MPmon v5.2
User Guide

an *Application Monitoring*
Utility

Table of Contents

1. Introduction

This user guide provides detailed information on the operation and administration of Mobius Partners' Application Monitoring utility, MPmon.

MPmon performs comprehensive analysis of many operational aspects of high priority applications and notifies administrators if exceptions occur. Deploying MPmon to each server where the application resides dramatically improves visibility into and the stability of your high priority applications. MPmon enables you to quickly identify and deal with many issues that would otherwise simmer for hours or days before you become aware of them.

While MPmon was originally developed to monitor Micro Focus Software applications, is used to monitor any application running on a supported platform. The configuration examples provided in this document demonstrate how MPmon is used to monitor the Micro Focus Software application of Operations Bridge Reporter (OBR). The provided configuration files can be modified as needed to create new configuration files for monitoring specific OBR configurations or any other application.

MPmon sends notifications using one or more of the following methods:

- SNMP Traps
- Operations Manager i-series (OMi) messaging
- Trouble Tickets
- Email
- Pager
- Detailed Web-Based exception reports

MPmon can be easily customized to launch API calls to other management applications such as Trouble Ticket systems and other related applications.

MPmon can operate as a standalone solution or be used to add powerful capabilities to your existing application and system management tools. For the administrators responsible for the operation and availability of applications, MPmon enhances problem identification and resolution by providing in-depth analysis of processes, connectivity, log files, data, and databases to quickly identify and address issues before end-users are impacted.

Many aspects of systems and applications can be monitored for exceptions using MPmon. The following items are typically monitored:

- Verify required daemons and processes are running.
- Verify process run time durations are not excessive (process possibly hung).
- Verify applications are listening on and accepting connections on specified ports.
- Identify filesystems that have exceeded the specified capacity thresholds.
- Examine log files to verify successful process completion.
- Search log files for errors, warnings, return codes, and other conditions.
- Analyze database operation for the following:
    - Database processes are running.
    - Database is accepting user connections.
    - Database size is within the specified threshold.
- Run SQLs to show output in the MPmon reports.

MPmon is easily extended to perform other types of analysis based on MPmon's "EXEC" and "EXEC_SQL" command directives, which is used to execute a script, program, or SQL and capture the results.

Using the appropriate default MPmon configuration file for OBR, the following analysis is performed:

1. Verifies application data processing and management processes have completed successfully and identifies any errors.
2. Examines the application Database (Vertica, Oracle, or Sybase) log file to identify errors and warnings associated with the operation of the database.
3. Verifies required application processes are running:
    - Database processes
    - Data Collection and Processing processes
    - Web Server processes
    - SAP Business Objects processes
    - OBR Timer (trendtimer)
4. Monitors OBR processes to verify process run time durations are not excessive (process possibly hung).
5. Verifies application services are listening on and accepting connections on specified ports.
6. Verifies the database is listening on and accepting connections on specified ports.
7. Identifies server filesystems that have exceeded the specified capacity threshold (typically 80%).
8. Examines the application database to verify:
    - Database processes are running.
    - Database is accepting user connections.
    - Pollers are inserting data into the database.
    - Collected data is rolled to Delta level.
    - Data is summarized to hourly and daily levels.
    - Database size is within the specified threshold.
    - Track the size of application database tables and segments.

MPmon normally performs this analysis once per hour. If exceptions are found, MPmon notifies application administrators using the method appropriate for your environment:

- SNMP Traps
- Operations Manager i-series messaging
- Trouble Tickets
- Email
- Pager

MPmon provides email and pager notification to application administrators and other recipients listed in the MPmon Email and Pager notification file (i.e. MPmon_Email_Pager.List).

Notification can be sent via SNMP trap, which is configured via the MPmon Trap configuration file (i.e. MP_TrapGen.Config) and the MPmon configuration file (i.e. MPmon_Default.Config) using the new "T" (Trap) logging option. This feature enables integration with Micro Focus Network Node Manager i-series (NNMi) or any other SNMP trap receptor.

Notification can be sent to Micro Focus Operations Manager i-series (OMi), if OMi is in use and an OM operations or performance agent is installed on the server where MPmon is running.

When MPmon's OMi integration or SNMP trap notification features are utilized, the trap receptor's event handling and automatic action capabilities can be used. This enables more sophisticated handling of MPmon events and automatic actions such as the generation of trouble tickets based on the receiver's interface to a Trouble Ticket system.

MPmon is also able to make API (Application Program Interface) calls to Trouble Ticket systems and other related applications. If this feature is enabled, an API call will be launched to generate a trouble ticket within the user's trouble ticket system when errors are detected. The resulting trouble ticket will be populated with the corresponding error from the application being monitored by MPmon.

MPmon also provides detailed status reports to administrators daily via email. These comprehensive status reports contain information about critical services, batch processes, data table size, and the latest timestamp in data tables. This complete operational health report provides application administrators comprehensive insight into the operational health of the application server. This enables the administrator to quickly assess the condition of the application and to proactively address any issues that are identified. These status reports can be configured to send at the conclusion of daily summarization processing and/or at the end of the day to provide a snapshot of the application health for the day.

2. Monitoring Applications with MPmon

MPmon should be run on all systems that are running any part of the application to be monitored. Throughout this document examples are used that refer to monitoring HP's OBR. Like many applications, an OBR server can play different roles in a distributed OBR deployment. MPmon can easily be configured on each server to monitor only the functionality residing on that server.

MPmon uses a configuration file that is well documented and logically organized to enable application administrators to easily configure and tune MPmon as needed. Each section can be disabled by adding comment characters, "#" (pound signs / hash marks), to the beginning of each line in that section. A section should be disabled if the system where MPmon is located does not provide the functionality monitored by that section. Sections can be temporarily disabled if enhancements or testing is being performed. For example, when installing MPmon to monitor only the OBR Remote Collector, only sections that monitor the OBR collection processes, ports, and file system capacity should be enabled. All other sections can be removed or disabled (commented out).

2.1. Process Verification

MPmon verifies that vital processes are running. When used to monitor OBR, MPmon verifies the OBR processes are running and provides a list of all OBR related processes using the Mobius Partners *psobr* command. The following processes are normally included:

- Collection and data processing commands
- Database Application processes
  - Vertica
  - Oracle
  - Sybase
- Web service processes
- SAP Business Object processes
- Any other process that need monitoring (optional)

If the "-e" (send an Email) and "-p" (send a Page) logging syntax options are used, an email report is sent to all the recipients in the *{MOBIUS_HOME}/etc/MPmon_Email_Pager.List* file. If the "-p" option is only used and errors are found, a pager message of "Check OBR Processes on <hostname>" is sent to text-based pagers, mobile phones, or similar devices. If the "-T" (send a Trap) option is used and errors are found, a trap is sent to the designated trap recipient, as defined in the *{MOBIUS_HOME}/etc/ MP_TrapGen.Config* file.

2.2. Check Port Status

MPmon verifies that applications are listening on and accepting connections on the specified ports. When used to monitor an application, the following ports are normally checked:

- Application web services (normally 8443)
- OBR application (normally on various ports for Unix and Windows)
- SAP application (normally 6400 and 6410 for Unix and Windows)
- Vertica database (5433 for Unix and Windows)
- Oracle database (normally 1521 for Unix or 2030 for Windows)
- Postgres database (21425 for Unix and Windows)

If the "-e" (send an Email) and "-p" (send a Page) logging syntax options are used, an email report is sent to all the recipients in the *{MOBIUS_HOME}/etc/MPmon_Email_Pager.List* file. If the "-p" option is only used and errors are found, a pager message of "Check OBR processing on <hostname>" is sent to text-based pagers, mobile phones, or similar devices. If the "-T" (send a Trap) option is used and errors are found, a trap is sent to the designated trap recipient, as defined in the *{MOBIUS_HOME}/etc/ MP_TrapGen.Config* file.

2.3. Process Runtime Verification

MPmon monitors processes to verify that process run time durations are not excessive. An excessively long run time may be indicative of a hung process.

When used to monitor OBR, the following processes are normally verified:

- Collections: data collect process threshold default is 60 minutes.
- Aggregations: stage, load, aggregate threshold default is 60 minutes.
- Aging: data db_delete_data threshold default is 120 minutes.

If the "-e" (send an Email) and "-p" (send a Page) logging syntax options are used, an email report is sent to all the recipients in the *{MOBIUS_HOME}/etc/MPmon_Email_Pager.List* file. If the "-p" option is only used and errors are found, a pager message of "Process '<process_name>'[pid:<pid>] 300 mins old, running since 2019/06/20 16:58 on <hostname>" is sent to text-based pagers, mobile phones, or similar devices. If the "-T" (send a Trap) option is used and errors are found, a trap is sent to the designated trap recipient, as defined in the *{MOBIUS_HOME}/etc/MP_TrapGen.Config* file.

2.4. File System Capacity

MPmon monitors filesystem space on UFS, VXFS (Veritas), and TMPFS filesystems to confirm each has an acceptable level of free space. The free space threshold is set to 80% by default and can be set to an appropriate value for each filesystem. Certain filesystem types, such as removable media and remote mounted directories are ignored by default.

MPmon sends a notification when a filesystem capacity exception is identified but does not send additional notifications for the same filesystem unless the percentage of capacity used increases further. Additional notifications will only be sent if MPmon detects the percentage of the filesystem capacity used has increased since the pervious notification was sent. This feature prevents excessive notifications to recipients. However, the daily email log will contain WARNING messages indicating the filesystems usage still exceeds the threshold value.

If the "-e" (send an Email) and "-p" (send a Page) logging syntax options are used, an email report is sent to all the recipients in the *{MOBIUS_HOME}/etc/MPmon_Email_Pager.List* file. If the "-p" option is only used and errors are found, a pager message of "Check Filesystem Size on <hostname>" is sent to text-based pagers, mobile phones, or similar devices. If the "-T" (send a Trap) option is used and errors are found, a trap is sent to the designated trap recipient, as defined in the *{MOBIUS_HOME}/etc/MP_TrapGen.Config* file.

2.5. Database Statistics

MPmon analyzes application operations by verifying the data within the application's database and analyzing other critical database components.

When used to monitor OBR, The following aspects of the database are verified:

- Database processes are running.
- Database is accepting user connections.
- Collectors are inserting data into the database based on timestamp of collected data.
- Collected data is rolled to rate, hourly, and daily levels.
- The size of database is within the specified threshold.
- Size of OBR database tables and segments.

The resulting report provides additional database table timestamp information for all monitored tables as well as size and row counts for all tables.

If the "-e" (send an Email) logging syntax option is used, an email report is sent to all the recipients in the *{MOBIUS_HOME} /etc/MPmon_Email_Pager.List* file.

If the "-p" (send a Page) option is used and the database is larger than 80%, a pager message of "Check OBR Database Size of nn% on <hostname>" is sent to text-based pagers, mobile phones, or similar devices.

If the “-p” option is used and latest timestamp (max(*ta_period))* for collected or summarized data on the OBR Server is older than NN minutes from the current date, a pager message of “Check Collectors/Rollups on <hostname>” is sent to all the recipients in the *{MOBIUS_HOME} /etc/MPmon_Email_Pager.List* file.

If the “-T” (send a Trap) option is used and errors are found, a trap is sent to the designated trap recipient, as defined in the *{MOBIUS_HOME}/etc/MP_TrapGen.Config* file.

Typically the threshold is set at 60 minutes for Delta tables, 300 minutes for Hourly (if processed hourly) and 3300 minutes for Daily tables.

2.6. Database Application Log File Errors/Warnings

MPmon configuration files enable MPmon to analyze the operation of database applications and notify administrators of problems. The MPmon configuration files and templates provided are preconfigured to monitor Vertica and Oracle. SQL databases from other vendors can easily be monitored by MPmon by making the appropriate MPmon configuration file changes using the existing configuration files as a guide.

If the “-e” (send an Email) and “-p” (send a Page) logging syntax options are used, an email report is sent to all the recipients in the *{MOBIUS_HOME}/etc/MPmon_Email_Pager.List.* If the “-p” option is used and errors are found, a pager message of “Check Database error of NNNN on <hostname>” is sent to text-based pagers, mobile phones, or similar devices.

