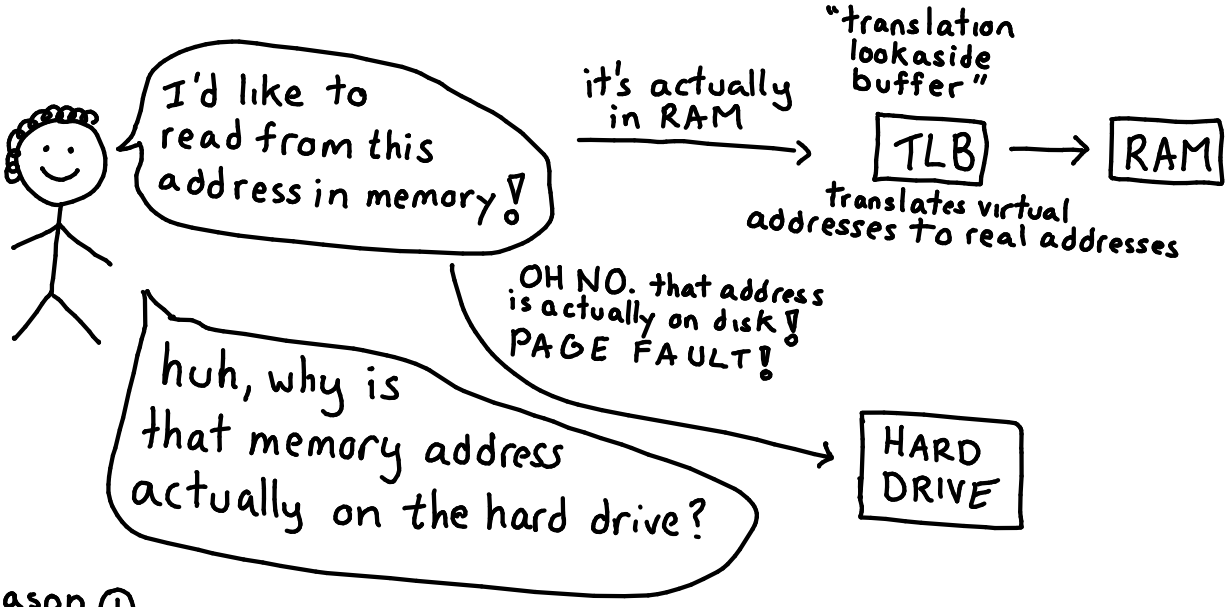
VIRTUAL MEMORY



reason ①

mmap

lets you map a bunch of stuff on disk into memory. None of it will actually get read from disk until you access the memory.

reason ②

swap

if you run out of memory, it gets saved to disk and your computer gets SUPER SLOW !!

assembly

JULIA EVANS
@b0rk

we hear computers "think in binary". But what does that MEAN??

your computer's memory (RAM)

0100100100010100010 0100

some of this is cat pictures

some of this is ★ programs ★

programs are binary

0100101101110101106

ooo what does THAT mean

those are INSTRUCTIONS

every CPU has an instruction set (x86 or ARM) usually

some instructions

jmp mov add xor push inc

instructions are numbers

the inc instruction ("increment") on x86 is 1000000 or 0x40

assemblers translate "human readable" assembly code into binary

assembly code → assembler → binary

```
mov $1, %rax      ...01001001
mov $1, %rdi      0....
xor %rdi, %rdi
```

register

the instruction register contains an address in RAM.

CPU → IR 0x5884

oo I will go look in RAM there and run the code I find!

"C" stands for linearizable

the CAP theorem

Julia Evans
@b0rk

from Martin Kleppmann's "A critique of the CAP theorem"

in distributed systems, network partitions happen

?!?

hello?

computer

someone unplugged a cable!

garbage collection!

too much network traffic!

if you want to be consistent you can't always be available

panda

elephant

you're gonna have to wait for an answer

"CP systems"

consul

zookeeper

etcd

chubby

when they reply, you can believe them, but they don't always give you answers

"AP systems" available + partition tolerant

this doesn't mean very much.

always return "lol"

very carefully considered weaker consistency model

You can call both of these "AP"

CAP is a very simple theorem

I read the whole proof! It took 10 minutes + there's no math

CAP won't help you reason about most systems

I have a replicated database what can you tell me?

nothing!!

