Contents to cover

* What is AWS?
  + What is cloud computing
  + Cloud computing deployment models
    - Cloud:
      * fully deployed in the cloud and all parts of the application run in the cloud.
      * Cloud-based applications can be built on low-level infrastructure pieces or can use higher level services that provide abstraction from the management, architecting, and scaling requirements of core infrastructure.
    - Hybrid:
      * A way to connect infrastructure and applications between cloud-based resources and existing resources that are not located in the cloud.
      * To extend, and grow, an organization's infrastructure into the cloud while connecting cloud resources to internal system.
    - On premises
      * Deploying resources on-premises, using virtualization and resource management tools, is sometimes called “private cloud”
      * Does not provide many of the benefits of cloud computing but is sometimes sought for its ability to provide [dedicated resources](https://aws.amazon.com/enterprise/private/?pg=TOCC)
  + Cloud computing benefits
    - Variable expenses: pay only for what you use
    - Cost optimization: focus on applications and customers
    - Capacity: scale in and scale out as needed
    - Economies of scale: benefit from customers’ aggregated usage
    - Speed and agility: minutes between wanting and having resources
    - Global in minutes: quickly deploy applications worldwide with AWS’s global infrastructure
  + AWS core service categories
* AWS Pricing
* Amazon EC2
  + What it is?
  + How it works?
  + Instance types
  + Amazon EC2 Auto Scaling
  + Amazon EC2 Load Balancer
* Lambda
  + Serverless compute services
  + What it is?
  + How it works?
* Global Infrastructure and Reliability
  + Regions
    - Each Region is designed to be isolated from the other Regions. This achieves the greatest possible fault tolerance and stability.
    - When you view your resources, you see only the resources that are tied to the Region that you specified. This is because Regions are isolated from each other, and we don't automatically replicate resources across Regions.
  + Availability Zones
    - Each Region has multiple, isolated locations known as Availability Zones.
    - When you launch an instance, you select a Region and a virtual private cloud (VPC), and then you can either select a subnet from one of the Availability Zones or let us choose one for you.
    - If you distribute your instances across multiple Availability Zones and one instance fails, you can design your application so that an instance in another Availability Zone can handle requests.
    - You can also use Elastic IP addresses to mask the failure of an instance in one Availability Zone by rapidly remapping the address to an instance in another Availability Zone.
* Interact with AWS services
* Networking
  + <https://www.youtube.com/watch?v=tD9vDv0uyI8>
  + Amazon VPC
    - Subnets
    - Internet gateway
    - Virtual private gateway
    - AWS Direct Connect
  + Network access control lists and security groups
    - Network ACL: stateless packet filtering
    - Security groups: stateful packet filtering
  + Regions of use (totally cannot communicate)
* Storage
  + Block storage
  + Object storage
  + File storage
* IAM

<https://towardsaws.com/making-a-simple-webpage-using-aws-and-flask-part-1-37afb1e6ee74>

**Hosting A Simple Webpage Using AWS EC2**

* Launch Instance
* Name and tags: Web Server
* Amazon Machine Image (AMI) – templates that contains all the necessary software configurations required to launch your instance
* Instance type
* Key pair (login) -> Create new key pair->RSA
  + System platform compatibility:
    - Linux users — .pem file format
    - Mac users — .pem file format
    - Windows PowerShell users — .pem file format
    - Windows PuTTY/Cygwin users — .ppk file format
  + Use PuTTYgen to convert .pem to .ppk
  + <https://www.c-sharpcorner.com/article/difference-between-pem-and-ppk/#:~:text=PEM%20(Privacy%20Enhanced%20Mail)%20is,client%2C%20it%20does%20not%20support%20>
* Network settings
  + Edit
    - Security group name: Web Server SG
    - Description: Security Group for Web Server
  + Add security group rule
    - Default first rule for SSH remotely to EC2 instance
    - Type: HTTP
      * Source: 0.0.0.0/0, ::/0 (all the IPv4 and IPv6 ranges respectively can access your AWS service(s)) [CDIR]
    - Type: Custom TCP Rule
      * Allow traffic to come from and go back to anywhere/wherever (respectively) when accessing Flask site
      * Port Range: 5000
      * Source: 0.0.0.0/0, ::/0 (all the IPv4 and IPv6 ranges respectively can access your AWS service(s)) [CDIR]
* Configure Storage: 8GiB, gp2
* Everything under ‘Advanced Details’ can be default
* Input for ‘User data’

