The installation manual of BreedingEIS

The system supports two deployment methods: 1) a method that utilizes a Docker container, which does not require secondary development and code adjustment but only needs to deploy an available environment; 2) a method that installs components step-by-step and start services. This method is applicable to computer staff with a certain development foundation, which needs to conduct secondary development through the code and install, deploy and test the modified program.

1. Utilize the Docker container

The system supports deployment based on the docker container. Please find all the deployment files in the compressed package named “docker. Zip”. The compressed package can be downloaded from Github (https://github.com/qikaijie/BreedingEIS-M) or downloaded from the BreedingEIS web client (www.nnyshj.com).

The whole system involves mysql5.7, nginx, Redis middleware and services and is based on the Java8 running environment. The database link account, nginx proxy address and other parameters can be adjusted in the deployment file according to the actual situation. The specific operation process is as follows:

Step 1, create a new directory on the server and extract the Docker package to the current directory;

Step 2, execute docker-compose up -d --build to build the image in the Docker directory;

Step 3, view the image through the Docker images command;

Step 4, access and use the services.

2. Installation components step-by-step and starting services

2.1 Install Nginx on the cloud server

2.1.1 installation

The installation steps are as follows:

Step 1, check and install the required dependent software

(1). gcc:nginx compilation depends on the gcc environment

Installation command: yum install gcc-c++

(2). pcre:(Perl Compatible Regular Expressions) is a Perl library, including a Perl compatible regular expression library. The http module of nginx uses pcre to parse regular expressions.

Installation command: yum install -y pcre pcre-devel

(3). Zlib: This library provides many ways to compress and decompress. nginx uses zlib to gzip the contents of the http packages.

Installation command: yum install -y zlib zlib-devel

(4). Openssl: a powerful secure socket layer cipher library, including the main cryptographic algorithms, common key and certificate encapsulation management functions and SSL protocol, and provides rich applications for testing or other purposes. Nginx not only supports the http protocol but also supports https (that is, http is transmitted over the ssl protocol).

Installation command: yum install -y openssl openssl-devel

Step 2, download the nginx source package

Download command: wget http://nginx.org/download/nginx-1.12.0.tar.gz

Step 3, decompress the source package and enter

(1). Decompression: tar -zxvf nginx-1.12.0.tar.gz

(2). Enter the extracted folder: cd nginx-1.12.0

Step 4, use the command to configure compilation parameters: (You can use./configure --help to query detailed parameters)

Command:

```
./configure \
        —prefix=/usr/local/nginx \
        —pid-path=/var/run/nginx/nginx.pid \
        —lock-path=/var/lock/nginx.lock \
        —error-log-path=/var/log/nginx/error.log \
        —http-log-path=/var/log/nginx/access.log \
        —with-http_gzip_static_module \
        —http-client-body-temp-path=/var/temp/nginx/client \
```

—http-proxy-temp-path=/var/temp/nginx/proxy \

—http-fastcgi-temp-path=/var/temp/nginx/fastcgi \

—http-uwsgi-temp-path=/var/temp/nginx/uwsgi \

—http-scgi-temp-path=/var/temp/nginx/scgi

Note: Before installation, you need to manually create the nginx folders specified above, including /var/temp, /var/temp/nginx, /var/run/nginx/ folders; otherwise, an error will be reported when starting.

Step 5, compile and install

Command: make && make install

You can enter/usr/local/nginx to check whether the file exists in conf, sbin and html folders. If so, the installation is successful.

2.1.2 Basic usage

2.1.2.1. Star nginx

(1). Enter the installation directory

cd /usr/local/nginx/sbin/

(2). Start./nginx

(3). if an error is reported: [emerg] open() "/var/run/nginx/nginx.pid" failed (2: No such file or directory)

You need to check whether the nginx folder does not exist under the/var/run folder. If it does not exist, you need to create a new one.

(4). Check whether to start： ps -ef grep nginx

If there are two processes, master and worker, the startup is successful.

