AWS Academy Cloud Security Foundations

Lab 3.1 Report: Using Resource -Based Policies to Secure an S3 Bucket

hamza taki
3ACI Info

1. Présentation du laboratoire et objectifs

Ce TP a pour vocation d'éclaircir la distinction entre les stratégies basées sur l'identité (IAM) et celles basées sur les ressources (spécifiquement pour les buckets S3). L'étude porte également sur l'interaction entre ces deux types de politiques afin de mettre en place un contrôle d'accès précis.

Objectifs clés :

- Assimiler la différence fondamentale entre les politiques liées aux identités et celles liées aux ressources.
- Illustrer le mécanisme d'emprunt de rôle (*Assume Role*) permettant à un utilisateur IAM d'ajuster ou d'élever ses privilèges.
- Observer les conséquences concrètes de ces configurations sur l'affichage et les opérations possibles au sein de la console AWS.

Architecture :

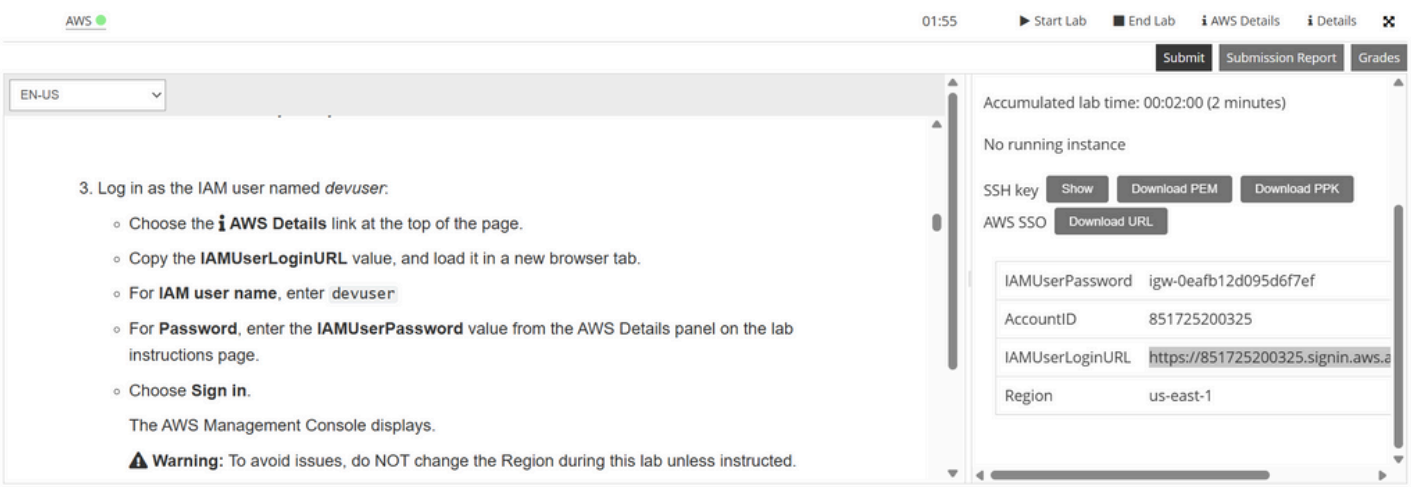
- **Architecture initiale :**
  Le point de départ est l'utilisateur devuser, membre du groupe DeveloperGroup.
  Architecture cible :
  La finalité est l'exploitation de rôles IAM dédiés (BucketAccessRole et

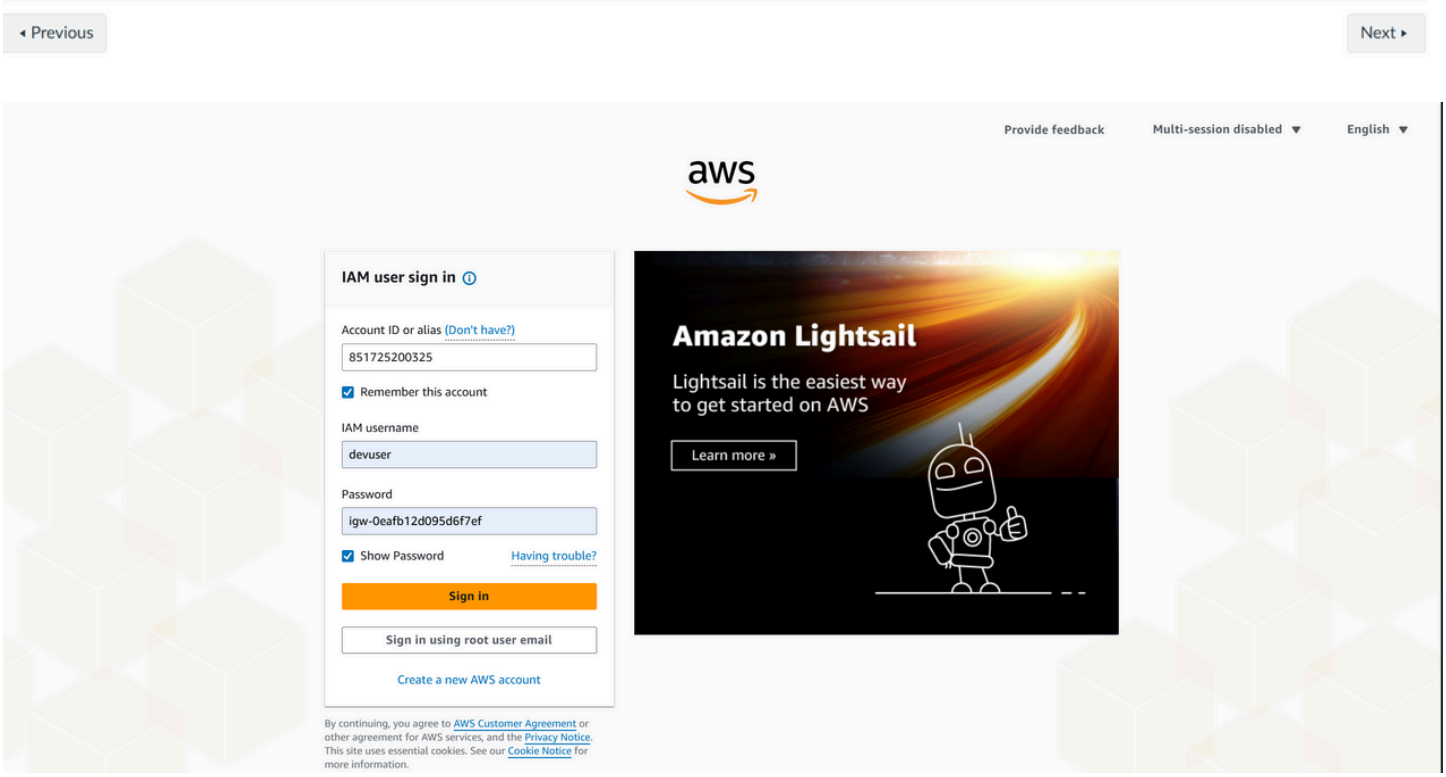
OtherBucketAccessRole) pour interagir avec des buckets S3 aux accès restreints.

2. Description détaillée des tâches

Tâche 1 : Accès à la console

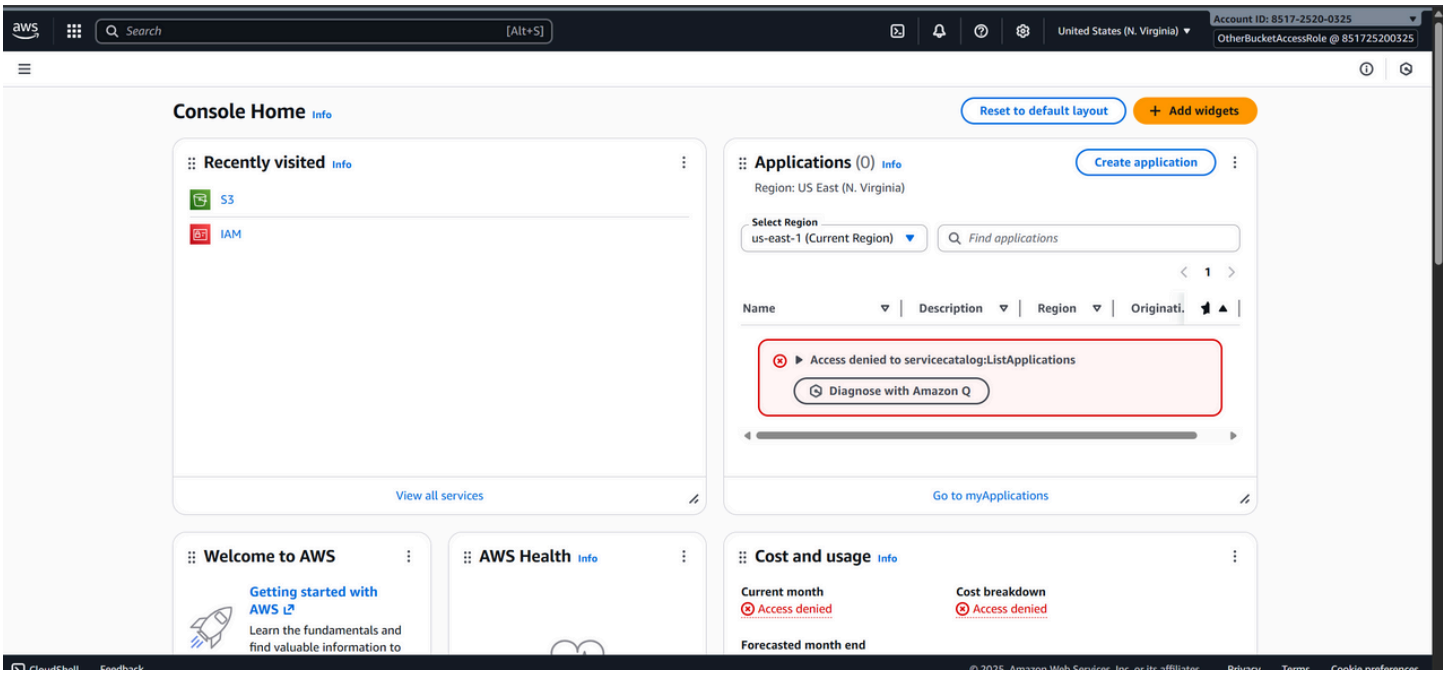
- **Action :** Authentifiez-vous sur le portail de gestion AWS à l'aide des identifiants attribués à l'utilisateur devuser.





