Commodity DataServer

# Best Practices Guide

MORNINGSTAR®

## Table of Contents

## Introduction

The Commodity DataServer is a patented high performance Time Series Database designed to provide exceptional data processing and retrieval performance.  The server is a Linux/Unix server running the Commodity DataServer application.  This application is a set of services and scheduled tasks which retrieves data from the Morningstar Commodity Data Warehouse, loads the data into the Time Series Database, and answers data requests. The system should be administered and maintained just like any database system such as Oracle or Microsoft SQL servers.    This document is designed to provide guidance on choosing a Backup and Disaster Recovery plan, Determining your Recovery Point Objective and recommendations for proper system administration to insure high system availability.  Following these guidelines will enable the Commodity DataServer to perform at the highest levels, provide uninterrupted service, and deliver the critical data needs for our valued clients.

There are several options available to insure your Commodity DataServer's availability, resilience, and uptime.  This document outlines all of the common options available, including how they will affect your users, their requirements, and limitations.  If none of the below options are appropriate to your application, please contact your sales representative and we can work with you to fulfill your needs.

## Commodity DataServer System Backup

While the Commodity DataServer system is robust, the most important supporting function a business must implement for the server is a routine system backup.  The frequency of backup should be on a schedule that will meet the businesses recovery objective.  A backup allows for a recovery to the point in time the system was last backed up, so the business is required to insure the gap between a systems recovery and the time it takes to bring the system back up to date is within an acceptable tolerance.

The time it takes a system to become current after the restoration of a backup is dependent on the Commodity DataServer systems performance.  Commodity DataServer systems which do not meet the published guidelines for minimum Commodity DataServer system requirements may require more frequent backups to insure the recovery window is small enough to allow the system to become current in the shortest acceptable time frame. Commodity DataServer system requirements can be found on www.morningstarcommodity.com.

A business must ask the system users what the longest amount of time the system could be unavailable or have outdated data before it impacts the business. The backup frequency should then be calculated to meet the business user's requirements. Morningstar Commodity's Account Managers and technical support staff are available to provide guidance and suggestions.

## Disaster Recovery

In the event that the Commodity DataServer system is completely unavailable or the site hosting the system goes offline, a Disaster Recovery (DR) plan should exist. If the service delivered by the Morningstar Solution is a critical business need, please include provisions in your DR Plan to accommodate this service.

Before integrating the Commodity DataServer into your production processes, it is important to consider the impact of a failure. Although failures (hardware or software) are very rare, as with any IT system, it can happen and proper precautions need to be taken to minimize the likelihood of a failure event so as to enable a quick recovery.

The Commodity DataServer should be an enterprise class system designed for high availability (often referred to as a server class machine). Typically found on these systems are redundant power supplies, integrated remote management, and system alerting.

If your company can go without access for several days without significant impact, a single Commodity DataServer for a given set of data is generally appropriate. If there would be a significant financial impact if the Commodity DataServer was not accessible for several minutes to several hours, Morningstar suggests having a backup Commodity DataServer. See the chart under the Importance of RPO to determine the redundancy method to implement.

If you have any concerns regarding Commodity DataServer uptime, please discuss them with your sales representative prior to deciding exactly what service to purchase. The business users of the Commodity DataServer system should be asked in the event the main system is lost, what is the tolerable amount of downtime.   This is another area where Morningstar Commodity's Account Managers and technical support staff are available to provide guidance and suggestions.

## The Importance of Recovery Point Objective

The acceptable time period it takes to have a Commodity DataServer system available with up to date data after a failure is called the Recovery Point Objective (RPO).  Because the majority of Commodity DataServer users require the Commodity DataServer system to be populated with current (up to date) data, the Recovery Time Objective (RTO) is considered to be equal to the RPO.

Generally, building a system that is designed to be highly available is significantly more expensive than building a simple system with many single points of failure.  As such, your RPO goal will define not only the backup schedule but also a recovery solution that meets the acceptable time frame of the systems users.  The importance placed on being able to query data from the Commodity DataServer and the impact of an outage should give you a good idea of the RPO required by the Commodity DataServer system business users.

