# Remote File System Suite

## Softwarepraktikum für Fortgeschrittene

Michael Kuhn

Parallele und Verteilte Systeme
Institut für Informatik
Ruprecht-Karls-Universität Heidelberg

2009-07-07

# FUSE

- Goal was to implement a global network file system
  - Needed to implement the underlying network file system first
- Should be implemented as a FUSE file system
  - Runs in user space
  - Relatively easy to implement
  - Relatively easy to maintain
- High performance
  - Microscope delivers 1 GB/s
- Transparent
  - Use existing file systems as storage
  - No striping of files

# Overview

- `rfsd` – Remote File System Daemon
  - Low-level network file system
- `librfs` – Remote File System Library
  - Abstracts protocol implementation
- `rfsc` – Remote File System Client
  - Basically a simple throughput and metadata benchmark
- `grfs` – Global Remote File System
  - High-level global network file system

- A separate protocol was designed
- Existing protocols did not meet the requirements
- SSH
  - Does not support unencrypted data channels
  - Data encryption makes transfers too slow
    - Not possible to deactivate the encryption
- FTP
  - Only possible to write a complete file or append data to it
  - File listings are hard to parse, because their format is not well-defined

- Implement our own protocol
- Separate control and data channels
  - No encryption
  - Control channel can be encrypted via SSH forwarding
- Should be as fast as possible
  - Microscope pumps out 1 GB/s
  - $6 \cdot 2$ servers
  - $\Rightarrow$ 100-200 MB/s
- Should be as transparent as possible
  - Use underlying local file system
  - Do not stripe files across servers
- Should be as safe as possible
  - Support replication

- Basically provide remote access to the local file system
  - Protocol very similar to POSIX
    - `pread()`, `pwrite()`, ...
  - Plus some fancy features, of course :-)
- Fully multi-threaded
  - Each connection handled in its own thread
  - Long-running operations do not block other connections
- Replication
  - Master-slave concept
  - One master, multiple slaves
  - All operations are replicated in a background thread
    - Pushed into the thread when the operation begins
    - Check whether the thread finished when the operation ends

