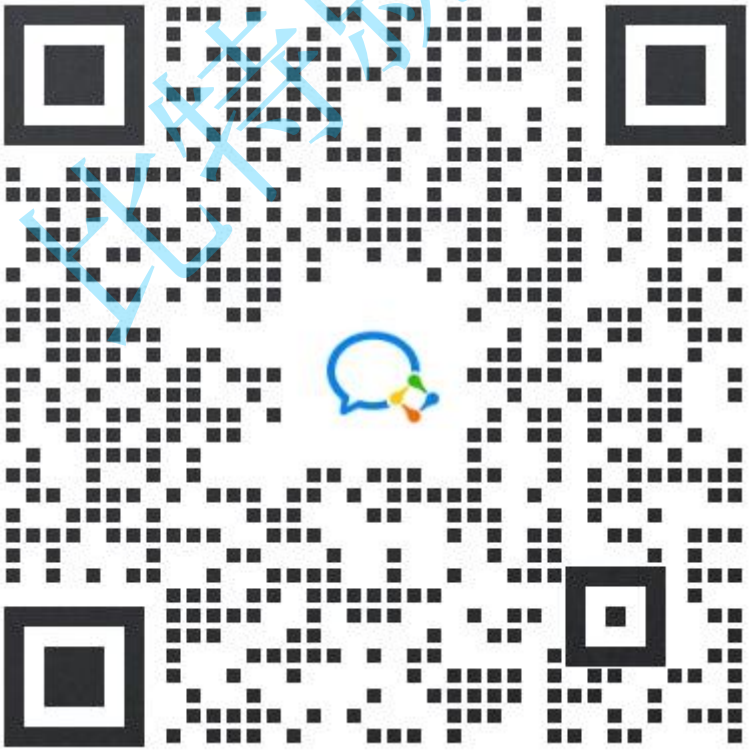
Docker 安装

版权说明

对比特项目感兴趣，可以联系这个微信。



实战目的

掌握如何安装 docker

各版本平台支持情况

- Server 版本

Platform	x86_64 / amd64	arm64 / aarch64	arm (32-bit)	s390x
CentOS	✓	✓		
Debian	✓	✓	✓	
Fedora	✓	✓		
Raspbian			✓	
RHEL				✓
SLES				✓
Ubuntu	✓	✓	✓	✓
Binaries	✓	✓	✓	

- 桌面版本

Platform	x86_64 / amd64	arm64 (Apple Silicon)
Docker Desktop for Linux	✓	
Docker Desktop for Mac (macOS)	✓	✓
Docker Desktop for Windows	✓	

Server 版本安装

Ubuntu 安装（以华为云 Ubuntu 20.04 为例）

安装依赖

1. 操作系统版本

```
Plain Text
Ubuntu Kinetic 22.10
Ubuntu Jammy 22.04 (LTS)
Ubuntu Focal 20.04 (LTS)
Ubuntu Bionic 18.04 (LTS)
```

2. CPU 支持

ARM 和 X86_64

安装 docker

1. 确定 CPU,可以看到我们的是 X86_64，是支持的，如果是 arm 一般会显示 aarch64

```
Shell
root@ecs-144421:~# uname -a
Linux 139-159-150-152 5.4.0-100-generic #113-Ubuntu SMP Thu Feb 3
18:43:29 UTC 2022 x86_64 x86_64 x86_64 GNU/Linux
```

2. 确定操作系统版本，本次我们使用的是 Ubuntu 20.04

```
Shell
root@ecs-144421:~# cat /etc/*release*
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=20.04
DISTRIB_CODENAME=focal
DISTRIB_DESCRIPTION="Ubuntu 20.04.4 LTS"
NAME="Ubuntu"
VERSION="20.04.4 LTS (Focal Fossa)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 20.04.4 LTS"
VERSION_ID="20.04"
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-
policies/privacy-policy"
VERSION_CODENAME=focal
UBUNTU_CODENAME=focal
```

3. 卸载旧版本，如果是新购买的云服务器是没有的，比如输入 docker 并没有这个命令，就不需要卸载

```
Shell

root@ecs-144421:~#  sudo apt-get remove docker docker-engine
docker.io containerd runc
Reading package lists... Done
Building dependency tree
Reading state information... Done
```

```
E: Unable to locate package docker-engine
```

4. 卸载历史版本

```
Shell

#卸载软件
sudo apt-get purge docker-ce docker-ce-cli containerd.io docker-
buildx-plugin docker-compose-plugin docker-ce-rootless-extras -y

#删除目录
sudo rm -rf /var/lib/docker
sudo rm -rf /var/lib/containerd

#这个是老师修改后的目录，根据实际情况设置
sudo rm -rf /data/var/lib/docker
sudo rm -rf  /etc/docker/daemon.json
```

5. 配置 docker 下载源

```
Shell
#curl 命令安装
sudo apt install curl -y
#创建 gpg key 目录
sudo mkdir -m 0755 -p /etc/apt/keyrings
#下载 gpg key
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg
--dearmor --yes -o /etc/apt/keyrings/docker.gpg
echo \
  "deb [arch=$(dpkg --print-architecture) signed-
by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/ubuntu \
  $(lsb_release -cs) stable" | sudo tee
/etc/apt/sources.list.d/docker.list > /dev/null
```

6. 安装

```
Shell
sudo apt-get update
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-
buildx-plugin docker-compose-plugin -y
```

查看安装结果

```
Setting up containerd.io (1.6.18-1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/containerd.service → /lib/systemd/system/contain
Setting up docker-compose-plugin (2.16.0-1~ubuntu.20.04~focal) ...
Setting up docker-ce-cli (5:23.0.1-1~ubuntu.20.04~focal) ...
Setting up pigz (2.4-1) ...
Setting up docker-ce-rootless-extras (5:23.0.1-1~ubuntu.20.04~focal) ...
Setting up docker-ce (5:23.0.1-1~ubuntu.20.04~focal) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /lib/systemd/system/docker.serv
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /lib/systemd/system/docker.socket.
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for systemd (245.4-4ubuntu3.15) ...
```

7. 自动启动配置

```shell
#配置加载
sudo systemctl daemon-reload
#启动服务
sudo systemctl start docker
#开启启动
sudo systemctl enable docker
#查看服务状态
sudo systemctl status docker
```

