Zero Knowledge Proofs

FRI-based Polynomial Commitments and Fiat-Shamir

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





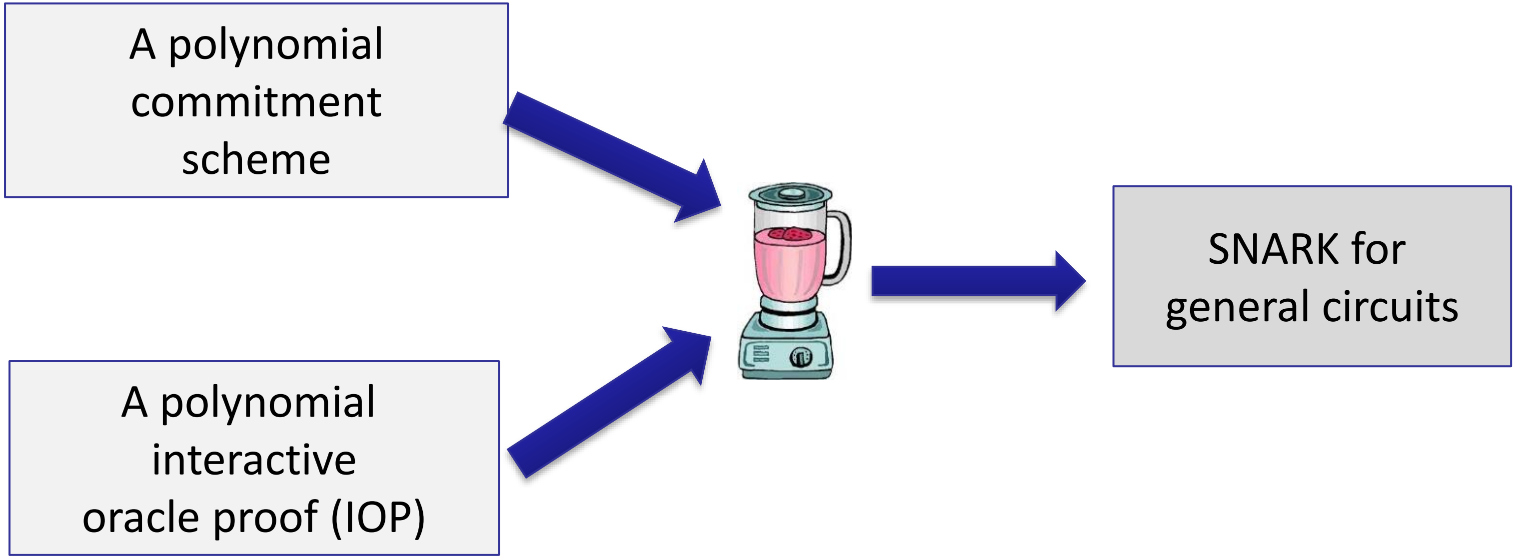
Stanford University

Berkeley UNIVERSITY OF CALIFORNIA

GEORGETOWN UNIVERSITY

TEXAS A&M UNIVERSITY

Let’s build an efficient SNARK



Recall: What is a Polynomial-IOP?

- P's first message in the protocol is a **polynomial** h**.**
  - V does **not** learn h in full.
    - The description size of h is as large as the circuit.
  - Rather, V is permitted to evaluate h at, say, **one** point.
  - After that, P and V execute a standard interactive proof.

Recall: What is a Polynomial Commitment Scheme?

- High-level idea:
  - P binds itself to a polynomial h by sending a short string $\mathrm{Com}(h)$.
  - V can choose x and ask P to evaluate $h(x)$.
  - P sends y, the purported evaluation, plus a proof π that y is consistent with $\mathrm{Com}(h)$ and x.
- Goals:
  - P cannot produce a convincing proof for an incorrect evaluation.
  - $\mathrm{Com}(h)$ and π are short and easy to generate; π is easy to check.

A Zoo of SNARKs

- There are several different polynomial IOPs in the literature.
- And several different polynomial commitments.
- Can mix-and-match to get different tradeoffs between P time, proof size, setup assumptions, etc.
  - Transparency and plausible post-quantum security determined entirely by the polynomial commitment scheme used.

Polynomial IOPs: Three classes

1. Based on interactive proofs (IPs).
2. Based on multi-prover interactive proofs (MIPs).
3. Based on constant-round polynomial IOPs.
   - Examples: Marlin, PlonK.

- Above SNARKs roughly listed in increasing order of P costs and decreasing order of proof length and V cost.
- Categories 1 and 2 covered in Lecture 4, Category 3 (PlonK) in Lecture 5.

Polynomial commitments: Three classes

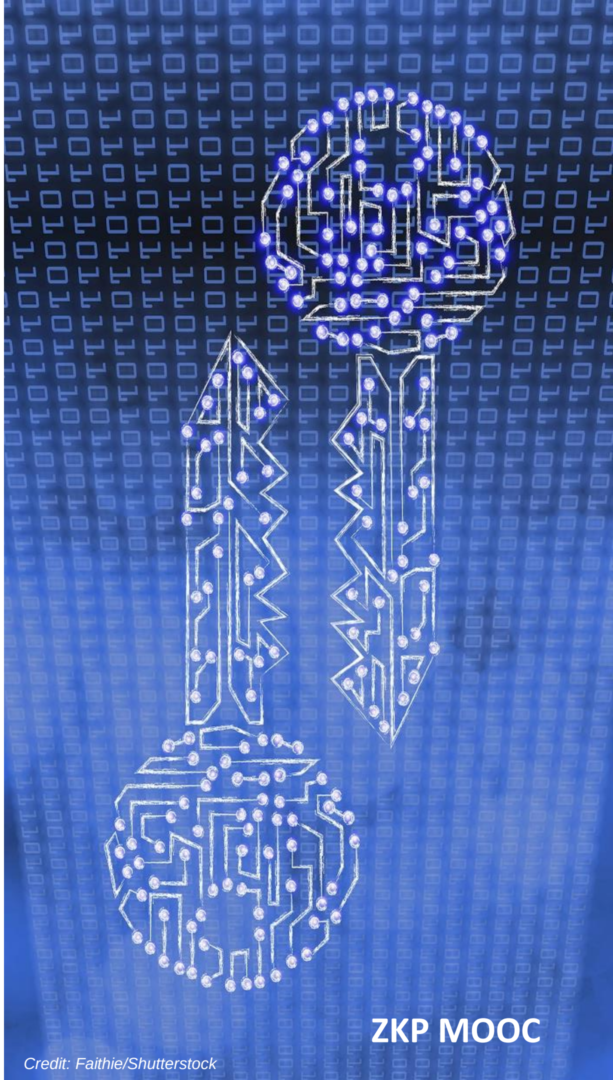
1. Based on pairings + trusted setup (**not** transparent **nor** post-quantum).
   - e.g., **KZG10** (Lecture 5 + 6).
   - Unique property: constant sized evaluation proofs.
2. Based on discrete logarithm (transparent, **not** post-quantum).
   - Examples: **IPA/Bulletproofs** (Lecture 6), Hyrax, Dory.
3. Based on IOPs + hashing (transparent **and** post-quantum)
   - e.g., **FRI** (will be covered today), Ligero, Brakedown, Orion (Lecture 7).

Polynomial commitments: Three classes

1. Based on pairings + trusted setup (**not** transparent **nor** post-quantum).
   - e.g., **KZG10** (Lecture 5 + 6).
   - Unique property: constant sized evaluation proofs.
2. Based on discrete logarithm (transparent, **not** post-quantum).
   - Examples: **IPA/Bulletproofs** (Lecture 6), Hyrax, Dory.

- Classes 1. and 2. are homomorphic.
  - Leads to efficient batching/amortization of P and V costs (e.g., when proving knowledge of several different witnesses).

Some specimens from the zoo



Credit: Faithie/Shutterstock

Highlights of SNARK Taxonomy: Transparent SNARKs

1. [Any polynomial IOP] + IPA/Bulletproofs polynomial commitment.
   - **Ex: Halo2-ZCash**
   - Pros: Shortest proofs among transparent SNARKs.
   - Cons: Slow V

Highlights of SNARK Taxonomy: Transparent SNARKs

2. [Any polynomial IOP] + FRI polynomial commitment.
   - **Ex: STARKs, Fractal, Aurora, Virgo, Ligero++**
   - Pros: Shortest proofs amongst plausibly post-quantum SNARKs.
   - Cons: Proofs are large (100s of KBs depending on security)

Highlights of SNARK Taxonomy: Transparent SNARKs

3. MIPs and IPs + [fast-prover polynomial commitments].
   - **Ex: Spartan, Brakedown, Orion, Orion+.**
   - Pros: Fastest P in the literature, plausibly post-quantum + transparent if polynomial commitment is.
   - Cons: Bigger proofs than 1. and 2. above.

