

# Eze EMS xAPI Getting Started Guide

This document contains information about getting started with Eze EMS xAPI.



**Confidentiality Notice:** The information included in this document is confidential information of SS&C Eze and is intended only for SS&C Eze and its affiliates, Eze EMS clients, and their respective employees.

## Table of Contents

|  |    |
|--|----|
| <b>What's New</b>                                | 4  |
| <b>Introduction</b>                              | 5  |
| Eze EMS xAPI Basics                              | 5  |
| Eze EMS xAPI Use Restrictions                    | 5  |
| Eze EMS xAPI Version                             | 6  |
| Download EMS xAPI                                | 6  |
| Developer Support                                | 6  |
| <b>Business Use Cases</b>                        | 7  |
| <b>About gRPC</b>                                | 8  |
| Overview   | 8  |
| <b>Getting Started Using gRPC UI</b>             | 9  |
| <b>Getting Started Using REST API</b>            | 11 |
| REST API on Swagger                              | 12 |
| REST API on Postman                              | 14 |
| Managed Cloud Platform (MCP)                     | 16 |
| <b>Getting Started Using Python Code Samples</b> | 17 |
| Prerequisites                                    | 17 |
| Initial Setup                                    | 17 |
| Connecting/Disconnecting                         | 18 |

|  |           |
|--|-----------|
| Standard Login (non-SRP) .....                       | 18        |
| Logging via SRP Method .....                         | 19        |
| Placing an Order .....                               | 20        |
| Getting Order Details .....                          | 23        |
| Getting Execution Details .....                      | 24        |
| Cancelling an Order .....                            | 24        |
| Subscribing to Market Data .....                     | 24        |
| <b>Eze EMS xAPI Frequently Asked Questions .....</b> | <b>27</b> |
| <b>Appendix A .....</b>                              | <b>29</b> |

## Legal Information

### Copyright

This document is the copyrighted work of Eze Castle Software LLC (“SS&C Eze”). SS&C Eze distributes this document pursuant to a subscription agreement containing confidentiality and license provisions and is solely for the benefit of its authorized licensees. This documentation may not be copied or transmitted, in whole or in part, in any form or by any means without the express written consent of SS&C Eze.

© 1997 to 2025 Eze Castle Software LLC. All Rights Reserved.

### Content

Information in this document is subject to change without notice. In the event that you are using a version of SS&C Eze products other than the most recent version, there may be discrepancies between the content of this documentation and the operation or visual presentation of your older version of the product. SS&C Eze does not warrant that this documentation is error free.

### Trademarks

SS&C Eze is a trademark of SS&C Technologies, Inc. All SS&C Eze company and product names are trademarks or registered trademarks of SS&C Technologies, Inc. or SS&C Eze.

All other company or product names mentioned herein are the trademarks or registered trademarks of their respective companies.

SS&C Eze

<http://www.ezesoft.com/>

## What's New

This release includes the following updates:

| Version No.    | Date         | Summary of Update  |
|----------------|--------------|--|
| v2025.8.0.3104 | Dec 24, 2025 | There are no documentation updates in this EMS xAPI release. |



**Note:** Refer to the [Revision History](#) section for detailed information on past release versions.

# Introduction

The purpose of this document is to help clients get started with the EMS xAPI application. This document provides a step-by-step process of using these APIs.

Eze EMS xAPI is robust and easy-to-use application that allows programmers and trading businesses to complete various trading workflows, and also access key information, including:

- Automating order routing - to smart order routers, algorithms and other trading systems.
- Routing orders to multiple brokers, dark pools, ATS, and MTFs via the Eze EMS Global Routing Network - across asset classes.
- Staging or routing single or pairs orders.
- Accessing balances, positions, executions, and other order details.
- Accessing comprehensive list and basket capabilities.

Although EMS xAPI can operate with all gRPC compatible languages, only Python language information is provided in this document as an example. Refer to this [link](#) for more information on gRPC.

## Eze EMS xAPI Basics

The Eze EMS xAPI operates in conjunction with your existing Eze EMS account permissioning and entitlements. **The Eze EMS xAPI is not a standalone data feed application that is provided to you independent of the Eze EMS.** Please contact Eze Client Service if you need to request or make changes to appropriate permissions for your account.

## Eze EMS xAPI Use Restrictions

As an Eze EMS xAPI user, you are prohibited from retransmitting any Eze Market Data using the Eze EMS xAPI, without the express prior written consent of Eze EMS and the exchanges or other third-party data providers (referred to as **“Sources”** in your end user agreement). Any unauthorized retransmission of Eze Market Data is a breach of your end user agreement and will cause immediate termination of your use of the Eze EMS, Eze Market Data, and the Eze EMS xAPI.

Any non-display usage of Eze Market Data, such as use of real-time data in algorithmic trading or program trading, is subject to the rules, regulations, and policies of the applicable exchanges and additional exchange fees may apply. In addition, you may have a non-display usage of Eze Market Data even if a display of real-time data occurs. Please review your Eze EMS end user agreement, and the exchanges' and third-party data providers' rules, regulations, and policies that apply to your use of the Eze EMS API (which apply to Eze EMS xAPI) and/or Eze Market Data. It is the sole responsibility of the Eze EMS xAPI user and each user receiving, directly or indirectly accessing or otherwise using Eze Market Data to determine whether your receipt, access or use is reportable and/or fee liable.

