



GITHUB ACTIONS

DR. ISAAC GRIFFITH

IDAHO STATE UNIVERSITY

Outcomes



After today's lecture you will be able to:

- Understand the basic concepts of cloud computing
- Understand the basic services offered by AWS



The Cloud

CS 3321



- 2006 - Amazon Web Services (AWS) began offering IT infrastructure services to businesses
- We now call this cloud computing
- In the cloud
 - No longer need to plan and procure servers
 - Now we can spin up hundreds or thousands of servers in minutes

Cloud computing allows you to replace upfront capital infrastructure expenses with low variable costs that scale with business.



- **Cloud Computing** - The on-demand delivery of
 - compute power
 - database storage
 - applications, and
 - other IT resourcesvia a cloud services platform across the internet
- Provides a simple way to access servers, storage, databases and many other applications

Advantages of Cloud Computing



- Trade capital expense for variable expense
- Benefit from massive economies of scale
- Stop guessing capacity
- Increase speed and agility
- Stop spending money running and maintaining data centers
- Go global in minutes

Types of Cloud Computing



- Goal is to provide Devs and IT Departments with the ability to focus on what matters
- As Cloud Computing has grown, so has the number of models and deployment strategies, each providing
 - different levels of control
 - different levels of flexibility
 - different levels of management
- Three key Models have emerged over time:
 - Infrastructure as a Service (IaaS)
 - Platform as a Service (PaaS)
 - Software as a Service (SaaS)

⌘ Computing Models

CS 3321

- **IaaS** – Procure the infrastructure, as needed, rather than own it
 - Abstracts the basic building blocks of IT into a service which can be provisioned
 - These include: networking, computers (virtual and physical), and storage space
 - Provides high levels of flexibility and capability
- **PaaS** – Procure the platform, as needed, rather than own it
 - Abstracts the away the management of underlying infrastructure
 - Allows for focus to be placed on deployment and management of applications
 - No need to worry about resource procurement, capacity planning, software maintenance, patching, or other underlying aspects of running an application
- **SaaS** – Procure the software, as needed, rather than own it
 - Provides the completed product that is run and managed by the service provider
 - No need to consider how the service is maintained or underlying infrastructure managed
 - *Example:* Web-based Email (think GMail here at the University)



- **Cloud**

- Cloud-application fully deployed to the cloud and ran in the cloud
- Built upon low-level infrastructure controlled by high-level services abstracting management, architecture, and scaling requirements

- **Hybrid**

- Provides a means to connect existing on-premise IT infrastructure with cloud infrastructure

- **On-premises**

- Private cloud, using virtualization and resource management
- Significantly limits the benefits of cloud computing, but provides dedicated resources
- Similar to the legacy IT approach

AWS Cloud Platform

CS 3321



- **AWS Management Console**

- The main Web UI from which you manage all components of your AWS infrastructure/account
- Can also be done via the AWS Console Mobile App

- **AWS Command Line Interface**

- Command line tool that allows management of AWS services via the command line
- Provides scripting capabilities (i.e., DevOps)

- **AWS SDK**

- Allows your Apps to manage AWS via an API tailored to platform and language

AWS Compute

CS 3321

- Amazon EC2 - Elastic Compute Cloud
 - Basis for all of AWS compute capabilities
 - Allows for scalable and resizable compute capacity in the cloud
 - Uses a web service to obtain and configure capacity
 - Simple to obtain and boot new instances



EC2 Instance Types

- **On-Demand Instances**
 - Pay for compute capacity by the hour
 - Can increase/decrease based on demand while only paying the hourly rate
- **Reserved Instances**
 - Provide up to 75% discount on On-Demand
 - Provides flexibility to change families, OS types, and tenancies
- **Spot Instances**
 - Provide up to 90% discount on On-Demand
 - Allows you to take advantage of unused compute capacity

- Amazon Elastic Container Registry (ECR)
 - Provides functionality similar to docker hub but on AWS
 - Stores, manages, and deploys docker container images
 - Integrates with ECS (next slide) and simplifies workflow
- ECR hosts your images and provides Identity and Access Management (IAM) resource-level control for the repo.
- **Note:** you only pay for the amount of data you store and transfer

- Amazon Elastic Container Service
 - Highly scalable, high-performance container orchestration service for Docker containers
 - Provides ease of use to run and scale containerized apps
- **Goal** is to eliminate need to install/operate your own container orchestration service

- Easiest way (on AWS) to launch and manage a virtual private server
- Plan includes the following
 - A VM
 - SSD based storage
 - Data transfer
 - DNS management
 - Static IP address
 - Low, predictable price

- Enabled developers, scientists, and engineers to
 - Easily and efficiently run hundreds of thousands of batch computing jobs
 - Think **High-Performance** or **Super-computing**
 - Dynamically provisions the optimal quantity and type of compute resources
 - CPU or memory-optimized instances
 - Based on the volume and specific resource requirements of the batch job
- No need to install/manage
 - Batch computing software
 - Server clusters
- Plans, schedules and executes your jobs across the AWS compute services

- Service for deploying and scaling web apps and services developed in
 - Java
 - .NET
 - PHP
 - Node.js
 - Python
 - Ruby
 - Go
 - Docker
- Uses common servers such as Apache, Nginx, Passenger and IIS

- A ECS compute engine that allows Container operations without requiring you to manage servers or clusters
- No need to do any of the following
 - Provision VMs
 - Configure VMs
 - Scale VM Clusters
- Minimizes your decisions as it takes care of all this and
 - deciding when to scale your clusters
 - optimizing cluster packing

No more managing infrastructure, just focus on building apps

- ECS has two modes: **Fargate Launch** and **EC2 Launch**
- **Fargate Launch** - All you need to do is:
 - package your app in a container
 - specify the CPU and memory requirements
 - define networking policies
 - launch the app
- **EC2 Launch** - provides a bit more control
 - server-level control
 - granular control over the infrastructure that runs your apps
 - ECS manages your cluster of servers, tracks CPU, memory and other resources
 - You are responsible for provisioning, patching, and scaling server clusters

- Run Code without provisioning or managing servers
- All you pay for is the compute time you consume, with no cost if your code isn't running
- You can run your code without any type of application or backend service
 - Effectively zero administration
- Just upload your code, and let Lambda take care of everything else
 - High Scalability
 - High Availability

- AWS Serverless Application Repository
 - provides a means to quickly deploy code samples, components, and apps
 - can be used for web and mobile backends
 - can be used for data processing, logging, monitoring, IoT, etc.
- Uses the AWS Serverless Application Model (SAM) template to define AWS resources used
- You can also publish your own apps and share them across a team, an organization, or the community at large.

Things To Do



1. Get yourself an AWS account
2. Start learning about these different technologies
 - I would start with AWS Fargate and Elastic Beanstalk
3. Start considering how you might connect github, dockerhub, and aws together

For Next Time



Idaho State
University

Computer
Science

- Review the Reading
- Review this Lecture
- Come to Class





Are there any questions?