Think correctly.

Why WE like FreeBSD

Presented By: Yukaia & d3c4f

Who are we?

Who are we?

Yukaia

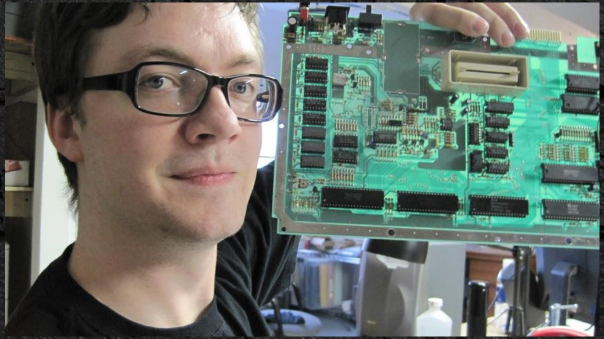
Who are we?

Yukaia

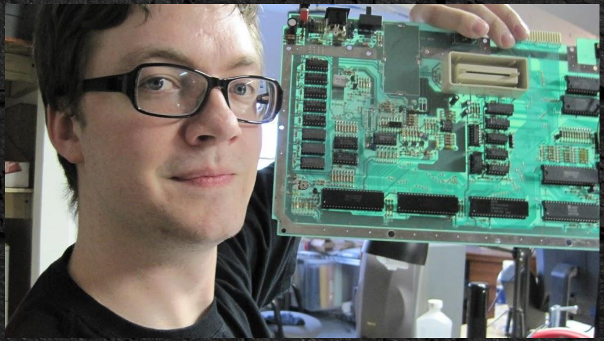
- Systems Administrator
- Automobile Enthusiast
- PortAParty Co-Creator
- theTransistor Founder
- Blackhat Volunteer
- Dr.Pepper Connoisseur

Who are we?

d3c4f



Who are we?



d3c4f

- Software Developer
- Systems, DB, Net Administrator
- PortAParty Co-Creator
- theTransistor Founder
- Caffeine Addict
- OpenWest Core Team
- BlackHat Volunteer
- Electronic Badge Designer

So what DO we like about FreeBSD?

So what DO we like about FreeBSD?

- Consistency & Stability
- Adheres to its standards
- ZFS (Zetabyte Filesystem)
- Fast & Lightweight
- Ports / Portmaster
- Jails / BHYVE
- PF
- The FreeBSD Handbook

What do WE use FreeBSD for?

Things we actually are using it for, or have used FreeBSD for in the past.

What do WE use FreeBSD for?

- PortAParty
  - GameCache, File Shares, Web Servers, (future) Game Servers, ELK / Data, Firewall, SNORT IDS/IPS, Network QOS, Routing
- Home
  - OpenVPN, NAS, Web Servers, GIT Server, Seedboxes, Disaster Recovery, Firewall, IDS/IPS, QOS, Routing
- Work / Fun
  - Web Servers, IRC Bouncers, Disaster Recovery, OpenVPN, Asterisk, Data Collection and Analysis, Firewalls, IDS / IPS, QOS, Routing, etc

The History of FreeBSD

The History of FreeBSD

1969

UNIX is created at Bell Labs. This was mostly written in PDP-7 assembly.

The History of FreeBSD

1973

UNIX is rewritten in C.

The History of FreeBSD

1974

UNIX is licensed to universities. For educational purposes, it is distributed with source code.

The History of FreeBSD

1974-1980

Berkley releases free patches for all of it's work on UNIX, so long as you have a license for UNIX.

The History of FreeBSD

1980

4BSD (Berkeley Software Distribution) is released by Berkeley as a complete system. It comes with Curses, Job Control in CSH, and more. At this point, it still does not have networking support.

The History of FreeBSD

1984

4.2BSD is released. This includes; Fast File System (or UFS), a modified version of BBNs TCP/IP implementation, and a handful of other changes. This is also the first version that Beastie the BSD Daemon appears (USENIX). DARPA begins funding the BSD project.

The History of FreeBSD

1986

4.3BSD is released. Machine dependent code is broken out to help make BSD more portable. BSD creates its own TCP/IP implementation, but DARPA pushes for BBNs TCP/IP implementation and that is included instead.

The History of FreeBSD

1989

Network Release 1, a separate release of just the networking code from 4.3BSD, is made freely available under the BSD license. This contained no AT&T code, which would have required users to pay for it. This is one of the first projects under BSD license.

The History of FreeBSD

1991

Net/2 is released. This is almost a complete OS. It is a rewrite of all the AT&T utilities in BSD. There are only a few AT&T files remaining in the Kernel, these are removed for the release of Net/2.

