

Table des matières

INTRODUCTION	3
I. CONTEXTE D'UTILISATION DE KEYCLOAK	4
1. Dans un réseau communautaire	4
2. Cas particulier : Dans une université (Ecole National Supérieure Polytechnique de Yaoundé 1).....	4
II. GÉNÉRALITÉ SUR KEYCLOAK	4
1. Historique	4
2. Qu'est-ce que Keycloak ?	5
3. Présentation succincte de l'interface du logiciel	5
a. Section configure	6
b. Section manage	6
4. Caractéristiques de Keycloak	6
a. Distributions de Keycloak	6
b. Composants	6
c. Spécificités	7
d. Les méthodes d'authentification prises en charge par keycloak	8
5. Comparaison entre Keycloak et d'autres outils similaires	8
a. Okta.....	8
b. Auth0	9
c. Microsoft Azure Active Directory	9
d. Gluu.....	10
e. FreeIPA.....	10
6. Pourquoi choisir Keycloak comme solution ?	10
III. IMPLÉMENTATION	11
1. Outils utilisés pour notre implémentation.....	11
2. Procédure de travail Keycloak sur un système sécurisé à clé	11
3. Captures d'écran et détails sur l'implémentation	12
IV. DIFFICULTÉS RENCONTRÉES ET PERSPECTIVES	13
1. Difficultés rencontrées	13
2. Perspectives pour résoudre les problèmes rencontrés	13
CONCLUSION	14
ANNEXES.....	15

GUIDE D'INSTALLATION ET DE CONFIGURATION DE KEYCLOAK.....	15
1. SOUS WINDOWS.....	15
2. SOUS UBUNTU (18.04).....	19
RÉFÉRENCES	30

INTRODUCTION

L'authentification pour un système informatique est un processus permettant audit système de s'assurer de la légitimité d'une demande d'accès faite par une entité (être humain ou un autre système...) afin d'autoriser l'accès de cette entité à des ressources du système (systèmes, réseaux, applications...) conformément au paramétrage du contrôle d'accès. Bon nombre d'outils ont été développés dans le but de gérer les authentifications à l'instar de Keycloak, qui fait l'objet de notre exposé. De ce fait, il sera question pour nous par la suite de statuer sur ce logiciel à travers une présentation du contexte et des généralités, appuyée par une implémentation.

I. CONTEXTE D'UTILISATION DE KEYCLOAK

Dans cette partie, nous présenterons les contextes pertinents dans lequel nous pouvons faire appel à keycloak.

1. Dans un réseau communautaire

L'authentification pour toute structure (estudiantine, entreprise, etc..) est vitale. Dans un réseau communautaire par exemple, la solution d'authentification résout de nombreux problèmes à l'instar de :

- Le contrôle d'accès à différents services (commerce électronique, services de télésanté, l'information sur la sécurité publique, ...)
- Traçabilité des actions dans le réseau communautaire
- La sécurisation du réseau communautaire
- La gestion des droits dans le réseau
- ...

2. Cas particulier : Dans une université (Ecole National Supérieure Polytechnique de Yaoundé 1)

La communauté universitaire n'est pas à négliger, car elle présente de nombreux enjeux. En effet, certains services sont très spécifiques à des universités précises, qu'ils n'ont pas besoins de déployer sur des serveurs trop éloignés pour que les étudiants/personnel de l'établissement aient accès. À l'instar de la messagerie (texte, voix) au sein du campus, la consultation des notes, la consultation des informations et communiqués importants, les échanges de fichiers, l'inscription, les repositories, ... Tous les services précédemment cités nécessitent d'être sécurisés et keycloak est un le choix idéal pour cela.

II. GÉNÉRALITÉ SUR KEYCLOAK

Dans cette partie, nous décrivons aussi précisément que possible le logiciel d'authentification keycloak.

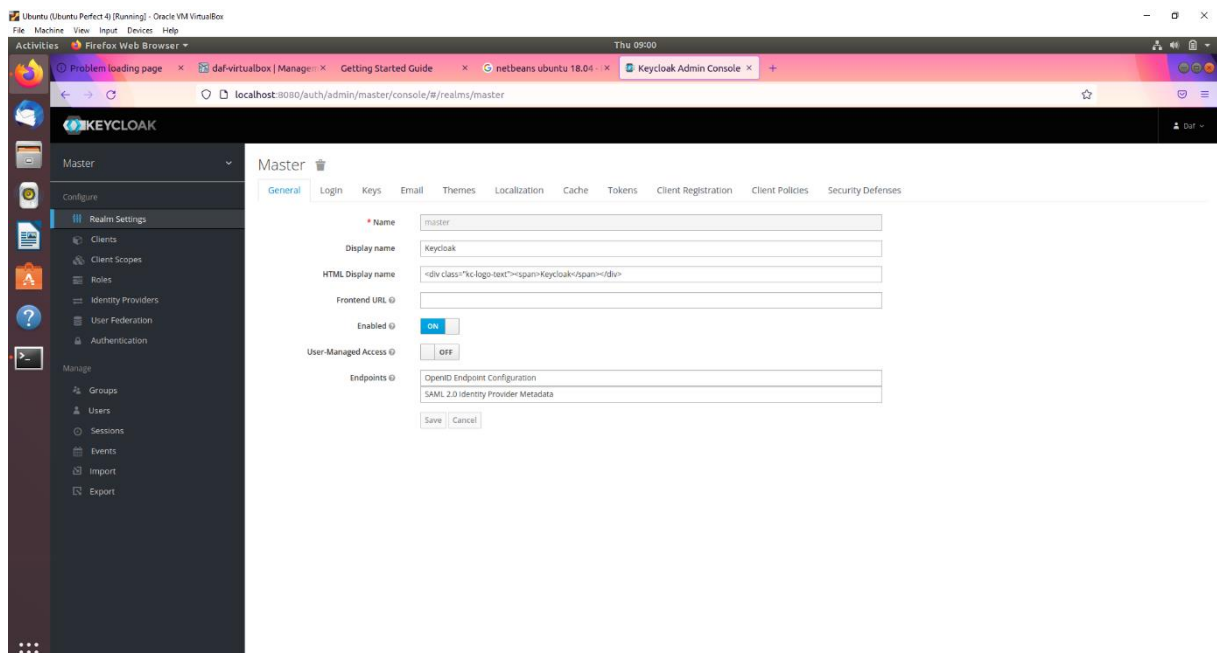
1. Historique

La première version de production de Keycloak a eu lieu en septembre 2014, le développement ayant commencé environ un an plus tôt. En 2016, Red Hat est passé du produit RH SSO basé sur le framework PicketLink à basé sur le projet en amont Keycloak. Cela faisait suite à une fusion de la base de code PicketLink dans Keycloak. Depuis mars 2018, JBoss.org redirige l'ancien sous-site jbosssso vers le site Web 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 AS vers Wildfly étant un exemple plus communément reconnu.

2. Qu'est-ce que Keycloak ?

Keycloak est un outil de gestion des identités et des accès, il facilite la sécurisation des applications et des services avec peu ou pas de code. Il permet aux utilisateurs de s'authentifier via Keycloak plutôt que d'utiliser des applications individuelles. Cela signifie que leurs applications n'ont pas à gérer les formulaires de connexion, à authentifier les utilisateurs ou à stocker les utilisateurs. Les utilisateurs n'ont pas à se reconnecter pour accéder à une autre application une fois qu'ils se sont connectés à Keycloak à partir d'une application particulière. De plus, Keycloak est un outil open source actuellement sous licence Apache Licence 2.0 et actuellement à la version 15.0.2 (2021-10-09). Il s'agit d'un projet en Amont pour Red Hat SSO (Red Hat Single Sign-on), c'est une initiative de Red Hat qui vise à offrir une Authentification Unique et qui permet de sécuriser des applications web en fournissant des fonctionnalités web Single Sign-on (SSO) basées sur des normes populaires telles que SAML 2.0, OpenID Connect et OAuth 2.0.

3. Présentation succincte de l'interface du logiciel



a. [Section configure](#)

- **Master** : permet de créer des realms c'est-à-dire
- **Clients** : permet de créer des clients
- **Client scopes** : permet de définir la portée, les droits des clients
- **Roles** : permet de définir les rôles
- **Identify providers**
- **User federation**
- **Authentication**

b. [Section manage](#)

- **Groups** :
- **Users** :
- **Sessions** :
- **Events** :
- **Import** :
- **Export** :

4. [Caractéristiques de Keycloak](#)

a. [Distributions de Keycloak](#)

Actuellement Keycloak a trois distributions principales disponible sur :

❖ **Serveur**

L'application autonome est téléchargeable à partir de la page Keycloak sous la forme d'une archive tar ou zip avec tous les scripts, documents et ressources nécessaires pour fonctionner normalement. Pour l'instant, il existe deux versions principales de cette distribution : l'une est alimentée par le serveur WildFly tandis que l'autre est alimentée par Quarkus.

❖ **Image Docker**

Distribution appropriée pour Docker, Podman, Kubernetes et OpenShift. Il existe deux images Docker officielles pour Keycloak : l'une est conservée dans Quay Container Registry-**quay.io/keycloak/keycloak**, la seconde est conservée dans Docker Hub-**jboss/keycloak**. Les deux sont téléchargeables avec la commande « docker pull ».

❖ **Opérateur**

Distributions pour kubernetes et OpenShift basée sur Operator SDK.

b. [Composants](#)

Keycloak est composé de deux composants principaux à savoir :

- ❖ **Un serveur Keycloak**
- ❖ **Un adaptateur d'application**

c. Spécificités

❖ **Prise en charge de plusieurs protocoles**

Pour l'instant, Keycloak prend en charge trois protocoles différents, à savoir - OpenID Connect, OAuth 2.0 et SAML 2.0.

