

Institut de la Francophonie  
pour Informatique

Institut National  
des Télécommunications

Mémoire de fin d'études  
**Support de sources  
d'authentification multiples dans  
un portail de travail collaboratif**

**DANG Quang Vu**

Responsable de stage: **Christian BAC**

Ce stage a été réalisé au sein du projet **PFTCR** du **département  
Informatique**  
de l'**Institut National des Télécommunications(INT)**

Hanoi, novembre 2006

## **Remerciements**

J'adresse toute ma gratitude à mon responsable de stage, M. Christian BAC, pour sa disponibilité, son soutien constant et son aide précieuse durant ce stage.

Je voudrais également remercier M. Olivier BERGER, M. Benoît HAMET pour leurs collaborations serrées, leurs aides tout au long de mon stage.

Un grand merci à toutes les personnes du département Informatique à l'INT pour leurs aides et leur gentillesse pendant mon séjour en France.

Enfin, je voudrais exprimer mon entière reconnaissance envers ma famille, mes amis et mes professeurs à l'IFI pour leurs soutiens, leurs aides et leurs encouragements.

## Résumé

Dans le cadre du stage effectué dans le département Informatique à l'INT à Evry (France) nous avons abordé le support de sources d'authentification multiples pour les plate-formes de travail collaboratif se basant sur phpGroupWare comme PicoLibre, ProGet. L'utilisation de fédération d'identité est une solution pour utiliser des sources d'authentification multiples.

L'approche proposée et développée dans le cadre du stage est l'utilisation Shibboleth pour créer la fédération d'identités et l'utilisation de l'authentification de serveur Web(par exemple le mod\_auth\_ldap, mod\_auth\_mysql... dans Apache) pour participer à la fédération d'identité. L'Intégration de Shibboleth permet d'offrir l'authentification de type SSO (Single Sign On), l'autorisation.

Un adaptateur est implémenté dans phpGroupWare et utilisé non seulement pour Shibboleth mais encore pour autre mécanisme d'authentification externe. L'adaptateur peut devenir utile également pour d'autres sources d'authentification par Apache(via par exemple mod\_auth\_ldap, mod\_auth\_mysql), Cet adaptateur est intégré dans le code-base standard du module APIs de la nouvelle version 0.9.18 de phpGroupWare.

L'intégration de Shibboleth dans phpGroupWare sera utilisé par les plates formes PicoLibre dans la fédération d'identité de GET/INT.

# Table des matières

Remerciements.....	2
Résumé.....	3
Chapitre 1 . Introduction.....	7
1.1 Contexte du stage.....	7
1.2 Objectifs du stage.....	7
1.3 Organisation du rapport.....	8
Chapitre 2 . État de l'art.....	9
2.1 PhpGroupWare.....	9
2.2 PicoLibre.....	10
2.3 TWiki.....	11
2.4 Single Sign-On (SSO).....	12
2.4.1 Architecture classique de SSO.....	13
2.4.2 SAML.....	15
2.4.3 Approches de SSO.....	16
2.4.3.1 Approche centralisée.....	16
2.4.3.2 Approche fédérative.....	17
2.5 Le choix de Shibboleth.....	19
Chapitre 3 . Shibboleth.....	21
3.1 Composants de Shibboleth.....	21
3.1.1 Fournisseur de services.....	21
3.1.2 Fournisseur d'identités.....	22
3.1.3 Service de découverte.....	23
3.2 Assertions SAML de Shibboleth.....	23
3.3 Fonctionnement de Shibboleth avec WAYF et SSO.....	24
3.4 Fédération de Shibboleth.....	28
3.4.1 Méta données.....	28
3.4.2 Relation de confiance entre les membres d'une fédération. .	29
Chapitre 4 . Réalisation.....	30
4.1 Amélioration d'authentification de PicoLibre.....	30
4.1.1 Schéma d'authentification standard dans phpGroupWare....	30
4.1.2 Shibboleth et Apache pour service de SSO.....	32
4.1.3 Problèmes dans environnement mélangé et legs.....	32
4.2 Implémentation de l'adapteur dans phpGrouWare pour Shibboleth	
.....	33
4.2.1 Utiliser l'authentification via Apache dans le phpGroupWare	34
4.2.2 Mapping REMOTE_USER vers le compte de phpGroupWare	35
4.2.3 Additions à phpGroupWare.....	36
4.2.4 Configuration d'accès à phpGroupWare via Apache.....	37
4.2.4.1 Contrôle d'accès en mode full-apache.....	37
4.2.4.2 Contrôle d'accès en mode semi-apache.....	38
4.2.4.3 Configuration phpGroupWare et Shibboleth pour	
PicoLibre.....	40
Conclusions.....	42
Bibliographie.....	43
Annexe A: Assertions de SAML.....	44
Assertions d'authentification.....	44
Assertions d'attribut.....	44
Assertions signées.....	45
Annexe B: Additions au phpGroupWare.....	46

La paquet phpgwapi.....	46
Nouveau module sso dans phpGroupWare.....	46
Nouvelle table dans base de données.....	47
Annexe C: Intégration de Twiki dans Picolibre.....	47
picolibre_twiki.....	47
Implémentation dans Picolibre.....	47
Implémentation dans TWiki.....	48

## Table des figures

Figure 1: Architecture générale de phpGroupWare.....	10
Figure 2: Architecture générale de PicoLibre.....	11
Figure 3: Authentification sans SSO.....	12
Figure 4: Architecture classique de SSO.....	14
Figure 5: Les produits se basant sur SAML.....	15
Figure 6: Approche centralisée.....	17
Figure 7: Approche fédérative.....	18
Figure 8: Modèle Liberty Alliance.....	18
Figure 9: Modèle Shibboleth.....	19
Figure 10: Composants de SP.....	22
Figure 11: Composants de IdP.....	23
Figure 12: Fonctionnement de Shibboleth.....	28
Figure 13: Schéma standard d'authentification.....	31
Figure 14: Architecture de phpGroupWare avec l'adapteur.....	34
Figure 15: Identification en mode full apache .....	38
Figure 16: Identification en mode semi-apache.....	40
Figure 17: Diagramme des classe d'adapteur pour Shibboleth.....	46

## **Chapitre 1 .Introduction**

### **1.1 Contexte du stage**

Le Groupe des Écoles des Télécommunications (GET) comprend plusieurs grandes écoles d'ingénieurs et de management ainsi que des centres de recherche situés principalement à Paris (ENST), Brest (ENST Bretagne) et Évry (INT) en France. Le groupe compte actuellement 470 enseignants-chercheurs et 500 thésards dans ses laboratoires.

PicoLibre est un système de logiciel libre développé à GET. Il fournit une plate forme de travail de collaboration en se basant sur phpGroupWare et des autres outils de logiciel libre. Plusieurs plate formes de PicoLibre ont été déployées à GET, et les développeurs ou les chercheurs peuvent utiliser des services de plusieurs telles plate formes.

Actuellement les services accessibles par le Web dans PicoLibre nécessitent très souvent une authentification. Différents comptes sont créés sur chaque plate forme, pour la même personne. Il se pose plusieurs problèmes : comptes multiples, authentifications multiples, sécurité, différents mécanismes d'authentification, aspects multi-établissements, autorisations etc. Un mécanisme de type Single Sign-On (SSO) semble nécessaire entre ces plate formes qui permettent à un utilisateur de ne procéder qu'à une seule authentification pour accéder à plusieurs applications. De plus il est nécessaire de créer une fédération d'identité qui regroupe un ensemble établissements pour supporter les sources d'authentification multiples.

### **1.2 Objectifs du stage**

Le cadre du stage est effectué dans le projet structurant PFTCR(Plate-Formes de Travail Collaboratif pour la Recherche) de l'équipe de systèmes répartis dans le département Informatique à l'INT. Le sujet du stage est le support de sources d'authentification multiples pour les plate-formes de travail collaboratif de type PicoLibre. L'objectif est de fournir un point sur l'état de l'art en matière de solutions de partage d'authentification du type Shibboleth. Cela doit notamment permettre de fournir des fonctionnalités de Single Sign-On pour les utilisateurs PicoLibre et ProGet, ainsi que de s'orienter vers un réseau de plate-

formes distribuées avec la fédération d'identité. Cela permettra idéalement de prescrire les adaptations nécessaires dans le système d'informations GET, et d'implémenter les adaptations requises du côté phpGroupWare.

### ***1.3 Organisation du rapport***

L'état de l'art est réalisé dans le chapitre 2, il donne une vue générale sur la plate forme PicoLibre et ses logiciels. Ce chapitre présente aussi une synthèse sur la mécanisme Single Sign-On et ses approches.

Une description de Shibboleth ses composants, les fonctionnalités de Shibboleth, et de la fédération d'identités de type Shibboleth sont présentées dans le chapitre 3. Le chapitre 4 présente l'approche et les résultats obtenus dans la phase de réalisation. Le rapport est terminé par des conclusions, bibliographie et annexes.

## **Chapitre 2 .État de l'art**

### **2.1 PhpGroupWare**

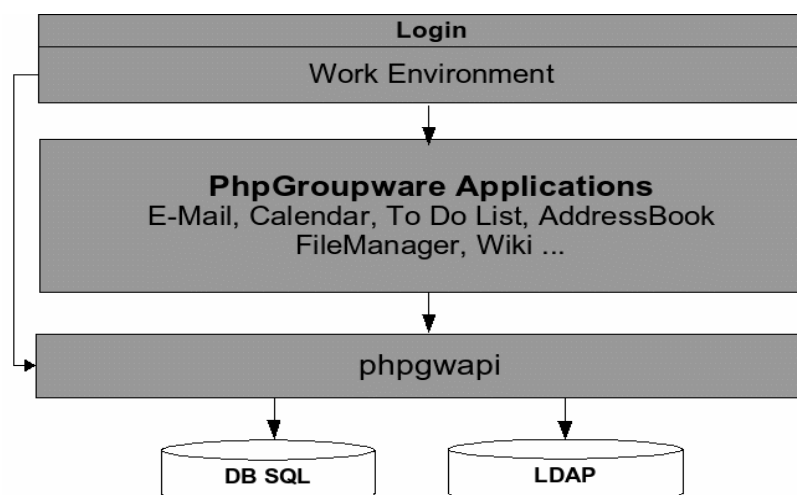
PhpGroupWare est un projet OpenSource écrit en php sous licence GNU General Public Licence (GPL). Il est un serveur d'applications, et est déployé sur le système d'information d'un établissement. Il fournit des applications bureautiques en mode Web ainsi que des applications pour le travail collaboratif. Les utilisateurs disposent d'un login et d'un mot de passe pour s'authentifier, et accèdent ainsi à leur application. Chaque utilisateur peut définir et configurer les services qu'il souhaite utiliser. Toutes les informations sont stockées dans une seule base de donnée, ce qui facilite les sauvegardes et les restaurations.

PhpGroupWare est mutli plate-formes. Il est indépendant de la plate forme parce que il est codé en php. Il peut fonctionner sous différents serveurs Web (Apache, IIS...) et systèmes d'exploitation.

PhpGroupWare a des possibilités infinies grâce aux APIs. PhpGroupWare propose déjà de nombreuses applications (au moins toutes les standards), mais il fournit en plus des APIs complètes pour permettre à client de concevoir et d'intégrer sa propre application. Depuis des APIs, beaucoup de développeurs ont conçus de nouvelles applications. Alors Le projet phpGroupWare s'enrichit régulièrement de nouvelles fonctionnalités

Les applications de bases de phpGroupWare proposées dans le package par défaut sont un gestionnaire d'applications, client mail (POP3, IMAP...), forums, notes, agendas, calendriers, Preferences (configuration), gestionnaire de fichiers ...

La figure 1 montre l'architecture générale de phpGroupWare.



