A framework for Data Management and Transfer in Grid Environments*

Haojie Zhou[1,2], Xingwu Liu[1] Liqiang Cao[1], Li Zha[1]

[1] 1 Research Center for Grid and Service Computing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100080, China
[2]Graduate School of the Chinese Academy of Sciences, Beijing 100080, China
{Zhouhaojie, xlw,caolq, char}@software.ict.ac.cn

Abstract. The main obstacles to grid file management come from the fact that grid file resources are typically stored in heterogeneous and distributed environment and accessed through various protocols. In this paper, we propose a grid file management system called Vega [1][2] Hotfile2 for data-intensive application in widely distributed systems and grid environments. Widely distributed and heterogeneous storage resource can converge into a single uniform storage resource using virtualization techniques in Hotfile2. It is a grid middleware provides a uniform file resources view and a high level interface to access widely distributed and heterogeneous data resources. Presented here are two services that we believe are fundamental to any grid file management system: high-performance, secure transport and metadata management. In this paper, we discuss many aspects of our design and implementation in Hotfile2, present the file system interface designed to support distributed applications such as batch job. Experiments are done to illustrate its performance, extensibility, and various other features.

Keywords: VEGA, GOS , Hotfile2, distributed file management, data grid

1 Introduction

Grid computing applications require the efficient management and transfer of terabytes or petabytes of data in wide area, distributed computing environments. Hence, many grid file systems and grid file management tools have been developed, among which are Grid FTP [3], GASS [4] and the Storage Resource Broker (SRB) [5] and so on.

Unfortunately, these grid file systems typically use incompatible and unpublished protocols for accessing data and therefore each requires a private client. Secondly

* This work is supported in part by the China Ministry of Science and Technology 863 Program (Grant No. 2005AA119010), the National Natural Science Foundation of China (Grant No. 69925205), and the China National 973 Program (Grant No. 2003CB317000 and No. 2005CB321807), and Chinese Academy of Sciences Distinguished Scholars Fund (Grant No. 20014010).

these grid file systems can not be dynamically deployed to or undeployed, so they dissatisfy the requirement of managing and using the file resource in grid environment. Thirdly, to avoid unauthorized access, most network firewalls block all but HTTP socket ports. But the existing grid file management systems need special socket port which may bring security problem. We design and implement a grid middle ware Hotfile2 to cover these problems.

The China National Grid aims to construct a grid environment and illustrate the function of grid environments. The Vega GOS V2[1] [2] and its the associated utilities have been implemented based on J2SE and Apache Tomcat/axis to validate the Vega Grid architecture and core techniques, such as agora, grip, and the EVP grid resource space model. Most core services of the Vega GOS has been heavily debugged and integrated into a package which is called Vega GOS V2 and has been released and deployed on the China National Grid. The Hotfile2 is the system level software in the integrated grid environment provides data service for GOS and its applications deployed in Vega GOS V2 such as batch service. It also offers the grid users a separated file space.

Based on Web service technology and HTTP protocol, this paper address the design and implementation of a three-layer file system framework and develops a web-service based grid file transfer protocol. The Hotfile2 system has been integrated into China National Grid Project and applied to many research projects (e.g. Biologic Gene Test and Navigation Simulation).Listed below are some features:

- Hotfile2 can be dynamically deployed, maintained and used, so it is extensible for user's demand on storage resources.
- Hotfile2 is a protocol between services and file naming in Hotfile2 is based on service name and service address.
- Control channel and data channel in Hotfile2 is HTTP based, provides security and generic performance file transfer only using HTTP socket.

The rest of this paper is organized as follows: section 2 presents our file system framework names Grid File Address Space Model, naming, file transfer protocol and security in Hotfile2; section 3 discusses the Hotfile2 deployment in GOS environment; section 4 presents how to using Hotfile2 to access files resource to support batch developed for GOS; section 5 discusses the performance of Hotfile2 contrasting to GridFTP and GASS; section 6 introduces some related works; section 7 concludes this paper, with further work briefly discussed..

