TCSS 462/562: (SOFTWARE ENGINEERING FOR) CLOUD COMPUTING

Containerization II

Wes J. Lloyd
School of Engineering and Technology
University of Washington – Tacoma

1

OFFICE HOURS – FALL 2025

- **Tuesday:**
  - 6:00 to 7:00 pm - CP 229 & Zoom
- Or email for appointment

- *Office Hours set based on Student Demographics survey feedback*
- ** - Friday office hours may be adjusted or canceled due meeting conflicts or other obligations. Adjustments will be announced via Canvas.*

2

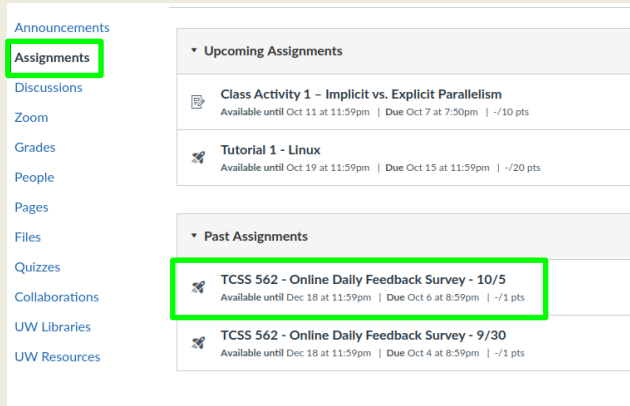
OBJECTIVES – 11/25

- **Questions from 11/20**
- Tutorials Questions
- Class Presentations Schedule - Cloud Technology or Research Paper Review
- Tutorial 8: Serverless Beyond Java, Container-Based Functions
- Containerization
- Kubernetes