CAP

user space vs. kernel space

Julia Evans
@b0rk

drawings.jvns.ca

the Linux kernel has millions of lines of code

- read + write files
- decide which programs get to use the CPU
- make the keyboard work

When Linux kernel code runs, that's called

kernel space

When your program runs, that's

user space

time for a context switch to kernel space

your program switches back and forth

```
str = "my string"
x = x+2
file.write(str)   <- switch to kernel space
y = x+4           <- aand we're back to user space!
str = str * y
```

timing your process

```
$ time find /home
0.15 user   0.73 system
```

time spent in your process

time spent by the kernel doing work for your process

What's a thread?

Julia Evans
@b0rk

drawings.jvns.ca

a process can have lots of threads

process id	thread id
1888	1888
1888	1892
1888	1893
1888	2007

threads share memory

thread 1: I'm going to write "3" to address 2977886

I can overwrite that if I want! — thread 2

but they can run different code

thread 1: I'm doing a calculation!

I'm waiting for a network request to finish! — thread 2

sharing memory can cause problems

"race conditions"

memory: 23

thread 1: I'm going to add 1 to that number! ⟷ same time ⟷ I'm going to add 1 to that number! — thread 2

RESULT: 24 ← WRONG (should be 25)

if you have 8 CPU cores, you can run code for 8 threads at the SAME TIME

1 5
2 6
3 7
4 8
CPU cores

SO BUSY !! !!

Julia Evans
@b0rk

copy on write

drawings.jvns.ca

every time you start a new process on Linux, it does a fork() or "clone" which copies the parent process

old ⟷ SAME ⟷ new

the cloned process has EXACTLY the same memory

3GB of RAM

old → 3GB of RAM ← new

copying all the memory every time we fork would be slow and a waste of space.

the new process isn't even gonna use that memory most of the time!

so Linux lets them share RAM instead of copying

oh no! won't the processes pollute each others' memory?

how do we make this work?!

Linux marks all the memory for both processes as read-only (in the page table)

① I'm going to write to the shared memory!

② CPU: UH OH that is not allowed! Linux! PAGE FAULT!

③ Linux: no problem! I will just make a copy of that piece of memory.

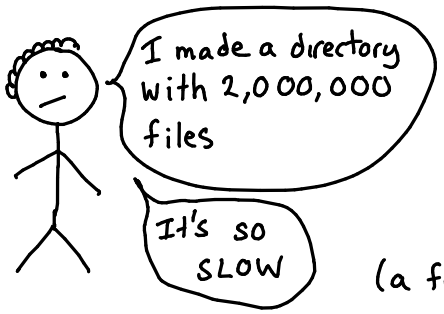
④ everyone is happy ♡

directories + symlinks

@b0rk
Julia Evans
drawings.jvns.ca

What's a directory?

filename	inode number
awesome.jpg	279932
blah.txt	13227
cumberbatch	233333



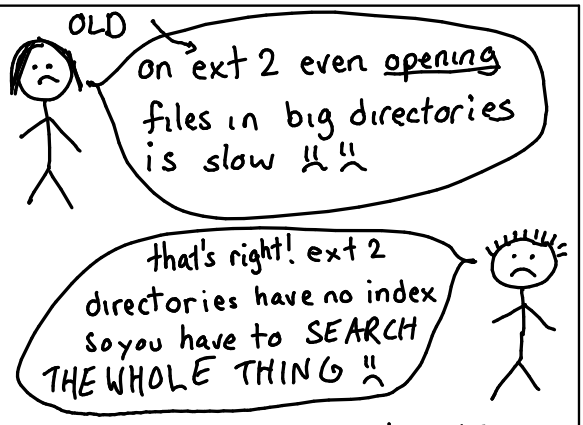
listing your directory is gonna be REAL SLOW

(a few seconds at least)

what's a symlink?

it's just a file with the name of another file in it!

```
$ readlink my-cool-link
/home/julia/long-complicated-file-name
```



ext 2 is OLD though. ext3 is OK.

more at drawings.jvns.ca

☆ the stack ☆

(in a C program)

