Cloud Computing
Intro to Amazon Web Services

Hi, My Name is Tegar Imansyah

± 8 YoE in Tech Industry:

- Current: **Lead Infrastructure Engineer @ Zero One Group** (Software Agency)
- Previous: Several domestic and foreign projects
- Previous: Software Architect @ Alterra Indonesia (Fintech)
- Previous: Founder @ Ngabarin IoT System (Telkom-funded Startup)

Several times in Tech Communities:

- Active: **Mentor @ Bearmentor.com (Fullstack Bootcamp)**
- Active: **Tutor @ Rakamin Academy (Corporate Bootcamp)**
- Coming Soon: **Bongkar.Cloud (SaaS Internal Developer Platform)**
- Previous:
  - Capstone Project Advisor @ Bangkit Academy 2022-2024
  - Chief Organizer @ Python Conference Indonesia 2019
  - Lead Organizer @ Python Surabaya 2017-2019
  - Speaker of Python @ Conference Indonesia 2018
  - Speaker of Python @ Conference Malaysia 2018

And, How About You?

Please share about

- You
- Your learning journey
- What you expect in our sessions

Full Agenda

1. Introduction About Author, AWS and Things to Know Before Started
2. Getting Started with Compute services in AWS
3. Storage Solution in AWS
4. Several Managed Database Service in AWS
5. Security Improvement in AWS
6. Autoscaling and Monitoring in AWS
7. Best Practices in Cloud (Well Architecture)

> Make these sessions a pointer for you to explore more about it.
> Don’t expect you will become an AWS/DevOps Expert just by listening on this session.
>
> **You need practice, practice and practice!**

Preparation Before Start

- Create an AWS account or shared with other students
  - Using AWS Educate
  - Using AWS Normal Account
- Familiarize with AWS Console before automate anything
- Create an IAM User
  - Your registered user is root user, it has all permissions
  - You also can create multiple user with different secret key
- Set tagging and cost budget alert
- Register Github
- Install Git, Docker, AWS CLI and Terraform (For practice last day)

