TCSS 562: SOFTWARE ENGINEERING FOR CLOUD COMPUTING

Cloud Computing Concepts and Models - II

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

2

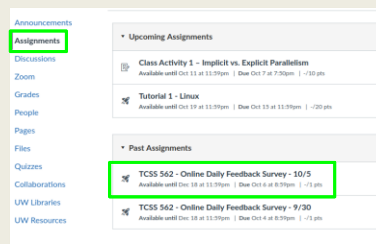
OBJECTIVES – 10/26

- **Questions from 10/21**
- Tutorial 3 - Best Practices with EC2
- Tutorial 4 – Intro to FaaS – AWS Lambda
- **From: Cloud Computing Concepts, Technology & Architecture: Chapter 4: Cloud Computing Concepts and Models:**
  - ..
  - Cloud delivery models
  - Cloud deployment models
- **2nd hour:**
  - TCSS 562 Term Project
  - Team Planning - Breakout Rooms

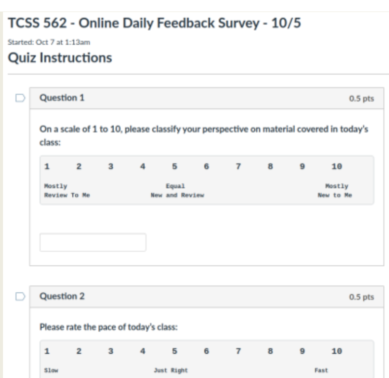
3

ONLINE DAILY FEEDBACK SURVEY

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



4



5

MATERIAL / PACE

- Please classify your perspective on material covered in today's class (24 respondents):
- 1-mostly review, 5-equal new/review, 10-mostly new
- **Average – 6.71 (↑ - *previous 6.00*)**
- Please rate the pace of today's class:
- 1-slow, 5-just right, 10-fast
- **Average – 5.75 (↑ - previous 5.54)**

6

FEEDBACK FROM 10/21

- ***Could you give a high level overview of how you could build a completely serverless web application?***
  What would the application architecture look like?
  How would the static web hosting work?
  How would the client interact with the serverless backend?
- AWS Lambda can generate dynamic content as needed
- Store static content in persistent data stores (S3, RDS)
- API Gateway can serve static content on request
  - Backend provider: cloudfront or Lambda
- Persist user session state using data stores (S3, etc.)
  - Provide users unique session key (unique identifier)
  - Session key to subsequent FaaS function calls
  - Session keys can enable any FaaS function to query user state from a relational DB or object store

7

FEEDBACK - 2

- An example serverless web application is discussed on the following blog:
- https://medium.com/hootsuite-engineering/developing-a-fully-serverless-web-app-84ff897de8a1
- Services used:
- **Lambda** — serverless compute
- **IAM & Cognito** — permissions and authentication
- **DynamoDB** — data storage
- **S3** — static website hosting and file storage
- **SES** — sending emails
- **Cloudwatch** — debugging and scheduling events
- **Cloudfront** — hosting the production single page application (SPA)
- **API Gateway** — http endpoints to front our Lambda functions

8

OBJECTIVES – 10/26

- **Questions from 10/21**
- Tutorial 3 - Best Practices with EC2
- Tutorial 4 – Intro to FaaS – AWS Lambda
- **From: Cloud Computing Concepts, Technology & Architecture: Chapter 4:** Cloud Computing Concepts and Models:
  - ..
  - Cloud delivery models
  - Cloud deployment models
- **2nd hour:**
  - TCSS 562 Term Project
  - Team Planning - Breakout Rooms

9

OBJECTIVES – 10/26

- **Questions from 10/21**
- Tutorial 3 - Best Practices with EC2
- Tutorial 4 – Intro to FaaS – AWS Lambda
- **From: Cloud Computing Concepts, Technology & Architecture: Chapter 4:** Cloud Computing Concepts and Models:
  - ..
  - Cloud delivery models
  - Cloud deployment models
- **2nd hour:**
  - TCSS 562 Term Project
  - Team Planning - Breakout Rooms

