Amazon cloud web service :- 12/04/22

* Physical
* Vertical.
* Instance.

Aws, GCP, Azur, IBM, Bluemix	– Cloud.

1. Choose AMI :-

* My AMI's :- Image.
  * we will get Entire set-up of that particular image. ex:- Tomcat

* Aws Marterplace :- Vendors.
  * Here we have to pay rent per hour.

* Community AMI's :- Instant store instance.
  Here when we loss the instance data is also lost.

  EBS :-
  Elastic block storage.

2. Choose an instance type :- There're different types of families present. (Types of CPU's)
   ex:- T series, E, D etc.,

* General Instance :-

* Compute instance :-

* Memory instance
* Storage instance
* G.P.U instance

→ Burstable performance instance :- (T_2 instance)

When the user is doubled then if not loose the users here not to impact the user increase to instance from (t_2 medium - t_2 larg), here (BPI) comes into the picture, here credits are add for the hours instance not running and this credits are used for increasing instance storage and this is applied for only that day.

3. Configure instance :-

Number of instance :-

purchasing option :-

EC2 on Demand :- pay for what you use
Has the highest cost but no upfront payment.

EC2 Reserve instance :- 75% discount
Reservation period 1 or 3.
if paid upfront more discount.

EC2 spot instance :- 90% discount, but if it taken for 1 hours after that someone bids for more that instance will go to that person.

EC2 Dedicated Hosts :- Same pricing as on Demand instance

Network :- default

Subnet :- region availability.

Hostname type :-

placement Group :- for clusters ; spread, partition

Cluster	Spread	Partition
* Same availibity - Zone * Very fast * low latency.	* different availibity Zone. * limited to 7 instance * Not sufficient for huge work	* up to 7 partition * up 100 instance * for more effective * KAFKA, HBFC

Advanced Details :-

User data :- bootable scripts

4. & Add storage :- We can change the size up to 50GB and we can had Volumes.

5. Add tags :- providing the 'Name'
ex:- webserver
Tomcat
SonarQube etc.,

6. Security Group:- We will provide the security ports for the access to the apps.

eg:- 10.0.0.0/28 SSH

HTTP

(or)

(up front) Create the security Group and then configure with that security Group to the instances.

Here access is given for who can use (or) open the instance

13/04/22

Creating Templets (Reusability).

templet-name and other requirements upfront.

want to change instance type then 'stop' it and can change the type.

Similarly Termination protection if 'enable' then it can't be deleted to delete it we have to 'disable' it.

Similarly user data also stop and then change the the user data.

System dogs also can be seen

Images can be created with whatever the particular image is having.

Elastic Block store :- (EBS)

Volumes :- (Backup) (8GB) is default volume for any instance. to modify this volume the action and modify the volume

** root volumes can't be used straight forward we have run some command (growpart).

we can also create the volumes upfront on the Zone level and create volume.

Backup's (Snapshots) :- Backup of volume (data)

create snapshot :- Give name.

Lifecycle Manager :- for automating the snapshots.

Policy Schedule :-

name :-

frequency :- Daily/weekly

Every :- Hours

Starting :-

Retention type :-

Retain :-

This lifecycle policy for important tool like Jenkins, sonar etc.,

We want to access the instance in another region

Ex :- If in Ohio then I want to use in mumbai.

two way either :- entire state (or) data

Entire state of instance :-

create a image once this is ready take to the other region (copy ami) destination region and copy. hence image created on other region, now create server with this image.

Only data :- then create a snapshot of that particular volume then select that snapshot and copy it there they will ask the region, give the region and hence it is copied to that region and creat a volume from the snapshot

and attach this volume to the server, hence you got the data from one region to another.

Encryption :-

We can encrypt the volume while creating the server, now if encrypt data after server creation then create the snapshot (Actions, copy the snapshot and encrypt it) we will get an encrypted snapshot, now create a volume with that snapshot and now we can create a server with encrypted volume.

Elastic Ip's :- static Ip (Allocate Elastic Ip) create and then associate this Ip by choosing an instance. there we will get an static Ip.
If we want to release then deassociate the Ip and release it.

