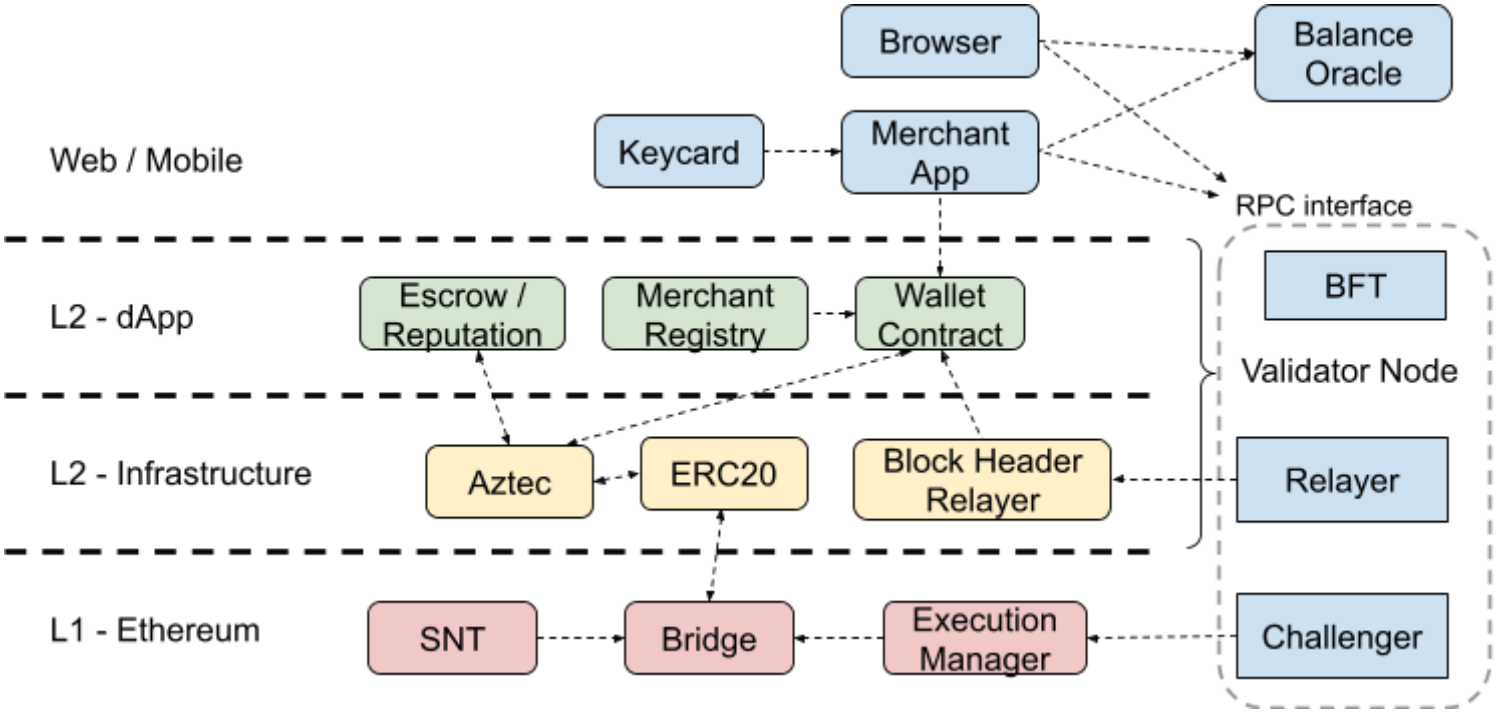
StatusPay with ORUs



Components:

Bridge:

A contract on the mainnet that bootstraps the ORU. it caters to validators, users and observers.

