**AWS Classroom training**

**Date: 06/12/2018**

**Day 1:**

**Introduction to cloud computing:**

**Datacentre:**

**Capex: Capital expenses**

Traditional CAPEX model – Buy the dedicated HW and depreciate it over a period of time.

CapEx refers to a Capital expenditure while OpEx refers to an Operational expenditure. Capital expenditure is incurred when a business acquires assets that could be beneficial beyond the current tax year. For instance, it might buy brand new equipment or buildings. Also, it could upgrade an existing asset to boost its value beyond the current tax year. CapEx is also known as a Capital expense.

**Why they go for migration?**

After 4 years of warranty, need to renew by AMC contract for maintenance.

Instead of that, companies go for new hardware with new features which covers again 4 years of warranty.

**OPEX Model** – Operating expenses – Use a shared cloud infrastructure and pay as one uses it.

Operational expenditure consists of those expenses that a business incurs to run smoothly every single day. They are the costs that a business incurs while in the process of turning its inventory into an end product. Hence, depreciation of fixed assets that are used in the production process is considered OpEx expenditure. OpEx is also known as an operating expenditure, revenue expenditure or an operating expense.

OpEx – Operating expenses are fully deducted in the accounting period they were incurred. All funds spent when converting inventory into throughput falls under OpEx. This includes employee wages, repair and maintenance of equipment, rental fees, and utility bills and so on. If a business invests in real estate, these spending is approved as CapEx budget and the expense is grouped under CapEx. However, all the costs incurred when managing such an income generating building falls under OpEx.

Operational expenses, maintenance.

## CapEx Summary

* Purchase of fixed assets.
* Preparation of the purchased asset so it can be appropriate for business use.
* Fixing of asset’s problems,
* Restoration of an asset’s value through upgrading
* Adapting a machine to a different use

Operating Expenditures Summary

* License fees
* Advertising costs
* Legal and attorney fees
* Telephone and power overheads
* Insurance fees
* Property management costs
* Property taxation expenses
* Vehicle fuel and repair costs
* Leasing commissions
* Salary and wages
* Raw materials and supplies
* Office overheads

Why we go for cloud?

**Benefits of cloud:**

* Pay for what you use.
* Cost reduction by not managing own datacentre
* CPU/memory seems to be shared across servers in cloud.
* On Demand Self Service – service based on the request.
* Broad Network Access – access wherever
* Resource Pooling - CPU/memory seems to be shared across different AZ(traditionally datacentre)
* Rapid Elasticity – Autoscaling
* Measured Service – cloudwatch,

**Datacentre types:**

**Example: In which datacentre you host for websites, icici & cloudnloud**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **DC Types** | **Priority** | **Availability** | | **Application classification** |
| Tier 1 | NA | Standalone |  | BCV4 |
| Tier 2 | Normal | DEV/TEST/UAT |  | BCV3 |
| Tier 3 | Critical | BCP/DR |  | BCV2 |
| Tier 4 | Most Critical | High Redundancy | 99.999% availability | BCV1 |
|  |  |  |  |  |
| Tier 4 | High redundancy for all devices |  |  | **BCV - Business continuity value** |
| Tier 3 | Mainly for DR/BCP - critical |  |  |  |

**Traditional v/s Cloud:**

Pay for what you use.

Cost reduction by not managing own datacentre

**Cloud types:**

**On-premises:**

Example: vmware

**Iaas**

We need to manage the OS, resource will be provided by cloud vendor.

We need to manage OS. Middleware, runtime, date & Applications.

**Paas**

Only coding part need to be taken care, infra environment will be provided by the vendor.

Runtime environment will be provided.

Charged base on size used.

**Clarify with Vijay**

**Saas**

We will use the product provided by the vendor.

Charged based on per user per month.

**Anycast V/s GeoDNS**

**Anycast DNS** traditional service, it reaches the respective server where it is hosted.

Earlier, when we type google.com, it resolves to google.co.in.

Now in cloud **GEO DNS** is used – Geography DNS.

It loads the respective DNS server geographically, performance will be better., it reroutes to the location specific DNS server.

Goto whatismyip in your browser, it shows your location and ISP provider.

**RPO vs RTO:**

**Functional requirements:** client defines their requirement.

**Non- Functional requirements:** Technically we define what we do for the requirement

**RPO:** Recovery point objective.

**Example:**

Taking back at 5 pm on daily basis, next day morning server is crashed, so data lost from 5pm to next day.

We need to pay penalty as per the agreement.

To avoid this, we need to technically plan the backup strategy.

Snap shot technology can be used for backup

Every 4 hours , incremental backup can be scheduled.

**RTO: Recovery time objective**

Consider the same example as above, next day website is getting down.

We need to do maintenance to bring back the service.

For this we need to plan for cluster to have redundancy and make sure we meet the objective of max 4 hours accepted downtime.

**If data is critical, go for RPO**

**If availability is critical, go for RTO**

**AWS:**

**Availability Zones: (Traditionally Datacentre)**

(Location) Region Examples: Nvirginia, Canada, Sydney, OHIO

Multiple Zones within the regions to have high availability.

Login to AWS console click EC2 service , it will list the regions and AZ.

If we take N.california region, **region name is us-west-1**

Availability zones are us-west-1a, us-west-1b

**Practicals: Instance creation**

* 1. Select N.virginia
  2. Launch Instance
  3. Choose an Amazon Machine Image (AMI)
  4. Select redhat (i386& i686 32 bit, x86\_64 – 64 bit)
  5. Choose instant type.

Based on the usage , need to select the instance type.

Select T2.micro for free tier point of view.

For real time, we need analyse the usage and choose the instance type.

We need to discuss with developer, load testing base line report for the particular application.

Baseline report means 

|  |  |  |
| --- | --- | --- |
|  | **Baseline reporting** | |
| **Requests** | **CPU** | **Memory - GB** |
| 200 | 1 | 1 |
| 400 | 2 | 2 |
| 600 | 3 | 3 |

Mercury Quality centre tool is used for load testing

**Example:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Citibank.com page size | **Requests** | **CPU** | **Memory - GB** | Usage |
| 4 kb | 200 | 1 | 1 | 800kb |
| 4 kb | 400 | 2 | 2 | 1600 kb |
| 4 kb | 600 | 3 | 3 | 2400 kb |

Based on the usage select the number of instance which satisfies the resource requirement.

Always choose +20% with the expected resource requirement.

We need internal load test and AWS load test report before finalizing instance type.

* 1. Add storage

What is iops-> how much datainput and dataoutput

Select addstorage.

* 1. Then select configure security group

Spoofing – by default functionality in firewall.

Incoming traffic will have session id created in firewall, outgoing traffic will happen only if session id is valid.

TCP vs UDP

TCP – secured channel , 3ways handshake happens.

For shorter distance TCP is preferred, TCP will wait for acknowledgement for every packets, udp doesnot wait.

* 1. Configure SG,

Select Anywhere always.

* 1. Download Keypair
  2. Launch instance
  3. 2/2 checks ready means, instance is completely ready

½ means , with our configuration., instance is getting populated

* 1. Download putty and puttygen from the below link.

<https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html>

open puttygen, import give the pem key downloaded convert it to ppk file,

Use it for login to the created instance.

Create multiple ppk files with the pem key and handover repective ppk file to the user.

**Note:** Public ip – dynamic, elastic ip – static ip

* 1. Create another instance with public ip disabled, once instance launched, create elastic IP and associate it to the instance.

Once tested, deassociate and release,orelese it will be charged.

**Multiple instances Launch:**

**Spot instance: (Bidding)**

Mainly used for DEV environment – least priority servers.

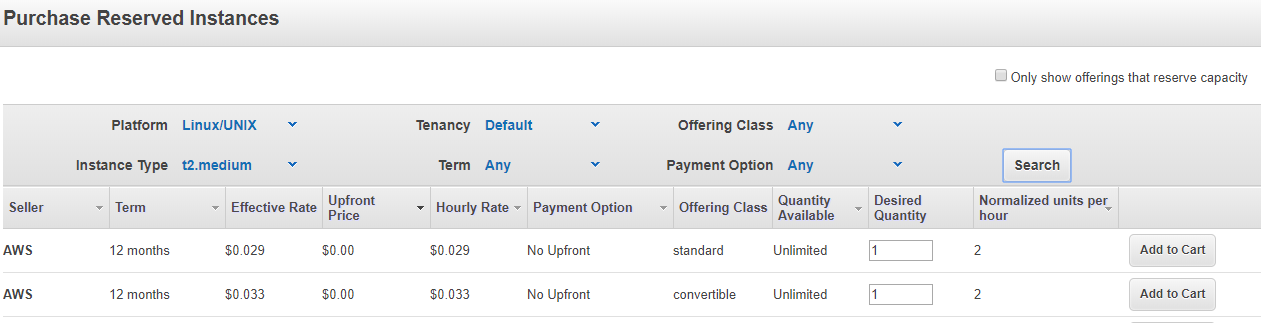
Used for cost saving.

