
polyswarm-client Documentation

PolySwarm

Jun 07, 2019

CONTENTS

1	API Reference	1
1.1	conftest	1
1.2	balancemanager	1
1.3	polyswarmclient	3
1.4	ambassador	19
1.5	microengine	20
1.6	arbiter	23
1.7	worker	25
	Python Module Index	27
	Index	29

API REFERENCE

This page contains auto-generated API reference documentation¹.

1.1 conftest

1.2 balancemanager

1.2.1 Submodules

`balancemanager.__main__`

Module Contents

`balancemanager.__main__.logger`

`balancemanager.__main__.validate_optional_transfer_amount` (*ctx, param, value*)

`balancemanager.__main__.validate_transfer_amount` (*ctx, param, value*)

`balancemanager.__main__.polyswarm_client` (*func*)

`balancemanager.__main__.cli` (*log, client_log, log_format*)

Entrypoint for the balance manager driver

Parameters

- **log** (*str*) – Log level for balancemanager module logs
- **client_log** (*str*) – Log level for all polyswarmclient module logs
- **log_format** (*str*) – Choose either json, or text log format

`balancemanager.__main__.deposit` (*polyswarmd_addr, keyfile, password, api_key, testing, insecure_transport, denomination, all, amount*)

Entrypoint to deposit NCT into a sidechain

Parameters

- **polyswarmd_addr** (*str*) – Address for the polyswarmd server
- **keyfile** (*str*) – Path to the keyfile
- **password** (*str*) – Password to unlock keyfile

¹ Created with sphinx-autoapi

- **api_key** (*str*) – ApiKey to access polyswarmd
- **testing** (*int*) – Number of tests to run
- **insecure_transport** (*bool*) – Flag to allow use of http instead of https
- **denomination** (*str*) – Choose to interpret amount as nct, nct-gwei, or nct-wei
- **all** (*bool*) – Choose to deposit the entire homechain balance
- **amount** (*float*) – Amount of Nectar (NCT) to transfer

`balancemanager.__main__.withdraw (polyswarmd_addr, keyfile, password, api_key, testing, insecure_transport, denomination, all, amount)`

Entrypoint to withdraw NCT from a sidechain into the homechain

Parameters

- **polyswarmd_addr** (*str*) – Address for the polyswarmd server
- **keyfile** (*str*) – Path to the keyfile
- **password** (*str*) – Password to unlock keyfile
- **api_key** (*str*) – ApiKey to access polyswarmd
- **testing** (*int*) – Number of tests to run
- **insecure_transport** (*bool*) – Flag to allow use of http instead of https
- **denomination** (*str*) – Choose to interpret amount as nct, nct-gwei, or nct-wei
- **all** (*bool*) – Choose to withdraw the entire sidechain balance
- **amount** (*float*) – Amount of Nectar (NCT) to transfer (if not all)

`balancemanager.__main__.maintain (polyswarmd_addr, keyfile, password, api_key, testing, insecure_transport, denomination, maximum, withdraw_target, confirmations, minimum, refill_amount)`

Entrypoint to withdraw NCT from a sidechain into the homechain

Parameters

- **polyswarmd_addr** (*str*) – Address for the polyswarmd server
- **keyfile** (*str*) – Path to the keyfile
- **password** (*str*) – Password to unlock keyfile
- **api_key** (*str*) – ApiKey to access polyswarmd
- **testing** (*int*) – Number of tests to run
- **insecure_transport** (*bool*) – Flag to allow use of http instead of https
- **denomination** (*str*) – Choose to interpret amount as nct, nct-gwei, or nct-wei
- **maximum** (*int*) – Maximum balance to before starting a withdrawal from sidechain
- **withdraw_target** (*int*) – Target value after performing a withdrawal
- **confirmations** (*int*) – Number of confirmations to wait to confirm a transfer occurred
- **minimum** (*float*) – Value of NCT on sidechain where you want to transfer more NCT
- **refill_amount** (*float*) – Value of NCT to transfer anytime the balance falls below the minimum

1.2.2 Package Contents

`balancemanager.logger`

`balancemanager.RELAY_LEEWAY = 5`

`balancemanager.convert` (*client, denomination, amount*)

Convert the amount from it's original precision to 18 decimals

class `balancemanager.BalanceManager` (*client, denomination, transfer_all, amount, testing=0, chains=None*)

Bases: `object`

Balance manager is used for single transfer events in either direction. Create a client, choose a chain and amount then run it.

run (*self*)

Starts the client on whichever chain this uses.

run_one-shot (*self*)

Runs `run_task` once

class `balancemanager.Deposit` (*client, denomination, transfer_all, amount, testing=0*)

Bases: `balancemanager.BalanceManager`

Deposit only version of Balance Manager

class `balancemanager.Withdraw` (*client, denomination, transfer_all, amount, testing=0*)

Bases: `balancemanager.BalanceManager`

Withdraw only version of Balance Manager

class `balancemanager.Maintainer` (*client, denomination, confirmations, minimum, refill_amount, maximum, withdraw_target, testing=0*)

Bases: `object`

This class maintains a balance on the sidechain. It requires a base setup of a minimum balance. Optionally, it can take a maximum balance, so that earnings can automatically be transferred back to the homechain.

run (*self*)

Starts the client. Have to run with both chains, or lots of nonce errors

1.3 polyswarmclient

1.3.1 Submodules

`polyswarmclient.abstractambassador`

Module Contents

`polyswarmclient.abstractambassador.logger`

`polyswarmclient.abstractambassador.MAX_TRIES = 10`

`polyswarmclient.abstractambassador.BOUNTY_QUEUE_SIZE = 10`

`polyswarmclient.abstractambassador.MAX_BOUNTIES_IN_FLIGHT = 10`

`polyswarmclient.abstractambassador.MAX_BOUNTIES_PER_BLOCK = 1`

`polyswarmclient.abstractambassador.BLOCK_DIVISOR = 1`

```
class polyswarmclient.abstractambassador.QueuedBounty (artifact_type, amount,
                                                         ipfs_uri, duration,
                                                         api_key=None)
```

Bases: object

```
__repr__ (self)
```

```
class polyswarmclient.abstractambassador.AbstractAmbassador (client, testing=0,
                                                             chains=None,
                                                             watchdog=0, sub-
                                                             mission_rate=0)
```

Bases: abc.ABC

```
classmethod connect (cls, polyswarmd_addr, keyfile, password, api_key=None, testing=0, inse-
                    cure_transport=False, chains=None, watchdog=0, submission_rate=0)
```

Connect the Ambassador to a Client.

Parameters

- **polyswarmd_addr** (*str*) – URL of polyswarmd you are referring to.
- **keyfile** (*str*) – Keyfile filename.
- **password** (*str*) – Password associated with Keyfile.
- **api_key** (*str*) – Your PolySwarm API key.
- **testing** (*int*) – Number of testing bounties to use.
- **insecure_transport** (*bool*) – Allow insecure transport such as HTTP?
- **chains** (*set (str)*) – Set of chains you are acting on.

Returns Ambassador instantiated with a Client.

Return type *AbstractAmbassador*

```
run (self)
```

Run the Client on all of our chains.

polyswarmclient.abstractarbiter

Module Contents

```
polyswarmclient.abstractarbiter.logger
```

```
polyswarmclient.abstractarbiter.MAX_STAKE_RETRIES = 10
```

```
class polyswarmclient.abstractarbiter.AbstractArbiter (client, testing=0, scan-
                                                         ner=None, chains=None,
                                                         artifact_types=None)
```

Bases: object

```
classmethod connect (cls, polyswarmd_addr, keyfile, password, api_key=None, test-
                    ing=0, insecure_transport=False, scanner=None, chains=None, arti-
                    fact_types=None)
```

Connect the Arbiter to a Client.

Parameters

- **polyswarmd_addr** (*str*) – URL of polyswarmd you are referring to.
- **keyfile** (*str*) – Keyfile filename.

- **password** (*str*) – Password associated with Keyfile.
- **api_key** (*str*) – Your PolySwarm API key.
- **testing** (*int*) – Number of testing bounties to use.
- **insecure_transport** (*bool*) – Allow insecure transport such as HTTP?
- **scanner** (*AbstractScanner*) – Scanner for scanning artifacts
- **chains** (*set (str)*) – Set of chains you are acting on.
- **artifact_types** (*list (ArtifactType)*) – List of artifact types you support

Returns Arbiter instantiated with a Client.

Return type *AbstractArbiter*

run (*self*)
Run the Client on the Arbiter's chains.

polyswarmclient.abstractmicroengine

Module Contents

polyswarmclient.abstractmicroengine.logger

```
class polyswarmclient.abstractmicroengine.AbstractMicroengine (client, testing=0,  
                                                                scanner=None,  
                                                                chains=None,  
                                                                arti-  
                                                                fact_types=None)
```

Bases: object

```
classmethod connect (cls, polyswarmd_addr, keyfile, password, api_key=None, test-  
                     ing=0, insecure_transport=False, scanner=None, chains=None, arti-  
                     fact_types=None)
```

Connect the Microengine to a Client.