8. 检查安装结果查看版本

```shell
root@ecs-144421:~# docker version
Client: Docker Engine - Community
 Version:           23.0.1
 API version:       1.42
 Go version:        go1.19.5
 Git commit:        a5ee5b1
 Built:             Thu Feb  9 19:46:56 2023
 OS/Arch:           linux/amd64
 Context:           default

Server: Docker Engine - Community
 Engine:
  Version:          23.0.1
  API version:      1.42 (minimum version 1.12)
  Go version:       go1.19.5
  Git commit:       bc3805a
  Built:            Thu Feb  9 19:46:56 2023
  OS/Arch:          linux/amd64
  Experimental:     false
 containerd:
  Version:          1.6.18
  GitCommit:        2456e983eb9e37e47538f59ea18f2043c9a73640
 runc:
```

```
  Version:          1.1.4
  GitCommit:        v1.1.4-0-g5fd4c4d
 docker-init:
  Version:          0.19.0
  GitCommit:        de40ad0
```

9. 更详细查看 docker 信息

```
Shell
root@ecs-144421:~# docker info
Client:
 Context:    default
 Debug Mode: false
 Plugins:
  buildx: Docker Buildx (Docker Inc.)
    Version:  v0.10.2
    Path:     /usr/libexec/docker/cli-plugins/docker-buildx
  compose: Docker Compose (Docker Inc.)
    Version:  v2.16.0
    Path:     /usr/libexec/docker/cli-plugins/docker-compose
  scan: Docker Scan (Docker Inc.)
    Version:  v0.23.0
    Path:     /usr/libexec/docker/cli-plugins/docker-scan

Server:
 Containers: 0
  Running: 0
  Paused: 0
  Stopped: 0
 Images: 0
 Server Version: 23.0.1
 Storage Driver: overlay2
  Backing Filesystem: extfs
  Supports d_type: true
  Using metacopy: false
  Native Overlay Diff: true
  userxattr: false
 Logging Driver: json-file
 Cgroup Driver: cgroupfs
 Cgroup Version: 1
 Plugins:
  Volume: local
  Network: bridge host ipvlan macvlan null overlay
  Log: awslogs fluentd gcplogs gelf journald json-file local
```

```
logentries splunk syslog
 Swarm: inactive
 Runtimes: io.containerd.runc.v2 runc
 Default Runtime: runc
 Init Binary: docker-init
 containerd version: 2456e983eb9e37e47538f59ea18f2043c9a73640
 runc version: v1.1.4-0-g5fd4c4d
 init version: de40ad0
 Security Options:
  apparmor
  seccomp
   Profile: builtin
 Kernel Version: 5.4.0-100-generic
 Operating System: Ubuntu 20.04.4 LTS
 OSType: linux
 Architecture: x86_64
 CPUs: 1
 Total Memory: 1.936GiB
 Name: 139-159-150-152
 ID: 82ec3110-9c2b-4922-90c3-57a9bf3ff082
 Docker Root Dir: /var/lib/docker
 Debug Mode: false
 Registry: https://index.docker.io/v1/
 Experimental: false
 Insecure Registries:
  127.0.0.0/8
 Live Restore Enabled: false

WARNING: No swap limit support
```

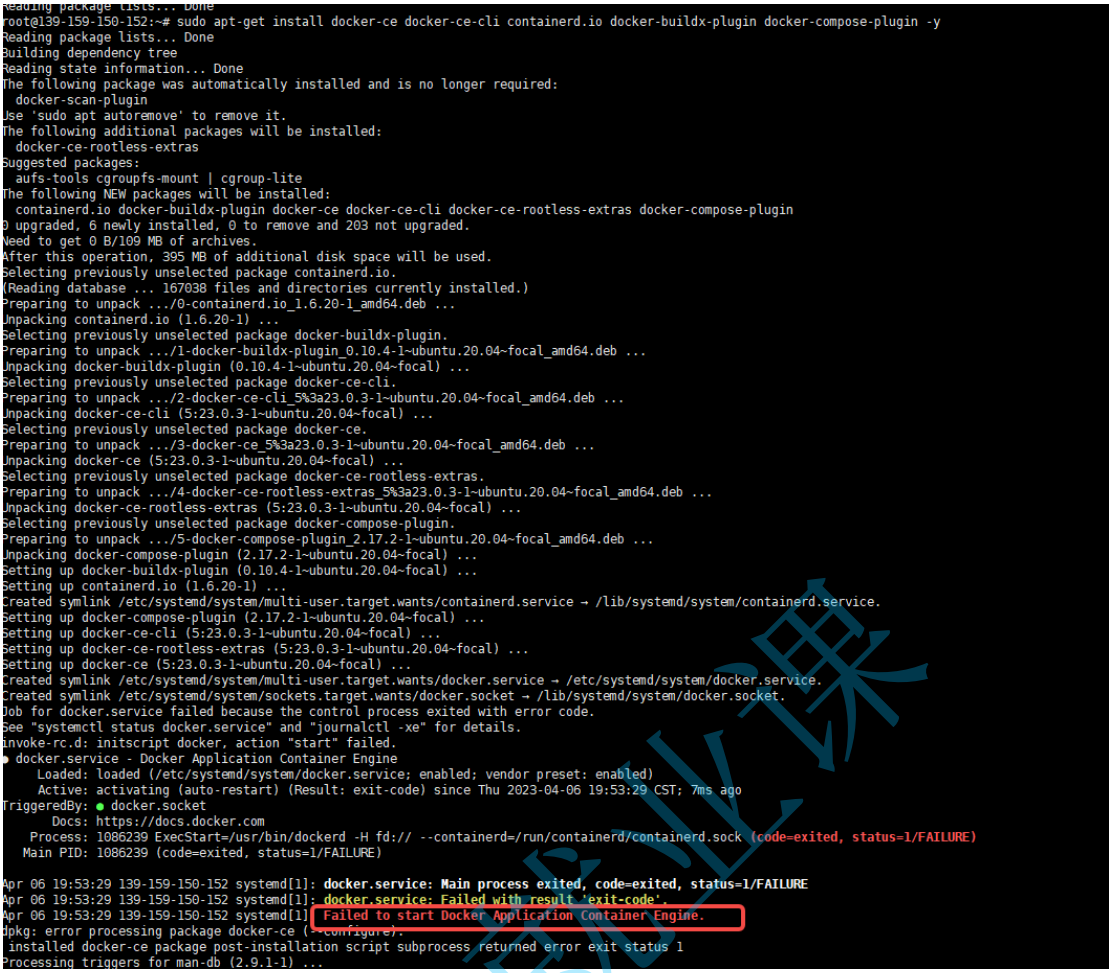
10. 执行 hello-world 可以看到 Hello from Docker,表面 docker 服务正常

```
Shell
root@ecs-144421:~#  sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest:
sha256:6e8b6f026e0b9c419ea0fd02d3905dd0952ad1feea67543f525c73a0a79
0fefb
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working
```

```
correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the
Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image
which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client,
which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container
with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

常见报错

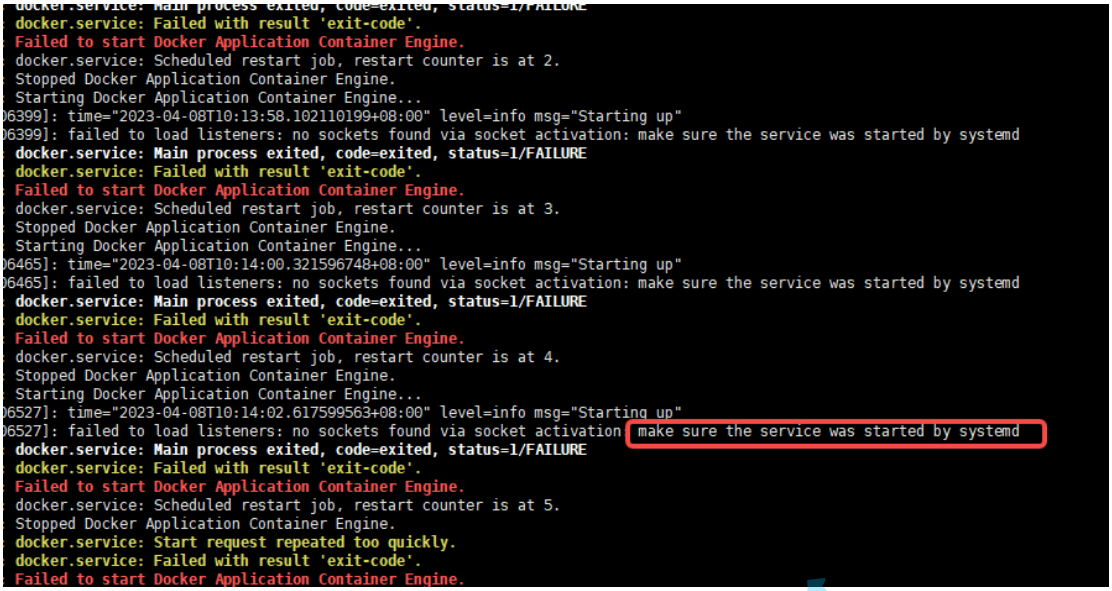
1. 报错截图:



2. 查看日志

