TCSS 562: SOFTWARE ENGINEERING FOR CLOUD COMPUTING

Introduction to Cloud Computing

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

TR 5:00-7:00 PM

1

OFFICE HOURS – FALL 2021

- **Tuesdays:**
  - 4:00 to 4:30 pm - CP 229
  - 7:15 to 7:45+ pm – ONLINE via Zoom
- **Thursdays**
  - 4:15 to 4:45 pm – ONLINE via Zoom
  - 7:15 to 7:45+ pm – ONLINE via Zoom
- Or email for appointment
- Zoom Link sent as Canvas Announcement

> Office Hours set based on Student Demographics survey feedback

October 19, 2021	TCSS562: Software Engineering for Cloud Computing [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L6.2

2

OBJECTIVES – 10/19

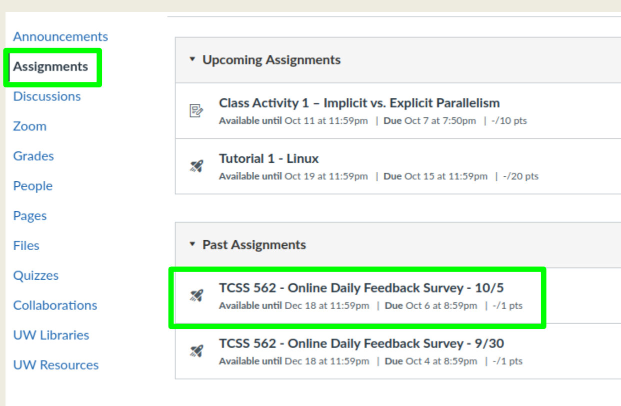
- Questions from 10/14
- Tutorial 3 - Best Practices with EC2
- From: Cloud Computing Concepts, Technology & Architecture: Chapter 3: Understanding Cloud Computing
  - Terminology
  - Benefits of cloud adoption
  - Risks of cloud adoption
- From: Cloud Computing Concepts, Technology & Architecture: Chapter 4: Cloud Computing Concepts and Models:
  - Roles and boundaries
  - Cloud characteristics
- 2nd hour:
  - TCSS 562 Term Project
  - Team Planning

October 19, 2021	TCSS562:Software Engineering for Cloud Computing [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L6.3

3

ONLINE DAILY FEEDBACK SURVEY

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



October 19, 2021	TCSS562: Software Engineering for Cloud Computing [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L6.4

4

TCSS 562 - Online Daily Feedback Survey - 10/5

Started: Oct 7 at 1:13am

Quiz Instructions

Question 1 0.5 pts

On a scale of 1 to 10, please classify your perspective on material covered in today's class:

1	2	3	4	5	6	7	8	9	10
Mostly Review To Me				Equal New and Review					Mostly New to Me

Question 2 0.5 pts

Please rate the pace of today's class:

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

MATERIAL / PACE

- Please classify your perspective on material covered in today's class (30 respondents):
- 1-mostly review, 5-equal new/review, 10-mostly new
- **Average – 6.30 (↑ - *previous 5.81*)**

- Please rate the pace of today's class:
- 1-slow, 5-just right, 10-fast
- **Average – 5.33 (↑ - previous 5.04)**

FEEDBACK FROM 10/14

- ***Where did you get the AWS architecture diagrams from in your slides?***
  - For the term project these were created using a Linux program called dia
  - AWS specific symbols were downloaded as a package and add to dia

OBJECTIVES – 10/19

- **Questions from 10/14**
- Tutorial 3 - Best Practices with EC2
- **From: Cloud Computing Concepts, Technology & Architecture: Chapter 3: Understanding Cloud Computing**
  - Terminology
  - Benefits of cloud adoption
  - Risks of cloud adoption
- **From: Cloud Computing Concepts, Technology & Architecture:** Chapter 4: Cloud Computing Concepts and Models:
  - Roles and boundaries
  - Cloud characteristics
- **2nd hour:**
  - TCSS 562 Term Project
  - Team Planning

OBJECTIVES – 10/19

- **Questions from 10/14**
- Tutorial 3 - Best Practices with EC2
- **From: Cloud Computing Concepts, Technology & Architecture: Chapter 3: Understanding Cloud Computing**
  - Terminology
  - Benefits of cloud adoption
  - Risks of cloud adoption
- From: Cloud Computing Concepts, Technology & Architecture: Chapter 4: Cloud Computing Concepts and Models:
  - Roles and boundaries
  - Cloud characteristics
- **2nd hour:**
  - TCSS 562 Term Project
  - Team Planning

KEY TERMINOLOGY

- **On-Premise Infrastructure**
  - ?
- **Cloud Provider**
  - ?
- **Cloud Consumer**
  - ?
- **Scaling**
  - **Vertical scaling**
    - Scale up: ?
    - Scale down: ?
  - **Horizontal scaling**
    - Scale out: ?
    - Scale in: ?

