Amazon S3

01. S3 Overview

1. L'origine et Le besoin

Lancé en **2006**, Amazon S3 est l'un des tout premiers services proposés par AWS. L'idée d'Amazon était de créer un système de stockage "infini", ultra-disponible, et accessible de n'importe où via de simples requêtes HTTP (via une API Web).

Avant le stockage objet, on gérait des serveurs de fichiers traditionnels (NAS/SAN) ou des disques durs (stockage de blocs comme AWS EBS). Les problèmes ? C'est difficile à scaler, cher, et limité en capacité. S3 répond au besoin de **stocker des volumes massifs de données non structurées** (images, vidéos, fichiers de logs, sauvegardes, fichiers statiques de sites web) de manière :

- **Hautement disponible et durable** (conçu pour une durabilité de 99,999999999 % - les fameux "11 neuf").
- **Scalable horizontalement** : Tu n'as jamais besoin de prévoir la taille du disque. S3 grossit à l'infini avec tes données.
- **Économique** : Tu ne payes que ce que tu consommes réellement.

---

2. Différence entre Objets et Buckets

S3 est un service de **stockage d'objets** (et non un système de fichiers classique).

- **Bucket (Le conteneur) :** C'est l'équivalent d'un dossier racine ou d'un "lecteur". Tout objet dans S3 doit résider dans un bucket. C'est au niveau du bucket que l'on configure la région AWS, la sécurité (chiffrement, politiques d'accès), et les options comme le versioning.
- **Objet (La donnée) :** C'est le fichier que tu déposes. Contrairement à un fichier sur un disque dur, un objet est une entité autonome qui contient :
  - La donnée elle-même (la séquence de bits du fichier).
  - Des métadonnées (taille, type, date de création, etc.).
  - Un identifiant unique (la clé).

---

3. Le Fullpath : Préfixe vs Nom d'objet

Dans S3, **les dossiers n'existent pas physiquement**. C'est une architecture totalement plate (*flat*). L'illusion des dossiers est créée par la structure du nom de la clé (le *Key Name*).

Prenons l'URL (ou fullpath) suivante : `s3://mon-bucket-examen/images/2026/photo.jpg`

- **Bucket :** `mon-bucket-examen`
- **Key (Nom complet de l'objet) :** `images/2026/photo.jpg`

- **Prefix (Préfixe) :** `images/2026/`

La console AWS va utiliser les barres obliques (`/`) comme délimiteurs pour t'afficher une interface avec des dossiers cliquables, mais pour S3, il s'agit simplement d'une chaîne de caractères unique.

---

4. Le Naming : Global vs Account-Regional Namespace

Là encore, bravo ! C'est une nouveauté majeure introduite par AWS début 2026. Tu as désormais le choix entre deux modèles d'espace de nommage (*namespaces*) :

A. Le modèle Historique : Shared / Global Namespace

C'est le mode par défaut que tout le monde connaît. Le nom du bucket doit être **unique au monde**, tous comptes AWS confondus. Si quelqu'un a pris `mon-bucket-data`, tu ne peux pas l'utiliser.

B. La Nouveauté 2026 : Account-Regional Namespace

Pour éviter la "guerre" des noms de buckets et simplifier l'automatisation (Infrastructure as Code), AWS permet maintenant de créer des buckets dans un espace de nommage propre à ton compte et à ta région.

- **Le principe :** S3 ajoute automatiquement un suffixe obligatoire à la fin de ton nom de bucket comprenant ton **ID de compte AWS** et la **Région** (ex: `mon-bucket-123456789012-us-east-1-an`).
- **L'avantage :** Grâce à ce suffixe géré par AWS, tu as la garantie que le nom est disponible et aucun autre compte AWS externe ne pourra te "voler" ce nom ou utiliser ce suffixe. Tu peux ainsi réutiliser la même structure de nom de manière prédictible d'un environnement à l'autre (Dev, Staging, Prod).

---

5. Taille maximale d'un objet (Max object size)

- La taille totale d'un bucket est **illimitée**.
- La taille maximale d'un **seul objet** dans S3 est de **50 To (Terabytes)**.
- La taille maximale pour un upload classique en une seule requête HTTP (Single PUT) est de **5 Go (Gigabytes)**.

---

6. Multi-part Upload (Téléchargement en plusieurs parties)

Si tu veux uploader un fichier qui dépasse 5 Go, tu **dois** utiliser le Multi-part Upload. Cependant, AWS recommande fortement de l'utiliser pour tous les fichiers dès qu'ils dépassent **100 Mo**.

Comment ça marche ?

Le fichier est découpé en plusieurs morceaux (*parts*) qui sont envoyés en parallèle vers S3. Une fois toutes les parties reçues, S3 les réassemble pour recréer l'objet final.

Les avantages pour l'examen :

1. **Meilleure bande passante :** Les morceaux sont envoyés en parallèle.
2. **Résilience aux pannes réseau :** Si l'upload du morceau 4 échoue sur un fichier de 50 Go, tu as juste à renvoyer le morceau 4, pas tout le fichier.
3. **Pause et reprise :** Tu peux mettre l'upload en pause et le reprendre plus tard.

> ⚠ **Alerte SAA-C03 :** Les morceaux non réassemblés (en cas d'upload abandonné) restent stockés dans S3 et te sont facturés ! Pour éviter cela, la bonne pratique est de configurer une **S3 Lifecycle Policy** (politique de cycle de vie) pour supprimer automatiquement les *Incomplete Multipart Uploads* après quelques jours.

---

7. Les Métadonnées (Metadata)

Chaque objet possède des métadonnées sous forme de paires **Clé/Valeur**. On en distingue deux types :

- **System Metadata (Métadonnées système) :** Gérées par AWS (ex: `Date de création`, `Taille`, `Chiffrement`, `Content-Type` comme `image/jpeg`).
- **User-Defined Metadata (Métadonnées utilisateur) :** Tu peux ajouter tes propres informations lors de l'upload (ex: `statut: confidentiel`, `application: CRM`). Elles doivent obligatoirement commencer par le préfixe `x-amz-meta-`. Elles ne peuvent plus être modifiées après l'upload (il faut réécrire l'objet pour les changer).

