ErisX

# ErisX REST API v3.2

Please contact ErisX sales representatives or Client Services for more information on this documentation.

## Contents

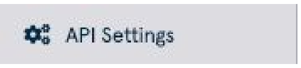
# 1 Change History

| Date | Message(s) or Section | Description |
|---|---|---|
| 20190731 | | Version 1 |
| 20190809 | Filters | In python, filters should be specified in the **json** argument of requests.post function, not in the **data** argument.<br>Some new filters have been added to the different endpoints.<br>Each token will now be valid for 60 seconds, instead of the previous 30 seconds |
| 20190819 | Trades Response | The trades response will now include 3 new fields (tcr_id, client_order_id, fix_id) |
| 20190925 | Trades Response | The trades response will now include one new field: product_code |
| 20200130 | | Version 1.5 |
| | Messages updated; Account, Balances and Trades. | New fields introduced to expose Futures information to clients (highlighted in green). |
| | New message added; Positions | New positions endpoint to query the positions for a given account. |
| 20200301 | | Version 2.0 |
| | Movements & Trades Filter changes | Removed asset_type filter from Movements and fee_type filter from Trades. |
| | New endpoints: deposit_address, linked_accounts, build_withdrawal_request and submit_withdrawal_request | A set of new endpoints to allow users to process deposits and withdrawals. |
| | Authentication | Removed Python 2 authentication example |
| 20200326 | | Version 3.0 |
| | New Service: Block Trades API Service | Added new endpoints to interact with the Block Trade API |
| | New section: Balances Calculations | New section that details balances calculations |
| | Funding Password Signing | Modified the javascript example for easier use |
| 20200505 | | Version 3.1 |
| | New Service: Order Management API Service | Added new endpoints to interact with the Order Management system via REST API |
| | Movements Endpoint | Defined possible values for Type field in the movements endpoint response |
| 20200709 | | Version 3.2 |
| | New endpoint: Closeouts | Add new endpoint to get Closeouts information |

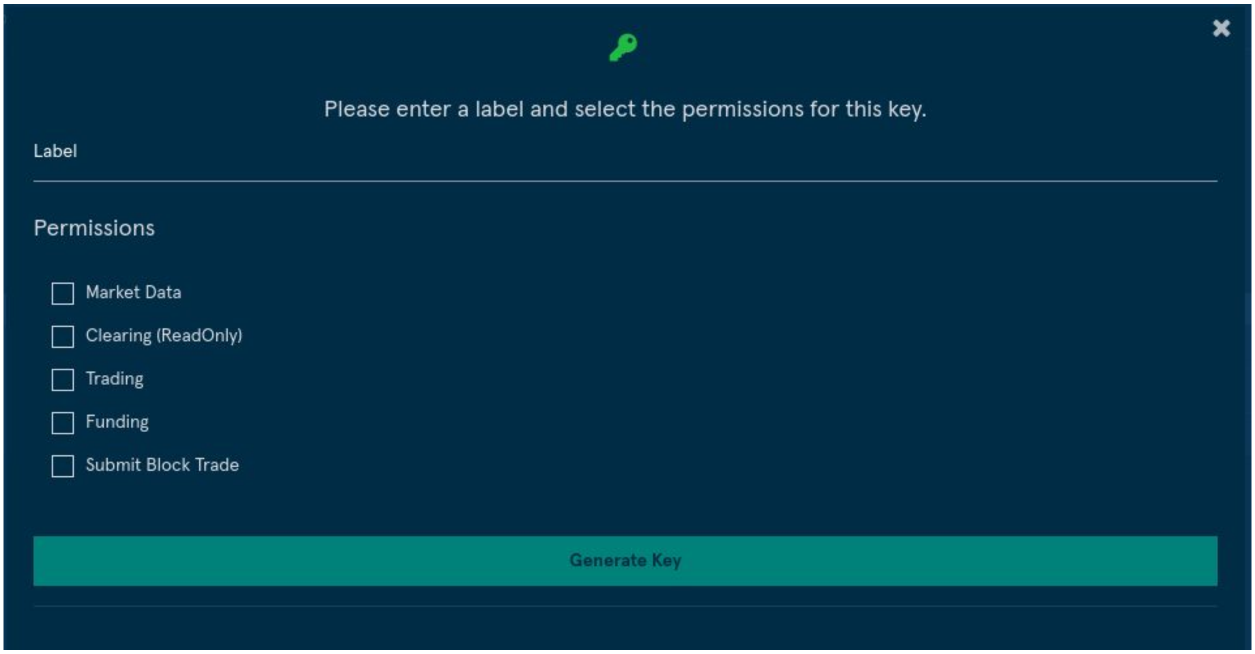
# 2 General Concepts

## 2.1 API Credentials

In order to sign your API requests, you will need to create a set of API Credentials.

From the Eris Member Portal, navigate to the dropdown next to your username in the top right of the page and select  API Settings.

After clicking **Create New API Key** you will be asked to select the permissions you want to enable.



Please enter a label and select the permissions for this key.

Label

Permissions

- □ Market Data
- □ Clearing (ReadOnly)
- □ Trading
- □ Funding
- □ Submit Block Trade

### API Key permissions

- **Market Data:** An API key can query historical data or subscribe to real time data.
- **Trading:** Allows an API key to enter, modify and cancel orders.
- **Clearing (ReadOnly):** Allows an API key to query information about their clearing accounts.
- **Funding:** Allows an API key to initiate withdrawal requests.
- **Submit Block Trade:** Allows an API key to submit Block Trades.

When ready click **Generate Key** and you will be presented with two pieces of information that must be kept safe as they will be needed for authentication of calls to the end points and will not be shown again:

- **API key**
- **Secret**

## 2.2 Authentication

A json web token should be generated using the HS256 algorithm on the API key, secret and timestamp as described in the examples below. This token should be included in the header of every request.

- **Timestamp:** The authentication token requires a Unix Epoch timestamp.
- **Token Age:** Each token will only be valid for 60 seconds after the timestamp.

Notes:

- In python use the **pyjwt** package to generate the token (https://pyjwt.readthedocs.io/en/latest/).
- Note that some jwt encoding functions may return a byte array rather than a string, and some languages require explicit conversion. For example, in Python, you must use the **decode()** function.
- Be aware that there must be a blank space between **Bearer** and the token.

Javascript Example:

```
const jwt = require('jsonwebtoken');
const axios = require('axios').default;
const apiKey = '9106676d85f1163f.d1ba2efac8bc1e0a';
const secret = '31b6b61606588580';
var payload = {
 iat: Date.now(),
 sub: apiKey
};
var token = jwt.sign(payload, secret, { algorithm: 'HS256'});
axios({
  method: 'POST',
  url: `https://clearing.erisx.com/api/v1/${endpoint_name}`,
  timeout: 30000, // 30 seconds
  headers: {
    Authorization: `Bearer ${token}`,
  },
  data: {}
});
```

Python 3 Example:

```
import jwt
import time
import requests

def gen_token(secret, api_key):
   unix_timestamp = int(round(time.time()))
   payload_dict = {'sub': api_key, 'iat': unix_timestamp}
```

```
    return jwt.encode(payload_dict, secret, algorithm='HS256').decode('utf-8')

my_secret = '31b6b61606588580'
my_api_key = '9106676d85f1163fgd1ba2efac8bc1e0a'
url = 'https://clearing.erisx.com/api/v1/'
token = gen_token(my_secret, my_api_key)
requests.post(url= url + endpoint_name, headers={'Authorization': 'Bearer ' +
token}, json={}) # Be aware that there is a blank space after Bearer
```

## 2.3 Funding Password Signing

In order to enhance security in funding related operations, some endpoints of the Clearing API require a two step process authentication. The first authentication is based on the API credentials and the token derivation described in the section above. The second authentication is based on the generation of an encrypted signature based on the member user's Funding Password. In order to correctly perform the signature, the clearing member should follow the following steps:

- Generate a key from the funding password using the Password-Based Key Derivation Function 2 (PBKDF2) with the following parameters:
  - Hashing algorithm: SHA-256
  - Password: Clearing member user's Funding Password.
  - Salt: Auth ID for the clearing member user, which can be found in the response from the endpoint Build Withdrawal Request in the field 'auth_id'.
  - Iterations: 100,000
  - Derive Key Length: 32 bytes (256 bits).
- Generate a canonical signature for the appropriate message using the Elliptic Curve Digital Signature Algorithm (ECDSA) with the elliptic curve SECP256k1, where the key used in the signature should be the key generated in the previous step. Encoding of the signature should be in DER format.
- Encode the signature using Base58 encoding. Note that some base58 encoding functions may return a byte array rather than a string, and some languages require explicit conversion. For example, in Python, you must use the decode() function.

This signature will enable the ErisX ClearingHouse to validate  the funding password without the clearing member having to expose the funding password at any moment over the internet, which provides a higher layer of security for the safe keeping of the clearing member's credentials.

Javascript Example:

```
const crypto = require('crypto');
const bs58 = require('bs58');
const ecdsa = require('ecdsa');
const pbkdf2 = require('pbkdf2-sha256');
const BigInteger = require('bigi');

function sha256(str) {
  return crypto.createHash('sha256').update(str).digest();
}

function privateKeyFromPassword(authId, password) {
return pbkdf2(password, authId, 100000, 32);
}

function signMessage(message, privateKey) {
  let shaMsg = sha256(message);
  let signature = ecdsa.sign(shaMsg, BigInteger.fromBuffer(privateKey));
  return signature.toDER();
}

const privateKey = privateKeyFromPassword(auth_id, password);
const signature = bs58.encode(signMessage(request_data, privateKey));
```

Python3 Example:

```
import hashlib
import ecdsa
from ecdsa.util import sigencode_der_canonize
import base58

def privateKeyFromPassword(authId, password):
      return hashlib.pbkdf2_hmac(
          hash_name='sha256',
          password=password.encode(),
          salt=authId.encode(),
          iterations=100000,
          dklen=32)

def signMessage(message, authId, password):
      privateKey = privateKeyFromPassword(authId, password)
      sk = ecdsa.SigningKey.from_string(privateKey, curve=ecdsa.SECP256k1)
      signature = sk.sign_deterministic(
          message.encode(),
          sigencode=sigencode_der_canonize,
          hashfunc=hashlib.sha256)
      return base58.b58encode(signature).decode('ascii')

signature = signMessage(message, authId, password)
```

## 2.4 Filters

Some API calls allow the use of filters. These filters provide a greater level of flexibility to queries. Ultimately, providing more efficient requests and a better API experience.

The filter query has the following json type format. Multiple filters can be applied in a single request to best tailor the query. In python, filters should be given under the **json** argument of the requests.post function.

```
"filter": [{"attr": "attribute_name","op": "eq","value": "attribute_value" }]
```

| Field | Value |
|---|---|
| filter | Name of the query parameter |
| attr | Name of the attribute that wants to be used in the query |
| op | Operations present in the query:<br>'eq' - equal<br>'ne' - not equal<br>'gt' - greater than<br>'gte' - greater than or equal<br>'lt' - less than<br>'lte' - less than or equal |
| value | Value or array of values of the attribute to which the query will compare. |

## 2.5 Sorting

Queries also provide the ability to sort the results using the following format.

```
"sort": [{ "attr": "attribute_name", "value": "desc" }]
```

| Field | Value |
|---|---|
| sort | Name of the query parameter |
| attr | Name of the attribute that wants to be used in the query |
| value | Direction of the sort: 'desc' - descending or 'asc' - ascending |

## 2.6 Pagination

Some requests can be paginated. The offset and limit parameters on the request allows the user to choose how many results should be included in the return message and where the results should begin.

Maximum number of results per request is 100.

These two parameters are optional and available parameters in all endpoints except in the Balances endpoint.