❖ **Authentification unique**

Keycloak a un support complet pour Single Sign-On et Single Sign-Out. Les utilisateurs n'ont pas à se reconnecter pour accéder à une autre application une fois qu'ils se sont connectés à Keycloak à partir d'une application particulière. La déconnexion unique permet aux utilisateurs de se déconnecter une fois pour être déconnectés de toutes les applications qui utilisent Keycloak. Il en est de même pour la connexion.

❖ **Console d'administration**

Keycloak propose une interface graphique Web où l'on peut « cliquer » sur toutes les configurations requises par une instance pour fonctionner comme on le souhaite.

❖ **Identité et accès de l'utilisateur**

Keycloak peut être utilisé comme un gestionnaire d'accès et d'identité d'utilisateur autonome en nous permettant de créer une base de données d'utilisateurs avec des rôles et des groupes personnalisés. Ces informations peuvent également être utilisées pour authentifier les utilisateurs au sein de notre application et en sécuriser certaines parties en fonction de rôles prédéfinis.

❖ **Fournisseurs d'identité sociale**

De plus, Keycloak permet d'utiliser des fournisseurs d'identité sociale. Il prend en charge Google, Twitter, Facebook, Stack Overflow mais, en fin de compte, l'on doit tous les configurer manuellement à partir du panneau d'administration. La liste complète des fournisseurs d'identité sociale pris en charge et leur manuel de configuration se trouvent dans la documentation de Keycloak.

❖ **Gestion des accès aux API**

Garantit que les appels avec accès authentifié peuvent entrer dans les API en général

❖ **Personnalisation des pages**

Keycloak permet de personnaliser toutes les pages qu'il affiche pour des utilisateurs. Ces pages sont au format «.ftl » de sorte que l'on puisse utiliser des HTML balises et des CSS styles classiques pour adapter la page au style de votre application et à la marque de votre entreprise. L'on peut également mettre du JavaScript personnalisés dans le cadre de la personnalisation des pages afin que les possibilités soient illimitées.

❖ **Keycloak comme gestionnaire de mots de passe**

Permet à l'administrateur de gérer, d'authentifier et de réinitialiser les mots de passe.

d. [Les méthodes d'authentification prises en charge par keycloak](#)

❖ **Authentification classique**

Permet aux utilisateurs de se connecter de la manière classique, c'est-à-dire avec un username et un mot de passe.

❖ **L'authentification multi facteur**

Utilise plusieurs facteurs pour vérifier l'identité d'un utilisateur pour une connexion. L'on peut utiliser par exemple la vérification via un sms de confirmation lors d'une tentative de connexion.

❖ **L'authentification via la connexion sociale**

Keycloak offre la possibilité de se connecter via un compte de réseau social (Facebook, ...).

5. [Comparaison entre Keycloak et d'autres outils similaires](#)

La comparaison sera faite sous forme de présentation des points forts et des points faibles des autres alternatives face à Keycloak.

a. [Okta](#)

❖ **Les points forts**

- Fourni une gestion sécurisée des identités et une authentification unique à n'importe quelle application, que ce soit dans le cloud, sur un site ou sur un appareil mobile.
- Usage plus intuitif
- Prend en charge la gestion de conformité
- Prend en charge la connexion sans mot de passe (permet de se connecter en utilisant d'autres méthodes telles que les clés SSH)
- Prend en charge la surveillance d'accès (représentation en direct, heures de travail)

❖ **Les points faibles**

- Plus cher que Keycloak
- Plus lent
- Lors de l'importation d'un fichier csv (ajout de nouveaux utilisateurs par exemple) dans okta, nous devons nous assurer que les noms de colonne doivent être les mêmes que la politique de répertoire d'okta, sinon une erreur d'importation peut survenir.
- Pas facile d'ajouter des applications pas déjà prises en charge

b. [Auth0](#)

❖ Les points forts

- Auth0 est un service cloud qui fournit un ensemble d'API et d'outils unifiés permettant l'authentification unique et la gestion des utilisateurs pour toute application, API ou appareil IoT
- Est classé en tant qu'authentification basée sur les risques (RBA)
- Complet et hautement personnalisable
- Utilisation de Javascript partout
- Les crochets et les règles offrent la possibilité d'étendre tous les workflows d'authentification possibles

❖ Les points faibles

- Les changements effectués dans un onglet affectent les changements dans un autre onglet.
- Très cher (pour la version payante)
- Quelques fois, l'interface utilisateur Auth0 ne reflète pas les dernières modifications apportées
- Certaines des options d'administration ne sont pas disponibles via l'API

c. [Microsoft Azure Active Directory](#)

❖ Les points forts

- Azure Active Directory est une solution cloud complète de gestion des identités et des accès qui fournit un ensemble robuste de fonctionnalités pour gérer les utilisateurs et les groupes et aider à sécuriser l'accès aux applications, y compris les services en ligne Microsoft comme Office 365 et un monde d'applications SaaS non Microsoft.
- Bénéficie d'un accès conditionnel en plus d'une authentification multi facteur pour aider à protéger et régir l'accès
- Possède des outils pour intégrer facilement l'identité dans des applications et services

❖ Les points faibles

- Plus onéreux que Keycloak
- Trop de gestion et très peu d'automatisation rendent les tâches supplémentaires requises fastidieuses
- La configuration et la migration vers Azure Active Directory sont pénibles et beaucoup plus complexes.
- Certaines fonctionnalités de sécurité avancées ne sont pas disponibles sans un abonnement Azure Active Directory Premium
- La politique de licence de Microsoft est beaucoup trop complexe pour être comprise par l'utilisateur type
- Microsoft Azure Active Directory jusqu'à présent possède une incapacité à s'intégrer à MacOS. En effet, les erreurs d'authentification peuvent être répandues pour les utilisateurs distants utilisant les systèmes d'exploitation MacOS.

d. [Gluu](#)

❖ **Points forts**

- Permet de gérer la conformité des authentifications
- Permet la connexion sans mots de passe
- Est compatible avec le déploiement sur mobile (Android, iOS)
- Est mieux adapté pour les grandes entreprises

❖ **Les points faibles**

- Peu de fonctionnalités sont disponibles sur la version basique
- Est plus onéreux que Keycloak
- Fonctionnement moins intuitif

e. [FreelPA](#)

❖ **Les points forts**

- Gère la conformité d'une authentification (Aide à faire des évaluations de risques, assure la compréhension des politiques et que les procédures soient suivies)
- Très adapté pour les grandes entreprises

❖ **Les points faibles**

- Pas de gestion des accès aux API (Il n'est pas garanti que les appels avec accès authentifié peuvent se passer dans les API)
- Pas de gestion des demandes d'accès (Il n'y a aucune aide pour visualiser et gérer les demandes d'accès soumises par des utilisateurs ou le personnel)
- Pas de gestion de rôles (Pas d'aide à la régulation d'accès au système en fonction des rôles des individus)
- Ne prend pas en charge la connexion sociale ; c'est-à-dire ne permet pas aux utilisateurs de s'authentifier avec les coordonnées de leurs réseaux sociaux

6. [Pourquoi choisir Keycloak comme solution ?](#)

Tout d'abord, Keycloak est une solution totalement gratuite ; ce qui est un atout considérable car généralement la plupart des outils dotés des fonctionnalités telles que Auth0 ou Okta sont payants.

Deuxièmement, Keycloak prend en charge les protocoles standard suivants : **OAuth 2.0**, **OpenID Connect** et **SAML 2.0**. Cette prise en charge signifie que tout outil ou application prenant en charge l'intégration avec les protocoles ci-dessous peut-être connecté à Keycloak (par exemple, des applications d'entreprise telles que Red Hat Ansible Tower ou SAP Business Intelligence Platform). Par ailleurs, le fait de prendre en charge ces trois protocoles d'authentification différents donnent la possibilité de couvrir de nombreuses applications avec des exigences de sécurité différentes avec un seul outil.

De plus, Keycloak peut être utilisé avec des bases de données utilisateurs existantes telles que LDAP, Active Directory, car dispose d'un mécanisme de synchronisation avec de tels

fournisseurs d'identité. Aussi, il prend en charge les fournisseurs d'identité sociale comme Google ou Facebook et fournit une interface graphique Web qui facilite les modifications de configuration. Sans compter la faible difficulté d'intégration à des projets déjà existants ou nouveaux.

III. IMPLÉMENTATION

Notre implémentation consistera à sécuriser une application React avec keycloak en mettant sur pied un système d'authentification d'utilisateurs et de permission sur l'application. Il s'agit en fait d'une bibliothèque numérique à laquelle sont affiliés des membres et des libraires ; et dont l'accès à des pages est autorisé ou non en fonction de leur différent rôle.

