Cloud Computing

What is Cloud Computing?

- Cloud computing is the on-demand delivery of compute power, database storage, applications, and other IT resources
- Through a cloud services platform with pay-as-you-go pricing
- You can provision exactly the right type and size of computing resources you need
- You can access as many resources as you need, almost instantly
- Simple way to access servers, storage, databases and a set of application services
- Amazon Web Services owns and maintains the network-connected hardware required for these application services, while you provision and use what you need via a web application.

The Deployment Models of the Cloud

Private Cloud:	Public Cloud:	Hybrid Cloud:
Cloud services used by a single organization, not exposed to the public.	**Cloud resources owned and operated by a thirdparty cloud service provider delivered over the Internet.**	**Keep some servers on premises and extend some capabilities to the Cloud**
Complete control	**Six Advantages of Cloud Computing**	**Control over sensitive assets in your private infrastructure**
Security for sensitive applications		**Flexibility and costeffectiveness of the public cloud**
Meet specific business needs		

The Five Characteristics of Cloud Computing

- On-demand self service:
  - Users can provision resources and use them without human interaction from the service provider
- Broad network access:
  - Resources available over the network, and can be accessed by diverse client platforms
- Multi-tenancy and resource pooling:
  - Multiple customers can share the same infrastructure and applications with security and privacy
  - Multiple customers are serviced from the same physical resources
- Rapid elasticity and scalability:

    - Automatically and quickly acquire and dispose resources when needed
    - Quickly and easily scale based on demand
- **Measured service:**
    - Usage is measured, users pay correctly for what they have used

Six Advantages of Cloud Computing

- **Trade capital expense (CAPEX) for operational expense (OPEX)**
    - Pay On-Demand: don't own hardware
    - Reduced Total Cost of Ownership (TCO) & Operational Expense (OPEX)
- **Benefit from massive economies of scale**
    - Prices are reduced as AWS is more efficient due to large scale
- **Stop guessing capacity**
    - Scale based on actual measured usage
- **Increase speed and agility**
- **Stop spending money running and maintaining data centers**
- **Go global in minutes:** leverage the AWS global infrastructure

Problems solved by the Cloud

- **Flexibility:** change resource types when needed
- **Cost-Effectiveness:** pay as you go, for what you use
- **Scalability:** accommodate larger loads by making hardware stronger or adding additional nodes
- **Elasticity:** ability to scale out and scale-in when needed
- **High-availability and fault-tolerance:** build across data centers
- **Agility:** rapidly develop, test and launch software applications

Types of Cloud Computing

- **Infrastructure as a Service (IaaS)**
    - Provide building blocks for cloud IT
    - Provides networking, computers, data storage space
    - Highest level of flexibility
    - Easy parallel with traditional on-premises IT
- **Platform as a Service (PaaS)**
    - Removes the need for your organization to manage the underlying infrastructure

    - Focus on the deployment and management of your applications
- **Software as a Service (SaaS)**
    - Completed product that is run and managed by the service provider

Example of Cloud Computing Types

- **Infrastructure as a Service:**
    - Amazon EC2 (on AWS)
    - GCP, Azure, Rackspace, Digital Ocean, Linode
- Platform as a Service:
    - Elastic Beanstalk (on AWS)
    - Heroku, Google App Engine (GCP), Windows Azure (Microsoft)
- Software as a Service:
    - Many AWS services (ex: Rekognition for Machine Learning)
    - Google Apps (Gmail), Dropbox, Zoom

Pricing of the Cloud – Quick Overview

- AWS has 3 pricing fundamentals, following the pay-as-you-go pricing model
- **Compute:**
    - Pay for compute time
- **Storage:**
    - Pay for data stored in the Cloud
- **Data transfer OUT of the Cloud:**
    - Data transfer IN is free
- Solves the expensive issue of traditional IT

AWS Cloud Use Cases

- AWS enables you to build sophisticated, scalable applications
- Applicable to a diverse set of industries
- Use cases include
    - Enterprise IT, Backup & Storage, Big Data analytics
    - Website hosting, Mobile & Social Apps
    - Gaming

AWS Global Infrastructure

- AWS Regions
- AWS Availability Zones

- AWS Data Centers
- AWS Edge Locations / Points of Presence
- https://infrastructure.aws/

AWS Regions

- AWS has Regions all around the world
- Names can be us-east-1, eu-west-3...
- A region is a **cluster of data centers**
- **Most AWS services are region-scoped**

How to choose an AWS Region?