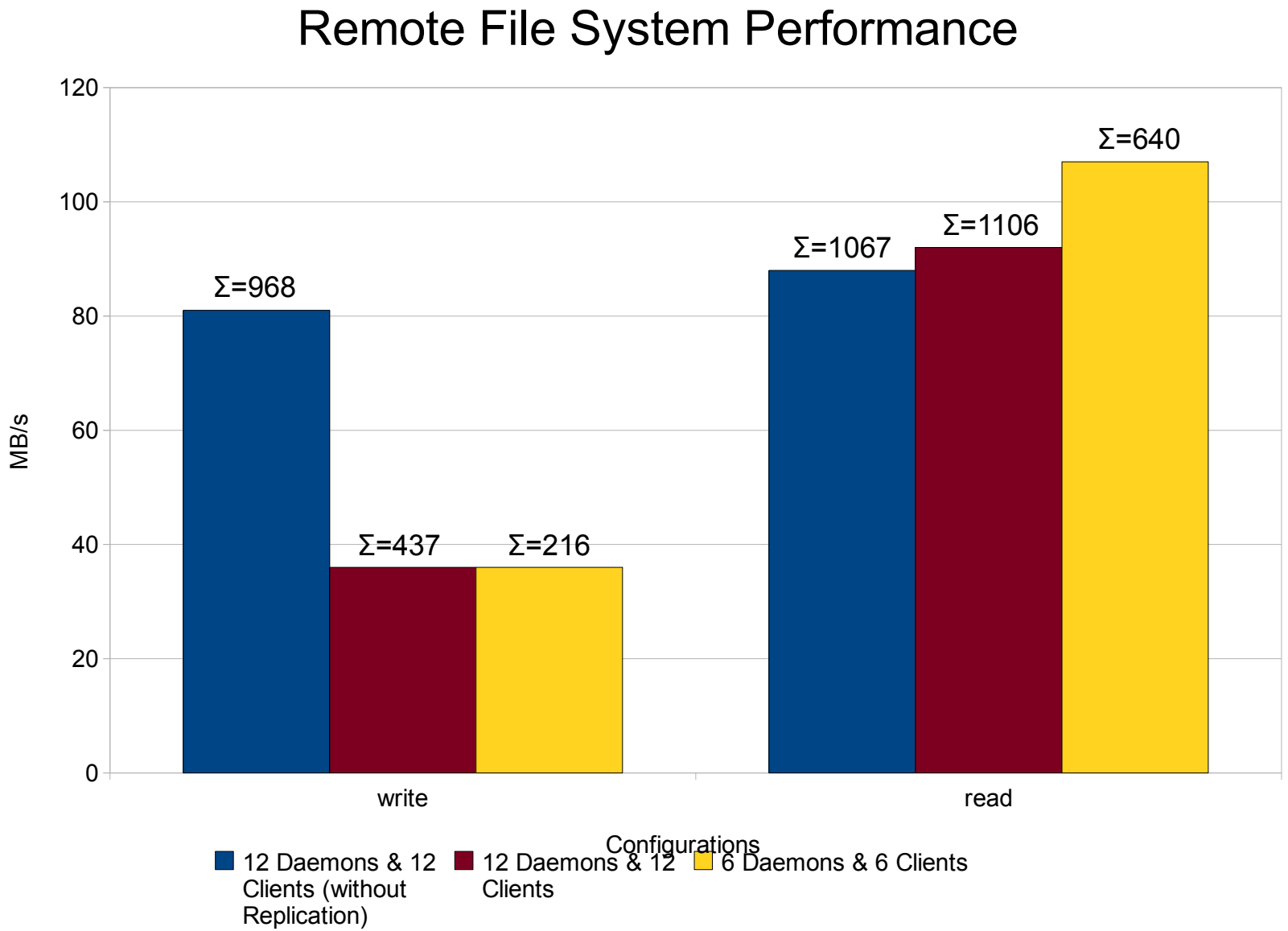
- Logging
  - All operations are logged when a slave is offline
  - Kept in memory and written to log file
  - Replayed when slave comes online
- `chroot`-like restrictions
  - All accesses can be restricted to a sub-tree of the file system

- Hide all the “ugly” implementation details :-)
- Good error reporting via GError
  - Part of GLib
- Some operations require multiple steps
  - For example: `rfs_read()` (“open”), `rfs_read_do()` (“pread”), `rfs_read_end()` (“close”)

