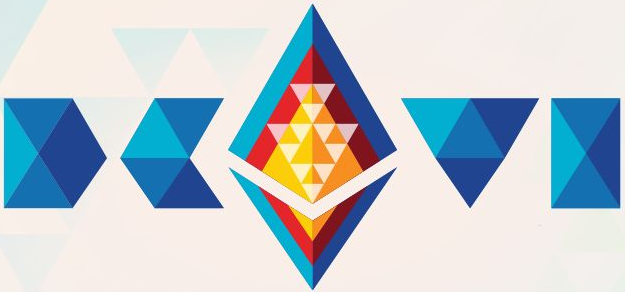


Onboard The World Into Your Rollup dApp with BLS Wallet

Jacob Caban-Tomski &
James Zaki

Privacy & Scaling Explorations, Ethereum Foundation

What We'll Cover (Agenda)

1. Team & Project Outcomes
2. BLS Signatures & Aggregation
3. BLS Wallet Today (w/ Examples)
   a. Multicall
   b. Sponsored Transactions
   c. Account Recovery
   d. Upgradable
4. Where to next?
5. Questions

Section 1

Team & Project Outcomes

Jacob Caban-Tomski

Blake Duncan

John Guilding

Andrew Morris

Kautuk Kundan

James Zaki

PSE

ZKP

L2

Privacy & Scaling Exploration

Project Outcomes

- Enable low cost dApps on L2s/Rollups
  - Reduce transaction data rolled up to L1

Project Outcomes

- Enable low cost dApps on L2s/Rollups
  - Reduce transaction data rolled up to L1

150 ERC20 Transfers	*Txn Size (Bytes)*	*Txn Size Reduction*	*Txn*
Regular	26850 (179 * 150)		0x8d64...7589
BLS w/ Pubkey Hash	19670	**26.7%**	0x5a1b....7c87

- Need to measure L1 gas costs on L2 mainnets (Arbitrum, Optimism, others)
- How low can we go? Address book, other indexing...

Project Outcomes

- Enable low cost dApps on L2s/Rollups
  - Reduce transaction data rolled up to L1
- Improved Wallets
  - Account Recovery
  - Upgradable Functionality

Project Outcomes

- Enable low cost dApps on L2s/Rollups
  - Reduce transaction data rolled up to L1
- Improved Wallets
  - Account Recovery
  - Upgradable Functionality
- Make dApps easier to use
  - Multicall
  - Sponsored Transactions

Section 2

BLS Signatures & Aggregation

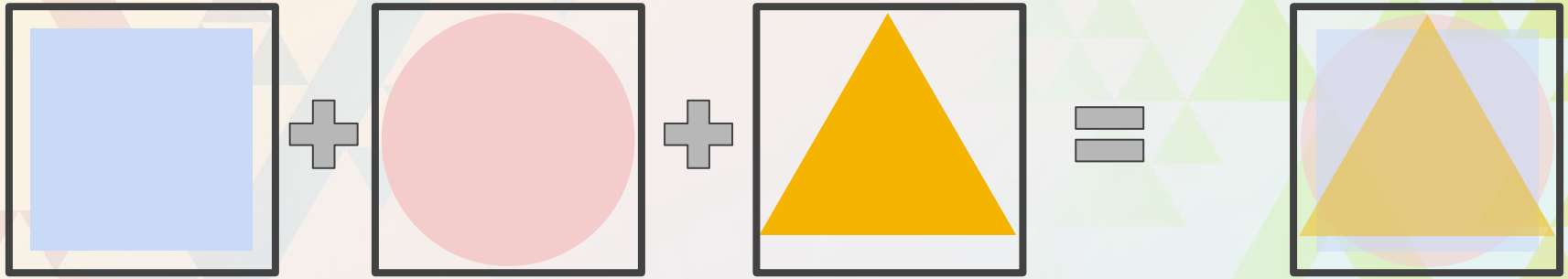
BLS: Boneh-Lynn-Shacham (2002)

Pairing cryptography based signature scheme used in Consensus Layer, ZCash, & other projects

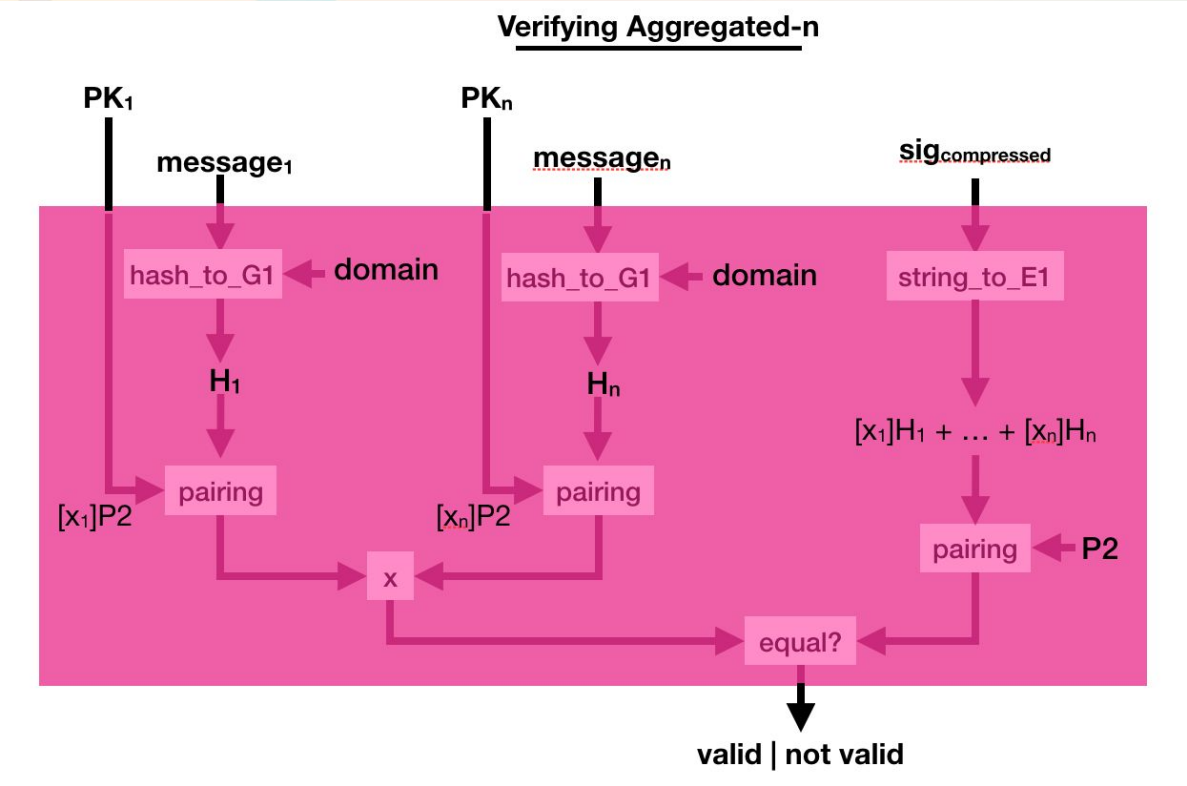
- Deterministic for a key & message
- Validators use BLS to sign protocol messages (BLS-12-381)
- Execution layer supports BN-254 via EIP197
  - *Maybe* BLS-12-381 in future via EIP2537

