# UNIVERSITÉ KASDI MERBAH OUARGLA

**Faculté des Nouvelles Techniques d'Informations et de la Communication**

**Département d'Informatique et de Technologies d'Information**


**Mémoire**

# MASTER PROFESSIONNEL

**Domaine :** Mathématiques et Informatique

**Filière :** Informatique

**Spécialité :** Réseaux Convergence et sécurité


**Présenté par :** BENGLIA Abla

BENHAMMOUDA Asma


## Thème

# Authentification SSO des services web du campus universitaire d'Ouargla

Soutenu publiquement le : juin 2015

A soutenir publiquement en Juin 2015, devant le jury composé de :

| Nom & Prénom | Qualité | Lieu |
|---|---|---|
| Mr.MAHDJOUB Med. Bachir | Président | Univ. Ouargla |
| Mr. BELHADJ Mourad | Examinateur | Univ. Ouargla |
| Mr. EUSCHI Salah | Rapporteur | Univ. Ouargla |
| Mr. KAFI Med. Radouane | Rapporteur | Univ. Ouargla |

Année Universitaire : 2014/2015

# *Remerciements*

*En premier lieu nous tenons à remercier Allah, pour nous avoir données la force à accomplir ce travail.*

*Nous tenons à remercier vivement* ***M. kafi Mohammed Radouane*** *et* ***M. Euschi Salah*** *notre encadreurs et qui a fourni des efforts, par leurs informations, leurs conseils et leurs encouragements.*

*Nous tenons à remercier tout* ***les ingénieurs informatiques*** *du rectorat de l'****Université Kasdi Merbeh Ouargla.***

*Nous tenons à remercier tout* ***les ingénieurs informatiques*** *du* ***Université du Ghardaïa.***

*Nous tenons à remercier tous les enseignants de* ***Département d'Informatique et de Technologies d'Information*** *surtout* ***les membres de jurys, M. Belhadj Mourad*** *et* ***Azzaoui Nadjat***

*Nos plus chaleureux remerciements pour tous ceux qui de prés et de loin ont contribué à réalisation de ce projet.*

# Dédicaces

*Ce travail est dédié :*

*A* ***mon cher père****, pour son soutien constant et inconditionnel, pour ses conseils et ses encouragements.*

*A la mémoire de* ***ma mère****, qui est toujours dans mon cœur.*

*A ma chère douzièmes mère* ***FATNA*** *pour ses encouragements, son soutien inconditionnel.*

*A la mémoire de ma chère* ***grande mère TIMA****, pour ses conseils toujours dans ma vie.*

*A mon chère frère* ***AZZEDDINE*** *pour ses conseils et ses encouragements.*

*À tous mes proches de la famille* ***BENHAMMOUDA****, et plus*

*Particulièrement,* ***mes sœur****,* ***mes frères****,* ***mes oncles*** *et* ***mes tantes*** *tout à son nom et sans oublier les familles :*

***BENCHABANA*** *et plus particulièrement* ***HAMZA.***

***BERGIGUA*** *et plus particulièrement* ***ABD -LBAKI, YOUCEF, SOUAD.***

***GACCEM*** *et plus particulièrement* ***MERIEM, ISMAHANE, ABD SLEM.***

*À tous mes chers amis :* ***Mouslima, Meriem et Rabab, Sabah, Khadra , Mabroka , Naziha, Hayet, Messaouda , Radia, Hadda , Linda, Toumia, Kahina, Nour, Ibbtissem, Soufiane , Abdelhakim ,Sabri,*** *et* ***mes collègues de l'Université de Ouargla , les étudiants de 2 ém année Réseaux Convergence et Sécurité promo 2015*** *et plus particulièrement ma binôme* ***ABLA .***

*Et à tous ce qui m'a enseigné tout au long de ma vie scolaire ;*

***ASMA***

# *Dédicaces*

*Grace à **Dieu** voilà notre travail terminé et il est temps pour moi de partager ma joie avec tous ceux qui m'ont soutenu et encouragé. Et à travers cette modeste thèse je tiens à présenter mes sincères dédicaces à **Mes Parents** qui ont consacré leurs vies à notre éducation et à faire notre bonheur et qui m'encouragent toujours pour achever mes études tout en espérant de voir les fruits de leurs sacrifices. A **mes chères sœurs** et **mes chers frères,** et toute **ma famille, à mes chers amis** ……..*

*A l'ensemble des amis que j'ai connu pendant mes études et à ceux qui m'ont prodigué leurs vifs conseils, à encouragements et témoigné de leur amitié, surtout le **groupe RCS**.*

*A ma binôme **Asma** et **sa famille.***

***ABLA***

# TABLES DES MATIERES

## LISTE DES FIGURES

## LISTE DES TABLES

# Abréviations

| | |
|---|---|
| **AAP** | **Attribute Acceptance Policy** |
| **ACL** | **Access Control List** |
| **ARP** | **Attribute Release Policy** |
| **ASP** | **Application Service Provider** |
| **CAS** | **Central Authentication Service** |
| **DN** | **Distinguished Name** |
| **DNS** | **Domain Name Service** |
| **FTP** | **File Transfer Protocol** |
| **http** | **Hypertext Transfer Protocol** |
| **HTTPS** | **Hypertext Transfer Protocol Secure** |
| **IDP** | **Identity Provider** |
| **IMAP** | **Internet Message Access Protocol** |
| **JSP** | **JavaServer Pages** |
| **LDAP** | **Lightweight Directory Access Protocol** |
| **NIS** | **Network Information Service** |
| **OASIS** | **Advancing Open Standards For The Information Society** |

| | |
|---|---|
| **PC** | **Personal Computer** |
| **PGT** | **Proxy Granting Ticket** |
| **HP** | **Hypertext Preprocessor** |
| **PT** | **Proxy Ticket** |
| **SAML** | **Security Assertion Markup Language** |
| **SASL** | **Simple Authentication And Security Layer** |
| **SOAP** | **Simple Object Access Protocol** |
| **SI** | **Système Information** |
| **SP** | **Service Provider** |
| **SQL** | **Structured Query Language** |
| **SSO** | **Single Sign On** |
| **SSL** | **Secure Sockets Layers** |
| **ST** | **Service Ticket** |
| **TCP/IP** | **Transmission Control Protocol / Internet Protocol** |
| **TGC** | **Ticket Granting Cookie** |
| **TLS** | **Transport Layer Security** |
| **URL** | **Uniform Resource Locator** |
| **URN** | **Uniform Resource Name** |

| | |
|---|---|
| **WAYF** | **Where Are You From** |
| **XML** | **Extensible Markup Language.** |

## *Résumé*

Notre travail consiste à réaliser une authentification SSO (Single Sign-On), pour les différentes services de l'université Kasdi Merbeh Ouargla (Zimbra, Joomla et Dokeos). Le concept choisi comme solution est de mettre en place un système d'authentification SSO de toutes les applications par mécanisme 'CAS' et fédérative d'identités 'Shibboleth', afin de résoudre le problème de savoir comment permettre aux utilisateurs de l'université (étudiants, enseignants) de s'authentifier une seule fois pour un accès global aux différents services tout en rassurant de leur identité. Pour le partage de ressources pédagogique au sein de la communauté universitaire, nous avons ajouté à notre solution le concept de fédération, en utilisant le système de Shibboleth. Ce système permet l'authentification inter-universités et assure un partage sécurisé des ressources pédagogiques à l'échèle national ou/et international


Mots clés: Authentification, SSO, Ldap, CAS, Shibboleth, Fournisseur d'identité.


---


## *Abstract*

The present research aims to realize an authentication SSO (Single Sign -On) for the various services (Zimbra , Joomla and Dokeos ) at university Kasdi Merbeh Ouargla. The concept has been chosen as a solution to allow the users of the university (students, teachers) to authenticate once for global access to the various services while reassuring their identity by implementing an SSO authentication system for all the applications by mechanism 'CAS' and federal identity 'Shibboleth'. Moreover, in order to share the educational resource within the academic community, the concept of federation using Shibboleth system has been added to the proposed solution, This latter allows the inter-university authentication and insure the secure sharing of the educational resources at the national level and/or the international one.


Key words: Authentication, SSO, LDAP, CAS, Shibboleth, Identity Provider.


---

# Cahier Des Charges

**THEME : Authentification SSO des services web de campus universitaire.**

**Définition des mots clés :**

**Authentification :** L'authentification est la procédure qui consiste, pour un système informatique, à vérifier l'identité d'une personne ou d'un ordinateur afin d'autoriser l'accès de cette entité à des ressources (systèmes, réseaux, applications…). L'authentification permet donc de valider l'authenticité de l'entité en question..

**SSO :** service d'authentification unique pour l'accès à un ensemble d'application.

**LDAP :** est un protocole, ce qui signifie que son rôle est de présenter des informations. Un serveur LDAP agit en tant qu'intermédiaire entre une source de données et un client.

**CAS :** est proposé comme mécanisme d'authentification centralisé de web SSO

**Shibboleth :** Shibboleth est le moyen technique utilisé par les universités françaises (mais aussi beaucoup d'universités à travers le monde) pour répondre à la problématique de fédération d'identités.

**Fournisseur d'identité :** organisation membre d'une fédération gérant l'identité informatique d'un ensemble d'utilisateurs et offrant un service d'authentification à ses utilisateurs, afin de s'authentifier sur le réseau.

**Problématique**

Comment permettre aux utilisateurs des services web de l'université Kasdi Merbah de s'authentifier une seule fois pour un accès global aux différentes ressources tout en se rassurant de leur identité, plus la fédération entre les universités.

**Travaux préliminaires :**

- ✓ Présenter les avantages des systèmes d'authentification centralisée
- ✓ Présenter les apports d'authentification SSO et d'un fournisseur d'identités
- ✓ Présenter la solution motivée et son fonctionnement
- ✓ Donner une architecture du système.

**Moyens :**

- ✓ Utilisation des systèmes d'exploitations LINUX distribution Centos version 6.3 et Microsoft Windows pour les clients.
- ✓ Utilisation des logiciels libre tels que OpenLDAP, SAMBA, Tomcat, Shibboleth, CAS client et CAS server.

**Fonctionnalités attendues :**

- ✓ Authentification centralisée avec l'annuaire LDAP.
- ✓ Contrôleur de domaine avec le serveur SAMBA.
- ✓ Serveur de SSO déployé avec CAS.
- ✓ Déployer le fournisseur d'identité avec la brique Shibboleth pour avoir la possibilité d'accéder à une fédération.

Durée du projet : 4 mois

**Encadreur : M. kafi Mohammed Radouane.**

**M. Euschi Saleh**

# Introduction Générale

La croissance d'applications web, et donc avec autant des mots de passe et des login à mémoriser, les utilisateurs font de leur mieux : ils inscrivent ces codes secrets dans leur agenda papier, les notent sur des Post-it qu'ils collent autour de leur écran ou, plus simple, laissent leurs connexions ouvertes lorsqu'ils quittent leur poste de travail. Et ce, afin de ne pas avoir à répéter le rituel quotidien de l'accès sécurise a leurs applications. Pour Arrêter cette prolifération de mots de passe et login, un remède existe : le Single Sign-On (SSO). Il s'agit d'une gamme d'outils qui mémorise à la place des utilisateurs l'ensemble de leurs codes secrets et les présente automatiquement à chaque application qui en fait la demande. L'utilisateur n'a qu'à s'authentifier une seule fois, le plus souvent à l'ouverture de sa session de travail, le reste est pris en charge par le SSO. Ces informations de login et mots passe peuvent être stockes de façon centralisée ou localement, sur une base de données.

L'objectif de notre projet est faire une étude sur les systèmes d'authentification unique et les solutions de Single Sign-On que nous avons choisi pour réaliser notre projet.

## ORGANISATION DU MEMOIRE

Ce mémoire est organisé comme suit : Nous commencerons par une **introduction générale** où on introduit notre thème de recherche ainsi que les objectifs de notre travail. Dans **le premier chapitre** nous présenterons des généralités sur les SSO, les Objectifs du système, les avantages du SSO, l'architecture d'un SSO web, exemples des solutions SSO et en fin nous citerons quelques projets universitaires. **Dans le chapitre 2,** nous trouverons une conception et une étude théorique sur la première partie de notre choix de solution CAS. **Le chapitre 3,** présentera une conception et une étude théorique sur la deuxième partie de notre choix de solution Shibboleth. **Le chapitre 4,** sera consacré à la mise en ouvre de notre solution, et enfin, nous avons conclu notre mémoire par une **conclusion générale** et nous avons également exposé des perspectives pour des travaux futurs.

# Chapitre 1 SSO (Single Sign-On)

## 1. INTRODUCTION

Le SSO, Single Sign On ou authentification unique, est une architecture complexe permettant idéalement, à un utilisateur, de s'authentifier une seule fois, puis d'avoir un accès libre à toutes les ressources (applications, données, ...) aux quelles il est autorisé à accéder.

Le SSO est donc une architecture informatique composée de plusieurs systèmes tel que le centre d'authentification, les serveurs applicatifs, etc., permettant de propager l'identité d'un utilisateur déjà authentifié à toutes les applications [13].dans ce chapitre nous parlons de l'architecture de SSO, son objectif et quelques exemples de projets universitaires dans ce domaine de SSO ...etc.

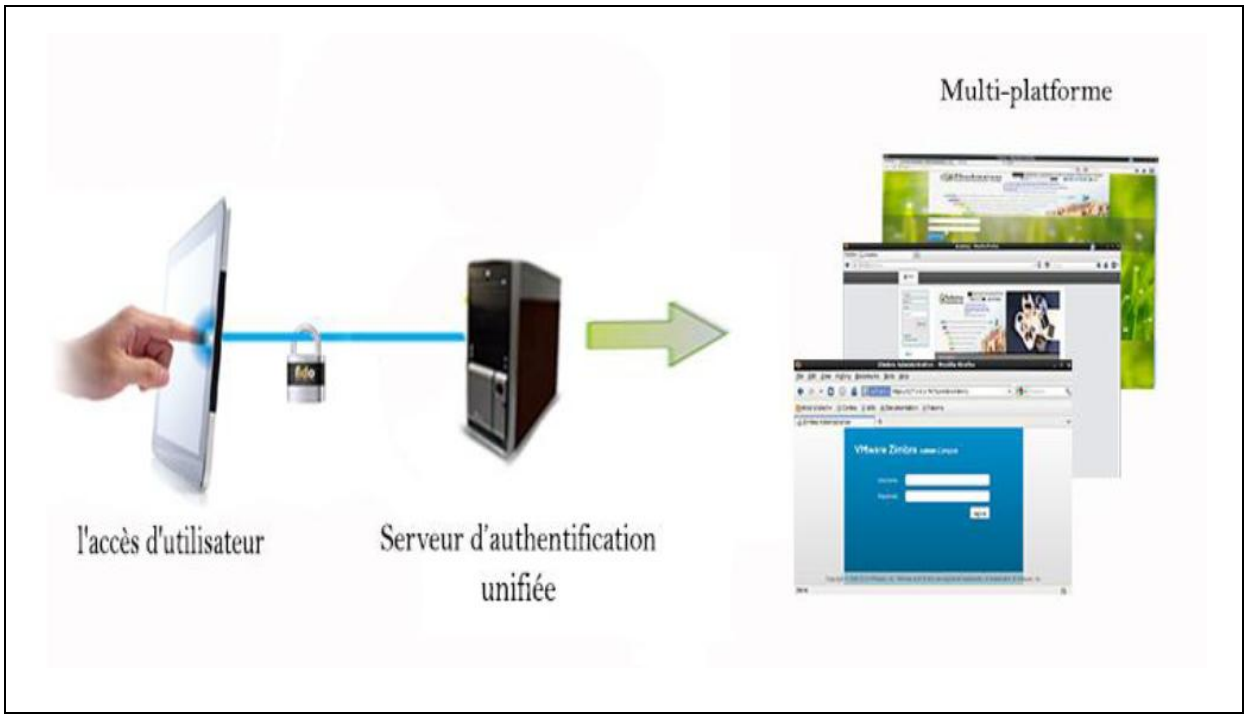



**Figure 1 : Authentification ou unifiée des utilisateurs.**

## 2. Objectif d'un Système SSO

Single Sign-On, noté SSO consiste à centraliser l'authentification afin de permettre à l'utilisateur d'accéder à toutes les ressources (machines, systèmes, réseaux ) .

Le Single Sign On unifie les comptes de l´utilisateur en un couple unique login/mot de passe. L´utilisateur s´authentifie une seule fois lors de son premier accès à l´application.

Ce système d´authentification est l´occasion de donner de bonnes habitudes aux utilisateurs : ils ne doivent délivrer leur mot de passe qu´au seul serveur d´authentification dont la bannière de login et l´URL sont bien identifiés [13].