```
#journalctl 是操作系统日志查看命令
#-e 表示从末尾看
#-u 表示看哪个系统组件的，我们的组件是 docker
journalctl -eu docker
```

查看报错



3. 执行如下命令修复

```Shell
systemctl daemon-reload
systemctl start docker
```

实战经验

Docker 镜像源修改

对于使用 systemd 的系统（Ubuntu 16.04+、Debian 8+、CentOS 7），在配置文件 /etc/docker/daemon.json 中加入：

```JSON
{
  "registry-mirrors": ["https://docker.mirrors.ustc.edu.cn/"]
}
```

重新启动 dockerd：

```Plain Text
sudo systemctl restart docker
```

参考：https://mirrors.ustc.edu.cn/help/dockerhub.html

Docker 目录修改

Docker 默认的安装目录为/var/lib/docker,这里面会存放很多很多镜像，所以我们在安装的时候需要考虑这个目录的空间，有三种解决方案。

（1）将/var/lib/docker 挂载到一个大的磁盘，这种一般我们能控制挂载目录，像腾讯云这种云厂商在安装 K8s 的节点的时候提供了挂载选项，可以直接挂载这个目录过去

（2）安装之前挂载一个大的磁盘，然后创建一个软链接到/var/lib/docker，这样就自动安装到我们空间比较大的磁盘了

（3）安装了 docker，然后发现忘了配置这个目录，我们需要修改 docker 的配置文件

```shell
#假定我们磁盘的大的目录为 /data
mkdir -p /data/var/lib/docker
# 编辑配置文件
vi /etc/docker/daemon.json
# 输入下面的 json
{
"data-root": "/data/var/lib/docker"
}

# 加载配置
sudo systemctl daemon-reload
# 重启 docker
sudo systemctl restart docker
#查看 docker 状态
sudo systemctl status docker
```

配置文件信息/etc/docker/daemon.json

修改前在/var/lib/docker 下

```
root@ecs-144421:~# ll /var/lib/docker
total 52
drwx--x--- 12 root root 4096 Mar 10 15:57 ./
drwxr-xr-x 46 root root 4096 Mar 10 15:57 ../
drwx--x--x  4 root root 4096 Mar 10 15:57 buildkit/
drwx--x---  2 root root 4096 Mar 10 15:57 containers/
-rw-------  1 root root   36 Mar 10 15:57 engine-id
drwx------  3 root root 4096 Mar 10 15:57 image/
drwxr-x---  3 root root 4096 Mar 10 15:57 network/
drwx--x---  3 root root 4096 Mar 10 15:57 overlay2/
drwx------  4 root root 4096 Mar 10 15:57 plugins/
drwx------  2 root root 4096 Mar 10 15:57 runtimes/
drwx------  2 root root 4096 Mar 10 15:57 swarm/
drwx------  2 root root 4096 Mar 10 16:01 tmp/
drwx-----x  2 root root 4096 Mar 10 15:57 volumes/
```

修改后在/data/var/lib/docker 下

