ZetaChain

The Multichain Solution to Blockchain’s Gas Problem ?

The Problem?

Computation on some chains (ex. ETH Mainnet) can be **very expensive**

Other chains have cheaper gas prices...

But Valuable Assets are stored in the Main Chain…

"

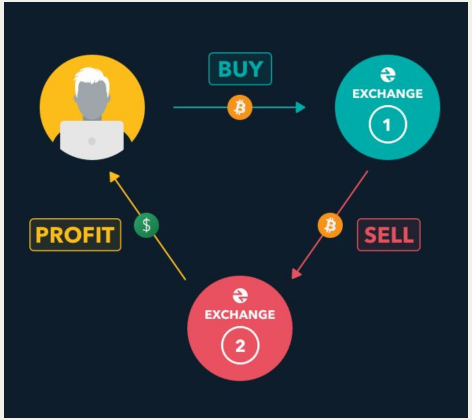
Computational Arbitrage.

Daniyal after CSCD71 Tutorial

Arbitrage

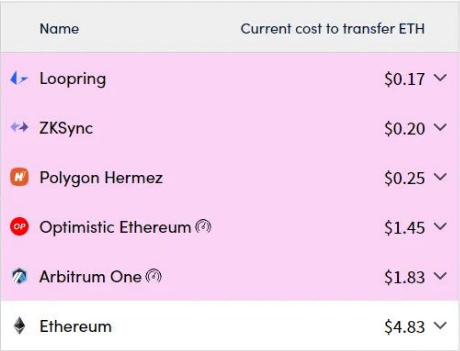
/ˈärbəˌträZH/

purchase and sale of the same asset in different markets to profit from differences in listing prices



Why not apply it to Compute ?

Compute where prices are low, use result where prices are high



Name	Current cost to transfer ETH
Loopring	$0.17
ZKSync	$0.20
Polygon Hermez	$0.25
Optimistic Ethereum	$1.45
Arbitrum One	$1.83
Ethereum	$4.83

"

Universal Blockchain

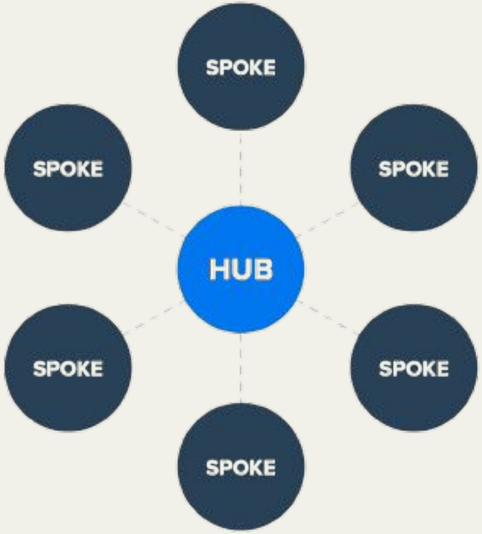
ZetaChain Docs

What is **ZetaChain?**

- Layer 1 Blockchain
- EVM-compatible (Smart Contracts)
- Cosmos BFT Consensus (Proof of Stake)
- Purpose? Provide **cross-chain** smart contract interactions

Hub and Spoke Architecture

ZetaChain (Hub) **connects** external EVM chains (Spokes)



Core Validators

Participate in the core Proof-of-Stake consensus on ZetaChain, using the Comet BFT consensus protocol.

Observer-Signer Validators

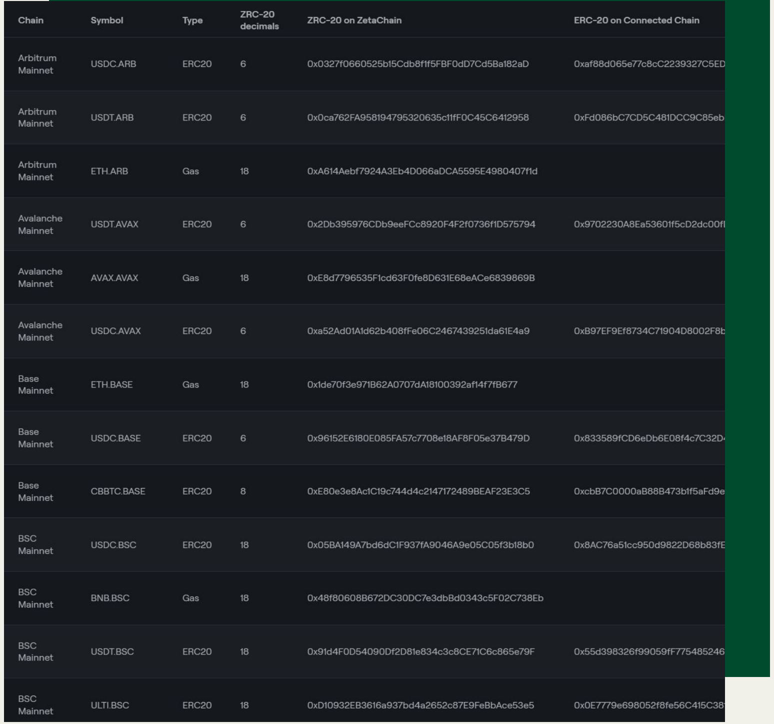
Monitor and vote *between* ZetaChain and external chains,

for specific **events** signifying steps of cross-chain transactions (CCTx)

ZRC-20 Tokens

ZetaChain’s token standard represents tokens from external blockchains

- A **Stablecoin** with value pegged to the respective gas token value



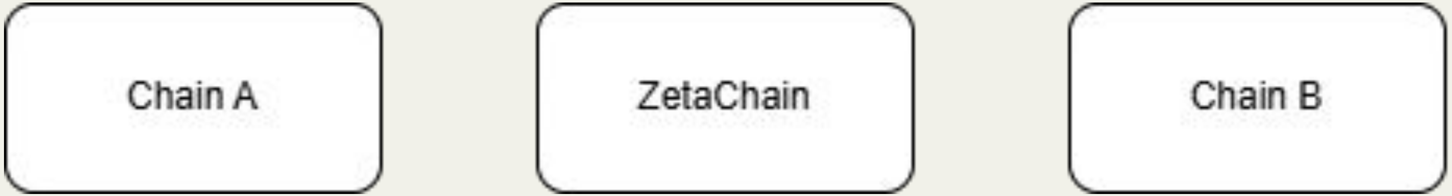
Chain	Symbol	Type	ZRC-20 decimals	ZRC-20 on ZetaChain	ERC-20 on Connected Chain
Arbitrum Mainnet	USDC.ARB	ERC20	6	0x0327f0660525b15Cdb8f1f5FBF0dD7Cd5Ba182aD	0xaf88d065e77c8cC2239327C5ED
Arbitrum Mainnet	USDT.ARB	ERC20	6	0x0ca762FA958194795320635c11fF0C45C6412958	0xFd086bC7CD5C481DCC9C85eb
Arbitrum Mainnet	ETH.ARB	Gas	18	0xA614Aebf7924A3Eb4D066aDCA5595E4980407f1d	
Avalanche Mainnet	USDT.AVAX	ERC20	6	0x2Db395976CDb9eeFCc8920F4F2f0736f1D575794	0x9702230A8Ea53601f5cD2dc00f
Avalanche Mainnet	AVAX.AVAX	Gas	18	0xE8d7796535F1cd63F0fe8D631E68eACe6839869B	
Avalanche Mainnet	USDC.AVAX	ERC20	6	0xa52Ad01A1d62b408fFe06C2467439251da61E4a9	0xB97EF9Ef8734C71904D8002F8b
Base Mainnet	ETH.BASE	Gas	18	0x1de70f3e971B62A0707dA18100392af14f7fB677	
Base Mainnet	USDC.BASE	ERC20	6	0x96152E6180E085FA57c7708e18AF8F05e37B479D	0x833589fCD6eDb6E08f4c7C32D
Base Mainnet	CBBTC.BASE	ERC20	8	0xE80e3e8Ac1C19c744d4c2147172489BEAF23E3C5	0xcbB7C0000aB88B473b1f5aFd9e
BSC Mainnet	USDC.BSC	ERC20	18	0x05BA149A7bd6dC1F937fA9046A9e05C05f3b18b0	0x8AC76a51cc950d9822D68b83fE
BSC Mainnet	BNB.BSC	Gas	18	0x48f80608B672DC30DC7e3dbBd0343c5F02C738Eb	
BSC Mainnet	USDT.BSC	ERC20	18	0x91d4F0D54090Df2D81e834c3c8CE71C6c865e79F	0x55d398326f99059fF775485246
BSC Mainnet	ULTI.BSC	ERC20	18	0xD10932EB3616a937bd4a2652c87E9FeBbAce53e5	0x0E7779e698052f8fe56C415C38

Cross-Chain Transaction (CCTx) Workflow

3 Things are Certain in Life: Death, Taxes and Undocumented Code

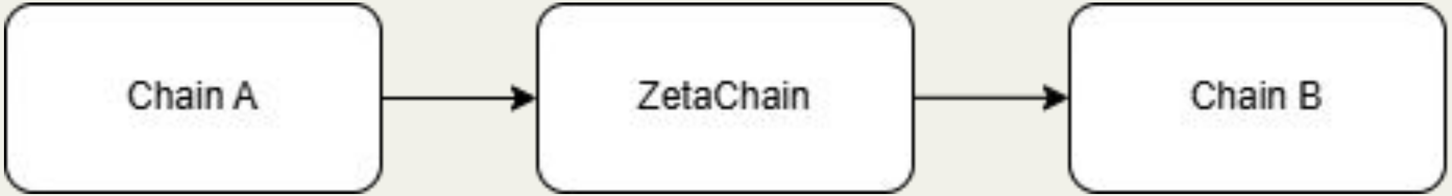
Every Story needs Characters

(it’s not Alice, Bob and Mallory)



Every Story needs Characters

(it’s not Alice, Bob and Mallory)

