Operating Systems

Joseph Hallett

April 29, 2024

It has been brought to my attention that I have a reputation...

CSS TIMES

Home Sport Crossword Art Quiz

BREAKING: University of Bristol Lecturer Caught Red-Handed Using a Windows Machine!

By Joe Barlow

February 18 2023

In a shocking turn of events, renowned computer science lecturer Joseph Halett of the University of Bristol was caught using a Windows machine instead of his beloved Linux software.

I have some opinions about this...

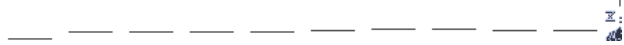
Firstly:

> *I'd just like to interject for a moment. What you're referring to as Linux, is in fact, GNU/Linux, or as I've recently taken to calling it, GNU plus Linux. Linux is not an operating system unto itself, but rather another free component of a fully functioning GNU system made useful by the GNU corelibs, shell utilities and vital system components comprising a full OS as defined by* POSIX. *:-P*

I have some more opinions about this...

Secondly:

> When asked about the incident, Halett simply stated: "I needed to check my email, and my Linux machine was down for maintenance. I had no choice but to use a Windows machine. I swear, it's a one-time thing."



Is *entirely* fair.

- I will ~~give exam answers~~ personally thank any student who gives me a working `mbsync` config for the University exchange service.

But mostly...

- My *official* University computers are Macs
- My *personal* laptop I write most of my lectures on usuall runs:

```
$ uname -a
OpenBSD bananaslug 7.5 GENERIC#79 amd64
```

Well except....

```
$ uname -a
Linux bananaslug 6.8.7-arch1-1 #1 SMP PREEMPT_DYNAMIC Wed, 17 Apr 2024 15:20:28 +0000 x86_64 GNU/Linux
```

(Y'see I'm teaching Software Tools... and need to be able to test the VMs ...otherwise I have to use ~~one of those awful~~ *those really rather lovely* lab machines).
(My server runs OpenBSD still though...)

Confessions of an OS fan

I <3 operating systems
But when I look round the labs I see most people using Windows

- A few Macs
- A few people running Linux in Windows
- And the odd Linux machine (typically Ubuntu or Pop/OS)

Where have all the weird OSs gone?

So what's the plan?

In this lecture:

- A bit of a history lesson
- Go through how different OSs grew
- Where different features came from
- Show you some of the *weirder* OSs

In the lab:

- Lets install some different OSs
- *Play* with the different systems

None of this is examinable

- If you don't want to come to the lab that's fine
- But you're gonna miss out on some weird tech!

In the beginning

There were *mainframe* computers...

- Large complicated bits of *big iron*
  - (as opposed to measely server *tin*)
- You could run a whole business off of one (and we still do)

The earliest computers (1950s) ran programs directly loaded into memory off of *punched cards*.

- But after a while we started to think it'd be nice to have some utilities stored
- Help you read data and load programs automatically
- So by the 1960s the first OSs started to appear...

Some of the *earliest* mainframe OSs included IBM's System/360

Oh and by the way these things still exist...

IBM still sell mainframes.
And the modern z/OS is still compatible with System/360 programs

- (and Linux)

Why would you use one?

- Big overpowered server machines
- With *really* large and fast numbers of CPUs and memory
- Good for running whole organisations off of
- Good for data-center scale analytics

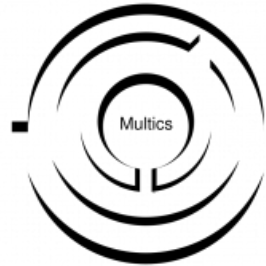
And then...

Dennis Ritchie and Ken Thompson (and others) were working on a mainframe OS called Multics

- The *Multiplexed Information and Computing Service*
- ...first OS to have a *shell* that'd be recognisable today
- ...it had a new release last year!

