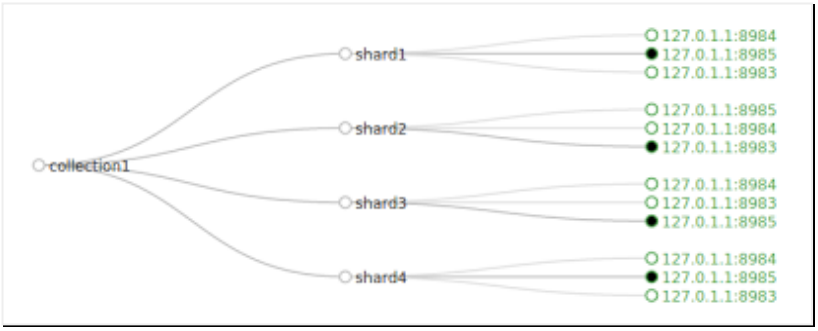
csrelatedstuff

2016. február 13., szombat

Installing/configuring SolrCloud (5.4.1) and ZooKeeper (3.4.6)

The purpose of this entry is to give a complete tutorial to setup SolrCloud with an external ZooKeeper with a single collection. The post is based on Andrew Rowland's Install Solr 5 and Zookeeper in a production environment tutorial, but some other related problems are discussed here too, for example how to set ZooKeeper to support large files, how to use external libraries in Solr, etc.

The setup



The setup is as seen above: we are given a collection of some data, and we want to create 4 shards with 3 replications. Of course, these numbers (i.e. the number of shards and replicas) can be modified arbitrarily. However, we will use these in the description.
We assume we have 3 servers we want to install Solr+ZooKeeper on. Thus on each server a Solr service and a ZooKeeper server is to be installed. ZooKeeper will store the configuration files and will realize the communication between the separate instances. The cluster will function until the majority (>50%) of the ZooKeeper instances function, but one can still do queries, even if the cloud is not functioning (this was not tested yet, only red in the documentation).

ZooKeeper install and configuration

To install ZooKeeper, one just has to download the compressed package and decompress it. The configuration is done through cfg files, usually stored in the conf subdirectory. Each server will have a unique id, a number between 1 and 255. Likewise, each server has to specify its own directory where the files and configuration stuff are stored. In the specified directory a file called myid must be created with the chosen unique id residing in it.
We will create 3 servers (now on localhost, but in case of physically different servers one can easily modify the corresponding IPs) identified by `1`, `2` and `3`, and having communication ports towards Solr `2181`, `2182` and `2183`. They also have to communicate between each other, and these communication ports will be `2888:3888`, `2889:3889`, `2890:3890`, where the first one is used by the followers to connect to the leader and the second one is for leader election (see the ZooKeeper documentation for more details).
The directories we use will be: `/var/lib/zookeeperdata/1`, `/var/lib/zookeeperdata/2` and `/var/lib/zookeeperdata/3`.
Without commenting on the meaning of the other parameters we list here the contents of the configuration files:

```
tickTime=2000
dataDir=/var/lib/zookeeperdata/1
dataLogDir=/var/lib/zookeeperdata/1
clientPort=2181
initLimit=5
syncLimit=10
minSessionTimeout=4000
maxSessionTimeout=60000
server.1=localhost:2888:3888
server.2=localhost:2889:3889
server.3=localhost:2890:3890
```

zoo.cfg hosted with ❤ by **GitHub** **view raw**

```
tickTime=2000
dataDir=/var/lib/zookeeperdata/2
dataLogDir=/var/lib/zookeeperdata/2
clientPort=2182
initLimit=5
syncLimit=10
minSessionTimeout=4000
maxSessionTimeout=60000
server.1=localhost:2888:3888
server.2=localhost:2889:3889
server.3=localhost:2890:3890
```

zoo2.cfg hosted with ❤ by **GitHub** **view raw**

```
tickTime=2000
dataDir=/var/lib/zookeeperdata/3
dataLogDir=/var/lib/zookeeperdata/3
clientPort=2183
initLimit=5
syncLimit=10
minSessionTimeout=4000
maxSessionTimeout=60000
server.1=localhost:2888:3888
server.2=localhost:2889:3889
server.3=localhost:2890:3890
```

zoo3.cfg hosted with ❤ by **GitHub** **view raw**

In order to be able to store and work with larger files one has to set the `jute.maxbuffer` parameter (in all clients and servers) to the desired maximum value, which by default is set to 1MB. This must be done usually, since the synonym files tend to be larger than 1MB. Thus, we set parameter in the following locations:

1. in `zkCli.sh` script from the bin subdirectory of ZooKeeper: add it after the (last) `"$JAVA"` command, that is `"$JAVA" "-Djute.maxbuffer=20000000" ...`

```
"$JAVA" "-Djute.maxbuffer=20000000" "-Dzookeeper.log.dir=${ZOO_LOG_DIR}" "-Dzo
     -cp "$CLASSPATH" $CLIENT_JVMFLAGS $JVMFLAGS \
     org.apache.zookeeper.ZooKeeperMain "$@"
```

zkCli.sh hosted with ❤ by **GitHub** **view raw**

2. in `zkServer.sh` at the start option: `nohup "$JAVA" "-Djute.maxbuffer=20000000" ...`

```
    nohup "$JAVA" "-Djute.maxbuffer=20000000" "-Dzookeeper.log.dir=${ZOO_LOG_D
    -cp "$CLASSPATH" $JVMFLAGS $ZOOMAIN "$ZOOCFG" > "$_ZOO_DAEMON_OUT" 2>&1 <
```

zKServer.sh hosted with ❤ by **GitHub** **view raw**

3. it has to be in Solr too (in 2 places), but we will specify exactly how to parametrize Solr in the next section.

