TCSS 562: SOFTWARE ENGINEERING FOR CLOUD COMPUTING

Cloud Computing Concepts and Models

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

1

OFFICE HOURS – FALL 2025

- **Thursdays:**
  - 6:00 to 7:00 pm - CP 229 & Zoom
- **Fridays**
  - 11:00 am to 12:00 pm – ONLINE via 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

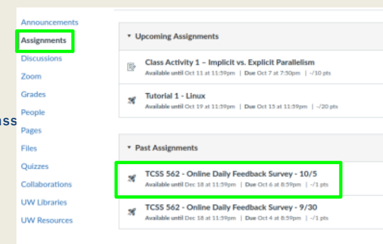
OBJECTIVES – 10/21

- **Questions from 10/16**
- Tutorials Questions
- Tutorial 4 – Intro to FaaS – AWS Lambda
- Background on AWS Lambda for the Term Project - II
- **From: Cloud Computing Concepts, Technology & Architecture: Chapter 4: Cloud Computing Concepts and Models:**
  - Roles and boundaries
  - Cloud characteristics
  - Cloud delivery models
  - Cloud deployment models
- Team Planning - Breakout Rooms

3

ONLINE DAILY FEEDBACK SURVEY

- Daily Feedback Quiz in Canvas – Take After Each Class
- **1-point Extra Credit** for completing online
- **2-points Extra Credit** for completing in-person in class
- 36 points possible
- 2.5% added to final course grade for (36/36)



4

WARNING

- DO NOT SUBMIT BOTH A PAPER AND AN ONLINE SURVEY OR YOU WILL LOOSE POINTS
- CANVAS WILL AUTOMATICALLY REPLACE THE PAPER SURVEY SCORE (2 PTS) WITH THE ONLINE SURVEY (1 PT)
- *** COMPLETE ONLY ONE SURVEY FOR EACH CLASS SESSION ***
- WE WILL NOT BE ABLE TO DUPLICATE CHECK SURVEYS FOR EACH CLASS SESSION AND MAKE CORRECTIONS

5

MATERIAL / PACE

- Please classify your perspective on material covered in today's class (43 respondents, 25 in-person, 18 online):
- 1-mostly review, 5-equal new/review, 10-mostly new
- **Average – 7.00 (↑ - *previous 6.91*)**
- Please rate the pace of today's class:
- 1-slow, 5-just right, 10-fast
- **Average – 5.44 (↑ - previous 5.14)**

6

FEEDBACK FROM 10/21

- ***What's the difference between AWS Lambda's function instance "mini-VMs" and regular VM?***
- AWS Lambda uses "Firecracker" micro-VMs to host serverless workloads such as functions and containers
  - Firecracker is the hyper-visor (also called a virtual-machine monitor) used to create Firecracker microVMs
  - Suggested Reading - Paper: https://www.usenix.org/conference/nsdi20/presentation/agache
- Regular VMs, such as those available from Amazon EC2 are full-featured, general purpose virtual servers
  - 5th generation and beyond use AWS Nitro nearly full HW virtualization:
  - Suggested Reading:
  - https://docs.aws.amazon.com/whitepapers/latest/security-design-of-aws-nitro-system/traditional-virtualization-primer.html

7

MICROVMS VS. VMS

Firecracker Micro VMs on AWS	EC2 VM
No direct connections allowed. The guest OS can only be accessed via a serverless function or container code	Users can directly connect using SSH for console sessions to interact with the guest OS
Micro VMs run only Amazon Linux 2023	VMs run any OS selected by the user
Boot time ~125 ms	Boot time 3-8+ sec, (more w/ special OS or software)
Based on Firecracker which is based on Google crosvm which is based on KVM	Full-featured VMs based on AWS Nitro hypervisor based on KVM (Linux Kernel Virtual Machine)
Massively stripped down virtual computer w/ simplified device model: no BIOS, no PCI	
Internal hidden IP addresses	Public/private IP addresses

8

FEEDBACK - 2

- ***If making a function call (to AWS Lambda) and my internet disconnects, will the function keep running on the instance and keep the result for me or does it simply terminate?***
- If a client connection to an AWS Lambda function fails, the function should continue to run
- The issue is how is the result provided to the user
- If the result is not saved (persisted) somewhere, and only returned as a REST response object, the result is lost
- If the result is persisted in a data store, such as the simple storage service (S3), then the client can retrieve the result later on

9

FEEDBACK - 3

- ***If we want to get the response from an asynchronous call, do we need to make a synchronous call? (b/c async call has no response)***
- No. To obtain the response from an asynchronous call, the result must be fetched from a data store
- The **simple storage service (S3)** is most commonly used to persist data, but any database service can be used.
- Common alternatives:

1. DynamoDB (No SQL DB)
2. Amazon RDS & Aurora (managed relational database service)
3. SQS – Simple Queuing Service
4. SNS – Simple Notification Service
5. Elasticache
6. Document DB
7. Amazon MQ

10

FEEDBACK - 4

- ***Do asynchronous client calls to a server, save the client time moreso than synchronous calls ?***
- YES
- A synchronous call blocks the calling thread to wait for a result from the server
- If the programmer has not designed the client to be multi-threaded, then the client essentially is frozen while waiting for a results from the server – it can do nothing but wait
- The programmer can "spawn" a thread for the synchronous call while the parent thread performs other work
- Multi-threaded programming is more complex and resource intensive, however
- An asynchronous call frees the main thread immediately to do other work, and does not block and wait

11

FEEDBACK - 5

- ***When should synchronous vs. asynchronous client calls to a server be used ?***
- Asynchronous calls are best for long operations
  - Maintaining a network connection for more than 30-seconds is error prone
  - Example: mobile device traveling down I-5 switching cell towers
- Synchronous calls block the client program unless it is a multi-threaded client
  - No other work can happen while waiting
  - Good for short calls that are expected to quickly return a result (within a few seconds)
- Clients and servers can run out of "connections" if too many synchronous sessions occur simultaneously
  - Asynchronous calls close connections and lower the burden

12

FEEDBACK - 6

- ***Are AWS Lambda functions similar to Redis message queues ?***
- No
- AWS Lambda functions provide a general-purpose compute platform for running serverless function code for up to 15-minutes
- Redis queue is a message or job queue built using Redis as the underlying data store. Redis is not a queueing system itself, but Redis Lists are highly effective for building and managing queues
- Queues are a form of a persistent data store – not a general compute platform

13

AWS LAMBDA: VCPU SCALING W/ MEMORY

Function Memory	CPU time share
1769 MB	100 % = 1 vCPU
2389 MB	150 % = 1.5 vCPUs
3008 MB	200 % = 2 vCPUs
4158 MB	250 % = 2.5 vCPUs
5307 MB	300 % = 3 vCPUs
6192 MB	350 % = 3.5 vCPUs
7076 MB	400 % = 4 vCPUs (1 HT)
7960 MB	450 % = 4.5 vCPUs (1.5 HT)
8845 MB	500 % = 5 vCPUs (2 HT)
9543 MB	550 % = 5.5 vCPUs (2.5 HT)
10240 MB	600 % = 6 vCPUs (3 HT)

Based on:
https://stackoverflow.com/questions/66522916/aws-lambda-memory-vs-cpu-configuration

14

OBJECTIVES – 10/21

- Questions from 10/16
- **Tutorials Questions**
- Tutorial 4 – Intro to FaaS – AWS Lambda
- Background on AWS Lambda for the Term Project - II
- **From: Cloud Computing Concepts, Technology & Architecture: Chapter 4: Cloud Computing Concepts and Models:**
  - Roles and boundaries
  - Cloud characteristics
  - Cloud delivery models
  - Cloud deployment models
- Team Planning - Breakout Rooms

