Auto scaling

Monitor existing resources utilization and when certain thresholds are breached to take action.
Ex. CPU utilization more than 80% -> add instance trigger,
When CPU utilization is less than 35% -> remove instance.
Cloud watch alarms are used for the real time monitoring and helps autoscaling to trigger the action that needs to be taken.
Use launch configuration to define the configuration of the EC2 instances that you will need to spawn out.

Launch template is the new version of Launch Configuration.
It is very similar to the 7 steps process of EC2 creation.
Network card - can be made the same even when the underlying EC2 changes So that the same IP gets assigned to the EC2 instance which ever is created by the auto scaling group.

ASG can be made available across different AZs. So it can handle EC2s from different AZs.

Any modification to the Launch template can be changed and another version of the launch template will be created. So modification to the instances in the autoscaling group is performed by making changes to the launch template.

Instances types and weightage can be specified. Like adding x number of On-Demand instances and adding more instances from spot for cost effectiveness.

Scale-in prevention will hold the max number instances rolled out. Else scale in will remove the instance with the load decrease.

Schedule scaling is possible.
Step scaling - with the traffic/any parameter in monitoring we can define the number instance to be added as scaling. Bulk scaling(step scaling according to the configuration.
Version upgrade can be done with ELB holding the front with few working instances and asg (auto scaling group) on the background changes the server from 1 to 2.

Percent of scale / instances scale - any number of instances or the number of instances which equals to the percent of of the

Activity tab will maintain all the action taken by the Asg, instance removed/added modified etc.

It is self healing and creates a new instance to maintain to desired instance count(or to manage the load)

Amazon Machine Image or AMI

Create an image from an instance to create a snapshot(backup) of an AMI.
Region specific
The base configuration is available on the image and hence the AMI can be used inside in asg to launch similar instances with the same configuration and services running on.
Transfer from one region to another is possible thru ‘Copy AMI’ option. But incurs cost according to the size of the AMI.
Before deleting the snapshot of AMI you have to deregister the AMI from the snapshot

Note: When we manually create snapshots of a running instance, it restarts the EC2 instance

AWS Command Line Interface

Shell script to communicate with AWS cloud.
Gives way into Infrastructure as code.
boto3(python)
Configure with -name, access ID, Secret key, region (output type)
Can create/rename/delete/edit resources from remote user PC/laptop when the IAM has granted the user access and access key to use CLI.
Boto3 (python) scripts can be written and executed thru CLI terminal to programmatically design, maintain the infrastructure

Forms of Storage on Cloud

Instance store(EC2)
Present inside the EC2 server. Lost on termination unless saved specifically on some other form of storage.
EBS
Added external harddisk fixed to one server at a time.
EFS
Added shared storage can connect with multiple servers(EC2) at the same time.
S3
Used from CLI/scripts, Lambda(events, SQS), cloudfront, EC2, other managed services and default amazon storage location

Block Storage - EBS

EBS volume and EC2 instance needs to be in the same region to be able to attach.

Snapshots needs to be created when the disk is unmounted(not in use)
Else there is an option for the snapshot to lose the transactional data.(Any copy or paste action going on in the EC2)

Network Storage - EFS Concepts

Same mount point across all the AZs.
Performance mode needs to be defined before using the EFS. Else migration has to be done for better performance.
Use the credits for faster processing similar to EC2.
Many EC2 instances can connect to it simultaneously.
EFS port number - 2049
EFS utilized one IP form the subnets CIDR block.
Security group needs to be opened for 2049 to utilize and access EFS.
The Storage can be encrypted by default.
Lifecycle management can be configured.
Policy can be assigned to restrict access to write, read and root access.
Locking a file for writing needs to be handled by the application. EFS does not lock for writes.

Object Storage - Simple Storage Service

1) Object or block based store.
2) Unique name for buckets.
3) Soft limit is present at 100 - can be increased based on request to AWS
4) max object size - 5TB / small chunk uploads in 5GB - multipart upload.
5) Object added thru http put operation
6) Version to track all the objects that have been stored.
7) Lifecycle management - out of the box
8) availability - 99.999999999% (Stores 10 copies of your data)
9) Region to region copy can be done by user
10) CFN(Cloudfront)-Delivery and cache -> S3 and CFN sync and server content seamlessly.
11) Static websites can be launched with S3 and CFN. Without EC2 at all.
12) Static part of the web application can be put on S3. load on the application server decreases.
13) Eventual consistency for existing files. Read after consistency for over-writes.
14) time for consistency depends on the size of the file.
15) Name of the bucket and **Key** - path of the file.
16) Encryption - aws managed key or custom key can be used.
17) Cost saving - infrequent access, less redundant, glacier (very infrequent access(for audit log))

Additional features:

1) 1000 buckets hard limit per region.
2) 3 to 63 character names.
3) can copy access policy from other bucket configurations.
4) Object lock (write lock) can be enabled or disabled while creating the bucket.
5) S3 is a global service but managed region wise.
6) Copying to a cross region is possible thru replica creation.