2.6.2. Vertica Error Log File Errors/Warnings

MPmon monitors Vertica database operation by examining the error log file for "error" and "warning” messages.

2.6.1. Oracle Alert Log File Errors/Warnings

MPmon monitors the Oracle database operation by examining the alert log file for "ORA-" errors and warnings. Critical messages found in the Oracle log result in immediate notification of administrators. Less critical messages may only generate status report notifications intended to prompt the administrator to investigate as time permits. The notification method used for each error condition can be easily adjusted as needed.

Please review the Oracle Config file (*MPmon_Default.Config_[Unix|Windows]_v5.0_Oracle*) in the *{MOBIUS_HOME}/MPmon/etc/misc* directory for all monitored “ORA-“ messages.

2.7. Process Completion Status

MPmon monitors log files for errors and process completion messages. A typical example of monitoring process completion status would be confirming the OBR rollup processes have completed successfully.

Using the default MPmon configuration file for OBR, MPmon monitors processing by examining the OBR log file looking for the following:

- Data cleanup processing
- Data summarization processing

If Mobius's "*timeit*" utility is in use on the system, uncomment the CHECK_LOG section in the *MPmon_Default.Config* file which examines the *{MOBIUS_LOG}/Time.log* file for Return Code values. A Return (Exit) Code value of "0" is expected for a successful completion of processes initiated by "*timeit*" while unsuccessful completions terminate with a Return Code that is not "0". This step identifies processes that terminated with return codes other than "0".

By default, the logging option of "-L" (Log only for reporting) is used with summarization processing checks. If a rollup error is encountered (normally in the middle of the night), the error will be noted in the MPmon output, but no email or pager notification will be sent. The OBR administrator should carefully review the emailed morning status report for potential errors and take corrective action if necessary.

3. Using MPmon

MPmon is a low-overhead "command line" application for Unix and Windows applications. By design, MPmon uses minimal system resources. For example, a command line interface is used instead of a more resource intensive graphical user interface (GUI). As a result, MPmon has little or no performance impact on the server while monitoring an application. MPmon can be installed on production servers without concern of the impact on other applications.

Mobius uses Cygwin by Red Hat, Inc. for Windows to provide a Unix shell environment on Windows servers. Like MPmon, Cygwin requires very minimal system resources. Please refer to the MPmon Release Notes for more detailed information on Cygwin.

3.1. MPmon Usage and Command Line Options

The options for MPmon at the command line are:

mpmon [**-i**] [**-o**] [**-e** | **-ef**] [**-p** | **-pf**] [**-T**] [**-html**] [**-h** | **-?**] [**-V**] [**-d** | **-dd**]

-i **<Input_Configuration_Filename>**
Pathname to an alternate configuration file. Default Config file is *{MOBIUS_HOME}/etc/MPmon_Default.Config*

-o **<Output_Report_Filename>**
Pathname to an alternate output report file. Default report log file is *{MOBIUS_HOME}/Log/MPmon.log*

-e Send Email of the Log and print to stdout (standard output) even if no error is found. Default Email file is *{MOBIUS_HOME}/etc/MPmon_Email_Pager.List*

-ef **<Email_Pager_List_Filename>**
Send Email and/or Pages using Filename of alternate list of recipients. Use this option if the work environment has multiple shifts of Administrators responsible for application monitoring (i.e. 24-hours/day operation with 3 working shifts).

-html Creates an HTML report even if no error is found. The HTML reports are located in *{MOBIUS_HOME}/reports/MPmon*.

-p Only send Pages and Email if errors are found, but still print to stdout. Default Pager file is *{MOBIUS_HOME}/etc/MPmon_Email_Pager.List*

-pf **<Email_Pager_List_Filename>**
Same as the "**-ef**" option. Send Pages using Filename of alternate list of recipients.

-T Send Traps to the designated trap receptor if errors are found and also send an email. Trap Generation configuration file is *{MOBIUS_HOME}/etc/MP_TrapGen.Config.*

-h / **-?** Display this help page

-V / **-v** Display the Version and Build of MPmon

-d / **-dd** Display debug or detailed debug information on stdout

MPmon is designed to run on a scheduled basis as often as necessary. If possible, it should be run when database activity is relatively low. For example, a good time to run MPmon is before a collection cycle is to start to simplify the report output (i.e. 59 minutes after the hour).

Note: Using **MPmon** with or without options displays output on the user's screen (stdout). The user can then scroll through the output looking for errors, which are clearly marked.

mpmon -e displays output on the user's screen and also emails all recipients in the MPmon Email_Pager notification file, even if no errors are found. The user can then scroll through the email looking for clearly marked errors.

mpmon -p displays output on the user's screen and only if exceptions are found, pages all recipients in the MPmon Email_Pager notification file with a notice to research the exception. Email is also sent to all recipients in the MPmon notification file. The email sent includes the complete MPmon report and a subject field that states that an error has occurred.

mpmon -html displays output on the user's screen and creates an HTML report even if no exceptions are found. The HTML reports are located by default in *{MOBIUS_HOME}/reports/MPmon*. The "-html" option also launches a Perl script of "MP*create_html_index.*pl" which updates an "*index.html*" file in the specified HTML directory and all sub-directories. The Perl script also appends a Virtual-Mapping entry into the OBR web service in order to easily access the MPmon reports. At any web browser use the following URL:

http://<OBR_Server>/MPreports_Admin/index.html

mpmon -T displays output on the user's screen and only if exceptions are found, sends traps to the designated trap receptors. The default MPmon trap configuration file is *{MOBIUS_HOME}/etc/MP_TrapGen.Config*. This file should be modified at time of installation and can contain more than one trap destination server. The following Trap Varbinds are defined for Trap Generation:

1.3.6.1.4.1.11.2.17.14.0.2.1 - hostname
1.3.6.1.4.1.11.2.17.14.0.2.2 - application
1.3.6.1.4.1.11.2.17.14.0.2.3 - message
1.3.6.1.4.1.11.2.17.14.0.2.4 - timestamp
1.3.6.1.4.1.11.2.17.14.0.2.5 - severity (currently only set to Unknown)
1.3.6.1.4.1.11.2.17.14.0.2.6 - "Detailed reports at:"
1.3.6.1.4.1.11.2.17.14.0.2.7 - URL to MPmon reports avialble on the server
1.3.6.1.4.1.11.2.17.14.0.2.8 - blank
1.3.6.1.4.1.11.2.17.14.0.2.9 - blank
1.3.6.1.4.1.11.2.17.14.0.2.10 - blank
1.3.6.1.4.1.11.2.17.14.0.2.11 - blank
1.3.6.1.4.1.11.2.17.14.0.2.12 - blank
1.3.6.1.4.1.11.2.17.14.0.2.13 - blank
1.3.6.1.4.1.11.2.17.14.0.2.14 - blank
1.3.6.1.4.1.11.2.17.14.0.2.15 - blank

3.2. Automating MPmon

The default installation schedules MPmon to run at the following times:

- At 59 minutes after every hour (**mpmon -p**).
- At 7:29am to capture the status of OBR after morning processing (**mpmon -e**).
- At 11:29pm to capture the status of OBR at the end of the day (**mpmon -e**).

Scheduled initiation of MPmon is performed by CRON, which is a standard process scheduling tool for Unix and Cygwin. Mobius chose CRON instead of OBR's trendtimer process or the Windows scheduler because MPmon should not be initiated by or be dependent on the application that MPmon is monitoring. Launching MPmon from trendtimer would violate that guideline.

The CRON schedule (called Crontab) for MPmon can be edited by typing in the following at a Unix or Cygwin command prompt:

crontab -e (**crontab -l** just lists the schedule)

NOTE: For Unix, you may need to set your default editor to "**vi**" by executing at the command line "**export EDITOR=vi**" for KSH/Bash or "**setenv EDITOR vi**" for CSH.

NOTE: For Windows, you will need to access crontab via the **mpEditCron.sh** script from a Cygwin window. This script will run "crontab -e" with the proper permissions.

The default Crontab entries for MPmon are listed below. Notice that all entries are commented out (disabled). After MPmon configuration is complete and you are ready to use MPmon in production, uncomment the entries in the Crontab so MPmon will be initiated every hour and twice a day. Some knowledge of the "**vi**" editor is required. Please refer to the MPmon Install Guide for instructions.

NOTE: For Unix, if the file of "*cron.allow*" exists in either */etc/cron.d*, */var/cron*, or */usr/lib/cron*, login as 'root' and edit the file and add a line at the bottom containing the user admin name of the application. This is required to allow the user to use the CRON utility.

```
# Run MPmon every hour (page) and every day (email) after rollups (7:54am)
# or at the end of the day (11:54pm) looking for potential OV issues.
#59 * * * * {MOBIUS_HOME}/MPmon/bin/MPmon.sh -p > /dev/null 2>&1
# Creates a daily MPmon report after the daily processing
#29 7 * * * {MOBIUS_HOME}/MPmon/bin/MPmon.sh -e > /dev/null 2>&1
# Creates a daily MPmon report for the entire day
#29 23 * * * {MOBIUS_HOME}/MPmon/bin/MPmon.sh -e > /dev/null 2>&1
# Creates a MPmon report for OM monitoring
#15 * * * * /opt/Mobius/MPmon/bin/MPmon.sh -i /opt/Mobius/etc/
MPmon_OM.Config -o /opt/Mobius/Log/MPmon_OM.log > /dev/null 2>&1
# Archive and compress the MPmon daily log file
#0 0 * * * /opt/Mobius/bin/MParchivelog.sh /opt/Mobius/Log/
MPmon_Archive.log
# Archive and compress the Time.log log file
#0 0 * * * /opt/Mobius/bin/MParchivelog.sh /opt/Mobius/Log/Time.log
```

3.3. Output Distribution

When MPmon detects exceptions, small notification messages are sent to defined recipients (pagers, mobile phone, Trap Receptor, OM). These small messages do not contain complete MPmon output detail due to the limitations imposed by the receiving device or application.

Complete MPmon output details are stored on the server where MPmon resides. If the "-html" option is used, Administrators will have access to comprehensive MPmon reports using a standard browser, as long as connectivity to the server is provided. The Administrator will not only have access to the most recent MPmon reports, but will also be able to access MPmon reports for the previous several days.

Ensure that Administrators can quickly resolve problems detected by MPmon by providing them with rapid browser access to these comprehensive MPmon reports.

To ensure that Administrators can always quickly access these MPmon reports, traps sent by MPmon contain the URL to the MPmon HTML reports.

Using this feature requires that a web server be used on the system and that MPmon be configured to place HTML reports into the appropriate location under that web server. When using MPmon with OBR, the OBR web server is used to provide access to MPmon reports and authenticate users attempting to access those reports.

MPmon output is displayed via stdout and recorded to a default log file when run from the command line. The default log file location and name is:

{MOBIUS_HOME}/Log/MPmon.log

An alternate log file can be used by specifying the file name using the "-o" option as in:

mpmon -o {MOBIUS_HOME}/Log/MPmon_OMi.log

MPmon is normally executed via CRON by entries in the *shrboadmin/vertica* user's crontab file. If the "-p" option is used and an error is discovered, a short email message is sent to a text pager or cell phone pager (i.e.: <phone_number>@<service_provider>.com). One or more of the following strings are included in the message:

From: <Company Name/Initials>

Subject: OBR Error on <hostname>

Message: Check Process <process_name>, not running

No service running on port <hostname>:<port_num>

Process '<process_name>' [pid:<pid>] NN mins old, running since <timestamp> on <hostname>

Check Filesystem of <fs>, has grown to <NN%>, the threshold value is <YY%>

The database table <table_name> is not found

The database table <table_name> exists but contains no data

Data in <table_name> is NN minutes old, MAX ta_period is <time_stamp>

Check Database of PMDB, has grown to NN% above the threshold value of YY%

Found 1 trend_sum in OBR.log

3.4. Error Logging

MPmon writes errors from STDOUT to a separate log file of {MOBIUS_HOME}/Log/ *MPmon_Error.log*. This log file is concatenated to the end of the screen display if interactively running MPmon or right after the title banner as "Summary or Error in the MPmon report that is created for emailing and HTML viewing.

