2006 OpenIB Workshop
NFS-RDMA

NFS-RDMA

James Lentini
Network Appliance
jlentini@netapp.com

Tom Tucker
Open Grid Computing
tom@opengridcomputing.com

February 7, 2006

Agenda

- Overview
- NFS-RDMA Protocol
- Linux Client Architecture
- Linux Server Architecture
- Status and Future Work

NFS

- The **Network File System (NFS)** is a widely used distributed file system.
- NFS is layered on the **Remote Procedure Call (RPC)** and **External Data Representation (XDR)** protocols.
- Three major versions are in use today: v2, v3, and v4

NFS-RDMA

- **NFS-RDMA** is
  - an RPC-layer protocol that allows NFS to use RDMA networks (such as Infiniband and iWARP)
  - A transparent solution for applications, NFS protocol features, and NFS users
  - A significant performance boost to clients
    - Reduces client CPU overhead
    - Utilizes high-bandwidth, low-latency fabrics
  - A single-wire host cluster solution

Standardization

NFS-RDMA is being standardized in the IETF NFSv4 Working Group:

http://www.ietf.org/html.charters/nfsv4-charter.html

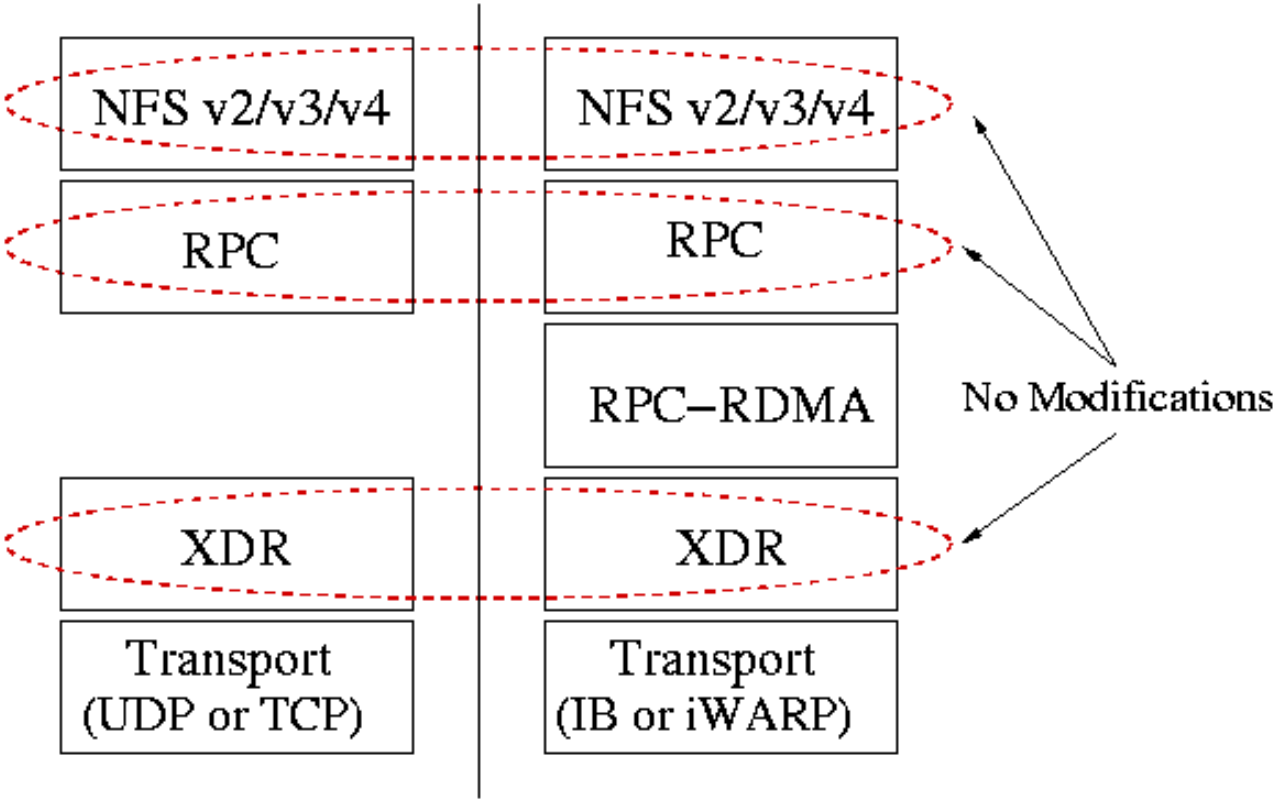
- RDMA Transport for ONC RPC
  - Describes the RPC-RDMA protocol for sending RPC messages on an RDMA transport
- NFS Direct Data Placement
  - Describes the NFSv2/v3/v4 mapping to RPC-RDMA operations

NFS-RDMA Implementations

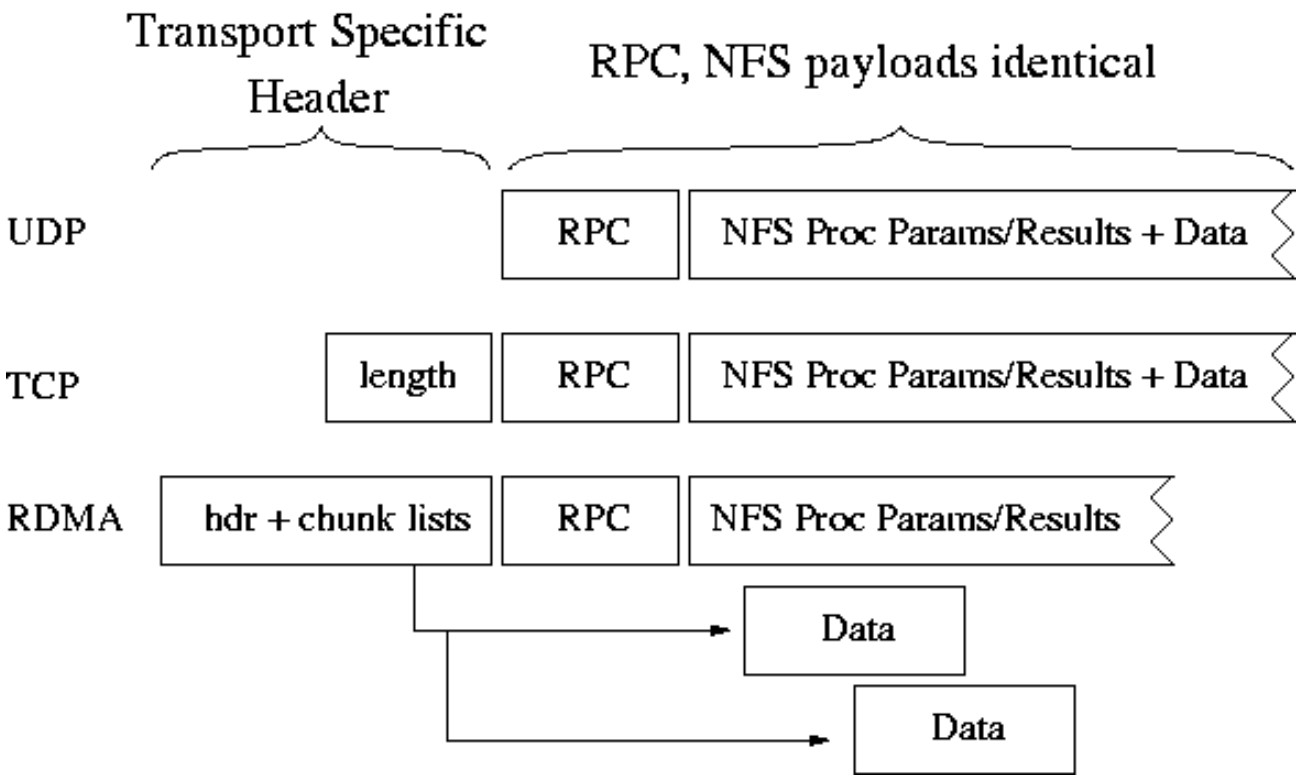
Multiple implementations:

- Linux
  - Client
  - Server
- Solaris
  - Client
  - Server
- NetApp
  - Server

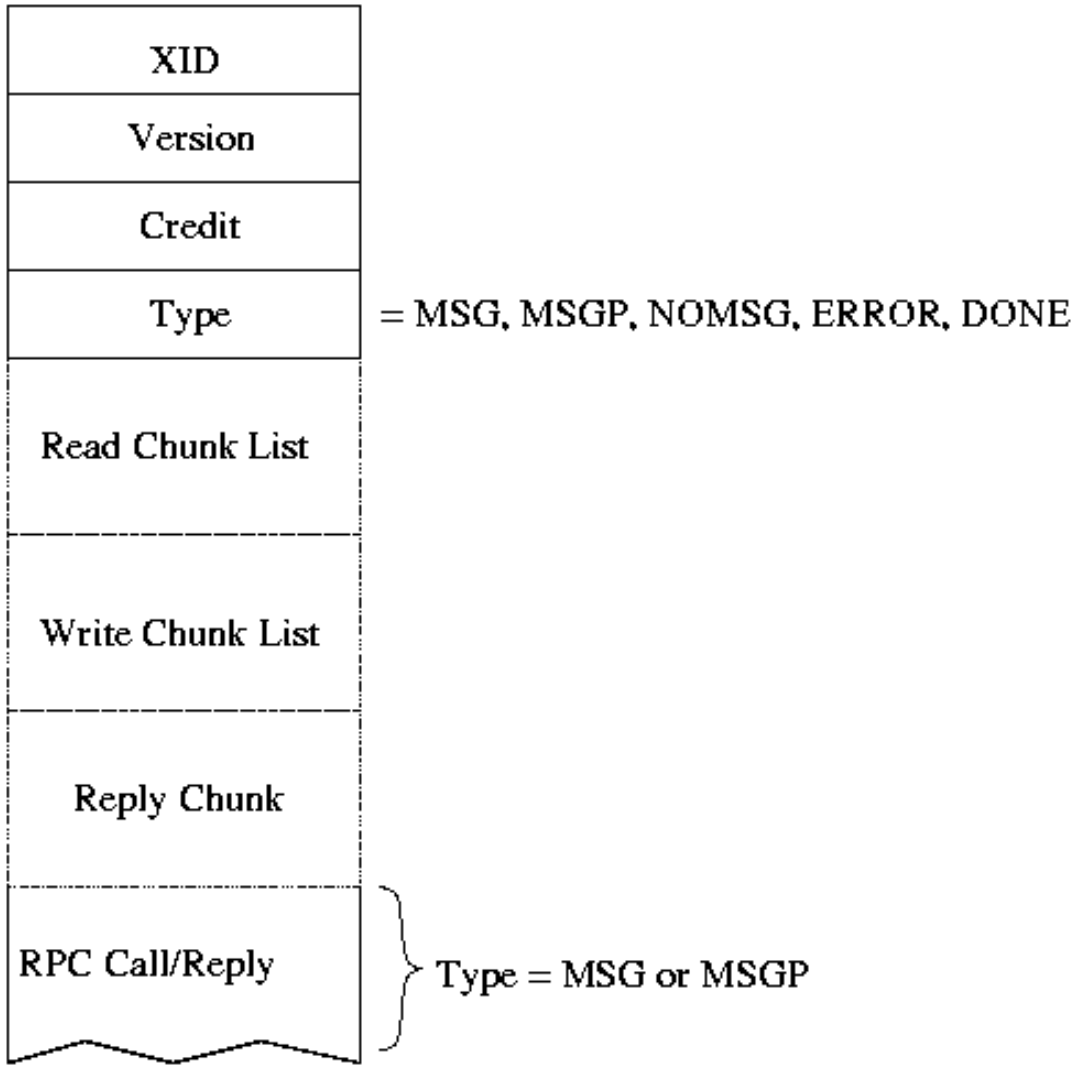
NFS vs NFS-RDMA



RPC vs RPC-RDMA



RPC-RDMA Header



RPC-RDMA Operations

- RPC-RDMA can transfer RPC packets using
  - RDMA Send
  - RDMA Read/Write
- NFS-RDMA restricts RDMA Read/Write usage
  - only the server can use RDMA Read/Write