And most importantly...

BLS Signature Aggregation



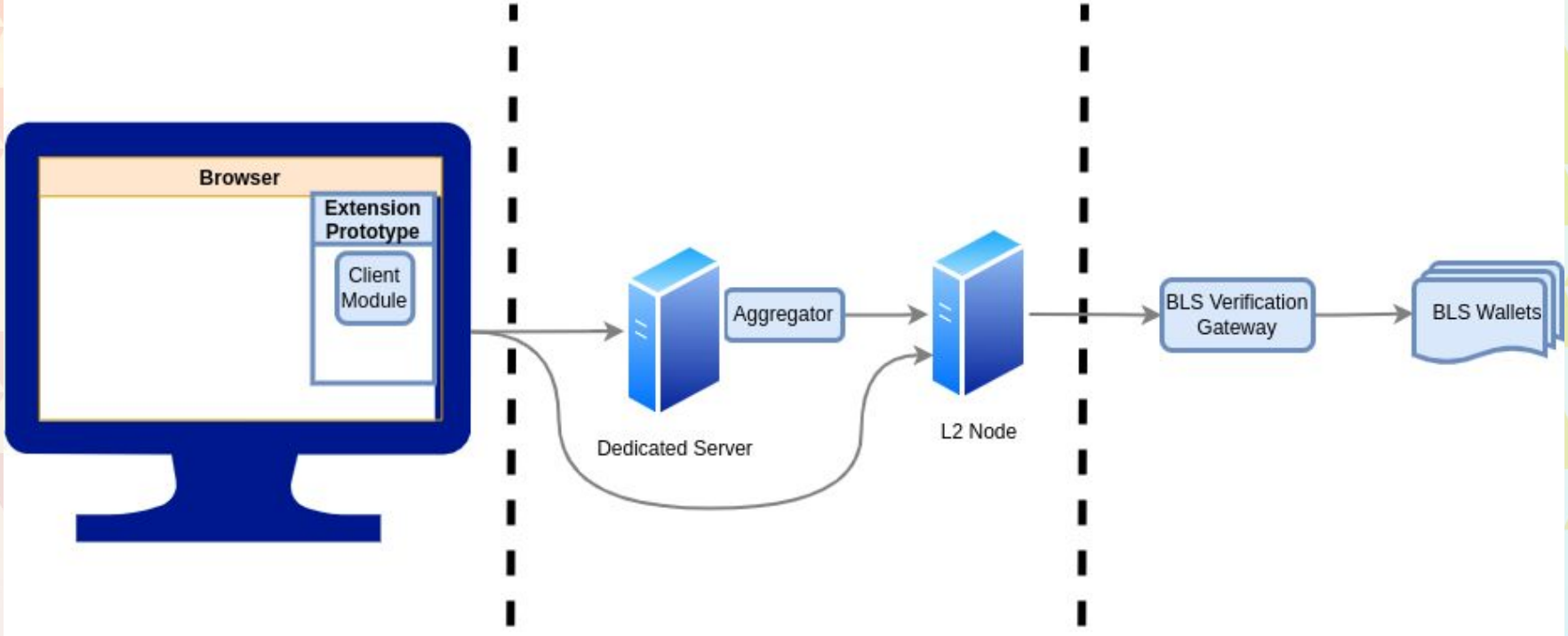
- Many signatures -> one signature
- Great for reducing data rolled up to L1
  - Currently: Transaction data + ECDSA signature **per transaction**
  - With BLS: Transaction data + **single** aggregated BLS Signature **for all transactions**



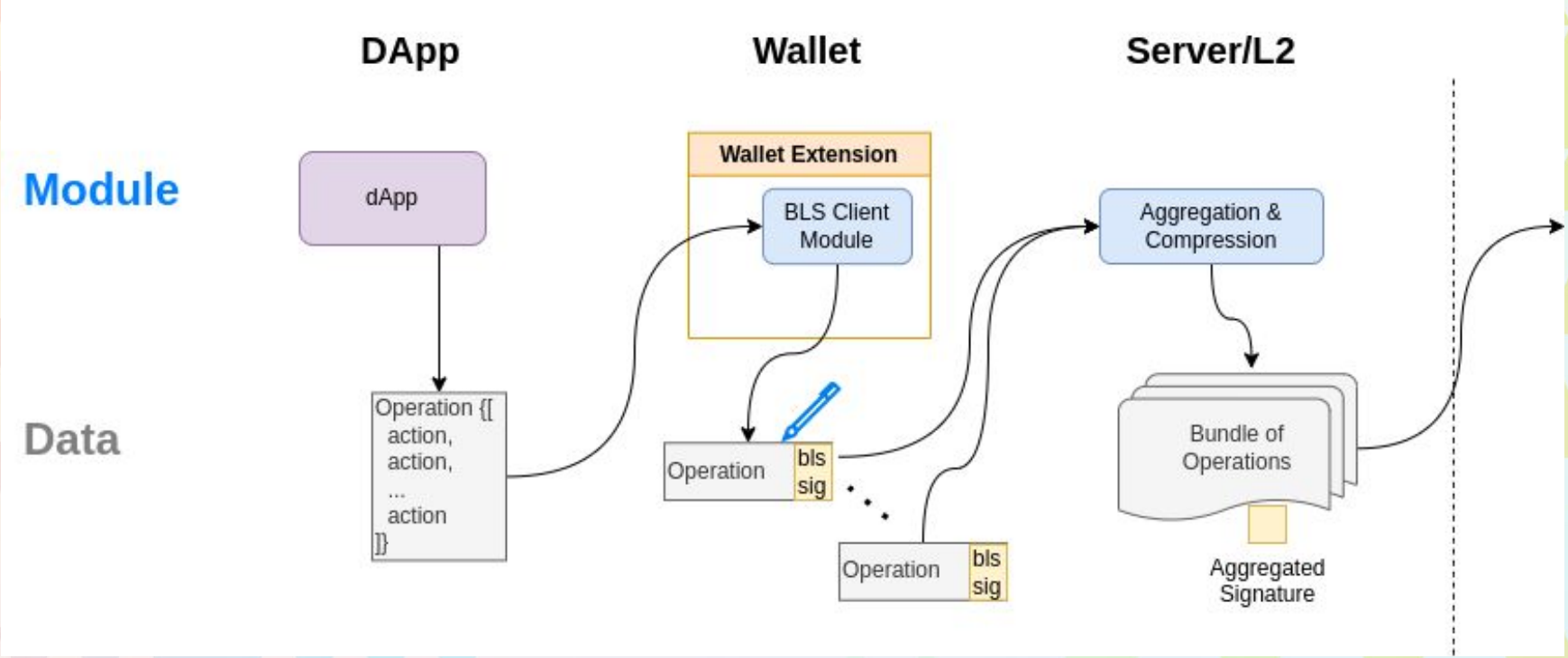
https://www.cryptologie.net/article/472/what-is-the-bls-signature-scheme/

