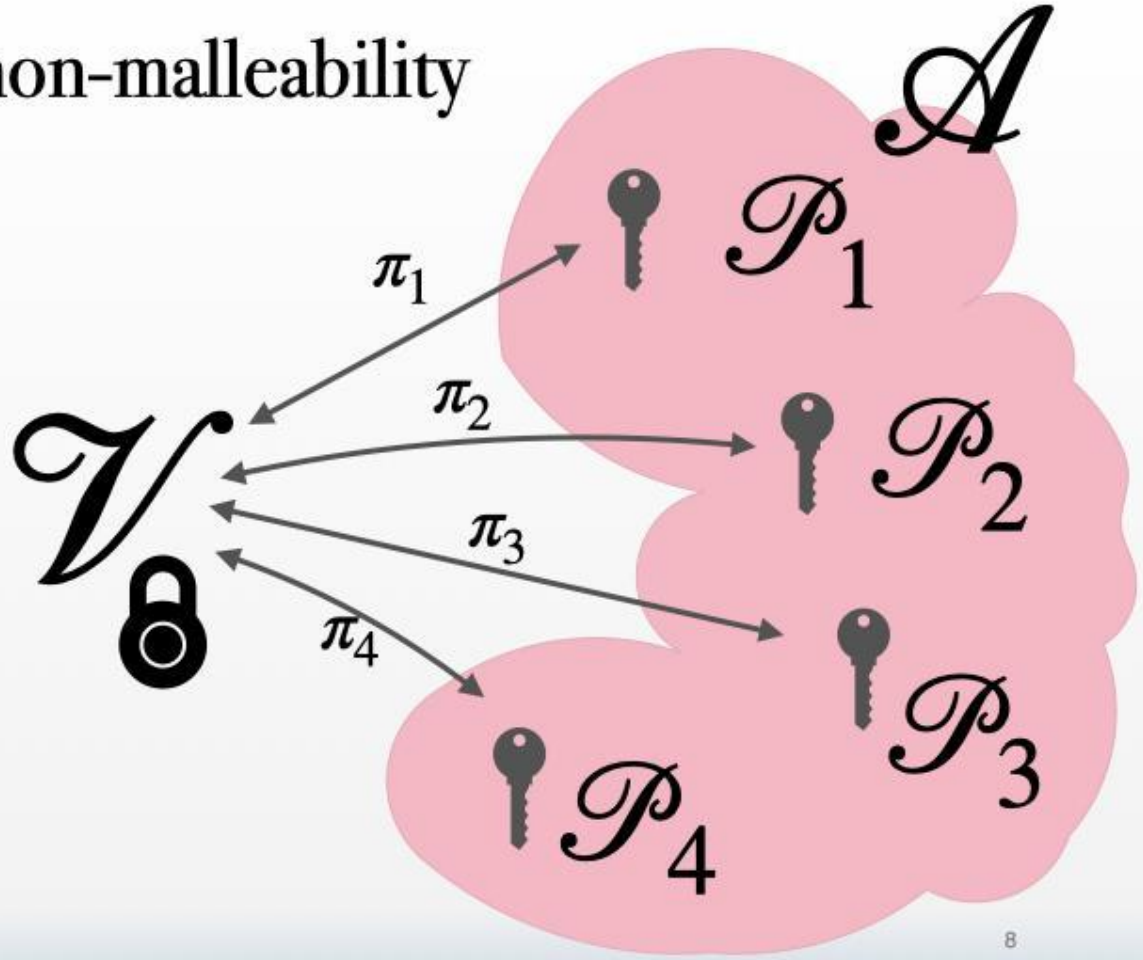
Knowledge soundness: $\mathcal{A}^{\mathcal{H}}(p, srs; r)$
cannot convince the verifier $\mathcal{V}$ on x
unless it knows w such that $R(x, w)$

Extraction: $\mathcal{E}_{\mathcal{A}}(p, srs, r, Q_{\mathcal{H}}) = w$

The importance of non-malleability

Knowledge soundness: $\mathcal{A}^{\mathcal{H}}(p, srs; r)$ cannot convince the verifier $\mathcal{V}$ on x unless it knows w such that $R(x, w)$

Extraction: $\mathcal{E}_{\mathcal{A}}(p, srs, r, Q_{\mathcal{H}}) = w$



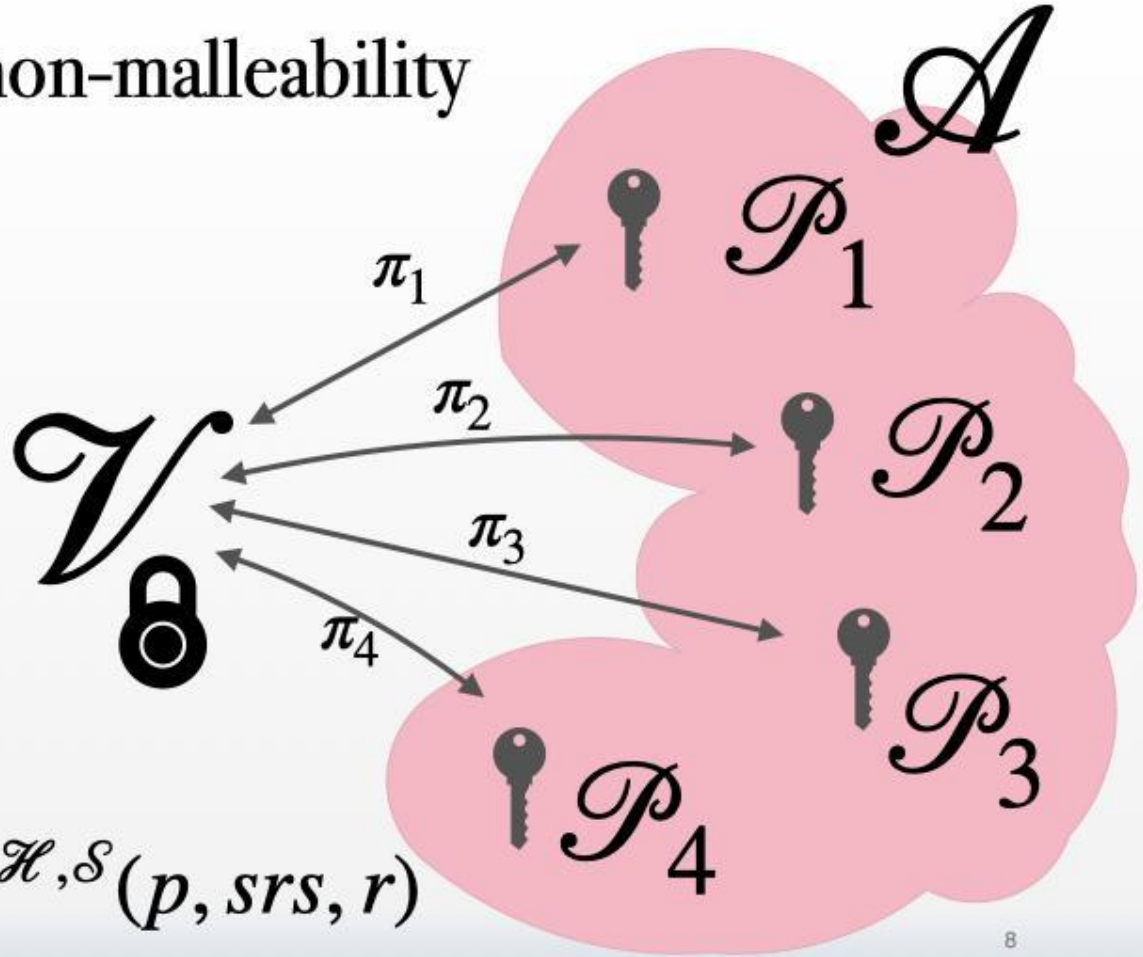
The importance of non-malleability

Knowledge soundness: $\mathcal{A}^{\mathcal{H}}(p, srs; r)$ cannot convince the verifier $\mathcal{V}$ on x unless it knows w such that $R(x, w)$

Extraction: $\mathcal{E}_{\mathcal{A}}(p, srs, r, Q_{\mathcal{H}}) = w$

Simulation-extractability: $\mathcal{A}^{\mathcal{H},\mathcal{S}}(p, srs; r)$ cannot convince the verifier $\mathcal{V}$ on x unless it knows w such that $R(x, w)$ **given access to simulated proofs**

Extraction: $\mathcal{E}_{\mathcal{A}}(p, srs, r, Q_{\mathcal{H}}, Q_{\mathcal{S}}) = w$



$$\mathcal{A}^{\mathcal{H}}(p, srs, r) \mapsto \mathcal{A}^{\mathcal{H},\mathcal{S}}(p, srs, r)$$

Simulation-extractability

$$\mathcal{A}^{\mathcal{H},\mathcal{S}}(p, srs, r) = x, \pi$$

Simulation-extractability

$$\mathcal{A}^{\mathcal{H},\mathcal{S}}(p, srs, r) = x, \pi$$

$\mathcal{S}(x)$ responds with a simulated proof for x

Simulation-extractability

$$\mathcal{A}^{\mathcal{H},\mathcal{S}}(p, srs, r) = x, \pi$$

$\mathcal{S}(x)$ responds with a simulated proof for x

When adversary wins?
$\mathcal{A}$ returns an instance—proof pair (x, π) such that
- π is not a simulated proof
- $\mathcal{A}$ doesn't know the witness

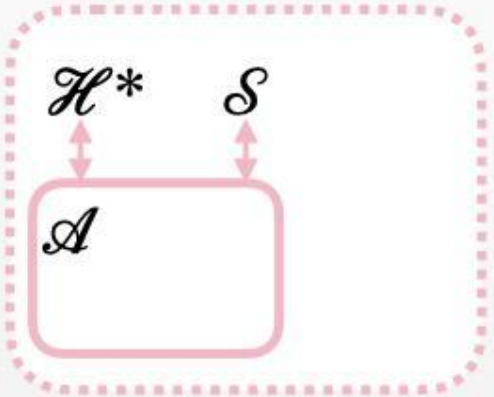
Simulation-extractability

$$\mathcal{A}^{\mathcal{H},\mathcal{S}}(p, srs, r) = x, \pi$$

$\mathcal{S}(x)$ responds with a simulated proof for x

When adversary wins?
$\mathcal{A}$ returns an instance—proof pair (x, π) such that
- π is not a simulated proof
- $\mathcal{A}$ doesn't know the witness

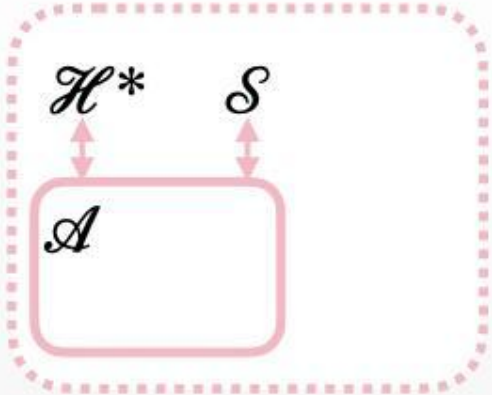
$$\mathcal{E}_{\mathcal{A}}(p, srs, r, Q_{\mathcal{H}}, Q_{\mathcal{S}}) = w \qquad \mathcal{H}$$



Some issues with extraction

$\mathcal{E}_{\mathcal{A}}(p, srs, r, Q_{\mathcal{H}}, Q_{\mathcal{S}}) = w$ $\mathcal{H}$

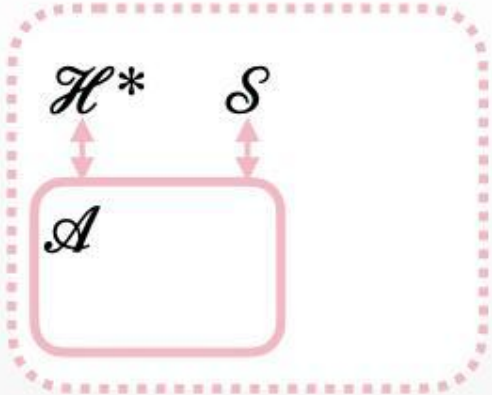
$\mathcal{A}^{\mathcal{H},\mathcal{S}}(p, srs, r) = x, \pi$



Some issues with extraction

$\mathcal{E}_{\mathcal{A}}(p, srs, r, Q_{\mathcal{H}}, Q_{\mathcal{S}}) = w$ $\mathcal{H}$

$\mathcal{A}^{\mathcal{H},\mathcal{S}}(p, srs, r) = x, \pi$

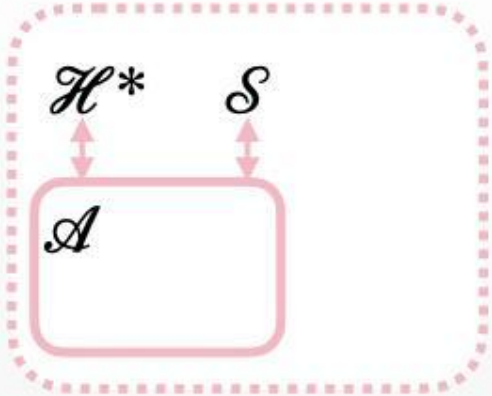


QUESTION 1
What if $\mathcal{A}$ outputs a re-randomized simulated proof?

