第 6讲 Linux 账户和登录安全运维、启动过程、内核配置

本讲主要内容

涉及3个部分：

- 账户和登录安全运维
- 启动过程
- 内核配置

账户和登录安全运维

账户安全是系统安全的第一道屏障，也是一项核心安全工作。

下面介绍一些在Linux server上常用到的加强账户安全的做法，主要包括：

- 删除特殊的用户和用户组
- 关闭系统不需要的服务
- 密码安全策略
- 合理使用su，sudo命令
- 删减系统登录欢迎信息
- 禁止 Control-Alt-Delete 键盘关闭命令

删除特殊的用户和用户组

Linux 系统安装完毕后，默认有很多不必要的用户和用户组。

如果不需要某些用户或用户组，应立即删除它们，以免被利用。

Linux系统中可以删除的默认用户和用户组大致如下：

可删除的用户：

- adm，管理用户
- lp，打印用户
- sync，同步

- shutdown，关机
- halt，悬挂
- news
- uucp，unix间拷贝
- operator
- games
- gopher

删除命令格式： `sudo userdel <某个用户名>`

可删除的用户组

- adm
- lp
- news
- uucp
- games
- dip
- pppusers
- popusers
- slipusers

删除命令格式： `sudo groupdel <某个用户组名>`

有时，某些用户仅仅作为进程调用或用户组调用，并不需要登录功能，此时可以禁止其登录功能。执行下列命令：

```
usermod -s /sbin/nologin <某个用户名（例如nagios）>
```

关闭不必要的服务

Linux Server 默认安装后会有一些不必要的服务，这些服务是自启动的。

从安全角度讲，开放的服务越多就越不安全，所以除非必要服务都应将其关闭。

具体关闭哪些服务，这需要考虑服务器的用途。

下面列出的服务往往是服务器运行所必须的，建议启动他们，保证系统正常运行。

- acpid
  - 电源管理服务
- apmd
  - 高级电源管理服务，监控电池性能

- kudzu
  - 检查硬件变更服务
- crond
  - linux 进程服务
- atd
  - 类似crond，类似windows计划任务
- keytables
  - 用于装载镜像键盘，视需而定
- iptables
  - 内置防火墙
- xinetd
  - 支持多种网络服务的核心守候进程
- xfs
  - x window桌面系统必要服务
- netword
  - 启动网络服务
- sshd
  - 远程加密访问
- syslog
  - 系统日志服务

而下面举出一些常见的无用服务，可以考虑关闭。

- anacron
  - 用来保证在系统关机时错过的定时任务可以在系统开机之后在执行
- auditd
  - 审计工具 ，常用来对文件修改进行监听
- autofs
  - 实现光驱，软盘等的动态自动挂载
- avahi-daemon
  - 运行在客户机上实施查找基于网络的Zeroconf service的服务守护进程。
  - 该服务可以为Zeroconf网络实现DNS服务发现及DNS组播规范。
- avahi-dnsconfd
  - 在没有 DNS 服务的局域网里发现基于 zeroconf 协议的设备和服务，与Bonjour类似
- bluetooth
- cpuspeed
- firstboot
  - Fedora系统的特有的调度任务
  - Fedora系统第一次启动时，执行一次特定任务

- gpm
  - 终端鼠标指针支持（无图形界面）
- haldaemon
  - 维护连接到系统的设备的数据库
- hidd
  - 管理所有可见的蓝牙设备
  - 维护键盘/鼠标等蓝牙输入设备
- ip6tables
  - IPv6软件防火墙
- ipsec
  - IPSec虚拟专用网络
- isdn
  - 专用数字线路，一种互联网的接入方式
- lpd
  - 打印机管理程序
- mcstrans
  - SELinux内核安全套件
- messagebus
  - 进程间通讯服务，与 DBUS 交互
- netfs
  - 自动挂载网络中的共享文件空间
- nfs
  - 标准文件共享方式
- nfslock
- nscd
  - 用户/用户组/DNS解析缓存服务
- pcscd portmap
  - 提供智能卡读卡器支持
- readahead_early
  - 预先加载特定的应用到内存中以提供性能
- restorecond
  - 给 SELinux 监测和重新加载正确的文件上下文
- rpcgssd
  - NFS v4
- rpcidmapd
  - NFS v4
- rstatd
  - 系统内核性能统计
- sendmail

    - IMAP/POP3邮件工具
- setroubleshoot
    - 服务器安全审计工具，从CentOS 6.x开始合并入auditd
- yppasswdd ypserv
    - 用于NIS用户修改服务器端的密码

关闭服务的操作命令为：

```
chkconfig --level 345 <服务名> off
```

执行完成后重启服务器。

chkconfig 命令说明

chkconfig命令检查、设置系统的各种服务。这是Red Hat公司遵循GPL规则所开发的程序，它可查询操作系统在每一个执行等级中会执行哪些系统服务，其中包括各类常驻服务。

