

# Loading Triarch Data into the MIM Server



12301 Research Blvd.  
Building IV, Suite 410  
Austin, TX 78759

U.S. Help Desk Phone: +1-800-546-9646 (or direct +1-512-697-3000), select ext. 3400  
U.K. Help Desk Free Phone: 0800 032 6063  
Europe Help Desk Phone: +44 20 7190 2947  
Help Desk Email: [support@lim.com](mailto:support@lim.com)  
+1-512-697-3001 (Fax)

Part Number: 083\_38  
Date: March 6, 2008

Copyright © 2002-2008 by Logical Information Machines, Inc.

Patented May, 1995 U.S. Patent No. 08/392, 612

All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of Logical Information Machines, Inc.

Restricted Rights Legend

Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subdivision (c) (1) (ii) of the Rights in Technical Data and Computer Software clause at 252.227-7013.

Logical Information Machines, Inc.  
120 North LaSalle Street  
Suite 2150  
Chicago, IL 60602

(312) 456-3000

Product names mentioned herein are for identification purposes only and may be trademarks and/or registered trademarks of their respective companies.

While every precaution has been taken in the preparation of this manual, we assume no responsibility for errors or omissions. Neither is any liability assumed for damages resulting from the use of the information contained herein. Logical Information Machines, Inc. may revise this publication from time to time without notice.

---

# Table of Contents

<b>Loading Triarch Data into the MIM Server .....</b>	<b>1</b>
Overview .....	1
Expectations .....	2
LIM Requirements .....	2
Client Requirements .....	2
Initial Setup .....	2
Setting Up the Queue .....	3
Checking the Queue .....	3
Removing the Queue .....	3
Loading the Library Path .....	4
Seeding the Database .....	4
Setting Up the Configuration File .....	7
How to Run/Stop Programs .....	9
Starting the ssl_xmim Program .....	9
Stopping the ssl_xmim Program .....	9
Starting the xmim_rt_update Program .....	9
Stopping the xmim_rt_update Program .....	10
Checking the Status .....	10
Pulling Data from the MIM Server .....	10
Troubleshooting .....	11
Trouble Fetching Data .....	11
Debugging .....	11
<b>Index .....</b>	<b>13</b>