Some issues with extraction

$\mathcal{E}_{\mathcal{A}}(p, srs, r, Q_{\mathcal{H}}, Q_{\mathcal{S}}) = w$ $\mathcal{H}$

$\mathcal{A}^{\mathcal{H},\mathcal{S}}(p, srs, r) = x, \pi$



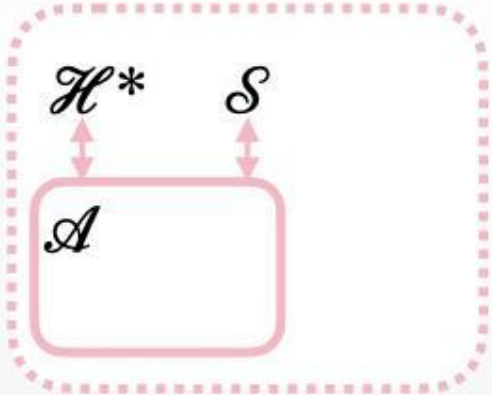
QUESTION 1
What if $\mathcal{A}$ outputs a re-randomized simulated proof?

REQUIREMENT 1
Unique response property: It is infeasible to provide two transcripts that
- are different, and
- match on the first k messages, and
- are both acceptable

Some issues with extraction

$\mathscr{E}_{\mathscr{A}}(p, srs, r, Q_{\mathscr{H}}, Q_{\mathcal{S}}) = w$ $\mathscr{H}$

$\mathscr{A}^{\mathscr{H},\mathcal{S}}(p, srs, r) = x, \pi$



QUESTION 1
What if $\mathscr{A}$ outputs a re-randomized simulated proof?

REQUIREMENT 1
Unique response property: It is infeasible to provide two transcripts that
- are different, and
- match on the first k messages, and
- are both acceptable

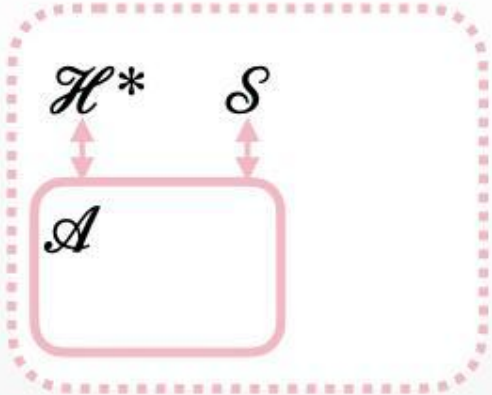
QUESTION 2
How can $\mathscr{E}$ provide simulated proofs?

Some issues with extraction

$\mathscr{E}_{\mathscr{A}}(p, srs, r, Q_{\mathscr{H}}, Q_{\mathcal{S}}) = w$

$\mathscr{H}$

$\mathscr{A}^{\mathscr{H},\mathcal{S}}(p, srs, r) = x, \pi$



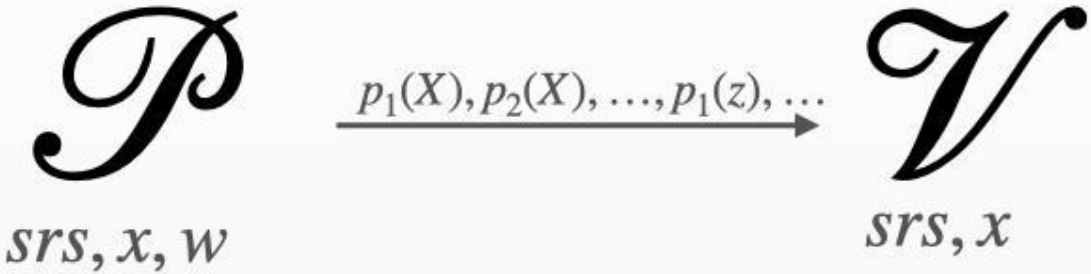
QUESTION 1
What if $\mathscr{A}$ outputs a re-randomized simulated proof?

QUESTION 2
How can $\mathscr{E}$ provide simulated proofs?

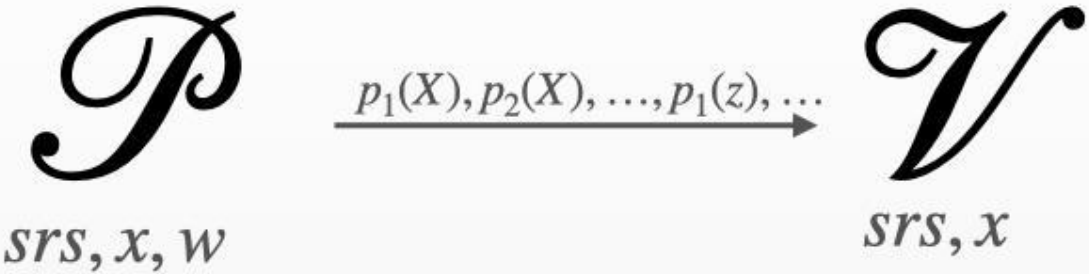
REQUIREMENT 1
Unique response property: It is infeasible to provide two transcripts that
- are different, and
- match on the first k messages, and
- are both acceptable

REQUIREMENT 2
Trapdoor-less zero-knowledge: there exists a simulator that provides valid proofs without using SRS's trapdoor

How are zkSNARK-s built?



How are zkSNARK-s built?



Plonk:

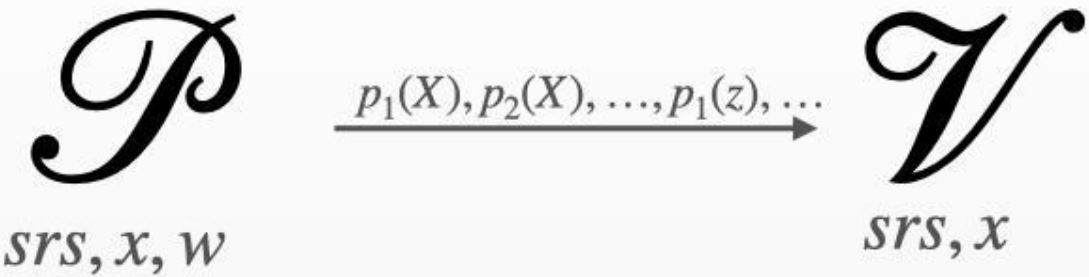
$w = (w_1, \ldots, w_{3n})$

$p_1(X) = w_1 + w_2X + \ldots w_nX^{n-1} + \ldots$

$p_2(X) = w_{n+1} + w_{n+2}X + \ldots w_{2n}X^{n-1} + \ldots$

$p_3(X) = w_{2n+1} + w_{2n+2}X + \ldots w_{3n}X^{n-1} + \ldots$

How are zkSNARK-s built?



Plonk:

$w = (w_1, \ldots, w_{3n})$

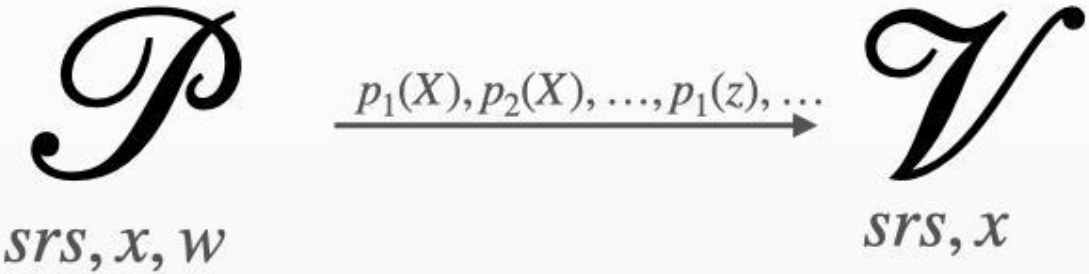
$p_1(X) = w_1 + w_2 X + \ldots w_n X^{n-1} + \ldots$

$p_2(X) = w_{n+1} + w_{n+2} X + \ldots w_{2n} X^{n-1} + \ldots$

$p_3(X) = w_{2n+1} + w_{2n+2} X + \ldots w_{3n} X^{n-1} + \ldots$

Observation:
Some of the polynomials $p_i(X)$ **encode** the witness

How are zkSNARK-s built?



Plonk:

$w = (w_1, \ldots, w_{3n})$

$p_1(X) = w_1 + w_2 X + \ldots w_n X^{n-1} + \ldots$

$p_2(X) = w_{n+1} + w_{n+2} X + \ldots w_{2n} X^{n-1} + \ldots$

$p_3(X) = w_{2n+1} + w_{2n+2} X + \ldots w_{3n} X^{n-1} + \ldots$

Observation:
Some of the polynomials $p_i(X)$ **encode** the witness

$\text{Decode}(p_1(X), \ldots) = w$

How to extract?

Plonk:

$w = (w_1, \ldots, w_{3n})$

$p_1(X) = w_1 + w_2 X + \ldots w_n X^{n-1} + \ldots$

$p_2(X) = w_{n+1} + w_{n+2} X + \ldots w_{2n} X^{n-1} + \ldots$

$p_3(X) = w_{2n+1} + w_{2n+2} X + \ldots w_{3n} X^{n-1} + \ldots$

How to extract?

Plonk:

$w = (w_1, \ldots, w_{3n})$

$p_1(X) = w_1 + w_2 X + \ldots w_n X^{n-1} + \ldots$

$p_2(X) = w_{n+1} + w_{n+2} X + \ldots w_{2n} X^{n-1} + \ldots$

$p_3(X) = w_{2n+1} + w_{2n+2} X + \ldots w_{3n} X^{n-1} + \ldots$

Observation:

$\deg(p_1) = \deg(p_2) = \deg(p_3) = n + 2$

If we evaluate $p_1(X), p_2(X), p_3(X)$ at $n + 3$ points then we can reveal their coefficients

How to extract?

Plonk:

$w = (w_1, \ldots, w_{3n})$

$p_1(X) = w_1 + w_2 X + \ldots w_n X^{n-1} + \ldots$

$p_2(X) = w_{n+1} + w_{n+2} X + \ldots w_{2n} X^{n-1} + \ldots$

$p_3(X) = w_{2n+1} + w_{2n+2} X + \ldots w_{3n} X^{n-1} + \ldots$

Observation:

$\deg(p_1) = \deg(p_2) = \deg(p_3) = n + 2$

If we evaluate $p_1(X), p_2(X), p_3(X)$ at $n + 3$ points then we can reveal their coefficients

Idea:
Let the extractor learn **multiple evaluations** of polynomials

How to extract?

Plonk:

$w = (w_1, \ldots, w_{3n})$

$p_1(X) = w_1 + w_2 X + \ldots w_n X^{n-1} + \ldots$

$p_2(X) = w_{n+1} + w_{n+2} X + \ldots w_{2n} X^{n-1} + \ldots$

$p_3(X) = w_{2n+1} + w_{2n+2} X + \ldots w_{3n} X^{n-1} + \ldots$

Observation:

$\deg(p_1) = \deg(p_2) = \deg(p_3) = n + 2$

If we evaluate $p_1(X), p_2(X), p_3(X)$ at $n + 3$ points then we can reveal their coefficients

Idea:
Let the extractor learn **multiple evaluations** of polynomials

$$(z_1, p_1(z_1)), \ldots, (z_{n+3}, p_1(z_{n+3}))$$

How to extract?

Plonk:

$w = (w_1, \ldots, w_{3n})$

$p_1(X) = w_1 + w_2 X + \ldots w_n X^{n-1} + \ldots$

$p_2(X) = w_{n+1} + w_{n+2} X + \ldots w_{2n} X^{n-1} + \ldots$

$p_3(X) = w_{2n+1} + w_{2n+2} X + \ldots w_{3n} X^{n-1} + \ldots$

Observation:

$\deg(p_1) = \deg(p_2) = \deg(p_3) = n + 2$

If we evaluate $p_1(X), p_2(X), p_3(X)$ at $n + 3$ points then we can reveal their coefficients

Idea:
Let the extractor learn **multiple evaluations** of polynomials

$$(z_1, p_1(z_1)), \ldots, (z_{n+3}, p_1(z_{n+3}))$$

Lagrange **interpolation**

How to extract?