Used for testing, Vulnerability assessment servers.

**Reserved instance: (Guaranteed business)**

PROD servers can be selected under this category based to the criticality and availability need.

In real time launch instance will not be used, reserved instance option is used.



If we select convertible class, we can change the instance type during the tenure.

**Note:** Availability zone should be selected as any, check the “only show offerings that reserve capacity” box and set availability zone as any.

If not, only resource will be assigned to one AZ

**Dedicated hosts:**

Resources will be assigned dedicated to the instance, cheaper, but we need to take care of OS image (bare metal as a service)

**EC2 Scheduled Reserved Instances:**

Used for backup servers, where we can use the instance as scheduled basis

**Load balancer:**

**First configure your instance as webserver:**

Add 80 port in SG in inbound.

Yum install httpd –y

Systemctl restart httpd

Systemctl enable httpd

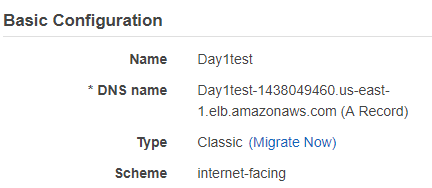
Echo “Autoscaling” > /var/www/html/index.html

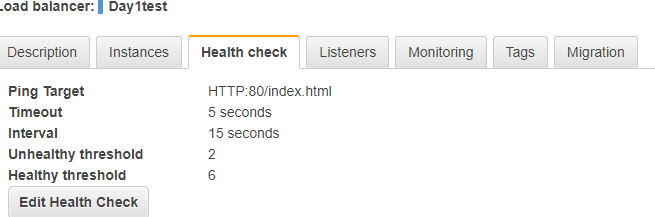
Keepalive signal will be sent across nodes in the load balancing.

Load balancing configuration,

Create another instance and configure webserver.

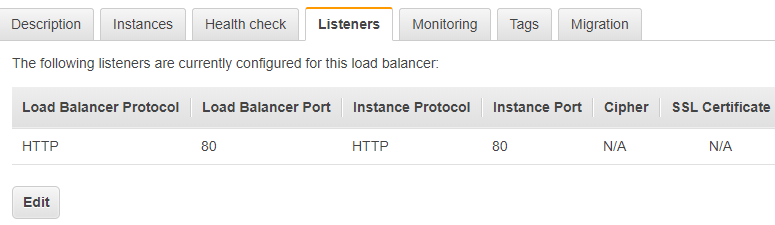
Make the index.html file with different content for second server, so that we can validate the functionality of load balancing.





Timeout-- > it will check for 5 seconds for every 15 seconds interval, if 2 consecutive timeouts, it will put in unhealthy state, once instance is up, consecutive 6 success attempts make the instance back to load balancer

Need to ask Vijay about data replication across nodes in load balancer



**Day2 :**

**What is ephemeral disk**

It’s a temporary disk(used for rootvg) added to your instance.

Data will be lost if we terminate.

Create two instances in 2 different availability zones.

Go to Volumes , label the respective volumes with respect to it’s server.

Volume is restricted to its AZ, we cannot attach it to the different AZ.

Create new volume and attach it to the instance.

[root@ip-172-31-85-188 ~]# fdisk -l

WARNING: fdisk GPT support is currently new, and therefore in an experimental phase. Use at your own discretion.

Disk /dev/xvda: 10.7 GB, 10737418240 bytes, 20971520 sectors

Units = sectors of 1 \* 512 = 512 bytes

Sector size (logical/physical): 512 bytes / 512 bytes

I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk label type: gpt

Disk identifier: 9D1DC67C-CE93-45F3-BB78-138B371E7678

# Start End Size Type Name

1 2048 4095 1M BIOS boot

2 4096 20971486 10G Microsoft basic

Disk /dev/xvdf: 1073 MB, 1073741824 bytes, 2097152 sectors

Units = sectors of 1 \* 512 = 512 bytes

Sector size (logical/physical): 512 bytes / 512 bytes

I/O size (minimum/optimal): 512 bytes / 512 bytes

**Autoscaling:**

In autoscaling, Launch configuration means PROFILE.

First we need to create 2 profiles.

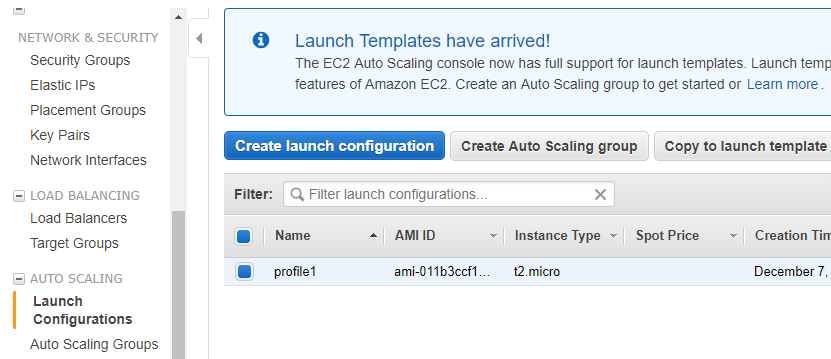
Then we need to create autoscaling group, make sure you keep profile name and group name as same , for tracking purpose.

In free tier, we can use max group size as 5.

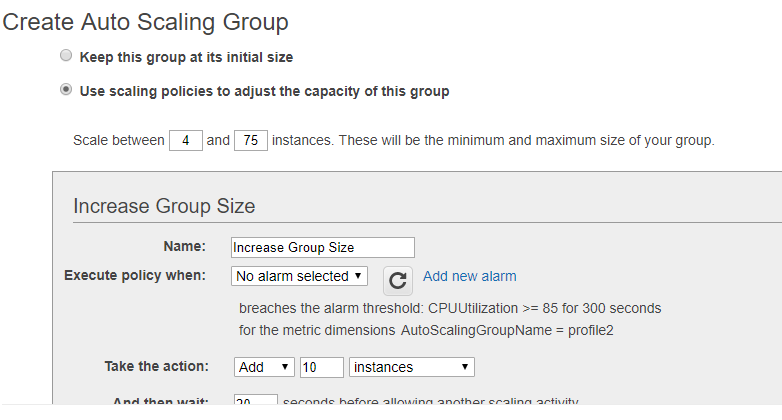
Put the instances in different AZ, to make sure it get proper resourcing.

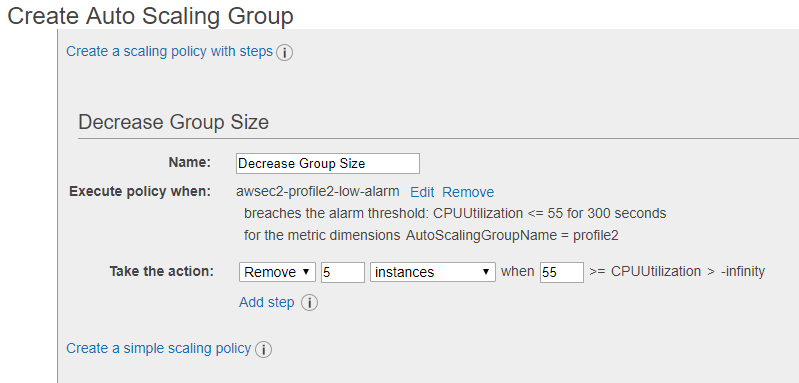
**CPU utilization in auto scaling group is average of all nodes.**

