FIRST CLASS

3 components : 2 projects + presentations. All in groups

I) Impl. Project
- Educational hypervisor
- Rip out parts. Make you fix.
- Educational. How hypervisors work → core of virtual machines.
- End: fully working hypervisor.
- You will know how hypervisors work inside out
- contribute to industrial hypervisors.
  - Amazon Aws kernel
  - Microsoft Hyper-V.

→ Very hard & Intensive. Pace well.
→ 5 labs.
→ Each lab with deadline,
... previous lab.

- comfortable with C & somewhat assembly language.
- Will build on ...
- Googling.
- 50% of grade.
- LAB-0 DEADLINE: 24th January.

II) Open-source Project.

- Free-form project.
- Find repo. related to virtualization.
- Containers: Docker, Kubernetes, LXC, etc.
- Contribute to repository.
  - ↳ Issues
    - Github: Developers post issues & often tag them: help wanted, good first issue.
    - Go to issue, comment: students of UT, taking virt. course, wish to contribute.
    - This is what we have in mind, etc.
      - Easy → 3 easy.
      - Medium → 2 medium
      - Hard → 1 hard.
  - Not necessary to have code ... at the

merged; write a report ... end
- what you did
- Roadblocks
- Results.

If merged, extra credit.
- Any issue,
  - Just ; or Documentation
  - merged, extra credit of 0.5%
- Understand open-source contribution. Habit of PR, getting feedback, merging. Good for resume too
- Highly encourage.

- Each issue, cleared by me.
  ↳ Easy, medium, hard.
  Hard issue / medium issue, 1% extra credit.
  - How picky.
  - communication
- Start soon!
  - Don't wait until end.
- Deadline to get issues approved by me: Feb-23.
- Check-in with me.

↳ If issue seems easy at first but turns out difficult, let me know.

→ many times, many open-source repositories have a toxic culture, rude developers. Let me know.

→ Get it cleared from me.

- Normally people do well
  - work on issues of their interest
  - start early.
- Failure mode! start in April, & then struggle.
- 30% of Grade.

III) Presentations / Demos.

- Each group: Present a paper on Virtualization / Demo a system.
- Sheet is posted online, papers, can be claimed on FCFS basis.
- Demo: either what's mentioned, or something else of interest to you: reach out to me.
- 15 minutes, 2-3 mins questions.

- 1st class : Feb 9th. 20% of Grade.

Groups : Each Component:
Impl - project.
Open-source project +
Presentations.

Not required to have same group.

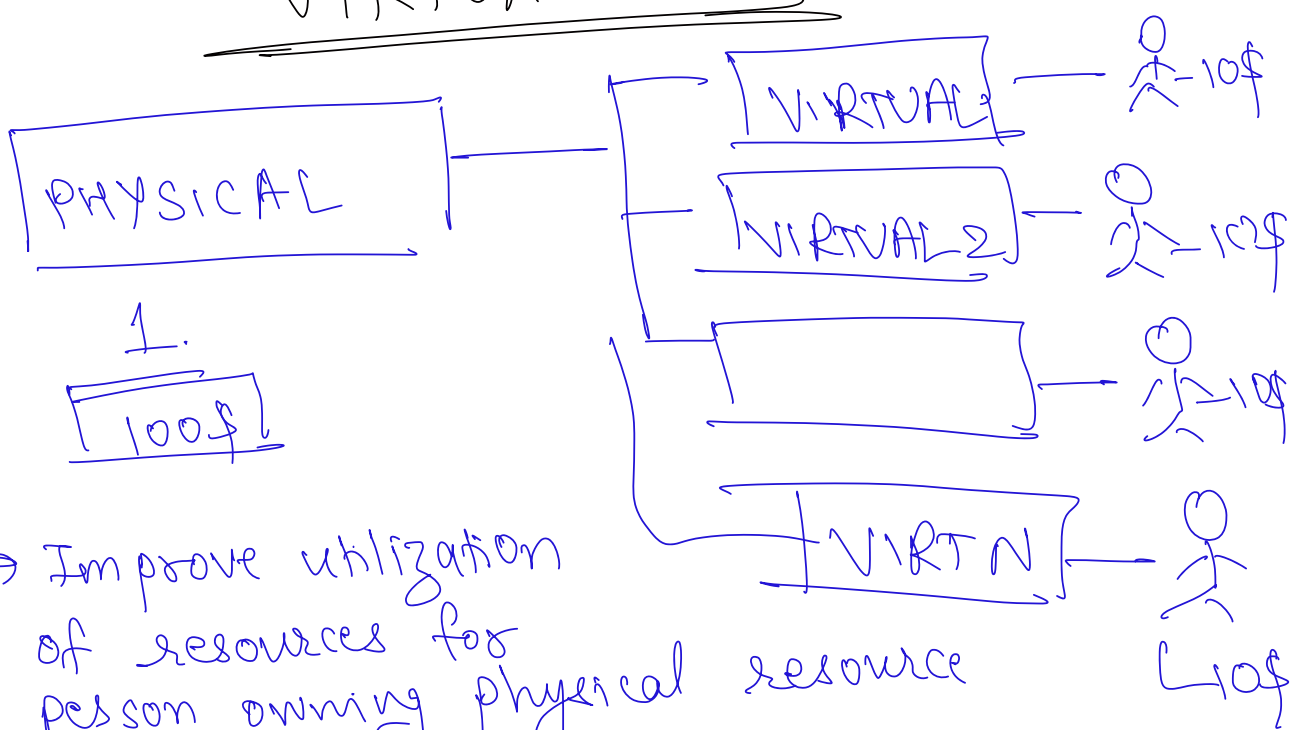
DO NOT CREATE YOUR OWN GROUPS ON CANVAS.
Already pre-created groups. just add yourself to pre-created groups
cannot grade you if you create own group.