Load Balancing :- Route traffic and check healths of the server and SSL Termination

creating Load Balancer :- restricting ports on LB level.

Classic Load Balancer :-

Name :-

Security :-

Advanced details :- it will tell if the server is running (or) not.

we can access with DNS name as well.

" This is 'round Robin Algorithm' "

Port configuration :-

Stickiness :- seconds.

for the given time (sec) it will route to the only 1 server.

(ALB) 18/04/22

Application load Balancer :- Smart load balancing.

Load Balancer

create Load Balancer

Name :-

V. P. C

Target = Instance.

For micro server based application.

Network load Balancer :- Similar to classic load Balancer

* works on 'TCP' ports
* For processing millions of requests per second.

** IAM :- Indentity Access Managment.

i) User Groups
ii) Users
iii) Roles
iv) Policies

User

Add user Groups

copy same as existing user.

Policies
↓
Set of permissions.

This is to restrict the users on user level.

User group

Create a group.

group name :- Tester.

add group level permissions and creat a group

***

MFA :- Multi Factor Authentication.

↓

users:-

particular user

securing credentials.

MFA device :- Manage

Continue

Download app

Show QR code.

Scan code.

we will get 2 codes.

imp { MFA code 1
MFA code 2

Need this code to login along with user & password.

Roles:-

Managing users (or) managing serviceses.

TO create a Custom role

Image id (AWS) (1) page → there you can find.

Instead of using the "Access key and secret keys" we use 'Roles' in AWS.

8:00.

Roles

Create role.

AWS service

EC2 → Next

⇒ Give access

- Permission policies.

} Custom role created.

Role name

ec2-s3

Now, Instance → Action

Instance setting

Attach/Replace IAM role

IAM role [ec2-s3]

Policies :-

Granual level of access.

This is to create our own policy.

list

Read

write

Tag

Permissions

while creating role we attach the IAM at stage (3).

19/04/22

Simple Notification Service :- (SNS)

First create a topic.

create Topic.

FIFO ◉ Standard.

Name

Email

create topic.

⇒ Subscription :-

create sub

Topic ARN

Protocol : Email

Endpoint : email id

create subscription.

⇒ Now on E-mail "mail" will be sent and click on link.

Cloud watch :- Monitoring Tool in AWS.

↓

Service.

For ex:- EC2 Service has monitoring service for every 5 min and it's free. (data points are collected).

Detail monitoring is cost based.

Cloud watch

⇒ Dashboard

create dashboard.

Name :

Select widget :

☑ Metrics ○ logs

Add metric graph

Browse.

Search for instance and add metric

create weboget.

Dash boards are not reagion Level, so we can use dash boards Global level.

Alarms :- To keep an eye on servers we use alarms.

it can also take some actions as well.

⇒ All alarm.

→ Metrics.

→ EC2

→ instance level metric.

copy past the instance id.

it's a Graphical representation

Cloud watch :- Log streaming.

in ELK it will reformate the data but in cloud watch.

it will store in log group but not formate any date.

Integration b/w EC2 to cloud watch.

service to service communication.

using roles and Policies.

create the policy. (from Pdf).

Go to IAM

policies

create policy.

Json

⇒ Past code.

review.

Name - aws-log-group-policy

create policy.

Now, create the role using the policy.

IAM

Roles

create role.

EC2 (Next)

select ~~@~~ EC2 policy we created.

Next

Role name:- aws-log-group-role

create role.

Now, Go to instance select server and actions

instance setting then Attach/replace the role

IAM role : aws-log-group-role.

apply.

close

Now, to push the logs from EC2-cloudwatch we need the agents for that install the agent in server.

cd Downloads/.

Yum install awslogs (awslogs is agent).

cd /etc/awslogs/.

ls.

vi awscli.conf

=> change to region.

vi awslog.conf

=> remove all the entries.
and paste from (pdf).

file to be updated by us.
region =
file = /var/log/(service name.log) → real time.
log_group_name = (service name.log) → real time.

