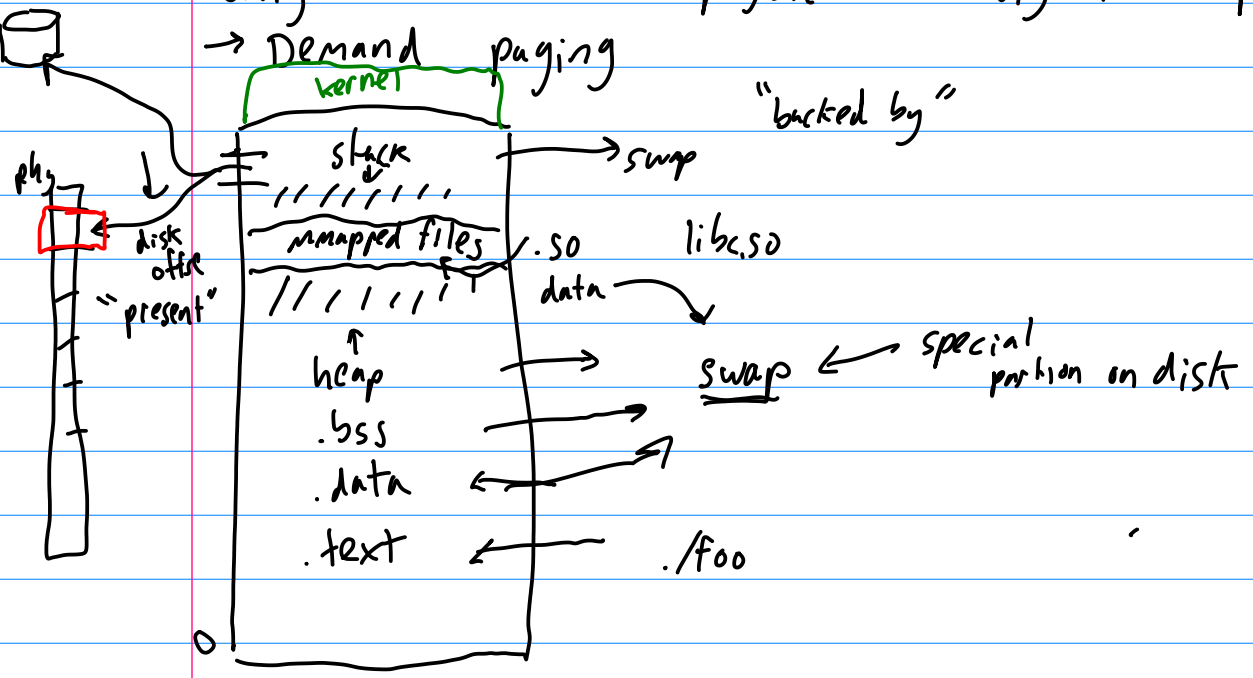
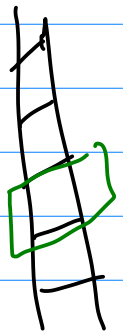
CS 3214 lecture #22 "virtual memory wrap up + Networking intro"

Virtual memory OOM killer

only hold data in physical memory that processes actually use
→ Demand paging



what happens if a page isn't mapped?
→ kernel space? s/w permission check fails
unmapped? SIGSEGV

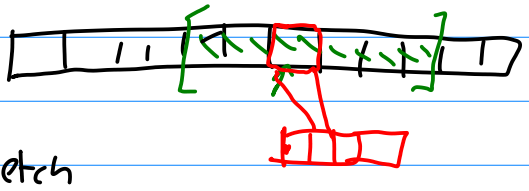


→ heap page? sbrk(big)
- alloc new page "lazy allocation" "minor fault"
- swap page from disk