注意：chkconfig不是立即自动禁止或激活一个服务，它只是简单的改变了符号连接。

选项：

- --add：增加所指定的系统服务，让chkconfig指令得以管理它，并同时在系统启动的叙述文件内增加相关数据；
- --del：删除所指定的系统服务，不再由chkconfig指令管理，并同时在系统启动的叙述文件内删除相关数据；
- --level<等级代号>：指定读系统服务要在哪一个执行等级中开启或关毕。
    - level选项可以指定要查看的运行级而不一定是当前运行级。对于每个运行级，只能有一个启动脚本或者停止脚本。
    - 当切换运行级时，init不会重新启动已经启动的服务，也不会再次去停止已经停止的服务。

level等级代号列表：

- 等级0表示：表示关机
- 等级1表示：单用户模式
- 等级2表示：无网络连接的多用户命令行模式
- 等级3表示：有网络连接的多用户命令行模式
- 等级4表示：不可用
- 等级5表示：带图形界面的多用户模式
- 等级6表示：重新启动

如何增加一个服务？

- 1.服务脚本必须存放在/etc/ini.d/目录下；

- 2. `chkconfig --add servicename` 在chkconfig工具服务列表中增加此服务，此时服务会被在/etc/rc.d/rcN.d中赋予K/S入口了；
- 3. `chkconfig --level 35 mysqld on` 修改服务的默认启动等级。

Linux 口令保护

在任何系统中，口令对安全性都起着非常重要的作用。不好的口令可能会导致组织的资源受到损害。为此，组织中的每个人，无论是普通用户还是管理员，都应该遵守口令保护策略。

Linux的口令策略存放在/etc/login.defs文件中。
密码策略存放在：/etc/pam.d/中

下面给出一些在选择口令或加固口令时必须遵循的规则：

口令的创建策略

应遵循如下规则：

- 一个用户在组织中的所有账号不能使用相同的口令。
- 所有与访问相关的口令都应该互不相同。
- 当同一个用户既有系统级账号又有普通账号时，系统级账号的口令一定与其他账号的口令不同。

口令的保护策略

应遵循如下规则：

- 口令应该被看做敏感的和机密的信息，因此不能与任何人分享。
- 不应该通过任何电子通讯工具，如电子邮件，来共享口令。
- 永远不要在手机上或者调查问卷中透露口令。
- 不要使用能向攻击者提供线索的口令提示。
- 不要与任何人分享公司的口令，包括行政人员、管理者、同事、甚至家庭成员。
- 不要将口令以书面形式存储在办公室的任何地方。如果将口令存储在移动设备上，那么一定要进行加密。
- 不要使用应用程序的记忆口令功能。
- 如果怀疑口令可能被泄露，那么要尽早上报安全事件并更改口令。

口令的更改策略

应遵循以下规则：

- 所有用户和管理员必须定期更改其口令，至少每季度修改一次。
- 组织的安全审计人员必须进行随机检查，检查任何用户的口令是否能够被猜出或者被破解。

PAM模块

Linux3中集成了一种叫做可插入式认证模块（Pluggable Authentication Modules）的安全验证方式，能够用它来完成上面所示的任务。

在Linux系统中，Login服务为用户提供系统登录服务，它提示用户输入相应的用户名和密码来验证用户的有效性，当使用PAM后，这个验证过程可以由PAM来代替。

可插入认证模块（简称PAM）具有可插入功能的一种独立于应用程序之外的验证方式。使用PAM后，应用程序可以不需要集成验证功能，而由PAM来完成。

PAM具有很大的灵活性，系统管理员可以通过它为应用程序自由选择需要使用的验证方式。

PAM 的各个模块一般存放在 /lib/security/ 或 /lib64/security/ 中，以动态库文件的形式存在（可参阅 dlopen(3)），文件名格式一般为 pam_*.so。

PAM 的配置文件可以是 /etc/pam.conf 这一个文件，也可以是 /etc/pam.d/ 文件夹内的多个文件。如果 /etc/pam.d/ 这个文件夹存在，Linux-PAM 将自动忽略 /etc/pam.conf。

/etc/pam.conf 类型的格式如下：

```
服务名称　工作类别　控制模式　模块路径　模块参数
```

/etc/pam.d/ 类型的配置文件通常以每一个使用 PAM 的程序的名称来命令。比如 /etc/pam.d/su，/etc/pam.d/login 等等。还有些配置文件比较通用，经常被别的配置文件引用，也放在这个文件夹下，比如 /etc/pam.d/system-auth。

