TCSS 562: SOFTWARE ENGINEERING FOR CLOUD COMPUTING

Cloud Computing:

Intro to Cloud Computing
Cloud Delivery Models

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

FEEDBACK FROM 10/17

- Automatic Scaling: How do you automate growing and shrinking resources in the cloud?
- Consider how scaling is done for different cloud delivery models?
- IaaS: EC2
- PaaS: Elastic Beanstalk
- FaaS: AWS Lambda

REVIEW – 10/17

- What are the risks of adopting cloud computing?
  - Increased security vulnerabilities
  - Reduced operational governance / control
  - Network latency
  - Performance monitoring
  - Limited portability
  - Geographical issues
- Cloud Computing Roles:
  - Cloud provider, cloud consumer, cloud service owner, cloud resource administrator, cloud auditor, cloud brokers, cloud carriers

October 22, 2018	TCSS562: Software Engineering for Cloud Computing [Fall 2018] School of Engineering and Technology, University of Washington - Tacoma	L8.3

REVIEW - 2

- Cloud expands boundaries
  - Organization boundary, Trust boundary
- Cloud characteristics
  - On-demand usage
  - Ubiquitous access
  - Measured Usage (for billing)
  - Cloudwatch Metrics
  - What concerns results from Multitenancy in the cloud?
  - What is Resource Elasticity in the cloud?

October 22, 2018	TCSS562: Software Engineering for Cloud Computing [Fall 2018] School of Engineering and Technology, University of Washington - Tacoma	L8.4

OBJECTIVES

- From: Cloud Computing Concepts, Technology & Architecture:
- Cloud Computing Concepts and Models
  - Roles and boundaries
  - Cloud characteristics
  - Cloud delivery models
  - Cloud deployment models

OBJECTIVES

- From: Cloud Computing Concepts, Technology & Architecture:
- Cloud Computing Concepts and Models
  - Roles and boundaries
  - Cloud characteristics
  - Cloud delivery models
  - Cloud deployment models

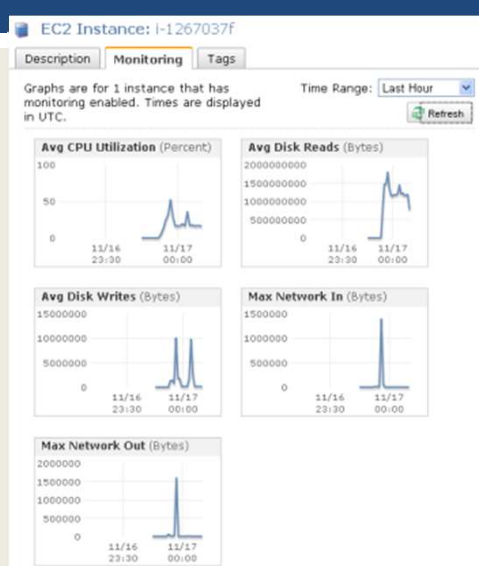
MEASURED USAGE

- Cloud platform tracks usage of IT resources
- For billing purposes
- Enables charging only for IT resources actually used
- Can be time-based (minute, hour, day)
- Can be throughput-based (MB, GB)

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

October 22, 2018	TCSS562: Software Engineering for Cloud Computing [Fall 2018] School of Engineering and Technology, University of Washington - Tacoma	L8.7

EC2 CLOUDWATCH METRICS

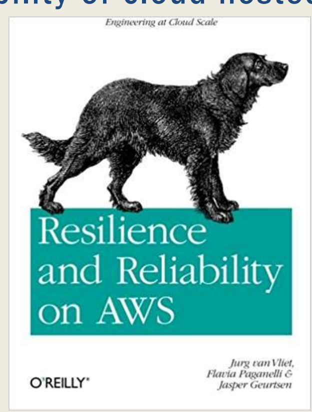


October 22, 2018	TCSS562: Software Engineering for Cloud Computing [Fall 2018] School of Engineering and Technology, University of Washington - Tacoma	L8.8

EC2 CLOUDWATCH METRICS

RESILIENCY

- Distributed redundancy across physical locations
- 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



OBJECTIVES

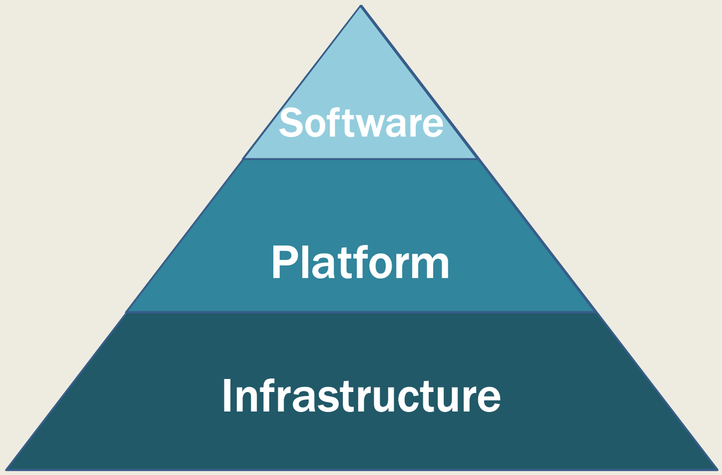
- From: Cloud Computing Concepts, Technology & Architecture:
- Cloud Computing Concepts and Models
  - Roles and boundaries
  - Cloud characteristics
  - Cloud delivery models
  - Cloud deployment models

CLOUD DELIVERY MODELS

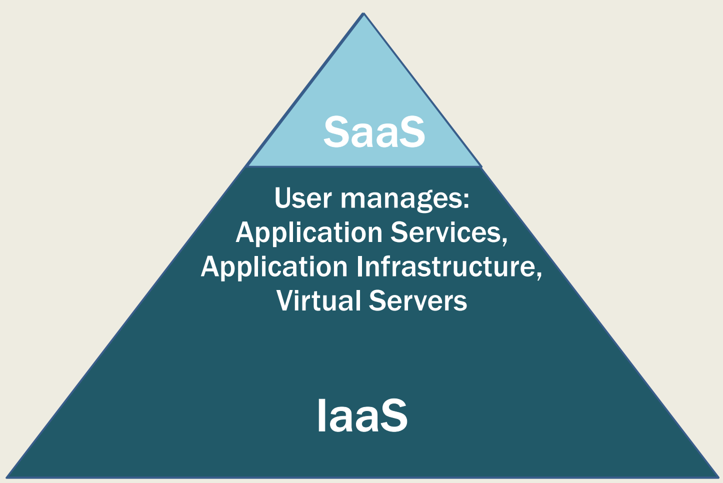
- What is the appropriate level of abstraction?
- How should applications be deployed?
  - IaaS, PaaS, SaaS, DbaaS, FaaS
- How do we ensure Quality-of-Service?
  - Performance, Availability, Responsiveness, Fault Tolerance

