

云存储应用技术

实验一：虚拟化技术

丁烨

dingye@dgut.edu.cn

网络空间安全学院

2019-09-26



東莞理工學院
DONGGUAN UNIVERSITY OF TECHNOLOGY

❖ 任课老师

❖ 丁焯，9A-305 室， dingye@dgut.edu.cn

❖ 课程网站

❖ <https://dingye.me/cloud.html>

❖ 课件、作业要求、各类通知、消息均在此网站发布

❖ 教材与参考书

❖ 以讲义为主

❖ 成绩

❖ 考勤 10%、平时作业 30%、大作业 60%

❖ 教学目标

- ❖ 掌握基本的云存储概念
- ❖ 掌握云存储技术相关的软件及工具库
- ❖ 了解作为云存储技术人员所具备的相关能力

❖ 教学方法

- ❖ 理论课：讲授知识点和云存储基本思想，提供相关阅读资料
- ❖ 实验课：通过项目实践真正体会和运用云存储技术相关的软件及工具库

❖ 课程安排

- ❖ 理论课：12 节，共 36 学时
- ❖ 实验课：6 节，共 18 学时，实验课为 UNIX 环境

❖ 实验课

1. 虚拟化技术
2. 网络存储
3. 文件托管服务
4. 对象存储
5. 消息队列
6. 期末大作业

搭建实验环境

Docker 基础应用



- ❖ Docker
- ❖ <https://www.docker.com/>
- ❖ 开源免费，源代码：<https://github.com/docker/docker-ce>
- ❖ Docker 是一个开放源代码软件项目，让应用程序部署在容器下的工作可以自动化进行，借此在 Linux 操作系统上，提供一个额外的软件抽象层，以及操作系统层虚拟化的自动管理机制
- ❖ Docker 利用 Linux 核心中的资源分离机制，例如 cgroups 以及命名空间（namespaces），来创建独立的容器（containers）
- ❖ 容器使得 Docker 可以在单一 Linux 实体下运作，避免创建个虚拟机造成的额外负担
- ❖ Docker 是目前最主流的虚拟机解决方案

❖ Docker 的优点

- ❖ 不严格符合虚拟机标准，而是采用 Linux 核心资源分离技术实现虚拟化
- ❖ 性能卓越，接近物理机性能
- ❖ 部署虚拟机（容器）不需要部署整个操作系统，操作快速便捷
- ❖ 拥有庞大的社区，提供了大量开源免费（以及收费）的操作系统镜像

❖ Docker 的缺点

- ❖ 仅支持 Linux，其他操作系统需要借助其他虚拟机虚拟 Linux 环境
- ❖ MacOS: 借助 HyperKit
- ❖ Windows: 借助 Hyper-V 或 VirtualBox（已淘汰）

❖ 容器 (Container)

- ❖ 一般虚拟机的概念里，一个操作系统镜像 (image) 可以用于创建多个虚拟机 (VM)
- ❖ 每个虚拟机拥有独立的虚拟内存、虚拟 CPU、虚拟硬盘，操作系统完全隔离
- ❖ 虚拟机 (虚拟硬盘) 可以被复制并重启，从而产生分支

- ❖ Docker 的概念里，一个硬盘镜像 (image) 可以用于创建多个容器 (Container)
- ❖ Docker 一般不会直接使用操作系统镜像
- ❖ 容器共享宿主机的内存、CPU，但硬盘相对独立 (采用分块镜像模式)
- ❖ 容器共享宿主机的部分操作系统指令，如 cgroups、命名空间 (namespaces) 等
- ❖ 容器可以重新打包成为新的硬盘镜像，镜像可以被用于创建新的容器，从而产生分支
- ❖ 容器内没有自己的内核，而且也没有进行硬件虚拟，因此容器要比传统虚拟机更为轻便

❖ 容器的销毁

- ❖ 不同于一般的虚拟机，Docker 的容器使用一次就会立刻销毁，且不会保留数据
- ❖ 这样设计的原因是因为容器相当于一个应用，应用使用完毕就不需要存在
- ❖ 如果需要保持容器持续运行，需要使用 daemon 保持后台运行
- ❖ 如果需要保留数据，需要将容器的部分文件映射到宿主机

❖ 使用 Dockerfile 定制镜像

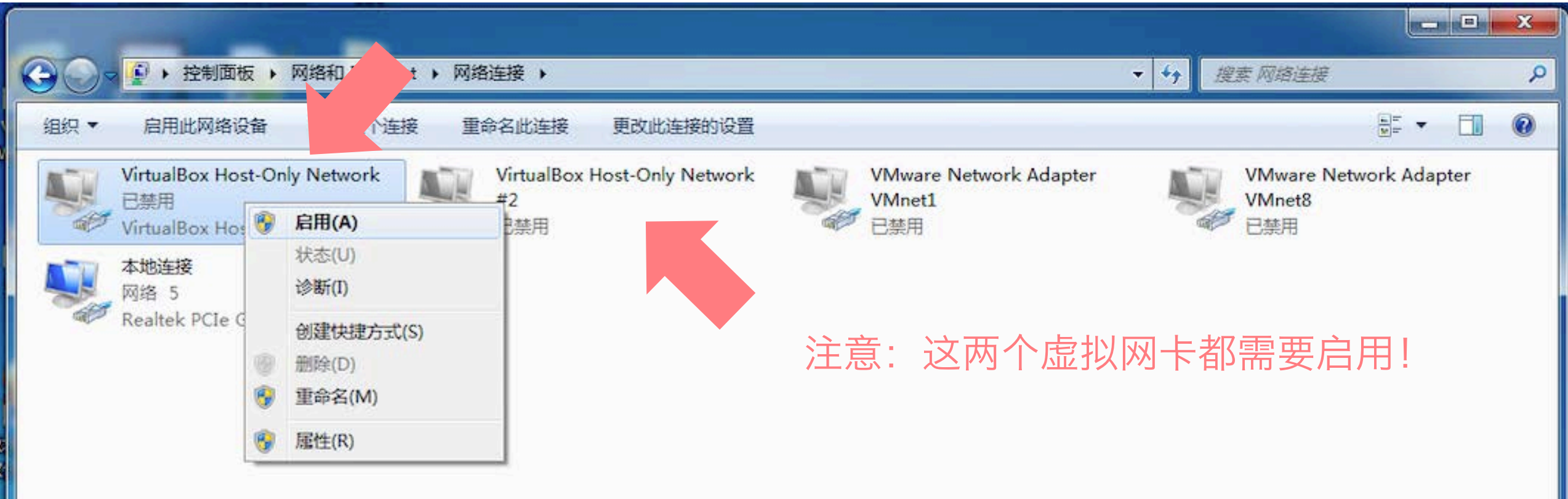
- ❖ Docker 允许用户使用 Dockerfile 来定制镜像，从而不用保存大体积的硬盘镜像
- ❖ 过程类似于 MIDI 和 MP3 的区别

❖ Windows 7

- ❖ 由于实验室的操作系统仅支持 Windows 7，我们采用了已经淘汰的 **Docker Toolbox**
- ❖ 如使用实验室的机器，Docker 环境已经搭建完毕
- ❖ 如自己携带电脑，请使用 Ubuntu 18.04 / macOS Mojave / Windows 10

