Esup-DSS-Client

Esup-DSS-Client est une nouvelle passerelle, developpée dans le cadre du projet Esup-Signature, entre le poste client et DSS Signature (sous système de signature de la commission européenne). Cette application remplace l'utilisation du client NexU qui n'est plus maintenu depuis 2018.

Esup-DSS-Client est à installer sur les postes clients des utilisateurs qui ont besoin de signer avec un certificat local ou materiel.

Le code de l'application NexU (sous license EUPL) a été partiellement repris pour coder cette nouvelle application. Les principaux changements par rapport à NexU sont :

- Compatibilté OpenJDK et OpenJFX
- Suppression des dependances non livrées avec NexU qui empéchait la compilation
- Ajout d'un plugin utilisant les capacités d'OpenSC pour se connecter aux périphériques materiel de manière native sous Linux, Windows et macOS

Pour fonctionner, le module OpenSC doit être installé sur la machine. Ce projet est disponible ici : https://github.com/OpenSC/OpenSC. Comme l'accès au support crypto est natif (pcsc, apdu) il n'est pas nécessaire d'installer un pilote sur la machine cliente.

GitHub

Le code source est disponible ici : https://github.com/EsupPortail/esup-dss-client

Les installateurs : https://github.com/EsupPortail/esup-dss-client/releases/latest

Ressource

Exemple de notice d'installation / utilisation du client de L'université de Rouen

Changelog

v1.2.1-RELEASE-08/04/2024

Versions :

- OpenSC 0.25.1

Améliorations :

- Transmission des messages erreurs

v1.2-RELEASE-29/03/2024

Versions :

- OpenSC 0.25.0

v1.1-RELEASE-15/12/2023

Versions :

- OpenSC 0.24.0

Améliorations :

- Driver propriétaire configurable via UI
- Amélioration UI

v1.0-RELEASE-21/06/2023

Versions :

- OpenJDK 17 minimum
- Compatible Linux, Windows 64 et MacOS
- OpenSC 0.23.0

Fonctionnalités :

- Signature via OpenSC
- Signature via le magasin de clés windows
- Signature via PKCS12

Fonctionnement en lien avec Esup-Signature

Le fonctionnement d'Esup-DSS-Client (communication entre le navigateur et le client) est décit sur cette page d'archive à propos de l'application NexU : Application NexU (Archive)

Pour résumer, lorsqu' Esup-DSS-Client est démarré, un serveur web se lance sur le poste client aux adresses suivantes http://localhost:9795 et https://localhost:9895. Les vues web d'Esup-Signature qui permettent de signer, importent un fichier javascript hébergé à l'adresse http://localhost:9795. Lors de la signature le navigarteur communique avec la clé via les commandes javascript issues d'Esup-DSS-Client.

Esup-signature fait référence à l'adresse http://localhost:9875 pour eviter les problèmes de certificats. Cependant, certains navigateurs sont susceptibles de refuser d'importer le script dans la page web d'esup-signature.

De ce fait Esup-DSS-Client **ne fonctionne pas sous Safari** et il peut arriver que Firefox lève une alerte de securité. Nous n'avons jamais rencontré de problème avec Chrome quelque soit l'OS.

À l'université de Rouen la signature est concluante avec un certificat obtenu auprès de certinomis : Offre SERVEUR 2 étoiles / Cachet 2 étoiles G2 - sur carte. L'autorité de certification est reconnue par la trustlist française sous le nom "Certinomis - Prime CA G2"

Le materiel reçu est une clé Feitian Technologies, Inc. SCR301 avec une carte Gemalto pris en charge par OpenSC (pilote "idprime : Gemalto IDPrime").

Ici la liste des matériels supportés par OpenSC : https://github.com/OpenSC/OpenSC/wiki/Supported-hardware-%28smart-cards-and-USB-tokens%29

Installation :

Pour simplifier l'installation d'Esup-DSS-Client, un installateur est disponible. Celui-ci va installer Zulu (distribution comprenant OpenJdk et OpenJFX), et OpenSC.

