

REPUBLIQUE DU SENEGAL

Un Peuple – Un But – Une Foi

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR DE LA RECHERCHE ET DE L'INNOVATION

UNIVERSITE GASTON BERGER DE SAINT-LOUIS



U.F.R DE SCIENCES APPLIQUEES ET DE TECHNOLOGIE

DEPARTEMENT INFORMATIQUE

Mémoire de fin d'études pour l'obtention du diplôme de

Master en Réseaux, Sécurité et Systèmes Distribués (R2SD)

Sujet : Etude et Mise en place d'une solution d'authentification unique (SSO) au Ministère de l'Education Nationale (MEN)

Présenté par :	Maître de stage :	Encadrant Pédagogique :
Dioubayrou Ka	M. Malick FAM	Pr. Chérif Diallo

Année universitaire 2020/2021

DEDICACES

Je dédie ce travail à nos chers parents :

Aucune dédicace ne saurait exprimée notre respect, notre amour éternel et notre considération pour les sacrifices que vous avez consentis pour notre instruction et notre bien-être.

Nos dédicaces vont également à l'endroit de notre charmante épouse et notre fille :

Vous qui m'avez toujours soutenu, merci de votre confiance et d'être pour moi une source de motivation.

REMERCIEMENTS

Nous ne saurions amorcer la rédaction de ce mémoire sans remercier ALLAH le tout puissant, qui a toujours guidé nos pas vers le chemin de la connaissance, par sa protection et sa bénédiction. Nous exprimons notre gratitude et nos sincères remerciements :

A notre maître de stage M. Malick FAM, administrateur système et réseaux (IT MEN) pour son soutien professionnel constant, sa disponibilité, ses encouragements et ses multiples recommandations qui ont été indispensables à la réalisation de ce travail.

A notre encadrant pédagogique, Pr. Chérif Diallo pour son soutien moral et scientifique, sa disponibilité et ses conseils.

Aux membres du jury qui ont accepté d'évaluer ce modeste travail.

A M. Pape Khole Dioh, administrateur système et réseaux au MEN, de par sa disponibilité, son soutien, ses conseils et recommandations.

A M. Seyni Ndiaye Fall, coordonnateur du SIMEN de m'avoir accueilli dans son service.

A l'ensemble du corps professoral de l'UFR SAT de l'UGB, particulièrement ceux de la section Informatique, de par les efforts que vous avez fournis pour nous accompagner durant toute notre formation.

A toute l'équipe du SIMEN qui nous a accompagné tout au long de cette expérience et qui a eu la gentillesse de faire de ce stage, un moment très profitable.

A notre grand frère Bara Ka, Doctorant à l'UGB, de m'avoir accueilli à l'université, soutenu dans les moments les plus difficiles.

A nos amis Bernard Diatta et Adrien Diatta, pour la correction.

A nos parents, notre épouse et notre fille pour leur compréhension, leurs prières, leur patience et leur soutien sans faille.

A toute les membres de notre famille.

A nos amis et camarades de classe.

A tout ceux qui ont participé de près ou de loin à la réussite de ce travail.

AVANT-PROPOS

L'UGB, l'Université Gaston Berger de Saint-Louis du Sénégal a été officiellement créée par la loi n° 90-03 du 02 Janvier 1990 et est dotée d'un statut définitif par le décret n° 096-597 du 10 Juillet 1996, d'un nom par le décret n°096-1016 du 04 Décembre 1996. Se trouvant à une dizaine de kilomètres de la ville de Saint-Louis, l'université est composée de l'Institut polytechnique de Saint-Louis et de huit unités de formation et de recherche que sont :

- Lettres et sciences humaines
- Sciences juridiques et politiques
- Sciences économiques et de gestion
- Sciences de la santé
- Sciences agronomiques, d'aquaculture et de technologie alimentaire
- Civilisations, religions, arts et communication
- Sciences Appliquées et de technologie

L'unité de formation et de recherche des sciences appliquées et de technologie (UFR SAT) est composée de trois sections : Mathématiques, Physique appliquée et Informatique.

Dans la section informatique sont dispensés des cours de la Licence 1 au Master 2 en informatique pour une polyvalence dans les métiers du numérique. La formation en master dure deux années et est sanctionnée par un mémoire de fin d'études soutenu devant un jury.

C'est dans cette optique que pour l'obtention du diplôme de master en informatique spécialité réseaux, sécurité et systèmes distribués (**M2 R2SD**), les étudiants doivent présenter un mémoire à la fin de la formation ou un rapport de stage. C'est dans ce dessein que nous avons intégré SIMEN, service dans lequel nous sommes chargés de mener un projet informatique dont la teneur est consignée dans ce rapport.

RESUME

La multiplication et la diversité des systèmes d'authentification liées aux ressources numériques devient de plus en plus un enjeu majeur pour les entreprises et les administrations. En effet, chaque application est soumise à une procédure de contrôle d'accès spécifique.

Ces opérations à réaliser peuvent nécessiter un contrôle d'accès et des droits associés entrainant parfois une confusion pour l'utilisateur. En guise d'exemple, la perte ou l'oublie des mots de passe.

Une solution est l'utilisation de l'authentification unique (SSO) qui permet à un utilisateur d'avoir accès à l'ensemble des ressources du réseau auxquelles il a droit, en s'authentifiant qu'une seule fois. L'utilisateur disposera alors d'une session SSO valide pour l'ensemble des applications ou ressources protégées suivant ses droits d'accès. De plus, un SSO améliore la sécurité, facilite la gestion des comptes utilisateurs et l'ergonomie du système d'information.

Le présent mémoire consiste à étudier et mettre en place une solution d'authentification unique via l'application open source, Keycloak pour répondre aux objectifs de notre structure.

Mots clefs : Keycloak, Openid Connect, OAuth 2, Sécurité, WebSSO, Authentification unique, Identité, Contrôle d'Accès, Autorisation, Token

SIGLES ET ABREVIATIONS

IT : Information Technology

MEN : Ministère de l'Education Nationale

SSO : Single Sign On

SIMEN : Système d'Information et de Management de l'Education Nationale

HTTP : Hypertext Transfer Protocol

HTTPS : Hypertext Transfer Protocol Secure

SSL : Secure Sockets Layer

TLS : Transport Layer Security

IdP : Identity Provider

SP : Service Provider

SMTP : Simple Mail Transfer Protocol

CAS : Central Authentication Service

SAML2 : Security Assertion Markup Language 2.0

OAuth 2 : Open Authorization 2.0

OIDC : OpenID Connect

LDAP : Lightweight Directory Access Protocol

AD : Active Directory

IEN : Identifiant de l'Education Nationale

TIC : Technologies de l'Information et de la Communication

AAA : Authentification, Authorization, Accouting

API : Application Programming Interface

AWS : Amazon Web Service

PHP : HyperText Pre-processor

SaaS : SSO as a Service

CSRF : Cross Site Request Forgery

SAT : Sciences Appliquées et de Technologies

UFR : Unité de Formation et de Recherche

UGB : Université Gaston Berger de Saint-Louis

WWW : World Wide Web

XSS : Cross-Site Scripting

URL : Uniform Resource Locator

FS : Fournisseur de Service

FI : Fournisseur d'Identité

ST : Service Token

OTP : One Time Password

XML : eXtensible Markup Language

TGC : Token Granting Ticket

MIME : Multipurpose Internet Mail Extensions

LDIF: LDAP Data Interchange Format

IANA : Internet Assigned Numbers Authority

OU : Organizational Unit

OASIS : Organization for the Adevancement of Structured Information Standards

URI : Uniform Resource Identifier

IETF : Internet Engineering Task Force

JWT : Json Web Token

LISTE DES FIGURES et DES CAPTURES

Figure 1 : Organigramme du SIMEN 4
Figure 2 : Architecture de l'infrastructure existante du MEN 6
Figure 5 : Diagramme de séquence de l'authentification avec Open Id Connect 19
Figure 6 : Diagramme de séquence de l'accès SAML2 23
Figure 7 : Diagramme de séquence de l'authentification CAS 24
Figure 3 : Schéma d'une requête HTTP 25
Figure 4 : Principe d'une attaque par fixation de session 30
Figure 8 : Protocoles et environnements compatibles avec Evidian 44
Figure 9 : Diagramme de séquence de l'authentification avec Evidian 44
Figure 10 : Single Sign-On de Systancia 46
Figure 11 : Diagramme de séquence de l'authentification de Keycloak 47
Figure 12 : Architecture globale de Lemonldap-ng 49
Figure 13 : Diagramme de séquence de l'authentification avec Lemonldap-ng 50
Figure 14 : Cas d'utilisation de AWS SSO 53
Figure 15 : Diagramme de séquence de l'authentification avec CAS 55
Figure 16 : Diagramme de séquence de l'authentification avec Keycloak 60
Figure 17 : Architecture de base de Docker 62
Figure 18 : Architecture mode autonome 63
Figure 19 : Architecture finale après déploiement de la solution 68

Capture 1 : Etat des conteneurs (Nginx, instances Keycloak) Docker 77
Capture 2 : Etat de Keycloak après l'arrêt d'une de ses instances 77
Capture 3 : Accès à la console d'administration 78
Capture 4 : Configuration authentification 2 Facteur pour l'administrateur 78
Capture 5 : Code à usage unique pour l'administrateur 79
Capture 6 : Création de domaine Keycloak 79
Capture 7 : Console d'administration de Keycloak 80
Capture 8 : Enregistrement d'un client 84
Capture 9 : Synchronisation avec AD 86

Table des matières

DEDICACES i
REMERCIEMENTS ii
AVANT-PROPOS iii
RESUME iv
SIGLES ET ABREVIATIONS v
LISTE DES FIGURES et DES CAPTURES vii
INTRODUCTION GENERALE 1
Chapitre 1 : Présentation générale 3
Introduction 3
1. Présentation de la structure d'accueil 3
1. 1. Missions 4
1. 2. Domaine d'activités 5
1. 3. Etude de l'infrastructure existante 5
2. Présentation du projet 7
2. 1. Contexte 7
2. 2. Problématique 7
2. 3. Objectifs attendus 7
3. Moyens mis à la disposition du stagiaire 8
Conclusion 9
Chapitre 2 : Principaux concepts et Normes de Single Sign On 10
Introduction 10
1. Principes AAA 10
1. 1. Authentification en général 10
1. 2. Autorisation 12
1. 3. Accès et traçabilité 12
2. Single Sign On (SSO) 13
2. 1. Avantages 13
2. 2. Faiblesses 14
3. Les différentes approches du Single Sign-On 15
3. 1. Approche centralisée 15
3. 2. Approche fédérative 15
3. 3. Approche coopérative 15
4. Normes et protocoles 16

4. 1. Normes 16
4. 2. Protocoles 16
4. 2. 1. OAuth2 16
4. 2. 2. Openid Connect 17
4. 2. 3. SAML2 21
5. Méthodes d'authentification HTTP et Vulnérabilité 25
5. 1. Méthode Basic 26
5. 2. Méthode Digest 26
5. 3. Authentification par cookie 27
5. 4. Attaques et Préventions 27
5. 4. 1. Attaque CSRF 28
5. 4. 3. Fixation de session 29
5. 5. Rappels sur HTTP 30
6. Généralité sur les annuaires électroniques 34
6. 1. Qu'est-ce qu'un annuaire ? 35
6. 2. A quoi sert un annuaire ? 35
6. 3. Différence entre annuaire et base de données 35
6. 4. Protocole LDAP 36
6. 4. 1. Les classes d'objets 37
6. 4. 2. Les schémas 37
6. 5. Active Directory 38
6. 5. 1. Domaine Active Directory 39
6. 5. 2. Les unités d'organisation 40
6. 5. 3. La forêt Active Directory 41
6. 5. 4. Le schéma Active Directory 41
Conclusion 41
Chapitre 3 : Etat de l'art et études comparatives entre quelques solutions SSO 43
Introduction 43
1. Evidian 43
1. 1. Avantages 45
1. 2. Coût 45
2. Systancia Access 45
2. 1. Avantages 46
2. 2. Coût 46
3. Keycloak 47
3. 1. Avantages 48

3. 2. Coût 48
4. LemonLDAP::NG 48
4. 1. Avantages 51
4. 2. Coût 51
5. Microsoft Azure Active Directory 51
5. 1. Avantages 51
5. 2. Coût 52
6. AWS SSO 52
6. 1. Avantages 53
6. 2. Coût 54
7. CAS 54
7. 1. Avantages 55
7. 2. Coût 56
Conclusion 56
Chapitre 4 : Justification et implémentation de la solution 57
Introduction 57
1. Choix de Keycloak 57
2. Présentation de Keycloak 58
2. 1. Diagramme de séquence de l'authentification avec Keycloak 60
3. Vue d'ensemble des outils technologiques utilisés 60
4. Déploiement de Keycloak 63
4. 1. Mode autonome 63
4. 2. Mode cluster de domaine 68
4. 2. 1. Installation et configuration 70
Conclusion 89
CONCLUSION GENERALE 90
Webographie : 92
Bibliographie : 93

INTRODUCTION GENERALE

Dans un monde plus en plus connecté, l'émergence des TIC (Technologies de l'Information et de la Communication) et plus particulièrement du réseau de communication, Internet est sans conteste la prouesse du millénaire réalisée par les hommes. Cette émergence a abouti à une croissance exponentielle d'échanges d'applications, des services web via les réseaux informatiques. L'accès à ces applications et services nécessitent l'utilisation des identifiants pour accéder à du contenu ou à certaines fonctionnalités. Or, le fait d'être constamment en connexion, déconnexion et être obliger de mémoriser des dizaines de mots de passe est devenu une réelle contrainte.

En plus de cette dernière, le fait de devoir se réauthentifier régulièrement peut nuire à la productivité tant pour les entreprises que pour les administrations. De plus, cela augmente le travail du help desk sur la gestion des mots de passe. Ce qui fait de nos jours, vu le nombre de plateformes dont les utilisateurs ont accès, il est important de maîtriser et faciliter leurs administrations.

Outre la productivité, la sécurité est devenue un enjeu majeur et nécessaire dans les Systèmes d'Information modernes. Ainsi, les données et les applications ne demeurent plus au sein des organisations mais sont hébergées sur des Cloud communs à plusieurs d'entre-elles et une seule application vulnérable peut suffire à compromette l'ensemble des ressources.

Le déploiement des annuaires LDAP, au-delà de leur apport fonctionnel pour la gestion des groupes, a permis de simplifier la situation en utilisant un référentiel d'authentification commun à la majorité des applications. Ainsi, pour les applications utilisant ce référentiel d'authentification, l'utilisateur peut utiliser un identifiant et un mot de passe unique. Avec l'unification de ces identités, l'application des restrictions sur un ou des groupes d'utilisateurs devient plus facile.

C'est pourquoi la mise en place d'un système de Single Sign On (SSO) doit permettre à tout utilisateur de saisir une seule fois son mot de passe pour accéder à toutes les applications web dont il a accès. Ceci améliore ainsi à la fois l'ergonomie d'accès aux applications et la sécurité du système d'information (SI) en limitant la circulation des mots de passe sur le réseau. Elle permet également aux utilisateurs ayant droit sur les ressources de réduire le nombre de mots de passe à retenir, et aussi une meilleure maîtrise de la diffusion des données à caractère personnel au sein du réseau.

Ce mémoire comporte cinq chapitres :

Le premier chapitre présente la structure d'accueil ainsi que le projet de stage.

Le deuxième chapitre parle des principaux concepts et normes de Single Sign On.

Le troisième chapitre mettra l'accent sur l'état de l'art et l'étude comparative de quelques solutions SSO.

Enfin le quatrième chapitre revient sur la justification du choix de Keycloak et de son implémentation.

Chapitre 1 : Présentation générale

Introduction

Ce chapitre présente d'abord la structure d'accueil ensuite la présentation du cadre général du projet à travers la problématique, les objectifs visés et enfin les moyens mis à la disposition du stagiaire.

1. Présentation de la structure d'accueil

L'éducation joue un rôle essentiel dans le développement économique de notre pays. C'est pour cette raison qu'elle constitue un des fondements majeurs de tous les régimes qui se sont succédés au Sénégal de l'indépendance à nos jours.

Il convient de poursuivre la modernisation du système d'information en vue d'une meilleure utilisation des TIC (Technologies de l'Information et de la Communication) dans la conception, la mise en œuvre et le suivi-évaluation des politiques éducatives du gouvernement du Sénégal.

C'est dans cette optique que SIMEN fut créé en juillet 2016 pour mieux répondre aux attentes sur :

- Les orientations de l'école du futur dégagées à travers :
  - Le PAQUET (Programme d'Amélioration de la Qualité, de l'Equité et de la Transparence).
  - Les 11 décisions présidentielles issues des Assises de l'Education et de la Formation
- Les Dysfonctionnements et incohérences notés sur le système d'information précédent

Dès lors le SIMEN permet une meilleure utilisation des technologies de l'information et de la communication dans la conception, la mise en œuvre et le suivi-évaluation des politiques éducatives.

SIMEN (Système d'Information et de Management de l'Education Nationale) est le « bras technique » du MEN (Ministère de l'Education Nationale). Il vise une refonte complète du système d'information du MEN en offrant une suite d'outils interconnectés centralisant des données d'organisation, de communication et de collaboration, accessibles à tout moment via différentes interfaces et différents appareils (ordinateurs, smartphones, tablettes, etc.).

SIMEN, est un service informatique rattaché à la DAGE (Direction de l'Administration Générale et de l'Equipement du MEN) à sa tête un coordonnateur. Avec un budget annuel qui

dépend de celui du Ministère de l'éducation nationale, des partenariats et projets ; SIMEN a plus d'une vingtaine d'employés répartie en trois composantes que sont :

- L'Ingénierie Logicielle, qui veille à la modernisation, à l'harmonisation, à l'automatisation et à la digitalisation des outils logiciels misent à la disposition du MEN ;
- L'Infrastructure, la Connectivité et l'Equipement, qui veille sur l'intranet et l'extranet du ministère de l'éducation nationale. Il assure l'accès à internet et aux ressources partagées de l'ensemble du système éducatif. Veille et accompagne l'utilisation et la mise en niveau de l'ensemble des outils informatiques du ministère.
- La Formation et la Communication, accompagne la conduite au changement, de l'innovation numérique et la formation des agents du ministère aux nouvelles technologies et aux processus de travail mis en place.

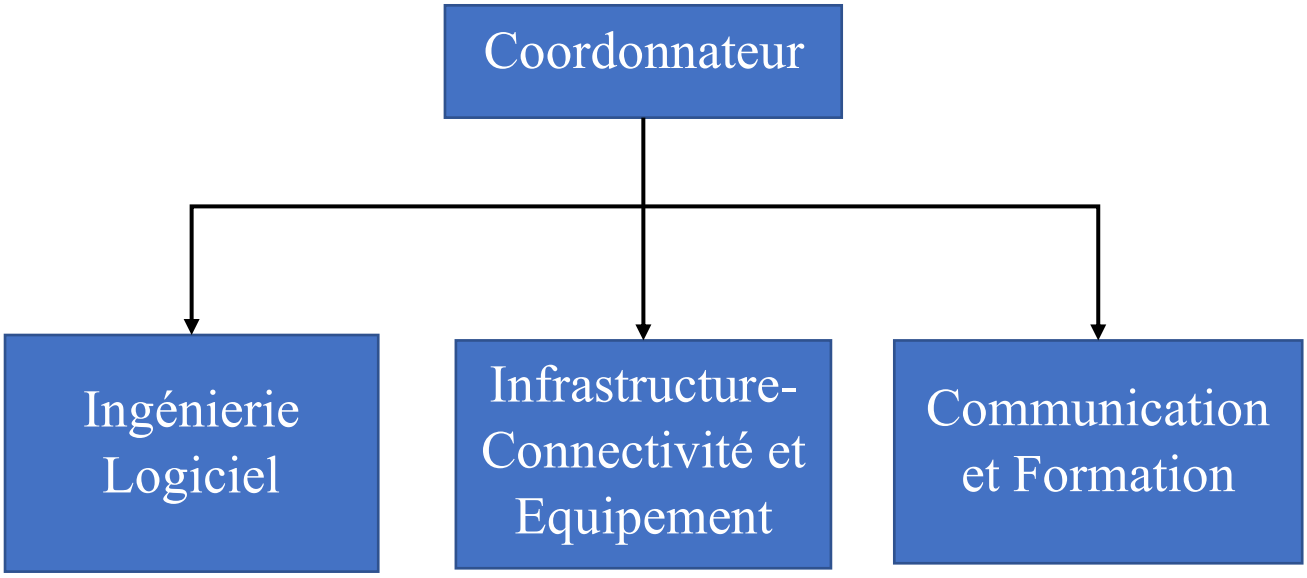


Figure 1 : Organigramme du SIMEN

1. 1. Missions

Ayant pour objectif la refonte complète du système d'information du MEN, SIMEN à travers une démarche cohérente a plusieurs missions :

- Faire l'Identifiant de l'Education Nationale (IEN) la clé qui permet d'accéder au système d'information et de management. Le meilleur usage de cette clé permet de centraliser et de faciliter l'accès au système d'information (SI) tout en ayant une meilleure transparence sur l'ensemble de ces accès. De ce fait un identifiant unique

est attribué à chaque membre du personnel de l'éducation nationale ainsi qu'aux élèves, enseignants et parents d'élèves. Cet identifiant permet de se connecter au système d'information, l'accès au réseau wifi, aux courriels professionnels, aux ressources numériques et aux différentes plateformes dédiées.

- Renforcer les capacités de gestion et de pilotage du système éducatif et contribuer à l'amélioration des enseignements et apprentissages ainsi que la communication, à travers une généralisation de l'utilisation du numérique à l'école.
- Une meilleure utilisation des TIC dans la conception
- La mise en œuvre et le suivi-évaluation des politiques éducatives du gouvernement.
- Renforcement des capacités de gestion et de pilotage du système éducatif.
- Engager et stimuler les élèves
- Outiller les personnels de l'éducation nationale
- Optimiser le fonctionnement des établissements
- Transformer les enseignements et apprentissages

1. 2. Domaine d'activités

SIMEN offre une palette de services dans le domaine de la technologie de l'information et de la communication. Les services du SIMEN sont orientés open source et réalisés selon les besoins et l'exploration du système éducatif. Il répond par conséquent aux réels problèmes d'informatique et de la communication des acteurs du MEN.

Depuis sa mise en place, le service ne cesse d'imprégner les acteurs de l'éducation nationale dans le domaine de :

- L'ingénierie logiciel
- Le cloud Computing
- La virtualisation
- La sécurité
- La communication

1. 3. Etude de l'infrastructure existante

SIMEN est plus orienté ingénierie logicielle en développant des plateformes pour l'automatisation et la digitalisation des tâches quotidiennes des acteurs du MEN. Pour l'hébergement de ses plateformes et le stockage de ses ressources.

Il dispose 2 abonnements cloud :

- Le cloud Orange, qui héberge l'ensemble de ses serveurs d'applications, ses serveurs de bases de données et proxy.
- Le cloud de Microsoft Azure qui héberge l'ensemble de ses annuaires.

L'hébergement de ses plateformes et ressources sur un cloud lui ont permis d'assurer une haute disponibilité, une transparence, une sécurité, ... de son infrastructure. Pour l'échange de données entre applications, SIMEN dispose des micro services et des API jouant un rôle de middleware entre ses différentes plateformes et services.

Pour la sécurité de ses plateformes, il délègue l'authentification à Microsoft Azure pour profiter de l'expérience SSO d'Azure avec les comptes Microsoft office 365, ainsi l'ensemble du personnel éducatif se logue une seule fois sur une application donnée pour avoir accès à l'ensemble des applications du domaine "education.sn" dont il a droit durant toute sa session.

- Architecture

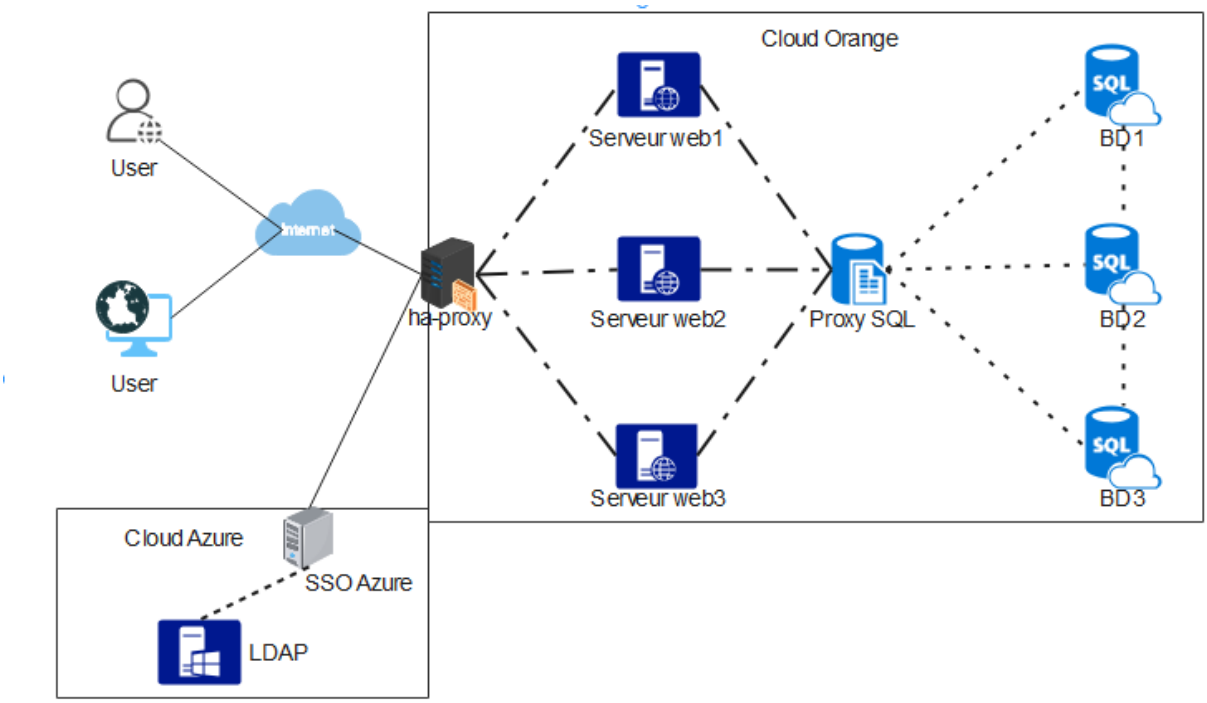


Figure 2 : Architecture de l'infrastructure existante du MEN

2. Présentation du projet

2. 1. Contexte

Les entreprises comme les administrations connaissent un réel besoin de coopérer en extériorisant leurs ressources ; particulièrement web qui ont l'apanage d'être plus rapide et moins coûteuses à mettre en œuvre, délivrant souvent les services informatiques de la complexité d'administration de l'infrastructure. Les applications Web permettent aussi de simplifier, d'accélérer et d'amplifier les échanges entre une structure et ses partenaires. Cependant, à chaque connexion sur une application, l'utilisateur doit fournir un identifiant et un mot de passe spécifique à la plateforme Web visitée. Chaque utilisateur doit donc retenir plus d'une dizaine d'identifiants et de mots de passe, plus ou moins complexes selon les exigences de la sécurité de l'application.

Soucieux des pertes ou oublis des paramètres de connexion, les utilisateurs préfèrent les noter ou utiliser des combinaisons de mot de passe faible emmenant ainsi des failles de sécurité du système d'information. Il s'avère alors nécessaire de mettre en place une solution palliative pour contourner le problème d'authentification multiple et réduire le travail du help desk sur la réinitialisation des mots de passe.

2. 2. Problématique

Les systèmes d'information d'aujourd'hui sont complexes car composés de diverses ressources telles que des applications internes, des applications web, des micro services et/ou différents systèmes d'exploitation. L'utilisateur se voit obliger de saisir à chaque fois ses identifiants d'authentification. Une telle situation ne cesse de poser des problèmes. Non seulement, la saisie manuelle coûte un temps précieux, mais il est extrêmement difficile de se souvenir de plusieurs mots de passe souvent complexes. Ainsi, les utilisateurs appellent très souvent l'équipe help desk pour des oublis de mots de passe.

