the globus alliance
www.globus.org

Striped GridFTP Server Framework and Architecture

Raj Kettimuthu,
ANL and U. Chicago

ARGONNE NATIONAL LABORATORY • UNIVERSITY OF CHICAGO • epcc • KTH VETENSKAP OCH KONST • NCSA • Univa • USC VITERBI SCHOOL OF ENGINEERING

Outline

- Introduction
- Features
- Motivation
- Architecture
- Globus XIO
- Experimental Results

What is GridFTP?

- In Grid environments, access to distributed data is very important
- Distributed scientific and engineering applications require:
  - Transfers of large amounts of data between storage systems, and
  - Access to large amounts of data by many geographically distributed applications and users for analysis, visualization etc
- GridFTP - a general-purpose mechanism for secure, reliable and high performance data movement.

Features

- Standard FTP features get/put etc, third party control of data transfer
  - User at one site to initiate, monitor and control data transfer operation between two other sites
- Authentication, data integrity, data confidentiality
  - Support Generic Security Services (GSS)-API authentication of the connections
  - Support user-controlled levels of data integrity and/or confidentiality

Features

- Parallel data transfer
  - Multiple transport streams between a single source and destination
- Striped data transfer
  - 1 or more transport streams between m network endpoints on the sending side and n network endpoints on the receiving side

Features

- Partial file transfer
  - Some applications can benefit from transferring portions of file
  - For example, analyses that require access to subsets of massive object-oriented database files
- Manual/Automatic control of TCP buffer sizes
- Support for reliable and restartable data transfer

Motivation

- Continued commoditization of end system devices means the data sources and sinks are often clusters
  - A common configuration might have individual nodes connected by 1 Gbit/s Ethernet connection to a switch connected to external network at 10 Gbit/s or faster
  - Striped data movement - data distributed across a set of computers at one end is transferred to remote set of computers

Motivation

- Data sources and sinks come in many shapes and sizes
  - Clusters with local disks, clusters with parallel file system, geographically distributed data sources, archival storage systems
  - Enable clients to access such sources and sinks via a uniform interface
  - Make it easy to adapt our system to support different sources and sinks

Motivation

- Standard protocol for network data transfer remains TCP.
- TCP's congestion control algorithm can lead to poor performance
  - Particularly in default configurations and on paths with high bandwidth and high round trip time
- Solutions include:
  - Careful tuning of TCP parameters, multiple TCP connections and substitution of alternate UDP based reliable protocols
- We want to support such alternatives

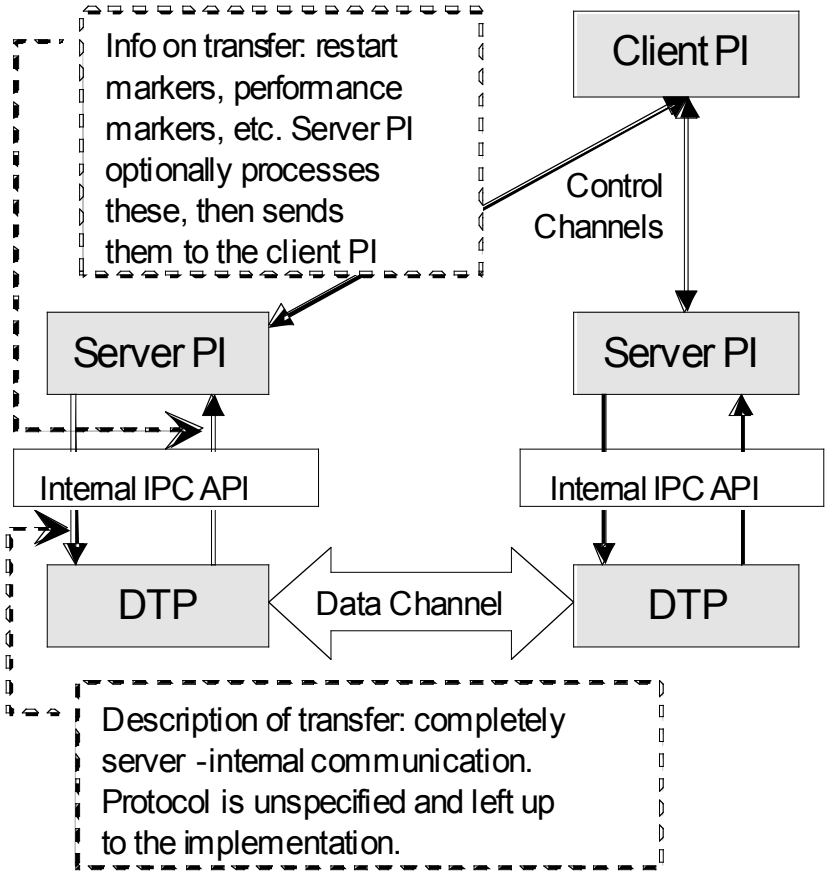
Architecture

- GridFTP (and normal FTP) use (at least) two separate socket connections:
  - A control channel for carrying the commands and responses
  - A Data Channel for actually moving the data
- GridFTP (and normal FTP) has 3 distinct components:
  - Client and server protocol interpreters which handle control channel protocol
  - Data Transfer Process which handles the accessing of actual data and its movement via the data channel

Architecture

- These components can be combined in various ways to create servers with different capabilities.
  - For example, combining the server PI and DTP components in one process creates a conventional FTP server
  - A striped server might use one server PI on the head node of a cluster and a DTP on all other nodes.

Globus GridFTP Architecture



Architecture

- The server PI handles the control channel exchange.
- In order for a client to contact a GridFTP server
  - Either the server PI must be running as a daemon and listening on a well known port (2811 for GridFTP), or
  - Some other service (such as inetd) must be listening on the port and be configured to invoke the server PI.
- The client PI then carries out its protocol exchange with the server PI.

