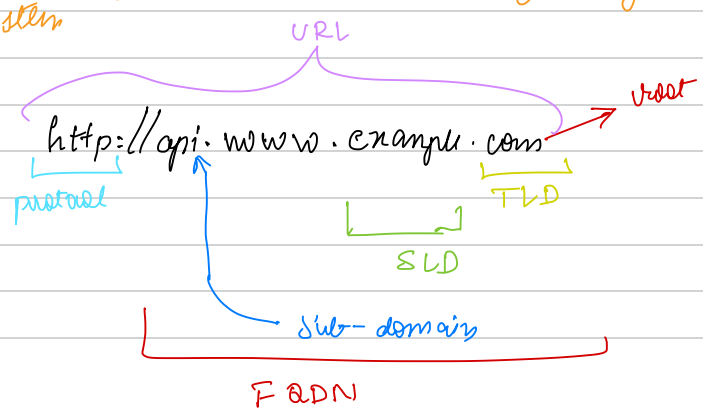
§ Route 53 *

Q What is DNS ?

→ Domain Name system which translates the human friendly hostnames to machine IP addresses

* Domain registrar → To register your domain { Route 53, Go Daddy }
* DNS records → A, AAAA, CNAME
* Zone file → contain DNS records
* IANA → Internet Assigned Number Authority
* Name server → resolves a DNS query.
* ICANN → Internet corporation for Assigned Names & Numbers.
* Top level domain (TLD) → .com, .uk, .in, .gov, .org
* Second level domain (SLD) → google.com
* Fully Qualified Domain Name (FQDN) → the complete or domain name for a specific computer or host. absolute domain name.

belonging to the tree hierarchy of domain name system



Some imp. concepts!

DNS server → A machine that translates domain names into IP addresses.

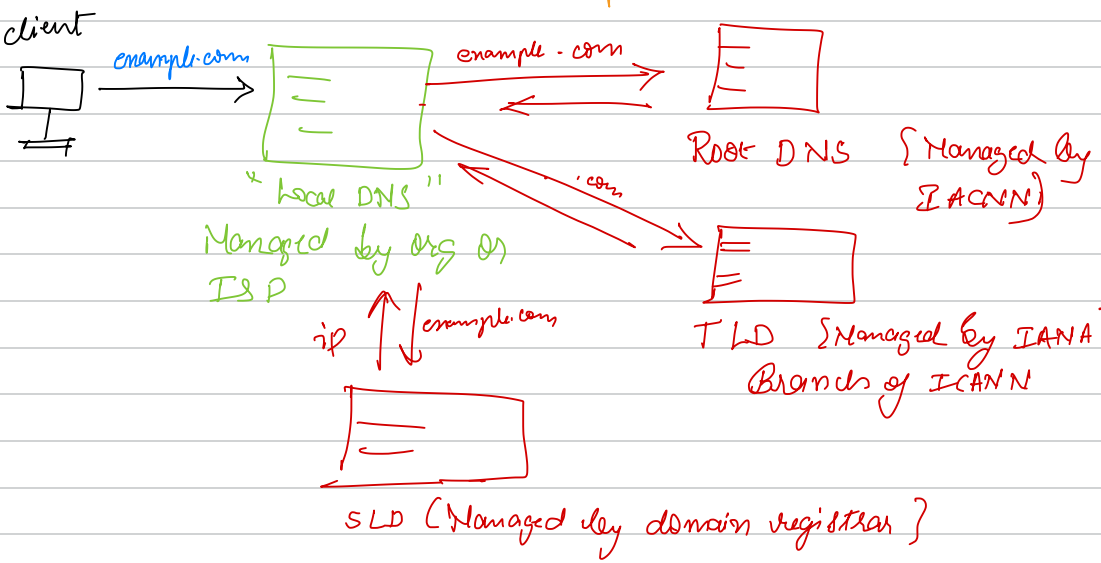
DNS Zone → It is a portion of the DNS namespace that is managed by a specific organization or administration. (it's hosted on a DNS server!)
It starts at a domain within the tree & can also be extend down into subdomains so the multiple subdomains can be managed.