2. 3. Objectifs attendus

La mise en place d'un système d'authentification centralisée et unique permet de combler plusieurs besoins à la fois :

- Augmenter la sécurité et obéir aux contraintes réglementaires.

  En mettant un point de passage obligé entre l'utilisateur et les applications, une structure peut contrôler facilement les accès de manière efficace. De plus, ces accès et les opérations d'administrations sont historisés, ce qui facilite les audits. Des rapports

périodiques peuvent être mis à la disposition des utilisateurs et responsables de la sécurité en fonction des droits d'accès tels que, les instantanés sur les objets gérés, les situations à risque, la surveillance des points d'attention, l'activité et la solution. Tout cela facilite la conformité à des exigences de confidentialité, de l'intégrité et de disponibilité.

- Réduire les coûts de fonctionnement du service informatique.

  En multipliant les mots de passe, souvent pour d'excellentes raisons, la productivité de l'utilisateur baisse et la qualité du travail se fait sentir. Mais ces « coûts cachés » ont aussi une face visible : jusqu'à 30% des appels au help desk résultent de mots de passe oubliés. Cette charge sera considérablement allégée par un SSO, avec un retour sur investissement facile à estimer. D'autres études montrent que la mise en place d'un SSO permet de gagner jusqu'à 24 heures de travail par an et par utilisateur à temps plein.
- Ouvrir l'information au monde extérieur sans risque.

  Cette demande est de plus en plus fréquente : l'accès au Web est devenu aisé, mais les employés continuent à avoir des difficultés à accéder depuis l'extérieur aux applications intranet. Médecins qui doivent consulter des dossiers médicaux, ingénieurs sur un chantier, commerciaux dans leurs hôtels, banques qui pour plusieurs activités ont l'obligation de continuité d'activité : un SSO permet un accès transparent et sécurisé aux applications Web, même de l'extérieur.

Disposant déjà d'un SSO payant qui pallie à ces problèmes, SIMEN envisage de mettre en place son propre SSO avec une solution open source. L'objectif du projet est la mise en place d'une solution d'authentification unique (SSO) propre au MEN.

3. Moyens mis à la disposition du stagiaire

Pour bien mener à terme notre projet, SIMEN a mis à notre disposition l'ensemble des ressources matérielles, logicielles et humaines nécessaires afin d'y parvenir :

- L'accès au portail de Microsoft Azure : pour accéder aux annuaires de l'entreprise
- Une licence office 365 pour accéder à l'ensemble de leurs plateformes
- L'autorisation d'accès à certains serveurs de productions
- L'accès à la documentation officielle du service

Conclusion

A travers les projets qu'il a eu à mener depuis sa création, SIMEN est devenu un pilier du MEN. Ce service a permis, de faciliter les travaux collaboratifs entre les différents acteurs de son système d'information en faisant fi de la distance, de faciliter la mobilité des enseignants et élèves, aux parents d'élèves de suivre la performance de leurs enfants, etc. Il lui arrive parfois de faire appel à une expertise d'un particulier pour la réalisation de certains modules de ses projets les plus innovants.

Chapitre 2 : Principaux concepts et Normes de Single Sign On

Introduction

L'authentification est un moyen qui consiste à vérifier l'identité d'un sujet avant de lui donner accès à une ressource. Généralement, l'authentification est précédée d'une identification qui permet à ce sujet de se faire reconnaitre du système par un élément dont ce dernier lui avait doté. La complexité des systèmes d'information d'aujourd'hui (plusieurs applications web à protéger, un grand nombre d'acteurs, ...) fait que les organisations font recours à l'authentification unique.

Dans ce chapitre, il sera question d'aborder en premier lieu des principes d'authentification, en deuxième lieu du Single Sign-On, en troisième lieu des normes et protocoles de SSO, en quatrième lieu des protocoles SSO et enfin de la généralité sur les annuaires.

1. Principes AAA

L'acronyme AAA (Authentification, Authorization, Accouting) correspond à la base à un protocole qui réalise trois fonctions : Authentification, Autorisation, Accès et traçabilité. Il est un modèle de sécurité implémenté dans certains équipements qui réalisent un système de gestion des accès.

1. 1. Authentification en général

L'authentification est une phase qui permet à une entité d'apporter la preuve de son identité. Elle vient en complément de l'identification. Pour s'authentifier, le sujet ajoute une preuve à l'identification. Ce sont ces preuves qu'on appelle facteurs d'authentification. Elles peuvent être réparties en quatre classes de facteurs d'authentification :

- Facteur mémoriel (ce que l'entité sait) : il s'agit du facteur d'authentification le plus utilisé, car il n'est pas coûteux. Il est utilisé principalement pour contrôler les accès logiques et se base sur quelque chose que l'entité est le seul à connaitre et qu'il pourra reproduire sur un terminal pour valider son authentification. Par exemple un mot de passe, une phrase de passe, un code PIN, etc.
- Facteur matériel (ce que l'entité a) : dans ce cas l'authentification est basée sur un objet possédé par le sujet désirant s'authentifier. Il peut s'agir d'un objet, d'un périphérique physique, etc. Ce moyen est souvent utilisé pour les contrôles d'accès physique, car il

présente un très bon rapport sécurité/coût. Par exemple une carte d'identité, une carte magnétique, un badge, etc.

- Facteur corporel (ce que l'entité est) : ce type de facteur est basé sur les traits physiques de l'entité. Ce dernier va donc s'authentifier grâce à son corps. C'est un moyen sûr de vérifier l'identité d'une personne à condition d'avoir du matériel adapté ce qui est généralement cher. Dans ce type d'authentification, on retrouve tout ce qui touche à la biométrie : scan d'empreinte digitale, géométrie de la main, scan de l'iris ou encore scan facial, etc.
- Le lieu où se trouve le sujet et l'heure en cours sont parfois considérés comme le quatrième et le cinquième facteur de l'authentification.

L'authentification est la procédure qui consiste, pour un Système d'Information (SI), à s'assurer de la légitimité d'une demande d'accès faite par un sujet qui peut être un individu, un autre système, un périphérique physique ou un programme informatique. Ceci permet d'autoriser ou non ce sujet à accéder aux ressources du SI conformément aux règles d'accès définies. L'authentification permet donc de valider la légitimité de l'accès afin que le système puisse attribuer à ce sujet les données ou attributs d'identité pour cette session. Ces attributs sont détenus par le système ou peut être fournis par le sujet lors de l'authentification. C'est à partir des éléments issus de ces deux processus que le contrôle d'accès aux ressources du système peut être paramétré.

La vérification de l'identité des parties communicantes est en réalité la base de tous les systèmes existants. Il fut complété au fil des années, mais reste historiquement la première modélisation de la sécurité d'un système. Dans le cas d'un individu, l'authentification consiste, en général, à s'assurer que celui-ci possède une preuve de son identité. Le système utilise un facteur d'authentification ou une combinaison de facteurs d'authentification dans la phase de vérification des identités.

Cette phase de vérification fait intervenir un protocole d'authentification qui peut être de trois types. Nous distinguons tout d'abord l'authentification simple ne reposant que sur un seul élément ou facteur par exemple le mot de passe. Ensuite, il existe l'authentification forte qui requiert au moins deux facteurs différents. Enfin, les architectures informatiques actuelles reposent sur le concept d'authentification unique fortifiée.

1. 2. Autorisation

L'autorisation est la deuxième phase de la triade AAA. Elle agit une fois que le sujet s'est authentifié. C'est dans cette phase qu'on donne ou non accès à la ressource demandée, en fonction de la politique de contrôle d'accès. Cette dernière est le plus souvent basée sur trois principes :

- Classification des informations : chaque information nécessite un certain niveau d'accès pour le consulter.
- Niveau d'accès des utilisateurs : détermine le niveau d'accès de chaque utilisateur.
- Permission de l'utilisateur : détermine quels droits (lecture, mise à jour, lecture et mise à jour) l'utilisateur aura sur les ressources du système d'information.

A travers ces principes, une fois le sujet authentifié, il lui sera attribué certains droits sur ces ressources. Ces droits peuvent être plus restreints ou plus simples comme lors d'un accès sur un serveur mail ou autoriser le sujet à accéder à un serveur physique de l'entreprise.

1. 3. Accès et traçabilité

La dernière des trois phases de la triade AAA est désignée par le terme « Accounting » qui peut être traduit dans ce contexte par traçabilité. Une fois les sujets authentifiés et ayant obtenu une autorisation d'accès, le système garde une trace de toutes les actions effectuées par ces sujets. Ces actions peuvent être enregistrées dans les logs du système ou dans une base de données. Ainsi, l'administrateur du système pourra consulter ces logs ou la base de données afin de vérifier les actions des sujets.

La traçabilité a pour but de comptabiliser les accès aux ressources, suivre les opérations effectuées en vue d'auditer les performances, les erreurs, détecter d'éventuelles attaques en surveillant les tentatives d'accès non autorisées ou d'imputer les actions des sujets comme les débuts ou fins de connexion, de transaction.

La traçabilité est très importante pour assurer une bonne sécurité et une intervention rapide en cas d'incident. Elle agit tout d'abord de manière préventive. Les utilisateurs se sachant surveillés, font attention à leurs agissements. Ensuite, grâce à la traçabilité on peut détecter des actions suspectes voire interdites et mieux cibler la source d'un problème pour intervenir, ou corriger celui-ci. Enfin, elle permet d'avoir une trace légale des actions effectuées sur le système d'information.

2. Single Sign On (SSO)

Le Single Sign On (SSO) est un système d'authentification unique permettant à une entité de se connecter à un ensemble d'applications avec un seul identifiant. Ces applications peuvent être sur un ou plusieurs domaines différents et sur des environnements souvent hétérogènes. Le principe de ce système est de centraliser l'authentification sur un serveur qui joue le rôle de fournisseur d'identité aux applications. Une fois que le sujet s'authentifie sur ce serveur, ce dernier indique à chaque application via des cookies de session que cette entité s'est bien authentifiée. Cela lui évite d'avoir à se connecter aux applications une à une durant toute sa session.

On parle de WebSSO lorsque l'authentification unique ne concerne que des applications Web c'est-à-dire des applications client-serveur dont le client est un navigateur Web comme Google chrome, Mozilla, Microsoft Edge, etc.

Le WebSSO s'inscrit dans des architectures 100% Web, de type portail extranet/intranet par exemple. Il couvre par conséquent que des applications Web, mais permet en général de garantir plus de sécurité que le eSSO (SSO d'entreprise spécifique à certaines applications cibles), via la mise en place des règles de contrôle d'accès avancées. Il n'est pas intrusif sur les postes de travail. En revanche il peut l'être sur les applications, sauf s'il est déployé dans des composants de type reverse proxy en frontal des applications Web à protéger.

Un SSO quant à lui permet de protéger des applications Web, des applications bureaux et des applications nécessitant un client lourd. Ce dernier est une application spécifique installé sur les hôtes du réseau qui se charge de renseigner l'identifiant et le mot de passe à la place de l'utilisateur. Pour ce faire, l'application peut lire les données de session issues du système d'exploitation ou d'un certificat détenu par l'utilisateur comme une carte professionnelle par exemple.

2. 1. Avantages

Le système SSO utilise un ou des serveurs d'authentification centralisés que toutes les entités du système (les applications, les utilisateurs ou d'autres systèmes) utilisent pour s'authentifier, combinant ceci avec des techniques logicielles pour s'assurer que ces entités n'aient pas à entrer leurs identifiants plus d'une fois.

Les différents avantages de l'authentification unique sont tout d'abord la réduction de la fatigue de saisie des identifiants de connexion (login et mot de passe) due au manque de

souplesse lié à l'utilisation de différentes combinaisons de nom d'utilisateur et de mot de passe, une réduction du temps passé à saisir ces identifiants pour le même ou plusieurs comptes ainsi que la gestion des oublis de mot de passe dans le cadre du support informatique et limite les risques liés au comportement des utilisateurs comme l'écriture des mots de passe sur le post-in. La centralisation des systèmes d'authentification sans sollicitation multiple des applications protégées devient plus facile.

En outre, une solution SSO améliore la sécurité de l'authentification. En effet, les mots de passe ne circulent plus entre les applications et les serveurs d'authentification car ils sont remplacés par des jetons qui peuvent prendre plusieurs formes.

Enfin, un SSO permet d'étendre plus facilement les services offerts aux utilisateurs comme la gestion des mots de passe, la fédération d'identité, le contrôle d'accès au portail qui délivre un jeton et crée une session. Les applications contactent le portail pour vérifier la validité d'un token (jeton) d'accès afin de recueillir des informations identitaires ou autres.

2. 2. Faiblesses

Un système d'authentification unique donnant potentiellement accès à de nombreuses ressources une fois l'entité authentifiée, les pertes peuvent être lourdes si une personne mal intentionnée a accès aux informations d'identification de cette entité.

Avec un système SSO, il est alors important de renforcer la sécurité.

Une attention particulière doit donc être prêtée aux informations personnelles des utilisateurs. Pour cela une méthode d'authentification forte devra être implémentée telle que l'authentification unique à double facteurs. Une authentification forte est une authentification combinant plusieurs facteurs d'authentification par exemple combiner le couple login/mot de passe avec une confirmation de l'identité par un code envoyé à l'entité désirant s'authentifier avec une carte à puce, une biométrie, etc.

Dans certains cas particuliers, comme le Webmail par exemple, c'est l'application qui doit se connecter au serveur. L'application n'ayant pas le mot de passe de l'utilisateur, il lui est impossible de rejouer l'authentification afin d'accéder au service. C'est très bien pour la sécurité mais cela peut être un problème. Dans ce cas, le SSO peut transmettre l'identification et le mot de passe de l'utilisateur à l'application mais cela doit rester exceptionnel et différentes méthodes plus ou moins sécurisées existent pour cette contrainte.

3. Les différentes approches du Single Sign-On

Il existe trois types d'approches pour la mise en œuvre d'un système d'authentification unique : l'approche centralisée, l'approche fédérative et l'approche coopérative.

3. 1. Approche centralisée

Le principe de base est ici de disposer d'une base de données globale ou centralisée de tous les utilisateurs ou d'un annuaire. Cela permet également de centraliser la gestion de la politique de sécurité.

Un exemple de mise en œuvre est le logiciel libre **LemonLDAP::NG**, un autre exemple est le logiciel libre **Keycloak**. Ces deux exemples de WebSSO conviennent pour sécuriser des applications Web. Il faudra se tourner vers d'autres logiciels lorsque le besoin est de mettre en œuvre une solution SSO à la fois pour des applications Web et pour des applications métiers à l'intérieur de l'entreprise.

Cette approche est principalement destinée à des services dépendants tous d'une même organisation.

3. 2. Approche fédérative

Dans cette approche, dont le système **Liberty Alliance** est le principal exemple, 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 ce dernier avec les services partenaires. Lorsque l'utilisateur souhaite accéder à un service offert par un des partenaires, il choisit le fournisseur d'identité sur lequel il souhaite s'authentifier puis ses données d'identité sont transmises au partenaire.

Cette approche 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.

3. 3. Approche coopérative

Cette approche, dont les systèmes Shibboleth et CAS sont les principaux représentants, part du principe que chaque utilisateur dépend d'une des entités partenaires. Ainsi, lorsqu'il cherche à accéder à un service du réseau, l'utilisateur est authentifié par le partenaire dont il dépend. Cependant, chaque service du réseau gère indépendamment sa propre politique de sécurité.

Comme dans l'approche fédérative, l'approche coopérative est développée pour répondre à un besoin de gestion décentralisée des utilisateurs et des services, où chaque utilisateur ou service dépend d'une des entités partenaires.

4. Normes et protocoles

4. 1. Normes

Une norme est un document de référence élaboré par un organe de normalisation reconnu comme l'ISO par exemple.

Une norme définit les règles, les caractéristiques, les bonnes pratiques et les recommandations applicables à des produits, des services, des méthodes, des processus ou des organisations. Elle est élaborée par consensus entre l'ensemble des parties prenantes. D'ailleurs, la participation à l'activité de normalisation permet aux acteurs de se placer en situation de veille technologique, d'anticiper les évolutions et donc de rester compétitifs.

Une norme dans le secteur informatique est une spécification technique émise par un organisme de normalisation et destinée à harmoniser l'activité d'un secteur (par exemple langage C, HTML, XML, HTTP, …). Il existe des normes de sécurité des systèmes d'information, des normes de communication, des normes de sécurité WEP, etc.

4. 2. Protocoles

La notion de protocole peut être définie comme étant un ensemble de règles et de formats (sémantiques et syntaxiques) établi entre des entités homologues (en général deux) pour améliorer les propriétés du service de communication. La notion de protocole se retrouve à tous les niveaux du modèle en couche.

Le SendMail par exemple s'appuie sur le protocole SMTP, les serveurs WWW sur le protocole HTTP. Le système d'authentification unique s'appuie sur un ensemble de protocole.

4. 2. 1. OAuth2

Le protocole OAuth2 (Open-standard Authorization version 2) successeur du protocole OAuth1, est un Framework d'autorisation permettant à une tierce application d'accéder à un service web.

Largement utilisé dans le domaine du web avec notamment Facebook ou encore Google, OAuth est devenu incontournable. Il a été formalisé en octobre 2012, OAuth2 est né d'un besoin

simple : « comment une application peut-elle accéder à une ressource protégée au nom de son propriétaire sans connaitre ses identifiants (login et mot de passe) ? ».

Avec la démocratisation des applications web et la prolifération des applications mobiles, différentes applications sont amenées à interagir entre elles. Ainsi, un site web A pourrait utiliser les données d'un réseau social connu afin d'inscrire un utilisateur. De plus OAuth2 est utilisé pour fournir l'accès API à d'autres services (sites web, API, applications, …) pour le compte d'un utilisateur.

En d'autres termes, on pourrait dire que OAuth2 consiste à donner à une application extérieure connectée, la possibilité de faire des actions en votre nom dans une autre application (dans laquelle vous avez bien sûr un compte). On pourrait prendre l'exemple d'API LinkedIn qui permettrait à d'autres applications RH (Ressources humaines) de récupérer votre profil LinkedIn en votre nom.

La façon générale dont fonctionne ce protocole consiste à permettre à une application d'avoir un jeton d'accès (qui représente l'autorisation d'un utilisateur pour que le client accède à ses données) qu'elle peut utiliser pour authentifier une demande à un point de terminaison d'API.

4. 2. 2. Openid Connect

Openid Connect est un protocole récent porté par la fondation Openid en remplacement du protocole Openid. Il a été créé en 2014. Ce standard est adopté par de nombreux acteurs, comme Google qui l'utilise pour son authentification en remplacement de Openid 2. L'Etat Français a lui aussi fait le choix de Openid Connect pour sa plateforme France-Connect.

Le protocole Openid Connect (OIDC) définit les échanges pour mettre en œuvre un point d'authentification unique et des services de partage d'informations. Il s'appuie sur le standard OAuth2.0 qui définit des mécanismes permettant à une application tierce d'obtenir le consentement de l'utilisateur pour accéder à ses ressources tout en protégeant ses authentifications.

OIDC complète le protocole OAuth2.0 :

- En ajoutant l'ID token, un Json Web Token (JWT) contenant des informations sur l'authentification de l'utilisateur ;
- Et en normalisant la récupération d'informations sur l'utilisateur.

Openid Connect fait appel aux notions de Relying Party (RP) et d'Openid Provider (OP). Le RP qui est l'application protégée par Openid Connect, se connecte à l'OP (le service d'authentification implémentant le protocole Openid Connect) pour authentifier l'utilisateur. Ce dernier s'authentifie et l'OP retourne la requête confirmant la réussite. Ensuite, le RP envoie une requête à l'OP pour demander les informations de l'utilisateur sur les données transmises afin d'obtenir son consentement.

Openid Connect utilise les dernières normes telles que Json en lieu et place de XML (eXtensible Markup Language) ou JWT encodé en base64 pour récupérer les informations utilisateurs. Le JWT est en fait constitué de trois JSON encodés en base64 concaténés avec des points : l'entête précisant la méthode de signature utilisée, les données utilisateur et la signature.

OIDC autorise les clients à vérifier l'identité d'un utilisateur final en se basant sur l'authentification fournie par un serveur d'autorisation, en suivant le processus d'obtention d'une simple information du profil de cet utilisateur. Ce processus est basé sur un dispositif interopérable de type REST. Il permet à un éventuel client, y compris web, mobile et JavaScript, de demander et recevoir des informations sur la session authentifiée et sur l'utilisateur final car il permet l'échange d'attributs d'identité. Cet ensemble de spécifications est extensible, supporte des fonctionnalités optionnelles telles que le chiffrement des données d'identité, la découverte dynamique de fournisseurs OpenID et la gestion de sessions.

Toute cette procédure peut être schématisée comme suit :

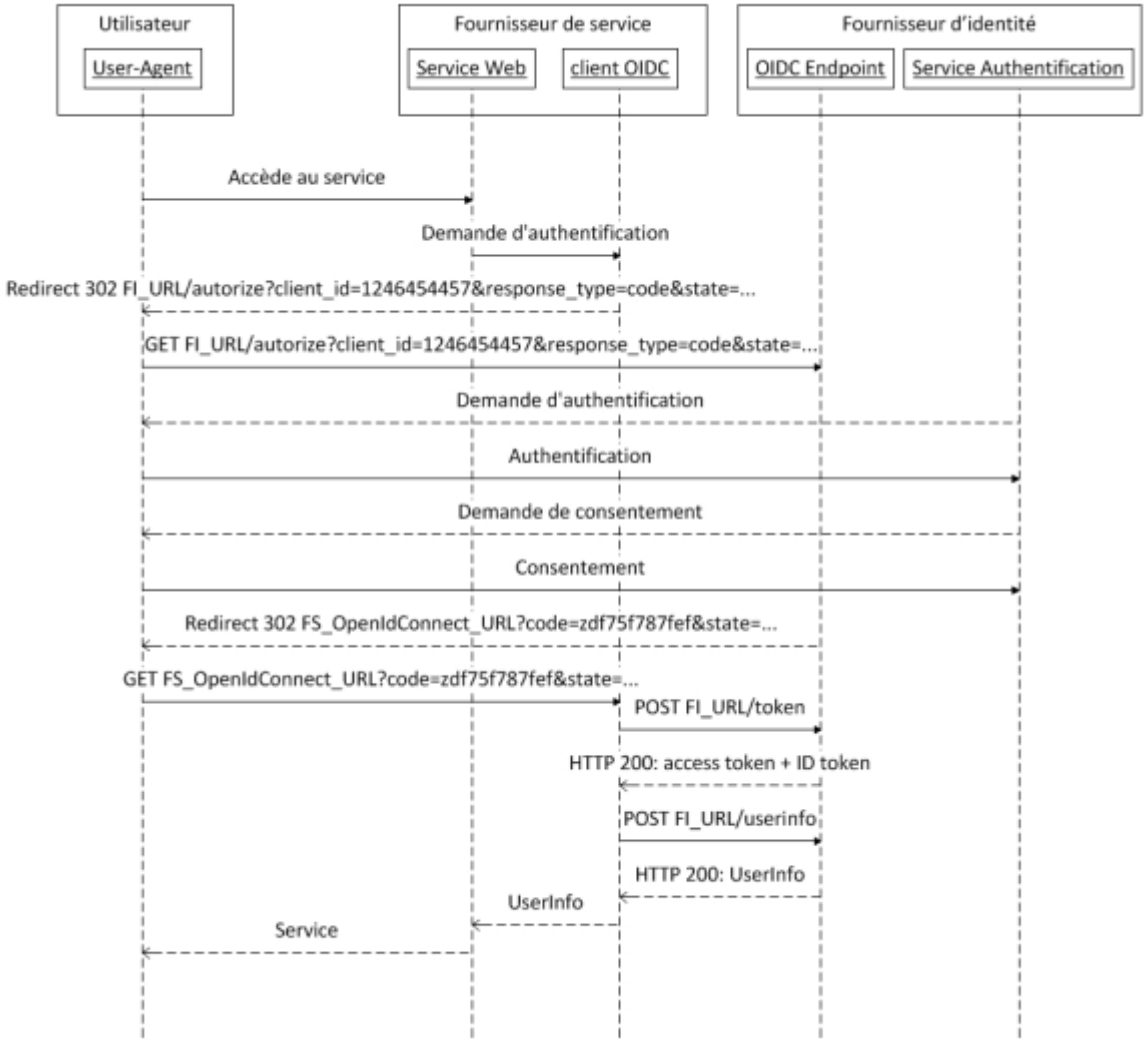


Figure 3 : Diagramme de séquence de l'authentification avec Open Id Connect

1- L'utilisateur souhaite accéder à une ressource protégée ;
2- Le fournisseur de service effectue une demande d'authentification par redirection. Les principaux paramètres sont les suivants :
   - « client_id » : l'identifiant du client OIDC enregistré dans le système du fournisseur d'identité,
   - « response_type » : détermine le diagramme de séquence et les éléments envoyés dans la réponse de l'OIDC Endpoint. Dans le cadre de cette séquence, sa valeur est « code »,
   - « scope » : ce paramètre détermine les « claims » demandés par le fournisseur de service. Par exemple, il peut contenir les valeurs « openid » et « profile » pour

demander des informations définies par la norme OIDC sur le profil de l'utilisateur. Les claims peuvent également être demandés par le paramètre « claims » ou directement dans un JWT. La liste possible des scopes est définie par la norme OIDC,

- « redirect_URI » : l'URL de redirection vers le client OIDC du fournisseur de service,
- « state » : ce paramètre contient une valeur aléatoire liée à la session de l'utilisateur, il est renvoyé à la suite de l'authentification de l'utilisateur. Il permet de contrer les attaques de type CSRF,
- « nonce » : ce paramètre contient une valeur aléatoire unique, il est renvoyé dans les paramètres de l'ID Token. Il permet de contrer les attaques de type rejeu ;

3- « OIDC Endpoint » (ou Autorization Endpoint) demande à l'utilisateur de s'authentifier
4- L'utilisateur s'authentifie auprès du fournisseur d'identité ;
5- « OIDC Endpoint » demande le consentement à l'utilisateur de fournir les informations demandées selon le « scope » ou le « claims » ;
6- L'utilisateur donne son consentement ;
7- « OIDC Endpoint » redirige l'utilisateur vers le client OIDC grâce à la redirect_URI et en incluant les paramètres suivants :
   - « code » : ce paramètre correspond au code d'autorisation généré par l'OIDC Endpoint,
   - « state » : ce paramètre correspond à celui envoyé dans la requête de demande d'authentification ;
8- Le client OIDC du fournisseur de service effectue une requête de serveur à serveur vers l'OIDC Endpoint (ou le Token Endpoint) avec les paramètres suivants :
   - « code » : ce paramètre correspond à « authorization code » précédemment retourné par l'OIDC Endpoint,
   - « grant_type » : ce paramètre indique à l'OIDC Endpoint (ou Token Endpoint) le type de cinématique, soit la valeur « authorization_code » dans le cas décrit ;
9- Le serveur répond sous format Json comprenant les éléments suivants :
   - « access_token » : contient l'accès token qui permet au client OIDC de récupérer les informations consenties par l'utilisateur suite à son authentification,
   - « token_type » : précise le type de token reçu et les modalités pour récupérer les ressources associées. La norme OIDC utilise uniquement le type « Bearer ». Il

autorise toute personne ou entité possédant un access token de ce type d'accéder aux ressources associées sans s'authentifier auprès du Ressource Endpoint,

- « expires_in » : durée de validité de l'access token,
- « id_token » : contient l'ID token

10- Le client OIDC effectue une requête de serveur à serveur vers le UserInfo Endpoint en mettant l'access token précédemment reçu dans l'entête Authorization ;

11- Le serveur d'identité vérifie la validité de l'access token reçu puis fournit dans la réponse le UserInfo contenant les claims associés au scope.