Parameters

- **polyswarmd_addr** (*str*) – URL of polyswarmd you are referring to.
- **keyfile** (*str*) – Keyfile filename.
- **password** (*str*) – Password associated with Keyfile.
- **api_key** (*str*) – Your PolySwarm API key.
- **testing** (*int*) – Number of testing bounties to use.
- **insecure_transport** (*bool*) – Allow insecure transport such as HTTP?
- **scanner** (*Scanner*) – *Scanner* instance to use.
- **chains** (*set (str)*) – Set of chains you are acting on.
- **artifact_types** (*list (ArtifactType)*) – List of artifact types you support

Returns Microengine instantiated with a Client.

Return type *AbstractMicroengine*

run (*self*)
Run the *Client* on the Microengine's chains.

polyswarmclient.abstractscanner

Module Contents

polyswarmclient.abstractscanner.logger

class polyswarmclient.abstractscanner.ScanResult (*bit=False, verdict=False, confidence=1.0, metadata=""*)

Bases: object

Results from scanning one artifact

`__repr__` (*self*)

class polyswarmclient.abstractscanner.AbstractScanner

Bases: abc.ABC

Base *Scanner* class. To be overwritten with other scanning logic.

polyswarmclient.balanceclient

Module Contents

polyswarmclient.balanceclient.logger

class polyswarmclient.balanceclient.BalanceClient (*client*)

Bases: object

polyswarmclient.bloom

Module Contents

polyswarmclient.bloom.FILTER_BITS

polyswarmclient.bloom.HASH_FUNCS = 8

polyswarmclient.bloom.logger

polyswarmclient.bloom.w3

polyswarmclient.bloom.get_chunks_for_bloom (*value_hash*)

Bloom filter helper function. Turn a value hash into a series of chunks.

Parameters *value_hash* (*bytes*) – Hash of to be encoded into the Bloom filter.

Yields *chunk* (*bytes*) – Chunks of the value hash.

polyswarmclient.bloom.chunk_to_bloom_bits (*chunk*)

Bloom filter helper function. Turn a chunk into a series of actual bytes.

Parameters *chunk* (*bytes*) – Byte encoded chunk.

polyswarmclient.bloom.get_bloom_bits (*value*)

Bloom filter helper function. Get the Bloom bits of a given value.

Parameters *value* (*bytes*) – Value to be encoded into the Bloom filter.

```
class polyswarmclient.bloom.BloomFilter (value=0)
    Bases: numbers.Number

    value

    __int__ (self)

    add (self, value)
        Add a single byte value to the Bloom filter.

        Parameters value (bytes) – Byte encoded value to add to Bloom filter.

    extend (self, iterable)
        Add an iterable of byte values to the bloom filter.

        Parameters iterable (Iterable[bytes]) – Iterable of byte values.

    classmethod from_iterable (cls, iterable)
        Instantiate a bloom filter from a given iterable.

        Parameters iterable (Iterable[bytes]) – Iterable of byte values.

        Returns Instantiated BloomFilter.

        Return type BloomFilter

    __contains__ (self, value)

    __index__ (self)

    __combine (self, other)

    __or__ (self, other)

    __add__ (self, other)

    __icombine (self, other)

    __ior__ (self, other)

    __iadd__ (self, other)
```

`polyswarmclient.bountiesclient`

Module Contents

`polyswarmclient.bountiesclient.logger`

```
class polyswarmclient.bountiesclient.PostBountyTransaction (client, artifact_type,
                                                            amount, bounty_fee,
                                                            artifact_uri,
                                                            num_artifacts, duration, bloom)

    Bases: polyswarmclient.transaction.AbstractTransaction

    get_path (self)

    get_body (self)

    has_required_event (self, transaction_events)
```

```

class polyswarmclient.bountiesclient.PostAssertionTransaction (client,
                                                                bounty_guid,
                                                                bid,          asser-
                                                                tion_fee,   mask,
                                                                commitment)

    Bases: polyswarmclient.transaction.AbstractTransaction

    get_body (self)
    get_path (self)
    has_required_event (self, transaction_events)

class polyswarmclient.bountiesclient.RevealAssertionTransaction (client,
                                                                bounty_guid,
                                                                index,  nonce,
                                                                verdicts,
                                                                metadata)

    Bases: polyswarmclient.transaction.AbstractTransaction

    get_path (self)
    get_body (self)
    has_required_event (self, transaction_events)

class polyswarmclient.bountiesclient.PostVoteTransaction (client,      bounty_guid,
                                                           votes, valid_bloom)

    Bases: polyswarmclient.transaction.AbstractTransaction

    get_path (self)
    get_body (self)
    has_required_event (self, transaction_events)

class polyswarmclient.bountiesclient.SettleBountyTransaction (client,
                                                                bounty_guid)

    Bases: polyswarmclient.transaction.AbstractTransaction

    get_path (self)
    get_body (self)
    has_required_event (self, transaction_events)

class polyswarmclient.bountiesclient.BountiesClient (client)
    Bases: object

```

polyswarmclient.config

Module Contents

```

polyswarmclient.config.logger
polyswarmclient.config.validate_apikey (ctx, param, value)
Validates the API key passed in through click parameters

polyswarmclient.config.init_logging (loggers, log_format, loglevel=logging.WARNING)
Logic to support JSON logging.

class polyswarmclient.config.LoggerConfig (loggers,                                log_format,
                                           log_level=logging.WARNING)

```

LEVELS**configure** (*self*)**set_level** (*self*, *new_level*)**polyswarmclient.corpus****Module Contents****polyswarmclient.corpus.logger****polyswarmclient.corpus.MALICIOUS_BOOTSTRAP_URL****polyswarmclient.corpus.ARCHIVE_PASSWORD****class** polyswarmclient.corpus.DownloadToFileSystemCorpus (*base_dir=None*)

Bases: object

mal_path**benign_path****download_and_unpack** (*self*)**_get_pth_listing** (*self*, *p*)**get_malicious_file_list** (*self*)**get_benign_file_list** (*self*)**download_truth** (*self*)**polyswarmclient.events****Module Contents****polyswarmclient.events.logger****class** polyswarmclient.events.Callback

Bases: object

Abstract callback class which is the parent to a number of child callback classes to be used in different scenarios.

Note: Classes which extend *Callback* are expected to impliment the *run* method.

register (*self*, *f*)

Register a function to this Callback.

Parameters **f** (*function*) – Function to register.**remove** (*self*, *f*)

Remove a function previously assigned to this Callback.

Parameters **f** (*function*) – Function to remove.**class** polyswarmclient.events.OnRunCallbackBases: *polyswarmclient.events.Callback*

Called upon entering the event loop for the first time, use for initialization

class polyswarmclient.events.OnNewBlockCallback
Bases: *polyswarmclient.events.Callback*
Called upon receiving a new block, scheduled events triggered separately

class polyswarmclient.events.OnNewBountyCallback
Bases: *polyswarmclient.events.Callback*
Called upon receiving a new bounty

class polyswarmclient.events.OnNewAssertionCallback
Bases: *polyswarmclient.events.Callback*
Called upon receiving a new assertion

class polyswarmclient.events.OnRevealAssertionCallback
Bases: *polyswarmclient.events.Callback*
Called upon receiving a new assertion reveal

class polyswarmclient.events.OnNewVoteCallback
Bases: *polyswarmclient.events.Callback*
Called upon receiving a new arbiter vote

class polyswarmclient.events.OnQuorumReachedCallback
Bases: *polyswarmclient.events.Callback*
Called upon a bounty reaching quorum

class polyswarmclient.events.OnSettledBountyCallback
Bases: *polyswarmclient.events.Callback*
Called upon a bounty being settled

class polyswarmclient.events.OnInitializedChannelCallback
Bases: *polyswarmclient.events.Callback*
Called upon a channel being initialized

class polyswarmclient.events.Schedule
Bases: object
Generic Schedule class. Uses a PriorityQueue data structure to store Events.

empty (*self*)
Return True if the queue is empty.
Returns Is the queue empty.
Return type boolean

peek (*self*)
Return True if the queue is empty.
Returns Tuple at the front of the queue if the queue is full, else *None*.
Return type (block, event)

get (*self*)
Pop the lowest valued block in the queue.
Returns The lowest valued block in the PriorityQueue.
Return type (block, event)

put (*self*, *block*, *event*)

Add a tuple (block, event) to the PriorityQueue. Block signifies the priority of the event.

class polyswarmclient.events.**Event** (*guid*)

Bases: object

Generic Event class. Stores GUID and can compare for equality and order Events.

Parameters **guid** (*str*) – GUID of the event.

