Exam guide

Thursday, June 3, 2021 12:25 PM

aws training and certification

AWS Certified Developer – Associate (DVA-C01) Exam Guide

Introduction

This AWS Certified Developer - Associate (DVA-C01) examination is intended for individuals who perform a developer role.

It validates an examinee's ability to:

- Demonstrate an understanding of core AWS services, uses, and basic AWS architecture best practices.
- Demonstrate proficiency in developing, deploying, and debugging cloud-based applications using AWS.

Recommended AWS Knowledge

- 1 or more years of hands-on experience developing and maintaining an AWS based application
- In-depth knowledge of at least one high-level programming language
- Understanding of core AWS services, uses, and basic AWS architecture best practices
- Proficiency in developing, deploying, and debugging cloud-based applications using AWS
- Ability to use the AWS service APIs, AWS CLI, and SDKs to write applications
- Ability to identify key features of AWS services
- Understanding of the AWS shared responsibility model
- Understanding of application lifecycle management
- Ability to use a CI/CD pipeline to deploy applications on AWS
- Ability to use or interact with AWS services
- Ability to apply a basic understanding of cloud-native applications to write code
- Ability to write code using AWS security best practices (e.g., not using secret and access keys in the code, instead using IAM roles)
- Ability to author, maintain, and debug code modules on AWS
- Proficiency writing code for serverless applications
- Understanding of the use of containers in the development process

Exam Content

Response Types

There are two types of questions on the examination:

- **Multiple choice:** Has one correct response and three incorrect responses (distractors).
- **Multiple response:** Has two or more correct responses out of five or more options.

Select one or more responses that best complete the statement or answer the question. Distractors, or incorrect answers, are response options that an examinee with incomplete knowledge or skill would likely choose. However, they are generally plausible responses that fit in the content area defined by the test objective.

Unanswered questions are scored as incorrect; there is no penalty for guessing.

Unscored Content
Your examination may include non-scored questions that are placed on the test to gather statistical information. These questions are not identified on the form, and do not affect your score.

Exam Results
The AWS Certified Developer - Associate (DVA-C01) examination is a pass or fail exam. The examination is scored against a minimum standard established by AWS professionals guided by certification industry best practices and guidelines.

Your results for the examination are reported as a score from 100–1,000, with a minimum passing score of 720. Your score shows how you performed on the examination as a whole and whether or not you passed. Scaled scoring models are used to equate scores across multiple exam forms that may have slightly different difficulty levels.

Your score report contains a table of classifications of your performance at each section level. This information is designed to provide general feedback concerning your examination performance. The examination uses a compensatory scoring model, which means that you do not need to "pass" the individual sections, only the overall examination. Each section of the examination has a specific weighting, so some sections have more questions than others. The table contains general information, highlighting your strengths and weaknesses. Exercise caution when interpreting section-level feedback.

Content Outline
This exam guide includes weightings, test domains, and objectives only. It is not a comprehensive listing of the content on this examination. The table below lists the main content domains and their weightings.

Domain	% of Examination
Domain 1: Deployment	22%
Domain 2: Security	26%
Domain 3: Development with AWS Services	30%
Domain 4: Refactoring	10%
Domain 5: Monitoring and Troubleshooting	12%
TOTAL	**100%**

Domain 1: Deployment
- 1.1 Deploy written code in AWS using existing CI/CD pipelines, processes, and patterns
- 1.2 Deploy applications using Elastic Beanstalk
- 1.3 Prepare the application deployment package to be deployed to AWS
- 1.4 Deploy serverless applications

Domain 2: Security
- 2.1 Make authenticated calls to AWS services
- 2.2 Implement encryption using AWS services
- 2.3 Implement application authentication, and authorization

Domain 3: Development with AWS Services
- 3.1 Write code for serverless applications
- 3.2 Translate functional requirements into application design
- 3.3 Implement application design into application code
- 3.4 Write code that interacts with AWS services by using APIs, SDKs, and AWS CLI

Domain 4: Refactoring
- 4.1 Optimize application to best use AWS services and features
- 4.2 Migrate existing application code to run on AWS

Domain 5: Monitoring and Troubleshooting

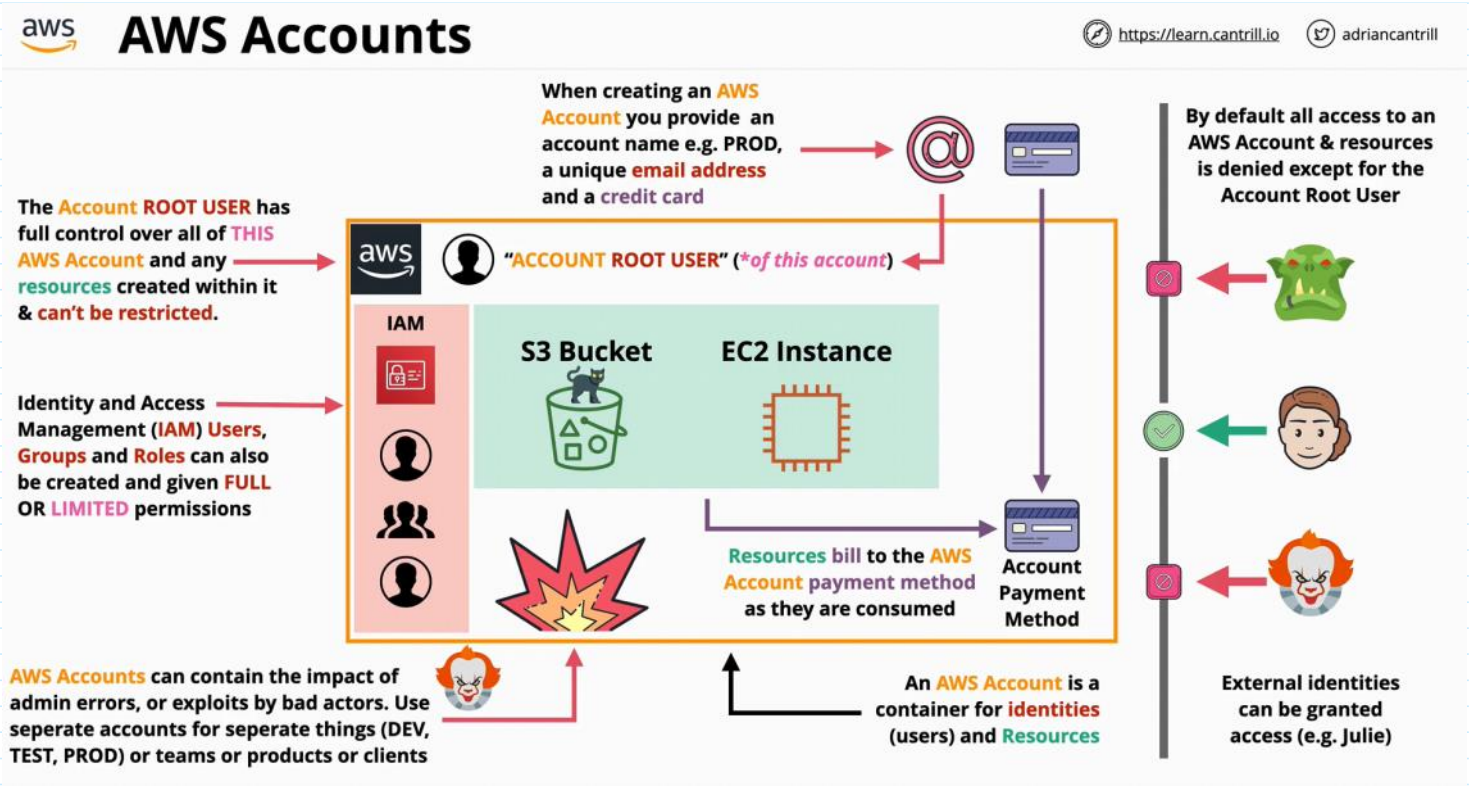
5.1 Write code that can be monitored

5.2 Perform root cause analysis on faults found in testing or production

AWS Accounts

Thursday, June 3, 2021 1:16 PM

ROOT	**has full access in the AWS account**
IAM	**has no access at all initially** **access needs to be granted by the root user**



Management/General	dora.theodora89+AWS1@gmail.com	MDLT
Production	dora.theodora89+AWS1@gmail.com	MDLT

IAM identities start with **no permissions** on an AWS Account, but can be granted permissions (almost) up to those held by the Account Root User.
IAM also is trusted full by the account.

What is cloud computing

Friday, June 4, 2021 3:31 PM

1. Introduction

1.1 Authority

The National Institute of Standards and Technology (NIST) developed this document in furtherance of its statutory responsibilities under the Federal Information Security Management Act (FISMA) of 2002, Public Law 107-347.

NIST is responsible for developing standards and guidelines, including minimum requirements, for providing adequate information security for all agency operations and assets; but such standards and guidelines shall not apply to national security systems. This guideline is consistent with the requirements of the Office of Management and Budget (OMB) Circular A-130, Section 8b(3), "Securing Agency Information Systems," as analyzed in A-130, Appendix IV: Analysis of Key Sections. Supplemental information is provided in A-130, Appendix III.

This guideline has been prepared for use by Federal agencies. It may be used by nongovernmental organizations on a voluntary basis and is not subject to copyright, though attribution is desired.

Nothing in this document should be taken to contradict standards and guidelines made mandatory and binding on Federal agencies by the Secretary of Commerce under statutory authority, nor should these guidelines be interpreted as altering or superseding the existing authorities of the Secretary of Commerce, Director of the OMB, or any other Federal official.

1.2 Purpose and Scope

Cloud computing is an evolving paradigm. The NIST definition characterizes important aspects of cloud computing and is intended to serve as a means for broad comparisons of cloud services and deployment strategies, and to provide a baseline for discussion from what is cloud computing to how to best use cloud computing. The service and deployment models defined form a simple taxonomy that is not intended to prescribe or constrain any particular method of deployment, service delivery, or business operation.

1.3 Audience

The intended audience of this document is system planners, program managers, technologists, and others adopting cloud computing as consumers or providers of cloud services.

1

2. The NIST Definition of Cloud Computing

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model is composed of five essential characteristics, three service models, and four deployment models.

Essential Characteristics:

On-demand self-service. A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service provider.

Broad network access. Capabilities are available over the network and accessed through standard mechanisms that promote use by heterogeneous thin or thick client platforms (e.g., mobile phones, tablets, laptops, and workstations).

Resource pooling. The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand. There is a sense of location independence in that the customer generally has no control or knowledge over the exact location of the provided resources but may be able to specify location at a higher level of abstraction (e.g., country, state, or datacenter). Examples of resources include storage, processing, memory, and network bandwidth.

Rapid elasticity. Capabilities can be elastically provisioned and released, in some cases automatically, to scale rapidly outward and inward commensurate with demand. To the consumer, the capabilities available for provisioning often appear to be unlimited and can be appropriated in any quantity at any time.

Measured service. Cloud systems automatically control and optimize resource use by leveraging a metering capability[1] at some level of abstraction appropriate to the type of service (e.g., storage, processing, bandwidth, and active user accounts). Resource usage can be monitored, controlled, and reported, providing transparency for both the provider and consumer of the utilized service.

Service Models:

Software as a Service (SaaS). The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure[2]. The applications are accessible from various client devices through either a thin client interface, such as a web browser (e.g., web-based email), or a program interface. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings.

Platform as a Service (PaaS). The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages, libraries, services, and tools supported by the provider.[3] The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly configuration settings for the application-hosting environment.

Infrastructure as a Service (IaaS). The capability provided to the consumer is to provision processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, and deployed applications; and possibly limited control of select networking components (e.g., host firewalls).

Deployment Models:

Private cloud. The cloud infrastructure is provisioned for exclusive use by a single organization comprising multiple consumers (e.g., business units). It may be owned, managed, and operated by the organization, a third party, or some combination of them, and it may exist on or off premises.

Community cloud. The cloud infrastructure is provisioned for exclusive use by a specific community of consumers from organizations that have shared concerns (e.g., mission, security requirements, policy, and compliance considerations). It may be owned, managed, and operated by one or more of the organizations in the community, a third party, or some combination of them, and it may exist on or off premises.

Public cloud. The cloud infrastructure is provisioned for open use by the general public. It may be owned, managed, and operated by a business, academic, or government organization, or some combination of them. It exists on the premises of the cloud provider.

Hybrid cloud. The cloud infrastructure is a composition of two or more distinct cloud infrastructures (private, community, or public) that remain unique entities, but are bound together by standardized or proprietary technology that enables data and application portability (e.g., cloud bursting for load balancing between clouds).

[1] Typically this is done on a pay-per-use or charge-per-use basis.

[2] A cloud infrastructure is the collection of hardware and software that enables the five essential characteristics of cloud computing. The cloud infrastructure can be viewed as containing both a physical layer and an abstraction layer. The physical layer consists of the hardware resources that are necessary to support the cloud services being provided, and typically includes server, storage and network components. The abstraction layer consists of the software deployed across the physical layer, which manifests the essential cloud characteristics. Conceptually the abstraction layer sits above the physical layer.

[3] This capability does not necessarily preclude the use of compatible programming languages, libraries, services, and tools from other sources.

On demand self-service: Cloud can provision capabilities as needed **without requiring human interaction.**
- virtual machines, storage databases, networking
- using a web page, terminat to access those
- no delay

Broad network access: Capabilities are available over the **network** and accessed through **standard mechanisms.**
- HTTP, HTTPS, SSH, Remote Desktop, VPN

Resource pooling:
- There is sense **of location independence**... no control or knowledge over the exact location of the resources
- Resources are **pooled** to serve multiple consumers using **a multi-tenant model**
  - pooling: isolates the customer: your data is only visible for you.

Rapid Elasticity:
- Capabilities can be **elastically provisions and** released to **scale** rapidly outward and inward with the demand
- To the consumer, the capabilities available for provisioning often **appear unlimited**.

Measured Service: Resource usage can be **monitored, controlled, reported** and **billed**.
- on demand billing
  - usage is monitored and **then** billed

1 On-Demand Self-Service	Provision and Terminate using a UI/CLI without human interaction.
2 Broad Network Access	Access services over any networks, on any devices, using standard protocols and methods.
3 Resource Pooling	Economies of scale, cheaper service.
4 Rapid Elasticity	Scale UP (OUT) and DOWN (IN) automatically in response to system load
5 Measured Service	Usage is measured. Pay for what you consume.

[3] This capability does not necessarily preclude the use of compatible programming languages, libraries, services, and tools from other sources.

Public & Private & Hybrid & Multi Cloud

Saturday, June 5, 2021 12:49 PM

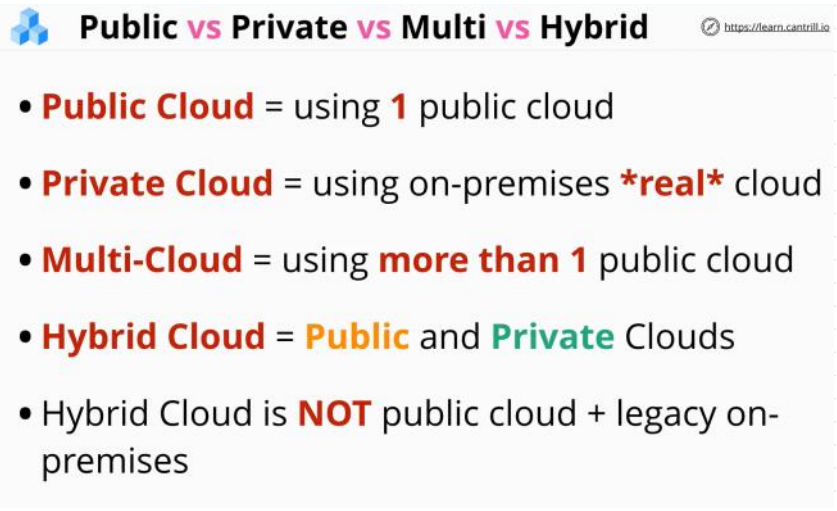
Public Cloud
- classify as a cloud
- available to the general public

Multiple-Cloud
- one app using multiple clouds (AWS, Azure, Google, etc)
- if a vendor fails, part of the system can still work
- multiple clouds can be used a **one** single environment
    - but in this situation they rely on the lowest common feature set
    - loosing what makes each vendor unique

Private Cloud
- meets all 5 characteristics of the cloud

Hybrid Cloud
- using **private** cloud in **conjuncture** with **public** cloud
- private cloud and public cloud working in a single environment
- using same tooling, interfaces and process to interact with both

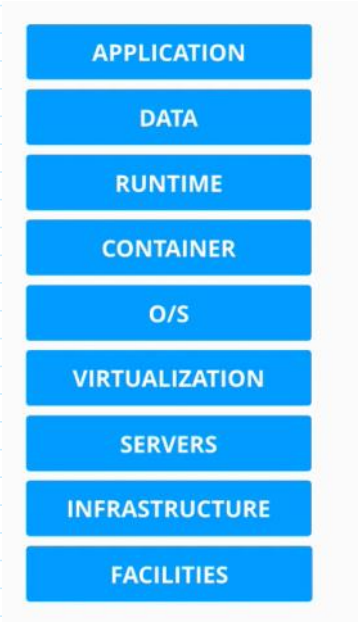
Hybrid environment
- using public cloud with a traditional infrastructure



Cloud service modules

Saturday, June 5, 2021 1:04 PM

Infrastructure stack Some parts are managed by **you**, some are managed by the vendor.



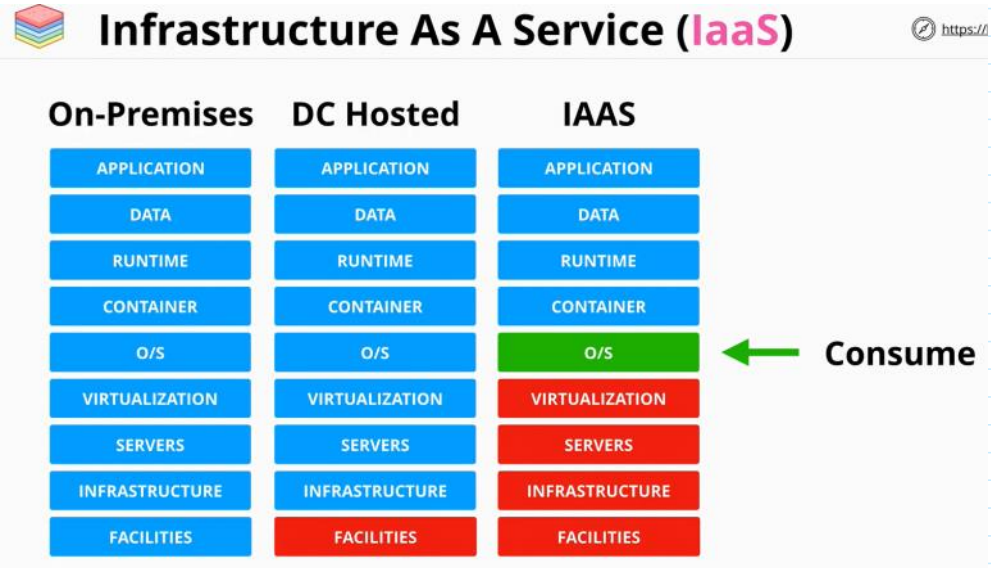
Unit of consumption - what you need to pay that you had consumed.
- part that are you are responsible to manage

In a "on-premises" server (business owns this server)
- the business needs to own/buy each part of the infrastructure
- the business needs to manage all parts of the infrastructure

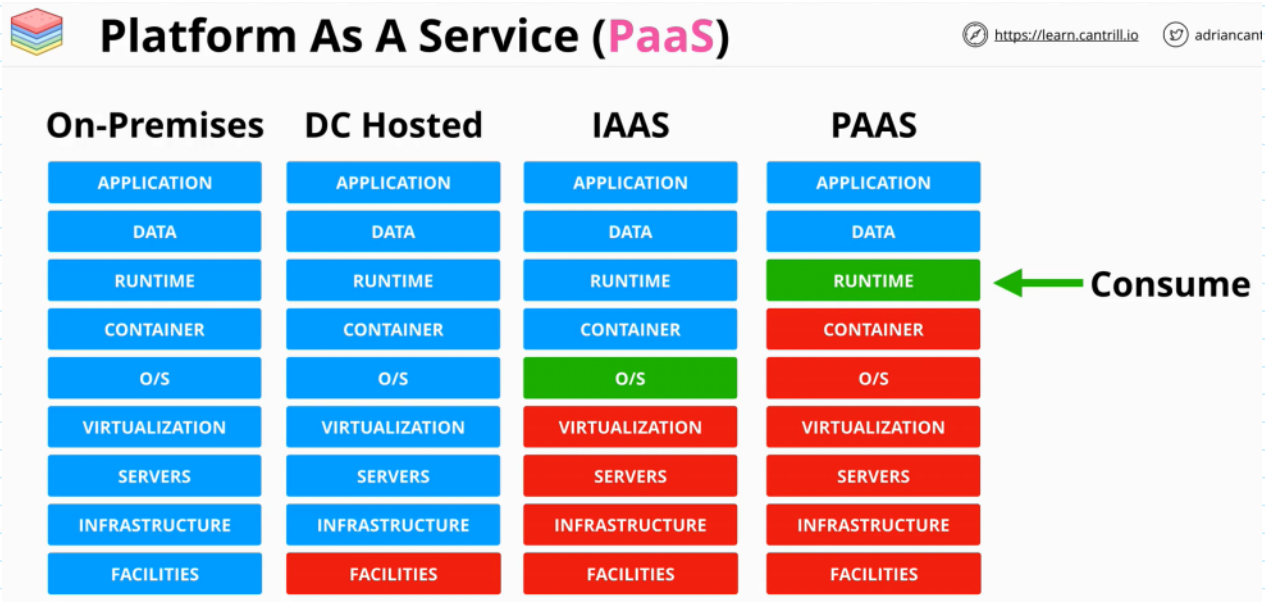
In "DC Hosted"
- the vendor paid an managed the building, the power, air condition and staff

Infrastructure as a service **(IAAS)**

With IAAS you pay per second/minute/hour for the virtual machine - when you use the virtual machine.
- the client ignores the building, maintain the infrastructure, hardware, and staff costs
- EC2

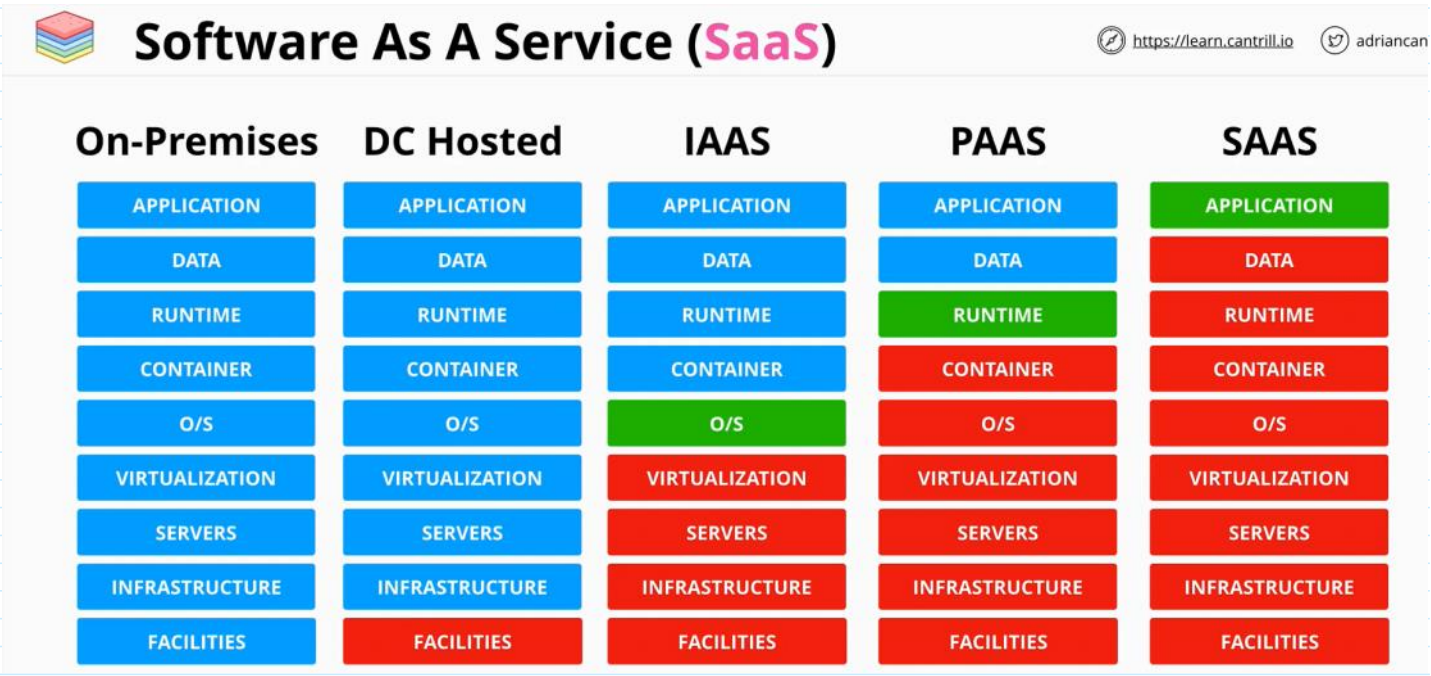


Platform as a service (**PAAS**)
With PAAS is aimed for developers who have an application and only want to run it, without worrying about any of the infrastructure.
- the apps is put in the runtime environment
- the vendor manages the containers, OS, virtualization, servers, etc



Software as a service (**SAAS**)
In this case you consume the application. The application is what you get a service. (Email, Netflix, Dropbox, OneDrive, office, etc)

- there is no much control how the app works

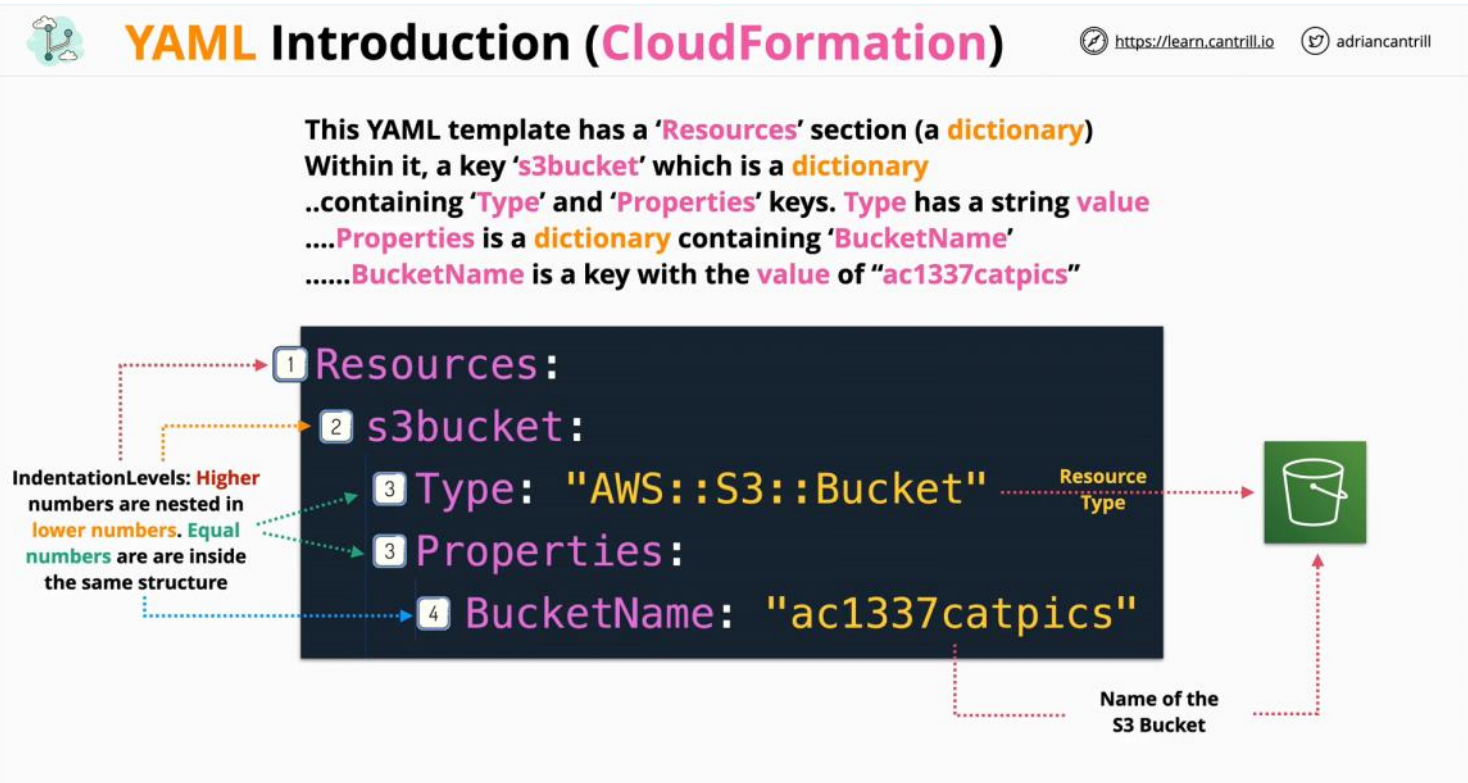


YAML

Saturday, June 5, 2021 1:25 PM

YAML is one of the languages, that Cloud uses as a template.