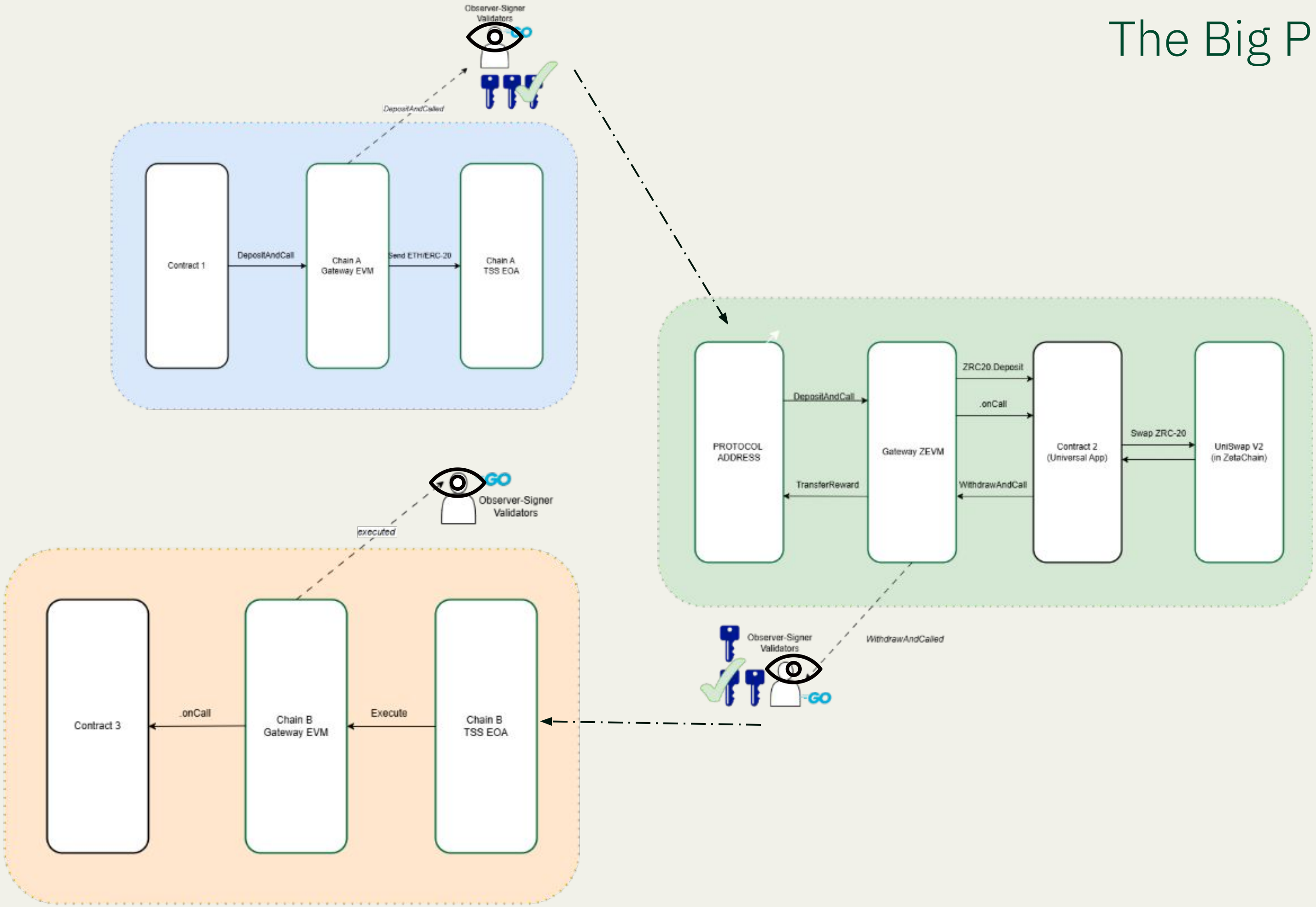


Chain A

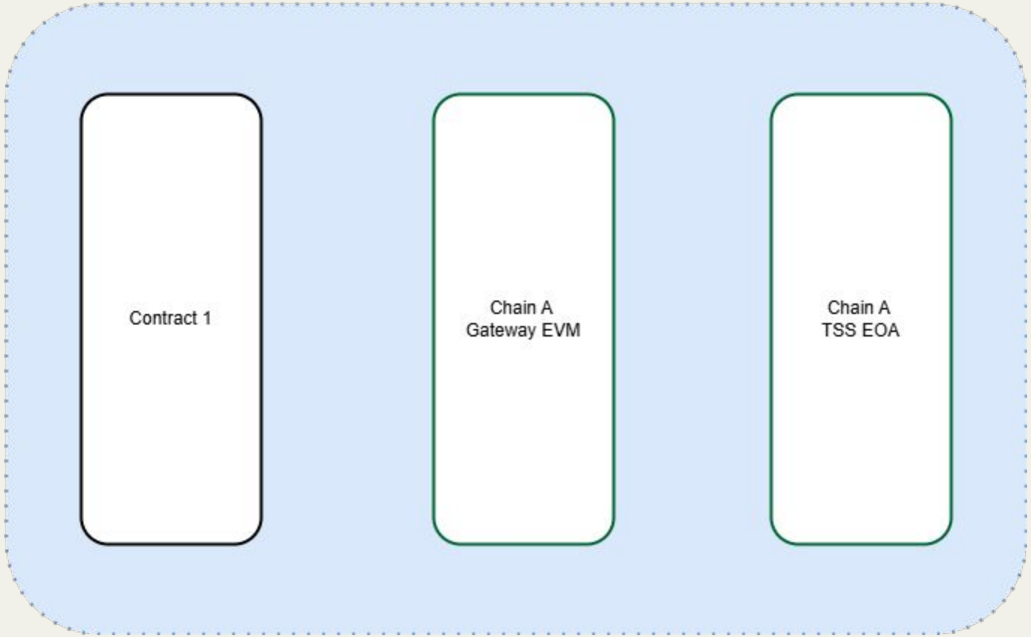
ZetaChain

Chain B

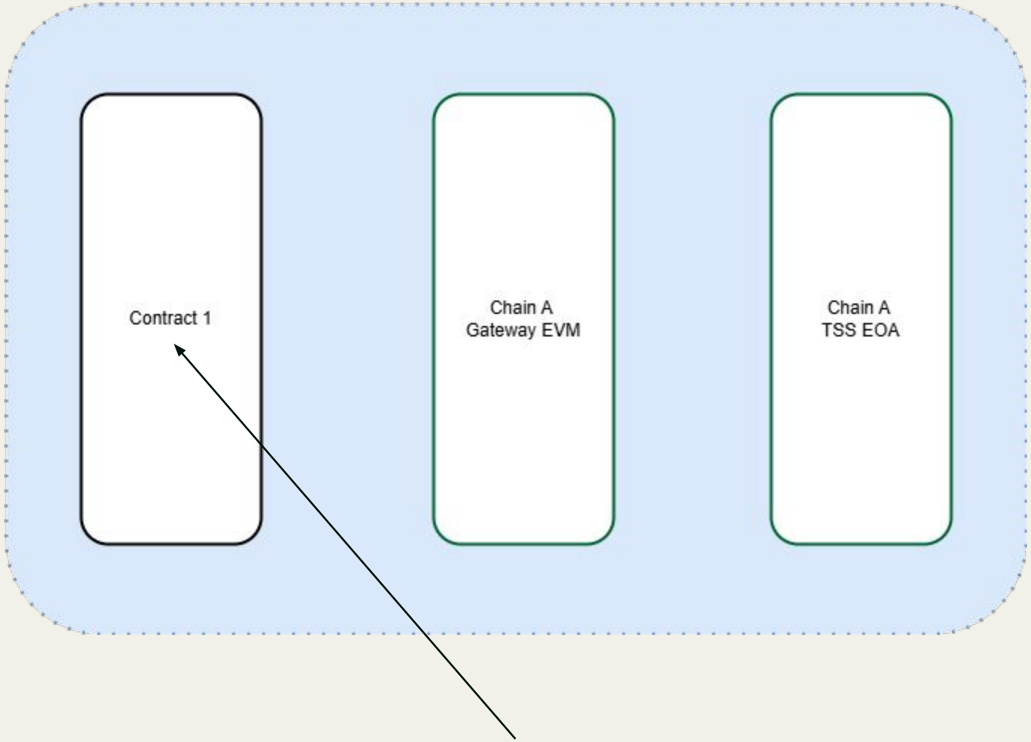
The Big Picture



Inside Chain A

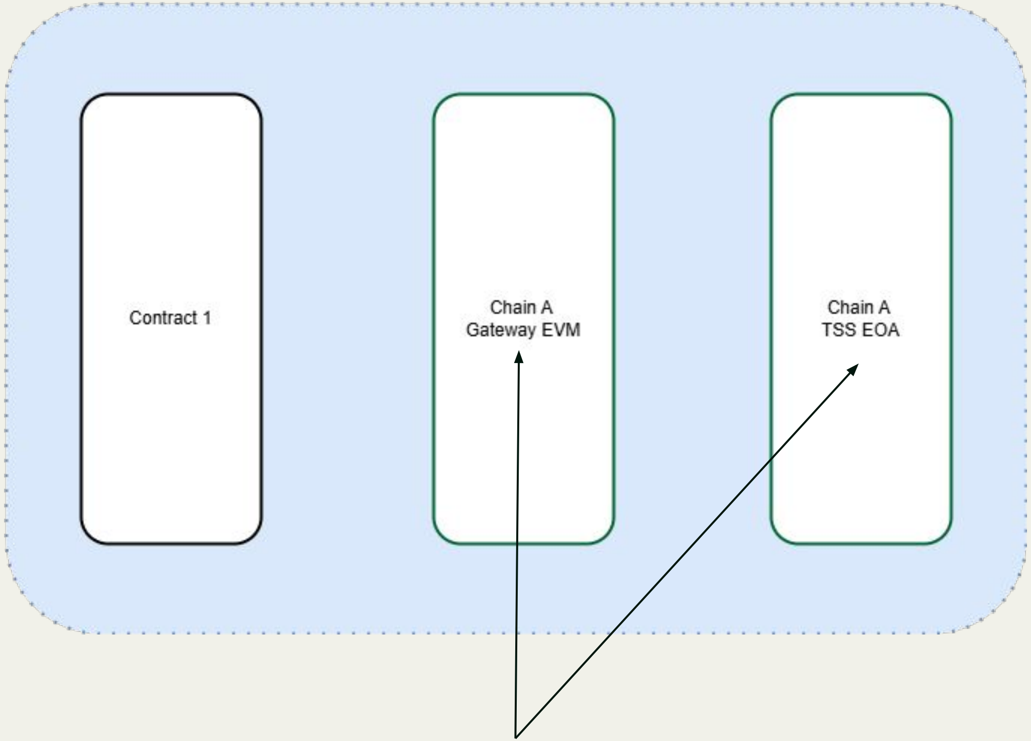


Inside Chain A



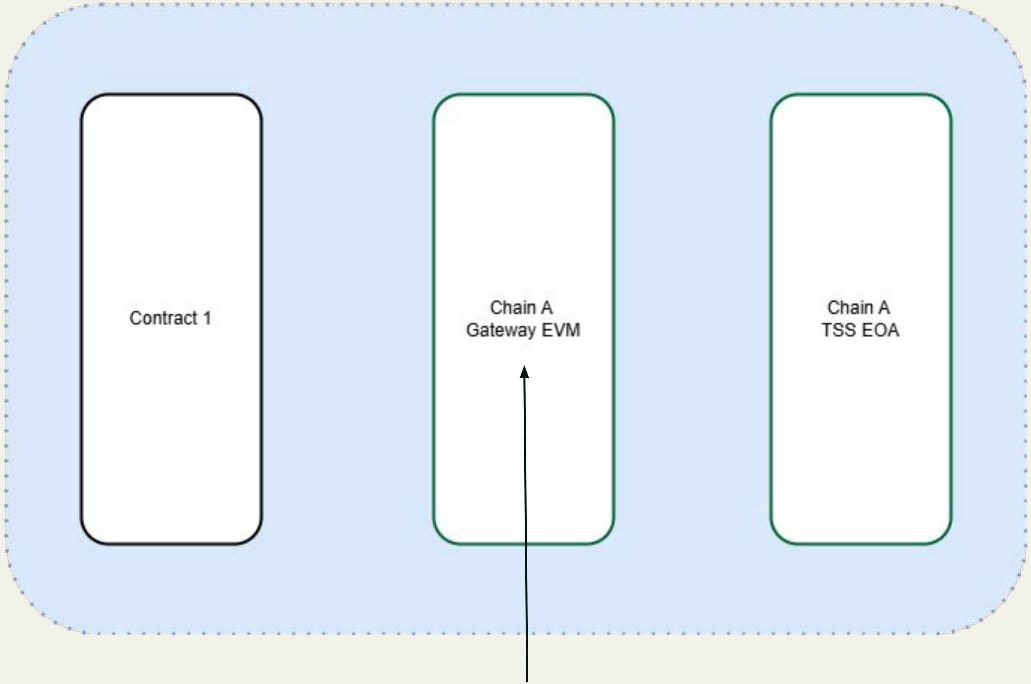
Contract we deployed on Chain A

Inside Chain A



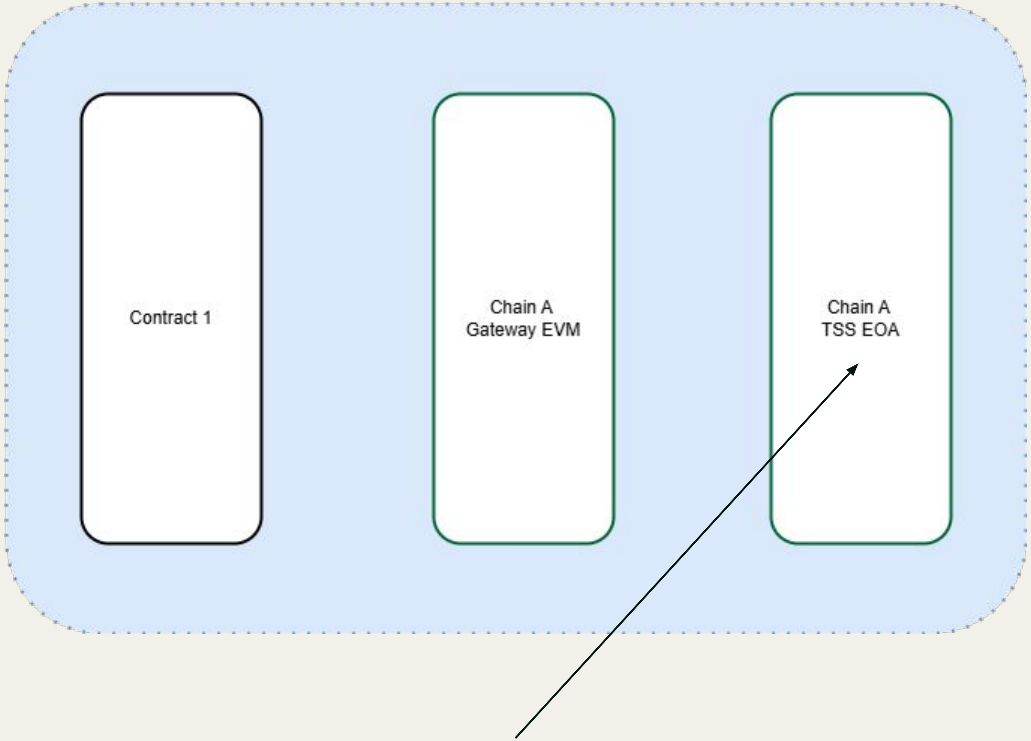
Deployed by “Zeta” on Chain A

Inside Chain A



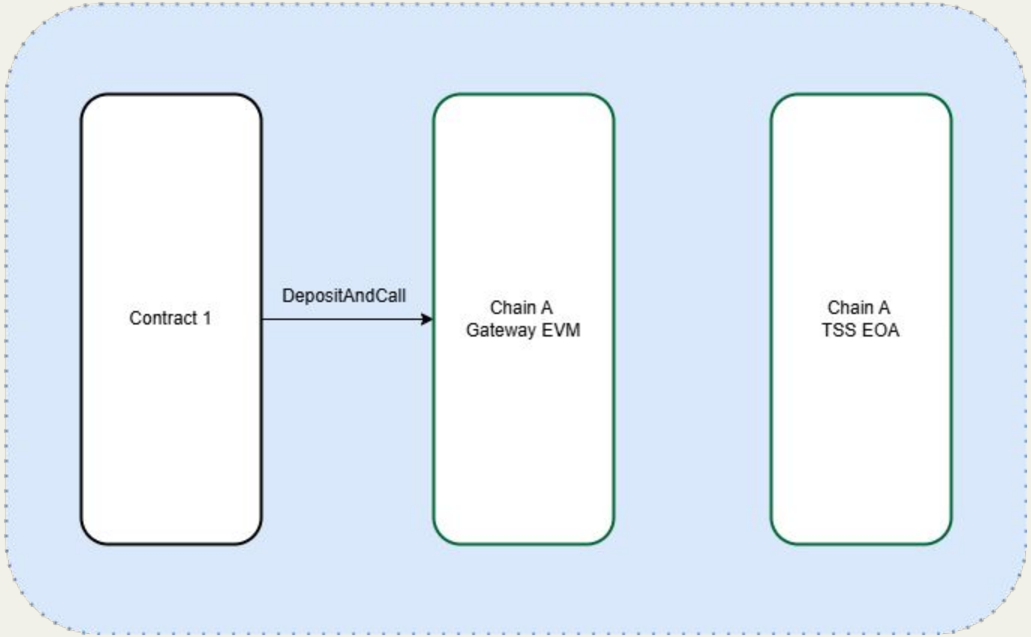
Found in all supported External Chains to handle transactions into and out of ZetaChain

Inside Chain A



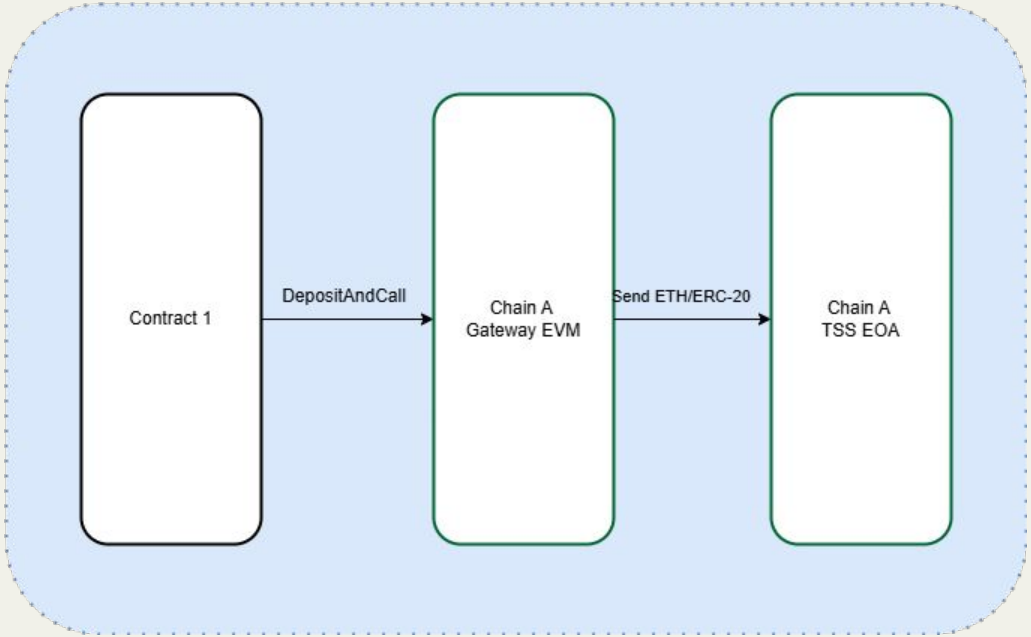
Bank for the External Chain’s native token

Inside Chain A



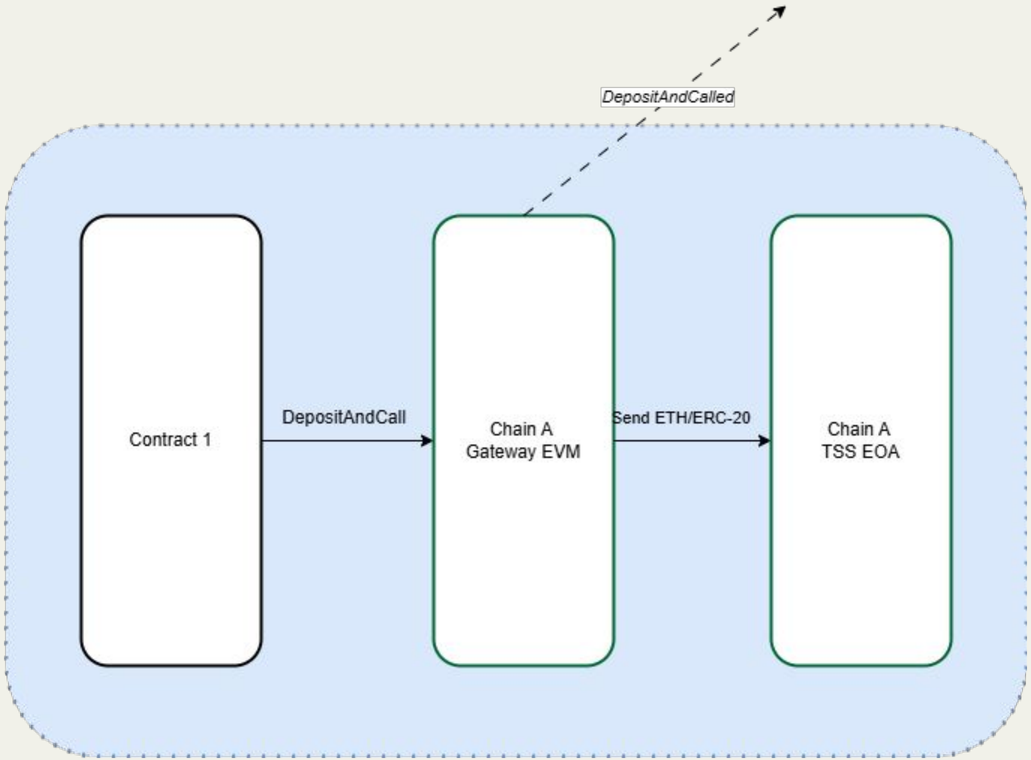
Invokes Chain A Gateway EVM’s *DepositAndCall* with transaction data

