SRM Space Tokens

Scalla/xrootd

Andrew Hanushevsky
Stanford Linear Accelerator Center
Stanford University

27-May-08

http://xrootd.slac.stanford.edu

Outline

- Introduction
  - SRM Static Space Token Refresher
- Space Tokens in the Scalla Architecture
  - Disk partitions as a space token paradigm
    - How it was done
  - Space usage and quotas by space token
- New Stuff
  - Proxies unlimited
  - Announcements
- Conclusion & Future Outlook

SRM Static Space Tokens

- Encapsulate fixed space characteristics
  - Type of space
    - E.g., Permanence, performance, etc.
  - Imply a specific quota
- Using a particular arbitrary name
  - E.g., data, dq2, mc, etc
- Typically used to create new files
  - Think of it as a space profile

SRM Space Tokens & Paths

- Static Space Tokens may be redundant
  - True if exclusive correspondence exists

Token	***Path***
dq2	/atlas/dq2/….
mc	/atlas/mc/….

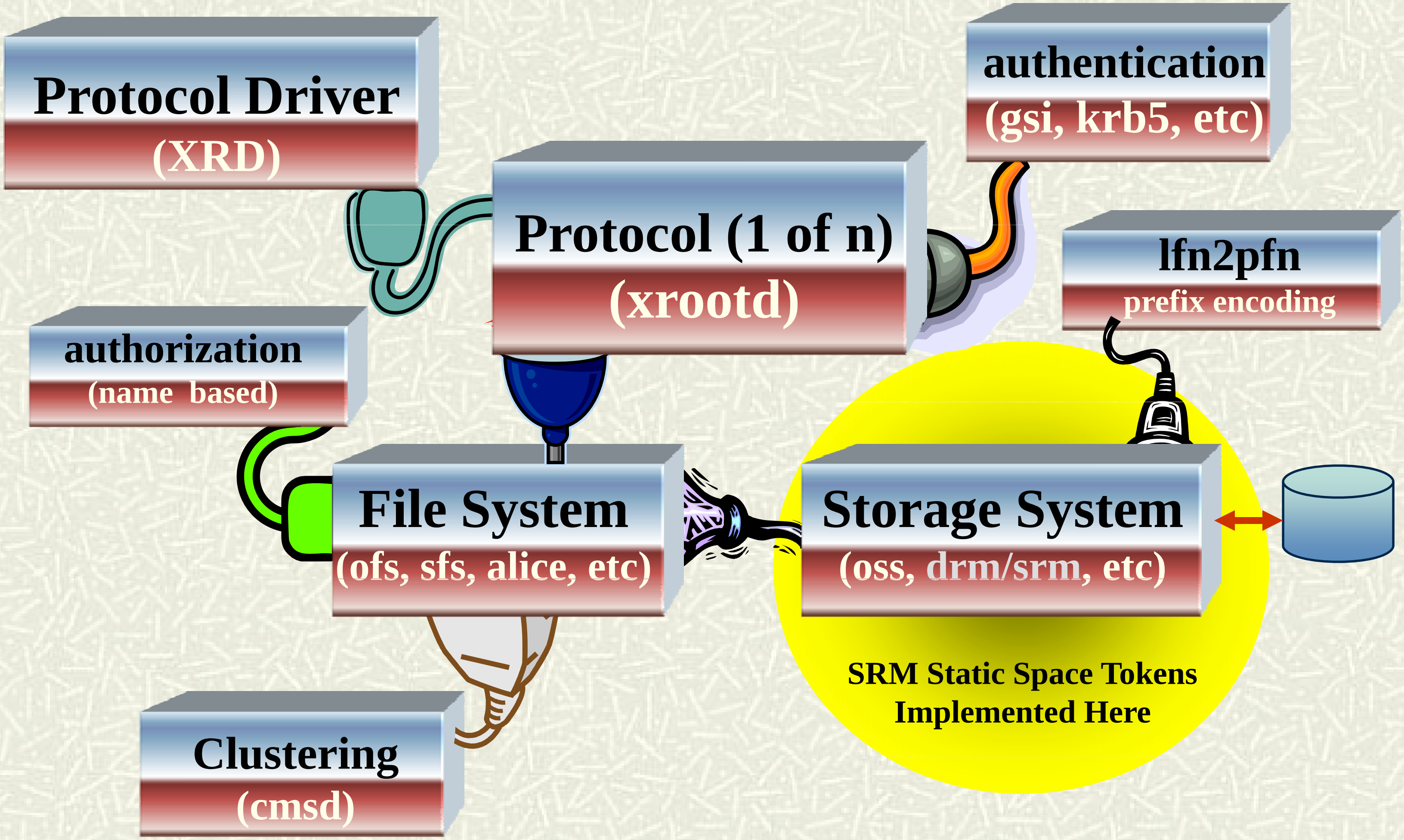
  - Space tokens useful for overlapping namespaces
- Makes space token utility non-obvious
  - But I digress, so let’s move on….