- human readable
- define data
- configuration
- Key:Value pair
- supports:
  - strings
  - integers
  - floats
  - booleans
  - nulls
  - Lists cats : ["a","b","c"] or cats:
    - "a"
    - "b"
    - "c"
  - dictionary
    - turtle:
      - name: "Pixel"
        color: ["green", "grey"]
      - name: "Sid"
        color: ["black", "green"]
- indentation matters

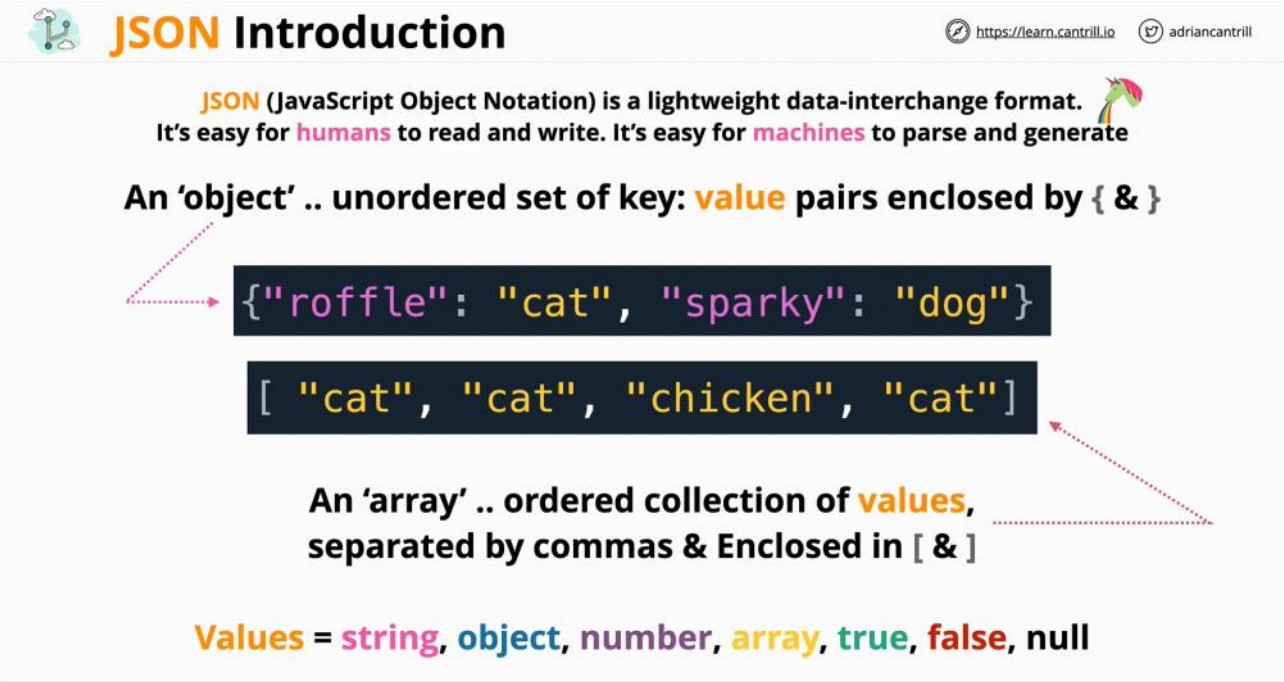


JSON

Saturday, June 5, 2021 1:39 PM

JavaScript Json Notation (JSON) - used broadly
- used for data-interchange format
- easy for humans to read and write
- easy for machines to parse and generate
- does not care about indentation - has brackets

JSON **object** is an unordered set of key:value pairs enclosed by curly brackets {} - dictionary
JSON **array** is an ordered collection of values, separated by commas and enclosed in square brackets [] - list
Values can be:
- string
- object
- integers
- array
- booleans
- null



COLLECTION OF KEY: VALUE PAIRS

{ JSON OBJECT }

```
{
  "cats": ["roffle", "truffles", "penny", "winkie"],
  "colors": ["mixed", "mixed", "grey", "white"],
  "numofeyes": ["2", "2", "2", "1"]
}
```

Each key - "cats", "colors", "numofeyes" has a value
For each example above, this is an array

The top level is a collection of unordered key:value pairs, where the value is a JSON object

Each Nested Object is a collection of key: value pairs where the value can be a scalar, a list or a JSON object

```
{
  "roffle" : {
    "color" : "mixed",
    "numofeyes": "2"
  },
  "truffles" : {
    "color": "mixed",
    "numofeyes": "2"
  },
  "penny" : {
    "color": "grey",
    "numofeyes": "2"
  },
  "winkie": {
    "color" : "white",
    "numofeyes": "1"
  }
}
```

Four Nested JSON Objects

{ JSON OBJECT }

This JSON template has a 'Resources' key, its value is a JSON OBJECT
Within it, a key 's3bucket', its value is a JSON OBJECT
..containing 'Type' and 'Properties' keys. Type has a string value
....Properties is a JSON OBJECT containing 'BucketName'
.....BucketName is a key with the value of "ac1337catpics"

```
{
  "Resources": {
    "s3bucket": {
      "Type": "AWS::S3::Bucket",
      "Properties": {
        "BucketName": "ac1337catpics"
      }
    }
  }
}
```

Resource Type

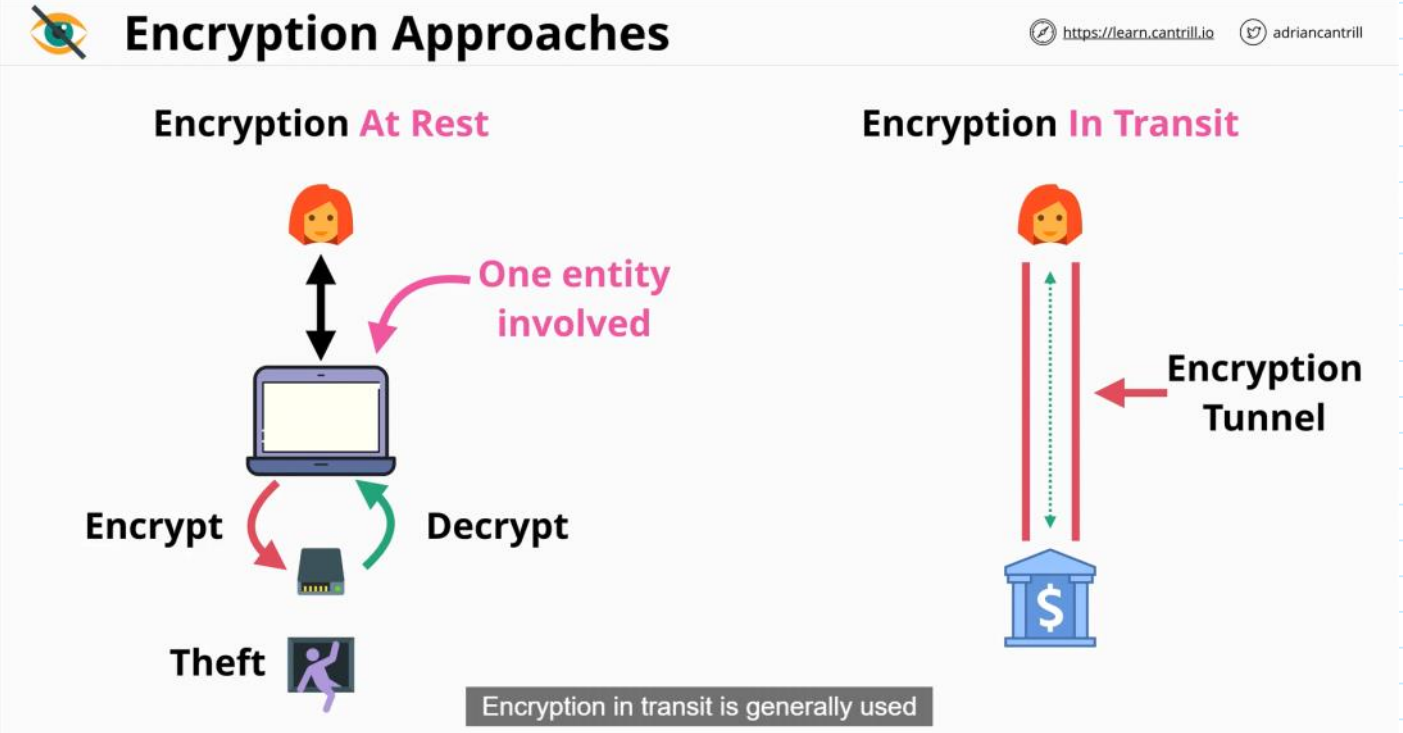
Name of the S3 Bucket

Encryption

Saturday, June 5, 2021 1:51 PM

Encryption Approaches

Encryption at rest	Encryption in transit
• Designed to protect against theft • Encrypts all the data written to the storage • Decrypts the data as is read from the storage • Is usually used if only one entity involved • who know the encryption and decryption key • Used by cloud environments as data is saved in shared hardware	• Aims to protect data as is in transit between two places • The data is encrypted on the "way" and decrypted when it arrives to the destination • Is usually used when multiple entities are involves (send and receiver)



Encryption concepts

Plaintext - text, images or anything that can be read by a person
Algorithm - code/maths which combined with an encryption key can encrypt plaintext ->outputs **cypher text**
Key - is a password
Cypher text - encrypted data
Decryption - taxes cyphertext and the key to output plain text

Keys

Symmetric encryption	Asymmetric encryption
• all entities use the same algorithm	• all entities use the same algorithm

• all entities have the same key
• the key is transferred to all entities in a secure way
  • transporting the key is a problem
  • usually is transferred before the cypher text
	• the entities agree on keys
	  • a public key
	  ○ used to generate cypher text
	  • a private key
	  ○ used to decrypt the cypher text
	• The key do not need to be exchanged in advance
	  • the reader needs to have the private key
	  • the writer needs just the public key
	• **Encryption does not prove identity**
	• Signing - using the private key, where the receiver uses the public key of the sender to check the signature
	  • is used to verify identity
	  • is called **key signing**

Steganography

Sometimes encryption is not enough. Steganography is another layer of protection. Hiding a message within a picture, or something else.

IAM

Friday, June 4, 2021 1:38 PM

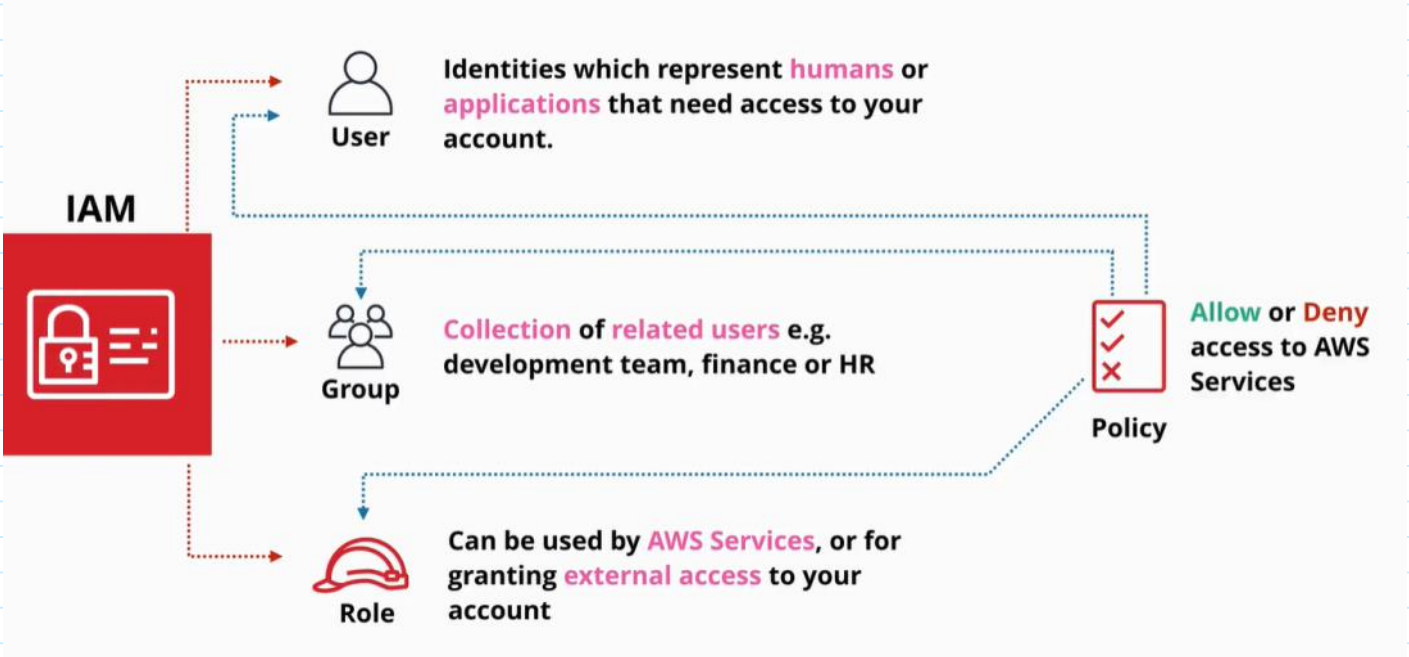
Each account has a **root** user that has **full permissions** on the AWS accounts. - unrestricted access
The **root** user cannot be restricted in anyway.

Other accounts to have access to AWS account can be users, groups or applications. These accounts should be restricted to what they needs to achieve, and are called **least privilege access**.

Every AWS accounts comes with a trusted **IAM** and its own database. This **IAM** belonging to the accounts, is an instance dedicated to **this** account, and is separated from the other AWS accounts.
When a user is added, the account automatically trusts the new user as much as it trusts IAM.

IAM allows to create policies for each users, group, role. These policies state what is allowed or restricted for each identity.

IAM identities start with no permissions on an AWS Account, but can be granted permissions (almost) up to those held by the Account Root User.



IAM has 3 main jobs:

1. CRUD identity (IDP)
2. Authenticate identities
3. Authorize identities. Allows them or denies them actions based on authentication.

IAM is:

- free, but there are some limits.
- is a local service, for the AWS account/infrastructure
- controls what identities can do : allows/denies via policies
- controls only local permissions

There is limit of **5000 IAM user/account**
IAM is a **global service** so this is a global limit
An **IAM user** can be a member of maximum **10 groups**

Create admin (IAM) user

Friday, June 4, 2021 1:57 PM

1. click to IAM console
2. click on user (left)
3. click add user
4. for an admin account, select the:
    a. Attach existing policies
    b. **AdministratorAccess**

This account now has the same privileges as the root user of the account, but is privileges can be restricted by the root user.

IAM Access Keys

Friday, June 4, 2021 2:34 PM

1. user and password are required when using the AWS console
2. access keys are required when using the terminal

An IAM user has 1 user and 1 password, and it cannot have more than that.

- **username** is public
- **password** is private
- **MFA** is private

IAM Access Keys

- **Access Key ID** - like a username
- **Secret Access Key** - as password (can only be seen once and cannot be changed)

Rotated Access Keys - change the access keys (delete and create a new one)

- can be used only by IAM **users** (roles can use one)

Create access keys: (only 2 access keys can be on one user)

1. go to security credentials
2. create access key

Configure the terminal:

aws configure --profile name-of-the-profile	configure a profile
aws s3 ls --profile iamadmin-management	ls s3

Identity Policies

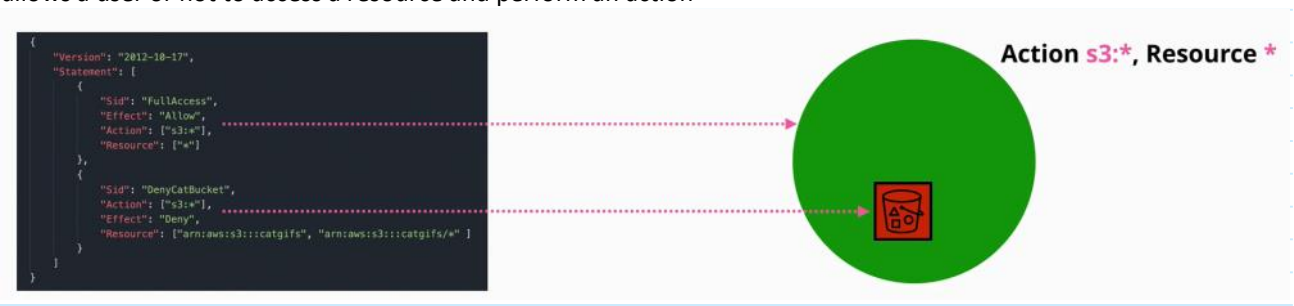
Thursday, June 24, 2021 1:34 PM

IAM identities are users, groups and roles.
An IAM policy is a set of security statements. It grants or denies access to AWS products and features to any identity that uses that policy.

Identity policies also known as **policy documents**, are created using JSON.
A policy document is one or more statements.
Once an entity is authenticated, and AWS knows what policies an identity has.

A statement only applies if the interaction the identity has with the AWS matches the action and the resource.
A **statement has:**

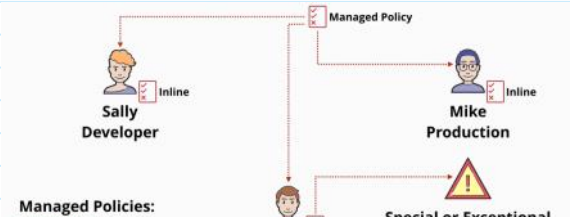
- **SID** (statement ID)
  - optional filed
  - identifies a statement
- **Action**
  - matches one or more actions
  - it can be very specific
  - service:action
  - wildcards can be used to match all actions of a service S3:*
  - there are three options
    - a specific individual action
    - a wild card
    - a list of multiple independent actions
- **Resource**
  - are the same as actions, but they match a **resource**
  - matches individual AWS resources or a list of AWS resources, wild cards are acceptable as well
- **Effect**
  - can be **allowed** or **denied**
  - allows a user or not to access a resource and perform an action



  - in the picture above, effect allowed and denied are granted over a S3 bucket.
    - in this case **overall access** is **allowed**, but **constrained** on the catgifs bucket
  - **Explicit deny wins always**
  - **Explicit allow** wins but not in favor of **explicit deny**
  - **If an effect is not specified, than is implicit denied**
  - **By default a policy is DENIED**
  - **RULE: deny, allow, deny**

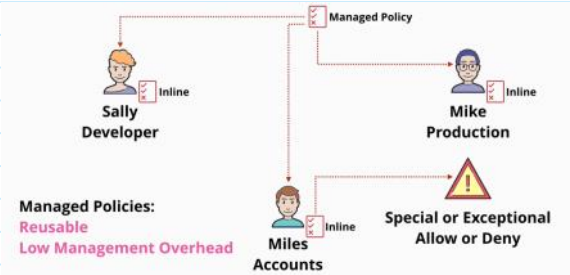
Inline policies - are granted for each entity individually
- used for exceptions - special circumstances

Managed policy - one policy attached to many identities
- AWS managed policies
- customer managed policies - created by the owner



Inline policies - are granted for each entity individually
- used for exceptions - special circumstances

Managed policy - one policy attached to many identities
- AWS managed policies
- customer managed policies - created by the owner



IAM Users and ARNs

Thursday, June 24, 2021 3:55 PM

IAM Users are an identity used for anything requiring long-term AWS access e.g. Humans, Applications or service accounts

These are called **PRINCIPLES**, and for the principles to be able to do anything it needs to authenticate and be authorized.

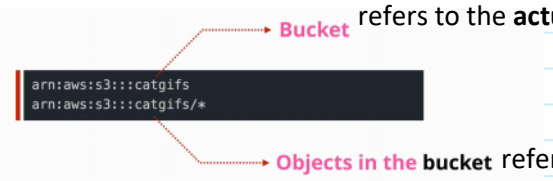
The process is like this:

1. the principle makes a request to IAM to access/interact resources
2. the principle need to authenticate as an IAM used
    a. can be achieved in two ways: **login credentials** or **access keys**
        i. **humans used login credentials**
        ii. **applications use access keys**
3. Now the principle is an authenticated entity and can start interacting with AWS
4. AWS knows which polies applies to the identity **(authorization)**

AMAZON RESOURCE NAME (ARN)

ARN are uniquely identify resources within any AWS accounts

ARNs can always identify single resources in the same account weather they are individual resources in the same account or different accounts.
ARNs are used in IAM policies which are generally attached to identities such as IAM users, and they have a defined format.



refers to the **actual bucket**

refers to **all the objects in the bucket**

ARN structure:

arn:partition:service:region:accountID:resourceType/resourceID

if any are omitted it means it does not refer to one in particular (like regions) - DOES NOT MEAN IT INCLUDES ALL!!!

There is limit of **5000 IAM user/account**
IAM is a **global service** so this is a global limit
An **IAM user** can be a member of maximum **10 groups**

IAM Groups

Thursday, June 24, 2021 5:00 PM

IAM groups are containers for IAM users.

- you cannot login into a group
- the group is used to manage policies easier for users
- facilitates the management of IAM users

An **IAM user** can be part of multiple **IAM groups!**

- it does **not EXIT** by **default** a general **IAM Group** that applies to all IAM users.
  - if a group like this is needed, it needs to be created and managed by the admin
- **groups within groups does not exist**
- **IAM groups** contain users
- **IAM** groups contain permissions attached
- **there is a limit of 300 groups/account** (can be extended by a support ticket)
- can hold identity permissions
- admin groupings of IAM users

Policies can be attached to resources

- these policies can grant access to different IAM users
- allows or denies access to IAM users to identities
- id does this by referencing this identities using an ARN
- a policy on a resource and reference IAM users and IAM roles by using ARN
- groups are not identities and cannot be referenced as principal in a policy
  - resources cannot grant access to IAM groups

IAM Roles

Sunday, June 27, 2021 1:54 PM

A single **principle** uses an IAM users.

IAM roles are also identities

- used by multiple principles (not just one as users)
  - they can be multiple AWS users
  - they can me multiple external applications, users, etc.
- if you cannot identify the number of principles that use an identify then is a candidate for an IAM role
- if you have more than 5000 users (the limit of IAM users) it can also be a candidate for an IAM role
- used on temporary basis
- a role **represents a level of access inside an AWS account**
- **an IAM user is a representation of a user for long term**
- **a role borrows the permissions on a short period of time**

IAM users have permission policies attached to them in inline or managed policies

- these policies define what permission an identity gets inside AWS

IAM roles

- have two types of policies that can be attached to them
  - trust policy
    - which identities can assume the role
    - can refer different identities: IAM users, other roles and AWS services, identities in other AWS accounts
    - if an identity is allowed, AWS generates temporary security credentials (like access keys)
      - they are time limited
      - can be renewed
      - the identities can access the resources specified in the **permission policy**
  - permission policy
    - every time the temporary credentials are used are checked against the permission policy

Roles are real identities, as IAM users, roles can be referenced in resource policies.

- are used in AWS organizations, allowing us to log into one account in the organization and access different account without having to login again
- they are really useful when having a large number of accounts

Temporary credentials, for roles, are generated by a service called Secure Token Service (STS). sts:AssumeRole

When and where to use IAM Roles

YES	NO
Services that need permissions	
Anything that is not an identity	
Very useful in emergency situations - break glass situation - used when absolutely required	
When adding AWS in an existing corporate environment **Roles are often used when you want to reuse the existing identities for use in AWS** - external accounts cannot be used directly - allowing a role to be assumed by an external account (or multiple) - called **ID federation**	
Giving access to users of an app,	

Hardcoding access keys is not advisable as is a security concern, and whenever the keys change, they need to be hardcoded again.
In this scenario, the **IAM Role** is the perfect solution.

to some resources - called web identity federation	

AWS Organizations

Sunday, June 27, 2021 5:11 PM

AWS Organizations is a product that **allows companies to manage multiple AWS accounts** in a cost effective way.

An account is needed to create an organization.