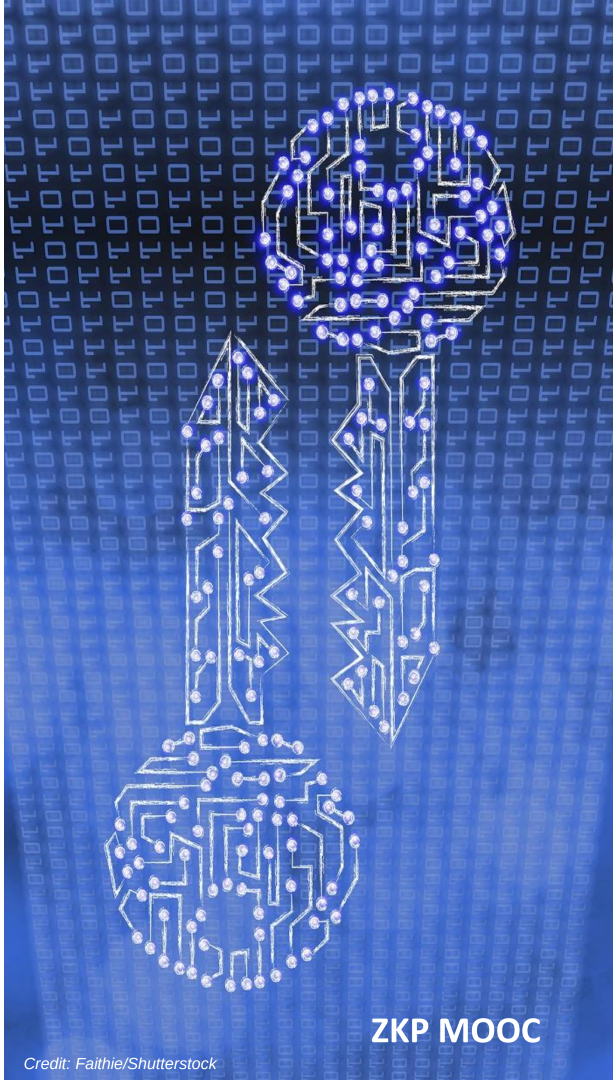
Highlights of SNARK Taxonomy: **Non-transparent SNARKS**

1. Linear-PCP based:
   - **Ex: Groth16**
   - Pros: Shortest proofs (3 group elements), fastest V.
   - Cons: Circuit-specific trusted setup, slow and space-intensive P, not post-quantum

Highlights of SNARK Taxonomy: **Non-transparent SNARKS**

2. Constant-round polynomial IOP + KZG polynomial commitment:
   - **Ex: Marlin-KZG, PlonK-KZG**
   - Pros: Universal trusted setup.
   - Cons: Proofs are ~4x-6x larger than Groth16, P is slower than Groth16, also not post-quantum.
     - Counterpoint for P: can use more flexible intermediate representations than circuits and R1CS.

FRI (Univariate) Polynomial Commitment



Credit: Faithie/Shutterstock

Recall: Univariate Polynomial Commitments

1. Let q be a degree-$(k-1)$ polynomial over field $\mathbb{F}_p$.
   - E.g., $k = 5$ and $q(X) = 1 + 2X + 4X^2 + X^4$
2. Want P to succinctly commit to q, later reveal $q(r)$ for an $r \in \mathbb{F}_p$ chosen by V.
   - Along with associated “evaluation proof”.

Recall: Initial Attempt from Lecture 4

- P Merkle-commits to all evaluations of the polynomial q.
- When V requests $q(r)$, P reveals the associated leaf along with opening information.

Recall: Initial Attempt from Lecture 4

- P Merkle-commits to all evaluations of the polynomial q.
- When V requests $q(r)$, P reveals the associated leaf along with opening information.
- Two problems:

1. The number of leaves is $|\mathbb{F}|$, which means the time to compute the commitment is at least $|\mathbb{F}|$.
   - Big problem when working over large fields (say, $|\mathbb{F}| \approx 2^{64}$ or $|\mathbb{F}| \approx 2^{128}$).
   - Want time proportional to the degree bound d.
2. V does not know if f has degree at most k!

Fixing the first problem (Want P time linear in degree, not field size)

- Rather than P Merkle-committing to **all** $(p-1)$ evaluations of q, P only Merkle-commits to evaluations $q(x)$ for those x in a carefully chosen **subset** Ω of $\mathbb{F}_p$.

Fixing the first problem (Want P time linear in degree, not field size)

- Rather than P Merkle-committing to **all** $(p-1)$ evaluations of q, P only Merkle-commits to evaluations $q(x)$ for those x in a carefully chosen **subset** Ω of $\mathbb{F}_p$.
- Ω has size $\rho^{-1}\, k$ for some constant $\rho \leq 1/2$, where k is the degree of q.
  - $\rho^{-1} \geq 2$ is called the "FRI blowup factor".
  - ρ is called the "rate of the Reed-Solomon code" used.

Fixing the first problem (Want P time linear in degree, not field size)

- Rather than P Merkle-committing to **all** $(p-1)$ evaluations of q, P only Merkle-commits to evaluations $q(x)$ for those x in a carefully chosen **subset** Ω of $\mathbb{F}_p$.
- Ω has size $\rho^{-1}\,k$ for some constant $\rho \leq 1/2$, where k is the degree of q.
  - $\rho^{-1} \geq 2$ is called the "FRI blowup factor".
- Strong tension between P time and verification costs:
  - The bigger the blowup factor, the slower P is, because it has to evaluate q on more inputs and Merkle-hash the results.
  - But the smaller the verification costs will be.

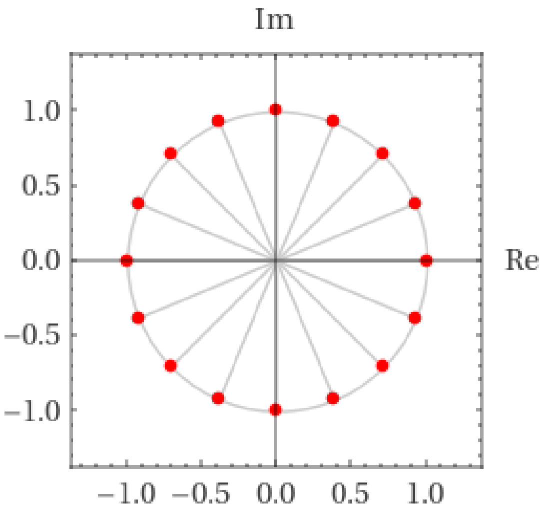
Fixing the first problem (Want P time linear in degree, not field size)

- Rather than P Merkle-committing to **all** $(p-1)$ evaluations of q, P only Merkle-commits to evaluations $q(x)$ for those x in a carefully chosen **subset** Ω of $\mathbb{F}_p$.
- Ω has size $\rho^{-1}\,k$ for some constant $\rho \leq 1/2$, where k is the degree of q.
  - $\rho^{-1} \geq 2$ is called the "FRI blowup factor".
- Strong tension between P time and verification costs:
  - The bigger the blowup factor, the slower P is, because it has to evaluate q on more inputs and Merkle-hash the results.
    - Proof length will be about $(\lambda/\log(\rho^{-1})) \cdot \log^2(k)$ hash values.
    - λ is the security parameter a.k.a. "λ bits of security" (more on this later)

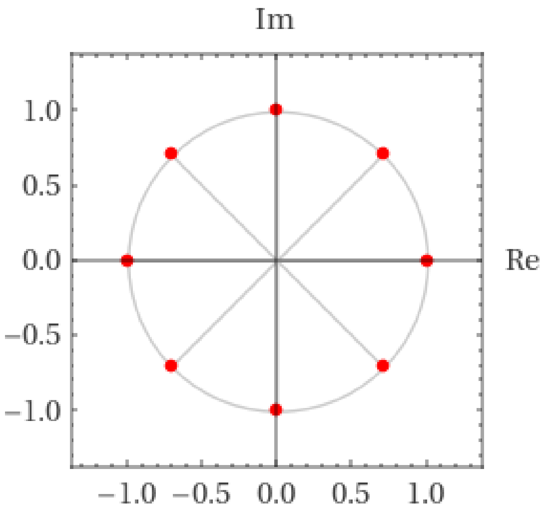
The key subset: roots of unity

- Let $n = \rho^{-1} k$. Assume n is a power of 2.
- The key subset Ω comprises all nth roots of unity in $\mathbb{F}_p$.
  - x such that $x^n = 1$. Equivalently, $x^n - 1 = 0$.

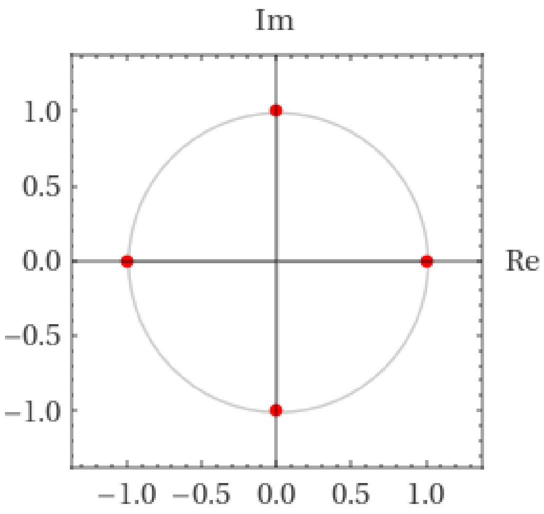
Roots of Unity visualized



16^{th} roots of unity



8th roots of unity



4th roots of unity

The key subset: roots of unity

- **Fact:** Let $\omega \in \mathbb{F}_p$ be a ***primitive*** n'th root of unity. That is, n is the smallest integer such that $\omega^n = 1$. Then $\Omega = \{ 1, \omega, \omega^2, \ldots, \omega^{n-1} \}$.

