Safe Transaction Service

Safe Transaction Service keeps track of transactions sent via Safe contracts. It indexes these transactions using events (L2 chains) and tracing (L1 chains) mechanisms.

Key Features:

- Blockchain indexing
- : Executed transactions, configuration changes, ERC-20/721 transfers, and onchain confirmations are automatically indexed from the blockchain.
- Offchain transaction signatures
- : Transactions can be sent to the service, enabling offchain signature collection. This feature helps inform owners about pending transactions awaiting confirmation for execution.
- Offchain messages
- : The service can collect offchain signatures to confirm messages followingEIP-1271(opens in a new tab)
- .
- Transaction decoding
- : The service keeps getting source and ABIs from contracts that interact with Safe to decode these interactions.

Technology stack overview

Safe Transaction Service is aDjango(opens in a new tab) app written in Python with the following architecture:

- Gunicorn(opens in a new tab)
- : A Python WSGI HTTP Server.
- Celery(opens in a new tab)
- : A task queue with focus on real-time processing, while also supporting task scheduling. Safe Transaction Service has a scheduler (for periodic tasks), a worker indexer to consume and execute indexing tasks, and a contracts worker mainly to get metadata from contracts.
- RabbitMQ(opens in a new tab)
- : A distributed message broker system Celery uses to share messages between the scheduler, workers, and the Django application.
- PostgreSQL(opens in a new tab)
- : An open source object-relational database system.
- Redis(opens in a new tab)
- : An open source, in-memory data structure store used for caching by the Safe Transaction Service.
- safe-eth-py(opens in a new tab)
- : A library to interact with Safe and blockchains.

Blockchain indexing

Safe Transaction Service can index automatically executed transactions, configuration changes, ERC-20/721 transfers, and onchain confirmations. The indexer is running onworker-indexer by different periodictasks(opens in a new tab) .

ERC-20 and ERC-721 are indexed usingeth_getLogs (opens in a new tab) filtered by the Transfer topickeccak('Transfer(address,address,uint256)') .

Safe creation, executed transactions, configuration changes, and onchain confirmations are indexed differently depending on whether the chain is L1 or L2.

For L1 chains, the indexer calls tracing methods. For the oldest blockstrace_filter (opens in a new tab) is used filtering by singleton address of Safe contracts, and for the latest blockstrace_block (opens in a new tab) is used, astrace_filter takes longer to return updated information.trace_block will be used if the block depth is lower thanETH_INTERNAL_TXS_NUMBER_TRACE_BLOCKS . The environment variables indexing uses are definedhere(opens in a new tab) .

For L2 chains, the indexing is by events with theeth_getLogs (opens in a new tab) method with the corresponding topics.

From Safe creation, the Transaction Service stores each contract change on theSafeStatus model asnonce ,owners , etc. The latest and current status of a Safe is stored asSafeLastStatus for easy database access and optimization.

The following endpoints show the current indexing status of the Safe Transaction Service:

- /v1/about/indexing/

Response example:

{ "currentBlockNumber" :

9773327 ,

// Last block on blockchain "erc20BlockNumber" :

9773326 ,

// Last block indexed for erc20/721 events "erc20Synced" :

true , "masterCopiesBlockNumber" :

9773327 ,

// Last block indexed for executed transactions, ether transfers, configuration changes, etc. "masterCopiesSynced" :

true , "synced" :

true }

Reorgs handling

Every block is marked asnot confirmed during indexing unless it has some depth (configured via theETH_REORG_BLOCKS environment variable). Unconfirmed blocks are checked periodically to see if the blockchainblockHash for that number changed before it reaches the desired number of confirmations. If that's the case, all blocks from that block and related transactions are deleted, and indexing is restarted to the last confirmed block.

Note: No offchain signatures, transactions, or messages are lost in this process. Only onchain data is removed.

Offchain transaction signatures

Safe Transaction Service can collect offchain transaction signatures, allowing the owners to share their signatures to reach the required threshold before executing a transaction and spending less gas than onchain approvals.