❖ Windows 7

- ❖ 部分实验室机器的 Docker 虚拟网卡被禁用，需要去控制面板的“网络和 Internet”中的“网络连接”中启用



注意：这两个虚拟网卡都需要启用！

- ❖ Windows 7
- ❖ 找到开始菜单或桌面的 Docker QuickStart Terminal 快捷方式
- ❖ 启动 Docker Toolbox



Docker
Quickstart
Terminal



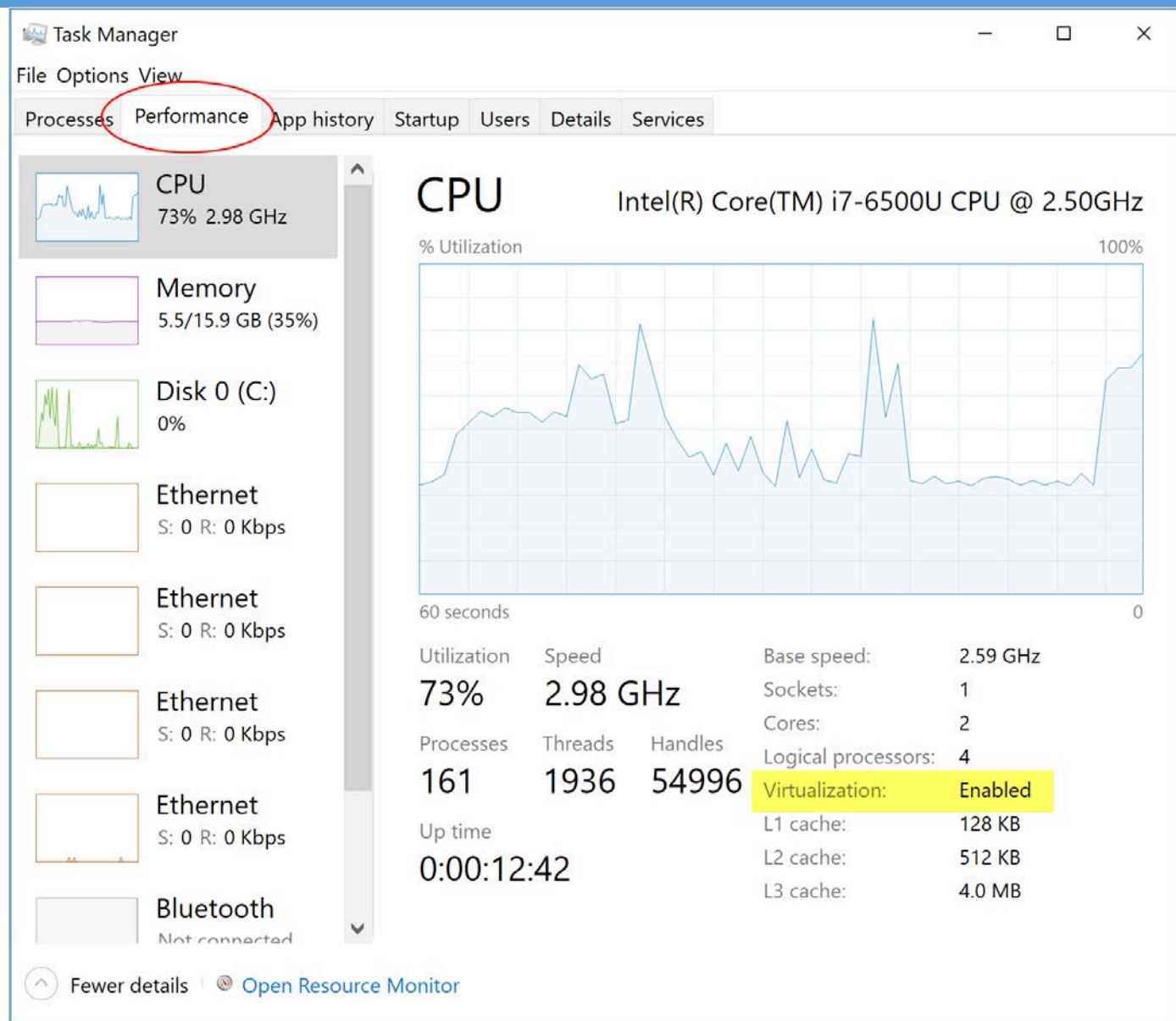
Kitematic
(Alpha)



Oracle VM
VirtualBox

❖ Windows 10

- ❖ 安装 Docker 前，需要确认 BIOS 的虚拟化功能已经成功开启
- ❖ 可以使用任务管理器的性能标签查看虚拟化是否已开启



The screenshot shows the Windows Task Manager Performance tab. The 'Performance' tab is selected and circled in red. The main display shows system metrics for the CPU, Memory, Disk 0 (C:), Ethernet, and Bluetooth. The CPU section is expanded, showing a graph of % Utilization over 60 seconds and a table of specifications.

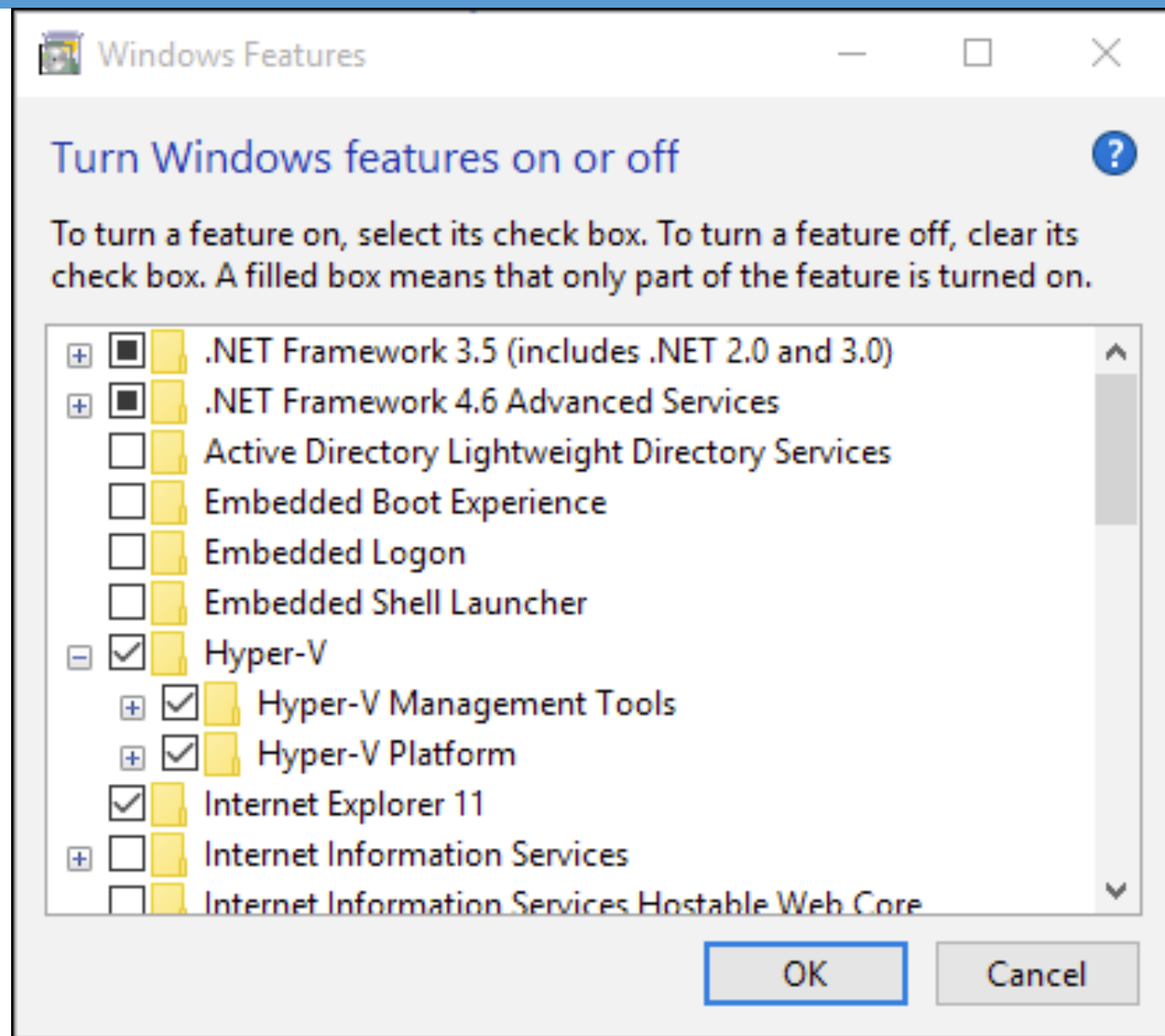
Metric	Value
CPU	73% 2.98 GHz
Memory	5.5/15.9 GB (35%)
Disk 0 (C:)	0%
Ethernet	S: 0 R: 0 Kbps
Ethernet	S: 0 R: 0 Kbps
Ethernet	S: 0 R: 0 Kbps
Ethernet	S: 0 R: 0 Kbps
Bluetooth	Not connected