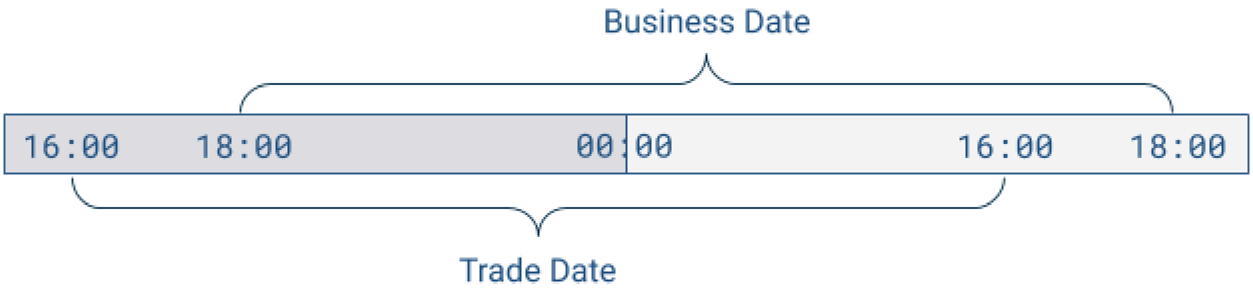
```
"offset":0, "limit":10
```

| Field | Value |
|---|---|
| offset | Integer. The number of entries to skip (default: 0). |
| limit | Integer. Maximum number of results to be returned (default: 100). |

## 2.7 Trade Date and Business Date

A new trade date starts at 4:00:00pm CST and finishes at 3:59:59pm CST the following day. All Exchange (trading) activity will be included in the appropriate trade date depending on the time of the activity. (I.e. trading activity at 2019-01-01 15:59:59 CST will be included in 2019-01-01 trade date but trading activity at 2019-01-01 16:00:00 CST will be included in 2019-01-02 trade date).

A new business date starts at 6:00pm CST and finishes at 5:59pm CST of the following day. All asset movement activity (Deposits, Withdrawals) will be included in the appropriate business date depending on the time of the asset movement. (I.e. a deposit made at 2019-01-01 17:59:59 CST will be included in 2019-01-01 business date but a deposit made at 2019-01-01 18:00:00 CST will be included in the 2019-01-02 business date).



The relevance of these two time frames is important for understanding the calculation of the Opening Balance in the Balances endpoint.

## 2.8 Rate Limiting

Requests are throttled per IP address. Limit: 5 requests in a 10 second period.

When the rate limit is exceeded, a response with status **429 -> Too Many Requests** is returned.

If the limit is exceeded the IP address will be restricted from making new requests for 60 seconds.

We highly recommend adding logic to your application to gracefully process the 429 To Many Requests message. We suggest that if the limit is breached, your application will pause for the required time in order to be within the rate limiting again. An application that repeatedly breaches the limit will  keep extending the restricted period, thus, preventing your application to function correctly again.

# 3 Clearing API Service

This API service enables clients to interact with their Clearing accounts in order to extract data regarding their activity.  All requests and responses are application/json content type.

All Clearing API endpoints are private and every request needs to be signed using the authentication method described in the Authentication section. Some endpoints require additional signing authentication as described in the Funding Password Signing section.

## 3.1 REST API Endpoint URL

- **Production: https://clearing.erisx.com/api/v1**
- **New Release (test): https://clearing.newrelease.erisx.com/api/v1**

## 3.2 Funds Designation

All customer funds for trading on designated contract markets (futures exchanges like ErisX) must be kept apart ("segregated") from non customer funds.

ErisX currently supports three funds designations:

- N: Represents "non-segregated" funds held on behalf of members trading ErisX Spot products.
- P: Represents "member property" funds held on behalf of direct members trading ErisX futures products.
- S: Represents "segregated" funds held on behalf of the clients of Futures Commission Merchants (FCM's) trading ErisX futures products.

## 3.3 Balances Calculations

As defined in the Trade Date and Business Date section, Trade Date has a different start time and end time than Business Date. This difference has certain implications in how balances are calculated and provided in response to the different Clearing API endpoints.

### 3.3.1 Account Endpoint balance values

The response of the accounts endpoint, provides a summary of balances for the account along with other account information. The balances reflected include all Exchange and Clearing House activities up to the moment when the API request is made. Therefore, this value will include the latest information known for the account.

### 3.3.2 Balances Endpoint, Opening Balance calculation

The balances endpoint, provides a more detailed view of the balances for a particular account. The opening balance is generated using a reference to the current business date. Therefore, the following rules need to be considered to understand the value provided;

- Include all asset movements (Deposits, Withdrawals, etc.) prior to the beginning of the current business date.
- Include all trading activity for the trade date prior to the current business date.

Examples

**Request is within the same trade date and business date:**

The Opening balance for a request on Tuesday **@ 15:50** CT will include:

- Trades prior to Monday @ 16:00 CT
- Asset movements prior to Monday @ 18:00 CT

**Request is during a new trade date but still the same business date**
The Opening balance for a request on Tuesday **@ 16:30** CT will include:

- Trades up to Monday @ 16:00 CT
- Asset movements prior to Monday @ 18:00 CT

**Request is during a new trade date and new business date (same calendar date)**
The Opening balance for a request on a Tuesday **@ 18:30** CT will include:

- Trades up to Tuesday @ 16:00 CT
- Asset movements prior to Tuesday @ 18:00 CT

### 3.4 Accounts Endpoint

This endpoint will return a list of all accounts a member has available to them, as well as basic balance information.  More detailed balance information is returned in the getBalances endpoint.

- **HTTP Request Type:** POST
- **Endpoint:** /accounts
- **API security:** This API endpoint requires an authentication token with Clearing API read permission.

Inputs:

| Field | Value | |
|---|---|---|
| filter (optional) | Default: `"filter": [{ "attr": "account_id", "op": "eq", "value": member_account_id }]` | |
| | account_id | Account ID |
| offset (optional) | Number of elements to be offset in the request for pagination purposes | |
| limit (optional) | Limit of elements returned in the request | |

Example Requests:

```
requests.post(
    url="https://clearing.erisx.com/api/v1/accounts",
    headers={"Authorization": "Bearer " + token},
    json={})

requests.post(
    url="https://clearing.erisx.com/api/v1/accounts",
    headers={"Authorization": "Bearer " + token},
    json={
        "filter": [{
            "attr":
                "account_id",
            "op":
                "eq",
            "value": [
                "27ff6d34-523d-476d-9ad5-edeb373b83dc"
            ]
        }],
        "offset": 0,
        "limit": 10
    })
```

### Outputs

| Field | Value |
|---|---|
| count | Number of member accounts found |
| timestamp | Time of the request |
| accounts | List of all available accounts |
| account_id | Account ID |
| account_number | Account Number |
| fix_ids | List of all available FIX Trading IDs |
| member_users | Member users associated with the account |
| balances | Balances of the account at the time of the request |
| cti | Customer Type Indicator (For futures accounts) |
| origin | Origin (For futures accounts) |

### Example Response:

```
{
  "count": 1,
  "timestamp": "2018-01-01T06:00:00.000Z",
  "accounts": [
    {
      "account_id": "27ff6d34-523d-476d-9ad5-edeb373b83dc",
      "account_number": "DM-000001",
      "member_users": [
        "5c532a02f2530e906a9c065f"
      ],
      "balances": [
        {
          "asset_type": "USD",
          "amount": "100.5"
        },
        {
          "asset_type": "TBTC",
          "amount": "1.5"
        }
      ],
      "fix_ids": [
        "trading_id"
      ],
      "cti": 1,
      "origin": 2
    }
  ]
}
```

## 3.5 Balances Endpoint

This endpoint will return a detailed set of balance information for a given account.

- **HTTP Request Type:** POST
- **Endpoint:** /balances
- **API security:** This API endpoint requires an authentication token with Clearing API read permission.

Inputs

| Field | Value |
|---|---|
| account_id | Account ID |

Example Request:

```
requests.post(
    url="https://clearing.erisx.com/api/v1/balances",
    headers={"Authorization": "Bearer " + token},
    json={"account_id": "27ff6d34-523d-476d-9ad5-edeb373b83dc"})
```

Outputs

| Field | Value |
|---|---|
| account_id | Account ID |
| timestamp | Time of the request |
| report_date | Business date associated with the request |
| asset_type | Asset Type |
| opening_balance | Balance at the beginning of the corresponding business date |
| asset_movement | Amount of asset movements for the business date up to the time of the request. |
| spot_movement | Amount of Spot trade movements for the business date up to the time of the request. |

| | |
|---|---|
| closing_balance | Balance as of the time of the request. |
| change_in_balance | Change in balance between the beginning of the request’s business date and the time of the request. |
| exchange_fees | Exchange fees paid during the request’s trade date |
| clearing_fees | Clearing fees paid during the request’s trade date |
| other_fees | Other fees paid during the request’s business date |
| realized_p_and_l | Realized Profit and Loss in Futures trades |
| futures_delivery | Quantity of Futures contract delivered |
| total_equity | Total Equity |
| reserved_margin | Reserved Margin for Futures positions |
| total_excess_deficit | Total Excess Deficit |
| net_liquidating_value | Net Liquidating Value |
| available_to_trade | Balance available to trade (does not include working orders) |
| reserved_ote | Reserved OTE |
| fd | Funds designation |
| closing_price | Closing price of each asset at the end of the previous trade date |
| closing_price_date | Trade date to which the closing price belongs |
| usd_value | The USD equivalent balance for each asset based on the closing price of the previous trade date from the time of the request. |

Example Response:

```
{
  "account_id": "3cfb773e-3a71-42c8-bab5-037ca4ae616f",
  "timestamp": "2020-01-30T16:52:41.900Z",
  "report_date": "2020-01-30",
  "balances": [
    {
      "opening_balance": "100621.4286",
      "spot_movement": "0.0",
      "exchange_fees": "-1.8",
      "clearing_fees": "-0.2",
      "other_fees": "0.0",
      "asset_movement": "0.0",
      "realized_p_and_l": "81.8",
      "futures_delivery": "0.0",
      "closing_balance": "100701.2286",
      "total_equity": "100701.2286",
      "reserved_margin": "-21294.6",
      "total_excess_deficit": "79406.6286",
      "net_liquidating_value": "100701.2286",
      "available_to_trade": "79406.6286",
      "reserved_ote": "1375.4",
      "fd": "P",
      "asset_type": "USD",
      "closing_price": "1.0",
      "closing_price_date": "2020-01-28",
      "usd_value": "100701.2286",
      "change_in_balance": "79.8"
    },
    {
      "opening_balance": "0.79925",
      "spot_movement": "0.0",
      "exchange_fees": "0.0",
      "clearing_fees": "0.0",
      "other_fees": "0.0",
      "asset_movement": "0.0",
      "realized_p_and_l": "0.0",
      "futures_delivery": "0.0",
      "closing_balance": "0.79925",
      "total_equity": "0.79925",
      "reserved_margin": "0.0",
      "total_excess_deficit": "0.79925",
      "net_liquidating_value": "0.79925",
      "available_to_trade": "0.79925",
      "reserved_ote": "0.0",
      "fd": "P",
      "asset_type": "TBTC",
      "closing_price": "9068.0",
      "closing_price_date": "2020-01-28",
      "usd_value": "7247.599",
      "change_in_balance": "0.0"
    }
  ]
}
```

## 3.6 Movements Endpoint

This endpoint will return a detailed set of asset movements information for a given account.

- **HTTP Request Type:** POST
- **Endpoint:** /movements
- **API security:** This API endpoint requires an authentication token with Clearing API read permission.