If you need to launch a new application, where should you do it?

- **Compliance with data governance and legal requirements:** data never leaves a region without your explicit permission
- **Proximity to customers:** reduced latency
- **Available services within a Region:** new services and new features aren't available in every Region
- **Pricing:** pricing varies region to region and is transparent in the service pricing page

AWS Availability Zones

- Each region has many availability zones (usually 3, min is 2, max is 6). Example:
  - ap-southeast-2a
  - ap-southeast-2b
  - ap-southeast-2c
- Each availability zone (AZ) is one or more discrete data centers with redundant power, networking, and connectivity
- They're separate from each other, so that they're isolated from disasters
- They're connected with high bandwidth, ultra-low latency networking

AWS Points of Presence (Edge Locations)

- Amazon has 216 Points of Presence (205 Edge Locations & 11 Regional Caches) in 84 cities across 42 countries
- Content is delivered to end users with lower latency

Tour of the AWS Console

- **AWS has Global Services:**
  - Identity and Access Management (IAM)
  - Route 53 (DNS service)
  - CloudFront (Content Delivery Network)
  - WAF (Web Application Firewall)
- **Most AWS services are Region-scoped:**
  - Amazon EC2 (Infrastructure as a Service)
  - Elastic Beanstalk (Platform as a Service)
  - Lambda (Function as a Service)
  - Rekognition (Software as a Service)
- **Region Table:** https://aws.amazon.com/about-aws/global-infrastructure/regional-product-services

Shared Responsibility Model

- CUSTOMER = RESPONSIBILITY FOR THE SECURITY **IN** THE CLOUD
- AWS = RESPONSIBILITY FOR THE SECURITY **OF** THE CLOUD

Identity and Access Management

What Is IAM?

AWS Identity and Access Management (IAM) is a web service that helps you securely control access to AWS resources. You use IAM to control who is authenticated (signed in) and authorized (has permissions) to use resources.

IAM: Users & Groups

- IAM = Identity and Access Management, Global service
- **Root account** created by default, shouldn't be used or shared
- **Users** are people within your organization, and can be grouped
- **Groups** only contain users, not other groups
- Users don't have to belong to a group, and user can belong to multiple groups

IAM: Permissions

- Users or Groups can be assigned JSON documents called policies
- These policies define the permissions of the users
- In AWS you apply the least privilege principle: don't give more permissions than a user needs

IAM Policies Inheritance

IAM Policies Structure

- Consists of
  - Version: policy language version, always include "2012-10-17"
  - Id: an identifier for the policy (optional)
  - Statement: one or more individual statements (required)
- Statements consists of
  - Sid: an identifier for the statement (optional)
  - Effect: whether the statement allows or denies access (Allow, Deny)
  - Principal: account/user/role to which this policy applied to
  - Action: list of actions this policy allows or denies
  - Resource: list of resources to which the actions applied to

- Condition: conditions for when this policy is in effect (optional)

Example:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:Describe*",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "elasticloadbalancing:Describe*",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "cloudwatch:ListMetrics",
        "cloudwatch:GetMetricStatistics",
        "cloudwatch:Describe*"
      ],
      "Resource": "*"
```