Observation :

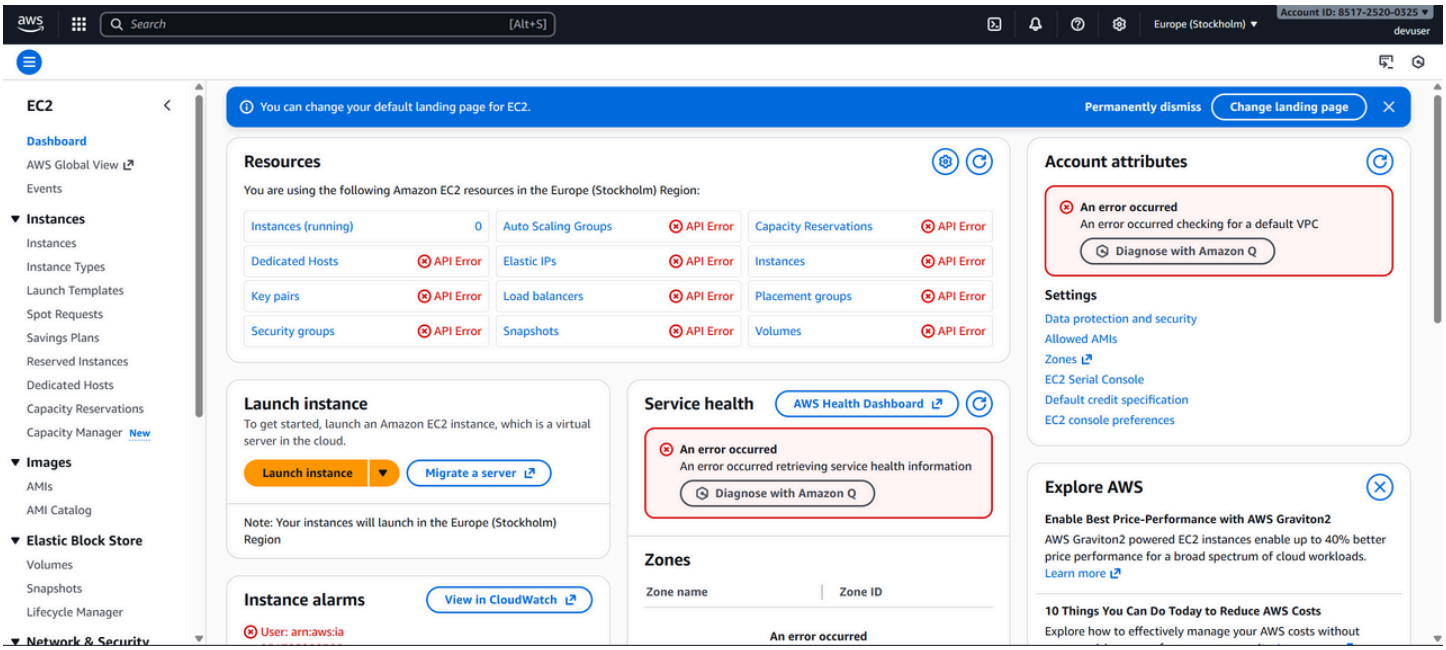
La connexion s’est effectuée avec succès, nous amenant directement sur la page d’accueil de la console AWS.



Tâche 2 : Tentative d’accès en lecture aux services AWS

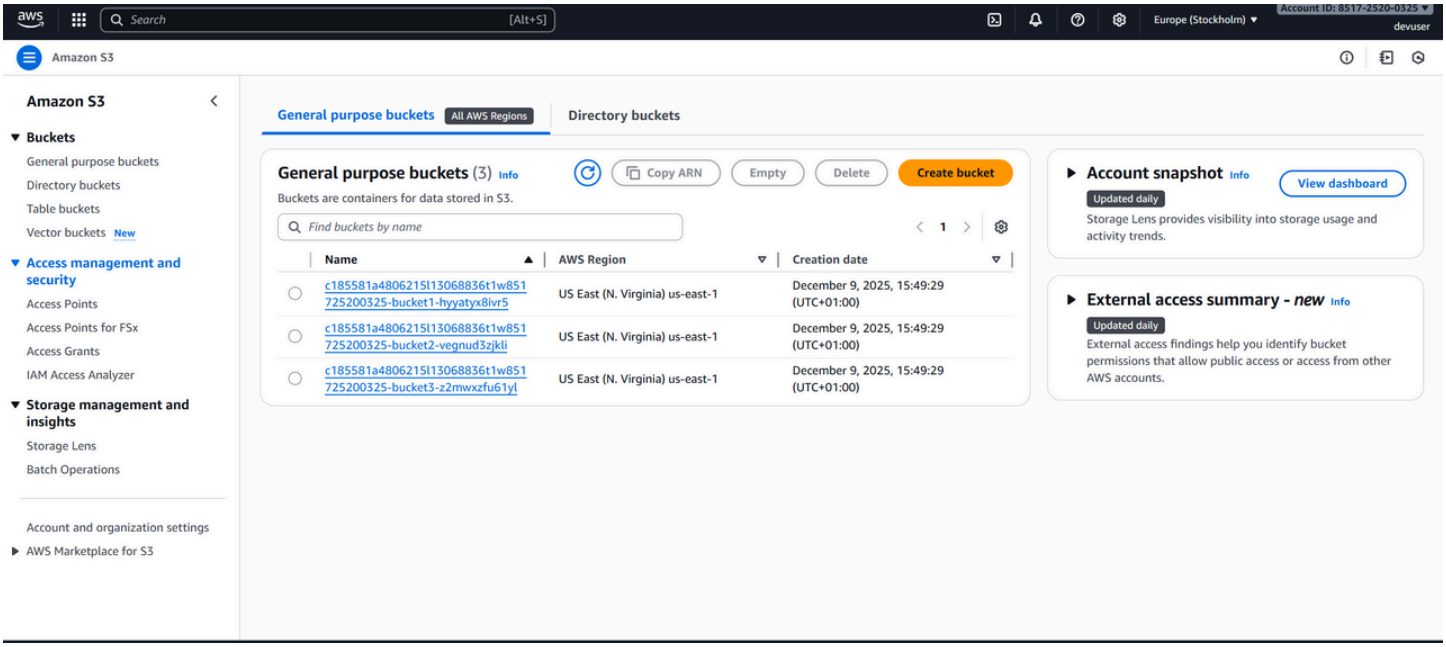
Action 1 : Accéder à Amazon EC2

- **Étape :** Rendez-vous sur le tableau de bord EC2 pour tenter de visualiser les instances existantes.
- **Observation :** Rendez-vous sur le tableau de bord EC2 pour tenter de visualiser les instances existantes.
- **Raison :**
  Ce blocage est dû au fait que l'utilisateur devuser ne dispose pas des autorisations IAM requises (telles que ec2:DescribeInstances) pour interroger les services EC2.



Action 2 : Accès à Amazon S3

- **Étape :** Navigation vers la console Amazon S3.

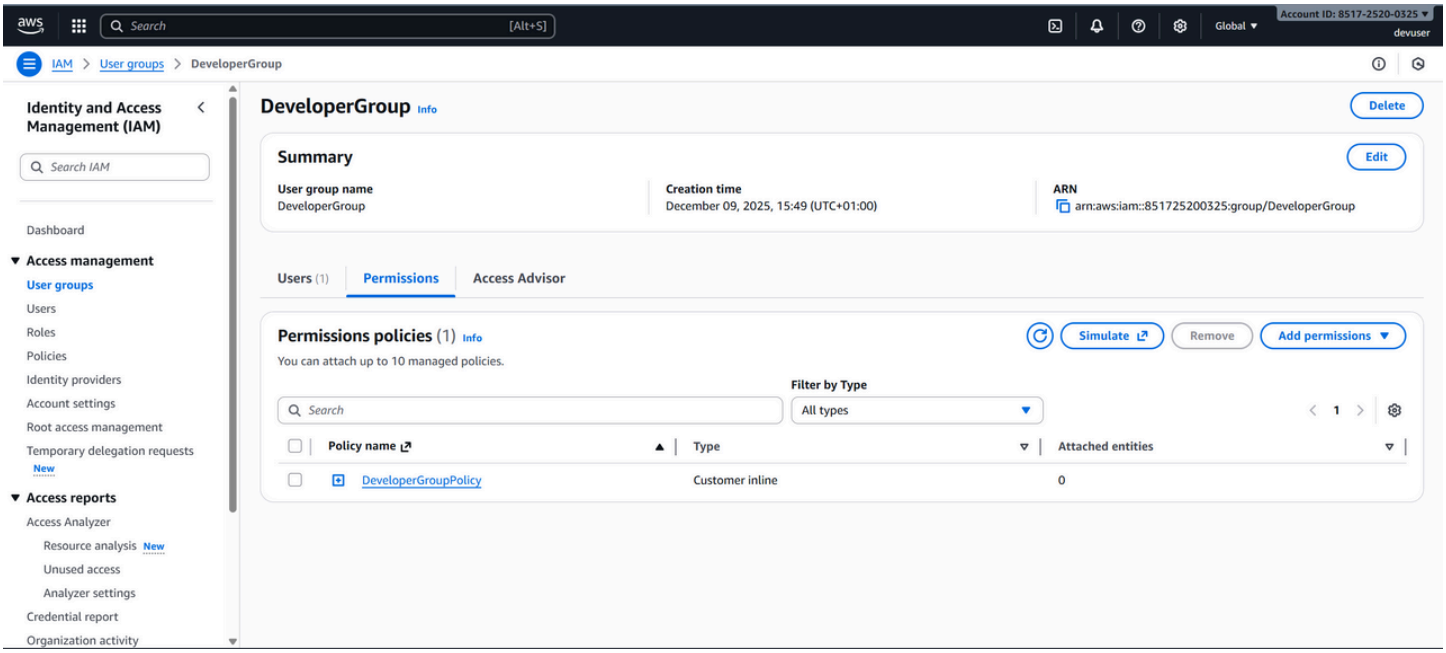


- **Observation :** Bien que la liste des buckets (bucket1, bucket2, bucket3) se soit affichée, la colonne « Access » indiquait « Insufficient permissions » ou « Error ».
- **Raison :** L'utilisateur devuser bénéficie de l'autorisation ListAllMyBuckets, ce qui rend les noms des buckets visibles. Toutefois, l'absence de permissions supplémentaires l'empêche de lire les configurations spécifiques ou de déterminer le statut d'accès public de ces buckets.

Tâche 3 : Analyse de la politique basée sur l'identité

Action :

- Accès à la console IAM → *Users* → *devuser* → *Groups* → *DeveloperGroup* → *Permissions*.