Inside Chain A



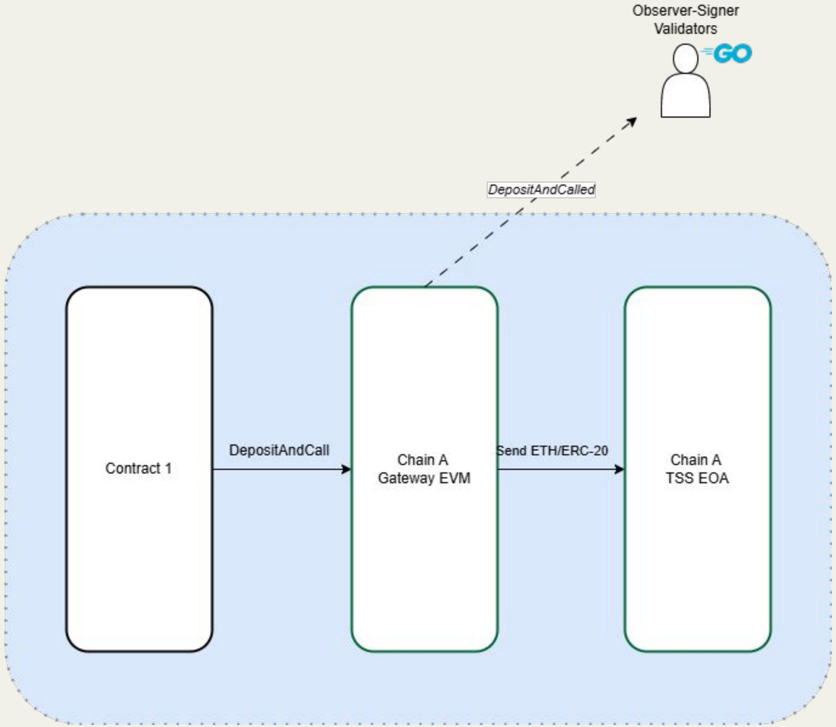
Send “*Gas for Remaining CCTx*” amount to Chain A TSS EOA

Inside Chain A



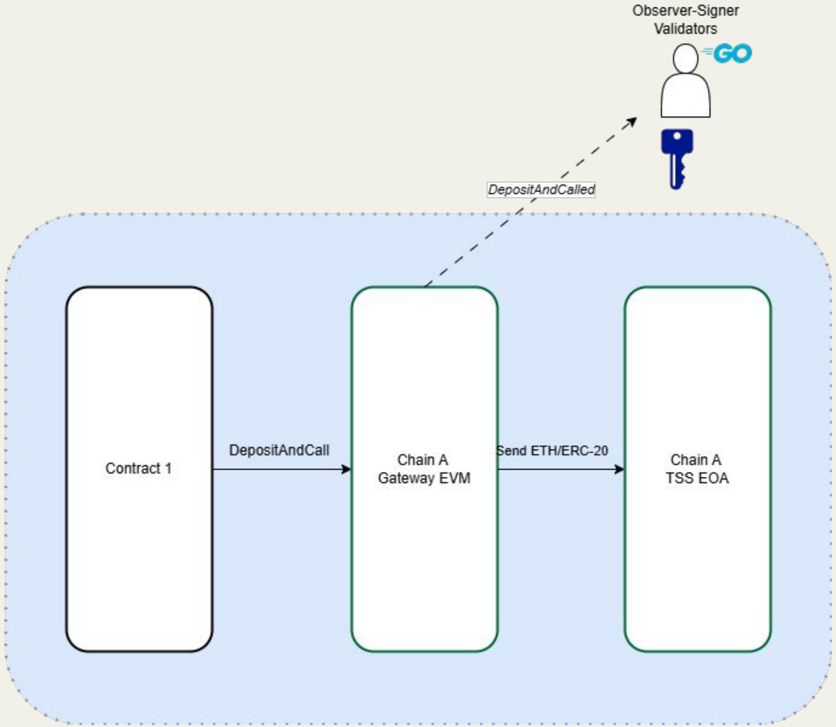
Emits *DepositAndCalled* **Event**
with transaction data

Validating Events from External Chain into ZetaChain



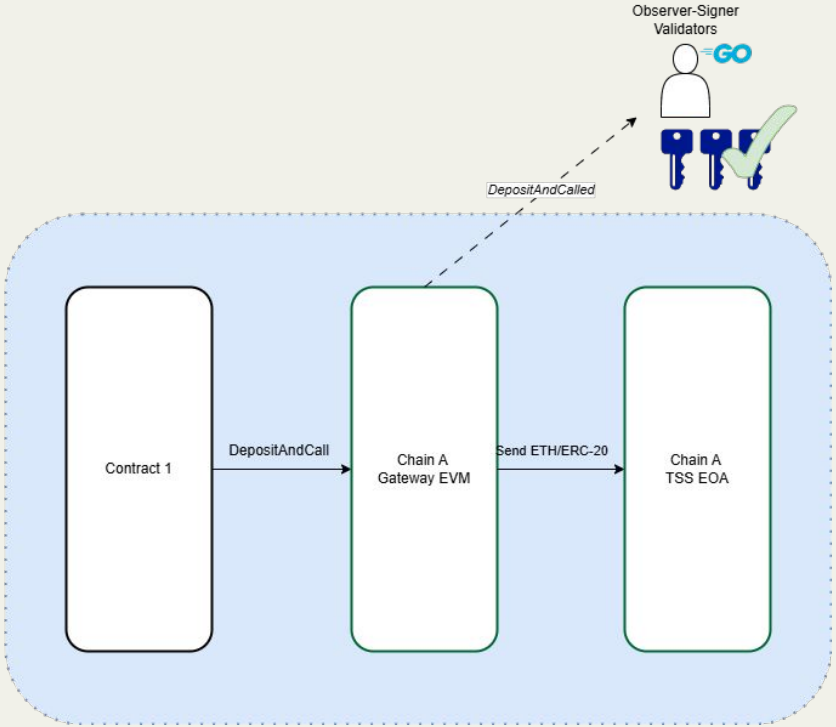
Observer-Signer validators monitor **blocks** in Chain A for any new events emitted

Validating Events from External Chain into ZetaChain



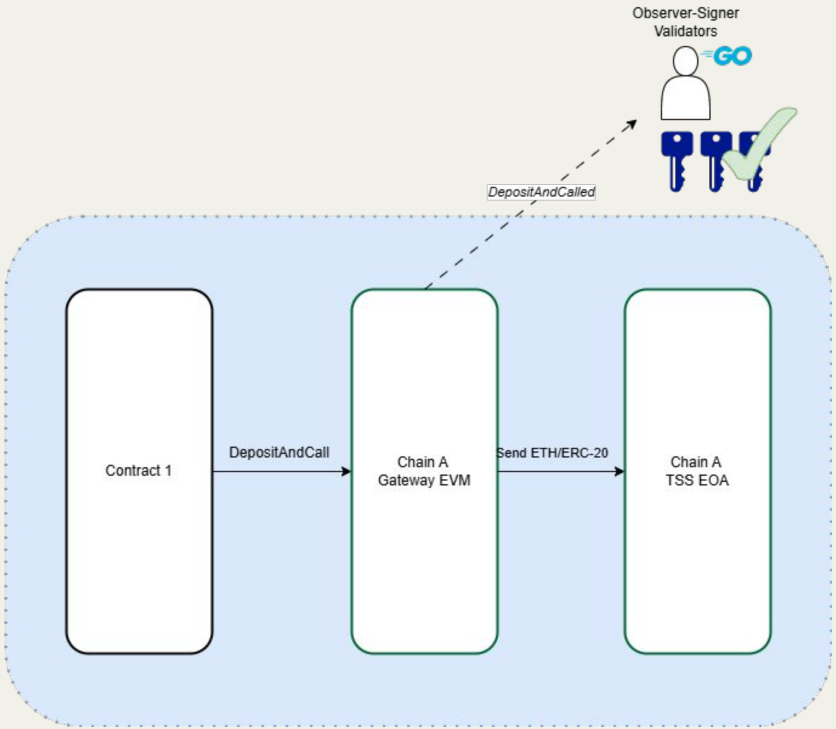
Vote on the validity of the observed pending "*inbound message*" by leveraging TSS

Validating Events from External Chain into ZetaChain



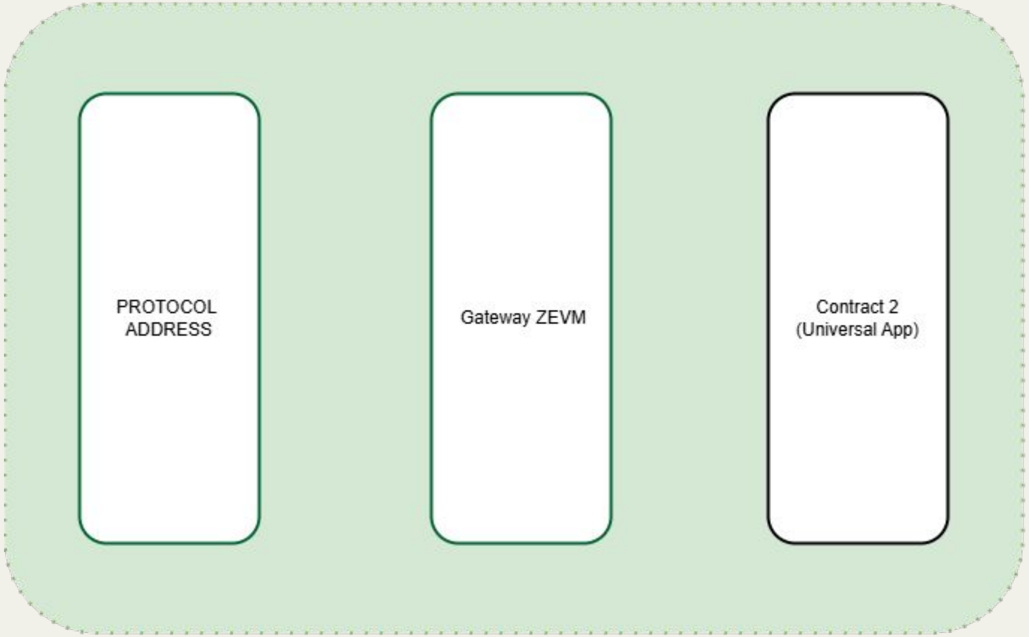
Once voting passes, **Protocol Address** initiates corresponding transaction in ZetaChain

