Session Keys - [early access feature]

Sessions can be used to send transactions from a dApp on behalf of a user without requiring their confirmation with a wallet.

The user is guaranteed that the dApp can only execute transactions that comply with the policies of the session up until the session expires. Let's go through the steps on how you could enable sessions in your dApps with Argent:

1. Installation

```

Copy npm install @argent/x-sessions

or

pnpm add @argent/x-sessions

or

yarn add @argent/x-sessions

```

1. Create an offchain session as a dapp

To do this, you first need to have a deployed account. This account will authorise the session and interact with the contracts of your dapp. You can get that by using an injected instance of a wallet:

```

Copy constaccount=windowStarknet.account

```

1. Come up with the list of permissions/approved methods for your sessions

Next up, you'll need to communicate a list of whitelisted methods with argent. The method property needs to be with the following format: method: 0x.. instead of the selector. You also need to generate the session keypair you want to grant these rights to.

```

Copy import{ OffchainSessionAccount, createOffchainSession }from"@argent/x-sessions"

import{ getStarkKey,utils }from"micro-starknet"// random key for this example constsessionSigner=utils.randomPrivateKey()

constrequestSession:RequestSession={ sessionKey:getStarkKey(sessionSigner), expirationTime:Math.floor( (Date.now()+1000*60*60*24)/1000 ),// set your time // set your list of contract addresses and methods allowedMethods:[ { contractAddress:"0x...", method:"transfer" } ] }

constgasFees:GasFees={ tokenAddress:"0x...", maximumAmount:{ low:"0x...", high:"0x..." } }

```

1. Sign the session with the account you created

```

Copy import{ createOffchainSession }from"@argent/x-sessions" // calls account.signMessage internally constsignedSession=createOffchainSession( requestSession, window.starknet.account, gasFees )

```

Using established sessions

Once you sign the session with the account you created, users can now use it to do transactions without having to approve again:

```

Copy import{ OffchainSessionAccount }from"@argent/x-sessions"

constoffchainSessionAccount=newOffchainSessionAccount( newRpcProvider({ nodeUrl:"url"}),// provider, in this example RpcProvider account.address,// current account address sessionSigner,// in this example has value utils.randomPrivateKey() -- utils from micro-starknet signedSession,// created session account// the actual account )

// this transaction should get executed without the user having to approve again consttx=sessionAccount.execute({ // lets assume this is a erc20 contract contractAddress:"0x...", selector:"transfer", calldata:[ "0x..." // ... ] })

```

Having done these, congratulations! you just enabled sessions in your dApp!

It's worth noting that session keys are at the time of writing, just supported on Argent Web Wallet as an early access feature, thus if you need to use them in your dApp, do reach out to us atdapps@argent.xyz .

Previous Cosigners and multi-signature verification Next Outside Execution

Last updated11 days ago