The key subset: roots of unity

- **Fact:** Let $\omega \in \mathbb{F}_p$ be a ***primitive*** n'th root of unity. That is, n is the smallest integer such that $\omega^n = 1$. Then $\Omega = \{ 1, \omega, \omega^2, \ldots, \omega^{n-1} \}$.
- **Fact:** Ω is a "multiplicative subgroup" of $\mathbb{F}_p$.
  - If x and y are both n'th roots of unity, then so is xy.
  - **Special case 1 (since $\boldsymbol{n}$ is even):** If x is a n'th root of unity, x^2 is a $(n/2)$'th root of unity.
  - **Special case 2 (since $\boldsymbol{n}$ is even):** if x is a n'th root of unity, so is $-x$.

The key subset: roots of unity

- **Fact:** Let $\omega \in \mathbb{F}_p$ be a ***primitive*** n'th root of unity. That is, n is the smallest integer such that $\omega^n = 1$. Then $\Omega = \{ 1, \omega, \omega^2, \ldots, \omega^{n-1} \}$.
- **Fact:** Ω is a "multiplicative subgroup" of $\mathbb{F}_p$.
  - If x and y are both n'th roots of unity, then so is xy.
  - **Special case 1 (since n is even):** If x is a n'th root of unity, x^2 is a $(n/2)$'th root of unity.
  - **Special case 2 (since n is even):** if x is a n'th root of unity, so is $-x$.
- **Fact:** Ω has size n if and only if n divides $p - 1$.

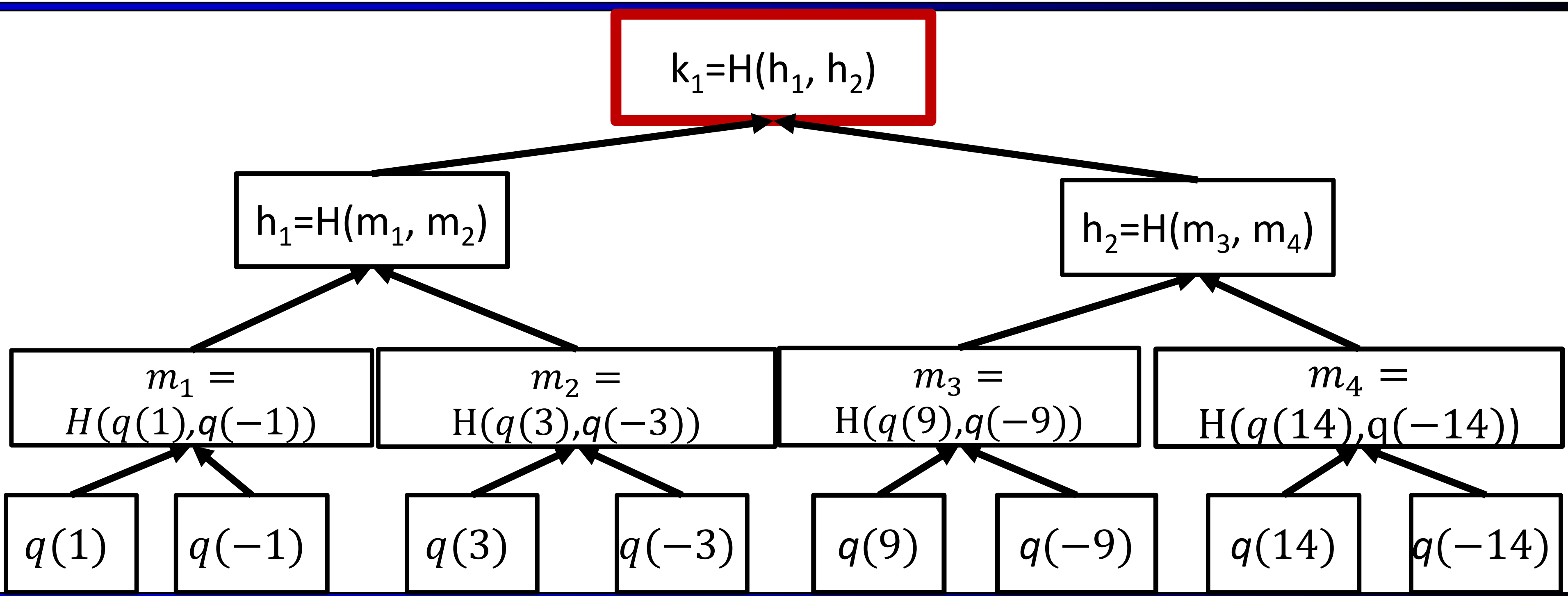
The key subset: roots of unity

- **Fact:** Let $\omega \in \mathbb{F}_p$ be a ***primitive*** n'th root of unity. That is, n is the smallest integer such that $\omega^n = 1$. Then $\Omega = \{ 1, \omega, \omega^2, \ldots, \omega^{n-1} \}$.
- **Fact:** Ω is a "multiplicative subgroup" of $\mathbb{F}_p$.
  - If x and y are both n'th roots of unity, then so is xy.
  - **Special case 1 (since n is even):** If x is a n'th root of unity, x^2 is a $(n/2)$'th root of unity.
  - **Special case 2 (since n is even):** if x is a n'th root of unity, so is $-x$.
- **Fact:** Ω has size n if and only if n divides $p - 1$.
  - This is why many FRI-based SNARKs work over fields like $\mathbb{F}_p$ with $p = 2^{64} - 2^{32} + 1$
    - $p - 1$ is divisible by 2^{32}.
    - Running FRI over the field can support any power-of-two value of n up to 2^{32}.

Roots of Unity: finite field example

- Consider the prime field $\mathbb{F}_{41}$ of size 41.
- 1^{st} roots of unity: {1}
- 2^{nd} roots of unity: {1, -1}
- 4^{th} roots of unity: {1, -1, 9, -9}.
- 8^{th} roots of unity: {1, -1, 9, -9, 3, -3, 14, -14}

FRI commitment to a univariate $q(X)$ in $\mathbb{F}_{41}[X]$ when $8 = \rho^{-1} k$



Fixing the second problem

- V needs to know that the committed vector is all evaluations over domain Ω of some degree-$(k-1)$ polynomial.
- Idea from the PCP literature: V "inspects" only a few entries of the vector to "get a sense" of whether it is low-degree.
  - Each query will add a Merkle-authentication path (i.e., $\log(n)$ hash values) to the proof.
- This turns out to be impractical.
  - Instead, the FRI "low-degree test" will be interactive.
  - The test will consist of a "folding phase" followed by a "query phase".
    - The folding phase is $\log(k)$ rounds. The query phase is one round.

The (interactive) low-degree test: Folding Phase

- Folding Phase:
  - ”Randomly fold the committed vector in half”.
    - This means pair up entries of the committed vector, have V pick a random field element r, and use r to “randomly combine” every two paired up entries.
  - This halves the length of the vector.
  - Have P Merkle-commit to the folded vector.

The (interactive) low-degree test: Folding Phase

- Folding Phase:
  - "Randomly fold the committed vector in half".
    - This means pair up entries of the committed vector, have V pick a random field element r, and use r to "randomly combine" every two paired up entries.
  - This halves the length of the vector.
  - Have P Merkle-commit to the folded vector.
  - The random combining technique is chosen so that the folded vector will have half the degree of the original vector.
  - Repeat the folding until the degree should fall to 0.
  - At this point, the length of the folded vector is still $\rho^{-1} \geq 2$. But since the degree should be 0, P can specify the folded vector with a single field element.

Folding phase (committed degree-3 polynomial in $\mathbb{F}_{41}[X]$ when $8 = 4\rho^{-1}$)

$q(1)$	$q(-1)$	$q(9)$	$q(-9)$	$q(3)$	$q(-3)$	$q(14)$	$q(-14)$

$$\frac{(r_1+1)}{2} q(1) + \frac{(r_1-1)}{-2} q(-1) := B(1)$$

$$\frac{(r_1+9)}{2\cdot 9} q(9) + \frac{(r_1-9)}{-2\cdot 9} q(-9) := B(-1)$$

$$\frac{(r_1+3)}{2\cdot 3} q(3) + \frac{(r_1-3)}{-2\cdot 3} q(-3) := B(9)$$

$$\frac{(r_1+14)}{2\cdot 14} q(14) + \frac{(r_1-14)}{-2\cdot 14} q(-14) := B(-9)$$

$$\frac{(r_2+1)}{2} B(1) + \frac{(r_2-1)}{-2} B(-1)$$

$$\frac{(r_2+9)}{2\cdot 9} B(9) + \frac{(r_2-9)}{-2\cdot 9} B(-9)$$

The (interactive) low-degree test: Query Phase

- P may have "lied" at some step of the folding phase, by not performing the fold correctly.
  - i.e., sending a vector that is **not** the prescribed folding of the previous vector.
  - To "artificially" reduce the degree of the (claimed) folded vector.
- V attempts to "detect" such inconsistencies during the query phase.