KEY TERMINOLOGY

- **On-Premise Infrastructure**
  - Local server infrastructure not configured as a cloud
- **Cloud Provider**
  - Corporation or private organization responsible for maintaining cloud
- **Cloud Consumer**
  - User of cloud services
- **Scaling**
  - **Vertical scaling**
    - Scale up: increase resources of a single virtual server
    - Scale down: decrease resources of a single virtual server
  - **Horizontal scaling**
    - Scale out: increase number of virtual servers
    - Scale in: decrease number of virtual servers

KEY TERMINOLOGY - 2

- **Cloud services:**
  - Broad array of resources accessible “as-a-service”
  - Categorized as Infrastructure (IaaS), Platform (PaaS), Software (SaaS)
- **Service-level-agreements (SLAs):**
  - Establish expectations for: uptime, security, availability, reliability, and performance

OBJECTIVES – 10/19

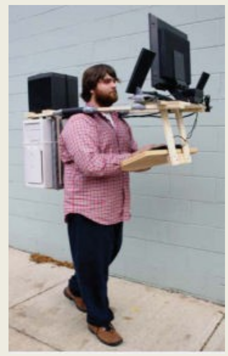
- **Questions from 10/14**
- Tutorial 3 - Best Practices with EC2
- **From: Cloud Computing Concepts, Technology & Architecture: Chapter 3: Understanding Cloud Computing**
  - Terminology
  - Benefits of cloud adoption
  - Risks of cloud adoption
- **From: Cloud Computing Concepts, Technology & Architecture: Chapter 4: Cloud Computing Concepts and Models:**
  - Roles and boundaries
  - Cloud characteristics
- **2nd hour:**
  - TCSS 562 Term Project
  - Team Planning

GOALS AND BENEFITS

- **Cloud providers**
  - Leverage economies of scale through mass-acquisition and management of large-scale IT resources
  - Locate datacenters to optimize costs where electricity is low
- **Cloud consumers**
  - Key business/accounting difference:
  - **Cloud computing enables anticipated capital expenditures to be replaced with operational expenditures**
  - Operational expenditures always scale with the business
  - Eliminates need to invest in server infrastructure based on anticipated business needs
  - Businesses become more agile and lower their financial risks by eliminating large capital investments in physical infrastructure

CLOUD BENEFITS - 2

- On demand access to pay-as-you-go resources on a short-term basis (less commitment)
- Ability to acquire "unlimited" computing resources on demand when required for business needs
- Ability to add/remove IT resources at a fine-grained level
- Abstraction of server infrastructure so applications deployments are not dependent on specific locations, hardware, etc.
  - The cloud has made our software deployments more agile...



CLOUD BENEFITS - 3

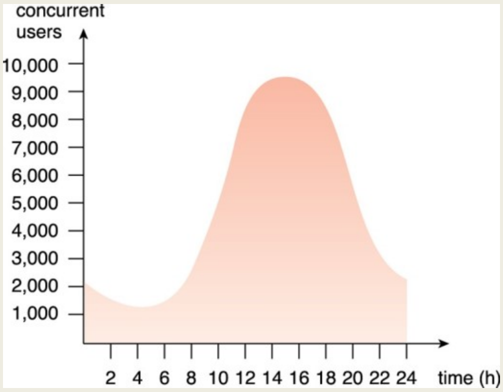
- Example: Using 100 servers for 1 hour costs the same as using 1 server for 100 hours
- Rosetta Protein Folding: Working with a UW-Tacoma graduate student, deployed science model across 5,900 compute cores on Amazon for 2-days...
- ***What is the cost to purchase 5,900 compute cores?***
- Dell Server purchase example:
  20 cores on 2 servers for ~$4,478...
- Using this ratio 5,900 cores costs $1.3 million (purchase only)

Gene Wilder, Charlie and the Chocolate Factory

17

CLOUD BENEFITS

- Increased scalability
  - Example demand over a 24-hour day →
- Increased availability
- Increased reliability



