VIF software

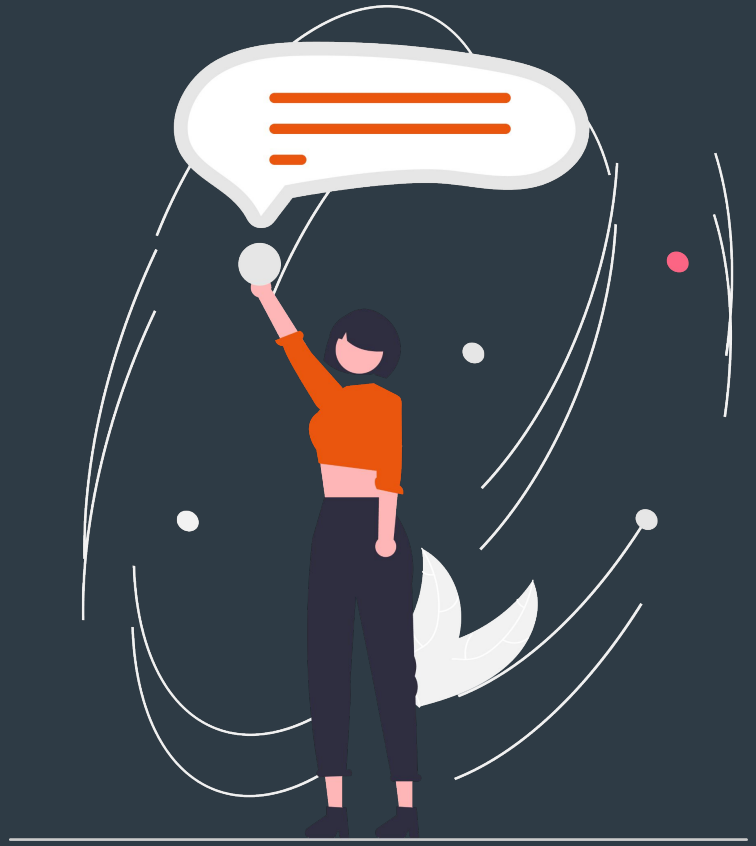
INNOVONS ENSEMBLE

Améliorer la sécurité et l'authentification des flux de VIF ERP

PUG 2025

Philippe LANDAIS - Quentin LE ROUX

Bonjour !



Speakers

- Philippe LANDAIS
  - Lead Tech
  - Progress, SQL

- Quentin LE ROUX
  - Lead Tech
  - Java, CI/CD, environnements de développement, conteneurs, JBoss/Wildfly

Sommaire

1. Historique de l’ERP VIF
2. Objectifs Cyber V1 ERP
3. Architecture ERP
4. Exemple de sécurisation de Flux ERP
5. Bilan & Questions

1. Historique de l’ERP VIF

1. Historique du produit VIF ERP

- 1981 - ERP v1 (VIF1) - COBOL
- 198X - ERP v2 (VIF2) - Unix Progress 6
- 1990 - ERP v3 (VIF3) - Progress “terminal” (tty)
- 1996 - ERP (VIF5_7) - Progress graphique 8
- 2004 - ERP (VIF5_7) - Update Progress 9
- 2008 - ERP (VIF5_7) - Update Progress OE10.2B
- 2008 - ERP (VIF5_7) - Ajout client lourd Java (orchestrateur) et serveur d’application Jboss
- 2017 - ERP (VIF5_7) - Migration JBoss -> Wildfly
- 2013 - ERP (VIF5_7) - Update Progress OE11.7
- 2024 - ERP (VIF5_7) - Update Progress OE12.8 (ajout du PASOE)
- 2025 - ERP (VIF5_7) - Cyber V1

