LO03 Notes II

Partie HAJJAM

Table des matières

1 引言

1.1 Unix

Unix 于 1969 年诞生于 **贝尔实验室 (Bell Laboratories, AT&T)**, 由 **Ken Thompson** 和 **Dennis Ritchie** 两人推动开发。

— 1973 年，**Unix** 的新版本被用 **C 语言**重新编写，从而成为一个可移植的版本。
— 1974 年, 第 4 版的 **Unix** 被「捐赠」给了 **加利福尼亚大学伯克利分校 (University of Berkeley, California)**。
— 这标志着 **Unix** 的两个版本之间开始出现分歧：**AT&T** 和 **BSD (Berkeley Software Distribution)**。
— 从 1977 年到 1979 年，**Ken Thompson** 和 **Dennis Ritchie** 对 **Unix** 进行了重写，使其真正实现了可移植性。
— 1980 年，**AT&T** 发布了首批 **Unix System V** 发行版许可证。
— 1984 年，**麻省理工学院 (MIT)** 创建了 **X Window** 标准：一个独立于 **Unix** 的多窗口图形系统，但它对该系统的成功起到了很大作用。

1.2 Unix 到 Linux 的演变

自己看 PPT

1.3 主要的 Unix 系统 (Les principaux Unix propriétaires)

— **IBM** : AIX®
— **HP** : HP-UX®, Tru64 UNIX®
— **SCO** : UnixWare®
— **SGI** : IRIX®
— **SUN** : Solaris®

1.4 主要的开源 Unix 系统 (Les principaux Unix Libres)

— **OpenBSD**
— **FreeBSD**
— **NetBSD**
— **Mac-OS X**
— **GNU/Linux**

1.5 The Open Group

— 持有注册商标 **UNIX®**。
— 发布 **Single UNIX Specification** 标准，整合了以下标准：
  — **X/Open Company**'s XPG4
  — **IEEE**'s POSIX Standards
  — ISO C

1.6 从 Unix 到自由软件 (Unix vers Logiciel Libre)

Richard Stallman（麻省理工学院研究员，`gcc` 和 `Emacs` 的作者）明确提出了 **自由软件 (logiciel libre)** 的概念：

> « ... un savoir scientifique doit être partagé en le distribuant, ... les codes source doivent être libres d'accès ... »

— 1984 年，启动 **GNU 项目**，目标：重新创建一个完整的类似 Unix 的操作系统，仅由自由软件组成。
— 1985 年，创建 **FSF (Free Software Foundation)**，以管理 **GNU 项目**。
— 注释：在黑客文化中，**"Free"** 意味着"自由"，而不一定是"免费"或"非商业化"。

1.7 从 Unix 到 GNU (Unix vers GNU)

GNU 项目 (Le projet GNU)：
— 基本原则：自由访问源码可以加速计算机科学的进步，因为创新依赖于源码的共享。
— **GNU** 的自由由四个基本原则定义：
  — 自由运行程序，适用于任何用途。
  — 自由研究程序的运行方式，并根据需要修改程序。
  — 自由分发程序副本。
  — 自由改进程序并公开改进成果，以造福社区。

1.8 从 Unix 到 GPL (Unix vers GPL)

GPL 许可 (General Public Licence)：
— 允许用户自由复制和分发被保护的软件，只要不限制其他人同样的权利。
— 要求任何在此许可下的衍生作品也必须受其保护。
— 当 **GPL** 提到自由软件时，指的是"自由 (liberté)"而非"免费 (gratuit)"。
— 注：**"Free"** 在英文中同时具有"自由"和"免费"的意思。

Copyleft 概念 (Le copyleft de la licence GPL):
— 1984 年由 **Stallman** 创建。
— 确保所有用户享有 4 项基本自由。
— 避免将 **GNU** 软件置于公共领域（无保护状态）。
— 规定重新分发软件时，必须保留复制和修改的自由。
— 鼓励开发者（企业、大学）加入并改进自由软件。
— **Copyleft** 的软件先声明版权，然后附加不可分割的分发和自由条款。

1.9 从 Unix 到开源 (Unix vers Open Source)

开源倡议 (Open Source Initiative, OSI):
— 1997 年，由 **Eric Raymond**、**Tim O'Reilly** 和 **Larry Augustin** 等人推动，提出“开源 (Open Source)”以标记源码开放的软件。
— **Open Source** 比 **GPL** 限制更少。
— **Open Source Definition** 是 **Debian 社会契约 (Debian Social Contract)** 的直接继承。
— **Open Source** 允许在专有代码与开源代码之间进行更多的混合。

1.10 从 Unix 到 GNU/Linux (Unix vers GNU/Linux)

GNU/Linux:
— 它是一个完全的 **Unix**。
— 它是一个在 **GPL (General Public License)** 许可下分发的自由软件。
— **Linux** 内核的源代码可以在 http://www.kernel.org/ 上获得。
— 组成部分:
  — **Linux** 内核（Unix 系统的克隆，由 Linus Torvalds 和一个通过互联网协作的开发团队编写）。
  — 来自 **GNU 项目**的自由软件组件（如 gcc, ...），符合 **Single UNIX Specification** 标准。
— 可用于所有平台（PC、工作站、集群、主机等）。
— 大多数专有 **Unix** 解决方案的供应商（IBM、HP、SiliconGraphics 等）都整合了 **GNU/Linux**。
— **Linux** 的管理模式模仿了 **UNIX System V (AT&T)**。

1.11 Linux 的发行版 (Linux les distributions)

— **RedHat** (www.redhat.fr):
  — 成立于 1994 年的美国公司。

  - 因引入 **Red Hat Package Manager (RPM)** 软件包管理系统而闻名。
- **Fedora** (fedora.redhat.com):
  - **RedHat** 的免费版本。
  - 使用 **RPM** 包管理系统。
- **Mandriva** (www.mandriva.com):
  - 著名的法国发行版。
  - 在教育领域及中小企业中非常流行。
  - 使用 **RPM** 包管理系统。
- **SUSE** (www.novell.com/linux/suse):
  - 1993 年在德国纽伦堡成立。
  - 2003 年被美国软件公司 **Novell** 收购。
  - 使用 **RPM** 包管理系统。
- **Debian** (www.debian.org):
  - 来源于社区的努力，而非企业。
  - 精心设计且非常灵活。
  - 严谨的安装与管理（不推荐初学者使用）。
  - 使用 **Debian** 软件包管理系统。
- **Ubuntu** (www.ubuntu-fr.org):
  - 基于 **Debian** 的流行发行版，始于 2004 年。
  - 名称来源于非洲班图语的“ubuntu”，意为“人性关怀”或“我因你们而存在”。
- **TurboLinux** (www.turbolinux.com):
  - 主要在亚洲（尤其是中国）流行的 Linux 发行版。

2 Unix 概述 (Généralités Unix)

Unix 是一个多用户“分时 (temps partagé)”操作系统，也是一个多任务系统。

2.1 系统特点 (Caractéristiques du système)

- 文件系统是一个分层的树状结构。
- 输入输出操作是通用的，外部设备从用户的角度来看如同文件。
- 系统 99% 是用 **C 语言**编写的，这使得应用程序调用内核变得简单。

2.2 系统启动 (Démarrage du système)

当机器启动时，会经历以下几个步骤:

- 给机器及其外设上电。

— 引导系统（加载 Linux 内核）。
— 挂载磁盘。
— 检查文件系统的完整性（**fsck**）。
— 切换到多用户模式。
— 启动服务。
最后进入登录阶段:

« Login : »

2.3 文本模式示例 (Exemple en mode texte)

```
login : xstra
passwd : § l'utilisateur xstra entre son mot de passe
**********************************************
** Le systeme est arrete le Lundi 19/05/03 **
** pour maintenance **
**********************************************
Lundi 25 Decembre 2017
Vous avez dix messages
Derniere connexion : Vendredi 22 Decembre 2017 a 18 h 04
xstra>
```

文本模式的登录示例

在此示例中，用户的名称是 **xstra**，登录提示符是 `xstra>`。

2.4 管理员的任务 (Missions de l'administrateur)

2.4.1 系统视角下的职责 (Point de vue système)

— 从用户的角度来看，不同版本的 **Unix** 非常相似。
— 从管理的角度来看，每个 **Unix** 系统都有自己的特点（例如与硬件相关的命令，或者厂商特有的扩展）。
操作系统的任务包括:
— 提供硬件资源：例如磁盘空间、中央处理器的执行时间、内存空间等；
— 在用户之间公平分配这些资源，以实现多用户系统的目标。

