Exam 2 Review Guide

Cluster Scheduling

- What is the architecture of a cluster scheduler? - See Lec 8, page 2
- What policies can the dispatcher use to assign queued requests or jobs? Lec 8, page 3.
- Understand various policies: round-robin, least loaded, weighted round robin, session-level, request level - what does each one do?

- SLURM and scheduling batch jobs

- Mesos - two level scheduler, see example of how it works lec 8, page 6
  - how Mesos allocates resources to frameworks using resource offers

- Borg - uses resource allocations (rather than resource offers)
  - Has the notion of preemption of running tasks for higher priority ones.

Virtualization

- What is the definition of virtualization?
- Using this definition - what does virtualization at different layers mean: hardware-level, OS-level, application-level

- Difference between emulation vs full virtualization
- Lec 8, page 14 - what is type 1? What is type 2?

- Lec 8, page 15 — how does type 1 works Why does it need sensitive instructions to cause a trap?
- Lec 8, page 19 - how does type 2 work? How does it handle sensitive instructions?
- Lec 8, page 20 - how paravirtualization works? Why does a guest OS need to be modified for paravirtualization? What are hyper calls? Why paravirtualized OS can run on older machines where sensitive instructions don't cause traps?
- How does memory virtualization works? Lec 8, page 22
- What is a virtual disk? A large file used to emulate a disk to a VM - Lec 8, page 23

- What are containers? How are they different from a VM? Lec 9, pages 3, 4.
- Benefits of containers over VMs - lec 9, page 4
- Disadvantage of containers over VMs - less isolation (e.g., if one containers causes an OS crash, all crash).
- Understand the following Linux container concepts:
  - (1) namespace (lec 9, page 6),
  - (2) cgroups (lec 9, page 7),

- (3) proportional share scheduling / weigted fair queuing (page 9-10) - how does it work?
- What is docker? How is it different from basic linux containers and what does it provide over containers? Lec9, Pages 12, 13,

Migration

- What is the difference between code migration and process migration?
- What is sender-initiated and receiver-initiated migration?
- Why do resources accessed by a process complicate the migration process?
- How should resources be handled when a process is migrated? See Lec 10, page 7
  - A resource can be moved, provide remote access, or substituted - when is each option viable - lec 10, page 8
- Why are local devices or communication endpoints / sockets hard to move? - Bound to a physical machine / fixed resources

- What is virtual machine migration?
- What is live VM migration?
- how does pre-copy migration work? How does post-copy migration work? (Lec 10, pages 11, 12, 13)
- Pros and cons of each approach
- How are network connections handled?
- Difference between cold, warm and hot migration - lec 10, page 15
- what is a snapshot? What is a virtual snapshot and what are its advantages? - What is copy on write?
- What is check point and restore (also known as suspend-resume) for container migration?
- Kubernetes— What is the concept of container orchestration? LEc 10, page 19
- What is a pod? How is different from a VM or a container? Lec 10, page 20
- How are applications mapped to pods? How are pods mapped to machines? LEc 10, page 20
- What is a Kubernetes service? - Lec 10, page 21
- What is a Kubernetes node and how is it managed by the control plane? High level concepts only - no implementation details are expected.

Cloud Computing

- What is a data center? Lec 11, page 4
- How does virtualization of servers simplify resource management in data centers? Lec 11, page 5, 6
- What is PUE? How does it capture the notion of efficiency? Lec 11, page 7
- Why are electricity costs high in a data center? Lec 11, page 7

- What is cloud computing and what are some benefits? - Lec 11, page 8
- Understand what IaaS, PaaS and SaaS clouds mean? Understand what public, private and hybrid clouds mean?
- Understand some advantages of each of these approaches
- What are the differences between IaaS and PaaS cloud platforms for deployment?

- Cloud storage - what is object storage vs file storage? LEc 11 page 15

Clock Synchronization

- what is a global clock
- why do clocks of machines need a distributed system need to be synchronized
- what is clock drift and max drift rate? How often do clocks need to be synchronized?
- what is Christian's algorithm and the Berkeley algorithm?
- why do we need to estimate round trip times in clock synchronization?
- Why does NTP prevent clocks from going backwards? what can go wrong if clocks go back?
- what is the general concept of GPS? why do we need to estimate the receiver's clock drift in order to estimate its coordinates?
- What is a logical clock (aka Lamport's clock)?
- What is happened before relationship? When is the happened before relationship undefined?
- How does the Lamport clock algorithm work? To understand it well, we suggest working with some toy examples and assigning clocks values as per the algorithm. Also see figures on Lec 12, page 18.
- Why is it not possible to compare two logical clock values to determine event ordering?
- How do vector clocks work? Use some toy examples to understand why vector clocks allow time stamps to be compared to infer event ordering.

Distributed Snapshots

- What is a distributed snapshot? Why are they useful?
- What is the global state of a system? What is a consistent global state? What is a consistent cut and how is it related to consistent global state?
- How does the distributed snapshot algorithm work? How do marker messages allow us to capture messages in transit?
- What is termination detection? Why is it useful?

Leader Election

- What is leader election and why is it useful in a distributed system?
- What is the Bully algorithm? What is its message overhead? What happens if a crashed leader recovers after a new leader with a lower process ID has been elected?
- What is ring based election? What is its message overhead? How can ring and bully handle multiple elections in progress?

Distributed synchronization

- what are some examples where we need a distributed lock rather than a centralized one?

- How does centralized mutual exclusion work? What are its properties/advantages/ disadvantages (Lec 15, pg 14).
- How does the decentralized mutual exclusion work using voting? How does it handle crash of a coordinator?
- How does Ricart and Agarwala algorithm work? Why does it have worse failure properties than a centralized locking algorithm?
- How does token ring algorithm work from a locking perspective?
- There is no need to go into complexity details on Lec 15, page 19, since it was not covered in class.
- What is Google's Chubby locking service ? Only basic concepts expected, no need to go into detail of Chubby.