→ code page?
- fetch from exe on disk

→ globals

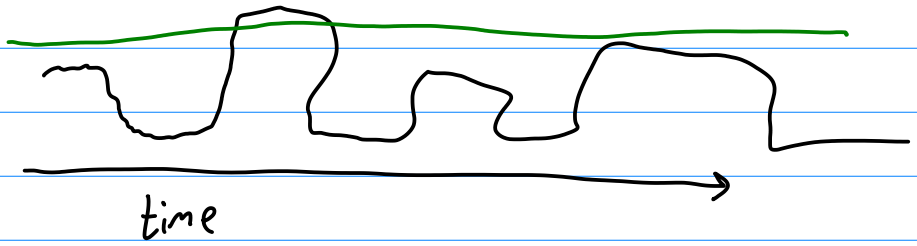
→ mmap MAP_POPULATE prefetch

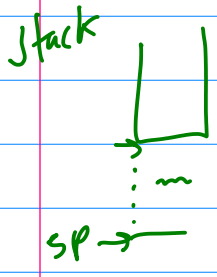


MEM ← virtual memory assigned / mapped

RSS ← present

working set





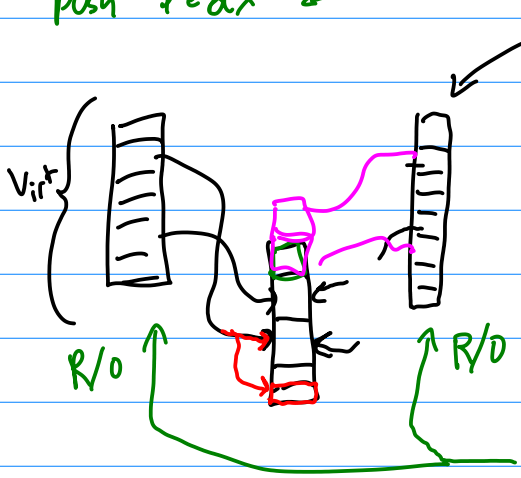
push %ebp

sub $20 %esp

push %eax ←

Laziness!

fork / exec



who gets the memory?

If we need to evict, who do we evict?

Page Replacement Algorithms / Policies

optimal: one used farthest in future

heuristics: LRU least recently used

- large looping
- approximate "access bits"

file data vs. process data

local vs. global

lots of chorce

Performance: $p * (\text{memory access time})$ ← 100ns

$+ (1-p) * (\text{page fault service time} + \text{memory access time})$ ← 10ms

99% hit 1000x slowdown

Thrashing: working set > phys mem

I/O

CPU util low

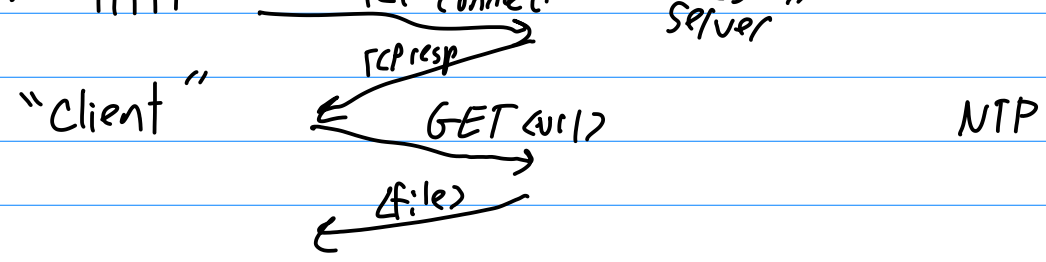
~ livelock

Networking: intro

Internet: // network of networks

network:
- hosts / edge devices
- routers / switches (network packets)
- communication links (fibers, copper, radio, etc.)

