

# docker镜像打包及上传指南

---

## docker镜像打包及上传指南

1. 背景
  - 1.1 War包
  - 1.2 Tomcat服务器
  - 1.3 修改Ubuntu的apt源为国内镜像源
    - 1.3.1 修改/etc/apt/sources.list文件的方式修改源（不建议）
    - 1.3.2 图形界面修改（建议）
  - 1.4 安装翻墙软件clash for windows
  - 1.5 在Ubuntu操作系统上安装Docker社区版
2. 实验:
  - 2.1 打包制作docker镜像
  - 2.2 docker镜像上传docker hub

## 1. 背景

---

### 1.1 War包

War包一般是在进行Web开发时，通常是一个网站Project下的所有源码的集合，里面包含前台HTML/CSS/JS的代码，也包含Java的代码。

当开发人员在自己的开发机器上调试所有代码并通过后，为了交给测试人员测试和未来进行产品发布，都需要将开发人员的源码打包成War进行发布。

War包可以放在Tomcat下的webapps或者word目录下，随着tomcat服务器的启动，它可以自动被解压。简单理解就是一个web项目，其中包含web的所有东西。

### 1.2 Tomcat服务器

Tomcat服务器是一个免费的开放源代码的Web应用服务器，属于轻量级应用服务器，在中小型系统和并发访问用户不是很多的场合下被普遍使用，是开发和调试JSP程序的首选，最新的Servlet和JSP规范总是能在Tomcat中得到体现。

### 1.3 修改Ubuntu的apt源为国内镜像源

由于Ubuntu软件源默认下载服务器为外网，下载速度较慢，因此常常需要修改下载源，修改源有图形 界面修改和文件修改两种方式，且二者相通，即采用一种方式修改后，另一方也随之发生改变，因此不必纠结采用哪种方法。下面简单介绍：

#### 1.3.1 修改/etc/apt/sources.list文件的方式修改源（不建议）

#### 1.3.2 图形界面修改（建议）

在右上角的设置中，选择System Settings，进入到Software&Updates。

![[Pasted image 20240227221255.png]]

中文![[Pasted image 20240227221306.png]]

英文

在这个界面里面download from中就可以选择了。建议选择mirros.aliyun.com阿里源。

![[Pasted image 20240227221330.png]]修改完成后点击Choose Server，其实就是选择不同的服务器，在这里可以选择网易，阿里或者清华的服务器。然后系统会提示reload一下可获取的软件源信息，因为更换源后软件清单等也需要随之更新。

![[Pasted image 20240227221348.png]]

## 1.4 安装翻墙软件clash for windows

Clash是翻墙软件。clash for windows是有桌面窗口的翻墙软件。

clash for windows 软件下载链接如下：

- Github上的代码：[https://github.com/Fndroid/clash\\_for\\_windows\\_pkg/releases](https://github.com/Fndroid/clash_for_windows_pkg/releases)

1. 在安装clash for windows软件时，需要根据自己电脑的操作系统内核信息下载对应的版本，使用命令 `uname -a` 查看当前操作系统内核信息。以Ubuntu系统为例，返回结果如下所示：

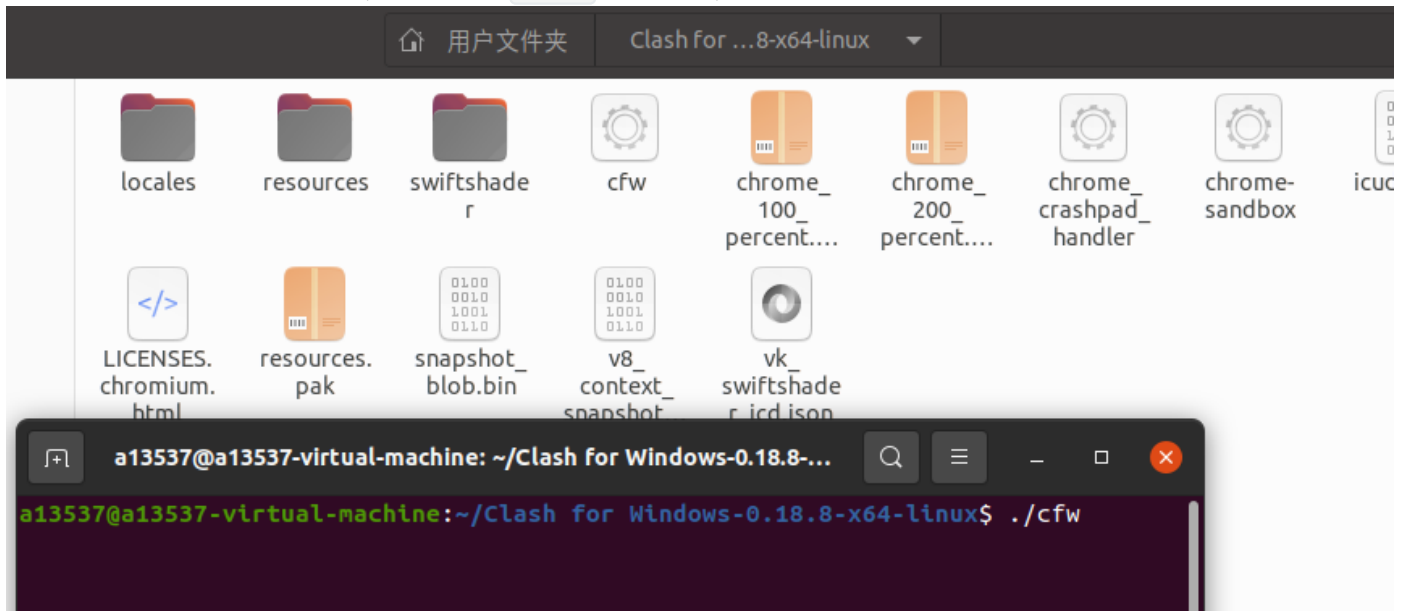
```
$ uname -a
Linux wxg6226-Precision-7920-Tower 5.14.0-1051-oem #58-Ubuntu SMP Fri Aug 26 05:50:00 UTC
2022 x86_64 x86_64 x86_64 GNU/Linux
```

