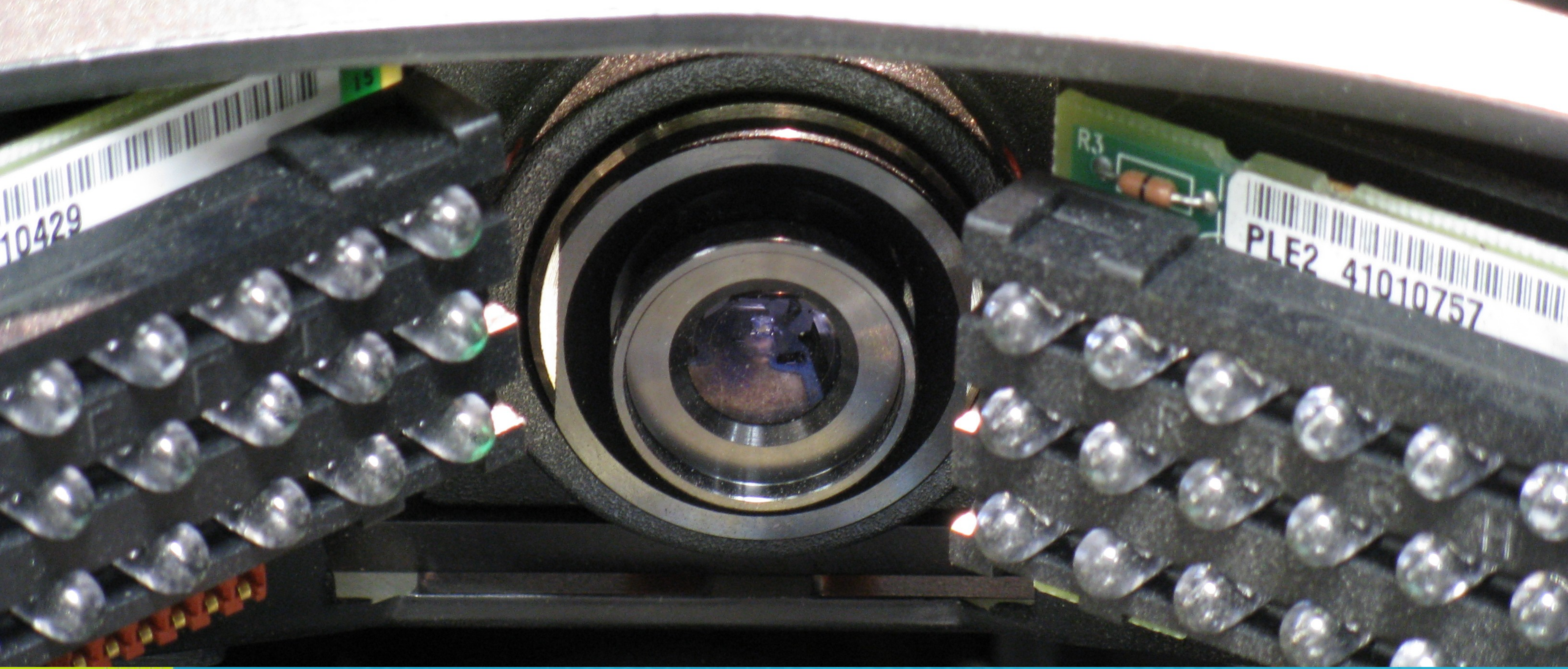

A *dmget* wrapper script

Peter Edwards
CSIRO Advanced Scientific Computing

CSIRO

Abstract

CSIRO requires some capabilities which are not provided as part of the standard DMF product, specifically:

- Rate limiting of file recalls to prevent accidental Denial of Service incidents
- ~~Population of a DMF cache (ie: a DCM), to be triggered by file recalls from tape by users~~
- Prediction of recall order

These unrelated features are implemented via a wrapper script around the standard “dmget” command.

