JENKINS: Bases

Table of Contents

Présentation

Jenkins est un serveur d'intégration et de déploiement continu que l'on peut installer et lancer sur un serveur local. Le projet Jenkins a débuté en 2004 et aujourd'hui c'est une solution adoptée par les entreprises qui souhaitent posséder leur propre infrastructure de CI.

Avantages

Jenkins est écrit en Java et est compatible avec les principaux systèmes d'exploitation. Les builds peuvent tourner sous environnement Linux, BSD, MacOS ou Windows.

Jenkins est totalement libre d'utilisation et open source. Jenkins supporte les plugins et dispose d'une

bibliothèque plugins très fournie pour l'ajout de fonctionnalités au serveur de CI.

Les plugins Pipeline permettent de faire tourner ses builds sur n'importe quel système d'exploitation, y compris Windows ou Mac OS, car Jenkins est écrit en Java.

Installation

Installer Jenkins sur Redhat, Fedora ou CentOS

Il existe des paquets binaires natifs pour Redhat, Fedora et CentOS. Il faut d'abord configurer le référentiel comme il suit :

```
sudo wget -O /etc/yum.repos.d/jenkins.repo
http://jenkins-ci.org/redhat/jenkins.repo
sudo rpm --import   http://pkg.jenkins-ci.org/redhat/jenkins-ci.org.key
```

Sur une nouvelle installation, il faudra peut-être installer le JDK:

```
sudo yum install java-1.6.0-openjdk
```

Ensuite, installer le paquet :

```
sudo yum install jenkins
```

Ceci installera la dernière version de Jenkins dans le répertoire /usr/lib/jenkins . Le répertoire de travail par défaut de Jenkins sera /var/lib/jenkins .

Démarrer Jenkins en utilisant la commande de service :

```
sudo service jenkins start
```

Jenkins va maintenant être exécuté sur le port par défaut : 8080 (http://localhost:8080/).

Les paramètres de configuration de Jenkins sont placés dans le fichier /etc/sysconfig/jenkins . Cependant,on peut définir le port HTTP en utilisant le paramètre JENKINS_PORT, par exemple. Les principales options de configuration sont listées ici :

- **JENKINS_JAVA_CMD**: La version de Java que vous voulez utiliser pour exécuter Jenkins
- **JENKINS_JAVA_OPTIONS**: Les options de ligne de commande à passer à Java, tels que les options de mémoire
- **JENKINS_PORT**: Le port sur lequel Jenkins sera exécuté

Configuration proxy NGINX

JENKINS comme service unique

Décommenter **default server config** dans `/etc/nginx/nginx.conf` et créer le fichier `/etc/nginx/conf.d/domain.conf`

```
    location / {
        proxy_set_header        Host $host:$server_port;
        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://127.0.0.1:8080;
        proxy_read_timeout  90;

        proxy_redirect      http://127.0.0.1:8080
https://jenkins.yourdomainname.com;

        proxy_http_version 1.1;
        proxy_request_buffering off;
        add_header 'X-SSH-Endpoint' 'jenkins.yourdomainname.com:50022'
always;
    }
```

JENKINS dans une subdirectory

Editer le fichier `/etc/sysconfig/jenkins`' pour ajouter **prefix=/jenkins** à **JENKINS_ARGS**

```
JENKINS_ARGS="--prefix=/jenkins"
```

Décommenter **default server config** dans `/etc/nginx/nginx.conf` et créer le fichier `/etc/nginx/conf.d/domain.conf`

```
server {
        listen       80;
        listen       [::]:80;
        server_name  _;

        return 301 https://$host$request_uri;
}

server {
        listen       443 ssl http2;
        listen       [::]:443 ssl http2;
```

```
        server_name  domain.com;
        access_log   /var/log/nginx/domain.com.access.log;
        root         /var/www/domain.com/public_html;

        ssl_certificate         /path/to/tls_bundle.crt;
        ssl_certificate_key     /path/to/tls_bundle.key;
        ssl_session_timeout 1d;
        ssl_session_cache shared:SSL:50m;
        ssl_session_tickets off;

       ssl_protocols TLSv1 TLSv1.1 TLSv1.2;

       ssl_ciphers 'ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-
SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:ECDHE-
ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-
SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-
SHA256';
       ssl_prefer_server_ciphers on;

        # HSTS (ngx_http_headers_module is required) (15768000 seconds = 6
months)
        add_header Strict-Transport-Security max-age=15768000;

        # OCSP Stapling ---
        # fetch OCSP records from URL in ssl_certificate and cache them
        ssl_stapling on;
        ssl_stapling_verify on;

        resolver 8.8.8.8;

        # Note that regex takes precedence, so use of "^~" ensures earlier
evaluation
       location ^~ /jenkins/ {
              proxy_http_version 1.1;
              proxy_set_header Upgrade $http_upgrade;
              proxy_set_header Connection "upgrade";

              proxy_pass             http://10.0.0.100:8080/jenkins/;

              proxy_redirect http:// https://;

              sendfile off;

              proxy_set_header   Host                 $host:$server_port;
              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    https;
              proxy_max_temp_file_size 0;

              # this is the maximum upload size
```

```
                client_max_body_size       10m;
                client_body_buffer_size    128k;

                proxy_connect_timeout      90;
                proxy_send_timeout         90;
                proxy_read_timeout         90;

                proxy_buffer_size          4k;
                proxy_buffers              4 32k;
                proxy_busy_buffers_size    64k;
                proxy_temp_file_write_size 64k;
        }
}
```