Validating Events from External Chain into ZetaChain

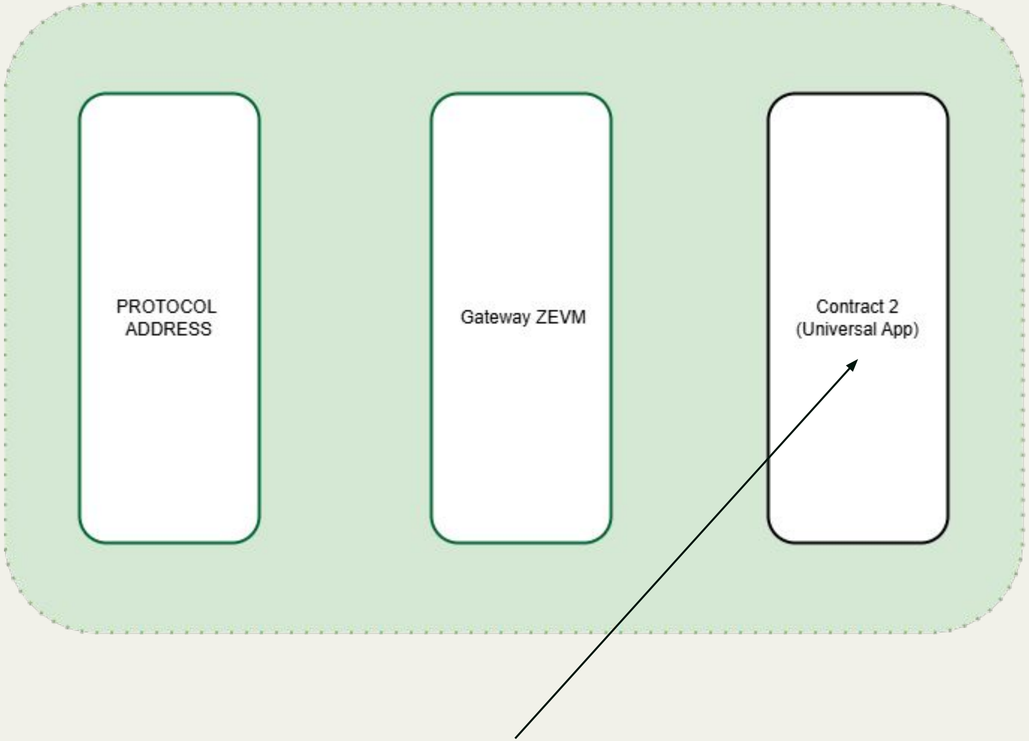


Important! Observer-Signer monitoring and voting all happens **off-chain**

Inside ZetaChain

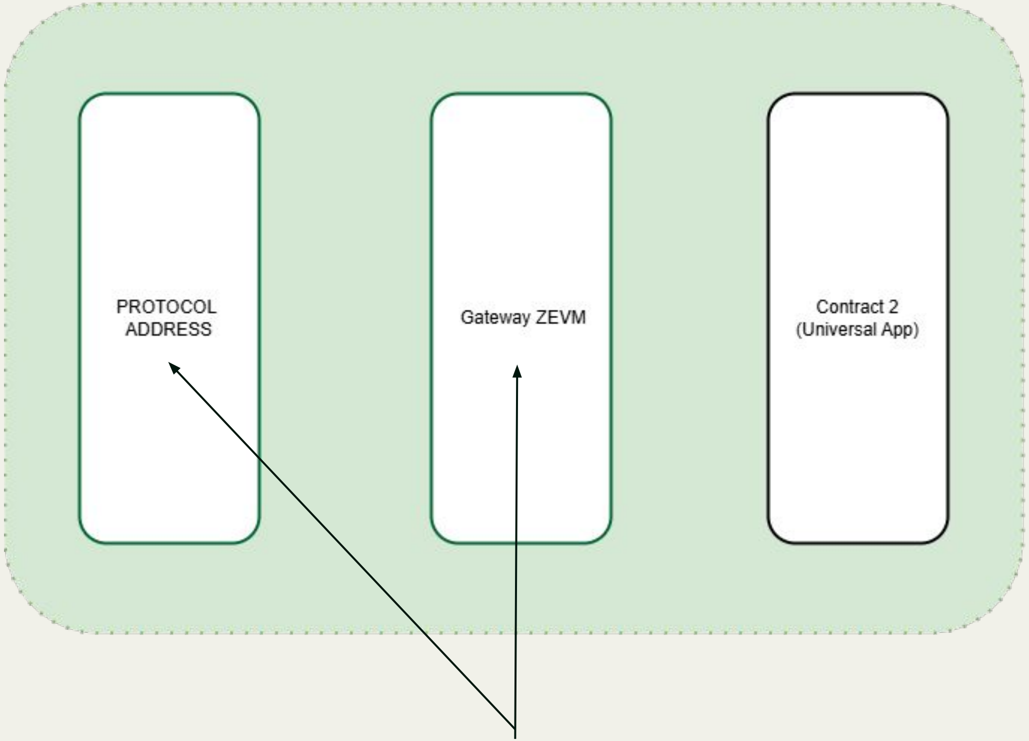


Inside ZetaChain



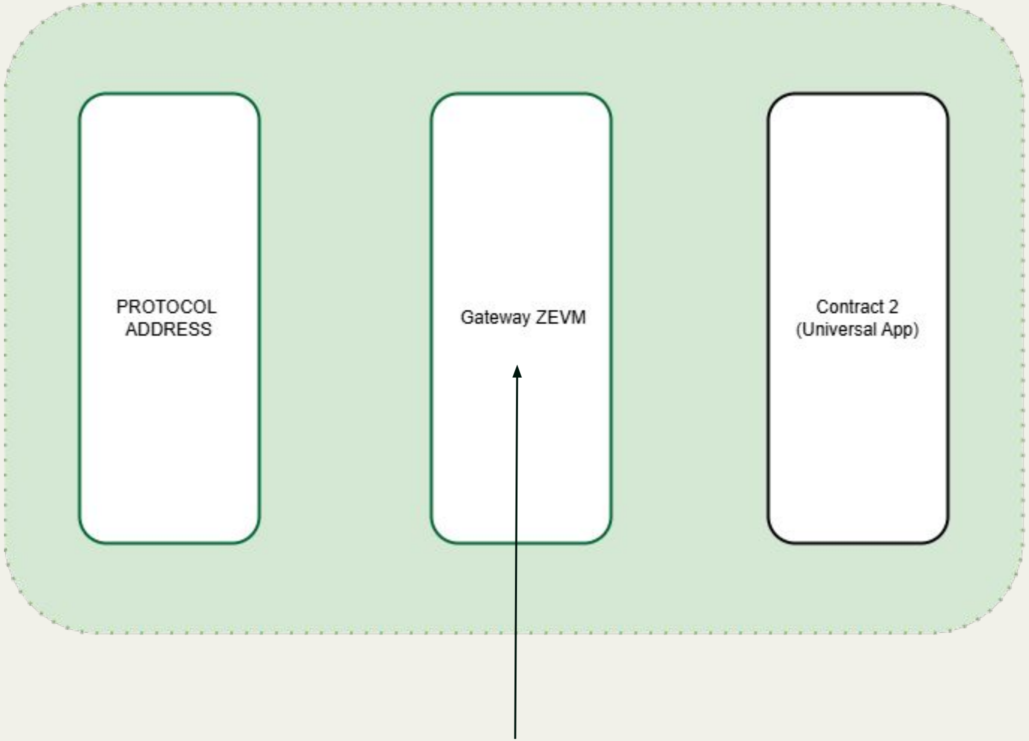
Contract that we deploy on ZetaChain

Inside ZetaChain



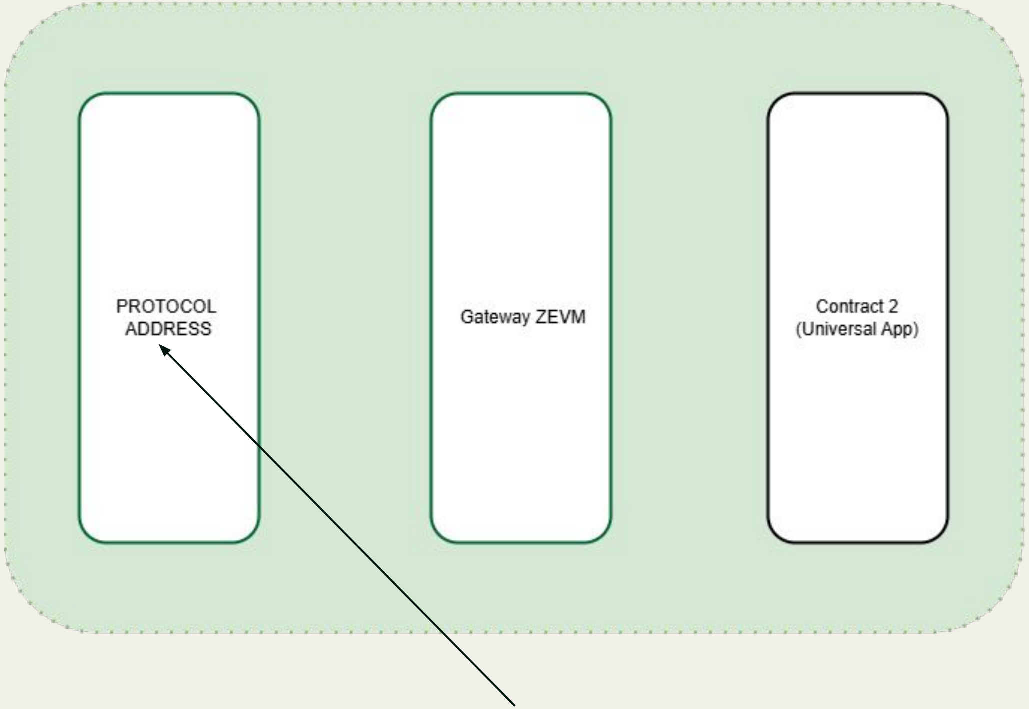
Deployed Contract/EOA on ZetaChain

Inside ZetaChain



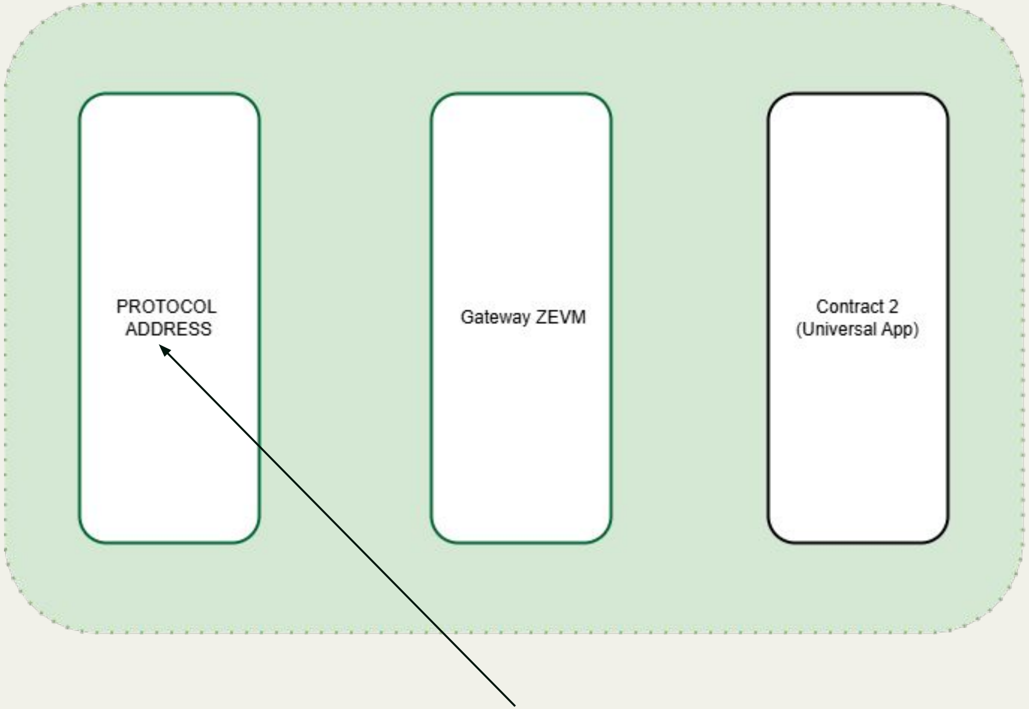
Handle transactions into and out of ZetaChain

Inside ZetaChain



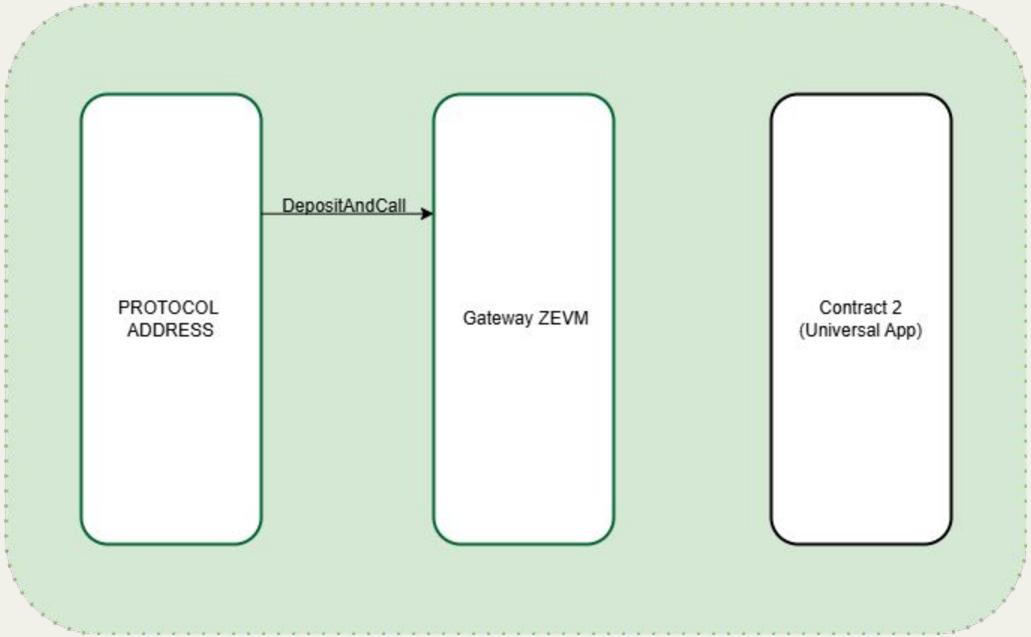
The only wallet with authority to pass incoming transaction calls to Gateway ZEVM

Inside ZetaChain



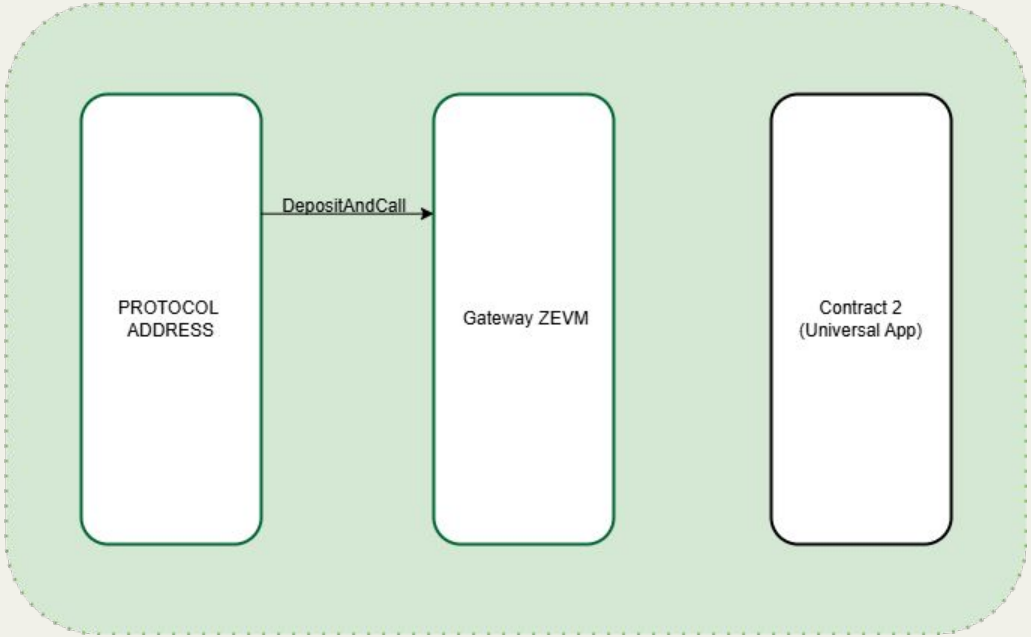
Think of it as an EOA that only the blockchain can initiate transactions from it

Inside ZetaChain



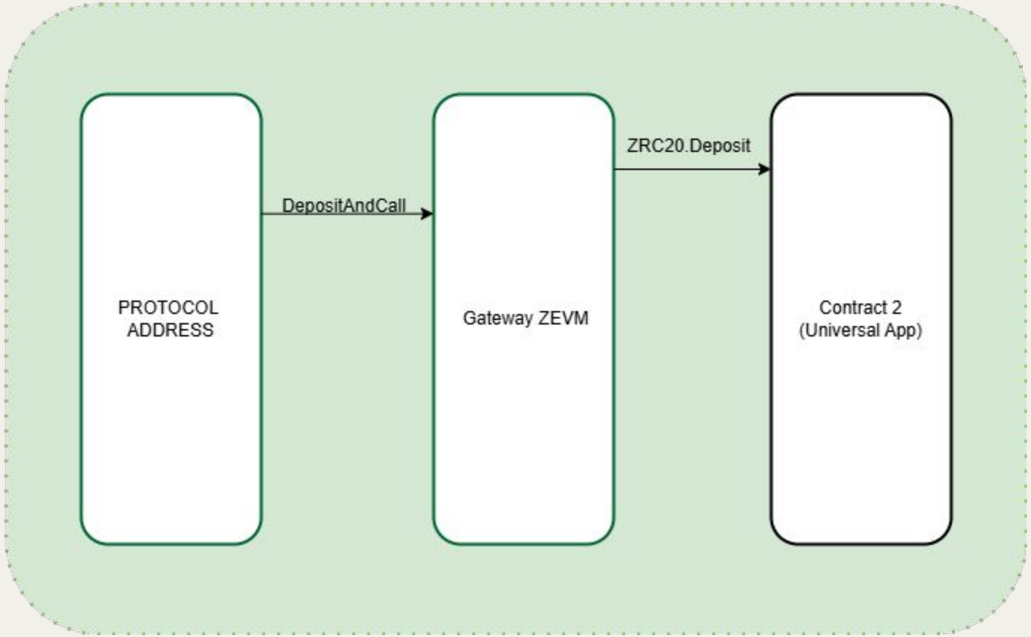
Invokes Gateway ZEVM’s *DepositAndCall*
with transaction data

Inside ZetaChain



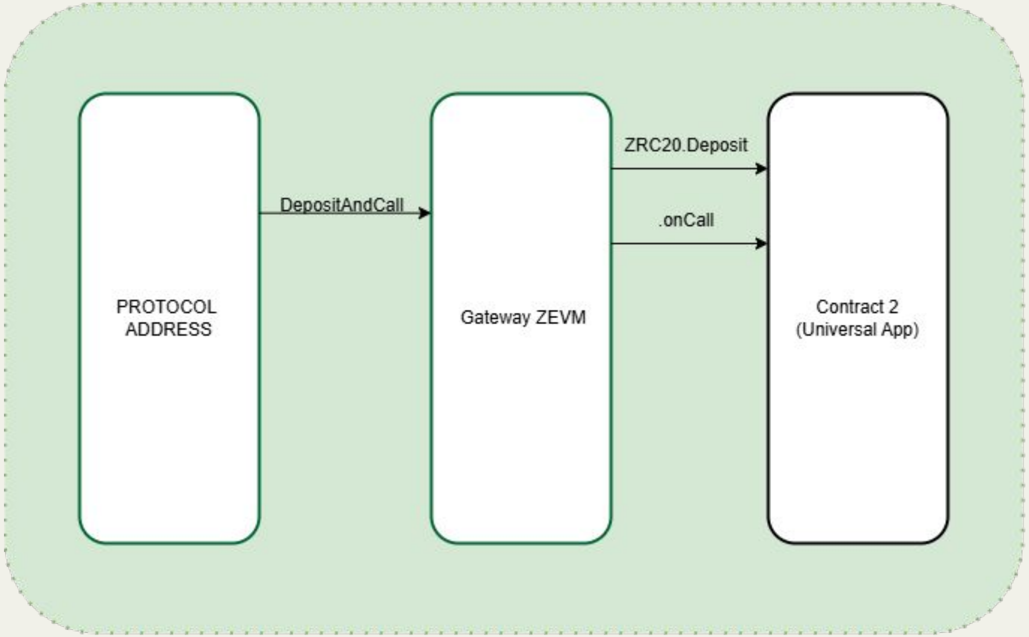
DepositAndCall mints “*Gas for Remaining CCTx*” amount of ZRC-20 representative of ETH/ERC-20 sent to TSS EOA

Inside ZetaChain



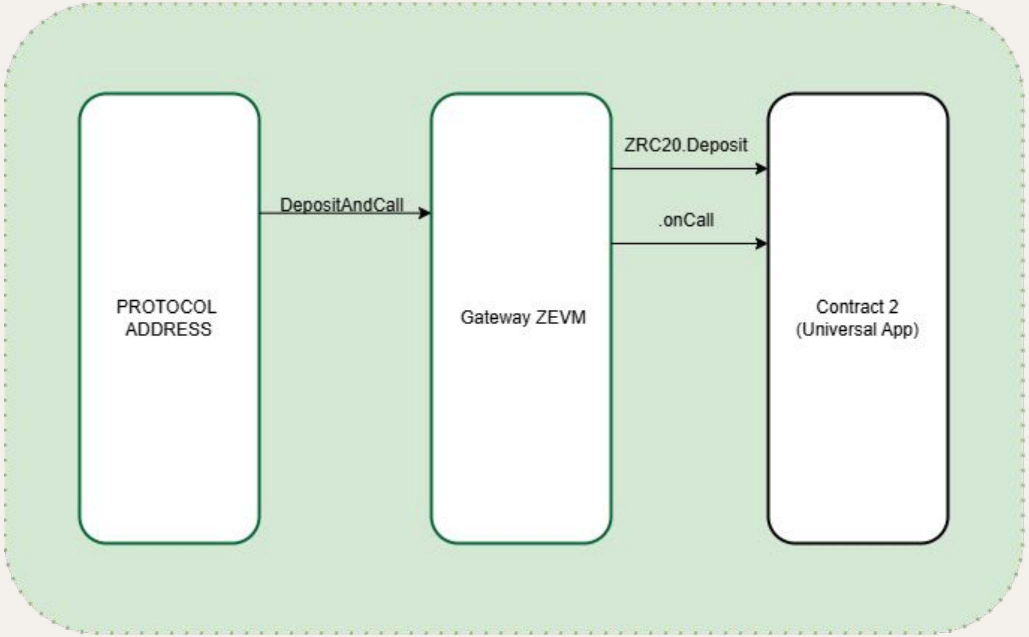
Minted ZRC-20 token is then transferred to Contract 2