In the event a primary Commodity DataServer becomes unavailable, the chart below shows the options for system designs necessary to meet RTOs from immediate to three days.  There are two scenarios to consider, one where the Commodity DataServer only receives Core Data feeds and the other where the system is loaded with proprietary custom data.

| Abbreviation | Description |
|---|---|
| **NS** | New server created and restored from backup |
| **RB** | Primary server restored from backup |
| **CS** | Primary server with Cold Standby data restored from backup |
| **HUS** | Primary Server with Hot Updated Standby |
| **HDR** | Primary Server with data replicated to Hot Standby |
| **HUSM** | Primary Server with Hot updated Standby and multiplexed custom data loads |

| RPO | Immediate | 1 hr | 4-8 hrs | 8-24 hrs | 1-5 days | 5-30 days |
|---|---|---|---|---|---|---|
| **Custom Data** | HUSM | HUSM | HDR/RB | HDR/RB | CS | NS |

| | | | | | | |
|---|---|---|---|---|---|---|
| **Core Data** | HUS | HUS | HUS/RB | CS/RB | CS | NS |

When a Commodity DataServer system is restored/recovered, the time it takes for the system to receive and processes updates must be considered. The length of time for the system to catch up from being offline will be dependent on the time it takes to receive the data the Morningstar Distribution system and the performance of the Commodity DataServer to process the data. The faster the system the faster the data will be processed and available for the business users. Systems that have been offline for more than two weeks may need to have a new database shipped from Morningstar Commodity.

## Commodity DataServer Operating Guidelines

The Commodity DataServer is a Linux/Unix server running the Commodity DataServer application. This application is a set of services and scheduled tasks. At the basic level the Commodity DataServer retrieves data from the Morningstar Commodity Data Warehouse, loads the data into the Time Series Database, and answers data requests.

### *System Recommendations and Requirements*

Morningstar Commodity is constantly evaluating hardware and performance metrics in order to determine system requirements. Please contact your Account Manager or visit Commodity DataServer Requirements (http://www.morningstarcommodity.com/node/13) for the current system requirements.

### Commodity DataServer Backups

Backing up a Commodity DataServer instance consists of copying the Commodity DataServer databases and configuration files while updates are not being applied. This can be difficult when custom data is being written to the database at unpredictable times, but is quite simple in most instances where custom data isn't being loaded, or is being loaded with standard update packages as Morningstar recommends. This document will first outline the simple process that should be used in most cases. Then it will build on it to cover the latter case where a backup window can't be used. The directions provided in this example are meant to be a generic guide and it should be understood that each environment may require adjustments due to specific system configurations.

A backup of the Commodity DataServer database can be taken by copying the dates, xmimrc, and data directories referenced in the server's .xmimrc file. Usually they reside in $LIMHOME, and make up the large majority of the space in $LIMHOME. Therefore Morningstar recommends backing up all of $LIMHOME which will include all server configuration, logs, etc.

When updating of the databases are done with standard formatted update packages, with either no custom data loaded or where custom processes use update packages, the backup process can be incorporated into the flow of updates. And since only one update package is applied to the server at a time, the database is not changing and is ready for backup. Example 1 is an example suitable for most cases where only standard update packages are used. In summary it uses an administrative update package upd_0_XTR_20080101 which is copied into the update stream via a cron job. This package calls a backup script $LIMHOME/custom/backupMimInstance.sh which in this case simply uses tar and gzip to make a copy of the Morningstar user's home directory $LIMHOME.

In a situation during which the database could be updated directly (not using packages, bmim_client, API, etc.) Morningstar recommends the use of filesystem snapshots or a similar mechanism to minimize the time required to copy the databases. The bmim_client command writer_wait can be used to quiescence the database for safe backup.

Example 2 is an example of a backup taken using ZFS snapshots.

