



ErisX Clearing WebAPI V1.0

Please contact ErisX sales representatives or help desk personnel for more information on this documentation.

## Contents

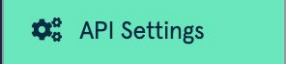
<b>General Concepts</b>	<b>2</b>
API Credentials	2
API Key permissions	2
Authentication	2
<b>Clearing Service</b>	<b>5</b>
REST API Endpoint URL	5
Filters	5
Sorting	5
Pagination	6
Rate Limiting	6
Trade Date and Business Date	6
<b>Clearing Methods</b>	<b>7</b>
Accounts	7
Balances	8
Movements	10
Trades	12
Asset Movement Status Requests	14
<b>Change History</b>	<b>17</b>

## 1 General Concepts

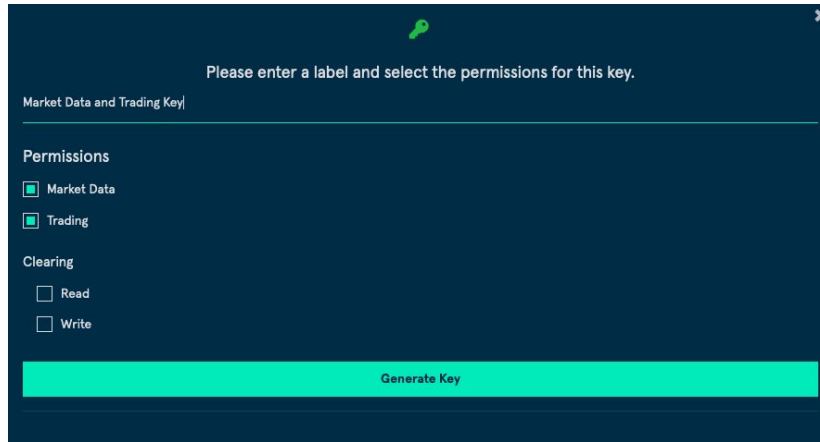
---

### 1.1 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 .

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



The screenshot shows a dark-themed form titled "Please enter a label and select the permissions for this key." with a close button (X) in the top right corner. Below the title is a text input field containing "Market Data and Trading Key". Underneath, there are three sections of permissions, each with a checkbox and a label: "Market Data" (checked), "Trading" (checked), and "Clearing" (unchecked). The "Clearing" section has two sub-options: "Read" (unchecked) and "Write" (unchecked). At the bottom of the form is a large red button labeled "Generate Key".

#### API Key permissions

- **Market Data:** An API key can query historical data or subscribe to real time data (Coming soon).
- **Trading:** Allows an API to enter, modify and cancel orders (Coming soon).
- **Clearing:** Allows an API key to query information about their clearing accounts.

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**

### 1.2 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/>).
- In python 3 you will need to use the **decode('utf-8')** function to convert the token from a bytes like object to a string.
- 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/${method_name}`,
  timeout: 30000, // 30 seconds
  headers: {
    Authorization: `Bearer ${token}`,
  },
  data: {}
});
```

#### Python 2 Example:

```
import jwt
import time

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')

my_secret = '31b6b61606588580'
my_api_key = '9106676d85f1163f.d1ba2efac8bc1e0a'
url = 'https://clearing.erisx.com/api/v1/'
token = gen_token(my_secret, my_api_key)
requests.post(url=url + method_name, headers={'Authorization': 'Bearer ' + token}, json={}) # Be aware that there is a blank space after Bearer
```

#### Python 3 Example:

```
import jwt
import time

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 = '9106676d85f1163f.d1ba2efac8bc1e0a'
url = 'https://clearing.erisx.com/api/v1/'
token = gen_token(my_secret, my_api_key)
requests.post(url= url + method_name, headers={'Authorization': 'Bearer ' +
token}, json={}) # Be aware that there is a blank space after Bearer
```

## 2 Clearing 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 methods are private and every request needs to be signed using the authentication method described.

### 2.1 REST API Endpoint URL

- <https://clearing.erisx.com/api/v1/>

### 2.2 Filters

Some methods 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: 'eq' - equal 'ne' - not equal 'gt' - greater than 'gte' - greater than or equal 'lt' - less than 'lte' - less than or equal
value	Value of the attribute to which the query will compare. It may also be an array of values.

### 2.3 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

	'asc' - ascending
--	-------------------

## 2.4 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 methods except in the Balances method.