这些文件的格式都保持一致：类型的格式如下

```
工作类别　控制模式　模块路径　模块参数
```

PAM模块工作类别

AM 的具体工作主要有以下四种类别（type）：account，auth，password 以及 session。

- account
  - 用于声明某用户能否使用某服务，但不负责身份认证。
  - 例如，可以检查用户能不能在一天的某个时间段登录系统、这个用户有没有过期、以及当前的登录用户数是否已经饱和等等。
  - 通常在登录系统时，如果连 account 这个条件都没满足的话，即便有密码也还是进不去系统的。
- auth：负责身份验证和授权
  - 一般来说，询问你密码的就是这个 type。

    - 假如你的验证方式有很多，比如一次性密码、指纹、虹膜等等，都应该添加在 auth 下。
    - auth 还用于赋权给用户某个组的组员身份等等。
- password
    - 负责密码相关策略。例如：“密码强度”策略设置等。
    - 注意，这里的密码不局限于 /etc/shadow 中的密码，有关认证 token 的管理都应该在此设置
    - 如果你使用指纹登录 Linux，在设置新指纹时，如果希望首先验证这是人的指纹而不是狗的指纹，也应该放在这里。
- session
    - 负责某个服务与用户的安全环境上下文。

PAM模块控制模式

用于定义各个认证模块在给出各种结果时 PAM 的行为，或者调用在别的配置文件中定义的认证流程栈。

该列有两种形式：

- 常见的“关键字”模式；
- 用方括号（[]）包含的“返回值=行为”模式。

“关键字”模式下，有以下几种控制模式：

- required
    - 如果本条目没有被满足，那最终本次认证一定失败，但认证过程不因此打断。
    - 整个栈运行完毕之后才会返回（已经注定了的）“认证失败”信号。
- requisite
    - 如果本条目没有被满足，那本次认证一定失败，而且整个栈立即中止并返回错误信号。
- sufficient
    - 如果本条目的条件被满足，且本条目之前没有任何required条目失败，则立即返回“认证成功”信号；
    - 如果对本条目的验证失败，不对结果造成影响。
- optional
    - 该条目仅在整个栈中只有这一个条目时才有决定性作用；
    - 否则无论该条验证成功与否都和最终结果无关。
- include
    - 将其他配置文件中的流程栈包含在当前的位置，就好像将其他配置文件中的内容复制粘贴到这里一样。
- substack
    - 运行其他配置文件中的流程，并将整个运行结果作为该行的结果进行输出。
    - 该模式和 include 的不同点在于认证结果的作用域：

        - 如果某个流程栈 include 了一个带 requisite 的栈，这个 requisite 失败将直接导致认证失败，同时退出栈；
        - 而某个流程栈 substack 了同样的栈时，requisite 的失败只会导致这个子栈返回失败信号，母栈并不会在此退出。

“返回值=行为”模式则较为复杂，但可以由此设计高度自定义的认证过程。

其格式如下：

```
[value1=action1 value2=action2 ...]
```

说明：

- valueN 的值是各个认证模块执行之后的返回值。
    - 有 success、user_unknown、new_authtok_reqd、default 等等数十种。
    - default 代表其他所有没有明确说明的返回值。
    - 返回值结果清单可以在 /usr/include/security/_pam_types.h 中找到。
- actionN 的值，确定哪一个验证规则能作为最终的结果。
    - ignore
        - 在一个栈中有多个认证条目的情况下，如果标记 ignore 的返回值被命中，那么这条返回值不会对最终的认证结果产生影响。
    - bad
        - 标记 bad 的返回值被命中时，最终的认证结果注定会失败。此外，如果这条 bad 的返回值是整个栈的第一个失败项，那么整个栈的返回值一定是这个返回值，后面的认证无论结果怎样都改变不了现状了。
    - die
        - 标记 die 的返回值被命中时，马上退出栈并宣告失败。整个返回值为这个 die 的返回值。
    - ok
        - 在一个栈的运行过程中，如果 ok 前面没有返回值，或者前面的返回值为 PAM_SUCCESS，那么这个标记了 ok 的返回值将覆盖前面的返回值。但如果前面执行过的验证中有最终将导致失败的返回值，那 ok 标记的值将不会起作用。
    - done
        - 在前面没有 bad 值被命中的情况下，done 值被命中之后将马上被返回，并退出整个栈。
    - N（一个自然数）
        - 功效和 ok 类似，并且会跳过接下来的 N 个验证步骤。如果 N = 0 则和 ok 完全相同。
    - reset
        - 清空之前生效的返回值，并且从下面的验证起重新开始。