The following endpoints let us propose a transaction and collect every confirmation (offchain signatures):

- POST /v1/safes/{address}/multisig-transactions/
- : Creates a new transaction. Requires at least one signature.
- POST /v1/multisig-transactions/{safe_tx_hash}/confirmations/
- : Adds a new confirmation. Needssafe_tx_hash
- .
- GET /v1/multisig-transactions/{safe_tx_hash}/
- : Returns all the multisig transaction information.
- GET /v1/multisig-transactions/{safe_tx_hash}/confirmations/
- : Returns the list of all confirmations to a multisig transaction.

The following sequence diagram shows a use case for a Safe shared by Alice and Bob where at least one confirmation for each one is required:

What's thesafe_tx_hash ?

safe_tx_hash is the unique identifier for a Safe transaction and is calculated using the EIP-712(opens in a new tab) standard: keccak256(0x19 || 0x1 || domainSeparator || safeTxHashStruct)

wheresafeTxHashStruct is thehashStruct of a Safe transaction.

The following example shows how to get asafe_tx_hash with safe-eth-py (opens in a new tab)

from gnosis . safe . safe_tx import SafeTx from gnosis . eth . ethereum_client import EthereumClient eth_client =

EthereumClient ("https://sepolia.gateway.tenderly.co") safe_tx =

SafeTx (eth_client, "0x4127839cdf4F73d9fC9a2C2861d8d1799e9DF40C" , "0xc6b82bA149CFA113f8f48d5E3b1F78e933e16DfD" , 10000000000000000 , "" , 0 , 0 , 0 , 0 , "0x0000000000000000000000000000000000000000" , "0x0000000000000000000000000000000000000000" , safe_nonce = 206) print (safe_tx.safe_tx_hash. hex ())

Output

0x34ae46cf7d884309a438a7e9a3161fa05dfc5068681ac3877a947971af845a18

Offchain messages

Safe Transaction Service can collect the necessary offchain signatures to confirm a message using EIP-1271(opens in a new tab) . The message can be a string (EIP-191 is used to get the hash) or an object EIP-712.

Messages endpoints

- GET /v1/safes/{address}/messages/
- : Returns the messages created for the given Safe address.
- POST /v1/safes/{address}/messages/

- : Creates a message with at least one signature.
- GET /v1/messages/{message_hash}/
- : Returns a message for a given message hash.
- POST /v1/messages/{message_hash}/signatures/
- : Adds another signature to the message with the given message hash.

The following sequence diagram shows a use case for a Safe shared by Alice and Bob where at least one signature for each one is required to confirm a message fully:

Message string example

Python

safe-eth-py is required for this example.

from gnosis . eth . ethereum_client import EthereumClient from gnosis . safe . safe import Safe from eth_account . messages import defunct_hash_message from eth_account import Account import requests

alice

Account . from_key ("Alice_key")

Message that we want to confirm

message

"Hello SafeMessages"

Hash EIP-191

message_hash

defunct_hash_message (text = message)

get message hash from safe

eth_client

EthereumClient ("https://sepolia.gateway.tenderly.co") safe_address =

"TheAliceAndBobSafeAddress" safe =

Safe (safe_address, eth_client) safe_message_hash = safe . get_message_hash (message_hash)

Alice is going to create the message on safe Transaction Service

First sign the safe_message_hash

signature_alice

alice . signHash (safe_message_hash)

Create the request

body

{ "message" : message , "safeAppId" :

0 , "signature" : signature_alice . signature . hex () } requests . post (f 'https://safe-transaction-sepolia.safe.global/api/v1/safes/ { safe_address } /messages/' ,json = body)

Message was created, let's request by message hash

response

requests . get (f 'https://safe-transaction-sepolia.safe.global/api/v1/messages/ { safe_message_hash. hex () } /') print (response. json ())

Adding Bob confirmation

bob

Account . from_key ("Bob_key") signature_bob = bob . signHash (safe_message_hash)

Create the request

body

{ "signature" : signature_bob . signature . hex () } requests . post (f 'https://safe-transaction-sepolia.safe.global/api/v1/messages/ { safe_message_hash. hex () } /signatures/' ,json = body)

Transaction decoder

The Safe Transaction Service can decode contract interactions. To achieve it, the service periodically gets source and ABIs from different sources like Sourcify, Etherscan, and Blockscout using thesafe-eth-py library.