Utilization	Speed	Base speed:	2.59 GHz	
73%	2.98 GHz	Sockets:	1	
Processes	Threads	Handles	Cores:	2
161	1936	54996	Logical processors:	4
Up time	0:00:12:42			
		Virtualization:	Enabled	
		L1 cache:	128 KB	
		L2 cache:	512 KB	
		L3 cache:	4.0 MB	

- ❖ Windows 10
- ❖ 如未开启虚拟化功能，需要进入 BIOS 开启
- ❖ 进入 BIOS：开机时按下 DEL 键
- ❖ 进入 CPU 设置或安全设置
- ❖ 开启虚拟化功能

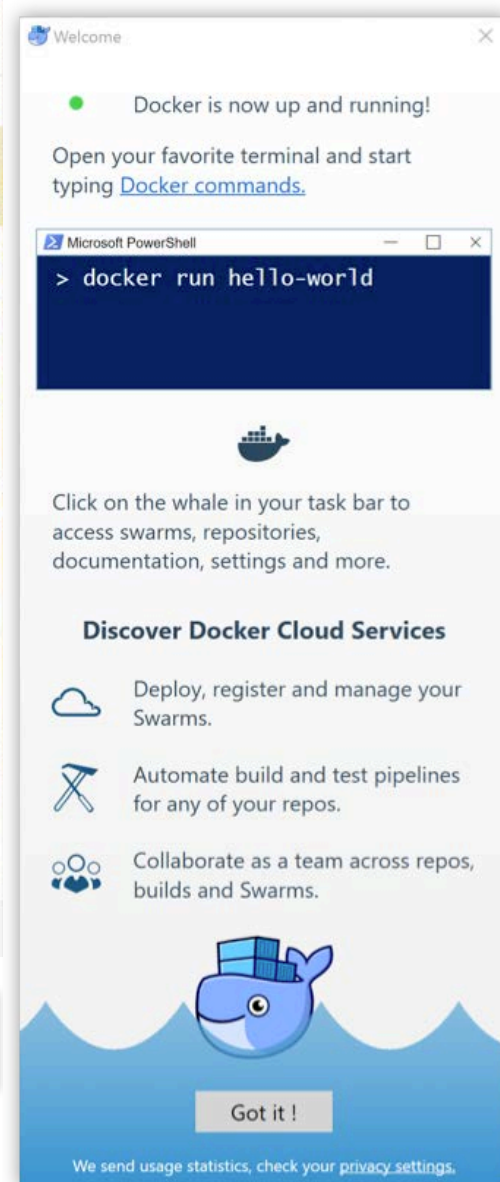
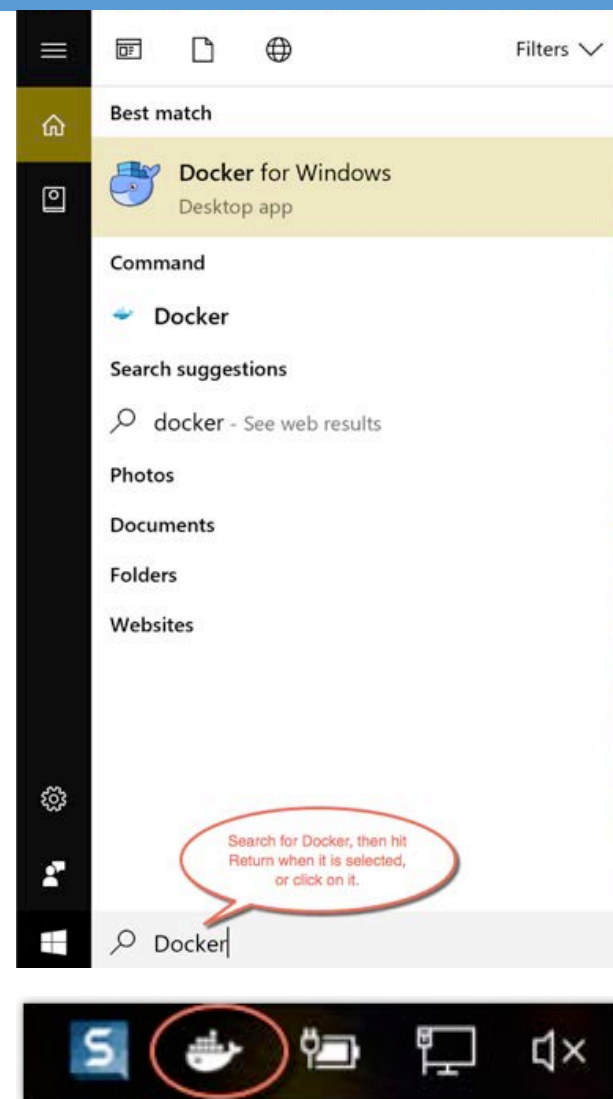


- ❖ Windows 10
- ❖ Windows 版本的 Docker 是基于 Hyper-V 技术实现的，因此在安装 Docker 前需要开启 Hyper-V
- ❖ Hyper-V 在控制面板的“添加 / 删除程序”中的“开启 / 关闭 Windows 功能”选项中
- ❖ 由于需要使用 Hyper-V，Docker 仅支持 64 位的 Windows 10



- ❖ Windows 10
- ❖ 下载运行安装包:
- ❖ <https://download.docker.com/win/stable/Docker%20for%20Windows%20Installer.exe>
- ❖ 如果上述文件下载较慢, 可以使用 DaoCloud 的镜像:
- ❖ https://download.daocloud.io/d/Docker_Mirror/Docker_for_Windows_Mac/17.03.1-ce/docker-for-windows-stable.msi