15

TUTORIAL 0

- Getting Started with AWS
- https://faculty.washington.edu/wlloyd/courses/tcss562/tutorials/TCSS462_562_f2025_tutorial_0.pdf
- Create an AWS account
- Create account credentials for working with the CLI
- Install awsconfig package
- Setup awsconfig for working with the AWS CLI

16

TUTORIAL 2 – OCT 21

- **Introduction to Bash Scripting**
- https://faculty.washington.edu/wlloyd/courses/tcss562/tutorials/TCSS462_562_f2025_tutorial_2.pdf
- Review tutorial sections:
- Create a BASH webservice client
  1. What is a BASH script?
  2. Variables
  3. Input
  4. Arithmetic
  5. If Statements
  6. Loops
  7. Functions
  8. User Interface
- Call service to obtain IP address & lat/long of computer
- Call weatherbit.io API to obtain weather forecast for lat/long

17

TUTORIAL 3 – OCT 30 (TEAMS OF 2)

- Best Practices for Working with Virtual Machines on Amazon EC2
- https://faculty.washington.edu/wlloyd/courses/tcss562/tutorials/TCSS462_562_f2025_tutorial_3.pdf
- Creating a spot VM
- Creating an image from a running VM
- Persistent spot request
- Stopping (pausing) VMs
- EBS volume types
- Ephemeral disks (local disks)
- Mounting and formatting a disk
- Disk performance testing with Bonnie++
- Cost Saving Best Practices

18

OBJECTIVES – 10/21

- Questions from 10/16
- Tutorials Questions
- **Tutorial 4 – Intro to FaaS – AWS Lambda**
- Background on AWS Lambda for the Term Project - II
- **From: Cloud Computing Concepts, Technology & Architecture: Chapter 4: Cloud Computing Concepts and Models:**
  - Roles and boundaries
  - Cloud characteristics
  - Cloud delivery models
  - Cloud deployment models
- Team Planning - Breakout Rooms

TUTORIAL 4 – TO BE POSTED

- Introduction to AWS Lambda with the Serverless Application Analytics Framework (SAAF)
- (link to be posted)
- Setting up a Java development environment (IDE)
- Introduction to Maven build files for Java
- Create and Deploy "hello" Java AWS Lambda Function
  - Creation of API Gateway REST endpoint
- Sequential testing of "hello" AWS Lambda Function
  - API Gateway endpoint
  - AWS CLI Function invocation
- Observing SAAF profiling output
- Parallel testing of "hello" AWS Lambda Function with faas_runner tool
- Performance analysis using faas_runner reports
- Two function pipeline development task: Caesar Cipher

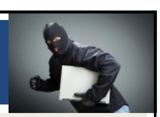
OBJECTIVES – 10/21

- Questions from 10/16
- Tutorials Questions
- Tutorial 4 – Intro to FaaS – AWS Lambda
- **Background on AWS Lambda for the Term Project - II**
- **From: Cloud Computing Concepts, Technology & Architecture: Chapter 4: Cloud Computing Concepts and Models:**
  - Roles and boundaries
  - Cloud characteristics
  - Cloud delivery models
  - Cloud deployment models
- Team Planning - Breakout Rooms

AWS LAMBDA PLATFORM LIMITATIONS - 2

- 10 concurrent function executions inside account (default)
- Function payload: 6MB (synchronous), 256KB (asynchronous)
- Deployment package: 50MB (compressed), 250MB (unzipped)
- Container image size: 10 GB
- Processes/threads: 1024
- File descriptors: 1024
- Function instances run Amazon Linux 2023
  - Based on a combination of Red Hat open-source Linux distributions: Fedora (versions 34, 35, 36) and CentOS 9 Stream
- Suggested Reading:
- https://docs.aws.amazon.com/lambda/latest/dg/gettingstarted-limits.html

CPUSTEAL



- *CpuSteal*: Metric that measures when a CPU core is ready to execute but the physical CPU core is busy and unavailable
- Symptom of over provisioning physical servers in the cloud
- Factors which cause *CpuSteal*: (x86 hyperthreading)
  1. Physical CPU is shared by too many busy VMs
  2. Hypervisor kernel is using the CPU
     - On AWS Lambda this would be the Firecracker MicroVM which is derived from the KVM hypervisor
  3. VM's CPU time share <100% for 1 or more cores, and 100% is needed for a CPU intensive workload.
- Man procfs – press "/" – type "proc/stat"
  - CpuSteal is the 8th column returned
  - Metric can be read using SAAF in tutorial #4

Snippet of sample output returned by SAAF (Tutorial 4)


```
{
   "version": 0.2,
   "lang": "python",
   "cpuType": "Intel(R) Xeon(R) Processor @ 2.50GHz",
   "cpuModel": 63,
   "vmuptime": 1551727835,
   "uuid": "d241c618-78d8-48e2-9736-997dc1a931d4",
   "vmID": "tiUCnA",
   "platform": "AWS Lambda",
   "newcontainer": 1,
   "cpuUsrDelta": "904",
   "cpuNiceDelta": "0",
   "cpuKrnDelta": "585",
   "cpuIdleDelta": "82428",
   "cpuIowaitDelta": "226",
   "cpuIrqDelta": "0",
   "cpuSoftIrqDelta": "7",
   "vmcpustealDelta": "1594",
   "frameworkRuntime": 35.72,
   "message": "Hello Fred Smith!",
   "runtime": 38.94
}
```

EFFECTS OF SCALING FUNCTION MEMORY ON CPU TIME SHARE

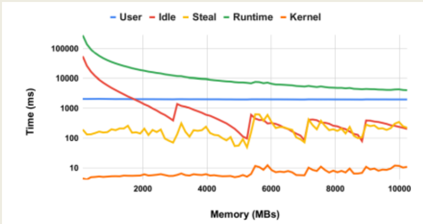
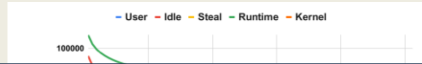


Figure 2: Linux CPU Utilization (log scale) vs. Function Memory for Sysbench Prime Number Generation

October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.25

25

EFFECTS OF SCALING FUNCTION MEMORY ON CPU TIME SHARE



Key observations:
- Runtime decreases as vCPUs and CPU time share increase
- CPU user time remains constant for the prime number generation task – work doesn't change
- CPU idle time gradually decreases as memory and vCPUs increase (the idle time is becoming active time)
- When the 4th vCPU is added, cpuSteal tracks closely with cpuIdle time (hyperthreading effect)
- There is more cpu Kernel time after the 4th vCPU is added

October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.26

26

FUNCTION INSTANCE LIFE CYCLES

- Function states:
- **COLD**: brand new function instance just initialized to run the request (more overhead)
  - Platform cold (first time ever run)
  - Host cold (function assets cached locally on servers)
- **WARM**: existing function instance that is reused
- All function instances persist for ~5 minutes before they begin to be "garbage collected" by the platform
  - 100% garbage collection may take up to ~30-40 minutes
- AWS Lambda appears to "recycle" infrastructure faster than other FaaS platforms
  - Presumably because of need, because the platform is busy

October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.27

27

WARM VS COLD FUNCTION INSTANCES

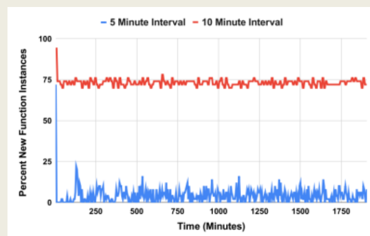


Figure 3: AWS Lambda Function Instance Replacement vs. Function Call Interval over 24-hours

October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L9.28

28

CPU SCALING VS MEMORY: OTHER PLATFORMS: 0 TO 3 GB

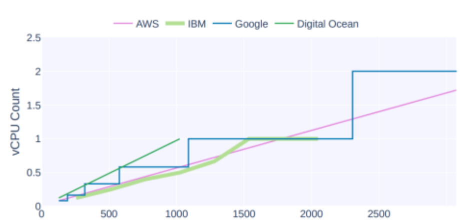


Fig. 2. Allocated vCPUs available at each memory setting on each platform.