2.4.2 Unix 管理员的核心任务 (Tâches principales de l'administrateur Unix)

1. 确保系统的正常运行:
   — 监控资源（磁盘、内存、CPU 等）;

— 计划资源的更新和演进。

2. 管理服务：
   — 管理用户；
   — 安装和配置服务及应用程序；
   — 计划更新和迁移。
3. 预防和处理事故，跟踪安全状况：
   — 安装安全更新；
   — 保障系统及应用程序的安全；
   — 管理备份；
   — 监督系统和应用程序。

2.4.3 详细任务描述 (Tâches détaillées)

— 管理用户账户（任务简单且可自动化）。
— 辅导和教育用户（回答问题，保持文档的更新）。
— 管理软件（安装、配置、更新）。
— 管理硬件（处理故障、更换、添加）。
— 确保系统及用户的安全（可靠的备份、定期检查、访问控制、资源滥用的监控）。
— 验证硬件与其用途的适配性（识别瓶颈）。
— 一线维护（诊断问题，联系厂商维护）。
— 日常管理（处理大小事务）。

2.4.4 扩展职责 (Responsabilités étendues)

Unix 管理员还需要承担以下职责：
— 外交与政策。
— 法律问题（如加密）。
— 调查（如网络侵权、非法内容）。
— 商务关系。
— 设备使用政策。

2.5 Root 账户 (Le compte root)

2.5.1 简介 (Introduction)

为了执行管理任务，**root 账户**被用于系统操作。它拥有访问系统完整性的权限。然而，由于权限管理机制相对原始，**root 账户**的使用需要特别注意以下几点：
— 记录每一项操作；
— 避免快速和未经验证的更改；

— 规划恢复机制以备后用。

这些要求突出了以下管理原则的重要性：

技能、严谨与常识 (compétence, rigueur et bon sens)

除 **root 账户**外，其他用户受到访问权限管理系统的限制。因此，难以将系统管理任务委托给普通用户。

2.5.2 Unix 支持的两种管理模式 (Deux modes d'administration dans Unix)

手动方式 (« À la main »)

— 手动编辑配置文件；
— 手动执行管理命令；
— 使用 RPM 或 DEBIAN 等包管理工具的手动操作；
— 编写和编辑命令脚本（例如：shell、perl、awk 等）。

基于软件的管理方式 (Avec des logiciels d'administration)

— 利用软件工具操作配置文件；
— 使用标准或特定的管理命令；
— 使用不可或缺的专用工具（例如针对 Unix 专有系统，如 HP 的 SAM 工具）；
— 针对 **Linux** 系统，可以使用 linuxconf、webmin、DrakConf 等工具。

3 文件系统(Le système de fichiers)

3.1 文件树状结构(Arborescence des fichiers)

了解文件和目录在文件系统中的分布规则对于以下方面是必要的：
— 确定备份(sauvegarde) 计划；
— 管理安全性(sécurité)；
— 在解决问题时高效行动；
— 安装非软件包(paquetage) 形式提供的软件；
— 等等。

根目录是 /。

UNIX 系统定义了不同类型的文件：
— 物理文件(fichiers physiques)；
— 目录(répertoires)；
— 链接(links)，包括符号链接(symboliques) 和硬链接(physiques)；
— 虚拟文件(fichiers virtuels)，仅存于内存中，包含系统信息；
— 设备文件(fichiers de périphériques)。

FHS (Filesystem Hierarchy Standard) 规范了 Unix 系统中文件系统的组织方式: http://www.pathname.com/fhs/。

3.2 目录结构(Arborescence des fichiers)

— `/bin` 主要命令(commandes);
— `/dev` 设备文件;
— `/etc` 配置文件(fichiers de configuration) 和启动脚本(scripts de boot);
— `/sbin` 主要系统命令(commandes système);
— `/home` 用户目录(répertoires des utilisateurs) 和项目(projets);
— `/mnt` 临时挂载点(points de montage temporaires), 如 CD-ROM、软盘(floppy)等;
— `/proc` 特殊进程(processus spéciaux) - 内核状态(état du noyau);
— `/tmp` 临时文件(fichiers temporaires);
— `/usr/bin` 非必要用户命令(commandes utilisateurs non essentielles);
— `/usr/include` 供编译器使用的头文件(fichiers d'en-têtes pour compilateurs);
— `/usr/lib` 静态和共享库(librairies statiques et partagées);
— `/usr/local` 用于本地软件(software locaux), 类似于 `/usr`;
— `/usr/sbin` 非必要系统命令(commandes système non essentielles);
— `/usr/share` 可在多个系统间共享的文件(fichiers pouvant être partagés entre systèmes);
— `/usr/share/man` 手册页(pages du manuel), 也可能为 `/usr/man`;
— `/var` 机器特定的文件(fichiers spécifiques à la machine), 包括系统(système)和应用程序(applications);
— `/var/spool` 存储(mail, imprimantes, jobs périodiques) 邮件、打印任务(imprimantes)、定期任务(jobs périodiques)。

3.3 示例(Exemples)

访问 `/home` 目录

— 绝对路径(chemin absolu) 是 `/home`。
— 使用命令 `pwd` 显示当前目录(répertoire courant)。

切换目录

— 进入 `/etc` 并列出其内容:

```
1    cd /etc
```

— 返回个人目录(répertoire personnel)（不使用绝对路径):

```
cd ~ 或者 cd
```

创建和管理文件

— 在个人目录中创建 `test_dir`:

```
mkdir ~/test_dir
```

— 在该目录中创建三个文件:

```
cd ~/test_dir
touch file1.txt file2.txt file3.txt
```

— 重命名 `file2.txt` 为 `renamed_file.txt`:

```
mv file2.txt renamed_file.txt
```

— 删除 `file3.txt` 和 `test_dir`:

```
rm file3.txt
cd ..
rmdir test_dir
```

3.4 权限管理(Gestion des permissions)

文件权限操作

— 创建一个文件 `example.txt`:

```
touch ~/example.txt
```

— 查看文件权限:

```
ls -l ~/example.txt
```

— 示例输出:

```
-rw-r--r--
```

修改权限

— 设置权限:

— 用户(propriétaire) 可读、可写、可执行;

— 组(groupe) 仅可读;

— 其他人(autres) 无权限。

```
chmod 740 ~/example.txt
```

— 确认权限变更：

```
ls -l ~/example.txt
```

— 示例输出：

```
-rwxr-----
```

— 以另一个用户访问文件，验证权限：

```
Permission denied
```

4 Unix 用户账户与用户组管理(Les comptes utilisateurs et groupes)

4.1 Unix 用户(Utilisateurs Unix)

Unix 用户的主要属性包括：

— 登录名(login)；

— 密码(mot de passe)；

— 唯一的用户标识符(UID)；

— 主用户组(groupe primaire, GID)；

— 备注信息(gecos)；

— 个人目录(répertoire principal, home directory)；

— 默认 Shell(interpréteur de commandes, shell)。

系统通过 UID 标识用户。超级用户 root 的 UID 为 0，这使其拥有对系统的完全访问权限。这些信息存储在文本文件(fichier texte) `/etc/passwd` 中，每个字段由“:”分隔。例如：

```
utseus:x:500:500:cours lo03:/home/utseus:/bin/bash
```

4.2 文件/etc/passwd 的格式(Format /etc/passwd)

4.2.1 一些建议

— 应按 UID 的数值顺序对`/etc/passwd` 文件进行排序：

```
# sort -t : +2n -3 /etc/passwd
```

— 不应存在无密码的账户。

4.2.2 字段(Field) 解析

字段 1：登录名(login) 即计算机识别用户的名称，通常限制为 8 个字符，但 Linux 系统允许更长的名称。建议：避免使用大写字母和带重音的字符。

字段 2：密码(mot de passe) 密码以不可逆加密形式存储。用户应使用强密码，以防止密码破解软件(software de crack)。

字段 3：UID 用户的唯一标识符，在 Linux 上为 `uid_t`（无符号整型）。

字段 4：GID 用户所属主组的唯一标识符，在 Linux 上为 `gid_t`（无符号整型）。可以使用命令：

```
% ls -ln
-rw-r--r-- 1 4332 1000 5544 Sep 15 22:47 cours.dvi
-rw-r--r-- 1 4332 1000 940 Sep 15 22:27 cours.tex
```

其中 UID 和 GID 分别为 4332 和 1000。

字段 5：gecos 存储用户的真实身份信息，在 BSD 系统上可包含电话号码、办公室号码等。可用命令 `chfn`（change finger）修改此字段。

字段 6：home directory 用户的默认个人目录(répertoire principal)。

字段 7：shell 用户的默认 shell(interpréteur de commandes)。如果该字段为空，则默认使用`/bin/sh`。可使用命令 `chsh` 修改默认 shell。

