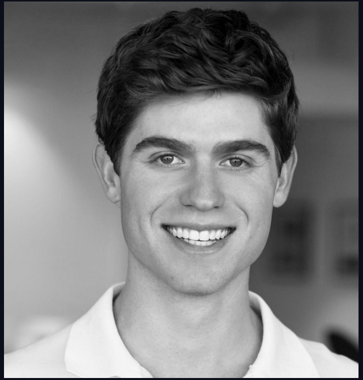
zetachain

With you today

Charlie McCowan
Head of DevOps & Cyber Security



Jonathan Covey
Head of Community

Agenda

- What’s Multichain? Omnichain?
- What is ZetaChain? How does it work?
- Workshop: Building Omnichain Smart Contracts
  - Omnichain NFT
  - Omnichain Swap
- Opportunities in Omnichain
- Q&A

Download Links

Presentation:
https://docs.google.com/presentation/d/1HJlga5J2l_l69hgdGEGQs6Q_mqySWpfRn5WGml5t2wU/edit?usp=sharing

GitHub Repo: https://github.com/zeta-chain/zetachain

Faucet: https://labs.zetachain.com/faucet

Explorer: https://explorer.zetachain.com

Docker Image: `docker pull ghcr.io/zeta-chain/zetachain`

Team and Intros!

Founder

Ankur

First Engineering/Product Manager at Coinbase. Co-founder of Basic Attention Token. Investor at Ribbit Capital. Advisor to Brave, 0x, VY Capital, and MobileCoin.

Engineering

Panruo

Professor at University of Houston, Ph.D in Computer Science at UC Riverside. Research focus in high performance and distributed computing.

Product

Brandon

Founded Yada (acq. 2020). Symbolic Systems at Stanford. Previously at Udacity, BuzzFeed. Early stage startup design/engineering advisor.

Partnerships

Ylong

Prev. partner for growth at crypto finance unicorn, previously co-founder of a sequoia-led DeFi project and investor at global crypto hedge fund.

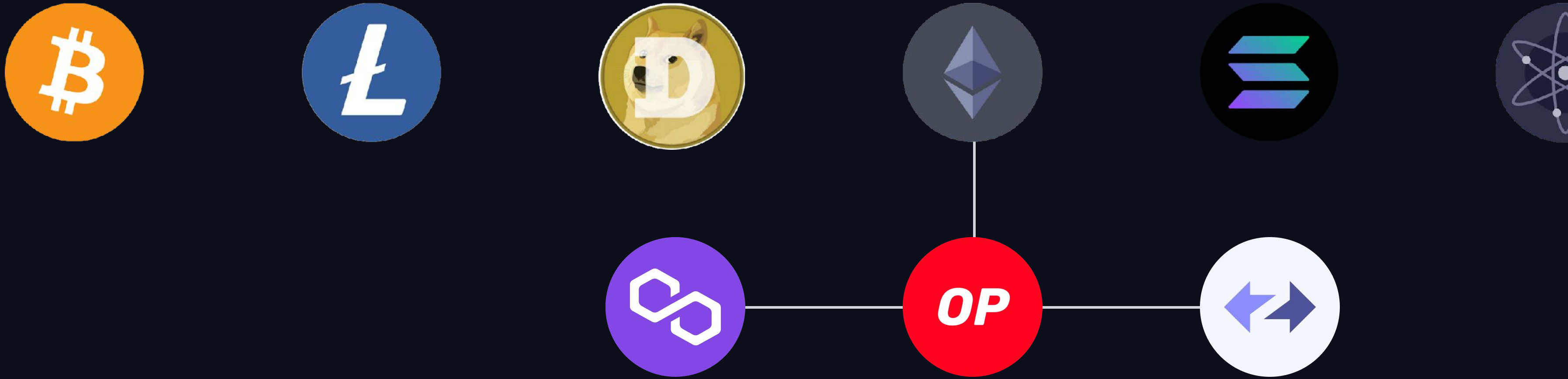
Marketing

Teigi

Head of Growth Marketing & Expansion at Blockchain.com, Chief of Staff and Marketing Director at OKEx, Bicentive, Microsoft.

Total Team Size: **25**

Crypto is becoming multichain



Checkpoint: Crypto Is multichain

Who has deployed contracts on at least two chains?

Who has tried other cross-chain solutions? (Rainbow Bridges, LayerZero, Axelar, etc)

Is anyone a maximalist?

Progression of multichain so far

New L1s and L2s (closed systems)

Centralized exchanges

Pairwise bridges with varying security models

Cross-chain bridge aggregators

Cross-chain messaging

Current multichain solutions are...

High-risk

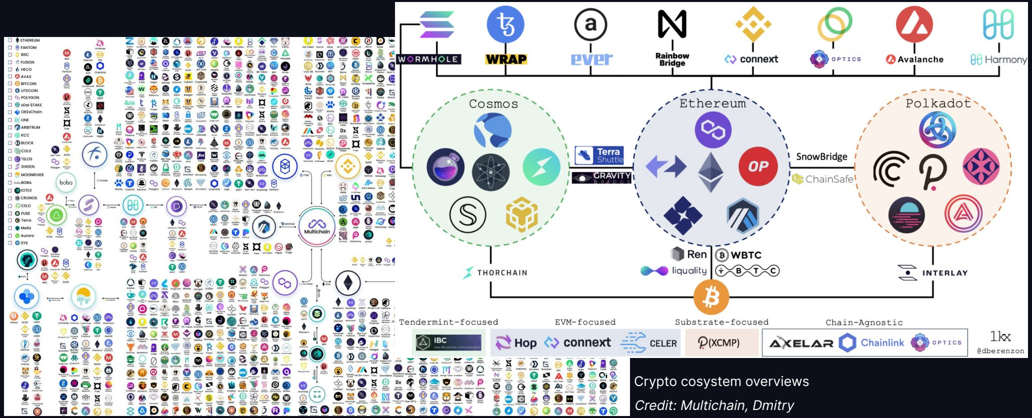
Locked assets, expensive exploits, and have varying trust models (often centralized, not trustless)

Fragmented

Separate products for every chain or asset pair, frustrating UX, siloed users and assets, and complexity beyond comprehension

Restrictive

With no standard for cross-chain, devs roll their own trust models/solutions or accept being bounded to certain chains/functionality



Crypto cosystem overviews

Credit: Multichain, Dmitry Berenzon

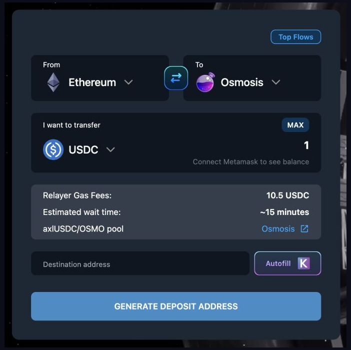
>100 public blockchains

>50 bridge projects

>1000s of wrapped assets

>$2b in bridge exploits

What people think interoperability is



Top Flows

From: Ethereum
To: Osmosis

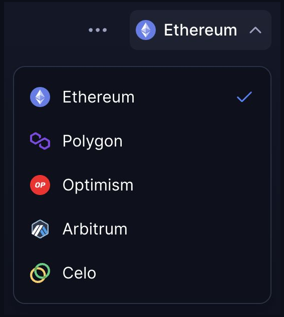
I want to transfer: USDC — 1 (MAX)
Connect Metamask to see balance

Relayer Gas Fees: 10.5 USDC
Estimated wait time: ~15 minutes
axlUSDC/OSMO pool: Osmosis

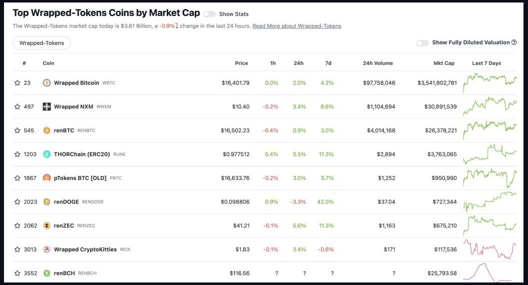
Destination address — Autofill

GENERATE DEPOSIT ADDRESS

Cross-chain swap and pairwise bridges



Supporting multiple chains for the same app



Top Wrapped-Tokens Coins by Market Cap — Show Stats

The Wrapped-Tokens market cap today is $3.61 Billion, a -0.9% change in the last 24 hours. Read More about Wrapped-Tokens

Wrapped-Tokens — Show Fully Diluted Valuation

#	Coin	Price	1h	24h	7d	24h Volume	Mkt Cap
23	Wrapped Bitcoin WBTC	$16,401.79	0.0%	2.0%	4.2%	$97,758,046	$3,541,802,781
497	Wrapped NXM WNXM	$10.40	-0.2%	3.4%	8.6%	$1,104,694	$30,891,539
545	renBTC RENBTC	$16,502.23	-0.4%	0.9%	3.0%	$4,014,168	$26,378,221
1203	THORChain (ERC20) RUNE	$0.977512	0.4%	5.5%	11.3%	$2,894	$3,763,065
1867	pTokens BTC [OLD] PBTC	$16,633.76	-0.2%	3.0%	5.7%	$1,252	$950,990
2023	renDOGE RENDOGE	$0.098806	0.9%	-3.3%	42.0%	$37.04	$727,344
2062	renZEC RENZEC	$41.21	-0.1%	5.6%	11.3%	$1,163	$675,210
3013	Wrapped CryptoKitties WCK	$1.83	-0.1%	3.4%	-0.6%	$171	$117,536
3552	renBCH RENBCH	$116.56	?	?	?	?	$25,793.58

Wrapped tokens everywhere

What interoperability can be

- Pay from + receive into any wallet
- No more wrapped tokens (nor related bridge hacks)
- Chain-agnostic NFTs + Fungible Tokens (ERC20s, etc.)
- Abstraction of network/chain to end-users
- DeFi, Games, Marketplaces, and more that operate with different chains seamlessly (not separately)
- Omnichain portfolio management and accounting
- The ability to use specialized blockchains for specific features
- And much, much more

How interoperability benefits Devs

- Lets you focus on building great Dapps
- Focus on Dapp features instead of blockchain chain integration
- Avoid Chain Lock-In
- More flexibility in the design and operation of your dapps
- Larger user base -- Connect with your users on any chain
- Interact with Bitcoin and other non programmable assets

The next step for multichain tooling

New L1s and L2s (closed systems)

Centralized exchanges

Pairwise bridges with varying security models

Cross-chain bridge aggregators

Cross-chain messaging

Omnichain Smart Contracts

We need a complete and universal toolkit to build real Omnichain applications.

ZetaChain is the only public, decentralized blockchain and smart contract platform built for Omnichain interoperability.