Denial of Service - Problem

DMF processes recalls and related requests in a FIFO fashion

- **recalling a large number of files (eg: with *dmget */**) can block other users' recalls.**
- **existing recalls can't easily be canceled or reordered.**

But DMF's use of tape is highly optimised

Denial of Service - Original Solution

- **Sort recalls by file modification time and retrieve in batches of 40 files or 10GiB.**

- **Batches from one user may be interleaved with those from others.**

- **DoS problem solved.**

- **Inefficient - no knowledge of locality on tape:**
  - one batch may mount dozens of tapes
  - files on the same tape may be in different batches resulting in unnecessary tape mounts

- **Solving the wrong problem – number of files or GiB is not the issue, number of tape mounts is.**

Denial of Service - Current Solution

- **Discover which tape(s) each file requires via *dmcatadm -c*   (the “-c” is <u>important</u>!)**
- **Sort them by tape, allowing for files which straddle tapes.**
- **Normally, a batch is all the files to be recalled from a single tape.**
- **But allow for extremes such as:**
  - very few files per tape (eg: 1)
  - enormous number (>20k) on same tape
  - filling filesystem

Cache Population – Problem

- **Standard DMF only supports placing files in a DCM at migration time, via *SELECT_VG* directives.**

- **Files are deleted or moved out of DCM based on criteria like inactivity, space used and others.**

- **Nothing reinstates them when they again become active.**

- **Code in the *run_dcm_admin.sh* task script is experimental and doesn't scale as it doesn't run continuously.**

Cache Population - Original Solution

- **Migration doesn't indicate an intention to reuse a file soon, but we cache files of up to 2MiB anyway**

- **A recall does, and should therefore trigger caching**

- **A log tailing script spotted a successful recall, and cached the dual-state file using *dmmove,* if under 2GiB**

- **This was based on the fact that the disk copy of a dual-state file is used as the source for *dmmove,* provided you don't move by BFID**

Cache Population – Current Solution

- ***The "current solution" was found to be based on a false assumption, which in fact made it less efficient. There was also a slight possibility of it allowing a user to flush the cache. It was withdrawn and the log scraping script was reinstated.***

Recall Order - Problem

When recalling a large group of files in one operation, files recalled earlier in the group may revert to offline state before the user has had a chance to process them.

Use of the new *-a* option (equivalent to a prior *touch -a* of the files) may help, but this is not guaranteed.

Recall Order – Solution

Provide a new “-l” parameter which instead of recalling files, just lists them in the order in which they would have been recalled

This allows recalls to be done later in background while processing files in the same order in foreground:

```
$ dmget -l list of all the files > $TMPDIR/lof
$ dmget < $TMPDIR/lof &
$ for f in $(cat $TMPDIR/lof); do
>       process_one_file $f
> done
$
```

A Bonus

Some feedback to set expectations for interactive users:

```
$ dmget *
You are recalling 12 of the 19 files specified.
The oldest currently queued recall request
has been waiting for 0h 2m
5 tape mounts may be required.
$
```

Future Developments

- **Some users run multiple dmgets in parallel to gain extra service, which can block others**
- **This could be solved by a recall manager daemon, which could make scheduling decisions about the order in which files should be recall using criteria such as:**
  - number of files
  - amount of data
  - past history
  - priority
- **No, not Fair Share Scheduler!**

Future Developments (cont'd)

- **Investigate the *sitelib.so SiteKernRecall* “hook”. This allows kernel-generated recalls to be trapped and accepted/rejected.**

  If it can tolerate delays then a recall manager could use it, enabling the management of both implicit and explicit recalls resulting in a holistic view and control of the recall process.

  (I have since been advised that delays will result in the DMF daemon blocking; a bad idea.)

Thank you

CSIRO ASC
Peter Edwards

Phone: +61 3 8601 3812
Email: Peter.Edwards@csiro.au
Web: http://hpsc.csiro.au/users/dmfug/Presentations_Oct09/dmget_wrapper/

Thank you