---

8. Les Tags (Étiquettes)

Similaires aux métadonnées, les tags sont des paires Clé/Valeur (jusqu'à 10 par objet), mais ils ont un rôle très différent et dynamique.

Différence cruciale avec les métadonnées :

- Les tags peuvent être modifiés **sans modifier ou réuploader l'objet**.
- **À quoi ça sert ?**
  1. **Gestion des coûts :** Analyser la facturation (ex: tag `Project: Apollo`).
  2. **Sécurité (ABAC - Attribute-Based Access Control) :** Autoriser un utilisateur à lire un objet uniquement si l'objet possède le tag `Security: Public`.
  3. **Cycle de vie :** Dire à S3 "Supprime tous les objets qui ont le tag `Temporary: True` après 30 jours".

---

9. Le Version ID (L'identifiant de version)

Par défaut, le **Versioning** est désactivé sur un bucket. Une fois activé, il ne peut plus être désactivé, seulement suspendu (*Suspended*).

- **Le principe :** Chaque fois que tu uploades un objet avec le même nom (la même clé), S3 ne l'écrase pas. Il crée une nouvelle version et lui attribue un **Version ID** unique.
- **Protection contre les erreurs :** Si tu supprimes un objet par mégarde, S3 ajoute simplement un **Delete Marker** (un marqueur de suppression) par-dessus. L'objet semble avoir disparu, mais il suffit de supprimer ce marqueur pour récupérer le fichier. Pour supprimer définitivement, il faut spécifier le *Version ID* exact.
- **Facturation :** Tu payes pour le stockage de **toutes** les versions. Si tu as 5 versions d'un fichier de 1 Go, tu payes pour 5 Go.

02. S3 Security

Par défaut, quand tu crées un bucket S3, tout est fermé. Personne n'a accès à rien, sauf le créateur du compte.

---

1. Les deux outils pour ouvrir l'accès

Pour donner une permission, tu choisis l'une de ces deux méthodes (ou les deux) :

Method A : La méthode "Utilisateur" (IAM Policy)

- **C'est quoi ?** Une feuille de permission que tu donnes à un utilisateur ou un rôle précis.
- **Logique :** Tu dis à l'utilisateur : *"Toi, tu as le droit d'aller lire le bucket X."*

Method B : La méthode "Bucket" (Bucket Policy)

- **C'est quoi ?** Une feuille de permission collée directement sur le bucket.
- **Logique :** Le bucket dit : *"J'autorise telle personne ou tel compte externe à venir chez moi."*

---

2. Comment S3 prend sa décision ?

Quand quelqu'un clique sur un fichier dans S3, AWS vérifie les permissions dans cet ordre :

1. **Y a-t-il une interdiction (`Deny`) ?** Si une règle dit "Interdit", c'est fini. L'accès est bloqué immédiatement, même si une autre règle disait oui.
2. **Y a-t-il une autorisation (`Allow`) ?** S'il n'y a pas de `Deny`, il faut au moins un `Allow` (soit sur l'utilisateur, soit sur le bucket) pour que ça passe.

---

3. Trois cas particuliers pour l'examen

Scénario 1 : Bloquer les fuites de données sur Internet

- **Le besoin :** Tu veux une sécurité absolue pour qu'aucun employé ne puisse rendre le bucket public par erreur.
- **La solution : Block Public Access (BPA)**. C'est un gros bouton de sécurité activé par défaut qui bloque tout Internet.

Scénario 2 : Partager un fichier privé avec un client externe

- **Le besoin :** Un client (qui n'a pas de compte AWS) doit télécharger une facture privée ou uploader sa photo.

- **La solution : Presigned URL (URL Présignée)**. Tu lui génères un lien magique, sécurisé et temporaire (valable 15 minutes par exemple).

Scénario 3 : Connecter tes serveurs privés à S3

- **Le besoin :** Tes serveurs AWS sont isolés du monde (pas d'Internet). Ils ont pourtant besoin de parler à S3.
- **La solution : VPC Gateway Endpoint**. Un tunnel privé qui relie tes serveurs à S3 sans jamais passer par Internet.

03. S3 Replication

La réplication S3 est un mécanisme **asynchrone** et **automatique** permettant de copier des objets d'un bucket source vers un ou plusieurs buckets de destination.

---

1. Pourquoi utiliser la réplication ? (Business & Architecture Cases)

En architecture cloud, on ne duplique pas la donnée sans une excellente raison réglementaire ou technique. Voici la logique derrière chaque cas d'usage :

- **Plan de reprise d'activité (Disaster Recovery) :** Si une région AWS entière devenait indisponible (scénario catastrophe), l'application doit pouvoir basculer sur une copie des données opérationnelles située dans une autre région géographique.
- **Conformité légale et souveraineté :** Certaines industries (banque, santé) obligent légalement à stocker des copies de sauvegarde à une distance minimale du centre de données principal, ou à l'inverse, à confiner strictement les données dans les frontières d'un même pays.
- **Sécurité absolue (Isolation des comptes) :** Pour se prémunir d'un ransomware ou d'un employé malveillant qui obtiendrait les accès administrateur du compte principal, on réplique les données vers un bucket appartenant à un **autre compte AWS** totalement isolé, avec des clés d'accès différentes.
- **Optimisation des performances (Latence) :** Si une entreprise a son siège à Paris mais possède une application ultra-utilisée par des clients à Singapour, répliquer les fichiers multimédias au plus près de ces utilisateurs réduit considérablement le temps de téléchargement.

