

Suave

Library to interact with the Suave MEVM precompiles.

Functions

[isConfidential](#)

Returns whether execution is off- or on-chain.

Output:

- b
- (bool
-): Whether execution is off- or on-chain.

[buildEthBlock](#)

Constructs an Ethereum block based on the provided data records.

Input:

- blockArgs
- ([BuildBlockArgs](#)
-): Arguments to build the block.
- dataId
- (DataId
-): ID of the data record with mev-share bundle data.
- namespace
- (string
-): Deprecated.

Output:

- blockBid
- (bytes
-): Block Bid encoded in JSON.
- executionPayload
- (bytes
-): Execution payload encoded in JSON.

[confidentialInputs](#)

Provides the confidential inputs associated with a confidential computation request. Outputs are in bytes format.

Output:

- confidentialData
- (bytes
-): Confidential inputs.

[confidentialRetrieve](#)

Retrieves data from the confidential store. Also mandates the caller's presence in theAllowedPeekers list.

Input:

- dataId
- (DataId
-): ID of the data record to retrieve.
- key

- (string
-): Key slot of the data to retrieve.

Output:

- value
- (bytes
-): Value of the data.

[confidentialStore](#)

Stores data in the confidential store. Requires the caller to be part of theAllowedPeekers for the associated data record.

Input:

- dataId
- (DataId
-): ID of the data record to store.
- key
- (string
-): Key slot of the data to store.
- value
- (bytes
-): Value of the data to store.

[contextGet](#)

Retrieves a value from the context.

Input:

- key
- (string
-): Key of the value to retrieve.

Output:

- value
- (bytes
-): Value of the key.

[doHTTPRequest](#)

Performs an HTTP request and returns the response.request is the request to perform.

Input:

- request
- ([HttpRequest](#)
-): Request to perform.

Output:

- httpResponse
- (bytes
-): Body of the response.

[ethcall](#)

Uses theeth_call JSON RPC method to let you simulate a function call and return the response.

Input:

- contractAddr
- (address
-): Address of the contract to call.
- input1
- (bytes
-): Data to send to the contract.

Output:

- callOutput
- (bytes
-): Output of the contract call.

extractHint

Interprets the bundle data and extracts hints, such as theTo address and calldata.

Input:

- bundleData
- (bytes
-): Bundle object encoded in JSON.

Output:

- hints
- (bytes
-): List of hints encoded in JSON.

fetchDataRecords

Retrieves all data records correlating with a specified decryption condition and namespace.

Input:

- cond
- (uint64
-): Filter for the decryption condition.
- namespace
- (string
-): Filter for the namespace of the data records.

Output:

- dataRecords
- (``): List of data records that match the filter.

fillMevShareBundle

Joins the user's transaction and with the backrun, and returns encoded mev-share bundle. The bundle is ready to be sent viaSubmitBundleJsonRPC .

Input:

- dataId
- (DataId
-): ID of the data record with mev-share bundle data.

Output:

- encodedBundle
- (bytes
-): Mev-Share bundle encoded in JSON.

newBuilder

Initializes a new remote builder session.

Output:

- sessionId
- (string
-): ID of the remote builder session.

newDataRecord

Initializes data records within the ConfidentialStore. Prior to storing data, all data records should undergo initialization via this precompile.

Input:

- decryptionCondition
- (uint64
-): Up to which block this data record is valid. Used during fillMevShareBundle
- precompie.
- allowedPeekers
- (`): Addresses which can get data.
- allowedStores
- (`): Addresses can set data.
- dataType
- (string
-): Namespace of the data.

Output:

- dataRecord
- ([DataRecord](#)
-): Data record that was created.

privateKeyGen

Generates a private key in ECDA secp256k1 format.

Input:

- crypto
- (CryptoSignature
-): Type of the private key to generate.

Output:

- privateKey
- (string
-): Hex encoded string of the ECDSA private key. Exactly as a signMessage precompile wants.

randomBytes

Generates a number of random bytes, given by the argument numBytes.

Input:

- numBytes
- (uint8
-): Number of random bytes to generate.

Output:

- value
- (bytes

-): Randomly-generated bytes.

[signEthTransaction](#)

Signs an Ethereum Transaction, 1559 or Legacy, and returns raw signed transaction bytes. txn is binary encoding of the transaction.

