Xrootd-SRM

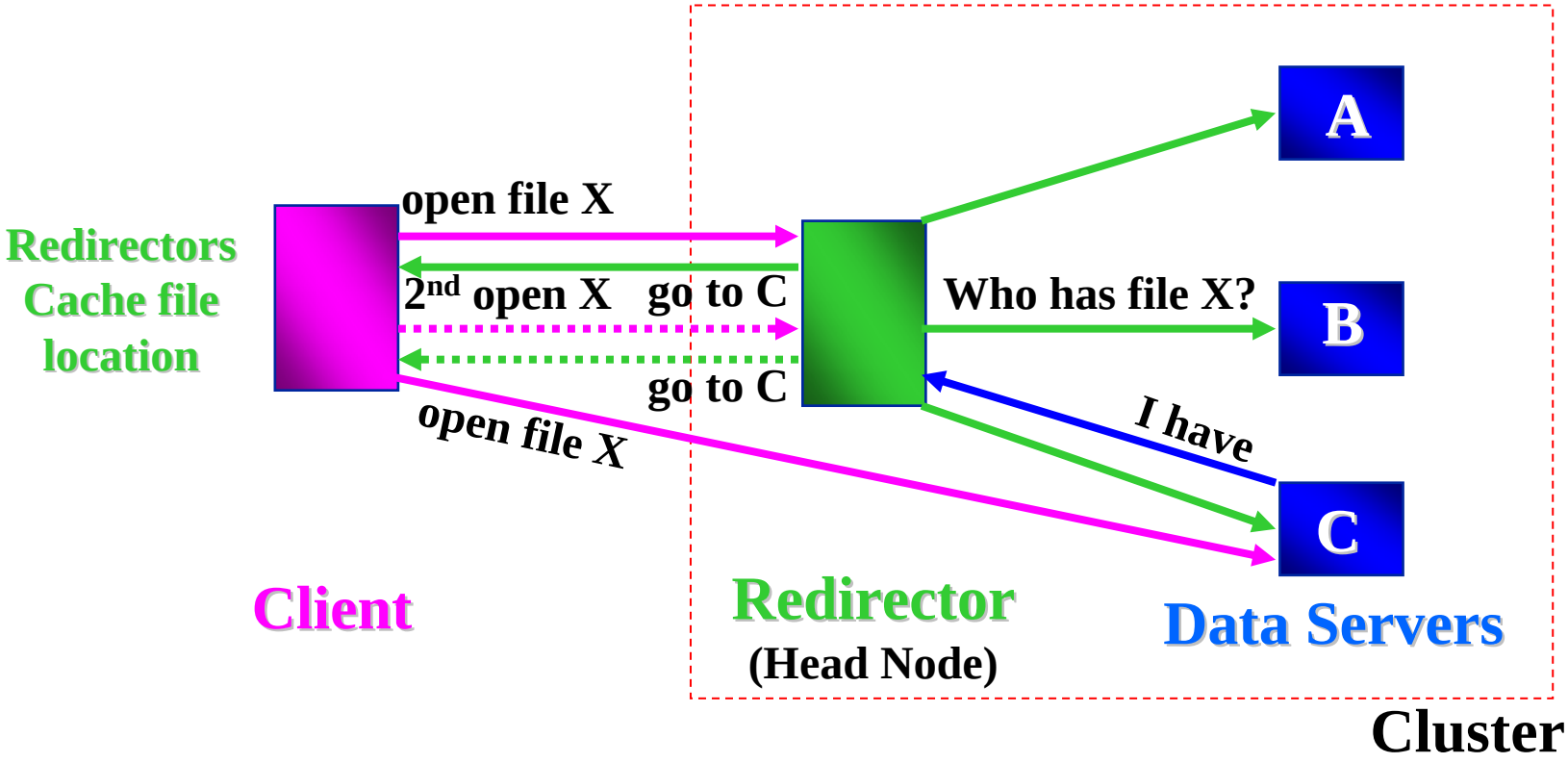
Andy Hanushevsky, SLAC
Alex Romosan, LBNL