The account that creates the AWS account, becomes the management account for that organization.
The management account is also called master account.
Using the management account

- you can invite other standard AWS accounts into the organization
  - the invited standard AWS accounts need to accept the invitation to the organizations
  - if the standard AWS accounts accept the invitation they become part of the organization
  - these accounts are changed from being standard AWS accounts to **member accounts of that organization**
- an organization has **only one** master accounts, and **zero or more member accounts**
- **the organization can create accounts directly within it**
  - it isn't an invite process
- with organizations, IAM users do not need to exit within each member AWS account
  - IAM roles can be used to allow IAM users to access other AWS accounts

The Organization has a hierarchy (like a reverted tree)

- the **root container** of the organization is at the top of the tree
  - do **not** confuse it with the **root user**
- other containers can be created - known as **organizational units (OUs)**

Organizational billing

- the billing method for each member account are removed
- the member accounts pass their billing through the management account of the organization also called **payer account**
- the single monthly bill generated for AWS organization, covers all the accounts of the organization

The organization has a service called Service Control Policies (SCPs)

- this service can restrict what each member account can do

Service Control Policies (SCP)

Friday, July 2, 2021 12:10 PM

SCP is a feature of AWS organizations, which can be used to restrict AWS accounts.

SCP is a JSON document (policy document) and can be attached to the
- organization as a whole
    - by attaching them to the root container
- one or more organizational units (OU)
- to individual AWS accounts

Service Control Policies inherit down the organizational tree.
- if the are attached to the organization it affects all accounts within the organization
- if they are attached to an organizational unit, they impact all the account inside the organizational unit
    - if there are nested OUs , it will affect the OUs inside the organization as well

The management account of the organization is special. If this account has policies attached directly, or through an organization unit, **the management account is not affected by any SCP!**

Service Control Policies are account permission boundaries and they limit what an account can do.

As mentioned before, we cannot restrict the account root user of an AWS account, but within an organization with the SCP we can restrict what an AWS account can do (with all the entities in that account)! Therefore, restricting indirectly the account root user.

SCP do not grant any permissions, they define the limit of what isn't allowed (and what is). You still need to give the identities in that AWS account permissions to AWS resources, **but SCPs will limit the permissions that can be assigned to individual identities.**

SCPs can be used in two ways:
1. Block by default and allow certain services: using an allow list
2. Allow by default and deny certain services: using a deny list (default)

When SCP is enabled in an account, AWS applies a default policy which is called a **full AWS access. This is applied to all OUs within the organization. - nothing is restricted**
- **this maintains the current functionalities in the organization (in the moment of creating the SCP)**

To use the allow list, the **FULL AWS access** list must be removed, and allowed services should be specified independently into a new policy. - **security improved**

SCP do not grant access to anything, they specify WHAT CAN AND CANNOT BE ALLOWED by identity policies within that account

Only what is allowed in the SCP and the identity policies is actually allowed.

CloudWatch

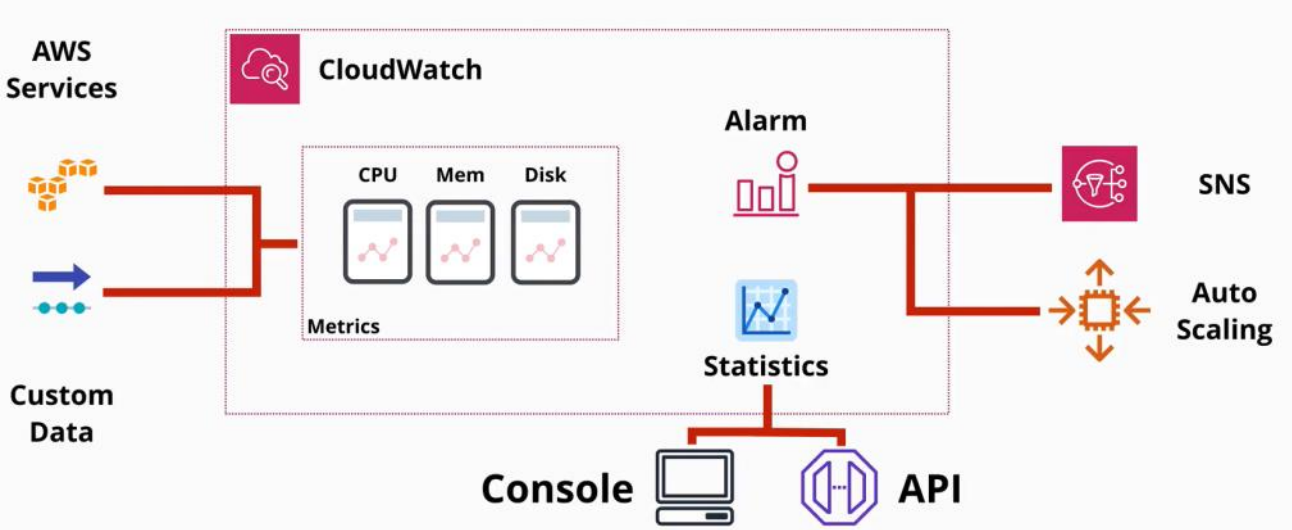
Tuesday, June 22, 2021 1:46 PM

Cloud Watch is a support service used by almost all AWS services - especially for operational services and monitoring.

Cloud watch performs three main jobs:
It collects and manages operational data

a. data that is generated by an environment

1. a collection of metrics, monitoring of metrics and actions based on metrics
    a. data relating to AWS products
        i. CPU utilization of EC2
        ii. numbers of visitors
2. Cloud Watch logs
    a. logging data
3. Cloud Watch events
    a. EC2 terminated, started or stop
    b. schedule an event



Namespace

- is a container for monitoring data, separating things in different areas
- all AWS data go into a namespace is **AWS/service** (e.g. AWS/EC2)

Metric is a collection of related datapoints in a time ordered structure.

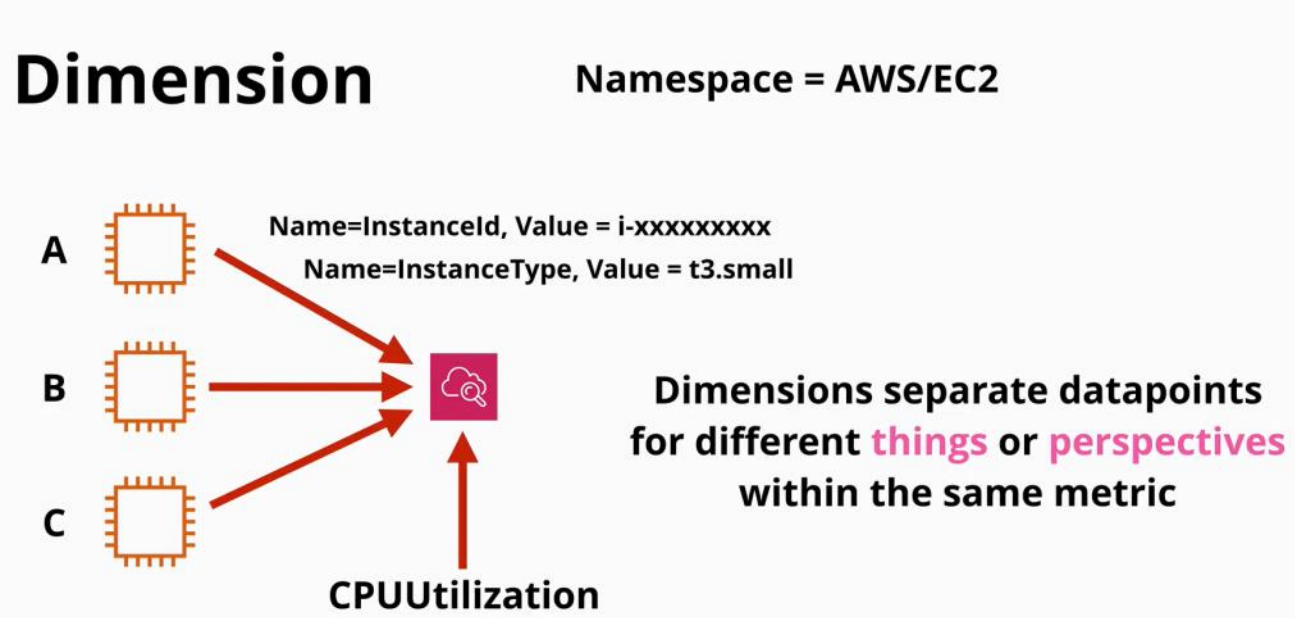
- CPU utilization
- network in an out
- disk utilization
- will start monitoring when the service is started, and it will stop when the service is disabled

Datapoints

- consists of two things:
    - time when measurement was conducted
    - and a value that represents the monitoring

Dimension

- are name value pairs which allow CloudWatch to separate things



Alarms

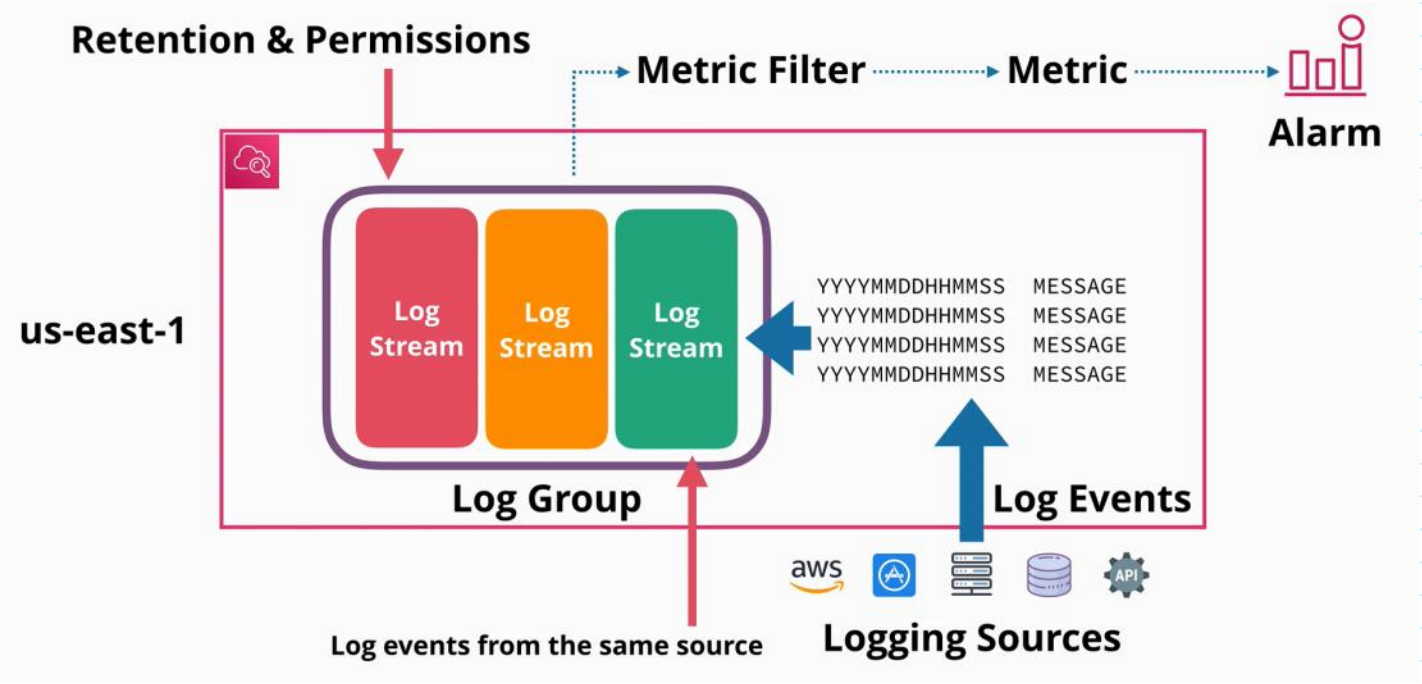
- are created and linked to a specific metric
- based on the alarm's configuration, an action can be performed if the condition for the alarm is met

CloudWatch Logs

Friday, July 2, 2021 1:22 PM

CloudWatch logs is a public service. YH endpoint in which applications connect to is hosted in the AWS public zone.

- which means you can used the product within the AWS VPCs or from on-premises environments and even other cloud platforms as long as you have internet and permissions
- allows you to **store, monitor** and **access** logging data
    - logging data is a piece of information data, and a timestamp
- has some built in integrations with AWS services such as EC2, VPC flow logs, lambda, CloudTrail, R93 and more
- each service that interacts with CloudWatch can store data directly inside the product
- AWS services can log into CloudWatch directly
- unified CloudWatch agent can be used to integrate outside services
- can generate metrics based on logs using the **metric filter**
- **is a regional service**
- **log streams** is a collection of log events from the same source
- **log groups** are containers for multiple log streams for the same type of logging
    - in log groups settings are located as well
        - this settings apply to all log streams inside that group
    - metric filters are defined here as well



CloudTrail Essentials

Friday, July 2, 2021 3:59 PM

Almost everything that can be done with an AWS account can be logged with CloudTrail product.

CloudTrail:

- logs APIs calls and account activities
  - each is called a **CloudTrail event**
    - is an activity in an AWS account
    - an activity is an action taken by an user, role or a service
- by default logs the last 90 days activity in the CloudTrail Event history
  - is enabled by default
  - is free
  - can be customized
- can be two types
  - **management events** (by default the CloudTrail only logs this)
    - provides information about operations performed on resources (control)
  - **data events** (not enabled by default)
    - contains information about resource operations performed on or in a resource
- stored the events in a definable S3 bucket (only if you configure a trail, otherwise you don't get an S3 bucket)
  - stored in JSON format
- a **CloudTrail trail** is a unit of **configuration** within the CloudTrail product
  - how configuration is done
  - **a trail logs events for the AWS region that it's created in**
  - a trail can be configured for one region, or for all regions.
- can be integrated with CloudWatch logs
  - in addition of adding into the S3 buckets dedicated for events, it adds it to the CloudWatch logs
- it is **not real-time** - has a delay (15 minutes)
- an organizational CloudTrail is available
  - can be created from the management account of the organization
  - can store all the information from **all the accounts** within the organization

AWS services are largely split up into regional services and global services. So when this different type of services log to CloudTrail they either log in the region the event is generated in, or if **they are** global (IAM, STS, CloudFront) they log into **us-east-1**. A trail needs to be enabled to capture that data!

S3 Buckets

Wednesday, June 16, 2021 6:49 PM

Simple storage service - S3

S3 is:
- global storage platform - regional resilient
- is a public service
- unlimited data
- multi-user access
- any type of data
- is economical
- **default** storage on AWS
    - **objects** - any data
    - **buckets -** containers for objects

Objects are files, made of:
- object **key** (file name)
- object **value** (the content of the file being stored)
- size range from 0MB to 5TB/object

Buckets
- **all buckets by default are private**
- data inside the bucket has a primary region (than can be changed)
- data never leaves the region (unless configured)
- **the name of the bucket needs to be unique across all regions in AWS**
- can hold unlimited number of objects
- has a flat structure - all files are saved on the root file (no folders in folders)
    - the user can save folders in folders, but in reality is a **flat structure**
    - folders **are called prefixes**
- **bucket names needs to be**
    - all lower case
    - no underscores
    - 3 - 63 characters - characters or digits
- there is a limit of number of buckets for an AWS account, and **it is a soft 100, and a hard limit 1000**
- is good for large scale, data storage, distribution or upload

Simple Storage S3 - is an AWS Public Service, is an object storage system and buckets can store an unlimited amount of data

S3 Security

Saturday, July 3, 2021 11:35 AM

S3 is private by default

- by default the account root user has access to the bucket only
- any other permissions need to be explicitly granted through:
  - **resource policy** - **who can access a resource**
    - can grant access to identities from other accounts - (other AWS accounts)
    - can allow or deny anonymous principals - (sharing a bucket with the world)
    -
```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"PublicRead",
      "Effect":"Allow",
      "Principal": "*",
      "Action":["s3:GetObject"],
      "Resource":["arn:aws:s3:::secretcatproject/*"]
    }
  ]
}
```
    - **principal filed defines what principals have access to a resource**
      - **if is there, is a resource policy**
  - **identity policy** - **what resources the identity can access**
    - access can be granted to identities in your account only

Bucket policies

Can be used:

- to control who can access objects (even targeting IP addresses)

```
{
    "Version": "2012-10-17",
    "Id": "BlockUnLeet",
    "Statement": [
      {
        "Sid": "IPAllow",
        "Effect": "Deny",
        "Principal": "*",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::secretcatproject/*",
        "Condition": {
           "NotIpAddress": {"aws:SourceIp": "1.3.3.7/32"}
        }
      }
    ]
}
```

  - this permission allows only identities from 1.3.3.7, otherwise the statement applies!
- is a type of **resource policy** is applied to a bucket
- a bucket can have **only one policy** but it can **have multiple statements**
- when an entity accesses a bucket, both the identity and the bucket policy apply

Access Control Lists (ACLs)

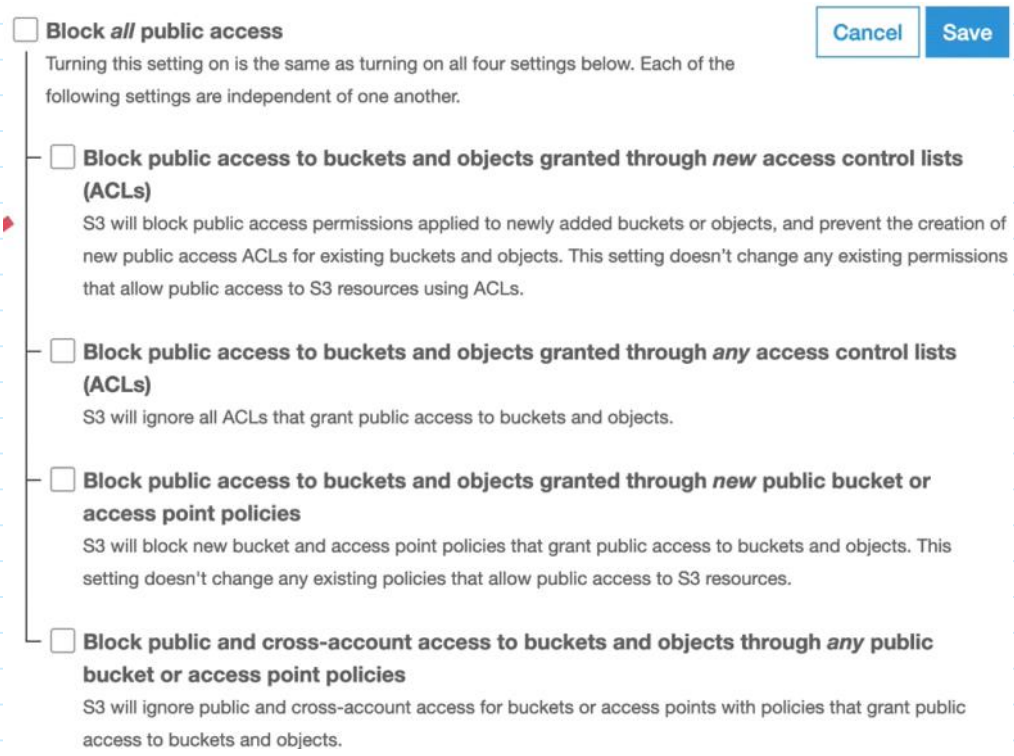
(not really used anymore)

- is a way to apply security to objects and buckets
- is a sub resource
- they are inflexible and allow very simple permissions
- you can use one ACL for multiple objects
    - one ACL per object and one ACL per bucket

Permission	Bucket	Object
READ	Allows grantee to list the objects in the bucket	Allows grantee to read the object data and its metadata
WRITE	Allows grantee to create, overwrite, and delete any object in the bucket	Not applicable
READ_ACP	Allows grantee to read the bucket ACL	Allows grantee to read the object ACL
WRITE_ACP	Allows grantee to write the ACL for the applicable bucket	Allows grantee to write the ACL for the applicable object
FULL_CONTROL	Allows grantee the READ, WRITE, READ_ACP, and WRITE_ACP permissions on the bucket	Allows grantee the READ, READ_ACP, and WRITE_ACP permissions on the object

Block Public Access

- Adds another level of security, which apply no matter what the **bucket policy says**.
    - the Apply only for public access
    - the do not apply to AWS identities



1. **if you are granting and denying permissions on lots of resources across AWS accounts, then** identity policies **should be used, as not all resources have** resource policies.

2. **if you have a preference to manage resources all in one place that single place needs to be** IAM**.**
3. **if you are working with permissions within the** same account**, then IAM is the place to create policies, as IAM** can manage policies only within that account identities.
4. **If you want to grant permission to a single resource to multiple users or everybody in one account then is much more useful to use** resource policies that give level permissions
5. **if you want to directly allow** anonymous identities access**, then again** resource polies are the best fit, as identity policies cannot control permissions for principles outside the account
6. **never user ACL unless you absolutely have to!**

S3 Static Website Hosting

Saturday, July 3, 2021 12:59 PM

S3 Static Website Hosting allows access to buckets and object via HTTP.

- when enabling it, you need to set an **index** and an **error** document
  - index home page - default page
  - error page - is accessed when something goes wrong
- needs to be HTML
- if you use a domain than the bucket name needs to match the domain's name
- the default endpoint is influenced by the bucket name and the region is in

Offloading

- if a website needs a database to load static media (images), S3 bucket can be used to store the media and retrieve it when needed it
- this is cheaper than other compute services as it is costumed-designed for the storage of large data at scale

Out of band pages

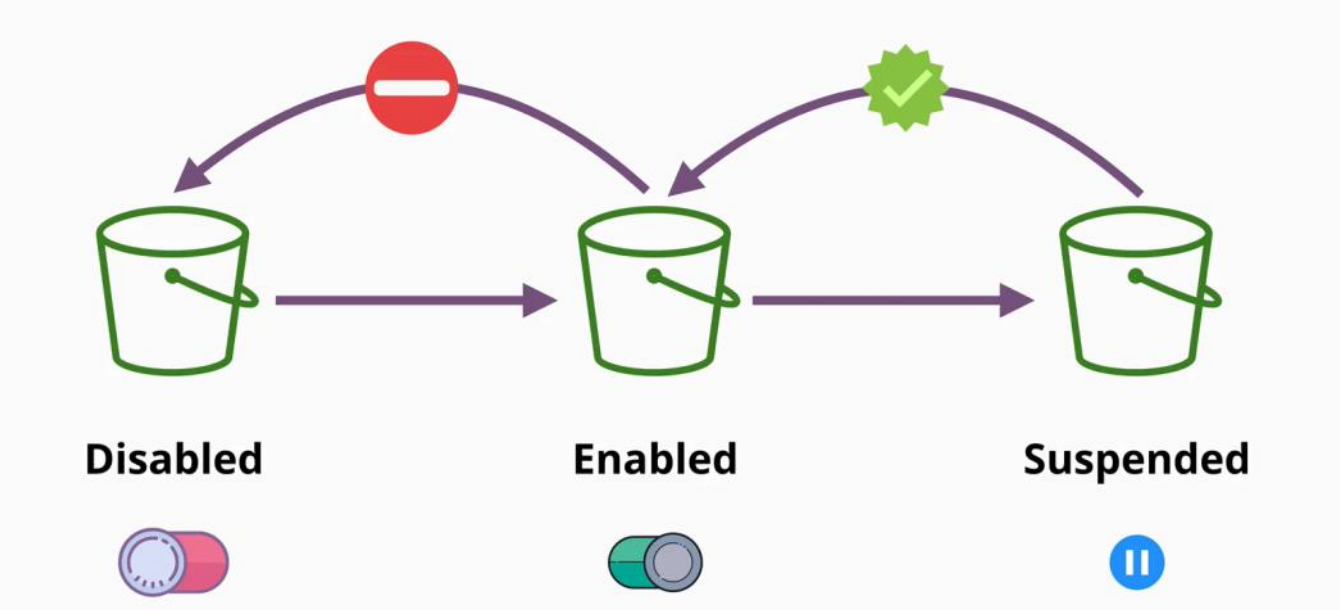
- another services used, in case the main server is offline or has any other interruptions, then we can change our DNS and point customers at a backup static website hosted on S3

Object Versioning and MFA Delete

Saturday, July 3, 2021 3:12 PM

Object Versioning

Object versioning starts up at the object level and is disabled by default. Once the object versioning is **enabled it cannot be enabled again, it can only be suspended and re-enabled again.**



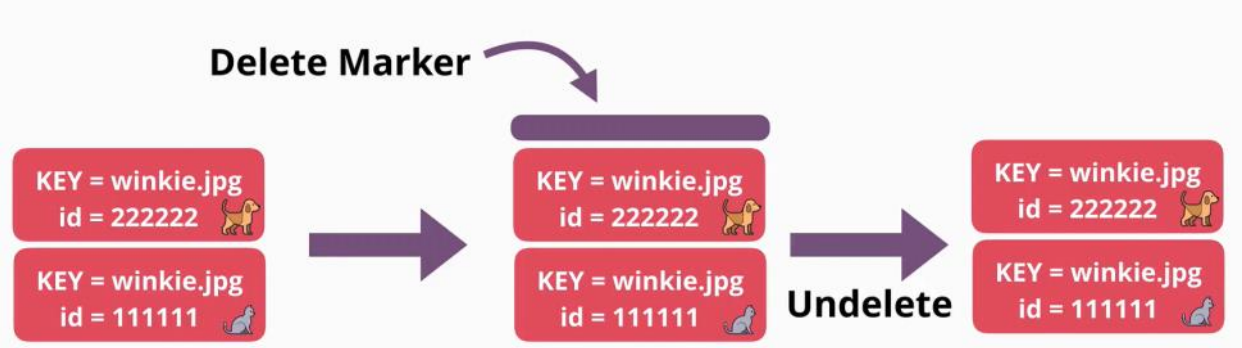
Without versioning enabled on a bucket, each object is identified solely by the object key, its name (which is unique inside the bucket). If an object is modified, the new version of the object replaces the old one.

Versioning allows to store multiple versions of the same object within a bucket. **Any operation that modify objects will generate a new version of that object.**

The latest version of an object is called **the current object** or **latest version.**

When requesting an object from a version bucket, and no version is specified then the current version is returned. Other versions can be requested by **specifying the ID**.

When an object is deleted in a version bucket, the current version of the object is marked as deleted, but the actual object is just hidden. **The delete marker is a special version of an object which hides all previous versions of that object.**
The delete marker can be deleted, which results in restoring the last current version of that object.



To truly delete an object, the version ID needs to be specified. If the current version of the object is deleted, and there are other versions of that object present, the most current version before the current version becomes the current version object.

Space within an bucket **is consumed by all the versions of objects**. (being billed for all). Even if you suspend the version service, the objects and their version withing that bucket are billed.
For zero costs, the bucket should be deleted.

MFA Delete (multi factor auth)

MFA Delete is enabled within the versioning configuration of a bucket.