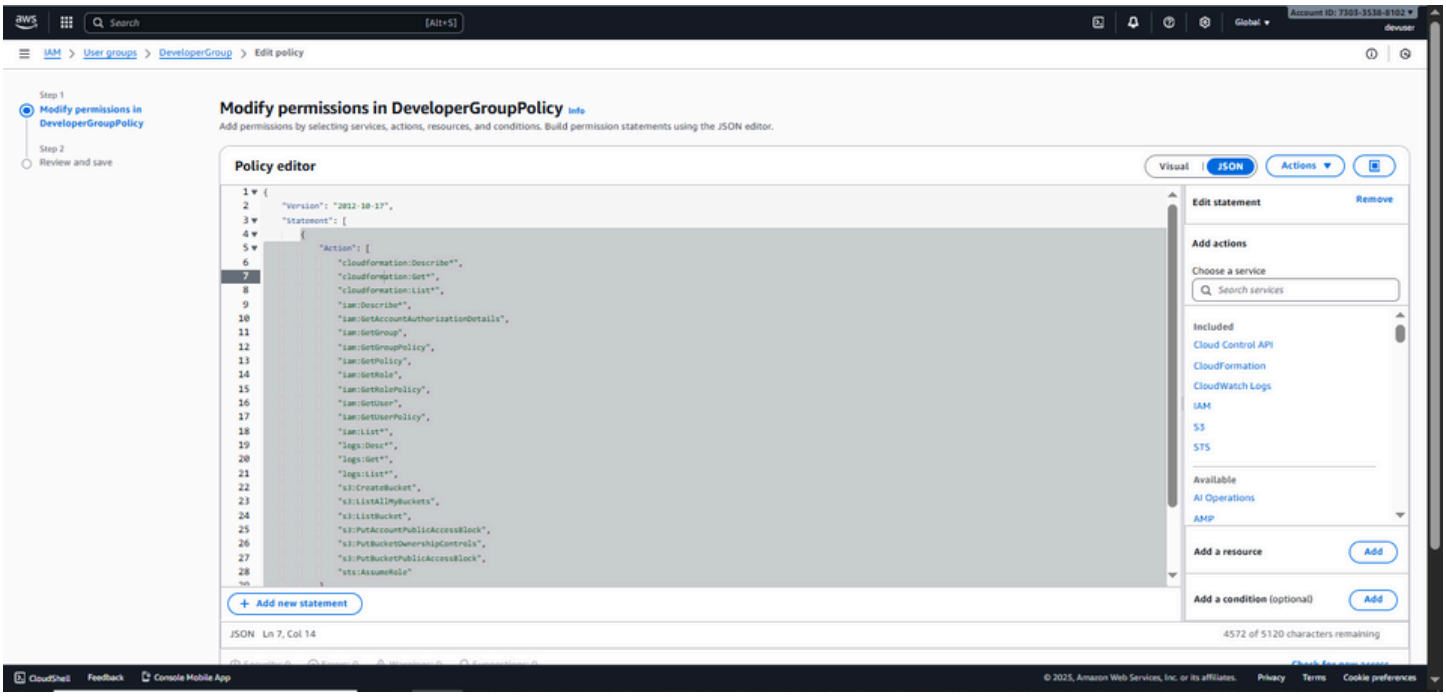


- **Ouverture de la politique DeveloperGroupPolicy afin d'examiner le fichier JSON.**

Analyse de la politique :
Il s'agit d'une politique basée sur l'identité (*Identity-Based Policy*) appliquée au groupe.
Actions autorisées : *Le document autorise les actions s3:ListAllMyBuckets, s3:CreateBucket et s3:ListBucket.*

- **Actions manquantes ou non autorisées :**L'analyse révèle l'absence totale de permissions ec2: (ce qui justifie les erreurs d'accès rencontrées sur le service EC2) ainsi que l'absence de la permission s3:PutObject au niveau global.

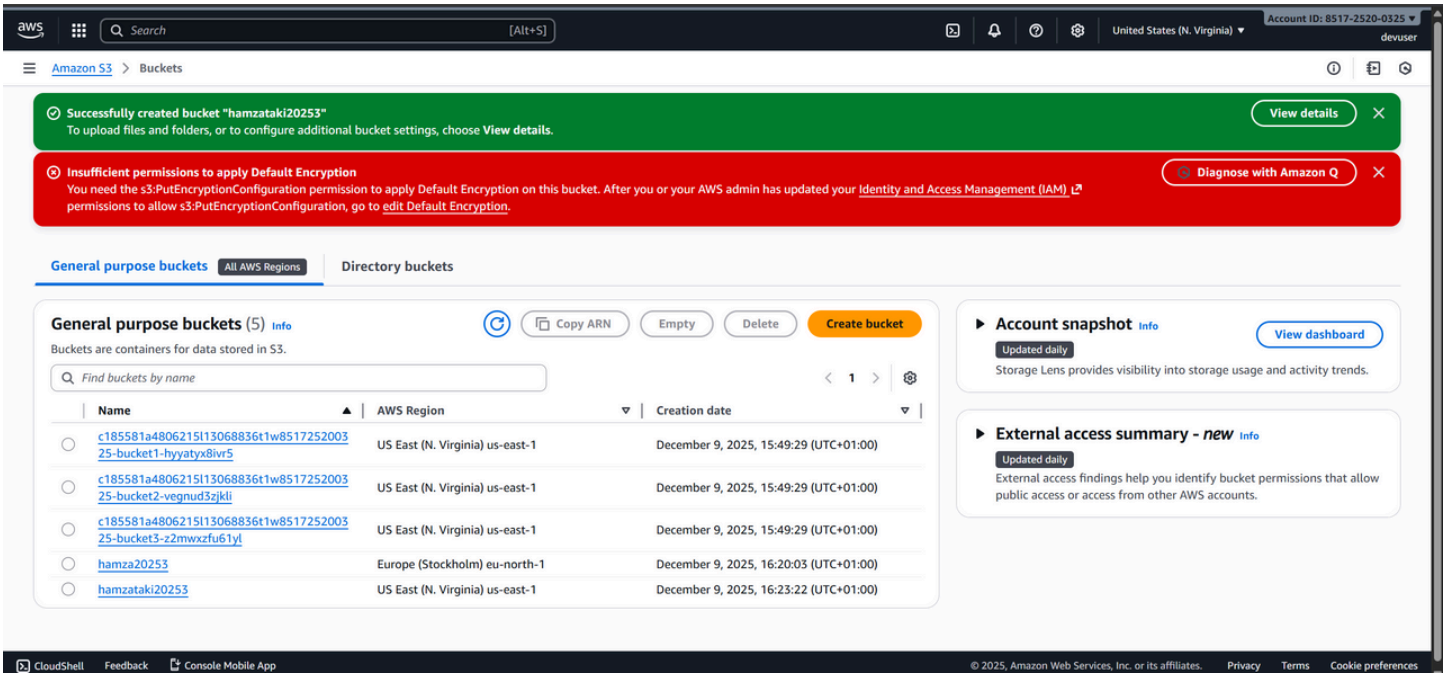
Cette configuration explique le comportement observé précédemment : l'utilisateur a le droit de voir la liste des buckets, mais il est bloqué pour toute autre opération administrative ou modification.



Tâche 4 : Tentative d’accès en écriture aux services AWS

Action 1 : Création d’un bucket

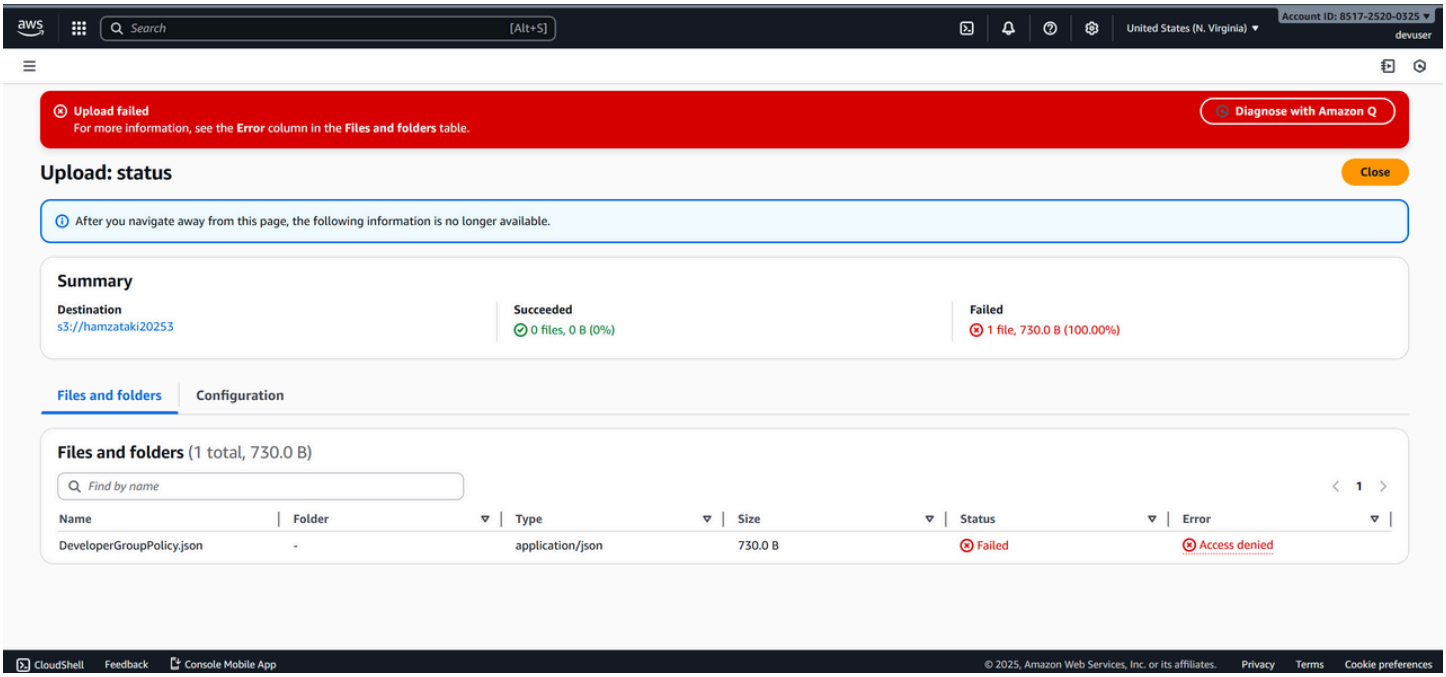
- **Étape :** Clic sur « Create bucket » puis création d’un bucket nommé *to4973*.



- **Résultat :** Succès.
- Raison : La politique *DeveloperGroupPolicy* autorise explicitement l’action *s3:CreateBucket*.