August, 2006

Motivation

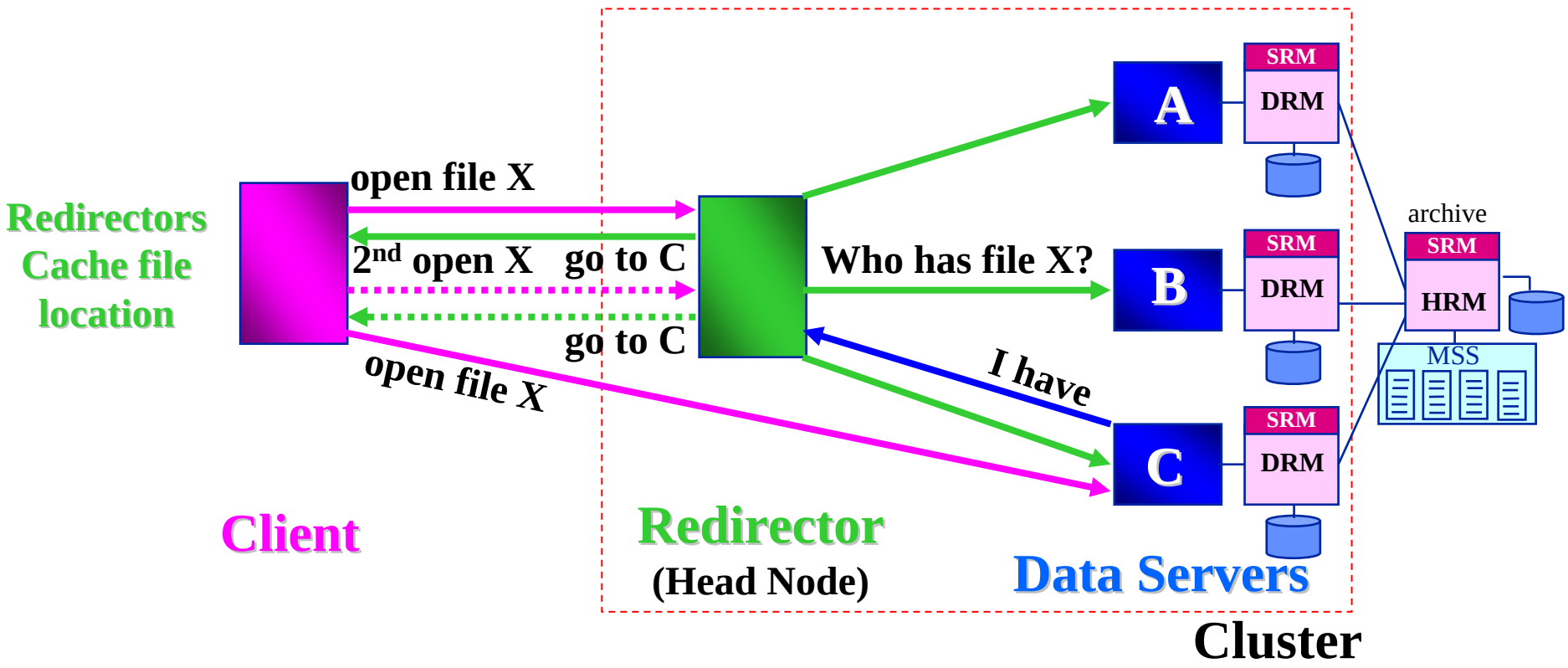
- xrootd provides low latency high bandwidth data access
  - Does not address disk cache management
- Need standard way to bring data into/out of disk cache
  - Comes with simple mps (Migration, Purge, Stage) system
    - Perl based scripts
    - Can interface to any kind of MSS
  - Lacks SRM interface for grid access
- Replace mps with LBL DRM/HRM
  - Automatically provides SRM access
  - xrootd architecture very amenable to extensions

