Fermilab U.S. DEPARTMENT OF ENERGY Office of Science

Intro to Data Management

Marc Mengel, **Pengfei Ding**

DUNE Software and Computing Tutorial

14th August, 2017

Overview of data management

- Storage volumes:

Storage systems	Path on GPVMs
BlueArc App	/dune/app/users/${USER}
BlueArc Data	/dune/data/users/${USER}; /dune/data2/users/${USER}
Scratch dCache	/pnfs/dune/scratch/users/${USER}
Persistent dCache	/pnfs/dune/persistent/users/${USER}
Tape-backed dCache	/pnfs/dune/tape_backed/users/${USER}

  - More volumes to be added (EOS at CERN, /pnfs at BNL etc.)
- Data handling tools:
  - *IFDH*
  - *SAM and SAM4Users*”
- Detailed info can be found via links provided the last slide.

Understanding storage volumes (I) - BlueArc

- BlueArc:
  - a Network Attached Storage (NAS) system;
  - App area, /dune/app
    - used primarily for code and script development;
    - should not be used to store data;
    - slightly lower latency;
    - smaller total storage (200 GB/user).
  - Data area, /dune/data or /dune/data2
    - used primarily for storing ntuples and small datasets (200 GB/user);
    - higher latency than the app volumes;
    - full POSIX access (read/write/modify);
    - not mounted on any of the GPGrid or OSG worker node;
    - throttled to have a maximum of 5 transfers at any given time.

DON'T USE BlueArc volumes in a grid job!

Understanding storage volumes (II) - dCache

- dCache:
  - A lot of data distributed among a large number of heterogeneous server nodes.;
  - Although the data is highly distributed, dCache provides a file system tree view of its data repository.
- Some facts users need to know:
  - dCache separates the namespace of its data repository from the actual physical location of the files;
  - the minimum data unit handled by dCache is a file.
  - files in dCache become *immutable*
    - Opening an existing file for write or update or append fails;
    - Opens can be queued until a dCache door (I/O protocols provided by I/O servers) is available (good for batch throughput but annoying for interactive use).

Understanding storage volumes (III) - dCache

Areas	Location	Storage type	Space	File lifetime	When disk/tape is full
Scratch	/pnfs/dune/scratch	Disk	No hard limit. Scratch area is shared by all experiments (>1PB as of today).	refer to the scratch lifetime plot: http://fndca.fnal.gov/dcache/lifetime/PublicScratchPools.jpg	LRU eviction policy, new files will overwrite LRU files.
Persistent	/pnfs/dune/persistent	Disk	190 TB	> 5 years	No more data can be written when quote is reached.
Tape-backed	/pnfs/dune/tape_backed	Tape	Pseudo-infinite	>10 years, Permanent storage.	New tape will be added.

Using dCache volumes at Fermilab (I)

- Detailed instruction can be found in DUNE wiki:
  https://cdcvs.fnal.gov/redmine/projects/dune/wiki/Using_DUNE's_dCache_Scratch_and_Persistent_Space_at_Fermilab

- Scratch dCache:
  - Copy needed files to scratch, and have jobs fetch from there, rather than from BlueArc
  - Least Recently Used (LRU) eviction policy applies in scratch dCache
  - Scratch lifetime: http://fndca.fnal.gov/dcache/lifetime/PublicScratchPools.jpg
  - NFS access is not as reliable as using ifdh, xrootd;
  - **Don't put thousands of files into one directory in dCache;**

  Note: Please do not use "rsync" with any dCache volumes.

Using dCache volumes at Fermilab (II)

- Storing files into persistent or tape-backed area is only recommended with “sam_clone_dataset” tool, or other tools that automatically declare locations to SAM.

- Grid output files should be written to the scratch area first. If finding those files are valuable for longer term storage, they can be put into the persistent or tape-backed area with SAM4users tool:
  - sam_add_dataset, create a SAM dataset for files in the scratch area;
  - sam_clone_dataset , clone the dataset to the persistent or tape-backed area;
  - sam_unclone_dataset , delete the replicas of the dataset files in the scratch area.