*Figure 1. Architecture générale de*

PhpGroupWare gère des permissions d'accès à ces applications de façon autonome, pour des utilisateurs connus localement, en basant sur un "compte" local, stocké dans un "annuaire" mis en oeuvre par exemple dans une base de données relationnelle MySQL, ou un annuaire LDAP. Pour pouvoir accéder à phpGroupWare, l'utilisateur doit avoir un compte qui détermine le profil de l'utilisateur : les droits d'accès pour la liste des applications que l'utilisateur peut utiliser. PhpGroupWare supporte des types d'authentifications: SQL, LDAP, et mail. . Ces méthodes d'authentification sont implémentés dans l'application, et utilisent ici un "annuaire" local, qui a été indiqué par l'administrateur dans la phase de configuration du serveur phpGroupWare, à la fin de la procédure d'installation. Elles n'offrent pas un service SSO, qui permette à phpGroupWare de donner un accès "transparent" sans nouvelle authentification pour les utilisateurs déjà connus dans d'autres services du système d'information de l'établissement.

## **2.2 PicoLibre**

La plateforme PicoLibre est une plate forme de travail collaboratif se basant sur les APIs et les applications de base de phpGroupWare. Elle héberge des projets, offre un bureau virtuel permettant aux utilisateurs de travailler de façon sécurisée dans leurs projets. C'est un espace de travail associé à un ensemble d'outils. Ces outils gèrent les différents registres d'activité de la vie d'un projet, communication au sein de l'équipe et communication externe (mailing-lists Sympa), planification des tâches, documentation du projet, suivi de bogues, mise à disposition des

sources (CVS).

Accessible à partir de tout navigateur Web, PicoLibre offre une maîtrise complète de sécurisation des accès en utilisant l'authentification de phpGroupWare . Un projet peut être visible de tout l'Internet alors qu'un autre n'est accessible que par un groupe de personnes identifiées. La figure 2 montre l'architecture générale de PicoLibre.

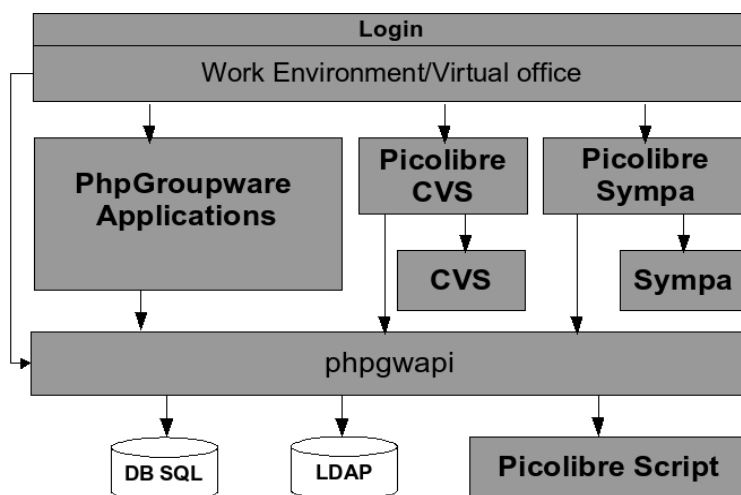


Figure 2: Architecture générale de PicoLibre

Aujourd'hui deux plate-formes PicoLibre sont déployées pour les enseignants chercheurs de GET : PicoLibreINT et PicoLibreENSTBrestagne. Une instance de PicoLibre est installée à l'AUF, pour permettre le développement de logiciels dans le cadre du programme Centres Linux et Logiciels Libres pour le Développement (C3LD).

PicoLibre est une plate-forme regroupant un ensemble des logiciels libres comme environnement de travail collaboratif. Pour étendre les fonctionnalités offertes on envisage l'intégration d'autres applications comme Twiki, Subversion ... De plus pour faciliter le déploiement inter sites, on décide d'intégrer un service SSO entre les applications constituant PicoLibre et les autres applications des établissements.

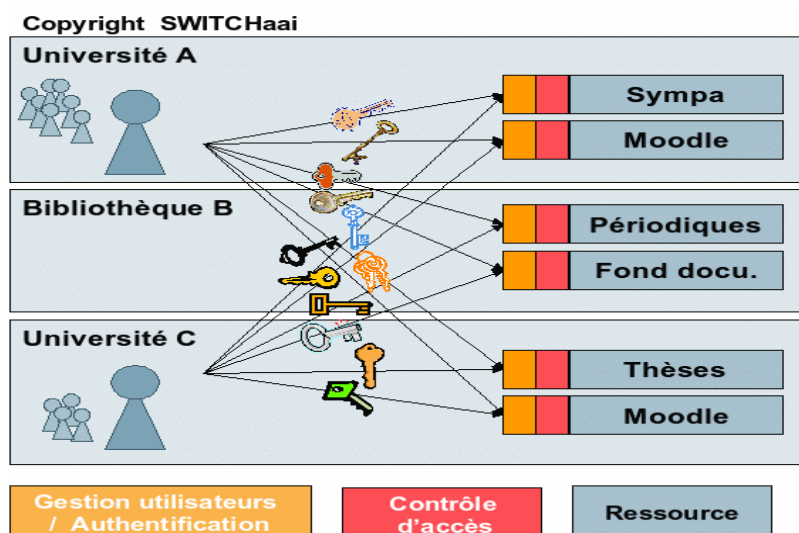
### 2.3 TWiki

Twiki est une plate forme de collaboration pour l'entreprise, flexible, puissante et simple à utiliser. C'est un Wiki structuré, typiquement utilisé pour héberger un espace relatif au développement d'un projet, un système de gestion de documents, une base de connaissances ou tout

autre outil de collaboration. Le contenu Web peut être créé de manière collaborative en utilisant juste un navigateur. Les utilisateurs sans connaissance en programmation peuvent créer des applications Web. Les développeurs peuvent étendre les fonctionnalités de TWiki avec des plug-ins.

## **2.4 Single Sign-On (SSO)**

Les services numériques accessibles par le Web (intranet, courrier électronique, forums, agendas, applications spécifiques) à disposition des utilisateurs se sont multipliés. Ces services nécessitent très souvent une authentification. La figure 3 montre la multiplication des authentifications des services accessibles par le Web en quelques années (sans SSO).



*Figure 3: Authentification sans SSO*

L'utilisation de techniques de synchronisation entre domaines d'authentification hétérogènes, puis de serveurs LDAP a permis la mise en oeuvre d'un compte unique (login / mot de passe) pour chaque utilisateur, ce qui est un progrès. Se posent maintenant les problèmes suivants :

- authentifications multiples: Il est nécessaire d'entrer son login/mot de passe lors de l'accès à chaque application.
- sécurité: Le compte étant unique, le vol de mot de passe devient un risque très important. On souhaite fortement que les applications n'aient pas connaissance du mot de passe.

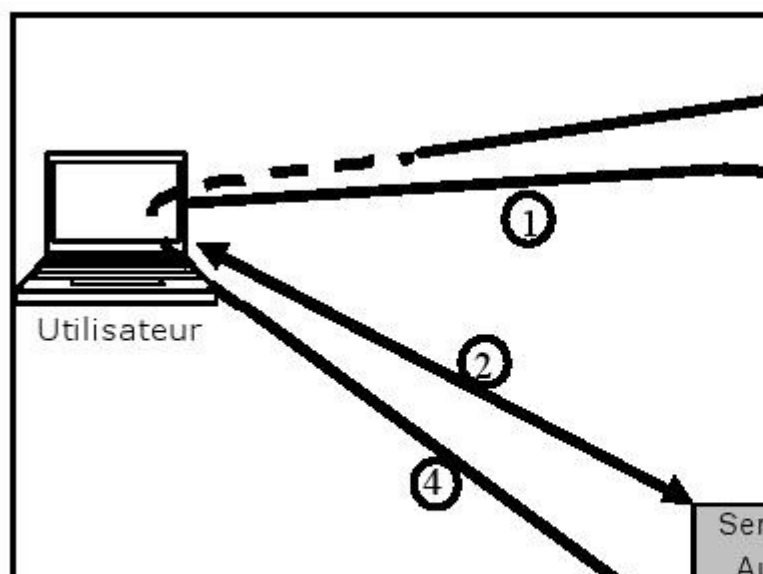
- différents mécanismes d'authentification: Maintenant on peut utiliser plusieurs mécanismes d'authentification comme les certificats X509 , un annuaire LDAP, mail... Il semble donc intéressant de disposer d'un service d'abstraction par rapport aux mécanismes d'authentification locaux.
- aspects multi-établissements: Le compte d'un utilisateur est unique à l'intérieur de l'établissement. Il serait souhaitable que l'accès à des ressources informatiques d'un autre établissement puisse se faire à l'aide du même compte.
- autorisations: il est nécessaire pour certaines applications de pouvoir disposer d'informations définissant les rôles des utilisateurs.

Les mécanismes de SSO (Single Sign-On : authentification unique, et une seule fois) tentent de répondre à ces problématiques. C'est un mécanisme permettant à un utilisateur de ne faire qu'une seule authentification pour accéder à plusieurs applications informatiques (ou sites Web sécurisés). Elles utilisent les techniques suivantes:

- une centralisation de l'authentification sur un serveur qui est le seul à recueillir les mots de passe des utilisateurs, à travers un canal chiffré .
- des redirections HTTP transparentes du navigateur client, depuis les applications vers le serveur d'authentification, puis du serveur vers les applications.
- le passage d'informations entre le serveur d'authentification et les applications à l'aide de cookies, ou de paramètres CGI de requêtes HTTP (GET ou POST).

### **2.4.1 Architecture classique de SSO**

L'architecture de la plupart des produits de SSO repose sur les concepts de base qui sont montrés dans la figure4 :



**Les applications** sont déchargées du travail d'authentification des utilisateurs. Cette tâche est assurée par un serveur d'authentification dédié.

**Le serveur d'authentification** délivre des tickets au client (maintient de la session d'authentification) et aux applications (transmission de l'identité de l'utilisateur). Ce second ticket transite également par le client.

L'application ne recueille jamais les éléments d'authentification de l'utilisateur (login + mot de passe).

Il existe une relation de confiance entre les applications et le serveur d'authentification. Par exemple des certificats X509 (utilisant des algorithmes asymétriques) peuvent être utilisés dans l'architecture du système pour créer la relation de confiance.

Le serveur d'authentification est l'élément central du système de SSO puisqu'il assure l'authentification de l'utilisateur, la persistance de sa connexion et la propagation de l'identité de l'utilisateur auprès des applications.

**L'agent d'authentification** est la brique du SSO intégrée à l'application cible par exemple sous forme d'une bibliothèque applicative ou d'un module apache. L'agent vérifie que l'utilisateur est authentifié. S'il ne l'est pas, il le redirige vers le serveur d'authentification. Si le client s'est déjà authentifié auprès du serveur d'authentification (attesté par la présence d'un cookie) le serveur le redirige directement vers l'agent d'authentification demandeur, de façon non bloquante. Lorsque

l'utilisateur revient du serveur d'authentification, authentifié, l'agent vérifie l'origine des données (les données peuvent être signées) et les transmet à l'application.

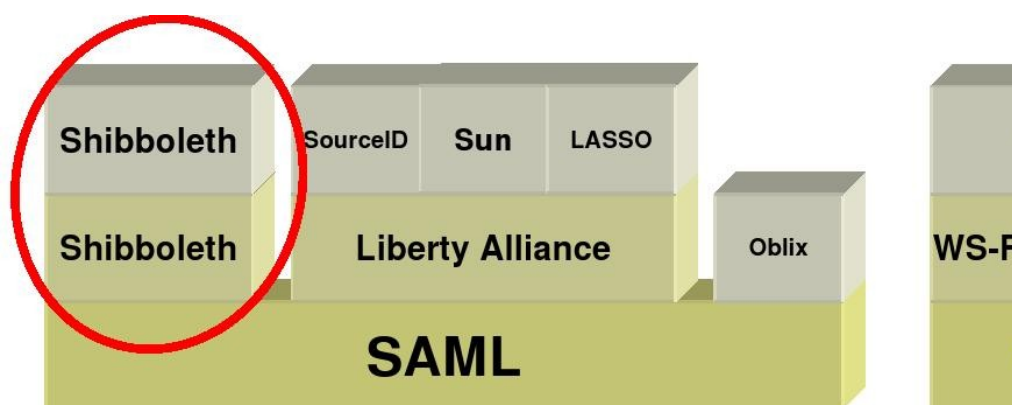
### **2.4.2 SAML**

Security Assertion Markup Language (SAML) est une norme de XML pour échanger des données d'authentification et d'autorisation entre les domaines de sécurité, c'est-à-dire, entre un fournisseur d'identité et un fournisseur de service. SAML est défini par le Comité technique de services de sécurité (Security Services Technical Committee OASIS).