### Inputs

| Field | Value | |
|---|---|---|
| filters (optional) | Default: `"filter": [{ "attr": "account_id", "op": "eq", "value": member_account_id }]` | |
| | account_id | Account ID |

| | | |
|---|---|---|
| | time | Start time using “op”:gte or gt and End time using “op”:lte or lt. If no time query is made it will return all the available data (subject to the specified limit) |
| offset (optional) | Number of elements to be offset in the request for pagination purposes | |
| limit (optional) | Limit of elements returned in the request | |
| Sort (optional) | Default: `"sort": [{ "attr": "time", "value": "desc"}]` | |

Example Request:

```
requests.post(
    url="https://clearing.erisx.com/api/v1/movements",
    headers={"Authorization": "Bearer " + token},
    json={
        "filter": [{
            "attr": "account_id",
            "op": "eq",
            "value": "27ff6d34-523d-476d-9ad5-edeb373b83dc"
        }, {
            "attr": "time",
            "op": "lte",
            "value": "2018-01-01T05:59:30.000Z"
        }, {
            "attr": "time",
            "op": "gte",
            "value": "2017-12-01T05:59:30.000Z"
        }],
        "sort": [{
            "attr": "time",
            "value": "asc"
        }],
        "offset": 0,
        "limit": 10
    })
```

### Outputs

| **Field** | **Value** |
|---|---|
| count | Number of results returned |
| description | Description of the asset movement |
| time | Timestamp of the asset movement |
| date | Business date of the asset movement |
| type | Type of the asset movement: **deposit**, **withdrawal**, **reversal**, **general**, **fee_rebate**, **delivery** |
| posting_summary | Details of the asset movement (account ID, Asset type, Key (specifies what the amount refers to), Amount and Report Date). List of available keys: |

| | |
|---|---|
| | ● "amount": General movement amount.<br>● "bank_fee": Bank Fees Charged<br>● "clearing_fee": Clearing House Fees Charged<br>● "exchange_fee": Trading Fees Charged<br>● "other_fees": Other Fees Charged |

Example Response:

```
{
 "result": {
   "count": 1,
   "movements": [
     {
       "description": "DEPOSIT 0.13057719 TBTC",
       "time": "2018-01-01T06:00:00.000Z",
       "type": "deposit",
       "posting_summary": [
               {
                 "account_id": "27ff6d34-523d-476d-9ad5-edeb373b83dc",
                 "asset_type" : "TBTC",
                 "key": "notional",
                 "amount": "0.25486",
                 "report_date": "2018-01-01"
               },
               {
                 "account_id": "27ff6d34-523d-476d-9ad5-edeb373b83dc",
                 "asset_type": "TBTC",
                 "key": "clearing_fee",
                 "amount": "0.00002549",
                 "report_date": "2018-01-01"
               },
               {
                 "account_id": "27ff6d34-523d-476d-9ad5-edeb373b83dc",
                 "asset_type": "TBTC",
                 "key": "exchange_fee",
                 "amount": "0.00022937",
                 "report_date": "2018-01-01"
               }
             ],
           }
         ]
}
```

## 3.7 Trades Endpoint

This endpoint will return a set of trade information for a given account.

- **HTTP Request Type:** POST
- **Endpoint:** /trades
- **API security:** This API endpoint requires an authentication token with Clearing API read permission.

### Inputs

| Field | Value | |
|---|---|---|
| filters (optional) | Default: `"filter": [{ "attr": "account_id", "op": "eq", "value": member_account_id }]` | |
| | account_id | Account ID |
| | time | Start time using "op":gte or gt and End time using "op":lte or lt. If no time query is made it will return all the available data (subject to the specified limit) |
| | trade_id | Trade ID |
| | side | Side of the trade (BUY, SELL) |
| | aggressor | Aggressor in the trade (Y, N) |
| | qty | Quantity of the trade |
| | px | Price of the trade |
| | qty_type | Base currency |
| | px_type | Quoted currency |
| | type | Types: futures, spot, delivery or reversal |
| offset (optional) | Number of elements to be offset in the request for pagination purposes | |
| limit (optional) | Limit of elements returned in the request | |
| Sort (optional) | Default: `"sort": [{ "attr": "time", "value": "desc"}]` | |

`Example Request:`

```
requests.post(
    url="https://clearing.erisx.com/api/v1/trades",
    headers={"Authorization": "Bearer " + token},
    json={
        "filter": [{
            "attr": "account_id",
            "op": "eq",
            "value": "27ff6d34-523d-476d-9ad5-edeb373b83dc"
        }
    }
)
```

### Outputs

| Field | Value |
|---|---|
| count | Number of results returned |

| | |
|---|---|
| trade_id | Trade ID of the trade |
| tcr_id | Trade Capture Report ID |
| client_order_id | Client Order ID |
| fix_id | FIX ID |
| time | Timestamp of the trade |
| description | Description of the trade |
| side | Side of the trade (BUY, SELL) |
| account_id | Account ID |
| aggressor | Aggressor of the trade (Y, N) |
| qty | Quantity |
| px | Price |
| clearing_fee | Clearing fee of the trade |
| exchange_fee | Exchange fee of the trade |
| product_code | Product code |
| qty_type | Base currency |
| px_type | Quote currency |
| fee_type | Fee currency |
| report_date | Business date of the trade |
| contract_symbol | Contract Symbol |
| asset_type | Asset Type |
| trader_type | Trade Type |
| record_type | Record Type |
| notional | Notional Amount |
| total_amount | Total Amount charged to the Account |
| trade_report_id | Trade Report ID |
| customer_account_ref | Customer Account Reference |
| product_suffix | Product Type: SP, FUT |
| state | State of the Trade |
| expiration_time | Expiration date and Time of the futures contract involved in the trade |
| cti | CTI |
| origin | Origin |

### Example Response:

```
{
  "count": 1,
  "trades": [
    {
      "account_id": "27ff6d34-523d-476d-9ad5-edeb373b83dc",
      "contract_symbol": "TBTCZ9",
      "asset_type": "TBTC",
      "px_type": "USD",
      "side": "BUY",
      "trade_type": "REGULAR",
      "record_type": "T",
      "qty": "1.0",
      "px": "6994.0",
      "notional": "699.4",
      "aggressor": "Y",
      "fee_type": "USD",
      "exchange_fees": "0.001",
      "clearing_fees": "0.001",
      "total_amount": "699.402",
      "tcr_id": "477188150",
      "trade_report_id": "1125899907429878",
      "trade_id": "B2019196081HP00",
      "customer_account_ref": "buy_side",
      "fix_id": "1",
      "product_suffix": "FUT",
      "state": "posted",
      "time": "2018-01-01T06:00:00.000000Z",
      "expiration_time": "2030-01-01T06:00:00Z",
      "cti": 1,
      "origin": 1,
      "product_code": "TBTC/USD",
      "client_order_id": "1",
      "description": "BUY 1.0 TBTCZ9 @ 6994.0 USD"
    }
  ]
}
```

## 3.8 Requests Endpoint

This endpoint will return the asset movements requests made by the appropriate account and their current status.

- **HTTP Request Type:** POST
- **Endpoint:** /requests
- **API security:** This API endpoint requires an authentication token with Clearing API read permission.

### Inputs

| Field | Value | |
|---|---|---|
| filters (optional) | Default: `"filter": [{ "attr": "account_id", "op": "eq", "value": member_account_id }]` | |
| | account_id | Account ID |
| | time | Start time using “op”:gte or gt and End time using “op”:lte or lt. If no time query is made it will return all the available data (subject to the specified limit) |
| | asset_type | Asset type (BTC, BCH, ETH, LTC) |
| | amount | Amount of the request |
| | transaction_type | Request type (withdrawal, deposit) |
| offset (optional) | Number of elements to be offset in the request for pagination purposes | |
| limit (optional) | Limit of elements returned in the request | |
| Sort (optional) | Default: `"sort": [{ "attr": "time", "value": "desc"}]` | |

Example Request:

```
requests.post(
    url="https://clearing.erisx.com/api/v1/requests",
    headers={"Authorization": "Bearer " + token},
    json={
        "filter": [{
            "attr": "account_id",
            "op": "eq",
            "value": "27ff6d34-523d-476d-9ad5-edeb373b83dc"
        }],
        "sort": [{
            "attr": "time",
            "value": "asc"
        }],
        "offset": 0,
        "limit": 10
    })
```

### Outputs

| Field | Value |
|---|---|
| count | Number of results returned |
| account_id | Account ID |
| dest_address | Destination Address of the request (Digital Assets Only) |
| time | Timestamp of the asset movement request |
| asset_type | Asset type |
| amount | Amount of the request |
| fee | Fees |
| fee_type | Fee currency |
| transaction_type | Transaction type (deposit, withdrawal) |
| state | State of the request (pending, rejected, posting) |

Example Response:

```
{
  "result": {
    "count": 1,
    "requests": [
      {
        "account_id": "27ff6d34-523d-476d-9ad5-edeb373b83dc",
        "dest_address": "2MvQsnz92K5DmKe9fd2GHPz4oRKJXoAR1m4",
        "time": "2018-01-01T06:00:10.000Z",
        "asset_type": "TBTC",
        "amount": "-0.0001",
        "fee": "-0.0000001",
        "fee_type": "TBTC",
        "transaction_type": "withdrawal",
        "state": "pending"
      }
    ]
  }
```

## 3.9 Positions Endpoint

This endpoint will return the list of open positions for each account.

- **HTTP Request Type:** POST
- **Endpoint:** /positions
- **API security:** This API endpoint requires an authentication token with Clearing API read permission.

### Inputs

| Field | Value |
|---|---|

| | | |
|---|---|---|
| filters (optional) | Default: `"filter": [{ "attr": "account_id", "op": "eq", "value": member_account_id }]` | |
| | account_id | Account ID |
| | contract_symbols | Contract symbol |
| offset (optional) | Number of elements to be offset in the request for pagination purposes | |
| limit (optional) | Limit of elements returned in the request | |
| Sort (optional) | Default: `"sort": [{ "attr": "time", "value": "desc"}]` | |

Example Request:

```
requests.post(
    url="https://clearing.erisx.com/api/v1/positions",
    headers={"Authorization": "Bearer " + token},
    json={
        "filter": [
            {
                "attr": "account_id",
                "op": "eq",
                "value": "27ff6d34-523d-476d-9ad5-edeb373b83dc"
            },
            {
                "attr": "contract_symbols",
                "op": "eq",
                "value": "BTCW44"
            },
        ],
        "sort": [{
            "attr": "time",
            "value": "asc"
        }],
        "offset": 0,
        "limit": 10
    }
)
```

### Outputs

| Field | Value |
|---|---|
| account_id | Account ID |
| positions | List of Open positions |
| contract_symbol | Exchange Contract Symbol |
| contract_code | Clearing House Contract Symbol |
| product_code | Product Code |
| closing_px_date | Date utilize for the closing price |
| total_long | Total Long positions open for a certain contract |

| | |
|---|---|
| total_short | Total short positions open for a certain contract |
| total_reserve_ote | Total reserved OTE for a certain contract |
| expiration_time | Expiration time of the contract |
| position_id | Position ID |
| qty | Quantity |
| px | Price |
| notional | Notional |
| reserve_margin_s | Reserve Margin for short position |
| reserve_margin_l | Reserve Margin for long position |
| et | Expiration Time of position |
| customer_account_ref | Customer Account Reference |
| cl_ord_id | Client Order ID |

Example Response:

```
[
  {
    "account_id": "27ff6d34-523d-476d-9ad5-edeb373b83dc",
    "positions": [
      {
        "contract_symbol": "BTCUSD1W",
        "contract_code": "BTCW44",
        "product_code": "BTC",
        "closing_px_date": "2018-01-01",
        "total_long": "400.0",
        "total_short": "0.1",
        "total_reserve_ote": "0.0",
        "expiration_time": "2019-01-01T06:00:00.000Z",
        "positions": [
          {
            "position_id": "t2",
            "qty": "-2",
            "px": "6000.0",
            "notional": "6000",
            "reserve_margin_s": "0.2",
            "reserve_ote": "-2000.0",
            "et": "2018-01-01T06:00:00.000Z",
            "customer_account_ref": "customer_account_ref",
            "cl_ord_id": "cl_ord_id"
          },
          {
            "position_id": "t1",
            "qty": "1",
            "px": "4000.0",
            "notional": "-4000",
            "reserve_margin_l": "400.0",
            "reserve_ote": "1000.0",
            "et": "2018-01-01T06:00:00.000Z",
            "customer_account_ref": "customer_account_ref",
            "cl_ord_id": "cl_ord_id"
          }
        ]
      }
    ]
  }
]
```

### 3.10 Closeouts Endpoint

This endpoint will return the list of closeouts for each account.

- **HTTP Request Type:** POST
- **Endpoint:** /closeouts
- **API security:** This API endpoint requires an authentication token with Clearing API read permission.

