SCIENCES SORBONNE UNIVERSITÉ

Travaux pratiques préparatoire : Zookeeper

Jonathan Lejeune

Objectif

Ce sujet de travaux pratiques préparatoire est à faire au préalable de la séance correspondante. Il vous guidera dans l'installation d'un environnement de travail pour la plate-forme Zookeeper sur UNIX.

Introduction

Zookeeper est une plate-forme distribuée écrite en Java permettant la configuration et la coordination de processus distants. Son fonctionnement se base sur un entrepôt de données clé-valeur, persistant et répliqué sur un groupes de machines serveurs qui doivent former un quorum.

Les différentes clés sont organisées de manière arborescente. Un nœud de cette arborescence est appelé **znode**. Un znode contient une valeur (la donnée du znode) et peut avoir des znode fils. Sur le même format que le nommage des fichiers sous UNIX, un znode est identifié de manière unique par un chemin absolu à partir d'un znode racine désigné par `/`. Ainsi un znode se désigne par l'ensemble des noms de ses ancêtres séparés par des `/` et de son nom relatif à son parent direct.

Zookeeper proposant un espace de stockage commun, il peut faire office de service de nommage (à l'instar de DNS, LDAP, etc.) ou bien d'entrepôt de données partagé pour stocker par exemple des méta-données ou des données de configuration des systèmes clients. Il offre également à ses clients un mécanisme de notification asynchrone sur l'état des znodes. Ainsi il est possible de construire des services plus élaborés comme une file de message, des mécanismes élection de leader ou de synchronisation entre des processus distants.

Zookeeper dispose de 2 modes de fonctionnement :

- **le mode standalone** : utilisé pour les phases de développement, il est possible de limiter le groupe de serveurs à un seul serveur : la machine locale.
- **le mode multi-serveur** : utilisé en production il permet de déployer Zookeeper sur plusieurs machines distinctes. Il est cependant possible d'émuler ce mode de fonctionnement sur une seule machine en démarrant plusieurs processus qui utiliseront le loopback de la machine locale. Ceci permet de tester le comportement de la plate-forme dans un contexte distribué lors des phases de développement.

Ce TME préparatoire vous permet de tester les deux modes sur votre machine locale.

Exercice 1 – Installation de Zookeeper

Question 1

Téléchargez la dernière version stable de la plateforme Zookeeper à l'URL suivante (privilégiez l'archive qui se termine par -bin.tar.gz) :

`https://dlcdn.apache.org/zookeeper/stable/`

Question 2

Extrayez le contenu de zookeeper-3.x.x.tar.gz dans votre home. Le répertoire produit sera le répertoire d'installation de zookeeper

Question 3

Ajoutez ensuite les lignes suivantes dans votre *.bashrc* :

```
export MY_ZOOKEEPER_HOME=<votre_repertoire_installation_zookeeper>
export PATH=$PATH:$MY_ZOOKEEPER_HOME/bin
```

Pour prendre en compte ces modifications dans votre terminal ouvert tapez :

```
source ~/.bashrc
```

Exercice 2 – Déploiement en mode standalone

Pour utiliser Zookeeper en mode standalone, il est nécessaire de créer le fichier de configuration *zoo.cfg* dans le répertoire *conf* de votre dossier d'installation. Ce fichier doit contenir au minimum trois paramètres :

- `tickTime` : indique en millisecondes l'unité de base de temps pour Zookeeper. Ainsi tout paramètre de temps dans Zookeeper que l'on verra par la suite (heartbeat, timeout, ...) ne s'exprime uniquement qu'en ticks.
- `dataDir` : indique le répertoire sur le système de fichier local du serveur où écrire les données des znodes
- `clientPort` : indique le port sur lequel les clients Zookeeper peuvent se connecter pour utiliser le service

Question 1

Créer le fichier *zoo.cfg* dans le répertoire *conf*. Dans notre cas, `tickTime` sera égal à 2000, `dataDir` sera le répertoire */tmp/zookeeper* et `clientPort` sera 2181. Le contenu du fichier doit donc être :

```
tickTime=2000
dataDir=/tmp/zookeeper
clientPort=2181
```

Question 2

Pour démarrer le serveur tapez dans votre terminal

```
zkServer.sh start
```

Ceci a pour effet de démarrer en tant que tâche de fond un serveur Zookeeper sur votre machine locale. Le fichier de configuration par défaut étant *conf/zoo.cfg* il est inutile de le spécifier dans la commande.