## Eze EMS xAPI Version

This document covers all the APIs and updates to the Eze EMS xAPI that are part of 2025.8.0.3104 release.

### Download EMS xAPI

You can begin using Eze EMS xAPI by downloading the necessary files from any of the following sources:

- [GIT Repository](#).
- [Eze EMS Knowledgebase](#).
- [EMS xAPI Portal](#) (Client access only. Login required).

### Developer Support

- If you are an existing Eze EMS User, [log in](#) to access developer support documentation and sample code.
- You can [contact us](#) or [request a demo](#) if you want to explore more about EMS xAPI.
- You can send us an e-mail [apisupport@ezesoft.com](mailto:apisupport@ezesoft.com) or call +1 312-442-8122.

## Business Use Cases

Mentioned below are some of the business use cases, where Eze EMS xAPI adds immense value to buy-side businesses:

- **Trade Execution API:** With the help of Eze EMS xAPI, you can add more flexibility and customization to your complex trading workflows. The broker-neutral trading APIs can seamlessly plug into your sophisticated trading strategies and enable you to achieve best execution and value for your strategies from the markets.
- **Market Data API:** Eze EMS xAPI has a range of real-time and historical time-series and Market Data. Subject to the terms of your Eze EMS end user agreement and prior written consent of Eze and applicable exchanges and third-party information providers, a buy-side business can consume the data from the Market Data API into proprietary workflows to enhance predictive models and automated strategies to trade equities, futures, options etc.

Below are some of the use cases supported on Eze EMS xAPI:

- CRUD operations for Single and Pair Orders.
- Order Utility functions such as Checking Positions and Balances.
- Security and Symbol Guides.

# About gRPC

## Overview

gRPC is a modern open source high performance Remote Procedure Call (RPC) framework that can run in any environment. It can efficiently connect services in and across data centers with pluggable support for load balancing, tracing, health checking and authentication. It can use protocol buffers as both its Interface Definition Language (IDL) and as its underlying message interchange format.

For more information on gRPC, refer to this [link](#).

For information about the languages supported on gRPC, refer to this [link](#).

# Getting Started Using gRPC UI

Eze EMS xAPI users can launch a web-based interface (gRPC UI) allowing you to more easily review EMS xAPI's services and methods and use request/response parameters directly through your web browser.



**Note:** The gRPC Web UI does not support streaming APIs (e.g., `SubscribeOrderInfo`, `SubscribeLevel1Ticks`).

Follow the steps below to start using the gRPC Web UI:

1. Visit the <https://uatwebxapi.realtick.com/> url. The Swagger setup with all the Eze EMS xAPI (gRPC) is displayed, shown below.

The screenshot shows the SS&C Eze EMS xAPI gRPC UI. At the top, there are dropdown menus for 'Service name' (set to 'UtilityServices') and 'Method name' (set to 'Connect'). Below these are tabs for 'Request Form', 'Raw Request', 'Response', and 'History'. The 'Request Form' tab is active. Under 'Request Metadata', there is a table with one row and two columns: 'Name' and 'Value'. There is a red 'X' icon next to the first column and a green '+' icon next to the second column, with the text 'Add item' below it. Under 'Request Data', there is a section titled 'ConnectRequest' with four fields: 'UserName' (string), 'Domain' (string), 'Password' (string), and 'Locale' (string). Each field has a checkbox and a text input field. Below this is a 'Request Timeout' section with a text input field containing '10' and the word 'seconds' next to it. At the bottom is a large 'Invoke' button.

2. Navigate to **UtilityServices** section and click **connect** API. The **Request Form** pane opens.

3. To enter the details, enable the parameter checkbox.
4. Enter your **UserName**, **Domain**, **Password**, and **Locale** details.



**Note:** Contact your SS&C Eze client service representative if you need any assistance.



**Note:** Your account will be temporarily locked for 3 minutes if you attempt to log in three (3) times within 60 seconds, regardless of whether the attempts are successful or unsuccessful.

5. Click **Invoke**.
6. The status is displayed in the **Response** pane. If your login is successful, a **UserToken** is generated.



**Note:** The generated **UserToken** is needed for all the subsequent API calls.



**Note:** Contact your SS&C Eze client service representative for assistance on the gRPC Web UI.

## Getting Started Using REST API

To streamline API development and testing, you can now leverage **Swagger** and **Postman** to interact with EMS REST APIs.

- [\*\*Swagger UI\*\*](#) provides an interactive interface directly from our API documentation, allowing you to explore endpoints, send requests, and view responses—all within a browser.
- [\*\*Postman\*\*](#) offers a powerful desktop and web-based environment for building, testing, and automating API workflows. It supports advanced features like environment variables, scripting, and collection management.

## REST API on Swagger

The EMS xAPI Swagger UI is ideal for quick testing and understanding API behavior without needing external tools. It also supports authentication methods like Bearer Tokens, making it easy to simulate authorized requests.

Follow the steps below to start using the EMS REST API on Swagger:

1. Visit the <https://emsuatxapi.taltrade.com:9001/index.html> url. The Swagger setup with all the Eze EMS xAPI (REST) is displayed.
2. Navigate to the **Authentication APIs** section and click on **connect** API. The **Parameters** pane opens.
3. Click **Try it out** to enable the fields in the **Parameters** pane.