Le problème le plus important que SAML essaye de résoudre est le problème Single Sign-On de Web (SSO). Les solutions de SSO au niveau d'Intranet abondent (en utilisant des Cookie par exemple) mais prolongeant ces solutions au delà de l'Intranet a été problématique et a mené au développement des technologies de propriété industrielle. SAML est devenu la norme définitive pour beaucoup de solutions du Web SSO dans l'espace de problème de gestion d'identité.

SAML suppose que le principal (souvent un utilisateur) dépend au moins d'un fournisseur d'identité et que le fournisseur d'identité fournisse des services locaux d'authentification au principal. Cependant, SAML n'indique pas l'exécution de ces services locaux. En effet, SAML ne s'inquiète pas de comment des services locaux d'authentification sont mis en application.

Il existe des produits visant à construire une fédération d'identité en se basant sur SAML comme le montre la figure 5.



**Shibboleth** est un projet à source ouvert de Internet2 visant à construire la fédération d'identité pour les établissements et les autres partenaires.

Il se base sur le standard SAML pour échanger l'assertion d'authentification et des attributs d'utilisateur.

**Liberty Alliance** est un ensemble de spécifications publiques rédigées par un consortium d'industriels. **LASSO** est une bibliothèque C libre qui implémente les spécifications de Liberty Alliance.

Outre SAML, il existe aussi des spécifications portées par le consortium WS-I (Web Services interoperability), notamment WS-Security (Web Services Security) et WS-Federation (Web Services Federation Language)

.

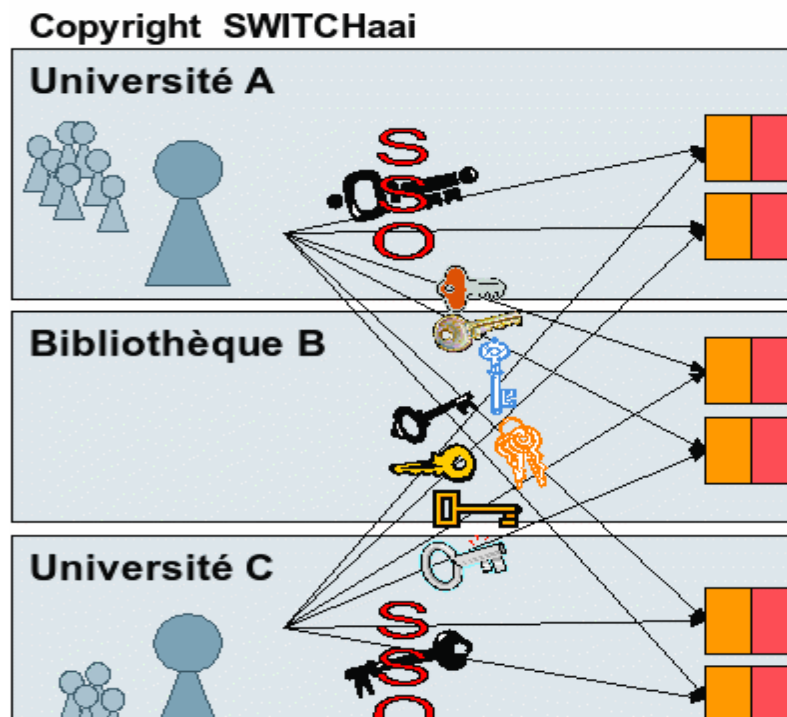
### **2.4.3 Approches de SSO**

Il y a deux approches principales pour offrir le service SSO: l'**approche centralisée (modèle Passport)** et l'**approche fédérative**.

#### **2.4.3.1 Approche centralisée**

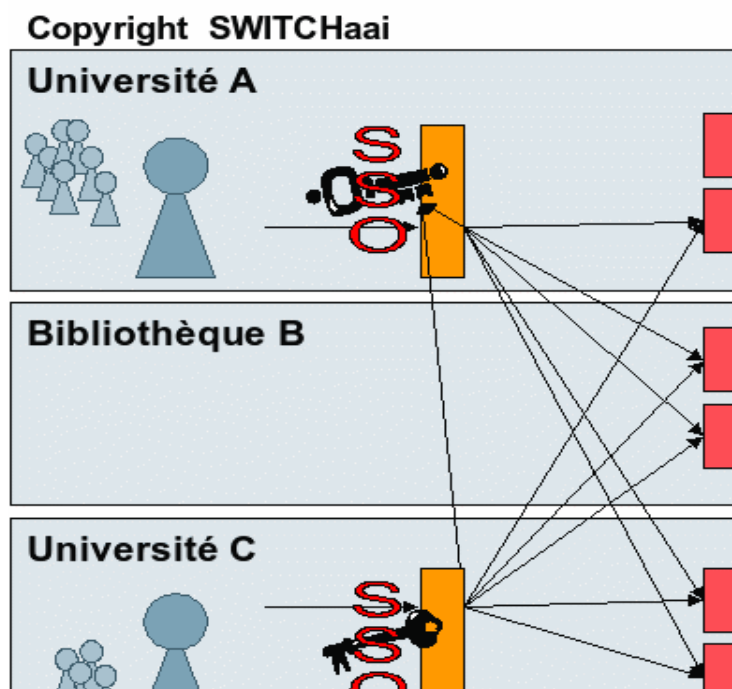
Le principe de base de l'approche centralisée est de disposer d'une base de données globale et centralisée de tous les utilisateurs. Cela permet également de centraliser la gestion de la politique de sécurité d'offrir le service SSO. Un exemple de mise en œuvre est le logiciel libre CAS (Central Authenticate Service).

Cette approche est principalement destinée à des services dépendant tous d'un même établissement, par exemple à l'intérieur d'une société.



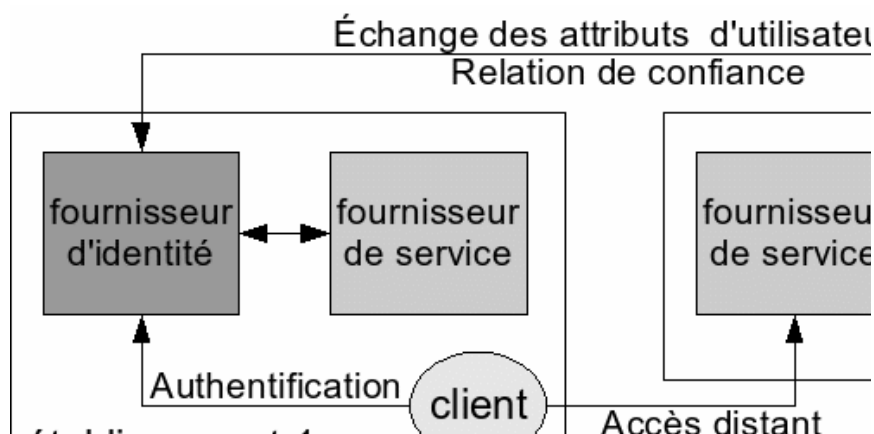
#### **2.4.3.2 Approche fédérative**

Le principe de base de l'approche fédérative est de créer une fédération d'identité qui regroupe un ensemble d'établissements. Normalement chaque établissement a un fournisseur d'identité et un fournisseur de service. La base de données d'utilisateurs est distribuée et il y a la propagation d'identité entre les membres dans la fédération. Alors l'utilisateur doit seulement s'authentifier avec le fournisseur d'identité de son établissement pour accéder aux tous les fournisseurs de service dans la fédération d'identité.



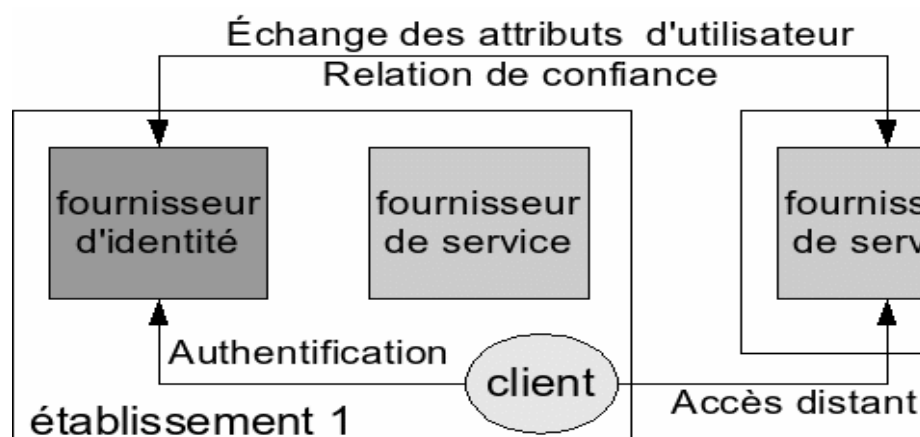
Il y a deux produits de fédération qui se base sur SAML:

**Liberty Alliance (Modèle distribué)**, chaque service gère une partie des données d'un utilisateur (l'utilisateur peut donc disposer de plusieurs comptes), mais partage les informations dont il dispose sur l'utilisateur avec les services partenaires. Ce modèle a été développé pour répondre à un besoin de gestion décentralisée des utilisateurs, où chaque service partenaire désire conserver la maîtrise de sa propre politique de sécurité, comme par exemple un ensemble de sites marchands indépendants d'un point de vue commercial et organisationnel.



**Shibboleth (Modèle coopératif)**, part du principe que chaque utilisateur dépend d'un des établissements. Lorsqu'il accède à un service de la fédération, l'utilisateur est authentifié par son

établissement. Le fournisseur d'identité gère l'authentification et fournit des attributs de l'utilisateur et le fournisseur de service gère le contrôle d'accès. Ce modèle répond notamment aux besoins de structures institutionnelles dans lesquelles les utilisateurs sont dépendants d'un établissement, comme par exemple les universités, les laboratoires de recherche, etc.



## **2.5 Le choix de Shibboleth**

Dans le contexte du support de sources d'authentification multiples pour les plate-formes de travail collaboratif ProGET, PicoLibre les fonctionnalités offertes par Shibboleth correspondent aux principes suivants:

- support Single Sign-On
- support Sources d'authentification multiples avec aspects multi-établissements
- contrôle d'accès se basant sur les attributs d'utilisateur
- basé sur le standard SAML, SSL

De plus la topologie d'une fédération de type Shibboleth correspond bien à la structuration d'un ensemble d'établissements d'université. Dans lesquelles les utilisateurs (étudiants, professeurs, chercheurs...) dépendent de son établissement. L'établissement gère ses utilisateurs de façon indépendante des autres établissements de la fédération. Ceci permet de partager beaucoup de types de ressource comme les ressources statiques (pages html, fichiers .pdf, .doc ...), les applications, les cours en ligne ...

Au niveau d'intégration Shibboleth est un logiciel libre et un produit

complet «prêt à l'emploi». Il se paramètre pour définir les relations entre les acteurs de la fédération en proposant des scénarios adaptés au contexte universitaire et les connecteurs avec le système d'information. Il s'interface bien avec les briques pré existantes d'un système d'information. Il est déployé à grande échelle dans plusieurs pays (USA, Finlande, Suisse et Grande-Bretagne). De nombreuses applications ont intégré Shibboleth avec de bons retours d'expérience. Le CRU (Comité Réseau des Universités) prépare l'ouverture d'un service de fédération propagation d'identités pour les établissements d'enseignement supérieur qui se base sur Shibboleth.

Enfin Shibboleth est un projet actif et ouvert. La version 2.0 de Shibboleth sera compatible avec SAML 2.0. et les besoins d'interopérabilité entre Shibboleth avec les autres produits se basant sur SAML (Lasso,..) seront très probablement satisfaits.

