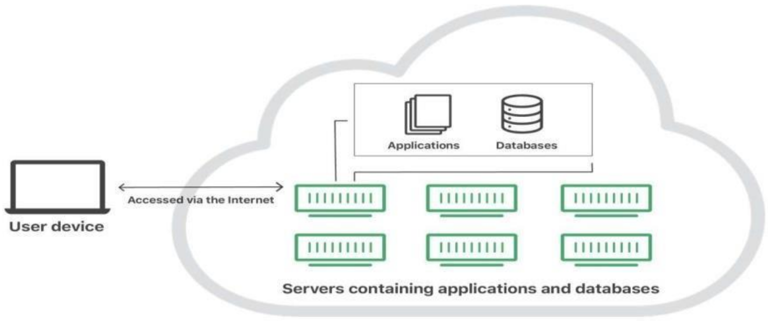
Introduction to Cloud Computing:

What is the cloud: "The cloud" refers to servers that are accessed over the Internet, and the software and databases that run on those servers. Cloud servers are in data centres all over the world. By using cloud computing, users and companies do not have to manage physical serversthemselves or run software applications on their own machines.


Applications
Databases
User device
Accessed via the Internet
Servers containing applications and databases

The Cloud

The cloud enables users to access the same files and applications from almost any device, because the computing and storage takes place on servers in a data centre, instead of locally on the user device. Therefore, a user can log into their Instagram account on a new phone after their old phone breaks and still find their old account in place, with all their photos, videos, and conversation history. It works the same way with cloud email providers like Gmail or Microsoft Office 365, and with cloud storage providers like Dropbox or Google Drive.

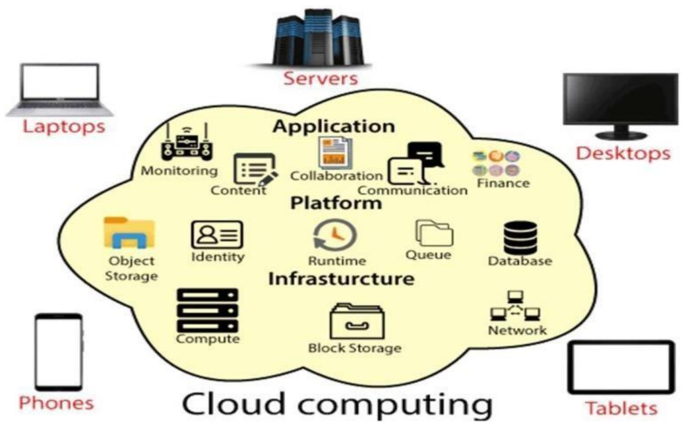
For businesses, switching to cloud computing removes some IT costs and overhead: for instance, they no longer need to update and maintain their own servers, as the cloud vendor they are using will do that. This especially makes an impact for small businesses that may not have been able to afford their own internal infrastructure but can outsource their infrastructure needs affordably via the cloud. The cloud can also make it easier for companies to operate internationally, because employees and customers can access the same files and applications from any location.

Definition of Cloud Computing:

The term "Cloud Computing" refers to services provided by the cloud that is responsible for delivering of computing services such as servers, storage, databases, networking, software, analytics, intelligence, and more, over the Cloud (Internet).

Cloud Computing provides an alternative to the on-premises data center. With an on- premises data center, we must manage everything, such as purchasing and installing hardware, virtualization, installing the operating system, and any other required applications, setting up the network, configuring the firewall, and setting up storage for data. After doing all the set-up, we become responsible for maintaining it through its entire lifecycle.

However, if we choose Cloud Computing, a cloud vendor is responsible for the hardware purchase and maintenance. They also provide a wide variety of software and platform as a service. We can take any required services on rent. The cloud computing services are charged based on usage.



The cloud environment provides an easily accessible online portal that makes handy for the user to manage the compute, storage, network, and application resources.

Advantages of cloud computing:

1. **Cost:** It reduces the huge capital costs of buying hardware and software.
2. **Speed:** Resources can be accessed in minutes, typically within a few clicks.
3. **Scalability:** We can increase or decrease the requirement of resources according tothe business requirements.
4. **Productivity:** While using cloud computing, we put less operational effort. We do not need to apply patching, as well as no need to maintain hardware and software. So, in this way, the IT team can be more productive and focus on achieving business goals.
5. **Reliability:** Backup and recovery of data are less expensive and extremely fast for business continuity.
6. **Security:** Many cloud vendors offer a broad set of policies, technologies, and controls that strengthen our data security.

Cloud computing shares characteristics with:

1. **Client–server model**—*Client–server computing* refers broadly to any distributedapplication that distinguishes between service providers (servers) and service requestors (clients).
2. **Grid computing**—A form of distributed and parallel computing, whereby a 'superand virtual computer' is composed of a cluster of networked, loosely coupled computers acting in concert to perform very large tasks.
3. **Fog computing**—Distributed computing paradigm that provides data, compute, storage and application services closer to the client or near-user edge devices, such asnetwork routers. Furthermore, fog computing handles data at the network level, on smart devices and on the end-user client-side (e.g., mobile devices), instead of sending data to a remotelocation for processing.
4. **Mainframe computer**—Powerful computers used mainly by large organizations for critical applications, typically bulk data processing such as census; industry and consumer statistics; police and secret intelligence services; enterprise resource planning;and financial transaction processing.
5. **Utility computing—**The packaging of computing resources, such as computation andstorage, as a metered service similar to a traditional public utility, such as electricity.

6. **Peer-to-peer—**A distributed architecture without the need for central coordination.Participants are both suppliers and consumers of resources (in contrast to the traditional client-server model).
7. **Green computing—**Study and practice of environmentally sustainable computing orIT.
8. **Cloud sandbox—**A live, isolated computer environment in which a program, code orfile can run without affecting the application in which it runs.

Characteristics of Cloud Computing

1. Agility for organizations
2. Cost reductions, Centralization of infrastructure in locations with lower costs.
3. Device and location independence, which means no maintenance, required.
4. Pay-per-use means utilization and efficiency improvements for systems that are oftenonly 10–20% utilized.
5. Performances are being monitored by IT experts i.e., from the service provider end.
6. Productivity increases which results in multiple users who can work on the same data simultaneously.
7. Time may be saved as information does not need to be re-entered when fields arematched
8. Availability improves with the use of multiple redundant sites
9. Scalability and elasticity via dynamic ("on-demand") provisioning of resources on a fine- grained, self-service basis in near real-time without users having to engineer for peak loads.
10. Self-service interface.
11. Resources thatare abstracted or virtualized.
12. Security can improve due to centralization of data

Cloud Services

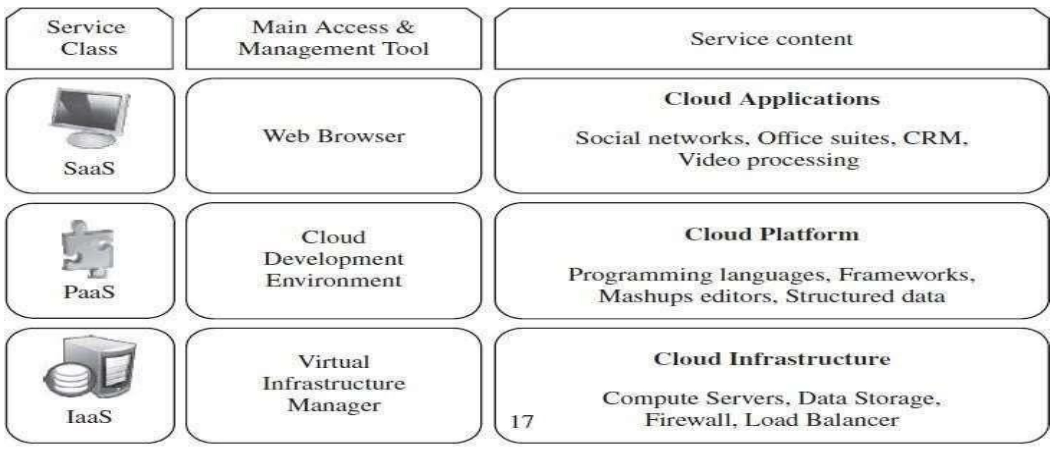
Cloud computing services are divided into three classes, according to the abstractionlevelof the capability provided and the service model of providers, namely:

1. Infrastructure as a Service,
2. Platform as a Service, and Software as a Service.

Infrastructure as a Service

A cloud infrastructure enables on-demand provisioning of servers running several choices of operating systems and a customized software stack. Infrastructure services are considered as the bottom layer of cloud computing systems. Offering virtualized resources (computation, storage, and communication) on demand is known as Infrastructure as a Service(IaaS).

One of the best examples is Amazon Web Services mainly offers IaaS, which inthecase of its EC2 service means offering VMs with a software stack that can be customized similar to how an ordinary physical server would be customized.



Service Class	Main Access & Management Tool	Service content
SaaS	Web Browser	**Cloud Applications** Social networks, Office suites, CRM, Video processing
PaaS	Cloud Development Environment	**Cloud Platform** Programming languages, Frameworks, Mashups editors, Structured data
IaaS	Virtual Infrastructure Manager	17 **Cloud Infrastructure** Compute Servers, Data Storage, Firewall, Load Balancer

FIGURE 1.3. The cloud computing stack.

Platform as a Service

A *cloud platform* offers an environment on which developers create and deploy applications and do not necessarily need to know how many processors or how much memory that applications will be using. In addition, multiple program- ming models and specialized services (e.g., data access, authentication, and payments) are offered as building blocks to new applications.

Google AppEngine, an example of Platform as a Service, offers a scalable environment for developing and hosting Web applications, which should be written in specific programming languages such as Python or Java, and use the services' own proprietary structured object data store.

Software as a Service

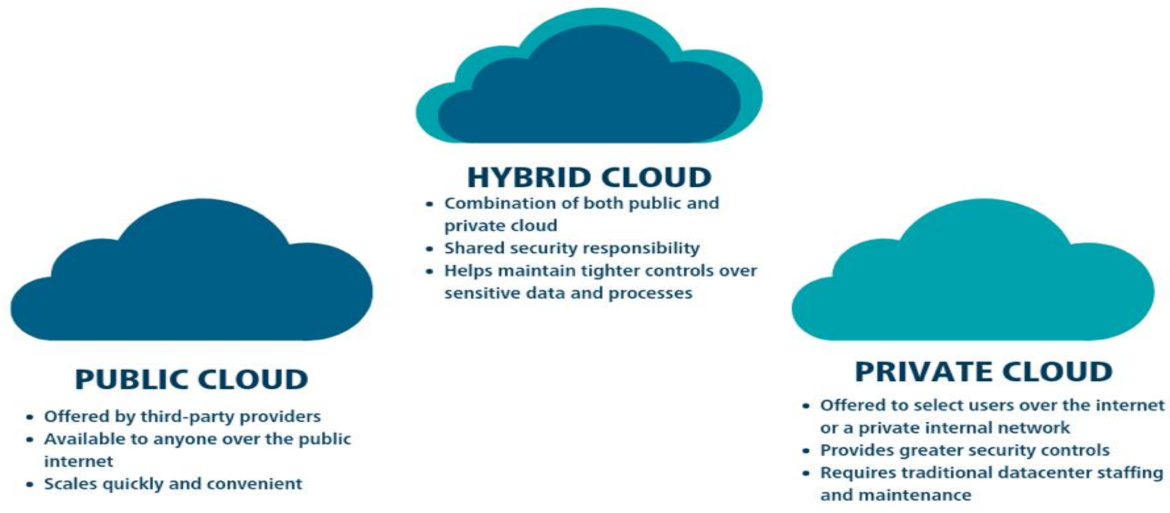
Traditional desktop applications such as word processing and spreadsheet can now be accessed as a service in the Web. This model of delivering applications, known as Software as a Service (SaaS), alleviates the burden of software maintenance for customers and simplifies development and testing for providers.

Salesforce.com, which relies on the SaaS model, offers business productivity applications (CRM) that reside completely on their servers, allowing customers to customize and access applications on demand.

Types of cloud computing

First, you need to determine the cloud computing architecture that you will implement. There are three distinct ways to deploy cloud services: on a public cloud, private cloud, or hybrid cloud. Each has its advantages and disadvantages, and which one you choose will depend on your data and the level of security and management you need:

- **Public Cloud:** Public cloud providers, like Google Cloud Platform (GCP), Amazon Web Services (AWS), and Microsoft Azure, to name a few, deliver computing resources like servers and storage over the Internet. All hardware, software, and other supporting infrastructure is owned and managed by the cloud provider. You access these services and manage your account using a web browser.
- **Private Cloud:** With private clouds, computing resources are governed, owned, and operated by a single organization. A private cloud is one in which we maintain the services and infrastructure on a private network.
- **Hybrid Cloud:** This is the combination of public cloud and private cloud. A hybrid cloud gives your business greater flexibility, more deployment options, and helps optimize your existing infrastructure, security, and compliance.


HYBRID CLOUD
• Combination of both public and private cloud
• Shared security responsibility
• Helps maintain tighter controls over sensitive data and processes
PUBLIC CLOUD
• Offered by third-party providers
• Available to anyone over the public internet
• Scales quickly and convenient
PRIVATE CLOUD
• Offered to select users over the internet or a private internal network
• Provides greater security controls
• Requires traditional datacenter staffing and maintenance

Note:

1) Runtime:
   Runtime is **the period of time when a program is running**. It begins when a program is opened (or executed) and ends with the program is quit or closed.
   The other phases include:

   Edit time
   When the source code of the program is being edited. This phase includes bug fixing, refactoring, and adding new features.

   Compile time
   When the source code is translated into machine code by a compiler. The result is an executable.

   Link time
   When all the necessary machine code components of a program are connected such as external libraries. These connections can be made by the compiler (called static linking) or by the operating system (called dynamic linking).

   Distribution time
   When a program is transferred to a user as an executable or source code. Most of the time a program is downloaded from the internet, but it can also be distributed via CD or USB drive.

   Installation time
   When the distributed program is being installed on the user's computer.

2) Middleware
   Middleware is software and cloud services that provide common services and capabilities to applications and help developers and operators build and deploy applications more efficiently. Middleware acts like the connective tissue between applications, data, and users.

3) Network
   A network consists of two or more computers that are linked in order to share resources (such as printers and CDs), exchange files, or allow electronic communications.
   Types of Network includes: Metropolitan Area Networks (MAN), a Wireless LAN (WLAN), or a Wireless WAN (WWAN).

4) Server
   A server is a computer program or device that provides a service to another computer program and its user, also known as the client.

5) Virtualization
   Virtualization is the "creation of a virtual (rather than actual) version of something, such as a server, a desktop, a storage device, an operating system or network resources".

6) Hypervisor
   A hypervisor is computer software or hardware that enables you to host multiple virtual machines. Each virtual machine is able to run its own programs

The Amazon Web Services (AWS) platform provides more than 200 fully featured services from data centers located all over the world, and is the world's most comprehensive cloud platform.

Amazon web service is an online platform that provides scalable and cost-effective cloud computing solutions. AWS is a broadly adopted cloud platform that offers several on-demand operations like compute power, database storage, content delivery, etc.

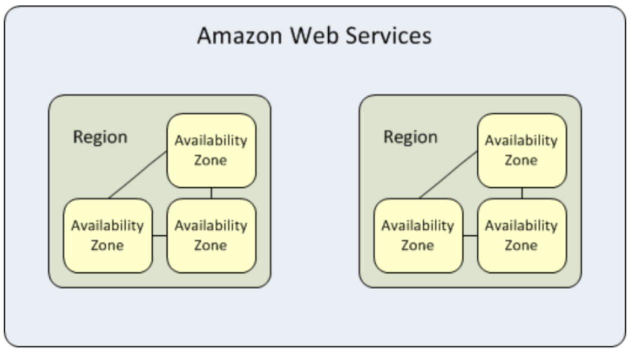
In the year 2006- AWS cloud products were launched

What are AWS Regions?