If you are not sure that MPmon is operating correctly investigate by viewing the *MPmon_Error.log* file after running "**mpmon -d**" for debug output or "**mpmon -dd**" for detailed debug output.

MPmon will produce standard non-zero Return Codes if MPmon terminates with an error condition:

100 – if there is an issue with a syntax argument when launching MPmon

200 – if there is an issue with the MPmon License file

300 – there is an issue with the MPmon configuration file

400 – if there is an issue with the MPmon Contacts file (i.e. {MOBIUS_HOME}/etc/ MPmon_Email_Pager.List)

500 – if there is an issue with the MPmon launch command of "mpmon.sh"

3.5. Archive Logging

The *MPmon.log* is overwritten each time MPmon executes. When **MPmon** execution has completed, the newly created *MPmon.log* file is concatenated to the end of the *MPmon_Archive.log* file for historical tracking.

The **MParchivelog.sh** process will make a daily backup and compress the archive file with a naming convention of *MPmon_Archive.log.<day_of_week>*. Each day, **MParchivelog.sh** replaces the archived log file for the same day of the previous week, with the new log file.

MParchivelog.sh is called from CRON on both Unix and Windows servers.

If desired, the **MParchivelog.sh** program can be modified with the **vi** editor to turn on a compression utility, possibly "**gzip**" or "**compress**".

3.6. MPmon Support Files

This section provides information on configuration files required by MPmon. The files are located in the *{MOBIUS_HOME}/etc* directory.

1. **MPmon_Default.Config** MPmon's default Configuration file used every hour. During installation, this file must be modified as needed based on the application to be monitored by MPmon. See the MPmon Configuration File in Chapter 4. The {*MOBIUS_HOME}/MPmon/etc/misc* directory contains several variations of this configuration file for different applications that might have been pre-configured in the past. Any of these files can be copied into place as *MPmon_Default.Config* or can be used with minimal changes with the **mpmon -i** option. Simply update the header information in the file as needed.

2. **MPmon_FullCheck.Config** A copy of the MPmon default Configuration file used for the twice daily checks, but is configured to include extra checking that is not desired during the hourly checks. These extra checks can include using the CHECK_DB section, extra log checking, and/or running extra scripts or SQLs via the EXEC and EXEC_SQL command directives like MPcheckpolling.sh. See the MPmon Configuration File in Chapter 4. Run MPmon via "*mpmon -i MPmon_FullCheck.Config*". When creating this file using the *MPmon_Default.Config* file as a template, first finish making any modifications to the default configuration before making the copy.

3. **MPmon_Email_Pager.List** This file contains the list of recipients who should receive email and pager notifications from MPmon when initiated with “-e“ and “-p“ options. The email addresses and pager numbers of the Applications Administrators and other persons responsible for maintaining the monitored application should be included in this file.

4. **MP_TrapGen.Config** This file contains the list of trap receptors (hostname and SNMP port number) one per line.

5. **MPmon_OMi.Config** This configuration file is used when MPmon is configured to send alert notifications to Operations Manager i-series. This configuration file for OMi includes CHECK_PORT and CHECK_TIME sections. This slimmed down configuration file enables MPmon to execute these checks more frequently than the typical hourly interval. When MPmon is configured to send alert notifications to OMi, uncomment the line in Crontab that initiates the following process every 15-minutes:

   mpmon -i {MOBIUS_HOME}/etc/MPmon_OMi.Config -o {MOBIUS_HOME}/Log/MPmon_OM.log

Below is a list of support files required by MPmon. These files are located in the *{MOBIUS_HOME}/MPmon/...* directories.

6. **bin/mpmon.sh** Startup program for MPmon. During installation, this file may need to be edited to correctly set environment variables if non-default directory paths are in use. See Installation Instructions in Chapter 5.

7. **bin/show_[Vertica|Oracle|Sybase]Log.sh** Displays any messages in the Oracle alert log file or the Vertica error log file with today's date.

8. **bin/MParchivelog.sh** Archives and compresses (if compression is enabled after install) the MPmon_Archive.log file daily at midnight by changing the log file to include the day of week at the end of the filename. Daily archived log files are overwritten every seven days. **MParchivelog.sh** is initiated by a crontab process.

9. **bin/mpchgdbpass.pl** A utility to change the encrypted database password by displaying a new version of the MP_DATABASE_INFO line. Then copy/paste the updated version into an of the MPmon Config files that are used to access a database.
   Usage is:
   /opt/Mobius/bin/mpchgdbpass.pl -help
   /opt/Mobius/bin/mpchgdbpass.pl -c /opt/Mobius/etc/MPmon_Default.Config -u <db_user>

   You will be prompted for the current database password, twice. Then use the output from mpchgdbpass.pl to update the MP_DATABASE_INFO line in any of the MPmon Config files that will be accessing the database.

10. **etc/misc** Miscellaneous alternate MPmon configuration files and OMi integration files. MPmon configuration files differ slightly between various applications to monitor. Such as if MPmon is monitoring applications using; the Vertica, Oracle, or Sybase database, Remote Collectors, OBR using a remote database server, or applications using a remote Oracle database server.

11. **../Docs/*** MPmon documentation (User's Guide, Release Notes, MPmon Install Guides (READMEs), and Cygwin Install Guide (Windows Only).

12. **tmp/*** MPmon temporary files used to track filesystem and database usage.
    Do NOT touch or modify!

13. **var/*** MPmon log files used for troubleshooting.
    Do NOT touch or modify these files!

4. Configuring MPmon

MPmon can be configured to monitor any application by modifying the MPmon default configuration file, adding email and page recipients to the MPmon Email Pager file, and optionally defining SNMP Trap receptors in the Trap Generation configuration file. You may also setup MPmon to make API calls to access a trouble ticket system or other management application that you would like to integrate with MPmon.

4.1. MPmon Configuration File

The MPmon Configuration file is divided into sections that instruct MPmon to analyze different aspects of the application being monitored. Each section can be removed or commented out as needed. Each section can also be modified as needed to include/exclude any part of the application being monitored.

The configuration file has 3 columns for ease of reading and modification.

```
#=============================================================================#
# Variable Name  =  Value of Variable         # Comment                      #
#=============================================================================#
```

The Variable Name column defines the section that MPmon will process. The possible options are:

Variable Name	Definition
MP_*	Defines the various MPmon startup variables (i.e. log path, location of default configuration file, Email_Pager file, …).
HEADER	Prints a header using the supplied text in a banner as shown above. (#=====…)
EXEC	Executes an O/S command or user created script from outside of MPmon and displays the results to stdout and the log file.
EXEC_SQL	Executes an SQL script (SELECT statement only) and displays the results to stdout and the log file.
CHECK_DB	Checks the database to determine that it is available, return the size and row count of database tables, and verify that space utilization is below the specified threshold value.
CHECK_FS	Checks the O/S to determine if the specified filesystem is available and if space utilization is below the specified threshold value.
CHECK_LOG	Checks the specified log file using the specified “search_string”, time stamp, and logging option (see below).
CHECK_PORT	Checks the O/S to determine if the specified ports (network sockets) are accepting connections.
CHECK_PROCESS	Checks the O/S to determine if the specified process is running.
CHECK_TABLE	Checks the database to determine if the specified table is available and verify that the latest timestamp is later than the specified number of minutes.
CHECK_TIME	Analyze O/S processes to determine if the specified process has been running longer than specified number of minutes.

The order of the sections in the configuration file determines the order of the output in the MPmon report log file.

The default configuration file is *{MOBIUS_HOME}/etc/MPmon_Default.Config* and will vary in content between Unix and Windows and of course between various applications being monitored.

Available Logging Options:

All "CHECK_XXXXXX" and "EXEC" commands have logging options as the last configurable parameter. This logging option tells MPmon where to direct the output. By default, the output goes to stdout (Standard Output, the screen) and the log file.

Available logging options:

P = Send a short SMS/Page to a text-based pager, mobile phone, or similar device *only* if MPmon detected an error condition had occurred. Also if an error had been detected, send an email containing the full MPmon log file to the recipients in the *{MOBIUS_HOME}/etc/ MPmon_Email_Pager.List* file in order to show all available details of the error condition.

E = Send an Email of the full log file upon completion of MPmon. This feature enables status reports that are useful for reviewing the application and server status after daily processing and at the end of the day.

B = Send both an Email regardless of errors and a Page upon finding an error.

T = Send a Trap to the trap receptors defined in the *{MOBIUS_HOME}/ etc/MP_TrapGen.Config* file if MPmon detects an error condition has occurred.

L = Log only the output to the log file and ignore any potential errors.

Examples:

```
...,ORA-01578,,yyyy/MM/dd,P
..., stage,,y-MM-dd,E
..., loader,ERROR,,y-MM-dd HH:,B
..., delete_data,ERROR,y-MM-dd HH:,T
EXEC = /bin/uptime,L
```

4.1.1. MPmon Startup Variables

Sample section:

```
#============================================================================#
# Define MPmon Startup Variables
#
MP_COMPANY     = <Company_Name>.{hostname} # Unique/short company name/initials
MP_SENDER      = <Sender_Email>            # From email address
MP_DIR         = {MOBIUS_HOME}/MPmon       # Path of MPmon install dir
MP_CONTACTFILE = {MOBIUS_HOME}/etc/MPmon_Email_Pager.List  # Contacts Info
MP_MAIL_SERVER = mail.<Mail_Server>.com    # Name of the email server
#MP_DATABASE_INFO = {HOSTNAME},<port>,<username>,<password>,{DBname/SID} # DB Info
```

Definition of Variables:

MP_COMPANY = <Company_Name/Initials>.{hostname}
Unique and short Company Name or Initials for the Email "FROM" Address (i.e. MP.hpobr100-a1).

MP_SENDER = <from_Sender_email_address>
Valid email address that MPmon will use as the sender of this email for companies that authenticate the user before sending the email.
(i.e. MPmon_Support@Mobius.com).

MP_DIR = <full_MPmon_install_pathname>
Full pathname of MPmon install directory. Also {MOBIUS_HOME}
(i.e. /opt/Mobius or D:\Mobius).

MP_CONTACTFILE = <full_MPmon_Email_Pager_pathname>
Full pathname to the *MPmon_Email_Pager.List* file that contains all email addresses and pager addresses of the OBR Administrators.

MP_MAIL_SERVER = <default_mail_server>
Default company POP3 mail server, (i.e. mail.<Company_Name>.com)

MP_DATABASE_INFO = [<hostname>|<ip address>],<port_number>,
<db_username>,<db_password>,[{DB_Name|Oracle_SID}]

For Vertica: {HOSTNAME}, Port **5433**,User **pmdb_admin**, password, and "PMDB"
For Oracle: {HOSTNAME}, Port **1521**,User **dsi_dpipe**, password, and {ORACLE_SID}
For Sybase: {DSQUERY}, Port **2052** (Unix) or **5000** (Windows), User **dsi_dpipe**, and password

NOTE: Password is encrypted via the *MPchgdbpass.pl* utility.

Use the following to view the command syntax of "$MP_HOME/bin/mpchgdbpass.pl –h".