这说明本机是x86系统架构，因此需要下载的clash for windows软件版本是：Clash.for.Windows-0.20.4-x64-linux.tar.gz

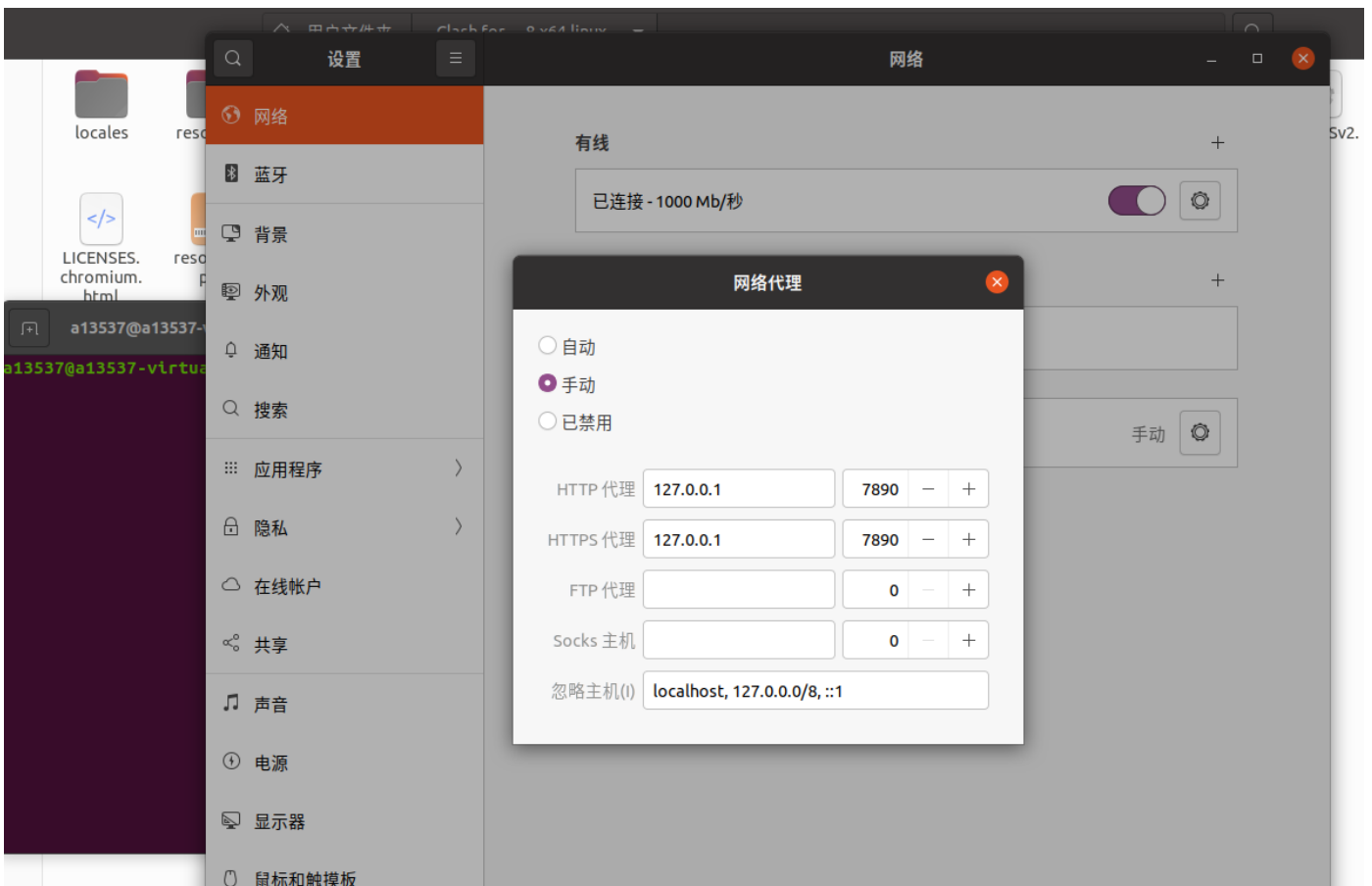
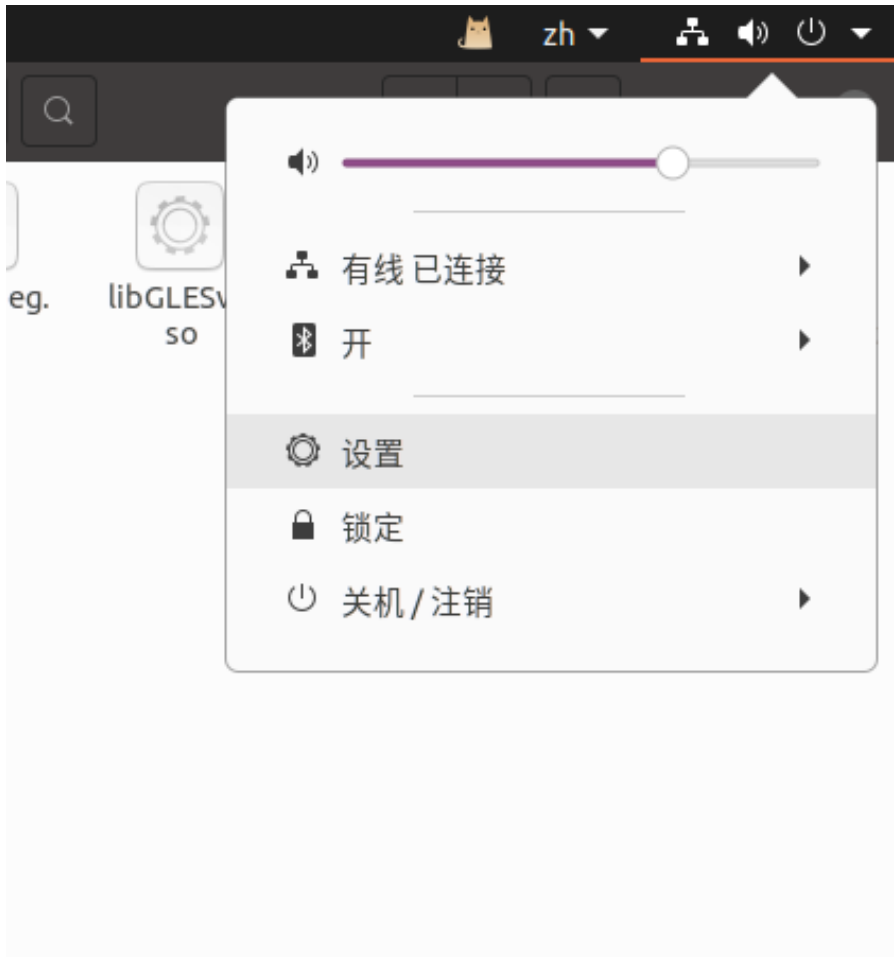
2. 之后对压缩包进行解压，在Linux系统下，使用下面命令进行解压，来维持文件权限：

```
tar xzvf Clash.for.Windows-0.20.4-x64-linux.tar.gz
```

然后在命令行下进入到该目录，使用命令 `./cfw` 进行运行，如下图所示：



3. 设置代理 打开设置 -> 网络 -> 网络代理，将http/https代理指向本机clash默认端口7890（注意：电脑中网络代理的端口号设置需要和clash for windows中的端口号保持一致，否则将出现网络无法连接问题。clash界面启动页可以修改Port）



注意：为防止每次重启clash for windows出现该软件端口号随即更新问题，可以将该软件的端口号固定（点击端口号旁边 箭头循环符号）

↑ 0 B/s

↓ 0 B/s

# Clash for Windows v0.20.4

General

Port

Allow LAN

Log Level info

IPv6

Clash Core 2022.08.26 Premium (34553)

