MT4 Link
MQL4 Server/Client App like communications framework

version	1.00
Date	2019-10-31
Developer	Chamal Abayaratne

Mt4link framework is a development platform which make life easier for developers who are looking to develop Meta Trader 4 applications that communicate over the internet.

For applications to communicate over the internet, they need a server-client configuration. Mt4link is developed specially for mql4 platform with a n easy to work with API.

Features:

- Control long polling speed remotely
- Instructions can be sent to all clients or by individual trading terminals
- Built in support for file transfer (upload and Download within the sandbox)
- <4KB network usage per request
- No need of dll module imports for clients

Architectural overview

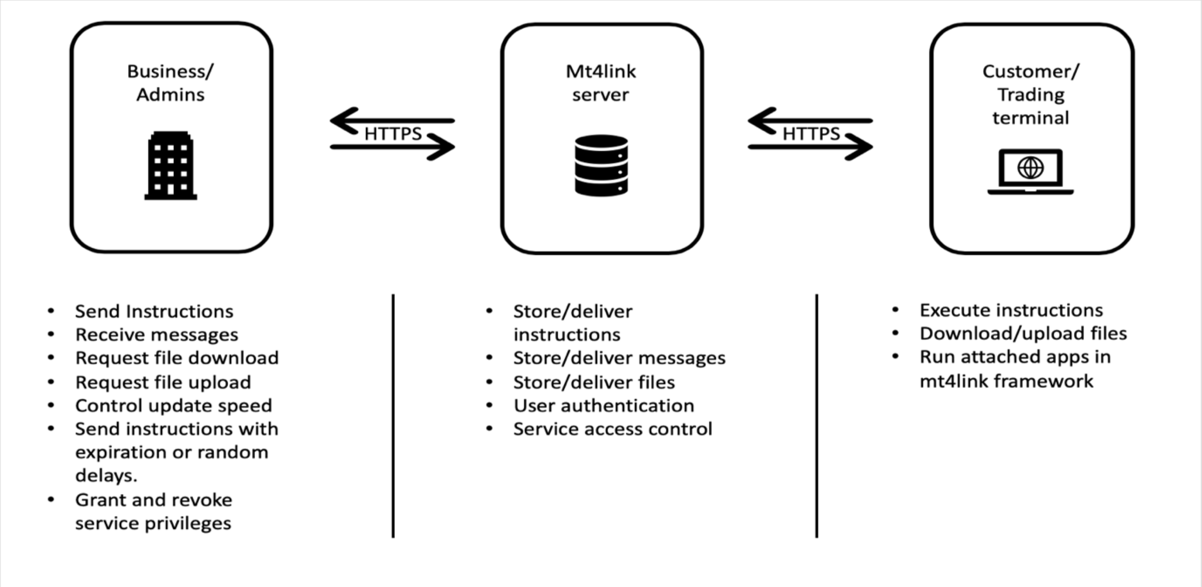


Figure 1.0

Server Deployment is the easiest step of all

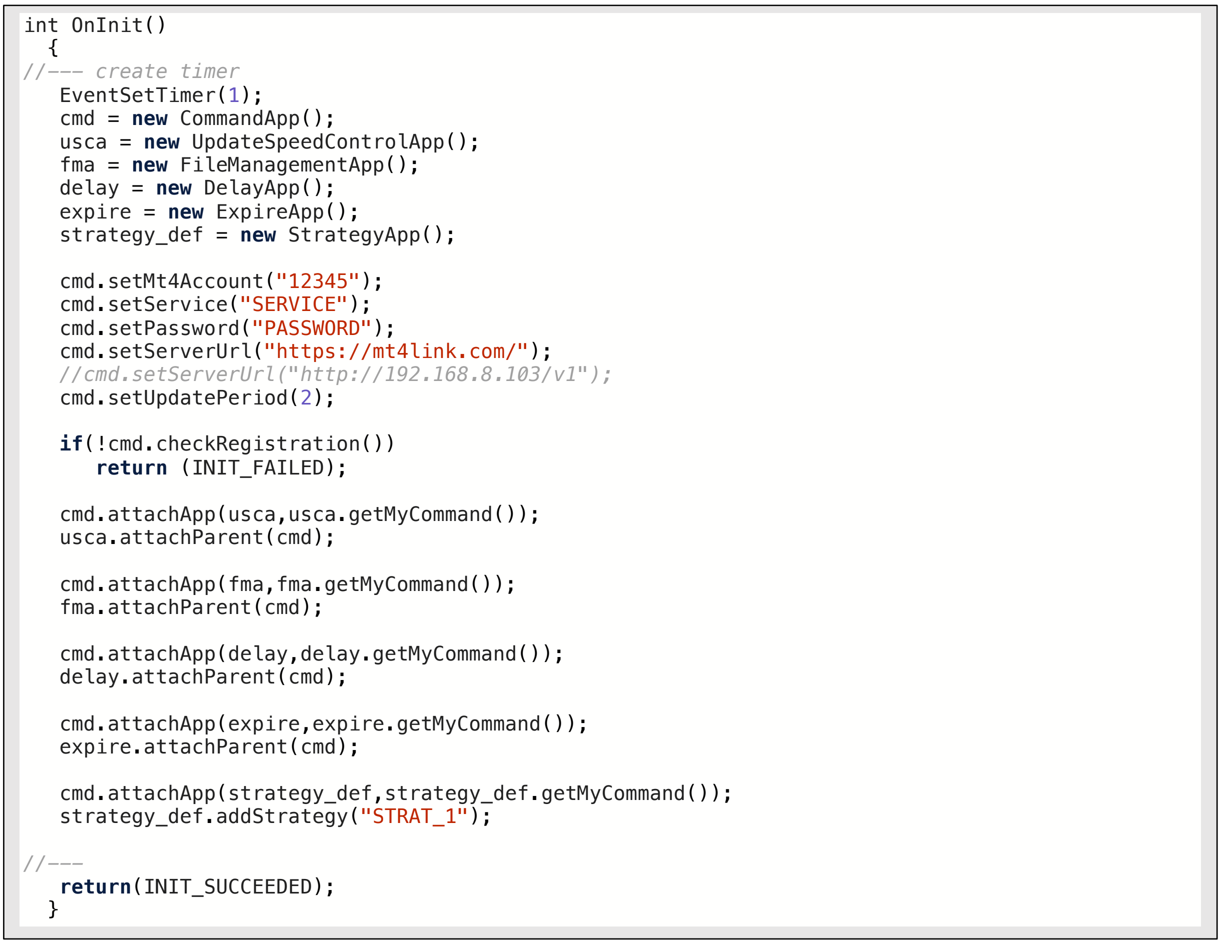
Deploying an intermediate server is the most frustrating step of all client-server configurations. However, mt4link uses a deploy-and-forget approach eliminating need to update servers to accommodate new functionality every time.

Server requirements:

1. Debian based Linux distribution (tested on 16.04 LTS)
2. Apache 2.4.41
3. MySQL 5.7
4. Python 2.7

Client Deployment Example

example.mqh



```
int OnInit()
  {
//--- create timer
   EventSetTimer(1);
   cmd = new CommandApp();
   usca = new UpdateSpeedControlApp();
   fma = new FileManagementApp();
   delay = new DelayApp();
   expire = new ExpireApp();
   strategy_def = new StrategyApp();

   cmd.setMt4Account("12345");