- when is enabled it means that MFA is required to change bucket versioning state
- MFA is required to delete versions

- Enabled in **versioning configuration**
- MFA is required to change bucket **versioning state**
- MFA is required to **delete versions**
- Serial number (MFA) + Code passed with API CALLS

S3 Performance Optimization

Wednesday, July 7, 2021 12:32 PM

When uploading to S3, and object is loaded as a single blob of data in a single stream

- a file is uploaded and becomes an object via the **S3:PutObject API**. **This happens a single stream**
- **if a stream fails, the whole upload fails**, which needs a full restart
- the speed and reliability of the upload process is limited by the limit of 1 stream of data
- when transferring data between two points, the speed that the data will transfer matches the lower speed from the two points
- any upload with the **put** method, has a limit of maximum 5GB

A solution for this is **multipart upload**

- improves the speed and reliability of uploads to S3
- it does its job by dividing the data into individual parts
    - the minimum size of the original blob of data is 100 MB
    - a blob can be split into maximum of 10,000 parts
        - each part can range from 5MB to 5GB
        - the last part is called a leftover, and can be smaller than 5MB if needed
    - each part can fail and be restarted in isolation
        - the whole data transfer does not need to be restarted if a part fails
    - the transfer rate in this case is the speed of transfer of all parts

Accelerated Transfer

- we do not have control of the path, that the transfer is made from point A to point B
- transfer acceleration uses the network of AWS edge location
- S3 transfer acceleration needs to be enabled, as by default is switched off
    - there are few constrains to enabled it
        - the bucket name needs to not contain any periods
        - needs to be DNS compatible in its naming
- when the transfer acceleration is enabled, the data being uploaded instead of going to the S3 directly it immediately enters the closest best performing AWS edge location
- from here the data is transferred through AWS network, a network that is fully controlled by AWS - direct link
    - the private AWS network is designed to link regions to other regions

s3-accelerate-speedtest.s3-accelerate.amazonaws.com/en/accelerate-speed-comparsion.html

Key Management Service (KMS)

Wednesday, July 7, 2021 1:43 PM

Key Management Service is not part of S3, but is used in most AWS services **that use encryption**. and S3 uses encryption.

KMS is a **regional and public service.** (is a separate product in each region)

- it is public that you need permission to access it
- allows you store, create and manage **cryptographic keys**
- these keys are used to convert plain text to cipher text and vice-versa
- can handle symmetric and asymmetric keys
- can perform cryptographic operations such as encryption and decryption
- **the keys never leave the product** - **provides FIPS 140-2 (Level 2 or 3)**
    - the keys are locked inside KMS service
    - its function is to secure and make sure that the keys never leave the service

Customer Master Key (CMK)

The main thing that KMS manages is **CMS** (Customer master keys)

- are used by KMS for cryptographic operations
- can be used by entities, applications and AWS services
- CMS are logical - a container for the actual physical master keys
- The contain:
    - a key ID - used as an identifier of the key
    - creation date
    - a key policy
    - a description
    - a state of the keys: active or inactive
- can be generated or imported by KMS
- **can be used to encrypt or decrypt data up to 4KB**

The process:

1. After peaking a region a key can be created (a CMK)
2. A CMK logical container is created which contains physical baking key material
    a. This is what KMS creates, stores and manages
    b. the CMK are never stored in a persistent store if they are not encrypted (KMS encrypts the key before storing it on the disk)
3. Request to encrypt some data
    a. this is achieved by making an encryption call, providing a key and the data to be encrypted
4. KMS accept the data and if the entity has permissions to use the CMK
    a. the KMC decrypt the key
    b. uses the decrypted key to encrypt the plain text data to cypher text
5. KMS returns the data to the entity

1. To decrypt the data, an entity needs to request the decrypt
2. KMC does not need to be told which CMK to use to decrypt the data
    a. that information is encoded into the cypher text of the data that needs to be decrypted
3. KMS will decrypt the CMK
4. Will use de decrypted CMS to decrypt the cypher text
5. Will return the data to the entity

During both processes the **CMK does not leave the KMS, the decrypted CMK is not stored on the**

disk and at every step the entity needs permissions to perform this operations

- **EACH OPERATION NEEDS DIFFERENT PERMISSIONS**
  - **Administrative permissions** - delete and create keys
  - **Usage permissions** - encrypt and decrypt

Role separation is where different roles are given different access rights within a product.

Data Encryption Keys (DEKs)

DEKs are another type of key which KMS can generate using a CMK.

- can be used to encrypt data larger than 4 KB
- this is linked to a specific CMK
- KMS does not store the DEK in any way
  - it provides it to the key to the service or entity that needs it and then it discards it

The process:

1. When a data encryption key is generated KMS provides two versions of that encryption key
   a. plain text version
   b. cypher text version
2. Encrypt data with the plain text version key
   a. once the encryption is done, the plain text key is discarded
3. The encrypted key and the encrypted data are stored only

KMS does not perform the encryption or decryption on the data larger than 5 KB
- the identity or the service using KMS does

KMS does not track the usage of the data encryption keys

S3 when using encryption, generates a data encryption key for every object!

Key Concepts

1. **CMKs** are isolated to a region and they never leave that region
2. The never leave the **KMS** service. They cannot be extracted
3. There are two types of CMKs:
   a. **AWS managed CMKs**
      i. created automatically by AWS when a service uses KMS encryption
   b. **Customer Manages CMKs**
      i. are created explicitly by the customer
      ii. are much more configurable
   c. **both support key rotation**
      Rotation is the process when the physical backing material is changed
      - with **AWS Managed Keys** this cannot be disabled and is set to rotate material every 1,095 days (3 years)
      - with **Customer Managed Keys** the rotation is optional and happens once a year if its enabled
        - CMK contains the current backing key and all the previous backing keys used for backing material
        - aliases are a shortcut to a particular CMK - are also per region

Every **CMK has a customer policy**

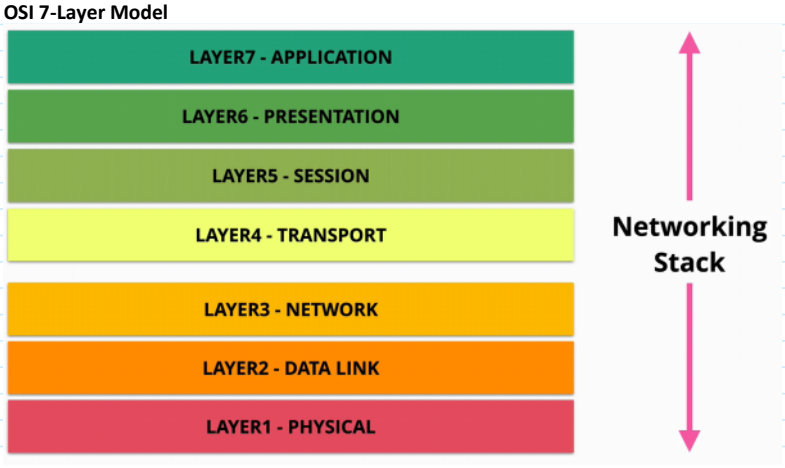
- **KMS has to be explicitly told they keys trust the AWS account they are in.**
  - the key trusts an account, so that the account can manage the keys
  - **IAM is trusted by the account, and the account needs to be trusted by the key**

Networking

Monday, June 7, 2021 3:32 PM

Whenever you use a Cloud system, is done via networking.

OSI 7-Layer Model

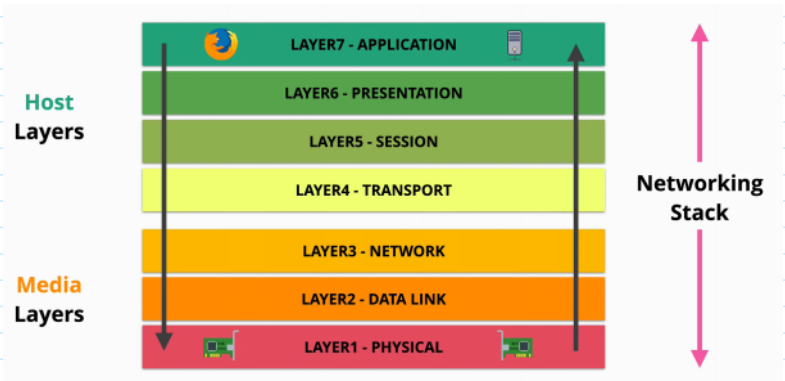


Host Layers

- **How data is chopped off and reassembled, and how is formatted so that is understandable by both sides of the network connection**

Media Layers

- **How date is moved from point A to point B**

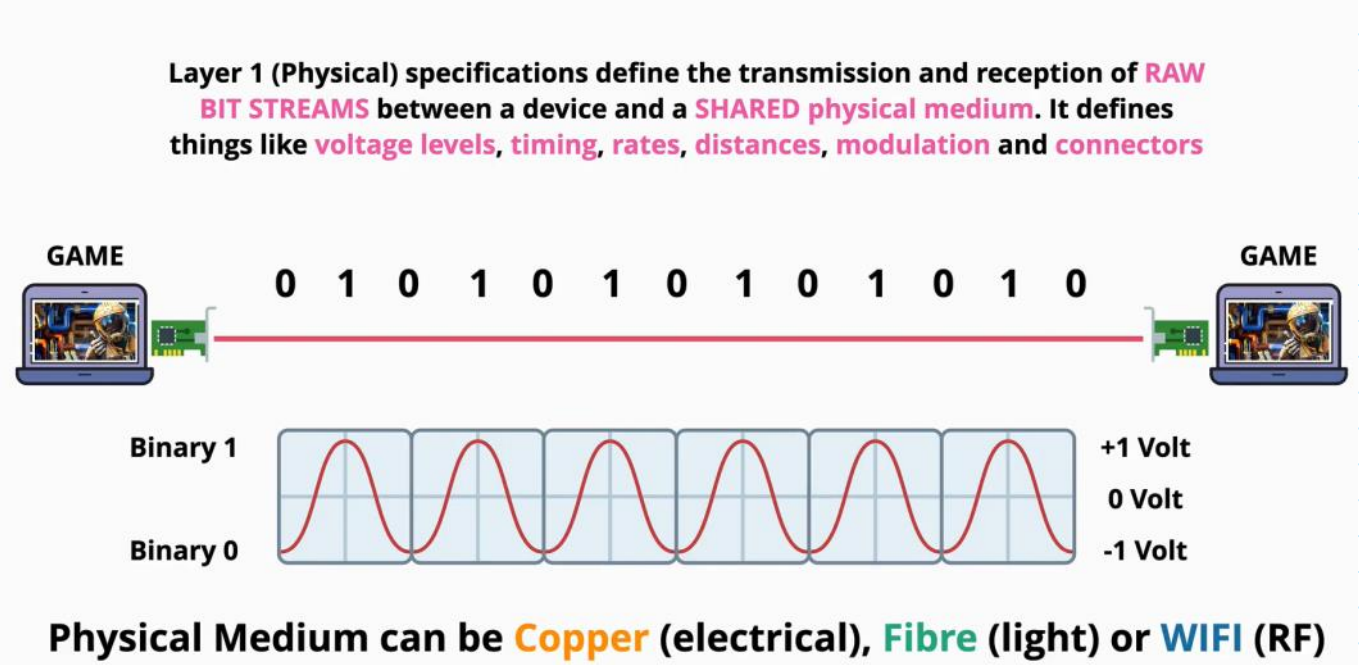


Layer 1 - Physical Layer

Monday, June 7, 2021 3:38 PM

Physical later is the networking **card**.
The connection between cards is called a **physical medium** which is usually a copper cable, optical fiber or WIFI.

For Copper cable, voltage is used to transmit data, using a low voltage as 0, and a high voltage as 1.

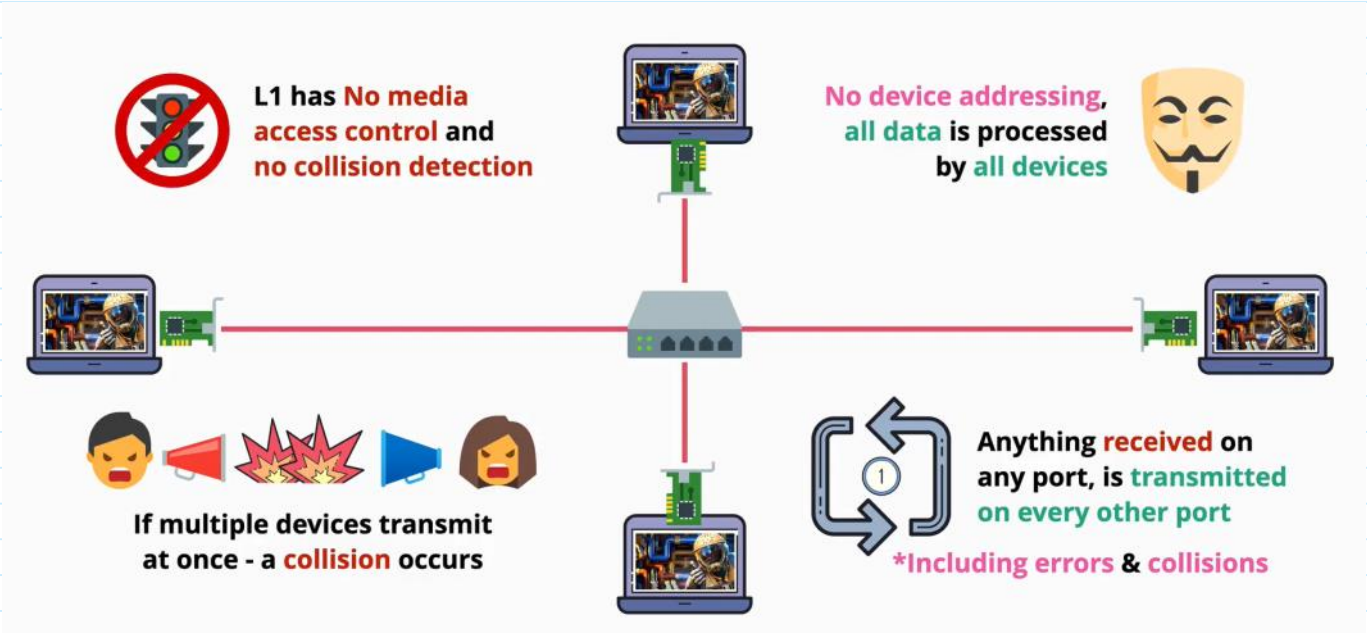


If more devices need to communicate, a **hub** is needed. A **hub** job is to transmit to every device, what is receiving from a device. Anything received on any port, is transmitted to all the other ports!

In this layer, none of the connected device has addressing (identities - names).

Two devices might transmit at the same time, fact that might results into a collision - only one device can transmit at a time, so that the information transmitted can be read by the other devices.

This layer has no media control access, or any collision detection, therefore, is a network is created using only layer 1, collision will be present.


L1 has No media access control and no collision detection
No device addressing, all data is processed by all devices
If multiple devices transmit at once - a collision occurs
Anything received on any port, is transmitted on every other port
*Including errors & collisions

Layer 2 - Data Link Layer

Monday, June 7, 2021 3:56 PM

The Data Link layer runs over the physical layer.

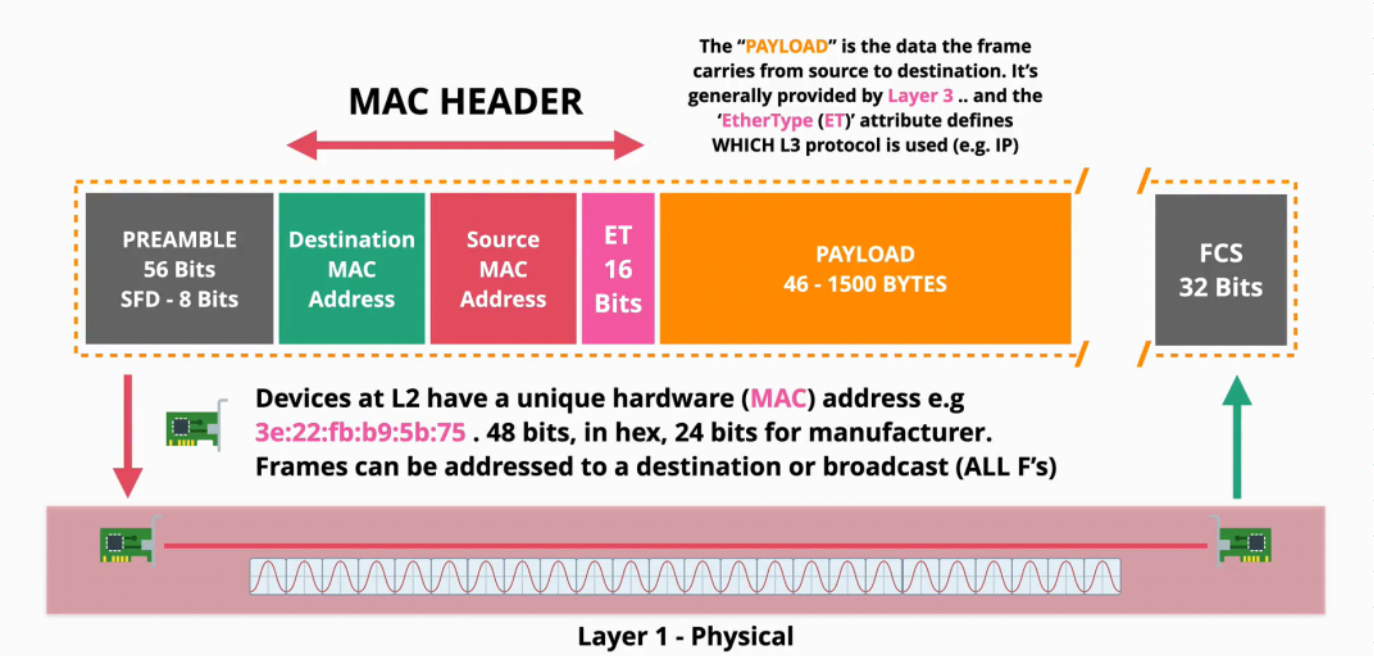
The Data Link layer uses frames to send information between devices. Devices using this layer has a unique hardware address called a **MAC address**, which is represented by hexadecimal values.
This address is not based on any software, it is uniquely attached to a piece of hardware.

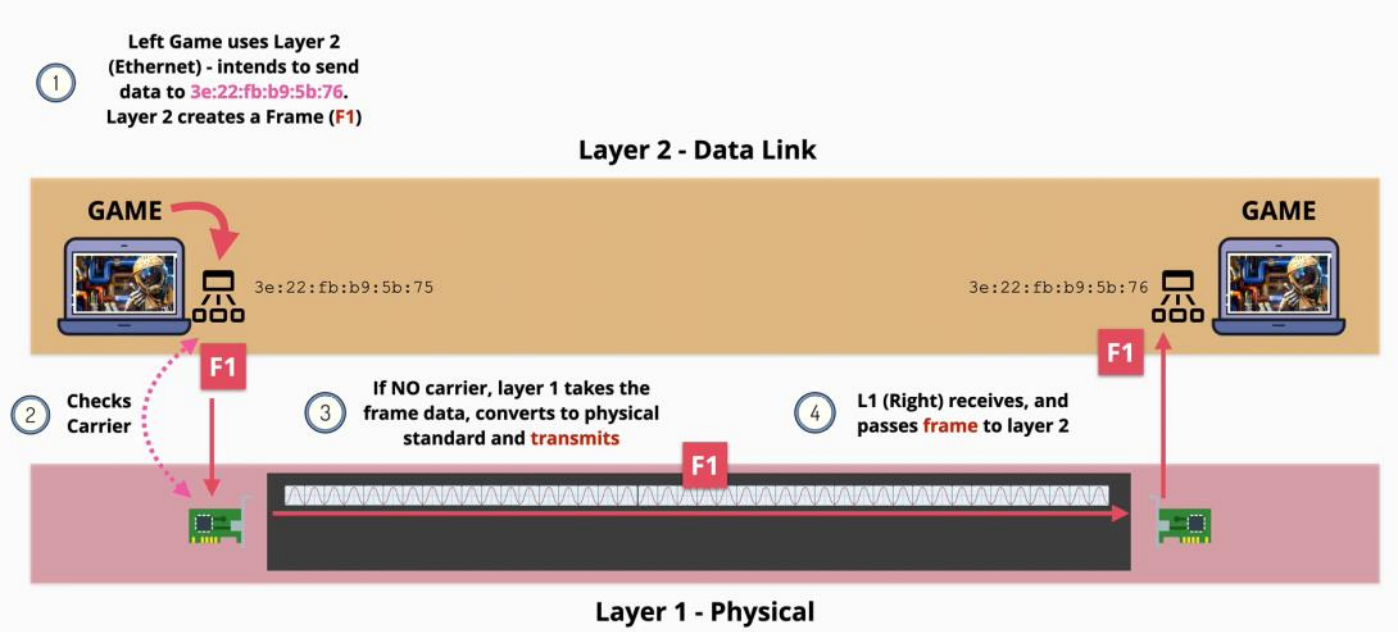
The MAC address is made by two parts:

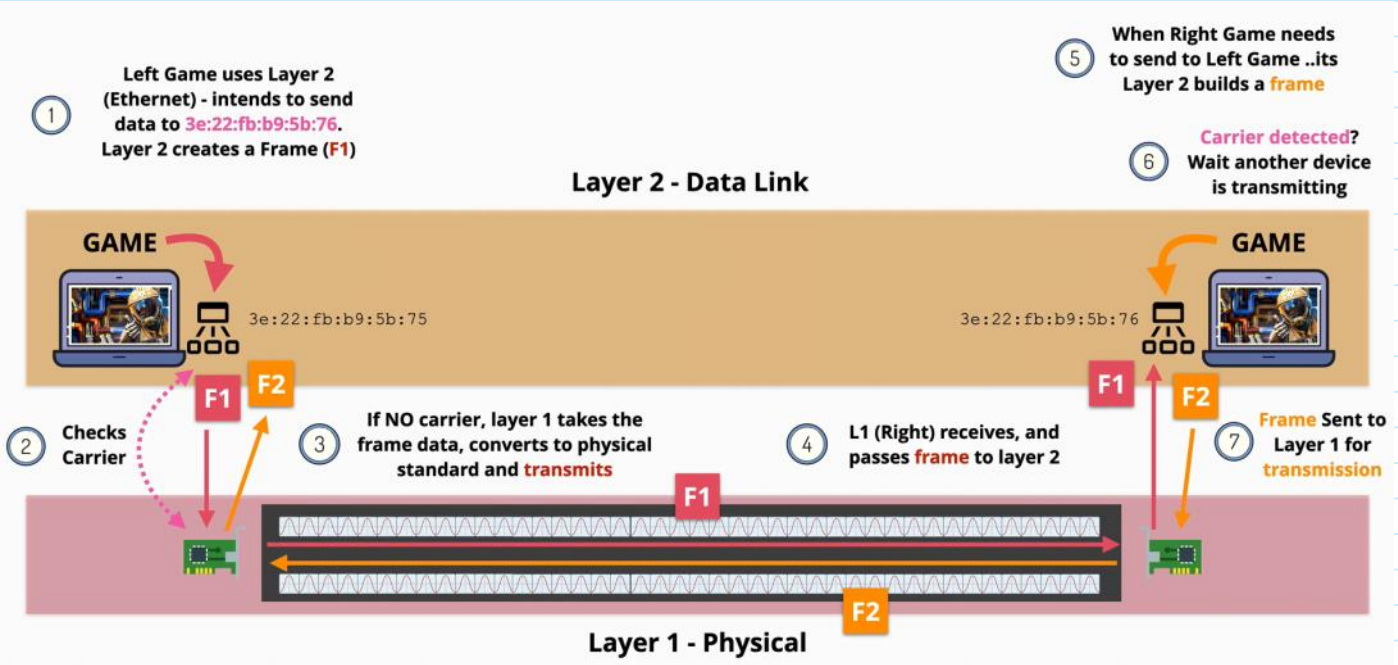
1. The OUI - organizational unique identifier
2. The NIC - network interface controller

As the data link layer is based on layer one, the information shared by layer 1. For simplicity, layer 2 transmits a frame, via the layer 1, even layer 1 does not understands what it is sharing.

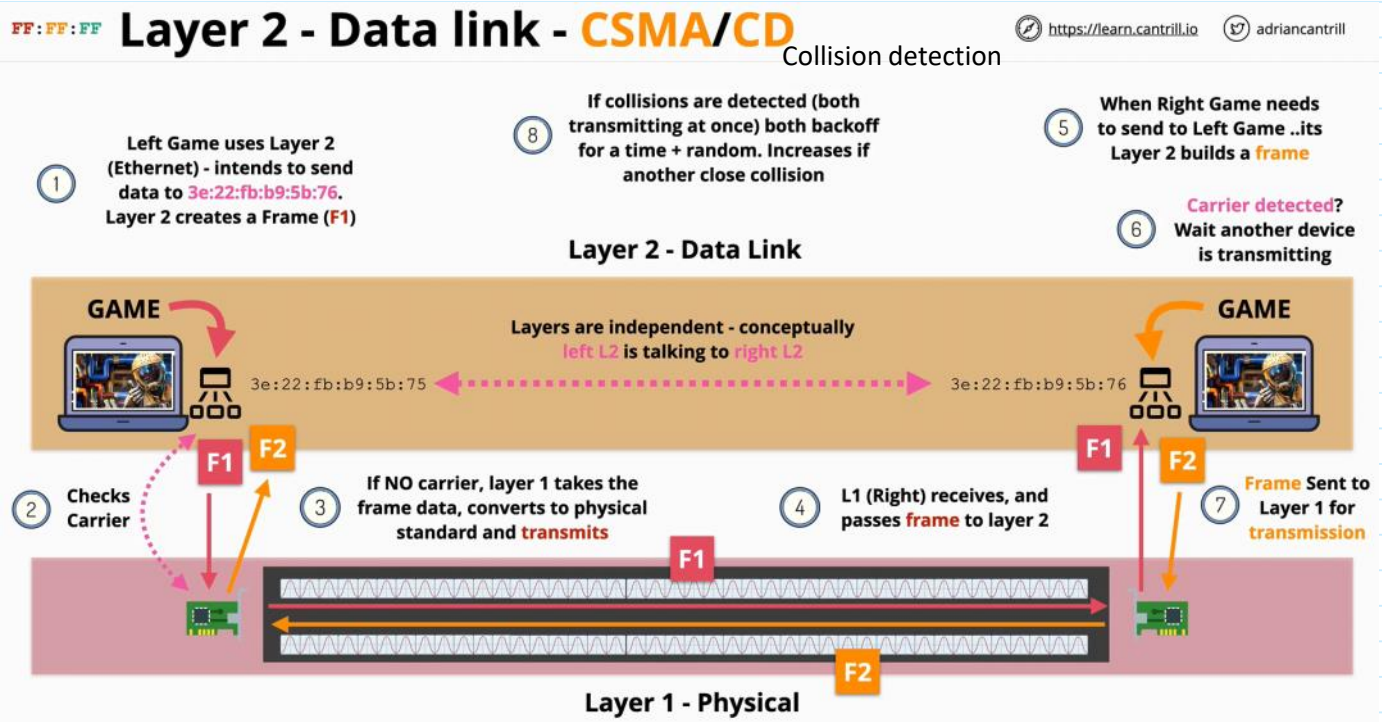
Layer 2 brings identifiable devices, senders and receivers to life.