# Loading Triarch Data into the MIM Server

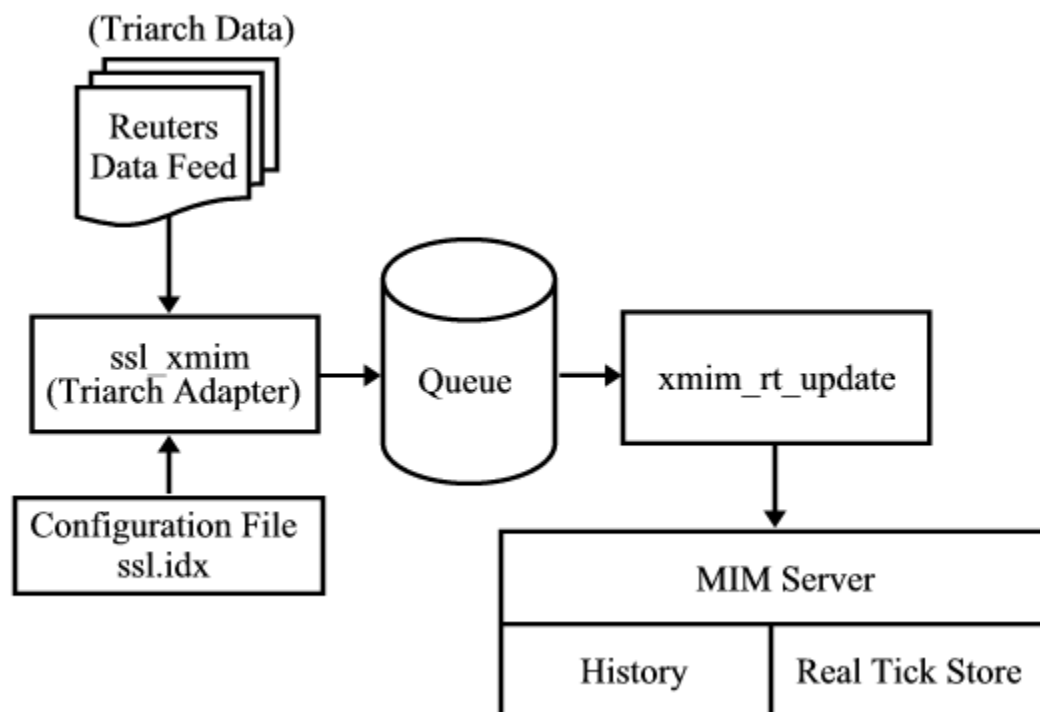
## Overview

This document shows the process for pulling Triarch data from a Reuters® data feed into the MIM server for use with LIM's historical analysis tools.

The following diagram shows how the data is passed from the Reuters data feed into the queue then into the MIM server. There are two programs that process the data: `ssl_xmim` and `xmim_rt_update`. The `ssl_xmim` program pulls the data from the Reuters data feed and puts the data into the queue. The `xmim_rt_update` program then pulls the data from the queue and puts it into the MIM server.

The configuration file `ssl.idx` is where you tell the program what kind of data needs to be pulled from the Reuters data feed. The queue is a temporary holding area for the data while it's being processed into the MIM server.

There are two areas in the MIM server for storing data: "History" and the "Real Tick Store". Client data is stored in the Real Tick Store repository while the History repository is for LIM processed data. The History repository is optimized for fast reading of the data while the Real Tick Store is optimized for fast writing of the data.



## Expectations

The requirements and expectations for loading Triarch data into the MIM are outlined below.

## LIM Requirements

### LIM Administrator

An administrator from LIM will come to the client's site and setup the following two items:

- MIM Installation: Solaris machine and MIM server.
- Databases to collect tick information and nightly updates.

## Client Requirements

### Tick Collection Administrator

The client is responsible for having one dedicated person who can administer tick collections. This person must understand UNIX and be proficient in vi and shell commands. This person must have an understanding of the following data collection concepts:

1. Loading the Triarch SSL libraries and how to access Reuters data from the Solaris machine. The SSL demo program `tickr` is used to test access to Reuters data.
2. How to configure tick collections: The client must provide a map from Reuters RIC codes to MIM symbols in order to setup the `ssl.idx` file. The client will need to coordinate with LIM to setup tick collection software that must be installed on the Solaris machine.
3. How to start/stop processes. The tick collection programs: `ssl_xmim` and `xmim_rt_update` are used.
4. How to load meta data and seed with the BMIM program.
5. If futures data is collected, must understand how futures in the MIM works.
6. How to retrieve collected ticks from the MIM.
7. How to verify that ticks have been collected.

## Initial Setup

There are a few items that need to be setup before data can be passed to the MIM server.



All the binaries are located in `/home/lim/xmim/bin`.

## Setting Up the Queue

To setup the queue, install the `xmim_rt_setup` utility by entering the following:

```
%touch xmim_rt_queue_1
```

This creates the empty file `xmim_rt_queue_1`. Create the file with any name you want. Next, enter:

```
%xmim_rt_setup xmim_rt_queue_1
```



You must run the `xmim_rt_setup` utility each time the machine boots.

## Checking the Queue

Enter the following commands to check that the queue was created:

```
%xmim_rt_stat xmim_rt_queue_1
```

This will return a message stating: “The queue is empty.” You can also check the queue after data begins to be pulled from Reuters. The utility will show how many messages are in the queue.

## Removing the Queue

The following instructions outline how to remove the queue if needed. Rebooting the machine will always free all allocated queue resources.

