Globus Toolkit

Managed GridFTP

Raj Kettimuthu

Argonne National Laboratory and

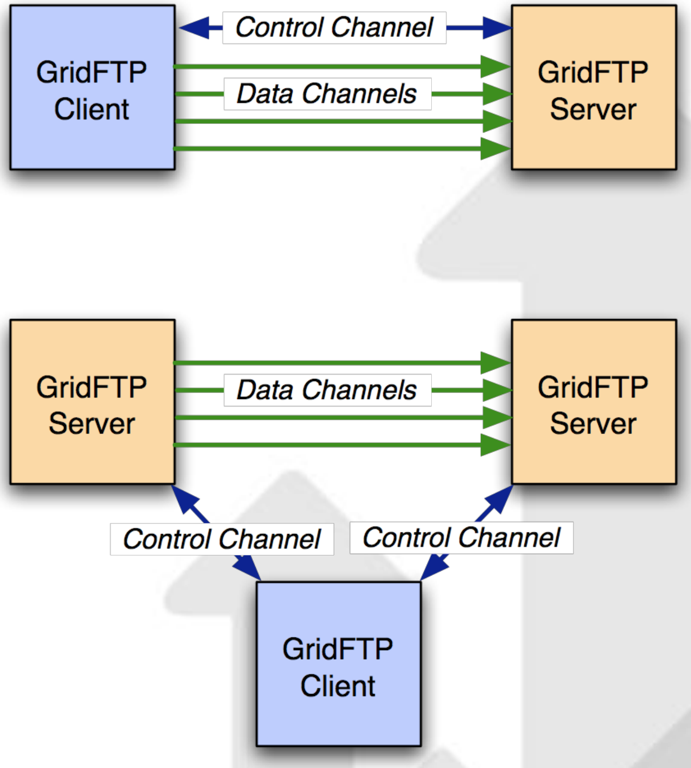
The University of Chicago

GridFTP

- **High-performance, secure data transfer protocol optimized for high-bandwidth wide-area networks**
- **Based on FTP protocol - defines extensions for high-performance operation and security**
- **Multiple independent implementations can interoperate**
  - Fermi Lab and U. Virginia have home grown servers that work with ours

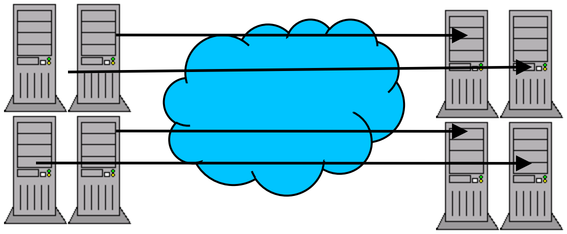
GridFTP

- **Two channel protocol like FTP**
- **Control Channel**
  - Communication link (TCP) over which commands and responses flow
  - Low bandwidth; encrypted and integrity protected by default
- **Data Channel**
  - Communication link(s) over which the actual data of interest flows
  - High Bandwidth; authenticated by default; encryption and integrity protection optional