   cmd.setService("SERVICE");
   cmd.setPassword("PASSWORD");
   cmd.setServerUrl("https://mt4link.com/");
   //cmd.setServerUrl("http://192.168.8.103/v1");
   cmd.setUpdatePeriod(2);

   if(!cmd.checkRegistration())
      return (INIT_FAILED);

   cmd.attachApp(usca,usca.getMyCommand());
   usca.attachParent(cmd);

   cmd.attachApp(fma,fma.getMyCommand());
   fma.attachParent(cmd);

   cmd.attachApp(delay,delay.getMyCommand());
   delay.attachParent(cmd);

   cmd.attachApp(expire,expire.getMyCommand());
   expire.attachParent(cmd);

   cmd.attachApp(strategy_def,strategy_def.getMyCommand());
   strategy_def.addStrategy("STRAT_1");

//---
   return(INIT_SUCCEEDED);
  }
```

How MT4 link framework works under the hood

The mt4link-server delivers instructions to clients based on service accounts. Clients will only get instructions from service to which they have registered to/ Provided credentials. This control is implemented in server level.

The client side of mt4link-client will continuously check with mt4link-server for new updates. Updates frequency will default to one request per second. However, this can be controlled by an instruction (see USCA command).

The main application which handles incoming instructions is also called mt4link internally. This app delivers instructions to other mt4link standard apps or user defined apps.

Notes:

1. All instructions which belong to a particular service account is visible to all subscribers/mt4link-clients.
2. Mt4link uses http POST requests over the HTTPS protocol. HTTP requests will be redirected to https. Standard port 443 for https is used.

