

EVM and the XRP Ledger

Introduction



HazardCookie

Developer Advocate, Ripple

Twitter: @hazardcookie

Github: hazardcookie



<https://linktr.ee/hazardcookie>

XRP Ledger (XRPL) launched in 2012 to address limitations of crypto and fiat currencies for financial use cases, specifically payments



Over a decade, XRPL has matured into one of the most robust layer-1 blockchains

100%

decentralized blockchain with 600+ nodes processing transactions and maintaining the ledger

1750+

unique apps and exchanges on mainnet built by a diverse set of global developers

4.5M+

active XRP wallet holders around the world

100+

Proof-of-Association validators operated by universities, exchanges, businesses, & individuals

2.6B+

transactions processed representing over \$1T in value moved between counterparties

\$35B+

market capitalization of XRP, making it the ~5th largest cryptocurrency

Intentionally designed to enable scalable blockchain development



Proven

Supports large scale use cases and long term projects with **2.6B+ successful transactions, more than Ethereum**, without failure or security breach since 2012



Ready to Go

Access complete blockchain functionality, from tokenizing assets to advanced payments, **without needing to learn, build, and maintain** complex smart contracts



Fast, Cheap, Green

Carbon neutral blockchain settling transactions every **3-5 seconds** at fractions of a cent per transaction for mass market adoption

Powerful protocol features provide the building blocks to innovate

Native DEX

First on-chain DEX in the world, trading and moving tokens anywhere in seconds with competitive liquidity

Issued Assets

Ability to represent digital currencies, legal obligations, fungible tokens, and other asset classes on the ledger

Non-Fungible Tokens

Non-fungible tokens with built-in royalties and all trades are handled by the DEX

Token Asset Controls

Controls for token issuers and holders to enhance security and regulatory compliance

Advanced Payments

Use advanced payment capabilities like “Escrow” and “Checks” to build smart applications without smart contracts

EVM Sidechain



Join the XRPL EVM Discord!



Why EVM Sidechain?

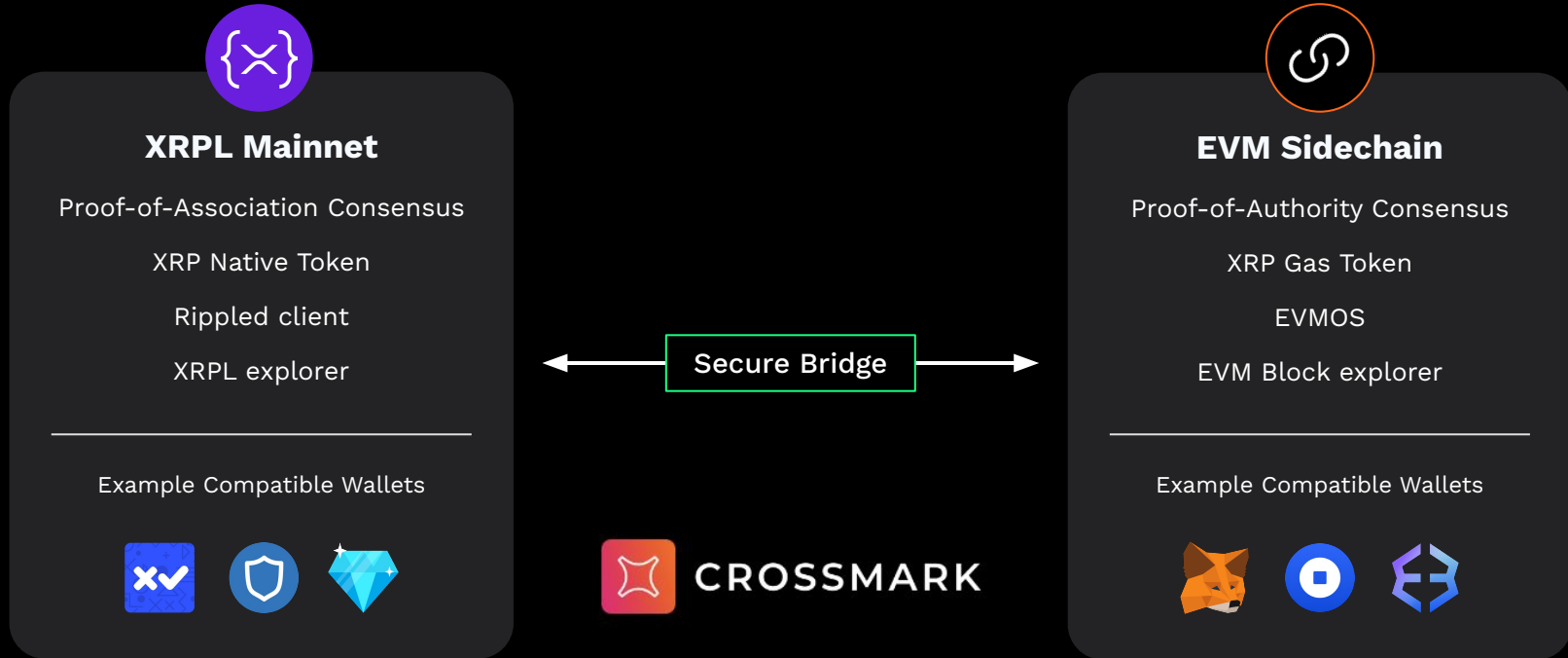
- Lack of general purpose smart contract support
- Target a larger number of EVM ecosystem developers
- Demand from institutions and central banks for evm focused applications