Inside ZetaChain



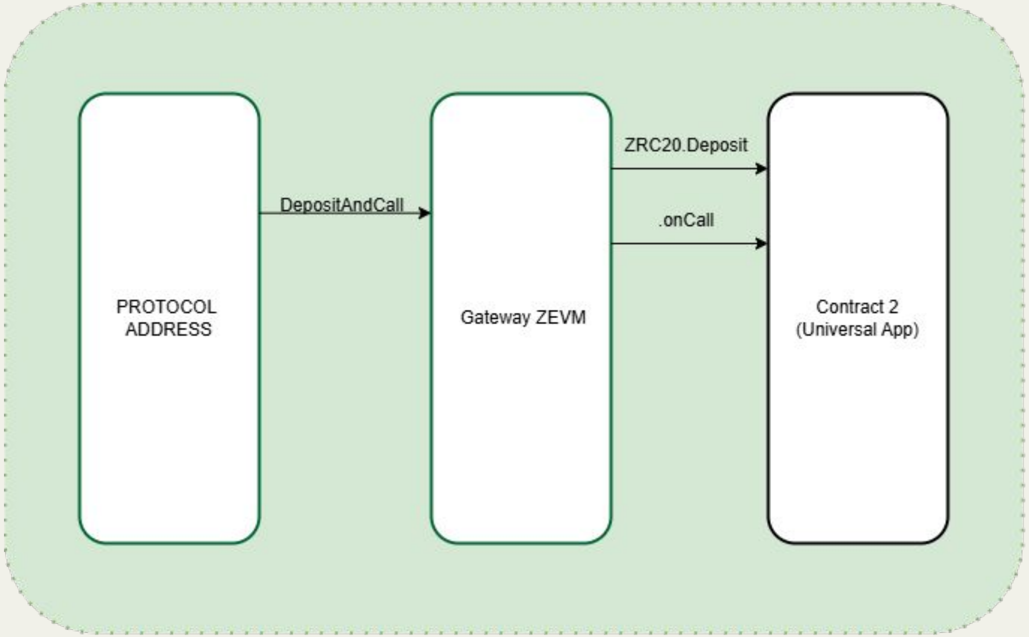
Lastly, *DepositAndCall* invokes Contract 2’s *.onCall* with transaction data passed

Inside ZetaChain



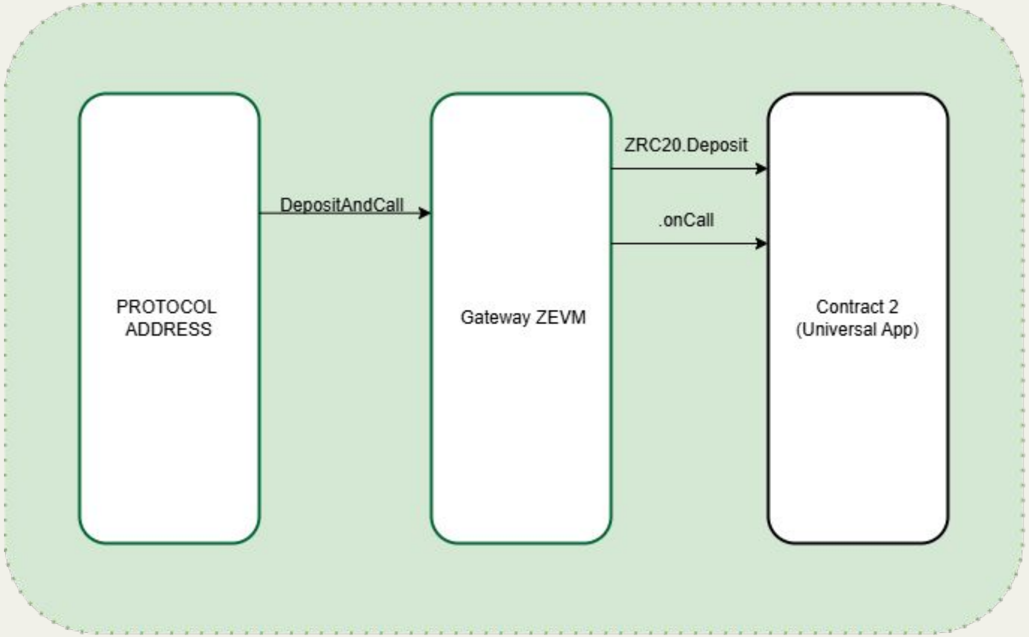
All Universal Apps on ZetaChain need to implement a *.onCall* method

Inside ZetaChain



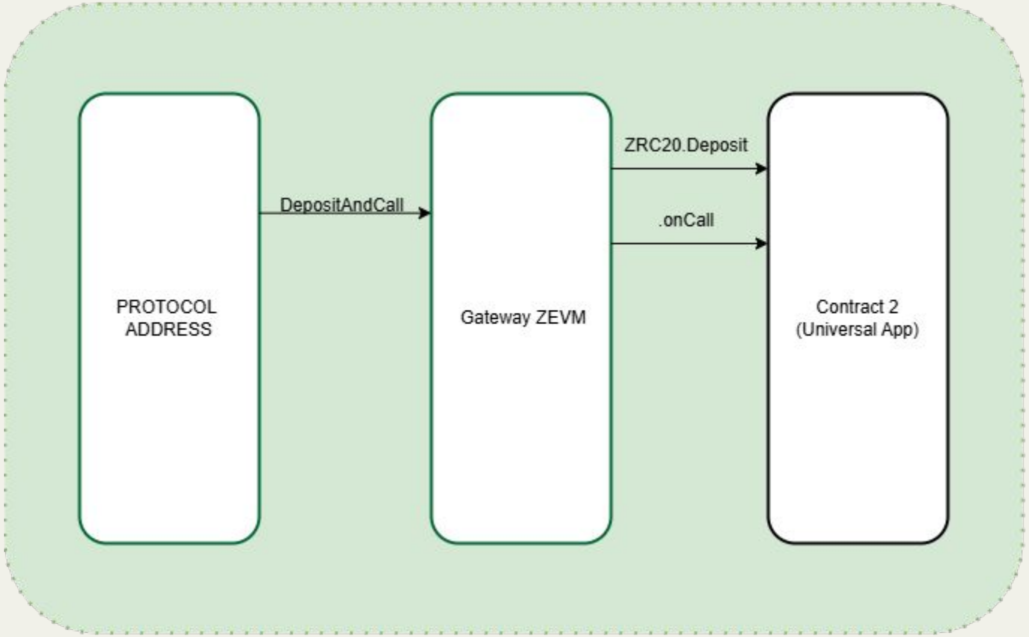
Entry point for any incoming transaction from an external chain to ZetaChain

Inside ZetaChain



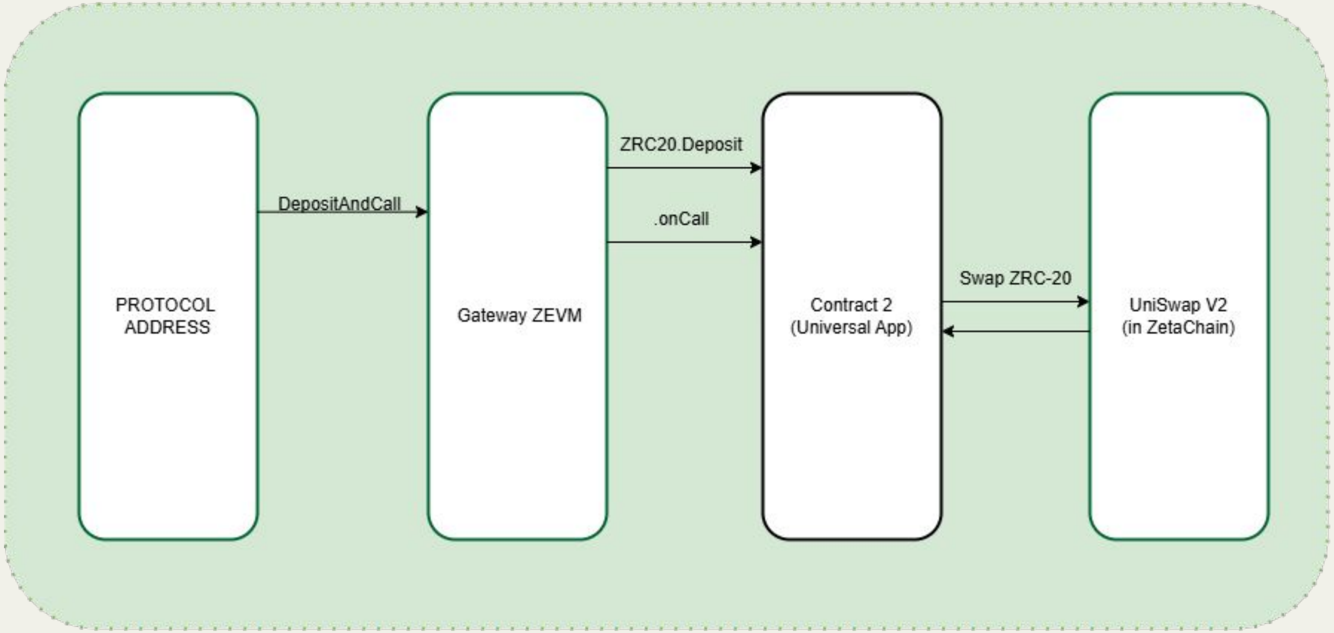
Before passing transaction to Chain B,
Contract 2 (Universal App) **is responsible for:**

Inside ZetaChain



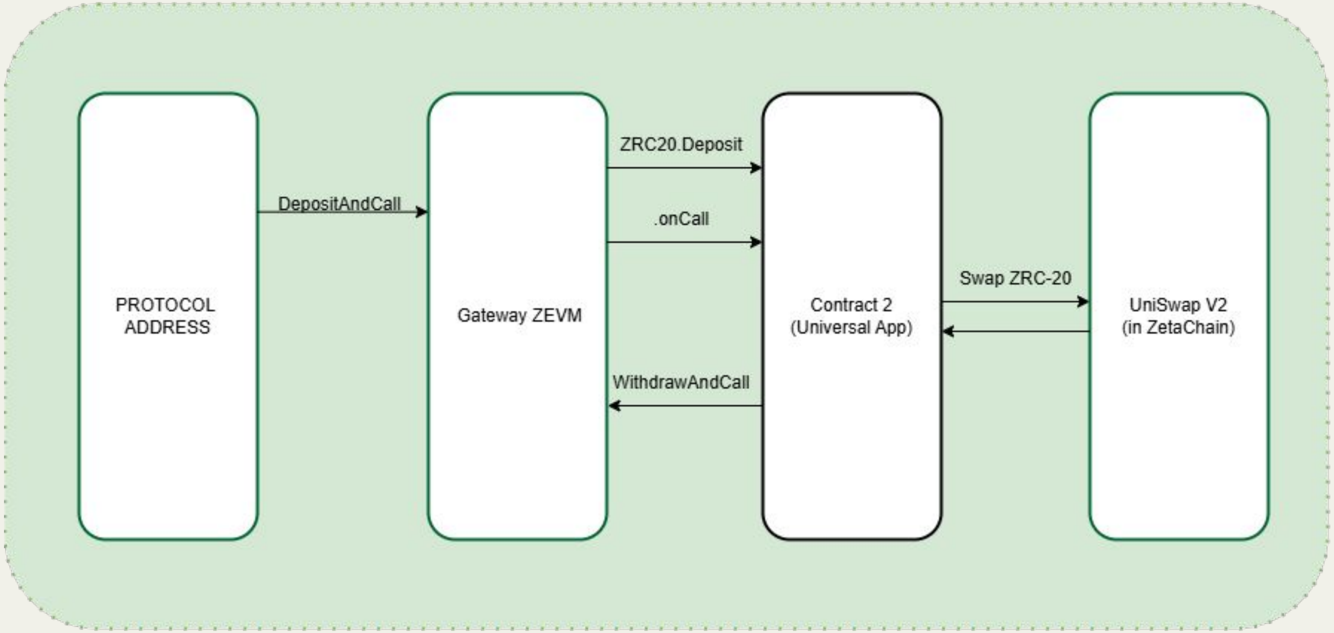
1. Estimating Gas Breakdown for Outgoing Transaction in the Deposited *ZRC-20* units

Inside ZetaChain



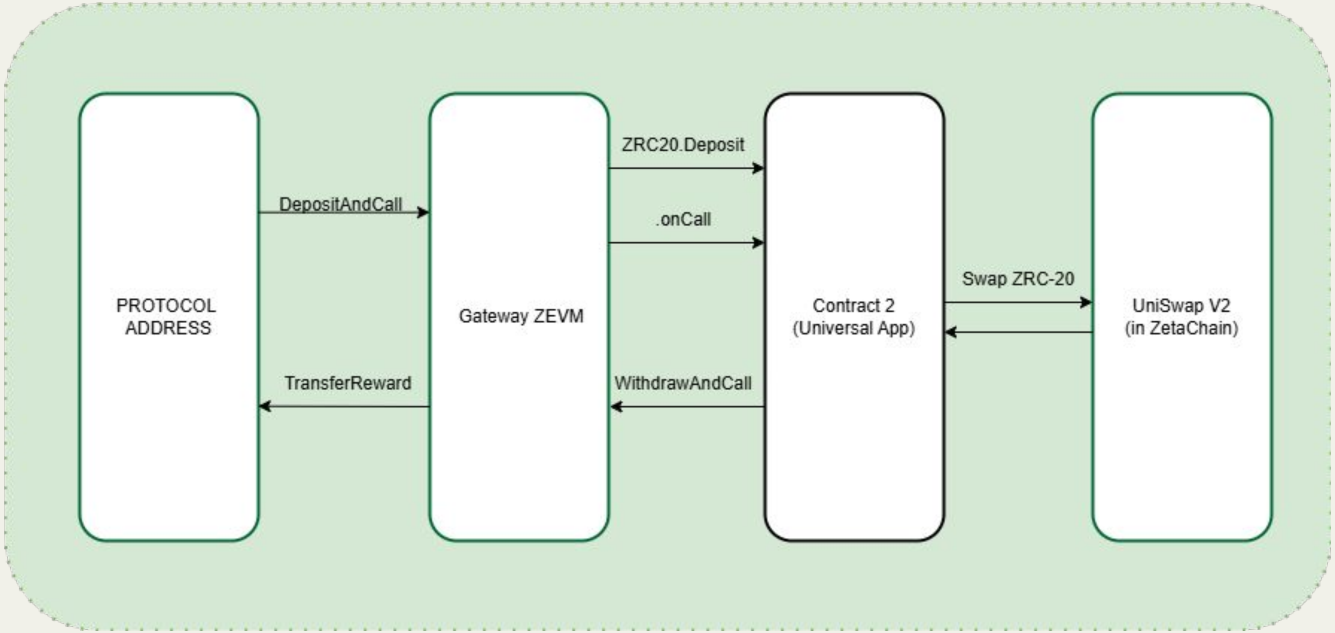
2. Converting Estimated Gas to Chain B’s Native Token Unit by doing a swap with **Uniswap V2** (found in ZetaChain)

Inside ZetaChain



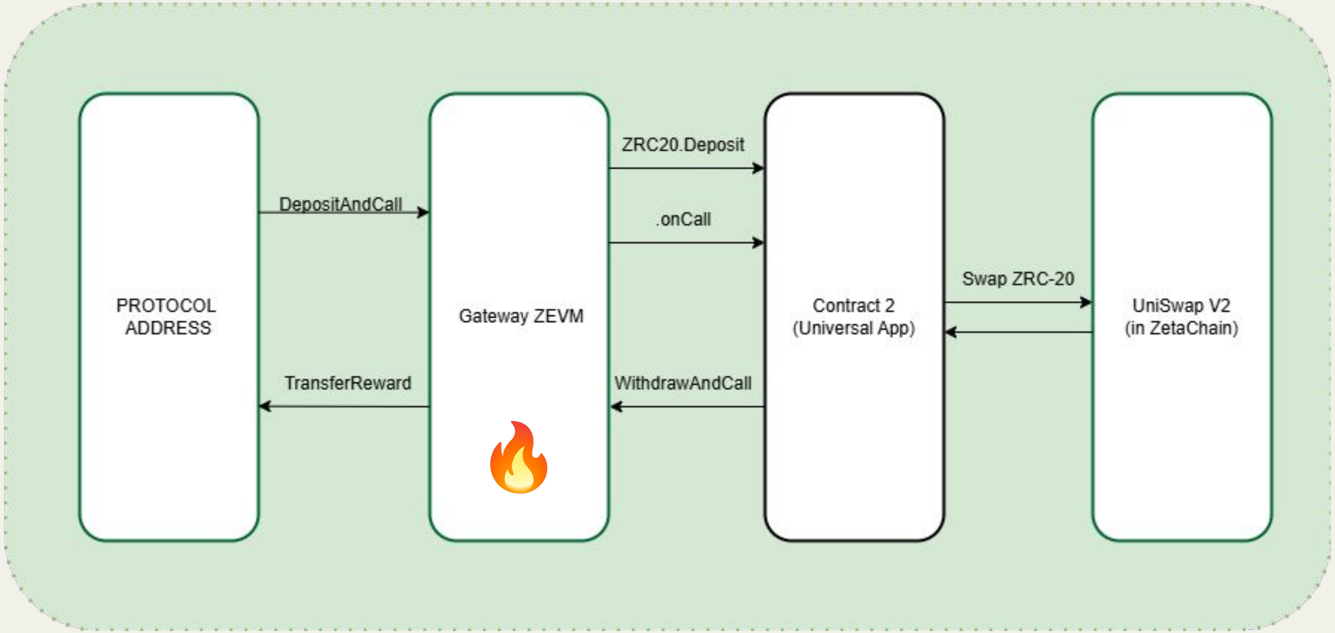
Contract 2 invokes Gateway ZEVM’s *WithdrawAndCall* with transaction data