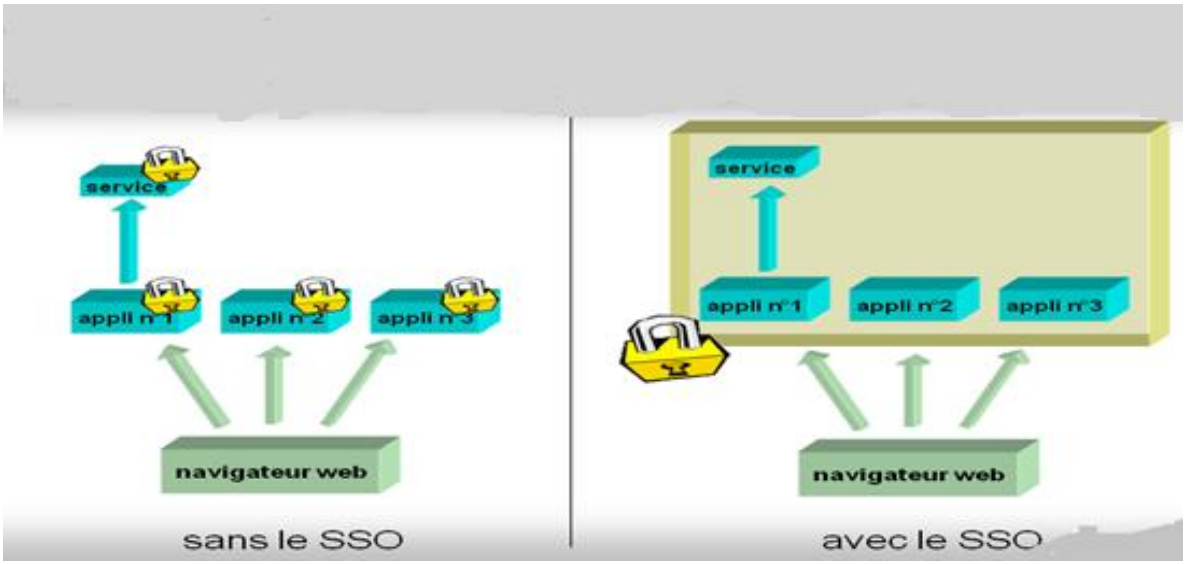



**Figure 2 : Principe de SSO [10].**

## 3. Les Avantages Du Single Sign-On

La mise en place d'un système SSO apporte beaucoup d'avantage, en voici une liste non exhaustive qui présente les principaux que l'on peut recenser. Le SSO permet l'amélioration de la sécurité [13]:

- En limitant le nombre de mot de passe pour chaque utilisateur.
- En limitant le nombre d'échange réseau pour l'authentification.
- En centralisant l'authentification dans un seul système:

- moins de risque de faille
- permet un système de trace plus facile
- limite les risques d'inconsistance

Le SSO permet l'amélioration du confort des utilisateurs:

- En limitant le nombre de saisies pour les authentifications (mot de passe).
- En limitant les appels au support informatique (perte de mot de passe, blocage de compte, etc.)
- Permettant bien souvent une centralisation sur un portail des applications [13].

## 4. A prendre en considération lors de l'utilisation SSO

Les services numériques accessibles par le web (intranet, courrier électronique, forums, agendas, applications spécifiques) à disposition des étudiants, enseignants, personnels administratifs se sont multiplies en quelques années. Ces services nécessitent très souvent une authentification.

L'utilisation de techniques de synchronisation entre domaines d'authentification hétérogènes, puis de serveurs LDAP a permis la mise en œuvre d'un compte unique (*login* / mot de passe) pour chaque utilisateur, ce qui est un progrès. Se posent maintenant les problèmes suivants [18]:

- **Authentifications multiples :** il est nécessaire d'entrer son *login*/mot de passe lors de l'accès a chaque application.
- **Sécurité:** le compte étant unique, le vol de celui-ci entraine un risque très important. La sécurisation de l'authentification devient donc primordiale. Il est également fortement souhaitable que les applications n'aient pas connaissance du mot de passe.
- **Différents mécanismes d'authentification:** certains utilisateurs disposent de certificats X509, qui pourraient servir à l'authentification. En outre, il n'est pas exclu que l'utilisation de LDAP a cette fin ne soit pas remplacée à terme par autre chose, et que certaines politiques d'établissement exigent l'utilisation de bases de données additionnelles. Il semble donc intéressant de disposer d'un service d'abstraction par rapport au(x) mécanisme(s) d'authentification local (aux) [18].

## 5. L'architecture Classique d'un Single Sign-On

L'architecture de la plupart des produits de SSO est inspirés de Kerberos ; ils utilisent largement sa terminologie et partagent ses concepts de base qui sont les suivants :

- Les applications sont déchargées du travail d'authentification des utilisateurs. Cette tâche est assurée par un serveur d'authentification dédie.
- Le serveur d'authentification délivre des tickets au client (maintien 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 (couple identifiant + mot de passe par exemple).
- Il existe une relation de confiance entre les applications et le serveur d'authentification. A noter que Kerberos n'utilise que des techniques de cryptographie symétriques, l'utilisation de certificats X509 (utilisant des algorithmes asymétriques) peut permettre de simplifier l'architecture du système [9] ou renforcer la sécurité du système.
- **Le serveur d'authentification :** 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'utilisateur fournit ses éléments d'authentification au serveur d'authentification. Si le mode d'authentification est le mot de passe, la phase d'authentification implique la vérification du mot passe de l'utilisateur auprès d'une base de référence. La plupart des systèmes de SSO implémentent plusieurs backend d'authentification (/etc/Password, NIS, LDAP) [9].
- **L'agent d'authentification :** 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 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 [9].

Les principaux modèles d'architecture SSO sont :

- le SSO côté client (client-side SSO).
- le SSO côté serveur (server-side SSO).

## 5.1. **SSO Coté Client**

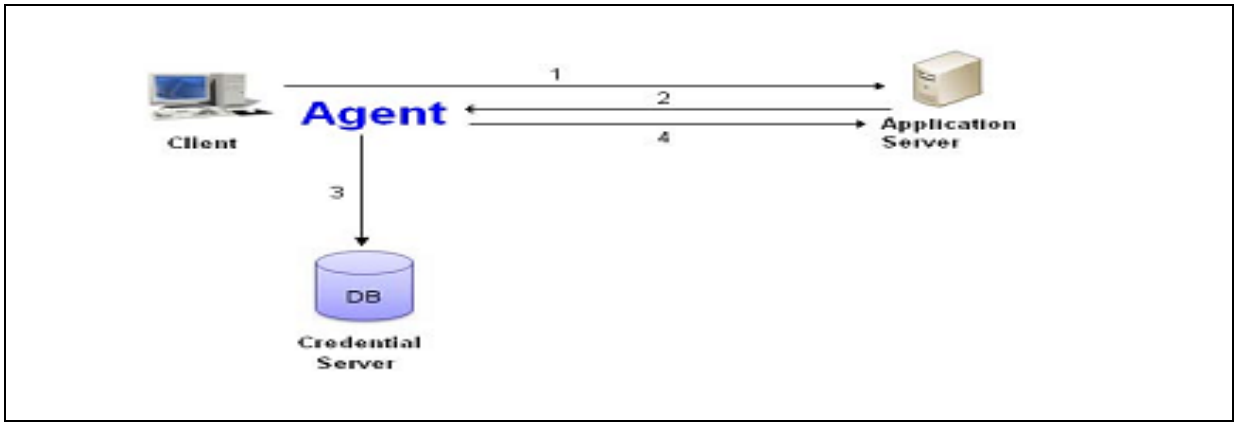



**Figure 3 : Client Side SSO with centre credential database[1] .**

1. Le client souhaite accéder à une application.
2. L'application demande ses informations d'identification. L'agent présent sur le poste client intercepte la demande.
3. L'agent vérifie les informations d'identification dans le serveur centralisé des identifications.
4. L'agent simule l'utilisateur réel et envoie les informations d'identification à l'application. Ainsi, l'identification est transparente pour l'utilisateur.

## 5.2. SSO Coté Serveur

Il existe deux types de SSO "Server-Side" : ceux qui utilisent un **reverse proxy** et ceux utilisant des agents "**serveurs**". Avec cette architecture, il n'est pas nécessaire d'installer un agent sur chacun des PC de l'utilisateur [1].

### 5.2.1. Architecture Avec Agent

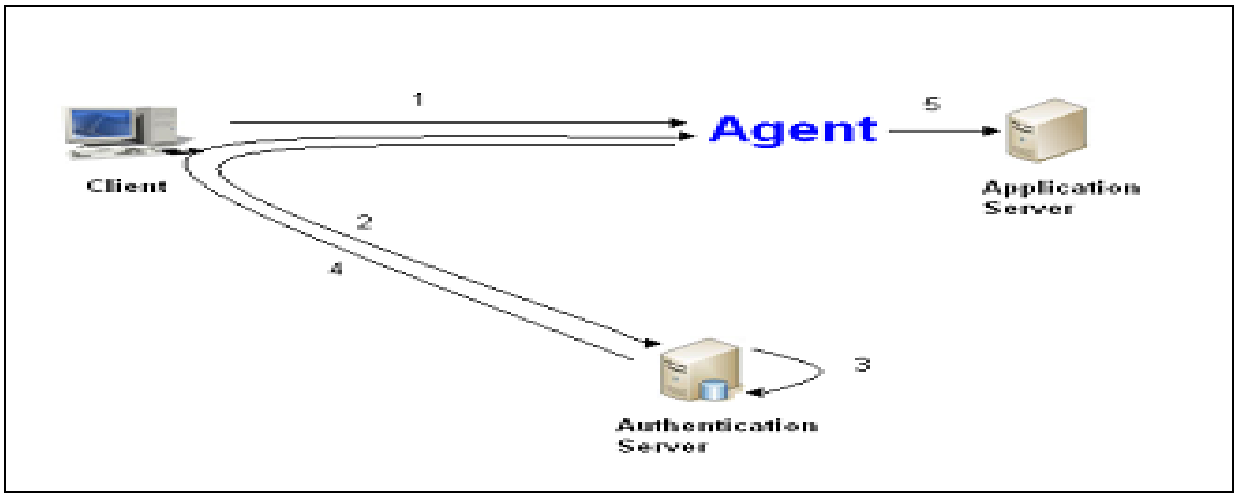



**Figure 4 : Server Side SSO with Agent on Application server [1].**

1. Le client souhaite accéder à une application. L'agent présent sur le serveur d'application intercepte la demande [1].
2. L'agent vérifie que l'utilisateur est authentifié: s'il ne l'est pas, l'agent le redirige vers le Serveur d'authentification. Cette redirection peut apparaître comme un portail ou une fenêtre. L'utilisateur fournit ses informations d'identification pour le serveur d'authentification [13].
3. Le serveur d'authentification vérifie l'identité de l'utilisateur dans la base de données de référence.
4. Une fois l'utilisateur authentifié, le serveur d'authentification renvoie un cookie HTTP sur le poste de l'utilisateur qui permet de maintenir la session de l'utilisateur.
5. L'agent le transfert sur le serveur d'application [1].

### 5.2.2. Architecture Avec Reverse Proxy

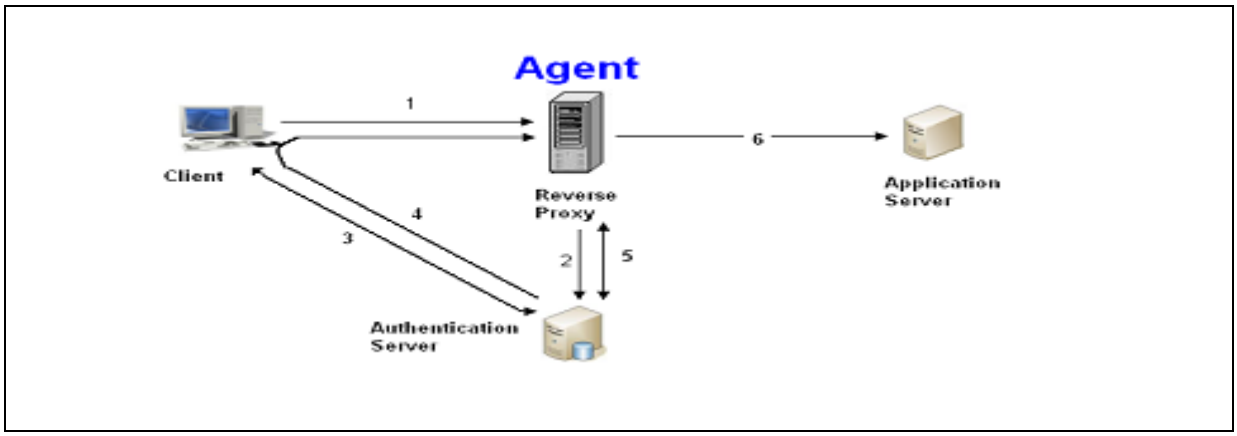



**Figure 5 : Server Side SSO with Agent on Reverse Proxy [1].**

1. Le navigateur Web tente de se connecter à l'application Web. Toute demande de connexion pour une application est redirigée vers le reverse proxy.
2. L'agent sur le reverse proxy intercepte la demande et vérifie l'authentification de l'utilisateur via le serveur d'authentification.
3. Si l'utilisateur n'est pas authentifié, le serveur d'authentification demande à l'utilisateur des informations d'identification. L'utilisateur fournit ses informations d'identification pour le serveur d'authentification.
4. Le serveur d'authentification envoie un jeton jouant le rôle de cookie et redirige le navigateur.
5. L'agent sur le reverse proxy intercepte la demande et vérifie l'authentification de l'utilisateur via le serveur d'authentification avec le jeton. Le serveur d'authentification envoie le login et l'autorisation des informations qui est associée avec le jeton.
6. L'agent permet d'accéder à la demande [1].

# 6. Les différentes approches du SSO

## 6.1. L'approche Centralisée

Le Principe de cette approche est de disposer d'une base de données centralisée contenant tous les utilisateurs. Cela permet également de centraliser la gestion de la politique de sécurité. Un exemple de mise en œuvre est LDAP [1].

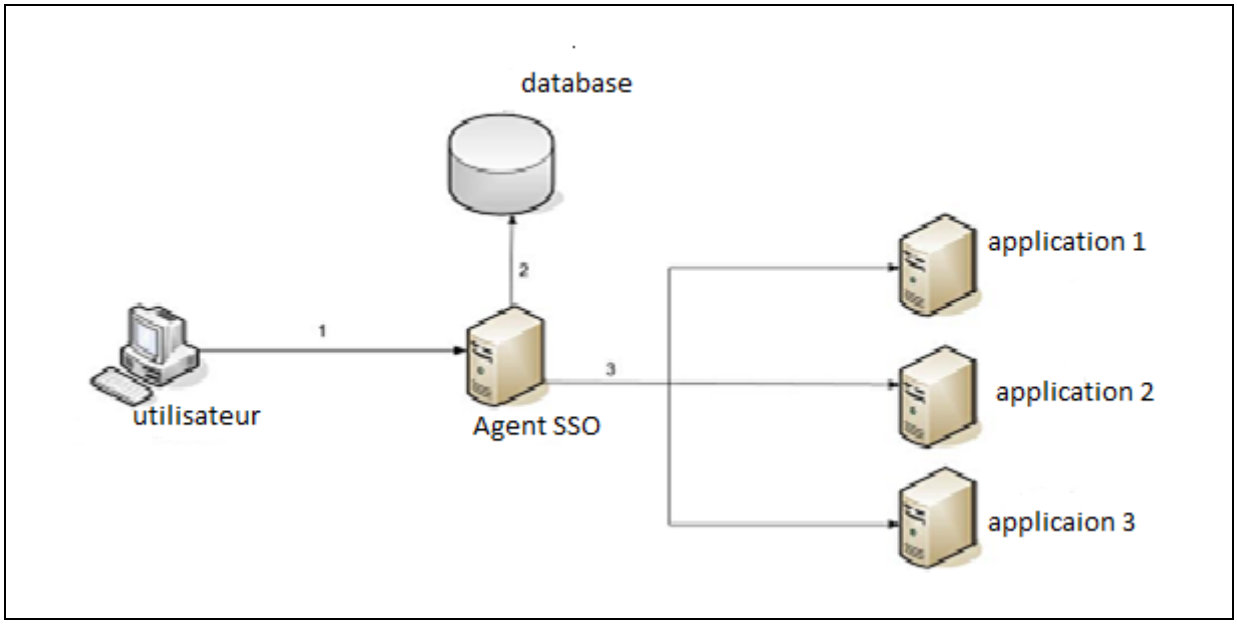



**Figure 6 : Centrallisation de SSO [1].**

1. Le client souhaite accéder à une application. dans ce cas, l'agent qui s'exécute sur un Reverse Proxy intercepte la demande.
2. L'agent authentifie l'utilisateur dans une base de données d'authentification qui peut être un annuaire LDAP.
3. Une fois l'utilisateur authentifié, il peut accéder à l'application [1].

