Backpressure in FreeBSD I/O Stack

M. Warner Losh

Netflix, Inc.

BSDCan 2017

http://people.freebsd.org/~imp/talks/bsdcan2017/bsdcan2017.pdf

Outline

Netflix Background

- Netflix Network
- Netflix Issues

FreeBSD I/O Stack

- Overview
- struct buf
- Interfaces

Back Pressure

2017

NETFLIX

Netflix

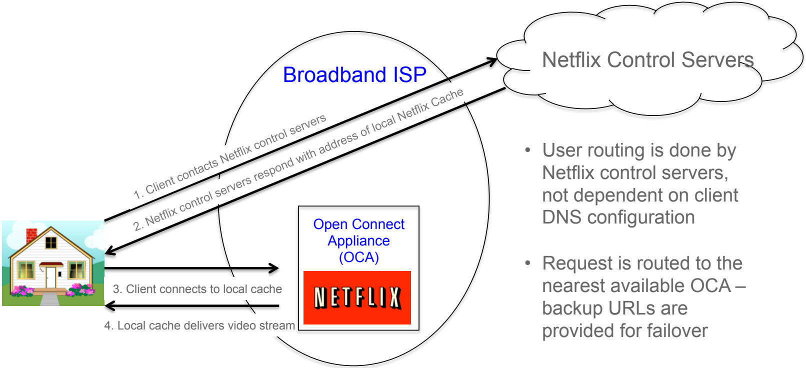
- Internet Video
- Content Distribution Network (CDN)
- Operating at Scale
- Anticipating the Future

Netflix Open Connect

- According to Sandvine, Netflix streams ˜1/3 of Internet Traffic
- Netflix has own CDN (OpenConnect)
- Streams mutliple terabits per second

http://blog.streamingmedia.com/wp-content/uploads/2014/02/2013CDNSummit-Keynote-Netflix.pdf

Netflix Open Connect Appliance (OCA)


Broadband ISP
Netflix Control Servers
1. Client contacts Netflix control servers
2. Netflix control servers respond with address of local Netflix Cache
Open Connect Appliance (OCA)
NETFLIX
3. Client connects to local cache
4. Local cache delivers video stream

- User routing is done by Netflix control servers, not dependent on client DNS configuration
- Request is routed to the nearest available OCA – backup URLs are provided for failover
- ISP controls client to OCA mapping/clustering/ failover via BGP

Source: Netflix

Netflix OCA Types

- Netflix Storage Appliance (HDD with small SSD offload)
- Netflix Flash Appliance (SSD or NVMe based)
- Netflix Global Appliance (HDD and medium SSD offload)
- Netflix possible future appliances:
  - HDD with NVMe
  - SSD with NVMe
  - HDD with SSD and NVMe

Diverse Storage Profiles

- Storage profiles are changing
- Latency ranging from sub μs to 100's *ms* (6 orders of magnitude)
- History dependent behavior
  - SLC page pools (few percent of drive)
  - Emergency garbage collection
  - Scattered writes but single reads
- Workload dependent performance
  - Read / Write Mix
  - Drive idle time
  - Bandwidth vs IOPS

FreeBSD Issues

- VM/Buffer Cache schedules most I/O in system
- Buffer Cache tries to be nice to I/O system
  - Limits number of dirty buffers
  - Limits number of bytes being written concurrently
  - Uses Hi/Lo water marks to schedule work
  - Mostly static allocation of resources at boot
  - Limits generally Global
- CAM I/O Scheduler smooths out some performance quirks
  - Throttling here inefficient
  - Interacts poorly with global limits

Outline

- Netflix Background
  - Netflix Network
  - Netflix Issues
- **FreeBSD I/O Stack**
  - Overview
  - struct buf
  - Interfaces
- Back Pressure

FreeBSD I/O Stack

System Call Interface	
Active File Entries	
OBJECT/VNODE	
File Systems	
Page Cache	Upper ↑
GEOM	Lower ↓
Disk Driver	
Protocol/Transport	
Host Storage Adapter	
Newbus Bus Space busdma	

After Figure 7.1 in The Design and Implementation of the FreeBSD Operating System, 2015.

