2019年9月11日 - 基础环境搭建、Docker部署

作者: 雷世光

主要包含：MySQL 主从数据库、RabbitMQ、Redis、sentinel、nacos、ES、Zipkin，也介绍了 docker 的一些使用...

一、MySQL 数据库安装

一开始在 windows 中安装了两个 mysql 8.0，但和 nacos 有些不兼容。为了更快速解决问题，重新在 linux 机器上安装了两个 mysql：

```
docker run --name mysql-master --privileged=true -v /home/mysql/master-data:/var/lib/mysql -p 3306:3306 -e MYSQL_ROOT_PASSWORD=root -d
xiaochunping/mysql-master
docker run --name mysql-slave --privileged=true -v /home/mysql/slave-data:/var/lib/mysql -p 3307:3306 --link mysql-master:master -e
MYSQL_ROOT_PASSWORD=root -d xiaochunping/mysql-slave
```

安装好之后，可以使用 docker ps 查看已经启动的容器。
命令：docker exec -it mysql-master /bin/bash 进入该容器。
命令：mysql -u root -p
接下来就是开放权限，开放远程登陆什么的了...

进入主数据：

```
docker exec -it mysql-master /bin/bash
mysql -u root -p
//增加权限
grant replication slave on *.* to 'test'@'%' identified by '123456';
flush privileges;
// 查看主的状态
show master status;
```

进入从数据库，开启同步

```
docker exec -it mysql-slave /bin/bash
mysql -u root -p
//设置从库，其中的master_log_file和master_log_pos从主数据库的状态给到
change master to master_host='master', master_user='test', master_password='123456', \
master_port=3306, master_log_file='mysql-bin.000003', master_log_pos=589, master_connect_retry=30;
//开始同步
start slave;
//查看从库的状态，正常情况下 SlaveIORunning 和 SlaveSQLRunning 都是Yes
show slave status\G;
```

其中关于 master_host 中的 ip ，指的是容器的独立ip,可以通过docker inspect --format='{{.NetworkSettings.IPAddress}}' 容器名称|容器id查询容器的ip
参考：基于Docker的Mysql主从复制搭建

反复尝试之后，开启主从时，日志的 master_log_pos 一定要匹配。也就是 master 在 show master status 拿到 master_log_pos 时，先不要做改动。设置好了从库之后，再进行。
如果要同步老数据，就需要 dump 还原了。

关于主从复制的原理：

1. 首先，mysql主库在事务提交时会把数据库变更作为事件Events记录在二进制文件binlog中；mysql主库上的sys_binlog控制binlog日志刷新到磁盘。
2. 主库推送二进制文件binlog中的事件到从库的中继日志relay log,之后从库根据中继日志重做数据库变更操作。通过逻辑复制，以此来达到数据一致。
3. Mysql通过3个线程来完成主从库之间的数据复制：其中BinLog Dump线程跑在主库上，I/O线程和SQl线程跑在从库上。当从库启动复制（start slave）时，首先创建I/O线程连接主库，主库随后创建Binlog Dump线程读取数据库事件并发给I/O线程，I/O线程获取到数据库事件更新到从库的中继日志Realy log中去，之后从库上的SQl线程读取中继日志relay log 中更新的数据库事件并应用。

插播：Windows 中的安装

(如果已经在 linux 中安装好了，就不需要再 windows 中安装)
(以 windows 安装 MySQL Community Server 8.0.17 版本为例，采用 MySQL installer for Windows 进行安装)
1、官网下载：https://dev.mysql.com/downloads/mysql/
2、按照步骤安装，期间提示输入 root 用户密码；
3、安装完成后注意开启对应端口...
4、可以使用刚安装好的 shell 客户端访问数据库，或者使用脚本：

```
mysql -u root -p
-- 切换到 mysql 库
use mysql
```

5、允许某些账户远程访问...

```
-- 查看访问权限
SELECT user, host FROM user;
-- 修改访问权限
update user set host = '%' where user = '用户名';
-- 用户授权
GRANT ALL PRIVILEGES ON *.* TO 'root'@'%'WITH GRANT OPTION;
-- 刷新访问权限
FLUSH PRIVILEGES;
```

5、新增数据库，前一步安装的数据库是系统数据库，我们要用的话，需要安装自己的库。类似于 oracle 的表空间

```
-- 登陆 MySQL 服务
CREATE DATABASE 数据库名;
-- 或者用这个同时设定字符集：CREATE DATABASE IF NOT EXISTS [数据库名] DEFAULT CHARSET utf8 COLLATE utf8_general_ci;
```