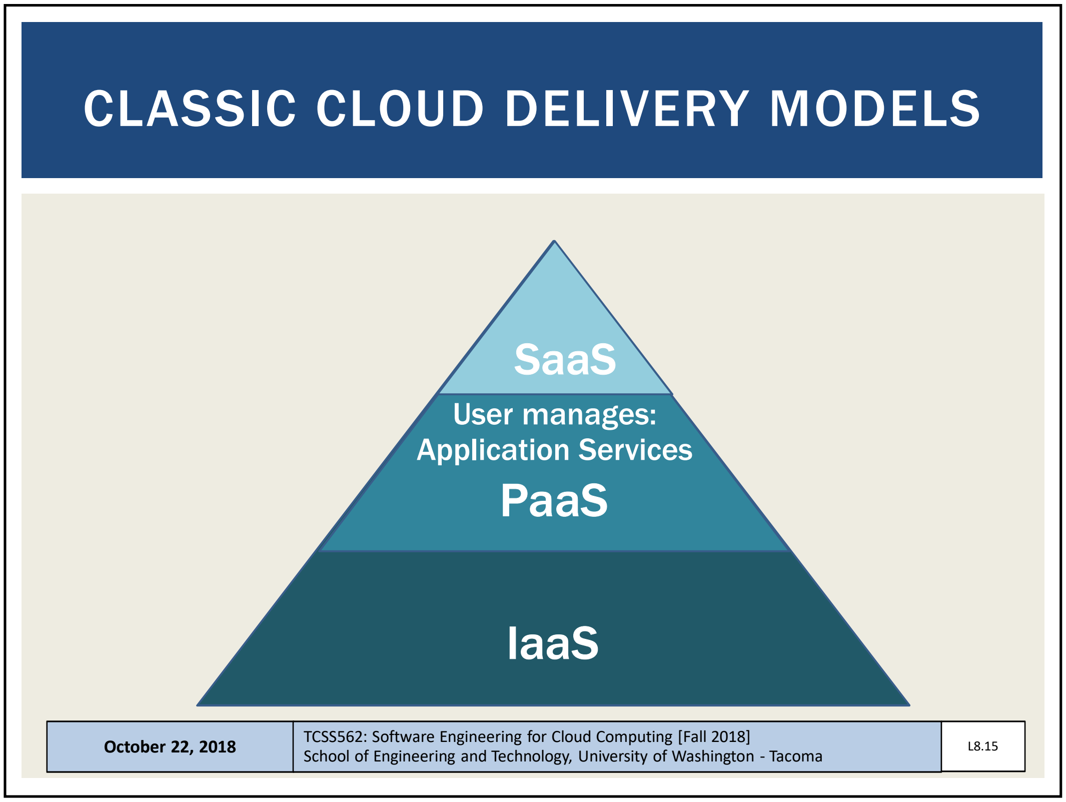
- How is *scalability* provided?
- How do we minimize hosting costs?
  - How do we estimate hosting costs?