4. 2. 3. SAML2

Le protocole SAML (Security Assertion Markup Language) est un protocole d'échange d'information liée à la sécurité et plus spécifiquement à l'authentification, basé sur du XML. Il a été créé en 2002 par l'organisation OASIS (Organization for the Adevancement of Structured Information Standards, est un consortium mondial qui travaille pour la standardisation de formats de fichiers ouverts basés notamment sur du XML). A la même année, les protocoles Shibboleth et ID-FF (Liberty Alliance) ont fusionné pour devenir SAML2. Il a pour objectifs de permettre l'authentification et l'échange d'informations d'identités situés dans des organisations différents.

SAML est basé sur la notion de cercle de confiance. Il impose l'enregistrement préalable des fournisseurs de services (Service Provider) et des fournisseurs d'identités (Identity Provider) par l'échange mutuel des métadonnées entre les partenaires. Il autorise plusieurs méthodes : GET, POST, Artfact GET ou Artfact POST. Les méthodes « Artfact » permettent de passer des jetons en lieu et place de la requête. De plus, SAML permet de gérer des conditions ou des contextes d'authentification. Les applications peuvent également imposer des contraintes sur l'identification.

Le principe de fonctionnement est le suivant (cf. figures 6). L'application initiale appelée fournisseur de service, délègue l'authentification de l'utilisateur aux fournisseurs d'identités. Un fournisseur de services (FS) peut faire appel à plusieurs fournisseurs d'identités (FI). Lors de la phase suivante, l'utilisateur s'authentifie auprès du FI de son choix puis est redirigé vers le FS d'origine. Le FI transmet au FS une assertion SAML contenant l'ensemble des attributs nécessaires. D'un point de vue technique, SAML définit la structure des échanges de messages au format XML, appelés les assertions. Il décrit également les cas d'utilisation en précisant les échanges des messages ainsi que les paramètres envoyés et reçus.

De manière complémentaire, les relations de confiance techniques entre les fournisseurs s'appuient sur des certificats qui permettent ainsi de garantir des communications sécurisées entre eux.

Les assertions SAML sont basées sur les couches SOAP, XML Encryption et XML Signature. SOAP est le protocole d'encapsulation standard des messages XML, utilisé principalement par les Web services.

XML Encryption est le protocole standard de chiffrement des messages XML. Il a la particularité de pouvoir chiffrer la globalité du message ou simplement un sous-ensemble précis. Cela permet d'avoir par exemple un document XML en clair avec des valeurs d'attributs chiffrées.

XML Signature est le protocole standard de signature des messages XML. Tout comme XML Encryption, il permet de cibler l'élément à signer. Cela permet à plusieurs intervenants de signer chacun une partie différente du document XML.

SP (Service Provider) et IdP (Identity Provider) sont deux entités qui ont connaissance chacune l'une de l'autre en termes d'identifiant et de certificat. Les messages XML qui transitent sur le réseau sont donc chiffrés par la clef publique du destinataire, seul capable de déchiffrer le message avec sa clef privée. L'émetteur signe ses assertions avec sa clef privée permettant au destinataire de vérifier sa provenance.

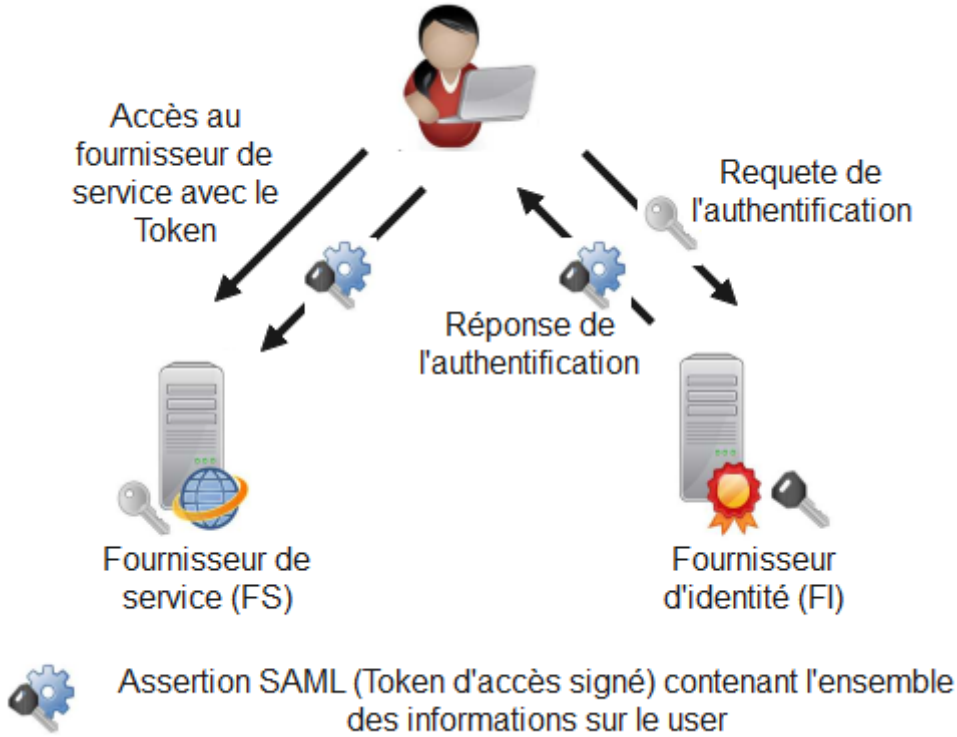


Figure 4 : Diagramme de séquence de l'accès SAML2

4. 2. 4. CAS

CAS est un protocole d'authentification centralisé fournissant un service SSO pour les applications Web. Il s'agit d'un serveur d'authentification basé sur des routines java fonctionnant sur des moteurs de type Apache, Tomcat.

CAS ne traite pas les besoins liés aux autorisations (droits applicatifs), aux fédérations d'identités (pas de connecteur SAML2 ou Openid Connect) et au transport d'attributs. CAS ne permet pas de protéger des applications dites « sur étagère ».

En outre, la base d'authentification est locale, au niveau de l'établissement. Les aspects inter-établissements ne sont donc pas pris en compte. Les mots de passe utilisateurs ne circulent qu'entre le navigateur et le serveur d'authentification à travers un canal sécurisé HTTPS.

L'utilisation de cookies exclusivement privés dans CAS (passage de tickets entre serveur d'authentification et applications uniquement sous forme de paramètres de GET HTTP) lui permet d'être opérationnel sur des serveurs situés dans des domaines DNS différents. Un module Apache « mod_cas » permet d'utiliser CAS pour protéger l'accès à des documents web statiques, les librairies clientes ne pouvant être utilisées dans ce cas. Un module PAM « pam_cas » permet de « cassifier » des services non web, tels que FTP, IMAP.

Son architecture est basée sur trois composants principaux : le serveur CAS, le navigateur web et le client CAS.

L'authentification est centralisée sur un serveur CAS. Celui-ci est le seul acteur du mécanisme CAS qui relaye les mots de passe des utilisateurs à l'annuaire centralisé LDAP. Il a pour rôle d'authentifier les utilisateurs et de certifier l'identité de la personne authentifiée aux clients CAS par son ticket de service.

Le client CAS est une application Web embarquant une bibliothèque cliente ou un client CAS intégré. Il ne délivre les ressources qu'après être assuré que le navigateur se soit authentifié auprès du serveur CAS. Le client s'assure également à intervalle régulier de la validité de la session.

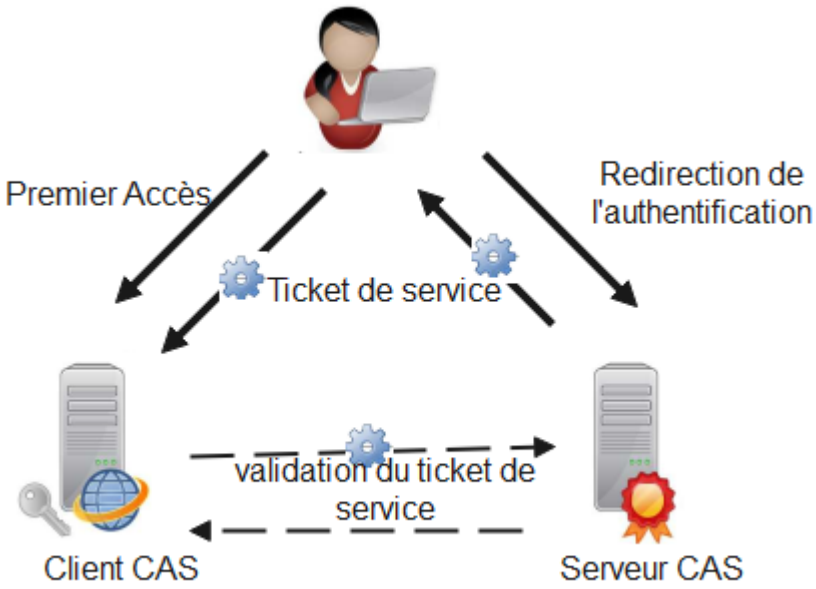


Figure 5 : Diagramme de séquence de l'authentification CAS

- Authentification d'un utilisateur : un utilisateur non authentifié, ou dont l'authentification a expiré, accède au serveur CAS. Ce dernier lui propose un formulaire d'authentification, dans lequel il est invité à saisir son identifiant et son mot passe.
- Si les informations d'authentification sont correctes, le serveur renvoie au navigateur un cookie appelé TGC (Ticket Granting Cookie). Le TGC est un cookie privé et sécurisé transmis uniquement au serveur CAS avec une durée de vie limitée à celle de la session du navigateur.

- Accès à une ressource protégée après l'authentification : le navigateur tente d'accéder à une ressource protégée sur un client CAS. Le client redirige le navigateur vers le serveur CAS afin d'authentifier l'utilisateur. Le navigateur, précédemment authentifié auprès du serveur CAS, lui présente le TGC (Ticket Granting Cookie). Le serveur CAS délivre au navigateur un Service Ticket (ST). Il s'agit d'un ticket ne comportant aucune information personnelle. Il n'est utilisable que par le navigateur qui en a fait la demande. Dans le même temps, le serveur CAS redirige le navigateur vers le service demandeur en lui passant le Service Ticket. Le Service Ticket est alors validé auprès du serveur CAS par le client CAS qui délivre la ressource au navigateur. Toutes les redirections sont transparentes pour l'utilisateur.
- Accès à une ressource protégée sans authentification préalable : Si l'utilisateur ne s'est pas authentifié auprès du serveur CAS avant d'accéder à une ressource web, son navigateur est redirigé vers le serveur CAS qui lui propose alors un formulaire d'authentification. Ensuite, suivent les mêmes étapes présentées précédemment.

5. Méthodes d'authentification HTTP et Vulnérabilité

L'acronyme HTTP signifie HyperText Transfer Protocol (peut être traduit par : Protocole de Transfert Hypertexte). Ce protocole de transmission définie la communication entre un client (le plus souvent un navigateur Web) et un serveur web. Il est développé pour le WWW (World Wide Web). HTTPS (avec S pour Secured, soit « sécurité ») est la variante sécurisée par l'usage des protocoles TLS (Transport Layer Security) et SSL (Secure Socket Layer).

La communication se déroule de la manière suivante :

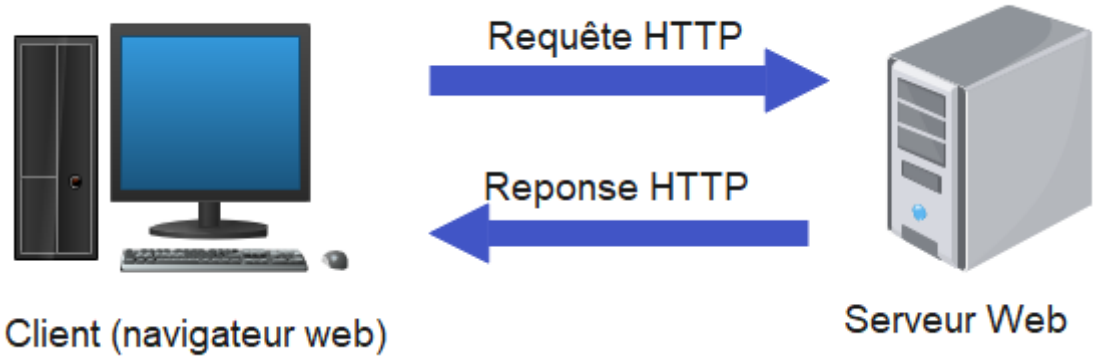


Figure 6 : Schéma d'une requête HTTP

L'ordinateur de l'internaute utilise le navigateur pour envoyer une requête à un serveur Web. Cette requête demande une ressource contenue sur le serveur. Ce dernier cherche les informations, puis il peut être amené à interpréter les résultats (exemple : PHP, Java, …) pour finalement envoyer la réponse au client. Cette réponse contient les entêtes du protocole HTTP et normalement le contenu demandé.

Pour protéger les ressources du serveur web, le protocole HTTP dispose de certaines méthodes d'authentification permettant de s'assurer de l'identité d'un utilisateur. Ces méthodes ne sont pas sans vulnérabilités.

5. 1. Méthode Basic

Il s'agit d'un mécanisme permettant de demander un identifiant et un mot de passe au client avant qu'il n'accède à une page web. Cette méthode est implémentée sur le serveur. Il permet à un ensemble de pages (appelé domaine) d'être protégé par une authentification.

Cette méthode est définie par la spécification HTTP. Elle est la plus commune et simple à mettre en œuvre mais aussi la moins sécurisée car elle transmet le mot de passe codé en base 64 qui se décode aisément. Elle n'est recommandée qu'avec une connexion chiffrée.

Les différentes étapes d'une authentification basique sont :

- Le client demande une ressource au serveur web.
- Le serveur répond avec une erreur (code HTTP 401), demandant une authentification.
- Le client refait la demande contenant les détails d'authentification dans la requête.
- Le serveur vérifie les informations et envoie la ressource demandée, ou une autre erreur (403 par exemple).

5. 2. Méthode Digest

Cette méthode est plus sécurisée que la méthode Basic car elle ne transmet pas le mot passe en clair, mais impose de stocker celui-ci (ou son hachage, qui suffit pour s'identifier et peut donc être considéré comme mot de passe) en clair. Même si elle est plus sûre que la méthode Basic, elle reste tout de même sensible aux attaques (interception de communication…) et plus sensible encore à certaines attaques (vol de fichier de mots de passe).

En d'autres termes, l'authentification d'accès Digest est une méthode qu'un client et un serveur peuvent utiliser pour échanger des informations d'identification sur HTTP. Elle utilise une combinaison de mot de passe et d'autres éléments d'information pour créer un hachage

MD5 qui est ensuite envoyé au serveur d'authentification. L'envoie du hachage évite les problèmes avec la transmission d'un mot de passe en texte clair, une faille de l'authentification d'accès Basic.

5. 3. Authentification par cookie

Cette méthode utilise les cookies HTTP qui sont définies comme étant un petit texte contenu d'habitude dans un fichier Json envoyé par un serveur d'authentification à un client HTTP pour authentifier les utilisateurs et surtout pour maintenir la session ouverte car l'authentification dépend d'un autre système.

Les étapes d'authentifications sont les suivantes :

- Le client envoie une requête d'authentification au serveur
- En cas de succès, le serveur répond en lui incluant l'entête « Set-Cookie » contenant le domaine de validité du cookie, l'identifiant de session, la durée de validité, éventuellement un chemin d'accès à la ressource et des options.
- Pour chaque requête suivante, le client doit envoyer son cookie de session
- Lors de la déconnexion, le serveur positionne l'entête « Set-Cookie » à vide ce qui a pour effet de supprimer le cookie du client

La taille maximale d'un cookie est de 4 Ko (kilo octet). Un cookie est valide jusqu'à la fermeture de la session ou jusqu'à une durée spécifiée. De plus, il n'est utilisable que sur un domaine. Les cookies peuvent être « HTTPOnly » ou « Secure ».

Un cookie « HTTPOnly » ne peut être lu que par un navigateur et non par des applications JavaScript ou Java par exemple. Un cookie « Secure » est valide uniquement avec le protocole HTTPS.

Il est important de noter que la méthode d'authentification par cookie est vulnérable à l'attaque CSRF (Cross-Site Request Forgery). Elle nécessite donc de prendre des mesures de sécurité spécifiques comme l'utilisation de jetons anti-CSRF.

5. 4. Attaques et Préventions

Tout ordinateur connecté à un réseau informatique est potentiellement vulnérable à une attaque.

Une « attaque informatique » est l'exploitation d'une faille d'un système informatique (système d'exploitation, logiciel ou bien même sur les actions de l'utilisateur) à des fins non connues par l'exploitant du système et généralement préjudiciables.

Sur internet, des attaques ont lieu en permanence en raison de plusieurs attaques par minute sur chaque machine connectée. Ces attaques sont pour la plupart lancées automatiquement à partir de machines infectées (par des virus, chevaux de Troie, vers, etc.), à l'insu de leur propriétaire.

Pour contrer les attaques sur le protocole HTTP, il est indispensable de connaître ces principaux types d'attaques afin de mettre en œuvre des dispositions préventives.

5. 4. 1. Attaque CSRF

L'objet de cette attaque est de transmettre à un utilisateur authentifié une requête HTTP falsifiée qui pointe sur une action interne du site, afin qu'il l'exécute sans en avoir conscience et en utilisant ses propres droits. L'utilisateur devient donc, complice d'une attaque sans même s'en rendre compte. L'attaque étant actionnée par l'utilisateur, un grand nombre de systèmes d'authentification est contourné.

La principale contre-mesure est d'utiliser des jetons de validité dans les formulaires (Synchronizer Token Pattern). Il s'agit de faire en sorte qu'un formulaire posté ne soit accepté que s'il a été produit quelques minutes ou secondes auparavant en générant côté client un jeton avec une période de validité. Le jeton anti-CSRF doit être transmis en paramètre par chaque formulaire et vérifié côté serveur, ce qui en complexifie la mise en œuvre. Le jeton peut être valide pour une session au lieu d'une requête pour permettre au client l'utilisation de plusieurs onglets dans son navigateur.

D'autres contre-mesures permettent de se prévenir de cette attaque :

- Demander des confirmations à l'utilisateur pour les actions critiques, au risque d'alourdir l'enchaînement des formulaires.
- Demander une confirmation de l'ancien mot de passe à l'utilisateur pour changer celui-ci ou changer l'adresse électronique du compte.
- L'utilisation des jetons de validité dans les formulaires est basé sur la création du token via le chiffrement d'un identifiant utilisateur, un nonce et un horodatage. Le serveur doit vérifier la correspondance du jeton envoyé en recalculant cette valeur et en la comparant avec celle reçue
- Éviter d'utiliser des requêtes HTTP GET pour effectuer des actions. Cette technique va naturellement éliminer des attaques simples basées sur les images, mais laissera passer les attaques fondées sur JavaScript, lesquelles sont capables très simplement de lancer des requêtes HTTP POST.

- Effectuer une vérification du référent sur les pages sensibles c'est-à-dire connaître la provenance du client permet de sécuriser ce genre d'attaques. Ceci consiste à bloquer la requête du client si la valeur de son référent est différente de la page d'où il doit théoriquement provenir.

5. 4. 2. Vol de cookies

Le vol de cookies est un procédé par lequel l'attaquant va obtenir un accès non autorisé à des informations confidentielles. Pour cela, il va voler les cookies qui contiennent les informations nécessaires pour authentifier un utilisateur à un serveur web distant. Il suffit juste que la valeur existe dans la requête pour être authentifiée auprès du serveur. Dès lors, n'importe quelle personne possédant la valeur du cookie est en mesure de s'authentifier avec l'identité de l'utilisateur.

Ce détournement peut être effectué par le pirate en utilisant un ordinateur entre le client et le serveur ou en récupérant directement les cookies stockés sur l'ordinateur de l'utilisateur. Par défaut, la valeur du cookie est envoyée en clair, chaque partie ayant accès au réseau peut donc renifler la valeur.

Pour se munir de cette attaque, la communication entre le client et le serveur doit être chiffrée en utilisant par exemple le protocole HTTPS. Toutefois, même avec l'existence d'une connexion HTTPS, le cookie est lisible par défaut avec JavaScript via la propriété « document.cookie ». De ce fait, l'utilisation d'un Cross-site Scripting (**XSS**) permet quand même au pirate d'obtenir la valeur du cookie.

Pour contrer cette menace, les éditeurs de navigateurs ont introduit « **HTTPOnly-flag** » qui permet de cacher la valeur du cookie depuis JavaScript.

5. 4. 3. Fixation de session

L'attaque par fixation de session est une attaque proche du vol de session. Cette attaque permet à une personne mal intentionnée de déterminer l'identifiant de session d'une autre personne. En connaissant le SID d'un utilisateur, il est possible d'utiliser et de récupérer sa session pour soi.

Cette attaque repose sur le fait que, lorsqu'un utilisateur s'authentifie, un nouveau SID ne lui est pas attribué, ce qui rend possible la réutilisation de son SID. Avec cette attaque, l'attaquant fixe le SID de l'utilisateur, avant même que celui-ci n'ouvre une session sur le serveur cible. Cette méthode empêche ainsi à l'attaquant de devoir récupérer par la suite le SID de l'utilisateur. Pour ce faire, l'attaquant ouvre une session sur le serveur et obtient un SID

valide. Puis, il transmet ce dernier à sa victime, à l'aide d'un lien, par exemple au travers d'un courriel. La victime clique sur le lien et se connecte au serveur. Une fois la connexion établie, l'attaquant peut envoyer de nouvelles requêtes aux serveurs et comme il possède le même SID, il a alors accès aux données. La session a été détournée.

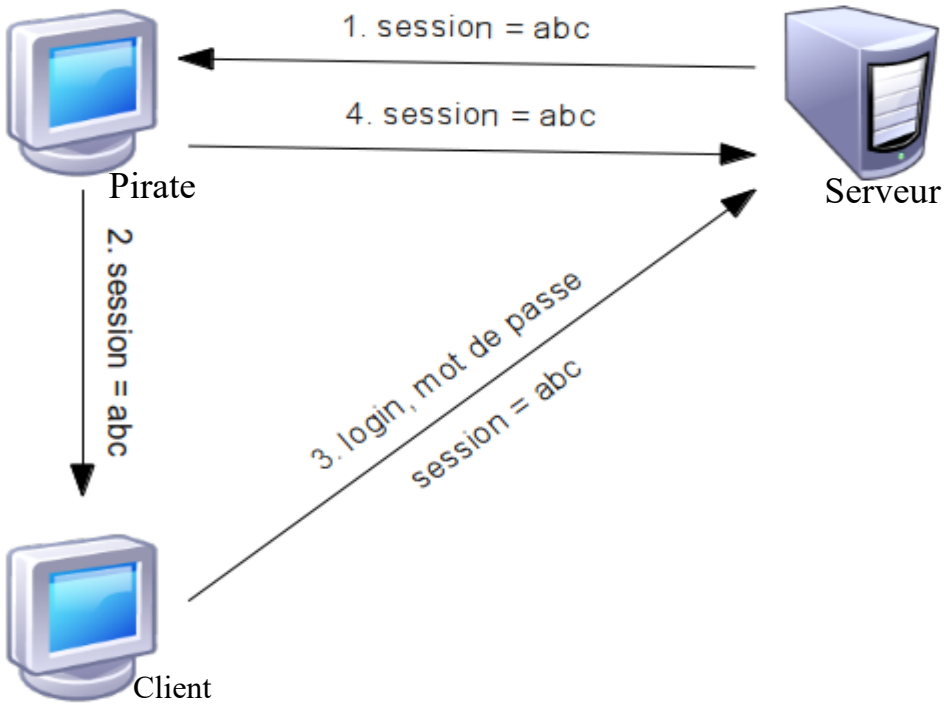


Figure 7 : Principe d'une attaque par fixation de session

1. L'attaquant crée une session sur le serveur ;
2. L'attaquant transmet ce numéro de session à la victime, par exemple via un lien ;
3. La victime s'authentifie, en utilisant ce numéro de session ;
4. L'attaquant peut maintenant utiliser le compte de la victime.

Il existe une technique simple pour se protéger contre cette attaque : lors de chaque login réussi, le serveur doit créer un nouveau numéro de session. En PHP, il suffit d'utiliser la fonction « **session_regenerate_id() ».**

Cette fonction créera un nouveau numéro de session et l'enverra au client, sans perdre les données de la session, évitant ainsi que l'attaquant puisse accéder aux données du client.

5. 5. Rappels sur HTTP

5. 5. 1. Méthodes courantes

Les méthodes HTTP les plus courantes sont :

- GET : Requête de la source située à l'URL spécifiée. Elle permet de récupérer une ressource (par exemple une page web).
- HEAD : Requête de l'en-tête de la ressource située à l'URL spécifiée.
- POST : Envoi de données au programme situé à l'URL spécifiée.
- PUT : Envoi de données à l'URL spécifiée.
- DELETE : Suppression de la ressource située à l'URL spécifiée.

La méthode GET est utilisée pour récupérer les données d'une ressource spécifique. Il s'agit de la méthode HTTP la plus utilisée. Une particularité de cette méthode est que les paramètres sont transmis dans l'URL sous la forme «?nom1=valeur1&nom2=valeur2&... ». Cette liste de valeurs est appelée « QueryString ».

Par exemple : « /test/demo_form.php?name1=value1&name2=value2 ». Tous les langages de programmation incluent des librairies CGI qui proposent des méthodes pour découper et récupérer ces paramètres qui peuvent être les valeurs transmises par le biais d'un formulaire.

A la différence de GET, la méthode POST permet d'envoyer des données au serveur pour créer ou mettre à jour une ressource. Autre différence avec GET, la méthode POST transmet tous les paramètres dans le corps de la requête.

La différence entre PUT et POST tient au fait que PUT est une méthode idempotente. Une requête PUT, envoyée une ou plusieurs fois avec succès, aura toujours le même effet (il n'y a pas d'effet de bord). À l'inverse, des requêtes POST successives et identiques peuvent avoir des effets additionnels, ce qui peut revenir par exemple à passer plusieurs fois une commande.

5. 5. 2. URL, URI et QueryString

Une URI (Uniform Resource Identifier, Identifiant de ressource uniforme en français) est une chaine qui fait référence à une ressource. Les plus courantes sont les URL (une URI particulier), qui identifient une ressource en donnant son emplacement sur le serveur Web. Au contraire, les URI font référence à une ressource grâce à son nom, dans un environnement donné, par exemple le code ISBN d'un livre.

Une URL (Uniform Ressource Locator), couramment appelée adresse web, est une référence à une ressource web. Elle permet de spécifier la localisation de cette ressource sur un serveur ou un mécanisme pour la retrouver.

Dans la majorité des cas, une URL fait référence à une page web mais peut également être utilisée pour les protocoles FTP (transfert de fichiers), JDBC (accès à la base de données) etc.

QueryString (chaine de requête) fait partie d'un localisateur de ressources uniformes (URL) qui attribue des valeurs aux paramètres spécifiés. Une chaîne de requête comprend généralement des champs ajoutés à une URL de base par un navigateur web ou une autre application cliente, par exemple dans un cadre d'un code HTML, en choisissant l'apparence d'une page ou en sautant vers des positions dans le contenu multimédia.

Toutes les URL sont conformes à la syntaxe URI. Cette syntaxe consiste en une séquence hiérarchique de 5 éléments : URI = schema:[//authority]path[?query][#fragment].

- Le schéma précise le protocole utilisé
- Authority et path précisent le chemin d'accès à la ressource
- La query est une partie de l'URL contenant les paramètres transmis par le navigateur au serveur web
- Le fragment correspond à une ancre spécifique de la page.

