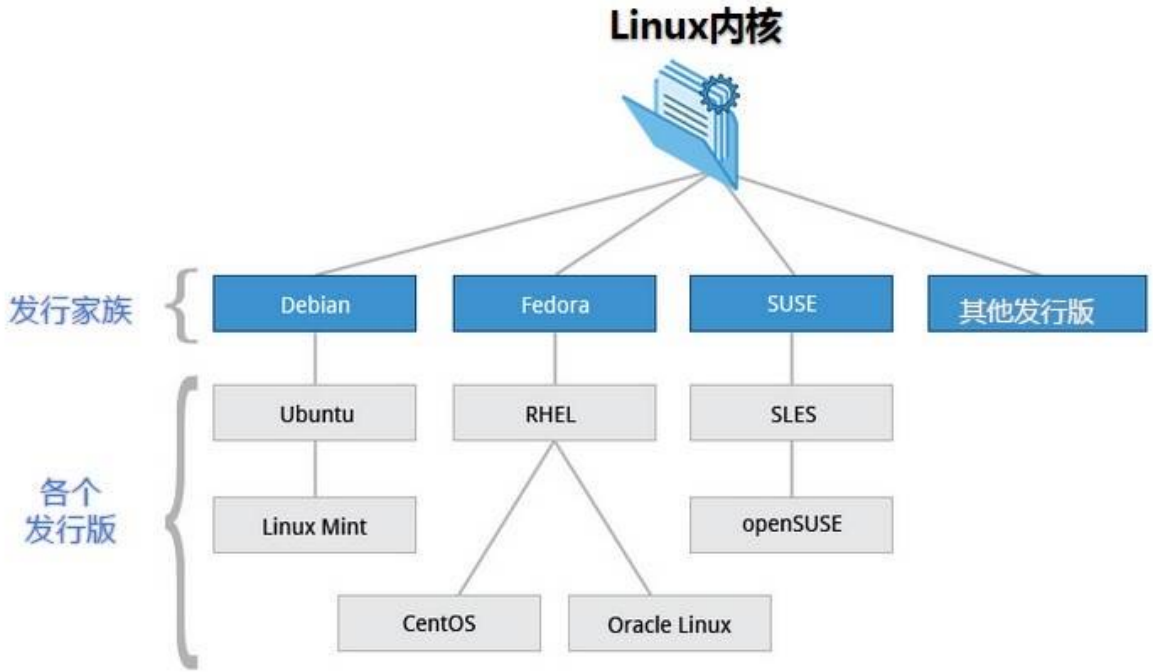
工具介绍和环境配置

廖东海

ctrlz.donghai@gmail.com

Linux

- Linux 内核最初只是由芬兰人林纳斯·托瓦兹（Linus Torvalds）在赫尔辛基大学上学时出于个人爱好而编写的。
- Linux 是一套免费使用和自由传播的类 Unix 操作系统，是一个基于 POSIX 和 UNIX 的多用户、多任务、支持多线程和多 CPU 的操作系统。
- Linux 能运行主要的 UNIX 工具软件、应用程序和网络协议。它支持 32 位和 64 位硬件。Linux 继承了 Unix 以网络为核心的设计思想，是一个性能稳定的多用户网络操作系统。



