Contents

[**IAM** 3](#_Toc46801145)

[Key concepts 3](#_Toc46801146)

[Using Roles to access AWS more securely 3](#_Toc46801147)

[**S3 – Simple Storage Service** 4](#_Toc46801148)

[Features: 4](#_Toc46801149)

[Storage classes: 4](#_Toc46801150)

[Charges: 4](#_Toc46801151)

[S3 Security and Encryption 5](#_Toc46801152)

[Versioning – typical versioning 5](#_Toc46801153)

[Locking policies 5](#_Toc46801154)

[S3 Performance 6](#_Toc46801155)

[S3 Select 6](#_Toc46801156)

[AWS Organisations 6](#_Toc46801157)

[Sharing S3 buckets between accounts 6](#_Toc46801158)

[S3 Cross region replication 6](#_Toc46801159)

[Transfer Acceleration 6](#_Toc46801160)

[Data sync 7](#_Toc46801161)

[CloudFront 7](#_Toc46801162)

[Signed URLs and Cookies 7](#_Toc46801163)

[Snowball 8](#_Toc46801164)

[Athena vs Macie 8](#_Toc46801165)

[Storage Gateway 8](#_Toc46801166)

[**EC2 – Elastic Cloud Compute** 9](#_Toc46801167)

[Pricing 9](#_Toc46801168)

[Instance types 9](#_Toc46801169)

[Security Groups 10](#_Toc46801170)

[EBS 10](#_Toc46801171)

[Moving EBS volumes / EC2 instances to another AV 10](#_Toc46801172)

[Moving EBS volumes / EC2 instances to another region 10](#_Toc46801173)

[Snapshots 11](#_Toc46801174)

[AMI types 11](#_Toc46801175)

[ENI vs ENA vs EFA 11](#_Toc46801176)

[Encrypted root volumes and snapshots 12](#_Toc46801177)

[Spot Instances & Spot Fleets 12](#_Toc46801178)

[EC2 Hibernate 12](#_Toc46801179)

[CloudWatch & CloudTrail 13](#_Toc46801180)

[AWS Command Line (CLI) 13](#_Toc46801181)

[Bootstrap Scripts 13](#_Toc46801182)

[Instance metadata 13](#_Toc46801183)

# **IAM**

* New users appear without any rights

## Key concepts

* Users
  + Basically the different accounts
  + Can have two sets of login details – aws secret & key, and username and password
    - Username + password is for console access
    - Secret & key is for programmatic access
* Groups
  + Groups of users. Used to categorise bundles of users (say, developers, accounting, etc…)
* Policies
  + Basically rules. Rights to do this and that.
  + In aws, anything that is not explicitly allowed is implicitly forbidden.
* Roles
  + Compilations of policies (think groups for policies instead of users)
  + Can be attached to stuff such as ec2 instances for example

## Using Roles to access AWS more securely

Add a role – go to the EC2 insance -> Actions -> Instance Settings -> Attach/Replace IAM role -> Select the role

Allows us to interact with AWS resources (as per the policy of the role) from the AWS CLI without the need to pass along credentials. Can be assigned via the console or CLI and applies globally.

Why? – Easier to manage than credentials on each EC2 instance, more secure (as they can be fine-grained better than accounts, and also if an EC2 gets hacked, it remains the only point of entry).

# **S3 – Simple Storage Service**

Globally accessible (but locally stored), object-based (think files, like photos, movies, documents, etc) storage service. Organised in buckets (basically folders). No min file size, max 5TB. Due to it being globally acessible, an s3 bucket needs a globally unique name (as it gets accessed via its url).

Super high durability – 99.999999999% durability, 99.9% availability

When uploading a file, if the upload was successful you’ll get a http 200 code

S3 objects consist of:

* Key (the name of the file)
* Value (the content of the file, the sequence of bytes)
* Version ID (in case of multiple versions)
* Metadata
* Subresources (ACL’s, torrents)

### Features:

* Tiered storage
* Lifecycle management (say, when this file is 30 days old, move it to Glacier Deep Archive)
* Versioning
* Encryption
* MFA (Multifactor authentication) delete
* Control access to data via ACL’s and bucket policies

