Authentification SSO pour le parapheur électronique

Version 4.5

Libriciel SCOP

Document

Auteur	Lukas Hameury	***Date de diffusion***	15/05/2017
Chef de projet	Stéphane Vast	***N° de version***	4.5

Évolution du document

Version	Auteur	Nature des changements	Date
1.0	Lukas Hameury	Rédaction du document	15/05/2017

Licence

Table des matières

1. CAS

1.1. Resources nécéssaires

1.1.1. Pré-requis

Certains pré-requis sont à mettre en place en amont de la configuration :

- Le serveur CAS doit avoir connaissance de l'url du serveur i-Parapheur (DNS ou hosts)
- L'AC du certificats serveur i-Parepheur doit être importé dans le keystore du serveur CAS (au niveau de la JVM)
- La v4 du i-Parapheur doit avoir été lancée au moins une fois

1.1.2. Fichiers

Les fichiers suivants sont nécessaires pour la configuration de l'authentification SSO-CAS :

- `cas-alfresco.amp`
- `cas.properties`
- un keystore contenant la clé publique du certificat SSL du serveur CAS
- le package `conf.tgz` contenant les fichiers de configuration modifiés ainsi que les scripts de modifications de fichiers.

1.1.3. Informations

Les informations suivantes sont aussi nécessaires pour configurer le parapheur :

- l'adresse FQDN du i-Parapheur
- l'adresse du serveur CAS (adresse complète de la page de login)

1.2. Étapes de mise en place

On commence par extraire la configuration dans le dossier `/opt/_install/cas/` :

```
mkdir -p /opt/_install/cas && cd /opt/_install/cas
wget http://para…/cas-install.tgz
tar zxvf cas-install.tgz
cd /opt/iParapheur
```

1.2.1. Déploiement de l'AMP

1. Arrêter le service i-Parapheur

   ```
   /etc/init.d/alfresco stop
   ```

2. Copier `cas-alfresco.amp` dans le dossier amp

   ```
   cp /opt/_install/cas/cas-alfresco.amp /opt/iParapheur/amps
   ```

3. Déployer cet amp dans le war alfresco

   ```
   /opt/iParapheur/iparaph-updateAMP.sh
   ```

1.2.2. Keystore

1. Génération du keystore via le script `add_chain_jks.sh`

```
cd /opt/iParapheur && /opt/_install/cas/config/add_chain_jks.sh **URL_SERVEUR_CAS**
```

2. Ajouter dans `/opt/iParapheur/tomcat/script/ctl.sh` en Ligne **13** et **26** dans la variable **JAVA_OPTS**

```
-Djavax.net.ssl.trustStore=/opt/iParapheur/truststore.jks
-Djavax.net.ssl.trustStorePassword=libriciel
```

1.2.3. Fichiers de configurations

1. Copier le fichier `web.xml` d'alfresco

```
cp /opt/_install/cas/config/web.xml /opt/iParapheur/tomcat/webapps/alfresco/WEB-INF/web.xml
```

2. Copier le fichier `cas.properties`

```
cp /opt/_install/cas/config/cas.properties /opt/iParapheur/tomcat/shared/classes/cas.properties
```

3. Copier le fichier `share-config-custom.xml`

```
cp /opt/_install/cas/config/share-configcustom.xml
/opt/iParapheur/tomcat/shared/classes/alfresco/web-extension/
```

4. Modifier la chaîne d'authentification d'alfresco

```
vim /opt/iParapheur/tomcat/shared/classes/alfresco-global.properties
```

Ajouter la propriété suivante :

- `authentication.chain=external1:external`

1.2.4. Éxecution du script de configuration

Tout d'abord, il faut copier le script de mise en place dans le dossier**/opt/iParapheur** et l'adapteur au serveur

```
cp /opt/_install/cas/config/cas-config.sh /opt/iParapheur
vim /opt/iParapheur/cas-config.sh
```

Modifier si besoin les lignes suivantes :

- **11** `URL_CAS_base="cas.test.adullact.org\/cas"`
- **13** `URL_CAS_login="\/login"`
- **15** `URL_PARAPHEUR="https:\/\/parapheur.test.adullact.org"`
- **17** `URL_PROXY_TICKET="\/iparapheur\/proxyTicket"`

Enfin, il faut exécuter le script et relancer le parapheur :

```
chmod +x /opt/iParapheur/cas-config.sh
/opt/iParapheur/cas-config.sh
service alfresco start
```

1.2.5. Fonctionnement attendu

Lors de l'accès au i-Parapheur, une redirection automatique doit être faite vers la page d'authentification du SSO CAS.
Suite au login, l'utilisateur doit être redirigé vers l'acceuil du i-Parapheur.