Inputs

<table>
<tr><th>Field</th><th colspan="2">Value</th></tr>
<tr><td rowspan="6">filters (optional)</td><td colspan="2">Default: <code>"filter": [{ "attr": "account_id", "op": "eq", "value": member_account_id }]</code></td></tr>
<tr><td>account_id</td><td>Account ID</td></tr>
<tr><td>closeout_id</td><td>Closeout ID</td></tr>
<tr><td>product_symbol</td><td>Product Symbol</td></tr>
<tr><td>contract_symbol</td><td>Contract symbol</td></tr>
<tr><td>time</td><td>Closeout Time. Format( yyyy-mm-ddThh:mm:ssZ)</td></tr>
<tr><td>closeout_id</td><td colspan="2">Closeout unique identifier</td></tr>
<tr><td>report_date</td><td colspan="2">Trade Date for closeout</td></tr>
<tr><td>time</td><td colspan="2">Timestamp of closeout</td></tr>
<tr><td>description</td><td colspan="2">Summary of closeout</td></tr>
<tr><td>account_id</td><td colspan="2">Account ID</td></tr>
<tr><td>account_label</td><td colspan="2">Account Label</td></tr>
<tr><td>product_symbol</td><td colspan="2">Product</td></tr>
<tr><td>contract_symbol</td><td colspan="2">Contract</td></tr>
<tr><td>contract_code</td><td colspan="2">Clearing contract code</td></tr>
<tr><td>qty</td><td colspan="2">Quantity of closeout</td></tr>
<tr><td>expiration_time</td><td colspan="2">Contract expiration time</td></tr>
</table>

| | |
|---|---|
| realized | Realized PnL in closeout |
| closeout_items | List of positions involved in the closeout |
| position_id | Position ID |
| qty | Quantity |
| px | Price |
| notional | Notional |
| trade_date | Trade date when position was opened |
| qty_type | Quantity type |
| px_type | Price type |
| trd_type | Trade Type |
| trade_report_id | Trade Report ID |
| trade_id | Trade ID |
| customer_account_ref | Customer Account Reference |
| cl_ord_id | Client Order ID |
| offset (optional) | Number of elements to be offset in the request for pagination purposes |
| limit (optional) | Limit of elements returned in the request |
| Sort (optional) | Default: `"sort": [{ "attr": "time", "value": "desc"}]` |

Example Request:

```
requests.post(
    url="https://clearing.erisx.com/api/v1/closeouts",
    headers={"Authorization": "Bearer " + token},
    json={
        "filter": [
            {
                "attr": "account_id",
                "op": "eq",
                "value": "27ff6d34-523d-476d-9ad5-edeb373b83dc"
            },
            {
                "attr": "closeout_id",
                "op": "eq",
                "value": "asdvsevsvd"
            },
            {
                "attr": "product_symbol",
                "op": "eq",
                "value": "BTC"
            },
            {
                "attr": "contract_symbol",
                "op": "eq",
                "value": "BTCZ0"
            },
            {
                "attr": "time",
                "op": "ne",
                "value": "2018-01-01T06:00:00.000Z"
            },
        ],
        "sort": [{
            "attr": "time",
            "value": "asc"
        }],
        "offset": 0,
        "limit": 10
    }
)
```

### Outputs

| Field | Value |
|---|---|
| account_id | Account ID |
| positions | List of Open positions |
| contract_symbol | Exchange Contract Symbol |
| contract_code | Clearing House Contract Symbol |
| product_code | Product Code |
| closing_px_date | Date utilize for the closing price |
| total_long | Total Long positions open for a certain contract |
| total_short | Total short positions open for a certain contract |
| total_reserve_ote | Total reserved OTE for a certain contract |
| expiration_time | Expiration time of the contract |
| position_id | Position ID |
| qty | Quantity |
| px | Price |
| notional | Notional |
| reserve_margin_s | Reserve Margin for short position |
| reserve_margin_l | Reserve Margin for long position |
| et | Expiration Time of position |
| customer_account_ref | Customer Account Reference |
| cl_ord_id | Client Order ID |

Example Response:

```
{
  "closeout_id": "E2emX5HhHRk42ksSh1UTcq2Zvasd2UFotDFLQmsw7T1i",
  "report_date": "2020-07-09",
  "time": "2020-07-08T23:17:29.000Z",
  "description": "CLOSEOUT OF 3 ETBTN0 RESULTING IN P&L -0.00007 BTC",
  "account_id": "3cfb77ac-3a71-42c8-bcb5-037ca485616f",
  "product_symbol": "ETBT",
  "contract_symbol": "ETBTN0",
  "fd": "P",
  "realized": "-0.00007",
  "qty": "3.0",
  "contract_code": "JUL 20",
  "expiration_time": "2020-07-31",
  "account_label": "DM-ABCD",
  "closeout_items": [
    {
      "position_id": "1_281474991500193",
      "qty": "2.0",
      "notional": "0.05246",
      "px": "0.02623",
      "p_and_l": "-0.00006",
      "side": "Long",
      "trade_date": "2020-06-23",
      "qty_type": "ETH",
      "px_type": "BTC",
      "trd_type": "REGULAR",
      "trade_report_id": "1125899966000769",
      "trade_id": "1_281474991500193",
      "cl_ord_id": "abdc-15929313433862211",
      "customer_account_ref": null
    },
    {
      "position_id": "1_281474991500370",
      "qty": "1.0",
      "notional": "0.02621",
      "px": "0.02621",
      "p_and_l": "-0.00001",
      "side": "Long",
      "trade_date": "2020-06-23",
      "qty_type": "ETH",
      "px_type": "BTC",
      "trd_type": "REGULAR",
      "trade_report_id": "1125899966001477",
      "trade_id": "1_281474991500370",
      "cl_ord_id": "abdc-15929313733842262",
      "customer_account_ref": null
    },
    {
      "position_id": "2_281474993186795",
      "qty": "3.0",
      "notional": "-0.0786",
      "px": "0.0262",
      "side": "Short",
      "trade_date": "2020-07-09",
      "qty_type": "ETH",
      "px_type": "BTC",
      "trd_type": "REGULAR",
      "trade_report_id": "1125899972747178",
      "trade_id": "2_281474993186795",
      "cl_ord_id": "abdc-15942502490209066",
      "customer_account_ref": null
    }
  ],
}
```

## 3.11 Deposit Address Endpoint

This endpoint will return the address to which a client can deposit funds for a specified digital asset.

- **HTTP Request Type:** POST
- **Endpoint:** /deposit_address
- **API security:** This API endpoint requires an authentication token with Clearing API read permission.

Inputs

| Field | Value |
|---|---|
| account_id | Account ID |
| asset_type | Asset Type |
| funds_designation | Types: N, P, S. (See Funds Designation section for reference) |

Example Request:

```
requests.post(url="https://clearing.erisx.com/api/v1/deposit_address",
              headers={"Authorization": "Bearer " + token},
              json={"account_id": "27ff6d34-523d-476d-9ad5-edeb373b83dc",
                 "asset_type": "BTC",
                  "funds_designation": "N"})
```

Outputs

| Field | Value |
|---|---|
| address | Address hash |
| asset_type | Digital Asset type |

| | |
|---|---|
| account_id | Account ID |
| state | Status of the request |
| funds_designation | Types: N, P, S. (See Funds Designation section for reference) |

Example Response:

```
{
  "address": "2NFVP4gnh4j6GtW8bz2wpXijnWEJ8EAySRq",
  "asset_type": "TBTC",
  "account_id": "ac171a7c-a0de-4e8a-9ce6-8a83d7e3ddd8",
  "state": "succeeded",
  "funds_designation": "N"
}
```

## 3.12 Linked Account Endpoints

This endpoint will return information regarding any digital asset or bank accounts linked to the appropriate clearing member.

- **HTTP Request Type:** POST
- **Endpoint:** /linked_accounts
- **API security:** This API endpoint requires an authentication token with Clearing API read permission.

Inputs

| Field | Value | |
|---|---|---|
| filters (optional) | Default: `"filter": [{ "attr": "account_id", "op": "eq", "value": member_account_id }]` | |
| | id | ID identifying a particular digital asset address or bank account. |
| | member_id | Clearing Member ID |
| | asset_type | Symbol corresponding to the appropriate asset |
| | label | Label given to the linked account when it was input into the system |
| | state | State of the request to add a new linked account. Valid values: pending, approved, rejected |
| | usable_at | Time at which the linked account is ready for the user to use |
| | type | Type of account. Valid values: bank, crypto, collateral |
| offset (optional) | Number of elements to be offset in the request for pagination purposes | |
| limit (optional) | Limit of elements returned in the request | |

Example Request:

```
requests.post(
    url="https://clearing.erisx.com/api/v1/linked_accounts",
    headers={"Authorization": "Bearer " + token},
    json={
        "filter": [{
            "attr": "id",
            "op": "eq",
            "value": ["5e4bef801ef35c09af0b42ce", "5e4bef911ef35c2fbf0b42d0"]
        }],
    })
```

### Outputs

| Field | Value |
|---|---|
| count | Count of linked accounts returned in the response |
| accounts | List of linked accounts |
| label | Label of the linked account as input in the system by the user |
| asset_type | Asset type of the linked account |
| usable_at | Time from which the linked account is usable to use |
| member_id | ID of the clearing member to which the linked account belongs |
| state | Status of the request (pending, approved, rejected) |
| id | ID of the linked account. This id will be the one used in the withdrawal endpoint to specify to which linked accounts the funds be withdrawn |
| created_at | Time at which the linked account was added to the system |
| type | Type of the linked account (crypto, bank or collateral) |
| address | For digital asset's linked accounts, address hash of the wallet |
| account_number | For bank accounts, last 4 digits of account number |
| routing_number | For bank accounts, routing number |
| bank_name | For bank accounts, bank name |

Example Response:

```
{ "count": 2, "accounts": [ {
      "label": "External ETH Wallet",
      "asset_type": "ETH",
      "usable_at": "2020-02-21T14:07:14.855Z",
      "member_id": "5e2b07559228bfd8841fd0ad",
      "state": "pending",
      "address": "bbbbbbb",
      "id": "5e4bef911ef35c2fbf0b42d0",
      "created_at": "2020-02-18T14:07:14.859Z",
      "type": "crypto"
    },{
      "label": "External Checking 0000",
      "asset_type": "USD",
      "usable_at": "2020-02-21T14:07:00.363Z",
      "member_id": "5e2b07559228bfd8841fd0ad",
      "state": "pending",
      "account_number": "0000",
      "routing_number": "011401533",
      "bank_name": "Chase",
      "id": "5e4bef801ef35c09af0b42ce",
      "created_at": "2020-02-18T14:07:00.365Z",
      "type": "bank"
    }]}
```

## 3.13 Withdrawal Request Endpoints

This section describes the procedure for a clearing member to request a withdrawal via the Clearing API. It is composed of two endpoints. First a request to an endpoint will be required where the clearing member specifies the details of the transaction, this request will return a response including all the necessary information that composes a valid transaction. A second request to the second endpoint is then required, where the user will specify the transaction message, which is the response from the first requests, signed by a secure hash of the funding password following the procedure indicated in the Funding Password Signature section.

### 3.13.1 Build Withdrawal Request

This endpoint will enable member users to retrieve all necessary information in order to submit a withdrawal request via the Clearing API.

- **HTTP Request Type:** POST
- **Endpoint:** /build_withdrawal_request
- **API security:** This API endpoint requires an authentication token with Clearing API Funding permissions.
- **Notes:**
    - This request will not initialize a withdrawal request. It will only provide the data required to initialize a withdrawal.
    - The asset of the withdrawal will be inferred based on the Linked Account ID provided. I.e. if the clearing member specified a Linked Account ID that corresponds to a BTC linked account, the endpoint will infer that the clearing member wants to withdraw BTC.