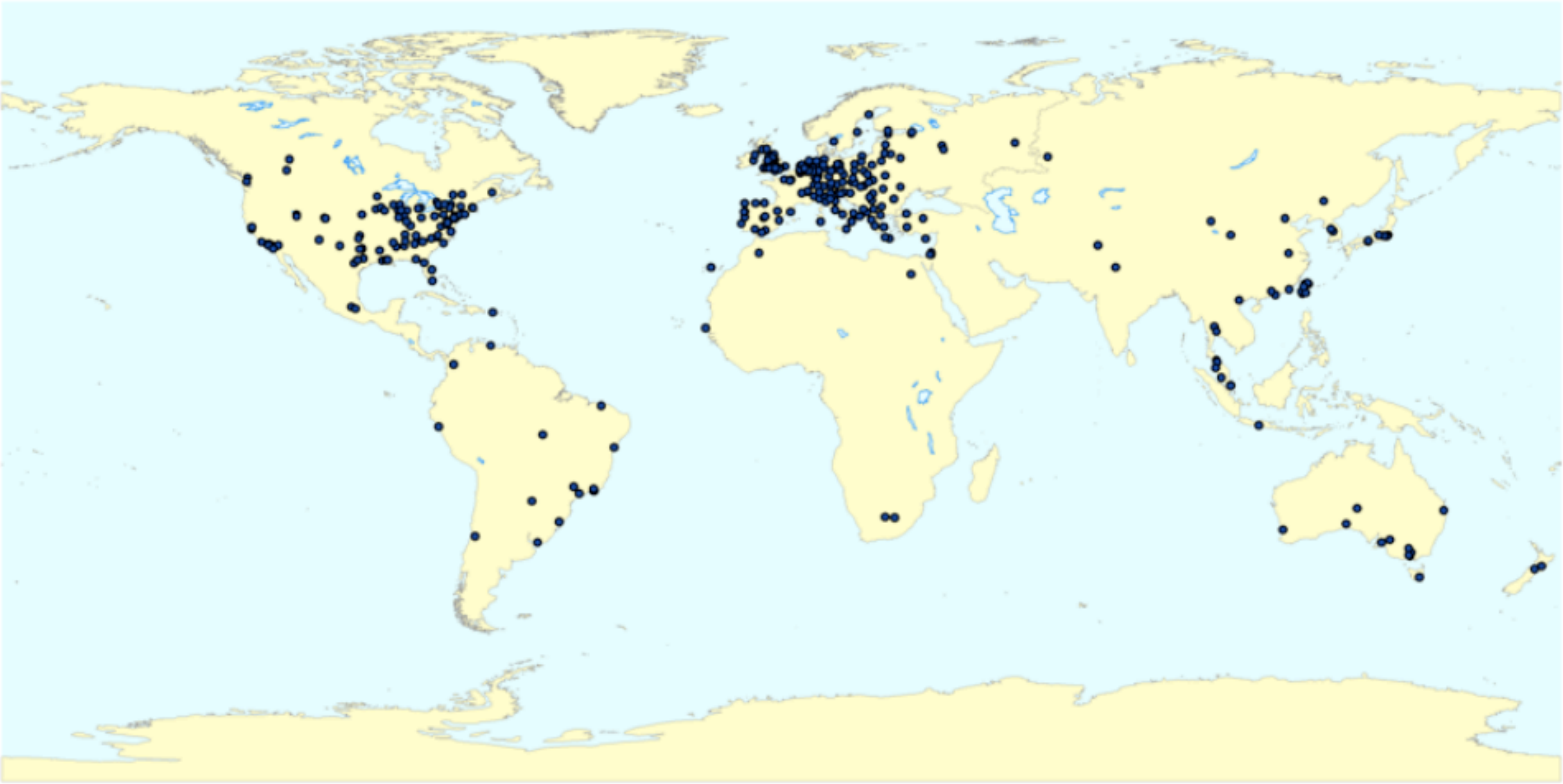
Globus GridFTP

- Parallel TCP streams, optimal TCP buffer
- Non TCP protocol such as UDT
- Cluster-to-cluster data movement