18

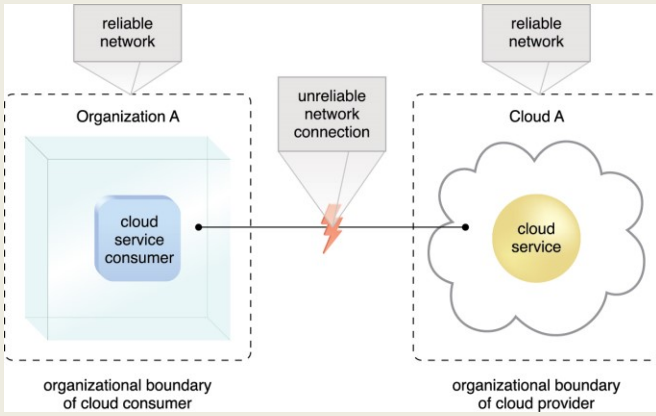
OBJECTIVES – 10/19

- **Questions from 10/14**
- Tutorial 3 - Best Practices with EC2
- **From: Cloud Computing Concepts, Technology & Architecture: Chapter 3: Understanding Cloud Computing**
  - Terminology
  - Benefits of cloud adoption
  - Risks of cloud adoption
- **From: Cloud Computing Concepts, Technology & Architecture: Chapter 4: Cloud Computing Concepts and Models:**
  - Roles and boundaries
  - Cloud characteristics
- **2nd hour:**
  - TCSS 562 Term Project
  - Team Planning

CLOUD ADOPTION RISKS

- **Increased security vulnerabilities**
  - Expansion of trust boundaries now include the external cloud
  - Security responsibility shared with cloud provider
- **Reduced operational governance / control**
  - Users have less control of physical hardware
  - Cloud user does not directly control resources to ensure quality-of-service
  - Infrastructure management is abstracted
  - Quality and stability of resources can vary
  - Network latency costs and variability

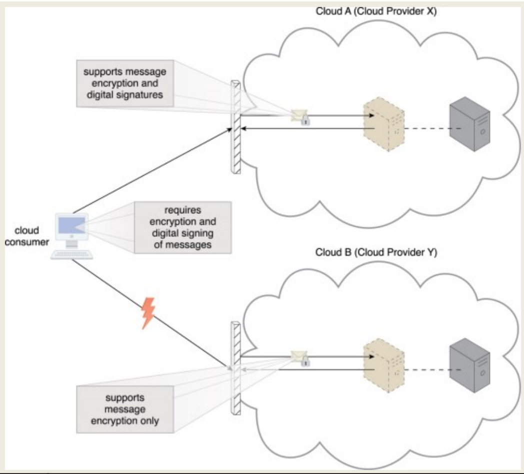
NETWORK LATENCY COSTS



CLOUD RISKS - 2

- **Performance monitoring of cloud applications**
  - Cloud metrics (AWS cloudwatch) support monitoring cloud infrastructure (network load, CPU utilization, I/O)
  - Performance of cloud applications depends on the health of aggregated cloud resources working together
  - User must monitor this aggregate performance
- **Limited portability among clouds**
  - Early cloud systems have significant “vendor” lock-in
  - Common APIs and deployment models are slow to evolve
  - Operating system containers help make applications more portable, but containers still must be deployed
- Geographical issues
  - Abstraction of cloud location leads to legal challenges with respect to laws for data privacy and storage

CLOUD: VENDOR LOCK-IN



OBJECTIVES – 10/19

- **Questions from 10/14**
- Tutorial 3 - Best Practices with EC2
- **From: Cloud Computing Concepts, Technology & Architecture: Chapter 3: Understanding Cloud Computing**
  - Terminology
  - Benefits of cloud adoption
  - Risks of cloud adoption
- **From: Cloud Computing Concepts, Technology & Architecture:** Chapter 4: Cloud Computing Concepts and Models:
  - Roles and boundaries
  - Cloud characteristics
- **2nd hour:**
  - TCSS 562 Term Project
  - Team Planning

CLOUD COMPUTING: CONCEPTS AND MODELS

October 19, 2021
TCSS562: Software Engineering for Cloud Computing [Fall 2021]
School of Engineering and Technology, University of Washington - Tacoma
L6.25

