



pNFS

Sam Falkner
Sun Microsystems, Inc.



pNFS

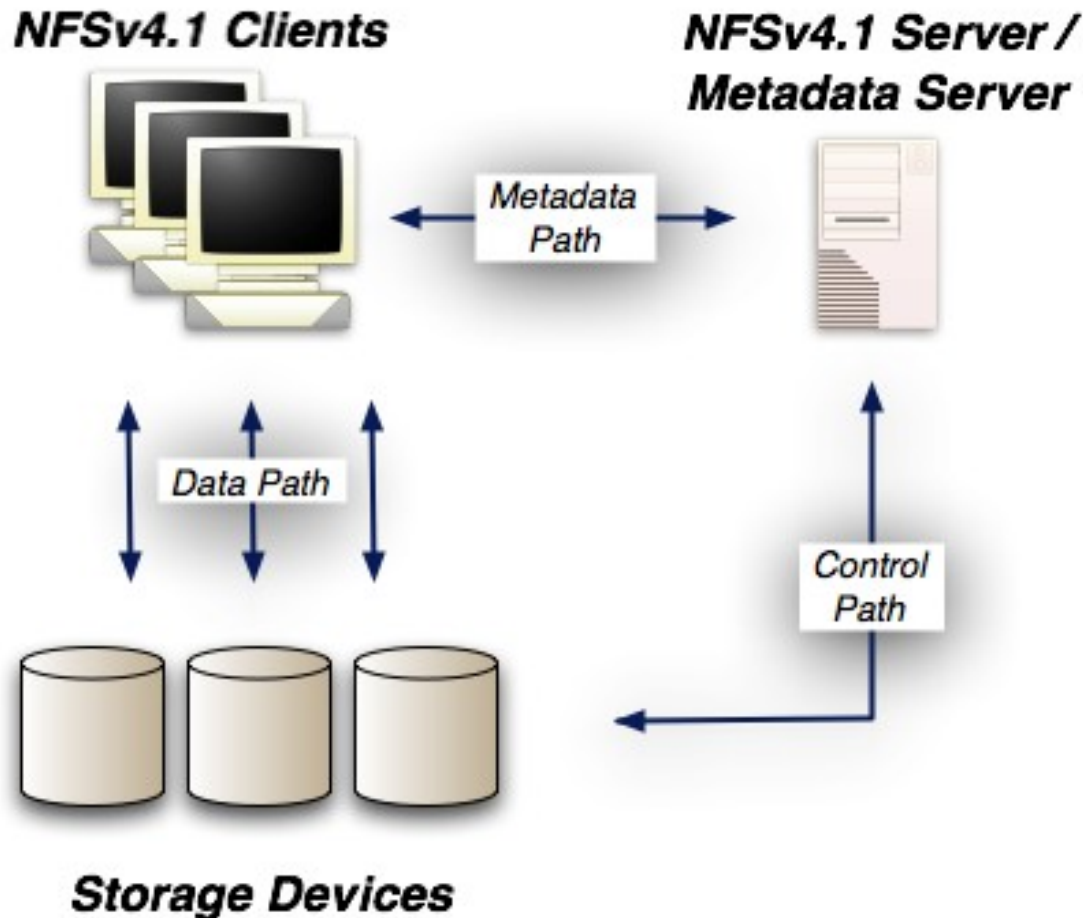
- Extension to NFS to enable *parallel data transfer*
 - > A file resides on one or more NFS servers
 - > Files in a directory may have no physical relationship
 - > /pnfs/file1 can be on dserv1 and dserv2, /pnfs/file2 can be on dserv3
- Provides for a single, unified name space
 - > Single name space, mount spans many data servers
- Striped (RAID-0) data layouts are defined
 - > Simple multipathing supported

NFSv4.1 and pNFS

- IETF Draft standard
 - > <http://www.nfsv4-editor.org/draft-14/draft-ietf-nfsv4-minorversion1-14.html>
- Major pieces of NFSv4.1 include:
 - > pNFS
 - > Sessions
 - > Directory Delegations
- Standardization expected in early '08