Example For Instructional purposes only – The instructions below may require system dependant modifications.

```
#!/bin/sh

myFrame=`echo $0 | cut -d_ -f2`;
myDate=`echo $0 | cut -d_ -f4`;
myEmail='limupdates@lim.com';
execDate=`date '+%Y%m%d_%H%M'`;
user=`whoami`;

myLog="$LIMHOME/updates/logs/log.${myFrame}_$myDate"

# Functions
endScript() {
  currentUser=$( id | cut -d'(' -f2 | cut -d')' -f1 );
  hostId=$( hostid );
  userHostid="$currentUser@$hostId";
  echo $userHostid;
  if [ "$1" = "" ]
  then
```

```
    cat $myLog | mail -s "MIM Server Backup Message - $userHostid -
Successful" $CUSTNOTICES
    exit
  else
    cat $myLog | mail -s "MIM Server Backup Message - $userHostid - ERROR"
$CUSTNOTICES
    exit $1
  fi
}

# Add commands here.
$LIMHOME/custom/backupMimInstance.sh

endScript;

--------------------------------------------------

$LIMHOME/custom/.cust_limrc

backupPath=/export/limBackups

--------------------------------------------------

$LIMHOME/custom/backupMimInstance.sh
#!/bin/ksh

. $LIMHOME/custom/.cust_limrc

tar cf - -C $LIMHOME . | gzip -c >
$backupPath/${execDate}_${user}_mimBakup.tgz

--------------------------------------------------
crontab -l
5 3 * * * cp /home/lim/custom/upd_0_XTR_20080101 /home/lim/updates

--------------------------------------------------
```

## Commodity DataServer *System Monitoring*

▪The Commodity DataServer system is in constant motion processing data and running a series of system checks to alert should there be issues.  These alerts can automatically be sent to designated email addresses.  If possible, the designated Commodity DataServer system administrator or administration team should consider receiving the alerts.

▪The Morningstar Commodity Data Warehouse features the ability to receive system status notifications and alerts.  Failure to allow the Commodity DataServer system to provide status reporting back to the Morningstar Commodity Data Warehouse prevents Morningstar Commodity from being able to proactively diagnose system issues.

Update monitoring is aided by email notifications sent out when the Commodity DataServer runs the update process. There are four types of email notifications that are sent out to the email addresses defined in the .limrc file.

1. SUCCESSFUL – no errors encountered, update process complete
2. FAILURE – Errors encountered, update process was not able to continue. Errors must be located and corrected for normal database operation.
3. NOTICE – This message is a notification that a scheduled update process could not execute because another update process is already running, Only one update process at a time may run, no action is required.
4. WARNING (check log messages) – An error was encountered during the update process, but the update was able to complete. These errors are not critical and usually pertain to an individual series that did not update correctly. The error should be located and action taken to correct it.

Update logs are stored in $LIMHOME/updates/logs. These log files should be examined when errors occur to identify the error. If you need help in determining the error send the error message to commoditydata-support@morningstar.com for further assistance.

Disk monitoring is performed each time cron_updates.sh runs. Part of the output is a df of the Commodity DataServer DBA partition. Do not let the system run out of disk space as this can corrupt the database.

Server logging is enabled by default at level 0 (lowest level). Log files are stored in $LIMHOME/tmp and can contain information about who is accessing the database, what is being accessed and types of access. Log files are stored in the common log format utilized by Web servers.

The xmim_svr_info program provides information about the server and the ability to stop the server.

```
xmim_svr_info [-h] [-s host] [-p server_number] [-k] [-r] [-n]
```

Where (Optional entries are designated with brackets [ ].

| | |
|---|---|
| `[-h]` | Help/Usage. Print this message and exit. |
| `[-s host]` | Specifies the host that the master server is running on. The default will be the local host. |
| `[-p server_number]` | Specifies the *server_number* of the desired master server. If this argument as well as the -k and -r options are not specified, information will be returned for every master server that is running on the given host. The default server number if the -k or -r options are used will be 0. |
| `[-k]` | Specifies to kill a given master server. Note that the user must be either the owner of the server process or the super-user to kill a server. The default will be to kill master server 0 on the local host. |
| `[-r]` | Specifies to re-read the .xmimrc file which effectively causes the databases and any libraries/macros to be re-loaded. This will be affected only for one particular master server, the default being server 0 on the local host. |
| `[-n]` | Specifies to open a new log file with the standard log filename and begin logging to this new file. Simply renaming a log file will not result in logging to a new file because the file descriptor will still point to that file. Thus, to achieve archiving of log files, the old log file must be renamed and then the -n flag used to open a new log file, taking out a new file descriptor for it. |

## Commodity DataServer *Service Management Procedures*

The Commodity DataServer is service which runs as an application on a Unix/Linux system.  The most important item to note when shutting down the service is to insure that the system is not actively in the middle of a process update (writing data).   The technical discussion below is provided as a sample guide on how to pause, stop, check, and restart the service.   For assistance or questions regarding the steps below, please contact support at commoditydata-support@morningstar.com.