Etat des lieux VIF ERP en 2025

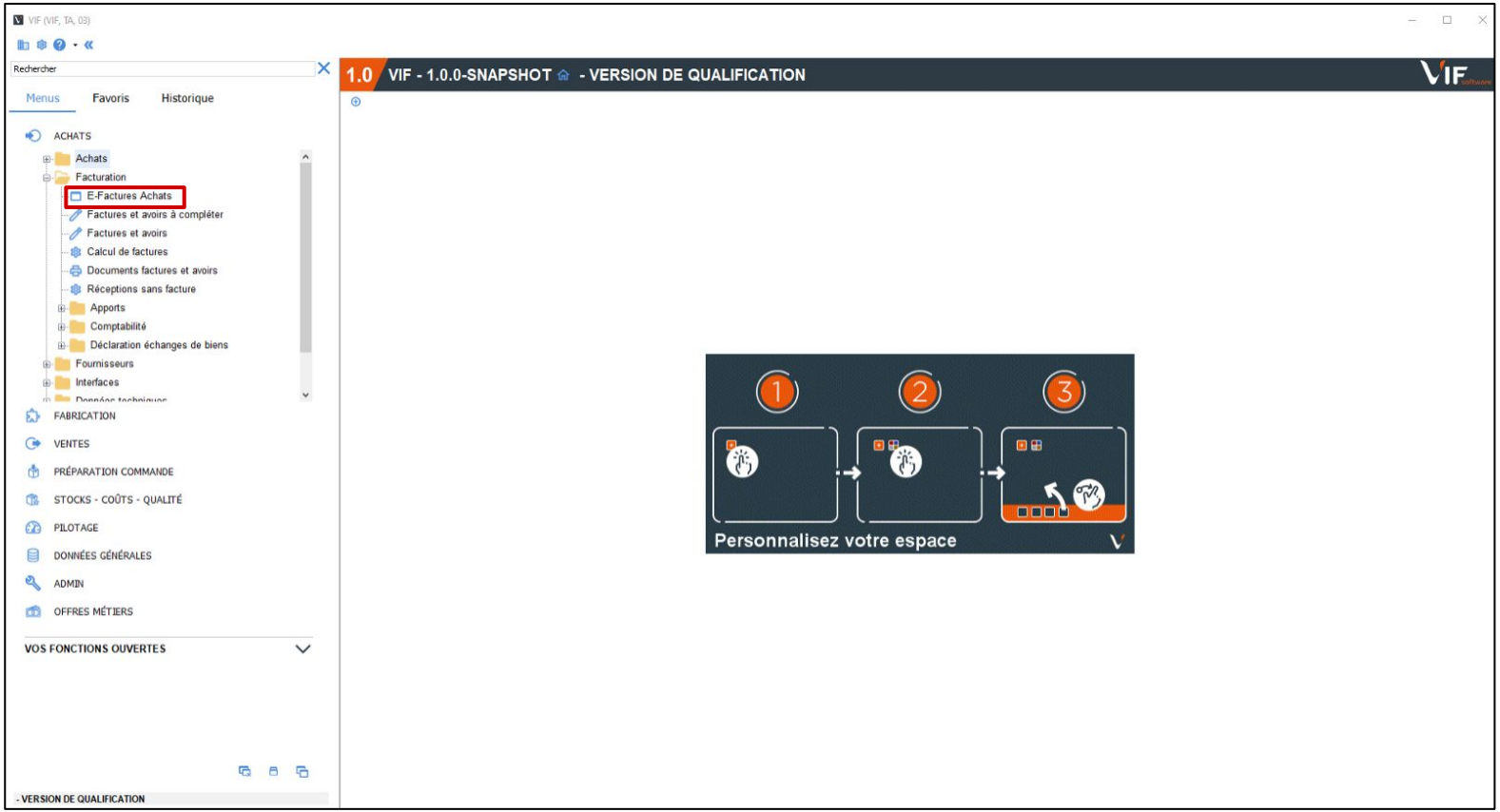
- 5 millions de lignes de code Progress
- 2 millions de lignes de code Java
- Stack technique récente:
  - Progress 12.8.6 (Base, PASOE, ABL)
  - Java 17
  - Wildfly 26.1.3.Final/Jakarta 8
  - RHEL 9