Space Tokens & xrootd

- Space attribute concept already part of xrootd
  - Embodied by notion of *cgroup*
    - A *cgroup* is a logical name for one or more file systems
- Implemented in the standard oss plug-in
  - Used by default libXrdOfs.so plug-in
- The real work was to support SRM concepts
  - Largely in the area of virtual quotas
    - Opportunity to greatly improve the implementation

Stanford
Linear
Accelerator
Center

Where Do Space Tokens Apply?



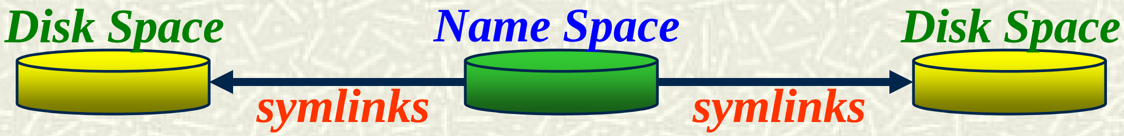
SRM Static Space Tokens
Implemented Here

Partitions as a Space Token Paradigm

- Disk partitions map well to SRM space tokens
  - A set of partitions embody a set of space attributes
    - Performance, quota, etc.
  - A static space token defines a set of space attributes
    - Partitions and static space tokens are interchangeable
- xrootd already supports multiple partitions
  - Real as well as virtual partitions
    - Can leverage this support for SRM space token support
- So, on to xrootd partition management

Partition Architecture

- N real partitions can be aggregated
  - Each aggregation is called a virtual partition
  - Uniform name space across all partitions
    - Real partitions are space load balanced
    - Reduces the granularity of failure
  - Implemented via symlinks from a name space
    - Name space itself resides in a real partition



Virtual vs Real Partitions

Virtual Partitions | **Real Partitions**

```
oss.cache public /store1 xa
oss.cache dq2    /store2 xa
oss.cache data   /store3 xa
```

What's this?

- Simple two step process
  - Define your real partitions (one or more)
    - These are file system mount-points
  - Map virtual partitions on top of real ones
    - Virtual partitions can share real partitions
  - By convention, virtual partitions equal static token names
    - Yields implicit SRM space token support