## 6.2. L'approche Fédérative

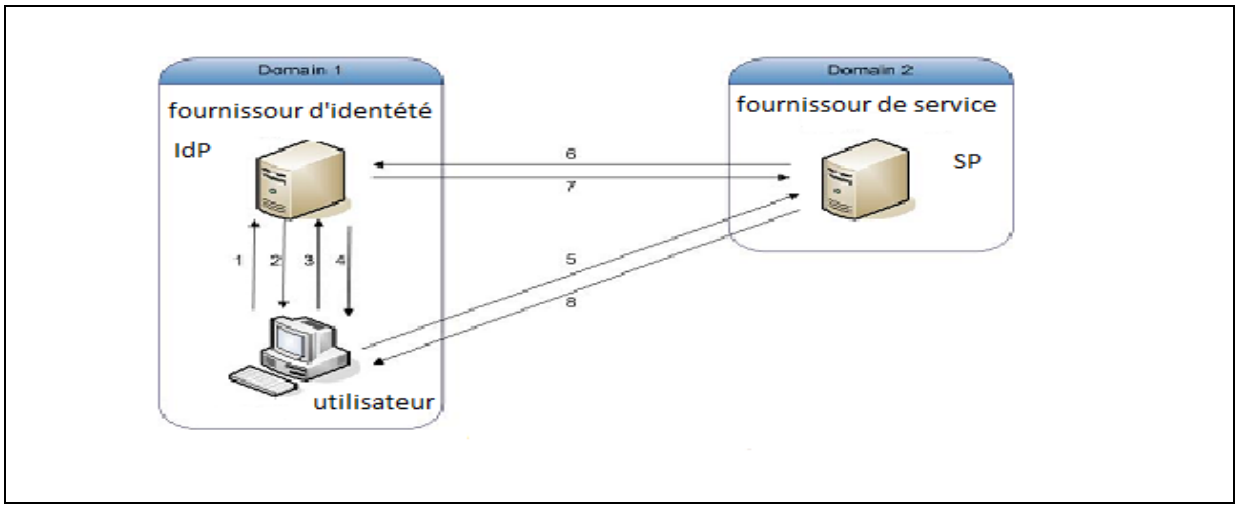



**Figure 7 : La fédération de SSO [1].**

1. L'utilisateur se connecte au fournisseur d'identité (IDP).
2. Après une authentification réussie, l'IDP envoie à l'utilisateur des informations sur les applications auxquelles il peut accéder.
3. L'utilisateur clique sur le lien Service Provider (SP) dans le portail. Il s'agit d'un lien spécial, qui ne se connecte pas directement au SP.
4. L'IDP reçoit la demande et crée une « Identité Assertion » (un identifiant d'identité). L'IDP conserve cette « Identité Assertion » avec un "artefact" pointeur dans son cache. Puis, l'IDP renvoie une réponse redirigée vers le navigateur client.
5. Le navigateur est redirigé vers le SP avec «l'artefact».
6. Le SP reçoit cette demande et contacte l'IDP avec « l'artefact » "pour demander « L'Identité Assertion » réelle.
7. L' IDP reçoit la demande, et vérifie cette entrée dans la table des « Identity Assertion » en cache en utilisant « l'artefact » comme index. Il crée une « Identité Assertion » au format SAML, et le renvoie à la SP [1].

8. Le SP extrait les informations utilisateur de l' « Identité Assertion » reçue.

Enfin, après une Authentification locale réussie, l'utilisateur est autorisé à accéder au service [13].

## 10. Différentes Solutions de SSO

Le tableau suivant présente quelques-unes des Sign-On (SSO) simples implémentations disponibles [7]:

| Nom du produit | Description |
|---|---|
| SOLUTION OpenSSO/OpenAM | Portant le nom d'oracle OpenSSO en juillet 2008 puis le nom OpenAM depuis 2010, c'est l'un des serveurs d'authentifications unique et de fédération complet de l'OpenSource. |
| SOLUTION SAML | SAML pour Security Assertion Markup Language permet entre autres la délégation d'authentification et sert de fondation à deux autres normes, Shibboleth et Liberty Alliance. |
| SOLUTION CAS | C'est un service d'authentification centralisé SSO pour les applications Web, inspiré de Kerberos et basé sur le protocole HTTP(S). |
| SOLUTION SHIBBOLETH | Shibboleth est développé depuis 2001 par Internet2 et désigne à la fois une norme et un produit open source. C'est une extension de SAML qui enrichit ses fonctionnalités de fédération d'identités. |
| SOLUTION OpenID | implémenté et utilisé par les sociétés clés de l'Internet (Yahoo, MySpace, Google, Microsoft...), propose un protocole ouvert pour une gestion décentralisée d'identités, mettant l'utilisateur au centre des décisions le concernant. |

| | |
|---|---|
| SOLUTION LIBERTY ALLIANCE | implémenté par IBM et utilisé par Sun et Novell, utilise des jetons SAML. Ce modèle a été développé pour répondre à un besoin de gestion décentralisée des utilisateurs |

**Tableau 1 : des solutions de SSO**

## 11. Quelques Projets Universitaire

Répondant à la problématique de l'authentification dans les applications web, de nombreuses universités en Europe et aux Etats-Unis, ont développé des solutions locales de SSO qui sont devenues des produits.

- **Pub cookie (université de Washington) [15] :** Un produit très bien packagé et documenté. Les échanges entre le serveur d'authentification (login server) et les agents d'authentification (Apache module) sont cryptés avec des clés symétriques ; un serveur de clés assure la distribution de ces clés sur les différents serveurs. Le module Apache permet de définir précisément l'ensemble des ressources à contrôler.
- **PAPI (RedIris - Espagne) [14] :** PAPI géré l'authentification et l'autorisation pour l'accès a des ressources. Le système est utilisé pour contrôler l'accès a des bibliothèques universitaires espagnoles. Le système impose que l'utilisateur s'authentifie avant de contacter le gestionnaire de ressources. Autre inconvénients : une liste exhaustive des ressources doit 'être maintenue au niveau du serveur d'authentification.
- **CAS (université de Yale) [11]**

Ce produit est le plus inspiré de Kerberos et bénéficie d'un bon niveau de sécurité. Il est le seul à prendre en charge la délégation de privilèges dans les architectures multi tiers (proxied crédential) grâce a l'utilisation de tickets de délégation. Sa gestion de tickets « non rejouables » (invalides après la première vérification) offre un très bon niveau de sécurité. CAS souffre initialement d'un manque de documentation, mais l'équipe du projet Esup-Portail (projet de campus numérique basé sur des logiciels libres) a largement contribué à combler ce manque [12].

## 12. Choix de solution

Les solutions d'authentification présentées ci-dessus nous ont permis de voir certaines particularités ou certains avantages des unes par rapport aux autres. Pour répondre aux exigences de notre système, les particularités impressionnantes des solutions CAS et Shibboleth ne nous ont pas laissé seulement le choix de l'un d'entre eux. Nous avons donc associé ces deux solutions.

## 13. Conclusion

Après cette étude en déduire que l'Authentification SSO des services web avec notre choix de solution CAS-Shibboleth est un nouveau projet pour notre université (Kasdi Merbah Ouargla), qui offre des perspectives pour maitre une coopération et la convergence entre UKMO et les autres universités algériennes, ces dernières permettent de faire un espace de confiance et d'échange des services entre elles. Tout ça avec une bonne sécurité basée sur des mécanismes d'authentifications fédérées.

# Chapitre 2 *Conception CAS (Central Authentication Service*

## 1. Introduction

Plusieurs solutions de Single Sign-On (authentification unique et unifiée) sont d'ores et déjà disponibles dans le commerce. On décrit une solution libre simple, riche et sure, CAS (central Authentification Service) est un système SSO pour le web développé par Shawn Bayem de l'université Yale. Dans ce chapitre nous présentons pourquoi nous avons choisi ce mécanisme ?, nous définissons ce système, son architecture et son fonctionnement de base on parlera de l'authentification avec annuaire LDAP et enfin voir comment CAS-ifier des services c-à-d comment faire pour qu'un service s'authentifie grâce à un mécanisme CAS.

## 2. Le Choix de CAS

CAS est en utilisation dans plusieurs universités américaines, avec des authentifications internes Kerberos ou LDAP, ce qui permet d'être confiant sur sa fiabilité. La sécurité est assurée par les dispositifs suivants :

- **Sécurité**
  - Le mot de passe n'est transmis qu'au serveur d'authentification, nécessairement à travers un canal crypté.
  - Utilisation de tickets « opaques » et à usage unique.
- **Mécanisme n-tiers**
  - Utilisation de services sans transmission du mot de passe
- **Portabilité (librairies clientes)**
  - Java, Perl, JSP, ASP, PHP, PL/SQL...
  - Un module Apache (mod_cas) permet d'utiliser CAS pour protéger l'accès à des documents web statiques [7].
  - Un module PAM (*pam_cas*) permet de « CAS-ifier » des services non *web*, tels que FTP, IMAP, ... [7].
  - Adaptation aisée des applications

- **Pérennité**
  - Développé par l'Université de Yale
  - En utilisation dans les universités (américaines notamment)

# 3. Le Mécanisme CAS

## 3.1 Architecture :

L'architecture de CAS est basée sur 3 principaux acteurs le serveur CAS, les navigateurs web et le client CAS :

- **Le serveur CAS :** L'authentification est centralisée sur un serveur CAS. Ce serveur est le seul acteur du mécanisme CAS qui relaye les mots de passe des utilisateurs à l'annuaire centralisé LDAP. Son rôle est double :
  - Authentifier les utilisateurs ;
  - Certifier l'identité de la personne authentifiée aux clients CAS par son ticket de service. [4]
- **Les navigateurs Web :** Ils doivent satisfaire les contraintes suivantes pour bénéficier de tout le confort de CAS :
  - Permettre le protocole HTTPS
  - Savoir stocker des cookies (en particulier, les cookies privés ne devront être retransmis qu'au serveur les ayant émis pour garantir la sécurité du mécanisme CAS).

Ces exigences sont satisfaites par tous les navigateurs classiquement utilisés, à savoir Microsoft Internet Explorer (depuis la version 5.0), Netscape Navigator (depuis la version 4.7) et Mozilla Firefox (depuis la première version). [4]

- **Le client CAS :** Une application Web munie d'une bibliothèque cliente ou d'un client CAS intégré est alors appelé « client CAS ». Il ne délivre les ressources qu'après s'être assuré que le navigateur qui l'accède se soit authentifié auprès du serveur CAS. Le client s'assure également à intervalle régulier de la validité de la session CAS. [4]

## 3.2 Fonctionnement de base :

- **Authentification d'un utilisateur :**

  Un utilisateur non authentifié, ou dont l'authentification à expiré, et qui accède au serveur CAS. Se dernier lui propose un formulaire d'authentification, dans les quel il est invite d'enter son identifient et son mot passe (Figure 8).

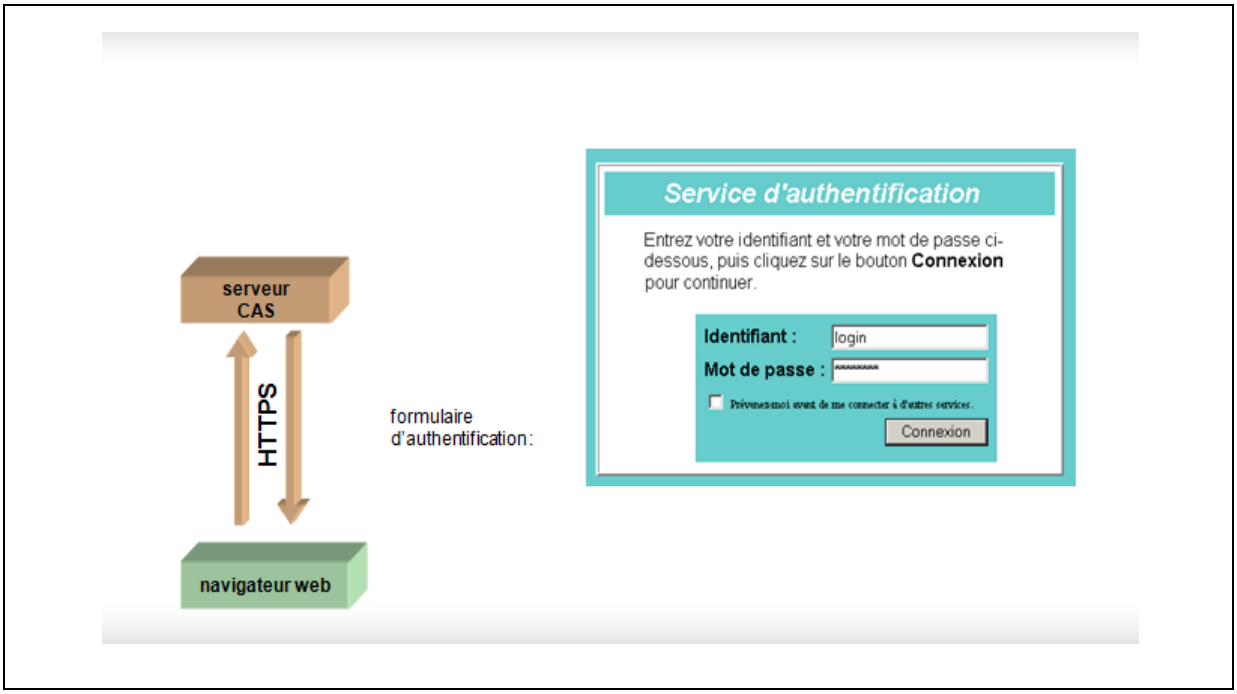



**Figure 8 : Premier accès d'un navigateur [10]**

Si les informations d'authentification sont correctes, le serveur renvoie au navigateur **un cookie** appelé **TGC (Ticket Granting Cookie)** (Figure 9).

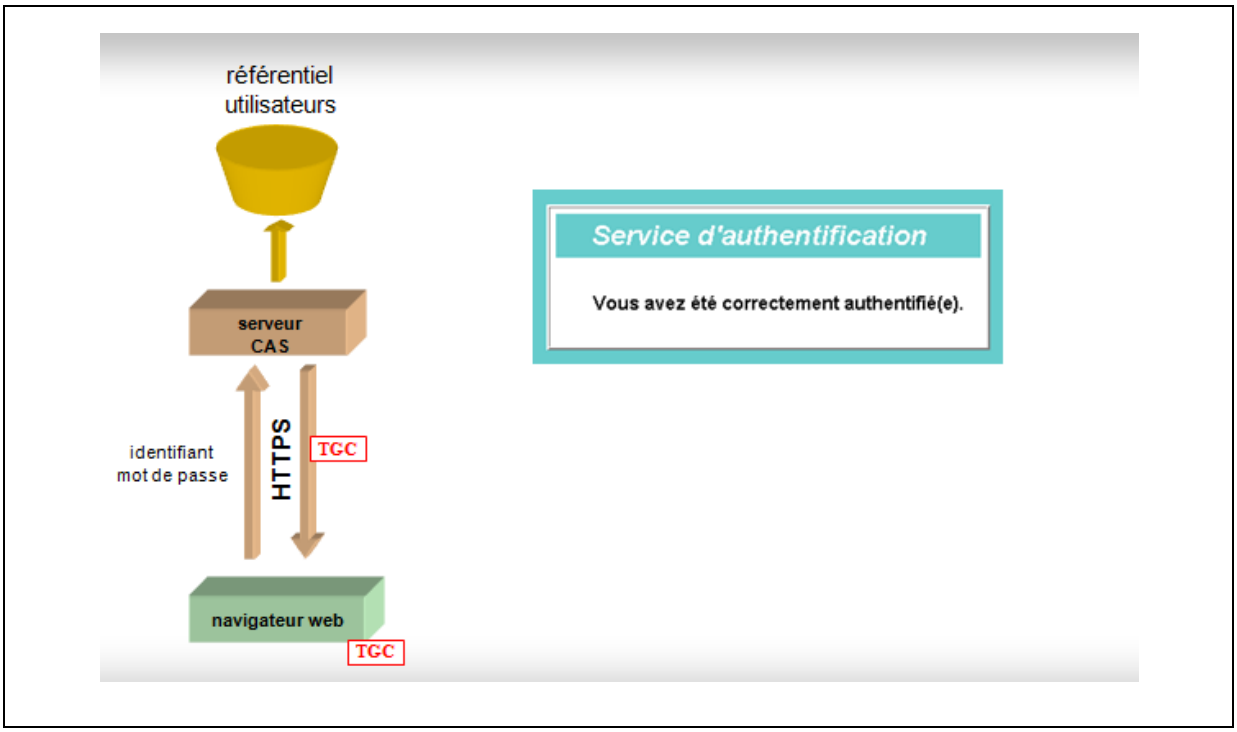



**Figure 9 : Authentification d'un navigateur auprès de serveur CAS [10]**

- **TGC (Ticket Granting Cookie)** [4] :

  – Passeport du navigateur auprès du serveur CAS.