EVM Compatibility on different blockchains

Blockchain Ecosystem	EVM Compatibility Solution/Project
Ethereum	Native
Solana	Neon
Polkadot	Moonbeam
Cosmos	Evmos
Polygon	zkEVM
BNB Chain	BNB Smart chain
Avalanche	Avalanche C-chain
XRPL	EVM Sidechain

What is it?

The EVM Sidechain enables the ability to interact or deploy smart contracts written in Solidity with a secure bridge to XRPL Mainnet



Why is that interesting?

EVM apps can now access and benefit from the XRPL ecosystem

01

Bridge to the XRPL ecosystem

Any Solidity app written for Ethereum / EVM can access liquidity and user base of XRPL Mainnet

02

Optimized for DeFi

Secure bridges, enhanced scalability and fast transaction finality makes the EVM optimized for financial use cases, like DeFi and payments

03

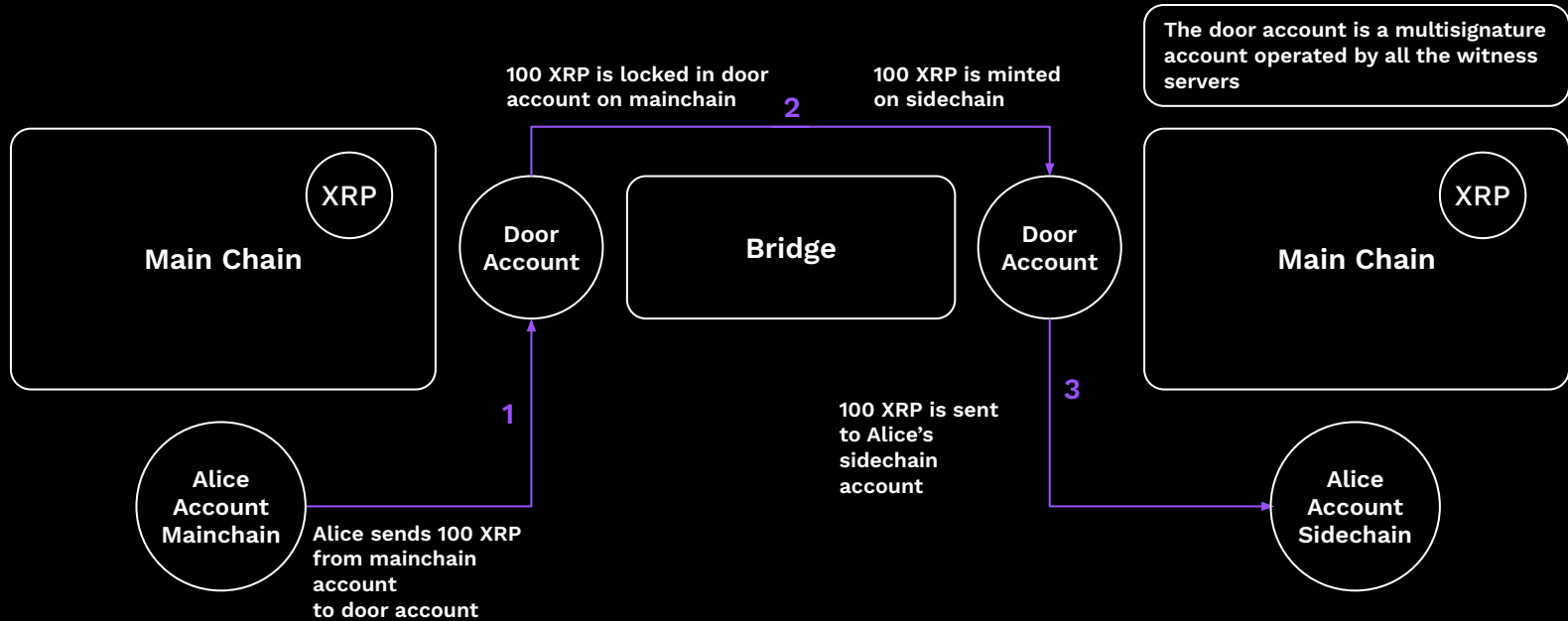
Easy to Build

Build using familiar Ethereum-based tools, wallets, explorers, and apps like MetaMask, Foundry, and Truffle

