

dCache Status

Timur Perelmutov
for the dCache team
OSG Storage Seminar
Fermilab
June 31, 2009

Outline

- Dcache Features and Plans
 - New Features in dCache 1.9
 - Release Policies and Schedule
 - SRM Status and Plans
- Best Practices
 - Data Safety
 - Configuration Best Practices

New Features

- Reliable Deletion Registration
- XACML gPlazma plug-in
- Logging Context Propagation
- New Pool Code, Pool Migration Module
- Chimera
- Access Control Lists
- Info Service
- Gridftp Door

Reliable Deletion Registration

- Eliminates leak of pool space
- PNFS outage could result in false “File Not Found” errors
- Precious files could not be safely deleted by pool automatically
- Reliable Deletion Registration gives **authoritative positive** answer to question “Was file Deleted?” or “Is file safe to Delete?”
- **Requires** PNFS with Trash DB support

XACML gPlazma plug-in

- Authorization Mapping using GUMS and SCAS
- Example configuration

```
# XACML-based grid VO role mapping
xacml-vo-mapping="ON"
xacml-vo-mapping-priority="5"
XACMLmappingServiceUrl="https://fledgling09.fnal.gov:8443/
  gums/services/GUMSXACMLAuthorizationServicePort"
# Time in seconds to cache the mapping in memory
xacml-vo-mapping-cache-lifetime="180"
```

Log4j and Logging Context Propagation

- dCache moved to Log4j for logging
- New log4j management commands

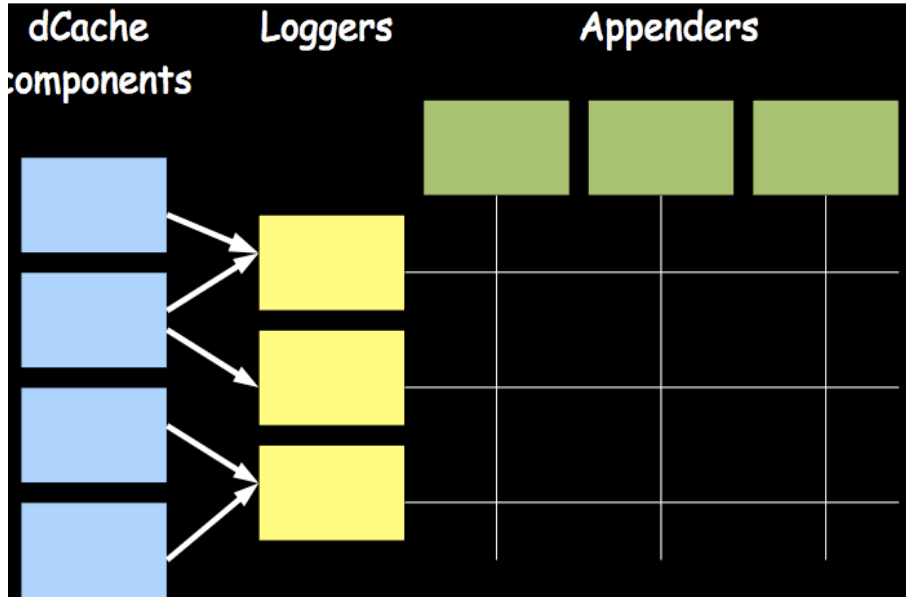
```
(System@dCacheDomain) admin > help log4j
log4j appender ls
log4j appender set <appender> -|OFF|FATAL|ERROR|WARN|INFO|DEBUG|ALL
log4j logger ls [-a]
log4j logger attach <logger> <appender>
log4j logger detach <logger> <appender>
log4j logger set <logger> -|OFF|FATAL|ERROR|WARN|INFO|DEBUG|ALL
```

- Logging Context Propagation

- Have you ever asked
 - What caused this Error Message Printout in Service A Log?
 - Are Error logs for services B and C Related?
 - propagates stack of names of services invocations leading to the logging of the given message
- ```
22 Jun 2009 14:36:39 (fap112_1) [v2:srmGet:17529671 PinManager
PoolSetSticky 000100000000000000002935850] java.lang.RuntimeException:
Internal repository error
```

# Log4j

(illustration by Gerd Berhmann, NDGF)