Inside ZetaChain



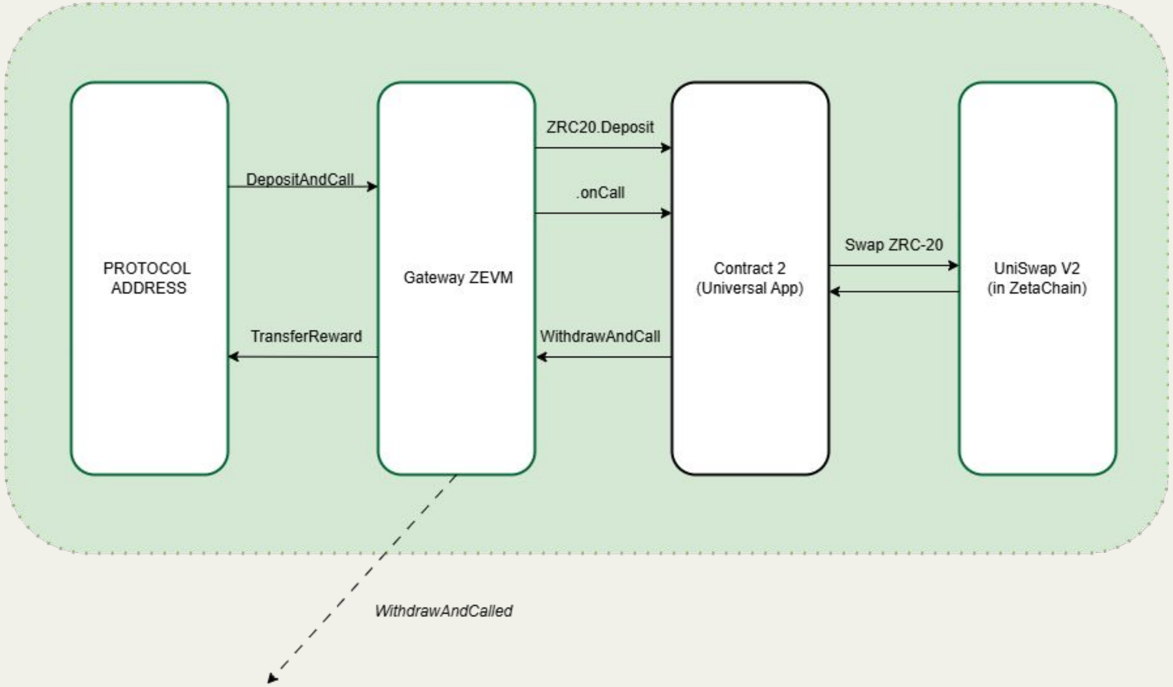
Send ZRC-20 amount that will be rewarded to facilitating Validators to the Protocol Address

Inside ZetaChain



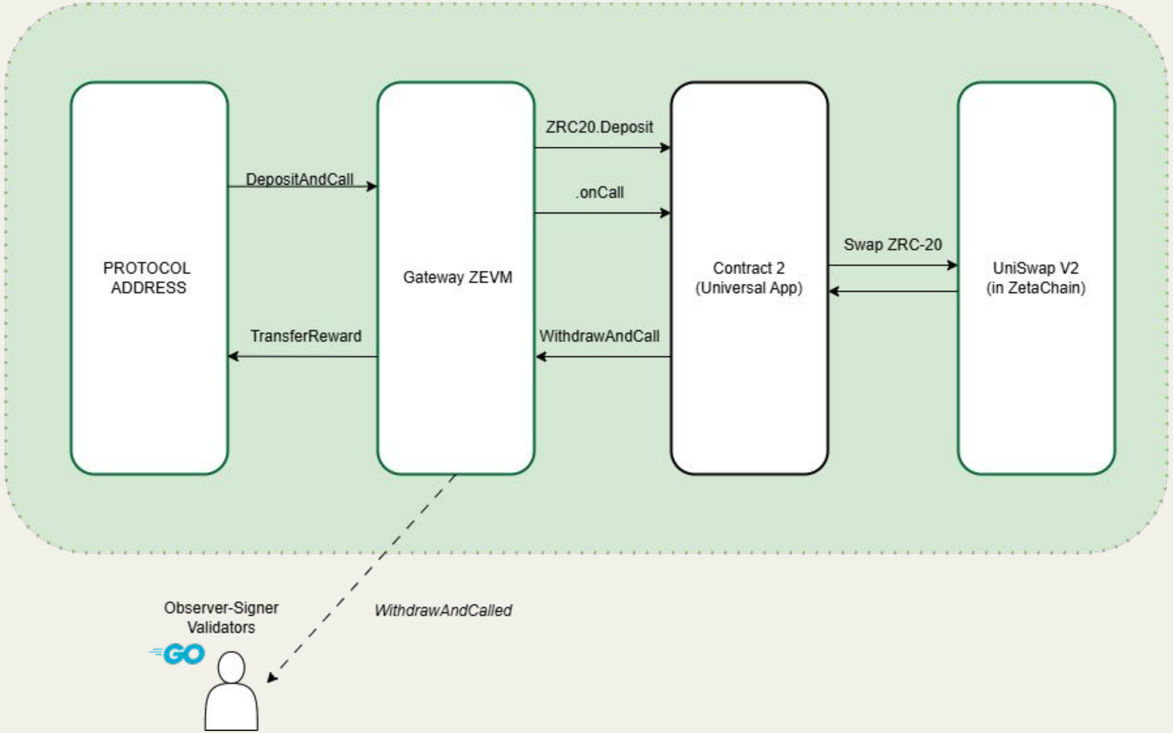
Burn ZRC-20 amount that will be withdrawn from Chain B's TSS EOA to cover for CCTx execution on Chain B

Inside ZetaChain



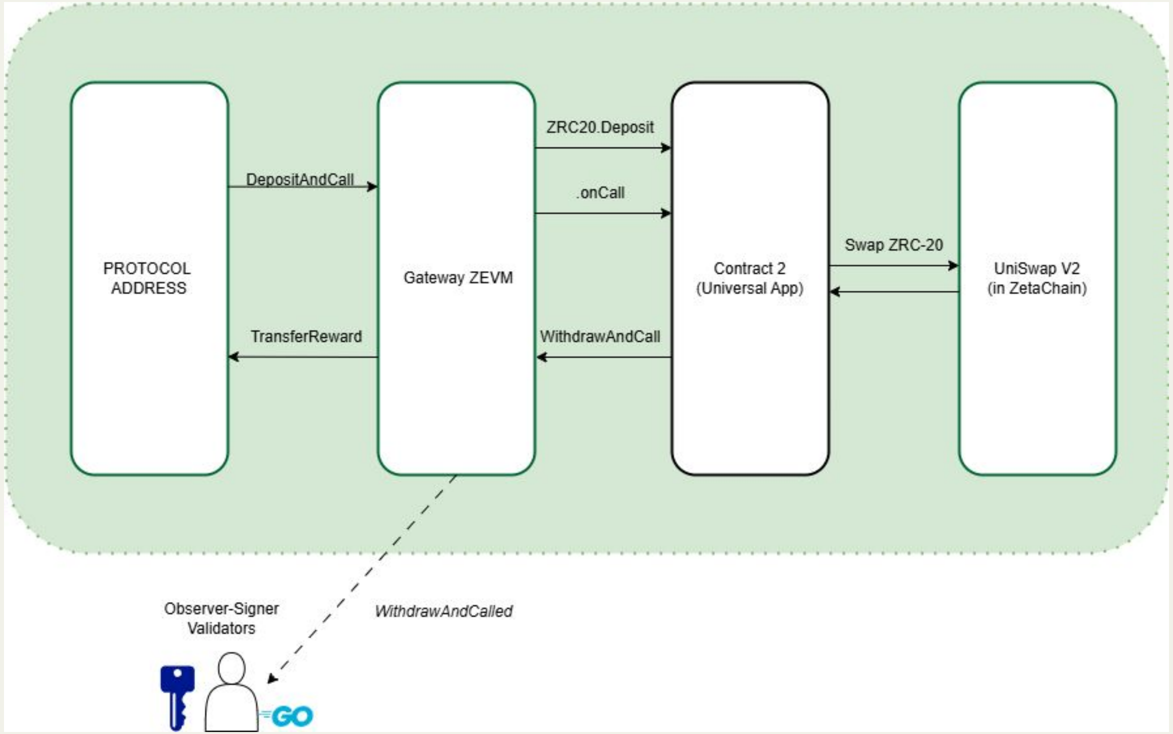
Emits a *withdrawAndCalled* **event** with transaction data

Validating Events from ZetaChain into External Chain



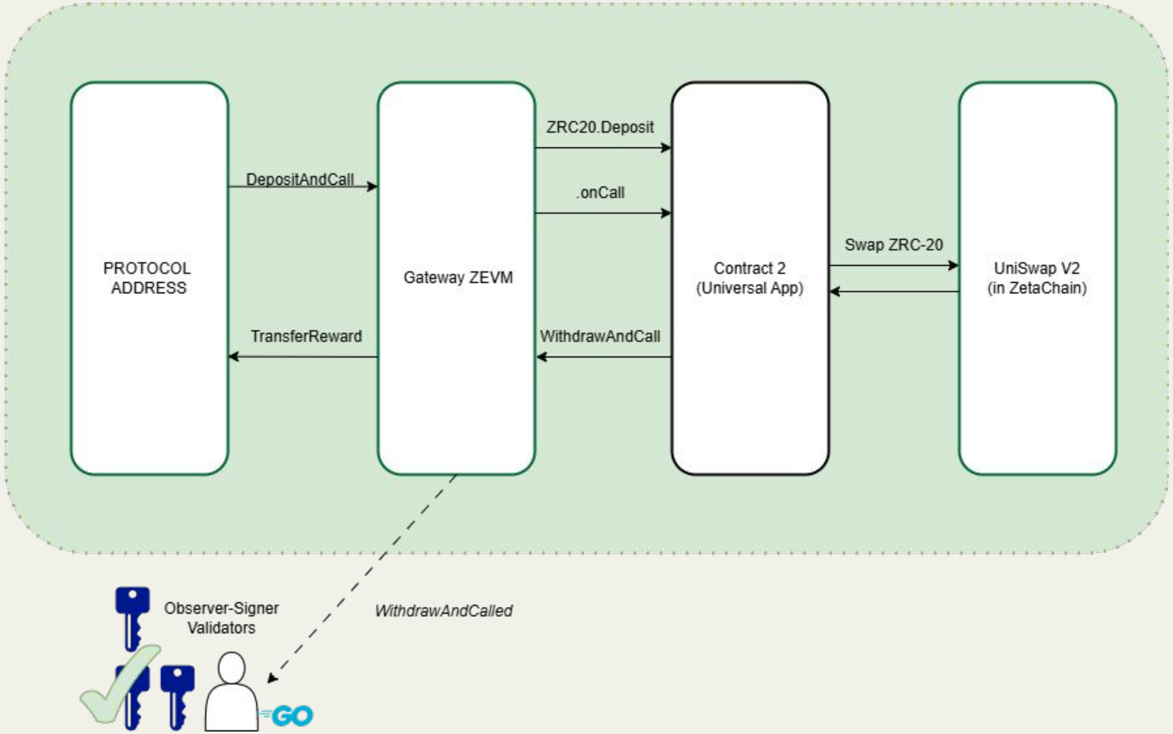
Observer-Signer validators monitor **blocks** in ZetaChain for any new events emitted

Validating Events from ZetaChain into External Chain



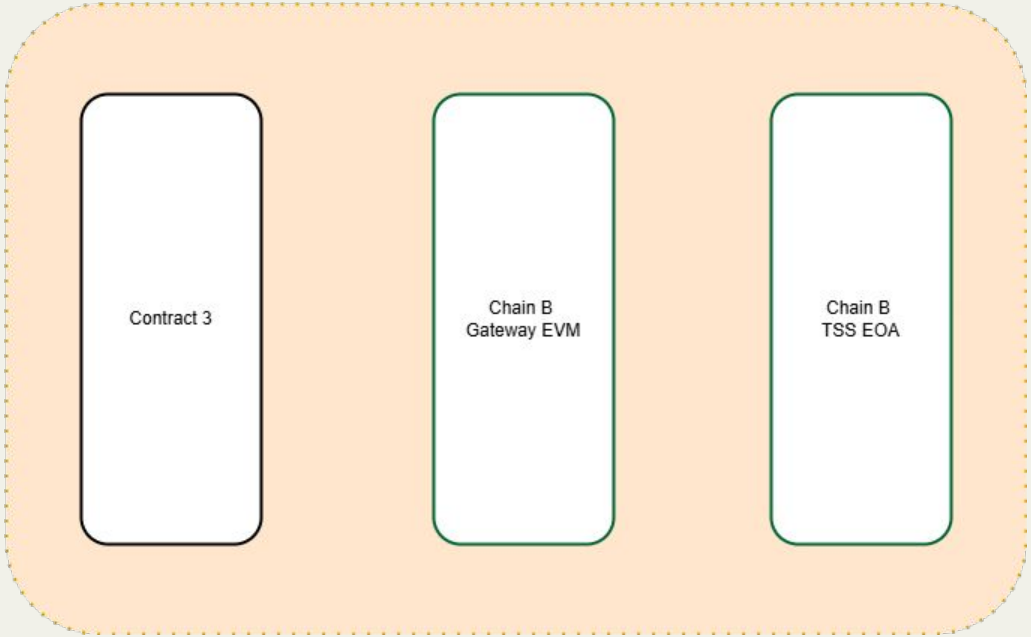
Vote on the validity of the observed pending "*outbound message*" by leveraging TSS

Validating Events from ZetaChain into External Chain

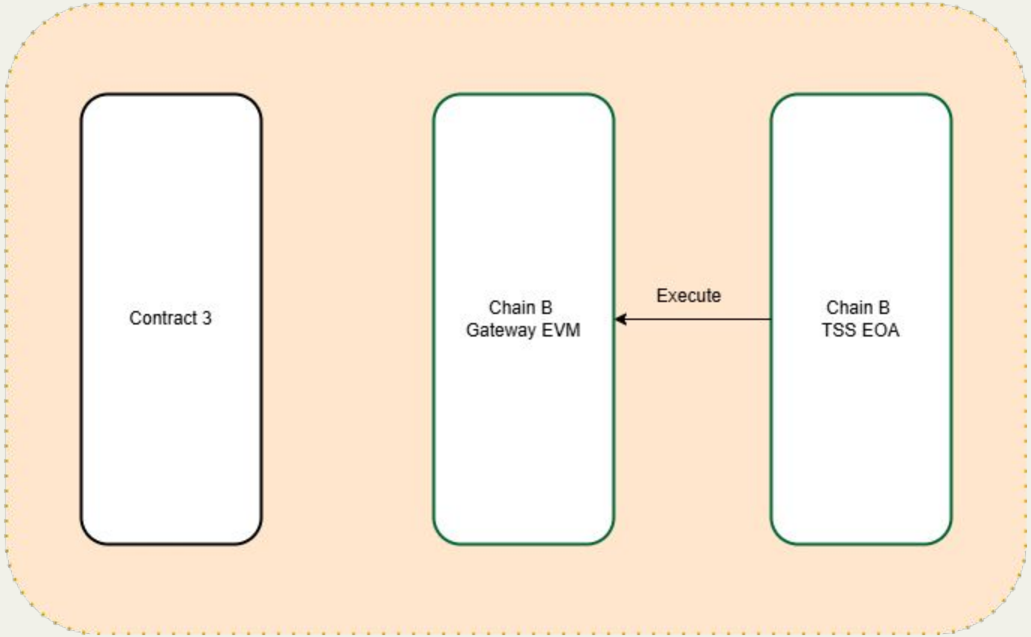


Once voting passes, **Chain B’s TSS EOA** initiates corresponding transaction in Chain B