Architecture

- When the server PI receives a command that requires DTP activity, the server PI passes the description of the transfer to DTP
- After that, DTP can carry out the transfer on its own.
- The server PI then simply acts as a relay for transfer status information.
  - Performance markers, restart markers, etc.

Architecture

- PI-to-DTP communications are internal to the server
  - Protocol used can evolve with no impact on the client.
- The data channel communication structure is governed by data layout.
  - If the number of nodes at both ends is equal, each node communicates with just one other node.
  - Otherwise, each sender makes a connection to each receiver, and sends data based on data offsets.

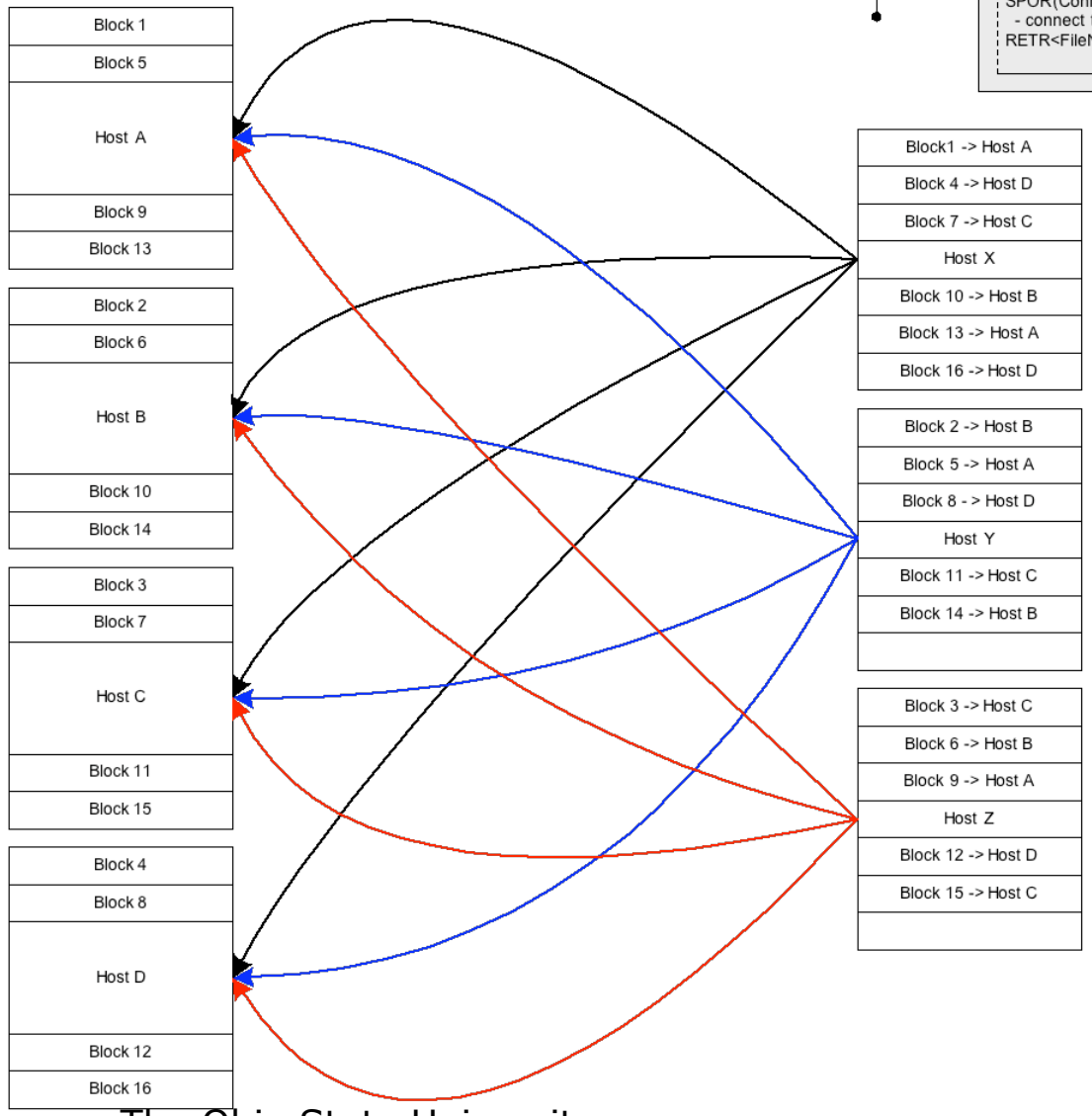
the globus alliance
www.globus.org

18-Nov-03

GridFTP Striped Transfer

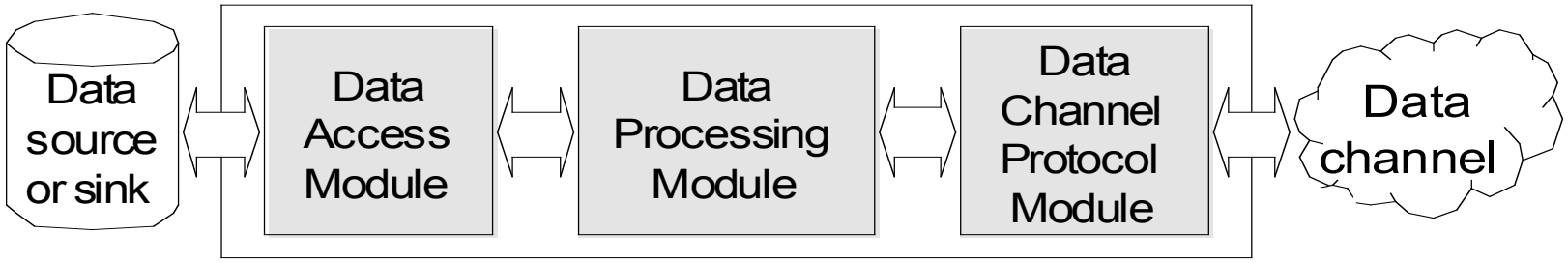
MODE E
SPAS (Listen)
- returns list of host:port pairs
STOR<FileName>