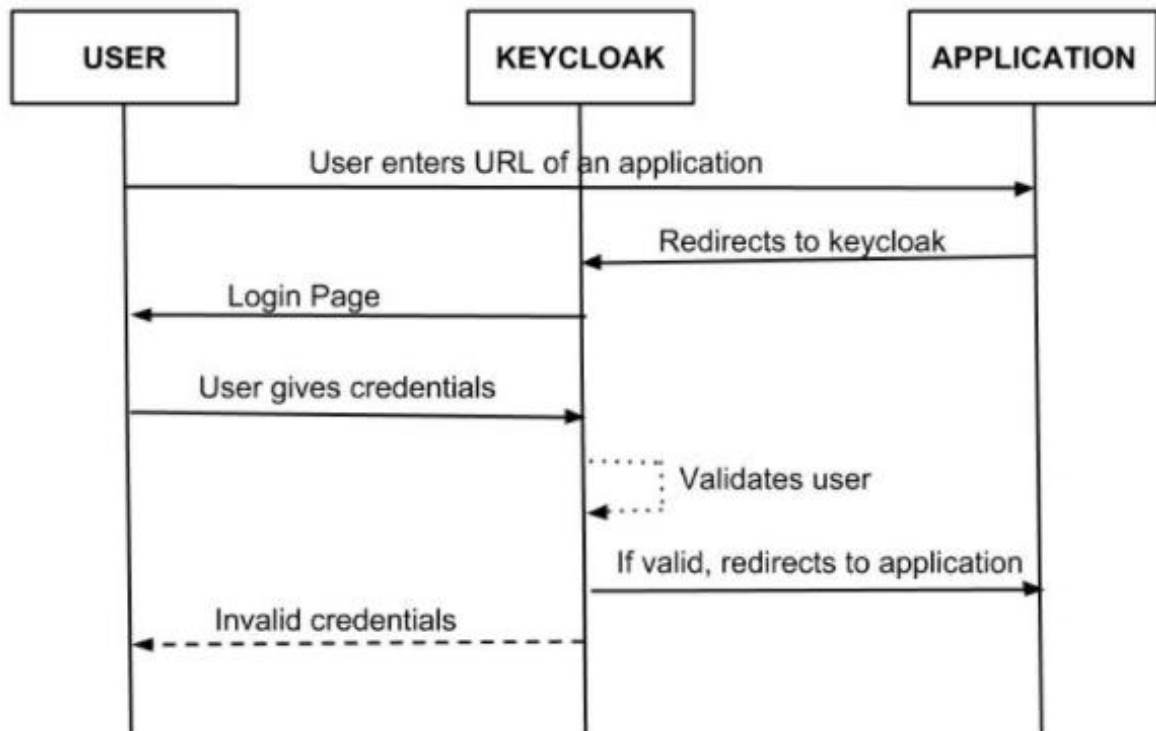
1. Outils utilisés pour notre implémentation

- Système d'exploitation : Windows
- Package : Keycloak-js
- Environnement de développement : Visual Studio code
- Langage de programmation : Javascript
- Framework : React JS

2. Procédure de travail Keycloak sur un système sécurisé à clé

- Un utilisateur clique depuis une page publique pour naviguer vers une zone protégée de l'application. Le lien vers cette zone protégée se trouve dans les paramètres de l'application dans la console d'administration de keycloak.
- L'utilisateur sera en effet redirigé vers la page d'authentification keycloak. Après avoir fourni le nom d'utilisateur et le mot de passe, keycloak redirige à nouveau l'utilisateur vers l'application avec un code valable sur une très courte période.
- L'application communique ce code à keycloak avec l'ID de l'application et le secret de l'application, puis les réponses keycloak avec le jeton d'accès, le jeton d'identification et un jeton d'actualisation. Votre application n'aura besoin que d'un seul de ces jetons pour voir quelles réclamations l'utilisateur a, et selon les réclamations, l'utilisateur se verra accorder ou se verra refuser l'accès aux URL protégées demandées.

Est joint ci-après un schéma illustratif résumant la procédure susmentionnée



3. [Captures d'écran et détails sur l'implémentation](#)

IV. DIFFICULTÉS RENCONTRÉES ET PERSPECTIVES

1. Difficultés rencontrées

Bien que l'expérience, l'étude de keycloak ait été enrichissante, il est tout à fait logique que nous ayons rencontrés quelques difficultés à l'instar de :

- Du fait que c'était la première fois pour nous d'entendre parler de ce logiciel, il nous a fallu un certain temps afin de nous approprier son fonctionnement.
- Nous avons dû apprendre aussi le langage React JS pour notre implémentation.
- Certaines commandes pour l'installation de keycloak sur Ubuntu étaient obsolètes.
- La plupart des tutoriels et d'exemples que nous avons suivi pour créer une implémentation étaient soit beaucoup trop complexe pour des novices, soit obsolètes.
- Lors de l'installation sur Ubuntu, nous avons rencontré un problème avec le port 8080 car du fait qu'apache utilise de base ce port, cela créait un conflit avec Keycloak car celui-ci est défini sur le même port.

2. Perspectives pour résoudre les problèmes rencontrés

En rapport avec les problèmes susmentionnés, voici quelques solutions que nous avons apportées/proposées :

- Toujours vérifier (sur Ubuntu) les ports utilisés par les différents serveurs que l'on possède et les changer si nécessaire.
- Dans le guide user nous avons mis à jour quelques commandes qui nécessitaient d'être mises à jour
-

CONCLUSION

ANNEXES

GUIDE D'INSTALLATION ET DE CONFIGURATION DE KEYCLOAK

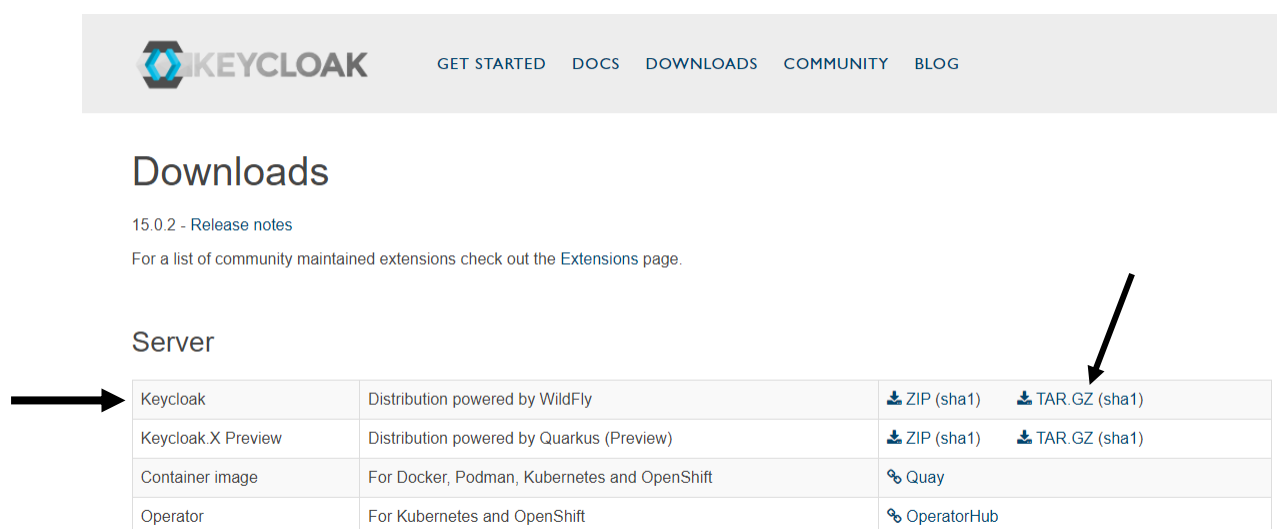
1. SOUS WINDOWS

Cette section décrit comment installer et démarrer un serveur Keycloak en mode autonome, se connecter à la console d'administration Keycloak.

Etape1 : Télécharger keycloak server

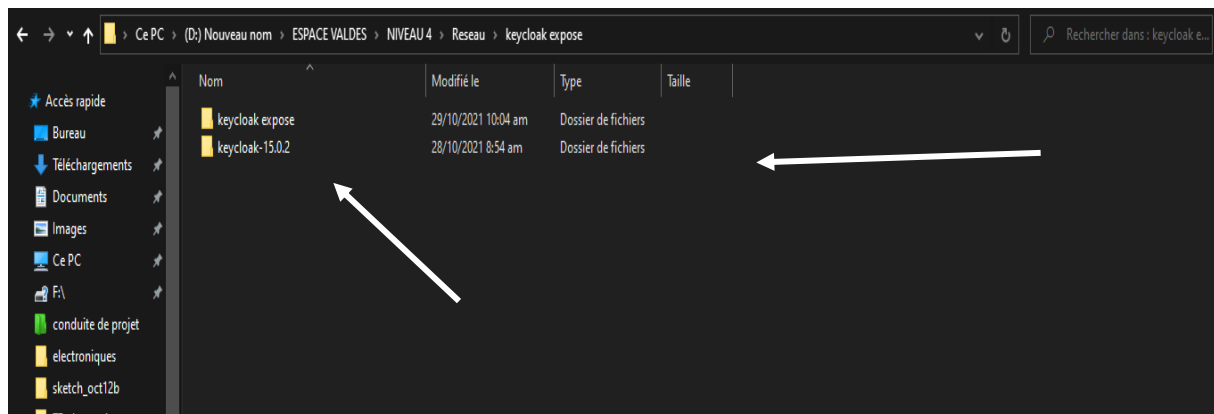
L'installation du serveur peut se faire sur Linux ou Windows. Le fichier ZIP de téléchargement du serveur contient les scripts et les binaires pour exécuter le serveur Keycloak, dans ce document nous montrerons comment installer le serveur Keycloak sur Windows.

1. Télécharger **keycloak-15.0.2.[zip]** depuis les [téléchargements Keycloak](https://www.keycloak.org/downloads.html) (depuis l'adresse <https://www.keycloak.org/downloads.html>).