While You are at it

Open this repo: https://github.com/zero-one-group/welcome-entry-level

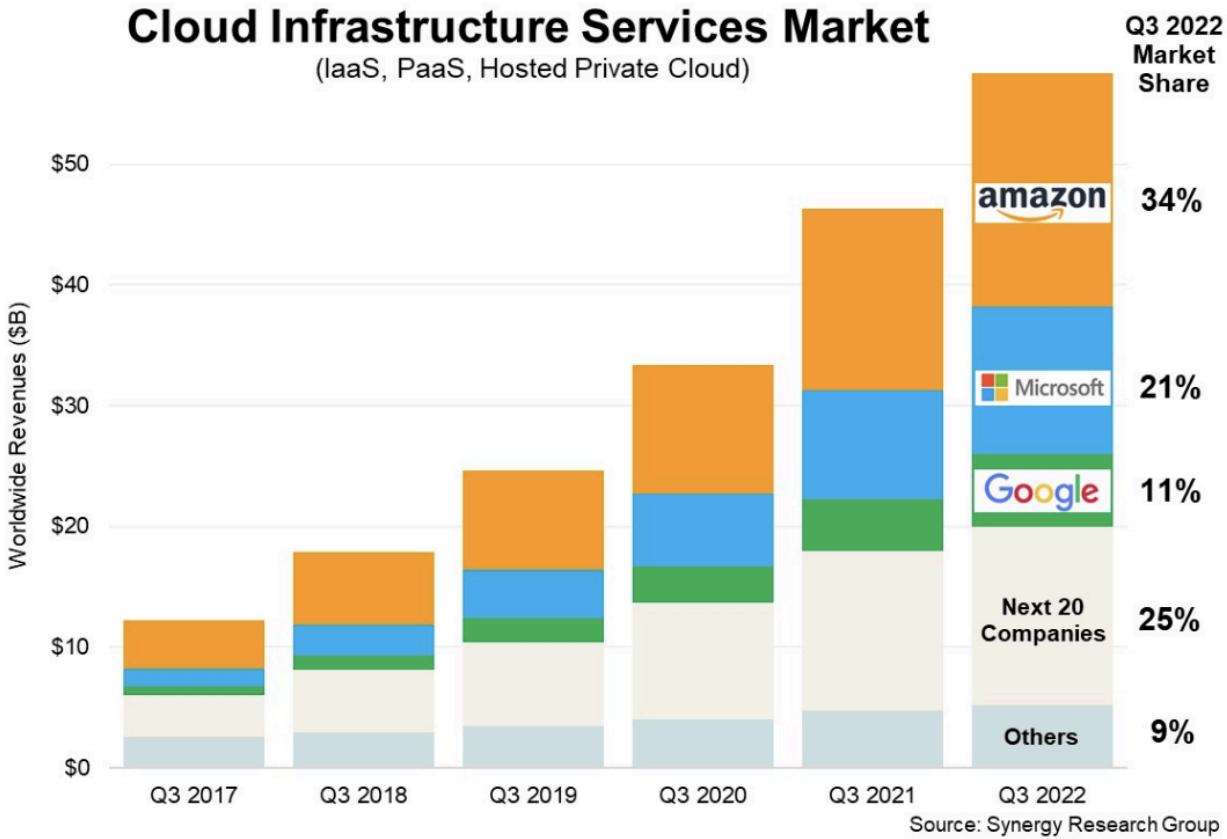
TLDR

- Register Github Student Pack
- Learn from free resources
- Don’t afraid to spend money worth a cup of coffee
- Additional:
  - Get mentoring from ADP List
  - Read and write docs in English
  - Contribute to community (Blog, Commit, Issue, Discussion)
  - Ask for Purchasing power parity (PPP) is you found a good resources

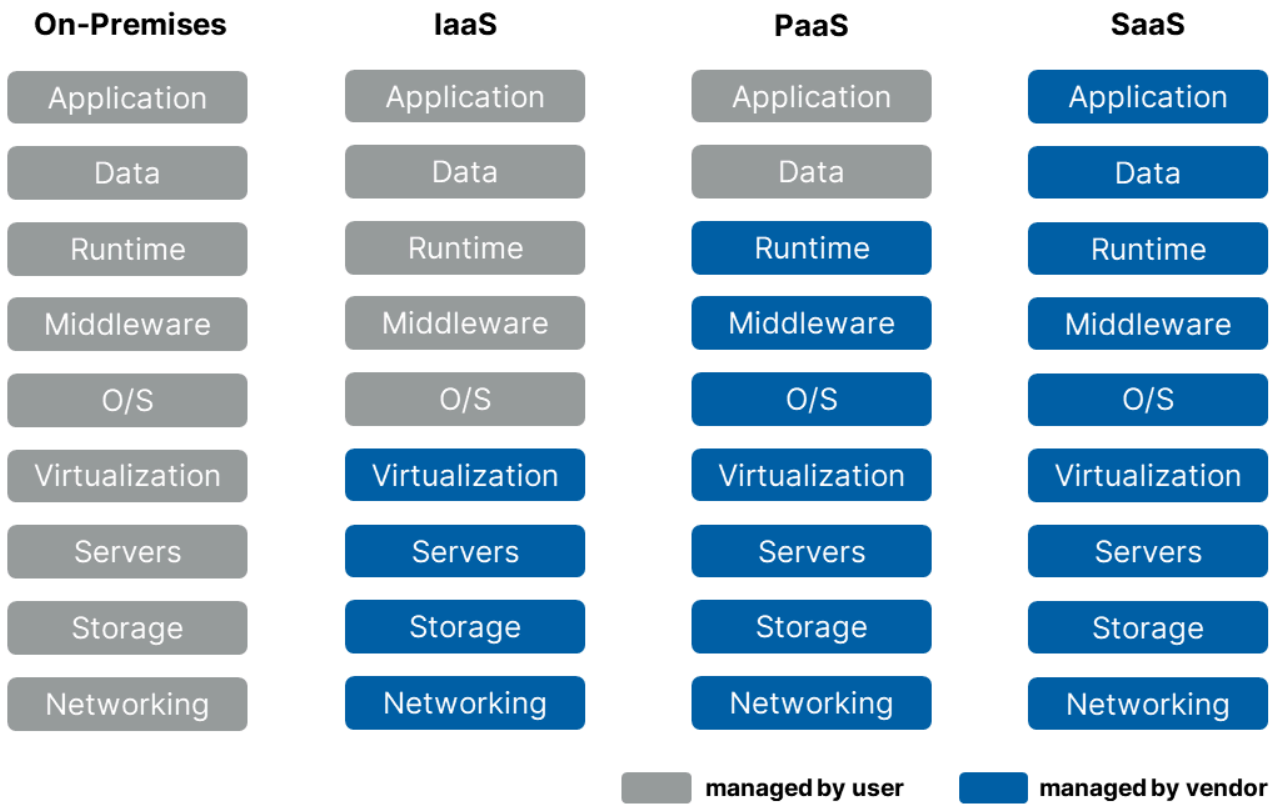
Before we started, anyone know about how the internet works?