Plonk:

$w = (w_1, \ldots, w_{3n})$

$p_1(X) = w_1 + w_2 X + \ldots w_n X^{n-1} + \ldots$

$p_2(X) = w_{n+1} + w_{n+2} X + \ldots w_{2n} X^{n-1} + \ldots$

$p_3(X) = w_{2n+1} + w_{2n+2} X + \ldots w_{3n} X^{n-1} + \ldots$

Observation:

$\deg(p_1) = \deg(p_2) = \deg(p_3) = n + 2$

If we evaluate $p_1(X), p_2(X), p_3(X)$ at $n + 3$ points then we can reveal their coefficients

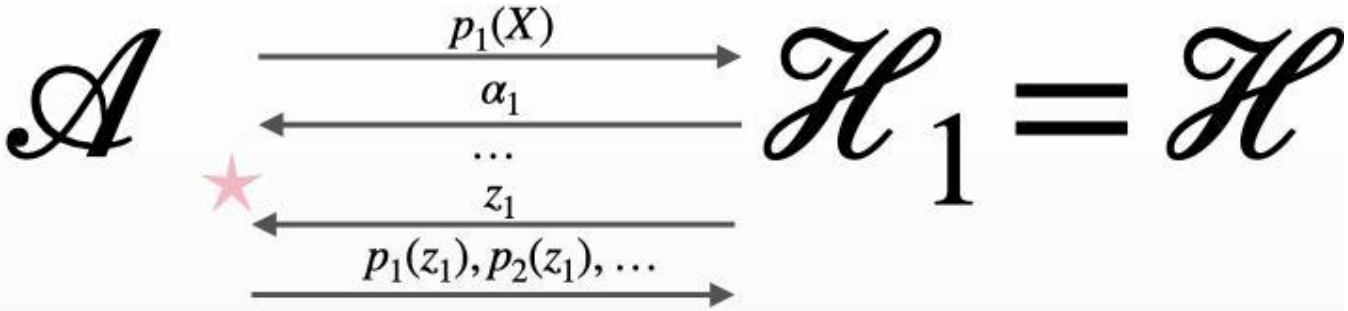
Idea:
Let the extractor learn **multiple evaluations** of polynomials

$$(z_1, p_1(z_1)), \ldots, (z_{n+3}, p_1(z_{n+3}))$$

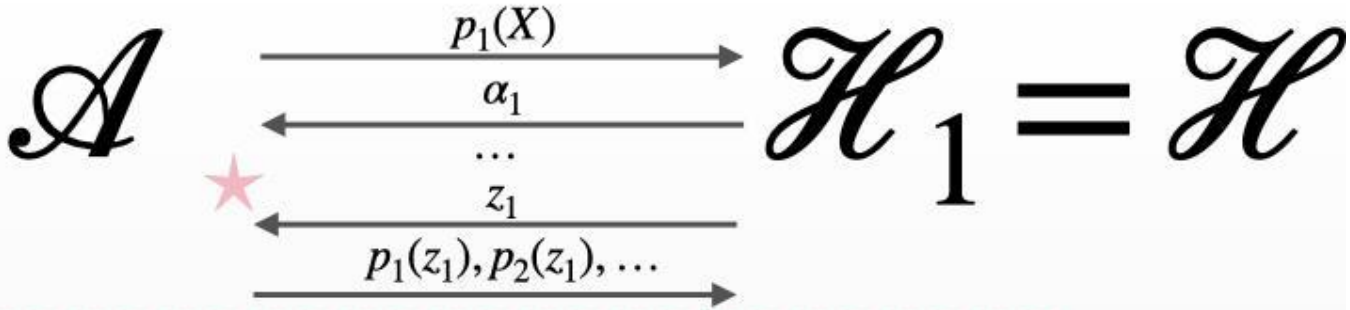
Lagrange **interpolation**

$$p_1(X)$$

Rewinding

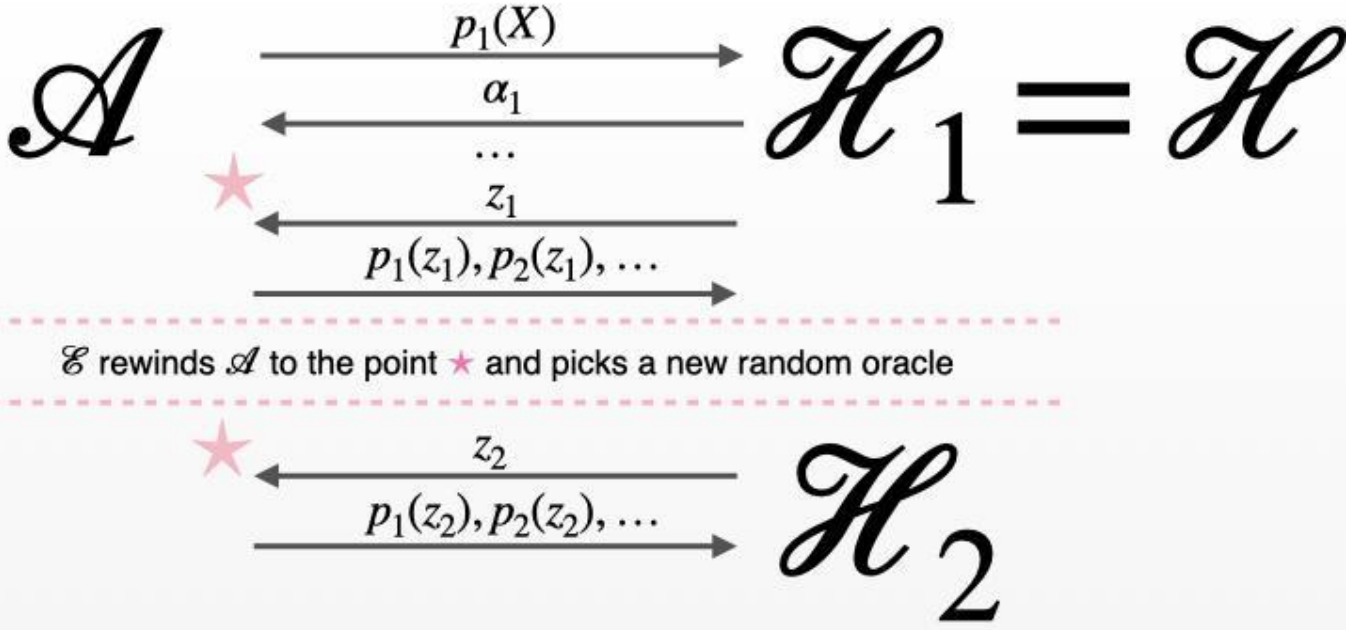


Rewinding

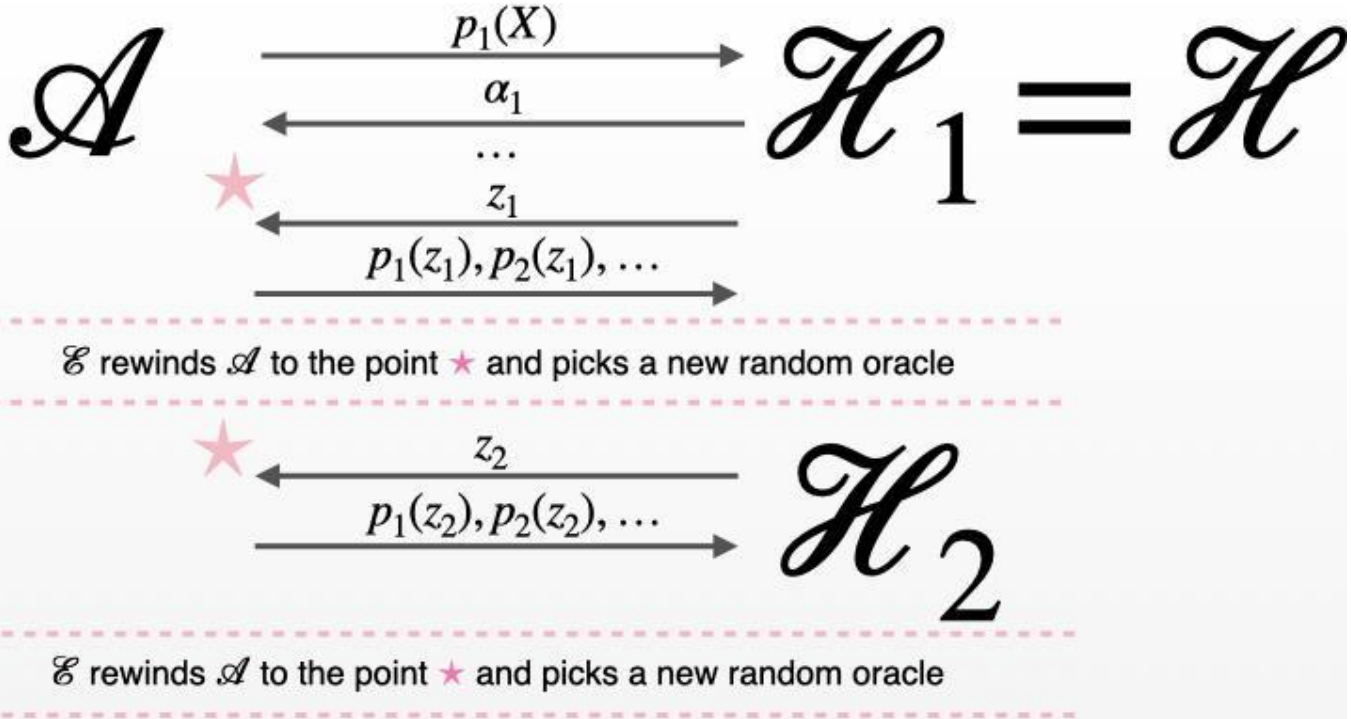


$\mathscr{E}$ rewinds $\mathscr{A}$ to the point ★ and picks a new random oracle

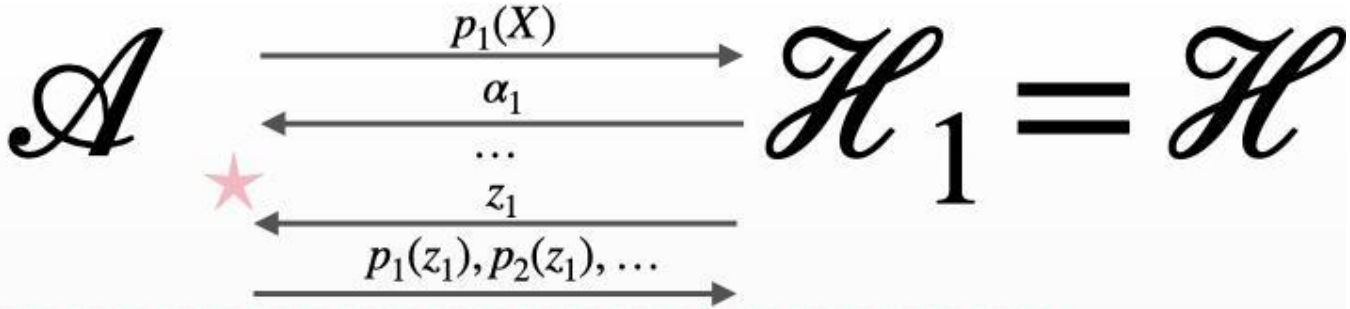
Rewinding


$\mathscr{A}$
$p_1(X)$
α_1
…
z_1
$p_1(z_1), p_2(z_1), \ldots$
$\mathscr{H}_1 = \mathscr{H}$
$\mathscr{E}$ rewinds $\mathscr{A}$ to the point ★ and picks a new random oracle
z_2
$p_1(z_2), p_2(z_2), \ldots$
$\mathscr{H}_2$

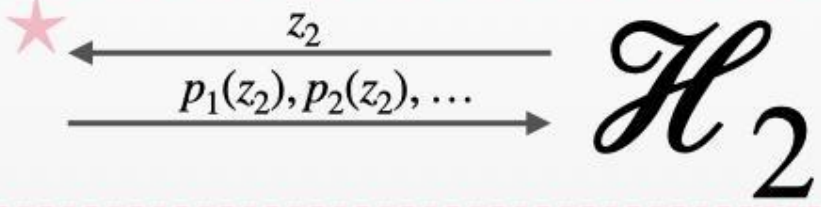
Rewinding


$\mathscr{A}$
$p_1(X)$
α_1
…
z_1
$p_1(z_1), p_2(z_1), \ldots$
$\mathscr{H}_1 = \mathscr{H}$
$\mathscr{E}$ rewinds $\mathscr{A}$ to the point ★ and picks a new random oracle
z_2
$p_1(z_2), p_2(z_2), \ldots$
$\mathscr{H}_2$
$\mathscr{E}$ rewinds $\mathscr{A}$ to the point ★ and picks a new random oracle