**In the console, go to Autoscaling, create launch configuration to create profile.**



**Create Autoscaling group as below:**



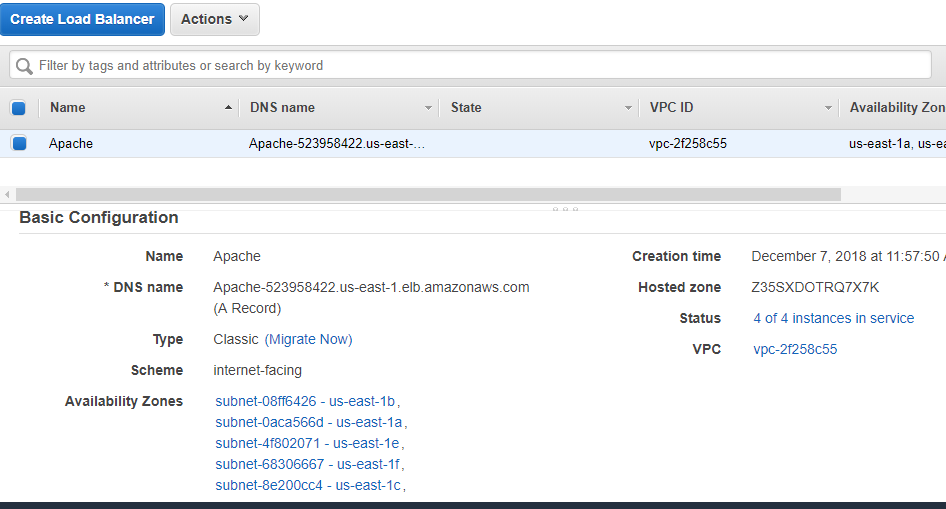


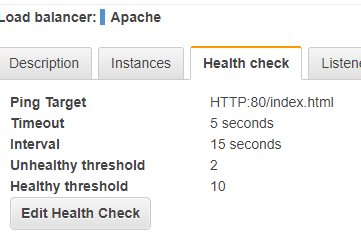
Then launch autoscaling

4 instances will be created after we configure auto scaling

**Now create with load balancing.**

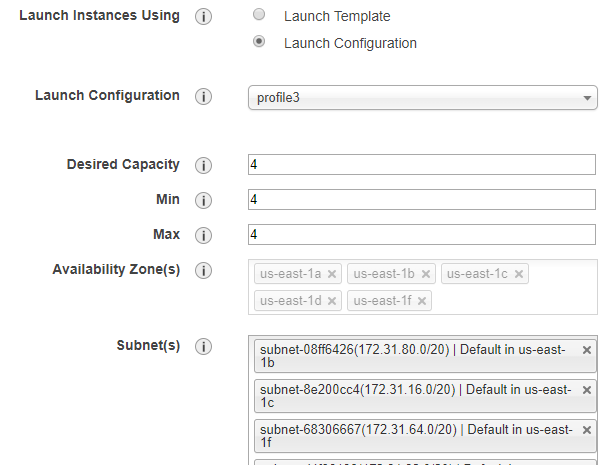
**Note: there will be no instances while creating load balancer, it will create once we create autoscaling.**

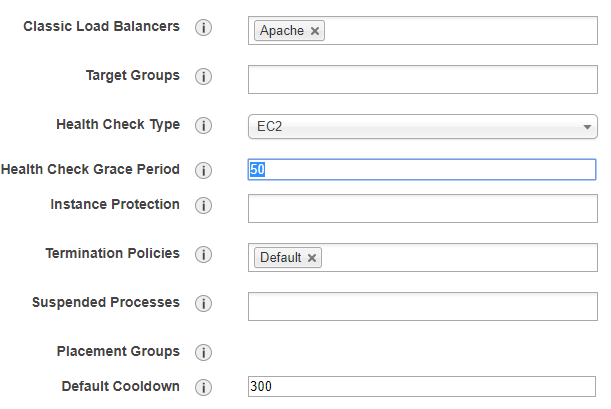




**Now create profile and define boot strap during configuration of profile itself.**

**Create auto scaling group with this profile defining load balance which we created.**



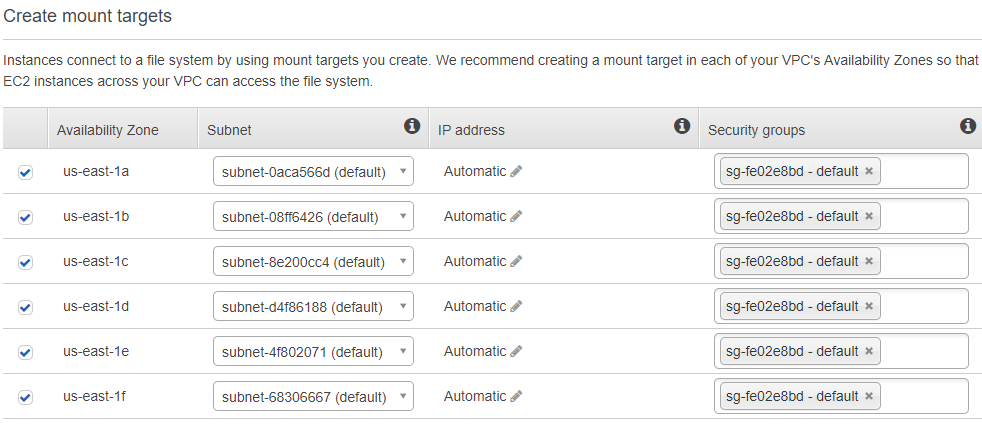


**New instances will be launched, once we done with auto scaling.**

**Go to browser and use the load balancer DNS name and see html content is accessible (what we created in boot strap)**

**EFS**

**Goto EFS service and create filesystem.**



**Provide the security group you created already.**

We recommend creating a mount target in each of your VPC's Availability Zones so that EC2 instances across your VPC can access the file system.

### Choose performance mode

We recommend **General Purpose** performance mode for most file systems. **Max I/O** performance mode is optimized for applications where tens, hundreds, or thousands of EC2 instances are accessing the file system — it scales to higher levels of aggregate throughput and operations per second with a tradeoff of slightly higher latencies for file operations.

General Purpose



Max I/O



### Choose throughput mode

We recommend **Bursting** throughput mode for most file systems. Use **Provisioned** throughput mode for applications that require more throughput than allowed by **Bursting** throughput. [Learn more](https://docs.aws.amazon.com/console/efs/provisioned_throughput)

|  |
| --- |
| Bursting |
| Provisioned |
|  |

### Choose performance mode

We recommend **General Purpose** performance mode for most file systems. **Max I/O** performance mode is optimized for applications where tens, hundreds, or thousands of EC2 instances are accessing the file system — it scales to higher levels of aggregate throughput and operations per second with a tradeoff of slightly higher latencies for file operations.

General Purpose



Max I/O



### Choose throughput mode

We recommend **Bursting** throughput mode for most file systems. Use **Provisioned** throughput mode for applications that require more throughput than allowed by **Bursting** throughput. [Learn more](https://docs.aws.amazon.com/console/efs/provisioned_throughput)

|  |
| --- |
| Bursting |
| Provisioned |
|  |

**Give next and give create filesystem in next screen.**

**Once filesystem is created, follow below steps to mount it.**

1 yum install -y nfs-utils

2 mkdir efs

3 mount -t nfs4 fs-16bc56f7.efs.us-east-1.amazonaws.com:/ /efs/

fs-16bc56f7.efs.us-east-1.amazonaws.com  DNS name will be provided once file system is created.

**S3 buckets:**

**Object storage** can be accessed via https/http

**Block storage** is a local storage.

In S3 service, region will be changed to Global

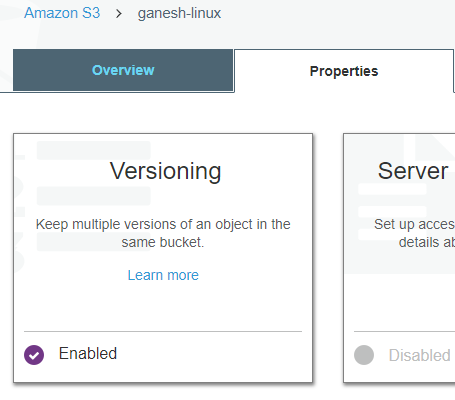
Create a s3 bucket and upload a file.

Uncheck block and remove messages from bucket level

Upload a file and set its permission,

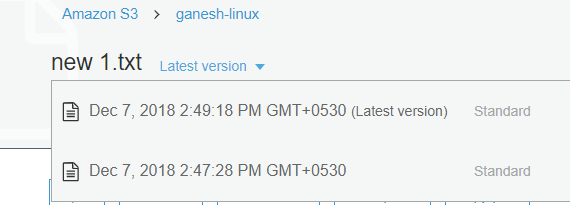
Use the link to access the file content.

Go the properties of the s3 and enable versioning

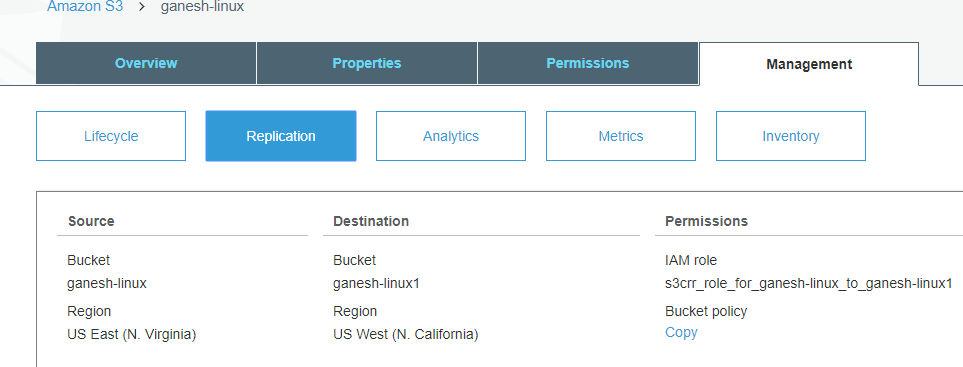


Update the contents of the same file with same name.