---

2. Les deux types de Réplication

AWS sépare la réplication en fonction des frontières géographiques des infrastructures :

CRR (Cross-Region Replication)

- **Définition :** Source et destination sont dans des **Régions AWS différentes** (ex: `eu-west-3` Paris vers `us-east-1` N. Virginia).
- **Logique :** Répond aux besoins de **Disaster Recovery** mondial et de **réduction de latence** géographique.

SRR (Same-Region Replication)

- **Définition :** Source et destination sont dans la **même Région AWS**.

- **Logique :** Utilisé pour l'**agrégation de logs** (centraliser les logs de 50 buckets applicatifs d'une même région vers un seul bucket d'analyse) ou pour respecter les contraintes strictes de souveraineté des données qui interdisent le transfert transfrontalier.

---

3. Les Prérequis Techniques (Requirements) ?? *Point chaud de l'examen*

Si une question de l'examen mentionne que la réplication "ne fonctionne pas", la cause se trouve presque toujours dans le non-respect d'un de ces critères :

- **Le Versioning est obligatoire :** Il doit être explicitement activé sur le bucket source **ET** sur le bucket destination.
  - *Pourquoi ?* Sans versioning, si deux objets portent le même nom, l'un écraserait l'autre de manière asynchrone, créant des conflits de données impossibles à résoudre automatiquement pour S3.
- **Le Rôle d'exécution IAM :** S3 doit recevoir l'autorisation d'agir en ton nom.
  - *Pourquoi ?* Par défaut, les services AWS sont isolés. S3 a besoin d'un rôle IAM pour obtenir le droit de lire (`s3:GetObjectVersion`) dans la source et d'écrire (`s3:ReplicateObject`) dans la destination.
- **La Bucket Policy de destination (Cas multi-comptes) :** Si la destination est dans un compte AWS B, le propriétaire du compte B doit ajouter une politique de bucket autorisant explicitement le rôle IAM du compte A à déposer des objets.

---

4. Fonctionnement logique & Options Avancées

Comprendre le comportement par défaut de S3 permet d'éviter les pièges sur les options spécifiques.

Sans S3 Batch Replication (par défaut)

Par défaut, la réplication ne regarde pas le passé. Dès que la règle est activée, **seuls les nouveaux objets** (ou les nouvelles versions d'objets existants) sont répliqués.

Avec S3 Batch Replication

- **Pourquoi ?** Si tu as déjà 10 To de données historiques dans un bucket et que tu actives la réplication aujourd'hui, ces 10 To ne sont pas répliqués.
- **Comment ?** Tu dois utiliser **S3 Batch Replication** (qui s'appuie sur S3 Batch Operations) pour scanner le bucket et répliquer de manière rétroactive les objets préexistants.

04. S3 Storage Classes

Amazon S3 propose 8 classes de stockage adaptées à différents cycles de vie des données.

Source : https://aws.amazon.com/fr/s3/storage-classes/

---

1. Vue d'ensemble des 8 Classes de Stockage

Pour l'examen, imagine ces classes sur un axe allant de la donnée ultra-active (chaude) à la donnée archivée (froide).

1. S3 Standard (La classe par défaut)

- **Cas d'usage :** Données fréquemment consultées, sites web statiques, distribution de contenus, assets d'applications actives.
- **Logique :** Conçue pour une haute disponibilité et une latence de l'ordre de la milliseconde. Tu payes le stockage au prix fort, mais l'accès aux données (lecture/écriture) est très économique.

2. S3 Intelligent-Tiering (L'optimisation automatique)

- **Cas d'usage :** Données aux patterns d'accès inconnus, imprévisibles ou changeants.
- **Logique :** AWS déplace automatiquement tes objets entre deux niveaux (Fréquent et Infréquent) en fonction de ton utilisation, sans impact sur les performances.
- **Piège examen :** Il y a de légers frais de gestion mensuels par objet pour l'automatisation. Si tes fichiers sont minuscules (quelques Ko) ou si tu connais parfaitement tes patterns d'accès, ce n'est pas la solution optimale.

3. S3 Standard-Infrequent Access (S3 Standard-IA)

- **Cas d'usage :** Données moins consultées mais qui requièrent un accès immédiat (milliseconde) en cas de besoin (ex: rapports mensuels, sauvegardes récentes).
- **Logique :** Le coût du stockage par Go est plus bas que le Standard, mais **tu payes des frais à chaque fois que tu récupères (retires) un fichier**.
- **Piège examen :** Il y a une taille minimale d'objet facturée (128 Ko) et une durée de rétention minimale facturée (30 jours).

> C'est-à-dire que tu peux tout à fait y placer des fichiers plus petits, mais tu seras facturé comme s'ils faisaient 128 Ko.
>
> Exemple concret :

> - Si tu uploades un fichier de **10 Ko** dans **S3 Standard**, tu payes pour **10 Ko**.
> - Si tu places ce même fichier de **10 Ko** dans **S3 Standard-IA**, S3 l'accepte, mais ton compteur de facturation affichera **128 Ko**. Tu payes donc "dans le vide" pour 118 Ko non utilisés.

4. S3 One Zone-Infrequent Access (S3 One Zone-IA)

- **Cas d'usage :** Données non critiques ou facilement reproductibles qui ne sont pas souvent consultées (ex: copies secondaires de sauvegardes, miniatures d'images générées automatiquement).
- **Logique :** Identique à Standard-IA, mais stocké dans **une seule Zone de Disponibilité (AZ)** au lieu de trois. Le coût de stockage est réduit de 20%.
- **Risque architectural :** Si l'AZ physique subit un sinistre (incendie, inondation), les données sont définitivement perdues.

5. S3 Glacier Instant Retrieval

- **Cas d'usage :** Archives médicales, images d'archives d'actualités, données de conformité rarement consultées mais requises en quelques millisecondes si besoin.
- **Logique :** C'est la classe d'archive la plus rapide. Le coût de stockage est très bas, le coût de retrait est élevé, mais la livraison est immédiate (millisecondes).
- Durée minimale : 90 jours.