*# Gestion des utilisateurs et identifiants

Jenkins permet d'identifier et de gérer les utilisateurs de plusieurs façons, depuis une la base de données intégrée pour les petites équipes jusqu'à l'intégration avec des annuaires d'entreprise, avec de nombreuses autres options entre les deux.

Gestion des utilisateurs dans la base de données intégrée à Jenkins

Le moyen le plus simple pour gérer des comptes utilisateurs dans Jenkins est d'utiliser la base de données interne de Jenkins. C'est une bonne option si on veut garder les choses simples, car peu de configuration est nécessaire.

Création des utilisateurs

Les utilisateurs qui ont besoin de se connecter au serveur Jenkins peuvent s'enregistrer et créer un compte par eux-mêmes, et, en fonction du modèle de sécurité choisi, un administrateur peut ensuite décider ce que ces utilisateurs sont autorisés à faire.

Liste des utilisateurs connus de Jenkins

Jenkins ajoute automatiquement tout utilisateur de gestionnaire de sources à cette base de données dès qu'un changement est effectué dans le code source surveillé par Jenkins. Ces noms d'utilisateurs sont utilisés principalement pour enregistrer le responsable de chaque tâche de build. On peut voir la liste des utilisateurs connus en cliquant sur l'entrée de menu Personnes .

Si on clique sur un utilisateur de cette liste, Jenkins affiche différentes informations à propos de cet utilisateur, incluant son nom complet et les tâches de build auxquelles il a contribué. De là, on peut aussi modifier ou compléter les détails à propos de cet utilisateur, comme son mot de passe ou son adresse email. Afficher les builds auxquels un utilisateur participe

Activation des utilisateurs

Un utilisateur apparaissant sur cette liste ne peut pas nécessairement se connecter à Jenkins. Pour pouvoir se connecter, l'utilisateur doit avoir un mot de passe configuré. Il y a essentiellement deux façons de faire cela. Si on a configuré l'option “Autoriser les utilisateurs à s'enregistrer”, les utilisateurs peuvent simplement se connecter avec leur nom d'utilisateur SCM et fournir leur adresse email et leur mot de passe sans confirmation. Autrement, on peut activer un utilisateur en cliquant sur l'option de menu Configurer dans l'écran de détails utilisateur, et fournir une adresse email et un mot de passe.

Gestion des identifiants des serveurs dans la base interne

Le plugin **credentials** permet de stocker des identifiants dans Jenkins.

Le plug-in des informations d'identification fournit une API normalisée permettant à d'autres plug-in de stocker et de récupérer différents types d'informations d'identification.

La visibilité de ces identifiants peut être limitée à certains projets à l'intérieur de jenkins, par exemple les identifiants des commutateurs définis dans le projet réseau peuvent être vus depuis ce projet, mais pas de la racine ni d'un autre projet.

Construire un build avec paramètre

La fonction **“Construire un projet free-style”** sert à builder (construire) un projet. On peut y intégrer tous les outils de gestion de version avec tous les systèmes de build. Il est même possible d'utiliser Jenkins pour tout autre chose qu'un build logiciel.