Hands-on session

- Required setups;
- Access files in scratch dCache:
  - Write, read and delete files;
  - Streaming files with xrootd in ROOT or art (applies to persistent dCache too).
- Store files to persistent or tape-backed dCache:
  - Declare a dataset with files in scratch area;
  - Clone the dataset to persistent or tape-backed area;
  - Remove replicas of the dataset in the scratch area;
  - Validate dataset and what to do when a file is missing;
  - Retire a dataset.
- Commands in this session can be found in Indico or here: https://cdcvs.fnal.gov/redmine/attachments/download/43161/dune_data_handling_tutorial_commands.txt

Setups

```
# On GPVM (e.g. dunegpvm01.fnal.gov)

# setup UPS etc.
source /cvmfs/dune.opensciencegrid.org/products/dune/setup_dune.sh

# Getting a valid certificate and VOMS proxy
kx509
voms-proxy-init --noregen -rfc -voms dune:/dune/Role=Analysis

# Setup fife_utils, current version is v3_1_0
setup fife_utils

# set experiment name
export EXPERIMENT=dune

# setup ROOT (not needed by data management itself, but we will use
# ROOT in this tutorial to show how to use files interactively).
setup root v6_08_06d -f Linux64bit+2.6-2.12 -q e14:nu:prof
```

Access file in dCache (I) – copy files to scratch

```
# Create a directory in scratch area for this tutorial
export SCRATCH_DIR=/pnfs/dune/scratch/users/${USER}/tutorial
ifdh mkdir_p ${SCRATCH_DIR}

# Write files to scratch dCache (best to have files written in local
# disk or BlueArc first and then copy to the scratch area with ifdh
# or xrootd)

# create four 5MB dummy files, these files will be used for
# demonstration of data handling. You do not need to create the dummy
# files. You can use files of your own.
for i in `seq 0 3`; do \
head -c 5242880 /dev/urandom > ~/dummy_${USER}_${i}.bin; \
done

# copy files into scratch dCache with “ifdh cp”.
ifdh cp -D ~/dummy_${USER}_[0-3].bin ${SCRATCH_DIR}
# To explore other options available with “ifdh cp”, just type “ifdh”.
```

Access file in dCache (II) – delete files in scratch

```
# delete files with ”ifdh rm”
ifdh rm ${SCRATCH_DIR}/dummy_${USER}_0.bin
for i in seq `1 3`; do \
ifdh rm ${SCRATCH_DIR}/dummy_${USER}_${i}.bin; \
done

# Copy files to scratch dCache using xrootd
xrdcp ~/dummy_${USER}_[0-3].bin ${SCRATCH_DIR}
# or
xrdcp ~/dummy_${USER}_*.bin \
root://fndca1.fnal.gov:1094//pnfs/fnal.gov/usr/dune\
/scratch/users/${USER}/tutorial

# note that one should convert the path to scratch dCache to URI
# recognized by xrootd:
# e.g. from: /pnfs/dune/scratch/users/${USER}/dummy_${USER}_1.bin
#        to: root://fndca1.fnal.gov:1094//pnfs/fnal.gov/usr/dune\
#         /scratch/users/${USER}/dummy_${USER}_1.bin
```

Access file in dCache (III) – streaming with xrootd

```
# Converting the path to xrootd URI using ifdhc
ifdh getUrl /pnfs/scratch/users/$USER/tutorial root

# copy a root file to scratch dCache (you can use your own root file,
# this file is used only for demonstrating streaming root file with
# xrootd in ROOT).
ifdh cp $ROOTSYS/tutorials/hsimple.root ${SCRATCH_DIR}

# access the file in dCache via xrootd in ROOT
root -l root://fndca1.fnal.gov:1094//pnfs/fnal.gov/usr/dune\
/scratch/users/${USER}/tutorial/hsimple.root

# art can also take the xrootd URI
```

Store files to persistent/tape-backed area (I) - declare a SAM dataset with files in scratch area

```
# choose a dataset name, better to be user, purpose and time specific
export TUTORIAL_DATASET=${USER}_tutorial_`date +%y%m%d%H%M`_01

# Add a SAM dataset for files in dCache scratch area
sam_add_dataset -n ${TUTORIAL_DATASET} -d ${SCRATCH_DIR}
# Instead of the “-d” option, it can take “-f” option followed by a
# text file containing a list of paths to files

# NOTE: sam_add_dataset will change the filename with UUID prefix.
ls ${SCRATCH_DIR}

# List files in the dataset
samweb list-definition-files ${TUTORIAL_DATASET}
```