FreeBSD I/O Stack High Level Overview

- Upper half of I/O Stack focus of VM system
  - Buffer cache
  - Memory mapped files / devices
  - Loosely coupled user actions to device action
- GEOM handles partitioning, compression, encryption
  - Filters data (compression, encryption)
  - Muxes Many to one (partitioning)
  - Muxes One to Many (striping / RAID)
- CAM handles queuing and scheduling
  - Shapes flows to device
  - Limits requests to number of slots
  - Enforces rules (eg tagged vs non-tagged)
  - Multiplexes shared resources between devices

struct buf – What's in it?

- Maps a vnode + offset + len to memory / vm_pages
- List membership and bookkeeping
- Flags to note state
- struct bufobj
- biodone routine
- Credentials

struct buf – How's it used

- Schedules I/O to lower layers
- Tracks read ahead, write behind
- Caches most frequent blocks
- Managing working sets via pagers
- Buffer daemon

Buffer Daemon

- Runs from time to time
- Schedules dirty buffers for write
- Wakes up any processes sleeping about to dirty buffers
- Blocks on static limits

Buffer Cache Interfaces

- getblk and friends
- bread / bwrite and friends (bdwrite, bawrite, etc)
- bstrategy
- bufwait, bufsync, bufwrite, bufstrategy

struct bufobj

- Ties together the vnode and bufs to lower layers
- BO_STRATEGY decides what to do with the request (queue it, translate it, etc)
- BO_SYNC Do a VOP_SYNC to flush data on vnode
- BO_WRITE Write data with runningbufs enforcement
- BO_BDFLUSH Flush all dirty buffers asynchronously

Pagers

- Associates pages in VNODE or process with backing store
- Reads / writes pages
- Manages VM objects that back bufs.
- vnode_pager, swap_pager, device_pager, default_pager, phys_pager

Current write down path

- Before dirtying buffer, call `bwillwrite`, sleep if too many dirty buffers.
- Prepare buffer by dirtying it with data and locking pages
- call BO_WRITE (possibly sleeping for runningbuf in `bcanwrite`)
- call BO_STRATEGY
- g_vfs_strategy
- geom processes I/O
- bufdone

Outline

Netflix Background

- Netflix Network
- Netflix Issues

FreeBSD I/O Stack

- Overview
- struct buf
- Interfaces

Back Pressure

Back Pressure Design

- Each device publishes current capacity
- Lower levels pass this to the upper layers
- Upper layers limits requests voluntarily
- Old interfaces emulate old model
- New interfaces allow upper layers more flexibility

New: Submission/Completion Record

- Time scale for I/O quantum
- Bitmask: IOP or BW limited (or both)
- IOPS available in next quantum
- BW available in next quantum
- Estimates are based on estimated capacity of drive less scheduled I/O

New: BIO_IOCAP I/O Command

- Returns the instantaneous capacity estimate of the device
- Call is synchronous, but immediate
- Complicated GEOM like gmirror, graid responsible for coming up with something sensible
- Should be consistent with submission and completion reports.

New: BIO_* flags

- BIO_BP_NO_AUTO disables global back pressure for clients that know the new protocol
- BIO_BP_NO_SLEEP return EAGAIN if the request would exceed the device's current capacity.

New: Default I/O scheduler

- New I/O scheduler for bio
- Default behavior: check old global limits
- Other schedulers are possible

New: effective per-device runningbufs

- If device estimates capacity, then never exceed write capacity (either by sleeping or returning EAGAIN)
- Default I/O scheduler will estimate 1/2 of queue depth
- CAM Adaptive I/O scheduler limits based on it's estimates of the disk.

Problems

- Code still quite green
- Knowing when drive saturated hard problem
- CAM I/O scheduler work not done
- Analysis for starvation and other unfair behavior
- Interaction with Buffer Daemon
  - Global pool vs device information
  - PID control would be better at cleaning buffers
  - Lower-levels can know how much will likely be needed, but no connection to Buffer Daemon

Questions

Questions?
Comments?

Warner Losh

wlosh@netflix.com
imp@FreeBSD.org

http://people.freebsd.org/~imp/talks/bsdcon2017/slides.pdf