

第1章 安装mysql

1. 下载mysql 5.7

下载地址

```
1 https://downloads.mysql.com/archives/community/  
2 https://downloads.mysql.com/archives/get/p/23/file/mysql-5.7.28-linux-glibc2.12-  
x86_64.tar.gz
```

2. 创建目录

```
1 mkdir -p /data/soft  
2 mkdir -p /data/mysql_3306/
```

3. 下载并解压软件

```
1 cd /data/soft  
2 wget https://downloads.mysql.com/archives/get/p/23/file/mysql-5.7.28-linux-  
glibc2.12-x86_64.tar.gz  
3 tar xzf mysql-5.7.28-linux-glibc2.12-x86_64.tar.gz -C /opt/  
4 mv /opt/mysql-5.7.28-linux-glibc2.12-x86_64 /opt/mysql-5.7.28  
5 ln -s /opt/mysql-5.7.28 /opt/mysql
```

4. 设置环境变量

```
1 echo "export PATH=$PATH:/opt/mysql/bin" >>/etc/profile  
2 source /etc/profile  
3 mysql -V
```

5. 清除遗留环境

```
1 rpm -qa|grep mariadb  
2 yum remove mariadb-libs -y  
3 rm -rf /etc/my.cnf
```

6. 安装mysql依赖包

```
1 yum install -y libaio-devel
```

7. 创建mysql普通用户并授权

```
1 useradd -s /sbin/nologin -M mysql
2 chown -R mysql:mysql /data/
3 chown -R mysql:mysql /opt/mysql*
```

8.初始化数据库

```
1 mysqld --initialize-insecure --user=mysql --basedir=/opt/mysql --
  datadir=/data/mysql_3306/
```

9.编辑mysql配置文件

```
1 cat> /etc/my.cnf <<EOF
2 [mysqld]
3 user=mysql
4 basedir=/opt/mysql
5 datadir=/data/mysql_3306
6 socket=/tmp/mysql.sock
7 [mysql]
8 socket=/tmp/mysql.sock
9 EOF
```

10.准备启动脚本并启动数据库

```
1 cp /opt/mysql/support-files/mysql.server /etc/init.d/mysqld
2 chkconfig --add mysqld
3 systemctl start mysqld
4 netstat -lntup|grep 3306
```

11.前台启动

```
1 /opt/mysql/bin/mysqld --basedir=/opt/mysql --datadir=/data/mysql_3306 --plugin-
  dir=/opt/mysql/lib/plugin --user=mysql --log-error=/data/mysql_3306/log/mysql.log --
  pid-file=/data/mysql_3306/db01.pid --socket=/tmp/mysql.sock
```

12.修改root密码

```
1 mysqladmin password
```

13.登陆mysql

```
1 mysql -uroot -p123456
```

第2章 安装SonarQube

1.安装java环境

```
1 yum install java -y
```

2.解压并创建软链接

```
1 unzip sonarqube-7.0.zip -d /opt/  
2 ln -s /opt/sonarqube-7.0/ /opt/sonarqube
```

3.创建普通用户并更改授权

```
1 useradd sonar -M -s /sbin/nologin  
2 chown -R sonar.sonar /opt/sonarqube*
```

4.配置sonarqube数据库连接信息

```
1 [root@sonar ~]# vim /opt/sonarqube/conf/sonar.properties  
2 sonar.jdbc.username=root  
3 sonar.jdbc.password=123456  
4 sonar.jdbc.url=jdbc:mysql://localhost:3306/sonar?  
  useUnicode=true&characterEncoding=utf8&rewriteBatchedStatements=true&useConfigs=maxP  
  erformance&useSSL=false
```

5.指定sonarqube启动用户

```
1 vim /opt/sonarqube/bin/linux-x86-64/sonar.sh  
2 RUN_AS_USER=sonar
```

6.创建sonarqube数据库

```
1 mysql -uroot -p123456 -e 'create database sonar default character set utf8;'  
2 mysql -uroot -p123456 -e 'show databases;'
```