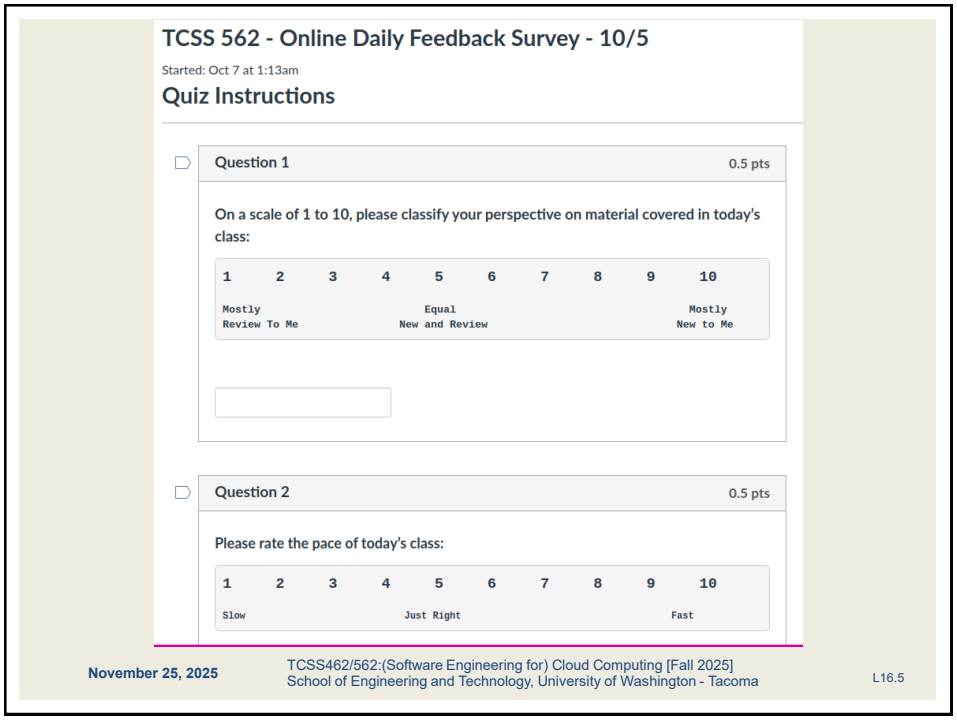
3

ONLINE DAILY FEEDBACK SURVEY

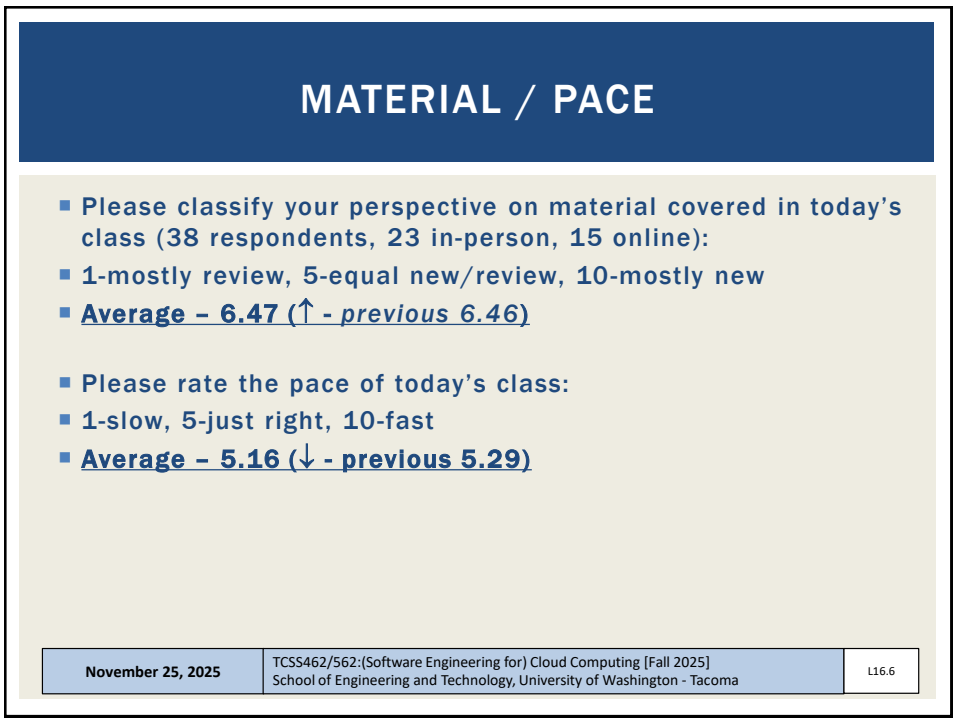
- Daily Feedback Quiz in Canvas – Take After Each Class
- Extra Credit for completing



4


TCSS 562 - Online Daily Feedback Survey - 10/5
Started: Oct 7 at 1:13am
Quiz Instructions
Question 1
0.5 pts
On a scale of 1 to 10, please classify your perspective on material covered in today's class:
1 2 3 4 5 6 7 8 9 10
Mostly Review To Me
Equal New and Review
Mostly New to Me
Question 2
0.5 pts
Please rate the pace of today's class:
1 2 3 4 5 6 7 8 9 10
Slow
Just Right
Fast
November 25, 2025
TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025]
School of Engineering and Technology, University of Washington - Tacoma
L16.5

5


MATERIAL / PACE
Please classify your perspective on material covered in today's class (38 respondents, 23 in-person, 15 online):
1-mostly review, 5-equal new/review, 10-mostly new
Average – 6.47 (↑ - previous 6.46)
Please rate the pace of today's class:
1-slow, 5-just right, 10-fast
Average – 5.16 (↓ - previous 5.29)
November 25, 2025
TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025]
School of Engineering and Technology, University of Washington - Tacoma
L16.6

6

FEEDBACK FROM 11/20

- ***Is the execution of the software in a container affected by the performance of the PC? If the PC runs slow does the SW in the container run slow as well?***
- YES - A container is just a collection of Linux processes (managed as a cgroup)
- The software processes run on Linux and compete for resources along with everything else
- If the host is slow / overloaded, this will impact the container(s)
- Ideally a Docker host runs nothing except containers, and then resource quotas can restrict CPU & memory resources for different containers (cgroups)