xrootd Cluster



Client sees all servers as xrootd data servers

xrootd Cluster



Client sees all servers as xrootd data servers

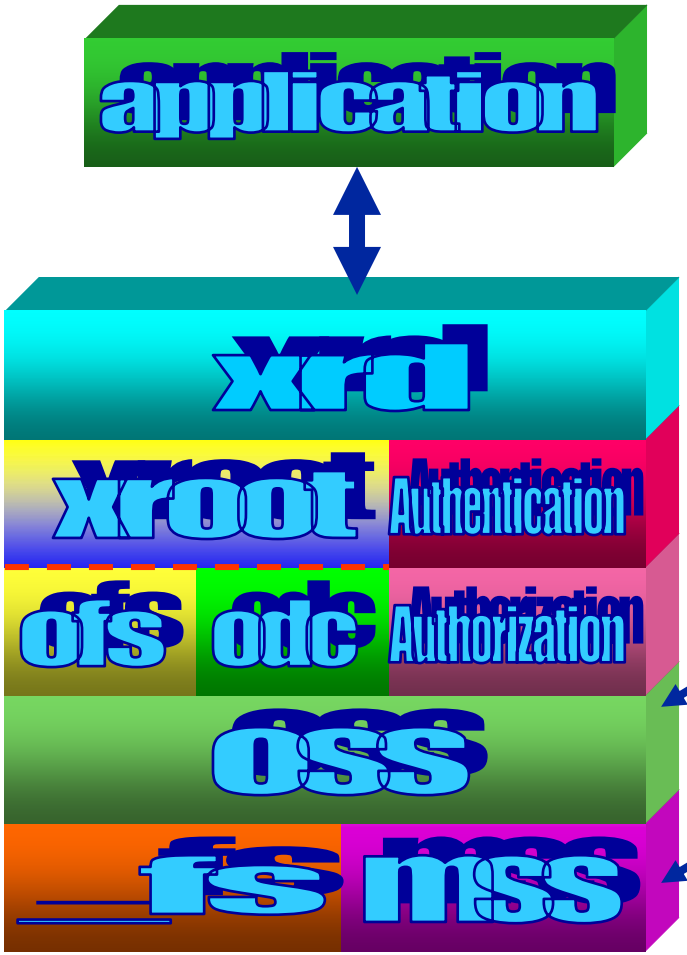
LFN to SURL and PFN mappings

- We assume a one-to-one mapping between:
  - LFN and the SURL-remote
  - LFN and the local-PFN

- Example
  - LFN: test.raw
  - remote-SURL:
    - srm://garchive.nersc.gov/nersc/gc5/romosan/test.raw
  - Local-PFN:
    - file://tmp/test.raw
  - Local-PFN implemented as a symlink to the TURL

- Comments
  - Machine and Directory of “remoteroot” site is a parameter given to xrootd-SRM
    - E.g. garchive.nersc.gov/nersc/gc5/romosan
  - Directory of “localroot” path is a parameter given to xrootd-SRM
    - E.g. tmp
  - At BNL: one-to-one LFN to SURL mapping also exists, so it is not necessary to consult the STAR file catalog

xrootd Server Architecture



Protocol & Thread Manager

Protocol Layer

Filesystem Logical Layer
xrootd.fslib ***path***

Filesystem Physical Layer
ofs.osslib ***path***

Filesystem Implementation

Extensions to xrootd:

- ***SRM-OSS implementation is a derived class of OSS base class***
- ***Replace fs+MSS with the DRM***

Extensions to xrootd OSS layer to accommodate SRM

- OSS layer externalized as a plug-in
  - Trivial as this was originally done for Objectivity/DB
    - Some symbol massaging needed to allow static default and dynamic shared library implementations
- Some existing classes augmented
  - XrdOss: newDir(), newFile(), newProxy() added
    - Ease initialization task for derived classes
  - XrdOssFile: Close(), Open() virtualized
    - Ease wrapping of concrete class implementation
  - XrdOssSys: Create(), Stage() virtualized
    - Ease wrapping of concrete class implementation

Derived OSS layer Plugin

- **Configuration options added**
  - srm.nshost — host running the nameserver
  - srm.nsport — nameserver port
  - srm.srmname — name of the srm
  - srm.msshost — host running mass storage system
  - srm.mssaccess — mss access type (gsi, login, none, krb5)
- **XrdOssSrm Class**
  - Overrides Create() and Stage() in XrdOssSys
    - Create() uses DRM to create a new file
    - Stage() uses SRM/HRM to retrieve a remote file
- **XrdSrmFile Class**
  - Overrides Close() and Open() in XrdOssFile
    - Close() informs DRM that the file is no longer in use
    - Open() informs DRM that the file is in use
- **In general a very simple extension to base class**
  - Packaged so that no 3rd party dependency exists for compilation
    - XrdOssSrm.cc, XrdOssSrm.hh, XrdSrmConfig.cc
    - Plus libsrmapi which abstracts the corba & globus dependencies