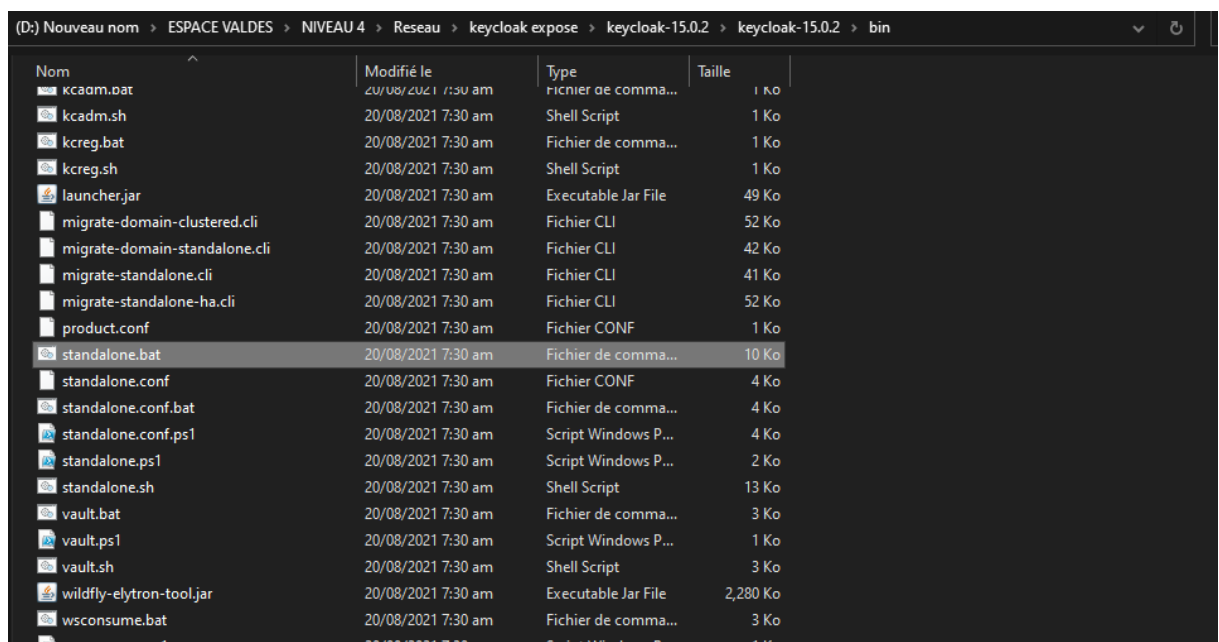
Server		
Keycloak	Distribution powered by WildFly	ZIP (sha1) TAR.GZ (sha1)
Keycloak X Preview	Distribution powered by Quarkus (Preview)	ZIP (sha1) TAR.GZ (sha1)
Container image	For Docker, Podman, Kubernetes and OpenShift	Quay
Operator	For Kubernetes and OpenShift	OperatorHub

2. Placer le fichier dans un répertoire de votre choix.
3. Décompresser le fichier ZIP à l'aide de l'unzip utilitaire approprié, tel que unzip ou le faire avec le logiciel Winrar.



Etape 2 : Lancement et réglage du port utilisé par Keycloak

1. Aller dans le répertoire bin de la distribution du serveur.
2. Exécuter le script « standalone » de démarrage, soit en double cliquant sur « standalone.bat » si l'on n'a aucun service fonctionnant sur le port 8080 soit en fournissant une valeur pour la `jboss.socket.binding.port-offset` approprié pour notre système. Cette valeur est ajoutée à la valeur de base de chaque port (8080) ouvert par le serveur Keycloak. Dans cet exemple, **100** est la valeur pour cela taper la commande « **standalone.bat -Djboss.socket.binding.port-offset=100** » depuis le répertoire « bin ».



- Exécution de la commande pour le réglage du port et le lancement du serveur Keycloak.


```
C:\Windows\System32\cmd.exe
D:\ESPACE VALDES\NIVEAU 4\Reseau\keycloak expose\keycloak-15.0.2\keycloak-15.0.2\bin>standalone.bat -Djboss.socket.binding.port-offset=100
```

➤ Démarrage du Serveur Keycloak.

```
C:\Windows\System32\cmd.exe - standalone.bat -Djboss.socket.binding.port-offset=100
D:\ESPACE VALDES\NIVEAU 4\Reseau\keycloak expose\keycloak-15.0.2\keycloak-15.0.2\bin>standalone.bat -Djboss.socket.binding.port-offset=100
Calling "D:\ESPACE VALDES\NIVEAU 4\Reseau\keycloak expose\keycloak-15.0.2\keycloak-15.0.2\bin\standalone.conf.bat"
JAVA_HOME is not set. Unexpected results may occur.
Set JAVA_HOME to the directory of your local JDK to avoid this message.
=====
JBoss Bootstrap Environment

JBOSS_HOME: "D:\ESPACE VALDES\NIVEAU 4\Reseau\keycloak expose\keycloak-15.0.2\keycloak-15.0.2"

JAVA: "java"

JAVA_OPTS: "-Dprogram.name=standalone.bat -Xms64M -Xmx512M -XX:MetaspaceSize=96M -XX:MaxMetaspaceSize=256M -Djava.net.preferIPv4Stack=true -Djboss.modules.system.pkgs=org.jboss.byteman -Djava.awt.headless=true --add-exports=java.base/sun.nio.ch=ALL-UNNAMED --add-exports=jdk.unsupported/sun.misc=ALL-UNNAMED --add-exports=jdk.unsupported/sun.reflect=ALL-UNNAMED"
=====
```

➤ Lancement du serveur Keycloak

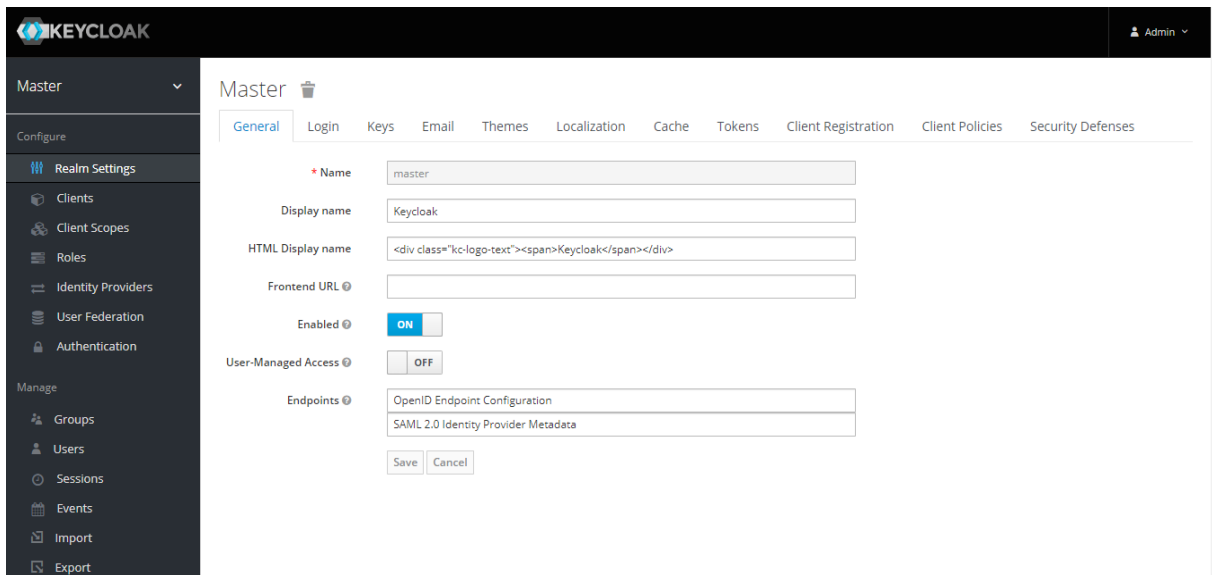
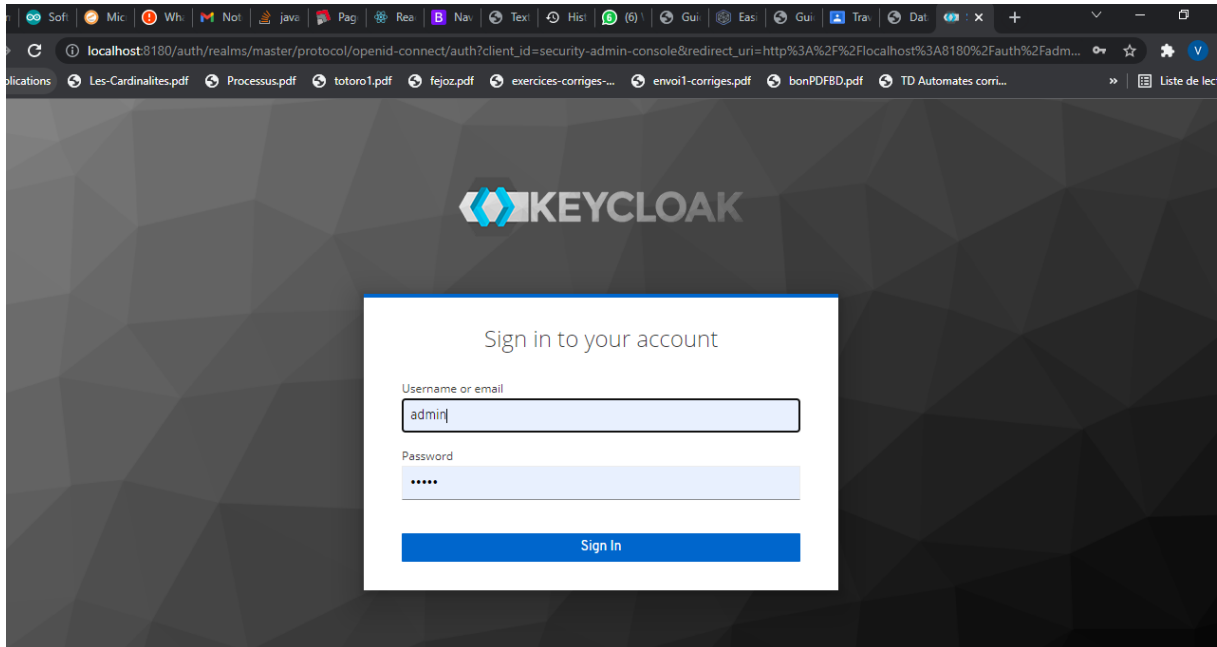
```
C:\Windows\System32\cmd.exe - standalone.bat
11:30:30,671 INFO [org.jboss.resteasy.resteasy_jaxrs.i18n] (ServerService Thread Pool -- 57) RESTEASY002200: Adding class resource org.keycloak.services.resources.JsResource from Application class org.keycloak.services.resources.KeycloakApplication
11:30:30,671 INFO [org.jboss.resteasy.resteasy_jaxrs.i18n] (ServerService Thread Pool -- 57) RESTEASY002220: Adding singleton resource org.keycloak.services.resources.RealmsResource from Application class org.keycloak.services.resources.KeycloakApplication
11:30:30,671 INFO [org.jboss.resteasy.resteasy_jaxrs.i18n] (ServerService Thread Pool -- 57) RESTEASY002220: Adding singleton resource org.keycloak.services.resources.RobotsResource from Application class org.keycloak.services.resources.KeycloakApplication
11:30:30,671 INFO [org.jboss.resteasy.resteasy_jaxrs.i18n] (ServerService Thread Pool -- 57) RESTEASY002220: Adding singleton resource org.keycloak.services.resources.admin.AdminRoot from Application class org.keycloak.services.resources.KeycloakApplication
11:30:30,687 INFO [org.jboss.resteasy.resteasy_jaxrs.i18n] (ServerService Thread Pool -- 57) RESTEASY002210: Adding provider singleton org.keycloak.services.util.ObjectMapperResolver from Application class org.keycloak.services.resources.KeycloakApplication
11:30:30,858 INFO [org.wildfly.extension.undertow] (ServerService Thread Pool -- 57) WFLYUT0021: Registered web context: '/auth' for server 'default-server'
11:30:31,172 INFO [org.jboss.as.server] (ServerService Thread Pool -- 43) WFLYSRV0010: Deployed "keycloak-server.war" (runtime-name : "keycloak-server.war")
11:30:31,313 INFO [org.jboss.as.server] (Controller Boot Thread) WFLYSRV0212: Resuming server
11:30:31,313 INFO [org.jboss.as] (Controller Boot Thread) WFLYSRV0025: Keycloak 15.0.2 (WildFly Core 15.0.1.Final) started in 44680ms - Started 594 of 872 services (584 services are lazy, passive or on-demand)
11:30:31,328 INFO [org.jboss.as] (Controller Boot Thread) WFLYSRV0060: Http management interface listening on http://127.0.0.1:9990/management
11:30:31,328 INFO [org.jboss.as] (Controller Boot Thread) WFLYSRV0051: Admin console listening on http://127.0.0.1:9990
```

