TCSS 562: SOFTWARE ENGINEERING FOR CLOUD COMPUTING

Cloud Computing Concepts and Models - III

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

OFFICE HOURS – FALL 2024

- **Tuesdays:**
  - 2:30 to 3:30 pm - CP 229
- **Friday – this week:**
  - 1:00 pm to 2:00 pm – ONLINE via Zoom
- Or email for appointment

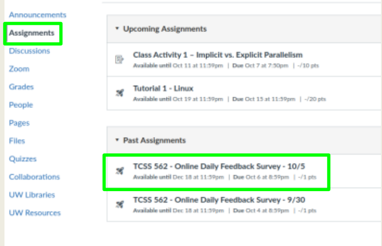
> Office Hours set based on Student Demographics survey feedback

OBJECTIVES – 10/24

- **Questions from 10/22**
- Tutorials Questions
- Tutorial 5 - Files in S3 and CloudWatch Events
- From: Cloud Computing Concepts, Technology & Architecture: Chapter 4: Cloud Computing Concepts and Models:
  - Cloud computing delivery models
  - Cloud deployment models
- AWS Overview and demo
- Upcoming:
  - (Tuesday) Activity 2 - Horizontal Scaling in the Cloud
  - Term Project Planning

ONLINE DAILY FEEDBACK SURVEY

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

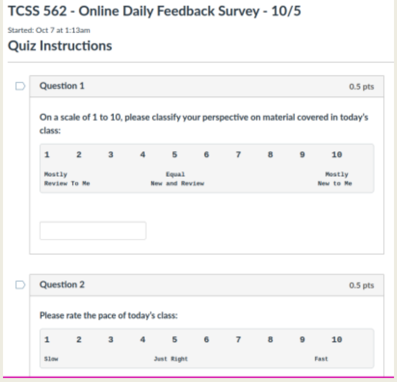


Class Activity 1 – Implicit vs. Explicit Parallelism
Available until Oct 11 at 11:59pm | Due Oct 7 at 7:50pm | -/10 pts

Tutorial 1 - Linux
Available until Oct 19 at 11:59pm | Due Oct 15 at 11:59pm | -/20 pts

TCSS 562 - Online Daily Feedback Survey - 10/5
Available until Dec 18 at 11:59pm | Due Oct 6 at 8:59pm | -/1 pts

TCSS 562 - Online Daily Feedback Survey - 9/30
Available until Dec 18 at 11:59pm | Due Oct 4 at 8:59pm | -/1 pts