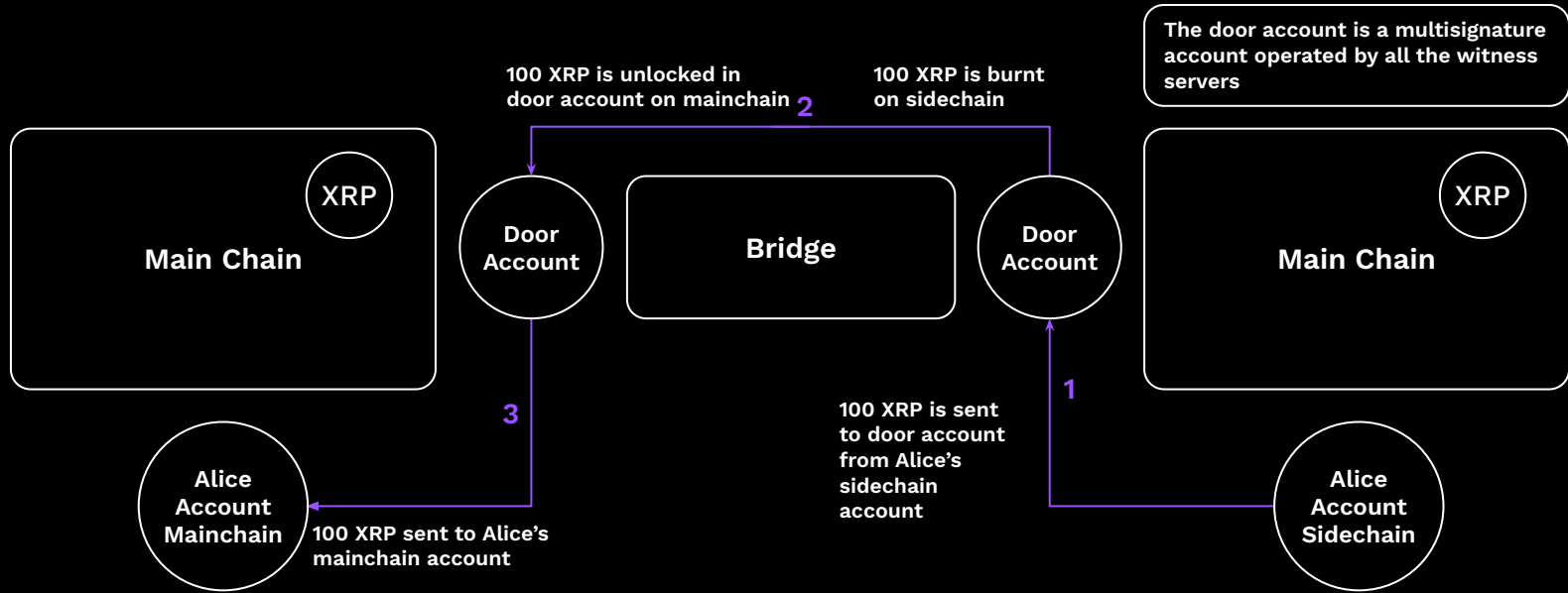
Network Details

- Consensus
 - Powered by CometBFT
 - Tendermint fork
 - Byzantine fault tolerance engine
 - 1 block finalization time
- Execution
 - Cosmos SDK
 - EVMOS
- Validators
 - POA Cosmos SDK

Sidechains - Flow of Funds Mainchain -> Sidechain



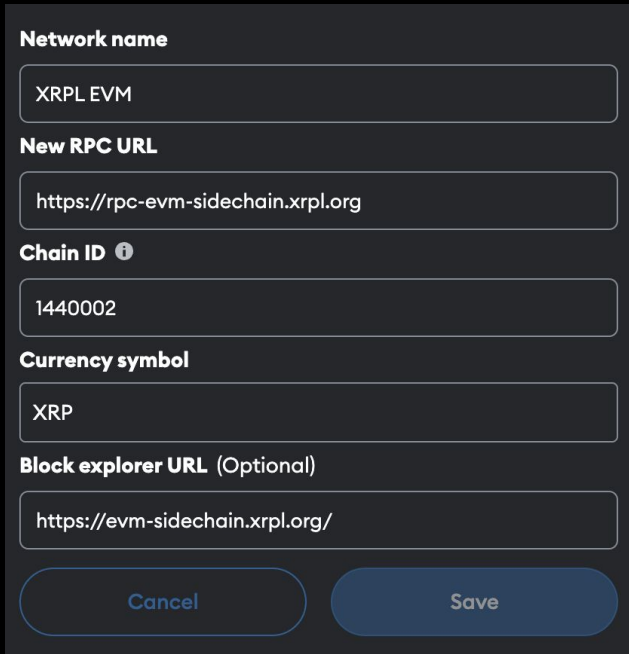
Sidechains - Flow of Funds Sidechain -> Mainchain



Setting up Metamask

Add a custom network using the details below:

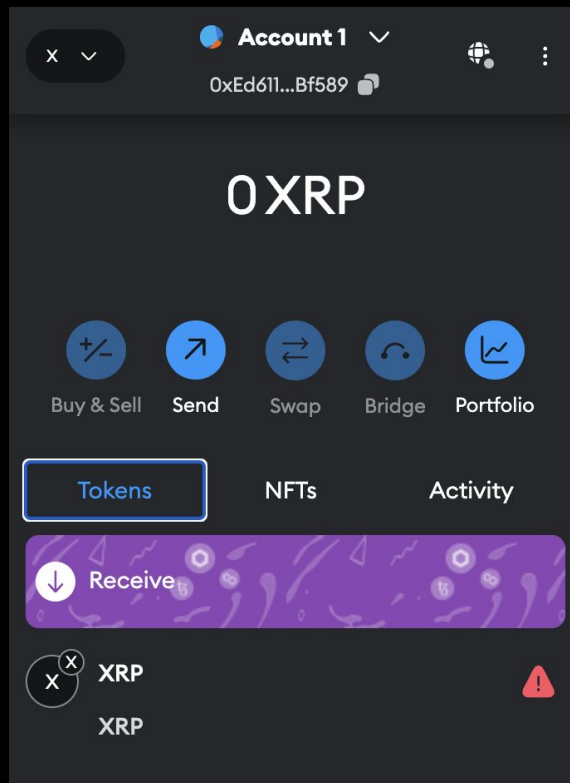
- **Network Name** : XRPL EVM Sidechain
- **New RPC URL** : <https://rpc-evm-sidechain.xrpl.org>
- **Chain ID** : 1440002
- **Currency Symbol** : XRP
- **Block Explorer** : <https://evm-sidechain.xrpl.org>



The image shows a screenshot of the Metamask interface for adding a custom network. The form is titled 'Add Custom Network' and contains several input fields and buttons. The fields are: 'Network name' with the value 'XRPL EVM', 'New RPC URL' with the value 'https://rpc-evm-sidechain.xrpl.org', 'Chain ID' with the value '1440002', 'Currency symbol' with the value 'XRP', and 'Block explorer URL (Optional)' with the value 'https://evm-sidechain.xrpl.org/'. At the bottom of the form are two buttons: 'Cancel' and 'Save'.