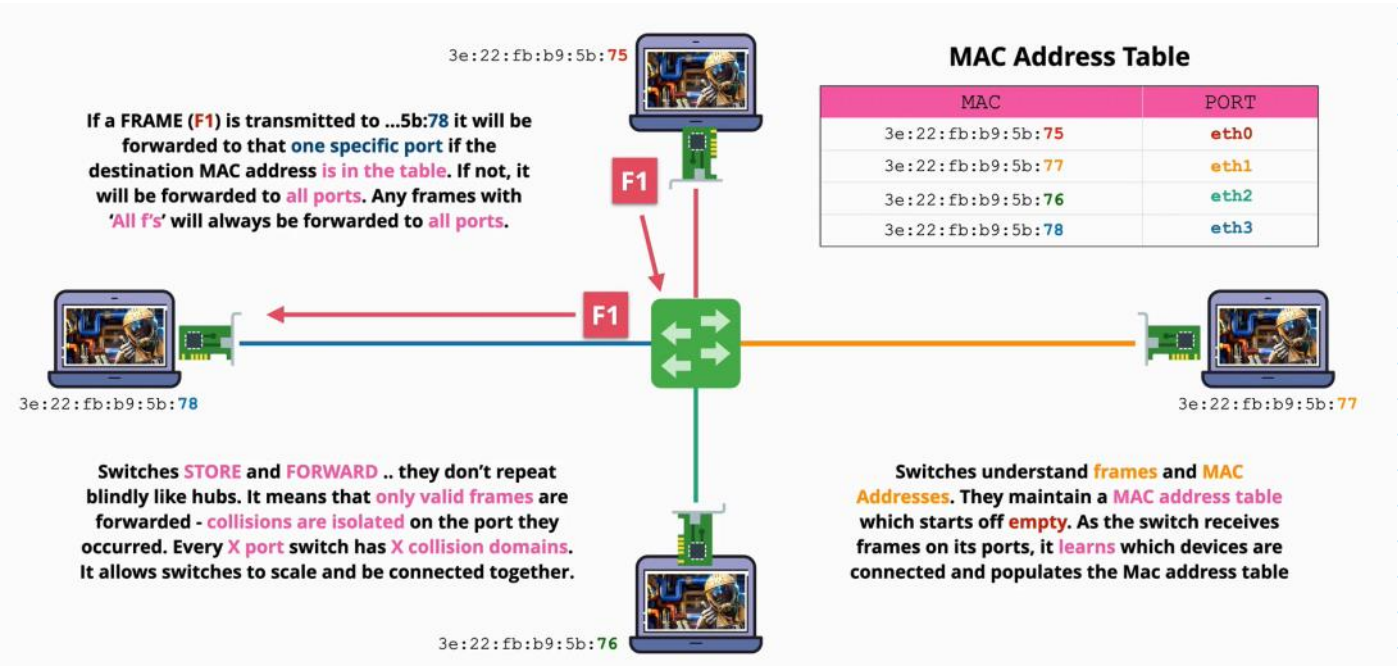




The data is **encapsulated** into the frame when transmitted, and extracted when it reaches the receiver.



A **switch** is a layer two hub, that understands frames.



Decimal <=> Binary

Monday, June 7, 2021 4:35 PM

133.33.33.7 -> dotted decimal notation

133	10000101
33	00100001
33	00100001
7	00000111

Position	1	2	3	4	5	6	7	8
Binary Position Value	128	64	32	16	8	4	2	1
Binary Value								

128	64	32	16	8	4	2	1

Layer 3 - Network Layer

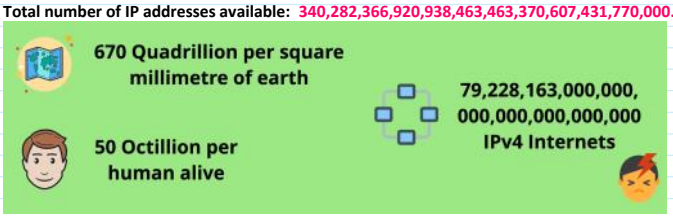
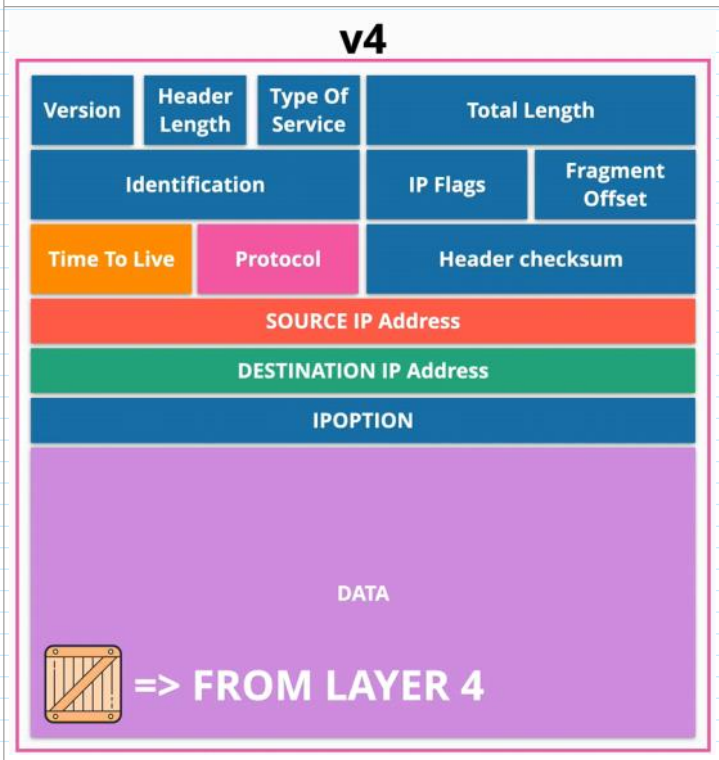
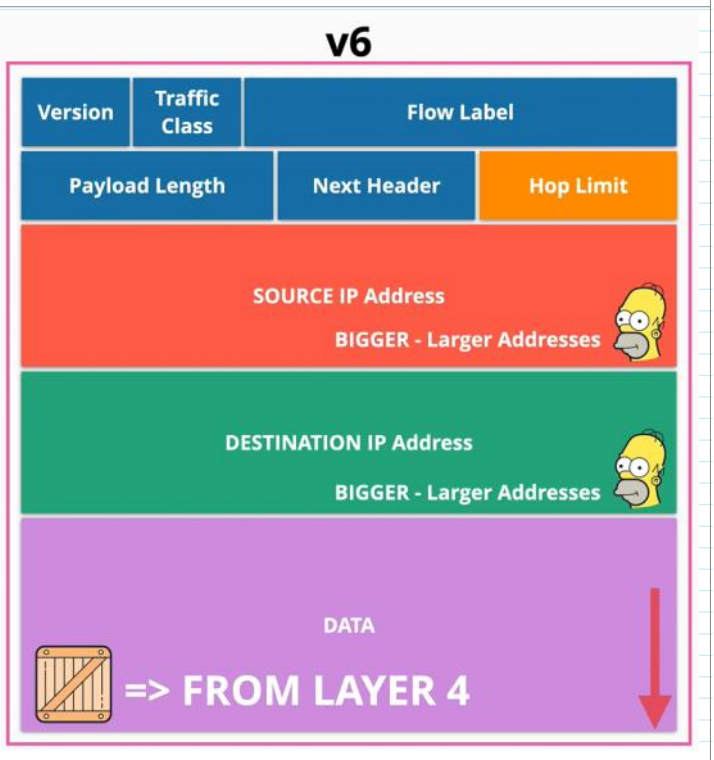
Tuesday, June 8, 2021 2:42 PM

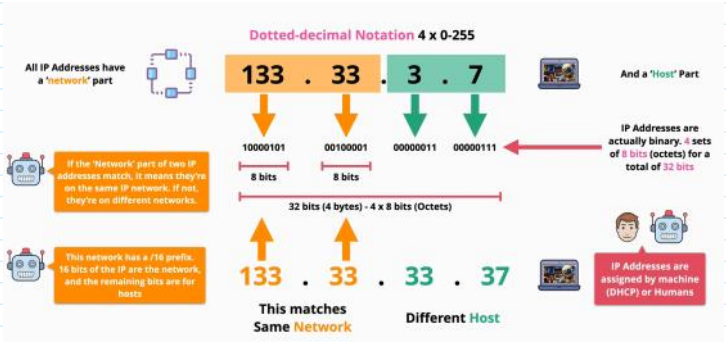
Ethernet is the most popular layer 2 connection.

Layer 3 is the common protocol which can spam multiple different Layer 2 networks.
IP address comes with Layer 3.
Routers are layer 3 devices which move packets of data across different networks.

Packets - data unit used by the layer 3 protocol.
- are very similar to frames
  - within frames the source and the destination are local
  - with packets, the destination and source could be on different sides of the planet
- packets never change when transitioning from source to destination
  - the frames are changed as the packets go from one local network to another
  - the packets are encapsulate in the frame

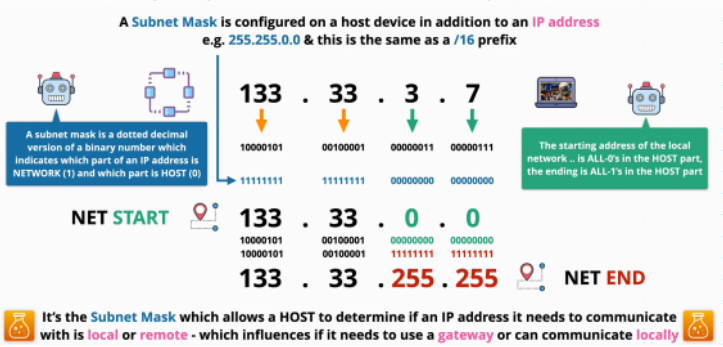
IP addresses

version 4	**version 6**
• destination IP • source IP • protocol - which protocol is used • ICMP, TCP, UDP • data • time to live (TTL) - if the packets cannot reach its destination, this number declares how many hops the packet can take (so that it won't travel forever) before being discarded • **Total number of IP addresses available:** 4,294,967,296	• destination IP • source IP • data • hop limit (time to live) • **Total number of IP addresses available:** 340,282,366,920,938,463,463,370,607,431,770,000… • 
	



First two digits - **network**
Last two digits - **host**

Subnet masks allows a host to determine if an IP address is local or remote. The result allows the sender to know if it needs a gateway (router) or it can communicate locally.



255.255.0.0 -> 11111111.11111111.00000000.00000000 /16 (ones)

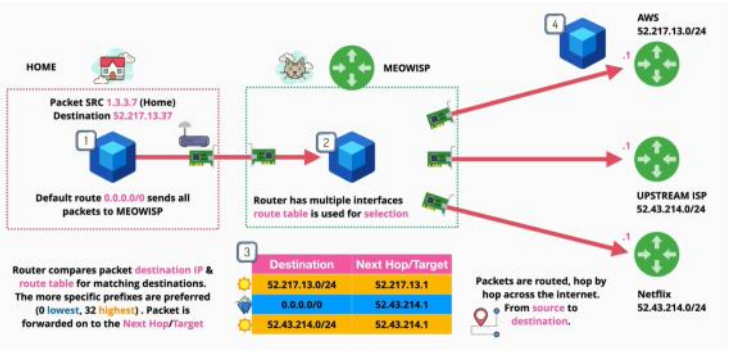
everything with ones represents the network
everything with zeros represents the host

All data generated by your router is by default sent through the internet service provider.
The internet provider has multiple interfaces route table to forward the data

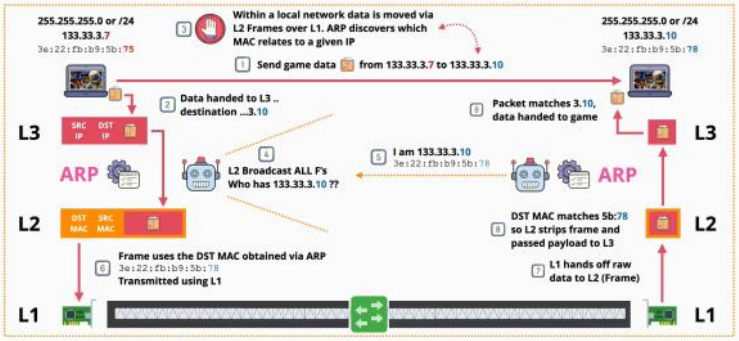
1. **each router has multiple route tables - a collection of routes**
2. **every packet that arrives to the internet provider will check its destination, and if a match is found the data is sent to the next hop according to its destination**

0.0.0.0/0 - default route

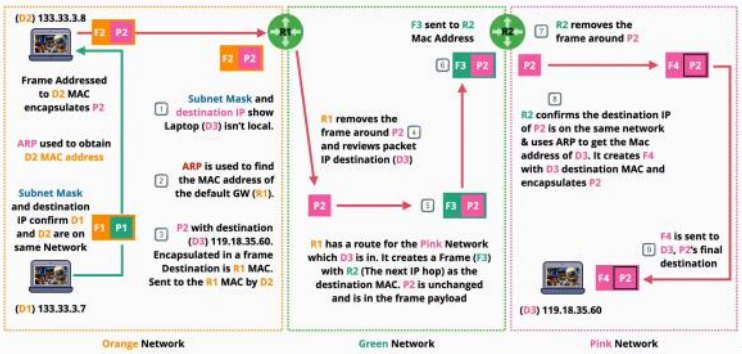
3. **each time a hop is made, the frame where the packet is encapsulated is changed, according to the new destination**



Address Resolution Protocol (ARP)

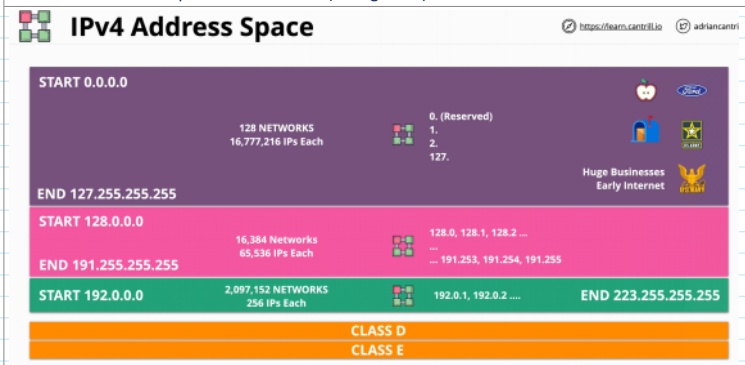


ARP is responsible of finding the **MAC** address of the destination/next hop **host.**



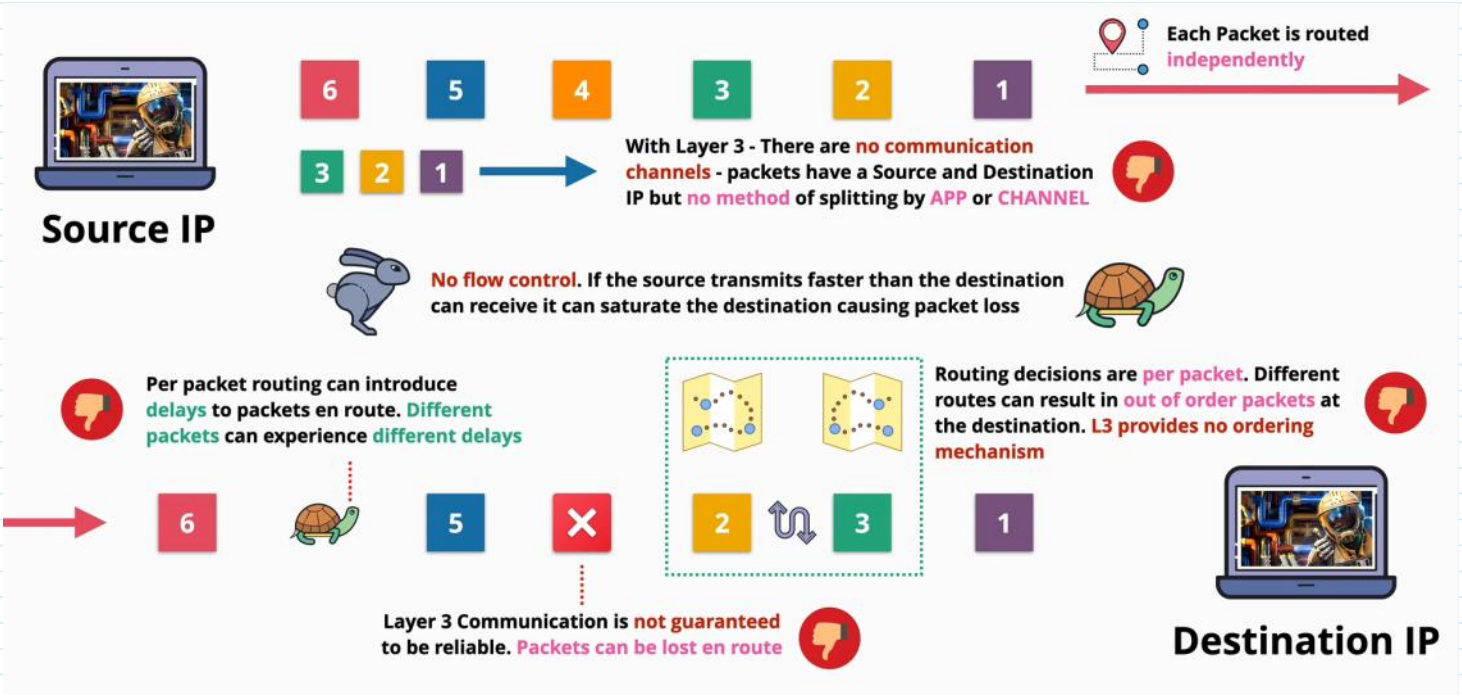
1. **If routers receive packets within a frame that are not for them, they are checking the routing table, match the next hop and create a new frame for the packet, before it sends the frame to a new router**
2. **However, if a host receives a frame that is not destinated for it, it drops the frame and packet.**

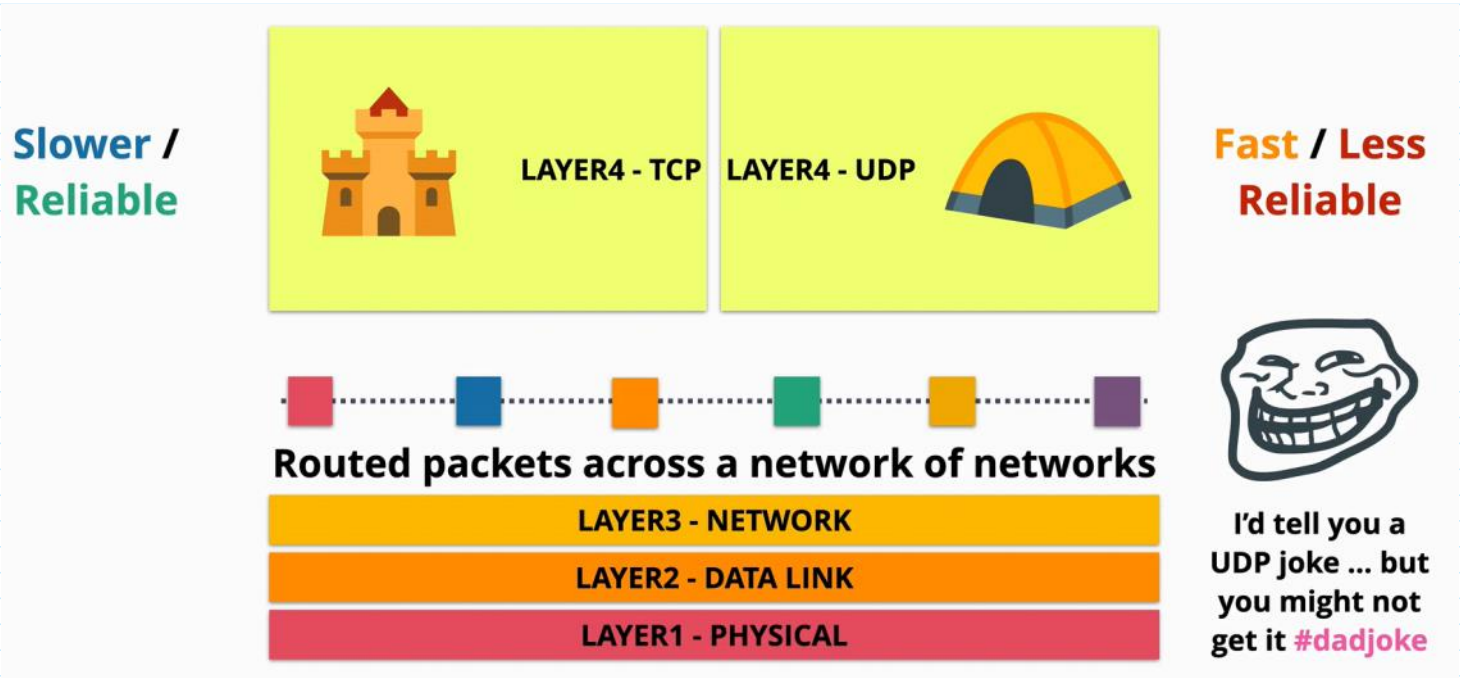
1. **Layer 3 has:**
    a. **routes - where to send the packet**
    b. **route tables**
    c. **ARP - find the mac address for a certain IP**
    d. **IP addresses**
    e. can deliver packets out of order (wrong order)



Layer 4 - Transport Layer

Wednesday, June 9, 2021 1:36 PM



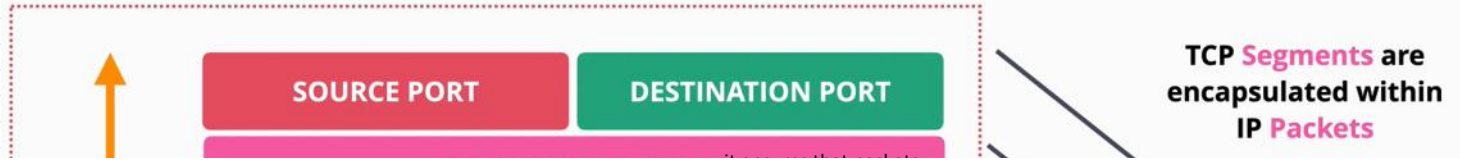


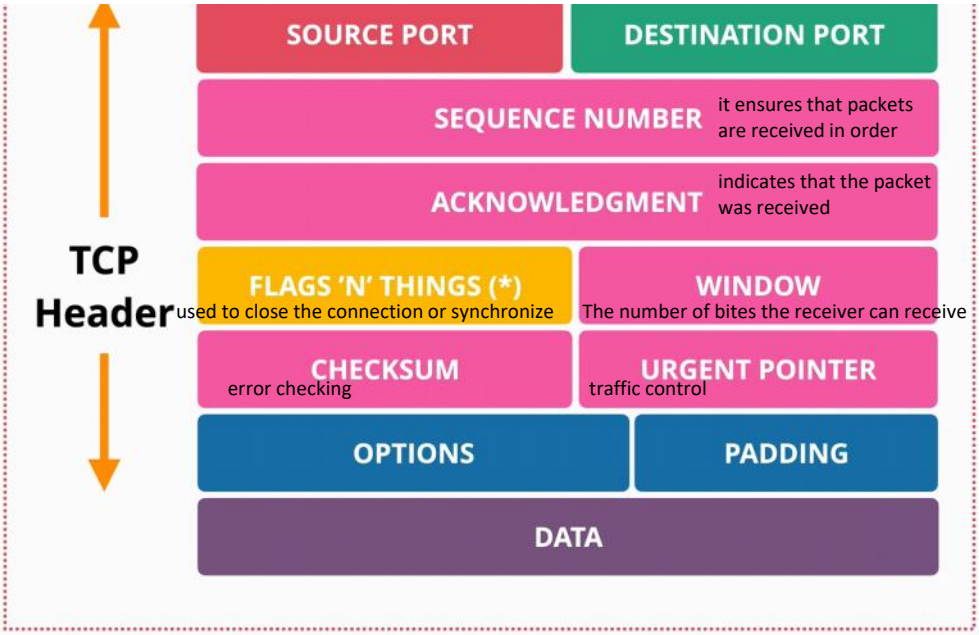
Layer 4 runs over Layer 3, therefore it needs IP addresses.

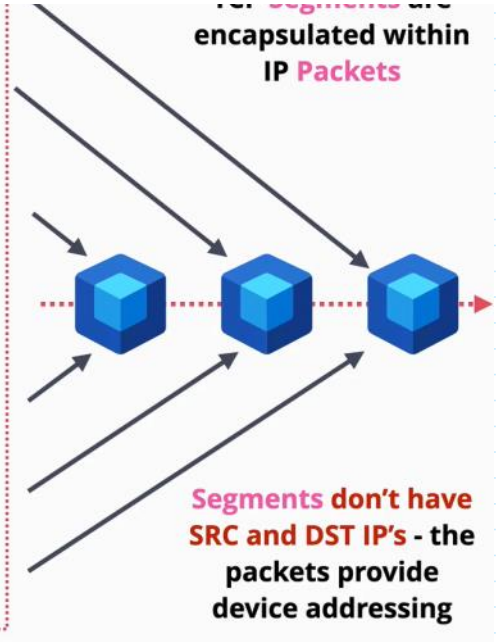
- **TCP -** has a reliable connection
- **UDP -** is al about performance

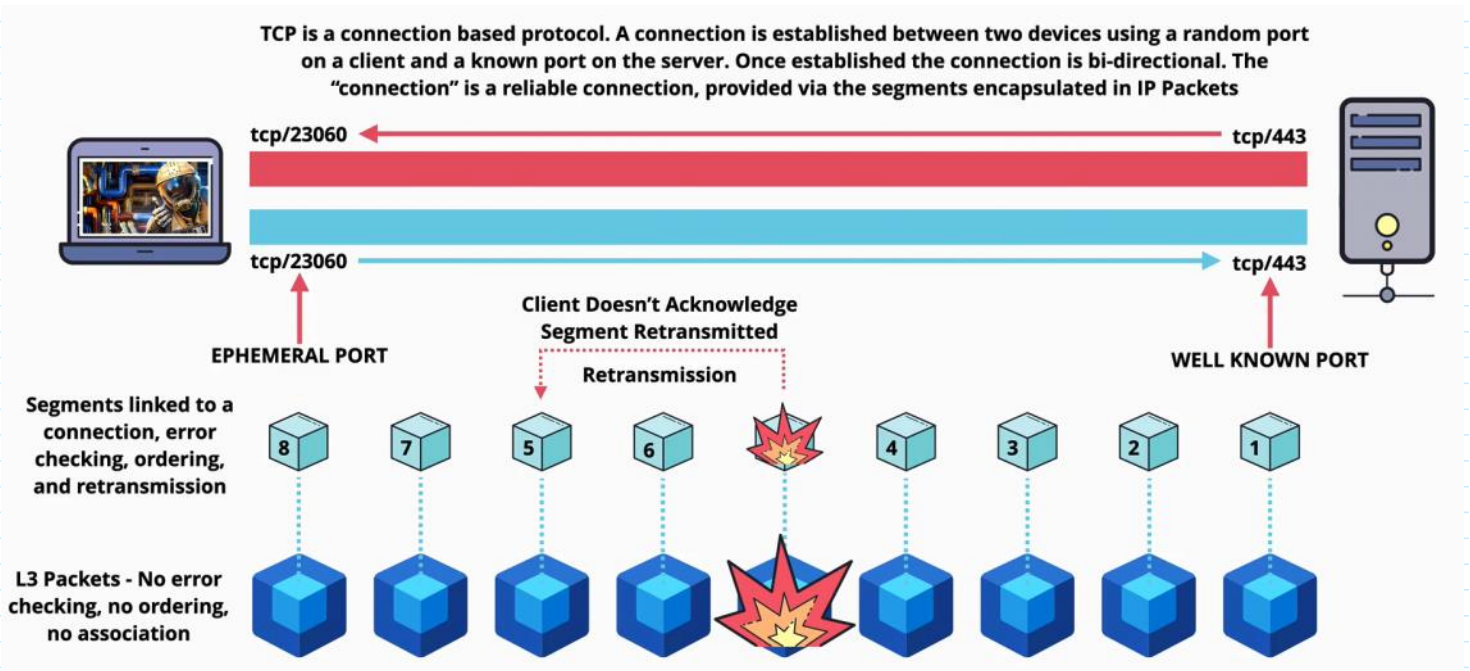
TCP introduces the term of **segments,** which are another type of containers as packets and frames are.