MODE E
SPOR(Connect)
- connect to the host-port pairs
RETR<FileName>



Data transfer pipeline

- Data Transfer Process is architecturally, 3 distinct pieces:
  - Data Access Module, Data processing module and Data Channel Protocol Module



Data access module

- Number of storage systems in use by the scientific and engineering community
  - Distributed Parallel Storage System (DPSS)
  - High Performance Storage System (HPSS)
  - Distributed File System (DFS)
  - Storage Resource Broker (SRB)
  - HDF5
- Use incompatible protocols for accessing data and require the use of their own clients

Data access module

- It provides a modular pluggable interface to data storage systems.
- Conceptually, the data access module is very simple.
- It consist of several function signatures and a set of semantics.
- When a new module is created, programmer implements the functions to provide the semantics associated with them.

Data processing module

- The data processing module - provides ability to manipulate the data prior to transmission.
  - Compression, on-the-fly concatenation of multiple files etc
  - Currently handled via the data access module
  - In future we plan to make this a separate module

Data channel protocol module

- This module handles the operations required to fetch data from, or send data to, the data channel.
  - A single server may support multiple data channel protocols
  - the MODE command is used to select the protocol to be used for a particular transfer.
- We use Globus eXtensible Input/Output (XIO) system as the data channel protocol module interface
  - Currently support two bindings: Stream-mode TCP and Extended Block Mode TCP

Extended block mode

- Striped and parallel data transfer require support for out-of-order data delivery
- Extended block mode supports out-of-sequence data delivery
- Extended block mode header

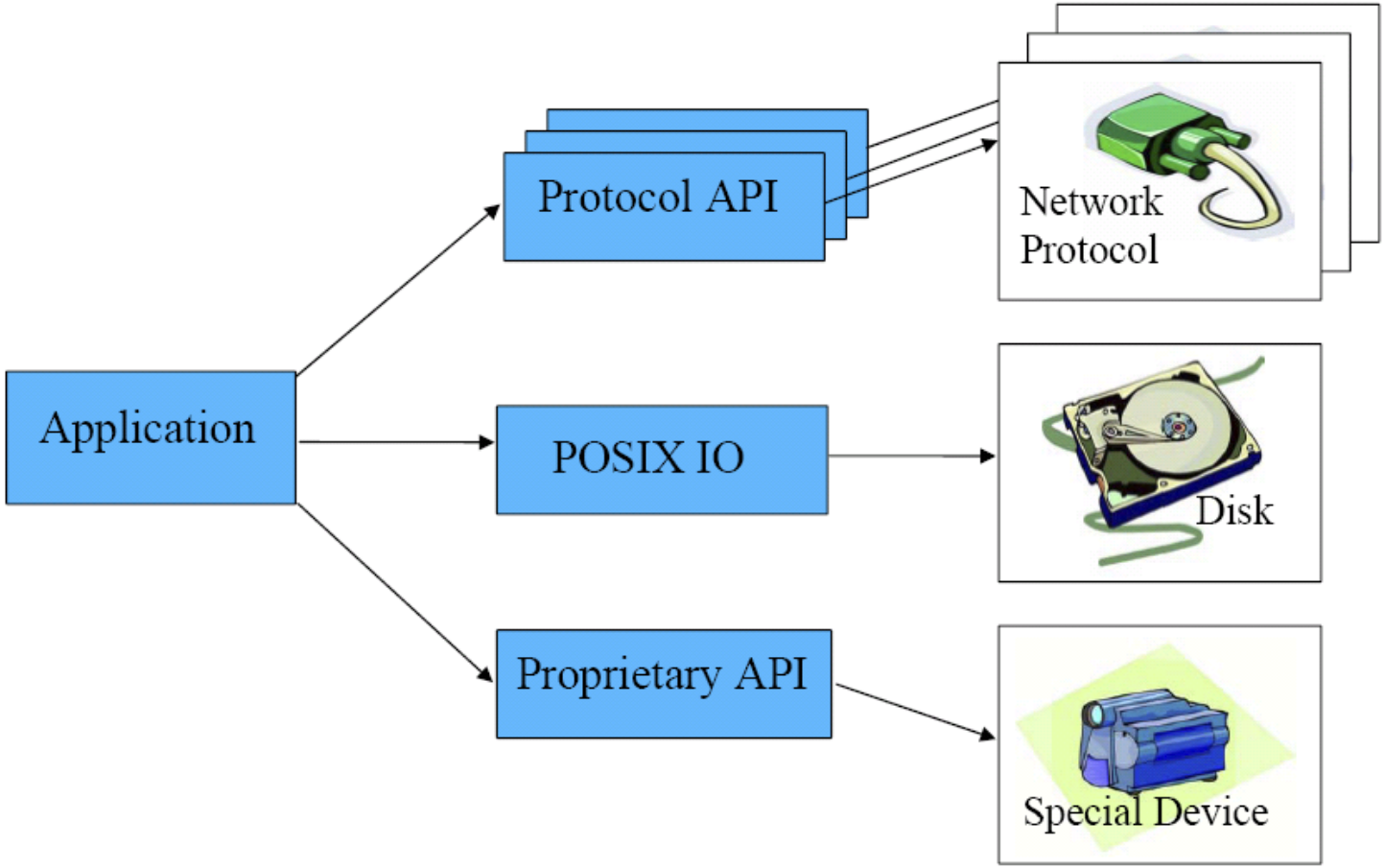
Descriptor 8 bits	Byte Count 64 bits	Offset 64 bits

