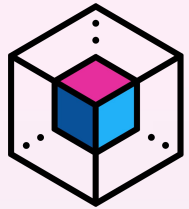


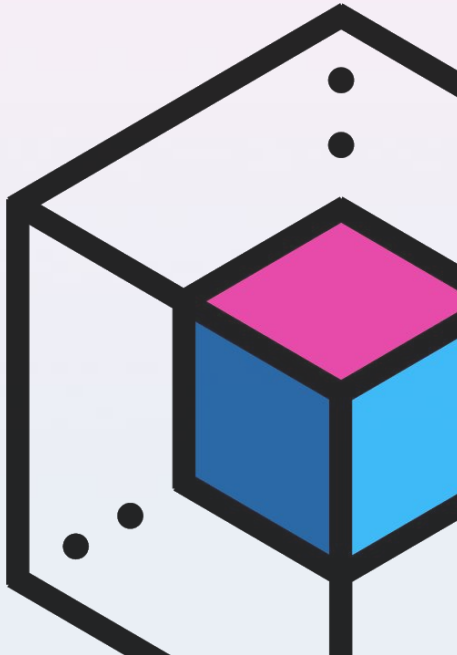
enigma

Private Smart Contracts on Ethereum

Enigma's Vision

Our mission is to create products and systems that accelerate the **adoption** and **usability** of decentralized technologies.

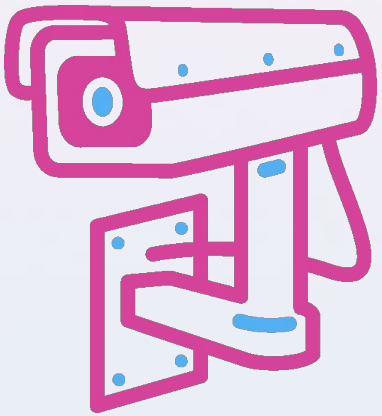
Salad🥗 is the user-friendly Ethereum mixer



The Problem of Privacy

Data on blockchains is public by default.

This greatly limits potential applications.

