Linux简介

1 Unix&Linux的发展历史

1969年以前：Multics

早期的操作系统为批处理操作系统，输入设备只有卡片阅读机，输出设备只有打印机。程序员是在读卡纸上进行编程的，然后通过打孔机进行打孔，打孔表示 1，没打孔表示 0。这样的操作系统有个缺陷：如果计算机需要处理多个任务，那么后面的任务需要排队等待!

1960年初，麻省理工（MIT）开发了划时代的兼容分时系统 CTSS（Compatible Time-Sharing System），它允许操作系统在多个任务之间来回切换。那时候计算资源非常宝贵，都是通过终端连接主机的形式来共享计算资源的。这样，兼容分时操作系统的优势就体现出来了：它让用户感觉是在独享主机的资源。不过那时最先进的主机也只能支持 30 个左右的终端机。

为了增强大型主机的资源利用率，让主机资源可以同时被更多用户使用，1965年，贝尔实验室（Bell Lab）、麻省理工学院（MIT）以及通用电器公司（GE）共同发起了 Multics（MULTiplexed Information and Computing System）计划，Multics 是一个全面通用的分时操作系统，目的是让大型主机可以同时支持 300 个以上的终端机。不幸的是，在 1969年，由于进度落后，资金短缺，贝尔实验室认为 Multics 项目不可能取得成功，于是选择退出该计划。

Multics 计划其实并不算失败，它后面也取得了不俗的成就。不过这个我们且按下不表，因为我们要讲另一个更加波澜壮阔的历史。

1969年：Ken Thompson 发明了 Unics

1969年 Bell Lab 退出了 Multics 计划，参与该项目的 Ken Thompson 和 Dennis Ritchie 等人回到了贝尔实验室。由于比较空闲，这一年 Ken Thompson 写了一个游戏——太空旅行（Space Travel）。但是 Bell Lab 当时使用的还是批处理操作系统，所以玩游戏不是很方便。后来，Ken Thompson 在库房发现了一台老旧的且被闲置的 PDP-

7 小型机。于是他将游戏移植到该机器上，但是发现游戏在新机器上运行地很慢。于是 Ken Thompson 就准备针对这台机器重新写一个操作系统内核。

8 月份左右 Thompson 有一个月的休假期，而且他的妻子孩子也回老家探亲了。于是 Thompson 就利用这一个月时间，以及他从 Multics 项目中获得的灵感，将庞大而复杂的 Multics 做了大幅地简化。实验室的同事也因此戏称这个系统为 Unics。

Thompson 在设计 Unics 时，有两个重要的思想：

- 所有的程序或者系统设备都是文件 (一切皆文件)。
- 程序的目的只有一个，那就是有效地完成目标。

这就是著名的 KISS 原则（Keep it simple stupid），这也是 "Unix 哲学"的根本原则。有一本著作叫《The Art of Unix》就是讲 Unix 的设计原则的。

而也是在这一年，在地球的另一端 —— 芬兰，Linus Travalds 诞生了。

1973**年：Unix 内核正式诞生**

由于 Thompson 写的 Unics 操作系统实在太好用了，在贝尔实验室内部广为流传，并且经过多次改版。但是因为 Unics 是以汇编语言编写的，高度依赖硬件，而且当时计算机机器架构都不太相同，所以每次安装 Unics 系统，都要重新修改汇编代码，很不方便！

于是 Thompson 与 Dennis Ritchie 合计将 Unics 用高级语言进行改写，从而提高系统的可移植性。他们先选择了 BCPL ，然后又尝试PASCAL，发现编译出来的内核性能都不是很好。于是，两人 (主要是Dennis Ritchie) 决定自己开发一套新的语言：这就是大名鼎鼎的 C 语言。

1973年，Dennis Ritchie 用 C 语言重写了 Unics 的内核，最后发行出 Unix 的正式版本。 “重写”说起来简单，实际上并不是简单的汇编到C 语言的翻译，这里边包含了很多的发明和创造。比如，在此期间引入的 “pipe” (管道) 功能后来成为了 Unix 的一大优势。而管道的概念也是由贝尔实验室的 Douglas McIlroy (当时 Bell Lab 的主管) 发明的。

1977**年：Unix 的重要分支 BSD 诞生**

贝尔实验室隶属于 AT&T，而 Unix 诞生时，AT&T 对 Unix 采取比较开放的态度。所以 Unix 诞生之后，便与学术界开始合作。其中最重要的便是与加州大学伯克利分校 (UC Berkeley) 的合作了。UC Berkeley 的 Bill Joy 在取得了 Unix 的内核源码后，便着手修改成适合自己机器的版本。同时在该版本增加了很多工具软件与编译程序，最终将它

命名为 Berkeley Software Distribution（BSD），BSD 是 Unix 很重要的一个分支。Bill Joy 也是 Sun Microsystems 公司的创办者之一！

1979年：AT&T 发布支持 x86 架构带商业版权的 System V 第七版

由于 Unix 的高度可移植性与强大的性能，加上当时没有版权纠纷，很多商业公司开始了 Unix 操作系统的开发。 他们将 Unix 操作系统移植到自家的硬件平台上，从而推出适合自家硬件平台的 Unix 系统，例如 AT&T 的 System V、IBM 的 AIX，HP 的 HP/UX ...