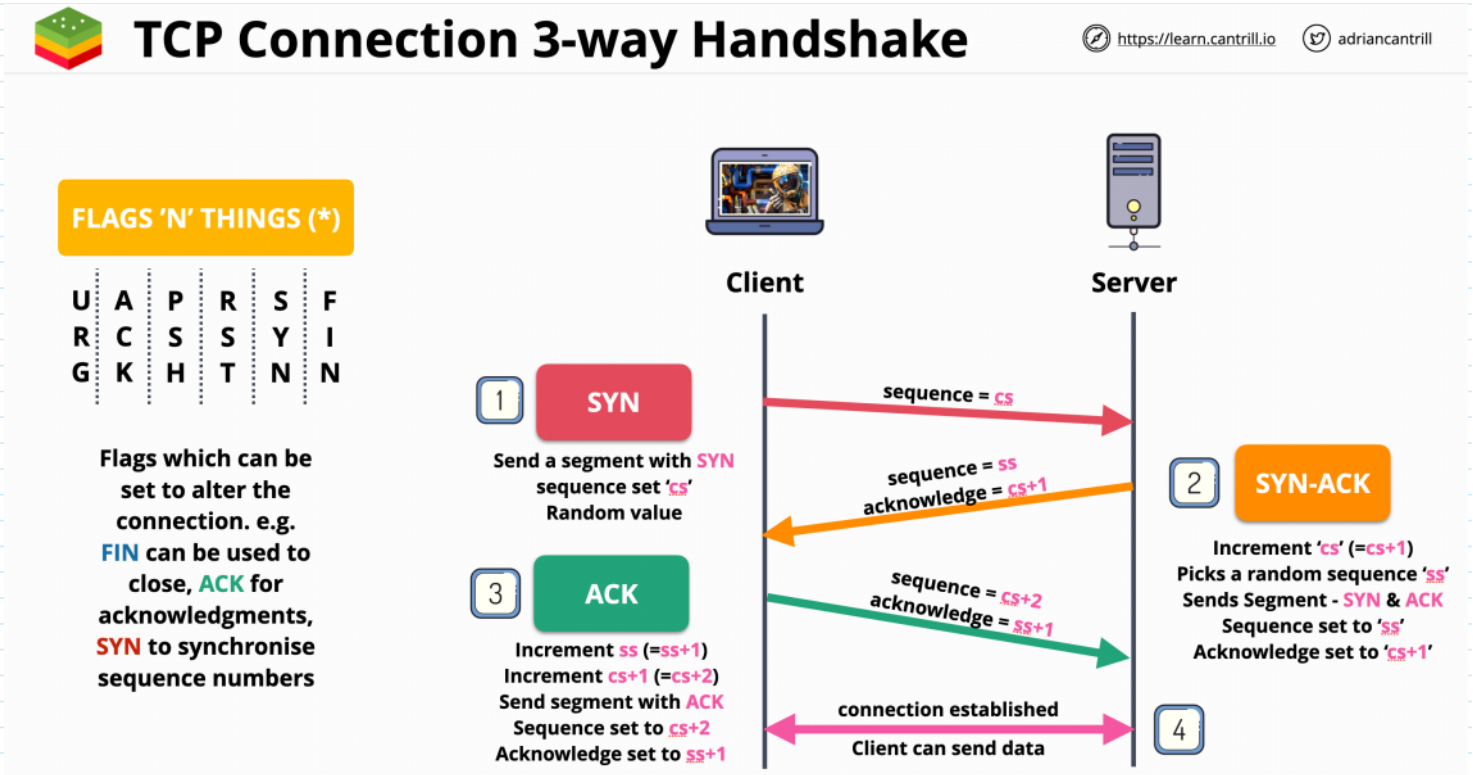
- are encapsulated in packets
- they do not have source and destination information because they the IP packets for the transit

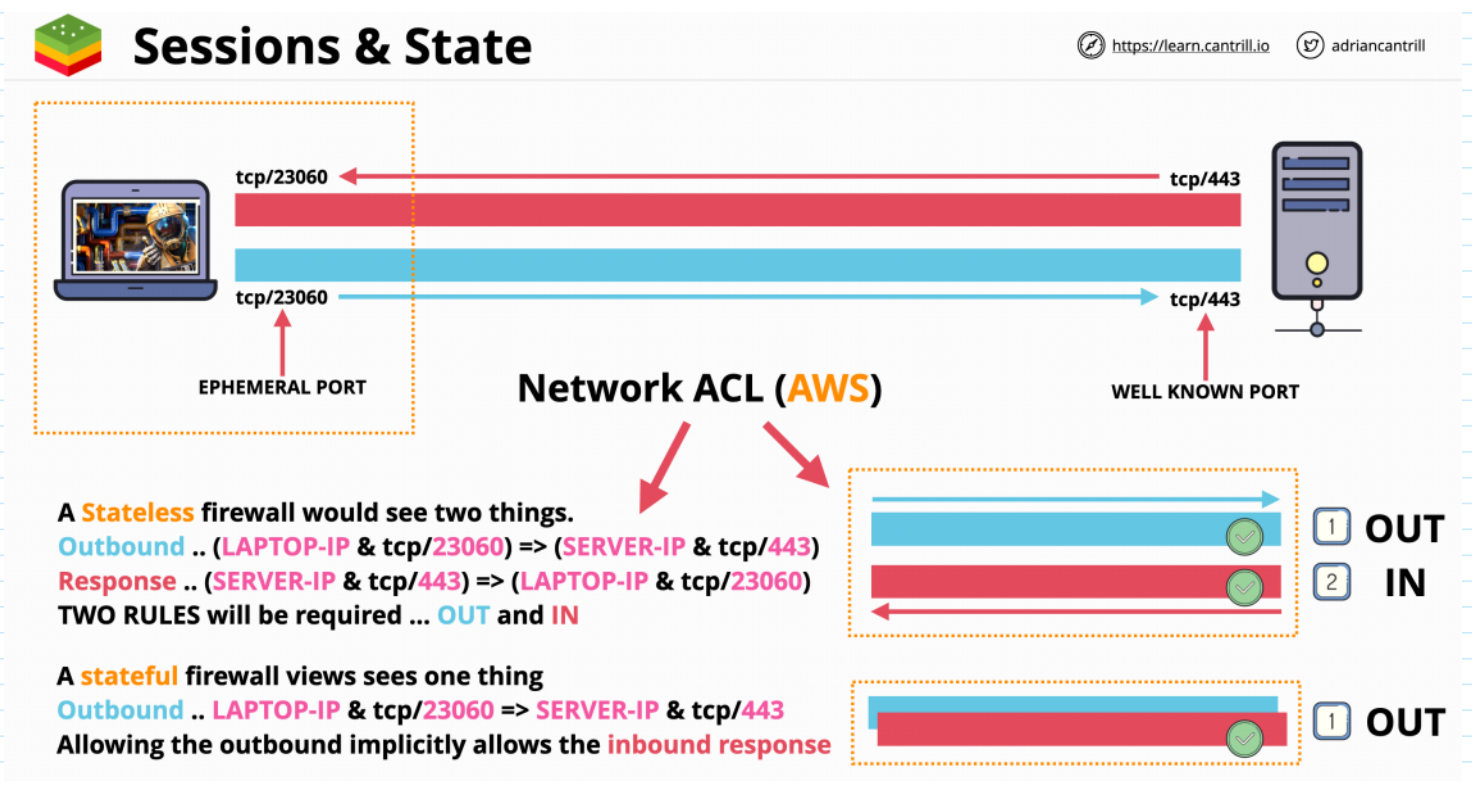



TCP Header
SOURCE PORT
DESTINATION PORT
SEQUENCE NUMBER
it ensures that packets are received in order
ACKNOWLEDGMENT
indicates that the packet was received
FLAGS 'N' THINGS (*)
used to close the connection or synchronize
WINDOW
The number of bites the receiver can receive
CHECKSUM
error checking
URGENT POINTER
traffic control
OPTIONS
PADDING
DATA


encapsulated within IP Packets
Segments don't have SRC and DST IP's - the packets provide device addressing


TCP is a connection based protocol. A connection is established between two devices using a random port on a client and a known port on the server. Once established the connection is bi-directional. The "connection" is a reliable connection, provided via the segments encapsulated in IP Packets
tcp/23060
tcp/443
tcp/23060
tcp/443
EPHEMERAL PORT
Client Doesn't Acknowledge Segment Retransmitted
Retransmission
WELL KNOWN PORT
Segments linked to a connection, error checking, ordering, and retransmission
L3 Packets - No error checking, no ordering, no association
8
7
5
6
4
3
2
1





Stateless firewall does not understand the state of a connection.
Rules

1. Outbound: send -> **OUT**
2. Response: receive **-> IN**

Stateful firewall understands the states of the TCP segments.
It sees the outbound and the response traffic as one, therefore once the connection is established, it automatically allows the response.

TCP Well Known Ports

- tcp/80 - HTTP & tcp/443 HTTPS
- tcp/22 - SSH
- tcp/25 - SMTP (email)
- tcp/21 - TELNET
- tcp/3389 - REMOTE DESKTOP PROTOCOL
- tcp/3306 - MySQL/MariaDB/Aurora

Network Address Translation (NAT)

Wednesday, June 9, 2021 3:51 PM

Layer 5 - Session Layer

Wednesday, June 9, 2021 1:40 PM

Network Address Translation (NAT)

Wednesday, June 9, 2021 4:14 PM

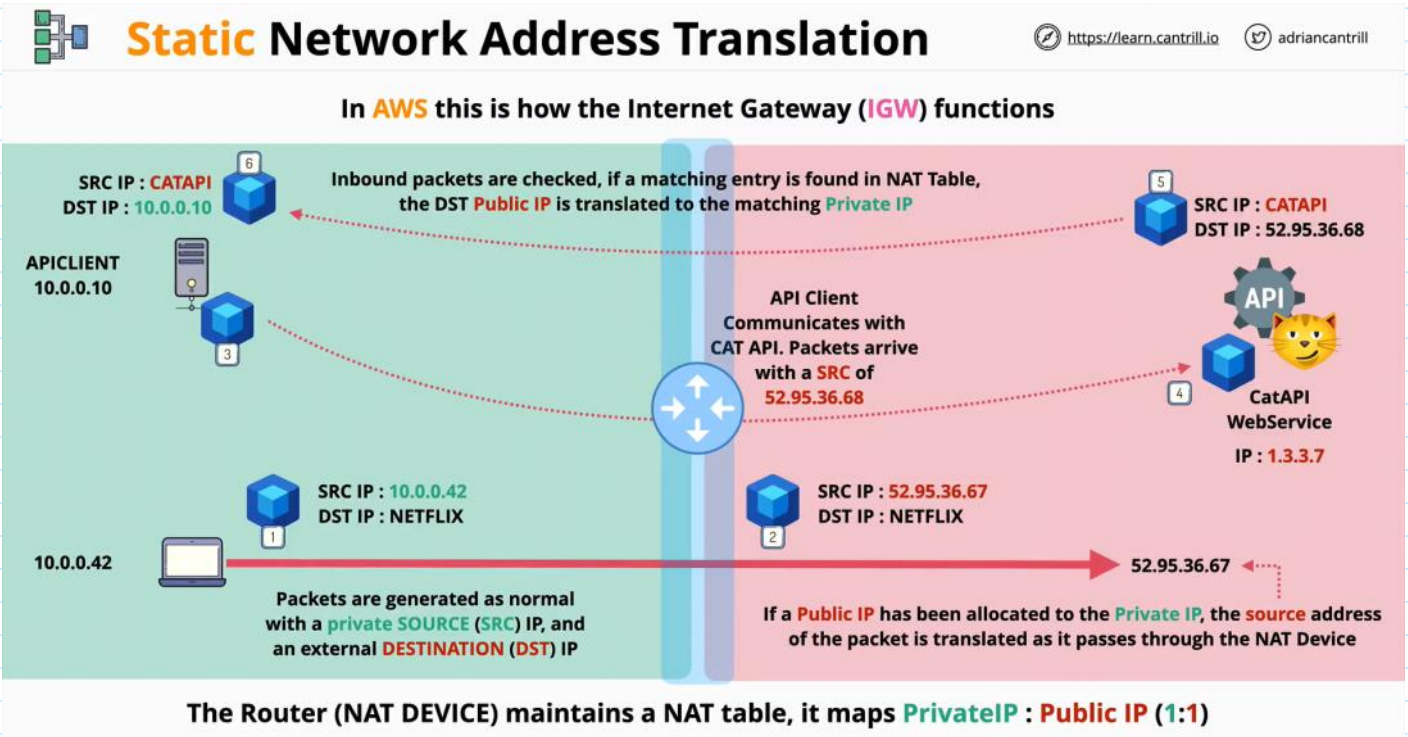
NAT is a process that is designed to address the growing shortage of IPv4 addresses.

- NAT is designed to overcome **IPv4 shortages**
- ... also provides some **security benefits**
- Translates **Private** IPv4 addresses to **Public**
- **Static NAT** - **1 private** to **1 (fixed) public address** (IGW) **one to one**
- **Dynamic NAT** - **1 private** to **1st available Public** **one to many (from a pool)**
- Port Address Translation (**PAT**) - **many private** to **1 public** (NATGW) **many to one**
- **IPv4 only** ... makes no sense with IPv6

IPv6 does not need NAT, as there is **no** need for **private IP addressed**

Static NAT

Private IP addresses cannot communicate over the internet with the public IP addresses.

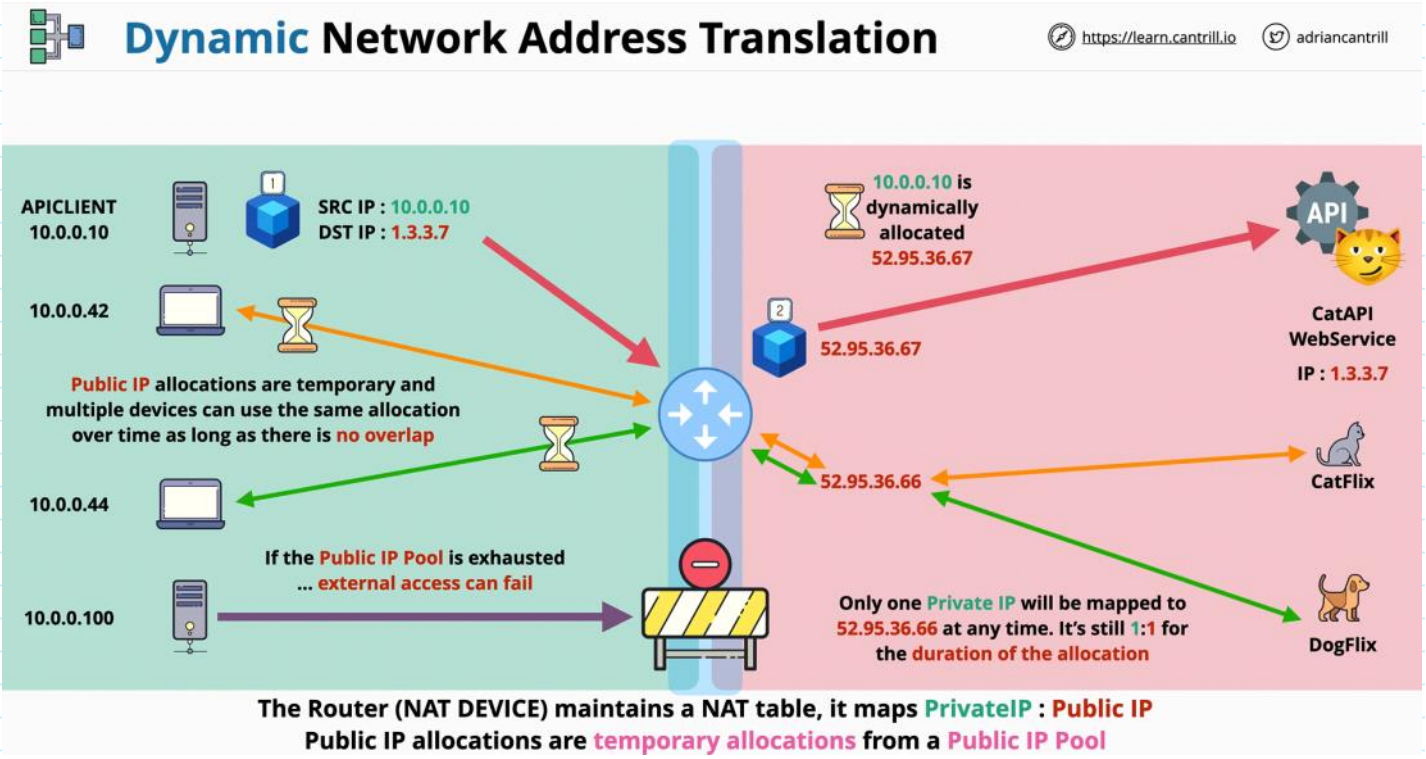


Is important to remember that in all the time, the two private devices, are never assigned any public IP address. The NAT device, is routing the packets to the private devices.

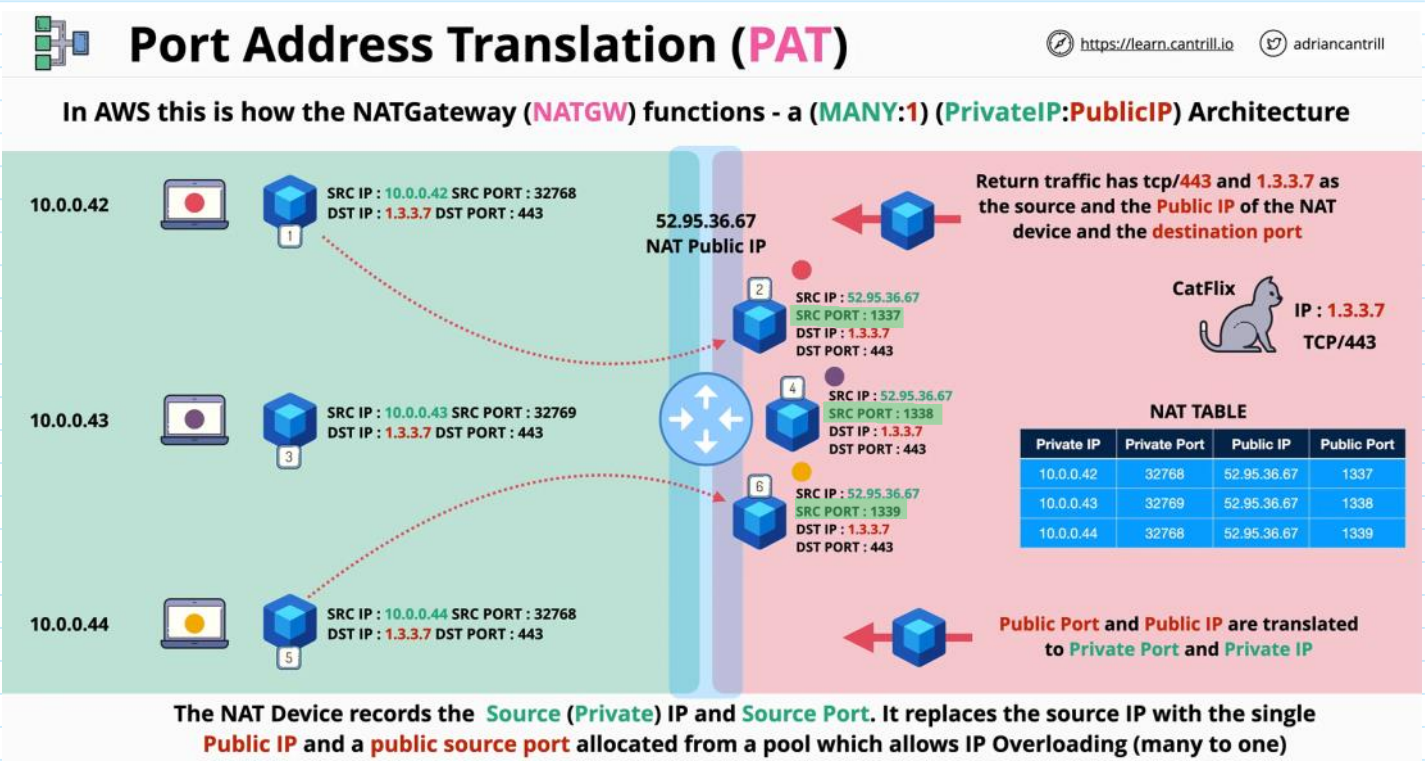
Dynamic NAT

Dynamic NAT is applied when there are not enough public IP addresses for all the private devices,

therefore a pool of public IP address is temporary allocated and used as they are needed.

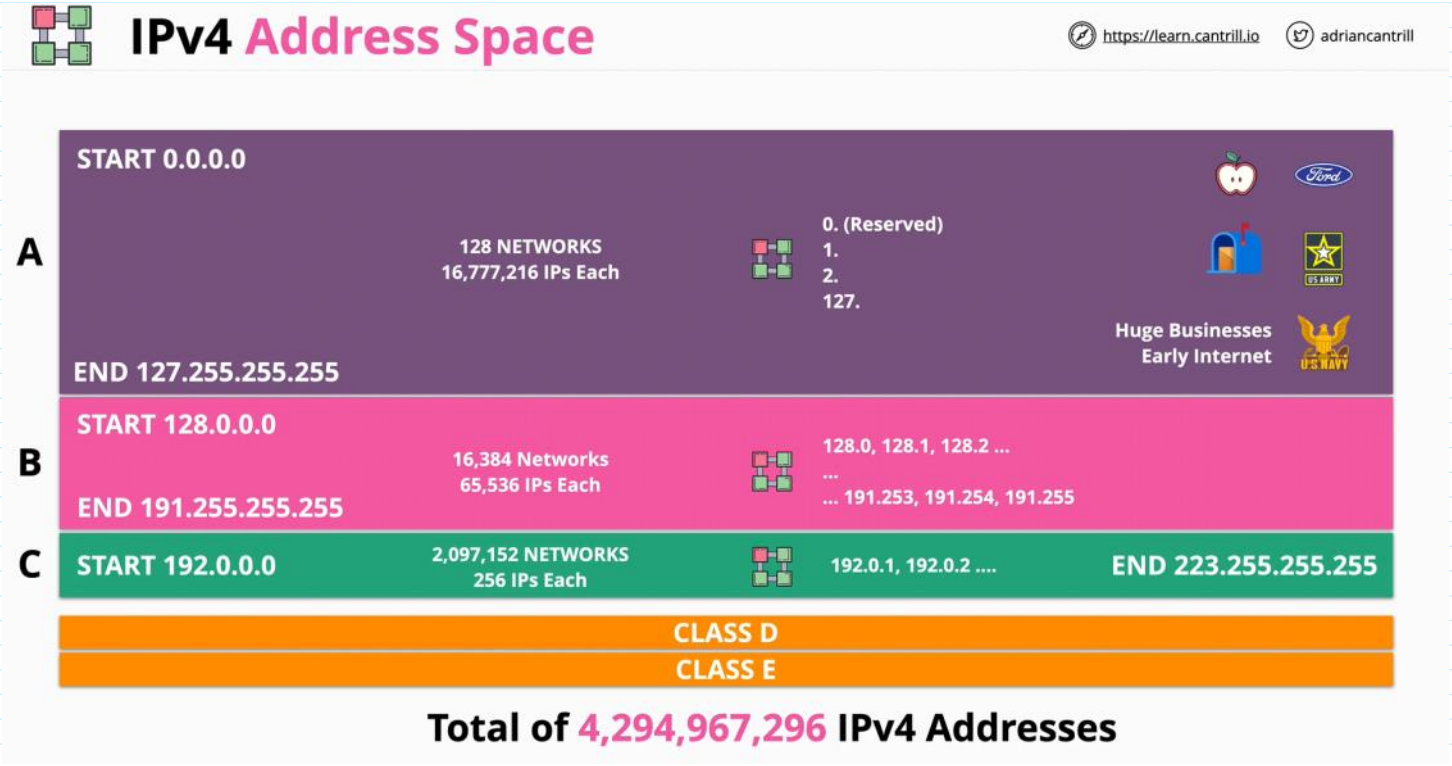


Port Address Translation (PAT)



IPv4 Addressing and Subnetting

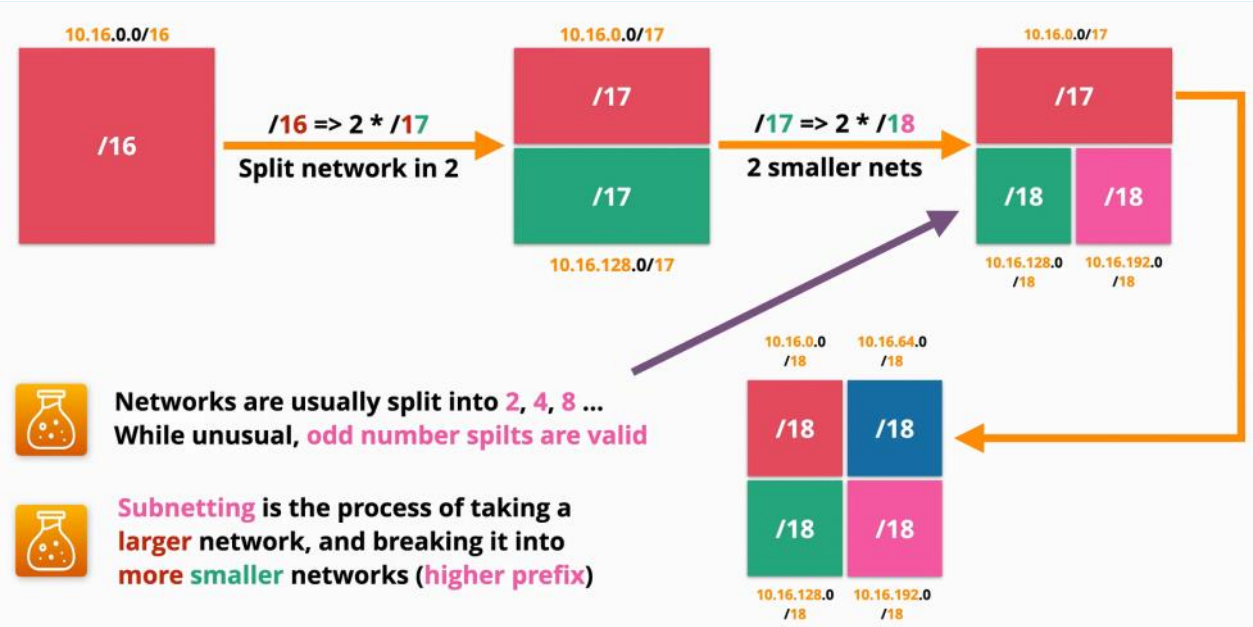
Wednesday, June 9, 2021 5:07 PM



10.16.0.0 /16 (first 16 zeros in binary) - means that the last two octets are available for hosts or subnetting.