DNS zone file → A plain text file stored in DNS servers that contains an actual representation of the zone & contains all the records for every domain within the zone. must start with a Start of Authority (SOA) record which contain important info including contacts of zone admin

Reverse look up zone → contains mapping from an IP address to the host (for troubleshooting, spam filtering & bot detection)

Zone apex or domain apex → records where the record name is the same as zone's domain name.

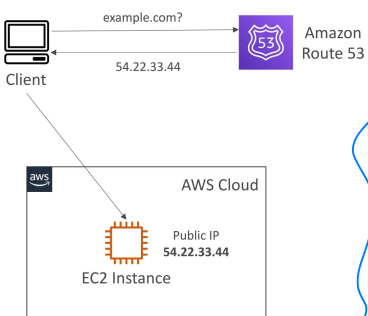
Web servers *
public ip = 9.10 & 11.12



Amazon Route 53

- A highly available, scalable, fully managed and *Authoritative* DNS
  - Authoritative = the customer (you) can update the DNS records
- Route 53 is also a Domain Registrar
- Ability to check the health of your resources
- The only AWS service which provides 100% availability SLA
- Why Route 53? 53 is a reference to the traditional DNS port

you control our DNS



SLA → Service - Level Agreement
It is a contract b/w a service provider & its customers that documents what services the provider will furnish

Route 53 – Records

- How you want to route traffic for a domain
- Each record contains:
  - **Domain/subdomain Name** – e.g., example.com
  - **Record Type** – e.g., A or AAAA
  - **Value** – e.g., 12.34.56.78
  - **Routing Policy** – how Route 53 responds to queries
  - **TTL** – amount of time the record cached at DNS Resolvers
- Route 53 supports the following DNS record types:
  - (must know) A / AAAA / CNAME / NS
  - (advanced) CAA / DS / MX / NAPTR / PTR / SOA / TXT / SPF / SRV

Record types

1. A → hostname to IPv5
2. AAAA → hostname to IPv6
3. CNAME → hostname to hostname
   - target is a domain name with A or AAAA record.
   - Can't create a CNAME record for the top node of a DNS namespace (Zone Apex)
     - → www.example.com ✓
     - example.com ✗
4. NS → Name servers for the hosted zone (they are DNS names or IP addr. of the servers that can respond to the DNS queries for the hosted zone)

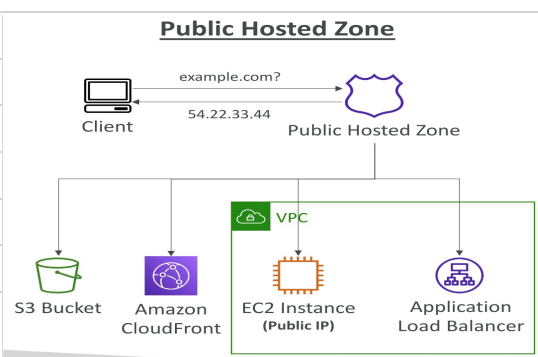
Q Hosted zone?

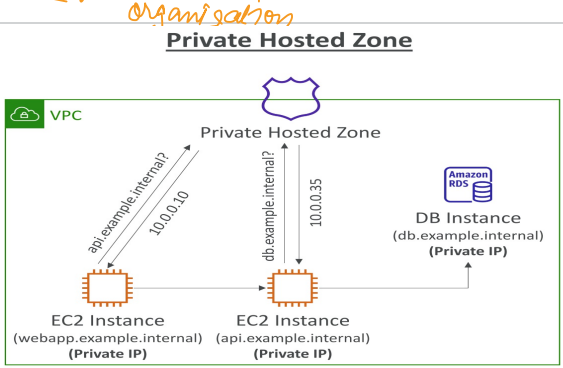
→ A container for records that defines how to route traffic to a domain & its subdomain.