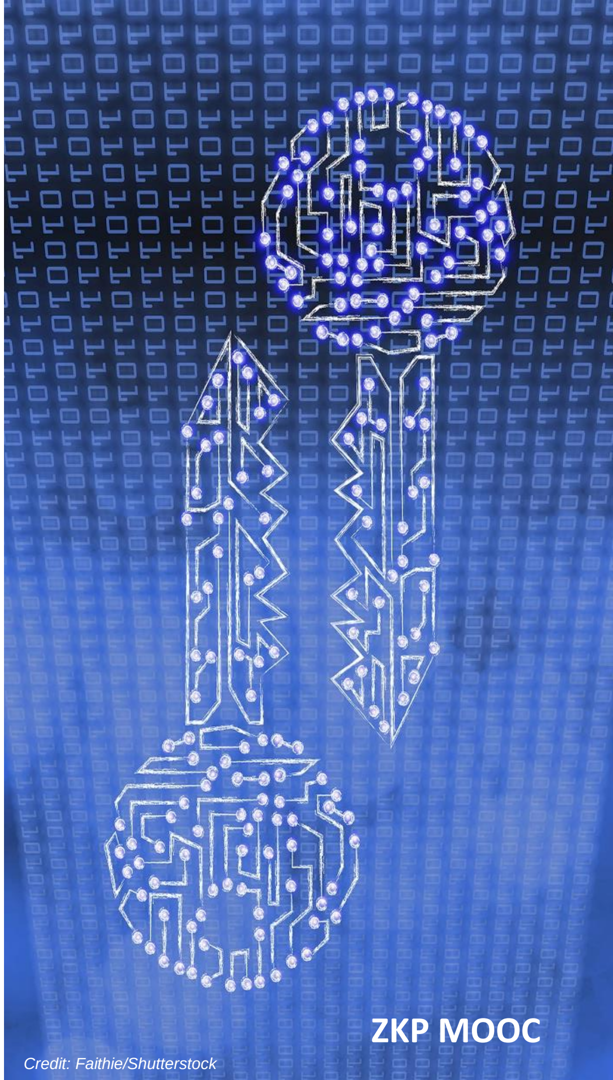
The (interactive) low-degree test: Query Phase

- P may have "lied" at some step of the folding phase, by not performing the fold correctly.
  - i.e., sending a vector that is **not** the prescribed folding of the previous vector.
  - To "artificially" reduce the degree of the (claimed) folded vector.
- V attempts to "detect" such inconsistencies during the query phase.
- Query phase: V picks about $(\lambda/\log(\rho^{-1}))$ entries of each folded vector and confirming each is the prescribed linear combination of the relevant two entries of the previous vector.

The (interactive) low-degree test: Query Phase

- P may have "lied" at some step of the folding phase, by not performing the fold correctly.
  - i.e., sending a vector that is **not** the prescribed folding of the previous vector.
  - To "artificially" reduce the degree of the (claimed) folded vector.
- V attempts to "detect" such inconsistencies during the query phase.
- Query phase: V picks about $(\lambda/\log(\rho^{-1}))$ entries of each folded vector and confirming each is the prescribed linear combination of the relevant two entries of the previous vector.
- **Proof length (and V time): roughly ($\boldsymbol{\lambda/\log(\rho^{-1})}$) $\boldsymbol{log(k)^2}$ hash evaluations.**

Back to the folding phase: more details



The (interactive) low-degree test: Folding Phase

- Split $q(X)$ into "even and odd parts" in the following sense.
  - $q(X) = q_e(X^2) + X\, q_o(X^2)$
  - E.g., if $q(X) = 1 + 2X + 3X^2 + 4X^3$.
    - Then $q_e(X) = 1 + 3X$ and $q_o(X) = 2 + 4X$.
    - Note that both q_e and q_o have (at most) half the degree of q.
- V picks a random field element r and sends r to P.
- The prescribed "folding" q is: $q_{fold}(Z) = q_e(Z) + r q_o(Z)$
- Clearly deg(q_{fold}) is half the degree of q itself.

The (interactive) low-degree test: Folding Phase

- Recall: $q(X) = q_e(X^2) + X\, q_o(X^2)$
- Recall: The prescribed "folding" q is: $q_{fold}(Z) = q_e(Z) + r q_o(Z)$.

The (interactive) low-degree test: Folding Phase

- Recall: $q(X) = q_e(X^2) + X\, q_o(X^2)$
- Recall: The prescribed "folding" q is: $q_{fold}(Z) = q_e(Z) + r q_o(Z)$.
- Fact: Let x and $-x$ be n'th roots of unity and $z = x^2$. Then:

$$q_{fold}(\mathrm{z}) = \frac{(r+x)}{2x} q(x) + \frac{(r-x)}{-2x} q(-x).$$

The (interactive) low-degree test: Folding Phase

- Recall: $q(X) = \ q_e(X^2) + X\, q_o(X^2)$
- Recall: The prescribed "folding" q is: $q_{fold}(Z) = \ q_e(Z) + rq_o(Z)$.
- Fact: Let x and $-x$ be n'th roots of unity and $z = x^2$. Then:

$$q_{fold}(\mathrm{z}) = \frac{(r+x)}{2x} q(x) + \frac{(r-x)}{-2x} q(-x).$$

- Proof: Clearly $q(x) = \ q_e(z) + xq_o(z)$.
- In other words, if $r = x$ then $q_{fold}(\mathrm{z})= q(x)$.
- Similarly, if $r = -x$ then $q_{fold}(\mathrm{z}) = q(-x)$.

The (interactive) low-degree test: Folding Phase

- Recall: $q(X) = q_e(X^2) + X\, q_o(X^2)$
- Recall: The prescribed "folding" q is: $q_{fold}(Z) = q_e(Z) + r q_o(Z)$.
- Fact: Let x and $-x$ be n'th roots of unity and $z = x^2$. Then:

$$q_{fold}(\mathrm{z}) = \frac{(r+x)}{2x} q(x) + \frac{(r-x)}{-2x} q(-x).$$

- Proof: Clearly $q(x) = q_e(z) + x q_o(z)$.
- In other words, if $r = x$ then $q_{fold}(\mathrm{z})= q(x)$.
- Similarly, if $r = -x$ then $q_{fold}(\mathrm{z}) = q(-x)$.
- The fact follows because it gives a degree-1 function of r with exactly this behavior at $r = -x$ and $r = x$, and any two degree-1 functions of r that agree at two or more inputs must be the same function.

Folding phase (committed degree-3 polynomial in $\mathbb{F}_{41}[X]$ when $8 = 4\rho^{-1}$)

$q(1)$	$q(-1)$	$q(9)$	$q(-9)$	$q(3)$	$q(-3)$	$q(14)$	$q(-14)$

$\frac{(r_1+1)}{2}q(1)+\frac{(r_1-1)}{-2}q(-1) := B(1)$	$\frac{(r_1+9)}{2\cdot 9}q(9)+\frac{(r_1-9)}{-2\cdot 9}q(-9) := B(-1)$	$\frac{(r_1+3)}{2\cdot 3}q(3)+\frac{(r_1-3)}{-2\cdot 3}q(-3) := B(9)$	$\frac{(r_1+14)}{2\cdot 14}q(14)+\frac{(r_1-14)}{-2\cdot 14}q(-14) := B(-9)$

$\frac{(r_2+1)}{2}B(1)+\frac{(r_2-1)}{-2}B(-1)$	$\frac{(r_1+9)}{2\cdot 9}B(9)+\frac{(r_1-9)}{-2\cdot 9}B(-9)$

The (interactive) low-degree test: Folding Phase

- Recall: $q(X) = q_e(X^2) + X\, q_o(X^2)$
- Recall: The prescribed "folding" q is: $q_{fold}(Z) = q_e(Z) + r q_o(Z)$.
- The fact that the map $x \longmapsto x^2$ is 2-to-1 on $\Omega = \{ 1, \omega, \omega^2, \ldots, \omega^{n-1} \}$ ensures that the relevant domain halves in size with each fold.
  - Other domains, like $\{0, 1, 2, \ldots\ n - 1\}$, don't have this property.

