架构师之路三 - 系统可用性

作者: jianzh5

- 原文链接: https://ld246.com/article/1586248002000
- 来源网站:链滴
- 许可协议:署名-相同方式共享 4.0 国际 (CC BY-SA 4.0)

SpringCloud Alibaba系列文章已经写了16篇了,基本框架大体完成,业务相关的逻辑还需要根据项本身的业务进行梳理改造。

今天将是本系列的最后一篇 - SpringCloud容器化部署(理论上最后一篇,不排除后面会对相关组件行升级)

看在写了这么多的份上是不是应该点个在看呢?



开启Docker远程访问

由于我是在windows上进行开发没有安装docker,所以需要找一台安装好docker的服务器并开启远访问。使用mac的同学请忽略。

- 打开docker配置文件
- vi /lib/systemd/system/docker.service
- 开放2376端口

找到ExecStart=/usr/bin/dockerd所在行,在后面追加-H tcp://0.0.0.0:2376 -H unix:///var/run/doker.sock,修改完成的效果如下:

ExecStart=/usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock -H tcp://0 0.0.0:2376 -H unix:///var/run/docker.sock

● 重启docker服务

systemctl daemon-reload systemctl restart docker.service

● 使用netstat查看端口

netstat -nptl

[root@b:	ingo-172	system]# netstat -npt	1		
Active :	Internet	connections (only ser	vers)		
Proto R	ecv-Q Se	nd-Q Local Address	Foreign Address	State	PID/Program name
tcp	Θ	0 127.0.0.1:25	0.0.0.0:*	LISTEN	1619/master
tcp	Θ	0 0.0.0.0:22	0.0.0.0:*	LISTEN	1686/sshd
tcp6	Θ	0 ::1:25	:::*	LISTEN	1619/master
tcp6	Θ	0 :::9600	:::*	LISTEN	21620/docker-proxy
tcp6	Θ	0 :::5601	:::*	LISTEN	21592/docker-proxy
tcp6	Θ	0 :::2376	:::*	LISTEN	21415/dockerd
tcp6	Θ	0 :::9100	:::*	LISTEN	21683/docker-proxy
tcp6	Θ	0 :::9200	:::*	LISTEN	21578/docker-proxy
tcp6	Θ	0 :::5044	:::*	LISTEN	21637/docker-proxy
tcp6	Θ	0 :::9300	:::*	LISTEN	21566/docker-proxy
tcp6	Θ	0 :::22	:::*	LISTEN	1686/sshd

● 访问/info, 确定端口正常开放



docker-maven-plugin 构建 docker镜像

在开始打包之前一定要先在bom和common模块执行mvn clean install 命令, 否则打包不成功。

看过本系列文章的人一定也知道docker 和 docker-compose的相关指令了, 这里就不再说明。



优秀的同学, 绝对是优秀的同学

• 在需要构建组件的模块引入docker-maven-plugin插件

```
<plugin>
```

```
<groupId>com.spotify</groupId>
  <artifactId>docker-maven-plugin</artifactId>
  <version>1.2.2</version>
  <configuration>
    <imageName>${project.artifactId}</imageName>
    <imageTags>
       <imageTag>latest</imageTag>
    </imageTags>
    <!--指定Dockerfile路径-->
    <dockerDirectory>${project.basedir}/src/main/docker</dockerDirectory>
    <dockerHost>http://xxx.xx.xx:2376</dockerHost>
    <resources>
       <resource>
         <targetPath>/</targetPath>
         <!--${project.basedir}/target-->
         <directory>${project.build.directory}</directory>
         <!--${project.artifactId}-${project.version}-->
         <include>${project.build.finalName}.jar</include>
       </resource>
    </resources>
  </configuration>
</plugin>
```

这里端口与前面开放的端口保持一致。

● 在模块的 src/main/docker 目录下建立Dockerfile文件

FROM openjdk:8-jdk-alpine VOLUME /tmp ADD cloud-gateway-1.0.0.jar app.jar RUN sh -c 'touch /app.jar' ENTRYPOINT ["java","-Djava.security.egd=file:/dev/./urandom","-jar","/app.jar"]

每个模块需要自行修改 ADD 指令

● 在模块下执行如下命令构建docker镜像

mvn clean package docker:build -DskipTests

执行效果如下:

[INFO] Building image auth-service Step 1/5 : FROM openjdk:8-jdk-alpine

Pulling from	library/openj	dk				
e7c96db7181b:	Pulling fs l	ayer				
f910a506b6cb:	Pulling fs l	ayer				
c2274a1a0e27:	Pulling fs l	ayer				
f910a506b6cb:	Downloading	[======================================	>]	238B/238B		
f910a506b6cb:	: Verifying Checksum					
f910a506b6cb:	Download com	plete				
e7c96db7181b:	Downloading	[>	1	32.63kB/2.757MB		
e7c96db7181b:	Downloading	[======>	1	534.2kB/2.757MB		
e7c96db7181b:	Downloading	[======>	1	566.1kB/2.757MB		
e7c96db7181b:	Downloading	[======>	1	600.8kB/2.757MB		
e7c96db7181b:	Downloading	[=======>	1	705.1kB/2.757MB		
e7c96db7181b:	Downloading	[========>	1	736.9kB/2.757MB		
c2274a1a0e27:	Downloading	[>	1	528.5kB/70.73MB		
e7c96db7181b:	Downloading	[========>	1	768.8kB/2.757MB		
e7c96db7181b:	Downloading	[=========>	1	835.4kB/2.757MB		
e7c96db7181b:	Downloading	[=====>	1	865.8kB/2.757MB		
e7c96db7181b:	Downloading	[===========>	1	957kB/2.757MB		
e7c96db7181b:	Downloading	[===========>	1	987.4kB/2.757MB		
e7c96db7181b:	Downloading	[======>	1	1.09MB/2.757MB		
e7c96db7181b:	Downloading	[=================>	1	1.119MB/2.757MB		