Now, start the agent.

service awslogsd status.
service awslogsd start
service awslogsd status.

cd /var/log.
ls
touch application.log.
ls
echo "This is test data" >> application.log.
" " " " " " " " } creating log.
" " " " " " " "

Now, to check go to AWS

log groups and check.

Now, to get the alarms.

applications_logs.

⇒ metric filter.

⇒ filter patter (which logs you want).

ex:- Error.

Next

metric details:-

key word :- ERROR.

and give details.

and create the metric details and set alarm

Cloud watch = creating dash boards, alarams, log stream

- ming, scheduling triggering the events (Lamda).

with prior to monitoring

Lamda :- execution of scripts and automation

without servers is Lamda (Lambda).

* Server less Executions = Lambda

Lambda.

Functions.

create functions.

Scratch.

Function :- EC2-start.

Runtime :- Python 3.9.

Permissions :-

To authenticate create the policy. format.

IAM

Policy.

Jason

⇒ Paste the code (pdf).

Name :- Lamda policy.

[create policy]

Role

Create role

Aws service.

Lambda.

☑ Lambda policy

[next]

Role name :- Lambda role.

NOW, on Lambda:

✓ We an existing role

→ Existing role.

Lambda-role

| create function |

EC2-start

code Test monitor Conf

↓

Take code from pdf

place the instance id's in the script.

Events (cronjob)

Rules.

Black to cloudwatch event.

create rules.

○ Event pattern

○ Servic name - EC2

○ Event type - EBS

(or) ○ schedule.

○ fixed rate of

○ cron expression.

⊕ Add target (many types).

Ex:- Lambda function

EC2-stop

20/04/22

Auto scaling :-

* Scaling up went Load is Max.
* Scaling in when Load in Min

Launch config

creat Launch config.

Name :- My-LC

AMI :-

Instance type :- t2.micro.

Assign security group :-

New security group

✓ existing one

Key pair

Choose existing pair.

Existing pair :- Vamshi-devops

| create Launch Config |

NOW, create Launch Conf group.

Auto Scaling Group

Create Auto Scaling group.

Name :- my-ASG

Launch template :- re-usability

✓ Launch config :- NO re-usability

My-LC

Next

VPC

⇒ Select in all the Zones (Available Zones).

NO Load balancer.

Health checks

Additional settings

monitoring.

☑ Enable the cloudwatch.

Group size.

mini capacity	1
max capacity	5
desired capacity	1

Scaling policy :-

Target Tracking policy.

⇒ Metric type

Average cpu utilization } — it is only for scale up.

Target value

70

instance need

180 sec

↓

But in cloudwatch All alarms we can see that scale in is also created

⇒ Add notification

SNS Topic :- email

Create auto scaling Group

NOW to check auto scale up and in so, put the load on the server for that

Sudo amazon-linux-extras install epel -y

Sudo Yum install stress -y

Stress

stress --cpu 90 --timeout 420 &

"&" it will run in backed in linux 'd' in docker

"top" → to check the load on the server

on only Tracking policy but we also have

Step scaling, scaleout in dynamic scaling.

If we want to clean up auto scaling then go to auto scaling and delete it.

*RDS :- Data base. (Transcational data).

Relational data base server.

Data base

Create DB.

Standar create (or) Easy create.

**How many day's backup can be maintained on RDS :- 35 days.

⇒ Log exports :- Go with all

create data base.

Data base is ready.

now, to connect this RDS we have to have a Server and set-up mysql on the server and connect to the data-base

21/04/22

RDS

↓

connectivity and security

Endpoint and Port.

On server to connect to RDS

* mysql -h endpoint -P 3306 -u admin -p

Password : the Passw which set earlier while RDS set-up

⇒ for that install mysql on the server

* Yum install mysql

* mysql -h (database from aws (RDS)) -P 3306 -u admin -p

Password :

Now, this can be seen in 'cloudwatch' in logs

'log Groups'.

After the completion we can delete the RDS.

Relational database server

Cloud Formation service :- From (AWS) side.

cloud Formation templet :- create and Manage resource with Templets.