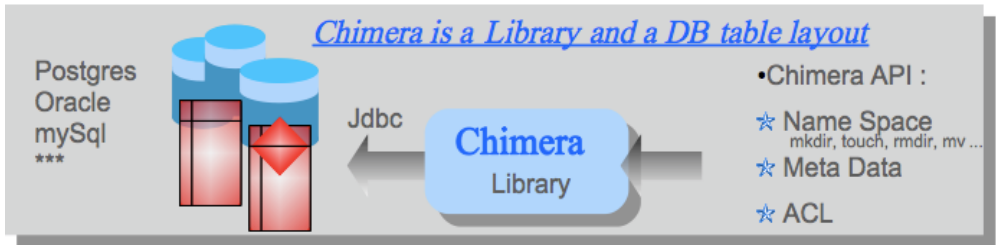
# New Pool

- Highly Modular
- Improved Pool Space Accounting
- Support for Berkley DB Pool Repository
  - Fast startup for large pools
  - Reduced memory usage
- Migration Module for copying files between pools
  - Flexible policies
  - Asynchronous
  - Preserves Sticky Flag, works with PinManager

# Chimera

- PNFS Namespace Replacement
- Features
  - Chimera is a database schema and a library for working with it
  - All abstractions are implemented as tables and relations instead of app. logic
  - PnfsManager, NFS3/4 demons talk directly to DB
  - Compatible with any JDBC enabled database including PostgreSQL and Oracle
- In production at Several German sites and at NorduGrid Tier1
- Prerequisite for NFS4.1 and some ACL features (inheritance)

Graphics by Patrick Fuhrmann



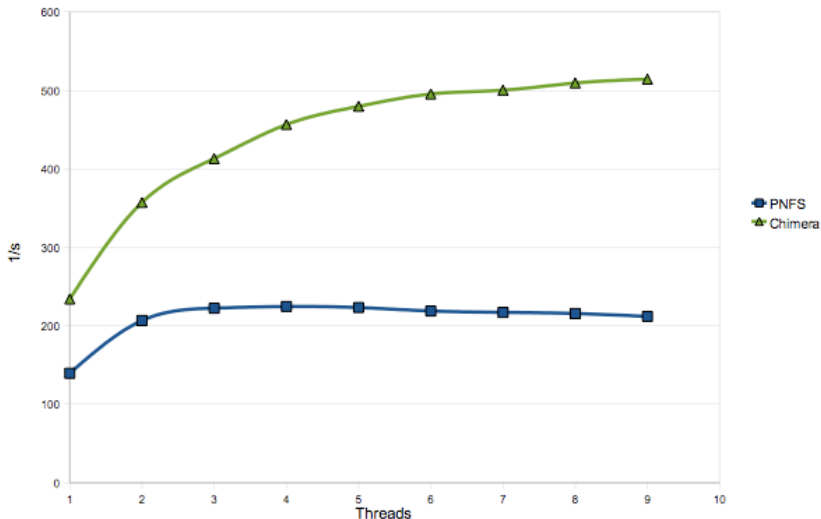
# Chimera Performance

- Scales with # of CPUs
- No Table level locks, improves parallelism
- One Transaction per one Namespace operation, hence no inconsistent states
- Allows sites to leverage in-house DBA expertise

# Scalability of Chimera

(Slide and measurements by Gerd Berhmann, NDGF)

pnfs id + storage info lookup



# Chimera Maintainability

- Usage of regular (customizable) SQL queries for file system maintenance : e.g.
  - space used by users / groups
  - which files have how many copies
  - limited only by the creativity of sys admin
- For backup and high availability services, the native db mechanism can be used.
- easy to use consistence checks against pool repository and tape repository (just SQL queries)

# Migration to Chimera

- Migration is described in detail at <http://trac.dcache.org/projects/dcache/wiki/pnfsDump2MigratePnfs2Chimera>
- Some German sites migrated without dcache.org support and did just find following the wiki page.
- For a regular Tier II migration can be easily done within a day.
- Migration should be tested beforehand to find possible show stoppers (see wiki page above)
- People should use the [support@dcache.org](mailto:support@dcache.org) if there are issues during the test migration.

# Disclaimer

- Chimera is the dCache namespace of the future
- PNFS will remain supported (Possibly by Fermilab part of the dcache collaboration only) until all supported sites migrate to Chimera.
- Certain advanced functionality will not be available in dCache+PNFS configuration
- PNFS is not a product in active development