```
root@ecs-144421:~# ll /data/var/lib/docker
total 52
drwx--x--- 12 root root 4096 Mar 10 16:29 ./
drwx--x--x  3 root root 4096 Mar 10 16:29 ../
drwx--x--x  4 root root 4096 Mar 10 16:29 buildkit/
drwx--x---  2 root root 4096 Mar 10 16:29 containers/
-rw-------  1 root root   36 Mar 10 16:29 engine-id
drwx------  3 root root 4096 Mar 10 16:29 image/
drwxr-x---  3 root root 4096 Mar 10 16:29 network/
drwx--x---  3 root root 4096 Mar 10 16:29 overlay2/
drwx------  4 root root 4096 Mar 10 16:29 plugins/
drwx------  2 root root 4096 Mar 10 16:29 runtimes/
drwx------  2 root root 4096 Mar 10 16:29 swarm/
drwx------  2 root root 4096 Mar 10 16:29 tmp/
drwx-----x  2 root root 4096 Mar 10 16:29 volumes/
```

CentOS 安装

安装依赖

1. 支持的操作系统

```
Plain Text
CentOS 7
CentOS 8 (stream)
CentOS 9 (stream)
```

2. 支持的 CPU

```
Plain Text
ARM/X86_64
```

安装 Docker

1. 确认操作系统

```
Bash
[root@centos1 ~]# cat /etc/*release*
CentOS Linux release 7.9.2009 (Core)
Derived from Red Hat Enterprise Linux 7.9 (Source)
NAME="CentOS Linux"
VERSION="7 (Core)"
ID="centos"
ID_LIKE="rhel fedora"
VERSION_ID="7"
PRETTY_NAME="CentOS Linux 7 (Core)"
ANSI_COLOR="0;31"
CPE_NAME="cpe:/o:centos:centos:7"
HOME_URL="https://www.centos.org/"
BUG_REPORT_URL="https://bugs.centos.org/"

CENTOS_MANTISBT_PROJECT="CentOS-7"
CENTOS_MANTISBT_PROJECT_VERSION="7"
REDHAT_SUPPORT_PRODUCT="centos"
REDHAT_SUPPORT_PRODUCT_VERSION="7"

CentOS Linux release 7.9.2009 (Core)
CentOS Linux release 7.9.2009 (Core)
cpe:/o:centos:centos:7
```

2. 确认 CPU 架构

```
Shell
[root@centos1 ~]# uname -a
Linux centos1 3.10.0-1160.71.1.el7.x86_64 #1 SMP Tue Jun 28
15:37:28 UTC 2022 x86_64 x86_64 x86_64 GNU/Linux
```

3. 卸载旧版本

```
Plain Text
```

```
sudo yum remove docker \
                  docker-client \
                  docker-client-latest \
                  docker-common \
                  docker-latest \
                  docker-latest-logrotate \
                  docker-logrotate \
                  docker-engine
```

4. 卸载历史版本

```
#删除机器上的包
sudo yum remove docker-ce docker-ce-cli containerd.io docker-
buildx-plugin docker-compose-plugin docker-ce-rootless-extras

#执行卸载
sudo rm -rf /var/lib/docker
sudo rm -rf /var/lib/containerd

#这个是老师修改后的目录，根据实际情况设置
sudo rm -rf /data/var/lib/docker
sudo rm -rf  /etc/docker/daemon.json
```

5. 配置仓库

```
Shell
[root@centos1 ~]# ll /etc/yum.repos.d/
total 40
-rw-r--r--. 1 root root 1664 Nov 23  2020 CentOS-Base.repo
-rw-r--r--. 1 root root 1309 Nov 23  2020 CentOS-CR.repo
-rw-r--r--. 1 root root  649 Nov 23  2020 CentOS-Debuginfo.repo
-rw-r--r--. 1 root root  314 Nov 23  2020 CentOS-fasttrack.repo
-rw-r--r--. 1 root root  630 Nov 23  2020 CentOS-Media.repo
-rw-r--r--. 1 root root 1331 Nov 23  2020 CentOS-Sources.repo
-rw-r--r--. 1 root root 8515 Nov 23  2020 CentOS-Vault.repo
-rw-r--r--. 1 root root  616 Nov 23  2020 CentOS-x86_64-
kernel.repo

[root@centos1 ~]# sudo yum install -y yum-utils

[root@centos1 ~]# sudo yum-config-manager   --add-repo
https://download.docker.com/linux/centos/docker-ce.repo
Loaded plugins: fastestmirror
```

```
adding repo from: https://download.docker.com/linux/centos/docker-ce.repo
grabbing file https://download.docker.com/linux/centos/docker-ce.repo to /etc/yum.repos.d/docker-ce.repo
repo saved to /etc/yum.repos.d/docker-ce.repo
[root@centos1 ~]# ll /etc/yum.repos.d/
total 44
-rw-r--r--. 1 root root 1664 Nov 23  2020 CentOS-Base.repo
-rw-r--r--. 1 root root 1309 Nov 23  2020 CentOS-CR.repo
-rw-r--r--. 1 root root  649 Nov 23  2020 CentOS-Debuginfo.repo
-rw-r--r--. 1 root root  314 Nov 23  2020 CentOS-fasttrack.repo
-rw-r--r--. 1 root root  630 Nov 23  2020 CentOS-Media.repo
-rw-r--r--. 1 root root 1331 Nov 23  2020 CentOS-Sources.repo
-rw-r--r--. 1 root root 8515 Nov 23  2020 CentOS-Vault.repo
-rw-r--r--. 1 root root  616 Nov 23  2020 CentOS-x86_64-kernel.repo
-rw-r--r--. 1 root root 1919 Apr  5 07:45 docker-ce.repo
# 配置使用国内源
[root@centos1 yum.repos.d]# sed -i 's@//download.docker.com@//mirrors.ustc.edu.cn/docker-ce@g' /etc/yum.repos.d/docker-ce.repo
```

6. 安装最新版本

```
Plain Text
sudo yum install -y docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
```

7. 启动 docker

```
Bash
#配置加载
sudo systemctl daemon-reload
#启动服务
sudo systemctl start docker
#开启启动
sudo systemctl enable docker
#查看服务状态
sudo systemctl status docker
```

8. 检查安装结果查看版本

```
Shell
[root@centos1 ~]# docker version
```

```
Client: Docker Engine - Community
 Version:           23.0.3
 API version:       1.42
 Go version:        go1.19.7
 Git commit:        3e7cbfd
 Built:             Tue Apr  4 22:04:18 2023
 OS/Arch:           linux/amd64
 Context:           default