I mean, what happen when you press "enter" after writting "google.com" in address bar?

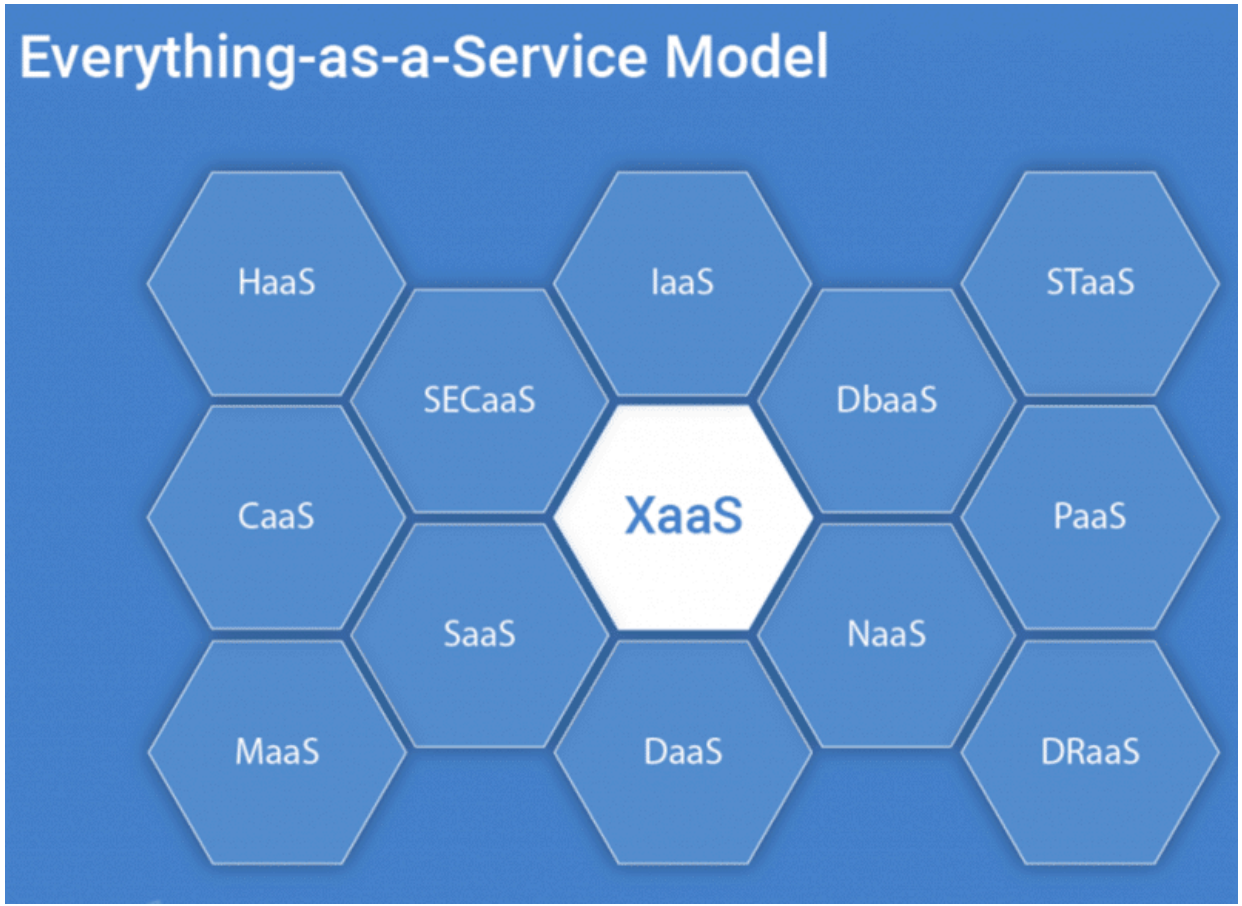
Cloud Market Share



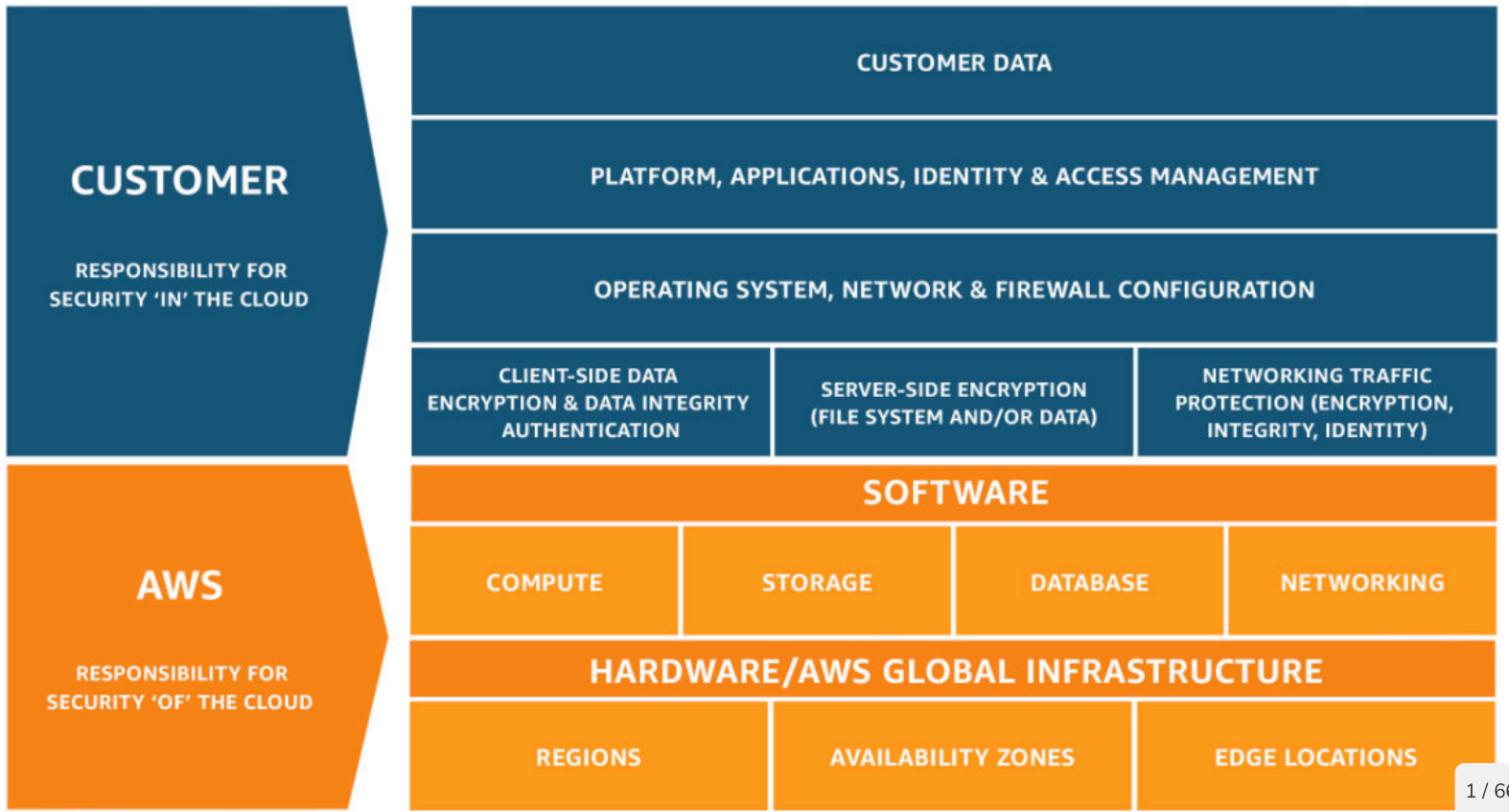
Cloud Models



Cloud XaaS


Everything-as-a-Service Model
HaaS
IaaS
STaaS
SECaaS
DbaaS
CaaS
XaaS
PaaS
SaaS
NaaS
MaaS
DaaS
DRaaS

Cloud Shared Responsibility

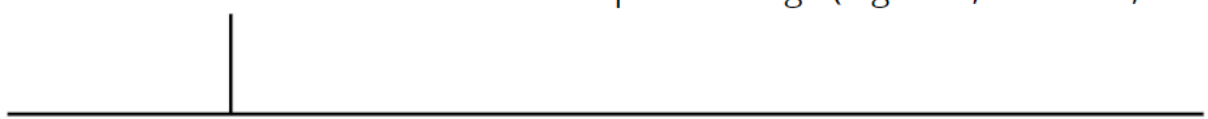


Cloud Pricing Models

Reserved to use the resource for the specific amount of time. Usually monthly or yearly.

vs

Pay-As-You-Go Sometimes called on-demand. You only pay for specific usage (e.g time, resource, storage)



Instance

Pay as long as the instance is up.