7.编写systemd启动文件

```
1 cat >/usr/lib/systemd/system/sonar.service<<'EOF'  
2 [Unit]  
3 Description=sonar  
4  
5 [Service]
```

```
6 ExecStart=/opt/sonarqube/bin/linux-x86-64/sonar.sh start
7 ExecStop=/opt/sonarqube/bin/linux-x86-64/sonar.sh stop
8 Type=forking
9 User=sonar
10 Group=sonar
11
12 [Install]
13 WantedBy=multi-user.target
14 EOF
15 systemctl daemon-reload
```

8.启动SonarQube

```
1 systemctl start sonar.service
```

9.检查服务

```
1 [root@sonar ~]# netstat -lntup|grep java
2 tcp        0      0 127.0.0.1:32000      0.0.0.0:*            LISTEN
   18202/java
3 tcp6       0      0 :::9000              :::*                  LISTEN
   18305/java
4 tcp6       0      0 127.0.0.1:9001      :::*                  LISTEN
   18227/java
5 tcp6       0      0 127.0.0.1:36949     :::*                  LISTEN
   18475/java
```

10.启动报错

使用systemd启动后失败，查看es日志发现提示max file descriptors太低：

```
1 [root@sonar /opt/sonarqube/logs]# tail -f /opt/sonarqube/logs/es.log
2 2020.05.14 09:51:19 INFO es[][o.e.t.TransportService] publish_address
   {127.0.0.1:9001}, bound_addresses {127.0.0.1:9001}
3 2020.05.14 09:51:19 WARN es[][o.e.b.BootstrapChecks] max file descriptors [4096]
   for elasticsearch process is too low, increase to at least [65536]
4 2020.05.14 09:51:19 WARN es[][o.e.b.BootstrapChecks] max virtual memory areas
   vm.max_map_count [65530] is too low, increase to at least [262144]
```

解决方法：

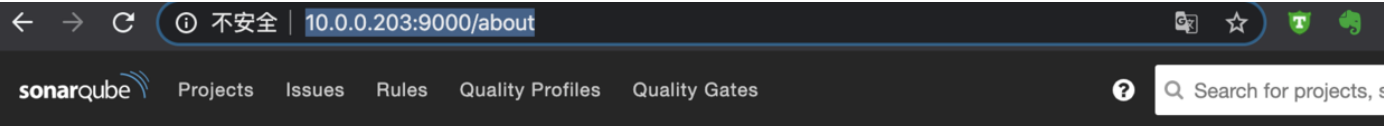
```
1 echo "vm.max_map_count=262144" >> /etc/sysctl.conf
2 echo "root - nofile 65536" >> /etc/security/limits.conf
3 echo "sonar - nofile 65536" >> /etc/security/limits.conf
4 sysctl -p
```

第3章 初始化SonarQube

1.使用admin登陆

登陆地址为:

1 | http://10.0.0.203:9000/about



Continuous Code Quality

Log in

Read documentation

0
Projects Analyzed

账号密码均为admin:

Log In to SonarQube

admin

.....

Log in

Cancel

2.生成token

在输入框内输入jenkins，然后点击Generate生成token，需要保存好这个token，后面会用到

Welcome to SonarQube!

Want to quickly analyze a first project? Follow these 2 easy steps.

1 Provide a token

jenkins: **be400d585a529e6e2152e6742fe3f5cb3fc803d2** ✖

The token is used to identify you when an analysis is performed. If it has been compromised, you can revoke it at any point of time in your user account.

[Continue](#)

3.选择项目类型

Welcome to SonarQube!

[Skip this tutorial](#)

Want to quickly analyze a first project? Follow these 2 easy steps.

[Find it back anytime in the Help section](#)

1 Provide a token

✓ jenkins: be400d585a529e6e2152e6742fe3f5cb3fc803d2

2 Run analysis on your project

What is your project's main language?

