虚拟机配置

第一次实验课是设置您将在课程的其余部分使用的虚拟机(VM)。这将使为编程任务安装所有依赖项变得容易，从而节省了安装单个包的繁琐工作，并确保开发环境是正确的。

- 第1步：安装Vagrant

Vagrant 是一种使用单个“Vagrantfile”中给出的指令自动配置虚拟机的工具。

macOS 和 Windows：您需要使用适合您计算机的正确下载链接来安装 Vagrant：https://www.vagrantup.com/downloads.html。

仅限 Windows：安装结束时系统会要求您重新启动计算机。 单击“是”立即执行此操作，或稍后手动重新启动，但不要忘记这样做，否则 Vagrant 将无法工作！

Linux：首先，通过运行命令 `sudo apt-get update` 确保您的软件包安装程序是最新的。 要安装 Vagrant，您的计算机上必须有“Universe”存储库； 运行 `sudo apt-add-repository Universe` 来添加它。 最后，运行 `sudo apt-get install vagrant` 安装 vagrant。

- 第2步：安装VirtualBox

VirtualBox 是一个 VM 提供商（管理程序）。

macOS 和 Windows：您需要使用适合您计算机的正确下载链接安装 VirtualBox：https://www.virtualbox.org/wiki/Downloads。 这些链接位于“VirtualBox 7.x.x platform packages”标题下。

对于 macOS Big Sur：安装后，您需要进入 `System Preferences > Security & Privacy` 并允许来自 Oracle 的系统软件更新（安装过程中可能会出现相关提示）。 然后，重新启动您的 Mac。

仅限 Windows：使用所有默认安装设置，但您可以取消选中“安装后启动 Oracle VirtualBox 7.x.x”复选框。

Linux：运行命令 `sudo apt-get install virtualbox`。 如果 `virtualbox` 尚未添加到您的软件包存储库中（如果 apt-get 提示“virtualbox has no installation candidate”），请按照此处的说明进行操作：https://www.virtualbox.org/wiki/Linux_Downloads

注意：这还将在您的计算机上安装 VirtualBox 应用程序，但您永远不需要运行它，尽管它可能会有所帮助（请参阅步骤 6）。

- 步骤 3：安装 Git（以及 Windows 上支持 SSH 的终端）

Git 是一个分布式版本控制系统。

macOS 和 Windows：您需要使用适合您计算机的正确下载链接来安装 Git：https://git-scm.com/downloads。

仅限 macOS：打开 .dmg 安装文件后，您将看到一个包含 .pkg 文件（即安装程序）的 Finder 窗口。 正常打开它可能会提示您无法打开，因为它来自身份不明的开发者。 要覆盖此保护，请右键单击 .pkg 文件并选择“打开”。 这将显示一个提示，询问您是否确定要打开它。 选择“是”。 这将带您进入（直接）安装。

仅限Windows：在安装过程中，您将有许多选项可供选择； 对于本课程来说，使用所有默认值就足够了（您可以取消选中最后的“查看发行说明”）。 安装包括一个支持 SSH 的 Bash 终端，通常位于 `C:\Program Files\Git\bin\bash.exe`。 您应该使用它作为本课程中的终端，除非您更喜欢另一个支持 SSH 的终端（命令提示符将不起作用）。 随意创建它的快捷方式； 但是，将可执行文件复制并粘贴到其他地方将不起作用。

Linux：`sudo apt-get install git`。

- 第 4 步：安装 X 服务器

您将需要一个 X Server 来向虚拟机输入命令。

macOS：安装 `XQuartz` https://www.xquartz.org/。 您需要注销并重新登录才能完成安装（如最后的提示所述）。

Windows：安装 `Xming` https://sourceforge.net/projects/xming/。 使用默认选项并取消选中最后的“Launch Xming”。