Section 3

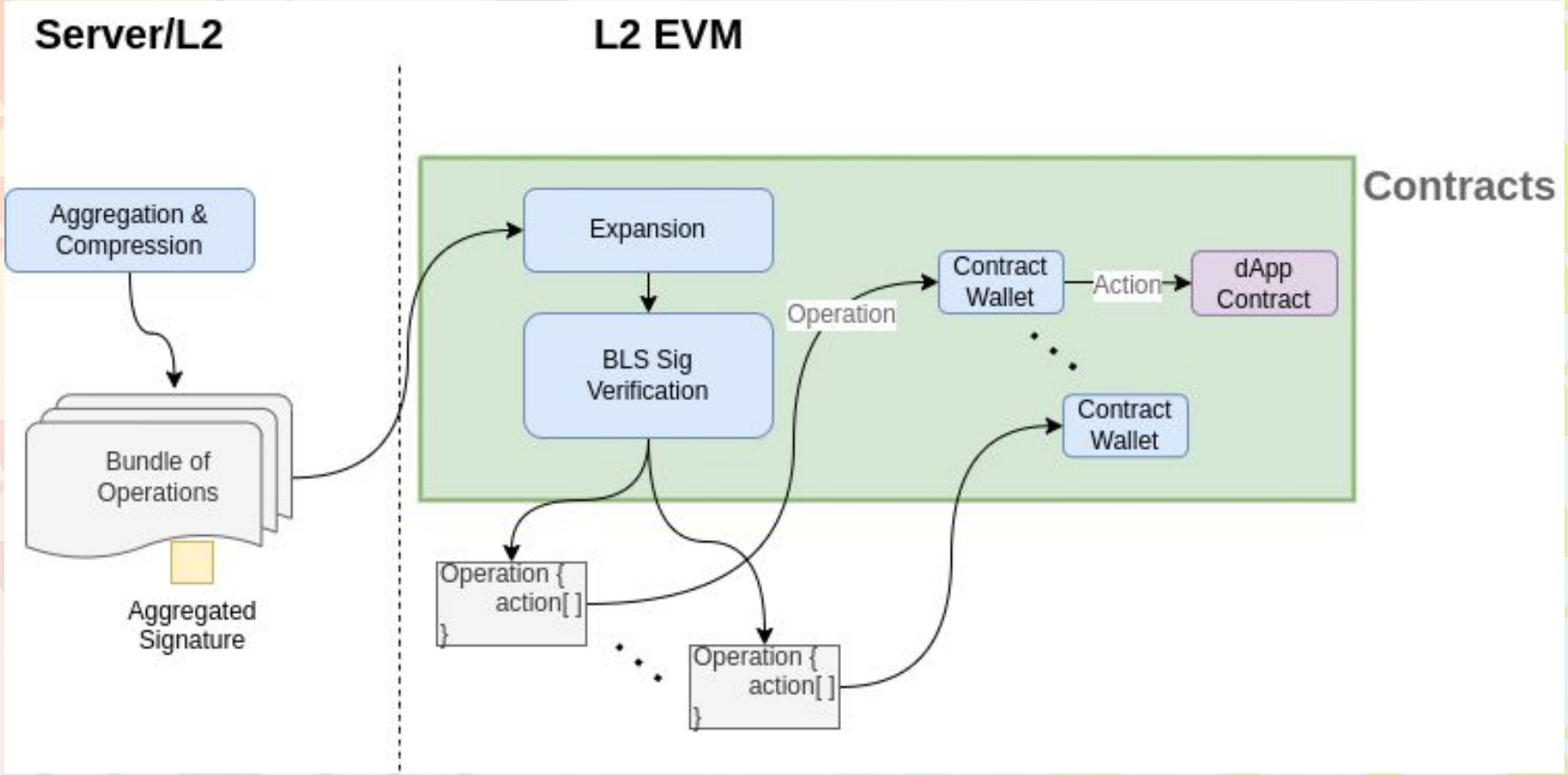
BLS Wallet Today



https://github.com/web3well/bls-wallet/blob/main/docs/system_overview.md



https://github.com/web3well/bls-wallet/blob/main/docs/system_overview.md

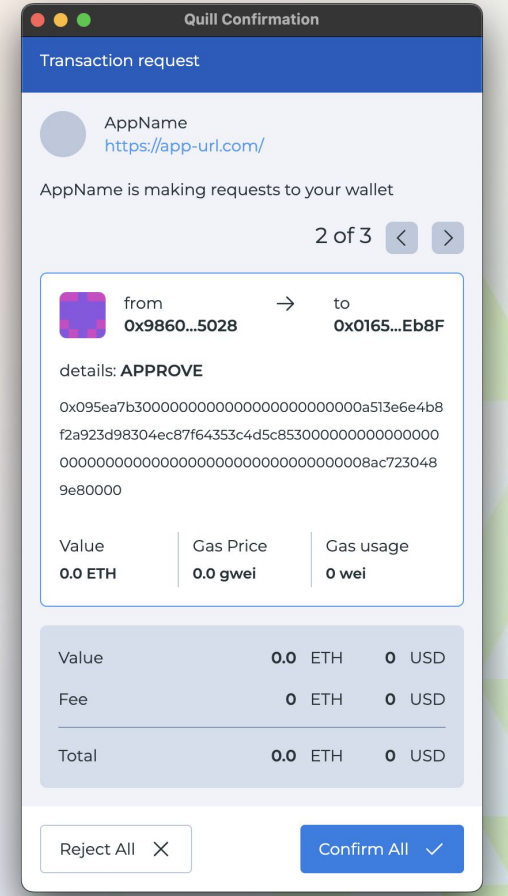


https://github.com/web3well/bls-wallet/blob/main/docs/system_overview.md

Multicall/action

npm v0.7.3 **npm install bls-wallet-clients**

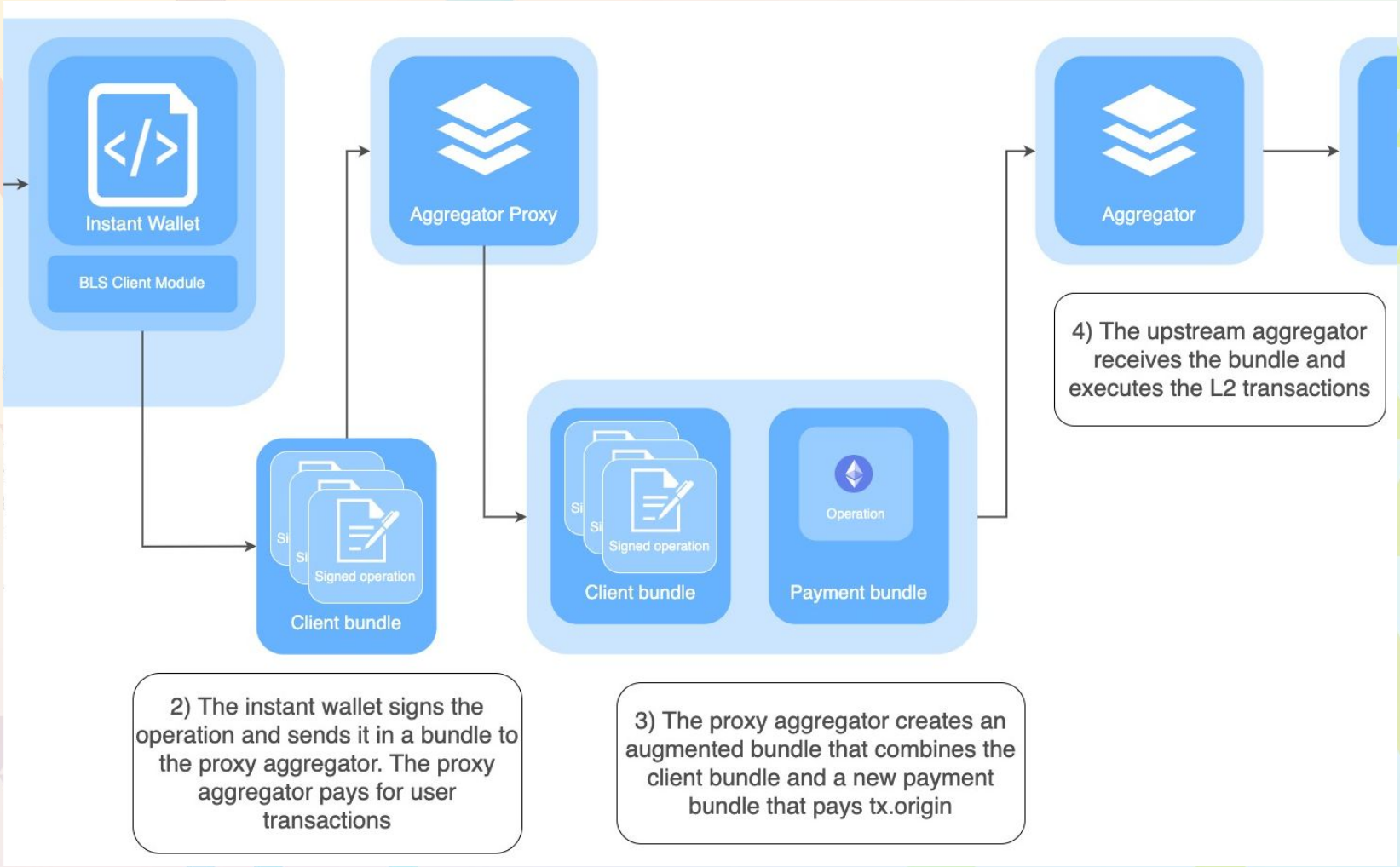
```
const bundle = wallet.sign({
  nonce: await wallet.Nonce(),
  // All actions in this operation are atomic
  actions: [
    {
      ethValue: 0,
      contractAddress: erc20Contract.address,
      encodedFunction: erc20Contract.address.interface.encodeFunctionData(
        "approve",
        [dexContract.address, amount],
      ),
    },
    {
      ethValue: 0,
      contractAddress: dexContract.address,
      encodedFunction: dexContract.address.interface.encodeFunctionData(
        "swap", [
          erc20Contract.address,
          amount,
          otherERC20Contract.address
        ],
      ),
    },
  ],
});
```

