D4D4D4;--ch-t-background: #1E1E1E;--ch-t-lighter-inlineBackground: #1e1e1ee6;--ch-t-editor-background: #1E1E1E;--ch-t-editor-foreground: #D4D4D4;--ch-t-editor-rangeHighlightBackground: #ffffff0b;--ch-t-editor-infoForeground: #3794FF;--ch-t-editor-selectionBackground: #264F78;--ch-t-focusBorder: #007FD4;--ch-t-tab-activeBackground: #1E1E1E;--ch-t-tab-activeForeground: #ffffff;--ch-t-tab-inactiveBackground: #2D2D2D;--ch-t-tab-inactiveForeground: #ffffff80;--ch-t-tab-border: #252526;--ch-t-tab-activeBorder: #1E1E1E;--ch-t-editorGroup-border: #444444;--ch-t-editorGroupHeader-tabsBackground: #252526;--ch-t-editorLineNumber-foreground: #858585;--ch-t-input-background: #3C3C3C;--ch-t-input-foreground: #D4D4D4;--ch-t-icon-foreground: #C5C5C5;--ch-t-sideBar-background: #252526;--ch-t-sideBar-foreground: #D4D4D4;--ch-t-sideBar-border: #252526;--ch-t-list-activeSelectionBackground: #094771;--ch-t-list-activeSelectionForeground: #fffffe;--ch-t-list-hoverBackground: #2A2D2E; }

Web3Auth Signer

In this guide, you will learn how to create aWeb3Auth(opens in a new tab) signer that can be added as a Safe owner and used to initialize any of the kits from the Safe{Core} SDK.

Check out theSafe Signers demo app(opens in a new tab) on GitHub to follow along this guide with the completed code.

Prerequisites

- Node.js and npm(opens in a new tab)
- .
- AWeb3Auth account(opens in a new tab)
- and a Client ID.

Install dependencies

npm yarn pnpm _10 npm install @web3auth/modal @web3auth/base @web3auth/ethereum-provider

Steps

Imports

Here are the necessary imports for this guide.

_10 import { CHAIN_NAMESPACES, WEB3AUTH_NETWORK } from '@web3auth/base' _10 import {

EthereumPrivateKeyProvider } from '@web3auth/ethereum-provider' _10 import { Web3Auth } from '@web3auth/modal' In addition, you will need to import a web3 library of your choice to use in the "Get the provider and signer" section. In this guide, we are usingviem .

Get the Client ID

Check Web3Auth documentation on how to create a new project in theidashboard(opens in a new tab) andget the client ID(opens in a new tab) .

Once you have it, you need to initialize theWEB3AUTH_CLIENT_ID variable with it.

_10 const WEB3AUTH_CLIENT_ID = // ...

Initialize Web3Auth

Web3Auth initialization is done by instantiating theWeb3Auth class with aclientId , aprivateKeyProvider and some other configuration parameters likeweb3AuthNetwork , depending on if you are using it in a development or production environment. The full list of parameters can be checked in theInstantiating Web3Auth guide(opens in a new tab) from Web3Auth.

To instantiate theprivateKeyProvider you need to define the configuration of the selected chain for the signer with several properties like the ones listed below as an example for Sepolia testnet.

After initializing theweb3auth instance, you need to call theinitModal() method.

_22 const chainConfig = { _22 chainNamespace: CHAIN_NAMESPACES.EIP155, _22 chainId: '0xaa36a7', _22 rpcTarget: 'https://ethereum-sepolia-rpc.publicnode.com', _22 displayName: 'Ethereum Sepolia Testnet', _22 blockExplorerUrl: 'https://sepolia.etherscan.io', _22 ticker: 'ETH', _22 tickerName: 'Ethereum', _22 logo: 'https://cryptologos.cc/logos/ethereum-eth-logo.png' _22 } _22 _22 const privateKeyProvider = new EthereumPrivateKeyProvider({ _22 config: { chainConfig } _22 }) _22 _22 const web3auth = new Web3Auth({ _22 clientId: WEB3AUTH_CLIENT_ID, _22 privateKeyProvider, _22 web3AuthNetwork: WEB3AUTH_NETWORK.SAPPHIRE_MAINNET _22 }) _22 _22 await web3auth.initModal()

Login

To login with an email address or social account you need to call the following method, that will open the popup and request the user to submit the login form.

_10 const web3authProvider = await web3auth.connect()

Get the provider and signer

Once the user is logged in, you can get theprovider andsigner , which is the externally-owned account of the user that was derived from its credentials.

viem You can instantiate the provider usingviem and the following imports:

_10 import { createWalletClient, custom } from 'viem' _10 import { sepolia } from 'viem/chains' _10 const provider = createWalletClient({ _10 chain: sepolia, _10 transport: custom(web3authProvider) _10 }) _10 _10 const signer = await provider.getAddresses())[0] With theprovider andsigner you are ready to instantiate any of the kits from the Safe{Core} SDK and set up or use this signer as a Safe owner.

Logout

Finally, to logout the user, call thelogout() method.

_10 await web3auth.logout()

Recap and further reading

After following this guide, you are able to create a Safe signer using Web3Auth and get theprovider andsigner required to initialize the kits from the Safe{Core} SDK.

Learn more about Web3Auth by checking the following resources:

- Web3Auth website(opens in a new tab)
- Web3Auth documentation(opens in a new tab)
- Web3Auth quickstart guide(opens in a new tab)

Privy Onramp Was this page helpful?

Report issue