10

CLOUD COMPUTING: CONCEPTS AND MODELS

11

OBJECTIVES – 10/26

- **Questions from 10/21**
- Tutorial 3 - Best Practices with EC2
- Tutorial 4 – Intro to FaaS – AWS Lambda
- **From: Cloud Computing Concepts, Technology & Architecture: Chapter 4:** Cloud Computing Concepts and Models:
  - ..
  - Cloud delivery models
  - Cloud deployment models
- **2nd hour:**
  - TCSS 562 Term Project
  - Team Planning - Breakout Rooms

12

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 26, 2021 | TCSS562: Software Engineering for Cloud Computing [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma | L8.13

13

SERVERLESS COMPUTING

Introducing Cloud 2.0

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

14

SERVERLESS COMPUTING

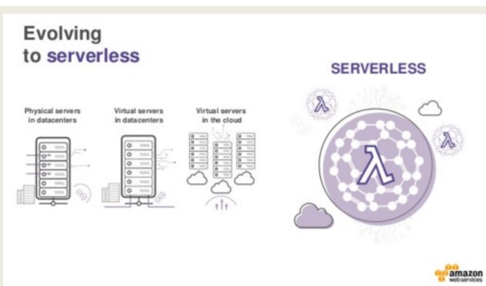
15

SERVERLESS COMPUTING

October 26, 2021 | TCSS562: Software Engineering for Cloud Computing [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma | L8.16

16

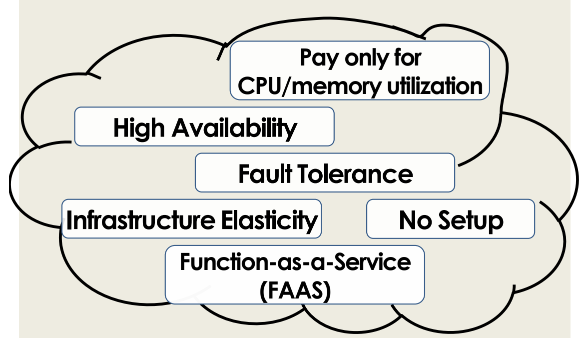
SERVERLESS COMPUTING - 2



October 26, 2021 | TCSS562: Software Engineering for Cloud Computing [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma | L8.17

17

SERVERLESS COMPUTING



18

SERVERLESS COMPUTING

Why Serverless Computing?