## Examples for Commodity DataServer shutdown, start-up, and application checks

The steps below are meant to be examples and may vary depending on system OS version.

- login as the user
- change directory to `$LIMHOME` which is typically `LIMHOME=/home/lim`
- `cd $LIMHOME`

## Manually Pausing Updates and Prevent Running

This may be necessary if the system is busy or you want to restart the server but do not want the system to start processing updates. To do so you must set the system to not run the update process. Change .limrc CRONRUN variable to "no" to disable cron_updates.sh updates.

```
perl -i.bak -p -e 's/RUNCRON="yes"/RUNCRON="no"/g' .limrc
grep RUNCRON= .limrc
```

## Checking the Service Status

It is also highly advisable that before stopping the Commodity DataServer database to check that updates are not running by looking for port 4091

```
ps -ef | grep 4091
```

If there is any output from the above command that contains xmim_slave_server then the process is running. Do not attempt to stop the server when this process is running.

## Estimating Processing Remaining Time

If port 4091 is present you can count outstanding update packages in $LIMHOME/updates/upd* to estimate time to completion.

```
ls $LIMHOME/updates/upd*|wc
```

Repeat above to watch progress

## Stopping the Commodity DataServer Service

Once updates have finished stop the Commodity DataServer with the following command:

```
server.info -k
```

Example output from the command:

```
/home/lim/xmim/bin/xmim_svr_info -k -p 0
xmim server 0 on hans has been killed successfully
```

## Service Checking

To check if the Commodity DataServer is running, execute the command below:

```
server.info
```

Example output

```
/home/lim/xmim/bin/xmim_svr_info -p 0
Cannot connect to server 0 on host hans ( server not running or
busy )
It's now safe to shutdown the OS.
```

## MIM Startup procedure and checks

On a normal reboot the Commodity DataServer db should auto start from `./etc/init.d/xmimserver{PORT}.` To check services have started login as the user and run the `server.info` command

```
 [lim@hans ~]$ server.info
/home/lim/xmim/bin/xmim_svr_info -p 0

master server 6400 on hans is running
        process id:     22128
        owner:          lim
        locked:         0
        log_level:      0
        log_file:       /home/lim/tmp/.xmim_server_6400.log
        database 0:     /home/lim/data/xmim.mim
        database 1:     /home/lim/data.gii/xmim.mim
        database 2:     /home/lim/data.pvm/xmim.mim
        database 3:     /home/lim/data.cust/xmim.mim
        license:        /home/lim/license/xmim.ids
        version:        Version 4.6.34.03, 64-bit, Compiled Wed Dec 31 10:57:58
CST 2008
```

```
        there are 5 slave servers registered

        slave server 0:
                host name:               hans
                process id:              22127
                owner:                   lim
                status:                  running
                last dispatched:         Thu May 14 14:52:31 2009

        slave server 1:
                host name:               hans
                process id:              22338
                owner:                   lim
                status:                  running
                last dispatched:         Thu May 14 14:52:23 2009

        slave server 2:
                host name:               hans
                process id:              n/a
                owner:                   n/a
                status:                  not running
                last dispatched:         n/a

        slave server 3:
                host name:               hans
                process id:              n/a
                owner:                   n/a
                status:                  not running
                last dispatched:         n/a
```

▪If the Commodity DataServer service didn't start on boot use the command start.server to restart the Commodity DataServer, login as the user and `cd` to home directory.

```
start.server
```

▪If cron_updates.sh was disabled by changing `.limrc CRONRUN` variable, it must be reset to "yes" to enable cron_updates.sh updates.

To do so:

```
cd $LIMHOME
perl -i.bak -p -e 's/RUNCRON="no"/RUNCRON="yes"/g' .limrc
grep RUNCRON= .limrc
```

▪After a system restart, you may want to verify that the Commodity DataServer Service is running. To do so, check if the db is readable by using xmim_get.

```
xmim_get -p 0 -r 1 CL -c 1 Close | tail
```

▪Example output

```
[lim04@euro ~]$ xmim_get -p o -r 1 CL -c 1 Close| tail
05/04/2009, 54.47000
05/05/2009, 53.84000
05/06/2009, 56.34000
05/07/2009, 56.71000
05/08/2009, 58.63000
05/11/2009, 58.50000
05/12/2009, 58.85000
05/13/2009, 58.02000
```

▪To verify that updates are working post a system reset or changes were made to the schedule:
Look at the crontab and verify that `cron_updates.sh` is scheduled to run.
Example below.

