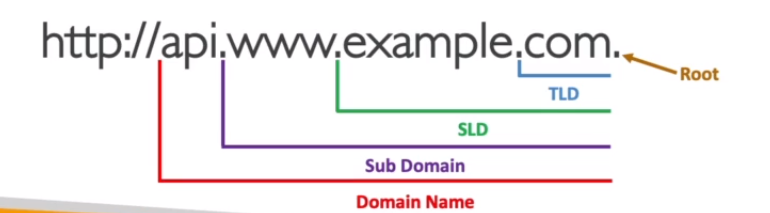
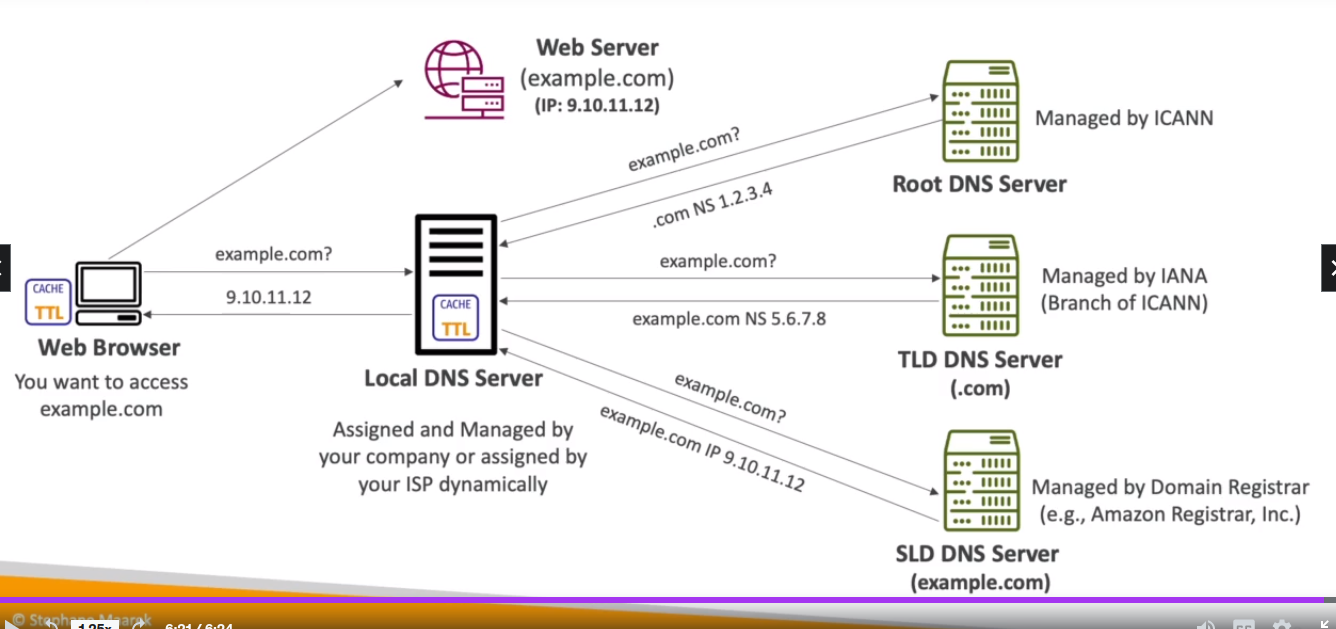
Route 53

udemy

horizontal line

# 8 Jul, 2022

**DNS**

* Translates Human friendly hostnames into machine IP address
* Hierarchical naming structure
  + Top level domain(TLD): .com, .in
  + Second level domain( SLD): amazon.com,google.com
  + 
* **Terminologies**
  + Domain registrar: ROute53, GoDaddy
  + DNS records: A,AAAA,CNAME,NS
  + Zone file: contains DNS records
  + Name server: resolves DNS queries
  + 