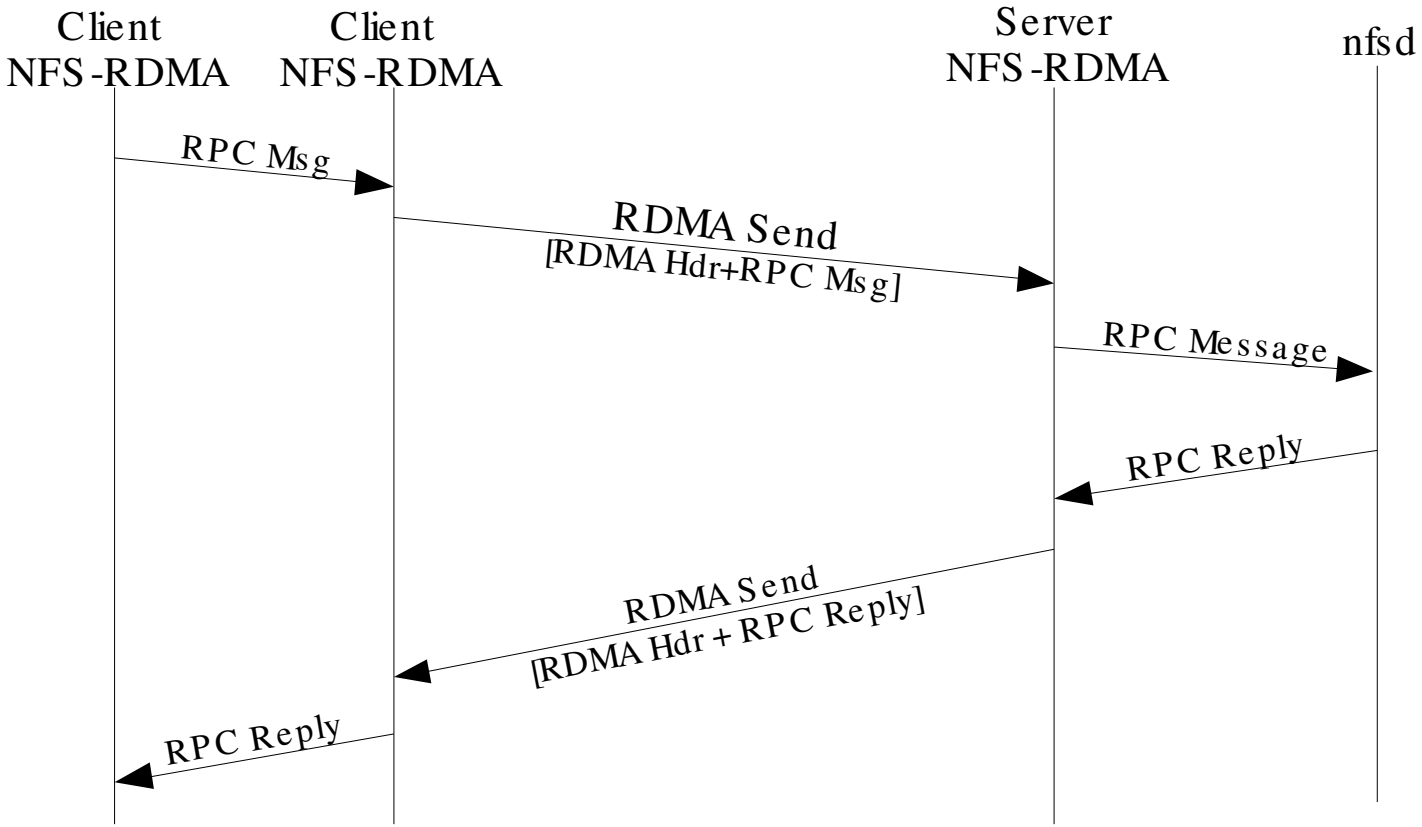
RPC-RDMA Operations (cont.)

- Inline: RDMA Send used to transport protocol headers and data when size is below a configurable threshold.
- Read Chunk: RDMA Read used to transfer data from the client to the server. Used for NFS WRITE and SYMLINK operations.

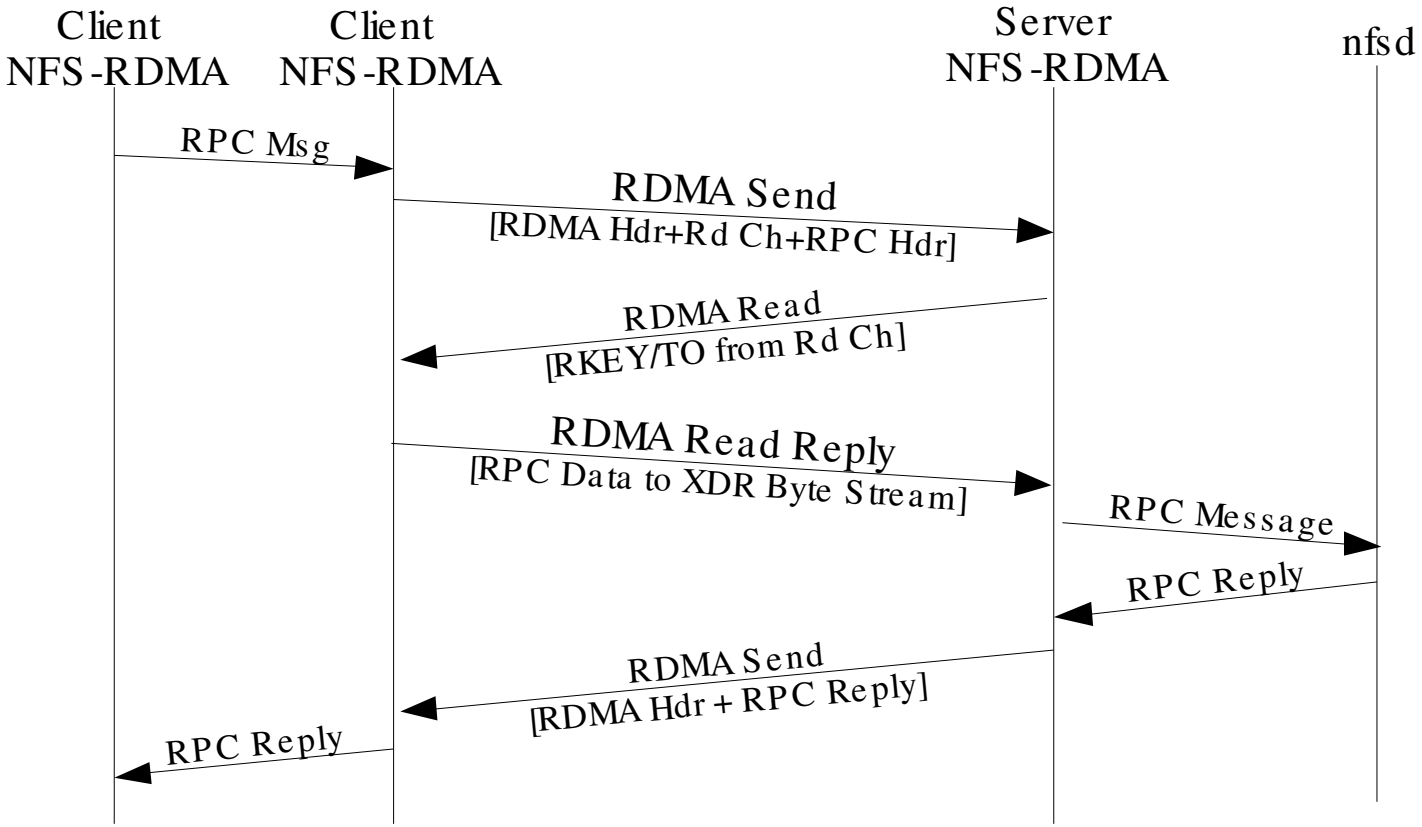
RPC-RDMA Operations (cont.)

