TP3 : OpenStack Swift – Stockage Objet et Configuration Avancée

---

Objectifs pédagogiques

- Comprendre le rôle et le fonctionnement du service **Swift**.
- Explorer les **concepts de stockage objet**.
- Créer et configurer différents **types de conteneurs** (privés, publics, versionnés).
- Tester les mécanismes de **réplication et de durabilité**.
- Vérifier le fonctionnement et la résilience du système.

Pré-requis

- Une installation OpenStack fonctionnelle (DevStack ou environnement académique).
- Un projet utilisateur configuré avec accès à Horizon ou au client OpenStack.
- Une image d'authentification (via Keystone) fonctionnelle.
- Un minimum de 1 Go d'espace disque libre.

Partie 1 – Création et gestion de conteneurs

Objectif

Comprendre la structure logique de Swift : **compte** → **conteneur** → **objet**.

Étapes

1. Créer deux conteneurs différents :

    - Un conteneur pour des **images**.
    - Un conteneur pour des **sauvegardes**.
2. Explorer les options de visibilité :
    - Conteneur **privé** (accessible uniquement à l'utilisateur).
    - Conteneur **public** (accès via URL publique).
3. Définir une **politique de stockage** différente pour chaque conteneur (si disponible dans votre déploiement).

Vérifications

- Vérifier que les conteneurs apparaissent bien dans l'interface.
- Tester l'accès à un conteneur public via navigateur.
- Confirmer qu'un conteneur privé demande une authentification.

Partie 2 – Téléversement et gestion d'objets

Objectif

Apprendre à stocker, organiser et récupérer des fichiers dans Swift.

Étapes

1. Envoyer plusieurs fichiers dans chaque conteneur :
    - Documents texte ou PDF.
    - Fichiers images.
    - Archives compressées (.zip ou .tar).
2. Ajouter des **métadonnées** à certains objets (exemple : auteur, type, date).
3. Modifier ou remplacer un objet existant.
4. Supprimer un fichier obsolète.

Vérifications

- Vérifier la liste des objets dans chaque conteneur.
- Vérifier que les métadonnées sont bien enregistrées.
- Confirmer que les fichiers supprimés ne sont plus accessibles.

Partie 3 – Versioning et politiques de cycle de vie

Objectif

Mettre en place la **gestion des versions** pour un conteneur et comprendre son utilité.

Étapes

1. Créer un conteneur “versionné” pour stocker des documents modifiables.
2. Activer la fonctionnalité de **versioning** (via Horizon ou configuration).
3. Importer plusieurs versions d’un même fichier (même nom, contenu différent).
4. Identifier comment Swift gère les anciennes versions.
5. Étudier les politiques de rétention ou d’expiration automatiques (si disponibles).

Vérifications

- Lister les versions d’un même fichier.
- Vérifier que les anciennes versions sont récupérables.
- Observer le comportement du conteneur lors de la suppression d’une version.

Partie 4 – Réplication et durabilité

Objectif

Observer comment Swift assure la **tolérance aux pannes** et la **résilience des données**.

Étapes

1. Identifier la **configuration de réplication** du cluster Swift (nombre de copies).
2. Explorer le concept de **ring** :
    - Comprendre comment Swift détermine où stocker chaque objet.
    - Étudier la répartition des partitions sur les nœuds.
3. Simuler (ou imaginer) la perte d'un nœud de stockage :
    - Que se passe-t-il pour les objets répliqués ?
    - Comment Swift restaure les copies manquantes ?

Vérifications

- Vérifier que chaque objet est présent sur plusieurs nœuds (si possible).
- Vérifier la cohérence après une opération de réplication.
- Comprendre comment Swift équilibre automatiquement la charge.