Question 3

En tapant `netstat -na | grep tcp | grep LISTEN` vous pourrez remarquer que le port 2181 est bien utilisé.

Question 4

En tapant la commande `zkServer.sh status` vous pourrez vous assurez que le mode d'exécution est bien le mode standalone.

Question 5

Pour arrêter le serveur, tapez dans votre terminal :

```
zkServer.sh stop
```

Question 6

Démarrez le serveur en premier plan (le terminal ne rend pas la main) avec la commande

```
zkServer.sh start-foreground
```

Question 7

Dans un autre terminal, tapez :

```
zkCli.sh -server localhost:2181
```

Ceci a pour effet de se connecter sur le serveur et de démarrer une session client Zookeeper via un terminal pour pouvoir manipuler les znodes.

Question 8

Dans le terminal client tapez la commande `help` pour connaître toutes les commandes disponibles. on peut trouver entre autre :

- `ls` : pour connaître la liste des fils d'un znode passé en argument. Vous pouvez essayé `ls /`.
- `stat` : affiche des méta-données sur le znode.
- `ls2` : fusion de la commande `ls` et de la commande `stat`
- `set` : permet d'affecter une valeur à un znode existant.
- `get` : affiche la valeur (+ info de `stat`) hébergée sur le znode
- `create` : permet de créer un nouveau znode. Vous pouvez créer par exemple le fils `toto` de `/` ayant la valeur "bonjour" avec la commande `create /toto bonjour`
- `quit` : permet de se déconnecter du serveur et d'arrêter le client

Question 9

Pour arrêter le serveur en mode foreground, faite un ctrl-C sur son terminal.

Exercice 3 – Déploiement en mode multi-serveur

Ce mode permet de déployer plusieurs serveur sur un ensemble de machines afin de pouvoir répliquer les données. Dans cet ensemble un nœud fait office de **leader**. Ce leader est désigné par un protocole d'élection au moment du démarrage des différents serveurs ou lorsque le leader tombe en panne. Les autres nœuds sont appelés des **followers**.

Dans un environnement réellement distribué, il faut que Zookeeper soit installé sur chaque machine du cluster et que leur fichier de configuration contiennent en plus des paramètres de base vus dans l'exercice précédent, les paramètres suivants :

- `initLimit` : le temps maximum en nombre de `tickTime` pour qu'un follower puisse se connecter et se synchroniser sur le leader. Le paramétrage de cette valeur dépend de la quantité de données gérée par Zookeeper. Plus cette quantité est grande, plus la valeur doit être grande.
- `syncLimit` : le temps maximum en nombre de `tickTime` pour qu'un follower puisse synchroniser son arborescence de znodes avec le reste du système
- une séries de paramètres pour lister l'ensemble des serveurs ainsi que leurs ports d'écoute. Le format de ces paramètres est `server.x=hostname:port1:port2` où :
  - ◇ `x` est un entier compris entre 1 et 255 attribué arbitrairement et fait office d'identifiant de serveur au sein du groupe (=le *sid*)

- ◇ `hostname` désigne le nom DNS ou l'adresse IP de la machine hébergeant le serveur `x`
- ◇ `port1` désigne le port d'écoute pour le protocole pour la synchronisation des données entre les différents serveurs
- ◇ `port2` désigne le port d'écoute pour le protocole d'élection de leader

Enfin pour que chaque serveur connaisse son identifiant `x` au sein du groupe de serveurs, il faut créer à la racine de son répertoire `dataDir` un fichier `myid` dont le contenu se résume à `x`.

Par la suite nous allons déployer Zookeeper en mode multi-serveur mais sur la même machine locale. Ceci nécessite de paramétrer correctement les ports d'écoute et les répertoire de données afin d'éviter les conflits.

