15-410

"Now that we've covered the 1970's..."

Plan 9
Apr. 9, 2007

Dave Eckhardt

Bruce Maggs

Synchronization

Survey

- How many have installed *nix on a box?
  - Windows?
- How many have *done an in-place upgrade?*
- How many have a personally owned box with multiple users?
  - Done an upgrade?
- What does “PC” stand for?

Today: Plan 9 from Bell Labs

Overview

What style of computing?

- **The death of timesharing**
- **The "Unix workstation problem"**

Design principles

Runtime environment

File servers (TCP file system)

Name spaces

Timesharing

One computer per ...

- City: Multics
- Campus: IBM mainframe
- Department: minicomputer

Benefits

- Sharing, protection easy inside “the community”
  - Easy to add a “user” to access control list (or user group)
- Administration amortized across user base
  - Backups & printers, too...

The *Personal Computing* Revolution

Consequence of the microprocessor

Get *your own* machine!

No more "disk quota"

***You* decide which software is on the box**

- **Upgrade whenever *you* want**
  - **Mainframe sysadmin's schedule is *always* too (fast xor slow)**

Great!

The Rallying Cry

One of the Alto's most attractive features is that it does not run faster at night.

- **Butler Lampson?**

The Personal Computing *Disaster*

You do your own backups

- **Probably not!**

You do emergency security upgrades

- **Day or night!**

Sharing files is hard, risky

- **machine:/usr/... (until it retires)**

Every machine you use has different software

- **If you're lucky, packages are just missing**
- **If you're unlucky, they're there with subtly wrong versions**
  - **Or different machines have different fonts –whee!**

Hybrid Approach

Distributed computing

- Centralize “the right” resources
  - Backed-up, easily-shared file systems
  - Complex (licensed) software packages
  - Version management / bug patches
- Access those resources from a fast local machine

Which OS on the servers?

- Don't care – black boxes

Which OS on the workstation?

Workstation Operating Systems

Unix?

- Good: It's the system you're used to using
- Bad: Administer it yourself
  - /etc/passwd, /etc/group, anti-relay your sendmail...

Windows

- Your very own copy of VMS!
- Support for organization-wide user directory
- Firm central control over machine
  - “install software” is a privilege
- Access to *services* is tied to *machines*
- Firmly client/server (no distributed execution)

Workstation Operating Systems

Mac OS 9

- Your own ... whatever it was

Mac OS X

- Your own Unix system! (see above)

VM/CMS or MVS!!!

- IBM PC XT/370
- Your own *mainframe!*
  - You and your *whole family* can (must) administer it

The “Network Computer”

- Your own display, keyboard, mouse
- Log in to a real computer for your real computing
- Every keystroke & every mouse click cross the net
  - Every font glyph...
- Also known as
  - Thin client, X terminal, Windows Terminal Services
- Once “The Next Big Thing”
  - (thud)

The Core Issues

Who defines and administers resources?

What travels across the network?

- **X terminal: keystrokes, bitmaps**
- **AFS: files**

Are legacy OS's right for this job?

The Plan 9 Approach

"Build a UNIX out of little systems"

- **...not "a system out of little Unixes"**

Compatibility of essence with Unix

- **Not real portability**

Take the good things

- **Tree-structured file system**
- **"Everything is a file" model**

Toss the rest (ttys, *signals!*)

Design Principles

"Everything is a file"

- Standard *naming system* for all resources: pathnames

"Remote access" is the common case

- Standard *resource access protocol:* "9P"
- Used to access any file-like thing, remote or local

Personal namespaces

- Naming *conventions* keep it sane

A practical issue: Open Source

- Unix source not available at "Bell Labs", its birthplace!