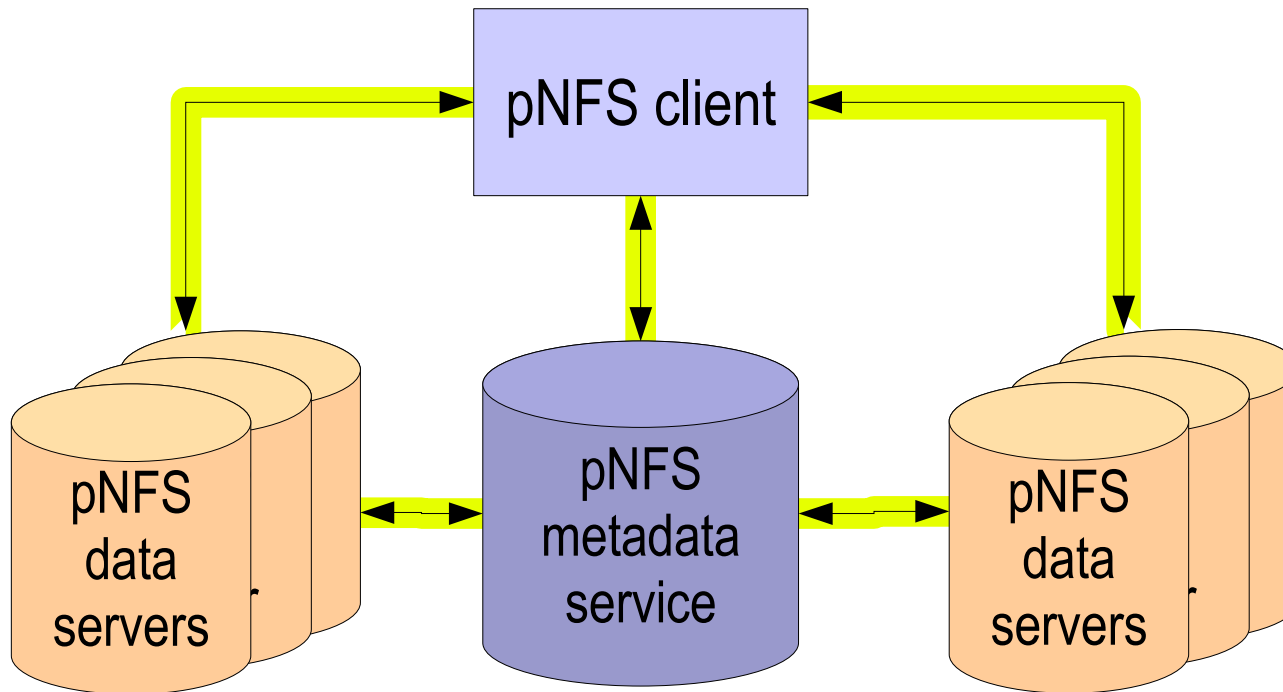
pNFS Data Access Methods

Files (NFSv4.1) (*), **Objects** (OSD), and **Blocks** (iSCSI)

