## Workshop : Maîtriser Amazon SNS pour les notifications et l'automatisation

### Objectif général

- Comprendre le fonctionnement d'Amazon SNS et son rôle dans les architectures AWS.
- Configurer des notifications multi-canaux (e-mail, SMS, Lambda).
- Intégrer SNS à S3 pour un cas d'usage pratique d'automatisation.

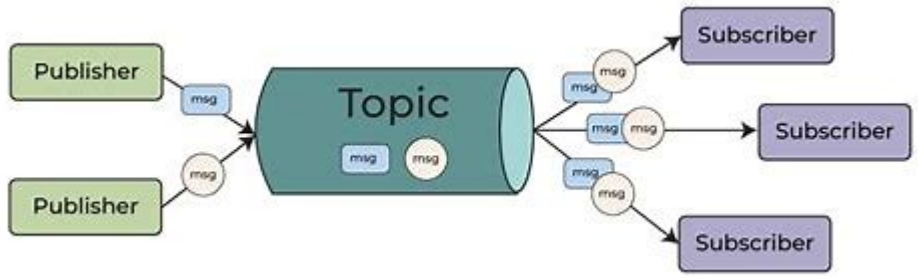
### Prérequis

- Compte AWS (Free Tier activé).
- Accès à la console AWS.
- Une adresse e-mail et un numéro de téléphone personnels pour tester.
- (Optionnel) AWS CLI installé et configuré.
- Un bucket S3 déjà créé (ou à créer pendant le TP).

## Qu'est-ce que le modèle pub/sub ?

Pub/sub est un modèle de communication où les émetteurs de messages (publishers) et les récepteurs de messages (subscribers) ne se parlent pas directement. Au lieu de ça, ils passent par un intermédiaire (ici, un "topic" dans SNS) qui distribue les messages. C'est comme un système de messagerie décentralisé.

- Publication (pub) : Quelqu'un envoie un message sans savoir qui le recevra.
- Abonnement (sub) : Quelqu'un s'inscrit pour recevoir ces messages quand ils sont envoyés.




## Explication avec une analogie : le journal

### Imagine un journal local :

1. Publisher (l'éditeur) : Le journal écrit une nouvelle (ex. : "Il pleut aujourd'hui").
2. Topic (le journal lui-même) : C'est le canal qui transporte la nouvelle. L'éditeur ne sait pas qui lit le journal.

3. Subscribers (les abonnés) : Les lecteurs qui ont souscrit un abonnement (par exemple, toi, ton voisin, une bibliothèque). Ils reçoivent le journal automatiquement.

- L'éditeur ne va pas livrer le journal à chaque personne individuellement ; il le publie une fois, et le système (les livreurs) s'occupe de le distribuer aux abonnés.
- Si quelqu'un n'est pas abonné, il ne reçoit rien

**Qu'est-ce que SNS ?**

Amazon Simple Notification Service (SNS) est un service de messagerie basé sur le modèle publication/abonnement (pub/sub). Il permet d'envoyer des notifications à des utilisateurs ou de déclencher des actions dans des systèmes en publiant des messages vers des "topics" (sujets), auxquels des abonnés (subscribers) sont connectés.

**Analogie simple :**
"Imaginez une radio : une station (le publisher) diffuse un message à travers une fréquence (le topic), et tous ceux qui sont branchés sur cette fréquence (les subscribers) reçoivent le signal. SNS fonctionne pareil !"

**Exemple rapide :**
Un fichier est uploadé dans S3 (publisher) → SNS envoie un e-mail à un admin et déclenche une fonction Lambda (subscribers).

=> Dans SNS, c'est pareil : un **publisher** envoie un message à un **topic**, et tous les **subscribers** inscrits à ce topic reçoivent le message

## Comment ça marche dans SNS ?

1. **Publisher** :
    - C'est la source qui envoie un message. Ça peut être :
        - Un service AWS (ex. : S3 quand un fichier est uploadé).
        - Toi, manuellement via la console ou un script.
    - Exemple : "Un fichier test.txt a été uploadé dans S3."
2. **Topic** :
    - C'est une sorte de "boîte aux lettres" virtuelle créée dans SNS (ex. : "UploadAlerts").
    - Le publisher envoie son message au topic, pas directement aux destinataires.
3. **Subscribers** :
    - Ce sont les "abonnés" qui ont dit : "Je veux recevoir les messages de ce topic."
    - Ça peut être :
        - Une adresse e-mail.
        - Un numéro de téléphone (SMS).
        - Une fonction Lambda.