A truly interoperable L1

- PoS, Tendermint Consensus
- Chain and layer-agnostic TSS
- Single-step transactions
- Unified liquidity

Omnichain Smart Contracts

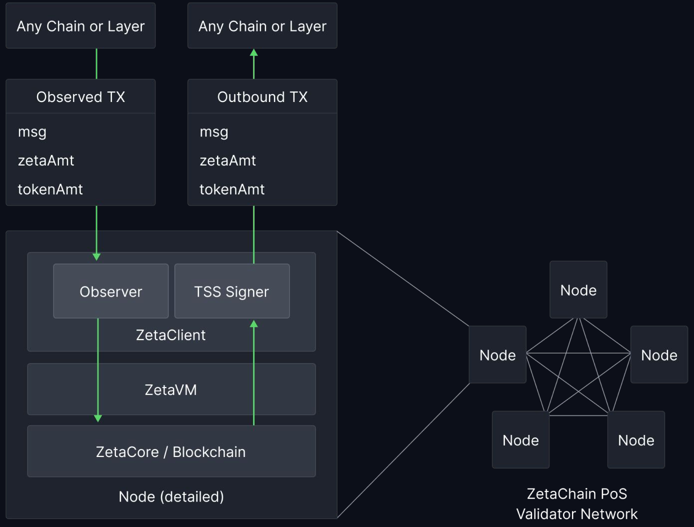
Manage assets across layers and chains enabling even Bitcoin smart contracts

Cross-chain messaging

Existing smart contracts can send data and value across chains and layers with simple function calls

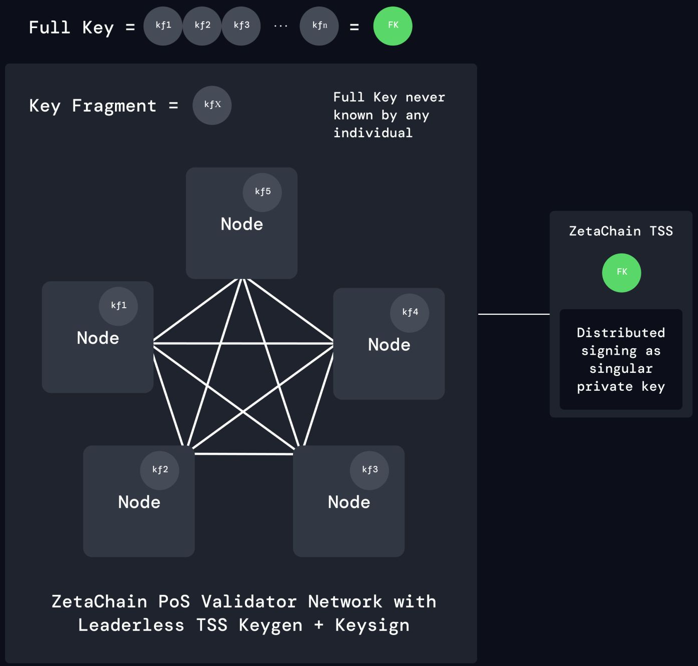
Hyper-Connected Node Architecture

- Nodes observe and reach consensus connected chains’ transactions
- Nodes built on battle-tested Cosmos SDK, Tendermint Consensus
- UTXO-like cross-chain transactions enable safe, chain-agnostic message passing and value transfer
- EVM-compatible smart contracts on ZetaVM can read from and write to external chains.



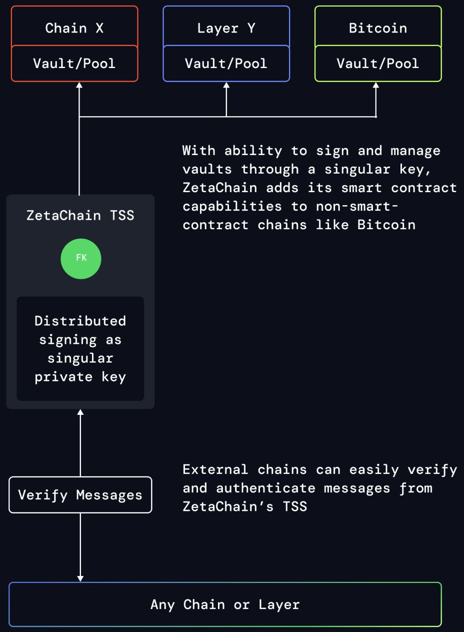
Hardened Security

- Decentralized PoS network
- TSS keygen and keysign
- Extensive defense against arbitrary and infinite minting
- ZETA intermediary token
    - Minimizes attack surface
    - Simplifies apps


Full Key = kf1 kf2 kf3 ··· kfn = FK
Key Fragment = kfX
Full Key never known by any individual
kf5 Node
kf1 Node
kf4 Node
kf2 Node
kf3 Node
ZetaChain PoS Validator Network with Leaderless TSS Keygen + Keysign
ZetaChain TSS
FK
Distributed signing as singular private key

Externally Managed Assets

- Verify and sign transactions on any connected chain
- Manage assets on multiple chains as easily as a normal L1 contract can on a single chain
- Bring Omnichain smart contract logic to all chains, even non-smart-contract chains like **Bitcoin, Dogecoin**
- Cold-wallet level security with hot-wallet level functionality, on every chain


Chain X
Vault/Pool
Layer Y
Vault/Pool
Bitcoin
Vault/Pool
With ability to sign and manage vaults through a singular key, ZetaChain adds its smart contract capabilities to non-smart-contract chains like Bitcoin
ZetaChain TSS
FK
Distributed signing as singular private key
Verify Messages
External chains can easily verify and authenticate messages from ZetaChain's TSS
Any Chain or Layer

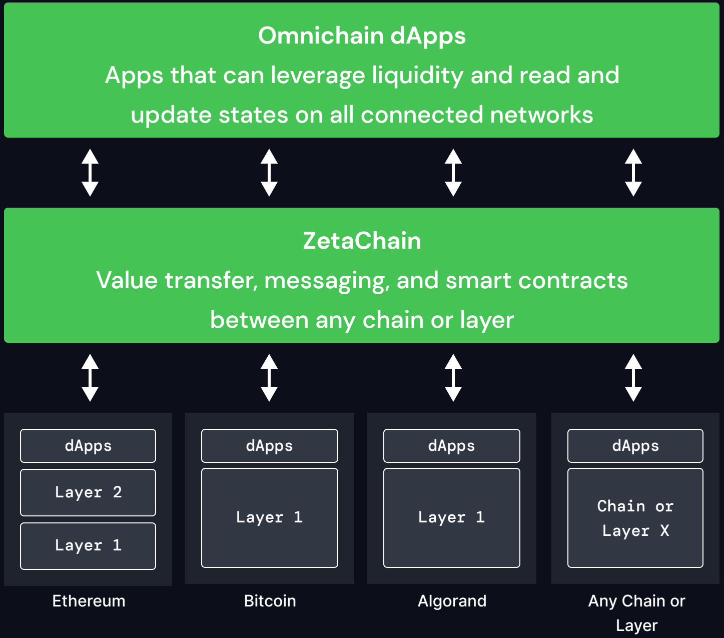
A simpler, powerful dev experience

Messaging

- Simple, intuitive cross-chain messaging API. Adding cross-chain to an existing dApp is as simple as: send, receive, handle-revert.

Smart Contracts

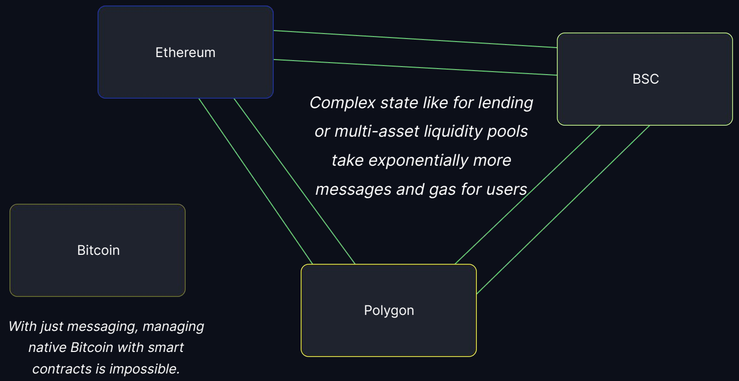
- EVM-compatible Omnichain smart contracts let existing developer skills translate 1:1
- Build complex multichain applications from a single smart contract, as if all on one chain.
- Easily deploy and extend existing EVM protocols like Curve, Aave, etc. for multichain.

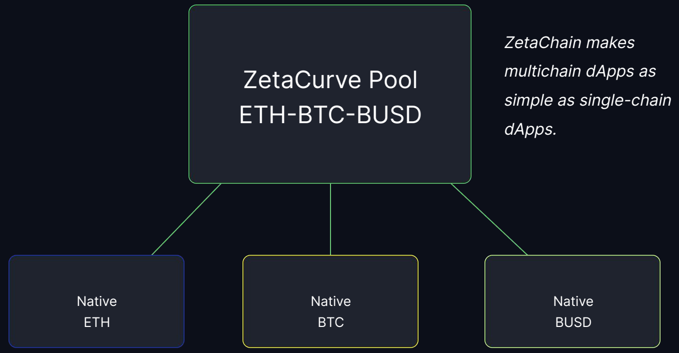

Omnichain dApps
Apps that can leverage liquidity and read and update states on all connected networks
ZetaChain
Value transfer, messaging, and smart contracts between any chain or layer
dApps
Layer 2
Layer 1
Ethereum
dApps
Layer 1
Bitcoin
dApps
Layer 1
Algorand
dApps
Chain or Layer X
Any Chain or Layer

A new era of interoperability

Cross-chain messaging enables 1-to-1 chain interactions for existing contracts, but more complex apps are complicated and/or impossible.

Chain-agnostic, interoperable smart contracts let you manage native assets and data from a single place, enabling a higher level of interoperability.


Ethereum
BSC
Complex state like for lending or multi-asset liquidity pools take exponentially more messages and gas for users
Bitcoin
Polygon
With just messaging, managing native Bitcoin with smart contracts is impossible.


ZetaCurve Pool ETH-BTC-BUSD
ZetaChain makes multichain dApps as simple as single-chain dApps.
Native ETH
Native BTC
Native BUSD

How does it compare?

	ZetaChain	Cosmos	Polkadot	THORChain	Bridges/Messaging
General Smart Contracts	**Yes**	**Yes**	**Yes**	No	No
Cross-Chain Value Transfer	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**
Cross-Chain Message Passing	**Yes**	**Yes**	**Yes**	No	**Yes**
Chain-Agnostic	**Yes**	Only Cosmos/IBC Chains	Only Parachains	**Yes**	Application-specific chain-pairs
Settlement	**Real Time Native Settlement**	Wrapped	Wrapped	**Real Time Native Settlement**	Wrapped, risk of redeem
Native Bitcoin Vault Management	**Yes**	No	No	**Yes**	Application-specific
Omnichain Smart Contracts	**Yes**	Only Cosmos/IBC Chains	Only Parachains	No	No

Ecosystem Opportunities

- Universal payments, tools, & operational improvements
- Omnichain DAOs: governance, asset management, clubs
- Omnichain NFTs: art projects, memberships, marketplaces, proof of ownership
- Omnichain DeFi: Curve, Aave, Compound, Yearn
- Omnichain Identity: universal usernames/domains, decentralized multichain wallets, multichain portfolio mgmt
- Hundreds of other ideas no one else has thought of yet!

Checkpoint - Opportunities

- Is anyone working on these types of projects?
- Any multichain projects you are excited about?