Solaris

```
0,5,10,15,20,25,30,35,40,45,50,55 * * * * /home/lim/cron_updates.sh
```

Linux

```
*/5 * * * * /home/lim/cron_updates.sh
```

Run this command to monitor successful loading of updates

```
tail -f ~/config/load_udpates.hst
```

▪The output will show the load update history, the status of the update and the last attempt.

▪Example output

```
[lim@hans ~]$ tail -10 $LIMHOME/config/load_updates.hst
hans: Thu May 14 14:20:50 BST 2009 Process make_data in
/home/lim/updates/unpacked/upd_8_net_20090514.d/net
hans: Thu May 14 14:21:02 BST 2009 cleanup processed
upd_8_net_20090514
hans: Thu May 14 14:21:02 BST 2009 finish  load_updates.sh 9.3
status 0  ###< successful update
hans: Thu May 14 14:52:04 BST 2009 start   load_updates.sh 9.3
hans: Thu May 14 14:52:04 BST 2009 BMIM Version 4.6.34.03, 64-
bit, compiled Wed Dec 31 10:57:58 CST
```

```
hans: Thu May 14 14:52:20 BST 2009 Unpack begin
upd_0_app_20090514
hans: Thu May 14 14:52:20 BST 2009 Unpack complete
upd_0_app_20090514
hans: Thu May 14 14:52:21 BST 2009 Process make_data in
/home/lim/updates/unpacked/upd_0_app_20090514.d/app
hans: Thu May 14 14:52:31 BST 2009 cleanup processed
upd_0_app_20090514
```

- NOTE: Status 0=Successful, Status 20=nothing to do or network issue if repeated over several hours. Status 50=> an error has occurred that needs attention.

### *DB Replication*

The ability to quickly perform file system snapshots can be leveraged to meet growing business requirements. While utilizing snapshots to enable more robust and timely backups is common, using this feature to provide replication of Commodity DataServer data between systems or even regional sites is also possible. Commodity DataServer DB replication via snapshots enables organizations to synchronized custom data. Snapshot synchronization does come with some limitations. The primary ones to consider are the size of the data to be replicated and the network capacity to facilitate replication. Also, when instantiating a replicated database, the application will do a reset of established connection. This will cause any ongoing queries or reads of the database to be reset as well.

Below is an example of how to implement simple one way daily snapshot synchronization to DR server. Because environments vary widely, please contact technical support to review your considerations for snapshot synchronization.

**Overview**

- What has been synchronized?
  - /home/lim - only the directories below
    - /home/lim/xmim/library - location for custom macros used via Query Languange
    - /home/lim/pub2/config - publisher set-up configuration for non-custom data (Morningstar delivered)

- /home/limih - Everything except for
    - mimSync/logs
    - mimSync/exclude
    - mimSync/rsync_backup
    - .ssh
    - .netrc
    - updates/upd_0_xtr_20010112

- Mimprimary is synchronized with mimsecondary though it's backup process run out of cron.

```
19 4 * * *
/home/limih/mimTools/mimBackup/mimBackup_writerWait_mimSync.sh >>
/home/limih/mimTools/mimBackup/logs/log 2>&1
```

- The system uses writer_wait to quiesce the Commodity DataServer, then snap the file system. Once this is complete, the process creates a package and scps it to mimsecondary. When mimsecondary processes this special administrative package it stops the Commodity DataServer instance, and uses rsync to sync all of $LIMHOME over to mimsecondary except for the explicitly excluded files/directories.

## Commodity DataServer Environmental

As with any high performance computing equipment, an organization should take responsibility for the environment needed by the Commodity DataServer to operate properly. This includes reliable power, UPS system, secure location, network connectivity, cooling, and system monitoring.

