Mise en place SSO avec Keycloak

Installation keycloak et intégration pour grafana

Déploiement SSO Keycloak avec intégration Grafana

? Objectif

Mettre en place une authentification unique (SSO) avec Keycloak pour centraliser l'authentification de tous les services (Grafana, Jenkins, Vault, etc.) avec possibilité d'activer la 2FA ultérieurement.

?? Architecture finale

```
Internet (HTTPS) → Nginx (reverse proxy + SSL termination)
                    ↓
                    ├─→ sso.sanjy.fr:443 → Keycloak:30180 (HTTP interne)
                    └─→ grafana.sanjy.fr:443 → Grafana:3000 (HTTP interne)

Réseau Podman interne "keycloak_sso" :
  - Keycloak (conteneur)
  - Grafana (conteneur)
  → Communication directe keycloak:8080
```

? 1. Installation Keycloak

Structure des répertoires

```
mkdir -p ~/project/keycloak
cd ~/project/keycloak
```

podman-compose.yaml (Keycloak)

```
version: '3.8'

services:
  keycloak:
    container_name: keycloak
    image: docker.io/keycloak/keycloak:latest
    command: start-dev
    environment:
      KC_BOOTSTRAP_ADMIN_USERNAME: admin
      KC_BOOTSTRAP_ADMIN_PASSWORD: <VOTRE_MOT_DE_PASSE>
      KC_PROXY_HEADERS: xforwarded
      KC_HOSTNAME_URL: https://sso.sanjy.fr
      KC_HTTP_ENABLED: "true"
    ports:
      - "30180:8080"
    volumes:
      - data:/opt/keycloak/data
    restart: unless-stopped
    networks:
      - sso

networks:
  sso:
    driver: bridge

volumes:
  data:
```

Configuration DNS

Ajouter un enregistrement A dans OVH :

- **Type** : A
- **Sous-domaine** : sso

- **Cible** : IP de mon VPS

Génération certificat SSL

```
sudo certbot certonly --manual --preferred-challenges dns -d sso.sanjy.fr
```

Suivre les instructions pour ajouter l'enregistrement TXT dans OVH.

Configuration Nginx pour Keycloak

Créer `/etc/nginx/conf.d/sso.conf` :

```
# Redirection HTTP → HTTPS
server {
    listen 80;
    listen [::]:80;
    server_name sso.sanjy.fr;
    return 301 https://$server_name$request_uri;
}

# HTTPS
server {
    listen 443 ssl http2;
    listen [::]:443 ssl http2;
    server_name sso.sanjy.fr;

    # Certificats SSL
    ssl_certificate /etc/letsencrypt/live/sanjy.fr/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/sanjy.fr/privkey.pem;

    # Paramètres SSL
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers HIGH:!aNULL:!MD5;
    ssl_prefer_server_ciphers on;
    add_header Strict-Transport-Security "max-age=31536000" always;

    # Reverse proxy vers Keycloak
    location / {
        proxy_pass http://localhost:30180;
```

```
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto https;
        proxy_set_header X-Forwarded-Host $host;
        proxy_set_header X-Forwarded-Port 443;

        proxy_buffering off;
        proxy_http_version 1.1;

        # WebSocket support
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";

        # Timeouts
        proxy_connect_timeout 90;
        proxy_send_timeout 90;
        proxy_read_timeout 90;
    }
}
```

HTTPS

server { listen 443 ssl http2; listen [::]:443 ssl http2; server_name sso.sanjy.fr;

```
# Certificats SSL
ssl_certificate /etc/letsencrypt/live/sanjy.fr/fullchain.pem;
ssl_certificate_key /etc/letsencrypt/live/sanjy.fr/privkey.pem;

# Paramètres SSL
ssl_protocols TLSv1.2 TLSv1.3;
ssl_ciphers HIGH:!aNULL:!MD5;
ssl_prefer_server_ciphers on;
add_header Strict-Transport-Security "max-age=31536000" always;

# Reverse proxy vers Keycloak
location / {
    proxy_pass http://localhost:30180;
    proxy_set_header Host $host;
```

```
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto https;
    proxy_set_header X-Forwarded-Host $host;
    proxy_set_header X-Forwarded-Port 443;

    proxy_buffering off;
    proxy_http_version 1.1;

    # WebSocket support
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection "upgrade";

    # Timeouts
    proxy_connect_timeout 90;
    proxy_send_timeout 90;
    proxy_read_timeout 90;
}
```