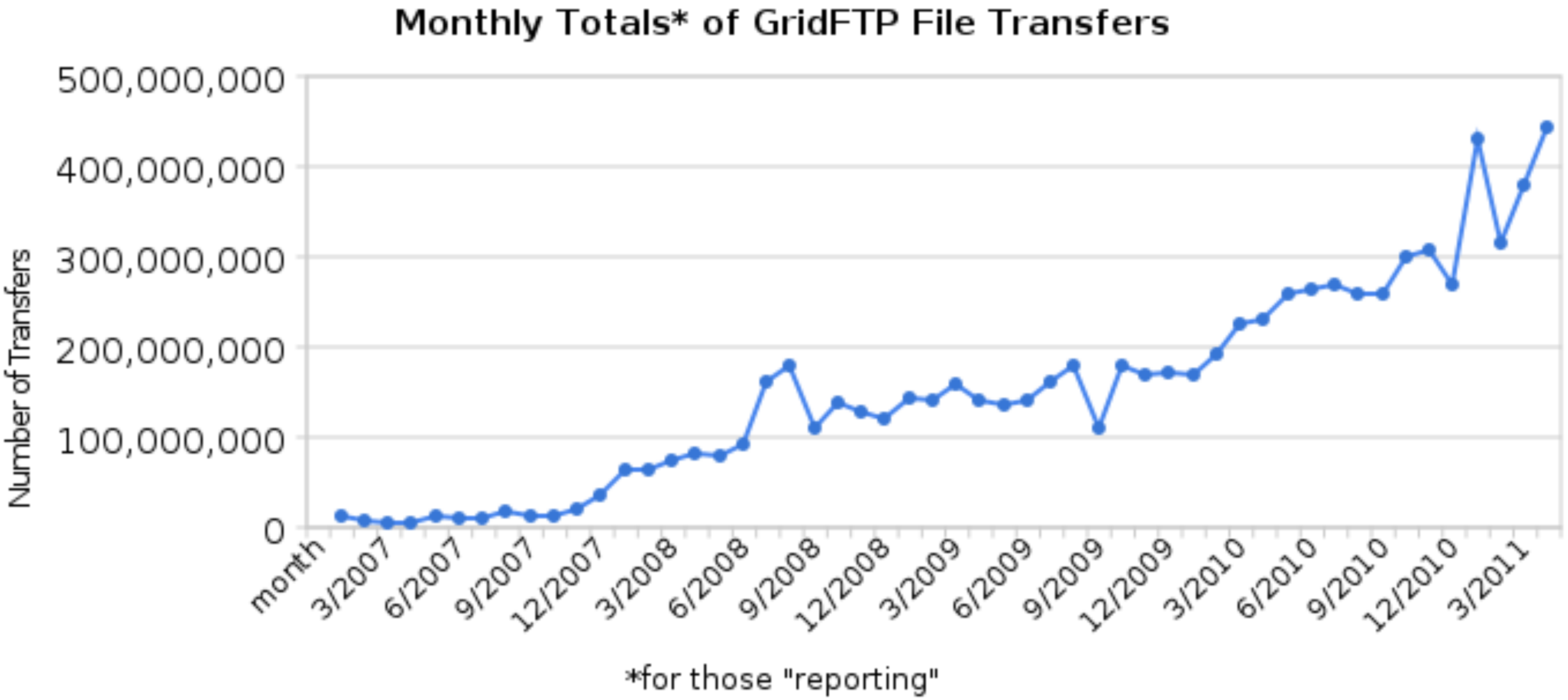
- SSH, Grid Security Infrastructure (GSI)
- Transfer checkpointing

GridFTP Servers Around the World



Created by Tim Pinkawa (Northern Illinois University) using MaxMind's GeoIP technology (http://www.maxmind.com/app/ip-locate).

GridFTP Usage



GridFTP: On Demand Service

- Transfer requests happen immediately
  - We do not queue, or delay transfers
  - An established session means an active transfer
- Transfer data as fast as possible
- Resources are limited
  - Data transfers are heavy weight operations
  - Sometimes hardware is too busy
    - Adding another transfer can cause thrashing
    - Collective system throughput goes down

Why Doesn’t GridFTP Queue

- Backward compatibility with legacy FTP
- Even for an idle session
  - Active TCP control channel
    - Part of the 959 protocol.
    - A session is defined by a TCP connection
  - Fork/setuid process
    - Robustness
    - File system/OS permissions
  - OS buffer space
    - Data channels require large TCP OS buffers

If GridFTP Always Said Yes

- OOM: the out of memory handle
  - OS optimistic provision of TCP buffers
  - Random processes will be killed
  - Meltdown
- Shared FS overuse
  - Pushing the I/O throughput beyond optimal
  - Causing OOM on IOD machines
- Shares of bandwidth too small
  - 1 Million transfers at 500b/s each?
  - OR 10 transfers at 100Mb/s each

Simultaneous Sessions

- Goal: Collective throughput
  - Entire servers bytes transferred / time
    - Not the number of transfers at once
- Only reasons for more than 1 connection
  - Provide an interactive service for many
  - One session does not use all of the local resource
    - The remote side is the bottleneck
  - Hide control messaging overhead in another sessions data transfer payload

Resource Protection

- GridFTP
  - Connection rejection is a feature
  - It SHOULD say no
  - Intended to scale to system transfer rates
  - To scale up add more data nodes
- Limits need to be in place to protect
  - Knowing it is ok to say ‘no’ is step 1

