Netflix OpenConnect & FreeBSD

BSDCan DevSummit
May 15, 2013
Who are we?

- Scott Long <scottl@netflix.com>
  - FreeBSD 20+ year veteran
  - Former Release Engineer
  - Adaptec, Yahoo!, Netflix

- Alistair Crooks <agc@netflix.com>
  - Unix since V6, BSD since 4.1c
  - pkgsrc founder
  - NetBSD security-officer, core team
  - Wasabi, VISA Europe, Yahoo!, Netflix
What is Netflix Streaming?

• Amazon Web Services
  – Website, Business Functions, Authentication
  – Data Science
  – Encoding/Encryption
  – Command and Control

• Content Servers
  – Was Big-3 CDNs
  – Moving to “OpenConnect”
What does OpenConnect do?

- Brings content closer to the customer
- Saves ISPs and Netflix money on peering and transit costs
- Augments existing CDN capacity
What is OpenConnect?

- Webserver for terabits of static traffic
- Content delivery network - peering and embedding
- FreeBSD 9, nginx webserver, Bird BGP
- Off-the-shelf PC components
- High-Density, ISP-friendly Chassis
- http://openconnect.netflix.com
Typical Traffic Pattern
Building Block Architecture

• Horizontally and vertically scalable
• 1 box = 10% of the Netflix library
• 1 box = 5,000-15,000 streams
• 1 box = 60-80% bandwidth offload
• Fail-in-place design
• Fault tolerance via distributed copies, client-server feedback loop
Building Block Architecture
Initial design goals

• Modest compute resources
• ~10Gbps of traffic
• Maximized capacity: No RAID!
• No hot swap drives, few user-serviceable parts
• No SAS expander or other single-points-of-failure
• 600W power footprint, reasonable airflow, data-center friendly
Revision A Hardware

- Supermicro X9SCM-F, Intel E3-1260L
- Custom chassis, 4U x 25” deep
- 36 3TB Seagate Barracuda HDDs
- 2 Crucial M4 512GB SSDs
- 2 16-port LSI SAS/SATA
- 32 GB RAM
- Dual port Intel 10 GbE Fibre
- 8,000 - 10,000 clients, 8.5Gbps
Revision A Hardware
Revision C Hardware

- Custom chassis, 4U x 20” deep
- Supermicro X9SRL-F Motherboard
- Intel E5-2650 8-Core Xeon, 64GB RAM
- 36 Hitachi Enterprise 4TB HDD's
- 6 Crucial M4 512GB SSD's
- 4 8 port LSI SAS
- 2 Dual-port Chelsio 10GbE Fibre
- 15,000 clients, 15-18Gbps
Revision C Hardware
Revision D Hardware

- 1U Chassis
- Supermicro X9SRH-7F Motherboard
- Intel E5-2650 8-Core Xeon, 64GB RAM
- 14 Crucial M5 960GB SSDs
- Onboard 8-port LSI SAS
- Quad-port Chelsio 10GbE Fibre
- >20,000 connections, >20Gbps
Revision D Hardware
Structured Cabling
Why FreeBSD?

- Availability of expertise, outstanding community
- Works well, good vendor support
- No GPL
- Features used:
  - SUJ
  - gmirror – boot drive only
  - AIO
  - Dtrace, HWPMC
  - TCP Stack, modular CC
Netflix Contributions

- Camcontrol mods to download SATA firmware
- IPv6 ref counting fixes
- ixgbe interrupt mitigation, RX optimizations
- Fixes for isci driver for firmware download
- Collaboration with FF, Isilon on Unmapped I/O
- VM/VFS Tuning: vfs.read_min
Unmapped I/O
vfs.read_min
More than just code

• Community sponsorship
  – FreeBSD Foundation
  – MeetBSD, EuroBSDCon

• Working with Intel
  – Improve community relationships
  – Monthly meeting to discuss issues

• Advocate for FreeBSD with Supermicro, Seagate, HGST, LSI, Adaptec, etc
Challenges and Future Work

- **Disk I/O**
  - I/O scheduling
  - Command queue management
  - GEOM

- **Network**
  - Pipelining RX path
  - TCP Congestion Control
  - Traffic Classification/Prioritization
Challenges and Future Work

• Filesystem
  – Layout optimized for streaming
  – Journaling/SU bugs

• VM/Buffer/Cache
  – aio_sendfile()
  – LRU cache policy = worst case scenario

• FreeBSD 10
Review - what does an OCA do?

- Serves HTTP range requests to clients
- Communicates with control plane in AWS
- Allows ISPs to specify AS and CIDRs
- Hardware fail-in-place
- Serve and fill simultaneously
- In ISP or IX locations
- Currently serves 20%+ of US internet
OpenConnect Software

- FreeBSD 9.1 Stable
  - Sync every week with freebsd.org by **svn merge**
  - **nanobsd** is used to make 2 embedded images

- Nginx 1.2/1.4
  - Formerly sync’ed every week by **svn merge**
  - Now by **hg up**

- And....
Other parts of the system

- 2 images
  - 1 custom production-ready image
  - 1 GENERIC image; prod embedded in trash

- Scripts and programs
  - For nginx, bird/bird6, normal system configuration
  - For communications with control plane
  - Reporting and monitoring

- Netflix-specific ports tree
Packaging

- 51 ports/packages
  - some bespoke ones
  - fast digest functions
  - control plane communications and reporting
- Ports tree is location independent
- Sandbox builds in a chroot are used
  - avoid build system leaks
  - Binary packages on systems
What’s different?

- saved-options file as part of meta-data
- metadata versions are saved as part of pkg
- a single package defines OCA firmware level
- no indirection through system `.mk` files
- single script to make all packages in a chroot
- no version number necessary on command line
- no chroot building for src yet
Repository

- Subversion - easy to sync with freebsd/nginx
- Git mirror (but we know where the git user lives)
- Formerly sync'ed with Perforce
- Websvn for web-based access
  - primary source of truth for most users
- JIRA integration - ticketing and code review
Installer

• One size fits all
• Hardware-based profiles used
  – easy to add new hardware
  – try out new boards, memory or motherboards
• Disk sizes automatically calculated
Lessons learned

• Package-based approach
  – allows us to upgrade individual machines
  – is never used

• Cross-building of packages would be good
  – aio_mlock experiments with nginx
  – need a kernel with that system call in it
More lessons learned

• nanobsd’s /cfg is useful, but can be dangerous
  – need to umount before rebooting
• tracking stable has been good for us
• control plane-controlled firmware-refresh nice
• from previous lives - no local patches
Any questions?

Alistair Crooks
agc@netflix.com

Scott Long
scottl@netflix.com