Strategy Studio: LimeCPPExecutionHandler (Execution Handler)

LimeCPPExecutionHandler is Strategy Studio's binary interface to Lime Brokerage's Trading Server. Orders can be submitted to many equity and options venues, including several dark pools. The available order parameters are extensive and include many route-specific settings.

This execution handler may be selected by including the following line in the server's main configuration file:

- EXECUTION_HANDLER_MODE=LimeCPPExecutionHandler

Session Level Settings

Session-level settings are specified in the file execution_handler-conf/lime_cpp_execution.config with format:

#StrategyStudioAccount,StrategyStudioFirm,LimeAccount,TS3Host,[SecondaryTS3Host,]TS3Username,TS3Password,CancelAllOnDisconnect,Infiniband,IsOptions

- StrategyStudioAccount
  - Sets the Strategy Studio account that will be associated with this session. This value in conjunction with OptionsEnabled must be unique for each session.
- StrategyStudioFirm
  - Sets the firm this session will be associated with for auto-marking purposes.
- LimeAccount
  - This corresponds to a Lime account and will be provided by Lime.
- TS3Host
  - Specifies the IP of the TS3 server for this session.
- SecondaryTS3Host
  - Optional; secondary host for failover purposes; will be attempted if the primary fails.
- TS3Username
  - This is the username for the Lime account.
- TS3Password
  - This is the password for the Lime account.
- CancelAllOnDisconnect (true/false)
  - Determines the behavior for open orders on disconnect. The recommended value is true.
- OptionsEnabled (true/false)
  - If true, routes options orders for StrategyStudioAccount through this session.

External fills originating from Portal or another EMS may be routed to a strategy by populating tag 9050 with an instance name. This corresponds to the 'client order data' field in Portal's Order Entry window. Additionally, a default instance can be configured by adding AttributeExternalOrdersTo=*strategyName* on its own line to lime_cpp_execution.config.

Automatic Side Marking

By default, automatic side marking in the LimeCPPExecutionHandler is disabled. This determines whether new sell orders will automatically have their side marked based on the firm's position. To enable this behavior, add AutoMarkSide=*true* on a line by itself. Lime ACS configuration will be required (instructions below) if auto-marking is enabled.

Lime's TradingServer also has automatic side marking capabilities. When possible, it is recommended that clients enable this on their accounts to avoid rejects that can occur if Strategy Studio's side determination conflicts with Lime's Trading Server rule (if for example in-flight data has changed the appropriate side).

Lime ACS Configuration

Automatic side marking requires Lime ACS to initialize positions. The connections to ACS are configured with execution_handler-conf/lime_acs.config by entries with format:

#Hostname,Username,Password,[Account[|Account]...]

- Hostname
  - Specifies the IP for Lime's ACS server.
- Username
  - The Lime username for the ACS account.
- Password
  - The password for the ACS account.
- Account
  - An optional pipe-delimited list of the TS3 accounts associated with this ACS connection. If omitted, this connection will be used for all TS3 accounts.

Route Level Settings

Route level settings are specified in execution_handler-conf/lime_route.config. This file follows the same format as lime_route.config required by the LimeFIXExecutionHandler; details can be found in that execution handler's documentation.

Custom Order Parameters

Common order properties are set using OrderParams's fields. Additional properties that are not supported by OrderParams can be set via OrderParams's custom_params in the same manner as with the LimeFIXExecutionHandler.

Note that the first element of the std::pair<int, std::string> pushed into custom_params is expected to be the FIX tag that corresponds to the Lime C++ API setting desired. It's also important to note that where a field's expected value format differs, the FIX version should be preferred. For example, in the Lime C++ API, the nearPegOffset field expects an integer between 0 and 100 in increments of 10, while Lime's FIX tag 9069 expects a floating point number between 0 and 1 in increments of tenths. The FIX version is expected in custom_params so that the adapters can be used interchangeably.