- ❖ Windows 10
- ❖ 安装完毕后，从开始菜单中找到 Docker 图标并点击运行
- ❖ 运行之后，会在任务栏看到多了一个鲸鱼图标，这个图标表明了 Docker 的运行状态
- ❖ 每次点击鲸鱼图标会弹出操作菜单
- ❖ 如需操作 Docker，请使用 PowerShell



- ❖ Ubuntu 18.04
- ❖ 自动安装 Docker:
- ❖ `curl -fsSL get.docker.com -o get-docker.sh`
- ❖ `sudo sh get-docker.sh --mirror Aliyun`

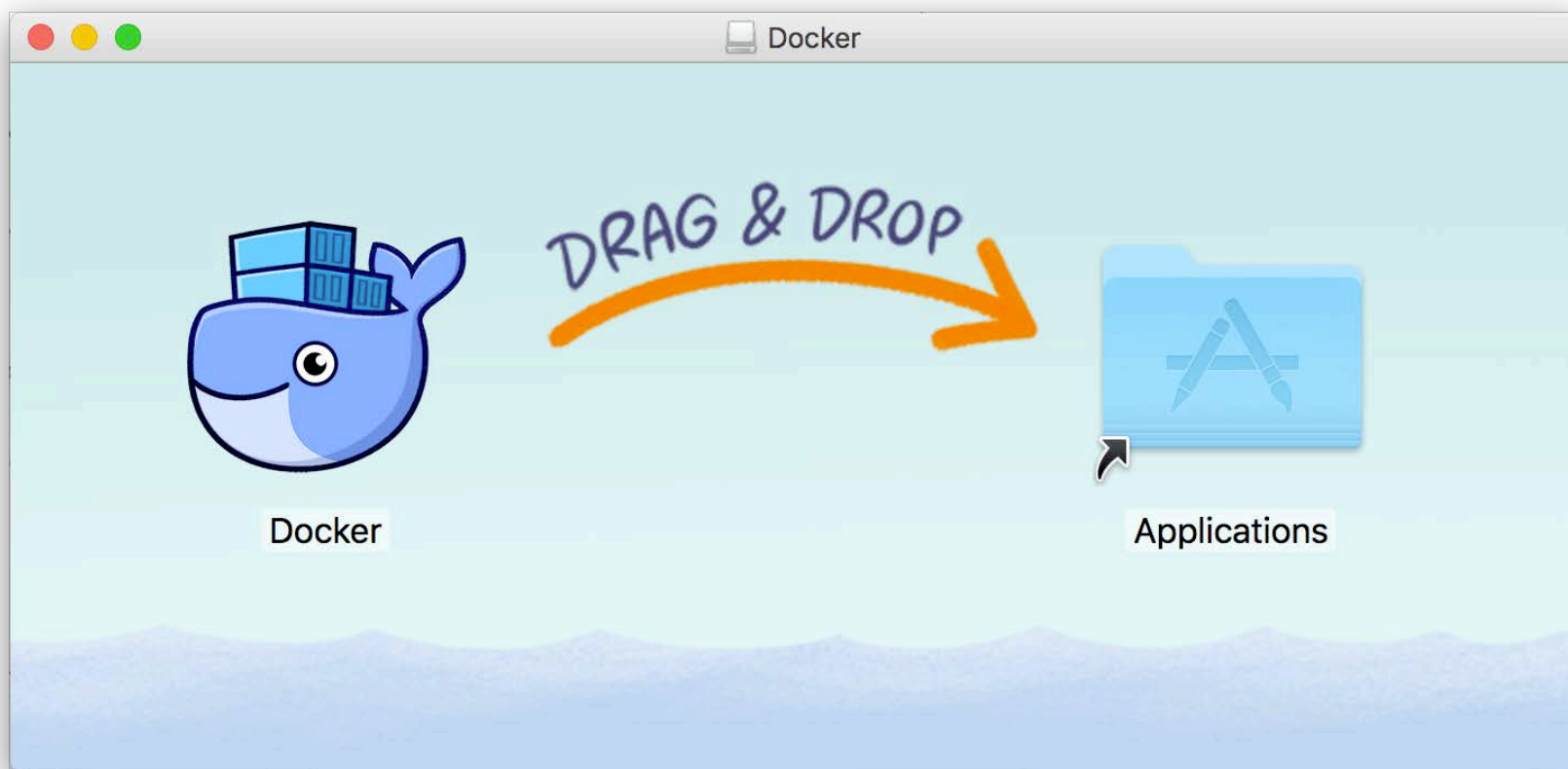
- ❖ 将 Docker 设置为自动启动的服务:
- ❖ `sudo systemctl enable docker && sudo systemctl start docker`

- ❖ 安装完毕后, 如果希望以非管理员身份访问 Docker:
- ❖ `sudo usermod -aG docker <username>`

- ❖ macOS Mojave
- ❖ 使用 Homebrew:
- ❖ `brew cask install docker`
- ❖ 如果上述文件下载较慢, 可以使用 DaoCloud 的镜像:
- ❖ https://download.daocloud.io/d/Docker_Mirror/Docker_for_Windows_Mac/17.03.1-ce/docker-for-mac-stable.dmg

❖ macOS Mojave

- ❖ 下载完毕后，将 Docker 拖入应用程序目录

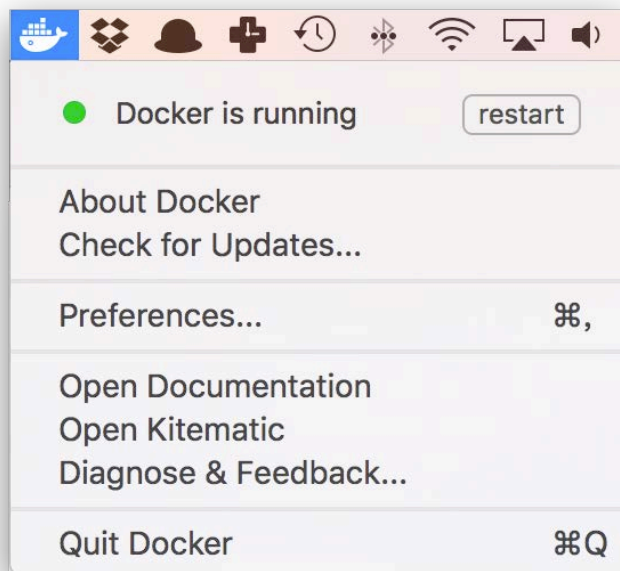


搭建实验环境

安装 Docker (macOS Mojave)

❖ macOS Mojave

- ❖ 从应用中找到 Docker 图标并点击运行
- ❖ 运行之后，会在右上角菜单栏看到多了一个鲸鱼图标，这个图标表明了 Docker 的运行状态
- ❖ 每次点击鲸鱼图标会弹出操作菜单



- ❖ 镜像加速器
- ❖ 国内从 Docker Hub 拉取镜像有时会遇到困难，此时可以配置镜像加速器
- ❖ 国内很多云服务商都提供了国内加速器服务，例如：
- ❖ Azure 中国镜像：<https://dockerhub.azk8s.cn>
- ❖ 阿里云加速器：<https://cr.console.aliyun.com/cn-hangzhou/mirrors>
- ❖ 七牛云加速器：<https://reg-mirror.qiniu.com>
- ❖ 如有需要，请参考理论课讲义《03 - 虚拟化技术》第 68-71 页