Action 2 : Téléversement d’un objet

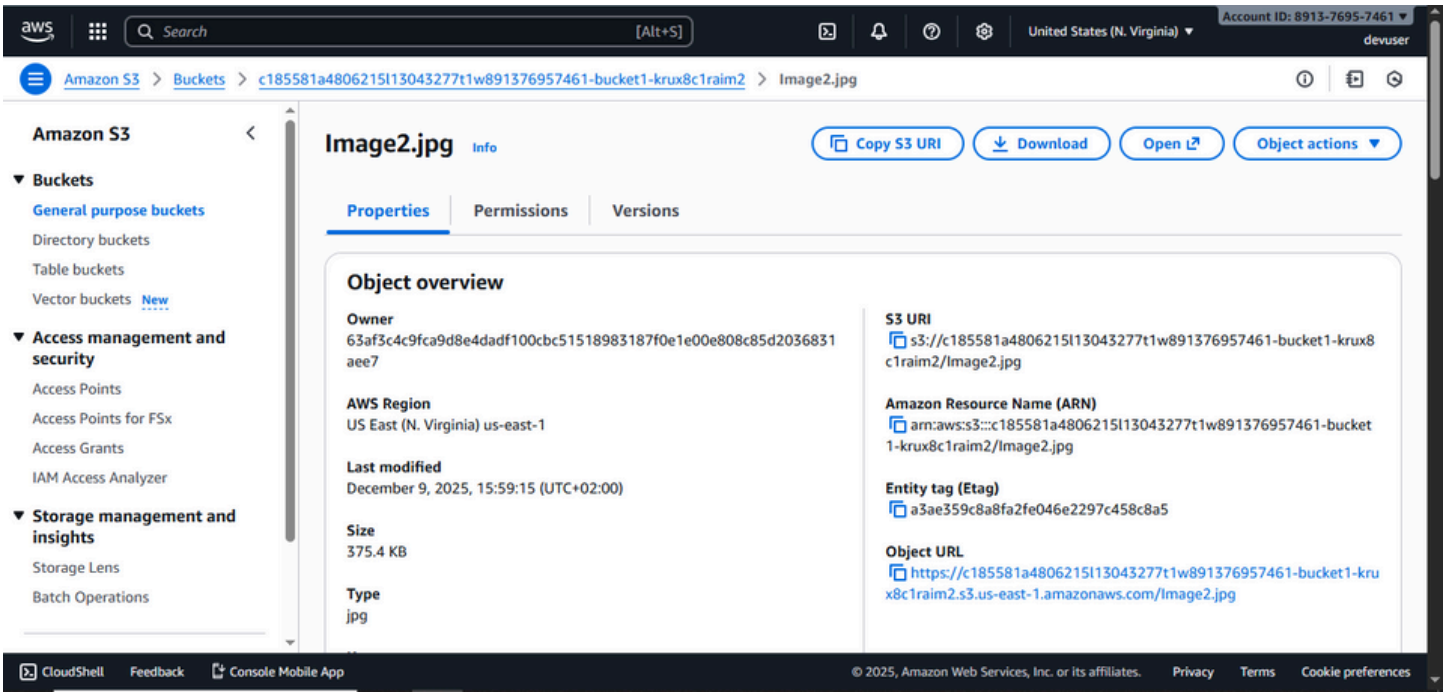
- **Étape :** Ouverture du bucket nouvellement créé, puis tentative de téléverser *Image1.jpg*.
- **Résultat :** Échec (message *Upload failed*).
- **Raison :** Bien que nous soyons propriétaires du bucket que nous venons de créer, la politique IAM associée à *devuser* ne contient pas l'autorisation *s3:PutObject*. Dans AWS, la permission de créer une ressource n'implique pas automatiquement la permission de modifier son contenu si la politique reste restrictive.

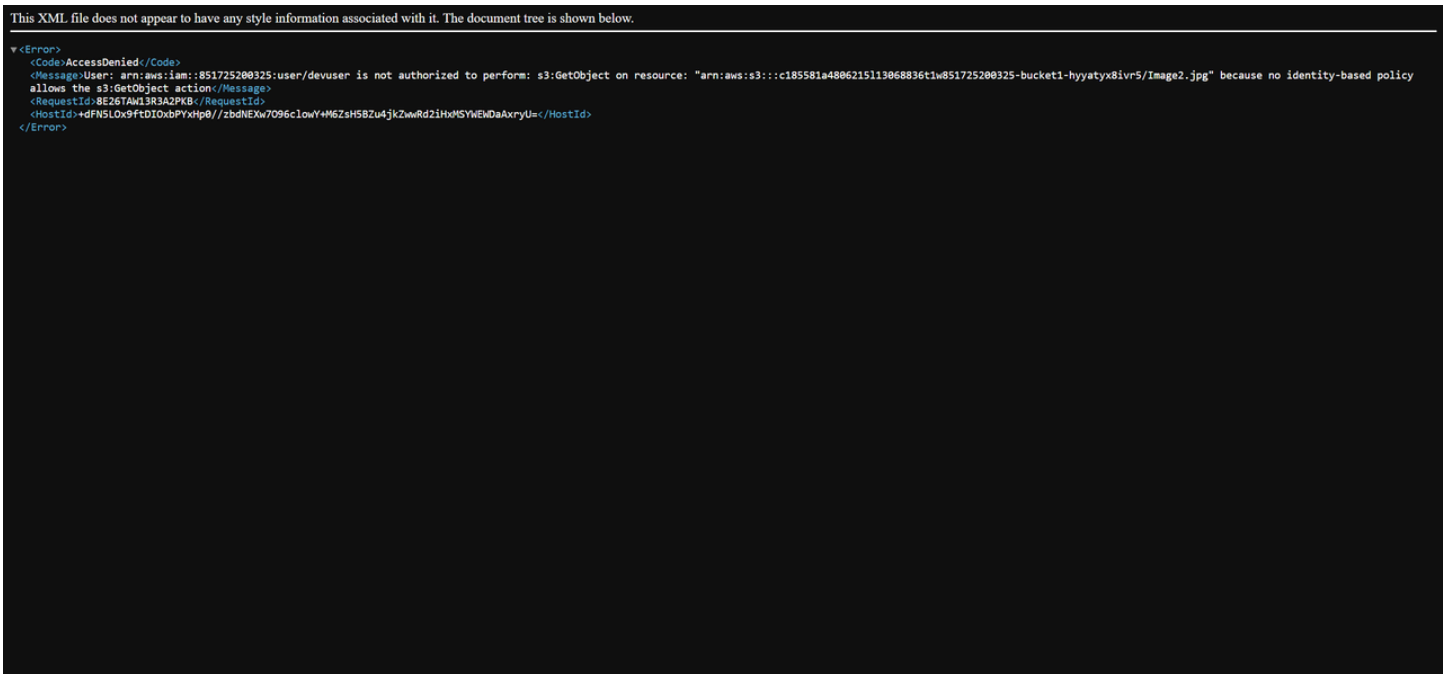


Tâche 5 : Assumer un rôle IAM

Test préalable : tentative de téléchargement avant d'assumer le rôle

- **Action :** Navigation vers *bucket1* puis tentative de téléchargement d'un fichier existant.
- **Résultat :** Échec (erreur *Access Denied*).
-

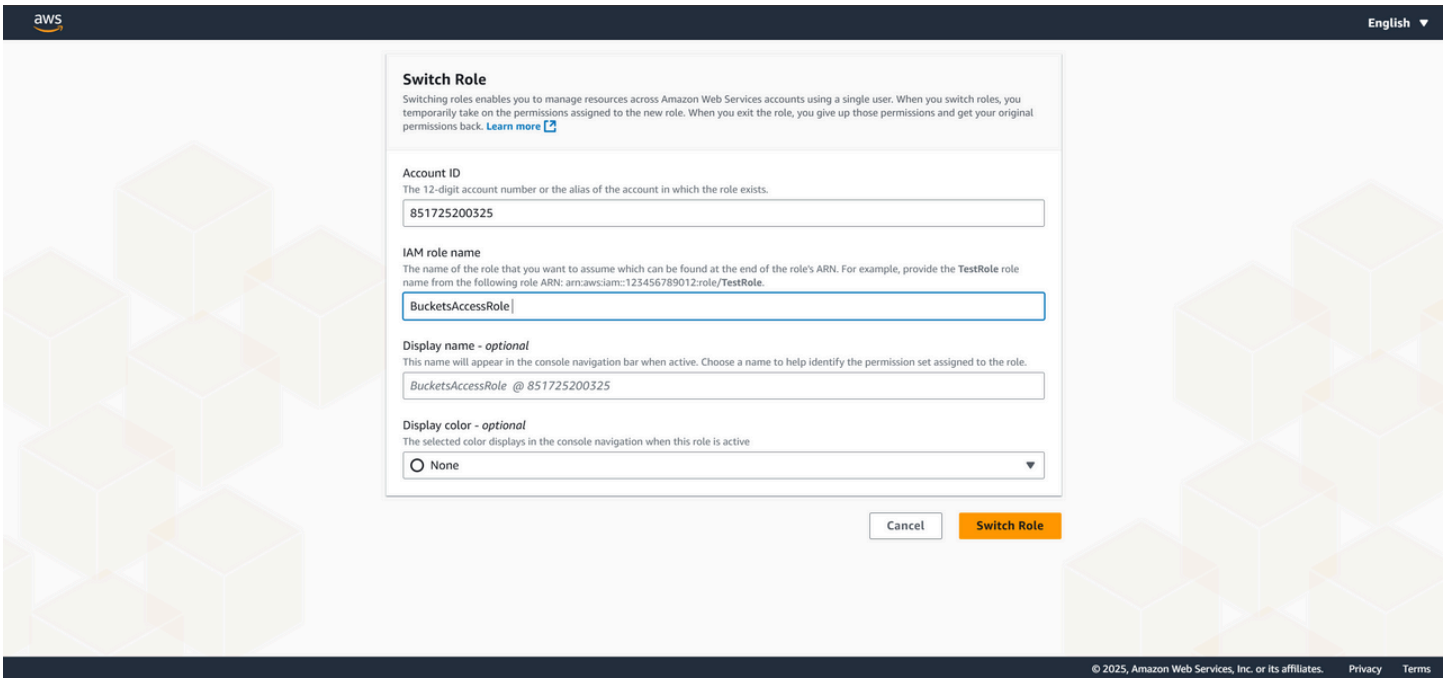




Raison :

La politique basée sur l'identité de *devuser* (*DeveloperGroupPolicy*) autorise *s3:ListAllMyBuckets* et *s3:ListBucket*, mais ne contient pas la permission *s3:GetObject*, indispensable pour télécharger un objet depuis un bucket. Cela confirme que les permissions par défaut de l'utilisateur limitent l'accès en lecture au contenu des objets, même s'il peut voir la liste des buckets.