5. 5. 3. Entêtes et cookies

Les entêtes sont des valeurs transmises dans la partie en-tête de la requête HTTP. Elles sont transmises sous la forme « Nom:valeur ». Par exemple :

- Host : spécifie le nom de domaine du site s'il y'a plusieurs sites à la même adresse IP.
- Connection : définit le type de connexion
- Close : la connexion est fermée après réponse
- Content-type : spécifie le type MIME du corps de la requête (HTML, Json, etc…)
- Content-Lengh : précise la longueur du corps de la requête
- Cookie : permet d'envoyer au serveur les cookies qui ont été enregistrés. Un cookie a toujours un nom et une valeur.

Les types (MIME) Multipurpose Internet Mail Extensions sont des identifiants de format de données sur internet en deux parties. L'IANA, Internet Assigned Numbers Authority (autorité qui attribue les adresses IP, entre autres) définit actuellement les types suivants : application, audio, font, image, message, model, multipart, text et video, ainsi que les quatre arbres de sous-types suivants : arbre standard (sans préfixe), arbre éditeur (préfixe vnd.), arbre personnel (préfixe prs.) et arbre non enregistré (préfixex.). Par exemple, le type de médias text/html;charset=UTF-8 est composé du type text, du sous type html de l'arbre standard et du paramètre optionnel charset=UTF-8.

Le cookie est défini par le protocole de communication HTTP comme étant une suite d'information envoyée par un serveur HTTP à un client. Ce dernier le retourne lors de chaque interrogation du même serveur sous certaines conditions.

Les cookies permettent aux développeurs de sites web de conserver des données utilisateur afin de faciliter la navigation et permettre aussi l'utilisation de certaines fonctionnalités. Ils sont envoyés en tant qu'en-tête HTTP par le serveur au navigateur qui le renvoie inchangé à chaque fois qu'il accède au serveur. Un cookie peut être utilisé pour une authentification, une session et pour stocker une information spécifique de l'utilisateur. Si une durée de validité est précisée, il est stocké dans le cache du navigateur jusqu'à cette date. À défaut, il disparaît lorsque le navigateur est fermé.

5. 5. 4. Codes HTTP courants

Les codes HTTP sont classés par type. Ci-dessous, les codes les plus usuels :

- Les codes **20x** → Réussite. Ces codes indiquent le bon déroulement de la transaction. Par exemple, « 200 = OK : la requête a été accomplie correctement »
- Les codes **30x** → Redirection. Ces codes indiquent que la ressource n'est plus à l'emplacement indiqué. Par exemple : « 301 = MOVED : les données demandées ont définitivement été transférées à une nouvelle adresse » et « 302 = FOUND : les données demandées doivent être redemandées avec une nouvelle URL ».
- Les codes **40x** → Erreurs dues au client. Ces codes indiquent que la requête est incorrecte. Par exemple : « 400 = BAD REQUEST : La syntaxe de la requête est mal formulée ou est impossible à satisfaire », « 401 = UNAUTHORIZED : le paramètre du message donne les spécifications des formes d'autorisation acceptables. Le client doit reformuler sa requête avec les bonnes données d'autorisation », « 403 = FORBIDDEN : l'accès à la ressource est interdit » et « 404 = NOT FOUND : Le serveur n'a rien trouvé à l'adresse spécifiée ».
- Les codes **50x** → Erreurs dues au serveur. Ces codes indiquent qu'il y a eu une erreur interne du serveur. Par exemple « 500 = INTERNAL SERVER ERROR : le serveur a rencontré une condition inattendue qui l'a empêché de donner suite à la demande ».

5. 5. 5. HTTP et TLS/SSL

Afin de sécuriser les échanges HTTP, une couche SSL/TLS est utilisée pour chiffrer les échanges entre le client et le serveur. Le protocole SSL est basé sur le principe du chiffrement asymétrique, couple clef privée/clef publique.

La propriété des algorithmes asymétriques est qu'un message chiffré par une clef privée sera lisible par tous ceux qui possèdent la clef publique correspondante. À l'inverse, un message chiffré par une clef publique n'est lisible que par le propriétaire de la clef privée correspondante.

Le problème vient de la transmission de cette clef publique. Si celle-ci n'est pas sécurisée, un attaquant peut se positionner entre l'entité et son public en diffusant de fausses clefs publiques puis intercepte toutes les communications, lui permettant d'usurper l'identité du diffuseur des clefs publiques (attaque de type man in the middle). Les certificats résolvent le problème du canal sécurisé grâce à la signature d'un tiers de confiance.

Un certificat électronique est un ensemble de données contenant au moins une clef publique, des informations d'identifications et au moins une signature. L'entité signataire (Autorité de Certification) est la seule autorité permettant de prêter confiance (ou non) à l'exactitude des informations du certificat.

Lorsqu'un diffuseur d'information veut diffuser une clef publique, il effectue une demande de signature auprès d'une autorité de certification (CSR). L'autorité de certification reçoit la clef publique et l'identité du diffuseur. Après avoir vérifié cette clef et l'identité du diffuseur par des moyens conventionnels, elle les place dans un conteneur qu'elle signe en utilisant sa clef privée. Le fichier résultant est le certificat. Il est alors renvoyé au diffuseur qui le conserve pour sécuriser un canal de communication donné (par exemple la communication entre son site internet et les utilisateurs).

Le protocole HTTPS permet d'utiliser ce mécanisme de certificat pour authentifier le serveur ou le client.

6. Généralité sur les annuaires électroniques

Dans le cadre de la mise en place d'une authentification unique, une grande quantité d'informations relative aux acteurs sont manipulées. Ces informations doivent être stockées dans une base d'information afin d'en assurer la cohérence, la sécurité, l'accessibilité et la centralisation.

Ainsi la base d'information étant beaucoup plus sollicitée en lecture, le choix doit être porté sur les annuaires LDAP qui sont réputés être performants pour les accès en lecture au détriment des bases de données.

6. 1. Qu'est-ce qu'un annuaire ?

Un annuaire se présente comme une base de données spécialisée, c'est-à-dire qu'on peut y mettre des informations et les consulter. Sa principale caractéristique est l'accès aux données par des recherches multicritères.

Des données de natures très diverses peuvent figurer dans un annuaire électronique :

- Répertoire téléphonique
- Liste des ressources matérielles de l'entreprise (inventaire)
- Base clients
- Base fournisseurs
- Base utilisateurs …

Tout l'intérêt de l'annuaire apparait lorsqu'il est relié avec d'autres applications telles que des applications de messagerie, de travail de groupe et des services applicatifs Web en Intranet et extranet.

6. 2. A quoi sert un annuaire ?

Le besoin d'une base d'annuaire central découle de deux problèmes majeurs :

Quelque soit le type de données, l'annuaire a pour objectif d'une part de les rendre accessible aux personnes concernées (ayant le droit de les consulter), d'autre part de permettre à ces mêmes personnes de mettre à jour certaines de ces données.

Il permet aussi de centraliser les informations et les rendre disponibles, via le réseau, à des applications, des systèmes d'exploitation ou des utilisateurs. Ainsi, il facilite la gestion et la mise en place d'une politique de sécurité commune sur ces informations.

6. 3. Différence entre annuaire et base de données

Bien qu'un annuaire soit comparable à une base de données pour un grand nombre de fonctionnalités, il en diffère sur de nombreux points.

- Un annuaire est très performant en consultation (c'est-à-dire en lecture ou en recherche). En revanche, un annuaire n'est pas très adapté pour des opérations de mises à jour

fréquentes (écriture). Les données contenues dans un annuaire sont en effet beaucoup plus pérennes, et il est donc totalement inutile d'optimiser les fonctions de mise à jour.

- Une base de données doit, généralement supporter des applications qui la remettent constamment à jour. Cela signifie que la fonctionnalité écriture dans une base de données est importante et doit par conséquent être optimisée.
- Un annuaire LDAP organise les données de manière arborescente, tandis que les bases de données le font au sein de tableaux à deux dimensions.

6. 4. Protocole LDAP

LDAP (Lightweight Directory Access Protocol, ou Protocole d'accès aux annuaires légers) est un protocole standard permettant de gérer des annuaires, c'est-à-dire d'accéder à des bases d'informations sur les utilisateurs d'un réseau par l'intermédiaire du protocole TCP/IP. Les bases d'informations sont généralement relatives à des utilisateurs, mais elles sont parfois utilisées à d'autres fins comme pour gérer du matériel dans une entreprise.

Il est normalisé par l'IETF (Internet Engineering Task Force) dans sa version 3 en 1997, LDAP est un protocole client-serveur et offre quatre modèles prédéfinis. L'objectif de cette modélisation est de favoriser le partage et de simplifier la gestion des informations concernant des personnes, et plus généralement toutes les ressources de l'entreprise, ainsi que des droits d'accès des utilisateurs sur ces ressources.

Les modèles LDAP :

- Modèle d'information : il définit la nature des données stockées dans l'annuaire. Celles-ci sont constituées d'un ensemble d'enregistrements dans lequel chaque enregistrement est l'instance d'une classe d'objet comportant une série d'attributs. Chaque attribut est défini par un type (entier, chaine de caractère, etc.) et contient un ou plusieurs valeurs, qui peuvent être obligatoires ou non.
- Modèle de désignation : il définit la façon d'organiser et de désigner les entrées dans l'annuaire. Les données sont classées de façon hiérarchique dans un arbre, reflétant en général l'organisation de l'entreprise. Le nom d'une entrée contient le nom des différents nœuds de l'arbre puis l'identifiant de celle-ci.
- Modèle des services : il décrit les fonctions offertes par un annuaire LDAP. Ces fonctions comprennent la recherche et la consultation des entrées de l'annuaire, la mise à jour de celui-ci, et l'authentification des utilisateurs auprès de ces services.

- Modèle de sécurité : il définit la manière de s'identifier de façon sécurisée à un annuaire LDAP et le concept des droits d'accès aux différents objets de l'annuaire. La gestion de la sécurité est pointue car elle permet de définir des droits d'accès non seulement au niveau d'un objet, mais aussi au niveau d'un attribut de cet objet.

6. 4. 1. Les classes d'objets

La classe d'objet d'un annuaire LDAP permet de spécifier les attributs possibles pour une entrée représentant un objet particulier du monde réel. Cette liste d'attributs est subdivisée en deux parties :

- La première liste définit quels sont les attributs obligatoires que doit posséder une entrée,
- La seconde liste caractérise les attributs optionnels que peut posséder une entrée.

En effet, les classes d'objet d'un annuaire LDAP sont définies par plusieurs caractéristiques. En premier lieu, une classe d'objet possède au moins un nom de façon à l'identifier facilement. En second lieu, indiquer si la classe est structurelle ou auxiliaire. Les classes structurelles sont les seules qui permettent de construire des entrées. Ce qui signifie que toute entrée doit posséder au moins une classe structurelle. Les classes auxiliaires permettent de rajouter facilement un certain nombre d'attributs à une entrée existante.

Les classes d'objets ont une autre particularité très intéressante. On peut construire de nouvelles classes à partir de classes existantes. Autrement dit, la notion d'héritage existe également pour les classes d'objets.

6. 4. 2. Les schémas

Les annuaires permettent de stocker et de gérer des identités, on y trouve des objets représentants des personnes et des attributs permettant d'identifier et de définir la personne comme le nom, prénom, téléphone, etc. L'ensemble des types d'objets possible dans l'annuaire, et pour chaque objet l'ensemble des attributs utilisables est défini dans l'annuaire. Le schéma contient la syntaxe et la liste des attributs connus de l'annuaire.

- Extension de Schéma : parfois il arrive qu'on veuille stocker dans l'annuaire des informations de nature particulières pour les besoins propres à ses applications. Si le schéma d'origine ne le permet pas, on peut réaliser une extension de schéma. L'extension de schéma consiste à définir pour un annuaire de nouveaux types d'objets, ou de nouveaux attributs pour un type d'objet existant.

- Attributs LDAP : un attribut est une valeur contenue dans une entrée. Une entrée peut contenir plusieurs attributs.
- Format d'échange de données LDIF : LDIF (LDAP Data Interchange Format ou Format d'échange des données LDAP) permet l'exportation ou l'importation des données depuis ou vers un annuaire LDAP. Il décrit le format de fichier de texte qui contient tout ou une partie des données d'un annuaire.

6. 5. Active Directory

Active Directory (AD) est un annuaire implémenté sur les systèmes d'exploitation Microsoft depuis Windows Server 2000. Comme pour les annuaires, il s'appuie sur la norme LDAP. Beaucoup d'améliorations ont été apportées depuis. Il comprend généralement l'ensemble des comptes nécessaires à l'authentification des ordinateurs et utilisateurs d'une entreprise.

L'objectif principal d'Active Directory est de fournir des services centralisés d'identification et d'authentification sur un réseau d'ordinateurs utilisant tout type de système d'exploitation.

Il permet également l'attribution et l'application de stratégies ainsi que l'installation de mises à jour critiques par les administrateurs. AD répertorie les éléments d'un réseau administré tels que les comptes des utilisateurs, les serveurs, les postes de travail, les dossiers partagés, les imprimantes etc. Un utilisateur peut ainsi facilement retrouver des ressources partagées, et les administrateurs peuvent contrôler leurs utilisations grâce à des fonctionnalités de distribution, de duplication, de partitionnement et de sécurisation de l'accès aux ressources répertoriées. Si les administrateurs ont indiqué les attributs convenables, il sera possible d'interroger l'annuaire pour obtenir, par exemple, « toutes les imprimantes en couleur à une étage d'un bâtiment ».

Le service d'annuaire Active Directory peut être mis en œuvre sur n'importe quelle version Windows Server.

Active directory stocke ses informations et paramètres dans une base de données distribuée sur un ou plusieurs contrôleurs de domaine, la réplication étant prise en charge nativement. La taille d'une base Active Directory peut varier de quelques centaines d'objets, pour de petites installations, à plusieurs millions d'objets, pour des configurations volumineuses.

Le contrôleur de domaine Active Directory contient des composants physiques et logiques. Les composants physiques vont englober plusieurs éléments clés dans un domaine AD. Ces derniers peuvent être matériels ou logiciels :

- Le contrôleur de domaine contient une copie de données de l'Active Directory
- La base de données et le dossier « sysvol », qui vont contenir l'ensemble des informations d'Active Directory (propriétaires des comptes utilisateurs, ordinateurs, …). Chaque contrôleur de domaine du AD en contient une copie.
- Le serveur catalogue global, qui contient une copie partielle des attributs (nom, prénom, adresse de l'utilisateur, …) des objets de la forêt. Il permet d'effectuer des recherches rapides sur un des attributs d'un objet ou d'un domaine différent de la forêt.

Tous ces composants fonctionnent avec des composants logiques, ces derniers permettent de mettre en place la structure AD souhaitée. Ainsi, il est possible de trouver les composants suivants :

- Les partitions, qui sont des sections de la base de données Active Directory. Nous pouvons y trouver la partition de configuration, la partition de domaine, la partition DNS, …
- Le schéma Active Directory contient les attributs de tous les objets qui peuvent être créés dans AD.
- Le domaine permet la mise en place d'une limite administrative pour les objets utilisateurs et ordinateurs.
- Une arborescence de domaine qui contient une suite de domaine partageant un espace de noms DNS contigu.
- La forêt Active Directory contient l'ensemble des domaines Active Directory
- Le site Active Directory permet de découper un domaine en plusieurs parties, ceci afin de limiter et contrôler la réplication entre deux sites distants.
- L'unité d'organisation permet d'appliquer une stratégie de groupe mais également de mettre en place une délégation.

6. 5. 1. Domaine Active Directory

Un domaine Active Directory est un regroupement logique de comptes utilisateurs, ordinateurs ou de groupes. Les objets qui y sont créés, sont stockés dans une base de données présente sur tous les contrôleurs de domaine Active Directory. Cette base de données peut stocker plusieurs types d'objets :

- Le compte utilisateur permet d'effectuer une authentification et d'autoriser des accès aux différentes ressources partagées. Il représente une personne physique ou une application.
- Le compte ordinateur permet d'authentifier la machine sur laquelle l'utilisateur ouvre une session.
- Les groupes permettent de regrouper des comptes utilisateurs et ordinateurs dans le but de leur autoriser l'accès à une ressource, de mettre en place une délégation…

6. 5. 2. Les unités d'organisation

Les unités organisationnelles (OU - Organizational Unit) sont des objets conteneurs qui permettent le regroupement de comptes d'utilisateur ou d'ordinateur. La création de ce type d'objet est généralement opérée afin d'assigner une stratégie de groupe à l'ensemble des objets présents dans le conteneur d'où sa fonction première. Sa deuxième fonction est la mise en place d'une délégation afin de permettre à une personne différente de l'administrateur de gérer les objets présents dans le conteneur.

Ainsi les unités organisationnelles représentent une hiérarchie logique dans le domaine Active Directory (il est possible de les imbriquer, on parle alors d'OU parent et d'où enfant). Il est par exemple possible de créer une unité d'organisation par ville ou même par type d'objet (utilisateur, ordinateur, …). Lors de la création du domaine, des dossiers système et des unités d'organisation sont par défaut présents

- Dossier Builtin : stocke par défaut les administrateurs, opérateurs d'impression, …
- Dossier Utilisateurs : dossier par défaut lors de la création d'un nouvel utilisateur. Il contient par défaut le compte administrateur et les différents groupes administrateurs (admins du domaine, administrateurs de l'entreprise…).
- Dossier Computers : répertoire par défaut, les comptes ordinateurs sont positionnés à cet endroit lors de l'ajout de nouveaux postes de travail.
- Unité d'organisation Domain Controllers : emplacement par défaut pour les comptes des contrôleurs de domaine. Cette unité organisationnelle est la seule présente lors de la création du domaine. La stratégie de groupe, Default Domain Controller Policy est positionnée sur cette unité d'organisation.

6. 5. 3. La forêt Active Directory

Une forêt est constituée d'un ou plusieurs domaines Active Directory. On parle de domaine racine pour le premier domaine de la forêt. De plus, ce dernier donne son nom à la forêt. Dans notre maquette le domaine racine est « education.sn », la forêt a donc le nom de ce dernier, soit « education.sn ». On trouve dans une forêt Active Directory une seule configuration et un seul schéma, ceux-ci sont partagés par l'ensemble des contrôleurs de domaine présents dans la forêt. Elle a également pour but la mise en place d'une frontière de sécurité, les autres forêts n'ont aucun droit sur elle et aucune donnée n'est répliquée à l'extérieur de la forêt.

Une forêt Active Directory est donc composée d'une suite de domaines appelée également une arborescence de domaines, ces derniers partagent un espace de noms contigus. La relation entre les domaines d'une même arborescence est de type parent/enfant. Un domaine qui dispose d'un espace de noms différent fait partie d'une arborescence différente.

Le domaine représente lui aussi une limite de sécurité car l'objet utilisateur qui permet l'authentification d'une entité (personne physique de l'entreprise, …) est défini par domaine. Ce dernier contient au moins un contrôleur de domaine, deux étant recommandés pour des raisons de disponibilité. Ce type de serveur a la responsabilité de l'authentification des objets utilisateurs et ordinateurs dans un domaine AD.

6. 5. 4. Le schéma Active Directory

Le schéma Active Directory est un composant qui permet de définir les objets ainsi que leurs attributs pouvant être créés dans Active Directory. Lors de la création d'un nouvel objet, le schéma est utilisé afin de récupérer les attributs de ce dernier et leurs syntaxes (booléen, entier, ...).

Conclusion

Le système d'authentification unique est devenu primordiale pour toutes les entreprises et les administrations. Centralisant ainsi toutes les authentifications en un point et utilisant parfois des informations confidentielles qui sont échangées entre fournisseur de service et fournisseur d'identité, son déploiement doit être accompagné avec une fortification de la sécurité du SI. Nous avons vu dans ce chapitre l'ensemble des notions, méthodes et protocoles permettant le déploiement d'un système d'authentification unique pour les applications web.

Chapitre 3 : Etat de l'art et études comparatives entre quelques solutions SSO

Introduction

L'authentification unique et centralisée est au cœur du développement et de l'innovation numérique. Il permet entre autres de décharger la phase d'authentification aux applications et la déléguée à un tiers serveur qui permet de l'unifier sur un ensemble d'application via des cookies de sessions sécurisées. Il existe plusieurs solutions payantes et open sources pour l'authentification unique.

Parmi ces solutions nous pouvons citer :

1. Evidian

Evidian est un logiciel de gestion des accès et des identités pour les entreprises et les professionnels. Il est payant, développé et maintenu à jour par **Atos**.

Evidian remplace le couple login et mot de passe des utilisateurs par une authentification unique, tel qu'un mot de passe, la biométrique, une carte à puce ou un badge magnétique. L'accès est immédiat, que les applications soient internes ou externes à l'entreprise. Le résultat pour les utilisateurs est un accès simplifié à leurs applications : ils se conforment automatiquement à la politique de sécurité. Et en supprimant les tâches administratives fastidieuses, Evidian simplifie la gestion du système d'information. Il propose des API afin de faciliter l'intégration des applications et le SaaS (SSO as a Service).

Logiciel en mode SaaS, est donc compatible avec la plupart des systèmes d'informations d'entreprise ainsi qu'avec la plupart des systèmes d'exploitation comme Windows, Mac OS et Linux car il est accessible depuis un navigateur web.

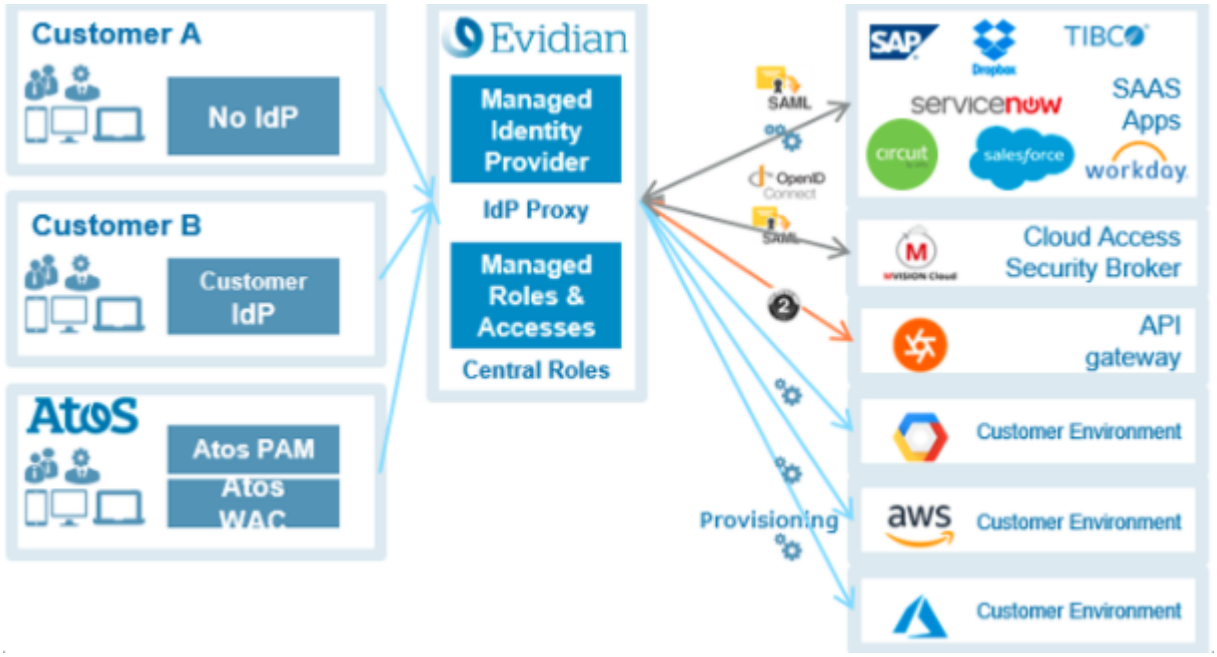


Figure 8 : Protocoles et environnements compatibles avec Evidian

Source : https://www.evidian.com/fr/produits/web-sso/

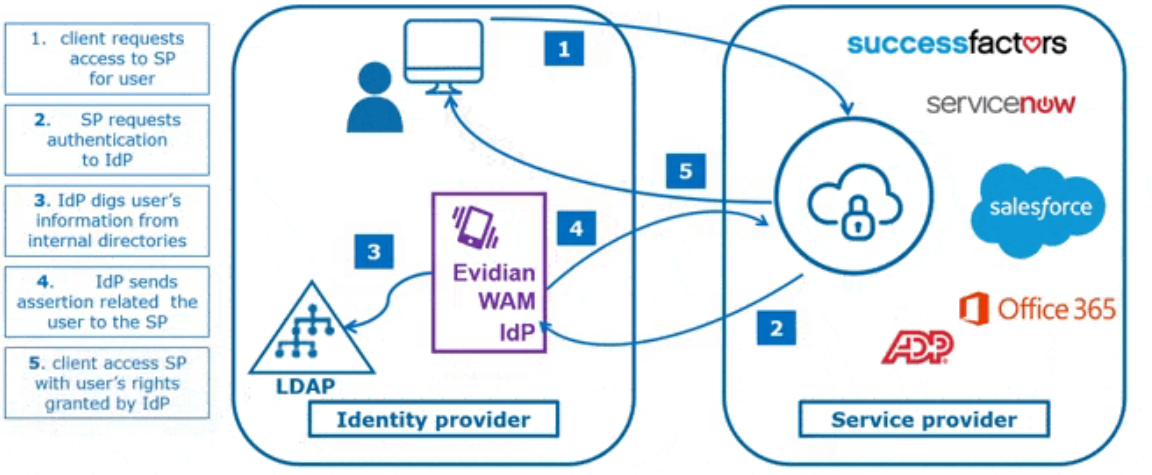


Figure 9 : Diagramme de séquence de l'authentification avec Evidian

Source : https://www.evidian.com/fr/produits/web-sso/

1- L'utilisateur à travers un navigateur envoie une requête à la ressource partagée

2- L'utilisateur est redirigé sur le portail Evidian pour qu'il s'authentifie
3- Evidian vérifie l'identité de l'utilisateur sur l'annuaire LDAP
4- Un code d'autorisation et un token d'accès sont envoyés à la ressource protégée
5- Le token est envoyé au navigateur et l'accès est établie

Ce token est utilisé pour accéder aux autres applications sur une durée définie (la session).

1. 1. Avantages

- Contrôle d'accès
- Authentification unique
- Autorisation multifactorielle
- Gestion des mots de passe
- Service d'annuaire
- Synchronisation des données
- Sauvegarde des données

1. 2. Coût

Ce logiciel propose une version d'essai de 15 à 30 jours et le prix dépend du profil choisi (Standard, Pro, Entreprise, Premium), de l'entreprise demandeur, du nombre de ressources à protéger et du nombre d'utilisateur. Le prix est négocié à la demande.

2. Systancia Access

Systancia Access est une solution de contrôle des accès par authentification unique pour l'accès aux applications de manière simples et sécurisés. Systancia Access, anciennement Avencis SSOX, est un produit permettant de gérer de manière sécurisée et transparente l'authentification des utilisateurs. On-premise ou cloud (client lourd, web ou mobile). Il propose une traçabilité complète, du SSO pour des applications client-serveur ou web avec différents modes d'authentification : authentification forte, adaptative ou fédérative (fédération d'identités, par gestion et injection de token dans les applications).

Les modules principaux faisant partie de la suite d'outils SSO sont :

- e-SSO : SSO pour tous les types d'applications qui intègrent la notion de délégation, de comptes multiples.
- Web-SSO : solution qui permet l'accès aux applications web internes ou externes à partir d'un ordinateur, d'une tablette, d'un smartphone, etc.

- SSOX Mobile : qui permet l'accès web interne ou externe avec une authentification unique
- Avencis SSOX Fédération permet de gérer les utilisateurs et les autorisations dans un tiers référentiel.

Cette solution simplifie et uniformise donc les accès aux applications tout en renforçant les politiques de mots de passe applicatif en limitant les contraintes sur les utilisateurs.



source: www.ilex-international.com

Figure 10 : Single Sign-On de Systancia

L'accès aux applications est protégé par le SSO qui fédère l'identité de ses utilisateurs à un annuaire LDAP ou une base de données.