Etape 3 : Lancement de la Console d'administration Keycloak

1. Confirmer que le serveur Keycloak est en cours d'exécution. si l'on a juste double cliquer sur le script « standalone.bat » du répertoire « bin » ouvrez <http://localhost:8080/auth> dans votre navigateur Web mais si l'on a ajouté la valeur 100 au port par défaut de lancement du serveur Keycloak en

tapant la commande « **standalone.bat -Djboss.socket.binding.port-offset=100** » ouvrons plutôt <http://localhost:8180/auth> dans votre navigateur Web.

2. Entrer « **admin** » comme username et password, si la console d'administration s'ouvre, l'on a lancé avec succès votre serveur Keycloak.



2. SOUS UBUNTU (18.04)

Ouvrir le terminal et suivre très exactement les étapes ci-dessous :

Étape 1 : installer le JDK

\$ **java -version** pour vérifier si java est installé. Si java n'est pas installé, il sera affiché « java : commande non trouvée ». Il suffira donc d'exécuter les commandes ci-dessous pour installer Java :

\$ **sudo apt-get update**

\$ **sudo apt-get install default-jdk -y**

Après l'installation, il faut vérifier si Java est correctement installé en exécutant la commande \$ **java -version** qui est supposé afficher ceci :

```
openjdk version "11.0.3" 2019-04-16
OpenJDK Runtime Environment (build 11.0.3+7-Ubuntu-1ubuntu218.04.1)
OpenJDK 64-Bit Server VM (build 11.0.3+7-Ubuntu-1ubuntu218.04.1, mixed mode, sharing)
```

Étape 2 : Télécharger et extraire le serveur Keycloak

Tout d'abord, l'on change de répertoire en /opt et télécharge Keycloak dans ce répertoire.

\$ **cd/opt**

\$ **sudo wget https://www.keycloak.org/downloads/keycloak-15.0.2.tar.gz**

Ceci fait, on extrait le package tar et on renomme le répertoire extrait en keycloak . Ce sera le répertoire d'installation de Keycloak

\$ **sudo tar -xvzf keycloak-15.0.2.tar.gz**

\$ **sudo mv keycloak-15.0.2 /opt/keycloak**

Étape 3 : Créer un utilisateur et un groupe pour Keycloak

Nous ne devons pas exécuter Keycloak sous l'utilisateur root pour des raisons de sécurité. Nous créerons donc un keycloak de groupe et ajouterons un keycloak utilisateur.

De plus, le répertoire personnel de l'utilisateur keycloak sera le répertoire d'installation de Keycloak, c'est -à- dire /opt/ keycloak.

\$ **sudo groupadd keycloak**

\$ **sudo useradd -r -g keycloak -d /opt/keycloak -s /sbin/nologin keycloak**
\$ **sudo useradd -r -g keycloak -d /opt/keycloak -s /sbin/nologin keycloak**

Étape 4 : Modifier l'autorisation et la propriété du répertoire d'installation de Keycloak

Ensuite, nous allons modifier la propriété et l'autorisation du répertoire /opt/keycloak . Nous donnerons également des autorisations exécutables au répertoire / opt/keycloak/bin/ . Dans le répertoire /opt, exécutons les commandes suivantes :

```
$ sudo chown -R keycloak: keycloak
```

```
$ sudo chmod o+x /opt/keycloak/bin/
```

Étape 5 : Création d'un fichier de service SystemD pour Keycloak

Créons un répertoire de configuration pour Keycloak sous le répertoire /etc sous le nom keycloak.

```
$ cd /etc/
```

```
$ sudo mkdir keycloak
```

Copions ensuite le fichier de configuration Keycloak / opt / keycloak / docs / contrib / scripts / systemd /wildfly.conf dans /etc/keycloak/ et renommons-le en keycloak.conf

```
$ sudo cp/opt/keycloak/docs/contrib/scripts/systemd/wildfly.conf/etc/keycloak/  
keycloak.conf
```

Ensuite, copions le script de lancement Keycloak (de launch.sh) sous / opt / keycloak / docs / contrib / scripts / systemd / à / opt / keycloak / bin / répertoire

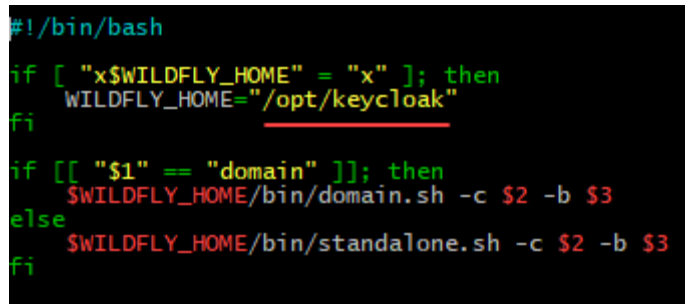
```
$ sudo cp /opt/keycloak/docs/contrib/scripts/systemd/launch.sh /opt/keycloak/bin/
```

Nous devons faire de keycloak user le propriétaire de ce script pour qu'il puisse l'exécuter :

```
$ sudo chown keycloak : /opt/keycloak/bin/launch.sh
```

Ensuite, nous devons corriger le chemin d'installation de Keycloak dans launch.sh, donc ouvrons launch.sh dans un éditeur. Mettre à jour le chemin d'installation de Keycloak comme indiqué ci-dessous dans la capture puis enregistrer et quitter :

```
$ sudo nano /opt/keycloak/bin/launch.sh
```



```
#!/bin/bash  
  
if [ "x$WILDFLY_HOME" = "x" ]; then  
    WILDFLY_HOME="/opt/keycloak"  
fi  
  
if [[ "$1" == "domain" ]]; then  
    $WILDFLY_HOME/bin/domain.sh -c $2 -b $3  
else  
    $WILDFLY_HOME/bin/standalone.sh -c $2 -b $3  
fi
```

Maintenant, copions le fichier de définition de service (wildfly.service) sous /opt/keycloak/docs/contrib/scripts/systemd/ dans le répertoire /etc/systemd/system/ et renommons-le en keycloak.service

```
$ sudo cp /opt/keycloak/docs/contrib/scripts/systemd/wildfly.service
/etc/systemd/system/ keycloak.service
```

Par la suite, ouvrons keycloak.service dans un editeur

```
$ sudo nano /etc/systemd/system/ keycloak.service
```

Apportons les modifications marquées en gras ou simplement copier/coller le contenu ci-dessous tel quel puis sauvegarder et quitter.

```
[Unit]
Description=The Keycloak Server
After=syslog.target network.target
Before=httpd.service[Service]
Environment=LAUNCH_JBOSS_IN_BACKGROUND=1
EnvironmentFile=/etc/keycloak/keycloak.conf
User=keycloak
Group=keycloak
LimitNOFILE=102642
PIDFile=/var/run/keycloak/keycloak.pid
ExecStart=/opt/keycloak/bin/launch.sh $WILDFLY_MODE
$WILDFLY_CONFIG $WILDFLY_BIND
StandardOutput=null[Install]
WantedBy=multi-user.target
```