__eq__ (*self*, *other*)

__lt__ (*self*, *other*)

class polyswarmclient.events.**RevealAssertion** (*guid*, *index*, *nonce*, *verdicts*, *metadata*)

Bases: [polyswarmclient.events.Event](#)

An assertion scheduled to be publically revealed

Parameters

- **guid** (*str*) – GUID of the bounty being asserted on
- **index** (*int*) – Index of the assertion to reveal
- **nonce** (*str*) – Secret nonce used to reveal assertion
- **verdicts** (*List[bool]*) – List of verdicts for each artifact in the bounty
- **metadata** (*str*) – Optional metadata

class polyswarmclient.events.**OnRevealAssertionDueCallback**

Bases: [polyswarmclient.events.Callback](#)

Called when an assertion is needing to be revealed

class polyswarmclient.events.**VoteOnBounty** (*guid*, *votes*, *valid_bloom*)

Bases: [polyswarmclient.events.Event](#)

A scheduled vote from an arbiter :param guid: GUID of the bounty being voted on :type guid: str :param votes: List of votes for each artifact in the bounty :type votes: List[bool] :param valid_bloom: Is the bloom filter submitted with the bounty valid :type valid_bloom: bool

class polyswarmclient.events.**OnVoteOnBountyDueCallback**

Bases: [polyswarmclient.events.Callback](#)

Called when a bounty is needing to be voted on

class polyswarmclient.events.**SettleBounty** (*guid*)

Bases: [polyswarmclient.events.Event](#)

A bounty scheduled to be settled :param guid: GUID of the bounty being asserted on :type guid: str

class polyswarmclient.events.**OnSettleBountyDueCallback**

Bases: [polyswarmclient.events.Callback](#)

Called when a bounty is needing to be settled

polyswarmclient.log_formatter

Module Contents

class polyswarmclient.log_formatter.**ExtraTextFormatter**

Bases: logging.Formatter

Custom formatter that adds extra fields to the message string

format (*self, record*)

Takes a LogRecord and sets record.message = record.msg % record.args This one does some extra work. It searches the record dict for some extra keys we use in the client. (specified as extra= in the logger statement) If it finds one, it grabs the dict and adds an extra %s arg to record.msg, and the dict value to the record.args tuple.

class polyswarmclient.log_formatter.JSONFormatter

Bases: pythonjsonlogger.jsonlogger.JsonFormatter

Class to add custom JSON fields to our logger. Presently just adds a timestamp if one isn't present and the log level. INFO: <https://github.com/madzak/python-json-logger#customizing-fields>

add_fields (*self, log_record, record, message_dict*)

polyswarmclient.offersclient

Module Contents

polyswarmclient.offersclient.logger

class polyswarmclient.offersclient.OffersClient (*client*)

Bases: object

OffersClient to handle offers. Presently stores a given client and parameters.

polyswarmclient.parameters

Module Contents

class polyswarmclient.parameters.Parameters (*p*)

Bases: object

Trivial wrapper around a dict but protected via a RWLock to allow updates

polyswarmclient.producer

Module Contents

polyswarmclient.producer.logger

polyswarmclient.producer.KEY_TIMEOUT = 20

class polyswarmclient.producer.Producer (*client, redis_uri, queue, time_to_post*)

polyswarmclient.relayclient

Module Contents

polyswarmclient.relayclient.logger

class polyswarmclient.relayclient.RelayDepositTransaction (*client, amount*)

Bases: *polyswarmclient.transaction.AbstractTransaction*


```

    get_path (self)
    get_body (self)
    has_required_event (self, transaction_events)
class polyswarmclient.relayclient.RelayWithdrawTransaction (client, amount)
    Bases: polyswarmclient.transaction.AbstractTransaction
    get_path (self)
    get_body (self)
    has_required_event (self, transaction_events)
class polyswarmclient.relayclient.RelayClient (client)
    Bases: object

```

polyswarmclient.stakingclient

Module Contents

```

polyswarmclient.stakingclient.logger
class polyswarmclient.stakingclient.StakeDepositTransaction (client, amount)
    Bases: polyswarmclient.transaction.AbstractTransaction
    get_path (self)
    get_body (self)
    has_required_event (self, transaction_events)
class polyswarmclient.stakingclient.StakeWithdrawTransaction (client, amount)
    Bases: polyswarmclient.transaction.AbstractTransaction
    get_path (self)
    get_body (self)
    has_required_event (self, transaction_events)
class polyswarmclient.stakingclient.StakingClient (client)
    Bases: object

```

polyswarmclient.transaction

Module Contents

```

polyswarmclient.transaction.logger
polyswarmclient.transaction.LOG_MSG_ENGINE_TOO_SLOW = PLEASE REVIEW YOUR SCANNING LOGIC. B
class polyswarmclient.transaction.NonceManager (client, chain)
    Manages the nonce for some Ethereum chain
    all_finished (self)
        Check that all tasks have finished
    mark_update_nonce (self)
        Call this when the nonce is out of sync. This sets the update flag to true. The next acquire after being set
        will trigger an update

```

class polyswarmclient.transaction.**AbstractTransaction** (*client, verifiers*)

Used to verify and post groups of transactions that make up a specific action.

For instance, when approving some funds to move, and calling a contract function that will consumer them.

has_required_event (*self, transaction_events*)

Checks for existence of events in transaction logs, ensuring successful completion

Returns True if the required event was in the list, false otherwise

get_path (*self*)

Get the path of the route to build this transaction

Returns Polyswarmd path to get the transaction data

Return type str

get_body (*self*)

Build the payload to send to polyswarmd :returns: Dict payload

verify (*self, transactions*)

Check the given transactions against known expectations

Parameters **transactions** (*list*) –

Returns True if transactions match expectations. False otherwise

Return type (bool)

polyswarmclient.utils

Module Contents

polyswarmclient.utils.logger

polyswarmclient.utils.TASK_TIMEOUT = 1.0

polyswarmclient.utils.MAX_WAIT

polyswarmclient.utils.MAX_WORKERS = 4

polyswarmclient.utils.to_string (*value*)

polyswarmclient.utils.sha3_256 (*x*)

polyswarmclient.utils.sha3 (*seed*)

polyswarmclient.utils.int_to_bytes (*i*)

polyswarmclient.utils.int_from_bytes (*b*)

polyswarmclient.utils.bool_list_to_int (*bs*)

polyswarmclient.utils.int_to_bool_list (*i, expected_size*)

polyswarmclient.utils.guid_as_string (*guid*)

polyswarmclient.utils.calculate_commitment (*account, verdicts, nonce=None*)

polyswarmclient.utils.configure_event_loop ()

polyswarmclient.utils.asyncio_join ()

Gather all remaining tasks, assumes loop is not running

polyswarmclient.utils.asyncio_stop ()

Stop the main event loop

`polyswarmclient.utils.exit(exit_status)`

Exit the program entirely.

`polyswarmclient.utils.check_response(response)`

Check the status of responses from polyswarmd

Parameters `response` – Response dict parsed from JSON from polyswarmd

Returns True if successful else False

Return type (bool)

`polyswarmclient.utils.is_valid_ipfs_uri(ipfs_uri)`

Ensure that a given ipfs_uri is valid by checking length and base58 encoding.

Parameters `ipfs_uri` (*str*) – ipfs_uri to validate

Returns is this valid?

Return type bool

`polyswarmclient.verifiers`

Module Contents

`polyswarmclient.verifiers.logger`

`polyswarmclient.verifiers.UNKNOWN_PARAMETER = XXX`

class `polyswarmclient.verifiers.DecodedTransaction` (*to, value, data, abi, signature, parameters*)

This is a decoded representation of the transaction object returned by polyswarmd.

classmethod `from_transaction` (*cls, transaction, abi*)

Parse a transaction from data returned from polyswarmd.

Parameters

- **transaction** (*dict*) – Transaction to be simplified
- **abi** (*str, list[str]*) – ABI of the expected function call

Returns If valid, returns a SimplifiedTransaction

Return type *DecodedTransaction*

Raises **ValueError** – If invalid transaction is provided

`__repr__` (*self*)

class `polyswarmclient.verifiers.AbstractTransactionVerifier` (*parameters*)

Verifier is used to verify the details of a single transaction.

ABI = [' ', []]

verify (*self, transaction*)

Called when a list of transactions were returned from polyswarmd. This function will verify the transactions, and determines if the transactions are expected.