25

OBJECTIVES – 10/19

- **Questions from 10/14**
- Tutorial 3 - Best Practices with EC2
- **From: Cloud Computing Concepts, Technology & Architecture: Chapter 3: Understanding Cloud Computing**
  - Terminology
  - Benefits of cloud adoption
  - Risks of cloud adoption
- **From: Cloud Computing Concepts, Technology & Architecture: Chapter 4: Cloud Computing Concepts and Models:**
  - Roles and boundaries
  - Cloud characteristics
- **2nd hour:**
  - TCSS 562 Term Project
  - Team Planning

October 19, 2021	TCSS562:Software Engineering for Cloud Computing [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L6.26

26

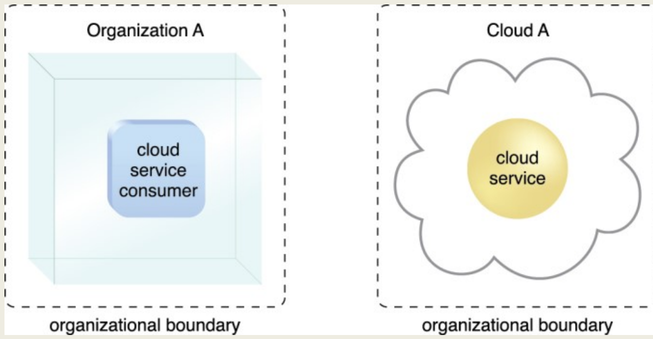
ROLES

- **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)

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

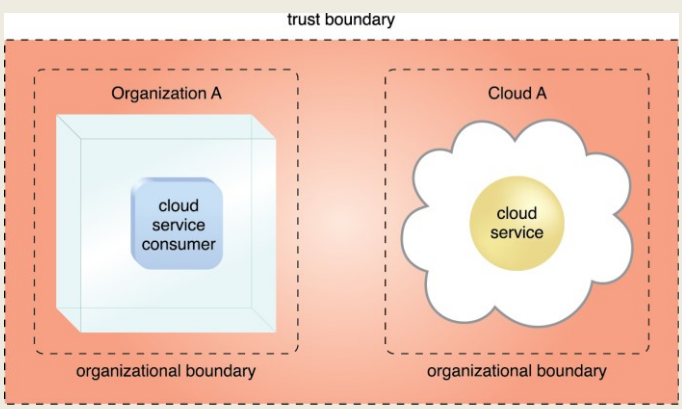
ORGANIZATION BOUNDARY



October 19, 2021	TCSS562: Software Engineering for Cloud Computing [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L6.29

29

TRUST BOUNDARY



October 19, 2021	TCSS562: Software Engineering for Cloud Computing [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L6.30

30

OBJECTIVES – 10/19

- **Questions from 10/14**
- Tutorial 3 - Best Practices with EC2
- **From: Cloud Computing Concepts, Technology & Architecture: Chapter 3: Understanding Cloud Computing**
  - Terminology
  - Benefits of cloud adoption
  - Risks of cloud adoption
- **From: Cloud Computing Concepts, Technology & Architecture: Chapter 4: Cloud Computing Concepts and Models:**
  - Roles and boundaries
  - Cloud characteristics
- **2nd hour:**
  - TCSS 562 Term Project
  - Team Planning

October 19, 2021	TCSS562:Software Engineering for Cloud Computing [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L6.31

31

CLOUD CHARACTERISTICS

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

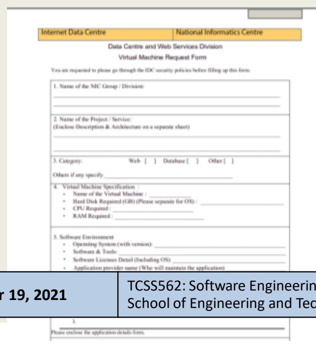
- Assessing these features helps measure the value offered by a given cloud service or platform

October 19, 2021	TCSS562: Software Engineering for Cloud Computing [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L6.32

32

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



Internet Data Centre | National Informatics Centre

Data Centre and Web Services Division

Virtual Machine Request Form

You are requested to please go through the IDC security policies before filling up this form.

1. Name of the NIC Group / Division:

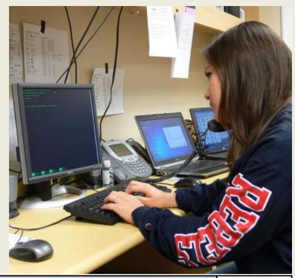
2. Name of the Project / Service:
(Enclose Description & Architecture on a separate sheet)

3. Category: Web [ ] Database [ ] Other [ ]
Others if any specify

4. Virtual Machine Specification :
   - Name of the Virtual Machine :
   - Hard Disk Required (GB) (Please separate for OS) :
   - CPU Required :
   - RAM Required :

5. Software Environment
   - Operating System (with version):
   - Software & Tools:
   - Software Licenses Detail (Including OS)
   - Application provider name (Who will maintain the application)

Please enclose the application details form.