Connection Caps

- As a function of system memory
  - Cap = |mem| / (2MB + avg(BWDP))
  - Never more than |mem| / 4MB
- As a function of system bandwidth
  - Cap = min(FS.BW, Net.BW) / (Target average transfer rate)

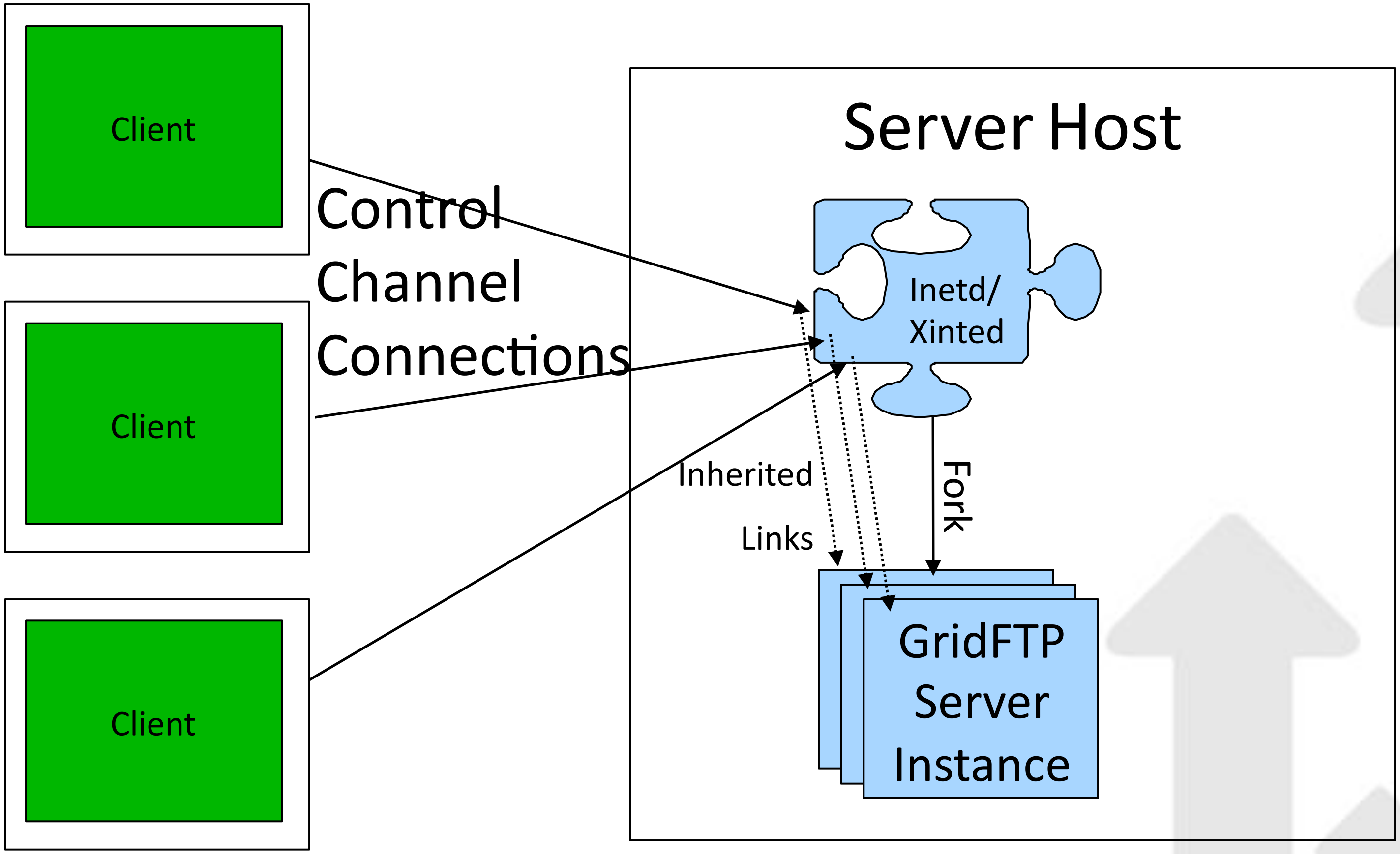
```
service gsiftp
{
        instances          = 20
        socket_type        = stream
        wait               = no
        env               += GLOBUS_LOCATION=…
        env               += LD_LIBRARY_PATH=…
        server             = /usr/local/globus-5.0.3/
sbin/globus-gridftp-server
        server_args        = -i -p 2811
        disable            = no
}
```

```
% globus-gridftp-server –connection-max 20
```