#### Inputs

| **Field** | **Value** |
|---|---|
| account_id | Account ID from which the withdrawal will be made |
| linked_account_id | Linked Account ID to which the withdrawal will be sent. This value can be found in the field ‘id’ from the linked_accounts endpoint response. |
| funds_designation | Types: N, P, S. (See Funds Designation section for reference) |
| amount | Amount that will be withdrawn |

Example Request:

```
requests.post(
    url="https://clearing.erisx.com/api/v1/build_withdrawal_request",
    headers={"Authorization": "Bearer " + token},
    json={'account_id': '48b7d9c5-55c5-4693-b5ec-10a97f7b2333',
          'linked_account_id': '5e4bef4b1ef35c96160b42cb',
          'funds_designation': 'S',
          'amount': '0.001'})
```

### Outputs

| Field | Value |
|---|---|
| account_id | Account ID from which account the withdrawal will be made |
| auth_id | Authentication ID needed to generate the signature in the Submit Withdrawal Request endpoint |
| linked_account_id | Linked Account ID to which the withdrawal will be sent. |
| asset_type | Asset Type of the withdrawal. Inferred based on the linked_account_id provided |
| funds_designation | Types: N, P, S. (See Funds Designation section for reference) |
| amount | Amount of the withdrawal request |
| request_data | Base64-encoded withdrawal transaction as specified with the parameters above. |

Example Response:

```
{
  "auth_id": "auth0|5e2b2eaeb9f8b40eaf22ec20",
  "account_id": "48b7d9c5-55c5-4693-b5ec-10a97f7b2333",
  "linked_account_id": "5e4bef4b1ef35c96160b42cb",
  "asset_type": "TBTC",
  "amount": "0.001",
  "funds_designation": "N",
  "request_data":
"WyI1ZTJiMDc1NTkyMjhiZmQ4ODQxZmQwYWQiLCJhdXRoMHw1ZTJiMmVhZWI5ZjhiNDBlYWYyMmVjMjAiLC
JhYWFhYWFhYSIsIlRCVEMiLCIwLjAwMSIsIjE1ODMxODk2NTk0OTYiLCI0OGI3ZDljNS01NWM1LTQ2OTMtY
jVlYy0xMGE5N2Y3YjIzMzMiLCI1ZTRiZWY0YjFlZjM1Yzk2MTYwYjQyY2IiLCJTIl0="
}
```

### 3.13.2 Submit Withdrawal Request

This endpoint enables member users to submit a withdrawal request.

- **HTTP Request Type:** POST
- **Endpoint:** /submit_withdrawal_request
- **API security:** This API endpoint requires an authentication token with Clearing API Funding permissions as well as Funding Password signature security.

### Inputs

| Field | Value |
|---|---|
| request_data | Base64-encoded transaction data for the withdrawal that will be submitted. This value can be obtained from the response of the endpoint ‘build_withdrawal_request’ in the field ‘transaction_data’ |
| signature | Signature created using the member user’s funding password as described in the section Funding Password Signing |

`Example Request:`

```
requests.post(
      url="https://clearing.erisx.com/api/v1/submit_withdrawal_request",
      headers={"Authorization": "Bearer " + token},
      json={"request_data":
'WyI1ZTJiMDc1NTkyMjhiZmQ4ODQxZmQwYWQiLCJhdXRoMHw1ZTJiMmVhZWI5ZjhiNDBlYWYyMmVjMjAiLC
JhYWFhYWFhYSIsIlRCVEMiLCIwLjAwMSIsIjE1ODMxODk2NTk0OTYiLCI0OGI3ZDljNS01NWM1LTQ2OTMtY
jVlYy0xMGE5N2Y3YjIzMzMiLCI1ZTRiZWY0YjFlZjM1Yzk2MTYwYjQyY2IiLCJTIl0=',
           "signature":
'AN1rKvtNrCF7mWSMbQ3hc4gUtY57C8CREkRNVU3cDdE8QibHgcrvTRmnmjU1SZpX7hDcWr9r8E6Q4Z7HWF
U3JGnTuMDdHPyG4'})
```

Output

| Field | Value |
|---|---|
| request_data | Transaction data for the submitted withdrawal |

`Example Response:`

```
{
  "request_data":
"WyI1ZTJiMDc1NTkyMjhiZmQ4ODQxZmQwYWQiLCJhdXRoMHw1ZTJiMmVhZWI5ZjhiNDBlYWYyMmVjMjAiLC
JhYWFhYWFhYSIsIlRCVEMiLCIwLjAwMSIsIjE1ODMxODk2NTk0OTYiLCI0OGI3ZDljNS01NWM1LTQ2OTMtY
jVlYy0xMGE5N2Y3YjIzMzMiLCI1ZTRiZWY0YjFlZjM1Yzk2MTYwYjQyY2IiLCJTIl0="
}
```

# 4 Block Trade API Service

This API service enables members to submit for processing negotiated Spot and Futures Block Trades in a programmatic way.  All requests and responses are application/json content type.

All Block Trade API endpoints are private and every request needs to be signed using the authentication endpoint described in the Authentication.

## 4.1 REST API Endpoint URL

- **Production: https://clearing.erisx.com/api/v1**
- **New Release (test): https://clearing.newrelease.erisx.com/api/v1**

## 4.2 Block Trade States

A Block Trade submission can have different states during its processing life:

- **Accepted:** Block Trade has been accepted and processed in The Match Engine.
- **Cleared:** Block Trade has been cleared by The Clearing House.
- **Rejected:** Block Trade has been rejected.

## 4.3 Error Codes

Upon submission of a Block Trade through the Submit Block Trade Endpoint, the Block Trade can be rejected for different reasons. In that case, the system will respond with a HTTP error, the response will contain **"state"** equal to **"rejected"** and it will reflect an **"error_code"** specifying the cause of the rejection. The table below explains what each error code represents for easier interpretation:

| Error Code | Rejection Type |
|---|---|
| 1 | Unknown Symbol |
| 2 | Invalid Quantity |
| 3 | Invalid Price |
| 4 | Invalid Negotiated Block Trade Time |
| 5 | Instrument Closed |
| 6 | Trading is Halted |
| 7 | Invalid Buyer Account ID |
| 8 | Invalid Seller Account ID |
| 9 | Buyer Account Not Enabled for Futures |
| 10 | Seller Account Not Enabled for Futures |
| 11 | Buyer Account Not Enabled for Spot |
| 12 | Seller Account Not Enabled for Spot |
| 13 | Duplicate Request |

| | |
|---|---|
| 14 | Buyer Missing Regulatory Field |
| 15 | Seller Missing Regulatory Field |
| 16 | Buyer Insufficient Purchasing Power |
| 17 | Seller Insufficient Purchasing Power |
| 18 | Rejected |

## 4.4 Duplicate Requests

The Match Engine will identify a request as a duplicate if all values specified in the request are identical to a previously sent request. If any of the values specified in the request change, it will not be considered as a duplicate anymore.

**Note:** If a user sends a request, which gets rejected with any error code and then the user attempts to send the exact same request again, the second request will always be rejected due to a duplicate request.

To avoid generating duplicate requests, it is recommended that the users generate new Client Order Ids on every new request, even if the request is a resubmission due to the original request being rejected.

## 4.5 Submit Block Trade Endpoint

This endpoint enables member users to submit a Block Trade.

- **HTTP Request Type:** POST
- **Endpoint:** /submit_block_trade
- **API security:** This API endpoint requires an authentication token with Block Trade API permissions.

Inputs

| Field | Value |
|---|---|
| contract_symbol | Contract Symbol |
| price | Price |
| quantity | Quantity |
| negotiated_time | Time at which the Block Trade was negotiated between the Buyer and Seller |
| sell_side | Details regarding the Seller Account |
| buy_side | Details regarding the Buyer Account |
| account_label | Account Label |
| cl_ord_id | Client Order ID |
| Customer_account_ref (conditional) | Customer Account Reference |

| | |
|---|---|
| sender_sub_id | Sender Sub ID that identifies the Trader that negotiated the Block Trade |
| sender_location | Sender Location that identifies the location of the Trader that negotiated the Block Trade |

Example Request:

```
requests.post(
   url="https://clearing.erisx.com/api/v1/submit_block_trade",
   headers={"Authorization": "Bearer " + token},
   json={
       "contract_symbol": "BTC/USD",
       "price": "1000.0",
       "quantity": "3.0",
       "negotiated_time": "2020-01-01T14:27:00.000Z",
       "sell_side": {
           "account_label": "DM-122221",
           "cl_ord_id": "sell_side_order",
           "customer_account_ref": "sell_side",
           "sender_sub_id": "sell_fix",
           "sender_location": "US,IL"
       },
       "buy_side": {
           "account_label": "DM-77661",
           "cl_ord_id": "buy_side_order",
           "customer_account_ref": "buy_side",
           "sender_sub_id": "buy_fix",
           "sender_location": "US,IL"
       }
   })
```

Output

| Field | Value |
|---|---|
| request_id | Request ID of the Block Trade |
| state | State of the Block Trade submission |
| error_code | Rejection Error Code. See Error Codes for reference. Only on rejected requests |
| email | Submitter's member user email address. Only on rejected requests |

Example Responses:

```
{
  "request_id": "AmZuyvUdJvoa3HshZuhyCosBSvnrxVbZfJr5RtA92EQT",
  "state": "accepted"
}

{
  "error": {
    "request_id": "AmZuyvUdJvoa3HshZuhyCosBSvnrxVbZfJr5RtA92EQT",
    "state": "rejected",
    "error_code": 1,
    "email": "user@email.com"
  }
}
```

Request Block Trade Information Endpoint

This endpoint enables member users to request information regarding a previously submitted Block Trade.

- **HTTP Request Type:** POST
- **Endpoint:** /block_trade_requests
- **API security:** This API endpoint requires an authentication token with Submit Block Trade API permissions.

Inputs

| Field | Value | |
|---|---|---|
| filters (optional) | Default: `"filter": [{ "attr": "account_id", "op": "eq", "value": member_account_id }]` | |
| | request_id | Block Trade submission request ID. Value provided in the response of the [Submit Block Trade Endpoint.](#) |
| | contract_symbol | Contract Symbol |
| | trade_date | Trade Date |
| | state | State of the Block Trade. Values: accepted, cleared, rejected. |
| | account_labels | Account Label(s) |
| offset (optional) | Number of elements to be offset in the request for pagination purposes | |
| limit (optional) | Limit of elements returned in the request | |
| sort (optional) | Default: `"sort": [{ "attr": "time", "value": "desc"}]` | |

`Example Request:`

```
requests.post(
   url="https://clearing.erisx.com/api/v1/block_trade_requests",
   headers={"Authorization": "Bearer " + token},
   json={
       "filter": [{
           "attr": "request_id",
           "op": "eq",
           "value": "5c267f0ee4b0974b5367fd35"
       }, {
           "attr": "contract_symbol",
           "op": "eq",
           "value": "BTC/USD"
       }, {
           "attr": "trade_date",
           "op": "eq",
           "value": "2020-01-01"
       }, {
           "attr": "state",
           "op": "eq",
           "value": "pending"
       }, {
           "attr": "account_labels",
           "op": "eq",
           "value": "27ff6d34-523d-476d-9ad5-edeb373b83dc"
       }]
   })
```

### Output

| Field | Value |
|---|---|
| count | Count of Block Trades that meet the query filter |
| block_trade_requests | List of Block Trades Requests |
| request_id | Block Trade submission request ID |
| contract_symbol | Contract Symbol |
| qty | Quantity |
| px | Price |
| trade_date | Trade Date |
| negotiated_time | Time at which the Block Trade was negotiated between the Buyer and Seller |
| submitted_time | Time at which the Block Trade has submitted |
| created_by | Member user who submitted the Block Trade |
| state | State of the Block Trade |
| buy_side | Details regarding the Seller Account |
| sell_side | Details regarding the Buyer Account |
| account_id | Account ID |
| account_label | Account Label |
| side | Side |
| customer_account_ref | Customer Account Reference |
| cl_ord_id | Client Order ID |
| sender_sub_id | Sender Sub ID that identifies the Trader that negotiated the Block Trade |
| sender_location | Sender Location that identifies the location of the Trader that negotiated the Block Trade |

| | |
|---|---|
| error_code | Rejection Error Code. Only in rejected Block Trades. |

Example Response:

```
{
  "count": 1,
  "block_trade_requests": [
    {
      "request_id": "AmZuyvUdJvoa3HshZuhyCosBSvnrxVbZfJr5RtA92EQT",
      "contract_symbol": "BTC/USD",
      "qty": "5.0",
      "px": "8900.0",
      "trade_date": "2020-01-01",
      "negotiated_time": "2020-01-01T06:00:00.000Z",
      "submitted_time": "2020-01-01T06:00:00.000Z",
      "created_by": "carl.doe@email.com",
      "state": "pending",
      "buy_side": {
        "account_id": "27ff6d34-523d-476d-9ad5-edeb373b83dc",
        "side": "BUY",
        "customer_account_ref": "buy_side",
        "cl_ord_id": "buy_side",
        "sender_sub_id": "buy_side_fix",
        "sender_location": "IL,US"
      },
      "sell_side": {
        "account_id": "73e36d47-0fe6-4bba-84d7-d981d9f9459d",
        "side": "SELL",
        "customer_account_ref": "sell_side",
        "cl_ord_id": "sell_side",
        "sender_sub_id": "buy_side_fix",
        "sender_location": "IL,US"
      }
    }
  ]
}
```

## 4.6 Block Trade Volume Endpoint

This endpoint enables member users to obtain information regarding the total volume that the different products have traded via Block Trades in the ErisX market for the current trade date.

