Dockerfile 结合 dockercompose 搭建 C++微服务

版权说明

对比特项目感兴趣，可以联系这个微信。

实战目的

之前我们构建的 C++都是运行完退出的应用程序，我们构建一个长时间运行的 C++服务来感受下。

实战步骤

1. 准备目录

```
C++
mkdir -p /data/maxhou/mydockerfile/mycppms
mkdir -p /data/maxhou/mydockerfile/mycppms/cppweb
mkdir -p /data/maxhou/mydockerfile/mycppms/nginx
```

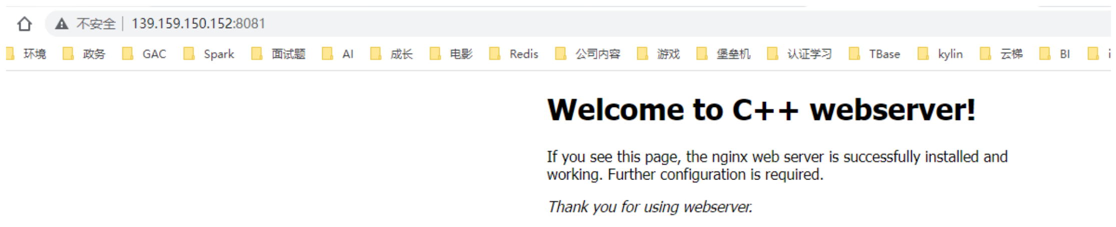
2. 进入目录 cd /data/maxhou/mydockerfile/mycppms/cppweb，编写源代码 main.cpp

```
C++
#include <iostream>
#include <netinet/in.h> //sockaddr_in 结构体头文件
#include <string.h> //memset()头文件
#include <assert.h> //assert()头文件
#include <unistd.h> //close()头文件
#include <pthread.h>


struct pthread_data{
    struct sockaddr_in client_addr;
    int sock_fd;
};

using namespace  std;

void *serverForClient(void *arg);

int main(){
    int socket_fd;
    int conn_fd;
    int res;
    int len;

    struct sockaddr_in sever_add;
    memset(&sever_add,0,sizeof(sever_add)); //初始化
    sever_add.sin_family = PF_INET;
    sever_add.sin_addr.s_addr = htons(INADDR_ANY);
    sever_add.sin_port = htons(8081);
    len = sizeof(sever_add);

    //socket()
    int option = 1;
    socket_fd =  socket(AF_INET,SOCK_STREAM,0);
    assert(socket_fd >= 0);
    setsockopt(socket_fd, SOL_SOCKET, SO_REUSEADDR, &option,
sizeof(option));

    //bind()
    res = bind(socket_fd,(struct sockaddr*)&sever_add,len);
    perror("bind");
    assert(res != -1);

    //listen()
    res = listen(socket_fd,1);
    assert(res != -1);
    cout<<"server init"<<endl;

    while(1){
        struct sockaddr_in client;
        int client_len = sizeof(client);
        //accept()
        conn_fd = accept(socket_fd,(struct
sockaddr*)&client,(socklen_t *)&client_len);

        pthread_data pdata;
        pthread_t pt;
        pdata.client_addr = client;
        pdata.sock_fd = conn_fd;
        std::cout<<"in "<<conn_fd<<endl;
        pthread_create(&pt, NULL, serverForClient, (void
*)&pdata);
    }

    return 0;
}


void *serverForClient(void *arg){
    struct pthread_data *pdata = (struct pthread_data*)arg;
    int conn_fd = pdata->sock_fd;

    std::cout<<"process "<<conn_fd<<endl;

    if(conn_fd < 0) cout << "error" << endl;
    else{
        char request[1024];
        int len = recv(conn_fd,request,1024,0);

        if(len <= 0){
            close(conn_fd);
            return nullptr;
        }

        request[strlen(request) + 1] = '\0';
        char buf[520] = "HTTP/1.1 200 ok\r\nconnection:
close\r\n\r\n";//HTTP 响应
        int s = send(conn_fd,buf,strlen(buf),0);//发送响应
        if(s <=0 ){
            perror("send");
            return nullptr;
        } else{
            char buf2[1024] = "\n"
                              "<!DOCTYPE html>\n"
                              "<html>\n"
                              "<head>\n"
                              "<title>Welcome to C++ web
server!</title>\n"
                              "<style>\n"
                              "html { color-scheme: light
dark; }\n"
                              "body { width: 35em; margin: 0
auto;\n"
                              "font-family: Tahoma, Verdana,
Arial, sans-serif; }\n"
                              "</style>\n"
                              "</head>\n"
                              "<body>\n"
                              "<h1>Welcome to C++
webserver!</h1>\n"
                              "<p>If you see this page, the nginx
web server is successfully installed and\n"
                              "working. Further configuration is
required.</p>\n"
                              "\n"
                              "<p><em>Thank you for using
webserver.</em></p>\n"
                              "</body>\n"
                              "</html>";
            int s2 = send(conn_fd,buf2,strlen(buf2),0);
            if(s2<=0){
                perror("send");
                return nullptr;
            }
            //发送响应
            close(conn_fd);
        }

    }
    return nullptr;
}
```