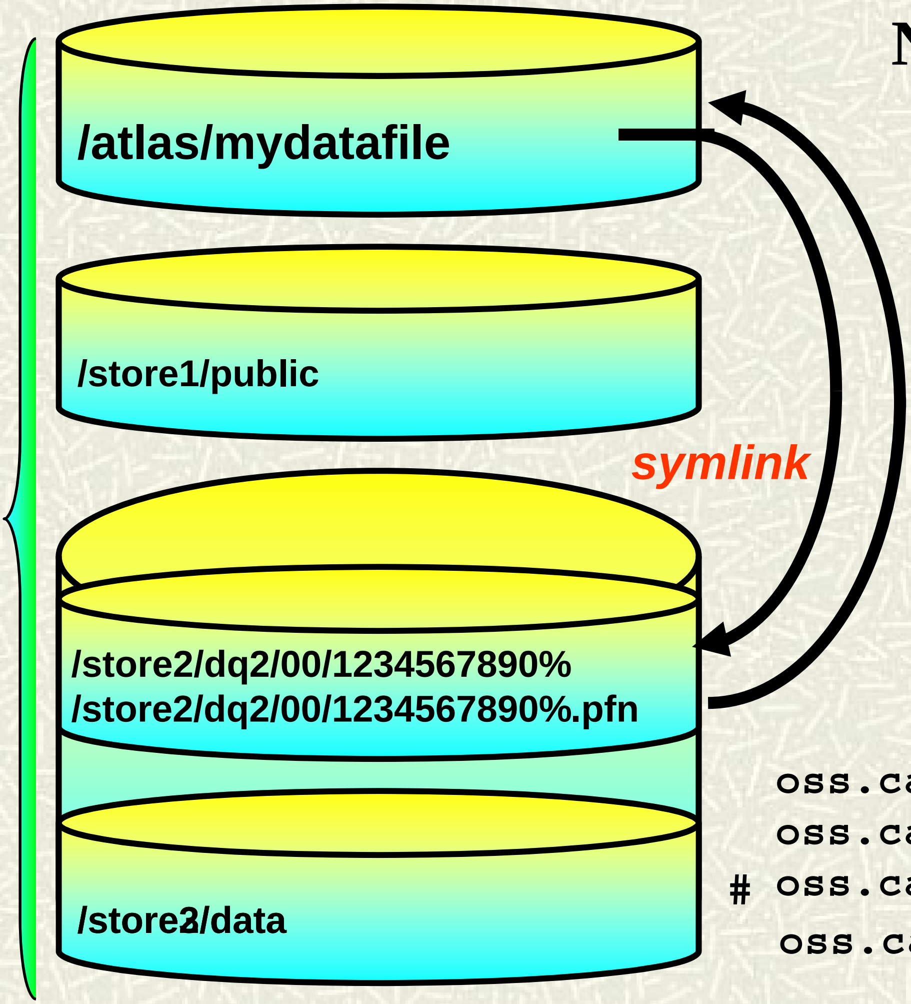
Introducing xa Partitions

- Original oss partition architecture was limited
  - Simplistic symlink target names
    - Constrained file path length to 255 or less
    - Could not automatically track assigned space tokens
- The **xa** option introduced for SRM support
  - Supports paths up to 1024 characters
  - Automatically tracks assigned space token
    - Tracks usage for real *and* virtual partitions
- Both supported for backward compatibility
  - The **xa** version is now preferred in all cases

Partition Aggregation

Independent Filesystems
Mount Points
/store1
/store2
/store3

Virtual partitions
dq2 & data
now share same
Real partition



Name Space Partition
Holds Name Space
Can also optionally hold data files

Symlink conventions simplify audits & name space recovery

```
  oss.cache public /store1 xa
  oss.cache dq2    /store2 xa
# oss.cache data   /store3 xa
  oss.cache data   /store2 xa
```

Partition Selection

- Partitions selected by virtual partition name
  - Configuration file:

    ```
    oss.cache public /store1 xa
    oss.cache dq2    /store2 xa
    oss.cache data   /store3 xa
    ```

  - New files “cgi-tagged” with virtual partition name
    - root://host:1094//atlas/mydatafile?cgroup=dq2
      - The default is “public”
  - File allocated in a real partition associated with the named virtual partition
    - By convention, the name is the SRM space token name

Real vs Virtual Partitions

- A real partition represents a hard quota
  - Non-overlapping virtual partitions are real
  - Simple and very effective
    - Typically not efficiently utilized
- Shared real partitions
  - Overlapping virtual partitions are virtual
  - Provide better space utilization, but…
    - Need usage tracking and quota management

Partition Usage Tracking

- Usage is tracked by partition
  - Automatic for real partitions
  - Configurable for virtual partitions
    - `oss.usage {nolog | log dirpath}`
- As Virtual Partition ⇔ SRM Space Token
  - Usage is also automatically tracked by space token
- POSIX getxattr() returns usage information
  - See Linux man page

Partition Quota Management

- Quotas applied by partition
  - Automatic for real partitions
  - Configurable for virtual partitions
    - `oss.usage quotafile` ***filepath***
- POSIX getxattr() returns quota information
  - Used by Fuse/xrootd to enforce quotas
    - Fuse has view of the complete cluster
      - Using the cluster name space daemon

