实验环境搭建

准备开发、构建和运行环境

在开始实验之前，需要准备一个适合自己的开发、构建和运行环境，根据同学自身（和自己的电脑）情况不同，下面提供几种方案以供选择。

方案一：使用助教提供的虚拟机（适用于 Windows）

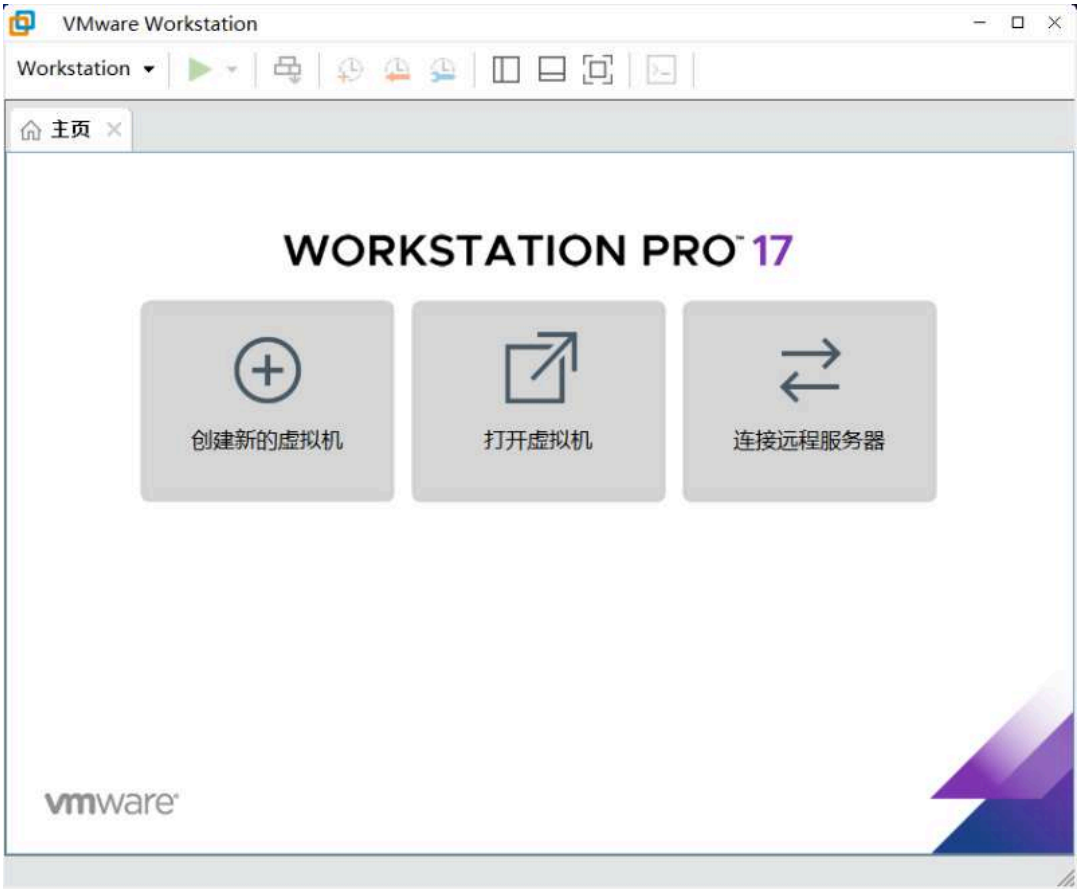
VMware Workstation 下载

VMware Workstation 是一个桌面虚拟化工具，允许用户在个人计算机上创建和运行多个虚拟机。VMware Workstation Pro 17.5.2 对个人使用完全免费，可以参考上交网络信息中心中的“使用说明”来完成下载和安装。