System Architecture

Reliable machine-room *file servers*

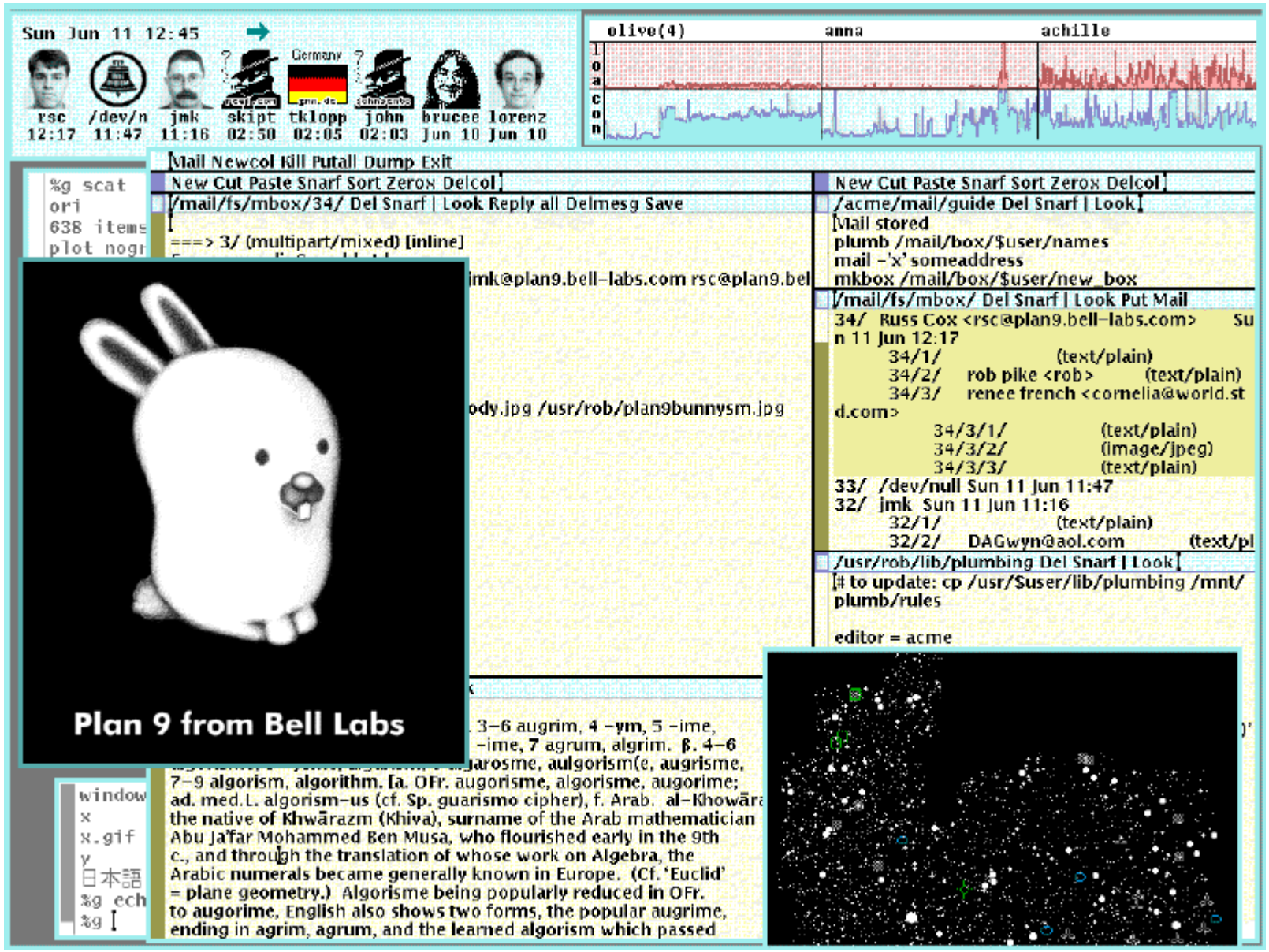
- **Plan 9's eternal versioned file system**

Shared-memory multiprocessor *cycle servers*

- **Located near file servers for fast access**

Remote-access workstation *terminals*

- **Access your *view* of the environment**
- **Don't *contain* your environment**
- **Disk is optional**
  - **Typically used for faster booting, file cache**
- **“Root directory” is located on your primary file server**