The Quota File

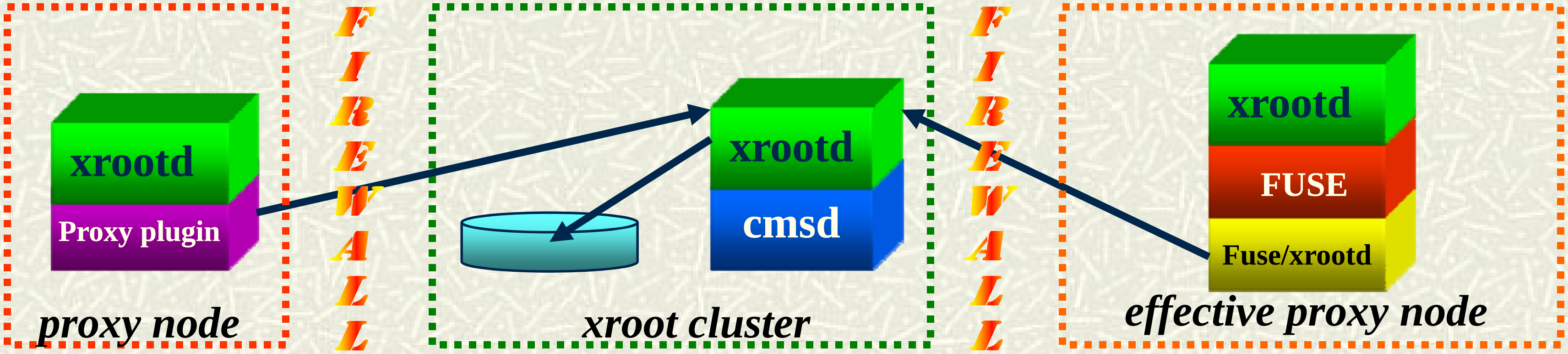
- Lists quota for each virtual partition
  - Hence, also a quota for each static space token
  - Simple multi-line format
    - ***vpname nnnn*****[k | m | g | t]\n**
  - Re-read whenever it changes
- Useful only in the context of the cnsd xrootd
  - Quotas need to apply to the whole cluster
- Investigating native integration with the redirector
  - Currently, only FUSE/xrootd enforces quotas

Other Considerations

- Files cannot be easily reassigned space tokens
  - Must manually “move” file across partitions
    - Partitions 1-to-1 correspondence with space tokens
- Can always get original space token name
  - Use file-specific getxattr() call
- Quotas for virtual partitions are “soft”
  - Time causality prevents hard limit
    - Use real partitions if hard limit needed

Proxies Unlimited

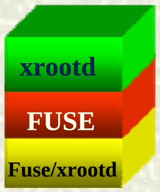
- Classic Proxy Server
  - Restricted to a very specific role
- Introducing **FUSE** as a proxy
  - All cluster features available
    - We are still investigating this exciting concept



FUSE Proxy Transfer Rates

SLAC	#Streams	BNL MB/s	CERN MB/s
scp	1	0.7	0.4
gridftp	1	1.4	0.4
xrdcp	1	2.2	0.6
xrdcp	5	5.6	2.2
xrdcp	11	10.0	4.8
xrdcp	15	10.1	6.1