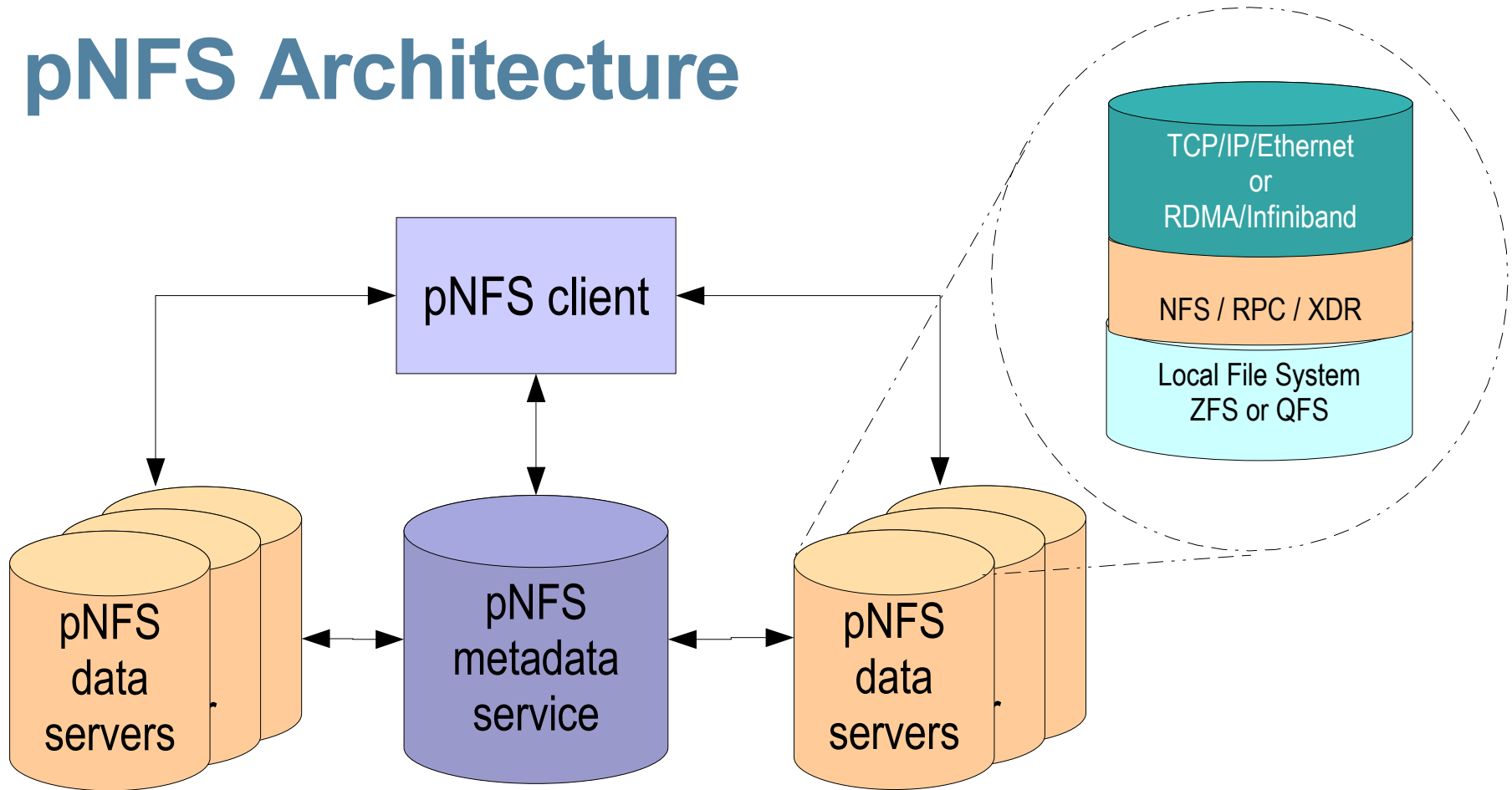


Security

- Kerberos support on metadata, data and control paths



pNFS Architecture



NFS-over-RDMA

- NFS data can be transferred via NFS-over-RDMA protocol using Infiniband transport
 - > Prototype allows 980MB/sec reads, 650MB/sec writes
 - > Benchmark done with traditional NFSv4.0 (not pNFS)
 - > *Each* parallel thread in pNFS could go approx this fast
 - > 980 MB/sec is very near physical bridge bandwidth of x2100
 - > Same systems = 235 MB/sec over TCP/IP/10G Ethernet
 - > RDMA transfers are *much* more efficient than TCP/IP
- New NFS-over-RDMA standard code expected in Solaris Q4CY07
- Complimentary to pNFS parallelism

Current status

- pNFS prototype code running now on Solaris Nevada
 - > Interoperability demonstrated at IETF Bakeathon 6/07
 - > IETF Bakeathon held 10/07
 - > Tested against other pNFS implementations
- Current implementation available for download:
<http://opensolaris.org/os/project/nfsv41/downloads/>
 - > Includes client, metadata server and data server code and binaries

