Theory and Practice of Succinct Zero Knowledge Proofs

Lecture 02: Modern zkSNARK Constructions
Succinct Non-Interactive Arguments (SNARGs)

\[ \tilde{O}(\log(F)) \]

Prover
- \( \mathbf{pk} \): proving key
- \( \mathbf{x} \): public input
- \( \mathbf{w} \): private witness

Verifier
- \( \mathbf{vk} \): verifying key
- \( \mathbf{x} \): public input

I know \( \mathbf{w} \) s.t. \( F(\mathbf{x}, \mathbf{w}) = 1 \)

\[
\text{Mic94, Groth10, GGPR13, Groth16...}
..., \text{GWC19, CHMMVW20, ...}
\]
Succinct Non-Interactive Arguments (SNARGs)

• **Completeness**: If \((F, x, w) \in \mathcal{R}\),
  \[
  \Pr \left[ V(vk, x, \pi) = 1 : (pk, vk) \leftarrow \text{Setup}(F), \pi \leftarrow P(pk, x, w) \right] = 1.
  \]

• **Soundness**: If \((F, x, w) \notin \mathcal{R}\), for all efficient provers \(\tilde{P}\)
  \[
  \Pr \left[ V(vk, x, \pi) = 1 : (pk, vk) \leftarrow \text{Setup}(F), \pi \leftarrow \tilde{P}(pk, x) \right] \approx 0
  \]

• **Succinctness**: \(|\pi| = O(\log |F|)\)
What if there’s always a witness?

**Soundness**: If \((F, x, w) \not\in \mathcal{R}\), then for all efficient provers \(\tilde{P}\)

\[
\Pr \left[ V(vk, x, \pi) = 1 : (pk, vk) \leftarrow \text{Setup}(F), \pi \leftarrow \tilde{P}(pk, x) \right] \approx 0
\]

- \(F(x, w) := \text{SHA2}(w) \overset{?}{=} x\) : there is always a preimage!
- \(F((m, pk), \sigma) := \text{VerifySignature}(pk, m, \sigma) \overset{?}{=} 1\) : if \(pk\) is a valid public key, there is always a valid signature!
- Generally many examples where **witness always exists!**
SNARGs of Knowledge (SNARKs)

- **Completeness**: For all \((F, x, w) \in \mathcal{R}\),
  \[
  \Pr \left[ V(vk, x, \pi) = 1 : (pk, vk) \leftarrow \text{Setup}(F), \pi \leftarrow P(pk, x, w) \right] = 1.
  \]

- **Knowledge Soundness**: If \(V(vk, x, \pi) = 1\), then \(\tilde{P}\) “knows” \(w\) such that \((F, x, w) \in \mathcal{R}\)

- **Succinctness**: \(|\pi| = O(\log |F|)\)
SNARGs of Knowledge (SNARKs)

- **Completeness**: For all $(F, x, w) \in \mathcal{R}$,
  \[
  \Pr \left[ V(vk, x, \pi) = 1 : (pk, vk) \leftarrow \text{Setup}(F), \quad \pi \leftarrow \mathcal{P}(pk, x, w) \right] = 1.
  \]

- **Knowledge Soundness**: For each efficient $\tilde{\mathcal{P}}$ there exists an extractor $\mathcal{E}$ such that
  \[
  \Pr \left[ V(vk, x, \pi) = 1 \quad \land \quad (F, x, w) \notin \mathcal{R} \right] \approx 0
  \]
  
  \[
  \pi \leftarrow \tilde{\mathcal{P}}(pk, x), \quad w \leftarrow \mathcal{E}_{\tilde{\mathcal{P}}}(pk, x)
  \]

- **Succinctness**: $|\pi| = O(\log |F|)$
What about privacy?

- \( F(x, w) := \text{SHA2}(w) \overset{?}{=} x \) : Does proof reveal info about preimage?

- \( F((m, pk), \sigma) := \text{VerifySignature}(pk, m, \sigma) \overset{?}{=} 1 \) : Does proof reveal info about which signature was used?

- \( F(x = \text{score}, w = \text{credit_hist}) := \text{CreditModel}(w) \overset{?}{=} x \) : Does proof reveal info about credit history?