Rewinding



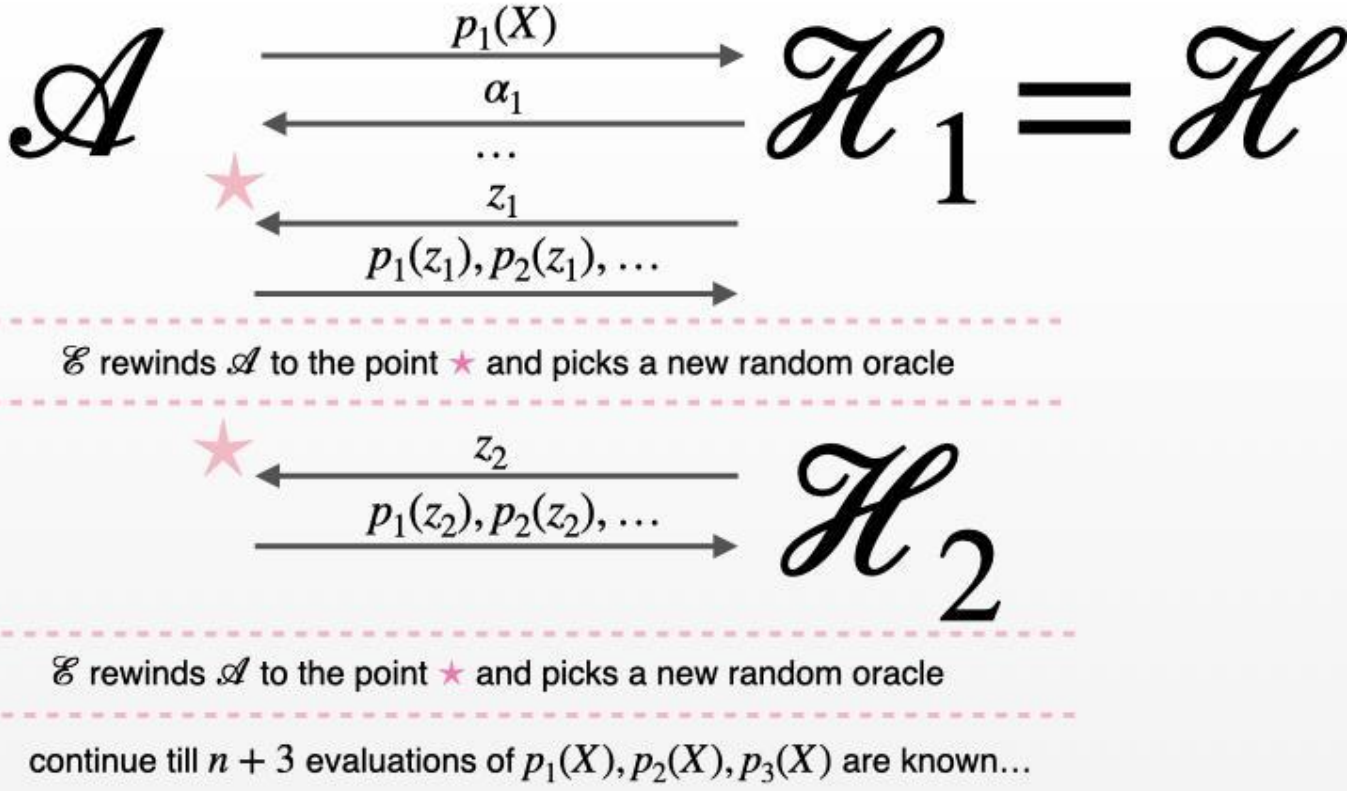
$\mathscr{E}$ rewinds $\mathscr{A}$ to the point ★ and picks a new random oracle



$\mathscr{E}$ rewinds $\mathscr{A}$ to the point ★ and picks a new random oracle

continue till $n + 3$ evaluations of $p_1(X), p_2(X), p_3(X)$ are known…

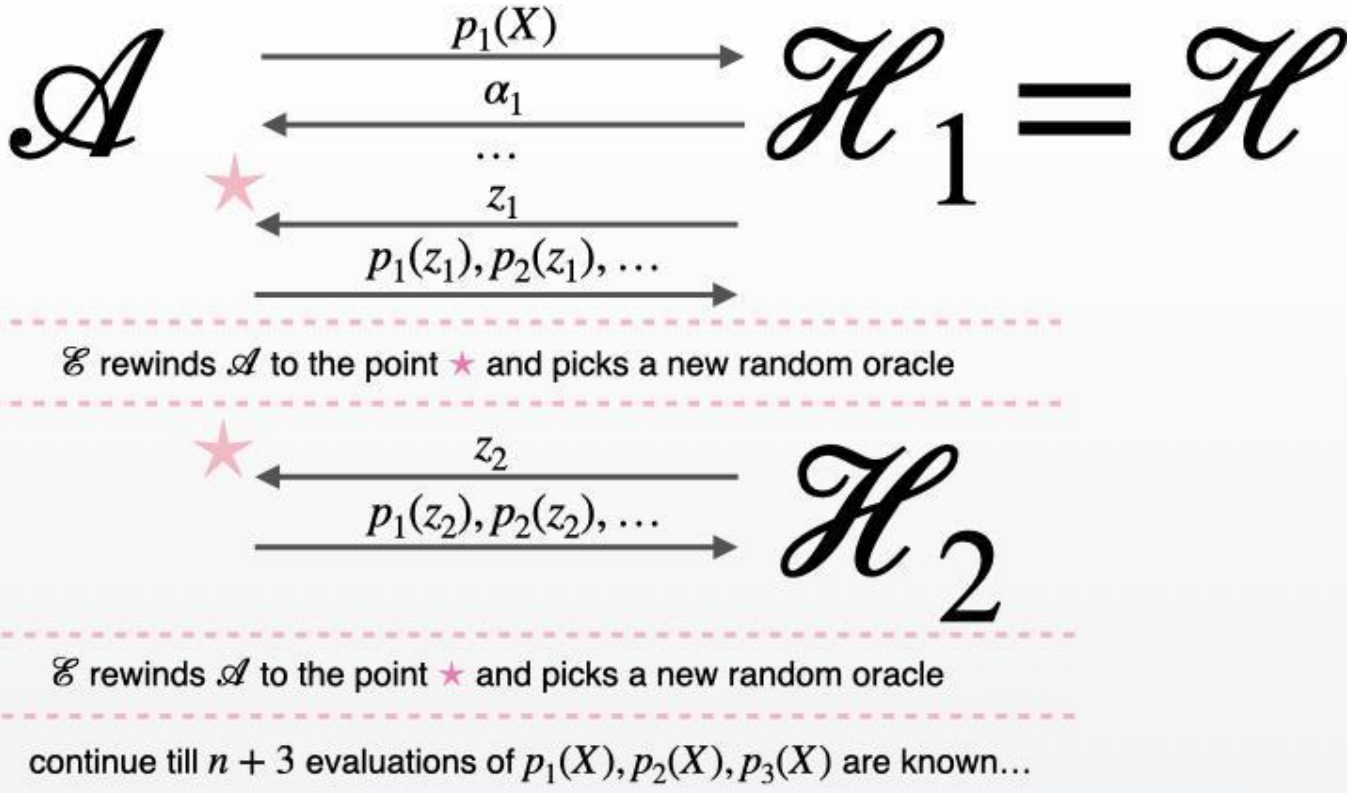
Rewinding



QUESTION 3

How do we know that rewinding reveals the witness? Or the adversary doesn't break if it sees z_2?

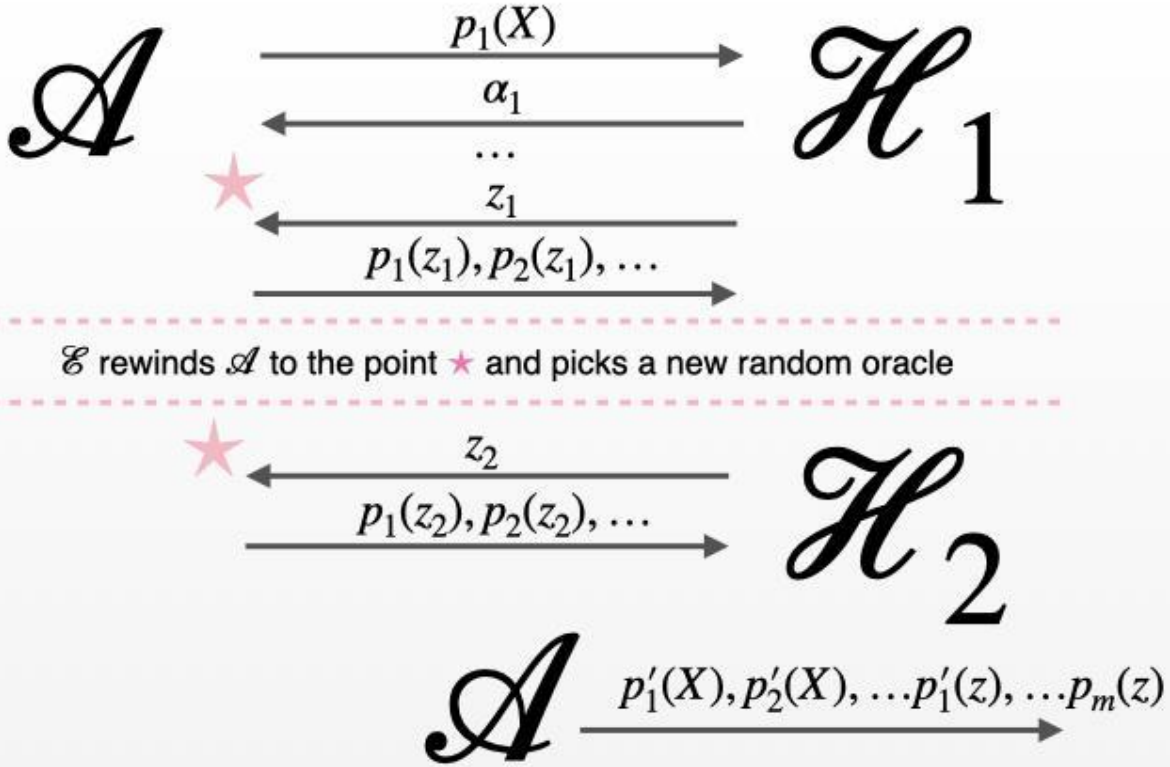
Rewinding



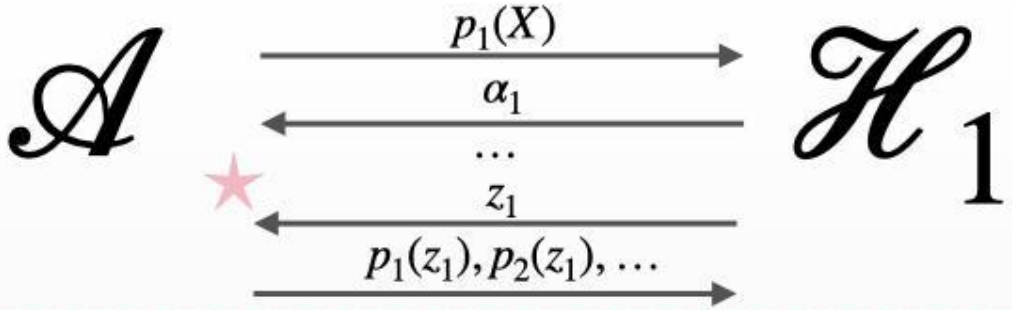
QUESTION 3
How do we know that rewinding reveals the witness? Or the adversary doesn't break if it sees z_2?