6. S3 Glacier Flexible Retrieval

- **Cas d'usage :** Sauvegardes de bases de données, archives annuelles d'entreprise où un délai de récupération est acceptable.
- **Logique :** Moins cher que l'Instant Retrieval, mais l'accès n'est plus immédiat. Tu as trois options de récupération :
  - *Expedited (Accéléré) :* 1 à 5 minutes (très cher).
  - *Standard :* 3 à 5 heures.
  - *Bulk (En vrac) :* 5 à 12 heures (gratuit/très économique).
- Durée minimale : 90 jours.

7. S3 Glacier Deep Archive (L'archive ultime)

- **Cas d'usage :** Conservation légale (données fiscales stockées pendant 10 ans), conformité réglementaire stricte où les données ne seront probablement jamais lues.
- **Logique :** Le coût de stockage le plus bas de tout AWS. En contrepartie, le temps de récupération est très long :
  - *Standard :* 12 heures.
  - *Bulk :* 48 heures.
- Durée minimale : 180 jours.

---

2. Le cas à part : S3 Express One Zone ?

Introduit pour répondre à des besoins de calcul intensif moderne, **S3 Express One Zone** change radicalement l'architecture traditionnelle de S3.

- **Pourquoi a-t-il été créé ?** Pour des performances extrêmes à faible latence. Les applications comme le Machine Learning (entraînement de modèles), l'analyse Big Data financière (Athena/EMR à grande échelle) ou le rendu 3D passaient trop de temps à attendre que S3 lise/écrive les données.
- **La différence technique :** C'est une classe de stockage de type **Directory Bucket** (alors que toutes les autres sont des *General Purpose Buckets*).
  - Les données sont stockées dans **une seule AZ**, au plus près des instances de calcul (EC2, ECS, EKS) de l'utilisateur.
  - Il offre des **performances jusqu'à 10 fois supérieures** à S3 Standard, avec des latences constantes de l'ordre de la **milliseconde à un chiffre (single-digit millisecond)**.
  - Il permet de traiter des **millions de requêtes par seconde.**

05. Advanced Amazon S3

Questions :

- C'est quoi Event bridge ?
- C'est quoi SNS / SQS ?
- fef

1. S3 Lifecycle Rules (Politiques de Cycle de Vie)

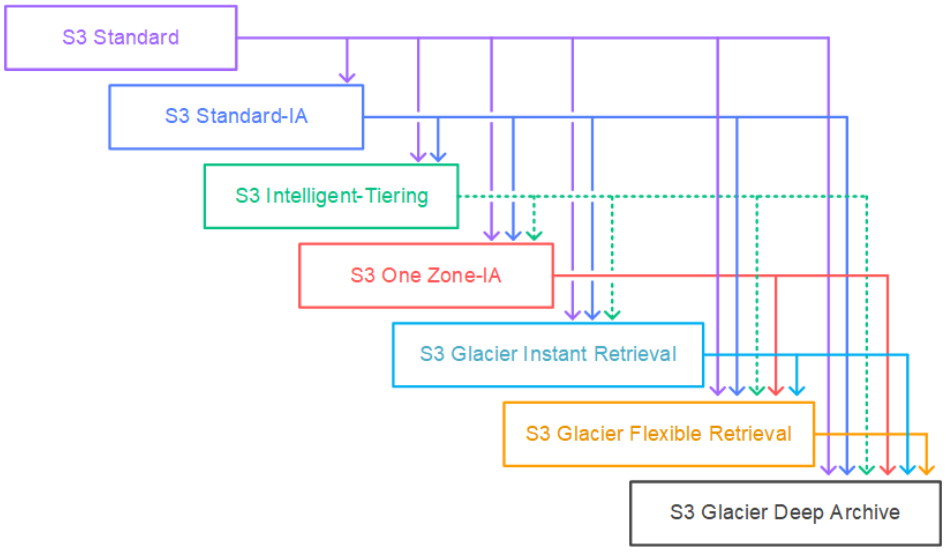
Transition Actions & The Transition Ladder (L'Échelle de Transition)

Pourquoi ? Automatiser la réduction des coûts de stockage au fil du temps sans intervention humaine. Les données vieillissent et deviennent généralement moins "chaudes".

Comment ? : Tu définis des règles pour déplacer les objets d'une classe de stockage à une autre après un nombre de jours précis. Tu ne peux pas transitionner vers le "haut" (du froid vers le chaud). Il existe un ordre strict imposé par AWS.

Remarque : AWS impose des délais minimaux avant de pouvoir passer à la classe suivante.

Exemple : Pour passer de *S3 Standard* à *S3 Standard-IA*, l'objet doit avoir **au moins 30 jours**. Si tu configures une règle à 15 jours, S3 refusera la transition.



Source : https://docs.aws.amazon.com/AmazonS3/latest/userguide/lifecycle-transition-general-considerations.html

Expiration Actions

- **Pourquoi ?** Pour supprimer automatiquement les données devenues inutiles (ex: logs de plus de 90 jours) ou nettoyer l'espace.
- **Cas particulier (Versioning) :** Tu peux configurer l'expiration pour supprimer définitivement les **anciennes versions** d'un objet (Noncurrent Versions) tout en gardant la version actuelle, ou supprimer les "Delete Markers" périmés.
  - A EXPLIQUER

Filtres : Prefix & Tags Rules

Une règle de cycle de vie ne s'applique pas forcément à tout le bucket. Tu peux cibler :

- Un **Prefix** (chemin/dossier) : ex, uniquement les objets dans `logs/2026/`.
- Des **Object Tags** (clés/valeurs) : ex, uniquement les objets marqués `Environment: Development`.

Amazon S3 Storage Class Analysis

Comment savoir s'il faut passer à Standard-IA après 30 jours ou 60 jours ?