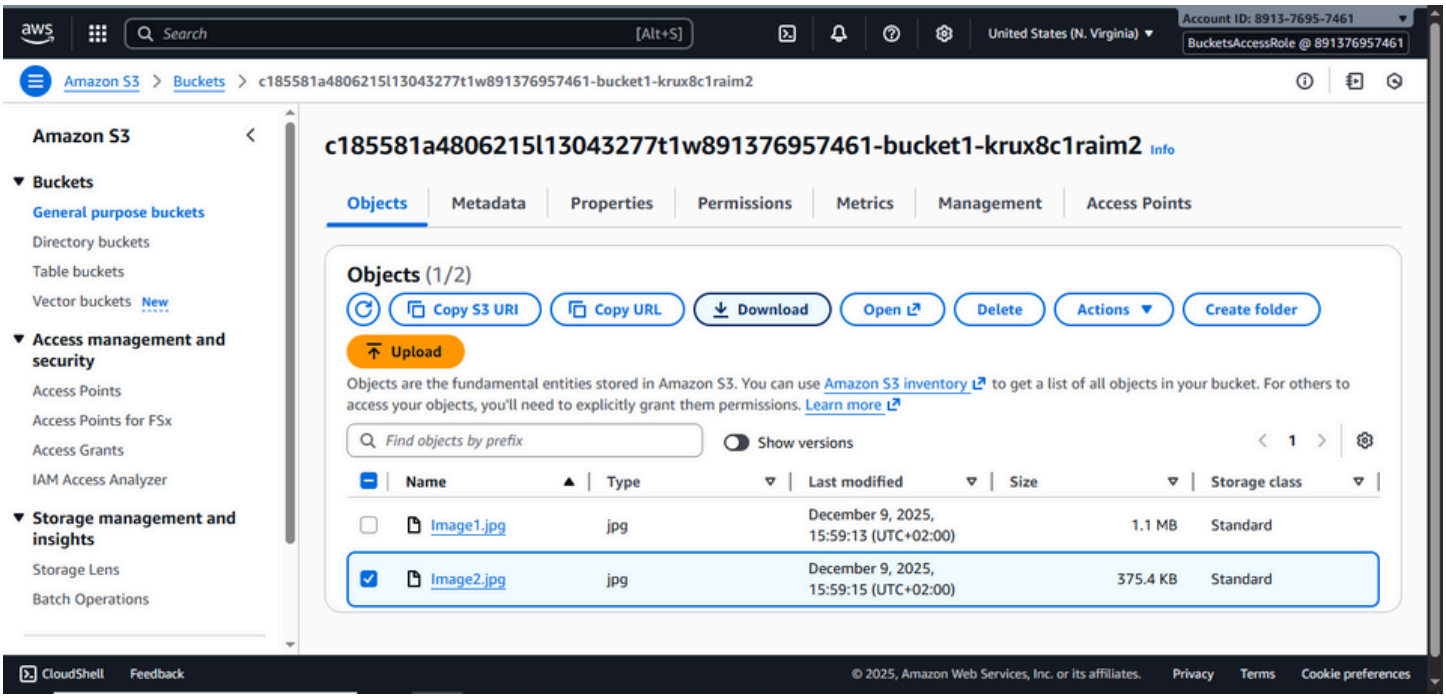
Action : (tu peux m'envoyer la suite pour la reformuler)

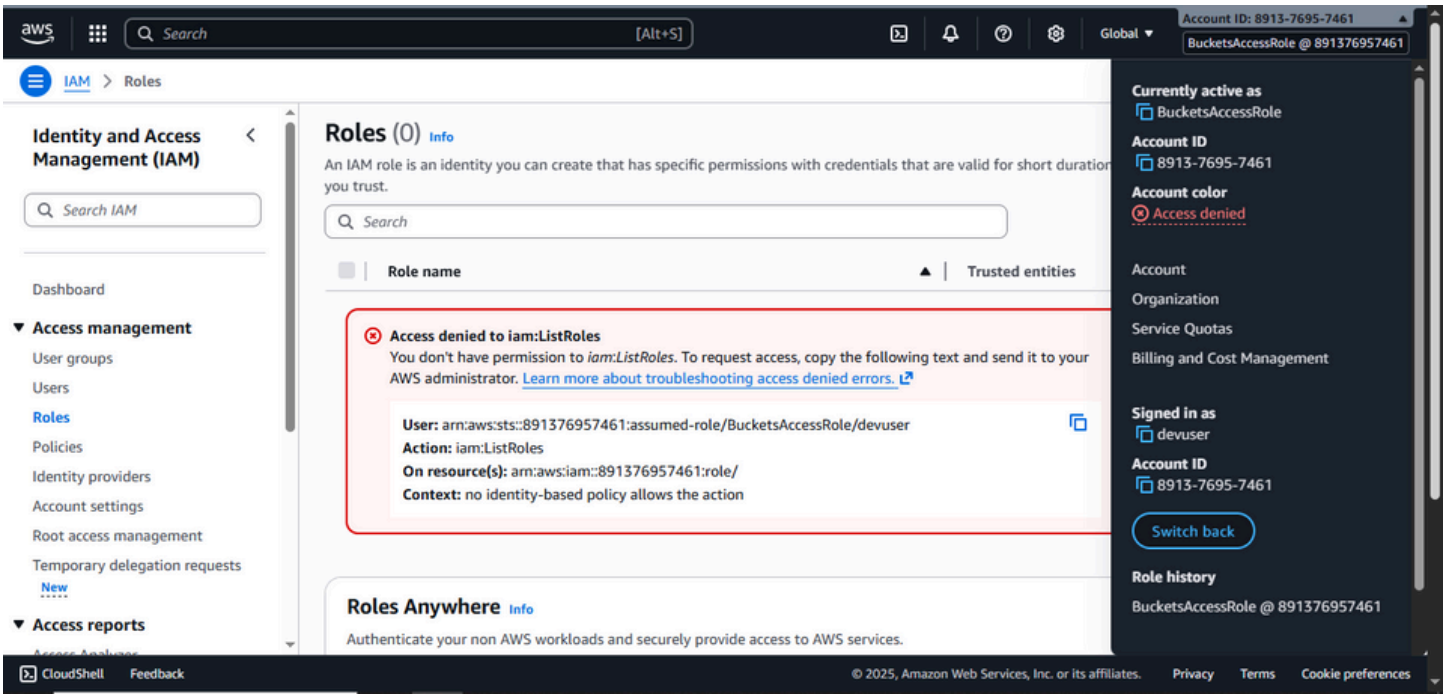


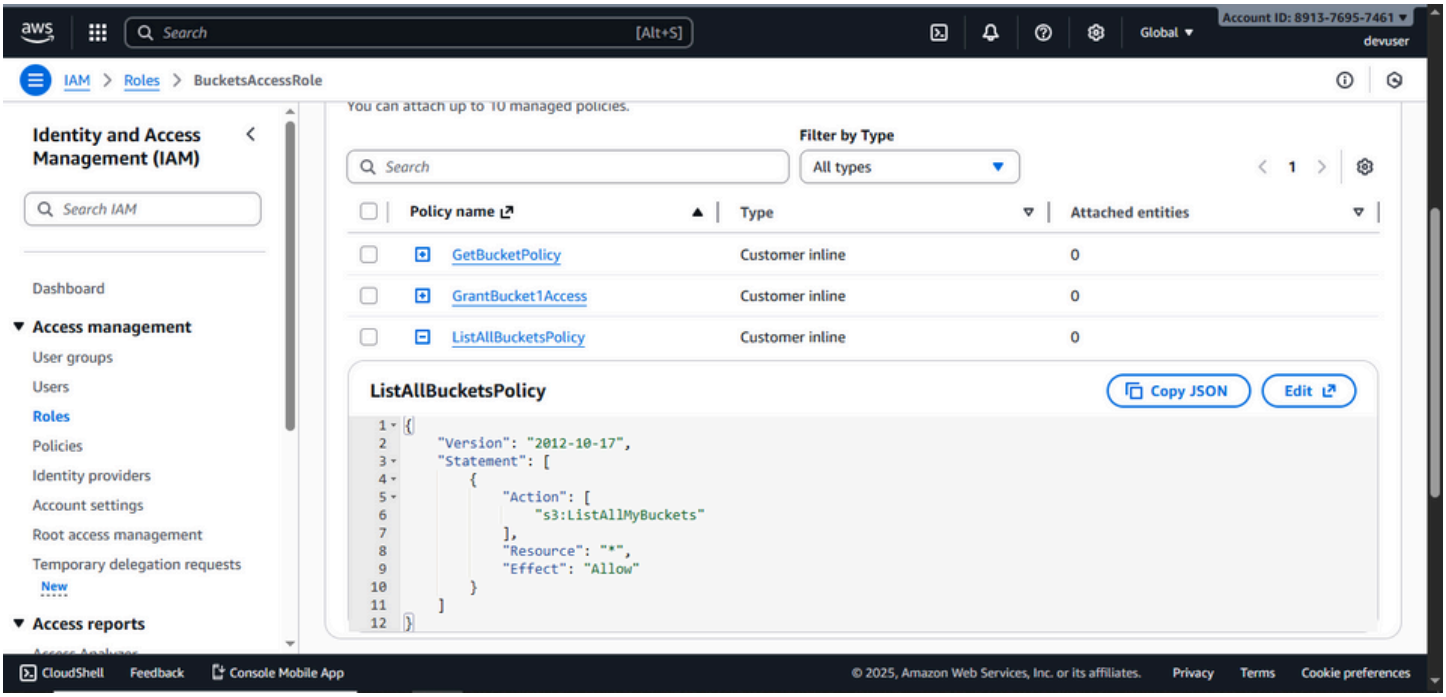
Action :

- Ouverture du menu utilisateur en haut à droite puis clic sur Switch Role.
- Saisie de l'ID du compte ainsi que du nom du rôle : BucketsAccessRole.
- Changement visible : L'identité affichée en haut à droite a été remplacée par *BucketsAccessRole*, confirmant que le rôle a bien été assumé.

