NetBSD -- Open Source Operating System

- Berkeley CSRG -- Unix V7 -> BSD
- Net 1 and Net 2 tapes
- 386BSD, DDJ, Jolitz
- 386BSD -> Patch set -> NetBSD & FreeBSD
- 0.8 20 Apr 1993 (i386 only)
  - First commit Mar 21, 1993
- 1.0 26 Oct 1994
  - Amigas
  - i386-family PCs
  - some m68k-based Macintoshes
  - PC532 (Portmaster...P. Nelson)
  - SPARCstation (sun4c) -class machines
- 2.0 9 Dec 2004
  - 54 different System architectures
  - 17 CPU families
- 4.0 Dec 2007 -- Built on WWU old cluster
- 7.1 current "stable" version
- 8.0 cut and being worked on...
Distributions ... 

- Base system
- X Window system
- Package System

- Package vs Port
  - port: causing NetBSD to run on a new machine
  - package: a 3rd party program ready to run on NetBSD
  - Package system used on a variety of systems

- Local copy of NetBSD 7.1

- /home/phil/public/NetBSD-7.1
  - iso image
  - source "tar balls"
    - `tar xzf src.tgz` -- example command to extract source
Source Code Layout

- `/usr/src` -- (normal) root of full tree
  - `src/...` indicates root of source tree

- `src/sys` -- root of system source
- `src/sys/arch` -- machine dependent code
- `src/sys/dev` -- machine independent devices
- `src/sys/kern` -- core routines of kernel ...
- `src/sys/sys` -- kernel include files not found elsewhere
- `src/lib` -- library sources including system call source
Building Kernels

- cd sys/arch/i386/conf
  - cd sys/arch/XXXX/conf -- for another arch!
- cp GENERIC MYKERN
- edit MYKERN for hardware / options
- config MYKERN
- cd ../compile/MYKERN
- (make -j8 depend ; make -j8 ) >& ERRS &
Build system in src/

- build.sh: created for cross build
- Standard order: build tools, build requested
  - ./build.sh -j 8 distribution
    - -j 8 says up to 8 make jobs at the same time
  - ./build.sh -j 8 kernel=GENERIC
    - created directory .../i386/compile/obj/GENERIC
  - USETOOLS=no ./build.sh ....
    - supposed to not use "special tools directory"
    - Works best with tools "./build.sh -j8 tools"
- Build system complex -- may need some time to learn it.
- For this class, use the by hand method.
- Capturing errors!
  - make >& ERRS & (requires bash)
  - tail -f ERRS
  - make > ERRS 2>&1 & (sh or ksh)
  - tail -f ERRS
Installing a new kernel

- cp netbsd /netbsd
- cp netbsd /netbsd-2008-03-30
- cd /; ln netbsd-2008-03-30 netbsd

Better yet, don’t overwrite your install kernel: netbsd
- cp netbsd /netbsd.test

On boot, interrupt boot process
- Drop to boot prompt (option 5?)
- boot netbsd.test
  - boot flag:  -s -- single user mode
  - /etc/ttys:  secure flag

Shutting down your system:
- shutdown -hp now  -- halt and power down
- shutdown -r now  -- reboot
  - Assumes /sbin is on your PATH, also include /usr/sbin
  - Members of group "operator" can run shutdown as that user
All relative to the src/sys directory unless stated different

- Process abstraction
  - "single thread of control"
  - "multiple threads of control"
- sys/proc.h -- process table

- States: (p_stat)
  - Creation
  - Active
  - Stopped
  - Zombie

- (threads are runnable and stopped, sys/lwp.h)
- curlwp -- current running thread
- curproc -- curlwp->l_proc

- Process switch
  - mi_switch: kern/kern_synch.c
  - cpu_switch: arch/X/X/locore.s
Interrupt Priority Levels

- Hardware levels, priority interrupts, blocked interrupts
- Setting Hardware Levels (i386/include/intr.h)
  - spl0() -- normal
  - splhigh
  - softclock, net, tty, bio, imp, clock, vm
- Race conditions -- top half vs bottom half
  
  ```c
  x = splbio();
  <<< code >>>
  splx(x);
  ```
- Deadlock problems
- tsleep() / wakeup() -- Old versions
- convar(9)
  - cv_init(), cv_destroy(), cv_wait(), cv_signal(), cv_broadcast()
- Multi-processor issues (locking)
  - Use of convar(9) for synchronization
System Startup!

- `start`, arch/pc532/pc532/locore.s (~phil/public/pc532)
- `init532()`, arch/pc532/pc532/machdep.c
- or check out arch/vax/vax/machdep.c
- `main()`, kern/init_main.c

Kernel Entry

- System Calls (Top Half Entry)
  - trap instruction
  - `lib/libc/sys`, `lib/libc/arch/ns32k/SYS.h`
  - `arch/pc532/pc532/trap.c`, locore.s
  - dispatch table / hardware dependent

- Interrupts (Bottom Half Entry)
  - interrupts, vectored, priority
  - Other traps ... divide by zero, page fault
Devices

Device Classes: Block and Character (sys/sys/conf.h)

- **Block**
  - Disk, tape ...
  - File system mounts block devices only
  - Reads/Writes through buffer cache
  - API
    - open(), close(), strategy(), ioctl(), dump(), psize(), discar()

- **Character**
  - direct access
    - may be sequential, possibly random access
    - no buffer cache
  - API
    - open(), close(), read(), write(), ioctl(), stop(),
      tty(), poll(), mmap(), kqfilter(), discard()

- Some devices have both block and character interfaces
- Many have only character interfaces
- /dev/...., mknod name [c|b] major minor
Kernel view
- sys/conf.h
  - bdevsw
  - cdevsw
- dispatch tables -- points to driver entries
- NetBSD 1.6 and before -- initialized in arch/x/x/conf.c
- NetBSD 2.0 and later -- config builds definitions
  - Definitions in sys/arch/x/conf/majors.x
  - And in sys/conf/majors
  - dir compile/KERNNAME
    - devsw.c -- bdevsw and cdevsw tables
    - ioconf.c -- configure() database
- Major numbers index Xdevsw tables.
- Minor numbers passed to driver as "unit number"
- chrtobltbl: maps character major -> block major or NODEV
Vnode -- file system basics

- Kernel struct with information, sys/vnode.h
- each file and directory gets a vnode
- vnode operations via a dispatch table
- each vnode has a pointer to vnode ops (vop)
- namei() in kern/vfs_lookup.c ... lookup() creates vnodes!
- each vnode has a pointer to private data for the FS
Following a system call

Assume \( fd = \text{open} ("/dev/somecdev", \text{O_RDONLY}, 0) \)

Follow: \( \text{ret} = \text{read} (\text{fd}, \text{buffer}, 100) \)

- read(), lib/libc/arch/vax, lib/libc/sys
- syscall(), sys/arch/vax/vax/{intvec.S, syscall.c}
- sysent[], kern/init_sysent.c, sys/systm.h
- sys_read(), kern/sys_generic.c
- dofileread(), kern/sys_generic.c
  - fp->f_ops->fo_read, sys/file.h (struct fileops)
- struct fileops vnops: kern/vfs_vnops.c
- vn_read: kern/vfs_vnops.c
- VOP_READ: kern/vnode_if.c (VCALL, sys/vnode.h)
- v_op: sys/vnode.h
- spec_read: miscfs/specfs/spec_vnops.c
- dev_read: dev/isa/satlink.c, .../X/conf/majors.X