Cloud Technologies

* Cloud computing
  + Outsourcing your IT infrastructure over the internet
  + Rather than on premise computers and databases you use ones provided to you by a cloud provider (AWS,Azure)
  + You access these resources over the internet
* IaaS
  + Infrastructure as a service
  + You are getting “hardware” from a cloud provider
    - EC2(Virtual Computer), EBS(Diskdrive)
* PaaS
  + Platform as a service
  + Hardware + software that serves as a base for you to develop on
  + NOT a completed application for an end user
    - RDS(A virtual machine that Amazon has preconfigured and optimized for Database workloads), s3 (Way to store files)
* SaaS
  + Software as a service
  + Completed application available over the web
    - Gmail, Netflix
* Sandwich analogy
  + IaaS – making a sandwich where the store gives you flour, eggs and large uncut and uncooked pieces of meat
  + PaaS – pre-sliced bread and cold cuts
  + SaaS- the deli gives you a completed sandwich

AWS Amazon Web Services

* Amazon’s cloud computer provide (largest in the world)
  + Main competitors
    - Azure
    - IBM
    - Google cloud platform (GCP)
* Regions
  + Geographical area where services are provided
    - US east 1 (northern virginia)
    - Australia
    - Munich
* Availability zones (AZs)
  + Data centers/ Server farms in a region
  + Most regions have several AZs
  + They provide redundancy in case one goes down the other ones can serve as a backup
* Pros
  + Scalability
    - You can quickly create and remove servers for your application/company
      * React dynamically to demand
      * Save money because you only pay/commission servers for when you need them
  + Simplify maintenance and deployment for your developers and DevOps personnel
    - Easier to manage a company’s IT infrastructure when you can access and control all online
* Cons
  + You have to have internet
    - Not good if you have unstable connections
  + Ties you to a cloud provider
    - You are at the mercy of their prices and might not be able to easily transition between cloud providers
  + You do give up some customizability
    - When you commission a virtual machine you might not be able to control certain aspects of it
      * What type of process it has
  + You have to trust your cloud provider’s security

Horizontal vs Vertical Scaling

* Vertical scaling
  + You commission a larger EC2 to handle incoming requests when demand gets to high
* Horizontal Scaling
  + You have a load balancer which will forward request to many smaller ec2s
  + You will commission more or less of these ec2s based on demand
  + Preferred for most web applications
    - It takes less time to spin up smaller ec2s
    - Easier to scale back
    - If one EC2 crashes the others are fine and handle requests

**EC2**

* Elastic Compute Cloud
* Virtual Machine in the cloud
* Premiere AWS service
* Can be specialized with different CPU and RAM
* **AMI** (Amazon Machine Image)
  + Snapshot of the EC2
  + Includes the OS, installed software and files
  + Can you an AMI as a blueprint to create EC2s
* **EBS** (Elastic Block Store)
  + The hard drive for an EC2
  + They detachable and can be copied and used on other EC2s
* **Security Group**
  + A virtual firewall around your cloud instances (usually your EC2)
  + Allows you to restrict what IP addresses and ports are allowed through to your EC2
* **VPC (Virtual Private Cloud)**
  + Virtual network to connect your cloud resources
  + Essentially a LAN abstraction
    - Make sure that all instances can communicate with each other but are isolated from the outside world
* **IAM (Identity Access Management)**
  + NOT security for the applications you deploy on AWS
  + It is security/permissions for your AWS account
  + Policy is permission to do something (Like spin up an EC2)
    - Users
      * People who are given polices
    - Roles
      * AWS services that you can give policies
      * A machine that you give polices
        + Ex You can grant an EC2 to spin up more EC2 instances
        + Load balancer is an example of an ec2 that needs a role that gives it polices to create more EC2s
* **Route53**
  + Amazon’s Domain name service
  + If you want to buy a domain for your application
* **S3** (Simple Storage Service)
  + Object based storage
    - Buckets
      * A container for objects
    - Objects
      * Files
  + Allows you to store any type of file and access it on Amazon
  + Very popular service
  + **Static Website hosting**
    - Hosting the front end of an application on an s3 bucket
  + **S3 Glacier**
    - Another version of S3 for cold storage
    - Its cheaper than traditional S3 however getting objects from glacier can take several hours
      * Not suitable for in use web applications
    - Use case is for holding old records
* **RDS (Relational Database Service)**
  + A relational database in the cloud
    - Really just an EC2 that has been optimized to serve as a database
* **Snowball and Snowmobile**
  + Moving massive amounts of data between an on premise database to a cloud database could literally takes years/decades
    - Even if with fastest internet possible
  + AWS provides snowball and snowmobile to PHYSICALLY copy and move the data

**Serverless Computing**

* Creating full stack applications without directly commissioning EC2s to run your application
  + You are letting AWS take care all the deployment and scaling details
  + You as programmer spend more time developing and less time on DevOps
* **API Gateway**
  + API layers endpoints
* **AWS Lambda**
  + Handler functions
* **Dynamo DB**
  + A database to store information
* Pros
  + Minimizes time spent configuring your application for deployment
  + Scales incredibly well
    - Literally scales per function call
* Cons
  + Lose some configuration control over the application
    - Harder to set up custom environments for your application to run
  + Pins you to the cloud provider
    - Can’t easily migrate your code

Linux

* Operating System originally designed by Linus Torvalds
  + Open source version of Unix
    - Operating system developed by AT&T
* Very popular for cloud servers
  + Free!!!!!
  + Lightweight (less bloat than Windows)
  + Open source the code highly vetted for security
* Distributions of Linux (very similar but slightly versions of Linux)
  + RedHat
  + Debian
  + Ubuntu
    - Amazon Linux
* Most Linux distributions have not GUI
  + Just Terminal
  + GUIs have made you weak!!!
* Common Linux commands
  + Ls
  + Cd
  + Yum or deb for intsalling
  + Sudo
  + Grep
  + Echo
  + Cat
  + Man
  + Nano
  + Vim :q!
  + Su
  + Touch
  + Mkdir
  + Chmod