AWS Regions are separate geographic areas that AWS uses to house its infrastructure. These are distributed around the world so that customers can choose a region closest to them in order to host their cloud infrastructure there. The closer your region is to you, the better, so that you can reduce network latency as much as possible for your end-users. You want to be near the data centres for fast service.

What are AWS Availability Zones?

An AWS Availability Zone (AZ) is the logical building block that makes up an AWS Region. There are currently 69 AZs, which are isolated locations— data centres — within a region. Each region has multiple AZs and when you design your infrastructure to have backups of data in other AZs you are building a very efficient model of resiliency, i.e. a core concept of cloud computing.
See the below image from AWS documentation for a visual representation of Availability Zones within Regions.



Advice for using AWS Availability Zones. There are several reasons why a good strategy with regard to AZs comes in handy in several different situations. Just to cite some of the most common use cases, 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. This is like an emergency load balancer without using an actual load balancer.

In general, AWS Availability Zones give you the flexibility to launch production apps and resources that are highly available, resilient/fault-tolerant, and scalable as compared to using a single data centre. Having more options and backups is better!

Edge locations are AWS data centers designed to deliver services with the lowest latency possible. Amazon has dozens of these data centers spread across the world. They're closer to users than Regions or Availability Zones, often in major cities, so responses can be fast and snappy.

AWS IAM

What is IAM?
AWS Identity and Access Management (IAM) is a service provided by AWS that lets you control access to your AWS resources. IAM enables you to control who can access your resources (authentication) and in which ways (authorization).

Authentication in IAM
Authentication or identity management in AWS IAM consists of the following identities:

Users: An IAM user interacts with your AWS resources from the AWS console and the AWS CLI. By default, a new IAM user has no access to any AWS resource.
Groups: An IAM group consists of IAM users and permissions assigned to those users.
Roles: An IAM role is an entity with a specific set of permissions.

Authorization in IAM
IAM Policies determine authorization or access management in IAM by granting specific permissions to various IAM identities.

What is an IAM Role?
An IAM role is an IAM identity that you can create in your AWS account and assign specific permissions.

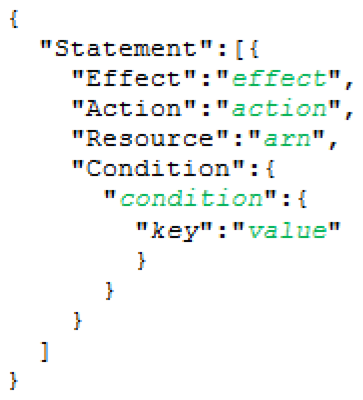
An IAM role is similar to an IAM because it is an IAM identity that has specific permissions associated with it. These permissions determine what the identity can and cannot do.

However, one significant difference between an IAM role and an IAM user is that a role is assumable by anyone who needs it. A role does not have standard long-term credentials (like passwords) associated with it. AWS generates temporary security credentials when an IAM role is assumed.

What is an IAM Policy?
An IAM policy is a document with a set of rules. Each IAM policy grants a specific set of permissions.

Policies are attached to IAM identities like Users, Groups, and Roles. Each IAM policy has a unique name.

Policy Document Structure



```
{
  "Statement":[{
    "Effect":"effect",
    "Action":"action",
    "Resource":"arn",
    "Condition":{
      "condition":{
        "key":"value"
        }
      }
    }
  ]
}
```

IAM Policies are built using a combination of the below elements:

- Version: Defines the version of the policy language. Always use the latest version.
- Statement: This argument is used as a parent element for the different statements in the policy.
- Sid: This is an optional element that allows us to define a statement ID.
- Effect: This element can have the values `Allow` or `Deny`.
- Action: The list of actions related to the policy.

- Resource: Defines the list of resources to which the policy is applied. For resource-based policies, this is optional since the policy applies to the resource that has it attached.
- Principal: Defines the identities that are allowed or denied access to resource-based policies.
- Condition: Defines some conditions under which the policy applies. This element is practical when we need to achieve custom rules for fine-grained access.

Policy Types

IAM Policies are one of the most basic blocks of access management in AWS since they define the permissions of an identity or a resource. For every request, these policies are evaluated, and based on their definition; the requests are allowed or denied. Let's look at the different types of policies that exist in AWS.

- **Identity-based policies** are policies attached to identities(users, groups, roles) and provide them with the required permissions. Identity-based policies could be either **Managed policies or Inline policies**. **Managed policies** are either prepared and managed by AWS for common use cases(AWS managed policies) or custom policies created by users(Customer managed policies) suitable for achieving fine-grained control. **Inline policies** are used when we need to make a policy part of a principal's entity and maintain a strict one-to-one relationship between them.
- **Resource-based policies** are attached directly to resources and specify permissions for specific actions on the resource by some principals.
- **IAM permissions boundaries** define the maximum permissions for an IAM entity and are used as safeguards.
- **Access control lists(ACLs**) are attached to resources and control cross-account permissions for principals from other accounts.
- **Organizations Service Control Policies(SCPs)** specify the maximum level of permissions for an organization's accounts. These policies are used to limit the permissions that can be assigned within member accounts.
- **Session policies** are advanced policies used during temporary sessions for roles or federated users.

Identity Based Policy Example :

Identity-based policies are attached to one or more IAM identities. They explicitly state what a user or role is allowed (or denied) to do.

Alice's (who manages the data of secret agents) identity-based policy (or probably part of it) can look like this:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "FullSecretAgentsAccess",
      "Effect": "Allow",
      "Action": [
        "dynamodb:*"
      ],
      "Resource": "arn:aws:dynamodb:us-west-2:123456789012:table/SecretAgents"
    }
  ]
}
```

Alice is Allowed to do everything in the SecretAgents table in the 123456789012 account in the us-west-2 region. The * wildcard character indicates that she can list (Scan) or query (Query) the items, or even delete a particular item (DeleteItem). The permissions and the APIs almost always map one-to-one to each other, i.e. Alice can use the aws dynamodb delete-item CLI command to delete an item from the table.

It's important to note that there is no Principal in the identity-based policies. Principals are not needed here;

they are assigned to a user, group or role (i.e. an identity) and it doesn't make much sense to define the Principal key (and it's not even allowed).

Resource-based policies Example

They are attached to a resource, for example, to an S3 bucket. Bucket policies are the most well-known examples for resource-based policies.

A bucket policy for the gadgets-for-agents bucket can look like this:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "GadgetsReadOnlyAccess",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:user/alice"
      },
      "Action": [
        "s3:GetObject",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::gadgets-for-agents",
        "arn:aws:s3:::gadgets-for-agents/*"
      ]
    }
  ]
}
```

This policy Allows Alice (who is a user of the 123456789012 account) to list the objects in the bucket as well as to read the content of the objects stored in the bucket.

Resource-based policies have Principals because they need to be tied to a user, role or group of users.

The * as the value of the Principal key refers to every user (can be related to the account or a public user).

IAM Roles vs. Policies

IAM Roles manage who has access to your AWS resources, whereas IAM policies control their permissions.

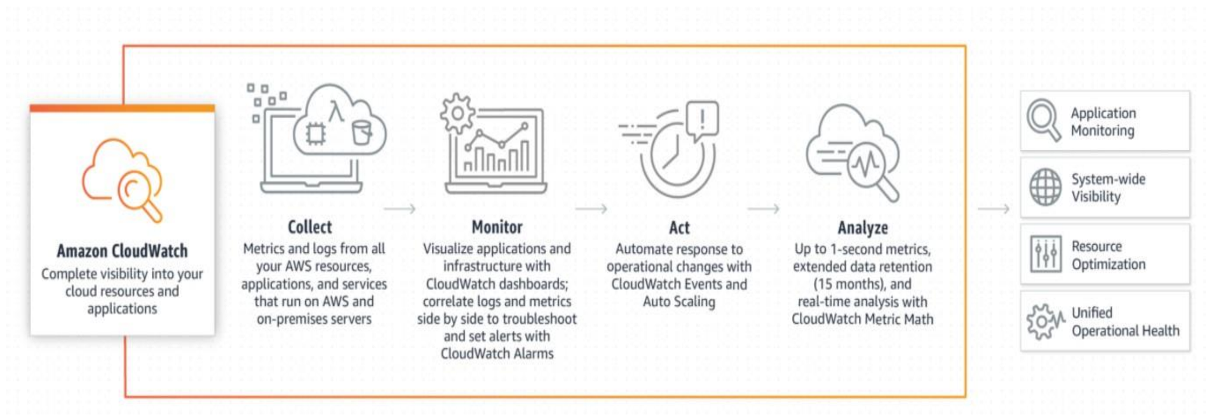
A Role with no Policy attached to it won't have to access any AWS resources.

A Policy that is not attached to an IAM role is effectively unused. Permissions listed in an IAM policy are only enforced when that policy is attached to an IAM identity.

Therefore, you should IAM roles and policies together to manage the security of your AWS resources.

CloudWatch

Amazon CloudWatch monitors your Amazon Web Services (AWS) resources and the applications you run on AWS in real time


Amazon CloudWatch
Complete visibility into your cloud resources and applications
Collect
Metrics and logs from all your AWS resources, applications, and services that run on AWS and on-premises servers
Monitor
Visualize applications and infrastructure with CloudWatch dashboards; correlate logs and metrics side by side to troubleshoot and set alerts with CloudWatch Alarms
Act
Automate response to operational changes with CloudWatch Events and Auto Scaling
Analyze
Up to 1-second metrics, extended data retention (15 months), and real-time analysis with CloudWatch Metric Math
Application Monitoring
System-wide Visibility
Resource Optimization
Unified Operational Health

Default metrics of EC2 instance: Network usage CPU Usage

Metrics:
Metrics are data about the performance of your systemsBasic monitoring: which polls for every 5 minutes Detailed monitoring: which polls for every 1 minute.

Alarm:

CloudWatch Alarms feature allows you to watch CloudWatch metrics and to receive notifications when the metrics fall outside of the levels (high or low thresholds) that you configure

Ex:
If CPU utilization goes beyond the static threshold alarm goes to alarm stateThree states in CW Alarm:
InAlarm
Insufficient
OK state

Events: An Event indicates change in AWS environmentEvent
Resource: Which resource you want to monitor
Event target: to alert the event change through notifications

Logs:
CloudWatch Logs enables you to centralize the logs from all your systems, applications, andAWS services

follow the link and try to reproduce the same:
Ex: https://www.youtube.com/watch?v=F4IE69V-iuw

For reference:
https://www.javatpoint.com/aws-cloudwatch-ec2

Simple Notification Service

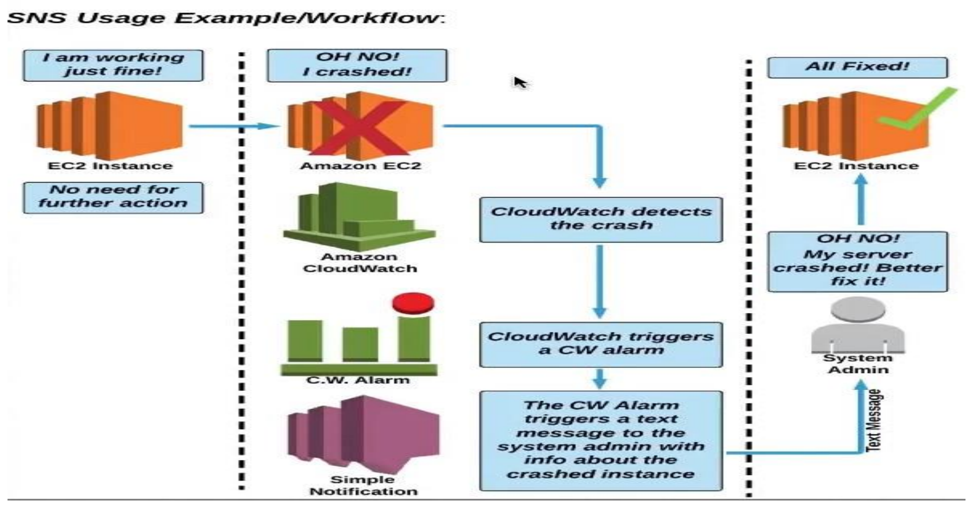
- ***Amazon Simple Notification Service is a notification service provided as part of AmazonWeb Service.***
- ***It provides a low-cost infrastructure for the mass delivery of messages, predominantly to mobile users***

Amazon SNS is a web service that manages sending messages to the subscribing endpoint. There are two clients of SNS:

- Subscribers
- Publishers

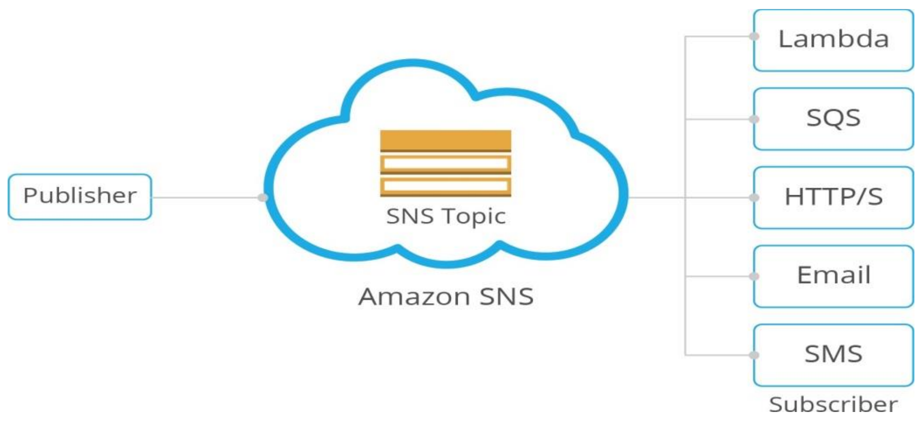
Publishers are also known as producers that produce and send the message to the SNS which is a logical access point

Subscribers such as web servers, email addresses, Amazon SQS queues, AWS Lambda functions receive the message or notification from the SNS over one of the supported protocols (Amazon SQS, email, Lambda, HTTP, SMS).


