Quelques pistes de recherche de problèmes

Les causes les plus fréquentes de dysfonctionnement sont :

- requêtes https filtrées par un firewall ou un routeur
- certificats https non valides, ou chaine de certification non transmise par le serveur W3.
  Dans les exemples qui suivent, ent.univ.fr est un proxy cas, auth.univ.fr est le serveur CAS.

S'assurer qu'un firewall ou un routeur ne filtre pas les requêtes

A l'aide de wget, on va s'assurer que les connexions https directes sont autorisées (adapter les ports TCP en fonction de votre configuration.

Coté ent

```
wget -O /tmp/cas.log "https://auth.univ.fr:443/proxyValidate?ticket=PT-1-xxx&service=[https://foo.fr]"
```

Le fichier /tmp/cas.log devrait contenir :

```
<cas:serviceResponse xmlns:cas='http://www.yale.edu/tp/cas'>
<cas:authenticationFailure code='INVALID_TICKET'>
ticket 'PT-1-xxx' not recognized
</cas:authenticationFailure>
</cas:serviceResponse>
```

Coté serveur CAS

Même technique, en choisissant une URL valide. ex :

```
wget \-O /tmp/ent.log "https://ent.univ.fr:443/HealthCheck.html"
```

et /tmp/ent.log doit contenir :

```
<html>
<body>
Health Check ent2
</body>
</html>
```

Controler les certificats

Nous supposons ici que les certificats serveurs du CRU sont utilisés.

Deux méthodes possibles.

Depuis un navigateur W3

La procédure peut différer légèrement d'un navigateur à l'autre.

S'assurer que seul le certificat de l'AC racine du CRU est présente dans le magasin de certificats : pas le certificat ac-serveur ni de certificat du serveur lui-même.

Lancer https://auth.univ.fr

Aucun warning ne devrait être généré par le navigateur.

Cliquer sur le cadenas en bas à droite du navigateur.

Voir le détail du certificat. En particulier, le certificat doit être vu comme valide, et la chaîne de certification doit être affichée :

ac-racine -> ac-serveur -> auth.univ.fr

Faire la même chose avec https://ent.univ.fr

Avec openssl

Faire :

```
openssl s_client -host auth.univ.fr -port 443
```

Devrait sortir qq chose comme cela :

```
CONNECTED(00000003)
depth=2 /C=FR/O=CRU/CN=ac-racine/emailAddress=ca-admin@cru.fr
verify error:num=19:self signed certificate in certificate chain
verify return:0
---
Certificate chain
0 s:/C=FR/O=0541508W/CN=auth.univ-nancy2.fr/emailAddress=reseau@univ-nancy2.fr
i:/C=FR/O=CRU/CN=ac-serveur/emailAddress=ca-admin@cru.fr
1 s:/C=FR/O=CRU/CN=ac-serveur/emailAddress=ca-admin@cru.fr
i:/C=FR/O=CRU/CN=ac-racine/emailAddress=ca-admin@cru.fr
2 s:/C=FR/O=CRU/CN=ac-racine/emailAddress=ca-admin@cru.fr
i:/C=FR/O=CRU/CN=ac-racine/emailAddress=ca-admin@cru.fr
---
Server certificate
-----BEGIN CERTIFICATE-----
MIIEWzCCA0OgAwIBAgICAjgwDQYJKoZIhvcNAQEEBQAwUDELMAkGA1UEBhMCRlIx
....
-----END CERTIFICATE-----
subject=/C=FR/O=0541508W/CN=auth.univ-nancy2.fr/emailAddress=reseau@univ-nancy2.fr
issuer=/C=FR/O=CRU/CN=ac-serveur/emailAddress=ca-admin@cru.fr
---
No client certificate CA names sent
---
SSL handshake has read 3662 bytes and written 340 bytes
---
New, TLSv1/SSLv3, Cipher is DHE-RSA-AES256-SHA
Server public key is 1024 bit
SSL-Session:
.... Verify return code: 19 (self signed certificate in certificate chain)
---
```

Il Faut s'assurer que la chaine de certification est bien transmise.

Utilitaire testHTTPS

Le programme Java testHTTPS est autonome, il permet de faire une requête https vers un serveur web.

Télécharger : testHTTPS.tar.gz

Voici le script de lancement fourni dans le tar.gz :

```
export JAVA_HOME=/usr/java/jdk1.5
HOST=auth.univ-nancy2.fr
PORT=443
#HTTP_VER=HTTP/1.1
#ENCODING=ISO-8859-1
#
#### ATTENTION #####
# Le parametre host est obligatoire.
# le defaut est 443 pour le port, "HTTP/1.0" pour le protocole, "ISO-8859-1" pour l'encoding
# l'ordre des parametres est important. Par exemple, si on veut specifier HTTP_VER (HTTP/1.0, par exemple),
# il faut decommenter les parametres PORT et HTTP_VER
#
#$JAVA_HOME/bin/java testHTTPS $HOST $PORT $HTTP_VER $ENCODING
$JAVA_HOME/bin/java -Djavax.net.ssl.trustStore=/Cert/ac-racine-cru.keystore testHTTPS $HOST $PORT $HTTP_VER
$ENCODING
#$JAVA_HOME/bin/java -Djavax.net.ssl.trustStore=/Cert/esup-portail.keystore testHTTPS $HOST $PORT $HTTP_VER
$ENCODING
```

Cet utilitaire permet de s'assurer que que keystore passé dans le paramètre javax.net.ssl.trustStore permet à la JVM de valider le certificat (ou la chaine de certification) proposée par le serveur https.

Si le paramètre javax.net.ssl.trustStore n'est pas précisé, c'est le keystore de la JVM (jre/lib/security/cacerts) qui est utilisé pour la validation du certificat.