Description

Assign Redmine project

- **Redmine website**: adresse url du serveur redmine
- **Redmine project identifier**: Redmine project identifier

Gestion des paramètres

L'activation de l'option **Ce build a des paramètres** permet de passer de manière fixe ou interactive des paramètres de différents types lors du build:

- Extended Choice Parameter
- Git Parameter
- JIRA Issue Parameter

- JIRA Release Version Parameter
- List Subversion tags (and more)
- Non-Stored Password Parameter
- Paramètre “Mot de passe”
- Paramètre String
- Paramètre booléen
- Paramètre choix

Exemple de paramètre texte

Par exemple on veut utiliser un script pour passer les commandes “screen-length disable” (mode wrap) et “dis cur” (pour lister la configuration) sur des commutateurs

- **Name**: Nom du paramètre
- **Default Value:** liste simplement les commandes à passer sur les commutateurs
- **Description**: Description

Exemple de paramètre "Extended Choice Parameter"

Permet de définir des paramètres multi critères de type :

- **Basic Parameter Types**
- **Multi-level Parameter Types**
- **JSON Parameter Type**

Par exemple pour passer les commandes sur les commutateurs on a besoin des indentifiants/mot de passe de chacun d'eux.

Pour des raisons de sécurité les identifiants et mot de passe des commutateurs sont stockés dans la base interne, il faut donc les récupérer pour pouvoir les passer en paramètre au script.

Des scripts groovy permettent de construire un menu déroulant de ce identifiants en utilisant **Basic Parameter Types**:

- **Name**: Nom du paramètre
- les paramètres **Parameter type**, **Number of Visible Items**,**Delimiter** et, **Quote Value** servent à définir le type (menu déroulant = Parameter type = Multi Lines), le contenu (une liste d'items délimités par une virgule = Delimiter = ,) et l'apparence du menu déroulant (une fenètre limitée à 5 idems = Number of Visible Items=5)
- **Source for Value** définit le contenu de la liste d'items(Value, Property File, Groovy Script ou Groovy Script File)
- **Source for Default Value** définit les valeurs par défaut de la liste d'item (Default Value, Default Property File, Default Groovy Script ou Default Groovy Script File)
- **Source for Value Description** définit les libellés de laliste d'items (Description, Description Property File, Description Groovy Script, Description Groovy Script File)

Gestion de code source

Jenkins est un serveur d'intégration continue, ce qui signifie qu'il doit extraire le code source d'un référentiel de code source et créer du code.

Jenkins supporte nativement certains outils de gestion de code source tels que Subversion et CVS. Pour utiliser d'autres outils tel Git, il suffit d'installer le plug-in Jenkins.

Les options de contrôle de gestion sont les suivantes:

- **Aucune**
- **Git**
- **Mercurial**
- **SCLM**
- **Subversion**

Ce qui déclenche le build

- Déclencher les builds à distance (Par exemple, à partir de scripts)
- Build when a change is pushed to Gogs
- Construire après le build sur d'autres projets
- Construire périodiquement
- GitHub hook trigger for GITScm polling
- Scrutation de l'outil de gestion de version

Environnements de Build

- **Mask passwords and regexes**: permet de masquer les paramètres de mot de passe, ou tout autre type de paramètres de construction ou expressions rationnelles sélectionnés pour le masquage dans l'écran de configuration principal de Hudson / Jenkins (Manage Hudson> Configure System).
- **Use secret text(s) or file(s)**
- **Provide Configuration files**
- **Send files or execute commands over SSH before the build starts**: Pour ennvoyer des fichiers ou exécuter des commandes sur SSH avant le début de la construction
- **Send files or execute commands over SSH after the build runs**: Pour envoyer des fichiers ou exécuter des commandes sur SSH après l'exécution de la génération
- **Create a formatted version number**: Pour créer un numéro de version formaté selon un masque définit
- **Execute shell script on remote host using ssh**: Exécuter un script shell sur un hôte distant à l'aide de ssh
- **Generate Release Notes**
- **Set Build Name**
- **With Ant**

Build

On peut exécuter un script bash en utilisant les paramètres décrits plus haut ..

```
#USERNAME=${CREDENTIALS%:*}
#PASSWORD=${CREDENTIALS#*:}
#sh $WORKSPACE/netshow $USERNAME $PASSWORD $commandes
#credits=$(echo $hosts | sed -e 's/[{}]/''/g'| sed -e 's/\"/''/g'| sed -e
's/:/=/g' | sed -e 's/,/;/g')
#eval "$credits"
set +x
IFS=','
for line in $hosts
    do
    ip_address=$(echo $line | awk -F"|"  '{print $1}')
    user_id=$(echo $line | awk -F"|"  '{print $2}')
    user_password=$(echo $line | awk -F"|"  '{print $3}')
    sh $WORKSPACE/netshow $user_id $user_password $ip_address
done
```

ou passer des commandes

```
java -jar /jenkins-cli.jar -s http://user:password@xx.xx.xxx.xx:8383/ get-
job reseau/netshow > /tmp/netshow.xml
```

Actions à la suite du build

Par exemple on peut Publier le résultat de la construction en tant que git-notes en utilisant le plugin **Git Publisher**