}

````
### Démarrage

```bash
# Tester la config Nginx
sudo nginx -t

# Recharger Nginx
sudo systemctl reload nginx

# Démarrer Keycloak
podman-compose up -d

# Vérifier les logs
podman-compose logs -f keycloak
````

Attendre de voir "Keycloak started" dans les logs.

````

? 2. Configuration Keycloak

Accès à l'interface

Accéder à `https://sso.sanjy.fr` et se connecter avec :

- **Username** : admin
- **Password** : celui défini dans KC_BOOTSTRAP_ADMIN_PASSWORD

Création du Realm

1. Cliquer sur **"master"** en haut à gauche
2. Cliquer sur **"Create realm"**
3. **Realm name** : `sanjy.fr`
4. Cliquer sur **"Create"**

Création d'un utilisateur

1. Menu de gauche : **Users**
2. Cliquer sur **"Add user"**
3. Renseigner :
   - **Username** : votre nom d'utilisateur
   - **Email** : votre email
   - **First name** : votre prénom
   - **Last name** : votre nom
   - Activer **"Email verified"** (toggle en bleu)
4. Cliquer sur **"Create"**
5. Onglet **"Credentials"** → **"Set password"**
   - Entrer le mot de passe (2 fois)
   - **Désactiver** "Temporary"
   - Cliquer sur **"Save"** puis confirmer

Création du client OIDC Grafana

1. Menu de gauche : **Clients**
2. Cliquer sur **"Create client"**

Écran 1 - General Settings :

- **Client type** : OpenID Connect
- **Client ID** : `grafana`
- Cliquer sur **"Next"**

Écran 2 - Capability config :

- Activer **"Client authentication"** (toggle en bleu)
- Cliquer sur **"Next"**

Écran 3 - Login settings :

- **Root URL** : `https://grafana.sanjy.fr`
- **Valid redirect URIs** : `https://grafana.sanjy.fr/*`
- **Web origins** : `https://grafana.sanjy.fr`
- Cliquer sur **"Save"**

Récupérer le Client Secret :

1. Onglet **"Credentials"**
2. **Copier le "Client secret"** (le sauvegarder précieusement)

? 3. Intégration Grafana

Configuration DNS

Ajouter un enregistrement A dans OVH :

- **Type** : A
- **Sous-domaine** : grafana
- **Cible** : IP de mon VPS

Modification du podman-compose.yaml (Grafana)

Ajouter Grafana au réseau Keycloak et configurer OAuth :

```
grafana:
  image: docker.io/grafana/grafana:latest
  container_name: grafana
  restart: unless-stopped
  user: "472"
  volumes:
    - grafana_data:/var/lib/grafana:Z
    - ./grafana/provisioning:/etc/grafana/provisioning:Z
```

```
  environment:
    - GF_SECURITY_ADMIN_USER=admin
    - GF_SECURITY_ADMIN_PASSWORD=<VOTRE_MOT_DE_PASSE>

    # URL publique Grafana
    - GF_SERVER_ROOT_URL=https://grafana.sanjy.fr
    - GF_SERVER_DOMAIN=grafana.sanjy.fr
```

Ajouter un enregistrement A dans OVH :
- **Type** : A
- **Sous-domaine** : grafana
- **Cible** : IP de mon VPS

Modification du podman-compose.yaml (Grafana)