- **HTTP Request Type:** POST
- **Endpoint:** /block_trade_volume
- **API security:** This API endpoint requires an authentication token with Market Data API permissions.

### Inputs

No inputs are required.

Example Request:

```
requests.post(
   url="https://clearing.erisx.com/api/v1/block_trade_volume",
   headers={"Authorization": "Bearer " + token},
   json={})
```

### Output

| Field | Value |
|---|---|
| result | List of Products |
| contract | Contract Symbol |
| volume | Total Volume |
| trade_date | Trade Date |

Example Response:

```
[
    {
      "contract": "BTC/USD",
      "volume": "20.0",
      "trade_date": "2020-01-01"
    }
]
```

# 5 Order Management API Service (NEW)

This API service allows for order management via  RESTful API endpoints. Users can submit, modify, cancel, and query for the status of their orders. This API was designed to conform to the same data model for our FIX and Websocket endpoints.

## 5.1 REST API Base Endpoint URL

- **Production:** https://trade-api.erisx.com/rest-api
- **New Release (test):** https://trade-api.newrelease.erisx.com/rest-api

## 5.2 PartyID

A partyID is a unique identifier that identifies a user and it's tied to a particular account, it is used to identify the account that is performing an order management action.
A partyID is required for all order related messages. An ErisX member may have multiple partyIDs depending on their account setup.

Users should use the Party IDs List endpoint to get a list of partIDs that they are enabled for.

## 5.3 ClOrderID

ClOrderID is a unique identifier which must be submitted with each new order related API call. The ClOrderID must be prepended with the intended user's PartyID. The format of the Client Order ID (**ClOrderID)** must follow the convention below and be unique for the trading session.

**ClOrdID** = PartyID-[user specified value]

The length of the user specified value should not be more than 40 characters.

## 5.4 Time in Force

The supported time in force values are described in the following table.

| Expiry Condition | Description |
|---|---|
| Day | Orders submitted with this expiry condition that have not been executed will be expired by the system at the end of the ErisX trading day in which they were entered. |
| Good Till Cancel (GTC) | Orders with this expiry condition remain open and active until either executed or explicitly canceled by the client. |
| Good Till Date (GTD) | With this time in force, the submitting client specifies the date at which an order is to be expired if not already executed. |
| Fill or Kill (FOK) | Unless the full quantity of the order can be executed immediately at the specified price or better, an order with this expiry condition will be canceled. |
| Immediate or Cancel (IOC) | Orders with the expiry condition will be canceled unless a specified minimum quantity can be executed immediately at the specified price or better. Any remaining unfilled quantity is canceled. |

## 5.5 Order Types

ErisX supports the following order types.

- **Limit** - An order to buy or sell at a specific price or better.
- **Stop-Limit** - An order that combines the features of a stop order and a limit order. The stop price acts as a trigger to enter a limit order into the market.

## 5.6 Minimum Permitted Order Entry Size

There is a minimum permitted order entry size maintained on ErisX platform. Orders sent for amounts less than the permitted minimum order entry size will be rejected.

## 5.7 Timestamping / TransactTime

Messages sent by client applications will need to include TransactTime. The system will validate the value sent down to one second precision and accuracy.

Responses from the match engine will include TransactTime and will be sent with nanosecond precision. The format is YYYYMMDD-HH:MM:ss.SSSSSSSSS.

The timestamp on outgoing messages will represent the time the corresponding message was received by the FIX gateway that resulted in the update.

## 5.8 Table's Legend

| Req | Explanation |
|---|---|
| Y | Field is always required. |
| N | Field is not required. |
| O | Field is optional. |
| C | Field is conditional upon the message type and/or other field values. |
| F | Field is required only for Futures. |

## 5.9 Execution Reports

ErisX sends Execution Report messages to:

- Confirm the receipt of an order
- Confirm changes to an existing order
- Reply to order status messages
- Relay order fill information on active orders

In a normal workflow, ErisX responds with an Execution Report message to indicate that a request has been accepted for the following requests:

- *new-order-single*
- *replace-order-single*
- *cancel-order-single*
- *cancel-all*

ErisX will respond with an Execution Report indicating the status of an order for the following requests:

- *order*
- *order-mass-status*

For a multiple filled order, the ExecType (150) field reports information on the individual fill and the OrdStatus (39) field reports information on the overall order status.

| Field | Req | Value |
|---|---|---|
| correlation | Y | Alphanumeric string submitted by users or generated by ErisX if user does not submit a correlation value in the request |
| type | Y | **ExecutionReport** |
| orderID | Y | Unique order identifier assigned by ErisX |
| clOrdID | Y | Client assigned order id |
| origClOrdID | Y | Original client assigned order id submitted on the order. |
| execID | Y | Completed trade identifying number. |
| execType | Y | The execution report's type. New, Canceled, Replace, Rejected, Expired, Fill Status,Order Status |
| ordStatus | Y | The current state of chain of orders. **NEW**, **PARTIAL FILLED**, **FILLED**, **CANCELED**, **REPLACED**, **REJECTED**, **EXPIRED** |
| OrdRejReason | O | Optional when Rejected. |
| account | O | The clearing account name as agreed to by ErisX and the client or else defaulted by the system. |
| symbol | Y | The order currency pair. |
| side | Y | Order side: **BUY** or **SELL** |
| orderQty | Y | Order quantity. Not sent if Canceled or Rejected. |
| ordType | O | Supported values are: Limit order or Stop-Limit order |
| price | Y | Required for Limit orders |
| stopPrice | C | The price at which a stop order becomes effective. |
| currency | O | The currency for the amount specified in OrderQty field. |
| lastQty | O | Quantity bought/sold for this fill. Present when ExecType = F. |
| lastPrice | O | Price at which the current or last fill was made. |
| lastSpotRate | O | Price for the last fill. Not sent for status requests |
| leavesQty | Y | Amount of order open for further execution. |
| cumQty | Y | Total amount of an order currently executed in a chain of partial fills. |
| avgPrice | O | The average price at which the order was filled or partially filled. |
| tradeDate | O | Trades completed after 4 pm CT show the next business day. |
| transactTime | O | Execution Reports will be sent with nanosecond precision - YYYYMMDD-HH:MM:SS.sssssssss |
| commission | O | Actual Commission (Only for Fills and Partial Fills) |
| commCalculated | Y | Calculated Commission |
| commType | Y | 3 = Absolute (Total monetary amount) |
| commCurrency | Y | Currency Commission (USD, BTC) |

| | | |
|---|---|---|
| minQty | O | Minimum Quantity to be filed for IOC orders. |
| text | O | Descriptive text message |
| matchingType | O | Indicates whether or not the maker's price was resting in the book at the time of the match. |
| lastRptRequested | O | In response to the Mass Order Status request. **True** on the last execution report to report. |
| massStatusReqID | C | Mass status unique request ID |
| timeInForce | Y | How long an order remains in effect: **Day**, **Good Till Cancel**, **Good Till Date**, **FillOrKill**, **ImmediateOrCancel** |
| expireDate | Y | Expiry date in YYYYMMDD format. |
| partyIDs | O | Party ID of the account |
| sendingTime | O | Time at which the request was sent by the member user. |
| accountType | F | **1**=Customer, **2**=House |
| custOrderCapacity | F | CTICode (customer type indicator)<br>**1** = Member Trading for own account<br>**2** = Clearing firm trading for its Prop Account<br>**3** = Member trading for another member<br>**4** = All other |
| targetLocationId | F | Used to identify the geographical location of the user that entered the order: [Country],[State if in US] eg; US, IL or UK |
| customerAccountRef | O | Customer Account Reference (FCM Back office Account) |
| targetSubId | F | Value used to identify the user that entered the order |
| postOnly | O | Indicate if the newOrderSingle is Post-Only or not.<br>**N** = No Post-Only type.<br>**Y** = Post-Only type. |
| unsolicitedCancel | O | Flag to identify if the cancel was unsolicited |

## 5.10 Request Rejected

In case a request is rejected, ErisX will respond with an rejection type message, indicating that the request was rejected and specifying reject reason.

| **Field** | **Req** | **Value** |
|---|---|---|
| correlation | O | Alphanumeric string submitted by users or generated by ErisX if user does not submit a correlation value in the request |
| type | Y | Rejection Type |
| clOrdID | Y | Client assigned order id |
| message | O | Message indicating Reject Reason |
| requestType | O | Type of Response that was rejected |
| rejectTime | O | Time at which the rejection took place |
| orderID | O | Unique order identifier assigned by ErisX |
| origClOrdID | O | Original client assigned order id submitted on the order. |
| ordStatus | O | The current state of chain of orders. **REJECTED** |

| | | |
|---|---|---|
| account | O | The clearing account name as agreed to by ErisX and the client or else defaulted by the system. |
| transactTime | O | Execution Reports will be sent with nanosecond precision - YYYYMMDD-HH:MM:SS.sssssssss |
| cxlRejResponseTo | O | Message type of rejected request |
| cxlRejReason | O | Rejection Reason |
| text | O | Text with description of rejection reason |
| targetSubId | O | Value used to identify the user that entered the order |
| targetLocationID | O | Used to identify the geographical location of the user that entered the order: [Country],[State if in US] eg; US, IL or UK |
| error | O | Error type |
| details | O | Details of the rejected order |

### Example

```
{
  "correlation": "foo123",
  "type": "OrderReject",
  "ordStatus": "REJECTED",
  "clOrdID": "Trader_A_OM-15891446998782",
  "message": "INSTRUMENT CLOSED",
  "requestType": "ExecutionReport",
  "rejectTime": "20200510-21:04:59.956"
}

 Or

{
  "correlation": "foo123",
  "type": "OrderCancelReject",
  "orderID": "281474976751461",
  "clOrdID": "Trader_A-158915190894326",
  "origClOrdID": "Trader_A-15891453855913",
  "ordStatus": "REJECTED",
  "account": null,
  "transactTime": "20200510-23:05:08.969609029",
  "cxlRejResponseTo": "ORDER_CANCEL_REPLACE_REQUEST",
  "cxlRejReason": "BROKER_EXCHANGE_OPTION",
  "text": "INVALID/MISSING FIELD - TAG 21",
  "senderSubId": null,
  "senderLocationId": null
}

Or
```

```
{
  "correlation": "133348085259403",
  "type": "ERROR_MESSAGE",
  "error": "Invalid symbol",
  "details": "clOrdID = Trader_A_OM-1592898202181"
}


Or

{
  "correlation": "1234",
  "type": "AuthenticationError",
  "message": "authentication failed"
}
```

## 5.11 New Order Single

This endpoint enables member users to submit a new order into the CLOB.

- **HTTP Request Type:** POST
- **Endpoint:** /new-order-single
- **API security:** This API endpoint requires an authentication token with Trading API permissions.