SNS Usage Example/Workflow:
I am working just fine!
EC2 Instance
No need for further action
OH NO! I crashed!
Amazon EC2
Amazon CloudWatch
C.W. Alarm
Simple Notification
CloudWatch detects the crash
CloudWatch triggers a CW alarm
The CW Alarm triggers a text message to the system admin with info about the crashed instance
All Fixed!
EC2 Instance
OH NO! My server crashed! Better fix it!
System Admin
Text Message

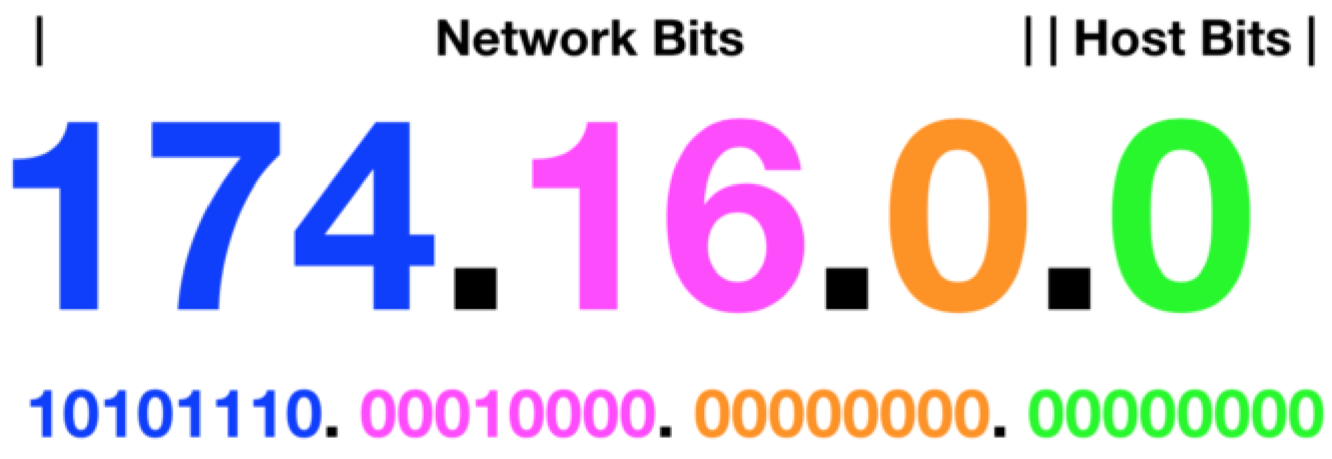
Topic:

An Amazon SNS topic is a logical access point that acts as a communication channel



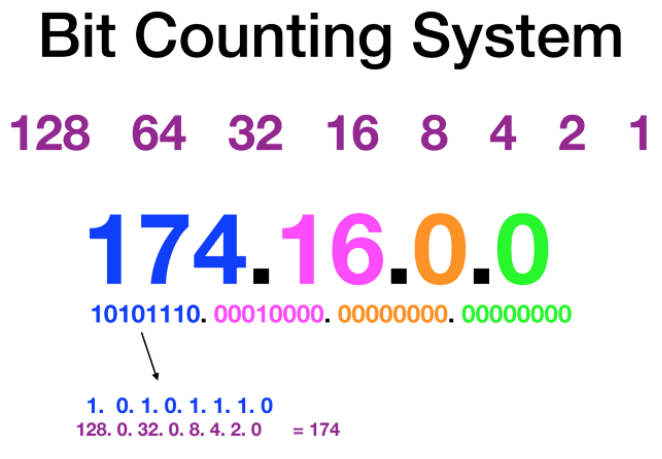
Understanding CIDR Notation and IP Address Range

Finding out what IP addresses are usable within a range can be tricky for people who are not familiar with networking. This article will help you whether you're trying to find the correct IP range for AWS when configuring VPCs and subnets or just want to understand what the numbers in your IP address mean.



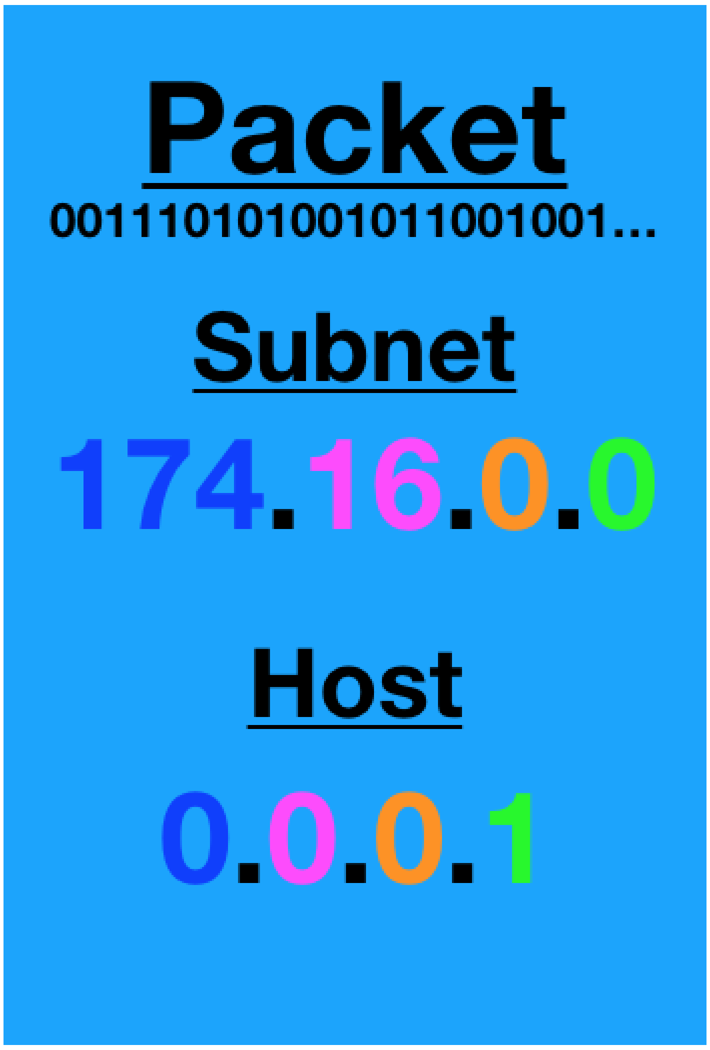
Example IP address.

Let's begin with dissecting this sample IP address. The numbers separated by periods "174.16.0.0" are a numerical representation of underlying binary digits (For example "10101110" is represented by "174"). The first three numbers usually represent the network bits (address used to identify network/subnet) and the last number usually for the host bits (address used to identify a host/destination).



Easy **system to count binary octets.**

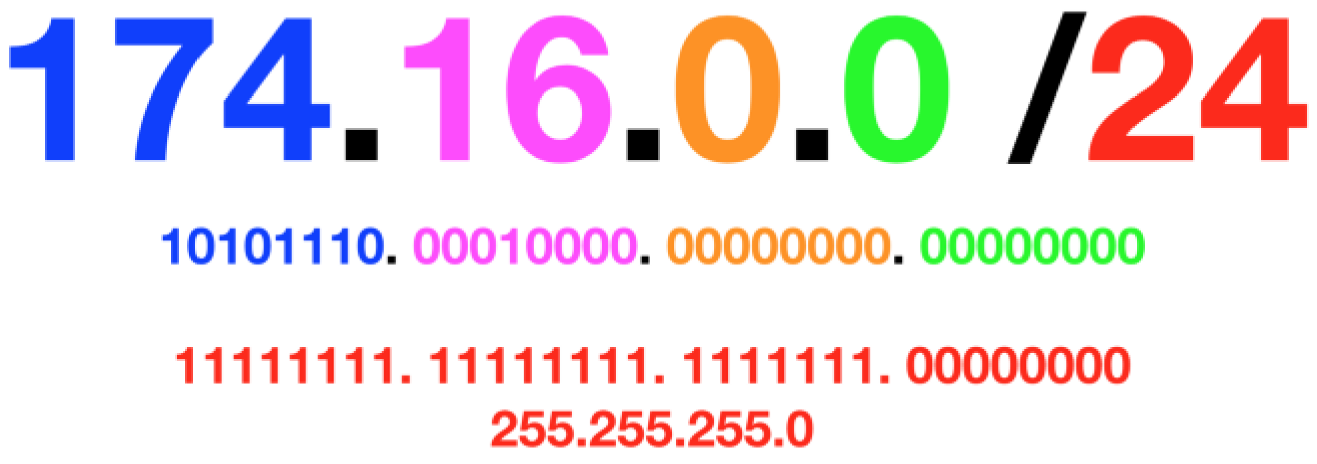
This is an easy system in case you're unfamiliar with binary numbers. Each bit is either "1" to represent "on" or a "0" to represent "off". All 8 bits in an octet (0.0.0.0.0.0.0.0) can be represented by numeric value (128.64.32.16.8.4.2.1) and the bit within the octet decides whether that value is "on" or "off". So to write "2" as a binary octet you would write "00000010" and to write "3" as a binary octet you would write "00000011".



Example of a packet being sent from the web to 174.16.0.1

Here’s an example of a packet that would be sent from the web to your device when you make a request to a website. You can imagine this packet being filled with information you’re downloading and waiting to receive. When the packet is sent from the web it is given the appropriate IP address it needs to arrive to. In this case our “174.16.0.1” is interpreted in two layers. First “174.16.0.0” (the network bits) tells the network that this packet is destined for the 174.16.0.0 subnet. Your modem/router with a host address of 0.0.0.1 would be a part of this subnet and would pick up that packet. The packet would then be routed to the device that requested it by your router.

So far so good but this brings up an issue of scarcity. Each number represents 8 bits (called an octet) for a total of 32 bits. This means that the total number of possible IP addresses is 2^{36} (~4 billion). This finite number becomes a problem when you consider there’s twice the amount of people in the world than available IP addresses, not to mention people who have different IP addresses for their laptop, phone or even washing machine. This is where CIDR notation comes in.


174.16.0.0 /24

10101110. 00010000. 00000000. 00000000

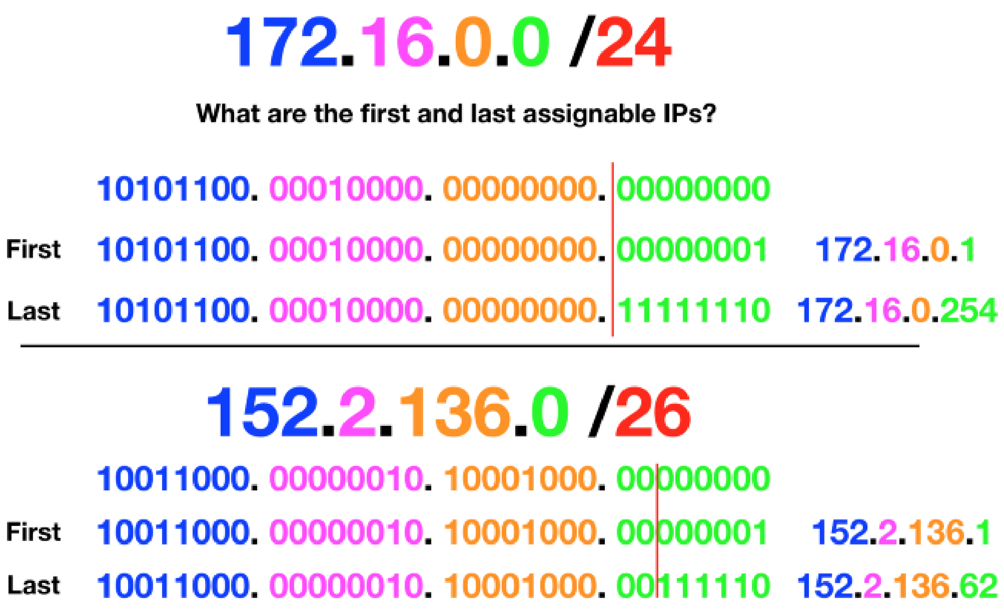
11111111. 11111111. 1111111. 00000000

255.255.255.0

CIDR Classless Inter Domain Routing.

CIDR notation saved us from running out of IP addresses with some neat mathematics. An easy way to understand CIDR is to think of the notation on the end as the amount of bits that will be allocated to the network. In this example "/24" would mean that the first 24 bits are allocated to the network (10101110.00010000.00000000) and the remaining 8 bits would be allocated to the host (.00000000). Another way of writing "/24" would be "11111111.11111111.11111111.00000000" the 1s representing the first 24 bits as "on". This could also be written as "255.255.255.0" which is called the "subnet mask".

The host bits are what we need to identify an IP range. In IP addresses, the host bits reserve all 0s (00000000) for the network address and all 1s (11111111) for broadcasting address. Because those are always used we subtract 2 from our possible IP addresses. The first usable IP address will then be the network bits "00000000"+1 (00000001) . In this example that would mean "174.16.0.0" is the network address and "174.16.0.1" is the first usable IP address. The last usable IP will be the host bits all turned on except the last (11111110). In this example that would be "174.16.0.254". For this example, our IP range would be "174.16.0.1/24" to "174.16.0.254/24". So what happens when the CIDR notation tells you there's more or less than 24 network bits ?



172.16.0.0 /24

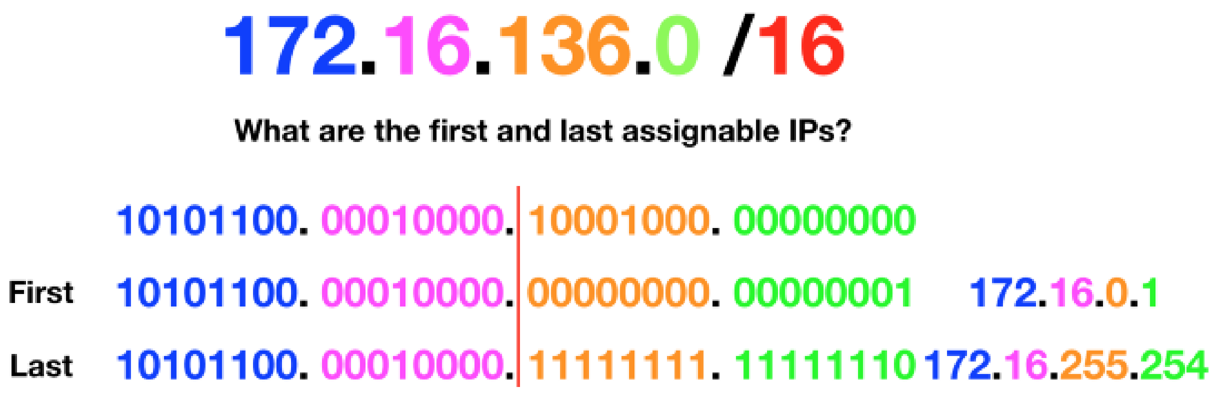
What are the first and last assignable IPs?

```
      10101100. 00010000. 00000000.|00000000
First 10101100. 00010000. 00000000.|00000001   172.16.0.1
Last  10101100. 00010000. 00000000.|11111110   172.16.0.254
```

152.2.136.0 /26

```
      10011000. 00000010. 10001000. 00|000000
First 10011000. 00000010. 10001000. 00|000001   152.2.136.1
Last  10011000. 00000010. 10001000. 00|111110   152.2.136.62
```

In this example the host bits are separated by a red line. In our first example we know there are 8 host bits remaining (because 32 bits in total minus 24 network bits = 8 host bits). This means that the total possible IP addresses in this range are 2^8 (256 total IP addresses minus 2 addresses reserved for network and broadcasting). In the next example we've moved the red line to illustrate only having 6 host bits left (because 32 bits minus 26 network bits = 6 host bits). That means the total possible IP addresses in this range are 2^6 (64 total IP addresses minus 2 addresses reserved for network and broadcasting).



172.16.136.0 /16

What are the first and last assignable IPs?

```
      10101100. 00010000.|10001000. 00000000
First 10101100. 00010000.|00000000. 00000001   172.16.0.1
Last  10101100. 00010000.|11111111. 11111110   172.16.255.254
```

Now our CIDR notation tells us that there will be 16 network bits and 16 host bits. Our method will work the same but overlap into the network bits. With more host bits there will be many more usable IP addresses in this range. The total possible IP addresses in this range would be 2^{16} (65536 total IP addresses minus 2 addresses reserved for network and broadcasting). Services like AWS limit the amount of host bits you can use as to not waste any unused IP addresses (typically they have an allowable subnet mask/CIDR notation range of /16 to /28).

Hopefully this article has demystified the concept of CIDR notation for you and you can now comfortably choose an appropriate IP address or range without wondering what all the seemingly random numbers mean.

AWS CloudTrail

- AWS CloudTrail helps you enable governance, compliance, operational, and risk auditing of the AWS account.
- CloudTrail helps to get a history of AWS API calls and related events for the AWS account.
- CloudTrail records actions taken by a user, role, or AWS service.
- CloudTrail tracking includes calls made by using the AWS Management Console, AWS SDKs, Command-line tools (CLI), APIs, and higher-level AWS services (such as AWS CloudFormation)
- CloudTrail helps to identify which users and accounts called AWS, the source IP address the calls were made from, and when the calls occurred.
- CloudTrail is enabled on your AWS account when you create it.
- CloudTrail is per AWS account and per region for all the supported services.
- CloudTrail AWS API call history enables security analysis, resource change tracking, and compliance auditing.
- CloudTrail event history provides a viewable, searchable, and downloadable record of the past 90 days of CloudTrail events.
- CloudTrail integrates with AWS Organizations and provides an organization trail that enables the delivery of events in the management account, delegated administrator account, and all member accounts in an organization to the same S3 bucket, CloudWatch Logs, and CloudWatch Events.
- CloudTrail Insights can be enabled on a trail to help identify and respond to unusual activity.
- CloudTrail Lake helps run fine-grained SQL-based queries on events.