```
  }
 ]
}
```

IAM – Password Policy

- Strong passwords = higher security for your account
- In AWS, you can setup a password policy:
  - Set a minimum password length
  - Require specific character types:
    - including uppercase letters
    - lowercase letters
    - numbers
    - non-alphanumeric characters
- Allow all IAM users to change their own passwords
- Require users to change their password after some time (password expiration)
- Prevent password re-use

IAM Roles for Services

- Some AWS service will need to perform actions on your behalf
- To do so, we will assign permissions to AWS services with IAM Roles
- Common roles:
  - EC2 Instance Roles
  - Lambda Function Roles
  - Roles for CloudFormation

IAM Security Tools

- IAM Credentials Report (account-level)
- a report that lists all your account's users and the status of their various credentials
- IAM Access Advisor (user-level)
- Access advisor shows the service permissions granted to a user and when those services were last accessed.
- You can use this information to revise your policies.

IAM Guidelines & Best Practices

- Don’t use the root account except for AWS account setup
- One physical user = One AWS user
- **Assign users to groups** and assign permissions to groups
- Create a **strong password policy**
- Use and enforce the use of **Multi Factor Authentication (MFA)**
- Create and use Roles for giving permissions to AWS services
- Use Access Keys for Programmatic Access (CLI / SDK)
- Audit permissions of your account with the IAM Credentials Report
- **Never share IAM users & Access Keys**

Shared Responsibility Model for IAM

AWS	YOU
Infrastructure (global network security)	**Users, Groups, Roles, Policies management and monitoring**
Configuration and vulnerability analysis	**Enable MFA on all accounts**
Compliance validation	**Rotate all your keys often, Use IAM tools to apply appropriate permissions, Analyze access patterns & review permissions**

Multi Factor Authentication - MFA

- Users have access to your account and can possibly change configurations or delete resources in your AWS account
- You want to protect your Root Accounts and IAM users
- MFA = password you know + security device you own
- Main benefit of MFA: if a password is stolen or hacked, the account is not compromised

MFA devices options in AWS

- Virtual MFA device (Support for multiple tokens on a single device.)
  - Google Authenticator (phone only)
  - Authy (multi-device)
- Universal 2nd Factor (U2F) Security Key (Support for multiple root and IAM users using a single security key)
  - YubiKey by Yubico (3rd party)
- Hardware Key Fob MFA Device
- Hardware Key Fob MFA Device for AWS GovCloud (US)

How can users access AWS ?

- To access AWS, you have three options:
  - AWS Management Console (protected by password + MFA)
  - AWS Command Line Interface (CLI): protected by access keys
  - AWS Software Developer Kit (SDK) - for code: protected by access keys
- Access Keys are generated through the AWS Console
- Users manage their own access keys
- Access Keys are secret, just like a password. Don't share them
- Access Key ID ~= username
- Secret Access Key ~= password

What's the AWS CLI?

- A tool that enables you to interact with AWS services using commands in your command-line shell
- Direct access to the public APIs of AWS services
- You can develop scripts to manage your resources
- It's open-source https://github.com/aws/aws-cli
- Alternative to using AWS Management Console

What's the AWS SDK?

- AWS Software Development Kit (AWS SDK)
- Language-specific APIs (set of libraries)
- Enables you to access and manage AWS services programmatically
- Embedded within your application
- Supports
  - SDKs (JavaScript, Python, PHP, .NET, Ruby, Java, Go, Node.js, C++)

    - Mobile SDKs (Android, iOS, ...)
    - IoT Device SDKs (Embedded C, Arduino, ...)
- Example: AWS CLI is built on AWS SDK for Python

IAM Section – Summary

- **Users:** mapped to a physical user, has a password for AWS Console
- **Groups:** contains users only
- **Policies:** JSON document that outlines permissions for users or groups
- **Roles:** for EC2 instances or AWS services
- **Security:** MFA + Password Policy
- **AWS CLI:** manage your AWS services using the command-line
- **AWS SDK:** manage your AWS services using a programming language
- **Access Keys:** access AWS using the CLI or SDK
- **Audit:** IAM Credential Reports & IAM Access Advisor

Advance Security Concepts

AWS STS (SecurityToken Service)

- Enables you to create **temporary, limited- privileges credentials** to access your AWS resources
- Short-term credentials: you configure expiration period