Verifier is the adversary now!
Zero Knowledge SNARKs (zkSNARKs)

- **Completeness**: For all \((F, x, w) \in \mathcal{R}\), ...
- **Knowledge Soundness**: For each efficient \(\tilde{P}\) there exists an extractor \(E\) such that ...
- **Zero Knowledge**: Proof reveals no information to \(V\) other than validity of \(w\)
- **Succinctness**: \(|\pi| = O(\log |F|)\)
Zero Knowledge SNARKs (zkSNARKs)

- **Completeness**: For all \((F, x, w) \in \mathcal{R}\), …

- **Knowledge Soundness**: For each efficient \(\tilde{P}\) there exists an extractor \(E\) such that …

- **Zero Knowledge**: For all \((F, x, w) \in R\), and all efficient \(\tilde{V}\) there exists an **simulator** \(\text{Sim}\) such that
  \[
  \Pr \left[ V(vk, x, \pi) : \pi \leftarrow \text{Sim}(pk, x) \right] = \Pr \left[ V(vk, x, \pi) : \pi \leftarrow P(pk, x, w) \right]
  \]

- **Succinctness**: \(|\pi| = O(\log |F|)\)
Doesn’t this break soundness?

\[
\Pr \left[ V(vk, x, \pi) : (pk, vk) \leftarrow \text{Setup}(F) \right. \left. \pi \leftarrow \text{Sim}(pk, x) \right] = \Pr \left[ V(vk, x, \pi) : (pk, vk) \leftarrow \text{Setup}(F) \right. \left. \pi \leftarrow P(pk, x, w) \right]
\]

Sim has same success probability as honest prover!

This is actually okay: we provide Sim with additional powers!

- Interactive case: Sim can rewind verifier
- Non-interactive case: Sim gets “trapdoor”/secret information
zk Marker Demo
What about succinct verification?

**Succinctness:** $|\pi| = O(\log |F|)$

- $F(x, w) = \text{SHA2}^{10^6}(w) \neq x$:
  Do I need to compute $10^6$ hashes to verify proof?

- $F(x = \text{score}, w = \text{credit_hist}) = \text{CreditModel}(w) \neq x$:
  Do I need to evaluate complex model to verify proof?
Strongly Succinct zkSNARKs

- **Completeness**: For all \((F, x, w) \in \mathcal{R}\), ...

- **Knowledge Soundness**: For each efficient \(\tilde{P}\) there exists an extractor \(E\) such that ...

- **Zero Knowledge**: For all \((F, x, w) \in \mathcal{R}\), and all efficient \(\tilde{V}\) there exists a simulator \(\text{Sim}\) such that

\[
\Pr \left[ V(\text{vk}, x, \pi) : \pi \leftarrow \text{Sim}(\text{pk}, x) \right] = \Pr \left[ V(\text{vk}, x, \pi) : \pi \leftarrow P(\text{pk}, x, w) \right]
\]

- **Succinctness**: 
  \[|\pi| = O(\log |F|)\]
  
  and \(\text{Time}(V) = O(\log |F|, |x|)\)
Constructing zkSNARKs
Starting point: Trivial NP Protocol

**Problem 1:** Non-succinct proof!
**Problem 2:** Non-succinct verification!
**Problem 3:** Not hiding at all!
Strawman 1: Hash the witness

Problem 1 solved: Succinct proof!
Problem 2: How to verify?
Problem 3: Still might not be hiding!
Strawman 2: Commit to the witness

Problem 1 solved: Succinct proof!
Problem 2: How to verify?
Problem 3: Still might not be hiding!
Commitment Schemes

\[ \text{Commit}(w; r) \rightarrow \text{cm} \]
satisfying the following properties

- **Binding**: For all efficient adv. \( \mathcal{A} \),
  \[ \Pr \left[ \text{Commit}(w; r) = \text{Commit}(w'; r') : (w, r, w', r') \leftarrow \mathcal{A} \right] \approx 0 \]
  (no adv can open commitment to two diff values)

- **Hiding**: For all \( w, w' \), and all adv. \( \mathcal{A} \),
  \[ \mathcal{A}(\text{Commit}(w; r)) = \mathcal{A}(\text{Commit}(w'; r')) \]
  (no adv can learn committed value, i.e. comms are indistinguishable)
A standard construction

Let $H$ be a cryptographic hash function. Then

\[
\text{Commit}(w; r) := H(w, r)
\]

is a commitment scheme
Strawman 2: Commit to the witness

Problem 1 solved: Succinct proof!
Problem 2: How to verify?
Problem 3 solved: COMM hides w!
Performing checks on committed data?
What does V do in the Trivial NP proof?

Evaluate $F(x, w)$!

To apply this to our commitment-based protocol, do we need a “fully-homomorphic” commitment?
Homomorphic Commitments?

Pair of algorithms with the following syntax:

- **Commit**$(w; r) \rightarrow cm$
  - Commits to the message

- **Eval**$(F_x, cm) \rightarrow F(x, w)$
  - Evaluates a function over the committed message, and outputs the result in the clear.
Strawman 3: Homomorphic Commitments

1. cm := Commit(w; r)

Completeness: Follows from that of commitment
Knowledge Soundness: Follows from Trivial NP Proof
Succinct pf size: Follows if eval. proof is succinct
ZK: ???

Problem 1: This would violate ZK: no hiding!
Problem 2: All constructions are inefficient!
Idea: Ask Prover to help

**Triple** of algorithms with the following syntax:

- **Commit**($m; r$) $\rightarrow$ $\text{cm}$
  - Commits to the message
- **ProveEval**($F, m; r$) $\rightarrow$ ($F(m)$, $\pi$)
  - Returns proof of correct evaluation of $F(m)$
- **CheckEval**($F, \text{cm}, v, \pi$) $\rightarrow$ $b \in \{0,1\}$
  - Checks that $\pi$ is a valid proof that $F(m) = v$, where $m$ is the msg inside $\text{cm}$

**Does this work?**
Strawman 4: Functional Commitments

\begin{align*}
\text{Prover}(pk, x, w) \\
1. \text{Commit}(w; r) \\
2. \text{ProveEval}(F_x, w; r)
\end{align*}

\begin{align*}
\pi_{F_x}
\end{align*}

\begin{align*}
\text{Verifier}(vk, x) \\
\text{CheckEval}(F_x, cm, 1, \pi)
\end{align*}

Completeness: Follows from that of (ProveEval, CheckEval)
Knowledge Soundness: Ditto
ZK: Follows from hiding
Succinct pf size: Follows if eval. proof is succinct

Are we done?

No! We just pushed the problem one layer down!
Problem: This is a zkSNARK for $F$!

**Triple** of algorithms with the following syntax:

- **Commit**$(m; r) \rightarrow cm$
  - Commits to the message
- **ProveEval**$(F, m; r) \rightarrow (F(m), \pi)$
  - Returns proof of correct evaluation of $F(m)$
- **CheckEval**$(F, cm, v, \pi) \rightarrow b \in \{0,1\}$
  - Checks that $\pi$ is a valid proof that $F(m) = v$, where $m$ is the msg inside $cm$
Let’s Reassess Our Status

- Trivial NP Proof System
- Very Complex Commitment

Compiler

zkSNARK
How about we rebalance?

More Complex Proof System

Simpler Commitment

Compiler

zkSNARK

How do we find appropriate balance?
What commitment schemes exist?

**Polynomial commitments:**
- $F_z(m)$: Interpret $m$ as univariate poly $f(X)$ in $\mathbb{F}[X]$ and evaluate at $z$

**Multilinear commitments:**
- $F_{\vec{z}}(m)$: Interpret $m$ as multilinear poly $f(X)$ in $\mathbb{F}[\vec{X}]$ and evaluate at $\vec{z}$

**Vector commitments:**
- $F_i(m)$: Interpret $m$ as vector $\vec{v}$ in $\mathbb{F}^n$ and return $v_i$

**Inner-product commitments:**
- $F_{\vec{q}}(m)$: Interpret $m$ as vector $\vec{v}$ in $\mathbb{F}^n$ and return $\langle \vec{v}, \vec{q} \rangle$

Which to pick?
A: Polynomials!

- Reed Soloman code: Polynomial
- Zero-Knowledge Proof Systems: Polynomials
- Secret Sharing: Polynomial Evaluations
- Identity Testing: Polynomials equal?
- FFTs: Polynomials
- FRI: FFTs -> Polynomials
- SNARK: Polynomials
- STARK: SNARK
- Security Parameter: Polynomial
- Lagrange: Polynomial

11:28 AM · Oct 25, 2021 · Twitter Web App
Let’s pick polynomials

- ???
- Polynomial Commitment
- Compiler
- zkSNARK