4.1.2. Checking Processes

Sample section:

```
#================================================================================#
# Check if the following processes are running
#
HEADER = Executing PSOBR . . . . . .
EXEC   = sh -f {DPIPE_HOME}/bin/psobr,L       # Run psobr command
HEADER = Checking for necessary OBR & database processes . . . . . .
EXEC   = /bin/uptime,L                        # Get system uptime
CHECK_PROCESS = trendtimer,B                  # OBR Timer process
CHECK_PROCESS = postgres,B                    # Postgres database processes
CHECK_PROCESS = adminServer,B                 # OBR Admin Console service process

CHECK_PROCESS = vertica,B                     # Vertica process(es)
CHECK_PROCESS = v_pmdb_node0,B                # Vertica database processes
or
CHECK_PROCESS = ora_smon_{HOSTNAME},B         # Oracle process(es)
CHECK_PROCESS = ora_pmon_{HOSTNAME},B         # Oracle process(es)
CHECK_PROCESS = ora_dbw0_{HOSTNAME},B         # Oracle process(es)
CHECK_PROCESS = ora_lgwr_{HOSTNAME},B         # Oracle process(es)
CHECK_PROCESS = ora_ckpt_{HOSTNAME},B         # Oracle process(es)
CHECK_PROCESS = tnslsnr,B                     # Oracle process(es)
or
CHECK_PROCESS = dataserver,B                  # Sybase process(es)
CHECK_PROCESS = RUN_{DSQUERY},B               # Sybase process
#CHECK_PROCESS = RUN_SYB_BACKUP,B             # Optional Sybase Backup Server
```

Definition of Variables:

EXEC = <command>,<logging_option>,<optional_timeout>

Execute an O/S command or user created script as if typed from command line. See above for logging options.

Default <optional_timeout> is 5-minutes.

CHECK_PROCESS = <OS_process_name>,<logging_option>>,<optional_timeout>

Determine if specified process is running.

The Optional Timeout is in seconds and the default timeout is 300 seconds (5-minutes).

4.1.3. Checking for Hung Processes

Sample section:

```
#=============================================================================#
# Check if the following Processes are taking too long to run
#
HEADER = Checking for hung processes . . . . . . . . . . . . . . . . . . . . .
CHECK_TIME = collect,60,B                  # Check OBR data collection
CHECK_TIME = stage,60,B                    # Check OBR data staging
CHECK_TIME = loader,60,B                   # Check OBR data loading
CHECK_TIME = aggregate,60,B                # Check OBR aggregation of the data
CHECK_TIME = db_delete_data,120,B          # Check OBR data aging
CHECK_TIME = mpmon,60,B                    # Check MPmon application monitoring
```

Definition of Variables:

CHECK_TIME = <process>:<minute_threshold>,<logging_option>,<optional_timeout>

This directive should be used to determine if core processes, possibly collection and summarization related, are running longer than expected and are possibly hung. In Unix and Cygwin, this can also be verified by typing: "***psobr***" or "***ps -ef***" for Unix and "***ps -efW***" for Cygwin.

The Optional Timeout is in seconds and the default timeout is 300 seconds (5-minutes).

4.1.4. Checking Ports

Sample section:

```
#============================================================================#
# Check if the following ports (network sockets) are available
#
HEADER = Checking for available network sockets (ports). . . . . .

CHECK_PORT = {HOSTNAME}:5433,B              # Linux Vertica Server
CHECK_PORT = {HOSTNAME}:8443,B              # HTTPS access to OBR BO Consoles
CHECK_PORT = {HOSTNAME}:21412,B             # HTTPS access to OBR Admin Console
or
CHECK_PORT = {HOSTNAME}:1521,B              # Unix and Windows Oracle Server
CHECK_PORT = {HOSTNAME}:2030,B              # Also Windows Oracle Server
or
CHECK_PORT = {HOSTNAME}:2052,B              # Unix Sybase Server
#CHECK_PORT = {HOSTNAME}:4200,B             # Optional Sybase SYB_BACKUP Server

CHECK_PORT = {HOSTNAME}:80,B                # HTTP access to Application Server
#CHECK_PORT = {HOSTNAME}:443,B              # Optional SSL to Application Server
```

Definition of Variables:

CHECK_PORT = <hostname>:<port_number>,<logging_option>,<optional_timeout>

Determine if specified application port is accepting connections. In Unix and Cygwin, these ports can also be verified by typing:
netstat -af inet -n | grep <*port_number*> -for Unix
netstat -a -n | grep <*port_number*> -for Cygwin

The Optional Timeout is in seconds and the default timeout is 300 seconds (5-minutes).

NOTE: The port being monitored does not need to be just the localhost. Ports on remote hosts (servers) can also be monitored: Simply change the {HOSTNAME} to be the hostname of a remote server, provided that the remote server is reachable (via ping) from the MPmon server.

i.e.:

- the port on a remote Database Server
- the port on a remote server being used to send an error detected by MPmon in order to generate a trouble ticket

4.1.5. Checking Filesystems

Sample section:

```
#================================================================================#
# Check filesystems for available space
#
HEADER = Checking OBR related File Systems for available space . . . . . .
EXEC   = df -k,L                            # Run df -k to list disk free space
CHECK_FS = /,80,B                           # Check if filesystem > NN%
CHECK_FS = /opt/HP,80,B                     # Check if filesystem > NN%
CHECK_FS = /tmp,80,B                        # Check if filesystem > NN%
```

Definition of Variables:

CHECK_FS = <drive:|mount_point>,<%_threshold>,<logging_option>,<optional_timeout>

Determine if specified filesystem has exceeded the specified capacity threshold

The Optional Timeout is in seconds and the default timeout is 300 seconds (5-minutes).

4.1.6. Checking Database Tables

Sample section:

```
#=============================================================================#
# Check the following data tables that ta_period is not older than <NN> minutes
#
HEADER = Checking OBR Data Tables and max ta_periods . . . . . .
# OBR System Management Content Pack
CHECK_TABLE = SR_SM_CPU,300,ta_period,B                 # System Resource CP
CHECK_TABLE = SR_SM_DISK,300,ta_period,B                # System Resource CP
CHECK_TABLE = SR_SM_NODE_RES,300,ta_period,B            # System Resource CP
CHECK_TABLE = SH_SM_CPU,300,ta_period,B                 # System Resource CP
CHECK_TABLE = SH_SM_DISK,300,ta_period,B                # System Resource CP
CHECK_TABLE = SH_SM_NODE_RES,300,ta_period,B            # System Resource CP
CHECK_TABLE = SD_SM_CPU,3300,ta_period,B                # System Resource CP
CHECK_TABLE = SD_SM_DISK,3300,ta_period,B               # System Resource CP
CHECK_TABLE = SD_SM_NODE_RES,3300,ta_period,B           # System Resource CP

# OBR Virtualization Environment Content Pack
CHECK_TABLE = SR_VI_VM,60,ta_period,B                   # Virtualization Env CP
CHECK_TABLE = SH_VI_VM,300,ta_period,B                  # Virtualization Env CP
CHECK_TABLE = SD_VI_VM,3300,ta_period,B                 # Virtualization Env CP
...
```

Definition of Variables:

CHECK_TABLE = <data_tablename>,<minute_threshold>,<timestamp_column>,
<logging_option>,<optional_timeout>

Check the specified data table to verify that the max timestamp (ta_period) is not older than the specified threshold value. Typically the threshold is 60 minutes for Delta tables, 300 minutes for Hourly (if rolled hourly), and 3300 minutes for Daily tables.

The Optional Timeout is in seconds and the default timeout is 300 seconds (5-minutes).

NOTE: The currently supported databases are; Vertica, Oracle, Sybase IQ, and Sybase ASE.

4.1.7. Checking Database Size

Sample section:

```
#==============================================================================#
# Check the Database for available space
#
HEADER = Checking OBR Database and Transaction Log space . . . . . .
CHECK_DB = PMDB,80,B                        # Check if OBR database/log > NN%
```

Definition of Variables:

CHECK_DB = <db_name>,<%_threshold>,<logging_option>,<optional_timeout>

Check specified database for available space and email and page if it exceeds the specified threshold value of <%_threshold>.

Default <optional_timeout> is in seconds and the default value is 30 minutes.

Displayed Output:

The result of CHECK_DB displays the following:

- all OBR data tables listed along with the Number of Rows and Kbytes per table
- table names sorted by size in Rows, then Kbytes
- if Oracle, summary of tablespace size and percent available
- overall summary of database sized used and percent available
    - if Oracle, a summary per tablespace with an overall summary of all tablespaces at the end of CHECK_DB
    - if Sybase, a total summary of space used in the databases of “dpipe_db” and “tempdb” at the end of CHECK_DB

NOTE: If Oracle, the “Percent Used” is computed from the “Max Size MB” possible with the space available within each tablespace. Because the “Max Size MB” can be the entire size of the partition size available when “Auto Extensible” is turned on, the “Percent Used” may appear to be quite small.

```
CHECK_DB: Tablespace DPIPE_RATE_IND_SEG is OK at 0% used. Threshold: 80.0%
CHECK_DB:
CHECK_DB:       Filename            = C:\OBR\DPIPE_RATE_IND_SEG_001.DBF
CHECK_DB:       Size MB             = 492 MB
CHECK_DB:       Used MB             = 12 MB
CHECK_DB:       Avail MB            = 480 MB
CHECK_DB:       Percent Used        = 0 %
CHECK_DB:       Max Size MB         = 32767.98 MB
CHECK_DB:       Autoextensible      = YES
CHECK_DB:       Extend Increment MB = 246 MB
```

NOTE: The currently supported databases are; Oracle, Sybase IQ, and Sybase ASE.

4.1.8. Checking Log Files

Sample section:

```
#=============================================================================#
# Check Vertica log file for the following strings
#
#HEADER = Checking Vertica log for errors & warnings . . . . . .
#CHECK_LOG = /opt/vertica/log/agent_vertica.err,WARNING,,yyyy/MM/dd,E # Show WARNINGs

#=============================================================================#
# Check HPA Collector log file for the following strings and errors
#
#HEADER = Checking hpacollector.log file. . . . . .
#CHECK_LOG = {APP_LOG}/hpacollector.log,ERROR,,y-MM-dd,L     # Check for daily errors
#CHECK_LOG = {APP_LOG}/hpacollector.log,ERROR,,y-MM-dd HH:,E # Check for hourly errors

#=============================================================================#
# Check DB Collector log file for the following strings and errors
#
#HEADER = Checking dbcollector.log file. . . . . .
#CHECK_LOG = {APP_LOG}/dbcollector.log,ERROR,,y-MM-dd,L     # Check for daily errors
#CHECK_LOG = {APP_LOG}/dbcollector.log,ERROR,,y-MM-dd HH:,E # Check for hourly errors

#=============================================================================#
# Check Loader log file for the following strings and errors
#
#HEADER = Checking hpacollector.log file. . . . . .
#CHECK_LOG = {APP_LOG}/loader.log,ERROR,,y-MM-dd,L     # Check for daily errors
#CHECK_LOG = {APP_LOG}/loader.log,ERROR,,y-MM-dd HH:,E # Check for hourly errors

#=============================================================================#
# Check Stage log file for the following strings and errors
#
#HEADER = Checking stage.log file. . . . . .
#CHECK_LOG = {APP_LOG}/stage.log,ERROR,,y-MM-dd,L      # Check for daily errors
#CHECK_LOG = {APP_LOG}/stage.log,ERROR,,y-MM-dd HH:,E  # Check for hourly errors
```

Definition of Variables:

HEADER = <header_text_of_new_section>

Print header line with following text

CHECK_LOG = <logfile_full_path>,<1st search_string>,<2nd search_string >,<time_format>,
<logging_option>,<optional_timeout>

Check a log file for a given search string.

i.e. <Pathname_of_Log_File> - Full pathname of the log file to monitor
<1st search string> - Used for initial search, case insensitive
<2nd search string> - Used for initial search, case insensitive
<time_format> - Timestamp for log search, see formats below, also any string can be used instead of a time format
<logging_option> - P=page, E=email, B=both, T=trap, L=log only
<optional_timeout> - Default is 300 seconds (5 minutes)

Explanation of Search Strings:

MPmon uses standard RegEx (Regular Expression patterns) syntax to search log files. Case in point, when searching for the string of “stage,error,y-MM-dd”, MPmon parses through the log file looking for:

1. any occurrences that match the supplied time stamp of “y-MM-dd”
   Any string can be searched regardless if it is a time format or not.
2. search the extracted output looking for the string of “trend_sum”
3. then search for any potential “error” (or “warning”) conditions

The only deviation to this standard is searching for keywords that also do NOT contain extra values. Such as:

```
…,find "Error" not [ 1131| 1142| 1608| Continue processing],,yyyy/MM/dd HH:,…
```

The above entry will search the log file, first looking for any lines containing the time stamp down to the hourly level, then any lines containing the keyword “Error”, but NOT containing keywords of “ 1131”, “ 1142”, “ 1606”, and the string “ Continue processing”. Any keyword(s) within the syntax of “not [keyword1|keyword2| keyword3]” including leading or trailing spaces, separated by “ | “ (pipe signs) will be filtered out of the search.