CloudTrail Events

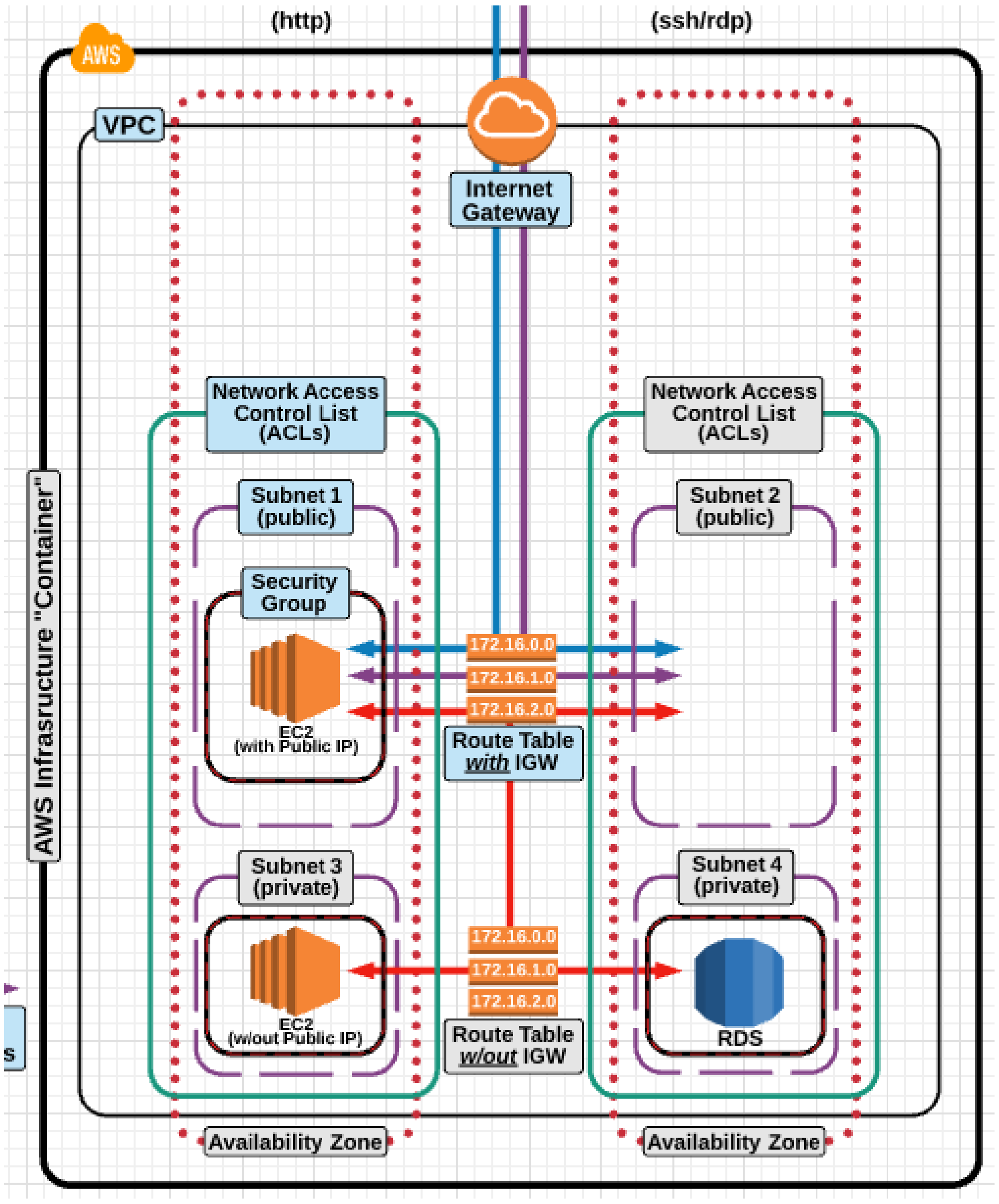
- An event in CloudTrail is the record of activity in an AWS account.
- CloudTrail events provide a history of both API and non-API account activity made through the AWS Management Console, AWS SDKs, command line tools, and other AWS services.
- CloudTrail has the following event types
  - **Management Events**
    - Management events provide information about management or control plane operations that are performed on resources.
    - Includes resource creation, modification, and deletion events.
    - By default, trails log all management events for the AWS account.
  - **Data Events**
    - Data events provide information about the resource or data plane operations performed on or in a resource.
    - Includes data events like reading and writing of objects in S3 or items in DynamoDB.
    - By default, trails don’t log data events for the AWS account.

CloudTrail Enabled Use Cases

- Track changes to AWS resources
  - Can be used to track creation, modification or deletion of AWS resources
- Compliance Aid
  - easier to demonstrate compliance with internal policy and regulatory standards
- Troubleshooting Operational Issues
  - identify the recent changes or actions to troubleshoot any issues
- Security Analysis
  - use log files as inputs to log analysis tools to perform security analysis and to detect user behavior patterns

Virtual Private Cloud

Amazon Virtual Private Cloud (Amazon VPC) lets you provision a logically isolated section of the AWS Cloud where you can launch AWS resources in a virtual network that you define like EC2 instance Databases.



Subnet

AWS defines a subnet as a range of IP addresses in your VPC. You can launch AWS resources into a selected subnet. A public subnet can be used for resources connected to the internet and a private subnet for resources not connected to the internet.

The netmask for a default subnet in your VPC is always 20, which provides up to 4,096 addresses per subnet, with few of them reserved for AWS use. The VPC can span multiple availability zones, but the subnet is always mapped to a single availability zone.

Public and Private Subnets

There are two types of subnets: public and private. A public subnet is used for resources that must be connected to the internet; web servers are an example. A public subnet is made public because the main route table sends the subnets traffic that is destined for the internet to the internet gateway.

Private subnets are for resources that don't need an internet connection, or that you want to protect from the internet; database instances are an example.

Internet Gateway

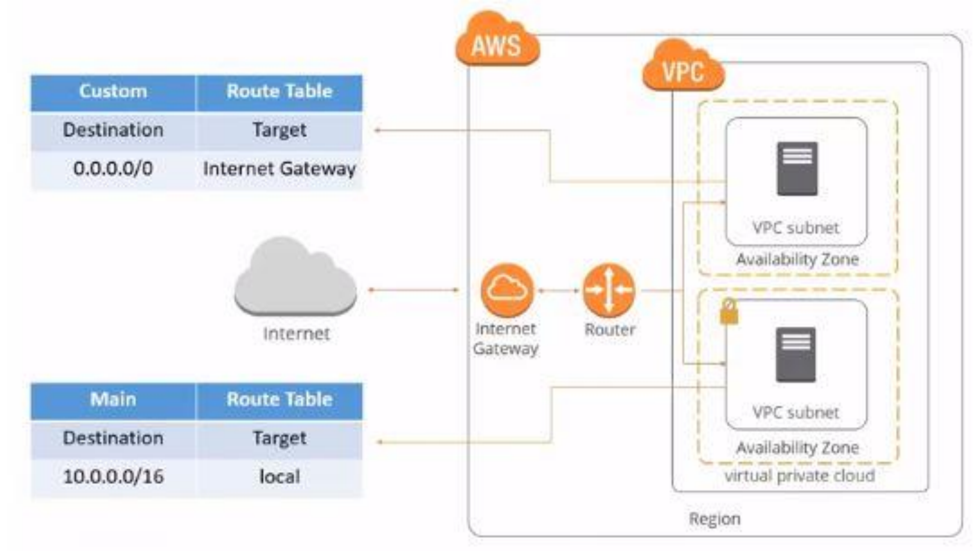
An internet gateway is a redundant, horizontally scaled, and is a highly available VPC component. It enables communication between instances in your VPC and the internet. Therefore, it imposes no availability risks or bandwidth constraints on your network traffic.

To give your VPC the ability to connect to the internet, you need to attach an internet gateway. Only one internet gateway can be attached per VPC. Attaching an internet gateway is the first stage in permitting internet access to instances in your VPC.

Route Table

Amazon defines a route table as a set of rules, called routes, which are used to determine where network traffic is directed.

Each subnet has to be linked to a route table, and a subnet can only be linked to one route table. On the other hand, one route table can have associations with multiple subnets. Every VPC has a default route table, and it is a good practice to leave it in its original state and create a new route table to customize the network traffic routes associated with your VPC. The route table diagram is as shown:



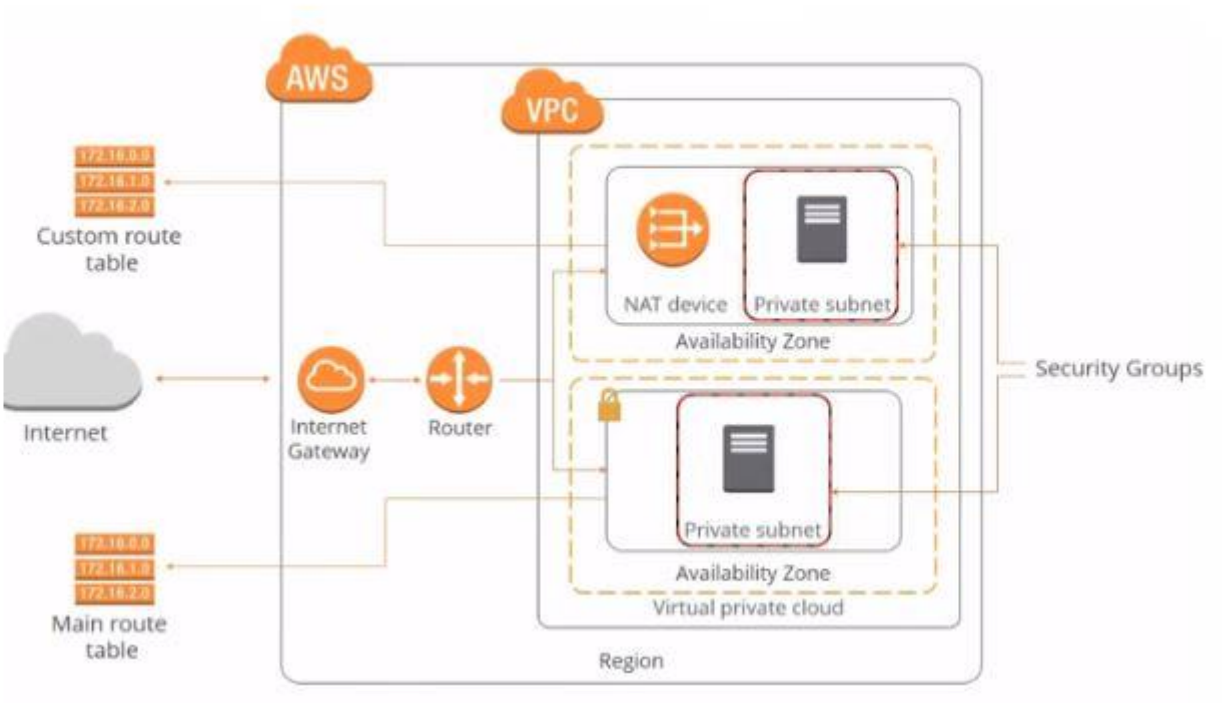
In this example, we've added two route tables: the main route table and the custom route table. The new route table or the custom route table informs the internet gateway to direct

internet traffic to the public subnet. However, the private subnet is still associated with the default route table, the main route table that does not allow internet traffic. All traffic inside the private subnet remains local.

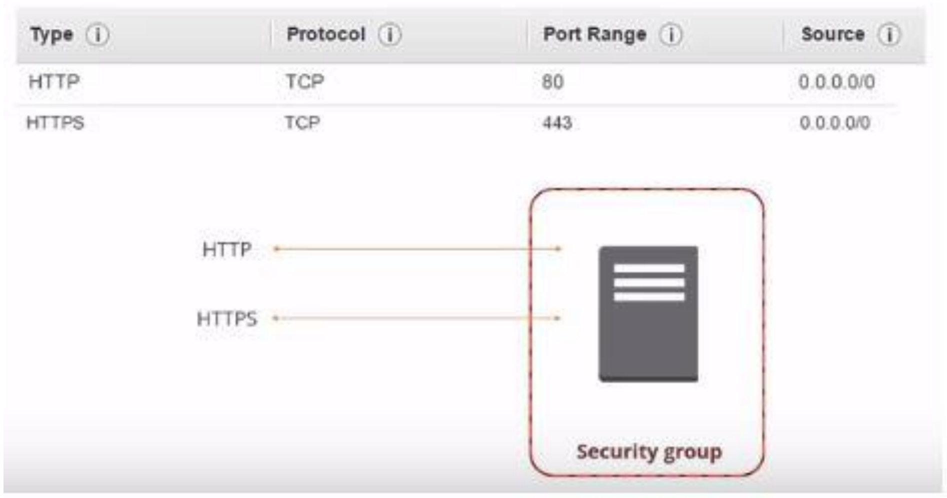
Security Groups and Network ACLs

Amazon defines a security group as a virtual firewall that controls the traffic for one or more instances. Rules are added to each security group, which allows traffic to or from its associated instances. Basically, a security group controls inbound and outbound traffic for one or more EC2 instances. It can be found on both the EC2 and VPC dashboards in the AWS web management console.

Security group diagram:



Security Groups for Web Servers
A web server needs HTTP and HTTPS traffic at the least to access it. The following is an example of the security group table:



Type	Protocol	Port Range	Source
HTTP	TCP	80	0.0.0.0/0
HTTPS	TCP	443	0.0.0.0/0

Here, we allow HTTP and HTTPS, the ports associated with them, and the sources from the internet. All traffic is allowed to those ports, so any other traffic that arrives on different ports would be unable to reach the security group and the instances inside.