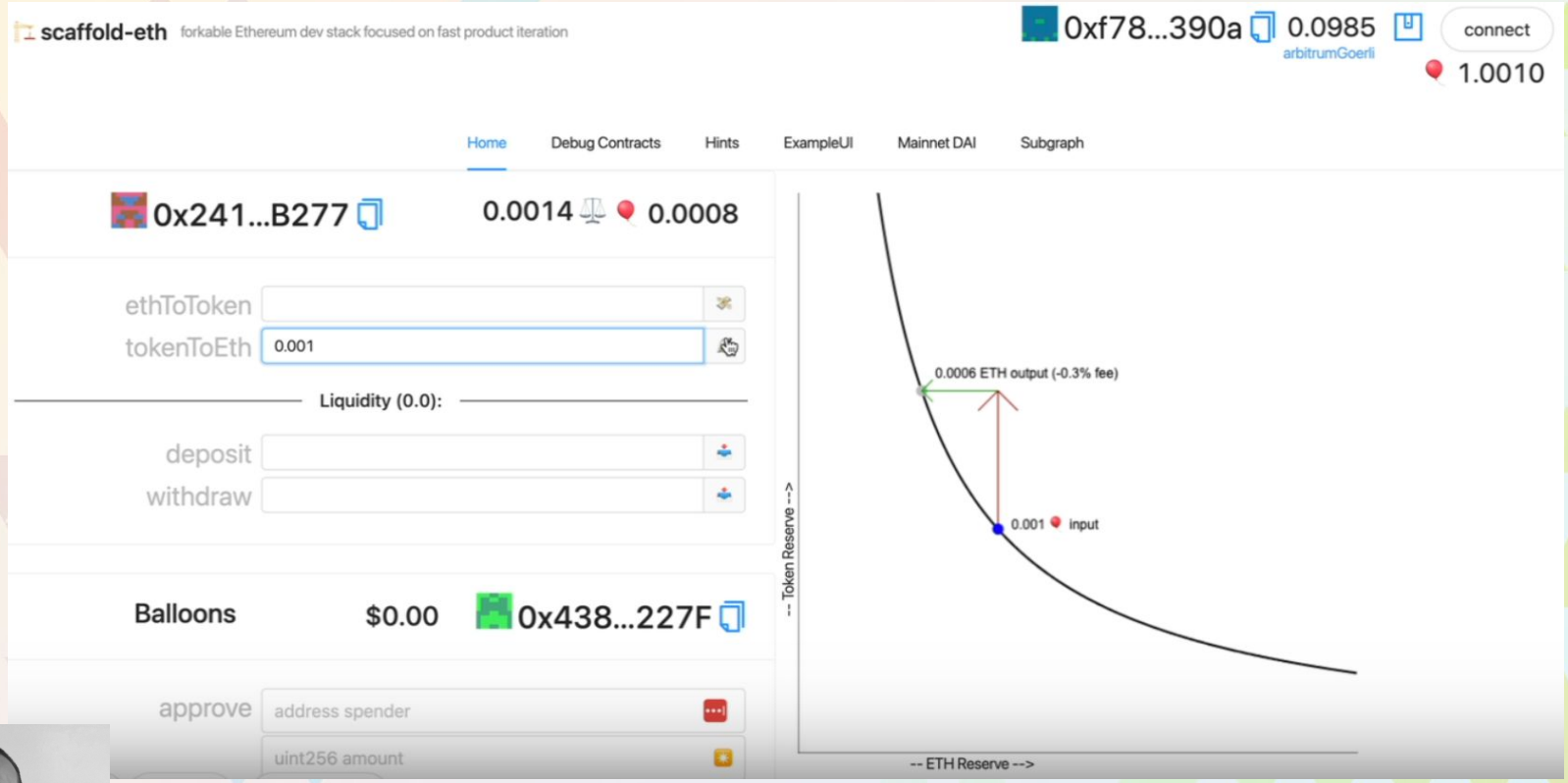


QUILL

https://github.com/web3well/bls-wallet/blob/main/docs/use_bls_wallet_dapp.md

Sponsored Transaction


Instant Wallet
BLS Client Module
Aggregator Proxy
Aggregator
Signed operation
Client bundle
Signed operation
Client bundle
Operation
Payment bundle
4) The upstream aggregator receives the bundle and executes the L2 transactions
2) The instant wallet signs the operation and sends it in a bundle to the proxy aggregator. The proxy aggregator pays for user transactions
3) The proxy aggregator creates an augmented bundle that combines the client bundle and a new payment bundle that pays tx.origin



Repo: https://github.com/JohnGuilding/single-pool-dex
App: https://single-pool-dex-react-app.vercel.app/

Sponsored Transaction via Contract

- Aggregator checks if ETH/token balance is higher post bundle execution
- Contract can pay *tx.origin*
- Can gate via allowlist, NFT ownership, ZKP proof
- Allows anyone (MEV Bots?) to be a bundle submitter
- Still more research to be done

Recover

Recover

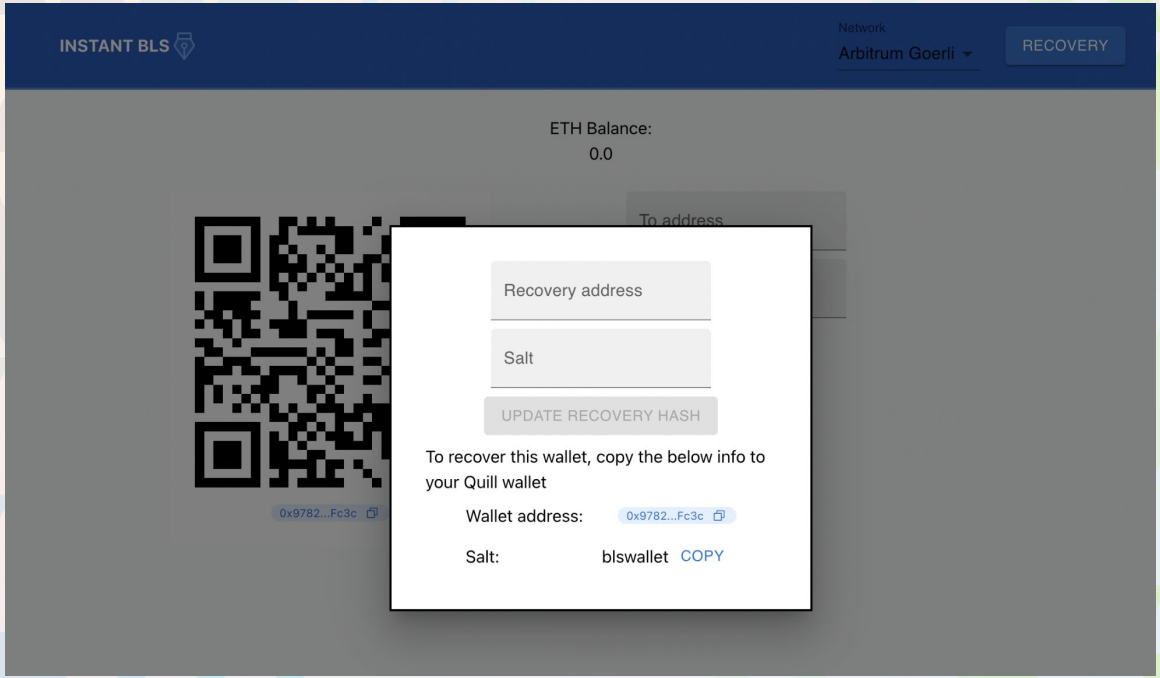
```
/**
Recovers a wallet, setting a new bls public key.
@param walletAddressSignature signature of message containing only the wallet address
@param blsKeyHash calling wallet's bls public key hash
@param salt used in the recovery hash
@param newBLSKey to set as the wallet's bls public key
 */
function recoverWallet(
    uint256[2] memory walletAddressSignature,
    bytes32 blsKeyHash,
    bytes32 salt,
    uint256[BLS_KEY_LEN] memory newBLSKey
) public {
    IWallet wallet = walletFromHash[blsKeyHash];
    bytes32 recoveryHash = keccak256(
        abi.encodePacked(msg.sender, blsKeyHash, salt)
    );
    if (recoveryHash == wallet.recoveryHash()) {
        safeSetWallet(walletAddressSignature, newBLSKey, wallet);
        wallet.recover();
    }
}
```