Product Plans

- pNFS client based on Solaris Nevada
- Metadata server and data servers based on Solaris Nevada
- RAID-0 supported
- Integrated with NFS-over-RDMA
- Kerberos support (as always)
- Expected mid-2008

NFSv4.1 pNFS OpenSolaris Project

- Design and implementation is being done in the open
- <http://opensolaris.org/os/project/nfsv41/>
 - > Code and Binaries
 - > Documentation
 - > Email discussion list
 - > Flash demos
 - > pNFS Basics – A demo of the OpenSolaris prototype
 - > More to come...

How it works...

- A “*Layout*” manages the location of file data
- A client asks for a “*Layout*” when it wants to access the data
- The “*Layout*” contains a list of *deviceids* (shorthand references for the storage devices) and striping information
- The granting of a “*Layout*” gives the client the permission to access the data servers directly.

How it works... (cont.)

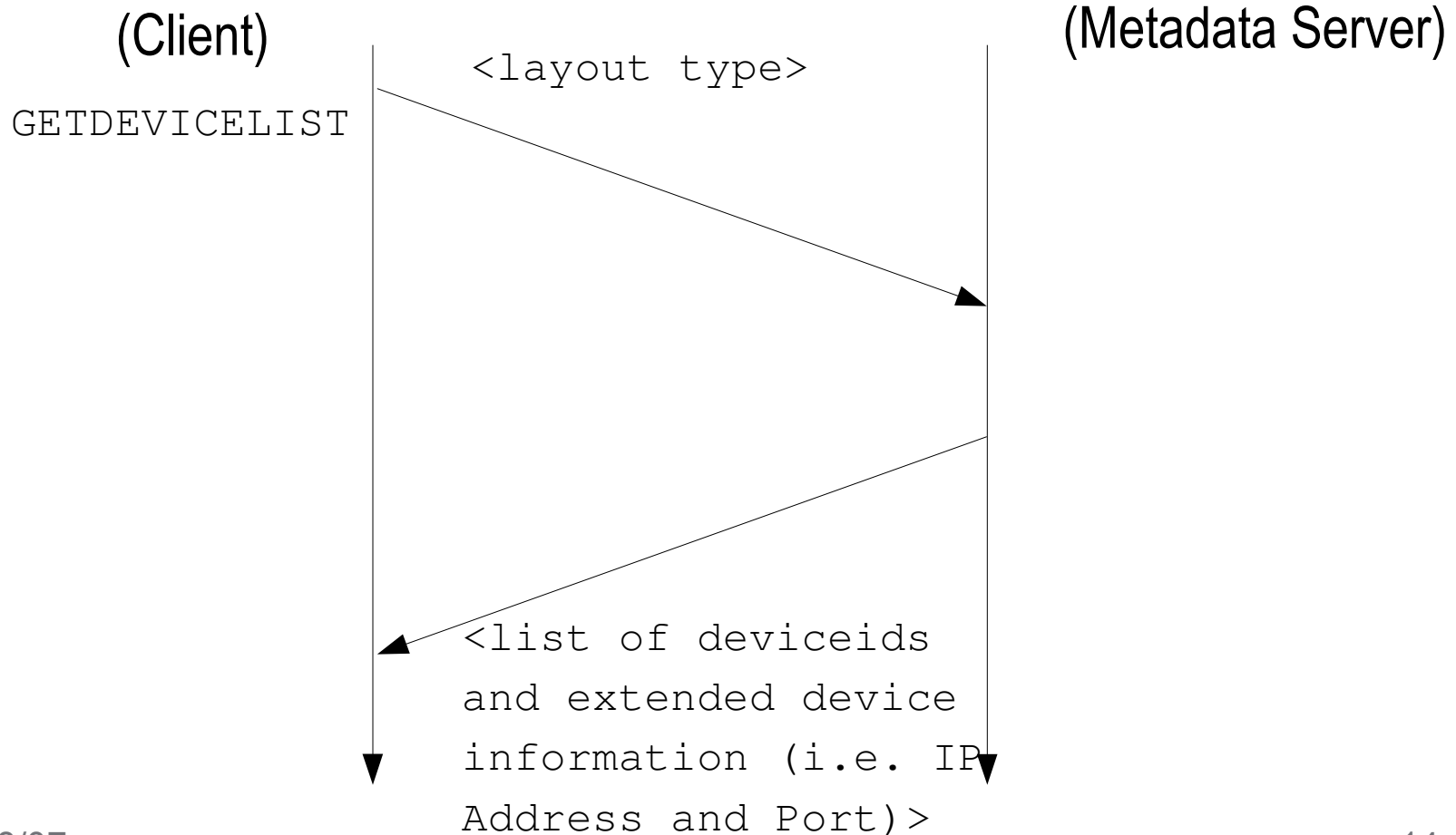
- New operations being introduced
 - > GETDEVICELIST
 - > GETDEVICEINFO
 - > LAYOUTGET
 - > LAYOUTCOMMIT
 - > LAYOUTRETURN
 - > CB_LAYOUTRECALL
 - > CB_RECALL_ANY
 - > CB_RECALLABLE_OBJ_AVAIL
- New attributes being introduced:
 - > fs_layout_type, layout_alignment, layout_blksize, layout_type, layout_hint, mdsthreshold