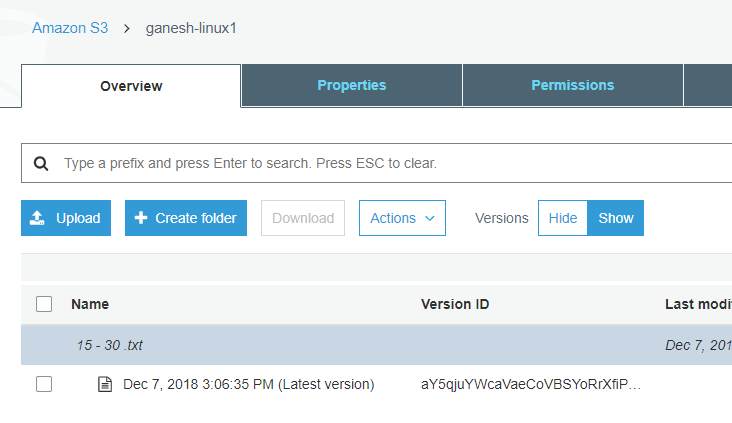
It will list both the files as different verions.



Enable Replication as below, to configure replication between 2 buckets in different regions.



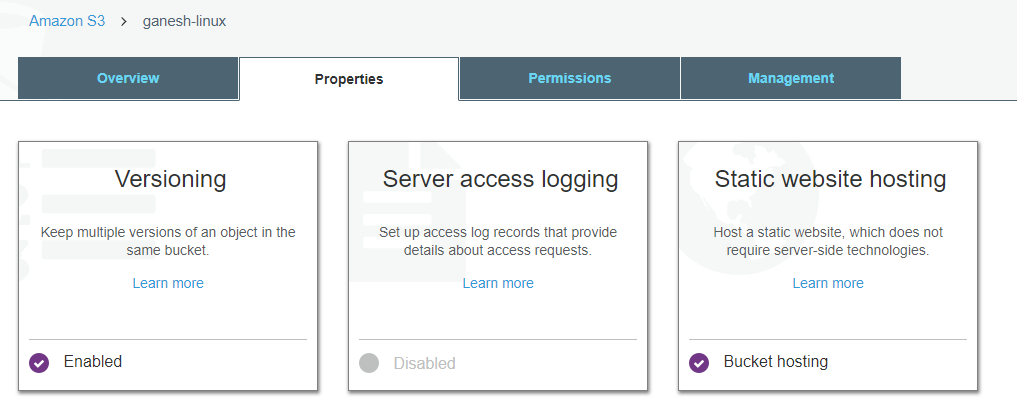
Upload a new file to source bucket, it will be replicated to the destination bucket



Why we need more s3 bucket replicas?

Static , Dynamic website difference.

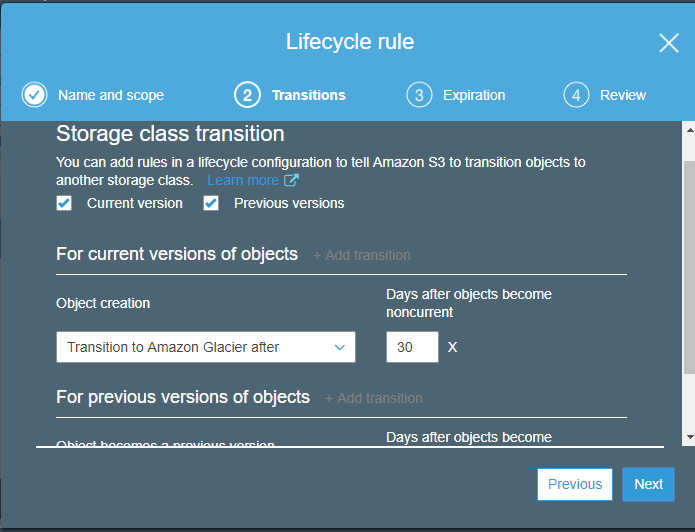
Enabling static website.

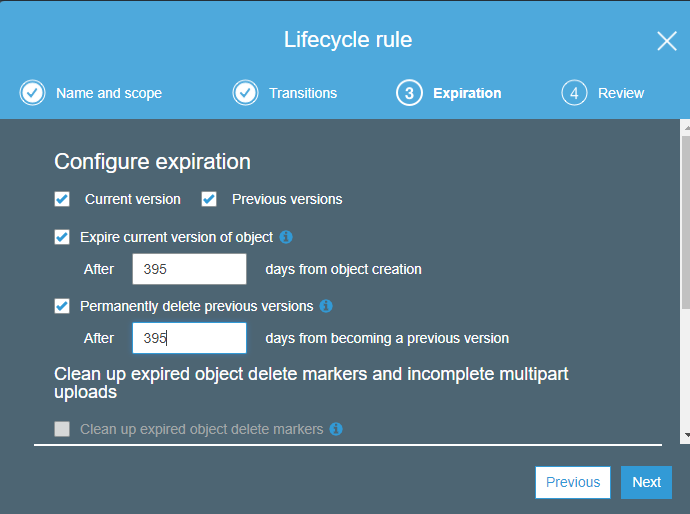


Glacier

Support has to be contacted to get the data from Glacier.

O-Auth used to validate





**Day3:**

**IAM:**

**To have a customized AWS link for a particular domain**

**Goto IAM dashboard and create the alias name like below**

IAM users sign-in link:

https://ganesh623.signin.aws.amazon.com/console

then go to “delete your root access keys” in the same page, and create a root key, credentials will be downloaded to a csv file.

This user name and password will be used for automation from ansible/python

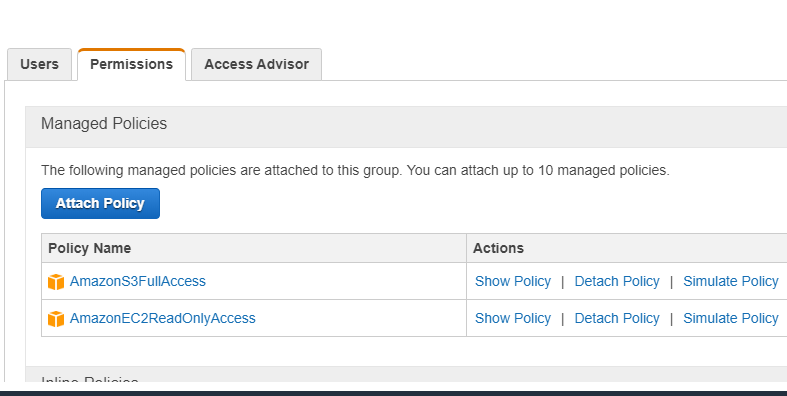
**Authentication:**

**Validating correct user or not**

**Authorization:**

**What user he is and what he is supposed to do**

**Create group first and add policy to it as below,**



Now create user ec2-operator and try to launch instance, it will not allow as per the above set group policy.

**MFA:**

Goto IAM dashboard,

Activate MFA

Use the mobile app, google authenticator and scan the QA code,

MFA will be active now.

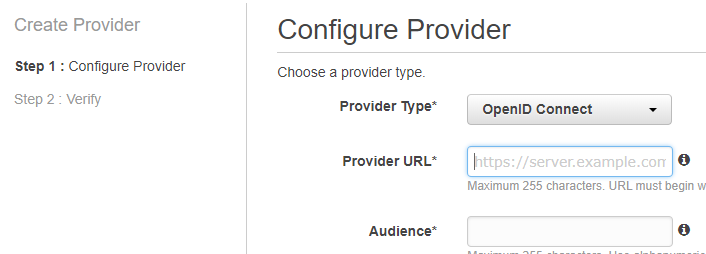
It’s like soft token (like RSA)

2 way authentication is enabled now.