2. 1. Avantages

- L'authentification forte multi-facteurs et multi-device
- Unification de l'authentification forte, du Web Access Management, de sa fédération des identités et de l'entreprise SSO
- Fonctionnalités d'authentification Out of Band et Bluetooth
- Utilisation des protocoles standards de fédération d'identités : SAML, Openid connect, OAuth2 et WS-Federation
- Adaptabilité aux nouveaux usages : Kiosque, Roaming, terminaux, cloud
- Portabilité des cinématiques SSO sur les tablettes et Smartphones

2. 2. Coût

Systancia en plus d'être une solution SSO, il est un logiciel de sécurité informatique utilisé par les professionnels, les entreprises, les administrations, etc. Le tarif du logiciel est disponible sur demande.

3. Keycloak

Keycloak est un logiciel open source permettant d'instaurer une méthode d'authentification unique à travers la gestion par identité et par accès. Initialement développé par les équipes JBoss, le projet est depuis mars 2018 sous la gérance de Red Hat qui l'utilise en amont de sa solution RH-SSO.

Keycloak se définit comme une application rendant possible la sécurisation de n'importe quelle application web moderne avec un apport minimum en termes de code.

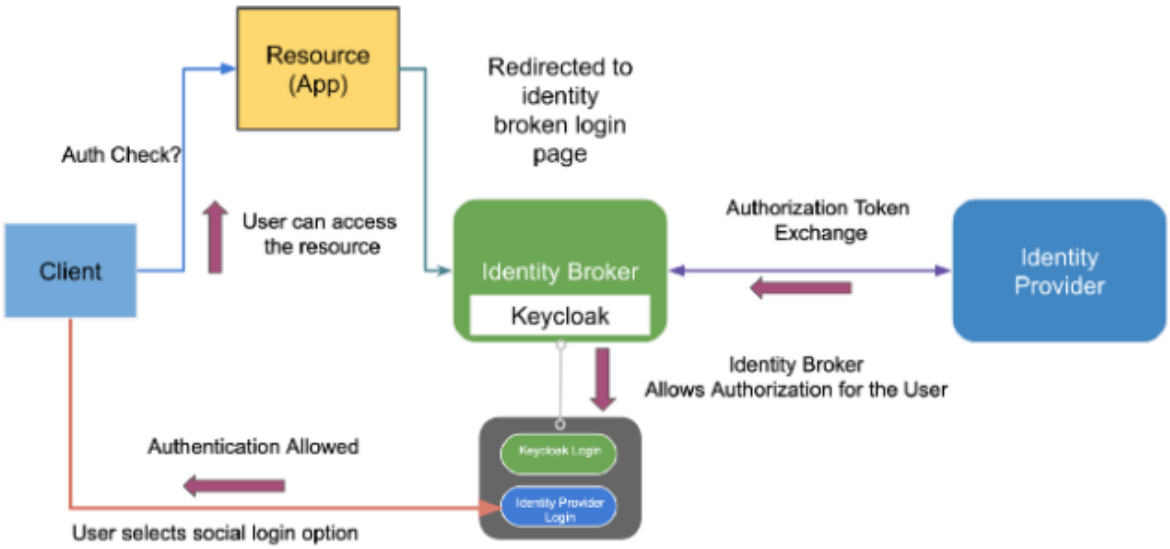


Figure 11 : Diagramme de séquence de l'authentification de Keycloak

Source : https://medium.com/@didelotkev/facebook-as-identity-provider-in-keycloak-cf298b47cb84

- L'utilisateur tente d'accéder à l'application protégée
- S'il n'est pas authentifié, il est redirigé vers la page de connexion Keycloak.
- L'utilisateur peut s'authentifier à l'aide de la connexion Keycloak ou utiliser le bouton de connexion via Facebook, ou GitHub, ou Google, etc.
- L'identité choisie échangera un jeton d'autorisation du fournisseur d'identité, qui demande à l'utilisateur d'approuver les autorisations.

- Une fois, l'approbation est fournie par l'utilisateur. Keycloak créera une session d'authentification pour l'utilisateur.

- L'utilisateur est maintenant autorisé à accéder à la ressource demandée.

3. 1. Avantages

- Inscription dynamique des utilisateurs
- Authentification unique sur l'ensemble d'un domaine
- Authentification en deux étapes
- Intégration avec un annuaire LDAP
- Possibilité de customisation des interfaces utilisateurs par un système de thème avec traductions et feuilles de style en cascade
- API pour piloter les fonctionnalités sans passer par l'interface utilisateur

3. 2. Coût

Keycloak est un logiciel à code ouvert. Sa mise en place nécessité des compétences en codages et quelques notions en sécurité. Il peut être déployé dans un containeur docker dernière un reverse proxy. Son coût est moindre mais peut-être important s'il est déployé par un particulier.

4. LemonLDAP::NG

Lemonldap-ng est une solution open source, créé en 2004 par la Gendarmerie nationale française. Il fournit une solution d'authentification unique distribuée avec gestion centralisée des droits sous licence GPL.

Lemonldap-ng (LLNG), solution essentiellement Web-SSO mais pouvant également servir aux applications client lourd. Il peut fonctionner en mode reverse proxy, applicatif, fédération d'identité ou mixte.

LLNG est compatible avec la plupart des protocoles de fédération d'identité (Openid Connect, SAML, CAS, OAuth 2). Il peut être fournisseur de services ou d'identités. Une autre particularité spécifique et peu courante est la possibilité de l'utiliser comme relais protocolaire (Protocol Relay in Proxy) entre ses différents systèmes de fédération d'identité très hétérogènes ou entre deux instances de différents systèmes de fédération d'identité très hétérogènes ou entre deux instances de LLNG.

LLNG gère et propose les services d'authentification unique et d'autorisation en fournissant des en-têtes HTTP aux applications protégées pour qu'elles puissent exploiter des informations d'identité ou autres. Ces en-têtes nommés cookies de session ou encore tokens d'accès permettent également de contrôler les accès et d'imputer les actions aux utilisateurs. Il est donc une solution AAA complète.

Les autorisations d'accès sont construites par association d'une expression et d'une règle. Les expressions régulières utilisent le moteur interne de Perl (PCRE) et sont appliquées à chaque URL. La règle est calculée pour déterminer si l'utilisateur est autorisé ou non à accéder à la ressource demandée.

Tous les accès au portail, d'ouverture de sessions et les accès aux applications protégées sont tracés et journalisés. LLNG fonctionne avec les serveurs web Apache, Nginx (protocoles FastCGI/uWSGI) ou compatibles Plack (Starman, Corona, etc).

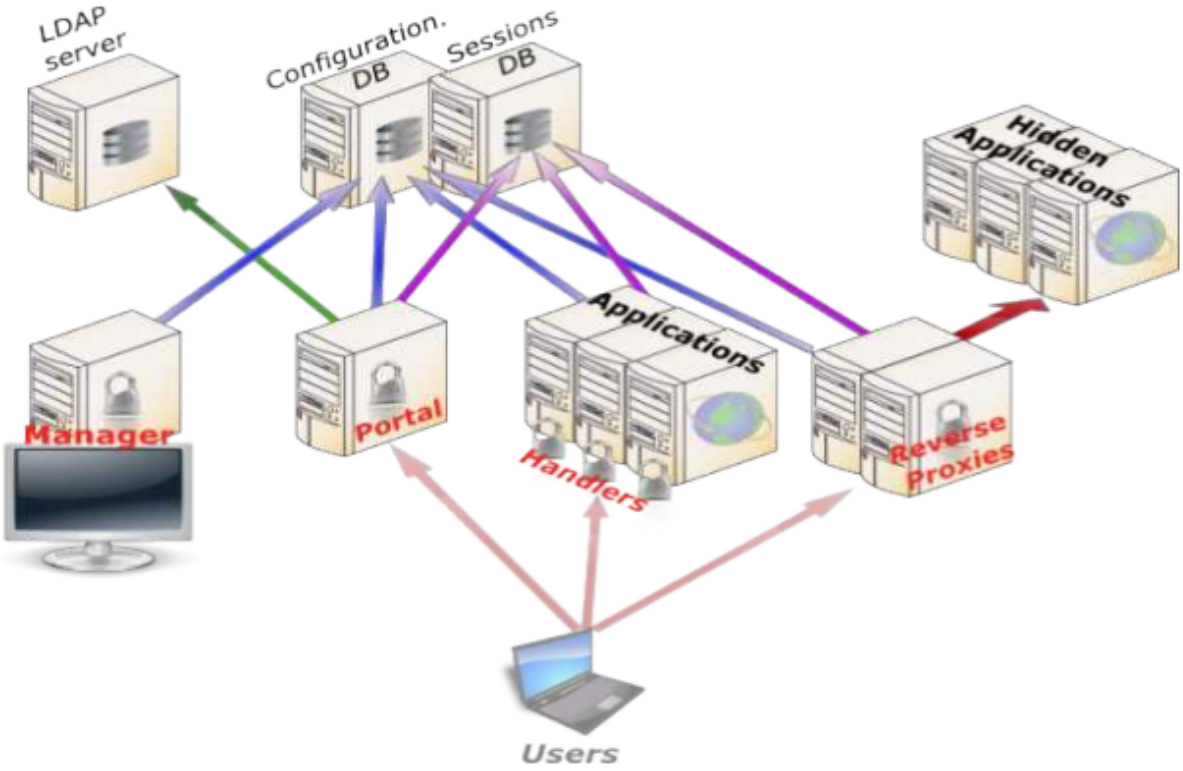


Figure 12 : Architecture globale de Lemonldap-ng

Source : https://lemonldap-ng.org/documentation/presentation

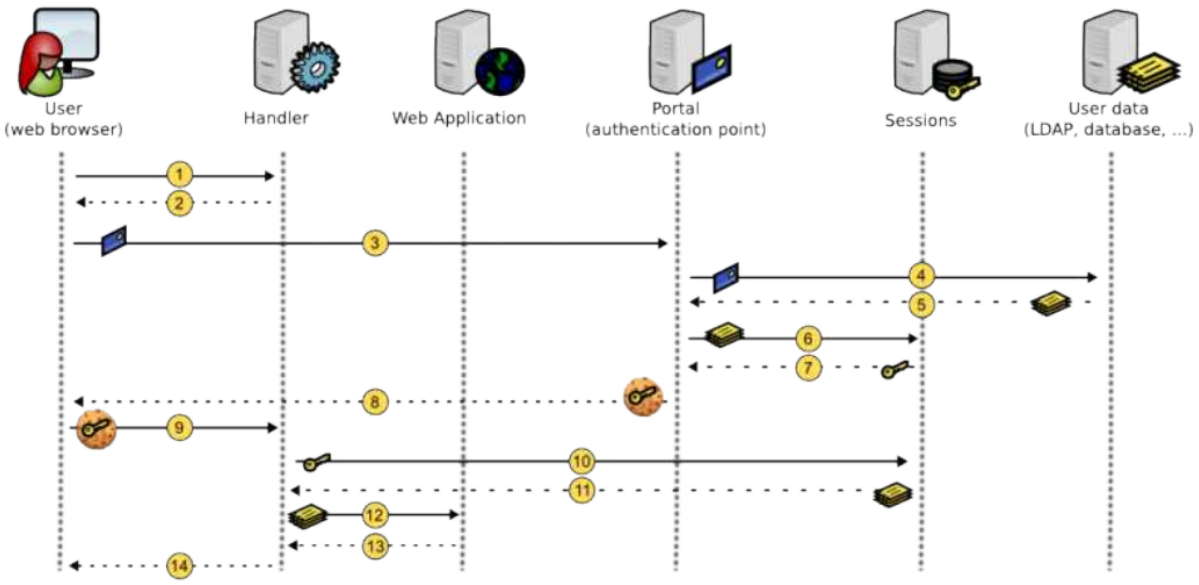


Figure 13 : Diagramme de séquence de l'authentification avec Lemonldap-ng

Source : https://lemonldap-ng.org/documentation/presentation

1- Un utilisateur tente d'accéder à une application protégée. Sa requête est interceptée par le handler

2- Si le handler ne détecte pas de cookie SSO, il redirige l'utilisateur vers le portail afin qu'il s'authentifie

3- L'utilisateur s'authentifie auprès du Portail

4- Le Portail vérifie l'authentification

5- Si l'utilisateur s'authentifie avec succès, le Portail collecte les informations sur l'utilisateur auprès de la base des utilisateurs

6- Le Portail crée ensuite une session pour sauvegarder les informations sur l'utilisateur

7- Le Portail récupère la clef de session

8- Le Portail crée un cookie SSO contenant uniquement la clef de session obtenue précédemment

9- L'utilisateur est redirigé vers l'application protégée muni de son cookie de session

10- Le Handler récupère la clef de session transmise par le cookie SSO et collecte les informations utilisateur

11- Le Handler sauvegarde dans son cache (15s) les données de l'utilisateur

12- Le Handler vérifie les règles d'accès et transmet les informations de l'utilisateur à

l'application

13- L'application envoie la réponse au Handler

14- Le Handler transmet la réponse de l'application à l'utilisateur

Ensuite, le Handler vérifiera le cookie SSO et le droit d'accès à chaque requête HTTP.

4. 1. Avantages

- Journaliser tous les accès aux applications
- Authentification unique avec une politique de renforcement des mots de passes et des règles d'accès spécifiques
- Synchronisation avec un annuaire centralisé dans un domaine ou plusieurs domaines
- Authentification double facteur

4. 2. Coût

Ce logiciel peut être implémenté et mis en production par un spécialiste qui le maitrise et faisant parti des ressources humaines de l'entreprise sans coût supplémentaire. Il engendre un coût en voulant exploiter tous ses avantages avec l'aide d'expertise d'un particulier ou d'une structure spécialisée en intégration de solution open source.

5. Microsoft Azure Active Directory

Microsoft Azure Active Directory est une solution cloud complète de gestion des identités et des accès qui associe des services d'annuaire de base, une gestion des accès aux applications et une protection avancée des identités.

Azure Active Directory (Azure AD) est le service de gestion de l'accès et des identités, basé sur le cloud de Microsoft. Ce dernier propose une solution SSO as a Service (SaaS) avec les comptes clients office 365 et le portail Azure pour la fédération d'identité.

5. 1. Avantages

Il permet aux utilisateurs de se connecter et d'accéder aux ressources suivantes :

- Ressources externes telles que Microsoft 365, le portail Azure et des milliers d'autres applications SaaS.
- Ressources internes telles que les applications situées sur votre réseau intranet ainsi que les applications cloud développées par votre propre organisation.

5. 2. Coût

Azure AD est proposé en quatre éditions : gratuite, applications Office 365, premium P1 et premium P2. L'édition gratuite est incluse dans un abonnement à un service commercial en ligne, par exemple Azure, Dynamics 365, Intune et Power Platform.

Le coût de l'abonnement dépend du nombre d'utilisateurs et du nombre d'applications (fournisseurs de service) à protéger. Il est négocié selon les besoins de l'entreprise et de son statut.

6. AWS SSO

AWS SSO est une solution d'authentification unique d'Amazon. Il permet de connecter, les identités des utilisateurs dans AWS une seule fois, puis de gérer les accès de manière centralisée au sein de votre organisation. Il offre un environnement d'administration propre à l'abonné pour la gestion des accès et la sécurité des applications. Il est possible de créer les identités des utilisateurs directement dans AWS SSO, ou de les importer depuis un compte Microsoft Active Directory ou d'un fournisseur d'identités qui se base sur des normes, telles que Okta Universal Directory ou Azure AD.

Avec AWS SSO, vous bénéficiez d'une expérience d'administration unifiée permettant de définir, personnaliser et attribuer des accès précis. Les utilisateurs bénéficient d'un portail utilisateur pour accéder à tous les comptes AWS et applications cloud qui leurs sont attribués. Il est possible de configurer AWS SSO de manière flexible afin de l'exécuter en parallèle ou de remplacer la gestion des accès aux comptes AWS via AWS IAM.

AWS SSO est un service basé sur le cloud qui simplifie la gestion de l'accès SSO aux comptes AWS, aux applications métiers courantes et aux applications compatibles avec le protocole SAML 2.

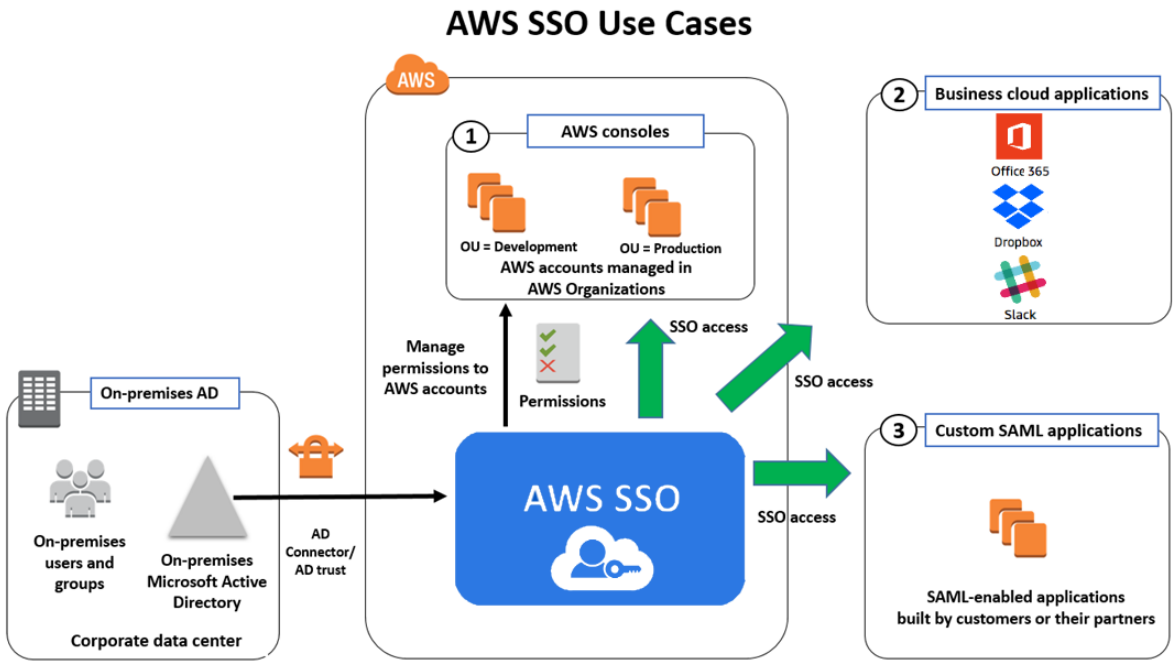


Figure 14 : Cas d'utilisation de AWS SSO

Source :

https://www.google.com/url?sa=i&url=https%3A%2F%2Faws.amazon.com%2Fblogs%2Fsecurity%2Fintroducing-aws-single-sign-on%2F&psig=AOvVaw39keE9mtS0kbeAjyKV72hB&ust=1642981747714000&source=images&cd=vfe&ved=0CAsQjRxqFwoTCNCxhf7PxvUCFQAAAAAdAAAAABAq

6. 1. Avantages

- Intégration à AWS Organization
- Accès SSO à AWS et aux applications cloud
- Créer et gérer des utilisateurs et des groupes d'utilisateur
- Tirer parti de vos identités d'entreprise existantes
- Compatibilité avec les applications cloud couramment utilisées
- Faciliter la configuration et la surveillance de l'utilisation
- Coexistence entre les utilisateurs, rôles et stratégies existants
- Gestion des identités sans frais

6. 2. Coût

Le coût de l'abonnement AWS SSO dépend de la complicité du système d'information à sécuriser (architecture, nombre d'utilisateur, nombre d'applications à protéger, le nombre de domaine, des dépendances avec d'autres cloud ou fournisseurs d'identités, etc.) et de l'entreprise (son statut). Il est négocié sur demande.

7. CAS

CAS (Central Authentification Service) est un service d'authentification centralisé pour les applications web, inspiré de Kerberos et basé sur le protocole HTTP(S). CAS est une solution open source originellement développée dans les années 2000 par l'université de Yale. En 2004, CAS est passé dans le giron du groupement d'intérêts JA-SIG.

Il est un serveur d'authentification unique avec une architecture simple accessible via un navigateur, composé de servlets java, qui fonctionne sur tout moteur de servlets (Tomcat par exemple), ce qui fait ses points forts. Il est très répandu dans les intranets d'universités, mais également utilisé dans certaines entreprises.

CAS permet de faire du SSO entre plusieurs sites web, y compris dans des domaines différents, en utilisant des tokens à usage unique. Les applications n'ont jamais accès au mot de passe de l'utilisateur, et obtiennent le login de celui-ci en interrogeant le serveur CAS. Le protocole de communication utilisé est CAS, et il est possible de fournir des informations supplémentaires au login de l'utilisateur (groupes, nom, prénom, etc).

CAS peut utiliser différents types de backend en tant que base utilisateur, tels qu'un annuaire LDAP, une base de données relationnelles, des fichiers à plats, etc. Il fournit également un système de proxy, permettant à une application de transférer l'identification à une autre application, web ou non, en backend, tel qu'un web service ou un serveur IMAP.

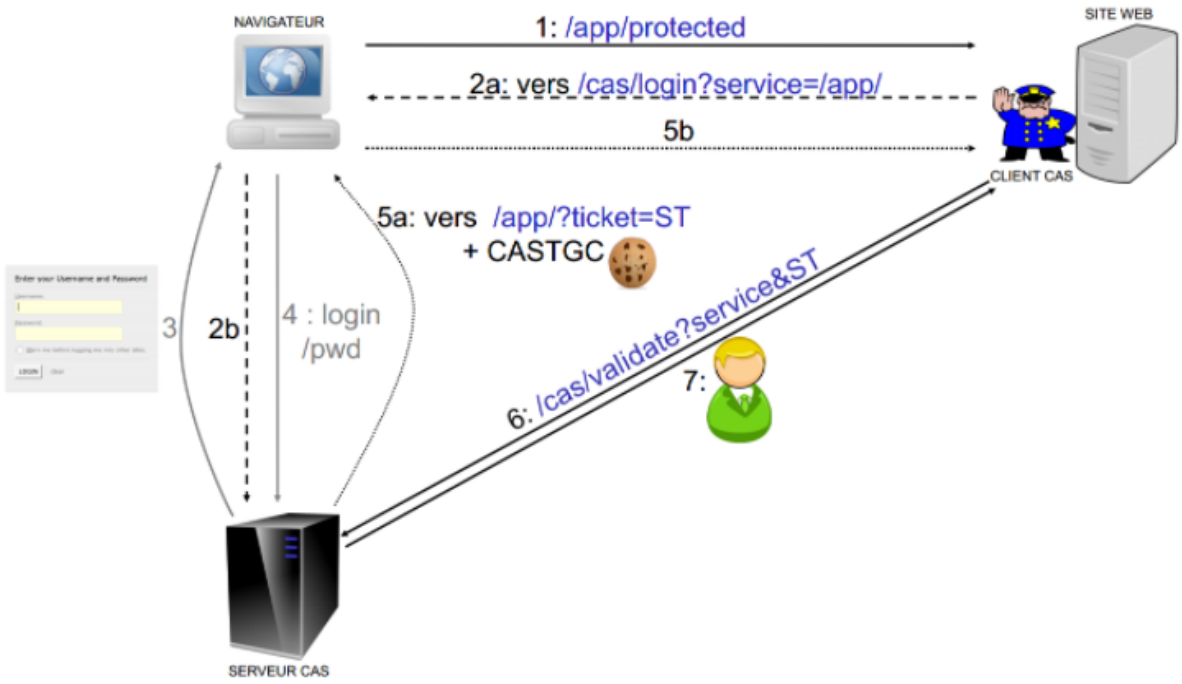


Figure 15 : Diagramme de séquence de l'authentification avec CAS

Source : https://aldian.developpez.com/tutoriels/javaee/authentification-centralisee-sso-cas/

1- Le client envoie une requête à l'application web protégée

2a- Le serveur web constate que le client n'est pas authentifié. Il le redirige vers le serveur CAS.

2b- Le client contact le serveur CAS

3- Une page d'authentification est présentée au client

4- L'utilisateur renseigne son login et son mot de passe

5a- Un cookie de session est envoyé au navigateur du client

5b- Le navigateur présente ce cookie au serveur web

6- La session est validé par le serveur CAS

7- L'utilisateur a accès à l'ensemble des applications

7. 1. Avantages

- Disponibilité, peut consommer jusqu'à 1000 requêtes d'authentification par seconde
- Interopérabilité

- Authentification unique
- Le mot de passe ne circule qu'entre le serveur d'authentification et le navigateur du client via un canal crypté
- Le token ne porte pas d'information sensible et il n'est pas rejouable
- Multi backends
- Possibilité d'une authentification inter domaine

7. 2. Coût

CAS, une solution open source, son implémentation et son déploiement peuvent être sans coût. Pour une entreprise son implémentation et son déploiement peuvent engendrer un coût, si l'entreprise fait appel à une expertise d'un particulier.

Conclusion

Les solutions d'authentification unique sont multiples et variées. Chacune présente des avantages et un coût potentiel de déploiement et d'administration. Selon le statut, les besoins, le nombre d'utilisateurs, le nombre d'applications et la sensibilité des données à protéger, de l'entreprise. Cette dernière peut choisir l'une des solutions SSO ou les combinées pour sécuriser ses ressources (applications lourdes ou applications légères).

Chapitre 4 : Justification et implémentation de la solution

Introduction

L'objectif de ce chapitre est de procéder à l'installation et au déploiement de la solution retenue pour atteindre les objectifs de notre projet en exposant les outils et technologies utilisés. Pour ce faire, nous essayerons en premier lieu de justifier le choix de la solution retenue, en deuxième lieu de présenter cette solution, en troisième lieu de définir les outils technologiques utilisés et en dernier lieu de voir le déploiement de Keycloak.

1. Choix de Keycloak

De l'étude comparative entre les différentes solutions d'authentification unique, il en ressort que ces solutions présentent un fort potentiel en termes de niveau de sécurité offert, de support d'échange d'information en toute sécurité, d'interopérabilité, etc. Cependant, toutes ces solutions bien qu'efficaces en termes de gestion d'identité et d'accès, présentent des insuffisances en termes d'interopérabilité, de disponibilités et de mise en échelle.

Après l'étude des différentes solutions d'authentification unique, notre choix s'est porté sur Keycloak pour plusieurs raisons :

- Vue le système d'information du MEN qui comprend plusieurs acteurs qui ont accès à ses plateformes et ressources numériques avec leurs identifiants :
  - Plus d'une vingtaine d'applications et plateformes web pour les acteurs du système d'information à sécuriser ;
  - Plus de 3.000.000 d'élèves dont 483.842 qui sont déjà enrôlés sur le SI et accèdent à ses plateformes et applications avec leurs identifiants ;
  - Plus de 122.408 enseignants utilisent ces plateformes ;
  - Plus de 500.000 parents d'élèves utilisent la plateforme planète parent ;
  - Plus de 3.000 agents du ministère accèdent tous les jours à certaines de ces plateformes ;
- Les fonctionnalités offertes par Keycloak répondent à nos objectifs ;
- De plus, il est une solution open source, soutenue par une large communauté active et ouverte. Il s'interface bien avec n'importe quel système d'information.
- Il peut gérer presque une infinité de comptes utilisateurs avec la possibilité de traitement de plus de 10.000 requêtes simultanées.
- La possibilité de le déployer en cluster pour une haute disponibilité.

- Il est extensible et facile à déployer.
- Il n'est pas exigeant en termes de ressource.
- Il s'exécute sur n'importe quel OS (Operating System) qui a au moins JDK 8 installé.

2. Présentation de Keycloak

Keycloak est une solution à code ouvert permettant d'instaurer une méthode d'authentification unique au travers la gestion par identité et par accès. Il se définit comme une solution de gestion des identités et des accès rendant possible la sécurisation de n'importe quelle application web avec un apport minimum en termes de code. Il implémente les protocoles standard (**Openid Connect**, **OAuth 2**, et **SAML 2**).