合理使用**su、sudo**命令

su命令是一个用户切换命令，完成普通用户与超级用户之间的切换。

为了保证服务器安全，几乎所有的服务器都**禁止**了超级用户直接登录系统。而是使用普通用户登录，需要提权时使用su命令。

但如果有多个普通用户均使用su，也存在安全隐患，这将泄露root口令。

替代方法就是使用sudo命令，可以行使一定的特权，但又不需要知道root口令。**所以sudo比su命令安全，我们推荐使用sudo。**

普通用户如果想运行sudo命令，必须在/etc/sudoers中进行配置以下内容：

```
# 运行某个用户账户以特权身份访问 /bin/more /etc/shadow
用户账户名  ALL = /bin/more /etc/shadow

# 设置某个用户可以不输入密码执行特权程序（有利于自动执行某些进程）
用户账户名  ALL = NOPASSWORD : /etc/init.d/nagios restart

# 如果不限制user账户的访问范围，可以使用下面语句
用户账户名  ALL = (ALL)

# 不需要输入密码的特权设置
用户账户名  ALL =  (ALL) NOPASSWD : ALL
```

删减系统登录欢迎信息

为了防止系统欢迎信息不泄露信息给别有用心的人，通常要修改欢迎信息。

这些信息存放在4个文件中：

- /etc/issue
  - 记录了OS名称和版本号
  - 本地终端登录、本地虚拟控制台登录时会显示信息
- /etc/issue.net
  - 记录了OS名称和版本号
  - 通过ssh、telnet登录时会显示信息。
  - 默认情况下不显示，需要显示可以修改/etc/ssh/sshd_config，添加 `Banner /etc/issue.net`
- /etc/redhat-release
- /etc/motd
  - 系统公告信息，用户登录时显示。
  - 管理员可用于发布一些软硬件更新信息、警告信息。黑客常用此发表震慑信息。

issue 内各代码意义：

- \d 本地端时间日期；
- \l 显示第几个终端机接口；
- \m 显示硬件等级 (i386/i486/i586/i686...)；
- \n 显示主机网络名称；
- \o 显示 domain name；
- \r 操作系统版本 (相当于 uname -r)
- \t 显示本地端时间；
- \s 操作系统名称；
- \v 操作系统版本。

注：ubuntu与别的linux不同，直接修改/etc/motd文件重登录后无效。因为这里/etc/motd是一个符号链接，指向/var/run/motd，应该是一个启动后在生成的文件。在版本10.04中，找到生成的脚本在目录/etc/update-motd.d/中。修改后用：sudo run-parts /etc/update-motd.d 去执行就会立即见到效果，而不用反复注销登录。

禁止 Control-Alt-Delete 键盘关闭命令

Linux默认配置下，同时按下 Control-Alt-Delete 系统将自动重启，这个策略很不安全。

在 Centos 5.x以下禁止的方法是：

```
vi /etc/inittab

# 找到如下内容，在前面加上#，注释掉。
ca::ctrlaltdel:/sbin/shutdown -t3 -r now

# 然后执行
telinit -q
```

在 Centos 6 以下禁止的方法是：修改/etc/init/control-alt-delete.conf文件，找到以下内容，然后注释掉：

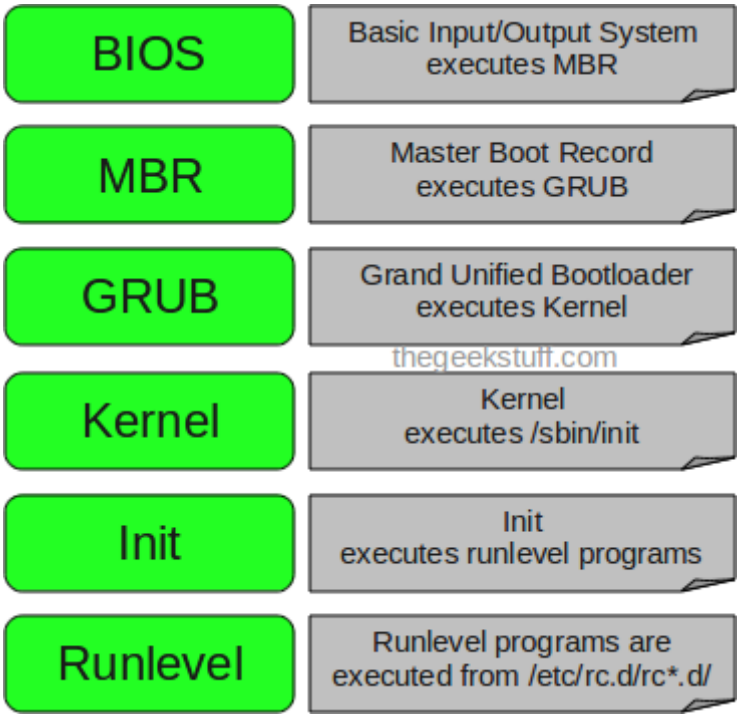
```
exec /sbin/shutdown -r now "Control-Alt-Delete pressed"
```

