PROBLEM STATEMENT

Navigating cross-chain bridging today poses several challenges

Notoriously vulnerable to hacks

Lack of support for non-EVM chains

No consistency in bridge interfaces nor common standard

OUR SOLUTION:

First true atomic bridge-less cross-chain assets swap.

SecuredJamon protocol allows users to make atomic swap between their assets in between NEAR and any other chain (even with not evm-compatible).

Protocol exchange stages:

1

Seller who wants to exchange his asset at any chain (i.e. DAI at Etherium) to tokens in NEAR(i.e USDC) creates offer with our market contract

2

In time of offer creation market contract publish unique derived address for this offer (Derived market address)

3

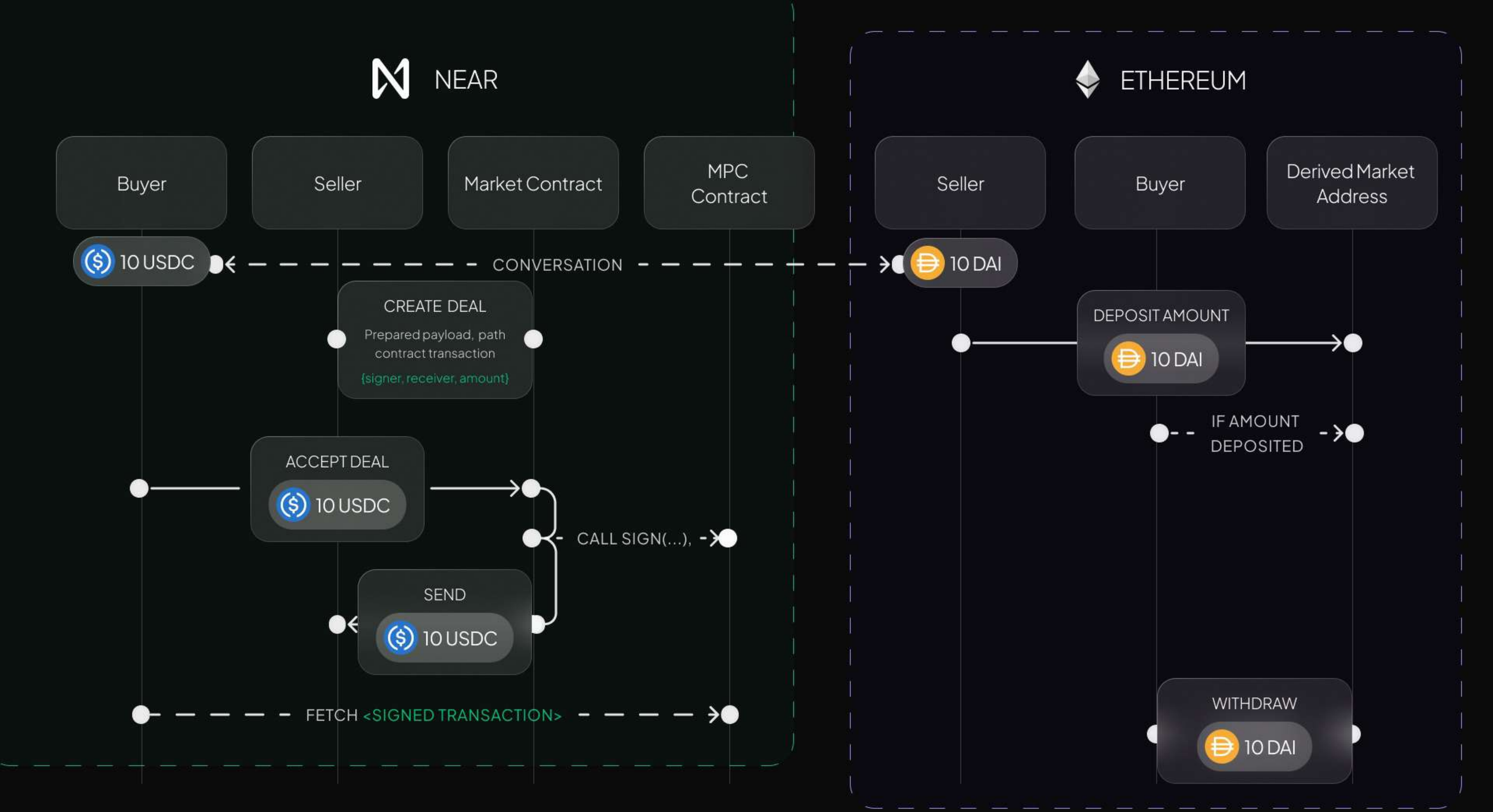
Seller put his DAI to Derived market address at Etherium network (for now these money controlled only by original market contract on NEAR via MPC)

4

Buyer put his USDC to market contract to accept offer. In single transaction NEAR USDC goes to seller & market contract unlock possibility to sign withdraw from Derived market address in Etherium

5

Buyer (and only him) may call market contract any time to create signature to withdraw money from Derived market address at ETH via NEAR MPC.