1.3. Contexte E-Collectivité

Lors de la mise en place du i-Parapheur avec E-collectivité, l'accès à l'API REST doit être **CAS-ifié** afin que la communication puisse se faire sans problème sur le portail.
Un accès **hors-CAS** doit également être possible pour utilisation via l'application Web.

Nous allons donc ajouter une nouvelle entrée pour l'API classique.

Via la configuration suivante, l'accès pour le portail E-collectivité sera : `http://iparapheur.demo.local:8080/alfresco/wcs`

Attention, toutes ces actions sont à effectuer application arrêtée.

1.3.1. CAS-ification de l'API REST

Éditer le fichier `/opt/iParapheur/tomcat/webapps/alfresco/WEB-INF/web.xml`.

Après les blocs `<filter-mapping>` rajouter les blocs :

```
    <filter-mapping>
        <filter-name>CAS Validation Filter</filter-name>
        <url-pattern>/wcs/*</url-pattern>
    </filter-mapping>
    <filter-mapping>
        <filter-name>CAS Authentication Filter</filter-name>
        <url-pattern>/wcs/*</url-pattern>
    </filter-mapping>
    <filter-mapping>
        <filter-name>CAS to Alfresco Authentication Filter</filter-name>
        <url-pattern>/wcs/*</url-pattern>
    </filter-mapping>
```

1.3.2. Nouvelle entrée d'API

Toujours après les `<filter-mapping>` rajouter le bloc :

```
  <filter-mapping>
     <filter-name>WebScript Authentication Filter</filter-name>
     <url-pattern>/wcsip/*</url-pattern>
  </filter-mapping>
```

Puis, après les bloc `<servlet-mapping>` :

```
  <servlet-mapping>
     <servlet-name>wcapiServlet</servlet-name>
     <url-pattern>/wcsip/*</url-pattern>
  </servlet-mapping>
```

Il faut ensuite adapter l'application WEB afin d'utiliser la nouvelle entrée d'API. Pour cela, il faut éditer le fichier `/opt/iParapheur/tomcat/webapps/iparapheur/WEB-INF/surf.xml` , et remplacer `/alfresco/wcs` par `/alfresco/wcsip` dans `<endpoint-url>`.

Suite à ces manipulations, redémarrer le parapheur.

2. LEMONLDAP::NG

2.1. Configuration du LemonLDAP::NG

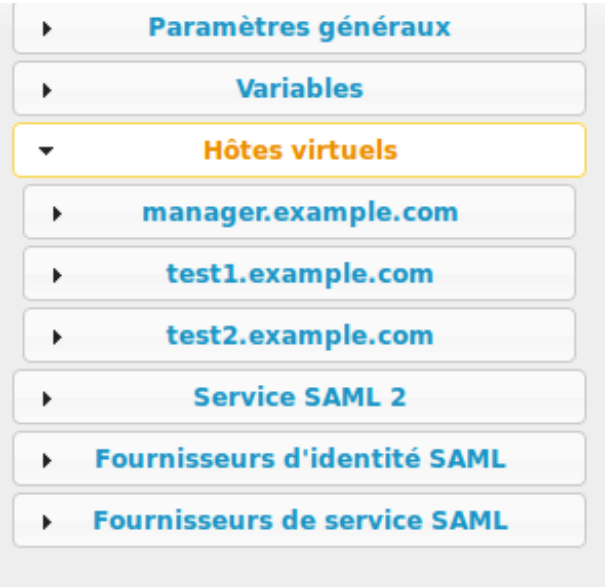
Cette configuration est basée sur un LemonLDAP::NG version 1.4 d'exemple en `*.example.com`.

> Attention ! Le i-Parapheur 4.2+ utilisant la technologie web-socket, Apache2 doit avoir le module `proxy_wstunnel` actif. Cette fonctionnalité n'est incluse qu'à partir de la version 2.4 d'Apache2.