Onboarding UX (using recover)

Wallet

```
function setRecoveryHash(bytes32 hash) public onlyThis {
    if (recoveryHash == bytes32(0)) {
        recoveryHash = hash;
        clearPendingRecoveryHash();
        emit RecoveryHashUpdated(bytes32(0), recoveryHash);
    }
    else {
        pendingRecoveryHash = hash;
        pendingRecoveryHashTime = block.timestamp + 604800;
        emit PendingRecoveryHashSet(pendingRecoveryHash);
    }
}
```

Onboarding UX (using recover)



https://medium.com/@blakecduncan/how-does-wallet-recovery-work-2c0f380192e8

Section 4

Where to next?

Less rollup data to L1

- Aggregate signatures
  - Leverage Account Abstraction? EIP2938
  - Focus on BLS only contract wallet

The journey so far

Less rollup data to L1

- Aggregate signatures
- Preliminary optimisations
  - Parameter deduplication

The journey so far

Less rollup data to L1

- Aggregate signatures
- Preliminary optimisations
- Wallet features
  - Sponsored txs, multi-action
  - Recoverable, upgradable

The journey so far

Less rollup data to L1

- Aggregate signatures
- Preliminary optimisations
- Wallet features

- EIP4337 enters the arena!
  - Bls-wallet contracts going for audit
  - Modify to be compatible

The journey so far

Less rollup data to L1

Where to next?

Less rollup data to L1

- Aggregate signatures
  - For lowest tx costs
  - 4337 compatibility
- Payment options
  - Direct (alt mem pool optional)

Where to next?