## Chapitre 3 .Shibboleth

Shibboleth permet de constituer une fédération de sites qui partagent des méthodes d'authentification hétérogènes. C'est une extension de SAML qui enrichit ses fonctionnalités de fédération d'identités en facilitant pour un ensemble de partenaires la mise en place de deux fonctionnalités importantes, la délégation d'authentification et la propagation d'attributs.

### 3.1 Composants de Shibboleth

Le concept de base de Shibboleth est basé sur trois composants: le fournisseur d'identité (IdP), le fournisseur de service (SP) et le service de découverte(WAYF).

#### 3.1.1 Fournisseur de services

Le fournisseur de services (Service Provider ou SP) est une partie de Shibboleth écrit en C/C++ qui gère des ressources accédées par les utilisateurs dans la fédération. Il propose des ressources protégées sur la base d'un contexte de sécurité SAML. Le module **mod\_shib** est module plug-in pour le serveur web (Apache) qui contrôle l'accès des utilisateurs en se basant sur les attributs. Les attributs sont obtenus à partir d'un autre composant du SP qui fonctionne en mode daemon (Shibboleth daemon shibd). Ces attributs sont transmis vers les ressources souhaitées.

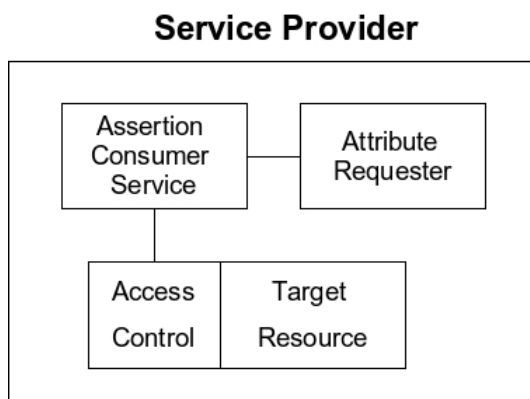
Comme le montre la figure10, un SP est composé de trois sous-composants qui sont appelés : consommateur d'assertions(Assertion Consumer Service), demandeur d'attributs (Assertions Requester),et contrôleur d'accès(Acces Controler).

Le **consommateur d'assertions** agit comme un pré-filtre. C'est lui qui redirige vers l'IdP lorsque l'utilisateur n'est pas authentifié. Il peut être implémenté au niveau du serveur HTTP (par un module Apache ou un filtre J2EE par exemple) ou encore par une librairie, appelée par un applicatif web. Lorsque l'utilisateur est authentifié, alors le consommateur d'assertions transmet le nameIdentifier au demandeur d'attributs.

Le **demandeur d'attributs** est chargé de la récupération des attributs des utilisateurs auprès de l'IdP. Il peut être implémenté comme un démon

(dédié, interrogeable par les processus du SP) ou par une librairie, interrogeable par un applicatif web. Les attributs récupérés par le demandeur d'attributs sont fournis au contrôleur d'accès.

Le **contrôleur d'accès** est chargé d'autoriser ou non l'accès aux ressources demandées. Comme le consommateur d'assertions, il peut être implémenté au niveau du serveur HTTP.



*Figure 10: Composants de SP*

### **3.1.2 Fournisseur d'identités**

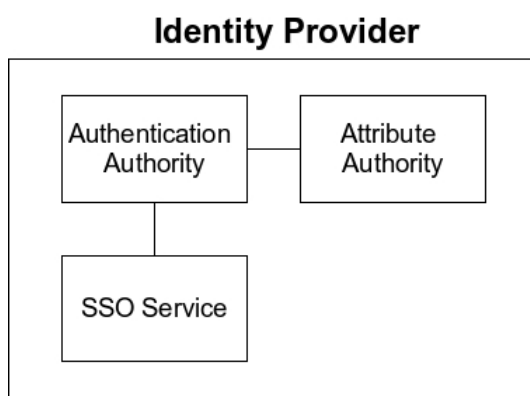
Le fournisseur d'identités (Identity Provider ou IdP) est une entité écrite en Java authentifiant les utilisateurs et fournissant leurs attributs. En général l'IdP est une partie dans le système d'information (SI) de l'établissement et est un membre dans la fédération. Lorsqu'un utilisateur veut accéder à un service offert par les membres de la fédération, il utilise l'IdP de son organisation pour s'authentifier. Comme le montre la figure 11, l'IdP est composé de 3 sous-composants appelés : service d'authentification (Authentication Service ou SSO Service), autorité d'authentification (Authentication Authority), et autorité d'attributs (Attribute Authority).

Le **service d'authentification** (SSO Service) est chargé de l'authentification des utilisateurs vis-à-vis de l'ensemble de l'IdP. C'est lui qui, par exemple, demande à l'utilisateur un couple user/password, puis le valide auprès de la base d'authentification du SI. Les implémentations du service d'authentification peuvent être très variées, depuis un module Apache authentifiant les utilisateurs auprès d'un annuaire LDAP, jusqu'à un client de Single Sign-On .

Le service d'authentification est chargé de transmettre à l'autorité d'authentification l'identifiant unique de l'utilisateur au sein du SI. N'importe quel système d'authentification web peut être utilisé (formulaire applicatif, domaine HTTP, certificat X509 [12], Single Sign-On).

L'**autorité d'authentification** associe le nameIdentifier à l'identifiant de l'utilisateur.

L'**autorité d'attributs** délivre, en réponse à une demande d'un SP, les attributs de l'utilisateur correspondant à un nameIdentifier. L'association entre l'identifiant de l'utilisateur et le nameIdentifier est maintenue par l'autorité d'authentification. Les attributs de l'utilisateur sont récupérés dans le SI de l'établissement, plusieurs sources pouvant être envisagées (annuaire LDAP, base de données...).



*Figure 11: Composants de IdP*

### **3.1.3 Service de découverte**

Le service de découverte (Where Are You From ou WAYF) est un composant supplémentaire dans la fédération qui permet à l'utilisateur de choisir son IdP. Le WAYF peut être utilisé par le SP pour déterminer l'IdP préféré de l'utilisateur avec ou sans interaction de l'utilisateur. Le WAYF est essentiellement un proxy pour la demande d'authentification passée du SP au service SSO de l'IdP.

## **3.2 Assertions SAML de Shibboleth**

Dans Shibboleth, des assertions SAML sont transférées à partir des fournisseurs d'identité aux fournisseurs de service. Les assertions

contiennent les déclarations que les fournisseurs de service peuvent utiliser pour prendre des décisions de contrôle d'accès. Trois types de déclarations sont indiqués par SAML :

- déclaration d'authentification (Authentication statements)
- déclaration d'attribut (Attribute statements)
- déclaration de décision d'autorisation (Authorization decision statements)

Les déclarations d'authentification affirment au fournisseur de service que le principal(souvent un utilisateur) a authentifié avec le fournisseur d'identité en utilisant une méthode particulière d'authentification.

Les déclarations d'attribut fournissent des informations additionnelles au principal de sorte que les fournisseurs de service peuvent prendre des décisions relatives au contrôle d'accès.

Dans certaines situations, il peut être préférable que l'application délègue une décision de contrôle d'accès à un composant ou à un service différent. Dans ce cas, le fournisseur de service indique au service la ressource qui est accédée et le service émet une décision d'autorisation qui dicte si le principal est autorisé à accéder à la ressource.

### **3.3 Fonctionnement de Shibboleth avec WAYF et SSO**

Dans cette partie, nous considérons le cas où un SP est un membre dans une fédération créée par le service WAYF. Donc le SP est accessible à des utilisateurs des établissements différents. Le problème est que le SP ne sait pas rediriger le navigateur vers le bon IdP pour réaliser l'authentification. Le service WAYF est utilisé pour résoudre ce problème. Son rôle est d'orienter les utilisateurs pour sélectionner leur IdP. Le processus de première requête non authentifiée vers un SP est suivant:

(1) Le client demande une ressource cible au SP:

<https://porphyre.int-evry.fr/picolibre>

Le SP exécute un contrôle de sécurité sur la ressource cible. Si un contexte de sécurité associé au SP existe déjà passer directement à l'étape 14.

(2) Le SP redirige le client vers le serveur WAYF. Trois paramètres sont associés à l'URL de redirection(Voir dans phase 3).

(3) Le client demande le service de WAYF

```
https://falcon.int-evry.fr/wayf/?
target=https://porphyre.int-evry.fr/picolibre&
shire=https://porphyre.int-evry.fr/shibboleth/SSO/POST&
providerId=https://porphyre.int-evry.fr/picolibre
```

Le service de WAYF traite la demande d'authentification et vérifie un cookie. Si l'utilisateur a déjà le cookie, sauter les étapes 4 et 5 sinon un formulaire en HTML est retourné au client.

(4) Le WAYF renvoie un formulaire en HTML au client pour choisir l'IdP préféré. Les paramètres de la demande d'authentification (montrés dans l'étape précédente) sont codés dans des champs cachés.

(5) L'utilisateur choisit un IdP à partir de la liste

(6) Le WAYF met à jour le cookie avec l'IdP préféré de l'utilisateur et redirige le client vers le service de SSO. Trois paramètres sont apposés à l'URL de la redirection (Voir dans phase 7).

(7) Le client effectue son authentification auprès de l'IdP

```
https://falcon.int-evry.fr/idp-picolibre?
target=https://porphyre.int-evry.fr/picolibre&
shire=https://porphyre.int-evry.fr/shibboleth/SSO/POST&
providerId=https://porphyre.int-evry.fr/picolibre
```

Le service de SSO traite la demande d'authentification et exécute un contrôle de sécurité. Si l'utilisateur n'a pas un contexte valide de sécurité alors l'IdP identifie le principal. Une fois que le principal a été identifié, le service de SSO obtient une déclaration d'authentification à partir de l'autorité d'authentification.

(8) Le service de SSO répond avec un document contenant un formulaire HTML :

```
<form method="post"
  action="https://porphyre.int-evry.fr/shibboleth/SSO/POST" ...>
  <input name="TARGET" type="hidden"
    value="https://porphyre.int-evry.fr/picolibre" />
  <input name="SAMLResponse" value="response" type="hidden" />
  ...
  <input type="submit" value="Submit" />
</form>
```

La valeur du paramètre TARGET a été préservée de l'étape 7. La valeur du paramètre de SAMLResponse est le codage base64 d'un élément signé `<samlp:Response>` tel que celui ci-dessous :

```
<samlp:Response
  xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol"
  MajorVersion="1" MinorVersion="1"
  IssueInstant="2004-12-05T09:22:02Z"
  Recipient="https://porphyre.int-evry.fr/shibboleth/SSO/POST"
  ResponseID="c7055387-af61-4fce-8b98-e2927324b306">
  <!-- insert ds:Signature element here -->
```

```
<samlp:Status><samlp:StatusCode Value="samlp:Success"/></samlp:Status>
<!-- insert SAML assertion here -->
</samlp:Response>
```

(9) Le client envoie une requête POST au service du consommateur d'assertions au SP. Pour automatiser la soumission de la forme, la ligne suivante du Javascript (ou son équivalent) peut apparaître dans le document HTML qui contient le formulaire

```
window.onload = function() { document.forms[0].submit(); }
```

(10) Le service du consommateur d'assertions analyse la demande de POST, valide la signature sur l'élément <samlp:Response>, crée un contexte de sécurité au SP et passe le contrôle au demandeur d'attributs pour lancer l'échange d'attributs. Le demandeur d'attributs envoie un message de SAML en forme de message de SOAP à l'autorité d'attributs