- Write Chunk: RDMA Write used to transfer data from the server to the client. Used for NFS READ and READLINK operations.
- Reply Chunk: RDMA Write used to transfer data from the server to the client when size is above a configurable threshold.

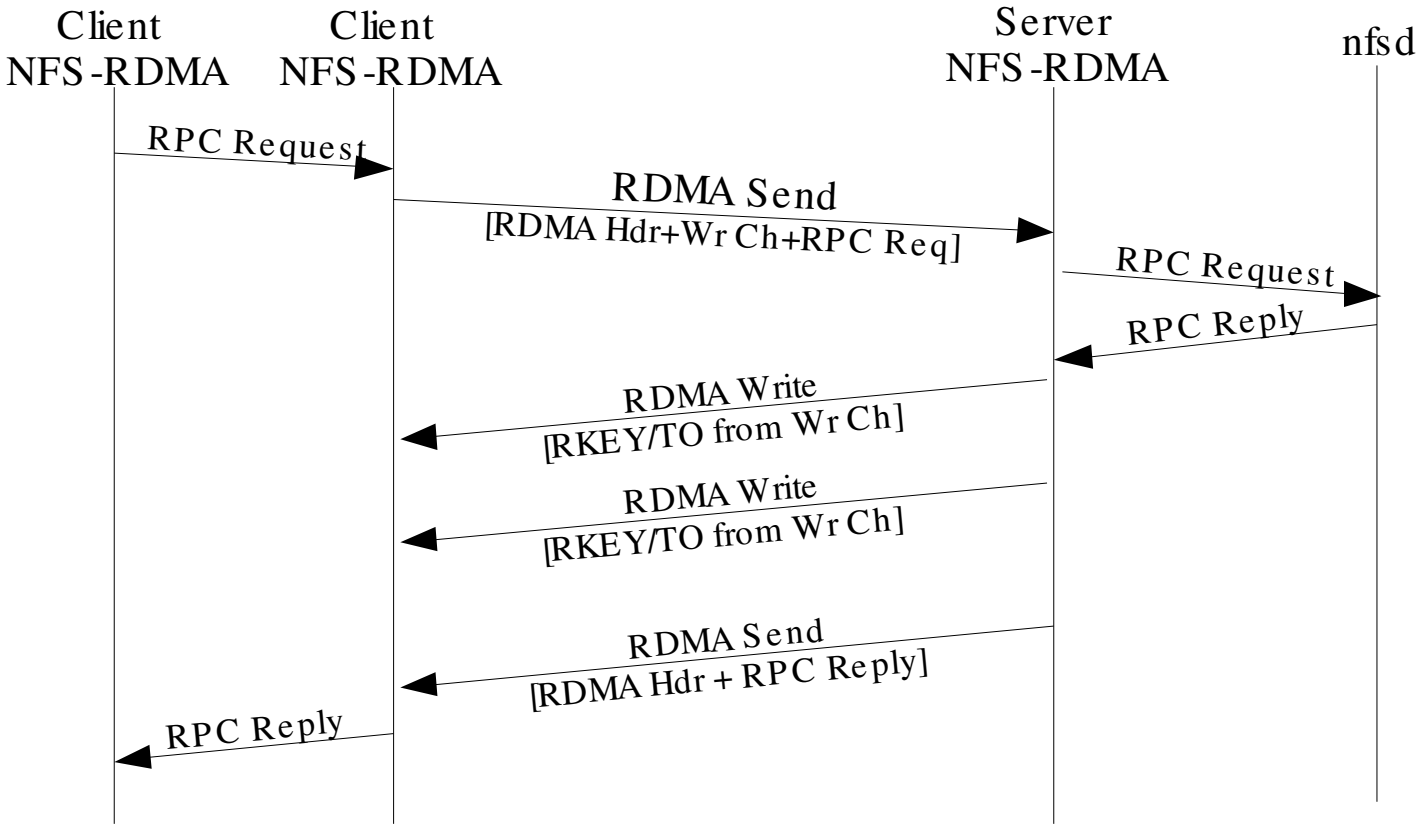
RPC-RDMA Inline



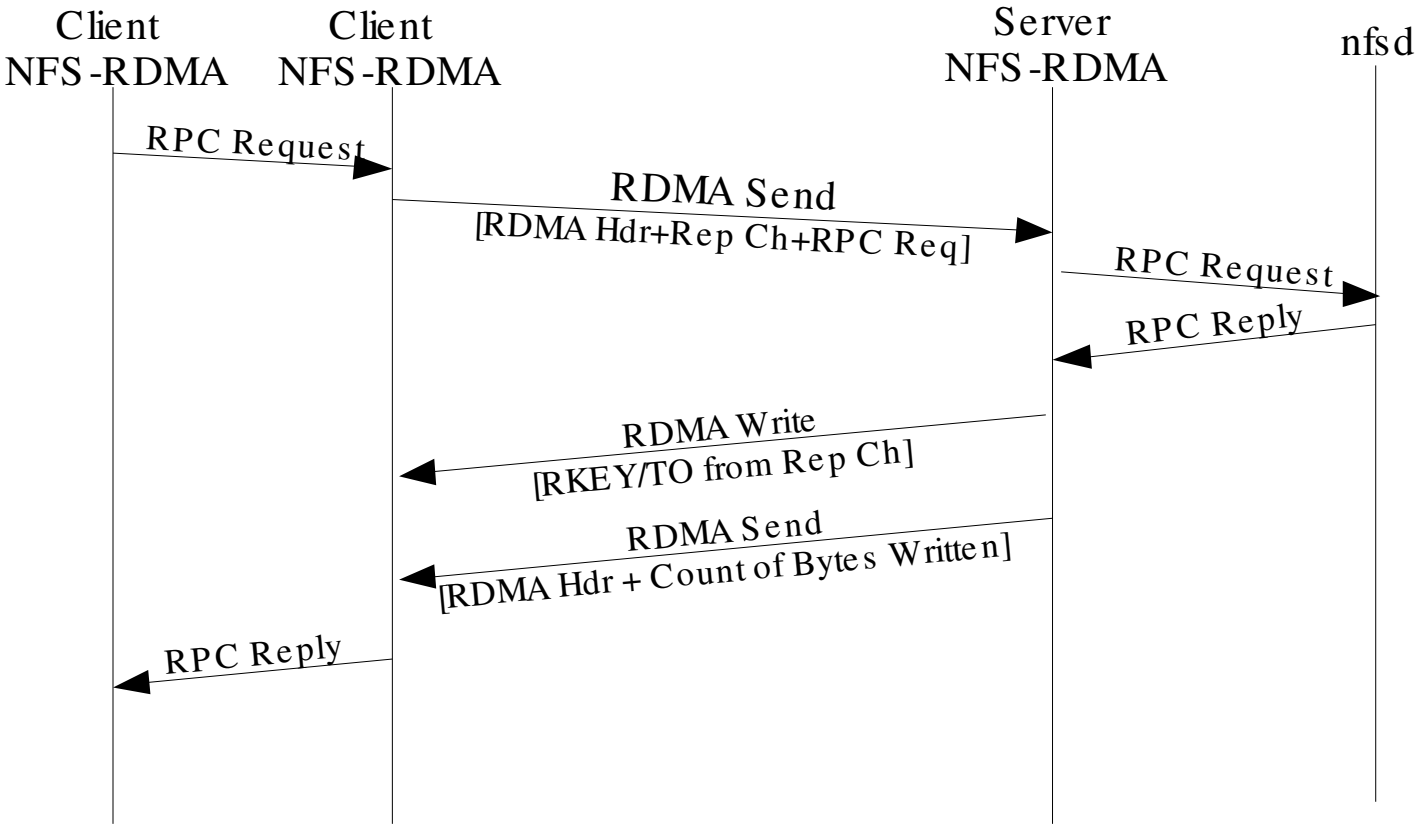
RPC-RDMA Read (NFS Write)