Server: Docker Engine - Community
 Engine:
  Version:          23.0.3
  API version:      1.42 (minimum version 1.12)
  Go version:       go1.19.7
  Git commit:       59118bf
  Built:            Tue Apr  4 22:02:01 2023
  OS/Arch:          linux/amd64
  Experimental:     false
 containerd:
  Version:          1.6.20
  GitCommit:        2806fc1057397dbaeefbea0e4e17bddfbd388f38
 runc:
  Version:          1.1.5
  GitCommit:        v1.1.5-0-gf19387a
 docker-init:
  Version:          0.19.0
  GitCommit:        de40ad0
```

9. 更详细查看 docker 信息

```
Shell
[root@centos1 ~]# docker info
Client:
 Context:    default
 Debug Mode: false
 Plugins:
  buildx: Docker Buildx (Docker Inc.)
    Version:  v0.10.4
    Path:     /usr/libexec/docker/cli-plugins/docker-buildx
  compose: Docker Compose (Docker Inc.)
    Version:  v2.17.2
    Path:     /usr/libexec/docker/cli-plugins/docker-compose
```

```
Server:
 Containers: 0
  Running: 0
  Paused: 0
  Stopped: 0
 Images: 0
 Server Version: 23.0.3
 Storage Driver: overlay2
  Backing Filesystem: xfs
  Supports d_type: true
  Using metacopy: false
  Native Overlay Diff: true
  userxattr: false
 Logging Driver: json-file
 Cgroup Driver: cgroupfs
 Cgroup Version: 1
 Plugins:
  Volume: local
  Network: bridge host ipvlan macvlan null overlay
  Log: awslogs fluentd gcplogs gelf journald json-file local
logentries splunk syslog
 Swarm: inactive
 Runtimes: io.containerd.runc.v2 runc
 Default Runtime: runc
 Init Binary: docker-init
 containerd version: 2806fc1057397dbaeefbea0e4e17bddfbd388f38
 runc version: v1.1.5-0-gf19387a
 init version: de40ad0
 Security Options:
  seccomp
   Profile: builtin
 Kernel Version: 3.10.0-1160.71.1.el7.x86_64
 Operating System: CentOS Linux 7 (Core)
 OSType: linux
 Architecture: x86_64
 CPUs: 2
 Total Memory: 1.795GiB
 Name: centos1
 ID: 0b3c79d5-957d-4d04-a856-ac15a2a09db2
 Docker Root Dir: /var/lib/docker
 Debug Mode: false
 Registry: https://index.docker.io/v1/
 Experimental: false
 Insecure Registries:
```

```
  127.0.0.0/8
 Live Restore Enabled: false
```

10. 执行 hello-world 可以看到 Hello from Docker,表面 docker 服务正常

```
Shell
[root@centos1 ~]# sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest:
sha256:ffb13da98453e0f04d33a6eee5bb8e46ee50d08ebe17735fc0779d0349e
889e9
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working
correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the
Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image
which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client,
which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container
with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```


实战经验

Docker 镜像源修改

对于使用 systemd 的系统（Ubuntu 16.04+、Debian 8+、CentOS 7）， 在配置文件 /etc/docker/daemon.json 中加入：

```JSON
{
  "registry-mirrors": ["https://docker.mirrors.ustc.edu.cn/"]
}
```

重新启动 dockerd：

```Plain Text
sudo systemctl restart docker
```

参考：https://mirrors.ustc.edu.cn/help/dockerhub.html

Docker 目录修改

Docker 默认的安装目录为/var/lib/docker,这里面会存放很多很多镜像，所以我们在安装的时候需要考虑这个目录的空间，有三种解决方案。

（1）将/var/lib/docker 挂载到一个大的磁盘，这种一般我们能控制挂载目录，像腾讯云这种云厂商在安装 K8s 的节点的时候提供了挂载选项，可以直接挂载这个目录过去

（2）安装之前挂载一个大的磁盘，然后创建一个软链接到/var/lib/docker，这样就自动安装到我们空间比较大的磁盘了

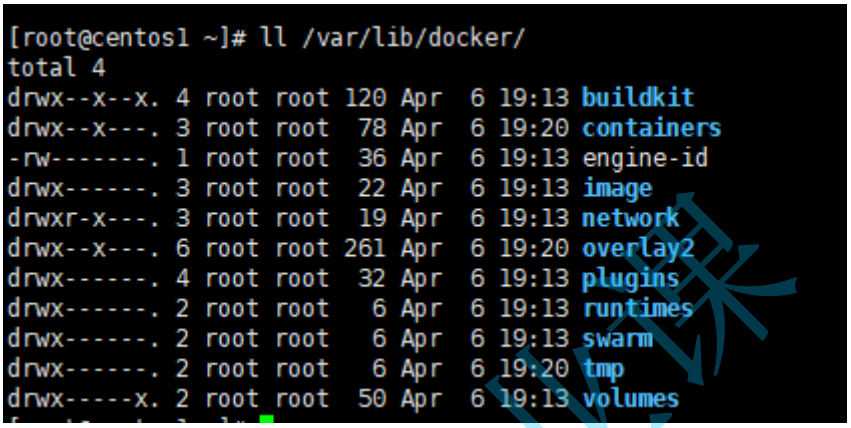
（3）安装了 docker，然后发现忘了配置这个目录，我们需要修改 docker 的配置文件

```Shell
#假定我们磁盘的大的目录为 /data
mkdir -p /data/var/lib/docker
# 编辑配置文件
vi /etc/docker/daemon.json
# 输入下面的 json
{
"data-root": "/data/var/lib/docker"
}

# 加载配置
sudo systemctl daemon-reload
# 重启 docker
sudo systemctl restart docker
#查看 docker 状态
```