You must remove the queue before removing the file `xmim_rt_queue_1` otherwise the queue can only be removed by rebooting. The following command removes the queue and all of its allocated resources without rebooting.

```
%xmim_rt_setup -k xmim_rt_queue_1
```

This command removes the file `xmim_rt_queue_1`:

```
%rm xmim_rt_queue_1
```

## Loading the Library Path

Enter the following command in the shell startup script for the user's login account. The following command line entry is for a Bourne shell:

```
Export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:<path to ssl libraries>
```

## Seeding the Database

In the MIM server, there are two data locations: the History and the Real Tick Storage areas.

For each relation/column that you want real tick data for the in the Real Tick Storage area you must have one tick data point in the History area.

Check the History to make sure that historical data exists for each relation column. If no historical data exists you must add a data point. Putting the data points into the History is referred to as “seeding the database”.

The following outlines the process for seeding the database.

## Setting Up Relations/Columns in the History

The first step in setting up the History is to make sure that the relation and column values needed for the Real Tick Store exist in the History area of the MIM server. If the relations/columns that you need do not exist in the History on the MIM server than they must be added using the `bmim_client` program together with BMIM scripts.

For detailed documentation on writing BMIM scripts for setting up symbols, relations and columns see the section on “[Updating a Database](#)” in the “BMIM Scripting Language” chapter of the *MIM Data and Development Guide*.

For information on symbol naming requirements see the “[Database and Data Guidelines](#)” chapter and the “[Futures Type Data Naming Conventions](#)” chapter in the *MIM Data and Development Guide*.

The following shows a brief example of how to add a symbol and a new relation and column to the History database. The example script is named `makerelations.txt`.



“# Grouping ...” is a BMIM comment and will be ignored when you run the script.

```

# Grouping 1
lock_files
relation_add {
name=MyData;
parent=TopRelation;
type=category;
}
# Grouping 2
relation_add {
name=MYDATA1;
parent=MyData;
type=normal;
description="ACME Data";
}
# Grouping 3
column_add {
name=MyColumn;
parent=TopColumn;
type=category;
}
# Grouping 4
column_add {
name=Storage;
parent=MyColumn;
type=normal;
}
# Grouping 5
relation_column_add {
relation=MYDATA1;
column=MyColumn:Storage;
type=base;
}
unlock_files

```

The following explains the example script:

The `lock_files` command is required to lock the database when adding or changing data. At the end of the script, the `unlock_files` command is a required field to unlock the database.

**Grouping 1:** The `relation_add` command creates the top category “MyData”.

- “name” designates the new name of the category.
- “parent” by definition is the category above which the new category will be created, in this case the top relation.
- “type” in this case is set to “category” to create a category. Other options include: “normal” and “base”. “Normal” is used to create symbols. “Base” is used with `relation_column_add` to create a relationship between a relation (symbol) and a column. Other options include “futures”, “futures\_continuous” and “futures\_contract”. See the “[BMIM Scripting Language](#)” chapter in the *MIM Data and Development Guide* for more detailed information.

**Grouping 2:** The `relation_add` command creates the symbol “MYDATA1”.

- “name” designates the new symbol name.
- “parent” shows that the symbol will go in the newly created “MyData” category.
- “type” being set to “normal” designates that a symbol will be created.
- “description” defines a description for the symbol.

**Grouping 3:** The `column_add` command creates the top column “MyColumn”.

- “name” designates the new column category.
- “parent” defines the category to sit below the TopColumn in the hierarchy.
- “type” is set to “category” to define “MyColumn” as a new column.

**Grouping 4:** The `column_add` command creates the column “Storage”.

- “name” designates the new column name.
- “parent” shows that the column will be created in the “MyColumn” category.
- “type” is set to “normal” because it will not have sub-categories and will be storing data.

**Grouping 5:** The `relation_column_add` command links the new symbol “MYDATA1” to the new column “MyColumn:Storage”.

- “relation” designates the symbol to be linked.
- “column” shows the column to be linked.
- “type” being set to “base” designates that the data will be entered daily.