October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.29

29

CPU SCALING VS MEMORY: OTHER PLATFORMS: 0 TO 3 GB

Key observations:
- Google only supports strict memory steps
- AWS gradually increases the CPU time share as memory is increased
- IBM is similar but slope is not constant
- Digital Ocean only scales up to 1 GB

Fig. 2. Allocated vCPUs available at each memory setting on each platform.

October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.30

30

ELASTIC FILE SYSTEM (AWS EFS)

- Traditionally AWS Lambda functions have been limited to 500MB of storage space
- Recently the Elastic File System (EFS) has been extended to support AWS Lambda
- The Elastic File System supports the creation of a shared volume like a shared disk (or folder)
  - EFS is similar to NFS (network file share)
  - Multiple AWS Lambda functions and/or EC2 VMs can mount and share the same EFS volume
  - Provides a shared R/W disk
  - Breaks the 500MB capacity barrier on AWS Lambda
- *Downside: EFS is expensive: ~30 ¢/GB/month*
- **Project**: EFS performance & scalability evaluation on Lambda

31

SERVERLESS APPLICATION – DESIGN TRADEOFFS

- Serverless file systems: EFS, docker container, extended /tmp
- Service/function composition / decomposition
- Switchboard architecture
- Application control flow
- Programming language comparison ***course theme w/ LLMs)***
- FaaS platforms: AWS, Azure, Google, etc.
- Alternate data services/backends for application state, large data transfer, short to long term data persistence
- Performance variability
  - Temporal: 24 hour, 7 days, etc. (diurnal patterns?)
  - Geospatial: By Region, availability zone
  - From HW heterogeneity (alternate CPUs)

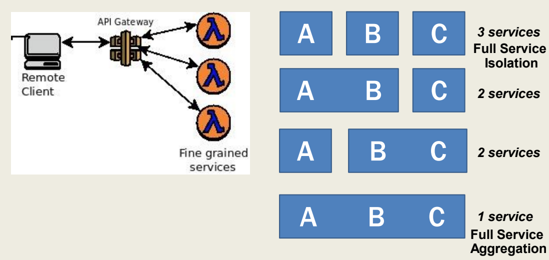
32

SERVERLESS FILE STORAGE COMPARISON PROJECT

- Elastic File System (EFS):
  Performance, Cost, and Scalability Evaluation in the context of AWS Lambda / Serverless Computing
  - EFS provides a file system that can be shared with multiple Lambda function instances in parallel
- Using a common use case, compare performance and cost of extended storage options on AWS Lambda:
  - Docker container support (up to 10 GB) – read only
  - Ephemeral /tmp (up to 10 GB) – read/write
  - EFS (unlimited, but costly) – read/write
  - image integration with AWS Lambda – performance & scalability

33

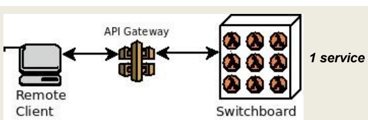
SERVICE COMPOSITION



Other possible compositions: group by library, functional cohesion, etc.

34

SWITCH-BOARD ARCHITECTURE



Single deployment package with consolidated codebase (Java: one JAR file)

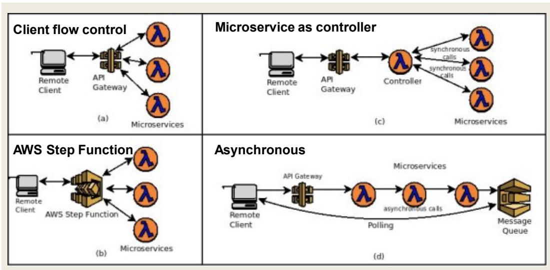
Entry method contains "switchboard" logic
Case statement that route calls to proper service

Routing is based on data payload
Check if specific parameters exist, route call accordingly

Goal: reduce # of COLD starts to improve performance

35

APPLICATION FLOW CONTROL - 3



36

PROGRAMMING LANGUAGE COMPARISON

- FaaS platforms support hosting code in multiple languages
- AWS Lambda- common: Java, Node.js, Python
  - Plus others: Go, PowerShell, C#, and Ruby
- Also Runtime API ("BASH") which allows deployment of binary executables from any programming language
- August 2020 – Our group's paper:
- https://tinyurl.com/y46eq6np
- If wanting to perform a language study either:
  - Implement in C#, Ruby, or multiple versions of Java, Node.js, Python
  - OR implement different app than TLQ (ETL) data processing pipeline

37

FAAS PLATFORMS

- Many commercial and open source FaaS platforms exist
- TCSS562 projects can choose to compare performance and cost implications of alternate platforms.
- Supported by SAAF:
- AWS Lambda
- Google Cloud Functions
- Azure Functions
- IBM Cloud Functions
- Apache OpenWhisk *(open source, deploy your own FaaS)*
- Open FaaS *(open source, deploy your own FaaS)*

38

DATA PROVISIONING

- Consider performance and cost implications of the data-tier design for the serverless application
- Use different tools as the relational datastore to support service #2 (LOAD) and service #3 (EXTRACT)
- SQL / Relational:
- Amazon Aurora (serverless cloud DB), Amazon RDS (cloud DB), DB on a VM (MySQL), DB inside Lambda function (SQLite, Derby)
- NO SQL / Key/Value Store:
- Dynamo DB, MongoDB, S3

39

PERFORMANCE VARIABILITY

- Cloud platforms exhibit performance variability which varies over time
- Goal of this case study is to measure performance variability (i.e. extent) for AWS Lambda services by hour, day, week to look for common patterns
- Can also examine performance variability by availability zone and region
  - Do some regions provide more stable performance?
  - Can services be switched to different regions during different times to leverage better performance?
- Remember that performance = cost
- If we make it faster, we make it cheaper…

40

CPU STEAL CASE STUDY

- On AWS Lambda (or other FaaS platforms), when we run functions, how much CpuSteal do we observe?
- How does CpuSteal vary for different workloads? (e.g. functions that have different resource requirements)
- How does CpuSteal vary over time hour, day, week, location?
- How does CpuSteal relate to function performance?

41

CPU ARCHITECTURE & PERFORMANCE

- X86_64 Intel
  - Intel Xeon Platinum 8259 CL @ 2.5 GHz
- ARM64 Graviton2

42

OBJECTIVES – 10/21

- Questions from 10/16
- Tutorials Questions
- Tutorial 4 – Intro to FaaS – AWS Lambda
- Background on AWS Lambda for the Term Project - II
- From: Cloud Computing Concepts, Technology & Architecture: Chapter 4: Cloud Computing Concepts and Models:
  - **Roles and boundaries**
  - Cloud characteristics
  - Cloud delivery models
  - Cloud deployment models
- Team Planning - Breakout Rooms

October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.43

43

CLOUD COMPUTING: CONCEPTS AND MODELS

October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.44

44

ROLES

- **Cloud provider**
  - Organization that provides cloud-based resources
  - Responsible for fulfilling SLAs for cloud services
  - Some cloud providers "resell" IT resources from other cloud providers
    - Example: Heroku sells PaaS services running atop of Amazon EC2
- **Cloud consumers**
  - Cloud users that consume cloud services
- **Cloud service owner**
  - Both cloud providers and cloud consumers can own cloud services
  - A cloud service owner may use a cloud provider to provide a cloud service (e.g. Heroku)