Rechargeons la configuration du gestionnaire systemd et activons le service keycloak au démarrage du système.

```
$ sudo systemctl daemon-reload
```

```
$ sudo systemctl enable keycloak
```

Pour démarrer le service du système keycloak :

```
$ sudo systemctl start keycloak
```

Une fois le service démarré, nous pouvons vérifier l'état en exécutant la commande ci-dessous :

```
$ sudo systemctl status keycloak
```

Si le service a démarré avec succès, nous devrions voir quelque chose comme ci-dessous :

```
● keycloak.service - The Keycloak Server
   Loaded: loaded (/etc/systemd/system/keycloak.service; enabled; vendor preset: enabled)
   Active: active (running) since Sat 2019-07-27 16:59:21 UTC; 28min ago
     Main PID: 8281 (launch.sh)
       Tasks: 51 (limit: 2360)
    CGroup: /system.slice/keycloak.service
            └─8281 /bin/bash /opt/keycloak/bin/launch.sh standalone standalone.xml 0.0.0.0
               └─8282 /bin/sh /opt/keycloak/bin/standalone.sh -c standalone.xml -b 0.0.0.0
                  └─8374 java -D[Standalone] -server -Xms64m -Xmx512m -XX:MetaspaceSize=96M -XX:MaxMetaspaceSize=256m -Djava.net.preferIPv4Stack=true -Djboss.modules.system.pkgs
Jul 27 16:59:21 : systemd[1]: Started The Keycloak Server.
```

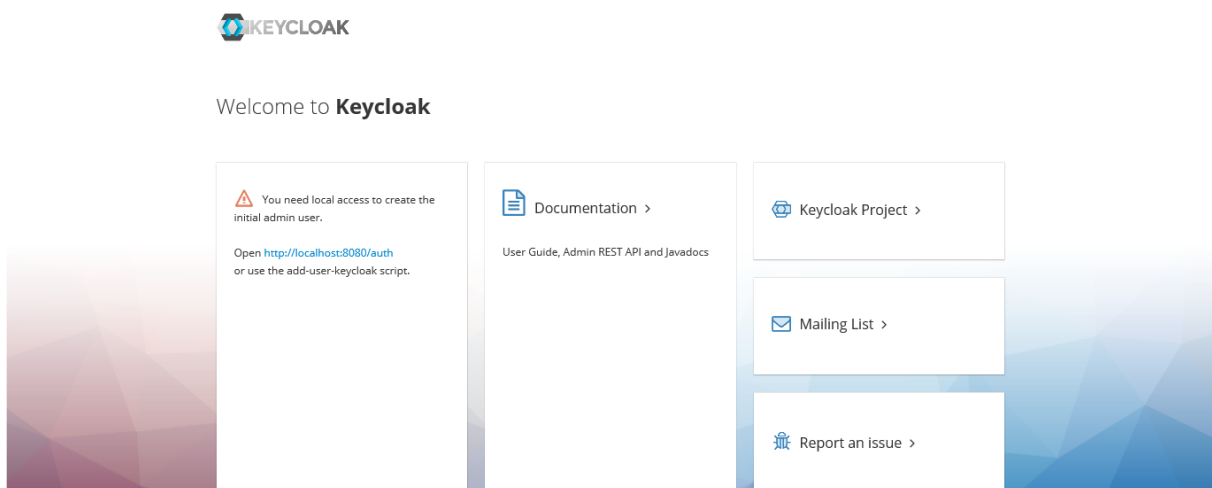
L'état actif, comme mis en évidence, ci-dessus vérifie que le service est opérationnel. Nous pouvons également suivre les journaux du serveur Keycloak avec la commande ci-dessous :

\$ sudo tail -f /opt/keycloak/standalone/log/server.log

```
2019-07-28 12:04:13,925 INFO [org.jboss.resteasy.resteasy_jaxrs.i18n] (ServerService Thread Pool -- 66) RESTEASY002210: Adding provider singleton org.keycloak.services.util.ObjectMapperResolver
2019-07-28 12:04:13,928 INFO [org.jboss.resteasy.resteasy_jaxrs.i18n] (ServerService Thread Pool -- 66) RESTEASY002220: Adding singleton resource org.keycloak.services.resources.WelcomeResource
2019-07-28 12:04:13,929 INFO [org.jboss.resteasy.resteasy_jaxrs.i18n] (ServerService Thread Pool -- 66) RESTEASY002220: Adding singleton resource org.keycloak.services.resources.RealmsResource
2019-07-28 12:04:13,930 INFO [org.jboss.resteasy.resteasy_jaxrs.i18n] (ServerService Thread Pool -- 66) RESTEASY002220: Adding singleton resource org.keycloak.services.resources.admin.AdminResource
2019-07-28 12:04:14,073 INFO [org.wildfly.extension.undertow] (ServerService Thread Pool -- 68) WFLYUT0021: Registered web context: '/auth' for server 'default-server'
2019-07-28 12:04:14,191 INFO [org.jboss.as.server] (ServerService Thread Pool -- 53) WFLYSRV0010: Deployed "keycloak-server.war" (runtime-name : "keycloak-server.war")
2019-07-28 12:04:14,288 INFO [org.jboss.as] (Controller Boot Thread) WFLYSRV0212: Resuming server
2019-07-28 12:04:14,291 INFO [org.jboss.as] (Controller Boot Thread) WFLYSRV0060: Http management interface listening on http://0.0.0.0:9990/management
2019-07-28 12:04:14,291 INFO [org.jboss.as] (Controller Boot Thread) WFLYSRV0051: Admin console listening on http://0.0.0.0:9990
2019-07-28 12:04:14,291 INFO [org.jboss.as] (Controller Boot Thread) WFLYSRV0025: Keycloak 6.0.1 (WildFly Core 8.0.0.Final) started in 23626ms - Started 580 of 842 services (560 services are lazy, passive or on-demand)
```

Accédez maintenant au serveur Keycloak à l'adresse : <http://adresseip:8080/auth/>

NB : pour obtenir l'adresse IP de la machine il suffit de taper la commande ifconfig



Étape 6 : Créer l'utilisateur administrateur initial

Comme indiqué sur la page de destination, nous devons créer un compte administrateur initial pour pouvoir accéder à la console d'administration Keycloak. Keycloak n'est fourni avec aucun compte administrateur configuré prêt à l'emploi.

Le compte administrateur nous permettra de créer un administrateur qui peut se connecter à la console d'administration du domaine maître afin que nous puissions commencer à créer des domaines, des utilisateurs et enregistrer des applications à sécuriser par Keycloak.

Si nous accédons à Keycloak depuis localhost sur un navigateur, nous pouvons facilement créer cet utilisateur administrateur en accédant à <http://localhost:8080/auth>



Welcome to Keycloak

Please create an initial admin user to get started.

Username	<input type="text"/>
Password	<input type="password"/>
Password confirmation	<input type="password"/>
<input type="button" value="Create"/>	

[Documentation](#) | [Administration Console](#)

[Keycloak Project](#) | [Mailing List](#) | [Report an issue](#)



Spécifions simplement le nom d'utilisateur et le mot de passe pour cet administrateur initial et nous sommes prêts à partir.

Puisque nous accédons au serveur depuis l'extérieur de localhost, nous devons utiliser le script bash (add-user-keycloak.sh) disponible dans le répertoire /opt/keycloak/bin/ pour créer le compte administrateur initial.

\$ sudo /opt/keycloak/bin/add-user-keycloak.sh -r master -u <nom d'utilisateur> -p <mot de passe>

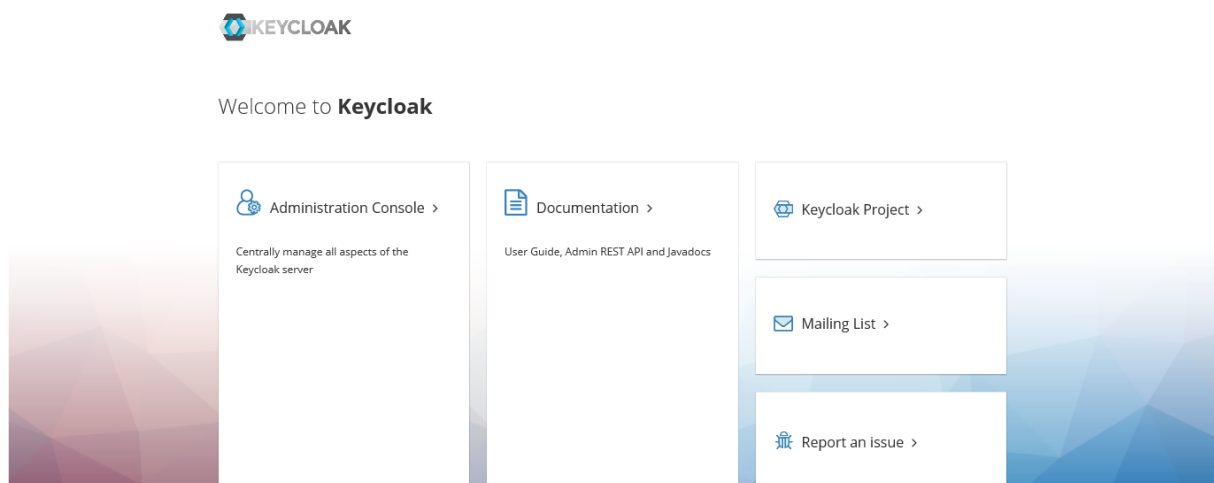
Lorsque Keycloak est démarré pour la première fois, Keycloak crée un domaine prédéfini pour nous. Ce royaume initial est le royaume maître. C'est le niveau le plus élevé dans la hiérarchie des royaumes. Les comptes d'administrateur de ce domaine sont autorisés à afficher et à gérer tout autre domaine créé sur l'instance de serveur.