Workshop:
Building Omnichain smart contracts (1 of 2)

- Multi-Chain NFT
- Omnichain Swap
- Bonus: Omnichain Curve

Workshop:
Building Omnichain smart contracts (2 of 2)

Omnichain NFT

https://www.zetachain.com/docs/developers/quickstart-tutorials/deploy-an-Omnichain-nft

Omnichain Swap

https://www.zetachain.com/docs/developers/quickstart-tutorials/deploy-first-zevm-contract

Omnichain Curve

https://www.zetachain.com/docs/developers/omnichain-smart-contracts/examples/curve-s
ample-on-zevm/

Workshop: ForeSight Task Submission

https://github.com/openbuildcommuntiy/ForesightX-HackerHouse-HK

```
{
    “wallet_address”: “0x1234”,
    “omnichain_swap_contract_address”: “0x1234”,
    "nft_contract_address_goerli": "0x1234",
    "nft_contract_address_bsc": "0x1234",
    "curve_contract_address": "0x1234",
}
```

Checkpoint - Workshop

Who Is Planning To Complete These Tasks During This Session?

What Operating Systems?

- Linux
- Mac
- Windows

Docker

Workshop: Setup Local Environment

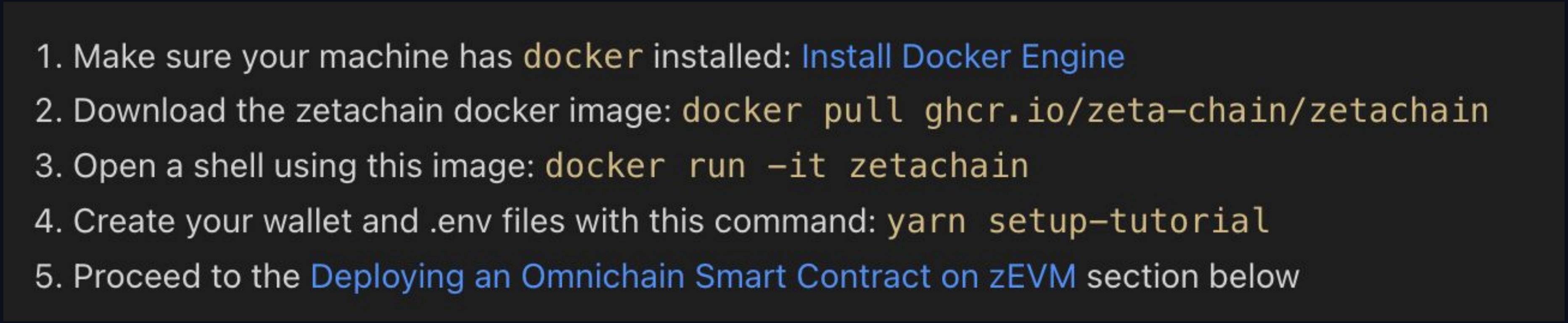
This tutorial will use the zetachain public repository as a starting point.

```
git clone git@github.com:zeta-chain/zetachain.git
```

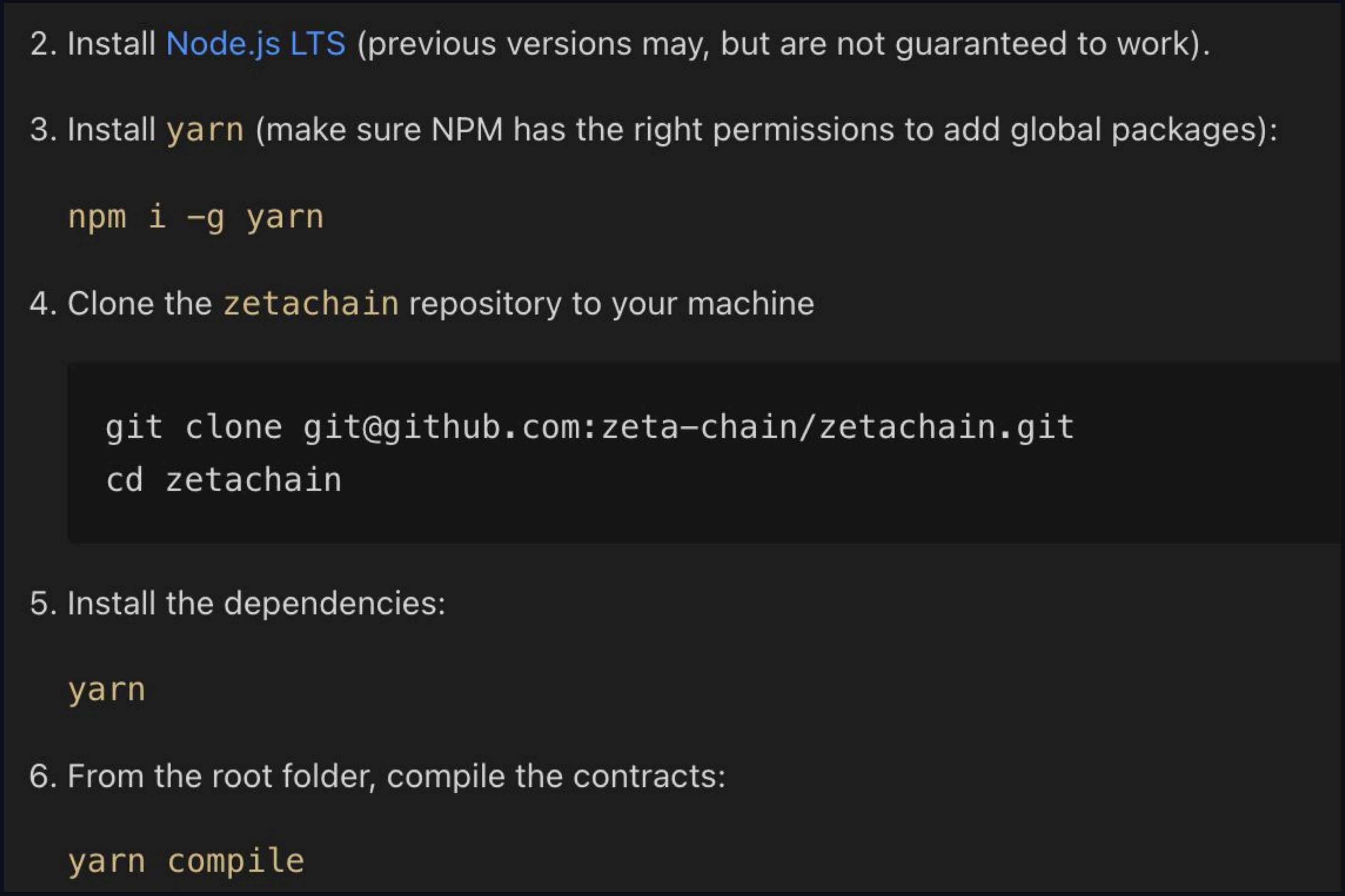
Two options

- Docker
- Configure Local Env Manually

Workshop: Setup Local Environment



1. Make sure your machine has `docker` installed: Install Docker Engine
2. Download the zetachain docker image: `docker pull ghcr.io/zeta-chain/zetachain`
3. Open a shell using this image: `docker run -it zetachain`
4. Create your wallet and .env files with this command: `yarn setup-tutorial`
5. Proceed to the Deploying an Omnichain Smart Contract on zEVM section below



2. Install Node.js LTS (previous versions may, but are not guaranteed to work).

3. Install `yarn` (make sure NPM has the right permissions to add global packages):

   ```
   npm i -g yarn
   ```

4. Clone the `zetachain` repository to your machine

   ```
   git clone git@github.com:zeta-chain/zetachain.git
   cd zetachain
   ```

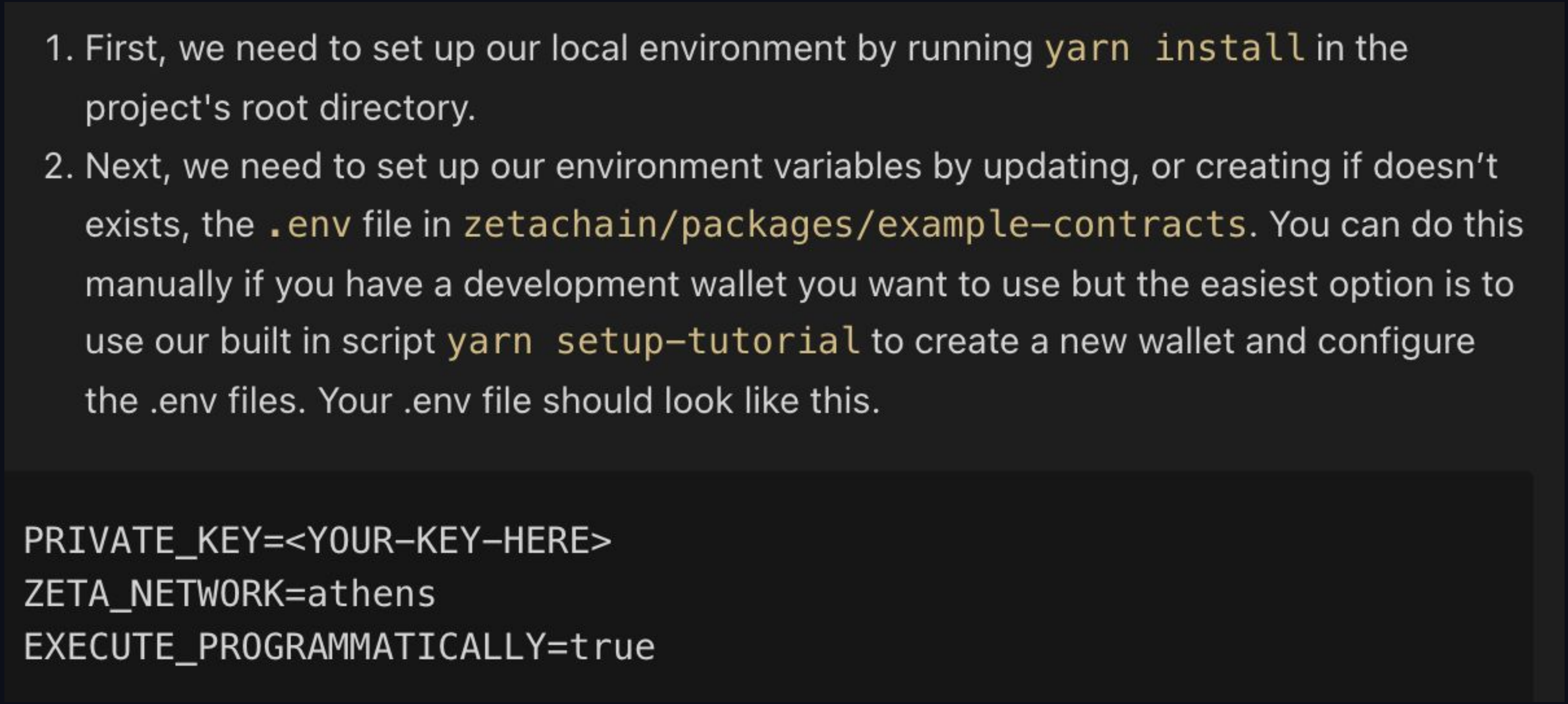
5. Install the dependencies:

   ```
   yarn
   ```

6. From the root folder, compile the contracts:

   ```
   yarn compile
   ```

WorkShop - Setup Local Environment



1. First, we need to set up our local environment by running `yarn install` in the project's root directory.
2. Next, we need to set up our environment variables by updating, or creating if doesn't exists, the `.env` file in `zetachain/packages/example-contracts`. You can do this manually if you have a development wallet you want to use but the easiest option is to use our built in script `yarn setup-tutorial` to create a new wallet and configure the .env files. Your .env file should look like this.

```
PRIVATE_KEY=<YOUR-KEY-HERE>
ZETA_NETWORK=athens
EXECUTE_PROGRAMMATICALLY=true
```

WorkShop - ZetaChain Compatibility

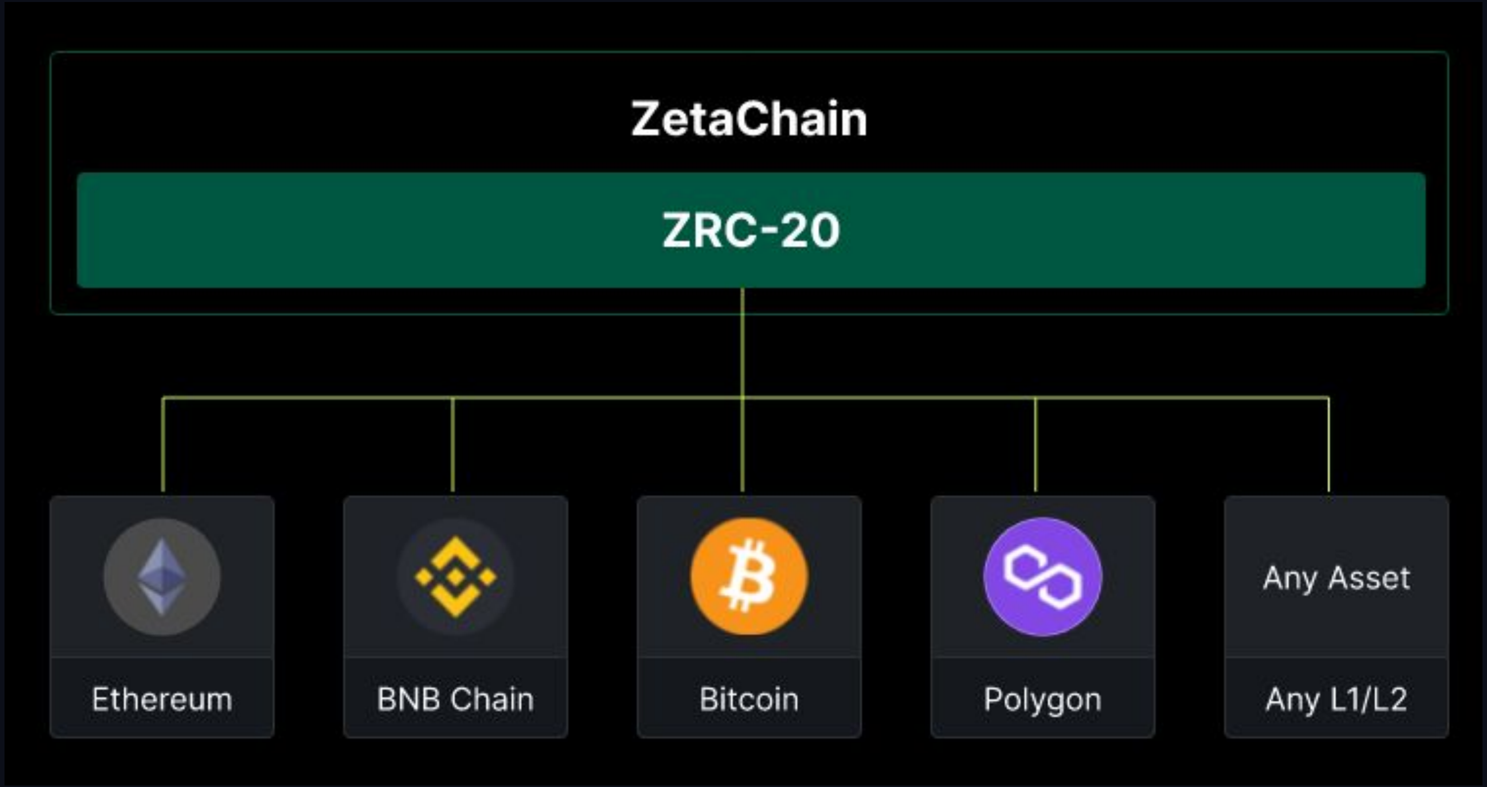
- ZetaChain functions like a regular EVM Network
- Same tooling: Hardhat, Metamask, Remix, Brownie, etc
- Cosmos Txs show up as synthetic EVM transactions
- Same languages: Solidity, JS/TS

Checkpoint - Setup Local Environment

- Cloned Repo
- Setup Local Environment
    - Setup Wallet
    - Setup .env file

- What are we doing again?
    - https://www.zetachain.com/docs/developers/quickstart-tutorials/deploy-first-zevm-contract/

WorkShop - ZRC20 Overview

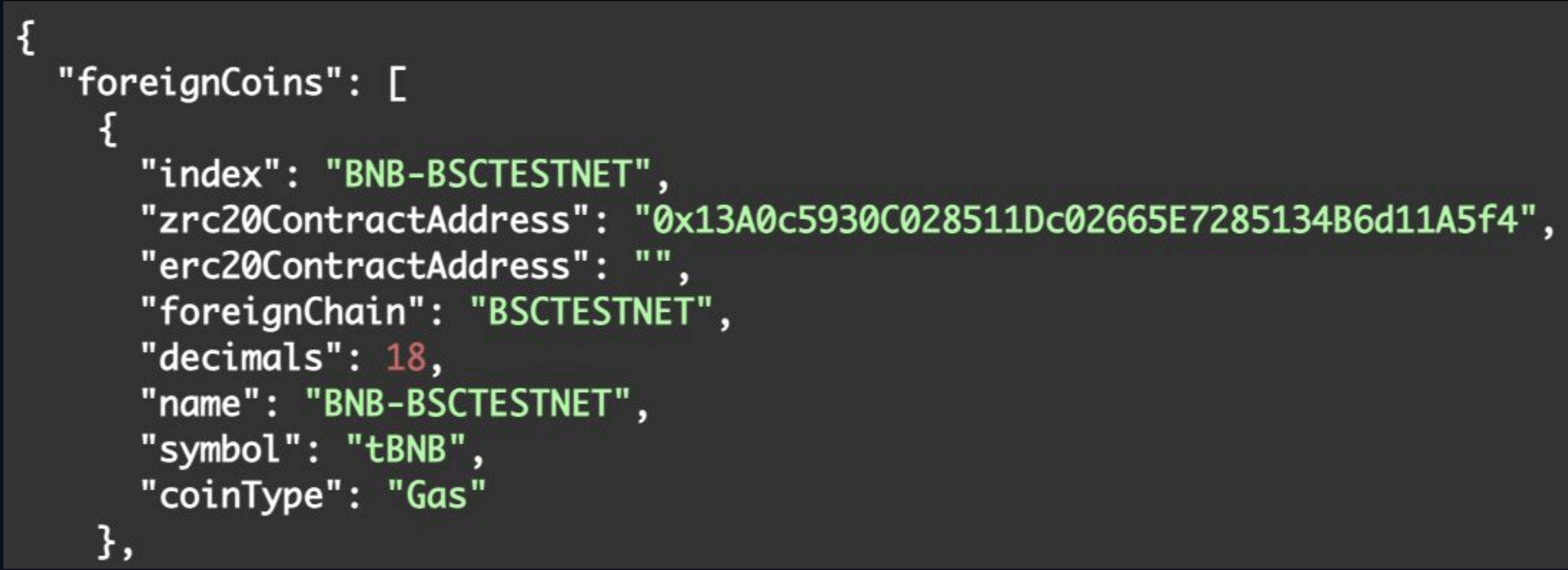


WorkShop - ZRC20 Overview

- Guide: https://www.zetachain.com/docs/developers/Omnichain-smart-contracts/zrc-20/
- Extension of the standard ERC-20 tokens
- No dApp contracts are needed on foreign chains.
- Manage assets across all ZetaChain-connected chains
- Works with any fungible token, including Bitcoin, Dogecoin, gas assets, etc
- User sends/deposits assets to the ZetaChain TSS address

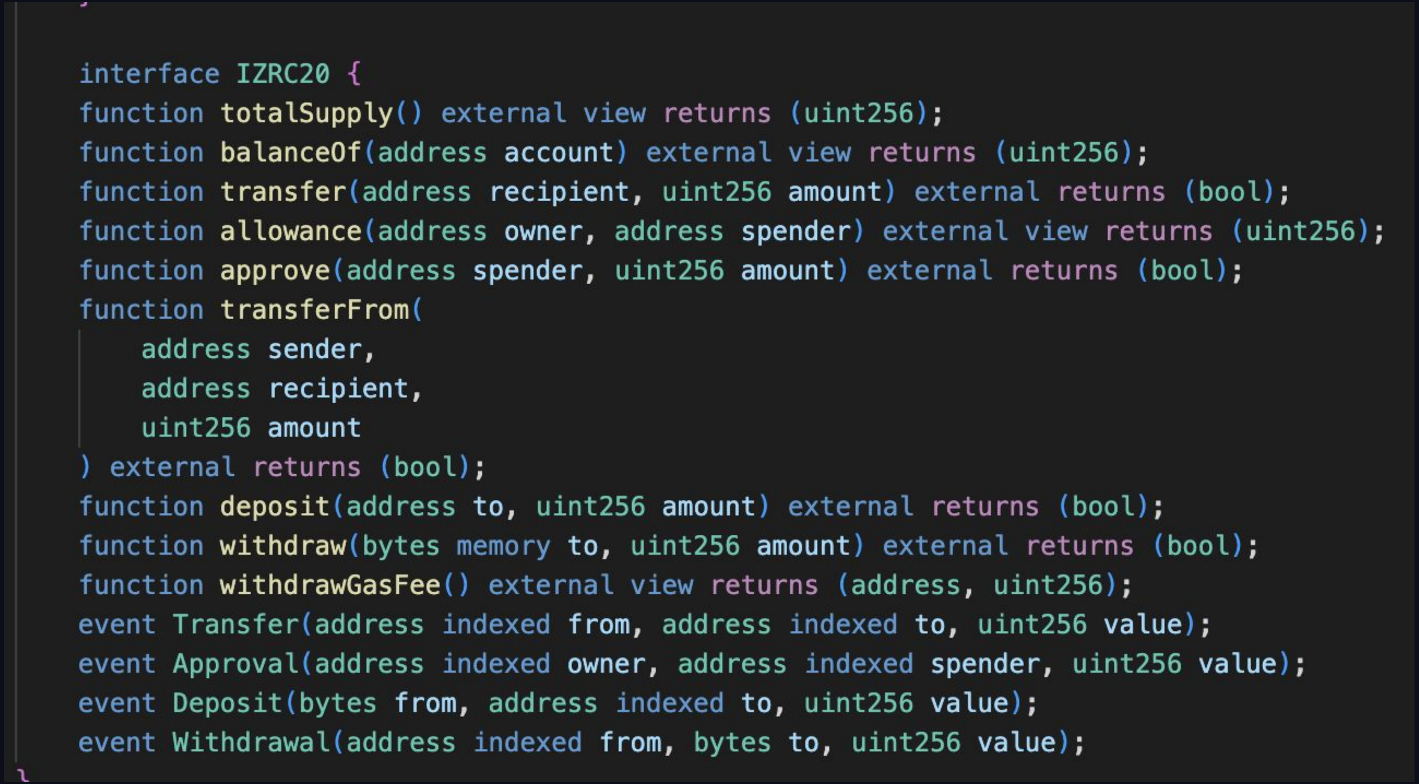
WorkShop - ZRC20 Overview

https://api.athens2.zetachain.com/#/Query/ZetachainZetacoreFungibleForeignCoinsAll