Linux：X 服务器已预安装！

- 第 5 步：克隆课程 Git 存储库

打开您的终端（如果使用 Windows，请使用步骤 3 中提到的终端）并 cd 到计算机上要保存本课程文件的任意位置。

运行 git clone https://github.com/kyleatprinceton/COS461-Public 从 GitHub 下载课程文件。

`cd COS461-Public/assignments` 进入课程作业目录。

- 第 6 步：使用 Vagrant 配置虚拟机

从您刚刚进入的 `assignments` 目录中，运行命令 `vagrant up` 来启动虚拟机并根据 Vagrantfile 对其进行配置。 您可能需要等待几分钟。 您可能会看到红色的warnings/errors，例如“default: dpkg-preconfigure: unable to re-open stdin: No such file or directory”，但您不必担心它们。

注意 1：以下命令允许您随时停止虚拟机（例如当您完成当天的作业时）：

`vagrant suspend`： 将保存虚拟机的状态并停止它。
`vagrant halt`：将正常关闭虚拟机操作系统并关闭虚拟机电源。
`vagrant destroy`：将从系统中删除虚拟机的所有痕迹。 如果您在虚拟机上保存了重要文件（例如您的作业解决方案），请不要使用此命令。

此外，命令 `vagrant status` 将允许您检查机器的状态，以防您不确定（例如正在运行、关闭电源、已保存...）。 您必须位于包含 Vagrantfile 的目录的某个子目录中才能使用上述任何命令，否则 Vagrant 将不知道您所指的是哪个虚拟机。

注意 2：步骤 2 中安装的 VirtualBox 应用程序提供了一个可视化界面作为这些命令的替代方案，您可以在其中查看虚拟机的状态并打开/关闭其电源或保存其状态。 不过，不建议使用它，因为它没有与 Vagrant 集成，并且输入命令应该不会慢。 它也不是初始 vagrant up 的替代方案，因为这会创建虚拟机。

- 第 7 步：测试 SSH 到 VPN

从终端运行 `vagrant ssh`。 这是您每次想要访问虚拟机时都会使用的命令。 如果有效，您的终端提示符将更改为 `vagrant@cos461:~$`。 所有进一步的命令都将在虚拟机上执行。 然后，您可以运行 `cd /vagrant` 来访问常规操作系统和虚拟机之间共享的课程目录。

由于这种共享目录结构，Vagrant 特别有用。 您无需将文件复制到虚拟机或从虚拟机复制文件。 Vagrantfile 所在的分配目录中的任何文件或目录都会在您的计算机和虚拟机之间自动共享。 这意味着您可以从 VM 外部使用您选择的 IDE 来编写代码（但仍然必须在 VM 内构建和运行）。

命令 logout 将随时停止 SSH 连接。

Windows 用户的额外注意事项
DOS (Windows) 和 Unix (Linux/MacOS) 中的行结尾符号不同。 在前者中，它们由回车符和换行符（CRLF 或“\r\n”）表示，而在后者中，它们仅由换行符（LF 或“\n”）表示。 鉴于您从 Windows 运行 `git pull`，git 会检测您的操作系统并在下载时向文件添加回车符。 这可能会导致运行 Ubuntu (Unix) 的虚拟机内出现解析问题。 幸运的是，这似乎只影响我们为测试编写的 shell 脚本（*.sh 文件）。 Vagrantfile 设置为自动将所有文件转换回 Unix 格式，因此您不必担心这一点。 但是，如果您想编写/编辑 shell 脚本来帮助自己进行测试，或者如果您在处理其他类型的文件时遇到此问题，请使用预安装的 dos2unix 程序。 运行 dos2unix [file] 将其转换为 Unix 格式（在 VM 中编辑/运行之前），并运行 unix2dos [file] 将其转换为 DOS 格式（在 Windows 上编辑之前）。 从虚拟机运行时需要执行此操作的一个很好的提示是一些涉及 ^M（回车符）的错误消息。 在 Windows 上编辑时需要执行此操作的一个很好的提示是缺少新行。 请记住，仅当您想要编辑 shell 脚本时才需要执行此操作。