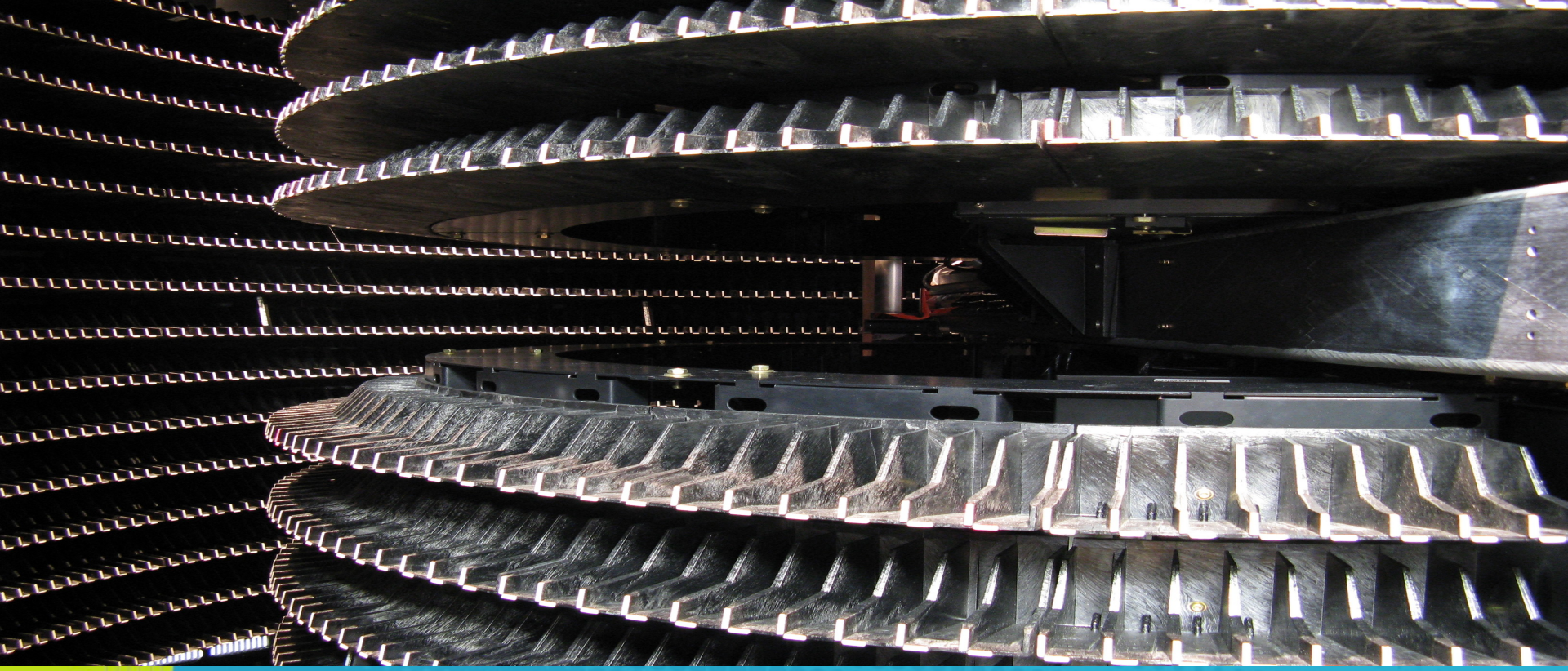


www.csiro.au

Load Sharing Recalls Between Multiple Volume Groups

Peter Edwards
CSIRO Advanced Scientific Computing

CSIRO

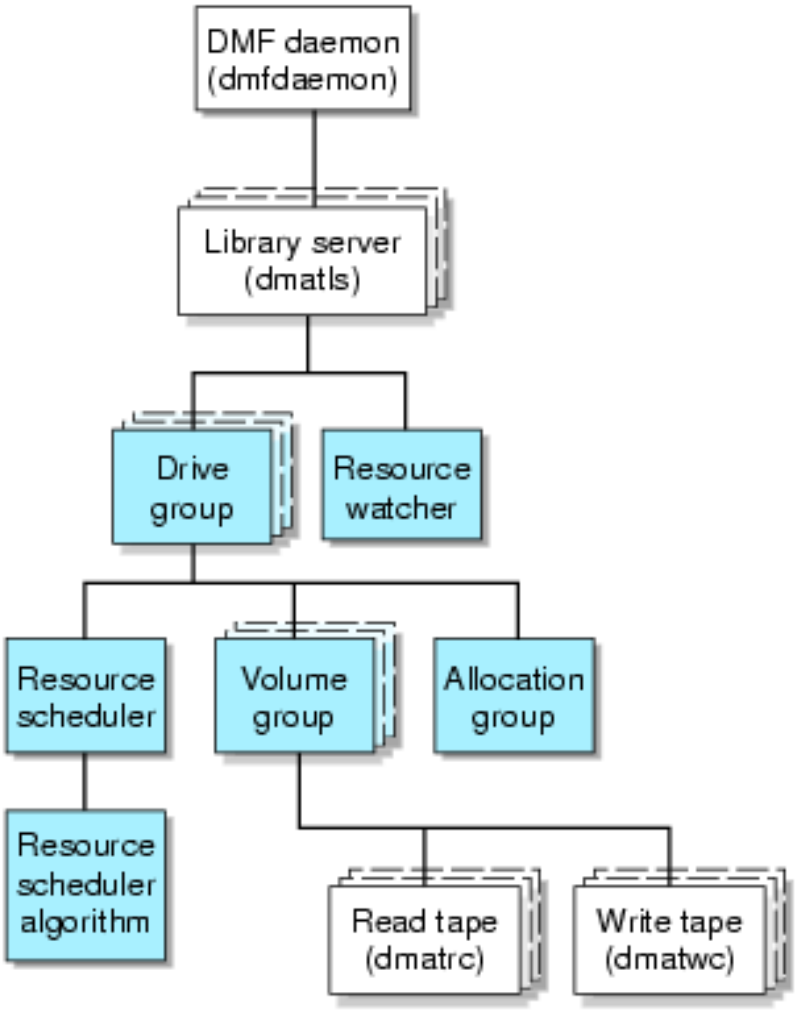
Abstract

- **It will be shown how to predict which tapes will be required by a volume group for future recalls and in what order.**
- **This capability will then be used to distribute recalls across multiple volume groups, thereby increasing the utilisation of the site's tape drives.**

Overview

- A quick review of the relationship between Volume Groups (VGs) and Drive Groups (DGs).
- How a VG recalls files from tape.
- How this is used to show the order in which tapes will be requested to satisfy currently queued recall requests.
- How both of these can be used together to transfer recall requests from a heavily loaded VG to another with idle tape resources.

Volume Groups and Drive Groups



How a VG uses tapes for recalls

- **Files are recalled from tape in units of one or more "chunks".**
- **When a VG receives a recall request for a file, it finds out the tape(s) on which the chunk(s) reside.**
- **If a chunk is on a tape currently being used for other recalls:**
  - its details are passed to the *dmatrc* process owning the tape, which adds it to the queue of other chunks for that tape.
- **If a chunk is on a tape not currently mounted for recalls:**
  - it's queued inside the VG, ordered by time.
  - When the VG gets permission to mount a tape, it chooses the tape required by the oldest chunk queued.
  - After the new *dmatrc* process has mounted the tape, it is passed details of all chunks to be recalled from that one tape, irrespective of their age.

How a VG uses tapes for recalls (cont'd)

- **When the final chunks for a file are read, the daemon is notified and entries are deleted from queues, even though the tape may still be in use for other requests.**
  - This may cause the impression that some recalls are "queue jumping" because they are not being processed in the order in which they arrived.
- **If *dmatrc* has a problem mounting or reading the tape:**
  - It informs the VG of the chunks that it couldn't process.
  - The VG passes details of the files containing those chunks on to the DMF daemon.
  - The daemon then reissues the file recall request to another VG if possible – the “secondary” VG for the file.

dmorder

The main purpose of *dmorder* is to allow you to answer queries like:

- **"How long do I have to wait?"**
- **"Who's doing the implicit recalls?"**
- **"Who's got the most recalls queued?"**
- **“Who's been waiting longest?”**