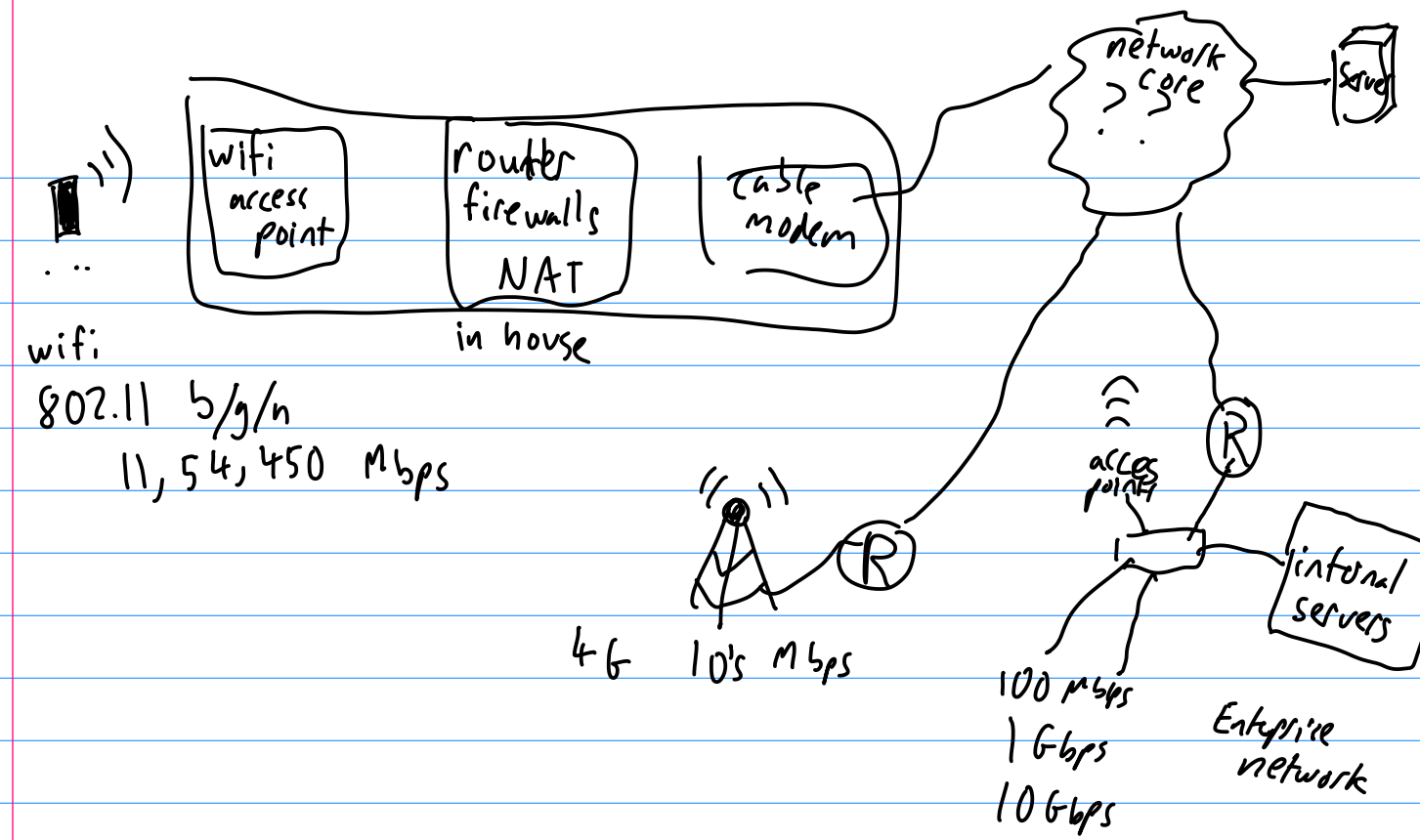
Protocols: HTTP



Client / servers

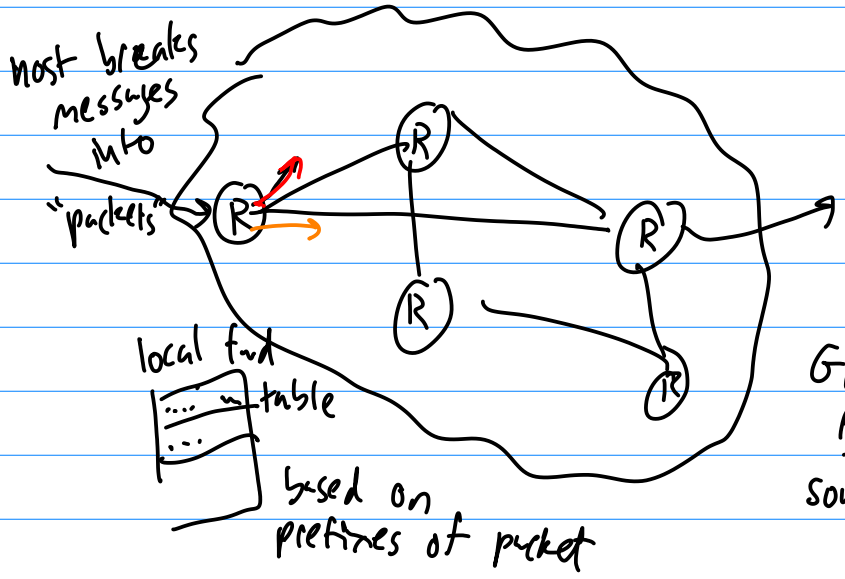
↑ edge nsks

↑ data center giving



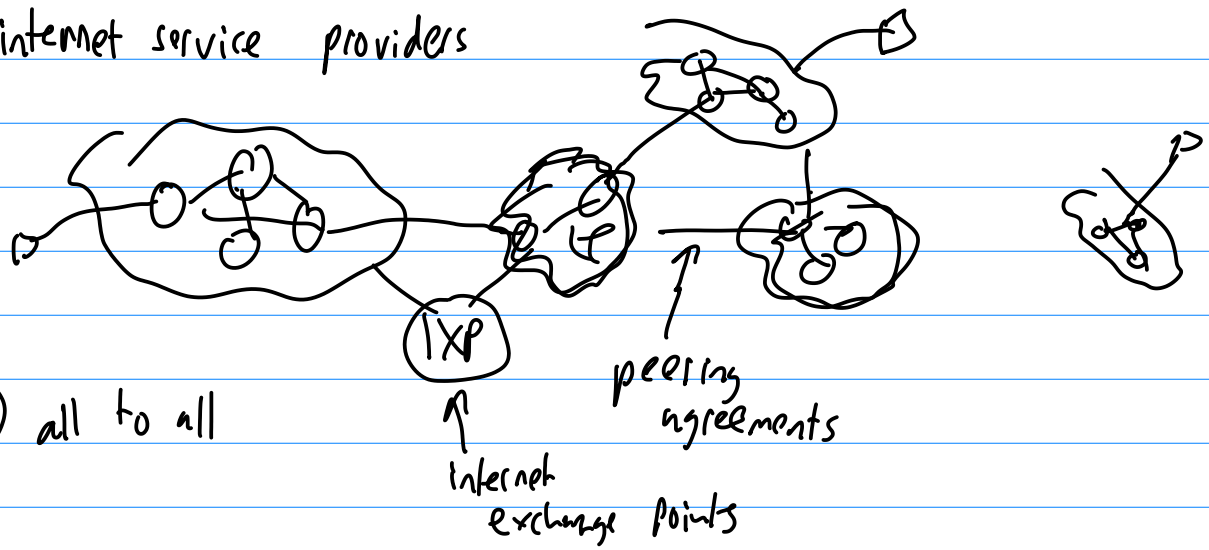
"network of networks"