> Pour rendre Apache2.2 (Debian ou Ubuntu) compatible, suivre cette procédure : https://blog.crystalyx.net/utiliser-socket-io-et-les-websockets-avec-apache/

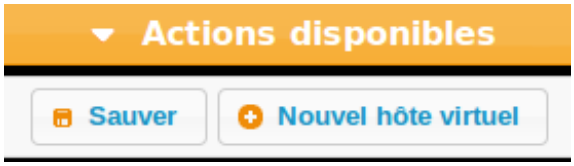
2.2. Création de l'hôte virtuel

Premièrement, se rendre sur http://manager.example.com/ puis sélectionner "Hôtes virtuels" sur le menu de gauche.



Hôtes virtuels

A partir de cet écran, cliquer sur le bouton "Nouvel hôte virtuel".



Nouvel hôte virtuel

Puis, donner un nom à cet hôte (dans le cas de l'exemple, ce serc `iparapheur.example.com`). Il faut ensuite créer le vhost dans apache, un exemple de configuration ci-dessous :

```
<VirtualHost *:80>
    ServerName parapheur.example.com
    Redirect permanent / https://parapheur.example.com/
```

```
</VirtualHost>

<VirtualHost *:443>
    ServerName parapheur.example.com

    #SSO protection
    PerlHeaderParserHandler Lemonldap::NG::Handler
    RequestHeader set X-Forwarded-Proto "https"
    SSLProxyEngine On
    SSLEngine on
    SSLCertificateKeyFile /etc/apache2/ssl/parapheur.example.com.key
    SSLCertificateFile /etc/apache2/ssl/parapheur.example.com.pem
    SSLCACertificatePath /etc/apache2/ssl/validca

    # Changer les en-têtes "Location" dans les redirections
    Header edit Location ^http:(.*) https:$1

    ProxyRequests On
    ProxyPreserveHost Off

    # Reverse-Proxy
    RewriteEngine On
    RewriteCond %{REQUEST_URI}  ^/socket.io            [NC]
    RewriteCond %{QUERY_STRING} transport=websocket    [NC]
    RewriteRule /(.*)           ws://parapheur.test.org:8081/$1 [P,L]

    ProxyPass        /socket.io http://parapheur.test.org:8081/socket.io
    ProxyPassReverse /socket.io http://parapheur.test.org:8081/socket.io

    ProxyPass / https://parapheur.test.org/
    ProxyPassReverse / https://parapheur.test.org/

    # Changer le domaine des cookies
    ProxyPassReverseCookieDomain lemonldap parapheur.example.com
    ErrorLog /var/log/apache2/parapheur-error.log
    CustomLog /var/log/apache2/parapheur-access.log combined
</VirtualHost>
```

2.3. Configuration des règles spécifiques

Dans l'hôte nouvellement créé, sélectionner "Règles" dans le menu de gauche. Il faut commencer par autoriser l'accès à l'application aux utilisateurs. Pour cela, modifier la règle **default**, en renseignant comme valeur **accept**. Il faut ensuite créer de nouvelles règles.

2.3.1. Déconnexion

Cette règle permet la déconnexion de l'utilisateur de LemonLDAP::NG et la redirection vers http://auth.example.com/ lors de la déconnexion au i-Parapheur.

Déconnexion

2.3.2. Applet de signature

Ceci est une règle spécifique pour l'accès à l'applet de signature. Sans cette règle, java n'autorisera pas la récupération et le lancement de l'applet.

Applet

2.4. En-têtes HTTP

Ici sera renseigné l'en-tête permettant l'authentification sur le i-Parapheur. Nous nous baserons sur les valeurs par defaut de LemonLDAP::NG lors de la création du hôte virtuel, c'est à dire un header **Auth-User** ayant pour valeur l'**uid** de l'utilisateur.

Pour rendre fonctionnelle l'authentification via header HTTP, il faut modifier les fichiers de propriétés du i-Parapheur.

- `/opt/iParapheur/tomcat/shared/classes/alfresco-global.properties` :

```
parapheur.auth.external.header.authorize=true
```

- `/opt/iParapheur/tomcat/shared/classes/iparapheur-global.properties` :

```
parapheur.auth.external.header.name=Auth-User
```

```
parapheur.auth.external.header.regexp=.*
```