The History of FreeBSD

1991 (Later)

Net/2 becomes the base for two x86 ports: the open-source 386BSD, and closed-source BSD/386 (later BSD/OS) from BSDi. 386BSD doesn’t last very long, but becomes the foundation for NetBSD and FreeBSD.

The History of FreeBSD

1992

AT&T/USL sues BSDi and Berkeley over licensing concerns. An injunction on the distribution of Net/2 was issued for the duration of the lawsuit.

Note:

Linus Torvalds has said that if 386BSD had been available at the time, he probably would not have created Linux.

The History of FreeBSD

1993

FreeBSD 1.0 is released.

The History of FreeBSD

1994, January

4.4BSD is released. The AT&T/USL lawsuit against BSDi and Berkeley is settled, Berkeley wins! 3 Files have to be removed and 70 have to be updated to show AT&T/USL copyright notices (out of 18,000 files).

The History of FreeBSD

1994, June

4.4BSD-Lite and 4.4BSD-Encumbered are released. 4.4BSD-Lite contains no AT&T code.

The History of FreeBSD

1994, November

FreeBSD 2.0 is released. The original 386BSD code is replaced with 4.4BSD-Lite code.

Other BSD Versions and Projects

Other BSD Versions and Projects

- HardenedBSD (shout-out to lattera)
- OpenBSD
- NetBSD
- PFSense / OPNSense
  - (built on FreeBSD, for now)
- DragonflyBSD
- OpenDarwin
- PC-BSD

Other BSD Versions and Projects

HardenedBSD

- A security-hardened edition of FreeBSD
- Features:
  - ASLR (Address-Space Layout Randomization)
  - SECADM (Security Admin)
    - IntegriForce: Kernel-Based File Integrity System
    - ASLR can be enabled on a per jail/binary.
  - Segvgaurd
  - and many more security features

Other BSD Versions and Projects

OpenBSD

- Created by Theo de Raadt (a NetBSD Founder)
- Fork of NetBSD 1.0
- OpenSSH, OpenNTPD, PF, LibreSSL, OpenBGPD, and more are project that all originate from OpenBSD.

Other BSD Versions and Projects

DragonflyBSD

- Fork of FreeBSD 4.8 by Matthew Dillon
- Largely considered a research OS
- Is now very different from FreeBSD on a technical level

Other BSD Versions and Projects

PC-BSD

- The most popular graphical BSD desktop environment
- Based on FreeBSD (not a fork)
- Provides a graphical installer
- Creators of Lumina desktop environment, which is written for FreeBSD

FreeBSD / *BSD Stats & Who uses *BSD?

FreeBSD / *BSD Stats & Who uses *BSD?

- NetFlix
  - About 36.5% of ALL *peak* internet traffic in North America!

NOTE: It’s been estimated that about ½ of all North American internet traffic is running on FreeBSD.

FreeBSD / *BSD Stats & Who uses *BSD?

- Juniper, Cisco, Brocade, etc
  - Cisco: IronPort and their malware sandbox infrastructure as ran by their Talos Security Research Group
  - Many major products built on FreeBSD
- Other Gateways, Firewalls, Routers, etc

FreeBSD / *BSD Stats & Who uses *BSD?

- Mach Kernal (OSX)
  - OSX is based on NeXTSTEP which is based on the Mach Kernel and contains portions of BSD
  - The kernel that Apple uses/maintains is the XNU Kernel XNU stands for X is Not Unix, it's opensource and is used in iOS, OSX and Darwin.

FreeBSD / *BSD Stats & Who uses *BSD?

- Sony's Orbis OS (PS4)

FreeBSD / *BSD Stats & Who uses *BSD?

- The FreeBSD TCP/IP stack is considered the reference.

Break out the safeword. Time to talk BSD

Break out the safeword. Time to talk BSD

We’re going to cover, a little more in-depth:

- FreeBSD -vs- Linux
  - Similarities
  - Differences
- Features WE like
  - ZFS, Jails, BHYVE, Consistency of file locations, Speed / Security, PortMaster, ZSH/OMZSH

FreeBSD -vs- Linux: Similarities

FreeBSD -vs- Linux: Similarities

- Installation / CLI is very much the same
- Both are heavily based around the CLI
- Most file locations are the same
- Many of the same packages
- FREE (as in source)
- Both intend to be POSIX compliant
- Heavily technical user base
- Enterprise support is available for both

FreeBSD -vs- Linux: Differences

FreeBSD -vs- Linux: Differences

- FreeBSD is developed as an entire OS
  - All other BSD systems are the same
  - BSD OSes are developed independently of each other (including the Kernel). They often share patches and applications.
- Linux is just a Kernel