**Note:** Remove MFA immediately once tested, it will be highly charged

**Identity provider:**

**Go to configure provider and select openID connect.**



**Cloud front:**

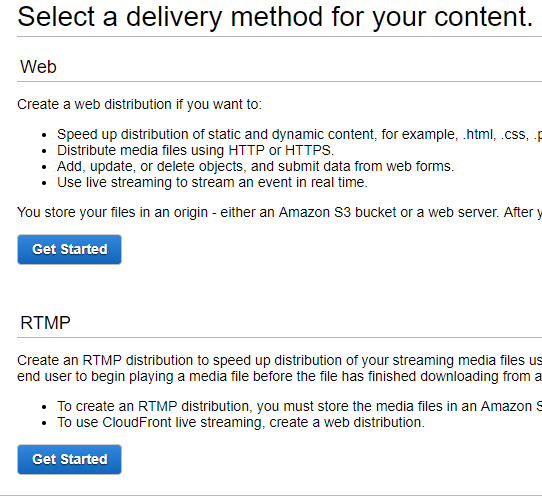
Cache memory: data will be cached for certain period, during that time, data will be retrieved from cache.

Content Delivery network – CDN

Edge load balancer – region wise it will be there and it will be connecting to nearest server to perform the response in faster way.

Example: Bigip, radware

**Creating CDN:**



RTMP – used for vidoes/animation (real time messaging protocol)

Web – used for files

Select web

In origin, provide the s3 bucket created for cloudfront, upload a jpg image in this s3 bucket.

Set ttl, default and max value

For example if it’s set for 4 hours, it will be caching within this time period,

After expiry, again it will check for server and identify the edge load balancer and caching will happen for next 4 hours.

s3 bucket url:

https://s3.amazonaws.com/gannycloudfront/krishna.jpg

Domain name for cloud front: d2bhio8v5lo9lg.cloudfront.net

change url like below

<https://d2bhio8v5lo9lg.cloudfront.net/krishna.jpg>  same image is accessed through CDN.

**Networking concept:**

Only Switching – layer 2

Switching and routing – layer 3

**Datacentre zones**

Webservers – redzone

Yellow zone – database servers (caching)

Green zone – app db server not exposed to internet

Web application server – tomcat, apache – dynamic hosting.

Horizontal scaling – increasing servers

Vertical – increases resources in servers.

Net.ipv4\_forwarder  need to enable this for outside network.

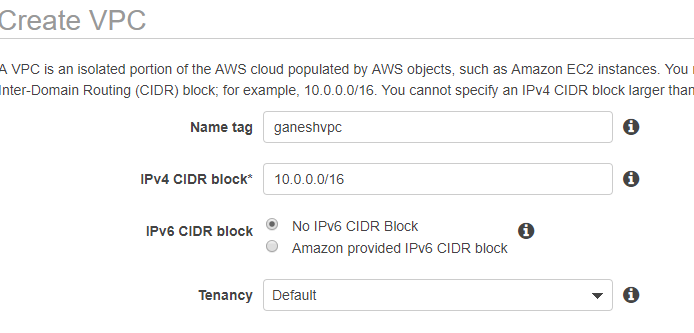
**Ceph storage: highly used nowadays, used all small storage disks and provision as a largers storage**

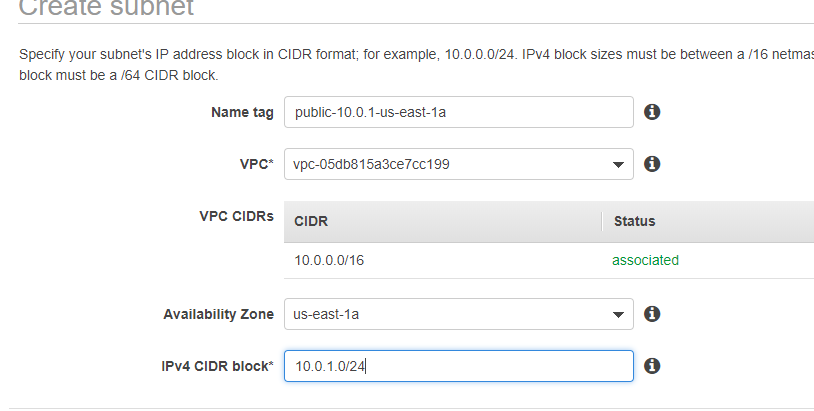
**VPC:**

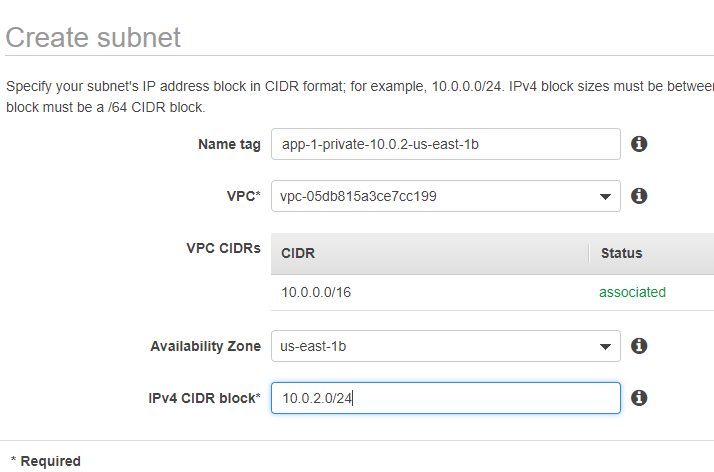
**Before creating vpc, choose the vlan for all the instances.**

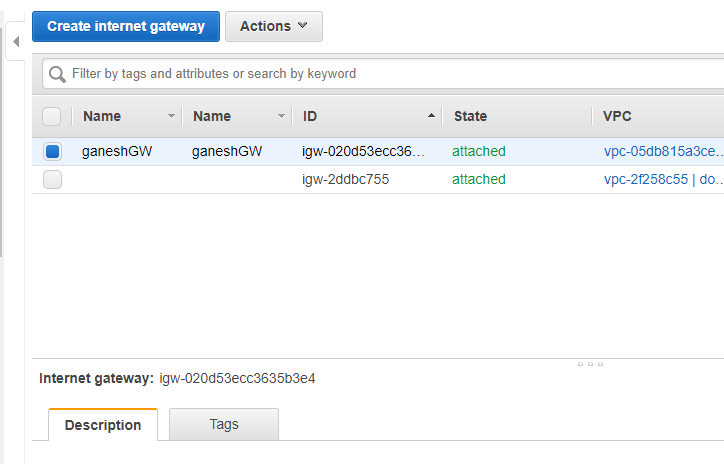
**10.0.0.0/16 – choose from vlan 10.0.1.0/24 , 10.0.2.0/24**

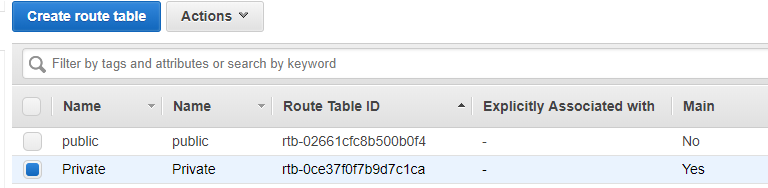
**First rename the default vpc to don’t delete, if deleted it will not be recreated.**



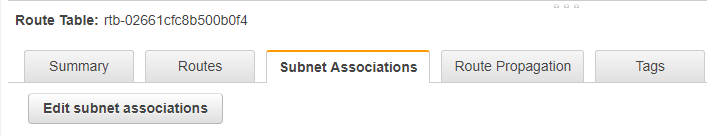


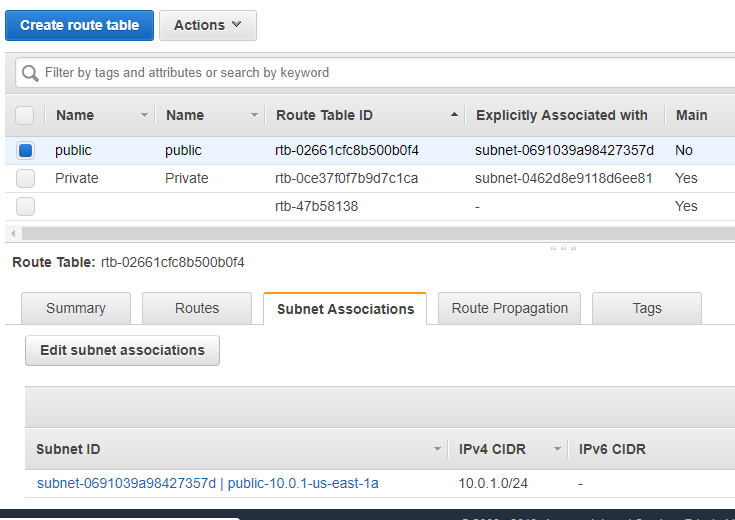






**Associate with subnet**





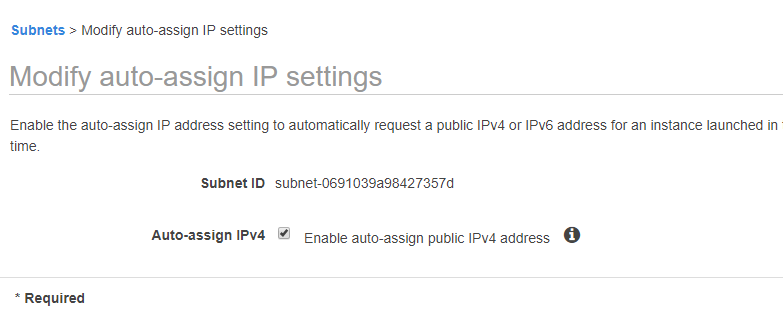
If 0.0.0.0/0 bot rule is not added, we cannot able to add during instance creation

Add this route to both public and private route table.

