FreeBSD Operations at Limelight Networks

The densely architected Limelight CDN bypasses the public Internet via 80+ interconnected locations for truly unlimited global reach.

ASN 22822
Stay here to hear about

Scale out operations using FreeBSD

Limelighters at BSDCan 2015

Kevin Bowling - presenter
Sean Bruno (sbruno@freebsd.org)
Hiren Panchasara (hiren@freebsd.org)
Jason Wolfe
Chris Christensen
Johannes Meixner (xmj@freebsd.org)
What does Limelight do?

Fast, Efficient Web Delivery at Global Scale

- **Website, Apps & APIs**
- **Edge Server Functionality**
  - HTTP & HTTPS Delivery
  - SmartPurge
  - Cloud Storage
  - Cache
- **Limeligh Network**
  - Automatic Replication
  - Dynamic Content Acceleration
  - Request for Web Content
- **DDoS Attack Scrubbing Center**
  - Detected Attack Traffic
  - Cleaned Traffic Returned
  - DDoS Attack Detection & Mitigation
- **Viewers**
POP Architecture

- DWDM gear
- Redundant Internet routers w/full routes
- Large chassis switches or spine switching mesh
- TOR switching
- Lots of these:
Let’s talk about Ops

There are plenty of folks talking about appliance, embedded, academic use. This talk was borne out of the desire to see more large “ops” installations talking about BSD.

Main difference.. systems are fluid - software and configuration are rolled out as standard operating procedure

Think: large web sites, API-centric companies, service providers
Workload almost exclusively consists of Internet facing services
My Entry Phase One: Analysis

My background: 10+ years professional Linux SysEng

“UNIX Aficionado” - but just a BSD observer - ran m0n0wall-> pfSense, dabbled with Net/Open/Free, AIX, IRIX, Solaris, etc

Start at LLNW - intrigued by BSD edge. “How are so few BSD people doing so much?”

Answer: BSD software and mindset. “I need to get involved with THAT”

Equal Answer: Observability trumps everything else
We use Zabbix and are generally happy with it

Getting it to run at scale took some doing, but it has been reliable

Key insight: use an API driven monitoring system.. monitoring should be configured by CM. Monitoring must be part of service entry into production. Monitoring should be part of testing/QA.

It’s 2015, stop deploying nagios
**Ops: Metrics**

**OpenTSDB**
HBase clusterfsck but write-scalable, somewhat read scalable
“So what you have a metrics dumping ground” Sean Chittenden @BSDCan - yeah more or less
I’m not so happy with this but it is sometimes better than nothing

**Jut.io**
Interesting data flow language startup, easy to aggregate data from multiple sources and APIs
Not quite metrics but mentioned here: Splunk, ELK
SaltStack is a Configuration Management tool built on an Orchestration bus. We think this is genius.
Limelight @ SaltConf15 [https://www.youtube.com/watch?v=4lhVOhPJABQ](https://www.youtube.com/watch?v=4lhVOhPJABQ)

DTrace-ifying 2000 machines:
salt 'cds*' cmd.script salt://local/dsack.d | grep DSACK

CM changes are a feedback loop
Changes to the CM system happen on the fly in containers

Simple state example

```bash
{% from "network_time/map.jinja" import ntp with context %}
{% if ntp['ntp-pkg'] %}
  ntp-pkg:
    pkg.installed:
      name: {{ ntp['ntp-pkg'] }}
  ntpd_conf:
    file.managed:
      - source: salt://network_time/files/ntp.conf
      - template: jinja
      - context:
        - config: {{ ntp['client_settings'] }}
          require:
            - pkg: ntp-pkg
        ntpd-service:
          service.running:
            - name: {{ ntp['ntp-service'] }}
            - enable: True
            - require:
              - pkg: ntp-pkg
          watch:
            - file: ntpd_conf
```
We use git to maintain two branches of FreeBSD, which we call:

- `llbsd-head` - follows FreeBSD.org HEAD branch with LLNW patches
- `llbsd-stable` - follows FreeBSD.org 10-STABLE branch with LLNW patches

**buildotron** - Jenkins jobs turn tags against these branches into built artifacts for deployment

**Vagrant** - offer developers and operators a production-like environment on their laptops
- helps greatly for new hires

**Packer** - boot ISO, add extra stuff, produce Vagrant Box
- we produce production Linux images with Packer.. much easier on FreeBSD because we can plug into build system

**Configuration Management** - extras for prod images and vagrant images
My Entry Phase Two: Deliberate Use

Attract a src team
Upstream all the things
Use ports best practices

Make the system do what you want deliberately, not accidentally
Starting a src team

The more scaled out, the more dividends src influence pay
- FreeBSD 8 -> 10 while reducing custom patch stack
- Multiqueue em driver
- ipfw on inbound only
- PLMTUD implementation
- calloutng fixes
- TCP customization
- MFCs as needed

How do you do it?
Watch or offer on freebsd-jobs@freebsd.org
Recruit at conferences
Do cool stuff sensibly and people will come to you
Working with src

Develop against HEAD
MFC to -STABLE
Do internal RelEng
Deploy to prod
Monitor
  Analyze
  Change
  Repeat

OODA loop or most simply a feedback loop
My Entry Phase Three: Now

Identify and support key features and **community**
Show company we are more **effective** and enlighten people that want to be the same
Empower service owners and operators

- Key technologies:
  - Base system building blocks
  - Poudreire + pkg
  - SaltStack

Elsewhere, perhaps?
  - ZFS
  - VIMAGE Jails (iocage)
We are a proud sponsor of the FreeBSD Foundation and think they are doing an excellent job

https://www.freebsdfoundation.org/donate/sponsors
Keep Calm
Deploy *BSD To Prod

Thank you!
Intel’s em driver on FreeBSD and Linux only uses one tx and rx queue.

Sean found through some digging that the hardware is actually capable of 2 tx and rx queues and patched the driver to use them.

On a lagg cds box, this distributes network processing path over 4 cores.

Previously ~1.2gbps with quality problems to ~1.9gbps with good quality (em FIFO seems too small so rare drops but no way to fix that in software)

We have 1000s of these boxes in production
Increasing performance and efficiency requires understanding both the application and OS (kernel, base libs)

I am a poor stand in for Brendan Gregg, but his books and talks are a fantastic resource for companies developing or deploying any software