High-throughput cloud storage over faulty networks

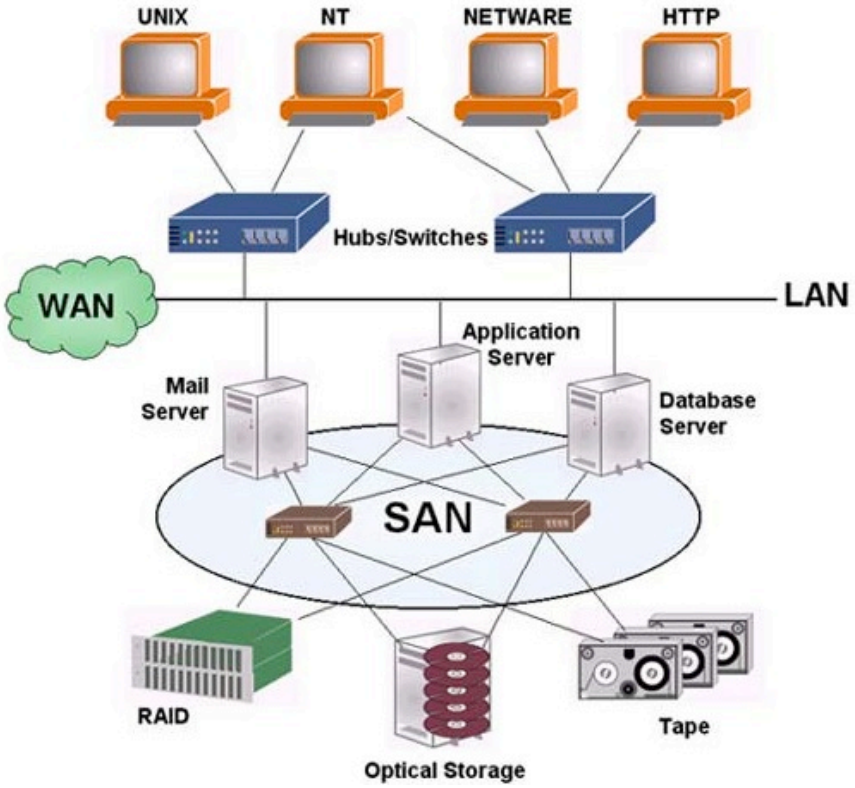
Yogesh Vedpathak
Cleversafe, Inc.

Agenda

- How cloud storage is different than SAN/NAS
- Challenges in achieving high throughput
- Achieving high throughput and resilient communication
  - Message based data transfer
  - Multipath communication
- Conclusion
- Questions