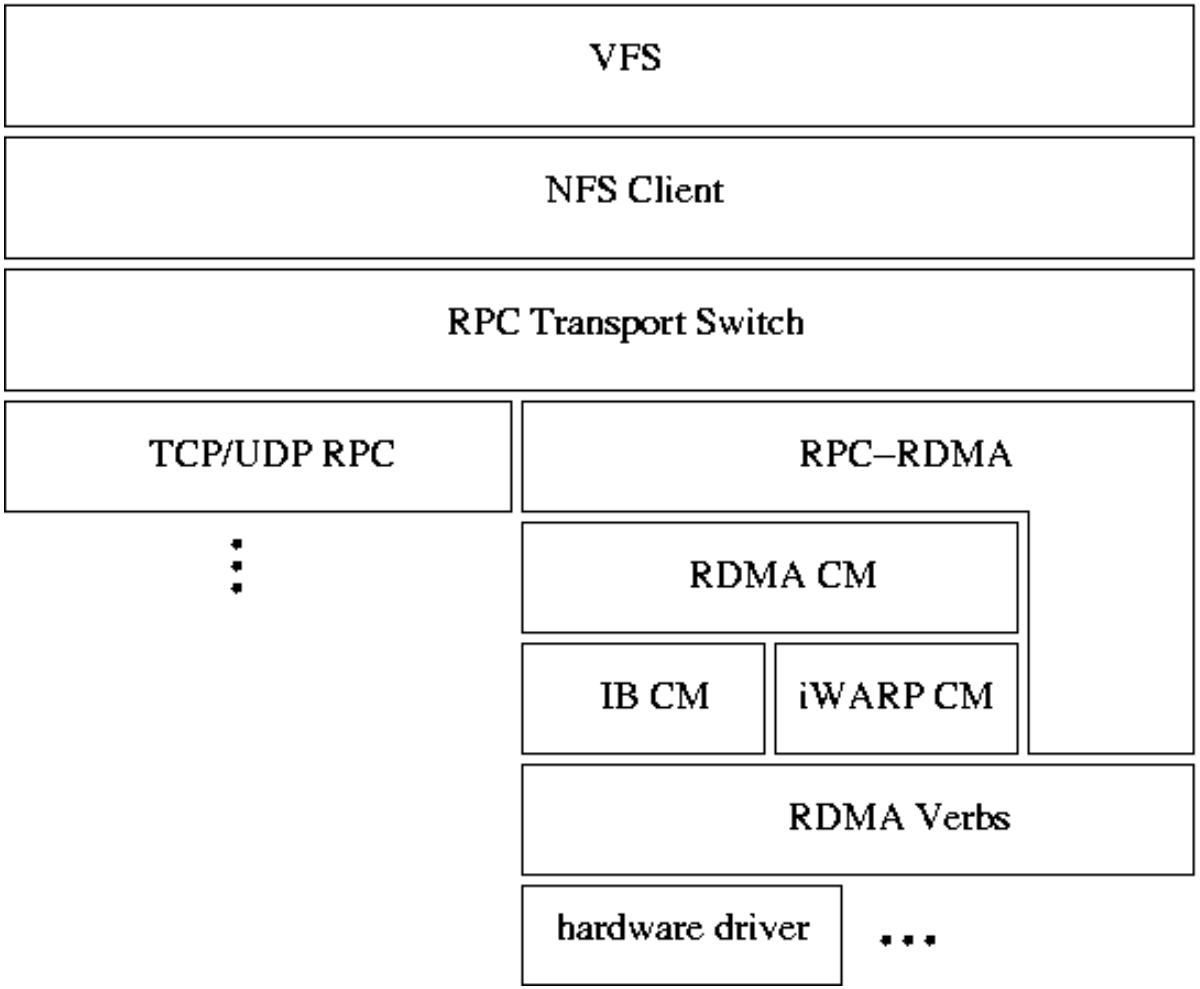
RPC-RDMA Write (NFS Read)



RPC-RDMA Reply (NFS Readdir)



