Full integration of EVM in Discord

Alpha document

Contents

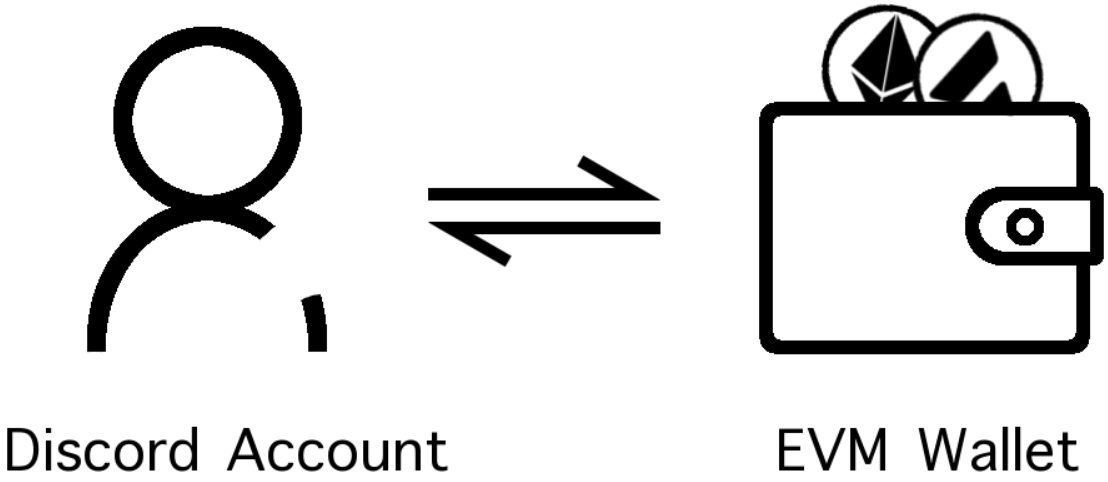
1 Fully integrated Discord Wallet **2**
1.1 Transactions 3
1.2 Services 3

2 Infrastructure insight **5**

3 Security **5**

1 Fully integrated Discord Wallet

We once hoped that discord developers would implement any form of web3 integration in their native app. They considered it but facing the unexplained, though categorical, refusal from g@m3rs, they had to back off. However they left us quite enough tools to build our own.



Using a dedicated wallet linked to your discord account, we provide this integration. You own your wallet and only your Discord Account can use it through our application. You are the sole owner of your private key. Using password encryption we **never store nor read your private key**. To ensure that we never hold plain text private keys, you must create your wallet, encrypt your private key with your password then communicate it along with your hashed password (we can provide a NodeJS script to make sure that you encrypt your private key and hash your password using the right algorithms). We might implement a way to do it directly in the application in production but it would never be as secured as if you do it locally.

Once created and integrated into our database, you will be able to log into our service using Discord most recent feature. Once logged in, you can use your wallet to execute any of the following implemented transactions and contract interactions until you decide to log off (or after some preset timer), leaving us with no access to your wallet.

As any other discord based service, you can use it seemlessly on your phone, your computer or your web browser, as long as you are connected to your discord account. Remember that you hold your private key so you can also use the same exact wallet on any other wallet application.

1.1 Transactions

The whole point for us to have your wallet fully integrated into our service is to be able to provide you with every functionalities that we see fit. By having full control upon those functionality, we can either update them or create new ones. This allows us to adapt our strategies as scammer strategies also evolve, for example, classic honeypot detection is sometime bypassed by scammer contracts but as soon as we identify the bypass we can add additional threat checks. As a user, you can report it and participate in the script adaptation or benefit from other user reports. Here's a list of implemented and envisioned functionalities :

- **Transfer**: classic transfer to any address and easy transfer to any other member of the discord server, as long as they made their address public for server members.
- **Swap**: send a swap transaction on most popular DEXs, on implemented EVM chains. Used to swap between ERC20 or ETH equivalents that we trust.
- **Liquidity Snipe**: initiate addLiquidity transaction monitoring in the mempool, automatically call SafeSwap when it happens.
- **SafeSwap**: send a swap transaction as with the Swap functionality, checks that you would be able to swap it back for untrusted contracts, scan contracts for known "NO GO".
- **Mempool Monitoring**: gives you the ability to be alerted when it detects that a predetermined transaction is soon to be validated. Add an extra layer against rug pulls for example.
- **ERC20 Listing Monitoring**: classic ERC20 listing monitoring on every implemented EVM chains.

Functionalities such as Minting, Minting Snipe, Bridge Integration or even Basic Front Run are envisioned to be implemented in the future. We will consider user needs.