4.3 创建新账户(Création d'un nouveau compte)

创建账户需执行以下步骤：

1. 选择 UID 和 GID；
2. 选择 home directory；
3. 选择符合本地策略的登录名；
4. 选择默认 shell；
5. 将用户信息添加至`/etc/passwd`（或 NIS 数据库）；
6. 在`/etc/group` 文件中添加该用户；
7. 创建 home directory；
8. 复制默认环境配置文件(.profile, .cshrc, .xsession 等)；
9. 使用 `chown` 和 `chgrp` 为 home directory 设置正确的权限；
10. 初始化密码(mot de passe)。

4.4 超级用户 root(Le compte root)

root 账户的特殊性由于 root 的 UID 为 0，它具有最高权限，需遵守以下规则：

— PATH 变量中不应包含 “.”（避免本地命令覆盖系统命令）；
— 默认 umask 应为 022，以确保某些文件可被普通用户访问；
— 避免将 root 的 home directory 设为/，以防止污染根目录。

4.5 虚拟账户(Les comptes fictifs)

虚拟账户主要用于文件所有权管理。例如：

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:
daemon:x:2:2:daemon:/sbin:
```

其中：
— bin 账户是大多数可执行文件的所有者；
— daemon 账户用于运行系统守护进程(daemon)。

4.6 Unix 用户组(Groupes Unix)

/etc/group 文件用于定义用户组(groupe)。其格式如下：

```
nom_du_groupe:mot_de_passe:GID:liste_utilisateurs
```

用户可使用以下命令查询所属用户组：

```
# id
uid=501(utseus) gid=501(utseus) groupes=501(utseus),100(users)
   ,101(cvs),200(compta)

# groups
utseus users cvs compta
```

4.7 密码管理(Mots de passe cachés)

最初，密码存储在/etc/passwd，但因该文件需全体用户可读，为增强安全性，密码已转移至/etc/shadow 文件。/etc/shadow 使用 MD5 算法加密，并添加密码有效期管理字段。例如：

```
utseus:c8NuTn3ScFJfA:12460:0:99999:7:::
```

其中：
— 第 4 字段：最短有效期(0 天)；
— 第 5 字段：最长有效期(99999 天)。

4.8 用户和组管理(Gestion des utilisateurs et groupes)

4.8.1 基本操作

— 创建用户 testuser:

```
sudo useradd -m testuser
```

— 添加用户描述:

```
sudo usermod -c "Utilisateur de test" testuser
```

— 更改默认 shell 为/bin/zsh:

```
sudo usermod -s /bin/zsh testuser
```

— 删除用户及其 home directory:

```
sudo userdel -r testuser
```

4.8.2 账户安全

— 更改密码:

```
sudo passwd testuser
```

— 锁定账户:

```
sudo passwd -l testuser
```

— 解锁账户:

```
sudo passwd -u testuser
```

注意: 如果两个用户拥有相同 UID，他们将共享权限，并被系统视为同一用户，可能导致权限混乱。

5 任务调度(Ordonnancement des tâches)

5.1 任务调度器 cron(Le planificateur de tâches cron)

任务调度器 cron 允许每个用户使用 `crontab` 命令配置程序或脚本的定时执行，可以是某个指定的日期或周期性执行。超级用户 root 也可以使用此功能。然而，对于涉及系统管理的定期任务，我们通常更倾向于使用以下目录:

— `/etc/cron.hourly` - 每小时执行;

— `/etc/cron.daily` - 每天执行;

— `/etc/cron.weekly` - 每周执行;

— /etc/cron.monthly - 每月执行;

— /etc/cron.d - 允许更精确的任务调度。

前四个目录的使用方式很简单:只需在相应目录下复制一个可执行文件(fichier exécutable) 或创建一个符号链接(lien),即可以定期执行该文件。

5.2 特殊的/etc/cron.d 目录

/etc/cron.d 目录的使用较为特殊,该目录中的文件是文本文件(fichier texte),其格式与 crontab 命令所使用的格式几乎相同。例如:

```
MAILTO=root
# mn hr day month dow user command
0 4 * * * postgres /usr/lib/postgresql/bin/do.maintenance -a
```

此条目表示:

— 每天的 4 :00 执行;

— 每月的所有天;

— 每周的所有天(如果要排除周末,则用 1-5)。

文件/etc/crontab 控制/etc/cron.* 目录(不包括/etc/cron.d)中的任务执行时间。用户创建的计划任务存储在/var/spool/cron 目录下,每个用户对应一个文件。

5.3 用户任务管理(Gestion des utilisateurs et groupes)

示例 1:解释 crontab 中的任务

```
30 2 * * 1-5 /usr/local/bin/backup.sh >> /var/log/backup.log 2>&1
```

该任务:

— 在每天凌晨 2 :30 执行;

— 每天执行,无论日期和月份;

— 仅在周一至周五执行(1-5 表示工作日);

— 执行/usr/local/bin/backup.sh 脚本;

— 将标准输出和错误输出重定向到/var/log/backup.log。

示例 2:计划多个任务

— 每天 7 :15 执行/usr/bin/monitor.sh:

```
    15 7 * * * /usr/bin/monitor.sh
```

— 每周一和周四 14 :45 执行:

```
    45 14 * * 1,4 /usr/bin/monitor.sh
```

示例 3：解决 crontab 任务不执行的问题如果 `30 8 * * * /home/user/script.sh` 未执行，可能的解决方案：

— 检查 cron 是否运行：

```
systemctl status cron
```

— 确保脚本具有执行权限：

```
chmod +x /home/user/script.sh
```

— 手动执行测试：

```
/home/user/script.sh
```

— 查看 cron 日志（某些系统）：

```
grep CRON /var/log/syslog
```

— 脚本可能需要环境变量，在任务的开头添加：

```
SHELL=/bin/bash
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/
sbin:/usr/bin
30 8 * * * /home/user/script.sh
```

示例 4：检查 crontab 语法

```
crontab -l
```

5.4 cron 与 at 的区别

— **cron**：用于计划周期性任务(tâches récurrentes)，按照设定的时间表执行；

— **at**：用于计划一次性任务(tâche unique)，在指定时间执行一次。

示例 5：使用 `at` 命令

```
echo "/home/user/script.sh" | at 10:00 tomorrow
```

该命令将在次日 10 :00 执行`/home/user/script.sh`。

6 事件日志服务(Services de journaux d'évènements)

6.1 系统事件：syslog (Les évènements systèmes : syslog)

UNIX 系统用于收集由系统或应用程序生成的事件的机制非常高效。Unix 使用一种统一的日志系统 `syslog`，它开发于 20 世纪 80 年代。

系统事件的记录由两个程序（守护进程）管理：`klogd` 和 `syslogd`。

— klogd 简单地监听来自内核的消息；
— 主要的工作由 syslogd 完成。当 syslogd 启动时，它读取文件 /etc/syslog.conf，并根据配置决定每条消息应该记录到哪个文件中。

可以根据消息的来源和/或其严重级别将这些消息重定向到文件、控制台等，甚至可以将其转发到另一台机器的 Syslog。

默认情况下，几乎所有消息都会被记录到位于目录 /var/log 中的文件中。

6.2 配置 syslog 的规则

文件 /etc/syslog.conf 中的一行配置规则格式如下：

```
facility.priority[,facility.priority,...] action
```

消息来源被称为“功能”(facilités)。以下是功能列表：
— auth：与安全性或身份验证相关的消息；
— authpriv：与 auth 类似，但可能包含私人信息；
— cron：由任务调度器（如 cron 或 at）生成的消息；
— daemon：系统守护进程生成的消息；
— kern：内核生成的消息；
— lpr：打印子系统生成的消息；
— mail：邮件子系统生成的消息；
— mark：由 Syslog 生成的时间标记；
— news：USENET 管理子系统生成的消息；
— syslog：内部 Syslog 消息；
— user：用户生成的通用消息；
— uucp：UUCP 子系统生成的消息；
— local0 到 local7：管理员根据需要定义的消息类别。

6.3 优先级级别和示例配置(Les niveaux de priorités et exemple syslog.conf)

6.3.1 优先级级别(Les niveaux de priorités)

优先级按照严重性排序如下：
— debug：用于调试(debugging) 的消息；
— info：信息(informations)；
— notice：正常但重要的(normal mais significatif) 消息；
— warning：警告(avertissement)；
— err：错误条件(conditions d’erreur)；

— `crit`：严重条件(conditions critiques)；

— `alert`：需要立即处理的操作(action à gérer immédiatement)；

— `emerg`：系统不可用(système inutilisable)。

一个优先级不仅表示它本身，还代表所有比它更高优先级的等级，除非它前面加了一个符号“=”，在这种情况下它仅表示自身。优先级“none”表示忽略相关消息，而“*”表示所有优先级。在优先级前使用“!”表示忽略该优先级。

6.3.2 syslog 配置示例(Exemple syslog.conf)

以下是/etc/syslog.conf 配置文件的示例：