Ajouter Grafana au réseau Keycloak et configurer OAuth :

```yaml
grafana:
  image: docker.io/grafana/grafana:latest
  container_name: grafana
  restart: unless-stopped
    # URLs internes (communication Grafana → Keycloak via réseau Podman)
    - GF_AUTH_GENERIC_OAUTH_TOKEN_URL=http://keycloak:8080/realms/sanjy.fr/protocol/openid-connect/token
    - GF_AUTH_GENERIC_OAUTH_API_URL=http://keycloak:8080/realms/sanjy.fr/protocol/openid-connect/userinfo

    # Configuration OAuth Keycloak
    - GF_AUTH_GENERIC_OAUTH_ENABLED=true
    - GF_AUTH_GENERIC_OAUTH_NAME=Keycloak
    - GF_AUTH_GENERIC_OAUTH_ALLOW_SIGN_UP=true
    - GF_AUTH_GENERIC_OAUTH_CLIENT_ID=grafana
    - GF_AUTH_GENERIC_OAUTH_CLIENT_SECRET=<VOTRE_CLIENT_SECRET>
    - GF_AUTH_GENERIC_OAUTH_SCOPES=openid email profile

    # URL publique (utilisée par le navigateur)
    - GF_AUTH_GENERIC_OAUTH_AUTH_URL=https://sso.sanjy.fr/realms/sanjy.fr/protocol/openid-connect/auth
    - GF_AUTH_SIGNOUT_REDIRECT_URL=https://sso.sanjy.fr/realms/sanjy.fr/protocol/openid-
```

```
connect/logout?redirect_uri=https://grafana.sanjy.fr

    - GF_AUTH_GENERIC_OAUTH_ROLE_ATTRIBUTE_PATH=contains(groups[*], 'admin') && 'Admin' ||
contains(groups[*], 'editor') && 'Editor' || 'Viewer'
  ports:
    - "3000:3000"
  networks:
    - monitoring
    - keycloak_sso  # Ajouter le réseau Keycloak
  depends_on:
    - prometheus

networks:
  keycloak_sso:
    external: true
```

Points importants :

- Les URLs `TOKEN_URL` et `API_URL` utilisent `http://keycloak:8080` (communication interne entre conteneurs)
- L'URL `AUTH_URL` utilise `https://sso.sanjy.fr` (utilisée par le navigateur de l'utilisateur)
- Remplacer `<VOTRE_CLIENT_SECRET>` par le secret récupéré dans Keycloak

Configuration Nginx pour Grafana

Créer `/etc/nginx/conf.d/grafana.conf` :

```
# Redirection HTTP → HTTPS
server {
    listen 80;
    listen [::]:80;
    server_name grafana.sanjy.fr;
    return 301 https://$server_name$request_uri;
}

# HTTPS
server {
    listen 443 ssl http2;
    listen [::]:443 ssl http2;
    server_name grafana.sanjy.fr;
```

```
    # Certificats SSL
    ssl_certificate /etc/letsencrypt/live/sanjy.fr/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/sanjy.fr/privkey.pem;

    # Paramètres SSL
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers HIGH:!aNULL:!MD5;
    ssl_prefer_server_ciphers on;
    add_header Strict-Transport-Security "max-age=31536000" always;

    # Reverse proxy vers Grafana
    location / {
        proxy_pass http://localhost:3000;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto https;
        proxy_set_header X-Forwarded-Host $host;
        proxy_set_header X-Forwarded-Port 443;

        # NE PAS mettre proxy_redirect off; (bloque les redirections OAuth)

        proxy_buffering off;
        proxy_request_buffering off;
        proxy_http_version 1.1;

        # WebSocket support
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";

        # Timeouts
        proxy_connect_timeout 90;
        proxy_send_timeout 90;
        proxy_read_timeout 90;
    }
}
```

Important : Ne pas ajouter `proxy_redirect off;` car cela empêche Nginx de réécrire correctement les redirections OAuth.