Cet outil analytique observe tes patterns d'accès aux données pendant plusieurs semaines. Il génère des graphiques et te recommande la politique de cycle de vie idéale (le nombre de jours exact) pour maximiser tes économies.

---

2. S3 Requester Pays

- **Pourquoi ?** Tu es une entreprise de recherche (Compte A) qui héberge un dataset de 5 To très populaire. Si des milliers de clients (Compte B, C, D) téléchargent tes fichiers, tu vas payer une fortune en frais de transfert de données sortantes (Data Transfer Out / Egress).
- **Le fonctionnement :** En activant Requester Pays, tu restes responsable du coût du stockage des données, mais c'est l'utilisateur qui télécharge l'objet qui paye les coûts de transfert (Egress) et de requête (GET).
- **La condition absolue (Identité AWS) :** Pour que cela fonctionne, le demandeur doit obligatoirement être authentifié auprès d'AWS. Les accès anonymes/publics sont automatiquement rejetés sur un bucket Requester Pays. AWS doit savoir quel compte débiter !

---

3. Event Notifications

Filtrage des événements

- **Pourquoi ?** Déclencher une action automatique lorsqu'un événement survient dans S3 (ex: exécuter un script dès qu'une image est uploadée).

Gestion des Permissions IAM : Les Politiques de Ressources

Pour que S3 puisse envoyer une notification à un autre service, il a besoin d'une autorisation. Contrairement à d'autres services, cela ne se fait pas via un rôle IAM classique attaché à S3, mais

via des **Resource-based Policies** :

Exemples :

- SNS (Simple Notification Service) : Tu dois modifier la SNS Topic Policy pour autoriser le Principal `s3.amazonaws.com` à exécuter l'action `SNS:Publish`.
- SQS (Simple Queue Service) : Tu dois modifier la SQS Queue Policy pour autoriser `s3.amazonaws`.com à faire un `SQS:SendMessage`.
- Lambda : Tu dois ajouter une Resource Policy sur la fonction Lambda (`lambda:InvokeFunction`) autorisant le bucket S3 spécifique à l'appeler.

EventBridge (Advanced Filtering)

- Pourquoi passer par EventBridge ? Les notifications S3 natives sont limitées (un seul prefix/suffix par règle, pas de filtrage sur la taille de l'objet, etc.).
- Avantages EventBridge : 1. Permet un filtrage ultra-avancé (ex: déclencher une alerte uniquement si l'objet fait plus de 1 Go ET possède le tag Confidential).
  2. Permet d'envoyer l'événement vers beaucoup plus de destinations (plus de 18 cibles AWS comme Step Functions, Kinesis, API Destinations).

4. S3 Performance & Optimisation

S3 est conçu pour être massivement scalable par défaut, mais pour l'examen, tu dois connaître les limites physiques et les techniques d'optimisation.

Les limites par Préfixe (La règle d'or)

Les performances de S3 sont calculées **par seconde et par préfixe (dossier)**. Un bucket n'a pas de limite globale, chaque préfixe est indépendant :

- **3 500 requêtes** de modification (`PUT / COPY / POST / DELETE`) par seconde par préfixe.
- **5 500 requêtes** de lecture (`GET / HEAD`) par seconde par préfixe.
- *Astuce architecture :* Si tu as besoin de 11 000 GET/s, crée deux dossiers distincts (`/folder1` et `/folder2`) et répartis tes fichiers.

Multipart Upload

- **Pourquoi ?** Obligatoire pour les fichiers de **plus de 5 Go**, mais fortement recommandé dès **100 Mo**.
- **Comment ?** Découpe le fichier en plusieurs morceaux et les uploade en parallèle. Si un morceau échoue à cause du réseau, tu ne recommences que ce morceau, pas tout le fichier.

S3 Transfer Acceleration

- **Pourquoi ?** Un utilisateur en Australie doit uploader un fichier de 2 Go vers un bucket situé à Dublin. La traversée de l'internet public va être lente et instable.
- **Comment ?** Le fichier est envoyé vers l'**Edge Location** CloudFront (réseau privé mondial d'AWS) la plus proche de l'utilisateur en Australie. Le reste du voyage jusqu'à Dublin se fait sur le réseau de fibre optique ultra-rapide d'AWS.

S3 Byte-Range Fetches

- **Pourquoi ?** Tu as un fichier vidéo de 50 Go sur S3 et tu veux juste lire les 5 premières minutes, ou tu veux accélérer le téléchargement d'un gros fichier.
- **Comment ?** Tu passes une commande HTTP `Range` pour demander des segments d'octets spécifiques (ex: les premiers 10 Mo). Permet de paralléliser la lecture d'un même fichier pour doper les performances.

5. S3 Batch Operations (Opérations en Lot)

- **Pourquoi ?** Tu as un bucket avec **100 millions d'objets existants**. On te demande d'ajouter un tag sur chacun d'eux, ou de les chiffrer, ou de changer leur classe de stockage.
- **Pourquoi c'est mieux qu'un script fait maison ?**
  - *Script maison (EC2 + SDK) :* Tu vas devoir gérer les pannes de réseau, le multi-threading, le throttling (limites S3), le suivi de ce qui a réussi ou échoué, et payer l'infrastructure de calcul. Cela prendrait des jours de dev et de run.
  - *S3 Batch Operations :* C'est un service entièrement managé (Serverless). Tu lui fournis un fichier manifeste (la liste des objets à traiter via S3 Inventory), tu choisis l'action (ex: *Replace all tags*), et AWS gère la mise à l'échelle, la reprise après erreur et te fournit un rapport final complet.

6. S3 Storage Lens

- **Pourquoi ?** Comprendre, analyser et optimiser le stockage à l'échelle d'une multinationale (**toute une AWS Organization**, des dizaines de comptes, des milliers de buckets).