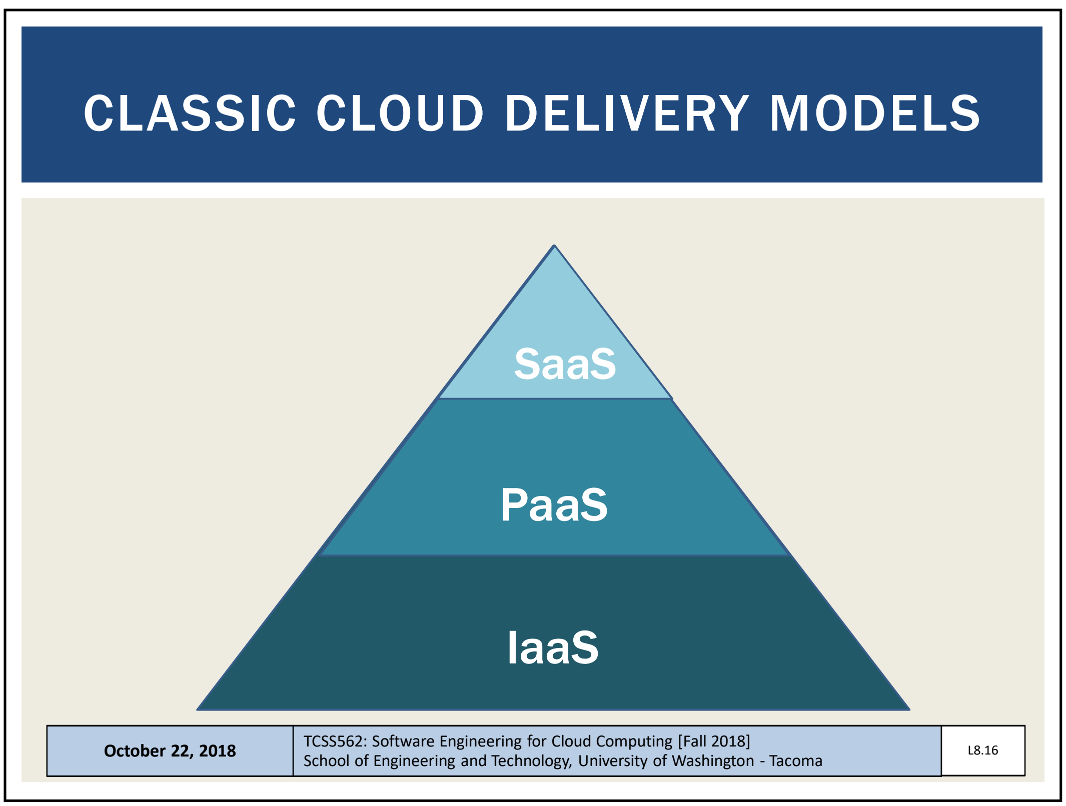
CLASSIC CLOUD DELIVERY MODELS

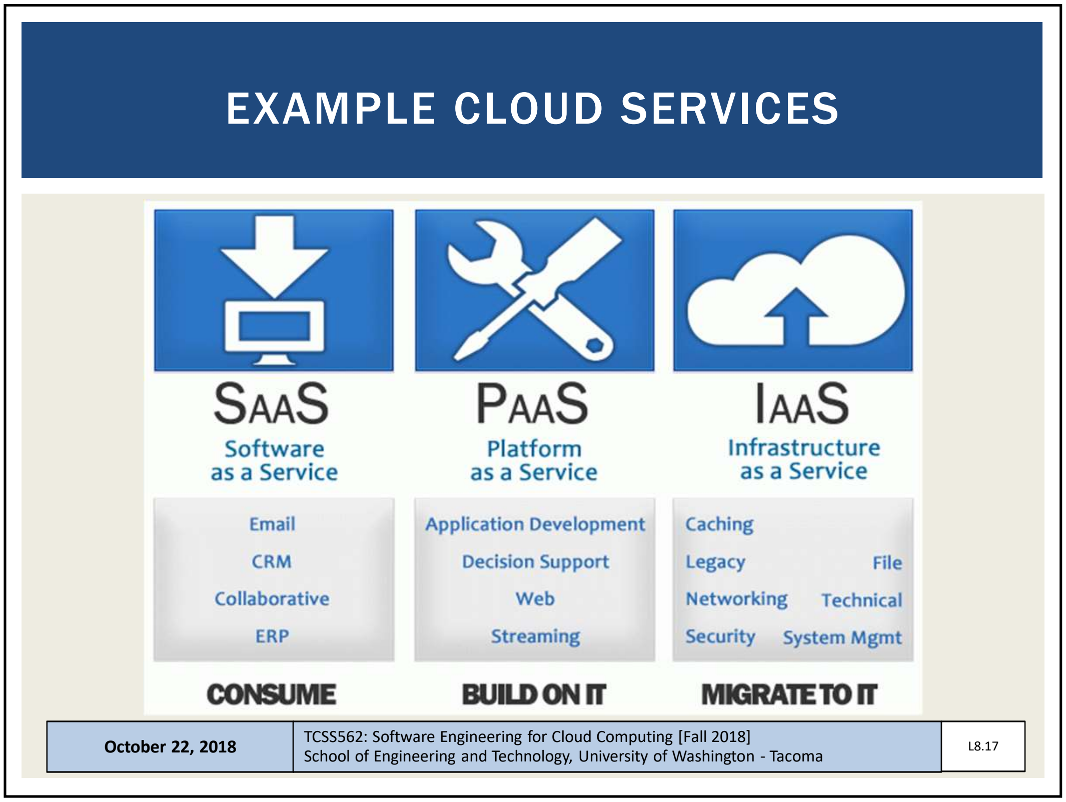


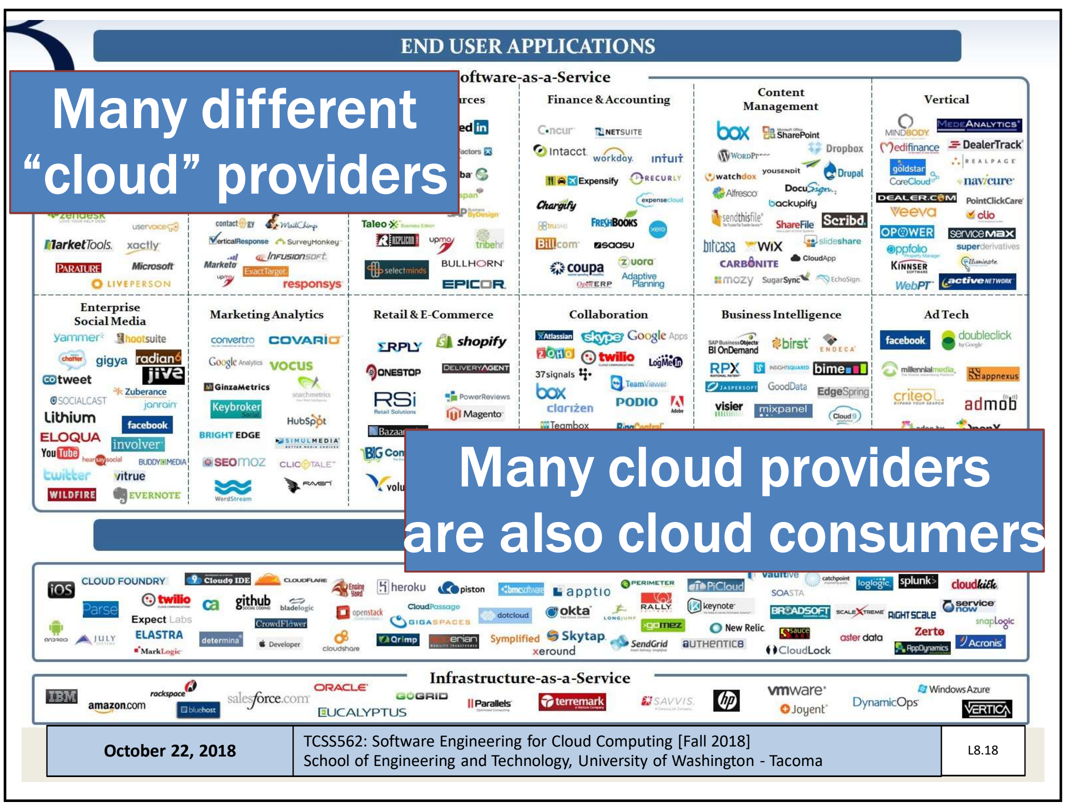
CLASSIC CLOUD DELIVERY MODELS




CLASSIC CLOUD DELIVERY MODELS
SaaS
User manages:
Application Services
PaaS
IaaS
October 22, 2018
TCSS562: Software Engineering for Cloud Computing [Fall 2018]
School of Engineering and Technology, University of Washington - Tacoma
L8.15

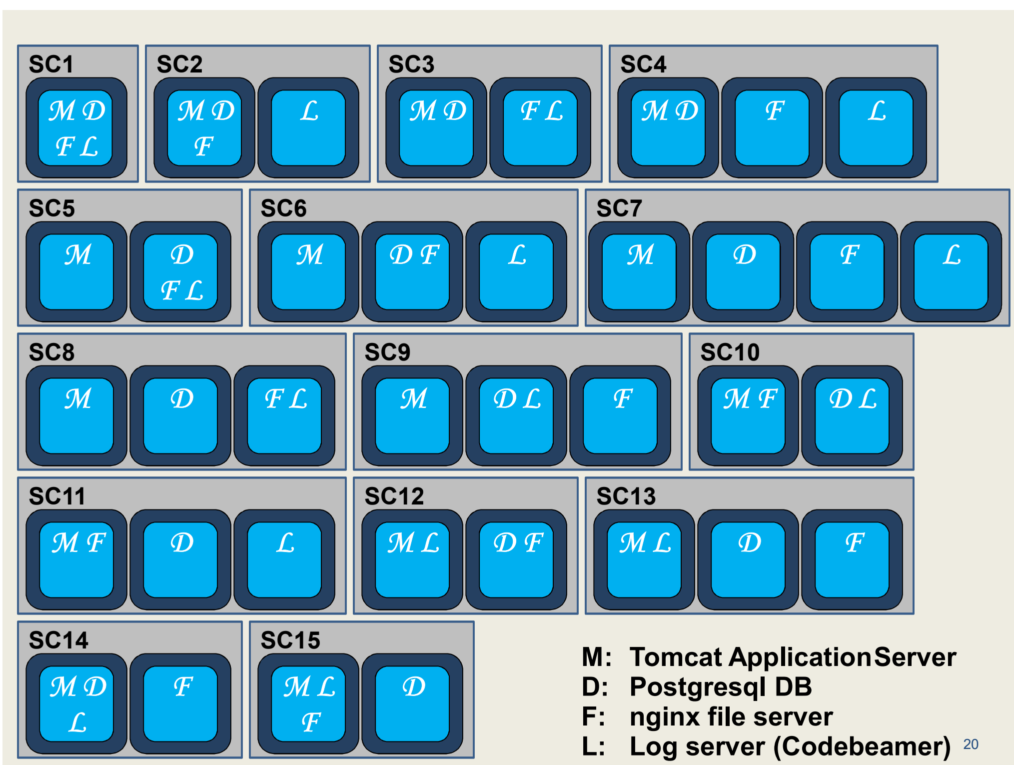

CLASSIC CLOUD DELIVERY MODELS
SaaS
PaaS
IaaS
October 22, 2018
TCSS562: Software Engineering for Cloud Computing [Fall 2018]
School of Engineering and Technology, University of Washington - Tacoma
L8.16

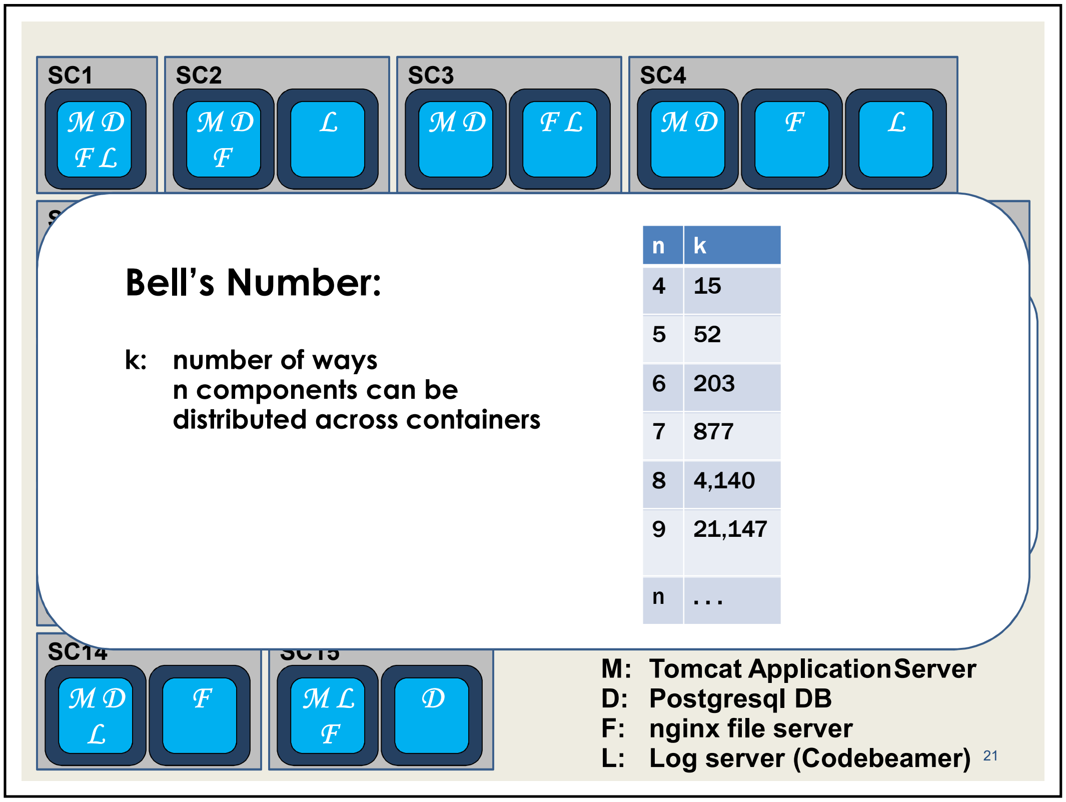

EXAMPLE CLOUD SERVICES
SAAS
Software as a Service
Email
CRM
Collaborative
ERP
CONSUME
PAAS
Platform as a Service
Application Development
Decision Support
Web
Streaming
BUILD ON IT
IAAS
Infrastructure as a Service
Caching
Legacy
File
Networking
Technical
Security
System Mgmt
MIGRATE TO IT
October 22, 2018
TCSS562: Software Engineering for Cloud Computing [Fall 2018]
School of Engineering and Technology, University of Washington - Tacoma
L8.17