Question 1

Nous allons déployer 3 serveurs. Il faut donc avoir 3 fichiers de configurations différents (*zooNode1.cfg*, *zooNode2.cfg* et *zooNode3.cfg*) dans le répertoire *conf*. Dans chaque fichier les dossiers de données et les ports d'écoute (`clientPort`, `port1` et `port2`) doivent être différents. Ainsi

- le contenu de *zooNode1.cfg* est :

```
tickTime=2000
initLimit=10
syncLimit=2
dataDir=/tmp/zookeeper1
clientPort=2181
server.1=localhost:2888:3888
server.2=localhost:2889:3889
server.3=localhost:2890:3890
```

- le contenu de *zooNode2.cfg* est :

```
tickTime=2000
initLimit=10
syncLimit=5
dataDir=/tmp/zookeeper2
clientPort=2182
server.1=localhost:2888:3888
server.2=localhost:2889:3889
server.3=localhost:2890:3890
```

- le contenu de *zooNode3.cfg* est :

```
tickTime=2000
initLimit=10
syncLimit=5
dataDir=/tmp/zookeeper3
clientPort=2183
server.1=localhost:2888:3888
server.2=localhost:2889:3889
server.3=localhost:2890:3890
```

Question 2

Créez à la racine de chaque répertoire de données un fichier *myid* qui contiendra l'identifiant du serveur responsable de ce répertoire. Dans notre cas le fichier */tmp/zookeeper1/myid* contiendra 1, le fichier */tmp/zookeeper2/myid* contiendra 2, etc.

Question 3

Pour lancer chaque serveurs il faut taper la même commande que l'exercice précédent (`zkServer.sh start`) mais en précisant en deuxième argument, le nom du fichier de configuration correspondant. Ainsi dans notre exemple il faudra lancer les commandes suivantes :

```
zkServer.sh start zooNode1.cfg
zkServer.sh start zooNode2.cfg
zkServer.sh start zooNode3.cfg
```

Question 4

Taper pour chaque serveur la commande `zkServer.sh status` en précisant toujours en deuxième argument le fichier de configuration correspondant. Identifier le nœud leader.

Question 5

Arrêter le nœud leader avec la commande `zkServer.sh stop <fichier_conf_leader>` et vérifier qu'un autre leader a été élu.

Question 6

Relancer l'ancien leader. Remarquer qu'il prend désormais un rôle de follower.

Question 7

(facultatif) Vous pouvez refaire la même manipulation en démarrant les serveur en mode foreground dans 3 terminaux différents afin de visualiser les interactions entre chaque serveur via les affichages.

Exercice 4 – Configuration d'Eclipse

Afin d'utiliser *Eclipse* pour éditer et compiler des programmes clients Zookeeper, il faut déclarer une nouvelle librairie permettant de déclarer dans le build path d'un projet Eclipse (= le classpath du point de vue de la JVM) les différents fichiers jars de Zookeeper.

Question 1

Pour ce faire :

- Dans la barre de menu de Eclipse cliquez sur *Window*
- Cliquez sur *Preferences*
- Déroulez l'onglet *Java* sur le panneau de gauche
- Déroulez l'onglet *Build Path*
- Cliquez sur le menu *User Libraries*
- Cliquez sur le bouton *New...*
- Nommez la librairie `zookeeper-3.x.y` (où `x` et `y` sont a remplacer par la version que vous avez téléchargée)
- Sélectionnez la nouvelle librairie créée et cliquez sur le bouton *Add External JARs...*
- Ajoutez l'ensemble des *Jar* présents à la racine du dossiers *lib* du dossier d'installation
- cliquez sur *Apply and Close* pour valider.

Question 2

Créez un projet java.

Question 3

Ajouter la librairie créée précédemment afin de programmer avec l'API Java de Zookeeper. Pour ce faire :

- Sélectionnez *Build Path* dans le menu contextuel (clic droit sur le dossier) du projet puis *Add Libraries...*
- Sélectionnez *User Library* puis *Next >*
- Cochez la librairie puis cliquez sur *Finish.*
- Vérifiez que la librairie a bien été ajoutée au projet dans le Package Explorer d'Eclipse.