Home Directory [Open Folder](#)

GeoIP Database 2022-09-23 16:34

Service Mode [Manage](#)

TUN Mode

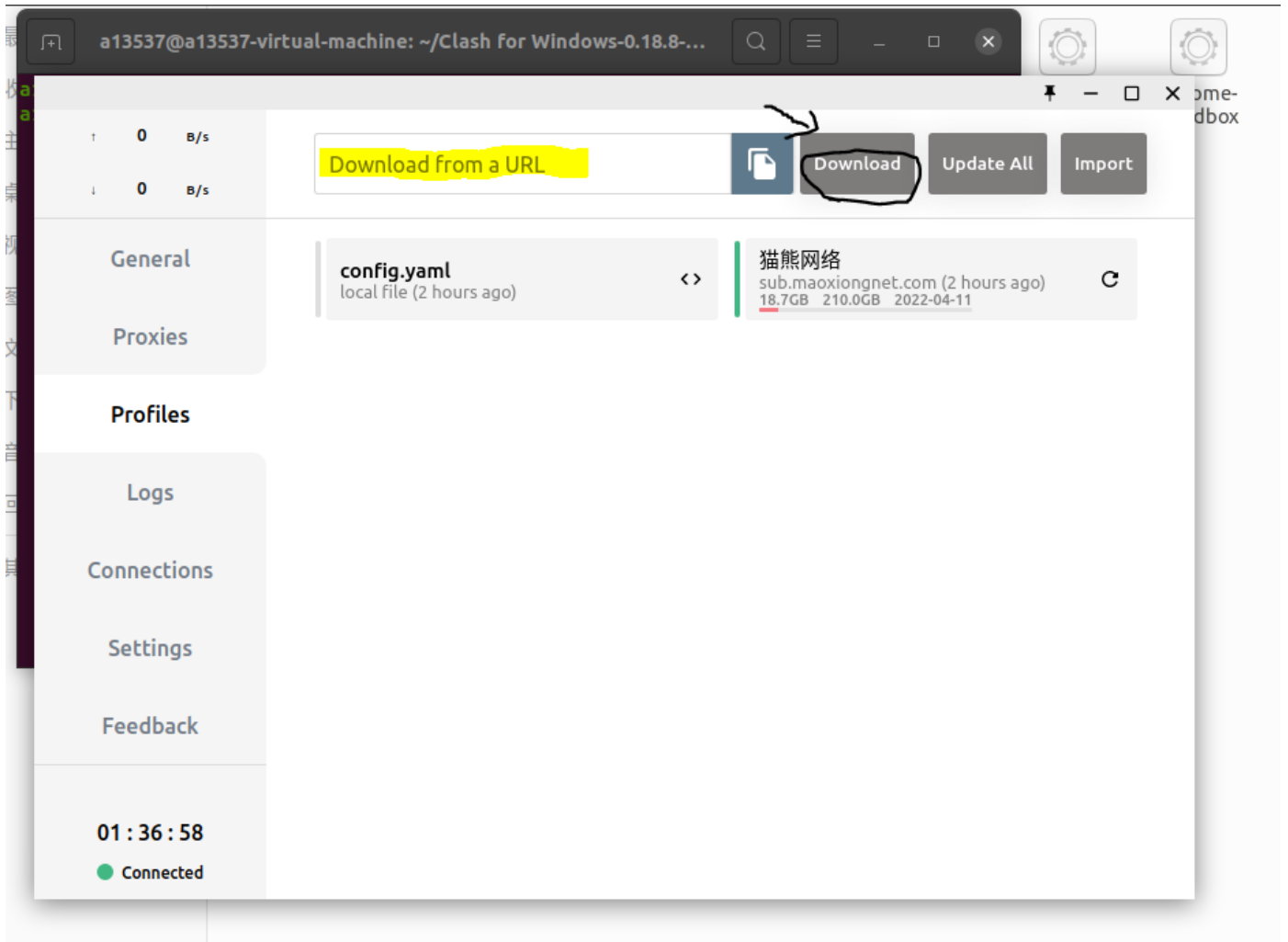
Mixin

Start with Linux

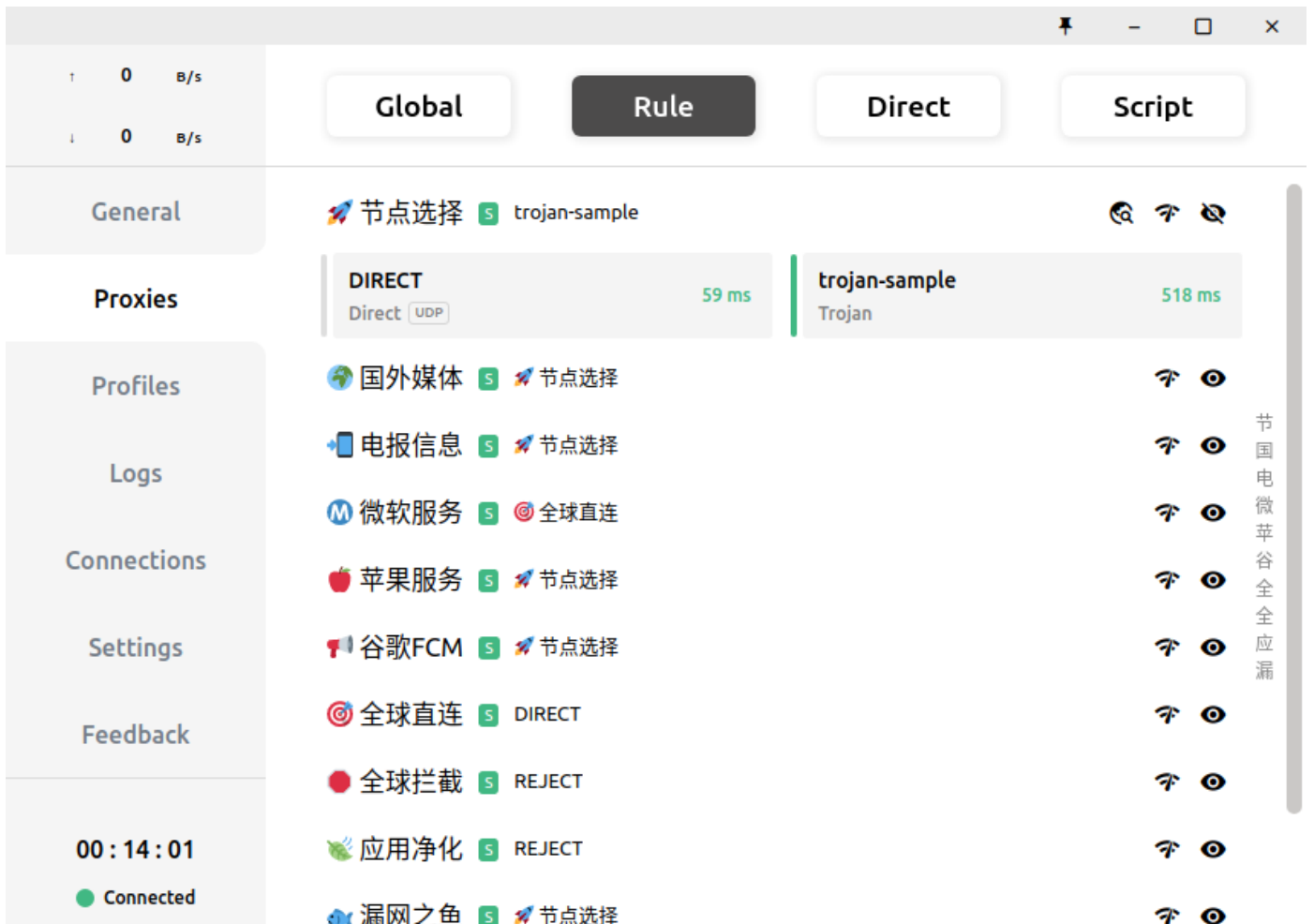
01:13:02

● Connected

4. 设置订阅链接 从代理商服务商处获取订阅链接，填入clash,选好节点即可



5. 设置完成订阅链接之后，需要对clash的代理规则进行设置，选择Rules.



下载之后，对软件进行安装，安装过程参考链接如下：

- 参考知乎：[ubuntu 20.04 下安装使用clash for windows \(最简单版本\)](#)

## 1.5 在Ubuntu操作系统上安装Docker社区版

参考：<https://docs.docker.com/engine/install/ubuntu/>

其他操作系统上的Docker安装，要自己另外查步骤。

前期准备

```
sudo apt-get update

sudo apt-get install \
  apt-transport-https \
  ca-certificates \
  curl \
  gnupg-agent \
  gnupg2 pass\
  software-properties-common

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

sudo apt-key fingerprint 0EBFCD88
```

```
sudo add-apt-repository \  
  "deb [arch=amd64] https://download.docker.com/linux/ubuntu \  
  $(lsb_release -cs) \  
  stable"
```

安装docker engine (community edition) 免费的社区版

```
sudo apt-get update  
sudo apt-get install docker-ce docker-ce-cli containerd.io
```

