# Database over NFS

## beepy’s personal perspective

Brian Pawlowski

Vice President and Chief Architect

Network Appliance

beepy@netapp.com

# What is a database?

# A Database™

- A structured data storage scheme that allows
  - Fast access to records within the set of data based on a variety of queries
    - Transactional commits ensuring updates are atomic and unique
  - Ability to generate reports based on relationships and filters of the data
  - Ability to recover consistent points within a stream of transactions following an error

# Simple database musings

- Assume a database is 250GB or bigger
  - Just to get in the ballpark
- Databases are deployed typically to store business critical data
  - Even a small instance for a small business will be holding customer billing information
- Databases are assumed to be highly available and highly reliable
  - Consider any business interaction in your life - a bank, credit card, airline ticket, mutual fund, telephone bill, Fandango - you expect business transactions to work perfectly

| Storage Consolidation | Data Center Operations | Business Continuance | Distributed Enterprise |
|---|---|---|---|
| Infrastructure | Backup/Recovery | Data Availability | Data Access |
| Scalability | Capacity Allocation | Disaster Recovery | Application Acceleration |
| High Reliability | Central Management | Time to Recovery | Streaming Media |
| Investment Protection | Performance Management | Economical Protection | Security |

# Implications for storage

# First thoughts

- Availability
  - Even the simplest database deployment assumes redundant storage - mirroring or parity RAID
  - Simple deployments assume recovery in face of data loss (if only from backup tape)
- Availability too
  - Clustering of components including fans, power supplies, processors and datapaths

# Second thoughts

- Database workloads are distinctive
  - Primary concern is random I/O workload due to keyed transactions
    - Though data is laid contiguously on disk, assume it was written randomly and later read randomly
  - There are sequential components
    - The redo logs and transaction logs for recovery
    - Data mining operations for analysis execute sequential scans
- But first order concerns are with the random I/O component

# Third thoughts

- Networks are as fast as direct attach storage connections
  - Started 100:1, now 1:1 (or better)
  - Commodity, high performance switches
- Flexible architecture for storage deployment
  - Improves resource utilization
- Regardless of access method data management is required in storage device

# Fourth thoughts

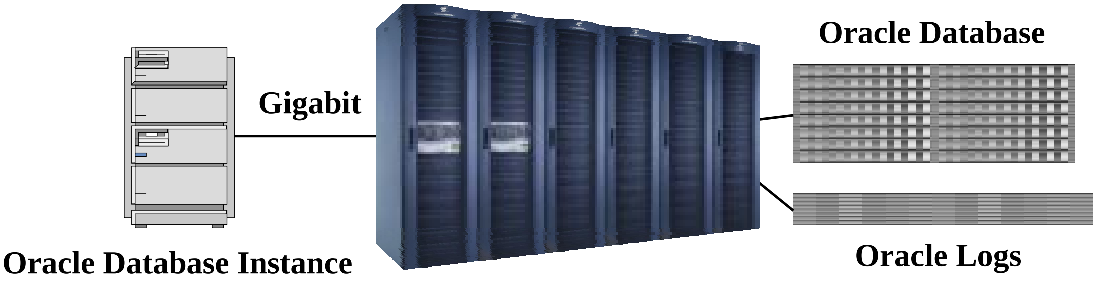
- Networking storage yields other benefits
  - Offload data mining and backups
  - Enable disaster recovery solutions
- One thing I’ll say for it, networking’s cool

NFS INDUSTRY CONFERENCE

# e.g. Database recovery

## Database Recovery Scenario - An Example

- **300 GB database and the entire database requires recovery**
  - **Tape recovery time is 60 GB/hour**
  - **Normal recovery time is 5 hours + log replay time**
- **SnapRestore reverts volume to same state as when backup was taken.  Duration - 3 minutes**
  - **Total recovery time:  3 minutes + log replay time**



# Common themes

- Networking simplifies backup and disaster recovery
  - Offload data mining and backups
  - Enable disaster recovery solutions
- The storage system is more than a JBOD
  - Snapshots or other fast backup method
  - No single point of failure
- Provisioning new and reconfiguring old storage must be transparent
  - Common maintenance tasks must be non-disruptive

# Why NFS?

# First, why IP networking

- No significant difference between networking technologies (FC vs. Ethernet)
  - Similar latency or bandwidth
- Ethernet and TCP/IP costs lower
  - People
  - Tools
  - Infrastructure

# Enter iSCSI

- iSCSI provides storage access capabilities similar to FC SAN
  - But uses standard Ethernet and TCP/IP
  - Cost reduction over FC deployment
- Both techniques simply encapsulate SCSI commands over a network
  - But don’t provide anything much more than a direct connect SCSI cable

# Wait, we already have…

- NFS provides high speed data access on a network
- Sophisticated data management architectures can be built
- Leverages existing infrastructure

# Files vs. blocks

- Things a block storage device knows:
  - All blocks may have data
  - The geometry of underlying RAID devices
- Things a file storage device knows:
  - User friendly naming and organization of data
  - The set of blocks that have allocated data, comprise a file, and comprise a file system
  - The set of users that have permission to access each block
  - Which application has locks on each block
  - The geometry of underlying RAID devices

# NFS (NetApp) filers today

- Provide sophisticated data management
  - Snapshots and IP mirroring
  - Full redundancy
  - Fast recovery
  - Online expansion (simple to grow storage pools), offload volume management
- High performance scalable solution
- Cost effective near-line storage solutions for disaster recovery and archiving