The screenshot shows the Swagger UI interface. At the top, there's a header with the title 'UtilityServices'. Below it, a list of API endpoints under the 'GET' method: '/api/v1/utility/todays-balances', '/api/v1/utility/todays-activity', '/api/v1/utility/todays-activity-book', '/api/v1/utility/todays-breakdown-positions', '/api/v1/utility/todays-net-positions', and '/api/v1/authentication/connect'. The last endpoint, '/api/v1/authentication/connect', is highlighted with a blue border. Below the list is a 'Parameters' pane with a table. The table has two columns: 'Name' and 'Description'. It contains four rows: 'UserName' (string, query), 'Domain' (string, query), 'Password' (string, query), and 'Locale' (string, query). Each row has an input field next to it. To the right of the table is a 'Try it out' button.

4. Enter your **UserName**, **Domain**, **Password**, and **Locale** details.



**Note:** Contact your SS&C Eze client service representative if you need any assistance.



**Note:** Your account will be temporarily locked for 3 minutes if you attempt to log in three (3) times within 60 seconds, regardless of whether the attempts are successful or unsuccessful.

5. Click **Execute**.

6. The status is displayed in the **Responses** pane. If your login is successful, a **UserToken** is generated. You can use this **UserToken** to explore the REST APIs.



**Note:** The generated **UserToken** is needed for all the subsequent API calls.

7. You can also set the user token for your session by including it as a bearer token. Follow these steps:
  - a. Click **Authorize** button.
  - b. Enter the user token in the **Value** field.
  - c. Click **Authorize**.

Once set, you do not need to enter the user token for each API call.



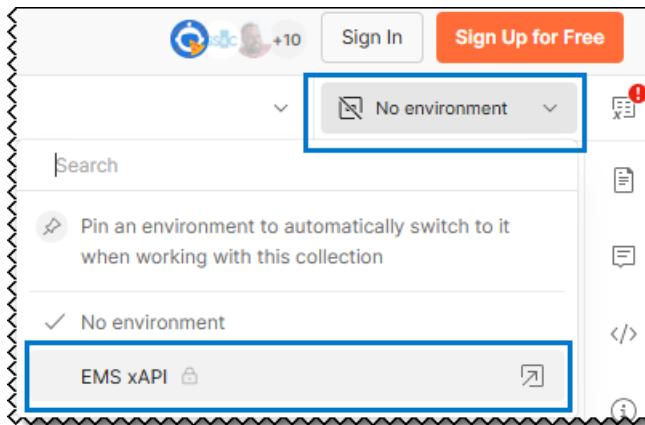
**Note:** Swagger does not support the Subscribe APIs (e.g., `SubscribeOrderInfoRequest`).

## REST API on Postman

The EMS xAPI Postman collections supports various authentication schemes, including Bearer Tokens, making it easier to test secured endpoints and manage session-level credentials like UserToken.

Follow the steps below to start using the EMS REST API on Postman:

1. Visit the [SSNC-Eze-EMS-xAPI](#) path. The pre-configured Postman setup for EMS xAPI (REST) is launched.
2. Select the **EMS xAPI** environment from the top-right dropdown list, shown below.



3. Navigate to the Authentication API using the left index, **authentication > connect API > GET**.

A screenshot of the Postman application interface. On the left, the sidebar shows a collection named "EMS xAPI REST" under "SS&amp;C Eze". The "authentication" folder is expanded, and the "connect" folder is selected, with its "GET The ConnectRequest obj..." endpoint highlighted with a blue box. In the main panel, a request is being built for this endpoint. The method is set to "GET", and the URL is "({baseUrl})/api/v1/authentication/connect?UserName=...". The "Params" tab is selected, showing five query parameters: "Key" (checked), "UserName" (checked), "Domain" (checked), "Password" (checked), and "Locale" (checked). Each parameter has a description: "A valid user...", "A valid domain...", "A valid password...", and "The region ...". There are also tabs for "Auth", "Headers", "Body", "Scripts", "Settings", and "Cookies".

4. Click **Params** tab.
  - a. Enter your **UserName**, **Domain**, **Password**, and **Locale** details. Contact your SS&C Eze client service representative if you need any assistance.



**Note:** Your account will be temporarily locked for 3 minutes if you attempt to log in three (3) times within 60 seconds, regardless of whether the attempts are successful or unsuccessful.

5. Click **Send**. The status is displayed in the **Responses** pane.

If your login is successful, a **UserToken** is generated. You will need this token for subsequent API calls.

6. **Bearer Token:** You can also set the user token for your session by including it as a bearer token. Follow these steps:
  - a. Select an API (e.g., **connect**).
  - b. Click the **Authorization** tab.
  - c. From the **Auth Type** dropdown list, select **Bearer Token**.
  - d. In the **Token** field, enter the generated user token, as shown below.

Once set, you do not need to enter the user token for each API call.

The screenshot shows the Swagger UI interface for an 'EMS xAPI' endpoint. The URL is `GET {{baseUrl}}/api/v1/authentication/connect?UserName=`. The 'Authorization' tab is selected. In the 'Auth Type' dropdown, 'Bearer Token' is chosen. To the right, there is a 'Token' input field containing a user token. A tooltip explains that the authorization header will be automatically generated when the request is sent. Other tabs visible include 'Overview', 'Params', 'Headers (8)', 'Body', 'Scripts', 'Settings', and 'Cookies'.