October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.45

45

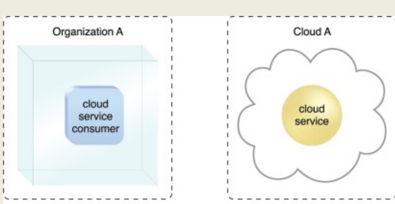
ROLES - 2

- **Cloud resource administrator**
  - Administrators provide and maintain cloud services
  - Both cloud providers and cloud consumers have administrators
- **Cloud auditor**
  - Third-party which conducts independent assessments of cloud environments to ensure security, privacy, and performance.
  - Provides unbiased assessments
- **Cloud brokers**
  - An intermediary between cloud consumers and cloud providers
  - Provides service aggregation
- **Cloud carriers**
  - Network and telecommunication providers which provide network connectivity between cloud consumers and providers

October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.46

46

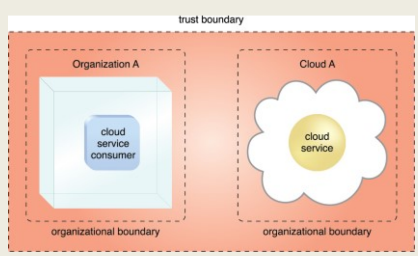
ORGANIZATION BOUNDARY



October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.47

47

TRUST BOUNDARY



October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.48

48

OBJECTIVES – 10/21

- Questions from 10/16
- Tutorials Questions
- Tutorial 4 – Intro to FaaS – AWS Lambda
- Background on AWS Lambda for the Term Project - II
- From: Cloud Computing Concepts, Technology & Architecture: Chapter 4: Cloud Computing Concepts and Models:
  - Roles and boundaries
  - **Cloud characteristics**
  - Cloud delivery models
  - Cloud deployment models
- Team Planning - Breakout Rooms

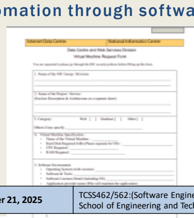
CLOUD CHARACTERISTICS

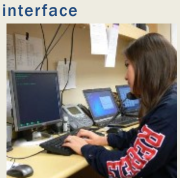
- On-demand usage
- Ubiquitous access
- Multitenancy (resource pooling)
- Elasticity
- Measured usage
- Resiliency

- Assessing these features helps measure the value offered by a given cloud service or platform

ON-DEMAND USAGE

- The freedom to self-provision IT resources
- Generally, with automated support
- Automated support requires no human involvement
- Automation through software services interface





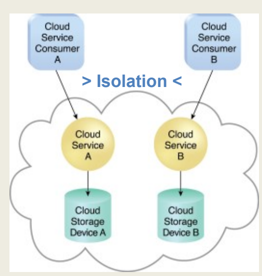
UBIQUITOUS ACCESS

- Cloud services are widely accessible
- Public cloud: internet accessible
- Private cloud: throughout segments of a company's intranet
- 24/7 availability

MULTITENANCY

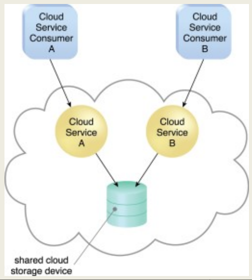
- Cloud providers pool resources together to share them with many users
- Serve multiple cloud service consumers
- IT resources can be dynamically assigned, reassigned based on demand
- Multitenancy can lead to performance variation

SINGLE TENANT MODEL



MULTITENANT MODEL

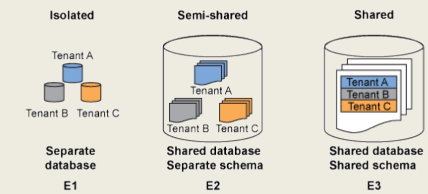
- Resource is "multiplexed" and share amongst multiple users
- Goal is to increase utilization
- Often server resources are underutilized
- There are many "sunk costs" whether usage is 0% or 100%
- Cloud computing tries to maximize "sunk cost" investments through multi-tenancy



October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.55

55

MULTITENANT DATABASE



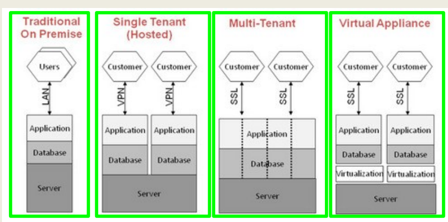
- Many users on a single database instance
- *What issues may occur when sharing a single database instance?*

October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.56

56

MULTITENANCY OF RESOURCES

- Where is the multitenancy?
  - >> What is shared? What is isolated?



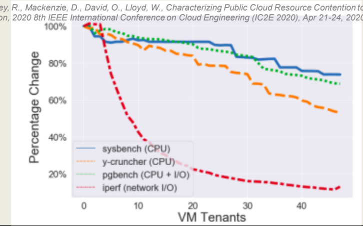
October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.57

57

RESOURCE CONTENTION FROM MUTLI-TENANCY

- Despite best efforts at isolation, co-resident VMs on a single cloud server running identical benchmarks simultaneously do not perform equally.

From Han, X., Schooley, R., Mackenzie, D., David, O., Lloyd, W., Characterizing Public Cloud Resource Contention to Support Virtual Machine Co-residency Prediction, 2020 8th IEEE International Conference on Cloud Engineering (IC2E 2020), Apr 21-24, 2020.



Up to 48 VMs sharing same server !!

October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.58

58

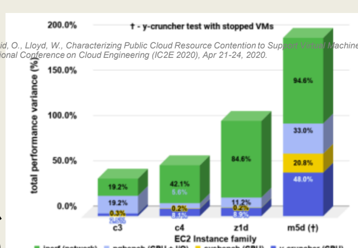
RESOURCE CONTENTION FROM MUTLI-TENANCY - 2

- Performance variation from multi-tenancy is increasing as cloud servers add more CPU cores

From Han, X., Schooley, R., Mackenzie, D., David, O., Lloyd, W., Characterizing Public Cloud Resource Contention to Support Virtual Machine Co-residency Prediction, 2020 8th IEEE International Conference on Cloud Engineering (IC2E 2020), Apr 21-24, 2020.

- Running many idle operating system instances can impose significant overhead for some workloads

Maximum potential resource contention (i.e. worst-case scenario) →



October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.59

59

ELASTICITY

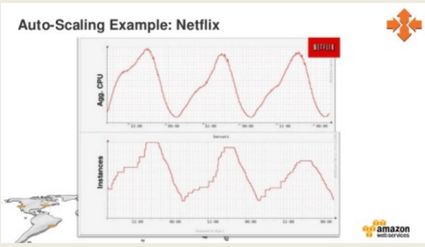
- Automated ability of cloud to transparently scale resources
- Scaling based on runtime conditions or pre-determined by cloud consumer or cloud provider
- Threshold based scaling
  - `CPU-utilization > threshold_A, Response_time > 100ms`
  - Application agnostic vs. application specific thresholds
  - Why might an application agnostic threshold be non-ideal?
- Load prediction
  - Historical models
  - Real-time trends

October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.60

60

PREDICTABLE DEMAND

- AWS EC2 Scaling Example:



From: Kejariwal, A., 2013, March. Techniques for optimizing cloud footprint. In 2013 IEEE Int. Conf. on Cloud Engineering (IC2E), pp. 258-268.

