RPC/RDMA Last Call Issues

Tom Talpey

NFSv4 Interim WG

Ann Arbor, MI

Sept 15, 2006

Drafts

- RDMA Transport for ONC RPC
  - “RPC/RDMA”
  - http://www.ietf.org/internet-drafts/draft-ietf-nfsv4-rpcrdma-03.txt
- NFS Direct Data Placement
  - “NFS DDP”
  - http://www.ietf.org/internet-drafts/draft-ietf-nfsv4-nfsdirect-03.txt
- NFS/RDMA Problem Statement
  - http://www.ietf.org/internet-drafts/draft-ietf-nfsv4-nfs-rdma-problem-statement-04.txt

Reported Issues

- NFS DDP and Problem Statement
  - No issues
- RPC/RDMA
  - rpcbind/netid, and/or port assignment
  - Record marking vs RPC/RDMA framing
  - XDR roundup overhead

rpcbind / well-known port

- Issue – iWARP in non-“step-up” mode
  - Can’t share port 2049 between record marking (TCP) and RPC/RDMA
  - Affects NFSv2, NFSv3, NFSv4.0
- Non-issue – NFSv4.1/iWARP in “step-up”
  - Connects in TCP, uses v4.1 session to negotiate RDMA
  - “steps up” to RDMA during Session exchange
  - Reuses port 2049
- Non-issue – IB
  - IB RDMA and IPoIB use different port spaces

Proposal in draft

1. Assign a new Netid to RDMA
   - NFS server chooses port (any port)
   - rpcbind advertises service on port
   - Problem – NFSv4.0 doesn’t use rpcbind
2. Assign a new RDMA well-known port
   - E.g. 2050

- Together, would support all versions

Comment received

- Assign a new netid and always use rpcbind/portmap
- Issues
  - Guidance on choice of port
  - What about NFSv4.0?

- => We still need the port assignment
- Consensus?

Record Marking / RPC/RDMA

- No discussion of shifting from TCP record marking to RPC/RDMA framing

- Actually out of scope for the RPC/RDMA draft, and NFS DDP too
  - These only document what happens in RDMA mode
  - Only NFSv4.1 allows step-up

- => add text to NFSv4.1 RDMA discussion?

XDR Roundup (new issue)

- When bulk data is not multiple of 4 bytes, XDR “rounds up” marshalled data
- Over TCP, this simply adds up to 3 bytes to the XDR stream
- Over RDMA, this typically adds an extra RDMA chunk, for up to 3 zeroed bytes
  - Adds an RDMA Read operation to NFS writes
  - Needless overhead

Solution(s)

- Defer RDMA processing until NFS decode
  - NFS Read count <= XDR encoded?
    - Roundup
  - This is undesirable because it means changing NFS to accommodate a transport
- Encode a “roundup” indicator in RPC/RDMA read chunks
  - Easily done in client XDR marshalling
  - Easily processed in server XDR unmarshalling

Assuming you agree…

- How to encode the “roundup”?
  - Need to include a roundup flag
  - Chunk would still point at valid, rounded-up data

Rev the RPC/RDMA protocol?

- Add a boolean flag word to the rpcrdma_read_chunk
  - Define RPC/RDMA protocol version 2
  - Or just change version 1
  - 5 (known) existing implementations to change

A: Overload the XDR discrim?

- The rpcrdma_read_chunk is encoded as an array:
  - xdr_discrim (1 or 0) [4 bytes]
  - Handle [4 bytes]
  - Length [4 bytes]
  - Offset (address) [8 bytes]
  - Position (xdr offset) [4 bytes]
- Define xdr_discrim==2 as padding?