Parameters **transaction** – Transaction representation returned from polyswarmd

Returns True if valid and expected

`__repr__` (*self*)

```
class polyswarmclient.verifiers.NctApproveVerifier(amount)
    Bases: polyswarmclient.verifiers.AbstractTransactionVerifier
    ABI = ['approve', ['address', 'uint256']]
    verify(self, transaction)

class polyswarmclient.verifiers.NctTransferVerifier(amount)
    Bases: polyswarmclient.verifiers.AbstractTransactionVerifier
    ABI = ['transfer', ['address', 'uint256']]
    verify(self, transaction)

class polyswarmclient.verifiers.PostBountyVerifier(artifact_type, amount, artifact_uri, num_artifacts, duration, bloom)
    Bases: polyswarmclient.verifiers.AbstractTransactionVerifier
    ABI = ['postBounty', ['uint128', 'uint256', 'uint256', 'string', 'uint256', 'uint256']]
    verify(self, transaction)

class polyswarmclient.verifiers.PostAssertionVerifier(bounty_guid, bid, mask, commitment)
    Bases: polyswarmclient.verifiers.AbstractTransactionVerifier
    ABI = ['postAssertion', ['uint128', 'uint256', 'uint256', 'uint256']]
    verify(self, transaction)

class polyswarmclient.verifiers.RevealAssertionVerifier(bounty_guid, index, nonce, verdicts, metadata)
    Bases: polyswarmclient.verifiers.AbstractTransactionVerifier
    ABI = ['revealAssertion', ['uint128', 'uint256', 'uint256', 'uint256', 'string']]
    verify(self, transaction)

class polyswarmclient.verifiers.PostVoteVerifier(bounty_guid, votes, valid_bloom)
    Bases: polyswarmclient.verifiers.AbstractTransactionVerifier
    ABI = ['voteOnBounty', ['uint128', 'uint256', 'bool']]
    verify(self, transaction)

class polyswarmclient.verifiers.SettleBountyVerifier(bounty_guid)
    Bases: polyswarmclient.verifiers.AbstractTransactionVerifier
    ABI = ['settleBounty', ['uint128']]
    verify(self, transaction)

class polyswarmclient.verifiers.StakingDepositVerifier(amount)
    Bases: polyswarmclient.verifiers.AbstractTransactionVerifier
    ABI = ['deposit', ['uint256']]
    verify(self, transaction)

class polyswarmclient.verifiers.StakingWithdrawVerifier(amount)
    Bases: polyswarmclient.verifiers.AbstractTransactionVerifier
    ABI = ['withdraw', ['uint256']]
    verify(self, transaction)
```

1.3.2 Package Contents

```

class polyswarmclient.BalanceClient (client)
    Bases: object

class polyswarmclient.BountiesClient (client)
    Bases: object

class polyswarmclient.StakingClient (client)
    Bases: object

class polyswarmclient.OffersClient (client)
    Bases: object
    OffersClient to handle offers. Presently stores a given client and parameters.

class polyswarmclient.RelayClient (client)
    Bases: object

class polyswarmclient.NonceManager (client, chain)
    Manages the nonce for some Ethereum chain

    all_finished (self)
        Check that all tasks have finished

    mark_update_nonce (self)
        Call this when the nonce is out of sync. This sets the update flag to true. The next acquire after being set
        will trigger an update

polyswarmclient.asyncio_join()
Gather all remaining tasks, assumes loop is not running

polyswarmclient.asyncio_stop()
Stop the main event loop

polyswarmclient.configure_event_loop()

polyswarmclient.exit (exit_status)
Exit the program entirely.

polyswarmclient.MAX_WAIT

polyswarmclient.check_response (response)
Check the status of responses from polyswarmd

    Parameters response – Response dict parsed from JSON from polyswarmd

    Returns True if successful else False

    Return type (bool)

polyswarmclient.is_valid_ipfs_uri (ipfs_uri)
Ensure that a given ipfs_uri is valid by checking length and base58 encoding.

    Parameters ipfs_uri (str) – ipfs_uri to validate

    Returns is this valid?

    Return type bool

polyswarmclient.logger

polyswarmclient.w3

polyswarmclient.REQUEST_TIMEOUT = 300.0

```

```
polyswarmclient.MAX_ARTIFACTS = 256
```

```
polyswarmclient.RATE_LIMIT_SLEEP = 2.0
```

```
class polyswarmclient.Client (polyswarmd_addr,    keyfile,    password,    api_key=None,
                             tx_error_fatal=False, insecure_transport=False)
```

Bases: object

Client to connected to a Ethereum wallet as well as a polyswarmd instance.

Parameters

- **polyswarmd_addr** (*str*) – URI of polyswarmd you are referring to.
- **keyfile** (*str*) – Keyfile filename.
- **password** (*str*) – Password associated with keyfile.
- **api_key** (*str*) – Your PolySwarm API key.
- **tx_error_fatal** (*bool*) – Transaction errors are fatal and exit the program
- **insecure_transport** (*bool*) – Allow insecure transport such as HTTP?

```
class __GetArtifacts (client, ipfs_uri, api_key=None)
```

Bases: object

```
run (self, chains=None)
```

Run the main event loop

Parameters **chains** (*set (str)*) – Set of chains to operate on. Defaults to { ‘home’, ‘side’ }

```
sign_transactions (self, transactions)
```

Sign a set of transactions

Parameters **transactions** (*List [Transaction]*) – The transactions to sign

Returns The signed transactions

Return type List[Transaction]

```
static to_wei (amount, unit='ether')
```

```
static from_wei (amount, unit='ether')
```

```
get_artifacts (self, ipfs_uri, api_key=None)
```

Get an iterator to return artifacts.

Parameters

- **ipfs_uri** (*str*) – URI where artificats are located
- **api_key** (*str*) – Override default API key

Returns `__GetArtifacts` iterator

```
schedule (self, expiration, event, chain)
```

Schedule an event to execute on a particular block

Parameters

- **expiration** (*int*) – Which block to execute on
- **event** (*Event*) – Event to trigger on expiration block
- **chain** (*str*) – Which chain to operate on

1.4 ambassador

1.4.1 Submodules

`ambassador.__main__`

Module Contents

`ambassador.__main__.logger`

`ambassador.__main__.choose_backend(backend)`

Resolves amabassador name string to implementation

Parameters `backend` (*str*) – Name of the backend to load, either one of the predefined implementations or the name of a module to load (module:ClassName syntax or default of module:Ambassador)

Returns Ambassador class of the selected implementation

Return type (Class)

Raises (**Exception**) – If backend is not found

`ambassador.__main__.main(log, client_log, polyswarmd_addr, keyfile, password, api_key, backend, testing, insecure_transport, chains, watchdog, log_format, submission_rate)`

Entrypoint for the ambassador driver

Parameters

- **log** (*str*) – Logging level for all app logs
- **client_log** (*str*) – Logging level for all polyswarmclient logs
- **polyswarmd_addr** (*str*) – Address of polyswarmd
- **keyfile** (*str*) – Path to private key file to use to sign transactions
- **password** (*str*) – Password to decrypt the encrypted private key
- **backend** (*str*) – Backend implementation to use
- **api_key** (*str*) – API key to use with polyswarmd
- **testing** (*int*) – Mode to process N bounties then exit (optional)
- **insecure_transport** (*bool*) – Connect to polyswarmd without TLS
- **chains** (*List[str]*) – Chain(s) to operate on
- **watchdog** (*int*) – Number of blocks to look back and see if bounties are being submitted
- **log_format** (*str*) – Format to output logs in. *text* or *json*

`ambassador.eicar`

Module Contents

`ambassador.eicar.logger`

`ambassador.eicar.EICAR`

```
ambassador.eicar.NOT_EICAR = this is not malicious
ambassador.eicar.ARTIFACTS = [None, None]
ambassador.eicar.BOUNTY_TEST_DURATION_BLOCKS

class ambassador.eicar.Ambassador(client, testing=0, chains=None, watchdog=0, submission_rate=30)
    Bases: polyswarmclient.abstractambassador.AbstractAmbassador
    Ambassador which submits the EICAR test file
```

ambassador.filesystem

Module Contents

```
ambassador.filesystem.logger
ambassador.filesystem.ARTIFACT_DIRECTORY
ambassador.filesystem.ARTIFACT_BLACKLIST
ambassador.filesystem.BOUNTY_TEST_DURATION_BLOCKS

class ambassador.filesystem.Ambassador(client, testing=0, chains=None, watchdog=0, submission_rate=30)
    Bases: polyswarmclient.abstractambassador.AbstractAmbassador
    Ambassador which submits artifacts from a directory
```

1.5 microengine

1.5.1 Submodules

microengine.__main__

Module Contents

```
microengine.__main__.logger
microengine.__main__.choose_backend(backend)
Resolves microengine name string to implementation
```

Parameters **backend** (*str*) – Name of the backend to load, either one of the predefined implementations or the name of a module to load (module:ClassName syntax or default of module:Microengine)

Returns Microengine class of the selected implementation

Return type (Class)

Raises (**Exception**) – If backend is not found

```
microengine.__main__.main(log, client_log, polyswarmd_addr, keyfile, password, api_key, backend, testing, insecure_transport, chains, log_format, artifact_type)
```

Entrypoint for the microengine driver

Parameters

- **log** (*str*) – Logging level for all app logs

- **client_log** (*str*) – Logging level for all polyswarmclient logs
- **polyswarmd_addr** (*str*) – Address of polyswarmd
- **keyfile** (*str*) – Path to private key file to use to sign transactions
- **password** (*str*) – Password to decrypt the encrypted private key
- **backend** (*str*) – Backend implementation to use
- **api_key** (*str*) – API key to use with polyswarmd
- **testing** (*int*) – Mode to process N bounties then exit (optional)
- **insecure_transport** (*bool*) – Connect to polyswarmd without TLS
- **chains** (*list[str]*) – List of chains on which to scan artifacts
- **log_format** (*str*) – Format to output logs in. *text* or *json*
- **artifact_type** (*list[str]*) – List of artifact types to scan

microengine.clamav

Module Contents

microengine.clamav.logger

microengine.clamav.CLAMD_HOST

microengine.clamav.CLAMD_PORT

microengine.clamav.CLAMD_TIMEOUT = 30.0

class microengine.clamav.Scanner

Bases: *polyswarmclient.abstractscanner.AbstractScanner*

class microengine.clamav.Microengine (*client, testing=0, scanner=None, chains=None, artifact_types=None*)

Bases: *polyswarmclient.abstractmicroengine.AbstractMicroengine*

Microengine which scans samples through clamd.