### Inputs

| Field | Req | Value |
|---|---|---|
| correlation | O | Alphanumeric string |
| clOrdID | Y | Must start with **partyID-**. The partyId should be a real party ID.<br>Maximum length = partyID-[+40 characters] |
| currency | Y | The currency for the amount specified in the OrderQty. |
| side | Y | **BUY or SELL** |
| symbol | Y | Unique instrument identifier |
| partyID | Y | Party ID of the account |
| transactionTime | Y | See TransactTime description below. Time at which the order was submitted. |
| orderQty | Y | Order Quantity |
| ordType | Y | **LIMIT** or **STOP_LIMIT** |
| price | Y | Order price |
| stopPrice | C | Required if ordType = STOP_LIMIT<br>Stop price of the Stop Limit Order. For a buy order, the stop price must be set at least one tick below the limit price. For a sell order the stop price must be set at least one tick above the limit price. |
| timeInForce | O | **Day**(default), **GoodTillCancel**, **GoodTillDate, FillOrKill, ImmediateOrCancel** |
| expireDate | C | Only available for GoodTillCancel order. UTC format **YYYYMMDD** |

| | | |
|---|---|---|
| accountType | F | **1**=Customer, **2**=House |
| custOrderCapacity | F | CTICode (customer type indicator)<br>**1** = Member Trading for own account<br>**2** = Clearing firm trading for its Prop Account<br>**3** = Member trading for another member<br>**4** = All other |
| senderLocationId | F | Used to identify the geographical location of the user that entered the order: [Country],[State if in US] eg; US, IL or UK |
| customerAccountRef | O | Customer Account Reference (FCM Back office Account) |
| senderSubId | F | Value used to identify the user that entered the order |
| postOnly | O | Indicate if the newOrderSingle is Post-Only or not.<br>**N** = No Post-Only type (default).<br>**Y** = Post-Only type. |

Example Request:

```
requests.post(
   url="https://trade-api.erisx.com/rest-api/new-order-single",
   headers={"Authorization": "Bearer " + token},
   json={
       "clOrdID": "Trader_A-15891446998782",
       "currency": "BTC",
       "side": "BUY",
       "symbol": "BTCM0",
       "partyID": "Trader_A",
       "transactionTime": "20200510-21:04:59.878",
       "orderQty": "2",
       "ordType": "LIMIT",
       "price": "8500",
       "timeInForce": "Day",
       "accountType": "2",
       "custOrderCapacity": "1",
       "senderLocationID": "US,IL",
       "senderSubID": "Trader1",
       "postOnly": "Y"
   })
```

## Output

Example Response:

```
{
  "correlation": "foo234",
  "type": "ExecutionReport",
  "orderID": "281474976751461",
  "clOrdID": "Trader_A-15891453855913",
  "origClOrdID": "Trader_A-15891453855913",
  "execID": "281474976981534",
  "execType": "NEW",
  "ordStatus": "NEW",
  "ordRejReason": null,
  "account": "accountA",
  "symbol": "BTCM0",
  "side": "BUY",
  "orderQty": 2.0,
  "ordType": "LIMIT",
  "price": 8500.0,
  "stopPrice": 0.0,
  "currency": "BTC",
  "lastPrice": 0.0,
  "lastSpotRate": 0.0,
  "leavesQty": 2.0,
  "cumQty": 0.0,
  "avgPrice": 0.0,
  "tradeDate": null,
  "transactTime": "20200510-21:16:25.657163984",
  "commission": 0.0,
  "commCalculated": 0.0,
  "commType": "ABSOLUTE",
  "commCurrency": "USD",
  "minQty": 0.0,
  "text": null,
  "matchingType": null,
  "lastRptRequested": null,
  "timeInForce": "Day",
  "expireDate": "20200511",
  "lastQty": 0.0,
  "partyIDs": [
    "Trader_A"
  ],
  "sendingTime": "20200510-21:16:25.687",
  "targetLocationId": "US,IL,
  "custOrderCapacity": 1,
  "accountType": 2,
  "targetSubId": "Trader1,
  "customerAccountRef": null,
  "postOnly": "Y",
  "unsolicitedCancel": null
}
```

## 5.12 Replace Order Single

This endpoint enables member users to replace orders in the CLOB.

- **HTTP Request Type:** POST
- **Endpoint:** /replace-order-single

- **API security:** This API endpoint requires an authentication token with Trading API permissions.

Order parameters such as price, quantity and expiry condition can be amended on an outstanding order without having to cancel and resubmit the order.

By default, orders that have been partially filled cannot be modified unless the overfill protection logic is used. See section Overfill protection. A reject message will be received if attempting to modify a partially filled order without the use of overfill protection.

When modifying an existing order the associated IDs (origClOrdID and OrderID) are required.

Inputs

| Field | Req | Value |
|---|---|---|
| correlation | O | Alphanumeric string |
| clOrdID | Y | Must start with **partyID-**. The partyID should be a real party ID corresponding to the API credentials.<br>Maximum length = partyID-[+40 characters] |
| origClOrdID | Y | Must be the client ID of the original submitted order. |
| orderID | Y | Must be ErisX assigned ID of the original order. |
| currency | Y | Quote Currency of the product |
| side | Y | BUY or SELL |
| symbol | Y | Unique instrument identifier |
| timeInForce | O | **Day**(Default), **GoodTillCancel**, **GoodTillDate,** **FillOrKill,** **ImmediateOrCancel** |
| expireDate | C | Only available for GoodTillCancel order. UTC format **YYYYMMDD** |
| partyID | Y | Must match the partyID- on the clOrdID |
| transactionTime | Y | See TransactTime description below. Time at which the order was submitted. |
| orderQty | O | Order Quantity |
| price | O | Order price |
| ordType | Y | **LIMIT** or **STOP_LIMIT** |
| stopPrice | C | Stop price of the Stop Limit Order. For a buy order, the stop price must be set at least one tick below the limit price. For a sell order the stop price must be set at least one tick above the limit price. |
| overfillProtection | O | Required when trying to modify a partially filled order to specifically request "Overfill Protection" otherwise the modification is rejected.<br>**Y** = LeavesQty is set to requested quantity - CumQty<br>**N** = LeavesQty is set to the quantity requested in the cancel replace message |
| accountType | F | **1**=Customer, **2**=House |
| custOrderCapacity | F | CTICode (customer type indicator)<br>**1** = Member Trading for own account<br>**2** = Clearing firm trading for its Prop Account<br>**3** = Member trading for another member<br>**4** = All other |
| senderLocationId | F | Used to identify the geographical location of the user that entered the |

| | | |
|---|---|---|
| | | order: [Country],[State if in US] eg; US, IL or UK |
| customerAccountRef | O | Customer Account Reference (FCM Back office Account) |
| senderSubId | F | Value used to identify the user that entered the order |
| postOnly | O | Indicate if the newOrderSingle is Post-Only or not.<br>**N** = No Post-Only type (default).<br>**Y** = Post-Only type. |
| handlInst | O | **AutomatedExecutionOrderPrivate** (default) |

Example Request:

```
requests.post(
   url="https://trade-api.erisx.com/rest-api/replace-order-single",
   headers={"Authorization": "Bearer " + token},
   json={
       "clOrdID": "Trader_A-15891462613396",
       "origClOrdID": "Trader_A-15891453855913",
       "orderID": "281474976751461",
       "currency": "BTC",
       "side": "BUY",
       "symbol": "BTCM0",
       "partyID": "Trader_A",
       "transactionTime": "20200510-21:31:01.339",
       "ordType": "LIMIT",
       "handlInst": "AutomatedExecutionOrderPrivate",
       "orderQty": 3,
       "price": 8501,
       "timeInForce": "Day",
       "accountType": "2",
       "custOrderCapacity": "1",
       "senderLocationID": "US,IL",
       "senderSubID": "Trader1",
       "postOnly": "N"
   })
```

Output:

Example Response:

```
{
  "correlation": "foo345",
  "type": "ExecutionReport",
  "orderID": "281474976751461",
  "clOrdID": "Trader_A-15891462613396",
  "origClOrdID": "Trader_A-15891453855913",
  "execID": "281474976981536",
  "execType": "REPLACE",
  "ordStatus": "REPLACED",
  "ordRejReason": null,
  "account": "accountA",
  "symbol": "BTCM0",
  "side": "BUY",
  "orderQty": 3.0,
  "ordType": "LIMIT",
  "price": 8501.0,
  "stopPrice": 0.0,
  "currency": "BTC",
  "lastPrice": 0.0,
  "lastSpotRate": 0.0,
  "leavesQty": 3.0,
  "cumQty": 0.0,
  "avgPrice": 0.0,
  "tradeDate": null,
  "transactTime": "20200510-21:31:01.370919967",
  "commission": 0.0,
  "commCalculated": 0.0,
  "commType": "ABSOLUTE",
  "commCurrency": "USD",
  "minQty": 0.0,
  "text": null,
  "matchingType": null,
  "lastRptRequested": null,
  "timeInForce": "Day",
  "expireDate": "20200511",
  "lastQty": 0.0,
  "partyIDs": [
    "Trader_A"
  ],
  "sendingTime": "20200510-21:31:01.401",
  "targetLocationId": "US,IL",
  "custOrderCapacity": 1,
  "accountType": 2,
  "targetSubId": "Trader1",
  "customerAccountRef": null,
  "postOnly": "N",
  "unsolicitedCancel": null
}
```

## 5.13 Cancel Order Single

This endpoint enables member users to cancel an order in the CLOB.

- **HTTP Request Type:** POST
- **Endpoint:** /cancel-order-single
- **API security:** This API endpoint requires an authentication token with Trading API permissions.

When cancelling an existing order the associated IDs (origClOrdID and OrderID) are required.

### Inputs

| Field | Req | Value |
|---|---|---|
| correlation | O | Alphanumeric string |
| clOrdID | Y | Must start with **partyID-**. The partyID should be a real party ID corresponding to the API credentials.<br>Maximum length = partyID-[+40 characters] |
| origClOrdID | Y | Must be the client ID of the original submitted order. |
| orderID | Y | Must be ErisX assigned ID of the original order. |
| side | Y | BUY or SELL |
| symbol | Y | Unique instrument identifier |
| partyID | Y | Must match the partyID- on the clOrdID |
| transactionTime | Y | See TransactTime description below. Time at which the order was submitted. |
| ordType | Y | **LIMIT** or **STOP_LIMIT** |

### Example Request:

```
requests.post(
    url="https://trade-api.erisx.com/rest-api/cancel-order-single",
    headers={"Authorization": "Bearer " + token},
    json={
        "clOrdID": "Trader_A_OM-15891468713728",
        "origClOrdID": "Trader_A_OM-15891462613396",
        "orderID": "281474976751461",
        "side": "BUY",
        "symbol": "BTCM0",
        "partyID": "Trader_A_OM",
        "transactionTime": "20200510-21:41:11.371",
        "ordType": "LIMIT"
    })
```

### Outputs

### Example Response:

```
{
  "correlation": "foo456",
  "type": "ExecutionReport",
  "orderID": "281474976751461",
  "clOrdID": "Trader_A-15891468713728",
  "origClOrdID": "Trader_A-15891462613396",
  "execID": "281474976981538",
  "execType": "CANCELED",
  "ordStatus": "CANCELED",
  "ordRejReason": null,
  "account": "accountA",
  "symbol": "BTCM0",
  "side": "BUY",
  "orderQty": 3.0,
  "ordType": "LIMIT",
  "price": 8501.0,
  "stopPrice": 0.0,
  "currency": "BTC",
  "lastPrice": 0.0,
  "lastSpotRate": 0.0,
  "leavesQty": 0.0,
  "cumQty": 0.0,
  "avgPrice": 0.0,
  "tradeDate": null,
  "transactTime": "20200510-21:41:11.397795823",
  "commission": 0.0,
  "commCalculated": 0.0,
  "commType": "ABSOLUTE",
  "commCurrency": "USD",
  "minQty": 0.0,
  "text": "USER INITIATED",
  "matchingType": null,
  "lastRptRequested": null,
  "timeInForce": "Day",
  "expireDate": "20200511",
  "lastQty": 0.0,
  "partyIDs": [
    "Trader_A"
  ],
  "sendingTime": "20200510-21:41:11.427",
  "targetLocationId": "US,IL",
  "custOrderCapacity": 1,
  "accountType": 2,
  "targetSubId": "Trader1",
  "customerAccountRef": null,
  "postOnly": "N",
  "unsolicitedCancel": null
}
```

## 5.14 Cancel All Orders

This endpoint enables member users to cancel all working orders in the CLOB with a single request.

- **HTTP Request Type:** POST
- **Endpoint:** /cancel-all
- **API security:** This API endpoint requires an authentication token with Trading API permissions.

When cancelling an existing order the associated IDs (origClOrdID and OrderID) are required.

### Inputs

| Field | Req | Value |
|---|---|---|
| correlation | O | Alphanumeric string |
| partyID | Y | Must match the partyID- on the clOrdID |

Example Request:

```
requests.post(
    url="https://trade-api.erisx.com/rest-api/cancel-all",
    headers={"Authorization": "Bearer " + token}, json={"partyID": "trader1"})
```

### Outputs

| Field | Req | Value |
|---|---|---|
| correlation | O | Alphanumeric string |
| type | Y | CancelAllOrdersResponse |
| partyID | Y | Must match the partyID- on the clOrdID |
| message | Y | Message indicating whether the cancel all has been accepted or not. **Accepted**, **Rejected** |

Example Response:

```
{
 "correlation": "5535410536258384820",
 "type": "CancelAllOrdersResponse",
 "partyId": "trader1",
 "message": "Accepted"
}
```

## 5.15 Order Status Single

This endpoint enables member users to request information by order ID. It will return the latest Execution Report known for the order ID.