Note: Execute /nginx for start nginx. Here, you can - c to specify the loaded nginx configuration file as follows:

./nginx -c /usr/local/nginx/conf/nginx.conf

If - c is not specified, nginx will load the conf/nginx.conf file by default when starting. The address of this file can also specify the parameter of./configure when compiling and installing nginx (--conf-path= points to the configuration file (nginx. conf)).

2.1.2.2. Stop

(1). Kill (not recommended)

kill -9 processId

(2). Quick stop

cd /usr/local/nginx/sbin && ./nginx -s stop

This method is equivalent to determining the nginx process ID first and then using the kill command to force the process to be killed.

(3). Complete stop (recommended)

cd /usr/local/nginx/sbin && ./nginx -s quit

The stop step is used to stop after finishing the nginx process processing tasks.

2.1.2.3. Restart and reload configuration

(1). Stop and then start (recommended)

./nginx -s quit &&./nginx

(2). Reload configuration file

./nginx -s reload

2.1.2.4. Test

nginx is installed successfully. Start nginx and access nginx through the ip address.

2.2 Install jdk1.8+ on the Cloud Server (1.8 recommended)

The installation steps are as follows:

(1). Check whether other versions of JDK are installed. Execute the command rpm -qa | grep java. The results are as follows:

```
[root@elk-server work]# rpm -qa | grep java
tzdata-java-2017c-1.el7.noarch
java-1.7.0-openjdk-headless-1.7.0.151-2.6.11.1.el7_4.x86_64
java-1.7.0-openjdk-1.7.0.151-2.6.11.1.el7_4.x86_64
java-1.8.0-openjdk-1.8.0.151-1.b12.el7_4.x86_64
```

java-1.8.0-openjdk-headless-1.8.0.151-1.b12.el7_4.x86_64

python-javapackages-3.4.1-11.el7.noarch

javapackages-tools-3.4.1-11.el7.noarch

As shown above, there are currently 1.7 and 1.8 versions of openjdk, and we need to uninstall them;

(2). Uninstall the installed jdk found in the previous step, and execute the following command:

rpm -e --nodeps \

java-1.7.0-openjdk-headless-1.7.0.151-2.6.11.1.el7_4.x86_64 \

java-1.7.0-openjdk-1.7.0.151-2.6.11.1.el7_4.x86_64 \

java-1.8.0-openjdk-1.8.0.151-1.b12.el7_4.x86_64 \

java-1.8.0-openjdk-headless-1.8.0.151-1.b12.el7_4.x86_64

(3). Download jdk-8u161-linux-x64.tar.gz, official website address: http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html

(4). Unzip: tar -zxvf jdk-8u161-linux-x64.tar.gz

(5). Move the unzipped folder to the /usr/local directory: mv jdk1.8.0_161 /usr/local/

(6). Open the file /etc/profile and add the following content:

export JAVA_HOME=/usr/local/jdk1.8.0_161export JRE_HOME=${JAVA_HOME}/jreexport CLASSPATH=.:${JAVA_HOME}/lib/dt.JAVA_HOME/lib/tools.jar:${JRE_HOME}/libexport PATH=${JAVA_HOME}/bin:${PATH}

(7). Make the configuration take effect immediately: source /etc/profile

(8). View the current Java version information: Java -version. You can see the following basic contents:

```shell [root@elk-server ~]# java -version java version "1.8.0_161" Java(TM) SE Runtime Environment (build 1.8.0_161-b12) Java HotSpot(TM) 64-Bit Server VM (build 25.161-b12, mixed mode) ```

To date, CentOS7 has successfully installed JDK8.
```

2.3 Install mysqk5.7+on the Cloud Server (5.7 recommended)

The installation steps are as follows:

Step 1, download MySQL installation package:

[root@localhost local]# wget https://dev.mysql.com/get/mysql57-community-release-el7-11.noarch.rpm

Install MySQL installation source:

[root@localhost local]# yum -y localinstall mysql57-community-release-el7-11.noarch.rpm

Step 2: Install MySQL online

[root@localhost local]# yum -y install mysql-community-server

Note: waiting time is quite long.