Cependant il est possible de cloner le projet et le compiler localement. Dans ce cas il est possible d'obtenir le fichier esup-dss-client-jar-with-dependencies.jar. Pour le lancer directement il faut les pré-requis suivants :

- OpenJDK 17
- OpenJFX 11

La commande à passer est :

```
java --add-opens java.base/java.lang=ALL-UNNAMED --module-path <path-to-openjfx> --add-modules javafx.controls,javafx.fxml,javafx.base,javafx.media,javafx.graphics,javafx.swing,javafx.web --add-exports javafx.graphics/com.sun.javafx.application=ALL-UNNAMED --add-exports javafx.graphics/com.sun.javafx.tk=ALL-UNNAMED -Djdk.gtk.version=2.2 -jar esup-dss-client.jar
```

Sous Linux il peut être necéssaire d'ajouter cette librairie:

```
sudo apt-get install libnss3-tools
```

Contrôle du fonctionnement d'OpenSC

L'installation et la vérification d'OpenSc sont décrits sur cette page : OpenSC

Installation sous linux :

Un installateur "izpack" est disponible ici : https://github.com/EsupPortail/esup-dss-client/releases/latest/download/esup-dss-client-installer.jar

Installation sous macOS

La signature ne fonctionne pas sous Safari car celui-ci refuse l'intégration des scripts provenant de localhost.

Avec package

Recupérez le package https://github.com/EsupPortail/esup-dss-client/releases/latest/download/esup-dss-client.pkg

Il faut ensuite le lancer en cliquant sur le "bouton droit". Le package est bien signé avec le certificat du Consortium ESUP-Portail mais sans forcer il refusera de s'ouvrir (certainement car il contient des scripts sh)

Sans le package

Il faut installer OpenJDK 17 + openJFX (avec zulu par exemple : https://www.azul.com/downloads/?version=java-17-lts&os=macos&package=jre-fx#zulu)

Il faut aussi installer OpenSC. Un dmg est disponible ici : https://github.com/OpenSC/OpenSC/releases/download/0.24.0/OpenSC-0.24.0.dmg

Ces deux installation peuvent aussi être effectées via homebrew voir : https://brew.sh/

Les commandes à lancer seront :

```
brew install opensc
brew install zulu
```

Enfin lancer le jar https://github.com/EsupPortail/esup-dss-client/releases/latest/download/esup-dss-client.jar

```
java --add-opens java.base/java.lang=ALL-UNNAMED --module-path <path-to-openjfx> --add-modules javafx.controls,
javafx.fxml,javafx.base,javafx.media,javafx.graphics,javafx.swing,javafx.web --add-exports javafx.graphics/com.
sun.javafx.application=ALL-UNNAMED --add-exports javafx.graphics/com.sun.javafx.tk=ALL-UNNAMED -Djdk.gtk.
version=2.2 -jar esup-dss-client.jar
```

Sous Mac, le système est succeptible de proposer l'utilisation de la clé pour vérouiller la session ou l'élévation de droits (dans le cas où les pilotes sont installés).

Dans ce cas si le mot de passe est demandé par le système, c'est le code pin qu'il faut saisir. Une confusion sur trois saisies d'affilées va vérouiller le certificat !