(AA) chez l'IdP :

```
POST /shibboleth/AA/SOAP HTTP/1.1
Host: falcon.int-evry.fr/idp-picolibre
Content-Type: text/xml
Content-Length: nnn
SOAPAction: http://www.oasis-open.org/committees/security
```

```
<?xml version="1.1" encoding="ISO-8859-1"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <samlp:Request
      xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"
      xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol"
      MajorVersion="1" MinorVersion="1"
      IssueInstant="2004-12-05T09:22:04Z"
      RequestID="aaf23196-1773-2113-474a-fe114412ab72">
      <samlp:AttributeQuery
        Resource="https://porphyre.int-evry.fr/shibboleth">
        <saml:Subject>
          <saml:NameIdentifier
            Format="urn:mace:shibboleth:1.0:nameIdentifier"
            NameQualifier="https://falcon.int-evry.fr/idp-picolibre"
            3f7b3dcf-1674-4ecd-92c8-1544f346baf8
          </saml:NameIdentifier>
        </saml:Subject>
        <saml:AttributeDesignator
          AttributeName="urn:mace:dir:attribute-def:eduPersonPrincipalName"
          AttributeNamespace="urn:mace:shibboleth:1.0:attributeNamespace:uri"/>
        </samlp:AttributeQuery>
      </samlp:Request>
    </SOAP-ENV:Body>
  </SOAP-ENV:Envelope>
```

(11) L'autorité d'attribut chez l'IdP traite la demande et renvoie les attributs exigés au demandeur d'attributs :

```
HTTP/1.1 200 OK
```

```
Content-Type: text/xml
Content-Length: nnnn
```

```
<?xml version="1.1" encoding="ISO-8859-1"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <samlp:Response
      xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol"
      InResponseTo="aaf23196-1773-2113-474a-fe114412ab72"
      IssueInstant="2004-12-05T09:22:05Z"
      MajorVersion="1" MinorVersion="1"
      ResponseID="b07b804c-7c29-ea16-7300-4f3d6f7928ac">
      <samlp:Status>
        <samlp:StatusCode Value="samlp:Success"/>
      </samlp:Status>
      <!-- insert SAML assertion here -->
    </samlp:Response>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

(12) Le service du consommateur d'assertions filtre les attributs, met à jour le contexte de sécurité et redirige le client vers la ressource cible.

(13) Le client demande la ressource de cible au SP (encore) :

```
https://porphyre.int-evry.fr/picolibre
```

(14) Puisqu'un contexte de sécurité existe, le SP renvoie la ressource au client.

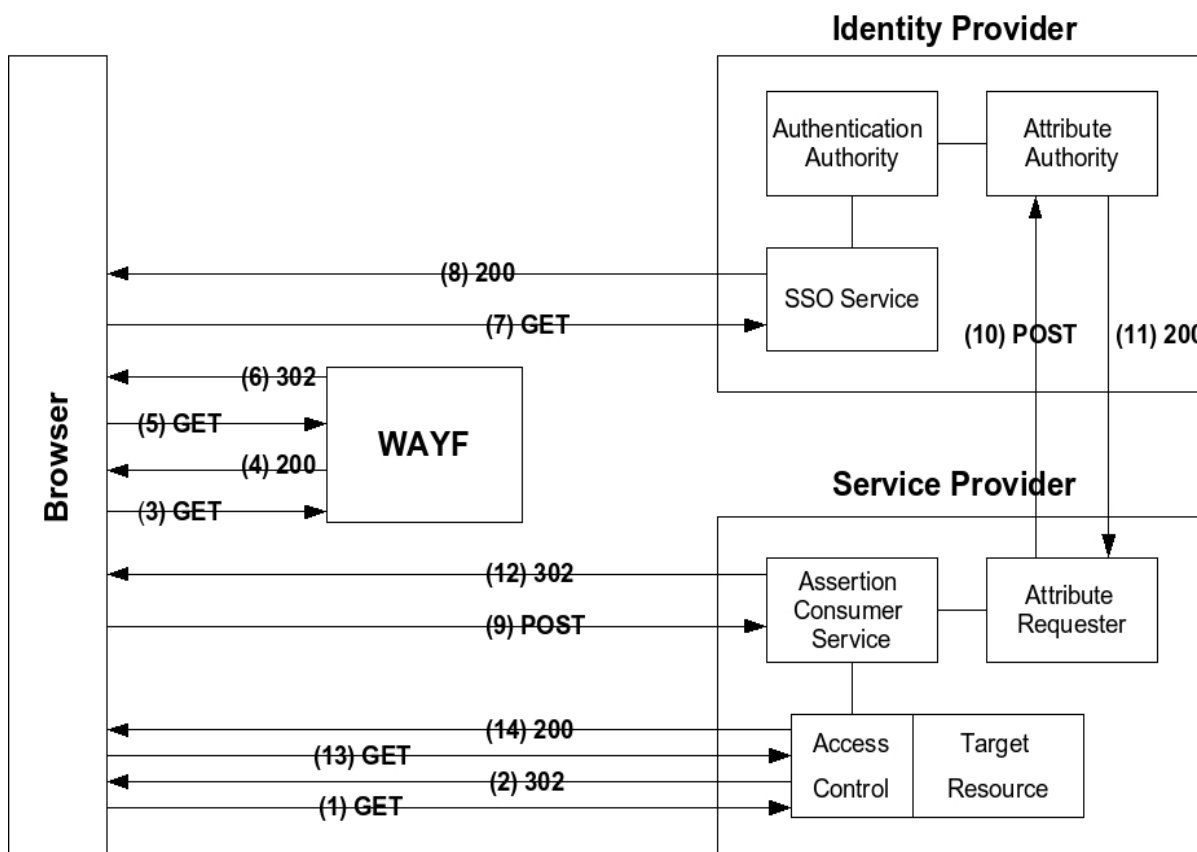


Figure 12: Fonctionnement de Shibboleth

### 3.4 Fédération de Shibboleth

#### 3.4.1 Méta données

L'IdP et le SP dans une fédération publient des informations sur eux-mêmes dans les dossiers spéciaux de XML appelés les dossiers de méta-données. Le membre fournit les informations requises comme:

- l'identifiant de service en forme d'un URN,
- l'intitulé du service,
- et le contact technique pour le service

Chaque fournisseur d'identité est décrit par trois éléments propres:

<md:IDPSSODescriptor>: le service d'authentification de fournisseur d'identité

<md:AuthnAuthorityDescriptor>: le service autorité d'authentification de fournisseur d'identité

<md:AttributeAuthorityDescriptor>: le service autorité d'attributs de

fournisseur d'identité

Chaque fournisseur de service est décrit par un élément propre `<md:SPSS0Descriptor>` associé au service consommateur d'assertions du fournisseur de service.

Les méta données contiennent aussi une liste des autorités de certification de confiance. les méta données sont gérées par la fédération et partagées par tous les membres qui doivent les synchroniser. Chaque membre peut définir avec quels partenaires il travaille dans ses méta données

### **3.4.2 Relation de confiance entre les membres d'une fédération**

Dans une fédération il faut avoir des relations de confiance entre les membres. Le SP repose sur les IdP pour vérifier une authentification sûre de ses utilisateurs. Il fait confiance dans les attributs d'utilisateur que les IdP propagent. Par contre l'IdP délivre des attributs sur ses utilisateurs aux SP alors il leur fait aussi confiance. Pour une fédération il doit y avoir un acteur définissant des engagements et centralisant leur gestion. Ceci permet d'éviter la multiplication des relations entre les différents fournisseurs dans la fédération. Cet acteur donne aussi les services centraux: tels que le service WAYF, la distribution des méta données.

Le fournisseur d'identité utilise une ARP( Attribute Release Policy) qui contient un ensemble de règles. Chaque règle définit un contexte d'application et des attributs avec des valeurs autorisées pour chaque attribut. Une ARP peut être définie pour un fournisseur de service à fin de filtrage des données.

Un mécanisme équivalent aux ARP est défini dans un fournisseur de service, permettant de filtrer les attributs reçus. Il est AAP (Attribute Acceptance Policy).

## Chapitre 4 .Réalisation

### 4.1 Amélioration d'authentification de PicoLibre

Actuellement, les utilisateurs de PicoLibre peuvent s'authentifier par une phase d'authentification de phpGroupWare qui se base sur une base de données relationnelle (mysql etc) ou annuaire de LDAP, qui fournit la base pour la gestion d'utilisateurs dans PicoLibre. Un utilisateur s'authentifie à phpGroupWare quand il veut accéder à la page d'accueil «bureau virtuel » de PicoLibre, qui contient la liste des projets auxquels il collabore.

Un utilisateur peut également s'authentifier directement auprès d'un des autres composants intégrés dans la plate forme,, tel que Sympa ou d'autres à venir (comme TWiki), sans passer par la page login de phpGroupWare.

#### 4.1.1 Schéma d'authentification standard dans phpGroupWare

PhpGroupWare gère des permissions d'accès à ses applications, pour des utilisateurs connus localement, en se basant sur un compte local, stocké dans un annuaire mis en oeuvre par exemple dans une base de données relationnelle MySQL, ou un annuaire LDAP.

Pour pouvoir accéder à phpGroupWare, l'utilisateur doit avoir un compte qui détermine le profil de l'utilisateur : les droits d'accès pour une liste d'applications que l'utilisateur peut utiliser. Le profil d'utilisateur est créé :

- soit par l'administrateur de phpGroupWare
- soit, par défaut, en mode auto-create qui est déclaré dans la phase de la configuration phpGroupWare dès que l'on installe phpGroupWare.

L'administrateur peut ainsi modifier le profil d'utilisateur, selon le rôle d'un utilisateur défini dans le schéma d'organisation d'un établissement.

Maintenant selon des types de campus stockant des comptes phpGroupWare supporte des types d'authentifications: SQL, LDAP, mail... Ce sont des méthodes d'authentification normale qui sont implémentées dans l'application et ne peuvent pas partager la session d'autre application.

Le processus d'accès aux applications phpGroupWare se compose de trois

phases comme le montre la figure13

1. Authentification : vérifier que l'utilisateur est bien la propriétaire d'un compte, en utilisant un objet de la classe **auth** qui recherche le compte et valide le mot de passe saisi dans l'annuaire d'authentification local.
2. Détermination du profil d'utilisateur : cette phase est réalisée par un objet de la classe **account** pour obtenir le profil d'utilisateur, après qu'il ait été authentifié avec succès. Si phpGroupWare est en mode "auto-create" et qu'aucun profil n'existe dans l'annuaire local, un nouveau profil peut être automatiquement créé par défaut.
3. Construction d'environnement de travail : en se basant sur le profil d'utilisateur, phpGroupWare va créer une session de travail et fournir l'accès à des applications que l'utilisateur a le droit d'utiliser.

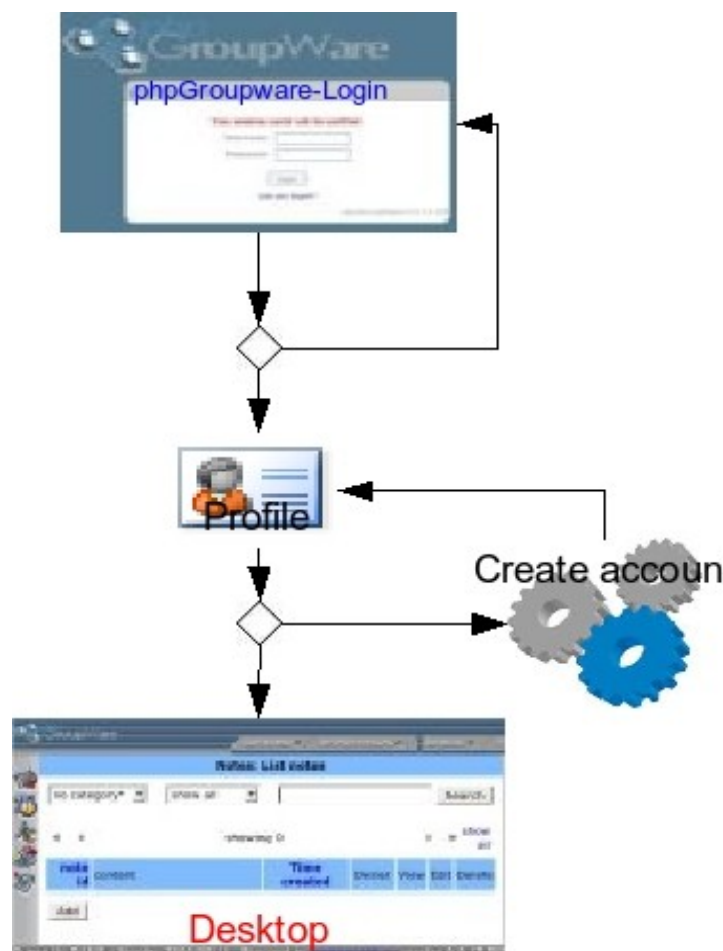


Figure 13: Schéma standard d'authentification

### **4.1.2 Shibboleth et Apache pour service de SSO**

Une autre manière d'assembler tous les mécanismes courants d'authentification dans les diverses applications Web installées sur le même système de PicoLibre, est l'utilisation du système de l'authentification de serveur Web (par exemple le `mod_auth_ldap`, `mod_auth_mysql...` dans Apache). Dans le cadre de travail on suppose que le serveur Apache est standard.

Mais si (à l'avenir) les applications composant la plate forme PicoLibre sont déployées réellement sur plusieurs serveurs Apache, on a besoin d'un mécanisme plus avancé pour partager cette authentification. On a besoin toujours également d'un service SSO avec d'autres applications déployées partout dans le système d'information, en dehors de la plate forme PicoLibre, à laquelle les utilisateurs auront déjà été authentifié.

Shibboleth peut aider pour résoudre tous ces besoins. Aujourd'hui beaucoup d'applications Web que PicoLibre intègre, comme Sympa, ou Twiki deviennent compatible avec Shibboleth. Alors Apache et Shibboleth pourront être un intégrateur des mécanismes d'authentifications.

Mais malheureusement, les mécanismes de l'authentification de phpGroupWare ne satisfont pas l'utilisation d'authentification via serveur Apache et Shibboleth. Il est nécessaire de développer un nouvel adaptateur d'authentification de phpGroupWare pour la combinaison de serveur Apache + de Shibboleth.

### **4.1.3 Problèmes dans environnement mélangé et legs**

PicoLibre peut utiliser la fédération de Shibboleth une fois que des adaptateurs ont été ajoutés à toutes ses applications. Mais Shibboleth ne peut pas être le mécanisme d'authentification exclusif utilisé. PicoLibre n'est pas dans un système typique de «Intranet» ou de «extranet», et il a des utilisateurs internes et externes. Il a besoin alors d'une certaine manière de dévier Shibboleth pour certains de ses utilisateurs.

Alors, un problème qui doit être résolu quand Shibboleth (ou un tel mécanisme externe d'authentification) est utilisé, est la réalisation d'un mapping entre l'utilisateur de Shibboleth et le compte préexistant dans l'application (par exemple le compte dans phpGroupWare).

Naturellement, si Shibboleth est déployée avant d'installer la plate forme PicoLibre, et tous les utilisateurs connus dans Shibboleth, et seulement

des utilisateurs de Shibboleth sont identifiés par les applications, alors un tel mapping est trivial. Mais il devient beaucoup plus difficile si Shibboleth est déployée sur un environnement existant avec beaucoup de comptes existant déjà dans PicoLibre.

Après avoir considéré les contraintes ci-dessus, on propose d'intégrer PicoLibre (par conséquent phpGroupWare) et Shibboleth avec une approche souple. En particulier, on essaye de faciliter l'intégration progressive des instances déployées de phpGroupWare, ceci afin de diminuer le fardeau de migration pour les administrateurs et les utilisateurs (recréation de comptes, etc.).

En conséquence, la conception des nouveaux mécanismes d'authentifications de PicoLibre devra alors supporter les cas suivants:

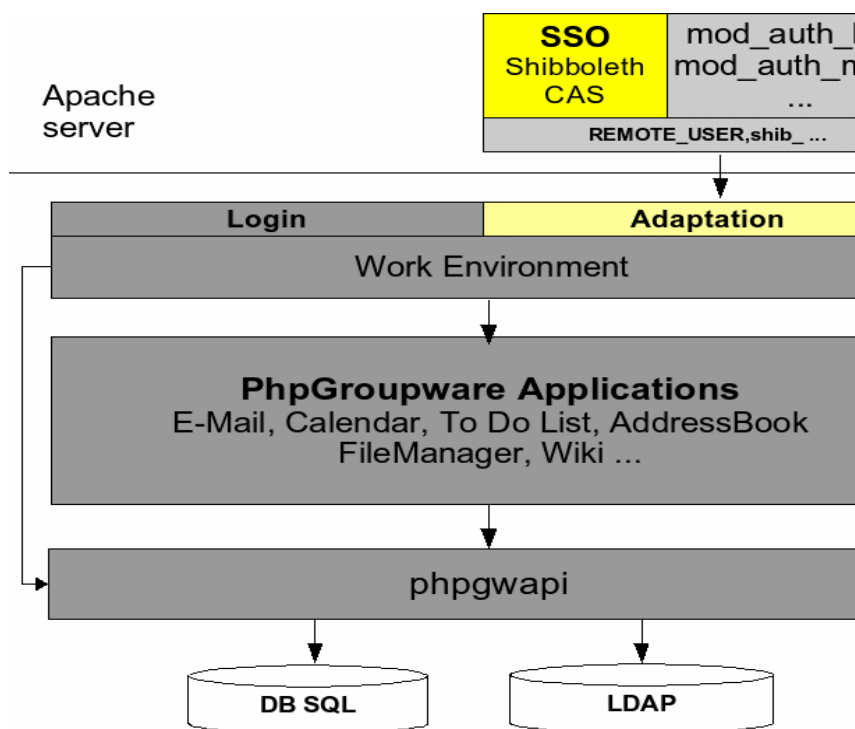
1. les nouveaux utilisateurs, qui sont connus dedans la fédération mais n'ont pas encore un compte dans PicoLibre, doivent pouvoir créer un nouveau compte dans PicoLibre/phpGroupWare.
2. les utilisateurs de PicoLibre avec un compte dans le phpGroupWare, qui accèdent à PicoLibre, et :
  - s'ils ont également un compte dans la fédération de Shibboleth alors ils peuvent faire un mapping de l'identité de Shibboleth vers leur compte de PicoLibre.
  - sinon (pour des personnes externes de la fédération), Ils peuvent accéder à PicoLibre en déviant le service SSO de Shibboleth.
3. les nouveaux utilisateurs externes de la fédération, qui peuvent demander l'enregistrement dans phpGroupWare.

La situation devrait être semblable pour n'importe quel mécanisme d'authentification externe comme Shibboleth. Alors on a essayé de proposer un mécanisme de mapping standard dans phpGroupWare qui peut être utilisé non seulement par Shibboleth mais par tout autre mécanisme d'authentification.

#### ***4.2 Implémentation de l'adapteur dans phpGroupWare pour Shibboleth***

Dans la phase d'implémentation un adapteur a été développé dans phpGroupWare qui permet à phpGroupWare de participer à la fédération de Shibboleth. Cet adapteur réalise la phase d'authentification via Apache

en vérifiant la variable `REMOTE_USER` et la phase mapping de `REMOTE_USER` vers un compte dans phpGroupWare. La figure 14 montre la nouvelle architecture de phpGroupWare avec cet adaptateur.



### 4.2.1 Utiliser l'authentification via Apache dans le phpGroupWare

En utilisant la méthode d'authentification via Apache, phpGroupWare n'authentifie pas des utilisateurs intérieurement, dans son annuaire de comptes (LDAP, MySQL, ...). Au lieu de cela, il dépendra de la variable d'environnement `REMOTE_USER` de la session de Apache (qui contient quelque chose comme l'identifiant ou email d'utilisateur), qui est défini quand la transaction de HTTP est authentifiée par le serveur Apache.

L'avantage de ce schéma est que si on a un schéma existant d'authentification de système d'information à l'aide des modules de Apache tels que le `mod_auth_ldap`, `mod_auth_mysql`, `mod_auth_cas`, `mod_auth_shib`, etc que on peut juste brancher directement à eux. phpGroupWare profite alors de ce mécanisme d'intégration avec des sources extérieures d'authentification comme Shibboleth.

Cependant, selon les dispositifs du navigateur et la mécanisme spécifique

utilisée, dans certains cas, l'identité de l'utilisateur peut être cachée dans l'état interne du navigateur Web: il peut lancer une session, mais ne pas pouvoir se déconnecter du phpGroupWare à moins que le navigateur ne soit fermé.

La configuration du serveur Apache pour phpGroupWare peut être comme (pour `mod_auth_ldap` contre un serveur de LDAP) :