### Storage classes:

* Standard (99.99% availability, 11x9 durability, can sustain the loss of two facilities concurrently)
* Infrequent Access (IA) – for data that you access infrequently, but when you do you want it to happen fast. Lower fee at rest, but has a retrieval fee
* One Zone IA – Same as above, but lower availability/durability
* Intelligent Tiering – will see how you use the data and change its class based on usage
* Glacier – lower cost, higher retrieval time (configurable, from minutes to hours), retrieval fee
* Glacier Deep Archive – very low cost, very high retrieval time (12h), retrieval fee

### Charges:

* Storage (GB/month)
* Requests
* Per tier
* Data transfer
* Transfer acceleration (making use of Cloud Front)
* Cross region replication

Remember for exam – MFA auth for delete, Read after Write for PUTS of new Objects, Eventual Consistency for overwrites or deletes (can take some time to propagate due to the replication)

### S3 Security and Encryption

Can config the bucket security via bucket policies as well as access control lists (ACLs). Bucket policies are for the whole bucket, while the ACLs are more granular and can reach individual file level

Can also be configured to create access logs that tracks all the requests made to that bucket. Those logs can then be sent to another bucket, even one in a different account. They can be server level (free for the logging, normal storage fees for the space taken by the logs themselves) or at the file level (through CloudWatch, fees apply)

Two types of encryption – in transit (SSL/TLS, think HTTPS) and at rest

* At rest server side – i.e. you upload an object and Amazon encrypts it
  + S3 Managed Keys – SSE-S3 (Server-Side Encryption S3) (keys come from Amazon)
  + AWS Key Management Service, Managed Keys – SSE-KMS (shared)
  + Server-Side Encryption with Customer Provided Keys – SSE-C (keys come from you)
* At rest client side – i.e. you encrypt the object and upload it to Amazon

### Versioning – typical versioning

* Once enabled, can only be suspended (paused). Cannot be disabled, and previous versions persist unless deleted (directly or via lifecycle rules). Also supports MFA delete (remember for exam)

### Locking policies

* S3 Object Lock – WORM (write once, read many) model. Idea is to pin the object and ensure it will not get changed for a while (or forever) after being uploaded. Can be applied at object level or at bucket level.
  + Governance mode – locks it for most users
  + Compliance mode – locks it for EVERYONE, INCLUDING ROOT. Be very careful when setting up the retention period of compliance mode locked objects
  + Legal hold – works like a retention period, but it does not expire automatically. It can, however, be applied and removed at will
* Glacier Vault Lock – basically the same thing, but for vaults inside of Glacier (as opposed to objects in S3). Also does not have the modes?

### S3 Performance

* Prefixes (i.e. folders in the bucket)
  + Supports up to 3500 put / 5000 get requests per second, per prefix (that is, folder in the bucket). So if we want to go above those limits, we should spread our files across more folders in the bucket.
* KMS request rates
  + When using SSE-KMS, we call GenerateDataKey (when uploading) and Decrypt (when downloading), so that does also add limitations
  + Quota is region specific, either 5500, 10 000, 30 000 requests per second
* Multipart uploads
  + Recommended for files above 100 MB
  + Required for files over 5 GB
  + Breaks up the object into chunks and uploads the diff chunks in parallel
  + Download equivalent is Byte-Range Fetches

S3 Select

* Using a SQL query to only download the subset of data that we need from out objects, as opposed to downloading the entire file and the extracting what was necessary
* Glacier Select – same but for Glacier

AWS Organisations – consolidate multiple AWS accounts. Centralised management, consolidated billing (very useful for getting into higher usage tiers for stuff like s3, so getting the volume discounts)

* Root account – enable MFA, use complex password, only use for billing (don’t deploy any resources there)
* Enable/disable services on account level by utilizing policies applied to OUs (Organisation Units) and putting the accounts in different OUs

### Sharing S3 buckets between accounts

* Bucket policies and IAM at the entire bucket level (programmatic only)
* Bucket ACLs and IAM at the object level (programmatic only)
* Cross-account IAM roles (programmatic & console)

### S3 Cross region replication

* Versioning must be enabled
* Replicates files added after versioning was put in place, but not ones that were there since before that
* New files and changes do get replicated, deletes do not

### Transfer Acceleration

* Uploading to an edge location instead of to the S3 bucket directly. Thus making use of the lower ping
* Done through a distinct url
* Can actually be a tad slower if the main region’s AVs are super close to you