- Use cases
    - Identity federation: manage user identities in external systems, and provide them with STS tokens to access AWS resources
    - IAM Roles for cross/same account access
    - IAM Roles for Amazon EC2: provide temporary credentials for EC2 instances to access AWS resources

Amazon Cognito (simplified)

- Identity for your Web and Mobile applications users (potentially millions)
- Instead of creating them an IAM user, you create a user in Cognito

What is Microsoft Active Directory (AD)?

- Found on any Windows Server with AD Domain Services
- Database of objects: User Accounts, Computers, Printers, File Shares, Security Groups
- Centralized security management, create account, assign permissions

AWS Directory Services

- **AWS Managed Microsoft AD**
  - Create your own AD in AWS, manage users locally, supports MFA
  - Establish “trust” connections with your on- premise AD
- **AD Connector**
  - Directory Gateway (proxy) to redirect to on- premise AD, supports MFA
  - Users are managed on the on-premise AD
- **Simple AD**
  - AD-compatible managed directory on AWS
  - Cannot be joined with on-premise AD

AWS IAM Identity Center (successor to AWS Single Sign-On)

- One login (single sign-on) for all your
  - AWS accounts in AWS Organizations
  - Business cloud applications (e.g., Salesforce, Box, Microsoft 365, ...)
  - SAML2.0-enabled applications
  - EC2 Windows Instances
- Identity providers
  - Built-in identity store in IAM Identity Center

Summary

- **IAM**
  - Identity and Access Management inside your AWS account
  - For users that you trust and belong to your company
- **Organizations**: manage multiple AWS accounts
- **Security Token Service (STS)**: temporary, limited-privileges credentials to access AWS resources
- **Cognito**: create a database of users for your mobile & web applications
- **Directory Services**: integrate Microsoft Active Directory in AWS

- **IAM Identity Center**: one login for multiple AWS accounts & applications

EC2

What is Amazon EC2?

Amazon Elastic Compute Cloud (Amazon EC2) provides scalable computing capacity in the Amazon Web Services (AWS) Cloud.

- EC2 is one of the most popular of AWS' offering
- EC2 = Elastic Compute Cloud = Infrastructure as a Service
- It mainly consists in the capability of :
  - Renting virtual machines (EC2)
  - Storing data on virtual drives (EBS)
  - Distributing load across machines (ELB)
  - Scaling the services using an auto-scaling group (ASG)
- Knowing EC2 is fundamental to understand how the Cloud works

EC2 sizing & configuration options

- Operating System (OS): Linux, Windows or Mac OS
- How much compute power & cores (CPU)
- How much random-access memory (RAM)
- How much storage space:
  - Network-attached (EBS & EFS)
  - hardware (EC2 Instance Store)
- Network card: speed of the card, Public IP address
- Firewall rules: **security group**
- Bootstrap script (configure at first launch): EC2 User Data

EC2 User Data

- It is possible to bootstrap our instances using an **EC2 User data** script.
- **bootstrapping** means launching commands when a machine starts
- That script is **only run once** at the instance **first start**
- EC2 user data is used to automate boot tasks such as:
  - Installing updates
  - Installing software
  - Downloading common files from the internet
  - Anything you can think of
- The EC2 User Data Script runs with the root user

EC2 Instance Types - Overview

- You can use different types of EC2 instances that are optimised for different use cases (https://aws.amazon.com/ec2/instance-types/)
  - General Purpose
  - Compute Optimized
  - Memory Optimized
  - Storage Optimized
  - Accelerated Computing
- AWS has the following naming convention: m5.2xlarge
- m: instance class
- 5: generation (AWS improves them over time)
- 2xlarge: size within the instance class

General Purpose

- Great for a diversity of workloads such as web servers or code repositories
- Balance between:
  - Compute
  - Memory
  - Networking

Compute Optimized

- Great for compute-intensive tasks that require high performance processors:
  - Batch processing workloads
  - Media transcoding
  - High performance web servers
  - High performance computing (HPC)
  - Scientific modeling & machine learning
  - Dedicated gaming servers

Memory Optimized

- Fast performance for workloads that process large data sets in memory
- Use cases:
  - High performance, relational/non-relational databases
  - Distributed web scale cache stores

- In-memory databases optimized for BI (business intelligence)
- Applications performing real-time processing of big unstructured data

Storage Optimized