**Amazon Route 53**

* FUlly managed DNS
* ROute 53 also a domain registrar
* Checks health of resources
* 100% availability SLA (service level agreement)
* 53 is a reference to the traditional DNS port
* **Records**
  + Decides how you want to route traffic for a domain
  + Components of a record
    - Domain name
    - Record type
    - value(IP)
    - Routing policy
    - TTL ( amt of time the record is cached )
  + Record types
    - A/AAAA/CNAME/NS
    - A- hostname mapped to ipv4
    - AAAA - IPV6
    - CNAME- hostname(1) mapped to hostname(2)
      * The hostname(2) must have an A or AAAA record
      * CNAME cname for a top node of DNS namespace
        + Egs- you cannot create CNAME for example.com but you can create CNAME for [www.example.com](http://www.example.com)
    - NS- Name server for hosted zone
      * Control how traffic is routed for a domain
      * **Hosted zones are containers** that define how to route traffic to a domain and its subdomain
        + Public Hosted zone- can access from internet
        + Private Hosted Zone- can access only from private network.
        + 

HAnds on-

Dig command,

**CNAME vs Alias**

CNAME

* Points a hostname to other hostname
* ONly works for non root domain name
  + So CNAME will work for demo.example.com but wont work for example.com

Alias

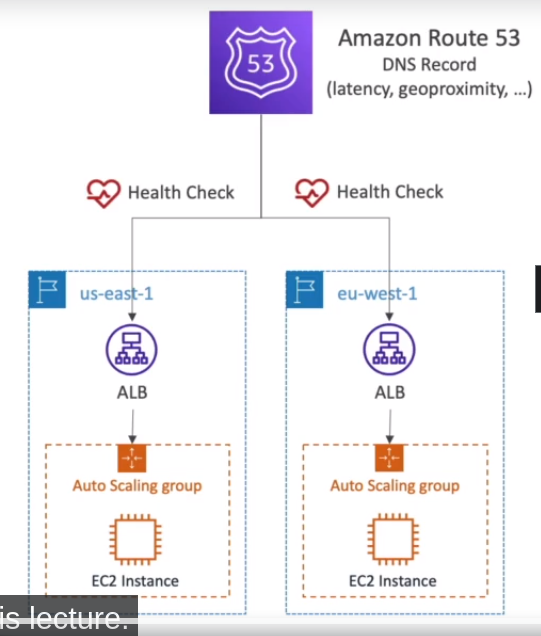
* Points a hostname to an AWS resource
* Specific to AWS route53
* Works for both root and non root
* Natice health check

**Routing Policy**

* Simple
  + Single
  + Multivalued- Multiple IP addresses in the record, client randomly selects one
  + No health checks
* Weighted
  + For multiple ip addresses, assign weights to them to help prioritize
  + Weight can be from 0 to 255
* Latency
  + Multiple ips, chooses ip closest to the client ( that gives minimum latency)
* Failover
  + One ec2 instance is active another is passive in case of a failover
* Geolocation
  + Based on user location. Egs- only people on france can access the endpoint ( maybe for french version of the app)
  + Website localization,restrict content distribution, load balancing etc. 
* Geoproximity
  + Can set bias to a specific region to have more traffic attracted there
* Multi value
  + Different from multivalued in simple routing policy
  + Upto 8 healthy instances
  + Shows only the unhealthy

**Health Checks**

* HTTP health checks are only for public resources
* Types
  + Monitors endpoints
  + Monitors other health checks
  + Monitors cloudwatch alarm
* Have own metric and can check in cloud watch metric



* 15 global health checkers will check the endpoint health
  + 30 sec interval
  + Supports HTTP , HTTPS and TCP
  + If > 18% healthy, route53 considers it to be healthy
* Success result is returned as 2xx and 3xx status codes
* First **5120** bytes of the response decides health checks are pass/fail
* We need to configure the endpoint’s security group to allow incoming requests from route53 health checkers
* **Calculated health checks**
  + Combines results of multiple health check into single health check
  + Can use and,or or not
  + Monitor upto 256 child health checks
* **Private hosted zone**
  + ROUTE 53 health checks are outside the vpc and they cannot access private endpoints
  + Instead create a cloudwatch metric and associate it with cloudwatch alarm and create a health check that checks the alarm