```
"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).

## 2.5 Rate Limiting

Requests are throttled per IP address. Limit: 15 requests per second.

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

If the limit is exceeded the IP address will not be allowed to make new requests for a 5 minute period.

## 2.6 Trade Date and Business Date

A new trade date starts at 4:00pm CST and finishes at 4:00pm CST of the following day. All 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 movements 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).

## 3 Clearing Methods

### 3.1 Accounts

This method 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` method.

- **HTTP Request Type:** POST
- **Method:** `/accounts`
- **API security:** This API method requires an authentication token.

#### Inputs

Field	Value	
filter (optional)	Default: <code>"filter": [{ "attr": "account_id", "op": "eq", "value": member_account_id }]</code>	
	<table border="1"> <tr> <td>account_id</td> <td>Account ID</td> </tr> </table>	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",
                           "15fd6v89-425d-296x-8ft6-esrc954j45pe"]
                }],
                "offset":0,
                "limit":10
              })
```

#### Outputs

Field	Value
count	Number of member user accounts found
timestamp	Time of the request
account_id	Account ID



account_number	Account Number
member_users	Member users associated with the account
balances	Balances of the account at the time of the request

Example Response:

```

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

```

## 3.2 Balances

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

- **HTTP Request Type:** POST
- **Method:** /balances
- **API security:** This API method requires an authentication token.

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 trade 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
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:

```

"result": {
  "account_id": "27ff6d34-523d-476d-9ad5-edeb373b83dc",
  "balances": [
    {
      "asset_type": "USD",
      "opening_balance": "247.0",
      "asset_movement": "150.0",
      "spot_movement": "-8000.0",
      "closing_balance": "547.0",
      "change_in_balance": "-300.0",
      "exchange_fees": "1.0",
      "clearing_fees": "1.0",
      "other_fees": "1.0",
      "closing_price": "1.0",
      "closing_price_date": "2018-01-01",
      "usd_value": "547.0"
    },
    {
      "asset_type": "TBTC",
      "opening_balance": "47.0",
      "asset_movement": "50.0",

```

```

    "spot_movement": "1.0",
    "closing_balance": "347.0",
    "change_in_balance": "-300.0",
    "exchange_fees": "1.0",
    "clearing_fees": "1.0",
    "other_fees": "1.0",
    "closing_price": "3987.34",
    "closing_price_date": "2018-01-01",
    "usd_value": "1383606.98"
  }
]
},
"timestamp": "2018-01-01T06:00:00.000Z",
"report_date": "2018-01-01"
}

```

### 3.3 Movements

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

- **HTTP Request Type:** POST
- **Method:** /movements
- **API security:** This API method requires an authentication token.

#### Inputs

Field	Value						
filters (optional)	Default: <code>"filter": [{ "attr": "account_id", "op": "eq", "value": member_account_id }]</code>						
	<table border="1"> <tr> <td>account_id</td> <td>Account ID</td> </tr> <tr> <td>time</td> <td>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)</td> </tr> <tr> <td>asset_type</td> <td>Asset</td> </tr> </table>	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
	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						
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: <code>"sort": [{ "attr": "time", "value": "desc"}]</code>						

#### 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
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: <ul style="list-style-type: none"> <li>• "amount": General movement amount.</li> <li>• "bank_fee": Bank Fees Charged</li> <li>• "clearing_fee": Clearing House Fees Charged</li> <li>• "exchange_fee": Trading Fees Charged</li> <li>• "other_fees": Other Fees Charged</li> </ul>

## 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"
}
],
}
]
}

```

### 3.4 Trades

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

- **HTTP Request Type:** POST
- **Method:** /trades
- **API security:** This API method requires an authentication token.