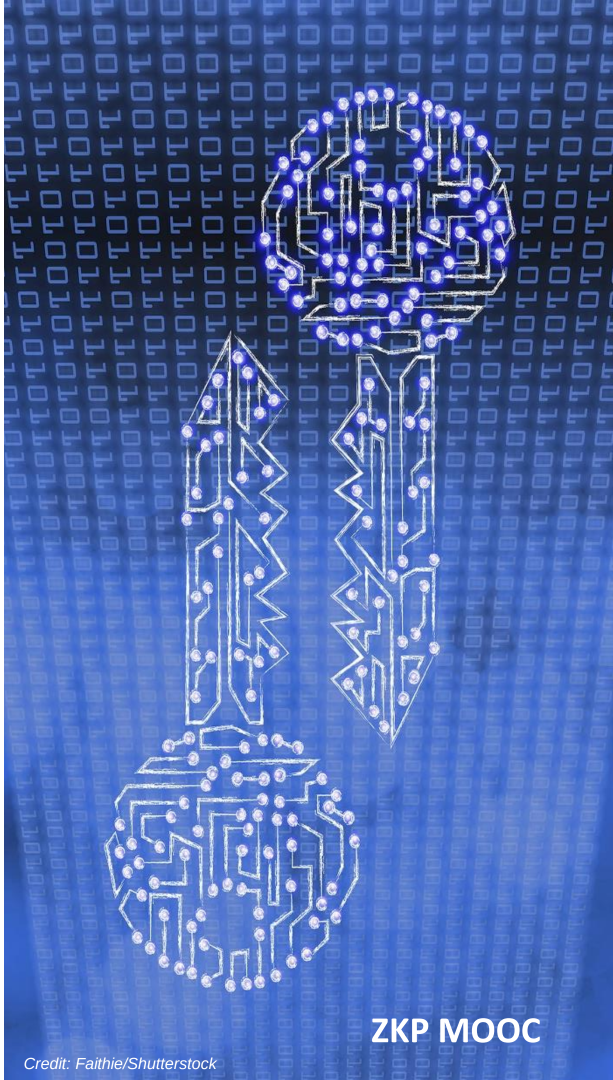
Compare to Lecture 7

- Lecture 7 covered a variety of polynomial commitments (Ligero, Brakedown, Orion) that are similar to FRI.
  - All use error-correcting codes.
  - The only cryptography used is hashing (Merkle-hashing + Fiat-Shamir).

Compare to Lecture 7

- Lecture 7 covered a variety of polynomial commitments (Ligero, Brakedown, Orion) that are similar to FRI.
  - All use error-correcting codes.
  - The only cryptography used is hashing (Merkle-hashing + Fiat-Shamir).
- The Lecture 7 schemes viewed a degree-d polynomial as $d^{1/2}$ vectors each of length about $d^{1/2}$ and performed "a single random fold on all these vectors".
  - This resulted in larger proofs (size roughly $d^{1/2}$), but some advantages (e.g., linear-time prover, field-agnostic).
  - Proof size can be reduced via SNARK composition (will be discussed in Lecture 10).
- FRI views a degree-d polynomial as a single vector of length $O(d)$ and "randomly folds it in half" logarithmically many times.

Sketch of the security analysis



The security analysis

- Recall: at the start of the FRI polynomial commitment, P Merkle-commits to a vector w claimed to equal q's evaluations over Ω.
  - Here, Ω is the set of n'th roots of unity in $\mathbb{F}_p$, where $n = \rho^{-1} k$.
  - And q is **claimed to** have degree less than k.

The security analysis

- Let δ be the "relative Hamming distance" of q from the closest polynomial h of degree $k-1$.
  - δ is the fraction of $x \in \Omega$ such that $h(x) \neq q(x)$.

The security analysis

- Let δ be the "relative Hamming distance" of q from the closest polynomial h of degree $k-1$.
  - δ is the fraction of $x \in \Omega$ such that $h(x) \neq q(x)$.
- Claim: P "passes" all t "FRI verifier queries" with probability at most $\frac{k}{p} + (1-\delta)^t$.

f.

The security analysis

- Let δ be the "relative Hamming distance" of q from the closest polynomial h of degree $k - 1$.
  - δ is the fraction of $x \in \Omega$ such that $h(x) \neq q(x)$.
- Claim: P "passes" all t "FRI verifier queries" with probability at most $\frac{k}{p} + (1 - \delta)^t$.
  - Caveat: this is only known to hold for δ up to $1 - \rho^{1/2}$, but is conjectured to hold for δ up to $1 - \rho$.
  - Most FRI deployments' security are analyzed under this conjecture.
  - Informal interpretation: FRI V accepts with probability at most about $(1 - (1 - \rho))^t = \rho^t$.
  - In other words, each of the t queries contributes about $\mathrm{Log2}(1/\rho)$ "bits of security".

The security analysis

- Let δ be the "relative Hamming distance" of q from the closest polynomial h of degree $k-1$.
  - δ is the fraction of $x \in \Omega$ such that $h(x) \neq q(x)$.
- Claim: P "passes" all t "FRI verifier queries" with probability at most $\frac{k}{p} + (1-\delta)^t$.
  - Caveat: this is only known to hold for δ up to $1 - \rho^{1/2}$, but is conjectured to hold for δ up to $1 - \rho$.
  - Most FRI deployments' security are analyzed under this conjecture.
  - Informal interpretation: FRI V accepts with probability at most about $(1-(1-\rho))^t = \rho^t$.
  - In other words, each of the t queries contributes about $\mathrm{Log2}(1/\rho)$ "bits of security".
  - E.g., if $\rho = \frac{1}{4}$, each FRI verifier queries contributes about 2 bits of security.
    - At the cost of roughly $\log(\mathrm{n})^2$ hash values included in the proof.

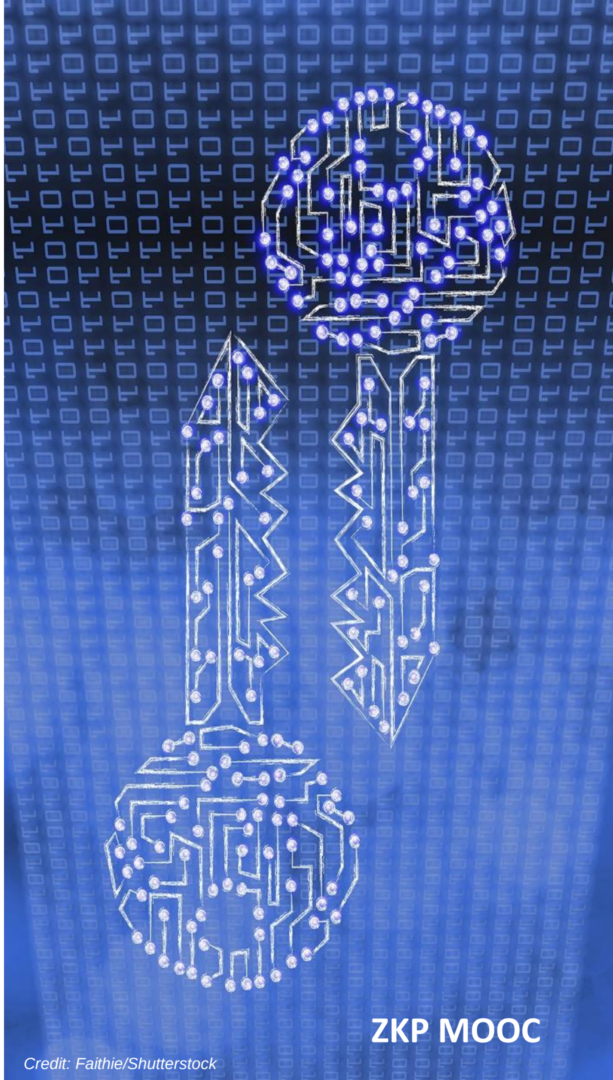
The security analysis

- Let δ be the "relative Hamming distance" of q from the closest polynomial h of degree $k-1$.
  - δ is the fraction of $x \in \Omega$ such that $h(x) \neq q(x)$.
- Claim: P "passes" all t "FRI verifier queries" with probability at most $\frac{k}{p} + (1-\delta)^t$.
  - Recall: $q_{fold}(Z) = q_e(Z) + r q_o(Z)$.
  - Can check: since q is δ-far from every degree-$(k-1)$ polynomial h, at least one of q_e or q_o must be δ-far from every degree-$(k/2-1)$ polynomial over the $(n/2)$-roots of unity.
  - Idea: A "random linear combination" of two functions, at least one of which is δ-far from degree-*d* polynomials, will also be is δ-far from degree-*d* with overwhelming probability.
  - The $\frac{k}{p}$ term bounds the probability that P "gets a lucky fold".
    - q_{fold} is close to degree-$(k/2-1)$ even though q is not close to degree-(k-1).

The security analysis

- Let δ be the "relative Hamming distance" of q from the closest polynomial h of degree $k-1$.
  - δ is the fraction of $x \in \Omega$ such that $h(\mathrm{x}) \neq q(x)$.
- Claim: P "passes" all t "FRI verifier queries" with probability at most $\frac{k}{p} + (1-\delta)^t$.
  - Idea 2: If P does "**not** get a lucky fold", then the "true" final folded function is δ-far from any degree-0 function.
  - But P is forced to send a degree-0 function as the final fold.
  - So at least one "fold" is done dishonestly by P.
  - In this case, each "FRI verifier query" detects a discrepancy in a fold with probability at least δ.
  - So **all** FRI verifier queries **fail** to detect the discrepancy with probability at most $(1-\delta)^t$.