La première version de production de Keycloak remonte à septembre 2014, le développement ayant commencé environ une année plus tôt. Adopté en 2016, par **Red Hat** pour son projet de « Red Hat SSO » basé sur le Framework « **PicketLink** » qui fut intégrer par la suite dans le projet Keycloak. Dans certaine mesure, Keycloak peut être considéré comme un remplacement de la solution open source « Red Hat JBoss SSO » qui était au paravent remplacé par « **PickeetLink** », devenue une fonctionnalité intégrée dans Keycloak. Depuis mars 2018, JBoss.org redirige l'ancien sous-site « jbosssso » vers le site de Keycloak. Le nom « Jboss » est une marque déposée et « Red Hat » a déplacé ses noms de projets open source en amont pour éviter d'utiliser « Jboss », « Jboss » comme « Wildfly » étant un exemple plus communément reconnu.

Keycloak est sur sa version 18. Parmi ces fonctionnalités nous pouvons citer :

- Single Sign On.
- La délégation d'authentification (IdP).
- La fédération des utilisateurs.
- Les mots de passe à usage unique (OTP).
- Inscription dynamique des utilisateurs.
- Social login.
- Authentification double facteurs.
- Réinitialisation de mot de passes.
- Intégration LDAP et Kerberos.
- Customisation des interfaces utilisateurs.

Keycloak permet aux utilisateurs de s'authentifier sur son portail plutôt que sur les applications individuelles. Cela signifie que ces applications n'ont pas à gérer les formulaires de connexion, l'authentification et le stockage des utilisateurs. Une fois connectée à Keycloak, les utilisateurs n'ont pas besoin de se reconnecter pour accéder à une autre application.

Cela s'applique également à la déconnexion. Keycloak fournit une déconnexion unique, ce qui signifie que les utilisateurs ne doivent se déconnecter qu'une seule fois pour être déconnecter sur toutes les applications du domaine déléguant leur authentification à Keycloak.

Avec sa fonctionnalité « Social Login », l'activation de l'authentification à travers les réseaux sociaux est facile à ajouter via la console d'administration. Il suffit de sélectionner le réseau social que vous souhaitez ajouter. Aucun code ou modification de votre application n'est requis. Keycloak peut également authentifier les utilisateurs avec les fournisseurs d'identité Openid Connect ou SAML 2 existants sur d'autres domaines. Encore une fois, il s'agit simplement de configurer le fournisseur d'identité via la console ou l'API d'administration.

Keycloak a un support intégré pour se connecter à un ou plusieurs serveurs d'annuaire (OpenLDAP et/ou Active Directory existants) pour la fédération d'identité. Il est possible également d'implémenter votre propre fournisseur d'identité si vous avez des utilisateurs dans d'autres magasins, comme une base de données.

Keycloak est composé de deux parties :

- Une console graphique et une API REST pour l'administration
- Des adaptateurs qui permettent aux applications d'interagir avec Keycloak

Grâce à la console ou à l'API REST, les administrateurs peuvent gérer de manière centralisée tous les aspects du serveur Keycloak. Ils peuvent gérer diverses fonctionnalités, configurer le courtage de certaines authentifications et la fédération d'identité. Ils peuvent créer et gérer des applications et des services, et définir des politiques d'autorisation précises. Ils peuvent également gérer les utilisateurs, y compris les autorisations et les sessions.

Grâce à la console de gestion de compte, les utilisateurs peuvent gérer leurs propres comptes, mettre à jour le profil et modifier les mots de passe. Les administrateurs peuvent également gérer les sessions ainsi que consulter l'historique du compte.

Keycloak intègre une fonction de gestion des autorisations, cette fonctionnalité permet de créer des rôles spécifiques pour un utilisateur ou pour un groupe d'utilisateur afin de restreindre l'accès aux ressources protégées par Keycloak. Ce dernier permet également d'avoir

l'historique (date et heure de connexion, durée de la session, le login et l'adresse IP du client, etc.) de l'ensemble des connexions sur les applications protégées. Il permet aussi de restreindre les accès par adresse IP.

2. 1. Diagramme de séquence de l'authentification avec Keycloak

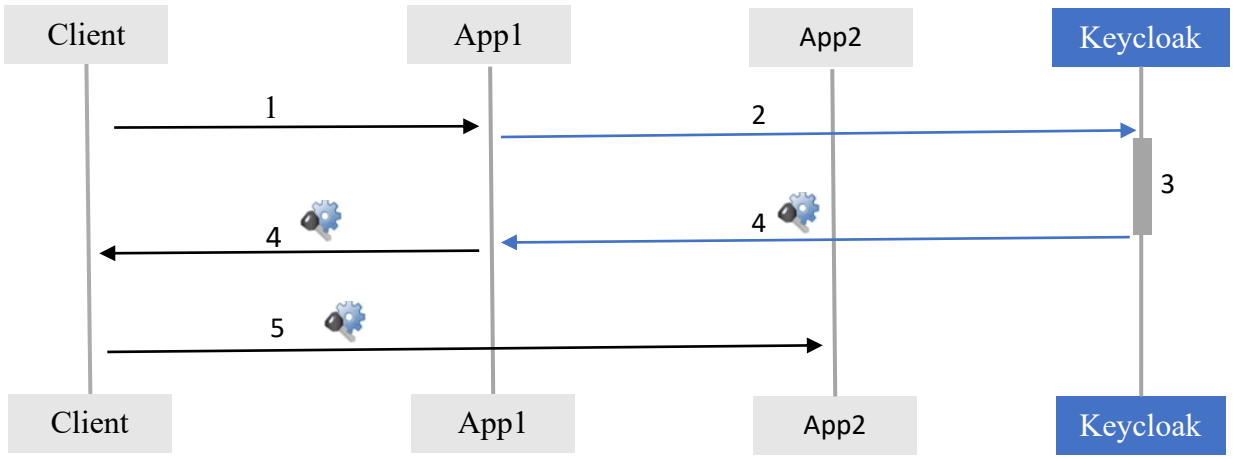


Figure 16 : Diagramme de séquence de l'authentification avec Keycloak

1- L'utilisateur tente d'accès à une application protégée par Keycloak via son navigateur.
2- L 'application redirige le client vers le portail d'authentification de Keycloak.
3- Keycloak vérifie l'identité de l'utilisateur sur sa base référentielle.
4- Si l'utilisateur s'est bien authentifié sur ce portail et qu'il a le droit d'accès à cette application, Keycloak lui délivre un token d'accès qui sera la clé passe partout pour ce navigateur sur toute application du domaine Keycloak suivant les droits d'accès de l'utilisateur. De même, toutes les informations sur l'utilisateur sont envoyées à travers ce flux excepté, son mot de passe.
5- L'application transmet le jeton au navigateur.
6- L'utilisateur utilise ce jeton pour accéder directement à une autre application dont il a accès.

3. Vue d'ensemble des outils technologiques utilisés

- Virtualisation

La virtualisation consiste à créer une version virtuelle d'un dispositif ou d'une ressource, comme un système d'exploitation, un serveur, un dispositif de stockage ou une ressource réseau. Elle peut donc, être considérée comme l'abstraction physique des ressources informatiques. En d'autres termes, les ressources physiques allouées à une machine virtuelle sont abstraites à partir de leurs équivalentes physiques. Chaque dispositif virtuel, qu'il s'agisse d'un disque, d'une interface réseau, d'un réseau local, d'un commutateur, d'un processeur ou d'une mémoire, correspond à une ressource physique sur un système informatique. Les machines virtuelles hébergées sur la machine hôte sont donc perçues par ce dernier comme des applications auxquelles il est nécessaire de dédier ou distribuer ses ressources. Cependant, il existe plusieurs types de virtualisation. Chaque type correspond à un cas d'utilisation spécifique.

- Conteneurs

Les conteneurs constituent une solution pour la gestion des ressources entre les applications. Le terme est dérivé des conteneurs d'expédition, une méthode standard pour stocker et expédier tout type de cargaison.

Le concept de base des conteneurs est né en 1979 avec la fonction Unix « chroot ». Cette dernière permet de créer plusieurs sessions isolées sur une même machine, de façon à partager parallèlement les ressources entre différents processus. Vu cette caractéristique particulière de cette fonction, celle-ci peut être considérée comme étant le précurseur de la conteneurisation.

Un conteneur enveloppe l'application d'un logiciel dans une boite invisible avec tout ce dont il a besoin pour s'exécuter. Cela comprend le système d'exploitation, le code de l'application, le runtime, les outils système et les librairies. Les conteneurs Docker sont construits à partir des images Docker. Ils sont légers, portables et permettent aux développeurs de créer, de déployer et d'exécuter efficacement des applications distribuées. En outre, ils permettent à une application d'être empaquetée et déplacée facilement, augmentant ainsi la simplicité d'une infrastructure.

- Docker

Docker est l'incarnation la plus moderne de la conteneurisation. Il s'agit d'un projet ouvert développé par la société Dotcloud (renommée par la suite Docker) sur la base LXC (Linux Containers, est une solution de virtualisation basée sur la capacité du noyau Linux à faire fonctionner des environnements isolés). Fondamentalement, Docker étend LXC avec le

Docker deamon. Ce dernier est une API (Application Programming Interface) installée entre le noyau du système d'exploitation et les VE, qui, ensemble, exécutent des processus isolés : processeur, mémoire, entrées/sorties (E/S), réseau, etc. Depuis la version 0.9 en 2014, Docker n'utilise plus LXC comme environnement d'exécution par défaut. Il l'a remplacé par sa propre bibliothèque Libcontainer (Docker, 2016b), implémentée en Go, un langage de programmation orientée service. Docker est un écosystème complexe mais intuitif pour le développement des conteneurs, il est riche en fonctionnalités, notamment son système d'image, ses registres et son interface de lignes de commandes. La figure suivante présente l'architecture de base Docker.

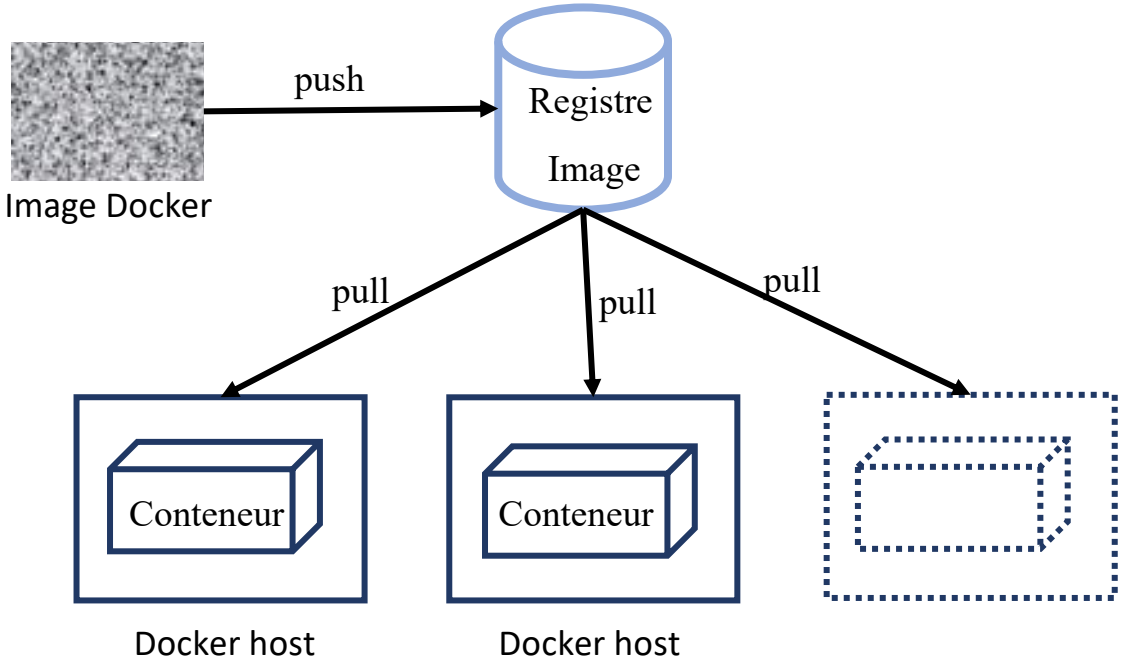


Figure 17 : Architecture de base de Docker

- Cluster

Le terme cluster (grappe en français) fait référence à une batterie d'ordinateurs, tous interconnectés, dans le but de partager des ressources informatiques pour augmenter une disponibilité d'un ou plusieurs services par exemple.

En d'autres termes, un cluster peut être :

- Soit un ensemble d'ordinateurs connectés en réseaux et capable de traiter un même travail. Par exemple le cumul de puissances des machines du cluster pour exécuter des algorithmes gloutons.

- Soit un ensemble d'ordinateurs connectés en réseaux dans le but de répartir des tâches sur chaque machine du cluster. Par exemple un cluster de tâches systèmes allouera à chaque machine du cluster un ensemble de processus à traiter suivant sa charge et sa puissance de calcul.

Par ailleurs, un cluster reste un système facilement administrable et modifiable. En effet, toutes les machines sont redondantes au niveau système et donc il suffira de designer un des nœuds comme "master". De plus, il suffira à un administrateur d'intégrer de nouvelles machines dans l'ensemble pour augmenter la puissance de son système.

4. Déploiement de Keycloak

Keycloak peut être installé et déployé en mode standalone (autonome) ou en mode domaine.

4. 1. Mode autonome

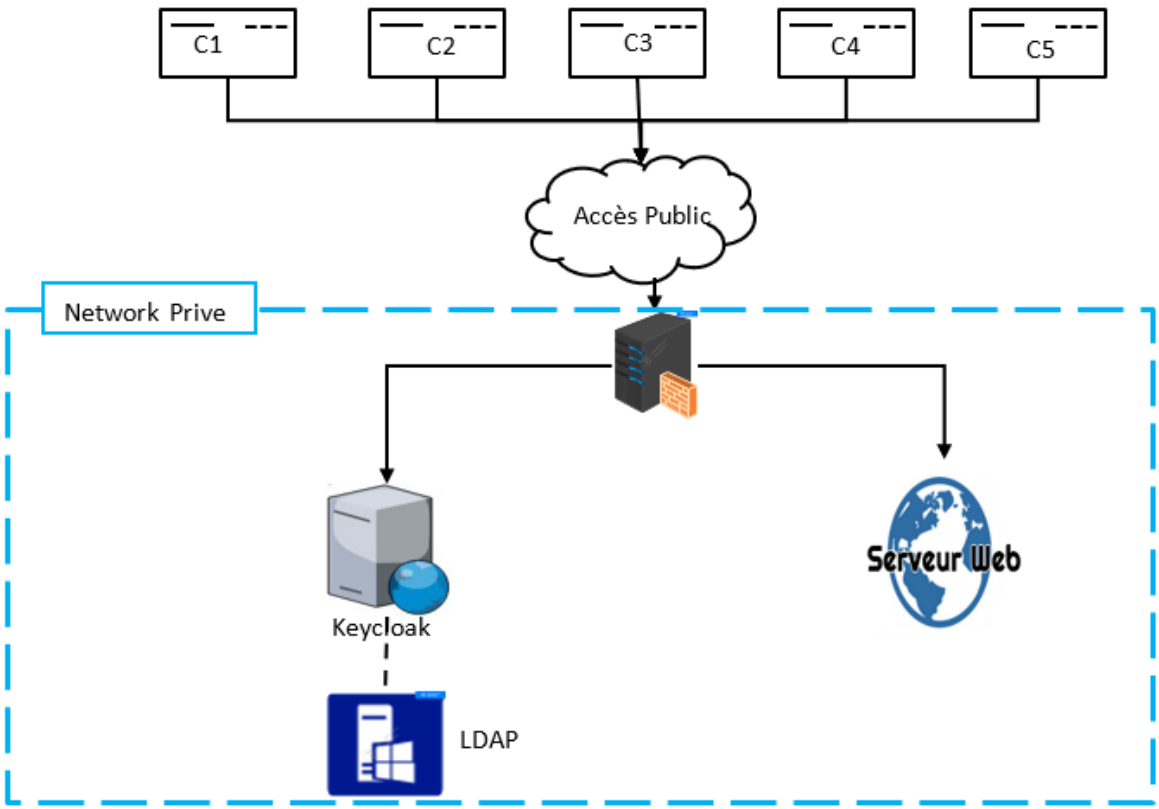


Figure 18 : Architecture mode autonome

Cette figure montre l'architecture de notre solution en local. Nous avons ici un ensemble de clients (C1, C2, C3, C4 et C5) qui peuvent accéder au serveur Keycloak à travers un proxy

reverse en frontal. Le serveur Web héberge l'ensemble des applications web déléguant leur sécurité à Keycloak. Le serveur Keycloak est synchronisé avec un serveur d'annuaire LDAP (AD) qui est la base de nos utilisateurs.

Ce mode de fonctionnement n'est utile que lorsque l'on souhaite exécuter une et une seule instance de serveur Keycloak. Il n'est pas utilisable pour les déploiements en cluster car tous les caches sont non distribués sauf les locaux uniquement. Il n'est pas recommandé d'utiliser ce mode en production car le système d'authentification aura un point de défaillance unique. Si le serveur d'authentification tombe en panne, les utilisateurs ne pourront plus accéder aux applications qu'il protège. Le déploiement de ce mode en local sert que de test et d'apprentissage de Keycloak. Nous avons installé Keycloak 15.0.2 sur une machine virtuelle Ubuntu 20.04.

L'utilisation d'une image docker pour installer Keycloak simplifie grandement son déploiement comparer à celle utilisée sans docker. Avec docker, l'installation se résume en une seule commande

docker run -p 8080:8080 -e KEYCLOAK_USER=user -e KEYCLOAK_PASSWORD=pwd jboss/keycloak :15.0.2

- Installation

Nous avons installé Keycloak pas à pas sans docker. Cette installation est nommée « baremetal », elle se base sur le serveur « Wildfly ».

Le serveur Ubuntu doit être à jour et openjdk-11-jdk doit y être installé, toutes les commandes sont exécutées en mode root :

apt-get update && apt-get dist-upgrade

apt-get install -y openjdk-11-jdk

Afin de ne pas lancer Keycloak en tant que root, créons un utilisateur Linux privilégié pour Keycloak puis installons la version 15.0.2 de Keycloak :

groupadd keycloak-user useradd -r -g keycloak-user -d /usr/lib/keycloak -s /sbin/nologin keycloak

cd /usr/lib

wget https://github.com/keycloak/keycloak/releases/download/15.0.2/keycloak-15.0.2.tar.gz

tar xzf keycloak-15.0.2.tar.gz

ln -s keycloak-15.0.2 keycloak

rm keycloak-15.0.2.tar.gz

chown -R keycloak : /usr/lib/keycloak-15.0.2

chmod o+x /usr/lib/keycloak/

- Configuration

mkdir /etc/keycloak

mkdir -p /var/run/keycloak

chown keycloak: /var/run/keycloak

cp /usr/lib/keycloak/docs/contrib/scripts/systemd/launch.sh /usr/lib/keycloak/bin/

cp /usr/lib/keycloak/docs/contrib/scripts/systemd/wildfly.conf /etc/keycloak/keycloak.conf

cp /usr/lib/keycloak/docs/contrib/scripts/systemd/wildfly.service /etc/systemd/system/keycloak.service

Editons les fichiers de configuration du serveur Keycloak avec la commande sed :

sed -i 's!wildfly!keycloak!g' /etc/systemd/system/keycloak.service

sed -i 's!/opt/keycloak!/usr/lib/keycloak!' /etc/systemd/system/keycloak.service

sed -i 's!/opt/wildfly!/usr/lib/keycloak!' /usr/lib/keycloak/bin/launch.sh

Puisque nous avons installé notre serveur derrière un reverse proxy, exécutons cette commande pour renseigner au serveur Keycloak l'URL du reverse proxy :

sed -i 's!\$3$!\$3 -Dkeycloak.frontendUrl=https://auth.keycloak.sn/auth!' /usr/lib/keycloak/bin/launch.sh

Autre possibilité est d'éditer le fichier …/tandalone.xml dans la balise « **spi** » et de renseigner l'URL du reverse proxy afin de mettre la variable **forceBackendUrlToFrontendUrl** sur **true**.

Le serveur Keycloak peut répondre sur une ou plusieurs interfaces. Pour associer une adresse IP à une interface du réseau, il est possible d'utiliser la commande :

standalone.sh -Djboss.bind.address = adresse IP

Ou de manière abrégée :

standalone.sh -b adresse IP ou encore éditer le fichier de configuration …/standalone.sh pour renseigner directement l'adresse IP dans la balise « **interfaces** ».

Au lieu de configurer notre serveur Keycloak en HTTPS, nous avons utilisé un reverse proxy en frontal sur notre environnement qui joue le rôle de mandataire entre le public (les utilisateurs externes) et notre serveur Keycloak.

Installons notre proxy reverse sur un serveur Nginx :

- Installation de Nginx

apt-get install -y nginx

- Configuration de Nginx

Nginx doit être configurer pour forwarder le trafic entrant vers notre serveur Keycloak. Ce proxy reverse permettra de sécuriser notre serveur, il est le point d'entrée unique de notre système d'authentification. Seule son adresse IP sera connue de l'extérieure. Il sera configuré pour autoriser seulement les trafics HTTPS afin que les jetons d'accès puissent être partagés en toute sécurité.

Désactivons le « vhost » préconfiguré par défaut dans Nginx:

unlink /etc/nginx/sites-enabled/default

Créons le fichier de configuration du reverse proxy dans le répertoire /etc/nginx/sites-available/

nano /etc/nginx/sites-available/keycloak.conf

Voici le contenu de ce fichier :

```
server {
    listen 80;
    server_name keycloak.domaine.sn;
    rewrite ^ https://keycloak.domaine.sn permanent;
}
server {
    listen 443 ssl;
    server_name keycloak.domaine.sn;
    ssl_certificate /etc/letsencrypt/live/keycloak.domaine.sn/fullchain.pem;
```

```
    ssl_certificate_key /etc/letsencrypt/live/keycloak.domaine.sn/privkey.pem;
    ssl_session_cache builtin:1000 shared:SSL:10m;
    ssl_protocols TLSv1 TLSv1.1 TLSv1.2 TLSv1.3;
    ssl_ciphers HIGH:!aNULL:!eNULL:!EXPORT:!CAMELLIA:!DES:!MD5:!PSK:!RC4;
    ssl_prefer_server_ciphers on;
    location / {
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_pass http://IP_SERVEUR_KEYCLOAK:8080;
    }
}
```

keycloak.domaine.sn doit correspond à l'url du serveur Keycloak et la directive « proxy_pass » doit mener au serveur Keycloak.

Créons une ligne symbolique pour activer notre virtuel host :

```
# ln -s /etc/nginx/sites-available/keycloak.conf /etc/nginx/sites-enabled/keycloak.conf
```

Pour les certificats SSL, utilisons les certificats gratuits de Debian.

En production, il faudra utiliser des certifications reconnus par les navigateurs qui sont fournies par une CA (Autorité de Certification) SSL/TLS.

Installons **Certbot** pour Nginx :

```
# apt install certbot python3-certbot-nginx
```

Demander un certificat SSL pour notre reverse proxy :

```
# certbot --nginx -d www.domaine.sn
```

--nginx spécifie que nous utilisons un serveur web nginx et www.domaine.sn doit correspondre au domaine DNS.

Les informations comme l'adresse email de l'administrateur, le nom de notre organisation, notre pays, le nom de notre région et le nom DNS de notre serveur seront demandés.

A la fin de la génération des certificats, il nous est demandé si nous souhaitions rediriger automatiquement le trafic HTTP vers HTTPS.

« Certbot » s'est occupé de configurer automatiquement le bloc correspondant sur le fichier de configuration du reverse proxy.

Les certificats « Let's Encrypt » ont une durée de validité de 90 jours et peuvent être renouvelés automatiquement.

Vérification de la configuration de nginx :

nginx -t

Redémarrons nginx pour valider les configurations :

systemctl restart nginx

A partir de là, nous avons quasiment un Keycloak fonctionnel. Le serveur vient avec un domaine par défaut nommé « master » qui gère la configuration de base du serveur.

Avant de démarrer notre serveur, Créons l'utilisateur admin avec la commande :

/usr/lib/keycloak/bin/add-user-keycloak.sh -r master -u administrateur

Un mot de passe pour le compte administrateur sera demandé.

Après avoir renseigné le mot de passe de l'administrateur, nous démarrons notre serveur Keycloak.

systemctl enable keycloak.service

systemctl start keycloak.service

La console d'administration de Keycloak est accessible via l'adresse IP du serveur Keycloak sur le port 8080.

4. 2. Mode cluster de domaine

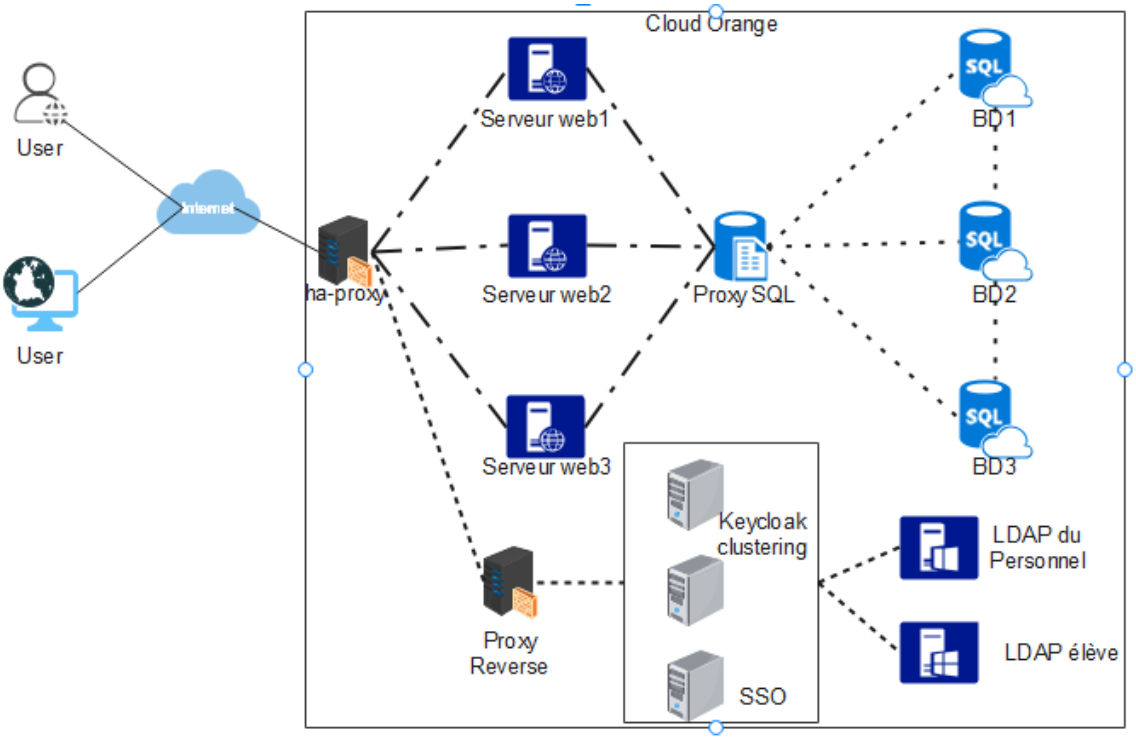


Figure 19 : Architecture finale après déploiement de la solution

Cette figure montre l'architecture du déploiement de notre solution en mode cluster de domaine. Nous avons ici un ensemble de clients qui peuvent accéder aux serveurs d'applications hébergeant les applications qui délèguent leur sécurité au serveur Keycloak (un cluster composé de deux instances de Keycloak) à travers un proxy reverse en frontal. Les instances de Keycloak sont synchronisées avec deux serveurs annuaires LDAP.

Le mode domaine est un moyen d'administrer et de publier de manière centralisée la configuration des serveurs de Keycloak en déploiement.