Client Architecture



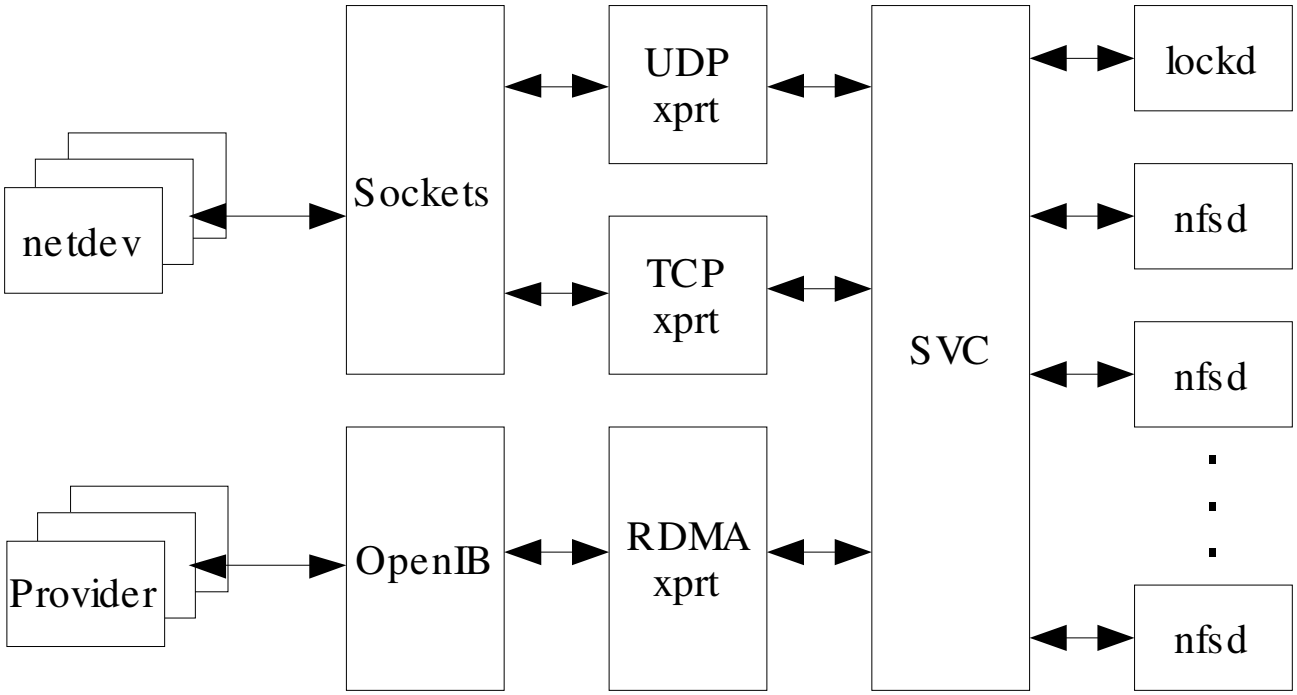
Client Data Structures

- Associated with each mount point
  - RDMA CM ID
  - QP
  - CQ
  - pool of pre-registered memory buffers (size is administratively controlled)

Client Memory Management

- Multiple memory management algorithms are possible. Five have been implemented:
  - Pure Inline (debug)
  - Memory Region
  - Synchronous Memory Window
  - Asynchronous Memory Window
  - Fast Memory Region (FMR)
  - Persistent (experimental)

Server Architecture



Server Architecture

- All RDMA operations (READ/WRITE) initiated by the server
- NFS Server is unaware of transport, processes only RPC
- Multiple concurrent RPC may be processed
- Credit based flow-control (advertised in RDMA header) used to back-pressure client
- Server registers all of physical memory to avoid registration overhead

RPC Server Processing (nfsd)

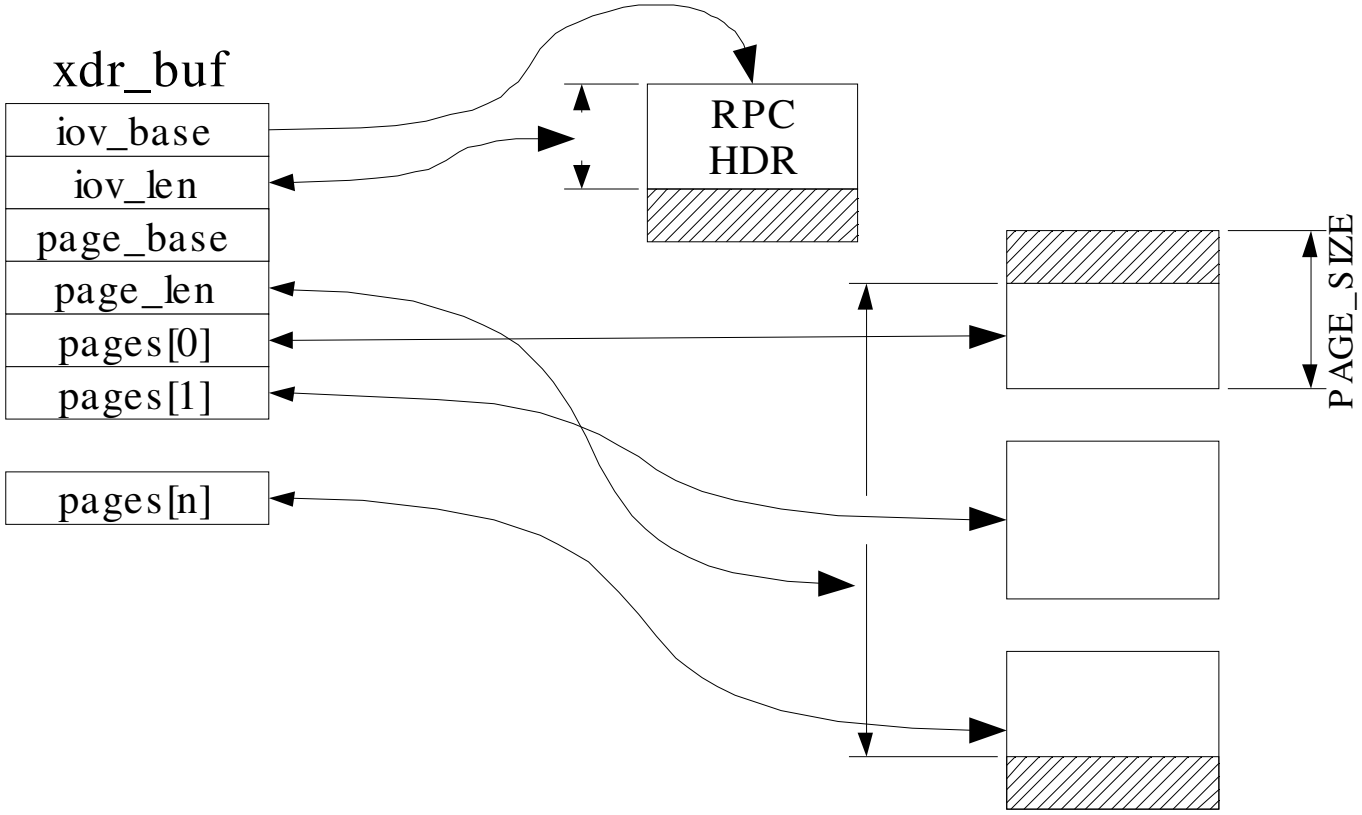
- svc_recv
  - xprt_recvfrom --> transport specific version
    - xdr_decode_rdma_header
    - if <read chunk>
      - RDMA_READ and wait
- svc_process
  - xprt_prep_reply_buf
    - insert RDMA header
    - xdr_decode args, call rpc procedure, xdr decode response
  - xprt_sendto
    - if <write chunk>
      - send data as RDMA_WRITE
      - send RDMA_SEND with RDMA header and RPC Reply
    - else if <reply chunk>
      - send header and pagelist as RDMA_WRITE
      - send RDMA_NOMSG with length