动手实践

参考：实验9Linux账户和登录安全运维实验

Linux 系统启动过程

x86下Linux启动过程



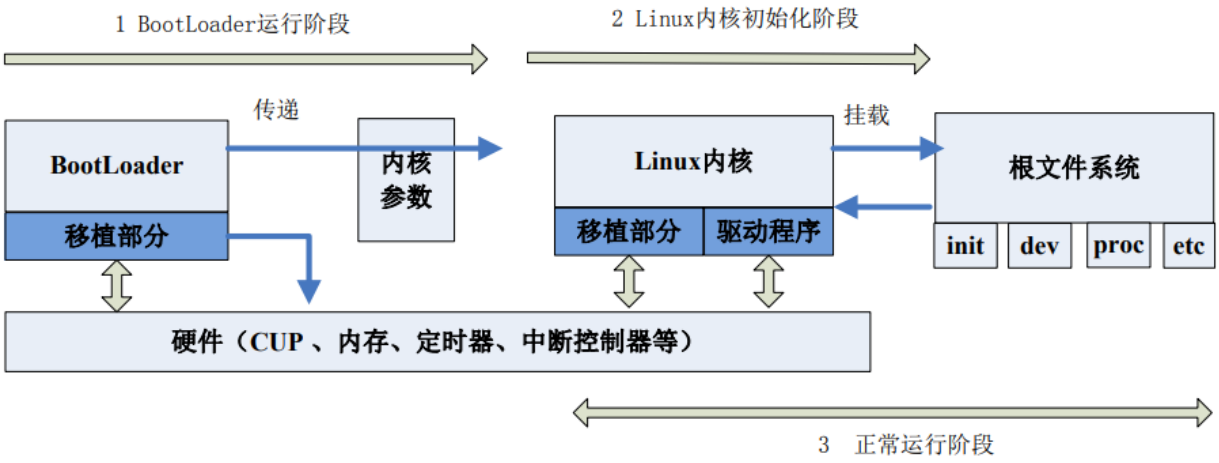
Linux系统的启动过程，可分为6个阶段：

- BIOS运行阶段
  - BIOS代表主板基本输入/输出系统
  - 执行一些系统完整性检查，搜索，加载并执行引导加载程序。
- MBR运行阶段
  - MBR代表主引导记录。
  - 它位于可引导磁盘的第一个扇区中。通常是/dev/hda或/dev/sda
  - 它包含有关GRUB（或旧系统中的LILO）的信息。
- GRUB运行阶段
  - GRUB代表Grand Unified Bootloader。
  - 如果您的系统上安装了多个内核映像，则可以选择要执行的一个。
  - Grub的配置文件是/boot/grub/grub.conf（/etc/grub.conf是此文件的链接）。
- Kernel运行阶段
  - 按照grub.conf中“ root =”中的指定挂载根文件系统；
  - 内核执行/ sbin / init程序；
  - 由于init是Linux内核执行的第一个程序，因此它的进程ID（PID）为1。
  - 内核将initrd用作临时根文件系统，直到启动内核并装入实际的根文件系统为止。
- Init运行阶段
  - 查看/ etc / inittab文件以确定Linux的运行级别。
  - 以下是可用的运行级别：0 –停止；1 –单用户模式；2 –多用户，无NFS；3 –完全多用户模式；4 –未使用；5 – X11；6 –重新启动

    - 初始化从/ etc / inittab识别默认的初始化级别，并使用它来加载所有适当的程序。默认运行级别设置为3或5
- Runlevel运行阶段
    - 根据您的默认初始化级别设置，系统将从以下目录之一执行程序。
    - 运行级别0 – /etc/rc.d/rc0.d/
    - 运行级别1 – /etc/rc.d/rc1.d/
    - 运行级别2 – /etc/rc.d/rc2.d/
    - 运行级别3 – /etc/rc.d/rc3.d/
    - 运行级别4 – /etc/rc.d/rc4.d/
    - 运行级别5 – /etc/rc.d/rc5.d/
    - 运行级别6 – /etc/rc.d/rc6.d/
    - 在/etc/rc.d/rc*.d/目录下，您会看到以S和K开头的程序。
    - 在启动过程中使用以S开头的程序。S用于启动。
    - 在关机期间使用以K开头的程序。K为杀戮。

嵌入式Linux启动过程

- BootLoader运行阶段
- Linux初始化阶段
- 系统的正常运行阶段



Linux Bootloader

引导加载程序（Bootloader）是系统加电后运行的第一段软件代码。

Bootloader是与系统硬件环境高度相关的初始化软件，它担负着初始化硬件和引导操作系统的双重责任。

Bootloader分为：

- Boot：启动、上电
- Loader：加载系统

了解Bootloader，对了解系统底层运行机制、优化和快速启动的研究都有重要的意义。

Bootloader特点

- 功能上
  - 初始化硬件设备
  - 建立内存空间的映射图
  - 调整系统的软硬件环境，以便操作系统内核启动
- 平台上
  - 不通用
  - 依赖于硬件CPU
  - 依赖于主板board
  - 不同的CPU有不同的BootLoader