操作系统内核（Kernel）必须跟硬件密切配合，因为操作系统需要管理硬件资源。早期生产计算机硬件的公司还没有统一的标准，每一个公司生产的硬件都不相同。因此每一个公司必须为自己的计算机硬件开发合适的 Unix 系统。而且，每一个公司开发出来的 Unix 系统是不能在其它硬件架构下工作的。另外，由于没有厂商针对个人计算机设计 Unix 系统（因为那时个人计算机并不流行）。因此，在当时是没有支持个人计算机的 Unix 系统的。

直到 1979 年，AT&T 推出了 System V 第七版，这个情况才有点改善。这一版最主要的特色是：可以支持 x86 架构的个人计算机。也就是说个人计算机终于可以用上 Unix 系统了。

不过 AT&T 出于商业考虑，在 System V 第七版发行时，特别声明了不再对学生提供源代码！这就制造了 Unix 业界的紧张气氛，同时也点燃了商业纠纷的战火~

这又是另一个波澜壮阔的历史，感兴趣的同学可以自行查看相关资料。目前存活下来，且还在被商业使用的大概就只有 System V 和 BSD 这两系了。

1984年：Minix 系统，GNU 项目

System V 第七版的版权声明对学术界造成了严重的影响，特别是教操作系统的教授。从1984 年开始，Andrew Tanenbaum（谭宁邦）教授出于教学的考虑，决定自己写一个 Unix Like 的内核程序。为了避免版权纠纷，Andrew Tanenbaum 完全不看 Unix 的内核源码！这完全就是一个Mini 版的 Unix 系统，所以被称为 Minix。1986 年，Minix 完成编写，并于次年出版书籍《Operating Systems：Design and Implementation》（书的附录中有 Minix 的源码哦）。Tanenbaum 还有另一本很出名的书籍《Modern operating systems》。

Minix 并不是免费的，必须通过磁盘磁带进行购买。不过售价非常便宜，而且还会随赠 Minix 源码。Minix 系统收到了广大用户的欢迎，很多用户强烈要求 Andrew Tanenbaum 对 Minix 进行改进。但是 Andrew Tanenbaum 教授只是把 Minix 定位在教育用途上面，无心对Minix 进行持续开发。这才有了后面 Linux 的故事...

在 Andrew Tanenbaum 教授开始编写 Minix 的1984年，一个伟大的计划正在悄然诞生，这就是日后和 Linux 双剑合璧一起改变世界的GNU 项目。GNU 项目，是一个自由软件集体协作项目，它是由 MIT 的 Richard Mathew Stallman（RMS）发起的，它的目标是创建一套完全自由的 Unix 操作系统。GNU 是 "Gnu is Not Unix" 的递归缩写。

1985 年，为了避免 GNU 所开发的自由软件被其他人所利用而成为专利软件， 所以 Richard Stallman 与律师草拟了有名的通用公共许可证（General Public License，GPL），并且称呼为 copyleft（相对于专利软件的 copyright）。

GNU 项目领导了计算机产业轰轰烈烈的自由软件运动，该项目也开发出了许多高质量的免费软件，其中包括有名的 Emacs 编辑系统、Bash Shell 程序、GCC 系列编译程序、GDB 调试程序等等。不过，对于 GNU 的最初构想『建立一个自由的 Unix 操作系统』来说，有这些优秀的程序仍然是不够的，因为，当时并没有『自由的 Unix 内核』存在。所以这些软件只能在那些有专利的 Unix 平台上工作，直到 Linux 的出现。

1988年：POSIX 标准

POSIX 是 Portable Operating System Interface 的缩写，而 X 表示其对 Unix API 的传承。POSIX 这个名称是由 Richard Stallman 向 IEEE 协会提议的一个名字，它最初的名字为 IEEE-IX。

POSIX 标准源于 1985 年的一个项目，该项目旨在保证各个操作系统之间的兼容性。也就是说，如果两个系统都遵循 POSIX 标准的话，那么在一个系统上能够运行的软件，在另一个系统上也可以运行。1988年，发布了 POSIX 标准的第一个版本。