Multics was big and complex

- Project was massively overrunning
- Bell Labs folks were frankly sick of it
- ~1970 they decided to write a *simpler* version for cheaper and more computers
- Called it *Unics*: *Uniplexed Information and Computing Service*



UNIX System V

Unics got renamed Unix by Brian Kernighan
UNIX System 5 (1983) is when things start to get interesting

- At this stage its a popular OS for ***many different*** computers
- Especially in universities...

IEEE chose it as the basis for their Operating System Standard IEEE 1003 (IEEE-IX)

- Richard Stallman suggests the name **POSIX** for it

Operating systems are freely shared

- Ken Thompson has a sabbatical at Berkeley and helps them install UNIX System 6
- Together with Bill Joy (`vi`) starts a ***distribution*** of UNIX: **BSD**

Also in the 1980s

Cheap(er) home *micro*-computers are starting to become available

- Typically running off of cassette tapes…
- But the fancier ones have disk drives and a *little* persistant storage

Typically run operating systems called *Disk Operating Systems* or DOS

- Apple DOS for the Apple][
- PCDOS for for IBM Personal Computer (later renamed MS-DOS)
- CP/M for DEC machines
- C64 DOS for the Commodore 64
- Atari DOS for the Atari

If you use Powershell or `cmd.exe` in the labs… you're using a descendant of DOS!

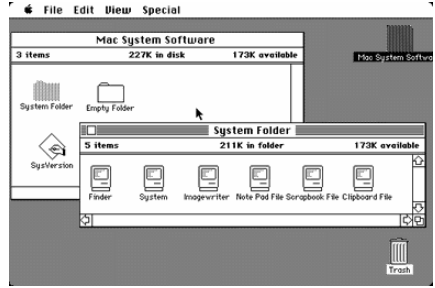
In Cupertino...

Apple make their first Mac OS (System 1) in 1984

- Licenses Xerox PARCs new *graphical user interface* and *desktop* metafores
- First widely available OS that looks like a modern OS
- 1985 Steve Jobs gets kicked out of Apple

Microsoft try and copy it for Windows 1.0 (1985)

- But it's rather clunky
- But by Windows 3.1 (1991) it's rather good...



Back in UNIX land...

By the early 1990s

- Andrew Tannenbaum has a teaching UNIX clone called *Minix*
- BSD is up to version 4.3

Copyright is starting to be a thing with computers

- UNIX owners getting snippy at BSD
- William and Lynne Jolitz at Berkely do a clean room reimplementation of the more protected bits
- Call it *386BSD*

Steve Jobs goes and founds a computer company called *NeXT*

- Bases their operating system on 4.3 BSD
- Calls it *NeXTSTEP*
- Initially a commercial failure, but by 1991 starting to be popular in academia
- Tim Berners-Lee makes the internet for it

Then theres this guy...

Creates his own kernel based on Minix

Anyone want to guess what he called it?

He thought about calling it *Linux* but decided it was to egotistical

- A colleague who uploaded it to the university's FTP server disagreed and renamed it
- Linus accepted that its now called *Linux*

It starts to get popular as a free UNIX clone
GNU userland ported to it *until they can build their own kernel* (HURD)

- They never do

Back in Cupertino

Apple without Steve doesn't do too well

- Apple spends its last pennies buying NeXT, under the agreement that Steve will be the new CEO
- Steve cancels the clones and weird projects

Starts the process of replacing Mac OS System 7/8 with a new OS that merges the work NeXT had been doing

- This will become MacOS X
- Ever done any Mac programming and wondered why all the classes descend from an `NSObject`?

You can run Mac OS System 7 in a browser!

- `https://system7.app`

First iMac comes out

- My Dad brings home a Bondi Blue iMac
- Had a ***built in modem*** and came with a Demon Internet sign up
- No floppy disks… but new Iomega Zip drives

Still the most beautiful computer ever

- I completely fell in love
- *Terrible* mouse though

MacOS X/macOS

Apple switches from the PowerPC architecture to i386 around 2000

- Mac OS System 9 becomes Mac OS 10… branded Mac OS X

Looks like a Mac

- Internally its NeXTStep with the Mach kernel from MIT (another UNIX clone)
- Userland from FreeBSD (check the man pages!)