The detection of contract interactions is done in a periodic task executed every hour formultisig-transaction andmodule-transactions or every six hours formultisend-transactions onworker-contracts-tokens . For every new contract, the service tries to download the source, and the ABI requests it first to Sourcify, then Etherscan, and as a last chance, Blockscout. It's important to know that not all these data sources are supported or configured for every network onsafe-eth-py . Supported and configured networks onsafe-eth-py :

- Sourcify supported networks(opens in a new tab)
- Etherscan configured networks(opens in a new tab)
- Blockscout configured networks(opens in a new tab)

Transaction decoder endpoint

- POST /v1/data-decoder/
- decode a transactiondata
- passed on body for ato
- contract address.

Example transaction decoder

curl

-X

'POST' \ 'https://safe-transaction-sepolia.safe.global/api/v1/data-decoder/' \ -H

'accept: application/json' \ -H

'Content-Type: application/json' \ -H

'X-CSRFToken: Gx1aRa8kIJGIAfReLAWwr9Q6dHv22dFt7VprdipLryHcxpfhk9aV0UDAhNz8gGYz' \ -d

'{ "data": "0x095ea7b3000000000000000000000000e6fc577e87f7c977c4393300417dcc592d90acf8ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff", "to": "0x4127839cdf4F73d9fC9a2C2861d8d1799e9DF40C" }'

Output:

{ "method": "approve", "parameters": [{ "name": "spender", "type": "address", "value": "0xe6fC577E87F7c977c4393300417dCC592D90acF8" }, { "name": "value", "type": "uint256", "value":

"115792089237316195423570985008687907853269984665640564039457584007913129639935" }] }

This decoded data is also included asdataDecoded inGET ofmultisig-transactions ,module-transactions andall-transactions endpoints.

[Service Architecture](#) [RPC Requirements](#)

.mui-style-byb1ir{box-sizing:border-box;-webkit-flex-direction:row;-ms-flex-direction:row;flex-direction:row;padding:24px;margin-top:24px;border-radius:8px;border:1px solid rgba(249,250,251,.1);}

.mui-style-v3z1wi{box-sizing:border-box;display:-webkit-box;display:-webkit-flex;display:-ms-flexbox;display:flex;-webkit-box-flex-wrap:wrap;-webkit-flex-wrap:wrap;-ms-flex-wrap:wrap;flex-wrap:wrap;width:100%;-webkit-flex-direction:row;-ms-flex-direction:row;flex-direction:row;-webkit-align-items:center;-webkit-box-align:center;-ms-flex-align:center;align-items:center;} .mui-style-9jjat2{box-sizing:border-box;display:-webkit-box;display:-webkit-flex;display:-ms-flexbox;display:flex;-webkit-box-flex-wrap:wrap;-webkit-flex-wrap:wrap;-ms-flex-wrap:wrap;flex-wrap:wrap;width:100%;-webkit-flex-direction:row;-ms-flex-direction:row;flex-direction:row;-webkit-flex-direction:column;-ms-flex-direction:column;flex-direction:column;-webkit-align-items:center;-webkit-box-align:center;-ms-flex-align:center;align-items:center;} .mui-style-13lb6is{margin:0;font-family:DM Sans,sans-serif;font-weight:400;font-size:16px;line-height:24px;text-align:center;font-weight:700;color:white;}@media (min-width:600px){.mui-style-13lb6is{font-size:18px;line-height:28px;}} Was this page helpful?