October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.61

61

MEASURED USAGE

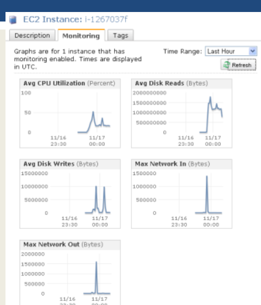
- Cloud platform tracks usage of IT resources
- For billing purposes
- Enables charging only for IT resources actually used
- Can be time-based (millisec, second, minute, hour, day)
  - Granularity is increasing…
- Can be throughput-based (data transfer: MB/sec, GB/sec)
- Can be resource/reservation based (vCPU/hr, GB/hr)

- Not all measurements are for billing
- Some measurements can support auto-scaling
- For example CPU utilization

October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.62

62

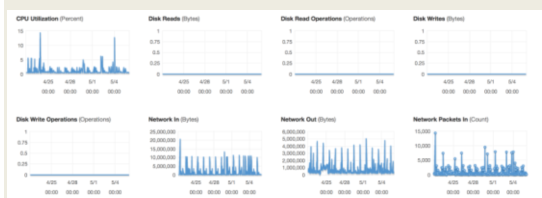
EC2 CLOUDWATCH METRICS



October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.63

63

EC2 CLOUDWATCH METRICS

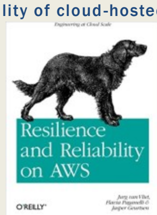


October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.64

64

RESILIENCY

- Distributed redundancy across physical locations (regions on AWS)
- Used to improve reliability and availability of cloud-hosted applications
- Very much an engineering problem
- No "resiliency-as-a-service" for user deployed apps
- Unique characteristics of user applications make a one-size fits all service solution challenging



October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.65

65

Elasticity is often provided using threshold based scaling. When can threshold based scaling (i.e. CPU utilization > 80%) under or over provision resources?

- When the application is primarily I/O bound, a CPU threshold may never be met, or be met too late to scale up. **A**
- When the current resource utilization does not reflect future system demand. **B**
- When the current resource utilization (e.g. CPU) is temporarily increased as a result of external factors (i.e. resource contention from other tasks) that does not correlate to system demand. **C**
- When an application will soon complete a parallel phase, before executing a largely sequential phase **D**
- All of the above **E**

October 24, 2016 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L10.66

66

When poll is active, respond at **pollev.com/wesleylloyd641**
Text **WESLEYLLOYD641** to **22333** once to join

The scaling threshold of "when CPU utilization > 80% scale up", is:

- An application specific threshold
- An application agnostic threshold

Start the presentation to see live content. For screen share software, share the entire screen. Get help at **pollev.com/app**

67

OBJECTIVES – 10/21

- Questions from 10/16
- Tutorials Questions
- Tutorial 4 – Intro to FaaS – AWS Lambda
- Background on AWS Lambda for the Term Project - II
- **From: Cloud Computing Concepts, Technology & Architecture: Chapter 4: Cloud Computing Concepts and Models:**
  - Roles and boundaries
  - Cloud characteristics
  - **Cloud delivery models**
  - Cloud deployment models
- Team Planning - Breakout Rooms

October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.68

68

CLOUD COMPUTING DELIVERY MODELS

- Infrastructure-as-a-Service (IaaS)
- Platform-as-a-Service (PaaS)
- Software-as-a-Service (SaaS)

Serverless Computing:
- Function-as-a-Service (FaaS)
- Container-as-a-Service (CaaS)
- Other Delivery Models

October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.69

69

CLOUD COMPUTING DELIVERY MODELS

- Infrastructure-as-a-Service (IaaS) delivery model
- Virtualization is a key-enabling technology of IaaS cloud
- Uses virtual machines to deliver cloud resources to end users

October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.70

70

CLOUD COMPUTING DELIVERY MODELS

Virtualization is key to sharing powerful servers among users by running *many* isolated private virtual computers known as virtual machines (VMs)

…VMs are the basis of cloud v1.0

October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.71

71

CLOUD COMPUTING DELIVERY MODELS

Virtual Machines are the building blocks for "Cloud Service Delivery Models"

They are the "vehicles" used to deliver compute resources to end users…

cloud 1.0

October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.72

72

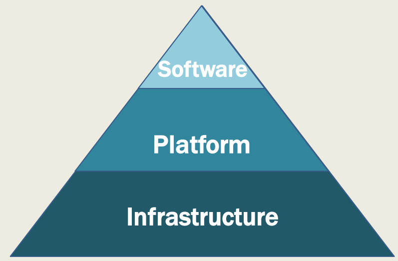
CLOUD DELIVERY MODELS

- What is the appropriate level of **abstraction**?
- How should applications be deployed?
  - IaaS, PaaS, SaaS, DbaaS, FaaS
- How do we ensure Quality-of-Service?
  - Performance, Availability, Responsiveness, Fault Tolerance
- How is ***scalability*** provided?
- As users, how do we minimize hosting costs?
  - How do we estimate hosting costs?

October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.73

73

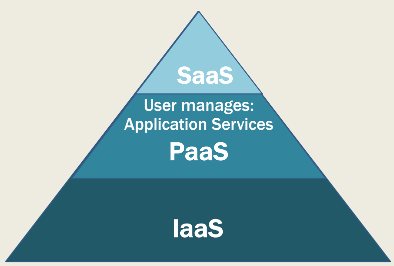
CLASSIC CLOUD DELIVERY MODELS



October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.74

74

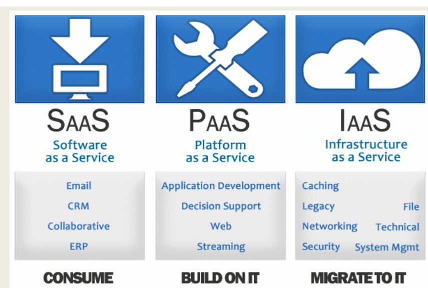
CLASSIC CLOUD DELIVERY MODELS



October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.75

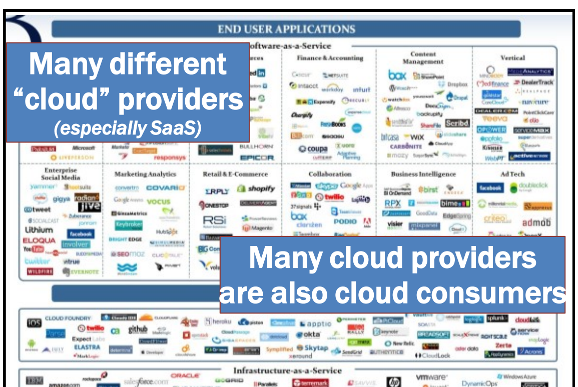
75

EXAMPLE CLOUD SERVICES



October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.76

76



October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.77

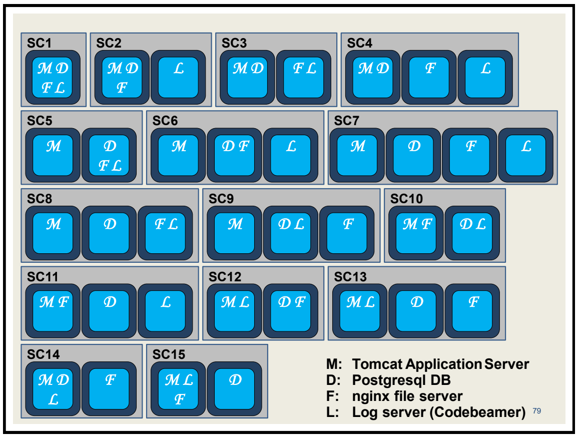
77

INFRASTRUCTURE-AS-A-SERVICE

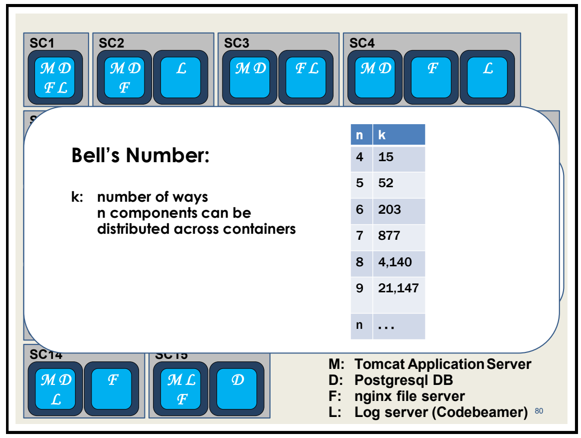
- Compute resources, on demand, as-a-service
  - Generally raw "IT" resources
  - Hardware, network, containers, operating systems
- Typically provided through virtualization
- Generally, not-preconfigured
- Administrative burden is owned by cloud consumer
- Best when high-level control over environment is needed
- Scaling is generally **not** automatic…
- Resources can be managed in bundles
- AWS CloudFormation: Allows specification in JSON/YAML of cloud infrastructures