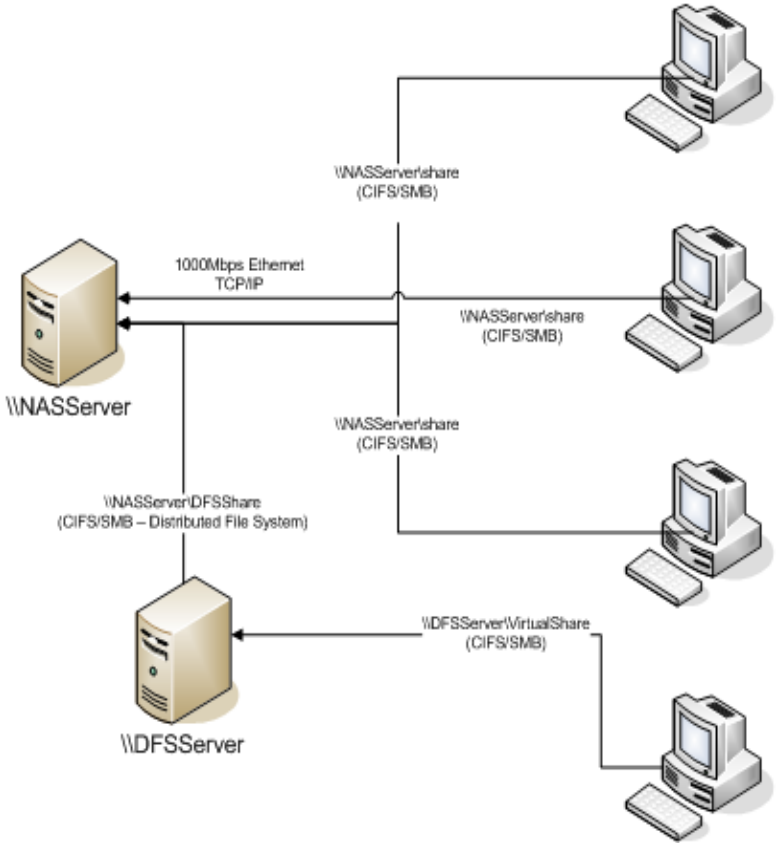
Storage Area Network

- Dedicated storage within LAN
- Support for block and iSCSI
- Doesn't rely on TCP

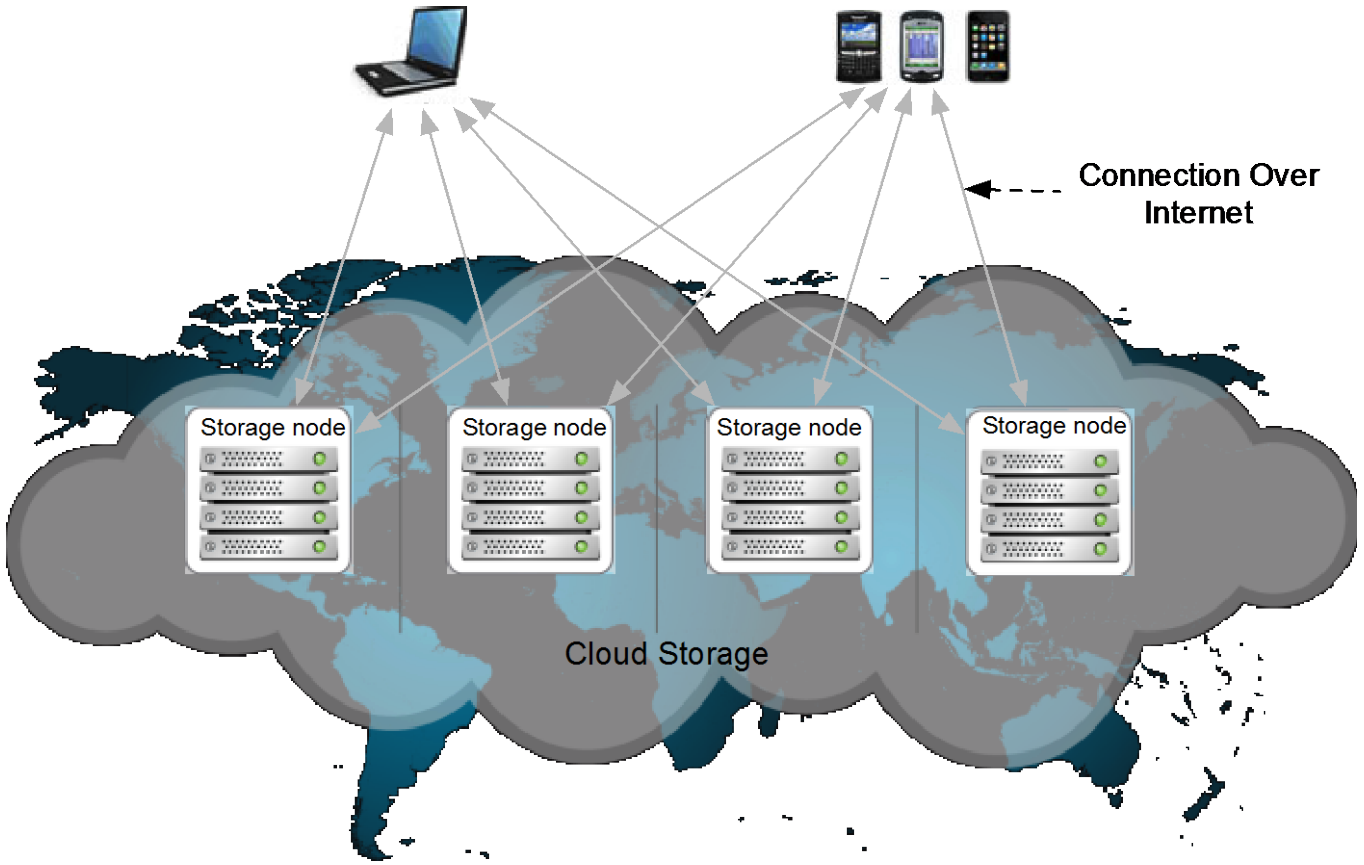


Network Attached Storage

- File level storage support for heterogeneous clients
- Supports NFS, SMB/CIFS
- XDR serialization over TCP/UDP