6、新建用户

```
-- 此处的localhost表示只允许本地登陆，改为 % 表示任意电脑都可以
CREATE USER 'aaa'@'%' IDENTIFIED BY '123456'
-- 授权
grant 权限 on 数据库.* to 用户名@登录主机 identified by "密码";
-- 授权示例（所有权限）
GRANT ALL PRIVILEGES ON demoDB.* TO 'demo'@'%';
-- 授权示例（部分权限）
GRANT select,update ON demoDB.* TO 'demo'@'%';
-- 刷新系统权限表
flush privileges;
```

7、删除用户，修改密码

```
-- 删除用户
Delete FROM user Where User='demo' and Host='localhost';
-- 删除账户及权限
drop user 用户名@'%';
drop user 用户名@ localhost;
-- 修改指定用户密码
update mysql.user set password=password('新密码') where User="demo" and Host="localhost";
-- 刷新系统权限表
flush privileges;
```

8、列出所有数据库，表，切换数据库...

```
-- 列出所有数据库
show databases;
-- 切换数据库
use '数据库名';
-- 列出所有表
show tables;
-- 显示数据表结构
describe 表名;
-- 删除数据库和数据表
drop database 数据库名;
drop table 数据表名;
```

按照这样，依次创建了如下 3 个数据库：

1. ag_admin_v1，用户管理，账号 agAdmin，密码 123456
2. ag_auth_v1，权限管理，账号 agAuth，密码 123456
3. course_classroom，业务系统的库，账号，courseClassroom，密码 123456

插播：多个 MySQL 实例的安装

（如果已经在linux中安装好了，就不需要在windows中安装）

有用到 mysql 的主从，于是就需要多个 MySQL 实例...参考官网

1. 打开命令行窗口，进入到已安装好了的 MySQL 目录，如：C:\Program Files\MySQL\MySQL Server 8.0\bin
2. 执行 mysqladmin -P 3307 -u root -p shutdown，意为：使用密码登陆 root 用户，关闭 3307 端口的 mysql 服务
3. 依据提示输入密码即可关闭该 mysql 服务
4. 重新下载 MySQL 的压缩包版本 MySQL Community Downloads
5. 新建文件夹 MySQL80SLAVE 用以解压 MySQL 压缩包；
6. 在新文件夹的根目录新建 my.ini 文件，内容附在末尾；
7. 执行 mysqld --initialize --user=mysql --console 进行初始化；
8. 成功后，会生成数据库密码，记录该密码；
9. 执行 mysqld –install MySQL80SLAVE 创建 windows 服务；
10. 修改注册表键值 HKEY_LOCAL_MACHINE -> SYSTEM -> CurrentControlSet -> services -> MySQL80SLAVE
    为 "C:\Program Files\MySQL\MySQL Server 8.0 SLAVE\bin\mysqld" –defaults-file="C:\Program Files\MySQL\MySQL Server 8.0 SLAVE\my.ini" MySQL80SLAVE
    意思为：指定 my.ini 的路径，如果不指定，那么默认会在 C:\ProgramData\MySQL 里面
11. 可以启动服务了~
12. 之后，登陆进去 mysql –u root –P 3308 –p
13. 输入初始密码...
14. 修改密码 ALTER USER 'root'@'localhost' IDENTIFIED BY 'new_password';
15. 依照之前《MySQL 数据库安装》中的提示，将用户开启，允许远程访问；

```
[mysqld]
# set basedir to your installation path
basedir=C:\\Program Files\\MySQL\\MySQL Server 8.0 SLAVE
# set datadir to the location of your data directory
datadir=C:\\Program Files\\MySQL\\MySQL Server 8.0 SLAVE\\data
port=3308
server_id=MySQL80SLAVE
```

部署过程中遇到的问题：

1. MySQL80SLAVE 服务安装之后无法运行
   当前的 mysql 是直接从 MySQL80 拷贝的，怕是有些不完整。于是便重新下载...
2. 连接 MySQL 提示：Server returns invalid timezone. Go to 'Advanced' tab and set 'serverTimezone' property manually.
   查看数据库的时区：show variables like 'time_zone'; 发现是 SYSTEM
   系统时区为 UTC+8，北京/香港 时间
   在连接参数中加上：serverTimezone=Hongkong