33

UBIQUITOUS ACCESS

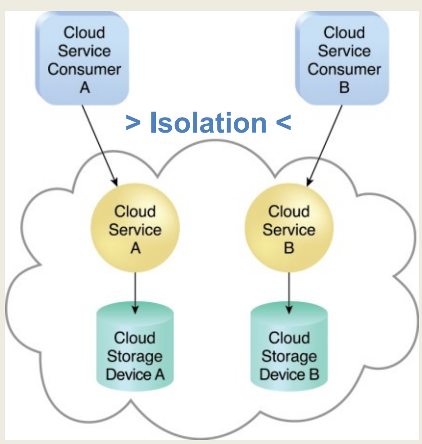
- Cloud services are widely accessible
- Public cloud: internet accessible
- Private cloud: throughout segments of a company's intranet
- 24/7 availability

34

MULTITENANCY

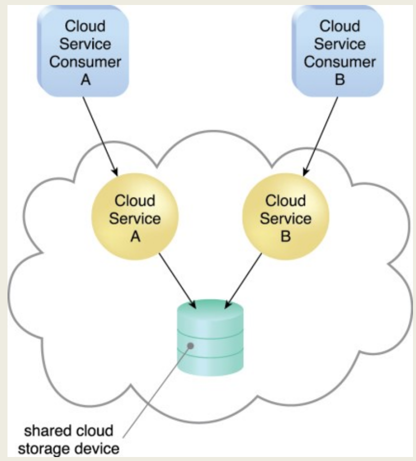
- Cloud providers pool resources together to share them with many users
- Serve multiple cloud service consumers
- IT resources can be dynamically assigned, reassigned based on demand
- Multitenancy can lead to performance variation

SINGLE TENANT MODEL



MULTITENANT MODEL

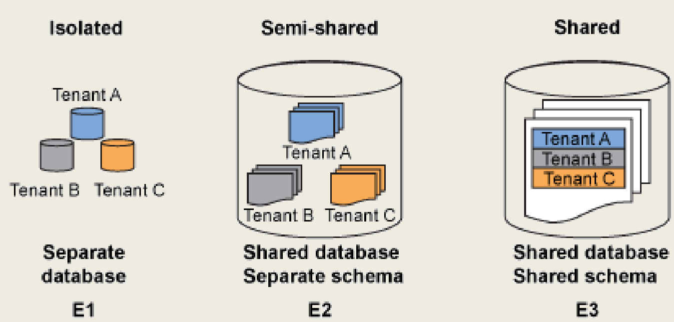
- Resource is "multiplexed" and share amongst multiple users
- Goal is to increase utilization
- Often server resources are underutilized
- There are many "sunk costs" whether usage is 0% or 100%
- Cloud computing tries to maximize "sunk cost" investments through **multi-tenancy**



October 19, 2021	TCSS562: Software Engineering for Cloud Computing [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L6.37

37

MULTITENANT DATABASE



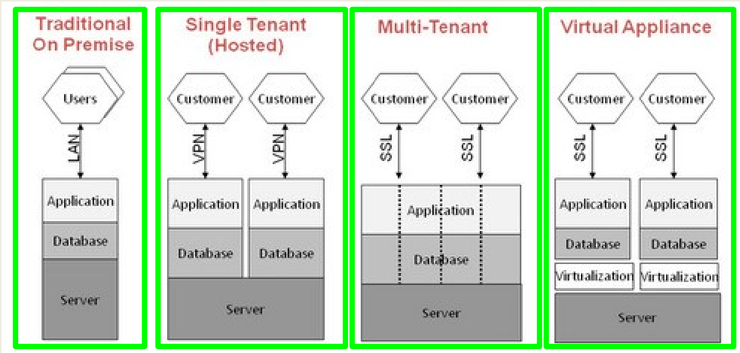
- Many users on a single database instance
- ***What issues may occur when sharing a single database instance?***

October 19, 2021	TCSS562: Software Engineering for Cloud Computing [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L6.38

38

MULTITENANCY OF RESOURCES

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

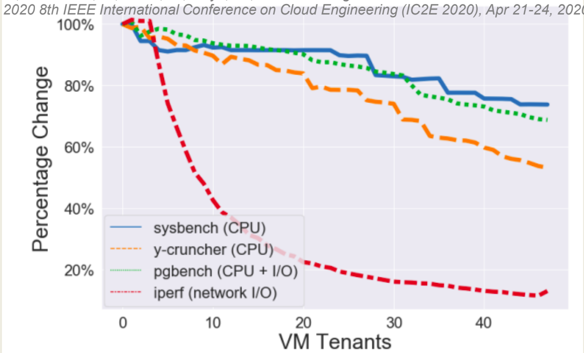


39

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 !!