REQUIREMENT 3
Rewinding-based knowledge soundness: It is feasible to extract the witness from the polynomials sent by the prover

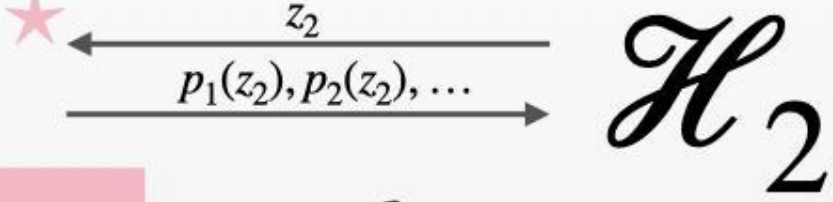
If k too big


$\mathscr{A}$
$p_1(X)$
α_1
…
z_1
$p_1(z_1), p_2(z_1), \ldots$
$\mathscr{H}_1$
$\mathscr{E}$ rewinds $\mathscr{A}$ to the point ★ and picks a new random oracle
z_2
$p_1(z_2), p_2(z_2), \ldots$
$\mathscr{H}_2$
$\mathscr{A}$
$p'_1(X), p'_2(X), \ldots p'_1(z), \ldots p_m(z)$

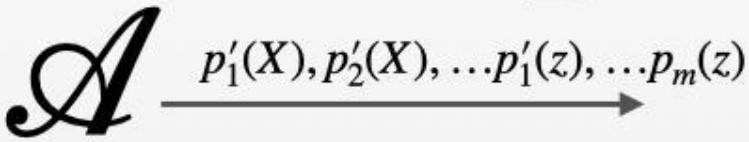
If k too big



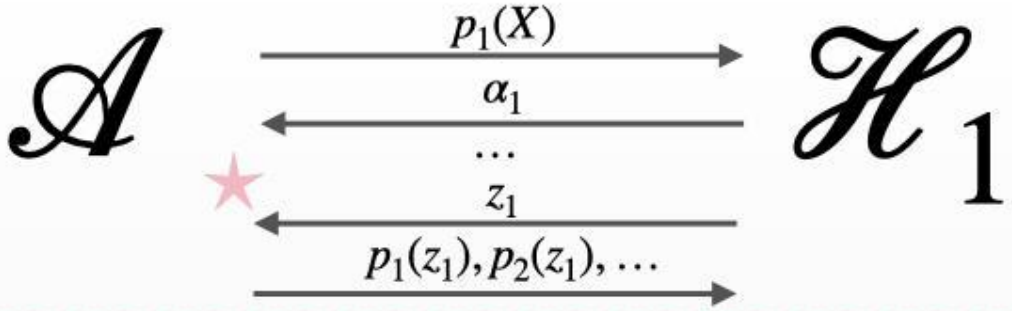
$\mathscr{E}$ rewinds $\mathscr{A}$ to the point ★ and picks a new random oracle



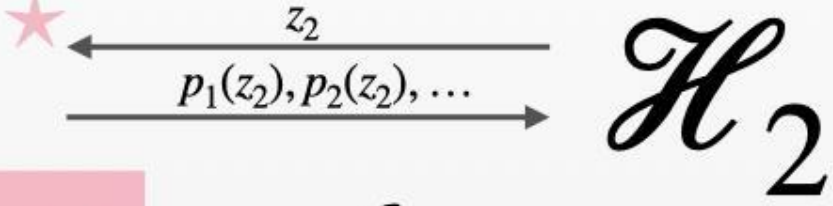
REQUIREMENT 1
Unique response property: It is infeasible to provide two transcripts that
- are different, and
- match on the first k messages, and
- are both acceptable



If k too big

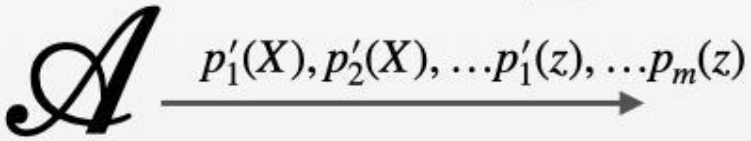


$\mathscr{E}$ rewinds $\mathscr{A}$ to the point ★ and picks a new random oracle



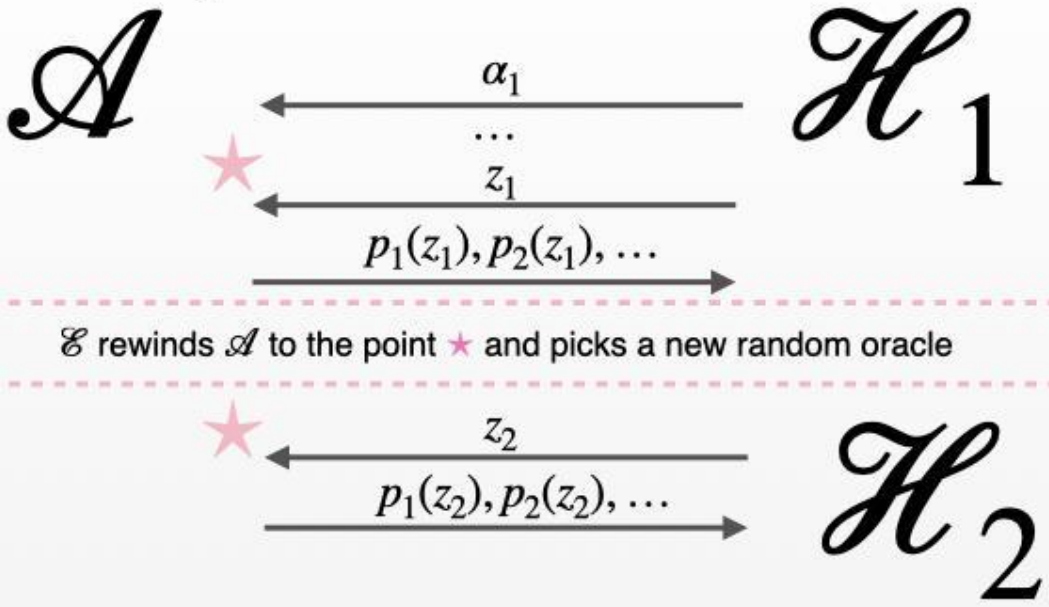
REQUIREMENT 1
Unique response property: It is infeasible to provide two transcripts that
- are different, and
- match on the first k messages, and
- are both acceptable

(challenge z should come after round k)

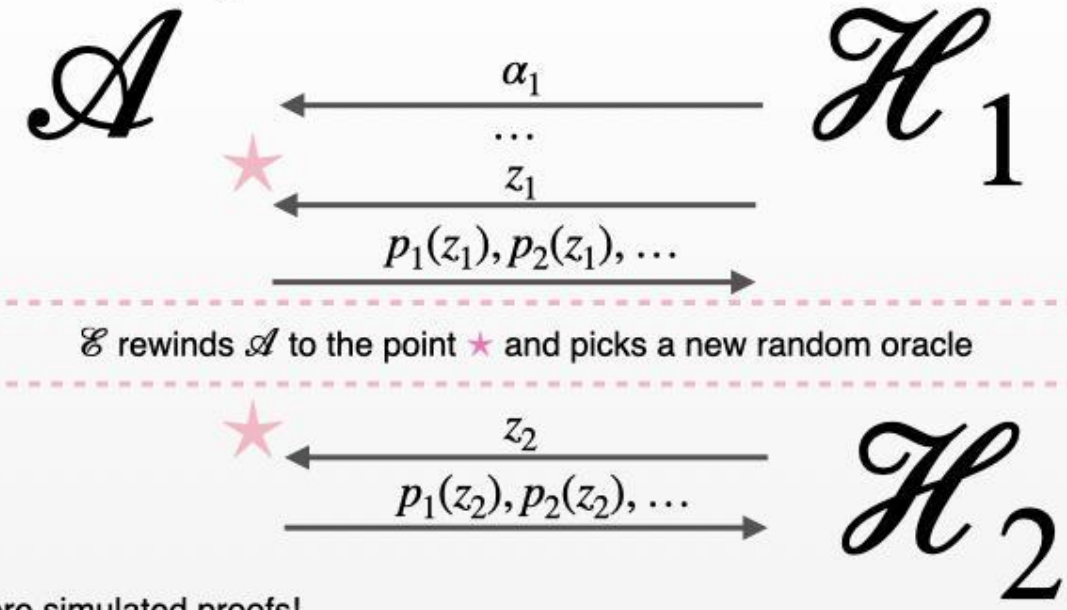


We cannot change the **simulated** proofs the adversary has seen

If $\mathscr{A}$ keeps to ask questions…

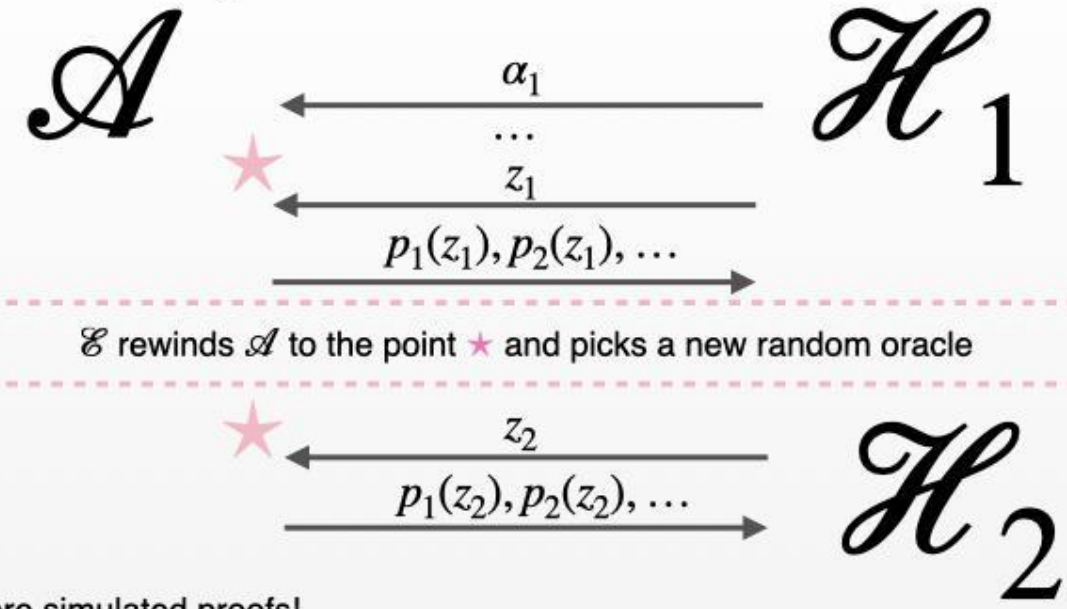


If $\mathscr{A}$ keeps to ask questions…



Give me more simulated proofs!

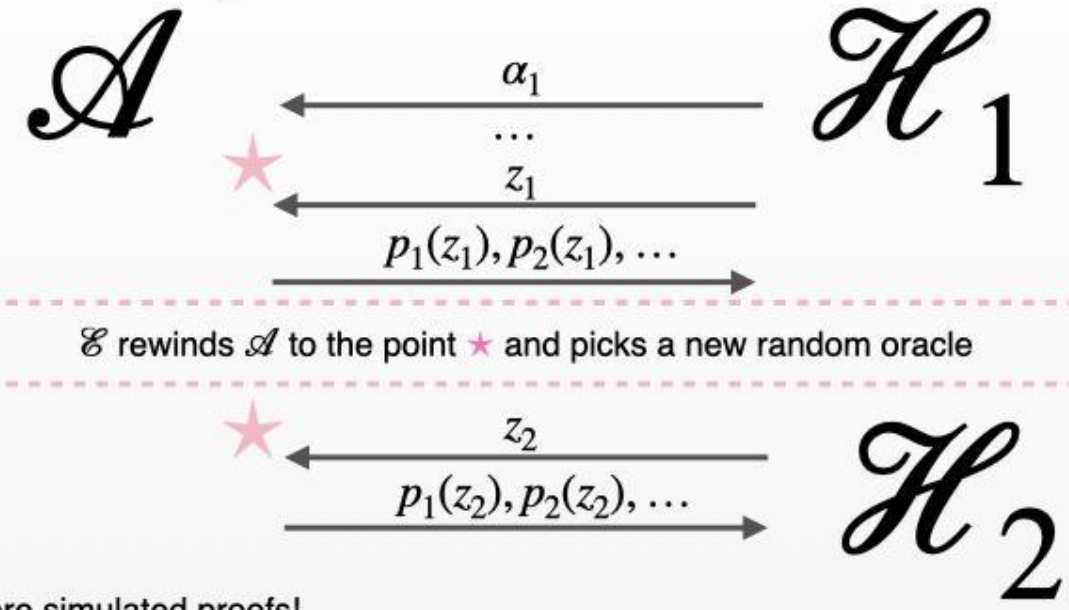
If $\mathscr{A}$ keeps to ask questions…



Give me more simulated proofs!

$$\mathscr{C}_{\mathscr{A}}(p, srs, r, Q_{\mathscr{H}}, Q_{\mathscr{S}})$$

If $\mathscr{A}$ keeps to ask questions…



Give me more simulated proofs!

$$\mathscr{C}_{\mathscr{A}}(p, srs, r, Q_{\mathscr{H}}, Q_{\mathcal{S}})$$

REQUIREMENT 2
Trapdoor-less zero-knowledge: there exists a simulator that provides valid proofs without using SRS's trapdoor

What Makes Fiat–Shamir zkSNARKs (Updatable SRS) Simulation Extractable?

What Makes Fiat–Shamir zkSNARKs (Updatable SRS) Simulation Extractable?

How extractor can provide simulated proofs?

What Makes Fiat–Shamir zkSNARKs (Updatable SRS) Simulation Extractable?

How extractor can provide simulated proofs?

Trapdoor-less zero-knowledge: there exists a simulator that provides valid proofs without using SRS's trapdoor

What Makes Fiat–Shamir zkSNARKs (Updatable SRS) Simulation Extractable?

How extractor can provide simulated proofs?