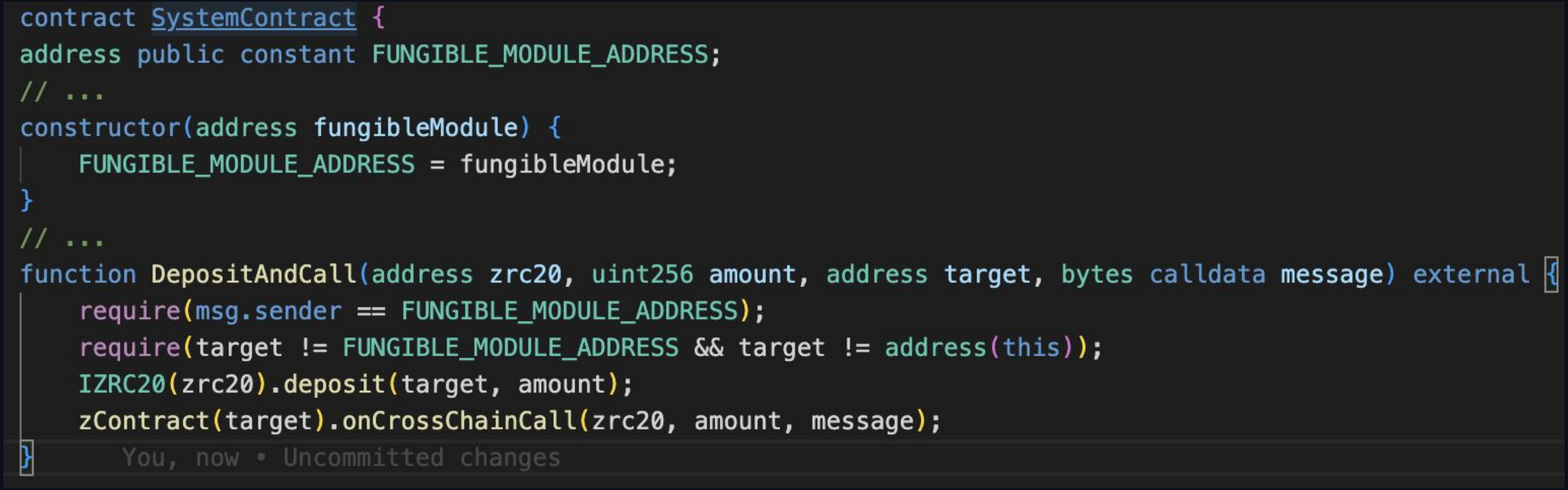
```
{
  "foreignCoins": [
    {
      "index": "BNB-BSCTESTNET",
      "zrc20ContractAddress": "0x13A0c5930C028511Dc02665E7285134B6d11A5f4",
      "erc20ContractAddress": "",
      "foreignChain": "BSCTESTNET",
      "decimals": 18,
      "name": "BNB-BSCTESTNET",
      "symbol": "tBNB",
      "coinType": "Gas"
    },
```

WorkShop - ZRC20 Interface



```
    interface IZRC20 {
    function totalSupply() external view returns (uint256);
    function balanceOf(address account) external view returns (uint256);
    function transfer(address recipient, uint256 amount) external returns (bool);
    function allowance(address owner, address spender) external view returns (uint256);
    function approve(address spender, uint256 amount) external returns (bool);
    function transferFrom(
        address sender,
        address recipient,
        uint256 amount
    ) external returns (bool);
    function deposit(address to, uint256 amount) external returns (bool);
    function withdraw(bytes memory to, uint256 amount) external returns (bool);
    function withdrawGasFee() external view returns (address, uint256);
    event Transfer(address indexed from, address indexed to, uint256 value);
    event Approval(address indexed owner, address indexed spender, uint256 value);
    event Deposit(bytes from, address indexed to, uint256 value);
    event Withdrawal(address indexed from, bytes to, uint256 value);
}
```

WorkShop - ZRC20 System Contract



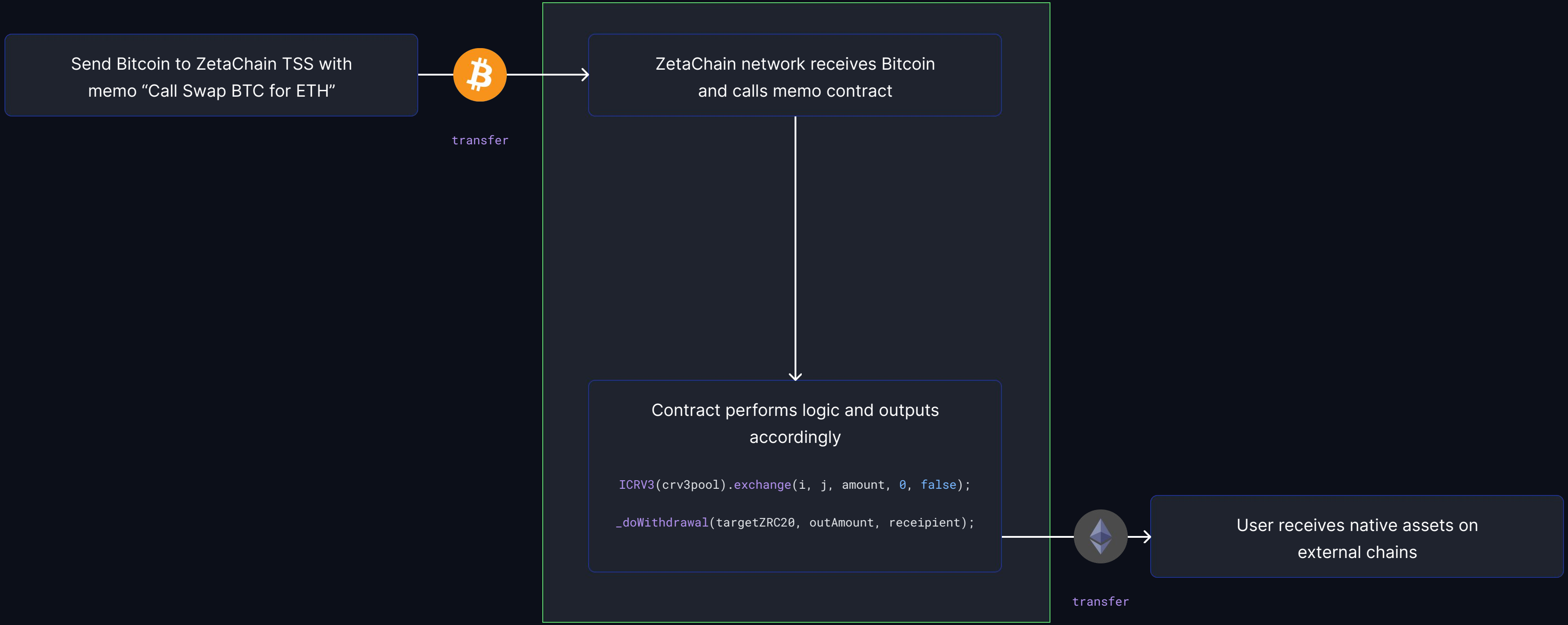
```
contract SystemContract {
address public constant FUNGIBLE_MODULE_ADDRESS;
// ...
constructor(address fungibleModule) {
    FUNGIBLE_MODULE_ADDRESS = fungibleModule;
}
// ...
function DepositAndCall(address zrc20, uint256 amount, address target, bytes calldata message) external {
    require(msg.sender == FUNGIBLE_MODULE_ADDRESS);
    require(target != FUNGIBLE_MODULE_ADDRESS && target != address(this));
    IZRC20(zrc20).deposit(target, amount);
    zContract(target).onCrossChainCall(zrc20, amount, message);
}       You, now • Uncommitted changes
```

Checkpoint - ZRC20 Wrap Up

How could you use ZRC20?

Questions?

Example: Omnichain Swap Visualized


Send Bitcoin to ZetaChain TSS with memo “Call Swap BTC for ETH”
transfer
ZetaChain network receives Bitcoin and calls memo contract
Contract performs logic and outputs accordingly
transfer
User receives native assets on external chains

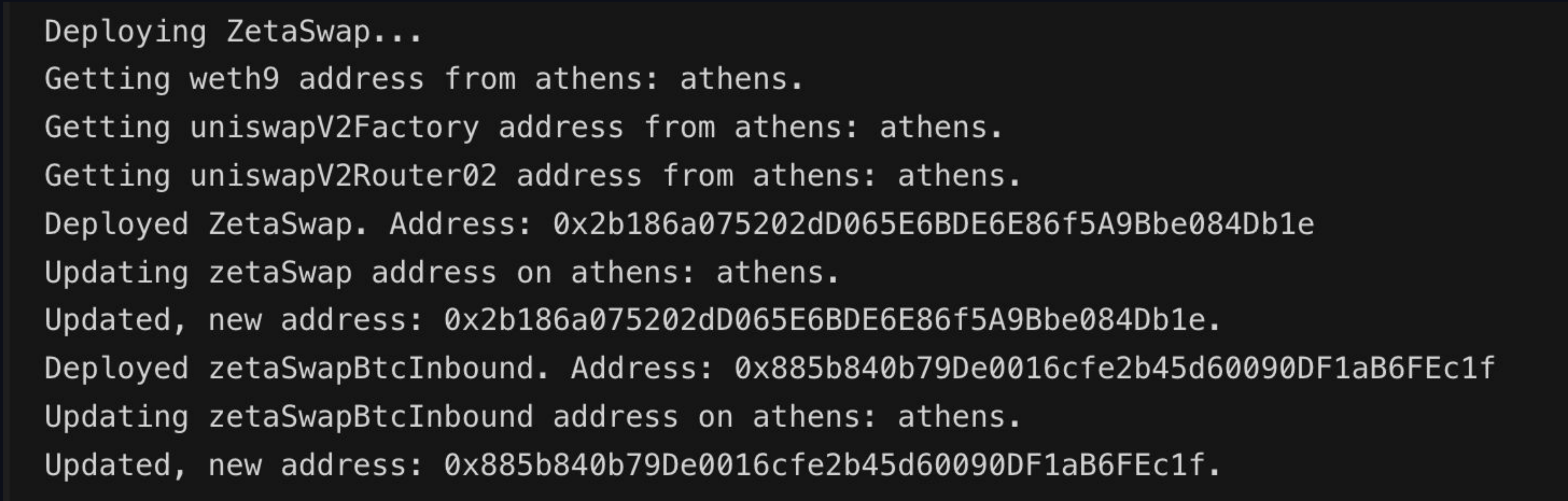
```
ICRV3(crv3pool).exchange(i, j, amount, 0, false);

_doWithdrawal(targetZRC20, outAmount, receipient);
```