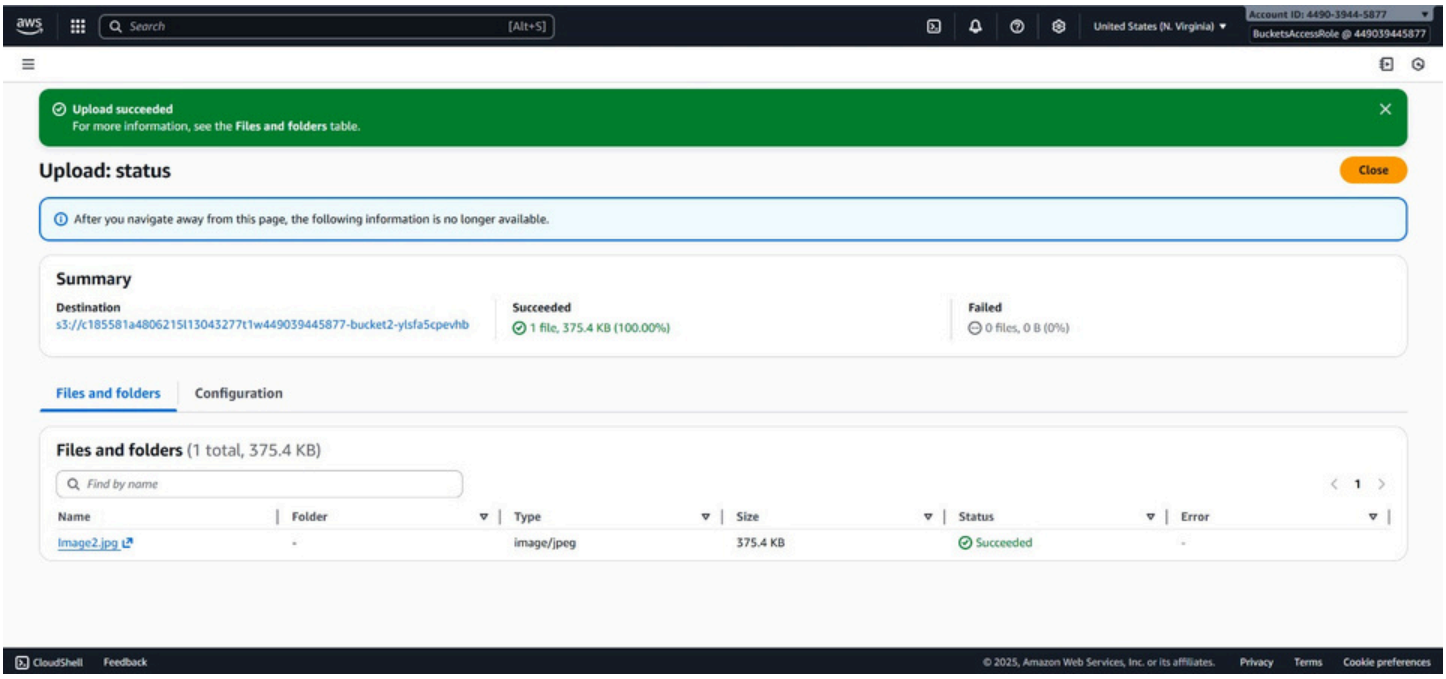
Test d'accès : (envoie-moi la suite et je te la reformule)







1. **Bucket 1 :** Le téléchargement des objets a fonctionné sans problème.
2. **Bucket 2 :** L'upload d'objets était autorisé, car le rôle disposait de la permission *s3:PutObject* spécifiquement sur *bucket2*.



Analyse :
En assumant le rôle, nous avons temporairement abandonné les permissions de *devuser* pour adopter celles de BucketsAccessRole. Ce rôle possédait une politique dédiée autorisant la lecture et/ou l'écriture sur certains buckets, des permissions que *devuser* ne détenait pas initialement.

Voici la reformulation :

Tâche 6 : Comprendre les politiques basées sur les ressources

Action :

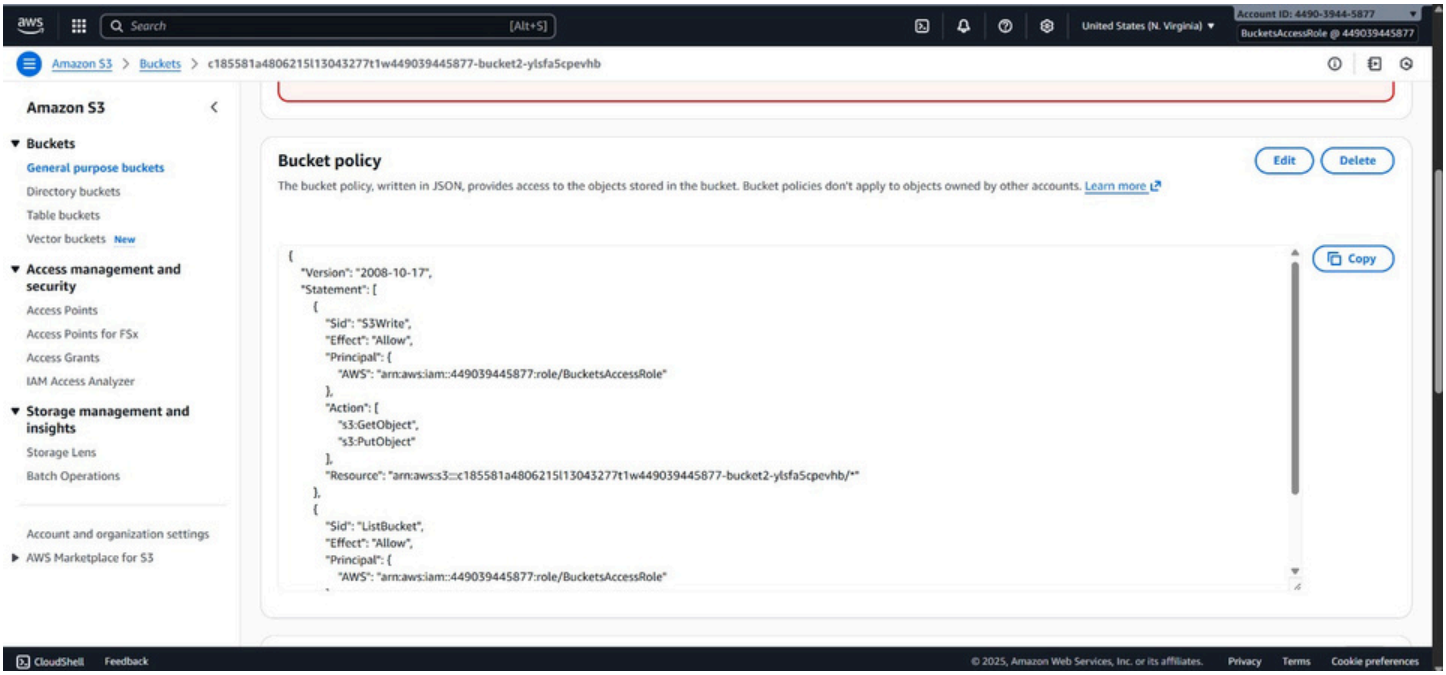
- En étant connecté avec le rôle *bucket-access-role*, nous avons navigué vers *Bucket 2* puis ouvert l'onglet **Permissions**.
- Nous avons consulté la **Bucket Policy**.

Analyse de la politique :
Il s'agit d'une **Resource-Based Policy**, c'est-à-dire une politique attachée directement à la ressource (le bucket S3), contrairement aux politiques basées sur l'identité qui sont liées à un utilisateur ou à un rôle IAM.

- La politique mentionnait explicitement *BucketsAccessRole* comme **Principal** autorisé.
- Cela confirme que l'accès à une ressource peut être accordé soit par la politique de l'utilisateur (Identity-Based Policy), soit par la politique de la ressource (Resource-Based Policy) lorsqu'il s'agit du même compte.
- Dans un contexte inter-comptes (cross-account), les deux types de politiques sont souvent nécessaires, mais dans ce laboratoire l'objectif principal était

d’observer comment le rôle obtenait l’autorisation via la bucket policy.

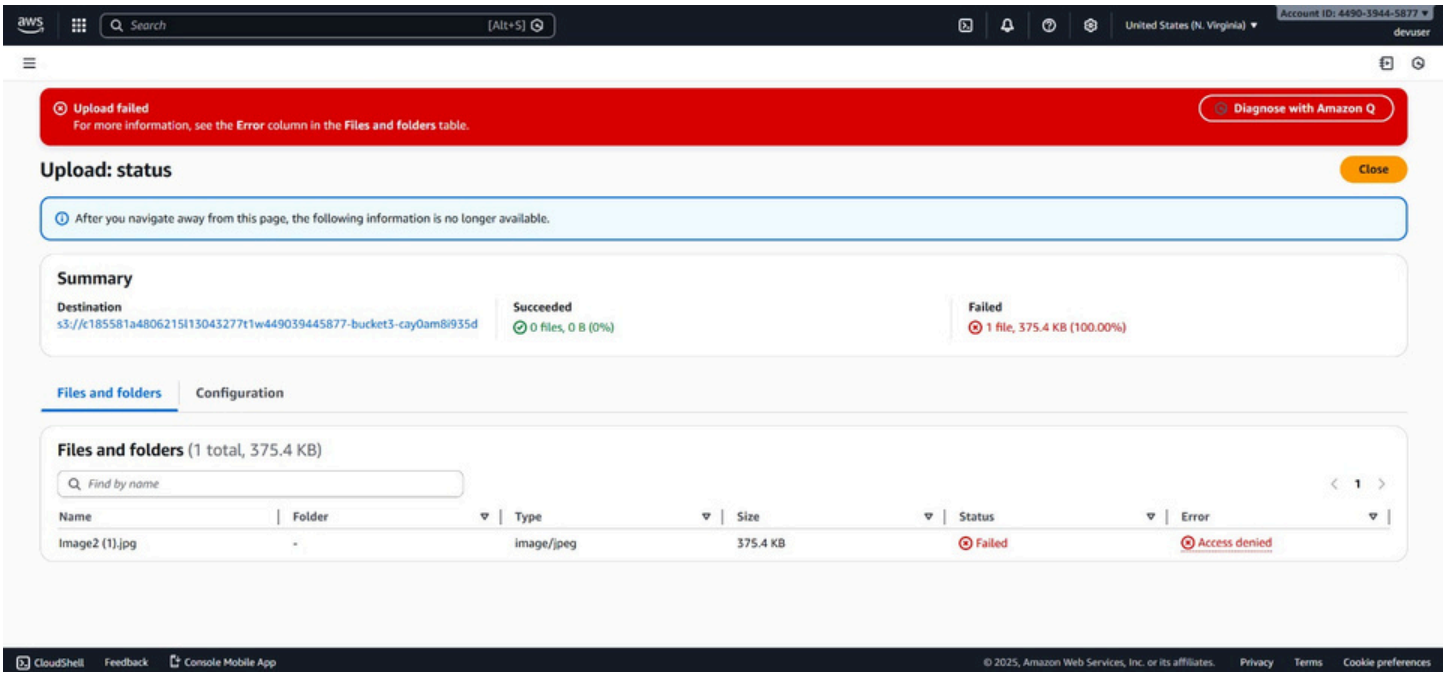
Si tu veux, envoie-moi la dernière partie (conclusion, résumé, etc.) et je la reformule aussi.