JULIA EVANS
@b0rk

your program has

-> local variables

```
int x = 2;
```

-> a function to return to

```
void parent() {
    do_thing();
}
```

-> function arguments

```
make_cat(name, fluffiness)
```

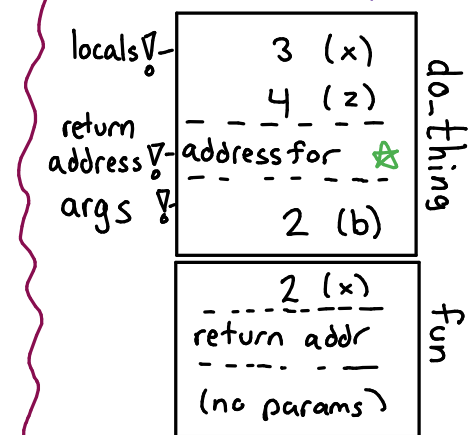
these all live in a part of memory called **the stack**

example program

```
int fun() {
    int x = 2;
    do_thing(2);
    int y = 4;
}
```

```
void do_thing(b) {
    int x = b+1;
    int z = 4;
    return;
}
```

the stack at ⇐



there's a limit to how big your stack can get! Exceed it and you get a STACK OVERFLOW

drawings.jvns.ca

CPU scheduling

Julia Evans
@b0rk

once upon a time...

I want to run a program!

Sorry I can only do 1 program at a time

COMP UTARR from the 60s or so

your computer today

I want to run!

program

no, me!

no, me!

ME

operating system

I have stuff to do too you know!

every CPU core can only run 1 program at a time

ok time to stop it's Jimmy's turn to use the CPU now

operating system

every program gets a few ms at a time

steps when we switch the running process

"context switch"

-> save:
- registers
- stack pointer
- which CPU instruction to start at next time

-> set up memory for new process

-> load new registers and stuff

all this takes time (2 microseconds?). It's ok to do but you don't want to be switching processes constantly

you don't use the CPU when you're waiting

hey I'm waiting for a network response

cool! I'll run other stuff until that comes back.

OS

how does DNS work?

Julia Evans
@b0rk

more of these at drawings.jvns.ca

DNS servers translate names to IP addresses

where's cats.com?

17.2.3.9 <- this is called an A record

most DNS queries get cached

where's cats.com?

8.8.8.8 Google DNS server. looks up cats.com for you and gives you an answer.

authoritative DNS server for cats.com

sometimes they tell you it's an alias (CNAME record)

where's best.cats.com

same place as cats.com

when an important DNS server dies

where's twitter.com?

8.8.8.8

xx

authoritative DNS server (dead)

I have no idea. it was at 172.2.3.9 but that DNS record expired and now the authoritative server is dead and AUGH

floating point

Julia Evans
@b0rk

more at: drawings.jvns.ca

a double is 64 bits.

that means there are 2^{64} different doubles

going up to 1.8×10^{308}

some double arithmetic

$2^{52} + 0.2 = 2^{52}$ (the next number after 2^{52} is $2^{52}+1$)

$1 + \frac{1}{2^{53}} = 1$ (the next number after 1 is $1+\frac{1}{2^{52}}$)

$3 \times 10^{100} =$ infinity ← infinity is a double

infinity - infinity = nan (not a number)

there are 2^{52} numbers between 1 and 2

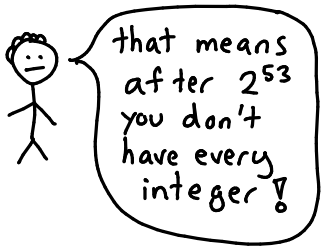
$1+\frac{1}{2^{52}}$, $1+\frac{2}{2^{52}}$, ...

2^{51} numbers between 2 and 4

2^{50} between 4 and 8

etcetera.

Javascript only has doubles (and Lua!)



printing doubles is nontrivial

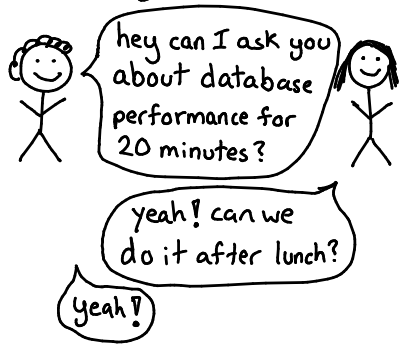
the shortest version of 25.648538980420e8 is 2.564854e9

↑ calculating the shortest representation takes time!

