I/O in a Flash

Evolution of ONTAP to Low-Latency SSDs

NetApp, Inc

February 28, 2024

© 2023 NetApp, Inc. All rights reserved.
Agenda

- Background
- Chronology
- Results
- Discussion
ONTAP, WAFL since the 90s

- Multi-protocol server with multi-tenancy
  - NFS, CIFS, iSCSI, FCP
- Snapshots, HA, DR
- Dedupe, compression, encryption, clones
- Designed to maximize HDD bandwidth
ONTAP, WAFL since the 90s

- Multi-protocol server with multi-tenancy
  - NFS, CIFS, iSCSI, FCP
- Snapshots, HA, DR
- Dedupe, compression, encryption, clones
- Designed to maximize HDD bandwidth

\[ SSD = ?? \]
ONTAP’s strong foundation

- The gamut of essential enterprise features
- Journaling
  - To convert random writes to large sequential writes at consistency points
- Storage efficiency techniques
  - Dedupe, compression, clones
Projects across the system

- Journaling
  - Allocation in multiples of erase block size
- Storage efficiency techniques
  - Inline versions of dedupe and compression
  - More efficiency via sub-block compaction
- Scheduling, write-path efficiencies and more..
Optimizing the read path
ONTAP Stack

- Protocol (e.g., NFS, SMB, SCSI)
- WAFL
- RAID
- Storage Drivers
- Storage Media
- Ethernet, Fibre channel
- storage connectivity
- NVRAM
Legacy Read Path

1. WAFIL read
2. RAID read
3a. Storage read
3b. Protocol reply
4. SSD Driver + Device
5. RAID read-done
6. WAFIL read-done
7. WAFIL read
8. Protocol reply

NetApp
Read Latency Breakdown
Read-Done
Fast-Path
WAFL Buffer Model

To Waffinity and Beyond
Think Global, Act Local
WAFL Buffer Model

**WAFL Buffer Cache**

- **Buffer Priority Queue**
- **WAFL Buffer**
  - FileID = 0x18
  - Level = 0
  - FBN = 11
  - PVBN = 71
  - Data
- **ioBuffer**
  - FileID = 0x18
  - Level = 0
  - FBN = 12
  - PVBN = 71
  - Data
- **4KB Page**

**Page Header Hash**

- **Page Header**
  - RefCnt = 2
  - Valid = True
  - In-flight = False
  - PVBN = 71
- **Hash**
  - Aggr ID, PVBN 71
- **Page Header**
  - RefCnt = 1
  - Valid = True
  - In-flight = False
  - PVBN = 72

To Waffinity and Beyond
Think Global, Act Local
Read-Done Fastpath
WAFL Reply
Fastpath
WAFL Reply
Fastpath
RAID Fast-Path
RAID Fast-Path

Diagram:
- REQ to Protocol
- WAFR read from Protocol to WAFL
- Storage read from WAFL to RAID
- SSD Driver + Device from RAID to Storage
- RESP from Storage to Protocol
- Protocol reply from Protocol to RESP
Fast-Path Analysis

![Graph showing latency comparison for different file systems and settings. The graph compares Baseline, WAF Reply Fastpath, WAF Reply + RAID Fastpaths, and Baseline with WAF Reply Fastpath and RAID Fastpaths. The graph indicates latency in milliseconds for different operations such as Protocol, Storage + RAID SW, Read Message, SSD Driver + Device, and Read-Done Message. The performance metrics are 640MB/s and 880MB/s.]
Latency v Throughput

- No Fastpath
- WAFL Reply Fastpath
- WAFL Reply + RAID Fastpaths

Throughput (MB/sec)

Average Latency (ms)
Technique

The layer bypass optimizations can be applied a high percentage of the time, and come with safety checks.

For exceptional conditions, the legacy code remains:
- adding, removing disks
- checksum errors

WAFL restart makes this simple.
Topspin Read Path

1. Storage read
2. Protocol
3. WAFL
4. RAID
5. Storage

Protocol

REQ

RESP

RESP
Topspin

Storage Location Cache

SLC Entry

ParentVer = 21
Dirty-bitmask = 0x0000
L1 PVBN Map...

WAFL Buffer Cache

Buffer Priority Queue

WAFL Buffer

FileID = 0x18
Level = 1
FBN = 0
PVBN = 16
Data

iobuffer

FileID = 0x18
Level = 0
FBN = 13
PVBN = 17
Data

4KB Page

Page Header Hash

Hash

PVBN 17

Page Header

RefCnt = 1
PVBN = 16
Touches = 0

Page Header

RefCnt = 1
PVBN = 17
Touches = 1

Hierarchical Attributes Cache

Volume HAC

FSID

0x1234

Volume Entry

SelfVer = 31
State = ONLINE...

Inode HAC

FH

Inode Entry

ParentVer = 31
SelfVer = 21
Size = 32MB...
TopSpin Read Path

- SLC provides file offset to block number mappings
- HAC stores state to enable gating checks for Read requests
- Page Headers can be accessed under a lock from any thread (look up block numbers in memory)
- `iobuffers` can be used as vehicles for I/O by code not running in a WAFL context
Topspin Analysis

Read Performance -- High-end system

Average Latency (ms)

Throughput (MB/sec)

- RR Baseline
- RR TopSpin
- SR Baseline
- SR TopSpin
Overcoming parallelism limitations

![Single Volume Random Read Graph]

- **Y-axis**: Average Latency (ms)
- **X-axis**: Throughput (MB/sec)
- **Legend**:
  - Baseline
  - TopSpin
Correctness

All reads see the effects of all writes that have been acknowledged*

Reads and writes are atomic and isolated

* the server has dispatched the acknowledgement
Correctness

Reads see the effects of all writes that have been acknowledged

**Fast-Paths**
- WAFL buffer state is always tested before I/O
- Fast-paths are only for reply

**TopSpin**
- SLC bitmaps are updated before any write is acknowledged
Correctness

Reads and Writes are atomic and isolated

Fast-Paths

WAFL execution model guarantees serial executions and atomicity
- Suspend/restart
- Waffinity

Topspin

All applicable SLC entries are locked
- by writes for bitmap updates
- by reads for gathering PVBNs
Lesson 1

Layer bypass for the common-case path is safe and effective as a way to reduce software overheads.
Lesson 2

Incremental optimization of ONTAP was the right approach

“Legacy” is not a bad word
Thank you!