# ACLs

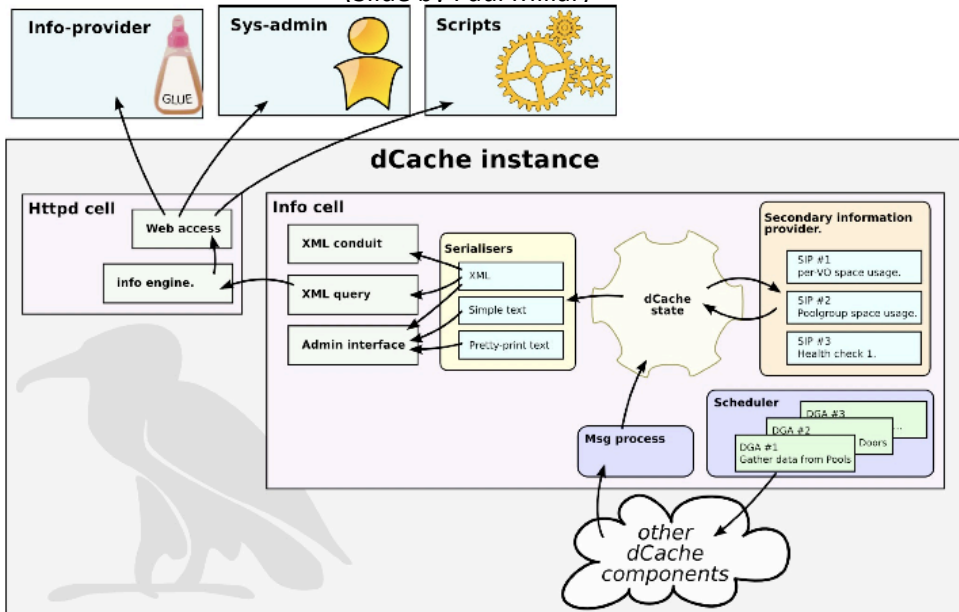
- Available in dCache 1.9.3 and higher
- subset of the NFS version 4 ACLs
- Used in conjunction with POSIX file permission mask
- ACLs are stored in the database on PnfsManager
- Work with Chimera (Full support) and PNFS (Only Subset of functions)
- SRM ACL is not yet implemented
  - ACL is enforced by the door after TURL is negotiated

# Info Service

- A robust, best-effort, “one-stop shop” overview of the current status of a dCache instance for external consumption.
  - Robust: the info service will continue to work independently of the rest of dCache.
  - Best-effort: there may be delays in information being updated (1 minute order-of-magnitude).
  - One-stop shop: you should be able to get all the information you require.
- It decouples updates from queries:
  - Querying is fast and robust

# Info Service

(Slide by Paul Millar)



# Gridftp V2

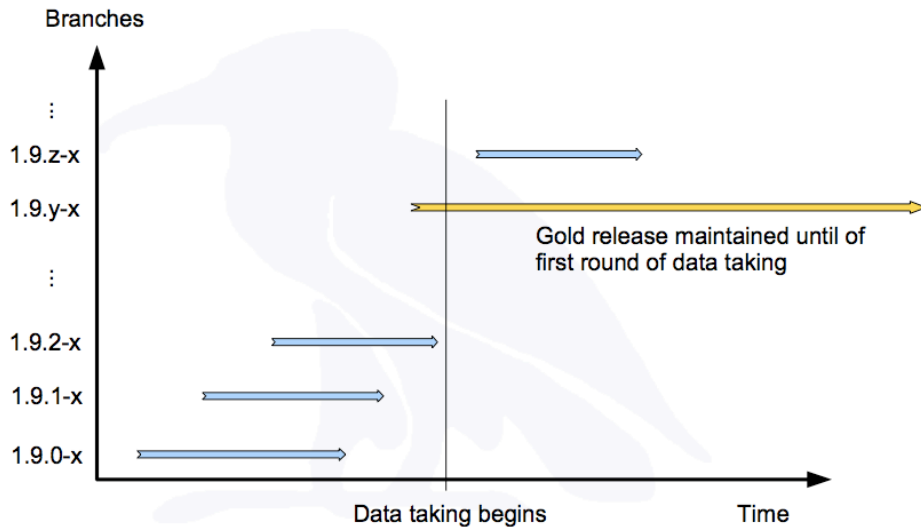
- CKSUM
  - Data Integrity
- MODEX
  - New mode for data transfer
- GETPUT
  - Allows specification of all transfer parameters in one command

# Release Policy

- 1.9.3 is available today
- 1.9.4, cut off on July 13, release a week later
- 1.9.5 TBD after 1.9.4 release
  - Golden release, for the duration of first round of data taking
- Time based releases
  - Feature releases are not bound for a particular release

# Release Policy

(Illustration by Gerd Behrmann)



# Features schedule

- ACL's (1.9.3)
- NFS4.1 (1.9.3) (beta and no security yet, not connected to gPlazma yet)
- Tape protection : tape access by production user (simple ACL's) (1.9.4)
- doors (esp. xrootd) will use light way mechanisms (smaller and after footprint) (1.9.5)
- Improved cost mechanism e.g. for p2p and writing into system. (1.9.5)
- NFS4.1 with kerberos and gPlazma (1.9.5)
- Experimental support for Terracotta powered clustered SRM (1.9.5)
- Merging of web pages and GUI as web application (in browser) (unknown)