L'exécution d'un cluster de serveur Keycloak en mode standard peut rapidement devenir agaçante à mesure que la taille du cluster augmente car chaque serveur gère sa propre base de configuration. Chaque fois qu'une modification est apportée à la configuration, il faut effectuer cette modification sur chaque nœud du cluster.
Le mode domaine résout ce problème en fournissant un emplacement central (une base de données externes) pour stocker et publier la configuration. Cela peut être complexe à mettre en place, mais cela vaut la peine au final car cette fonctionnalité est intégrée au serveur d'application « Wildfly » sur lequel tourne Keycloak.
Voici quelques-uns des concepts de base de l'exécution de Keycloak en mode cluster domaine.

- Contrôleur de domaine

Le contrôleur de domaine est un processus responsable du stockage, de la gestion et de la publication des configurations générales de chaque nœud du cluster. Ce processus est le point central à partir duquel les nœuds du cluster obtiennent leurs configurations.

- Contrôleur hôte

Ce contrôleur est responsable de la gestion des instances du serveur sur une machine spécifique. Il est configuré pour exécuter une ou plusieurs instances de serveur. Le contrôleur de domaine peut également interagir avec les contrôleurs hôtes sur chaque serveur pour gérer le cluster. Pour réduire le nombre de processus en cours d'exécution, le contrôleur de domaine agit également en tant que contrôleur hôte sur le serveur sur lequel il s'exécute.

- Profil de domaine

Un profil de domaine est un ensemble nommé de configuration qui peut être utilisé par un serveur pour démarrer. Le contrôleur de domaine peut définir plusieurs profils de domaine qui sont consommés par différents serveurs.

- Groupe de serveurs

Un groupe de serveurs est un ensemble de serveurs gérés et configurés comme un tout. Il est possible d'attribuer un profil de domaine à un groupe de serveurs et chaque service de ce groupe utilisera ce profil de domaine comme base de configuration.

Le contrôleur de domaine est démarré sur le nœud maître. La configuration du cluster réside dans ce contrôleur. Ensuite, le contrôleur hôte est démarré sur chaque nœud du cluster. Chaque configuration de déploiement du contrôleur hôte spécifie le nombre d'instances de serveur Keycloak qui seront démarrées sur cette machine. Lorsque le contrôleur hôte démarre, il démarre toutes les instances de serveur Keycloak. Ces dernières extraient leur configuration sur le contrôleur de domaine.

4. 2. 1. Installation et configuration

Il est possible d'installer et configurer indépendamment toutes les instances de serveur Keycloak dans un domaine. De plus, en mode cluster une base de données externe doit être partagée entre toutes les instances de Keycloak. Cette base contiendra l'ensemble des configurations de ces instances. Elle contiendra aussi une copie des utilisateurs issue de l'annuaire LDAP synchronisé avec tous les nœuds Keycloak du cluster.

Ces derniers doivent être installés sur un container docker de même que la base de données. L'installation de Keycloak se résume à cette commande :

#docker run -p 8080:8080 -e KEYCLOAK_USER=votre-admin -e KEYCLOAK_PASSWORD =password jboss/keycloak:15.0.2

Cette commande installe la version 15.0.2 de Keycloak puis crée l'utilisateur admin.

Elle pouvait être scindée en deux d'abord installer le serveur Keycloak sur le conteneur docker puis créer le compte administrateur comme suit :

docker run -p 8080:8080 jboss/keycloak:15.0.2

#docker run -e KEYCLOAK_USER=<username> -e KEYCLOAK_PASSWORD=<password > jboss/keycloak

Le numéro de port 8080 est le port par défaut utilisé par Keycloak, il peut être modifié.

- Installation de la base de données

Cette base sera partagée par l'ensemble des nœuds du cluster. Elle sera aussi installée sur un container docker. Keycloak est compatible avec plusieurs bases de données comme MySQL, P ostgreSQL, MariaDB, Oracle ou Microsoft SQL Server.

Avant d'installer la base de données, il faudra d'abord créer un réseau pour faciliter le partage de la base sur l'ensemble des instances du cluster.

docker network create keycloak-network

Puis créer une instance de la base de données MySQL à l'aide de l'image docker MySQL.

docker run --name mysql -d --net keycloak-network -e MYSQL_DATABASE=keycloak -e MYSQL_USER=keycloak -e MYSQL_PASSWORD=password -e MYSQL_ROOT_PASSWORD=root_password mysql

Nous avons fait ici le choix d'installer MySQL. A la place n'importe quelle base de données parmi celles citées précédemment peut être installée.

Et enfin démarrer chaque instance de Keycloak pour quelle se connecte à la base de données nouvellement créée.

docker run --name keycloak --net keycloak-network jboss/keycloak

- Configuration du domaine

Le mode autonome utilise le fichier de configuration « …/tandalone.xml » pour configurer ces éléments, alors que le mode domaine utilise le fichier de configuration « .../domaine.xml ». C'est dans ce fichier (domaine.xml) que les éléments tels que les connexions réseau, les caches, les connexions à la base de données, le profil de domaine et le groupe de serveurs Keycloak doivent être configurés. Toute modification apportée à ce fichier pendant l'exécution du contrôleur de domaine ne prendra pas effet et sera écrasée par le serveur. Keycloak est livré avec deux fichiers de configuration du contrôleur hôte (host-master.xml et host-slave.xml) qui sont dans le répertoire .../domaine/configuration/. « host.master.xml » est configuré pour démarrer un contrôleur de domaine, un équilibreur de charge et une instance de serveur Keycloak.

En production si vous utilisez un équilibreur de charge en frontal, modifier le fichier host-master.xml et commenter ou supprimer l'entrée « load-balancer » pour désactiver la répartition de charge.

Chaque instance de serveur Keycloak définie dans ces fichiers hôtes crée un répertoire de travail sous .../domaine/servers/{nom de l'instance du serveur}.

Le contrôleur hôte des autres instances de Keycloak nommé slave doit être configuré pour qu'il puisse communiquer avec le contrôleur de domaine en toute sécurité. A défaut de cela, les autres instances ne pourront pas obtenir la configuration centralisée du contrôleur de domaine. Pour configurer une connexion sécurisée entre les nœuds du cluster, il faudra créer une utilisation admin du nœud maître et un secret qui sera partagé entre eux. Pour cela, exécutons ce script :

…/bin/add-user.sh

L'option a) Management User a été choisi. Après avoir créé l'utilisateur admin et son mot de passe, deux questions nous ont été posées auxquelles nous avons répondu par « yes ». Le secret qui sera partagé, est généré à la fin.

Maintenant, copions la valeur de ce secret dans le fichier …/host-slave.xml au niveau de la balise <secret value=«valeur-secret »> contenue dans la balise <server-identities>.

Les serveurs Keycloak peuvent répondre sur une ou plusieurs interfaces. Pour associer une adresse IP à l'une de ses interfaces nous devons taper la commande suivante :

domain.sh -Djboss.bind.address = adresse IP

Ou de manière abrégée :

domain.sh -b adresse IP ou encore éditer le fichier de configuration standalone.sh pour renseigner directement l'adresse IP dans la balise « **interfaces** ».

Pour démarrer le service Keycloak, nous devons démarrer tous les containers docker. Le nœud maitre doit être démarré en premier. Si les instances de Keycloak n'ont pas été installées sans docker, voici les scriptes contenus dans le répertoire /bin de la distribution du Keycloak à exécuter sur chaque nœud du cluster pour les démarrer.

Sous Linux/Unix :

…/bin/domaine.sh --host-config=host-master.xml

Sous Windows :

…\bin\domaine.bat --host-config=host-slave.xml

Il est possible de sécuriser le cluster de Keycloak par un reverse proxy en frontal comme nous l'avons fait sur le mode autonome.

Cette mode de configuration est fastidieuse surtout dans un déploiement de grande envergure.

Pour déployer notre service Keycloak, nous avons opté d'utiliser docker-compose pour orchestrer l'exécution de tous les nœuds du cluster et leurs interactions. Pour un déploiement de grande envergure, Kubernetes doit être utilisé pour orchestrer le cluster de Keycloak.

Pour atteindre nos objectifs, nous avons créés un cluster avec trois instances de Keycloak tournant sur une même machine. Ces deux nœuds partagent une base de données commune, PostgreSQL.

Nous avons regroupé ces instances et la base de données dans un fichier docker_composite.yml :

version: '3'

```
volumes:
  postgres_data:
    driver: local

services:
  postgres:
    container_name: kc_db
    image: postgres:11
    volumes:
      - postgres_data:/var/lib/postgresql/data
    environment:
      POSTGRES_DB: name-db
      POSTGRES_USER: user-db
      POSTGRES_PASSWORD: password-db
    ports:
      - 5433:5432

  kc1:
    container_name: kc1
    image: jboss/keycloak:16.1.1
    volume: /simenthemes:/opt/jboss/keycloak/themes/simenthemes
    environment:
      DB_VENDOR: POSTGRES
      DB_ADDR: postgres
      DB_DATABASE: name-db
      DB_USER: user-db
      DB_PASSWORD: password-db
      KEYCLOAK_USER: admin-keycloak
      KEYCLOAK_PASSWORD: password-keycloak
    ports:
      - 8081:8080
    depends_on:
      - postgres
```

```
kc2:
  container_name: kc2
  image: jboss/keycloak:16.1.1
  volume: /simenthemes:/opt/jboss/keycloak/themes/simenthemes
  environment:
    DB_VENDOR: POSTGRES
    DB_ADDR: postgres
    DB_DATABASE: name-db
    DB_USER: user-db
    DB_PASSWORD: password-db
    KEYCLOAK_USER: admin-keycloak
    KEYCLOAK_PASSWORD: password-keycloak
    PROXY_ADDRESS_FORWARDING: "true"
  ports:
    - 8082:8080
  depends_on:
    - postgres

Kc3:
  container_name: kc3
  image: jboss/keycloak:16.1.1
  volume: /simenthemes:/opt/jboss/keycloak/themes/simenthemes
  environment:
    DB_VENDOR: POSTGRES
    DB_ADDR: postgres
    DB_DATABASE: name-db
    DB_USER: user-db
    DB_PASSWORD: password-db
    KEYCLOAK_USER: admin-keycloak
    KEYCLOAK_PASSWORD: password-keycloak
    PROXY_ADDRESS_FORWARDING: "true"
  ports:
    - 8083:8080
```

```
  depends_on:
    - postgres
```

Au lieu de configurer notre serveur Keycloak en HTTPS, nous avons utilisé un reverse proxy en frontal sur notre environnement qui joue le rôle de mandataire entre le public (les utilisateurs externes) et notre serveur Keycloak.

Installons notre proxy reverse sur un serveur Nginx :

- Installation de Nginx

apt-get install -y nginx

- Configuration de Nginx

Nginx doit être configuré pour forwarder le trafic entrant vers notre serveur Keycloak. Ce proxy reverse permettra de sécuriser notre serveur, il est le point d'entrée unique de notre système d'authentification. Seule son adresse IP sera connue de l'extérieur. Le serveur Nginx sera configuré pour autoriser seulement les trafic HTTPS afin que les jetons d'accès puissent être partagés en toute sécurité.

Désactivons le « vhost » préconfiguré par défaut dans Nginx :

unlink /etc/nginx/sites-enabled/default

Créons le fichier de configuration du reverse proxy dans le répertoire /etc/nginx/sites-available/

nano /etc/nginx/sites-available/keycloak.conf

Et le fichier de configuration du reverse proxy :

```
upstream backend {
   server 127.0.0.1:8081 fail_timeout=5s;
   server 127.0.0.2:8082 fail_timeout=5s;
   server 127.0.0.3:8083 fail_timeout=5s;
}
server {
   listen 80;
   server_name serverSSO.name.sn;
```

```
    if ($scheme != "https") {
        return 301 https://$host$request_uri;
    }
}
server {
    listen 443 ssl; # managed by Certbot
    server_name serverSSO.name.sn;
    ssl_certificate /etc/letsencrypt/live/serverSSO.name.sn/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/serverSSO.name.sn/privkey.pem;
    include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot

    location / {
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Host $host;
        proxy_set_header X-Forwarded-Server $host;
        proxy_set_header X-Forwarded-Port $server_port;
        proxy_set_header X-Forwarded-Proto $scheme;

        proxy_pass http://backend;
        proxy_connect_timeout 5s;
        proxy_buffer_size 128k;
        proxy_buffers 4 256k;
        proxy_busy_buffers_size 256k;
    }
}
```

Les instances de Keycloak et la base de données sont installées avec la commande :

docker-compose -f docker_composite.yml up

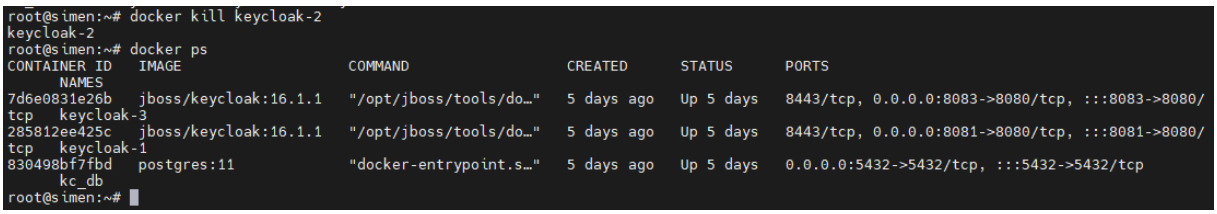
Avec la commande # docker ps, nous pouvons voir que notre cluster a été créé avec les trois instances de Keycloak et la base de données sont en train de tourner.



Capture 1 : Etat des conteneurs (Nginx, instances Keycloak) Docker

L'arrêt d'une ou deux de ces instances n'a pas d'incident sur la disponibilité du service SSO.

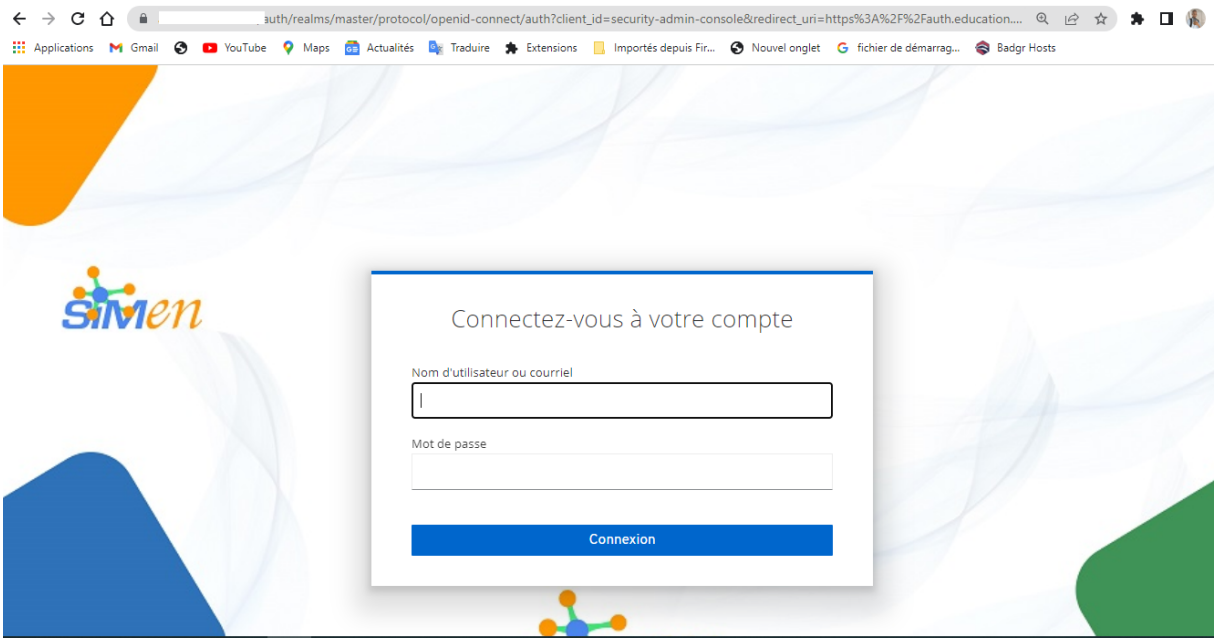
docker kill keycloak-2



Capture 2 : Etat de Keycloak après l'arrêt d'une de ses instances

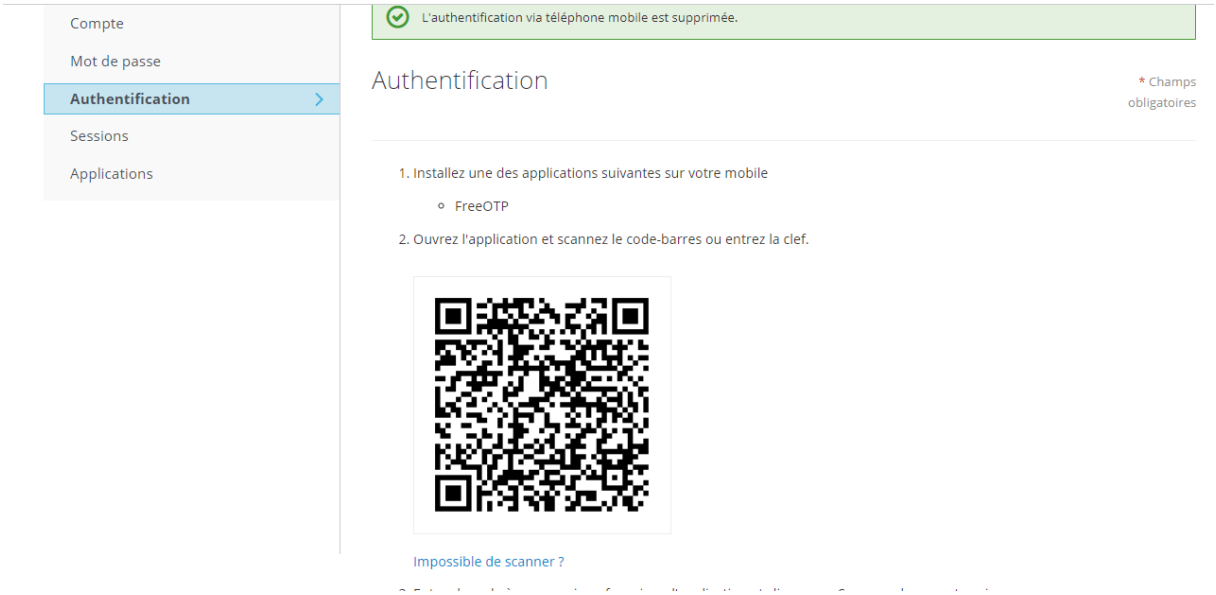
A l'arrêt du nœud 2, les nœuds 1 et 3 continuent de tourner.

Par ailleurs nous avons créé un thème personnalisé pour customiser notre serveur Keycloak. Ce thème, que nous avons nommé « simenthemes » hérite du thème de base « Keycloak ». Pour faire simple nous avons copié le répertoire (…/keycloak/themes/keycloak) du thème par défaut que nous avons modifié. Les feuilles de style CSS du login (la page de connexion des utilisateurs), admin (la page de connexion de l'administrateur), welcome (la page d'accueil de la console d'administration), account (page d'inscription et de réinitialisation de mot de passe) ont été modifiées.



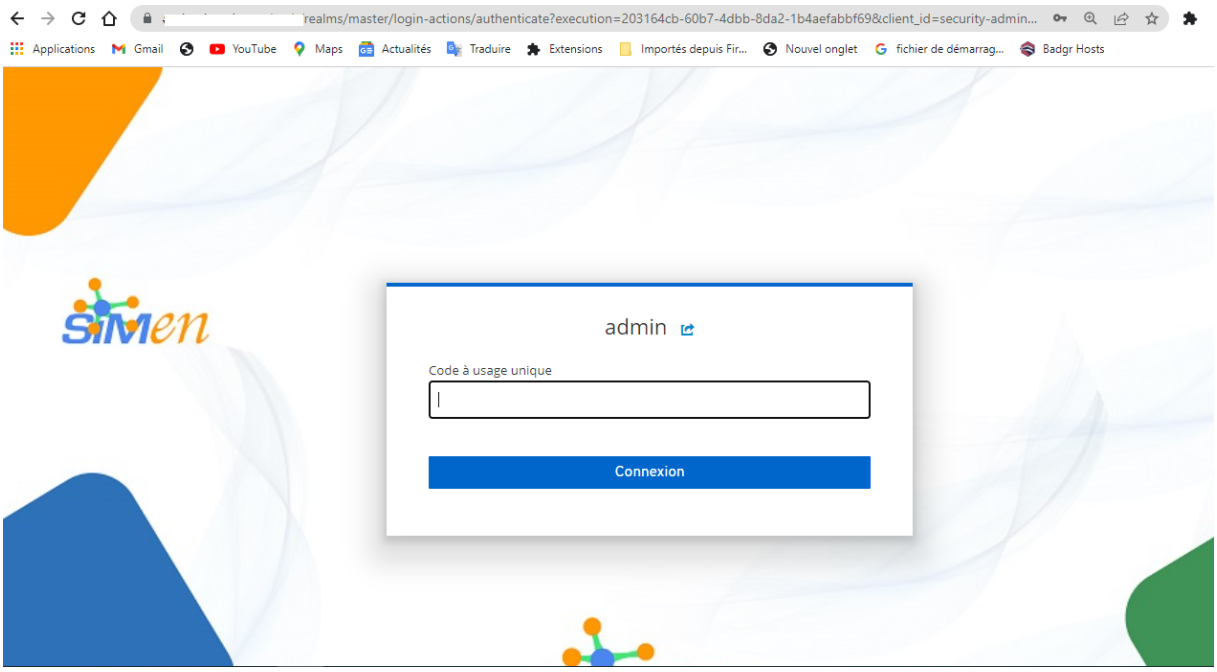
Capture 3 : Accès à la console d'administration

Nous avons configuré l'authentification double facteur pour le compte de l'administrateur avec OTP (One Time Password). OTP est un module d'authentification second facteur de Keycloak. Il est facilement configurable : il suffit juste de synchroniser le compte de l'administrateur avec l'application FreeOPT ou Google Authentificator. Cette synchronisation se fait par scan du code QR du module OTP de Keycloak.



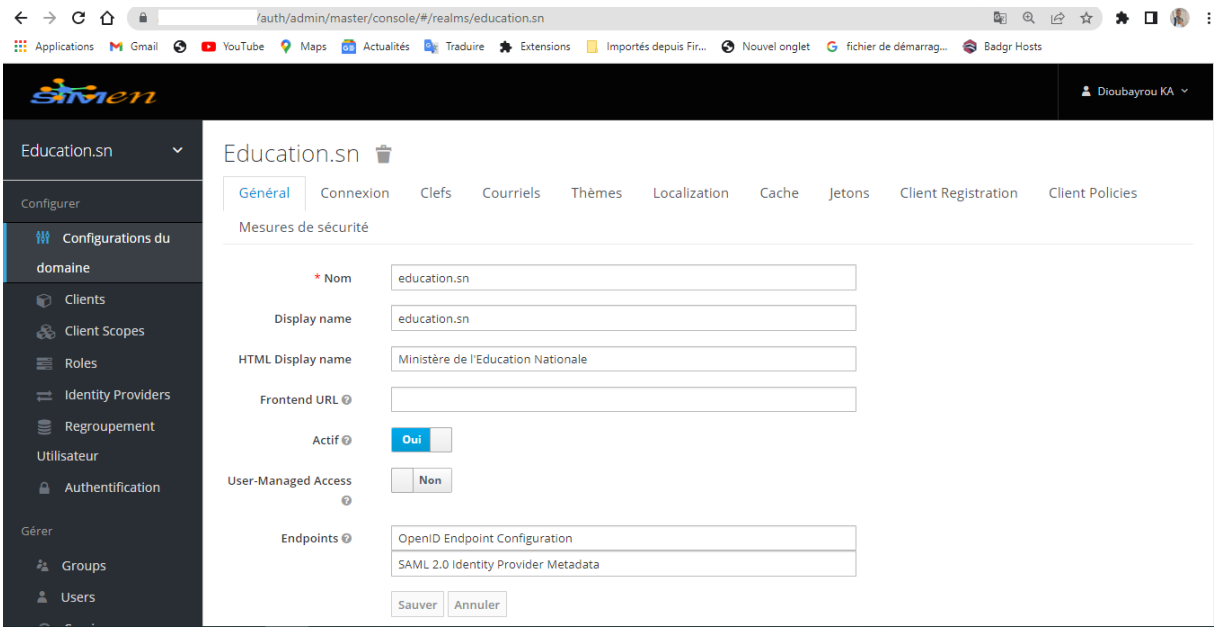
Capture 4 : Configuration authentification 2 Facteur pour l'administrateur

Une fois que l'administrateur renseigne son login et mot de passe, un code à usage unique lui est demandé. Ce code sera visible sur l'application FreeOPT.



Capture 5 : Code à usage unique pour l'administrateur

Nous allons créer un domaine nommé education.sn sur notre serveur. Ce domaine va contenir l'ensemble des applications (clients) à protéger. Il est possible d'importer directement un fichier Json contenant un domaine déjà configuré avec un ensemble de clients.



Capture 6 : Création de domaine Keycloak

Présentation de la console d'administration de Keycloak.

Sur la section configuration du domaine, nous avons :

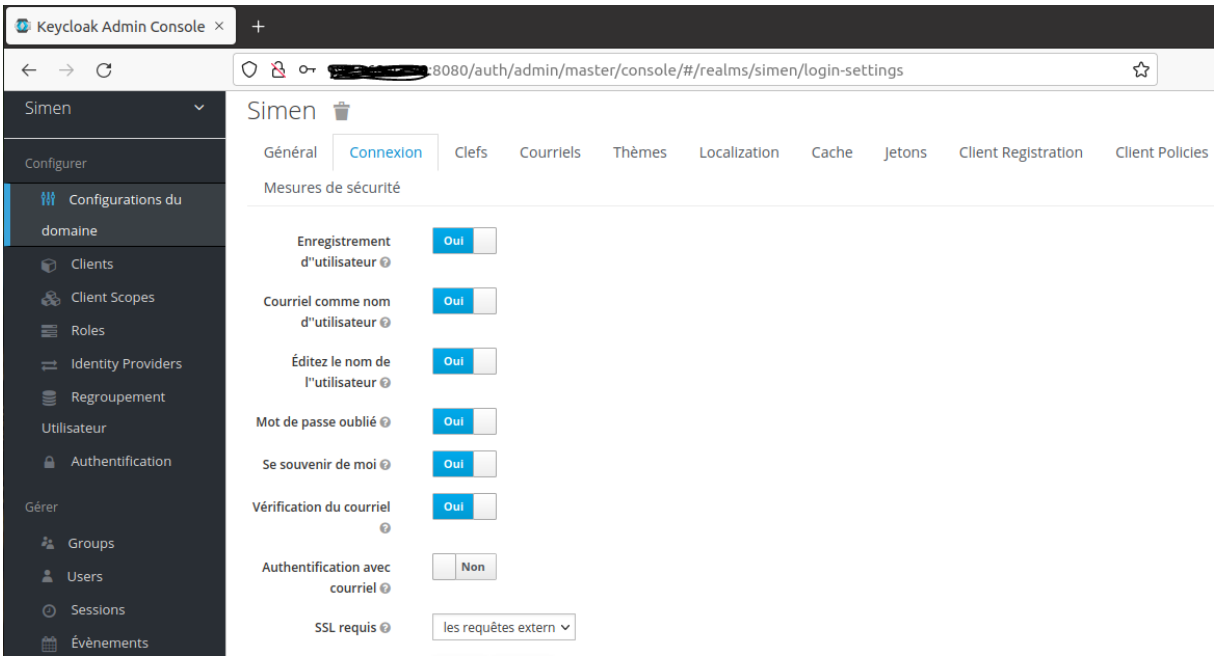
- L’onglet Général

Sur cet onglet nous pouvons, voir le nom du domaine qui a été créé, spécifier l’url du reverse proxy en frontal au besoin, activer ou désactiver un domaine, permettre aux utilisateurs de gérer leurs ressources et autorisations à l’aide de la console de gestion de compte et enfin voir les points de terminaux Openid Connect et SAML 2.