Annexe 1a: visuels client ERP administratif front

Orchestrateur Java

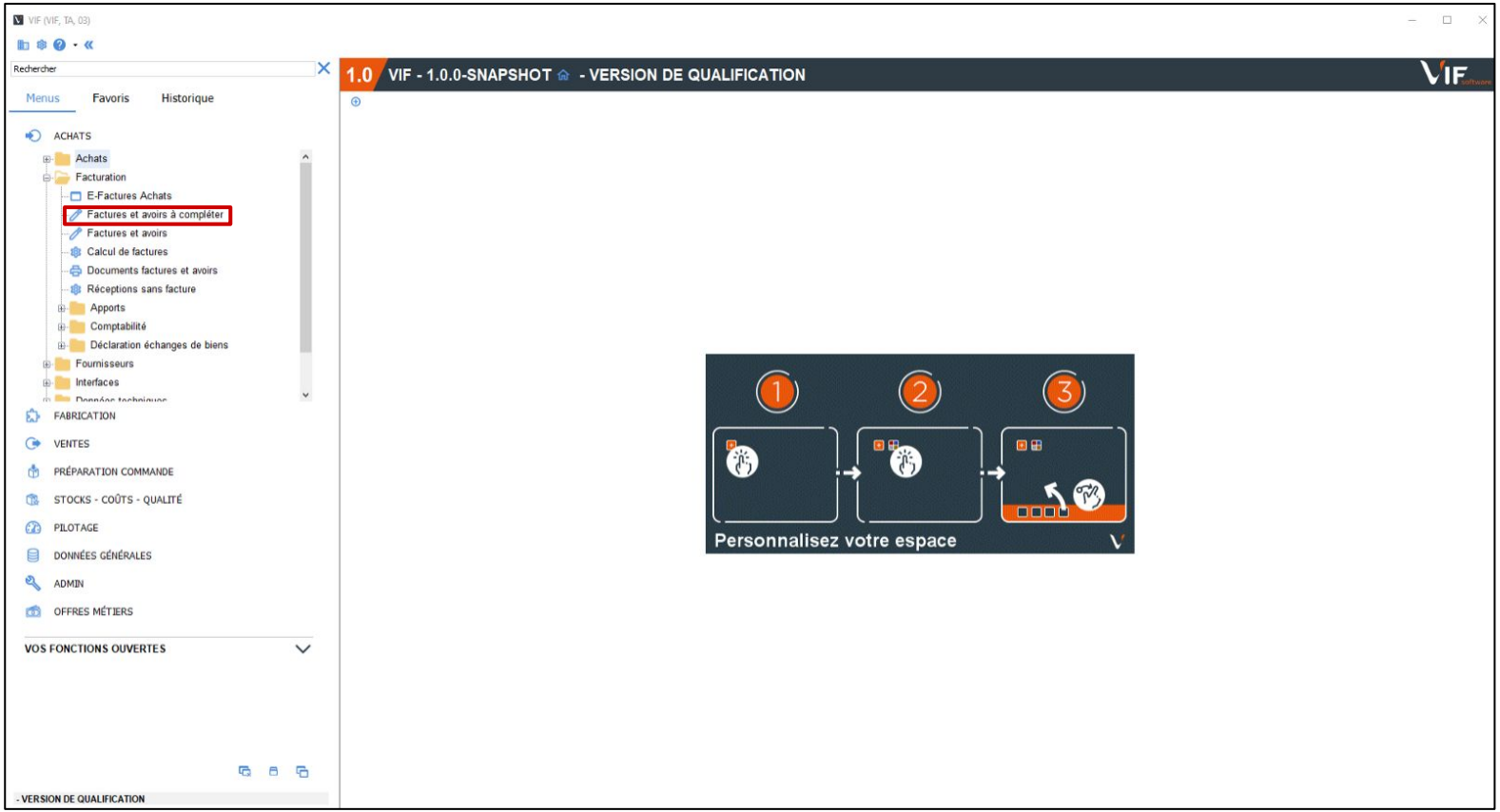
Annexe 1b: visuels client ERP administratif front

Fonction Java



Annexe 1c: visuels client ERP administratif front

Fonction Progress



2. Objectifs Cyber V1 ERP

2. Objectifs Cyber VIF ERP V1

- OpenId Connect Compatible
- Authentification des flux
- Chiffrement des flux
- Centralisation des logs
- Segmentation reseau
- Hardening
- ...

3. Architecture ERP

3. Architecture ERP

Complexité accrue par:

⚠️ Mix de technologies: Java, Progress

⚠️ Mix d'architecture: Client -> App Server -> DB, vs Client -> DB

⚠️ Nombre de clients: lourds, terminaux android, autres applications, outils, interface d'administration

⚠️ Mix de transports: HTTP, Remoting, Bus de message

4. Exemple de sécurisation de flux ERP

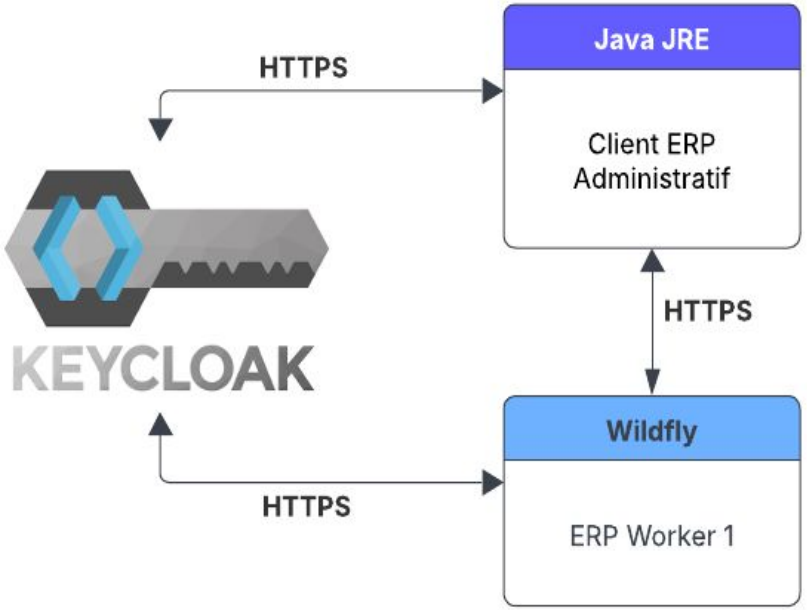
4. Exemple 1: Authentification du flux: Client Java -> Wildfly avec OpenId Connect (OAuth 2)

Objectifs:

- Authentifier le flux entre le client Java et le serveur d'application Wildfly avec OpenId Connect.