2 System architecture

2.1 Grid File Address Space Model

The Vega Grid Team developed a grid address space model called the EVP Grid Resource Space Model[2]. The model stratifies a grid into three layers: effective layer, virtual layer and physical layer. Native resources and services located at individual grid nodes are called physical resources (P). These resources and services are abstracted into location-independent virtual resources (V) at the grid operating system layer. The grid end users can presented with effective resources (E) views, e.g. through GOS V2 Portal. This hierarchical mechanism greatly facilitates solving such problems in Grid Systems as single resource image, uniformly resource access, and convexity of information [6].

According to the EVP Grid Resource Space Model for Grids, we developed a Grid file storage resource space model. Grid file storage resource space is composed of three layers.

Physical storage space: the collection of storage resources at grid nodes. Storage resource here means the file system on grid nodes. Note that grid nodes may be distributed and heterogeneous and the file system may be different.

Virtual storage space: the abstraction of physical storage resources provides uniform accessing interface. From the view of virtual storage space, storage resources are still distributed.

Effective storage space: provide a convenient interface for programmers or end-users, and resolves authentication, authorization and access control problems in grid systems.

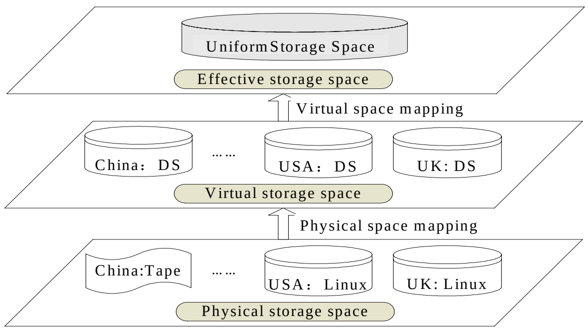

Fig. 1. The process of constructing a storage resource space.

Figure 1 describes a bottom up process to construct a storage resource space model. Physical storage space is composed of distributed, heterogeneous storage resources. The physical storage space can be transformed into distributed but homogeneous virtual storage resources through storage resource space mapping which is located in agoras (an implementation of VO in GOS V2). In virtual storage space, the heterogeneity of storage resource is eliminated but still distributed. The Grid effective storage space is composed of Grid users' storage space and can be transformed into virtual resource space in agoras through storage resource space mapping. In effective storage space, the distributed, heterogeneous storage resources can be viewed as a single storage space.

Since the storage resource space model is divided into effective storage space, virtual storage resource and physical storage space, files in our Hotfile2 system are divided into grid effective file, grid virtual file and grid physical file, and we construct a top-down EVP Grid File Address Space Model based on EVP Storage resource model.

Grid effective file: the file which is visible and operable by end-users and grid applications. The effective file name is exclusive in users' file space.

Grid virtual file: the exclusive identifier of grid file in whole file management system, which is similar to the file handle in Unix file system or the public file handle in NFS. The grid virtual file unifies the interface for accessing virtual file resource and the right expression for current user. It is a self-contained data structure for accessing physical storage resource rather than an abstraction of file. Grid virtual file is dynamic and created while file operation.

Grid physical file: grid physical file is the files stored at physical storage resource(e.g. NTFS file system on Windows or EXT3 file system on Linux) which may be distributed and heterogeneous.

A grid effective file is visible to grid user and grid applications. Through name mapping, an effective file in user space can be mapped into virtual file which is exclusive in the whole grid system including self-contained information for accessing physical file. A grid virtual file can be mapped into a grid physical file through virtual storage resource selection. While grid user or grid application accessing files in Hotfile2, where the file is stored and what kind the target file system is of is transparent.

The Grid File EVP Address Space Model unifies the grid user's file space. This section theoretically brings forward a method fro accessing files in heterogeneous, dynamic, distributed and autonomic storage resources.