Cloud Storage Architecture



Cloud Storage

- Stateless object based storage
- Multiple geographically distributed readers/writers
- Multiple geographically distributed storage nodes
- Heavily reply on IP transport layers (TCP/UDP)
- Internet is a primary means of transportation

Bandwidth Delay Product

- Product of data link capacity & end-to-end delay
- The value (in bytes) means
  - Amount of data in-flight at any given time
  - Or amount of data you need to saturate the link
- High-speed terrestrial network: 1 Gbit/s, 1 ms RTT
  - B x D = 1Gbit/s x 1ms = 125KiB

Challenges 1 of 3

- Data transfer channel has great impact on performance and reliability
  - Transfer 1GiB on network with BDP 125KiB
  - Connection goes down after transferring 75%
- Retry? Resend the 750MiB
- Resume? Resend only 125KiB
- Today's transport layers do not support "resume"

Challenges 2 of 3

- High-bandwidth, high-latency connections are common in wide-area corporate networks and multi-datacenter storage deployments
  - 45ms regional round trip within North America
  - 90ms transatlantic round trip
  - With 1 Gbit/s NIC, B x D would be 5MiB
- How to keep network pipe saturated to achieve high throughput?

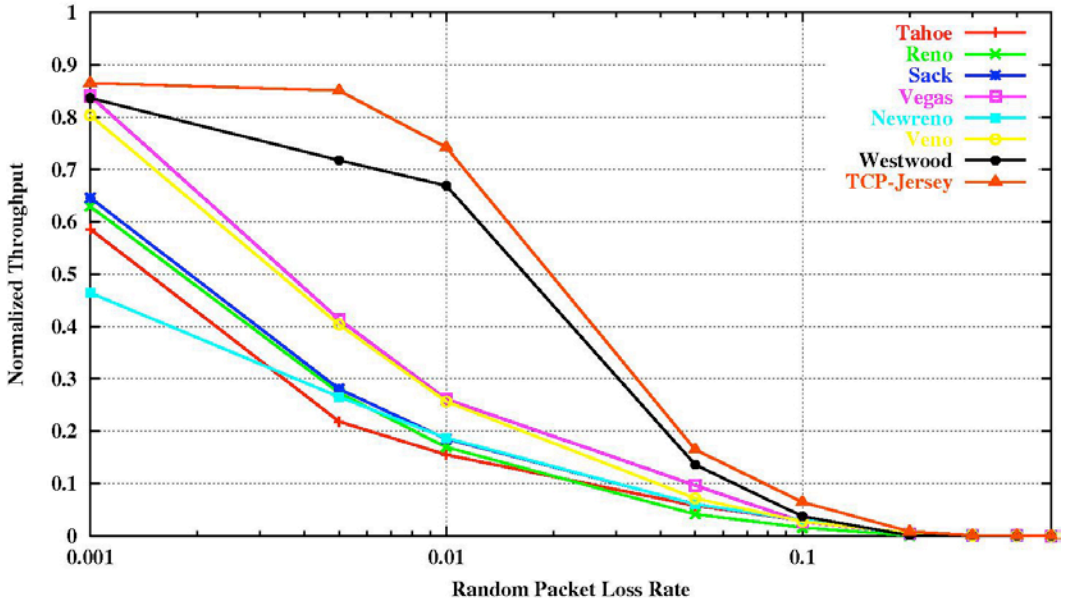
Challenges 3 of 3

- Packet loss introduces transmission delays on Internet connections
- How to design application to dynamically adapt with such delays?