Parameters

- **client** (*Client*) – Client to use
- **testing** (*int*) – How many test bounties to respond to
- **chains** (*set[str]*) – Chain(s) to operate on

microengine.eicar

Module Contents

microengine.eicar.logger

microengine.eicar.EICAR

class microengine.eicar.Scanner

Bases: *polyswarmclient.abstractscanner.AbstractScanner*

class `microengine.eicar.Microengine` (*client*, *testing=0*, *scanner=None*, *chains=None*, *artifact_types=None*)

Bases: `polyswarmclient.abstractmicroengine.AbstractMicroengine`

Microengine which tests for the EICAR test file.

Parameters

- **client** (*Client*) – Client to use
- **testing** (*int*) – How many test bounties to respond to
- **chains** (*set[str]*) – Chain(s) to operate on

`microengine.multi`

Module Contents

`microengine.multi.logger`

`microengine.multi.BACKENDS`

class `microengine.multi.Scanner`

Bases: `polyswarmclient.abstractscanner.AbstractScanner`

class `microengine.multi.Microengine` (*client*, *testing=0*, *scanner=None*, *chains=None*, *artifact_types=None*)

Bases: `polyswarmclient.abstractmicroengine.AbstractMicroengine`

Microengine which aggregates multiple sub-microengines

`microengine.producer`

Module Contents

`microengine.producer.logger`

`microengine.producer.REDIS_ADDR`

`microengine.producer.QUEUE`

`microengine.producer.TIME_TO_POST_ASSERTION = 4`

`microengine.producer.KEY_TIMEOUT = 20`

class `microengine.producer.Microengine` (*client*, *testing=0*, *scanner=None*, *chains=None*, *artifact_types=None*)

Bases: `polyswarmclient.abstractmicroengine.AbstractMicroengine`

`microengine.scratch`

Module Contents

`microengine.scratch.logger`

class `microengine.scratch.Scanner`

Bases: `polyswarmclient.abstractscanner.AbstractScanner`

class `microengine.scratch.Microengine` (*client, testing=0, scanner=None, chains=None, artifact_types=None*)

Bases: `polyswarmclient.abstractmicroengine.AbstractMicroengine`

Scratch microengine is the same as the default behavior.

Parameters

- **client** (*Client*) – Client to use
- **testing** (*int*) – How many test bounties to respond to
- **chains** (*set[str]*) – Chain(s) to operate on

`microengine.yara`

Module Contents

`microengine.yara.logger`

`microengine.yara.RULES_DIR`

class `microengine.yara.Scanner`

Bases: `polyswarmclient.abstractscanner.AbstractScanner`

class `microengine.yara.Microengine` (*client, testing=0, scanner=None, chains=None, artifact_types=None*)

Bases: `polyswarmclient.abstractmicroengine.AbstractMicroengine`

Microengine which matches samples against yara rules

1.6 arbiter

1.6.1 Subpackages

`arbiter.verbatimdb`

Submodules

`arbiter.verbatimdb.__main__`

Module Contents

`arbiter.verbatimdb.__main__.main` (*malicious, benign, output, log_format*)

`arbiter.verbatimdb.db`

Module Contents

`arbiter.verbatimdb.db.generate_db` (*db_file, malicious_dir, benign_dir*)

`arbiter.verbatimdb.db.insert` (*cursor, path, result*)

1.6.2 Submodules

`arbiter.__main__`

Module Contents

`arbiter.__main__.logger`

`arbiter.__main__.choose_backend(backend)`

Resolves arbiter name string to implementation

Parameters `backend` (*str*) – Name of the backend to load, either one of the predefined implementations or the name of a module to load (module:ClassName syntax or default of module:Arbiter)

Returns Arbiter class of the selected implementation

Return type (Class)

Raises (**Exception**) – If backend is not found

`arbiter.__main__.main(log, client_log, polyswarmd_addr, keyfile, password, api_key, backend, testing, insecure_transport, chains, log_format, artifact_type)`

Entrypoint for the arbiter driver

Parameters

- **log** (*str*) – Logging level for all app logs
- **client_log** (*str*) – Logging level for all polyswarmclient logs
- **polyswarmd_addr** (*str*) – Address of polyswarmd
- **keyfile** (*str*) – Path to private key file to use to sign transactions
- **password** (*str*) – Password to decrypt the encrypted private key
- **backend** (*str*) – Backend implementation to use
- **api_key** (*str*) – API key to use with polyswarmd
- **testing** (*int*) – Mode to process N bounties then exit (optional)
- **insecure_transport** (*bool*) – Connect to polyswarmd without TLS
- **chains** (*List[str]*) – Chain(s) to operate on
- **log_format** (*str*) – Format to output logs in. *text* or *json*
- **artifact_type** (*list[str]*) – List of artifact types to scan

`arbiter.clamav`

Module Contents

`arbiter.clamav.logger`

class `arbiter.clamav.Arbiter` (*client*, *testing=0*, *scanner=None*, *chains=None*, *artifact_types=None*)

Bases: `polyswarmclient.abstractarbiter.AbstractArbiter`

Arbiter which scans samples through clamd.

Re-uses the scanner from the clamav microengine

Parameters

- **client** (*Client*) – Client to use
- **testing** (*int*) – How many test bounties to respond to
- **chains** (*set[str]*) – Chain(s) to operate on

arbiter.producer

Module Contents

arbiter.producer.logger

arbiter.producer.REDIS_ADDR

arbiter.producer.QUEUE

arbiter.producer.TIME_TO_POST_VOTE = 4

class arbiter.producer.Arbitrator(*client*, *testing=0*, *scanner=None*, *chains=None*, *artifact_types=None*)

Bases: *polyswarmclient.abstractarbiter.AbstractArbitrator*

arbiter.verbatim

Module Contents

arbiter.verbatim.logger

arbiter.verbatim.ARTIFACT_DIRECTORY

arbiter.verbatim.EICAR

class arbiter.verbatim.Arbitrator(*client*, *testing=0*, *scanner=None*, *chains=None*, *artifact_types=None*)

Bases: *polyswarmclient.abstractarbiter.AbstractArbitrator*

Arbitrator which matches hashes to a database of known samples

1.7 worker

1.7.1 Submodules

worker.__main__

Module Contents

worker.__main__.logger

worker.__main__.choose_backend(*backend*)

Resolves scanner name string to implementation

Parameters

- **backend** (*str*) – Name of the backend to load, either one of the predefined implementations or the name of a module

- **load**(*to*) –
- (**module** – ClassName syntax or default of module:Scanner)

Returns Scanner class of the selected implementation

Return type (Class)

Raises (**Exception**) – If backend is not found

`worker.__main__.main(log, client_log, redis_addr, queue, backend, tasks, download_limit, scan_limit, api_key, testing, log_format)`

Entrypoint for the worker driver

Parameters

- **log**(*str*) – Logging level for all app logs
- **client_log**(*str*) – Logging level for all polyswarmclient logs
- **redis_addr**(*str*) – Address of redis
- **backend**(*str*) – Backend implementation to use
- **queue**(*str*) – Name of queue to listen on
- **tasks**(*int*) – Number of simultaneous tasks this worker runs
- **download_limit**(*int*) – Number of simultaneous downloads this worker can handle
- **scan_limit**(*int*) – Number of simultaneous scans this worker can handle
- **api_key**(*str*) – API key to use with polyswarmd
- **testing**(*int*) – Mode to process N bounties then exit (optional)
- **log_format**(*str*) – Format to output logs in. *text* or *json*

1.7.2 Package Contents

`worker.logger`

`worker.REQUEST_TIMEOUT = 5.0`

exception `worker.ApiKeyException`

Bases: `Exception`

exception `worker.ExpiredException`

Bases: `Exception`

class `worker.Worker(redis_addr, queue, task_count=1, download_limit=1, scan_limit=1, api_key=None, testing=0, scanner=None)`

Bases: `object`

run(*self*)

`polyswarm-client` is a convenient library on which to build PolySwarm market participants.

Here you'll find auto-generated documentation based on the `polyswarm-client` source code.

Consulting these low level details is unnecessary for developing a successful PolySwarm participant. Most users will instead want to consult the [PolySwarm Documentation](#).

