State Model

Aztec has a hybrid public/private state model. Aztec contract developers can specify which data is public and which data is private, as well as the functions that can operate on that data.

Public State

Aztec has public state that will be familiar to developers coming that have worked on other blockchains. Public state is transparent and is managed by the associated smart contract logic.

Internal to the Aztec network, public state is stored and updated by the sequencer. The sequencer executes state transitions, generates proofs of correct execution (or delegates proof generation to the prover network), and publishes the associated data to Ethereum.

Private State

Private state must be treated differently from public state and this must be expressed in the semantics of Aztec.nr.

Private state is encrypted and therefore is "owned" by a user or a set of users (via shared secrets) that are able to decrypt the state.

Private state is represented in an append-only database since updating a record would leak information about the transaction graph.

The act of "deleting" a private state variable can be represented by adding an associated nullifier to a nullifier set. The nullifier is generated such that, without knowing the decryption key of the owner, an observer cannot link a state record with a nullifier.

Modification of state variables can be emulated by nullifying the state record and creating a new record to represent the variable. Private state has an intrinsic UTXO structure and this must be represented in the language semantics of manipulating private state.

Abstracting UTXO's from App's / Users

The goal of the Aztec.nr smart contract library is to abstract the UTXO model away from an app user / developer, contract developers are the only actor who should have to think about UTXO's.

This is achieved with two main features:

1. Users sign over transactions, not over specific UTXO's
2. Aztec.nr contracts support developer definedunconstrained
3. getter functions to help dApp's make sense of UTXO's. e.ggetBalance()
4. . These functions can be called outside of a transaction context to read private state.

To be documented soon

- The lifecycle of a note
- Custom notes
- Injection of data by the kernel
- Nonce & contract address
- Custom nullifiers
- Emission of custom note data to L1
- Decrypting and storing encrypted note data

Further reading

Read more about how to leverage the Aztec state model in Aztec contractshere . Edit this page

Previous Concepts Next Public VM