Starting ZooKeeper servers

```
zkServer.sh start zoo.cfg
zkServer.sh start zoo2.cfg
zkServer.sh start zoo3.cfg
```

start_zookeepers.sh hosted with ❤ by **GitHub** **view raw**

Stopping ZooKeeper servers

```
zkServer.sh stop zoo.cfg
zkServer.sh stop zoo2.cfg
zkServer.sh stop zoo3.cfg
```

stop_zookeepers.sh hosted with ❤ by **GitHub** **view raw**

zkCli.sh

With this script you can connect to specific ZooKeeper server and execute some commands, such as list the files stored in ZK, put/get files, get information regarding the leaders, etc. The Solr configurations are stored in the configs directory, while the leaders information are stored in the /collections folder.

Solr install and configuration

We can simply install Solr as service by downloading it, extracting the `install_solr_service.sh` script from its bin folder, and executing the following commands (each on a given server):

```
./install_solr_service.sh solr-5.4.1.zip -s solr -p 8983
./install_solr_service.sh solr-5.4.1.zip -s solr2 -p 8984
./install_solr_service.sh solr-5.4.1.zip -s solr3 -p 8985
```

install_solr.sh hosted with ❤ by **GitHub** **view raw**

Thus we install Solr as service on each server under the names `solr`, `solr2` and `solr3` on ports `8983`, `8984`, `8985`, respectively. By default, Solr will be installed to `/opt/solr-5.4.1` with symbolic links to `/opt/solr`, `/opt/solr2` and `/opt/solr3`. For storing Solr data directories `/var/solr`, `/var/solr2` and `/var/solr3` are used.
Configuring Solr in done through the `solr.in.sh` (`solr2.in.sh`, `solr3.in.sh`) script, which in the newer Solr versions resides in `/etc/default`. Here we need to set 3 things:

1. set the addresses of the ZooKeeper servers (ensemble):

```
ZK_HOST="localhost:2181,localhost:2182,localhost:2183"
```

solr.in.sh1 hosted with ❤ by **GitHub** **view raw**

2. set the `jute.maxbuffer` parameter:

```
SOLR_OPTS="$SOLR_OPTS -Djute.maxbuffer=20000000"
```

solr.in.sh2 hosted with ❤ by **GitHub** **view raw**

3. set SolrCloud mode (this is quite optional, only the 1. step is mandatory when using SolrCloud):

```
SOLR_MODE="solrcloud"
```

solr.in.sh3 hosted with ❤ by **GitHub** **view raw**

The `jute.maxbuffer` parameter has to be set in one more location: in Solr's `zkcli.sh` script, different from `zkCli.sh` of ZooKeeper, residing in the `server/scripts/could-scripts` subdirectory of Solr:

```
PATH=$JAVA_HOME/bin:$PATH $JVM -Djute.maxbuffer=20000000 -Dlog4j.configuration=file
```

zkcli.sh hosted with ❤ by **GitHub** **view raw**

Now lets start first the ZooKeeper servers (as shown above) and the Solr instances on every server:

```
service solr start
service solr2 start
service solr3 start
```

start_solrs.sh hosted with ❤ by **GitHub** **view raw**

The next step is to create a collection (`collection1`) using an existing configuration (this can be done in other ways too, but in my opinion, this is the most straightforward way):

```
solr create_collection -c collection1 -d <path_to_conf_folder> -n collection1 -shar
```

create_collection.sh hosted with ❤ by **GitHub** **view raw**

Now in Solr's Cloud view (`http://localhost:8983/solr/#/~cloud`) must appear the cluster graphs shown at the beginning of this tutorial.

Using "external" libraries

If using external libraries (e.g. using `ICUFoldingFilterFactory` and `ICUTokenizerFactory` in `schema.xml`), in solrconfig.xml we have to specify where the corresponding jars can be found. We can accomplish this by simply using the `<lib ... />` tag in `solrconfig.xml`:

```
    <lib dir="/opt/solr-5.4.1/contrib/analysis-extras/lib" regex=".*\.jar"/>
```

solrconfig.xml hosted with ❤ by **GitHub** **view raw**

Testing the system

To test the system one has to upload some data to the collection:

```
curl "http://localhost:8983/solr/collection1/update" -H "Content-Type: text/xml" --
```

upload_data.sh hosted with ❤ by **GitHub** **view raw**

where data.xml contains some data using the given schema. By default, all the results from the existing shards will be combined and returned, but if only some shards we are interested in, one can specify this using the shards parameter:

```
http://localhost:8983/solr/collection1/select?q=*:*
http://localhost:8983/solr/collection1/select?q=*:*&shards=shard1
http://localhost:8983/solr/collection1/select?q=*:*&shards=shard1,shard2
```

query_solr.sh hosted with ❤ by **GitHub** **view raw**

(Mysterious?) Errors and warnings

Sometimes SolrCloud will return you some errors and warnings and may show unexpected behaviour. In these situations (usually when updating the collection) the Solr server (service) restarts solved the problem.

Bejegyezte: miafranc dátum: 11:40

G+1 Ajánlás a Google-on

Nincsenek megjegyzések:

Megjegyzés küldése

Írj megjegyzést...

Megjegyzés írása mint: miafranc (Goo

Kilépés

Közzététel Előnézet Értesítést kérek

Főoldal

Feliratkozás: Megjegyzések küldése (Atom)

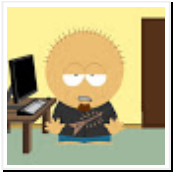
Blogarchívum

▼ 2016 (1)

▼ február (1)

Installing/configuring SolrCloud (5.4.1) and ZooKe...

Magamról



miafranc

Teljes profil megtekintése