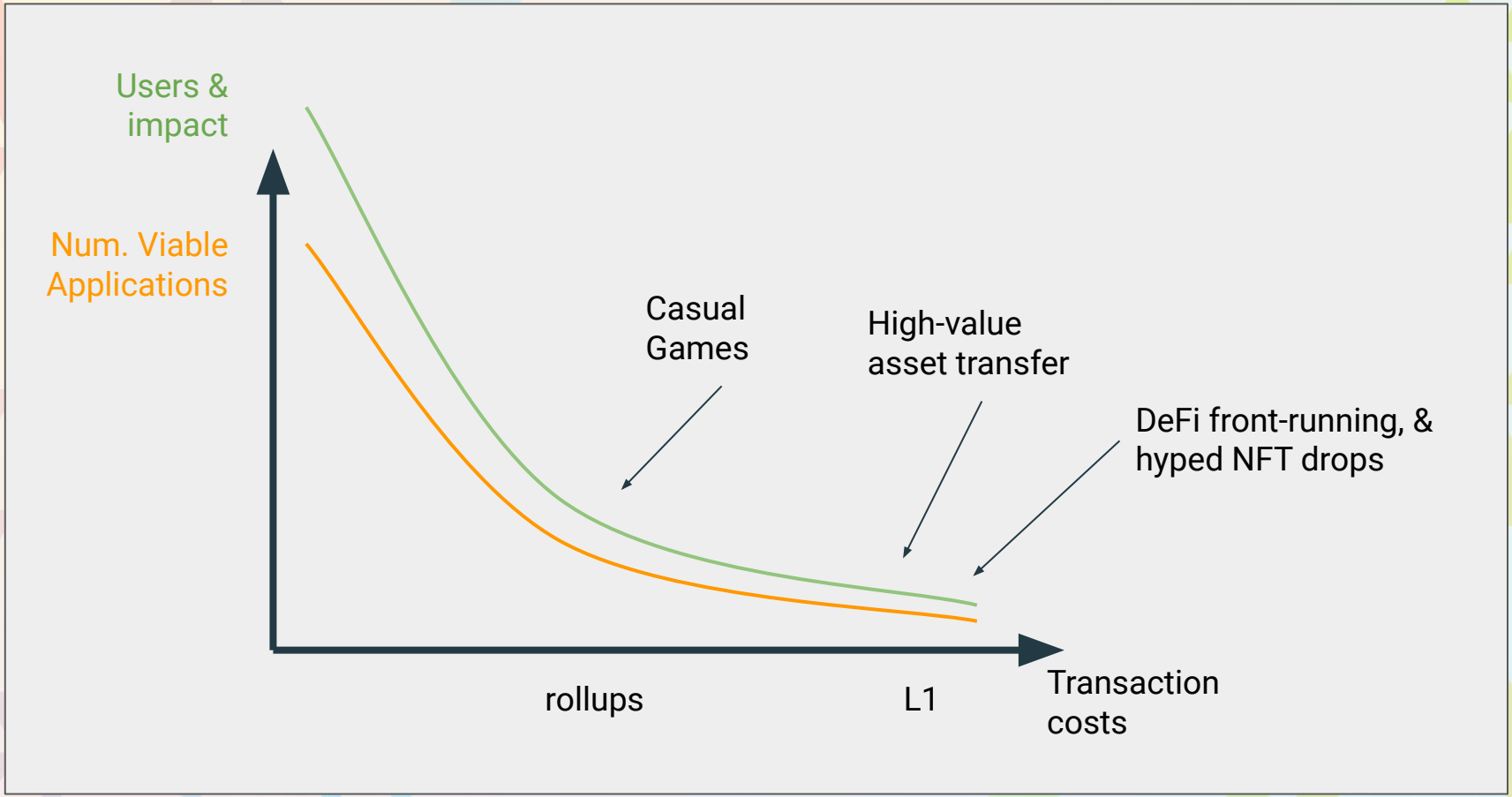
Less rollup data to L1

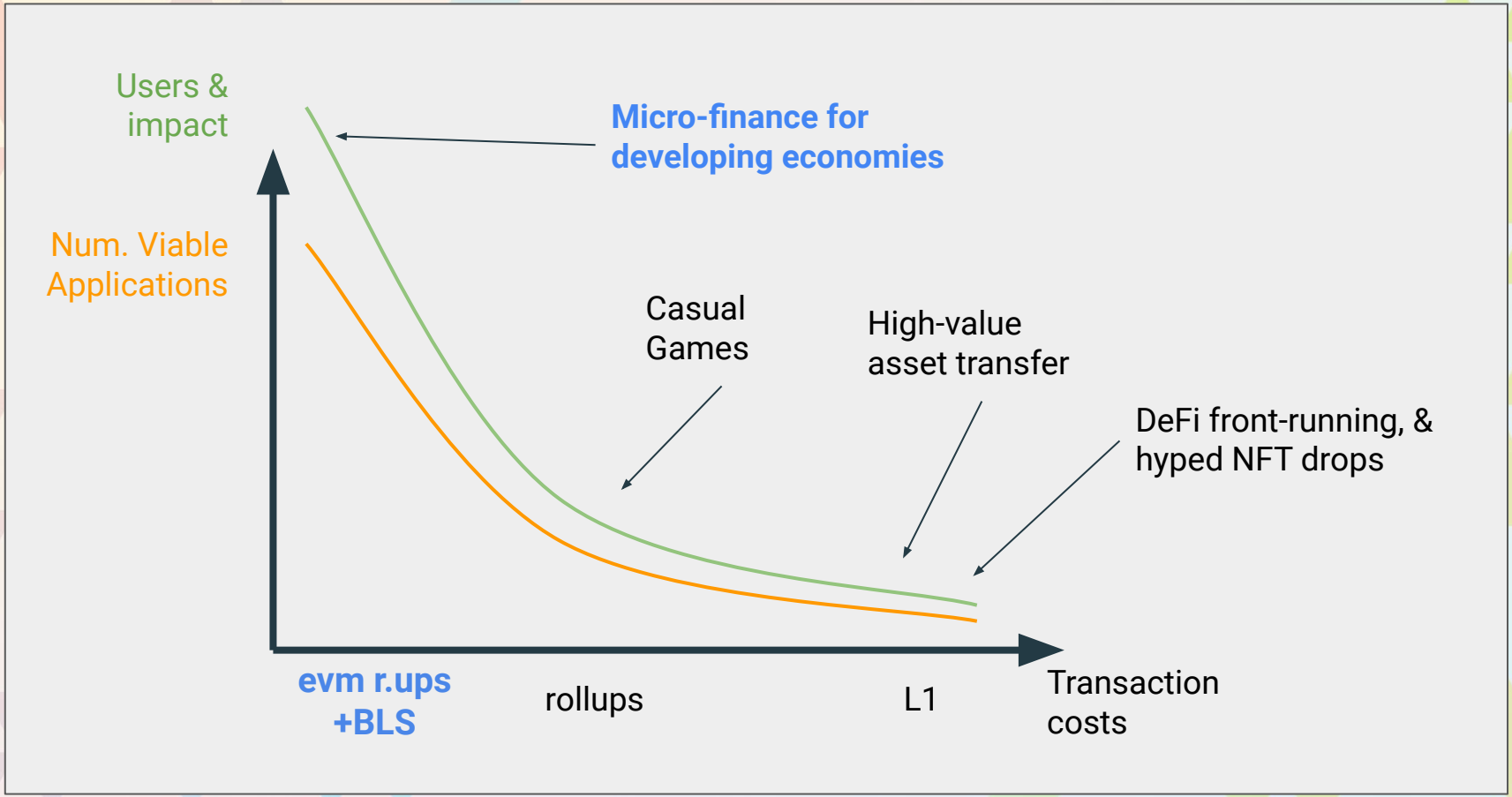
- Aggregate signatures
- Payment options

- Further optimisations
  - Small UserOp (gas params optional)
  - Public key mapping
  - Floating point
- Wallet features
  - Consider extracting to modules
- Benefit from EIP4844

Where to next?

Lower the $ entry-barrier,
Increase # viable solutions.


Users & impact
Num. Viable Applications
Casual Games
High-value asset transfer
DeFi front-running, & hyped NFT drops
rollups
L1
Transaction costs


Users & impact
Num. Viable Applications
Micro-finance for developing economies
Casual Games
High-value asset transfer
DeFi front-running, & hyped NFT drops
evm r.ups +BLS
rollups
L1
Transaction costs

Here’s the timeline.

BLS Wallet

Live on Arbitrum Nitro Goerli testnet!

Arbitrum/Optimism after audit fixes

Here’s the timeline.

BLS Wallet

Live on Arbitrum Nitro Goerli testnet!

Arbitrum/Optimism after audit fixes

Wallet Adoption

Direct integration(s) to help priced-out users.

Support web3 wallet integrations, via EIP4337 or directly.

Here’s the timeline.

BLS Wallet

Live on Arbitrum Nitro Goerli testnet!

Arbitrum/Optimism after audit fixes

Wallet Adoption

Direct integration(s) to help priced-out users.

Support web3 wallet integrations, via EIP4337 or directly.

Further Optimisations

Continue to make txs cheaper for the wallets that have integrated

Here’s the timeline.

BLS Wallet

Live on Arbitrum Nitro Goerli testnet!

