Aztec - Public<>Private Composability

Digging into the Aztec architecture

Aztec Labs

What is Aztec?

public/private zk-Rollup

zk-zk-Rollup

hybrid zk-Rollup

Actual zk-Rollup

encrypted zk-Rollup

Aztec is a privacy focused L2 on Ethereum

What is public execution?

Think Ethereum, Optimistic L2's, alt L1's etc...

Tx Request + Sig

A user sends a transaction request + signature to the p2p network via node

Ordering

The current block proposer or out of protocol builder picks transaction requests from the pool and orders into a block

Execution

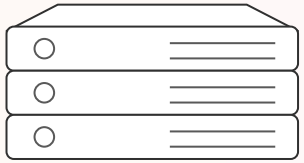
Each transaction request in the block is executed sequentially by the block proposer, modifying the state per the rules of the VM


AZTEC
EVM
PRIVACY
imgflip.com

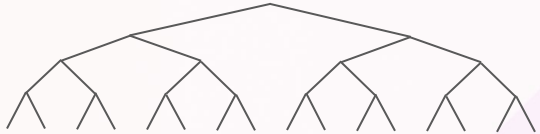
Features of Aztec?

- Permissionless contract deployment
- Private & Public states
- Private & Public function execution
- Composability:
  - calls between contracts
  - calls between private & public functions
  - messages between L2 & L1 via “portals”
- Account Abstraction

What is Aztec?

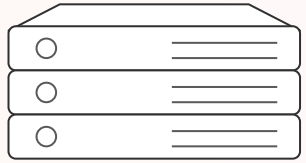


Eth Nodes

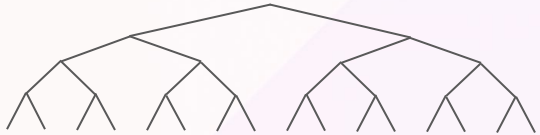


Ethereum state

L1 functions & states



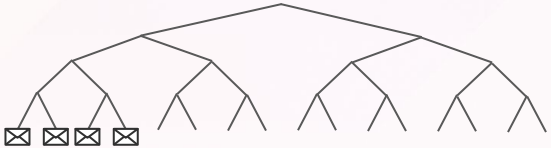
Rollup Sequencers



Public account-based state tree

‘Public L2’ functions & states

User device



Private, UTXO-based state tree

‘Private L2’ functions & states

7

Private state explained

UTXO State Model

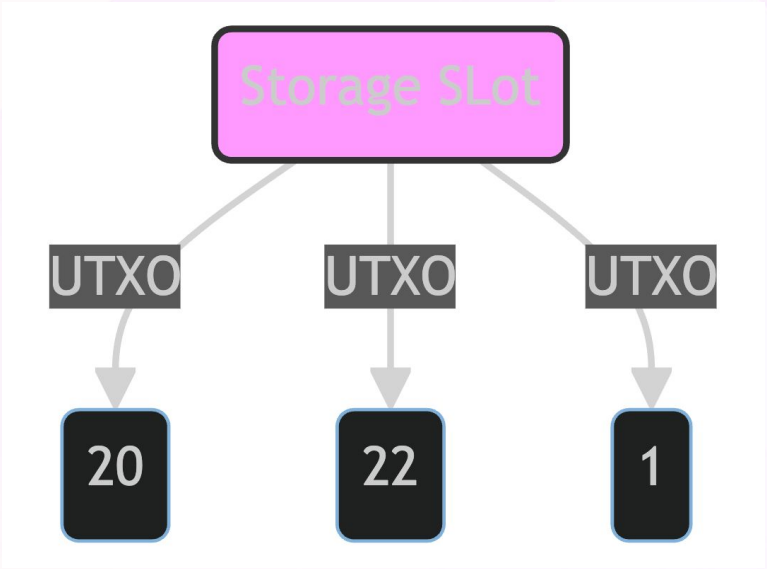
Private execution requires a different state model to prevent race conditions when accessing the same state.

Storage Slot's hold UTXO's

A storage slots value is represented by the sum of associated UTXO's.

State Diff's are easy!

A state diff for private execution is just the set of created and destroyed UTXO's.