The Known Attack on FRI



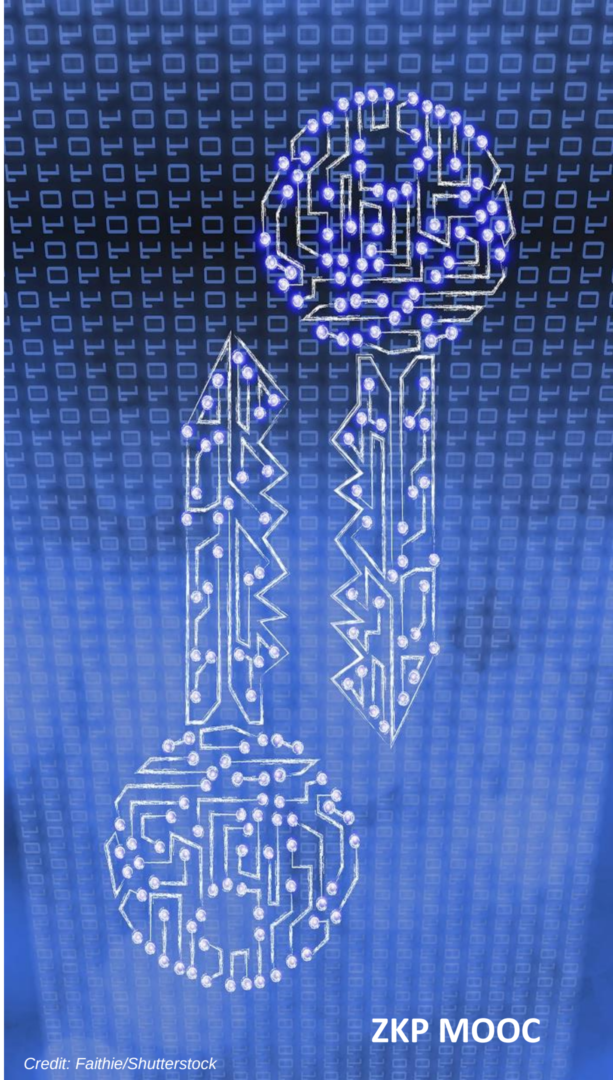
The known attack

- Recall: at the start of the FRI polynomial commitment, P Merkle-commits to a vector w claimed to equal q's evaluations over Ω.
  - Here, Ω is the set of n'th roots of unity in $\mathbb{F}_p$, where $n = \rho^{-1}\, k$.
  - And q is **claimed to** have degree less than k.
  - The following P strategy works for **any** q (even ones maximally far from degree-k) and passes **all** FRI verifier checks with probability ρ^t.

The known attack

- Recall: at the start of the FRI polynomial commitment, P Merkle-commits to a vector w claimed to equal q's evaluations over Ω.
  - Here, Ω is the set of n'th roots of unity in $\mathbb{F}_p$, where $n = \rho^{-1} k$.
  - And q is **claimed to** have degree less than k.
  - The following P strategy works for **any** q (even ones maximally far from degree-k) and passes **all** FRI verifier checks with probability ρ^t.
  - P picks a set T of $k = \rho n$ elements of Ω and computes a polynomial s of degree $k - 1$ that agrees with q at those points.
  - P folds s rather than q during the folding phase.
  - All t FRI verifier queries lie in T with probability ρ^t.

Polynomial Commitment from FRI



Credit: Faithie/Shutterstock

Recall: Initial Attempt from Lecture 4

- P Merkle-commits to all evaluations of the polynomial q.
- When V requests $q(r)$, P reveals the associated leaf along with opening information.
- New Problems with FRI:
  - P has only Merkle-committed to evaluations of q over domain Ω, not the whole field.
  - V only knows that q is ”not too far” from low-degree, not exactly low-degree.

A fix for both problems

- Recall the following FACT used in KZG commitments:
  - FACT: For any degree-d univariate polynomial q, the assertion "$q(r) = v$" is equivalent to the existence of a polynomial w of degree at most d such that
    - $q(X) - v = w(X)(X - r)$.
  - **So to confirm that $q(r) = v$, V applies FRI's fold+query procedure to the function $(q(X) - v)\ (X - r)^{-1}$ using degree bound $d - 1$.**

A fix for both problems

- Recall the following FACT used in KZG commitments:
  - FACT: For any degree-*d* univariate polynomial q, the assertion "$q(r) = v$" is equivalent to the existence of a polynomial w of degree at most d such that
    - $q(X) - v = w(X)(X - r)$.
  - **So to confirm that $\boldsymbol{q(r) = v}$, V applies FRI's fold+query procedure to the function $\boldsymbol{(q(X) - v)\,(X - r)^{-1}}$ using degree bound $\boldsymbol{d - 1}$.**
    - Whenever the FRI verifier queries this function at a point in Ω, the evaluation can be obtained with one query to q at the same point.

A fix for both problems

- Recall the following FACT used in KZG commitments:
  - FACT: For any degree-d univariate polynomial q, the assertion "$q(r) = v$" is equivalent to the existence of a polynomial w of degree at most d such that
    - $q(X) - v = w(X)(X - r)$.
  - **So to confirm that $\boldsymbol{q(r) = v}$, V applies FRI's fold+query procedure to the function $\boldsymbol{(q(X) - v)\,(X - r)^{-1}}$ using degree bound $\boldsymbol{d - 1}$.**
    - Whenever the FRI verifier queries this function at a point in Ω, the evaluation can be obtained with one query to q at the same point.
  - Can show: To pass V's checks in this polynomial commitment with noticeable probability, v has to equal $h(r)$, where h is the degree-d polynomial that is closest to q.

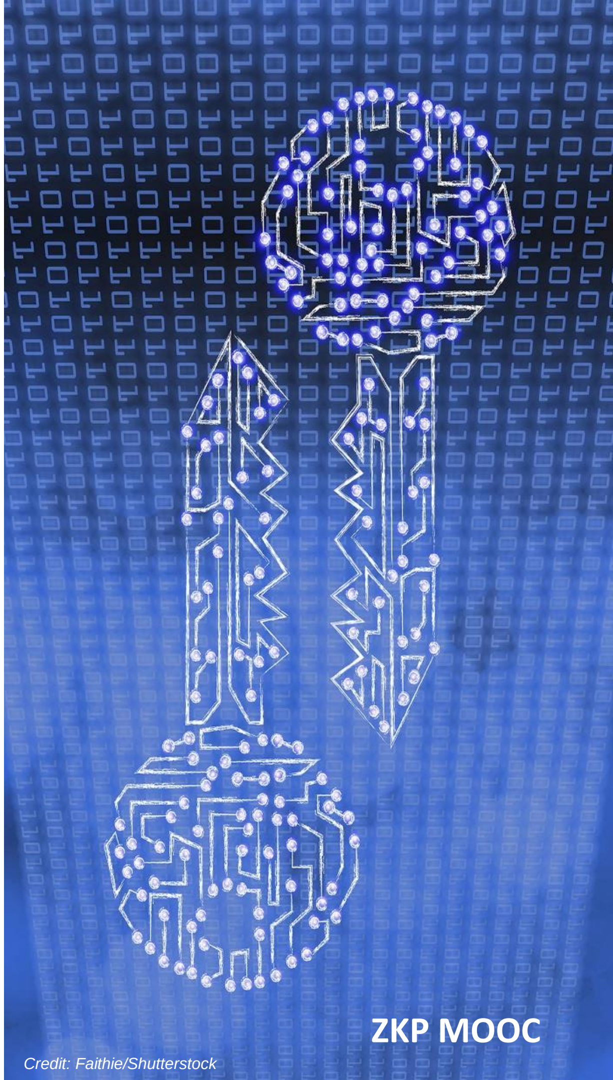
A fix for both problems

- Recall the following FACT used in KZG commitments:
  - FACT: For any degree-*d* univariate polynomial q, the assertion "$q(r) = v$" is equivalent to the existence of a polynomial w of degree at most d such that
    - $q(X) - v = w(X)(X - r)$.
  - **So to confirm that $\boldsymbol{q(r) = v}$, V applies FRI's fold+query procedure to the function $\boldsymbol{(q(X) - v)\,(X - r)^{-1}}$ using degree bound $\boldsymbol{d - 1}$.**
    - Whenever the FRI verifier queries this function at a point in Ω, the evaluation can be obtained with one query to q at the same point.
  - Caveat: The security analysis requires δ to be (at most) $(1 - \rho)/2$. Each FRI verifier queries brings (less than) 1 bit of security, not $\log 2(1/\rho)$ bits.

A fix for both problems

- Recall the following FACT used in KZG commitments:
  - FACT: For any degree-d univariate polynomial q, the assertion "$q(r) = v$" is equivalent to the existence of a polynomial w of degree at most d such that
    - $q(X) - v = w(X)(X - r)$.
  - **So to confirm that $\boldsymbol{q(r) = v}$, V applies FRI's fold+query procedure to the function $\boldsymbol{(q(X) - v)\,(X - r)^{-1}}$ using degree bound $\boldsymbol{d - 1}$.**
    - Whenever the FRI verifier queries this function at a point in Ω, the evaluation can be obtained with one query to q at the same point.
  - People are using FRI today as a weaker primitive than a polynomial commitment, which still suffices for SNARK security.
    - P is bound to a "small set" of low-degree polynomials rather than to a single one.