Solution:

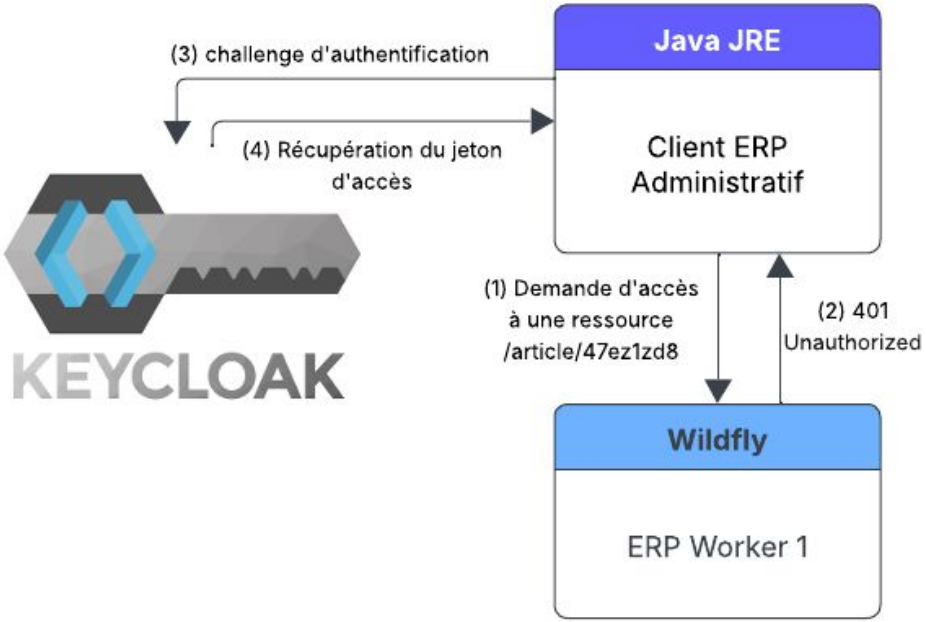
- Utilisation d'un IAM (Keycloak).
- Développement Java back & front pour compatibilité OpenId Connect.
- Configuration Wildfly (Elytron) pour validation du JWT (Json Web Token)..



4. Exemple 1: Authentification du flux Client Java -> Wildfly avec OpenId Connect (OAuth 2)

Objectifs:

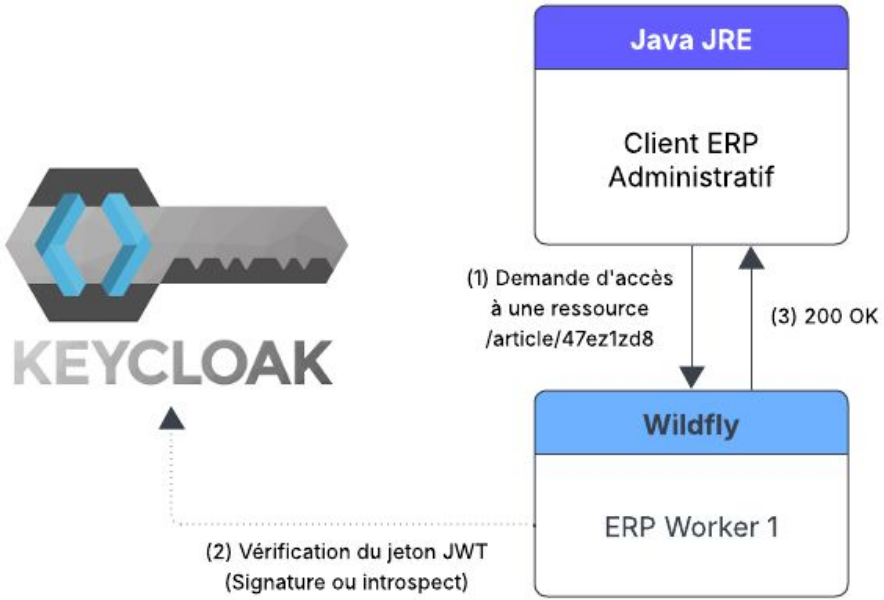
- Authentifier le flux entre le client Java ERP et le serveur d'application Wildfly avec OpenId Connect.



4. Exemple 1: Authentification du flux Client Java -> Wildfly avec OpenId Connect (OAuth 2)

Objectifs:

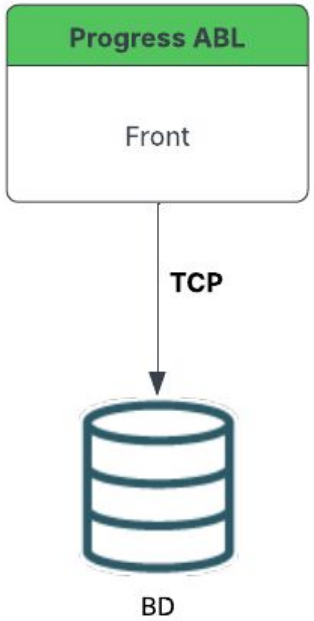
- Authentifier le flux entre le client Java ERP et le serveur d'application Wildfly avec OpenId Connect.