Julia Evans
@b0rk

asking good questions

find a good time

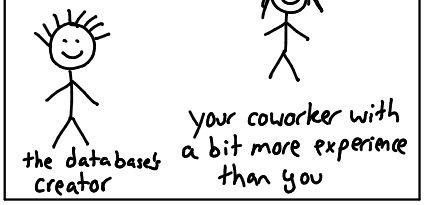


state what you know

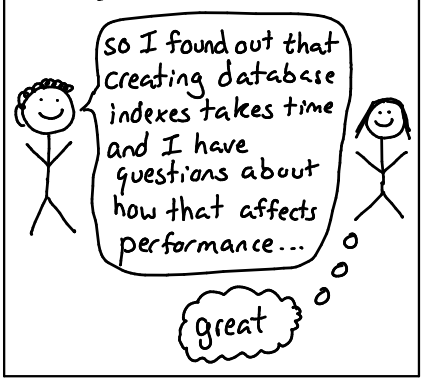
ask factual questions

choose who to ask

probably a better choice, has a good shot at answering your questions + way more time



do some research



profit

pipes

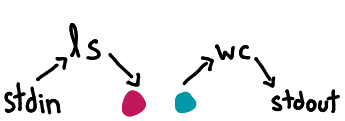
Julia Evans
@b0rk
drawings.jvns.ca

Sometimes you want to send the output of one process to the input of another

```
$ ls | wc -l
53
```

53 files!

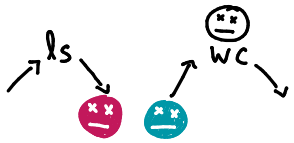
a pipe is a pair of 2 magical file descriptors



When ls does write(●, "hi")
wc can read it!
read(●)
-> "hi"

pipe buffers

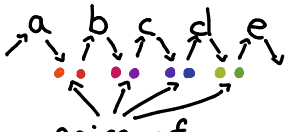
what if your target process dies?



ls gets sent SIGPIPE if ● gets closed (ls usually dies)

you can pipe SO MANY things together

```
$ a | b | c | d | e
```



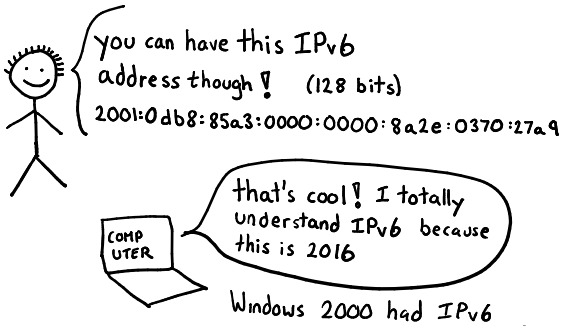
pairs of pipes

IPv6

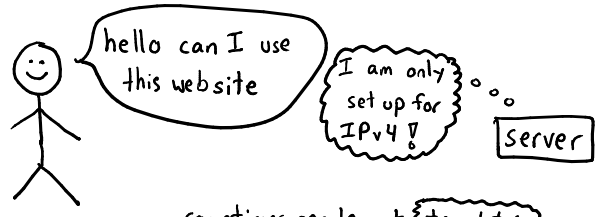
Julia Evans
@b0rk
drawings.jvns.ca

IPv4 addresses are 32 bits

What will we DO?!



Windows 2000 had IPv6 support. operating systems: SO READY



sometimes people put translators in the middle to turn IPv6 packets into IPv4 packets

adoption it's happening!!!

Google says 30% of American traffic they see is using IPv6

people were putting it off but we're REALLY RUNNING OUT of IPv4 addresses so now they have no choice

the "OSI model" for networking

Julia Evans
@b0rk

I don't always find it useful but it's good to know what "layer 4" means

what does "this is an L4 proxy" mean?

If a load balancer is labelled "L7" it usually means it looks at the Host: header inside your HTTP packets.