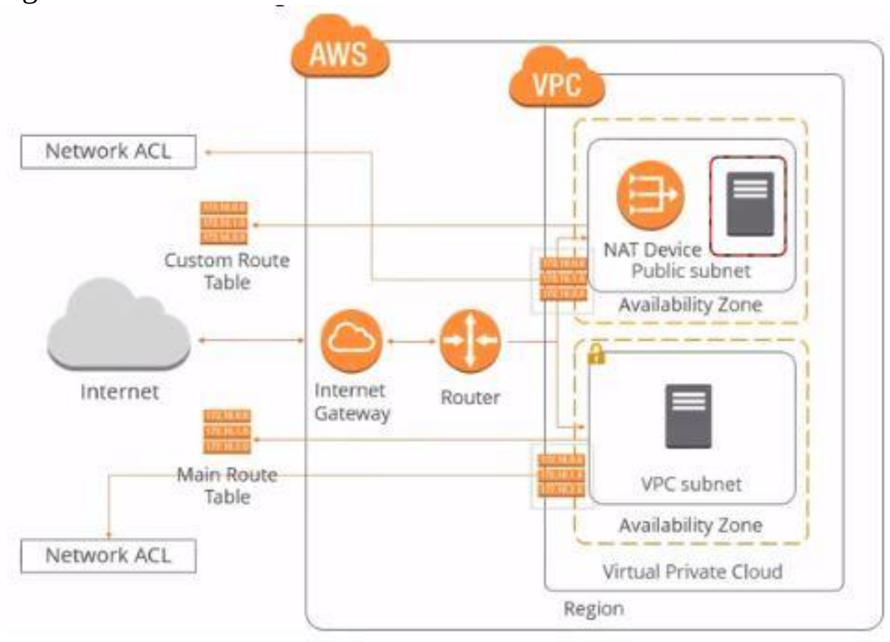
There are a few rules associated with security groups, such as:

- Security groups allow all outbound traffic by default. If you want to tighten your security, this can be done in a similar way as you define the inbound traffic
- Security group rules are always permissive. You can't create rules that deny access
- Security groups are stateful. If a request is sent from your instance, the response traffic for that request is allowed to flow in, regardless of the inbound security group rules
- The rules of a security group can be modified at any time, and apply immediately

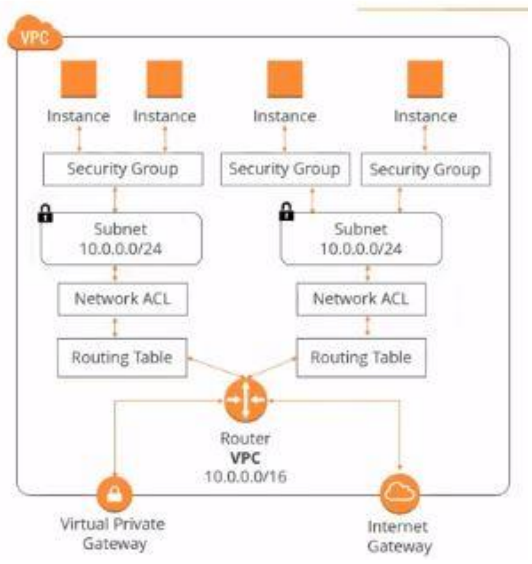
Network ACL

The Network Access Control List (ACL) is an optional security layer for your VPC. It acts as a firewall for controlling traffic flow o and from one or more subnets. Network ACLs can be set up with rules similar to your security groups

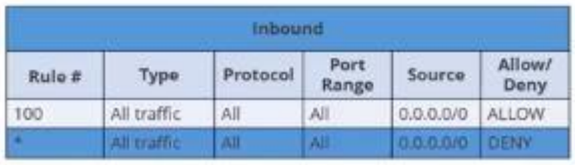
Here is the network diagram:



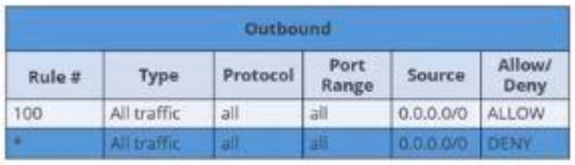
You can find the Network ACLs located somewhere between the root tables and the subnets. Here is a simplified diagram:



You can see an example of the default network ACL, which is configured to allow all traffic to flow in and out of the subnet to which it is associated.



Inbound					
Rule #	Type	Protocol	Port Range	Source	Allow/ Deny
100	All traffic	All	All	0.0.0.0/0	ALLOW
*	All traffic	All	All	0.0.0.0/0	DENY



Outbound					
Rule #	Type	Protocol	Port Range	Source	Allow/ Deny
100	All traffic	all	all	0.0.0.0/0	ALLOW
*	All traffic	all	all	0.0.0.0/0	DENY

Each network ACL includes a rule an * (asterisk) as the rule number. The rule makes sure that if a packet is identified as not matching any of the other numbered rules, traffic is denied. You can't modify or remove this rule.

For traffic coming on the inbound:

- Rule 100 would allow traffic from all sources
- Rule * would deny traffic from all sources

Network ACL Rules

- Every subnet in your VPC must be associated with an ACL, failing which the subnet gets automatically associated with your default ACL.
- One subnet can only be linked with one ACL. On the other hand, an ACL can be linked to multiple subnets.
- An ACL has a list of numbered rules that are evaluated in order, starting with the lowest. As soon as a rule matches, traffic is supplied regardless of any higher-numbered rules that may contradict it. AWS recommends incrementing your rules by a factor of 100. This allows for plenty of room to implement new rules at a later date.
- Unlike security groups, ACLs are stateless; responses to allow inbound traffic are subject to the rules for outbound traffic.

NAT Gateways Working

Your private network can communicate with other public networks with the help of NAT Gateways. They can be used for sending and receiving traffic from a single IP address while keeping hosts' identities private.
All the Instances within your private network are protected with the help of NAT Gateways, as it blocks incoming traffic and allows outgoing traffic.
You can use NAT Gateways in situations when you want to roll out a firmware update but also don't want to allow the software update servers to directly access devices connected to your private network.

AWS NAT Gateway Features

- A NAT Gateway supports TCP, UDP, and ICMP protocols.
- Up to 5 Gbps of bandwidth is supported by the NAT Gateway, which automatically scales 100 Gbps. For more bandwidth split your resources into multiple subnets while creating NAT Gateways for each subnet.
- Upto 1 million packets can be processed per second by the NAT Gateway which can scale up to 10 million packets per second automatically.
- NAT Gateways support IPv4 or IPv6 traffic.
- Up to 55,000 simultaneous connections for each unique destination can be supported by the NAT Gateway.

What is Bastion Host?

A Bastion host is a special-purpose server or an instance that is used to configure to work against the attacks or threats. It is also known as the 'jump box' that acts like a proxy server and allows the client machines to connect to the remote server. It is basically a gateway between the private subnet and the internet. It allows the user to connect private network from an external network and act as proxy to other instances.

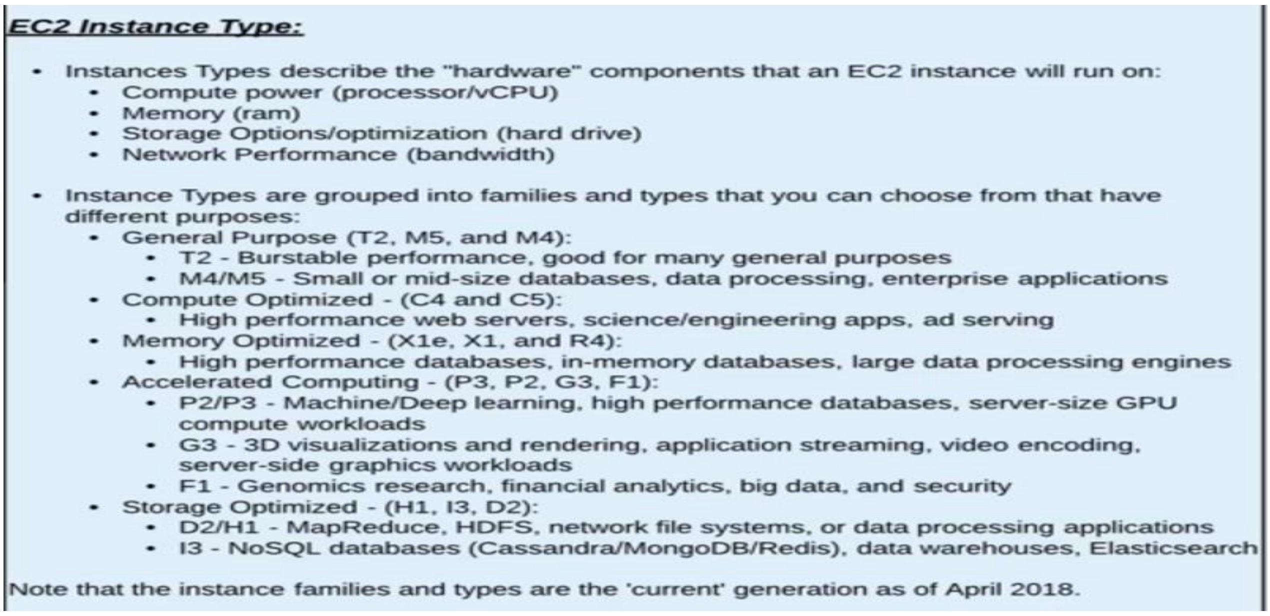
Why to use Bastion Host?

The complete scenario can be explained as suppose there is as clusters of instances in your public network. The public cloud allows you to create some private or isolated section of the cloud which can be used by the user for launching other services which are known as VPC (Virtual Private Network). So the user wants to create a medium or a communication channel to your VPC insecure environment. So there are many methods through which you can do this. The first decision you might use is providing an external IP address. You can assign some services with an external IP address to access it over the internet. But some users might not want to use external IP addresses and want to use SSH tool for more security to connect to the VPC. So now if you are not providing it with the external IP address then the alternate remains is that create another instance on the network which becomes a gateway for the private network to the internet. It acts as a trusted relay for inbound connections. This instance is called Bastion service.

Elastic Cloud Compute

*An **EC2 instance** is a virtual server in Amazon's Elastic Compute Cloud (**EC2)*****EC2 instance***

types

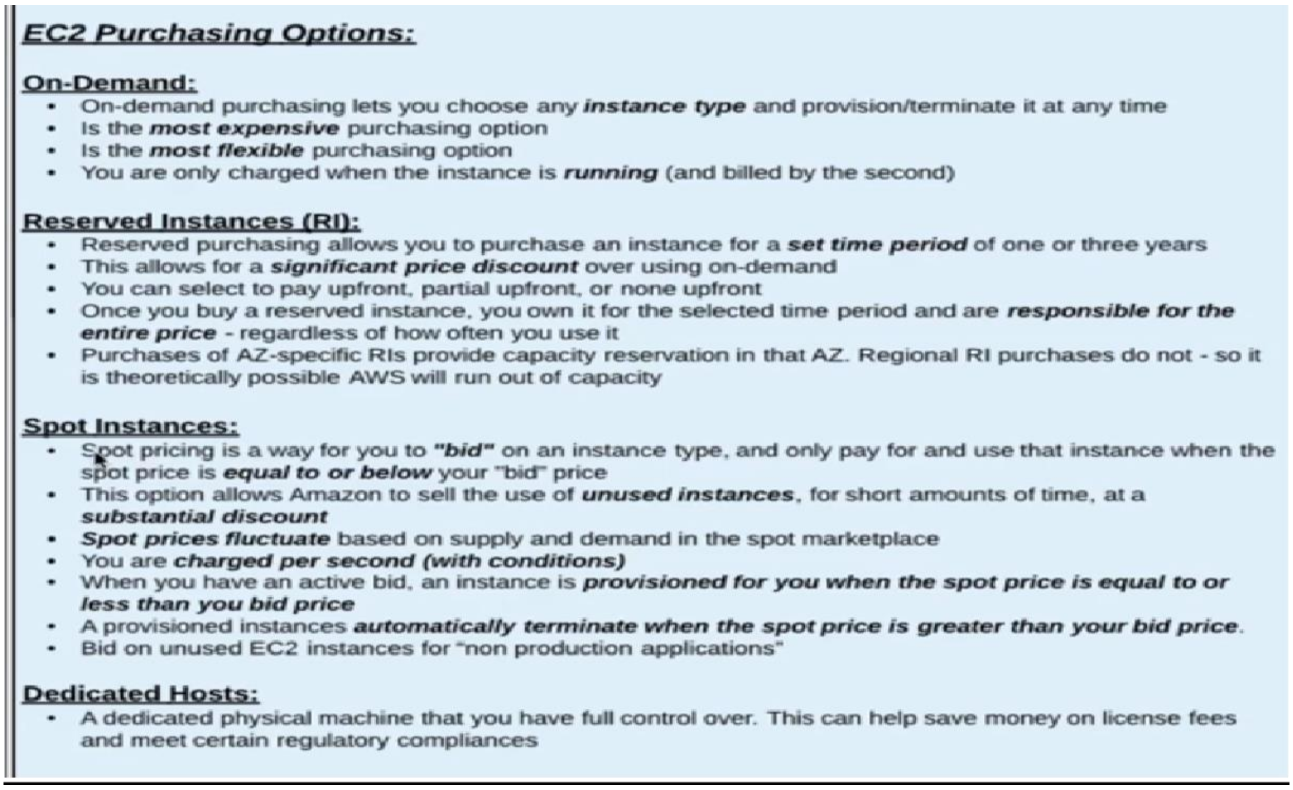


EC2 Instance Type:

- Instances Types describe the "hardware" components that an EC2 instance will run on:
  - Compute power (processor/vCPU)
  - Memory (ram)
  - Storage Options/optimization (hard drive)
  - Network Performance (bandwidth)
- Instance Types are grouped into families and types that you can choose from that have different purposes:
  - General Purpose (T2, M5, and M4):
    - T2 - Burstable performance, good for many general purposes
    - M4/M5 - Small or mid-size databases, data processing, enterprise applications
  - Compute Optimized - (C4 and C5):
    - High performance web servers, science/engineering apps, ad serving
  - Memory Optimized - (X1e, X1, and R4):
    - High performance databases, in-memory databases, large data processing engines
  - Accelerated Computing - (P3, P2, G3, F1):
    - P2/P3 - Machine/Deep learning, high performance databases, server-size GPU compute workloads
    - G3 - 3D visualizations and rendering, application streaming, video encoding, server-side graphics workloads
    - F1 - Genomics research, financial analytics, big data, and security
  - Storage Optimized - (H1, I3, D2):
    - D2/H1 - MapReduce, HDFS, network file systems, or data processing applications
    - I3 - NoSQL databases (Cassandra/MongoDB/Redis), data warehouses, Elasticsearch

Note that the instance families and types are the 'current' generation as of April 2018.

EC2 purchasing options:



EC2 Purchasing Options:

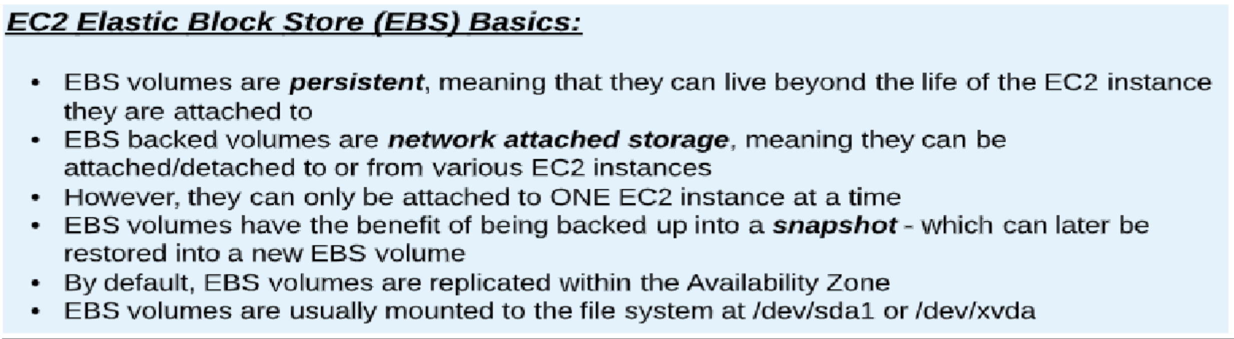
On-Demand:
- On-demand purchasing lets you choose any ***instance type*** and provision/terminate it at any time
- Is the ***most expensive*** purchasing option
- Is the ***most flexible*** purchasing option
- You are only charged when the instance is ***running*** (and billed by the second)

Reserved Instances (RI):
- Reserved purchasing allows you to purchase an instance for a ***set time period*** of one or three years
- This allows for a ***significant price discount*** over using on-demand
- You can select to pay upfront, partial upfront, or none upfront
- Once you buy a reserved instance, you own it for the selected time period and are ***responsible for the entire price*** - regardless of how often you use it
- Purchases of AZ-specific RIs provide capacity reservation in that AZ. Regional RI purchases do not - so it is theoretically possible AWS will run out of capacity

Spot Instances:
- Spot pricing is a way for you to ***"bid"*** on an instance type, and only pay for and use that instance when the spot price is ***equal to or below*** your "bid" price
- This option allows Amazon to sell the use of ***unused instances***, for short amounts of time, at a ***substantial discount***
- ***Spot prices fluctuate*** based on supply and demand in the spot marketplace
- You are ***charged per second (with conditions)***
- When you have an active bid, an instance is ***provisioned for you when the spot price is equal to or less than you bid price***
- A provisioned instances ***automatically terminate when the spot price is greater than your bid price***.
- Bid on unused EC2 instances for "non production applications"

Dedicated Hosts:
- A dedicated physical machine that you have full control over. This can help save money on license fees and meet certain regulatory compliances

Elastic Block Store

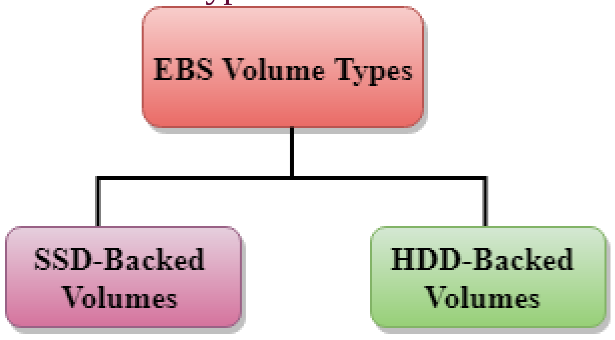


EC2 Elastic Block Store (EBS) Basics:

- EBS volumes are ***persistent***, meaning that they can live beyond the life of the EC2 instance they are attached to
- EBS backed volumes are ***network attached storage***, meaning they can be attached/detached to or from various EC2 instances
- However, they can only be attached to ONE EC2 instance at a time
- EBS volumes have the benefit of being backed up into a ***snapshot*** - which can later be restored into a new EBS volume
- By default, EBS volumes are replicated within the Availability Zone
- EBS volumes are usually mounted to the file system at /dev/sda1 or /dev/xvda

- EBS stands for **Elastic Block Store**.
- EC2 is a virtual server in a cloud while EBS is a virtual disk in a cloud.
- Amazon EBS allows you to create storage volumes and attach them to the EC2 instances.
- Once the storage volume is created, you can create a file system on the top of these volumes, and then you can run a database, store the files, applications or you can even use them as a block device in some other way.
- Amazon EBS volumes are placed in a specific availability zone, and they are automatically replicated to protect you from the failure of a single component.
- EBS volume does not exist on one disk, it spreads across the Availability Zone. EBS volume is a disk which is attached to an EC2 instance.
- EBS volume attached to the EC2 instance where windows or Linux is installed known as Root device of volume.

