AnonStake:
An Anonymous Proof-of-Stake Cryptocurrency via Zero-Knowledge Proofs and Algorand

Shashvat Srivastava
MIT Primes
Under the Direction of Ms. Kyle Hogan
Massachusetts Institute of Technology

October 13, 2018
Cryptocurrencies are a form of digital currency

- Use consensus methods instead of central authorities
- Use encryption to guarantee that currency can only be spent by proper owner
- First cryptocurrency: Bitcoin
Problems with Bitcoin

Bitcoin’s uses Proof-of-Work for “decentralized” consensus

Figure 1: Four entities (mining pools) hold 51% of the hash power in the network. (Source: blockchain.com, 2018)
Problems with Bitcoin, continued

Bitcoin’s uses Proof-of-Work for “decentralized” consensus

- Not decentralized
- Uses as much electricity as Switzerland
- Very slow: each block takes 10 minutes
- Possible solution: **Proof-of-Stake**
Proof-of-Stake

- Users reach consensus by voting (usually through committees)
- Voter’s impact is proportional to amount of money they have
- Assumption is that most money is held by honest users
- Heavily invested users want currency to perform well
Algorand is a fast Proof-of-Stake cryptocurrency, featuring
- Fast block times (∼1 minute)
- Low confirmation times
- Generally more robust to user corruption than other Proof-of-Stake cryptocurrencies
Figure 2: We will be focusing on modifying step one, sortition.
Anonymous Cryptocurrencies

Algorand is fully public; we want to make it anonymous. Some cryptocurrencies have a strong focus on anonymity (ZCash, Monero). Able to hide:

- The senders and receivers of the transaction
- The amount sent in the transaction
Goals

We want to create an anonymous cryptocurrency with Proof-of-Stake consensus.

- Algorand consensus needs users to know each other’s account balances
- Anonymity implies that user’s don’t know each other’s account balances
Goals

We want to create an anonymous cryptocurrency with Proof-of-Stake consensus.

- Algorand consensus needs users to know each other’s account balances
- Anonymity implies that user’s don’t know each other’s account balances
- **Solution:** Use zero-knowledge proofs
Zero-Knowledge Proofs

- Introduced as "Proofs that yield nothing but their validity"
- zkSNARKs can be used to prove validity of any NP statement

Figure 3: zkSNARKs can be used to prove that a (publicly-known) C-program will return True.
coins and coin commitments

\[ \text{coin } c = (v, pk, sn, \ldots) \]

\[ \text{coin commitment } cm \]
Use the same transaction structure as ZCash

An anonymous transaction consists of a serial number $sn$, a new coin commitment $cm^{new}$, and a zkSNARK proof
zkSNARK proof proves that:

- You own a valid coin:
  - You know a (secret) coin $c^{old}$ with (secret) commitment $cm^{old}$
  - $cm^{old}$ in \{all coin commitments\}
- The coin has not been spent yet:
  - You reveal the coin’s serial number $sn$
- You aren’t creating money:
  - You know (secret) coin $c^{new}$ that has commitment $cm^{new}$
  - The values of $c^{new}$ and $c^{old}$ are the same

Ultimately, proves that the transaction was valid.
Anonymous Sortition

General idea:

- Prove ownership of a secret coin
  - Same as before
- Prove coin has not been spent yet:
  - Prove the (secret) $sn$ of the coin is not in \{spent serial numbers\}
- Prove you aren’t trying to vote twice
  - Reveal the temporary serial number $tsn$ of the coin
- Prove that the user was selected from (secret) coin value $v$
Need For Speed

- Want to retain Algorand’s speed
- Even 7 second proof generation is too slow
- Our proof is much larger than a ZCash transaction
Need For Speed, continued

- Pursued many different methods
- Replace SHA256 hash with MiMC hash
Future Work

- Faster computations
- Compositional analysis of security
- Code implementation
Acknowledgements

- My mentor, Ms. Kyle Hogan
- MIT Primes
- Professor Gerovitch
- Professor Devadas