S3 Storage Lens fournit un tableau de bord centralisé découpé en plusieurs piliers d'analyse :

1. **Summary (Résumé) :** Vue globale sur la taille totale du stockage (en To) et le nombre d'objets.
2. **Cost-optimization (Optimisation des coûts) :** Identifie les buckets qui ont des versions obsolètes non nettoyées ou des fichiers qui devraient être migrés vers Glacier.
3. **Data protection (Protection des données) :** Te montre le pourcentage de buckets où le *Versioning*, *MFA Delete* ou la *Réplication* ne sont pas activés.
4. **Access management (Gestion des accès) :** Repère les buckets configurés avec des accès publics dangereux.
5. **Event / Performance / Activity :** Analyse la fréquence d'utilisation et les taux de requêtes.
6. **Status Code :** Met en avant les erreurs récurrentes (ex: beaucoup d'erreurs `403 Forbidden` ou `404 Not Found` sur un bucket spécifique).

Free vs. Paid (Gratuit vs Payant)

- **Version Gratuite :** Activée par défaut pour tous les comptes. Fournit 29 métriques d'activité de base avec un historique de rétention de **14 jours** (agrégé au niveau du bucket).
- **Version Payante (Advanced) :** Permet d'obtenir des métriques avancées (recommandations de coûts, codes d'erreur HTTP), d'analyser jusqu'au niveau du **Prefix**, et d'étendre la rétention des données à **15 mois** pour faire des analyses de tendance à long terme.

? Cheat Sheet Mentale pour l'Examen (Modules 3 à 6)

> 1. Erreur de transition de cycle de vie $\rightarrow$ Vérifier la règle des **30 jours minimum** pour Standard-IA.
> 2. Partager un gros dataset sans payer le trafic sortant $\rightarrow$ **S3 Requester Pays**.
> 3. Filtrage d'événement complexe ou cible exotique $\rightarrow$ **Amazon EventBridge**.
> 4. Atteinte des limites de requêtes S3 (HTTP 503) $\rightarrow$ **Créer des sous-préfixes (dossiers) supplémentaires** pour diviser la charge.
> 5. Action massive sur des millions d'objets *existants* $\rightarrow$ **S3 Batch Operations**.
> 6. Rapport de conformité/coûts de S3 sur 50 comptes AWS différents $\rightarrow$ **S3 Storage Lens**.

Tu te sens prêt à tester tes connaissances avec une mise en situation réelle d'examen sur l'ensemble de tes notes S3 ?

06. S3 Encryption

1. Chiffrement au repos

Le dilemme à l'examen repose toujours sur deux critères : **Qui gère la clé ?** et **Où se fait le chiffrement ?**

A. Server-Side Encryption (SSE) : AWS chiffre pour vous

Les données arrivent brutes chez AWS, et c'est S3 qui s'occupe de les chiffrer avant de les écrire sur le disque.

- **SSE-S3 (Chiffrement géré par S3)**
  - **Le concept :** AWS gère tout. La clé est créée, stockée et gérée automatiquement par AWS.
  - **Algorithme :** AES-256.
  - **Pourquoi ?** C'est la solution par défaut (gratuite et zéro gestion).
  - **Ce qu'il faut savoir pour l'examen :** Depuis 2023, **SSE-S3 est activé par défaut** sur tous les nouveaux buckets S3. Si vous n'en spécifiez aucun, c'est celui-là qui s'applique.
  - **Header HTTP à l'upload :** `"x-amz-server-side-encryption": "AES256"`

- **SSE-KMS (Chiffrement via AWS KMS)**
  - **Le concept :** Vous utilisez le service AWS Key Management Service (KMS) pour gérer les clés. Vous pouvez utiliser la clé AWS par défaut (`aws/s3`) ou créer votre propre clé personnalisée (Customer Managed Key).
  - **Pourquoi ?** Pour avoir un **contrôle d'accès strict** (qui a le droit d'utiliser la clé) et un **suivi d'audit (CloudTrail)** complet de chaque chiffrement/déchiffrement.
  - **Ce qu'il faut savoir pour l'examen :** Génère des coûts (appels API KMS).
    - Attention aux quotas de requêtes KMS en cas de fort trafic.
    - Pour déchiffrer un objet, l'utilisateur a besoin de la permission S3 **ET** de la permission `kms:Decrypt`.
  - **Header HTTP à l'upload :** `"x-amz-server-side-encryption": "aws:kms"`

- **SSE-C (Chiffrement avec clés fournies par le client)**
  - **Le concept :** Vous possédez et gérez la clé secrète, mais vous la passez à S3 dans la requête HTTPS pour qu'AWS fasse le travail de chiffrement. **AWS ne stocke jamais votre clé**. Une fois l'opération finie, AWS jette la clé de sa mémoire vive.
  - **Pourquoi ?** Pour les entreprises qui ont des obligations de conformité strictes interdisant de confier la gestion des clés à un tiers (même AWS).
  - **Ce qu'il faut savoir pour l'examen :** Si vous perdez la clé, vos données sont perdues à jamais. AWS a récemment renforcé la sécurité en désactivant le support SSE-C par défaut sur les nouveaux buckets (il faut l'autoriser explicitement via l'API si besoin).

    - **Header HTTP :** Vous devez envoyer la clé dans chaque requête HTTP (`x-amz-server-side-encryption-customer-key`).

B. Client-Side Encryption : Vous chiffrez avant d'envoyer

- **Le concept :** Vos données sont chiffrées **au sein même de votre application** (par exemple avec le SDK AWS / S3 Encryption Client) avant de franchir le réseau. AWS reçoit un fichier qui est déjà un bloc illisible.
- **Pourquoi ?** Sécurité maximale "End-to-End". AWS n'a aucun moyen de voir ce qu'il y a dans votre fichier, même s'ils le voulaient.

2. Chiffrement en transit (Encryption in Transit)

Le chiffrement en transit garantit que personne ne peut intercepter vos données entre votre client et les serveurs d'AWS.

