

# GMount: Build Your Grid File System on the Fly

Nan Dun, Kenjiro Taura and Akinori Yonezawa

Graduate School of Information Science and Technology, University of Tokyo

## Background

### Popular world-wide Grid computing

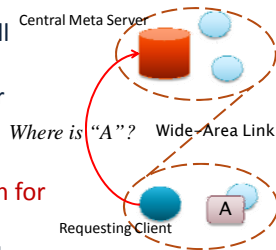
- Rich computing resources distributed over the world
  - InTrigger, Japan. <http://www.intrigger.jp>
  - T2K, Japan. <http://www.cc.u-tokyo.ac.jp/ha8000/>
  - Tsubame, Japan. <http://www.gsic.titech.ac.jp/~ccwww/>
  - Grid5000, France. <http://www.grid5000.fr>
  - DAS-3, Netherland. <http://www.cs.vu.nl/das3/>

### Increasing data sharing demands

- Distributed file systems (DFS)
  - Inner-cluster: NFS, PVFS, LusterFS, GlusterFS
  - Inter-clusters: Gfram, HadoopFS

## Problems of Conventional DFS

- Fixed resources at deploy time
- Administration Effort
  - Installation, configuration, NAT/Firewall
  - Need Privileges
- Potential problems of central meta server
  - Single-point-of-failure
  - Single-point-of-load
  - High-latency metadata operations even for close files
- Potential problems of recovery from crashes

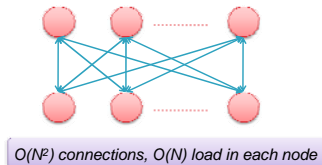


1

## Basic Ideas

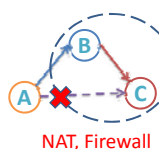
Goal: Every node sees the union of all other nodes' /share in its local /mnt

### Naïve All-Mount-All



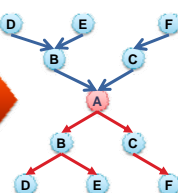
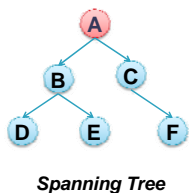
### Mount Primitives

If A mounts B and B mounts C, then:



- Both A and B can access C
- Load balancing
- Across NAT, firewall

### Scalable All-Mount-All



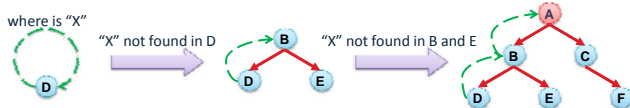
All-Mount-One  
Every descendant sees root's /mnt in its local /mnt

One-Mounts-All  
Root sees the union of all descendants' /share in its /mnt

K: The number of children  
Every node spans  $O(K)$  connections  
 $O(\log_k N)$  connections in total  
 $O(K)$  sshfs processes in each node  
 $O(K)$  sshd processes in each node

```
B$ sshfs A:/mnt B:/inter B:/share /mnt
C$ sshfs A:/mnt C:/inter C:/share /mnt
D$ sshfs B:/mnt D:/share /mnt
F$ sshfs C:/mnt F:/share /mnt
A$ sshfs B:/inter C:/inter A:/share /mnt
B$ sshfs E:/share D:/share B:/share /inter
C$ sshfs F:/share C:/share /inter
```

### Example: File lookup at node D



If the spanning tree is network topology aware, then the file operations are locality-aware in terms of network affinity.

## Future Work

- To enhance fault-tolerance and cache consistency
- To improve limited SSH transfer rate

3

## GMount DFS Available at <http://gxp.sourceforge.net>

- Instantaneous
  - Server-less, userspace, no configuration
  - Short construction time
- Grid-Enabled
  - Overcome NAT/Firewall
- Locality-Aware file operations
- Data recoverable from local filesystem

## Building Blocks

### File System in Userspace (FUSE)

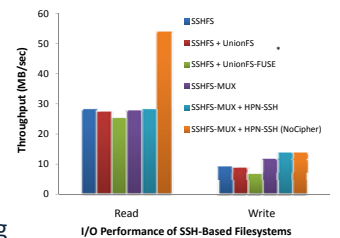
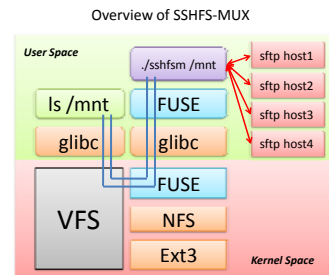
- A framework for quickly building userspace filesystems
- Available in most Linux machines (kernel version > 2.6.14)

### SSHFS Multiplex (SSHFS-MUX)

- Effort-less remote filesystem
  - As easy and good performance as SSHFS
  - Mount multiple remote hosts simultaneously
- Data recoverable from LocalFS

### Grid and Cluster Shell (GXP)

- Parallel and distributed shell
- Embarrassingly parallel processing
- Master-Worker framework

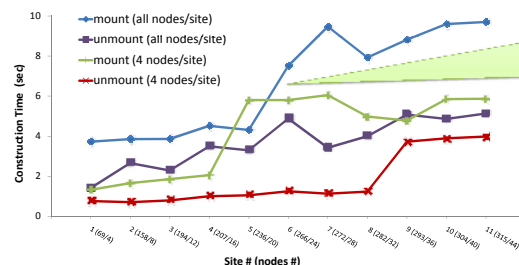


\* High Performance SSH/SCP (HPN-SSH)  
<http://www.psc.edu/networking/projects/hpn-ssh/>

2

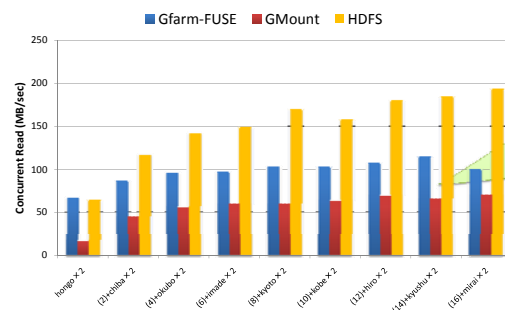
## Experiments

### Filesystem Construction Time



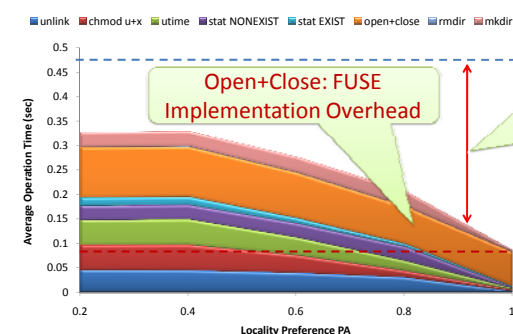
Scales well with system size  
Sensitive to network latency

### Parallel I/O Performance



I/O performance is related to:  
1) SSH throughput  
2) Spanning tree structure

### Metadata Operation Performance



Open+Close: FUSE Implementation Overhead

Saved Network Latency because of locality-aware meta-operation

4