NEAR
ETHEREUM
Buyer
Seller
Market Contract
MPC Contract
Seller
Buyer
Derived Market Address
10 USDC
CONVERSATION
10 DAI
CREATE DEAL
Prepared payload, path contract transaction
{signer, receiver, amount}
DEPOSIT AMOUNT
10 DAI
IF AMOUNT DEPOSITED
ACCEPT DEAL
10 USDC
CALL SIGN(...),
SEND
10 USDC
WITHDRAW
10 DAI
FETCH <SIGNED TRANSACTION>

WhyNFT Team

Sergey Miller
ML Engineer, Joom

discord: sergmiller

Ivan Cheprasov
SWE, Fluence Labs

discord: Half-PhD#4215

Olga Abramova
Designer, OKAD

Live Demo

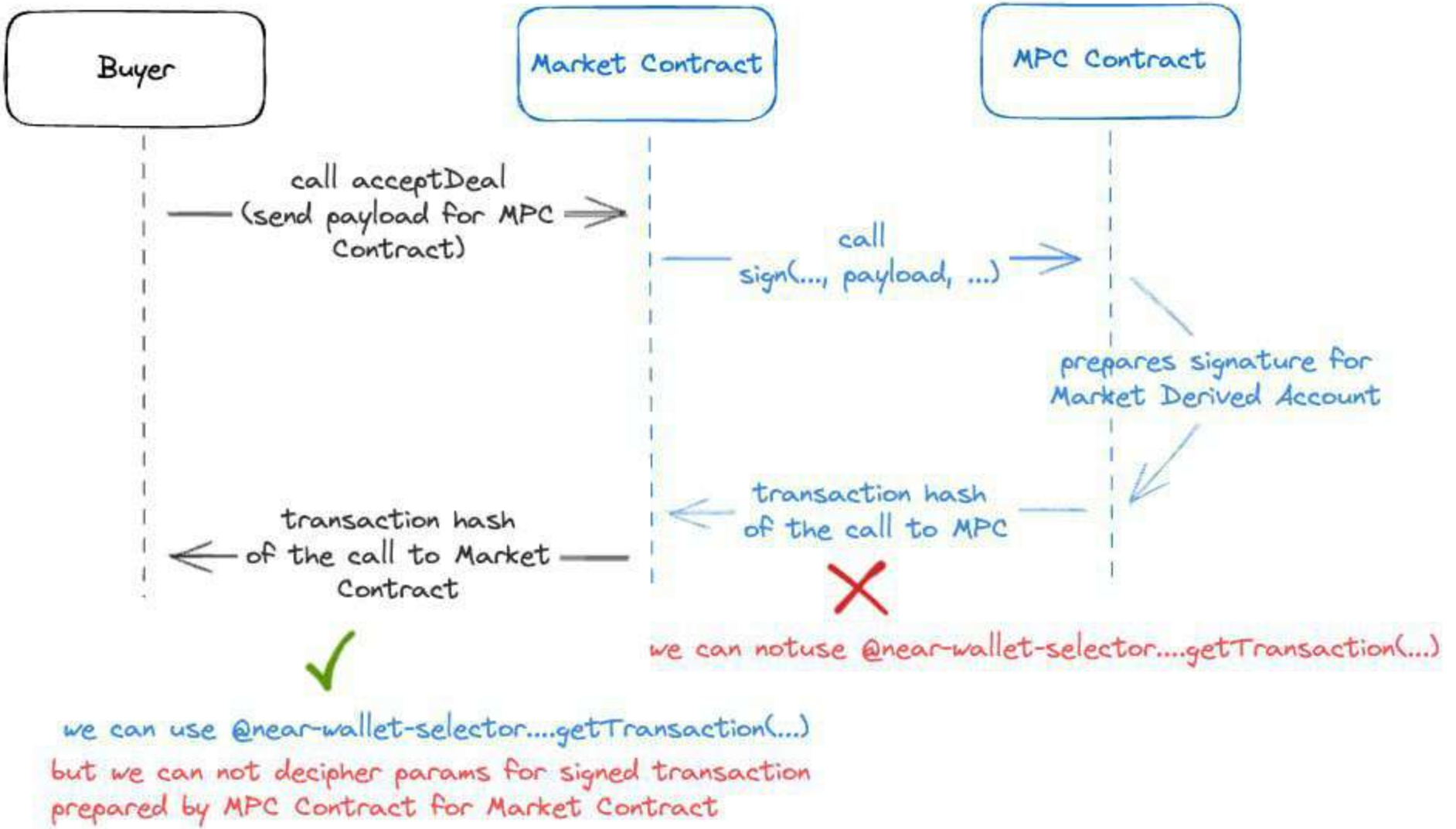
jamon-swap.why-nft.com

GitHub

Sources from ETHDam

Points To Improve

- [optimisation] Refactor Designfor of the Swap Page
- [optimisation] Optimise Contract and Scheme, tests
- [feature] Add Market Mechanics (public order list)
- [feature] Resolve Tokenomics for project
- Roadmap Alignment (discuss further steps after MVP)


Buyer
Market Contract
MPC Contract
call acceptDeal
(send payload for MPC
Contract)
call
sign(..., payload, ...)
prepares signature for
Market Derived Account
transaction hash
of the call to MPC
transaction hash
of the call to Market
Contract
we can notuse @near-wallet-selector....getTransaction(...)
we can use @near-wallet-selector....getTransaction(...)
but we can not decipher params for signed transaction
prepared by MPC Contract for Market Contract