Linux 基本上逐步实现了 POSIX 兼容，但并没有参加正式的 POSIX 认证。

微软的 Windows NT 声称部分实现了 POSIX 标准。

1991年：Linux 0.0.2 版发布

1988 年，Linus Torvalds 进入了赫尔辛基大学，并选读了计算机科学系。在读期间，接触到了 Unix 系统。当时整个赫尔辛基只有一部最新的 Unix 系统，同时仅提供 16 个终端（terminal）。早期的计算机仅有主机具有运算功能，terminal 仅负责提供 Input/Output 而已。这种情况，实在很难满足托瓦兹的需求。不久之后，他就知道有一个类似 Unix 的操作系统，并且与 Unix 完全兼容，还可以在 Intel 386 机器上面运行 —— Minix。Linus Torvalds 贷款购买了一台 Intel 386 个人计算机，安装了 Minix 操作系统，并通过 Minix 操作系统附带的源代码，学习到了很多内核程序设计的理念。

由 Andrew Tanenbaum 教授只愿意将 Minix 系统用于教学，不愿意强化系统功能，而 Linus Torvalds 是那种欲求不满的人，因此他萌生自己编写操作系统的想法。而 GNU 计划产出的 Bash Shell 和 GCC 编译程序等自由软件，让托瓦兹能够顺利的编写内核程序。最终，在 1991 年，他写出了能够在 Intel 386 计算机上运行的内核，并将其上传

到网络上，提供免费下载。当时 Linus Torvalds 是将该内核放在 Linux 目录下以供下载，所以该系统又被称为 Linux 系统。

后来，为了让更多 Unix 系统上的软件能够运行在 Linux 上，托瓦兹参考 POSIX 标准规范修改 Linux，提高了与 Unix 系统的兼容性。

Linux 成功的因素

1. 继承自 Minix 系统，没有版权纠纷。
2. GNU 提供了所需的工具软件，bash shell，GCC，GDB 等。
3. 秉承开源的原则，并可以免费下载（借由 Internet 广为流传）。
4. 遵循 POSIX 标准，兼容 Unix 系统。
5. Linus Torvalds 的英明领导。

2 Linux发行版本

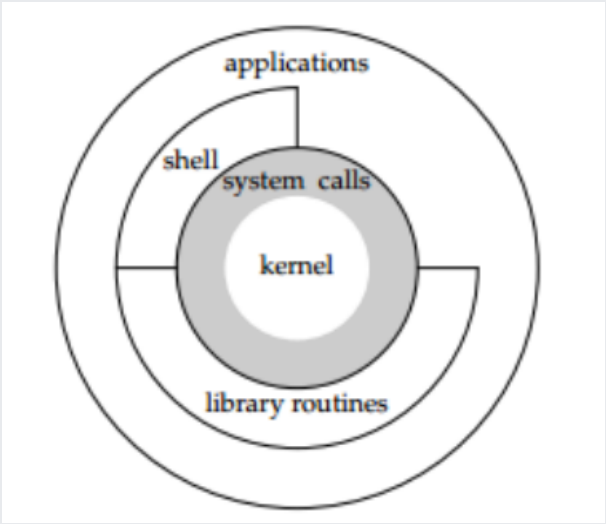
严格来说，Linux 是指操作系统的内核，但是如今 Linux 常用来指基于 Linux 的完整操作系统，内核则改以 Linux内核 称之。 通常情况下，Linux 被打包成供个人计算机和服务器使用的 Linux 发行版，我们可以简单的认为 Linux 发行版本就是 Linux内核和其它支撑软件的组合。

主流的发行版本有：

	免费版	收费版	包管理器
Debian	Ubuntu	Debian	apt
RedHat	CentOS	RedHat	yum
SuSE	...	...	...

3 Linux体系结构

Linux 的体系结构源自于 Unix，主要分为三层，从内到外依次是：内核 --> 系统调用 --> 应用层。



1. 内核(kernel)负责两个功能：管理计算机硬件资源；为上层应用程序提供运行环境。

   ```
   计算机的硬件资源有哪些?
       CPU，内存，外部设备
   内核模块：文件管理，进程调度，内存管理，网络通信，设备驱动...
   ```

2. 系统调用(system calls)：内核给上层应用程序提供的接口。

3. 库函数(library routines)：通常我们会把系统调用封装成库函数，主要的目的是方便程序员使用。系统调用往往设计得比较繁琐复杂，相对于系统调用而言，库函数的设计会更加友好。比如：`malloc`、`free`、`printf`、`scanf`等。

4. shell 是一个命令行解释器，它读取用户输入，然后执行命令，然后等待用户的下一次输入。

   ```
   for(;;) {
       read(cmd);
       execute(cmd);
   }
   命令：一般来说，就是一些简单的可执行程序。
   脚本：命令的集合。
   ```

   shell 拥有很多版本，我们将使用 bash(Bourne-again shell) ，它位于/bin/bash。