40

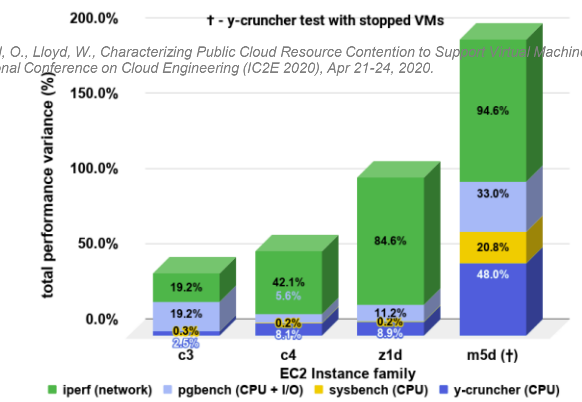
RESOURCE CONTENTION FROM MUTLI-TENANCY - 2

- 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) →



41

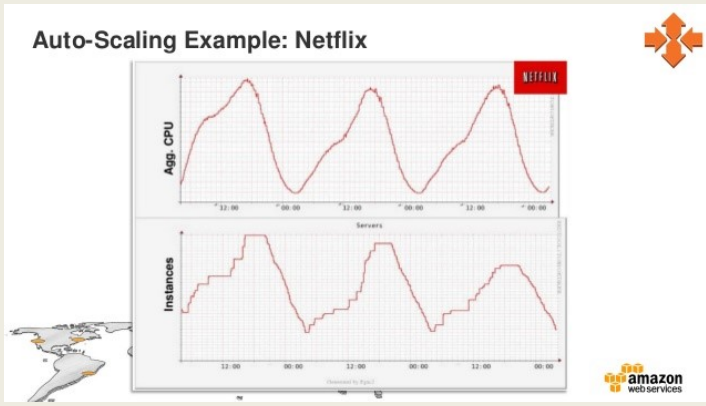
ELASTICITY

- 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

42

PREDICTABLE DEMAND

- AWS EC2 Scaling Example:



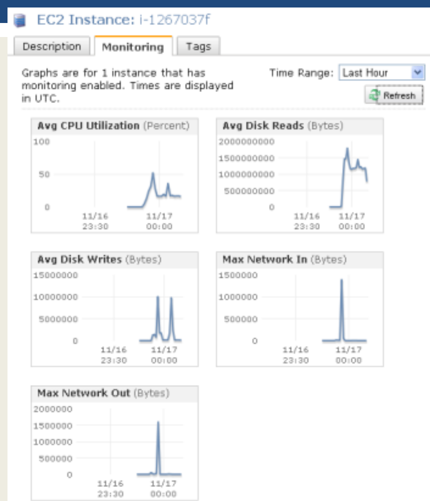
From: Kejariwal, A., 2013, March. Techniques for optimizing cloud footprint. In 2013 IEEE Int. Conf. on Cloud Engineering (IC2E), pp. 258-268.

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)

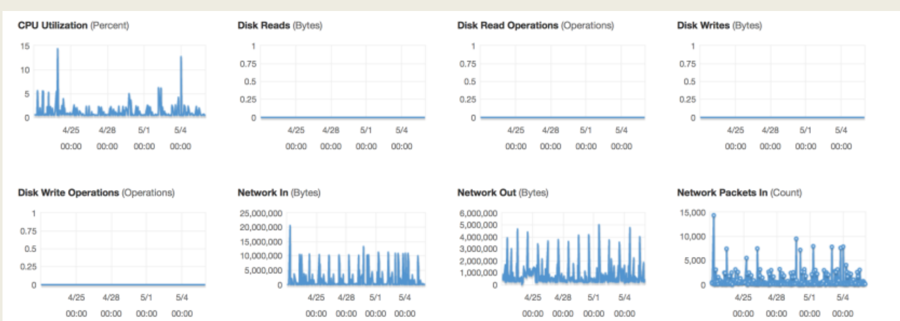
- Not all measurements are for billing
- Some measurements can support auto-scaling
- For example CPU utilization