NeXTSTEP apps become Mac things

- Only *Chess* in your Utilities folder is *completely* unchanged
- Some hangovers from *Classic Mac OS* still there (and a translation layer called *Rosetta*)

Server version becomes *fully POSIX/UNIX compliant* in order to deal with an advertising lawsuit

iOS released in 2006…

- Even more NeXTSTEP-y internally
- But completely new UI

Microsoft admit their OS is rubbish

Linux continues to grow, especially for servers

- *Many* distributions of Linux become available

In 1993 Microsoft adds the POSIX subsystem

- Then replaces it with Windows Services for UNIX (really BSD)
- Then replaces it with Windows Subsystem for Linux in 2016

Computer science students everywhere rejoice because they don't have to choose between having an easy life at uni, and having to install a weird OS that won't play games

- But it still is rubbish and janky

And so we arrive at the current day...

We have MacOS that is based on 4.3 BSD

We have Linux that is based on a clone of 4.3 BSD

We have Windows DOS + a janky Linux-layer

This makes me sad...

Innovation in OS design didn't end in 1986
Most users don't have a choice in what OS they use

- But we're experts...
- We should try and use weird systems
- We should help build new paradigms

It doesn't *have* to be a monoculture!

The BSDs are still going!

BSD is still out there

386BSD became *FreeBSD* when merged with BSD 4.4

NetBSD became a fork of 386BSD focussed on supporting more systems

OpenBSD forked NetBSD, audited the code and aggressively removed legacy cruft

DragonflyBSD forked *FreeBSD* to add a new filesystem

They're just like Linux for the most part

- But more holistically designed
- Better documented
- FreeBSD even has Linux emulation
  - OpenBSD had it but decided it was cruft, alongside bluetooth

You're probably even using BSD...

Played on a Nintendo Switch? Runs FreeBSD

Played on a PS3/PS4/PS5? Runs FreeBSD

Watched Netflix? All runs off of FreeBSD servers

Using ZFS? It wasn't developed on Linux...

`ssh`'d into the labs? OpenBSD maintain OpenSSH

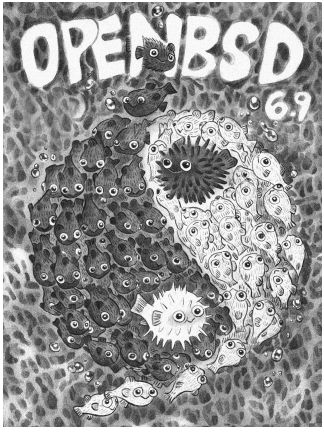
Used `tmux`? OpenBSD maintain it

Used a firewall on a Mac/PF? OpenBSD code

Using ROP? W $\oplus$ X memory? Stack canaries...? OpenBSD did the first implementations

Try running one next time you fancy trying a new OS!

I use OpenBSD!



I've been using it continuously since OpenBSD 6.9

- About 3 years now

What I like

- Really simple
- Either just works or is never going to work
- Or I can fix **without swearing** by looking at the code and fixing it
- No SystemD... just old school shell scripts
- Man pages contain everything
- Michael W. Lucas books
- Doesn't get in my way
- Fixes stupid design decisions

What I don't like

- Slow
- Virtualization sucks
- Some software missing (Dyalog APL, Signal, Torch, any games...)
- Fixing stupid design decisions breaks things occasionally