# Why not files and NFS?

- Overhead of network processing on host
  - But NFS (or iSCSI) is suitable for most database deployments
- Certification of applications
  - Block storage solutions are de facto standard for database
- Industry investment
  - Though simpler and more cost effective, blocks is he easy answer compare to NFS

# Alternatives to NFS?

- Local file system to direct attach storage
  - Provides name space and management
  - No networking limits flexibility (e.g. disaster recovery)
- Clustered file system
  - Typically SAN-based
  - Complicated deployment
  - Requires more support on database server

# NetApp and NFS

- NetApp has exposure to all OS / NFS Clients
  - AIX, Solaris, HP/UX, Linux, others
  - Lots of customer installations
- Applications of Interest
  - Databases (Oracle, DB2, others) - 40% of storage deployment
  - Large Apps (Lotus, SAP, Rational, SAS, others) - many database related
  - Home Directories / Smaller Apps
- Trends of Interest
  - NFS/NAS becoming infrastructure of choice for “DataCenters”

# Best practices

# Best practices

- Issues in database deployment on NFS can be decomposed to
  - Storage
  - Network
  - NFS server
  - NFS client
  - Database
- Always install recommended software releases and patches by vendor
  - For client and server and database
  - Investigate vendor tuning guides

# Storage

# Best practices

- Your storage solution should support
  - Fast backup and recovery (e.g. Snapshot and Snaprestore) that integrates with your database
  - Disaster recovery solution must exist (e.g. Snapmirror to a NetApp NearStore R150)
- Capacity on demand, on-line expansion, autosupport
- Premium customer support from vendor
- You get the picture

# Best practices

- Redundant components in storage to increase availability
  - Redundant  power circuits should be used
  - Redundant networks should go through redundant switches
- But recognize that redundancy is one aspect of availability
  - Does your solution minimize necessary scheduled downtime?
  - Network Appliance advocates simple to manage storage solutions

# Best practices

- The number of spindles serving data limits performance
  - But not all database deployments require large numbers of spindle
  - Know your database performance needs - choose a flexible storage solution to allow responding to changing needs
- Understand peak load requirements
  - Including performance in degraded mode (broken disk or controller)
  - When possible schedule common online maintenance (backups) for non-peak load time

# Best practices

- Optimize “volume” size to simplify backup and disaster recovery
  - Multiple volumes vs. a large single volume is pretty irrelevant from the database perspective
  - Consider disaster recovery solutions that also enable offload of backup
- Balance database deployment against other application deployment when using shared resources
  - Maintenance of unrelated applications can affect database availability

# Network

# Best practices

- Use adequate networking
  - Can choose 100BaseT or Gigabit based on performance requirements
  - Configure networks for autonegotiation and ensure full duplex operation
- Redundant networking for high availability
  - Including paths through different switches
- Separate database network from backup and other traffic network
  - Switched networking helps here
- “Jumbo frames” can help

# NFS server

# Best practices

- Tune sufficient NFS server contexts/messages for highly concurrent database client
  - Some servers are tunable
- NFS Version 3 is recommended
- TCP is recommended
  - Keep abreast of latest patches

# NFS server

- Investigate vendor tuning suggestions for NFS
  - Networking buffer space
  - TCP window size

The world is a file
nfs://industry.conf

NFS INDUSTRY CONFERENCE

# NFS client

# What do databases want?

- A name space and expandable storage
- Basic system services to fetch data
- Prefer OS to stay away otherwise
  - Databases like raw storage - because they prefer to manage their own buffer space

# NFS client directio

- directio means “no host caching”
  - Enabled in two ways
    - flag on the open command (O_DIRECT)
    - with a mount option
- Why direct I/O is important
  - Databases manage their own buffer pools
  - Host caching is ineffective and adds unneeded costs
  - Common high-performance database deployment
    - Mount database with directio
    - Give maximum memory to database buffer pool

# Best practices

- Support for large transfer sizes (32KB) and many outstanding requests
  - Some parts of database - data mining - need large sequential thruput performance
  - A well-behaved client will right size random I/O requests
- Problems seen historically in OS locking serialization and operation starvation
  - Writers blocking readers in a shared random access file is bad
  - Check with vendor for patches

# Best practices

- Mounts should be “hard” or “intr”
  - I really didn’t have to say that, did I?
- On some clients using multiple mount points or additional files can increase performance
  - But these seem to be artifacts and bugs waiting to be fixed!

# Database

# Database tuning

- Match database block size to storage device “block size” multiple
  - 4KB is minimum recommended for Network Appliance storage
  - May be tunable on tablespace basis
- Asynchronous I/O functions in database are platform dependent
- For example, DBWR threads in Oracle
  - Use Oracle tools to determine whether to increase
  - Consider tuning recommendations for multi-CPU hosts

# Best practices

- Assuming you can get OS buffering out of the way (directio when available)
  - Tune database buffer space to reduce I/O
  - Add memory in client - this can help some database deployments
- And more wonderful database vendor specific tuning

# A bright future

# Need for systematic study

- Database over NFS is very compelling
- Impediments to deployment are:
  - Client specific correctness and performance issues
  - Lack of tuning guidelines for customers
  - Lack of “customer support” for NFS deployments
  - Published benchmark data to demonstrate utility
- Easily corrected with cooperative NFS vendor investment

# NFS evolution

- DAFS (derived from NFS) shows that even client performance issues are solvable
  - NFS/RDMA effort in IETF attempting to standardize
- Enhanced protocol semantics for database use are possible

# Questions?

# Thanks go to

- Darrell Suggs - NetApp
- Giovanni Brignolo - NetApp
- Glenn Colaco - Sun
- Bruce Clarke - NetApp