mdtmGUI
Installation & Configuration Manual

Version 1.0.1

By Fermilab Network Research Group
Oct 2016

About

DOE's Advanced Scientific Computing Research (ASCR) office has funded Fermilab and Brookhaven National Laboratory to collaboratively work on the Multicore-Aware Data Transfer Middleware (MDTM) project (http://mdtm.fnal.gov). MDTM aims to accelerate data movement toolkits on multicore systems.

The MDTM research team has released several MDTM-related software packages – mdtmFTP, mdtmBBCP, mdtmGUI, and MDTM middleware (http://mdtm.fnal.gov/Releases.html).

- mdtmFTP and mdtmBBCP. mdtmFTP and mdtmBBCP are two MDTM-based data transfer tools. Both tools adopt an I/O-centric architecture, and use MDTM middleware to fully utilize the underlying multicore system.
- MDTM middleware. The middleware will hardness multicore parallelism to scale data movement toolkits on Data Transfer Nodes (DTNs). It schedules and assigns system resources based on the needs and requirements of data transfer applications (i.e., data transfer-centric scheduling). It also takes into account other factors, including NUMA topology, I/O locality, and QoS.
- mdtmGUI. mdtmGUI is a web-based DTN monitoring tool. It is able to provide online & real-time monitoring of DTN system status and configurations, online & real-time monitoring of data transfer status and progress, and online & real-time monitoring of the MDTM-based data transfer tool's status and configuration.

mdtmFTP, MDTM middleware, and mdtmGUI are researched, designed, and developed by Fermilab network research group.

mdtmBBCP is designed, and developed by BNL Computational Science Center.

This document describes the installation and basic use of mdtmGUI.

mdtmGUI is developed by Lauri Loebel Carpenter (lauri@fnal.gov).

The MDTM project PI is Dr. Wenji Wu (wenji@fnal.gov).

Intended Audience

This manual is intended for users and system administrators responsible for installing, running, and managing DTNs.

The manual assumes familiarity with multicore and DTN concepts.

System level requirements

1) System must have installed node.js. The mdtmGUI software has been developed/tested with node.js v0.12.8, which was the latest greatest release until rather recently (https://nodejs.org/en/download/releases/). The system installation of node.js should include the following modules installed globally (aka, installed by "root" via the command "npm install –g"):

```
$ npm -g ls --depth=0
/usr/local/lib
├── bower@1.4.1
├── cssunminifier@0.0.2
├── forever@0.14.1
├── npm@2.7.5
└── sails@0.11.0
```

- Download and install Node.js v0.12.8 software
  - Nodejs source code package can be downloaded from https://nodejs.org/en/download/releases/.
  - Or you can download Node.js v0.12.8 RPM packages from https://rpm.nodesource.com/pub_0.12/
    (Note: depending on your OS, you can go to folder "el/5", "el/6", "el/7", and "fc", respectively.)

- As root, run and install the following nodejs modules:
  root# npm install –g bower@1.4.1
  root# npm install –g cssunminifier@0.0.2
  root# npm install -g forever@0.14.1
  root# npm install –g sails@0.11.0

- Create a user account to run mdtmGUI
  root# useradd mdtmgui

2) System must be running monitorix (http://www.monitorix.org/). The following monitorix configuration (usually /etc/monitorix.conf) should be edited:

- PROC graph stanza must be edited for the maximum number of CPUs
- NET graph stanza must include all Ethernet NICs of interest
- FS graph stanza must include all mount points (file systems) of interest
- Account running mdtmGUI must be able to read the monitorix.conf configuration file, as well as the output monitorix rrd files.
- As root, restarting monitorix by running
  root# cd /etc/init.d
  root# monitorix restart

3) The following Linux command utilities must be installed and available to the account (e.g., mdtmgui) running the mdtmGUI server:

   - Numactl (http://oss.sgi.com/projects/libnuma/)
   - Lstopo, a hwloc application (https://www.open-mpi.org/projects/hwloc/)
   - Lshw (https://github.com/lyonel/lshw)
     - Lshw version 2.14 has bugs. The latest lshw version is 2.17. You can download PRM package from https://www.rpmfind.net/linux/RPM/dag/redhat/el7/x86_64/lshw-2.17-1.el7.rf.x86_64.html
   - Ethtool

4) The account "mdtmgui" that is used to run the mdtmGUI application must have "sudo" access to the "lshw" command. (This includes any developer who runs the server while developing/testing code).

   Please add the following lines to "/etc/sudoers":

   mdtmgui ALL=NOPASSWD:/usr/sbin/lshw
   Defaults:mdtmgui !requiretty

Running the mdtmGUI server

1) Login as user “mdtmgui”

2) Download mdtmGUI source code from http://mdtm.fnal.gov/Release.html

3) Unzip the downloaded software package

4) cd into the directory where you unzip the mdtmGUI code
   $ cd mdtmGUI

5) Run "*npm install*" (maybe more than once?) to install all of the node.js modules that are used locally within the package.

6) Test the server by running
   $ sails lift [--port 8080]

   Make sure that the server can start (that is, do the above "sails lift" command interactively and watch the output to make sure that there isn't any sort of fatal start-up error).

   Once it is established that the server is able to start without crashing, click Ctrl+C to exit.

