Privy

Privy is a simple toolkit for progressive authentication in web3. With Privy you can add features that allow you to onboard users across traditional and web3 authentication methods. Users can sign in to your app with a crypto wallet, an email address, their phone number, or even a social profile (e.g., Twitter or Discord). Users don’t need to have a wallet to explore your product.

tip Check out an end-to-end integration of Privy with Biconomy on thisexample app andrepo ! Read below to learn how to configure your app to create smart accounts for all your users using Privy and Biconomy. This guide assumes you are using React or a React based framework such as Next JS.

1. Install Privy and Biconomy

In your app's repository, install the@privy-io/react-auth SDK from Privy and the@biconomy/{account, bundler, common, core-types, paymaster} SDKs from Biconomy:

yarn add @privy-io/react-auth @biconomy/account @biconomy/bundler @biconomy/common @biconomy/core-types @biconomy/paymaster

2. Configure your app'sPrivyProvider

First, follow the instructions in thePrivy Quickstart to get your app set up with Privy.

Then, update theconfig.embeddedWallets property of yourPrivyProvider to have the following values:

- createOnLogin
- : 'users-without-wallets'. This will configure Privy to create an embedded wallet for users logging in via a web2 method (email, phone, socials), ensuring that all of your users have a wallet that can be used as an EOA.
- noPromptOnSignature
- : true. This will configure Privy to not show its default UIs when your user must sign messages or send transactions. Instead, we recommend you use your own custom UIs for showing users theUserOperations
- they sign.

Your PrivyProvider should then look like:

< PrivyProvider appId = ' insert-your-privy-app-id ' config = { { /*Replace this with your desired login methods*/ loginMethods :

['email' ,

'wallet'] , / *Replace this with your desired appearance configuration*/ appearance :

{ theme :

'light' , accentColor :

'#676FFF' , logo :

'your-logo-url' } embeddedWallets :

{ createOnLogin :

'users-without-wallets' , noPromptOnSignature :

true } } }

{ / *Your app's components* / } </ PrivyProvider

3. Configure your Biconomy bundler and paymaster

Go to theBiconomy Dashboard and configure a Paymaster and a Bundler for your app. Make sure these correspond to the desired network for your user's smart accounts. You can learn more about the dashboardhere

4. Initialize your users' smart accounts

When users log into your app, Privy provisions each user an embedded wallet, which is an EOA. In order to leverage the features of Biconomy's account abstraction, each user also needs a Biconomy smart account. You can provision Biconomy smart accounts for each user by assigning their embedded wallet as a signer for their smart account.

To start, after a user logs in, find the user's embedded wallet from Privy'suseWallets hook, and switch its network to your app's target network. You can find embedded wallet by finding the only entry in theuseWallets array with awalletClientType of 'privy'.

import

{ useWallets }

from

'@privy-io/react-auth' ;

...

// Find the embedded wallet const

{ wallets }

=

useWallets ( ) ; const embeddedWallet = wallets . find ((wallet)

=>

(wallet . walletClientType

===

'privy')) ; // Switch the embedded wallet to your target network // Replace '80001' with your desired chain ID. await embeddedWallet . switchChain (80001) ; Then, initialize a validation module for the user's smart account, by passing an ethers signer from the user's embedded wallet to Biconomy'sECDSAOwnershipValidationModule . This allows the user to authorize actions from their Biconomy smart account by signing messages with their Privy embedded wallet.

import

{ createSmartAccountClient ,

LightSigner

}

from

"@biconomy/account" ;

// Get an ethers provider and signer for the user's embedded wallet const provider =

await embeddedWallet . getEthersProvider ( ) ; const signer = provider . getSigner ( ) ;

const smartAccount =

await

createSmartAccountClient ({ signer : signer as

LightSigner , bundlerUrl :

"" ,

// <-- Read about this at https://docs.biconomy.io/dashboard#bundler-url biconomyPaymasterApiKey :

"" ,

// <-- Read about at https://docs.biconomy.io/dashboard/paymaster rpcUrl :

""

// <-- read about this at https://docs.biconomy.io/Account/methods#createsmartaccountclient }) ;

const address =

await smartAccount . getAccountAddress ( ) ; Details Note: if your app uses React, you can store the user's Biconomy smartAccount in a React context that wraps your application. This allows you to easily access the smart account from your app's pages and components. You can see an example of this in Privy'sexample app . Previous Particle Network Next Magic