4. Exemple 2: Authentification du flux Client Progress (ABL) -> Base de données

Objectifs:

- Authentifier le flux entre le client Progress (ABL) et la base de données (OAuth2 ? Comptes nominatifs ? autres ?)



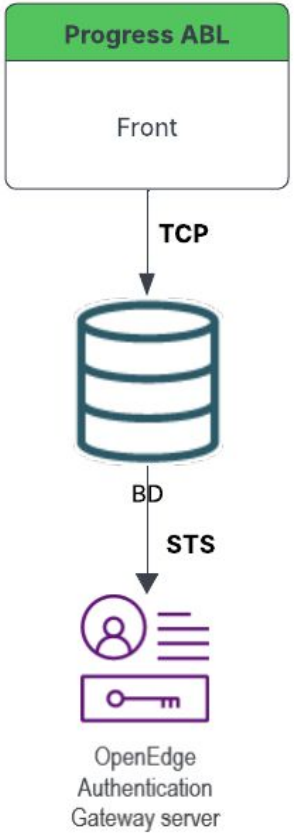
4. Exemple 2: Authentification du flux Client Progress (ABL) -> Base de données

Objectifs:

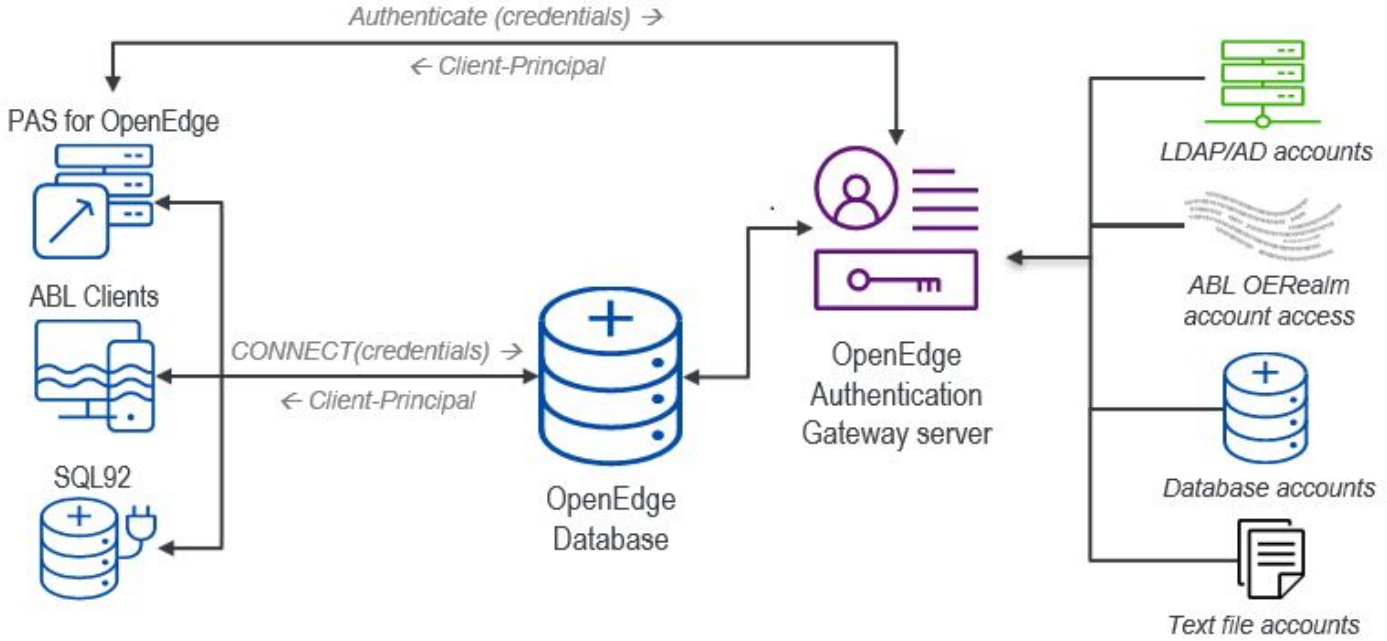
- Authentifier le flux entre le client Progress (ABL) et la base de données (OAuth2 ? Comptes nominatifs ? autres ?)

Solution 1: Utiliser le nouveau composant Progress OE Authentication Gateway.

+ permet d'unifier/centraliser et de déléguer l'authentification des composants Progress OE (PASOE, Base de données) à Authentication Gateway (voir Annexe 2).
- (native ABL Client ne supporte pas le protocole OAuth2)
- une brique supplémentaire à exploiter/maintenir
- une VM à provisionner supplémentaire (OE Gateway doit être sur une VM différente, pas d'accès en localhost/IP).
- non adapté pour le cas ERP VIF.