Store files to persistent/tape-backed area (II) - clone the dataset to persistent/tape-backed area

```
# If the files under scratch area worth being kept for longer time,
# they can be added to SAM first with sam_add_dataset, followed by
# copying to the persistent or tape-backed area.

# create a destination directory in the persistent area first
export PERSISTENT_DIR=/pnfs/dune/persistent/users/${USER}/tutorial
mkdir -p ${PERSISTENT_DIR}


# Copy the dataset to persistent area with sam_clone_dataset
sam_clone_dataset -n ${TUTORIAL_DATASET} -d ${PERSISTENT_DIR}

# Advanced tips for cloning large dataset:
# “sam_clone_dataset” has ”--njobs” option to launch multiple jobs to do
# the cloning. “launch_clone_jobs” can lauch grid jobs to do the cloning.
```

Store files to persistent/tape-backed area (III) - remove replicas in the scratch area

```
# check file locations, you will see two locations.
DUMMY_01=`samweb list-definition-files ${TUTORIAL_DATASET}|head -n 1`
samweb locate-file ${DUMMY_01}

# Remove replicas of the dataset files in the scratch area
sam_unclone_dataset -n ${TUTORIAL_DATASET} -d ${SCRATCH_DIR}

# List ${SCRATCH_DIR} to check if files are still there.
ls ${SCRATCH_DIR}

# check the file locations again, you will see only one location left
samweb locate-file ${DUMMY_01}
```

Store files to persistent/tape-backed area (IV) - validate dataset and deadling with missing files

```
# Validate dataset, that is to check if each files in a dataset exists
# in the storage volume
sam_validate_dataset -n ${TUTORIAL_DATASET}

# Let's move one file in the dataset and run "sam_validate_dataset"
FPATH=`samweb locate-file ${DUMMY_01}|cut -d ':' -f 2`
ifdh mv ${FPATH}/${DUMMY_01} \
sam_validate_dataset -n ${TUTORIAL_DATASET}

# When there is a file missing, one can either replace the file with
# a backup copy; or use "--prune" option to remove the file from the
# dataset; otherwise there will be errors when using SAM record for
# file access.
sam_validate_dataset -n ${TUTORIAL_DATASET} --prune

# Let's list the files in the dataset again
samweb list-definition-files ${TUTORIAL_DATASET}
```

Store files to persistent/tape-backed area (V) - retire dataset

```
# This will delete the dataset definition in SAM, retire all files
# contained in the dataset and delete them from disk. To be safe, use
# this command with “-j” (“--just_say”) option first to see what will
# be done before letting it take real action.
sam_retire_dataset -n ${TUTORIAL_DATASET} -j

# You can use “--keep_files” option if you don’t want to delete the
# files.
sam_retire_dataset -n ${TUTORIAL_DATASET} --keep_files

# Once the dataset being retired, you can revert the file names for the
# last copy of files with sam_revert_names
sam_revert_names -d ${PERSISTENT_DIR}
```

Summary

- We have just gone through a full lifecycle of dataset files in the hands-on session;
- Please follow these practices in your own data management tasks, and keep the following things in mind:
  - Avoid using BlueArc area for grid jobs;
  - Avoid using “rsync” on any dCache volumes;
  - Store files into dCache scratch area first;
  - Always have files under persistent or tape-backed area bookkept by SAM;
  - Access files in dCache volumes via NFS is not as reliable as using “ifdh” or “xrootd”.

More info

- More info can be found in the following wiki pages:
  - Understanding storage volumes
    https://cdcvs.fnal.gov/redmine/projects/fife/wiki/Understanding_storage_volumes
  - SAM4Users wiki
    https://cdcvs.fnal.gov/redmine/projects/sam/wiki/SAMLite_Guide
  - SAM wiki
    https://cdcvs.fnal.gov/redmine/projects/sam/wiki/User_Guide_for_SAM