The Fiat-Shamir Transformation and Concrete Security



Recall: Fiat-Shamir transformation

Public-Coin
Interactive Protocol

$\boldsymbol{P} \xrightarrow{\alpha} \boldsymbol{V}$

$\boldsymbol{P} \xleftarrow{\beta} \boldsymbol{V}$

$\boldsymbol{P} \xrightarrow{\gamma} \boldsymbol{V}$

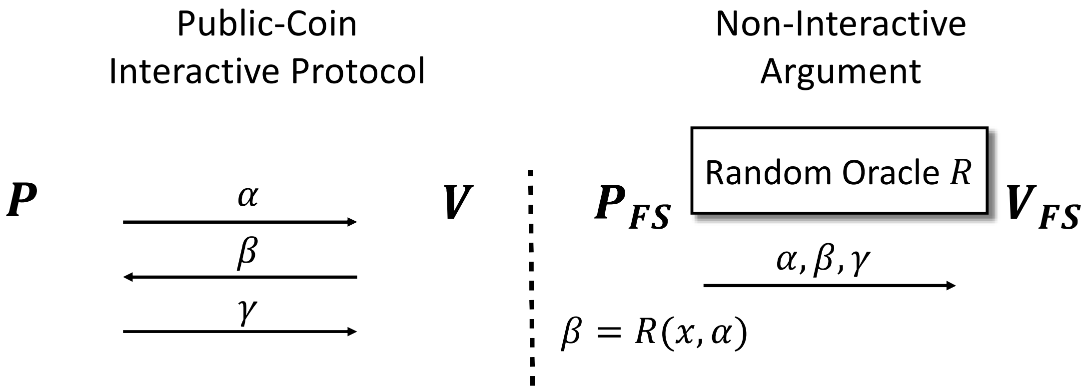
Non-Interactive
Argument

$\boxed{\text{Random Oracle } R}$

$\boldsymbol{P_{FS}} \xrightarrow{\alpha, \beta, \gamma} \boldsymbol{V_{FS}}$

$\beta = R(x, \alpha)$

Recall: Fiat-Shamir transformation



Grinding attack on Fiat-Shamir:

- P_{FS} iterates over first-messages α until it finds one such that $R(x, \alpha)$ is "lucky"

Recall: Fiat-Shamir transformation

Grinding attack on Fiat-Shamir:

- P_{FS} iterates over first-messages α until it finds one such that $R(x, \alpha)$ is "lucky"

Recall: Fiat-Shamir transformation

Grinding attack on Fiat-Shamir:

- P_{FS} iterates over first-messages α until it finds one such that $R(x, \alpha)$ is "lucky"

- Example: Suppose you apply Fiat-Shamir to an interactive protocol with 80 bits of statistical security (soundness error 2^{-80}).
  - With 2^b hash evaluations, grinding attack will succeed with probability 2^{-80+b}.
    - E.g., with 2^{70} hashes, successfully attack with probability about 2^{-10}.

Recall: Fiat-Shamir transformation

Grinding attack on Fiat-Shamir:

- P_{FS} iterates over first-messages α until it finds one such that $R(x, \alpha)$ is "lucky"

- Example: Suppose you apply Fiat-Shamir to an interactive protocol with 80 bits of statistical security (soundness error 2^{-80}).
  - With 2^b hash evaluations, grinding attack will succeed with probability 2^{-80+b}.
    - E.g., with 2^{70} hashes, successfully attack with probability about 2^{-10}.

Comparison:

For a collision-resistant hash function (CRHF) configured to 80 bits of security, the fastest collision-finding procedure should be a **birthday attack.**

Recall: Fiat-Shamir transformation

Grinding attack on Fiat-Shamir:

- P_{FS} iterates over first-messages α until it finds one such that $R(x, \alpha)$ is "lucky"

- Example: Suppose you apply Fiat-Shamir to an interactive protocol with 80 bits of statistical security (soundness error 2^{-80}).
  - With 2^b hash evaluations, grinding attack will succeed with probability 2^{-80+b}.
    - E.g., with 2^{70} hashes, successfully attack with probability about 2^{-10}.

Comparison:

With 2^k hash evaluations, finds a collision with a probability of only 2^{2k-160}.
For example, 2^{70} hash evaluations will yield a collision with a probability of $\mathbf{2^{-20}}$.

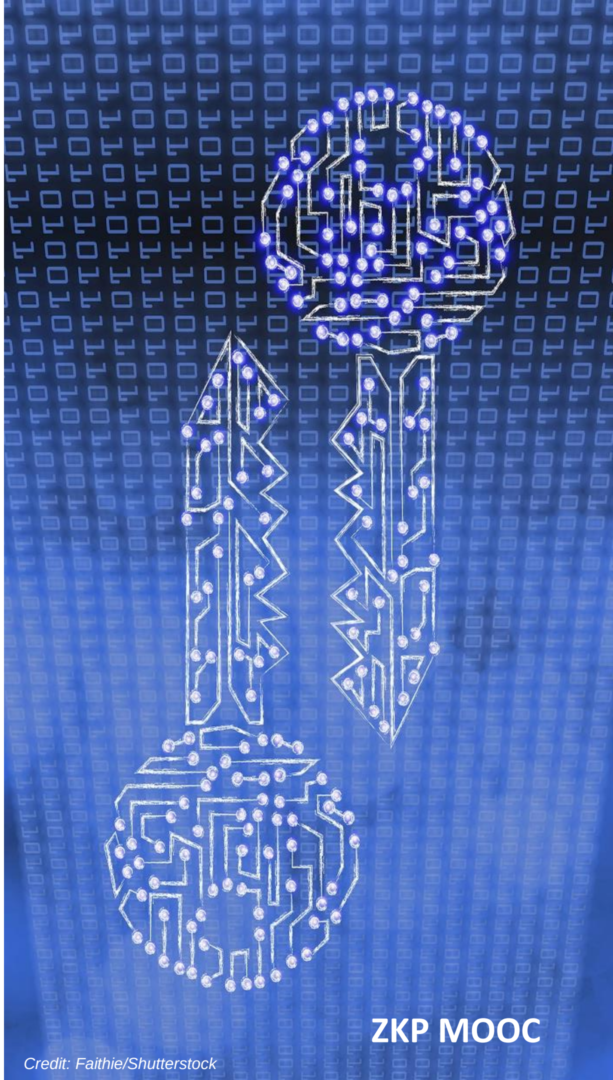
How many hashes are feasible today?

1. Today, the bitcoin network performs 2^{80} SHA-256 hashes roughly every hour.
   - At current prices, those hashes typically earn **less than $1 million** worth of block rewards.

How many hashes are feasible today?

1. Today, the bitcoin network performs 2^{80} SHA-256 hashes roughly every hour.
   - At current prices, those hashes typically earn **less than $1 million** worth of block rewards.
2. In January 2020, the cost of computing just shy of 2^{64} SHA-1 evaluations using GPUs was $45,000.
   - This puts 2^{70} hashes at about $3,000,000.
   - Likely less today, post-Ethereum-merge.

Interactive vs. Non-Interactive Security



Interactive Security

- A polynomial commitment scheme such as FRI, **when run interactively at "λ bits of security"**, has the following security guarantee
  - Assuming P cannot find a collision in the hash function used to build Merkle trees, a lying P cannot pass the verifier's checks with probability better than $2^{-\lambda}$.
  - **A lying P must actually interact with V to learn V's challenges, in order to find out if it receives a "lucky" challenge!**

Interactive Security

- For example, if $\lambda = 60$, then with probability at least 1- 2^{-30}, V will reject (at least) 2^{30} times before a lying P succeeds in convincing V to accept.
  - It seems unlikely that V would continue interacting with a P that has been caught in a lie 2^{30} times.
  - In many settings, interactive with V may take long enough that P wouldn't have time to make 1 billion attempts even if V were willing to consider each one.
    - E.g., One billion Ethereum blocks take 3 years to create (at one block per 12 seconds).

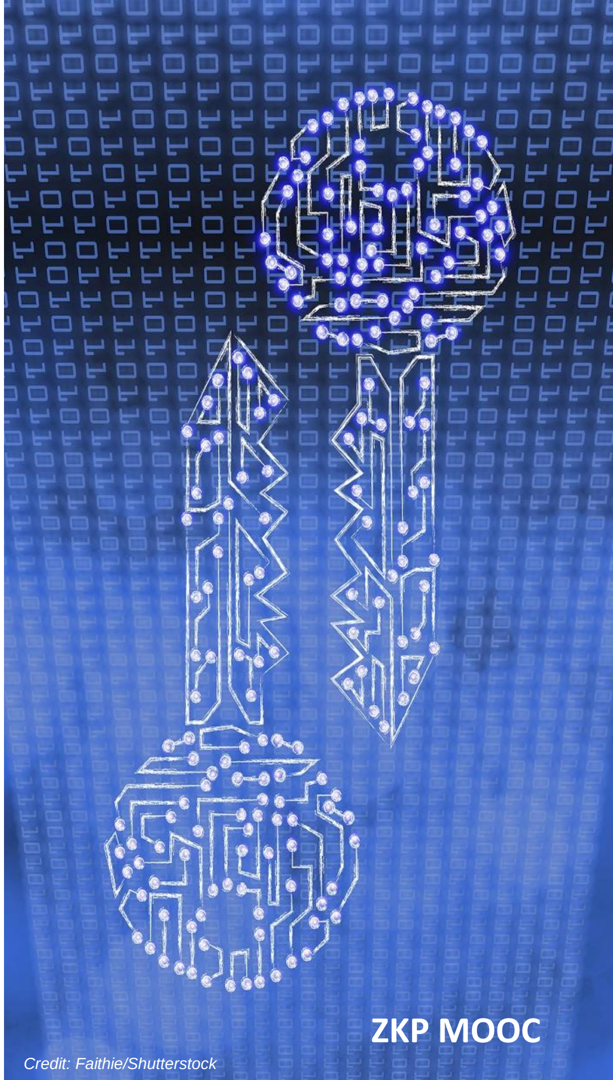
Non-interactive security

- Suppose Fiat-Shamir is applied to an interactive protocol such as FRI that was run at λ bits of interactive security.
  - The resulting **non-interactive** protocol has the following much weaker guarantee:
  - A lying P willing to perform 2^k hash evaluations can successfully attack the protocol with probability $2^{k-\lambda}$.
  - **A lying P can attempt the attack "silently".**
    - Unlike in the interactive case, P can perform a "grinding attack" without interacting with V until P receives a lucky challenge.

