

Transaction signatures

This guide explains how transactions are signed by the Safe owners using the Protocol Kit.

Before starting, check this guide's [setup](#) .

Create the transaction

The `createTransaction` method in the Protocol Kit allows the creation of new Safe transactions and returns an instance of the `EthSafeTransaction` class.

```
// Create a transaction to send 0.01 ETH const
```

```
safeTransactionData :
```

```
SafeTransactionDataPartial
```

```
= { to :
```

```
'0x90F8bf6A479f320ead074411a4B0e7944Ea8c9C1' , value :
```

```
'100000000000000000' ,
```

```
// 0.01 ETH data :
```

```
'0x' }
```

```
let safeTransaction =
```

```
await
```

```
protocolKit.createTransaction ({ transactions : [safeTransactionData] })
```

The returned `safeTransaction` object contains the transaction data (`safeTransaction.data`) and a map of the owner-signature pairs (`safeTransaction.signatures`). The structure is similar to the `EthSafeMessage` class but applied for transactions instead of messages.

We use `let` to initialize the `safeTransaction` variable because we will add the signatures later.

```
class
```

```
EthSafeTransaction
```

```
implements
```

```
SafeTransaction { data :
```

```
SafeTransactionData signatures :
```

```
Map < string ,
```

```
SafeSignature
```

```
=
```

```
new
```

```
Map () ... // Other properties and methods }
```

Sign the transaction

Once the `safeTransaction` object is created, we need to collect the signatures from the signers who will sign it.

Following our [setup](#) , we will sign a Safe transaction from `safe3_4` , the main Safe account in this guide. To do that, we first need to sign the same transaction with its owners: `owner1` , `owner2` , `safe1_1` , and `safe2_3` .

ECDSA signature

This applies to `owner1` and `owner2` accounts, as both are EOAs.

The `signTransaction` method takes the `safeTransaction` together with a `signingMethod` and adds the new signature to the `safeTransaction.signatures` map. Depending on the type of message, the `signingMethod` can take these values:

- `SigningMethod.ETH_SIGN`
- `SigningMethod.ETH_SIGN_TYPED_DATA_V4`

```
// Connect the EthAdapter from owner1 protocolKit =
```

```
await
```

```
protocolKit.connect ({ ethAdapter : ethAdapter1 })
```

```
// Sign the safeTransaction with owner1 // After this, the safeTransaction contains the signature from owner1 safeTransaction =
```

```
await
```

```
protocolKit.signTransaction ( safeTransaction , SigningMethod . ETH_SIGN )
```

```
// Connect the EthAdapter from owner2 protocolKit =
```

```
await
```

```
protocolKit.connect ({ ethAdapter : ethAdapter2 })
```

```
// Sign the safeTransaction with owner2 // After this, the safeTransaction contains the signature from owner1 and owner2 safeTransaction =
```

```
await
```

```
protocolKit.signTransaction ( safeTransaction , SigningMethod . ETH_SIGN_TYPED_DATA_V4 )
```

At this point, the `safeTransaction` object should look like this:

```
EthSafeTransaction { signatures :
```

```
Map ( 2 ) { '0x90f8bf6a479f320ead074411a4b0e7944Ea8c9C1'
```

```
=> EthSafeSignature { signer :
```

```
'0x90F8bf6A479f320ead074411a4B0e7944Ea8c9C1' , data :
```

```
'0x969308e2abeda61a0c9c41b3c615012f50dd7456ca76ea39a18e3b975abeb67f275b07810dd59fc928f3f9103e52557c1578c7c5c171ffc983afa5306466b1261f' , isContractSignature :
```

```
false } , '0xffcf8fdee72ac11b5c542428b35eef5769c409f0'
```

```
=> EthSafeSignature { signer :
```

```
'0xFFcf8FDEE72ac11b5c542428B35EEF5769C409f0' , data :
```

```
false } } , data : { ... } }
```

An ECDSA signature comprises two 32-byte integers (r , s) and an extra byte for recovery (v), totaling 65 bytes. In hexadecimal string format, each byte is represented by two characters. Hence, a 65-byte Ethereum signature will be 130 characters long. Including the 0x prefix commonly used with signatures, the total character count for such a signature would be 132.

The final part of the signature, either 1f or 1c , indicates the signature type.

- 0
- : Contract signature.
- 1
- : Approved hash.
- {27, 28} + 4
- : Ethereum adjusted ECDSA recovery byte for EIP-191 signed message.

- Other: Ethereum adjusted ECDSA recovery byte for raw signed hash.

The hexadecimal value 1c equals the decimal number 28, indicating that the signature is a typed data signature.

0x969308e2abeda61a0c9c41b3c615012f50dd7456ca76ea39a18e3b975abeb67f275b07810dd59fc928f3f9103e52557c1578c7c5c171ffc983afa5306466b1261f :

Smart contract signatures

1/1 Safe account

We need to connect the Protocol Kit to `safe1_1` and the `owner3` account (the only owner of `safe1_1`) and sign the transaction.

await

```
// Connect the adapter from owner3 and the address of safe1_1 protocolKit =
```

await

```
// Sign the transactionSafe1_1 with owner3 // After this, transactionSafe1_1 contains the signature from owner3 transactionSafe1_1 =
```

await

When signing with a child Safe account, we need to specify the parent Safe address to generate the signature based on the version of the contract.