  – A une durée de vie limitée à celle de la session du navigateur (une fois le navigateur fermé, la session est perdue).

  – C'est un cookie privé (n'est jamais transmis à d'autres serveurs que le serveur CAS).

  – Protégé (tous les requêtes des navigateurs vers le serveur CAS se font sous HTTPS).

- **Accès à une ressource protégée après l'authentification :**

  La figure ci-dessous présente la séquence d'accès à une ressource protégée par CAS :

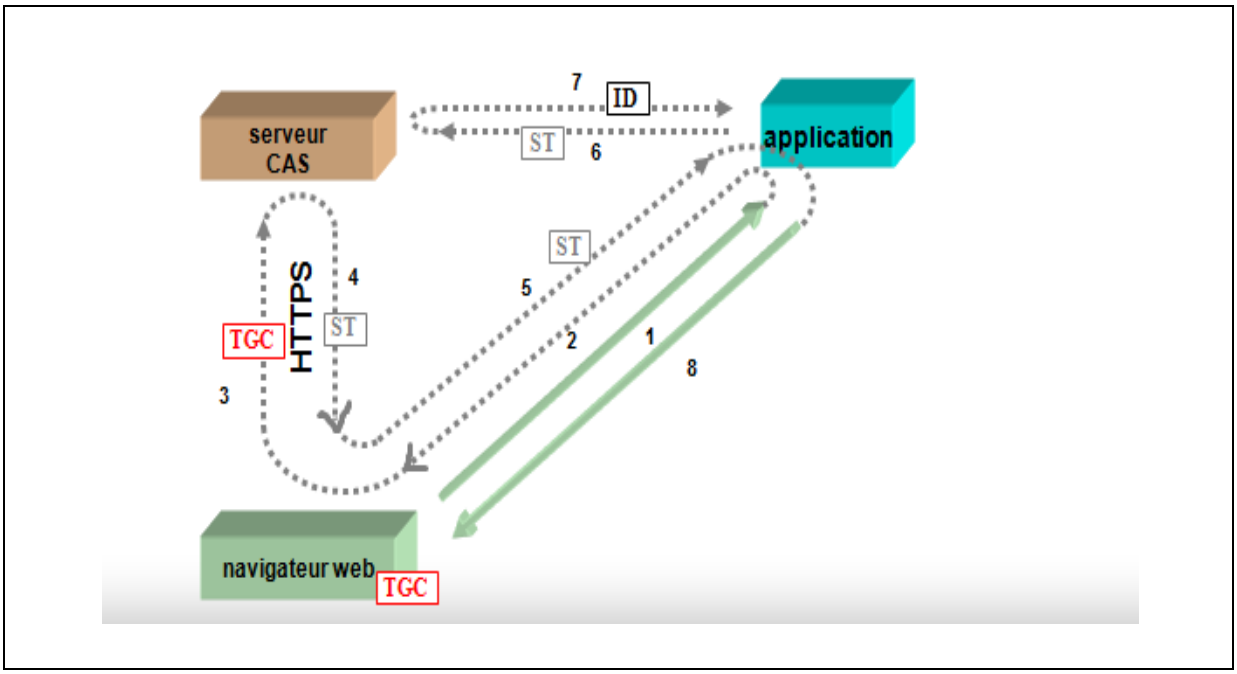



**Figure 10 : Redirection par le serveur CAS d'un navigateur vers un client CAS après authentification [10].**

Les étapes sont [4] :

- (1) : Le navigateur tente d'accéder à une ressource protégée sur un client CAS.
- (2) : Le client redirige le navigateur vers le serveur CAS dans le but d'authentifier l'utilisateur.
- (3) : Le navigateur, précédemment authentifié auprès du serveur CAS, lui présente le TGC (rappel : un cookie).
- (4) : Sur présentation du TGC, le serveur CAS délivre au navigateur un Service Ticket (ticket de service ou ST) ; c'est un ticket opaque, qui ne transporte aucune information personnelle ; il n'est utilisable que par le « service » (l'URL) qui en a fait la demande.
- (5) : En même temps, le serveur CAS redirige le navigateur vers le service demandeur en passant le Service Ticket.
- (6) et (7) : Le Service Ticket est alors validé auprès du serveur CAS par le client CAS, directement en HTTP.
- (8) : la ressource est délivrée au navigateur.

Toutes les redirections présentées précédemment sont transparents pour l'utilisateur comme le montre la figure ci-dessous :

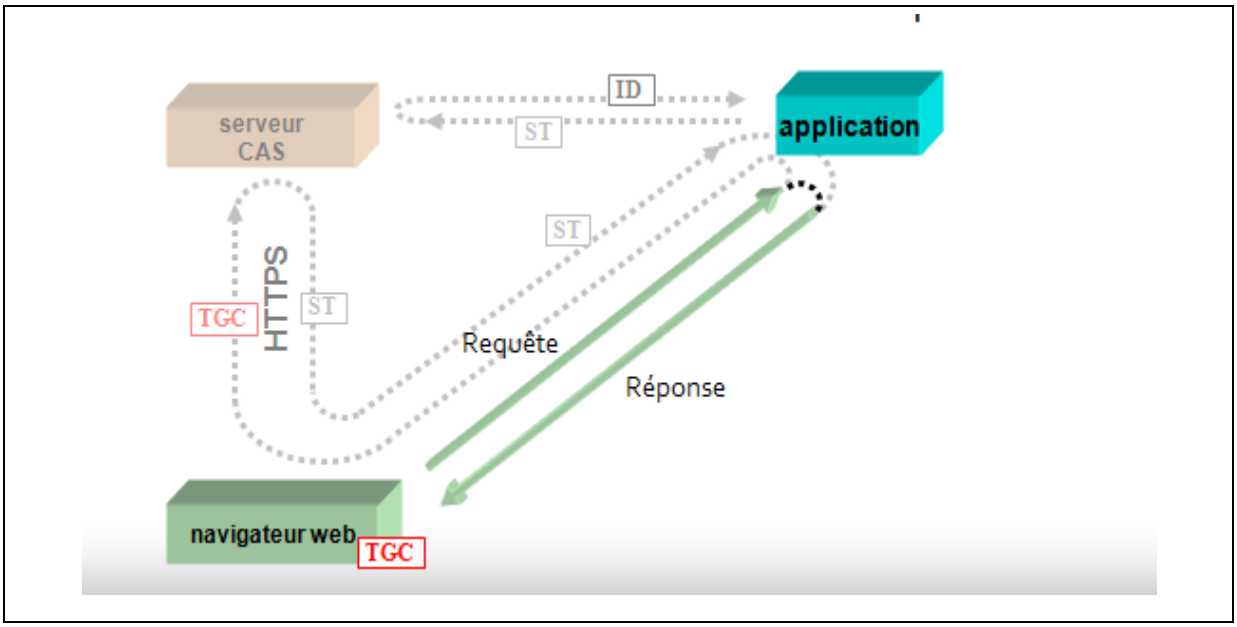
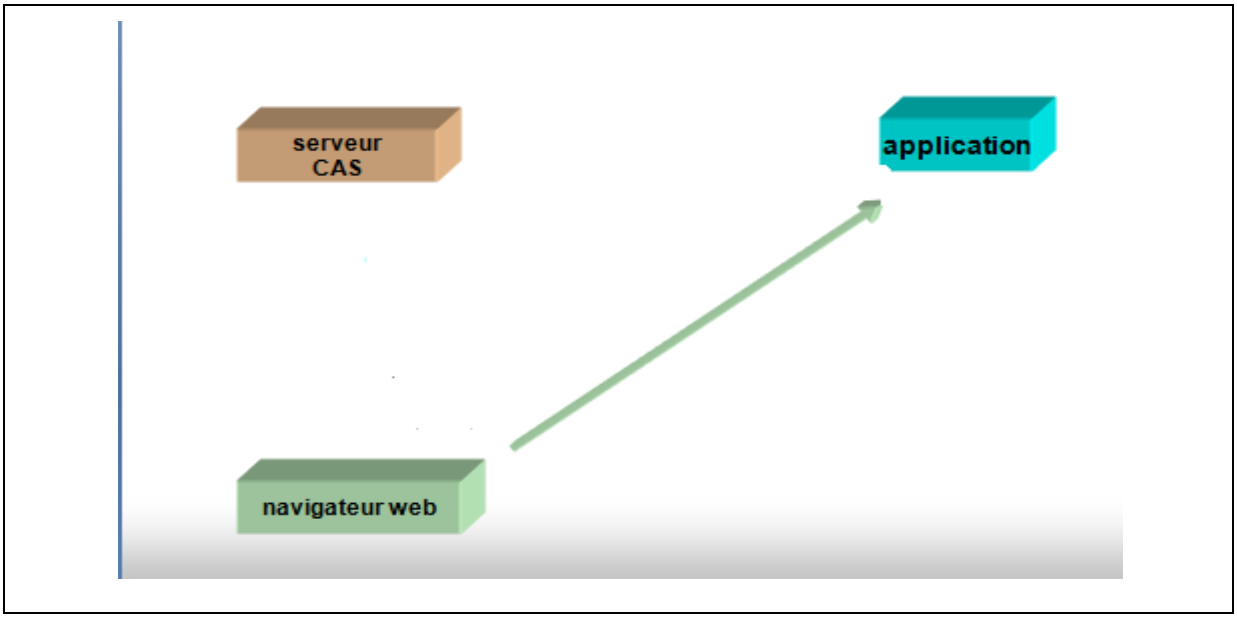



**Figure 11 : Vision par l'utilisateur du mécanisme d'authentification [10]**

- **Accès à une ressource protégée sans authentification préalable :**

  Si l'utilisateur n'est pas déjà authentifié auprès du serveur CAS avant d'accéder à une application web, son navigateur est, comme précédemment, redirigé vers le serveur CAS, qui lui propose alors un formulaire d'authentification [4] **(Authentification d'un utilisateur)** ensuite suivre les mêmes étapes qui ont été déjà définies précédemment **(Accès a une ressource protégée après l'authentification).**



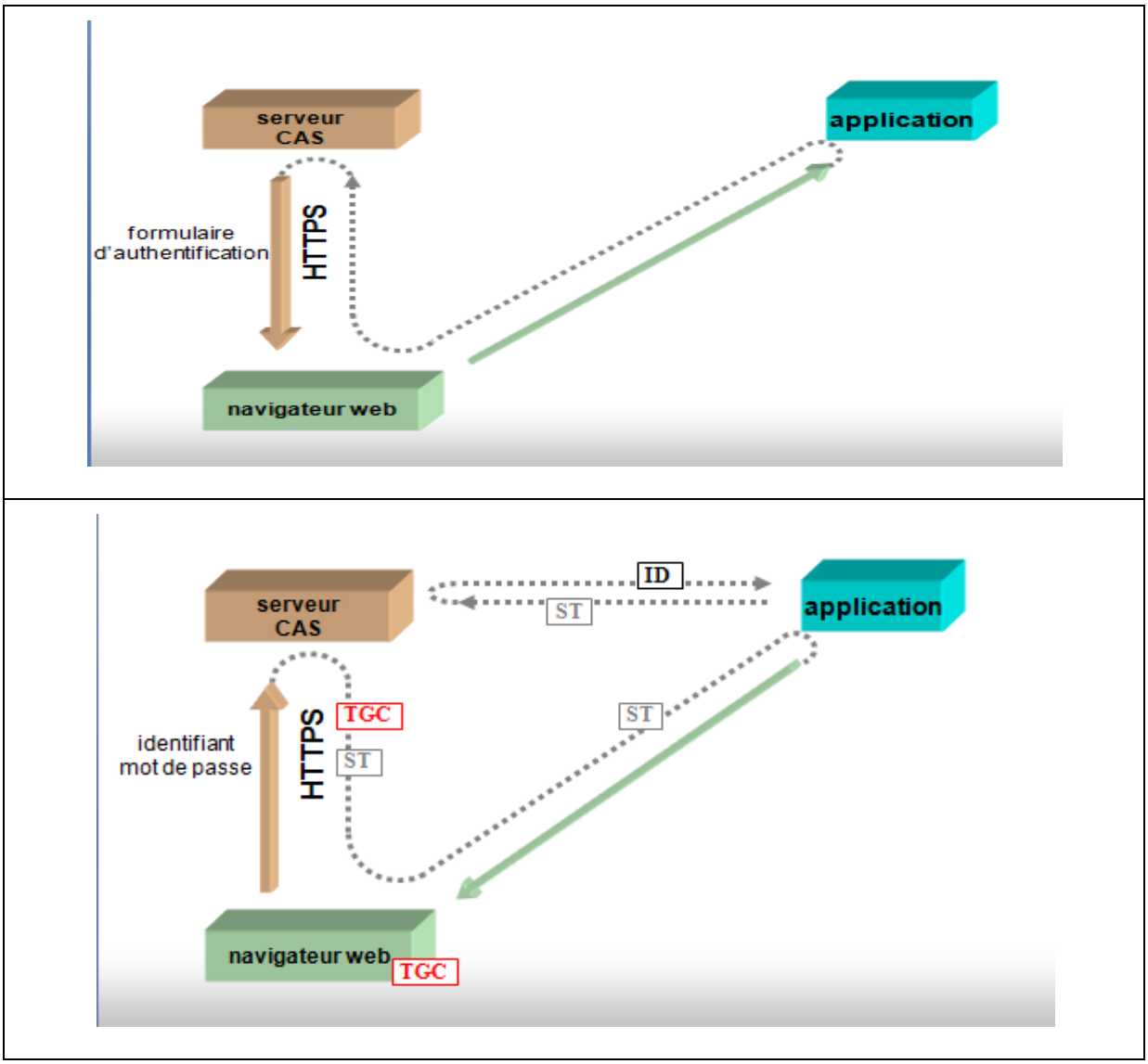



**Figure 12 : Accès à une application avant authentification [10]**

Nous remarquons qu'il n'est pas nécessaire d'être préalablement authentifié auprès du serveur CAS pour accéder à une application.

## 3.3 Fonctionnement multi-tiers

### 3.3.1 Les mandataires (Proxies) cas :

Le fonctionnement n-tiers de CAS consiste en la possibilité, pour un client CAS, de se comporter comme un navigateur. Un tel client CAS est alors appelé mandataire (proxy) CAS. Les exemples d'utilisation de mandataires sont divers :

- un **portail web**, pour lequel l'utilisateur s'est authentifié, peut avoir besoin d'interroger une application externe sous l'identité de l'utilisateur connecté (un *web service*).
- une **passerelle web** de courrier électronique (webmail), à laquelle un utilisateur s'est authentifié, a besoin de se connecter à un serveur IMAP pour relever le courrier électronique de l'utilisateur, sous son identité.

Dans le fonctionnement de base, le client CAS est toujours en lien direct avec le navigateur. Dans un fonctionnement n-tiers, le navigateur accède à un client CAS à travers un mandataire CAS. Le mécanisme de redirection vu dans le fonctionnement de base n'est alors plus applicable [10].

### 3.3.2 Fonctionnement n-tiers

Un mandataire CAS, lorsqu'il valide un *Service Ticket* pour authentifier un utilisateur, effectue, dans le même temps, une demande de PGT (*Proxy Granting Ticket*) (figure 13).