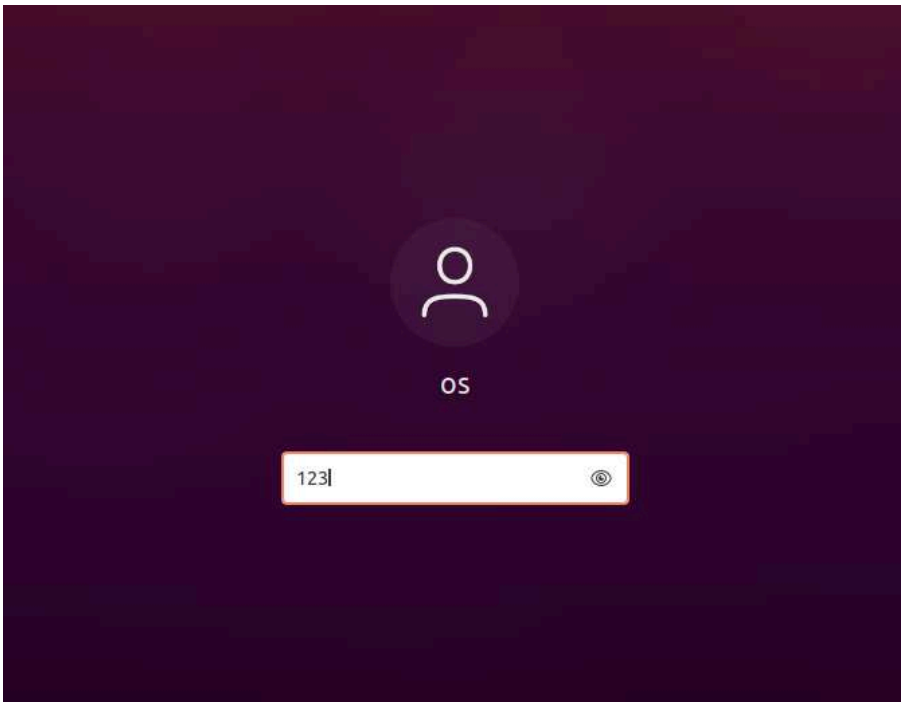
虚拟机镜像导入

助教提供的虚拟机镜像可以在交大云盘（提取码：`ics1`，大小：`4.55GB`，文件名：`os-lab-vm.ova`）内下载。

下载完成后，打开 VMware WorkStation 并选择“打开虚拟机”导入虚拟机镜像 `os-lab-vm.ova`。



虚拟机用户名为 `os`，密码均为 `123`。



MD5 校验

如果导入虚拟机失败，通常是因为镜像的完整性在下载中损坏，请检查你文件的大小以及 MD5，此文件的 MD5 为：

```
6c6f30518a6562080e92bbf0db05d8f5
```

在 windows 上获得文件的 md5 方式为在 `cmd` 或者 `powershell` 中输入如下命令：

```
certutil.exe -hashfile <os-lab-vm.ova-path> md5
```

若输出结果与上述 MD5 不匹配，则完整性受到破坏。请尝试重新下载虚拟机文件。

方案二：使用 Docker 并额外安装所需软件（适用于 Linux）

如果不想使用助教提供的虚拟机，可以在自己的 Linux 主机上安装所需的软件包，这里以 x86_64 平台的 Ubuntu 为例：

- 按照 Install Docker Engine on Ubuntu 的指示安装 Docker，以便使用构建镜像。
- 安装 `git, make, expect` 以便辅助构建。
- 安装 `binutils-aarch64-linux-gnu` 以便在 Docker 外使用 `aarch64-linux-gnu-objdump` 等工具。
- 安装 `gdb-multiarch` 以便在 Docker 外运行 `make gdb` 命令。
- 安装 `qemu-system-arm`，用于模拟 ARM 平台，运行 ChCore。
- 安装 `qemu-user`，用于运行炸弹程序。

方案三：使用 Docker 容器（适用于 Docker for Mac）

1. 按照 Install Docker Desktop on Mac 的指示安装 Docker
2. 启动 docker 镜像并后台运行:

```
$ docker run -dit --rm --name os-lab --mount type=bind,source=
<absolute/path/to/oslab-repo/on/your/host/machine>,target=/home/os-lab
jasonsjtu/oslab
```

3. 进入容器内:

```
$ docker exec -it os-lab bash
```

> 1. 需要把 `docker run` 指令中 `source=<xxx>` 的 `<xxx>` 替换为 *OS-Course-Lab 仓库*在**宿主机**中的路径
> 2. 需要在宿主机中完成代码编写，而在容器中进行项目构建、运行和测试

方案四：自己配置所有环境 (适用于 Linux，不建议使用)

以下是 ChCore Docker 构建环境的 `Dockerfile`，可参考该 `Dockerfile` 中所安装的工具，在你的环境中安装相应的**构建工具**。

```
# Dockerfile for ipads/chcore_builder.
FROM ubuntu:20.04
ENV DEBIAN_FRONTEND=noninteractive
ENV TZ=Asia/Shanghai
RUN apt-get update && \
    apt-get install -y \
    cmake=3.16.* \
    cmake-curses-gui=3.16.* \
    make \
    ninja-build \
    cpio \
    binutils \
    binutils-aarch64-linux-gnu \
    binutils-riscv64-linux-gnu \
    gcc=4:9.3.* \
    gcc-aarch64-linux-gnu=4:9.3.* \
    gcc-riscv64-linux-gnu=4:9.3.* \
    g++=4:9.3.* \
    g++-aarch64-linux-gnu=4:9.3.* \
    g++-riscv64-linux-gnu=4:9.3.* \
    grub-common \
    grub-pc-bin \
    xorriso && \
    apt-get clean && \
    rm -rf /var/lib/apt/lists/*
```