7

FEEDBACK - 2

- ***In this last class, you had said that Google put a lot of effort into Kubernetes - but why then did they give it away? Is it not a good technology anymore?***
- The motivation to open source Kubernetes and provide it for free is to encourage everyone to architect applications for Kubernetes
- Once cloud applications are bundled for Kubernetes, it makes them portable to any cloud provider with Kubernetes services
- The idea is to encourage more cloud users to migrate away from AWS and run their Kubernetes apps on Google Cloud
- This also had the affect of helping Azure gain Kubernetes workloads
- Kubernetes apps are called 'cloud native applications'. They don't use vendor specific services
- For object storage, for example, you bundle and deploy minio, your own object storage platform (https://min.io)

8

PRACTICE QUESTIONS

- ***In the public cloud, why is it advantageous for containers to be run on top of VMs?***
- ***On a private server, why is it advantageous for containers to be run on top of bare metal?***

9

OBJECTIVES – 11/25

- Questions from 11/20
- **Tutorials Questions**
- Class Presentations Schedule - Cloud Technology or Research Paper Review
- Tutorial 8: Serverless Beyond Java, Container-Based Functions
- Containerization
- Kubernetes

10

TUTORIAL 6 – NOV 23

- Introduction to Lambda III: Serverless Databases
- https://faculty.washington.edu/wlloyd/courses/tcss562/tutorials/TCSS462_562_f2025_tutorial_6.pdf
- Create and use Sqlite databases using sqlite3
- Deploy Lambda function with Sqlite3 database under /tmp
- Compare in-memory vs. file-based Sqlite DBs on Lambda
- Create an Amazon Aurora "Serverless" v2 MySQL database
- Using the AWS CloudShell in the same VPC (Region + availability zone) connect and interact your Aurora serverless database using the mysql CLI app
- Deploy an AWS Lambda function that uses the MySQL "serverless" database
- **'FREE PLAN': okay to use db.t3.micro, db.t4g.micro RDS MySQL VM – must indicate use of RDS VM on tutorial PDF**

11

TUTORIAL 7 – DEC 4

- Introduction to Docker
- https://faculty.washington.edu/wlloyd/courses/tcss562/tutorials/TCSS462_562_f2025_tutorial_7.pdf
- Must complete using c7i-flex.large ec2 instance & Ubuntu 24.04 (for cgroups v2)
- Use DOCX file for copying and pasting Docker install commands
- Topics:
  - Installing Docker
  - Creating a container using a Dockerfile
  - Using cgroups virtual filesystem to monitor CPU utilization of a container
  - Persisting container images to Docker Hub image repository
  - Container vertical scaling of CPU/memory resources
  - Testing container CPU and memory isolation

12

OBJECTIVES – 11/25

- Questions from 11/20
- Tutorials Questions
- **Class Presentations Schedule - Cloud Technology or Research Paper Review**
- Tutorial 8: Serverless Beyond Java, Container-Based Functions
- Containerization
- Kubernetes

GROUP PRESENTATIONS

- TWO OPTIONS:
- ***Cloud technology presentation***
- ***Cloud research paper presentation***
  - Recent & suggested papers will be posted at: http://faculty.washington.edu/wlloyd/courses/tcss562/papers/
- **Presentation dates:**
  - Tuesday November 25
  - Tuesday December 2, Thursday December 4
- **Peer Reviews**
  - Word DOCX review form posted, fill out, submit PDF on Canvas
  - Feedback shared with groups
  - TCSS 462: submit 4 total peer reviews in lieu of a group presentation

GROUP PRESENTATIONS

- 7 Presentation Teams
- 3 Cloud Technology Talks
- 4 Cloud Research Paper Presentations
- 1 one-person teams
- 2 two-person teams
- 4 three-person teams

- Thank you for the submissions

15

PRESENTATION SCHEDULE

- **<Tuesday November 25>**

1. Team 4: Xiaoling Wei, Bohan Xiong, Xu Zhu
Research paper: ***Serverless Replication of Object Storage across Multi-Vendor Clouds and Regions***
2. Team 1: William Hay
Cloud Technology: ***Amazon Athena***
3. *Robert Cordingly – Original Research Paper:* ***Sky Computing for Serverless: Infrastructure Assessment to Support Performance Enhancement*** *(IEEE/ACM UCC 2025 Practice Talk)*