Developers interested in low level `polyswarm-client` details can do so by navigating to “API Reference” on the left.

PYTHON MODULE INDEX

a

- `ambassador`, 19
- `ambassador.__main__`, 19
- `ambassador.eicar`, 19
- `ambassador.filesystem`, 20
- `arbiter`, 23
- `arbiter.__main__`, 24
- `arbiter.clamav`, 24
- `arbiter.producer`, 25
- `arbiter.verbatim`, 25
- `arbiter.verbatimdb`, 23
- `arbiter.verbatimdb.__main__`, 23
- `arbiter.verbatimdb.db`, 23

b

- `balancemanager`, 1
- `balancemanager.__main__`, 1

c

- `conftest`, 1

m

- `microengine`, 20
- `microengine.__main__`, 20
- `microengine.clamav`, 21
- `microengine.eicar`, 21
- `microengine.multi`, 22
- `microengine.producer`, 22
- `microengine.scratch`, 22
- `microengine.yara`, 23

p

- `polyswarmclient`, 3
- `polyswarmclient.abstractambassador`, 3
- `polyswarmclient.abstractarbiter`, 4
- `polyswarmclient.abstractmicroengine`, 5
- `polyswarmclient.abstractscanner`, 6
- `polyswarmclient.balanceclient`, 6
- `polyswarmclient.bloom`, 6
- `polyswarmclient.bountiesclient`, 7
- `polyswarmclient.config`, 8
- `polyswarmclient.corpus`, 9

- `polyswarmclient.events`, 9
- `polyswarmclient.log_formatter`, 11
- `polyswarmclient.offersclient`, 12
- `polyswarmclient.parameters`, 12
- `polyswarmclient.producer`, 12
- `polyswarmclient.relayclient`, 12
- `polyswarmclient.stakingclient`, 13
- `polyswarmclient.transaction`, 13
- `polyswarmclient.utils`, 14
- `polyswarmclient.verifiers`, 15

w

- `worker`, 25
- `worker.__main__`, 25

Symbols

`__add__()` (*polyswarmclient.bloom.BloomFilter method*), 7
`__contains__()` (*polyswarmclient.bloom.BloomFilter method*), 7
`__eq__()` (*polyswarmclient.events.Event method*), 11
`__iadd__()` (*polyswarmclient.bloom.BloomFilter method*), 7
`__index__()` (*polyswarmclient.bloom.BloomFilter method*), 7
`__int__()` (*polyswarmclient.bloom.BloomFilter method*), 7
`__ior__()` (*polyswarmclient.bloom.BloomFilter method*), 7
`__lt__()` (*polyswarmclient.events.Event method*), 11
`__or__()` (*polyswarmclient.bloom.BloomFilter method*), 7
`__repr__()` (*polyswarmclient.abstractambassador.QueuedBounty method*), 4
`__repr__()` (*polyswarmclient.abstractscanner.ScanResult method*), 6
`__repr__()` (*polyswarmclient.verifiers.AbstractTransactionVerifier method*), 15
`__repr__()` (*polyswarmclient.verifiers.DecodedTransaction method*), 15
`_combine()` (*polyswarmclient.bloom.BloomFilter method*), 7
`_get_pth_listing()` (*polyswarmclient.corpus.DownloadToFileSystemCorpus method*), 9
`_icombine()` (*polyswarmclient.bloom.BloomFilter method*), 7

A

ABI (*polyswarmclient.verifiers.AbstractTransactionVerifier attribute*), 15
ABI (*polyswarmclient.verifiers.NctApproveVerifier attribute*), 16

ABI (*polyswarmclient.verifiers.NctTransferVerifier attribute*), 16
ABI (*polyswarmclient.verifiers.PostAssertionVerifier attribute*), 16
ABI (*polyswarmclient.verifiers.PostBountyVerifier attribute*), 16
ABI (*polyswarmclient.verifiers.PostVoteVerifier attribute*), 16
ABI (*polyswarmclient.verifiers.RevealAssertionVerifier attribute*), 16
ABI (*polyswarmclient.verifiers.SettleBountyVerifier attribute*), 16
ABI (*polyswarmclient.verifiers.StakingDepositVerifier attribute*), 16
ABI (*polyswarmclient.verifiers.StakingWithdrawVerifier attribute*), 16
AbstractAmbassador (*class in polyswarmclient.abstractambassador*), 4
AbstractArbiter (*class in polyswarmclient.abstractarbiter*), 4
AbstractMicroengine (*class in polyswarmclient.abstractmicroengine*), 5
AbstractScanner (*class in polyswarmclient.abstractscanner*), 6
AbstractTransaction (*class in polyswarmclient.transaction*), 13
AbstractTransactionVerifier (*class in polyswarmclient.verifiers*), 15
`add()` (*polyswarmclient.bloom.BloomFilter method*), 7
`add_fields()` (*polyswarmclient.log_formatter.JSONFormatter method*), 12
`all_finished()` (*polyswarmclient.NonceManager method*), 17
`all_finished()` (*polyswarmclient.transaction.NonceManager method*), 13
Ambassador (*class in ambassador.eicar*), 20
Ambassador (*class in ambassador.filesystem*), 20
ambassador (*module*), 19
ambassador.__main__ (*module*), 19
ambassador.eicar (*module*), 19

ambassador.filesystem (module), 20
 ApiKeyException, 26
 Arbiter (class in arbiter.clamav), 24
 Arbiter (class in arbiter.producer), 25
 Arbiter (class in arbiter.verbatim), 25
 arbiter (module), 23
 arbiter.__main__ (module), 24
 arbiter.clamav (module), 24
 arbiter.producer (module), 25
 arbiter.verbatim (module), 25
 arbiter.verbatimdb (module), 23
 arbiter.verbatimdb.__main__ (module), 23
 arbiter.verbatimdb.db (module), 23
 ARCHIVE_PASSWORD (in module polyswarm-client.corpus), 9
 ARTIFACT_BLACKLIST (in module ambassador.filesystem), 20
 ARTIFACT_DIRECTORY (in module ambassador.filesystem), 20
 ARTIFACT_DIRECTORY (in module arbiter.verbatim), 25
 ARTIFACTS (in module ambassador.eicar), 20
 asyncio_join() (in module polyswarmclient), 17
 asyncio_join() (in module polyswarmclient.utils), 14
 asyncio_stop() (in module polyswarmclient), 17
 asyncio_stop() (in module polyswarmclient.utils), 14

B

BACKENDS (in module microengine.multi), 22
 BalanceClient (class in polyswarmclient), 17
 BalanceClient (class in polyswarm-client.balanceclient), 6
 BalanceManager (class in balancemanager), 3
 balancemanager (module), 1
 balancemanager.__main__ (module), 1
 benign_path (polyswarm-client.corpus.DownloadToFileSystemCorpus attribute), 9
 BLOCK_DIVISOR (in module polyswarm-client.abstractambassador), 3
 BloomFilter (class in polyswarmclient.bloom), 6
 bool_list_to_int() (in module polyswarm-client.utils), 14
 BountiesClient (class in polyswarmclient), 17
 BountiesClient (class in polyswarm-client.bountiesclient), 8
 BOUNTY_QUEUE_SIZE (in module polyswarm-client.abstractambassador), 3
 BOUNTY_TEST_DURATION_BLOCKS (in module ambassador.eicar), 20
 BOUNTY_TEST_DURATION_BLOCKS (in module ambassador.filesystem), 20

C

calculate_commitment() (in module polyswarm-client.utils), 14
 Callback (class in polyswarmclient.events), 9
 check_response() (in module polyswarmclient), 17
 check_response() (in module polyswarm-client.utils), 15
 choose_backend() (in module ambassador.__main__), 19
 choose_backend() (in module arbiter.__main__), 24
 choose_backend() (in module microengine.__main__), 20
 choose_backend() (in module worker.__main__), 25
 chunk_to_bloom_bits() (in module polyswarm-client.bloom), 6
 CLAMD_HOST (in module microengine.clamav), 21
 CLAMD_PORT (in module microengine.clamav), 21
 CLAMD_TIMEOUT (in module microengine.clamav), 21
 cli() (in module balancemanager.__main__), 1
 Client (class in polyswarmclient), 18
 Client.__GetArtifacts (class in polyswarm-client), 18
 configure() (polyswarmclient.config.LoggerConfig method), 9
 configure_event_loop() (in module polyswarm-client), 17
 configure_event_loop() (in module polyswarm-client.utils), 14
 conftest (module), 1
 connect() (polyswarm-client.abstractambassador.AbstractAmbassador class method), 4
 connect() (polyswarm-client.abstractarbiter.AbstractArbiter class method), 4
 connect() (polyswarm-client.abstractmicroengine.AbstractMicroengine class method), 5
 convert() (in module balancemanager), 3

