How to use a Particle Network signer with permissionless.js

Particle Network is an intent-centric, modular wallet-as-a-service (WaaS). By utilizing MPC-TSS for key management, Particle can streamline onboarding via familiar Web2 methods such as Google, emails, and phone numbers.

By combining permissionless.js with Particle, you can use Particle to enable a smooth social login experience, while using ZeroDev as the smart wallet to sponsor gas for users, batch transactions, and more.

Setup

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

- Refer to theParticle Network documentation site
- for instructions on setting up an application with the Particle Network.
- For a quick start, Particle Network provides a guide, availablehere
- .

Integration

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

Create the Particle Network object

After following the Particle Network documentation, you will have access to aParticleProvider object as shown below that you can use to create aSmartAccountSigner object:

```

import{ ParticleNetwork }from"@particle-network/auth" import{ ParticleProvider }from"@particle-network/provider" import{ providerToSmartAccountSigner }from"permissionless"

// Param options here will be specific to your project. See the Particle docs for more info. constparticle=newParticleNetwork({ projectId, clientKey, appId, chainName, chainId, }) constparticleProvider=newParticleProvider(particle.auth)

// Convert the particleProvider to a SmartAccountSigner constsmartAccountSigner=awaitproviderToSmartAccountSigner(particleProvider)

```

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, })

```
```