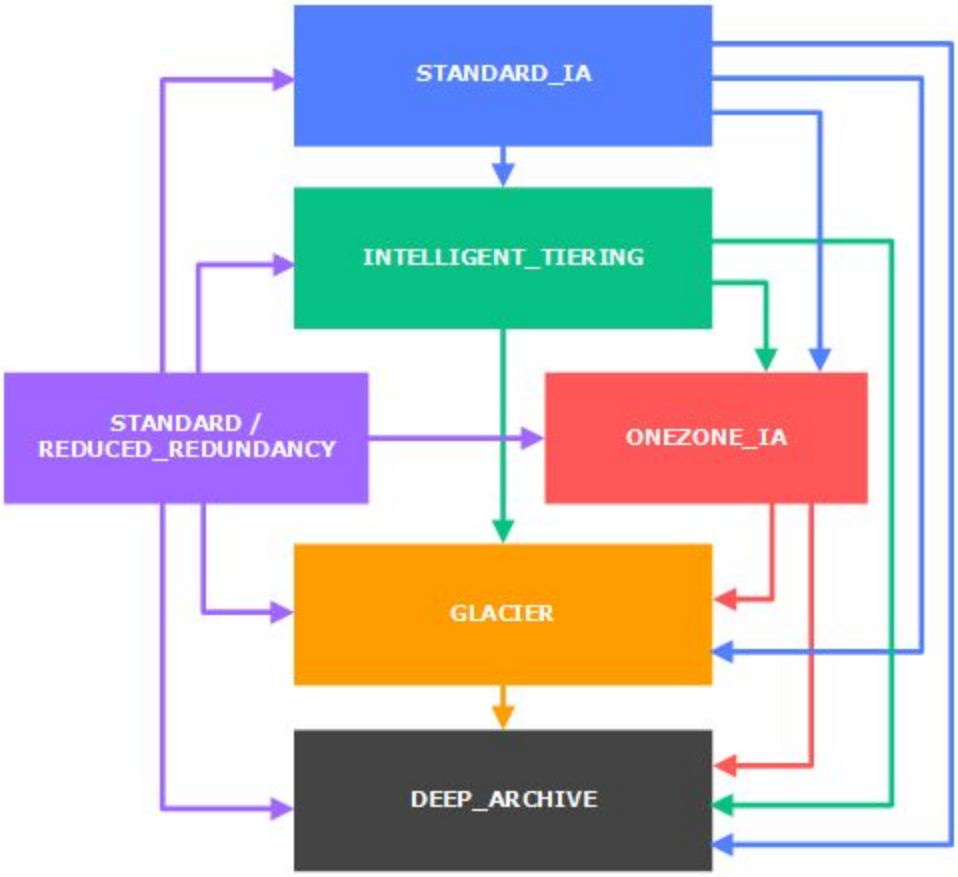
Properties:

1) Types of storages classes in S3:
2) Standard, Standard -Infrequent Access(IA), one -zone IA, Reduced Redundancy, Intelligent Tiering, Glacier, Glacier Deep archive.
3) Access Control list ACL to control the access to S3
4) Performance can be increased if using one bucket with many folder(prefixes) rather than using separate buckets for grouping the data
5) Server access audit trail to track all the access activities done on the S3.
6) AWS cloud trail data event - to track the user action done on S2 put/get/etc.
7) Event notification - change in S3 can be triggered as an event to destination including AWS services -> Lambda, SNS and SQS. Events like delete/put/get/copy/upload job completion.
8) Transfer acceleration - update and fetch at the same time and does faster fetches. Using cache from AWS backend.
9) Requester pays - Enables the user of the bucket to pay for the upload and storage cost rather than the owner of the bucket.

Lifecycle management and replica:

1) Workflow can be created for object transition from one class to another of storage using lifecycle management.
2) The whole cycle can be managed from inside S3 and all the configuration is guided and based on the number days it should be available on one class of the storage and finally It can be mapped for expiry(remove of the objects from the system)
3) Enable delete marker for deleting the object from the replica bucket to replicate the deletion of the object in replicated S3 buckets.
4) Replication enabled after few objects are present in the bucket - will not copy the already existing objects to the other bucket. Can only be done with an AWS support request or a manually executed program to do the copying.
5) Versioning can enabled and versioned objects can be viewed by enabling ‘List versions’
6) Deleting the files in the replica bucket cannot affect the original bucket. So we need to manually maintain the sync between the buckets if deletion activity is frequent in your architecture.

Possibility of combinations the lifecycle can be implemented:



Permissions to accessing Bucket:

1) Use a policy and restrict or grant access to any/all users according to the need.
2) CORS - Cross Origin Resource Sharing - resource from AWS services/programs to access the S3 bucket internally.
3) Access Points allow a whole set of resources, users and codes to access the bucket. If more number or unlisted users are going to access the bucket (from a department/3rd party/business) then endpoints are the best option to use.

Other options:

Batch operations, access analyzer, Storage lens(cross account view, only for analysis)

S3FS-FUSE -> open-source FUSE plugin to mount S3 as file system in the EC2 system/local system