搭建实验环境

Docker 基础应用

❖ docker run hello-world

```
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
1b930d010525: Pull complete
Digest: sha256:92c7f9c92844bbbb5d0a101b22f7c2a7949e40f8ea90c8b3bc396879d95e899a
Status: Downloaded newer image for hello-world:latest
```

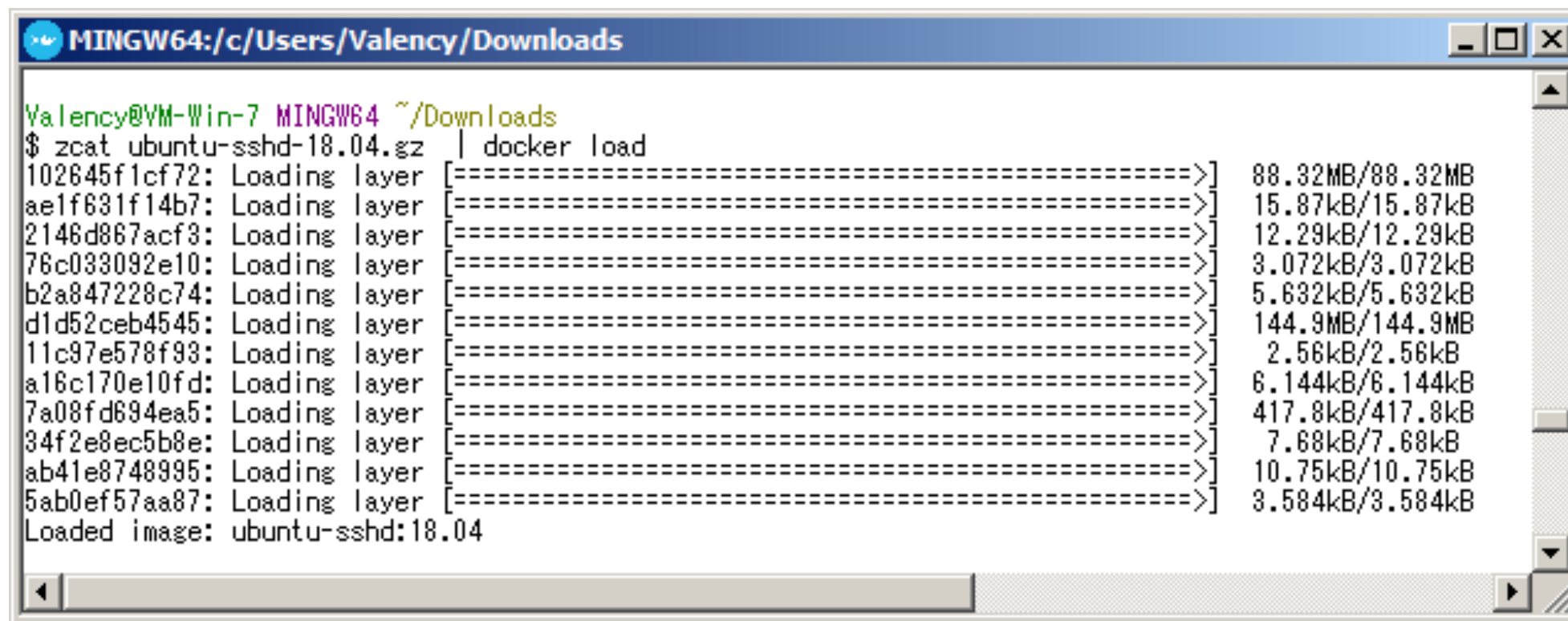
```
Hello from Docker!
This message shows that your installation appears to be working correctly.
```

```
...
```

```
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/
```

- ❖ 如果使用实验室电脑，该镜像已经导入 Docker，无需操作
- ❖ <https://www.avatarsys.org/reactor/data/ubuntu-sshd-18.04.gz>
- ❖ `zcat ubuntu-sshd-18.04.gz | docker load`

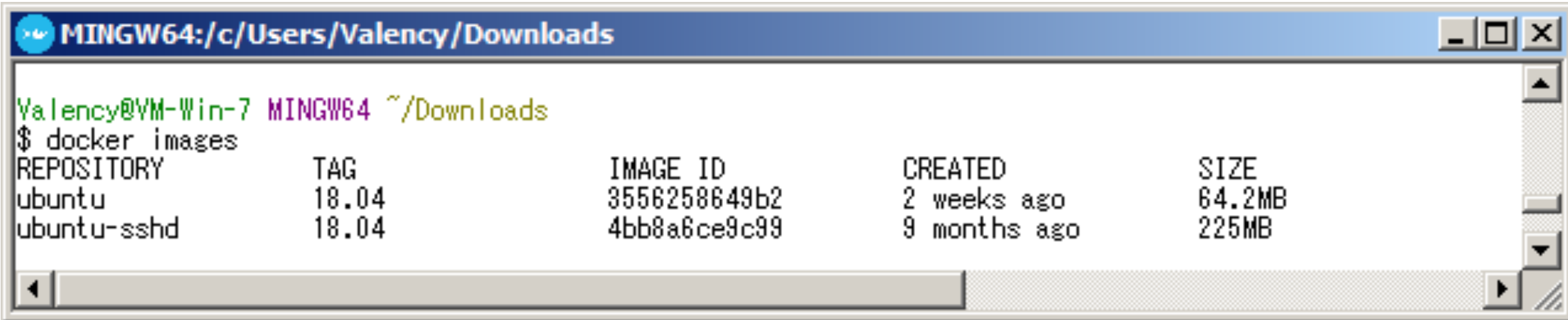


```
MINGW64:/c/Users/Valency/Downloads
Valency@VM-Win-7 MINGW64 ~/Downloads
$ zcat ubuntu-sshd-18.04.gz | docker load
102645f1cf72: Loading layer [=====>] 88.32MB/88.32MB
ae1f631f14b7: Loading layer [=====>] 15.87kB/15.87kB
2146d867acf3: Loading layer [=====>] 12.29kB/12.29kB
76c033092e10: Loading layer [=====>] 3.072kB/3.072kB
b2a847228c74: Loading layer [=====>] 5.632kB/5.632kB
d1d52ceb4545: Loading layer [=====>] 144.9MB/144.9MB
11c97e578f93: Loading layer [=====>] 2.56kB/2.56kB
a16c170e10fd: Loading layer [=====>] 6.144kB/6.144kB
7a08fd694ea5: Loading layer [=====>] 417.8kB/417.8kB
34f2e8ec5b8e: Loading layer [=====>] 7.68kB/7.68kB
ab41e8748995: Loading layer [=====>] 10.75kB/10.75kB
5ab0ef57aa87: Loading layer [=====>] 3.584kB/3.584kB
Loaded image: ubuntu-sshd:18.04
```

- ❖ 加载的镜像为 Ubuntu 18.04 的操作系统
- ❖ 对系统进行了更新，并预装了 OpenSSH Server
- ❖ 如有兴趣，可以查看本镜像的 Dockerfile:
- ❖ <https://github.com/valency/docker-sshd/blob/master/Ubuntu/Dockerfile>