FreeBSD -vs- Linux: Differences

- File locations are actually consistent
  - Many Linux distros like to a ignore /usr/local/ or even worse, ignore it just most of the time.
  - FreeBSD keeps software that is not part of the base OS install pretty much limited to /usr/local/
    - ex. /usr/local/etc/ for configs
    - ex. /usr/local/bin/ for binaries
    - ex. /usr/local/sbin/ for system daemons
    - ...and so on. (It’s Consistent)

FreeBSD -vs- Linux: Differences

- FreeBSD is a democracy. LINUX is a ~~dicktatorship~~ dictatorship.
  - FreeBSD changes are guided a core-team
    - Responsible for deciding overall goals and managing areas of development
    - If you have a “Commit Bit” you can vote in core-team elections.
  - Linux changes are controlled by Linus Torvalds

FreeBSD -vs- Linux: Differences

- FreeBSD comes directly from UNIX
  - Forked from BSD, which was started by a group at Berkeley as a research project by Bill Joy to extend UNIX functionality
- Linux was developed from the ground up
  - Started by Linus Torvalds

FreeBSD -vs- Linux: Differences

- FreeBSD is mostly POSIX compliant
- Linux, while currently mostly POSIX compliant, is moving away from these standards (ex. SystemD).

FreeBSD -vs- Linux: Differences

- FreeBSD is licensed under the BSD License
  - VERY permissive license. Not copy-left
  - Focus is on the user, only need to keep headers
  - Simplified 2-clause license
- Linux is licensed under GPLv2
  - Code is guaranteed freedom. Not the user
  - Comparatively complex code publishing requirements

FreeBSD -vs- Linux: Differences

- FreeBSD has a nicer (ioho) Init System
- Linux SysV / Upstart were a bit of a pain. Got replaced with a whole OS. (SystemD, yuck!)

FreeBSD -vs- Linux: Differences

- FreeBSD has better support for building packages from scratch. And promotes this method. (Still has pre-built packages)
- Linux promotes installing pre-built packages instead of building. (Still has build-from-source option)

FreeBSD features we like

FreeBSD features we like

Speed / Security

- FreeBSD is very organized. This helps greatly!
- FreeBSD is developed around speed and stability
- Includes many security features
  - Check out HardenedBSD!
    - Kernel-Level File Integrity Checks
    - ASLR
    - and tons more!
- https://www.freebsd.org/features.html for more information.

FreeBSD features we like

Ports

- Found in /usr/ports/
- Each individual port has its own directory
- Each has its own makefile, and description file
- Each includes patches it needs to build and run

FreeBSD features we like

Ports

- There are a couple ways to install/upgrade the ports tree:
  - portsnap
  - subversion

FreeBSD features we like

Ports

- portsnap ports tree installation
  - Download compressed Ports tree to /var/db/portsnap/
    - *# portsnap fetch*
  - Extract to /usr/ports/
    - *# portsnap extract*
  - Update at a later time:
    - *# portsnap fetch update*

FreeBSD features we like

Ports

- subversion ports tree installation
  - a little more involved (not bad though)
  - used if you need more control
  - we aren’t going to cover this in this talk.

FreeBSD features we like

Building and Installing Ports

FreeBSD features we like

Building and Installing Ports

Ports can be built and installed using a few methods.

- make
  - Using normal make commands to build packages
- portmaster
  - Package management for port building

FreeBSD features we like

Portmaster

FreeBSD features we like

Portmaster

- First build portmaster (after you install ports)

  -
    ```
    # cd /usr/ports/ports-mgmt/portmaster
    ```

  -
    ```
    # make install clean
    ```

FreeBSD features we like

Portmaster

- Find a port to install
  - *# whereis [program name]*
  - Browse the /usr/ports/ directory
  - Search https://freshports.org/

FreeBSD features we like

Portmaster

- Build and Install a port
  - *# portmaster [category]/[port_name]*
- Update all outdated ports
  - *# portmaster -a*
- Update all outdated, and clean distfiles
  - *# portmaster -ad*

FreeBSD features we like

pkg (pkgng)

FreeBSD features we like

pkg (pkgng)

“I don’t like building ports, it’s scary and takes a long time. Can’t I just get prebuilt binaries with default options?”

(yes, yes you can.)