1. **starts are 10.16.0.0**
2. **ends at 10.16.255.255**

/0	all internet
/8	class A
/16	class B
/24	class C
/32	a single IP



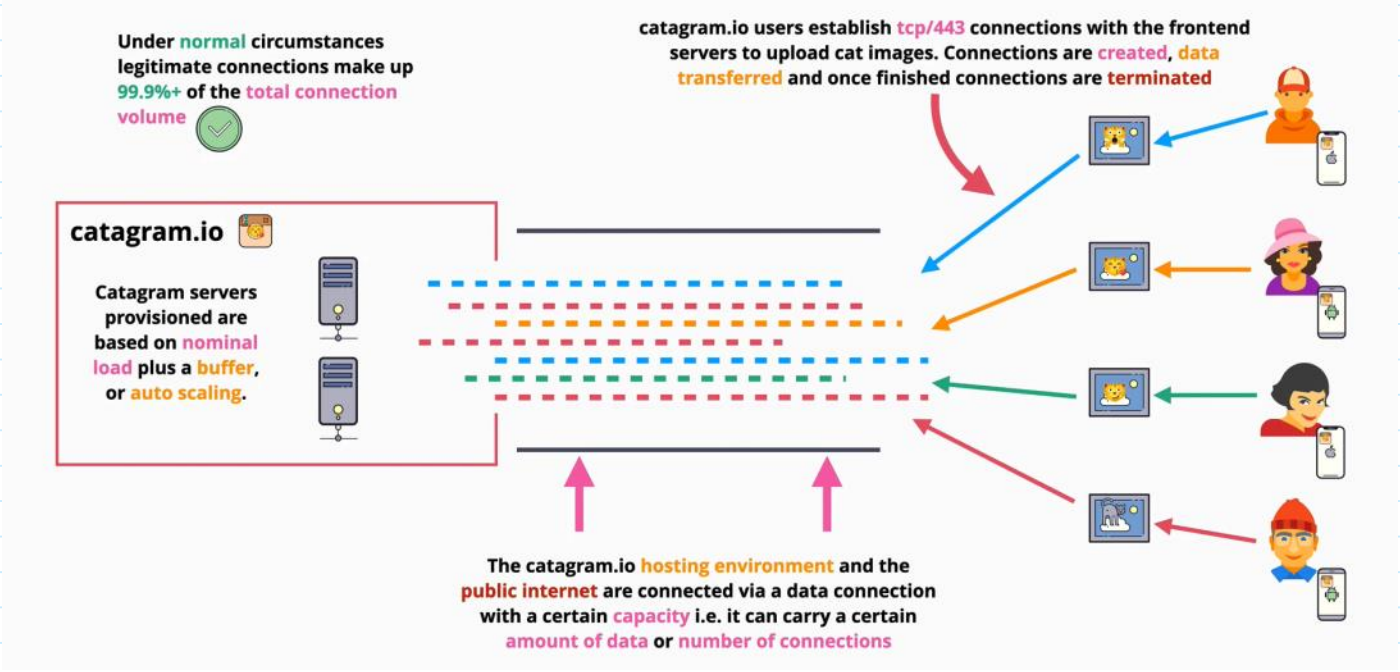
10.16.0.0	**00001010.00010000**.00000000.00000000/**16**	255
	00001010.00010000.00000000.00000000/17	128
	00001010.00010000.00000000.00000000/18	64

1	2	4	8	16	32	64	128	256
0	1	0	1	0	0	0	0	0
0	0	0	0	1	0	0	0	0

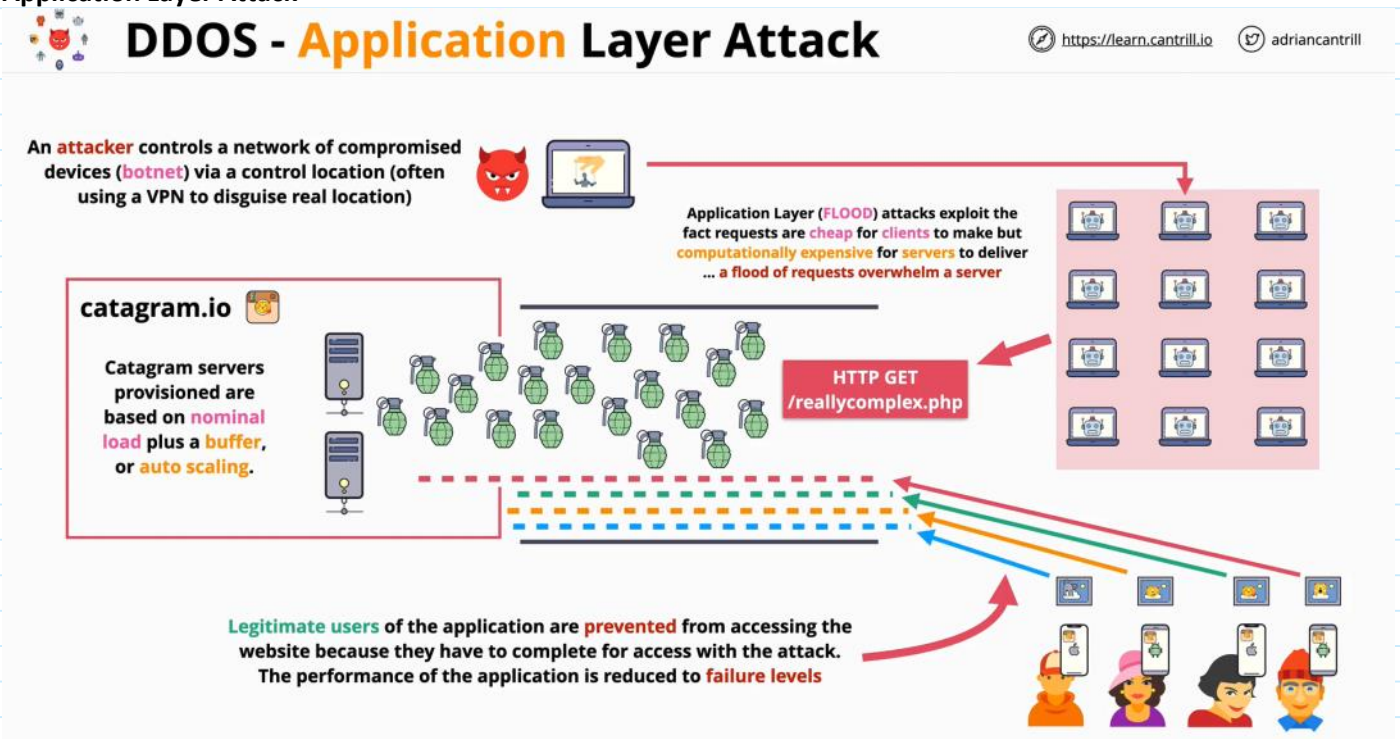
Distributed Denial of Service (DDos)

Thursday, June 10, 2021 1:26 PM

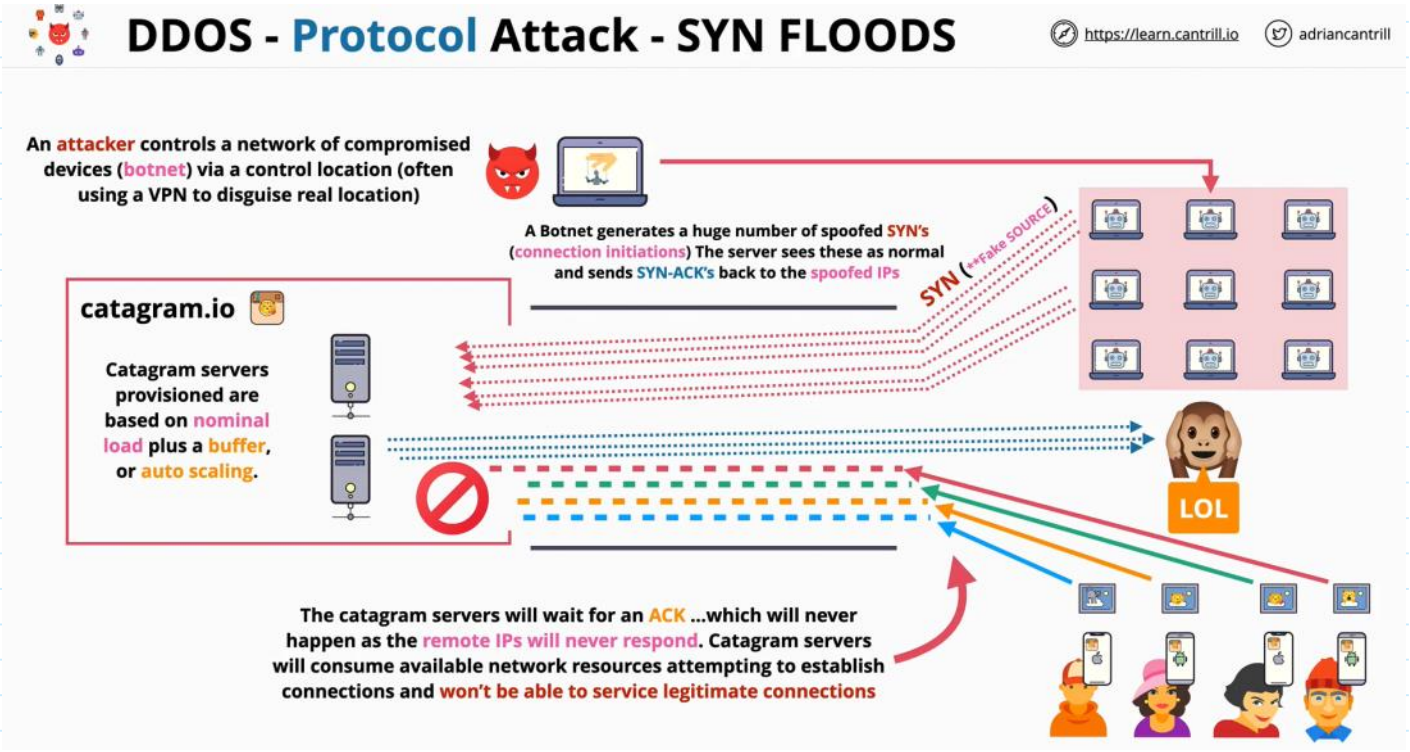
DDoS are attacks designed to **overload** websites.



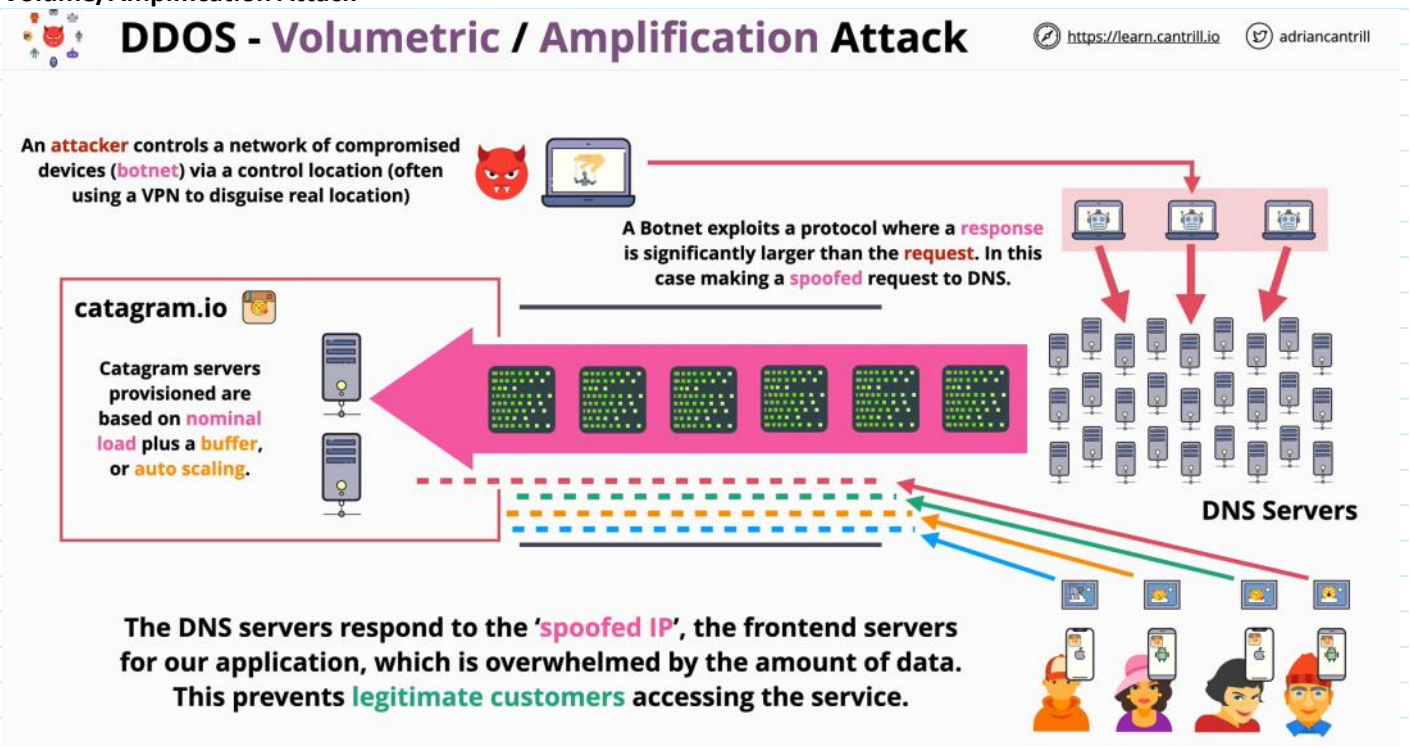
Application Layer Attack



Protocol Attack - SYN Floods



Volume/Amplification Attack



The attacks cannot be avoided with normal network securities measures. Mitigating a DDoS attack, you need to be careful to not block real users!

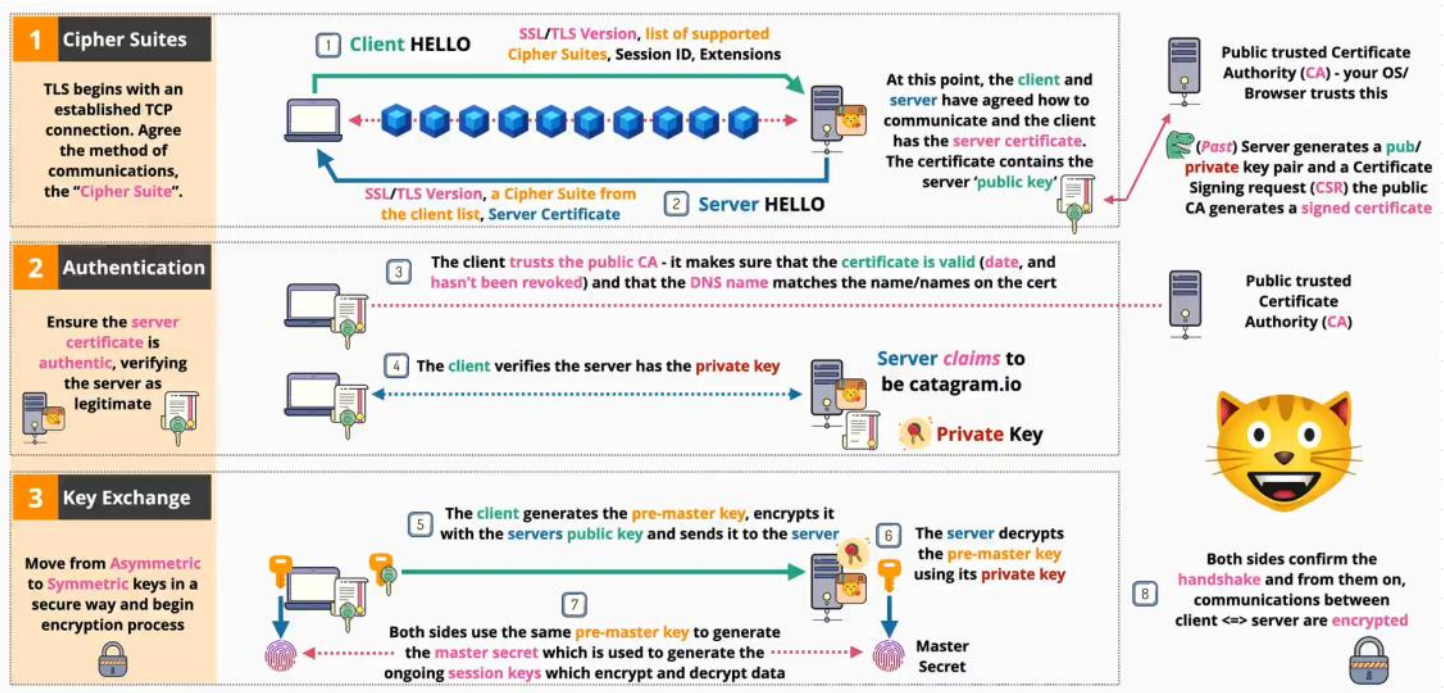
SSL and TLS

Thursday, June 10, 2021 1:52 PM

SSL -> Secure Sockets Layer
TLS -> Transport Layer Security

They both focus on the **privacy** and **data integrity** between the **client** and the **server**.

Privacy - communication are encrypted (asymmetric)
Identity verification - client/server verified
Reliable connection - protection against alteration



AWS Fundamentals

Thursday, June 10, 2021 1:52 PM

Public vs Private Services

Thursday, June 10, 2021 2:36 PM

Public and **private** keyword in AWS services are referring to the networking **only**!

When connecting to AWS:

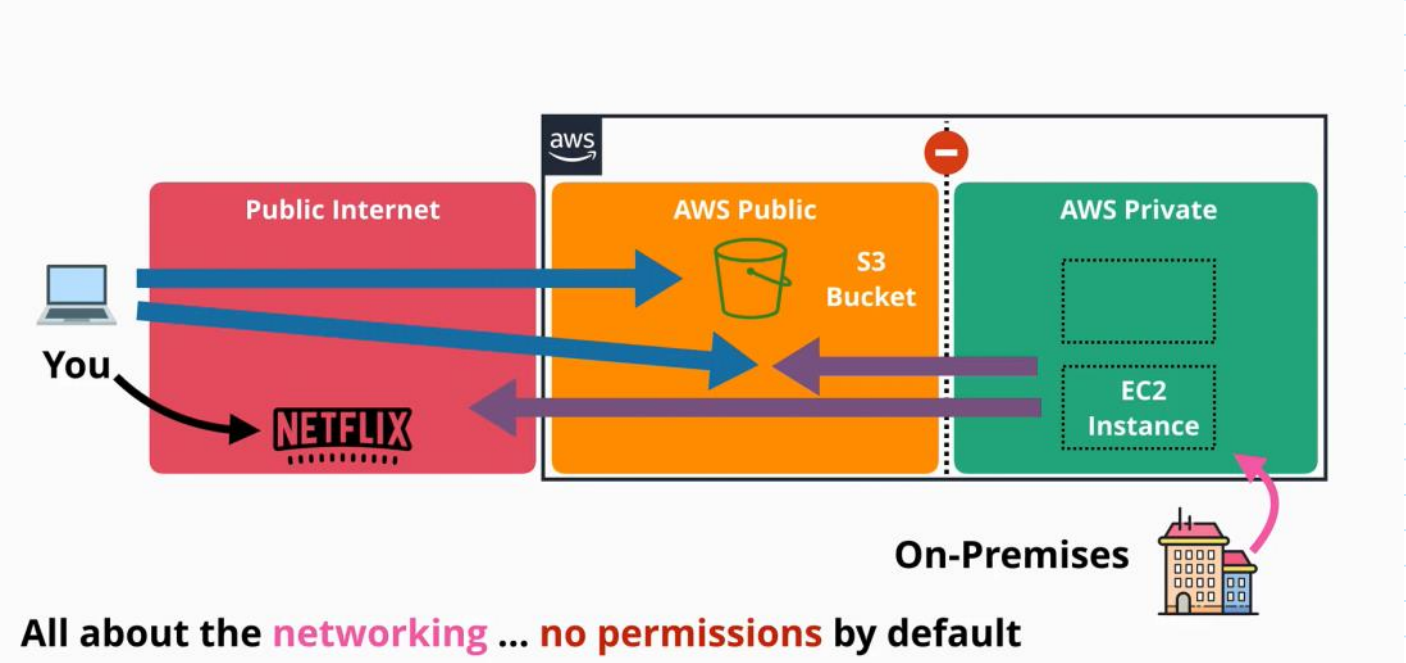
1. Connectivity is required?
2. Permissions granted for the user?

AWS is a **public cloud**, it can be connected over the public internet.

AWS has a zone called the AWS public zone, which is where public services run from, and is connected and can be accessed from the public internet.

The private zone is isolated by default, from the AWS public zone, and from the public internet. But this private zones can be configured to be public, meaning that part of them, is projected into the public zone and then accessed by a public zone.

By default private services can be accessed only from the same private networks or any on-premises networks connected to them.



AWS public service is located in the AWS Public Zone and anyone can connect but permissions are required to access the service.

AWS private service is located in a VPC, accessible from the VPC is located in and accessible from other VPCs or on-premises networks as long as private networking is configured.

AWS global infrastructure

Thursday, June 10, 2021 3:53 PM

AWS is a global cloud platform, which is a collection of smaller groupings of infrastructure connected together by a global high speed network.

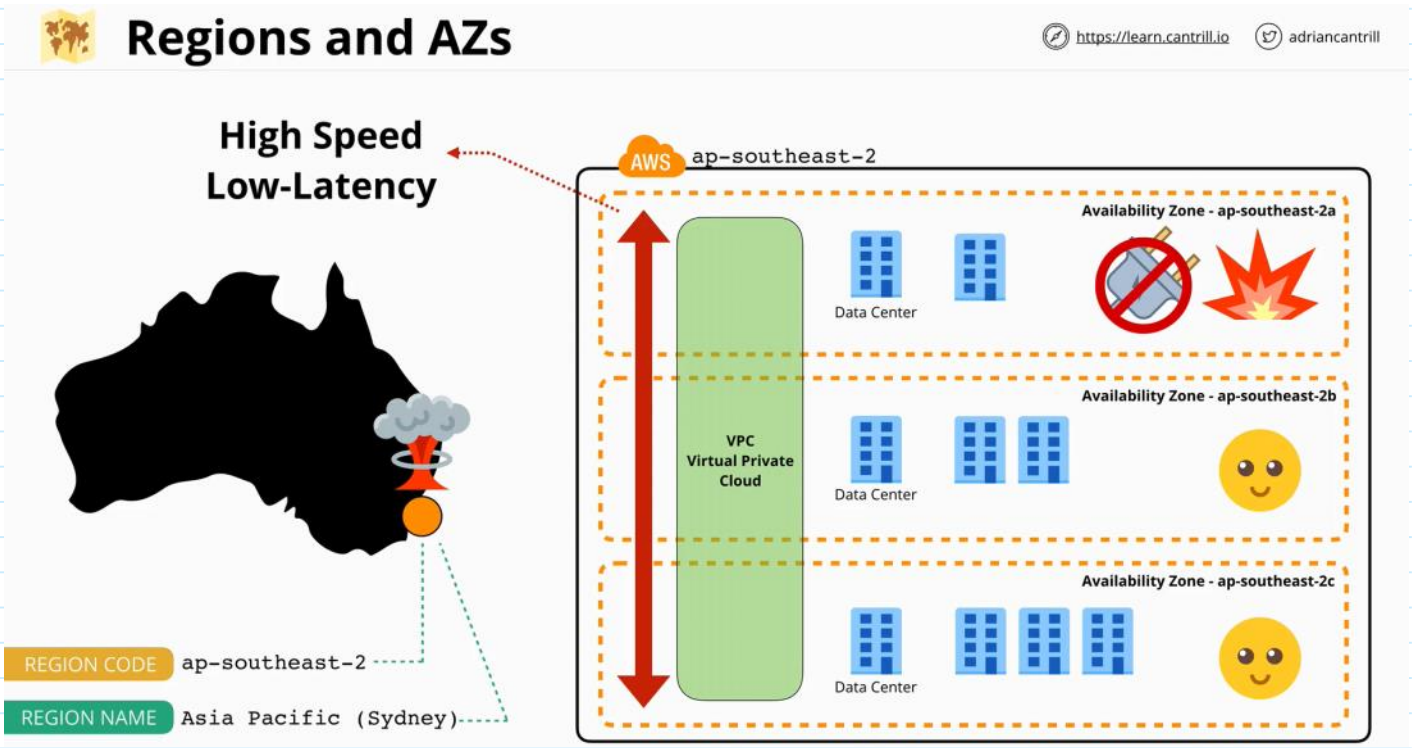
A **region** in AWS context, is an area of the world they had selected that encapsulates a full deployment of AWS infrastructure:

- compute services
- database products
- storage
- AI
- analytics
- etc

Edge locations are smaller than regions, and generally they only have content distribution services, and some types of edge computing.
The edge locations are placed in many more places than regions.
Usually, the further the services are from the clients, the slower the transfer is, and higher the latency.

Usually regions and edge location are used together by solutions architects.

Regions are 100% isolated from each other, meaning that if a disaster happens in one, the others won't be affected.
Regions have geopolitics and governance separation - the client is affected by the laws and regulations of the of the region of where the infrastructure is located.
Data saved in one region, won't leave that region unless configured otherwise.
Regions give location control, which allows the tunning of infrastructure and architecture.