Xrootd-SRM plugin operations

- **Getting files**
  - OFS calls OssFile::open(LFN, mode=read)
  - it checks if file exists locally
  - If not, it calls OssSys::Stage(LFN)
  - For OSS-SRM
    - it creates the SURL from LFN (and keeps mapping)
    - It calls srm::get(SURL)
    - After the file gets from HRM, it creates a symlink from the TURL to local-PFN; sets request_id in the XrdSrmFile class; return “ok”
    - If file does not exist or system down, it return an “error”
  - OFS calls OssFile::close(LFN); check mode=read
  - For OSS-SRM
    - If request_id is in the XrdSrmFile, it srm::release (request_id)
    - DRM releases file

Xrootd-SRM plugin operations

- **Putting files requirements**
  - Put in cache only
  - Put in cache and archive ASAP
    - Same but after a delay
  - Modify existing file in cache only
  - Modify existing file in cache and archive ASAP
    - Same but after a delay
  - Migrate existing file from cache to archive ASAP

Xrootd-SRM plugin operations

- **Putting files**
- **Case 1: new file, no archive**
  - **OFS calls OssFile::open(LFN, mode=write,new)**
    - **it calls OssSys::Create(LFN,mode)**
  - **For OSS-SRM**
    - **Generate local-PFN and SURL**
    - **Call srm::put(), get back TURL**
    - **it creates a symlink from local-PFN to the TURL**
  - **OFS calls OssFile::close(LFN); check mode=write**
  - **For OSS-SRM**
    - **Call srm::putDone (SURL)**

Xrootd-SRM plugin operations

- **Putting files**
- **Case 2: new file, with archive**
  - **OFS calls OssFile::open(LFN, mode=write,new)**
    - **it calls OssSys::Create(LFN,mode)**
  - **For OSS-SRM**
    - **Generate local-PFN and SURL**
    - **Call srm::put(SURL), get back TURL**
    - **it creates a symlink from local-PFN to the TURL**
  - **OFS calls OssFile::close(LFN); check mode=write**
  - **For OSS-SRM**
    - **Call srm::putDone (SURL)**
    - **DRM communicates with HRM to perform put-push**
    - **Once transfer is done, HRM returned "success" (success/failure logged in xrootd logs)**

Xrootd-SRM plugin operations

- **Putting files**
- **Case 3: modify existing file, no archive**
  - **OFS calls OssFile::open(LFN, mode=write,append)**
    - **it calls OssSys::Create(LFN,mode)**
  - **For OSS-SRM**
    - **Generate local-PFN and SURL**
    - **perform srm::get(SURL), return TURL (if file not in cache it will be brought in)**
    - **Call srm::put(,hint:modify), DRM returns request ID only**
    - **(symlink already exists)**
  - **OFS calls OssFile::close(LFN); check mode=write**
  - **For OSS-SRM**
    - **Call srm::putDone (SURL)**