# SRM status

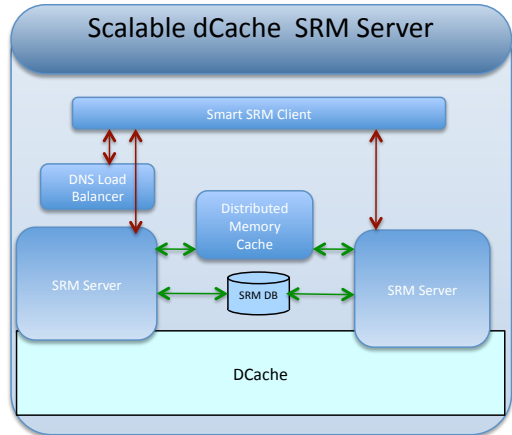
- (most) SRM functionality demanded by WLCG is implemented
  - Status and Error codes
  - Online state management by name
  - Space Management
- No new features until performance, stability, scalability and usability issues are addressed

# SRM Performance Improvements

- High CPU load due to GSI Authentication and Credential Delegation
  - cache public and private key pairs used in GSI authentication GSI handshake
  - work with Globus on improvements
  - consider https as a long term solution
- High volume of blocking unscheduled SRM LS operations cause high CPU load on SRM and PnfsManager and starvation of the network connection slots
  - convert SrmLs to a scheduled asynchronous operation
  - consider caching
- Large load due to the internal request timeouts and retries.
  - use lifetime attributes in internal dCache messaging
  - aggressively remove expiring requests
- High Load caused by client requests timeouts and retries
  - require clients to specify the request lifetimes (WLCG wide agreements covering all clients)
- User load can exceed system capacity
  - protection of the all dCache system resources by limiting total number of active operations of each type
  - return SRM Errors triggering exponential back-off of client request submission (WLCG wide agreements covering all clients)

# SRM Scalability

- SRM is a single point of entry into a storage
  - Natural bottleneck
  - Single point of failure
- Distributed SRM
  - Scalable
  - More reliable



# SRM Error handling and propagation

- SRM Is an interface to Storage
  - all storage errors are SRM errors
- Some errors can be handled automatically
  - example: PinManager may recover from temporary and permanent unavailability of data node
- Certain errors should not be retried internally
  - example: Data unavailable due to Tape error -> return error to the user immediately, not after 4 hours
- Propagate correct error messages though components, this may reduce investigation time and increase understanding
  - example: Data unavailable due to Tape error, return <UNAVAILABLE, "tape is in NOACCESS state"> instead of current <ERROR, "Lifetime expired">

# SRM Monitoring, Diagnostics tools and Documentation

- Providing more information about immediate state and usage statistics will increase understanding of the system and provide ability to detect and stop abusive usage
- Need to measure and graph all activity, and reasons for non-performance
- performance depends on proper configurations, defaults are too conservative for most systems
  - Proper documentation leads to better understanding of the configuration options by administrators

# Data Safety

- How should sites without large tape systems make sure their dCache files are safe?
  - Data on disk is not safe
  - End-to-end checksums guarantee integrity (you will know when you loose your data)
  - Resilient Manager decreases the probability of data loss by replication (is not compatible with Space Manager)
- What are the current best practices for data replication using the dCache Replica Manager?
  - Replica Count of 3 provides good preservation
  - Implement active checksum verification scans
  - Investigate Excluded files and files in Error State
  - Follow the Leader (US CMS T1)

# Configuration Best Practices (1)

## Namespace

- Namespace is the most critical component
  - Allocate proper hardware
  - DB disk is not shared
  - Follow PostgreSQL Configuration guidelines from <http://dCache.org> and <http://www.postgresql.org>
  - If PnfsManager queues lengths are routinely greater than 0, your system is overloaded

# Configuration Best Practices (2)

## PoolManager

- PoolManager Domain is also a dCache communication hub
- Do not deploy doors on PoolManager node
- Partition pool space as little as possible
- Make pools of different types be spread as wide as possible to increase hardware utilization in all usage scenarios

# Configuration Best Practices (3)

## Doors

- dCap doors are scalable, no need to have a large number
- Gridftp doors can be scalable with modern clients, but with most current clients they are not
  - Deploy multiple gridftp doors (one per pool node in US-CMS-T1)