WorkShop - Omnichain Swap

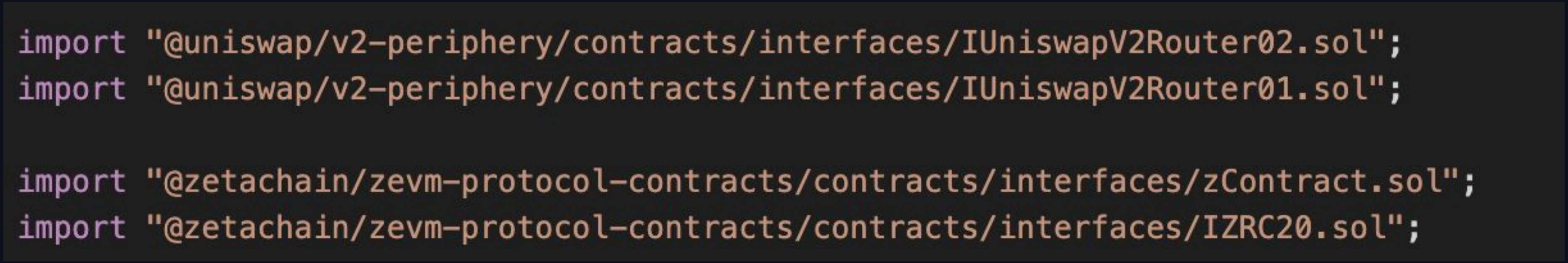
```
cd packages/zevm-example-contracts/
```

```
npx hardhat run scripts/zeta-swap/deploy.ts --network athens
```



```
Deploying ZetaSwap...
Getting weth9 address from athens: athens.
Getting uniswapV2Factory address from athens: athens.
Getting uniswapV2Router02 address from athens: athens.
Deployed ZetaSwap. Address: 0x2b186a075202dD065E6BDE6E86f5A9Bbe084Db1e
Updating zetaSwap address on athens: athens.
Updated, new address: 0x2b186a075202dD065E6BDE6E86f5A9Bbe084Db1e.
Deployed zetaSwapBtcInbound. Address: 0x885b840b79De0016cfe2b45d60090DF1aB6FEc1f
Updating zetaSwapBtcInbound address on athens: athens.
Updated, new address: 0x885b840b79De0016cfe2b45d60090DF1aB6FEc1f.
```

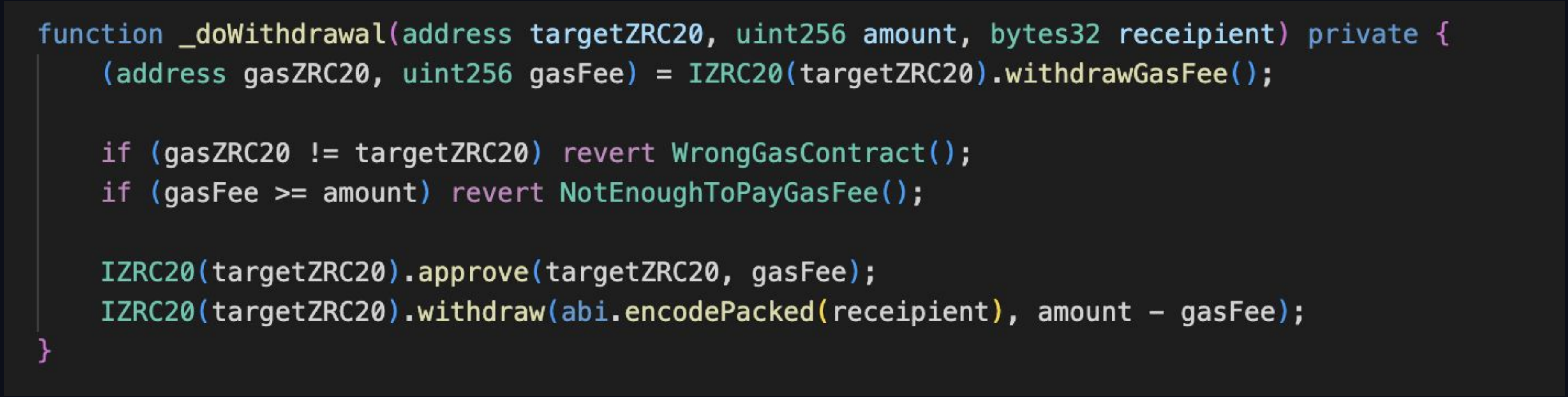
WorkShop - Swap Closer Look



```solidity
import "@uniswap/v2-periphery/contracts/interfaces/IUniswapV2Router02.sol";
import "@uniswap/v2-periphery/contracts/interfaces/IUniswapV2Router01.sol";

import "@zetachain/zevm-protocol-contracts/contracts/interfaces/zContract.sol";
import "@zetachain/zevm-protocol-contracts/contracts/interfaces/IZRC20.sol";
```

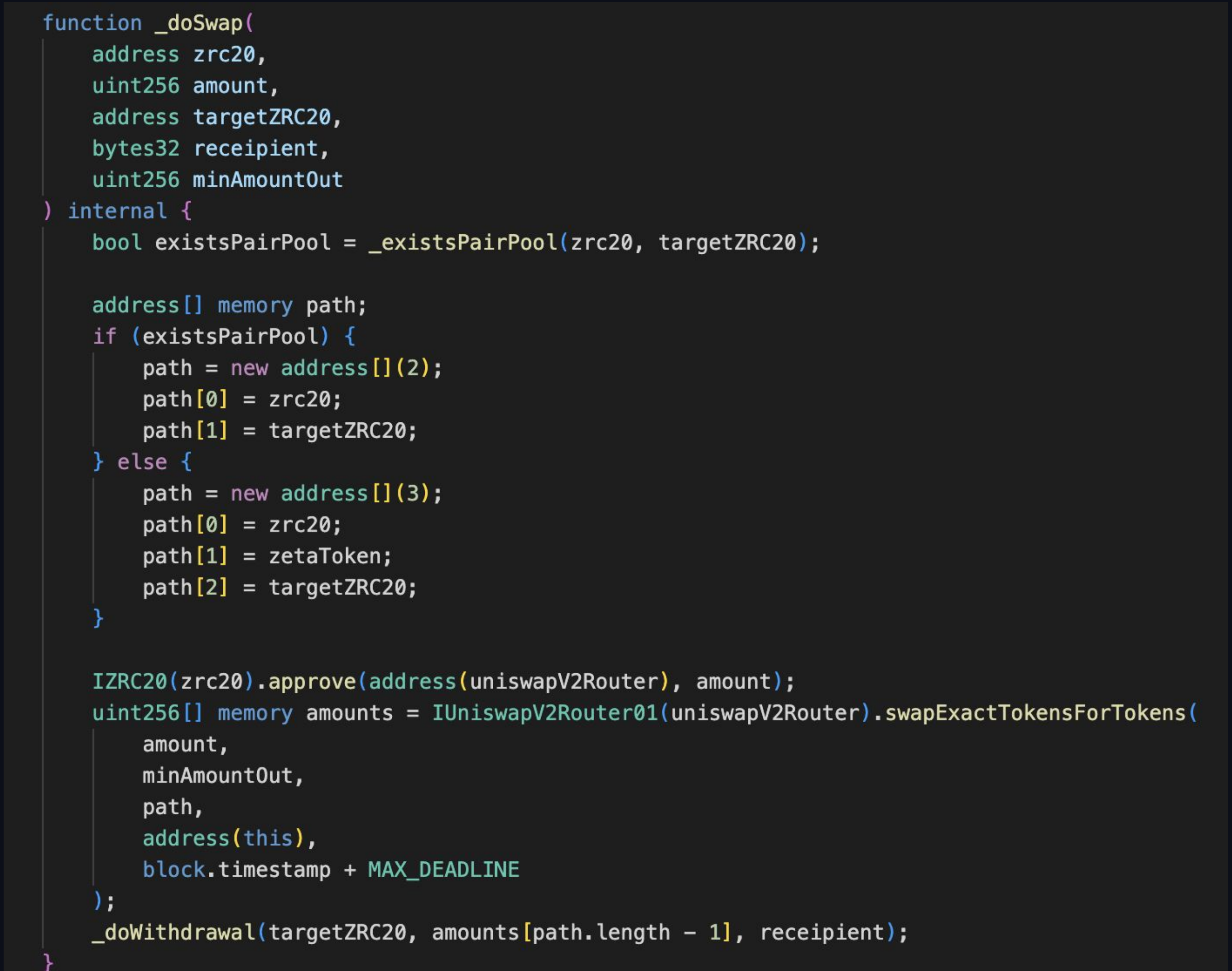
WorkShop - Swap Closer Look



```solidity
function _doWithdrawal(address targetZRC20, uint256 amount, bytes32 receipient) private {
    (address gasZRC20, uint256 gasFee) = IZRC20(targetZRC20).withdrawGasFee();

    if (gasZRC20 != targetZRC20) revert WrongGasContract();
    if (gasFee >= amount) revert NotEnoughToPayGasFee();

    IZRC20(targetZRC20).approve(targetZRC20, gasFee);
    IZRC20(targetZRC20).withdraw(abi.encodePacked(receipient), amount - gasFee);
}
```



```
function _doSwap(
    address zrc20,
    uint256 amount,
    address targetZRC20,
    bytes32 receipient,
    uint256 minAmountOut
) internal {
    bool existsPairPool = _existsPairPool(zrc20, targetZRC20);

    address[] memory path;
    if (existsPairPool) {
        path = new address[](2);
        path[0] = zrc20;
        path[1] = targetZRC20;
    } else {
        path = new address[](3);
        path[0] = zrc20;
        path[1] = zetaToken;
        path[2] = targetZRC20;
    }

    IZRC20(zrc20).approve(address(uniswapV2Router), amount);
    uint256[] memory amounts = IUniswapV2Router01(uniswapV2Router).swapExactTokensForTokens(
        amount,
        minAmountOut,
        path,
        address(this),
        block.timestamp + MAX_DEADLINE
    );
    _doWithdrawal(targetZRC20, amounts[path.length - 1], receipient);
}
```