### Data sync

* Synching data between on-prem and AWS (can be done hourly, daily, or weekly)
* Need to install an agent on-prem
* Synchs with S3, FSx for Windows, and EFS
* Mostly used to move large amounts of data from on-prem to AWS
* Needs NFS or SMB compatible file systems
* Can also replicate EFS to EFS (so, from one place in the cloud to another)

### CloudFront

* A Content Delivery Network (CND) utilizing edge locations to reduce latency and increase accessibility to data for end users. The data can come from S3, EC2, ELB, or Route 53
* Distribution – the collection of edge locations we are utilizing. We can have a web distribution, which is used for websites, and an RTMP distribution usually used for media streaming (deprecated Dec 31, 2020)
* Edge locations are not read only, they can also be written to (like with transfer acceleration)
* Objects are cached for the TTL (Time To Live), which is configurable
* You can clear cached objects (called invalidating the cache), but you will be charged (useful if you’ve changed smth on the data, but its old version is still cached on the edge location and delivered to the customers)
* Can restrict access to the data using signed urls and cookies (think all the online magazines behind a paywall)

### Signed URLs and Cookies

* Used to provide access to restricted resources (cloudfront & S3)
* Use URLs for individual files (1 file = 1 URL)
* Use Cookies for multiple files (1 cookie = multiple files)
* Those signed URLs/Cookies include a policy with
  + A URL expiration, IP ranges, Trusted signers (which AWS accounts can create signed URLs)
* In the case of CloudFront the user accesses CloudFront, which then accesses the data via OAI (Origin Access Identity). In the case of S3, the user accesses the data directly. The S3 signed URLs have a limited lifetime

Snowball – a petabyte scale data transport solution (basically a cool, huge, portable disk). Used to enable data transfer without involving the internet, in a more old-fashioned mail form. Receive the snowball by courier, put all the data on it, send it to Amazon back with another courier.

* Comes in 50 & 80 TB versions. Very secure (both physical protections and 256 bit encryption)
* Snowball Edge – 100 TB, comes with on-board storage and compute capabilities. Basically a portable mini cloud (can ensure your apps run without access to the cloud)
* Snowmobile – Exabyte-scale data transfer service. A huge data container, housing up to 100 PB each, pulled by a truck

### Athena vs Macie

* Athena
  + Query service enabling us to analyse and query data stored on S3 using standard SQL. Good for analytics
  + Pay per query / TB scanned
* Macie
  + Machine Learning powered service that scans through our S3 data and looks for sensitive (as in, personally identifiable) information. Security service
  + Includes Dashboards, Reports, and Alerting

### Storage Gateway

* Connects on-prem software to cloud-based storage. Can be a virtual device or a physical one
* It’s a VM image to be installed on a host in the data center (supports Microsoft Hyper-V and VMware ESXi) and associated with your AWS account
* Three types:
  + File gateway (NFS & SMB)
    - Store files in S3 buckets, accessed through a NFS (Network File System) mount point
    - Once the files are in S3, they are just like all other S3 files (as in, all the S3 features and policies can be applied to them – versioning, lifecycle, CRR)
  + Volume Gateway (iSCSI – Stored & Cached Volumes) – used to store copies of (virtual) hard disk drives
    - The data on the volumes is stored in S3, while snapshots of the volumes is stored on Amazon EBS. Act like other snapshots – incremental backups, capture only changed blocks
    - Stored volumes – primary data stored locally on the storage gateway, and then replicated on AWS. Done so that entire dataset is available on-prem for low-latency access, and AWS is just backup
    - Cached volumes – unlike with stored volumes, not the entire dataset is kept locally, but just the most frequently accessed elements. Again, everything is replicated on AWS
  + Tape Gateway (VTL – Virtual Tape Library)
    - If using tapes, this can leverage the existing tape infrastructure. Basically lets you create virtual tape cartriges on the cloud on S3

Note – check onto the different urls (virtual hosting, path, static website).

# **EC2 – Elastic Cloud Compute**

EC2 – resizable compute capacity in the cloud.

## Pricing

* On demand
* Reserved
  + 1-3 years, no/partial/full upfront pay
  + Scheduled - only available during a certain time of the day
  + Convertible – can upgrade at certain point (only upwards)
* Spot
  + Making use of unrequired capacity. Much cheaper than a standard instance, but can be terminated at any point if someone outbids us.
  + Note – if terminated by EC2, you get charged only for the time you used the instance. If terminated by yourself, you get charged for the full hour.
* Dedicated hosts
  + All machines on the host are ours. Good for compliance.