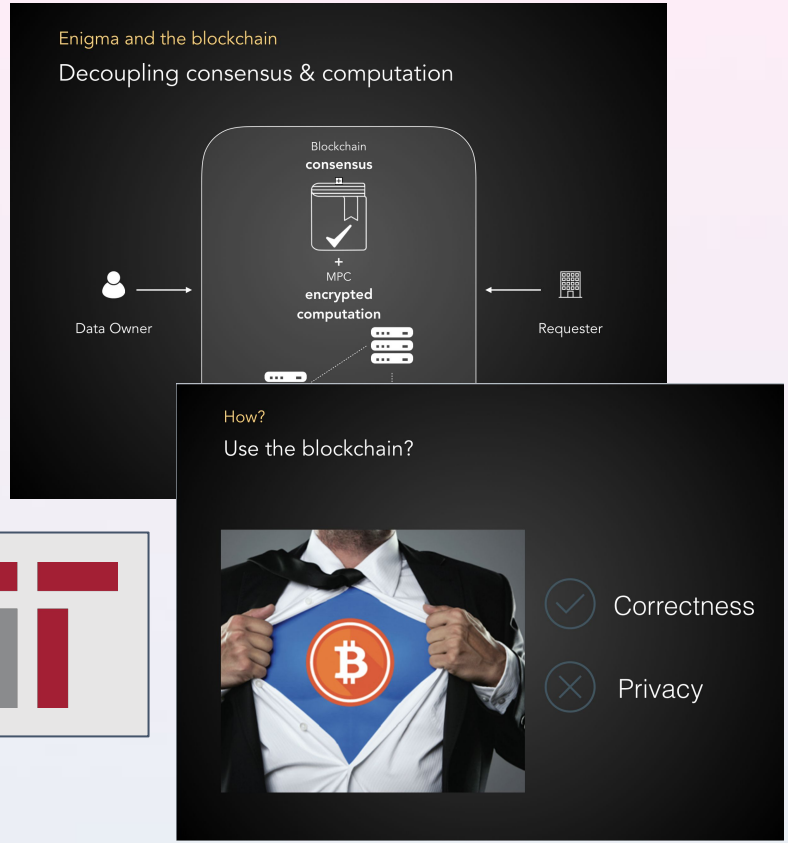


Our Pursuit Of Privacy

"Decentralizing Privacy" - 2015

"Enigma: Decentralized Computation Platform with Guaranteed Privacy" - 2015

1,000+ combined citations

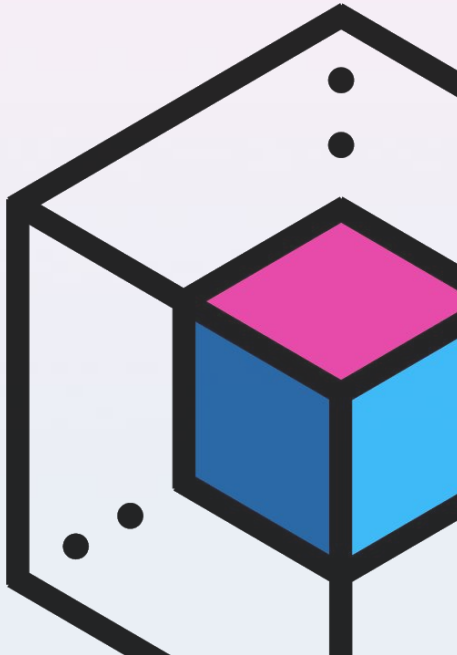


These are from MIT Bitcoin Expo 2016!

Enigma: A Protocol for Secure Computation

Enigma enables decentralized applications to compute over encrypted data.

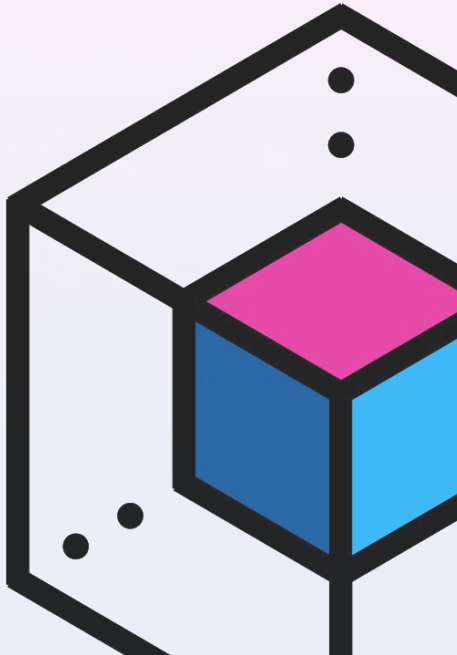
Secret contracts use private computation methods to allow data inputs to remain hidden even from nodes.



Enigma: Discovery

The first network that publicly enables secret contracts.

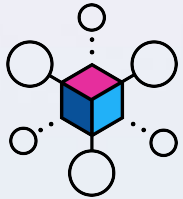
- Permissionless network
- Secret state
- Proof of Stake
- Compatible with Ethereum



ENIGMA STAKEHOLDERS

Developers deploy dApps

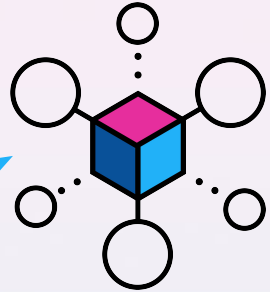
Users create tasks to use dApps



Workers execute tasks

DEVELOPERS build secret contracts

Developers write secret contracts in Rust, compiling to WASM - *compatible with Web3 stack*



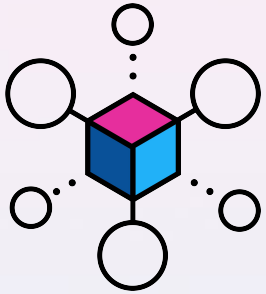
Secret contracts are deployed to Enigma Network

Ethereum blockchain

A hash of the Secret contract is sent to the Enigma Contract on Ethereum

USERS create tasks

1) Users interact with dApps by submitting encrypted data using Enigma.js library - this creates “Tasks”



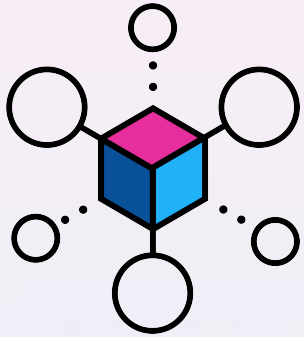
3) The Task is sent to the Enigma Network

2) A hash of the Task is sent to the Enigma Contract on Ethereum (known as the “TaskID”)