Load the script by entering the following command:

```
bmim_client makerelations.txt
```

## **Setting Up the Data Files**

Now that the relations and columns have been created in the History we need to load a data point for each symbol into the MIM.

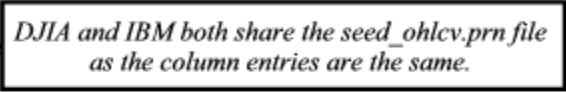
### **Creating BMIM Scripts for Loading Data**

Create a BMIM script to define what data is going to be inserted into the History. In the BMIM script there is a reference to a file that contains the data to be loaded. This file that can be shared by multiple entries in the BMIM script if the column data is the same for each entry.

In the following BMIM script, the file `seed_ohlcv.prn` is shared by the DJIA and IBM relations as the column information is the same for both symbols. The relation KG has different column entries (Bid and Ask) therefore a separate file `seed_ba.prn` is associated with this symbol entry.

The following example script is named `load_data.txt`.

```
lock files
facts_read {
  relation = DJIA;
  column = Date, Time, Close, Open, High, Low, Volume;
  file = seed_ohlc.v.prn;
  tick_intraday;
}
facts_read {
  relation = IBM;
  column = Date, Time, Close, Open, High, Low, Volume;
  file = seed_ohlc.v.prn;
  tick_intraday;
}
facts_read {
  relation = KG;
  column = Date, Time, Bid, Ask;
  file = seed_ba.prn;
  tick_intraday;
}
unlock files
```



*DJIA and IBM both share the seed\_ohlc.v.prn file as the column entries are the same.*

## Creating the Data File

Next, you will need the file containing the data points you want to input.

It is only necessary to input data for the Date and Time entries. For the rest of the column entries put a space and a comma e.g., Close, Open, High, Low, and Volume.

The following is an example of the `seed_ohlc.v.prn` data file.

```
20011130, 155900, , , , ,
```

## Loading the Data with `bmim_client`

Enter the following command to load the data into the History on the MIM server using the `bmim_client` program:

```
bmim_client load_data.txt
```

The MIM server is now prepped and ready for data to be sent across from the Reuters data feed through the queue and into the MIM server in the Tick Data Store.

## Setting Up the Configuration File

The configuration file is setup by the user to determine what data will be pulled from the Reuters data feed. For our example, the configuration file is named `ssl.idx` but the user can choose any name with the `.idx` extension.

Use this configuration file to setup the symbols and corresponding data information that you want pulled from the Reuters data feed. This configuration file will take the symbols and information listed and will convert the RIC names into the naming convention used by the MIM.

The following example shows one symbol in a configuration file. Enter the MIM symbol name on the left and the RIC name on the right. Use “{}” to enclose the corresponding date, time, and the corresponding MIM column value to be pulled from the Reuters data feed. For this example we’re looking for a Bar value.

Example of one symbol in the configuration file ssl.idx:

```
DJIA.DJI
{
dateTRADE_DATE
timeTRDTIM_1EST5EDT
BarTRDPRC_11.0
}
```

where:

- DJIA is the MIM relation name and .DJI is the Reuters RIC code.
- “date” is the date value used by the MIM and “TRADE\_DATE” is the Reuters field name that contains the data value.



If the date field is not included in the Reuters data packet, then the date is pulled from the computer’s clock setting.

- “time” is the time value used by the MIM and “TRDTIM\_1” is the Reuters field name that contains the time value. For each individual record you can specify the time zone. Set the time zone to “EST5EDT” for the east coast and to “CST6CDT” for the central time zone.
- “Bar” is the MIM column value. Examples would include values such as High, Low, Close, Volume, Bid, Ask or Bar. The Reuters field “TRDPRC\_1” will return the corresponding Reuters “Last Traded Price” value. “1.0” is the multiplier value. The data will be multiplied by the supplied value and is used, for example, to format the data to a specific decimal place. This value must be supplied. For more information about Reuters field names, please consult your SSL programming guide.