END USER APPLICATIONS
Software-as-a-Service
Many different "cloud" providers
Finance & Accounting
Content Management
Vertical
Enterprise Social Media
Marketing Analytics
Retail & E-Commerce
Collaboration
Business Intelligence
Ad Tech
Many cloud providers are also cloud consumers
Infrastructure-as-a-Service
October 22, 2018
TCSS562: Software Engineering for Cloud Computing [Fall 2018]
School of Engineering and Technology, University of Washington - Tacoma
L8.18

INFRASTRUCTURE-AS-A-SERVICE

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

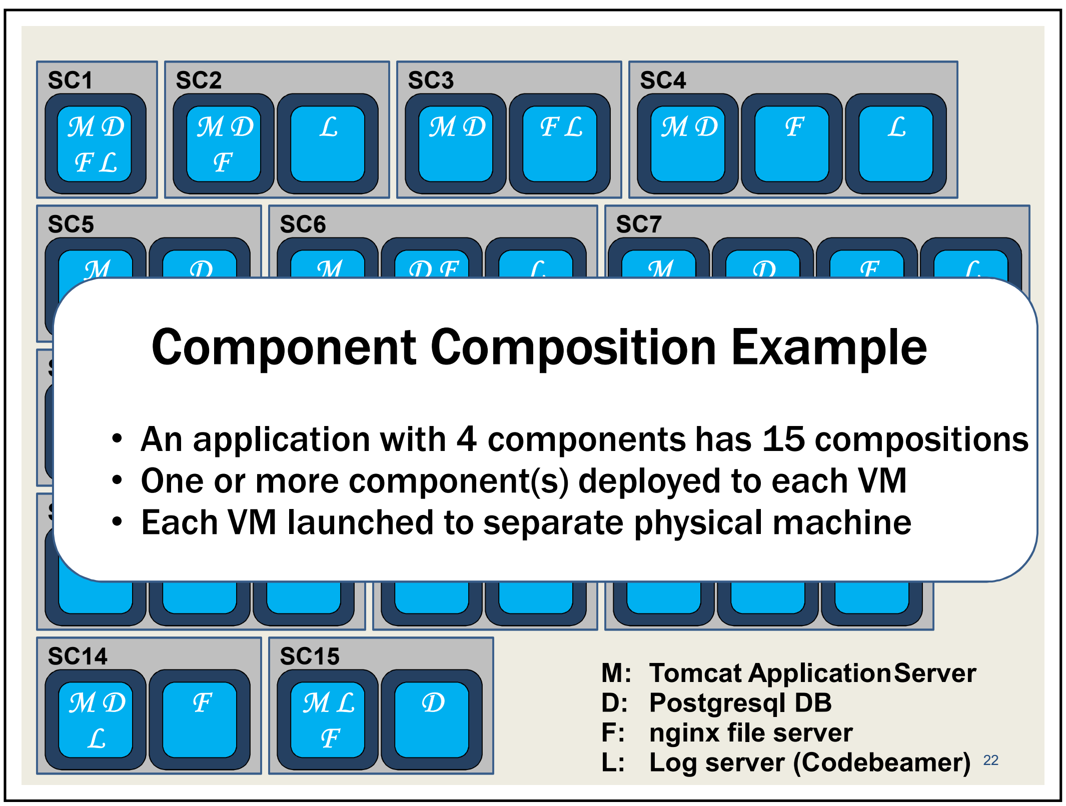




Bell's Number:

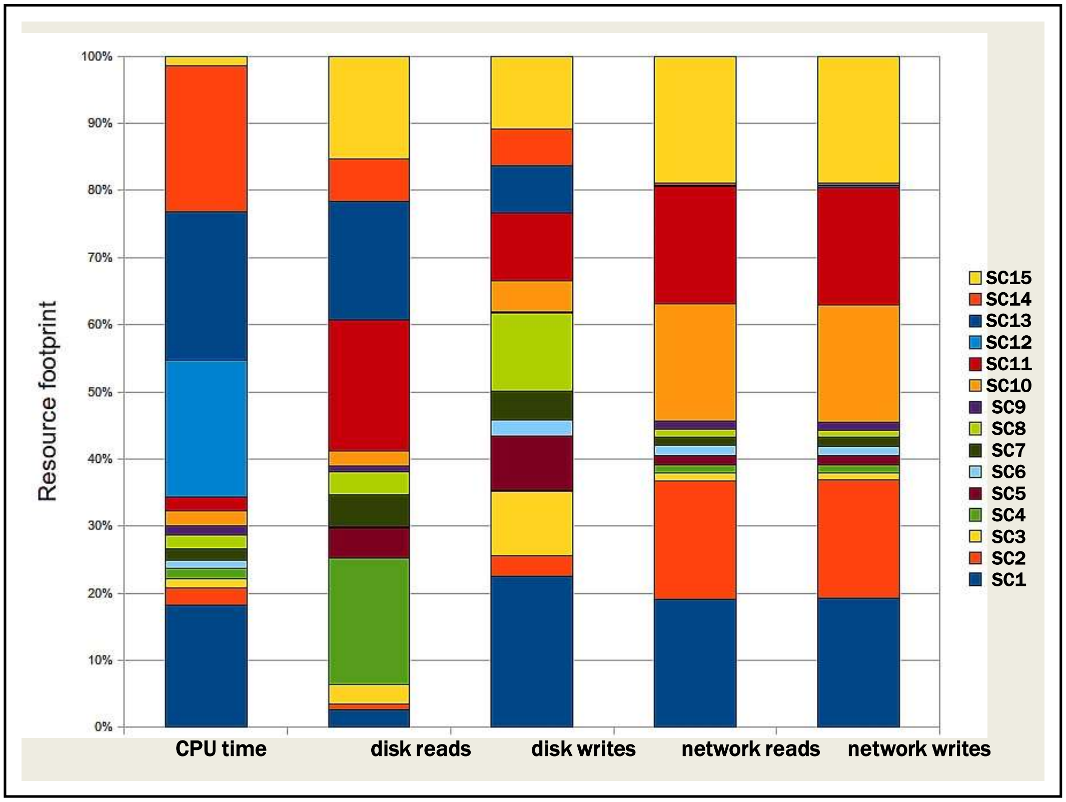
k: number of ways n components can be distributed across containers

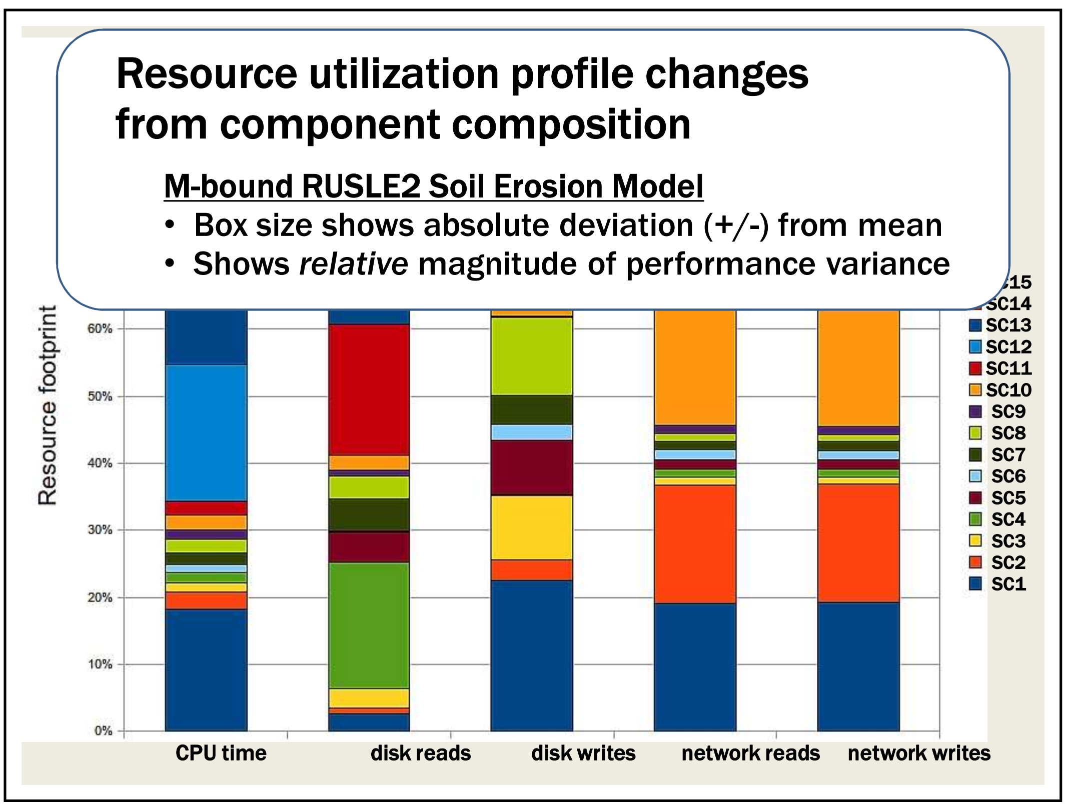
n	k
4	15
5	52
6	203
7	877
8	4,140
9	21,147
n	. . .