EBS Volume Types

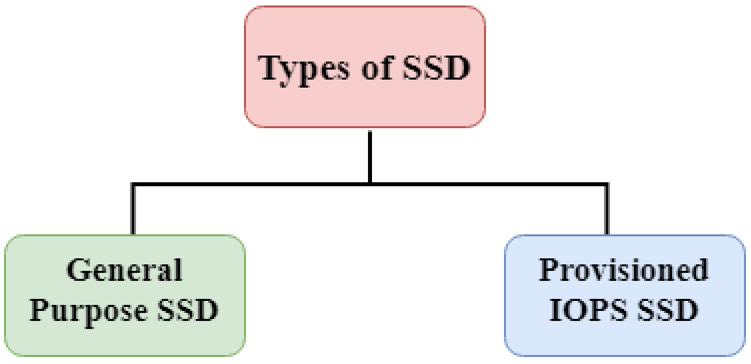


Amazon EBS provides two types of volume that differ in performance characteristics and price. EBS Volume types fall into two parts:

- SSD-backed volumes
- HDD-backed volumes

SSD

- SSD stands for solid-state Drives.
- In June 2014, SSD storage was introduced.
- It is a general purpose storage.
- It supports up to 4000 IOPS which is quite very high.
- SSD storage is very high performing, but it is quite expensive as compared to HDD (Hard Disk Drive) storage.
- SSD volume types are optimized for transactional workloads such as frequent read/write operations with small I/O size, where the performance attribute is IOPS.



SSD is further classified into two parts:

- General Purpose SSD
- Provisioned IOPS SSD

General Purpose SSD

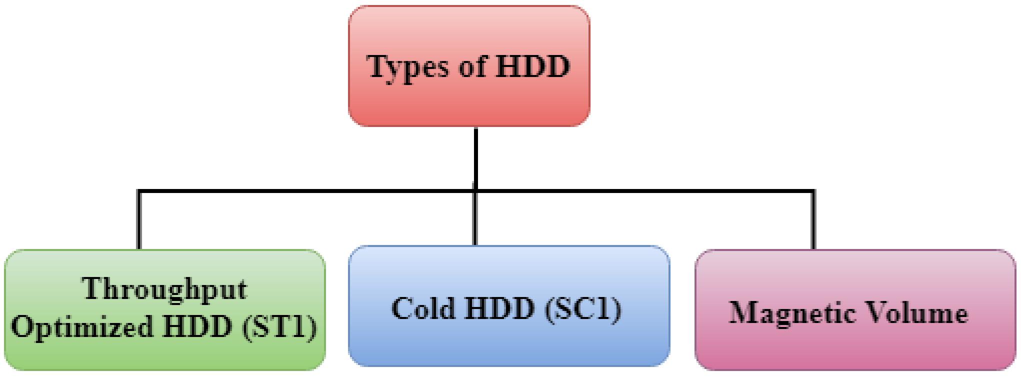
- General Purpose SSD is also sometimes referred to as a GP2.
- It is a General purpose SSD volume that balances both price and performance.
- You can get a ratio of 3 IOPS per GB with up to 10,000 IOPS and the ability to burst up to 3000 IOPS for an extended period of time for volumes at 3334 GiB and above. For example, if you get less than 10,000 IOPS, then GP2 is preferable as it gives you the best performance and price.

Provisioned IOPS SSD

- It is also referred to as IO1.
- It is mainly used for high-performance applications such as intense applications, relational databases.
- It is designed for I/O intensive applications such as large relational or NOSQL databases.
- It is used when you require more than 10,000 IOPS.

HDD

- It stands for Hard Disk Drive.
- HDD based storage was introduced in 2008.
- The size of the HDD based storage could be between 1 GB to 1TB.
- It can support up to 100 IOPS which is very low.



Throughput Optimized HDD (st1)

- It is also referred to as ST1.
- Throughput Optimized HDD is a low-cost HDD designed for those applications that require higher throughput up to 500 MB/s.
- It is useful for those applications that require the data to be frequently accessed.
- It is used for Big data, Data warehouses, Log processing, etc.
- It cannot be a boot volume, so it contains some additional volume. For example, if we have Windows server installed in a C: drive, then C drive cannot be a Throughput Optimized Hard disk, D: drive or some other drive could be a Throughput Optimized Hard disk.
- The size of the Throughput Hard disk can be 500 GiB to 16 TiB.
- It supports up to 500 IOPS.

Cold HDD (sc1)

- It is also known as SC1.
- It is the lowest cost storage designed for the applications where the workloads are infrequently accessed.
- It is useful when data is rarely accessed.
- It is mainly used for a File server.
- It cannot be a boot volume.
- The size of the Cold Hard disk can be 500 GiB to 16 TiB.
- It supports up to 250 IOPS.

Magnetic Volume

- It is the lowest cost storage per gigabyte of all EBS volume types.
- It is ideal for the applications where the data is accessed infrequently
- It is useful for applications where the lowest storage cost is important.
- Magnetic volume is the only hard disk which is bootable. Therefore, we can say that it can be used as a boot volume.

EBS Volume

- *Launch two ec2 instance in different az's(instance1 & instance2)*
- *Create EBS volume and attach it to instance1*
- *The volumes are attached to instance1 you can verify it by logging into instance1 and executing "lsblk" command, but it's not mounted you can verify it through by running command "df -TH"*
- *Mount the volume to instance1*
    - *Format the disk with ext4: "mkfs -t ext4 /dev/xvdf"*
    - *Create a directory in root: 1. "cd /" 2. "mkdir /mnt/mydisk"*
    - *Mount the disk: "mount /dev/xvdf /mnt/mydisk"*
    - *you can verify that disk is mounted by running "df -TH" command.*
- *Create some files*
- *Take a snapshot*
- *Unmount the disk*
    - ➤ *umount /mnt/mydisk*
    - ➤ *Detach the volume from ec2 instance.*
    - ➤ *delete the volume*
- *Create a new volume from snapshot*
- *Attach the volume to newly created instance2.*
- *Mount the volume to instance2*
    - ➤ *Create a directory in root: 1. cd / 2. mkdir /mnt/mydisk*
    - *mount /dev/xvdf /mnt/mydisk*

Data life cycle Manager:

You can use Amazon Data Lifecycle Manager to automate the creation, retention,and deletion of snapshots taken to back up your Amazon EBS volumes

Amazon machine image (AMI):

- *An AMI is a template that contains the software configuration (operating system,application server, and applications) required to launch your instance.*
- *You can launch multiple instances from a single AMI when you need multipleinstances with the same configuration.*

Difference between Snapshot and AMI

An EBS snapshot is a backup of a single EBS volume. The EBS snapshot contains all the data stored on the EBS volume at the time the EBS snapshot was created.

An AMI image is a backup of an entire EC2 instance. Associated with an AMI image is EBS snapshots. Those EBS snapshots are the backups of the individual EBS volumes attached tothe EC2 instance at the time the AMI image was created.

Simple storage service (S3)

Amazon S3 has a simple web services interface that you can use to store and retrieve any amount of data, at any time, from anywhere onthe web.

Single operation upload:

- *It's a traditional upload where you will upload the object in one part*
- *A single operation upload can upload the file up to 5GB in size.*

Upload object in parts:

- *Using multipart upload, you can upload the large objects up to 5TB.*
- *You can use multipart upload for the objects from 5MB to 5TB in size.*

Rules for bucket naming:

- *Bucket names must be between 3 and 63 characters long.*
- *Bucket names can consist only of lowercase letters, numbers, dots . and hyphens -.*
- *Bucket names must begin and end with a letter or number.*
- *Bucket names must not be formatted as an IP address (for example, 192.168.5.4).*
- *Bucket names can't begin with xn-- (for buckets created after February 2020).*

Limitation of S3 bucket:

- *Only 100 buckets can be created per account.*
- *Can hold unlimited objects*

S3 Storage classes:

- Standard:
  - Designed for general- and all-purpose storage
  - Default storage option
  - 99.999999999% object durability
  - 99.99% object availability
  - Most expensive storage class.
- Reduced Redundancy storage
  - Designed for non-critical objects
  - 99.99% object durability
  - 99.99% object availability
  - Less expensive than standard
- Infrequent access
  - *Designed for less frequently accessed objects.*
  - *99.999999999% object durability*
  - *99.99% object availability*

  Less expensive than reduced redundancy storage

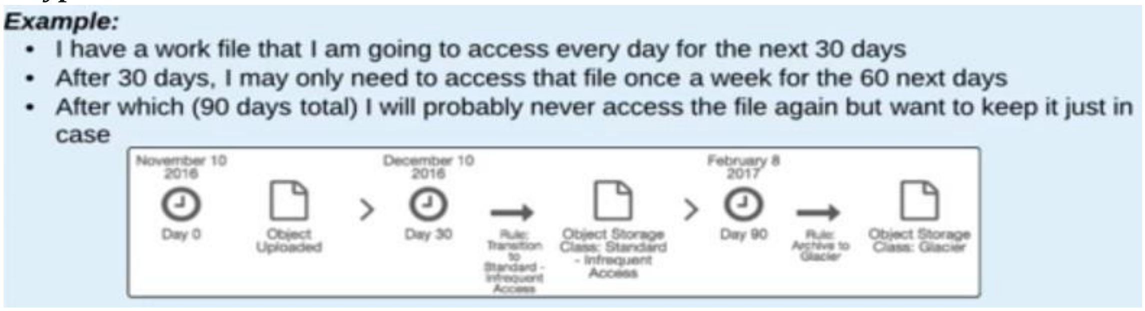
- Glacier
  - *Designed for long term archival storage*
  - *May take several hours to retrieve the objects from this storage*
  - *Cheapest s3 storage class*

S3 Life cycle policy:

An object lifecycle policy is a set of rules that automate the migration of the objectstorage class to different storage class

By default, lifecycle policies are disabled for a bucket

Encryption:



Example:

- I have a work file that I am going to access every day for the next 30 days
- After 30 days, I may only need to access that file once a week for the 60 next days
- After which (90 days total) I will probably never access the file again but want to keep it just in case

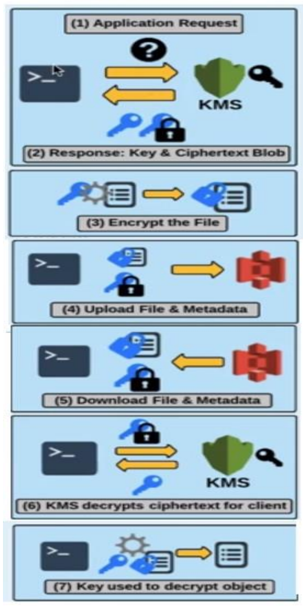
November 10 2016
Day 0
Object Uploaded
December 10 2016
Day 30
Rule: Transition to Standard - Infrequent Access
Object Storage Class: Standard - Infrequent Access
February 8 2017
Day 90
Rule: Archive to Glacier
Object Storage Class: Glacier

Two ways of protecting information with S3

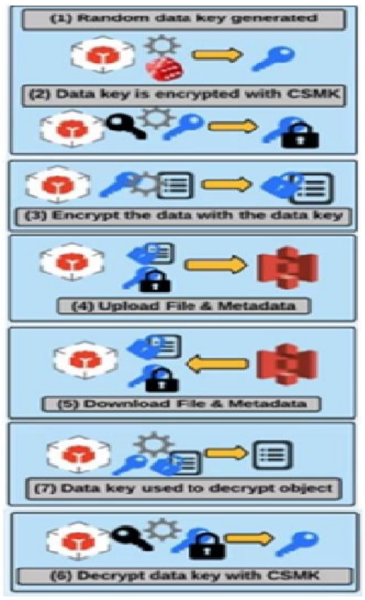
Encryption:

1. **In-transit/Client-side encryption:**

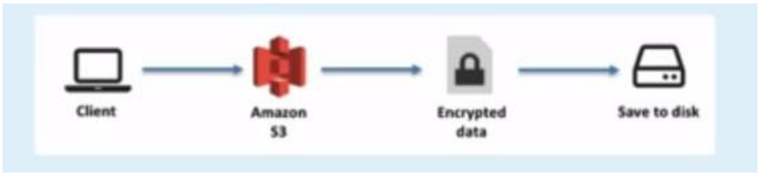
USing KMS


(1) Application Request
KMS
(2) Response: Key & Ciphertext Blob
(3) Encrypt the File
(4) Upload File & Metadata
(5) Download File & Metadata
KMS
(6) KMS decrypts ciphertext for client
(7) Key used to decrypt object

Client side Master KEy


(1) Random data key generated
(2) Data key is encrypted with CSMK
(3) Encrypt the data with the data key
(4) Upload File & Metadata
(5) Download File & Metadata
(7) Data key used to decrypt object
(6) Decrypt data key with CSMK

2. ***Server side/*****At rest*****:***



SSE Data Encryption

Within Amazon S3, Server Side Encryption (SSE) is the simplest data encryption option available. SSE encryption manages the heavy lifting of encryption on the AWS side, and falls into two types: SSE-S3 and SSE-C.

The SSE-S3 option lets AWS manage the key for you, which requires that you trust them with that information. With SSE-S3, you don't have access to see or encrypt data using the key directly, but you can be assured that the raw data you own is encrypted at rest by AWS's standard processes.

The SSE-C option similarly manages encryption and decryption of your data for you, but uses a key provided by you (the customer) and passed in to AWS with each request to encrypt or decrypt. AWS does not store your key with this method, so you are responsible for its safe keeping.

S3 Client-Side Data Encryption

S3 Client-Side Encryption puts all the responsibility for the encryption heavy lifting onto the user. Rather than allowing AWS to encrypt your data, you perform the encryption within your own data center and upload the encrypted data directly to AWS.