Le compte administrateur initial que nous venons de créer est associé au domaine maître. Ainsi, plus tard dans ce didacticiel, notre connexion initiale à la console d'administration se fera également via le domaine maître à l'aide des informations d'identification d'administrateur que nous venons de créer.

Redémarrons le service keycloak :

\$ sudo systemctl reload keycloak

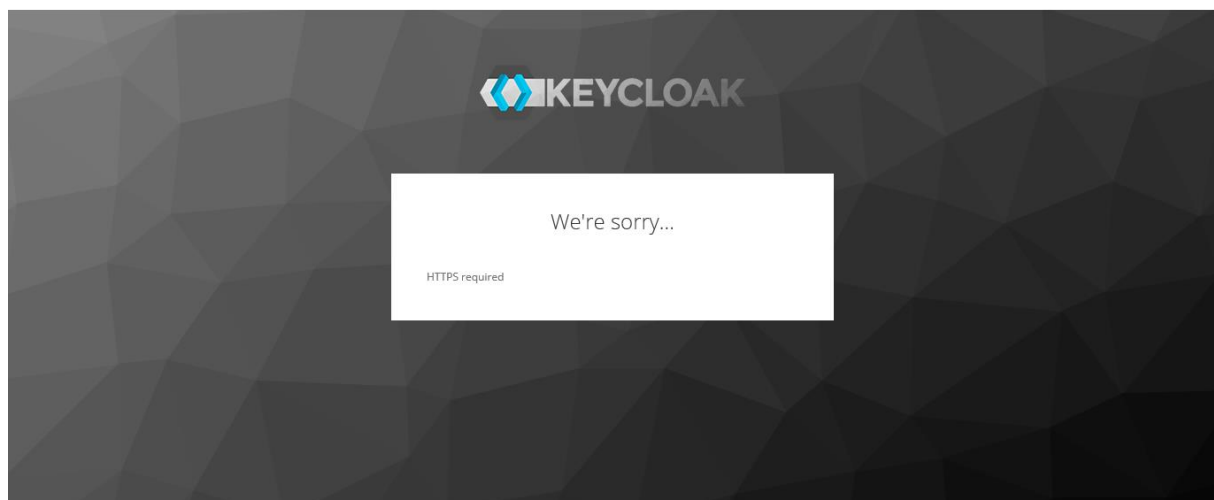
Une fois Keycloak redémarré, accédez à `http:// <instance-public-ip> :8080/auth/`



Comme nous pouvons le voir, le message nous demandant de créer l'utilisateur administrateur initial a disparu. Cliquez maintenant sur le lien Administration Console pour accéder à la console d'administration du domaine maître.

Étape 7: Désactivez SSL sur le domaine principal et connectez-vous à la console d'administration

Lorsque nous cliquons sur le lien Administration Console à l'étape précédente, nous obtenons le message d'erreur suivant :



L'erreur ci-dessus apparaît car Keycloak utilise désormais par défaut HTTPS pour toutes les adresses IP externes. Ce comportement par défaut s'applique également au domaine maître.

Keycloak peut fonctionner hors de la boîte sans SSL tant que nous restons fidèles à des adresses IP privées comme localhost, 127.0.0.1, 10.0.x.x, 192.168.x.x et 172.16.x.x. Si

SSL/HTTPS n'est pas configuré sur le serveur ou si nous essayons d'accéder à Keycloak via HTTP à partir d'une adresse IP non privée, nous obtiendrons l'erreur ci-dessus.

Pour contourner ce problème, nous avons besoin d'un moyen de désactiver SSL pour le domaine maître. Une façon de le faire consiste à utiliser les scripts Admin CLI qui sont empaquetés dans la distribution Keycloak Server. Nous pouvons trouver ces scripts dans le répertoire `/opt/keycloak/bin/`.

Le script Linux s'appelle `kcadm.sh` et le script pour Windows s'appelle `kcadm.bat`.

L'interface de ligne de commande d'administration fonctionne en effectuant des requêtes HTTP vers les points de terminaison REST d'administration. Leur accès est protégé et nécessite une authentification.

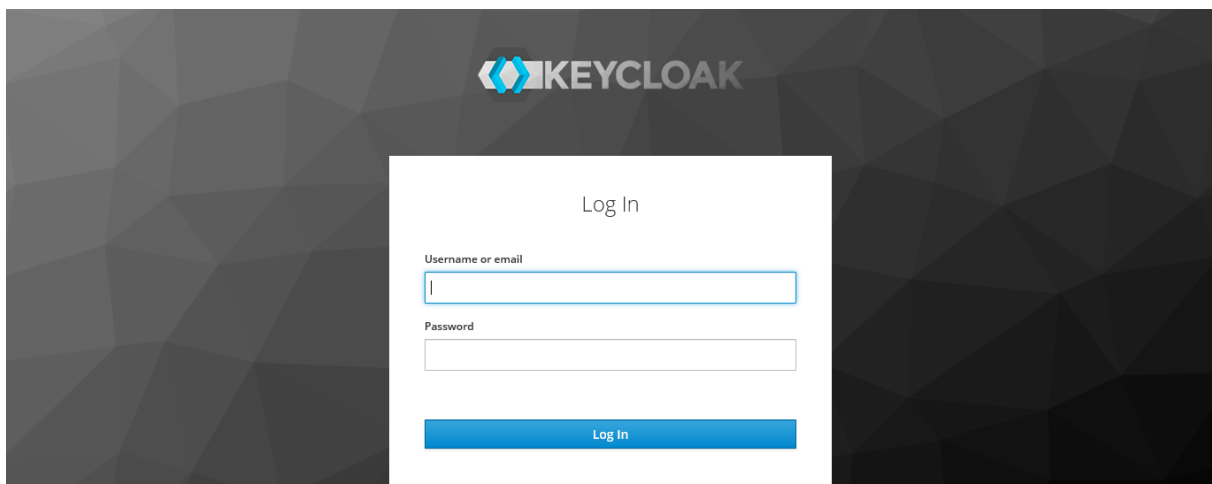
Nous pouvons démarrer une session authentifiée en fournissant les informations d'identification de l'utilisateur administrateur (créées à l'étape 6) et en nous connectant.

```
$ sudo /opt/keycloak/bin/kcadm.sh identifiants de configuration --server  
http://localhost:8080/auth --realm master --user <admin-username> --password <admin-  
password>
```

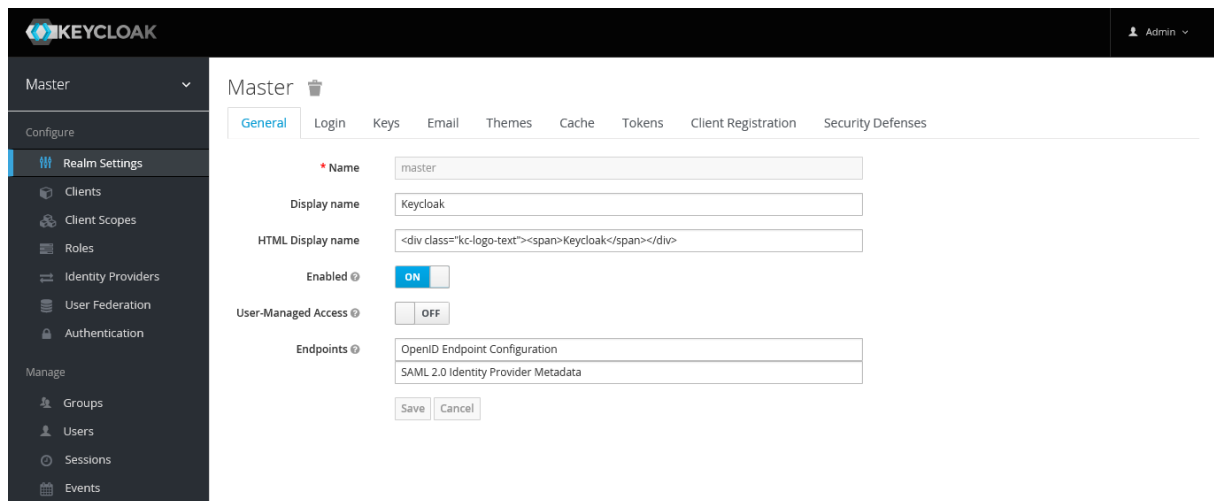
Une fois connecté, nous sommes prêts à effectuer des opérations de création, de lecture, de mise à jour et de suppression (CRUD). Pour désactiver SSL sur le domaine maître, nous pouvons utiliser la commande `update` :

```
$ sudo /opt/keycloak/bin/kcadm.sh update realms/master -s sslRequired=NONE
```

Nous n'avons pas besoin de redémarrer Keycloak pour cela, actualisez simplement la page d'erreur ou accédez à : `http:// <instance-public-ip> :8080/auth /admin/`



Pour se connecter, entrons le nom d'utilisateur et le mot de passe administrateur



Nous nous sommes connectés avec succès à la console d'administration du domaine maître. Nous pouvons désormais créer de nouveaux domaines, clients, rôles, groupes ou utilisateurs selon nos besoins.