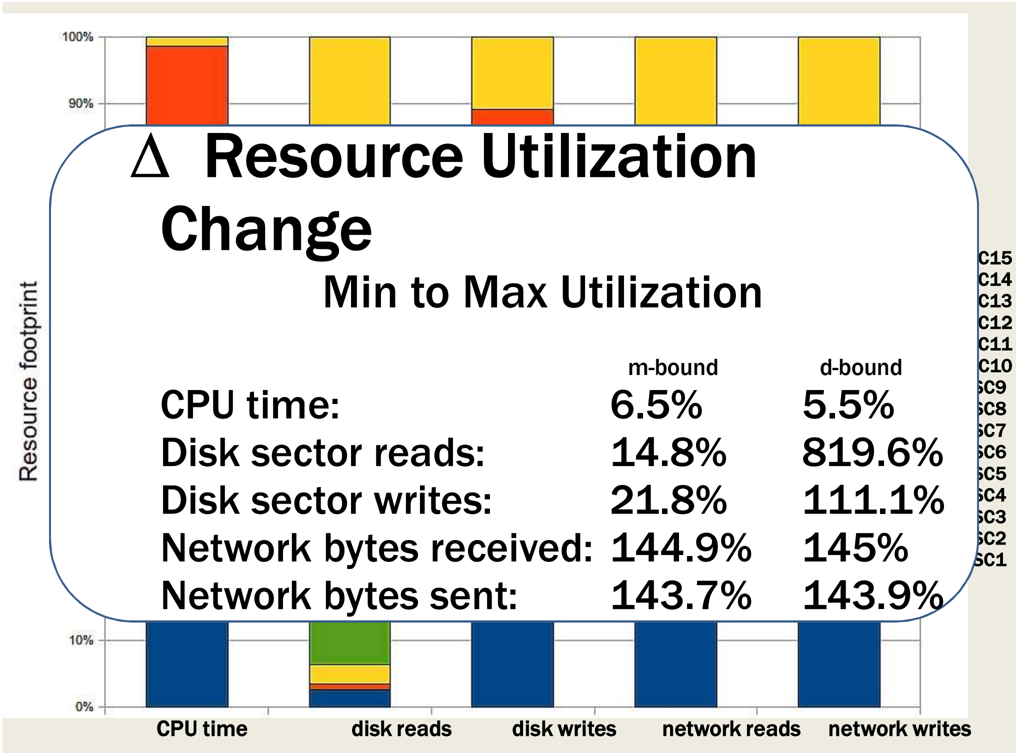


Component Composition Example

- An application with 4 components has 15 compositions
- One or more component(s) deployed to each VM
- Each VM launched to separate physical machine


Resource footprint
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
CPU time
disk reads
disk writes
network reads
network writes
SC15
SC14
SC13
SC12
SC11
SC10
SC9
SC8
SC7
SC6
SC5
SC4
SC3
SC2
SC1


Resource utilization profile changes from component composition
M-bound RUSLE2 Soil Erosion Model
• Box size shows absolute deviation (+/-) from mean
• Shows *relative* magnitude of performance variance
Resource footprint
60%
50%
40%
30%
20%
10%
0%
CPU time
disk reads
disk writes
network reads
network writes
SC13
SC12
SC11
SC10
SC9
SC8
SC7
SC6
SC5
SC4
SC3
SC2
SC1



Δ Resource Utilization Change

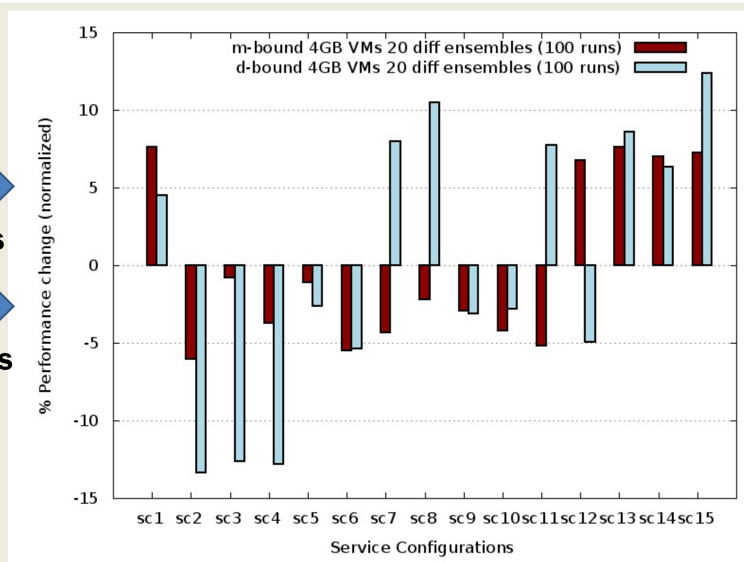
Min to Max Utilization

	m-bound	d-bound
CPU time:	6.5%	5.5%
Disk sector reads:	14.8%	819.6%
Disk sector writes:	21.8%	111.1%
Network bytes received:	144.9%	145%
Network bytes sent:	143.7%	143.9%

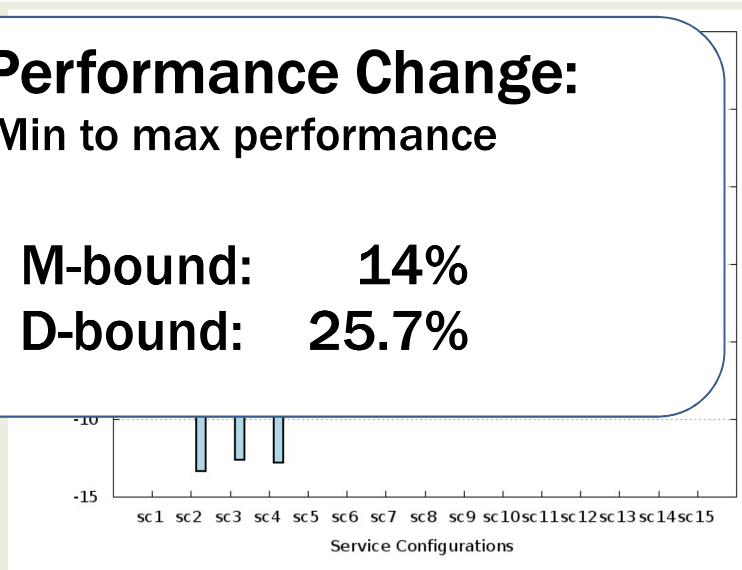
PERFORMANCE IMPLICATIONS OF APPLICATION DEPLOYMENTS

Slower deployments

Faster deployments



PERFORMANCE IMPLICATIONS OF APPLICATION DEPLOYMENTS



Slo

Fa

Δ Performance Change:

Min to max performance

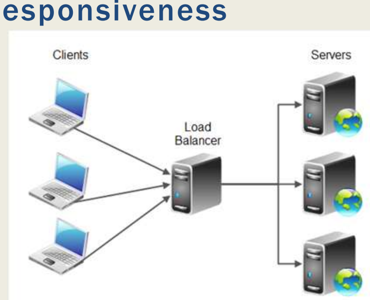
M-bound: 14%

D-bound: 25.7%

PLATFORM-AS-A-SERVICE

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

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

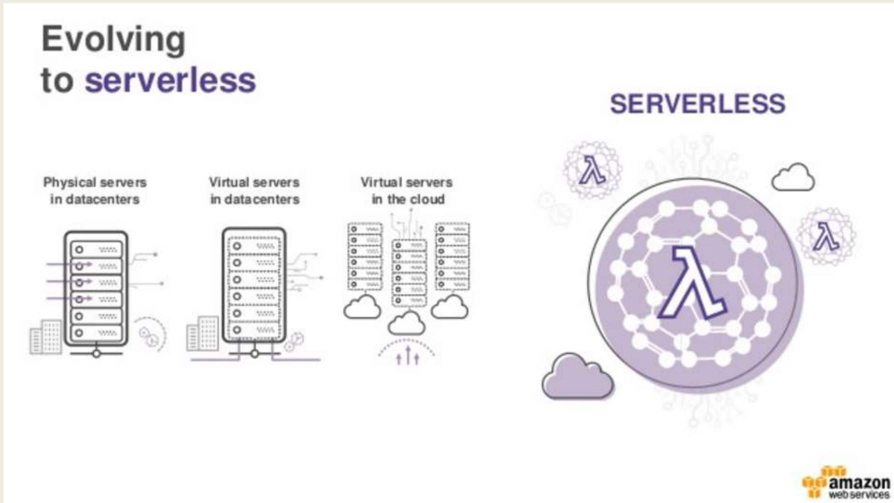


USES FOR PAAS

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

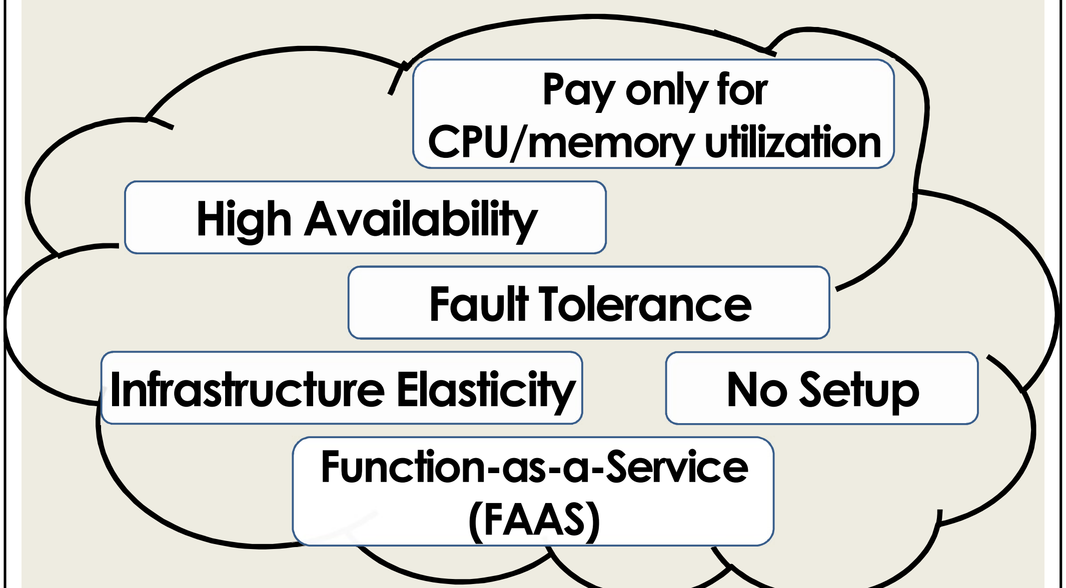
SERVERLESS COMPUTING

SERVERLESS COMPUTING - 2



October 22, 2018 | TCSS562: Software Engineering for Cloud Computing [Fall 2018]
School of Engineering and Technology, University of Washington - Tacoma | L8.31

SERVERLESS COMPUTING



SERVERLESS COMPUTING

Why Serverless Computing?

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

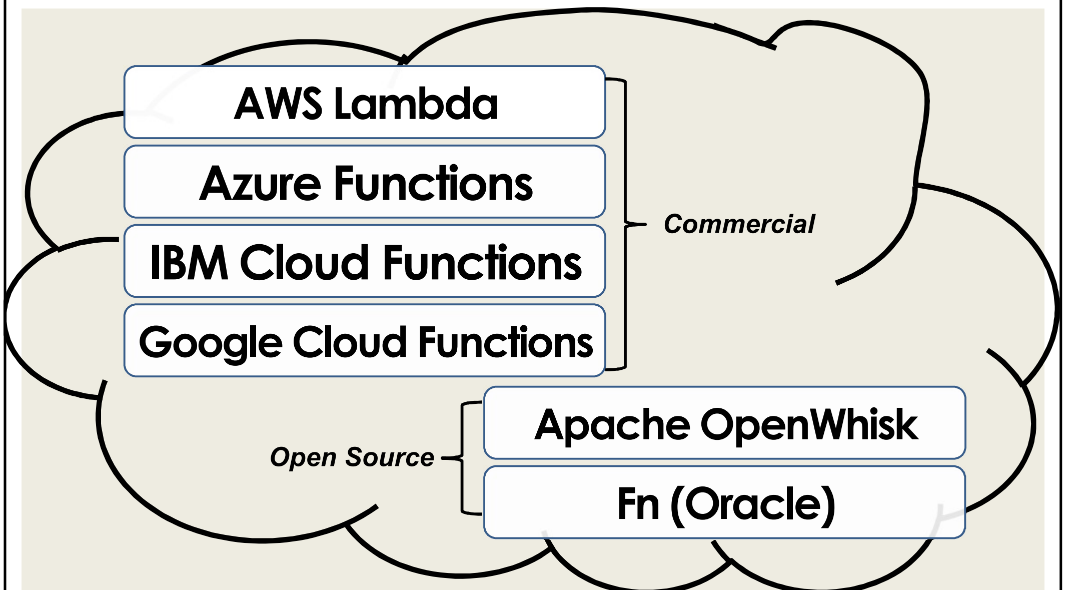
…they are built into the platform

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…

FAAS PLATFORMS



AWS LAMBDA

Using AWS Lambda

Bring your own code
- Node.js, Java, Python, C#
- Bring your own libraries (even native ones)

Simple resource model
- Select power rating from 128 MB to 3 GB
- CPU and network allocated proportionately

Flexible use
- Synchronous or asynchronous
- Integrated with other AWS services

Flexible authorization
- Securely grant access to resources and VPCs
- Fine-grained control for invoking your functions

Images credit: aws.amazon.com

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

October 22, 2018	TCSS562: Software Engineering for Cloud Computing [Fall 2018] School of Engineering and Technology, University of Washington - Tacoma	L8.37

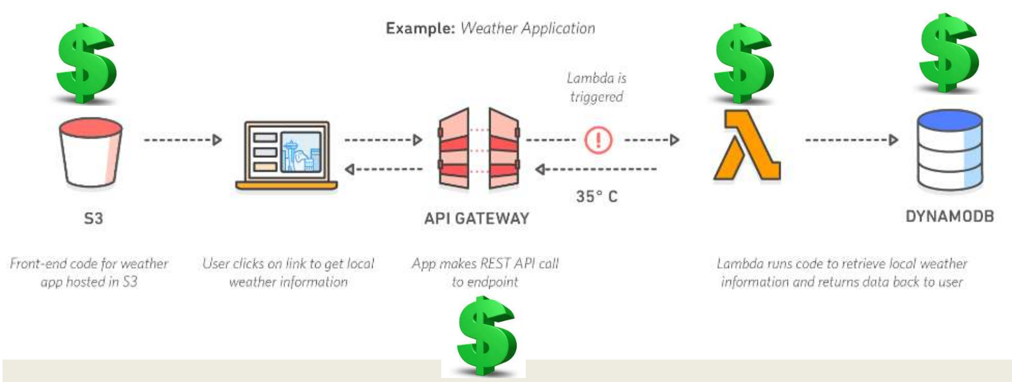
FAAS PLATFORMS - 3

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

October 22, 2018	TCSS562: Software Engineering for Cloud Computing [Fall 2018] School of Engineering and Technology, University of Washington - Tacoma	L8.38