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                Equal                           Mostly
Review To Me        New and Review                  New to Me
```

Question 2 0.5 pts

Please rate the pace of today's class:

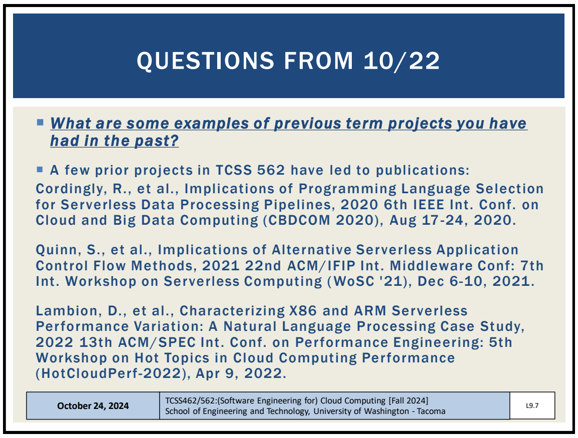
```
1     2     3     4     5     6     7     8     9     10
Slow                  Just Right                    Fast
```

MATERIAL / PACE

- Please classify your perspective on material covered in today's class (39 respondents):
- 1-mostly review, 5-equal new/review, 10-mostly new
- **Average – 6.05 (↑ - *previous* 6.01)**

- Please rate the pace of today's class:
- 1-slow, 5-just right, 10-fast
- **Average – 5.74 (↑ - previous 5.24)**

- **Response rates:**
- TCSS 462: 26/42 – 61.9%
- TCSS 562: 13/20 – 65.0%



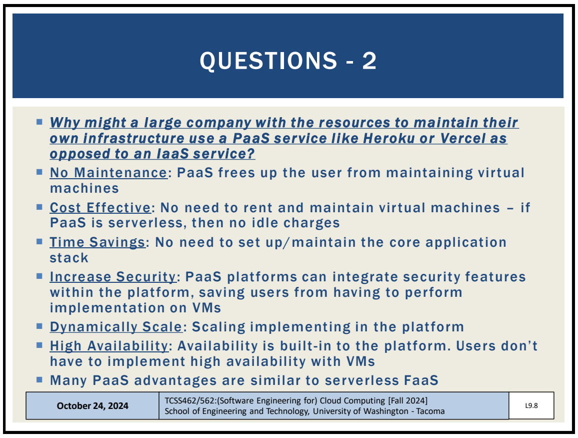
QUESTIONS FROM 10/22

- ***What are some examples of previous term projects you have had in the past?***
- A few prior projects in TCSS 562 have led to publications:

Cordingly, R., et al., Implications of Programming Language Selection for Serverless Data Processing Pipelines, 2020 6th IEEE Int. Conf. on Cloud and Big Data Computing (CBDCOM 2020), Aug 17-24, 2020.

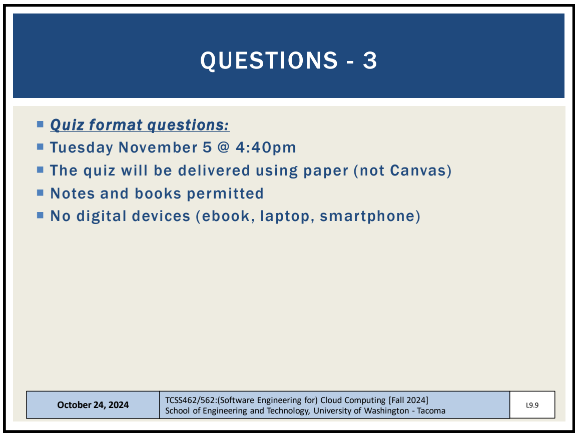
Quinn, S., et al., Implications of Alternative Serverless Application Control Flow Methods, 2021 22nd ACM/IFIP Int. Middleware Conf: 7th Int. Workshop on Serverless Computing (WoSC '21), Dec 6-10, 2021.

Lambion, D., et al., Characterizing X86 and ARM Serverless Performance Variation: A Natural Language Processing Case Study, 2022 13th ACM/SPEC Int. Conf. on Performance Engineering: 5th Workshop on Hot Topics in Cloud Computing Performance (HotCloudPerf-2022), Apr 9, 2022.



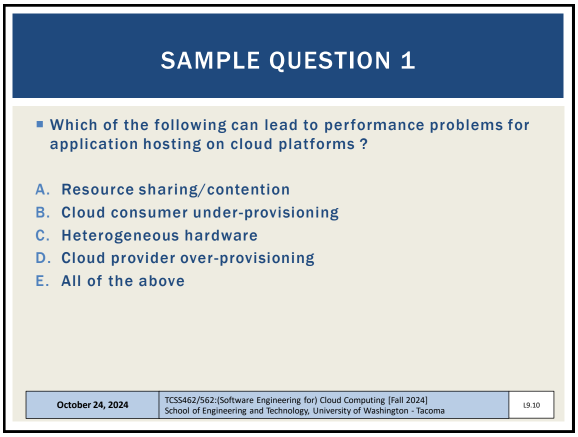
QUESTIONS - 2

- ***Why might a large company with the resources to maintain their own infrastructure use a PaaS service like Heroku or Vercel as opposed to an IaaS service?***
- No Maintenance: PaaS frees up the user from maintaining virtual machines
- Cost Effective: No need to rent and maintain virtual machines – if PaaS is serverless, then no idle charges
- Time Savings: No need to set up/maintain the core application stack
- Increase Security: PaaS platforms can integrate security features within the platform, saving users from having to perform implementation on VMs
- Dynamically Scale: Scaling implementing in the platform
- High Availability: Availability is built-in to the platform. Users don't have to implement high availability with VMs
- Many PaaS advantages are similar to serverless FaaS



QUESTIONS - 3

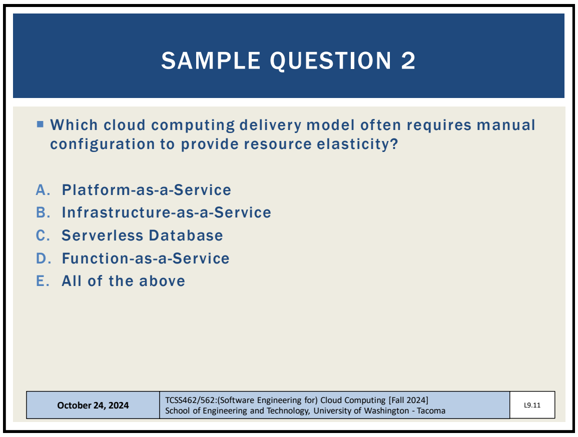
- ***Quiz format questions:***
- Tuesday November 5 @ 4:40pm
- The quiz will be delivered using paper (not Canvas)
- Notes and books permitted
- No digital devices (ebook, laptop, smartphone)



SAMPLE QUESTION 1

- Which of the following can lead to performance problems for application hosting on cloud platforms ?

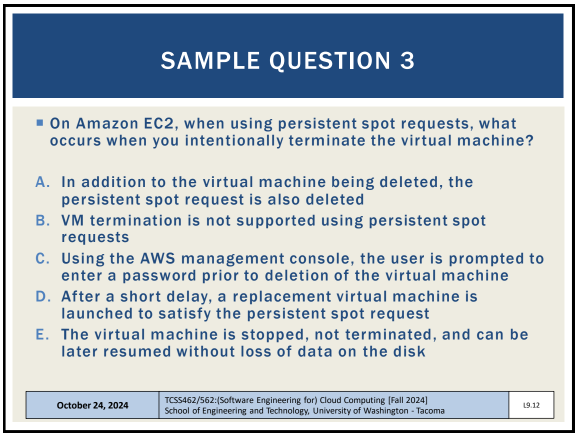
A. Resource sharing/contention
B. Cloud consumer under-provisioning
C. Heterogeneous hardware
D. Cloud provider over-provisioning
E. All of the above



SAMPLE QUESTION 2

- Which cloud computing delivery model often requires manual configuration to provide resource elasticity?

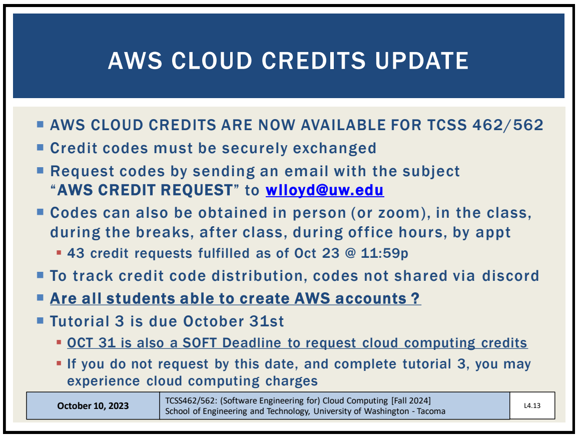
A. Platform-as-a-Service
B. Infrastructure-as-a-Service
C. Serverless Database
D. Function-as-a-Service
E. All of the above



SAMPLE QUESTION 3

- On Amazon EC2, when using persistent spot requests, what occurs when you intentionally terminate the virtual machine?

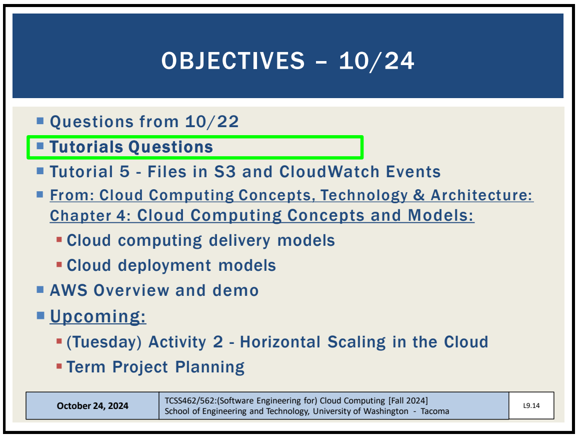
A. In addition to the virtual machine being deleted, the persistent spot request is also deleted
B. VM termination is not supported using persistent spot requests
C. Using the AWS management console, the user is prompted to enter a password prior to deletion of the virtual machine
D. After a short delay, a replacement virtual machine is launched to satisfy the persistent spot request
E. The virtual machine is stopped, not terminated, and can be later resumed without loss of data on the disk



AWS CLOUD CREDITS UPDATE

- **AWS CLOUD CREDITS ARE NOW AVAILABLE FOR TCSS 462/562**
- Credit codes must be securely exchanged
- Request codes by sending an email with the subject "**AWS CREDIT REQUEST**" to **wlloyd@uw.edu**
- Codes can also be obtained in person (or zoom), in the class, during the breaks, after class, during office hours, by appt
  - 43 credit requests fulfilled as of Oct 23 @ 11:59p
- To track credit code distribution, codes not shared via discord
- **Are all students able to create AWS accounts ?**
- Tutorial 3 is due October 31st
  - OCT 31 is also a SOFT Deadline to request cloud computing credits
  - If you do not request by this date, and complete tutorial 3, you may experience cloud computing charges

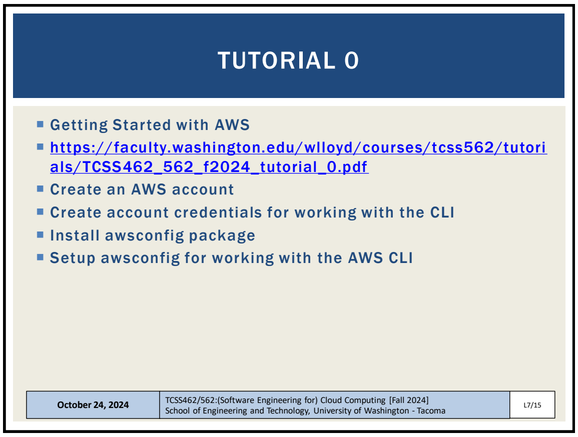
13



OBJECTIVES – 10/24

- Questions from 10/22
- **Tutorials Questions**
- Tutorial 5 - Files in S3 and CloudWatch Events
- From: Cloud Computing Concepts, Technology & Architecture: Chapter 4: Cloud Computing Concepts and Models:
  - Cloud computing delivery models
  - Cloud deployment models
- AWS Overview and demo
- Upcoming:
  - (Tuesday) Activity 2 - Horizontal Scaling in the Cloud
  - Term Project Planning

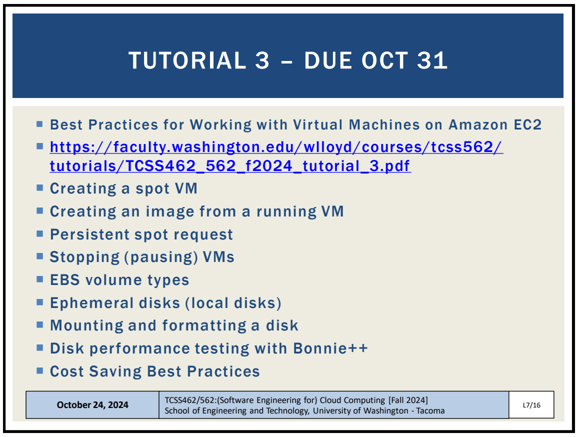
14



TUTORIAL 0

- Getting Started with AWS
- https://faculty.washington.edu/wlloyd/courses/tcss562/tutorials/TCSS462_562_f2024_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

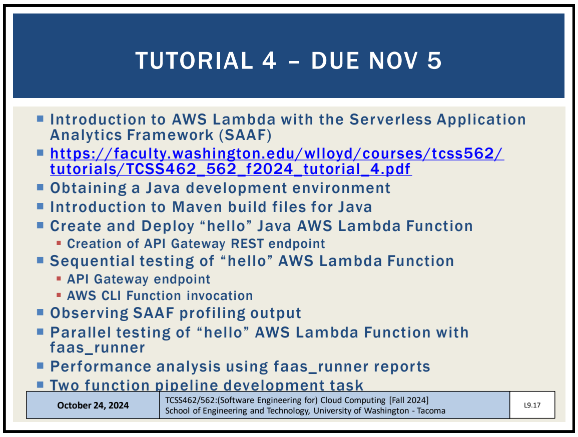
15



TUTORIAL 3 – DUE OCT 31

- Best Practices for Working with Virtual Machines on Amazon EC2
- https://faculty.washington.edu/wlloyd/courses/tcss562/tutorials/TCSS462_562_f2024_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

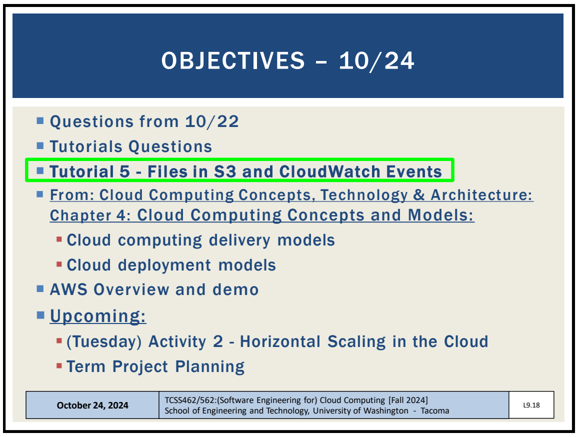
16



TUTORIAL 4 – DUE NOV 5

- Introduction to AWS Lambda with the Serverless Application Analytics Framework (SAAF)
- https://faculty.washington.edu/wlloyd/courses/tcss562/tutorials/TCSS462_562_f2024_tutorial_4.pdf
- Obtaining a Java development environment
- 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
- Performance analysis using faas_runner reports
- Two function pipeline development task

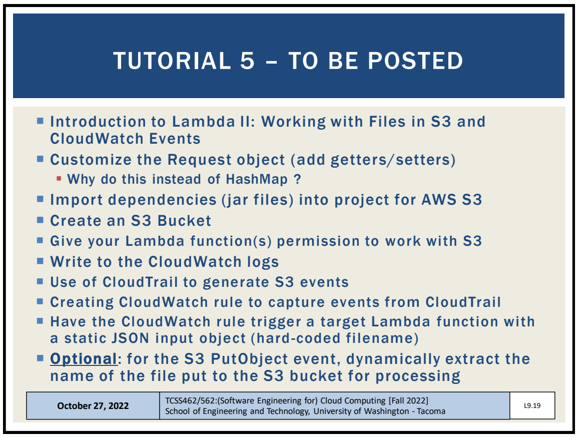
17



OBJECTIVES – 10/24

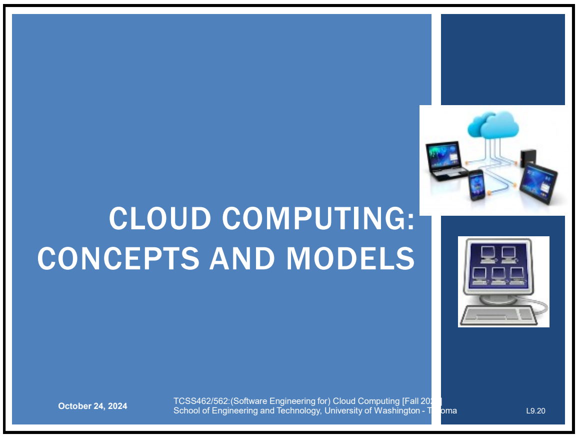