# Configuration Best Practices (4)

## Pools

- System Disks separate from Data disks
- EXT3 used to suffer from fragmentation in near full state with active read/write delete
- XFS performed well for Fermilab Deployment
- Anecdotal info that EXT3 +Latest Kernel work
- Conservatively set number of movers
  - Separate queues for LAN/WAN access, dCap/ Gridftp protocols

# Configuration Best Practices (5)

## SRM

- Dedicated Host
- TOMCAT\_MAX\_THREADS is equal to number connected clients
  - Defaults are usually too conservative
  - Increase if you have CPU+MEM
- Server Socket Backlog parameter was set too low
  - New parameter TOMCAT\_ACCEPT\_COUNT in srm\_setup.env
  - at least 1024, Setting higher requires changing OS limits
  - Greatly reduces number of failed/abandoned requests due to connection failures

# Configuration Best Practices (6)

## SRM

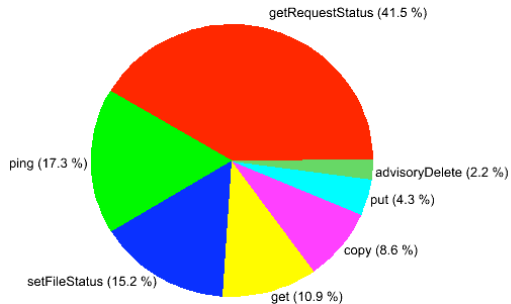
- 6 types of scheduled requests
  - Get, Put, Copy, BringOnline, LS, ReserveSpace
  - Each type has its own scheduler
- Understand Scheduler parameters
  - `Srm<Type>ReqThreadQueueSize`
  - `srm<Type>ReqThreadPoolSize`
  - `srm<Type>ReqMaxWaitingRequests`
  - `srm<Type>ReqReadyQueueSize`
  - `srm<Type>ReqMaxReadyRequests`
  - `srm<Type>ReqMaxNumberOfRetries`
  - `srm<Type>ReqRetryTimeout`
  - `srm<Type>ReqMaxNumOfRunningBySameOwner`

# Questions?

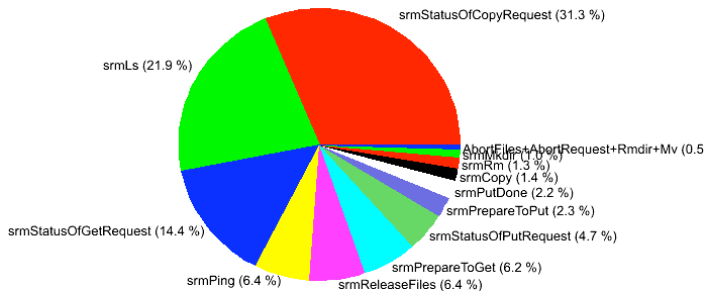
# Additional Slides

- SRM V1 Usage at FNAL
- SRM V2 Usage at FNAL
- V1 vs. V2
- SRM Execution graphs

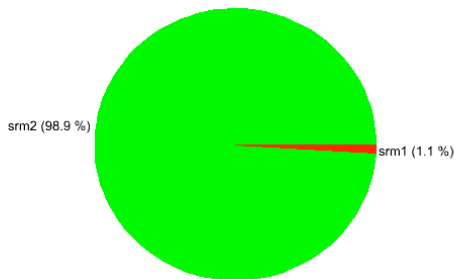
# US CMS T1 SRM V1 Usage (May 14, 2009)