2.2 Naming

According to the EVP Grid File Address Space Model, there are three kinds of files in grid: Grid effective file, Grid virtual file and Grid physical file. Presented below is the naming mechanism in Hotfile2.

A grid effective file is identified by its effective name, which is exclusive in grid user's file space. An effective name consists of two parts: the first part is the identifier of transfer protocol , with "efile://" indicating accessing file through HTTP and "efiles://" indicating transfer file through SSL(Security Socket Layer); the second part is the file path and the file in grid user's file space.

Example of effective file name in Hotfile2

```
efile://Gos/HelloWord.txt

means using TCP/IP to access file: /Gos/HelloWord.txt

efiles://Gos/HelloWord.txt

means using SSL to access file: /Gos/HelloWord.txt
```

Virtual file name and physical file name is transparent to end users. A virtual file name is composed of three parts: the first part is the transfer protocol "vfile"; the second is the data service address which only can be understood by Vega Router[15]; the third part is the file path including user information; the forth part is a exclusive identifier in the appointed data service.

Example of virtual file name in Hotfile2

```
vres://data service address %User Info%local identifier
```

Physical file name information dose not actually exist in Hotfile2 system and it is stored physical storage node. The virtual file name includes a file identifier which can be translated into physical file name on physical storage grid node. The format of physical file name is the same as a file name in traditional file system.

Effective file name, virtual file name and other metadata is stored in metadata service. Before grid user accessing a file in Hotfil2, metadata service should be invoked first to do naming mapping and access control and so on.

2.3 File Transfer Protocol

Control channel and data transfer channel in Hotfile2 transfer protocol are also based on HTTP. Hotfile2 transfer protocol includes server side and client side. The server side provides file operation interface based on SOAP which can work in grid independently. But using web service directly increases complexity in application programming. To resolve this problem, we provide file transfer protocol client. We provide file transfer client API for programmers.

File transfer protocol server side in GOS V2 is a Web Service runs on GOS Server composed of three modules: file transfer control module; file transfer model and local file system at the server node.

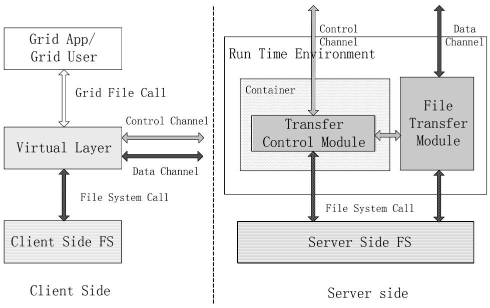

Fig. 2. The structure of Hotfile2 transfer protocol

There are three layers on the client side: Grid Application/ Grid User, virtual layer and client side local file system. Grid Applications and Grid Users operate files using file transfer client API. The Virtual layer hides the local or remote file operation. Upon receiving a local file request, the virtual layer using system call at client side local file system. if a remote file request is received, the virtual layer redirect the request to the control channel or data channel on the destination server.

2.4 Security

User authentication is not addressed in Hotfile2, it is solved in GOS V2. Under development is a community service provides authorization service which facilitates community-based access control implemented in GOS V2. To access Hotfile2 system, the grid user should be authorized to access metadata service and data service of Hotfile2 in community service.

Access control in Hotfile2 using ACL which is stored in metadata service. To share files in Hotfile2 file management system, we provides grant and revoke mechanism. Files and directories can be grant accessing permission or revoke accessing permission to another user or a group in local community by the owner. More flexible file sharing mechanism will be provided soon.

Meta data in Hotfile2 is secure while metadata service using GOS V2 security mechanism [16] which is WS-Security compatible using Handler chains mechanism and GOS context. Data can be secured while in transit by using SSL protocol, but it is stored using plain test in physical storage node and data can be illegal accessed and modified by the node administrator.

HTTP based Hotfile2 system is unsecured but has better performance in data transfer. HTTPS based Hotfile2 provides the same secure rank compares to GridFTP and GASS.