Field	Value
Network name	XRPL EVM
New RPC URL	https://rpc-evm-sidechain.xrpl.org
Chain ID	1440002
Currency symbol	XRP
Block explorer URL (Optional)	https://evm-sidechain.xrpl.org/

Should look like this afterwards



Bridge over some XRP

<https://bridge.devnet.xrpl.org>

From

Network

XRPL Devnet

Wallet

r97r9tqsaY...rbKR6XTT

To

Network

EVM Sidechain Devnet

Wallet

0xED611b2a...DB0Bf589

You send

Amount

9990

Token

XRP

You receive

Amount

9990

XRP

Max: 9,990 XRP

Bridge Transfer Fee ~ 0 XRP

Estimated time of arrival ~ 3 seconds - 3 minutes

Transfer

Account 1

0xED611...Bf589

1000 XRP

Buy & Sell

Send

Swap

Bridge

Portfolio

Tokens

NFTs

Activity

XRP

XRP

Create your own ERC20 token

```
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.19;
3
4 import "https://github.com/OpenZeppelin/openzeppelin-contracts/blob/v4.9.0/contracts/token/ERC20/ERC20.sol";
5
6 contract CornellToken is ERC20 {
7     constructor() ERC20("CornellToken", "Cornell") { ⛽ infinite gas 931000 gas
8         _mint(msg.sender, 420000 * 10 ** decimals());
9     }
10 }
```

Compile and deploy in Remix

Use “Injected Provider” to compile and deploy

SOLIDITY COMPILER

COMPILER +

0.8.19+commit.7dd6d404

☐ Include nightly builds

☐ Auto compile

☐ Hide warnings

Advanced Configurations >

Compile CyprusTest.sol

Compile and Run script

DEPLOY & RUN TRANSACTIONS

ENVIRONMENT

Remix VM (Shanghai)

Injected Provider - MetaMask

Remix VM (Shanghai)

Remix VM (Merge)

Remix VM (London)

Remix VM (Berlin)

Home CornellToken.sol X

```
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.19;
3
4 import "https://github.com/OpenZeppelin/open
5
6 contract CornellToken is ERC20 {
7     constructor() ERC20("CornellToken", "Cor
8         _mint(msg.sender, 420000 * 10 ** dec
9     }
10 }
11
12
13
```

View your token on the EVM explorer

Details

Token transfers

Internal txns

Logs

State

Raw trace

This is a testnet transaction only

Transaction hash

0x9a074c35adbb450dbf83b64829c814c019bd2eb2ff26141f859ef580dae480c

Status and method

Success

Block

7208852 | 7 Block confirmations

Timestamp

29s ago | Mar 23 2024 03:01:51 AM (-04:00 UTC) | Confirmed within <= 3,758 secs

From

0xEd611b2a0ccf0ef8Aa04EEEFDE9657e8DB08f589

To

[Contract CornellToken created]

Tokens minted

0x00...0000 → 0xEd...f589 for 420,000 Cornell

Value

0 XRP (\$0)

Transaction fee

0.001766964008245832 XRP (\$0.0010799684018398525184)

Gas price

0.000000001500000007 XRP (1.5000000007 Gwei)

Gas usage & limit by txn

1,177,976 | 1,177,976 100%

Gas fees (Gwei)

Base: 0.000000007 | Max: 1.500000008 | Max priority: 1.5

Burnt fees

0.000000000008245832 XRP (\$0.0000000000050398525184)

[View details](#)



EVM Sidechain Starter Repo



<https://github.com/hazardcookie/cornell-evm>