- **Le comment :** S3 supporte nativement les protocoles **HTTP** et **HTTPS (TLS)**.
- **Pour l'examen (Piège classique) :** Comment forcer le chiffrement en transit ? On utilise une **Bucket Policy** avec une condition explicite qui rejette (`Deny`) toutes les connexions non sécurisées.

Voici à quoi ressemble la règle absolue à connaître pour le SAA-C03 :

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "EnforceHTTPS",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::mon-bucket-examen",
        "arn:aws:s3:::mon-bucket-examen/*"
      ],
      "Condition": {
        "Bool": {
          "aws:SecureTransport": "false"
        }
      }
    }
  ]
}
```

Comme `aws:SecureTransport` est `false` (donc du HTTP classique), l'accès est bloqué (`Deny`).

Advanced S3 Concepts - Part 2

1. Amazon S3 CORS (Cross-Origin Resource Sharing)

Le contexte et le problème

Par défaut, les navigateurs web appliquent une règle de sécurité très stricte appelée **Same-Origin Policy** (Politique de même origine). Si un utilisateur visite `site-a.com`, le code JavaScript de ce site ne peut pas, par défaut, aller chercher des ressources (comme des polices, des images ou des scripts) sur `site-b.com` ou sur un bucket S3 `mon-bucket.s3.amazonaws.com`. Le navigateur bloque la requête pour éviter les attaques malveillantes.

À quoi ça sert ?

CORS est un mécanisme qui permet de dire explicitement au navigateur web : "*Hé, j'autorise le site A à venir piocher des ressources dans mon bucket S3*".

Ce qu'il faut retenir pour l'examen

- **C’est une politique du navigateur web :** Ce n'est pas S3 qui bloque la requête, c'est le navigateur de l'utilisateur qui refuse de charger la ressource parce que S3 n'a pas renvoyé les bons en-têtes d'autorisation.
- **La configuration :** Vous écrivez une règle CORS (en JSON) sur le bucket S3 pour lister les origines autorisées (ex: `https://mon-site-web.com`), les méthodes (GET, PUT) et les en-têtes.

2. S3 MFA Delete

Le contexte et le problème

Imaginez qu'un employé en colère ou qu'un pirate réussisse à voler les identifiants d'un administrateur. Il pourrait supprimer définitivement des données critiques de l'entreprise en quelques clics.

À quoi ça sert ?

MFA Delete ajoute une double sécurité physique. Même si vous avez les droits d'accès, AWS va vous demander de saisir un code MFA (Multi-Factor Authentication) pour deux actions ultra-critiques.

Ce qu'il faut retenir pour l'examen (Attention aux pièges !)

- **Quand le MFA est-il requis ?**
  - Pour **supprimer définitivement** une version d'un objet.
  - Pour **suspendre le versioning** du bucket.

- **Le versioning doit être activé :** Logique, puisque le MFA Delete sert principalement à protéger les versions des objets. Vous ne pouvez pas activer le MFA Delete si le versioning n'est pas activé.
- **Le piège de l'examen (Root Account) :** *Seul le compte racine (Root Account)* de la solution AWS peut activer ou désactiver le MFA Delete. Un utilisateur IAM, même avec les droits `AdministratorAccess`, ne peut pas le faire. (Note : en revanche, pour *supprimer* l'objet une fois configuré, l'utilisateur IAM peut le faire s'il a les droits ET le token MFA).

3. S3 Access Logs (Journaux d'accès)

Le contexte et le problème

Dans une grande entreprise ou pour des raisons de conformité légale, vous devez être capable de répondre à la question : "*Qui a consulté ou modifié ce fichier confidentiel mardi dernier à 14h ?*"

À quoi ça sert ?

Les **S3 Access Logs** enregistrent toutes les requêtes (GET, PUT, DELETE) envoyées à votre bucket. Cela fournit des enregistrements détaillés (IP source, utilisateur, type de requête, date) **à des fins d'audit** et de sécurité.

Ce qu'il faut retenir pour l'examen

- **Même région AWS obligatoirement :** Le bucket source (celui que vous surveillez) et le bucket cible (celui qui stocke les fichiers de logs) **doivent être dans la même région AWS**. AWS ne vous laissera pas envoyer des logs S3 d'une région à une autre directement via cette fonctionnalité.
- **Attention à la boucle de logging (Logging Loop) :** C'est le piège classique d'architecture. Si vous dites à votre `Bucket-A` d'écrire ses logs dans... `Bucket-A`, chaque écriture de log va générer une nouvelle ligne de log, qui générera une écriture, et ainsi de suite. Votre facture va exploser et le bucket va saturer. **Règle d'or :** Toujours stocker les logs dans un bucket cible *distinct*.

4. Pre-signed URLs (URLs pré-signées)

Le contexte et le problème

Vous avez des fichiers privés dans un bucket S3 (par exemple, des vidéos de formation payantes). Vous ne voulez pas rendre le bucket public. Comment permettre à un utilisateur qui a payé de télécharger sa vidéo sans lui créer un compte AWS ?

À quoi ça sert ?

Une **URL pré-signée** est une URL temporaire générée par vous (via du code) qui donne une autorisation d'accès temporaire à un objet S3 spécifique. Elle contient une clé de sécurité et une date d'expiration (par exemple, valide pendant 15 minutes).

Ce qu'il faut retenir pour l'examen

- **Héritage des permissions :** C'est le point clé. L'URL pré-signée est générée *au nom* d'un utilisateur IAM ou d'un rôle. L'utilisateur qui reçoit l'URL **hérite exactement des permissions de la personne/du rôle qui l'a créée**. Si le développeur qui a généré l'URL n'a pas le droit de lire le fichier, l'URL pré-signée ne fonctionnera pas (elle renverra un code 403 Forbidden).
- **Comment l'utiliser ?** Vous avez mentionné la console, le CLI ou le SDK. Précision importante pour le SAA-C03 : dans la vraie vie, on utilise le **SDK** (dans une fonction Lambda par exemple) pour les générer dynamiquement pour les utilisateurs de votre application. Mais vous pouvez aussi en générer rapidement via l'**AWS CLI** (pour tester) ou depuis la **Console S3** (via l'option "Partager avec une URL pré-signée").