3. 在宿主机上构建

```
C++
root@139-159-150-152:/data/maxhou/mydockerfile/mycppms# g++
main.cpp -o mycppweb -lpthread -std=c++11
root@139-159-150-152:/data/maxhou/mydockerfile/mycppms# ll
total 32
drwxr-xr-x  2 root root  4096 May 19 18:51 ./
drwxr-xr-x 14 root root  4096 May 19 17:42 ../
-rw-r--r--  1 root root  3557 May 19 18:50 main.cpp
-rwxr-xr-x  1 root root 18088 May 19 18:51 mycppweb*
```

4. 启动服务，测试服务是否能够正常运行



Welcome to C++ webserver!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

Thank you for using webserver.

查看控制台

```
C++
root@139-159-150-152:/data/maxhou/mydockerfile/mycppms# ./mycppweb
bind: Success
server init
in 4
process 4
```

5. 编写 c++应用的 Dockerfile

```C++
FROM centos:7 as buildstage
RUN  sed -e 's|^mirrorlist=|#mirrorlist=|g' \
         -e
's|^#baseurl=http://mirror.centos.org/centos|baseurl=https://mirro
rs.ustc.edu.cn/centos|g' \
         -i.bak \
         /etc/yum.repos.d/CentOS-Base.repo
RUN yum install -y centos-release-scl
RUN yum makecache
RUN yum install -y devtoolset-9-gcc devtoolset-9-gcc-c++
devtoolset-9-binutils make
WORKDIR /src
COPY ./main.cpp .
RUN source /opt/rh/devtoolset-9/enable&& g++ main.cpp -o mycppweb
-lpthread
CMD ["/src/mycppweb"]

FROM centos:7
COPY --from=buildstage /src/mycppweb /
CMD ["/mycppweb"]
```

6. 进入目录/data/maxhou/mydockerfile/mycppms/nginx 编写配置文件 bit.conf

```C++
upstream  backend {
        server mycppweb:8081 weight=1;
        server mycppweb2:8081 weight=2;
}

server {
    listen 80;
    access_log off;
    location / {
        proxy_pass http://backend;
    }
}
```

7. 编写 nginx 的 Dockerfile

```
C++
FROM nginx:1.24.0
COPY ./bit.conf /etc/nginx/conf.d/
CMD ["nginx","-g","daemon off;"]
ENTRYPOINT ["/docker-entrypoint.sh"]
```

8. 进入目录/data/maxhou/mydockerfile/mycppms 编写 docker-compose.yml

```
C++
services:
  web:
    image: mynginx:v3.0
    build:
      context: ./nginx
    ports:
      - 8112:80
    depends_on:
      mycppweb:
          condition: service_started
  mycppweb:
    build:
      context: ./cppweb
    image: mycppweb:v2.0
  mycppweb2:
    image: mycppweb:v2.0
```