至此，同一台计算机中，多个 MySQL 实例即部署完成。

二、RabbitMQ 部署运行

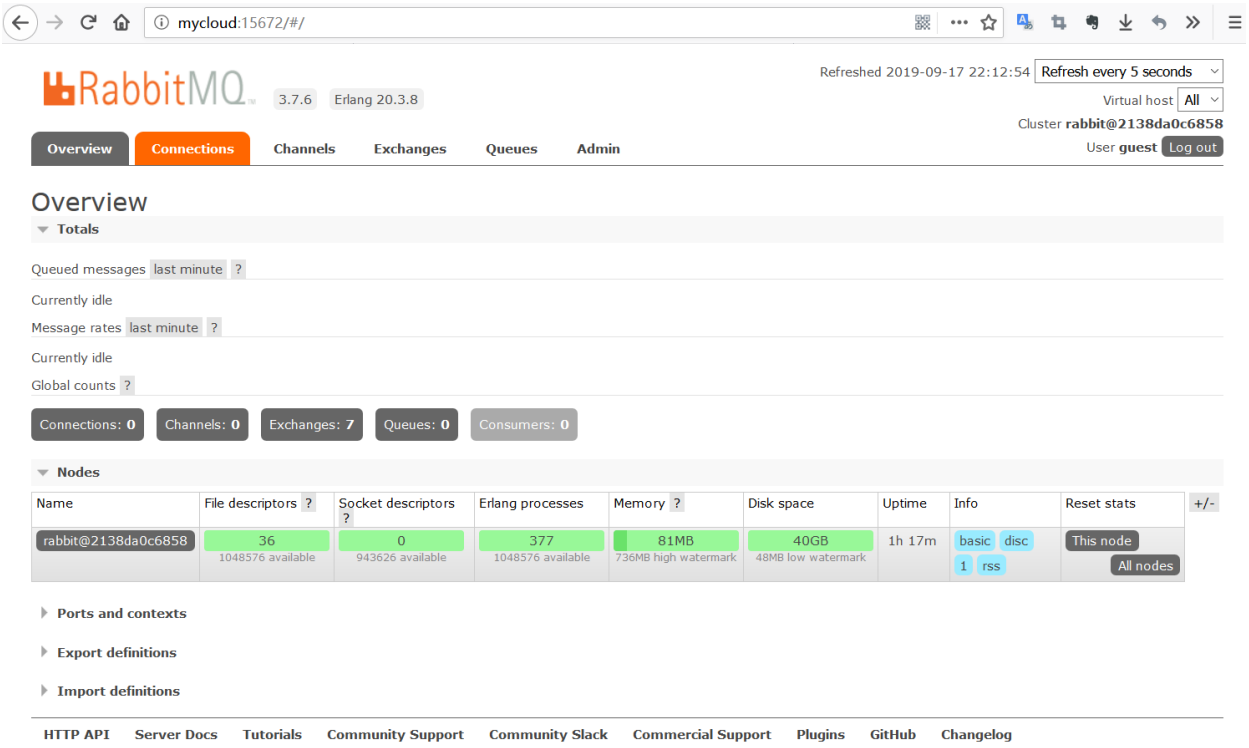
这儿直接使用 docker 部署：docker run -d --name rabbitmq -p 5671:5671 -p 5672:5672 -p 4369:4369 -p 25672:25672 -p 15671:15671 -p 15672:15672 -p 15674:15674 -p 15670:15670 -p 15673:15673 --restart=always xiaochunping/rabbitmq:management

部署过程中遇到问题：

1. 存在同名的容器：docker: Error response from daemon: Conflict. The container name "/rabbitmq" is already in use by container "e0a824f78890a8e9c176c03a0555333a4a7c30729cca3c2cbf939510f7834a08". You have to remove (or rename) that container to be able to reuse that name.于是使用命令：docker ps -a 显示所有存在的容器
   并使用 docker container rm rabbitmq 命令，把对应容器删除。
2. 端口占用：docker: Error response from daemon: driver failed programming external connectivity on endpoint rabbitmq (e682a293c7a9673b0da972796b82237c3edb3621dea76dd893a2b6dbbf849fab): Error starting userland proxy: listen tcp 0.0.0.0:5672: bind: address already in use.
   于是使用命令：netstat -apn | grep 5672 检查端口占用，发现是一个 id 为 3139 的 java 程序在进行监听
   使用命令：cd /proc/3139 查看该进程的文件夹，使用命令 ls -ail 发现该进程的执行路径是 /usr/java/jdk1.8.0_211-amd64/bin/java，很可能是开机自动启动的程序
   使用命令：systemctl 查看会伴随系统启动的程序，找到了一个 activemq.service loaded active running ActiveMQ service，很显然就是它了！
   使用命令： systemctl stop activemq.service 停止该服务，并使用 systemctl disable activemq.service 使其不自动启动。参考systemctl命令

再次执行 rabbitMQ 的部署，发现部署成功，使用 docker ps 可以看到当前已经启动的容器，使用 free -h 查看当前内存还够...
如果要删除该容器，使用命令 docker container stop xxx 终止容器和 docker container prune 执行删除，或者使用 docker rm -f xxxx 直接删除

浏览器访问：<ip>:15672 就可以访问到启动的 rabbitMQ，使用默认的账号密码（guest/guest）即可登陆



三、redis 部署运行

docker 命令：docker run -d --name redis -p 6379:6379 redis redis-server --requirepass "redis123456" --appendonly yes