- **<Tuesday December 2>**

1. Team 5: Sparsha Jha, Chris Biju
Cloud Technology: ***Intelligent Optimization of Distributed Pipeline Execution in Serverless Platforms: A Predictive Model Approach***

16

PRESENTATION SCHEDULE - 2

- **<Thursday December 4>**

1. Team 3: Jiameng Li, Naomi Nottingham, Headley Brissett
Research paper: ***A Perfect Fit? – Towards Containers on Microkernels***
2. Team 2: Ruby Plangphatthanaphanit, Junjia Li, Ari Yin
Cloud Technology: ***CI/CD in the Cloud (GitHub Actions + Cloud Deploy)***
3. Team 8: Aamena Suzzane, Dhruva Bhat
Research paper: ***CoFaaS: Automatic Transformation-based Consolidation of Serverless Functions***
4. Team 6: Han Zhang, Sahil Bhatt, Pengcheng Cao
Cloud Technology: ***AWS Amplify***

17

OBJECTIVES – 11/25

- Questions from 11/20
- Tutorials Questions
- Class Presentations Schedule - Cloud Technology or Research Paper Review
- **Tutorial 8: Serverless Beyond Java, Container-Based Functions**
- Containerization
- Kubernetes

18

TUTORIAL 8 – TO BE POSTED

- **Python Based AWS Lambda Functions w/ SAAF, and Container-Based AWS Lambda Functions**
- Not Required, available for extra credit (scored out of 0)
  - adds points to overall tutorials score
  - 10 pts for Python Functions / 15 pts for Container Based Function
- Tasks
  - Build/Deploy/Test Python-based Lambda Functions
  - Deploy and Test Container Based AWS Lambda Function
    - Requires Docker Engine installation on local VM
  - Create role to use CLI/publish script
  - Use a config file to specify container-based function details
  - Update bash script to deploy hello function
  - Build Docker container locally, Publish to Elastic Container Registry
  - Create new 'hello' Lambda Function based on Container image
  - Test Container-based 'hello' AWS Lambda Function
  - Adapt your function to run sysbench prime number generation
  - Test prime number generation performance on AWS Lambda vs. memory

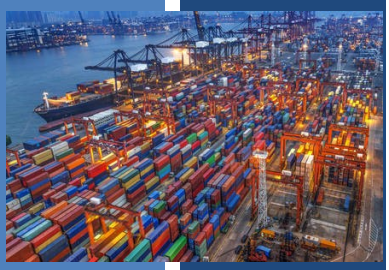
TUTORIAL 9 – TO BE POSTED

- **Introduction to AWS Step Functions and Amazon Simple Queue Service (SQS)**
- Not Required, available for extra credit (scored out of 0)
  - adds points to overall tutorials score
- Tasks
  - Adapt Caesar Cipher Lambda functions for use with AWS Step Functions
  - Create AWS Step Functions State Machine
  - Create a BASH client to invoke the AWS Step Function
  - Create Simple Queue Service Queue for messages
  - Add message to SQS queue from AWS Lambda function
  - Modify AWS Step Function Bash client script to retrieve AWS Step Function result from SQS queue

OBJECTIVES – 11/25

- Questions from 11/20
- Tutorials Questions
- Class Presentations Schedule - Cloud Technology or Research Paper Review
- Tutorial 8: Serverless Beyond Java, Container-Based Functions
- **Containerization**
- Kubernetes

21

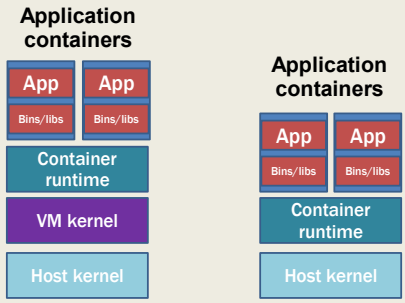


CONTAINERIZATION

22

CONTAINER ISOLATION

- Is the host isolated from application containers?
- Are application containers isolated from each other?