Server Implementation

- Almost all code in net/sunrpc
- Existing SVC infrastructure leveraged
  - Additional transport specific func. ptr added where necessary
    - prep_reply_buf, put_sock
  - Very few changes to nfsd/lockd proper
    - Addition of call to create RDMA transport
    - New transport type names

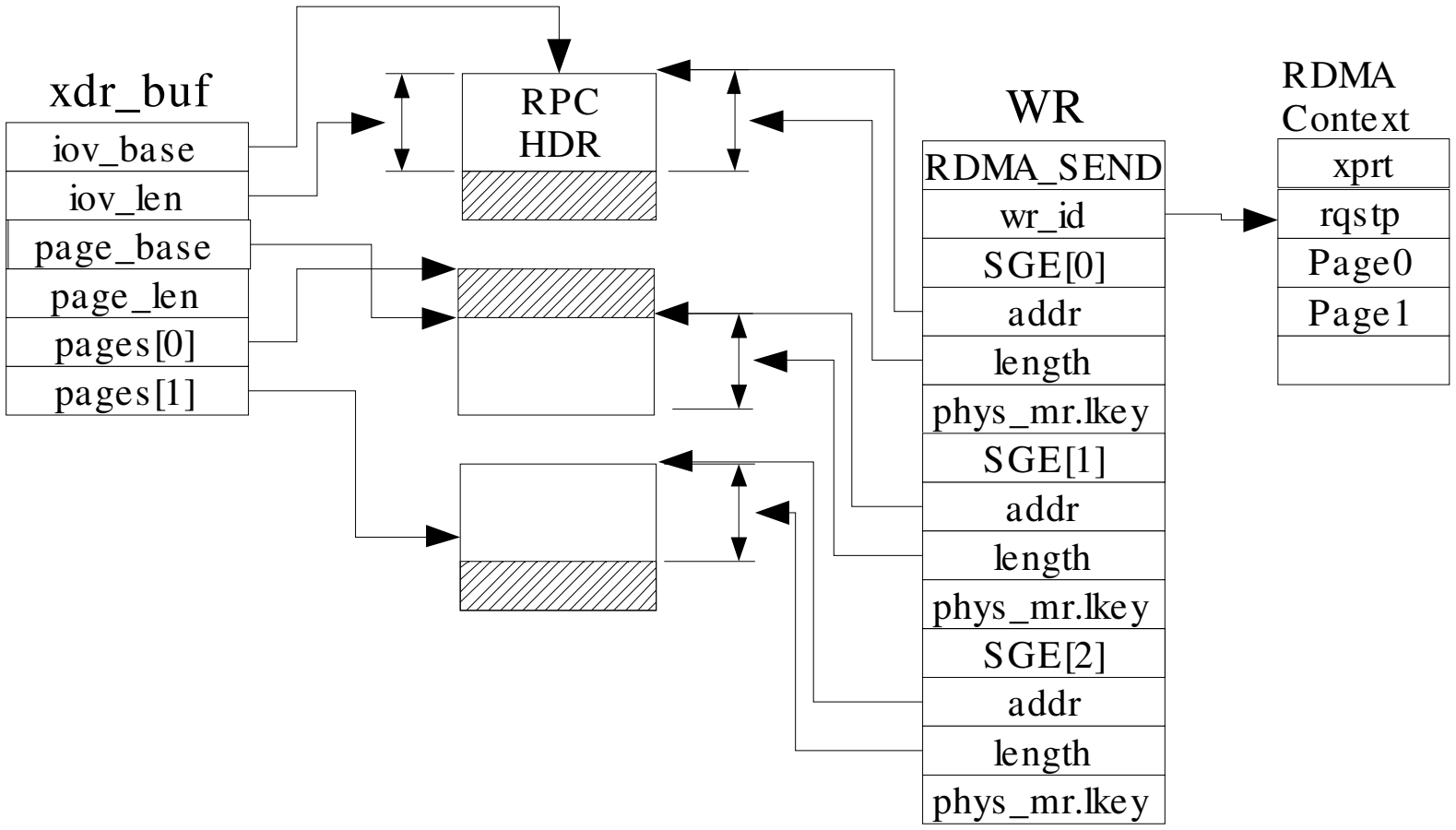
Server XDR Buffers

Buffer representation of RPC Requests and Replies

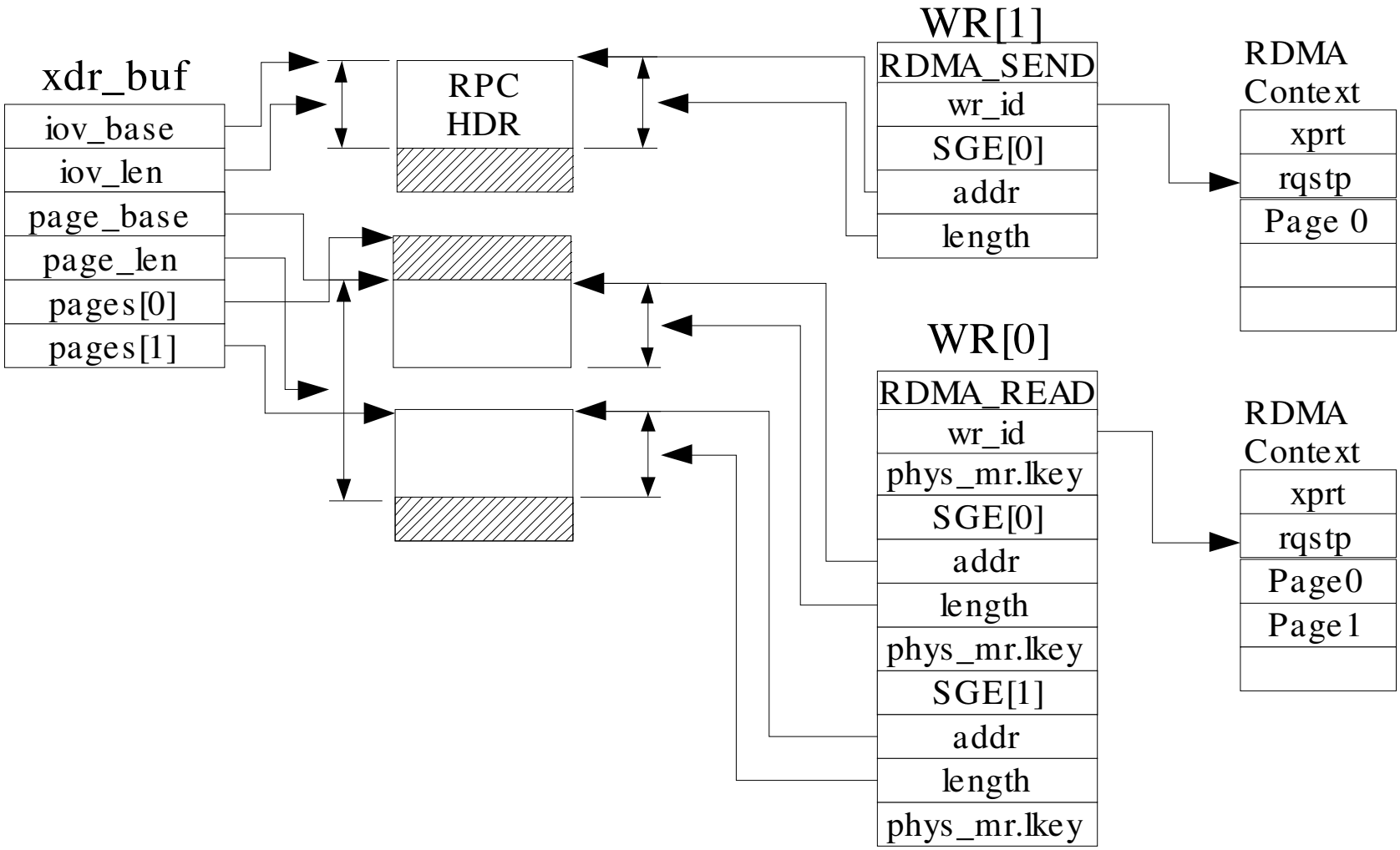


Sending an xdr_buf inline

