Enhanced Client or Proxy (ECP)

Cas d'utilisation

Le profile ECP permet à une entité donnée, généralement un navigateur web, Proxy (passerelle WAP), d'accéder aux ressources d'un Fournisseur de Service "SP" en relayant l'authentification d'un utilisateur auprès d'un Fournisseur d'Identité "IDP".

Intégration en python

Ci dessous un exemple d'utilisation de Lasso pour implémenter le profile ECP en python.

Accès depuis une application ECP à un service

Une application de type ECP veut accéder à une ressource d'un fournisseur de service (SP). Pour cela elle doit informer le SP qu'elle est capable de se comporter en ECP en incluant les en-têtes HTTP suivants dans sa requête :

- l'en-tête *HTTP_ACCEPT* initialisé à *application/vnd.paos+xml*
- l'en-tête *HTTP_PAOS* initialisé à *urn:liberty:paos:2003:08*

Le Fournisseur de Service retourne une requête d'authentification

Le SP vérifie si l'application est authorisé à accéder ou non à la ressource demandée. Si non, alors il construit une requête d'authentification qu'il retourne à l'ECP sous la forme d'un message PAOS grâce au code Lasso ci-dessous :

```
login = lasso.Login(sp_lasso_server)
login.initAuthnRequest(None, lasso.HTTP_METHOD_SOAP)
login.request.nameIDPolicy.format = lasso.SAML2_NAME_IDENTIFIER_FORMAT_PERSISTENT
login.request.nameIDPolicy.allowCreate = True
login.request.forceAuthn = False
login.request.isPassive = False
```

Le SP peut inclure un message de type relay state qui lui sera retourné à la fin du processus d'authentification :

```
login.msgRelayState = 'xxxxxx'
```

Le message PAOS est construit avec les appels suivants :

```
login.buildAuthnRequestMsg()
paos_authn_request = login.msgBody
```

l'ECP transmet la requête d'authentification au fournisseur d'identité (IDP)

Une fois la requête reçue, l'ECP en extrait le message SOAP qu'il va devoir retransmettre à l'IDP :

```
ecp = lasso.Ecp()
ecp.processAuthnRequestMsg(paos_authn_request)
soap_request_msg = ecp.msgBody
```

l'IDP authentifie l'utilisateur et retourne une réponse

A ce stade, l'IDP traite la requête et authentification l'utilisateur, puis retourne un message à l'ECP :

```
login = lasso.Login(idp_lasso_server)
login.processAuthnRequestMsg(soap_request_msg)
lasso_login_dump = login.dump()
login.mustAuthenticate()
```

L'utilisateur est authentifié, l'IDP peut construire la réponse ECP :

```
login.validateRequestMsg(True, True)
```

```
login.buildAssertion(lasso.SAML2_AUTHN_CONTEXT_PASSWORD_PROTECTED_TRANSPORT,
    None, None, None, None)
login.buildResponseMsg(None)
soap_response_msg = login.msgBody
```

l'ECP retransmet la réponse au SP

L'ECP traite la réponse de l'IDP et extrait le message PAOS a retourner au SP :

```
ecp.processResponseMsg(soap_response_msg)
assertionConsumerURL = ecp.assertionConsumerURL
paos_response_msg = ecp.msgBody
```

le SP vérifie le message et valide l'authentification de l'utilisateur

le SP traite la réponse retournée par l'ECP et vérifie que l'utilisateur a bien été authentifié par l'IDP :

```
login = lasso.Login(sp_lasso_server)
login.processResponseMsg(paos_response_msg)
login.acceptSso()
name_identifier = login.nameIdentifier.content
```

Si le SP récupéré le message relay state qu'il avait envoyé auparavant :

```
relay_state = login.msgRelayState
```