- Descriptor is used to indicate if this block is a EOD marker, restart marker etc

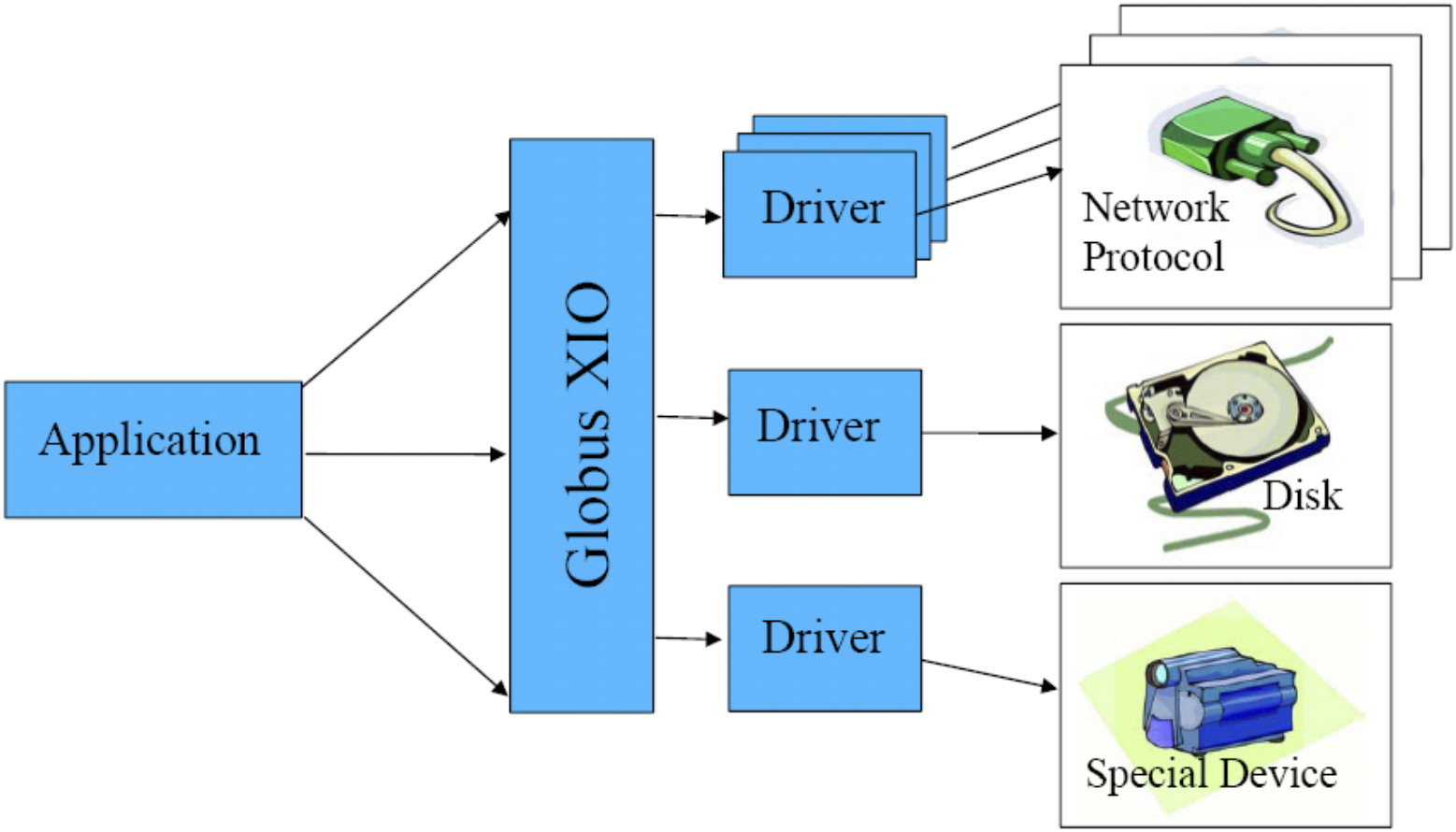
Globus XIO

- Simple Open/Close/Read/Write API
- Make single API do many types of IO
- Many Grid IO needs can be treated as a stream of bytes
- Open/close/read/write functionality satisfies most requirements
- Specific drivers for specific protocols/ devices

