# SFS Troubleshooting and Monitoring Tools

**Roland Laifer**

**Scientific Supercomputing Centre (SSCK)**
**University of Karlsruhe**
**Germany**

**Laifer@rz.uni-karlsruhe.de**

# Abstract

- **SFS problem investigation tools**
  - **`show_name` translates SFS client and server IDs into readable names**
  - **`job_scan` displays batch job history of a dedicated user, node or time**
- **SFS performance monitoring tools**
  - **`show_server_perf` shows current SFS server performance**
  - **`show_client_perf` identifies current jobs with high I/O performance**
- **SFS quota handling tools**
  - **`show_quota` displays a user's quotas in easy readable format**

# show_name

- **Goals**
  - Translate SFS client IDs of evlview output into readable names
    - IDs are InfiniBand IP addresses (some in hex format) or Quadrics NIDs
  - Translate SFS server IDs of client log files into readable names
- **Input data**
  - Site specific file with IP addresses or NIDs and corresponding names
  - Log messages which contain IP addresses or NIDs
- **Example**

```
# evlview -m -f 'age<1' | grep -i lustreerr | ~/bin/show_name -i ib
Apr 23 11:09:13 xc2-ls1 kernel: LustreError:
   6358:0:(vibnal_cb.c:1816:kibnal_close_conn_locked()) Closing
   conn to 0xa16019b: error -116 rx# 129905 tx# 135598 (0xA16019B
   is xc2n67)
```

# job_scan

- Goals
  - Identify a corresponding job if a node had a problem
    - e.g. if logcheck email contains a Lustre error with node name and timestamp
  - Identify all jobs which have been running at a dedicated time
    - e.g. to find out which user possibly caused slow file system response times
- Input data
  - Our site specific batch system trace file
    - Most batch systems have job accounting files with the required information
  - Node name, user name, job ID, or date and time on command line
- Example

```
# job_scan -d 2007/04/25,14:46:00 -p xc2n460
Jobid  User           Start             End  Status  Nodes
42081   bob  04/25,14:45:02  04/25,14:46:07      ok  xc2n[449-464]
```

# show_server_perf

- **Goals**
  - Display current SFS server I/O performance on a terminal
    - Alternative is using a web browser which does not display current data
- **Input data**
  - HP's tool collectl gathers Lustre server performance data below /proc
    - `show_server_perf` simply calls `collectl -sl -oh -c1 -i1`
- **Example**

```
# ./show_server_perf
          #mdsCls Getatt  Reint   sync
xc2-ls1:        2      2      1      0
xc2-ls2:       34      2      2      0
          #KBRead  Reads KBWrit Writes
xc2-ls3:        0      0     10      4
xc2-ls4:        0      0   1048     11
xc2-ls5:       47      6 133597    159
xc2-ls6:        0      0  52448     55
```

# show_client_perf

- **Goals**
  - Identify which users create high throughput and metadata performance
- **Input data**
  - HP's tool collectl gathers Lustre client performance data below /proc
    - Use pdsh to start collectl on all nodes
    - Use collectl's exception limits LusKBS and LusReints to find nodes with high I/O values
    - Use batch system commands to show current jobs on these nodes
- **Example**

```
# ./bin/client_perf.sh
job-id   user    ...  node
-------------    ...  ----
 42074    bob    ...  2
 42380    bob    ...  484
 41905    joe    ...  598-603
```

# show_quota

- **Goals**
  - **Display quotas of the current user group for both file systems**
    - **Shows easy readable output**
    - **Explains required site specific actions if quotas are exceeded**
  - **Job management system calls `show_quota` during job submit**
    - **Displays a warning if usage rate is above 95 %**
    - **Aborts job submit if usage rate is above 98 %**

- **Example**

```
# show_quota -s
Quotas for group ssck:
File system /lustre/data:  500.00 GB quota limit,  183.39 GB used
   (36 %), 526256 files created
File system /lustre/work:  190.73 GB quota limit,    8.80 GB used
   ( 4 %),     27 files created
```