```
# 所有来自内核(kern)的消息都会发送到控制台(console)。
kern.*                          /dev/console

# 不属于debug级别，且不涉及mail、authpriv和cron功能的消息
# 会被收集到一个文件中。
*.info;mail.none;authpriv.none;cron.none    /var/log/messages

# 仅包含debug级别的消息（authpriv功能除外）的消息
# 会被记录到一个特定文件中。
*.=debug;authpriv.none          /var/log/debug

# 与安全性相关的信息。
authpriv.*                      /var/log/secure

# 邮件子系统的消息。
mail.info                       /var/log/maillog

# 计划任务的消息。
cron.*                          /var/log/cron

# 所有连接的用户接收紧急级别(emerg)的消息。
*.emerg                         *

# 所有消息都会被发送到另一台机器的Syslog服务。
*.*                             @supervision
```

6.4 日志轮转工具 logrotate (Les évènements systèmes : logrotate)

`logrotate` 工具允许对由 `Syslog` 或其他应用程序生成的日志进行自动且定期的轮转（通过任务调度器 `cron`）。

默认的配置文件是 `/etc/logrotate.conf`。每个子系统可以在目录 `/etc/logrotate.d` 中添加自己的配置文件以定义其日志轮转规则。

以下示例展示了 `logrotate` 在 3 次轮转基础上工作的过程：

— **第 1 周 (semaine 1)**：日志文件 `messages` 为当前活动日志文件；

— **第 2 周 (semaine 2)**：`messages` 被重命名为 `messages.1`；

— **第 3 周 (semaine 3)**：`messages` 被重命名为 `messages.1`，而 `messages.1` 被重命名为 `messages.2`；

— **第 4 周 (semaine 4)**：同样，`messages` 和先前的文件按顺序重命名，`messages.2` 被重命名为 `messages.3`；

— **第 5 周 (semaine 5)**：最旧的日志文件 `messages.3` 被删除，新日志轮转开始。

轮转逻辑的可视化如下：

— **第 1 周 (semaine 1)** : `messages (actif)`

— **第 2 周 (semaine 2)** :

  — `messages (actif)`

  — `messages.1 (semaine 1)`

— **第 3 周 (semaine 3)** :

  — `messages (actif)`

  — `messages.1 (semaine 2)`

  — `messages.2 (semaine 1)`

— **第 4 周 (semaine 4)** :

  — `messages (actif)`

  — `messages.1 (semaine 3)`

  — `messages.2 (semaine 2)`

  — `messages.3 (semaine 1)`

— **第 5 周 (semaine 5)** :

  — `messages (actif)`

  — `messages.1 (semaine 4)`

  — `messages.2 (semaine 3)`

  — `messages.3 (semaine 2)`

  — 删除(supprimer): `messages.3 (semaine 1)`

6.5 logrotate 配置示例(Les évènements systèmes : logrotate.conf)

为了为特定的日志文件配置 logrotate,可以在目录 /etc/logrotate.d/application 中创建如下文件:

```
/usr/application/log/*.log {
    daily
    rotate 3
    size=1M
    compress
    dateext
    missingok
    notifempty
}
```

配置选项解释:

— compress: 使用 gzip 对日志进行压缩。

— create mode owner group: 指定新日志文件的权限(如 chmod) 以及所有者。

— daily, weekly, monthly: 日志轮转的时间间隔，分别为每天、每周和每月。

— size size[G|M|k]:当日志文件的大小超过指定值时触发轮转。支持比特(bit)、千比特(Kilobit)、兆比特(Megabit) 和吉比特(Gigabit) 单位。

— mail address: 将日志文件发送到指定的电子邮件地址。

— olddir directory: 将旧日志文件移动到指定的目录。

— rotate count: 保留指定数量的旧日志文件。

— dateext:将轮转日期附加到日志文件名中,格式为 YYYYMMDD。例如:access_log-20090529.gz。

— notifempty: 如果日志文件为空，则不会进行轮转。

— missingok: 如果日志文件不存在，则不会报错，也不会中断轮转过程。

6.6 logrotate 配置文件示例 (Les évènements systèmes : exemple logrotate.conf)

以下是位于 /etc/logrotate.conf 的 logrotate 配置文件示例，描述了日志轮转的各种规则:

```
# Fichier de configuration de logrotate dans /etc/logrotate.conf

# Les logs sont compressés avant rotation : compress
compress
```

```
# Fréquence de rotation par défaut : 每天 (daily)
daily
# 如果文件为空也执行轮转，便于逐日查看日志
ifempty
# 默认情况下，每天执行一次轮转。我们保留366天的日志
rotate 366
# 同时，超过366天的日志将被删除
maxage 366
# 即将过期的文件通过邮件发送
maillast
# 定义接收即将过期文件的邮箱地址
mail fhh@admin-linux.fr
# 如果日志文件不存在，不会报告错误
missingok
# 在日志归档时插入日期
dateext
# 当有多个日志文件被归档时，预旋转 (pre-rotate) 和后旋转 (post-
   rotate) 脚本只执行一次
sharedscripts
# 每次轮转时，新日志文件将使用0640权限，并属于root用户和root组
create 0640 root root
# 所有归档文件存储在 /var/log/archives 目录下
olddir /var/log/archives
# 配置 /var/log/argus/argus.log 的日志轮转
/var/log/argus/argus.log {
    olddir /var/log/archives/argus
}
# 包含 /etc/logrotate.d 中的额外日志配置文件
include /etc/logrotate.d
```

6.6.1 示例解析 (EXEMPLE)

— `*.info;mail.none;authpriv.none /var/log/messages` 所有信息级别(`info`)的日志,除了`mail`和`authpriv`服务的日志,被记录到`/var/log/messages`。

— `authpriv.* /var/log/secure` 所有与私有认证(`authpriv`) 相关的消息被记录到 `/var/log/secure`。

— `cron.* /var/log/cron` 所有计划任务(`cron`) 生成的消息被记录到 `/var/log/cron`。

问题 1：`/var/log/secure` 中的消息类型是什么？答：与安全性和身份认证相关的消息，如 SSH 登录或密码失败等。

问题 2：分析以下配置：

```
/var/log/apache2/*.log {
    weekly
    rotate 4
    compress
    missingok
    notifempty
}
```

— 日志每周轮转一次 (`weekly`)；
— 最多保留 4 个日志文件 (`rotate 4`)；
— 压缩轮转日志 (`compress`)；
— 忽略缺失的日志文件，不报告错误 (`missingok`)；
— 如果日志文件为空，则不进行轮转 (`notifempty`)。

问题 3：如果 `/var/log/apache2` 中的日志超过 4 周会发生什么？答：最旧的日志文件将被删除，只保留最近的 4 次轮转日志。

问题 4：配置一个新的日志轮转规则如果有一个服务 `server` 生成日志文件 `server.log`，需要将其包括在日志轮转中。可以在 `/etc/logrotate.d` 中创建如下配置文件：