TA: Hayley. PhD student with Vijay.
Hayley : mon 5-6 pm in GDC.
I : Thu 5-6 pm in GDC.
Expect to have many questions for Impl. Project. Please visit.

Lectures: Diff. from typical undergrad
First half - teaching
Second half: Hayley & I will present papers on Virtualization.

- Not a purely teaching course.
- Very fast-moving field: Nothing set in stone.
- Flavor of research & state-of-the-art.

VIRTUALIZATION



→ Improve utilization of resources for person owning physical resource

→ one computer
↳ Illusion of many small computers.

→ Physical case:
one user using resource.

→ Single user: pay for entire physical machine.
e.g. 1 person pays $100 for one hour

- Say, support 100 users.

VIRTUALIZE TO INCREASE UTILIZATION.

Economics.

one user: using 1% of resources wouldn't pay 100$/hr.
Divide into 100 resources, each person paying 10$/hr, make more money.
Resource fully utilized.

This is Amazon AWS
↳ Real world example
↳ Highly abstract.

CS Advantage : ASTRONOMY.

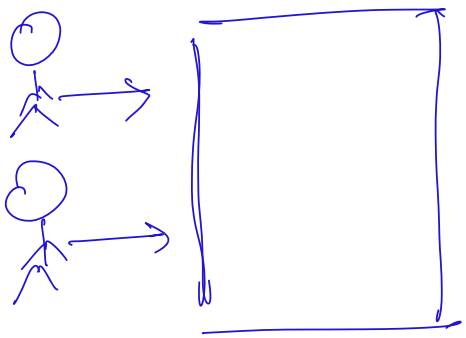
→ grads time-share resource.

→ time-sharing.

- 24 users.

TIME SHARE

VIRTUALIZATION



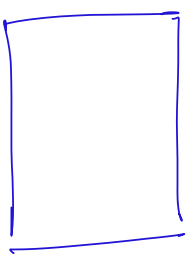
Key reason CS is busst to efficiency

- we can do research from home.

CHAMELEON.
TACC

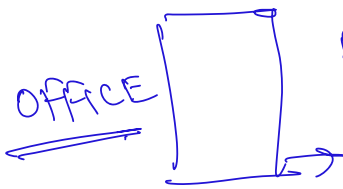
- we can access phy. resource via virtuali resources from home.

HISTORY



ENIAC Supercomputer.
(TIME SHARING) (1945)
→ submit jobs ↳ $500K
→ one-at-a-time ↳ equi. to. 6.2M$
→ submit results in punch cards

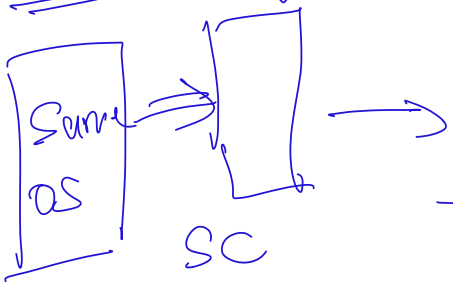
↳ Coarse-grained virtualization



PDP-11. Unix & C was built.

control time sharing

Expensive



– Computer labs of today.

No CPU.

multiple users at the same time

from users point of view: no time sharing

– Don't worry about anybody else.