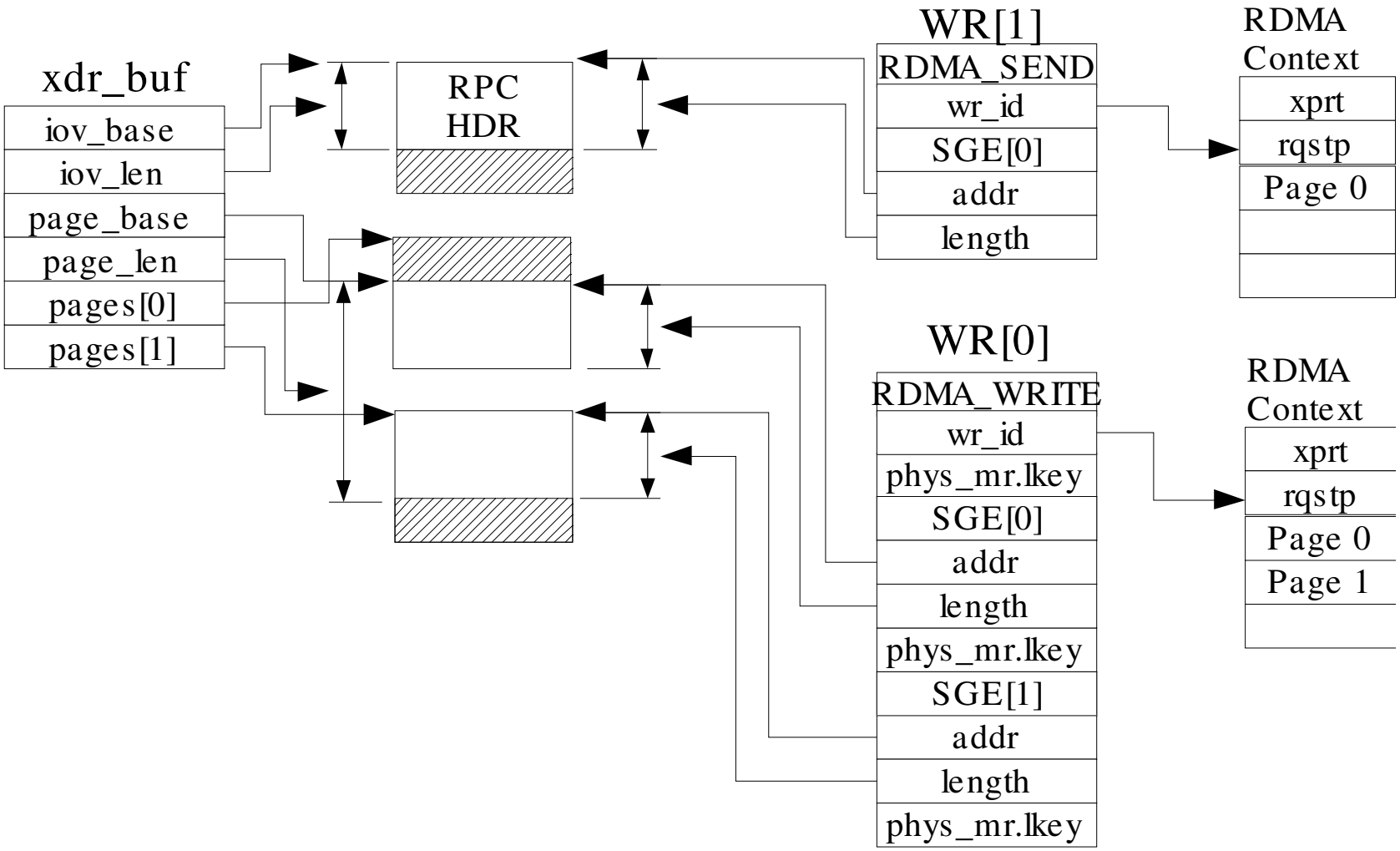
Single RDMA SEND for RPC Header and Data



Building an xdr_buf with Read Chunk



Sending an xdr_buf with Write Chunk



Status

- Client
  - All transfer modes implemented
  - Stable code (passes Connectathon tests)
- Server
  - All transfer modes implemented
  - Performance optimizations required
- Client (and soon Server) sources available at **http://sourceforge.net/projects/nfs-rdma**
- Future work:
  - Linux kernel.org submission