- L’onglet Connexion

Il permet de manager l’ensemble des actions lié aux comptes des utilisateurs et d’activer le module de sécurité qui dira à Keycloak d’accepter les requêtes externes que si elles sont en HTTPS. Nous pouvons activer l’enregistrement des utilisateurs pour que ces derniers puissent s’inscrire sur le service Keycloak. De plus, il est possible de permettre aux utilisateurs d’éditer leur nom d’utilisateur et de réinitialiser leur mot de passe.

Par ailleurs, cet onglet offre la possibilité d’activer l’authentification avec l’adresse électronique et de forcer la vérification de chaque nouvelle authentification à travers le courrier de l’utilisateur.



Capture 7 : Console d’administration de Keycloak

- L’onglet Clefs

Cet onglet permet de sélectionner les algorithmes de chiffrements à utiliser pour la signature des jetons d'accès.

- L'onglet Courriels

Elle permet de configurer un relais SMTP pour l'envoie de mails lors de la création de compte, de la réinitialisation des mots de passe et lors des confirmations de l'identité d'un compte.

- L'onglet Thèmes

Cet onglet permet de sélectionner et de définir le thème personnalisé comme thème par défaut de Keycloak.

- L'onglet localisation

Il permet de trouver rapidement un domaine ou une information spécifique sur un ou plusieurs domaines enregistrés sur notre serveur.

- L'ongle Cache

Cet onglet permet de vider le cache d'un domaine, le cache des comptes utilisateurs et le cache des clés de chiffrements.

- L'onglet Jetons

Il permet de manager les informations sur les tokens d'accès, de définir l'algorithme de signature des tokens, d'autoriser la révocation des « Refresh Token », de définir la durée maximale des sessions SSO, de limiter le nombre de session, etc.

- L'onglet de Sécurité

Cet onglet permet d'activer la détection des attaques par force brute, de limiter le nombre de tentative de connexion (par exemple bloquer le compte d'utilisateur après 5 mauvaises tentatives de connexion), de définir un délai entre deux connexions, de caractériser des entêtes spéciaux pour se protéger des injections de code JavaScript malicieux, de se protéger contre les attaques XSS (Cross-Site Scripting), etc.

- L'onglet Client Policies

Il permet de créer des règles spécifiques pour la sécurité des applications protégées par Keycloak.

- L’onglet Client Registration

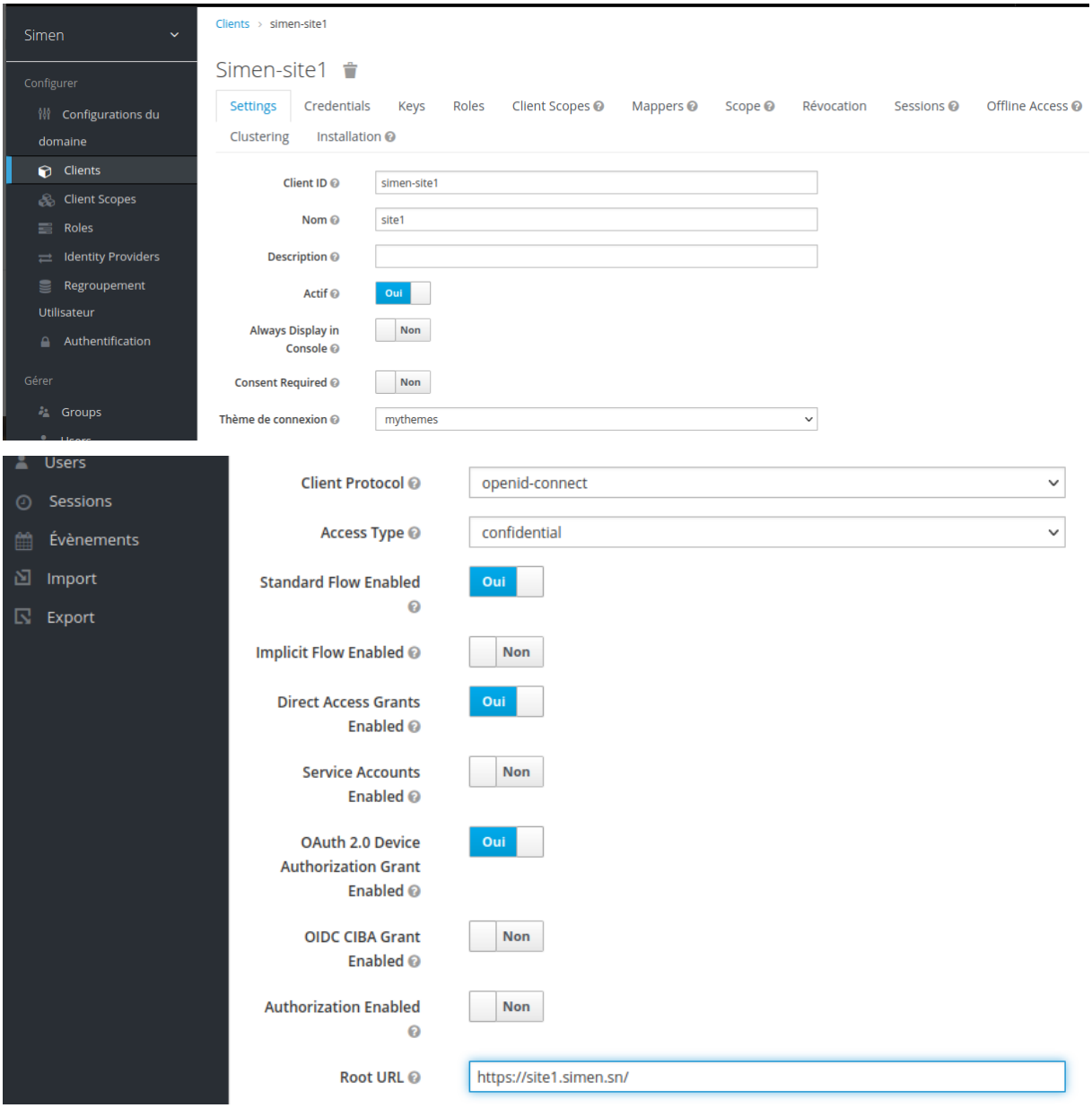
Cet onglet permet de créer des règles sur l’enregistrement des clients (applications, services web et API) dans Keycloak. Il permet aussi de restreindre les métadonnées publiées par Openid Connect et SAML 2 pour l’enregistrement dynamique des clients.

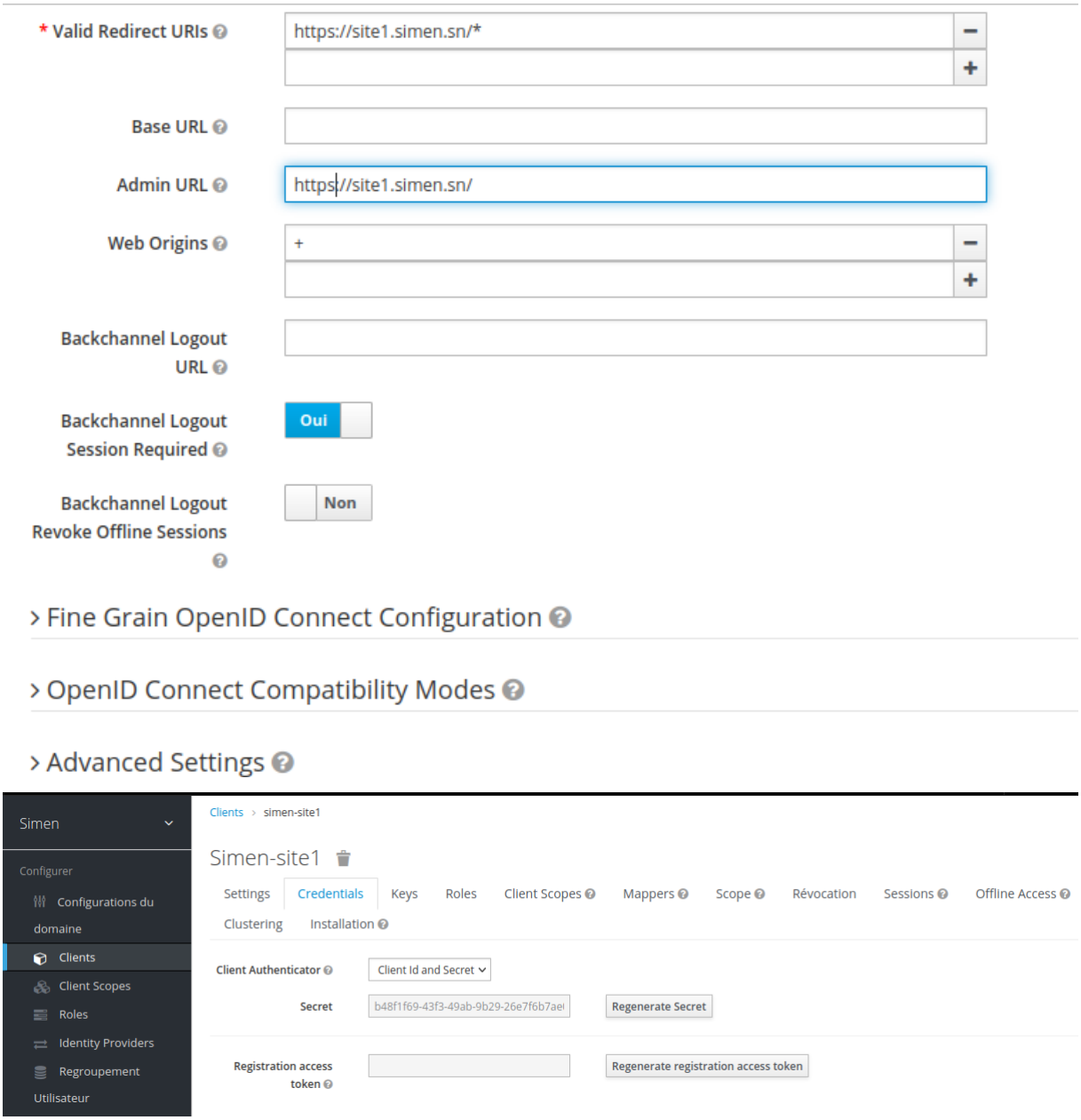
- Section Clients et Clients Scopes

Cette section permet de gérer les clients protégés par Keycloak. Lors de la création d’un nouveau client, son ID, son nom, son url de déconnexion (le protocole d’authentification Openid Connect ou SAML), le type d’accès (qui doit être « confidential » si le client est un service web ou une API, « public » si le client est une simple application web, « bearer-only » si le client est à la fois un service et une application web), doit être renseigné.

L’onglet Credentials permet de spécifier la manière de sécuriser un client par ses identifiants sur le serveur Keycloak (client id et Secret qui est un secret partagé entre le client et le serveur Keycloak) ou par un JWT signé ou par un certificat X509 ou encore avec un JWT signé avec le secret du client.

Voici une capture d’enregistrement d’un nouveau client :


Simen
Configurer
Configurations du domaine
Clients
Client Scopes
Roles
Identity Providers
Regroupement Utilisateur
Authentification
Gérer
Groups
Clients › simen-site1
Simen-site1
Settings
Credentials
Keys
Roles
Client Scopes
Mappers
Scope
Révocation
Sessions
Offline Access
Clustering
Installation
Client ID
simen-site1
Nom
site1
Description
Actif
Oui
Always Display in Console
Non
Consent Required
Non
Thème de connexion
mythemes
Users
Sessions
Évènements
Import
Export
Client Protocol
openid-connect
Access Type
confidential
Standard Flow Enabled
Oui
Implicit Flow Enabled
Non
Direct Access Grants Enabled
Oui
Service Accounts Enabled
Non
OAuth 2.0 Device Authorization Grant Enabled
Oui
OIDC CIBA Grant Enabled
Non
Authorization Enabled
Non
Root URL
https://site1.simen.sn/



Capture 8 : Enregistrement d'un client

- La section rôle

Elle permet de créer des rôles pour un ou groupe d'utilisateurs afin de restreindre les accès aux clients protégés par Keycloak.

- La section Identity Providers

Cette section permet la configuration des points d'authentification afin que les utilisateurs puissent s'authentifier auprès de Keycloak avec d'autres fournisseurs d'identités externes tel

que (Google, Microsoft, Github, Facebook, ...). Elle permet aussi de configurer l'authentification inter domaine.

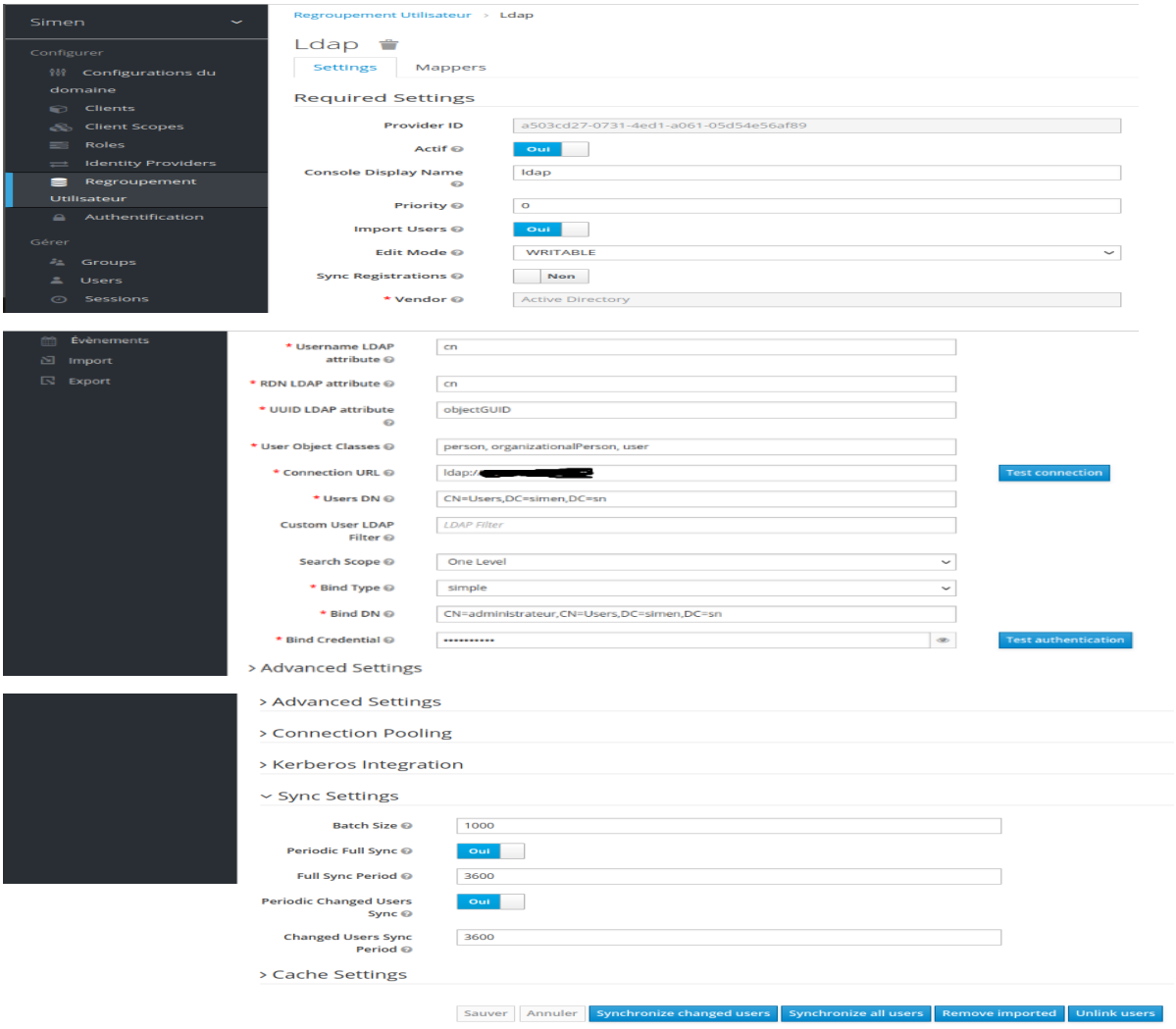
- La section Regroupement utilisateur

Elle permet de synchroniser Keycloak avec un fournisseur d'identités par exemple un annuaire LDAP (OpenLDAP ou Active Directory). Il est possible d'y ajouter plusieurs annuaires LDAP en les affectant de priorités.

L'onglet Mappers permet de spécifier les attributs des utilisateurs à importer.

Nous avons lié notre serveur Keycloak à un Active Directory en spécifiant l'adresse IP et le port du serveur AD. L'Edit Mode est défini en WRITABLE pour permettre la modification des informations sur les utilisateurs depuis les comptes utilisateurs. Active directory ne permet pas la modification de mot de passe en clair ; donc le port 636 doit être utilisé pour que les échanges entre Keycloak le serveur LDAP soit chiffré. La durée de rafraîchissement de la synchronisation est fixée à 1 H pour permettre plus de cohérence sur les informations des utilisateurs.

La capture suivante montre la configuration de la synchronisation entre notre serveur Keycloak et notre annuaire :



Capture 9 : Synchronisation avec AD

- La section Authentification

Cette section permet de créer, d'éditer, de forcer et de supprimer les paramètres d'authentification. Elle facilite aussi l'intégration de Kerberos sur Keycloak. Sur cette section, l'administrateur peut exiger des contraintes sur les mots de passe des utilisateurs. Par exemple, tout mot de passe doit contenir au moins une lettre majuscule, un ou des caractères spéciaux, une minuscule, un ou des chiffres et sa taille doit être au minimum de 8 caractères. Après avoir développé un module d'authentification double facteur par exemple, c'est sur cette section qu'il doit être appliqué.

- Section Gérer

Elle permet de gérer les utilisateurs et groupe d'utilisateurs. Il est possible de créer des utilisateurs directement sur Keycloak sans qu'ils soient dans l'annuaire ou qu'on leur attribue des rôles spécifiques.

Cette section permet également de voir l'historique des accès sur les clients Keycloak. Cette historique affiche l'adresse IP de l'utilisateur, son login, l'heure à laquelle il s'est logué, la durée de la session, etc. Elle permet aussi de déconnecter la session d'un utilisateur ou l'ensemble de ses sessions.

L'onglet évènement …..0.permet de visualiser l'ensemble des actions effectuées par l'administrateur.

L'onglet import permet d'importer un domaine déjà configuré avec l'ensembles des clients et les politiques d'accès.

L'onglet export facilite l'exportation du domaine sur un fichier Json.

Ces deux dernières fonctions facilitent la sauvegarde des configurations mais aussi la configuration des autres nœuds Keycloak qui sont sur des machines distinctes.

Pour que nos applications délèguent leur sécurité à notre serveur Keycloak afin que les utilisateurs bénéficient des sessions SSO, nous avons cloné le package de **Stavenmaguire** sur Github et nous l'avons adapté à notre besoin. Ce package joue le rôle de l'adaptateur entre les clients (applications ou services web) avec le serveur Keycloak. Toutes les applications relevant de notre domaine étant développé en PHP, seul cet adaptateur implémentant le protocole OAuth 2 suffit aux applications pour qu'elles interrogent le serveur SSO.

La large communauté de Keycloak fait qu'il peut protéger n'importe quelle application web. Il existe des adapteurs pour chaque langage web.

L'installation de l'adaptateur Stevenmaguire :

composer require stevenmaguire/oauth2-keycloak

A la fin de l'installation, le package doit être inclus dans le code des applications à protéger.

Voici un bout de code que nous avons écrit pour sécuriser une application PHP :

< ? php

require 'vendor/autoload.php';

```
session_start();
use Stevenmaguire;
$provider = new Stevenmaguire\OAuth2\Client\Provider\Keycloak([
  'authServerUrl'        => 'https://url_Keycloak:8080/auth',
  'realm'                => 'simen',
  'clientId'             => 'simen-site3',
  'clientSecret'         => 'e168ced9-dd11-483d-b700-e3c40272f7fd',
  'redirectUri'          => 'https://site1.simen.sn/',
  'encryptionAlgorithm'  =>  NULL,
  'encryptionKeyPath'    =>  NULL,
  'encryptionKey'        =>  NULL
]);
if (!isset($_GET['code'])) {
  $authUrl = $provider->getAuthorizationUrl();
  $_SESSION['oauth2state'] = $provider->getState();
  header('Location: '.$authUrl);
  exit;
} elseif (empty($_GET['state']) || ($_GET['state'] !== $_SESSION['oauth2state'])) {
  unset($_SESSION['oauth2state']);
  exit('Invalid state, make sure HTTP sessions are enabled.');
} else {
  try {
    $token = $provider->getAccessToken('authorization_code', ['code' => $_GET['code']);
  } catch (Exception $e) {
```

```
        header('Location: https://site1.simen.sn/'); exit;  //ligne ajoutée
        exit('Failed to get access token: '.$e->getMessage());
    }
try {
        $user = $provider->getResourceOwner($token);
        printf('Hello %s : !', $user->getName());
    } catch (Exception $e) {
        exit('Failed to get resource owner: '.$e->getMessage());
    }
   echo $token->getToken();
}
?>
```

Ce code permet à une application (https://site1.simen.sn par exemple) de déléguer sa sécurité au serveur Keycloak. Lorsqu'un utilisateur accède à cette application, on vérifie s'il a un code d'accès sinon nous lui demandons de s'authentifier sur Keycloak afin qu'il obtienne un token d'accès aux applications, ou bien nous vérifions son état par rapport à l'état de sa session précédente afin d'atténuer les attaques CRSF (Cross-Site Request Forgery). Si tout est conforme, l'utilisateur bénéficie d'une session SSO sur l'ensemble des applications du domaine de Keycloak. Les développeurs peuvent récupérer les informations des utilisateurs, une fois qu'il a un token d'accès pour d'autres traitement comme la restriction des accès.

Conclusion

Keycloak est une solution open source avec plusieurs fonctionnalités. Il peut être déployé de différentes manières selon les besoins. Ce chapitre a présenté l'implémentation et les différentes modes de déploiement de cette solution. La mise en place d'une solution d'authentification unique doit être accompagnée d'une authentification forte (avec au moins deux facteurs d'authentifications).

CONCLUSION GENERALE

Dans ce mémoire, nous nous sommes donnés comme objectifs d'étudier et d'implémenter l'authentification unique au Ministère de l'Education Nationale. L'atteinte de notre objectif nous a non seulement permis de mieux nous familiariser avec les réalités du monde professionnel ; mais aussi de mettre en pratique nos compétences tout en découvrant nombreuses plates formes dans le domaine des réseaux informatiques.

Par ailleurs, la réalisation de ce mémoire et le climat de travail (la rigueur, le sérieux, le partage, l'entre aide, etc.) de l'équipement SIMEN ont été des étapes très enrichissantes à tous les niveaux.

Sur le plan personnel tout d'abord, l'apprentissage de toutes les technologies utilisées puis la compréhension de l'architecture et le fonctionnement de Keycloak nous ont demandé beaucoup de travail de recherche, de documentation et d'investissement personnel. Ce qui était une occasion pour nous, de nous consacrer à la recherche documentaire sur l'Internet mais aussi sur les ouvrages généraux et articles scientifiques. A travers cette recherche également, nous avons eu l'opportunité d'analyser et/ou modifier au besoin le code Keycloak pour la compréhension de son fonctionnement. Toujours dans cette même logique, nous avons effectué de nombreux échanges fructueux avec mes camarades étudiants, nos collègues du MEN et nos encadrants. Ce n'est pas la complexité des technologies ou les concepts utilisés mais le nombre et la très grande quantité de connaissances ou de notions acquises et maîtrisées. Les très bonnes relations entretenues avec les autres membres de l'équipe toujours disponibles, en particulier mon tuteur qui a toujours eu la patience et pris le temps de répondre à toutes nos questions, nous ont permis d'apprendre, de progresser et d'avancer dans la recherche.

Avant de valider le choix de la solution Keycloak, notre choix s'était porté sur LemonLDAP-NG, mais un blocage majeur nous a poussé à changer de solution. A part ce blocage, nous n'avons pas eu de problème ni sur la documentation encore moins sur l'implémentation de la solution.

L'implémentation de la solution SSO règle le problème des multiples authentifications, mais engendre aussi des problèmes de sécurité dûs à l'unification des comptes utilisateurs. L'accès aux identités d'un utilisateur par une personne mal intentionnée lui procure l'accès à l'ensemble des applications du domaine dont cet utilisateur a accès. Pour durcir l'authentification, nous prévoyons d'implémenter une authentification double facteur sur notre solution SSO avec des codes à usage uniques qui seront délivrés par SMS.

Webographie :

[1] https ://www.web-d.be/post/94/l'attaquepar-fixation-de-session,-et-com http ment-s'en-prot%C3%A9ger.html, visité le 20/09/2021

[2] https://developer.mozilla.org/fr/docs/Glossary/URI, visité le 20/12/2021

[3] https://slideplayer.fr/slide/13821861/#:~:text=Une%20norme%20dans%20le%20secteur,%2C%20basic%2C%20HTML%E2%80%A6, visité le 02/11/2021

[4] https://fr.wikipedia.org/wiki/Authentification_unique, visité le 06/10/2021

[5] https://blog.devensys.com/aaa-authentication-authorization-ccounting/, visité le 11/11/2021

[6] https://ssi.ac-strasbourg.fr/bonnes-pratiques/recommandations/lidentification-et-lauthentification/#:~:text=L'authentification%20est%20une%20phase,secret%22%20que%20lui%20seul%20connait, visité le 11/11/2021

[7] https://www.education.sn/fr/standard/84, visité le 18/09/2021

[8] https://apprendre.auf.org/ressources-et-outils/simen-une-nouvelle-vision-du-management-de-leducation-au-senegal/, visité le 08/10/2021

[9] https://www.education.sn/fr/standard/82/, visité le 19/10/2021

[10] https://www.logiciels.pro/logiciel-saas/evidian-enterprise-sso/#:~:text=Nous%20avons%20identifi%C3%A9%20pr%C3%A8s%20de,13%20euros%20par%20licence%20utilisateur/, visité le 10/10/2021

[11] https://atos.net/fr/solutions/cybersecurite-produits/evidian-sso-identity-access-management/, visité le 12/11/2021

[12] https://aldian.developpez.com/tutoriels/javaee/authentification-centralisee-sso-cas, visité le 19/12/2021

[13] https://azure.microsoft.com/, visité le 20/11/2021

[14] https://aldian.developpez.com/tutoriels/javaee/authentification-centralisee-sso-cas/, visité le 28/11/2021

Bibliographie :

[15] Article Guide utilisateur de AWS SSO

[16] Mémoire de fin de cycle de Niang Pape Saer : Conception et implémentation d'une plateforme de fédération d'identité pour l'Université Virtuelle du Sénégal

[17] Mémoire de fin de cycle de Christophe Maudoux : Implémentation de l'authentification à double facteur dans la solution de SSO AAA (Authentification, Autorisation et Accès) LemonLDAP::NG 2.0

[18] Mémoire de fin de cycle de Moustapha Ndao : étude et optimisation d'une infrastructure hybride : cas EXPERT IS

[19] Tutoriel sur l'authentification centralisée SSO via le protocole CAS de Gauthier Perrineau

[20] CAS-jres2003-Article : Single Sign-On open-source avec CAS (Central Authentication Service) ; auteurs : Vincent Mathieu Université de Nancy 2, Pascal Aubry, IFSIC - Université de Rennes 1, Julien Marchal, Université de Nancy 2

[21] La thèse de Michel Kamel : Patrons organisationnels et techniques pour la sécurisation des Organisations Virtuelles

[22] Documentation officielle du SIMEN

[23] ANSSI Recommandations pour la sécurisation de la mise en œuvre du protocole OpenID Connect.