SSO Eurécia

How to manage SSO between Eurécia and external Applications

Purpose

This document describes the way to manage SSO connection between Eurécia and external applications. The users logged to the external application by entering his credentials then access to Eurécia without authenticate themselves again. They will be able to access to their Eurécia application datas.

Uses cases

1. The user authenticates to an external application
2. He clicks to a link to access to Eurécia. This link is formatted like
   https://plateforme .eurecia.com/eurecia/sso?source=src&token=
   Z2VvZmY7bGFuZ29zOzEwMTIzNBsxMDIzNDtnbGFuZ29zQG1vcnRnYWdl
   LmNvbTttb3J0Z2FnZTsyMBEyLTA1LTE0VBIwOjI1OjAyLjMxMFo7M0FDRE
   JFOTMzQzc0NTdDMTNDNjNCMjEzMjYxNjEzNERDNTgzRDA5RA==
3. The token is composed as:
   a. Email
   b. TimeStamp (GMT)
   c. Signature
4. Eurécia receives the token and decrypt it with his private key
5. The user access to Eurécia once the token is validated after decryption.

Before starting
Set the trust between the 2 applications

Setting the trust between Eurécia and the external application by exchanging X509 certificates. These keys will be used to decrypt the token. Eurécia will verify that the expeditor, the external application, is the good one by comparing the hash to the decrypted message.

How to generate and exchange certificates :

First of all, you need to generate an RSA key pair with a size of 1024 bytes and generate an X509 certificate from the public key.

An easy way to do this is to use the openssl tool as following :

```
openssl req -new -newkey rsa:1024 -days 365 -nodes -x509 -keyout privateKey.key -out xopenss509cert.cert
```

The Eurécia certificate is available through the Eurécia platform at the company configuration page.

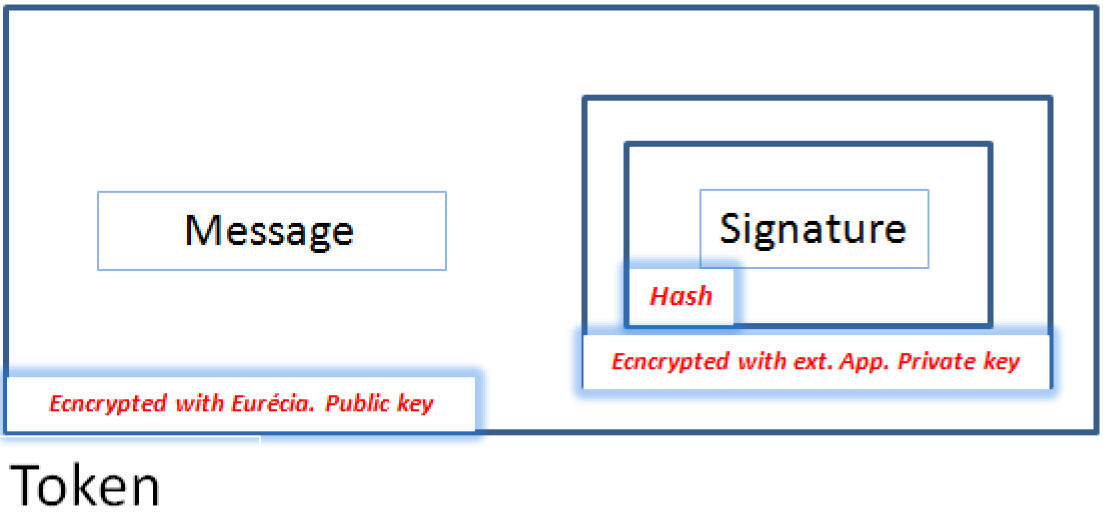
The same page allows you to upload your certificate and link it to the source name of your choice.