Choosing Transport Protocol

- UDP
  - Does not provide reliable data transfer
- SCTP
  - Message based (yet streaming) transfers
  - Not widely deployed
- TCP
  - Reliable transmission and flow control

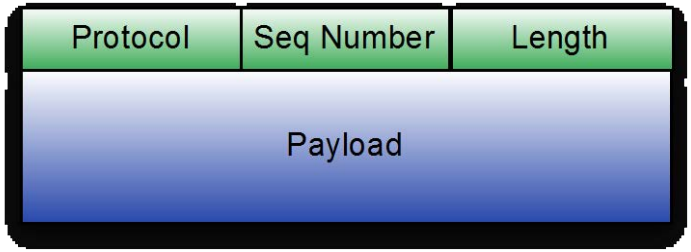
TCP Limitations



- Slow start strategy for congestion control limits in-flight data to the size of congestion window
- Single TCP connection can not be used to transfer more than one “stream” at a time

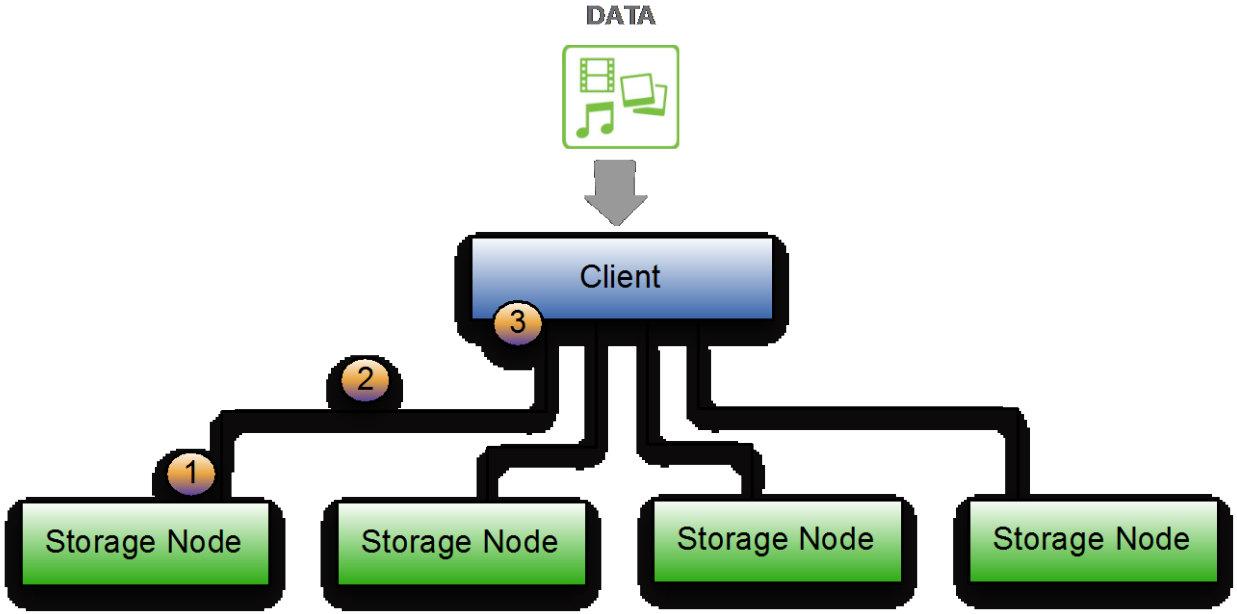
Message Based Transfer

- Dividing streams into discrete messages
- Each message has a header and payload data



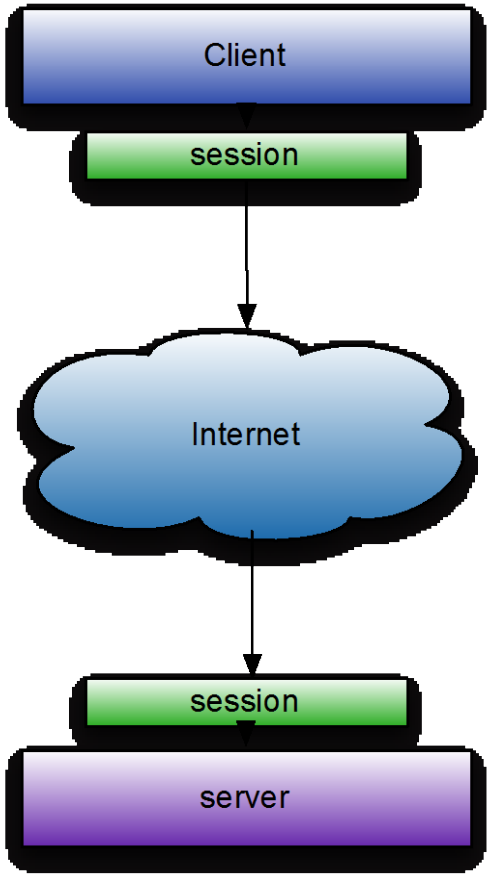
- For each request exists corresponding response
- Examples
  - Write request, Read request