插一句，如果不知道启动容器时候的命令是什么意思，可以到 hub.docker.com 上面进行查找。
比如，这儿为 redis 加上了密码，redis123456

而要如何连接呢？参考官网，可以使用比如 docker run -it --rm redis redis-cli -h redis 命令开启一个 redis 客户端。

提示错误信息：

1. 无法连接：Could not connect to Redis at -redis:6379: Name or service not known
   是怎么回事呢？docker 网络的问题，使用 docker network ls 可以查看当前可用的网络模式，默认为桥接。

那么，删除掉刚才的 redis 容器：docker rm -f redis
则，修改 docker 命令：docker run -d --name redis --network redis-network -p 6379:6379 redis redis-server --requirepass "redis123456" --appendonly yes

提示错误信息：

1. 未找到网络：docker: Error response from daemon: network redis-network not found.
   需要先创建网络：docker network create redis-network

再执行 redis 的 docker 命令，同时使用 redis-cli 进行登陆：docker run -it --network redis-network --rm redis redis-cli -h redis
发现连接上了，redis:6379>
而要设置一个键值呢？执行：set testKey hello

提示错误信息：

1. (error) NOAUTH Authentication required.
   是因为没有权限，要首先登陆才行，命令：auth redis123456，再次执行即可。
   如果要退出 redis-cli，使用 quit 命令即可

四、sentinel 1.6.1

这儿从官网下载了 cloud-sentinel.jar ，构建好了 build.sh，上传到云服务器，直接运行即可。

传输步骤：
1、云服务器中建立好文件夹，比如在 home 目录创建 docker 文件夹，把所有和 docker 有关的都放到这里面；
2、在 docker 中再创建 sentinel-jar 文件夹
3、ftp 连接到云服务器，我这儿使用的是 psftp；
4、cd 命令进入 ~/docker/sentinel-jar 文件夹，lcd 命令进入本地 cloud-sentinel.jar 所在文件夹，可以使用 dir 或 !dir 查看当前所处目录
5、put xxxx 的方式，将文件上传到云服务器

启动命令 ./build.sh

提示错误信息：

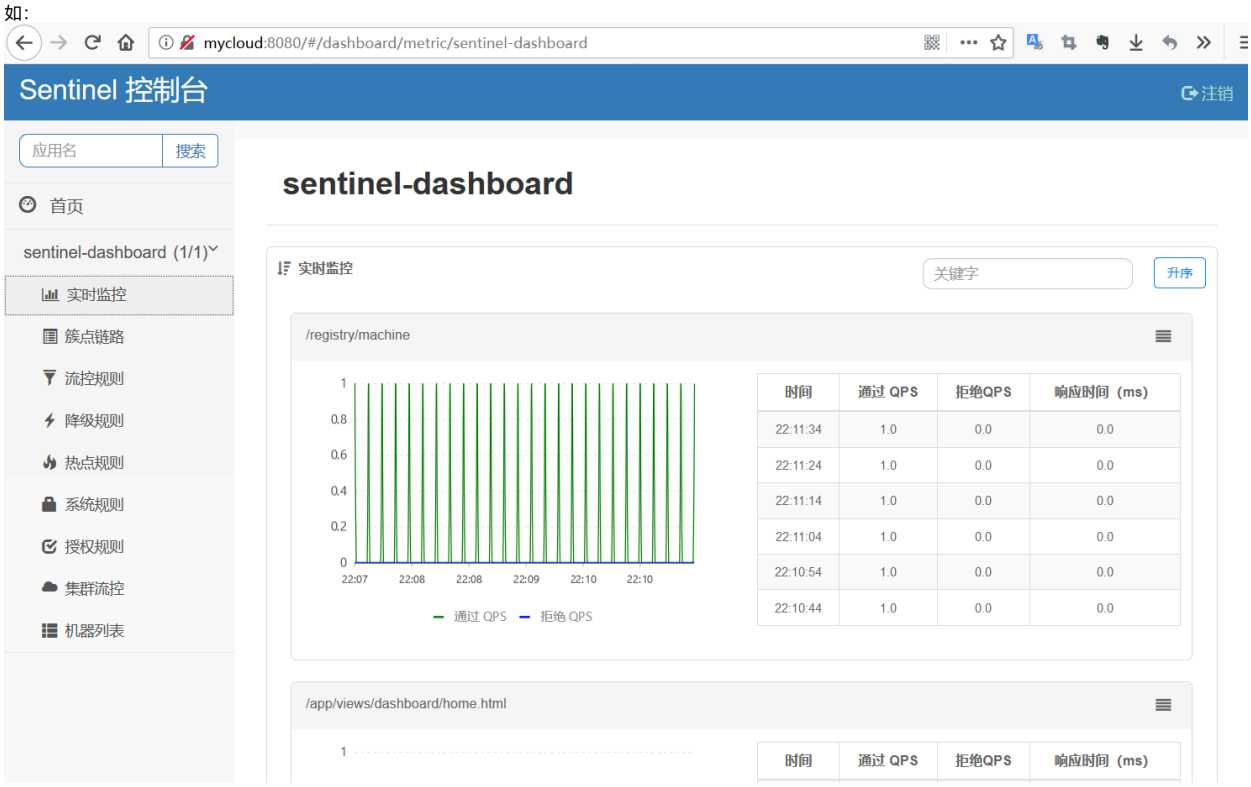
1. 无权限 -bash: ./build.sh: Permission denied
   命令 ls -l build.sh 查看当前权限为 -rw-r--r-- 1 root root 1098 Sep 17 21:33 build.sh，没有运行权限
   命令 chmod u+x build.sh 添加权限
   重新查看权限，确认具有了执行权限 -rwxr--r-- 1 root root 1098 Sep 17 21:33 build.sh
2. 有命令没有识别，如：./build.sh: line 17: $'\r': command not found
   执行： vim build.sh 编辑该文件
   输入 :set fileformat=unix 设置该文件格式
   输入 :wq 保存
   再次运行即可
3. log 文件不存在：./logs/sentinel.log: No such file or directory
   在当前目录新建该文件夹：mkdir logs