Suppose I want to restrict an app to just using the safe stuff from libc

Linux has Seccomp BPF for that!

```
static int install_filter(void) {
  struct sock_filter filter[] = {
    /* Grab the system call number */
    BPF_STMT(BPF_LD+BPF_W+BPF_ABS, syscall_nr),
    /* Jump table for allowed syscalls */
    BPF_JUMP(BPF_JMP+BPF_JEQ+BPF_J, __NR_rt_sigreturn, 0, 1),
    BPF_STMT(BPF_RET+BPF_K, SECCOMP_RET_ALLOW),
    #ifdef __NR_sigreturn
    BPF_JUMP(BPF_JMP+BPF_JEQ+BPF_K, __NR_sigreturn, 0, 1),
    BPF_STMT(BPF_RET+BPF_K, SECCOMP_RET_ALLOW),
    #endif
    BPF_JUMP(BPF_JMP+BPF_JEQ+BPF_K, __NR_exit_group, 0, 1),
    BPF_STMT(BPF_RET+BPF_K, SECCOMP_RET_ALLOW),
    BPF_JUMP(BPF_JMP+BPF_JEQ+BPF_K, __NR_exit, 0, 1),
    BPF_STMT(BPF_RET+BPF_K, SECCOMP_RET_ALLOW),
    BPF_JUMP(BPF_JMP+BPF_JEQ+BPF_K, __NR_read, 0, 1),
    BPF_JUMP(BPF_JMP+BPF_JEQ+BPF_K, __NR_write, 0, 1),

    /* Check that read is only using stdin */
    BPF_STMT(BPF_LD+BPF_W+BPF_ABS, syscall_arg(0)),
    BPF_JUMP(BPF_JMP+BPF_JEQ+BPF_K, STDIN_FILENO, 4, 0),
    BPF_STMT(BPF_RET+BPF_K, SECCOMP_RET_KILL),
```

We're not done yet!

```
    /* Check that write is only using stdout */
    BPF_STMT(BPF_LD+BPF_W+BPF_ABS, syscall_arg(0)),
    BPF_JUMP(BPF_JMP+BPF_JEQ+BPF_K, STDOUT_FILENO, 1, 0),
    /* Trap attempts to write to stderr */
    BPF_JUMP(BPF_JMP+BPF_JEQ+BPF_K, STDERR_FILENO, 1, 2),

    BPF_STMT(BPF_RET+BPF_K, SECCOMP_RET_ALLOW),
    BPF_STMT(BPF_RET+BPF_K, SECCOMP_RET_TRAP),
    BPF_STMT(BPF_RET+BPF_K, SECCOMP_RET_KILL),
  };

  struct sock_fprog prog = {
    .len = (unsigned short)(sizeof(filter)/sizeof(filter[0])),
    .filter = filter,
  };

  prctl(PR_SET_NO_NEW_PRIVS, 1, 0, 0, 0);
  prctl(PR_SET_SECCOMP, SECCOMP_MODE_FILTER, &prog);
  return 0;
}
```

Seccomp-BPF is an entire programming language enabling limited code to be run in the kernel without needing to be the superuser safely

- Really powerful
- Great for monitoring programs and debugging
- ...who doesn't enjoy writing assembly language in C?

And in OpenBSD land...

```
void install_filter(void) {
  pledge("stdio");
}
```

Pledge restrict program features

Unveil restrict program paths

To be fair Linux's Seccomp BPF is a lot more powerful than OpenBSD's pledge and unveil...

- But sometimes you just want to secure the app
- And sometimes you don't want to have to write C pseudo assembly for a firewall to do it

The Bell Labs team didn't retire after UNIX...

What do you do after you invent the OS that defines the modern world?

- You invent a new better OS!

Plan 9

- Released to research in 1992, and the public in 1995
- *EVERYTHING* is a file
- Rob Pike and Ken Thompson invent UTF-8 for it (on the back of a napkin)
- Terminals replaced with a GUI
- Designed for distributed systems (the cloud before we called it that)

Ideas from Plan 9 have appeared in modern OSs

- `/proc` `/sys` `debugfs` filesystems to allow configuration and monitoring of processes and the kernel through files
- Go compiler infrastructure is Plan 9's (***Rob Pike*** built both!)

Go to `www.9front.org` for a modern distribution

- *Beware:* lots of old school CS "humour" and memes
- MNT Reform laptop officially supports it?!

For example...

How do you read where the mouse is?

On Linux via an `ioctl`

On Plan 9 `cat /dev/mouse`

How do you draw to the screen?

On Linux More `ioctl` magic!

