CAPI LOAD TEST GUIDE

v1.0.0

Jan 22, 2026

Kambala Solutions Private Limited

CAPI LOADT TEST GUIDE v1.0.0

Proprietary Notice

Henceforth, Kambala Solutions Private Ltd. will also be referred as KSPL for all extent and usage in this document.

Document Name	CAPI LOAD TEST GUIDE
Document Code	KSPL_CAPI LOAD TEST GUIDE_001
Document Information	This document provides an overview of the CAPI load testing setup and execution.
Version	v1.0.0
Owner	KSPL IT
Classification	Confidential [External]

Document Version History

Version	Date	Author	Changes	Approved By
v1.0.0	Dec 23, 2025	Jyothi		Rohith

Table of Contents

1 Overview

The objective of this document is to perform load testing on the CAPI order placement flow to validate system performance, stability, and order latency under sustained and peak load conditions. This test ensures that the system can handle high order volumes across multiple investor IDs without errors or performance degradation.

1.1 Scope of Testing

- Order Placement through CAPI
- Buy and Sell Order
- Multiple Investor IDs
- Segment based testing (NSE/BSE/NFO/BFO)
- Performance metrics such as
    - Orders per second
    - Total order processed
    - Latency

1.2 Requirement for LoadTest

- Application Type : CAPI LoadTest Client (C++ executable)
- Execution Mode : Command - line
- Operating System : Linux
- Xml files : Configuration file (config.xml) and Input parameter based XML files.
- Admin User ID and password are required for CAPI login. Ensure that **Access Type : CAPI** is enabled for the Admin User ID.
- Orders can be placed using multiple Investor IDs, you can create those IDs either by:
    A. Using the File Uploader, or
    B. Creating them directly from NorenTrader.
- Make sure for Investor IDs, Account Id and User Id must be same.

2 Load Test Setup

2.1 Executable generation

```
g++ -O3 -std=c++17 -o OrderLatency
reference_code/OrderLatency.cpp
reference_code/cxxapi_client.cpp -L lib/ -lNorenApi_cxxC_o
-pthread -L /home/kambala/LIBS/ -lprotobuf -lcryptopp  -
lcrypto -lcurl -I include/
```

After successful compilation, the load-test executable will be generated.
The executable is executed using a configuration file and runtime parameters.

2.2 Setup Environment Values:

- Set the library path : export LD_LIBRARY_PATH with the directory where the CAPI libraries are stored.

2.2.1. example :

export LD_LIBRARY_PATH=/home/kambala/LIBS:/usr/local/lib64:/usr/local/lib

- Set License file path : export COZY_LICENSE_FILE_PATH with the directory where the CAPI.key file is stored.

2.2.2. example :

export COZY_LICENSE_FILE_PATH=/home/kambala/LIC/

- Enable logger type
  export COZY_LOGGER_LOG_TYPE=1

2.3 Execution Command Format

```
./<executable_path> <config_file_path> <admin_user>
<admin_password> <total_seconds> <order_per_second>
<investor_count> <input_parameter_based_xml_file_path>
```

2.3.1. **sample command :**

```
./OrderLatency config.xml ADMIN Admin@123 600 100 10
NSE_seg.xml >> log1.txt 2>&1 &
```

2.4 Configuration Details

When creating multiple instances, ensure the `<Client_Name>` tag in the `config.xml` file is **unique** for each one.

3 Runtime Parameter Explanation

PARAMATER	DESCRIPTION
ADMIN	Admin UserID
Admin@123	Admin Password
600	Total number of seconds
100	Orders per second(1buy and 1 sell)
10	Number of Investors ID used
NSE_seg.xml / BSE_seg.xml	Testcase values config file

3.1 Test Scenario

- Continuous order placement for configured duration.
  - In the above scenario, Test Duration : 600 seconds
- The order rate is **100 orders per second,** which means **100 buy** orders and **100 sell** orders are placed per second, resulting in a total of **200 orders.**
- Orders are placed at 10 millisecond intervals, with the buy order placed first,

followed by the sell order after a 10 microsecond gap.

3.2 Expected Outcome

- System should sustain configured load without crash.
- No unexpected disconnection or order drops.

4 Conclusion

This load test validates the readiness of the CAPI system to handle high-volume order traffic under real-world trading conditions. Results from this test can be used to identify performance bottlenecks and ensure production readiness.

5 ANNEXURE

5.1 SampleTestCase Configuration Input Parameters

```
<root>

 <OrderRemarks>API LOAD TEST</OrderRemarks>
 <CustomerFirm>C</CustomerFirm>
 <acOrdSrc>CPPAPI</acOrdSrc>

 <Buy>INVB</Buy>
 <Sell>INVS</Sell>

 <Scrip_Count>2</Scrip_Count>

 <TradingSymbol1>WINDMACHIN-EQ</TradingSymbol1>
 <ExchangeSegment1>NSE</ExchangeSegment1>
 <Quantity1>1</Quantity1>
 <Price1>8340</Price1>
 <Product1>M</Product1>
 <OrderType1>LMT</OrderType1>

<TradingSymbol2>MARICO28MAR24C495</TradingSymbol2>
 <ExchangeSegment2>NFO</ExchangeSegment2>
 <Quantity2>1</Quantity2>
 <Price2>2785</Price2>
 <Product2>H</Product2>
 <OrderType2>LMT</OrderType2>

</root>
```

Note: The current implementation follows a sequential increment pattern for Investor user IDs, such as INVB1, INVB2, etc. The same increment logic is applicable to INVS user IDs as well.