.mui-style-1c7bkxc{box-sizing:border-box;margin:0;-webkit-flex-direction:row;-ms-flex-direction:row;flex-direction:row;-webkit-box-pack:space-around;-ms-flex-pack:space-around;-webkit-justify-content:space-around;justify-content:space-around;margin-top:8px;} .mui-style-1bfhgmr{font-family:DM Sans,sans-serif;font-weight:500;font-size:0.875rem;line-height:1.75;text-transform:uppercase;min-width:64px;padding:6px 8px;border-radius:var(--mui-shape-borderRadius);-webkit-transition:background-color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,box-shadow 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,border-color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms;transition:background-color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,box-shadow 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,border-color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms;color:var(--mui-palette-primary-main);text-transform:inherit;padding:4px;margin-right:12px;min-width:0;border-radius:288px;}.mui-style-1bfhgmr:hover{-webkit-text-decoration:none;text-decoration:none;background-color:rgba(var(--mui-palette-primary-mainChannel) / var(--mui-palette-action-hoverOpacity));}@media (hover: none){.mui-style-1bfhgmr:hover{background-color:transparent;}}.mui-style-1bfhgmr.Mui-disabled{color:var(--mui-palette-action-disabled);}.mui-style-1bfhgmr:hover svg path{stroke:rgba(18, 255, 128, 1);} .mui-style-1qvgpq6{display:-webkit-inline-box;display:-webkit-inline-flex;display:-ms-inline-flexbox;display:inline-flex;-webkit-align-items:center;-webkit-box-align:center;-ms-flex-align:center;align-items:center;-webkit-box-pack:center;-ms-flex-pack:center;-webkit-justify-content:center;justify-content:center;position:relative;box-sizing:border-box;-webkit-tap-highlight-color:transparent;background-color:transparent;outline:0;border:0;margin:0;border-radius:0;padding:0;cursor:pointer;-webkit-user-select:none;-moz-user-select:none;-ms-user-select:none;user-select:none;vertical-align:middle;-moz-appearance:none;-webkit-appearance:none;-webkit-text-decoration:none;text-decoration:none;color:inherit;font-family:DM Sans,sans-serif;font-weight:500;font-size:0.875rem;line-height:1.75;text-transform:uppercase;min-width:64px;padding:6px 8px;border-radius:var(--mui-shape-borderRadius);-webkit-transition:background-color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,box-shadow 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,border-color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms;transition:background-color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,box-shadow 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,border-color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms;color:var(--mui-palette-primary-main);text-transform:inherit;padding:4px;margin-right:12px;min-width:0;border-radius:288px;}.mui-style-1qvgpq6::-moz-focus-inner{border-style:none;}.mui-style-1qvgpq6.Mui-disabled{pointer-events:none;cursor:default;}@media print{.mui-style-1qvgpq6{-webkit-print-color-adjust:exact;color-adjust:exact;}}.mui-style-1qvgpq6:hover{-webkit-text-decoration:none;text-decoration:none;background-color:rgba(var(--mui-palette-primary-mainChannel) / var(--mui-palette-action-hoverOpacity));}@media (hover: none){.mui-style-1qvgpq6:hover{background-color:transparent;}}.mui-style-1qvgpq6.Mui-disabled{color:var(--mui-palette-action-disabled);}.mui-style-1qvgpq6:hover svg path{stroke:rgba(18, 255, 128, 1);}

.mui-style-6ltvuo{font-family:DM Sans,sans-serif;font-weight:500;font-size:0.875rem;line-height:1.75;text-transform:uppercase;min-width:64px;padding:6px 8px;border-radius:var(--mui-shape-borderRadius);-webkit-transition:background-color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,box-shadow 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,border-color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms;transition:background-color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,box-shadow 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,border-color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms;color:var(--mui-palette-error-main);text-transform:inherit;padding:4px;min-width:0;border-radius:288px;}.mui-style-6ltvuo:hover{-webkit-text-decoration:none;text-decoration:none;background-color:rgba(var(--mui-palette-error-mainChannel) / var(--mui-palette-action-hoverOpacity));}@media (hover: none){.mui-style-6ltvuo:hover{background-color:transparent;}}.mui-style-6ltvuo.Mui-disabled{color:var(--mui-palette-action-disabled);}.mui-style-6ltvuo:hover svg path{stroke:rgba(255, 95, 114, 1);} .mui-style-1smzlp8{display:-webkit-inline-box;display:-webkit-inline-flex;display:-ms-inline-flexbox;display:inline-flex;-webkit-align-items:center;-webkit-box-align:center;-ms-flex-align:center;align-items:center;-webkit-box-pack:center;-ms-flex-pack:center;-webkit-justify-content:center;justify-content:center;position:relative;box-sizing:border-box;-webkit-tap-highlight-color:transparent;background-color:transparent;outline:0;border:0;margin:0;border-radius:0;padding:0;cursor:pointer;-webkit-user-select:none;-moz-user-select:none;-ms-user-select:none;user-select:none;vertical-align:middle;-moz-appearance:none;-webkit-appearance:none;-webkit-text-decoration:none;text-decoration:none;color:inherit;font-family:DM Sans,sans-serif;font-weight:500;font-size:0.875rem;line-height:1.75;text-transform:uppercase;min-width:64px;padding:6px 8px;border-radius:var(--mui-shape-borderRadius);-webkit-transition:background-color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,box-shadow 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,border-color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms;transition:background-color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,box-shadow 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,border-color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms;color:var(--mui-palette-error-main);text-transform:inherit;padding:4px;min-width:0;border-radius:288px;}.mui-style-1smzlp8::-moz-focus-inner{border-style:none;}.mui-style-1smzlp8.Mui-disabled{pointer-events:none;cursor:default;}@media print{.mui-style-1smzlp8{-webkit-print-color-adjust:exact;color-adjust:exact;}}.mui-style-1smzlp8:hover{-webkit-text-decoration:none;text-decoration:none;background-color:rgba(var(--mui-palette-