EC2 CLOUDWATCH METRICS



45

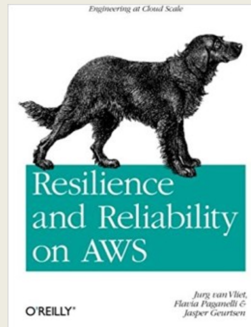
EC2 CLOUDWATCH METRICS



46

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 19, 2021 | TCSS562: Software Engineering for Cloud Computing [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma | L6.47

47

WE WILL RETURN AT 6:10 PM

48

OBJECTIVES – 10/19

- **Questions from 10/14**
- Tutorial 3 - Best Practices with EC2
- **From: Cloud Computing Concepts, Technology & Architecture: Chapter 3: Understanding Cloud Computing**
  - Terminology
  - Benefits of cloud adoption
  - Risks of cloud adoption
- **From: Cloud Computing Concepts, Technology & Architecture: Chapter 4: Cloud Computing Concepts and Models:**
  - Roles and boundaries
  - Cloud characteristics
- **2nd hour:**
  - TCSS 562 Term Project
  - Team Planning

49

TCSS 562 TERM PROJECT

50

TCSS 562 TERM PROJECT

- Build a serverless cloud native application
- Application provides case study to investigate architecture/design trade-offs
  - Application provides a vehicle to compare and contrast one or more trade-offs
- Alternate 1: Cloud Computing Related Research Project
- Alternate 2: Literature Survey/Gap Analysis

**- as an individual project*

DESIGN TRADE-OFFS

- **Service composition**
  - Switchboard architecture:
    - compose services in single package
    - Address COLD Starts
    - Infrastructure Freeze/Thaw cycle of AWS Lambda (FaaS)
  - Full service isolation (each service is deployed separately)
- **Application flow control**
  - client-side, step functions, server-side controller, asynchronous hand-off
- **Programming Languages**
- **Alternate FaaS Platforms**

DESIGN TRADE-OFFS - 2

- **Alternate Cloud Services (e.g. databases, queues, etc.)**
  - Compare alternate data backends for data processing pipeline
- **Performance variability (by hour, day, week, and host location)**
  - Deployments (to different zones, regions)
- **Service abstraction**
  - Abstract one or more services with cloud abstraction middleware: Apache libcloud, apache jcloud; make code cross-cloud; measure overhead

OTHER PROJECT IDEAS

- Elastic File System (EFS)
  Performance & Scalability Evaluation
- Docker container image integration with AWS Lambda – performance & scalability
- Resource contention study using CpuSteal metric
  - Investigate the degree of CpuSteal on FaaS platforms
    - What is the extent? Min, max, average
    - When does it occur?
    - Does it correlate with performance outcomes?
    - Is contention self-inflicted?
- & others

SERVERLESS APPLICATIONS

- **Extract Transform Load Data Processing Pipeline**
  - * >>>This is the STANDARD project<<< *
  - Batch-oriented data
  - Stream-oriented data
- **Image Processing Pipeline**
  - Apply series of filters to images
- **Stream Processing Pipeline**
  - Data conversion, filtering, aggregation, archival storage
  - What throughput (records/sec) can Lambda ingest directly?
  - Comparison with AWS Kinesis Data Streams and DB backend:
  - https://aws.amazon.com/getting-started/hands-on/build-serverless-real-time-data-processing-app-lambda-kinesis-s3-dynamodb-cognito-athena/
  - Kinesis data streams claims multiple GB/sec throughput
    - What is the cost difference?

SERVERLESS APPLICATIONS - 2

- **Map-Reduce Style Application**
  - Function 1: split data into chunks, usually sequentially
  - Function 2: process individual chunks concurrently (in parallel)
    - Data processing is considered to be Embarrassingly Parallel
  - Function 3: aggregate and summarize results
- **Image Classification Pipeline**
  - Deploy pretrained image classifiers in a multi-stage pipeline
- **Machine Learning**
  - Multi-stage inferencing pipelines
  - Natural Language Processing (NLP) pipelines
  - Training (?)