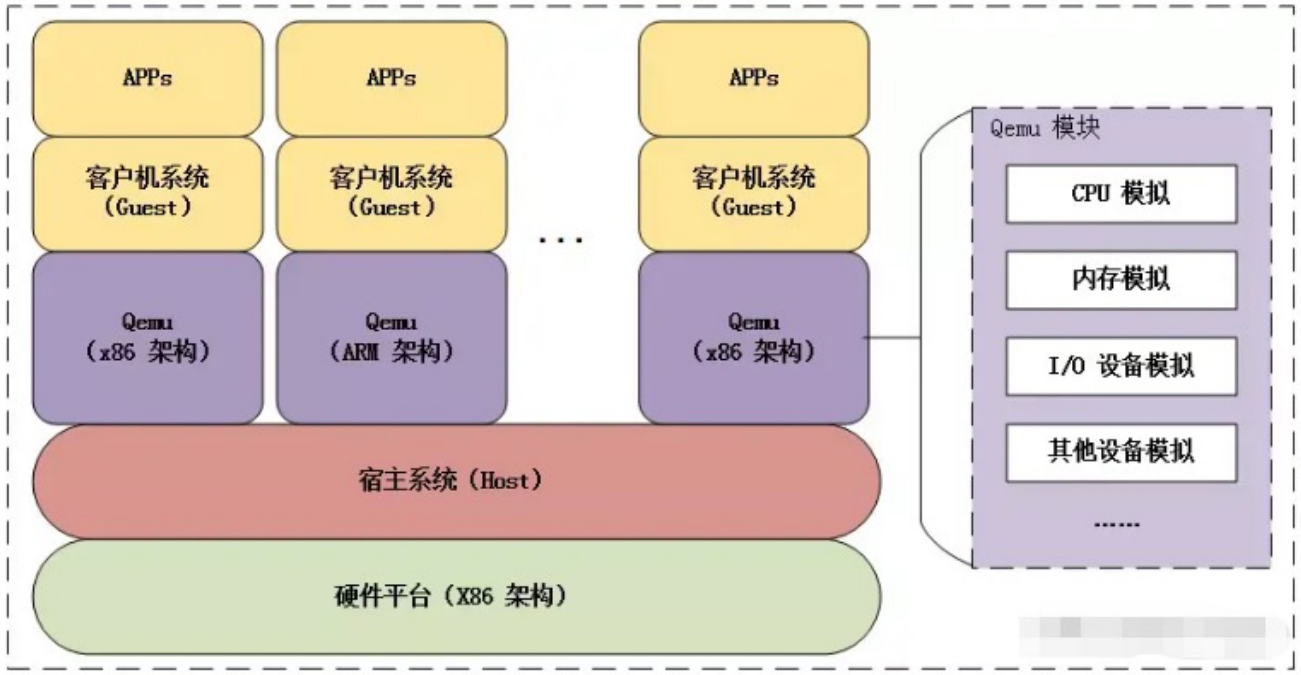
Linux

界面	界面统一，外壳程序固定所有 Windows 程序菜单几乎一致，快捷键也几乎相同	图形界面风格依发布版不同而不同，可能互不兼容。GNU/Linux 的终端机是从 UNIX 传承下来，基本命令和操作方法也几乎一致。
驱动程序	驱动程序丰富，版本更新频繁。默认安装程序里面一般包含有该版本发布时流行的硬件驱动程序，之后所出的新硬件驱动依赖于硬件厂商提供。对于一些老硬件，如果没有了原配的驱动有时很难支持。另外，有时硬件厂商未提供所需版本的 Windows 下的驱动，也会比较头痛。	由志愿者开发，由 Linux 核心开发小组发布，很多硬件厂商基于版权考虑并未提供驱动程序，尽管多数无需手动安装，但是涉及安装则相对复杂，使得新用户面对驱动程序问题（是否存在和安装方法）会一筹莫展。但是在开源开发模式下，许多老硬件尽管在 Windows 下很难支持的也容易找到驱动。HP、Intel、AMD 等硬件厂商逐步不同程度支持开源驱动，问题正在得到缓解。
使用	使用比较简单，容易入门。图形化界面对没有计算机背景知识的用户使用十分有利。	图形界面使用简单，容易入门。文字界面，需要学习才能掌握。
学习	系统构造复杂、变化频繁，且知识、技能淘汰快，深入学习困难。	系统构造简单、稳定，且知识、技能传承性好，深入学习相对容易。
软件	每一种特定功能可能都需要商业软件的支持，需要购买相应的授权。	大部分软件都可以自由获取，同样功能的软件选择较少。

Linux 命令大全 | 菜鸟教程 (runoob.com)

Qemu

- QEMU是一种通用的开源计算机仿真器和虚拟器。
- 当作为机器仿真器使用时，QEMU可以通过动态代码翻译机制（dynamic translation）在不同的机器上仿真任意一台机器（例如ARM板），并执行不同于主机架构的代码。
- 而当QEMU用作虚拟器时，QEMU的优点在于其实纯软件实现的虚拟化模拟器，几乎可以模拟任何硬件设备，但是也正因为QEMU是纯软件实现的，因此所有指令都需要QEMU转手，因此会严重的降低性能。