**Note:** Swagger does not support the Subscribe APIs (e.g., `SubscribeOrderInfoRequest`).

## Managed Cloud Platform (MCP)

You can use Postman with the Managed Cloud Platform (MCP) to generate a MCP server based on your selected API(s).

**To generate a MCP server:**

1. Navigate to the online MCP Generator: <https://www.postman.com/explore/mcp-generator>.
2. Click the **Search for APIs (required)** field at the center of the screen. The search window appears.



**Note:** You need to be logged in to use the search feature.

3. Enter **EMS xAPI REST** and select **SS&C Eze EMS xAPI** environment from the search results, shown below.

The screenshot shows the Postman search interface. In the top search bar, the text 'EMS xAPI' is typed. Below the search bar, there is a 'Clear' button. The search results show a workspace titled 'EMS xAPI REST'. The workspace description states: 'This workspace contains resources that you can demo and share with customers and partners.' It includes a thumbnail icon with a blue and white design, a 'SS&C Eze EMS xAPI' link, and a 'By SS&C Eze' link.

4. In the left index, select the desired APIs. Use the right pane to enable the checkboxes for those APIs.
5. Click **Add Requests** to add the selected APIs to the generator list.
6. Click **Generate** to create the downloadable MCP ZIP file.
7. Download the generated **ZIP** file.
  - a. Unzip the file.
  - b. Open the **README.md** file in the unzipped folder.
  - c. Follow the instructions in the **README.md** for the next steps.

# Getting Started Using Python Code Samples

## Prerequisites

The Python 3.9 is installed. If it is not installed already, download and install Python (<https://www.python.org/downloads/>).

## Initial Setup

Follow the steps below for initial setup:

1. Download and install Python (<https://www.python.org/downloads/>).
2. Run the following commands in command prompt to install **gRPC / gRPC tools** and **SRP**.

```
pip install grpcio==1.75.0
pip install grpcio-tools==1.75.0
pip install srp
pip install pycryptodomex
pip install protobuf==6.32.1
```



**Note:** For more information on gRPC refer to this [link](#).

3. Download the proto files from Git repository  
<https://github.com/ezesoft/xapi/tree/master/protos>.
4. Run the following commands in command prompt to generate the **order\_pb2.py**, **order\_pb2\_grpc.py**, **utilities\_pb2.py**, **utilities\_pb2\_grpc.py**, **market\_data\_pb2.py** and **market\_data\_pb2\_grpc.py** files.

```
python -m grpc_tools.protoc -I../protos --python_out=. --grpc_python_out=.
../protos/order.proto

python -m grpc_tools.protoc -I../protos --python_out=. --grpc_python_out=.
../protos/utilities.proto

python -m grpc_tools.protoc -I../protos --python_out=. --grpc_python_out=.
../protos/market_data.proto
```



**Note:** Replace **..../protos** with your local machine directory path containing **.proto** files.  
There are two (2) instances of it in each of the command above.

5. Download the **roots.pem** file from [gRPC GitHub](#) repository. Make sure you save this file in the same folder along with **\_pb2.py** files generated in the previous step.

The following sections provide information on Establishing Server Connection, Logging into SRP, Submitting an Order, Subscribing to Order Info, and Requesting and Watching Market Data.

 **Note:** The following examples assume you are comfortable working with Python and Eze EMS data via the command line. If you prefer to work with a GUI interface, we have provided instructions and a sample application as part of the [Eze EMS xAPI Python Sample Application Guide](#).

## Connecting/Disconnecting

### Standard Login (non-SRP)

In order to establish a secured connection with server, you must have the server's host IP address, port number and a local roots.pem (root certificates) file. If you need any information on these details contact your SS&C Eze client service representative.

EMS xAPI supports Secure Remote Password ([SRP](#)) protocol, refer the [Logging via SRP Method](#) for more details.

Follow the steps below to connect/disconnect EMS xAPI:

1. Import the utility modules generated during [step 4 of Initial Setup](#) above, to access the EMS xAPI Utility interface. You can use the below code to run the commands.

```
import grpc
import utilities_pb2 as util
import utilities_pb2_grpc as util_grpc
```

2. Use the below code to create a stub to gain access to the remote Utility interfaces. EMS xAPI requires Transport Layer Security (TLS), so you must establish an encrypted connection. You can use this [roots.pem](#) file or it is also included in the EMS xAPI repository provided by the SS&C Eze client service representative.

```
with open(r'.\roots.pem', 'rb') as f: cert = f.read()
channel = grpc.secure_channel('SERVER:PORT', grpc.ssl_channel_credentials(root_
certificates=cert))
util_stub = util_grpc.UtilityServicesStub(channel)
```



**Note:** The server details and authentication credentials are provided by SS&C Eze client service representative.

3. Use the stub to connect and authenticate with EMS xAPI.

```
connect_response = util_stub.Connect(util.ConnectRequest(UserName='USER',
Domain='DOMAIN', Password='PASSWORD', Locale='LOCALE'))

print('Connect result: ', connect_response.Response)
```



**Note:** On successful login you receive a unique glx2 UserToken in the response. Do not lose this token as it must be provided in all subsequent calls to the server.