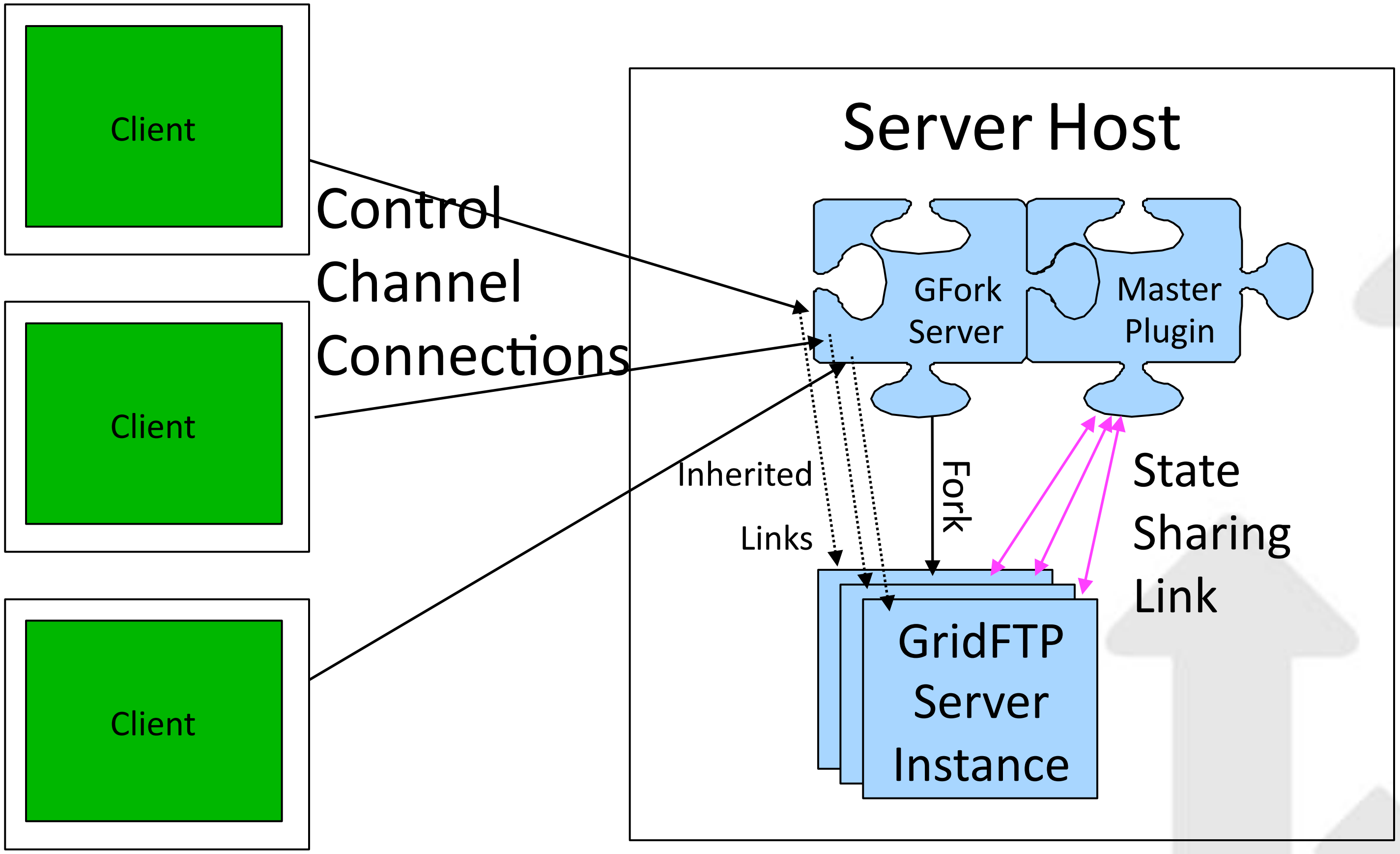
Resource Control

- To access the server
  - A user must be authenticated
  - Have read and write permissions and
  - Respect the total connection limit
- But beyond these requirements, there is no management or control
  - A user can hold a connection open indefinitely
  - Move an unlimited number of files (barring disk space or system quota constraints).
- A more flexible management is needed limit, prioritize and control