Both of them contain records on...

(for public domain)
Public hosted zone
→ how to route traffic on the internet

Private hosted zone (in a VPC)
→ how you route traffic within one or more VPC.
(private DNS within an organisation)





After creating a public domain > hosted zones > create record

{ record name, type of record, value, TTL & routing policy }

A record is created, now new record when typed in URL will take us to the ip address given by the value!

but since we put some garbage value so the website won't open :)

open cloudshell / powershell, or bash

for DNS lookups

$ nslookup } won't work so
$ dig } sudo yum install -y bind-utils

$ nslookup test.adithya.com → record name

this will give the DNS resolved ip address of that URL

Better $ dig test.adithya.com dig cmd shows the record type & TTL

→ now create an instance, no key pair for this instance connect] load the user data script.

user-data.sh

```
#!/bin/bash
yum update -y
yum install -y httpd
systemctl start httpd
systemctl enable httpd
# updated script to make it work with Amazon Linux 2023
CHECK_IMDSV1_ENABLED=$(curl -s -o /dev/null -w "%{http_code}" http://169.254.169.254/latest/meta-data/)
if [[ "$CHECK_IMDSV1_ENABLED" -eq 200 ]]
then
    EC2_AVAIL_ZONE="$(curl -s http://169.254.169.254/latest/meta-data/placement/availability-zone)"
else
    EC2_AVAIL_ZONE="$(TOKEN=`curl -s -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"`
&& curl -s -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/placement/availability-zone)"
fi
echo "<h1>Hello world from $(hostname -f) in AZ $EC2_AVAIL_ZONE </h1>" > /var/www/html/index.html
```

* Create three instances in different regions with some user data.

* Now launch a ALB in one of the region which we created the instance, internet facing IPv4 map in all subnets.
Select the security groups.

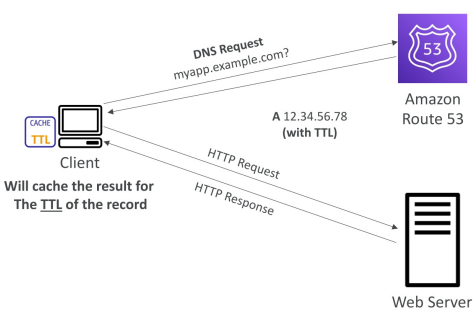
→ Create a new target group based on instance & register instances in that region to the target group.
& create the load balancer

→ After the load balancer is created we will have to link / add its DNS into our Route 53 records.

Route 53 – Records TTL (Time To Live)

- High TTL – e.g., 24 hr
  - Less traffic on Route 53
  - Possibly outdated records
- Low TTL – e.g., 60 sec.
  - More traffic on Route 53 ($$)
  - Records are outdated for less time
  - Easy to change records

- Except for Alias records, TTL is mandatory for each DNS record



The answer of DNS query is cached in client's computer!

☐	stephanetheteacher.com	NS
☐	stephanetheteacher.com	SOA
☐	test.stephanetheteacher.com	A

NS → Name of hosted zone

SOA → Start of Authority AKA the domain

A → A record in the domain

subdomain

(2mins)

TTL set करके फिर से उसके domain search किया तो we open up that instance but a 2min time will come in our system which caches the DNS record so even if we change the record in Route53 we will still get the same result

we can check the TTL with dig cmd!

CNAME vs Alias

- AWS Resources (Load Balancer, CloudFront...) expose an AWS hostname:
  - lb1-1234.us-east-2.elb.amazonaws.com and you want myapp.mydomain.com
- CNAME:
  - Points a hostname to any other hostname. (app.mydomain.com => blabla.anything.com)
  - ONLY FOR NON ROOT DOMAIN (aka. something.mydomain.com)