D

DecodedTransaction (class in polyswarm-client.verifiers), 15
 Deposit (class in balancemanager), 3
 deposit() (in module balancemanager.__main__), 1
 download_and_unpack() (polyswarm-client.corpus.DownloadToFileSystemCorpus method), 9
 download_truth() (polyswarm-client.corpus.DownloadToFileSystemCorpus method), 9
 DownloadToFileSystemCorpus (class in polyswarmclient.corpus), 9

E

EICAR (in module *ambassador.eicar*), 19
 EICAR (in module *arbiter.verbatim*), 25
 EICAR (in module *microengine.eicar*), 21
 empty() (polyswarmclient.events.Schedule method), 10
 Event (class in polyswarmclient.events), 11
 exit() (in module polyswarmclient), 17
 exit() (in module polyswarmclient.utils), 14
 ExpiredException, 26
 extend() (polyswarmclient.bloom.BloomFilter method), 7
 ExtraTextFormatter (class in polyswarmclient.log_formatter), 11

F

FILTER_BITS (in module polyswarmclient.bloom), 6
 format() (polyswarmclient.log_formatter.ExtraTextFormatter method), 12
 from_iterable() (polyswarmclient.bloom.BloomFilter class method), 7
 from_transaction() (polyswarmclient.verifiers.DecodedTransaction class method), 15
 from_wei() (polyswarmclient.Client static method), 18

G

generate_db() (in module *arbiter.verbatimdb.db*), 23
 get() (polyswarmclient.events.Schedule method), 10
 get_artifacts() (polyswarmclient.Client method), 18
 get_benign_file_list() (polyswarmclient.corpus.DownloadToFileSystemCorpus method), 9
 get_bloom_bits() (in module polyswarmclient.bloom), 6
 get_body() (polyswarmclient.bountiesclient.PostAssertionTransaction method), 8
 get_body() (polyswarmclient.bountiesclient.PostBountyTransaction method), 7
 get_body() (polyswarmclient.bountiesclient.PostVoteTransaction method), 8
 get_body() (polyswarmclient.bountiesclient.RevealAssertionTransaction method), 8
 get_body() (polyswarmclient.bountiesclient.SettleBountyTransaction method), 8

get_body() (polyswarmclient.relayclient.RelayDepositTransaction method), 13
 get_body() (polyswarmclient.relayclient.RelayWithdrawTransaction method), 13
 get_body() (polyswarmclient.stakingclient.StakeDepositTransaction method), 13
 get_body() (polyswarmclient.stakingclient.StakeWithdrawTransaction method), 13
 get_body() (polyswarmclient.transaction.AbstractTransaction method), 14
 get_chunks_for_bloom() (in module polyswarmclient.bloom), 6
 get_malicious_file_list() (polyswarmclient.corpus.DownloadToFileSystemCorpus method), 9
 get_path() (polyswarmclient.bountiesclient.PostAssertionTransaction method), 8
 get_path() (polyswarmclient.bountiesclient.PostBountyTransaction method), 7
 get_path() (polyswarmclient.bountiesclient.PostVoteTransaction method), 8
 get_path() (polyswarmclient.bountiesclient.RevealAssertionTransaction method), 8
 get_path() (polyswarmclient.bountiesclient.SettleBountyTransaction method), 8
 get_path() (polyswarmclient.relayclient.RelayDepositTransaction method), 12
 get_path() (polyswarmclient.relayclient.RelayWithdrawTransaction method), 13
 get_path() (polyswarmclient.stakingclient.StakeDepositTransaction method), 13
 get_path() (polyswarmclient.stakingclient.StakeWithdrawTransaction method), 13
 get_path() (polyswarmclient.transaction.AbstractTransaction method), 14
 guid_as_string() (in module polyswarmclient.utils), 14

H

`has_required_event()` (*polyswarm-client.bountiesclient.PostAssertionTransaction method*), 8

`has_required_event()` (*polyswarm-client.bountiesclient.PostBountyTransaction method*), 7

`has_required_event()` (*polyswarm-client.bountiesclient.PostVoteTransaction method*), 8

`has_required_event()` (*polyswarm-client.bountiesclient.RevealAssertionTransaction method*), 8

`has_required_event()` (*polyswarm-client.bountiesclient.SettleBountyTransaction method*), 8

`has_required_event()` (*polyswarm-client.relayclient.RelayDepositTransaction method*), 13

`has_required_event()` (*polyswarm-client.relayclient.RelayWithdrawTransaction method*), 13

`has_required_event()` (*polyswarm-client.stakingclient.StakeDepositTransaction method*), 13

`has_required_event()` (*polyswarm-client.stakingclient.StakeWithdrawTransaction method*), 13

`has_required_event()` (*polyswarm-client.transaction.AbstractTransaction method*), 14

`HASH_FUNCS` (*in module polyswarmclient.bloom*), 6

I

`init_logging()` (*in module polyswarmclient.config*), 8

`insert()` (*in module arbiter.verbatimdb.db*), 23

`int_from_bytes()` (*in module polyswarm-client.utils*), 14

`int_to_bool_list()` (*in module polyswarm-client.utils*), 14

`int_to_bytes()` (*in module polyswarmclient.utils*), 14

`is_valid_ipfs_uri()` (*in module polyswarm-client*), 17

`is_valid_ipfs_uri()` (*in module polyswarm-client.utils*), 15

J

`JSONFormatter` (*class in polyswarm-client.log_formatter*), 12

K

`KEY_TIMEOUT` (*in module microengine.producer*), 22

`KEY_TIMEOUT` (*in module polyswarmclient.producer*), 12

L

`LEVELS` (*polyswarmclient.config.LoggerConfig attribute*), 8

`LOG_MSG_ENGINE_TOO_SLOW` (*in module polyswarmclient.transaction*), 13

`logger` (*in module ambassador.__main__*), 19

`logger` (*in module ambassador.eicar*), 19

`logger` (*in module ambassador.filesystem*), 20

`logger` (*in module arbiter.__main__*), 24

`logger` (*in module arbiter.clamav*), 24

`logger` (*in module arbiter.producer*), 25

`logger` (*in module arbiter.verbatim*), 25

`logger` (*in module balancemanager*), 3

`logger` (*in module balancemanager.__main__*), 1

`logger` (*in module microengine.__main__*), 20

`logger` (*in module microengine.clamav*), 21

`logger` (*in module microengine.eicar*), 21

`logger` (*in module microengine.multi*), 22

`logger` (*in module microengine.producer*), 22

`logger` (*in module microengine.scratch*), 22

`logger` (*in module microengine.yara*), 23

`logger` (*in module polyswarmclient*), 17

`logger` (*in module polyswarm-client.abstractambassador*), 3

`logger` (*in module polyswarmclient.abstractarbiter*), 4

`logger` (*in module polyswarm-client.abstractmicroengine*), 5

`logger` (*in module polyswarmclient.abstractscanner*), 6

`logger` (*in module polyswarmclient.balanceclient*), 6

`logger` (*in module polyswarmclient.bloom*), 6

`logger` (*in module polyswarmclient.bountiesclient*), 7

`logger` (*in module polyswarmclient.config*), 8

`logger` (*in module polyswarmclient.corpus*), 9

`logger` (*in module polyswarmclient.events*), 9

`logger` (*in module polyswarmclient.offersclient*), 12

`logger` (*in module polyswarmclient.producer*), 12

`logger` (*in module polyswarmclient.relayclient*), 12

`logger` (*in module polyswarmclient.stakingclient*), 13

`logger` (*in module polyswarmclient.transaction*), 13

`logger` (*in module polyswarmclient.utils*), 14

`logger` (*in module polyswarmclient.verifiers*), 15

`logger` (*in module worker*), 26

`logger` (*in module worker.__main__*), 25

`LoggerConfig` (*class in polyswarmclient.config*), 8

M

`main()` (*in module ambassador.__main__*), 19

`main()` (*in module arbiter.__main__*), 24

`main()` (*in module arbiter.verbatimdb.__main__*), 23

`main()` (*in module microengine.__main__*), 20

`main()` (*in module worker.__main__*), 26

maintain() (in module *balancemanager.__main__*), 2
 Maintainer (class in *balancemanager*), 3
 mal_path (polyswarm-client.corpus.DownloadToFileSystemCorpus attribute), 9
 MALICIOUS_BOOTSTRAP_URL (in module *polyswarmclient.corpus*), 9
 mark_update_nonce() (polyswarm-client.NonceManager method), 17
 mark_update_nonce() (polyswarm-client.transaction.NonceManager method), 13
 MAX_ARTIFACTS (in module *polyswarmclient*), 17
 MAX_BOUNTIES_IN_FLIGHT (in module *polyswarm-client.abstractambassador*), 3
 MAX_BOUNTIES_PER_BLOCK (in module *polyswarm-client.abstractambassador*), 3
 MAX_STAKE_RETRIES (in module *polyswarm-client.abstractarbiter*), 4
 MAX_TRIES (in module *polyswarm-client.abstractambassador*), 3
 MAX_WAIT (in module *polyswarmclient*), 17
 MAX_WAIT (in module *polyswarmclient.utils*), 14
 MAX_WORKERS (in module *polyswarmclient.utils*), 14
 Microengine (class in *microengine.clamav*), 21
 Microengine (class in *microengine.eicar*), 21
 Microengine (class in *microengine.multi*), 22
 Microengine (class in *microengine.producer*), 22
 Microengine (class in *microengine.scratch*), 22
 Microengine (class in *microengine.yara*), 23
 microengine (module), 20
 microengine.__main__ (module), 20
 microengine.clamav (module), 21
 microengine.eicar (module), 21
 microengine.multi (module), 22
 microengine.producer (module), 22
 microengine.scratch (module), 22
 microengine.yara (module), 23