S3 Client-Side Encryption also comes in two options: server-side master key storage, and client-side master key storage.

In server-side master key storage, you can store your master key server-side in the AWS KMS (Key Management Service) service, and AWS will provide sophisticated key management software to manage sub-keys based on the master key that is used to encrypt your data.

In client-side master key storage, your master keys aren't stored on AWS's servers, and you take full responsibility for the encryption. Using this second approach is potentially the most secure, as your keys and data are never seen by Amazon servers in an unencrypted state. However, the level of security that you can achieve with this method depends on the integrity of your own processes and technology rather than AWS's.

KMS

AWS Key Management Store (KMS) is a managed service that enables you to easily encrypt your data.

AWS KMS provides a highly available key storage, management, and auditing solution for you to encrypt data within your own applications and control the encryption of stored data across AWS services.

AWS KMS allows you to centrally manage and securely store your keys. These are known as AWS KMS keys (formerly known as customer master keys (CMKs).

AWS KMS Keys

A KMS key consists of:

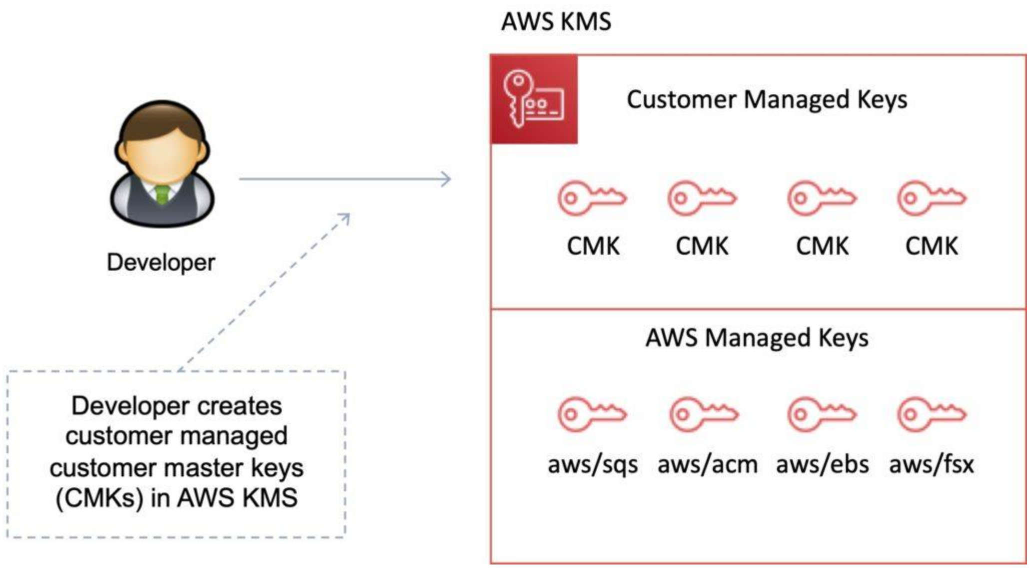
Alias.
Creation date.
Description.
Key state.

Key material (either customer provided or AWS provided).

1. KMS keys are the primary resources in AWS KMS.
2. The KMS key includes metadata, such as the key ID, creation date, description, and key state.
3. The KMS key also contains the key material used to encrypt and decrypt data.
4. AWS KMS supports symmetric and asymmetric KMS keys.
5. KMS keys are created in AWS KMS. Symmetric KMS keys and the private keys of asymmetric KMS keys never leave AWS KMS unencrypted.
6. By default, AWS KMS creates the key material for a KMS key.
7. A KMS key can encrypt data up to 4KB in size.
8. A KMS key can generate, encrypt, and decrypt Data Encryption Keys (DEKs).
9. A KMS key can never be exported from KMS (CloudHSM allows this).

AWS Managed KMS keys:
KMS keys managed by AWS are used by AWS services that interact with KMS to encrypt data.
They can only be used by the service that created them within a particular region.
They are created on the first time you implement encryption using that service.

Customer managed KMS keys:
These provide the ability to implement greater flexibility.
You can perform rotation, governing access, and key policy configuration.
You are able to enable and disable the key when it is no longer required.



Customer Managed KMS keys

Customer managed KMS keys are KMS keys in your AWS account that you create, own, and manage.

You have full control over these KMS keys, including establishing and maintaining their key policies, IAM policies, and grants, enabling and disabling them, rotating their cryptographic material, adding tags, creating aliases that refer to the KMS key, and scheduling the KMS keys for deletion.

Customer managed KMS keys incur a monthly fee and a fee for use in excess of the free tier.

AWS Managed KMS keys

AWS managed KMS keys are KMS keys in your account that are created, managed, and used on your behalf by an AWS service that is integrated with AWS KMS.

You cannot manage these KMS keys, rotate them, or change their key policies.

Auto Scaling

AWS Auto Scaling is exactly what you can understand from its name, a service that automatically manages and adjusts compute resources to maintain a consistent performance of applications, whether it involves scaling up or scaling down the resources. AWS Auto scaling service makes sure that there are enough instances or resources to run the application without any failure.

When a situation arises where servers get overburdened due to the increased load on the application, the AWS auto scaling service will scale up the servers by creating more servers with exactly the same configurations. This is where AMIs come into the picture. An AMI is created for the server on which the application is being hosted. This AMI is used to deploy more servers.

You can also create an AWS auto scaling group, which is basically a collection of instances for the sole purpose of AWS auto scaling. The size or the capacity of the AWS auto scaling group can be managed through the scaling policy.

Scaling Plans

Maintaining current instance level at all times: With this scaling plan, the user can configure an AWS auto scaling group to maintain a fixed number of running instances at all times.

Manual scaling: This scaling plan lets the user specify the desired capacity of the AWS auto scaling groups, and the auto scaling service manages the process of creating or terminating instances on its own.

Scaling based on a schedule: This scaling plan comes in handy in situations where the user can predict when the traffic on the application is going to increase. In such cases, the user can schedule the time when AWS auto scaling should be executed.

Scaling based on demand: This scaling plan lets the user define parameters that control the scaling procedure such as CPU utilization, memory, etc.

There is a total of two steps to configure auto scaling properties:

- First is to create a launch configuration
- Second is to create an AWS auto scaling group

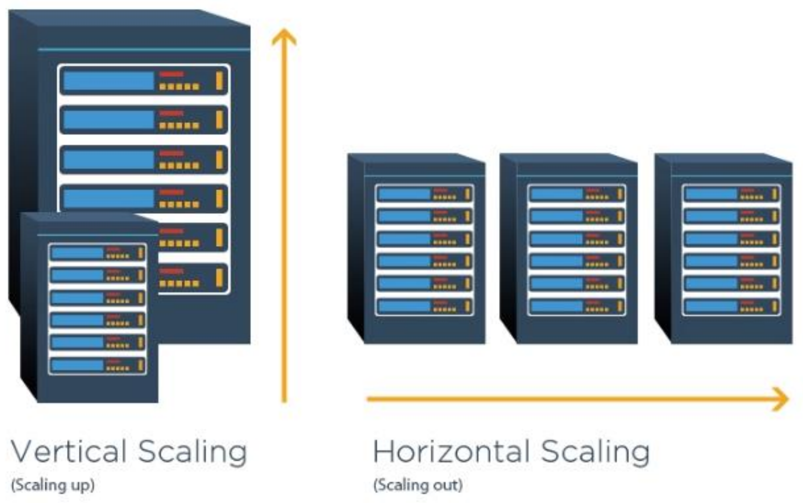
What Is Vertical Scaling?

Vertical scaling (aka scaling up) describes adding additional resources to a system so that it meets demand.

How is this different from horizontal scaling?

While horizontal scaling refers to adding additional nodes, vertical scaling describes adding more power to your current machines. For instance, if your server requires more processing power, vertical scaling would mean upgrading the CPUs. You can also vertically scale the memory, storage, or network speed.

Additionally, vertical scaling may also describe replacing a server entirely or moving a server's workload to an upgraded one.



Advantages of horizontal scaling

- **Scaling is easier from a hardware perspective** - All horizontal scaling requires you to do is add additional machines to your current pool. It eliminates the need to analyze which system specifications you need to upgrade.
- **Fewer periods of downtime** - Because you're adding a machine, you don't have to switch the old machine off while scaling. If done effectively, there may never be a need for downtime and clients are less likely to be impacted.
- **Increased resilience and fault tolerance** - Relying on a single node for all your data and operations puts you at a high risk of losing it all when it fails. Distributing it among several nodes saves you from losing it all.
- **Increased performance** - If you are using horizontal scaling to manage your network traffic, it allows for more endpoints for connections, considering that the load will be delegated among multiple machines.

Disadvantages of horizontal scaling

- **Increased complexity of maintenance and operation** - Multiple servers are harder to maintain than a single server is. Additionally, you will need to add software for load balancing and

possibly virtualization. Backing up your machines may also become a little more complex. You will need to ensure that nodes synchronize and communicate effectively.
- **Increased Initial costs** - Adding new servers is far more expensive than upgrading old ones.

Advantages of vertical scaling

- **Cost-effective** - Upgrading a pre-existing server costs less than purchasing a new one. Additionally, you are less likely to add new backup and virtualization software when scaling vertically. Maintenance costs may potentially remain the same too.
- **Less complex process communication** - When a single node handles all the layers of your services, it will not have to synchronize and communicate with other machines to work. This may result in faster responses.
- **Less complicated maintenance** - Not only is maintenance cheaper but it is less complex because of the number of nodes you will need to manage.
- **Less need for software changes** - You are less likely to change how the software on a server works or how it is implemented.

Disadvantages of vertical scaling

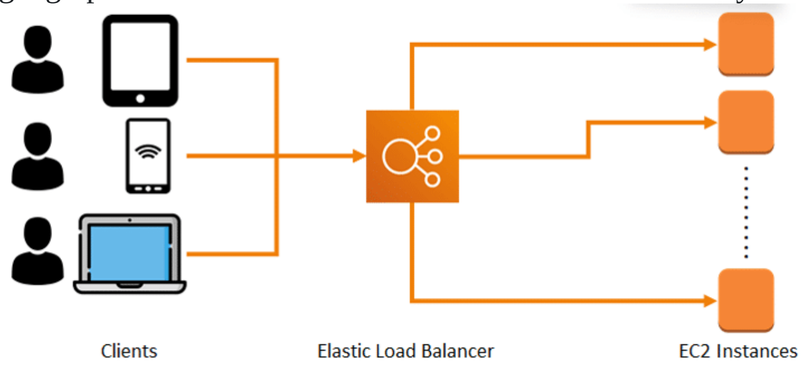
- **Higher possibility for downtime** - Unless you have a backup server that can handle operations and requests, you will need some considerable downtime to upgrade your machine.
- **Single point of failure** - Having all your operations on a single server increases the risk of losing all your data if a hardware or software failure was to occur.
- **Upgrade limitations** - There is a limitation to how much you can upgrade a machine. Every machine has its threshold for RAM, storage, and processing power.

Reference Link to configure the Auto scaling-

https://medium.com/@jawad846/amazon-ec2-auto-scaling-884ea50d2d

Elastic Load Balancer

Load balancer is a service which uniformly distributes network traffic and workloads across multiple servers or cluster of servers. Load balancer in AWS increases the availability and fault tolerance of an application.
AWS Elastic Load Balancer is the single point of contact to all the clients, they can be sent to the nearest geographic instance or the instance with the lowest latency.



The basic working principle is the Elastic Load Balancer accepts incoming traffic from its clients and then routes requests to the targets which the client want. If the load balancer finds an unhealthy target, then it will stop redirecting it's users there and will move with the other healthy targets until that target is declared healthy.

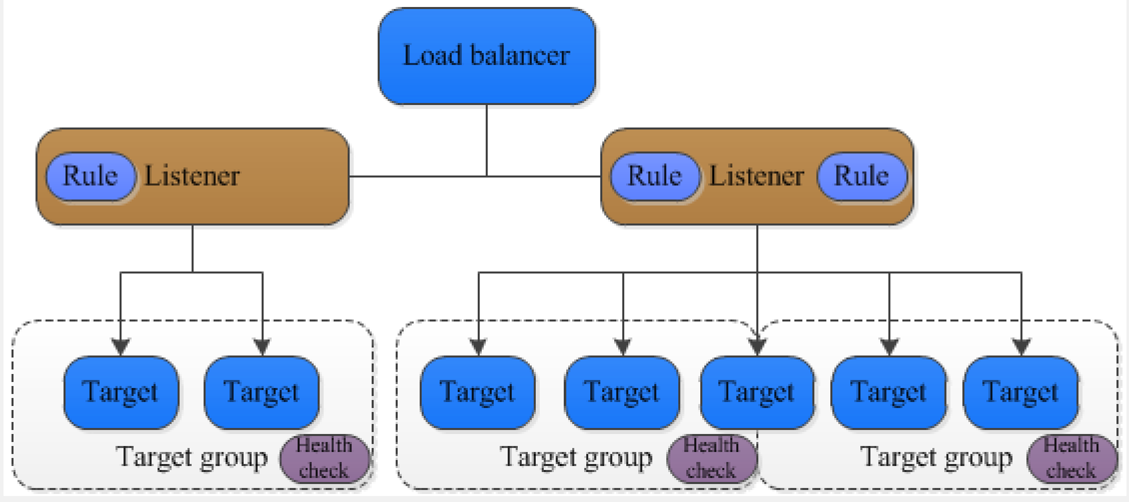
Application Load Balancers

After receiving the request Application Load Balancer analyzes the rules provide by the listener in priority order and determines the rule which has to apply. After that, it selects a target from the target group for the rule action.

An Application Load Balancer functions at the Open Systems Interconnection (OSI) model which is the seventh layer of the OSI model.

The User can analyze the rules of the listener and can modify it by sending it to different target groups based on the content of the application traffic even when the target is associated with multiple target groups.

Addition and removal of tags can do from the load balancers as per your needs. This can done without breaking the flow of your requests of the application.



AWS ELB – Application Load Balancers

Benefits of Application Load balancers-

- Load Balancer's performance improve in Application Load Balancer.
- Access logs containing information compress such that they may not require the additional space.
- Provides benefit for registering targets by IP address, including targets outside the VPC for the load balancer.

Network Load Balancers

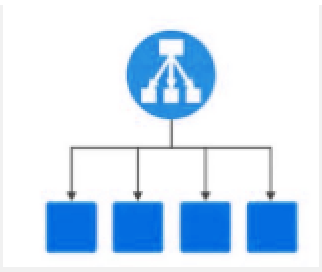
It is the fourth layer of the Open System Interconnection Model. After the load balancer receives a connection request, it selects a target from the group which targets for the default rule.

After enabling the availability zone Elastic Load Balancer creates the load balancer node in the availability zone. Each load balancer node automatically distributes traffic across the registered targets in its Availability Zone only.

Cross-zone Load Balancing enables to distribute traffic across the registered targets in all enabled Availability Zones.

Enabling Multiple Availability Zone can cause harm by increasing the fault tolerance of the applications and it will happen if each target group has at least one target in each enabled Availability Zone.

The problem can overcome in such a way that if one or more target groups do not have a healthy target in an Availability Zone, the IP address for the corresponding subnet from DNS is removed. If a person attempts again the request fails.