```
/var/log/server/server.log {
    rotate 7
    daily
    compress
    delaycompress
    missingok
    notifempty
    create 660 UTSEUSuser UTSEUSuser
}
```

— 日志每天轮转一次 (`daily`)；
— 最多保留 7 个轮转日志 (`rotate 7`)；
— 压缩所有轮转日志 (`compress`)；
— 延迟压缩上一次轮转日志 (`delaycompress`)；
— 如果日志文件缺失，则忽略 (`missingok`)；
— 如果日志为空，则不轮转 (`notifempty`)；
— 新创建的日志文件的权限为 660，属于 `UTSEUSuser` 用户和组 (`create 660 UTSEUSuser UTSEUSuser`)。

7 系统的启动与停止 (Démarrage et arrêt du système)

7.1 系统启动 (Démarrage du système)

7.1.1 启动与内核加载 (Lancement du système : boot et chargement du noyau)

在启动时，BIOS 执行位于可启动设备（硬盘、CD、U 盘等）首个扇区（512 字节）的主引导记录（Master Boot Record, MBR）。

- **MBR**：
  - 扫描磁盘以找到可引导分区（带有引导标志）；
  - 启动位于引导分区第一个扇区的引导加载程序（bootloader）。
- **引导加载程序 (bootloader)**：
  - 将内核加载到内存并执行；
  - 将初始虚拟文件系统 (`initrd.img`) 加载到内存（一个最小化的操作系统镜像）。
- 可能使用的引导加载程序：
  - Lilo (Linux Loader)；
  - Grub (Grand Unified Bootloader)。

7.1.2 内核加载后的操作 (Après le chargement du noyau)

- 内核加载完成后，启动第一个进程：`/bin/init`；
- `init` 是所有其他进程的父进程，这些进程由 `system fork()` 系统调用创建；
- `init` 读取 `/etc/inittab` 文件以了解：
  - 继续加载系统时应执行的文件；
  - 默认的运行级别 (*runlevel*)；
  - 在给定运行级别中应如何启动服务。

7.2 `/etc/inittab` 文件

- `/etc/inittab` 是一个文本文件，其中包含：
  - 以 # 开头的注释行；
  - 包含 4 个字段的条目，字段由冒号分隔，格式如下：

```
code:niveaux:action:processus
```

- 字段解释：
  - `code`：唯一标识符，可以是数字或字母；
  - `niveaux`：运行级别列表，如果为空，表示适用于所有级别；
  - `action`：指定在这些级别中如何处理进程；

— processus：要执行的进程。

— 修改配置后，可以通过以下命令向 init 发送 HUP 信号以重新加载配置：

```
kill -HUP 1
或
kill -sHUP 1
```

7.3 /etc/inittab 示例

```
# 默认运行级别
id:2:initdefault:

# 系统初始化
si::sysinit:/etc/rc.d/rc.sysinit

# 各种运行级别
l0:0:wait:/etc/rc.d/rc 0
l1:1:wait:/etc/rc.d/rc 1
l2:2:wait:/etc/rc.d/rc 2
l3:3:wait:/etc/rc.d/rc 3
l4:4:wait:/etc/rc.d/rc 4
l5:5:wait:/etc/rc.d/rc 5
l6:6:wait:/etc/rc.d/rc 6

# 监听 CTRL-ALT-DELETE
ca::ctrlaltdel:/sbin/shutdown -t3 -r now

# 启动图形模式 (xdm)
x:5:respawn:/opt/kde/bin/kdm -nodaemon

# 停机后自动关闭
pf::powerfail:/sbin/shutdown -f -h +2 "Power␣Failure;␣System␣
    Shutting␣Down"

# 如果电力恢复，则取消关机
pr:12345:powerokwait:/sbin/shutdown -c "Power␣Restored;␣Shutdown␣
    Cancelled"

# 创建虚拟终端 (tty1 到 tty6)
1:2345:respawn:/sbin/mingetty tty1
```

```
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6
```

7.4 启动脚本的步骤 (Les étapes des scripts de démarrage)

启动脚本执行以下任务：

— 分配主机名；
— 设置时区；
— 自动检查文件系统完整性 (`fsck`)；
— 挂载文件系统；
— 激活交换分区；
— 清理文件系统（如清空临时目录）；
— 配置网络接口；
— 启动本地守护进程；
— 启动网络守护进程并挂载 NFS 分区（如有必要）；
— 启动 `getty` 进程以允许用户登录。

7.4.1 示例守护进程

— `Kswapd`：管理内存与磁盘交换；
— `Nfsd`：网络文件服务器 (NFS)；
— `Httpd`：HTTP 服务；
— `Xinetd`：网络服务管理超级守护进程。

7.5 运行级别 (Niveaux ou Runlevels)

系统支持不同的运行级别，每次仅能有一个活动级别：

— `0`：关机；
— `1`：单用户模式（维护模式）；
— `2-5`：取决于操作系统，可能对应于多用户模式；
— `6`：重启。

运行级别的含义：

— `2`：多用户模式，无应用服务器；
— `3`：多用户模式，有应用服务器；
— `4, 5`：可能用于启动图形界面。

级别 0, 1, 6 的用途通常固定。

7.6 更改运行级别 (Changer de Niveaux ou Runlevels)

可以通过以下几种方式达到特定的运行级别 (*runlevel*):

— **在启动时指定**: 通过修改引导加载程序配置，或临时更改内核启动参数。

— **使用 `init` 或 `telinit` 命令**: 在系统运行时，可以切换到不同的运行级别，而无需重新启动。

— **设置默认运行级别**: 在 `/etc/inittab` 文件中修改 `initdefault` 行以配置下次启动时的默认运行级别。

— **使用 `shutdown` 命令**: `shutdown` 向 `init` 发送信号以切换运行级别。

对于 GRUB (*GRand Unified Bootloader*), 内核的默认启动参数位于 `/boot/grub/grub.conf` 文件中的 `kernel` 行。在 GRUB 启动过程中按下键“e”可以临时修改内核参数。

7.7 系统关机 (Arrêt du système)

由于内存缓冲区 (*buffers*) 的数据不会立即写入磁盘（异步写入），这提高了 I/O 性能，但在意外关机时可能丢失数据（现代日志文件系统减轻了此问题）。

关机命令 (`shutdown`) 例如，以下命令定义在 `/etc/inittab` 文件中，用于处理 `Ctrl-Alt-Del` 组合键:

```
1 ca::ctrlaltdel:/sbin/shutdown -r -t 4 now
```

以下是 `shutdown` 的快捷命令:

— `reboot`: `shutdown -r now`，重启;

— `halt`: `shutdown -h now`，关机;

— `fasthalt`: `shutdown -f now`，重启时不检查文件系统 (`fsck`)。

7.8 启动时启用或禁用服务 (Activer et désactiver des services au démarrage)

运行级别中激活的服务对应于 `/etc/rc.d/rcX.d/SNNservice` 链接，其中:

— `X`: 目标运行级别;

— `NN`: 用于确定服务启动顺序的编号;

— `service`: 位于 `/etc/rc.d/init.d` 目录中的脚本名称。

类似地，用于停止服务的链接名称为 `/etc/rc.d/rcX.d/KNNservice`。链接的语法为: `[S|K]XX<nom_du_script>`:

— `S`: 启动服务脚本，使用参数 `start`;

— `K`: 停止服务脚本，使用参数 `stop`;

— `XX`：表示启动顺序的编号。

这些链接可以通过以下方式管理：

— 手动创建或删除链接 (`ln -s`, `rm`)；

— 使用命令行工具 `chkconfig`（或 `update-rc.d`，适用于 Debian 系统）；

— 使用控制台工具 `ntsysv`；

— 使用图形化工具，例如 Webmin 或 RedHat 服务管理工具。

7.9 启动和停止服务 (Arrêter et démarrer des services)

`service` 命令用于启动或停止服务，其用法如下：

```
service <nom> <action>
```

其中：

— `nom`：服务名称（如 `/etc/init.d` 目录中的服务脚本）；

— `action`：执行的操作：

  — `stop`：停止服务；

  — `start`：启动服务；

  — `reload`：重新加载服务配置；

  — `status`：显示服务状态。

也可以使用以下语法：