```
<location /phpgroupware>
    AuthType Basic
    AuthName "phpGroupware"
    AuthLDAPEnabled on
    AuthLDAPAuthoritative on
    AuthLDAPURL ldap://my.openldap-server.com/dc=my_org,dc=org?uid
    require valid-user
</Location>
```

#### **4.2.2 Mapping REMOTE\_USER vers le compte de phpGroupWare**

Avec Shibboleth phpGroupWare est un membre dans la fédération d'identité qui consiste en plusieurs sources d'authentification. Un utilisateur est authentifié par le serveur Apache (via `mod_shib` dans Apache) phpGroupWare doit déterminer son profil en recherchant son compte dans phpGroupWare en se basant sur les attributs fournis par Shibboleth. Le choix des attributs à utiliser est difficile. Par défaut REMOTE\_USER est un identifiant d'utilisateur.

Par exemple, deux utilisateurs différents venant de deux IdP différents peuvent avoir un même l'identité (uid). Si on utilise un mapping trivial de REMOTE\_USER vers l'identifiant du compte alors il ne marche pas. Il faut utiliser d'autre valeur pour REMOTE\_USER (par exemple mail, numéro d'identité, numéro de passeport) qui est censé être une valeur unique parmi tous les IdP de la fédération d'identité.

Mais supposons que le choix d'attribut pour REMOTE\_USER est email, une personne peut appartenir à deux IdPs différentes où elle est connue avec deux email différents : par exemple, un email de département et un email d'entreprise. Pour supporter mieux ces situations, Il est nécessaire de développer un nouveau mécanisme de mapping, actif pendant la phase d'identification dans phpGroupWare, qui se base sur une table de mapping pour la projection de ces différentes valeurs d'attribut à une identification de compte dans phpGroupWare.

Il peut y avoir deux modes de fonctionnement disponibles dans

phpGroupWare, configuré par l'administrateur.

- Mapping trivial, dans le cas où REMOTE\_USER est une identification unique dans la fédération des identités.
- Mapping par table, dans le cas où chaque utilisateur n'a pas nécessairement une identification unique qui peut être identifiée dans le système de l'information de l'entreprise.

### **4.2.3 Additions à phpGroupWare**

Le code de phpGroupWare est amélioré avec la classe **auth\_remoteuser** et des classes **mapping** (qui sont décrits plus en détail dans l'annexe B). Après avoir passé la phase d'authentification de Shibboleth le script **login.php** réalisera la phase de mapping et créera la session de travail dans phpGroupWare.

Si la phase de mapping n'est pas succès:

- un nouveau compte peut être créé si le phpGroupware permet à des utilisateurs de créer les comptes (comme défini dans la configuration des phpGroupware : « Autocreation de compte »), le script **create\_account.php** fournit une interface pour créer le nouveau compte, basée sur les informations fournies par Shibboleth. Il aidera à rechercher les champs de description de l'utilisateur comme les noms, email, identifications, etc.
- ou un nouveau mapping peut être créé vers un compte existant dans phpGroupWare si le phpGroupware est configuré pour soutenir le mapping par table. Le script **create\_mapping.php** fournit la fonction pour créer un nouveau mapping si l'utilisateur a déjà un compte dans le phpGroupware. Pendant ce processus, l'utilisateur devra s'authentifier relativement au compte existant dans phpGroupware, avant de créer ce nouveau mapping afin que le système s'assure qu'il est bien le propriétaire du compte existant.

Si phpGroupware est configuré pour le mécanisme de mapping « trivial », il faut faire confiance entièrement à l'IdPs, et considérer que l'association des attributs de Shibboleth aux comptes locaux est non-ambigu.

Dans le cas où le mapping trivial est un succès pour la plupart des utilisateurs, mais qu'il échoue quand même pour un nombre restreint de cas, un mapping « séquentiel » pourrait être activé. Les deux

mécanismes seraient alors appliqués séquentiellement : mapping trivial d'abord, et puis mapping par table.

La mécanisme de mapping est utilisé non seulement pour Shibboleth mais encore pour autre mécanisme d'authentification externe. Alors l'adaptateur peut devenir utile également pour d'autres sources d'authentification par Apache, ainsi il est intégré dans la base de code standard du module APIs de phpGroupWare.

#### **4.2.4 Configuration d'accès à phpGroupWare via Apache**

L'accès aux pages de phpGroupWare et aux applications Web associées de PicoLibre peut être configuré par des directives du serveur Apache.

Comme expliqué plus haut, on peut utiliser phpGroupWare dans un environnement Intranet, ou un environnement mélangé tel que celui pour PicoLibre. Dans un environnement mélangé comme PicoLibre, où on utilise l'authentification par le serveur Apache et Shibboleth, Shibboleth identifie seulement des membres dans la fédération, et non des utilisateurs externes .

Par des directives spécifiques de configuration dans le serveur Apache, on peut concevoir différents points d'accès pour la même application Web, et choisir le contrôle d'accès par Apache obligatoire (appelé full-apache), ou facultatif (appelé semi-apache).

##### **4.2.4.1 Contrôle d'accès en mode full-apache**

Dans ce cas, Apache est comme un «emballage» autour de l'application entière de phpGroupWare, protégeant tous ses accès. Chaque accès à phpGroupWare doit passer par le module d'authentification de Apache (mod\_shib de Shibboleth). On accordera l'accès qu'aux utilisateurs déjà connus de la fédération d'identité . Ceci peut être réalisé par une configuration de serveur de Apache comme ce qui suit :