* AWS Template Format version
* Description
* Metadata
* Parameters.
* Mappings.
* conditions
* Resources :- Mandatotory section. (required section).
* Outputs :- This section is for outputs of cloud formation displayed on cloud formation console.

CloudFormation (AWS).

use a Simple Templet.

Lamp stack.

(code) → View in designer

Now, there is a GUI where we can create the Template

Here, we can View has a Component (or) as Template

Create template Designer . by UI .

Resource type

drag and drop this one
to the right side, what
-ever is needed

(or)

we can create with stack

in real time we will use Terraform (or) cloud formation

***

EFS :- Elastic File System :-(To manage the data .

To store dynamic data EFS is used.

Elastic file systems are automatically scalled.

Now,

Elastic file system

Create file system. (and fill in the details)

EFS will also have security Group.

with EFS two different servers can share the data.

b/w them

22/04/2022

S3 Buckets :-

To Manage the static data (images, video).

the date which can't change is static data.

Features :-

* Version control
* Antomatic Backup
* Storage classes
* Website hosting (ex:- Netflix).

⇒ Buckets are folders with unlimited storage but on the content level storage is there which is 5TB.

Global Serviceses :-

* IAM
* S3 Buckets
* Route 53.

Amazon S3

⇒ creati Bucket

Bucket Name :- "Name should be unique".

Aws Region :-

⇒ Object ownership

Block public access

Bucket Versioning :-

Default encryption :-

Then Go to the created Bucket and start uploading

the Bucket

create floder upload

Files and Folders Add files Add folder

Destination add files here.

permission

properties

→ upload then close

If we click on the file which we added then we can find a URL

⇒ Object URL

↓

Pass where we want to consume this item.

→ to access these we need to give the permissions.

So, go to Bucket first.

⇒ Permission.

→ Blocks public access.

and change the access and grant permissions.

and | Save changes |

Now, Allowing the access on content level. (or) item level.

⇒ Objects

→ Click on item.

→ make public using ACL

Now, after access is given then we can see the content using the URL

Properties:-

Versioning is present here. for that.

→ Bucket Versioning

☑ Enable.

To see the versions enable the ☑ Show versions in "objects"

If we upload a latest version then we have to give the permissions again.

→ Default encryption :-

☑ Enable

○ Amazon S3-managed keys (SSE-S3).

☑ AWS Key Management service key (SSE-KMS).

AWS KMS Key → To create our own key.

Key Managment service.

Key type.

Symmetric Asymmetric.

Key administration :- Who can manage the key (user)
(User)

Vamshi

Other AWS account :- Item level.

then [Finish]

Now, add this created key for the encryption.

Properties :-

Server access logs :-

Enable.

and then choose the path (Item).

→ To Deploy the Item (or) Launch the content using.

S3 bucket.

→ Static website hosting

☑ Enable

→ Hosting type

☑ Host website static.

→ Index document :- ex :- index.html.

↓

Launch page.

→ Error document :-

[Save changes]

Now, we can see website Domain Name. we can access the content with this.

Permissions :-

→ Bucket policy

Edit

Policy example

policy Generator.

↓

different policies can be created using this like, IAM, SNS, EBS, S3

Effect :- Allow.

principle :- *

Aws Service :- Amazon S3.

Actions :- (Give what actions you want).

Amazon resource name (ARN) :- (Bucket policy we can find 'ARN').

Add statement

Generate the policy → copy this policy

Done

Paste the policy in Bucket policy.

Object ownership :-

Access control list (ACL) :-

Grants the permission which are needed to access

then

Save changes

Metrics :- Items placed in buckets, Objects and size.

Mangement :- Manging the storage class (data).

life cycle Rules :-

Rule actions :- Current version, non current version (storing of this are not).

Storage class :- S-tandard - IA → 30 days → Intelligent Tiering → No limit days → One zone IA 30+30 = 60 days 30 days (or)

Replication Rules :-

Glacier instant Retrival → 90 days → Glacier flexible retrival → 180 days

only one zone.

60 days

Archived data

it will take hours to retrive data.

Glacier deep Archive

Hours of retrival

Automatic storage.