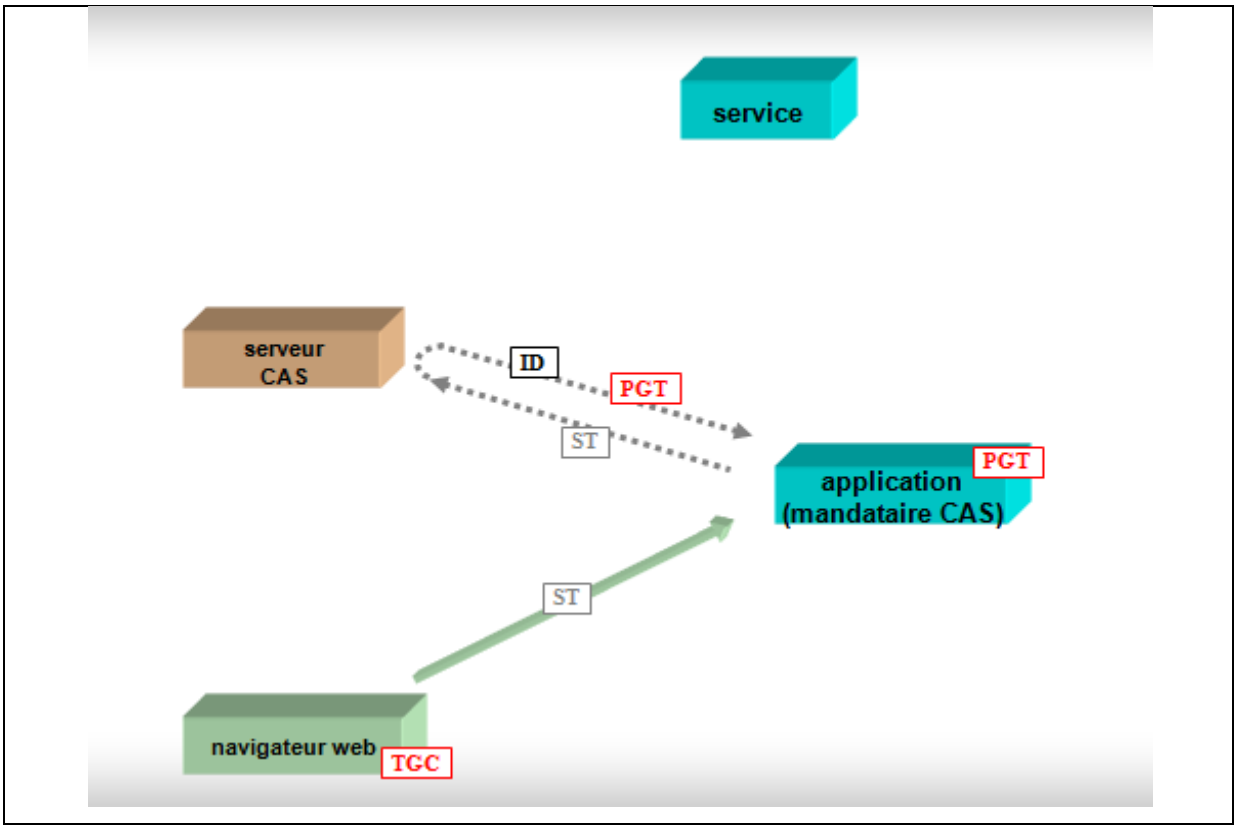



**Figure 13 : Récupération d'un PGT par un mandataire CAS auprès du serveur CAS [10]**

- **Le Proxy Granting Ticket (PGT) :** est le passeport d'un mandataire CAS, pour un utilisateur, auprès du serveur CAS. Le PGT est opaque, rejouable, et obtenu du serveur CAS par un canal chiffré. Comme le TGC, il est à durée de vie limitée (typiquement quelques heures). Le PGT est l'équivalent, pour les mandataires CAS, des TGCs pour les navigateurs. Il permet d'authentifier l'utilisateur auprès du serveur CAS, et d'obtenir ultérieurement du serveur CAS des Proxy Tickets, équivalents pour les mandataires des Service Tickets pour les navigateurs.
- **Les Proxy Tickets (PT) :** sont, comme les Service Tickets, validés par le serveur CAS pour donner accès à des ressources protégées. Il est le passeport des mandataires CAS auprès des clients CAS, et il est opaque, non rejouable, et à durée de vie très limitée (comme le Service Ticket, typiquement quelques secondes). [10]

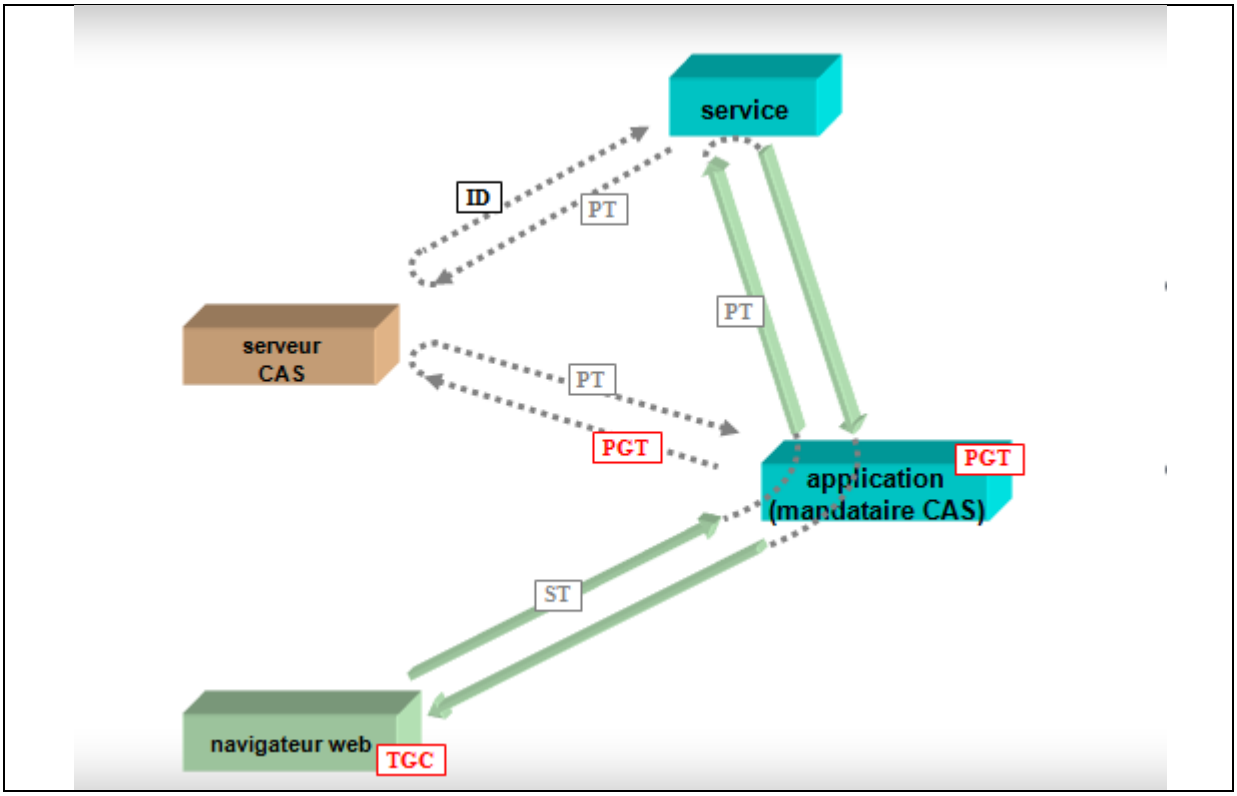



**Figure 14 : Validation d'un PT par un client CAS (service) accédé par un mandataire CAS (fonctionnement 2-tiers) [10].**

- Les mandataires CAS peuvent êtres chainée n-tiers (figure 15).

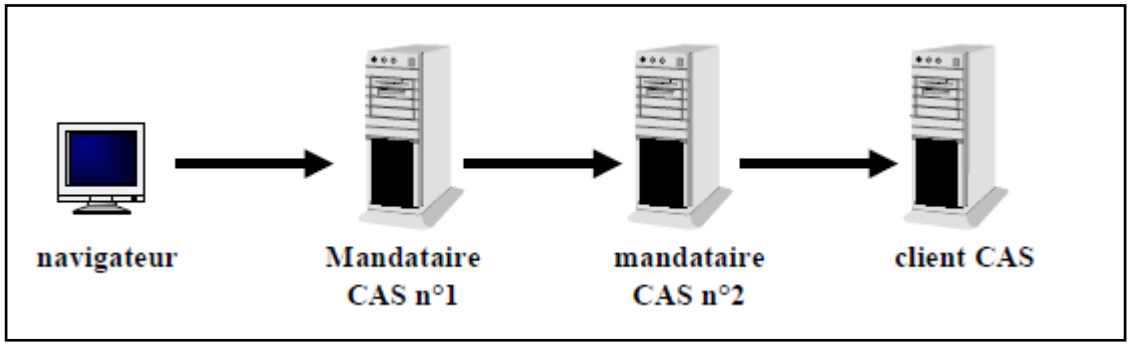



**Figure 15 : Schéma de fonctionnement n-tiers [10].**

## 4. L'authentification avec un annuaire Ldap

**Un annuaire électronique** est une base de donnée spécialise qui permet de partager des bases d'informations sur un réseau. Un service d'annuaire est rendu par le service NIS développé par SUN. C'est un protocole Client/serveur qui permet de diffuser des données de configuration (utilisateurs, mot passe, host etc..) entre les ordinateurs sur un réseau.[6]

**LDAP** signifie **LightWeight Directory Access Protocol** est un protocole d'annuaire fournit les services d'annuaire sur TCP/IP utilisant les mêmes concepts que DNS. LDAP apporte également de nombreuses garanties en terme de sécurité puisque des mécanismes de chiffrement (SSL, ou TLS) et d'authentification (SASL), couplé a des mécanismes de règles d'accès (ACL) permettent de protéger les transactions et l'accès aux données. [6]

LDAP fournit les services suivants :

- Un protocole réseau pour accéder a l'information contenue dans un annuaire.
- Un modèle d'information définissant la forme et le type de l'information.
- Un espace de nommage définissant comment l'information est organisée et référencée.
- Un modèle de distribution permettant de diffuser les données.
- Un protocole et un modèle de données extensible.
- …

LDAP est devenu un (le) standard de référentiel utilisateur, l'authentification sur un annuaire LDAP est aujourd'hui la méthode la plus souvent utilisée. Deux modes d'accès à l'annuaire sont proposés, selon la structure interne de l'annuaire [10]:

## 4.1 Accès direct a l'annuaire (ldap_fastbind)

Le mode **ldap_fastbind** peut être utilisé pour les annuaires dont le DN (Distinguished Name) d'un utilisateur peut être directement déduit de son nom de **login**. Pratiquement, il s'agit des annuaires dont tous les utilisateurs sont stockés au même niveau hiérarchique.

CAS tente alors une connexion à l'annuaire sur le DN de l'utilisateur et le mot de passe fourni. L'utilisateur est considéré comme authentifié si la connexion réussit [10].

Par exemple [10]:

**<authentication>**

**<ldap version="3" timeout="5">**

**<ldap_fastbind filter="uid=%u,ou=people,dc=univ-rennes1,dc=fr" />**

**<ldap_server host="ldap.ifsic.univ-rennes1.fr" port="389" secured="no" />**

**</ldap>**

**< /authentication>**

## 4.2 Recherche dans l'annuaire (ldap_bind)

Lorsque la déduction du DN de l'utilisateur à partir de son nom de connexion est impossible (Par exemple lorsque les utilisateurs de l'annuaire sont situés dans des hiérarchies différentes), il faut alors utiliser le mode **ldap_bind**, qui effectue une recherche du DN de l'utilisateur dans l'annuaire avant de tenter une connexion [10].

Par exemple [10]:

**<authentication>**
**<ldap version="3" timeout="5">**
**<ldap_bind search_base="dc=univ-rennes1,dc=fr" scope="sub" filter="uid=%u" bind_dn="admin" bind_pw="secret" />**
**<ldap_server host="ldap.ifsic.univ-rennes1.fr" port="389" secured="no" />**
**</ldap>**

**< /authentication>**

## 4.3 Redondance des annuaires

Afin d'assurer la tolérance aux fautes de l'annuaire LDAP, il est possible de spécifier non pas un seul annuaire LDAP, mais une liste d'annuaires, qui sont alors considérés comme des réplicas. La panne d'un annuaire est alors palliée par la présence de ses réplicas [10].

## 5. La CAS-ification des services

Le CAS-ification d'un service c'est le rendre compatible avec CAS, cela veux dire configurer ce service de telle sorte qu'il puisse être intégrer avec CAS. Nous présentons ci-dessous un exemple pour CAS-ifier une application mandataire CAS, et un autre pour les applications tiers.

```
<?php /* ------ exemple de mandataire CAS utilisant phpCAS --------*/
  include_once('CAS.php');
  phpCAS::proxy(CAS_VERSION_2_0,'auth.univ.fr',443,'');
  phpCAS::authenticateIfNeeded();
?>
<html><body>
<p>le login de l'utilisateur est <b><?php echo phpCAS::getUser(); ?></b>.</p>
<?php
  flush();
  // appel du service tiers
  if (phpCAS::serviceWeb('http://test.univ.fr/serviceWeb.php',$err_code,$output)) {
    echo 'Reponse du service tiers : <br>' . $output;
  }
?>
</body></html>
```

**Figure 16 : CAS-ifier une application mandataire CAS [10].**

```
<?php  /* ------ exemple de service CAS utilisant phpCAS --------*/
  include_once('CAS.php');
  phpCAS::client(CAS_VERSION_2_0,'cas.univ.fr',443,'');
  phpCAS::authenticateIfNeeded();

  // pour cet exemple, on retourne une petite phrase qui indique que l'authentification est correcte
  echo '<p>le login utilisateur est <b>' . phpCAS::getUser() . '</b>.</p>';
?>
```

**Figure 17 : CAS-ifier une application tiers [10].**

## 6. Conclusion

Dans ce chapitre nous avons parlé sur le première partie de notre solution qui est le 'CAS', son conception, son architecture, … etc. CAS permet l'implémentation de plusieurs méthodes d'authentification plus ou moins traditionnelles: annuaires LDAP, bases de données, domaines NIS, et fichiers. Cette classe peut être aisément étendue à d'autres méthodes d'authentification, selon les besoins des administrateurs de réseaux universitaire (Novell, Kerberos, Active Directory, ...). Et pour le chapitre suivant nous présenterons la conception de la suite de notre solution qui est 'Shibboleth'.

# Chapitre 3

# *Conception SHIBBOLETH*

## 1. Introduction

À l'origine, le mot hébreu **Shibboleth** permettait à un peuple de distinguer ses ennemis. Ces derniers étaient en effet incapables de prononcer le mot "Shibboleth", le prononçant "Sibboleth". Grâce à ce mot, une confiance pouvait être établie au sein d'une population bien définie. Aujourd'hui, Shibboleth est développé depuis 2001 par Internet2, et désigne à la fois une norme et un produit (open source). 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. Shibboleth a été conçu pour répondre aux besoins des communautés de l'enseignement supérieur et est déjà utilisé dans plusieurs pays : Etats-Unis, Angleterre, Suisse, Finlande, etc [9]. Dans ce chapitre nous présenterons ce mécanisme, ses composants, son fonctionnement avec WAYF et SSO etc.

## 2. SAML (Security Assertion Markup Language)

C'est un standard qui est définit par l'OASIS basé sur le langage XML pour l'échange de données d'authentification et d'autorisation entre les domaines de sécurité distribués. SAML permet aux partenaires de générer des assertions concernant l'identité, les attributs et les droits d'une entité (utilisateur, machine) et, les transférer à d'autres entités (organisations, applications, etc.). Il définit la syntaxe et les règles pour demander, créer, communiquer et utiliser les assertions SAML au format spécifié.

Les assertions SAML, encapsulées dans des messages SOAP et transférées par le biais du protocole HTTP, permettent à la fédération de surpasser les limites imposées par les différences entre les infrastructures déployées chez les différents partenaires.

SAML suppose qu'une entité (utilisateur ou application) possède déjà une identité unique gérée par un IdP qui fournit localement un service pour authentifier cette entité.

SAML ne spécifie pas (et ne s'intéresse pas à) comment le service d'authentification local est Implémenté, c'est de cette manière qu'il fournit une solution à l'hétérogénéité des systèmes d'authentification implémentés chez les différentes organisations [5].

### 2.1 Historique

- **SAML 1.0 :** a été adopté comme un standard OASIS en novembre 2002. Cette version de SAML s'intéressait à l'authentification unique basée sur le web. La Liberty Alliance rajoute à SAML 1.0 une pile de protocoles lui permettant de supporter la jonction de comptes et la fédération à base de rôles : SAML 1.1 est créé et a été considéré comme standard OASIS en septembre 2003 [5].
- **SAML 2.0 :** est standardisé en mars 2005, il unifie les briques de base SSO de SAML 1.0 et l'architecture de fédération d'identité développée par la Liberty Alliance dans SAML 1.1 et les travaux effectués par Internet2 dans leur solution nommée Shibboleth. SAML 2.0 offre une vision standardisée pour la fédération d'identité permettant aux organisations qui l'adoptent d'établir des liaisons entre elles sans se soucier de l'hétérogénéité de leurs infrastructures réseaux déployées [5].

## 3. Choix de solution Shibboleth

SAML 2.0, dans sa conception, fournit des solutions pour l'authentification unique (SSO), et la fédération d'identité. En plus Shibboleth est une extension de SAML2.0 De plus c'est un produit open source, soutenu par une communauté active et ouverte. Ouvrir l'accès à une ressource locale (thèses, cours en ligne) à d'autres Etablissements, Gérer un intranet pour une population disséminée dans plusieurs établissements etc… donc c'est une bonne solution pour notre projet.