Inside Chain B

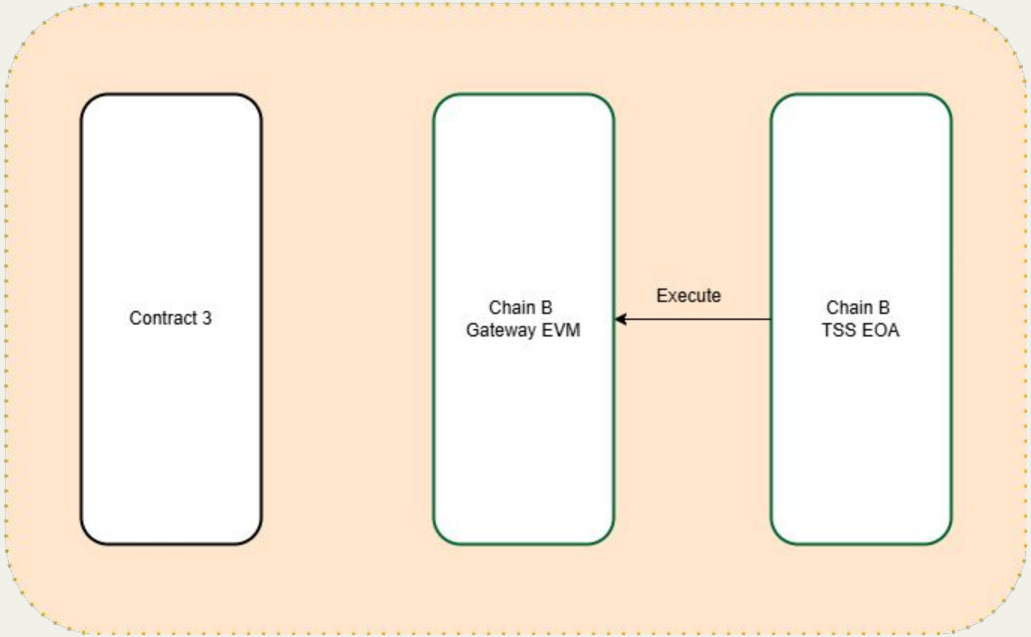


Inside Chain B



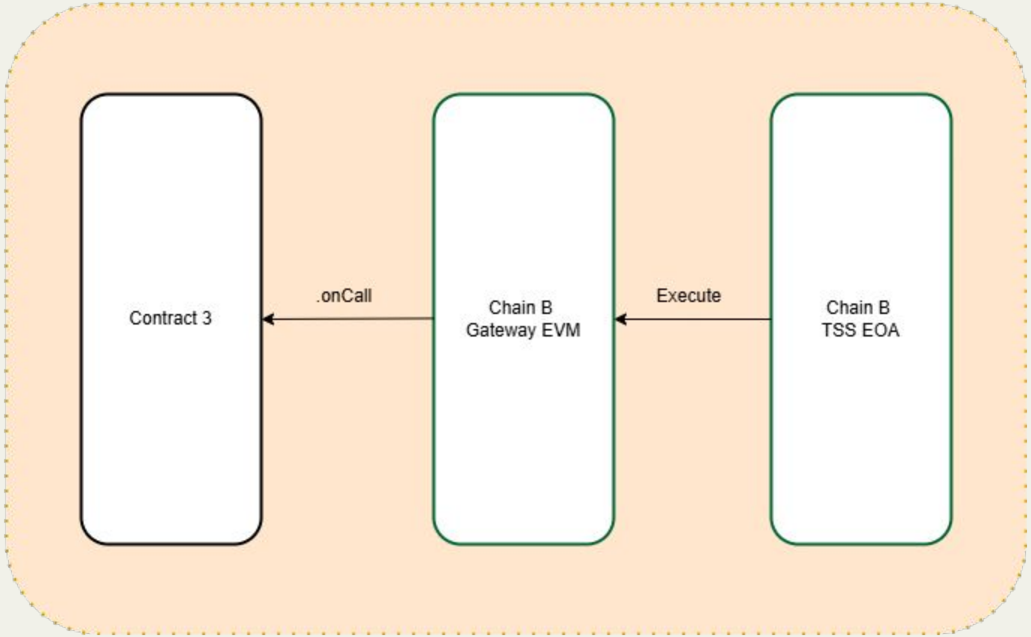
Chain B TSS EOA invokes *execute* of Chain B’s GatewayEVM with transaction data

Inside Chain B



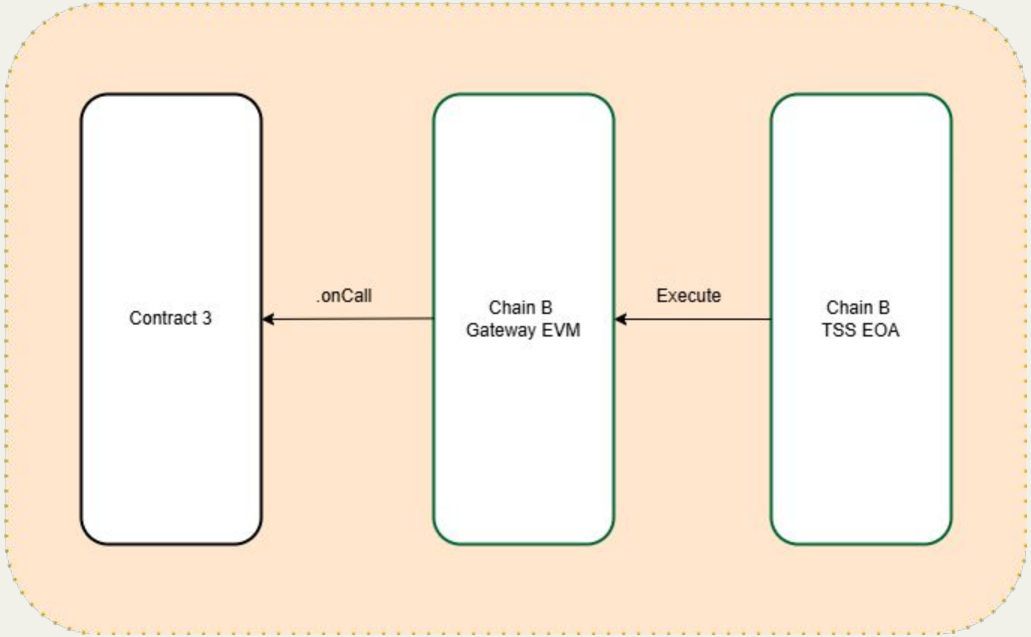
Only Chain B TSS EOA is authorized to invoke GatewayEVM's *execute*

Inside Chain B



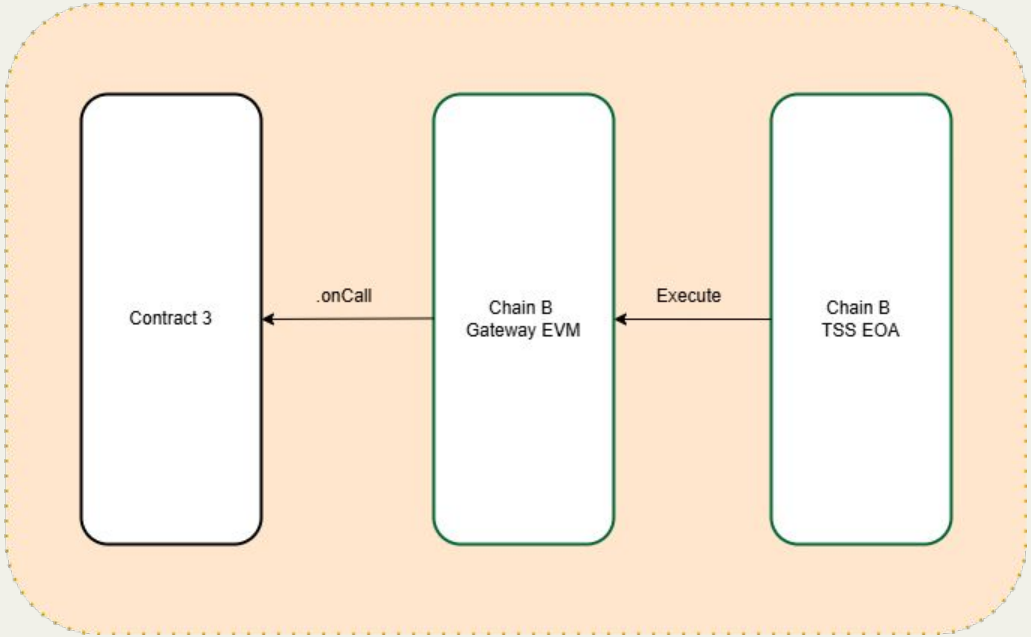
Execute invokes *.onCall* of our Contract 3 in Chain B with transaction data passed

Inside Chain B



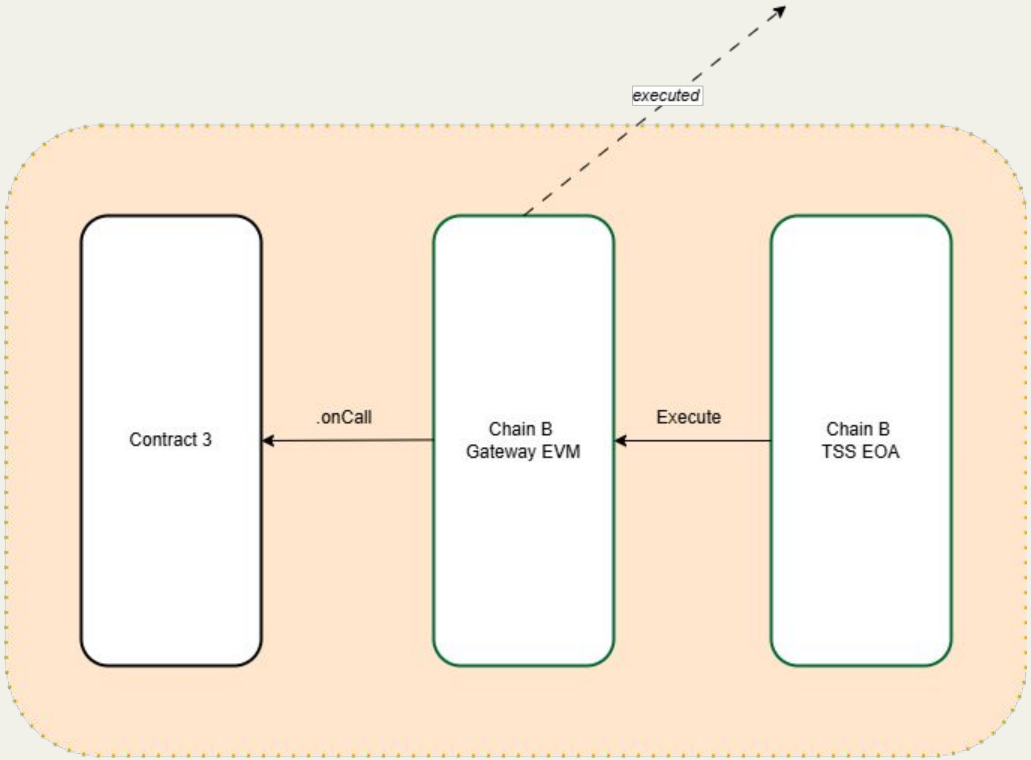
All Contracts on External Chain needs to implement a *.onCall* method

Inside Chain B



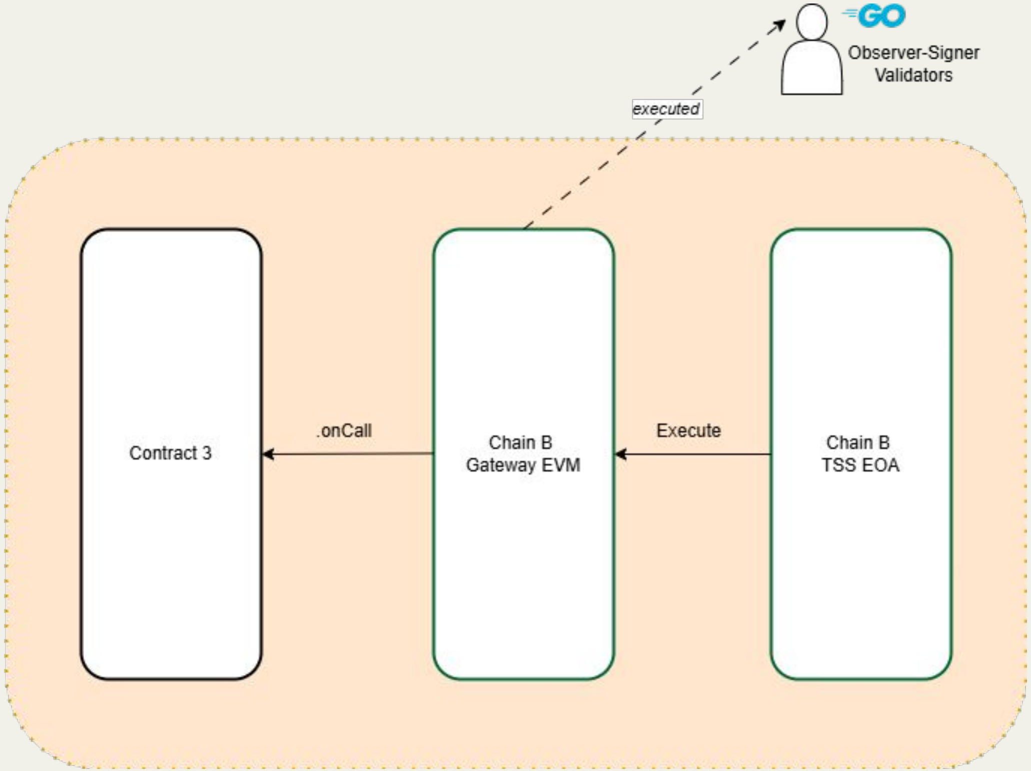
Entry point for any outgoing transaction from ZetaChain to Deployed External Chain

Inside Chain B



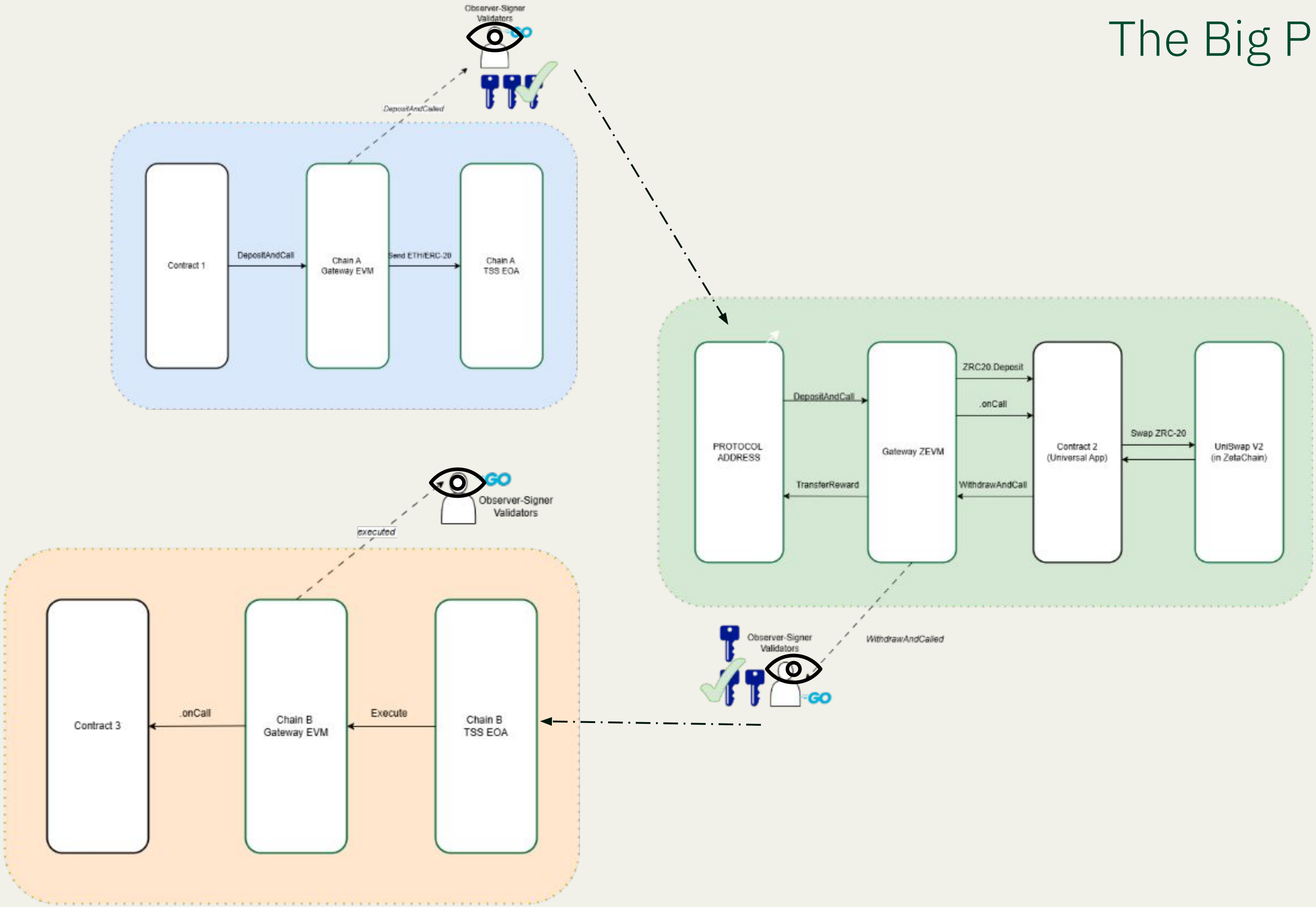
Once *.onCall* succeeds, Gateway EVM emits *executed* **event**

Inside Chain B



executed **event** will be monitored and voted by Observer-Signer Validators to confirm CCTx completion

The Big Picture



Back to our Problem

How to Perform Computational Arbitrage?