Trapdoor-less zero-knowledge: there exists a simulator that provides valid proofs without using SRS's trapdoor

What if $\mathscr{A}$ outputs a re-randomized simulated proof?

What Makes Fiat–Shamir zkSNARKs (Updatable SRS) Simulation Extractable?

How extractor can provide simulated proofs?

Trapdoor-less zero-knowledge: there exists a simulator that provides valid proofs without using SRS's trapdoor

What if $\mathscr{A}$ outputs a re-randomized simulated proof?

Unique response property: It is infeasible to provide two transcripts that
- are different, and
- match on the first k messages, and
- are both acceptable

What Makes Fiat–Shamir zkSNARKs (Updatable SRS) Simulation Extractable?

How extractor can provide simulated proofs?

Trapdoor-less zero-knowledge: there exists a simulator that provides valid proofs without using SRS's trapdoor

What if $\mathscr{A}$ outputs a re-randomized simulated proof?

Unique response property: It is infeasible to provide two transcripts that
- are different, and
- match on the first k messages, and
- are both acceptable

How do we know that rewinding reveals the witness?

What Makes Fiat–Shamir zkSNARKs (Updatable SRS) Simulation Extractable?

How extractor can provide simulated proofs?

Trapdoor-less zero-knowledge: there exists a simulator that provides valid proofs without using SRS's trapdoor

What if $\mathscr{A}$ outputs a re-randomized simulated proof?

Unique response property: It is infeasible to provide two transcripts that
- are different, and
- match on the first k messages, and
- are both acceptable

How do we know that rewinding reveals the witness?

Rewinding-based knowledge soundness: It is feasible to extract the witness from the polynomials sent by the prover

What Makes Fiat–Shamir zkSNARKs (Updatable SRS) Simulation Extractable?

How extractor can provide simulated proofs?

Trapdoor-less zero-knowledge: there exists a simulator that provides valid proofs without using SRS's trapdoor

What if $\mathscr{A}$ outputs a re-randomized simulated proof?

Unique response property: It is infeasible to provide two transcripts that
- are different, and
- match on the first k messages, and
- are both acceptable

How do we know that rewinding reveals the witness?

Rewinding-based knowledge soundness: It is feasible to extract the witness from the polynomials sent by the prover

https://eprint.iacr.org/2021/511

Additional slides

Snarky Signatures (aka succinct signatures of knowledge)

$R(x, w)$
Signing of m is possible only if w known
Verification done by using x

Updatable SE

Definition 2 (Updatable Simulation Extractability). *Let* $\Psi_{\mathsf{NI}} = (\mathsf{SRS}, \mathsf{P}, \mathsf{V}, \mathsf{Sim})$ *be a NIZK proof system with an updatable SRS setup. We say that* Ψ_{NI} *is* updatable simulation-extractable *with security loss* $\varepsilon_{\mathsf{se}}(\lambda, \mathsf{acc}, q)$ *if for any* PPT *adversary* $\mathcal{A}$ *that is given oracle access to setup oracle* UpdO *and simulation oracle* SimO *and that produces an accepting proof for* Ψ_{NI} *with probability* acc*, where*

$$\mathsf{acc} = \Pr\left[\begin{array}{c|c} \mathsf{V}(\mathsf{srs}, \mathsf{x}, \pi) = 1 \\ \wedge (\mathsf{x}, \pi) \notin Q \end{array} \middle| \begin{array}{c} r \leftarrow\$ \mathsf{R}(\mathcal{A}) \\ (\mathsf{x}, \pi) \leftarrow \mathcal{A}^{\mathsf{UpdO}, \mathsf{SimO}.\mathcal{H}, \mathsf{SimO}.\mathsf{P}'}(1^\lambda; r) \end{array}\right]$$

there exists an expected PPT extractor $\mathsf{Ext}_{\mathsf{se}}$ *such that*

$$\Pr\left[\begin{array}{c|c} \mathsf{V}(\mathsf{srs}, \mathsf{x}, \pi) = 1, \\ (\mathsf{x}, \pi) \notin Q, \\ \mathbf{R}(\mathsf{x}, \mathsf{w}) = 0 \end{array} \middle| \begin{array}{c} r \leftarrow\$ \mathsf{R}(\mathcal{A}), (\mathsf{x}, \pi) \leftarrow \mathcal{A}^{\mathsf{UpdO}, \mathsf{SimO}.\mathcal{H}, \mathsf{SimO}.\mathsf{P}'}(1^\lambda; r) \\ \mathsf{w} \leftarrow \mathsf{Ext}_{\mathsf{se}}(\mathsf{srs}, \mathcal{A}, r, Q_{\mathsf{srs}}, Q_{\mathcal{H}}, Q) \end{array}\right] \leq \varepsilon_{\mathsf{se}}(\lambda, \mathsf{acc}, q)$$

Here, srs *is the finalized SRS. List* Q_{srs} *contains all* (srs, ρ) *of update SRSs and their proofs, list* $Q_{\mathcal{H}}$ *contains all* $\mathcal{A}$*'s queries to* $\mathsf{SimO}.\mathcal{H}$ *and the (simulated) random oracle's answers,* $|Q_{\mathcal{H}}| \leq q$*, and list* Q *contains all* (x, π) *pairs where* x *is an instance queried to* $\mathsf{SimO}.\mathsf{P}'$ *by the adversary and* π *is the simulator's answer .*

Contribution

Define **updatable simulation extractability** for NIZKs

Analyze three properties **needed** for the NIZK to be simulation extractable

Show that a wide class (Plonk, Marlin, Lunar, Sonic) of **zkSNARKs** is simulation-extractable **out of the box**

no changes in the protocol needed $\mapsto$ **no efficiency loss**!

Snarky signature in the **updatable** setting

zkSNARKs properties

$$\mathcal{P}(srs, x, w) \xrightarrow{\pi} \mathcal{V}(srs, x)$$

$$R(x, w)$$

Completeness
Honest verifier always accepts a proof from an honest prover $\mathcal{P}$

Knowledge soundness
If verifier $\mathcal{V}$ accepts a proof for a statement x then the prover $\mathcal{A}(p, srs; r)$ **knows** w such that $R(x, w)$

$$\mathcal{E}_{\mathcal{A}}(p, srs, r) = w$$

Zero knowledge
Verifiers learns nothing besides the validity of the statement

$$\mathcal{S}(p, srs, td, x) \approx \mathcal{P}(p, srs, x, w)$$

Succinctness
The proof π is short

$$|\pi| = O(\log(|x| + |w|))$$

All the fuss with the SRS

$$td, srs \leftarrow \mathsf{KGen}(p)$$

Trusted party
Party that creates the SRS can easily provide valid proofs for false statements

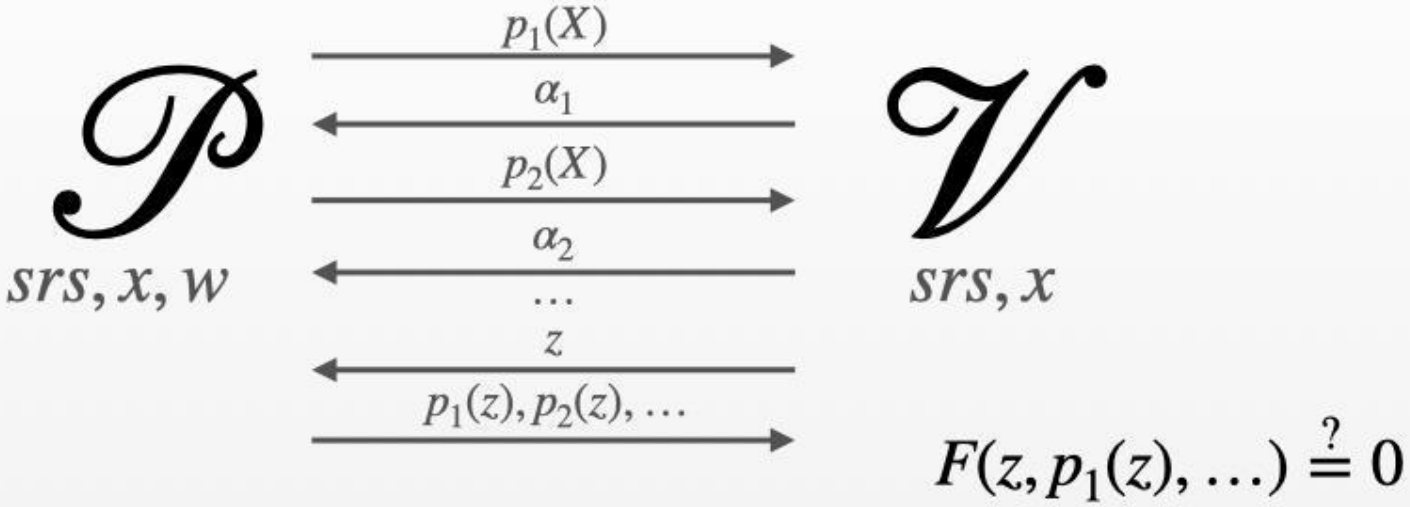
$$\mathcal{S}(p, srs, td, x)$$

Real life instantiation
MPC ceremonies to generate the SRS

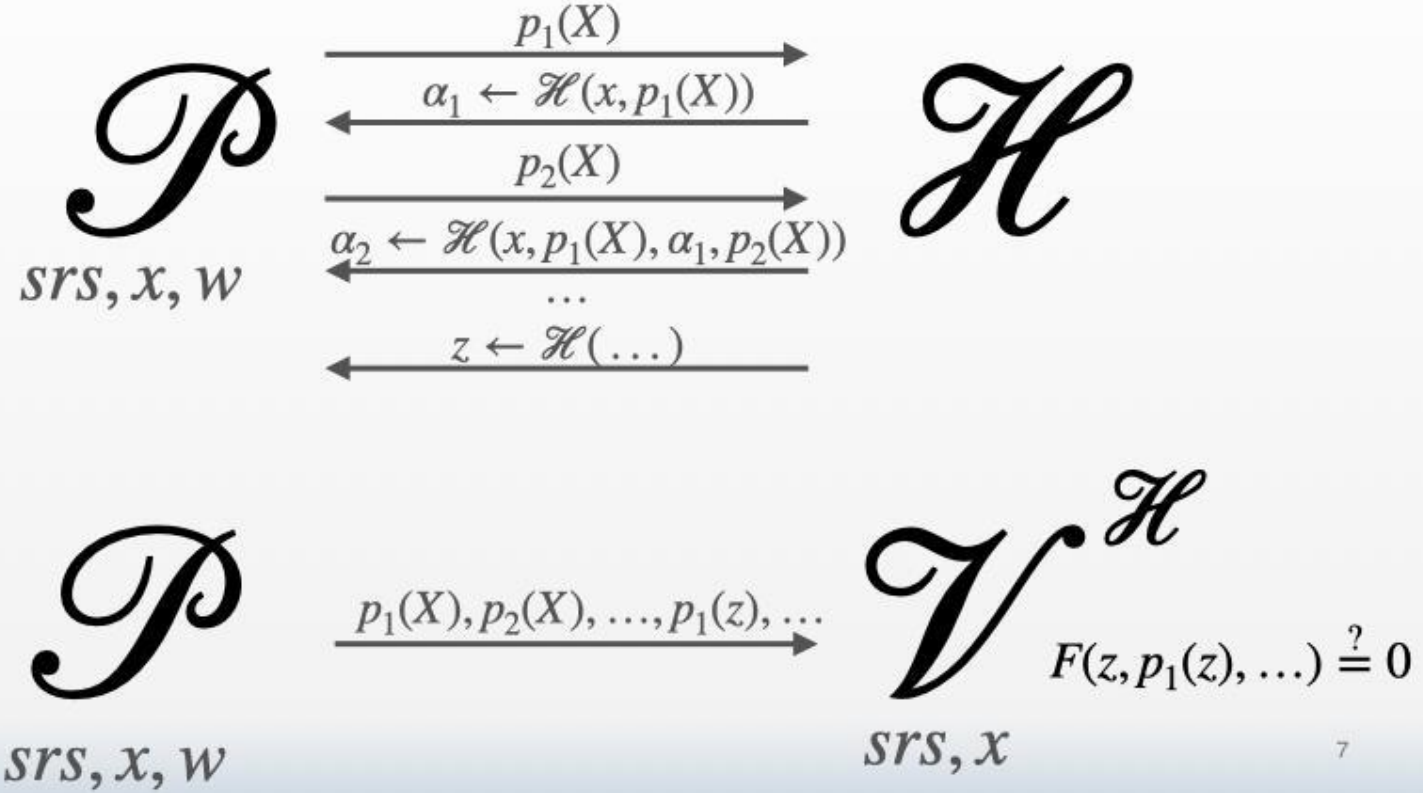
Updatable SRS
Everybody can take an existing SRS srs and **update** it to get an SRS srs'
One honest update is enough for the proof system to be sound

$$\mathsf{Update}(srs) \rightarrow srs', \pi_{upd}$$

How are zkSNARK-s built?