启动过程中，可以看到控制台输出：-Dproject.name=sentinel-dashboard -Dcsp.sentinel.log.dir=./logs -Dserver.port=8080 -Dlogging.file=./logs -Djava.security.egd=file:/dev/./urandom -Dcsp.sentinel.dashboard.server=localhost:8080

启动成功后，使用 jps 命令，可以查看到当前运行着的 java 程序，包含 sentinel-dashboard.jar 这么一项。

同时，可以在浏览器访问 <ip>/8080 进入到 sentinel 控制台，输入账号密码（sentinel/sentinel）完成登陆。

如：



五、nacos

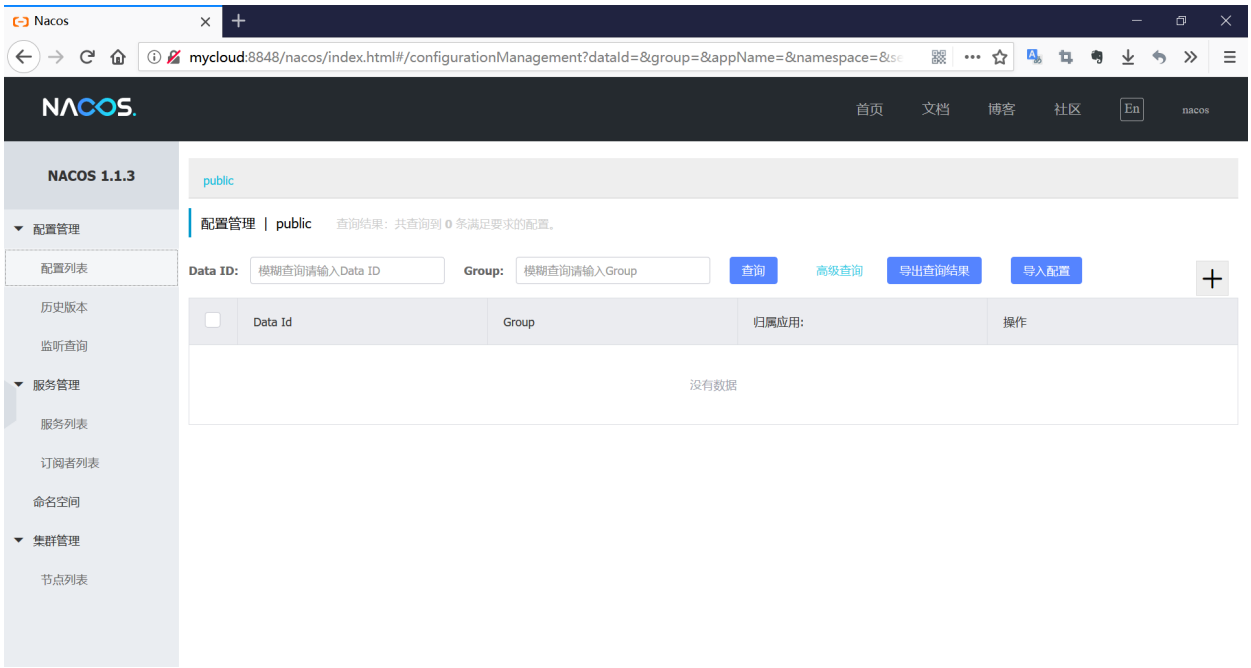
修改好了 standalone-mysql-study.yaml 中关于 mysql 的主从数据库连接之后，执行命令：docker-compose -f standalone-mysql-study.yaml start