- Great for storage-intensive tasks that require high, sequential read and write access to large data sets on local storage
- Use cases:
  - High frequency online transaction processing (OLTP) systems
  - Relational & NoSQL databases
  - Cache for in-memory databases (for example, Redis)
  - Data warehousing applications
  - Distributed file systems

EC2 Instance Types: example

Instance	vCPU	Mem (GiB)	Storage	Network Performance	EBS Bandwidth (Mbps)
t2.micro	**1**	**1**	**EBS-Only**	**Low to Moderate**	
t2.xlarge	**4**	**16**	**EBS-Only**	**Moderate**	
c5d.4xlarge	**16**	**32**	**1 x 400 NVMe SSD**	**Up to 10 Gbps**	**4,750**
r5.16xlarge	**64**	**512**	**EBS Only**	**20 Gbps**	**13,600**
m5.8xlarge	**32**	**128**	**EBS Only**	**10 Gbps**	**6,800**

t2.micro is part of the AWS free tier (up to 750 hours per month)

Introduction to Security Groups

- Security Groups are the fundamental of network security in AWS
- They control how traffic is allowed into or out of our EC2 Instances.
- Security groups only contain allow rules
- Security groups rules can reference by IP or by security group

Deeper Dive

- Security groups are acting as a “firewall” on EC2 instances
- They regulate:
  - Access to Ports
  - Authorised IP ranges – IPv4 and IPv6
  - Control of inbound network (from other to the instance)
  - Control of outbound network (from the instance to other)

Security Groups Diagram

Good to know

- Can be attached to multiple instances
- Locked down to a region / VPC combination
- Does live “outside” the EC2 – if traffic is blocked the EC2 instance won’t see it
- It’s good to maintain one separate security group for SSH access
- If your application is not accessible (time out), then it’s a security group issue
- If your application gives a “connection refused“ error, then it’s an application error or it’s not launched
- All inbound traffic is **blocked** by default
- All outbound traffic is **authorized** by default

Classic Ports to know

- 22 = SSH (Secure Shell) - log into a Linux instance
- 21 = FTP (File Transfer Protocol) – upload files into a file share
- 22 = SFTP (Secure File Transfer Protocol) – upload files using SSH
- 80 = HTTP – access unsecured websites
- 443 = HTTPS – access secured websites
- 3389 = RDP (Remote Desktop Protocol) – log into a Windows instance

EC2 Instance Launch Types

- On Demand Instances: short workload, predictable pricing
- Reserved: (1 & 3 years)
  - **Reserved Instances**: long workloads
  - **Convertible Reserved Instances**: long workloads with flexible instances
- Savings Plans (1 & 3 years): commitment to an amount of usage, long workload
- Spot Instances: short workloads, for cheap, can lose instances
- Dedicated Instances: no other customers will share your hardware
- Dedicated Hosts: book an entire physical server, control instance placement
- Capacity Reservations: reserve capacity in a specific AZ for any duration

On Demand Instance

- Pay for what you use:
  - Linux or Windows - billing per second, after the first minute
  - All other operating systems - billing per hour
- Has the highest cost but no upfront payment
- No long-term commitment
- Recommended for **short-term** and **un-interrupted workloads**, where you can't predict how the application will behave

Reserved Instances

- Up to 72% discount compared to On-demand
- You reserve a specific instance attributes (Instance Type, Region, Tenancy, OS)
- Reservation Period – 1 year (+discount) or 3 years (+++discount)
- Payment Options – No Upfront (+), Partial Upfront (++), All Upfront (+++)
- Reserved Instance's Scope – Regional or Zonal (reserve capacity in an AZ)
- Recommended for steady-state usage applications (think database)
- You can buy and sell in the Reserved Instance Marketplace
- Convertible Reserved Instance

    - Can change the EC2 instance type, instance family, OS, scope and tenancy
    - Up to 66% discount

Savings Plans

- Get a discount based on long-term usage (up to 72% - same as RIs)
- Commit to a certain type of usage ($10/hour for 1 or 3 years)
- Usage beyond EC2 Savings Plans is billed at the On-Demand price
- Locked to a specific instance family & AWS region (e.g., M5 in us-east-1)
- Flexible across:
    - Instance Size (e.g., m5.xlarge, m5.2xlarge)
    - OS (e.g., Linux, Windows)
    - Tenancy (Host, Dedicated, Default)

Spot Instances

- Can get a discount of up to 90% compared to On-demand
- Instances that you can “lose” at any point of time if your max price is less than the current spot price
- The MOST cost-efficient instances in AWS
- Useful for workloads that are resilient to failure
    - Batch jobs
    - Data analysis
    - Image processing
    - Any distributed workloads