From interactive to non interactive: Fiat–Shamir transformation



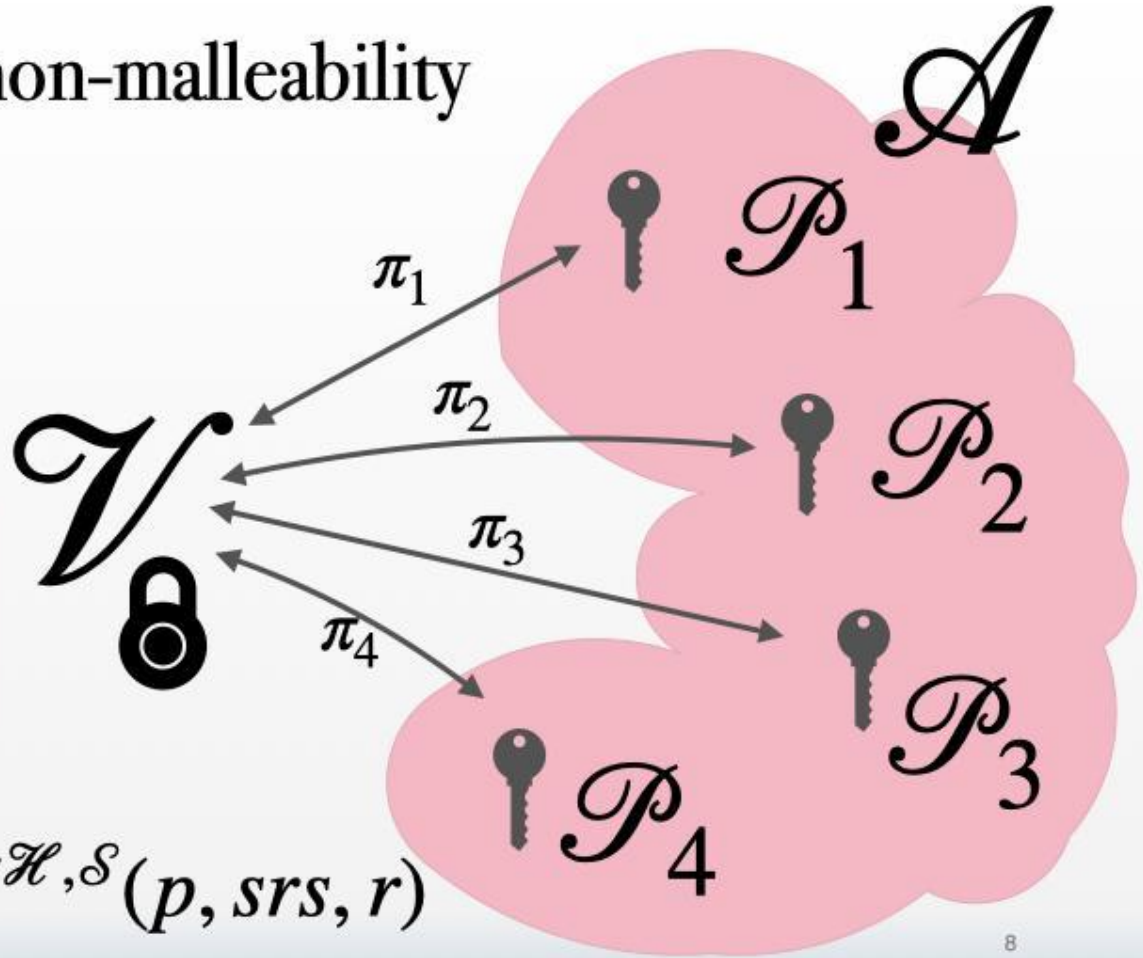
The importance of non-malleability

Knowledge soundness: $\mathcal{A}^{\mathcal{H}}(p, srs; r)$ cannot convince the verifier $\mathcal{V}$ on x unless it knows w such that $R(x, w)$

Extraction: $\mathcal{E}_{\mathcal{A}}(p, srs, r, Q_{\mathcal{H}}) = w$

Simulation-extractability: $\mathcal{A}^{\mathcal{H},\mathcal{S}}(p, srs; r)$ cannot convince the verifier $\mathcal{V}$ on x unless it knows w such that $R(x, w)$ **given access to simulated proofs**

Extraction: $\mathcal{E}_{\mathcal{A}}(p, srs, r, Q_{\mathcal{H}}, Q_{\mathcal{S}}) = w$



$$\mathcal{A}^{\mathcal{H}}(p, srs, r) \mapsto \mathcal{A}^{\mathcal{H},\mathcal{S}}(p, srs, r)$$

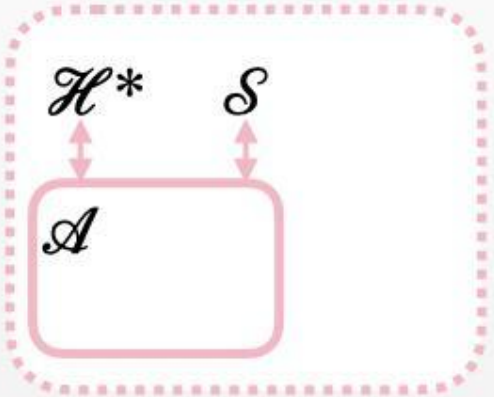
Simulation-extractability

$$\mathcal{A}^{\mathcal{H},\mathcal{S}}(p, srs, r) = x, \pi$$

$\mathcal{S}(x)$ responds with a simulated proof for x

When adversary wins?
$\mathcal{A}$ returns an instance—proof pair (x, π) such that
- π is not a simulated proof
- $\mathcal{A}$ doesn't know the witness

$$\mathcal{E}_{\mathcal{A}}(p, srs, r, Q_{\mathcal{H}}, Q_{\mathcal{S}}) = w \qquad \mathcal{H}$$

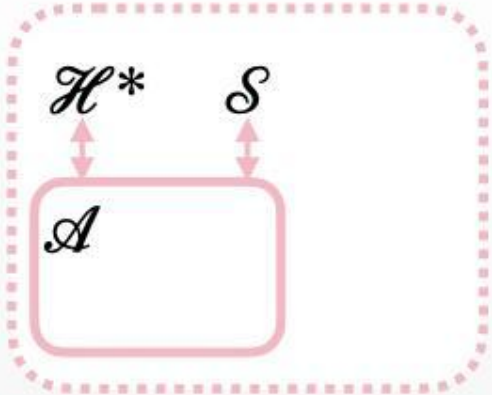


Some issues with extraction

$\mathscr{E}_{\mathscr{A}}(p, srs, r, Q_{\mathscr{H}}, Q_{\mathcal{S}}) = w$

$\mathscr{H}$

$\mathscr{A}^{\mathscr{H},\mathcal{S}}(p, srs, r) = x, \pi$



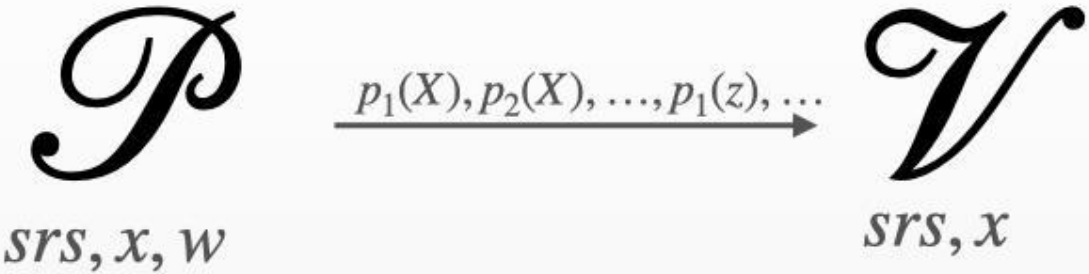
QUESTION 1
What if $\mathscr{A}$ outputs a re-randomized simulated proof?

QUESTION 2
How can $\mathscr{E}$ provide simulated proofs?

REQUIREMENT 1
Unique response property: It is infeasible to provide two transcripts that
- are different, and
- match on the first k messages, and
- are both acceptable

REQUIREMENT 2
Trapdoor-less zero-knowledge: there exists a simulator that provides valid proofs without using SRS's trapdoor

How are zkSNARK-s built?



Plonk:

$w = (w_1, \ldots, w_{3n})$

$p_1(X) = w_1 + w_2 X + \ldots w_n X^{n-1} + \ldots$

$p_2(X) = w_{n+1} + w_{n+2} X + \ldots w_{2n} X^{n-1} + \ldots$

$p_3(X) = w_{2n+1} + w_{2n+2} X + \ldots w_{3n} X^{n-1} + \ldots$

Observation:
Some of the polynomials $p_i(X)$ **encode** the witness

$\text{Decode}(p_1(X), \ldots) = w$

How to extract?

Plonk:

$w = (w_1, \ldots, w_{3n})$

$p_1(X) = w_1 + w_2 X + \ldots w_n X^{n-1} + \ldots$

$p_2(X) = w_{n+1} + w_{n+2} X + \ldots w_{2n} X^{n-1} + \ldots$

$p_3(X) = w_{2n+1} + w_{2n+2} X + \ldots w_{3n} X^{n-1} + \ldots$

Observation:

$\deg(p_1) = \deg(p_2) = \deg(p_3) = n + 2$

If we evaluate $p_1(X), p_2(X), p_3(X)$ at $n + 3$ points then we can reveal their coefficients

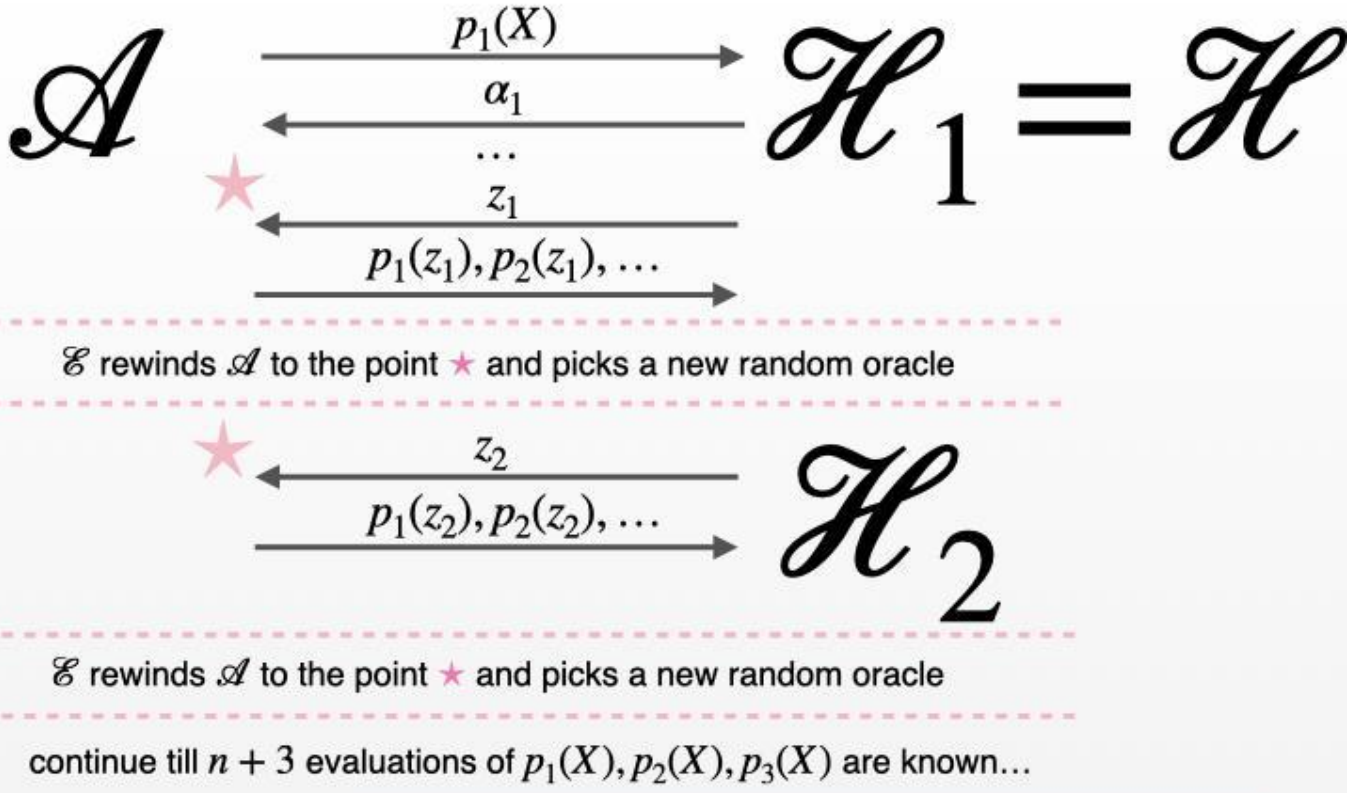
Idea:
Let the extractor learn **multiple evaluations** of polynomials