How it works... (cont.)

- Simplified example of how the Solaris prototype works:
 - 1. Mount a file system
 - 2. Open a file
 - 3. Read from the file

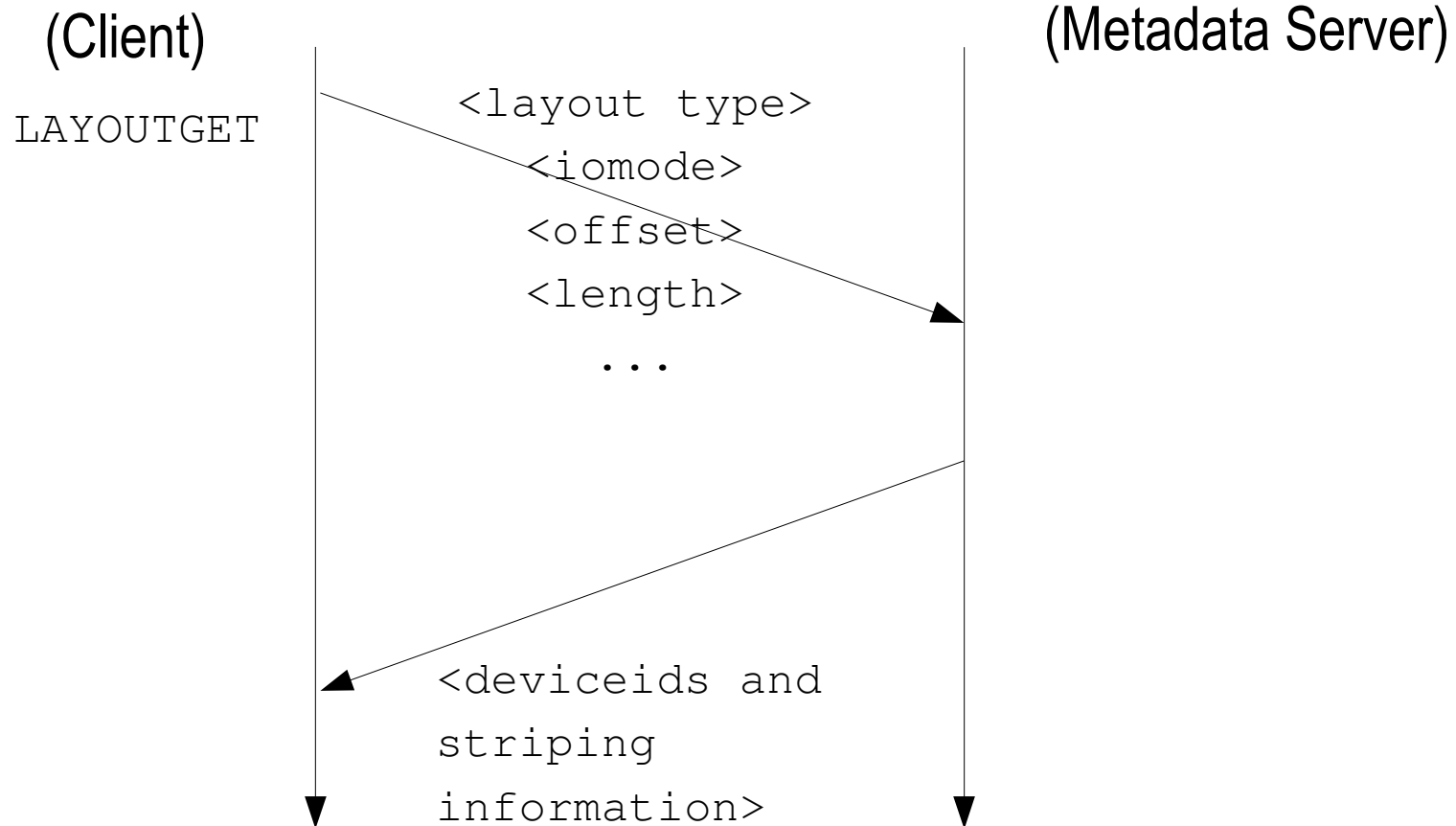
How it works... (cont.)