下面的命令可以列举所有不同版本docker engine (community edition)的引擎

```
$ apt-cache madison docker-ce  
  
docker-ce | 5:20.10.4~3-0~ubuntu-bionic | https://download.docker.com/linux/ubuntu  
bionic/stable amd64 Packages  
docker-ce | 5:20.10.3~3-0~ubuntu-bionic | https://download.docker.com/linux/ubuntu  
bionic/stable amd64 Packages  
docker-ce | 5:20.10.2~3-0~ubuntu-bionic | https://download.docker.com/linux/ubuntu  
bionic/stable amd64 Packages  
docker-ce | 5:20.10.1~3-0~ubuntu-bionic | https://download.docker.com/linux/ubuntu  
bionic/stable amd64 Packages  
docker-ce | 5:20.10.0~3-0~ubuntu-bionic | https://download.docker.com/linux/ubuntu  
bionic/stable amd64 Packages  
docker-ce | 5:19.03.15~3-0~ubuntu-bionic | https://download.docker.com/linux/ubuntu  
bionic/stable amd64 Packages  
docker-ce | 5:19.03.14~3-0~ubuntu-bionic | https://download.docker.com/linux/ubuntu  
bionic/stable amd64 Packages  
docker-ce | 5:19.03.13~3-0~ubuntu-bionic | https://download.docker.com/linux/ubuntu  
bionic/stable amd64 Packages
```

也可以不选择最新版本的docker engine, 转而选择特定版本的docker-ce引擎安装, 比如选择 `docker-ce=5:20.10.4~3-0~ubuntu-bionic`

```
sudo apt-get install docker-ce=<VERSION_STRING> docker-ce-cli=<VERSION_STRING>  
containerd.io
```

去docker hub申请一个免费账号。 <https://hub.docker.com/> 再用下面的命令, 在本地docker登录账号。

```
#Open up shell in administrator mode  
sudo -s  
docker login  
# Enter Docker ID & Password.  
exit
```

如果docker login无法访问docker hub网站。一般是命令行下面没有梯子工具。一般需要设置http\_proxy和https\_proxy。

走 http代理，具体查看自己的梯子工具的设置页面。下面假设了翻墙工具的http代理的端口是7890。自己要根据实际情况修改端口号。

```
git config --global http.proxy "http://127.0.0.1:7890"
git config --global https.proxy "http://127.0.0.1:7890"

export http_proxy="http://127.0.0.1:7890"
export https_proxy="http://127.0.0.1:7890"
```

或者走 socks5 代理（如 Shadowsocks）。下面假设了翻墙工具的sock5代理的端口是1089。自己要根据实际情况修改端口号。

```
git config --global http.proxy 'socks://127.0.0.1:1089'
git config --global https.proxy 'socks://127.0.0.1:1089'

export socks5="socks5://127.0.0.1:1089"
```

It is also OK to use proxy url `socks5://127.0.0.1:1089`. [According to this document](#), despite the name `http.proxy`, it should work for both HTTP and HTTPS repository urls.

测试docker run。

```
sudo docker info
sudo docker run hello-world
```

要熟悉docker可以先试试下面的[docker101tutorial里面的步骤](#)。

```
git clone https://github.com/docker/getting-started
cd getting-started
docker build -t docker101tutorial .
docker run -d -p 80:80 --name docker-tutorial docker101tutorial
```

## 2. 实验:

### 2.1 打包制作docker镜像

- 创建一个 Dockerfile 文件

```
# 先创建一个文件夹为docker-admin
mkdir docker-admin

# 进入文件夹docker-admin 并创建一个Dockerfile
cd docker-admin && vim Dockerfile
```

vim 指令:



```
:w 保存文件但不退出vi
:w file 将修改另外保存到file中，不退出vi
:w! 强制保存，不推出vi
:wq 保存文件并退出vi
:wq! 强制保存文件，并退出vi
:q 不保存文件，退出vi
:q! 不保存文件，强制退出vi
:e! 放弃所有修改，从上次保存文件开始再编辑命令历史
```

- 在新建的 Dockerfile 文件里，插入以下命令。

```
FROM docker.io/tomcat
MAINTAINER rstyro:
COPY admin.war /usr/local/tomcat/webapps
```

- 获取到 .war 文件

```
mv xxx.war admin.war
```

- 构建和运行镜像

```
# -t 参数 后面跟镜像名字和tag 注意别忘了后面的 . 点表示当前路径
docker build -t admin:1.0.0 .
# 镜像取名 admin 本机端口映射 8080
docker run --name=admin -p 8080:8080 -d admin:1.0.0
```