3 Deployment

The Hotfile2 server side is composed of two modules: metadata service module managing all metadata information in Hotfile2 and data service providing physical file storage and file transfer service. Currently, the Hotfile2 system integrates a single metadata service and multi data service to compose the Hotfile2.

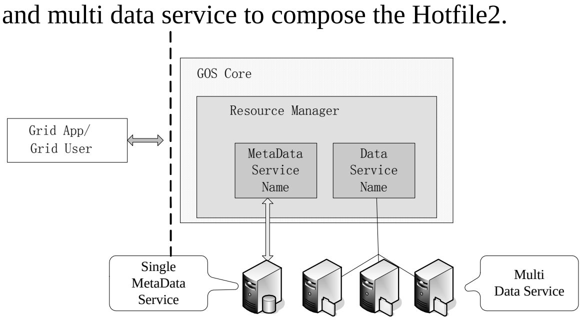

Fig. 3. Deploy structure of Hotfile2 in GOS

Metadata service and data service should be registered into Resource Manager System in GOS V2. Metadata service name and data service name constituted the effective storage space mentioned in Grid file address space model and the physical distributed metadata service and data service constituted the physical storage space. Data service name can be mapped to multi physical data service and physical data service can be dynamically registered into GOS System consequently. So Hotfile2 system provides a extensible storage space for user's dynamic demands.

Physical data service may be distributed and heterogeneous. It's unnecessary for grid applications and grid users to know the physical node position and the operation system running on the nodes, and where and how a file is stored. These are all transparent to end users.

4 Applications

We use batch service to show how Hotfile2 is used in grid applications.

In GOS V2, a batch is encapsulated as a web services. Batch Service is deployed on grid nodes and can be dynamically registered into GOS System. If a batch job is submitted a batch work through GOS Portal, GOS core will select an appropriate physical batch service to process the batch request according to the resource selection policy. There are two problems in processing a batch request:

- How to stagein file during submitting a batch while the processing node is unsure.
- How to stageout file after the batch is end.

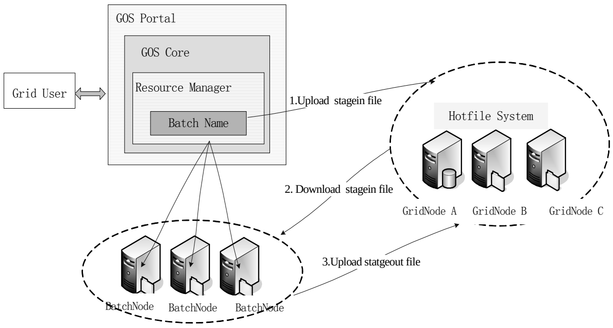

Fig. 4. Example of batch example using Hotfile2 in GOS V2

These problems can be solved using Hotfile2 system. When a user submits a batch job to grid system, the stagein file is uploaded to Hotfile2 (see phase 1 in Fig. 4.). After selecting a physical Batch service to processes the batch request, the stagein file is downloaded from Hotfile2 to the appointed node (phase 2 in Fig. 4.). While the processing is end, the stageout file will be upload to Hotfile2 again (phase 3 in Fig. 4.).As a result, grid users can view and download the stageout file in his own file space using tools or through the Hotfile2 user portal.

5 Performance Evaluation

Experiments are done to evaluate the performance of our Hotfile2. We deployed Data services on computers with 2 xeon 2.4G CPUs, 1GB memory and 36GB utltra3 SCSI hard disk running red hat Linux 7.3 and JVM 1.4.2. The Client is PC with PIII 1G CPU, 512M memory and 60GB ATA-100 hard disk running WinXP and JVM 1.4.2. The Server and Client are connected by 100M fast Ethernet.