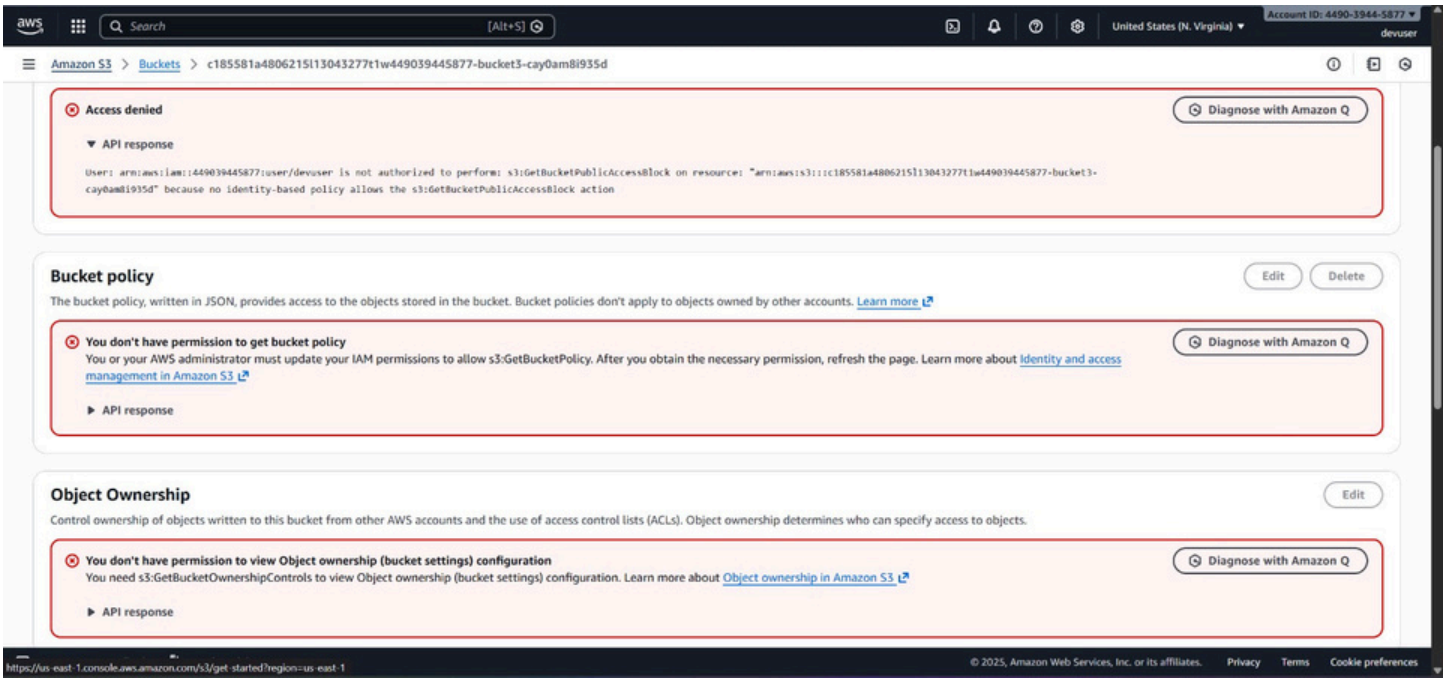


3. **Challenge Task:** : Téléversement dans Bucket 3

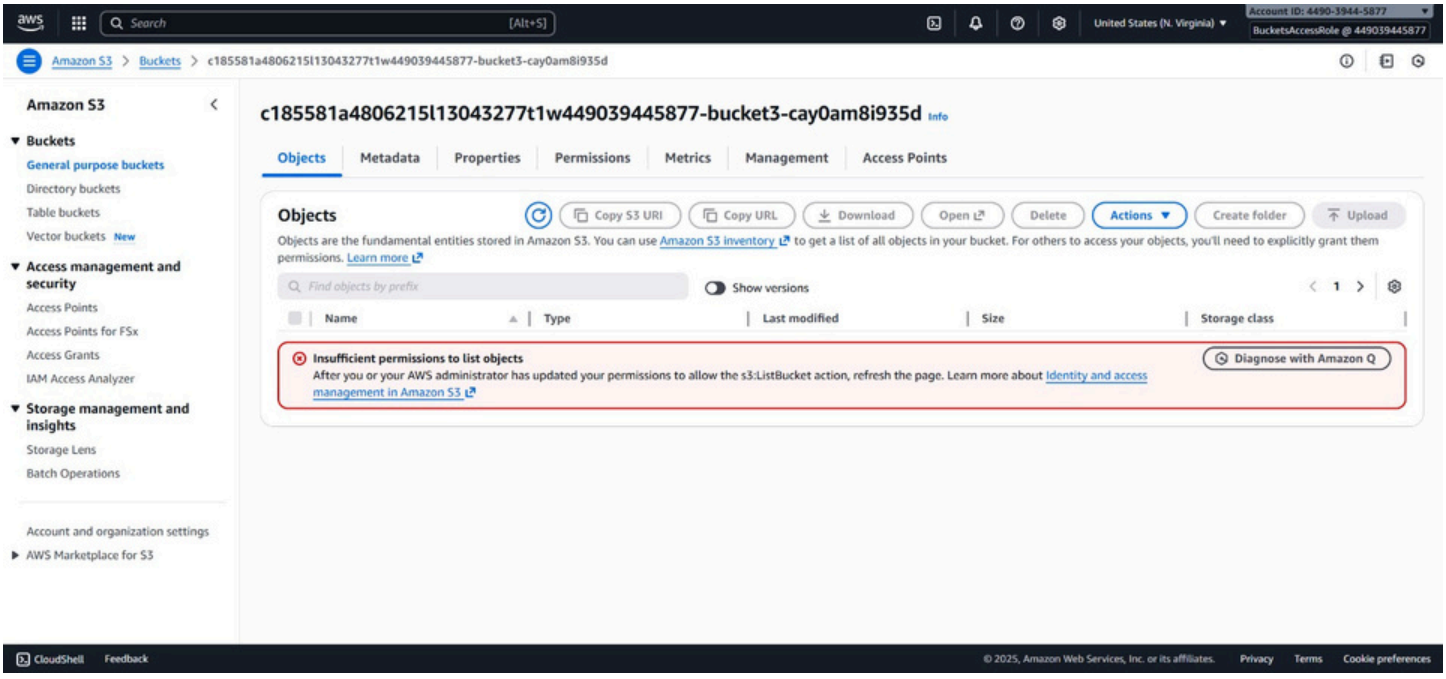
Objectif : Envoyer *Image2.jpg* dans *bucket3*.