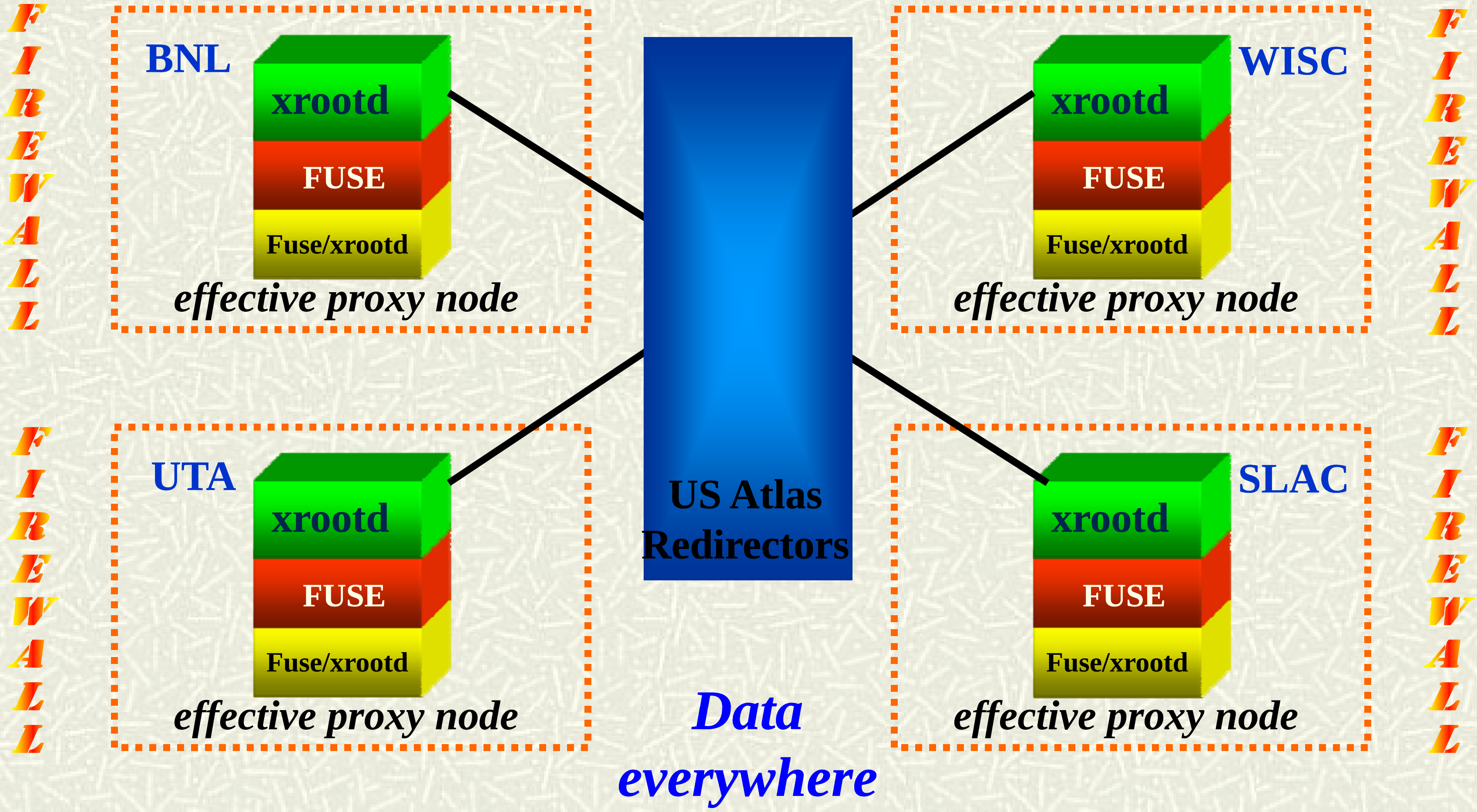
We don’t have any explanations yet.
We need to do more tests.
(especially with multi-stream gridftp)



effective proxy node

Stanford
Linear
Accelerator
Center

Secure Data Sharing Achievable



Data everywhere

Announcements!

- The CERN-based Scalla web page is online!
  - http://savannah.cern.ch/projects/xrootd/
- Scalla CVS repository is going public!
  - Will be located in afs with unrestricted read access
  - Planning on providing web access

Conclusion & Future Outlook

- Scalla/xrootd is living up to our expectations
  - Relatively easy to add SRM space token support
- More improvement are in the pipeline
  - Kernel memory direct data transfer (a.k.a. sendfile)
    - Significant reduction in CPU/Memory usage
  - Directed Support Services (**DSS**) Architecture
    - A take-off on Multics special files
      - Currently supporting Just-In-Time Alice Data Analysis
        - Bandwidth management during on-demand data transfers
    - Framework for simple intra-cluster resource management

Acknowledgements

- Current software Collaborators
  - Andy Hanushevsky, Fabrizio Furano
  - Root: Fons Rademakers, Gerri Ganis (security), Bertrand Bellenot (windows)
  - Alice: Derek Feichtinger, Andreas Peters, Guenter Kickinger
  - STAR/BNL: Pavel Jackl, Jerome Lauret
  - SLAC: Jacek Becla, Tofigh Azemoon, Wilko Kroeger
- Operational collaborators
  - BNL, CERN, CNAF, FZK, INFN, IN2P3, GSI, RAL, SLAC
- SLAC Funding
  - US Department of Energy
    - Contract DE-AC02-76SF00515 with Stanford University