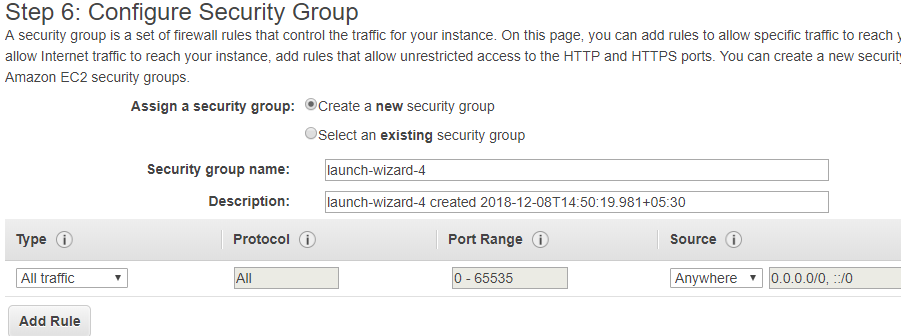
**Tips: Port Address Translation (PAT), is an extension to network address translation (NAT) that permits multiple devices on a local area network (LAN) to be mapped to a single public IP address. The goal of PAT is to conserve IP addresses.**

IP will be shown as internal only, to outside traffic it will be public.

Below option is only for instance created for public IP



open ec2 window in next tab --> first launch one instance and select your VPC and select public network and ensure the public IP option is enabled.in the security group open all



But in prod , don’t use this option, mention the ip range in source –custom

Example: 172.0.0.0/24 to 172.10.0.0/24

open ec2 window in next tab --> launch second instance and select your VPC and select private network --> in this security grou you have to say the source is 10.0.1.0/24

login public instance (jump box) --> ping your private machine --> try to login yur private server from your public machine

now to get internet access to your instace with private IP, we need to create Natting server with public IP,

lauch instance community AMIs search for nat select first imagecreate normally like public instance described above.

Go to routes private edit routes in target ,give the instance id of NAT

Once this is done, to enable IPV4 forwarding, right click nat instance  networking change source/dest.check and click disable.

After performing this action, outgoing traffic is enabled for private instance.

**Note: So incoming traffic to private instance happens through bastion instance.**

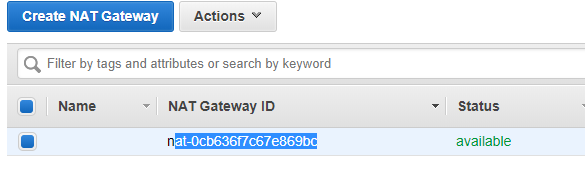
**Outgoing traffic happens through natting server**

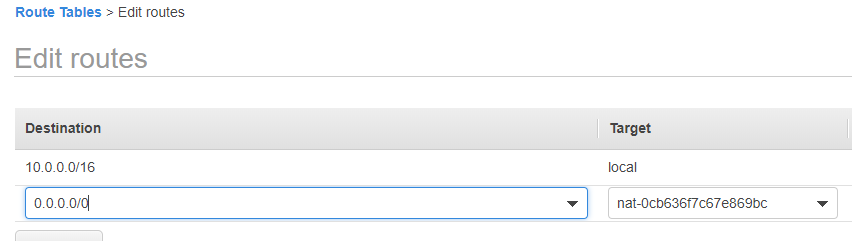
**Next scenario:**

**We need to create NAT gateway for all outgoing traffic from private instances.**

**I earlier scenario, we used natting server, which is not recommended in production.**

**After creating NAT gateway, copy the NAT gateway ID, and provide it it routing table of private instance,**





**Refer aws-cli.txt file for configuring AWS through cli mode.**

**For using ansible to manage AWS, below 3 packages are required,**

**yum install python2-boto python2-botocore python2-boto3 –y**

**Day4:**

Create 2 instances in two different regions with load balancer configured.

Validate both the region load balancers are working fine.

Note1: Snowball service used to move huge data from on-premises to AWS cloud.

Note2:

Aws direct connect

When used, dedicated line will be used between company and AWS, so data transmission happens through this instead of internet.

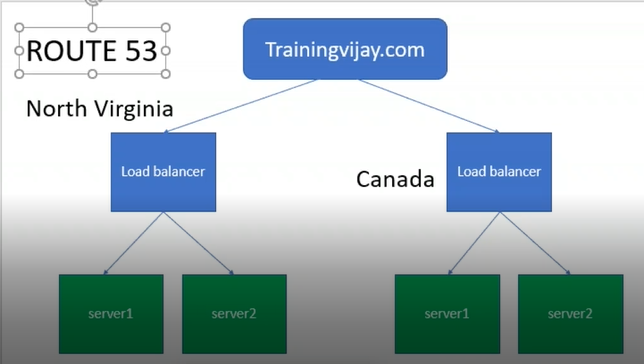
**Route53**

**Uses:**

If my office is in nvirginia and Canada, two load balancers in 2 different regions and if we use route 53

1. Divides traffic at route 53 level and gives to region wise LB, LB again distributes the traffic.
2. Load sharing can be controlled manually through route 53, by defining how much need to go to DC1 and how much to DC2
3. Whatever traffic comes to the website, we can maintain the latency across the datacentre - automatically controls the data flow.
4. Whatever traffic comes to the website, we can control from which DC traffic should go. – GEOlocation – traffic goes to nearby location.
5. Whatever traffic comes to the website, traffic can be moved to secondary region when primary goes down, and we can define it.
6. We can create alias to the website, like dev.inttra.com to test it and we can cutover and provide actual website once testing is over.

**Route53 configuration steps:**



Goto AWS console,select two regions, N.Virginia & Canada

Create two instances in each region

Copy bootstrap and configure in all instances

So now, green part in the above fig is completed.

Create Load balance in both the regions and associate the instances.

Check the webpage with the individual server instance’s public ip first.

Then check with load balance dns name and webpage should load with all instances html page.

Goto route53 service,

Goto register domain,choose a domain, choose .com domain

Purchase .com, which will be faster.

Continue,complete purchase.

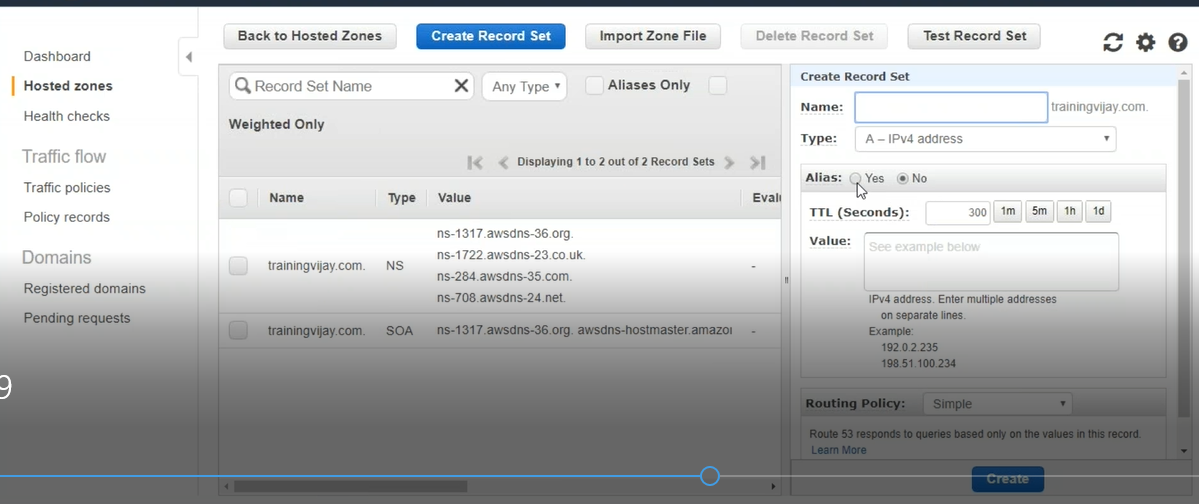
Go to billing dashboard, pay it.