# US CMS T1 SRM V2 Usage (May 14, 2009)

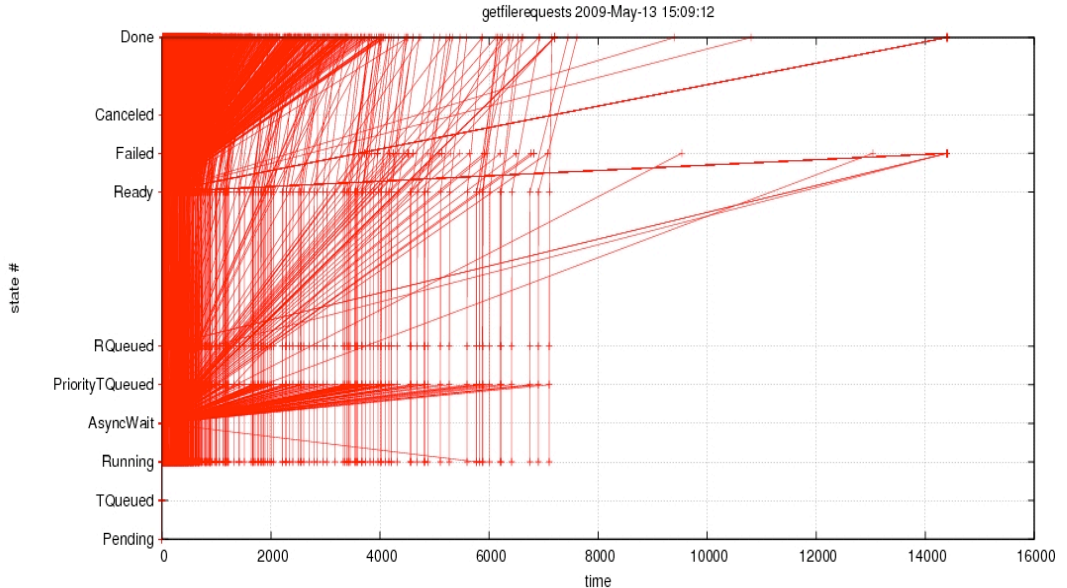


# SRM V1 vs. V@ Usage (May 14, 2009)

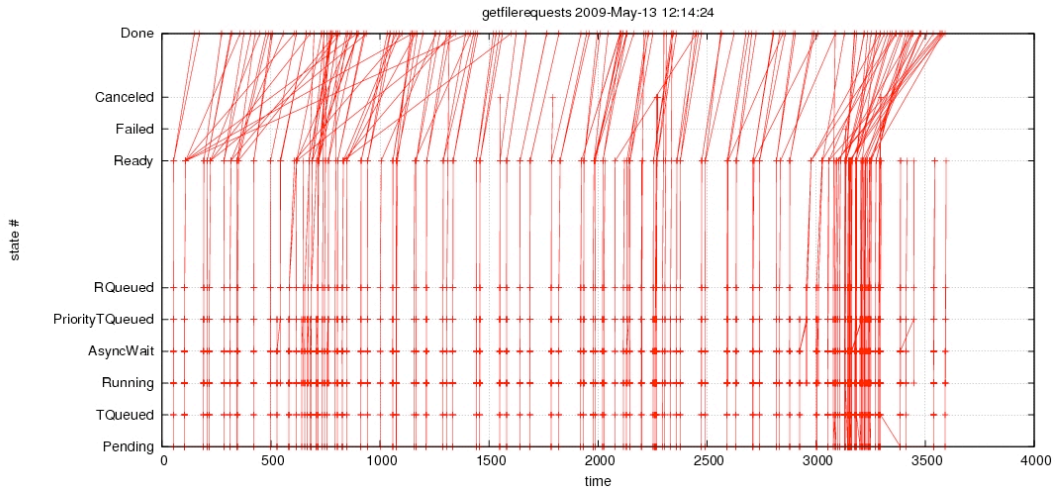


# Get dCache SRM Request Execution

Time is relative to the moment of submission

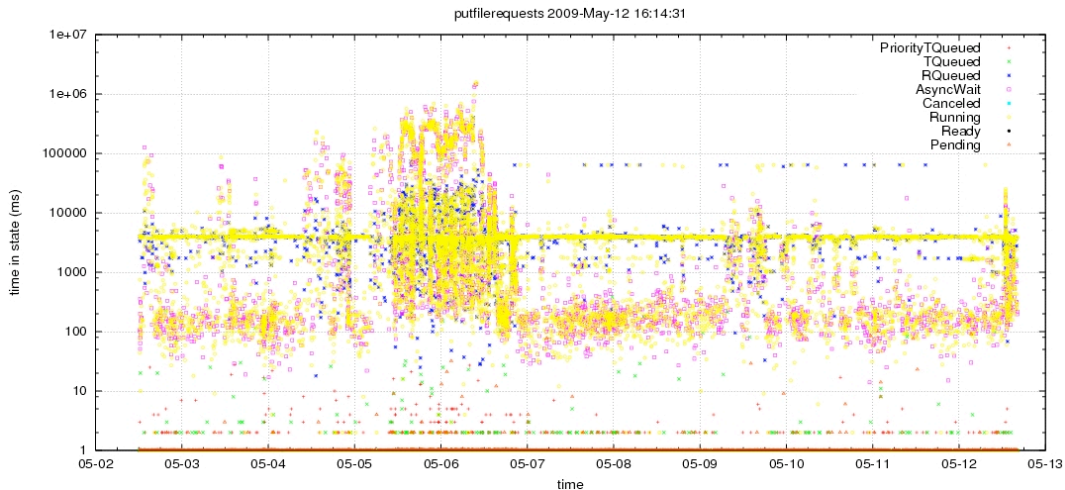


# Get dCache SRM Request Execution Absolute Time



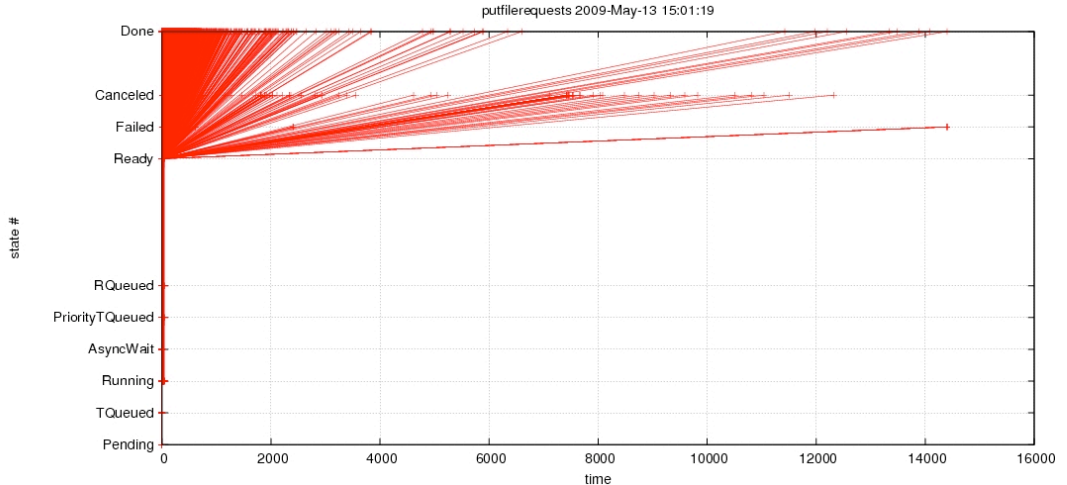