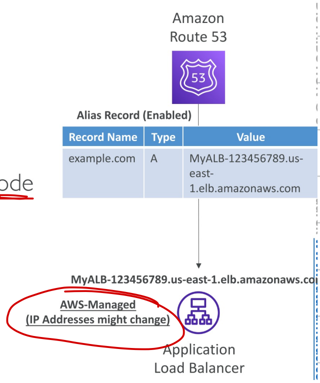
  → cannot be used for zone apex
- Alias:
  - Points a hostname to an AWS Resource (app.mydomain.com => blabla.amazonaws.com)
  - Works for ROOT DOMAIN and NON ROOT DOMAIN (aka mydomain.com)
  - Free of charge
  - Native health check

Route 53 – Alias Records

→ has automatic health checks

- Maps a hostname to an AWS resource
- An extension to DNS functionality
- Automatically recognizes changes in the resource's IP addresses
- Unlike CNAME, it can be used for the top node of a DNS namespace (Zone Apex), e.g.: example.com

  → can be used for zone Apex
- Alias Record is always of type A/AAAA for AWS resources (IPv4 / IPv6)
- You can't set the TTL



→ Also has health checks

Route 53 – Alias Records Targets

- Elastic Load Balancers
- CloudFront Distributions
- API Gateway
- Elastic Beanstalk environments
- S3 Websites
- VPC Interface Endpoints
- Global Accelerator accelerator
- Route 53 record in the same hosted zone

- You cannot set an ALIAS record for an EC2 DNS name

→ Route the DNS queries

Route 53 – Routing Policies

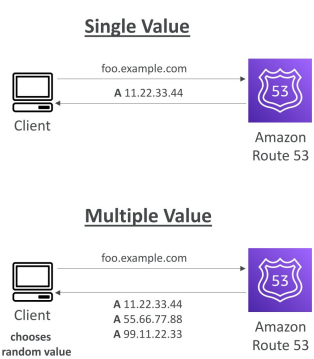
- Define how Route 53 responds to DNS queries
- Don't get confused by the word *"Routing"*
  - It's not the same as Load balancer routing which routes the traffic
  - DNS does not route any traffic, it only responds to the DNS queries
- Route 53 Supports the following Routing Policies
  - Simple
  - Weighted
  - Failover
  - Latency based
  - Geolocation
  - Multi-Value Answer
  - Geoproximity (using Route 53 Traffic Flow feature)

Routing Policies – Simple

- Typically, route traffic to a single resource
- Can specify multiple values in the same record
- If multiple values are returned, a random one is chosen by the client
- When Alias enabled, specify only one AWS resource
- Can't be associated with Health Checks

→ when mapping a domain to the ip there can be multiple values of ip addresses meaning same name for multiple ip's

→ But if "Alias" is enabled only ONE AWS resource can be specified

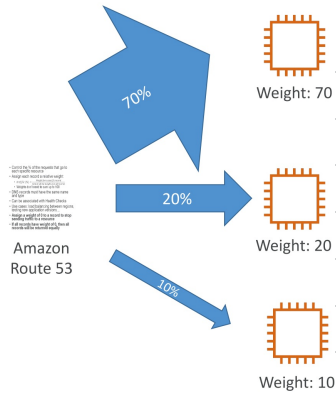


Routing Policies – Weighted

- Control the % of the requests that go to each specific resource
- Assign each record a relative weight:
  - $traffic\ (\%) = \frac{Weight\ for\ a\ specific\ record}{Sum\ of\ all\ the\ weights\ for\ all\ records}$
  - Weights don't need to sum up to 100
- DNS records must have the same name and type
- Can be associated with Health Checks
- Use cases: load balancing between regions, testing new application versions…
- Assign a weight of 0 to a record to stop sending traffic to a resource
- If all records have weight of 0, then all records will be returned equally

*0 weight => no request

*weights to ALL => all records returned equally.