Data Flow



- Total data 12MB; 3MB per storage node
- Payload per write request is 1MB
- Client sends 3 write requests atomically under single transaction

Maintaining A Session



- A stateful, logical association of connections
- New connection “binds” to session
- Keeps track of ongoing transactions
- Should be valid on both sides at all times
- Any open transactions are aborted when session closes

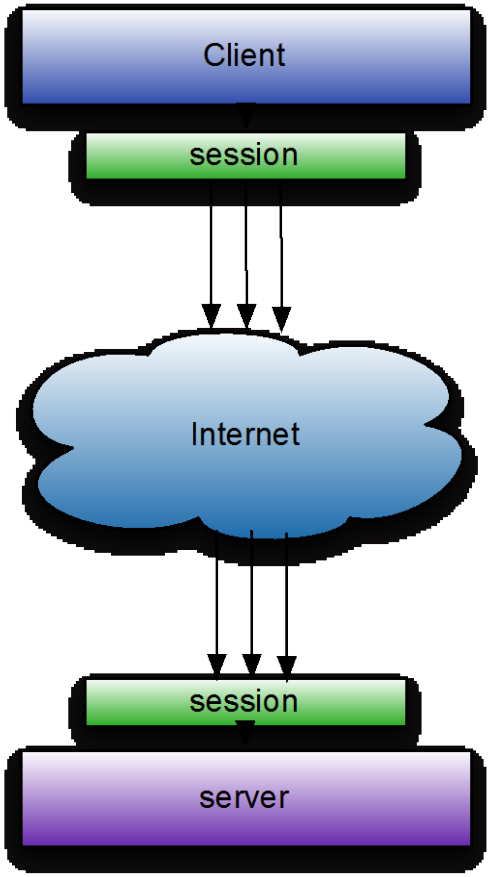
Request Prioritization 1 of 2

- Suppose we are writing 12GiB of data
- Need to send 3GiB per storage node
- Application is constantly streaming data on the network
- On 1Gib NIC it will take around 24 seconds (best case)
- What if a new request to read or look up data comes in?

Request Prioritization 2 of 2

- Per session
  - 3000 outstanding messages
  - Prioritized based on protocol type
  - Prioritized based on sequence number
- After writing data to threshold number of nodes
  - Messages can be deprioritized
  - Messages can be cancelled
- Key is to delay actual network write

Multipath Connection



- Start with a single connection
- Based on latency value add or remove connections
- Choosing connection to send message
  - Round robin
  - Based on minimum response time