Whether you have traffic or not.

Serverless

Pay for the traffic usage, e.g

combination of CPU, RAM & Durati

Introduction to Compute in AWS

AWS Compute Options

https://aws.amazon.com/products/compute/

- Amazon Lightsail (VM)
  - Easy-to-use cloud platform that offers you everything you need to build an application or website
- Amazon Elastic Compute Cloud (EC2) (VM)
  - Secure and resizable compute capacity (virtual servers) in the cloud
- AWS Fargate (Serverless Container)
  - Serverless compute for containers – NB: We need orchestrator to use fargate e.g. EKS or ECS
- AWS APP Runner (Serverless Container)
  - Build and run containerized applications on a fully managed service
- AWS Lambda (Serverless Function)
  - Run code without thinking about servers. Pay only for the compute time you consume

AWS Compute Orchestration

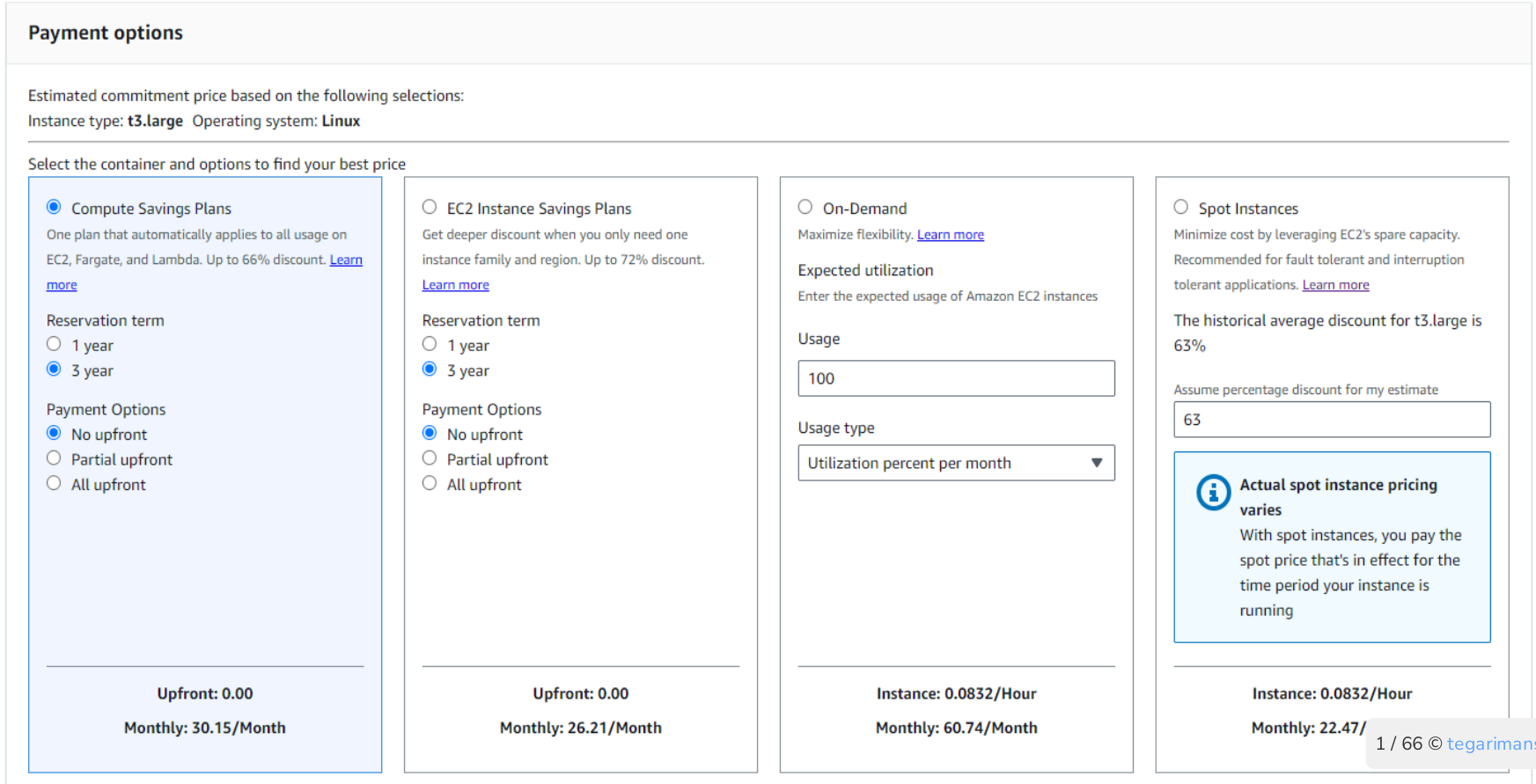
- AWS Elastic Beanstalk
- Amazon EC2 Auto Scaling Group
- AWS Batch
- Amazon Elastic Container Service
- Amazon Elastic Kubernetes Service