# Get dCache SRM Request Execution

## Time spent in different states

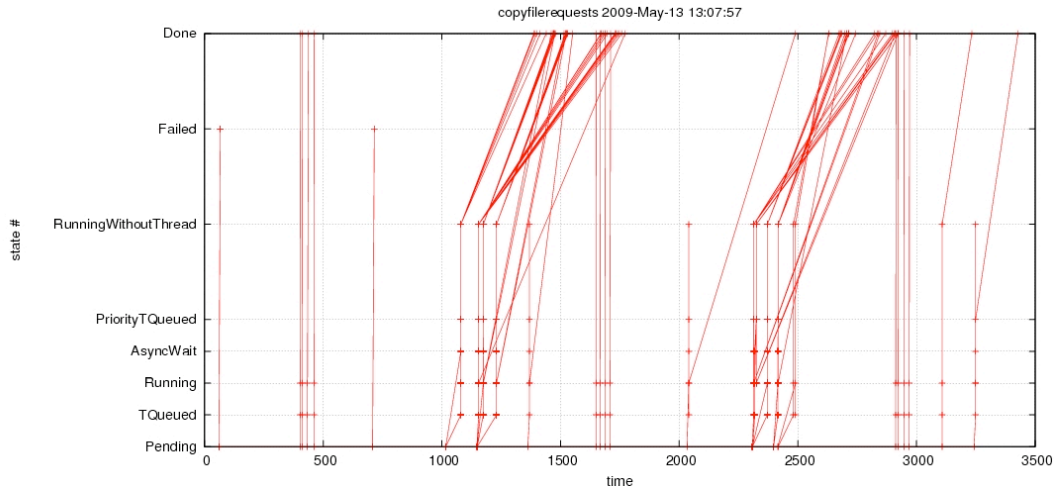


# Put dCache SRM Request Execution

## Time is relative to the moment of submission

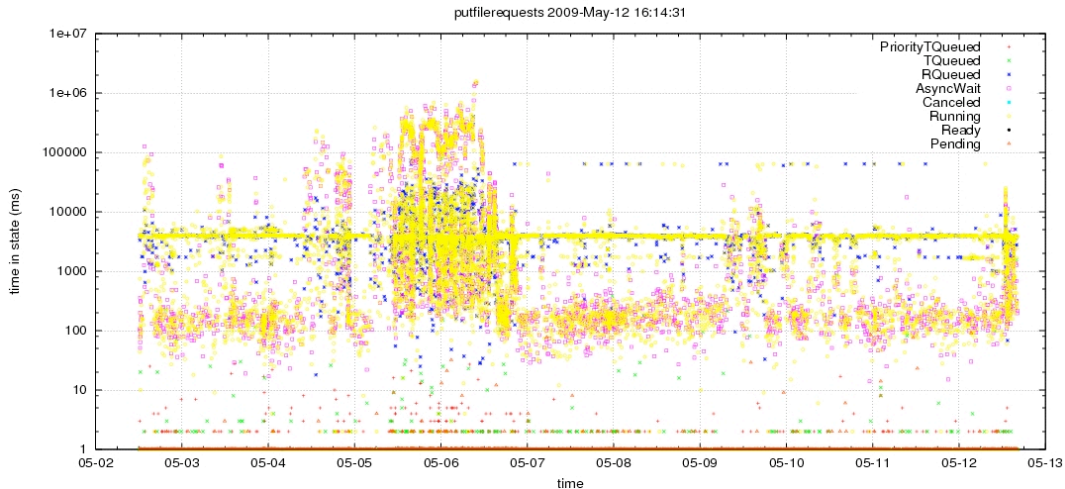


# Put dCache SRM Request Execution Absolute Time



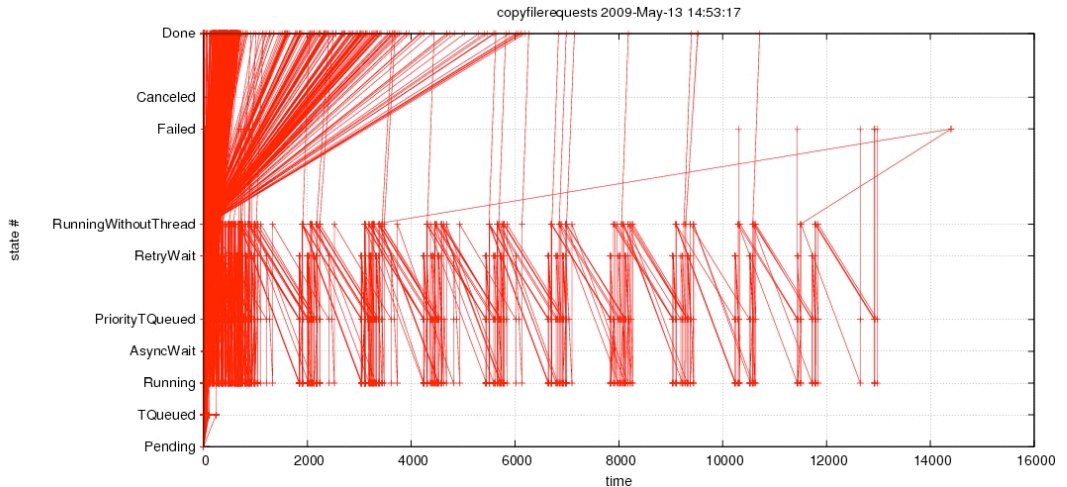