Sun Jun 11 12:45
rsc 12:17
/dev/n 11:47
jmk 11:16
skipt 02:50
tkloppe 02:05
john 02:03
brucee Jun 10
lorenz Jun 10
olive(4)
anna
achille
Mail Newcol Kill Putall Dump Exit
New Cut Paste Snarf Sort Zerox Delcol
/mail/fs/mbox/34/ Del Snarf | Look Reply all Delmesg Save
===> 3/ (multipart/mixed) [inline]
/acme/mail/guide Del Snarf | Look
Mail stored
plumb /mail/box/$user/names
mail -'x' someaddress
mkbox /mail/box/$user/new_box
/mail/fs/mbox/ Del Snarf | Look Put Mail
34/ Russ Cox <rsc@plan9.bell-labs.com>
34/1/ (text/plain)
34/2/ rob pike <rob> (text/plain)
34/3/ renee french <cornelia@world.std.com>
34/3/1/ (text/plain)
34/3/2/ (image/jpeg)
34/3/3/ (text/plain)
33/ /dev/null Sun 11 Jun 11:47
32/ jmk Sun 11 Jun 11:16
32/1/ (text/plain)
32/2/ DAGwyn@aol.com
/usr/rob/lib/plumbing Del Snarf | Look
# to update: cp /usr/$user/lib/plumbing /mnt/plumb/rules
editor = acme
Plan 9 from Bell Labs

Custom Namespaces

/bin/date means *your architecture*'s binary

/dev/cons means *your* terminal

- **Per-*window* devices (below)**

/mail/fs/mbox/25 is the 25th message in your box

No "links" - "hard" or "soft"

- **A link is something *in the file system* which causes everybody to buy into a naming illusion**
  - **Some illusions cause security holes, as we've seen**
- **In Plan 9, namespaces are *consensual* illusions**
  - **Stored in process control blocks, not in the file system**

The /bin File System

Look, Ma, no $PATH!

```
% bind /386/bin /bin
% bind -a /rc/bin /bin
% bind -a /usr/davide/386/bin /bin
```

/bin is a *union* directory

- **Each backing directory searched in order**

/dev/tty vs. /dev/cons

`% (process_foo <foo >bar ) >&errs`

- csh-speak for
  - Run “process_foo”
  - Standard input is “foo”
  - Standard output sent to “bar”
  - Standard error sent to “errs”

“process_foo” is pretty well connected to *files*

- What if it wants to talk *to the user?*

/dev/tty vs. /dev/cons

`% (process_foo <foo >bar ) >&errs`

- csh-speak for
  - Run “process_foo”
  - Standard input is “foo”
  - Standard output sent to “bar”
  - Standard error sent to “errs”

“process_foo” is pretty well connected to *files*

- What if it wants to talk *to the user?*

Unix solution –magic device “/dev/tty”

- Rummages through your process, guesses your terminal
  - See O_NOCTTY flag to open(2), see vhangup(2)-or don't...
- Opens /dev/ttyXX for you, returns that

/dev/tty vs. /dev/cons

```
% (process_foo <foo >bar ) >&errs
```

What if process_foo wants to talk *to the user?*

Plan 9 –your *namespace* contains /dev/cons

- **The right device is *mounted* as /dev/cons**
- **By whoever runs you**
  - **window manager, login, remote login**
- **Unix question: what is the name of the terminal I'm running on? ttyp7? ttyq9?**
- **Plan 9 answer: whoever connected you to your terminal arranged for it to have the conventional name - /dev/cons**

/dev/tty vs. /dev/cons

Unix remote login

- /dev/tty delegates to /dev/ttyp1
  - "pseudo-tty" - careful emulation of a serial line
- master (/dev/ptyp1) is managed by sshd
- ASCII characters flow across the network
- Your ssh client is running on /dev/ttyq3
  - Which is connected to a screen window by "xterm"
- What happens when you resize your xterm??