Xrootd-SRM plugin operations

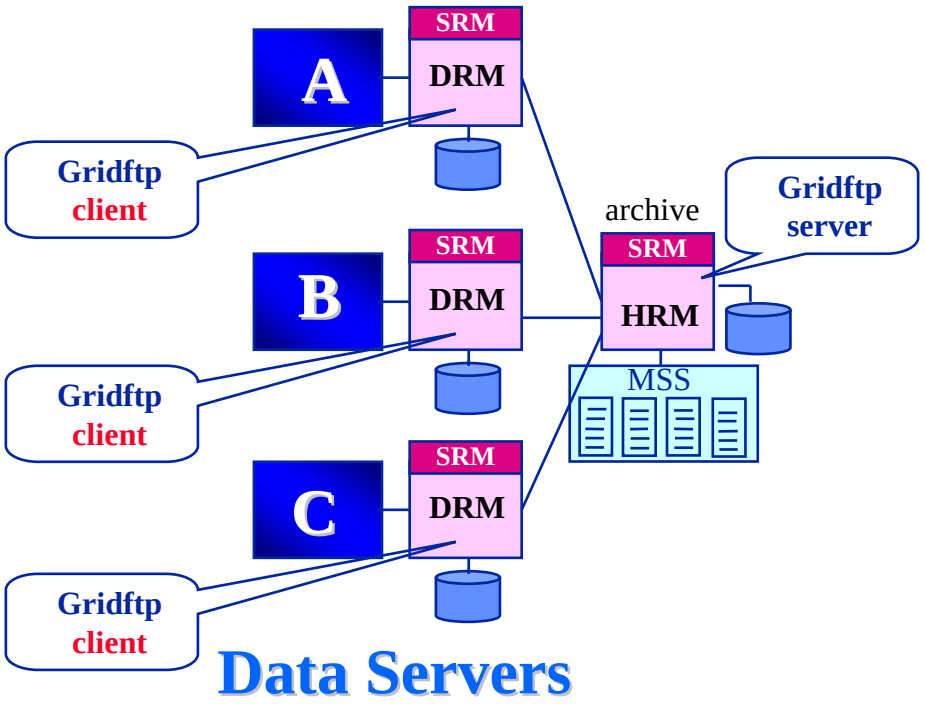
- **Putting files**
- **Case 4: modify existing file, with archive**
  - **OFS calls OssFile::open(LFN, mode=write,append)**
    - **it calls OssSys::Create(LFN,mode)**
  - **For OSS-SRM**
    - **Generate local-PFN and SURL**
    - **perform srm::get(SURL), return TURL (if file not in cache it will be brought in)**
    - **Call srm::put(SURL,hint:modify), DRM returns request ID only**
    - **(symlink already exists)**
  - **OFS calls OssFile::close(LFN); check mode=write**
  - **For OSS-SRM**
    - **Call srm::putDone (SURL)**
    - **DRM communicates with HRM to perform put-push**
    - **Once transfer is done, HRM returned "success" (success/failure logged in xrootd logs)**

Xrootd-SRM plugin operations

- **Putting files**
- **Case 5: copy a file from cache to archive**
  - **OFS calls OssFile::open(LFN, mode=write,sync)**
    - **it calls OssSys::Create(LFN,mode)**
  - **For OSS-SRM**
    - **Generate local-PFN and SURL**
    - **Call srm::put(SURL,hint=copy)**
    - **Check that file in cache, or return error**
    - **(symlink already exists)**
    - **DRM communicates with HRM to perform put-push**
  - **OFS calls OssFile::close(LFN); check mode=write,sync**
  - **For OSS-SRM**
    - **Once transfer is done, HRM returned "success" (success/failure logged in xrootd logs)**

Changes to SRM to accommodate xrootd needs

- **Implement put-push**
  - **Necessary since we want to have only Gridftp clients on xrootd servers**
  - **Gridftp servers on HRM (or remote SRMs) only**



Changes to SRM to accommodate xrootd needs

- **Implement srmPut with modes**
  - **Put in cache with no-archive / archive**
    - **Hint with targetSURL**
  - **Put modified files with no-archive / archive**
    - **Hint: modify**
  - **Copy from on-line to near-line**
    - **Hint: copy**
  - **Delayed archiving**
    - **Hint: delay=n seconds**

Current Status

- The xrootd-SRM activity is a collaboration between:
  - BNL – proposed the idea, providing early demonstrations, performing measurement, provide enhancements
  - SLAC – providing changes to xrootd, providing distribution mechanism with xrootd, planning to install at SLAC, where instead of HRM will use a DRM to a large disk cache
  - LBNL – providing plugins that interact with SRMs
    – providing changes to DRM for new functionality
- Planned installations
  - BNL - installed and started testing on current version, planning to install to work with real data in analysis tasks
  - SLAC - planning to install at SLAC, where instead of HRM will use a DRM to a large disk cache
  - Some functions still need to be implemented; specifically: remove, abort