## 4. Composants de Shibboleth

Les 3 principaux composants techniques de système Shibboleth sont : le fournisseur de services (SP), le fournisseur d'identité (IDP) et le service de découverte (WAYF) :

### 4.1 Fournisseur de services

C'est une entité proposant des ressources web sur la base d'un contexte de sécurité SAML et sera par la suite nommée SP (Service Provider). Le fournisseur de ressource a en particulier la charge de donner ou non l'accès aux ressources, en fonction des attributs utilisateur [7].

Un SP est composé de trois sous-composants (Figure 18) qui sont appelés : consommateur d'assertions (Assertion Consumer Service), demandeur d'attributs (Assertions Requester), et contrôleur d'accès (Acces Controler) :

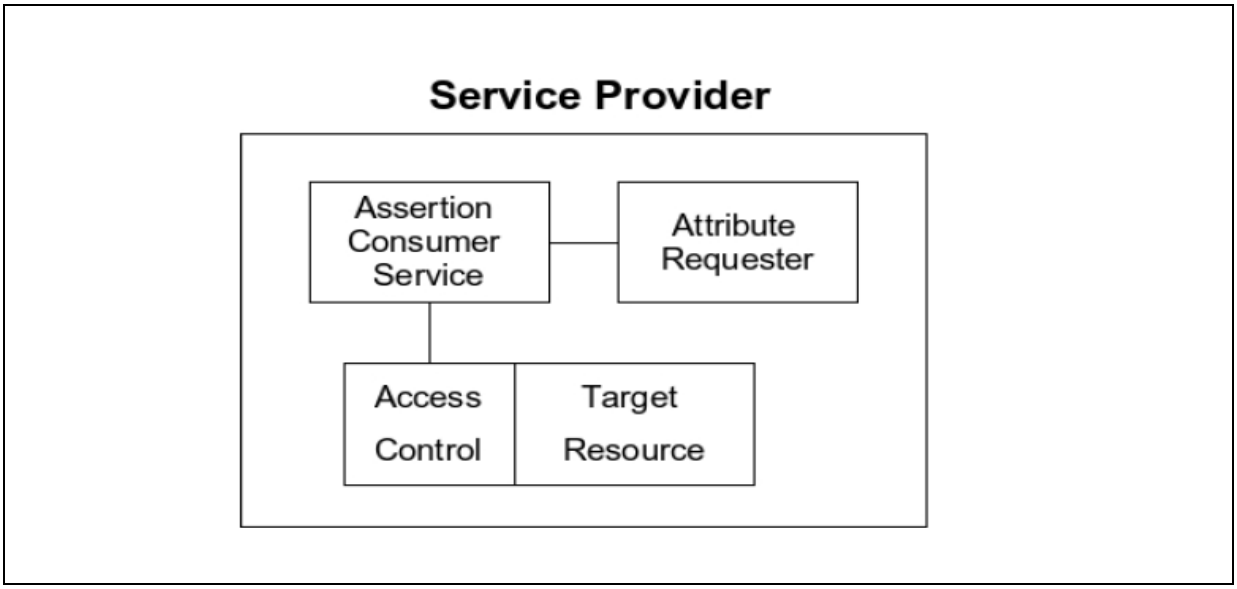



**Figure 18 : Composants de SP [2]**

- **Le consommateur d'assertions :** cet élément agit comme un filtre Web. C'est lui qui redirige vers l'Idp lorsque l'utilisateur n'est pas authentifié. Une fois l'utilisateur est authentifié, alors le consommateur d'assertions transmet le nameIdentifier au demandeur d'attributs [2].
- **Le demandeur d'attributs :** sur la base de l'artefact utilisateur (nameIdentifier), cet élément est chargé de récupérer auprès de l'IDP les attributs de l'utilisateur pour la requête courante. Les différents attributs récupérés sont transmis à un « contrôleur d'accès » [3].
- **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 [2].

## 4.2 Fournisseur d'identités

C'est 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 [2].

Un IDP est composé de 3 sous-composants (Figure 19) appelés : service d'authentification (Authentication Service ou SSO Service), autorité d'authentification (Authentication Authority), et autorité d'attributs (Attribute Authority) :

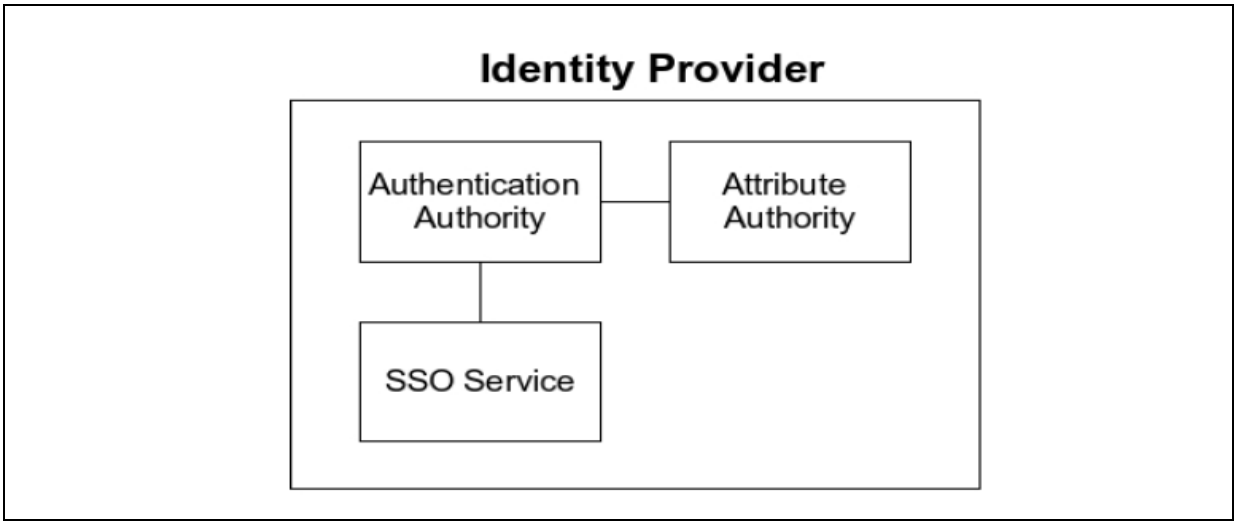



**Figure 19 : Composants d'IDP [2]**

- **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, Single Sign-On) [2].
- **L'autorité d'authentification :** associe le nameIdentifier à l'identifiant de l'utilisateur [2].
- **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…) [2].

### 4.3 Service de découverte (WAYF) :

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 du service SSO de l'IDP [2].

## 5. Assertions SAML de Shibboleth

Dans Shibboleth, des assertions SAML sont transférées à partir des IDPs aux SPs. Les assertions contiennent les déclarations que les SPs peuvent utiliser pour prendre des décisions de contrôle d'accès. Trois types de déclarations sont indiqués par SAML [2] :

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

**Les déclarations d'authentification :** affirment au SP que le principal (souvent un utilisateur) a authentifié avec le IDP en utilisant une méthode particulière d'authentification.

**Les déclarations d'attribut :** fournissent des informations additionnelles au principal de sorte que les SPs 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 SP 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 [2].

## 6. Fonctionnement de Shibboleth avec WAYF et SSO

Nous nous plaçons maintenant dans le cas où un SP est accessible à des utilisateurs rattachées à des établissements différents. Cela est par exemple le cas d'une université souhaitant mettre à disposition de tous les personnels de l'enseignement supérieur de sa région les archives de ses thèses et publications scientifiques.

Le problème qui se pose alors est que le SP ne sait pas vers quel IDP rediriger le navigateur pour l'authentification. Il est résolu grâce au WAYF, dont le rôle est d'orienter les utilisateurs pour sélectionner leur IDP [9].

## 6.1 Premier requête vers un SP

1. Lors de la première requête au SP, celui-ci ne sachant pas quel IDP sera utilisé.
2. Le SP redirige le navigateur vers le WAYF (Figure 20)

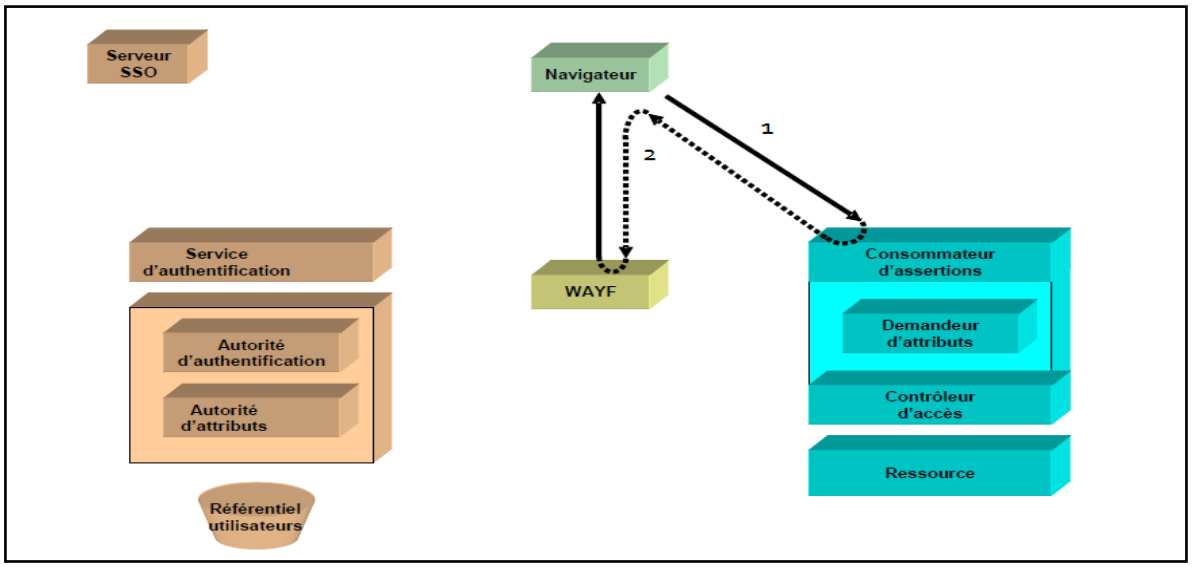



**Figure 20 : Redirection du SP vers le WAYF [9]**

3. Le WAYF affiche alors à l'utilisateur une liste d'IDP possibles.
4. La requête suivante, vers le WAYF, redirige le navigateur vers l'IDP choisi par l'utilisateur.
5. IDP redirige le navigateur vers le serveur SSO.
6. Le serveur SSO propose alors un formulaire d'authentification. (figure 8)

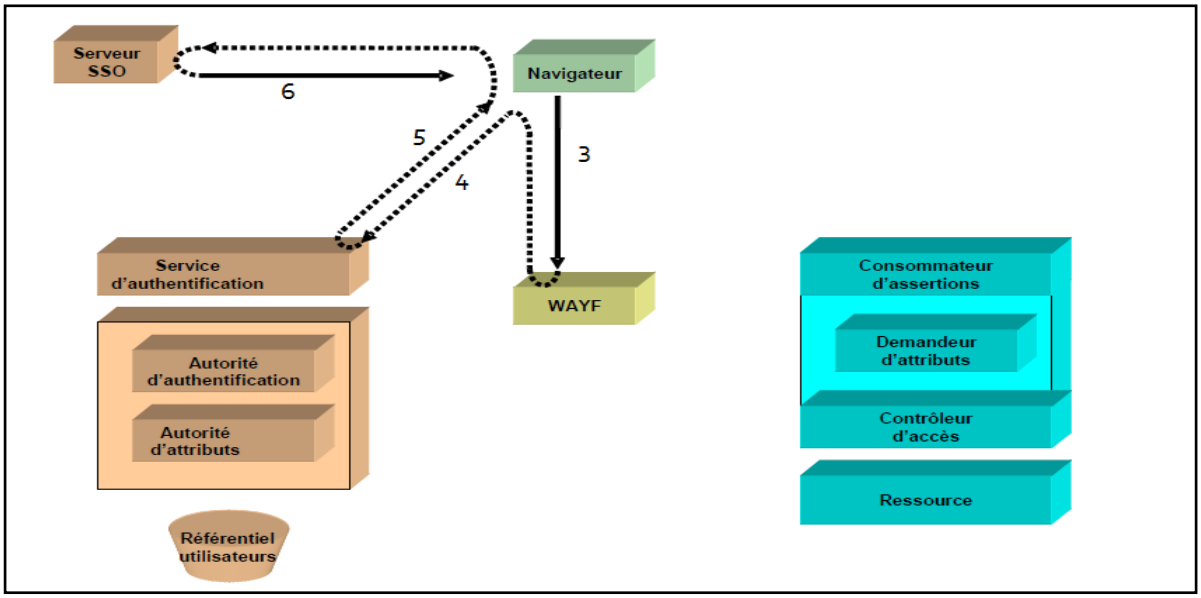



**Figure 21 : Redirection du WAYF vers l'IDP, puis le serveur SSO [9]**

Le navigateur s'authentifie alors auprès du serveur SSO, et l'authentification se déroule ensuite comme montre la Figure (22)

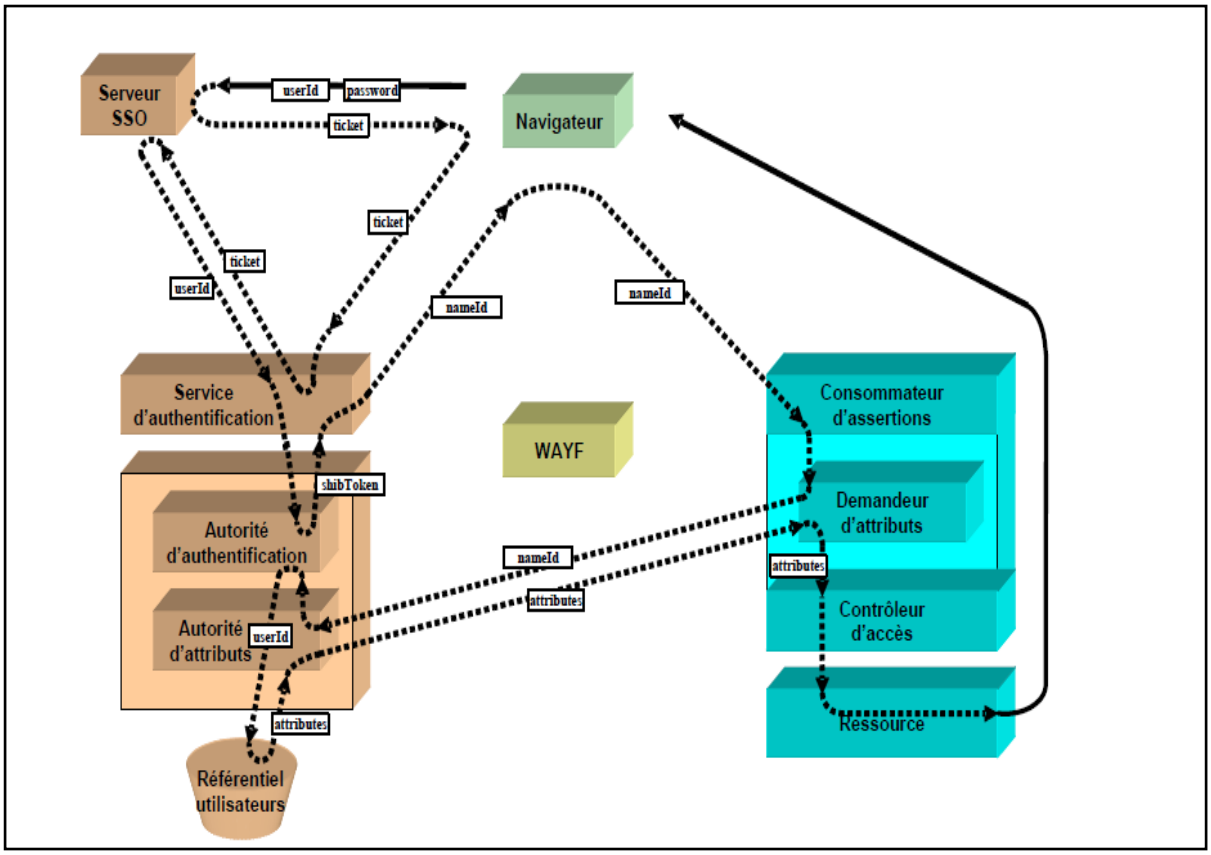



**Figure 22 : Redirection du serveur SSO vers l'IDP, puis le SP [9]**

Pour résumer, du point de vue de l'utilisateur (Figure 23), il :

1. Effectue une requête auprès du SP.
2. Reçoit une demande d'aiguillage du WAYF.
3. Sélectionne son IDP auprès du WAYF.
4. Reçoit une demande d'authentification du serveur SSO.
5. S'authentifie auprès du serveur SSO.
6. Reçoit une réponse du SP, qui autorise ou non l'accès à la ressource demandée.