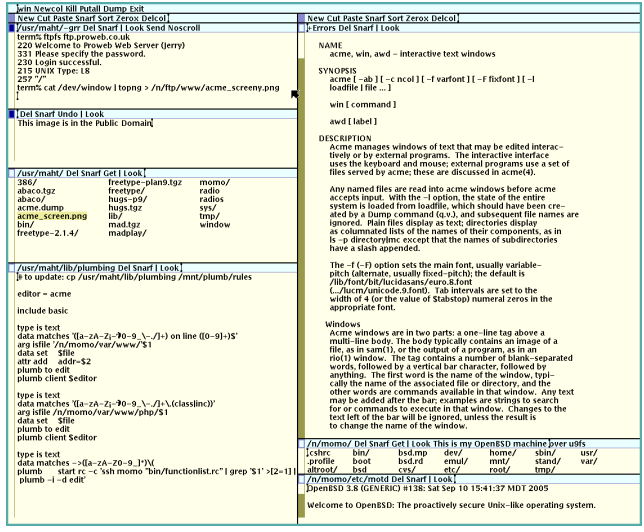
On Plan 9 Open `/dev/draw/new`

How do you find information about a process

On Linux In `/proc`

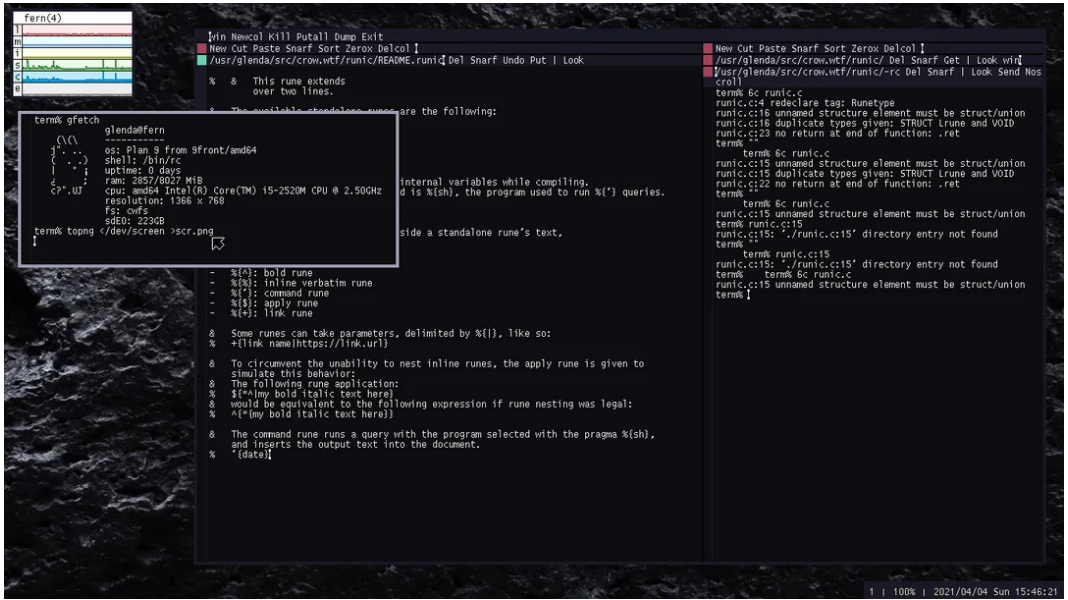
On Plan 9 ...where'd you think Linux copied this feature from? `;-)`

ACME and Plan 9



Available as part of 9`ports` in most other OSs.

People even theme it now?!