BootLoader支持的体系结构

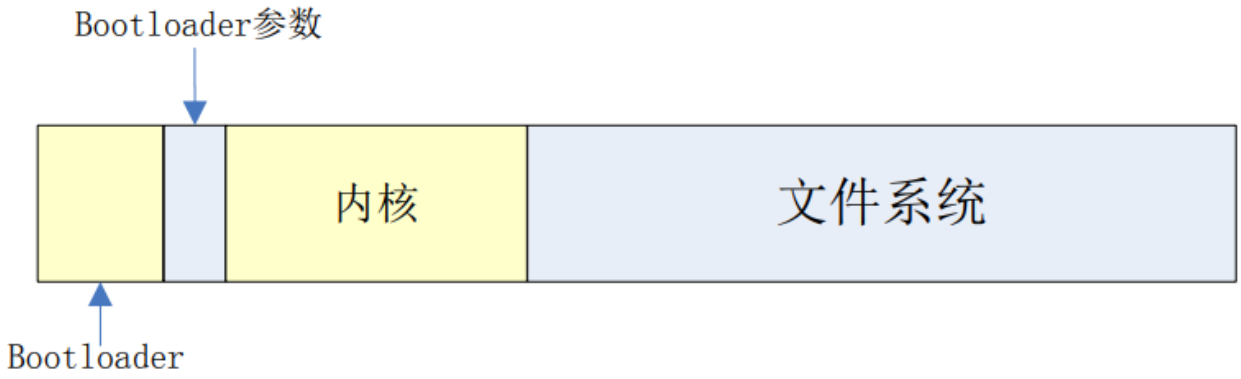
名称	功能说明	体系结构
LILO	LINUX的磁盘引导加载程序	x86
GRUB	LILO的GRU版	x86
Loadlin	从Dos引导Linux系统	x86
U-BOOT	通用引导加载程序	x86,ARM,PowerPC,MIPS等
RedBoot	以eCos为基础的引导程序	x86，ARM,PowerPC,MIPS,M68K
VIVI	为S3C24XX处理器引导Linux	ARM
ROLO	可替代BIOS,能从ROM引导Linux	x86
Etherboot	从以太网卡启动Linux系统的固件	x86
LinuxBIOS	以Linux为基础的BIOS替代品	x86
BLOB	来自LART计划的引导程序	ARM

BootLoader的安装

系统加电或复位后，所有的CPU通常都从某个由CPU制造商预先安排的地址上取指令。比如，基于ARM7TDMI的CPU在复位时通常都从地址 0x00000000取它的第一条指令。