Step 3, Start mysql service

[root@localhost local]# systemctl start mysqld

Step 4, Set the startup

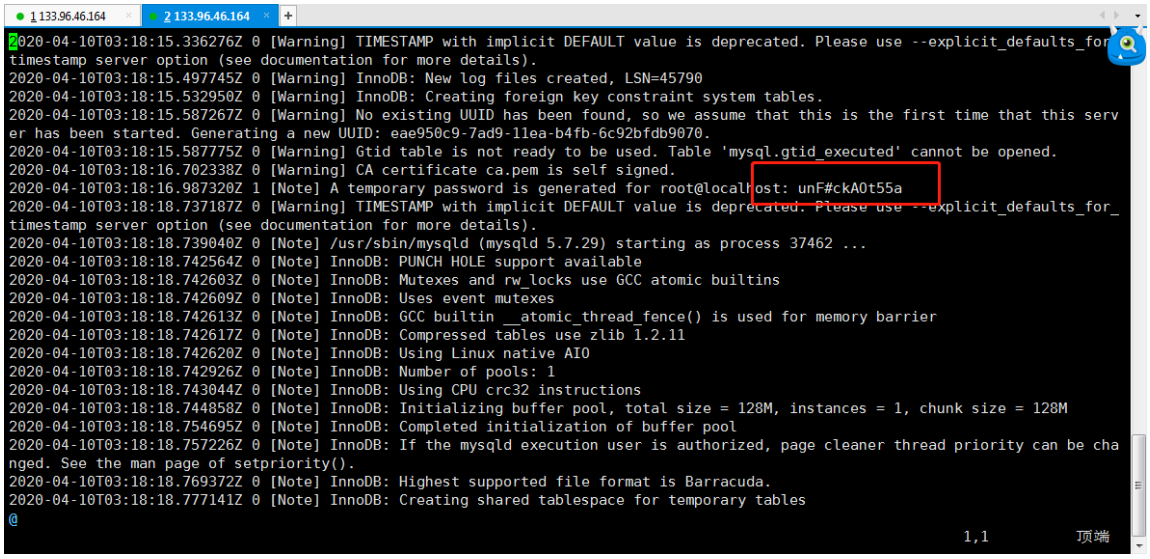
[root@localhost local]# systemctl enable mysqld

[root@localhost local]# systemctl daemon-reload

Step 5, modify the root login password

After MySQL installation, a temporary default password will be generated for the root in the /var/log/mysqld.log file.

[root@localhost local]# vim /var/log/mysqld.log



Remember the initial password: change the root password

[root@localhost local]# mysql -u root -p

```
[root@localhost local]# vim /var/log/mysqld.log
[root@localhost local]# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.7.29

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> ALTER USER 'root'@'localhost' IDENTIFIED BY 'daasan7ujm^YHN';
Query OK, 0 rows affected (0.00 sec)

mysql> GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' IDENTIFIED BY 'daasan7ujm^YHN' WITH GRANT OPTION;
Query OK, 0 rows affected, 1 warning (0.00 sec)

mysql>
```

mysql> ALTER USER 'root'@'localhost' IDENTIFIED BY 'daasan7ujm^YHN';

Query OK, 0 rows affected (0.00 sec)

#Set Remote Login

mysql> GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' IDENTIFIED BY

'daasan7ujm^YHN' WITH GRANT OPTION;

Query OK, 0 rows affected, 1 warning (0.00 sec)

Step 6, exit

mysql> exit

Step 7, the firewall opens port 3306

[root@localhost sysconfig]# cd /etc/sysconfig/[root@localhost sysconfig]# vim

iptables

Add the code as follows: -A INPUT -p tcp --dport 3306 -j ACCEPT

```
1 133.96.46.164    2 133.96.46.164
# sample configuration for iptables service
# you can edit this manually or use system-config-firewall
# please do not ask us to add additional ports/services to this defa
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A INPUT -p tcp --dport 1521 -j ACCEPT
-A INPUT -p tcp --dport 1433 -j ACCEPT
-A INPUT -p tcp --dport 6379 -j ACCEPT
-A INPUT -p tcp --dport 8089 -j ACCEPT
-A INPUT -p tcp --dport 3306 -j ACCEPT
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -j REJECT --reject-with icmp-host-prohibited
-A INPUT -p tcp --dport 80 -j ACCEPT
-A INPUT -p tcp --dport 7879 -j ACCEPT
-A INPUT -p tcp --dport 8000 -j ACCEPT
-A INPUT -p icmp --icmp-type 8 -j ACCEPT
COMMIT

~
~
~
```

Step 8, restart the firewall.