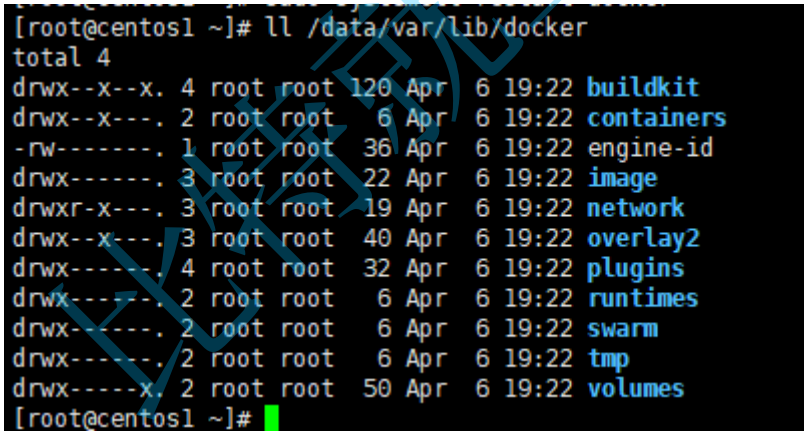
```
sudo systemctl status docker
```

配置文件信息/etc/docker/daemon.json

修改前在/var/lib/docker 下



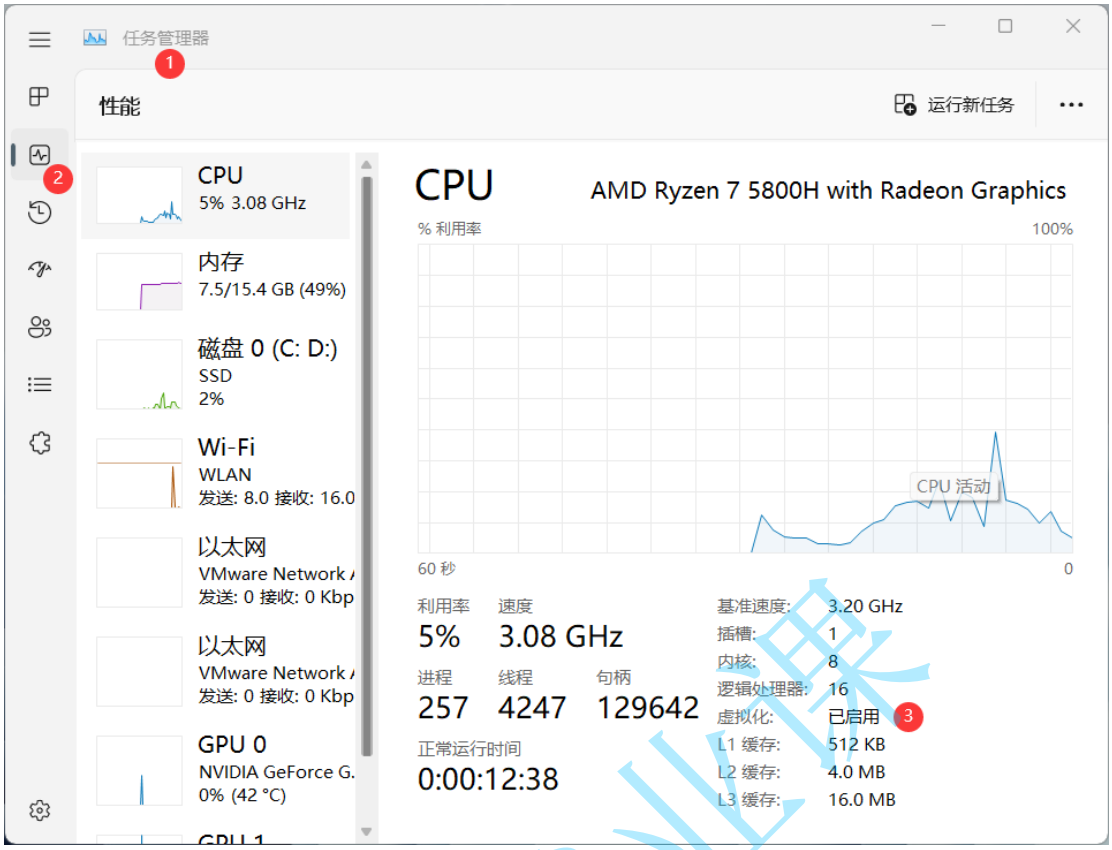
修改后在/data/var/lib/docker 下



GUI 版本安装（以 windows 11 为例）

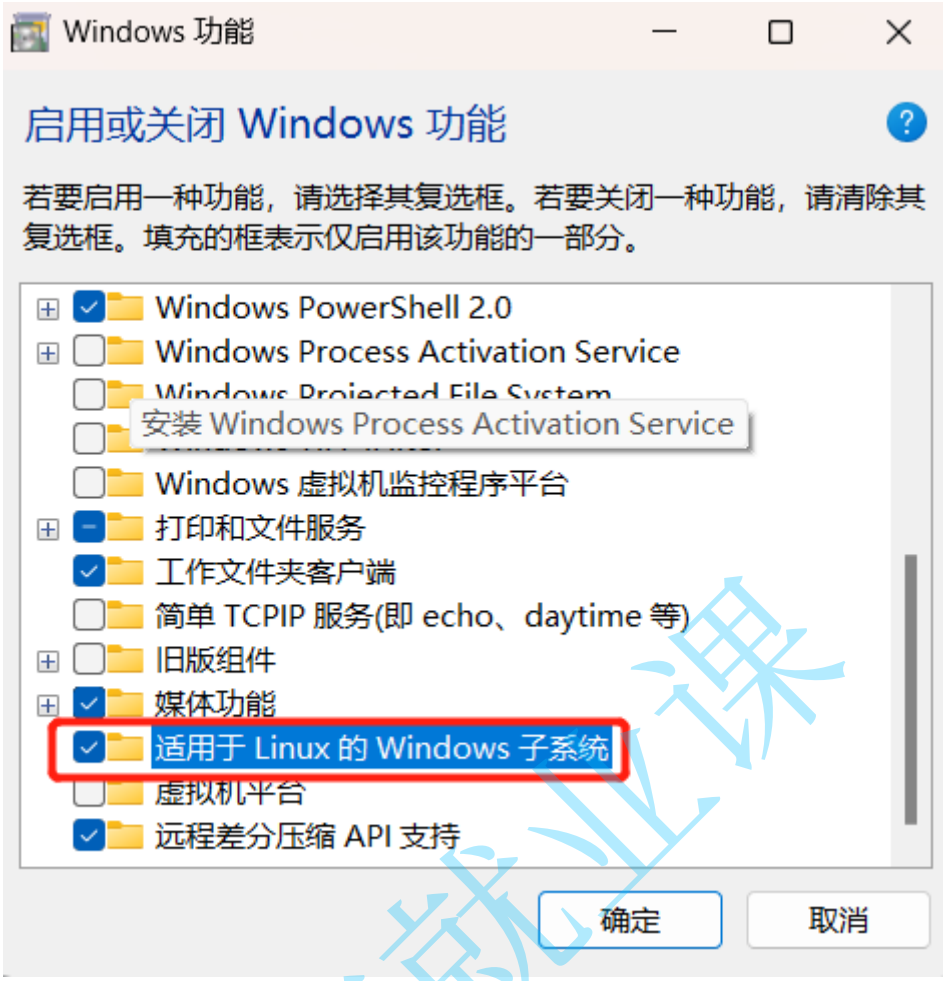
安装依赖

确定开启虚拟化

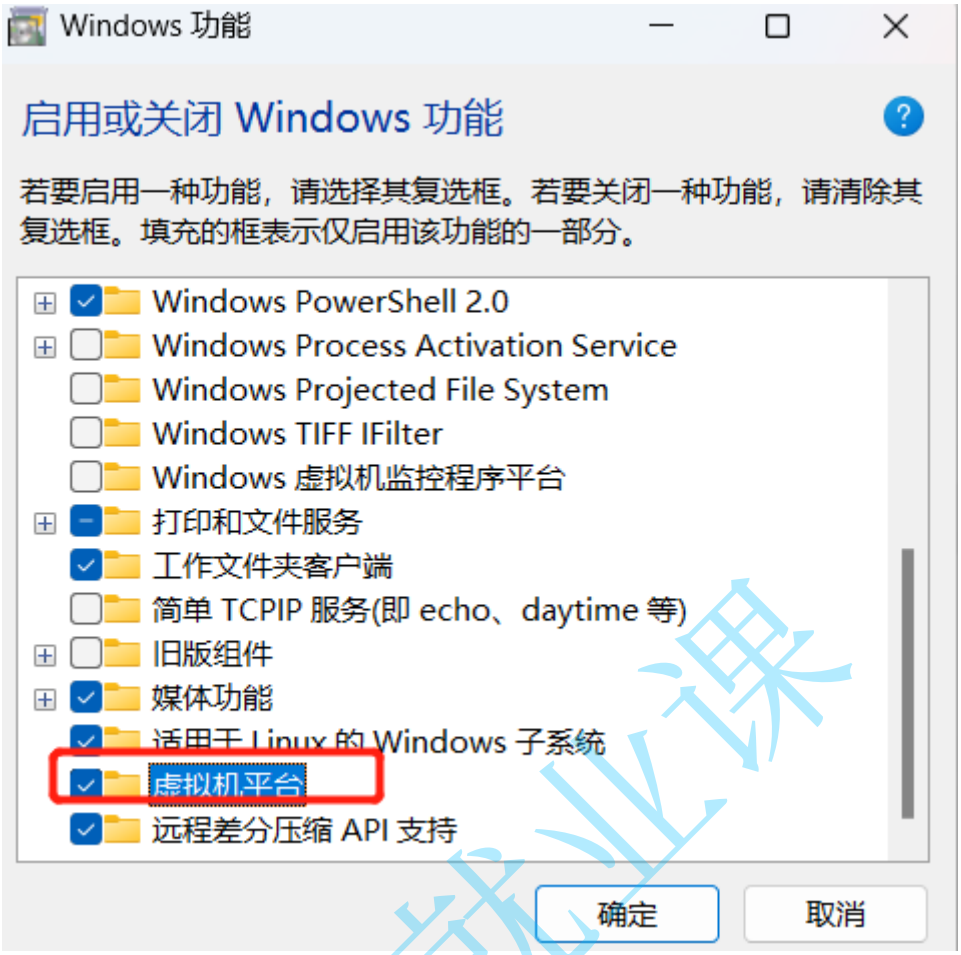


- 输入 win+i，然后输入启用或者关闭 windows 功能

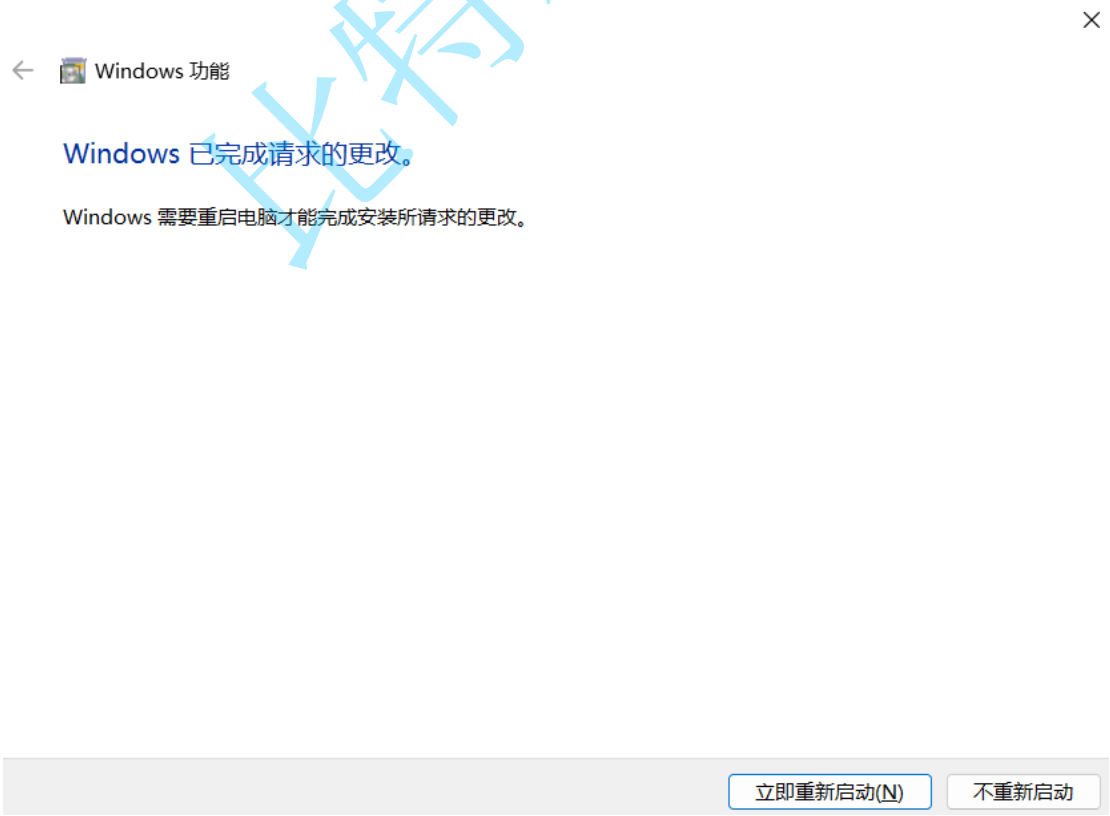
- 选择 windows 子系统和


Windows 功能
启用或关闭 Windows 功能
若要启用一种功能，请选择其复选框。若要关闭一种功能，请清除其复选框。填充的框表示仅启用该功能的一部分。
Windows PowerShell 2.0
Windows Process Activation Service
安装 Windows Process Activation Service
Windows 虚拟机监控程序平台
打印和文件服务
工作文件夹客户端
简单 TCPIP 服务(即 echo、daytime 等)
旧版组件
媒体功能
适用于 Linux 的 Windows 子系统
虚拟机平台
远程差分压缩 API 支持
确定
取消

和



- 重启电脑，完成系统设置



- 安装 WSL2
  - 以管理员权限运行 `PowerShell`
  - 查看版本，如果不是 2 需要更新到 2

```Shell
wsl --status
```

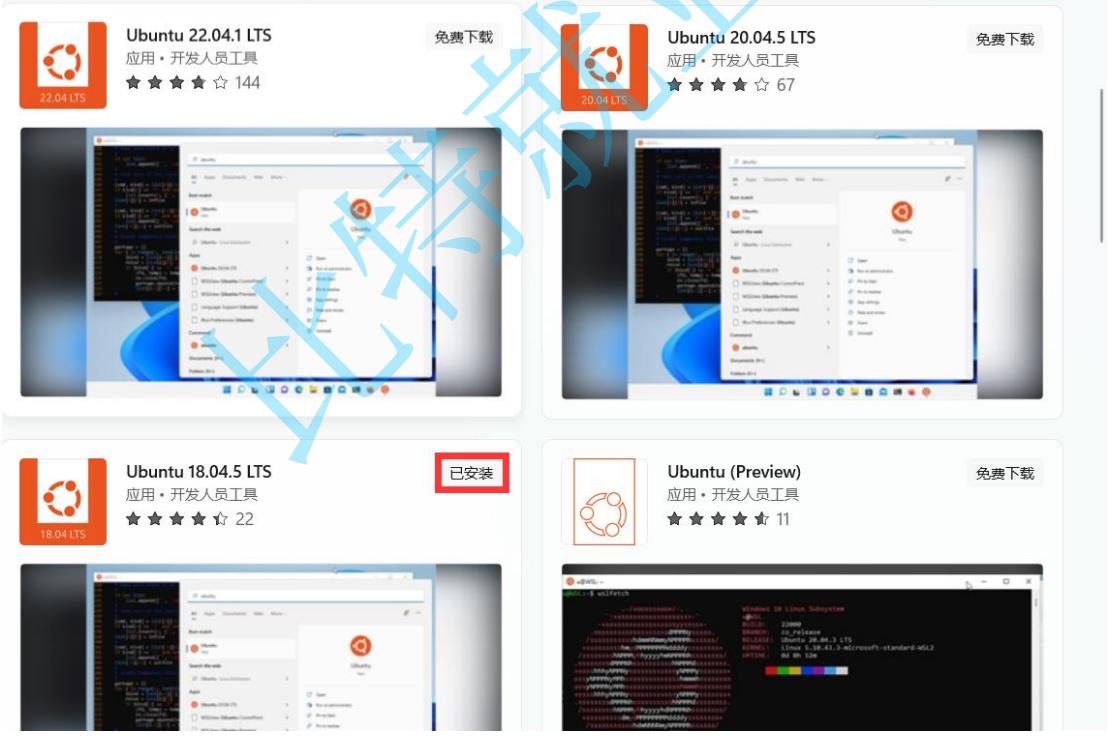