(via http://reddit.com/r/unixporn)

Qubes OS

A *reasonably* secure OS

- *"If you're serious about security QubesOS is the best OS available today. It's what I use, and free."* — Edward Snowdon

Designed by Joanna Rutkowska

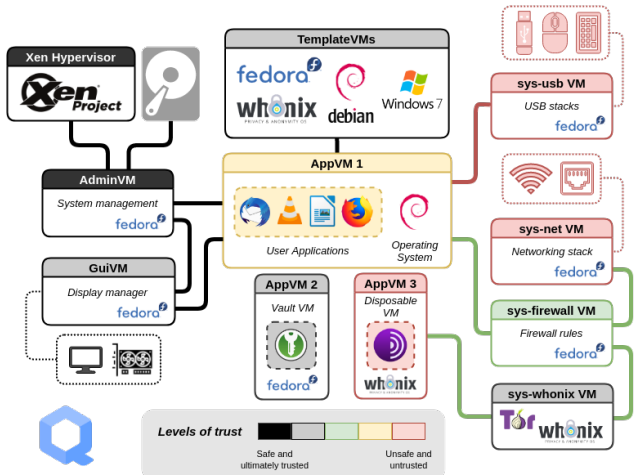
- Based on Xen and Linux *sort of*...

Basic ideas:

- Virtualise *everything*
- Nothing is permenant

`https://www.qubes-os.org`

Qubes Architecture



Downsides...

You need to be able to spawn lots of VMs

- So *somewhat* odd hardware requirements
- Its a *bit* slower
- Usability/ricing is an afterthought

But...

- It feels like something from the future
- Hot swapping distros in *seconds* (well... tens of seconds) makes you giddy
- I wrote my PhD thesis using it :-)

Ideas from Qubes are starting to appear in Linux and iOS

- *Fedora Silverblue* implements a **lot** of Qubes's ideas, but in a more usable way

Version 4.2 came out last year!

The Uxn CPU

What if we all die in global thermonuclear war/global warming/out of sheer ennui with modern life?

- No more making new computers
- Need to reuse the ones we already have!

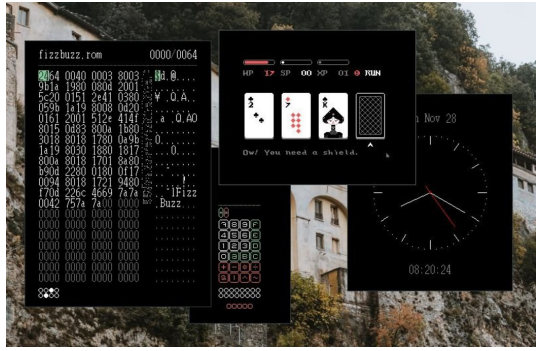
Whats the ***minimum viable computer*** we can use until the mutant-tuna enslave us all?

Uxn

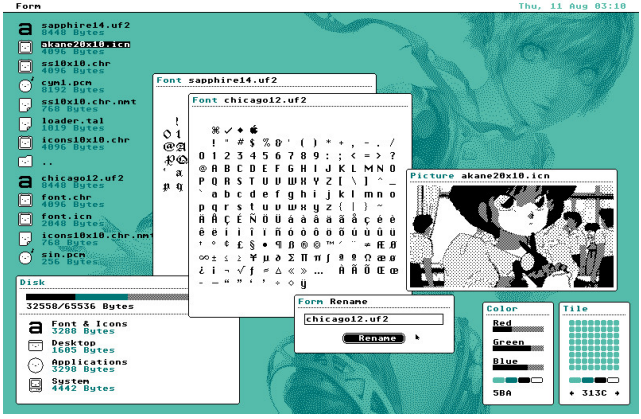
Designed by ***Hundred Rabbits*** (***Rek*** and ***Devine***) to be an OS that can run on other computers

- Can run on a 6502 processor (a NES), and fairly easy to port to anything else
- Stack based vm (Forth programming!)

