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.

Revision #6
Created 2026-06-03 08:24:47 UTC by Victor
Updated 2026-06-03 12:20:36 UTC by Victor