并留意在进行 ChCore lab 时，后面构建和运行时使用的命令的不同，需要添加形如以下的后缀:

```
$ make CHBUILD="./chbuild -l" # 更换 chbuild 命令， 通过 -l 指明使用宿主机而非 docker
环境
```

除了构建工具外，还需要安装其他**开发运行工具**，以 Linux 发行版 Ubuntu 举例，执行如下命令:

```
$ sudo apt update
$ sudo apt install git make expect
$ sudo apt install binutils-aarch64-linux-gnu
$ sudo apt install gdb-multiarch
$ sudo apt install qemu-system-arm
$ sudo apt install qemu-user
```

获取实验代码

拆弹实验代码发布在 GitHub 上，提交则直接使用 Canvas。

在你的实验环境的命令行中执行：

```
$ git clone https://github.com/SJTU-IPADS/OS-Course-Lab.git
$ cd os-course-lab
$ git checkout bomb-lab
```

完成实验

为了正确完成实验，你需要阅读仓库中的 `README.md` 和 `lab-instructions.pdf` 文件。其中，`README.md` 说明了如何让助教能够正确评阅你的实验结果，`lab-instructions.pdf` 文件则说明了实验的具体要求。另外，我们还提供了一份 `tools-tutorial.pdf` 文档，该文档对实验中可能用到的命令行工具做了比较详细的介绍。

请务必仔细阅读上述文档。为了更有效率地解答同学们在实验过程中遇到的问题，如果你的问题属于上述文档中已明确要求的步骤或介绍过的内容，可能不会被助教优先回复。

拆弹实验中只涉及简单的 git 操作，通常你只需要按顺序输入本文档中提供的命令即可。但后续的 ChCore 实验中，可能涉及到一些相对进阶的 git 操作，详见后续实验的文档。尽管在正常情况下，你也只需要按顺序输入命令即可完成后续实验中的 git 操作，但也有可能遇到一些问题。为了确保你能较为高效地解决所遇到的 git 问题，我们建议你花一定时间掌握 git 的基本概念和操作（这也是工作中极为重要的基础技能）。你可以参考以下参考资料：

- Git cheat sheet：https://education.github.com/git-cheat-sheet-education.pdf
- 《Pro Git》：https://git-scm.com/book/zh/v2
- learning git branching：https://learngitbranching.js.org/?locale=zh_CN
- 《提问的智慧》：https://github.com/ryanhanwu/How-To-Ask-Questions-The-Smart-Way/blob/main/README-zh_CN.md

---

提交实验结果

要完成本次实验，你只需要修改根目录下的 `ans.txt` 和 `student-number.txt` 两个文件。在将相应内容保存到上述两个文件后，执行下列命令来保存你的更改：

```
$ git add ans.txt student-number.txt
$ git commit -m "finish bomb-lab"
```

然后，执行下列命令打包你的实验代码，并将压缩包提交到 canvas 作业中即可。

```
$ tar --exclude=.git -zcf bomb-lab.tar.gz .
```

注意：出现 `tar: .: file changed as we read it` 提示是正常的，并不代表创建压缩包失败。

附录 1：Docker 换源

Docker 镜像的拉取速度有时会比较慢（甚至拉取不成功），可以考虑切换到国内的镜像源来增加拉取速度。虚拟机上为 Linux 系统，换源方式如下：

首先编辑 `/etc/docker/daemon.json` 文件（如果没有则需要创建此文件），写入以下内容：

```
{
    "registry-mirrors": [
        "https://docker.mirrors.aster.edu.pl",
        "https://dockerpull.com"
    ]
}
```

这些内容代表着不同的镜像源。

然后重启 docker 服务：

```
systemctl restart docker
```

最后使用如下命令检测换源是否成功：

```
docker info
```

输出中会 `Registry Mirrors` 内容。

附录 2：Docker 手动加载镜像

某些情况下换源可能仍无法拉取镜像，需要从交大云盘下载镜像手动进行加载。

下载镜像到本地后，执行以下命令即可：

```
docker load -i <镜像文件地址>
```

Linux/Windows 所需镜像链接：https://jbox.sjtu.edu.cn/l/w1kkMi (提取码：hrup)

MacOS 所需镜像链接： https://jbox.sjtu.edu.cn/l/813EYu (提取码：xtgt)