`https://100r.co/site/uxn.html`



Uxn now has an OS

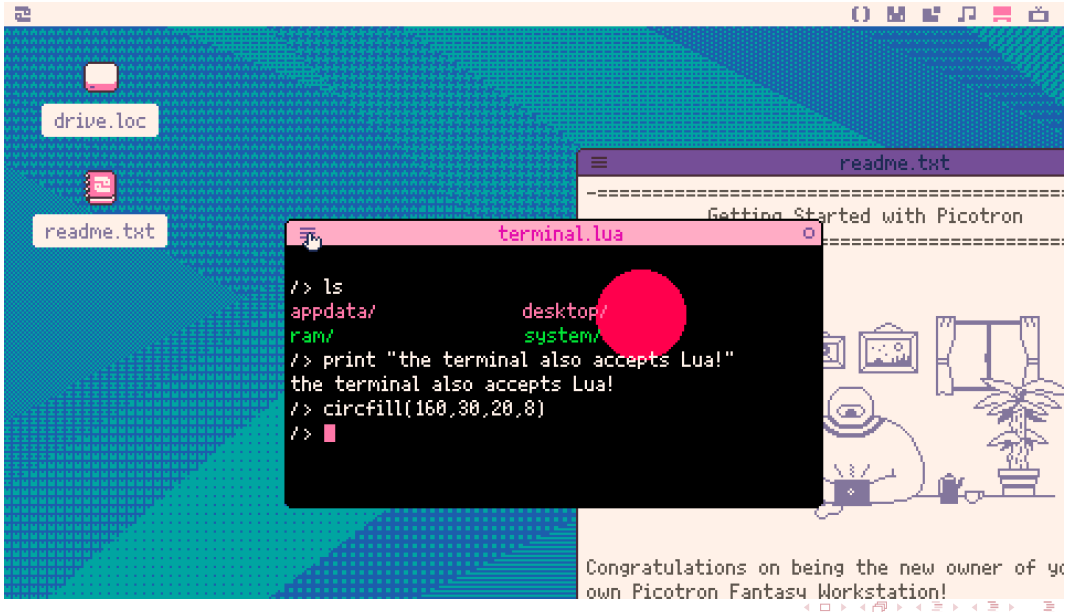


Because the Uxn virtual machine has been ported everywhere... how do you load files on a NES?

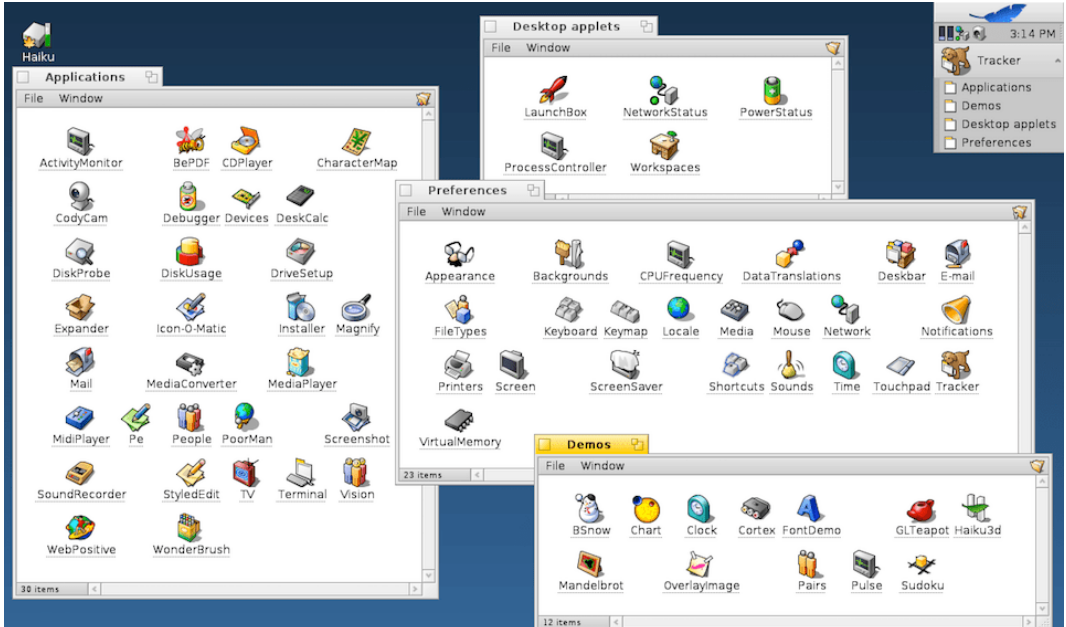
- You could do with a *real* OS to manage stuff

Potato Uxn is a simple little OS for the Uxn CPU!