❖ 查看已加载的镜像列表:

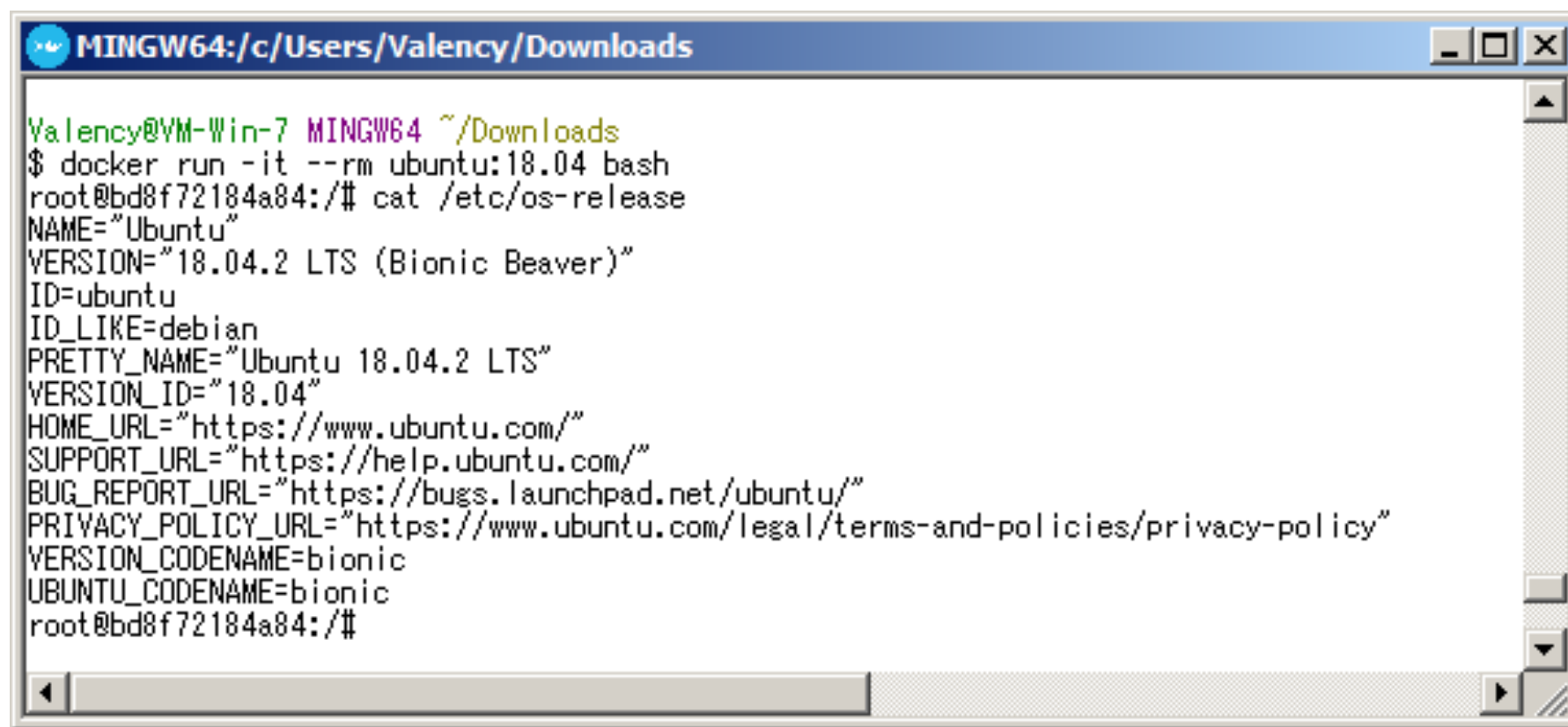
❖ docker images



```
MINGW64:/c/Users/Valency/Downloads

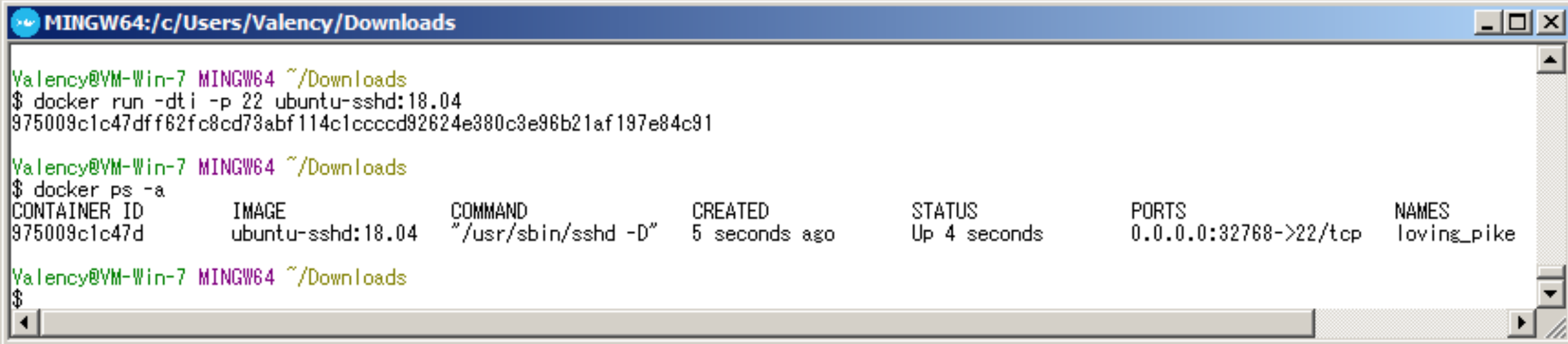
Valency@VM-Win-7 MINGW64 ~/Downloads
$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
ubuntu              18.04              3556258649b2       2 weeks ago        64.2MB
ubuntu-sshd         18.04              4bb8a6ce9c99       9 months ago       225MB
```

- ❖ 启用一个一次性的 Ubuntu 18.04 容器:
- ❖ `docker run -it --rm ubuntu:18.04 bash`
- ❖ 这行命令会启用一个一次性的 Ubuntu 18.04 虚拟机（容器），退出后会自动销毁



```
MINGW64:/c/Users/Valency/Downloads
Valency@VM-Win-7 MINGW64 ~/Downloads
$ docker run -it --rm ubuntu:18.04 bash
root@bd8f72184a84:/# cat /etc/os-release
NAME="Ubuntu"
VERSION="18.04.2 LTS (Bionic Beaver)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 18.04.2 LTS"
VERSION_ID="18.04"
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=bionic
UBUNTU_CODENAME=bionic
root@bd8f72184a84:/#
```

- ❖ 启动课程镜像的容器:
- ❖ `docker run -dti -p 22 ubuntu-sshd:18.04`
- ❖ 这行命令会启用一个在后台运行的 Ubuntu 18.04 虚拟机 (容器)