dmorder - sample output

```
# dmorder
Requests at 2009/09/25 14:48:00

Volume Group: sec (13 mounts)           Longest tape wait is 0:04:47
    *C56867 raf018[14:31:36, 14:31:36, 14:31:36, 14:31:36, 14:31:36, 14:31:36,
, 14:31:36, 14:31:36, 14:31:36] ngu038[14:31:36, 14:31:36, 14:31:36, 14:31:36,
    *C57239 mat236[14:47:35]
    *C57714 abb029[14:47:00]
    +R30630 lih[14:45:38]
    -C57054 kat024[14:43:23]
    -R30643 tha051[14:44:44]
     C56695 ste69f[14:45:58]
     C57729 sgisupport[14:46:27]
     C58404 car391[14:46:33]
     C57254 tha051[14:46:57, 14:46:57, 14:46:57]
     R30640 tha051[14:46:57, 14:46:57, 14:46:57, 14:46:57]
     R30617 tha051[14:46:57, 14:46:57, 14:46:57]
     R30486 tha051[14:46:57, 14:46:57, 14:46:57, 14:46:57, 14:46:57, 14:46:57,
, 14:46:57]

Volume Group: te2 (2 mounts)            Longest tape wait is 0:09:23
    *G61390 kat024[14:37:43]
    +G62204 srb001[14:38:47]

        1       abb029  Deborah Abbs,0392394660
        1       car391  Gary Carroll,0893336560
        2       kat024  Jack Katzfey,0392394562
        1       lih     Lawson Hanson,0396694763
        1       mat236  Richard Matear,0362325243
        7       ngu038  Kim Nguyen,0392394417
        14      raf018  Tony Rafter,0392394508
        1       sgisupport      SGI Support
        1       srb001  Jhan Srbinovsky,0392394577
        1       ste69f  Lauren Stevens,0392394552
        23      tha051  Marcus Thatcher,0392394540
Notes:
  - Shown for each file is its owner and the time of the recall request.
    Highlighted times indicate implicit recalls (green) or moves (blue).
  - Tapes in the same volume group will normally be used in the order shown.
  - Tapes in different volume groups have no relationship with each other.
  - Tapes which are already mounted are marked with a "*".
  - Tapes which are currently mounting are marked with a "+".
  - Tapes which are locked are marked with a "-".
```

dmorder – data sources

The data needed to predict future tape usage (for recalls and moves) comes from two places:

- **a slightly modified *dmstat* for:**
  - the DMF daemon's request queue
  - the VSN(s) of the tape(s) required for each request
  - the list of currently mounted/mounting tapes.
- ***dmvoladm* for the list of tapes with the *HLOCK* flag set (optional)**

dmorder – logic flow

- ***dmorder* groups recall/move requests by the tape(s) they will require and lists these tapes ordered by the age of the oldest request requiring them.**
- **Tapes which are currently mounted or mounting are shown ahead of the others, as they are in active use.**
- **VGs normally follow this order, but there is no guarantee. From time to time, for reasons which are not externally visible, it will mount a tape out of order.**
- **Another anomaly occurs when a file is to be recalled from a tape which is currently being used for migrations. This results in the recall blocking until the VG has finished writing to the tape.**

run_load_level – logic flow

- **Like *run_merge_mgr*, *run_load_level* lies in wait for idle drives**
- **It uses *dmorder* to determine the next few tapes to be used and sets their *HLOCK* flags**
- **The VG may attempt to mount these tapes, in which case it fails**
- **If so, the requests are returned to the daemon which reissues then to the secondary VG**
- **Either way, after a while, *run_load_level* clears *HLOCK***
- **Repeat indefinitely**

run_load_level – results

- **If the secondary VG's DG has 4 drives available, this script delivers the equivalent of about 3 extra drives to the recall process.**

- **When the rightful workload in that DG increases, *run_load_level* backs off.**

run_load_level - requirements

- **All files migrated to the targeted VG must have multiple copies.**
- **The tapes for the secondary copies must be in the silo.**
- **The two VGs concerned should be in different DGs or there's no point.**
- **You'll have to modify *run_scan_logs* to grep out all the extra error messages.**
- **You can only aim it at VGs which contain only primary copies.**

run_load_level - deficiencies

- **Sometimes it'll guess wrongly; but this does no harm.**
- **Because it relies on *dmstat*, which uses the Resource Watcher for almost all of its data, it will be unaware of non-DMF tape usage.**
- **Untested with OpenVault (but should work).**

run_load_level – deficiencies (cont'd)

- **Multiple target VGs not yet supported.**
- **Some configuration details not in *dmf.conf.***
- **As a convenience, *dmorder* shows details of the owners of the files being recalled, which it gets from the *passwd* file. No attempt has been made to add LDAP or NIS support.**

Conclusions

- ***dmorder*** **provides a useful tool to detect unusual patterns in users' recall activity, and to answer some common queries from the users (or administrators).**

- ***run_load_level*** **allows us to harness scarce drive resources which would otherwise lie idle during office hours.**

CSIRO ASC
Peter Edwards

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

Thank you