ZFS development on FreeBSD

OpenZFS Developer Summit 2014

Xin Li, iXsystems, Inc./The FreeBSD Project

About myself

- FreeBSD developer since 2004; Release Engineer and Deputy Security Officer of the project;
- Works for iXsystems on FreeNAS and related products;
- Involved with ZFS maintenance on FreeBSD, keeping the codebase up-to-date with Illumos (most of the time the FreeBSD development trunk would catch up with Illumos changes in the mean of a few days).

Current FreeBSD ZFS applications

- As a general purpose server operating system on bare metal servers;
- As a storage appliance (e.g. FreeNAS; SpectraLogic BlackPearl, etc.);
- As a virtualization container (ClusterHQ/HyberCluster, etc.)

Current FreeBSD release workflow

- One active development trunk ("head" or -CURRENT), one or two maintenance development branch (-STABLE).
- "Major" (X.0) evert about 2 years, cut from -CURRENT;
- "Minor" (X.Y) release every about 9 months, cut from -STABLE;
- In the interim, security updates and serious bug/regressions fixed via binary patching mechanism, freebsd-update against
- Users may choose to run -STABLE and roll their own releases;
- Developers are generally suggested to run -CURRENT

Current FreeBSD release workflow (in release cycle)

- Source management via branching;
- A target release date is announced with schedule by release engineering lead to developers;
- The development branch (either "head" or "stable") enters code slush (no major changes) and then code freeze (no changes without explicit re@ approval)
- Several BETA, RC's would be produced from the development branch in about 2 weeks interval, installed to various FreeBSD.org cluster systems.
- Eventually re@ would name the branch as "RELEASE", tag after build and announce it to the world

FreeBSD ZFS development - goals

- Make it easier to upstream/downstream code changes (use compatibility shims to provide Solaris-alike kernel interfaces, limit and contain FreeBSD specific changes in ZFS codebase, etc.)
- Continuously improve performance and reliability on FreeBSD;
- Keep tree up-to-date with upstream Illumos codebase; provide same or better boot-ability via FreeBSD boot loader.

Current FreeBSD workflow

- Based on subversion;
- Illumos ZFS code imported into a vendor branch, merged against development trunk "head/", validated and committed; The imported layout is intended to make it easy to upstream/downstream code changes easily.
- FreeBSD specific fixes/improvements committed to development trunk directly; generic fixes usually upstreamed via Illumos bug tracking system;
- Code merged to FreeBSD "stable" branch(es) after a settle period, generally 2 weeks or so.

Tree structure

- src/[sys/]cddl/contrib/opensolaris/: vendor code, layout same as Illumos's usr/src/
- src/[sys/]cddl/compat/: FreeBSD compatibility shims to provide Solaris like semantics
- src/sys/cddl/boot/: vendor code that is used and modified specifically for boot loaders.

ZFS development on FreeBSD

- Performance improvements by mitigating lock contentions (upstreamed when applicable to Illumos);
- zvol performance improvements (bypassing GEOM layer; FreeBSD specific issue);
- sendfile(2) related improvements (FreeBSD specific)

Code availability

- FreeBSD:
  - GitHub: https://github.com/freebsd/freebsd
  - Subversion (official): https://svn0.{us-west,us-east,eu,ru}.FreeBSD.org/base/
- FreeNAS:
  - GitHub: https://github.com/trueos/trueos

Q&A