Instance types mnemonic – Fight Dr McPxz AU

* F – FPGA (big dick shit, allows for the very chips to be reprogrammed. Think genomics research)
* I – (IOPS) - High Speed Storage (NoSQL, Data Warehousing)
* G – Graphics intensive (video encoding, gaming, 3D app streaming)
* H – High Disk Throughput (distributed file systems)
* T – Lowest cost, general usage
* D – Dense Storage (???)
* R – (RAM) Memory Optimised
* M – General purpose, higher cost (T’s big bro. Meant for heavier, more consistent workloads – no CPU burst credits unlike T’s system)
* C – Compute optimised
* P – Graphics/General Purpose GPU (Machine Learning, Crypto mining)
* X – Memory Optimised
* Z – High compute capacity / high memory footprint
* A – Arm-based workloads (???)
* U – Bare metal (physical server, avoid the overhead of hypervisor and VMs)

Note – can encrypt root device volumes (e.g. /dev/xvda for Linux) from the get-go (for default AMIs).

Termination protection is off by default. On an EBS-backed instance, the default is for the root volume to be deleted upon termination, but for additional volumes to not be deleted.

## Security Groups

Here we set the rules about accessing the instances (basically firewall settings – controlling ports). All inbound traffic is blocked by default. All outbound is allowed (hmm?). Rule changes take effect immediately.

Rules are stateful (hmmm?). If you create an inbound rule (e.g. open port 80 for requests from 0.0.0.0), an equivalent outbound rule is created automatically (i.e. open port 80 to return requests to anyone). Contrast with NACLs (Network Access Control Lists, VPC), which are stateless and we need to create an outbound rule for every inbound rule manually.

Can only allow things, cannot block (e.g. cannot block a specific IP). To blacklist things we need NACLs in the VPC settings.

Can attach multiple security groups, thus opening up everything from them (up to 5 by default, up to 16 by request). Can also attach any number of instances to a given security group. Max 60 rules (inbound or outbound, 120 total) per security group.

## EBS

EBS – Elastic Block Store (basically the VM’s VHD). Persistent Storage (survives stopping an instance, can even remain after instance termination). Automatically replicated within its own AV to provide high durability.

5 types

* General Purpose SSD (**gp2**) – balanced, standard
* Provisioned IOPS SSD (**io1**) – for databases
* Cold HDD (cheapest) (**sc1**) – cheapest, for infrequent access (file servers)
* Throughput Optimised HDD (**st1**) – Data Warehouses
* EBS Magnetic (**standard**) – old gen HDD. Up to 1TiB only (rest are up to 16)

IOPS vs Throughput – The former measures operations handled per second, whereas the later measures total bits handled per second (so, for larger operations)

EBS volumes should be in the same AV as their related EC2 instances (this is the default setup) – this is the only rational option, as otherwise there will be a lot of lag.

### Moving EBS volumes / EC2 instances to another AV

1. Create a snapshot of the root EBS volume
2. Create an image (AMI) from that snapshot (best to use HVM virtualisation type)
3. Create a new EC2 from that image, and select the desired new AV in the launch options (3 – Configure Instance Details – Subnet)

### Moving EBS volumes / EC2 instances to another region

1 and 2 – Same as above

1. Copy that image to a new region (Images -> AMIs -> Actions -> copy AMI -> new destination =region)
2. Use that image to create a new EC2 instance in that region

## Snapshots

Snapshots are point in time copies of EBS volumes (basically, the volume as it was at that time). They are incremental – only the changes since the last snapshot are saved (think .bak vs .trx on MS SQL). It is best practice to stop the instance before taking a snapshot of the root volume (also possible to do it with the instance running though). The first snapshot might take some time to create. We can create AMIs from snapshots.

Volumes exist on EBS, snapshots on S3. Can change EBS volumes on the fly, without stopping the instance (but might need to re-partition the file system of the instance to make use of the newly allocated space). This includes type (e.g. promote to Provisioned IOPS – io1) and size.

AMI types – EBS (external, persistent) vs Instance Store (internal, physical, ephemeral)

EBS – external volumes that attach to an instance. Practically independent of the instance, so they persist after shutdown (as long as we still pay for them).