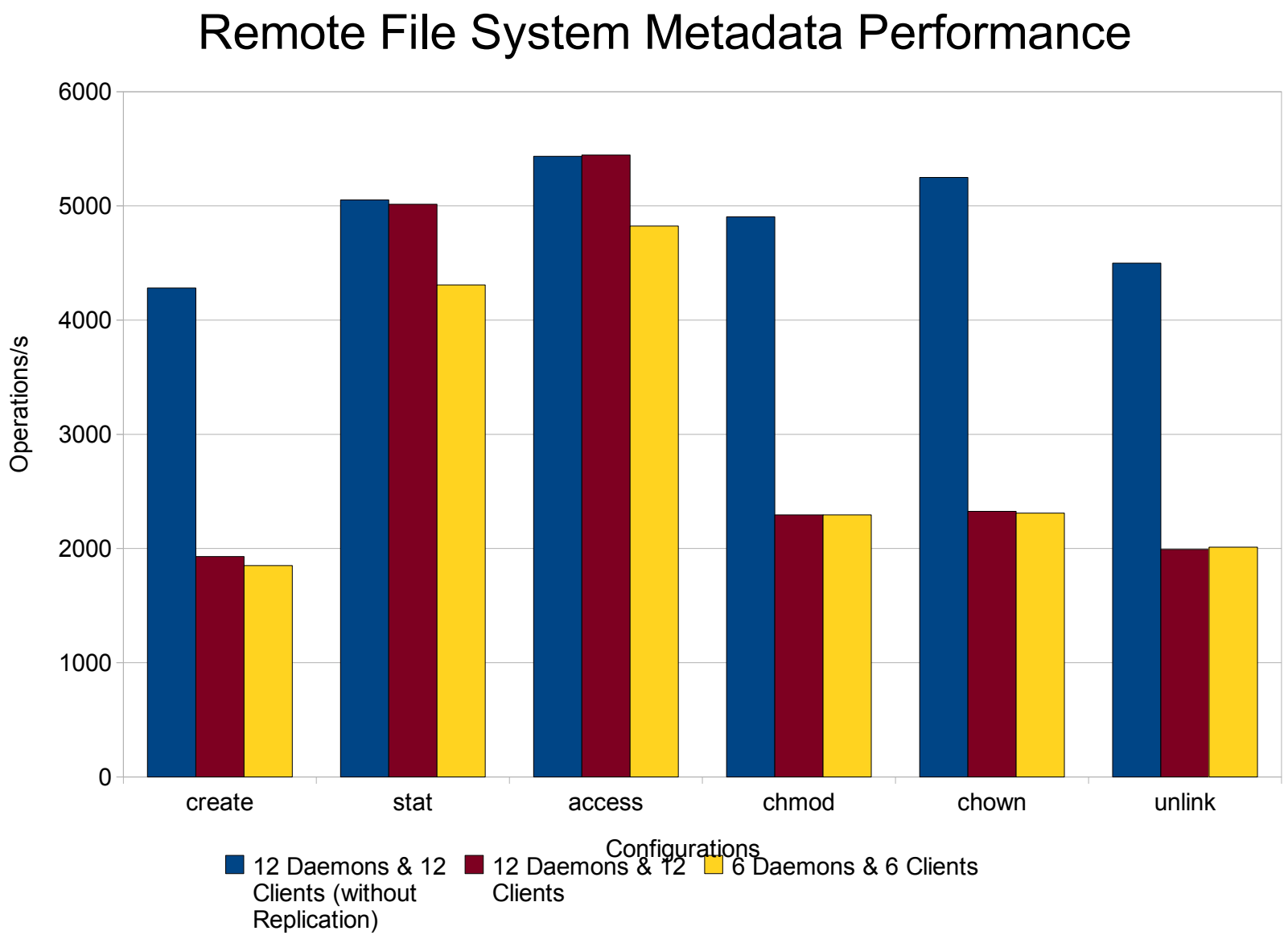
- Merge multiple file systems into one global namespace
- Example:
  - `serv1` has directory `/foo`, `serv2` has directory `/bar`
  - `$ grfs serv1:6666 serv2:6666 /grfs`
  - `$ ls /grfs`
  - `> foo bar`

