How to use a Turnkey signer with permissionless.js

Turnkey is a key infrastructure provider with a great developer API and a powerful security policy engine.

By combining permissionless.js with Turnkey, you can createcustodial AA wallets whose security is provided by Turnkey, with powerful functionalities such as sponsoring gas, batching transactions, etc.

Setup

To use Turnkey with permissionless.js, first create an application that integrates with Turnkey.

- Refer to theTurnkey documentation site
- for instructions on setting up an application with the Turnkey.
- For a quick start, Turnkey provides examples, availablehere
- .

Integration

Integrating permissionless.js with Turnkey is straightforward after setting up the project. Turnkey provides an Externally Owned Account (EOA) wallet to use as a signer with permissionless.js accounts.

Create the TurnkeyClient and a Turnkey viem account

After following the Turnkey documentation, you will have access to aTurnkeyClient . An example is shown below that you can use to create aSmartAccountSigner object:

```

import{ TurnkeyClient }from"@turnkey/http" import{ createAccount }from"@turnkey/viem" import{ walletClientToSmartAccountSigner }from"permissionless" import{ createWalletClient, http }from"viem"

// Param options here will be specific to your project. See the Turnkey docs for more info. constturnkeyClient=newTurnkeyClient({ baseUrl:""}, stamper)

constturnkeyAccount=awaitcreateAccount({ client: turnkeyClient, organizationId: subOrganizationId,// Your subOrganization id signWith: signWith,// Your suborganization signWith param. })

// Create a SmartAccountSigner from the turnkeyAccount constwalletClient=createWalletClient({ account: turnkeyAccount, transport:http("https://rpc.ankr.com/eth_sepolia"), })

constsmartAccountSigner=walletClientToSmartAccountSigner(walletClient)

```

Use with permissionless.js

SimpleAccount Safe Account Kernel Account Biconomy Account ```

SimpleAccount import{ signerToSimpleSmartAccount }from"permissionless/accounts" import{ createPublicClient, http }from"viem" import{ generatePrivateKey, privateKeyToAccount }from"viem/accounts" import{ sepolia }from"viem/chains"

exportconstpublicClient=createPublicClient({ transport:http("https://rpc.ankr.com/eth_sepolia"), chain: sepolia, })

constsmartAccount=awaitsignerToSimpleSmartAccount(publicClient, { signer: smartAccountSigner, factoryAddress:"0x9406Cc6185a346906296840746125a0E44976454", entryPoint:ENTRYPOINT_ADDRESS_V06, })

```
```