- Mount a file system “`mount mds : / /mnt`”



How it works... (cont.)

- Open a file “`open (/mnt/foo, ...)`”

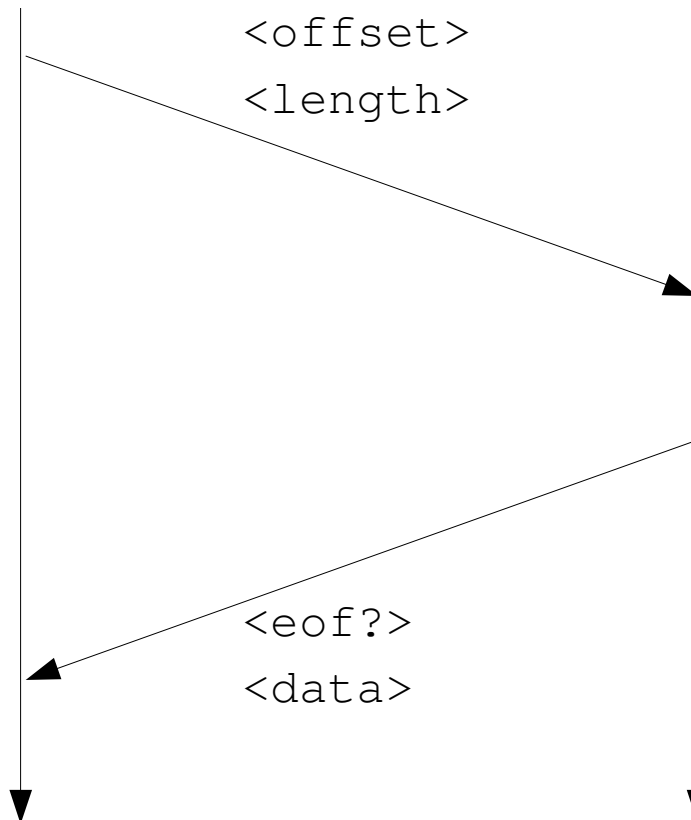


How it works... (cont.)

- Read from the file “`read (fd, ...)`”

(Client)

READ



(Data Server 1...n)

READ operations are sent in parallel to each data server needed to fill the request. The client aggregates the responses from each data server and returns the result to the user.

Questions?



Sam Falkner
sam.falkner@sun.com
<http://blogs.sun.com/samf>