```
<Location /phpgroupware>
...[any auth. mechanism here]
require valid-user
</Location>
```

Quand un utilisateur est authentifié, il procédera à la phase mapping, qui aidera de rechercher son compte local existant de phpGroupWare. La figure15 montre le processus login en mode de full-apache.

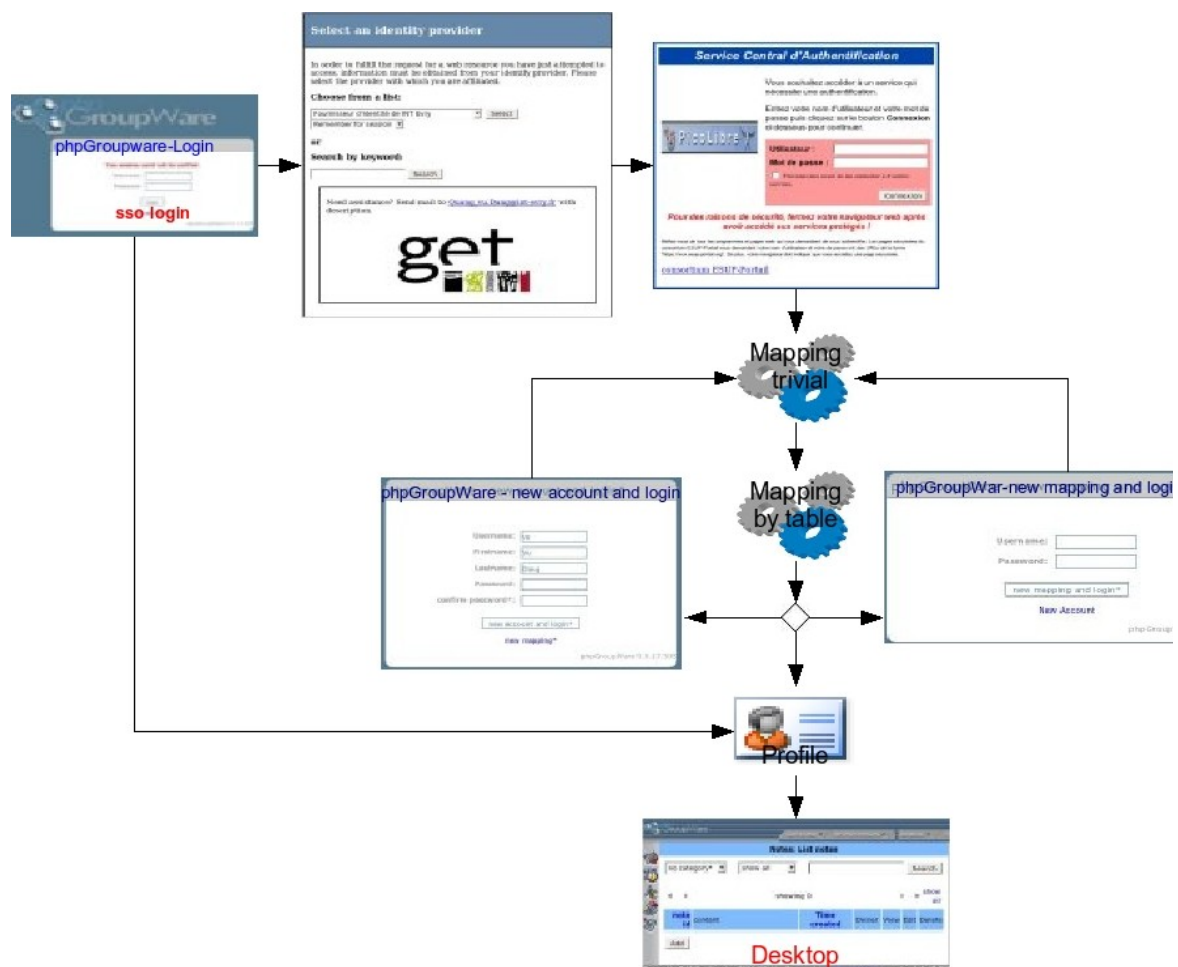


**/phpgroupware/phpgwapi/inc/sso**

```
<Location /phpgroupware/phpgwapi/inc/sso>  
  ...[any auth. mechanism here]  
  require valid-user  
</Location>
```

Ainsi pour accéder à phpGroupWare, l'utilisateur serait libre de choisir n'importe quelle méthode qui lui convient, en allant à différents points d'accès (URLs différent), par exemple :

- **http://server/phpgroupware/login.php** : ce qui va proposer le dialogue standard login + de mot de passe de phpGroupWare. Ce dialogue est utilisé par les utilisateurs locaux, et fournirait également un lien alternatif (appelé «SSO-login», par exemple)
- et  
**http://server/phpgroupware/phpgwapi/inc/sso/login\_server.php**  
par exemple : ce qui serait accédé du lien de SSO ci-dessus, ou directement, dont le but unique serait de diriger vers d'infrastructure d'authentification de Shibboleth et des pages de SSO.



La figure16 montre le processus login en mode « semi-apache ».

#### **4.2.4.3 Configuration phpGroupWare et Shibboleth pour PicoLibre**

Avec l'addition de la phase de mapping nécessaire, phpGroupware peut maintenant utiliser les services de SSO disponibles dans l'infrastructure de Shibboleth. Les plateformes de PicoLibre participeront alors au système d'information d'une manière standard pour la plupart de ses utilisateurs, accédant à la fédération d'identité pour les identifier.

Pour les besoins spécifiques de plate forme PicoLibre, phpGroupware sera configuré pour utiliser l'authentification en mode «semi-apache » décrite ci-dessus, puisque on a des utilisateurs externes.

Dans la configuration de Apache on utilisera l'authentification de Shibboleth:

```
<Location /phpgroupware/phpgwapi/inc/sso>
    AuthType shibboleth
```

```
    ShibRequireSession On
    require valid-user
</Location>
```

Comme expliqué ci-dessus, l'utilisateur connu de Shibboleth peut être connu de plusieurs de sources d'identité (IdP), et on doit définir quel attribut sera utilisé pour identifier l'utilisateur, et définir la stratégie de mapping des comptes de phpgroupware en conséquence.

On a utilisé l'email pour identifier l'utilisateur et le fichier de configuration **AAP.xml** du fournisseur de service associé à phpGroupware sera défini afin de transmettre la valeur de l'email de l'utilisateur pour REMOTE\_USER :

```
<AttributeRule name="urn:mace:dir:attribute-def:mail" Header="REMOTE_USER">
  <AnySite>
    <AnyValue />
  </AnySite>
</AttributeRule>
```

Le même genre de configuration peut alors être appliqué pour l'accès à d'autres applications intégrées de la plate forme PicoLibre telles que Sympa (ou TWiki à l'avenir) pour qu'ils bénéficient également d'authentification de phpGroupWare ou de l'authentification externe d'applications pour SSO.

## **Conclusions**

Dans le cadre du stage on a proposé une méthode pour intégrer phpGroupWare et Shibboleth pour permettre l'utilisation de mécanismes de SSO dans phpGroupWare et de supporter des sources d'authentification multiples en utilisant la fédération d'identité.

L'intégration se base sur l'utilisation des modules d'authentification Apache au lieu de passer par les mécanismes d'authentification interne de phpGroupWare et sur un mécanisme de mapping qui peut être utilisé par les mécanismes d'authentification externe. Un adaptateur a été développé dans la nouvelle version 0.9.18 de phpGroupWare. Il y a aussi un adaptateur pour la version courante de PicoLibre se basant sur phpGroupWare 0.9.16 pour tester dans la fédération d'identité de GET/INT.

Nous avons proposé plusieurs options pour la configuration et l'adaptation à d'autres environnements, afin de réaliser la solution la plus générique. Ils aident à s'adapter à plusieurs genres de situations, comme la disponibilité d'un environnement complètement opérationnel de Shibboleth ou un environnement existant avec beaucoup de comptes existant déjà dans PicoLibre.

Par des directives spécifiques de configuration dans le serveur Apache, on peut concevoir différents points d'accès pour la même application Web, et choisir le contrôle d'accès par Apache obligatoire (appelé full-apache), ou facultatif (appelé semi-apache).

Dans un contexte d'utilisation de Shibboleth pour PicoLibre, on a ajouté l'application picolibre\_twiki pour intégrer Twiki dans PicoLibre(en détail dans l'annexe C).

## **Bibliographie**

- [1] Site du projet PicoLibre, <http://www.picolibre.org/>.
- [2] Site du projet phpGroupWare, <http://www.phpgroupware.org/>.
- [3] Site du projet Shibboleth, <http://shibboleth.internet2.edu/>.
- [4] Berger O., C. Bac, and B. Hamet, 2006, *Integration of Libre Software Applications to Create a Collaborative Work Platform for Researchers at GET*, International Journal of Information Technology and Web Engineering 1 (3), 2006.
- [5] C. Bac, Berger O., B. Hamet, *ProGET : Plate-forme de travail collaboratif destinée aux enseignants/chercheurs du GET*.
- [6] Olivier Salaün, Florent Guilleux, Pascal Aubry, 2005, *Fédération d'identités et propagation d'attributs avec Shibboleth*, JRES2005.
- [7] Site du projet Switch, <http://www.switch.ch/aai/>.
- [8] Site de la fédération de CRU, <http://federation.cru.fr/>.
- [9] Vincent Mathieu, Pascal Aubry, Julien Marchal, 2003, *Single Sign-On open-source avec CAS (Central Authentication Service)*, JRES2003.
- [10] Olivier Salaün, 2003, *Introduction aux architectures web de Single Sign-on*, JRES2003.
- [11] Site du projet Twiki, <http://twiki.org/>.
- [12] Site du projet Sympa, <http://www.sympa.org/>.
- [13] Site du projet de serveur Apache, <http://httpd.apache.org/>.

## Annexe A: Assertions de SAML

### Assertions d'authentification