5.1 Meta Data Performance

While testing the performance of metadata service running on GOS, each metadata operation such as Create Dir and Create File is tested 10 times, and the average time is taken as its expectant performance. The results are illustrated in Figure 5. Note that the time consumption when using HTTPS is almost twice that using HTTP, but it pays if high security is more important.

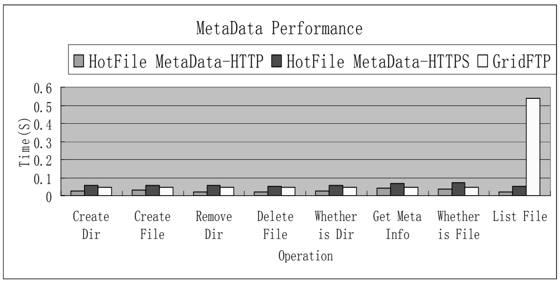

Fig. 5. Meta Service performance in Hotfile2

5.2 Data Transfer Performance

Fig. 6 and Fig. 7 show the Hotfile2 data transfer performance compared to GridFTP (deployed in GT3.2) and GASS.

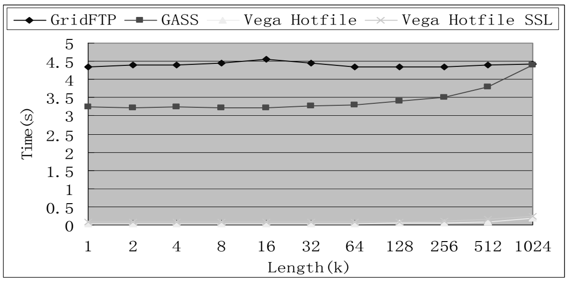

Fig. 6. Performance of three protocol while transfer files smaller than 1 megabyte

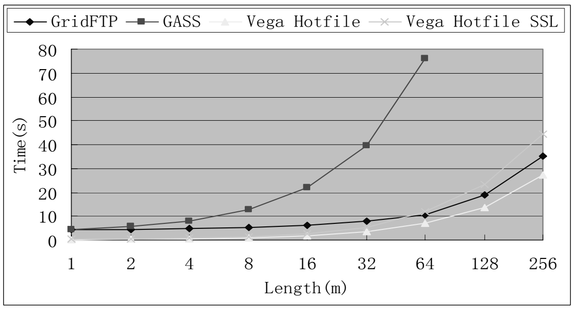

Fig. 7. Performance of three protocols while transfer files between 1megabyte and 256

GridFTP and GASS are significantly slower than Hotfile2 while transfer small files(smaller than 1 megabyte). Spending immovable time in constructing security data channel accounts for the low performance of GridFTP.

While transferring files ranging from 1 megabyte to 256 megabyte, GASS is the slowest protocol. Hotfile based on HTTP is the fastest because there is no encoding and decoding overhead in data transfer. Hotfile based on HTTPS is faster than GridFTP if the transferred files are smaller than 64 megabyte.

6 Related Work

GridFTP is an extensions of standard FTP protocol that create a universal grid-wide transport protocol which is an important part of Globus Toolkit[8] . It provides not only secure, high throughput data transfer even on high-speed wide area networks, but also third party transfers which allow the source, destination, or both to be striped, with arbitrary and potentially different topologies or even file access mechanisms. GridFTP is the foundation module in many data grid project.

GASS[4] (Global Access to Secondary Storage) is an regularly used data management and access protocol in Globus project. GASS defines a global name space via Uniform Resource Locators and allows applications to access remote files via standard I/O interfaces. High performance is achieved by incorporating default data movement strategies that are specialized for I/O patterns common in wide area applications and by providing support for programmer management of data movement.

The first edition of Hotfile[10,]implemented in VEGA GOS V1 is based on GridFTP and GASS. It wraps GridFTP and GASS or any other file transfer protocol compatible with Hotfile structure into a unified vegafile protocol providing a user level grid file systems with low overhead and high usability.