**Note:** Non-working orders will be available for queries within the same trade date.

- **HTTP Request Type:** GET
- **Endpoint:** /order/{partyID}/{orderID}
- **API security:** This API endpoint requires an authentication token with Trading API permissions.

### Inputs

No inputs required

Example Request:

```
requests.get(
    url="https://trade-api.erisx.com/rest-api/order/Trader_A_OM/281474976751461",
    headers={"Authorization": "Bearer " + token})
```

### Outputs

**Example Response:**

```
{
  "correlation": "foo123",
  "type": "ExecutionReport",
  "orderID": "281474976751461",
  "clOrdID": "Trader_A-15891468713728",
  "origClOrdID": "Trader_A-15891462613396",
  "execID": "281474976981538",
  "execType": "CANCELED",
  "ordStatus": "CANCELED",
  "ordRejReason": null,
  "account": "acc4",
  "symbol": "BTCM0",
  "side": "BUY",
  "orderQty": 3.0,
  "ordType": "LIMIT",
  "price": 8501.0,
  "stopPrice": 0.0,
  "currency": "BTC",
  "lastPrice": 0.0,
  "lastSpotRate": 0.0,
  "leavesQty": 0.0,
  "cumQty": 0.0,
  "avgPrice": 0.0,
  "tradeDate": null,
  "transactTime": "20200510-21:41:11.397795823",
  "commission": 0.0,
  "commCalculated": 0.0,
  "commType": "ABSOLUTE",
  "commCurrency": "USD",
  "minQty": 0.0,
  "text": "USER INITIATED",
  "matchingType": null,
  "lastRptRequested": null,
  "timeInForce": "Day",
  "expireDate": "20200511",
  "lastQty": 0.0,
  "partyIDs": [
    "Trader_A"
  ],
  "sendingTime": "20200510-21:41:11.427",
  "targetLocationId": "US,IL",
  "custOrderCapacity": 1,
  "accountType": 2,
  "targetSubId": "Trader1",
  "customerAccountRef": null,
  "postOnly": "N",
  "unsolicitedCancel": null
}
```

## 5.16 Order Mass Status

This endpoint enables member users to request information of all working orders.

- **HTTP Request Type:** POST
- **Endpoint:** /order-mass-status
- **API security:** This API endpoint requires an authentication token with Trading API permissions.

Inputs

| Field | Req | Value |
|---|---|---|
| correlation | O | Alphanumeric string |
| partyID | Y | Must match the partyID- on the clOrdID |

Example Request:

```
requests.post(
    url="https://trade-api.erisx.com/rest-api/order-mass-status",
    headers={"Authorization": "Bearer " + token}, json={"partyID": "Trader_A"})
```

Outputs

| Field | Req | Value |
|---|---|---|
| correlation | O | Alphanumeric string |
| type | Y | **MassOrderStatus** |
| orderStatus | Y | List of ExecutionReports for all working orders |

Example Response:

```
{
  "type": "MassOrderStatus",
  "correlation": "foo678",
  "orderStatuses": [
    {
      "correlation": "foo123",
      "type": "ExecutionReport",
      "orderID": "281474976751464",
      "clOrdID": "Trader_A-158914731055411",
      "origClOrdID": "Trader_A-158914731055411",
      "execID": "0",
      "execType": "ORDER_STATUS",
      "ordStatus": "NEW",
      "ordRejReason": null,
      "account": "accountA",
      "symbol": "BTC/USD",
      "side": "BUY",
      "orderQty": 0.1,
      "ordType": "LIMIT",
      "price": 8500.0,
      "stopPrice": 0.0,
      "currency": "BTC",
      "lastPrice": 0.0,
      "lastSpotRate": 0.0,
      "leavesQty": 0.1,
      "cumQty": 0.0,
      "avgPrice": 0.0,
      "tradeDate": null,
      "transactTime": "20200510-21:48:30.588817127",
      "commission": 0.0,
      "commCalculated": 0.0,
      "commType": null,
      "commCurrency": null,
      "minQty": 0.0,
      "text": null,
      "matchingType": null,
      "lastRptRequested": "N",
      "timeInForce": "Day",
      "expireDate": "20200511",
      "lastQty": 0.0,
      "partyIDs": [
        "Trader_A"
      ],
      "sendingTime": "20200510-22:27:09.989",
      "targetLocationId": null,
      "custOrderCapacity": 0,
      "accountType": 0,
      "targetSubId": null,
      "customerAccountRef": null,
      "postOnly": "N",
      "unsolicitedCancel": null
    },
    ...
    {
      "correlation": "foo1231",
      "type": "ExecutionReport",
      "orderID": "281474976751463",
      "clOrdID": "Trader_A-158914718167310",
      "origClOrdID": "Trader_A-158914718167310",
      "execID": "0",
      "execType": "ORDER_STATUS",
      "ordStatus": "NEW",
      "ordRejReason": null,
      "account": "accountA",
      "symbol": "BTC/USD",
      "side": "BUY",
      "orderQty": 0.1,
      "ordType": "LIMIT",
      "price": 8500.0,
      "stopPrice": 0.0,
      "currency": "BTC",
      "lastPrice": 0.0,
      "lastSpotRate": 0.0,
      "leavesQty": 0.1,
      "cumQty": 0.0,
      "avgPrice": 0.0,
      "tradeDate": null,
      "transactTime": "20200510-21:46:21.755875910",
      "commission": 0.0,
      "commCalculated": 0.0,
      "commType": null,
      "commCurrency": null,
      "minQty": 0.0,
      "text": null,
      "matchingType": null,
      "lastRptRequested": "N",
      "timeInForce": "Day",
      "expireDate": "20200511",
      "lastQty": 0.0,
      "partyIDs": [
        "Trader_A"
      ],
      "sendingTime": "20200510-22:27:09.990",
      "targetLocationId": null,
      "custOrderCapacity": 0,
      "accountType": 0,
      "targetSubId": null,
      "customerAccountRef": null,
      "postOnly": "N",
      "unsolicitedCancel": null
    }
  ]
}
```

## 5.17 Security List

This endpoint enables member users to get all available symbols.

- **HTTP Request Type:** GET
- **Endpoint:** /security-list
- **API security:** This API endpoint requires an authentication token with Trading API permissions.

### Inputs

No inputs required

Example Request:

```
requests.get(
    url="https://trade-api.erisx.com/rest-api/security-list",
    headers={"Authorization": "Bearer " + token})
```

### Outputs

| Field | Req | Value |
|---|---|---|
| correlation | O | Alphanumeric string |
| type | Y | **SecuritiesResponse** |
| currency | Y | Currency |
| symbol | Y | Symbol |
| SecurityDesc | O | Contract Description |
| minTradeVol | Y | Minimum order quantity |
| maxTradeVol | Y | Maximum Order Quantity |
| roundLot | Y | Minimum order quantity increment |
| minPriceIncrement | Y | Minimum price increment |
| product | Y | Product type |
| cfiCode | F | CFI Code |
| securityType | O | Contract type |
| maturityMonthYear | F | Contract Maturity month and year |
| contractMultiplier | F | Contract multiplier |
| securityExchange | F | Exchange where contract is listed |
| activation | F | Date when contract becomes active |
| lastEligibleTradeDate | F | Last Trade Date for contract |
| maturityDate | F | Contract Maturity Date YYYYMMDD |
| lastTradeTime | F | Contract Last Trade Time (in UTC) on the maturity date |
| expiryTime | F | Contract Expiry Time (in UTC) on the maturity date |

Example Response:

```
{
  "correlation": "foo123",
  "type": "SecuritiesResponse",
  "securities": [
    {
      "currency": "BTC",
      "symbol": "BTC/USD",
      "securityDesc": "BTC/USD",
      "minTradeVol": 0.01,
      "maxTradeVol": 100000,
      "roundLot": 0.01,
      "minPriceIncrement": 1,
      "product": "COMMODITY",
      "cfiCode": null,
      "securityType": null,
      "maturityMonthYear": null,
      "contractMultiplier": 0,
      "securityExchange": null,
      "activation": null,
      "lastEligableTradeDate": null,
      "maturityDate": null,
      "lastTradeTime": null,
      "expiryTime": null
    },
    ...
    {
      "currency": "BTC",
      "symbol": "BTCM0",
      "securityDesc": "BTCM0",
      "minTradeVol": 2,
      "maxTradeVol": 100000,
      "roundLot": 1,
      "minPriceIncrement": 1,
      "product": "COMMODITY",
      "cfiCode": "FCXXSX",
      "securityType": "FUT",
      "maturityMonthYear": "202006",
      "contractMultiplier": 0.1,
      "securityExchange": "ERISX",
      "activation": "20200131",
      "lastEligableTradeDate": "20200626",
      "maturityDate": "20200626",
      "lastTradeTime": "21:00:00Z",
      "expiryTime": "21:00:00Z"
    }
  ]
}
```

## 5.18 Security Status

This endpoint enables member users to status information regarding a specific contract.

- **HTTP Request Type:** GET
- **Endpoint:** /security-status

- **API security:** This API endpoint requires an authentication token with Trading API permissions.

### Inputs

| Field | Req | Value |
|---|---|---|
| symbol | Y | Symbol |

**Note:** For symbols that contain /, an appropriate encoding of the symbol needs to be made in order to obtain a valid URL.

`Example Request:`

```
requests.get(
    url="https://trade-api.erisx.com/rest-api/security-status",
    headers={"Authorization": "Bearer " + token}, params={"symbol": "BTC/USD" })
```

### Outputs

| Field | Req | Value |
|---|---|---|
| correlation | O | Alphanumeric string |
| type | Y | SecurityStatus |
| security | Y | Contract specification as described in Security List |
| securityTradingStatus | Y | Current Contract Trading Status: **READY_TO_TRADE_START_OF_SESSION**, **NOT_AVAILABLE_FOR_TRADING_END_OF_SESSION**, **TRADING_HALT**, **PRE_OPEN** |
| sessionEnd | Y | Status which indicates that a trading session has ended and statistics for the trading session should be reset |
| sendingTime | Y | Time at which the message was published from ErisX |
| transactTime | Y | Time at which the trading engine performed an action |

`Example Response:`

```
{
  "type": "SecurityStatus",
  "security": {
    "currency": "BTC",
    "symbol": "BTC/USD",
    "securityDesc": "BTC/USD",
    "minTradeVol": 0.01,
    "maxTradeVol": 100000,
    "roundLot": 0.0001,
    "minPriceIncrement": 1,
    "product": "COMMODITY",
    "cfiCode": null,
    "securityType": null,
    "maturityMonthYear": null,
    "contractMultiplier": 0,
    "securityExchange": null,
    "activation": null,
    "lastEligableTradeDate": null,
    "maturityDate": null,
    "lastTradeTime": null,
    "expiryTime": null
  },
  "securityTradingStatus": "READY_TO_TRADE_START_OF_SESSION",
  "sessionEnd": "CHANGE_OF_TRADING_SESSION",
  "correlation": "123456",
  "sendingTime": "20200618-21:00:00.037",
  "transactTime": "20200618-21:00:00.001000000",
}
```

## 5.19 Party IDs List

This endpoint enables member users to get a list of all available Party IDs to the member user.

- **HTTP Request Type:** GET
- **Endpoint:** /party-list
- **API security:** This API endpoint requires an authentication token with Trading API permissions.

### Inputs

No inputs required

Example Request:

```
requests.get(
    url="https://trade-api.erisx.com/rest-api/party-list",
    headers={"Authorization": "Bearer " + token})
```

### Outputs

| Field | Req | Value |
|---|---|---|
| correlation | Y | Alphanumeric string |
| type | Y | **PartyListResponse** |
| partyIds | Y | List of available trading party IDs |

Example Response:

```
{
  "partyIds": [
    "trader1"
  ],
  "correlation": "9182569783680059937",
  "type": "PartyListResponse"
}
```