Tentative 1 : En tant que *devuser*

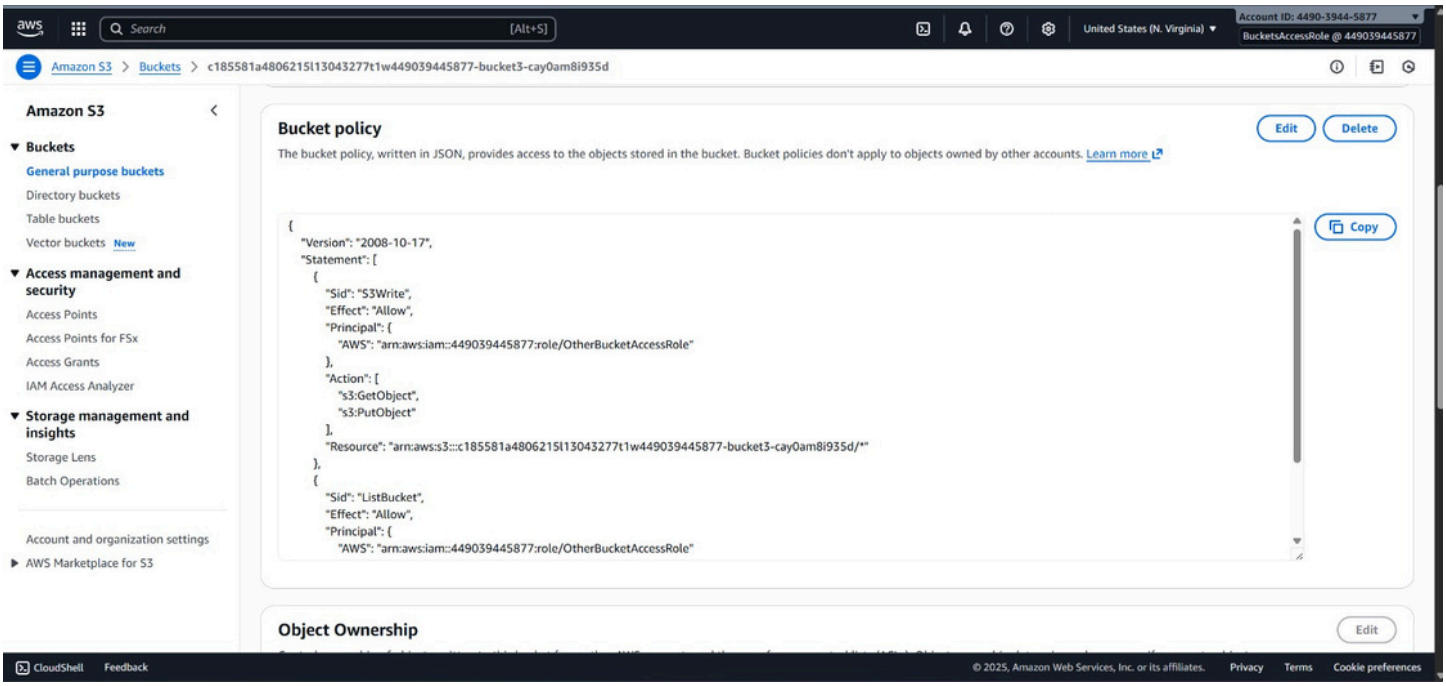




Tentative 2 : En utilisant le rôle *BucketAccessRole*

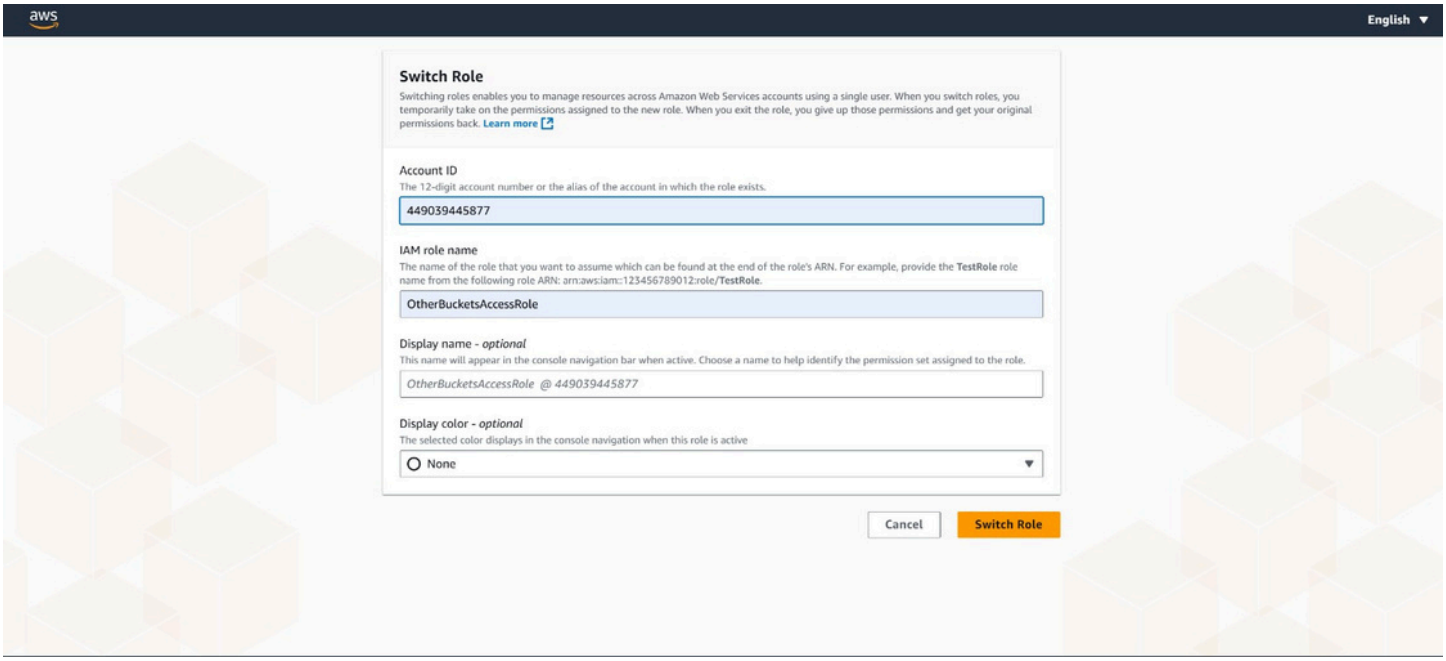


Résultat : Échec. La politique associée au rôle ne contenait aucune permission permettant d'interagir avec *bucket3*. Toutefois, nous pouvions consulter la bucket policy.



Solution :

1. Nous avons identifié un second rôle disponible dans le compte : **OtherBucketAccessRole**.
2. Nous avons de nouveau utilisé la fonctionnalité **Switch Role** pour passer du rôle *BucketsAccessRole* au rôle *OtherBucketAccessRole*.

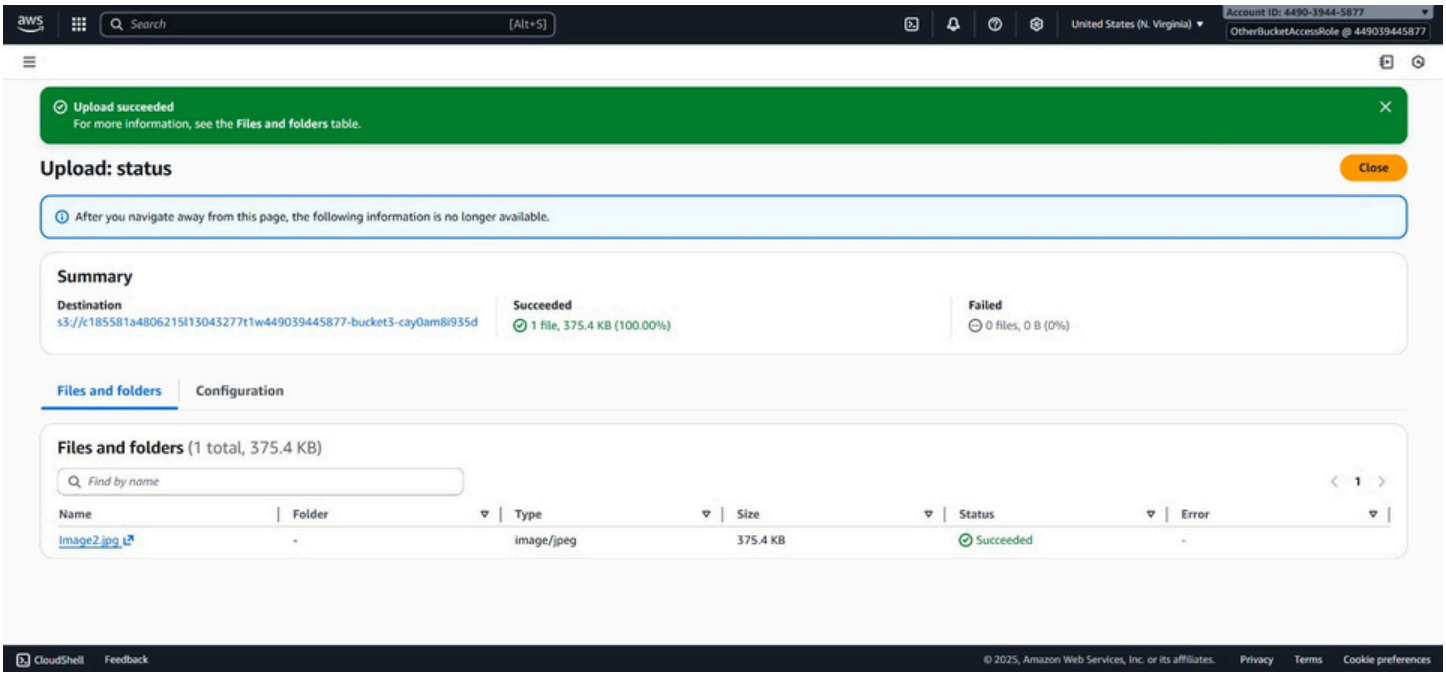


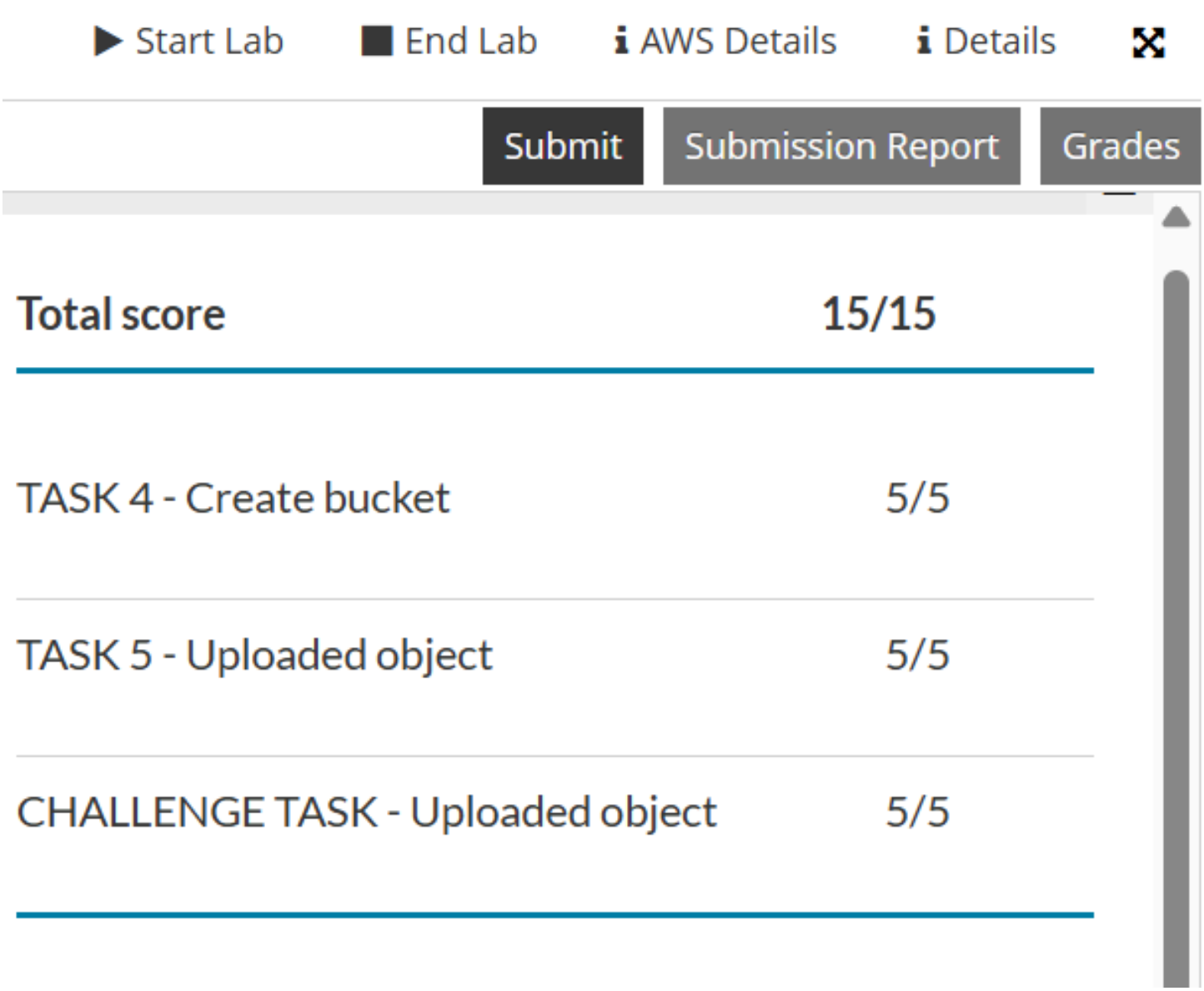
1. Nous avons navigué vers *bucket3*.
2. Nous avons cliqué sur **Upload**, sélectionné *Image2.jpg*, puis validé l'envoi.
3. **Résultat :** Succès.

Explication :

Le rôle **OtherBucketAccessRole** possédait une politique autorisant

explicitement l'action *s3:PutObject* sur la ressource *bucket3*. En assumant ce rôle spécifique, nous avons obtenu les permissions nécessaires pour réaliser la tâche.





Conclusion

Ce laboratoire a clairement démontré la flexibilité et la granularité du contrôle d'accès dans AWS :

1. **Les politiques d'identité** associées à *devuser* offraient un accès de base, mais limitaient les actions sensibles, comme la création d'instances EC2 ou l'écriture dans des buckets non autorisés.
2. **Les rôles IAM** nous ont permis d'élever temporairement nos permissions afin d'accomplir des tâches spécifiques (gestion ciblée de certains buckets) sans avoir à attribuer ces privilèges de manière permanente à l'utilisateur.
3. **Les politiques basées sur les ressources**, appliquées directement aux buckets S3, ont ajouté une couche supplémentaire de contrôle, en définissant précisément quels *principals* (utilisateurs ou rôles) étaient autorisés à interagir avec les données du bucket.