error-mainChannel) / var(--mui-palette-action-hoverOpacity));}@media (hover: none){.mui-style-1smzlp8:hover{background-color:transparent;}}.mui-style-1smzlp8.Mui-disabled{color:var(--mui-palette-action-disabled);}.mui-style-1smzlp8:hover svg path{stroke:rgba(255, 95, 114, 1);} .mui-style-9tyvvl{font-family:DM Sans,sans-serif;font-weight:500;font-size:0.875rem;line-height:1.75;text-transform:uppercase;min-width:64px;padding:6px 8px;border-radius:var(--mui-shape-borderRadius);-webkit-transition:background-color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,box-shadow 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,border-color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms;transition:background-color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,box-shadow 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,border-color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms;color:var(--mui-palette-primary-main);text-transform:inherit;color:rgba(249,250,251,.7);}.mui-style-9tyvvl:hover{-webkit-text-decoration:none;text-decoration:none;background-color:rgba(var(--mui-palette-primary-mainChannel) / var(--mui-palette-action-hoverOpacity));}@media (hover: none){.mui-style-9tyvvl:hover{background-color:transparent;}}.mui-style-9tyvvl.Mui-disabled{color:var(--mui-palette-action-disabled);} .mui-style-lmenmj{display:-webkit-inline-box;display:-webkit-inline-flex;display:-ms-inline-flexbox;display:inline-flex;-webkit-align-items:center;-webkit-box-align:center;-ms-flex-align:center;align-items:center;-webkit-box-pack:center;-ms-flex-pack:center;-webkit-justify-content:center;justify-content:center;position:relative;box-sizing:border-box;-webkit-tap-highlight-color:transparent;background-color:transparent;outline:0;border:0;margin:0;border-radius:0;padding:0;cursor:pointer;-webkit-user-select:none;-moz-user-select:none;-ms-user-select:none;user-select:none;vertical-align:middle;-moz-appearance:none;-webkit-appearance:none;-webkit-text-decoration:none;text-decoration:none;color:inherit;font-family:DM Sans,sans-serif;font-weight:500;font-size:0.875rem;line-height:1.75;text-transform:uppercase;min-width:64px;padding:6px 8px;border-radius:var(--mui-shape-borderRadius);-webkit-transition:background-color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,box-shadow 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,border-color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms;transition:background-color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,box-shadow 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,border-color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms,color 250ms cubic-bezier(0.4, 0, 0.2, 1) 0ms;color:var(--mui-palette-primary-main);text-transform:inherit;color:rgba(249,250,251,.7);}.mui-style-lmenmj::-moz-focus-inner{border-style:none;}.mui-style-lmenmj.Mui-disabled{pointer-events:none;cursor:default;}@media print{.mui-style-lmenmj{-webkit-print-color-adjust:exact;color-adjust:exact;}}.mui-style-lmenmj:hover{-webkit-text-decoration:none;text-decoration:none;background-color:rgba(var(--mui-palette-primary-mainChannel) / var(--mui-palette-action-hoverOpacity));}@media (hover: none){.mui-style-lmenmj:hover{background-color:transparent;}}.mui-style-lmenmj.Mui-disabled{color:var(--mui-palette-action-disabled);} Report issue