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

…they are built into the platform

19

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

20

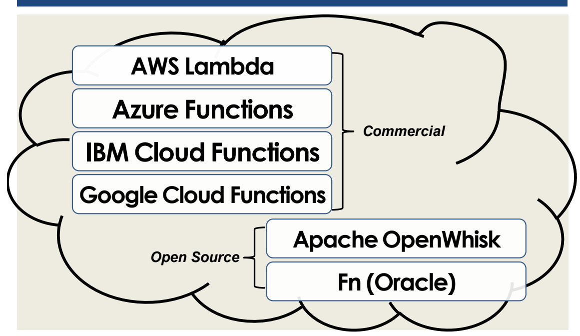
SERVERLESS VS. FAAS

- 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…

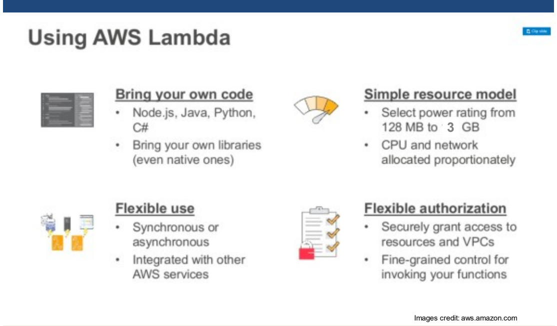
21

FAAS PLATFORMS



22

AWS LAMBDA



23

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

24

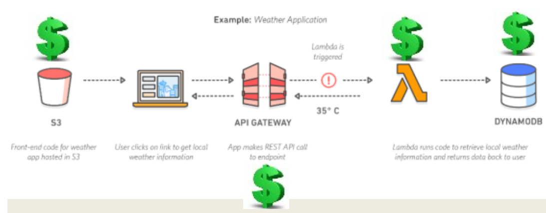
FAAS PLATFORMS - 3

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

25

CLOUD NATIVE SOFTWARE ARCHITECTURE

- Every service with a different pricing model



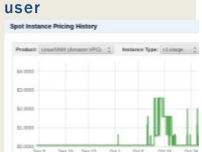
26

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 →



27

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

28

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: c4.large: $72/month, 10¢/hour, 24 hrs/day x 30 days
- **How much would hosting this workload cost on AWS Lambda?**

29

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**

30

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?**

31

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

32

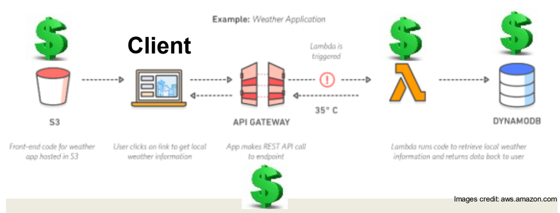
FAAS CHALLENGES

- 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?

33

VENDOR ARCHITECTURAL LOCK-IN

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



- Increased dependencies → increased hosting costs

34

PRICING OBFUSCATION

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

AWS Lambda Pricing

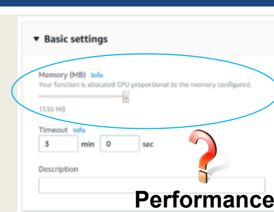
FREE TIER: first 1,000,000 function calls/month → FREE
first 400,000 GB-sec/month → FREE

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

35

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…"*

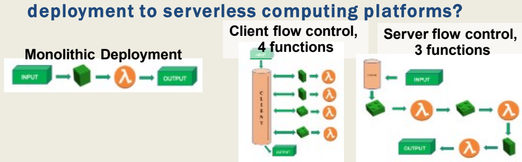


- **But how much memory do model services require?**

36

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

37

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

38

FUNCTION-AS-A-SERVICE

AWS Lambda Demo

39

39

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 26, 2021 | TCSS562: Software Engineering for Cloud Computing [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma | L8.40

40

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

October 26, 2021 | TCSS562: Software Engineering for Cloud Computing [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma | L8.41

41

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 26, 2021 | TCSS562: Software Engineering for Cloud Computing [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma | L8.42

42

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

43

OBJECTIVES – 10/26

- **Questions from 10/21**
- Tutorial 3 - Best Practices with EC2
- Tutorial 4 – Intro to FaaS – AWS Lambda
- **From: Cloud Computing Concepts, Technology & Architecture:** Chapter 4: Cloud Computing Concepts and Models:
  - ..
  - Cloud delivery models
  - Cloud deployment models
- **2nd hour:**
  - TCSS 562 Term Project
  - Team Planning - Breakout Rooms

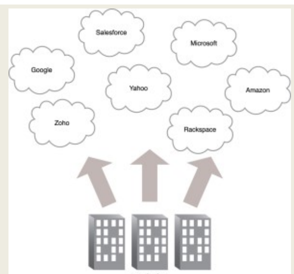
44

CLOUD DEPLOYMENT MODELS

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

45

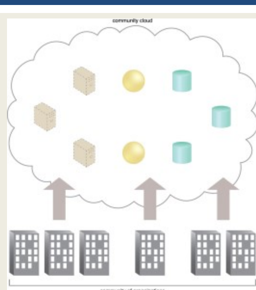
PUBLIC CLOUDS



46

COMMUNITY CLOUD

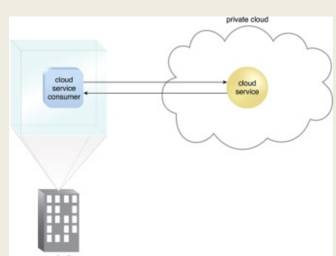
- Specialized cloud built and shared by a particular community
- Leverage economies of scale within a community
- Research oriented clouds
- Examples:
  - Bionimbus - bioinformatics
  - Chameleon
  - CloudLab