CLOUD NATIVE SOFTWARE ARCHITECTURE

- Every service with a different pricing model



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 →

FAAS COMPUTING BILLING MODELS

- AWS Lambda Pricing
- 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

October 22, 2018	TCSS562: Software Engineering for Cloud Computing [Fall 2018] School of Engineering and Technology, University of Washington - Tacoma	L8.41

WEBSERVICE HOSTING EXAMPLE

- ON AWS Lambda
- Each service call: 100% of 1 CPU-core
  100% of 4GB of memory
- Workload: 2 continuous client threads
- Duration: 1 month (30 days)
- ON AWS EC2:
- Amazon EC2 c4.large 2-vCPU VM
- Hosting cost: $72/month
  c4.large: 10¢/hour, 24 hrs/day x 30 days
- **How much would hosting this workload cost on AWS Lambda?**

October 22, 2018	TCSS562: Software Engineering for Cloud Computing [Fall 2018] School of Engineering and Technology, University of Washington - Tacoma	L8.42

PRICING OBFUSCATION

- Workload: 20,736,000 GB-sec
- FREE: - 400,000 GB-sec
- Charge: 20,336,600 GB-sec
- Memory: $339.00
- Invocations: 5,184,000 calls
- FREE: - 1,000,000 calls
- Charge: 4,184,000 calls
- Calls: $.84
- Total: $339.84
- BREAK-EVEN POINT = ~4,320,000 GB-sec-month

PRICING OBFUSCATION

- Workload: 20,736,000 GB-sec
- FREE: - 400,000 GB-sec
- Ch
- Me
- Inv
- FR
- Ch
- Calls: $.84
- Total: $339.84
- BREAK-EVEN POINT = ~4,320,000 GB-sec-month

Worst-case scenario = ~4.72x !

AWS EC2: $72.00

AWS Lambda: $339.84

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?

FACTORS IMPACTING PERFORMANCE OF FAAS COMPUTING PLATFORMS

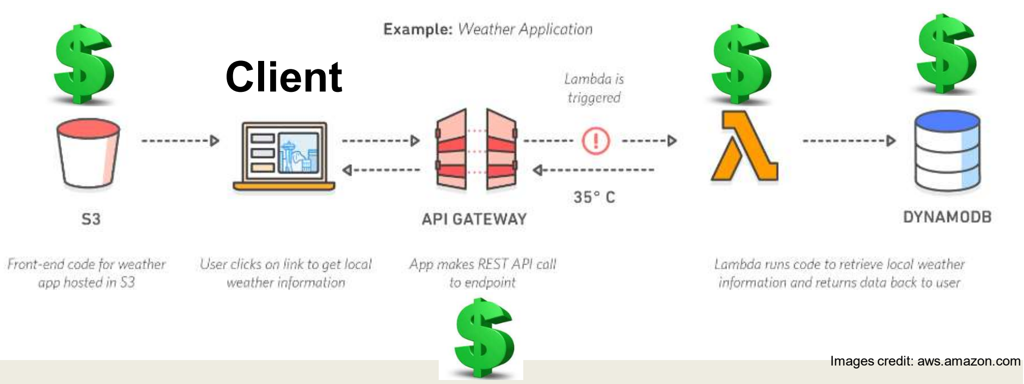
- Infrastructure elasticity
- Load balancing
- Provisioning variation
- Infrastructure retention: COLD vs. WARM
  - Infrastructure freeze/thaw cycle
- Memory reservation
- Service composition

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?

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:**

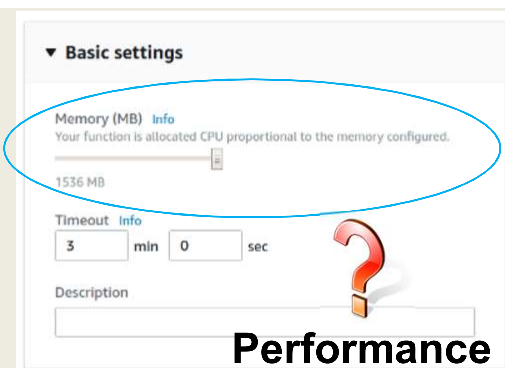
AWS Lambda Pricing

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

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

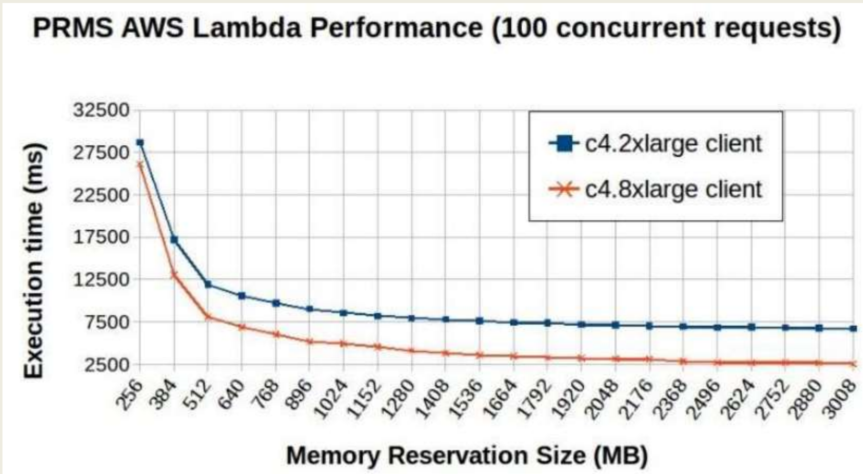
MEMORY RESERVATION QUESTION…

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



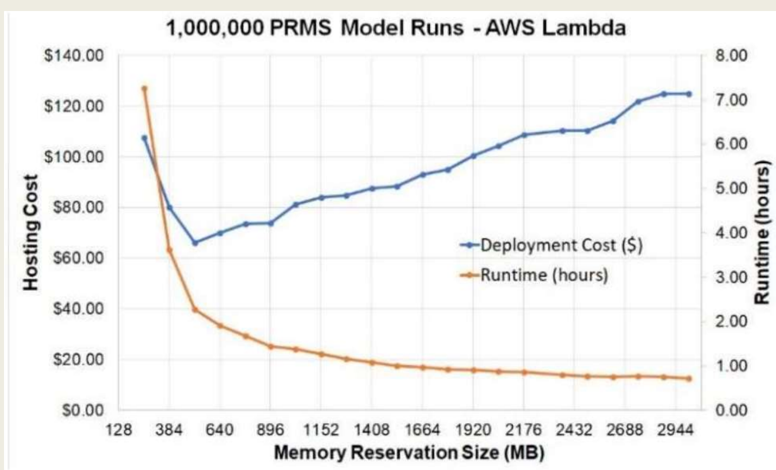
- **But how much memory do model services require?**

LAMBDA: PERFORMANCE VS MEMORY



October 22, 2018	TCSS562: Software Engineering for Cloud Computing [Fall 2018] School of Engineering and Technology, University of Washington - Tacoma	L8.51

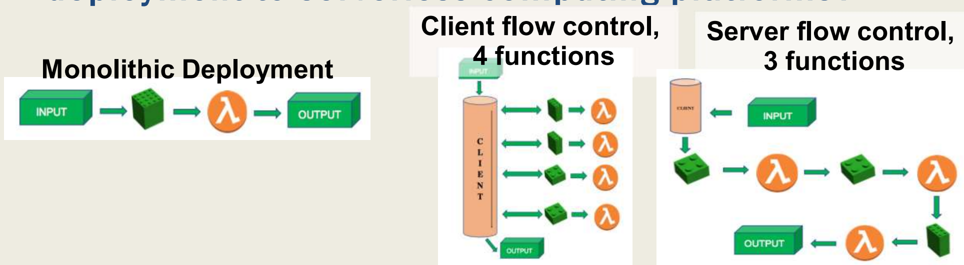
LAMBDA: OPTIMIZING COST OF 1,000,000 CALLS