基于CPU 构建的系统通常都有某种类型的固态存储设备被映射到这个预先安排的地址上。比如：ROM、EEPROM 或 FLASH 等。

因此在系统加电后，CPU将首先执行Bootloader 程序。



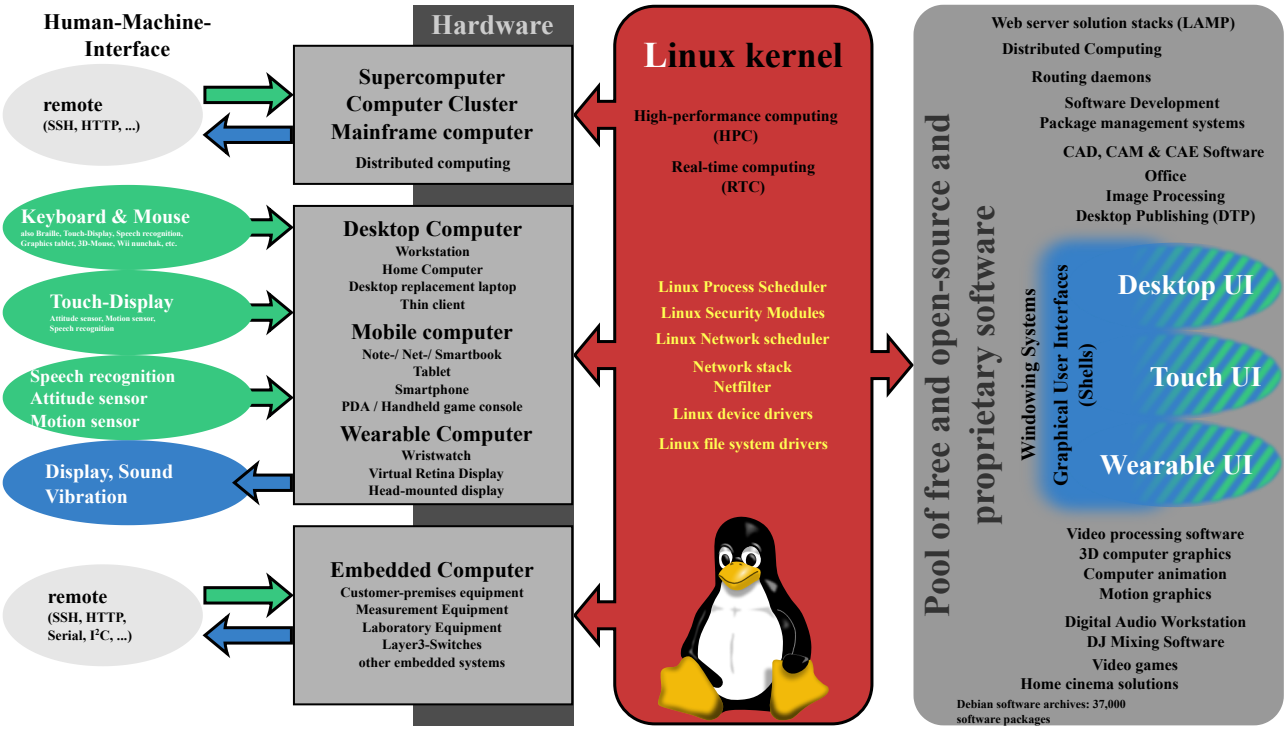
BootLoader的启动过程

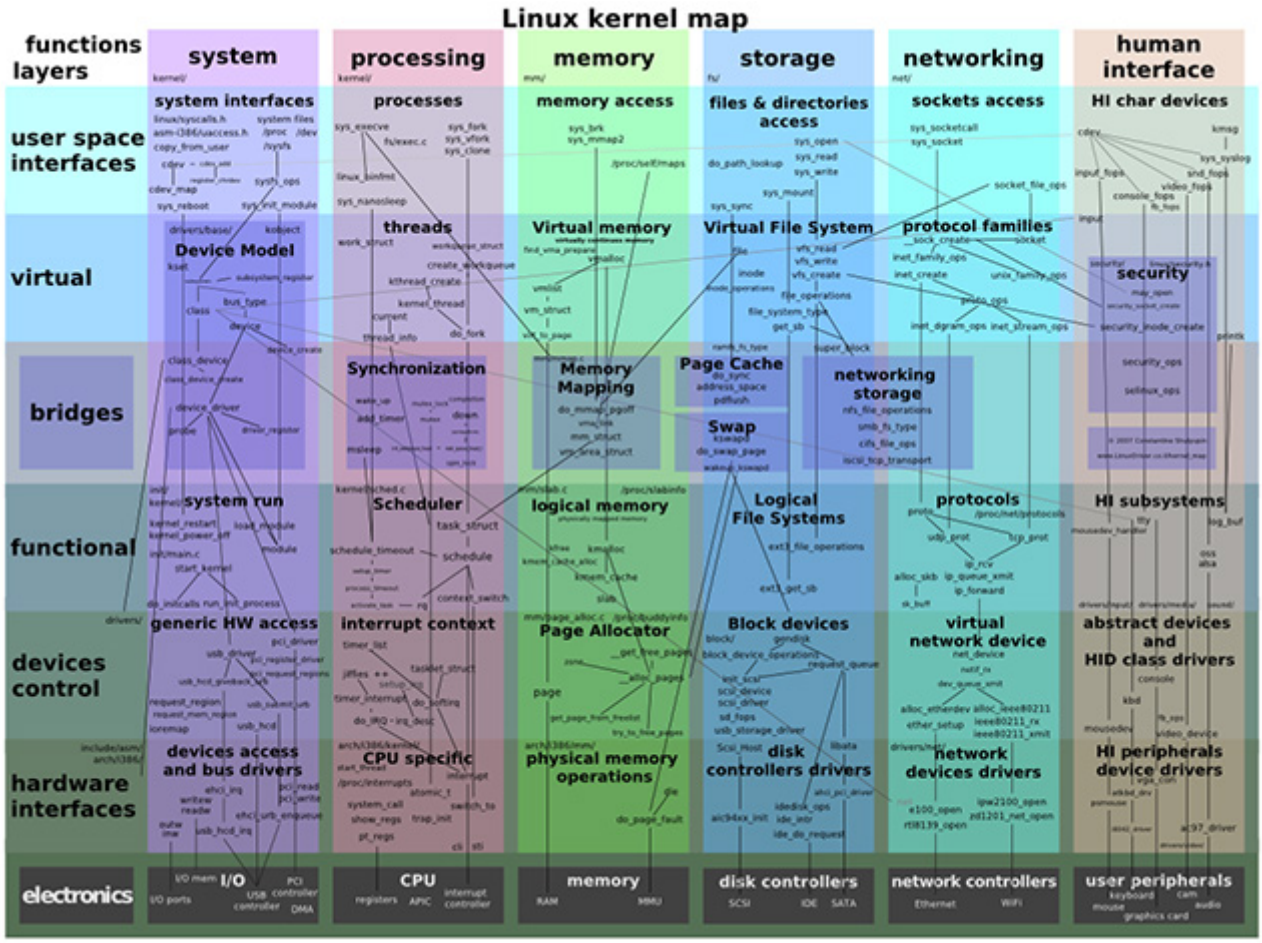
Boot Loader的启动过程可以是：

- 单阶段（Single Stage）
  - 一些只需完成很简单功能的boot loader可能是单阶段的
- 多阶段（Multi-Stage）
  - 通常多阶段的 Boot Loader 能提供更为复杂的功能，以及更好的可移植性。从固态存储设备上启动的 Boot Loader 大多都是 2 阶段的启动过程，也即启动过程可以分为 stage1 和 stage2 两部分。

Linux 内核

Linux是一个用C语言写成、符合POSIX标准的类Unix操作系统, 是最受欢迎的免费操作系统内核。“内核”指的是一个提供硬件抽象层、磁盘及文件系统控制、多任务等功能的系统软件。

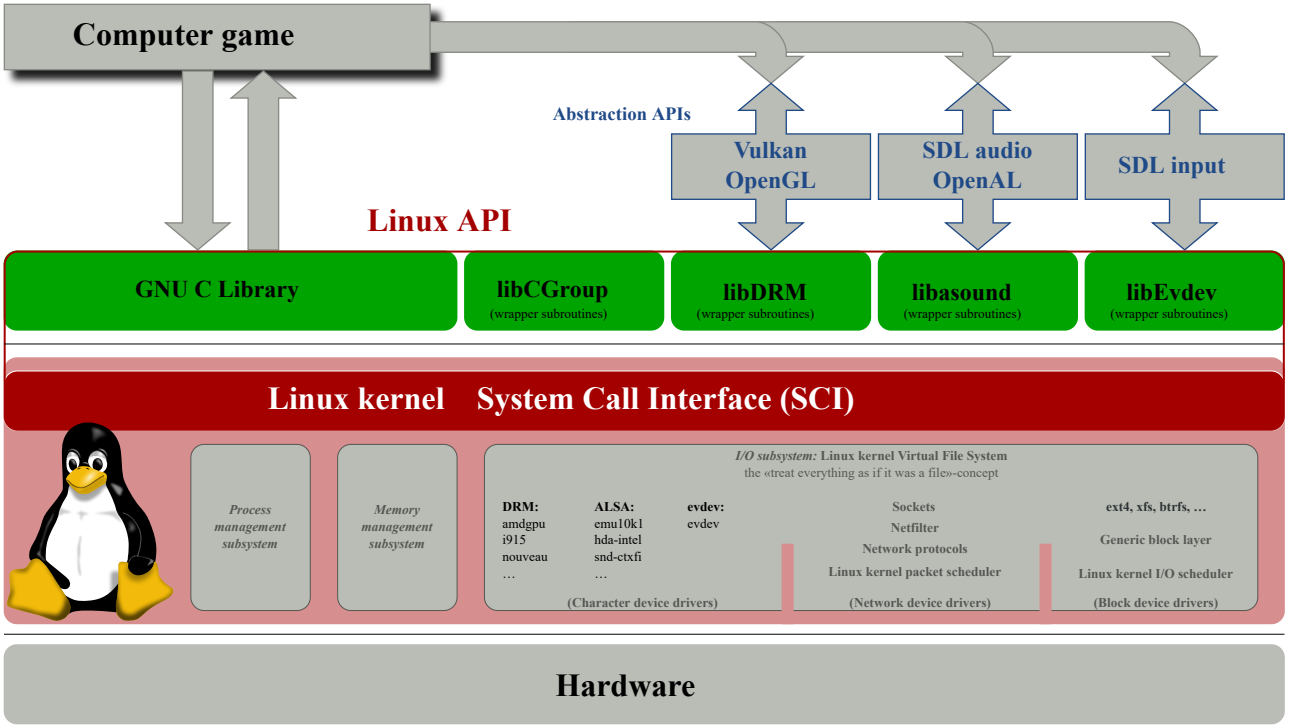




一个套内核源文件包括以下内容:

```
leo@ubuntu:~/Downloads/linux-5.2.11$ ls
arch      crypto          init      lib          net       sound
block     Documentation   ipc       LICENSES     README    tools
certs     drivers         Kbuild    MAINTAINERS  samples   usr
COPYING   fs              Kconfig   Makefile     scripts   virt
CREDITS   include         kernel    mm           security
```

linux 内核API



动手实践

参考：实验8Linux内核配置编译安装实验