N

NctApproveVerifier (class in *polyswarm-client.verifiers*), 15
 NctTransferVerifier (class in *polyswarm-client.verifiers*), 16
 NonceManager (class in *polyswarmclient*), 17
 NonceManager (class in *polyswarmclient.transaction*), 13
 NOT_EICAR (in module *ambassador.eicar*), 19

O

OffersClient (class in *polyswarmclient*), 17
 OffersClient (class in *polyswarmclient.offersclient*), 12

OnInitializedChannelCallback (class in *polyswarmclient.events*), 10
 OnNewAssertionCallback (class in *polyswarm-client.events*), 10
 OnNewBlockCallback (class in *polyswarm-client.events*), 9
 OnNewBountyCallback (class in *polyswarm-client.events*), 10
 OnNewVoteCallback (class in *polyswarm-client.events*), 10
 OnQuorumReachedCallback (class in *polyswarm-client.events*), 10
 OnRevealAssertionCallback (class in *polyswarmclient.events*), 10
 OnRevealAssertionDueCallback (class in *polyswarmclient.events*), 11
 OnRunCallback (class in *polyswarmclient.events*), 9
 OnSettleBountyDueCallback (class in *polyswarmclient.events*), 11
 OnSettledBountyCallback (class in *polyswarm-client.events*), 10
 OnVoteOnBountyDueCallback (class in *polyswarmclient.events*), 11

P

Parameters (class in *polyswarmclient.parameters*), 12
 peek() (polyswarmclient.events.Schedule method), 10
 polyswarm_client() (in module *balancemanager.__main__*), 1
 polyswarmclient (module), 3
 polyswarmclient.abstractambassador (module), 3
 polyswarmclient.abstractarbiter (module), 4
 polyswarmclient.abstractmicroengine (module), 5
 polyswarmclient.abstractscanner (module), 6
 polyswarmclient.balanceclient (module), 6
 polyswarmclient.bloom (module), 6
 polyswarmclient.bountiesclient (module), 7
 polyswarmclient.config (module), 8
 polyswarmclient.corpus (module), 9
 polyswarmclient.events (module), 9
 polyswarmclient.log_formatter (module), 11
 polyswarmclient.offersclient (module), 12
 polyswarmclient.parameters (module), 12
 polyswarmclient.producer (module), 12
 polyswarmclient.relayclient (module), 12
 polyswarmclient.stakingclient (module), 13
 polyswarmclient.transaction (module), 13
 polyswarmclient.utils (module), 14
 polyswarmclient.verifiers (module), 15

PostAssertionTransaction (class in polyswarm-client.bountiesclient), 7
 PostAssertionVerifier (class in polyswarm-client.verifiers), 16
 PostBountyTransaction (class in polyswarm-client.bountiesclient), 7
 PostBountyVerifier (class in polyswarm-client.verifiers), 16
 PostVoteTransaction (class in polyswarm-client.bountiesclient), 8
 PostVoteVerifier (class in polyswarm-client.verifiers), 16
 Producer (class in polyswarmclient.producer), 12
 put () (polyswarmclient.events.Schedule method), 10

Q

QUEUE (in module arbiter.producer), 25
 QUEUE (in module microengine.producer), 22
 QueuedBounty (class in polyswarm-client.abstractambassador), 3

R

RATE_LIMIT_SLEEP (in module polyswarmclient), 18
 REDIS_ADDR (in module arbiter.producer), 25
 REDIS_ADDR (in module microengine.producer), 22
 register () (polyswarmclient.events.Callback method), 9
 RELAY_LEEWAY (in module balancemanager), 3
 RelayClient (class in polyswarmclient), 17
 RelayClient (class in polyswarmclient.relayclient), 13
 RelayDepositTransaction (class in polyswarm-client.relayclient), 12
 RelayWithdrawTransaction (class in polyswarm-client.relayclient), 13
 remove () (polyswarmclient.events.Callback method), 9
 REQUEST_TIMEOUT (in module polyswarmclient), 17
 REQUEST_TIMEOUT (in module worker), 26
 RevealAssertion (class in polyswarmclient.events), 11
 RevealAssertionTransaction (class in polyswarmclient.bountiesclient), 8
 RevealAssertionVerifier (class in polyswarm-client.verifiers), 16
 RULES_DIR (in module microengine.yara), 23
 run () (balancemanager.BalanceManager method), 3
 run () (balancemanager.Maintainer method), 3
 run () (polyswarmclient.abstractambassador.AbstractAmbassador method), 4
 run () (polyswarmclient.abstractarbiter.AbstractArbiter method), 5
 run () (polyswarmclient.abstractmicroengine.AbstractMicroengine method), 5
 run () (polyswarmclient.Client method), 18

run () (worker.Worker method), 26
 run_oneshot () (balancemanager.BalanceManager method), 3

S

Scanner (class in microengine.clamav), 21
 Scanner (class in microengine.eicar), 21
 Scanner (class in microengine.multi), 22
 Scanner (class in microengine.scratch), 22
 Scanner (class in microengine.yara), 23
 ScanResult (class in polyswarm-client.abstractscanner), 6
 Schedule (class in polyswarmclient.events), 10
 schedule () (polyswarmclient.Client method), 18
 set_level () (polyswarmclient.config.LoggerConfig method), 9
 SettleBounty (class in polyswarmclient.events), 11
 SettleBountyTransaction (class in polyswarm-client.bountiesclient), 8
 SettleBountyVerifier (class in polyswarm-client.verifiers), 16
 sha3 () (in module polyswarmclient.utils), 14
 sha3_256 () (in module polyswarmclient.utils), 14
 sign_transactions () (polyswarmclient.Client method), 18
 StakeDepositTransaction (class in polyswarm-client.stakingclient), 13
 StakeWithdrawTransaction (class in polyswarm-client.stakingclient), 13
 StakingClient (class in polyswarmclient), 17
 StakingClient (class in polyswarm-client.stakingclient), 13
 StakingDepositVerifier (class in polyswarm-client.verifiers), 16
 StakingWithdrawVerifier (class in polyswarm-client.verifiers), 16

T

TASK_TIMEOUT (in module polyswarmclient.utils), 14
 TIME_TO_POST_ASSERTION (in module microengine.producer), 22
 TIME_TO_POST_VOTE (in module arbiter.producer), 25
 to_string () (in module polyswarmclient.utils), 14
 to_wei () (polyswarmclient.Client static method), 18

U

UNKNOWN_PARAMETER (in module polyswarm-client.verifiers), 15

V

validate_apikey () (in module polyswarm-client.config), 8

[validate_optional_transfer_amount\(\)](#) (in module `balancemanager.__main__`), 1
[validate_transfer_amount\(\)](#) (in module `balancemanager.__main__`), 1
[value](#) (`polyswarmclient.bloom.BloomFilter` attribute), 7
[verify\(\)](#) (`polyswarmclient.transaction.AbstractTransaction` method), 14
[verify\(\)](#) (`polyswarmclient.verifiers.AbstractTransactionVerifier` method), 15
[verify\(\)](#) (`polyswarmclient.verifiers.NctApproveVerifier` method), 16
[verify\(\)](#) (`polyswarmclient.verifiers.NctTransferVerifier` method), 16
[verify\(\)](#) (`polyswarmclient.verifiers.PostAssertionVerifier` method), 16
[verify\(\)](#) (`polyswarmclient.verifiers.PostBountyVerifier` method), 16
[verify\(\)](#) (`polyswarmclient.verifiers.PostVoteVerifier` method), 16
[verify\(\)](#) (`polyswarmclient.verifiers.RevealAssertionVerifier` method), 16
[verify\(\)](#) (`polyswarmclient.verifiers.SettleBountyVerifier` method), 16
[verify\(\)](#) (`polyswarmclient.verifiers.StakingDepositVerifier` method), 16
[verify\(\)](#) (`polyswarmclient.verifiers.StakingWithdrawVerifier` method), 16
[VoteOnBounty](#) (class in `polyswarmclient.events`), 11

W

[w3](#) (in module `polyswarmclient`), 17
[w3](#) (in module `polyswarmclient.bloom`), 6
[Withdraw](#) (class in `balancemanager`), 3
[withdraw\(\)](#) (in module `balancemanager.__main__`), 2
[Worker](#) (class in `worker`), 26
[worker](#) (module), 25
[worker.__main__](#) (module), 25