Routing Policies – Latency-based

- Redirect to the resource that has the least latency close to us
- Super helpful when latency for users is a priority
- Latency is based on traffic between users and AWS Regions
- Germany users may be directed to the US (if that's the lowest latency)
- Can be associated with Health Checks (has a failover capability)

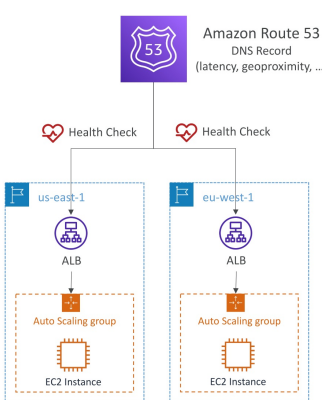
जहाँ भी less latency होगी उसी को पकड़ेगा !

Route 53 – Health Checks

- HTTP Health Checks are only for public resources
- Health Check => Automated DNS Failover:
  1. Health checks that monitor an endpoint (application, server, other AWS resource)
  2. Health checks that monitor other health checks (Calculated Health Checks)
  3. Health checks that monitor CloudWatch Alarms (full control !!) – e.g., throttles of DynamoDB, alarms on RDS, custom metrics, … (helpful for private resources)
- Health Checks are integrated with CW metrics

public resources

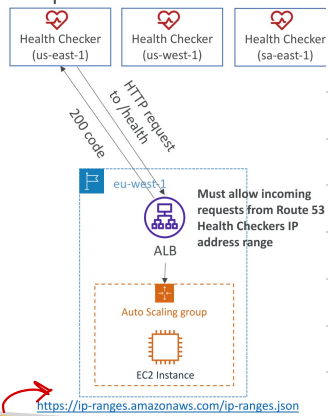
private resources or public (if needed)



Failure threshold → how many times should the health check fail in order to consider it as unhealthy.

Health Checks – Monitor an Endpoint

- About 15 global health checkers will check the endpoint health
  - Healthy/Unhealthy Threshold – 3 (default)
  - Interval – 30 sec (can set to 10 sec – higher cost)
  - Supported protocol: HTTP, HTTPS and TCP
  - If > 18% of health checkers report the endpoint is healthy, Route 53 considers it Healthy. Otherwise, it's Unhealthy
  - Ability to choose which locations you want Route 53 to use
- Health Checks pass only when the endpoint responds with the 2xx and 3xx status codes
- Health Checks can be setup to pass / fail based on the text in the first 5120 bytes of the response
- Configure you router/firewall to allow incoming requests from Route 53 Health Checkers



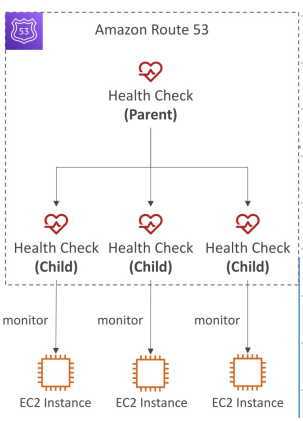
→ All the health checkers will send requests using http, https on TCP & out of all 15 global health checkers if more than 18% of them receive a status OK (200) then the endpoint is considered healthy.

so we must allow these ip ranges in the inbound rules for the endpoint i.e. will constantly fail health checks.

ip's of health checkers.

Route 53 – Calculated Health Checks

- Combine the results of multiple Health Checks into a single Health Check
- You can use OR, AND, or NOT
- Can monitor up to 256 Child Health Checks
- Specify how many of the health checks need to pass to make the parent pass
- Usage: perform maintenance to your website without causing all health checks to fail



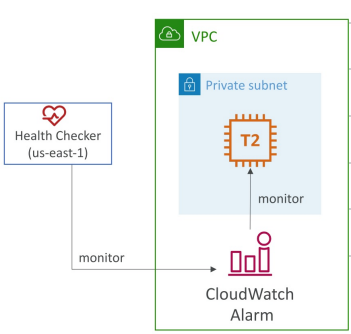
Health Checks – Private Hosted Zones

→ If the Alarm in alarming state then unhealthy.

- Route 53 health checkers are outside the VPC
- They can't access private endpoints (private VPC or on-premises resource)

But instead!