4. You can now verify the login succeeded and disconnect from the server.

```
if connect_response.Response == 'success':

    disconnect_response = util_stub.Disconnect(util.DisconnectRequest
(UserToken=connect_response.UserToken))

    print('Disconnect result: ', disconnect_response.ServerResponse)
```

You have now connected, authenticated, and disconnected from the server using EMS xAPI.

You can also use the code snippet from the [Git repository](#) to establish a secured server connection. You need to provide information for the Field names mentioned below.

| Field Name | Required? | Data Type | Accepted Values/Examples  |
|------------|-----------|-----------|---|
| UserName   | Yes       | string    | A valid user name   |
| Domain     | Yes       | string    | A valid domain name   |
| Password   | Yes       | string    | A valid password  |
| Locale     | Yes       | string    | The region the user wants to connect. Example:<br>AMERICAS, ASIA etc. |



**Note:** Your account will be temporarily locked for 3 minutes if you attempt to log in three (3) times within 60 seconds, regardless of whether the attempts are successful or unsuccessful.

## Logging via SRP Method

Secured Remote Password (SRP) is a mechanism that allows user to get authenticated without passing the password to the server. By implementing the SRP support in EMS xAPI server, you are able to send hash coded password during login. For more information on SRP refer to this [link](#).



**Note:** To login using SRP method your domain has to be SRP enabled. Contact your SS&C Eze client service representative for assistance.



**Note:** Enter a valid USER and DOMAIN details in the code snippet to login. Contact your SS&C Eze client service representative for more information.

The SRP login method uses **StartLoginSrp** and **CompleteLoginSrp** APIs to login. Use the code snippet from the [Git repository](#) to login to EMS xAPI server via SRP method. You need to provide information for the Field names mentioned below.

| Field Name    | Required? | Data Type | Accepted Values/Examples   |
|---------------|-----------|-----------|--|
| Identity      | Yes       | string    | A valid user identity (combination of user-name@domain)            |
| srpTransactId | Yes       | string    | Unique transaction ID per user                                     |
| strEphA       | Yes       | string    | This field is specific to SRP6 Protocol                            |
| strMc         | Yes       | string    | This field is specific to SRP6 Protocol                            |
| UserName      | Yes       | string    | A valid user name  |
| Domain        | Yes       | string    | A valid domain name  |
| Locale        | Yes       | string    | The region the user wants to connect. Example: AMERICAS, ASIA etc. |



**Note:** On successful login you receive a unique glx2 UserToken in the response. Do not lose this token as it must be provided in all subsequent calls to the server.



**Note:** Your account will be temporarily locked for 3 minutes if you attempt to log in three (3) times within 60 seconds, regardless of whether the attempts are successful or unsuccessful.

## Placing an Order

Follow the steps below to place an order:

1. Import the order modules generated during [step 4 of Initial Setup](#) above, to access the EMS xAPI Order interfaces.

```
import order_pb2 as ord
import order_pb2_grpc as ord_grpc
```

2. Create a stub to gain access to the remote Order interfaces. Since you are reusing the channel object; there is no need to recreate it.

```
ord_stub = ord_grpc.SubmitOrderServiceStub(channel)
```

3. Use the below code to submit the order request.

```
order_response = ord_stub.SubmitSingleOrder(ord.SubmitSingleOrderRequest
(Symbol='TSLA', Side='BUY', Quantity=5000, Route='ROUTE', Account='ACCOUNT',
OrderTag='MyOrderId', UserToken=connect_response.UserToken))

print('Order result: ', order_response)
```



**Note:** The route and account information are provided by SS&C Eze client service representative.



**Note:** When submitting an order, you can optionally populate the **OrderTag** property with a unique identifier so you can match an order event from EMS xAPI with an order in your system.

Alternatively, you can use the code snippet from the [Git repository](#) for submitting a single order using EMS xAPI server. Refer the table below and enter valid details in the code snippet to submit an order.

| Field Name | Required? | Data Type | Description   |
|------------|-----------|-----------|---|
| Symbol     | Yes       | string    | Valid ticker symbol. Example: AAPL, IBM or VOD.LSE etc. |

| Field Name   | Required? | Data Type | Description  |
|--------------|-----------|-----------|--|
| Side         | Yes       | string    | BUY, SELL, SELLSHORT NOTE: To send an order with side SELLSHORT, the extended field <b>SHORT_LOCATE_ID</b> must be assigned. The <b>SHORT_LOCATE_ID</b> is an ID assigned to short sell orders. Similarly, to send a <b>Buy To Cover</b> order, set the side to <b>BUY</b> and assign the extended field <b>TO_OPEN_POS</b> to the required volume |
| Quantity     | Yes       | int32     | Value > 0  |
| Route        | Yes       | string    | Route name as shown in Eze EMS. NOTE: This field is also referred to as <b>Exit Vehicle</b>  |
| Account      | Yes       | string    | Semi colon separated values that represent either Trade or Neutral accounts the user has permission to. Example: TAL;TEST;USER1;TRADE or TAL;TEST;USER2;NEUTRAL  |
| OrderTag     | No        | string    | Order Tag  |
| TicketId     | No        | string    | Ticket ID  |
| UserToken    | Yes       | string    | A server generated GUID that is given as response to the client during the first login   |
| Staged       | No        | bool      | TRUE or FALSE (Note: in order to send a staged order, this field becomes mandatory and has to be set as TRUE only)   |
| ClaimRequire | No        | bool      | TRUE or FALSE (Note: setting TRUE value envisages a user running Eze EMS who then claims the Order so it can switch from Pending to Live State)  |