LAYERS

1: electrical engineering stuff, wires, frequencies, wifi

2: Ethernet protocol + others

3: IP (IP addresses)

4: TCP + UDP (ports)

5+6: nobody ever talks about these

7: HTTP and friends

layer 3 networking tool

↑ ignores layer 4 and above

I only know about IP addresses! I don't even know what a port is let alone what the packet says

unix permissions

Julia Evans
@b0rk

3 kinds of things you can do to a file

read write execute

```
$ ls -l awesome.png
rw-  rw-  r--  bork staff
```

- rw- : bork can do this (user)
- rw- : staff can do this (group)
- r-- : ANYONE can do this

```
$ ls -l /bin/ping
rwsr-xr-x root root
```

setuid flag

This means ping always runs as root (who owns it), no matter who started ping

what's this 755 business?

7	means	rwx
6	→	rw-
5	→	r-x
4	→	r--

it's binary!

5 → 101 → r-x

755 means rwx r-x r-x

more weird permissions things

setgid

sticky bit

but I ran out of space

Julia Evans
@b0rk

page table (in 32 bit memory)

every process has its own memory space

0xaeff3000

at that address it says "cat"

for me it says "dog"

process 1

process 2

each address maps to a 'real' address in physical RAM

process 1 → 0x28ea4000

0xaeff3000 → process 3 → invalid!

process 2 → 0x3942f000

processes have a "page table" in RAM that stores all their mappings

0x12345000 -> 0xae925...
0x23f49000 -> 0x12345...

the mappings are usually 4kB blocks (4kB is the normal size of a "page")

every* memory access uses the page table

I need to access 0xae923456

CPU

the page table says the real address is 0x99234456

*sort of

when you switch processes...

kernel: here, use this page table instead now

okay thanks! — CPU

some pages don't map to a physical RAM address

process: I'm going access 0x00040000

EEP NO! BAD ADDRESS! — CPU

= segmentation fault =

Julia Evans
@b0rk

what's a MAC address?

more at: drawings.jvns.ca!

every computer on the internet has a network card

hello! you can call me 0a:58:ff:ea:05:97

network card

MAC address

when you make HTTP requests with Ethernet/wifi, every packet gets sent to a MAC address

router: here is a cat for 0a:58:...

yay!

0a:58

wait, how do I know someone else on the same network isn't reading all my packets?

you don't! that's one reason we use HTTPS + secure wifi networks

your router has a table that maps IP addresses to MAC addresses

router: a message for 192.0.2.77? I will send that to 0a:58:ff:ea:05:97!

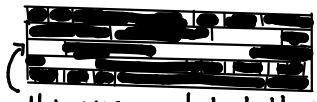
(read about ARP for more)

memory allocation

Julia Evans
@b0rk

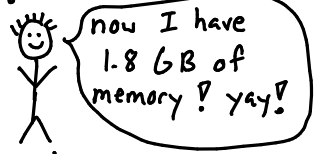
at any given time your program has a fixed amount of memory