October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.78

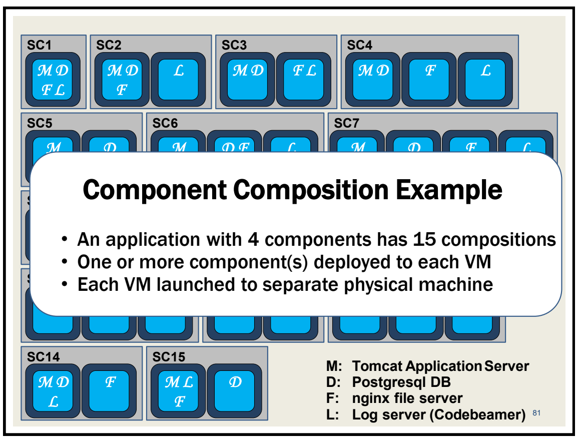
78



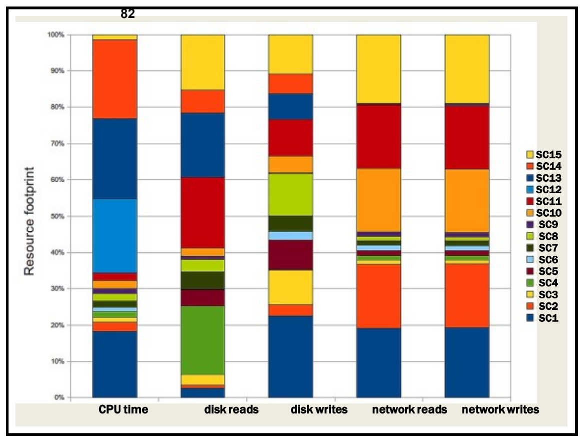
79



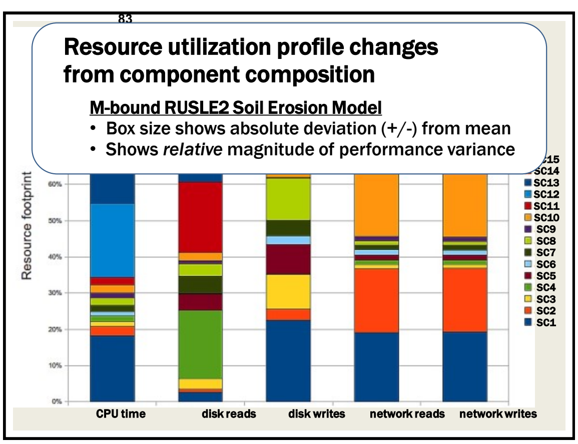
80



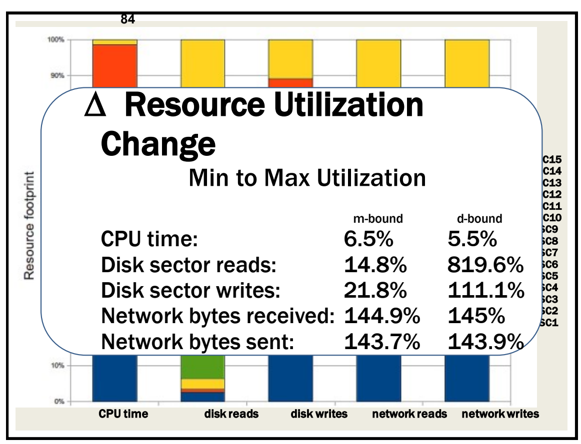
81



82



83

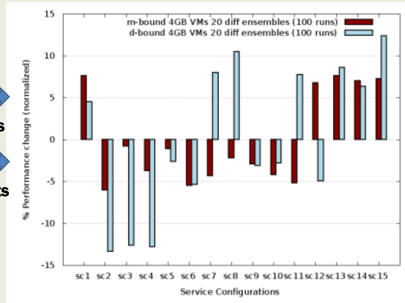


84

PERFORMANCE IMPLICATIONS OF APPLICATION DEPLOYMENTS

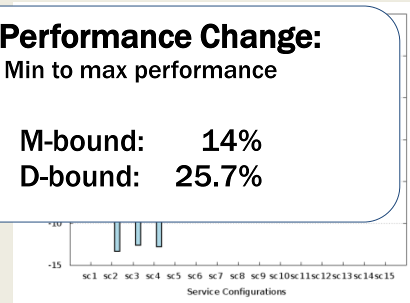
Slower deployments

Faster deployments



85

PERFORMANCE IMPLICATIONS OF APPLICATION DEPLOYMENTS



Δ Performance Change:

Min to max performance

M-bound: 14%

D-bound: 25.7%

86

CLOUD COMPUTING DELIVERY MODELS

- Infrastructure-as-a-Service (IaaS)
- Platform-as-a-Service (PaaS)
- Software-as-a-Service (SaaS)

Serverless Computing:

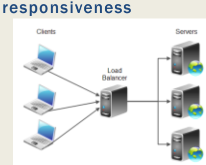
- Function-as-a-Service (FaaS)
- Container-as-a-Service (CaaS)
- Other Delivery Models

87

PLATFORM-AS-A-SERVICE

- Predefined, ready-to-use, hosting environment
- Infrastructure is further obscured from end user
- Scaling and load balancing may be automatically provided and automatic
- Variable to no ability to influence responsiveness

- Examples:
- Google App Engine
- Heroku
- AWS Elastic Beanstalk
- AWS Lambda (FaaS)



88

USES FOR PAAS

- Cloud consumer
  - Wants to extend on-premise environments into the cloud for “web app” hosting
  - Wants to entirely substitute an on-premise hosting environment
  - Cloud consumer wants to become a cloud provider and deploy its own cloud services to external users

- PaaS spares IT administrative burden compared to IaaS

89

CLOUD COMPUTING DELIVERY MODELS

- Infrastructure-as-a-Service (IaaS)
- Platform-as-a-Service (PaaS)
- Software-as-a-Service (SaaS)

Serverless Computing:

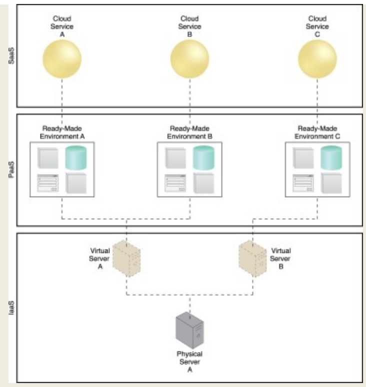
- Function-as-a-Service (FaaS)
- Container-as-a-Service (CaaS)
- Other Delivery Models

90

SOFTWARE-AS-A-SERVICE

- Software applications as shared cloud service
- Nearly all server infrastructure management is abstracted away from the user
- Software is generally configurable
- SaaS can be a complete GUI/UI based environment
- Or UI-free (database-as-a-service)
- SaaS offerings
  - Google Docs
  - Office 365
  - Cloud9 Integrated Development Environment
  - Salesforce

91



92

CLOUD COMPUTING DELIVERY MODELS

- Infrastructure-as-a-Service (IaaS)
- Platform-as-a-Service (PaaS)
- Software-as-a-Service (SaaS)

Serverless Computing:
- Function-as-a-Service (FaaS)
- Container-as-a-Service (CaaS)
- Other Delivery Models