Redémarrage des services

```
# Recharger Nginx
sudo nginx -t
sudo systemctl reload nginx

# Redémarrer Grafana
cd ~/grafana
podman-compose down
podman-compose up -d
```

Test de l'authentification

1. Ouvrir `https://grafana.sanjy.fr` en navigation privée
2. Cliquer sur **"Sign in with Keycloak"**
3. Se connecter avec l'utilisateur créé dans Keycloak
4. Être redirigé vers Grafana, connecté automatiquement

? Prochaines étapes

1. Activer la 2FA avec Aegis

1. Dans Keycloak, aller dans **Authentication** → **Required actions**
2. Activer **"Configure OTP"**
3. L'utilisateur devra scanner un QR code avec Aegis (ou autre app TOTP) à la prochaine connexion

2. Intégrer Jenkins

1. Créer un nouveau client OIDC dans Keycloak : `jenkins`
2. Installer le plugin "OpenID Connect Authentication" dans Jenkins
3. Configurer Jenkins avec les mêmes principes que Grafana

3. Intégrer Vault

1. Créer un nouveau client OIDC dans Keycloak : `vault`
2. Activer l'auth method OIDC dans Vault :

```
vault auth enable oidc
vault write auth/oidc/config \
    oidc_discovery_url="https://sso.sanjy.fr/realms/sanjy.fr" \
    oidc_client_id="vault" \
    oidc_client_secret="<SECRET>" \
    default_role="default"
```

4. Gérer les rôles et permissions

1. Dans Keycloak, créer des **Groups** : `admin`, `editor`, `viewer`
2. Assigner les utilisateurs aux groupes
3. Dans chaque client, mapper les groupes vers les rôles de l'application
4. Grafana utilisera automatiquement `GF_AUTH_GENERIC_OAUTH_ROLE_ATTRIBUTE_PATH` pour mapper les rôles

---

? Ressources

- Documentation Keycloak - Reverse Proxy
- Documentation Grafana - Generic OAuth
- OpenID Connect Protocol

Troubleshoot

? Problèmes rencontrés et solutions

1. Erreur "invalid_redirect_uri" avec port 30180

Symptôme : Keycloak génère des URLs avec `http://sso.sanjy.fr:30180` au lieu de `https://sso.sanjy.fr`

Cause : Keycloak ne prend pas en compte les headers du reverse proxy

Solution :

- Ajouter `KC_PROXY_HEADERS: xforwarded` dans la config Keycloak
- Ajouter `KC_HOSTNAME_URL: https://sso.sanjy.fr` pour forcer l'URL frontend
- S'assurer que Nginx envoie les headers `X-Forwarded-Proto: https` et `X-Forwarded-Port: 443`

2. Redirect URI en `localhost:3000`

Symptôme : L'URL de redirection OAuth contient `http://localhost:3000` au lieu de `https://grafana.sanjy.fr`

Cause : Grafana ne connaît pas son URL publique

Solutions :

- Ajouter `GF_SERVER_ROOT_URL=https://grafana.sanjy.fr` dans la config Grafana
- Ajouter `GF_SERVER_DOMAIN=grafana.sanjy.fr`
- **Retirer** `proxy_redirect off;` dans la config Nginx (empêche les réécritures)
- Redémarrer complètement Grafana avec `podman-compose down` puis `up`

3. Erreur "Page not found" sur le realm

Symptôme : Erreur 404 lors de l'accès aux endpoints Keycloak

Cause : Le nom du realm dans la config ne correspond pas au nom réel dans Keycloak

Solution :

- Vérifier le nom exact du realm dans l'interface Keycloak (en haut à gauche)
- S'assurer que ce nom est utilisé partout dans les configs Grafana
- Tester avec : `curl https://sso.sanjy.fr/realms/VOTRE_REALM/.well-known/openid-configuration`

4. Erreur "connection refused" sur TOKEN_URL

Symptôme : Grafana ne peut pas échanger le code OAuth contre un token

Cause : Grafana essaie d'accéder à `https://sso.sanjy.fr` depuis le conteneur mais ne peut pas atteindre l'IP publique

Solution :