Plan 9 remote login

- Shell's /dev/cons is a *remote file mount* of a window
- Same as if the window were local (albeit slower)

Per-Window Devices

X: a complex monolithic server somewhere

- House of a thousand mysteries
- *Not* on the 15-410 reading list: ICCCM
  - "Inter-client communication conventions manual"

Plan 9: Per-*window* devices

- I/O - /dev/mouse, /dev/cursor, /dev/cons
- Contents - /dev/text, /dev/window
- Window title - /dev/label (a 1-line text file)
- Working directory - /dev/wdir

```
% echo top > /dev/wctl
```

  - Requests window manager to bring your window to top

Per-Window Devices

Screen shot

```
% cp /dev/screen /tmp/screen-image
```

Window shot

```
% cp /dev/window /tmp/window-image
```

The Serial-Port File System

Look, Ma, no ioctl()!

```
% bind -a '#t' /dev
% echo b9600 > /dev/eia1ctl
% echo “foo” > /dev/eia1
```

The CD-Burner File System

Burn audio tracks to CD

```
% cdfs -d /dev/sdD0
```

- Uses /dev/sdD0/raw to send SCSI commands to disk
- Mounts itself as /mnt/cd in your namespace

```
% cp *.cda /mnt/cd/wa/
```

- Write CD-Audio tracks to “write audio” directory

```
% rm /mnt/cd/wa
```

- Remove writing directory to close session

```
% echo eject > /mnt/cd/ctl
```

The tar-ball File System

Rummage through a tar file

```
% fs/tarfs -m /n/tarball foo.tar
% cat /n/tarball/README
```

The TCP File System

Look, Ma, no finger command!

```
#!/bin/rc
# hold clone & ctl open during connection
{ conn=`{read} cd /net/tcp/$conn
  { echo 'connect 128.2.194.80!79' > ctl ;
    echo davide > data; cat data } < ctl
} < /net/tcp/clone
```

Look, Ma, no NAT proxy setup!

```
% import gateway.srv /net/tcp
```

The /tmp Problem

Unix /tmp: security hole generator

Programs write /tmp/program.3802398

- **Or /tmp/program.$USER.3432432**

No name collision "in practice"

- **Unless *an adversary* is doing the practicing**

```
% ln -s /tmp/program.3802398 /.cshrc
```

- **Now a setuid-root program will put your commands into root's .cshrc...**

Fixing /tmp

No inter-user security problem if *only one user!*

Plan 9 /tmp is per-user

- **User chooses what backs the /tmp name**
  - **Temporary "RAM disk" file system?**
  - **/usr/$user/tmp**

Matches (sloppy) programmer mental model

Plan 9 3-Level File Store

Exports one tree spanning many disks

- Users bind parts of the tree into namespaces

Original implementation –3-level store

- RAM caches disks, disks cache WORM jukebox

Plug-compatible modern implementation

- Hash-capability log-structured disk store

Daily snapshots, available forever

- /n/dump/1995/0315 is 1995-03-15 snapshot
- Time travel without “restoring from tape”
  - Public files are *eternally* public –be careful!

Plan 9 Process Model

New-process model

- fork()/mount()/exec()

System calls block

Task/thread continuum via rfork()

- Resources are shared/copied/blank
  - Namespace, environment strings
  - File descriptor table, memory segments, notes
  - Rendezvous space
- rfork() w/o “new process” bit edits current process

Process Synchronization

rendezvous(tag, value)

- Sleeps until a 2nd process presents matching tag
- Two processes swap values
- "Tag space" sharing via rfork() like other resources

Shared-memory spin-locks

Summary

Files, files, files

- "Plumber" paper
  - Programmable file server
  - Parses strings, extracts filenames
  - Sends filenames to programs
  - File, file, blah, blah, ho hum?
- Isn't it cleaner than
  - Sockets, ICCCM, RPC program numbers, CORBA?

Not just another reimplementation of 1970

- Every compile is a cross-compile
- Every debug is a remote cross-platform debug
- Unicode everywhere
- ...

More Information

"Gold Server" multi-computer environment approach

- How to build a system out of a bunch of Unixes
- Similar approach to Andrew
- Difficult
- http://www.infrastructures.org/papers/bootstrap/

Plan 9

- http://www.cs.bell-labs.com/plan9/

Disclaimer

A distributed system is a system in which I can't do my work because some computer has failed that I've never even heard of.

- **Leslie Lamport**