93

SERVERLESS COMPUTING

Introducing Cloud 2.0

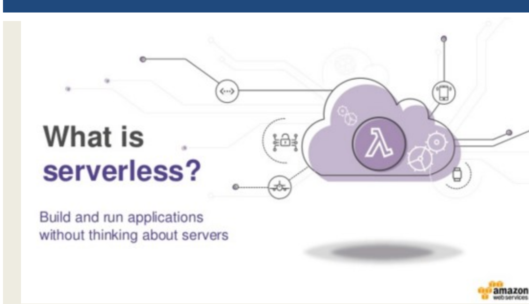
Image from: https://mobisoftinfotech.com/resources/blog/serverless-computing-deploy-applications-without-fiddling-with-servers/

94

SERVERLESS COMPUTING

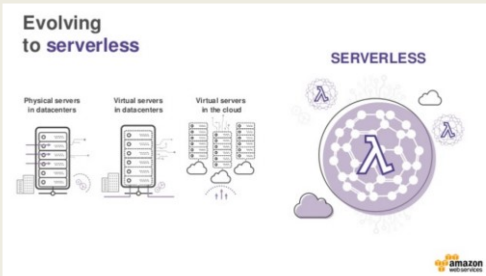
95

SERVERLESS COMPUTING



96

SERVERLESS COMPUTING - 2



97

SERVERLESS COMPUTING

- Pay only for CPU/memory utilization
- High Availability
- Fault Tolerance
- Infrastructure Elasticity
- No Setup
- Function-as-a-Service (FAAS)

98

SERVERLESS COMPUTING

Why Serverless Computing?

Many features of distributed systems, that are challenging to deliver, are provided automatically

…they are built into the platform

99

CLOUD COMPUTING DELIVERY MODELS

- Infrastructure-as-a-Service (IaaS)
- Platform-as-a-Service (PaaS)
- Software-as-a-Service (SaaS)

Serverless Computing:

- Function-as-a-Service (FaaS)
- Container-as-a-Service (CaaS)
- Other Delivery Models

100

SERVERLESS VS. FAAS

- Serverless Computing
- Refers to the avoidance of managing servers
- Can pertain to a number of “as-a-service” cloud offerings
- Function-as-a-Service (FaaS)
  - Developers write small code snippets (microservices) which are deployed separately
- Database-as-a-Service (DBaaS)
- Container-as-a-Service (CaaS)
- Others…

- Serverless is a buzzword
- This space is evolving…

101

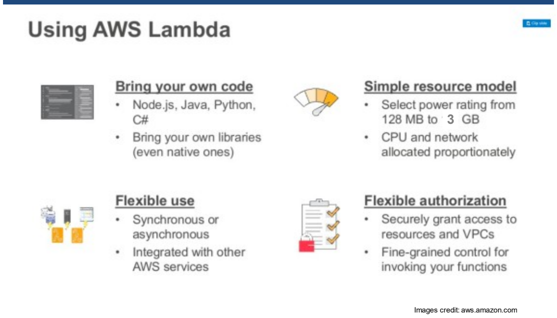
FAAS PLATFORMS

Commercial
- AWS Lambda
- Azure Functions
- IBM Cloud Functions
- Google Cloud Functions

Open Source
- Apache OpenWhisk
- Fn (Oracle)

102

AWS LAMBDA



103

FAAS PLATFORMS - 2

- New cloud platform for hosting application code
- Every cloud vendor provides their own:
  - AWS Lambda, Azure Functions, Google Cloud Functions, IBM OpenWhisk
- Similar to platform-as-a-service
- Replace opensource web container (e.g. Apache Tomcat) with abstracted vendor-provided **black-box** environment

104

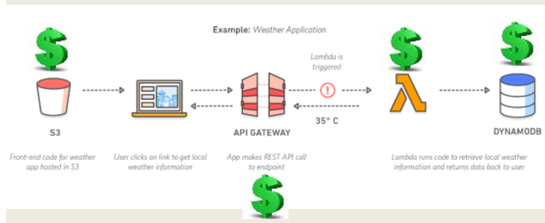
FAAS PLATFORMS - 3

- Many challenging features of distributed systems are provided automatically
- ***Built into the platform:***
- Highly availability (24/7)
- Scalability
- Fault tolerance

105

CLOUD NATIVE SOFTWARE ARCHITECTURE

- Every service with a different pricing model



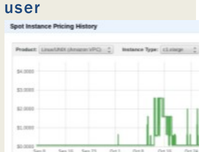
106

IAAS BILLING MODELS

- Virtual machines as-a-service at ¢ per hour
- No premium to scale:

```
      1000 computers   @    1 hour
=        1 computer    @ 1000 hours
```

- Illusion of infinite scalability to cloud user
- As many computers as you can afford
- Billing models are becoming increasingly granular
  - By the minute, second, 1/10th sec
- Auction-based instances: Spot instances →



107

PRICING OBFUSCATION

- **VM pricing:** hourly rental pricing, billed to nearest second is intuitive…
- **FaaS pricing:** non-intuitive pricing policies
- **FREE TIER:**
  first 1,000,000 function calls/month → FREE
  first 400,000 GB-sec/month → FREE
- Afterwards: *obfuscated pricing (AWS Lambda):*
  $0.0000002 per request
  $0.000000208 to rent 128MB / 100-ms
  $0.00001667 GB /second

108

WEBSERVICE HOSTING EXAMPLE

- ON AWS Lambda
- Each service call: 100% of 1 CPU-core
  100% of 4GB of memory
- Workload: 2 continuous client threads
- Duration: 1 month (30 days)

- ON AWS EC2:
- Amazon EC2 c4.large 2-vCPU VM
- Hosting cost: $72/month
  c4.large: 10¢/hour, 24 hrs/day x 30 days

- **How much would hosting this workload cost on AWS Lambda?**

109

PRICING OBFUSCATION

Worst-case scenario = ~2.32x !

AWS EC2: $72.00
AWS Lambda: $167.01

Break Even: 4,319,136 GB-sec

Two threads
@2GB-ea: ~12.5 days

- BREAK-EVEN POINT: ~4,319,136 GB-sec-month
  ~12.5 days 2 concurrent clients @ 2GB

110

FAAS PRICING

- Break-even point is the point where renting VMs or deploying to a serverless platform (e.g. Lambda) is exactly the same.
- Our example is for one month
- Could also consider one day, one hour, one minute
- **What factors influence the break-even point for an application running on AWS Lambda?**

111

FACTORS IMPACTING PERFORMANCE OF FAAS COMPUTING PLATFORMS

- Infrastructure elasticity
- Load balancing
- Provisioning variation
- Infrastructure retention: COLD vs. WARM
  - Infrastructure freeze/thaw cycle
- Memory reservation
- Service composition

112

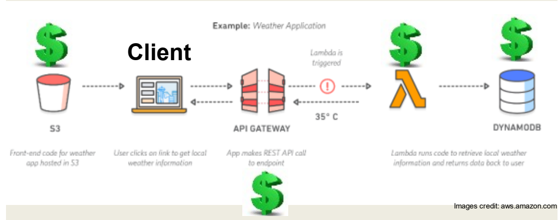
FAAS CHALLENGES

- Vendor architectural lock-in – how to migrate?
- Pricing obfuscation – is it cost effective?
- Memory reservation – how much to reserve?
- Service composition – how to compose software?
- Infrastructure freeze/thaw cycle – how to avoid?