- Créer un réseau Podman partagé entre Keycloak et Grafana
- Utiliser `http://keycloak:8080` pour `TOKEN_URL` et `API_URL` (communication inter-conteneurs)
- Garder `https://sso.sanjy.fr` uniquement pour `AUTH_URL` (utilisé par le navigateur)

Vérification de la communication :

```
# Tester depuis le conteneur Grafana
podman exec grafana curl http://keycloak:8080/realms/sanjy.fr/.well-known/openid-configuration
```

5. Données Keycloak non persistées

Symptôme : Le realm et les utilisateurs disparaissent après `podman-compose down/up`

Cause : Volume non persistant ou supprimé avec `podman-compose down -v`

Solution :

- Utiliser un volume nommé dans `podman-compose.yaml` :

```
volumes:
  - data:/opt/keycloak/data


volumes:
  data:
```

- **Ne jamais utiliser** `podman-compose down -v` (le `-v` supprime les volumes)
- Vérifier que le volume existe : `podman volume ls`

6. Permissions refusées sur bind mount

Symptôme : Erreur Java "Error while creating file" dans les logs Keycloak

Cause : User namespace mapping de Podman en rootless (UID sur l'hôte ≠ UID dans le conteneur)

Solution abandonnée : Bind mount avec `./data:/opt/keycloak/data:Z`

Solution retenue : Utiliser un volume nommé Podman qui gère automatiquement les permissions :

```
volumes:
  - data:/opt/keycloak/data
```

7. Préfixe réseau `keycloak_sso` au lieu de `sso`

Symptôme : Le réseau créé s'appelle `keycloak_sso` au lieu de `sso`

Cause : `podman-compose` préfixe automatiquement avec le nom du répertoire du projet

Solution : Forcer le nom exact dans la définition du réseau :

```
networks:
  sso:
    name: sso
    driver: bridge
```

? Vérifications finales

Vérifier Keycloak

```
# Endpoint OIDC du realm
curl https://sso.sanjy.fr/realms/sanjy.fr/.well-known/openid-configuration

# Doit retourner du JSON avec issuer, authorization_endpoint, etc.
```

Vérifier le réseau Podman

```
# Lister les réseaux
podman network ls | grep sso

# Inspecter le réseau
podman network inspect keycloak_sso
```

Vérifier la communication inter-conteneurs

```
# Depuis Grafana vers Keycloak
podman exec grafana curl http://keycloak:8080/realms/sanjy.fr/.well-known/openid-configuration

# Doit retourner du JSON (pas d'erreur de connexion)
```

Vérifier le volume Keycloak

```
# Lister les volumes
podman volume ls

# Inspecter le volume
podman volume inspect keycloak_data  # ou le nom de votre volume
```

Vérifier les logs

```
# Logs Keycloak
podman logs keycloak

# Logs Grafana (chercher "oauth", "keycloak")
podman logs grafana | grep -i oauth
```

? Points clés à retenir

URLs publiques vs URLs internes

- **AUTH_URL** : HTTPS public (`https://sso.sanjy.fr`) - utilisé par le navigateur
- **TOKEN_URL** et **API_URL** : HTTP interne (`http://keycloak:8080`) - communication serveur à serveur

Réseau Podman

- Les conteneurs sur le même réseau peuvent communiquer via leurs noms
- Utiliser `keycloak:8080` depuis Grafana au lieu de `localhost:30180`

Volumes Podman

- Les volumes nommés sont plus fiables que les bind mounts en rootless
- Ne jamais utiliser `podman-compose down -v` pour éviter de supprimer les données
- Backup : `podman volume export keycloak_data -o backup.tar`

Headers Nginx essentiels

```
proxy_set_header X-Forwarded-Proto https;
proxy_set_header X-Forwarded-Host $host;
proxy_set_header X-Forwarded-Port 443;
```

Ces headers sont essentiels pour que Keycloak génère les bonnes URLs.

Ne JAMAIS utiliser `proxy_redirect off;`

Cette directive empêche Nginx de réécrire les redirections, ce qui casse OAuth/OIDC.