WORKERS execute tasks

1) Workers receive tasks based on a stake (token)-weighted lottery

task



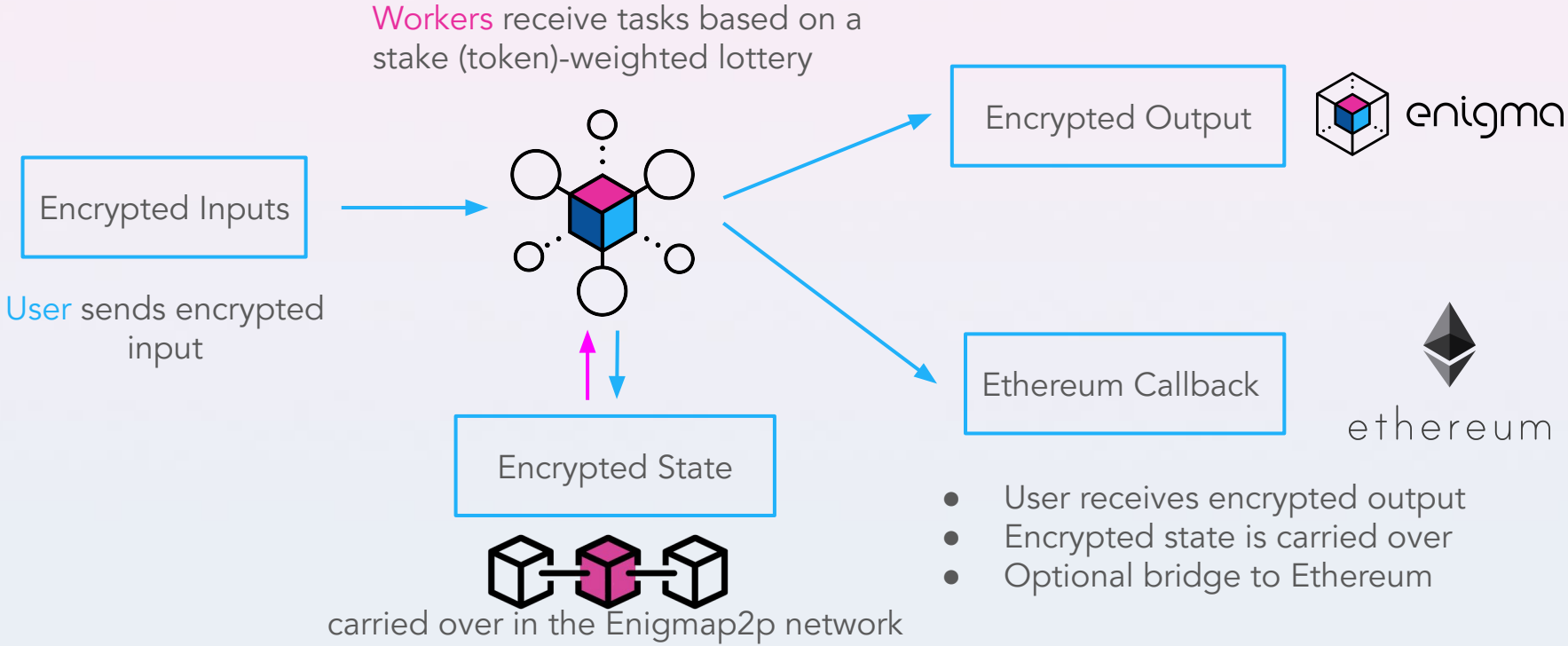
2) Only selected workers execute the task and disseminate the results across Enigmap2p network (libp2p)

3) Workers submit result to Ethereum callback, or after epoch ends

4) After callback, Enigma Contract releases payment to the worker

Ethereum blockchain

How Secret Contracts work?



Salad: How it Works

- A deposit contract on Ethereum which holds funds to be mixed.
- A secret contract on Enigma, which randomizes receipt addresses.
- A relayer/operator who is responsible for coordinating deposits and batching transactions.

SALAD

PRIVACY IS HEALTHY

Why We Built Salad

- Usability: Salad is non-interactive
  - Single user interaction with Salad (submit encrypted recipient address)
  - Two interactions needed (deposit, withdraw) in ZK implementations

- Cost: Salad has ~⅕ cost of ZK based mixers
  - a 10-participant mix on Enigma should cost ~250,000 gas
  - 1.5mn - 2mn gas for ZK-based implementations

What can you build this weekend?

- Integrate Salad to your favorite wallet / dApp
- Private voting within a DAO (DaoStack, Aragon)
- Private auctions and order books for DeFi
- Multiplayer games with secret state (rock paper scissors, battleship etc.)
- Games that use randomness like dice
- Quiz or word competition games like Jeopardy or Hangman
- Decentralized content management (access control with decentralized data marketplaces Ocean)
- Privacy preserving self-sovereign identity solutions (maybe with Sovrin)

For more ideas use the QR code

How can you build secret contracts for existing Ethereum dApps?

1. Run ganache using discovery-cli

2. Deploy the Ethereum dApp contract to the same ganache backend using Truffle

3. Integrate Enigma contract on Ethereum with Ethereum dApp contract to allow Enigma network to call Ethereum dApp function. *For example, in Salad, there's an Ethereum contract used for deposits. This contract is called by Enigma network*

4. Optional - the rest of the work is done on the web app level. *For example, import the salad WS client (a npm package) in the Ethereum dApp web app to enable the desired user workflow*

JOIN US!

- Want to build secret contracts right away?
  enigma.co/startnow

- Technical resource index

- Technical support on Discord

Check out Salad: github.com/enigmampc/coinjoin-poc
We talk here: forum.enigma.co
Telegram: @cankisagun