Inetd/Xinted



Globus Fork (GFork)

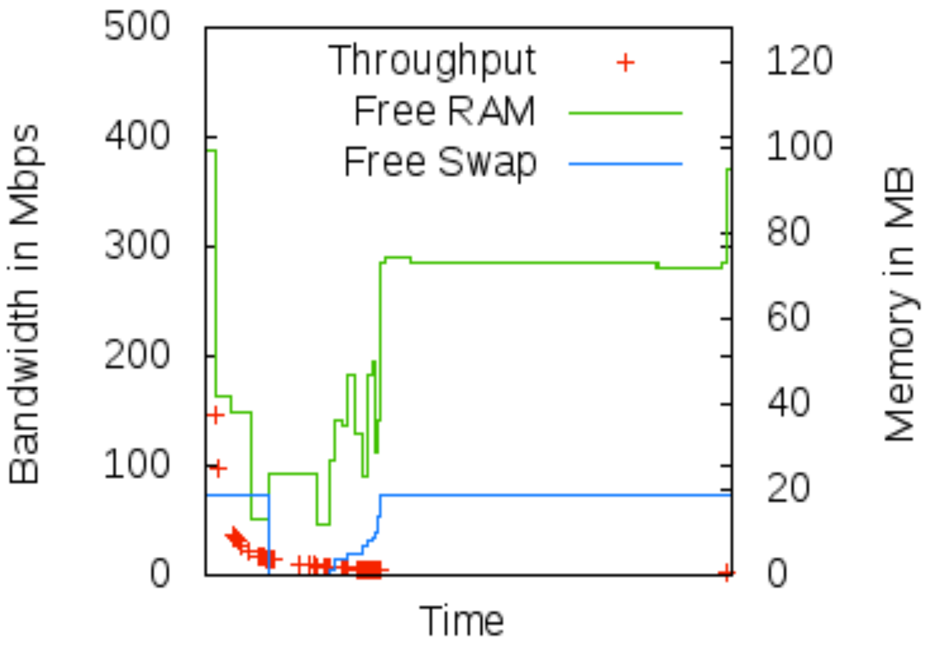


GFork Memory Manager

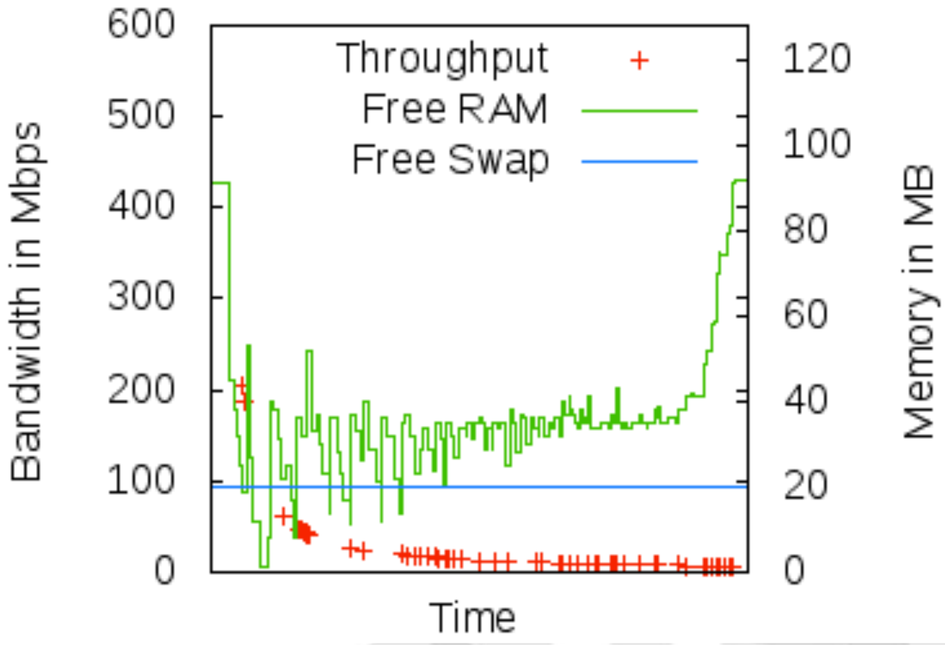
- Dynamically rations memory
  - 10% of the allowed connections get 90% of the memory
  - Remaining session get half of available memory
- Allows for high connection limits
  - |mem| / 2MB
- This limitation is different from the original connection limitation
  - Based on amount of used memory, not a static value based on total system memory.