2	Handle	length[1, 2 or 3]	offset…	position

B: Revise the chunk definition

- Add field to rpcrdma_read_chunk:
  – xdr_discrim (1 or 0) [4 bytes]
  – Handle [4 bytes]
  – Length [4 bytes]
  – Offset (address) [8 bytes]
  – Position (xdr offset) [4 bytes]
  – Flags (4 bytes)
- Define padding flag
- Revise to version 2

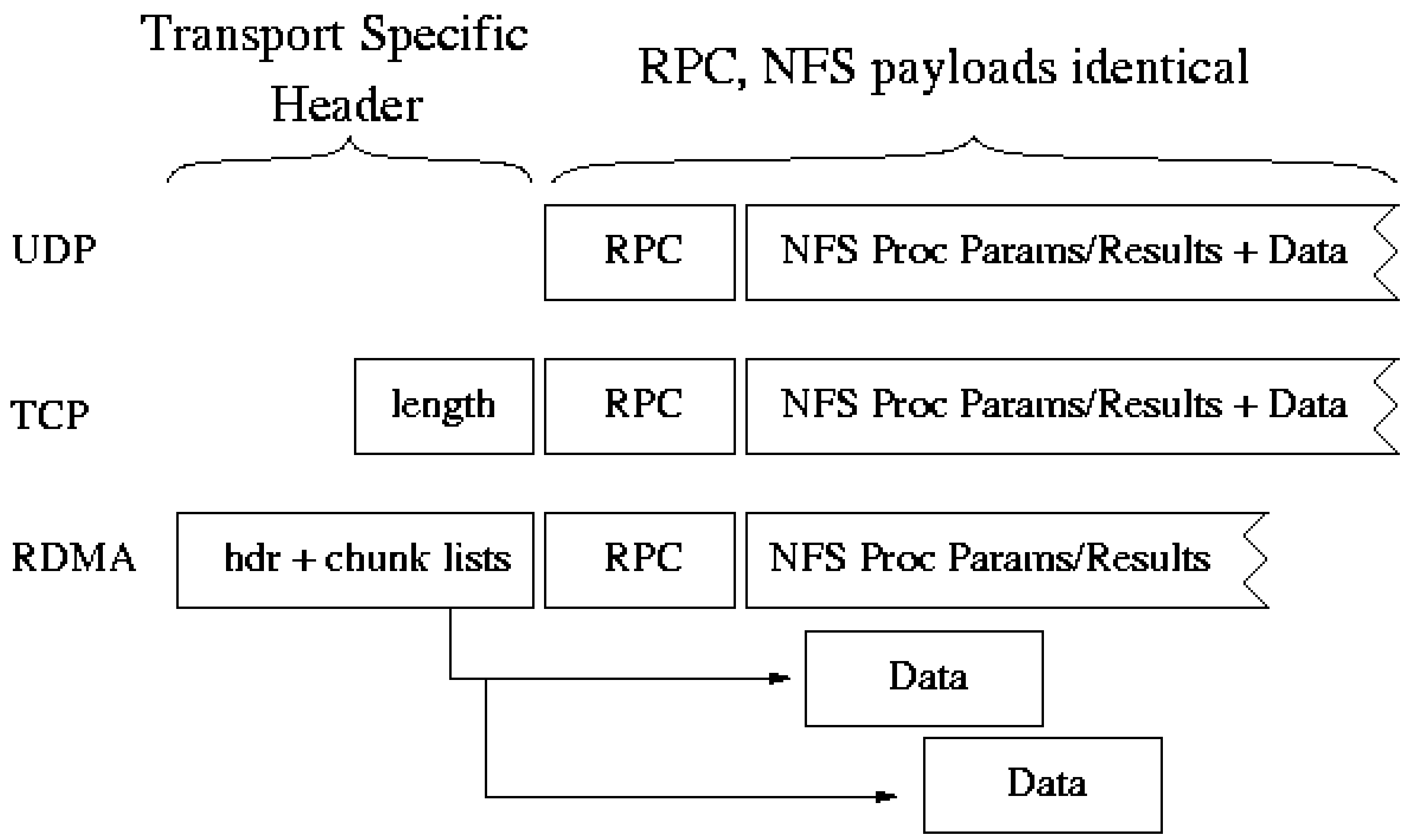
1	Handle	length[1, 2 or 3]	offset…	position	flag

I prefer “A”