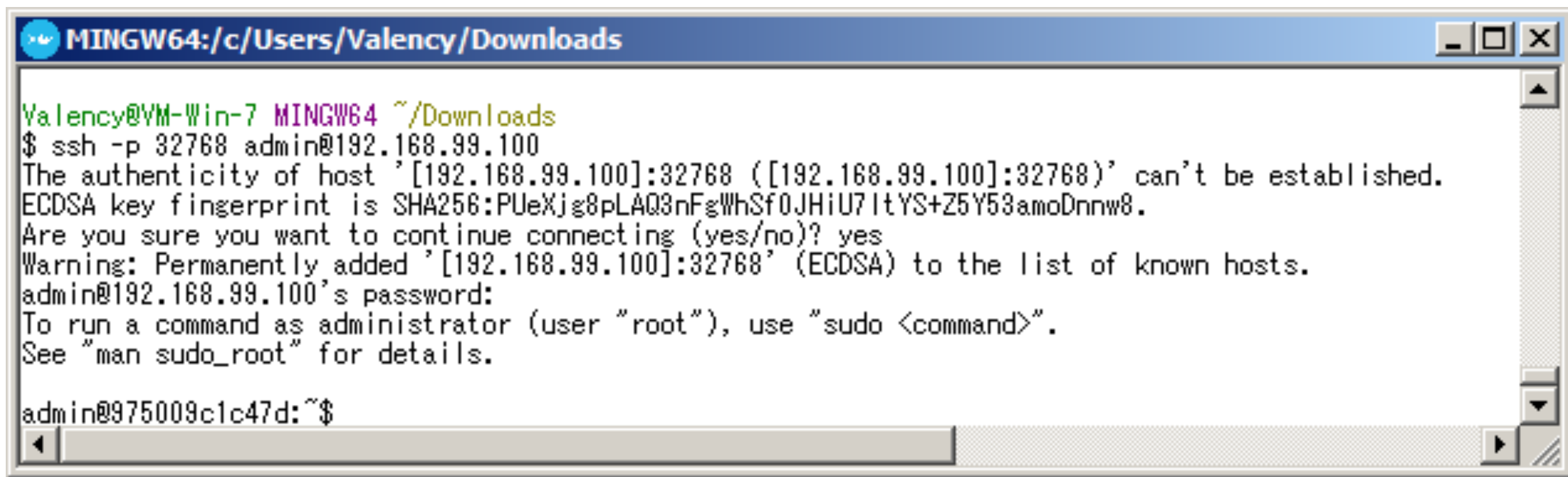


```
MINGW64:/c/Users/Valency/Downloads
Valency@YM-Win-7 MINGW64 ~/Downloads
$ docker run -dti -p 22 ubuntu-sshd:18.04
975009c1c47dff62fc8cd73abf114c1ccccd92624e380c3e96b21af197e84c91
Valency@YM-Win-7 MINGW64 ~/Downloads
$ docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
975009c1c47d	ubuntu-sshd:18.04	"/usr/sbin/sshd -D"	5 seconds ago	Up 4 seconds	0.0.0.0:32768->22/tcp	loving_pike

```
Valency@YM-Win-7 MINGW64 ~/Downloads
$
```

- ❖ 启动课程镜像的容器:
- ❖ 我们可以通过 SSH 远程连接到这个容器（密码是“screencast”）:
- ❖ `ssh -p 32768 admin@192.168.99.100`
- ❖ 注意：端口和 IP 地址均取决于机器配置，请仔细查看（通常是 127.0.0.1:32768）

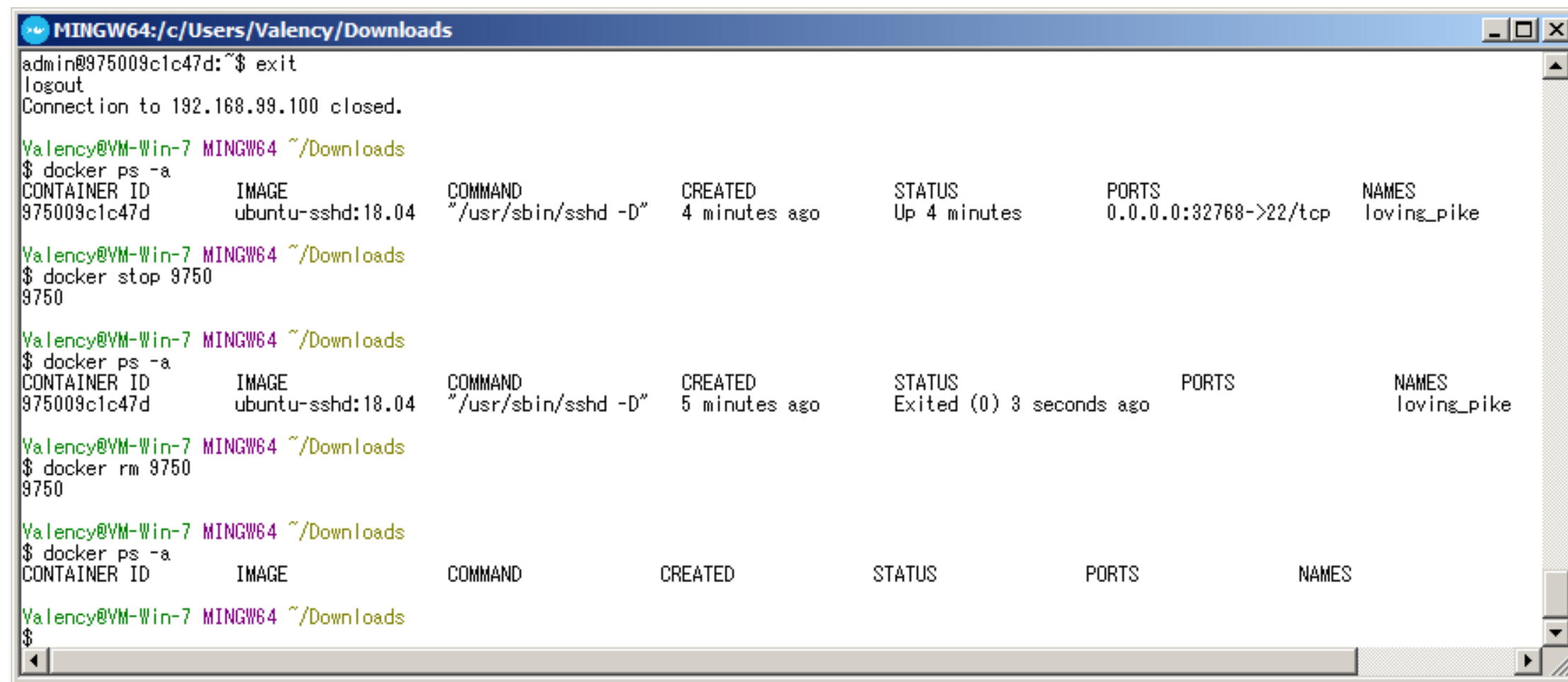


```
MINGW64:/c/Users/Valency/Downloads
Valency@VM-Win-7 MINGW64 ~/Downloads
$ ssh -p 32768 admin@192.168.99.100
The authenticity of host '[192.168.99.100]:32768 ([192.168.99.100]:32768)' can't be established.
ECDSA key fingerprint is SHA256:PUeXjg8pLAQ3nFgWhSf0JHiU7ItYS+Z5Y53amoDnnw8.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[192.168.99.100]:32768' (ECDSA) to the list of known hosts.
admin@192.168.99.100's password:
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

admin@975009c1c47d:~$
```

❖ 停止及销毁容器:

❖ `docker stop <container-id> && docker rm <container-id>`



```
MINGW64:/c/Users/Valency/Downloads
admin@975009c1c47d:~$ exit
logout
Connection to 192.168.99.100 closed.