Input:

- txn
- (bytes
-): Transaction to sign (RLP encoded).
- chainId
- (string
-): Id of the chain to sign for (hex encoded, with 0x prefix).
- signingKey
- (string
-): Hex encoded string of the ECDSA private key (without 0x prefix).

Output:

- signedTxn
- (bytes
-): Signed transaction encoded in RLP.

[signMessage](#)

Signs a message and returns the signature.

Input:

- digest
- (bytes
-): Message to sign.
- crypto
- (CryptoSignature
-): Type of the private key to generate.
- signingKey
- (string
-): Hex encoded string of the ECDSA private key.

Output:

- signature
- (bytes
-): Signature of the message with the private key.

[simulateBundle](#)

Performs a simulation of the bundle by building a block that includes it.

Input:

- bundleData
- (bytes
-): Bundle encoded in JSON.

Output:

- effectiveGasPrice
- (uint64
-): Effective Gas Price of the resultant block.

[simulateTransaction](#)

Simulates a transaction on a remote builder session.

Input:

- sessionId
- (string
-): ID of the remote builder session.
- txn
- (bytes
-): Txn to simulate encoded in RLP.

Output:

- simulationResult
- ([SimulateTransactionResult](#)
-): Result of the simulation.

[submitBundleJsonRPC](#)

Submits bytes as JSONRPC message to the specified URL with the specified method. As this call is intended for bundles, it also signs the params and adds X-Flashbots-Signature header, as usual with bundles. Regular eth bundles don't need any processing to be sent.

Input:

- url
- (string
-): URL to send the request to.
- method
- (string
-): JSONRPC method to call.
- params
- (bytes
-): JSONRPC input params encoded in RLP.

Output:

- errorMessage
- (bytes
-): Error message if any.

[submitEthBlockToRelay](#)

Submits a given builderBid to a mev-boost relay.

Input:

- relayUrl
- (string
-): URL of the relay to submit to.
- builderBid
- (bytes
-): Block bid to submit encoded in JSON.

Output:

- blockBid
- (bytes
-): Error message if any.

Structs

[BuildBlockArgs](#)

Arguments to build the block.

- slot
- (uint64
-): Slot number of the block.
- proposerPubkey
- (bytes
-): Public key of the proposer.
- parent
- (bytes32
-): Hash of the parent block.
- timestamp
- (uint64
-): Timestamp of the block.
- feeRecipient
- (address
-): Address of the fee recipient.
- gasLimit
- (uint64
-): Gas limit of the block.
- random
- (bytes32
-): Randomness of the block.
- withdrawals
- (`): List of withdrawals.
- extra
- (bytes
-): Extra data of the block.
- beaconRoot
- (bytes32
-): Root of the beacon chain.
- fillPending
- (bool
-): Whether to fill the block with pending transactions.

[DataRecord](#)

A record of data stored in the ConfidentialStore.

- id
- (DataId
-): ID of the data record.
- salt
- (DataId
-): Salt used to derive the encryption key.
- decryptionCondition
- (uint64
-): Up to which block this data record is valid.
- allowedPeekers
- (`): Addresses which can get data.
- allowedStores
- (`): Addresses can set data.
- version
- (string
-): Namespace of the data record.

[HttpRequest](#)

Description of an HTTP request.

- url
- (string
-): Target url of the request.

- method
- (string
-): HTTP method of the request.
- headers
- (``): HTTP Headers.
- body
- (bytes
-): Body of the request (if Post or Put).
- withFlashbotsSignature
- (bool
-): Whether to include the Flashbots signature.

[SimulateTransactionResult](#)

Result of a simulated transaction.

- egp
- (uint64
-): Effective Gas Price of the transaction.
- logs
- (``): Logs emitted during the simulation.
- success
- (bool
-): Whether the transaction was successful or not.
- error
- (string
-): Error message if any.

[SimulatedLog](#)

A log emitted during the simulation of a transaction.

- data
- (bytes
-): Data of the log.
- addr
- (address
-): Address of the contract that emitted the log.
- topics
- (``): Topics of the log.

[Withdrawal](#)

A withdrawal from the beacon chain.

- index
- (uint64
-): Index of the withdrawal.
- validator
- (uint64
-): ID of the validator.
- Address
- (address
-): Address to withdraw to.
- amount
- (uint64
-): Amount to be withdrawn. [Edit this page](#) [Previous](#) [MevShare](#) [Next](#) [Forge](#)