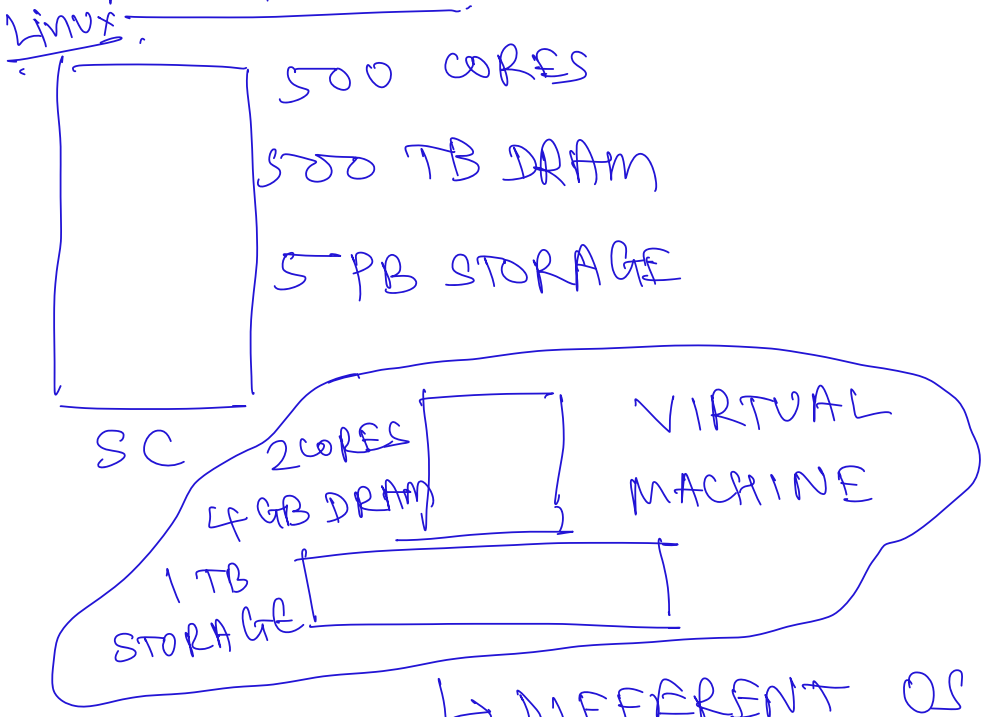
---

undergrad O.S.

– Virtualize CPU: time sharing
– Virtualize mem: page tables.
– Virtualize storage: home DIR.

- Someone crashes, everybody loses access.
- O.S. today. already support. virtualization.
- All running same software.

VIRTUAL MACHINES → ISOLATION



↳ DIFFERENT OS

↓

Windows

↳ Crash VM, SC doesn't go down.

VM were a revolution.

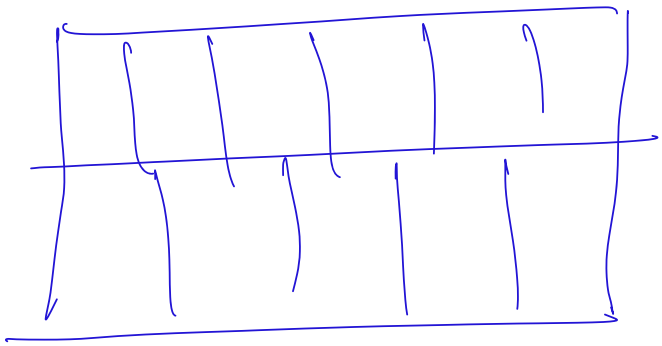
Previous: Accounts, Graphics, HR, etc. teams in a company

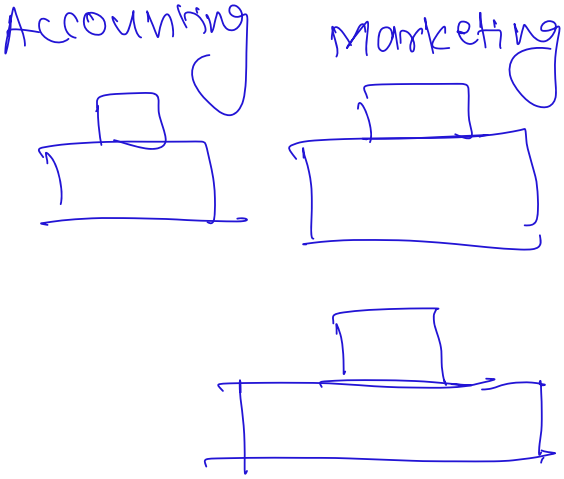
↳ Buy desktops for them & they used.

→ Expensive, underutilized.

Virtual machines came about.