23

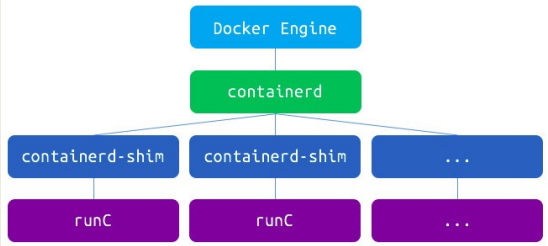
LXC (LINUX CONTAINERS)

- Operating system level virtualization
- Run multiple isolated Linux systems on a host using a single Linux kernel
- Control groups(cgroups)
  - Including in Linux kernels => 2.6.24
  - Limit and prioritize sharing of CPU, memory, block/network I/O
- Linux namespaces
- Docker initially based on LXC

24

OTHER DOCKER TOOLS

- **Docker Machine:** automatically provision and manage sets of docker hosts to form a cluster



- **Docker Swarm**: Clusters multiple docker hosts together to manage as a cluster.
- **Docker Compose**: Config file (YAML) for multi-container application; Describes how to deploy and configure multiple containers

25

CONTAINER ORCHESTRATION FRAMEWORKS

- Framework(s) to deploy multiple containers
- Provide container clusters using cloud VMs
- Similar to "private clusters"
- Reduce VM idle CPU time in public clouds
- Better leverage "sunk cost" resources
- Compact multiple apps onto shared public cloud infrastructure
- Generate to cost savings
- Reduce vendor lock-in

26

KEY ORCHESTRATION FEATURES

- Management of container hosts
- Launching set of containers
- Rescheduling failed containers
- Linking containers to support workflows
- Providing connectivity to clients outside the container cluster
- Firewall: control network/port accessibility
- Dynamic scaling of containers: horizontal scaling
  - Scale in/out, add/remove containers
- Load balancing over groups of containers
- Rolling upgrades of containers for application

27

CONTAINER ORCHESTRATION FRAMEWORKS - 2

- Docker swarm
- Apache mesos/marathon
- Kubernetes
  - Many public cloud provides moving to offer Kubernetes-as-a-service
- Amazon elastic container service (ECS)
- Apache aurora

- Container-as-a-Service
  - Serverles containers without managing clusters
  - Azure Container Instances, AWS Fargate…

28

OBJECTIVES – 11/25

- Questions from 11/20
- Tutorials Questions
- Class Presentations Schedule - Cloud Technology or Research Paper Review
- Tutorial 8: Serverless Beyond Java, Container-Based Functions
- Containerization
- **Kubernetes**

29

KUBERNETES

from: "The Kubernetes Book", Nigel Poulton and Pushkar Joglekar, Version 7.0, September 2020

30

KUBERNETES

- Name is from the Greek word meaning Helmsman
  - The person who steers a seafaring ship
  - The logo reinforces this theme
- Kubernetes is also sometimes called K8s
- Kubernetes is an application orchestrator

- Most common use case is to containerize cloud-native microservices applications
- What is an orchestrator?
  - System that deploys and manages applications

31

KUBERNETES – 2

Why does Google want to give Kubernetes away for free?

- Initially developed by Google
- **Goal:** *make it easier for potential customers to use Google Cloud*
- Kubernetes leverages knowledge gained from two internal container management systems developed at Google
  - Borg and Omega
- Google donated Kubernetes to the Cloud Native Computing Foundation in 2014 as an open-source project
- Kubernetes is written in Go (Golang)
- Kubernetes is available under the Apache 2.0 license
- Releases were previously maintained for only 8 months!
- Starting w/ v 1.19 (released Aug 2020) support is 1 year

32

GOALS OF KUBERNETES

1. Deploy your application
2. Scale it up and down dynamically according to demand
3. Self-heal it when things break
4. Perform zero-downtime rolling updates and rollbacks

- These features represent automatic infrastructure management

- Containerized applications run in container(s)
- Compared to VMs, containers are thought of as being:
  - Faster
  - More light-weight
  - More suited to rapidly evolving software requirements

33

CLOUD NATIVE APPLICATIONS

- Applications designed to meet modern software requirements including:
  - **Auto-scaling:** resources to meet demand
  - **Self-healing**: *required for high availability (HA) and fault tolerance*
  - **Rolling software updates**: with no application downtime for DevOPS
  - **Portability**: can run anywhere there's a Kubernetes cluster

34

WHAT IS A MICROSERVICES APP?