Name :- Keep this bucket in some other region.

in :- North Verginea

Status Rule :-

⊙ Enable

→ Choose a rule

- Apply to all

→ Destination

- Choose a bucket in this account (AWS)
- Specify a bucket in another account (AWS)

For that we have to have an account in another region Before that Disable KMS key if any is Enabled

Create bucket

Name :-

AWS Region :- Give target region

Bucket versioning :- whenever we go for replication both 'source' and 'destination bucket should be enabled with versioning.

- enable

Create bucket

Now, Buck Name | Browse

Choose the target path

IAM role

✓ choose from existing IAM role.

○ Enter IAM role ARN

IAM role.

[Create new role.]

Encryption

✓ Enable

[Save.]

→ Replicate existing objects

✓ Yes, replicate existing objects

[Submit]

Batch job

report

✓ All tasks.

path to completion report destination.

[Browse S3]

[Save]

Now,
the existing items are copied to the target bucket in another region.

Access point :-

Who can access the datasets in S3.

V.P.C :- → Networking ← 25/04/22

Virtual Private cloud :- Isolated cloud resource.

IPv4 CIDR.

ex:- 10.0.0.0 / 16 → Default 2^{32}

IP series — NO. of hosts

Here $2^{32-16} = 2^{16}$

$2^{32} = 0 = 1$ hosts

10.0.	→ Static	Address range
Class A	2^{24} Hosts	1 to 126
Class B	2^{16} Hosts	128 to 191
Class C	2^{8} Hosts	192 to 223

After the creation of V.P.C, default we will also get 1 Route table, 1 Network ACL (access control list) and 1 security group

Now, for server security we need to create subnets.

Create the Subnet.

V.P.C ID :-

Take the V.P.C that you created not the default one

IPv4 CIDR Block

Choose the same as IPv4 CIDR

10.0.1.0/24

$2^{32-24} = 2^{8} = 256$

We will get 256 Hosts.

Similarly create another subnet with name.

Private-Subnet.

10.0.2.0/24 (for private). 256 Ip's.

but we will get only 251, because first 5 are reserved.

1.0.0.0.0
10.0.0.1
10.0.0.2
10.0.0.3
10.0.0.4

} These are reserved for every subnet

* If created a subnet, How many Ip's I will get!

→ -5

Now, create a 'INTERNET GATEWAY' separately and then attach this to 'VPC' from "Actions" [attach VPC]

Now, the request can enter into V.P.C but can't access to the subnets servers because the subnets are blocking. Now to access the servers in the subnets we have to go through 'Route Table'. So, create a 'Route Table' with specified name, Now attach the 'Route Table' to 'Gateway Internet'

create route-table

→ select the route-table.

→ routes
↳ edit routes.

Add route.

ex:-

Destination	Target	Status	Propagated
10.0.0.0/16	local	Active	NO.
0.0.0.0/0	internet-gateway.	-	NO.

↓

This is for public.

save

Subnet associations.

Edit

☑ Public

☐ Private.

* Internet Gateway → Route Table → Server Public.

NOW, TO get the ip address for ip public server we need to enable

→ Subnets

→ public-subnet

→ Actions

→ Edit-Subnet setting

→ Auto-assign IP settings

☑ Enable auto-assign public IPV4 address

Now,

when creating the EC2 instance in the configuration section we need to select the V.P.C which we created

Network — Our V.P.C Name.

Subnet — Public Subnet-

Similarly for private-subnet can be created.

and for the public-subnet instance create the security from Launch wizard.

We can connect to the public servers as usual but for the private servers we can't connect normally. so, to connect to the private servers we need to connect from via 'Baston Hosts' (Public server)

Now, we need to place 'Pem' file in 'Baston Host (public) Server for that we use 'WinSCP'.

Laptop

Download.

Host name
public server IP → user name
ec2-user → login

drag and drop the pem file to right side.

Now, connected (still few steps).

change permissions.

ls -l

chmod 400 (pem file).

ls -l

Now, connected.

still Internet access is not there for private, so, here we will provide secure internet for private server called 'NAT Gate way'

NAT → Network address Translator

→ V.P.C

→ NAT Gateways.

create NAT Gateway. and this access is given to the public subnet.