| Field Name     | Required? | Data Type                   | Description  |
|----------------|-----------|-----------------------------|--|
| GoodFrom       | No        | google.protobuf.Timestamp   | Time at which order is first valid for execution   |
| TimeInForce    | No        | ExpirationType              | Time or date at which order is no longer valid. In case no value is provided, DAY expiration type is set by default. Refer SubmitSingleOrder API in Eze EMS Technical Reference Document |
| PriceType      | No        | PriceTypeEnum               | By default the price type is set to <b>Market</b> , in case no value is provided. Refer SubmitSingleOrder API in Eze EMS Technical Reference Document                                    |
| Price          | No        | google.protobuf.DoubleValue | Limit price submitted in order   |
| StopPrice      | No        | google.protobuf.DoubleValue | Stop Price   |
| UserMessage    | No        | string                      | User Message/Notes   |
| ExpirationDate | No        | google.protobuf.Timestamp   | Date at which order is no longer valid   |

## Getting Order Details

You can request order activity from EMS xAPI to see the status and details of an order. EMS xAPI offers you both unary (request-response) and streaming interface to retrieve order activity. Order activity includes orders you submitted to the system as well as activity on the order, such as executions (fills) or rejections.

Refer the code snippet for GetTodaysActivityJson interface provided in the [Git repository](#) for unary (request-response) type order activity. For streaming interface refer SubscribeOrderInfo code snippet provided in the [Git repository](#).

You can add optional filters to refine the data results by adding a single filter IncludeUserSubmitOrder to limit the request to order submission activity.

```
activity_response = util_stub.GetTodaysActivityJson(util.TodaysActivityJsonRequest
(IncludeUserSubmitOrder=True, UserToken=connect_response.UserToken))
```



**Note:** By default, all included filters are set as False.

In the [Placing an Order](#) section, an OrderTag was provided on the request. You can use the OrderTag to find a specific order and lookup the EMS xAPI Order ID.

```
xapi_order_id = df[(df['OrderTag']=='MyOrderId')]['OrderId'][0]
print('The xAPI OrderId for my order is ', xapi_order_id)
```



**Note:** You can use the EMS xAPI Order ID to cancel and modify existing orders.

## Getting Execution Details

You can use the same [Getting Order Details](#) interface to retrieve order execution details from EMS xAPI by changing the filter.

You can request fill activity from EMS xAPI by adding a single filter IncludeExchangeTradeOrder.

```
activity_response = util_stub.GetTodaysActivityJson(util.TodaysActivityJsonRequest
(IncludeExchangeTradeOrder=True, UserToken=connect_response.UserToken))
```

Refer the code snippet provided in the [Git repository](#) to get the execution details.

## Cancelling an Order

In the [Getting Order Details](#) section, you retrieved the EMS xAPI Order ID of an order placed in the [Placing an Order](#) section. In this section, you will use the Order ID to request the order be cancelled.

```
cancel_response = ord_stub.CancelSingleOrder(ord.CancelSingleOrderRequest(OrderId=xapi_order_id, UserToken=connect_response.UserToken))
print('Cancel result: ', cancel_response)
```

Refer the code snippet provided in the [Git repository](#) to cancel an order.

## Subscribing to Market Data

Using EMS xAPI you can request a market data snapshot as well as subscribe to future market data updates. This section covers both the snapshot and streaming cases for Level 1 market data.

**Follow the steps below to subscribe to market data:**

1. To access the EMS xAPI Market Data interfaces, import the market\_data modules generated during the [step 4 of Initial Setup](#). Since gRPC streaming is a blocking operation, data coming from the server should be processed on a separate thread so your application is not blocked. To do this, include the threading module as well.

```
from threading import Thread
import market_data_pb2 as md
import market_data_pb2_grpc as md_grpc
```

2. Use the following code to define a new function to process market data returned by the server:

```
def handle_data(response):
    try:
        for tick in response:
            if tick.Trdprc1.DecimalValue == 0.0:
                continue
            print('Received market data for {0}, Last: {1}'.format(tick.DispName,
            tick.Trdprc1.DecimalValue))
    except Exception as e:
        print(e)
```

3. Create a stub to gain access to the remote Market Data interfaces:

```
md_stub = md_grpc.MarketDataServiceStub(channel)
```

4. Request market data from the server. You can request data for one or more securities at a time:

```
response = md_stub.SubscribeLevel1Ticks(md.Level1MarketDataRequest(Symbols=
['TSLA'], Request=True, Advise=True, UserToken=connect_response.UserToken))
```



**Note:** If you want a snapshot of the market data (i.e., the current values), set the Request parameter to True. If you want to receive all subsequent market ticks (i.e., get future market data updates), set the Advise parameter to True.