Typical approach



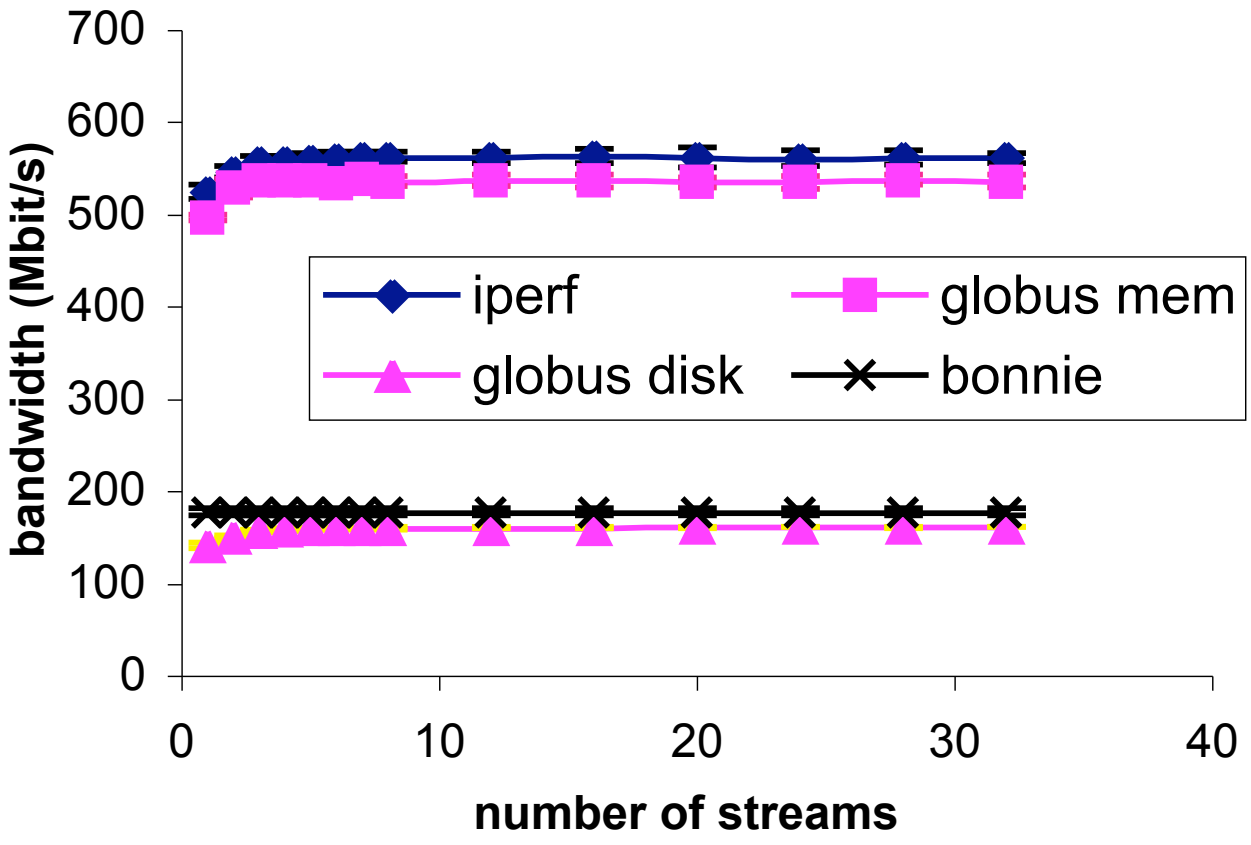
Globus XIO approach



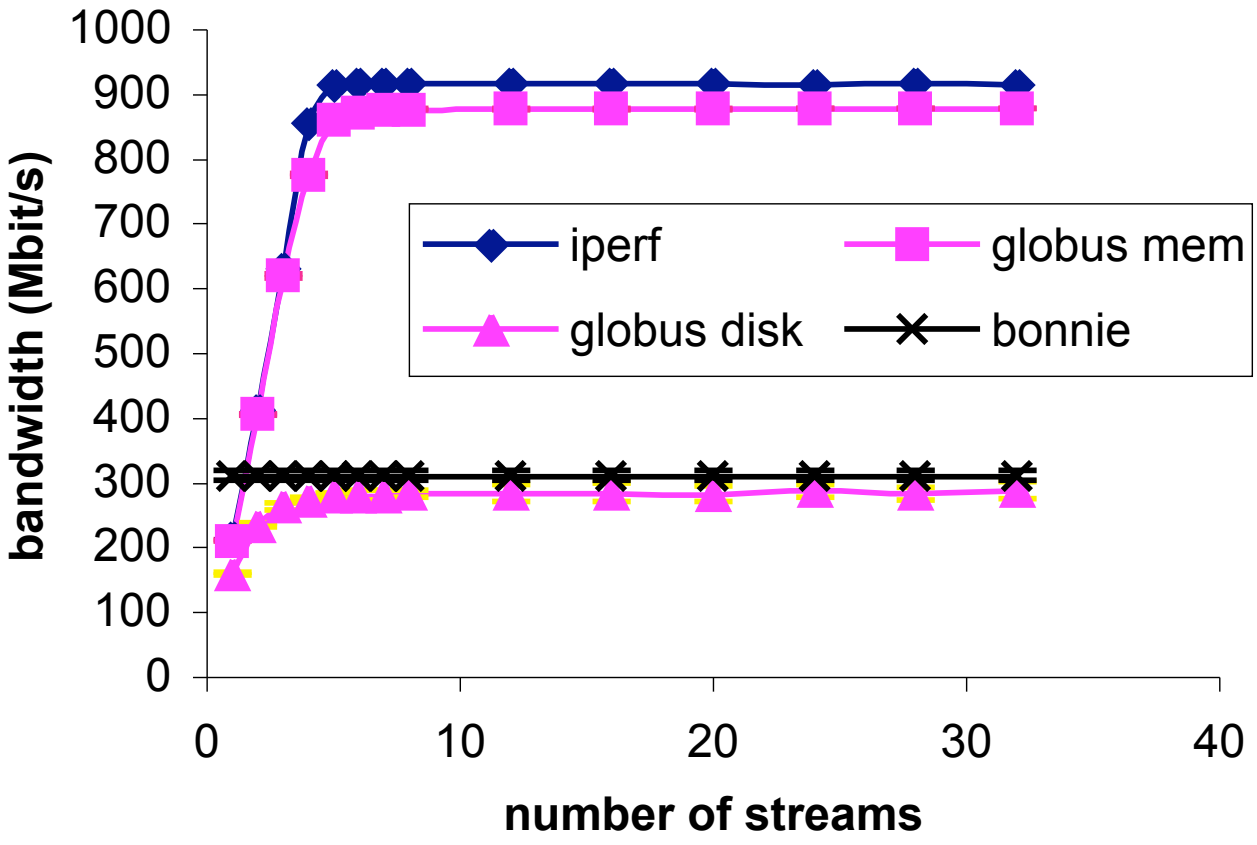
Experimental results

- Three settings
  - LAN - 0.2 msec RTT and 622 Mbit/s
  - MAN - 2.2 msec RTT and 1 Gbit/s
  - WAN - 60 msec RTT and 30 Gbit/s
  - MAN - Distributed Optical Testbed in the Chicago area
  - WAN - TeraGrid link between NCSA in Illinois and SDSC in California - each individual has a 1Gbit/s bottleneck link