- You can create a CloudWatch Metric and associate a CloudWatch Alarm, then create a Health Check that checks the alarm itself



Go to Route 53 > Health checks > create health checks

V

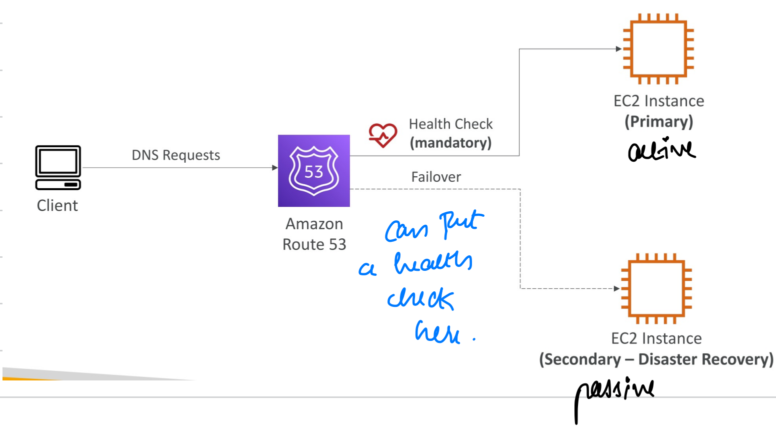
health checks for all the instances.

type →
- Endpoint
- calculated
- cloudwatch Alarm state

path → might need to specify if the endpoint has a path which returns health say /health

Adv. conf:
- Failure thresholds
- string matching → first 5120 By mei ढूँढ लेगा है तो
- latency graph
- invert health checks
- health checker regions
- do you want to create an alarm if health check fails.

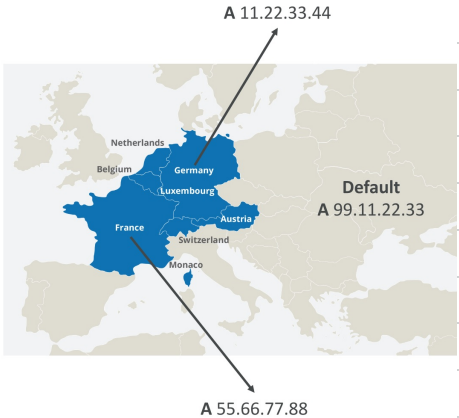
Routing Policies – Failover (Active-Passive)



→ create a record with routing policy of failover.

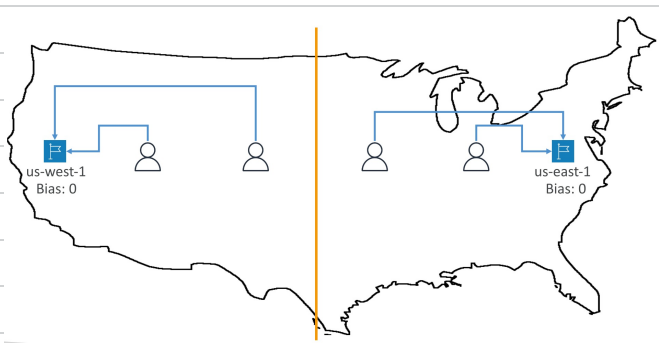
Routing Policies – Geolocation

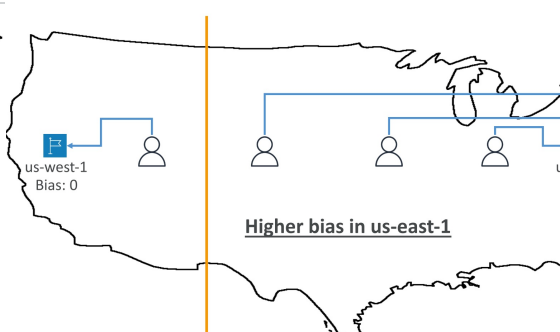
- Different from Latency-based!
- This routing is based on user location
- Specify location by Continent, Country or by US State (if there's overlapping, most precise location selected)
- Should create a "Default" record (in case there's no match on location)
- Use cases: website localization, restrict content distribution, load balancing, …
- Can be associated with Health Checks