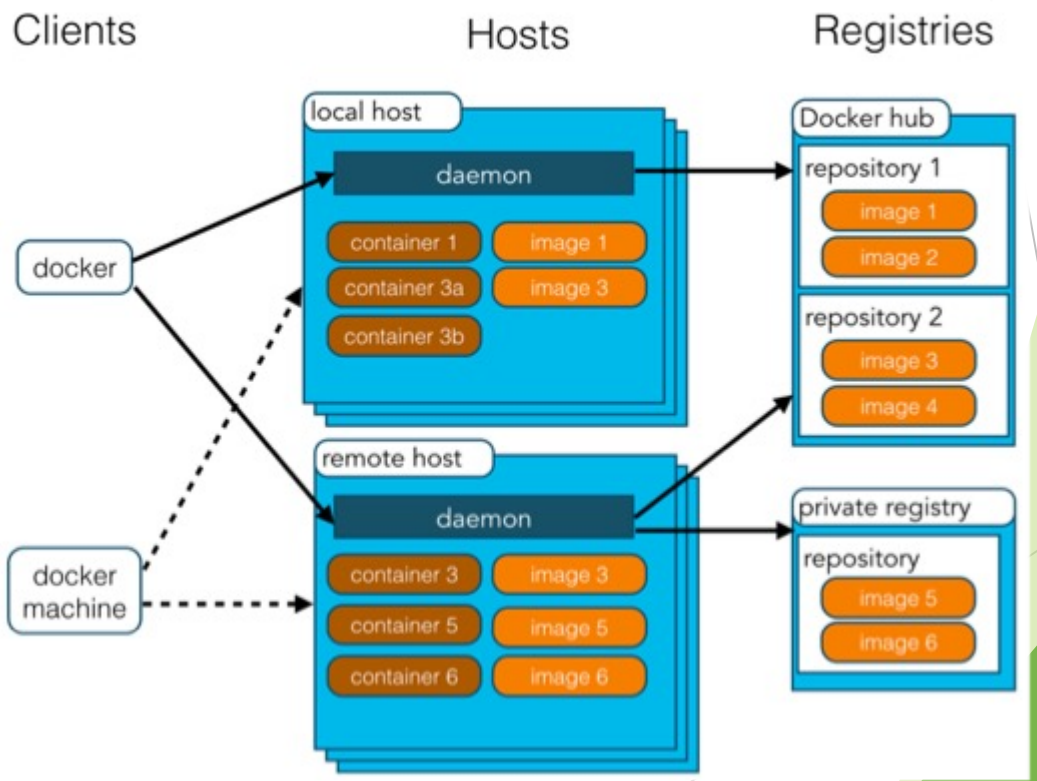
交叉编译工具链

- 本地编译可以理解为，在当前编译平台下，编译出来的程序只能放到当前平台下运行。平时我们常见的软件开发，都是属于本地编译。
- 交叉编译可以理解为，在当前编译平台下，编译出来的程序能运行在体系结构不同的另一种目标平台上，但是编译平台本身却不能运行该程序：
- **为什么会有交叉编译：**
  - 目标平台的运行速度往往比主机慢得多，许多专用的嵌入式硬件被设计为低成本和低功耗，没有太高的性能。
  - 整个编译过程是非常消耗资源的，嵌入式系统往往没有足够的内存或磁盘空间。
  - 即使目标平台资源很充足，可以本地编译，但是第一个在目标平台上运行的本地编译器总需要通过交叉编译获得。
  - 一个完整的Linux编译环境需要很多支持包，交叉编译使我们不需要花时间将各种支持包移植到目标板上。

Docker

- Docker 是一个开源的应用容器引擎，基于 Go 语言 并遵从 Apache2.0 协议开源。
- Docker 可以让开发者打包他们的应用以及依赖包到一个轻量级、可移植的容器中，然后发布到任何流行的 Linux 机器上，也可以实现虚拟化。
- Docker 包括三个基本概念:

- **镜像（Image）**：Docker 镜像（Image），就相当于是一个 root 文件系统。比如官方镜像 ubuntu:16.04 就包含了完整的一套 Ubuntu16.04 最小系统的 root 文件系统。
- **容器（Container）**：镜像（Image）和容器（Container）的关系，就像是面向对象程序设计中的类和实例一样，镜像是静态的定义，容器是镜像运行时的实体。容器可以被创建、启动、停止、删除、暂停等。
- **仓库（Repository）**：仓库可看成一个代码控制中心，用来保存镜像。



Docker Hello World | 菜鸟教程 (runoob.com)

sel4test环境配置（基于RISC-V）

- 后面有为大家准备好的docker环境。
- sel4test is a test suite for seL4.(seL4Test | seL4 docs)
- Host Dependencies | seL4 docs
- seL4/sel4test-manifest: Project to build and test seL4 for many different platforms (github.com)
  - Git-repo 是对 Git 构成补充的多（可以巨多的那种）代码库管理工具，简单说就是使用 Python 在 Git 基础上开发的一系列脚本命令。
- Supported Platforms | seL4 docs
- 推荐环境：
  - Windows：WSL or VM + docker
  - Mac: docker

上述的环境配置和代码拉取均需要魔法，有魔法的同学可以尝试自己配置，对于没有魔法的同学，这里有配置好的docker环境和拉取打包的最新test代码。（**密码：cy8qbc**）

VS code配置

- 在docker环境下需要与宿主机共享code目录，在宿主机下用vscode阅读和修改code源码，因此在启动容器时需要手动映射目录：
  - `docker run -it -v ./code:/home/seL4/workSpace/code 192059c7d7af /bin/bash`
- sel4 kernel用python脚本和自定义的proto文件自动生成了很多头文件，因此需要配置vscode进行build后查看源码，而默认使用的平台是x86，因此需要对build配置平台选项：

.vscode > {} settings.json > ...

```
{
    "cmake.configureOnOpen": true,
    "cmake.configureSettings": {
        "PLATFORM": "spike",
        "SIMULATION": true,
    },
    "cmake.sourceDirectory": "${workspaceFolder}/projects/sel4test"
}
```

- 编译和运行的命令在分享链接的README.md中。

问答