zk-channels

Private layer 2 payments on Ethereum

Inspired by the ETHSignapore hackathon project zkDAI which was inspired by the Zerocash paper.

Abstract

Private payments on Ethereum have struggled to gain traction due to their relatively high transaction cost. Combining the scalability of payment channels with the privacy provided by zk-snarks can enable a cost-effective and private payment network on Ethereum.

Description

zk-channels function by using “notes” tracked in an Ethereum smart contract. Each note has a private owner and value and is represented by a hash. There are two types of notes: Note

, and ChannelNote

. A Note

is represented by the hash of the owner, the value, and a salt. A ChannelNote

is represented by the hash of the owner, the value, the channel recipient, and a salt.

Note

s function much like UTXOs. A new Note

can be created by depositing the collateral token into the contract. Note

s can then be privately transferred by proving ownership. When a Note

is transferred, two new Note

s are created, one owned by the receiver, one owned by the sender with the remaining value. At any time, a Note

can also be redeemed for the underlying collateral token.

Note

s can also be used to open a payment channel by converting it into a ChannelNote

. ChannelNote

s can be spent using a signed message from the owner. ChannelNote

owners can also force the receiver to withdraw within a given period in order to unlock the ChannelNote

's remaining value.

A payment channel is created by the spender by first creating a ChannelNote

and then sending signed messages of increasing value to the receiver. Only the receiver can close the channel allowing them to use the last and highest value signed message. If the sender wishes to withdraw the remaining value in a ChannelNote

, they can force a withdrawal by kicking off a withdrawal period during which the receiver must withdraw or forfeit their payment.

Note - (owner, value, salt)

Represented by the hash of (owner, value, salt)

Actions

- createNote()
- Collects deposit and creates a Note

of equal value.

- transferNote()
- Validates proof, marks the Note

as spent and creates two new Note

s

- depositNote()
- Validates proof (snark not described), marks the Note

as spent and creates a ChannelNote

- redeemNote()
- Marks Note

as spent and transfers the deposit to owner

transferNote() zk-snark description

- Public inputs:

originalNote

, noteA

, noteB

- Private inputs:

sender

, value

, salt

, receiver

, valueA

, saltA

, valueB

, saltB

, senderPrivateKey

- Verifies:
- originalNote

is the hash of (sender, value, salt)

1. noteA

is the hash of (sender, valueA, saltA)

1. noteB

is the hash of (receiver, valueB, saltB)

1. valueA
2. valueB

= value

1. senderPrivateKey

is the private key of sender

- originalNote

is the hash of (sender, value, salt)

- noteA

is the hash of (sender, valueA, saltA)

- noteB

is the hash of (receiver, valueB, saltB)

- valueA
- valueB

= value

- senderPrivateKey

is the private key of sender

ChannelNote - (owner, value, receiver, salt)

Represented by the hash of (owner, value, receiver, salt)

Actions

- closeChannel()
- Validates proof, marks the ChannelNote

as spent, and creates two new Note

s

- forceWithdrawal()
- Validates proof (snark not described), kicks off a withdrawal period during which the receiver must withdraw. If no withdrawal is made, sender

can spend the ChannelNote

for a single Note

with equal value.

closeChannel() zk-snark description

- Public inputs:

originalNote

, noteA

, noteB

- Private inputs:

sender

, value

, salt

, receiver

, valueA

, saltA

, valueB

, saltB

, signature

, receiverPrivateKey

- Verifies:
- sender

is the signer recovered from signature

with the message (sender, value, receiver, salt, valueB)

1. originalNote

is the hash of (sender, value, receiver, salt)

1. noteA

is the hash of (sender, valueA, saltA)

1. noteB

is the hash of (receiver, valueB, saltB)

1. valueA
2. valueB

= value

1. receiverPrivateKey

is the private key of receiver

- sender

is the signer recovered from signature

with the message (sender, value, receiver, salt, valueB)

- originalNote

is the hash of (sender, value, receiver, salt)

- noteA

is the hash of (sender, valueA, saltA)

- noteB

is the hash of (receiver, valueB, saltB)

- valueA
- valueB

= value

- receiverPrivateKey

is the private key of receiver