Annexe 2: Progress OE Authentication Gateway



Each client connection to the OpenEdge Database is authenticated by the OpenEdge Authentication Gateway, which can read user information from a user account source or directory information service such as LDAP, Active Directory, OERealm, database, or text file accounts. PAS for OpenEdge can delegate user-ID/password authentication to the OpenEdge Authentication Gateway.

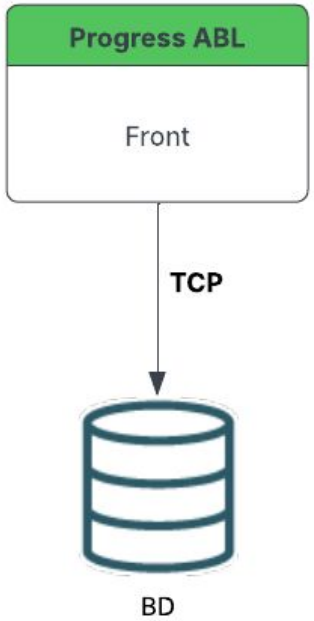
4. Exemple 2: Authentification du flux Client Progress (ABL) -> Base de données

Objectifs:

- Authentifier le flux entre le client Progress (ABL) et la base de données (OAuth2 ? Comptes nominatifs ? autres ?)

Solution 2: Définir une liste de **compte de services** dans la base de données (_User).

\+ simple à mettre en oeuvre
\- pas de comptes nominatifs



4. Exemple 3: Chiffrement du flux Wildfly -> PASOE

Objectifs:

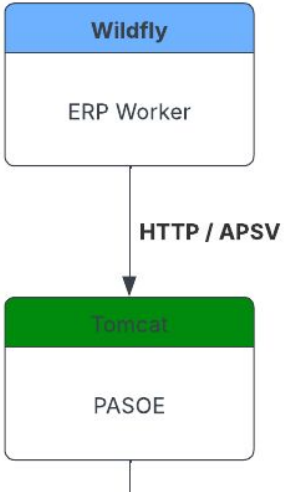
- Chiffrement du flux HTTP entre le Wildfly et le PASOE.

Pré-requis:

- Certificat serveur (type wildcard *.vif.fr) et sa clé privée
- Un ou plusieurs certificats intermediaries
- Certificat racine (Root CA)

Principe:

- Le serveur présente son certificat (avec ses certificats intermédiaires).
- Le client reçoit le certificat du serveur et valide la chaîne de confiance jusqu'au certificat racine.



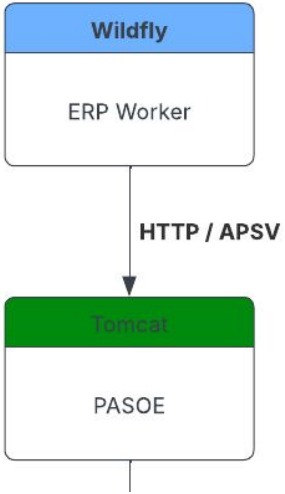
4. Exemple 3: Chiffrement du flux Wildfly -> PASOE

Objectifs:

- Chiffrement du flux HTTP entre le Wildfly et le PASOE.

Sur PASOE:

- Tomcat -> relativement standard pour ceux qui connaissent java (et documentation progress):
  - création du keystore .p12 avec keytool, protégé par une passphrase
  - import du certificat du serveur
  - configurer le **catalina.properties** du PASOE:
    - # JSSE keystore used by server.xml for its server key & certificates
    - psc.as.https.keypass=mypass
    - psc.as.https.keyalias=pasoe
    - psc.as.https.storeType=PKCS12



4. Exemple 3: Chiffrement du flux Wildfly -> PASOE

Objectifs:

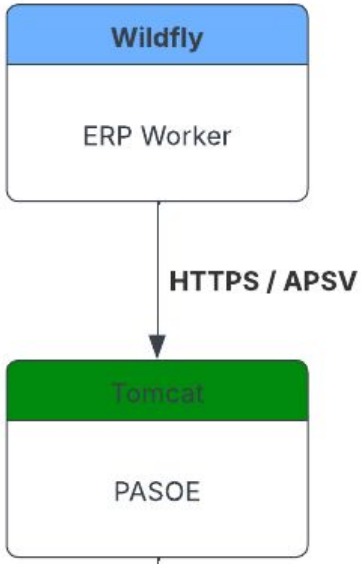
- Chiffrement du flux HTTP entre le Wildfly et le PASOE.

Sur Wildfly:

- Mise a jour du Java Open client progress: **o4glrt.jar** (version 12.8.6). Attention aux dépendances et potentiels conflits (surtout si monolithe).