Once paid, it will come under registered domain.

Now we can for route53 for testing.

Go to dashboard go to hosted zones

Create record set.



In Alias target, select the LB we created.(set yes to alias)

Routing policy-> select **simple, weighted** (to distribute load)

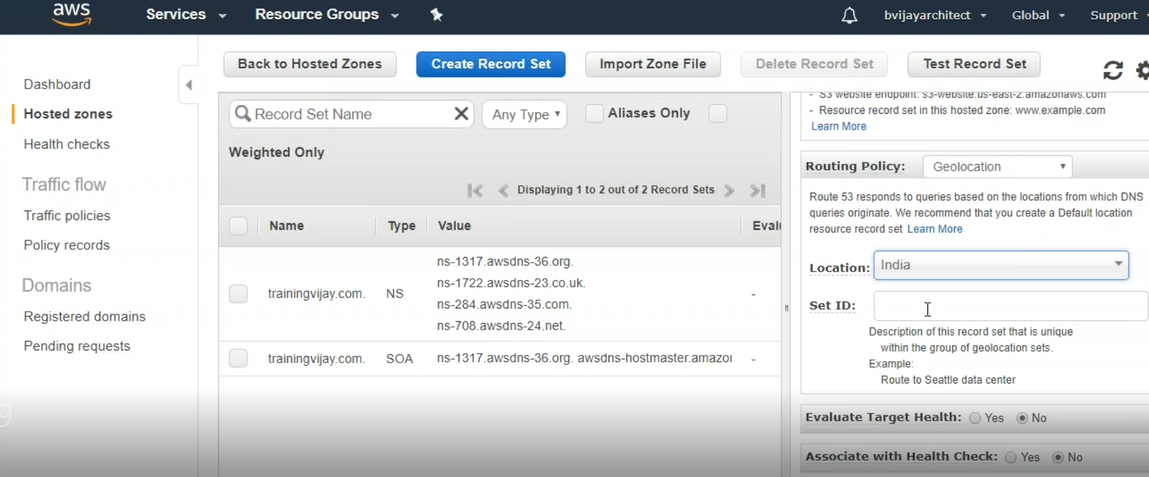
Give 80% to one region and 20% to other region.

Access trainingvijay.com and see whether u see region 1 or 2

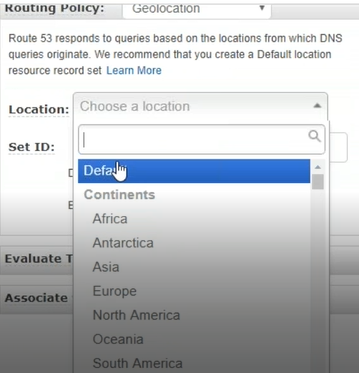
Now select routing policy as **latency**,

If we set, 50 percentage for both the regions. Based on the load, requests will be shared across regions.

Install tunnel bear sw (free sw) (for Geoloaction)

And select your location as India 

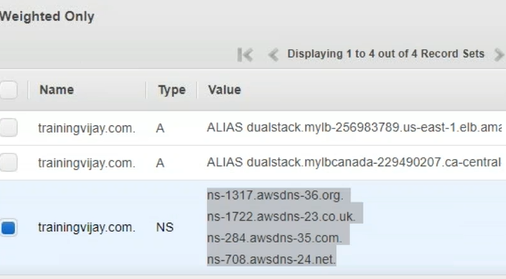
Select alias type to US region and choose default location.



Try accessing trainingvijay.com, try from mobile, it will be faster that in laptop.

Thorough tunnel bear, change your location and try to access the site.

Based on the record set created, route 53 will decide where the traffic should go.



Above are the DNS servers allocated for our website.

---------------------------------------------------------------------------------------------------------

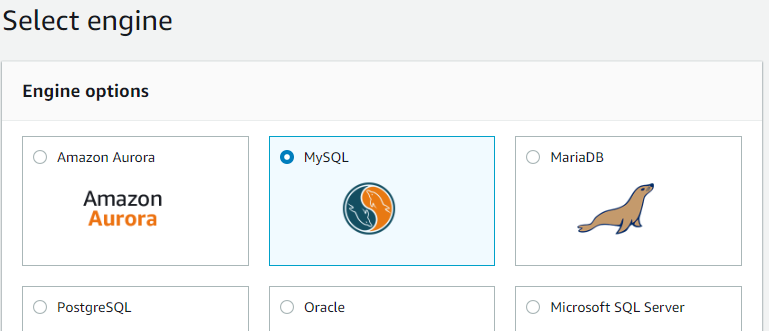
Database error – internal server error, we get in website, due to database issue. Application is not able to connect to database.

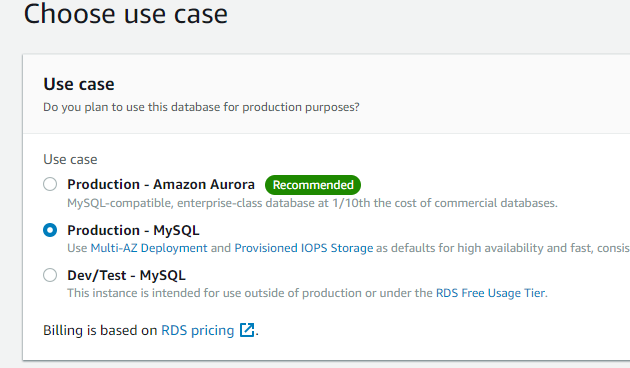
**Metadatabase and databases:**

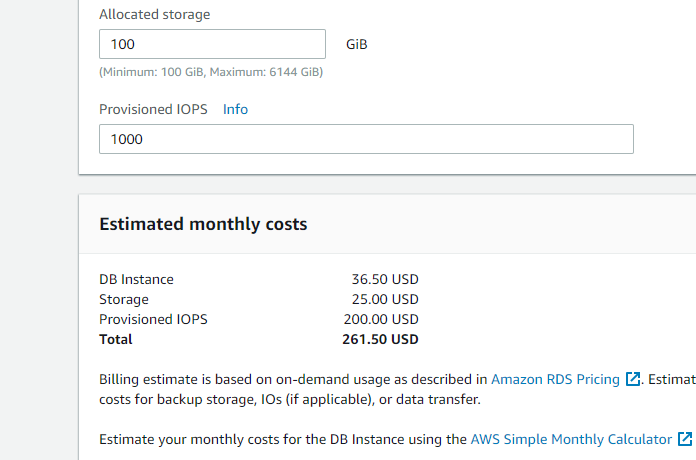
Meta database has the property of all the databases information , similar to inode concept in filesystem.

So applications connect to meta database to connect the specific database.

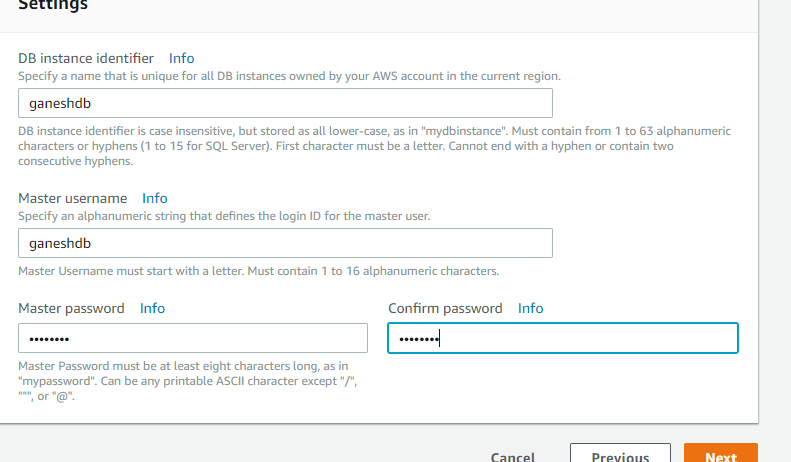
**RDS (Relational database services)**



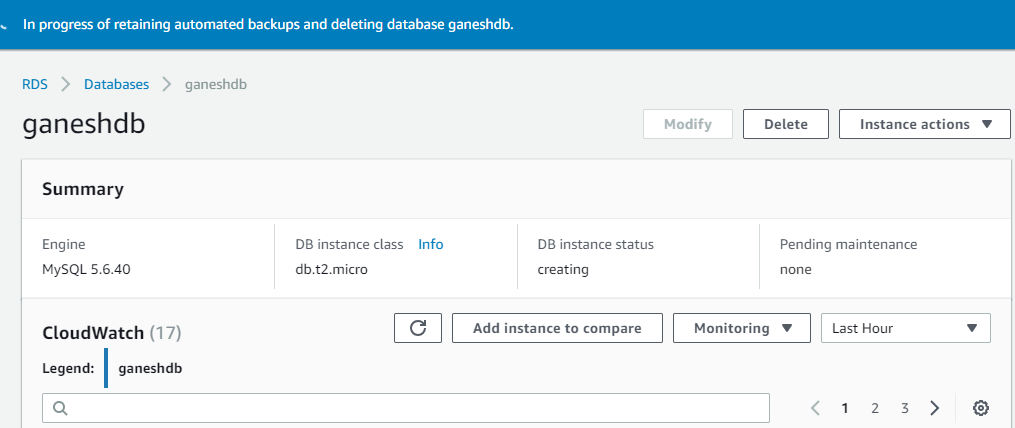




Go back, and select free tier, and configure db



Note:VPC should be there before creating this.