FreeBSD features we like

pkg (pkgng)

- Get information about a specific package:
  - *# pkg info [packagename]*
- Install a package:
  - *# pkg install [packagename]*
- Remove a package:
  - *# pkg delete [packagename]*

FreeBSD features we like

pkg (pkgng)

- Audit installed packages
  - *# pkg audit -F*
- Automatically remove leaf dependencies
  - *# pkg autoremove*
- Remove stale packages:
  - *# pkg clean*

FreeBSD features we like

pkg (pkgng) and portmaster notes

- Ports is latest stable release. pkg is usually only a few days behind. FreeBSD does weekly package builds on Wednesdays.
- If using both systems, you should not upgrade packages with pkg, it can break things. Only upgrade with ports.
- The best option is just to pick one or the other.

FreeBSD features we like

Service installation notes

- To automatically start a newly installed service, you typically just edit the /etc/rc.conf file.
- Make sure the following is inside that file
  - [service_name]_enable="YES"
- To start or stop the service manually
  - *# service [service_name] start*
  - *# service [service_name] stop*

FreeBSD features we like

ZFS (ZetaByte File System)

- 128 Bit, B-Tree, Copy-On-Write
- Every Block of Data has a checksum, stored in pointer
- Built-In Volume Manager, no need for LVM
- Block-level DeDup, snapshots, compression & clones
- RAID 0, 1, Z, Z2, Z3, and more
- Multiple Block-Caching Levels to RAM and SSD
- Keeps an intent log (ZIL) for data writes
- Offloaded ZILs enable asynchronous writes

FreeBSD features we like

BHYVE

- BSD Licensed, legacy-free HyperVisor
- Still fairly new
- Can run FreeBSD, OpenBSD, NetBSD and Linux VMs
- Windows support is in progress
- Supports AMD64 and i386 architectures

FreeBSD features we like

Jails

- OS Level Virtualization (aka containerization)
- Has been in FreeBSD since version 4 (March 2000)
- Creates a safe environment, away from the rest of the system.
- Processes ran inside a Jail cannot access files or resources outside of the Jail.

FreeBSD features we like

PF (OpenBSD Packet Filter)

- Integrated as part of the base since FreeBSD 5.3
- Complete, full featured firewall
- Provides QOS
- Utilizes ALTQ for traffic shaping
- Easy to read rules

FreeBSD features we like

PF (OpenBSD Packet Filter)

- Enable PF by editing /etc/rc.conf, add:
    - pf_enable=”YES”
    - pf_flags=”” #additional flags for pfctl startup
    - pf_rules=”/etc/pf.conf” #default ruleset
- If there is a LAN behind the firewall, or NAT is required
    - gateway_enable=”YES”
- Start PF manually
    - *# service pf start*

FreeBSD features we like

PF (OpenBSD Packet Filter)

- Enable PF Logging by editing /etc/rc.conf, add:
  - pflog_enable=”YES”
  - pflog_logfile=”/var/log/pflog”
  - pflog_flags=”” #additional options
- Start PF manually
  - *# service pflog start*

FreeBSD features we like

PF (OpenBSD Packet Filter)

- To control PF, use *pfctl*
    - Enable PF:
        - *# pfctl -e*
    - Disable PF:
        - *# pfctl -d*
    - Flush NAT, Filter, State, and Table Rules, and reload
        - *# pfctl -F all -f /etc/pf.conf*
    - Check configuration for error, do not load
        - *# pfctl -vnf /etc/pf.conf*

FreeBSD features we like

Linux Binary Compatibility

FreeBSD provides a Kernel Module that enables the use of most 32-bit Linux binaries (unmodified). In some situations Linux applications actually perform better on FreeBSD then they do on Linux.

- http://www.phoronix.com/scan.php?page=article&item=linux_games_bsd
- http://www.phoronix.com/vr.php?view=18989

FreeBSD features we like

Linux Binary Compatibility

- To enable 32-bit Linux Binary Support run:
  - *# kldload linux*
- Verify that the Kernel Module has been loaded:
  - *# kldstat*
- To enable at runtime, edit /etc/rc.conf
  - linux_enable=”YES”

Literature & References

- The FreeBSD Handbook
  - Some of the best project documentation
  - https://www.freebsd.org/doc/handbook/
- Books
  - http://www.nostarch.com/abs_bsd2.htm
- Links
  - https://www.over-yonder.net/~fullermd/rants/bsd4linux/01
  - https://youtu.be/5mv_oKFzACM
  - http://www.bsdnow.tv
  - http://bsdmag.org

Demo Time (time-permitting)

- Installation
- Get and extract the ports database
- Build and install portmaster
- Use portmaster to build and install vim-lite
- Show /etc/rc.d
- Start/Stop Service(s)
- Show directory structure
- ???

Q & A

(Ask now)

That's all, Folks!

We hope you enjoyed this presentation. Now get out there and install some FreeBSD / *BSD systems!