7 Conclusion and Future work

In this paper we have presented our design and implementation of Hotfile2. Hotile2 provides a uniform file resources view and a high level interface to access distributed, heterogeneous storage resources. It can be dynamically deployed and undeployed in GOS V2 environment hence providing an extensible storage resource space to meet the grid application requirements. Based on GSI, HTTPS and other security mechanism, it provides secure communication and transport in grid environments.

Based on HTTP, control channel and data transfer channel in Hotfile2 solve the firewall problem, and our protocol achieves better performance in transferring small and middle files comparing to GridFTP and GASS,.

Further improvement of Hotfile2 is now under way. For example, to implement reliable and third party file transfer, to ensure consistency in file replica and replica selection, and to define a set of file access interfaces which is compatible with the POSIX file interface.

References

1. Li Zha, Wei Li, Haiyan Yu, Jiping Cai "Service Oriented VEGA Grid System Software Design adn Evaluation" Chinese Journel of Computers Vol.28 No.4 April 2005 495~504
2. Zhiwei Xu, Wei Li, Li Zha, Haiyan Yu, Donghua Liu, Vega: A Computer Systems Approach to Grid Computing, Journal of Grid Computing, 2004, Vol.2, Issue 2：109~120
3. H Stockinger, A Samar, B Allcock, I Foster, K Holtman, and B Tierney. File and Object Replication in Data Grids. Journal of Cluster Computing, 2002, 5(3)305-314
4. Joseph Bester, Ian Foster, Carl Kesselman. GASS: A Data Movement and Access Service for wide Area Computing System. In: Proceedings of the Sixth Workshop on Input/Output in Parallel and Distributed Systems. ACM Press New York, NY, USA May, 1999
5. R Moore, A Rajasekar, and M Wan. The SDSC Storage Resource Broker. In: Procs. of CASCON'98, Toronto, 1998
6. Z.Xu A Model of Grid Address Space with Applications, VGSD- 2, (2002)
7. I. Foster, C. Kesselman, G. Tsudik, and S. Tuecke. A security architecture for computational grids. In ACM Conference on Computers and Security, pages 83~91. ACM Press, 1998.
8. I. Foster and C. Kesselman. The Globus Project: A Status Report. In Proceedings of the HeterogeneousComputing Workshop, pages 4–18. IEEE Press, 1998.
9. Berners-Lee, T., Masinter, L. and M. McCahill, "Uniform Resource Locators (URL)," RFC1738 prop, (1994)
10. Liqiang Cao ,Jie Qiu, Li Zha, Haiyan Yu, Wei Li"Design and Implementation of Grid File Management System Hotfile" Grid and Cooperative Computing - GCC 2004: Third International Conference pp 129-136
11. The POSIX standards: http://www.opengroup.org/onlinepubs/007904975/toc.htm.
12. B. White, A. Grimshaw, and A. Nguyen-Tuong, "Grid-based File Access: the Legion I/O Model" , in Proc. 9th IEEE Int. Symp. on High Performance Distributed Computing (HPDC), pp 165-173, (2000)
13. J.Kubiatowicz et al., "OceanStore: An Architecture for Global-Scale Persistent Storage",Proceedings of the Ninth international Conference on Architectural Support forProgramming Languages and Operating Systems (ASPLOS), (2000)
14. I.Foster, C. Kesselman, J. Nick, S. Tuecke; The Physiology of the Grid: An Open GridServices Architecture for Distributed Systems Integration, Open Grid Service Infrastructure WG, Global Grid Forum, (2002)
15. Hai Mo, Zha Li, Liu Haozhi, A Scalable Resource Locating Service in Vega Grid , Proceedings of the Fourth International Workshop on Grid and Cooperative Computing (GCC2005), Dec. 2005, Beijing, China pp 597-608.
16. Efficient and Loosely Coupled Security Mechanism in Vega Grid Honglei Dai, Shuoying Chen, Li Zha, Zhiwei Xu pp.714-720 Semantics, Knowledge and Grid SKG2005