Installation sous Windows

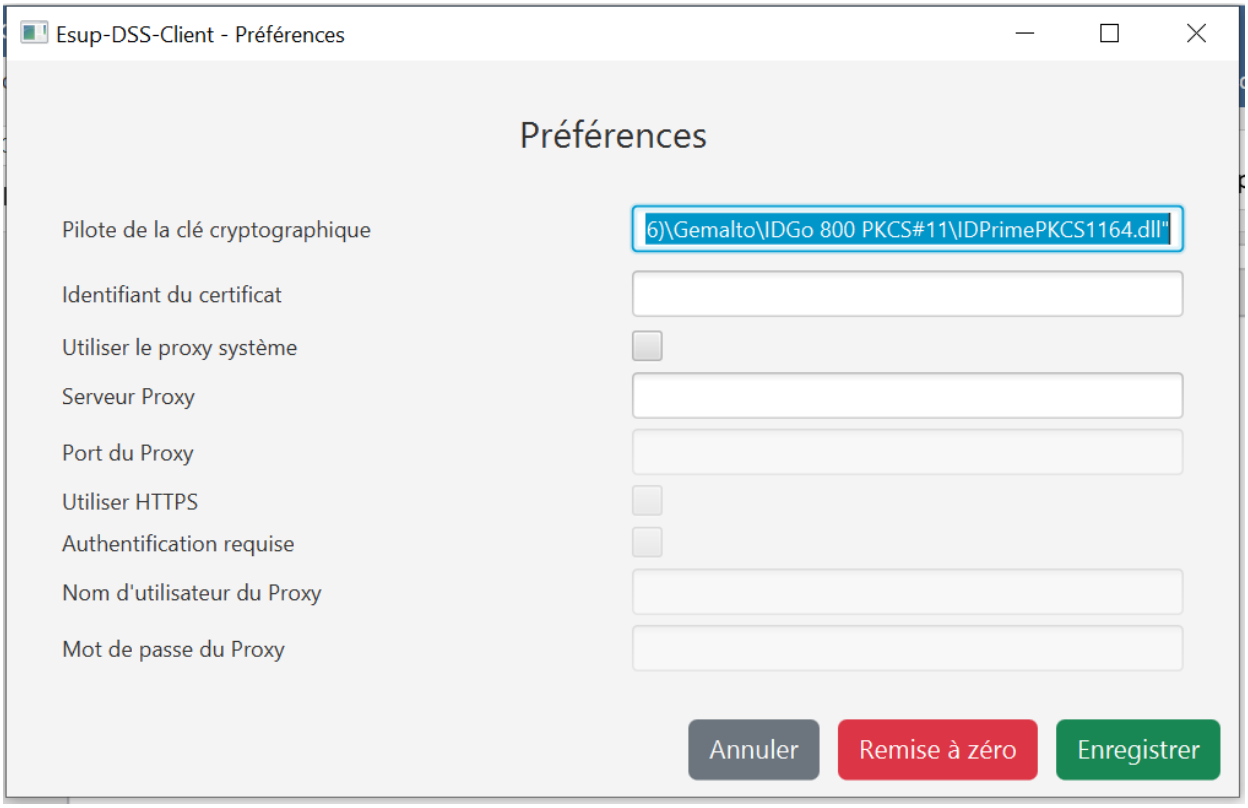
Installateur windows est disponible ici : https://github.com/EsupPortail/esup-dss-client/releases/latest/download/esup-dss-client-win64.zip

Configuration

Par défaut, losrque l'on selectionne OpenSC, Esup-DSS-Client va tenter de communiquer avec la clé cryptographique à l'aide de son implémentation "open". Or tous les supports ne sont pas forcément implémentés. Pour plus de détails sur le fontionnement d'OpenSC voir le page OpenSC

Pilote de la clé cryptographique

Depuis la version 1.1 d'Esup-DSS-Client, il est possible de spécifier un driver propriétaire via l'interface graphique en ouvrant la fenetre "Préférences" depuis l'icône de notification.



Sur la fenetre "Préférences" vous pouvez saisir le chemin vers le pilote de votre clé.

Voici des exemples testés à Rouen pour la clé Certinomis :

Os	Chemin
Linux	/usr/lib/pkcs11/libIDPrimePKCS11.so
MacOs	/usr/local/lib/libIDPrimePKCS11.dylib
Windows	"C:\Program Files (x86)\Gemalto\IDGo 800 PKCS#11\IDPrimePKCS1164.dll"

Les chemins sont à adaptés en fonction de votre sytème d'exploitation et de l'emplacement d'installation de vos pilotes.

À noter que le chemin sous windows ne fonctionne qu'avec des guillements

Identifiant du certificat

Le deuxième paramètre concerne l'identifiant du certificat. Comme précisé sur cette page, OpenSC#Driverpropri%C3%A9taire, l'identifiant change en fonction du pilote utilisé. Esup-DSS-Client doit touver automatiquement l'identifiant mais si ça n'est pas le cas, il est possible de le "forcé" via le champ "Identifiant du certificat". Pour l'obtenir on utilise le commande OpenSC :

```
pkcs11-tool --login --test --module <emplacement du driver>
```

Compilation / Obtention des installateurs

Tout d'abord il faut cloner le projet en local sur voitre machine :

```
git clone https://github.com/EsupPortail/esup-dss-client.git
```

Sous Linux

En lancant :

```
mvn clean install
```

Vous obtenez dans ./target les fichiers :

- esup-dss-client-installer.jar pour l'installation sous Linux
- esup-dss-client-win64.zip pour l'installation sous Windows

Pour obtenir le jar seul :

```
mvn clean package
```

Sous MacOS

Pour obtenir l'installateur PKG, il faut être sous macOS. Voici les pre-requis à installer :

- git (et donc les outils Xcode)
- brew (voir https://brew.sh/index_fr)
- maven via brew (brew install maven)

Modifier le code développeur dans src/izpack/pkg.sh

```
mvn clean package -Dmac.os=true
```


En cas d'erreur : Gtk-Message: 21:22:16.442: Failed to load module "canberra-gtk-module"

sudo apt-get install libcanberra-gtk-module