- Questions from 10/22
- Tutorials Questions
- **Tutorial 5 - Files in S3 and CloudWatch Events**
- From: Cloud Computing Concepts, Technology & Architecture: Chapter 4: Cloud Computing Concepts and Models:
  - Cloud computing delivery models
  - Cloud deployment models
- AWS Overview and demo
- Upcoming:
  - (Tuesday) Activity 2 - Horizontal Scaling in the Cloud
  - Term Project Planning

18

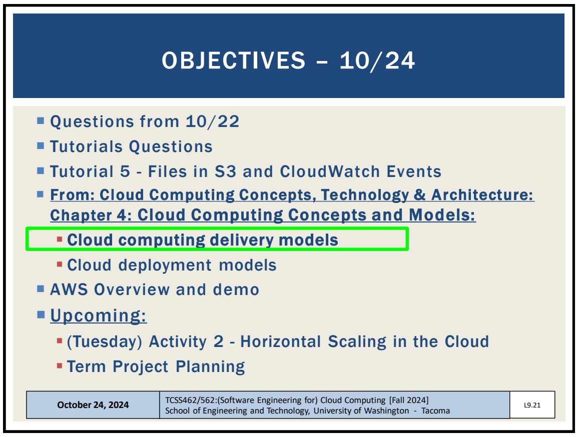


TUTORIAL 5 – TO BE POSTED

- Introduction to Lambda II: Working with Files in S3 and CloudWatch Events
- Customize the Request object (add getters/setters)
  - Why do this instead of HashMap ?
- Import dependencies (jar files) into project for AWS S3
- Create an S3 Bucket
- Give your Lambda function(s) permission to work with S3
- Write to the CloudWatch logs
- Use of CloudTrail to generate S3 events
- Creating CloudWatch rule to capture events from CloudTrail
- Have the CloudWatch rule trigger a target Lambda function with a static JSON input object (hard-coded filename)
- Optional: for the S3 PutObject event, dynamically extract the name of the file put to the S3 bucket for processing

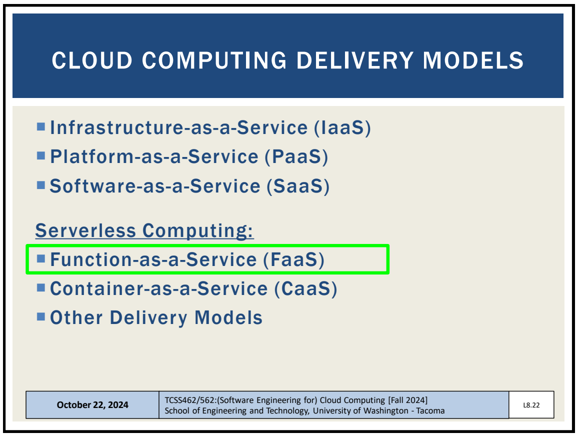


CLOUD COMPUTING: CONCEPTS AND MODELS



OBJECTIVES – 10/24

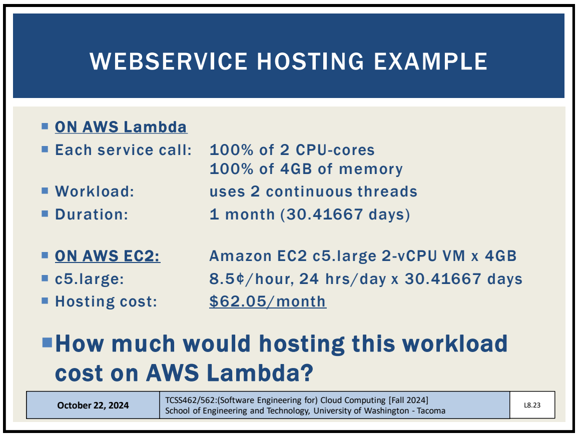
- Questions from 10/22
- Tutorials Questions
- Tutorial 5 - Files in S3 and CloudWatch Events
- **From: Cloud Computing Concepts, Technology & Architecture: Chapter 4: Cloud Computing Concepts and Models:**
  - **Cloud computing delivery models**
  - Cloud deployment models
- AWS Overview and demo
- Upcoming:
  - (Tuesday) Activity 2 - Horizontal Scaling in the Cloud
  - Term Project Planning



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

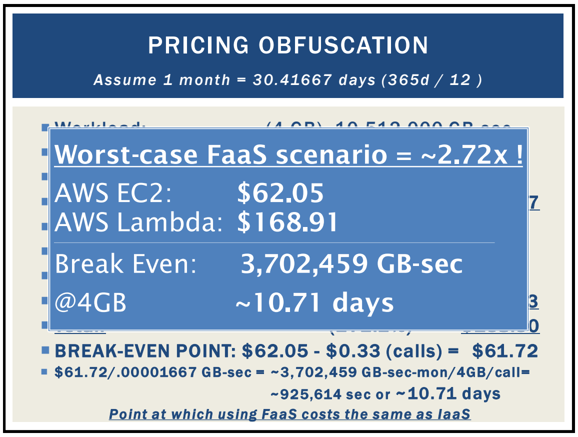


WEBSERVICE HOSTING EXAMPLE

- **ON AWS Lambda**
- Each service call: 100% of 2 CPU-cores
  100% of 4GB of memory
- Workload: uses 2 continuous threads
- Duration: 1 month (30.41667 days)

- **ON AWS EC2:** Amazon EC2 c5.large 2-vCPU VM x 4GB
- c5.large: 8.5¢/hour, 24 hrs/day x 30.41667 days
- Hosting cost: $62.05/month

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



PRICING OBFUSCATION

Assume 1 month = 30.41667 days (365d / 12)

Worst-case FaaS scenario = ~2.72x !

AWS EC2: $62.05
AWS Lambda: $168.91

Break Even: 3,702,459 GB-sec
@4GB ~10.71 days

- **BREAK-EVEN POINT: $62.05 - $0.33 (calls) = $61.72**
- **$61.72/.00001667 GB-sec = ~3,702,459 GB-sec-mon/4GB/call= ~925,614 sec or ~10.71 days**

Point at which using FaaS costs the same as IaaS

FAAS PRICING

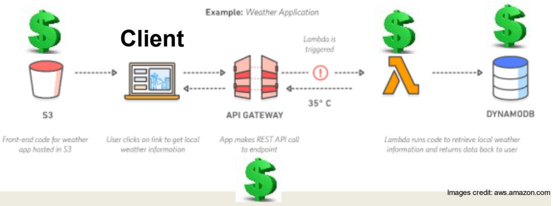
- Break-even point is the point where renting VMs or deploying to a serverless platform (e.g. Lambda) is exactly the same.
- Our example is for one month
- Could also consider one day, one hour, one minute
- **What factors influence the break-even point for an application running on AWS Lambda?**

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?
- Performance – what will it be?

VENDOR ARCHITECTURAL LOCK-IN

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



Images credit: aws.amazon.com

- Increased dependencies → increased hosting costs

PRICING OBFUSCATION

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

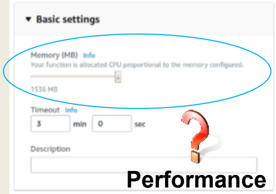
AWS Lambda Pricing

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

MEMORY RESERVATION QUESTION…

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

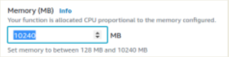


▼ Basic settings

Memory (MB) Info
Your function is allocated CPU proportional to the memory configured.
1536 MB

Timeout Info
3 min 0 sec

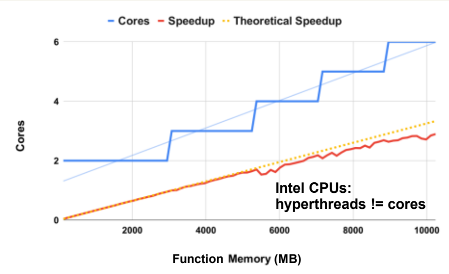
Description

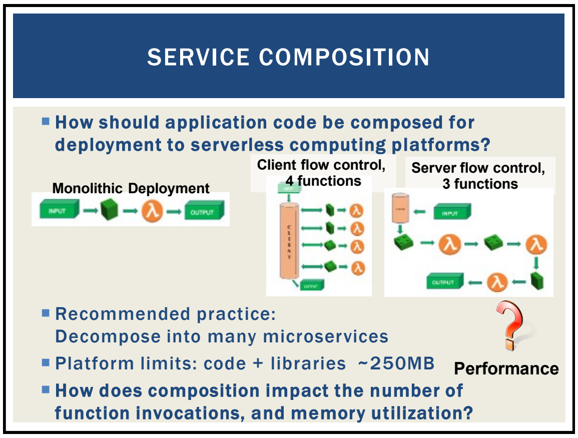


Memory (MB) Info
Your function is allocated CPU proportional to the memory configured.
10240 MB
Set memory to between 128 MB and 10240 MB

- **But how much memory do FaaS functions require?**

AWS LAMBDA COUPLES FUNCTION MEMORY TO CPU CORES & TIME SHARE

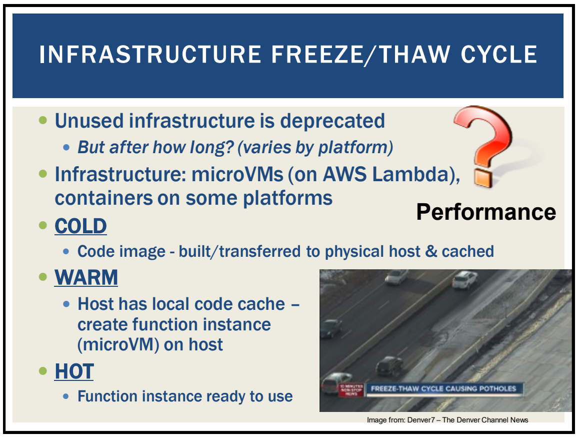




SERVICE COMPOSITION

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

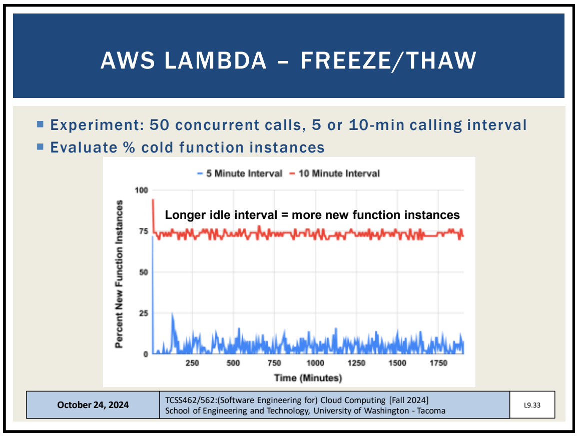
- Recommended practice: Decompose into many microservices
- Platform limits: code + libraries ~250MB
- How does composition impact the number of function invocations, and memory utilization?



INFRASTRUCTURE FREEZE/THAW CYCLE

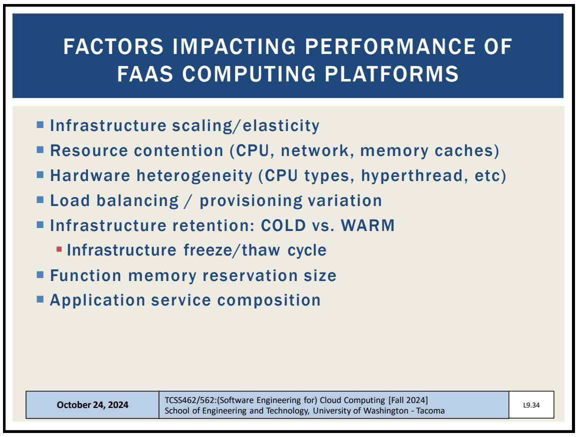
- Unused infrastructure is deprecated
  - *But after how long? (varies by platform)*
- Infrastructure: microVMs (on AWS Lambda), containers on some platforms
- **COLD**
  - Code image - built/transferred to physical host & cached
- **WARM**
  - Host has local code cache – create function instance (microVM) on host
- **HOT**
  - Function instance ready to use

Image from: Denver7 – The Denver Channel News



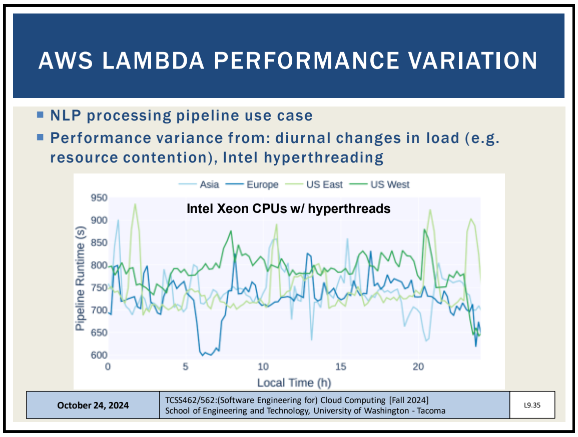
AWS LAMBDA – FREEZE/THAW

- Experiment: 50 concurrent calls, 5 or 10-min calling interval
- Evaluate % cold function instances



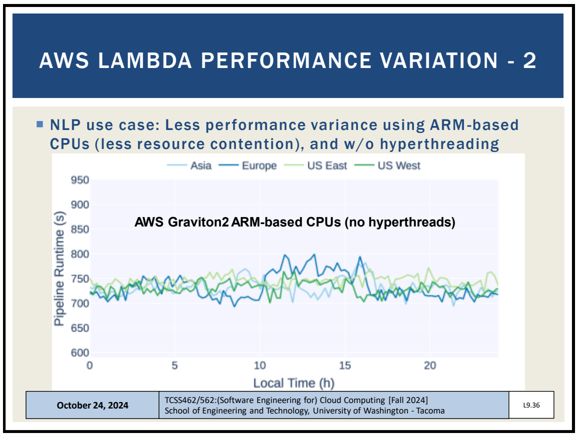
FACTORS IMPACTING PERFORMANCE OF FAAS COMPUTING PLATFORMS

- Infrastructure scaling/elasticity
- Resource contention (CPU, network, memory caches)
- Hardware heterogeneity (CPU types, hyperthread, etc)
- Load balancing / provisioning variation
- Infrastructure retention: COLD vs. WARM
  - Infrastructure freeze/thaw cycle
- Function memory reservation size
- Application service composition



AWS LAMBDA PERFORMANCE VARIATION

- NLP processing pipeline use case
- Performance variance from: diurnal changes in load (e.g. resource contention), Intel hyperthreading



AWS LAMBDA PERFORMANCE VARIATION - 2

- NLP use case: Less performance variance using ARM-based CPUs (less resource contention), and w/o hyperthreading

FUNCTION-AS-A-SERVICE

AWS Lambda Demo

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

CONTAINER-AS-A-SERVICE

- Cloud service model for deploying application containers (e.g. Docker containers) to the cloud
- Deploy containers without worrying about managing infrastructure:
  - Servers (virtual machines)
  - 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
  - Google Cloud Run
  - Azure Container Instances

CLOUD COMPUTING DELIVERY MODELS

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

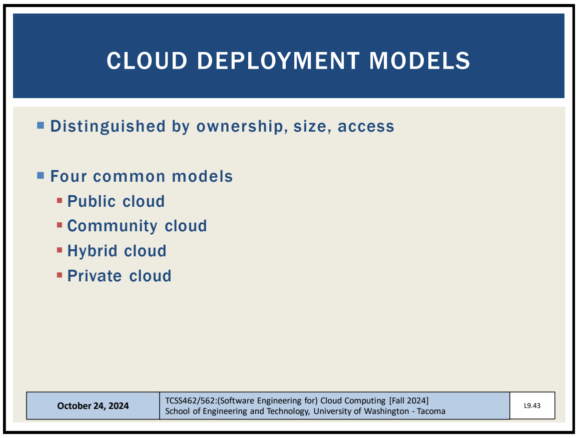
Serverless Computing:
- Function-as-a-Service (FaaS)
- Container-as-a-Service (CaaS)