587 MB ■ used □ free

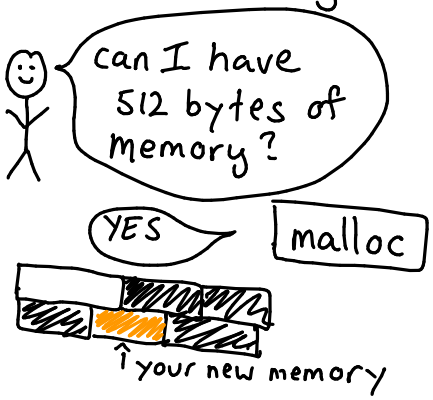


this was used but then got freed

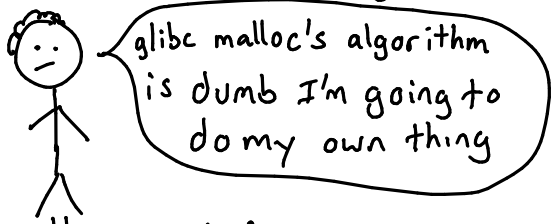
and it can ask the OS for more memory



your allocator tries to fill in unused pieces when you ask for memory

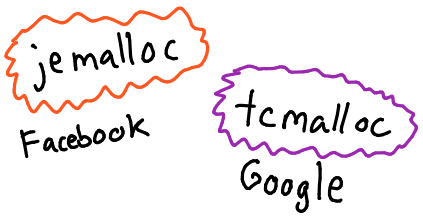


you can invent your own strategy to allocate memory



this is sort of normal to do if you care a LOT about performance

alternatives to libc malloc



Julia Evans
@b0rk

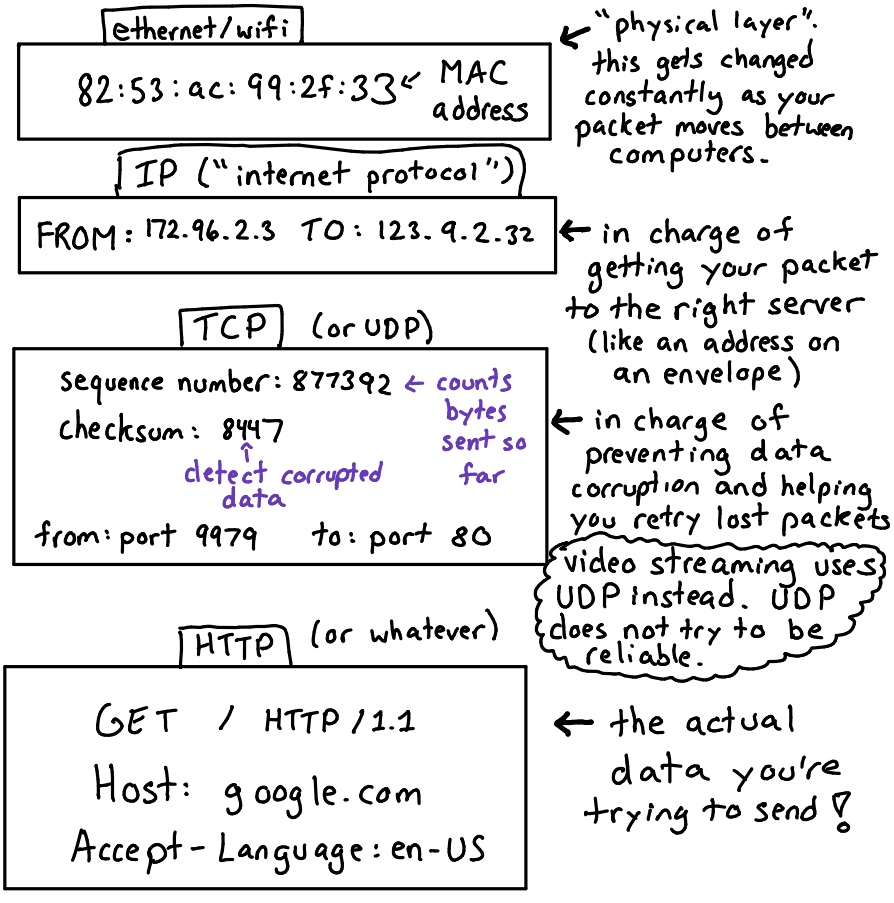
anatomy of a packet

more at drawings.jvns.ca

When you get a webpage like Facebook, it comes into your computer in many small packets

Let's see what those look like!

Packets are split into a few sections (or "headers")



networking concepts

Julia Evans
@b0rk

hey I want to understand all the networking stuff that happens when I go to google.com!

YES that is awesome. there are a lot of concepts but you can totally learn them all!

(knows many networking concepts now)

protocols

DNS
SSL/TLS
IP
TCP
UDP
ICMP (ping)
ARP
BGP
ethernet

other concepts

socket
packet
port
IP address
nameserver
NAT
router
TTL
checksum

+some more

it's a lot to learn but it's totally possible to learn how it all fits together to get you pictures of cats ♡

man pages = awesome

(sometimes. Quality may vary 😉)

Julia Evans
@b0rk

I found out I can get documentation for programs (like grep) with man grep!

but that's not all!! lots of other things have man pages too!

man pages are split up into 8 sections

① ② ③ ④ ⑤ ⑥ ⑦ ⑧

/usr/share/man/man5 has section 5 on my machine.