EthSafeTransaction { signatures :

```
=> EthSafeSignature { signer :
```

```
"0x5edb6ffe67dd935d93d07c634970944ba0b096f767b92018ad635e8b28effeea5a1e512f1ad6f886690e0e30a3fae2c8c61d3f83d24d43276acdb3254b92ea5b1f", isContractSignature :
```

```
false } } , data : { ... } }
```

To generate a Safe compatible signature, we use the `buildContractSignature` method, which takes an array of signatures and returns another signature that can be used with Safe accounts. After that, we add the signature from `safe1` to our initial transaction.

signatureSafe1 1

$$=$$

await

```
// Add the signatureSafe1 1 to safeTransaction // After this, the safeTransaction contains the signature from owner1, owner2 and safe1 1
safeTransaction.addSignature(signatureSafe1 1)
```

```
EthSafeSignature { signer :
```

"0x5edb6ffe67dd935d93d07c634970944ba0b096f767b92018ad635e8b28effeea5a1e512f1ad6f886690e0e30a3fae2c8c61d3f83d24d43276acdb3254b92ea5b1f".isContractSignature :

```
true }
```

The signatureSafe1_1.data signature should look like this:

[illegible]

Type	Description	Bytes	Value	Hex	Hex string	characters
1	0x Verifier Padded address of the contract that implements the EIP-1271 interface to verify the signature. The Safe signer address 32					

2/3 Safe account

$$=$$

```
=> EthSafeSignature { signer :
```

```

'0xFFcf8FDEE72ac11b5c542428B35EEF5769C409f0', data :
'0x4d63c79cf9d743782bc31ad58c1a316020b39839ab164caee7ecac9829f685cc44ec0d066a5dfe646b2fFeeb37575df131daf9c96ced41b8c7c4aea8dc5461801c' , isContractSignature :
false } , '0x215033cde0619d60b7352348f4598316cc39bc6e'
=> EthSafeSignature { signer :
'0x215033cdE0619D60B7352348F4598316Cc39bC6E' , data :
'0x5edb6ffe67dd935d93d07c634970944ba0b096f767b92018ad635e8b28effeea5a1e512f1ad6f886690e0e30a3fae2c8c61d3f83d24d43276acdb3254b92ea5b1f' , isContractSignature :
true } , '0xf75d61d6c27a7cc5788e633c1fc130f0f4a62d33'
=> EthSafeSignature { signer :
'0xf75D61D6C27a7CC5788E633c1FC130f0F4a62D33' , data :
'0x023d1746ed548e90f387a6b8ddba26e6b80a78d5bfb36e5bfc6fd63e136f8071db6e91c037fa36bde72159138bbb74fc359b35eb515e276a7c0547d5eaa042520d3e6565e5590641db447277243cf24711
' , isContractSignature :
true } } , data : { to :
'0x90F8bf6A479f320ead074411a4B0e7944Ea8c9C1' , value :
'100000000000000000' , data :
'0x' , operation :
0 , baseGas :
'0' , gasPrice :
'0' , gasToken :
'0x0000000000000000000000000000000000000000000000000000000000000000' , refundReceiver :
'0x0000000000000000000000000000000000000000000000000000000000000000' , nonce :
0 , safeTxGas :
'0' } }

```

Propose the transaction

To store the transactions and signatures off-chain, we need to call the Safe Transaction Service API - a centralized and open-source service that anyone can deploy and run.

The Safe Transaction Service is used by [Safe\(Wallet\)\(opens in a new tab\)](#) to store transactions and signatures by default.

To store a new transaction, we need to call the `proposeTransaction` from the API Kit, passing the Safe address, an object with the transaction, and a signature from one owner.

```

const
signerAddress
= ( await
ethAdapter1 .getSignerAddress () ) ||
'0x'
// Get the signature from owner1 const
signatureOwner1
=
safeTransaction .getSignature (signerAddress) as
EthSafeSignature
// Get the transaction hash of the safeTransaction const
safeTransactionHash
=
await
protocolKit .getTransactionHash (safeTransaction)
// Instantiate the API Kit // Use the chainId where you have the Safe account deployed const
apiKit
=
new
SafeApiKit ({ chainId })
// Propose the transaction await
apiKit .proposeTransaction ({ safeAddress : safe3_4 , safeTransactionData :
safeTransaction .data , safeTxHash : safeTransactionHash , senderAddress : signerAddress , senderSignature :
buildSignatureBytes ([signatureOwner1]) })

```

The transaction is now publicly available in the Safe Transaction Service with the signature of the owner who submitted it.

Confirm the transaction

To add the signatures from the remaining owners, we need to call the `confirmTransaction` , passing the `safeMessageHash` and a signature from the owner.

Once a transaction is proposed, it becomes available or [Safe\(Wallet\)\(opens in a new tab\)](#) . However, to execute the transaction, all the confirmations from the owners are needed.

```

const
signerAddress
= ( await
ethAdapter2 .getSignerAddress () ) ||
'0x'

```

[Report issue](#)