October 22, 2018	TCSS562: Software Engineering for Cloud Computing [Fall 2018] School of Engineering and Technology, University of Washington - Tacoma	L8.52

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

INFRASTRUCTURE FREEZE/THAW CYCLE

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

Performance

Image from: Denver7 – The Denver Channel News

FUNCTION-AS-A-SERVICE

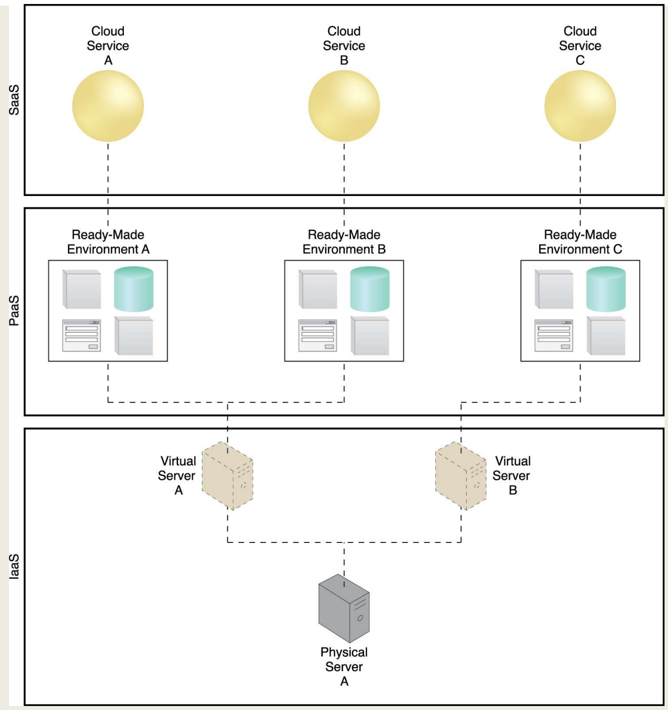
AWS Lambda Demo

55

SOFTWARE-AS-A-SERVICE

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

October 22, 2018 | TCSS562: Software Engineering for Cloud Computing [Fall 2018] School of Engineering and Technology, University of Washington - Tacoma | L8.56



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

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

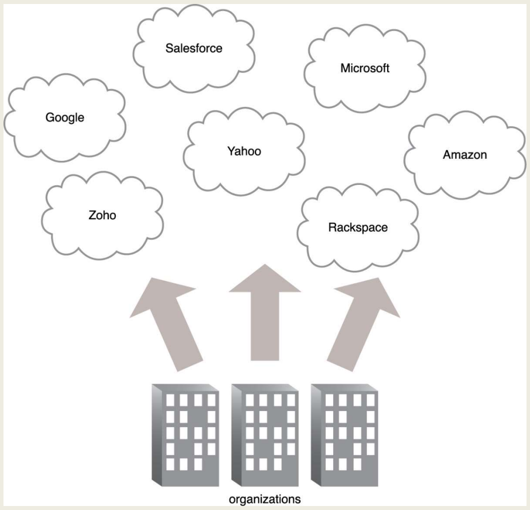
OBJECTIVES

- Cloud Computing Concepts and Models
  - Roles and boundaries
  - Cloud characteristics
  - Cloud delivery models
  - Cloud deployment models

CLOUD DEPLOYMENT MODELS

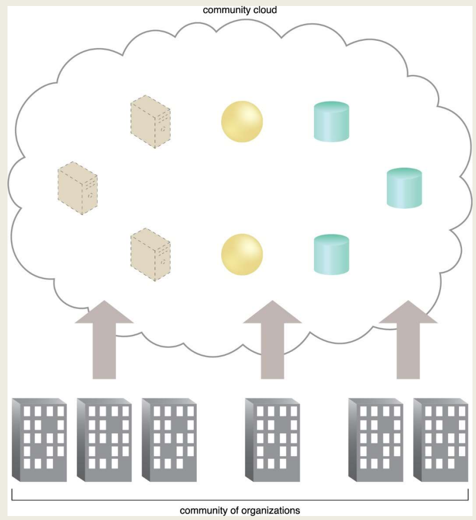
- Distinguished by ownership, size, access
- Four common models
  - Public cloud
  - Community cloud
  - Hybrid cloud
  - Private cloud

PUBLIC CLOUDS



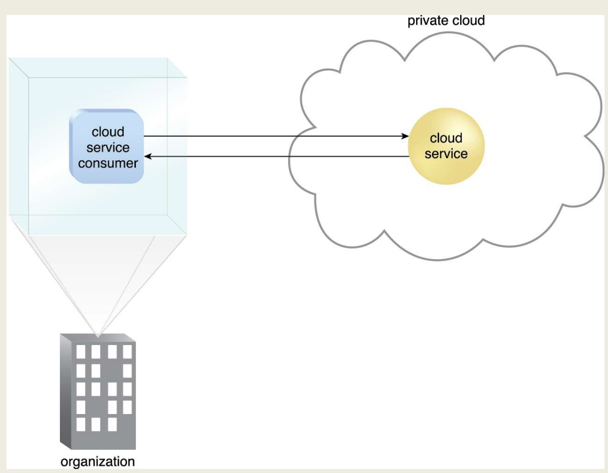
COMMUNITY CLOUD

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



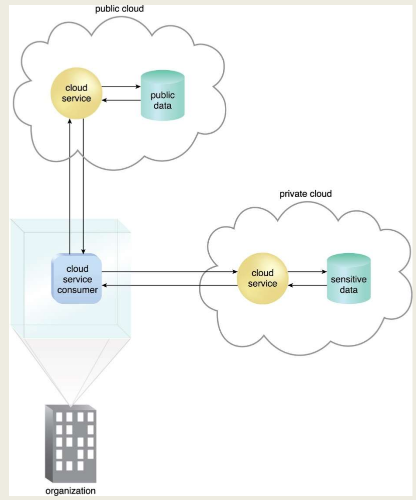
PRIVATE CLOUD

- 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

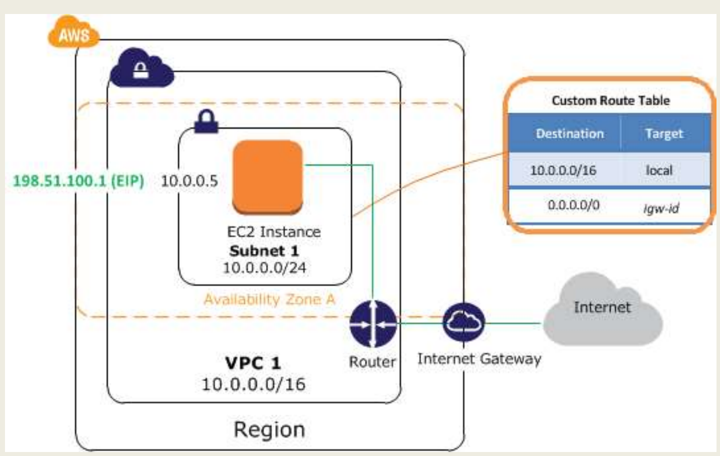
- Federated cloud
  - Simply means to aggregate two or more clouds together
  - Hybrid is typically private-public
  - Federated can be public-public, private-private, etc.
  - Also called inter-cloud
- Virtual private cloud
  - Google and Microsoft simply call these virtual networks
  - Ability to interconnect 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

QUESTIONS

October 22, 2018 | TCSS562: Software Engineering for Cloud Computing [Fall 2018]
School of Engineering and Technology, University of Washington - Tacoma | L8.67

SIMPLE VPC

- Recommended when using Amazon EC2



October 22, 2018 | TCSS562: Software Engineering for Cloud Computing [Fall 2018]
School of Engineering and Technology, University of Washington - Tacoma | L8.68

VPC SPANNING AVAILABILITY ZONES