ZFS
The Next Word...

Jeff Bonwick
Bill Moore
www.opensolaris.org/os/community/zfs
What's in the pipeline?

- Performance
- User quotas
- Pool recovery
- Triple-parity RAID-Z
- De-dup
- Encryption
- BP rewrite (huh?) & device removal
- Shadow migration
- Random cool features
Performance

- Hybrid storage pools
- New block allocator
- Raw scrub
- Parallel device open
- Zero-copy I/O
- Scrub prefetch
- Native iSCSI
- Sync mode
- Just-in-time decompression
ZFS Hybrid Storage Pools

• Separate log devices for fast synchronous writes
  > Enterprise-grade SLC flash SSD
    – Cheaper than NVRAM
    – A few GB is plenty
    – Easily clustered over standard SAS fabric

• Cache devices for fast random reads
  > Cheap, consumer-grade MLC flash
    – L2ARC: an eviction cache for the L1ARC (DRAM)
    – As much as necessary to hold working set
    – It's just a cache – failures are OK, no need to cluster
    – Everything is checksummed – no risk of silent errors

• Low-power, high-capacity disks for primary storage
ZFS Hybrid Pool Example

- 4 Xeon 7350 Processors (16 cores)
- 32GB FB DDR2 ECC DRAM
- OpenSolaris with ZFS

Configuration A:
(7) 146GB 10,000 RPM SAS Drives

Configuration B:
(1) 32G SSD Log Device
(1) 80G SSD Cache Device
(5) 400GB 4200 RPM SATA Drives
ZFS Hybrid Pool Performance

- If NVRAM were used, hybrid wins on cost, too
- For large configs (50T - 1PB+) cost is entirely amortized
User Quotas

• For enterprise customers
  > Finer grained answer to “where did my space go”

• For education customers
  > Many users, want quota per user
  > One fs / user is too many (unfortunately)

• User & group quotas with “deferred enforcement”
  > User may go over quota for several seconds
    (one transaction group) before system notices
    that they are over quota and returns EDQUOT

• Supports both SMB SIDs and POSIX UIDs/GIDs
User Quota Interface

- New properties
  - `userused@<user>`
  - `groupused@<group>`
  - `userquota@<user>`
  - `groupquota@<group>`
  - "zfs get" / "zfs set" like other properties

- `<user>` or `<group>` specified as:
  - Numeric POSIX ID (125829)
  - POSIX name (ahrens)
  - Numeric SID (S-1-123-456-789)
  - SID name (matthew.ahrens@sun)

- New subcommands: "zfs userspace" and "zfs groupspace"

- Display table, one line per user or group, e.g.:

<table>
<thead>
<tr>
<th>TYPE</th>
<th>NAME</th>
<th>USED</th>
<th>QUOTA</th>
</tr>
</thead>
<tbody>
<tr>
<td>POSIX User</td>
<td>ahrens</td>
<td>14M</td>
<td>1G</td>
</tr>
<tr>
<td>POSIX User</td>
<td>lling</td>
<td>258M</td>
<td>none</td>
</tr>
<tr>
<td>POSIX Group</td>
<td>staff</td>
<td>3.75G</td>
<td>32T</td>
</tr>
<tr>
<td>SMB User</td>
<td>marks@sun</td>
<td>103M</td>
<td>5G</td>
</tr>
</tbody>
</table>
Pool Recovery: The Problem

- ZFS pool integrity depends on explicit write ordering
  - Some cheap disks and USB bridges silently ignore it!
- Result: uberblock written before data it points to
  - Power loss can lead to complete pool failure

Synchronize Cache
Pool Recovery: The Solution

- Recover pool even if devices ignore write barriers
  - Check integrity of recent transaction groups at pool open
  - If damaged, rollback to earlier uberblock
  - Rollback made reliable by deferred block reallocation

- Status
  - Working code; finalizing user experience
Triple-Parity RAID-Z (RAIDZ3)

- Survives three-disk failure
  > Or, more likely: two-disk failure plus occasional bad reads
- Enables bigger, faster, high-BER disks
  > 30-40% of the bits on modern hard disks are ECC
  > With different zone recording tables, we could have:
    - 30-40% higher capacity
    - 30-40% higher bandwidth
    - Much more frequent errors, detected and corrected by ZFS

- Status
  > Integrated into OpenSolaris
  > Write-side “mind the gap” performance improvement
RAID-Z Write Throughput Results

2008.Q4

2009.Q3

0
200
400
600
800
1000
1200

mirror
rz1
rz2
rz2-wide
rz3-wide

Preliminary throughput data
Note throughput was also much more consistent
Dedup

• Only store one copy of identical data blocks
  > Three parts: on-disk, in-core, over-the-wire

• Key applications
  > Virtualization
  > Backup servers
  > Build environments

• Behaves like you'd expect
  > Transactional
  > Plays well with other ZFS features
  > No special hardware
  > Unlimited scale
ZFS Encryption

READ

- verify checksum
- decrypt
- decompress

ZFS

ZPL

DMU

SPA

ARC

ZIO

VDEV

VDEV

WRITE

- compress
- encrypt
- checksum
ZFS Encryption: Design Goals

• Encrypt all data and ZPL metadata (name, owner, etc)
  > All data on zvols can be encrypted
• Allow for secure delete
• Must not require special hardware
  > But should be able to take advantage of it
• Don't break Copy-On-Write semantics
• Integrate with existing ZFS admin model
• Support mix of ciphertext and cleartext datasets
ZFS Encryption: Key Management

- Dataset encryption requires two different types of keys
  - A user specified key called the “wrapping” key
  - A randomly generated dataset key wrapped by the user specified key
- This model simplifies such tasks as secure deletion
  - Get rid of the wrapping key and the data is deleted
- The wrapped key can also change without changing the user specified wrapping key
Dataset encryption can only be enabled at create time

- Specify keysource and encryption algorithm
- Enables SHA-256 checksum automatically

Keysource indicates location of the wrapping key

Two encryption algorithms supported initially

- AES-128-CCM
- AES-256-CCM (default when enabled)
BP Rewrite

- Move blocks and update all pointers atomically
  - Foundational ZFS technology
  - Enables device removal, on-line defrag, recompress, etc.
- Rocket science
  - Subtle, racy, hard to debug – and has to be perfect
- Status
  - Now: code works on quiescent pool
  - Soon: work with concurrent read/write activity
  - Finally: configuration changes, out-of-space issues
Shadow Migration

- Migrate data from third-party NAS
  - Minimal downtime, usable immediately
  - VFS/vnode interposer: new share faults in data from old

- Status
  - Done: basic functionality, background migration, analytics
  - To do: hard links, progress monitoring, error management
  - Initial version NFS-only
Random Cool Features

• Dynamic LUN expansion
• Snapshot holds
• Access-based enumeration
• Multi-mount protection
• Slog offline (and figured out most of slog removal)
ZFS
The Next Word...

Jeff Bonwick
Bill Moore
www.opensolaris.org/os/community/zfs