Memory Management

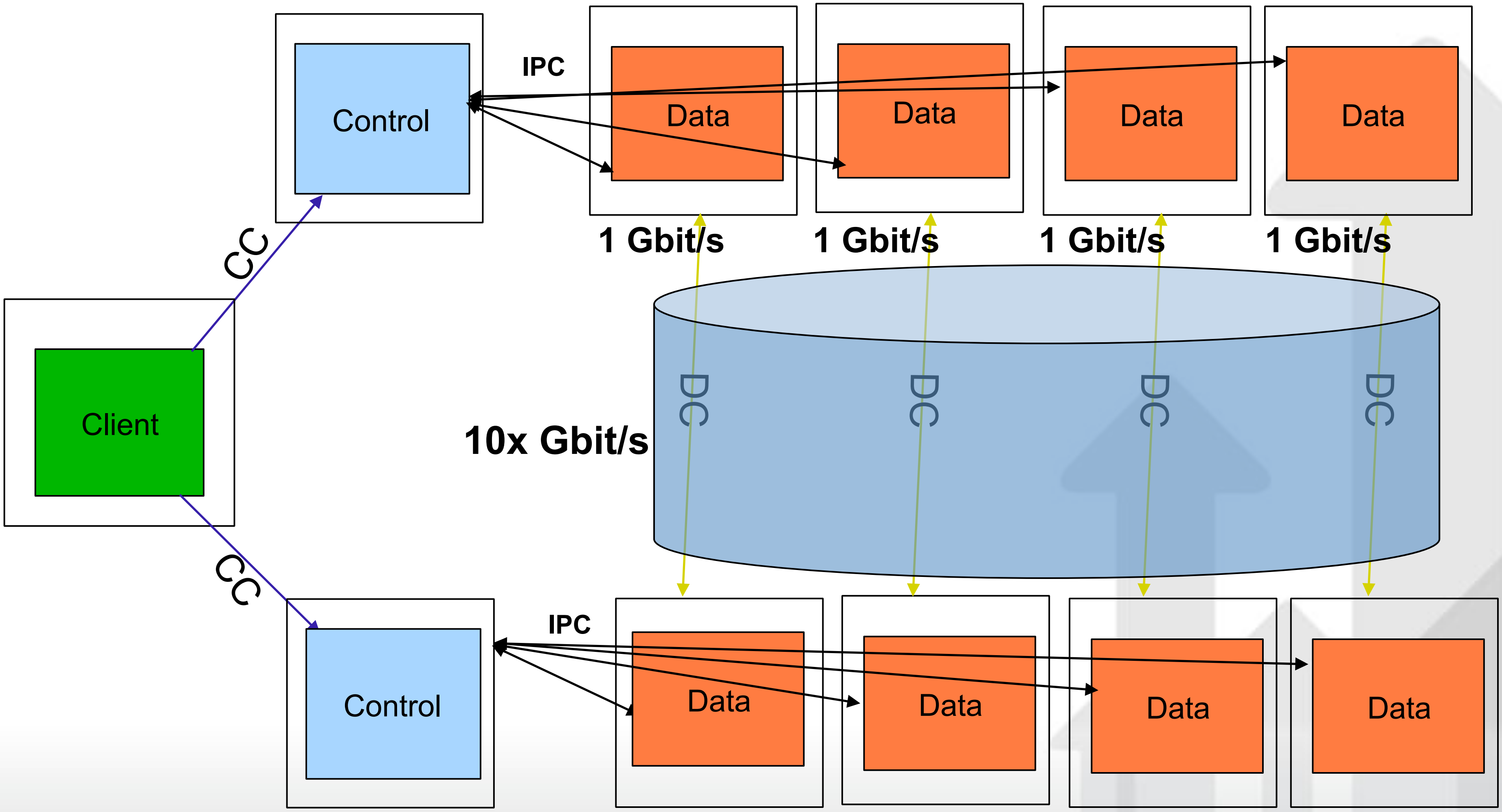
Performance without Memory Management


64 client, 128 MB RAM,
110.32 Mbps Total Throughput

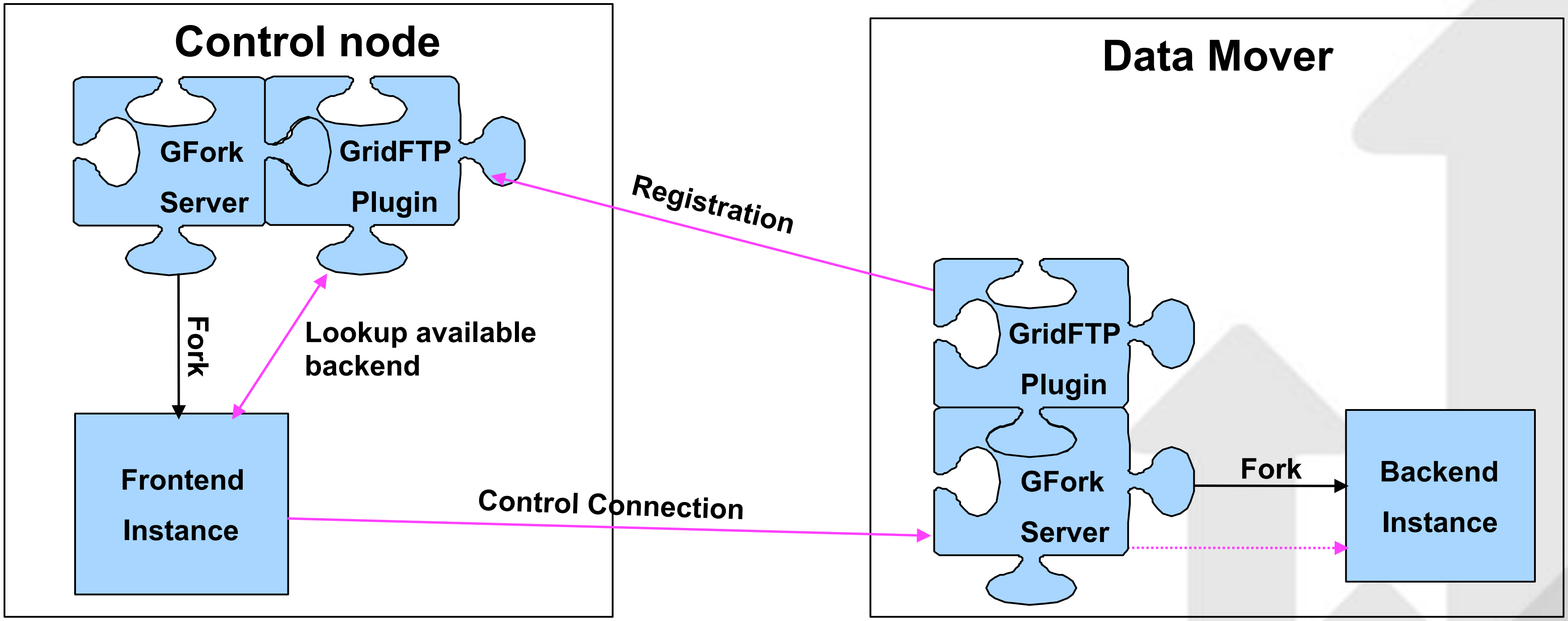
Performance with Memory Management


64 client, 128 MB RAM,
418.76 Mbps Total Throughput

Striped Data Movement



Dynamic Data Movers



Future work

- This is a start
- More sophisticated resource management capabilities are needed
- Better than best-effort service
- Service guarantees

Questions?