Java

C# or VB.NET

Other (JS, Python, PHP, ...)

What is your OS?

Linux

Windows

macOS

Define a unique project key

html

Done

Allowed characters are alphanumeric, '-', '_', '.' and ':', with at least one non-digit. 400 characters max.

点击Done之后会给我们生成提示信息：

第4章 安装插件

1.在线安装中文插件方法

Administration-->Marketplace-->chinese-->install

2.离线安装插件方法

离线安装的话只需要将插件压缩包解压到指定目录然后重启服务即可，解压之前可以先备份插件目录

```
1 mv /opt/sonarqube/extensions/plugins/ /opt/sonarqube/extensions/plugins_bak
2 tar xf sonar_plugins.tar.gz -C /opt/sonarqube/extensions/
```

3.重启服务

```
1 systemctl restart sonar.service
```

4.浏览器访问查看



第5章 安装客户端

1.jenkins主机安装客户端

我们需要将jenkins拉取的代码推送到SonarQube,所以需要在jenkins主机上安装sonar客户端:

```
1 unzip sonar-scanner-cli-4.0.0.1744-linux.zip -d /opt/
2 cd /opt/
3 ln -s sonar-scanner-4.0.0.1744-linux sonar-scanner
```

写入环境变量:

```
1 echo 'export PATH=$PATH:/opt/sonar-scanner/bin' >> /etc/profile
2 source /etc/profile
```

2.推送代码到SonarQube

进入代码目录执行推送命令

注意: 这里的推送命令是初始化的时候生成的, Dsonar.login的值也是初始化时候生成的token

```
1 cd /var/lib/jenkins/workspace/h5game/
2 /opt/sonar-scanner/bin/sonar-scanner \
3   -Dsonar.projectKey=html \
4   -Dsonar.sources=. \
5   -Dsonar.host.url=http://10.0.0.203:9000 \
6   -Dsonar.login=4f57dfb332463fa8220be49856a0f1d27c88a142
```

我们也可以将服务器相关的命令写入配置文件里，这样推送的命令可以精简一些：

```
1 vim /opt/sonar-scanner/conf/sonar-scanner.properties
2 sonar.host.url=http://10.0.0.203:9000
3 sonar.login=be400d585a529e6e2152e6742fe3f5cb3fc803d2
4 sonar.sourceEncoding=UTF-8
```

然后推送命令只需要指定两个选项即可：

```
1 cd /var/lib/jenkins/workspace/my-freestyle-job/
2 sonar-scanner \
3   -Dsonar.projectKey=html \
4   -Dsonar.sources=.
```

3.web页面查看扫描结果



4.执行报错解决

报错现象：推送的时候提示我们找不到node环境

```
INFO: Sensor SonarCSS Metrics [cssfamily]
INFO: Sensor SonarCSS Metrics [cssfamily] (done) | time=788ms
INFO: Sensor SonarCSS Rules [cssfamily]
ERROR: Error when running: 'node -v'. Is Node.js available during analysis? No CSS files will be analyzed.
org.sonarsource.nodejs.NodeCommandException: Error when running: 'node -v'. Is Node.js available during analysis?
    at org.sonarsource.nodejs.NodeCommand.start(NodeCommand.java:77)
    at org.sonarsource.nodejs.NodeCommandBuilderImpl.getVersion(NodeCommandBuilderImpl.java:171)
    at org.sonarsource.nodejs.NodeCommandBuilderImpl.checkNodeCompatibility(NodeCommandBuilderImpl.java:144)
    at org.sonarsource.nodejs.NodeCommandBuilderImpl.build(NodeCommandBuilderImpl.java:120)
    at org.sonar.css.plugin.StylelintCommandProvider.nodeCommand(StylelintCommandProvider.java:64)
    at org.sonar.css.plugin.CssRuleSensor.execute(CssRuleSensor.java:111)
```