[root@localhost sysconfig]# service iptables restart

Step 9, dispose the default code of mysql as utf-8

[root@localhost sysconfig]# vim /etc/my.cnf

Add the following code:

character_set_server=utf8

init_connect='SET NAMES utf8'

```
[mysqld]
character_set_server=utf8
init_connect='SET NAMES utf8'
#
# Remove leading # and set to the amount of RA
# cache in MySQL. Start at 70% of total RAM fo
# innodb_buffer_pool_size = 128M
#
# Remove leading # to turn on a very important
# changes to the binary log between backups.
# log_bin
#
# Remove leading # to set options mainly usefu
# The server defaults are faster for transacti
# Adjust sizes as needed, experiment to find t
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M
datadir=/var/lib/mysql
```

:wq Save Exit

Step 10, restart MySQL

[root@localhost data]# systemctl restart mysqld

Step 11, Log in as root to view the code

[root@localhost sysconfig]# mysql -u root -p

Enter password:

Welcome to the MySQL monitor. Commands end with ; or \g.

Your MySQL connection id is 2

Server version: 5.7.29 MySQL Community Server (GPL)

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

```
mysql> show variables like '%character%';+--------------------------+----------------------
------+| Variable_name                    | Value                                |+--------------
------------+----------------------------+| character_set_client         | utf8
|| character_set_connection | utf8                                 || character_set_database
| utf8                                 || character_set_filesystem | binary
|| character_set_results        | utf8                                 || character_set_server
| utf8                                 || character_set_system         | utf8
|| character_sets_dir            | /usr/share/mysql/charsets/ |+--------------------------+------
----------------------+8 rows in set (0.00 sec)
mysql>
```

Last local connection test:

2.4 Install Redis on the Cloud Server

2.4.1 installation

The installation steps are as follows:

(1) Download, decompress and compile Redis

```
$ wget http://download.redis.io/releases/redis-5.0.5.tar.gz
$ tar xzf redis-5.0.5.tar.gz
$ cd redis-5.0.5
$ make
```

(2) Enter the decompressed src directory and start Redis with the following command:

```
$ src/redis-server
```

You can use the built-in client to interact with Redis:

```
$ src/redis-cli
redis> set foo barOK
redis> get foo"bar"
```

2.4.2 Description and supplement

(1) Download

```
[root@VM_0_6_centos soft]# wget http://download.redis.io/releases/redis-5.0.5.tar.gz
```

--2019-12-14 18:06:49-- http://download.redis.io/releases/redis-5.0.5.tar.gz

Resolving download.redis.io (download.redis.io)... 109.74.203.151

Connecting to download.redis.io (download.redis.io)|109.74.203.151|:80... connected.

HTTP request sent, awaiting response... 200 OKLength: 1975750 (1.9 M) [application/x-gzip]

Saving to: ‘redis-5.0.5.tar.gz’

100%[==========>] 1,975,750 47.8KB/s in 34s

2019-12-14 18:07:25 (56.0 KB/s) - ‘redis-5.0.5.tar.gz’ saved [1975750/1975750]

[root@VM_0_6_centos soft]# ls

redis-5.0.5.tar.gz

(2) Decompression

[root@VM_0_6_centos soft]# tar xzf redis-5.0.5.tar.gz[root@VM_0_6_centos soft]# ls

redis-5.0.5 redis-5.0.5.tar.gz

(3) Compile

After decompressing (cd redis-5.0.5), enter the Redis directory compilation (make). This process takes a long time. Just wait patiently.