App Standard Commands

... - other instructions

1. CMD

```
CMD ALL ...
CMD [terminal number] ...
```

CMD must be called in each instruction. Otherwise, the instruction will be ignored for execution. The second parameter should be either ‘ALL’ or mt4 terminal number(int32).

ALL – Will be further procced by all mt4link-clients
Mt4 terminal number – Will only be processed by corresponding terminal

2. EXPIRE

```
... EXPIRE [UNIX time] ...
```

EXPIRE is an app that only passes control to the rest of the instruction if specified unix timestamp is not passed.

This command is mainly intended to set an expiration to instructions.

3. DELAY

```
... DELAY [Seconds] ...
```

DELAY pauses the app for random period of time but not more than specified in the second parameter. Then it will pass the control to the rest of the instruction.

This command is mainly intended to reduce server overload when a reply-to-instruction-by-message event occurs.

4. FMA

```
... FMA UPLOAD [file path]
... FMA DOWNLOAD [file path]
... FMA DELETE [file path]
```

This app can handle file upload and download to and from mt4link-server. Also is used to delete files in local storage.

The file path should be in relative to the mql4 sandbox.

FMA is limited to handle files only in mql4 sandbox.

5. USCA

```
... USCA [update period in seconds]
```

USCA controls the update speed of the mt4link-client with the mt4link-server.

Second parameter is the number of seconds between each update.

6. STRATEGY

```
... STRATEGY [strategy name] [trade controller commands]
```

STRATEGY is the most complex app out of the standard mt4link-client app bundle. This app handles all trading related operations.

STRATEGY works with another sub-app ‘trade controller’ to deliver its purpose. It is important to know how both apps work together.

This app basically allows to have some degree of control over incoming trade instructions. The strategy app will only open new orders only if the strategy name is added to the mt4link-client at the initialization stage.

All other instructions other than OPEN will be passed for further processing incase if the terminal contains orders which were previously subscribed.

a. Trade controller

Trade controller app takes instructions from the strategy app to perform trading operations.

Open order syntax:

```
[id] OPEN [BUY/SELL] [PENDING/MARKET] [SYMBOL] [PRICE] [MAGIC] [COMMENT]
```

Modify order syntax:

```
[id] MODIFY [SL/TP] [PRICE]
```

Close order syntax:

```
[id] CLOSE
```

Trading lot sizes should be defined at the initialization stage. This cannot be specified via the instruction.

Order comment can be set to the strategy name. This is done to easily distinguish orders from different strategies. However, persistence of order comments cannot be guaranteed because some brokers modify order comments to display broker-side actions to an order (ex- Margin calls, SL or TP hits).

Trade controller can handle both market and pending orders. For market orders slippage will always be 1.

Trade controller can automatically decide what type of order should be placed. No need to specify limit or stop suffixes. It will automatically determine order type by analyzing current market price and specified position bias.

A unique identifier for each order must be maintained.

b. Trade Context

Trade context refers to three factors which is required to perform an order operation. They are namely broker connection, trading permission and broker server status.

If the trade context is not ready the instruction will be cached and will be tried again later until success. The cached instructions will be saved in the disk therefore they will be available to re-execute later even after a system restart.

c. Error Handling

If market is closed the instruction is cached and will re-run when the market is open.

If broker returns an error the instruction will be tried three more times in defined intervals.

Syntax

Mt4link has no complex syntax rules. There are only two simple rules

1. Each command/parameter should be separated by only one space.
2. Beginning and the end of an instruction should not contain any spaces

Server Configuration (tested)

Server platform	Google Cloud Services, Compute Engine
Instance type	f1-micro (1 vCPU, 0.6 GB memory)
Operating System	Ubuntu 16.04.6 LTS
Web Server	Apache 2.4.41
Database	MySQL 14.14 Distribution 5.7.28, for Linux (x86_64)
Web server backend	Python 2.7 with libapache2-mod-wsgi

Client Configuration (tested)

Platform	Intel x86_64
Instance type(s)	Desktop/PC/Laptop/Virtual Machine
Operating System	MS windows 7,8,10
MetaTrader Version	4.00 build 1220

Performance Statistics (with approx. 10,000 records stored)

requests per second - HTTP	900-1200
requests per second - HTTPS	400-600
requests per second – HTTPS + DBMS	100-200
CPU load	~5% usage, 10 concurrent users

Many Possibilities with mt4link

Due to message/instruction passing architecture of mt4link, customers can request to create Meta Trader 4 applications which can enhance their communication with clients. Use sophisticated methods of customer service and many other possibilities.

Planned improvements and upgrades

1. Client logs upload upon request for troubleshooting
2. Periodic order commission calculate
3. Mt4link files to be stored based on account numbers