1.2 Services

Once we have those basic functionalities we can easily provide many services. Some of them are just the execution of previously described transactions, others combine them to achieve more. For example, as we keep monitoring every blockchain we implement discord integration for, we can easily provide basic alerts like free mints or others standardized and custom event. Depending on users needs, we can adapt and help you get closer to on-chain interactions. For the first iteration, services that will be included are the following :

- **On chart DEX trading**: Use your discord wallet to validate transaction executed by a click on a chart. Get charts from every ERC20 on every DEX that you need, place market limit orders such as market stop loss and take profit. Click in your browser and validate the exact order you needed in discord application.

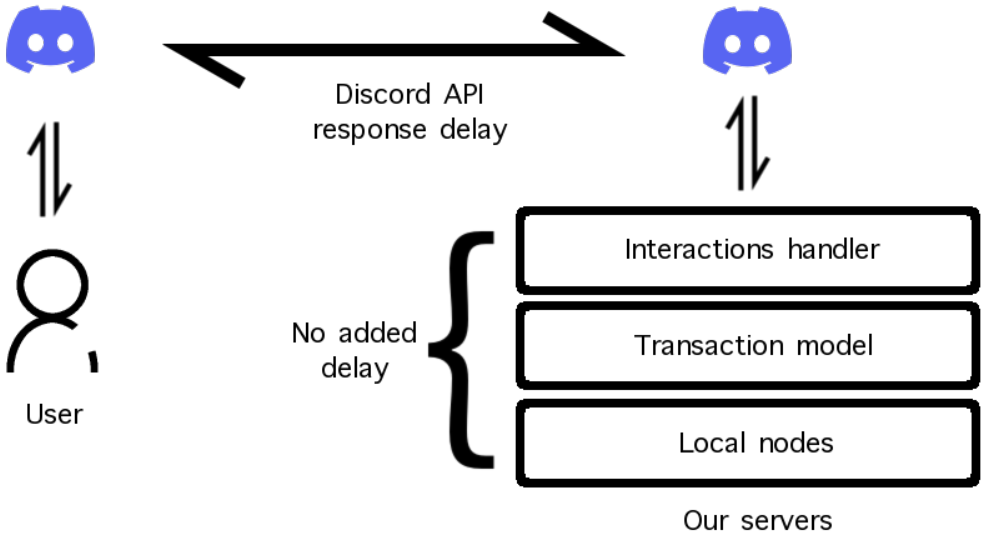
- **Safer Degen**: Use SafeSwap and monitoring to get you through most ERC20 scams. Dodge rug pulls, blacklisting and many other known way to pull your money.
- **Listing Cards**: ERC20 listing monitoring, SafeSwap and the discord integration allows use to provide you with listing cards. You can save, honeypot check, trade or monitor every token in a click. This card service is also available on custom ERC20 that you want.

  We want to make components of every services independent, so that you can access and understand them one by one. Some other services are just waiting for a discord integration as we already use them locally.

2 Infrastructure insight

As the functionalities we provide must be fast and reactive, the infrastructure architecture must be designed accordingly. By using Discord API, you lose overall response time. Operation like instant transfer or instant swap are slower that they would usually be with any other service. Still they are implemented so you can use them directly on discord even though they are not as efficient as they could be. However, once you initiate Liquidity Snipe or Mempool Monitoring from the discord interface, the only "slow" thing would be the notification which has to go back and forth to Discord servers before happening on your screen but transaction are initiated as soon as they arrive on our server.

With dedicated and load balanced nodes, you benefit from fast transactions and a mempool access. The limitation evoked in the previous paragraph is in fact no limitation at all when it comes to planned transactions and reaction to transactions initiated on the blockchain. For example, you decide to monitor an addLiquidity transaction for a specific ERC20 contract. You have to plan it one "Discord API response time" before the addLiquidity transaction actually occur on the chain, seems like something doable. More details about additional transaction time due to our infrastructure will come in a second paper.



3 Security

We provide an extra layer of security with password encrypted private keys. If our database were to be compromised, private keys would be safe enough by the time we activate Emergency Transfer of every assets you own to your predefined recovery address. As we wrote above, you can use your wallet with any other application, though we recommend

you keep a wallet dedicated to our application.

You can chose to activate Suspicious Activity Monitoring. By doing so, any transaction not emitted by our application would trigger an Emergency Transfer.

The password used to encrypt your key goes through Discord API which can be sniffed either on your side or our. The main security are your discord credentials. The service we provide is not attack proof but we estimate that we monitor our system enough to be reactive enough in case of such scenario. Worst case scenario would be for your back-up wallet to be compromised and an Emergency Transfer to be triggered right after that. Inform us the moment you suspect someone else to have access to your back-up wallet. This system **does not protect** you from compromised computer on your side, neither does any security system on other hot wallet application.

We care about your wallet security, we trust our system to be secured enough as we manage, monitor and connect every single piece of it. We will still investigate other convenient ways to log you on our service.

We ambition to self-host everything as this would reduce delay, boost security and make adjustments easier. We want to make it the cheapest it can be so that it can reach anyone. We want to provide the tools you need to progress in the space and that most people use. We would gladly hear suggestions. First server is soon to be opened for beta try.