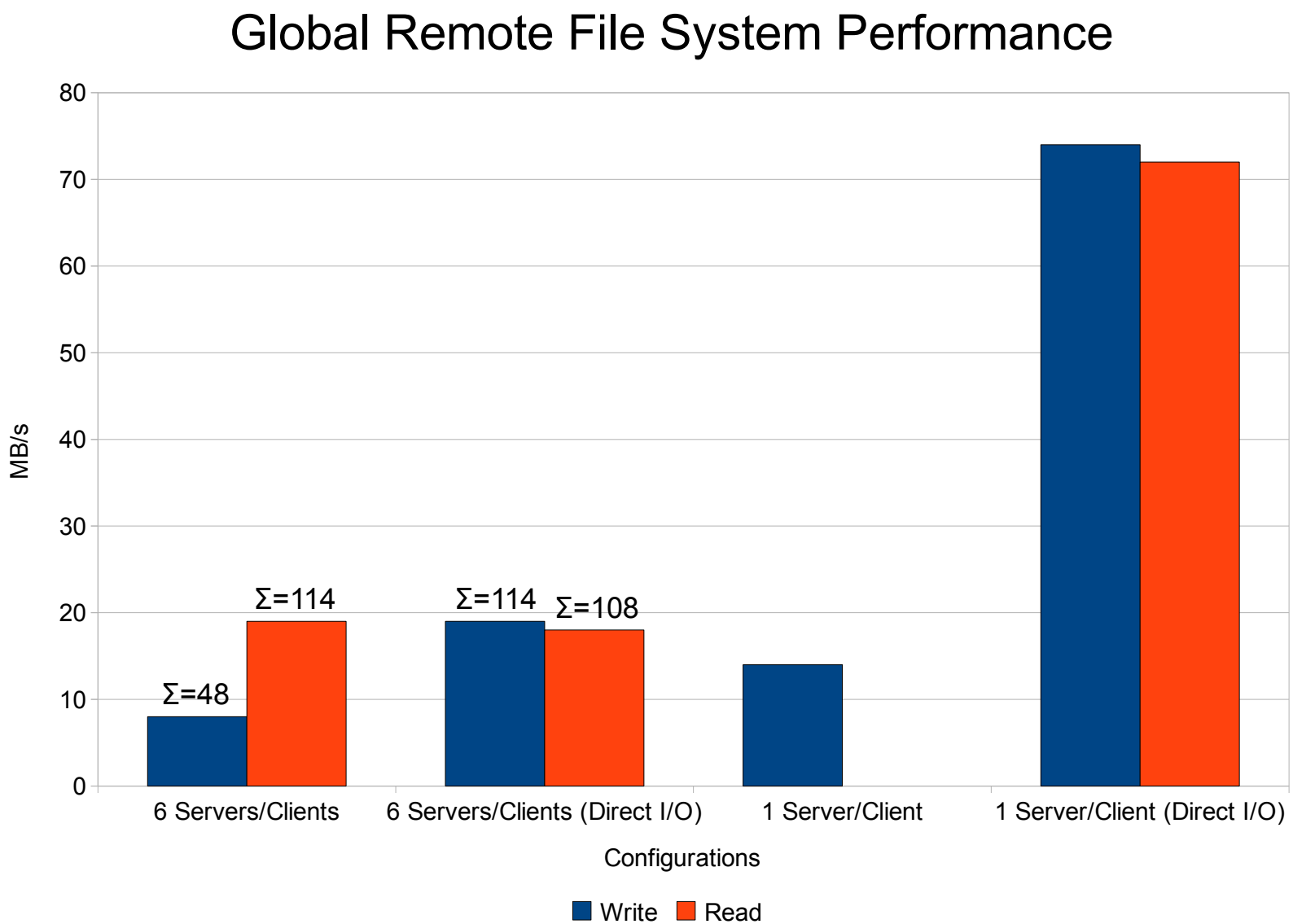
- High Availability
  - Continues to work when servers go offline
  - Provides a partial view of the file system
- `setuid`-like functionality
  - Supports FUSE's `allow_other` option
- Fast reads and writes
  - Keeps the state of read and write operations
  - Accesses to the same file are handled with very little overhead

- 12 Daemons & 12 Clients
  - One daemon and one client on each machine
- 6 Daemons & 6 Clients
  - One daemon on each of the first six machines
  - One client on each of the last six machines
- 6 Daemons & 12 Clients
  - One daemon on each of the first six machines
  - Two clients on each of the last six machines
- All numbers are per-client