EC2 Instance Types

https://aws.amazon.com/ec2/instance-types/

- General Purpose (M, T series)
  - T Series = Burstable, you only have only a few time to get full capacity
  - M series = When you need full capacity
  - T3 => Intel; T3a => AMD Epyc; T3g => AWS Graviton (Arm based, cheaper)
- Compute Optimized ©
- Memory Optimized ®
- Accelerated Computing
- Storage Optimized
- HPC Optimized

EC2 Pricing Model



Any Questions?

Feedback

- Student need to learn docker
  - We will have another session about app deployment with docker to several compute options

Bonus:
App Deployment with Docker

Requirements

- Github Account
  - Create a git repository remotely
- Gitpod
  - For cloud IDE

Or if you want to run it in local development instead of gitpod.

- In windows, I recommend using WSL2
- Docker
  - Pull, run and build docker
- Git CLI
  - Clone, pull and push the source code
- SSH client
  - Accessing server

Step-By-Step Locally

- Fork main repository
- Clone your repository
- Docker build
- Docker run
- Docker compose
- Docker push

Step-By-Step Deployment to App Runner

Step-By-Step Deployment to EC2 (or other linux VM)

Introduction to Storage Options in AWS

AWS Popular Storage Options

https://aws.amazon.com/products/storage/

- Elastic Block Storage (EBS)
  - "Harddrive" for your EC2 instance. Can be attached to one instance only.
- Elastic File System (EFS)
  - "Harddrive" that you can connect via Network File System (NFS) for your linux servers. Can be connected to multiple instances.
- Simple Storage Service (S3)
  - Serverless object-storage that has really high data durability (11 9s)

Usage of (Multiple) EBS in EC2

- Resize Existing EBS
- Add New EBS
  - Attach New EBS to EC2
  - Make it available in instance

Usage of EFS in multiple EC2

Usage of S3 from Backend and Frontend App

Usage of S3 as a Static File Hosting

Data Transfer between Storage Options with DataSync

Introduction to Database Options in AWS

AWS Popular Managed Database Services (1/2)

https://aws.amazon.com/products/database/

Database type	Examples	AWS service
Relational	Traditional applications, enterprise resource planning (ERP), customer relationship management (CRM), ecommerce	Amazon Aurora, Amazon RDS, Amazon Redshift
Key-value	High-traffic web applications, ecommerce systems, gaming applications	Amazon DynamoDB
In-memory	Caching, session management, gaming leaderboards, geospatial applications	Amazon ElastiCache, Amazon MemoryDB for Redis
Document	Content management, catalogs, user profiles	Amazon DocumentDB (with MongoDB compatibility)

AWS Popular Managed Database Services (2/2)

https://aws.amazon.com/products/database/

Database type	Examples	AWS service
Wide column	High-scale industrial apps for equipment maintenance, fleet management, and route optimization	Amazon Keyspaces
Graph	Fraud detection, social networking, recommendation engines	Amazon Neptune
Time series	Internet of Things (IoT) applications, DevOps, industrial telemetry	Amazon Timestream
Ledger	Systems of record, supply chain, registrations, banking transactions	Amazon Ledger Database Services (QLDB)

But, What Should You Learn First?