- Application consisting of many specialized parts that communicate and form a meaningful application
- Example components of a microservice eCommerce app:
  - Web front-end
  - Shopping cart
  - Logging service
  - Catalog service
  - Authentication service
  - Persistent data store
- **KEY IDEAS:**
- Each microservice can be coded/maintained by different team
- Each has its own release cadence
- Each is deployed/scaled separately
- Can patch & scale the log service w/o impacting others

KUBERNETES - 3

- Provides "an operating system for the cloud"
- Offers the de-facto standard platform for deploying and managing **cloud-native applications**
- OS: abstracts physical server, schedules processes
- Kubernetes: ***abstracts the cloud***, schedules microservices
- Kubernetes abstracts differences between private and public clouds
- Enable cloud-native applications to be **cloud agnostic**
  - i.e. they don't care *WHAT* cloud they run on
  - Enables fluid application migration between clouds
- Kubernetes provides rich set of tools/APIs to introspect (observe and examine) your apps

KUBERNETES - 4

- Features:
- A "control plane" – brain of the cluster
  - Implements autoscaling, rolling updates w/o downtime, self-healing
- A "bunch of nodes" – workers (muscle) of the cluster

- Provides orchestration
  - The process of organizing everything into a useful application
  - And also the goal of keeping it running smoothly

37

KUBERNETES - CLUSTER MANAGEMENT

- Master node(s) manage the cluster by:
  - Making scheduling decisions
  - Performing monitoring
  - Implementing changes
  - Responding to events
- *Masters implement the control plane of a Kubernetes cluster*

- Recipe for deploying to Kubernetes:
- Write app as independent microservices in preferred language
- Package each microservice in a container
- Create a manifest to encapsulate the definition of a **Pod**
- Deploy Pods to the cluster w/ a higher-level controller such as "Deployments" or "DaemonSets"

38

PODS

- Pod – atomic unit of deployment & scheduling in Kubernetes
- A Kubernetes Pod is defined to run a containerized application
- Kubernetes manages Pods, not individual containers
- Cannot run a container directly on Kubernetes
- All containers run through Pods

- Pod comes from "pod of whales"
- Docker logo shows a whale with containers stacked on top
- Whale represents the Docker engine that runs on a single host
- **Pods encapsulate the definition of a single microservice for hosting purposes**
- **Pods can have a single container, or multiple containers, if the service requires more than one**

39

DECLARATIVE SERVICE APPROACH

- **Imperative definition**: sets of commands and operations
  - Example: BASH script, Dockerfile
- **Declarative definition**: specification of a service's properties
  - What level of service it should sustain, etc.
  - Example: Kubernetes YAML files

- Kubernetes manages resources **declaratively**
- How apps are deployed and run are defined with YAML files
- YAML files are POSTed to Kubernetes endpoints
- Kubernetes deploys and manages applications based on declarative service requirements
- If something isn't as it should be: *Kubernetes automatically tries to fix it*

40

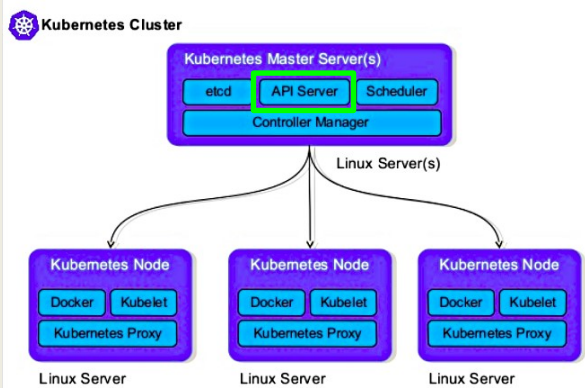
KUBERNETES MASTER

- Provide system services to host the control plane
- Simplest clusters use only 1 master – (i.e. no replication)
  - Suitable for lab and dev/test environments
- Production environments: masters are replicated ~3-5x
  - Provides fault tolerance and high availability (HA)
  - Cloud-based managed Kubernetes services offer HA deployments

41

MASTER SERVICES

- API Server
- Cluster store
- Controller Manager
- Scheduler
- Cloud controller



42