7) Start the server by running
   $ forever start app.js [—-port 8080] # to make sure server restarts on crashes

   Note: forever is a node.js application, https://github.com/foreverjs/forever

Running modes

mdtmGUI is designed and developed for MDTM-enabled DTNs. However, it can also be deployed to monitor and manage any networked computer systems.

An MDTM-enabled DTN is a DTN that runs an MDTM-based data transfer tool (e.g., mdtmFTP), along with MDTM middleware.

For an MDTM-enabled DTN, mdtmGUI provides a full set of functions:

- Online & real-time monitoring of system status and configurations
- Online & real-time monitoring of data transfer status and progress.
- Online & real-time monitoring of MDTM-based tool's status.

When mdtmGUI is deployed to monitor a Non-MDTM-enabled computer system, it provide a reduced set of functions:

- Online and real-time monitoring of system status and configurations.

CODE ORGANIZATION

- mdtmConfig.js

  This is the main configuration file, needs to be edited and then the server restarted. Lots of things are configured in here -- the colors to use, the number of "cores" to show on each plot on the "System> CPU Load" page, the paths to the CSV files that are written by the MDTM software for data transfer statistics, etc. The default mdtmConfig.js should be fairly self-explanatory, with full examples indicating required syntax, etc.

  When mdtmGUI runs, the “Configuration -> User Config" and “Configuration->Full Configuration” tabs show the contents of the configuration file.

  When mdtmGUI is deployed to monitor and manage non-mdtm-enabled computer systems, all MDTM features should be disabled by setting “$MDTM_MONITORING_ENABLED” to “false”.

- api/controllers/
  - MainController: handle the "index.ejs" front-page web requests
  - AjaxController: handle the main "nav panel" web requests
  - RrdController: handle the ajax callbacks related to obtaining monitorix “rrd fetch” data
  - SnapshotController: handle "the rest" of the ajax callbacks, which generally get a “snapshot" of "how some command output looks right now”

- api/services/
  - mdtmConfigService: called during startup, this builds a $CONFIG data structure of All Known Static Data (including the $USER_CONFIG information from the mdtmConfig.js file, and parsing the output of several other commands to obtain static information such as hardware, memory, colors to use, etc.).

  xxx.js: naming convention should make it fairly clear what each of the files in this area do.

- assets/
  - Javascript source code/scripts, css style sheets, images, jQuery packages, etc.

- views/
  - main/index.index.ejs - the "front page".
  - ajax/*.ejs - the "container" template for individual pages.
  - ajax/templates/*.ejs - the individual templates sucked into each "container".

MDTM-SPECIFIC DATA

When an MDTM-based data transfer tool (e.g., mdtmFTP) runs, it typically generates 3 CSV (comma-separated-value) files, as well as individual file-transfer-progress RRD files. These files are typically located at folder "/tmp". mdtmGUI accesses these files to render data transfer status and progress.

The 3 CSV files are:

- /tmp/mdtm-service.csv. It records the data transfer tool's running status and configuration.
- /tmp/mdtm-snapshots.csv. It records ongoing data transfer tasks.
- /tmp/mdtm-completed.csv. It records completed data transfer tasks.

Each record in "mdtm-snapshots.csv" and "mdtm-completed.csv" corresponds to a particular data transfer task, and includes a pointer to the RRD file specific to that transfer.

The CSV file locations are specified in the mdtmConfig.js configuration file.

If you want these CSV files to contain a header line describing the data, it must be a COMMENT line (i.e., preceded by a '#' character).

File transfer data (for mdtm-snapshots.csv and mdtm-completed.csv)

- Transfer_current, transfer_history: each record contains the following fields:
  - ***ts:*** current timestamp
  - ***sid:*** system ID (numeric) of the process
  - ***uid:*** user ID (numeric) of the user
  - ***t0:*** timestamp at the start of the file transfer
  - ***t1:*** current timestamp (for current transfer), or timestamp when the file transfer completed
  - ***src:*** numeric IP address for the file transfer source node
  - ***dst:*** numeric IP address for the file transfer destination node
  - ***size:*** number of bytes in the file
  - ***xfr:*** number of bytes already transferred at time t1
  - ***rate:*** rate as calculated by MDTM for the file transfer so far
  - ***rrd_file:*** full path specification to an RRD-formatted file containing transfer details
- Each rrd file (one per file xfr) contains individual records persisted over the duration of the file transfer. For each timestamp in the file, the RRD columns are:
  - ***sid:*** system ID (numeric) of the process
  - ***uid:*** user ID (numeric) of the user
  - ***src:*** numeric IP address for the file transfer source node
  - ***dst:*** numeric IP address for the file transfer destination node
  - ***size:*** number of bytes in the file
  - ***xfr:*** number of bytes transferred at this timestamp

Service Status data (mdtm-service.csv)

- Each record contains the following fields:
    - ***threadID:*** integer, identifier of the MDTM thread
    - ***type:*** integer, identifier of the thread-type
        - 0=disk, 1=network, 2=management
    - ***numa:*** integer, the NUMA node number
    - ***core:*** integer, cpu core number
    - ***device:*** string, indicates the physical device

Feedbacks

If you encounter any problems when installing, configuring, and running mdtmGUI, please send emails to Dr. Wenji Wu (wenji@fnal.gov) or Ms. Lauri Loebel Carpenter (lauri@fnal.gov).