5. Create and start the data handler thread and pass the iterable response object to the thread function:

```
t = Thread(target=handle_data, args=(response, ))
t.start()
```

6. When you need to shut down, cancel the request and join the thread to wait for it to terminate gracefully:

```
response.cancel()
t.join()
```

Alternatively, you can use the code snippet from the [Git repository](#) to watch market data using EMS xAPI server. You need to provide information for the Field names mentioned below.

| Field Name          | Required? | Data Type       | Description  |
|---------------------|-----------|-----------------|--|
| UserToken           | Yes       | string          | A server generated GUID that is given as response to the client during the first login |
| Symbols             | Yes       | repeated string | Valid ticker symbol. Example: AAPL, IBM or VOD.LSE etc.                                |
| RegionalExchangeIds | No        | repeated string | Regional exchange ID   |
| Request             | Yes       | bool            | If set to <b>True</b> , a current snapshot of the data will be retrieved               |
| Advise              | Yes       | bool            | If set to <b>True</b> , real-time updates from the server will be registered for       |

# Eze EMS xAPI Frequently Asked Questions

## 1. How to track an order?

When you create an order using EMS xAPI, you can specify an external identifier in the OrderTag property. EMS xAPI includes the OrderTag in all update messages. This mechanism allows you to match orders between your system and ours.

When you request order activity using either the **Unary** GetActivityJson or the **Streaming** SubscribeOrderInfo API, the OrderTag property will contain the value you specified.

In case you did not specify a value to OrderTag during the order creation, then you will have to match the orders between your system and ours based on properties such as symbol and quantity.

## 2. What is the difference between Unary and Streaming APIs?

GetActivityJson is an example of a **Unary** RPC. You make a request and get a single response. Unary is sometimes referred to as the request-response communication pattern.

SubscribeOrderInfo is an example of a **Streaming** RPC. You make a request and receive data from the server as it is available. Technically, when calling a Streaming API, you are provided an iterator object; each time you accesses the iterator (typically in a loop), you are blocked until data is available from the server.

Since streaming RPC blocks an application, you should loop through the iterator object from a dedicated thread.

A good example of a Streaming RPC is market data: you request live price updates for a security and receive the updates as they arrive from the exchange.

## 3. Can Eze OMS/Eze EMS user trigger automated compliance rule from EMS xAPI?

Yes, as an Eze EMS xAPI user you can trigger the automated compliance rule from EMS xAPI and run automated compliance checks for EMS xAPI originated order. This setting is configured in Eze OMS and is similar to the process in Eze EMS.

The FID used for this is **COMPLIANCE\_USER\_OPTION\_FLAGS (31640)** w/ Flag = **NO\_TOUCH\_AUTO\_COMPLIANCE (1)**. Additionally the extended fields that are to be included with the order are **EZE\_OMS\_MANAGER;EZE\_OMS\_TRADER;CUSTODIAN**

## 4. How to login using SRP?

EMS xAPI supports both SRP and non-SRP login methods. To login using SRP method your domain has to be SRP enabled. If your domain is SRP enabled, both the SRP and standard login methods work. If your domain is not SRP enabled, then only the standard login will work.

## 5. How can I know if my EMS xAPI session is active?

You can subscribe to **SubscribeHeartBeat** API, to know your EMS xAPI connection status. On subscribing to this API, you will be intimated and requested to reconnect if an active connection with the server is terminated.

## 6. Why was my account temporarily locked after multiple login attempts?

Your account will be temporarily locked for 3 minutes if you attempt to log in three (3) times within a 60-second period, irrespective of whether those attempts are successful or not. This security measure is in place to prevent unauthorized access and protect your account from potential misuse.

## Appendix A

The following APIs are provided as part of Eze EMS xAPI. For further information refer to the Eze EMS xAPI Technical Reference Document.

| Order APIs                       |                               |  |
|----------------------------------|-------------------------------|--|
| API Name                         | Input                         | Output                                     |
| <b>Order (Single Order APIs)</b> |                               |  |
| CancelSingleOrder                | CancelSingleOrderRequest      | CancelSingleOrderResponse                  |
| ChangeSingleOrder                | ChangeSingleOrderRequest      | ChangeSingleOrderResponse                  |
| SubmitAllocationOrder            | SubmitAllocationOrderRequest  | SubmitAllocationOrderResponse              |
| SubmitSingleOrder                | SubmitSingleOrderRequest      | SubmitSingleOrderResponse                  |
| SubmitTradeReport                | SubmitTradeReportRequest      | SubmitTradeReportResponse                  |
| <b>Order (Pair Order APIs)</b>   |                               |  |
| CancelPairOrder                  | CancelPairOrderRequest        | CancelPairOrderResponse                    |
| ChangePairOrder                  | ChangePairOrderRequest        | ChangePairOrderResponse                    |
| SubmitPairOrder                  | SubmitPairOrderRequest        | SubmitPairOrderResponse                    |
| <b>Order (Basket Order APIs)</b> |                               |  |
| SubmitBasketOrder                | BasketOrderRequest            | BasketOrderResponse                        |
| <b>Order (Miscellaneous)</b>     |                               |  |
| GetOrderDetailByDateRange        | OrderDetailByDateRangeRequest | OrderDetailByOrderTagResponse              |
| GetOrderDetailById               | OrderDetailByIdRequest        | OrderDetailByIdResponse                    |
| GetOrderDetailByOrderTag         | OrderDetailByOrderTagRequest  | OrderDetailByOrderTagResponse              |
| GetUserAccounts                  | UserAccountsRequest           | UserAccountsResponse                       |
| SubmitSeedData                   | SubmitSeedDataRequest         | SubmitSeedDataResponse                     |
| SubscribeOrderInfo               | SubscribeOrderInfoRequest     | stream Sub-<br>scribeOrderInfoResponse     |
| SubscribeOrderInfoJson           | SubscribeOrderInfoJsonRequest | stream Sub-<br>scribeOrderInfoJsonResponse |