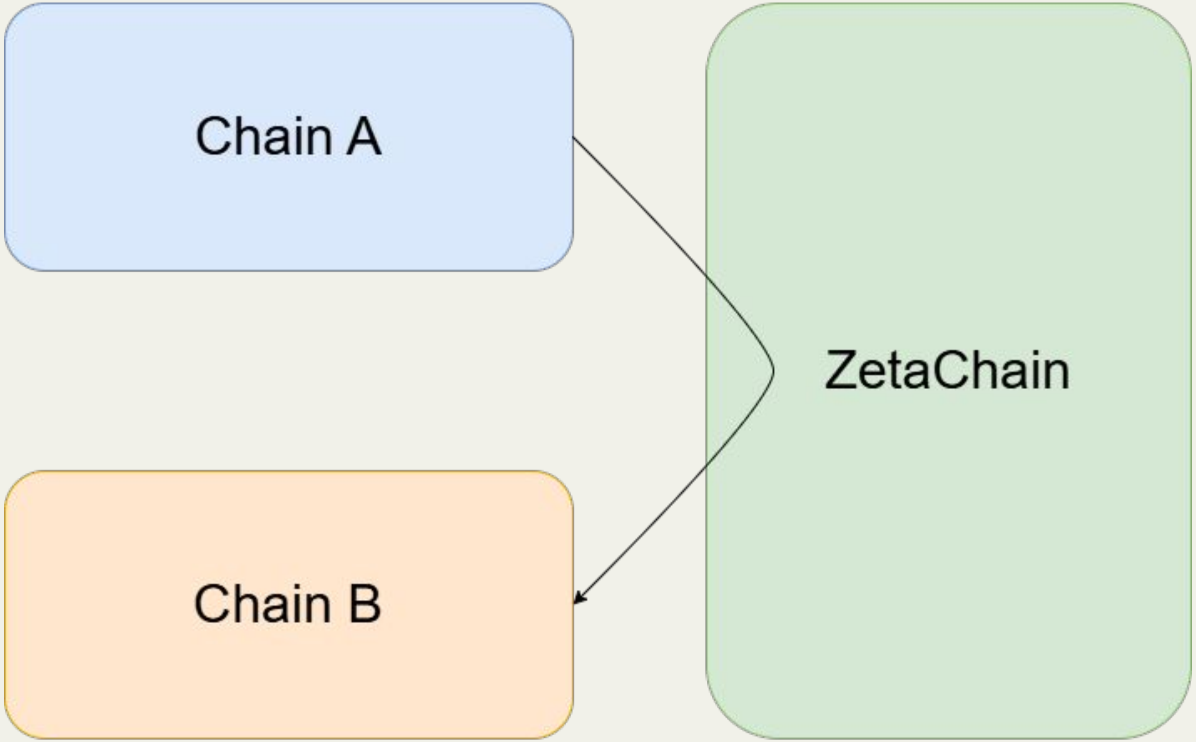
1

Naive Method

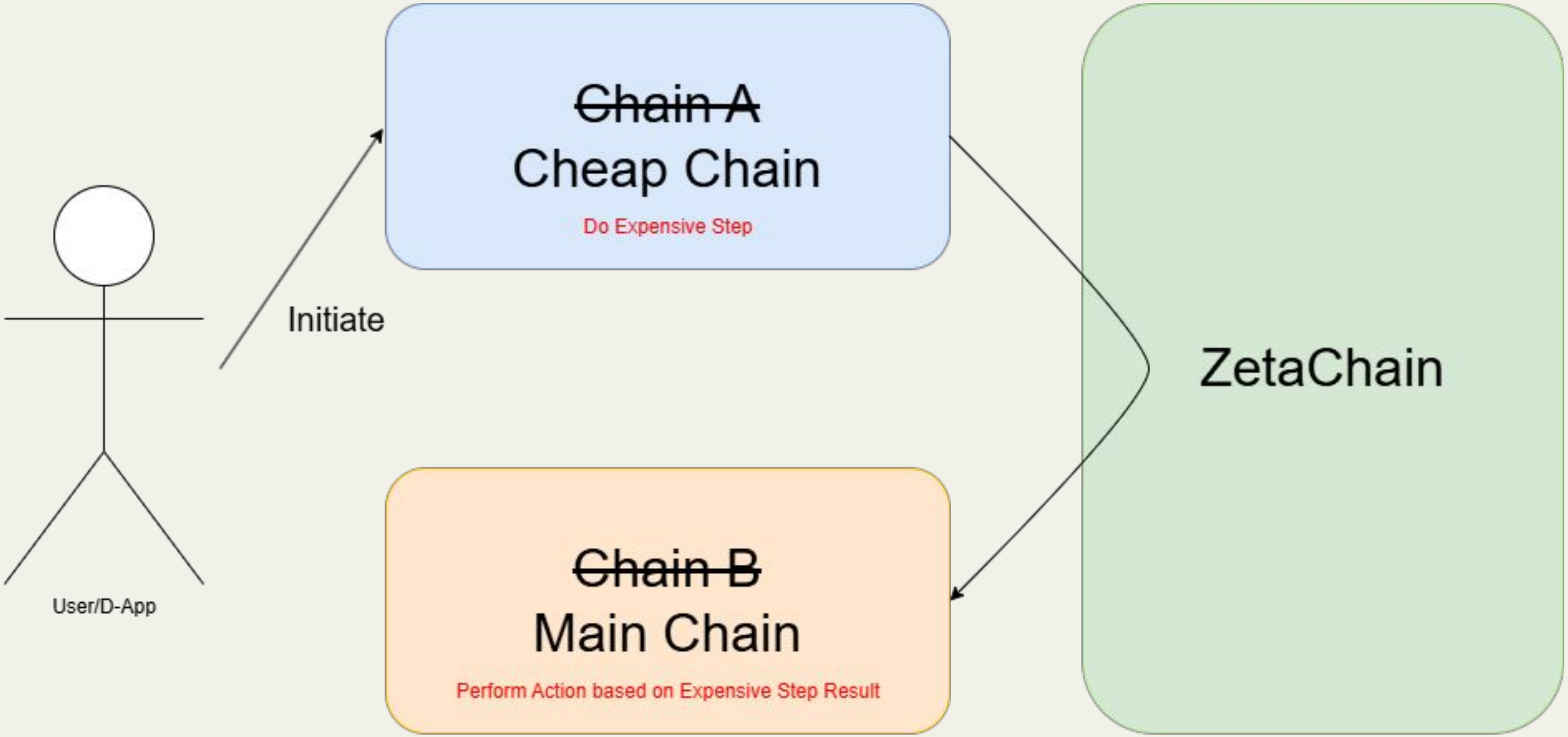
2

Proof of Concept Method

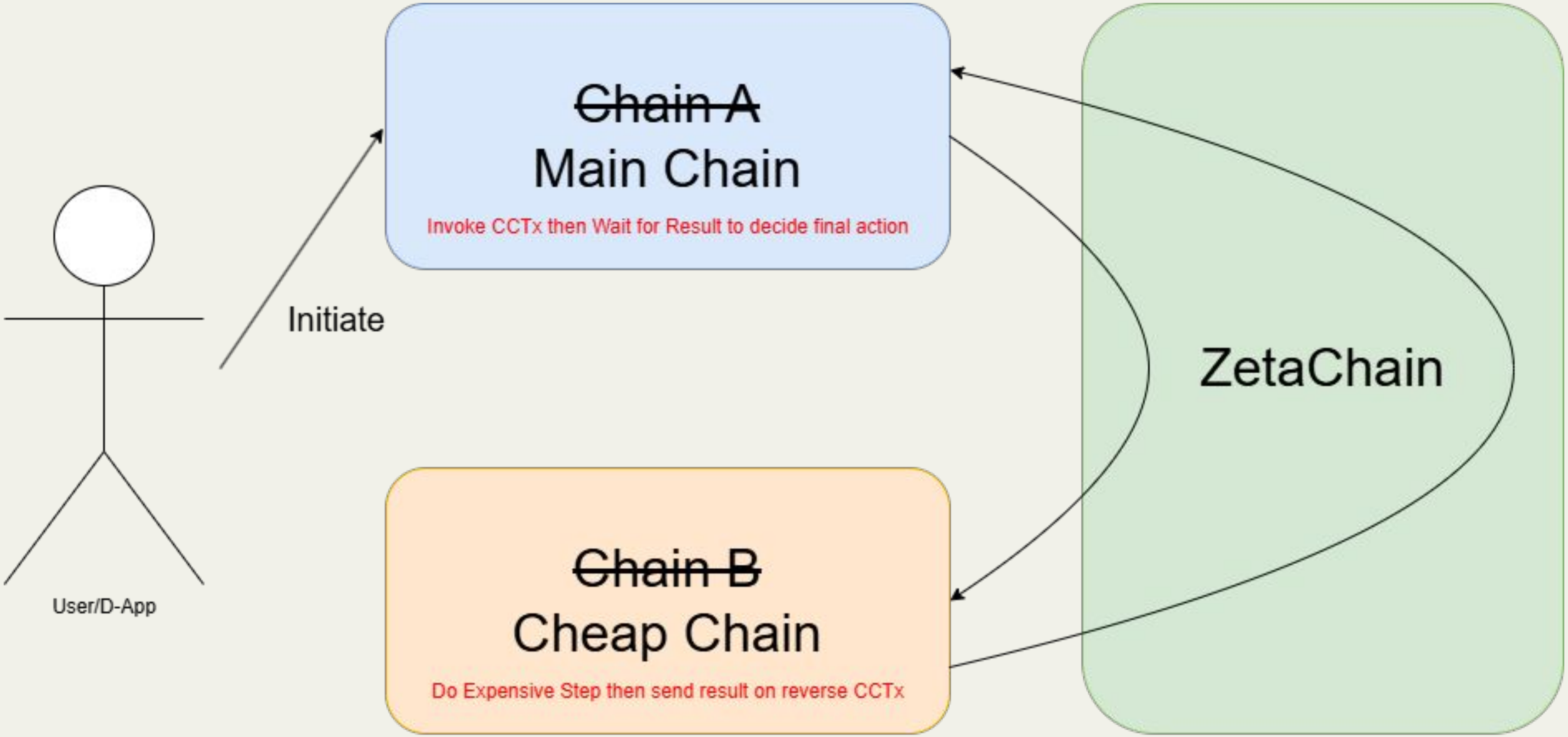
What We Have



Computation Arbitrage - The Naive Method (We also have PoC for this)

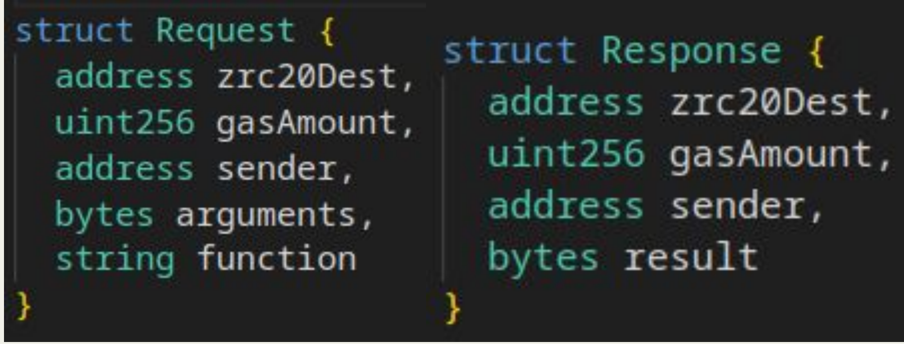


Computation Arbitrage - The Proof of Concept



~~Chain A~~
Main Chain
Invoke CCTx then Wait for Result to decide final action
Initiate
User/D-App
ZetaChain
~~Chain B~~
Cheap Chain
Do Expensive Step then send result on reverse CCTx

Two-Way "Promise-based" Transaction:

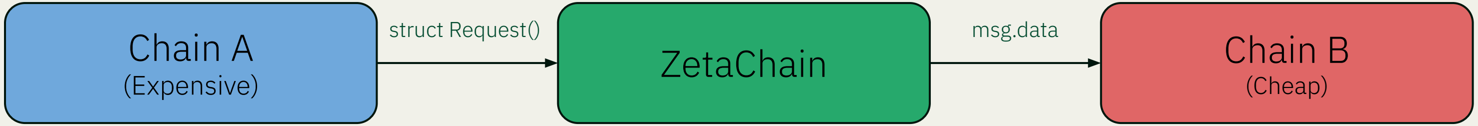


```
struct Request {
  address zrc20Dest,
  uint256 gasAmount,
  address sender,
  bytes arguments,
  string function
}
```

```
struct Response {
  address zrc20Dest,
  uint256 gasAmount,
  address sender,
  bytes result
}
```

Chain A begins a computation, encounters an **expensive** function

Wants to offload to Chain B for **cheaper** gas



Chain A (Expensive)

struct Request()

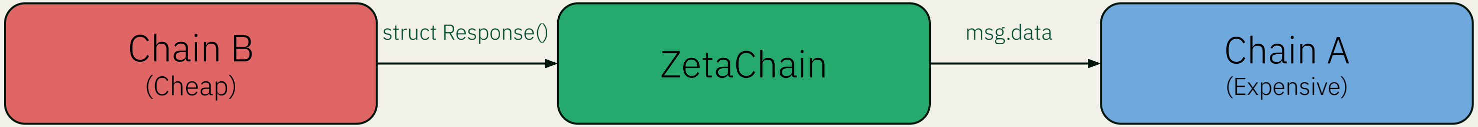
ZetaChain

msg.data

Chain B (Cheap)

Chain B computes the function requested on the content received

Packages the result and sends it back



Chain B (Cheap)

struct Response()

ZetaChain

msg.data

Chain A (Expensive)

Chain A can now take the result and continue whatever computation it intended.

Proof of Concept

We have Computational Arbitrage… But at What ‘*Cost*’?

Mais on ne peut vaincre le mal que par un autre mal

- Jean-Paul Sartre, I think

Observer-Signer Validator Concerns

- VOTING RUNS OFF-CHAIN
- ADMINS FEDERATE OBSERVER-SIGNER VALIDATORS
- CUSTOM CRYPTO LIBRARY...
- EVENTS MAY EXPOSE INTERNAL VARIABLES

Comet BFT has 5 second confirmation time but...

- OBSERVER-SIGNER VALIDATOR MONITOR **BLOCKS**
- THUS <u>WORSE CASE</u> SCENARIO:

  For CCTx to complete, it takes

  Chain A Confirmation Time +
  Chain B Confirmation Time + 5 Seconds

The Problem of Reverting Cross-Chain Transactions (CCTx)

- MAIN ISSUE - CCTX IS NON ATOMIC
- DEPENDING ON WHERE IT FAILS, MAY REQUIRE CROSS-CHAIN REVERT TRANSACTION
- AS A RESULT, GAS FEE INCREASES UPON FAILURE
- REVERT SCOPE IS ONLY WITHIN 1 TRANSACTION… NOT SUITABLE FOR 2 STEP PROCESSES

Even performing CCTx is **expensive** due to multiple transactions...

It may not even be worth the gas difference

NO TRUST MECHANISM BUILT IN...

Must trust Gateway, but Trusted Gateway Address is posted on their website...

Q&A

Top 3 Resources:

- https://github.com/zeta-chain/node/tree/develop/docs
- https://github.com/zeta-chain/protocol-contracts
- https://github.com/zeta-chain/example-contracts

Thank you for the semester!