GREAT

① programs

$man grep
$man ls

② system calls

$man sendfile

③ C functions

$man 3 printf
$man fopen

④ devices

$man null
for /dev/null docs

⑤ file formats

$man sudoers
for /etc/sudoers
–> $man proc

⑥ games

(not very useful)
man sl is good if you have sl though

⑦ miscellaneous

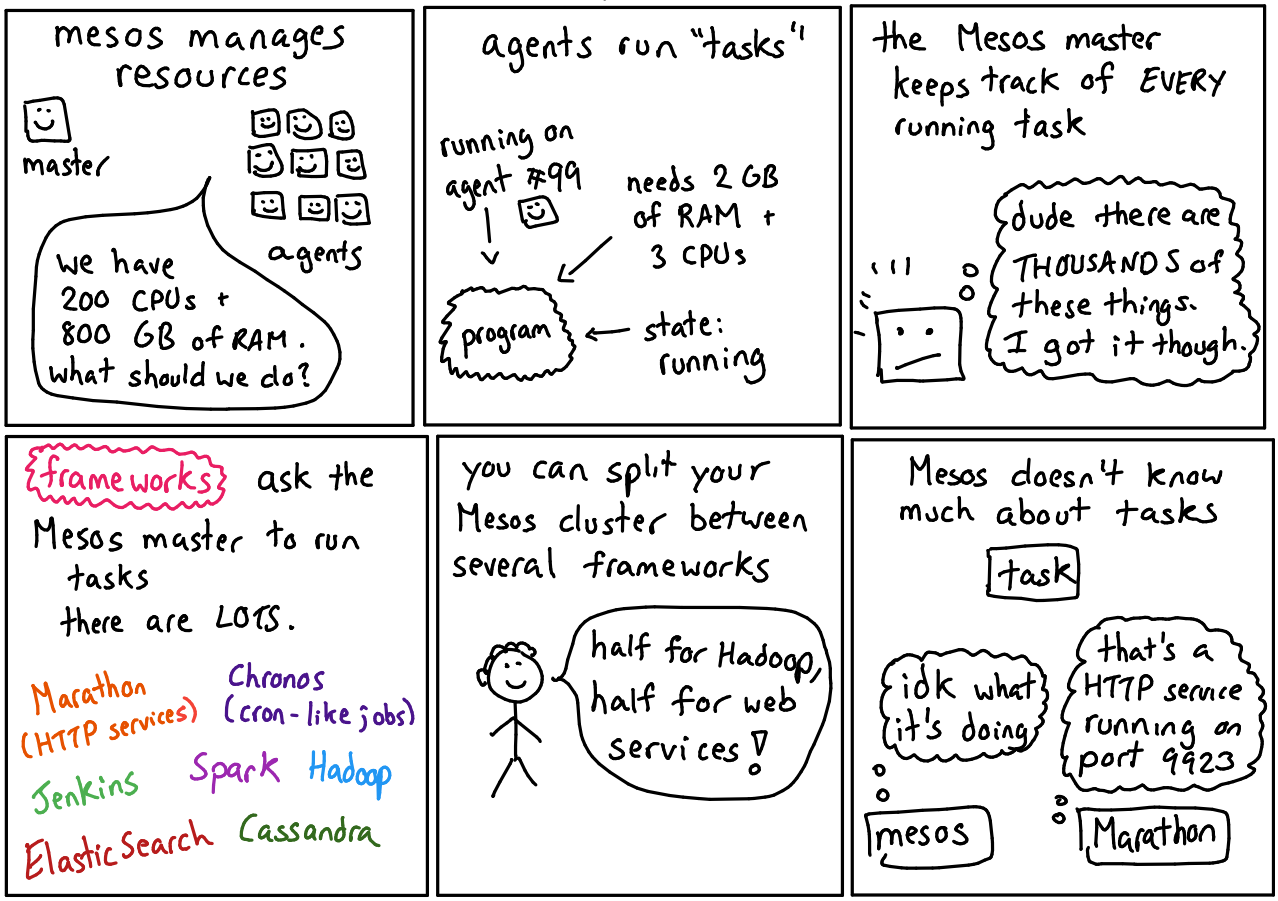
$man 7 pipe
$man 7 symlink
(these are cool!)

⑧ sysadmin programs

$man apt
$man chroot

mesos

Julia Evans
@b0rk



mutexes

drawings.jvns.ca

Julia Evans
@b0rk