- Power
    - The Commodity DataServer should be located in a data center where condition power is provided. The Commodity DataServer should have a Uninterruptable Power Supply (UPS) connected to the server with a minimum of 15 minutes runtime should power be lost.
- Security
    - Provisions should be made to insure that the physical system is hosted in a secure location such as in a data center or computer room.
- Connectivity Requirements

    - Both the Commodity DataServer and its users will need to be able to resolve the hostname or fully qualified domain name of the Commodity DataServer.  This is necessary due to the RPC protocol that is used by the Commodity DataServer software to communicate.
    - The Commodity DataServer will need to be able to resolve dist.morningstarcommodity.com and initiate TCP flows to it on destination port 443.
    - The Commodity DataServer will need to be able to send email to Morningstar for monitoring purposes.
    - Morningstar will need some form of access to the server for management purposes. Ideally, Morningstar will SSH into the server from the Internet.  If this is not appropriate for your environment, please bring it to the attention of your sales representative so that other options can be discussed.
- Monitoring
    - Like other highly critical systems, monitoring the Commodity DataServer system's hardware and performance is an advisable proactive measure.  Configuring the Commodity DataServer's system hardware into an organizations monitoring systems or even enabling a Commodity DataServer systems internal management to send alerts for predictive failures is highly encouraged.

## Commodity DataServer *System Performance*

The system on which the Commodity DataServer service operates is the single most determining factor in terms of the systems performance. Network latency can also adversely contribute to perceived poor system performance, however, this discussion is related only to Commodity DataServers where network latency is not a factor.

A typical Commodity DataServer will process hundreds of data updates per day.  By using a consistent data set Morningstar has developed a metric to determine the average seconds per update processed as a benchmark to evaluate performance.  Below is a chart showing various hardware platforms and the seconds per update.

Morningstar recommends utilizing the fastest CPUs available when considering a new Commodity DataServer system.  Below is a chart which shows Commodity DataServer system performance across various platforms*.

| System | CPU Speed | Seconds Per Package |
|---|---|---|
| Sparc V240 | **1000mhz** | **24** |
| Sparc V240 | **1200mhz** | **20** |
| Sparc V480 | **1500mhz** | **14** |
| Intel x86 | **3160mhz** | **7** |

* The data packages used for this test were randomly chosen and should not be used for comparison to other data packages processed on a customer's specific Commodity DataServer systems.
** Sun T2 Processors are not a recommended platform.
** Disk subsystem will also impact performance metrics, it is always recommended to use Tier 1 level disks with multiple drives to ensure adequate I/O performance. Recommended to have a minimum of 4 - 15k RPM SAS drives.

## Common Troubleshooting

PROBLEM: Commodity DataServer service not starting.
Solution: Look for .lck files (lock files) that may not have been removed from a previous shutdown.
    These files are located: LIMHOME/config/{load_updates.lck & or cron_updates.lck}

PROBLEM: Commodity DataServer DB System not up to date
Solution1: Verify that load_updates are cron_updates are running correctly.
Solution2: If the system was offline for a period of time, a large amount of data could be in the queue. Verify that load_updates is processing update packages.
Solution3: Server taking longer than usual to update, please contact Support

PROBLEM: System Alerting Emails not being received
Solution1: Verify with IT staff if any recent changes were made: Check Firewall Rules, Check Proxy Servers, Check All Spam Filters, Check SMTP setting

PROBLEM: Checksum errors reported in alerting logs for package updates

Solution1:  This is a warning and self correcting.  When package checksum values are not correct, the package is always resent.  This error is informational and does not need any user intervention to correct.

PROBLEM:  User can not connect to server with Commodity Charts or Query

Solution1: Network changes on User end  (DNS Name does not resolve)  hostname and dns name match

Solution2: Commodity DataServer server not running -- run server.info  and check if services are running

PROBLEM:  Load_Updates failed

Solution1:  Check Disk Space

Solution2:  Perform a Safe restart of Commodity DataServer Instance

Solution3:  Check for Zombie processes --  ps –ef | grep {limuser}- contact support for assistance

PROBLEM:  License Expired unexpectedly

Solution:  Changing hostname or IP address will cause the license to expire. (Verify that the hostname and IP address have not changed). Contact support for updated license. To verify currently installed License, run 'list_ids', to add a new License 'add_ids'; to request a new license run 'hostid' and share the output with Morningstar Support in order to have a new license generated.