Étape 8 : Configurer la console de gestion Keycloak

La console de gestion Keycloak nous permet de gérer différents aspects du serveur Keycloak. Par exemple, la configuration des sous-systèmes, la surveillance des serveurs, la gestion des déploiements ou le contrôle d'accès. Par défaut, la console de gestion n'est pas accessible à distance. Pour le rendre accessible, nous devons faire de petites modifications dans 3 fichiers. Alors commençons. Ouvrons keycloak.conf fichier sous / etc / keycloak / répertoire et ajoutons une ligne à la fin comme indiqué dans la capture ci-dessous puis enregistrer et quitter :

\$ sudo nano /etc/keycloak/keycloak.conf

```

GNU nano 2.9.3
# The configuration you want to run
WILDFLY_CONFIG=standalone.xml

# The mode you want to run
WILDFLY_MODE=standalone

# The address to bind to
WILDFLY_BIND=0.0.0.0

# The address console to bind to
WILDFLY_MANAGEMENT_CONSOLE_BIND=0.0.0.0

Get Help  Write Out  Where Is
Exit      Read File  Replace

```

Ouvrons maintenant launch.sh dans le répertoire /opt/keycloak/bin/ et modifiez son contenu comme indiqué ci-dessous et enregistrer :

\$ sudo nano /opt/keycloak/bin/launch.sh

```
GNU nano 2.9.3
#!/bin/bash
if [ "x$WILDFLY_HOME" = "x" ]; then
    WILDFLY_HOME="/opt/keycloak"
fi
if [[ "$1" == "domain" ]]; then
    $WILDFLY_HOME/bin/domain.sh -c $2 -b $3 -bmanagement $4
else
    $WILDFLY_HOME/bin/standalone.sh -c $2 -b $3 -bmanagement $4
fi

^G Get Help      ^O Write Out    ^W Where Is     ^K Cut Text
^X Exit          ^R Read File    ^\ Replace      ^U Uncut Text
```

Enfin, ouvrons le fichier de définition de service système de Keycloak (keycloak.service) sous /etc/systemd/system/ et apportez les modifications comme indiqué ci-dessous puis enregistrer et quitter :

\$ sudo nano /etc/systemd/system/keycloak.service

```
GNU nano 2.9.3 /etc/systemd/system/keycloak.service
[Unit]
Description=The Keycloak Server
After=syslog.target network.target
Before=httpd.service

[Service]
Environment=LAUNCH_JBOSS_IN_BACKGROUND=1
EnvironmentFile=-/etc/keycloak/keycloak.conf
User=keycloak
Group=keycloak
LimitNOFILE=102642
PIDFile=/var/run/keycloak/keycloak.pid
ExecStart=/opt/keycloak/bin/launch.sh $WILDFLY_MODE $WILDFLY_CONFIG $WILDFLY_BIND $WILDFLY_MANAGEMENT_CONSOLE_BIND
StandardOutput=null

[Install]
WantedBy=multi-user.target

^G Get Help      ^O Write Out    ^W Where Is     ^K Cut Text    ^J Justify
^X Exit          ^R Read File    ^\ Replace      ^U Uncut Text  ^T To Spell
^_ Cur Pos      ^C Go To Line   ^-U Undo       ^-E Redo
```

Puisque nous avons modifié le fichier de l'unité de service, nous devons informer le gestionnaire de systemd

\$ sudo systemctl daemon-reload

Redémarrons maintenant le service keycloak

\$ sudo systemctl reload keycloak

Une fois redémarré, accédez à la console de gestion Keycloak à l'adresse : `http://<instance-public-ip>:9990`



Nous pouvons accéder avec succès à la console de gestion, mais comme indiqué ci-dessus, nous avons besoin d'un utilisateur de gestion pour se connecter.

Nous pouvons utiliser le script `add-user.sh`, fourni avec la distribution du serveur Keycloak, pour créer un utilisateur de gestion. Exécutez le script avec la commande ci-dessous :

\$ sudo /opt/keycloak/bin/add-user.sh

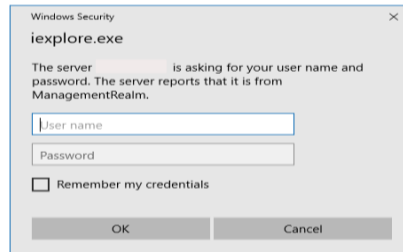
```
What type of user do you wish to add?
a) Management User (mgmt-users.properties)
b) Application User (application-users.properties)
(a): a

Enter the details of the new user to add.
Using realm 'ManagementRealm' as discovered from the existing property files.
Username : shani
Password recommendations are listed below. To modify these restrictions edit the add-user.properties configuration file.
- The password should be different from the username
- The password should not be one of the following restricted values {root, admin, administrator}
- The password should contain at least 8 characters, 1 alphabetic character(s), 1 digit(s), 1 non-alphanumeric symbol(s)
Password :
Re-enter Password :
What groups do you want this user to belong to? (Please enter a comma separated list, or leave blank for none)[ ]:
About to add user 'shani' for realm 'ManagementRealm'
Is this correct yes/no? yes
Added user 'shani' to file '/opt/keycloak/standalone/configuration/mgmt-users.properties'
Added user 'shani' to file '/opt/keycloak/domain/configuration/mgmt-users.properties'
Added user 'shani' with groups to file '/opt/keycloak/standalone/configuration/mgmt-groups.properties'
Added user 'shani' with groups to file '/opt/keycloak/domain/configuration/mgmt-groups.properties'
Is this new user going to be used for one AS process to connect to another AS process?
e.g. for a slave host controller connecting to the master or for a Remoting connection for server to server EJB calls.
yes/no? yes
To represent the user add the following to the server-identities definition <secret value="c2hhbmRlc2h0bW0JA==" />
```

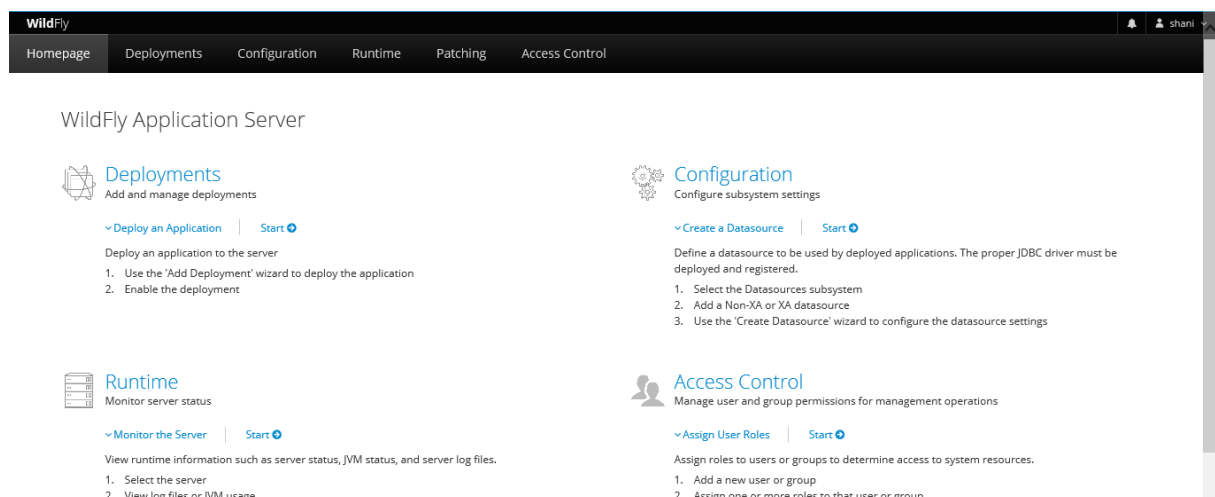
Une fois que l'on y est invité, sélectionnons pour ajouter un utilisateur de gestion et fournissons le nom d'utilisateur et le mot de passe souhaités. Dans la dernière invite, pour activer l'accès à distance pour cet utilisateur, écrivons `yes` ou `y`. Après avoir fourni les informations requises, le script vérifiera la création de l'utilisateur comme indiqué ci-dessus. Nous devons redémarrer le serveur Keycloak pour que notre utilisateur soit récupéré lors du démarrage.

\$ sudo systemctl restart keycloak

Maintenant, si nous accédons à nouveau à la console de gestion, elle demandera une autorisation de base HTTP. Fournissons les informations d'identification de l'utilisateur de gestion que nous avons créées ci-dessus et cliquez sur « OK » :



Une fois connecté avec succès, nous atterrissons dans la console de gestion



Nous venons de faire la configuration de base du serveur Keycloak et d'activer /configurer l'accès à distance à la console d'administration et de gestion. Ceci conclut notre tutoriel.

RÉFÉRENCES

❖ Liens web

- <https://www.saasworthy.com/product-alternative/5998/keycloak/>
- https://www.reddit.com/r/selfhosted/comments/fxotbi/experiences_with_keycloak_alternatives/
- https://www.keycloak.org/docs/latest/getting_started/index.html#securing-a-sample-application
- <https://medium.com/@hasnat.saeed/setup-keycloak-server-on-ubuntu-18-04-ed8c7c79a2d9>
- <https://www.keycloak.org/getting-started/getting-started-zip>
- https://www.keycloak.org/docs/latest/release_notes/
- <https://dzone.com/articles/easily-secure-your-spring-boot-applications-with-keycloak>
- <https://www.keycloak.org/downloads>
- <https://en.wikipedia.org/wiki/Keycloak>
- <https://medium.com/theoptimaltechnologist/what-is-keycloak-how-to-use-it-an-example-with-nodejs-part-1-25434e963fed>
- <https://www.g2.com/products/keycloak/competitors/alternatives>
- <https://www.g2.com/products/microsoft-azure-active-directory/reviews>

❖ Liens YouTube

- <https://youtu.be/FdYAdJkwynA>
- <https://youtu.be/vpgRTPFDHAW>
- <https://youtu.be/T0D4GF5YryI>

❖ Google

❖ Wikipédia

❖ Microsoft BING

❖ Duckduckgo