Private companies.





Private cluster of beefy machines.

Early 2000s

↳ Spin-up VMs, Downsize.

↳ Size appropriately

→ Take beefy servers & turn them into fungible commodity.

Opportunities & challenges:

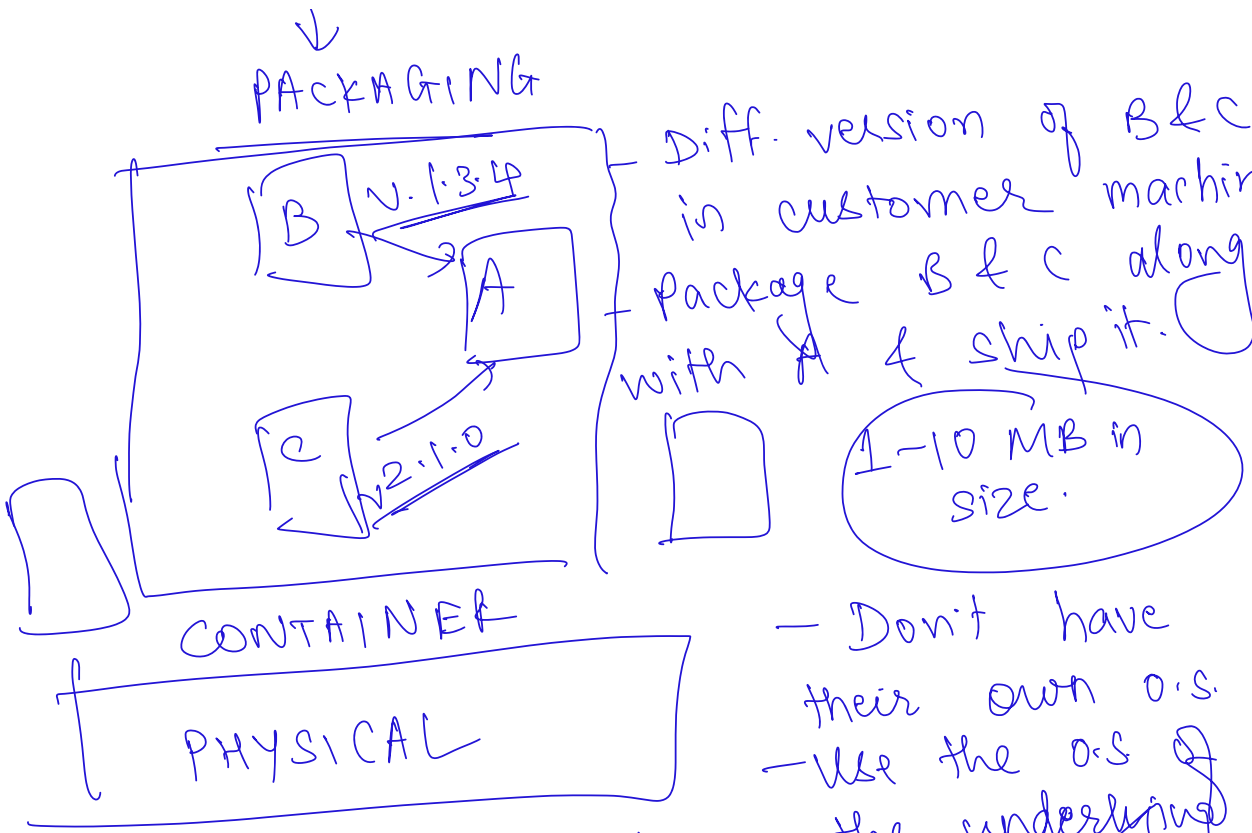
↳ update softwares, if people access it.

2005: Great but heavy.

↳ You need DRAM & Storage.

↳ Not great for all use cases.

SOFTWARE DEVELOPMENT



Diff. version of B & C in customer machine. Package B & C along with A & ship it.

1-10 MB in size.

- Don't have their own O.S.
- Use the O.S of the underlying machine.

DOCKER was born

Serverless → 2015.

→ Reduce granularity of virtualization

→ Function.

- You have a function, run it on Amazon & get result.

→ Don't care about container, VM.

→ Container on Amazon

↳ charged when container is running.

→ Serverless: charged when fn is executed.
→ Store sells stuff only for valentine's day.
↳ most of the year, charged. serverless
↳ more inactivity.

Limitations:
- stateless, global variables not allowed.
- use db. for example.
- not usable for stateful resources.

Fast moving field.

→ Every year, field has already moved
→ cutting edge
→ Build with impact, do research.
- You could have an insight for the next phase