解决方法：在jenkins服务器上安装nodejs环境,然后重新推送就不会再报错了：

```
1 cd /opt/
2 wget https://nodejs.org/dist/v12.13.0/node-v12.13.0-linux-x64.tar.xz
3 tar xf node-v12.13.0-linux-x64.tar.xz
4 mv node-v12.13.0-linux-x64 node
5 echo 'export PATH=$PATH:/opt/node/bin' >> /etc/profile
6 source /etc/profile
7 npm -v
8 node -v
```


第6章 与Jenkins集成

1.配置SonarQube凭证信息

在jenkins页面进入 系统管理-->系统配置-->找到sonar的配置

SonarQube servers

Environment variables

SonarQube installations

Enable injection of SonarQube server configuration as build environment variables

If checked, job administrators will be able to inject a SonarQube server configuration as environment variables in the build.

Add SonarQube

List of SonarQube installations

填写sonar服务器信息：

SonarQube servers

Environment variables

SonarQube installations

Enable injection of SonarQube server configuration as build environment variables

If checked, job administrators will be able to inject a SonarQube server configuration as environment variables in the build.

Name

sonar

Server URL

http://10.0.0.203:9000

Default is http://localhost:9000

Server authentication token

无

添加

此时点击添加没有反应，需要先保存

SonarQube authentication token. Mandatory when anonymous access is disabled.

Add SonarQube

List of SonarQube installations

高级...

Delete SonarQube

Locale

保存

应用

此时点击添加凭证按钮没有反应，没关系，先保存一下，然后回来再添加一次即可：

SonarQube servers

Environment variables

SonarQube installations

Enable injection of SonarQube server configuration as build environment variables

If checked, job administrators will be able to inject a SonarQube server configuration as environment variables in the build.

Name

sonar

Server URL

http://10.0.0.203:9000

Default is http://localhost:9000

Server authentication token

无

添加

SonarQube authentication token

Jenkins

Jenkins 凭据提供者

token. Mandatory when anonymous access is disabled.

Add SonarQube

List of SonarQube installations

高级...

Delete SonarQube

Jenkins 凭据提供者: Jenkins

填写sonar初始化的token信息：

Jenkins 凭据提供者: Jenkins

添加完成后就可以选择sonar的凭证了：

SonarQube servers

Environment variables ☐ Enable injection of SonarQube server configuration as build environment variables
If checked, job administrators will be able to inject a SonarQube server configuration as environment variables in the build.

SonarQube installations

Name

Server URL

Default is http://localhost:9000

Server authentication token

SonarQube authentication token. Mandatory when anonymous access is disabled.

List of SonarQube installations

2.配置sonar客户端家目录

点击系统管理-->全局工具配置-->找到SonarQube Scanner选项：

SonarQube Scanner

SonarQube Scanner 安装

☐ SonarQube Scanner

Name

SONAR_RUNNER_HOME jenkins 服务器上 sonar 客户端安装目录

☐ Install automatically

系统下SonarQube Scanner 安装列表

3.工程中配置sonar构建选项

添加构建步骤：

General 源码管理 构建触发器 构建环境 **构建** 构建后操作

☐ Delete workspace before build starts

☐ Use secret text(s) or file(s)

☐ Provide Configuration files

☐ Abort the build if it's stuck

☐ Add Changelog Information to Environment

☐ Add timestamps to the Console Output

☐ Generate Release Notes

☐ Inspect build log for published Gradle build scans

☐ With Ant

构建

填写详细信息：

填写参数：

```
1 sonar.projectName=${JOB_NAME}
2 sonar.projectKey=html
3 sonar.sources=.
```

4.调整构建执行顺序

注意!!! 这里我们还需要将构建顺序调整一下, 先执行代码扫描, 然后再发布版本

可以直接拖动选项块来调整顺序, 最终结果如下:

5.发布测试

我们可以使用git尝试发布代码, 然后查看执行是否成功:

```
1 git branch
2 git pull
3 vim index.html
4 git add .
5 git commit -m "v5.0 稳定版"
6 git push -u origin master
```

sonar查看是否发布: