Proof Systems and SNARKs

Dan Boneh
Where we are in the course

• Basics: Consensus protocols and Bitcoin

• Composable decentralized applications (e.g., on Ethereum)

  ⇒ Decentralized Finance (DeFi)

  ⇒ Scaling the blockchain:
     payment channels,
     Rollup (Proof-based or Optimistic),
     faster consensus

Last core topic: privacy -- private transactions on a public blockchain
Managing assets on a blockchain: key principles

• **Universal verifiability** of blockchain rules
  ⇒ all data written to the blockchain is public; everyone can verify
  ⇒ added benefit: interoperability between chains

• Assets are **controlled by signature keys**
  ⇒ assets **cannot** be transferred without a valid signature
    (of course, users can choose to custody their keys)
Naïve reasoning:

universal verifiability $\Rightarrow$ blockchain data is public

$\Rightarrow$ all transactions data is public

otherwise, how we can verify Tx?

not quite ...
### Public blockchain & universal verifiability

- **Tx data**: encrypted (or committed)

- **Proof $\pi$**: zero-knowledge proof that (reveals nothing about Tx data)
  1. plaintext Tx data is consistent with plaintext current state
  2. plaintext new state is correct
Public blockchain & universal verifiability

- **Tx data**: encrypted (or committed)
- **Proof** $\pi$: *zero-knowledge proof* that (reveals nothing about Tx data)
  1. plaintext Tx data is consistent with plaintext current state
  2. plaintext new state is correct
Zero Knowledge Proof Systems
(1) arithmetic circuits

• Fix a finite field $\mathbb{F} = \{0, ..., p - 1\}$ for some prime $p > 2$.

• **Arithmetic circuit:** $C : \mathbb{F}^n \rightarrow \mathbb{F}$
  
  • directed acyclic graph (DAG) where
    
    • internal nodes are labeled $+, -, \text{or} \times$
    
    • inputs are labeled $1, x_1, ..., x_n$
  
  • defines an $n$-variate polynomial with an evaluation recipe

• $|C| = \#$ multiplication gates in $C$
Boolean circuits as arithmetic circuits

Boolean circuits: circuits with AND, OR, NOT gates

Encoding a boolean circuit as an arithmetic circuit over $\mathbb{F}_p$:

- **AND**($x, y$) encoded as $x \cdot y$
- **OR**($x, y$) encoded as $x + y - x \cdot y$
- **NOT**($x$) encoded as $1 - x$

<table>
<thead>
<tr>
<th>$x$</th>
<th>$y$</th>
<th>OR($x, y$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Interesting arithmetic circuits

• $C_{\text{hash}}(h, m)$: outputs 0 if $\text{SHA256}(m) = h$, and $\neq 0$ otherwise

\[
C_{\text{hash}}(h, m) = (h - \text{SHA256}(m)), \quad |C_{\text{hash}}| \approx 20\text{K gates}
\]

• $C_{\text{sig}}((pk, m), \sigma)$: output 0 if $\sigma$ is a valid ECDSA signature of $m$ under $pk$
Let $x \in \mathbb{F}_p^n$. Two standard goals for prover $P$:

1. **Soundness**: convince Verifier that $\exists w$ s.t. $C(x, w) = 0$
   (e.g., $\exists w$ such that $[H(w) = x$ and $0 < w < 2^{60}]$)

2. **Knowledge**: convince Verifier that $P$ “knows” $w$ s.t. $C(x, w) = 0$
   (e.g., $P$ knows a $w$ such that $H(w) = x$)
The trivial proof system

Why can’t prover simply send $w$ to verifier?

• Verifier checks if $C(x, w) = 0$ and accepts if so.

Problems with this:

(1) $w$ might be secret: prover cannot reveal $w$ to verifier

(2) $w$ might be long: we want a “short” proof

(3) computing $C(x, w)$ may be hard: want to minimize Verifier’s work
Non-interactive Proof Systems (for NP)

Public arithmetic circuit: $C(x, w) \rightarrow \mathbb{F}_p$

- public input in $\mathbb{F}_p^n$
- secret witness in $\mathbb{F}_p^m$

setup: $S(C) \rightarrow$ public parameters $(S_p, S_v)$

Prover $P(S_p, x, w)$

Verifier $V(S_v, x, \pi)$

proof $\pi$

output accept or reject
A non-interactive proof system is a triple \((S, P, V)\):

- \(S(C) \rightarrow\) public parameters \((S_p, S_v)\) for prover and verifier
- \(P(S_p, x, w) \rightarrow\) proof \(\pi\)
- \(V(S_v, x, \pi) \rightarrow\) accept or reject
proof systems: properties (informal)

Prover $P(pp, x, w)$

Verifier $V(pp, x, \pi)$

proof $\pi$

accept or reject

Complete: $\forall x, w: C(x, w) = 0 \Rightarrow V(S_v, x, P(S_p, x, w)) = \text{accept}$

Proof of knowledge: $V$ accepts $\Rightarrow P$ “knows” $w$ s.t. $C(x, w) = 0$

in some cases, soundness is sufficient: $\exists w$ s.t. $C(x, w) = 0$

Zero knowledge (optional): $(x, \pi)$ “reveals nothing” about $w$
Goal: \( V \) accepts \( \Rightarrow P \) “knows” \( w \) s.t. \( C(x, w) = 0 \)

What does it mean to “know” \( w \)??

**informal def:** \( P \) knows \( w \), if \( w \) can be “extracted” from \( P \)
(a) Proof/argument of knowledge

Formally: \((S, P, V)\) is a **proof of knowledge** for a circuit \(C\) if for every adversary \(A = (A_0, A_1)\) such that

\[
S(C) \rightarrow (S_p, S_v), \quad (x, st) \leftarrow A_0(S_p), \quad \pi \leftarrow A_1(S_p, x, st):
\]

\[
\Pr[V(S_v, x, \pi) = \text{accept}] > 1/10^6 \quad \text{(non-negligible)}
\]

there is an efficient extractor \(E\) (that uses \(A_1\) as a black box) s.t.

\[
S(C) \rightarrow (S_p, S_v), \quad (x, st) \leftarrow A_0(S_p), \quad w \leftarrow E(S_p, x, st):
\]

\[
\Pr[C(x, w) = 0] > 1/10^6 \quad \text{(non-negligible)}
\]

If only for poly. time \(A \Rightarrow (S, P, V)\) is only an **argument of knowledge**.
Formally, \((S, P, V)\) is a proof of knowledge for a circuit \(C\) if

\[
\text{for every adversary } A = (A_0, A_1) \text{ such that } S(C) \xrightarrow{\cdot} (S_p, S_v), (x, st) \xleftarrow{} A_0(S_p), \pi \xleftarrow{} A_1(S_p, x, st):
\]

\[
\Pr[V(S_v, x, \pi) = \text{accept}] > \frac{1}{10}
\]

(non-negligible)

there is an efficient extractor \(E\) (that uses \(A_1\) as a black box)

\[
S(C) \xrightarrow{\cdot} (S_p, S_v), (x, st) \xleftarrow{} A_0(S_p), w \xleftarrow{} E(S_p, x, st):
\]

\[
\Pr[C(x, w) = 0] > \frac{1}{10}
\]

(non-negligible)

If only for poly. time \(A \Rightarrow (S, P, V)\) is only an argument of knowledge.

**Proof**: secure against unbounded cheating provers

**Argument**: secure against polynomial-time cheating provers
(b) Zero knowledge

(S, P, V) is zero knowledge if proof $\pi$ “reveals nothing” about $w$

Formally: (S, P, V) is zero knowledge for a circuit $C$
if there is an efficient simulator $Sim$,
such that for all $x \in \mathbb{F}_p^n$ s.t. $\exists w$: $C(x, w) = 0$
the distribution:

$$(S_p, S_v, x, \pi) \quad \text{where} \quad (S_p, S_v) \leftarrow S(C), \quad \pi \leftarrow P(x, w)$$

is indistinguishable from the distribution:

$$(S_p, S_v, x, \pi) \quad \text{where} \quad (S_p, S_v, \pi) \leftarrow Sim(x)$$

key point: $Sim(x)$ simulates proof $\pi$ without knowledge of $w$
(3) Succinct arguments: SNARKs

Goal: $P$ wants to show that it knows $w$ s.t. $C(x, w) = 0$

**Succinct:**

- Proof $\pi$ should be **short** [i.e., $|\pi| = O(|x|, \log(|C|), \lambda)$]
- Verifying $\pi$ should be **fast** [i.e., $\text{time}(V) = O(|x|, \log(|C|), \lambda)$]

Note: if SNARK is zero-knowledge, then called a zkSNARK
Goal: P wants to show that it knows \( w \) s.t. \( C(x, w) = 1 \)

Succinct:

- Proof \( \pi \) should be **short** [i.e., \( |\pi| = O(|x|, \log(|C|), \lambda) \)]
- Verifying \( \pi \) should be **fast** [i.e., time(\( V \)) = \( O(|x|, \log(|C|), \lambda) \)]

Note: if SNARK is zero-knowledge, then called a **zkSNARK**
An example

Prover says: I know \((x_1, ..., x_n) \in X\) such that \(H(x_1, ..., x_n) = y\)

**SNARK:** size\((\pi)\) and VerifyTime\((\pi)\) should be \(O(\log n)\)!!
An example

How is this possible ???