- Other Delivery Models

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

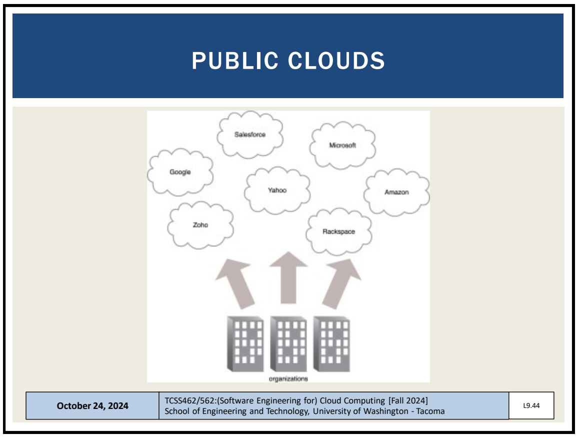
OBJECTIVES – 10/24

- Questions from 10/22
- Tutorials Questions
- Tutorial 5 - Files in S3 and CloudWatch Events
- From: Cloud Computing Concepts, Technology & Architecture: Chapter 4: Cloud Computing Concepts and Models:
  - Cloud computing delivery models
  - **Cloud deployment models**
- AWS Overview and demo
- Upcoming:
  - (Tuesday) Activity 2 - Horizontal Scaling in the Cloud
  - Term Project Planning

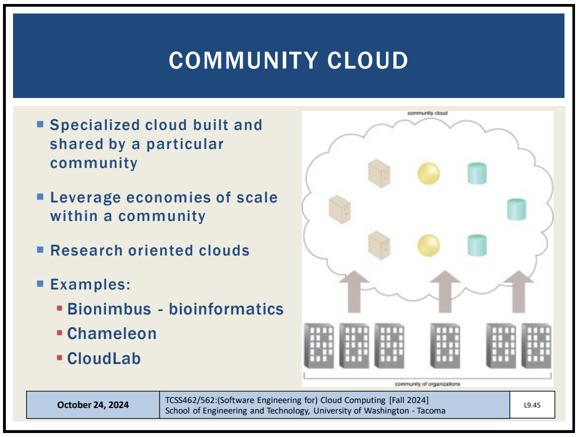


CLOUD DEPLOYMENT MODELS

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

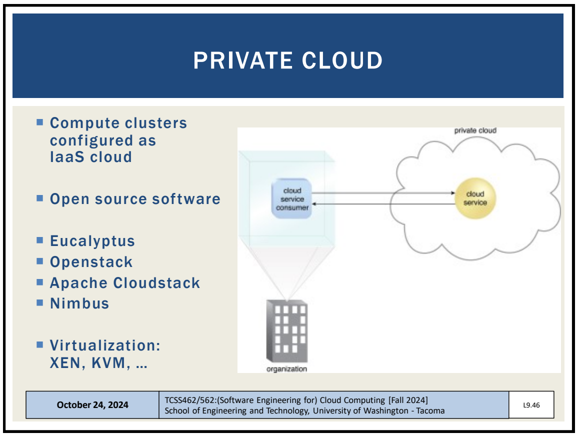


PUBLIC CLOUDS



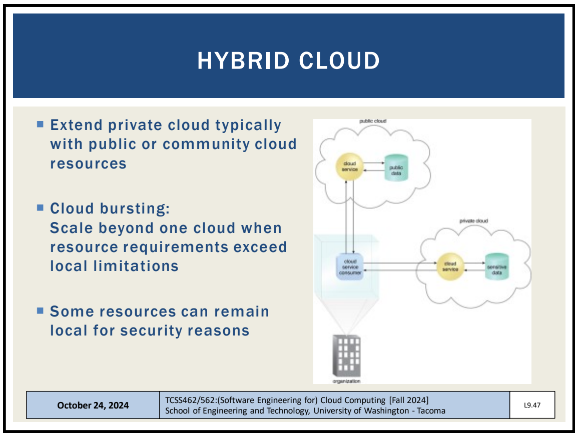
COMMUNITY CLOUD

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



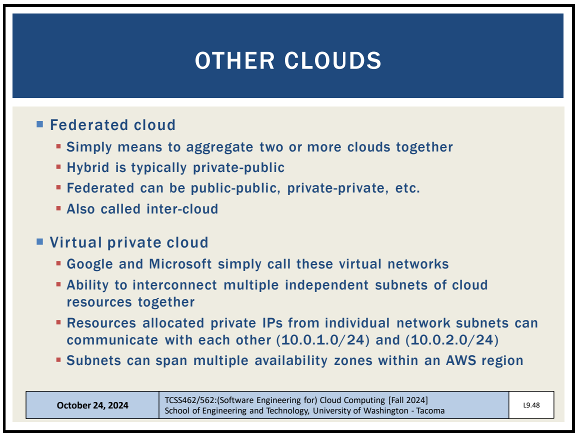
PRIVATE CLOUD

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



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



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

WE WILL RETURN AT 4:50 PM

OBJECTIVES – 10/24

- Questions from 10/22
- Tutorials Questions
- Tutorial 5 - Files in S3 and CloudWatch Events
- From: Cloud Computing Concepts, Technology & Architecture: Chapter 4: Cloud Computing Concepts and Models:
  - Cloud computing delivery models
  - Cloud deployment models
- AWS Overview and demo
- Upcoming:
  - (Tuesday) Activity 2 - Horizontal Scaling in the Cloud
  - Term Project Planning

AWS OVERVIEW AND DEMO

ONLINE CLOUD TUTORIALS

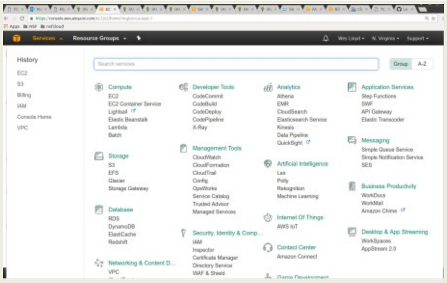
- From the eScience Institute @ UW Seattle: https://escience.washington.edu/
- Online cloud workshops
- Introduction to AWS, Azure, and Google Cloud
- Task: Deploying a Python DJANGO web application
- Self-guided workshop materials available online:
- https://cloudmaven.github.io/documentation/

- AWS Educate provides access to many online tutorials / learning resources:
- https://aws.amazon.com/education/awseducate/

LIST OF TOPICS

- AWS Management Console
- Elastic Compute Cloud (EC2)
- Instance Storage: Virtual Disks on VMs
- Elastic Block Store: Virtual Disks on VMs
- Elastic File System (EFS)
- Amazon Machine Images (AMIs)
- EC2 Paravirtualization
- EC2 Full Virtualization (hvm)
- EC2 Virtualization Evolution
- (VM) Instance Actions
- EC2 Networking
- EC2 Instance Metadata Service
- Simple Storage Service (S3)
- AWS Command Line Interface (CLI)
- Legacy / Service Specific CLIs
- AMI Tools
- Signing Certificates
- Backing up live disks
- Cost Savings Measures

AWS MANAGEMENT CONSOLE



AWS EC2

- Elastic Compute Cloud
- Instance types: https://ec2instances.info
  - **On demand instance** – full price
  - **Reserved instance** – contract based where customer guarantees VM rental for a fixed period of time (e.g. 1 year, 3 years, etc.) Deeper discounts with longer term commitments
  - **Spot instance** – portion of cloud capacity reserved for low cost instances, when demand exceeds supply instances are randomly terminated with 2 minute warning
    - Users can make diverse VM requests using different types, zones, regions, etc. to minimize instance terminations
    - Developers can design for failure because often only 1 or 2 VMs in a cluster fail at any given time. They then need to be replaced.
  - **Dedicated host** – reserved private HW (server)
  - Instance families - General, compute-optimized, memory-optimized, GPU, etc.

AWS EC2 - 2

- Storage types
  - Instance storage - ephemeral storage
    - Temporary disk volumes stored on disks local to the VM
    - Evolution: physical hard disk drives (HDDs)
    - Solid state drives (SSDs)
    - Non-volatile memory express (NVMe) drives (closer to DRAM speed)
  - EBS - Elastic block store
    - Remotely hosted disk volumes
  - EFS - Elastic file system
    - Shared file system based on network file system
    - VMs, Lambdas, Containers mount/interact with shared file system
    - Somewhat expensive

INSTANCE STORAGE

- Also called ephemeral storage
- Persisted using images saved to S3 (simple storage service)
  - ~2.3¢ per GB/month on S3
  - 5GB of free tier storage space on S3
- Requires "burning" an image
- Multi-step process:
  - Create image files
  - Upload chunks to S3
  - Register image
- Launching a VM
  - Requires downloading image components from S3, reassembling them… is potentially slow
- VMs with instance store backed root volumes not pause-able
- Historically root volume limited to 10-GB max– ***faster imaging…***

ELASTIC BLOCK STORE

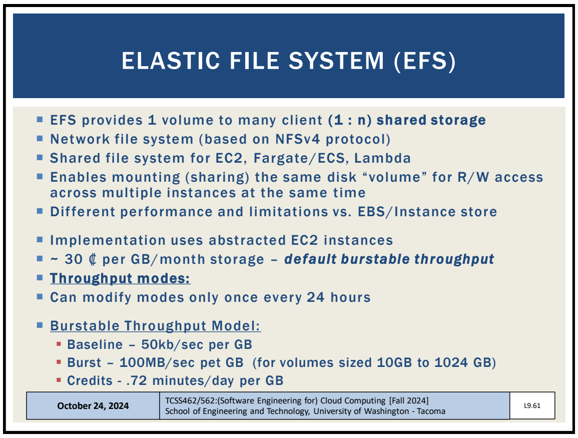
- EBS provides 1 drive to 1 virtual machine **(1 : 1)** **not shared)**
- EBS cost model is different than instance storage (uses S3)
  - ~10¢ per GB/month for General Purpose Storage (GP2)
  - ~8¢ per GB/month for General Purpose Storage (GP3)
  - 30GB of free tier storage space
- EBS provides "live" mountable volumes
  - Listed under volumes
  - **Data volumes**: can be mounted/unmounted to any VM, dynamically at any time
  - **Root volumes**: hosts OS files and acts as a boot device for VM
  - In Linux drives are linked to a mount point "directory"
- Snapshots back up EBS volume data to S3
  - Enables replication (required for horizontal scaling)
  - EBS volumes not actively used should be snapshotted, and deleted to save EBS costs…

EBS VOLUME TYPES - 2

- Metric: I/O Operations per Second (IOPS)
- **General Purpose 2 (GP2)**
  - 3 IOPS per GB, min 100 IOPS (<34GB), max of 16,000 IOPS
  - 250MB/sec throughput per volume
- **General Purpose 3** (GP3 – new Dec 2020)
  - Max 16,000 IOPS, Default 3,000 IOPS
  - GP2 requires creating a 1TB volume to obtain 3,000 IOPS
  - GP3 all volumes start at 3000 IOPS and 125 MB/s throughput
  - 1000 additional IOPS beyond 3000 is $5/month up to 16000 IOPS
  - 125 MB/s additional throughput is $5/month up to 1000 MB/s throughput

EBS VOLUME TYPES - 3

- **Provisioned IOPS (IO1)**
  - Legacy, associated with GP2
  - Allows user to create custom disk volumes where they pay for a specified IOPS and throughput
  - 32,000 IOPS, and 500 MB/sec throughput per volume MAX
- **Throughput Optimized HDD (ST1)**
  - Up to 500 MB/sec throughput
  - 4.5 ¢ per GB/month
- **Cold HDD (SC1)**
  - Up to 250 MB/sec throughput
  - 2.5 ¢ per GB/month
- **Magnetic**
  - Up to 90 MB/sec throughput per volume
  - 5 ¢ per GB/month

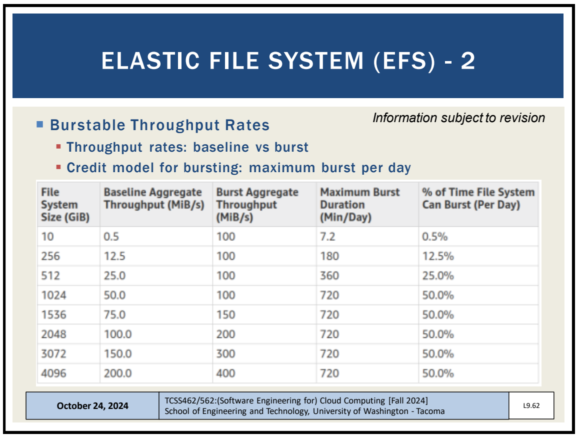