113

VENDOR ARCHITECTURAL LOCK-IN

- Cloud native (FaaS) software architecture requires external services/components



- Increased dependencies → increased hosting costs

114

PRICING OBFUSCATION

- **VM pricing:** hourly rental pricing, billed to nearest second is intuitive…
- **FaaS pricing:**

AWS Lambda Pricing

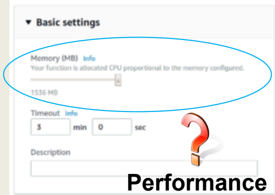
FREE TIER: first 1,000,000 function calls/month → FREE
first 400,000 GB-sec/month → FREE

- Afterwards: $0.0000002 per request
$0.000000208 to rent 128MB / 100-ms

115

MEMORY RESERVATION QUESTION…

- Lambda memory reserved for functions
- UI provides "slider bar" to set function's memory allocation
- Resource capacity (CPU, disk, network) coupled to slider bar:
"*every **doubling** of memory, **doubles** CPU…*"



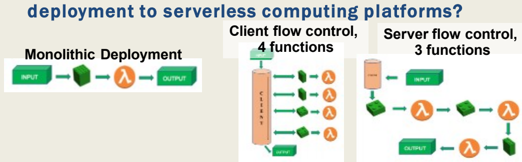
Performance

- **But how much memory do model services require?**

116

SERVICE COMPOSITION

- **How should application code be composed for deployment to serverless computing platforms?**



- Recommended practice: Decompose into many microservices
- Platform limits: code + libraries ~250MB
- **How does composition impact the number of function invocations, and memory utilization?**

Performance

117

INFRASTRUCTURE FREEZE/THAW CYCLE

- Unused infrastructure is deprecated
  - *But after how long?*
- Infrastructure: VMs, "containers"
- **Provider-COLD / VM-COLD**
  - "Container" images - built/transferred to VMs
- **Container-COLD**
  - Image cached on VM
- **Container-WARM**
  - "Container" running on VM

Performance

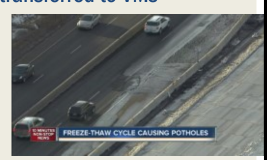


Image from: Denver7 – The Denver Channel News

118

FUNCTION-AS-A-SERVICE

AWS Lambda Demo

119

CLOUD COMPUTING DELIVERY MODELS

- Infrastructure-as-a-Service (IaaS)
- Platform-as-a-Service (PaaS)
- Software-as-a-Service (SaaS)

Serverless Computing:

- Function-as-a-Service (FaaS)
- Container-as-a-Service (CaaS)
- Other Delivery Models

120

CONTAINER-AS-A-SERVICE

- Cloud service model for deploying application containers (e.g. Docker) to the cloud
- Deploy containers without worrying about managing infrastructure:
  - Servers
  - Or container orchestration platforms
  - Container platform examples: Kubernetes, Docker swarm, Apache Mesos/Marathon, Amazon Elastic Container Service
  - Container platforms support creation of container clusters on the using cloud hosted VMs
- CaaS Examples:
  - AWS Fargate
  - Azure Container Instances
  - Google KNative

121

CLOUD COMPUTING DELIVERY MODELS

- Infrastructure-as-a-Service (IaaS)
- Platform-as-a-Service (PaaS)
- Software-as-a-Service (SaaS)

Serverless Computing:
- Function-as-a-Service (FaaS)
- Container-as-a-Service (CaaS)
- Other Delivery Models

122

OTHER CLOUD SERVICE MODELS

- IaaS
  - Storage-as-a-Service
- PaaS
  - Integration-as-a-Service
- SaaS
  - Database-as-a-Service
  - Testing-as-a-Service
  - Model-as-a-Service
- ?
  - Security-as-a-Service
  - Integration-as-a-Service

123

OBJECTIVES – 10/21

- Questions from 10/16
- Tutorials Questions
- Tutorial 4 – Intro to FaaS – AWS Lambda
- Background on AWS Lambda for the Term Project - II
- From: Cloud Computing Concepts, Technology & Architecture: Chapter 4: Cloud Computing Concepts and Models:
  - Roles and boundaries
  - Cloud characteristics
  - Cloud delivery models
  - **Cloud deployment models**
- Team Planning - Breakout Rooms

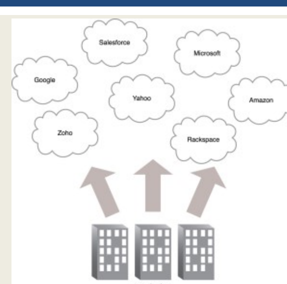
124

CLOUD DEPLOYMENT MODELS

- Distinguished by ownership, size, access
- Four common models
  - Public cloud
  - Community cloud
  - Hybrid cloud
  - Private cloud

125

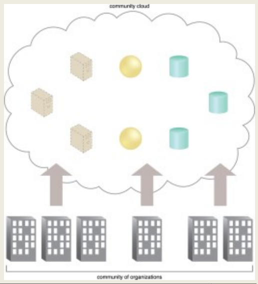
PUBLIC CLOUDS



126

COMMUNITY CLOUD

- Specialized cloud built and shared by a particular community
- Leverage economies of scale within a community
- Research oriented clouds
- Examples:
  - Bionimbus - bioinformatics
  - Chameleon
  - CloudLab

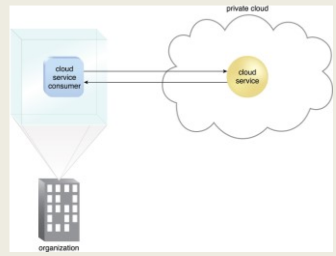


October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.127

127

PRIVATE CLOUD

- Compute clusters configured as IaaS cloud
- Open source software
- Eucalyptus
- Openstack
- Apache Cloudstack
- Nimbus
- Virtualization: XEN, KVM, …

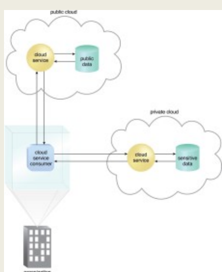


October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.128

128

HYBRID CLOUD

- Extend private cloud typically with public or community cloud resources
- Cloud bursting: Scale beyond one cloud when resource requirements exceed local limitations
- Some resources can remain local for security reasons



October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.129

129

OTHER CLOUDS

- Federated cloud
  - Simply means to aggregate two or more clouds together
  - Hybrid is typically private-public
  - Federated can be public-public, private-private, etc.
  - Also called inter-cloud
- Virtual private cloud
  - Google and Microsoft simply call these virtual networks
  - Ability to interconnect multiple independent subnets of cloud resources together
  - Resources allocated private IPs from individual network subnets can communicate with each other (10.0.1.0/24) and (10.0.2.0/24)
  - Subnets can span multiple availability zones within an AWS region

October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.130

130

OBJECTIVES – 10/21

- Questions from 10/16
- Tutorials Questions
- Tutorial 4 – Intro to FaaS – AWS Lambda
- Background on AWS Lambda for the Term Project - II
- From: Cloud Computing Concepts, Technology & Architecture: Chapter 4: Cloud Computing Concepts and Models:
  - Roles and boundaries
  - Cloud characteristics
  - Cloud delivery models
  - Cloud deployment models
- **Team Planning - Breakout Rooms**

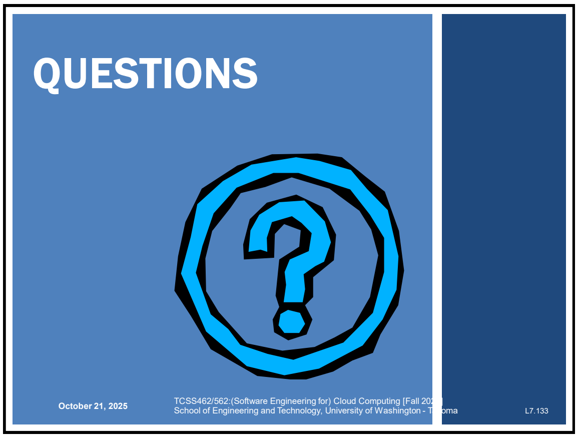
October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.131

131

TCSS 462/562 TERM PROJECT

October 21, 2025 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2025] School of Engineering and Technology, University of Washington - Tacoma | L7.132

132


QUESTIONS
October 21, 2025
TCSS462/562:(Software Engineering for) Cloud Computing
School of Engineering and Technology, University of Washington -
L7.133

133