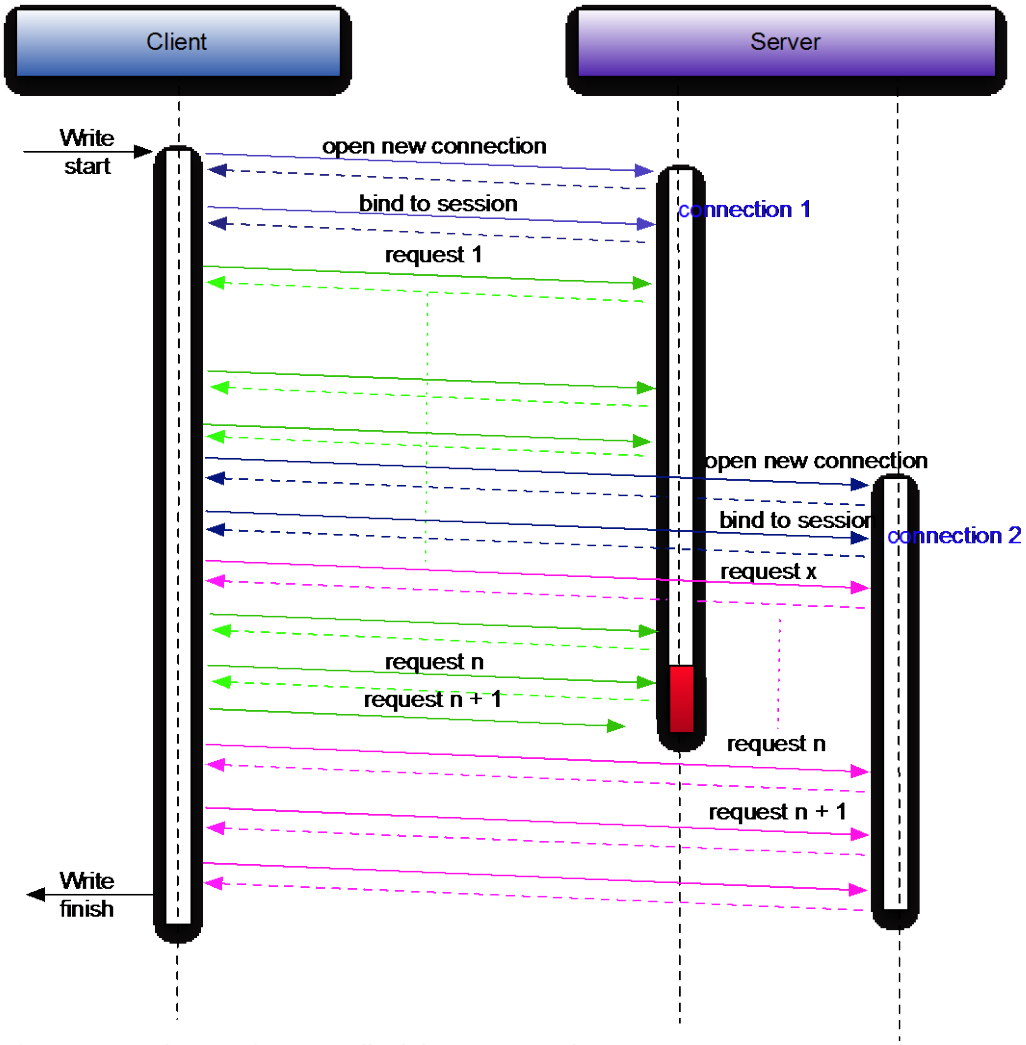
Multipath Connection

- More connections allow more data to be in flight
  - 1MB receive window size per socket
  - With 8 connections 8MB can be outstanding
  - In order to loose all 8MB all connections must be lost
- Single connection can be affected by packet loss and become slow
  - Client can resend a request on different connection if it doesn't receive response in certain time

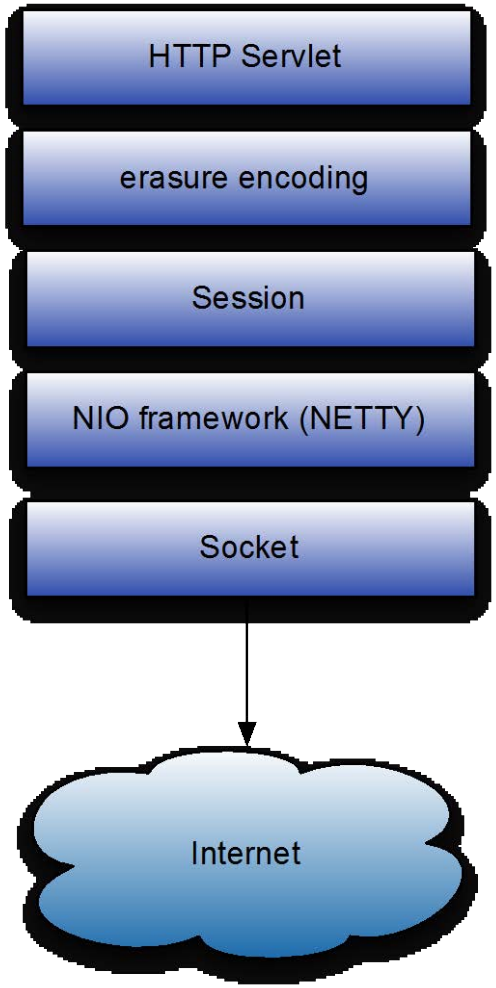
Message Playback

- The client can send same message multiple times
- Server however “executes” the message only once. Sends response to each message
- Client can ignore response received multiple times