```
/etc/init.d/<nom> <action>
```

8 文件管理系统 (Le système de gestion de fichiers)

8.1 文件的基本概念 (Les fichiers sous Unix)

在 Unix 中，一个文件具有以下特性：

— 始终以一个名称 (*nom*) 标识；

— 拥有唯一的 inode（包含关于文件的一些元信息）；

— 提供以下功能：

  — 打开 (*ouverture*)；

  — 关闭 (*fermeture*)；

  — 读取 (*lecture*)；

  — 写入 (*écriture*)。

文件可以是以下几种类型：

— 普通文件 (*ordinaire*)，也称为“正常”文件，表示为 `-`；

— 目录 (*répertoire*)，表示为 `d`；

— 符号链接 (*lien symbolique*)，表示为 `l`；
— 伪文件 (*pseudo-fichier*)，包括：
  — 字符设备 (*accès caractère par caractère*)，表示为 `c`；
  — 通信设备 (*dispositif de communication*)，表示为 `p`；
  — 块设备 (*accès par bloc*)，表示为 `b`。

8.2 文件系统概述 (Les systèmes de gestion de fichiers)

文件系统或文件管理系统是一种存储和组织信息的方式。它是一个按照特定格式组织的目录和文件集合。文件系统支持多种格式的读写操作。

根据文件系统的类型，数据的持久化支持可以是以下形式（示例并非详尽）：
— 本地磁盘分区（例如由 RAID 控制器管理的磁盘设备）；
— 可移动设备（例如软盘、CD-ROM）；
— 网络资源（例如 NFS 或 SMB 服务器）；
— 内存（例如 tmpfs 文件系统）；
— 虚拟空间（例如 proc、sysfs、usbdevfs 或 devpts 文件系统）；
— 模拟块设备（例如 ramdisk、loop、NBD、LVM、软件 RAID）。

8.3 创建文件系统 (Créer un système de fichiers)

最常用的文件系统是 `ext3`。创建一个新文件系统并将其永久挂载到目录树中需要以下步骤：

1. 配置一个块设备 (*périphérique de type bloc*)；
2. 使用命令 `mkfs` 创建文件系统；
3. 选择一个挂载点 (*point de montage*)；
4. 使用命令 `mount` 将文件系统挂载到挂载点；
5. 在文件 `/etc/fstab` 中添加一行，使文件系统在系统启动时自动挂载。

分区和格式化磁盘 (Partitionnement et formatage du disque dur)：
— **低级格式化 (Formatage bas niveau)**：通常由工厂完成；
— **分区 (Partitionnement)**：在操作系统安装时进行：
  — 工具：`fips, fdisk, PartitionMagic (DOS)`；
  — Linux 系统中：`fdisk, parted` 或图形化安装菜单。
— **高级格式化 (Formatage haut niveau)**：创建逻辑文件系统，取决于操作系统和目标文件系统：
  — `format`（Windows）：创建 FAT 或 NTFS 文件系统；
  — `mkfs`（Unix）：创建 Ext2、Ext3、FAT 等文件系统。
— 示例命令：

```
mkfs -t ext2 /dev/hda1
mkfs -t vfat /dev/fd0
```

— 日志文件系统 (*Système de fichiers journalisés*) 更具故障恢复能力。

多分区的优势:

— 增强系统安全性;

— 可以独立格式化某个分区而不影响其他分区;

— 某些分区可以设置为只读模式。

8.4 支持的文件系统类型 (Types de système de fichiers supportés)

以下是不同操作系统支持的文件系统类型:

操作系统 (Système d'exploitation)	支持的文件系统类型 (Types de système de fichiers supportés)
Dos	FAT16
Windows 95	FAT16
Windows 95 OSR2	FAT16, FAT32
Windows 98	FAT16, FAT32
Windows NT4	FAT, NTFS (版本 4)
Windows 2000/XP	FAT, FAT16, FAT32, NTFS (版本 4 和 5)
Linux	Ext2, Ext3, ReiserFS, Linux Swap, (FAT, NTFS, ...)
MacOS	HFS, MFS
SGI IRIX	XFS
FreeBSD, OpenBSD	UFS (Unix File System)
Sun Solaris	UFS (Unix File System)
IBM AIX	JFS (Journaled File System)

8.5 创建文件系统的示例 (Exemple de création d'un système de fichiers)

8.5.1 示例 1: 为分区 hda9 创建文件系统

```
# 使用 fdisk 创建分区
fdisk /dev/hda
[ 使用命令 "p" 显示分区表, "n" 创建新分区, "w" 写入更改到磁盘。
  如果需要, 可能需要重启。]

# 使用 mkfs 创建文件系统
```

```
mkfs -t ext3 /dev/hda9
# 挂载文件系统到 /opt
mount -t ext3 /dev/hda9 /opt
# 将挂载点添加到 /etc/fstab
echo "/dev/hda9␣/opt␣ext3␣defaults␣1␣2" >> /etc/fstab
```

8.5.2 示例 2：在“镜像文件”中创建文件系统

```
# 创建一个镜像文件
dd if=/dev/zero of=/var/local/myfs bs=1024 count=102400
[ 输入文件 = /dev/zero, 输出文件 = /var/local/myfs,
  bs = 块大小, count = 复制的块数量。]

# 使用 mkfs 创建文件系统
mkfs -t ext3 /var/local/myfs
[ 注意：由于 "/var/local/myfs" 不是块设备，系统会要求确认。]

# 挂载镜像文件到临时目录
mkdir /mnt/tmp
mount -o loop -t ext3 /var/local/myfs /mnt/tmp
将挂载点添加到 /etc/fstab
echo "/var/local/myfs␣/mnt/tmp␣ext3␣loop␣1␣2" >> /etc/fstab
```

8.6 修复文件系统 (Réparer un système de fichiers)

使用命令 `fsck` 可以检查文件系统的完整性，并在必要时尝试修复。注意：文件系统在检查时不能处于挂载状态。这通常在运行级别 1（单用户模式）下进行：

```
# 切换到运行级别 1
init 1
# 卸载文件系统
umount /usr
# 检查并修复文件系统
fsck -t ext3 -f /dev/hda7
# 重新挂载文件系统
mount /usr
# 切换到运行级别 5
init 5
```

另一种方法是在系统启动时强制检查所有本地文件系统的完整性。方法如下：

```
# 创建空文件 forcefsck
touch /forcefsck
# 重新启动系统
init 6
```

在修复 ext2 或 ext3 文件系统时，发现的孤立数据块会被放置在每个 ext2 或 ext3 文件系统根目录中的 lost+found 目录中。

8.7 挂载网络文件系统 (Monter un système de fichiers réseau)

以下是一个示例，描述如何挂载一个 Windows 文件共享：

— 共享名称：archives；

— 服务器名称：serveur；

— 域名称：WG；

— 用户名：utilisateur；

— 密码：mot_de_passe；

— 挂载点：/mnt/archives。

步骤如下：

```
# 创建挂载点
mkdir /mnt/archives
# 创建存储凭据的文件
touch /etc/passwd.smb
# 设置凭据文件的权限
chmod 600 /etc/passwd.smb
# 编辑凭据文件并添加以下内容
vi /etc/passwd.smb
username = utilisateur
password = mot_de_passe
[ 保存并退出 ]

# 挂载网络共享
mount -t smb \
  -o workgroup=WG,credentials=/etc/passwd.smb \
  //serveur/archives /mnt/archives
# 添加到 /etc/fstab
echo "//serveur/archives␣/mnt/archives␣smb␣\
␣␣workgroup=WG,credentials=/etc/passwd.smb␣0␣0" >> /etc/fstab
```

注意事项：

— 在移除可移动设备（如软盘、U 盘等）之前，必须使用命令 `umount` 将其卸载，以同步写入操作并清空磁盘缓存。
— 所有永久性挂载信息均存储在文件 `/etc/fstab` 中。

9 虚拟内存 (La mémoire virtuelle)

9.1 管理虚拟内存 (Gérer la mémoire virtuelle)

虚拟内存是一种允许运行比实际物理内存更大程序的技术。
系统可以使用以下两种存储方式来管理虚拟内存：
— 分区 (*partitions*)；
— 文件 (*fichiers*)。
注意：虚拟内存文件的性能较低。

通常使用一个分区。如果系统包含两个或更多磁盘，为每个磁盘创建一个专门用于虚拟内存的分区可以提高性能。

这些分区需要具有系统标识符 `0x82`。可以使用 Linux 的 `fdisk` 工具的 `t` 选项来修改新分区的标识符。

9.2 创建虚拟内存空间 (Créer un espace de mémoire virtuelle)

要配置一个新的虚拟内存空间并将其永久激活，需完成以下步骤：

1. 创建一个分区或空文件；
2. 使用 `mkswap` 命令为虚拟内存创建签名；
3. 使用 `swapon` 命令激活虚拟内存；
4. 在 `/etc/fstab` 文件中添加一行，以便虚拟内存在每次系统启动时自动激活；
5. 检查虚拟内存的状态。

以下是将一个 100 MB 的文件添加到虚拟内存的示例：

```
# 创建一个 100 MB 的空文件
dd if=/dev/zero of=/swapfile bs=1024 count=102400
.../...
# 创建虚拟内存签名
mkswap /swapfile
Initialisation de la version de l\textquotesingle espace de swap 1,
    taille = 104853 kB
# 激活虚拟内存
swapon /swapfile
# 将虚拟内存添加到 /etc/fstab 以便开机自动激活
```