Valency@VM-Win-7 MINGW64 ~/Downloads
$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
975009c1c47d       ubuntu-sshd:18.04  "/usr/sbin/sshd -D" 4 minutes ago      Up 4 minutes       0.0.0.0:32768->22/tcp loving_pike

Valency@VM-Win-7 MINGW64 ~/Downloads
$ docker stop 9750
9750

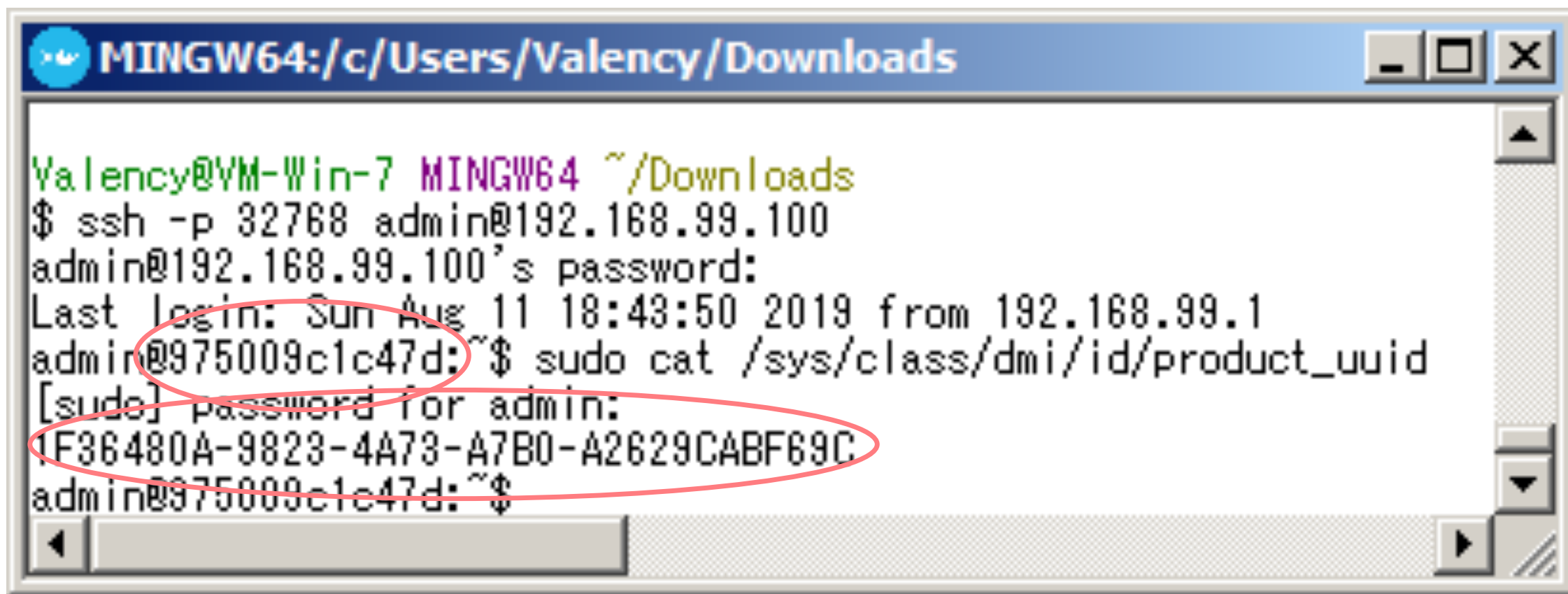
Valency@VM-Win-7 MINGW64 ~/Downloads
$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
975009c1c47d       ubuntu-sshd:18.04  "/usr/sbin/sshd -D" 5 minutes ago      Exited (0) 3 seconds ago           loving_pike

Valency@VM-Win-7 MINGW64 ~/Downloads
$ docker rm 9750
9750

Valency@VM-Win-7 MINGW64 ~/Downloads
$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES

Valency@VM-Win-7 MINGW64 ~/Downloads
$
```


- ❖ 成功创建并远程连接进入课程镜像的容器后，记录容器 ID
- ❖ 打印容器的 Product ID:
- ❖ `sudo cat /sys/class/dmi/id/product_uuid`

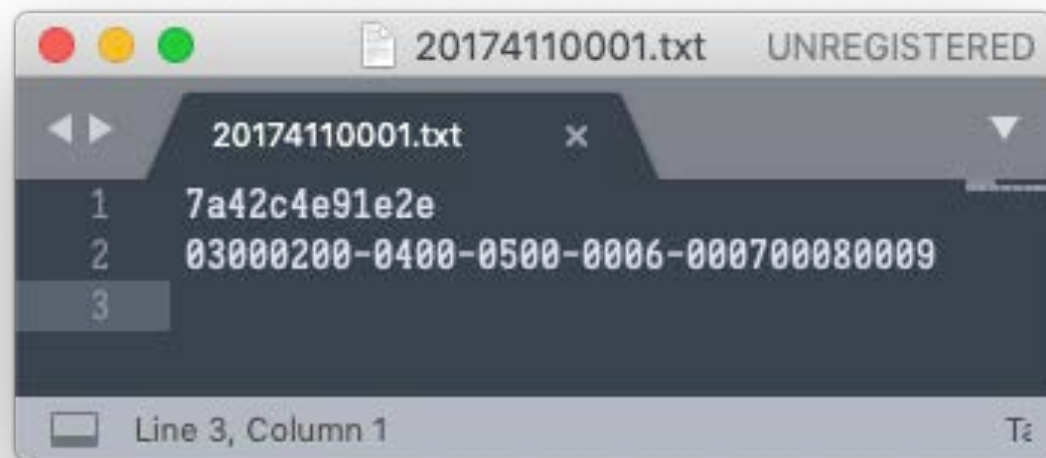


The image shows a Windows terminal window titled "MINGW64:/c/Users/Valency/Downloads". The terminal output is as follows:

```
Valency@VM-Win-7 MINGW64 ~/Downloads
$ ssh -p 32768 admin@192.168.99.100
admin@192.168.99.100's password:
Last login: Sun Aug 11 18:43:50 2019 from 192.168.99.1
admin@975009c1c47d:~$ sudo cat /sys/class/dmi/id/product_uuid
[sudo] password for admin:
1F36480A-9823-4A73-A7B0-A2629CABF69C
admin@975009c1c47d:~$
```

Red circles highlight the container ID "975009c1c47d" and the resulting Product ID "1F36480A-9823-4A73-A7B0-A2629CABF69C".

- ❖ 提交作业:
- ❖ 创建一个文本文件，命名为 “<student-id>.txt” ， 例如: 20174110001.txt
- ❖ 将容器 ID 和 Product ID 分两行写入文件，如右图所示
- ❖ 发送文件到: dingye@dgut.edu.cn
- ❖ 标题请注明: 046039 Assignment 1
- ❖ 正文请注明姓名和学号
- ❖ 不要发送其他任何文件，只需要发送一个 .txt 文件即可



```
20174110001.txt UNREGISTERED
20174110001.txt x
1 7a42c4e91e2e
2 03000200-0400-0500-0006-000700080009
3
Line 3, Column 1
```

Thanks!