#### 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
fee_type	Asset of fees

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" },
                     { "attr": "time", "op": "gt", "value":
"2018-01-01T06:00:00.000Z" },
                     { "attr": "trade_id", "op": "eq", "value": "A2019196081HP00" },
                     { "attr": "side", "op": "ne", "value": "BUY" },
                     { "attr": "aggressor", "op": "eq", "value": "Y" },
                     { "attr": "qty", "op": "gt", "value": "1.0" },
                     { "attr": "px", "op": "gt", "value": "6994.0" },
                     { "attr": "qty_type", "op": "eq", "value": "TBTC" },
                     { "attr": "px_type", "op": "ne", "value": "USD" },
                     { "attr": "fee_type", "op": "eq", "value": "USD" },
                     { "attr": "report_date", "op": "eq", "value": "2018-01-01" }
                 ],
                 "sort": [
                     { "attr": "time", "value": "asc" }
                 ],
                 "offset": 0,
                 "limit": 10
             })
```

### 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
qty_type	Base currency
px_type	Quote currency
fee_type	Fee currency
report_date	Business date of the trade

Example Response:

```

"result": {
  "count": 1,
  "trades": [
    {
      "trade_id": "A2019196081HP00",
      "tcr_id": "484548071",
      "client_order_id": "NRL17081620031",
      "fix_id": "PRTCE8HX6UY",
      "time": "2018-01-01T06:00:00.000Z",
      "description": "BUY 1.0 TBTC @ 6994.0 USD",
      "side": "BUY",
      "account_id": "27ff6d34-523d-476d-9ad5-edeb373b83dc",
      "aggressor": "Y",
      "qty": "1.0",
      "px": "6994.0",
      "clearing_fee": "140.8918",
      "exchange_fee": "139.8918",
      "qty_type": "TBTC",
      "px_type": "USD",
      "fee_type": "USD",
      "report_date": "2018-01-01"
    }
  ]
}

```

### 3.5 Asset Movement Status Requests

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

- **HTTP Request Type:** POST
- **Method:** /requests
- **API security:** This API method requires an authentication token.

Inputs

Field	Value
filters (optional)	Default: <code>"filter": [{ "attr": "account_id", "op": "eq", "value":</code>

	member_account_id }]										
	<table border="1"> <tr> <td>account_id</td> <td>Account ID</td> </tr> <tr> <td>time</td> <td>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)</td> </tr> <tr> <td>asset_type</td> <td>Asset type (BTC, BCH, ETH, LTC)</td> </tr> <tr> <td>amount</td> <td>Amount of the request</td> </tr> <tr> <td>transaction_type</td> <td>Request type (withdrawal, deposit)</td> </tr> </table>	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)
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"},
      { "attr": "time", "op": "gt", "value": "2018-01-01T06:00:00.000Z" },
      { "attr": "asset_type", "op": "eq", "value": "TBTC" },
      { "attr": "amount", "op": "gt", "value": 1 },
      { "attr": "transaction_type", "op": "eq", "value": "withdrawal" }
    ],
    "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
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": "2MvQsanz92K5DmKe9fd2GHPz4oRKJXoAR1m4",
      "time": "2018-01-01T06:00:10.000Z",
      "asset_type": "TBTC",
      "amount": "-0.0001",
      "fee": "-0.0000001",
      "fee_type": "TBTC",
      "transaction_type": "withdrawal",
      "state": "pending"
    }
  ]
}
```

## 4 Change History

Date	Message(s) or Section	Description
20190731		Version 1
20190809	Filters	In python, filters should be specified in the <b>json</b> argument of requests.post function, not in the <b>data</b> argument. Some new filters have been added to the different methods. 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)