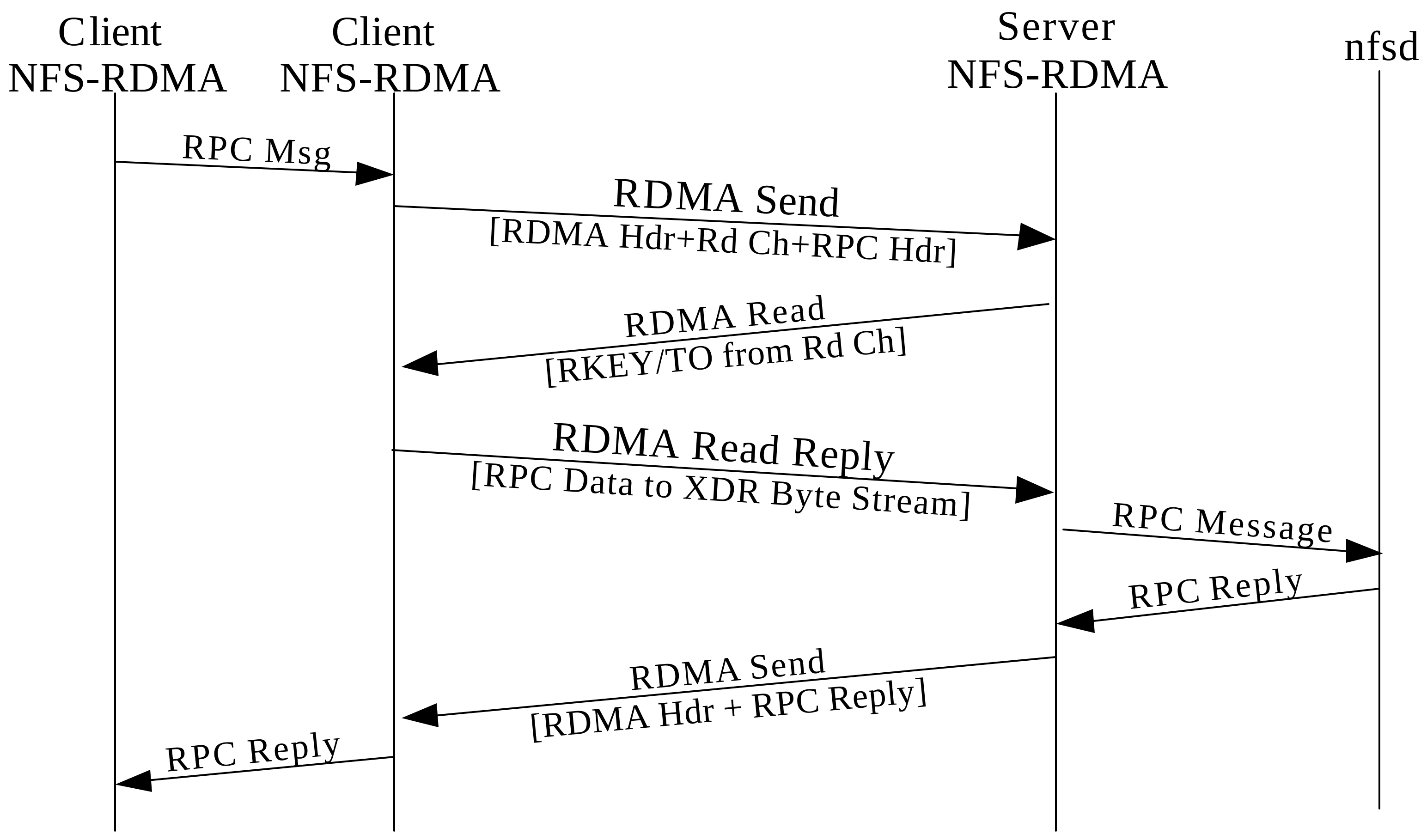
- Overload the xdr_discrim
  - This is framing, not formal XDR
  - All existing implementations compatible
  - Lightweight, transparent
  - Flexible – can declare “middle” chunks as padding (zero) too.
- But, “B” is more formal
- Opinions? (take to list after)