- Définir le Truststore du java open client avec
  `RunTimeProperties.setCertificateStore(progressTruststorePath);`

  Attention: le format du truststore est propre à Progress (par défaut **psccerts.zip** ou **psccerts.jar**.
  Celui ci est généré depuis une installation de Progress avec import des ROOT CA (**certutil / procert)**. On les trouve dans **/dlc12/certs/.**



4. Exemple 4: Authentification du flux Wildfly -> PASOE

Objectifs:

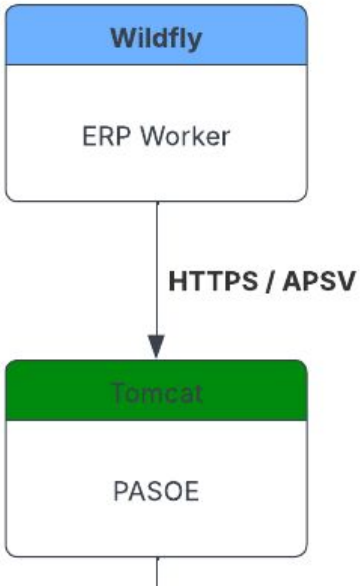
- Authentification du flux entre le Wildfly et le PASOE.

Sur PASOE:

- Ajout de l'authentification **BASIC** uniquement sur le transport **APSV** (le seul utilisé).
  - Fichier **oeablSecurity.properties**
    - **apsv.security.enable=basic**
    - **http.all.authmanager=extlocal**

    extlocal: Déclaration des utilisateurs (comptes de services) dans **users.properties**, les passwords sont encryptés.

    Le hash des passwords s'effectue avec l'utilitaire Progress: **genspringpwd** qui utilise l'algorithme **bcrypt** (lent, protection contre brut force attack, ...).



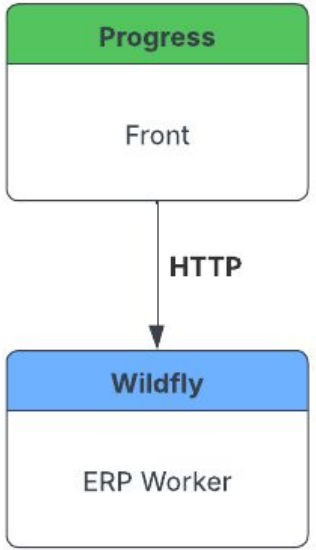
4. Exemple 5: Authentification des Web Services, Client Progress -> Wildfly

Objectifs :

- Authentification des appels de Web services à partir de Progress

Contexte :

- Pas d'authentification
- 2 cas : écrans et traitements déportés
- 2 protocoles utilisés : SOAP + REST
- Temps limité car lié à une version (vie courante à gérer)



4. Exemple 5: Authentification des Web Services, Client Progress -> Wildfly

Cas des WS REST

Contexte :

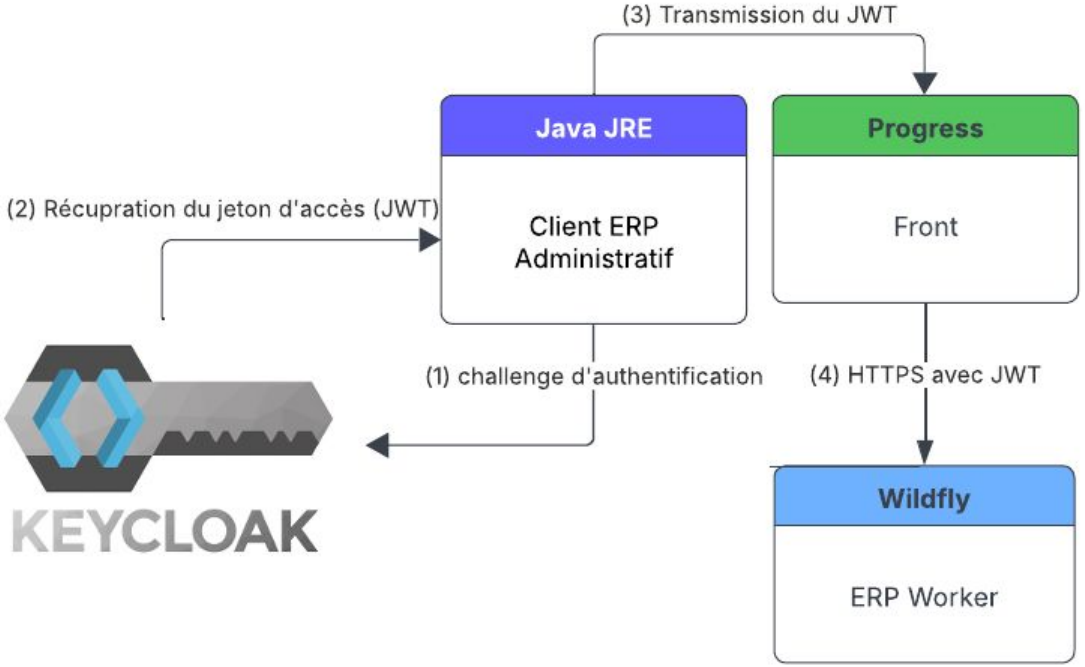
- Librairie centralisée
- Utilisation de socket