mesh of interconnected routers

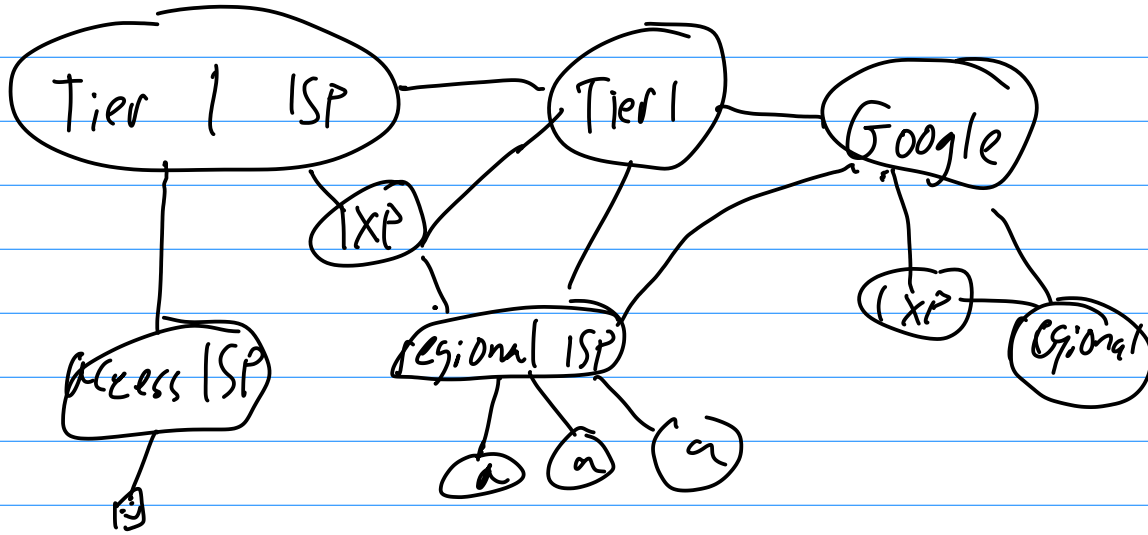


Global Routing
source/dest path
-routing algorithms

ISP internet service providers



$O(N^2)$ all to all


Tier 1 ISP
Tier 1
Google
IXP
IXP
Regional ISP
Access ISP