| Order APIs                     |                                    |  |
|--------------------------------|------------------------------------|--|
| API Name                       | Input                              | Output                                     |
| GetOrder-DetailByDateRangeJson | Order-DetailByDateRangeJsonRequest | stream Order-DetailByDateRangeJsonResponse |
| GetOrder-DetailByOrderIdJson   | Order-DetailByOrderIdJsonRequest   | Order-DetailByOrderIdJsonResponse          |
| GetOrderDetailByOrderTag       | OrderDetailByOrderTagRequest       | OrderDetailByOrderTagResponse              |
| SubscribeOrderInfoJson         | SubscribeOrderInfoJsonRequest      | stream Sub-<br>scribeOrderInfoJsonResponse |

| MarketData                      |   |  |
|---------------------------------|---|--|
| API Name                        | Input   | Output   |
| AddSymbols                      | AddSymbolsRequest                             | AddSymbolsResponse                             |
| GetDailyWeeklyMonthlyBars       | DailyWeeklyMonthlyBarsRequest                 | DailyWeeklyMonthlyBarsResponse                 |
| GetDescriptionFromOptionSymbol  | DescriptionFromOptionSymbolRequest            | DescriptionFromOptionSymbolResponse            |
| GetIntradayBars                 | IntradayBarsRequest                           | IntradayBarsResponse                           |
| GetLevel1MarketData             | Level1MarketDataRequest                       | Level1MarketDataRecordResponse                 |
| GetOptionChainForUnderlier      | OptionChainRequest                            | OptionChainResponse                            |
| GetOptionsAndGreekData          | OptionsAndGreekDataRequest                    | OptionsAndGreekDataResponse                    |
| GetOptionSymbolFromDescription  | OptionSymbolFromDescriptionRequest            | OptionSymbolFromDescriptionResponse            |
| GetSecurityData                 | SecurityDataRequest                           | SecurityDataResponse                           |
| GetSymbolFromAlternateSymbology | Sym-<br>bolFromAl-<br>ternateSymbologyRequest | Sym-<br>bolFromAl-<br>ternateSymbologyResponse |
| GetSymbolReferenceData          | SymbolReferenceDataRequest                    | SymbolReferenceDataResponse                    |

| MarketData                 |                                |                                 |
|----------------------------|--------------------------------|---------------------------------|
| API Name                   | Input                          | Output                          |
| GetSym-bolsFromCompanyName | Sym-bolsFromCompanyNameRequest | Sym-bolsFromCompanyNameResponse |
| GetTickData                | TickDataRequest                | TickDataResponse                |
| RemoveSymbols              | RemoveSymbolsRequest           | RemoveSymbolsResponse           |
| SubscribeLevel1Ticks       | Level1MarketDataRequest        | stream Level1MarketDataResponse |
| SubscribeLevel2Ticks       | Level2MarketDataRequest        | stream Level2MarketDataResponse |
| UnSubscribeLevel1Data      | UnSubscribeLevel1DataRequest   | UnSubscribeLevel1DataResponse   |
| UnSubscribeLevel2Data      | UnSubscribeLevel2DataRequest   | UnSubscribeLevel2DataResponse   |

| Utilities                    |                                  |                                   |
|------------------------------|----------------------------------|-----------------------------------|
| API Name                     | Input                            | Output                            |
| Connect                      | ConnectRequest                   | ConnectResponse                   |
| Disconnect                   | DisconnectRequest                | DisconnectResponse                |
| GetStrategyList              | StrategyListRequest              | StrategyListResponse              |
| GetTodaysActivity            | TodaysActivityRequest            | TodaysActivityResponse            |
| GetTodaysActivityBook        | TodaysActivityBookRequest        | TodaysActivityBookResponse        |
| GetTodaysActivityJson        | TodaysActivityJsonRequest        | TodaysActivityJsonResponse        |
| GetTodaysBalances            | TodaysBalancesRequest            | TodaysBalancesResponse            |
| GetTodaysBrokenDownPositions | TodaysBrokenDownPositionsRequest | TodaysBrokenDownPositionsResponse |
| GetTodaysNetPositions        | TodaysNetPositionsRequest        | TodaysNetPositionsResponse        |
| GetUserRouteProps            | GetUserRoutePropsRequest         | GetUserRoutePropsResponse         |
| GetUserRoutes                | GetUserRoutesRequest             | GetUserRoutesResponse             |

| Utilities               |                           |                            |
|-------------------------|---------------------------|----------------------------|
| API Name                | Input                     | Output                     |
| SubscribeHeartBeat      | SubscribeHeartBeatRequest | SubscribeHeartBeatResponse |
| Secured Remote Password |                           |                            |
| ChangePasswordSRP       | ChangePasswordSRPRequest  | ChangePasswordSRPResponse  |
| CompleteLoginSrp        | CompleteLoginSrpRequest   | CompleteLoginSrpResponse   |
| StartLoginSrp           | StartLoginSrpRequest      | StartLoginSrpResponse      |



**Note:** Only the mandatory API fields are mentioned here. Contact your SS&C Eze client service representative for a complete list of extended fields.