Token composition

Encoded algorithm used	BASE 64 (url Safe)
Encryption algorithm used	RSA Sha1 (http://hash.online-convert.com/sha1-generator)
Email	String = User email
Encoding String	UTF-8
Hash	Sha1
TimeStamp	yyyy"-"MM"-"dd"T"HH":"mm":"ss"Z"
Signature	SignedWithExternalAppPrivateKey(Hash(Email ;TimeStamp))

Example:

Email	Jean.dupont@eurecia.com
TimeStamp	2013-01-23T20:25:02.310Z
Signature	Email ;TimeStamp ;Z2VvZmY7bGFuZ29zOzEzNxMFo7zNEZBNTgzRDA5RA==


Message
Signature
Hash
Ecncrypted with ext. App. Private key
Ecncrypted with Eurécia. Public key
Token

Steps to follow by the external application before sending the request

1. Token=message;signature
2. Generate a TimeStamp with the GMT time zone
3. Build the *message* = Email;TimeStamp encoded in UTF-8. For example jean.dupont@eurecia.com; 2013-01-23T20:25:02.310Z
4. Build the *signature* to be sent. *signature*= Hash SHA-1 the message before encrypt it with the external application private key; let's say *signature=CryptedWithExternalAppPrivateKey(Hash(Email ;TimeStamp))*
5. Encrypt the Token with the public key given by Eurécia
6. Encode the crypted token with Base64 url safe
7. Send the Token with the url like https://plateforme.eurecia.com
   /sso?source=yourOrganisation&token=
   Z2VvZmY7bGFuZ29zOzEwMTIzNDsxMDIzNDtnbGFuZ29zQG1vcnRnYWdl
   LmNvbTttb3J0Z2FnZTsyNDEyLTA1LTE0VDIwOjI1OjByLjMxMFo7M0FDRE
   JFOTMzQzc0NTdDMTNDNjNCMjEzMjYxNjEzNEZBNTgzRDA5RA

Once Eurécia receives the request

1. Eurécia receives the Token then decrypt it with his private key. Get the result like message;signature
2. Eurécia explode the message with the " ; " separator into 3 elements, *jean.dupont@eurecia.com; 2013-01-23T20:25:02.310Z; CryptedWithExternalAppPrivateKey(Hash(Email ;TimeStamp))*
3. Eurécia decrypt the CryptedWithExternalAppPrivateKey(Hash(Email ;TimeStamp)) with the External application public key. So the result is Hash(Email ;TimeStamp)
4. Eurécia compare the Hash(jean.dupont@eurecia.com; 2013-01-23T20:25:02.310Z) to the Hash(Email ;TimeStamp) to confirm that the external application is the good one
5. Finally Eurécia compare the TimeStamp received with a newly generated TimeStamp and refuses to log in the user if the TimeStamp received is older than 1 hour. (This is configurable via the Eurécia platform)

JAVA Sample code

```
SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd'T'HH:mm:ss");
sdf.setTimeZone(TimeZone.getTimeZone("GMT"));
String message = "user@eurecia.com;"+sdf.format(new Date());;
PrivateKey myPrivateKey = getMyPrivateKey;
PublicKey eureciaPublicKey = getEureciaPublicKey();

//Compute and sign the hash
Signature sig = Signature.getInstance("SHA1withRSA");
sig.initSign(myPrivateKey);
sig.update(message.getBytes("UTF-8"));
byte[] digest = sig.sign();

//Concat the message
ByteArrayOutputStream outputStream = new ByteArrayOutputStream();
message += ";";
outputStream.write( message.getBytes("UTF-8") );
outputStream.write( digest );
byte[] messageBytes = outputStream.toByteArray( );

//Encrypt message
Cipher cipher = Cipher.getInstance("RSA/ECB/PKCS1PADDING");
cipher.init(Cipher.ENCRYPT_MODE, eureciaPublicKey);
byte[] cipherText = cipher.doFinal(messageBytes);
token = Base64.encodeBase64URLSafeString(cipherText);
```