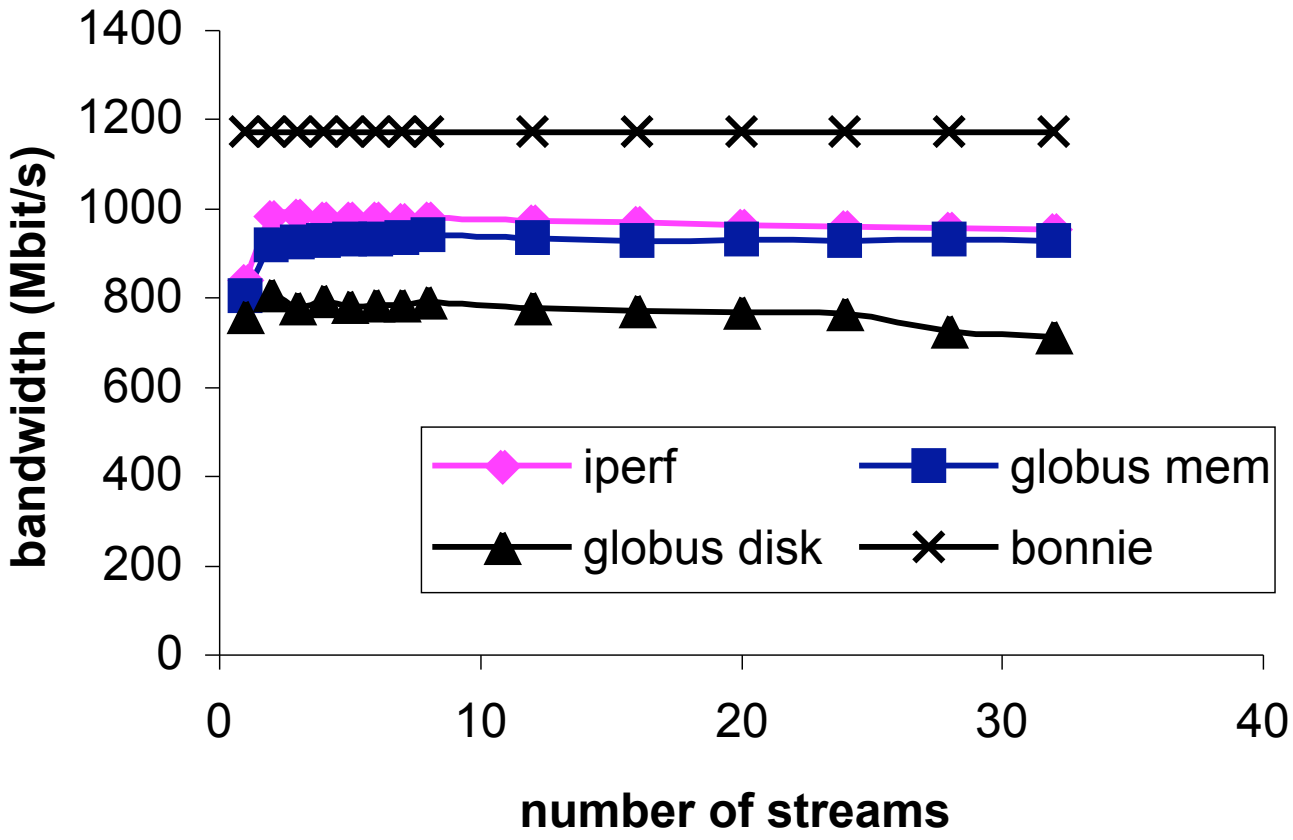
Experimental results - LAN



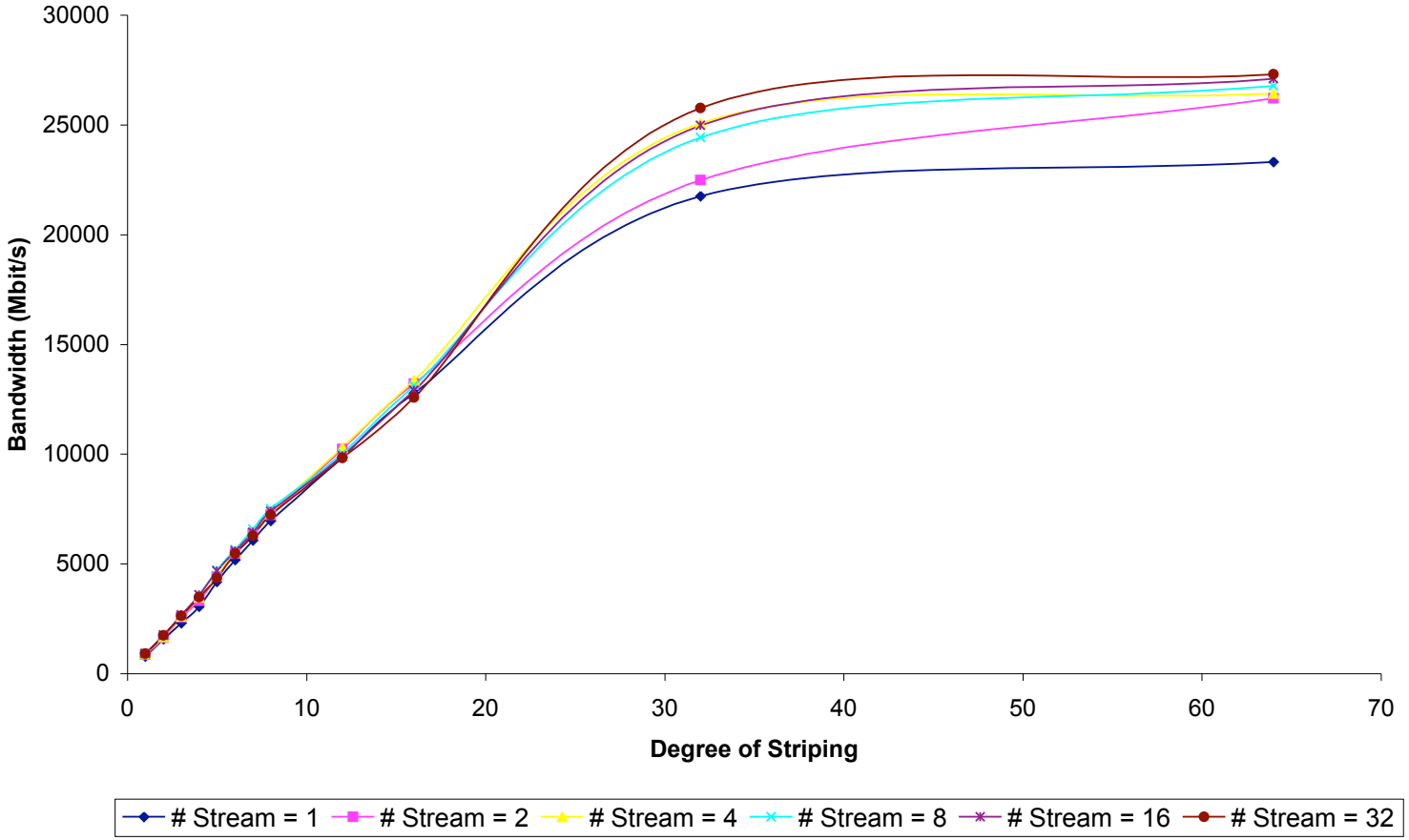
Experimental results - MAN



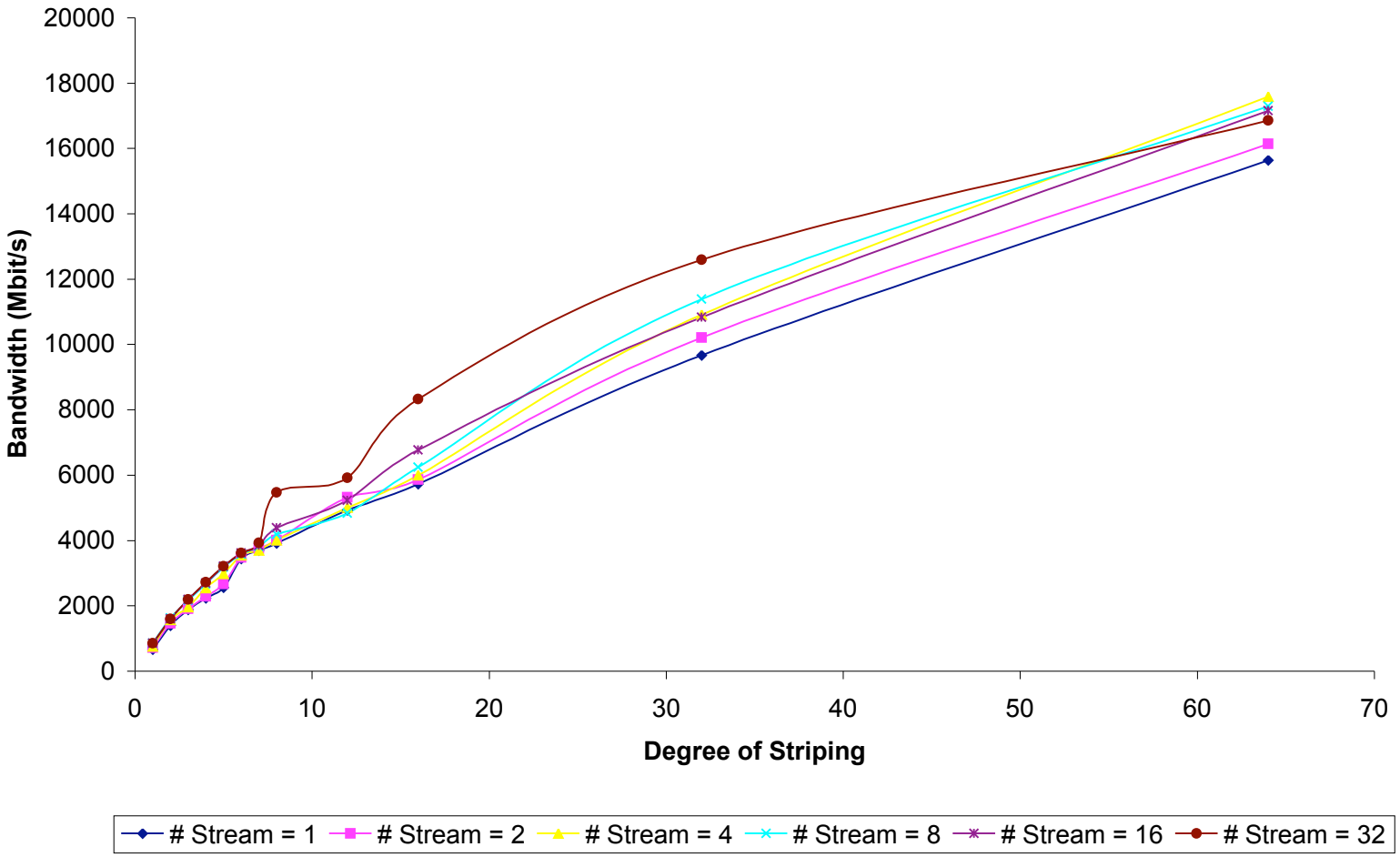
Experimental results - WAN



Memory to Memory Striping Performance



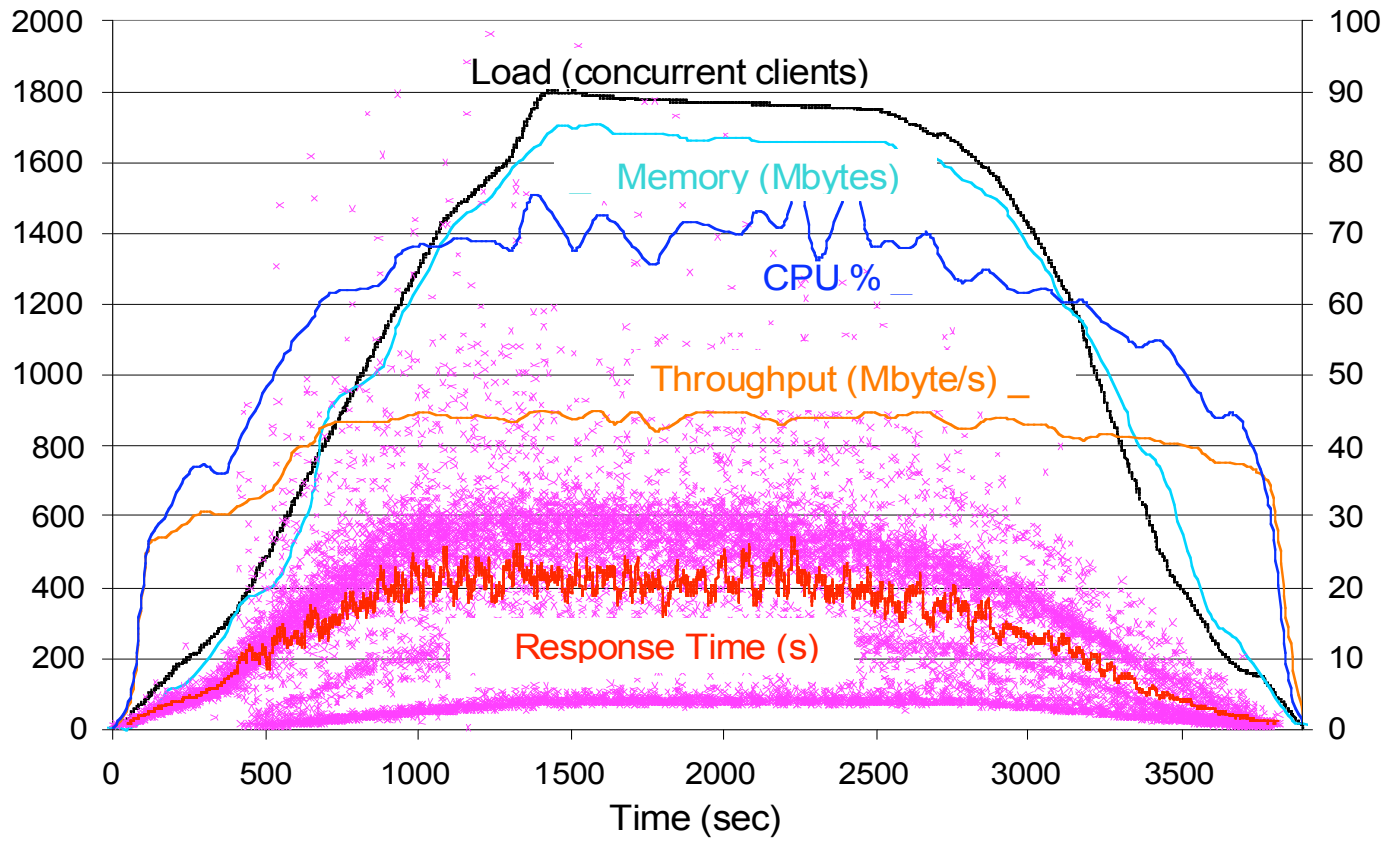
Disk to Disk Striping Performance



Scalability tests

- Evaluate performance as a function of the number of clients
- DiPerf test framework to deploy the clients
- Ran server on a 2-processor 1125 MHz x86 machine running Linux 2.6.8.1
  - 1.5 GB memory and 2 GB swap space
  - 1 Gbit/s Ethernet network connection and 1500 B MTU
- Clients created on hosts distributed over PlanetLab and at the University of Chicago (UofC)

Scalability Results



Left axis - load, response time, memory allocated

Right axis - Throughput and CPU %

Scalability results

- 1800 clients mapped in a round robin fashion on 100 PlanetLab hosts and 30 UofC hosts
- A new client created once a second and ran for 2400 seconds
  - During this time, repeatedly requests the transfer of 10 Mbyte file from server's disk to client's /dev/null
- Total of 150.7 Gbytes transferred in 15,428 transfers

Scalability results

- Server sustained 1800 concurrent requests with 70% CPU and 0.94 Mbyte memory per request
- CPU usage, throughput, response time remain reasonable even when allocated memory exceeds physical memory
  - Meaning paging is occuring