@charliemc0

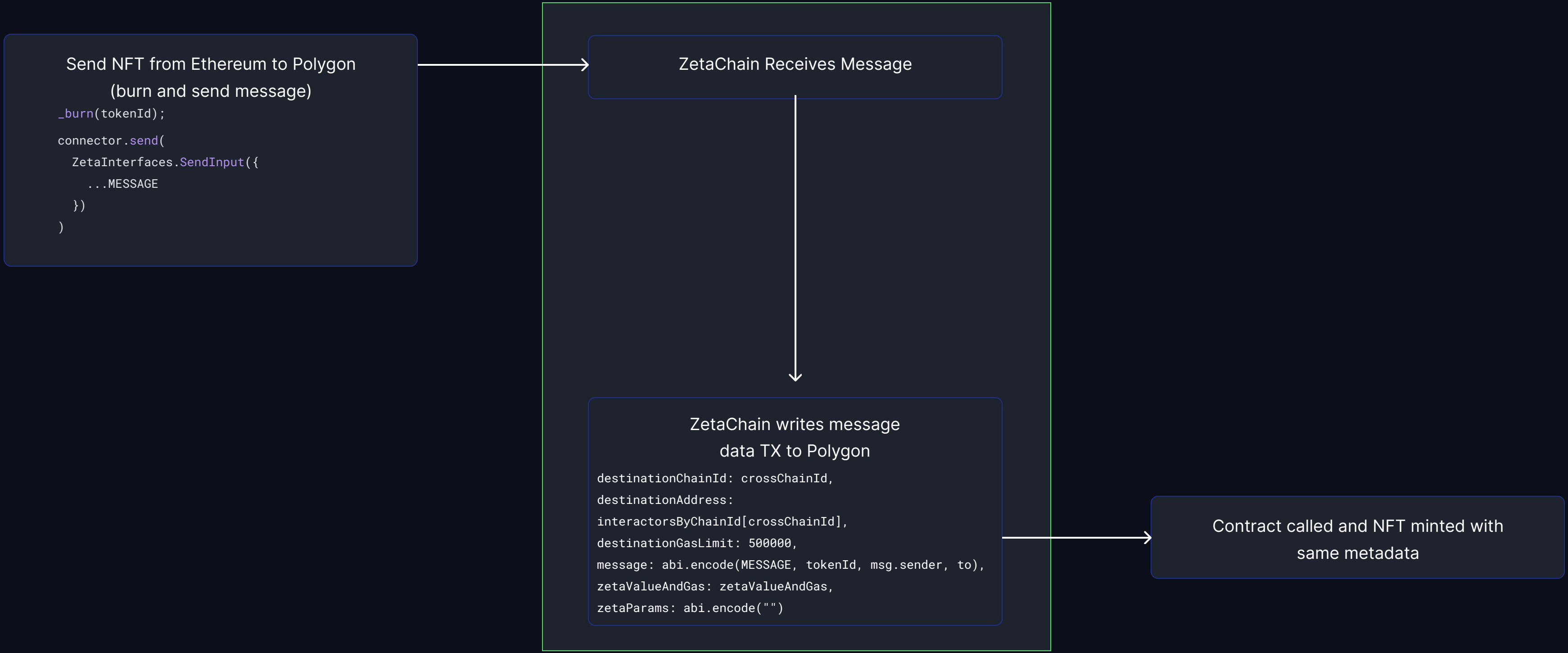
Checkpoint - Swap Deployment

Did you get the swap contract deployed?

Questions?

Next Up - Explaining ZRC20

Example: Omnichain NFT Visualized



Send NFT from Ethereum to Polygon
(burn and send message)

```
_burn(tokenId);

connector.send(
  ZetaInterfaces.SendInput({
    ...MESSAGE
  })
)
```

ZetaChain Receives Message

ZetaChain writes message
data TX to Polygon

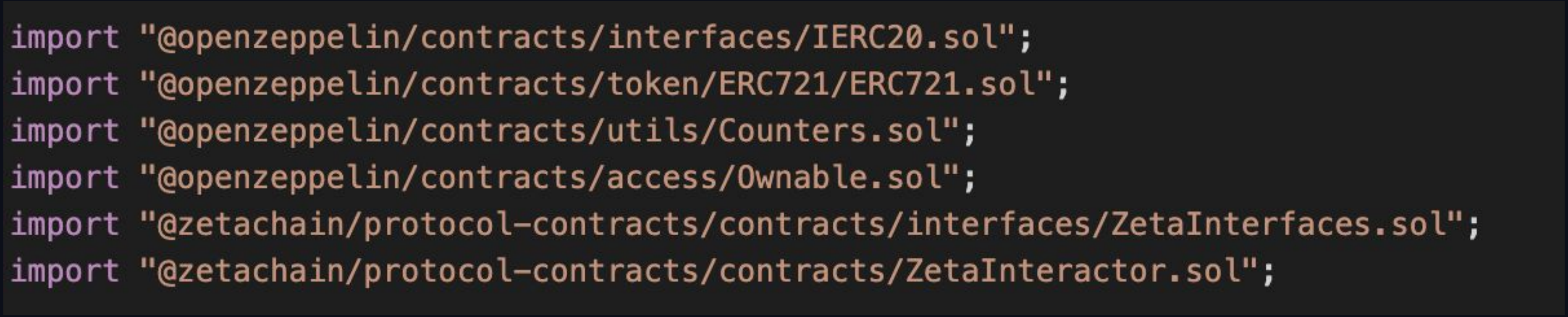
```
destinationChainId: crossChainId,
destinationAddress:
interactorsByChainId[crossChainId],
destinationGasLimit: 500000,
message: abi.encode(MESSAGE, tokenId, msg.sender, to),
zetaValueAndGas: zetaValueAndGas,
zetaParams: abi.encode("")
```

Contract called and NFT minted with
same metadata

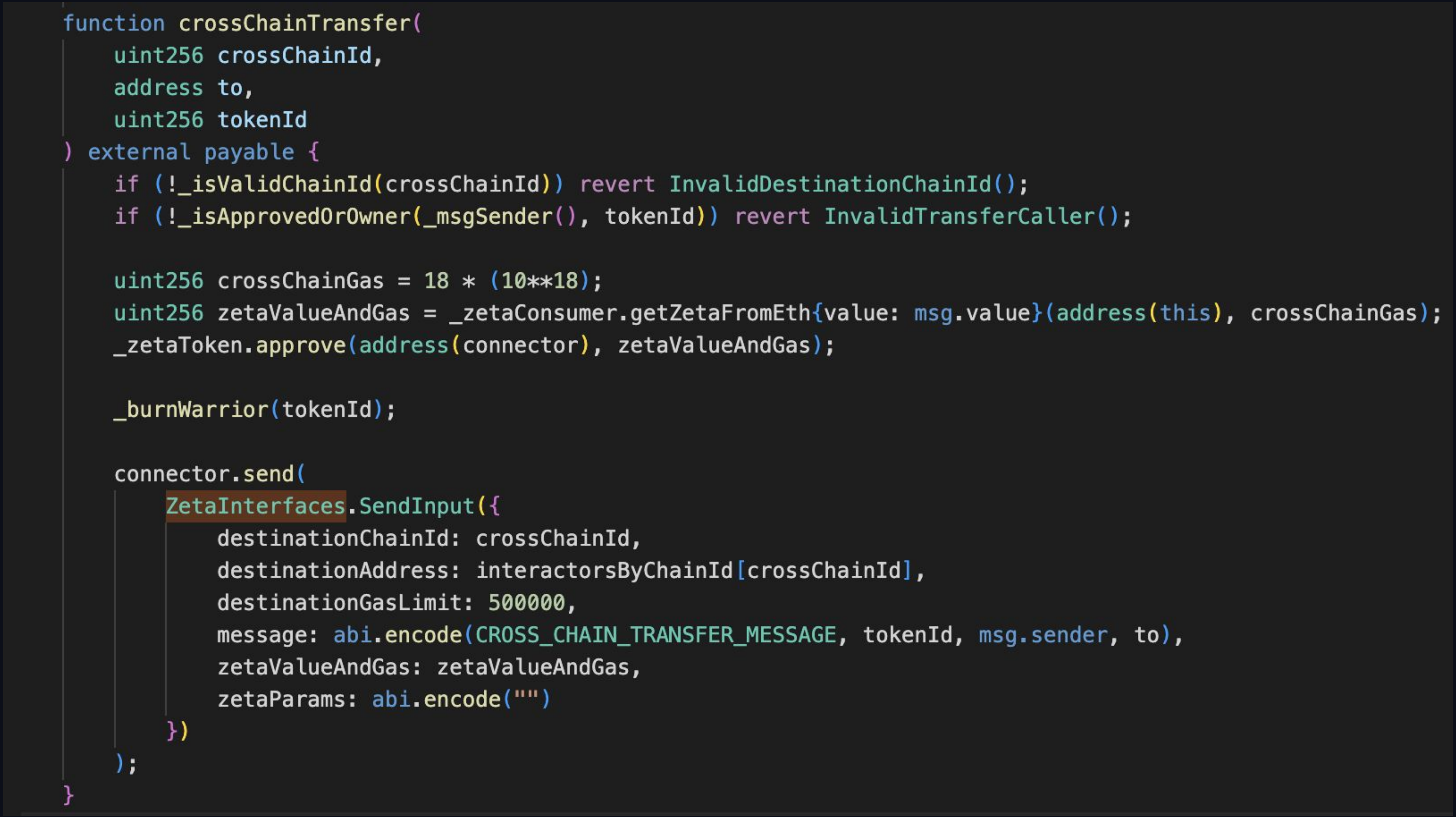
WorkShop - NFT Contract

packages/example-contracts/contracts/cross-chain-warriors/CrossChainWarriors.sol.

If you’re building on top of this, you can edit the file and redeploy using the same steps



```solidity
import "@openzeppelin/contracts/interfaces/IERC20.sol";
import "@openzeppelin/contracts/token/ERC721/ERC721.sol";
import "@openzeppelin/contracts/utils/Counters.sol";
import "@openzeppelin/contracts/access/Ownable.sol";
import "@zetachain/protocol-contracts/contracts/interfaces/ZetaInterfaces.sol";
import "@zetachain/protocol-contracts/contracts/ZetaInteractor.sol";
```



```
function crossChainTransfer(
    uint256 crossChainId,
    address to,
    uint256 tokenId
) external payable {
    if (!_isValidChainId(crossChainId)) revert InvalidDestinationChainId();
    if (!_isApprovedOrOwner(_msgSender(), tokenId)) revert InvalidTransferCaller();

    uint256 crossChainGas = 18 * (10**18);
    uint256 zetaValueAndGas = _zetaConsumer.getZetaFromEth{value: msg.value}(address(this), crossChainGas);
    _zetaToken.approve(address(connector), zetaValueAndGas);

    _burnWarrior(tokenId);

    connector.send(
        ZetaInterfaces.SendInput({
            destinationChainId: crossChainId,
            destinationAddress: interactorsByChainId[crossChainId],
            destinationGasLimit: 500000,
            message: abi.encode(CROSS_CHAIN_TRANSFER_MESSAGE, tokenId, msg.sender, to),
            zetaValueAndGas: zetaValueAndGas,
            zetaParams: abi.encode("")
        })
    );
}
```

WorkShop - Deploy NFT Contracts

```
cd packages/example-contracts/

npx hardhat run scripts/cross-chain-warriors/deploy.ts --network goerli

npx hardhat run scripts/cross-chain-warriors/deploy.ts --network bsc-testnet

npx hardhat run scripts/cross-chain-warriors/set-cross-chain-data.ts --network
goerli

npx hardhat run scripts/cross-chain-warriors/set-cross-chain-data.ts --network
bsc-testnet
```

- Retrieves the contract addresses
- Call the method crossChainWarriorsContract.setInteractorByChainId with two parameters
  - Destination chain
  - Address of the contract on that chain.