Backup

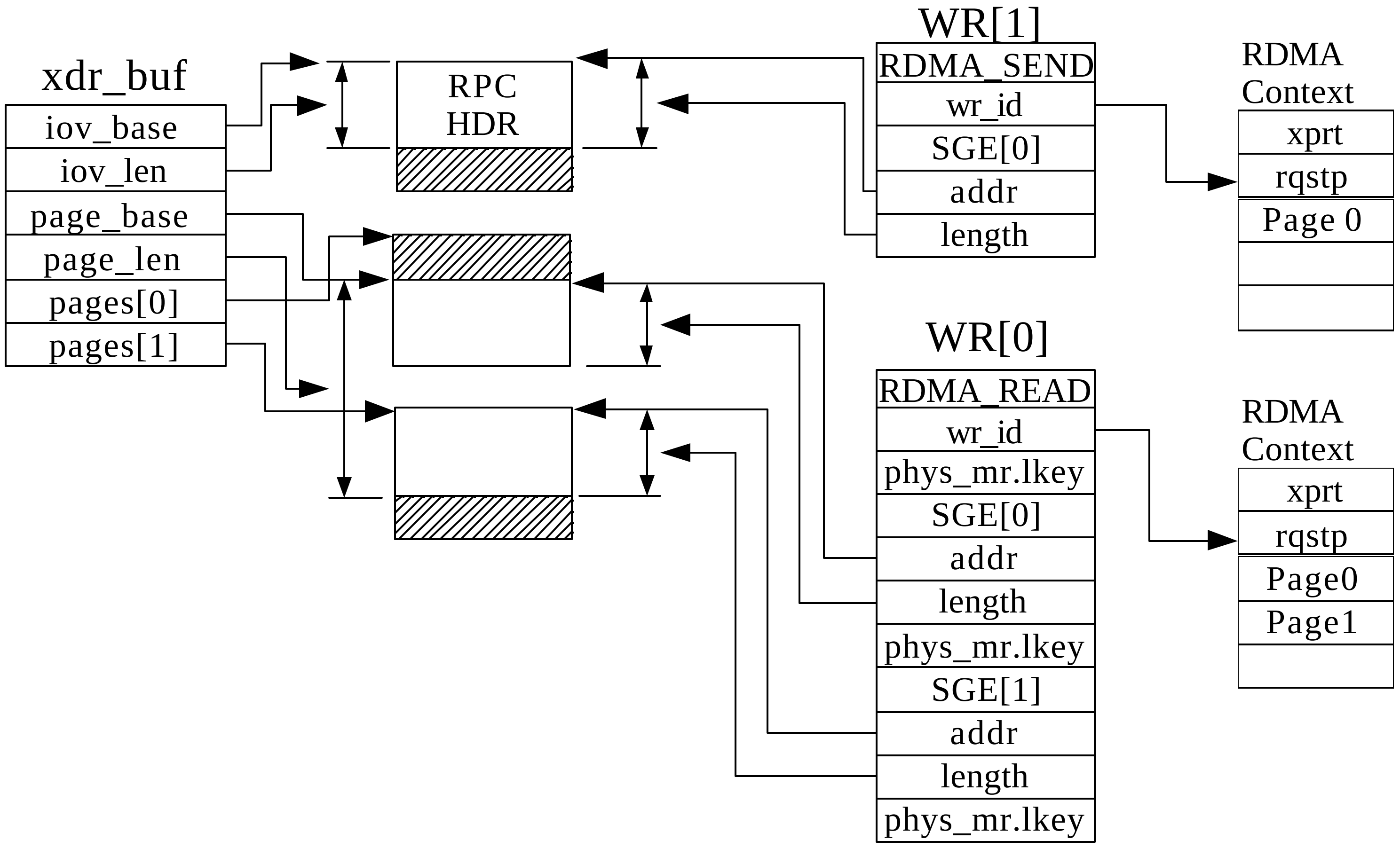
Chunking


Transport Specific Header
RPC, NFS payloads identical
UDP
RPC
NFS Proc Params/Results + Data
TCP
length
RPC
NFS Proc Params/Results + Data
RDMA
hdr + chunk lists
RPC
NFS Proc Params/Results
Data
Data

NFS Write transfer



Too much information?



xdr_buf
iov_base
iov_len
page_base
page_len
pages[0]
pages[1]
RPC
HDR
WR[1]
RDMA_SEND
wr_id
SGE[0]
addr
length
RDMA
Context
xprt
rqstp
Page 0
WR[0]
RDMA_READ
wr_id
phys_mr.lkey
SGE[0]
addr
length
phys_mr.lkey
SGE[1]
addr
length
phys_mr.lkey
RDMA
Context
xprt
rqstp
Page0
Page1