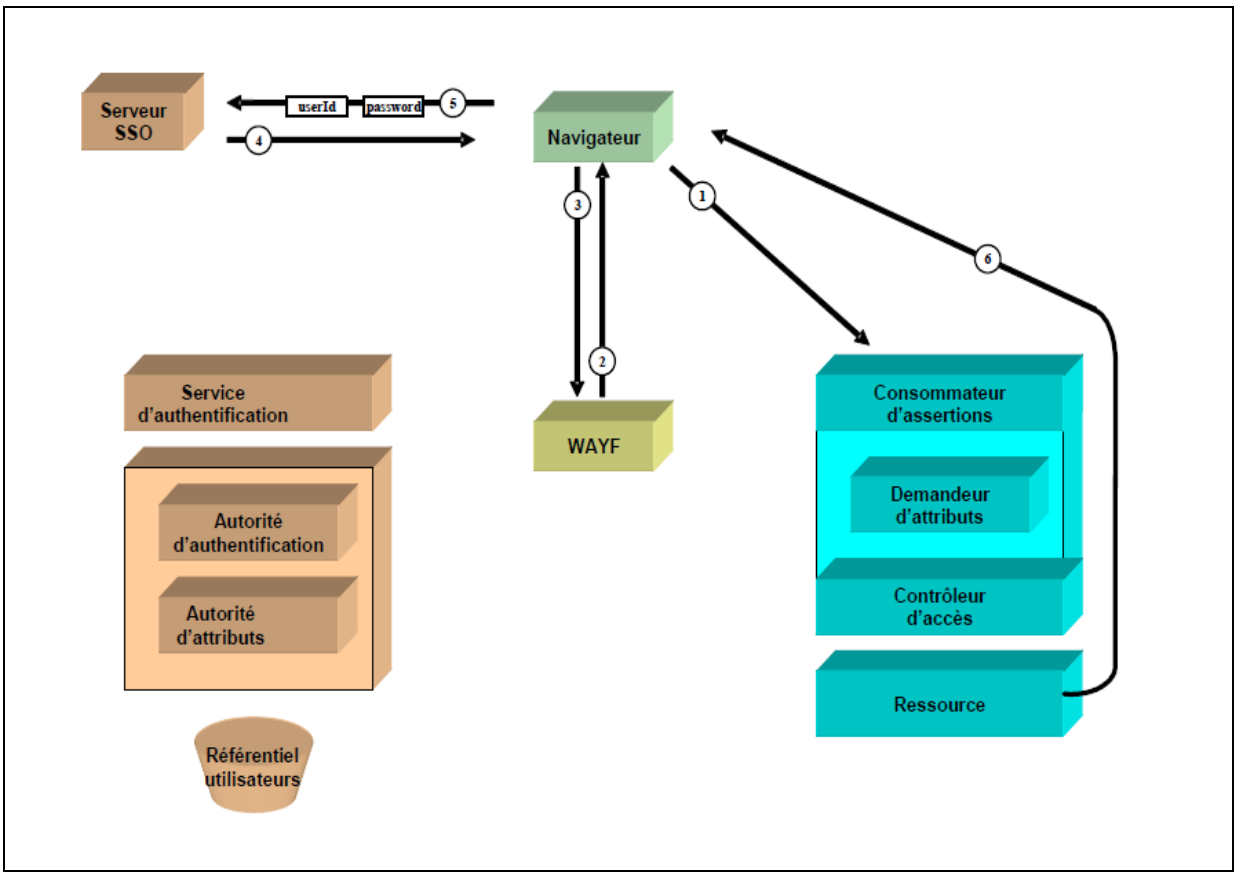



**Figure 23 : Point de vue de l'utilisateur dans un contexte SSO et WAYF [9]**

## 6.2 Requêtes suivantes vers le même SP

Une session étant mise en place entre le navigateur et le SP (le consommateur d'assertions du SP), ni le WAYF, ni l'IDP ni le serveur SSO n'interviennent plus par la suite pour l'accès au même SP (Figure 24) l'utilisateur :

1. Effectue une requête auprès du SP.

2. Reçoit une réponse du SP, qui autorise ou non l'accès à la ressource demandée.

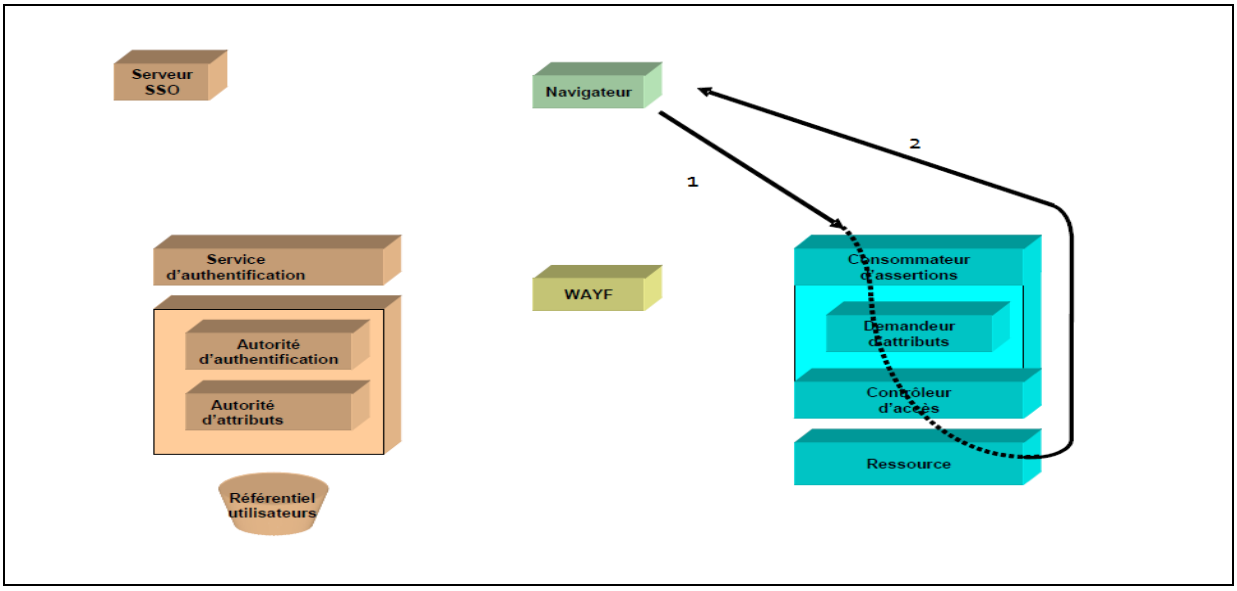



**Figure 24 : Requêtes suivantes vers le même SP dans un contexte SSO et WAYF [9]**

# 7. Fédération de Shibboleth

## 7.1 Méta données

Les Méta données sont composées d'une liste des membres de la fédération (fournisseurs d'identités et fournisseurs de services) d'une part et d'une liste des autorités de certification de confiance d'autre part [9]. La liste des membres est un fichier XML dont chaque enregistrement fournit les informations requises pour chaque entité à savoir :

- L'identifiant du service, sous la forme d'un URN (Uniform Resource Name).
- L'intitulé du service.
- Le contact technique pour le service.
- L'URL des services correspondants : autorité d'authentification et autorité d'attributs pour un fournisseur d'identités, consommateur d'assertions pour un fournisseur de services.

Chaque IDP est décrit par trois éléments propres [2]:

- **<md:IDPSSODescriptor>:** le service d'authentification d'IDP.
- **<md:AuthnAuthorityDescriptor>**: le service autorité d'authentification de SP.
- **<md:AttributeAuthorityDescriptor>**: le service autorité d'attributs d'IDP.

Chaque SP est décrit par un élément propre [2]:

- **<md:SPSSODescriptor>** associé au service consommateur d'assertions du SP.

Les métas 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 [2].

### 7.2 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 du méta donné.

- 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 **AAP (Attribute Acceptance Policy)** est défini dans un fournisseur de service, permettant de filtrer les attributs reçus. [2]

## 8. La sécurité en Shibboleth

La sécurité en shibboleth est renforcée et bien présentée :

- L'authentification utilisateur, peut être renforcée, par l'usage de certificats X.509 clients, par exemple en connectant CAS à Shibboleth.
- Les SP importent les certificats serveurs des IdP qu'ils autorisent à leur fournir une authentification utilisateur. Ainsi, le cercle de confiance est bien défini et sécurisé : Les SP n'acceptent pas de n'importe quel IdP une authentification utilisateur.

- Les IdP peuvent aussi importer les certificats des SP. Ainsi, seuls les SP autorisés peuvent demander une authentification utilisateur.
- Les assertions sont chiffrées par des clés asymétriques (cf. WS-Encryption).
- Les assertions sont signées par les certificats serveurs, des IdP pour les SP et des SP pour les IdP (cf. WS-Signature).
- Chaque donnée d'utilisateur dans les assertions à destination des SP peut être chiffrée et signée, indépendamment de l'assertion, par le certificat serveur de l'IdP.
- Des mécanismes peuvent être configurés dans le but de présenter à l'utilisateur l'ensemble des données le concernant à destination du SP, pour validation, avant que l'IdP ne les transmette. Ainsi, l'utilisateur est maître des données le concernant qu'il veut transmettre aux services distants [16].

## 9. Conclusion

Shibboleth est un produit complet et source ouvert, une solution de fédération d'identité et aussi une solution de web-SSO interne à notre université. La topologie de fédération basée sur Shibboleth convient de l'environnement de travail collaboratif qui rassemble des plates-formes différentes. Shibboleth résout les problèmes de sécurité liée à la circulation des mots de passe et on évite la multiplication des comptes d'un même utilisateur dans plusieurs universités. Il à la capacité de contrôler l'accès d'utilisateur grâce au pré filtre sur les attributs qui sont défini pour chaque ressource dans SP. Cette solution est une sécurité renforcée car Il se base sur le standard sécurisé SAML avec SSL. Le chapitre suivant sera consacré à la mise en œuvre de notre solution permettant de réalises l'objectif défini au début.

# Chapitre 4

# *Mise en ouvre*

## 1. INTRODUCTION

Une fois la méthodologie du travail décrit, il ne nous reste plus qu'à mettre sur pied notre système .Ainsi, dans ce paragraphe, il est question de présenter étape par étape le travail effectué et d'accompagner ces déférentes étapes par des résultats qui seront présentés par des captures d'écran.

## 2. LA MISE EN ŒUVRE

### 1. Présentation de l'intranet

L'intranet est la partie sécurisée de notre réseau informatique basé sur la même technologie qu'internet. L'intranet est connecté au réseau internet pour permettre la communication avec le monde extérieur. Et bien sur lorsque on dit intranet dans notre projet on veut dire le campus universitaire UKMO avec ses services web (Zimbra, Joomla, Dockeos).

### 2. Vue générale de l'implémentation

Notre travail pratique consiste à implémenter un contrôleur de domaine **SAMBA**, le synchroniser avec l'annuaire **LDAP** et ensuite gérer la centralisation de l'authentification avec ce dernier. En effet, plusieurs solutions ont été parcourues en fonction des systèmes d'exploitation Linux (**CentOS**) à la recherche du succès et la solution **Shibboleth-CAS** a été favorable.

Nous avons choisi « **univ-ouargla.dz** » comme domaine.

L'architecture simplifiée de notre système se présente comme suite :

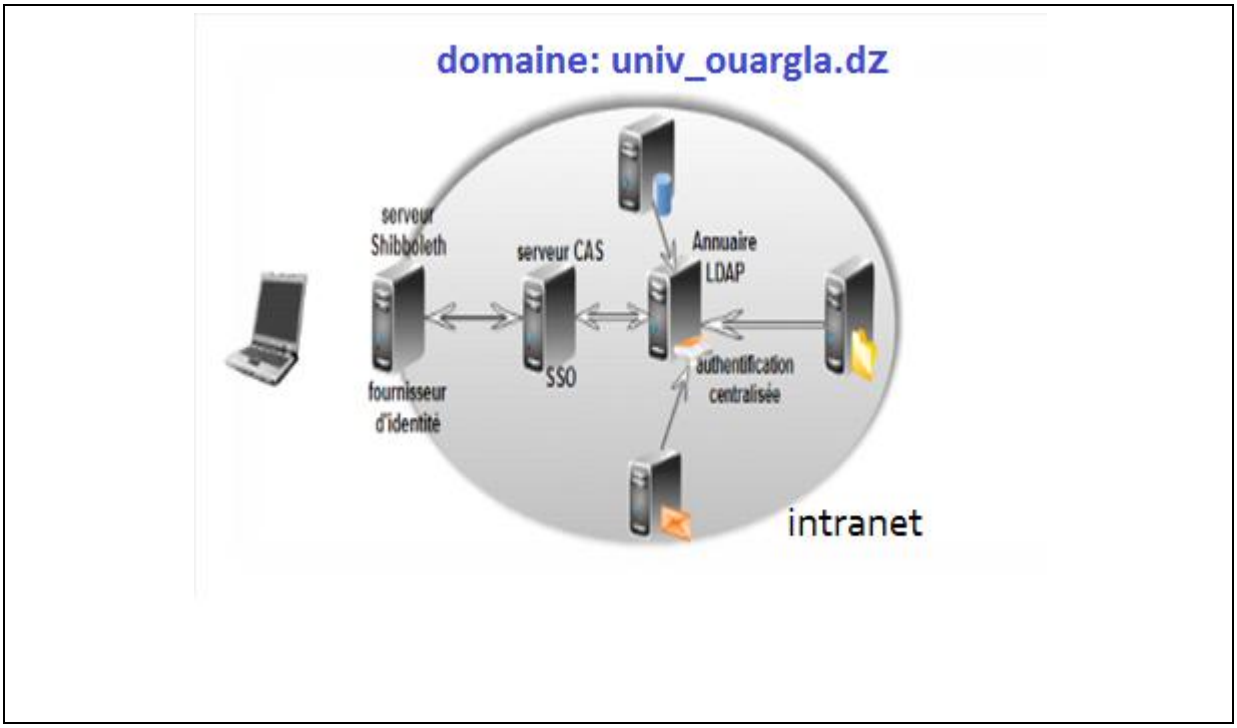



**Figure 25 : vue générale de notre implémentation**

## 3. Pourquoi choisir Centos Enterprise linux ?

. CentOS est une version axée sur la collectivité de Linux Red Hat Enterprise (RHEL). Il est très stable et compatible, ce qui est très souhaitable dans les environnements professionnels.

Voici quelques avantages de l'utilisation de Centos :

- Basé sur une solution commerciale de qualité supérieure.
- Stable et cohérent.
- Eh bien testé avant de le rendre disponible au public [25].

## 4. Les services web de campus UKMO

- **Zimbra :** est une plateforme complète de messagerie et de collaboration open source. Au menu : e-mail, contacts, calendrier, gestion de documents, partage de fichiers, liste de tâches, médias sociaux, mais aussi synchronisation avec d'autres postes de travail et périphériques mobiles [24].

- **Joomla :** est un système de gestion de contenu primé (CMS), qui vous permet de construire des sites Web et de puissantes applications en ligne. [google tra]

- **Dokeos :** est une plate-forme d'apprentissage en ligne Le logiciel s'appuie sur une architecture multilingue qui lui permet de supporter 34 langues. écrit en PHP, il utilise le SGBDR MySQL. Dokeos est aussi un réseau de sociétés de services qui fournissent du conseil et d'autres services : conseil, développement, formation, notamment auprès de grandes entreprises et des administrations publiques [19].

## 5. Configuration des serveurs

### 5.1. Installation et configuration de l'annuaire LDAP

- **Installation des packages nécessaire :**

```
[root@univ]# yum -y install openldap openldap-clients openldap-servers
```

- **Configuration de base :**

  – Le paquet «slapd» contient avant tout le démon qui permet de démarrer/arrêter/redémarrer le serveur OpenLDAP.

  – Gestion de la base de donnée : La gestion de la base de donnée va permettre de préciser plusieurs choses :

    - Le nom (suffixe) de la base de données
    - L'identité (DN) du gestionnaire de la base
    - L'endroit où seront stockés les différents fichiers représentant les données de l'annuaire.

**a) Le suffixe de la base :** c'est en quelque sorte l'identifiant générale de la base de données. Toutes les entrées de la base contiendront ce suffixe.

Il est définit ainsi : **dc=univ_ouargla ,dc=dz**

**b) Le gestionnaire de la base :** c'est une entrée spéciale de la base. Elle peut être virtuelle. Elle est gérée par la ligne **rootdn**. La solution la plus simple consiste à utiliser une forme en fonction du choix du suffixe.