* For EBS backed AMIs, the root volume is launched from an EBS snapshot

Instance store – physically attached to the instance, so they are ephemeral – they get deleted once the instance is terminated (not rebooted, but terminated. It cannot be stopped). This is so that the instance can be used by someone else while we aren’t using it

* For Instance store backed Amis, the root volume is launched from a template stored on S3

## ENI vs ENA vs EFA

ENI – Elastic Network Interface – **basic**ally a virtual network card. It allows for:

* One primary and some secondary private IPv4 addresses (from the VPC range)
* One elastic IPv4 address per private IPv4 address
* One public IPv4 address
* One or more IPv6 addresses, one or more Security Groups
* A MAC address, source/destination check flag, description

ENA – Enhanced Networking

* Utilizes single root I/O virtualisation (SR-IOV) to increase networking performance
* Helps with reducing latency, higher bandwidth
* No additional charge (but the EC2 instance has to support it)
* Can enable via ENA (Elastic Networking Adapter, **up to 100 Gbps**) or VF (Intel Virtual Function Interface, up to **10 Gbps**, used with older instances)

EFA – Elastic Fabric Adapter

* For **High-Performance Computing (HPC)** and **Machine Learning**. Faster than TCP communication, lower and more consistent latency. Can use OS bypass (Linux only).

## Encrypted root volumes and snapshots

New way of encrypting root volume – select encrypted as an option when creating the ec2

Old way (or when wanting to encrypt after having launched the ec2 already) – go to volumes, actions -> create snapshot, snapshots, actions -> copy, tick ‘Encrypt this snapshot’. Then, select copied snapshot, actions -> create image. Then launch from that image.

Snapshots of encrypted volumes are encrypted automatically – cannot launch an EC2 with an unencrypted root volume from their AMI. Same for volumes restored from encrypted snapshots.

Only unencrypted snapshots can be shared with other AWS accounts or made public.

## Spot Instances & Spot Fleets

Useful for things like big data, containerised workloads, CI/CD, web services.  
Don’t use for mission critical or persistent workloads or for databases.

Spot request – one time vs persistent.

* One time – if spot price goes beyond max price stipulated, instance gets terminated and that’s that.
* Persistent – if spot price goes beyond max price stipulated, instance gets terminated. Then, if it goes back down again, instance gets rebooted for as long as the price is under max price given.

Spot fleet – collection of spot instances (and optionally on-demand instances). So it tries to launch the requested number of spot instances, but it if can’t (say, there isn’t enough available capacity), it launches as many spots as it can and also some on-demands to fill in the remainder.

Spot Block – have a spot instance run for a finite duration (1 to 6 hours).

## EC2 Hibernate

Hibernate saves the current RAM contents to the EBS root volume. That way it boots much faster. Also, maintains the same instance ID. Requires that the root device volume is encrypted, there is enough free space on the root volume to fit the ram (and the ram must be less than 150GB), and it needs to be enabled upon ec2 launch. Cannot hibernate for more than 60 days.

Useful for processes that are long-running or take a long time to initialize.

## CloudWatch & CloudTrail

CloudWatch - monitoring service – Compute (EC2, Autoscaling Groups, Elastic Load Balancers, Route53 Health Checks), Storage & Content Delivery (EBS Volumes, Storage Gateways, CloudFront), etc…

Metrics are usually:

* CPU
* Network
* Disk
* Status Checks (Hypervisor – is it running, EC2 instance)

By default it monitors EC2 stuff every 5 min, can switch to every 1 min with detailed monitoring. Can also create CloudWatch alarms that trigger notifications, as well as dashboards, events, and logs.

Setting up a CloudWatch alarm:

* Recommended - Enable CloudWatch detailed monitoring.
* CloudWatch -> Alarms -> Create Alarm. Select Metrics, details and actions (as in – what to look for, datapoints (e.g. when 3 of the last 4 datapoints fulfill the criteria), what to do when the conditions are met – send a notification for example).

CloudTrail – logging service. Logs Console actions and API calls (what about CLI?). Can see who (users, accounts, IP’s), when (and what?).

## AWS Command Line (CLI)

Need an account’s access key id and secret access key in order to use the AWS CLI.

aws configure -> enter key id, secret key, region, output format

## Bootstrap Scripts

Basically a shell script to run upon instance creation -> !#/bin/bash

## Instance metadata

Information about the instance – e.g. IP address, instance ID, credentials, AMI, network, security groups, etc…

Find it at 169.254.169.254. For example, from within the EC2 –

* curl http://169.254.169.254/latest/meta-data/local-ipv4