Non-interactive security

- Suppose Fiat-Shamir is applied to an interactive protocol such as FRI that was run at λ bits of interactive security.
  - The resulting **non-interactive** protocol has the following much weaker guarantee:
  - A lying P willing to perform 2^k hash evaluations can successfully attack the protocol with probability $2^{k-\lambda}$.
  - **A lying P can attempt the attack "silently".**
    - Unlike in the interactive case, P can perform a ”grinding attack” without interacting with V until P receives a lucky challenge.
  - Higher security levels λ are necessary in this setting.
    - 60 bits of interactive security is fine in many contexts.
    - 60 bits of non-interactive security is not okay unless the payoff of a successful attack is minimal.

Fiat-Shamir security loss for many-round protocols can be huge



An interactive protocol

- Consider the following (silly) interactive protocol for the empty language (i.e., V **should** always reject).
- P sends a message (a nonce) which V ignores.
- V tosses a random coin, rejecting if it comes up heads and accepting if it comes up tails.
- The soundness error of this protocol is 1/2.
- If you sequentially repeat it λ times and accept only if every run accepts, the soundness error falls to $1/2^{\lambda}$.

Fiat-Shamir-ing this interactive protocol is **insecure**

- Recall: If you sequentially repeat it λ times and accept only if every run accepts, the soundness error falls to $1/2^{\lambda}$.
  - Consider Fiat-Shamir-ing this λ-round protocol to render it non-interactive.
  - A cheating prover P_{FS} can find a convincing "proof" for the non-interactive protocol with $O(\lambda)$ hash evaluations.

Fiat-Shamir-ing this interactive protocol is **insecure**

- Recall: If you sequentially repeat it λ times and accept only if every run accepts, the soundness error falls to $1/2^{\lambda}$.
  - Consider Fiat-Shamir-ing this λ-round protocol to render it non-interactive.
  - A cheating prover P_{FS} can find a convincing "proof" for the non-interactive protocol with $O(\lambda)$ hash evaluations.
  - Idea: P_{FS} grinds on the first repetition alone (i.e., iterate over nonces in the first repetition until one is found that hashes to tails. This requires 2 attempts in expectation until success.) Fix this first nonce m_1 for the remainder of the attack.

Fiat-Shamir-ing this interactive protocol is **insecure**

- Recall: If you sequentially repeat it λ times and accept only if every run accepts, the soundness error falls to $1/2^{\lambda}$.
  - Consider Fiat-Shamir-ing this λ-round protocol to render it non-interactive.
  - A cheating prover P_{FS} can find a convincing "proof" for the non-interactive protocol with $O(\lambda)$ hash evaluations.
  - Idea: P_{FS} grinds on the first repetition alone (i.e., iterate over nonces in the first repetition until one is found that hashes to tails. This requires 2 attempts in expectation until success.) Fix this first nonce m_1 for the remainder of the attack.
  - Then P_{FS} grinds on the second repetition alone until it finds an m_2 such that (m_1, m_2) hashes to tails. Fix m_2 for the remainder of the attack.
  - Then P_{FS} grinds on the third repetition, and so on.

The takeaway

- Applying Fiat-Shamir to a many-round interactive protocol can lead to a huge loss in security, whereby the resulting non-interactive protocol is totally insecure.

The takeaway

- Applying Fiat-Shamir to a many-round interactive protocol can lead to a huge loss in security, whereby the resulting non-interactive protocol is totally insecure.
- Fortunately, this security loss can be ruled out if the interactive protocol satisfies a stronger notion of soundness called *round-by-round* soundness.

The takeaway

- Applying Fiat-Shamir to a many-round interactive protocol can lead to a huge loss in security, whereby the resulting non-interactive protocol is totally insecure.
- Fortunately, this security loss can be ruled out if the interactive protocol satisfies a stronger notion of soundness called *round-by-round* soundness.
  - This means an attacker in the interactive protocol has to "get very lucky all at once" (in a single round)… it can't succeed by getting "a little bit lucky many times".
  - The sequential repetition of soundness error 1/2 is **not** round-by-round sound.
    - The attacker can "get a little lucky" each round and succeed (i.e., in each round with probability 1/2 it gets the "lucky" challenge Tails each round).
  - The sum-check protocol (Lecture 4) is an example of a logarithmic-round protocol that **is** known to be round-by-round sound.
  - Something analogous is known for Bulletproofs [AFK22, Wik21].

The takeaway

- Applying Fiat-Shamir to a many-round interactive protocol can lead to a huge loss in security, whereby the resulting non-interactive protocol is totally insecure.
- Fortunately, this security loss can be ruled out if the interactive protocol satisfies a stronger notion of soundness called *round-by-round* soundness.
- FRI is a logarithmic-round interactive protocol that is always deployed non-interactively today.
  - It has **not** been shown to be round-by-round sound.

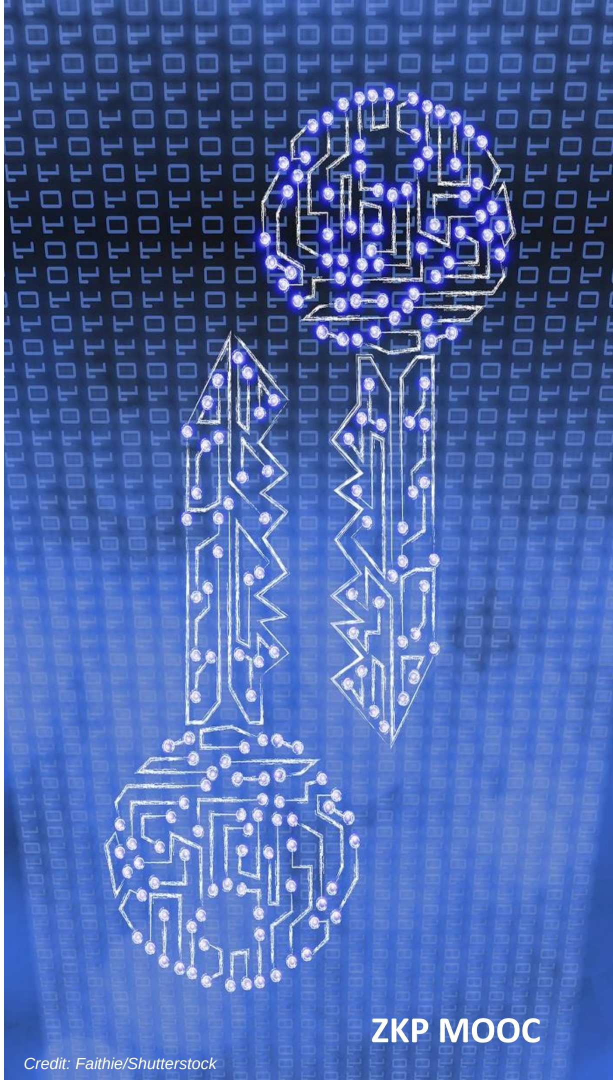
The takeaway

- Applying Fiat-Shamir to a many-round interactive protocol can lead to a huge loss in security, whereby the resulting non-interactive protocol is totally insecure.
- Fortunately, this security loss can be ruled out if the interactive protocol satisfies a stronger notion of soundness called *round-by-round* soundness.
- FRI is a logarithmic-round interactive protocol that is always deployed non-interactively today.
  - It has **not** been shown to be round-by-round sound.
- SNARK designers applying Fiat-Shamir to interactive protocols with more than 3 messages should show that the protocol is round-by-round sound if they want to rule out a major security loss.

END OF LECTURE

Next lecture:

SNARKs from Linear PCPs
(e.g., Groth16)



Example: Reed-Solomon encoding of a vector over $\mathbb{F}_{11}$.

a
2
1
1

$$a \xrightarrow{q_a(X) = 2 + X + X^2}$$

2	$q_a(0)$	0	$q_a(6)$
1	$q_a(1)$	5	$q_a(7)$
1	$q_a(2)$	0	$q_a(8)$
2	$q_a(3)$	7	$q_a(9)$
4	$q_a(4)$	4	$q_a(10)$
7	$q_a(5)$		

FRI (citation)

1. Recall from Lecture 5: n'th roots of unity

Let $\omega \in \mathbb{F}_p$ be a primitive k-th root of unity (so that $\omega^k = 1$).

- if $\Omega = \{1, \omega, \omega^2, \ldots, \omega^{k\text{-}1}\} \subseteq \mathbb{F}_p$ then $Z_\Omega(X) = X^k - 1$