本地构建镜像成功：

```
seucyber@localhost:~/docker-admin$ sudo docker build -t admin:1.0.0 .
[sudo] seucyber 的密码:
Sending build context to Docker daemon 11.26kB
Step 1/3 : FROM docker.io/tomcat
--> 040bdb29ab37
Step 2/3 : MAINTAINER rstyro
--> Using cache
--> 409848d77673
Step 3/3 : COPY admin.war /usr/local/tomcat/webapps
--> Using cache
--> b25f49cc2290
Successfully built b25f49cc2290
Successfully tagged admin:1.0.0
```

sudo docker image ls 指令可以查看本地的docker镜像。

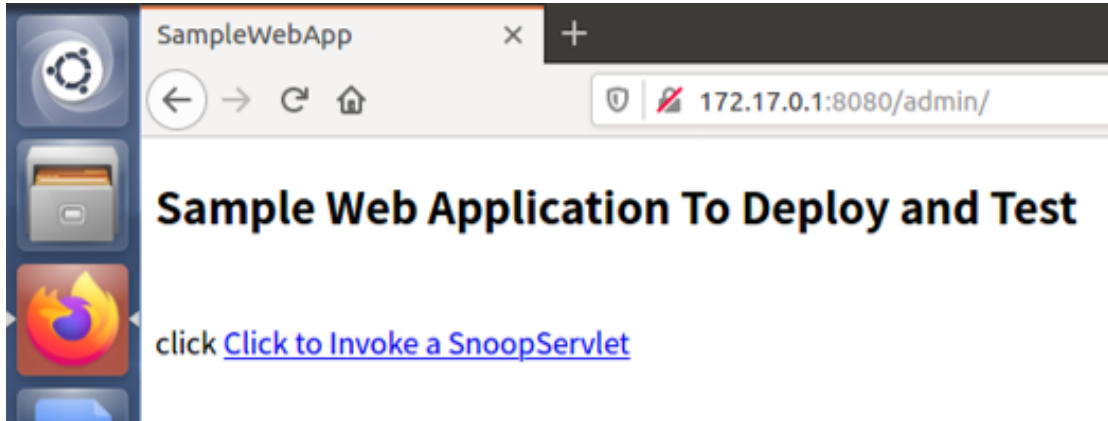
```
seucyber@master:~/docker-admin$ sudo docker image ls
REPOSITORY          TAG                IMAGE ID           CREATED           SIZE
cscyp/admin1.0.0   latest            b25f49cc2290     About an hour ago 649MB
admin               1.0.0            b25f49cc2290     About an hour ago 649MB
tomcat             latest           040bdb29ab37     8 weeks ago      649MB
bestwu/wechat     latest          53c371b7016c     18 months ago    941MB
bestwu/qq         office          d1a0bdfead00     2 years ago      792MB
```

此时已成功将 war 包部署到Tomcat，可通过 #ifconfig 查看网卡 ip 后在本地浏览器 localhost 查看。

```
seucyber@localhost:~/docker-admin$ ifconfig
docker0  Link encap:以太网  硬件地址 02:42:cc:34:d7:27
         inet 地址:172.17.0.1  广播:172.17.255.255  掩码:255.255.0.0
         inet6 地址: fe80::42:ccff:fe34:d727/64  Scope:Link
         UP BROADCAST RUNNING MULTICAST  MTU:1500  跃点数:1
         接收数据包:75211  错误:0  丢弃:0  过载:0  帧数:0
         发送数据包:136456  错误:0  丢弃:0  过载:0  载波:0
         碰撞:0  发送队列长度:0
         接收字节:3593164 (3.5 MB)  发送字节:205310462 (205.3 MB)
```

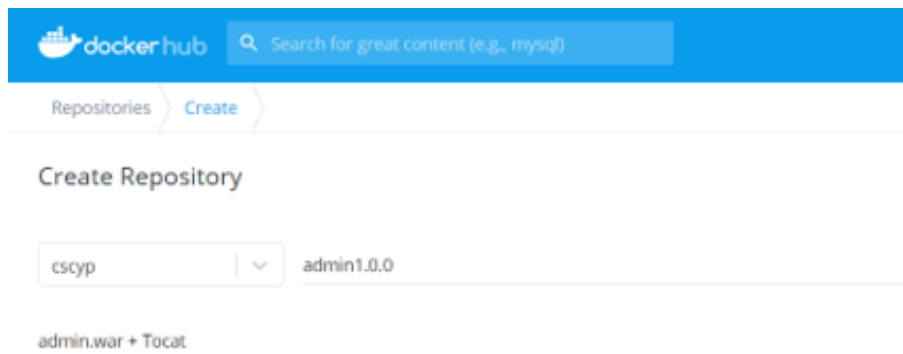
例如：我本地的网卡ip：172.17.0.1，在浏览器访问 `172.17.0.1:8080/admin/`