We will get a url(end point name), which needs to be given in the tns.ora file.

Don’t create this in free tier, it will be charged.

**AWS market place:** any startup company can create their customized Linux image and place it here.

How to create AMI?

Create instance

Create a volume

Create a snapshot

Create an Image  this will be listed under “My AMI’s” while lauching instance.

To delete:

Go to AMI deregister

Now delete the snapshot.

--

**Tags:**

While launching instances, Provide tag name to identify instance easily,

We can filter with tag names.

**Tips:**

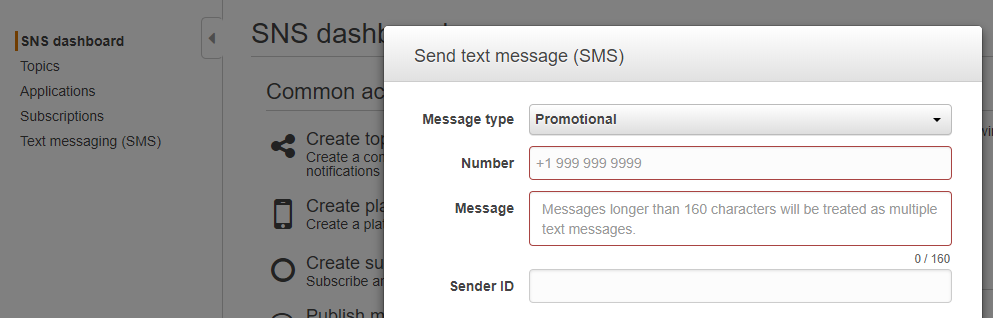
License Vs subscription

License – approval we get for OS support

Subsrciption – we get for product support

SNS dashboard.

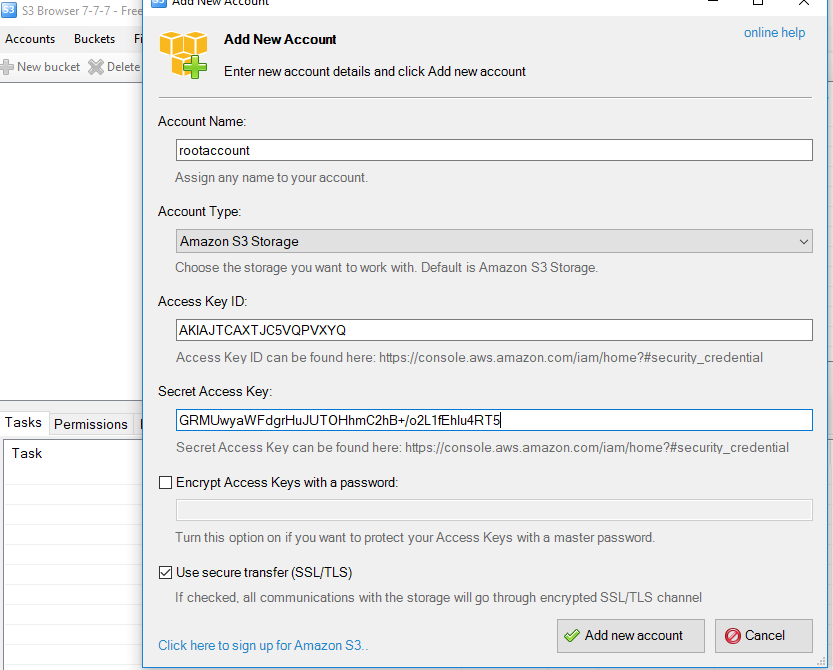
Whatever alert we trigger through other service will be processed through SNS server.

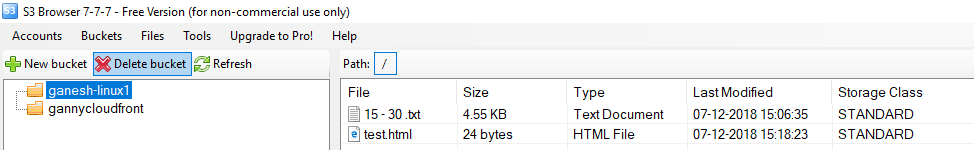


Here, promotional is a general message for offers,

Transactional is the official service alerts.

**Using S3 browser: (similar to winscp)**





We can use this tool to upload files to S3, only limited options are enabled with this free trial.

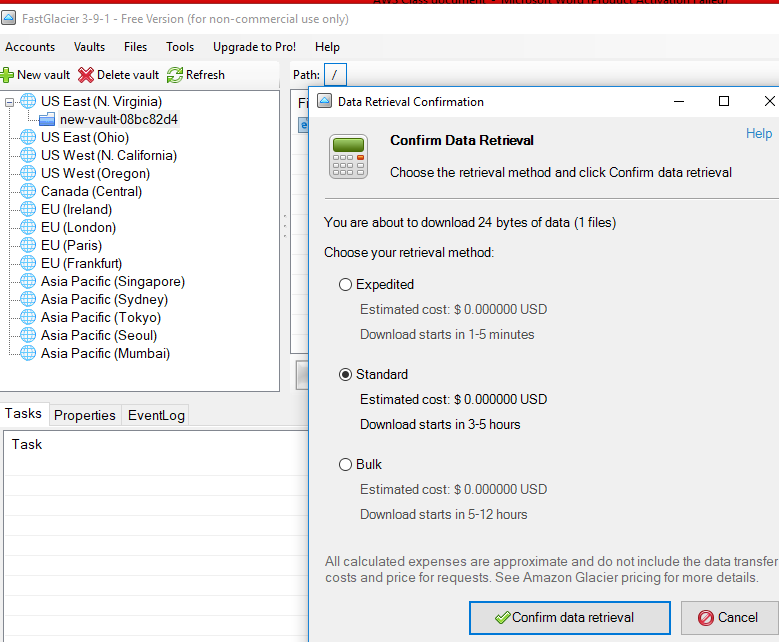
Upgrade to pro-version to get advanced features.

|  |  |  |  |
| --- | --- | --- | --- |
| **Product** | **Unit price (USD)** | **Quantity** | **Total** |
| S3 Browser | $29.95 |  | $29.95 |

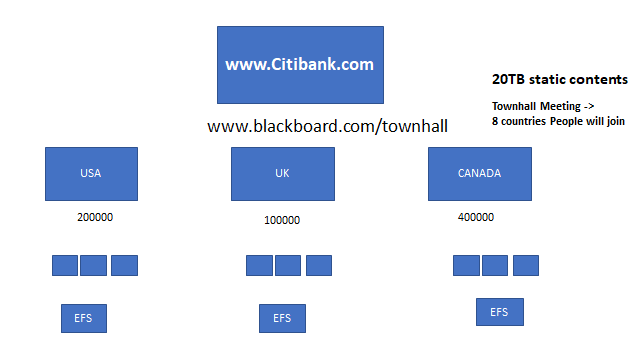
**FastGlacier**

**We can upload files to glacier.**

**For download , it will be charged for immediate downloads.**



**Scenario:**



Get the domain name, consider the hits

Since Canada is getting more hits, need to configure master DB in Canada region.

Consider other regions as slave DB

Using RDS to configure Canada as a master DB, others as slave – replication.

Calculate the usage for minimum level and max hits of the websites and decide how many min and max instances required.

1. Create VPC , (private,public,NAT)
2. Configure auto scaling with load balancer.
3. Need to point tns.ora on all app servers.
4. Now, to sync all the data in web servers, we need to use EFS. (Region level)
5. Now, to sync across all regions, we need to use S3 bucket with specified users and permissions.
6. We need to create managed instance an all regions and we are mounting EFS and S3 bucket on this server and configure rsync to sync between both EFS and S3.