提示错误信息：

1. 没有对应的容器，ERROR: No containers to start
   执行 docker-compose -f standalone-mysql-study.yaml up -d 对容器进行构建（会后台执行）
   如果用 ps 命令检查到服务没有启动，可以执行 start 命令再看
2. 启动之后，无法连接上...
   执行命令 ls --full-time 找到最新的 log，查看后发现错误信息。为了便于查看，用 psftp 工具把日志下载到本地。
   删除旧日志：rm -rf standalone-logs
   快速下载的命令：get -r /root/docker/nacos-docker/example/standalone-logs D:\Downloads\cloud\standalone-logs
3. 数据库无法连接：Failed to obtain JDBC Connection; message from server: "Host '182.138.102.60' is blocked because of many connection errors
   之前配置的时候，主数据库和从数据库是配置的一样的，修改成不一样即没有这个报错了
4. 数据库无法连接：Caused by: java.lang.NullPointerException
   at com.mysql.jdbc.ConnectionImpl.getServerCharset(ConnectionImpl.java:2983)
   原因是 SQL 8.0 有一些不兼容.....【至此，重新在 docker 中安装对应版本的 mysql】
5. 服务未启动：java.lang.RuntimeException: Nacos Server did not start because dumpservice bean construction failure : No DataSource set
   重新安装了 docker 的 mysql 数据库之后，先删除这三个服务，再重新构建，启动...
6. 问题总结的一些参考：MySQL8.0连接问题总结

使用 docker-compose -f standalone-mysql-study.yaml ps 可以查看当前服务的状态。
如果要删除该服务，可以使用 docker-compose -f standalone-mysql-study.yaml rm 命令。

那么，访问 <ip>:8848 就可以看到页面：



启动之后呢：

- 配置管理，表示配置中心
- 服务管理，表示 eureka 的服务注册中心

现在，没有配置，这儿就需要上传对应的配置，依次上传对应配置即可。

六、ES

在 docker 中添加配置文件：
vi /root/docker/elasticsearch/config/elasticsearch.yml

```
cluster.name: "docker-cluster"
network.host: 0.0.0.0
discovery.zen.minimum_master_nodes: 1

#设置HTTP开启状态
http.enabled: true

#设置运行跨域访问
http.cors.enabled: true
http.cors.allow-origin: "*"
http.max_content_length: 500mb
```

docker 部署：docker run -d --name es -p 9200:9200 -p 9300:9300 -e "discovery.type=single-node" -e "ES_JAVA_OPTS=-Xms512m -Xmx512m" -v /c/Users/leish/wslHome/docker/elasticsearch/data:/usr/share/elasticsearch/data -v /c/Users/leish/wslHome/docker/elasticsearch/config/elasticsearch.yml:/usr/share/elasticsearch/config/elasticsearch.yml elasticsearch:6.4.0