ELASTIC FILE SYSTEM (EFS)

- EFS provides 1 volume to many client **(1 : n) shared storage**
- Network file system (based on NFSv4 protocol)
- Shared file system for EC2, Fargate/ECS, Lambda
- Enables mounting (sharing) the same disk "volume" for R/W access across multiple instances at the same time
- Different performance and limitations vs. EBS/Instance store

- Implementation uses abstracted EC2 instances
- ~ 30 ₵ per GB/month storage – ***default burstable throughput***
- **Throughput modes:**
- Can modify modes only once every 24 hours

- Burstable Throughput Model:
  - Baseline – 50kb/sec per GB
  - Burst – 100MB/sec pet GB (for volumes sized 10GB to 1024 GB)
  - Credits - .72 minutes/day per GB

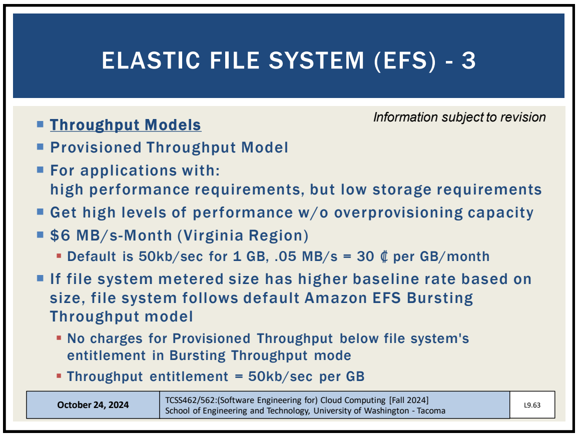


ELASTIC FILE SYSTEM (EFS) - 2

Information subject to revision

- Burstable Throughput Rates
  - Throughput rates: baseline vs burst
  - Credit model for bursting: maximum burst per day

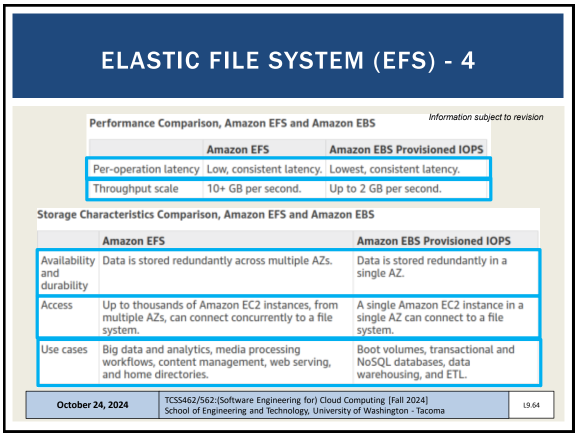
File System Size (GiB)	Baseline Aggregate Throughput (MiB/s)	Burst Aggregate Throughput (MiB/s)	Maximum Burst Duration (Min/Day)	% of Time File System Can Burst (Per Day)
10	0.5	100	7.2	0.5%
256	12.5	100	180	12.5%
512	25.0	100	360	25.0%
1024	50.0	100	720	50.0%
1536	75.0	150	720	50.0%
2048	100.0	200	720	50.0%
3072	150.0	300	720	50.0%
4096	200.0	400	720	50.0%



ELASTIC FILE SYSTEM (EFS) - 3

Information subject to revision

- **Throughput Models**
- Provisioned Throughput Model
- For applications with: high performance requirements, but low storage requirements
- Get high levels of performance w/o overprovisioning capacity
- $6 MB/s-Month (Virginia Region)
  - Default is 50kb/sec for 1 GB, .05 MB/s = 30 ₵ per GB/month
- If file system metered size has higher baseline rate based on size, file system follows default Amazon EFS Bursting Throughput model
  - No charges for Provisioned Throughput below file system's entitlement in Bursting Throughput mode
  - Throughput entitlement = 50kb/sec per GB



ELASTIC FILE SYSTEM (EFS) - 4

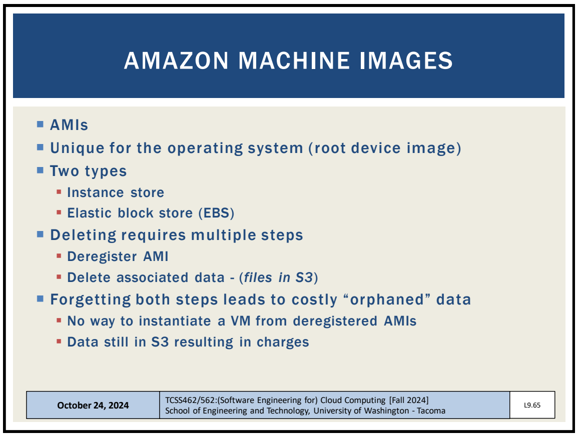
Information subject to revision

Performance Comparison, Amazon EFS and Amazon EBS

	Amazon EFS	Amazon EBS Provisioned IOPS
Per-operation latency	Low, consistent latency.	Lowest, consistent latency.
Throughput scale	10+ GB per second.	Up to 2 GB per second.

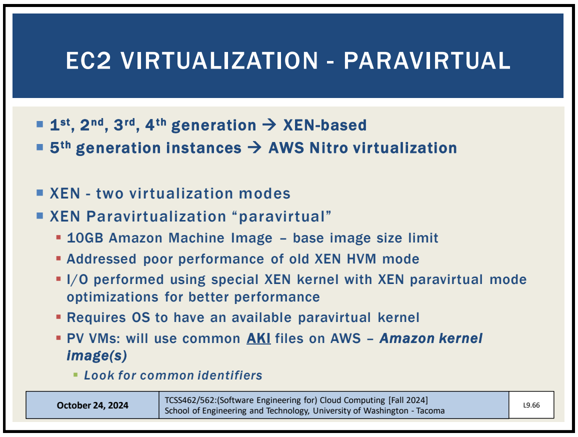
Storage Characteristics Comparison, Amazon EFS and Amazon EBS

	Amazon EFS	Amazon EBS Provisioned IOPS
Availability and durability	Data is stored redundantly across multiple AZs.	Data is stored redundantly in a single AZ.
Access	Up to thousands of Amazon EC2 instances, from multiple AZs, can connect concurrently to a file system.	A single Amazon EC2 instance in a single AZ can connect to a file system.
Use cases	Big data and analytics, media processing workflows, content management, web serving, and home directories.	Boot volumes, transactional and NoSQL databases, data warehousing, and ETL.



AMAZON MACHINE IMAGES

- AMIs
- Unique for the operating system (root device image)
- Two types
  - Instance store
  - Elastic block store (EBS)
- Deleting requires multiple steps
  - Deregister AMI
  - Delete associated data - (*files in S3*)
- Forgetting both steps leads to costly "orphaned" data
  - No way to instantiate a VM from deregistered AMIs
  - Data still in S3 resulting in charges



EC2 VIRTUALIZATION - PARAVIRTUAL

- **1st, 2nd, 3rd, 4th generation → XEN-based**
- **5th generation instances → AWS Nitro virtualization**

- XEN - two virtualization modes
- XEN Paravirtualization "paravirtual"
  - 10GB Amazon Machine Image – base image size limit
  - Addressed poor performance of old XEN HVM mode
  - I/O performed using special XEN kernel with XEN paravirtual mode optimizations for better performance
  - Requires OS to have an available paravirtual kernel
  - PV VMs: will use common **AKI** files on AWS – ***Amazon kernel image(s)***
    - *Look for common identifiers*

EC2 VIRTUALIZATION - HVM

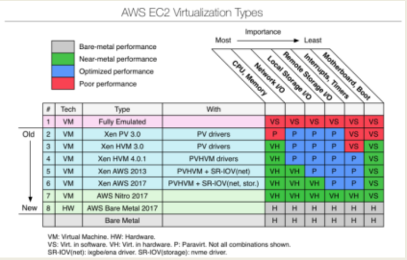
- XEN HVM mode
  - Full virtualization – no special OS kernel required
  - Computer entirely simulated
  - MS Windows runs in “hvm” mode
  - Allows work around: 10GB instance store root volume limit
  - Kernel is on the root volume (under /boot)
  - No AKIs (kernel images)
  - Commonly used today (*EBS-backed instances*)

EC2 VIRTUALIZATION - NITRO

- Nitro based on Kernel-based-virtual-machines
  - Stripped down version of Linux KVM hypervisor
  - Uses KVM core kernel module
  - I/O access has a direct path to the device
- Goal: provide indistinguishable performance from bare metal

EVOLUTION OF AWS VIRTUALIZATION

- From: http://www.brendangregg.com/blog/2017-11-29/aws-ec2-virtualization-2017.html


AWS EC2 Virtualization Types

#	Tech	Type	With	CPU, Memory	Network I/O	Local Storage I/O	Remote Storage I/O	Interrupts, Timers	Motherboard, Boot
1	VM	Fully Emulated		VS	VS	VS	VS	VS	VS
2	VM	Xen PV 3.0	PV drivers	P	P	P	P	VS	VS
3	VM	Xen HVM 3.0	PV drivers	VH	P	P	P	VS	VS
4	VM	Xen HVM 4.0.1	PVHVM drivers	VH	P	P	P	P	VS
5	VM	Xen AWS 2013	PVHVM + SR-IOV(net)	VH	VH	P	P	P	VS
6	VM	Xen AWS 2017	PVHVM + SR-IOV(net, stor.)	VH	VH	VH	P	P	VS
7	VM	AWS Nitro 2017		VH	VH	VH	VH	VH	VS
8	HW	AWS Bare Metal 2017		H	H	H	H	H	H
		Bare Metal		H	H	H	H	H	H

VM: Virtual Machine. HW: Hardware.
VS: Virt. in software. VH: Virt. in hardware. P: Paravirt. Not all combinations shown.
SR-IOV(net): ixgbe/ena driver. SR-IOV(storage): nvme driver.

INSTANCE ACTIONS

- Stop
  - Costs of “pausing” an instance
- Terminate
- Reboot

- Image management
- Creating an image
  - EBS (snapshot)
- Bundle image
  - Instance-store

EC2 INSTANCE: NETWORK ACCESS

- Public IP address
- Elastic IPs
  - Costs: in-use FREE, not in-use ~12 ¢/day
  - Not in-use (e.g. “paused” EBS-backed instances)

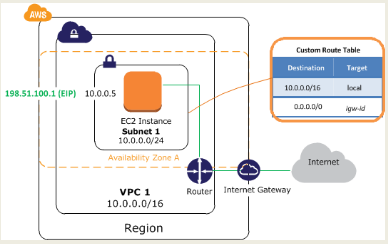
- Security groups
  - E.g. firewall

- Identity access management (IAM)
  - AWS accounts, groups

- VPC / Subnet / Internet Gateway / Router
- NAT-Gateway

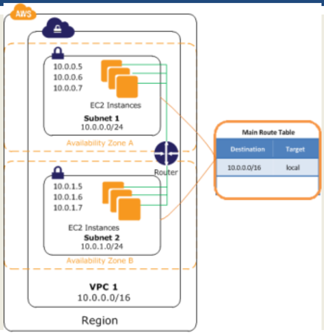
SIMPLE VPC

- Recommended when using Amazon EC2



Destination	Target
10.0.0.0/16	local
0.0.0.0/0	*igw-id*

VPC SPANNING AVAILABILITY ZONES



INSPECTING INSTANCE INFORMATION

- EC2 VMs run a local metadata service
- Can query instance metadata to self discover cloud configuration attributes

- Find your instance ID:

```
curl http://169.254.169.254/
curl http://169.254.169.254/latest/
curl http://169.254.169.254/latest/meta-data/
curl http://169.254.169.254/latest/meta-data/instance-id
; echo
```

- **`ec2-get-info`** command
- Python API that provides easy/formatted access to metadata

SIMPLE STORAGE SERVICE (S3)

- Key-value blob storage

- What is the difference vs. key-value stores (NoSQL DB)?

- Can mount an S3 bucket as a volume in Linux
  - Supports common file-system operations

- Provides eventual consistency

- Can store Lambda function state for life of container.

AWS CLI

- Launch Ubuntu 16.04 VM
  - **Instances | Launch Instance**

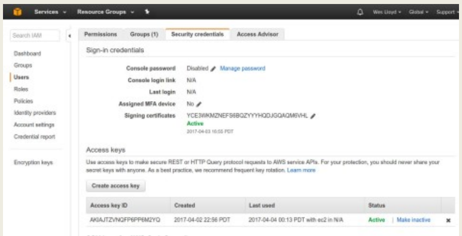
- Install the general AWS CLI
  - sudo apt install awscli

- Create config file

```
[default]
aws_access_key_id = <access key id>
aws_secret_access_key = <secret access key>
region = us-east-1
```

AWS CLI - 2

- Creating access keys: **IAM | Users | Security Credentials | Access Keys | Create Access Keys**



Sign-in credentials

Console password: Disabled
Console login link: N/A
Last login: N/A
Assigned MFA device: No
Signing certificates: YCE3WKMZNEFS6BQZYYYHQDJGQAQM6VHL
Active
2017-04-03 16:55 PDT

Access keys

Use access keys to make secure REST or HTTP Query protocol requests to AWS service APIs. For your protection, you should never share your secret keys with anyone. As a best practice, we recommend frequent key rotation. Learn more

Access key ID	Created	Last used	Status
AKIAJTZVNQFP6PP6M2YQ	2017-04-02 22:56 PDT	2017-04-04 00:13 PDT with ec2 in N/A	Active | Make inactive

AWS CLI - 3

- Export the config file
  - Add to /home/ubuntu/.bashrc

```
export AWS_CONFIG_FILE=$HOME/.aws/config
```

- Try some commands:
  - `aws help`
  - `aws command help`
  - `aws ec2 help`
  - `aws ec2 describes-instances --output text`
  - `aws ec2 describe-instances --output json`
  - `aws s3 ls`
  - `aws s3 ls vmscaleruw`

LEGACY / SERVICE SPECIFIC CLI(S)

- `sudo apt install ec2-api-tools`
- Provides more concise output
- Additional functionality

- Define variables in .bashrc or another sourced script:
- `export AWS_ACCESS_KEY={your access key}`
- `export AWS_SECRET_KEY={your secret key}`

- `ec2-describe-instances`
- `ec2-run-instances`
- `ec2-request-spot-instances`

- EC2 management from Java:
- http://docs.aws.amazon.com/AWSJavaSDK/latest/javadoc/index.html

- Some AWS services have separate CLI installable by package

AMI TOOLS

- Amazon Machine Images tools
- For working with disk volumes
- Can create live copies of any disk volume
  - Your local laptop, ec2 root volume (EBS), ec2 ephemeral disk

- Installation: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ami-tools-commands.html

- AMI tools reference:
- https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ami-tools-commands.html

- Some functions may require private key & certificate files

PRIVATE KEY AND CERTIFICATE FILE

- Install openssl package on VM

generate private key file
$openssl genrsa 2048 > mykey.pk

generate signing certificate file
$openssl req -new -x509 -nodes -sha256 -days 36500 -key mykey.pk -outform PEM -out signing.cert

- Add signing.cert to IAM | Users | Security Credentials | *- - new signing certificate - -*

- From: http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/set-up-ami-tools.html?icmpid=docs_iam_console#ami-tools-create-certificate

PRIVATE KEY, CERTIFICATE FILE

- These files, combined with your AWS_ACCESS_KEY and AWS_SECRET_KEY and AWS_ACCOUNT_ID enable you to publish new images from the CLI

- Objective:
1. Configure VM with software stack
2. Burn new image for VM replication **(horizontal scaling)**

- An alternative to bundling volumes and storing in S3 is to use a containerization tool such as Docker. . .

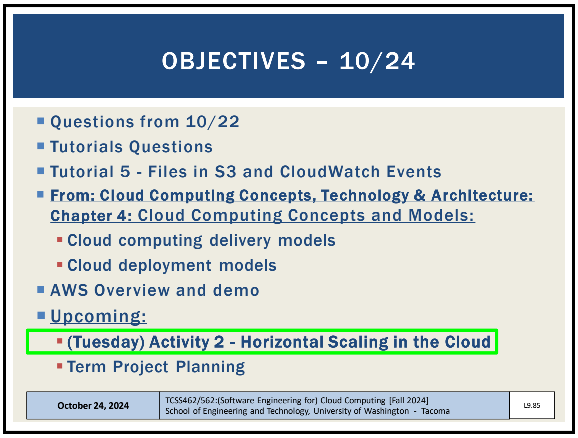
- Create image script . . .

SCRIPT: CREATE A NEW INSTANCE STORE IMAGE FROM LIVE DISK VOLUME