If wishing to define a string search to be very definitive, such as only filtering on the string of “,ERROR” instead of also including “,DEF_ERROR”, change the CHECK_LOG search string from:

```
CHECK_LOG = {TREND_LOG}/trend.log,loader,ERROR,y-MM-dd HH:,E
```

to:

```
CHECK_LOG = {TREND_LOG}/trend.log,delete_db,\,ERROR,y-MM-dd HH:,E
```

4.1.9. MPmon Time Format Syntax

To specify the time format, use a time pattern string. In this pattern, all ASCII letters are reserved as pattern letters, which are defined as the following:

Symbol	Meaning	Presentation	Example
G	era designator	(Text)	AD
y	year	(Number)	2019
M	month in year	(Text & Number)	June & 06
d	day in month	(Number)	10
h	hour in am/pm (1~12)	(Number)	12
H	hour in day (0~23)	(Number)	0
m	minute in hour	(Number)	30
s	second in minute	(Number)	55
S	millisecond	(Number)	978
E	day in week	(Text)	Monday
D	day in year	(Number)	189
F	day of week in month	(Number)	2 (2nd Wed in July)
w	week in year	(Number)	27
W	week in month	(Number)	2
a	am/pm marker	(Text)	PM
k	hour in day (1~24)	(Number)	24
K	hour in am/pm (0~11)	(Number)	0
z	time zone	(Text)	Pacific Standard Time
'	escape for text	(Delimiter)	
''	single quote	(Literal)	'

The count of pattern letters determines the format.

(Text): 4 or more pattern letters--use full form, less than 4--use short or abbreviated form if one exists.

(Number): the minimum number of digits. Shorter numbers are zero-padded to this amount. Year is handled special; that is, if the count of 'y' is 2, the Year will be truncated to 2 digits.

(Text & Number): 3 or over, use text, otherwise use number.

Any characters in the pattern that are not in the ranges of ['a'..'z'] and ['A'..'Z'] will be treated as quoted text. For instance, characters like ':', '.', ' ', '#' and '@' will appear in the resulting time text even they are not embraced within single quotes.

A pattern containing any invalid pattern letter will result in a thrown exception during formatting or parsing.

Examples Using the US Locale:

```
Format Pattern                         Result
--------------                         -------
MMM dd                           ->>   Mar 09
MMM dd HH:                       ->>   Mar 09 23:
yyyy/MM/dd HH:                   ->>   2019/03/09 23:
"yyyy.MM.dd G 'at' hh:mm:ss z"   ->>   2019.03.09 AD at 15:08:56 PDT
"EEE, MMM d, ''yy"               ->>   Thu, March 9, '18
"h:mm a"                         ->>   12:08 PM
"hh 'o''clock' a, zzzz"          ->>   12 o'clock PM, PDT
"K:mm a, z"                      ->>   0:00 PM, CST
"yyyyy.MMMMM.dd GGG hh:mm aaa"   ->>   2019.March.09 AD 12:08 PM
```

4.1.10. Calling External Programs

Sample section:

```
#==============================================================================#
# Check if the following processes are running
#
HEADER = Executing PSOBR . . . . . . . . . . . . . . . . . . . . . . . . . . .
EXEC   = sh -f {DPIPE_HOME}/bin/psobr,L       # Run psobr command
HEADER = Checking for necessary OBR & Database processes . . . . . . . . . . .
EXEC   = /bin/uptime,L                        # Get system uptime
```

Definition of Variables:

EXEC = <command>,<logging_option>,<optional_timeout>

<command>	-	Execute an O/S command or user created script as if typed from command line.
<logging_option>	-	P=page, E=email, B=both, T=trap, L=log only
<optional_timeout>	-	OPTIONAL - Force the program or user script being called from the EXEC command directive to timeout if there is no response after the <optional_timeout> value have been exceeded. The default timeout is 5-minutes for all command directives except CHECK_DB which is 30-minutes.

The MPmon Command Directive of EXEC will display an error if the program or script being launched fails in its execution. If the program or script being launched terminates with an exit code (return code) of anything that is not zero (0), then an MPmon error will be shown in the MPmon report along with any potential errors messages from the program or script.

Example of Error Message from MPmon and Perl script:

EXEC: *****ERROR Non-zero exit status. Command:
/opt/Mobius/MPmon/bin/MPcheck_polling.pl exit status:21

MPcheck_polling error: Cannot find {PMDB_HOME}/BSMRApp.log

The above EXEC message is from MPmon stating the user script of *MPcheck_polling.pl* terminated with a non-zero (0) exit (return) code.

The following message was displayed by the user script.

4.1.11. Calling External SQLs

Sample section:

```
#================================================================================#
# Check the Database for table growth and overall size
#
HEADER = Checking OBR Database and Table sizes . . . . . .
EXEC_SQL = {MP_HOME}/scripts/ VRT_TableSizeCount.sql,L
    or
EXEC_SQL = SELECT ANCHOR_TABLE_NAME "OBR Table Name"\,
cast(AVG(USED_BYTES/1024/1024) as Decimal (15\,2)) "MB Size"\, cast(AVG(ROW_COUNT)
as Integer) "Row Count" FROM PROJECTION_STORAGE WHERE PROJECTION_NAME in (SELECT
projection_name FROM projections WHERE is_super_projection=1) GROUP BY
PROJECTION_SCHEMA\, ANCHOR_TABLE_NAME ORDER BY cast(AVG(USED_BYTES/1024/1024/1024)
as Decimal (15\,2))\, cast(AVG(ROW_COUNT) as Integer)\, ANCHOR_TABLE_NAME,L
```

Definition of Variables:

EXEC_SQL = </Full pathname/SQL-File.sql>,<logging_option>,<optional_timeout>
or
EXEC_SQL = <SQL Statement>,<logging_option>,<optional_timeout>

< SQL Filename >	-	Execute an SQL SELECT statement in an SQL file. The filename must end in ".sql".
or		
< SQL Statement >	-	Execute an SQL SELECT statement as if typed from an SQL command prompt line.
<logging_option>	-	P=page, E=email, B=both, T=trap, L=log only
<optional_timeout>	-	OPTIONAL - Force the program or user script being called from the EXEC command directive to timeout if there is no response after the <optional_timeout> value have been exceeded. The default timeout is 5-minutes for all command directives except CHECK_DB which is 30-minutes.

NOTES: Only SELECT statements are allowed. No SQL statements that can update or alter the database or data, or delete from the database are allowed.

The SELECT statement is a single line. While the line can wrap to appear to be multiple lines in the MPmon Config file, there are no line breaks (carriage returns / line feeds) in the statement.

The MPmon Command Directive of EXEC_SQL will display an error if SQL being launched fails in its execution. If the SQL being launched terminates with an exit code (return code) of anything that is not zero (0), then an MPmon error will be shown in the MPmon report along with any potential errors messages from the program or script.

Example of Error Message from MPmon and Perl script:

EXEC_SQL: *****ERROR Non-zero exit status. SELECT statement exit status:21

4.2. MPmon Email and Pager File

The MPmon Email and Pager file should contain email addresses and pager addresses for the contacts that are responsible for the administration of the monitored application. The default location for the Email and Pager file is: *{MOBIUS_HOME}/etc/MPmon_Email_Pager.List*.

Multiple Email_Pager files can be used if there are multiple shifts (2 or 3 shifts) throughout the day or if there is a shift responsible for issues that might arise after-hours. Enable this by modifying the Crontab entries for "*shrboadmin*" and adding the required MPmon executions using the "**-ef**" option and appropriate Email_Pager file for each shift.

Pages are sent as small emails to pagers and other wireless devices capable of receiving email based text messages. MPmon does not dictate the use of any particular wireless service provider. However, the service used must enable the user to receive short text email messages (less than 100 characters) on a text pager, mobile phone, or similar device.

A typical pager address would look like:

972672XXXX@<service_provider>.net

The configuration file has 3 columns for ease of reading and modification. All entries can be commented out by adding a "#" sign at the beginning of the line when the contact goes on holiday or is not available during a period of time.

The Variable Name column defines the section that MPmon will process. The possible options are:

Variable Name	Definition
MP_EMAIL	Email Address of the recipient to receive the MPmon log file via email
MP_PAGER	Pager Address of the recipient to receive a MPmon text page (i.e.: 972672XXXX@<provider>.com)

Sample section:

```
#=============================================================================#
# Variable Name  =  Value of Variable         # Comment                      #
#=============================================================================#
#
MP_EMAIL = MPmon_Support@Mobius.com           # Mobius Support Email
MP_PAGER = 987654XXXX@<provider.com>          # Test pager entry
```

4.3. MPmon Trap Generation File

The MPmon Trap Generation file should contain hostname and SNMP port number (typically Port 162) of the trap recipients. The default location for the Trap Generation file is: *{MOBIUS_HOME}/etc/MP_TrapGen.Config*. Multiple trap recipients can be listed in this file. This enables the same trap to be sent to multiple trap recipients.

Executing **MPmon -T** displays output on the user's screen and only if exceptions are found, sends traps to the designated trap recipients. A MPmon report is also emailed to the recipients listed in the *{MOBIUS_HOME}/etc/MPmon_Email_Pager.List* file.

The configuration file has 3 columns for ease of reading and modification. All entries can be commented out by adding a "#" sign at the beginning of the line when not needed for a period of time.

The Variable Name column defines the section that MPmon will process. The possible options are:

Variable Name	Definition
MP_TRAP	Hostname, SNMP port number, and optional Read Community string (default of "public") to send a trap to.

Sample section:

```
#=============================================================================#
# Variable Name  =  Value of Variable   # Comment                             #
#=============================================================================#
#
MP_TRAP = MPnms01,162                   # Mobius primary NMS Server
#MP_TRAP = MPnms02,162                  # Mobius backup NMS Server
```

4.4. Configuring an API Launch

MPmon can launch API calls to other management applications. This feature enables, for example, an API call to be launched to generate a trouble ticket within the user's trouble ticket system when errors are detected. The resulting trouble ticket will be populated with the corresponding error from the application being monitored by MPmon.

The API call feature is customizable and specific usage of this feature will depend on the specific trouble ticketing system in use and how it is implemented. Using this feature may require modification (editing) of the MPmon startup script of **mpmon.sh** in the directory of *{MOBIUS_HOME}/MPmon/bin*.

With the modifications in place, it will require using the "**mpmon -p**" option to launch the API. Running MPmon without the "**-p**" option will only display the results to the screen.

Below is an example of populating a Remedy trouble ticket system. This is the section of **mpmon.sh** that will require editing.

1. Modify **`MP_APION=0`** and change it to: **`MP_APION=1`**

2. Modify the **`"Launching MPmon API"`** section below and insert the appropriate commands from the API application that will call the trouble ticket system and open a ticket with the supplied error message.

```
#============================================================================#
# Look for "*****ERROR"s in the MPmon.log and launch API to create a ticket
#
# Turn this section On (1) or Off (0). Default is Off and only works w/ Paging
MP_APION=0

if [ $MP_PAGERMODE -eq "1" ] && [ $MP_APION -eq "1" ] ; then
   MP_APICNT=0
   MP_APIIN=`grep "\*\*\*\*\* ERROR" ${MP_OUTFILE}`
   MP_APIOUT=`echo ${MP_APIIN}`
   MP_APICNT=`grep "\*\*\*\*\* ERROR" ${MP_OUTFILE} | wc -l`

   if [ ${MP_DEBUGMODE} = "1" ] ; then
      echo "MPmon API: ${MP_APIIN} has ${MP_APICNT} lines to act upon"
   fi

   if [ ${MP_APICNT} -gt 0 ] ; then
      echo "Launching MPmon API"
      # Launch Remedy ARS trouble ticket
      #cp ${MOBIUS_HOME}/MPmon/etc/input_file.ars ${MOBIUS_HOME}/tmp/...
      #echo "9       4       ${MP_APIOUT}" >> ${MOBIUS_HOME}/MPmon/tmp/...
      #/opt/<ARS_Dir>/bin/createARSentry -s <mail_server> -A /opt/<ARS_Dir>/...
   fi
fi
```

5. MPmon Installation Process

5.1. Prerequisites to Installation

1. The application to be monitored should be installed, configured, and fully operational before installing MPmon. If MPmon will be used to monitor OBR, OBR should be completely installed along with the database on a local or remote database server, OBR Content Packs and other modules prior to the installation of the MPmon.

