Proxy d'application : générer et utiliser un certificat pour les applications publiées avec Azure Key Vault

Présentation

- Certificat de signature SAML
- Géré dans Azure Key Vault
- Attaché à une App Registration
- Renouvellement automatique via GitHub Actions
- Zéro interruption (chevauchement des certificats)

Architecture globale dans Entra ID:

```
App Registration
 ├─ Certificats OAuth
 ├─ GitHub Actions (OIDC)
 ├─ Key Vault
 └─ Microsoft Graph


GitHub Actions
   │
   ▼ (OIDC, sans secret)
Azure Entra ID
   │
   ▼
Azure Key Vault  ←——— Génération / rotation du certificat
   │
   ▼
App Registration (SAML)
```

- Pas de secrets dans GitHub
- Authentification par OIDC Federated Credentials
- Conformité Microsoft / Zero Trust

Cas standard Microsoft OAuth :

- Protocole : OAuth 2.0 / OpenID Connect
- Méthode d’authentification : client_secret_post ou client_secret_basic
- Objet Entra ID utilisé : Inscription d’application
- SAML : NON ⇒ Application d’entreprise NON requise

Élément	Gérer SAML	Gérer certificats / secrets & automation
Inscription d'applications	NON	OUI
Applications d'entreprise	OUI	NON

Prérequis Entra ID

- Une App Registration existante
  - Type : Single tenant
  - Usage : SAML
- Un Key Vault
  - SKU Standard
  - Soft delete activé (par défaut)

Créer un Key Vaults (coffres de clé)

- créer un Key Vaults :
  - choisir l'abonnement
  - créer un groupe de ressources
  - définir le nom du coffre de clé
  - choisir la région et le niveau tarifaire standard

Lien d'information sur la tarification : https://azure.microsoft.com/fr-fr/pricing/details/key-vault/

- Accéder au coffre de clé :
  - dans le contrôle d'accès (IAM), donner le rôle **Agent des certificats Key Vault** plutôt que **Administrateur Key Vault** à l'utilisateur qui va créer et gérer les certificats
  - l'Onglet **Attributions de rôles** permet de vérifier les rôles affectés

Création de l’App Registration dans le Portail Azure

- Accéder au **portail Azure** puis **Microsoft Entra ID**.
- Choisir **Inscription d'applications**.
- Puis **Nouvelle inscription** :
  - Nom : signature-Valadon,
  - Type de compte : Locataire uniquement,
  - URI de redirection Web : https://your-domain/api/auth/callback/microsoft
  - Cliquer sur **S'inscrire**.

Noter :

- ID client
- ID du Tenant

Définir les autorisations d'API suivantes :

- openid
- profile
- email
- User.Read

Créer le secret client

- dans **Certificats et secrets**
- Choisir **Nouveau secret client**
- Description : **app-secret**
- Date d'expiration : 24 mois
- Puis **Ajouter**

Noter le secret client

Vérification

- Dans Authentification : l'URI de redirection Web doit être https://your-domain/api/auth/callback/microsoft
- Dans Authentification puis l'onglet **Paramètres** :
  - ne pas cocher **Jetons d’accès (utilisés pour les flux implicites)** ni. **Jetons d’ID (utilisés pour les flux implicites et hybrides)**
  - ne pas activer les flux clients publics
  - ne pas configurer de verrou de propriété d'instance d'application

Créer un certificat

- accéder au coffre de clé
- dans la rubrique **Objets**, choisir **Certificats**
- Cliquer sur **+ Générer / Importer**
- Définir les paramètres suivants :
  - Method : Generate
  - Name : documenso-cert
  - Type : Self-signed
  - Subject : CN=sign.educ-valadon-limoges.fr
  - Validité : 24 mois
  - Type de contenu : PCKS#12
  - Type d'actions de la durée de vie : Renouveller en fonction d'un pourcentage de la durée de vie
  - pourcentage de la durée de vie : 80%

Cliquer sur Créer

jouter le rôle : Agent des secrets Key Vault

Export du certificat

Une fois créé :

- cliquer sur le certificat
- puis télécharger :
  - .PFX (nécessaire pour proxy / serveur)
  - .CER (public uniquement)

Automatisation GitHub (rotation du secret)

- App Registration ⇒ Certificates & secrets ⇒ Informations d'identité fédérées
- Choisir **Ajouter un justificatif** avec les paramètres suivants :
  - Scenario : Actions GitHub déployant des ressources Azure
  - epo : ORG/REPO

Branch : main