Slot Value = 23

UTXOs are made easy

Token.balance(address)

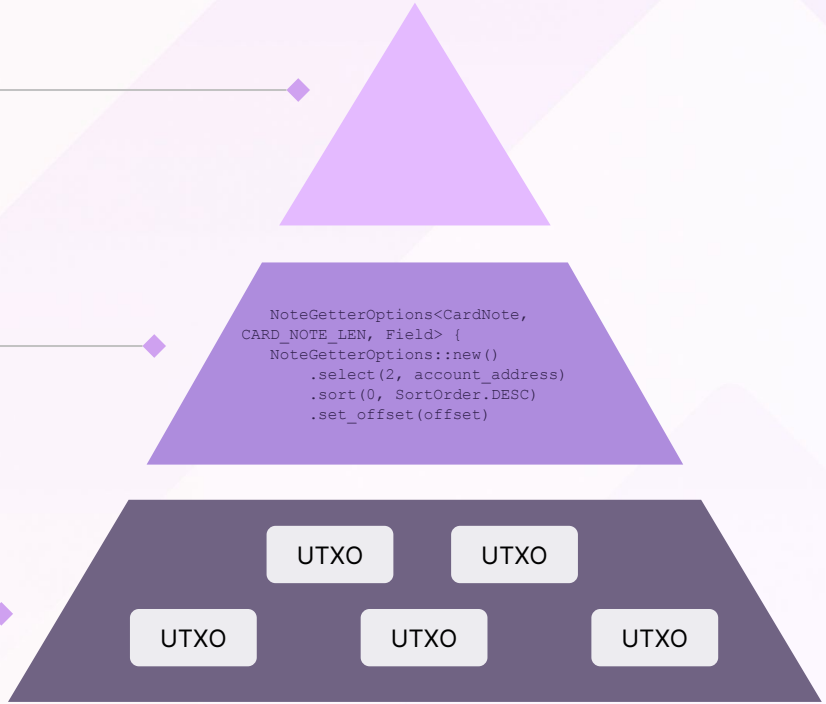
A dapp simply calls an unconstrained function to see the result, like magic.

Unconstrained functions

Contract developers write view logic for apps to decrypt and present UTXO's in a useful fashion to developers

Helper functions

Aztec.nr has built in functions to query + sort UTXO's in a contract



Private function example

```
#[aztec(private)]
fn discard_largest_card() {
    let storage = Storage::init(Context::private(&mut
context));
    let account = context.msg_sender();
    let card = actions::get_cards(storage.cards,
        NoteGetterOptions::new()
        .select(account)
        .sort(SortOrder.DESC)
        .set_limit(1))[0].unwrap()
    );

    assert(card.owner == account);

    actions::remove_card(storage.cards, card);
}
```

What is private execution?

Aztec, Aleo, Mina, Miden

Simulate Tx on Device

The user executes the transaction locally and creates a state diff.

Prove TX on Device

The user proves using a ZKP they executed the transaction correctly, following the rule of the “vm” and that the state diff is correct

TX proof + state diff

The user sends the state diff and proof to the p2p network for inclusion.

Rollup + State Updates

The zk proof is recursively verified in a rollup proof, and the state diff applied to the chain.

Benefits of private execution for devs

Privacy

Local execution via a ZKP allows private inputs, private state and encryption of the state diff

No Gas!

As private functions are executed locally, gas is not paid for compute, only the size of the state diff they apply.

Account Abstraction

Contracts define the rules under which their state can be updated,

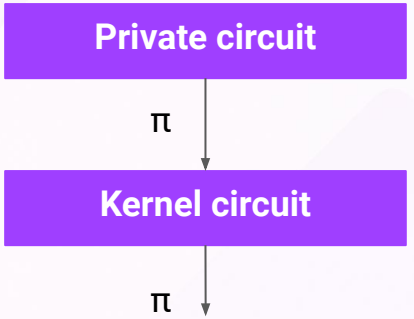
How do public and private functions work?

Private

- Executed and proved locally by the user.
- State updates + proof sent to the sequencer – (commitments, nullifiers, deployments + cross chain messages)
- Sequencer verifies proof, adds to rollup + updates state trees

Public

- User sends their public function call data as part of their transaction
- Executed by sequencer, then a proof is made by the proving network



No one learns which function was executed

When to use a private function?

- When working with private state (e.g. salary payments)
- When working with sensitive data, not stored on chain (e.g passports, addresses, etc)
- When you want to hide function execution or the algorithm you are running.
- When creating proof against historic data **(I owned this NFT at block X)**
- When you want to move something from public to private state

"Someone did something to some state in some function of some contract"

When to use a public function?