9. 构建镜像

```
C++
root@139-159-150-152:/data/maxhou/mydockerfile/mycppms# docker
compose build
[+] Building 0.1s (14/14) FINISHED
 => [internal] load build definition from Dockerfile
0.1s
 => => transferring dockerfile: 620B
0.0s
```

```
 => [internal] load .dockerignore
0.0s
 => => transferring context: 2B
0.0s
 => [internal] load metadata for docker.io/library/centos:7
0.0s
 => [buildstage 1/8] FROM docker.io/library/centos:7
0.0s
 => [internal] load build context
0.0s
 => => transferring context: 30B
0.0s
 => CACHED [buildstage 2/8] RUN  sed -e
's|^mirrorlist=|#mirrorlist=|g'          -e
's|^#baseurl=http://mirror.centos.org/centos|baseurl=https://mirro
rs.ustc.edu.c  0.0s
 => CACHED [buildstage 3/8] RUN yum install -y centos-release-scl
0.0s
 => CACHED [buildstage 4/8] RUN yum makecache
0.0s
 => CACHED [buildstage 5/8] RUN yum install -y devtoolset-9-gcc
devtoolset-9-gcc-c++ devtoolset-9-binutils make
0.0s
 => CACHED [buildstage 6/8] WORKDIR /src
0.0s
 => CACHED [buildstage 7/8] COPY ./main.cpp .
0.0s
 => CACHED [buildstage 8/8] RUN source /opt/rh/devtoolset-
9/enable&& g++ main.cpp -o mycppweb -lpthread
0.0s
 => CACHED [stage-1 2/2] COPY --from=buildstage /src/mycppweb /
0.0s
 => exporting to image
0.0s
 => => exporting layers
0.0s
 => => writing image
sha256:1da2046c40720555aefcf96026ee17c073b47a9e8740b6c009e61d06e54
a71dd
0.0s
 => => naming to docker.io/library/mycppweb:v2.0
0.0s
[+] Building 0.2s (7/7) FINISHED
 => [internal] load build definition from Dockerfile
```

```
0.1s
 => => transferring dockerfile: 160B
0.1s
 => [internal] load .dockerignore
0.0s
 => => transferring context: 2B
0.0s
 => [internal] load metadata for docker.io/library/nginx:1.24.0
0.0s
 => [internal] load build context
0.0s
 => => transferring context: 245B
0.0s
 => CACHED [1/2] FROM docker.io/library/nginx:1.24.0
0.0s
 => [2/2] COPY ./bit.conf /etc/nginx/conf.d/
0.1s
 => exporting to image
0.0s
 => => exporting layers
0.0s
 => => writing image
sha256:12a8486212281c0eea6fdbf2bcbf57abe105aa5ad59c653d001200f5f15
7da43
0.0s
 => => naming to docker.io/library/mynginx:v3.0
0.0s
```

10. 启动服务

```C++
root@139-159-150-152:/data/maxhou/mydockerfile/mycppms# docker
compose up -d
[+] Running 4/4
 ✔ Network mycppms_default        Created
0.1s
 ✔ Container mycppms-mycppweb2-1  Started
1.1s
 ✔ Container mycppms-mycppweb-1   Started
1.1s
 ✔ Container mycppms-web-1        Started
1.3s
```

11. 访问服务 6 次可以看到按照权重负载均衡到了 2 个服务上面

```C++
root@139-159-150-152:/data/maxhou/mydockerfile/mycppms# docker
logs -f mycppms-mycppweb-1
server init
bind: Success
in 4
process 4
in 5
process 5


root@139-159-150-152:/data/maxhou/mydockerfile/mycppms# docker
logs -f  mycppms-mycppweb2-1
bind: Success
server init
in 4
process 4
in 5
process 5
in 4
process 4
in 5
process 5
in 4
process 4
```

12. 清理资源

```C++
root@139-159-150-152:/data/maxhou/mydockerfile/mycppms# docker
compose down
[+] Running 4/4
 ✔ Container mycppms-mycppweb2-1  Removed
10.2s
 ✔ Container mycppms-web-1        Removed
0.2s
 ✔ Container mycppms-mycppweb-1   Removed
10.1s
 ✔ Network mycppms_default        Removed
0.1s
```