5. S3 Object Lock & Glacier Vault Lock

Le contexte et le problème

Dans les secteurs bancaires, de la santé ou du secteur public, la loi impose souvent de conserver des documents (ex: fiches de paie, dossiers médicaux) pendant plusieurs années **sans qu'absolument personne** (pas même le grand patron ou le pirate qui a volé le compte Root) ne puisse les modifier ou les supprimer.

À quoi ça sert ?

Ils implémentent le modèle **WORM** (*Write Once, Read Many* — Écrire une fois, lire plusieurs fois). Une fois le fichier écrit, il est gravé dans le marbre pour la durée choisie.

Ce qu'il faut retenir pour l'examen

S3 Object Lock (Au niveau de l'objet ou du bucket)

Il fonctionne avec deux modes principaux pour gérer la période de rétention (*Retention Period*) :

- **Mode Compliance (Conformité) :** Le mode ultra-strict. **Personne**, absolument personne (y compris l'utilisateur Root), ne peut supprimer ou modifier l'objet, ni changer le mode, ni raccourcir la durée de rétention. C'est ce qu'on utilise pour la stricte conformité légale (*compliant and data retention*).
- **Mode Governance (Gouvernance) :** Moins strict. La plupart des utilisateurs ne peuvent pas supprimer l'objet, mais vous pouvez accorder des permissions spéciales IAM (ex: `s3:BypassGovernanceRetention`) à certains administrateurs de confiance pour qu'ils puissent modifier les règles ou supprimer le fichier si besoin.
- **Legal Hold (Maintien juridique) :** C'est un interrupteur ON/OFF (sans date de fin). On l'active quand il y a une enquête policière ou un procès en cours. Tant qu'il est activé, l'objet ne peut pas être supprimé. Contrairement à la rétention classique, un utilisateur avec le droit `s3:PutObjectLegalHold` peut le désactiver à tout moment.

Glacier Vault Lock (Pour les archives à long terme)

- Ici, on crée une **Vault Lock Policy** (politique de verrouillage de coffre).

- **Le piège de l'examen :** Une fois que vous appliquez et *verrouillez* cette politique (vous avez un compte à rebours de 24h pour tester), **elle devient totalement immuable**. Plus personne ne peut la supprimer ou la modifier.

6. S3 Access Points (Points d'accès S3)

Le contexte et le problème

Imaginez un bucket S3 unique qui centralise toutes les données d'une entreprise (données financières, RH, marketing, tech). Historiquement, vous deviez écrire une **Bucket Policy** (politique de bucket) géante. Au bout de quelques mois, la politique fait 300 lignes, elle est illisible, et si vous faites une erreur pour accorder un accès au marketing, vous risquez de casser les accès de la finance.

À quoi ça sert ?

Les **Access Points** permettent de **simplifier la gestion des accès** en créant des "portes d'entrée" uniques et dédiées pour chaque équipe ou application. Au lieu d'une seule grosse politique de bucket, vous créez un point d'accès pour le marketing (avec sa propre mini-politique), un pour la finance, etc.

Ce qu'il faut retenir pour l'examen

- **Chaque point d'accès a sa propre politique ARN (Access Point Policy) :** Elle définit précisément ce que les utilisateurs qui passent par cette porte ont le droit de faire.
- **Origine réseau (VPC / Gateway Endpoint) :** Vous pouvez restreindre un point d'accès pour qu'il ne soit accessible **que depuis l'intérieur d'un VPC spécifique**. Pour ce faire, le trafic passe de manière privée par un **VPC Gateway Endpoint** (S3), garantissant que les données ne transitent jamais par l'Internet public.

> Pour restreindre l'accès à vos données S3 aux seules frontières de votre réseau d'entreprise, vous pouvez configurer un **S3 Access Point** pour qu'il soit exclusivement accessible depuis l'intérieur d'un **VPC (Virtual Private Cloud)** spécifique. Concrètement, l'application située dans votre VPC privé va interroger ce point d'accès en empruntant un **VPC Gateway Endpoint** (ou un *VPC Interface Endpoint* selon votre architecture), ce qui permet au trafic réseau de transiter intégralement via le réseau interne d'AWS sans jamais s'exposer sur l'Internet public. De cette manière, même si un utilisateur possède des identifiants valides, toute tentative de connexion via ce point d'accès en dehors du VPC ou depuis une connexion Internet classique sera automatiquement bloquée.

7. S3 Object Lambda

Le contexte et le problème

Vous avez un bucket contenant des images en haute résolution ou des fichiers de données clients au format JSON.

- L'équipe Marketing a besoin des images converties en miniature (JPG).
- L'équipe Analytics a besoin des fichiers JSON mais **sans** les données personnelles (RGPD / anonymisation).
- L'équipe Dev a besoin des fichiers JSON complets. Avant, vous deviez stocker 3 versions différentes du même fichier, ou créer une API complexe.

À quoi ça sert ?

S3 Object Lambda permet d'intercepter la requête de l'application appelante et de **modifier l'objet à la volée via une fonction AWS Lambda** avant de lui renvoyer.

Ce qu'il faut retenir pour l'examen

- **Un seul fichier source :** Il n'y a qu'une seule copie originale du fichier dans S3.
- **Transformation à la volée (On-the-fly) :** Lorsque l'application fait un `GetObject`, S3 Object Lambda déclenche une fonction Lambda. C'est ce code Lambda qui va, par exemple :
  - Anonymiser ou masquer des données sensibles (ex: remplacer les numéros de carte bleue par des `X`).
  - Redimensionner une image.
  - Convertir un format (ex: XML en JSON).
- L'application reçoit le fichier modifié comme si de rien n'était, de manière totalement transparente.