Routing Policies – Geoproximity

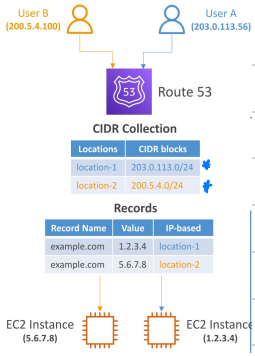
- Route traffic to your resources based on the geographic location of users and resources
- Ability to shift more traffic to resources based on the defined bias
- To change the size of the geographic region, specify bias values:
  - To expand (1 to 99) – more traffic to the resource
  - To shrink (-1 to -99) – less traffic to the resource
- Resources can be:
  - AWS resources (specify AWS region)
  - Non-AWS resources (specify Latitude and Longitude)
- You must use Route 53 Traffic Flow to use this feature





Routing Policies – IP-based Routing

- Routing is based on clients' IP addresses
- You provide a list of CIDRs for your clients and the corresponding endpoints/locations (user-IP-to-endpoint mappings)
- Use cases: Optimize performance, reduce network costs…
- Example: route end users from a particular ISP to a specific endpoint



Routing Policies – Multi-Value

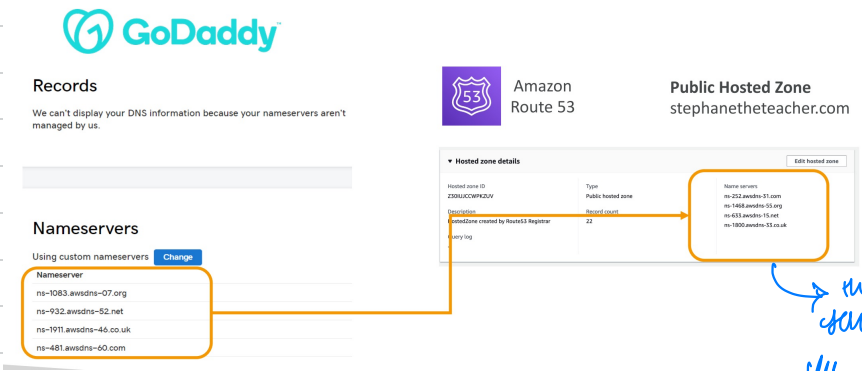
- Use when routing traffic to multiple resources
- Route 53 return multiple values/resources
- Can be associated with Health Checks (return only values for healthy resources)
- Up to 8 healthy records are returned for each Multi-Value query
- Multi-Value is not a substitute for having an ELB

Name	Type	Value	TTL	Set ID	Health Check
www.example.com	A Record	192.0.2.2	60	Web1	A
www.example.com	A Record	198.51.100.2	60	Web2	B
www.example.com	A Record	203.0.113.2	60	Web3	C

Domain Registar vs. DNS Service

- You buy or register your domain name with a Domain Registrar typically by paying annual charges (e.g., GoDaddy, Amazon Registrar Inc., …)
- The Domain Registrar usually provides you with a DNS service to manage your DNS records
- But you can use another DNS service to manage your DNS records
- Example: purchase the domain from GoDaddy and use Route 53 to manage your DNS records

GoDaddy as Registrar & Route 53 as DNS Service



these name servers are to be changed on GoDaddy website

→ Create a public hosted zone in route 53

→ So for every query the GoDaddy will have name servers pointing to the DNS server @ route 53 that will contain all the records

3rd Party Registrar with Amazon Route 53

- If you buy your domain on a 3rd party registrar, you can still use Route 53 as the DNS Service provider

1. Create a Hosted Zone in Route 53
2. Update NS Records on 3rd party website to use Route 53 Name Servers

- Domain Registrar != DNS Service
- But every Domain Registrar usually comes with some DNS features

Yearly Domain name charges $12 per month