‘’’

#!/bin/bash

sudo yum update -y (superuser do) (open-source command-line package-management utility) (update all presently installed packages)

sudo yum install -y httpd (install Apache HTTP Server)

sudo systemctl start httpd (init system)

sudo systemctl enable httpd (start automatically when server boots again)

‘’’

systemctl status httpd

cd /var/www/html (default root folder of the web server)

sudo nano index.html

<h1>Hello, World! Welcome to my Web Page!</h1>

Ctrl+S

Ctrl+X

* Launch Instance
* Terminate instance
* Check for EBS deletion
* Delete security group

**Lambda Demo**

Create role to access CloudWatch

1. IAM->Roles->AWS service->Lambda
2. AWSLambdaBasicExecutionRole
3. Name ‘Hello-world-lambda-role”

Create Lambda function

1. Functions->Create function->Use a blueprint->hello-world(nodejs)->Configure
2. Name
3. Use an existing role (the one just created)
4. Create function
5. console.log("Hello World");
6. console.log(`Logging scheduled event: ${event}`);

Create test event

1. Test (on upper left corner)
2. Event name
3. Change values to better names->Save
4. Change ‘event’ to ‘event.key1’->Deploy
5. Test

Go to CloudWatch

1. Events->Rules
2. Event Source: Schedule
3. Add target->Lambda function created just now->configure details->Name, description->Create rule
4. Check logs

Cleanup

1. Rules->delete the rule
2. Lambda->delete the function
3. IAM->delete role

<https://docs.aws.amazon.com/lambda/latest/dg/nodejs-handler.html>

<https://stackify.com/aws-lambda-with-node-js-a-complete-getting-started-guide/>

**Google Cloud Platform (GCP) vs Amazon Web Services (AWS)**

Here are the important pros/benefits of selecting AWS web services:

* Amazon Web Services (AWS) offers easy deployment process for an app
* You should opt for AWS when you have DevOps teams who can configure and manage the infrastructure
* You have very little time to spend on the deployment of a new version of your web or mobile app.
* AWS web service is an ideal option when your project needs high computing power
* Helps you to improve the productivity of the application development team
* A range of automated functionalities including the configuration, scaling, setup, and others
* It is a cost-effective service that allows you to pay only for what you use, without any up-front or long-term commitments.
* AWS allows organizations to use the already familiar programming models, operating systems, databases, and architectures.
* You are allowed cloud access quickly with limitless capacity.

Here are the pros/benefits of selecting Google cloud services:

* Offers higher productivity gained through Quick Access to innovation
* Employees can work from Anywhere
* Future-Proof infrastructure
* It provides a serverless environment which allows you to connect cloud services with a large focus mainly on the microservices architecture.
* Offers Powerful Data Analytics
* Cost-efficiency due to long-term discounts
* Big Data and Machine Learning products
* Offers Instance and payment configuration

<https://www.projectpro.io/article/aws-vs-gcp-which-one-to-choose/477>

1. Is GCP easier than AWS?

If you don't have prior experience with AWS, both technologies are equally easier and complex. GCP has a slight edge over this as it has a bare minimum and simpler implementation. But if your goal is to be proficient in market-dominant technology, then one should start with AWS. Also, suppose one already has a background in AWS. In that case, it becomes easier to transition into GCP and other Cloud technologies as the underlying principles are the same with varying implementation.

2. Which is better, AWS or GCP?

It depends more on the organization’s existing architecture and requirements.

3. Is GCP cheaper than AWS?

Yes, GCP is relatively cheaper than AWS. Its pricing model for services and products is minute-wise compared to the hourly computed charge model of AWS. It is closer to the pay-for-what-you-use model.

4. Will GCP take over AWS?

Only time will be able to tell if GCP will take over AWS. AWS has an already established foundation and grip in the market, which places it ahead of GCP. But surely GCP has been catching up, and the year-wise revenue report for both the companies is proof that GCP is growing quickly.

5. Is GCP more secure than AWS?

Both the technologies have many security features and provisions, but comparatively, AWS is more secure than GCP.

6. Is AWS faster than GCP?

GCP is, in fact, faster than AWS. In an experiment based on performance and efficacy, GCP was able to run more than 30K transactions per minute, thus giving more throughput than AWS.