        - Une URL HTTP.
    - Exemple : Ton e-mail et ton téléphone sont abonnés à "UploadAlerts". Quand S3 publie "Nouveau fichier !", tu reçois un e-mail ET un SMS.

## Pourquoi c'est utile ?

- **Déconnexion** : Le publisher (ex. : S3) n'a pas besoin de savoir qui reçoit le message ni combien de personnes sont abonnées. Il envoie et basta.
- **Flexibilité** : Tu peux ajouter ou retirer des subscribers (ex. : un nouvel e-mail) sans toucher au publisher.
- **Efficacité** : Un seul message peut atteindre plusieurs destinataires en même temps (e-mail, SMS, Lambda) sans duplication d'effort.

## Exemple concret avec SNS

Imaginons un scénario lié à S3 (puisque tu l'as enseigné) :

1. Tu crées un topic SNS appelé "FileUpdates".
2. Tu t'abonnes au topic avec :
    - Ton e-mail (ex. : toi@example.com).
    - Ton numéro de téléphone (ex. : +33123456789).
    - Une fonction Lambda pour traiter le fichier.
3. Tu configures S3 pour publier un message dans "FileUpdates" quand un fichier est uploadé.
4. Tu uploade "photo.jpg" dans S3.
5. Résultat :
    - Ton e-mail reçoit : "Nouveau fichier : photo.jpg".
    - Ton téléphone reçoit un SMS : "Nouveau fichier : photo.jpg".
    - Lambda s'exécute pour, par exemple, redimensionner l'image.

Ici :

- **Publisher** = S3 (il publie l'événement).
- **Topic** = "FileUpdates" (le canal).
- **Subscribers** = e-mail, SMS, Lambda (ils reçoivent tous le même message).

## Différence avec une communication directe

Sans pub/sub, S3 devrait :

- Envoyer un e-mail à toi directement.
- Puis envoyer un SMS séparément.
- Puis appeler Lambda manuellement. Avec pub/sub, S3 envoie UN message au topic, et SNS distribue à tous les abonnés automatiquement.

**Démo pas à pas**

1. **Créer un bucket S3** :
    - Aller dans **S3** → "Create Bucket".
    - Nom : sns-workshop-bucket-[votre-nom] (unique, ex. : sns-workshop-bucket-amel2025).
    - Région : Choisir une région (ex. : us-east-1).
    - Laisser les paramètres par défaut (pas de chiffrement ou versionnage nécessaire ici).
    - Cliquer sur "Create Bucket".
2. **Créer un topic SNS** :
    - Aller dans **SNS** → "Create Topic".
    - Nom : S3UploadAlerts.
    - Type : Standard.
    - Cliquer sur "Create Topic".
    - Noter l'ARN du topic (ex. : arn:aws:sns:eu-west-1:522814729956:S3UploadAlerts).
3. **Ajouter un abonnement e-mail au topic** :
    - Dans le topic S3UploadAlerts → "**Créer un abonnement**".
    - Protocole : "Email".
    - Endpoint : Entrer ton adresse e-mail (ex. : toi@example.com).
    - Cliquer sur "Create Subscription".
    - Vérifier ta boîte mail (y compris spams) et cliquer sur le lien "Confirm subscription" envoyé par AWS.
4. **Configurer S3 pour publier dans SNS** :
    - Retourner dans **S3** → Sélectionner sns-workshop-bucket-[votre-nom].
    - Onglet "Properties" → Section "Event notifications" → "Create event notification".
    - Nom de l'événement : FileUploadEvent.
    - Type d'événement : "All object create events" (couvre les uploads).
    - Destination : "SNS Topic".
    - Topic : Sélectionner S3UploadAlerts dans la liste.
    - Cliquer sur "Save changes".
5. **Tester l'intégration** :
    - Aller dans le bucket S3 → "Upload" → Ajouter un fichier (ex. : test.txt ou une image).
    - Attendre 10-20 secondes.
    - Vérifier ta boîte mail (et spams) pour un e-mail de SNS avec des détails sur l'upload (ex. : nom du fichier, bucket, heure).