Arbitrum/Optimism after audit fixes

Wallet Adoption

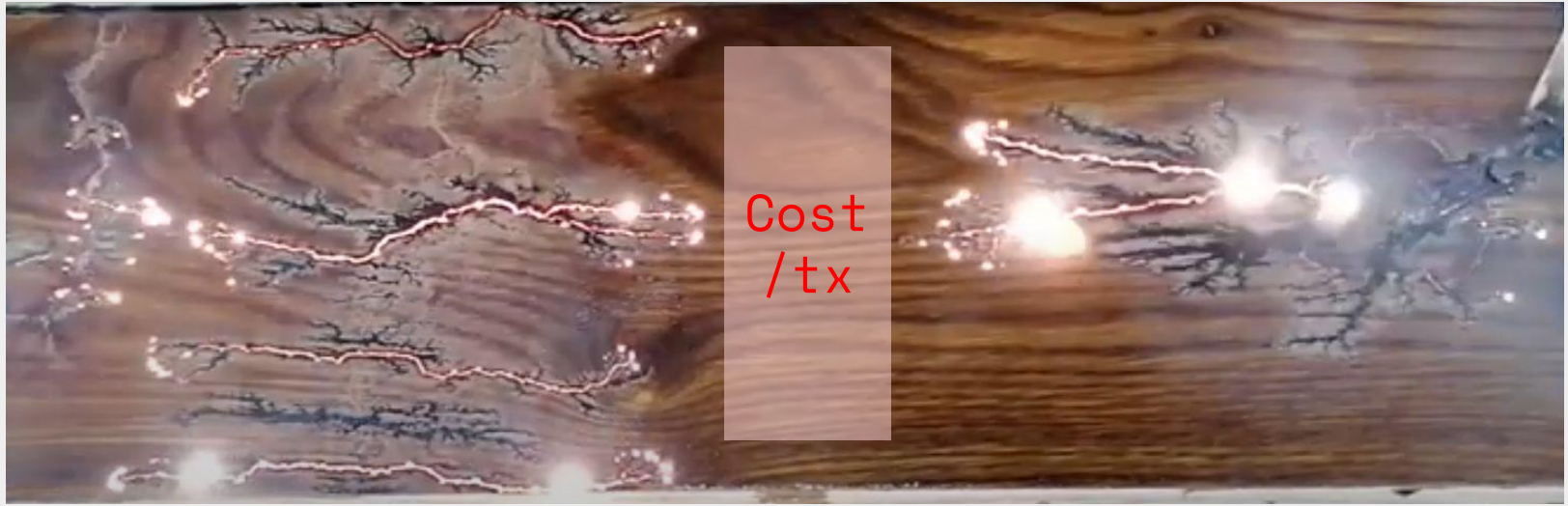
Direct integration(s) to help priced-out users.

Support web3 wallet integrations, via EIP4337 or directly.

Further Optimisations

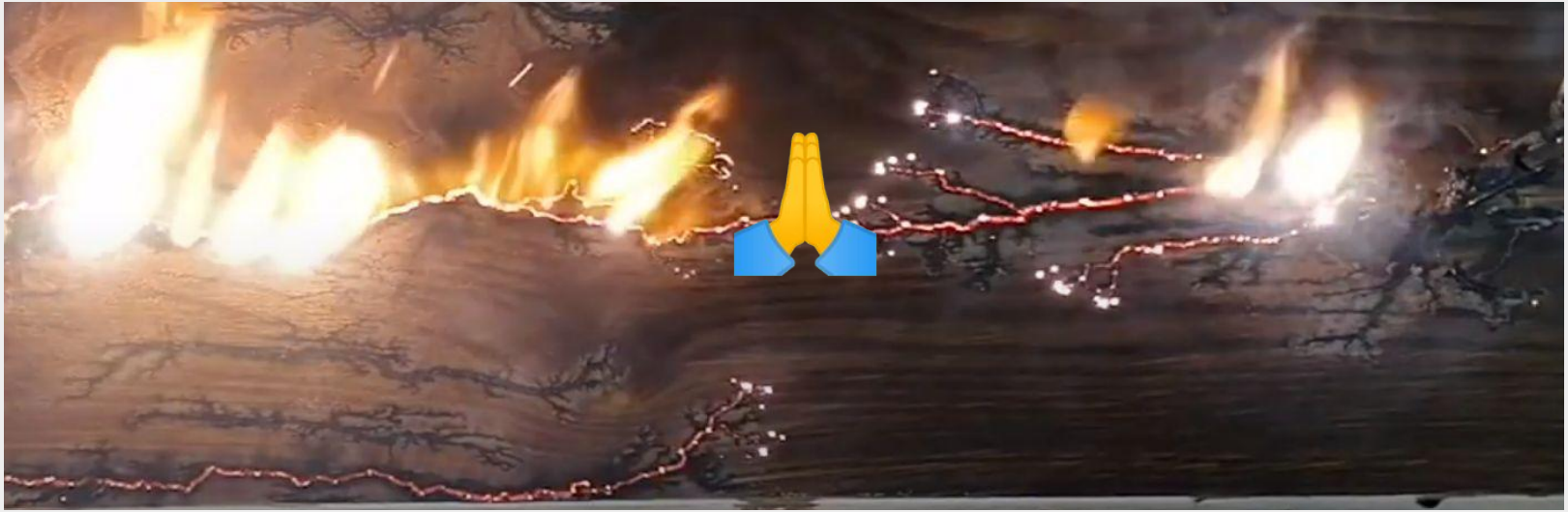
Continue to make txs cheaper for the wallets that have integrated

Things made even better with EIP4844 / Proto-Danksharding.



Real world problems

Web3 solutions

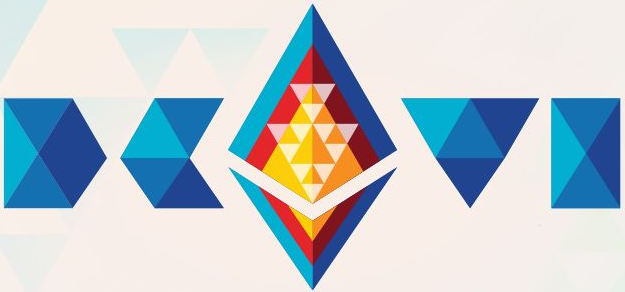


Real world problems solved by web3 solutions

https://blswallet.org

Learn More

In-browser demo,
Github,
Discord



Thank you!

Jacob Caban-Tomski &
James Zaki

Privacy & Scaling Explorations, Ethereum Foundation