● 构建完成后登陆服务器查看docker 镜像

docker images

[root@bingo-172 ~]# doc	ker images			
REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
cloud-gateway	latest	8f1cca46eddf	31 seconds ago	221MB
account-service	latest	36b9c0f967e2	3 minutes ago	245MB
auth-service	latest	c2ee694ae3aa	4 hours ago	159MB
		则必必留		

● 启动镜像,带上 --rm 指令便于删除容器。

docker run -d -p 5000:5000 --rm auth-service docker run -d -p 8010:8010 --rm account-service docker run -d -p 8090:8090 --rm cloud-gateway

● 查看是否正常启动

[root@bingo-1/2 ~	i# docker ps					
CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
6b3d9e795444	cloud-gateway	"java -Djava.securit_"	2 seconds ago	Up 2 seconds	0.0.0.0:8090->8090/tcp	cocky_elion
02be2efb4f64	account-service	"java -Djava.securit"	33 seconds ago	Up 33 seconds	0.0.0.0:8010->8010/tcp	romantic_babbage
clddcc2c5b67	auth-service	"java -Djava.securit"	About a minute ago	Up About a minute	0.0.0.0:5000->5000/tcp	keen_noether

● 使用postman进行测试

		h =	GET 🗸
		eaders (1) Body Pre-request Script Tests	thorization
ic5-b0e590a30d58	Bearer 1af4ae4a-be53-44fa-bdc5-	1	Authorizati
	value		key
		Preview JSON >>	Pretty Raw 1 + [] "status 3 "messag - "data": 5 "id": - "acco 7 "acco 8 "amou 9], 10 "succes
		ıp": 1585213972876	11 "timest 12 }

服务正常响应!

使用docker-compose启动

● 建立 cloud-service.yml 文件编排镜像 version: "3" services: auth-service: container name: auth-service image: auth-service:latest ports: - "5000:5000" restart: always cloud-gateway: container_name: cloud-gateway image: cloud-gateway:latest ports: - "8090:8090" restart: always account-service: container name: account-service image: account-service:latest ports: - "8010:8010" restart: always

• 将文件上传至服务器,使用如下脚本启动服务

docker-compose -f cloud-service up

(rotgoingo 1/2 app)# docker-compose -r clud-service.ymclup Creating network app default with the default driver Creating account-service diver	
Creating Guod-gateway down Creating Cloud-gateway down Attaching to auth-service, account-service, cloud-gateway main] trationDelegatesBeanPostProcessorChecker : Bean 'org.springf cloud.autoconfigure.ConfigureTionPropertiesBeanPostProcessorChecker : Bean 'org.springf main factors / 2020-03-26 09;27:19-592 INFO 1 [main] trationDelegatesBeanPostProcessorChecker : Bean 'org.springf	framework.cloud.autoconfigure.ConfigurationPropertiesRebinderAutoConfiguration' o scessed by all BeamPostProcessors (for example: not eligible for auto-proxying) remework.cloud.autoconfigure.configuration?
<pre>.cloud.autoconfigure.ConfigurationPropertiesRebinderAutoConfiguration\$SEnhancerBySpringCGLIB\$\$a17eb50a] is not eligible for getting pro auth-service auth-se</pre>	cessed by all BeanPostProcessors (for example: not eligible for auto-proxying)
All of pickey All	e set, falling back to default profiles: default
account-service ///// account-service ///// account-service ////// account-service ////// account-service /////// account-service //////// account-service //////// account-service //////// account-service //////// account-service //////// account-service /////// account-service //////// account-service //////// account-service //////// account-service ////////////////////////////////////	memork.cloud.autocontigure.contigurationroperiiesmedinderAutocontiguration ar ssed by all BeanPostProcessors (for example: not eligible for auto-proxying)
GET V /account-service/account/getByCode/jianzh5	
Authorization Headers (1) Body Pre-request Script Tests	Bearer 1af4ae4a-be53-44fa-bdc5-b0e590
key	value
Body Cookies Headers (10) Tests	
Pretty Raw Preview JSON V	
<pre>1 ~ { 2 "status": 100, 3 "message": "请求成功", 4 ~ "data": { 5 "id": 17, 6 "accountCode": "jianzh5", 7 "accountName": "jianzh5", 8 "amount": 89 9 }, 10 "success": true, 11 "timestamp": 1585215176728 12 } </pre>	

服务正常响应!

SpringCloud的容器化部署还是比较简单的,你也来试试好了!