AWS LAMBDA PLATFORM LIMITATIONS

- Maximum 10 GB memory per function instance
- Maximum 15-minutes execution per function instance
- Access to 500 MB of temporary disk space for local I/O
- Access up to 6 vCPUs depending on memory reservation size
- 1,000 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

- See: https://docs.aws.amazon.com/lambda/latest/dg/gettingstarted-limits.html

EXTRACT TRANSFORM LOAD DATA PIPELINE

- Service 1: **TRANSFORM**

- Read CSV file, perform some transformations
- Write out new CSV file

- Service 2: **LOAD**

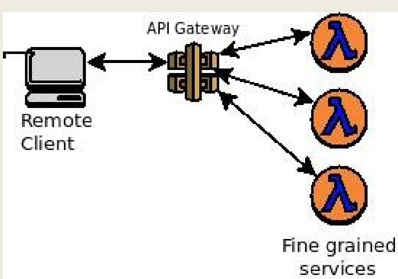
- Read CSV file, load data into relational database
- Cloud DB (AWS Aurora), or local DB (Derby/SQLite)
  - Derby DB and/or SQLite code examples to be provided in Java

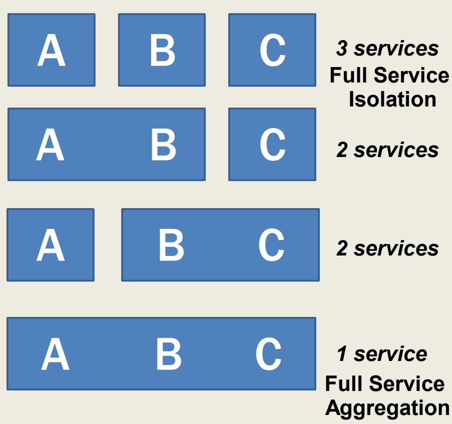
EXTRACT TRANSFORM LOAD DATA PIPELINE - 2

- Service 3: **QUERY**

- Using relational database, apply filter(s) and/or functions to aggregate data to produce sums, totals, averages
- Output aggregations as JSON

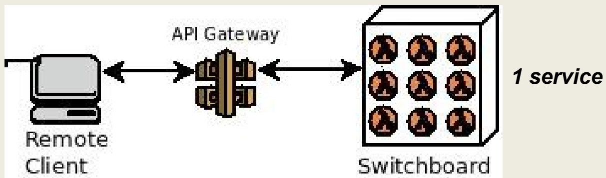
SERVICE COMPOSITION





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

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

APPLICATION FLOW CONTROL

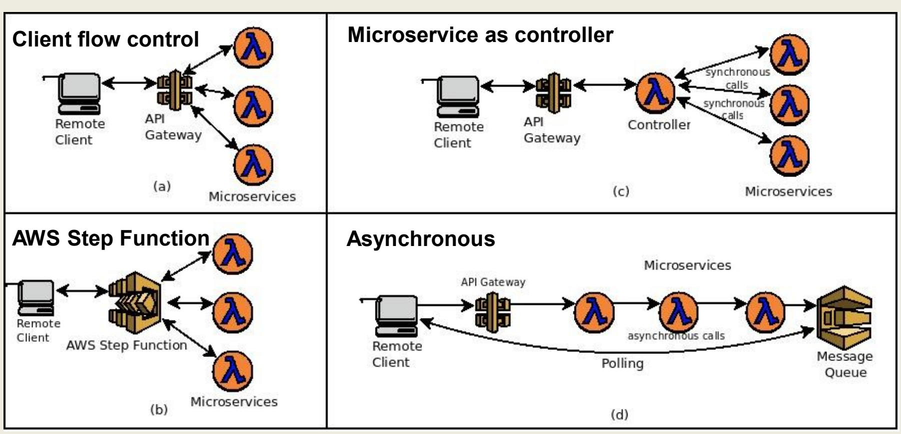
- **Serverless Computing:**
- AWS Lambda (FAAS: Function-as-a-Service)
- Provides HTTP/REST like web services
- Client/Server paradigm

- **Synchronous web service:**
- Client calls service
- Client blocks (freezes) and waits for server to complete call
- Connection is maintained in the "OPEN" state
- Problematic if service runtime is long!
  - Connections are notoriously dropped
  - System timeouts reached
- Client can't do anything while waiting unless using threads

APPLICATION FLOW CONTROL - 2

- **Asynchronous web service**
- Client calls service
- Server responds to client with OK message
- Client closes connection
- Server performs the work associated with the service
- Server posts service result in an external data store
  - AWS: S3, SQS (queueing service), SNS (notification service)

APPLICATION FLOW CONTROL - 3



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

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

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

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…

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

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*:
  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

CPUSTEAL 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?

71

QUESTIONS

72