Resilience:

1. **Globally** means that a service operates globally as a single product, and its data is replicated across multiple regions. This means that if a region fails, the services continue to run. (IAM and Route53 are global).
2. **Region** - this are services that operate within a single region, with one set of data per region. They replicate data in multiple **availability zone** within that region.
3. **AZ** -they run from a single availability zone.

Virtual Private Clouds (VPC)

Tuesday, June 15, 2021 3:56 PM

VPC is used to create private networks inside AWS, and is used to connect AWS private networks to the on premises networks when creating a hybrid environment.
VPC is also the service that allows you to connect to other cloud platforms when creating a multi-cloud deployment.

VPC is a virtual network inside AWS.

There are two types of VPC:

1. Default VPC - **maximum** one per region and can be deleted, removed or recreated
    a. created by AWS
    b. they come preconfigured in a very specific way
    c. they are less flexible than custom VPCs
    d. always have the CIDR: **172.31.0.0/16**
    e. **everything placed in the VPC subjects is assigned a public IPv4**
2. Custom VPCs - as many as you want
    a. they can be configured as they need to be
    b. require the configuration, end to end in **detail**
    c. they are **private** by default

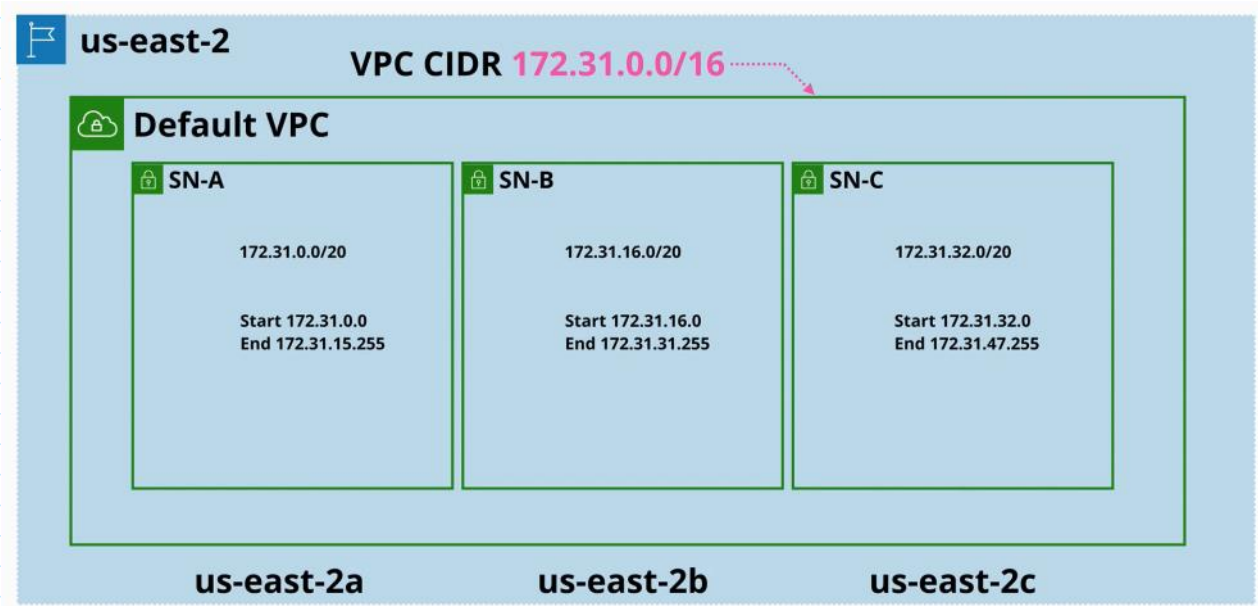
VPCs **cannot communicate** in between them, even if they are in the same AWS account, unless is stated otherwise. Basically, VPCs are by default **entirely private.**

Default VPC

Every VPC is allocated a range of IP addresses, called VPC Cider. Everything inside a VPC, uses the CIDER range of that VPC. Is anything needs to communicate with that VPC (if is allowed) it needs to communicate to that VPC CIDR.

The default VPC gets only one CIDR: **172.31.0.0/16**
This is one of the strengths, is always configured by default in the same way:

- has one subnet in every availability zone in its region
- each subnet uses a part of the VPC's range of IP addresses
- these cannot be the same or overlap as other subnets in the VPC
- this makes the VPCs **resilient**



Custom VPC

Custom VPCs can have multiple CIDR ranges.

Elastic Compute Cloud - EC2

Tuesday, June 15, 2021 6:02 PM

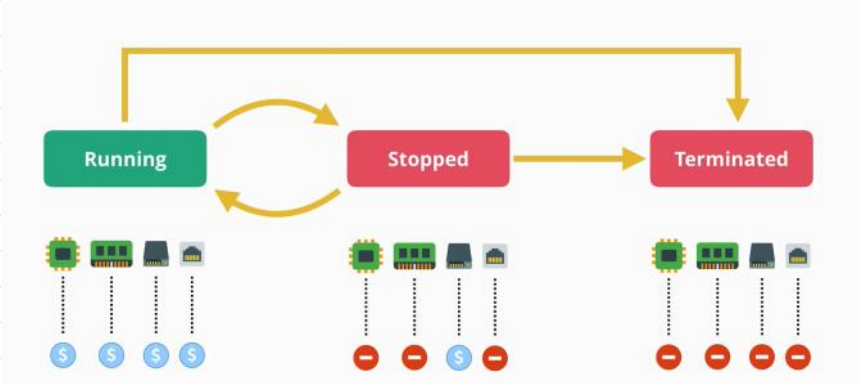
EC2 provides access to virtual machines known as instances.

EC2 is **IAAS** (infrastructure as a service)

- provides access to virtual machines (EC2 instances)
- an instance is an OS configured in a certain way with a certain set of allocated resources
- is a private AWS service by default
- for public access to an EC2 instance than the VPC running needs to support that public access
- is AZ resilient
- the developer managers the OS and everything up from the infrastructure stack
- is an on-demand billing by second or hour
- instance charge are present such as:
  - running the instance
  - storage
  - commercial software
- storage is of 2 types
  - on host storage - where the instance runs
  - EBS Elastic Block Store - network storage made available for the instance

ECS has **few** states

- **running**
- **stopped**
- **terminated** - one time action, that deletes the allocated storage



AMAZON MACHINE IMAGE (AMI)

- **is an image of an EC2 instance**
- is used to create an EC2 instance
- can be created from an EC2 instance
- contains attached permissions that control which accounts can or can't use the AMI
- can be public
- can be private (owner)
  - explicit permissions can be granted from the owner to other users
- contains the root volume (partition) - the driver that boots the OS
- can contains other volumes (other drivers)
- contains block device mapping
  - a configuration which link the volumes that the AMI has, and how they are linked withing the OS

Windows **Linux**

- a configuration which link the volumes that the AMI has, and how they are linked withing the OS

Windows

- **remote desktop control port 3389**

Linux

- **SHH protocol port 22**

What Permissions options does an AMI have?
Public access, Owner Only, Specific AWS accounts

Cloud Formation

Thursday, June 17, 2021 12:30 PM

Cloud formation is a tool that allows you to create, update and delete infrastructure in a AWS account.
Instead of deleting and creating resources manually, a template can be created.
The **CloudFormation** automates infrastructure

A cloud formation template is written in **YAML** or **JSON**.

Templates have:

- a list of **resources**
  - **the only mandatory part**
- description
  - lets the author to add a description
  - it is usually used to describe what the template does, its resources, cost of the template etc.
  - the description needs to follow directly after the AWS template format version!
- metadata
  - controls how different components are presented through the console UI
  - force how the UI presents the template
- parameters
  - fields that prompt the user for more information
- mappings
  - allows to create lookup tables
- conditions
  - actions that occur only if a condition is bet
- outputs
  - once the template is finished it will present outputs of what had been created, updated or deleted

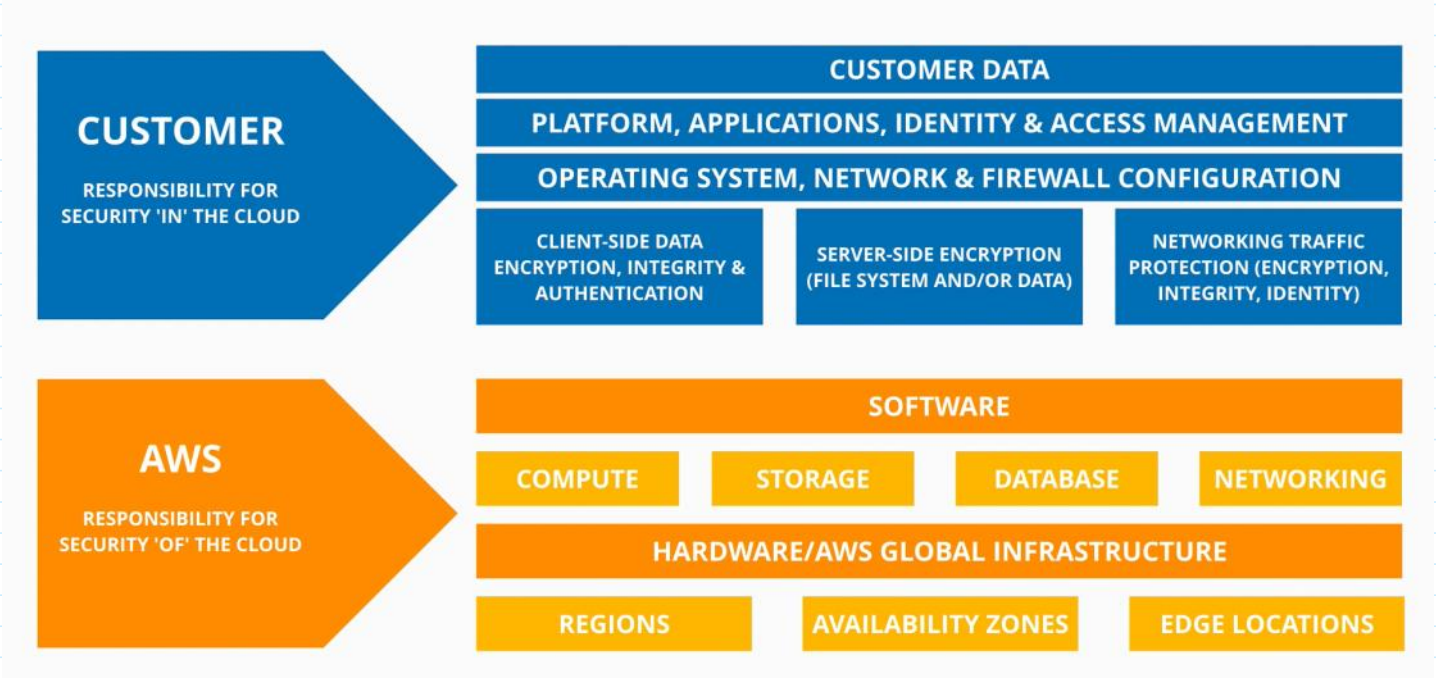
A template contains a logical stack of resources. When a template is executed, the Cloud formation creates a corresponding physical resource.

A CloudFormation Physical resource is a physical resource by creating a CloudFormation stack

Shared Responsibility Model

Tuesday, June 22, 2021 1:59 PM

Shared Responsibility Model



High-Availability vs Fault-Tolerance vs Disaster Recovery

Tuesday, June 22, 2021 3:06 PM

High-Availability (HA)

Maximize uptime

- aims to ensure an **agreed level of operational performance** usually uptime, for a higher than normal period
- -designed to be online as much as possible
- is not about the user experience
- is about maximizing the online time as much as possible
- this required backup (redundant) system to replace of a outbreak system
    - this action has a small outbreak time, and this is okey

Fault-Tolerance (FT)

Operating through failure

- is the property that enables a system to continue operating properly in the event of the failure of some or more components
- if a system has faults, it should continue properly until the faults are fixed
- system are designed to work through failure without disruption

Disaster Recovery (DR)

Keep the crucial and non-replaceable parts of the system safe

- **what we do when high-availability and fault-tolerance DO NO WORK**
- a set of policies, tools and procedures to enable the recovery or the continuation of vital technology infrastructure and systems, following a natural or human-induced disaster
- plan for an action plan when disaster occur
- is a multiple steps processes:
    - what happens before (pre-planning)
        - backup premises
        - virtual backup systems
        - offsite backup storage
        - DR testing
    - what happens after (disaster recovery process)

DNS

Tuesday, June 22, 2021 3:40 PM

DNS is a **discovery service**, it helps users to discover system on the internet.
DNS is resilient, distributed and scalable.

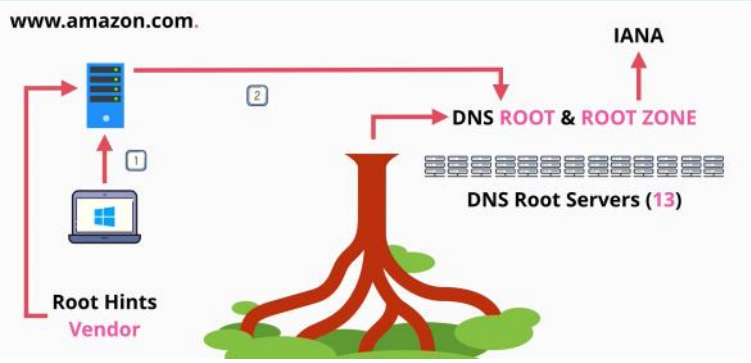
- it translates information machine into human and vice-versa
- translate a domain name to an IP address
- is huge, resilient and hard to distribute
- is a database, distributed and global
- each server has a ZONE file, that links a domain name to the correct IP address
    - that zone file is located anywhere on potentially one or two of millions of DNS name services.
    - the DNS resolver is sitting either on the internet router, or the provider

- **DNS Client** => your laptop, phone, tablet, PC
- **Resolver** => software on your device, or a server which queries DNS on your behalf
- **Zone** => A part of the DNS database (e.g. **amazon.com**)
- **Zonefile** => physical database for a zone
- **Nameserver** => where zonefiles are hosted

DNS is distributed, the data is stored in zone files, and the zone files are stored on name servers globally. The DNS needs a start point.

DNS flow: The DNS resolver needs to locate the correct name server for a given zone, query that name server and retrieve the information it needs, and then pass it back to the DNS client.

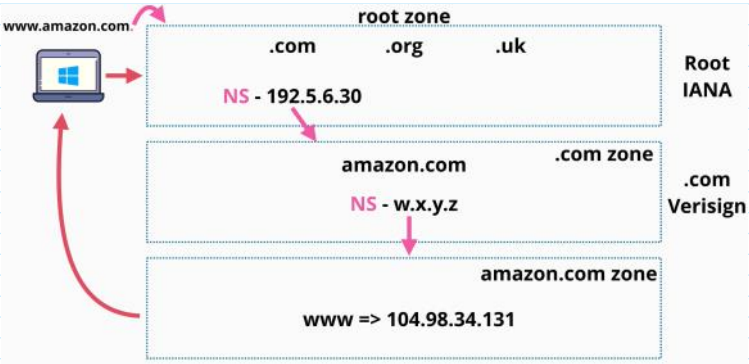
The DNS is structured as a tree. The DNS root (domain name), is stored on 13 special servicer known as **DNS root servers**. This servers are the entry point.



1. The DNS client asks a DNS resolver of given DNS name
2. using the root hints file, the DNS resolver communicated with one or more root servers to access the root zone and begin the process of finding the IP address

A local system (laptop, desktop, etc.). The local system will use a **DNS resolver server** which is located on the local router or internet provider. The vendor of the OS (Microsoft for Windows, OS maintaining group for Linux, etc.) supply the root hints file. The **root hints file** is a pointer to the DNS root servers.

Top level domains are in the root zone (.com, .org, .uk, etc.). This root zone database is managed the IANA. These root zones point to other servers that manages country codes zones.



The image above is how the chain of trust works in DNS.

- **Root Hints** => config points at the root servers IPs and addresses
- **Root Server** => hosts the DNS root zone
- **Root Zone** => points at TLD authoritative servers
- **gTLD** => generic top level domain (.com .org)
- **ccTLD** => country-code top level domain (.uk .eu etc)

Route 53

Wednesday, June 23, 2021 2:53 PM

Route 53 provides to main services:

1. Register domains
2. Host zone files and manage nameservers

Route53 is a global service with a single database, and it is globally resilient.

Register domains

Route 53 allows the registration of domains, and to do that it has relationships with major domain registers.
When registering a domain Route 53 follows the following steps:

1. registers a zone file (dora.org)
2. allocates a name service for that zone
3. puts the zone file in 4 name servers
4. communicates to .org registry to add the new domain into the zone file for the.org top level domain
    a. using name server records

Host zone

(DNS as a service)
This service allows hosting, creating and managing zone files.

Zone files are hosted on 4 managed name servers.
When a hosted zone is created , a number of server are allocated and linked to the hosted zone.
A hosted zone can be public (accessible on the public internet) or private which means that they are linked to one or more VPCs.

DNS Record Types

Wednesday, June 23, 2021 4:02 PM

Which type of organisation maintains the zones for a TLD

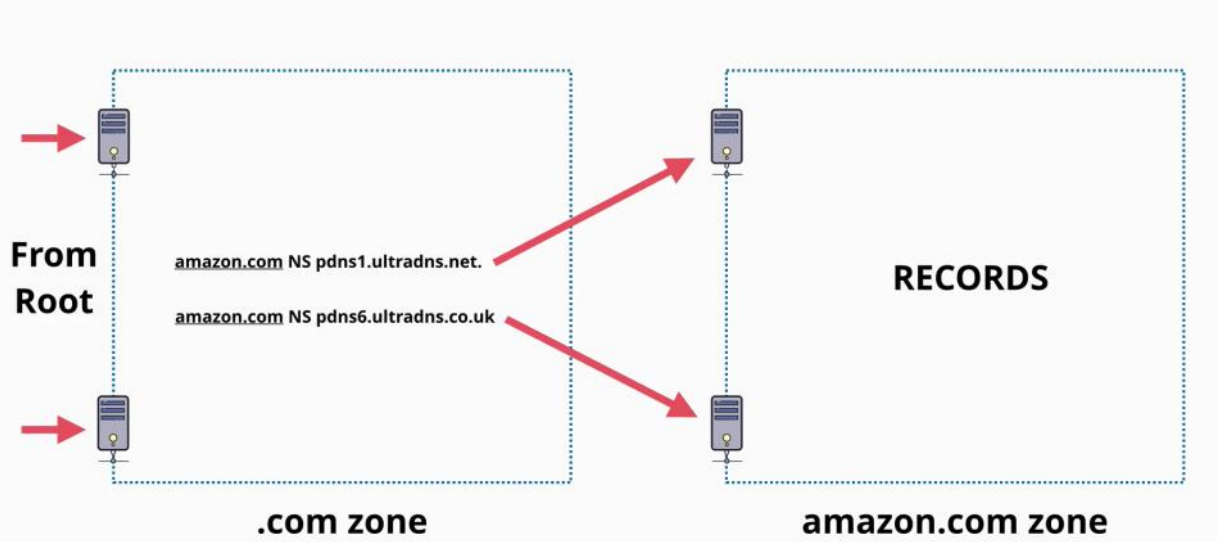
Registry

Which type of organisation has relationships with the .org TLD zone manager allowing domain registration?

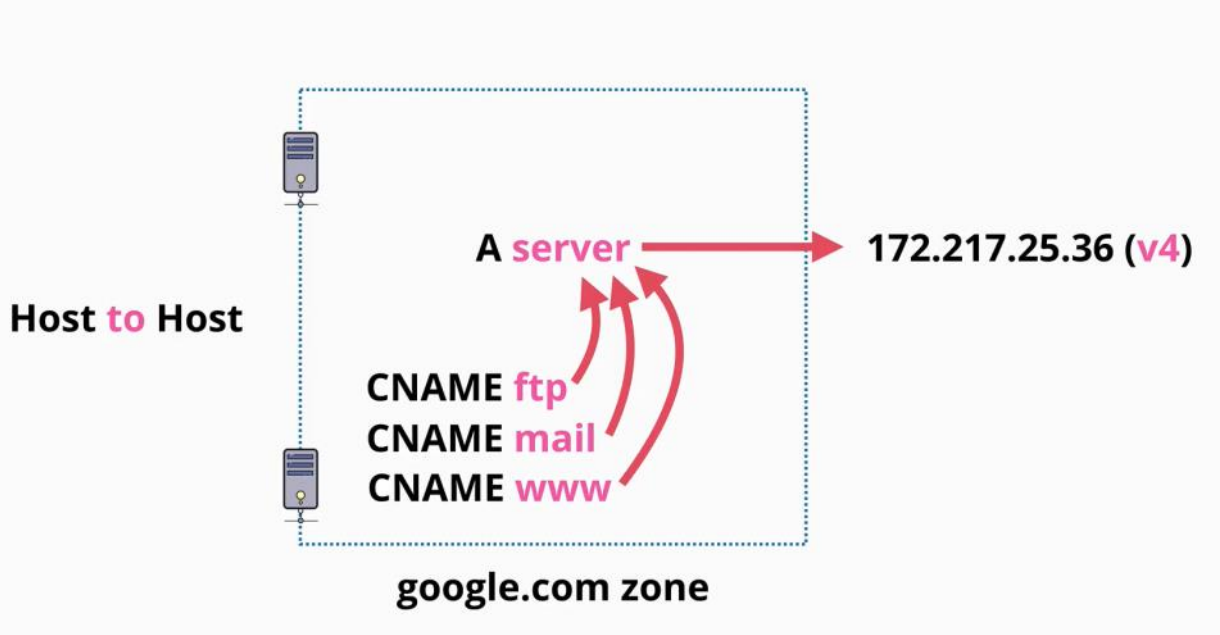
Registrar

Nameserver (NS)

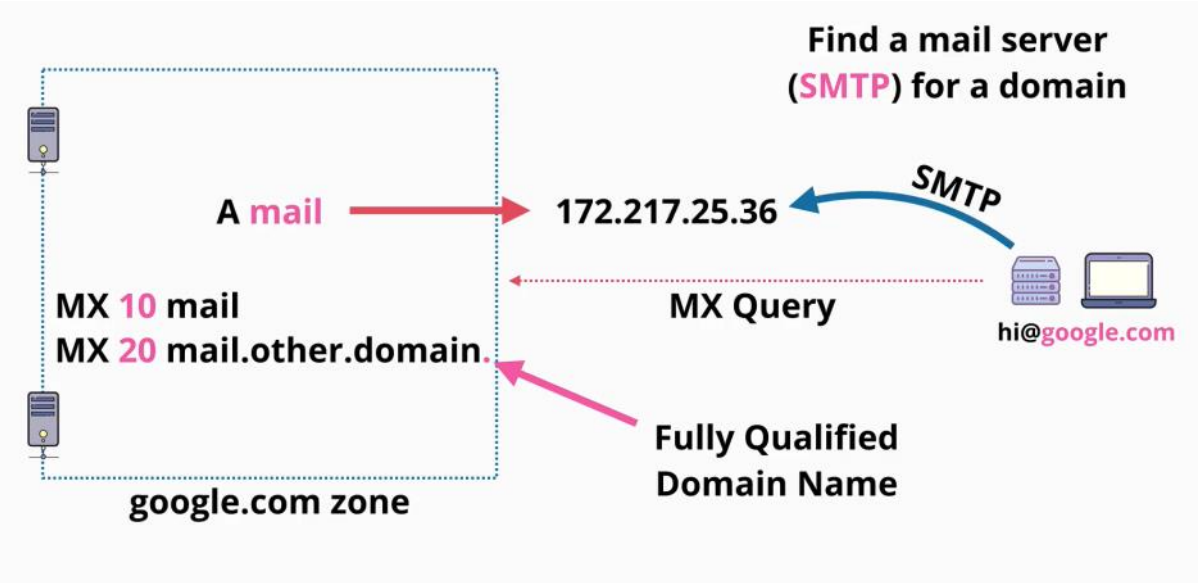
Nameservers are record types, which allow delegation to occur in DNS



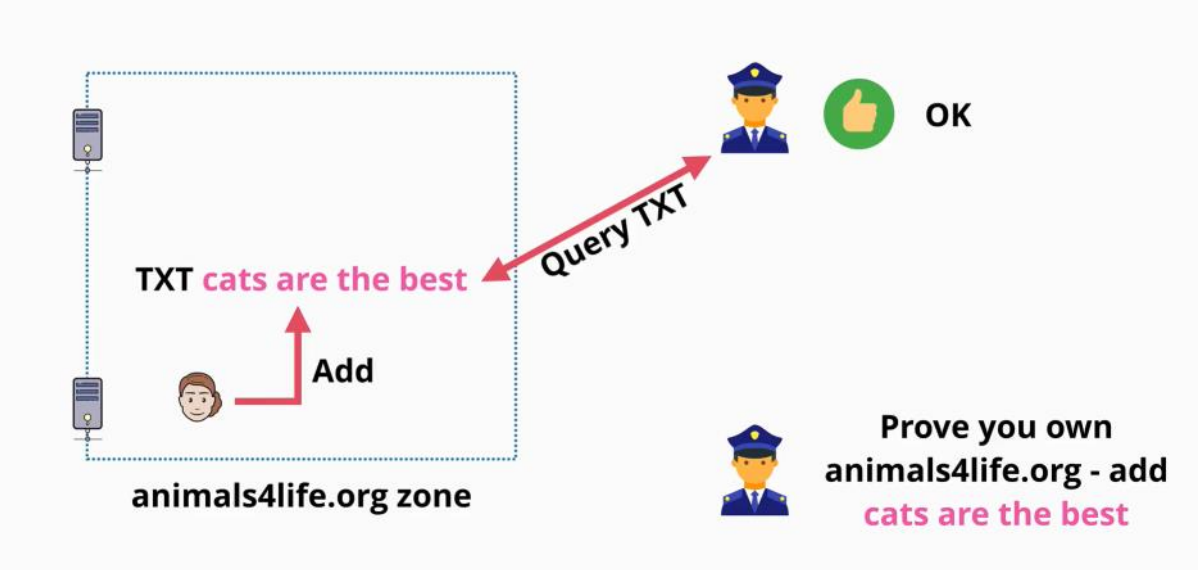
NS record - delegates control of .org to the .org registery
A record - > IPv4 addresses
AAAA record -> IPv6 addresses
CNAME record -> Cnames cannot point directly at an IP address, only at other names.



MX Records - are used as apart of a process (mail)



TXT Records - add arbitrary text to a domain - is used to prove ownership



DNS TTL - Time to live

ATTL value is set to DNS record, is a numeric value in seconds.
Walking the tree takes time, to talk with the root, and all the following levels to get the result needed. It is a lengthy processes.

The TTL specifies for how long the response values can be cached. This is set by the administrator. Once a request had been responded to, and the result is cached on the resolver server, other clients do not need to wait too much for a response.

Authoritative and non-authoritative responses are the same, so in this case TTL matters when things change so that responses are not delayed (if TTL is high).
When things change, the TTL must be changed to a low value, or have a low value of TTL all the time.