Checkpoint - NFT Contract

What did you do?

Next steps? - How to take this further

Idea for cross-chain NFT projects?

Create an Interoperable dApp

Omnichain Contracts

- Write in Solidity, deploy to Zetachain
- Create new contracts or update existing contracts with minimal changes
- Low gas fees
- Recommended for all new projects
- Recommended for existing projects that want ultimate interoperability

Make an Existing dApp Interoperable

Cross Chain Message Passing

- Minimal code changes to make your dApp deployed on Ethereum interoperable with other chains
- Existing contracts on Ethereum orchestrate assets on other connected chains, including Bitcoin, Polygon, etc.
- Useful for existing contracts/Dapps
- Only recommended when making a minimal amount of changes to an existing dApp

Bonus Example: Omnichain Curve App

https://www.zetachain.com/docs/developers/Omnichain-smart-contracts/examples/curve-sample-on-zevm/

- Deploy Curve Contract
- Setup TriPool using ZRC assets

Bonus Example: Omnichain Curve App

```solidity
function onCrossChainCall(
    address zrc20,
    uint256 amount,
    bytes calldata message
) external override {
        (address targetZRC20, bytes32 receipient, ) = abi.decode(message,
(address, bytes32, uint256));

    address[] memory path = new address[](2);
    path[0] = zrc20;
    path[1] = targetZRC20;
    IZRC20(zrc20).approve(address(crv3pool), amount);

    uint256 i = addr2idx(zrc20);
    uint256 j = addr2idx(targetZRC20);
    require(i >= 0 && i < 3 && j >= 0 && j < 3 && i != j, "i,j error");

    uint256 outAmount = ICRV3(crv3pool).exchange(i, j, amount, 0, false);
    _doWithdrawal(targetZRC20, outAmount, receipient);
}
```

```solidity
function _doWithdrawal(
        address targetZRC20,
        uint256 amount,
        bytes32 receipient
    ) private {
        (address gasZRC20, uint256 gasFee) =
IZRC20(targetZRC20).withdrawGasFee();

        if (gasZRC20 != targetZRC20) revert WrongGasContract();
        if (gasFee >= amount) revert NotEnoughToPayGasFee();

        IZRC20(targetZRC20).approve(targetZRC20, gasFee);
        IZRC20(targetZRC20).withdraw(abi.encodePacked(receipient),
amount - gasFee);
    }
```

Bonus Example: Omnichain Uniswap App

```
    // Called from external chain deposits
    function onCrossChainCall(address zrc20, uint256 amount, bytes calldata message) external override {
        address targetZRC20;
        address receipient;
        uint256 minAmountOut;
        (targetZRC20, recipient, minAmountOut) = abi.decode(message, (address,address,uint256));
        address[] memory path;
        path = new address[](2);
        path[0] = zrc20;
        path[1] = targetZRC20;
        // Approve the usage of this token by router02
        IZRC20(zrc20).approve(address(router02), amount);
        // Swap for your target token
        uint256[] memory amounts = IUniswapV2Router01(router02).swapExactTokensForTokens(amount, minAmountOut, path,
address(this), block.timestamp);
        // Withdraw amountto target recipient
        IZRC20(targetZRC20).withdraw(abi.encodePacked(recipient), amounts[1]);
    }
```

Want to build on ZetaChain?

- We’re providing resources, technical support, and more to partners interested in building the future of crypto with us.
- Grant program - get in touch with brandon@zetachain.com

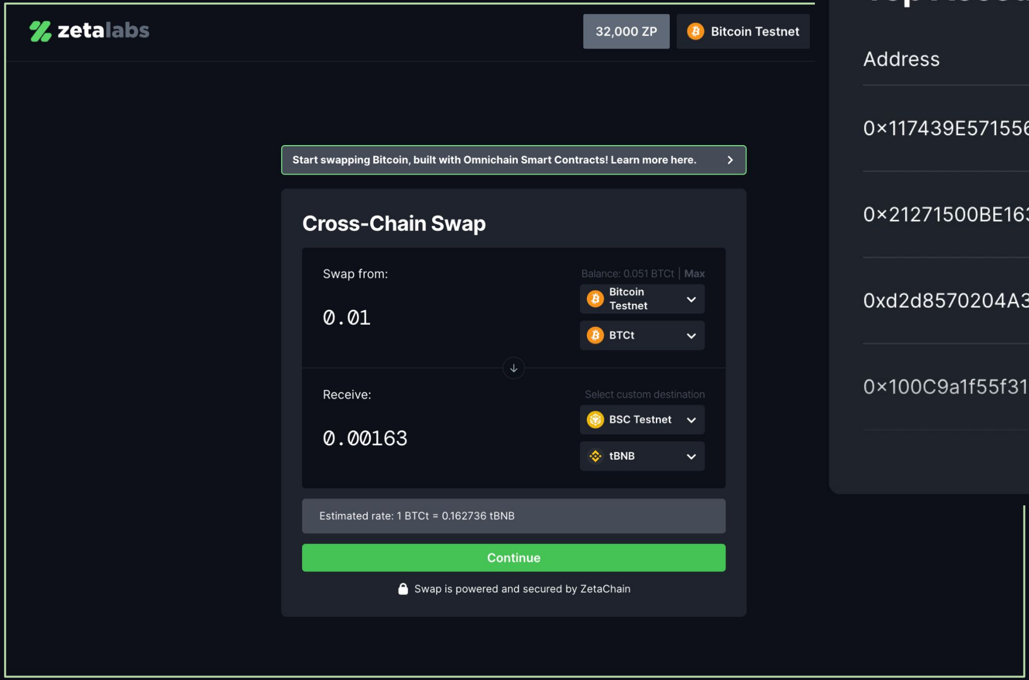
Try it out today

- We are continually adding usable contract templates and tutorials to our docs!
- Easily deploy in a matter of minutes: Omnichain NFTs, Uniswap or Curve swaps, Wrapless Value Transfer.
- Extend existing EVM contracts to deploy on zEVM or using messaging
- Examples and docs:
  - **zetachain.com/docs**
  - **github.com/zeta-chain/zetachain**

Hackathon Tracks

- Vertical 1: DeFi (Decentralized Finance)
- Vertical 2: UX (User Experience)
- Vertical 3: NFTs (Non-Fungible Tokens)
- Vertical 4: DAOs (Decentralized Autonomous Organizations)
- Vertical 5: Identity, Social, and Wallets

Testnet App (labs.zetachain.com)



zetalabs

32,000 ZP | Bitcoin Testnet

Start swapping Bitcoin, built with Omnichain Smart Contracts! Learn more here.

Cross-Chain Swap

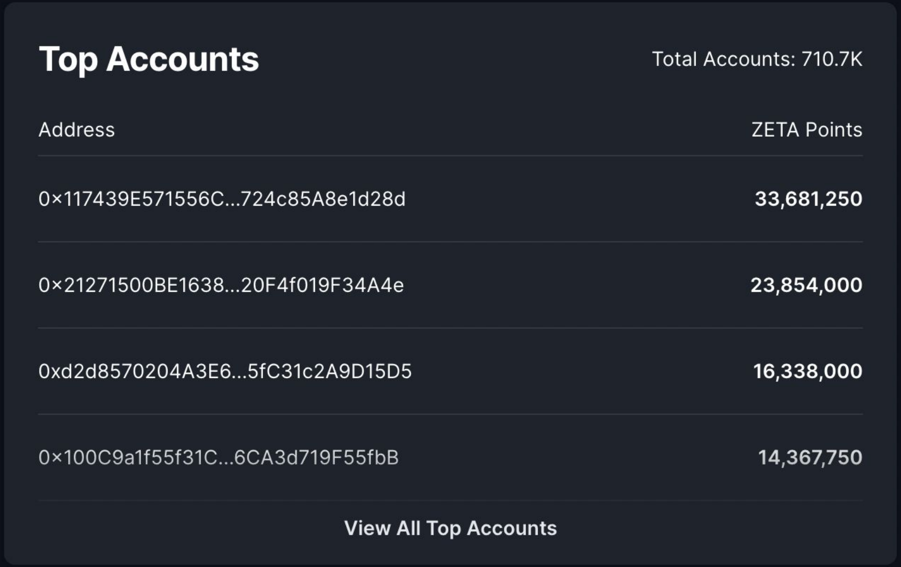
Swap from: 0.01 — Balance: 0.051 BTCt | Max — Bitcoin Testnet — BTCt

Receive: 0.00163 — Select custom destination — BSC Testnet — tBNB

Estimated rate: 1 BTCt = 0.162736 tBNB

Continue

Swap is powered and secured by ZetaChain



Top Accounts

Total Accounts: 710.7K

Address	ZETA Points
0×117439E571556C...724c85A8e1d28d	**33,681,250**
0×21271500BE1638...20F4f019F34A4e	**23,854,000**
0xd2d8570204A3E6...5fC31c2A9D15D5	**16,338,000**
0×100C9a1f55f31C...6CA3d719F55fbB	**14,367,750**

View All Top Accounts

Testnet Stats

8m+
Cross-Chain Transactions

700k+
Testnet Users

250+
dApps deployed

109m+
Internal Transactions

ZetaChain Roadmap

Q3-Q4 '21

Devnet launch with cross-chain value transfer app

- Blockchain development

Q1-Q2 '22

Testnet v1 launch

- Whitepaper release
- Omnichain Messaging API and Examples

Current

Q2-Q3 '23

Incentivized Testnet

- Omnichain Smart Contract support
- Bitcoin Support
- Extensive security audits

Q3-Q4 '23

Mainnet launch

- Omnichain DEX launch with existing CEX and wallet
- DeFi, NFT, and DAO dApp launches with dev ecosystem

Beyond

Omnichain ZETA governance

- Dev ecosystem growth and grant program
- PoS network and scalability improvement
- ... and much more

Thank you

Contact:	**brandon@zetachain.com**
Stay tuned at:	**https://zetachain.com**
	@zetablockchain
	https://discord.gg/zetachain

Learn more here

Appendix

Want to build on ZetaChain?

- We’re providing resources, technical support, and more to partners interested in building the future of crypto with us.
- Grant program - get in touch with brandon@zetachain.com

Scalable for real usage

5 seconds
Block time

>4000 tps*
Native transactions per second

<$0.01
Avg. ZetaChain TX fees (not incl. external tx fees)

>100 validators
Validator network scalability

*under favorable conditions