2. For Windows users, Cygwin (Unix for Windows) must be installed on the server before MPmon can be installed. Cygwin should be installed into a closely related directory such as D:\Mobius. See the README_Cygwin_Install_Guide to install Cygwin.

3. If required, Java v1.5 will be installed only into the Mobius home directory and will not affect any other Java applications. At the Command prompt, type "**java -version**" or "**jre**" to determine the version number or if installed at all.

4. Contact Mobius and obtain a temporary or permanent license file before starting the MPmon installation.

5. If MPmon was downloaded from the Mobius site, unTAR the install file into "/tmp/MPmon_install" and use this directory name during the install.

6. If using MPmon's OMi messaging feature, The following should be in place before installing MPmon:
   - The OMi environment must be completely operational.
   - The OM Agent should be installed and operation on the server where MPmon is to be installed.
   - The appropriate MPmon template for OMi should be pushed to the OM agent where MPmon is to be installed.

5.2. Installing MPmon

1. Login as the OBR Admin user of "*shrboadmin*" on a Unix OBR Server or as "*Administrator*" on a Windows OBR Server.

2. Change directory to where the MPmon install bits have been downloaded and unzipped.

3. For Windows, installing MPmon is a two-step process, please open and refer to the README_Cygwin_Install_Guide and README_MPmon_Win_Install_Guide throughout this install process:

   - Install and configure Cygwin for Windows using: **setup1_Cygwin_Windows.bat**
   - Install MPmon using the install procedure of: **setup2_MPmon_Windows.bat**

4. For Unix, please open and follow the installation steps detailed in the Unix install text file of: README_MPmon_Unix_Install_Guide. To view the install guides, use "**more**".

   - Install MPmon using the install procedure of: **setup_MPmon_Unix**

5.3. Uninstalling MPmon

1. Login as the user of "*shrboadmin*" for Unix or "Administrator" for Windows on the OBR Server.

2. Change directory to the OBR application home directory or type the alias of "pd".

   cd $MOBIUS_HOME

3. Remove the directory of the MPmon application.

   rm –r MPmon

   rm –r $MOBIUS_HOME/bin/mpmon

4. Remove the Java install as the user of "root".

   /bin/rpm -ev jre1.8-1.8.0_211-fcs.x86_64

5. If necessary, modify the ***$PMDB_HOME/lib/trendtimer.sched*** file and remove or comment out any lines that were added in the file that automates any MPmon process.

6. Use crontab to remove the Hourly and Daily MPmon schedules. Remove the following like entries in *shrboadmin's* crontab file using **crontab –e** or **crontab –r** (to remove all).

```
# Run MPmon every hour (page) and every day (email) after rollups
# or at the end of the day (11:54pm) looking for potential OV issues.
#59 * * * * {MOBIUS_HOME}/MPmon/bin/mpmon.sh -p > /dev/null 2>&1
# Creates a daily MPmon report after the morning rollups
#30 7 * * * {MOBIUS_HOME}/MPmon/bin/mpmon.sh -e > /dev/null 2>&1
# Creates a daily MPmon report for the entire day
#30 23 * * * {MOBIUS_HOME}/MPmon/bin/mpmon.sh -e > /dev/null 2>&1
# Creates a MPmon report for OM monitoring
#15 * * * * /opt/Mobius/MPmon/bin/mpmon.sh -i /opt/Mobius/etc/
MPmon_OMi.Config -o /opt/Mobius/Log/MPmon_OMi.log > /dev/null 2>&1
# Archive and compress the MPmon daily log file
#0 0 * * * /opt/Mobius/bin/MParchivelog.sh /opt/Mobius/Log/
MPmon_Archive.log
# Archive and compress the Time.log log file
#0 0 * * * /opt/Mobius/bin/MParchivelog.sh /opt/OBR/log/Time.log
```

NOTE: To uninstall Cygwin for Windows, please refer to the last section in the README_Cygwin_Install_Guide.rtf document.

NOTE: It is not necessary to uninstall the Mobius Utilities and Aliases that were installed with MPmon. They are standalone and not dependent on MPmon, only OBR.