    - Workloads with a flexible start and end time
- Not suitable for critical jobs or databases

Dedicated Hosts

- A physical server with EC2 instance capacity fully dedicated to your use
- Allows you to address compliance requirements and use your existing server- bound software licenses (per-socket, per-core, pe—VM software licenses)
- Purchasing Options:
    - On-demand – pay per second for active Dedicated Host
    - Reserved - 1 or 3 years (No Upfront, Partial Upfront, All Upfront)
- The most expensive option

- Useful for software that have complicated licensing model (BYOL – Bring Your Own License)
- Or for companies that have strong regulatory or compliance needs

Dedicated Instances

- Instances run on hardware that's dedicated to you
- May share hardware with other instances in same account
- No control over instance placement (can move hardware after Stop / Start)

Capacity Reservations

- Reserve On-Demand instances capacity in a specific AZ for any duration
- You always have access to EC2 capacity when you need it
- No time commitment (create/cancel anytime), no billing discounts
- Combine with Regional Reserved Instances and Savings Plans to benefit from billing discounts
- You're charged at On-Demand rate whether you run instances or not
- Suitable for short-term, uninterrupted workloads that needs to be in a specific AZ

Which purchasing option is right for me?

- On demand: coming and staying in resort whenever we like, we pay the full price
- Reserved: like planning ahead and if we plan to stay for a long time, we may get a good discount.
- Savings Plans: pay a certain amount per hour for certain period and stay in any room type (e.g., King, Suite, Sea View, ...)
- Spot instances: the hotel allows people to bid for the empty rooms and the highest bidder keeps the rooms. You can get kicked out at any time
- Dedicated Hosts: We book an entire building of the resort
- Capacity Reservations: you book a room for a period with full price even you don't stay in it

Price Comparison Example – m4.large – us-east-1

Price Type	Price (per hour)
On-Demand	**$0.10**
Spot Instance (Spot Price)	**$0.038 - $0.039 (up to 61% off)**
Reserved Instance (1 year)	**$0.062 (No Upfront) - $0.058 (All Upfront)**
Reserved Instance (3 years)	**$0.043 (No Upfront) - $0.037 (All Upfront)**
EC2 Savings Plan (1 year)	**$0.062 (No Upfront) - $0.058 (All Upfront)**
Reserved Convertible Instance (1 year)	**$0.071 (No Upfront) - $0.066 (All Upfront)**
Dedicated Host	**On-Demand Price**
Dedicated Host Reservation	**Up to 70% off**
Capacity Reservations	**On-Demand Price**

Shared Responsibility Model for EC2

AWS	USER
Infrastructure (global network security)	**Security Groups rules**
Isolation on physical hosts	**Operating-system patches and updates**
Replacing faulty hardware	**Software and utilities installed on the EC2 instance**
Compliance validation	**IAM Roles assigned to EC2 & IAM user access management, Data security on your instance**

EC2 Section – Summary

- EC2 Instance: AMI (OS) + Instance Size (CPU + RAM) + Storage + security groups + EC2 User Data
- Security Groups: Firewall attached to the EC2 instance
- EC2 User Data: Script launched at the first start of an instance
- SSH: start a terminal into our EC2 Instances (port 22)
- EC2 Instance Role: link to IAM roles
- Purchasing Options: On-Demand, Spot, Reserved (Standard + Convertible + Scheduled), Dedicated Host, Dedicated Instance

EC2 - Auto Scaling and Storage

Scalability & High Availability

- Scalability means that an application / system can handle greater loads by adapting.
- There are two kinds of scalability:
  - Vertical Scalability

    - Horizontal Scalability (= elasticity)
- Scalability is linked but different to High Availability
- Let’s deep dive into the distinction, using a call center as an example

Vertical Scalability

- Vertical Scalability means increasing the size of the instance
- For example, your application runs on a t2.micro
- Scaling that application vertically means running it on a t2.large
- Vertical scalability is very common for non distributed systems, such as a database.
- There’s usually a limit to how much you can vertically scale (hardware limit)

Horizontal Scalability

- Horizontal Scalability means increasing the number of instances / systems for your application
- Horizontal scaling implies distributed systems.
- This is very common for web applications / modern applications
- It’s easy to horizontally scale thanks the cloud offerings such as Amazon EC2

High Availability

- High Availability usually goes hand in hand with horizontal scaling
- High availability means running your application / system in at least 2 Availability Zones