提示错误信息：

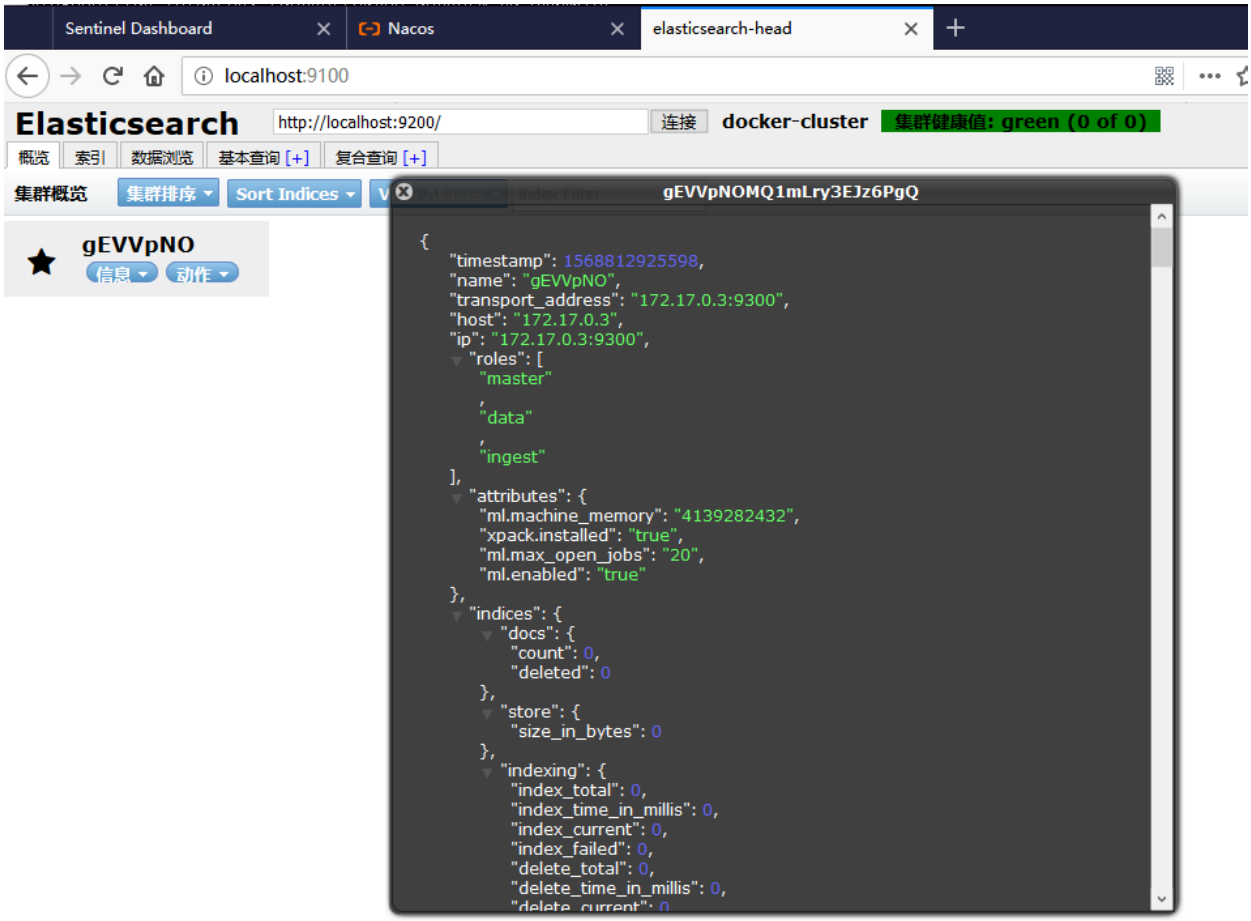
1. docker: Error response from daemon: OCI runtime create failed: container_linux.go:344: starting container process caused "process_linux.go:424: container init caused \"rootfs_linux.go:58: mounting \\\"/root/docker/elasticsearch/config/elasticsearch.yml\\\" to rootfs \\\"/var/lib/docker/overlay2/32e39474975082a6887133f07f4c31200411a226ffc46c3ebec3925441fc5056/merged\\\" at \\\"/var/lib/docker/overlay2/32e39474975082a6887133f07f4c31200411a226ffc46c3ebec3925441fc5056/merged/usr/share/elasticsearch/config/elasticsearch.y

caused \\\"not a directory\\\"\"": unknown: Are you trying to mount a directory onto a file (or vice-versa)? Check if the specified host path exists and is the expected type.
这儿想要把文件挂载出来，结果无法成功挂出来...把命令中对应关于文件的删除了。

docker 部署 es 插件：docker run -d --name es_head -p 9100:9100 mobz/elasticsearch-head:5

在 9100 端口即可访问~

这儿先不安装中文分词器，后续继续来..



七、Zipkin

使用docker-compose方式启动zipkin，由于上面我们部署过es，es也不直接默认把数据存在mysql性能要好，所以zipkin也把数据存到es中。

嗯，把 docker-compose 包上传到服务器，再一句命令：docker-compose -f docker-compose-elasticsearch.yml up -d
注意要修改 docker-compose-elasticsearch.yml 中的链接参数，可以是容器的主机的 ip 地址

提示错误信息：

1. 启动后很快又停止服务了：
   Exception encountered during context initialization - cancelling refresh attempt: org.springframework.beans.factory.BeanCreationException: Error creating bean with name 'rabbitMq' defined in zipkin2.server.internal.rabbitmq.ZipkinRabbitMQCollectorConfiguration: Invocation of init method failed; nested exception is java.io.UncheckedIOException: Unable to establish connection to RabbitMQ server: connect timed out
   分析原因可能是数据库配置不对，在容器中使用了 localhost，可能访问不到宿主机。
   于是使用 ip addr 看到当前的网卡信息，发现 10.0.75.1 即是宿主机 ip 地址；也可以直接在系统网卡中查看...
   配置到 docker-compose-elasticsearch.yml 中，删除容器，重新 up 之后，即完成，控制台输出：Started ZipkinServer in 4.395 seconds (JVM running for 5.7)