*rootdn « **cn=admin, dc= univ_ouargla , dc=dz** »*

– Le gestionnaire de la base doit se connecter à l'aide d'un mot de passe ; celui-ci est décrit par la ligne suivante

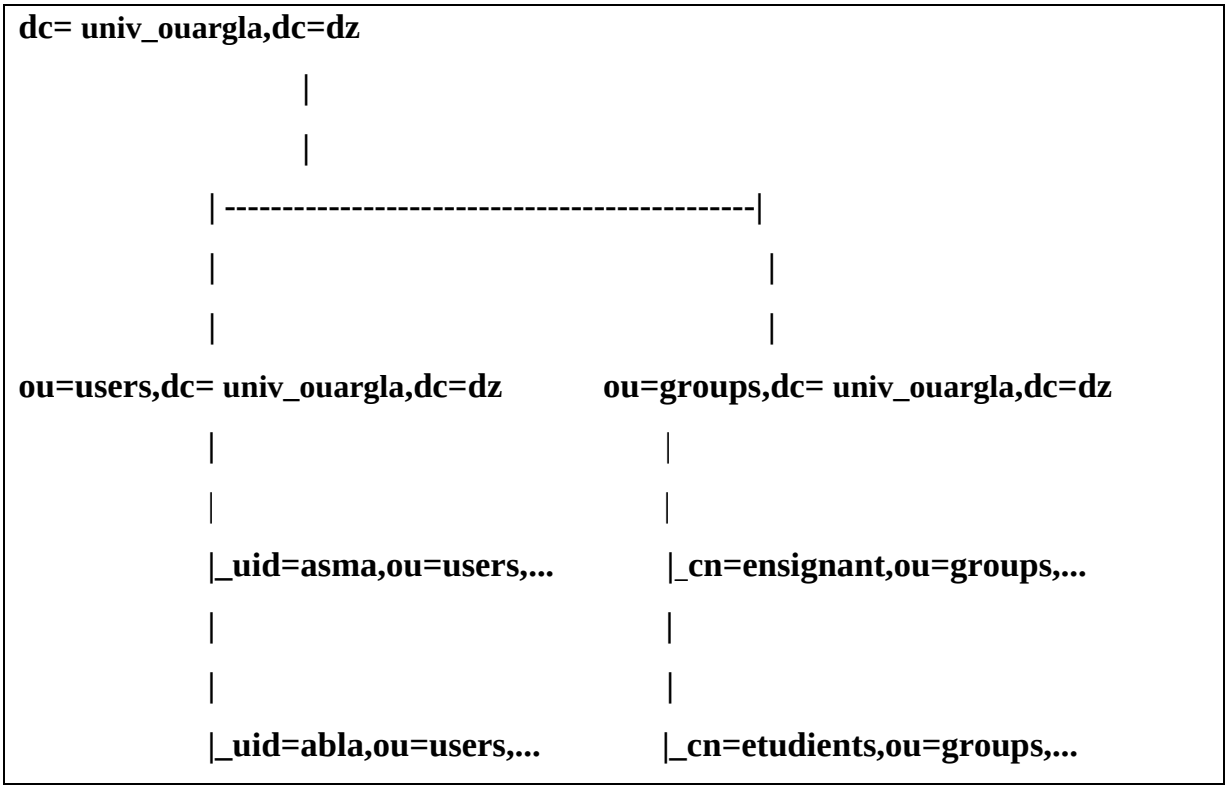
*rootpw secret*

– Pour terminer l'installation, il suffit de répondre aux questions posées par le système pour la configuration du démon slapd.

```
# See slapd.conf(5) for details on configuration options.
# This file should NOT be world readable.
#
include /usr/local/etc/openldap/schema/core.schema
include /usr/local/etc/openldap/schema/inetorgperson.schema
include /usr/local/etc/openldap/schema/cosine.schema
include /usr/local/etc/openldap/schema/nis.schema

pidfile /var/run/slapd.pid
argsfile /var/run/slapd.args

access to attr=entry filter=(organizationalStatus=parti)
by dn="cn=admin,dc=univ_ouargla.dz" read
by dn="cn==admin,dc=univ_ouargla.dz " write
by * none

access to * by * read

rootdn can always write!

database bdb
suffix "dc= univ_ouargla ,dc=dz"
rootdn "cn=admin, dc= univ_ouargla, dc=dz"

rootpw secret

directory /var/db/openldap-data

index default pres,eq
index objectClass

indexcn, s ,mail eq, sub, approx
```

- **La représentation hiérarchique des données :** LDAP organise les données de manière hiérarchique dans l'annuaire. Ceci signifie que toutes les informations découlent d'une seule et même "racine".

Création de la structure de l'annuaire :




- **Fichier .ldif :** Le format LDIF (Ldap Data Interchange Format) est un format standard définit par la RFC2849 qui permet de représenter sous forme de fichier texte les donnés présentes dans un annuaire. Il offre également une syntaxe qui permet de faire des modifications dans l'annuaire [26].

– Le fichier **structure.ldif** au format LDIF contient :

dn: **dc= univ_ouargla,dc=dz**

objectClass: dcObject

objectClass: organization

**dc= univ_ouargla**

o: **univ_ouargla**

description: **univ_ouargla**

dn: ou=users, **dc= univ_ouargla,dc=dz**

objectClass: top

objectClass: organizationalUnit

ou: users

dn: ou=groups, **dc= univ_ouargla,dc=dz**

objectClass: top

objectClass: organizationalUnit

ou: groups

dn: cn= **ensignant**,ou=groups, **dc= univ_ouargla,dc=dz**

- **Installation phpldapadmin :**

**[root@univ ~]# yum install phpldapadmin –y**

Loaded plugins: fastestmirror

Loading mirror speeds from cached hostfile

* base: centos.ipserverone.com

* epel: ftp.cuhk.edu.hk

* extras: centos.ipserverone.com

* updates: centos.maulvi.net

Setting up Install Process

Resolving Dependencies

--> Running transaction check

---> Package phpldapadmin.noarch 0:1.2.2-1.el6 will be installed

--> Processing Dependency: php-ldap for package: phpldapadmin-1.2.2-1.el6.noarch

--> Running transaction check

---> Package php-ldap.i686 0:5.3.3-3.el6_2.6 will be installed

--> Finished Dependency Resolution

– **Pour la configuration de phpldapadmin :**

```
#
#  Web-based tool for managing LDAP servers
#
Alias /phpldapadmin /usr/share/phpldapadmin/htdocs
Alias /ldapadmin /usr/share/phpldapadmin/htdocs
<Directory /usr/share/phpldapadmin/htdocs>
  Order Deny,Allow
  Deny from all
  Allow from 127.0.0.1 192.168.1.0/24
  Allow from ::1
</Directory>
```

– **Redémarrer le service httpd :**

```
[root@univ~]# /etc/init.d/httpd restart
Stopping httpd:                                [  OK  ]
Starting httpd:                                [  OK  ]
```

## 5.2. Configuration DNS

- **Installer BIND :**

```
[root@univ ~]# yum -y install bind bind-utils
```

- **Bind :** est le serveur DNS le plus utilisé sur Internet spécialement sur les systèmes de type UNIX [23].

- **Bind-utils :** il contient une collection d'utilitaires pour interroger les serveurs DNS (Domain Name System) de noms pour trouver des informations sur les hôtes Internet.

Ces outils vont vous fournir les adresses IP des noms d'hôte donné, ainsi que d'autres informations à propos de domaines enregistrés et les adresses réseau [20].

- La configuration du DNS est comme suit. On Modifie et on ajoute les entrées et on définit les zones. vi /etc/named.conf.

```
zone "." IN {
type hint;
file "named.ca";
};
zone "univ_org.dz" IN {
type master;
file "univ_ouargla.dz.zone";
allow-update { none; };
};
zone "1.168.192.in-addr.arpa" IN {
type master;
file "1.168.192.zone";
allow-update { none; };
};
include "/etc/named.rfc1912.zones";

include "/etc/named.root.key";
```

## 5.3. Installation et configuration de SAMBA

- **Serveur SAMBA :** L'utilisation de SAMBA comme simple lien entre les 2 systèmes d'exploitation permet simplement de partager des ressources (comme les imprimantes par exemple) et aussi de partager des fichiers et des répertoires. Et en tant que contrôleur principal de domaine permet quant à lui d'inscrire une machine sur un domaine du type Windows NT et de gérer ainsi les connexions aux serveurs (accès aux comptes, etc ...) [21].

- **Pour l'installation du Serveur SAMBA CentOs sur la machine avec la commande**

```
[root@univ~]# yum –y  install samba
```

- **Joindre le serveur SAMBA à l'annuaire LDAP :**

  LDAP fonctionne avec des schémas, par défaut 4 schémas sont déjà présents, pour utiliser samba avec LDAP il faut le schéma approprié. Celui-ci se trouve dans le paquet samba-doc. Il reste maintenant à éditer le fichier de configuration du serveur OpenLDAP :

```
[root@univ~]# vi /etc/openldap/slapd.conf
```

```
include /etc/openldap/schema/core.schema

include /etc/openldap/schema/cosine.schema

include /etc/openldap/schema/inetorgperson.schema

include /etc/openldap/schema/nis.schema

include /etc/openldap/schema/samba.schema

allow bind_v2

pidfile /var/run/openldap/slapd.pid

argsfile /var/run/openldap/slapd.args
```

## 5.4. Installation de PHP et MySQL open SSL

- **PHP** et **MySQL** sont les technologies de référence pour mettre en ligne des sites web à contenu dynamique comme les plateformes (Zimbra, Dockeos, Joomla)

- **MySQL :** permet de stocker et ranger des éléments dans une base de données.PHP fait le lien entre le serveur web et MySQL.
- **Open SSL :** Boite à outils de chiffrement comportant deux bibliothèques (une de cryptographie générale et une implémentant le protocole SSL), ainsi qu'un programme en ligne de commande permettant de réaliser de nombreuses opérations cryptographiques pour l'authentification. Cet utilitaire implémente les protocoles SSL et TLS [22].

```
[root@univ~]# yum install -y openssl*  php*  MySQL*
```

## 5.5. Installation Apache, java JDK and TOMCAT

- **Apache :** Apache est un serveur Web gratuit fonctionnant sous Linux et Windows NT.

```
[root@univ~]# yum –y  install  apache2
```

- **Java JDK :**

```
[root@univ~]# yum -y install  openjava-6-jdk
```

- **Tomcat :** Apache Tomcat est un serveur d'applications JAVA. C'est lui qui exécutera la brique Shibboleth IdP. Apache Tomcat 6.0.17 ou plus avancé est la version exigée pour démarrer le fournisseur d'identités [7].

```
[root@univ~]# yum -y install  tomcat6
```

## 5.6. Installation st configuration de l'IdP et SP de Shibboleth

- **Installation d'Idp :**

```
[root@univ~]#cd /tmp
```

```
[root@univ~]# wget http://shibboleth.net/downloads/identity-provider/2.3.8/ shibboleth-identityprovider 2.3.8-bin.zip

[root@univ~]# tar shibboleth-identityprovider-2.3.8-bin.zip
```

- **Installation de la brique Shibboleth SP :**

Vous allez télécharger un ensemble de fichiers de configuration personnalisés pour les besoins du l'implémentation à l'adresse : https://test.federation.renater.fr/exemples/conf_sp2_renater.tar.gz.

```
[root@univ~]#cd /tmp

[root@univ/tmp]#wget https://test.federation.renater.fr/exemples/conf_sp2_renater.tar.gz

[root@univ/tmp]#tar -zxvf conf_sp2_renater.tar.gz
```

### 5.7. Installation et configuration de CAS server

- **Installation de CAS server :**

```
[root@univ~]# cd /tmp

[root@univ/tmp]#wgethttp://www.ja-sig.org/downloads/cas/cas-server-3.3.5 release.tar.gz

[root@univ~]# tar -xvzf cas-server-3.3.5-release.tar.gz
```

## 4. CONCLUSION

Dans ce chapitre nous avons présenté l'environnement de notre implémentation, nous définissons les logiciels et les packages qui nous avons utilisé pour réaliser notre pratique, ensuite définir les commandes d'installation et de la configuration de chaque package utilisé.

# Conclusion Générale

L'authentification SSO améliore directement la qualité du service rendu aux personnes (étudiants, enseignants), Afin d'éliminer un grand fardeau En rappelant les noms utilisateurs et mots de passe pour accéder à différents services dans notre université de manière facile et en douceur.

La transmission d'identités et d'attributs entre universités, comme elle est envisagée dans Shibboleth, offre des nouvelles perspectives de coopération entre notre université et les autres universités algériennes, et pour quoi pas des universités étrangères pour bénéficier de partage des ressources en ligne, des formations , des cours, des conférences …etc. Cela augmente l'importance de notre université et les services qu'elle fournit, Ainsi l'augmentation du nombre de visiteurs et les utilisateurs de ces services, Cela est d'autant plus positive pour la position de notre université en termes de classement national, et pourquoi pas arabe, continental et mondial.

Grâce à ce projet, nous avons pu réaliser une partie de cette solution, nous n'avons pas eu le temps nécessaire pour réaliser tout le travail, le système Shibboleth est facile à installer mais son utilisation est très complexe, il nécessite des formations spécialisées et des compétences pour sa maitrise entière. Nous n'avons pas trouvé, à notre connaissance un organisme Algérien qu'il utilise. Malgré tout ça, nous avons fait tout notre possible pour le comprendre, l'installer et le configurer et nous espérons le maitriser entièrement dans l'avenir.

En perspectives, nous espérons dans le futur proche, compléter la réalisation de la mise en place de Shibboleth dans notre université et le généraliser à toutes les universités algériennes. Ce domaine de sécurité et d'authentification dans le milieu des services web universitaires nous intéresse beaucoup, et nous espérons pouvoir continuer de travailler dans ce domaine dans l'avenir, professionnellement et dans les entreprises.

# Bibliographie


**[1]** Cert-IST, Single Sign-On (SSO) – 1ère partie, avril 2010.

**[2]** DANG Quang Vu, Support de sources d'authentification multiples dans un portail de travail collaboratif, Institut National des Télécommunications, Hanoi, Novembre 2006.

**[3]** Jean-marie thia et Philippe Beraud, Kit de démarrage Extension Web SSO Fédéré Shibboleth Pour Les Technologies SharePoint, Microsoft France, UMPC, septembre 2009

**[4]** META-ANNUAIRE, Présentation de la solution CAS de Web SSO du COE, Direction Générale de l'Administration, Direction des Technologies de l'information, 2009

**[5]** Michel KAMEL, Patrons organisationnels et techniques pour la sécurisation des Organisations Virtuelles, UNIVERSITÉ DE TOULOUSE, 2008

**[6]** M. Wahl, T. Howes, et S. Kille. RFC 2251 - Lightweight Directory Access Protocol (v3). Technical report, Internet Engineering Task Force, December 1997

**[7]** Narcisse Kapdjou et Eric Marc Modo Nga , Mise en oeuvre système d'authentification centralisé SSO avec fournisseur d'identités, Université de Dschang/iut-fv de Bandjoun, 2012.

**[8]** Olivier Salaün , Introduction aux architectures web de Single Sign-on, Comité Réseau des Universités Campus de Beaulieu – Rennes , 15 Octobre 2003 .

**[9]** Olivier Salaün, Pascal Aubry et Florent Guilleux, Fédération d'identités et propagation d'attributs avec Shibboleth, Comité Réseau des Universités et IFSIC –Université de Rennes 1, 2005.

**[10]** Vincent Mathieu, Pascal Aubry, Julien Marchal, Single Sign-On open source avec CAS (Central Authentication Service), Université de Rennes 1, 2003.

**[11]** http://www.yale.edu/tp/auth/.

**[12]** http://www.esup-portail.org/consortium/espace/SS0_1B/

**[13]** http://wwwigm.univmlv.fr/~dr/XPOSE2009/Sign%20Sign%20On/aetI.html#avantage

**[14]** http://www.rediris.es/app/papi/index.en.html.

**[15]** http://www.pubcookie.org/.

**[16]** http://www.alcyonix.com/articles/shibboleth-un-sso

**[17]** http://www.commentcamarche.net/contents/authentification/sso.php

**[18]** www.esup-portail.org/.../espace/.../cas-jres2003-article.pdf

**[19]** http://fr.wikipedia.org/wiki/Dokeos

**[20]** http://translate.google.dz/translate?hl=fr&sl=en&u=https://apps.fedoraproject.org/packages/bind-utils&prev=search

**[21]** http://killbert.free.fr/linux/serveur_linux/serveur_samba.htm

**[22]** http://www.infonitec.com/definition-informatique-telecom/definition nformatique-telecom.php?id=1434

**[23]** http://fr.wikipedia.org/wiki/BIND

**[24]** http://zimbra.yaziba.net/zimbra/caracteristiques

**[25]** http://www.blog-nouvelles-technologies.fr/12898/choisir-la-meilleure-distribution-linux-pour-un-serveur-web/

**[26]** http://wawadeb.crdp.accaen.fr/iso/tmp/ressources/cvs.orion.education.fr/homepages/docbooks/ldap/ldif.html