Picotron


drive.loc
readme.txt
terminal.lua
/> ls
appdata/
desktop/
ram/
system/
/> print "the terminal also accepts Lua!"
the terminal also accepts Lua!
/> circfill(160,30,20,8)
/>
readme.txt
Getting Started with Picotron
Congratulations on being the new owner of yo
own Picotron Fantasy Workstation!

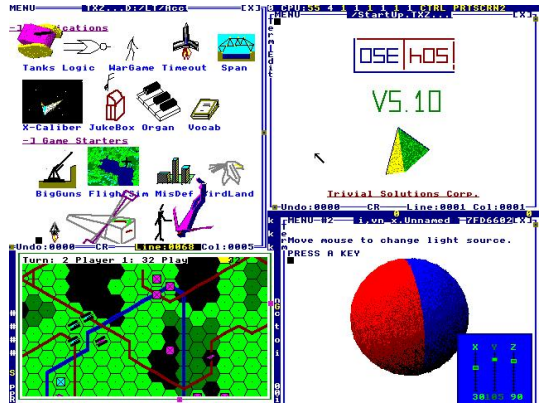
Haiku


Haiku
Applications
File Window
ActivityMonitor
BePDF
CDPlayer
CharacterMap
CodyCam
Debugger
Devices
DeskCalc
DiskProbe
DiskUsage
DriveSetup
Expander
Icon-O-Matic
Installer
Magnify
Mail
MediaConverter
MediaPlayer
MidiPlayer
Pe
People
PoorMan
Screenshot
SoundRecorder
StyledEdit
TV
Terminal
Vision
WebPositive
WonderBrush
30 items
Desktop applets
File Window
LaunchBox
NetworkStatus
PowerStatus
ProcessController
Workspaces
3:14 PM
Tracker
Applications
Demos
Desktop applets
Preferences
Preferences
File Window
Appearance
Backgrounds
CPUFrequency
DataTranslations
Deskbar
E-mail
FileTypes
Keyboard
Keymap
Locale
Media
Mouse
Network
Notifications
Printers
Screen
ScreenSaver
Shortcuts
Sounds
Time
Touchpad
Tracker
VirtualMemory
23 items
Demos
File Window
BSnow
Chart
Clock
Cortex
FontDemo
GLTeapot
Haiku3d
Mandelbrot
OverlayImage
Pairs
Pulse
Sudoku
12 items

Temple OS

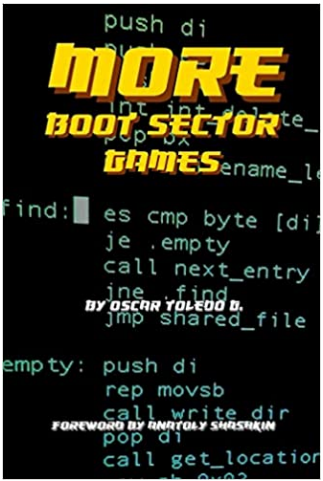
Temple OS is a 64bit OS written by Terry Davis

- Features its own compiler, filesystem, graphics and UI library
- Is fully runtime recompilable *(like a Lisp machine!)*
- Has 3D games and simulations
- An immence work of art (and devotion) by **1 person**



Note: Terry Davis was extremely ill, and died in 2018.

Or you could write your own!



Fantastic low-level assembly coding book that goes through the beginnings of building your own DOS-like OS

- But start with the first book ***Boot Sector Games*** if you're interested `;-)`
- Shows you how to implement ***Snake!***

Might even convince you assembly programming is fun?

- Or at least make you comfortable with the coding should you take COMSM0049 ~~Systems Software Security~~ *Hacking*
- Seriously… take me and Sana's unit… we break stuff!

And many more!

OSs are fun

Try something weirder!

Other OSs I haven't mentioned but you should try!

Redox UNIX but rewritten by people who really like *Rust*

Mezzano Lets rebuild the Lisp machines!

Emacs Comes with a bad text editor `;-)`

Or at least use the good Windows...