Réalisation :

- Réécriture de la partie communication socke pour utiliser les classes objets fournies par Progress (HTTPRequest)
- Transmission du token

Remarques :

- Propath à modifier (classes Progress)
- Gérer la transmission du token jusqu'aux fonctions Progress



4. Exemple 5: Authentification des Web services, Client Progress -> Wildfly

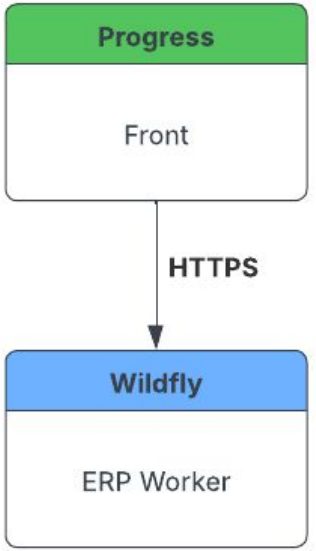
Cas des WS SOAP

Contexte:

- Une trentaine de services SOAP
- Librairie centralisée pour la connexion
- Partie description du WSDL éclatée (1 programme par service)

Réalisation:

- Problème rencontré pour passer l'authentification sur l'entête http
- Réécriture des WS pour les passer en REST
  - utilisation des instructions de conversion de temp-table vers json (vice versa)
  - objet de parsing



4. Exemple 5: Authentification des Web services, Client Progress -> Wildfly

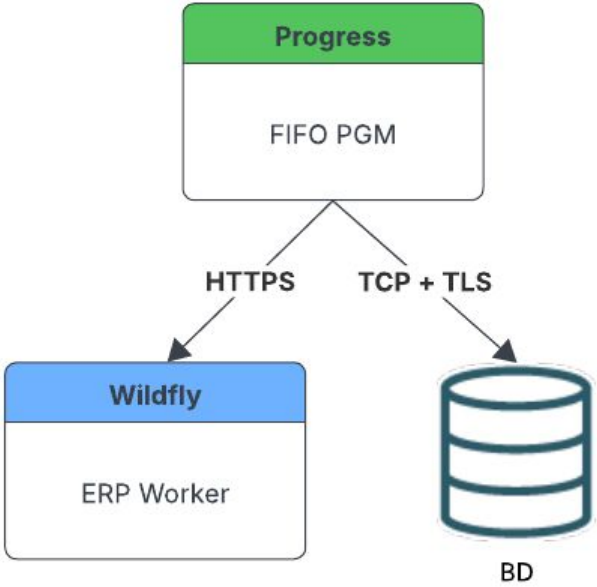
Cas des programmes déportés

Contexte:

- Traitements exécutés hors identification suite à action utilisateurs ou traitement planifié
- Pas de connexion à un IAM, pas de token

Réalisation:

- Utilisation d'un compte de service, pour se connecter à la base et échanger avec le serveur Wildfly



4. Exemple 5: Authentification des Web services, Client Progress -> Wildfly

Bilan :

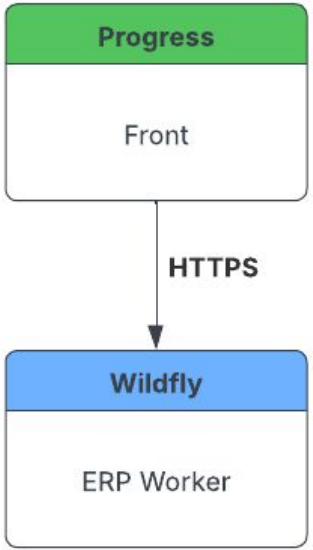
- Authentification des WS
- Réécriture des WS SOAP en REST (plus modernes)
- Utilisation des classes Progress

Attention :

- Propath à modifier suite utilisation classe Progress
- Attention aux temps de réponse (lenteur constatée sur le 1er appel)
- Appbuilder pas forcément idéal pour développer en objet

Futur :

- Reprendre le sujet VSCode
- Revoir notre analyse des sources



5. Bilan

5. Bilan

Respect des enjeux ERP Cyber V1:

- La quasi-totalité des flux externes et internes ERP sont chiffrés et authentifiés.
- Compatibilité avec les principaux IAM du marché (Keycloak, OKTA, Azure Entra ID)

Néanmoins:

- Authentification encore hétérogène (OAuth2 et comptes de services)
- Nouvelles procédures d'exploitation (configuration IAM client, rotation des certificats, rotation des comptes).
- Impact de performances à mesurer (surtout le TLS sur la base de données).

Merci de votre attention !

N’hésitez pas à poser vos questions