6. Sample MPmon Report

```
#=============================================================================#
#                              MPmon v5.2.2                                   #
#                        Powered By Mobius Partners                           #
#                  Copyright 2000 - 2019 Mobius Partners                      #
#               (www.mobiuspartners.com) All rights reserved                  #
#=============================================================================#
MPmon v5.2.2      (Build 2019-04-23 13:46:11) on host hpobr101-a1

MPmon started on: Thu June 20 12:26:29 CDT 2019
Evaluation Expires on: Thu Aug 20 12:26:29 CDT 2019


#=============================================================================#
# Executing PSOBR . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . #
#=============================================================================#


Checking the 'Status' of running Services:
#=============================================================================#

OBR Admin Services:
HPE_PMDB_Platform_Administrator is running
HPE_PMDB_Platform_DB_Logger is running
HPE_PMDB_Platform_IA is stopped
HPE_PMDB_Platform_IM is running

OBR Data Collection Services:
HPE_PMDB_Platform_Collection is running
HPE_PMDB_Platform_NRT_ETL is stopped

OBR Data Processing Services:
HPE_PMDB_Platform_Orchestration is running
HPE_PMDB_Platform_TaskManager Service is running
HPE_PMDB_Platform_JobManager Service is running


OBR BO Services ============================================================
UID          PID   PPID  C STIME TTY          TIME CMD
504      100277      1  0 Jun20 ?        00:00:00 /bin/sh
/opt/HP/BSM/BOE4/sap_bobj/enterprise_xi40/generic/bobjrestart.sh
/opt/HP/BSM/BOE4/sap_bobj/enterprise_xi40/generic/javalaunch.sh
504      100287 100277  0 Jun20 ?        00:59:43
/opt/HP/BSM/BOE4/sap_bobj/enterprise_xi40/linux_x64/sapjvm/bin/java
504      106301 100287  3 Jun20 ?        21:00:05
/opt/HP/BSM/BOE4/sap_bobj/enterprise_xi40/linux_x64//boe_cmsd
504      112170 100287  1 Jun20 ?        08:29:58
/opt/HP/BSM/BOE4/sap_bobj/enterprise_xi40/linux_x64/sapjvm/bin/java
504      112192 100287  0 Jun20 ?        00:06:34
/opt/HP/BSM/BOE4/sap_bobj/enterprise_xi40/linux_x64//boe_eventsd
504      112199 100287  0 Jun20 ?        01:48:35
/opt/HP/BSM/BOE4/sap_bobj/enterprise_xi40/linux_x64/sapjvm/bin/java
504      112221 100287  0 Jun20 ?        00:18:16
/opt/HP/BSM/BOE4/sap_bobj/enterprise_xi40//linux_x64/boe_xcprocd
504      112241 100287  0 Jun20 ?        00:06:41
/opt/HP/BSM/BOE4/sap_bobj/enterprise_xi40/linux_x64//ConnectionServer
504      112257 100287  0 Jun20 ?        00:52:30
/opt/HP/BSM/BOE4/sap_bobj/enterprise_xi40//linux_x64/boe_xccached
504      112280 100287  0 Jun20 ?        00:23:11
/opt/HP/BSM/BOE4/sap_bobj/enterprise_xi40/linux_x64//WIReportServer
```

```
504      112310 100287  0 Jun20 ?        00:12:28
/opt/HP/BSM/BOE4/sap_bobj/enterprise_xi40/linux_x64//boe_filesd
504      112337 100287  0 Jun20 ?        00:11:01
/opt/HP/BSM/BOE4/sap_bobj/enterprise_xi40/linux_x64//boe_jobsd
504      112385 100287  0 Jun20 ?        00:07:29
/opt/HP/BSM/BOE4/sap_bobj/enterprise_xi40/linux_x64//boe_filesd
504      446503 112221  0 Jun20 ?        00:42:37
/opt/HP/BSM/BOE4/sap_bobj/enterprise_xi40/linux_x64/sapjvm/bin/../bin/java
root      99817      1  0 Jun20 ?        00:31:20 /opt/HP/BSM//JRE64/bin/java -
Djava.util.logging.config.file=/opt/HP/BSM/PMDB/BOWebServer

OBR Vertica Process ======================================================
UID         PID   PPID  C STIME TTY          TIME CMD
root      33072      1  0 Jun20 ?        00:24:31 /opt/OV//nonOV/perl/a/bin/perl
/opt/HP/BSM//PMDB/lib/perl/sec/2.7.6/sec -
conf=/opt/HP/BSM//PMDB/config/SHRServer_IA.rule -
conf=/opt/HP/BSM//PMDB/config/BO_IA.rule -conf=/opt/HP/BSM//PMDB/config/Vertica_IA.rule
-input=/opt/HP/BSM//PMDB/log/IAEvent.log -input= -intevents -log
/opt/HP/BSM//PMDB/log/IAEngine.log
vertica   58361      1  0 Jun20 ?        00:10:46 /opt/vertica/spread/sbin/spread -c
/opt/HP/Vertica_Data/Catalog_Files/pmdb/v_pmdb_node0001_catalog/spread.conf
vertica   58363      1  4 Jun20 ?        1-05:21:49 /opt/vertica/bin/vertica -D
/opt/HP/Vertica_Data/Catalog_Files/pmdb/v_pmdb_node0001_catalog -C pmdb -n
v_pmdb_node0001 -h 10.200.101.204 -p 5433 -P 4803 -Y ipv4
vertica   61176  58363  0 Jun20 ?        00:03:40 /opt/vertica/bin/vertica-udx-zygote 16
14 58363 debug-log-off
/opt/HP/Vertica_Data/Catalog_Files/pmdb/v_pmdb_node0001_catalog/UDxLogs 60 17 0

OBR Postgres Process =====================================================
UID         PID   PPID  C STIME TTY          TIME CMD
postgres  13635      1  0 Jun20 ?        00:03:23 /opt/HP/BSM/Postgres/bin/postgres -D
/opt/HP/BSM/Postgres/data
postgres  13636  13635  0 Jun20 ?        00:00:00 postgres: logger process
postgres  13638  13635  0 Jun20 ?        00:02:26 postgres: checkpointer process
postgres  13639  13635  0 Jun20 ?        00:00:14 postgres: writer process
postgres  13640  13635  0 Jun20 ?        00:01:29 postgres: wal writer process
postgres  13641  13635  0 Jun20 ?        00:00:24 postgres: autovacuum launcher process
postgres  13642  13635  0 Jun20 ?        00:05:08 postgres: stats collector process
postgres  35050  13635  0 Jun20 ?        00:02:33 postgres: pmdb_admin dwabc
10.200.101.204(47675) idle
postgres  35055  13635  0 Jun20 ?        01:20:23 postgres: pmdb_admin dwabc
10.200.101.204(47676) idle
postgres  35059  13635 18 Jun20 ?        4-14:34:17 postgres: pmdb_admin dwabc
10.200.101.204(47680) SELECT
postgres 121529  13635  0 Jun20 ?        00:00:00 postgres: pmdb_admin dwabc
10.200.101.204(56203) idle
postgres 420941  13635  0 21:39 ?        00:00:00 postgres: pmdb_admin dwabc
10.200.101.204(46760) idle
postgres 424994  13635  0 21:40 ?        00:00:00 postgres: pmdb_admin dwabc
10.200.101.204(46921) idle in transaction
postgres 424995  13635  0 21:40 ?        00:00:00 postgres: pmdb_admin dwabc
10.200.101.204(46922) idle in transaction
postgres 427001  13635  0 21:41 ?        00:00:00 postgres: pmdb_admin dwabc
10.200.101.204(47063) idle in transaction
postgres 437802  13635  0 21:47 ?        00:00:00 postgres: pmdb_admin dwabc
10.200.101.204(48112) idle in transaction
postgres 457960  13635  0 21:56 ?        00:00:00 postgres: pmdb_admin dwabc
10.200.101.204(48955) idle in transaction
postgres 457961  13635  0 21:56 ?        00:00:00 postgres: pmdb_admin dwabc
10.200.101.204(48956) idle in transaction
postgres 477629  13635  0 22:05 ?        00:00:00 postgres: pmdb_admin dwabc
10.200.101.204(50219) idle in transaction
postgres 478094  13635  0 22:05 ?        00:00:00 postgres: pmdb_admin dwabc
```

```
10.200.101.204(52833) idle
postgres 519266  13635  0 22:24 ?        00:00:00 postgres: pmdb_admin dwabc
10.200.101.204(52835) idle
postgres 519273  13635  0 22:24 ?        00:00:00 postgres: pmdb_admin dwabc
10.200.101.204(52836) idle
postgres 520239  13635  0 22:24 ?        00:00:00 postgres: pmdb_admin dwabc
10.200.101.204(52860) idle
postgres 520377  13635  0 22:24 ?        00:00:00 postgres: pmdb_admin dwabc
10.200.101.204(52869) idle
postgres 520378  13635  0 22:24 ?        00:00:00 postgres: pmdb_admin dwabc
10.200.101.204(52870) idle
postgres 520379  13635  0 22:24 ?        00:00:00 postgres: pmdb_admin dwabc
10.200.101.204(52871) idle
postgres 522920  13635  0 22:26 ?        00:00:00 postgres: pmdb_admin dwabc
10.200.101.204(53075) idle

OBR Running Processes ================================================
UID          PID    PPID  C STIME TTY          TIME CMD
root      523238 492445 16 22:26 ?        00:00:02 /opt/HP/BSM/PMDB/bin/stage
config=/opt/HP/BSM/PMDB/stagerules/platform/CoreVirtualizationVMWare_0_Stage_K_VMWare_Cl
uster_0_stagerule.stg
root      523489  34279 15 22:26 ?        00:00:01 stage
config=/opt/HP/BSM/PMDB/stagerules/OMi10x_SM_VMWare_Bridge_stage.stg

Done on:  Thu Jun 20 22:26:51 CDT 201



#=========================================================================#
# Checking for necessary OBR & Vertica processes. . . . . . . . . . . . . #
#=========================================================================#

 22:26:51 up 25 days,  6:18,  1 user,  load average: 53.59, 52.61, 51.74

CHECK_PROCESS: process 'trendtimer' is running on hpobr10l-a1

CHECK_PROCESS: process 'vertica' is running on hpobr10l-a1

CHECK_PROCESS: process 'v_pmdb_node0' is running on hpobr10l-a1

CHECK_PROCESS: process 'postgres' is running on hpobr10l-a1

CHECK_PROCESS: process 'adminServer' is running on hpobr10l-a1

CHECK_PROCESS: process 'SHRServer_IA.rule' is running on hpobr10l-a1

CHECK_PROCESS: process 'flink.hprof' is running on hpobr10l-a1

CHECK_PROCESS: process 'sis_aggregate' is running on hpobr10l-a1

CHECK_PROCESS: process 'BOWebServer' is running on hpobr10l-a1


#=========================================================================#
# Checking for available network sockets (ports). . . . . . . . . . . . . #
#=========================================================================#

CHECK_PORT: Service is running on port hpobr10l-a1:5433
CHECK_PORT: Service is running on port hpobr10l-a1:21425
CHECK_PORT: Service is running on port hpobr10l-a1:8443
CHECK_PORT: Service is running on port hpobr10l-a1:21412
```

```
#=========================================================================#
# Checking for hung processes . . . . . . . . . . . . . . . . . . . . . . #
#=========================================================================#

CHECK_TIME: Process 'stage' is currently running and has not exceeded the runtime
threshold on hpobr10l-a1
CHECK_TIME: Process 'stage' is currently running and has not exceeded the runtime
threshold on hpobr10l-a1

CHECK_TIME: Process 'mpmon' is currently running and has not exceeded the runtime
threshold on hpobr10l-a1
CHECK_TIME: Process 'mpmon' is currently running and has not exceeded the runtime
threshold on hpobr10l-a1



#=========================================================================#
# Checking OBR related File Systems for available space . . . . . . . . . #
#=========================================================================#

Filesystem           1M-blocks  Used Available Use% Mounted on
/dev/mapper/vg_rhel6564b-lv_root
                       22054  5713     15215  28% /
tmpfs                   3937     1      3937   1% /dev/shm
/dev/sda1                477   106       346  24% /boot
/dev/sdb1              32254 30646         0 100% /.swap/swapfile2
/dev/sdc1             201585 51304    140041  27% /opt
/dev/sdd               50397 34879     12958  73% /mnt/OBR_Install

CHECK_FS: Used space for filesystem / is OK at 28% used

CHECK_FS: Used space for filesystem /opt/HP is OK at 27% used



#=========================================================================#
# Checking OBR Data Tables and max ta_periods . . . . . . . . . . . . . . #
#=========================================================================#

CHECK_TABLE: Latest collected data for SR_SM_CPU
CHECK_TABLE:            Current time: 2019-06-20 22:35:43.623207
CHECK_TABLE:           MAX ta_period: 2019-06-20 21:50:00

CHECK_TABLE: Latest collected data for SR_SM_DISK
CHECK_TABLE:            Current time: 2019-06-20 22:35:45.071095
CHECK_TABLE:           MAX ta_period: 2019-06-20 21:20:00
CHECK_TABLE:       Threshold minutes: 300
CHECK_TABLE: *****ERROR: Data in SR_SM_DISK is 3676 minutes old, MAX ta_period is 2019-
06-18 20:00

CHECK_TABLE: Latest collected data for SR_SM_NODE_RES
CHECK_TABLE:            Current time: 2019-06-20 22:35:45.286352
CHECK_TABLE:           MAX ta_period: 2019-06-20 21:50:00

CHECK_TABLE: Latest collected data for SH_SM_CPU
CHECK_TABLE:            Current time: 2019-06-20 22:35:45.972056
CHECK_TABLE:           MAX ta_period: 2019-06-20 20:00:00

CHECK_TABLE: Latest collected data for SH_SM_DISK
CHECK_TABLE:            Current time: 2019-06-20 22:35:46.574202
CHECK_TABLE:           MAX ta_period: 2019-06-20 21:00:00
CHECK_TABLE:       Threshold minutes: 300
```

```
CHECK_TABLE: *****ERROR: Data in SH_SM_DISK is 3696 minutes old, MAX ta_period is 2019-
06-18 21:00:00

CHECK_TABLE: Latest collected data for SH_SM_NODE_RES
CHECK_TABLE:            Current time: 2019-06-20 22:35:46.972222
CHECK_TABLE:           MAX ta_period: 2019-06-20 20:00:00

CHECK_TABLE: Latest collected data for SD_SM_CPU
CHECK_TABLE:            Current time: 2019-06-20 22:35:47.537187
CHECK_TABLE:           MAX ta_period: 2019-06-20 00:00:00

CHECK_TABLE: Latest collected data for SD_SM_DISK
CHECK_TABLE:            Current time: 2019-06-20 22:35:47.887222
CHECK_TABLE:           MAX ta_period: 2019-06-18 00:00:00
CHECK_TABLE:       Threshold minutes: 3300
CHECK_TABLE: *****ERROR: Data in SD_SM_DISK is 3756 minutes old, MAX ta_period is 2019-
06-18 00:00:00

CHECK_TABLE: Latest collected data for SD_SM_NODE_RES
CHECK_TABLE:            Current time: 2019-06-20 22:35:48.329934
CHECK_TABLE:           MAX ta_period: 2019-06-20 00:00:00

CHECK_TABLE: *****ERROR: The database table SR_VI_VM exists but contains no ta_period
data.

CHECK_TABLE: *****ERROR: The database table SH_VI_VM exists but contains no ta_period
data.

CHECK_TABLE: *****ERROR: The database table SD_VI_VM exists but contains no ta_period
data.



#=========================================================================#
# Checking OBR Database tables for row counts and size in bytes . . . . . #
#=========================================================================#

           anchor_table_name           | row_count | used_bytes
---------------------------------------+-----------+------------
 K_CI_Application_                     |         0 |          0
 K_CI_Business_Service_                |         0 |          0
 K_CI_Cust_Bridge_                     |         0 |          0
 K_CI_DNS_                             |         0 |          0
 K_CI_DT                               |         0 |          0
 K_CI_DT_                              |         0 |          0
 K_CI_Database_                        |         0 |          0
 K_CI_Group_Bridge_                    |         0 |          0
 K_CI_Loc_Bridge_                      |         0 |          0
 K_CI_Monitor_                         |         0 |          0
 K_CI_Network_Interface_               |         0 |          0
 K_CI_Network_Node_                    |         0 |          0
 K_CI_SM_Bridge_                       |         0 |          0
 K_CI_SoftwareElement_                 |         0 |          0
 K_CI_Subnet_                          |         0 |          0
 K_CI_Transaction_                     |         0 |          0
 K_CI_Type_                            |         0 |          0
 K_CI_Type_Bridge_                     |         0 |          0
 K_CI_VM_                              |         0 |          0
 K_CMDB_View_                          |         0 |          0
 K_Cluster_RP_VM_Bridge                |         0 |          0
 K_Customer_                           |         0 |          0
 K_DStore_Cluster_Bridge_              |         0 |          0
 K_Event_                              |         0 |          0
```

K_HIValue_	0	0
K_Host_Cluster_Bridge	0	0
K_Host_Cluster_Bridge_	0	0
K_Location_	0	0
K_Network_Metric_	0	0
K_Person_	0	0
K_RP_Host_Bridge	0	0
K_RP_Host_Bridge_	0	0
K_System_Metric_	0	0
K_UserGroup_	0	0
K_VMWare_Cluster_	0	0
K_VMWare_DataCenter_	0	0
K_VMWare_DataStore_	0	0
K_VMWare_DataStore_Cluster_	0	0
K_VMWare_Metric_	0	0
K_VMWare_ResourcePool_	0	0
K_VMWare_VMDisk_	0	0
K_VM_DStore_Bridge_	0	0
K_VM_Host_Bridge	0	0
K_VM_Host_Bridge_	0	0
K_VM_Metric_	0	0
K_VM_Metric_DT	0	0
K_VM_Metric_DT_	0	0
K_VMware_Resourcepool_Hosts_	0	0
SD42SD_Event_AssignByUser	0	0
SD42SD_Event_AssignByUserGroup	0	0
SD42SD_Event_byView	0	0
SD_Event_AssignByUser	0	0
SD_Event_AssignByUserGroup	0	0
SD_Event_Count	0	0
SD_Event_Duration	0	0
SD_Event_byView	0	0
SD_HIHistory	0	0
SD_HISev	0	0
SD_KPIStatus	0	0
SD_Network_Interface	0	0
SD_Network_Interface_Forecast	0	0
SD_Network_Node	0	0
SD_Network_Node_Forecast	0	0
SD_VI_VM	0	0
SD_VMWare_Cluster	0	0
SD_VMWare_DataStore	0	0
SD_VMWare_ResPool	0	0
SD_VMWare_VM	0	0
SH_Event_AssignByUser	0	0
SH_Event_AssignByUserGroup	0	0
SH_Event_Count	0	0
SH_Event_Duration	0	0
SH_Event_byView	0	0
SH_HIHistory	0	0
SH_HISev	0	0
SH_KPIStatus	0	0
SH_Network_Interface	0	0
SH_Network_Interface_Forecast	0	0
SH_Network_Node	0	0
SH_Network_Node_Forecast	0	0
SH_VI_VM	0	0
SH_VMWare_Cluster	0	0
SH_VMWare_DataStore	0	0
SH_VMWare_ResPool	0	0
SH_VMWare_VM	0	0
SR_NetworkInterface_Fact	0	0
SR_NetworkInterface_Fact_	0	0