→ Subnet

⊙ Public

→ Connectivity type.

⊙ public

Allocate elastic IP

Create NAT gateway

Now, this is created under public.

~~Nat~~ Gateway.

→ Route - Table.

select one private route table.

Routes

Edit routes

Destination	Target	status	Propagated
10.0.0.0/16	local	Active	NO.
0.0.0.0/0	Nat-Gateway.		

↓

Elastic Ip will look like

Save

→ Subnet associations

→ Edit Subnet associations

☑ private-subnet

Save

Private → Nat gateway → Route Table → Subnet associations.

"Network ACL" this will block the request coming on the servers which are inside subnets.

(ACL - Access control list)

(1) Network ACL is default created with V.P.C.

Now, we have to allow the traffic.

→ By default they will Deny the access to the servers placed inside subnets.

Create NACL

V.P.C (created V.P.C)

Create

Now, on created NACL.

→ Inbound rules.

→ Edit

Add rules.

→ similarly create the outbound rules.

→ Edit Subnet associations

☑ public subnet.

→ NACL :- Apply routing on Subnet level (Group) (Allow/Deny) option

→ Security Group :- Apply routing server level (single) No option.

Important components :-

→ V.P.C
→ Subnets
→ NACl
→ NAT gateway
→ Internet gateway
→ Route-Tables
→ security groups

We, can also have V.P.C logs.

Flow logs.

[Create]

Filter - All.

Destination - Cloud watch (attach here).

cloud watch

log

log group

create.

we can IAM role

create policies and roles.

giving cloud watch access.

Now, to analyse my V.P.C we have V.P.C flow logs.

V.P.C endpoints are used to connect to S3 buckets without Network gateways and internet.

26/04/22

→ V.P.C Peering.

TO connect V.P.C (2) in same region, in different account are different region.

→ Route 53:-

DNS:- Domain name service.

→ TO Map (or) associate Domain name to the server.

ex:- GoDaddy → Paid service.

AWS → Paid service. (Route 53)

↓

Route 53.

Domain registration

Freenom.Dom → Free domain service.

↓

register and get desired name

ex:- abc-zn.tk.

↓

Route 53

↓

DNS management

↓

Create hosted zone

Domain name - abczn.tk

Public hosted zone

After creating the hosted zone NS (Name server) and (SOA) A start of Authority.

Now, Take this name server 'Names' and copy somewhere

Now, create record (A records)

→ Record Name ---- blog abczn.tk.

→ Value : IP address form EC2

TTL (second) (Time to leave). Routing policy.

300 Simple routing.

create record

Now, similarly give blog (www) for that create another record.

Here we created two records without www and with www (abczn.tk)

Now, place this "NS" (Names) in the domain manager in "freenom" configure with names in 'AWS' Route 53

Different types of routings :-

1 * Simple Routing policy.

→ Here there's no load balancing ? it will send request to servers randomly.

2 * Weighted Routing policy :-

Here, we can have servers on different regions and one region has high configured server and other has low config.

we will give more weight of load to high config server min to low weighted server

3* Latency Routing policy :-

Redirect to the server that has the least latency close to Region.

Request will go to the nearest (region) servers.

4* Failure over routing :-

we can keep servers on different regions and send the request to the servers with good health but, by default we need to set the primary and secondary servers.

→ Route 53 will also check the Health of Servers (Health check).

→ we can set the health check by providing

Protocol
Ip address
Host name
port
path

Geo Location Routing Policy:-

on the country level
if we get request from particular country we will coping with the nearest server.

multi-value Routing policy:-

it is similar to Load Balancer.

Route 53 Routing policies:-

→ Simple routing
→ weighted routing
→ Latency
→ Fail over
→ Geo Location
→ multi-level

Route 53.

Hosted Zones.

Value

Alias.

↓

Choose endpoint

in this we can choose Load Balancer.

} we can route the request not only to server Ip, but route the request to different endpoints (VPC ELB etc)

** Server are placed in different regions; now I want to route traffic to different regions?

A:- Route 53.