- When interacting with public state or shared state (Uniswap pools)
- When interacting with current head of state
- e.g. making use of current timestamp, current block number etc.
- When broadcasting information to everyone
- When un-shielding assets (go from private to public state)
- to show the cards you have at the end of the game of poker,

“Ask a sequencer to execute a function and prove they did it correctly”

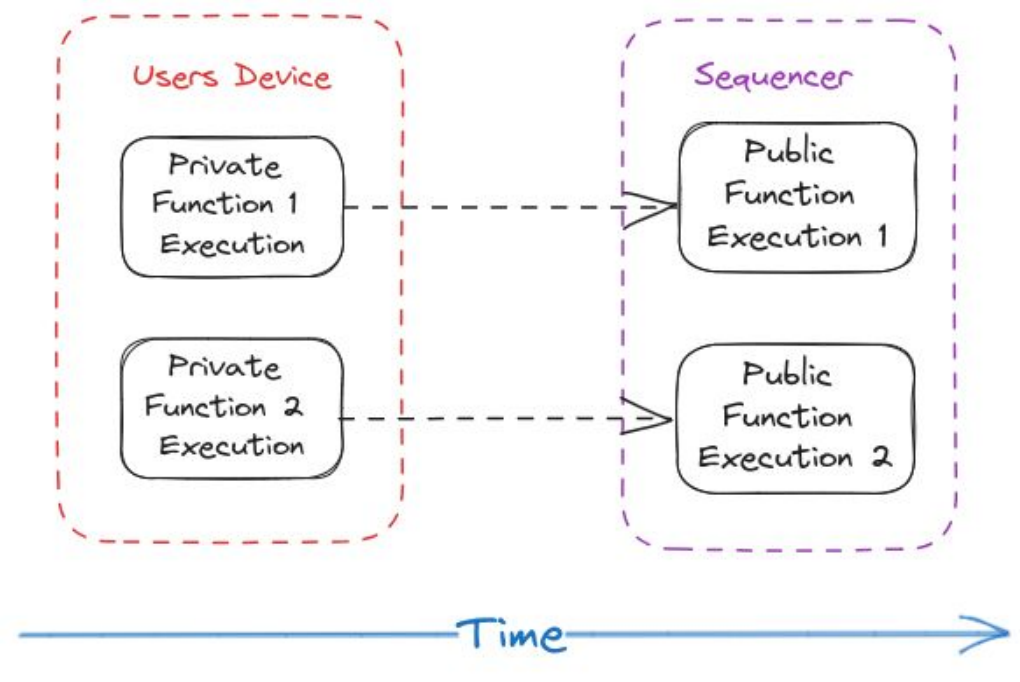
How do public and private functions work together?

Client Side Execution = privacy

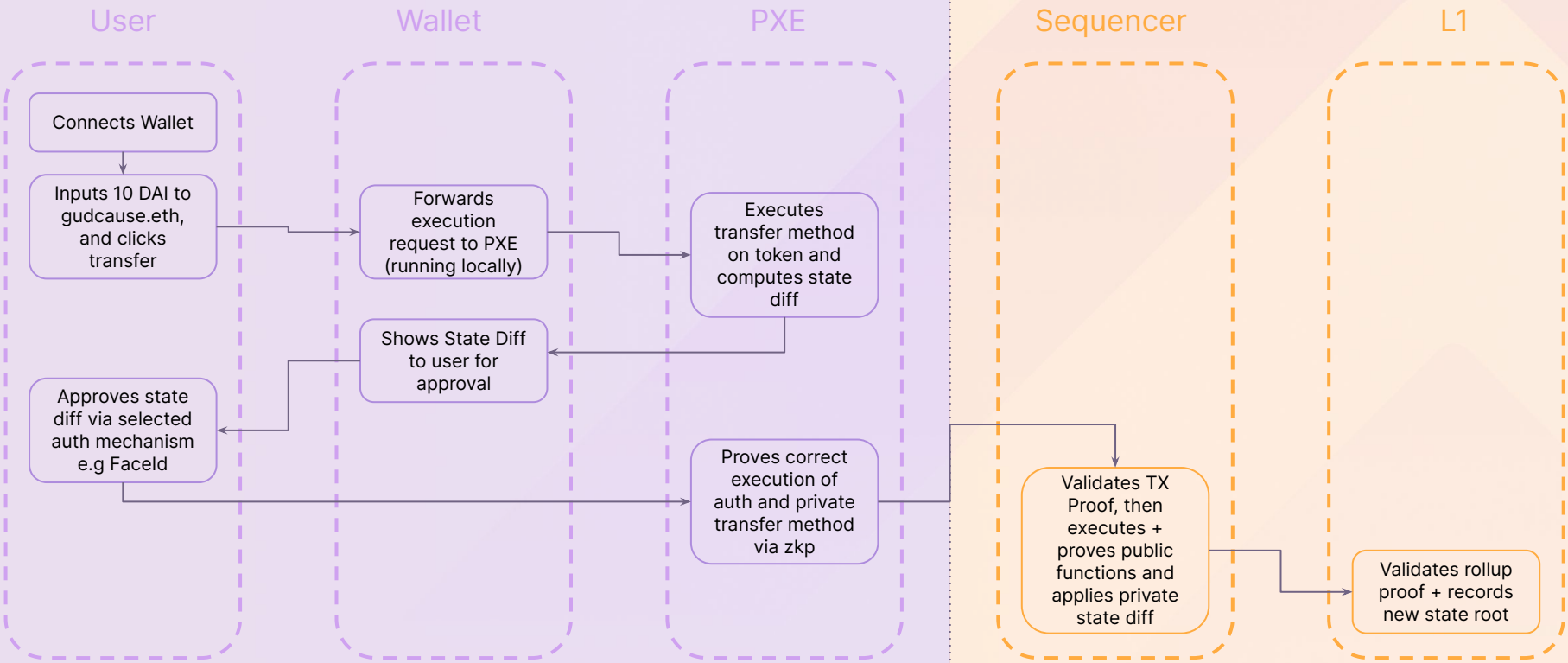
Private execution is proven on a users device as snark circuits, public execution is proven as a VM by the sequencer