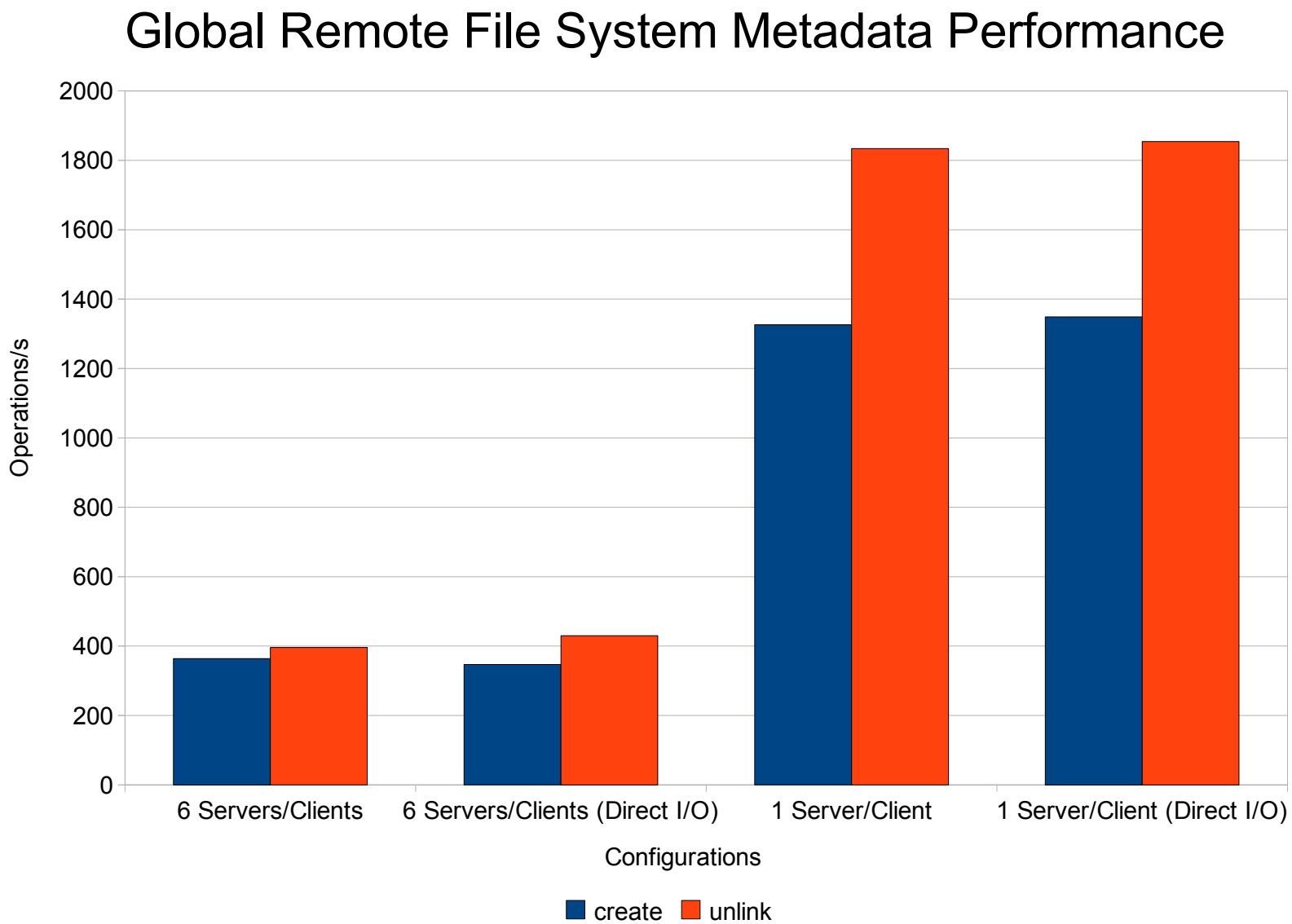



Replication: Significant performance drop


Remote File System Metadata Performance
Operations/s
6000
5000
4000
3000
2000
1000
0
create
stat
access
chmod
chown
unlink
Configurations
12 Daemons & 12 Clients (without Replication)
12 Daemons & 12 Clients
6 Daemons & 6 Clients




1 GBit/s maximum throughput – Read: 1.7 GB/s (cache)


Global Remote File System Metadata Performance
Operations/s
2000
1800
1600
1400
1200
1000
800
600
400
200
0
6 Servers/Clients
6 Servers/Clients (Direct I/O)
1 Server/Client
1 Server/Client (Direct I/O)
Configurations
create
unlink

- Global Remote File System performance is limited by FUSE
  - Read and write buffers are at most 128 KB in size
  - FUSE 2.8.0 pre-release supports up to 512 KB