# Put dCache SRM Request Execution

## Time spent in different states



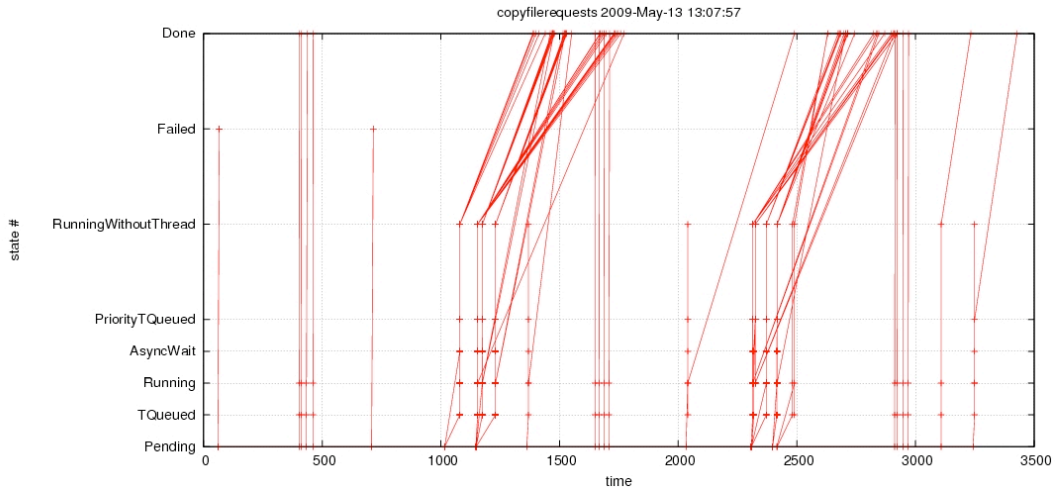
# Copy dCache SRM Request Execution

Time is relative to the moment of submission



# Copy dCache SRM Request Execution

Time is relative to the moment of submission



# Copy dCache SRM Request Execution

## Time spent in different states