47

PRIVATE CLOUD

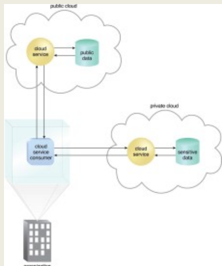
- Compute clusters configured as IaaS cloud
- Open source software
- Eucalyptus
- Openstack
- Apache Cloudstack
- Nimbus
- Virtualization: XEN, KVM, …



48

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 26, 2021 | TCSS562: Software Engineering for Cloud Computing [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma | L8.49

49

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 26, 2021 | TCSS562: Software Engineering for Cloud Computing [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma | L8.50

50

WE WILL RETURN AT ~6:30 PM

51

OBJECTIVES – 10/26

- **Questions from 10/21**
- Tutorial 3 - Best Practices with EC2
- Tutorial 4 – Intro to FaaS – AWS Lambda
- **From: Cloud Computing Concepts, Technology & Architecture: Chapter 4: Cloud Computing Concepts and Models:**
  - ..
  - Cloud delivery models
  - Cloud deployment models
- **2nd hour:**
  - TCSS 562 Term Project
  - Team Planning - Breakout Rooms

October 26, 2021 | TCSS562:Software Engineering for Cloud Computing [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma | L8.52

52

TCSS 562 TERM PROJECT

October 26, 2021 | TCSS562: Software Engineering for Cloud Computing [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma | L8.53

53

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*

October 26, 2021 | TCSS562: Software Engineering for Cloud Computing [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma | L8.54

54

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

55

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

56

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

57

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?

58

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 (?)

59

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

60

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

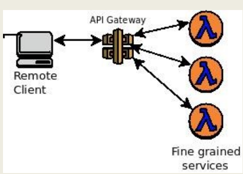
61

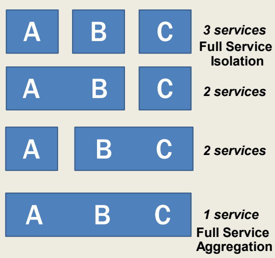
EXTRACT TRANSFORM LOAD DATA PIPELINE - 2

- Service 3: **QUERY**
- Using relational database, apply filter(s) and/or functions to aggregate data to produce sums, totals, averages
- Output aggregations as JSON

62

SERVICE COMPOSITION

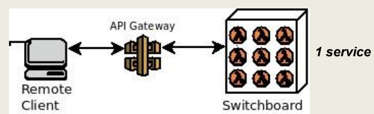




Other possible compositions: group by library, functional cohesion, etc.

63

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

64

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

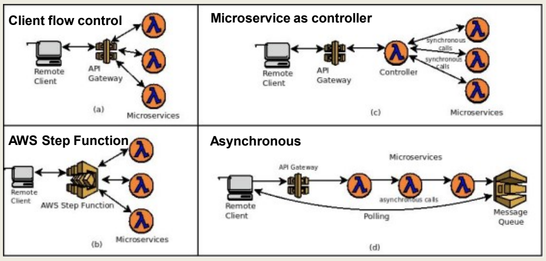
65

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)

66

APPLICATION FLOW CONTROL - 3



67

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

68

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

69

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

70

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…

71

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

72

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

October 26, 2021 | TCSS562: Software Engineering for Cloud Computing [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma | L8.73

73

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?

October 26, 2021 | TCSS562: Software Engineering for Cloud Computing [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma | L8.74

74

QUESTIONS-

October 26, 2021 | TCSS562: Software Engineering for Cloud Computing [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma | L8.75

75