Atomic Execution

A transaction is asynchronous yet atomic, if any function fails the tx will fail.



Transaction lifecycle



Aztec Labs

Code Example

```
#[aztec(private)]
fn unshield(
    from: AztecAddress,
    to: AztecAddress,
    amount: Field,
    nonce: Field,
) -> Field {
    if (from.address != context.msg_sender()) {
        assert_current_call_valid_authwit(&mut context, from);
    } else {
        assert(nonce == 0, "invalid nonce");
    }

    storage.balances.at(from).sub(SafeU120::new(amount));

    let selector = compute_selector("_increase_public_balance((Field),Field)");
    let _void = context.call_public_function(context.this_address(), selector, [to.address, amount]);

    1
}
```

Public and Private State

Different merkle trees for each

- Public Data Tree
- Note Hash Tree
- Nullifier Tree

Different “kernel” circuits too

Unaccessible across different contexts... or is it? 👀

Code Example

```
#[aztec(public)]
fn shield(
    from: AztecAddress,
    amount: Field,
    secret_hash: Field,
    nonce: Field,
) -> Field {
    if (from.address != context.msg_sender()) {
        // The redeem is only spendable once, so we need to ensure that you cannot insert multiple shields from the same message.
        assert_current_call_valid_authwit_public(&mut context, from);
    } else {
        assert(nonce == 0, "invalid nonce");
    }

    let amount = SafeU120::new(amount);
    let from_balance = storage.public_balances.at(from.address).read().sub(amount);

    let pending_shields = storage.pending_shields;
    let mut note = TransparentNote::new(amount.value as Field, secret_hash);

    storage.public_balances.at(from.address).write(from_balance);
    pending_shields.insert_from_public(&mut note);
    1
}
```

Code Example

```
#[aztec(private)]
fn redeem_shield(
    to: AztecAddress,
    amount: Field,
    secret: Field,
) -> Field {
    let pending_shields = storage.pending_shields;
    let secret_hash = compute_secret_hash(secret);
    // Get 1 note (set_limit(1)) which has amount stored in field with index 0 (select(0, amount)) and secret_hash
    // stored in field with index 1 (select(1, secret_hash)).
    let options = NoteGetterOptions::new().select(0, amount).select(1, secret_hash).set_limit(1);
    let notes = pending_shields.get_notes(options);
    let note = notes[0].unwrap_unchecked();
    // Remove the note from the pending shields set
    pending_shields.remove(note);

    // Add the token note to user's balances set
    storage.balances.at(to).add(SafeU120::new(amount));

    1
}
```

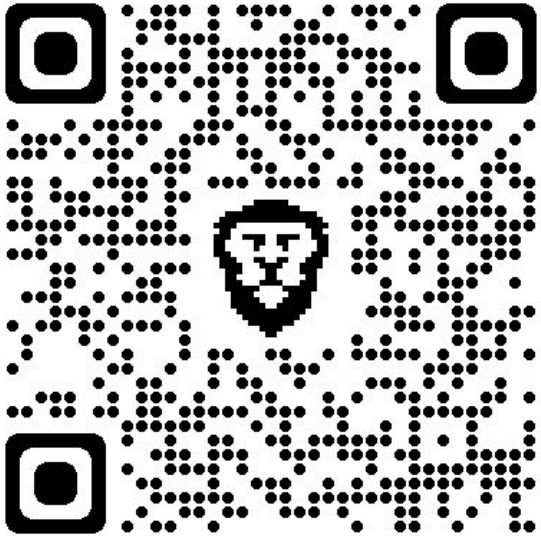
Wat buidl?

- Account contracts
  - Private access control rules
- Info hiding games
  - Poker (any card games)
- Identity
  - Private voting
  - Hide votes, hide tally, reveal result
- Defi applications
  - Private DEX
  - Borrowing / lending
- Many more, just starting to explore the design space

Aztec is **Ethereum, Encrypted**

Docs

docs.aztec.network



GitHub

github.com/AztecProtocol

Me! Contact Info:

@critesjosh_