八、完成

至此，基本环境中的所有内容即部署完毕。如果要看容器日志，可以：docker logs -f naos-standalone-mysql

在 docker for windows (本地) 的 docker 中：

```
ting@LEITING-THINKPAD:/c/Users/leish/wslHome/docker/docker-zipkin$ docker ps
CONTAINER
ID        IMAGE                                COMMAND                  CREATED             STATUS              PORTS
b7275022d49d        mobz/elasticsearch-head:5          "/bin/sh -c 'grunt s…"   8 minutes ago       Up 8 minutes        0.0.0.0:9100-
>9100/tcp                                                                                                       es_head
584d2973dab0        elasticsearch:6.4.0                "/usr/local/bin/dock…"   9 minutes ago       Up 9 minutes        0.0.0.0:9200-
>9200/tcp, 0.0.0.0:9300->9300/tcp                                                                               es
5c566eaff7af        redis                              "docker-entrypoint.s…"   22 minutes ago      Up 22 minutes       0.0.0.0:6379-
>6379/tcp                                                                                                       redis
572eee06819e        prom/prometheus:latest             "/bin/prometheus --c…"   24 minutes ago      Up 24 minutes       0.0.0.0:9090-
>9090/tcp                                                                                                       prometheus
b444606c2f38        nacos/nacos-server:latest          "bin/docker-startup.…"   24 minutes ago      Up 24 minutes       0.0.0.0:8848-
>8848/tcp, 0.0.0.0:9555->9555/tcp                                                                               nacos-standalone-mysql
83b015446b87        grafana/grafana:latest             "/run.sh"                25 minutes ago      Up 25 minutes       0.0.0.0:3000-
>3000/tcp                                                                                                       grafana
8ea51ca03aa1        xiaochunping/rabbitmq:management   "docker-entrypoint.s…"   5 hours ago         Up About an hour    0.0.0.0:4369-
>4369/tcp, 0.0.0.0:5671-5672->5671-5672/tcp, 0.0.0.0:15670-15674->15670-15674/tcp, 0.0.0.0:25672->25672/tcp
```

在远程的云主机中：

```
[root@mycloud ~]# docker ps
CONTAINER
ID        IMAGE                        COMMAND                  CREATED             STATUS              PORTS                     NAMES
46ca794cbdc3        xiaochunping/mysql-slave    "docker-entrypoint.s…"   9 hours ago         Up 54 seconds       0.0.0.0:3307-
>3306/tcp   mysql-slave
dbf72cc0c8f8        xiaochunping/mysql-master   "docker-entrypoint.s…"   9 hours ago         Up 9 hours          0.0.0.0:3306-
>3306/tcp   mysql-master
```

idea 中使用 Docker 插件也可以很方便查看到各个容器的信息，如图：

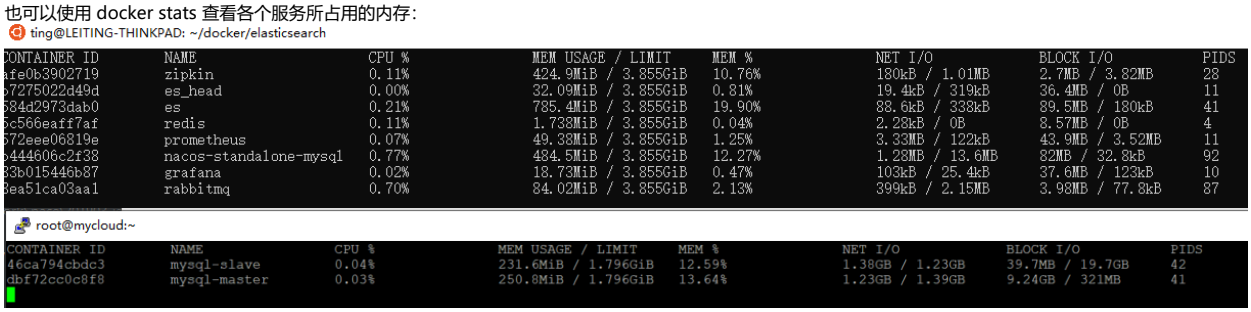
可用的服务汇总：

1. redis：6379
2. sentinel：http://localhost:8080/#/dashboard/degrade/sentinel-dashboard
3. nacos：http://localhost:8848/nacos/#/serviceManagement?dataId=&group=&appName=&namespace=
4. es监控：http://localhost:9100/
5. es：9200
6. zipkin：http://localhost:9411/zipkin/
7. mysql：cloud:3306
8. rabbitMQ：http://localhost:15672

如果重启操作系统后，可以首先查看各个容器的状态，docker ps -a，然后 docker restart id1 id2 id3 重启各个容器。其中 sentinel 是要执行 sh 命令...

```
## 重启所有
docker restart 8ea51ca03aa1 b7275022d49d 584d2973dab0 5c566eaff7af 572eee06819e b444606c2f38 83b015446b87 afe0b3902719
```

也可以使用 docker stats 查看各个服务所占用的内存：



两个数据库服务占用 600MB
其余的组件占用接近 2GB

我的云服务器只有 2GB，后面重新安装了 Docker For Windows 配合 Win 10 的 ubunt 子系统使用。具体参考： 2019年9月18日 - windows 10 上安装 docker，配合 wsl 使用