```
<saml:Assertion
  xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"
  MajorVersion="1" MinorVersion="1"
  AssertionID="a75adf55-01d7-40cc-929f-dbd8372ebdfc"
  IssueInstant="2006-07-05T09:22:02Z"
  Issuer="https://falcon.int-evry.fr/idp-picolibre">
  <saml:Conditions
    NotBefore="2006-07-05T09:17:02Z"
    NotOnOrAfter="2006-07-05T09:27:02Z">
    <saml:AudienceRestrictionCondition>
      <saml:Audience>http://porphyre.int-evry.fr/shibboleth</saml:Audience>
    </saml:AudienceRestrictionCondition>
  </saml:Conditions>
  <saml:AuthenticationStatement
    AuthenticationInstant="2006-07-05T09:22:00Z"
    AuthenticationMethod="urn:oasis:names:tc:SAML:1.0:am:password">
    <saml:Subject>
      <saml:NameIdentifier
        Format="urn:mace:shibboleth:1.0:nameIdentifier"
        NameQualifier="https://falcon.int-evry.fr/idp-picolibre">
        3f7b3dcf-1674-4ecd-92c8-1544f346baf8
      </saml:NameIdentifier>
      <saml:SubjectConfirmation>
        <saml:ConfirmationMethod>
          urn:oasis:names:tc:SAML:1.0:cm:bearer
        </saml:ConfirmationMethod>
      </saml:SubjectConfirmation>
    </saml:Subject>
  </saml:AuthenticationStatement>
</saml:Assertion>
```

### Assertions d'attribut

```
<saml:Assertion
  xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"
  MajorVersion="1" MinorVersion="1"
  AssertionID="a144e8f3-adad-594a-9649-924517abe933"
  IssueInstant="2006-07-05T09:22:05Z"
  Issuer="https://falcon.int-evry.fr/idp-picolibre">
  <saml:Conditions
    NotBefore="2006-07-05T09:17:05Z"
    NotOnOrAfter="2006-07-05T09:52:05Z">
    <saml:AudienceRestrictionCondition>
      <saml:Audience>http://porphyre.int-evry.fr/shibboleth</saml:Audience>
    </saml:AudienceRestrictionCondition>
  </saml:Conditions>
  <saml:AttributeStatement>
    <saml:Subject>
      <saml:NameIdentifier
        Format="urn:mace:shibboleth:1.0:nameIdentifier"
```

```
    NameQualifier="https://falcon.int-evry.fr/idp-picolibre">
      3f7b3dcf-1674-4ecd-92c8-1544f346baf8
    </saml:NameIdentifier>
  </saml:Subject>
  <saml:Attribute
    AttributeName="urn:mace:dir:attribute-def:mail"
    AttributeNamespace="urn:mace:shibboleth:1.0:attributeNamespace:uri">
    <saml:AttributeValue>
      quang_vu.dang@int-evry.fr
    </saml:AttributeValue>
  </saml:Attribute>
</saml:AttributeStatement>
</saml:Assertion>
```

## Assertions signées

```
<ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
  <ds:SignedInfo>
    <ds:CanonicalizationMethod
      Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
    <ds:SignatureMethod
      Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
    <ds:Reference URI="#c7055387-af61-4fce-8b98-e2927324b306">
      <ds:Transforms>
        <ds:Transform
          Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
signature" />
        <ds:Transform
          Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#">
          <InclusiveNamespaces
            PrefixList="#default saml samlp ds xsd xsi"
            xmlns="http://www.w3.org/2001/10/xml-exc-c14n#" />
          </ds:Transform>
        </ds:Transforms>
        <ds:DigestMethod
          Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
        <ds:DigestValue>TCDVSuG6grhyHbzhQFWFzGrxIPE=</ds:DigestValue>
      </ds:Reference>
    </ds:SignedInfo>
    <ds:SignatureValue>
      x/GyPbzmFEe85pGD3c1aXG4Vspb9V9jGCjwcRCKrtwPS6vdVNCcY5rHaFPYWkf+5
      EIYcPzx+pX1h43SmwviCqXRjRtMANWbHLhWAptaK1ywS7gFgsD01qjyen3CP+m3D
      w6vKhaqlledl0BYyrIzb4KkH04ahNyBVXbJwqv5pUaE4=
    </ds:SignatureValue>
    <ds:KeyInfo>
      <ds:X509Data>
        <ds:X509Certificate>
          <!-- insert Certificate here -->
        </ds:X509Certificate>
      </ds:X509Data>
    </ds:KeyInfo>
  </ds:Signature>
```

## Annexe B: Additions au phpGroupWare

### La paquet phpgwapi

Dans ce paquet on ajoute la classe **auth** dans le fichier *auth\_remoteuser.inc.php* qui correspond à la méthode d'authentification par Apache. Cette classe ne fait rien, il surcharge seulement la classe basse dans le processus l'authentification dans le phpGroupware.

Pour la phase de mapping on ajoute la classe **mapping** dans le fichier *mapping.inc.php*. Il hérite de la classe **mapping\_** qui est basé sur le compte existant dans le phpGroupware, pour faire le mapping trivial et pour valider le compte.

liste de classe:

```
class auth : auth_apache.inc.php
class mapping_ : mapping_ldap.inc.php
class mapping_ : mapping_sql.inc.php
class mapping_ : mapping_picolibre.inc.php
class mapping : mapping.inc.php
```

### Nouveau module **sso** dans phpGroupWare

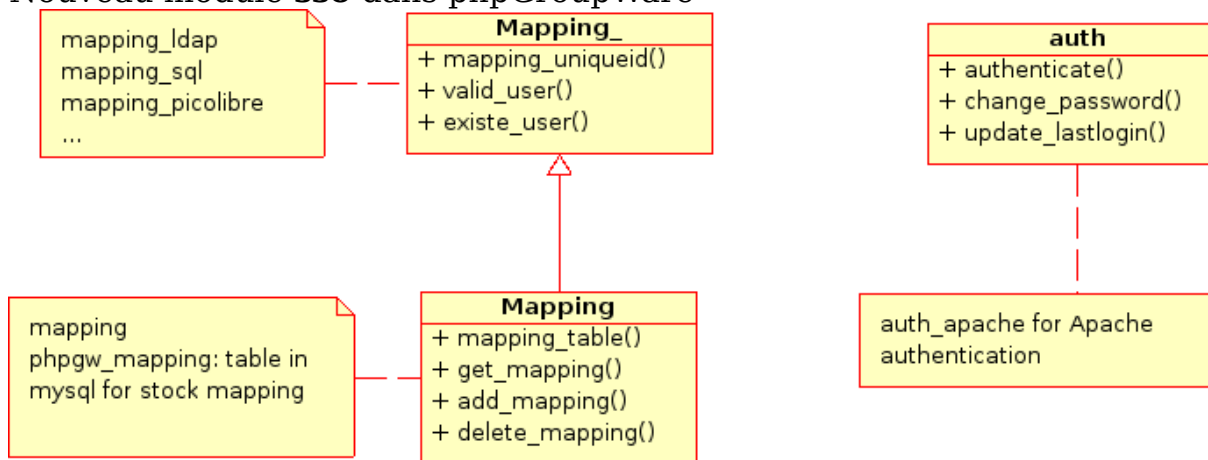


Figure 17: Diagramme des classe d'adapteur pour Shibboleth

### Nouveau module **sso** dans phpGroupWare

On ajoute un nouveau paquet sso pour l'authentification via Apache. Il se compose des scripts:

- */phpgwapi/inc/sso/login\_server.php* : pour le processus login
- */phpgwapi/inc/sso/create\_account.php* : interface permettant l'utilisateur de créer un nouveau compte
- */phpgwapi/inc/sso/create\_mapping.php*: iinterface permettant l'utilisateur de créer un nouveau mapping

- */preferences/mapping.php*: gère des mapping (Allow, Deny, Delete a mapping)

## **Nouvelle table dans base de données**

On ajoute une table *phpgw\_mapping* (attributs : *user\_ext*, *location*, *auth\_type*, *status*, *account\_lid*) pour faire mapping par table. Il lie des valeurs de REMOTE\_USER aux comptes locaux de phpGroupware d'utilisateur.

## **Annexe C: Intégration de Twiki dans PicoLibre**

### **picolibre\_twiki**

picolibre\_twiki est la nouvelle application phpGroupware, dans PicoLibre, qui gère les Webs TWiki pour les projets dans PicoLibre. Quand un projet est créé dans PicoLibre, cet application va créer un "Web" parent (nommé comme le projet) et trois sous-Webs optionnels (*Public*, *Private* et *Web*) pour ce projet. La contrôle d'accès sur les Web de TWiki se base alors sur le LDAP de PicoLibre. picolibre\_twiki permet ainsi à l'administrateur de projet de créer ou supprimer les Web dans son projet.

### **Implémentation dans PicoLibre**

L'application picolibre\_twiki se compose des fichiers/classes suivants:

picolibre\_twiki/index.php

picolibre\_twiki/inc/admin.inc.php

picolibre\_twiki/inc/class.picolibre\_twiki\_app\_rights.inc.php

picolibre\_twiki/inc/class.twiki\_bo.inc.php

picolibre\_twiki/inc/class.twiki\_so.inc.php

picolibre\_twiki/inc/class.twiki\_ui.inc.php

picolibre/twiki\_scripts/create\_twiki\_web.sh : Met à jour le fichier WebPreferences.txt des Webs de TWiki pour ajouter les droit d'accès dans TWiki, via le picolibre\_daemon

picolibre/twiki\_scripts/create\_twiki\_user.sh : Créer utilisateur TWiki quand un utilisateur est créé dans PicoLibre, via le picolibre\_daemon

Il y a un patch pour PicoLibre/phpgroupware qui touche des fichiers suivants :

/phpgroupware/new\_user.php

/phpgroupware/phpgwapi/inc/class.accounts\_picolibre.inc.php

/phpgroupware/picolibre\_current/inc/class.admin\_ui.inc.php

/phpgroupware/picolibre\_current/inc/class.current\_ui.inc.php

/phpgroupware/picolibre\_current/inc/class.project.inc.php

picolibre\_twiki demande des variables configuré dans Picolibre

- \$GLOBALS['phpgw\_info']['picolibre']['twiki\_host'] : l'adresse de TWiki
- \$GLOBALS['phpgw\_info']['picolibre']['twiki\_data'] : Répertoire data de TWiki pour modifier WebPreferences.txt
- \$GLOBALS['phpgw\_info']['picolibre']['twiki\_user\_admin'] et \$GLOBALS['phpgw\_info']['picolibre']['twiki\_passwd\_admin'] : login et mot de passe de l'administrateur de Picolibre (Le group Admins de Picolibre est ajouté dans le group TWikiAdminGroup de TWiki)

## **Implémentation dans TWiki**

Installer TWiki avec le plugin additionnel LdapContrib

Il y a trois Web templates additionnels dans TWiki utilisés par picolibre\_twiki:

**\_picolibre\_web** : Ajouter les droit ALLOWWEBCHANGE, ALLOWWEBRENAME pour Web de TWiki

**\_picolibre\_pri** : Ajouter les droit ALLOWWEBCHANGE, ALLOWWEBRENAME, ALLOWWEBVIEW pour Web de TWiki

**\_picolibre\_pub**:

**Main/PicolibreUserTemplate.txt** : Template utilisé pas script create\_twiki\_user.sh pour créer l'utilisateur

TWiki doit être configuré pour utiliser templatelogin, mais alors, pour envoyer un POST vers TWiki, via CURL depuis phpgroupware, il est nécessaire d'utiliser plutôt l'auth apache que via template. On configure donc un mode d'accès parallèle, pour picolibre\_twiki, qui utilise deux scripts CGI additionnels:

- picolibre\_manage : c'est un lien vers le script manage dans twiki/bin
- picolibre\_rename : c'est un lien vers le script rename dans twiki/bin

Dans le fichier .htaccess de TWiki on configure l'accès via apache/LDAP à ces scripts :

...

AuthType Basic

AuthLDAPEnabled on

AuthLDAPAuthoritative on

AuthLDAPURL ldap://localhost/dc=picolibre,dc=org?uid

...

# Si dans session de Apache a le variable REMOTE\_USER alors

TWiki ne demande pas d'authentification

```
<FilesMatch "(picolibre_manage|picolibre_rename)">
```

```
    require valid-user
```

```
</FilesMatch>
```

On préfère que les membres du groupe *Admins* de Picolibre soient les membres de `Main.TWikiAdminGroup` de TWiki, alors il faut ajouter le groupe *Admins* dans le liste des membre de `TWikiAdminGroup`.