访问成功界面如下：



## 2.2 docker镜像上传docker hub

- 首先申请一个docker hub 帐号，登录到dockehub点击一下按钮：create —> create repository ,取个名字，我起名为 admin1.0.0，可以选择增加对镜像的描述。



- 镜像打标签

```
#这里的tag不指定默认为latest
docker tag <existing-image> <hub-user>/<repo-name>[:<tag>]
```

- push镜像

1、本地登录 docker hub 账号，命令如下：

```
root@master:~# docker login
Username: 账号名
Password: 密码
Email: 邮箱地址
WARNING: login credentials saved in /root/.docker/config.json
Login Succeeded
```

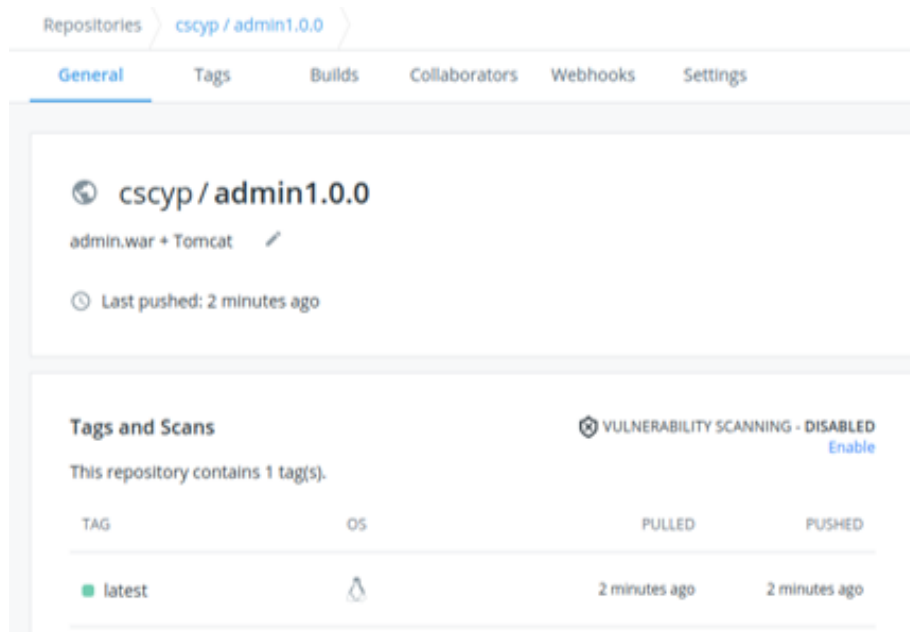
## 2、推送镜像，命令如下：

```
docker push <hub-user>/<repo-name>:<tag>
```

```
seucyber@master:~/docker-admin$ sudo docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, he
ad over to https://hub.docker.com to create one.
Username: cscyp
Password:
WARNING! Your password will be stored unencrypted in /home/seucyber/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
seucyber@master:~/docker-admin$ sudo docker push cscyp/admin1.0.0
The push refers to repository [docker.io/cscyp/admin1.0.0]
1b58ba180c4e: Pushed
9ddc8cd8299b: Mounted from library/tomcat
c9132b9a3fc8: Mounted from library/tomcat
8e2e6f2527c7: Mounted from library/tomcat
500f722b156b: Mounted from library/tomcat
7a9b35031285: Mounted from library/tomcat
7496c5e8691b: Mounted from library/tomcat
aa7af8a465c6: Mounted from library/tomcat
ef9a7b8862f4: Mounted from library/tomcat
a1f2f42922b1: Mounted from library/tomcat
4762552ad7d8: Mounted from library/tomcat
latest: digest: sha256:632714d281ba3b34ec8ca1648a25e401adc98c18c5c3e6d0f44a9e84ca85cc12 size: 2629
```

- 登录验证或下拉验证：



The screenshot shows the Docker Hub interface for the repository 'cscyp/admin1.0.0'. The page is titled 'Repositories > cscyp / admin1.0.0' and has tabs for 'General', 'Tags', 'Builds', 'Collaborators', 'Webhooks', and 'Settings'. The 'General' tab is active, showing the repository name 'cscyp / admin1.0.0', the file 'admin.war + Tomcat', and the status 'Last pushed: 2 minutes ago'. Below this, there is a section for 'Tags and Scans' with a 'VULNERABILITY SCANNING - DISABLED' indicator and an 'Enable' link. A table lists the tags in the repository:

TAG	OS	PULLED	PUSHED
latest	linux	2 minutes ago	2 minutes ago

或下拉镜像也可验证：

```
docker pull <hub-user>/<repo-name>:<tag>
```