SR_NetworkNode_Fact	0	0
SR_NetworkNode_Fact_	0	0
SR_Network_Interface	0	0
SR_Network_Interface_	0	0
SR_Network_Interface_Forecast	0	0
SR_Network_Interface_Forecast_	0	0
SR_Network_Node	0	0
SR_Network_Node_	0	0
SR_Network_Node_Forecast	0	0
SR_Network_Node_Forecast_	0	0
SR_System_Fact	0	0
SR_System_Fact_	0	0
SR_VI_VM	0	0
SR_VI_VM_	0	0
SR_VMWare_Cluster	0	0
SR_VMWare_Cluster_	0	0
SR_VMWare_DataStore	0	0
SR_VMWare_DataStore_	0	0
SR_VMWare_Datacenter_Fact	0	0
SR_VMWare_Datacenter_Fact_	0	0
SR_VMWare_ResPool	0	0
SR_VMWare_ResPool_	0	0
SR_VMWare_ResPool_Hosts	0	0
SR_VMWare_ResPool_Hosts_	0	0
SR_VMWare_VM	0	0
SR_VMWare_VM_	0	0
SR_VM_Fact	0	0
SR_VM_Fact_	0	0
T_SM_NodeExptns	1	68
T_Update_NetIf_time	1	68
T_Uptime_Scope	1	68
K_Shift_Time_Range	7	139
md_groupbridge_table	1	160
md_locationbridge_table	1	163
TMP_KEYCITYPE	10	190
K_VM_DStore_Bridge	1	193
K_DStore_Cluster_Bridge	1	200
K_CI_Cust_Bridge	1	206
K_Shift	1	208
K_CI_SM_Bridge	5	223
K_VIEW_CITYPE_HI	1	228
K_VIEW_CITYPE_KPI	1	228
SHR_Config	2	240
K_CI_Type_Bridge	1	249
K_Network_Metric	1	256
K_System_Metric	1	263
K_VMWare_Metric	1	263
K_VM_Metric	1	263
K_CMDB_View	1	272
K_KPI	1	311
md_bridge_table	1	318
K_HI	1	326
K_SM_PhysicalDisk_	1	330
temp_dict_total_nodes	5	348
K_VMWare_DataCenter	1	356
K_HIValue	1	361
K_VMWare_ResourcePool	1	367
K_VMware_Resourcepool_Hosts	1	367
K_CI_Application	1	401
K_UserGroup	1	401
dsi_dependency_list	33	408
K_Location	1	424
IM_TABLE_PROFILE	5	424

K_CI_Monitor	1	426
K_CI_SoftwareElement	1	426
K_CI_Business_Service	1	429
K_Person	1	437
K_CI_DNS	1	454
K_CI_Transaction	1	454
K_Customer	1	458
K_CI_Type	6	471
K_CI_Loc_Bridge	18	482
K_CI_Group_Bridge	18	488
K_VMWare_VMDisk	1	501
K_CI_Database	1	510
K_VMWare_DataStore_Cluster	1	522
K_CI_Subnet	1	566
K_SM_PhysicalDisk	2	594
bsmr_user	5	601
DATETIMERANGE	25	608
K_VMWare_Cluster	1	650
STREAM_0_cmdbviews_0_ALL_VIEWS	142	656
STREAM_0_relations_0_MSSQL_Deployment	12	670
K_VMWare_DataStore	1	721
K_SM_NetInterface_	4	734
IM_PM_OS_INFO	3	747
Deployed_ContentPack	5	765
K_SM_CPU_	8	772
STREAM_0_FILESYSTEM_0_SCOPE	12	838
K_SM_FileSystem_	2	880
md_schema	8	902
K_CI_Network_Interface	1	905
md_bridge_column	13	910
K_CI_Network_Node	1	967
K_SM_NetInterface	5	979
K_Group	34	979
K_SM_CPU	9	1070
STREAM_0_generic_0_SM_PA	39	1109
md_retention_profile	27	1134
K_Event	1	1143
STREAM_0_RSR_OVPA_0_SCOPE	24	1213
K_SM_FileSystem	3	1229
SR_SM_DISK_	12	1244
K_CI_Bridge	29	1264
IM_PM_APPS_INFO	3	1350
STREAM_0_generic_0_OMi10x	47	1351
STREAM_0_generic_0_ServiceHealth	47	1351
K_CI_VM	1	1362
STREAM_0_NETIF_0_SCOPE	24	1380
STREAM_0_relations_0_SM_PA	66	1441
SR_SM_FILESYSTEM_	36	1448
IM_DW_FACTTABLE	57	1535
dsi_reporting_status	16	1679
RSR_OVPA_Global_	36	1912
temp_SR_NodeExceptions	85	1918
SR_SM_Node_Exceptions	85	1940
K_Group_	284	1948
md_downtime_table	18	2083
K_CI	18	2155
SD42SD_SM_FILESYSTEM_FORE	30	2230
K_CI_Bridge_	78	2443
SR_SM_NETINTERFACE_	72	2540
STREAM_0_CPU_0_SCOPE	48	2590
md_downtime_column	144	2736
K_CI_	86	2898
dsi_reporting_apps	16	3067

```
 K_CI_Node                                |          5 |        3272
 md_fact_table                            |         39 |        3782
 md_aggregate_table                       |         46 |        4049
 SD42SD_SM_NODE_RES                       |         30 |        4136
 SR_SM_CPU_                               |        144 |        4296
 md_dimension_table                       |         56 |        4463
 SD_SM_NETINTERFACE                       |         60 |        4747
 LND_SCOPE_CONFIGURATION_NodeCNF          |        613 |        5432
 LND_SCOPE_CONFIGURATION_SysUx            |        613 |        5458
 SD_SM_DISK                               |          5 |        5589
 STREAM_0_NODE_RES_0_SCOPE                |         24 |        5659
 bsmr_report                              |         44 |        6038
 md_stage_table                           |         93 |        6174
 BO_CP_OBJECTS                            |        109 |        6758
 Shift_Fact                               |        426 |        7223
 K_CI_System                              |          9 |        9429
 SR_SM_NODE_RES_                          |         36 |        9543
 SD_SM_FILESYSTEM                         |         30 |       12479
 SH_SM_DISK                               |         96 |       12492
 K_CI_Node_                               |        651 |       12591
 STREAM_0_CONFIGURATION_0_SCOPE           |        612 |       13084
 SD_SM_NODE_AVAIL                         |         30 |       14639
 SH_SM_NODE_AVAIL                         |        696 |       14783
 dsi_reporting_components                 |        795 |       15800
 SD_SM_NODE_RES                           |         30 |       17160
 STREAM_0_unix_0_SM_PA                    |         72 |       17592
 SD_SM_NODE_EXCEPTIONS                    |         30 |       20939
 STREAM_0_nt_0_SM_PA                      |        216 |       22608
 md_dimension_column                      |        816 |       24297
 K_CI_System_                             |        867 |       25116
 SH_SM_NODE_EXCEPTIONS                    |        695 |       25943
 SD_SM_CPU                                |        120 |       26656
 md_fact_column                           |        894 |       28456
 md_aggregate_column                      |       1503 |       31233
 LND_SCOPE_GLOBAL_NodeGlobal              |       7366 |       32360
 SH_SM_FILESYSTEM                         |        695 |       33812
 LND_SCOPE_GLOBAL_SystUx                  |       7366 |       46212
 STREAM_0_GLOBAL_0_SCOPE                  |       7365 |       50876
 SR_SM_DISK                               |       1142 |       51065
 md_stage_column                          |       1702 |       53101
 SH_SM_NETINTERFACE                       |       1390 |       62772
 IM_DW_FACTTABLE_FILL                     |       6384 |       68194
 SR_SM_NODE_AVAIL                         |       8336 |       96847
 IM_PM_APPS_AVAIL                         |       8078 |      105185
 IM_PM_OS_INFO_FILLDETAIL                 |       8083 |      108676
 SR_SM_FILESYSTEM                         |       8331 |      123573
 IM_PM_WAS_USAGE                          |       8079 |      139842
 RSR_OVPA_Global                          |       8336 |      183524
 SH_SM_CPU                                |       2780 |      195926
 SH_SM_NODE_RES                           |        696 |      216282
 SR_SM_NETINTERFACE                       |      16662 |      266715
 IM_CONTENT_HEALTH_SUMMARY                |      56282 |      636559
 SR_SM_NODE_RES                           |       8343 |      835788
 SR_SM_CPU                                |      33324 |      899505
 DATETIME                                 |    4194304 |     1363147
 IM_CONTENT_HEALTH_TABLE                  |     369844 |     5700528
(315 rows)
```

```
#=========================================================================#
# Checking Vertica log for errors & warnings. . . . . . . . . . . . . . . #
#=========================================================================#

CHECK_LOG: ERROR,: 0 found!


#=========================================================================#
# Checking hpacollector.log file. . . . . . . . . . . . . . . . . . . . . #
#=========================================================================#

CHECK_LOG: ERROR,: 0 found!


#=========================================================================#
# Checking dbcollector.log file . . . . . . . . . . . . . . . . . . . . . #
#=========================================================================#

CHECK_LOG: ERROR,: 0 found!

CHECK_LOG: ERROR,: 0 found!


#=========================================================================#
# Checking stage.log file . . . . . . . . . . . . . . . . . . . . . . . . #
#=========================================================================#

CHECK_LOG: ERROR,: 3 found!

2019-06-20 22:08:28,511 ERROR,
com.hp.bto.pmdb.stage.processdata.ExecuteStage.executeDataMoveSQL , [ ABCBatchID:786662,
ABCStreamID:ETL_SystemManagement_PA@Dim_SystemNode, ABCStepID:Stage_OVPAGlobalConfig,
ABCProcessID:787726 ] [280473053] Failed to load rows to stage table K_CI_Node_ from
LND_SM_PA_unix_uni uni

2019-06-20 22:24:14,888 ERROR,
com.hp.bto.pmdb.stage.processdata.ExecuteStage.executeDataMoveSQL , [ ABCBatchID:786662,
ABCStreamID:ETL_SystemManagement_PA@Dim_SystemNode, ABCStepID:Stage_OVPAGlobalConfig,
ABCProcessID:788005 ] [924035780] java.sql.SQLSyntaxErrorException:
[Vertica][VJDBC](4856) ERROR: Syntax error at or near "WHERE"

2019-06-20 22:24:14,888 ERROR,
com.hp.bto.pmdb.stage.processdata.ExecuteStage.executeDataMoveSQL , [ ABCBatchID:786662,
ABCStreamID:ETL_SystemManagement_PA@Dim_SystemNode, ABCStepID:Stage_OVPAGlobalConfig,
ABCProcessID:788005 ] [924035780] Failed to load rows to stage table K_CI_Node_ from
LND_SM_PA_unix_uni uni
```

```
#=====================================================================#
#  MPmon Finished with ERROR(S) on: Thu Jun 20 22:27:06 CDT 2019
#=====================================================================#

CHECK_TABLE: *****ERROR: Data in SR_SM_DISK is 36076 minutes old, MAX ta_period is 2019-
06-18 21:20:00
CHECK_TABLE: *****ERROR: Data in SH_SM_DISK is 36096 minutes old, MAX ta_period is 2019-
06-18 21:00:00
CHECK_TABLE: *****ERROR: Data in SD_SM_DISK is 37356 minutes old, MAX ta_period is 2019-
06-18 00:00:00
CHECK_TABLE: *****ERROR: The database table SR_VI_VM exists but contains no ta_period
data.
CHECK_TABLE: *****ERROR: The database table SH_VI_VM exists but contains no ta_period
data.
CHECK_TABLE: *****ERROR: The database table SD_VI_VM exists but contains no ta_period
data.
```

Thank you for using MPmon!