$$(z_1, p_1(z_1)), \ldots, (z_{n+3}, p_1(z_{n+3}))$$

Lagrange **interpolation**

$$p_1(X)$$

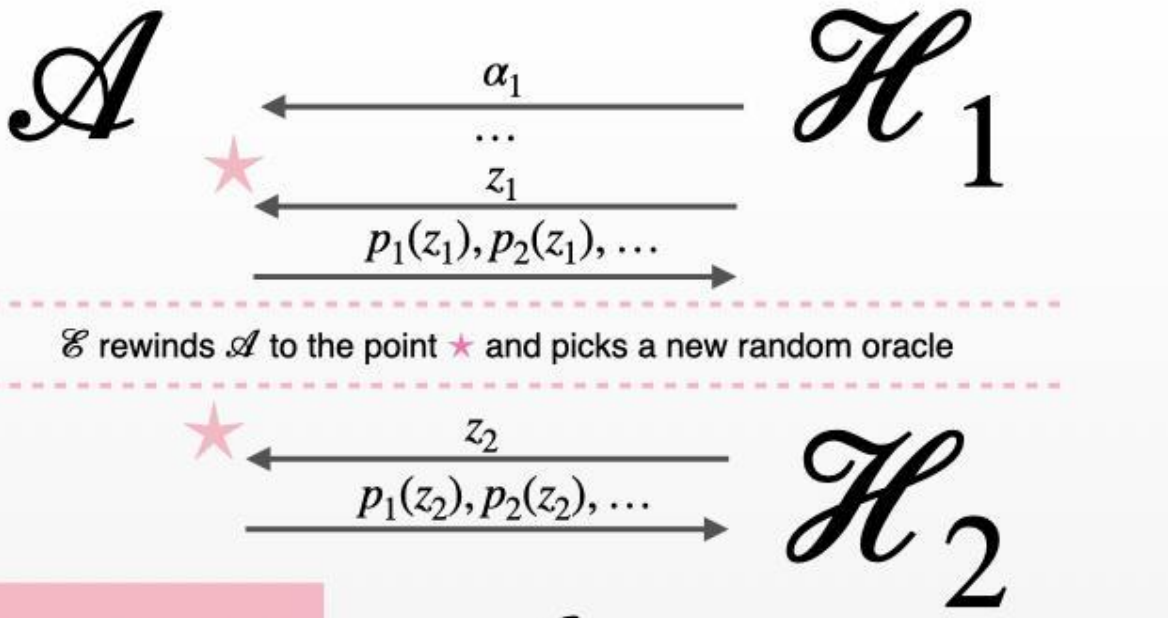
Rewinding



QUESTION 3
How do we know that rewinding reveals the witness? Or the adversary doesn't break if it sees z_2?

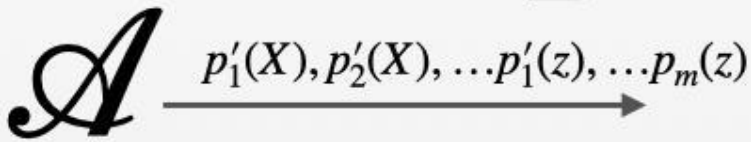
REQUIREMENT 3
Rewinding-based knowledge soundness: It is feasible to extract the witness from the polynomials sent by the prover

If k too big



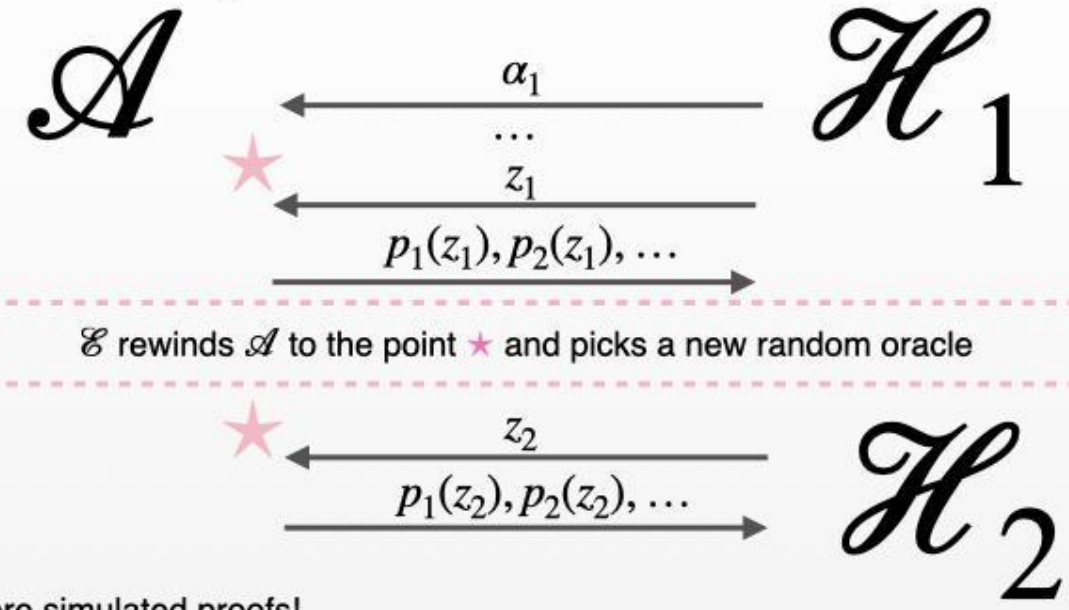
REQUIREMENT 1
Unique response property: It is infeasible to provide two transcripts that
- are different, and
- match on the first k messages, and
- are both acceptable

(challenge z should come after round k)



We cannot change the **simulated** proofs the adversary has seen

If $\mathscr{A}$ keeps to ask questions…



Give me more simulated proofs!

$$\mathscr{C}_{\mathscr{A}}(p, srs, r, Q_{\mathscr{H}}, Q_{\mathcal{S}})$$

REQUIREMENT 2
Trapdoor-less zero-knowledge: there exists a simulator that provides valid proofs without using SRS's trapdoor

What Makes Fiat–Shamir zkSNARKs (Updatable SRS) Simulation Extractable?

How extractor can provide simulated proofs?

Trapdoor-less zero-knowledge: there exists a simulator that provides valid proofs without using SRS's trapdoor

What if $\mathscr{A}$ outputs a re-randomized simulated proof?

Unique response property: It is infeasible to provide two transcripts that
- are different, and
- match on the first k messages, and
- are both acceptable

How do we know that rewinding reveals the witness?

Rewinding-based knowledge soundness: It is feasible to extract the witness from the polynomials sent by the prover

What Makes Fiat–Shamir zkSNARKs (Updatable SRS) Simulation Extractable?

How extractor can provide simulated proofs?

Trapdoor-less zero-knowledge: there exists a simulator that provides valid proofs without using SRS's trapdoor

What if $\mathscr{A}$ outputs a re-randomized simulated proof?

Unique response property: It is infeasible to provide two transcripts that
- are different, and
- match on the first k messages, and
- are both acceptable

How do we know that rewinding reveals the witness?

Rewinding-based knowledge soundness: It is feasible to extract the witness from the polynomials sent by the prover

Additional slides

Snarky Signatures (aka succinct signatures of knowledge)

$R(x, w)$
Signing of m is possible only if w known
Verification done by using x

Updatable SE

Definition 2 (Updatable Simulation Extractability). *Let* $\Psi_{\mathsf{NI}} = (\mathsf{SRS}, \mathsf{P}, \mathsf{V}, \mathsf{Sim})$ *be a NIZK proof system with an updatable SRS setup. We say that* Ψ_{NI} *is* updatable simulation-extractable *with security loss* $\varepsilon_{\mathsf{se}}(\lambda, \mathsf{acc}, q)$ *if for any* PPT *adversary* $\mathcal{A}$ *that is given oracle access to setup oracle* UpdO *and simulation oracle* SimO *and that produces an accepting proof for* Ψ_{NI} *with probability* acc*, where*

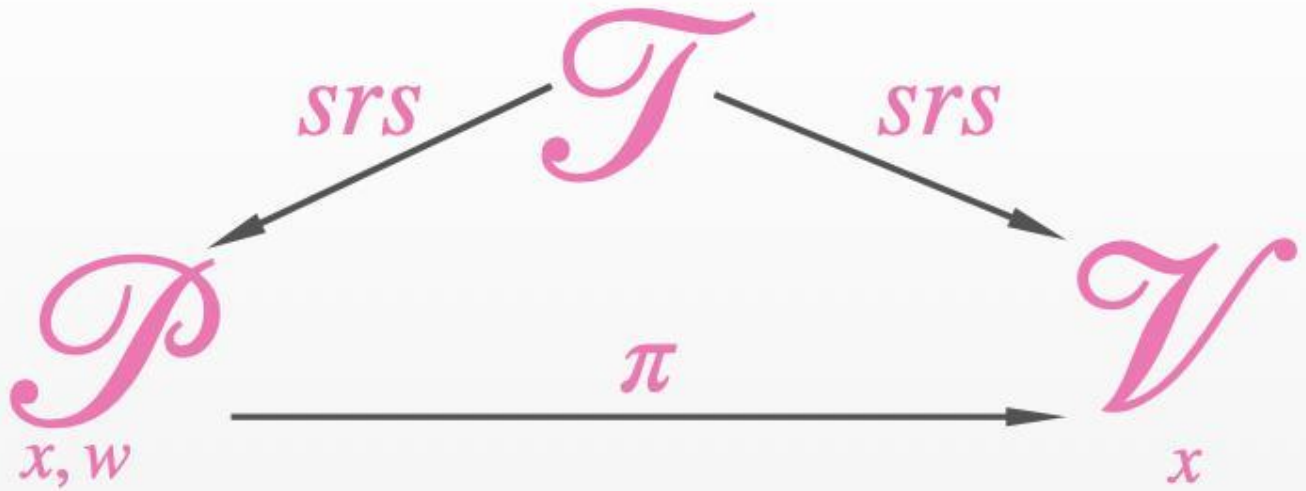
$$\mathsf{acc} = \Pr\left[\begin{array}{c|c} \mathsf{V}(\mathsf{srs}, \mathsf{x}, \pi) = 1 \\ \wedge (\mathsf{x}, \pi) \notin Q \end{array} \middle| \begin{array}{c} r \leftarrow\$ \mathsf{R}(\mathcal{A}) \\ (\mathsf{x}, \pi) \leftarrow \mathcal{A}^{\mathsf{UpdO}, \mathsf{SimO}.\mathcal{H}, \mathsf{SimO}.\mathsf{P}'}(1^\lambda; r) \end{array}\right]$$

there exists an expected PPT extractor $\mathsf{Ext}_{\mathsf{se}}$ *such that*

$$\Pr\left[\begin{array}{c|c} \mathsf{V}(\mathsf{srs}, \mathsf{x}, \pi) = 1, \\ (\mathsf{x}, \pi) \notin Q, \\ \mathbf{R}(\mathsf{x}, \mathsf{w}) = 0 \end{array} \middle| \begin{array}{c} r \leftarrow\$ \mathsf{R}(\mathcal{A}), (\mathsf{x}, \pi) \leftarrow \mathcal{A}^{\mathsf{UpdO}, \mathsf{SimO}.\mathcal{H}, \mathsf{SimO}.\mathsf{P}'}(1^\lambda; r) \\ \mathsf{w} \leftarrow \mathsf{Ext}_{\mathsf{se}}(\mathsf{srs}, \mathcal{A}, r, Q_{\mathsf{srs}}, Q_{\mathcal{H}}, Q) \end{array}\right] \leq \varepsilon_{\mathsf{se}}(\lambda, \mathsf{acc}, q)$$

Here, srs *is the finalized SRS. List* Q_{srs} *contains all* (srs, ρ) *of update SRSs and their proofs, list* $Q_{\mathcal{H}}$ *contains all* $\mathcal{A}$*'s queries to* $\mathsf{SimO}.\mathcal{H}$ *and the (simulated) random oracle's answers,* $|Q_{\mathcal{H}}| \leq q$*, and list* Q *contains all* (x, π) *pairs where* x *is an instance queried to* $\mathsf{SimO}.\mathsf{P}'$ *by the adversary and* π *is the simulator's answer .*

Zero-knowledge proofs



$$\langle \mathcal{S}(p, srs, td, x), \mathcal{V}(p, srs, x) \rangle \approx \langle \mathcal{P}(p, srs, x, w), \mathcal{V}(p, srs, x) \rangle$$