Request Delegation

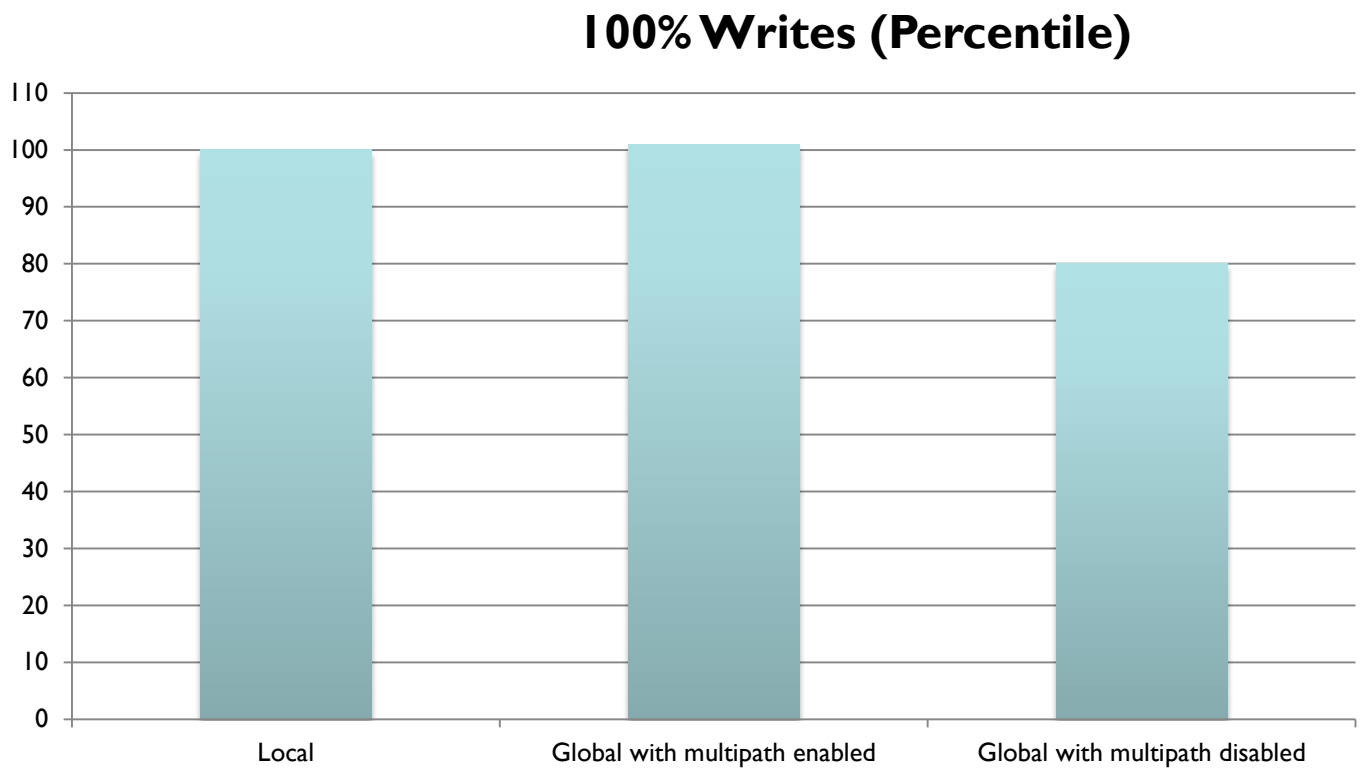


NIO Client Server Model



- One thread per connection
  - Context switching is expensive
- One core per connection
  - Limited by number of cores
- Async thread and NIO is effective when there are thousands of outstanding messages

Throughput Comparison



Site 1: 30ms
Site 2: 45ms
Site 3: 00ms

1Gib NIC
50 threads
~5 MiB objects

Conclusion

- Message based transfer
  - Allows prioritization and re-ordering of data
  - Unlike streaming protocol easy to resend
- Multipath connections
  - Provides high throughput communication
  - Allows fault tolerant communication

Questions

- Any questions?