Validators have the following functions available:

- register and leave a stake
- Submit anchors and claim rewards
- Unregister and collect stake

Users have the following options:

- Register new ERC20 tokens (implicit registration, no unique traits, only blank ERC20 / ERC721 features)
- Deposit funds
- Exit funds

Observers

- use the attached execution manager to challenge invalid transitions on the ORU

The bridge starts with only SNT as the first registered token. The genesis file for the validators is built from the first 4 registrations.

Block Header Relayer:

A contract operating on layer-2 providing the ability to receive Ethereum block headers and verify the including PoW. Once the header is accepted it is fitted into the chain of headers in the storage of the rollup. Given all block headers are relayed the rollup can now independently verify the valid tip of Ethereum. The block hashes can be used by the wallet contract to prevent double-spend and double-tap fraud by merchants with minimal modifications.

Aztec:

Aztec is a framework for Ethereum contract and offers shielded functions:

- **Mint / Burn -** Allows the creation and destruction of fully private zero-knowledge assets
- **Public / Private Range Prove** - that a zero-knowledge asset has a value greater or smaller than another zero-knowledge asset / or public integer without revealing the asset's value.
- **Send** - This proof is used to confidentially transfer value stored inside AZTEC

The framework can be integrated with status through a note registry contract.

Wallet Contract:

Similar to current version:

- with a modification to use Block Header Relayer contract as source of block headers instead of native solidity call.
- Calling the Aztec note registry instead of the ERC20 token directly.

Questions:

transfer from the status wallet to the keycard:

The status wallet receives an extra button (Deposit to StatusPay). When this button is pressed, any ERC20 is transferred to the ORU bridge with 3 steps:

- Approval of amount of tokens, if not already present
- Call to deposit() function of bridge, which will then pull in the funds using transferFrom() on the ERC20 contract.
- The Deposit event injects a transaction into the rollup, which will then cause a mint of tokens in a matching ERC20.
- The status wallet displays the amount as reduced on layer-1, and the balance is queried and displayed through the RPC of the ORU node. The funds are not blinded yet.
- The wallet issues a third transaction to deposit the funds into ZK through the Aztec library(blinding). The fund balance is now private.
- The wallet transfers parts of the blinded funds to the keycard. Because all blinded funds have a random value that needs to be kept off-chain, to be able to read and spend them, the wallet publishes the random value with the Balance oracle. The published compromise the blinding, and can be used by browsers to display the balance and by merchants to initialize transactions.

payment execution step by step with keycard only:

1. User loads the card with blinded funds.
2. The unblinded value is saved with the `balance oracle` backend.
3. The user wants to purchase something with a merchant now.
4. Merchant gets the random value from the `balance oracle`. (possible authentication?)
5. Merchant generates a signature of `random value` with the keycard of the user.
6. Merchant now knows the balance of the card..
7. Merchant constructs the zk-proof, containing:
   * the old balance as input
   * the new balance as output
   * the difference between old and new is his share.
   * proofs that the transaction limit has been respected.
8. Merchant sends the signed proof to the validator.
9. Validator includes the transaction and updates the `balance oracle`.

10. The `balance oracle` can also verify the proof for validity before updating its state.

How long will the interaction take until the merchant can let the customer go?

We see 2 options to realize this:

Operator guarantees: immediate finality, but centralized in terms of communication. Decentralization of communication leads to BFT-similarity

BFT consensus: Has quadratic overhead. A network of < 64 nodes can produce blocks within 2-4 seconds.

How will you integrate SNT into the payment network?

SNT integration:
The status network token is integrated:
- as reward token for submitting new anchors
- as reward token for block header relays
- as native token of the rollup, needed by merchants to pay for transaction gas.
- as staking token for the BFT validators
- as staking token for consensus challenges

Which transaction capacity (tps) does your proposal provide?

https://docs.google.com/spreadsheets/d/1ywhXffNw3sNzngvblu4hxE6d2ZbSQ_GjN2JFfl3vYFc

- 30 cents to 1.3$ per txn on layer 2
- Max 20 transactions per rollup block

Scaling Playground Presentation:

https://nutberry.github.io/playground/