```
echo "/swapfile␣swap␣swap␣defaults␣0␣0" >> /etc/fstab
# 检查虚拟内存的状态
swapon -s
Filename                        Type            Size    Used    Priority
/dev/hda3                       partition       524152  0       -1
/swapfile                       file            102392  0       -2
```

10 归档与压缩（Les archives et les commandes de compression）

10.1 归档文件（Archives）

`tar`（表示 *Tape ARchiver*）是一个 Unix 命令，用于管理一组文件和/或目录的归档。最初，它用于在磁带上操作归档文件。

现在，`tar` 命令广泛用于创建可以通过网络传输的归档文件（例如 `zip` 格式的文件）。

基本用法：

— 选项 `c`：创建归档；

— 选项 `f`：指定归档目标（文件或设备）。

以下示例演示了从目录 `/home` 创建两个归档：

```
cd /; tar cf /mnt/backups/backup-home.tar home
cd /; tar cf /dev/st0 home
```

第一个示例将归档保存为文件，第二个示例将归档写入 SCSI 磁带设备（如 DAT、DLT）。

注意：某些 `tar` 版本会存储文件的绝对路径，恢复归档时需小心处理。通常建议使用相对路径归档文件。

10.2 压缩归档文件（Compression des archives）

`tar` 命令本身不压缩文件。需要结合以下方法使用：

```
# 使用 compress 命令压缩
cd /; tar cf /mnt/backups/backup-home.tar home
# compress /mnt/backups/backup-home.tar

# 使用管道操作
cd /; tar cf - home | compress > /mnt/backups/backup-home.tar.Z
```

如果使用 GNU 项目提供的 `tar` 版本，可通过选项 `Z` 简化操作：

```
cd /; tar cZf /mnt/backups/backup-home.tar.gz home
```

10.3 压缩工具对比 (Les commandes compress, gzip et bzip2)

compress 是一种较老的压缩工具，主要为兼容性而使用。以下是 compress、gzip 和 bzip2 的对比：

命令 (commande)	解压命令	效率 (efficacité)	速度 (vitesse)	扩展名 (extension)
compress	uncompress	中等	中等	.Z
gzip	gunzip	良好	中等	.gz
bzip2	bunzip2	优秀	慢	.bz2

10.4 解压与列出归档内容 (Lister et extraire des archives)

列出归档内容：使用 t 和 v 选项列出归档内容，显示详细信息：

```
tar tvzf /mnt/backups/backup-home.tar.gz
.../...
```

解压归档内容：使用选项 x 提取归档文件：

```
mkdir /home/tmp-restauration
cd /home/tmp-restauration
tar xzf /mnt/backups/backup-home.tar.gz
```

如果在创建归档时使用了相对路径，内容将被解压到当前目录。否则，归档内容将恢复到原路径。

11 网络管理 (Administration réseau)

11.1 网络配置 (Configuration réseau)

网络配置的基础元素是网络接口 (*interface*)。一个系统通常有多个网络接口。这些接口根据类型命名，并按设备驱动程序加载的顺序以及其在总线上的物理位置编号。

常见的接口类型：

— lo：环回接口 (*loopback*)；

— ethN：以太网接口；

— pppN：PPP 链接。

lo 接口总是与 IP 地址 127.0.0.1 关联，系统安装过程会自动完成该关联。以太网接口在检测到时，也会在安装过程中自动配置。

11.2 加载网络驱动 (Les pilotes de périphériques réseau)

网络设备驱动程序通常以内核模块的形式提供，可以通过以下两种方式激活：

— 当 `ifconfig` 调用一个未知接口时，内核自动加载模块。需要在 `/etc/modules.conf` 文件中定义别名，例如：

```
alias eth0 3c59x
```

— 启动时通过脚本显式加载模块。

网络模块的参数可以在 `modules.conf` 文件中配置，例如：

```
alias eth0 ne
options ne io=0x220 irq=11
alias eth1 e1000
options e1000 Speed=1000 RxDescriptors=128
```

11.3 `ifconfig` 命令

`ifconfig` 命令用于激活和配置网络接口。选项 `-a` 显示系统中所有网络接口的详细信息，例如：

```
# ifconfig -a
eth0   Lien encap:Ethernet  HWaddr 00:20:ED:36:3C:EE
       inet adr:192.168.200.200  Bcast:192.168.200.255  Masque:255.255.255.0
       UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
       RX packets:13556 errors:0 dropped:0 overruns:0 frame:0
       TX packets:13292 errors:0 dropped:0 overruns:0 carrier:0
       collisions:10 txqueuelen:100
       RX bytes:1743502 (1.7 Mb)  TX bytes:3764520 (4.1 Mb)
lo     Lien encap:Boucle locale
       inet adr:127.0.0.1  Masque:255.0.0.0
       UP LOOPBACK RUNNING  MTU:16436  Metric:1
       RX packets:92894 errors:0 dropped:0 overruns:0 frame:0
       TX packets:92894 errors:0 dropped:0 overruns:0 carrier:0
       collisions:0 txqueuelen:0
       RX bytes:6522189 (62.5 Mb)  TX bytes:6522189 (62.5 Mb)
```

接口可以通过以下方式激活或禁用：

```
# 激活接口
ifconfig eth0 up

# 禁用接口
ifconfig eth0 down
```

11.4 标准网络配置文件 (Les fichiers standards de configuration réseau)

以下文件描述了系统 IP 层的配置：

— `/etc/protocols`：IP 协议；

— `/etc/services`：TCP 和 UDP 服务；

— `/etc/rpc`：RPC 服务；

— `/etc/hosts`：静态主机名；

— `/etc/networks`：静态网络名称。

这些文件大多作为数据库，内容标准化，适用于需要使用 DNS 服务器的网络。

11.5 `ifup` 和 `ifdown` 命令

`ifup` 命令用于激活或刷新网络接口配置，`ifdown` 命令用于禁用网络接口。例如：

```
# 激活接口
ifup eth0

# 禁用接口
ifdown eth0
```

选项 `-a` 可同时配置 `/etc/network/interfaces` 文件中定义的所有接口。

11.6 路由表管理

路由表的管理通过命令 `route` 实现。

要显示静态路由表，可以使用以下两种命令之一：

```
# 使用 route 命令查看路由表
route
Table de routage IP du noyau
Destination     Passerelle        Genmask         Indic Metric Ref    Use Iface
192.168.200.0   *                 255.255.255.0   U     0      0        0 eth0
default         lucky.luke.net    0.0.0.0         UG    0      0        0 eth0

# 使用 netstat -r 命令查看路由表
netstat -r
Table de routage IP du noyau
Destination     Passerelle        Genmask         Indic MSS Fenêtre irtt Iface
192.168.200.0   *                 255.255.255.0   U     0    0       0   eth0
default         lucky.luke.net    0.0.0.0         UG    0    0       0   eth0
```

查看路由表：route 和 netstat -r

注意：如果在命令中使用 `-n` 选项，IP 地址将不会被解析。这在需要快速查看路由表或避免 DNS 查询时非常有用。

11.7 静态路由 (Routes statiques)

11.7.1 添加路由

以下是三种常见的路由添加方式：

— **添加到单个主机的路由**：

```
route add -host 10.20.30.1 gw 192.168.200.2
```

添加主机路由

— **添加到网络的路由**：

```
route add -net 10.20.30.0 netmask 255.255.0.0 dev eth0
```

添加网络路由

— **添加默认路由**：

```
route add default gw gw.reseau.fr
```

添加默认路由

注意：路由目标可以是网关 (`gw`) 或接口 (`dev`)。IP 地址或网关名称均可。

11.7.2 删除路由

删除路由的示例：

```
route del -net 10.20.30.0 netmask 255.255.0.0
```

删除网络路由

11.8 名称解析 (Résolution des noms)

11.8.1 配置名称解析机制

名称解析通过以下配置文件实现：

— `/etc/host.conf`：控制名称解析顺序，例如：

```
order hosts,bind
```

/etc/host.conf 示例

— `/etc/nsswitch.conf`：定义解析顺序，例如：

```
passwd:     files
shadow:     files
group:      files
```

```
hosts:      files dns
```

/etc/nsswitch.conf 示例

11.8.2 DNS 配置

通过 /etc/resolv.conf 文件进行配置:

```
search luke.net
nameserver 192.168.200.1
nameserver 192.168.200.10
```

/etc/resolv.conf 示例

静态主机名在 /etc/hosts 文件中定义，例如:

```
127.0.0.1       localhost.localdomain localhost
192.168.200.201 jack.luke.net jack
192.168.200.1   gw
```

/etc/hosts 示例

11.9 超级服务管理 (Les super-serveurs inetd et xinetd)

11.9.1 inetd 配置

inetd 是一个超级守护进程，负责监听 TCP 和 UDP 端口。当某个端口接收到连接时，inetd 会启动相应的服务。

其配置文件位于 /etc/inetd.conf，示例如下:

```
ftp     stream  tcp   nowait  root  /usr/sbin/in.ftpd  in.ftpd
telnet  stream  tcp   nowait  root  /usr/sbin/in.telnetd in.telnetd
```

inetd.conf 配置示例

11.9.2 xinetd 配置

xinetd 是 inetd 的改进版，其配置文件为 /etc/xinetd.conf，支持更复杂的功能。例如:

```
service ftp
{
    socket_type = stream
    protocol = tcp
```

```
    wait = no
    user = root
    server = /usr/sbin/in.ftpd
}
```

xinetd.conf 配置示例

11.10 邮件中继配置 (Relais de messagerie)

11.10.1 使用 Postfix 配置

在 Postfix 中，可以通过设置 `relayhost` 参数实现中继配置，例如：

```
relayhost=[smtp-gw.domaine.fr]
```

Postfix 配置示例

配置文件路径为：`/etc/postfix/main.cf`。

11.10.2 使用 Sendmail 配置

在 Sendmail 中，有两种方式：

1. 使用 `m4` 生成配置：

```
define(`SMART_HOST',␣`[smtp-gw.domaine.fr]')
# 生成配置文件
make -C /etc/mail
```

m4 示例

2. 直接编辑 `sendmail.cf`：

```
DSsmtp-gw.domaine.fr
```

sendmail.cf 示例

11.11 ARP 工具 (Outils - arp)

11.11.1 La commande arp

`arp` 命令用于查看或修改网络接口的 ARP 缓存表。这张表可以是静态的或动态的，显示 IP 地址与 MAC 地址（以太网地址）之间的对应关系。

每次新请求时，接口的 ARP 缓存都会更新，并记录一个新的条目。该条目具有生存时间（TTL, *Time To Live*）。

以下是使用 `arp -va` 命令获取 ARP 缓存的示例：

```
? (192.168.1.2) at 00:40:33:2D:B5:DD [ether] on eth0
> Entries: 1   Skipped: 0   Found: 1
```

查看 ARP 缓存

在此结果中，我们可以看到 IP 地址与对应的 MAC 地址。此表仅包含一个条目。

常用选项：

— 添加静态条目：`arp -s 192.168.1.2 00:40:33:2D:B5:DD`

— 删除条目：`arp -d 192.168.1.2`

11.12 Route 工具 (Outils - route)

11.12.1 La commande route

`route` 命令用于定义数据包从源到目的地的路径。它可以配置主机、交换机或路由器的路由表。

路由类型：

— **静态路由 (*Routage statique*)**：明确规定数据包的路径。

— **动态路由 (*Routage dynamique*)**：通过算法根据网络拓扑动态更新路由表。

一般情况下，静态路由适用于本地网络，而动态路由适用于大型或扩展的网络。

11.12.2 路由表示例 (Exemple de table de routage)

以下是路由表的一个示例：

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref Use Iface
192.168.1.0     *               255.255.255.0   U     0      0   2   eth0
127.0.0.0       *               255.0.0.0       U     0      0   2   lo
default         192.168.1.9     0.0.0.0         UG    0      0   10  eth0
```

示例路由表

字段解释：

— **Destination**：路由目标的地址。

— **Gateway**：到达目标的网关地址。如果为 `*`，则直接连接。

— **Genmask**：子网掩码。

— **Flags**：状态标志（例如 `U` 表示启用，`G` 表示网关）。

— **Metric**：路由成本（默认为 0）。

— **Ref**：依赖该路由的数量。

— **Use**：路由表中该条目的使用次数。

— **Iface**：关联的网络接口，例如 `eth0`。

示例解释：- 第三行表示默认路由 (*default route*)，数据包通过网关 `192.168.1.9` 发送。

11.12.3 添加或删除路由 (Ajout ou suppression d'une route)

`route` 命令可以用来添加或删除路由：

— 添加路由：`route add [net | host] addr [gw passerelle] [metric coût] [netmask masque] [dev interface]`

— 删除路由：`route del [net | host] addr [netmask masque]`

参数解释：

— `net | host`：目标地址是网络或主机。

— `addr`：目标地址。

— `gw passerelle`：网关地址。

— `metric coût`：路由的成本值。

— `netmask masque`：目标地址的子网掩码。

— `dev interface`：与路由关联的网络接口。

11.12.4 路由添加与删除示例 (Exemples d'ajout et de suppression de routes)

以下是一些使用 `route` 命令添加和删除路由的示例：

```
# 为地址 127.0.0.1 添加路由，通过接口 lo
route add 127.0.0.1 lo
# 为网络 192.168.2.0 添加路由，通过接口 eth0
route add -net 192.168.2.0 eth0
# 为主机 saturne.foo.org 添加路由，通过接口 eth0
route add saturne.foo.org
# 为本地机器设置默认路由，使用网关 ariane
route add default gw ariane
# 为子网添加路由，指定子网掩码
route add duschmoll netmask 255.255.255.192
```

添加路由示例

删除路由示例：

```
# 删除针对网络 192.168.1.0 的路由
route del -net 192.168.1.0
# 删除指定网络的路由
route del -net toutbet-net
```

删除路由示例

11.13 Netstat 工具 (Outils - netstat)

11.13.1 La commande netstat

`netstat` 命令用于测试网络配置、查看连接状态、统计数据流量，特别适合监控服务器。

常用参数:

— `-a`: 显示所有连接状态。
— `-i`: 显示网络接口的统计信息。
— `-c`: 定期刷新网络状态。
— `-n`: 以数字形式显示地址和端口。
— `-r`: 显示路由表。
— `-t`: 显示 TCP 套接字信息。
— `-u`: 显示 UDP 套接字信息。

11.13.2 网络连接状态示例 (Etat des connexions réseau)

以下是 `netstat` 显示的网络连接状态示例:

```
Proto Recv-Q Send-Q Local Address           Foreign Address         State
Tcp   0      126    uranus.planete.n:telnet 192.168.1.2:1037        ESTABLISHED
Udp   0      0      uranus.plan:netbios-dgm *:*                     *
Udp   0      0      uranus.plan:netbios-ns  *:*                     *
```

网络连接状态示例

Unix 域套接字示例:

```
Proto RefCnt Flags       Type       State         I-Node   Path
unix  2      [ ]         STREAM     CONNECTED     1990     /dev/log
unix  2      [ ]         STREAM     CONNECTED     1989
unix  1      [ ]         DGRAM                    1955
```

Unix 域套接字示例

11.13.3 参数解释 (Explications sur les paramètres)

以下是 `netstat` 输出的关键字段说明:
— **Proto**: 协议类型，例如 TCP 或 UDP。
— **Recv-Q**: 接收队列中等待处理的字节数。
— **Send-Q**: 发送队列中等待传输的字节数。
— **Local Address**: 本地地址和端口。
— **Foreign Address**: 远程地址和端口。
— **State**: 连接状态，例如 `ESTABLISHED` 表示连接已建立。

连接状态可能的取值：

— `ESTABLISHED`：连接已建立。
— `SYN_SENT`：尝试连接远程主机。
— `SYN_RECV`：远程主机接受连接。
— `FIN_WAIT2`：连接已关闭。
— `CLOSE`：套接字未使用。
— `CLOSE_WAIT`：等待关闭的确认。
— `LISTEN`：监听连接请求。
— `UNKNOWN`：未知状态。

11.13.4 通过 `netstat -r` 查看路由表 (Affichage et état des tables de routage avec netstat)

以下是使用 `netstat -r` 命令查看路由表的示例：

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
192.168.1.0     *               255.255.255.0   U       1500 0       0   eth0
127.0.0.0       *               255.0.0.0       U       3584 0       0   lo
```

示例路由表

字段解释：

— **Destination**：数据包的目标地址。
— **Gateway**：网关地址；如果为 `*`，表示直接连接。
— **Genmask**：目标地址的子网掩码。
— **Flags**：标志，例如：
  — `G`：表示使用网关。
  — `U`：表示接口已启用。
  — `H`：表示此路由只能到达单一主机。
— **MSS**：最大分段大小。
— **Window**：TCP 窗口大小。
— **irtt**：初始往返时间。
— **Iface**：使用的网络接口，例如 `eth0` 或 `lo`。

11.14 其他网络工具 (Outils - divers)

以下工具在配置 Linux 网络时非常有用：

— 测试 IP 连接性：

```
    ping jo.luke.net
```

— 跟踪从一个系统到另一个系统的路由路径：

```
traceroute jo.luke.net
```

— 使用 DNS 解析网络名称：

```
host lucky.luke.net
host -t mx luke.net
host 192.168.200.1
```