  - `PowerShell` 运行命令更新 wsl 到最新版本

```Shell
PS C:\WINDOWS\system32> wsl --update
```

  - 设置 wsl 默认版本

```Shell
PS C:\WINDOWS\system32> wsl --set-default-version 2
```

- 通过微软应用商店安装 `Ubuntu 18.04.5`



- 启动安装好的 `Ubunt18.04`，如图表示启动成功

```Shell
Installing, this may take a few minutes...
Please create a default UNIX user account. The username does not
need to match your Windows username.
```

```
For more information visit: https://aka.ms/wslusers
Enter new UNIX username: zsc
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Installation successful!
To run a command as administrator (user "root"), use "sudo
<command>".
See "man sudo_root" for details.

zsc@LAPTOP-DIG6CK4H:~$
```

安装 docker

- 点击链接下载 Docker Desktop for Windows

- 下载好之后双击 `Docker Desktop Installer.exe` 开始安装
- 点击 `Docker Desktop` 桌面快捷方式运行 `Dokcer`
- 确认 `Docker` 安装成功, 在 `PowerShell` 中输入 `docker version` 命令确认 Client 和 Server 是否启动

```
Shell
PS C:\Users\zsc> docker version
Client:
 Cloud integration: v1.0.29
 Version:           20.10.21
 API version:       1.41
 Go version:        go1.18.7
 Git commit:        baeda1f
 Built:             Tue Oct 25 18:08:16 2022
 OS/Arch:           windows/amd64
 Context:           default
 Experimental:      true

Server: Docker Desktop 4.15.0 (93002)
 Engine:
  Version:          20.10.21
  API version:      1.41 (minimum version 1.12)
  Go version:       go1.18.7
  Git commit:       3056208
  Built:            Tue Oct 25 18:00:19 2022
```

```
 OS/Arch:           linux/amd64
 Experimental:      false
containerd:
 Version:           1.6.10
 GitCommit:         770bd0108c32f3fb5c73ae1264f7e503fe7b2661
runc:
 Version:           1.1.4
 GitCommit:         v1.1.4-0-g5fd4c4d
docker-init:
 Version:           0.19.0
 GitCommit:         de40ad0
```