[root@VM_0_6_centos soft]# cd redis-5.0.5[root@VM_0_6_centos redis-5.0.5]# ls00-RELEASENOTES COPYING Makefile redis.conf runtest-moduleapi srcBUGS deps MANIFESTO runtest runtest-sentinel testsCONTRIBUTING INSTALL README.md runtest-cluster sentinel.conf utils[root@VM_0_6_centos redis-5.0.5]# make

Compilation success flag:

Hint: It is a good idea to run 'make test' ;)

make[1]: Leaving directory `/var/soft/redis-5.0.5/src'[root@VM_0_6_centos redis-5.0.5]#

(4) Start Redis

Two startup modes: ① src/redis-server ② ./redis-server &

The first is foreground initiation. The redis-server program is executed in the src directory. Redis application is started the same way as the previous platform. You cannot exit the current window. Once you exit the window, the application terminates.

#foreground initiation [root@VM_0_6_centos redis-5.0.5]# src/redis-server

#Exit the window to view the running status [root@VM_0_6_centos redis-5.0.5]# ps -ef | grep redis

root 25002 18769 0 18:14 pts/0 00:00:00 grep --color=auto redis

The second is background startup. Execute./redis-server & in the src directory. Currently, close the window and check the Redis process. It still exists (recommended).

Background startup [root@VM_0_6_centos redis-5.0.5]# cd src[root@VM_0_6_centos src]#./redis-server &

Exit the window to view the running status [root@VM_0_6_centos src]# ps -ef | grep redis

root 25208 18769 0 18:16 pts/0 00:00:00./redis-server *:6379

root 25258 18769 0 18:16 pts/0 00:00:00 grep --color=auto redis

(5) Close Redis

① Shut down using the Redis client

Issue the shutdown command to the server, switch to the redis-3.2.9/src/ directory, and execute ./redis-cli shutdown. This method is recommended. Redis completes data operations before closing.

[root@VM_0_6_centos src]# ./redis-cli shutdown25208:M 14 Dec 2019 18:21:26.395 # User requested shutdown...25208:M 14 Dec 2019 18:21:26.395 * Saving the final RDB snapshot before exiting.25208:M 14 Dec 2019 18:21:26.405 * DB saved on disk25208:M 14 Dec 2019 18:21:26.405 # Redis is now ready to exit, bye bye...[1]+ Done ./redis-server

[root@VM_0_6_centos src]# ps -ef | grep redis

root 26054 18769 0 18:21 pts/0 00:00:00 grep --color=auto redis[root@VM_0_6_centos src]#

② Close with kill pid or kill -9 pid

This method does not consider whether the current application has data that is performing operations and closes the application directly.

First, use ps -ef | grep redis to determine the process number, and then use kill pid.

[root@VM_0_6_centos src]# ps -ef | grep redis

root 421 18769 0 19:05 pts/0 00:00:00./redis-server *:6379

root 454 18769 0 19:05 pts/0 00:00:00 grep --color=auto redis

[root@VM_0_6_centos src]# kill 421[root@VM_0_6_centos src]# 421:signal-handler (1576321600) Received SIGTERM scheduling shutdown...421:M 14 Dec 2019 19:06:40.079 # User requested shutdown...421:M 14 Dec 2019 19:06:40.079 * Saving the final RDB snapshot before exiting.421:M 14 Dec 2019 19:06:40.086 * DB saved on disk421:M 14 Dec 2019 19:06:40.086 # Redis is now ready to exit, bye bye...^C[1]+ Done ./redis-server

[root@VM_0_6_centos src]# ps -ef | grep redis

root 655 18769 0 19:06 pts/0 00:00:00 grep --color=auto redis[root@VM_0_6_centos src]#

This is finished.

2.5 Enter the BreedingEIS-M\BreedingEIS\management system ui folder, and package the front-end code with the following commands to generate a dist folder, which will be published to nginx for use.

cd management_system_ui

npm install --unsafe-perm --registry=https://registry.npm.taobao.org

npm run build:prod

2.6 The BreedingEIS-M\BreedingEIS\management system\sql\liuxn_yuzhong.sql file is run into mysql, and the corresponding database is established.

2.7 Enter the BBreedingEIS-M\BreedingEIS\management system folder, package the back-end code, generate yuzhong.jar file, and then publish it to the server to run.

Packaging command：call mvn clean package -Dmaven.test.skip=true

Run command：nohup java -jar yuzhong.jar >nohup.out 2>&1 &