Distributed File Storage in Globus

Max Berger <max@berger.name>

February 18th, 2004

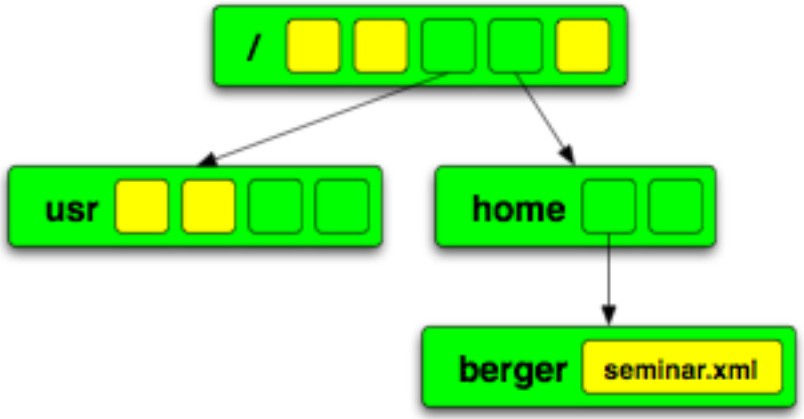
Overview

- File Storage Systems
- GridFTP
- Replica Catalog
- Replica Management System
- Other goals for Distribued File Storage

File Storage Systems

- Tree Storage system
- Meta Attributes
- Remote File Storage
- Distributed File Storage

Tree Storage system



A file system has excactly 1 root directory.

Each directory has

- Files
- Directories

Meta Attributes

- File name
- File size (speedup)
- File type, icon (HFS)
- Last modified date
- Last accessed date
- File creation date
- File owner
- File permissions

Remote File Storage

- Files are not stored on the local machine, but on a remote machine
- Common Goal: Transparency for user and applications
- Usual implementation: Locator for file storage consisting of **host** and **sharename** (e.g. `host:/dir` in NFS, `\\server\share` in SMB), path relative to that locator.
- **Problem:** files cannot be moved to a different host

Distributed File Storage

- Goal: Keep actual host out of file locator
- Solution: introduce **Realms** insted of single hosts.
- Locator now points to Realm, path relative to that locator
- Example AFS: Path `/afs/cs.ttu.edu/home/berger`
- Files can now be freely moved and replicated by the system administrator

GridFTP

- Goals
- What is GridFTP?
- FTP
- Implementation
- GridFTP is not
- Conclusion

Goals

- Grid Security Infrastructure and Kerberos support
- Third-party control of data transfer
- Parallel data transfer
- Striped data transfer
- Support for reliable and restartable data transfer
- Partial file transfer
- Manual control of TCP buffer size
- Integrated Instumentation

What is GridFTP?

- GridFTP is a **Protocol**
- Based on FTP (RFC 959)
- FTP Security extensions (RTC 2228)
- Feature negotiation mechanism for the File Transfer Protocol (RFC 2389)
- FTP Extensions (draft-ietf-ftpext-mlst-14)

GridFTP is mainly the FTP protocol with a few extensions

FTP

- Uses TCP
- Separate Data and Control Streams
- Active Mode: Sends Data
- Passive Mode: Receives Data
- Advantage: Third-Party controlled transfers
- Disadvantage: Not Proxy-able

Implementation

First Generation (regular ftp tools with gsi extensions)

- 1.1.3 GridFTP Server: gsi-wuftpd
- 1.1.3 GridFTP Client: gsi-ncftp

Second Generation:

- GT2GridFTP Server: based on wuftpd
- globus-url-copy: client

GridFTP is not

- an implementation, it is a protocol
- concerned with data replication, just transfer
- data secure, just login secure
- proxy-safe
- a distributed file storage

Conclusion

good for:

- high-speed transfer
- a local network
- massive amounts of data

Bad for:

- confidential data
- proxied environments

Replica Catalog

- Goals
- Terminology
- Implementation
- Usage
- Shortcomings
- Conclusion

Goals

- Remove physical host from file descriptor
- Support replicated data
- Support striped data
- Provide Meta-data

Terminology

- Collection: Set of items
- Location: Maps between logical item name and physical location
- Logical File Entry: Meta Attributes for item

Implementation

- Uses LDAP as database
- Could be any backend

Usage

Standalone-tool for

- registering collections / adding files
- searching for files
- delete collections / files

Shortcomings

- No automatic replication of data (yet)
- Metadata in single database
- Collections is nice, but tree is better
- Path to database has to be known

Conclusion

- Step towards distributed storage
- Good for large amount of data
- in few collections
- No support for modifying files
- Good for read-only, bad for read-write

Replica Management System

- Goals
- Terms
- Types of Consistency
- Raptor
- Metadata in Raptor
- Optor
- Conclusion

Goals

Combine GridFTP and Replica Catalog into one single system

- Transparent for the user / programmer
- Can do replica management
- Can be otimized

Terms

- Replica
- Master Copy
- Secondary Copy
- Consistency
- Filenames: Logical (LFN), Site (SFN)

Types of Consistency

- Read-Only replicas
- Read/Write replicas
- Versioning

Raptor

- Replica Mangement Service
- Single point of entry
- Only service to control copy, delete, modify
- Multible transport services
- Pluggable processing services
- Communication with other RMS
- GUIDs

Metadata in Raptor

- File: size, checksum, dates, type, alias
- Collections: elements, type
- Security: ownership, access permissions, local policies
- Management: expiration, master copy, transactions

Optor

- Replica Selection
- Access History Service
- Replica Initiation Service

Conclusion

- Finally: A distributed storage system!
- Good extensibility through plug-ins
- Most interesting: Replica Initiation Service

But:

- Multiple RMS still unclear
- Still single point of failure

Other goals for Distribued File Storage

- Transparency
- Fail-Safe
- Speed
- Conclusion

Transparency

- Location
- Access
- Replication
- Failure
- Concurrency
- Migration
- Language

Fail-Safe

- Disconnected operation
- Hoarding
- Automatic replicas
- Writeable replicas

Speed

- Striping
- Parallel transfers
- Caching
- Delayed write-backs
- Automatic migration

Conclusion

The way the file storage is handled in Globus is good for what it is intended to be

but: The file storge in SORCER will be better!