**SNARK**: size(π) and VerifyTime(π) should be $O(\log n)$ !!
Types of pre-processing Setup

Recall setup for circuit $C$: $S(C) \rightarrow$ public parameters $(S_p, S_v)$

Types of setup:

- **trusted setup per circuit:** $S(C)$ uses data that must be kept secret
  
  compromised trusted setup $\Rightarrow$ can prove false statements

- **updatable universal trusted setup:** $(S_p, S_v)$ can be updated by anyone

- **transparent:** $S()$ does not use secret data (no trusted setup)
Significant progress in recent years

- Kilian’92, Micali’94: succinct transparent arguments from PCP
  - impractical prover time

- GGPR’13, Groth’16, …: linear prover time, constant size proof $O_\lambda(1)$
  - trusted setup per circuit (setup alg. uses secret randomness)
  - compromised setup $\Rightarrow$ proofs of false statements

- Sonic’19, Marlin’19, Plonk’19, …: universal trusted setup

- DARK’19, Halo’19, STARK, …: no trusted setup (transparent)
## Types of SNARKs (partial list)

|                 | size of $|\pi|$ | size of $|S_p|$ | verifier time | trusted setup?         |
|-----------------|-------------|-------------|---------------|-----------------------|
| Groth’16        | $O(1)$      | $O(|C|)$    | $O(1)$        | yes/per circuit       |
| PLONK/MARLIN    | $O(1)$      | $O(|C|)$    | $O(1)$        | yes/updatable         |
| Bulletproofs    | $O(\log|C|)$ | $O(1)$      | $O(|C|)$      | no                    |
| STARK           | $O(\log|C|)$ | $O(1)$      | $O(\log|C|)$  | no                    |
| DARK            | $O(\log|C|)$ | $O(1)$      | $O(\log|C|)$  | no                    |
A typical SNARK software system

**DSL program**
- Circom,
- ZoKrates,
  ...

**compiler**

**SNARK friendly format**
- R1CS,
- AIR,
- TurboPlonk

**SNARK backend**

**Proof $\pi$**

**CPU heavy**

**setup**

$\pi(S_p, S_v)$

**accept/reject**

**verifier**

$x, \text{witness}$
def main(field x[2], private field w) -> (field):
    h = sha256packed( w )
    h[0] == x[0]       // check top 128 bits
    return 1

Goal: prove knowledge of a hash (SHA256) preimage of $x \in \{0,1\}^{256}$

- For a public $x$, prover knows $w \in \mathbb{F}_p$
- $\mathbb{F}_p$ is a 254-bit prime field

Compiled into an arithmetic circuits (R1CS) over $\mathbb{F}_p$
zkSNARK applications
Blockchain Applications

Scalability:
• SNARK Rollup  (zkSNARK for privacy from public)

Privacy:
• Private Tx on a public blockchain
  • Confidential transactions
  • Zcash

Compliance:
• Proving solvency in zero-knowledge
• Zero-knowledge taxes
A simple PCP-based SNARK

[Kilian’92, Micali’94]
A simple construction: PCP-based SNARK

The PCP theorem: Let $C(x, w)$ be a circuit where $x \in \mathbb{F}_p^n$. There is a proof system that for every $x$ proves $\exists w: C(x, w) = 0$ as follows:

- **Prover $P(S_p, x, w)$**: Proof $\pi$

- **Verifier $V(S_v, x)$**: Read only $O(\lambda)$ bits of $\pi$, output accept or reject

V always accepts valid proof. If no $w$, then V rejects with high prob.

Size of proof is $\text{poly}(|C|)$. (not succinct)
Converting a PCP proof to a SNARK

Prover $P(S_p, x, w)$
Verifier $V(S_v, x)$

Merkle

$h$

$h$

open $O(\lambda)$ positions of $\pi$

$O(\lambda)$ opening and Merkle proofs

Verifier sees $O(\lambda \log |C|)$ data $\Rightarrow$ succinct proof.

Problem: interactive
Making the proof non-interactive

The **Fiat-Shamir heuristic**:

- public-coin interactive protocol $\Rightarrow$ non-interactive protocol
  - public coin: all verifier randomness is public (no secrets)

Prover $P(S_p, x, w)$

Verifier $V(S_v, x)$

msg1

r

choose random bits $r$

msg2

accept or reject
Making the proof non-interactive

Fiat-Shamir heuristic: \( H: M \rightarrow R \) a cryptographic hash function

- idea: prover generates random bits on its own (!)

Prover \( P(S_p, x, w) \)
- generate msg1
- \( r \leftarrow H(x, \text{msg1}) \)
- generate msg2

Verifier \( V(S_v, x) \)
- \( \pi = (\text{msg1, msg2}) \)
- \( |\pi| = O(\lambda \log |C|) \)
- \( r \leftarrow H(x, \text{msg1}) \)
- accept or reject

Thm: this is a secure SNARK assuming \( H \) is a random oracle
Are we done?

Simple transparent SNARK from the PCP theorem

• Use Fiat-Shamir heuristic to make non-interactive
• We will apply Fiat-Shamir in many other settings

The bad news: an impractical SNARK --- Prover time too high

Better SNARKs: next lecture! Goal: Time(Prover) = O(|C|)
Next lecture: zkSNARK applications