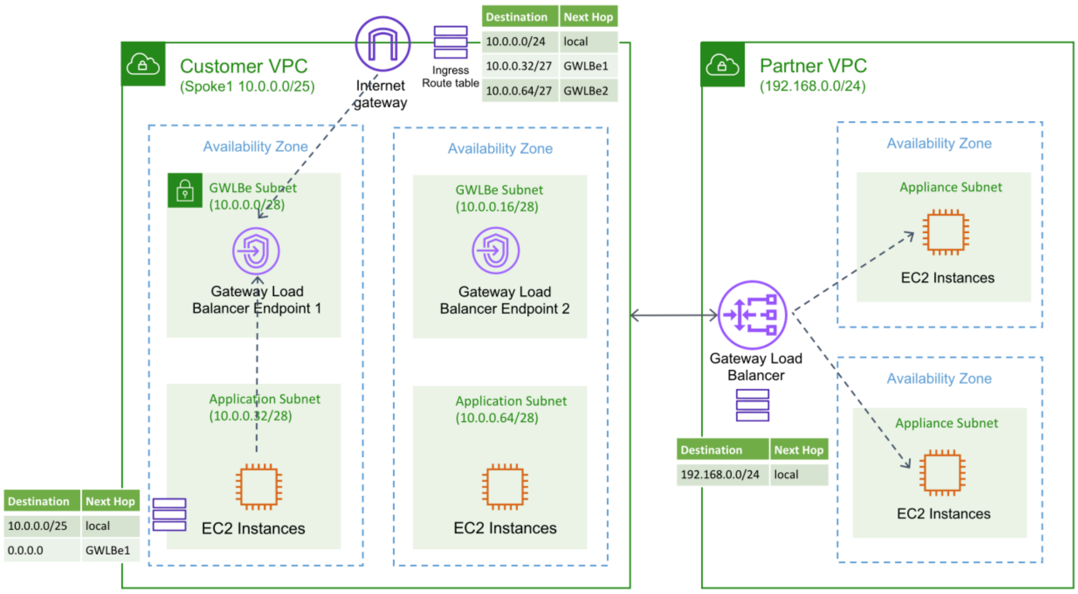
AWS ELB – NLB

Benefits of Network Load Balancers-

- NLB Provides the Support for static IP addresses for the load balancer.
- Provides support for registering targets by IP address which includes target outside the VPC for the Load Balancer.
- Provides support for monitoring the health of each service independently.

Gateway Load Balancer

- It makes it easy to deploy, scale, and manage your third-party virtual appliances.
- Provide you one gateway for distributing traffic across multiple virtual appliances, while scaling them up, or down, based on demand.
- It eliminates potential points of failure in your network and increases availability.
- Users can find, test, and buy virtual appliances from third-party vendors directly in AWS Marketplace.
- This integrated experience streamlines the deployment process, so users can see value from your virtual appliances more quickly—whether you want to work with the same vendors you do today, or trying something new.

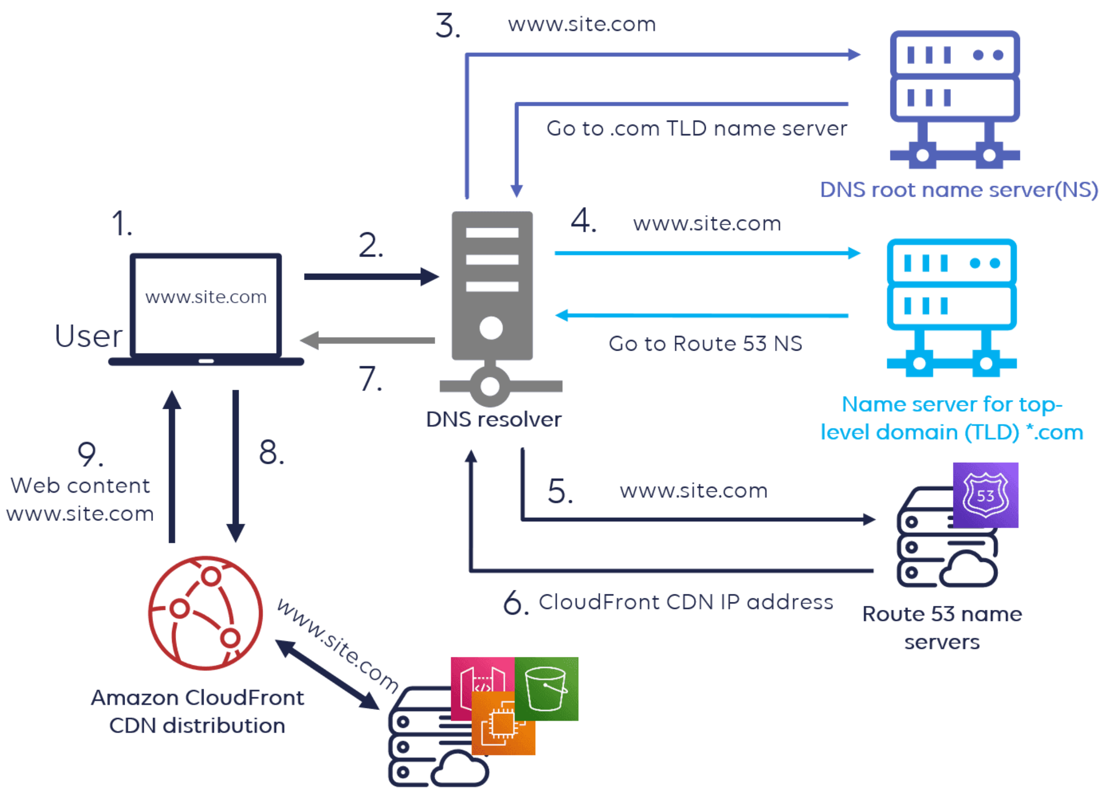


Limitations:

Regional Limit	
LB Per Region	20
Load Balancer Components Limit	
Gateway Load Balancers per VPC	**10**
Target groups with GENEVE protocol	**100**
Targets per Availability Zone per target group with GENEVE protocol	**300**

Amazon Route 53

Route 53 is a web service that is a highly available and scalable Domain Name System (DNS.)


3. www.site.com
Go to .com TLD name server
DNS root name server(NS)
1.
2.
User
www.site.com
4. www.site.com
Go to Route 53 NS
Name server for top-level domain (TLD) *.com
7.
DNS resolver
9. Web content www.site.com
8.
5. www.site.com
53
6. CloudFront CDN IP address
Route 53 name servers
www.site.com
Amazon CloudFront CDN distribution

How does Route 53 work?

1. A user opens a web browser and sends a request for www.site.com.
2. The request from www.site.com is routed to a DNS resolver, which is usually managed by the Internet Service Provider (ISP).
3. The ISP DNS resolver forwards the request from www.site.com to a DNS root name server.
4. The DNS resolver forwards the request from www.site.com again, this time to one of the top-level domain (TLD) name servers of .com domains. The .com domain name server responds with the names of the four Route 53 name servers associated with the example.com domain. The DNS resolver caches the four Route 53 name servers for future use.
5. The DNS resolver chooses a Route 53 name server and forwards the request from www.site.com to that Route 53 name server.
6. The Route 53 name server looks for the record www.site.com in the hosted zone site.com, gets its value, such as the alias of Amazon CloudFront distribution in the case of simple routing.
7. The DNS resolver finally has the right route (CloudFront IP) the user needs and returns the value for the user's web browser.
8. The web browser sends a request from www.site.com to the IP address of the CloudFront distribution.
9. The example CloudFront distribution returns the web page from cache or origin server for www.site.com to the web browser.

Routing Policies

1. Simple routing policy: Use for a single resource that performs a given function for your domain, for example, an Amazon EC2 instance that serves content for the example.com website.

2. Weighted: This allows you to assign weights to resource record sets. For instance, you can specify 25 for one resource and 75 for another, meaning that 25% of requests will go to the first resource and 75% will be routed to the second.

3. LBR (Latency based routing): Use when you have resources in multiple AWS Regions and you want to route end users to the AWS region that provides the lowest latency.

4. Failover: Use when you want to configure active-passive failover. More info in our blog post: Amazon Route 53: Health Checks and DNS Failover

5. Geolocation: This lets you balance the load on your resources by directing requests to specific endpoints based on the geographic location from which the request originates.

6. IP-based: With IP-based routing, you can create a series of Classless Inter-Domain Routing (CIDR) blocks that represent the client IP network range and associate these CIDR blocks with locations.

Technical Key words WRT DNS:

- Domain Name System (DNS)
- DNS Recursor
- DNS query
- Root Nameserver
- Time to Live
- DNS Resolver
- Hosted Zone
- DNS Records

AWS Secrets Manager

- It is a service provided by AWS to store secrets i.e. **passwords, credentials, third-party keys**, or any such confidential information.
- Secrets Manager allows you to **store** and **manage** access to these credentials.
- It allows you to easily **change or rotate** your credentials, thereby avoiding any code or config changes.
- Leverage Secrets Manager to store your credentials **instead of hard-coding** them in your code or config files.
- It allows you to replace hardcoded credentials in your code with an **API call** Secrets Manager to **retrieve** the secrets **programmatically.**
- It **encrypts** the protected text of a secret by using **AWS Key Management System**

Features

- **Rotate Secrets Safely:** Without worrying about updating or deploying the code, you can easily **rotate secrets** using Secrets Manager in order to ensure that your security and compliance requirements are met.
- **Manage Access with fine-grained Policies:** Using certain Identity and Access Management **(IAM)** policies, access to the **secrets can be managed**. For example, a policy can be created that allows developers to get the secrets when they are being used for development purposes.
- **Secure and audit secrets centrally:** By encrypting the secrets with encryption keys you can **secure** your secrets as well. This can be easily achieved using the Amazon Key Management Service **(KMS)** used to encrypt data.
- **Pay as you go:** Secrets Manager is a pay-as-you-go model. You will only be charged for the number of **secrets managed** by the Secrets Manager and the number of Secrets Manager **API calls** made.
- **Retrieve Secrets programmatically:** With Secrets Manager, you can programmatically retrieve encrypted secret values at **runtime**.

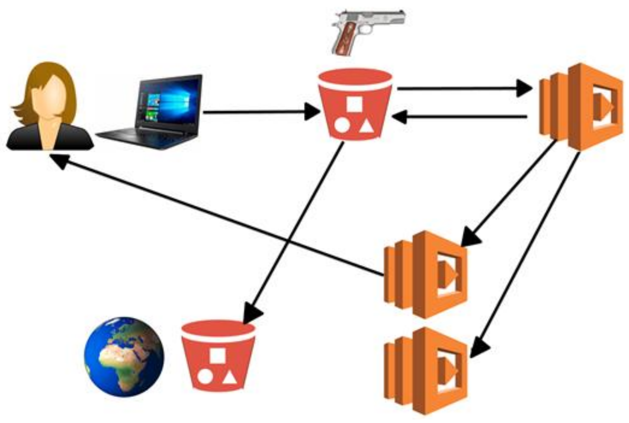
AWS LAMBDA

- Lambda is used to encapsulate Data centres, Hardware, Assembly code/Protocols, high-level languages, operating systems, AWS APIs.
- Lambda is a compute service where you can upload your code and create the Lambda function.
- Lambda takes care of provisioning and managing the servers used to run the code.
- While using Lambda, you don't have to worry about scaling, patching, operating systems, etc.

Lambda can be used in the following ways:

- It can be used as an event-driven compute service where AWS Lambda runs your code in response to events. These events could be changes to data in an Amazon S3 bucket or an Amazon DynamoDB table.
- It can be used as a compute service to run your code in response to HTTP requests using Amazon API calls made using AWS SDKs.

How does Lambda work



- User uploads an image to S3.
- S3 triggers an event, and this event is a Lambda function.
- Lambda function takes this image, and then encode the image. When an image is encoded, it gets stored in S3.
- The Lambda function might trigger other Lambda event which is returning image location back to the user.

- The Lambda might trigger another Lambda event that takes the image from the S3 bucket and stores it in another S3 bucket located anywhere in the world.

You can run Back-End code with AWS Lambda.

Here are some popular Back-End languages:

- Node.js -
- Python -
- Java -
- Kotlin -
- C# -

AWS Lambda Concepts

Function:

A function is a program or a script which runs in AWS Lambda. Lambda passes invocation events into your function, which processes an event and returns its response.

Runtimes:

Runtime allows functions in various languages which runs on the same base execution environment. This helps you to configure your function in runtime. It also matches your selected programming language.

Event source:

An event source is an AWS service, such as Amazon SNS, or a custom service. This triggers function helps you to executes its logic.

Lambda Layers:

Lambda layers are an important distribution mechanism for libraries, custom runtimes, and other important function dependencies. This AWS component also helps you to manage your development function code separately from the unchanging code and resources that it uses.

Log streams:

Log stream allows you to annotate your function code with custom logging statements which helps you to analyse the execution flow and performance of your AWS Lambda functions.

Amazon RDS

Amazon Relational Database Service (RDS) is a managed SQL database service provided by Amazon Web Services (AWS). Amazon RDS supports an array of database engines to store and organize data. It also helps in relational database management tasks like data migration, backup, recovery and patching.

Amazon RDS facilitates the deployment and maintenance of relational databases in the cloud. Cloud administrators use Amazon RDS to set up, operate, manage, and scale relational instances of cloud databases. Amazon RDS itself is not a database; It is a service used to manage relational databases.

How does Amazon RDS work?

Databases store large amounts of data that applications can draw upon to help them perform various tasks. A relational database uses tables to store data and is called relational because it organizes data points with defined relationships.

Administrators control Amazon RDS with the AWS Management Console, Amazon RDS API calls, or the AWS command-line interface. They use these interfaces to deploy database instances to which users can apply specific settings.

Amazon provides several instance types with different resources, such as CPU, memory, storage options, and networking capability. Each type comes in a variety of sizes to suit the needs of different workloads.

RDS users can use AWS Identity and Access Management to define and set permissions to access RDS databases.

mazon RDS database engines

An AWS customer can spin up to six types of database engines within Amazon RDS:

- **Amazon Aurora** is a proprietary AWS relational database engine. Amazon Aurora is compatible with MySQL and **PostgreSQL**.
- **RDS for MariaDB**is compatible with Maria DB, an open-source relational database management system (**RDBMS**) that's an offshoot of MySQL.
- **RDS for MySQL** is compatible with the MySQL open-source RDBMS.
- **RDS for Oracle Database**is compatible with several editions of Oracle Database, including bring-your-own-license and license-included versions.
- **RDS for PostgreSQL** is compatible with PostgreSQL open-source object-RDBMS.
- **RDS for SQL Server**is compatible with Microsoft SQL Server, an RDBMS.

EXTRAS

What is AWS CloudFront?

AWS CloudFront is a globally-distributed network offered by Amazon Web Services, which securely transfers content such as software, SDKs, videos, etc., to the clients, with high transfer speed.

Logo - CloudFront

Benefits of AWS CloudFront

- It will cache your content in edge locations and decrease the workload, thus resulting in high availability of applications.
- It is simple to use and ensures productivity enhancement.
- It provides high security with the 'Content Privacy' feature.
- It facilitates GEO targeting service for content delivery to specific end-users.
- It uses HTTP or HTTPS protocols for quick delivery of content.
- It is less expensive, as it only charges for the data transfer.

What is AWS Snowball?

AWS Snowball is a data transfer device. You can transfer your offline data from S3 buckets using AWS snowball. With AWS Snowball (Snowball), you can transfer hundreds of terabytes or petabytes of data between your on-premises data centers and Amazon Simple Storage Service (Amazon S3). It mainly Uses a secure storage device for physical transportation.

What is GuardDuty AWS?

Amazon GuardDuty is a threat detection service that continuously monitors for malicious activity and unauthorized behavior to protect your AWS accounts, Amazon Elastic Compute Cloud (EC2) workloads, container applications, Amazon Aurora databases (Preview), and data stored in Amazon Simple Storage Service (S3).

AWS Shield work?

A DDoS attack is made in an attempt to make an online service such as a website or web service unavailable by overwhelming it with malicious traffic. This is where AWS Shield comes in.

Amazon Route 53, Elastic Load Balancer and Amazon CloudFront DDoS protection is automatically provided by AWS Shield Standard, at no additional charge.