- RDS
  - Postgres, MySQL, Aurora
- DynamoDB
- ElastiCache

Database Type

SQL vs NoSQL, And several types of NoSQL

Why Should We Use Managed Database Services

Instead of installing in the same server with our app?

Create RDS PostgreSQL and RDS Aurora PostgreSQL

Then access them from EC2

Create ElastiCache for Redis

Then access them from EC2

Create and Connect to Serverless Database

Using DynamoDB

Introduction to Security Options in AWS

AWS Popular Security Services

https://aws.amazon.com/products/security

- Identity and Access Management (IAM)
- Security Groups
- Web Application Firewall (WAF)
- Shields – DDOS Protection
- Secret Manager

Identity and Access Management (IAM)

- Create an IAM User and Groups
- Create an IAM Policy
- IAM Access Key

Create an IAM Roles with EC2 Instance

Give permission to access other AWS Resource without secret key and password

Security Groups

Control the inbound and outbound traffic for associated resources.

AWS Web App Firewall (WAF)

Mitigate OWASP's Top 10 Web Application Vulnerabilities

AWS Shield

Maximize application availability and responsiveness with managed DDoS protection

AWS Secret Manager

Centrally manage the lifecycle of secrets

Introduction to Autoscaling and Monitoring Options in AWS

Agenda

- EC2 Auto Scaling Groups
- Cloudwatch
  - Metrics
  - Logs
  - Dashboard
  - Alarm
- EC2 RAM Monitoring
- Kubernetes (EKS) Auto Scaling Concept
  - Pod Autoscaling (Horizontal and Vertical)
  - Node Autoscaling

Create EC2 Launch Template and Auto Scaling Group

And with initial script that run in first start

Add Cloudwatch Metrics and Log

From EC2 Instances and RDS

Build Cloudwatch Dashboard

To understand what’s going on in wild

Add Cloudwatch Alarm

And Trigger EC2 ASG Automatically Scale Out / Scale In

Add EC2 RAM Metrics and Other Custom Metrics

From inside application to cloudwatch

Intro to Kubernetes Scaling

About Pod Scaling (Horizontal and Vertical) and Node Scaling

Best Practice in AWS

Agenda

- Overview of AWS Well-Architected
- Understanding DevOps
- Workshop Infrastructure as Code
- Workshop CI/CD

AWS Well-Architected Framework

Consist of Six Pillars

- Operational Excellence Pillar
- Security Pillar
- Reliability Pillar
- Performance Efficiency Pillar
- Cost Optimization Pillar
- Sustainability Pillar

> Each pillar has their own whitepaper documents and whitepaper for specific industry use case

> These are guideline that depends on your specific case, not a rigid list of services or action that you must do

> You can implement it step-by-step and not necessarily from a day 1

DevOps

- It’s a culture, not a role. Also refer to SRE or Platform Engineer.
- But if you are hired as a DevOps, typically you expected to
  - Improve software lifecycle
  - Add some automation tooling
  - Add observability tooling
- It’s not a entry-level job, since you need to
  - Understand both software development and operations (infra) very well
  - Understand that there are usually more than one solution for each problem and can find which is better for specific use case
  - Can understand and dig the problem really well, Listen to understand, Have a bargaining power, Have a persuasion skill and Deep understanding for what you will talk about

Understanding DevOps / SRE KPI

Most common KPI that can be used

- Deployment frequency
- Change-Failure rate
- Mean Time to Recovery (MTTR)
- Service Level Agreement, usually consist of
  - Latency
  - Error Rate
  - Availability

How to Achieve the KPI (1/2)

Here is something that you can do to measure and improve the KPI

- Implement Automation Tools
- Implement Observability Tooling
  - Uptime / Health Check
  - RED and USE Metrics
  - Logging
  - Dashboard and Alerting
- Disaster Recovery Plan
  - Fail deployment
  - Fail database migration
  - Wrong drop a database
  - Region downtime

How to Achieve the KPI (2/2)

- Considering Some of Cloud Migration Strategy (7R)
  - Refactor, Replatform, Repurchase, Rehost, Relocate
  - Retain and Retire
- Using some patterns or techniques
  - Multiple app environments (Dev, Staging, Production)
  - Feature Flag
  - Rollback / Revert Deployment or Changes

Workshop Infrastructure of Code

Workshop CI/CD


Thank You