The following table is an example of the type of data that might be pulled into the MIM server using the example configuration file:

<b>MIM Relation:</b>	<b>DJIA</b>
MIM Column:	Bar
Date:	12/3/2001
Time:	11:33:20
Value:	9700.00

## How to Run/Stop Programs

The following instructions show how to start and stop the program `ssl_xmim` that pulls data from Reuters and the program `xmim_rt_update` that pulls data from the queue into the MIM server.

### Starting the `ssl_xmim` Program

To start the `ssl_xmim` program to pull data from Reuters into the queue do the following:

```
%ssl_xmim -s <servicename> -q <queue_filename> - i <index_file> - a <appendix_file>
```

where:

- `<servicename>` this is the Reuters service name, in many cases it will be “IDN\_SELECTFEED”. Please consult your Reuters engineer for clarification.
- `<queue_filename>` for this example is `xmim_rt_queue_1`.
- `<index_file>` this is the configuration file `ssl.idx`.
- `<appendix_file>` this is the appendix file from the Triarch installation.

### Stopping the `ssl_xmim` Program

To shut down the `ssl_xmim` program enter the following commands:

```
%ps -ef | grep ssl_xmim
```

This will return the `<process_id>` number needed for the next command line below. The `<process_id>` number is the first number listed.

```
%kill <process_id>
```

### Starting the `xmim_rt_update` Program

To start the `xmim_rt_update` program to pull data from the queue into the MIM server enter:

```
%xmim_rt_update <queue_filename> <hostname> <port> <config_file> <pid_file>
```

where:

- <queue\_filename> for this example is `xmim_rt_queue_1`.
- <hostname> is the server host name.
- <port> is the server port number.
- <pid\_file> filename that the user creates.

## Stopping the `xmim_rt_update` Program

Use the following command to stop the `xmim_rt_update` program.

```
%cat <pid_file>  
%kill <process_id>
```

## Checking the Status

Use the `xmim_rt_stat` utility to check the number of messages in the queue.

## Pulling Data from the MIM Server

The data is now in the Real Tick Store on the MIM server and is ready to use with one of LIM's analytical tools, for example: XMIM, MIMIC or the `xmim_get` utility.

# Troubleshooting

## Trouble Fetching Data

If you are unable to fetch real tick data from the MIM, consider the following possibilities:

1. Did you remember to seed the database?
2. Are the times that you inserted within the relation trading times? For example, did you pick a weekend date or a time such as 3:00 p.m. when trading may not be occurring?

## Debugging

Use the **-p** operator to display each record sent to the queue.

Example usage:

```
%ssl_xmim -s <servicename> -q <queue_filename> - i <index_file> - a <appendix_file> -p
```

Use the **-v** operator to show what messages are coming out of the queue.

Example usage:

```
%xmim_rt_update -v <queue_filename> <hostname> <port> <config_file> <pid_file>
```



# Index

## B

BMIM Scripts for Loading Data, 6  
 bmim\_client, 7

## C

Check Status, 10  
 column\_add, 6  
 Configuration File, 7

## D

Debugging, 11

## H

History, 4

## L

lock\_files, 5

## P

Pull Data from MIM Server, 10

## R

Real Tick Storage, 4  
 relation\_add, 5  
 relation\_column\_add, 6  
 Requirements, 2  
 Run/Stop Programs, 9  
   Start ssl\_xmim Program, 9  
   Start xmim\_rt\_update Program, 9  
   Stop ssl\_xmim Program, 9  
   Stop xmim\_rt\_update Program, 10

## S

Setup, 2  
   Check Queue, 3  
   Load Library Path, 4  
   Remove Queue, 3  
   Seed Database, 4

  Setup Queue, 3  
 ssl\_xmim, 9, 9

## T

tickr, 2  
 Troubleshooting, 11

## X

xmim\_rt\_setup, 3  
 xmim\_rt\_stat, 3, 10  
 xmim\_rt\_update, 9, 10