- The goal of high availability is to survive a data center loss (disaster)

High Availability & Scalability For EC2

- Vertical Scaling: Increase instance size (= scale up / down)
    - From: t2.nano - 0.5G of RAM, 1 vCPU
    - To: u-12tb1.metal – 12.3 TB of RAM, 448 vCPUs
- Horizontal Scaling: Increase number of instances (= scale out / in)
    - Auto Scaling Group
    - Load Balancer
- High Availability: Run instances for the same application across multi AZ
    - Auto Scaling Group multi AZ

- Load Balancer multi AZ

Scalability vs Elasticity (vs Agility)

Scalability	Elasticity	Agility
ability to accommodate a larger load by making the hardware stronger (scale up), or by adding nodes (scale out)	once a system is scalable, elasticity means that there will be some “auto-scaling” so that the system can scale based on the load. This is “cloud-friendly”: pay-per-use, match demand, optimize costs	(not related to scalability - distractor) new IT resources are only a click away, which means that you reduce the time to make those resources available to your developers from weeks to just minutes.

What is load balancing?

- Load balancers are servers that forward internet traffic to multiple servers (EC2 Instances) downstream.

Why use a load balancer?

- Spread load across multiple downstream instances
- Expose a single point of access (DNS) to your application
- Seamlessly handle failures of downstream instances
- Do regular health checks to your instances
- Provide SSL termination (HTTPS) for your websites
- High availability across zones

Why use an Elastic Load Balancer?

- An ELB (Elastic Load Balancer) is a managed load balancer
  - AWS guarantees that it will be working
  - AWS takes care of upgrades, maintenance, high availability
  - AWS provides only a few configuration knobs

- It costs less to setup your own load balancer but it will be a lot more effort on your end (maintenance, integrations)
- 3 kinds of load balancers offered by AWS:
  - Application Load Balancer (HTTP / HTTPS only) – Layer 7
  - Network Load Balancer (ultra-high performance, allows for TCP) – Layer 4
  - Classic Load Balancer (slowly retiring) – Layer 4 & 7

What's an Auto Scaling Group?

- In real-life, the load on your websites and application can change
- In the cloud, you can create and get rid of servers very quickly
- The goal of an Auto Scaling Group (ASG) is to:
  - Scale out (add EC2 instances) to match an increased load
  - Scale in (remove EC2 instances) to match a decreased load
  - Ensure we have a minimum and a maximum number of machines running
  - Automatically register new instances to a load balancer
  - Replace unhealthy instances
- Cost Savings: only run at an optimal capacity (principle of the cloud)

Auto Scaling Groups Scaling Strategies

- Manual Scaling: Update the size of an ASG manually
- Dynamic Scaling: Respond to changing demand
  - Simple / Step Scaling
    - When a CloudWatch alarm is triggered (example CPU > 70%), then add 2 units
    - When a CloudWatch alarm is triggered (example CPU < 30%), then remove 1
  - Target Tracking Scaling
    - Example: I want the average ASG CPU to stay at around 40%
  - Scheduled Scaling
    - Anticipate a scaling based on known usage patterns
    - Example: increase the min. capacity to 10 at 5 pm on Fridays
- Predictive Scaling
  - Uses Machine Learning to predict future traffic ahead of time
  - Automatically provisions the right number of EC2 instances in advance

- Useful when your load has predictable time - based patterns

ELB & ASG Summary

- High Availability vs Scalability (vertical and horizontal) vs Elasticity vs Agility in the Cloud
- Elastic Load Balancers (ELB)
  - Distribute traffic across backend EC2 instances, can be Multi-AZ
  - Supports health checks
  - 3 types: Application LB (HTTP – L7), Network LB (TCP – L4), Classic LB (old)
- Auto Scaling Groups (ASG)
  - Implement Elasticity for your application, across multiple AZ
  - Scale EC2 instances based on the demand on your system, replace unhealthy
  - Integrated with the ELB

- EBS: Elastic Block Store, Volume is a network drive you can attach to your instances while they run
- EFS: network file system, can be attached to 100s of instances in a region
- EFS-IA: cost-optimized storage class for infrequent accessed files
- FSx for Windows: Network File System for Windows servers
- FSx for Lustre: High Performance Computing Linux file system

EBS Volumes

What’s an EBS Volume?

- An EBS (Elastic Block Store) Volume is a network drive you can attach to your instances while they run
- It allows your instances to persist data, even after their termination