

DPMon.doc

Copyright (C) 2009 DSRLab, LLC - All rights reserved.

DPMon - an independent distributed performance monitoring system.

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <<http://www.gnu.org/licenses/>>.

Contact: Alex Vrenios via email at alex@DSRLab.com or, by mail at
DSRLab, LLC
1565 E. Marconi Avenue
Phoenix, AZ 85022-3248 USA

Overview:

The user starts dpmon on a network node that is not part of the distributed processing effort. That is, the "cluster" comprises a master and several slave nodes on a network. An extra node which is also on the network may be used as a distributed performance monitor. There is no support for IPv6.

The dpmon process starts dpmond, the dpmon daemon, as an xinetd service (via broadcast on UDP port 6556) on each of the cluster's active nodes (the master and all of the slaves). Every dpmond process collects performance data from its node once each second and transmits a summary packet to dpmon, the monitor process. The dpmon process is the monitor and it displays the performance data. Use Ctrl-C to end the monitor; this will in turn kill the remote daemon processes.

Information Source:

The /proc file system contains several pseudo files in text format. The kernel updates these files regularly with CPU and memory utilization, and disk and network I/O counts. A problem with these files is that every release seems to present them in a slightly different format. Changes in format have a significant impact on the remote statistics gathering process, dpmond. Print these file and adjust the source code in dpmond.c according to whatever format you happen to have.

Acquisition Method:

CPU and real memory utilization requires a simple calculation of the current performance data. Disk and network I/O, however, are calculated from the difference of two values over a 1-second interval. Disk I/O totals from the 1st second are subtracted from totals in the 2nd second's data, yielding disk I/O's per second which is sent to the monitor, dpmon, in each packet.

Report Format:

A network address of 192.168.0.0, along with a netmask of 255.255.255.0 is assumed. The last digit (node number) of the IP address is used as an index into a list of host names that are displayed along with their performance statistics, as shown in the following example:

	C_PCT	M_PCT	DSK_R	DSK_W	NET_R	NET_W
felix	25	11	0	2	1	1
chris	16	10	0	0	1	1
robby	20	11	0	0	1	1

Note that CPU_% is the sum of user, nice and system mode utilization. MEM_% is the percentage of physical memory allocated. Disk reads and writes, and network reads and writes are simple counts. All of these values represent activity over the last "one second" of operation. You may wish to modify this default format, and you are encouraged to do so.

Further Details:

Most of the details are presented as constants and variables in dpmon.h, an include file. Note that xinetd services are supported by (1) adding a line entry to /etc/services, and (2) adding a small text file (named dpmon) to /etc/xinetd.d on each of the cluster's active nodes. You must restart xinetd once services is modified and the dpmon text file is installed, issue

```
#service xinetd restart
```

Be sure xinetd is installed (see rpm command) and that it is active in your run level (per the chkconfig command).

Use tar to install the source and include file in a single subdirectory. Use make to compile the two source files (dpmon.c and dpmond.c), creating the two executables (dpmon and dpmond) in your local bin subdirectory, which should be on the same level as the source: File makefile uses

```
-o../bin/dpmon and -o../bin/dpmond
```

so as not to interfere with the dpmon text file used by xinetd. Leave the include files (dpmon.h and copyright.h) in the same subdirectory as the two source files and the make file.

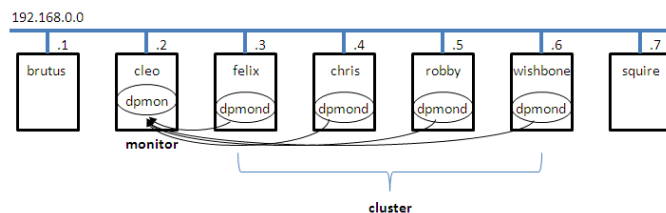
Configuration:

DPMon runs on the monitoring node, under a user ID of your choosing. It is assumed that the user has a "src" and a "bin" subdirectory, and that bin is in its execution pathway. Change into the src subdirectory and issue "tar -zxvf DPMon.tgz" to extract the files, as follows:

```
dpmon.txt      xinetd configuration file: as root, issue
                cp dpmon.txt /etc/xinetd.d/dpmon
dpmon.c        DPMon monitor process source
dpmond.c       DPMon remote process source
dpmon.h        DPMon include file
license.h      GNU public license
makefile       make utility file
copyright.h    DSRLab, LLC copyright
readme.txt     This text file
```

0. Correct the hex broadcast definition in dpmon.c if necessary
1. Compile dpmon and dpmond via the make utility
2. Change the user ID to whatever you chose, in file dpmon
3. Correct the fully qualified path name to the dpmond executable, in file dpmon.txt. Note that this path is assumed to be available to all active nodes in the cluster, perhaps through nfs mount (the Network File System), or through some other means.
4. If UDP port 6556 is unsuitable for your environment:
 - a. Change the value in dpmon.h, the include file
 - b. Change the value in dpmon.txt, the xinetd service config file
5. On every node to be monitored (as root):
 - a. Install dpmon.txt under /etc/xinetd.d/ as dpmon (drop the .txt)
 - b. Add line "dpmon 6556/udp" to /etc/services (change 6556, if desired, per task #3 above)
 - c. Issue "service xinetd restart" after tasks a) and b) above
6. Start dpmon on the monitor node and confirm that each of the nodes to be monitored has registered its presence with the monitor process, dpmon.
7. Type ctrl-C to terminate the remote performance processes and exit the monitor.

It is improper to define an xinetd service on the same node as the monitor. The monitor should run on a node on the same network as the cluster, but not be an active member of it.



Alex Vrenios
DSRLab, LLC
Phoenix, AZ