```
image=$1
echo "Burn image $image"
echo "$image" > image.id
mkdir /mnt/tmp
AWS_KEY_DIR=/home/ubuntu/.aws
export EC2_URL=http://ec2.amazonaws.com
export S3_URL=https://s3.amazonaws.com
export EC2_PRIVATE_KEY=${AWS_KEY_DIR}/mykey.pk
export EC2_CERT=${AWS_KEY_DIR}/signing.cert
export AWS_USER_ID={your account id}
export AWS_ACCESS_KEY={your aws access key}
export AWS_SECRET_KEY={your aws secret key}
ec2-bundle-vol -s 5000 -u ${AWS_USER_ID} -c ${EC2_CERT} -k ${EC2_PRIVATE_KEY}
--ec2cert /etc/ec2/amitools/cert-ec2.pem --no-inherit -r x86_64 -p $image -i
/etc/ec2/amitools/cert-ec2.pem
cd /tmp
ec2-upload-bundle -b tcss562 -m $image.manifest.xml -a ${AWS_ACCESS_KEY} -s
${AWS_SECRET_KEY}  --url http://s3.amazonaws.com --location US
ec2-register tcss562/$image.manifest.xml --region us-east-1 --kernel aki-
88aa75e1
```

COST SAVINGS MEASURES

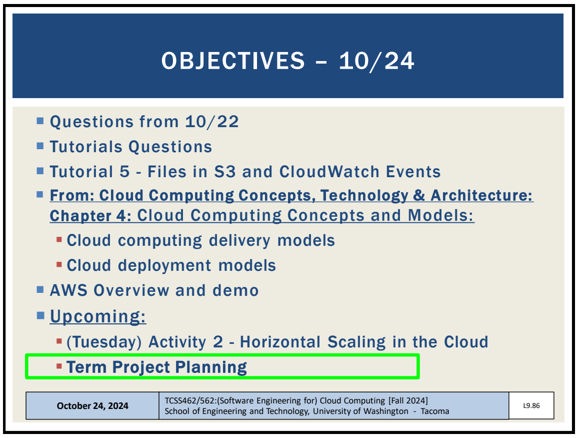
- ***From Tutorial 3:***
- #1: ALWAYS USE SPOT INSTANCES FOR COURSE/RESEARCH RELATED PROJECTS
- #2: NEVER LEAVE AN EBS VOLUME IN YOUR ACCOUNT THAT IS NOT ATTACHED TO A RUNNING VM
- #3: BE CAREFUL USING PERSISTENT REQUESTS FOR SPOT INSTANCES
- #4: TO SAVE/PERSIST DATA, USE EBS SNAPSHOTS AND THEN
- #5: DELETE EBS VOLUMES FOR TERMINATED EC2 INSTANCES.
- #6: UNUSED SNAPSHOTS AND UNUSED EBS VOLUMES SHOULD BE PROMPTLY DELETED !!
- #7: USE PERSISTENT SPOT REQUESTS AND THE “STOP” FEATURE TO PAUSE VMS DURING SHORT BREAKS



OBJECTIVES – 10/24

- Questions from 10/22
- Tutorials Questions
- Tutorial 5 - Files in S3 and CloudWatch Events
- **From: Cloud Computing Concepts, Technology & Architecture: Chapter 4:** Cloud Computing Concepts and Models:
  - Cloud computing delivery models
  - Cloud deployment models
- AWS Overview and demo
- Upcoming:
  - **(Tuesday) Activity 2 - Horizontal Scaling in the Cloud**
  - Term Project Planning

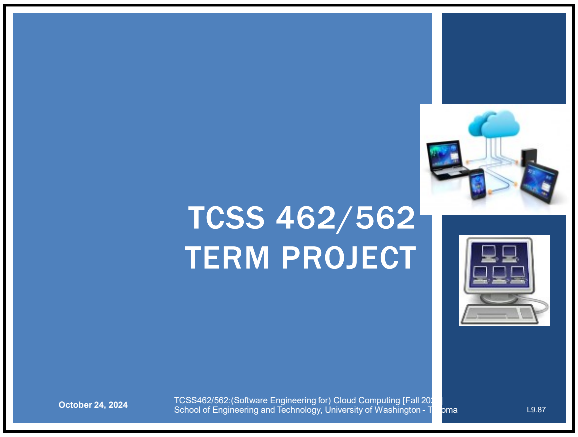
85



OBJECTIVES – 10/24

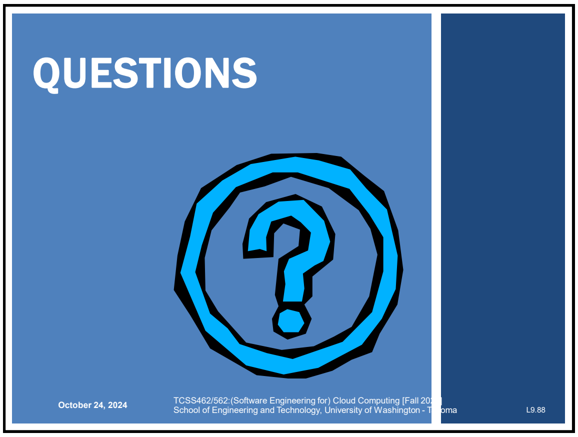
- Questions from 10/22
- Tutorials Questions
- Tutorial 5 - Files in S3 and CloudWatch Events
- **From: Cloud Computing Concepts, Technology & Architecture: Chapter 4:** Cloud Computing Concepts and Models:
  - Cloud computing delivery models
  - Cloud deployment models
- AWS Overview and demo
- Upcoming:
  - (Tuesday) Activity 2 - Horizontal Scaling in the Cloud
  - **Term Project Planning**

86



TCSS 462/562
TERM PROJECT

87



QUESTIONS

88