Linux 使用笔记

adamanteye

1. 发行版选择

以下评述基于我或朋友们的看法,其中我只尝试过 Arch, Gentoo, Debian 三种发行版.

1.1. Arch

兜兜转转最喜欢的发行版,因为 pacman 好用,软件包丰富,文档最完善,脚本 PKGBUILD 很好写, AUR 对用户很友好.

缺点是打包比较粗,这方面 Debian, OpenSUSE 或许算是打包细致的范例.

1.2. Alpine

目前只在 docker 中用过,听说 Alpine Edge 也可以被拿来当桌面发行版.

1.3. eweOS

跟 Arch 的哲学很像,但是 musl,现在还处于开发中.

1.4. Debian

自己的服务器上都在跑 Debian,因为它自由稳定,不喜欢 Ubuntu 正因为其是大公司的推广.

1.5. Gentoo

拿来在服务器上跑也会比较合适,不过源码分发比较折磨人.

emerge 以及一众工具好细碎,另外好慢.

1.6. Fedora

软件包更新甚至比 Gentoo 激进.

1.7. OpenSUSE

被称为最适合 KDE 的发行版,大概都是德国人开发的吧.

2. 多用户管理

2.1. 登入用户信息

who 与 w 命令可以查询当前登入的用户,不过在 gentoo prefix 下运行不会查到任何用户:

```shell
w
```

```
23:18:07 up  2:43,  1 user,  load average: 2.12, 3.14, 3.20
USER     TTY      LOGIN@   IDLE   JCPU   PCPU WHAT
adamante tty1     20:34    2:43m  6:51m  ?    footclient
```

```shell
who
```

```
adamanteye tty1         2025-01-25 20:34
```

3. 网络管理

3.1. 地址与路由

访问 4.ipw.cn 与 6.ipw.cn 可以看到出口 IP 地址.

快速 ping 所有主机:

```shell
nmap -sn 192.168.1.0/24
```

或者

```shell
for i in {1..254}; do sudo ping -c 1 -W 1 192.168.1.$i | grep "bytes from" && echo "192.168.1.$i is alive"; done
```

3.2. 端口

ss 命令由 iproute2 提供,功能与 netstat 类似,但信息更全.

3.3. interfaces

这是 Debian 上的传统方法.

配置 DHCP:

```
# /etc/network/interfaces
auto eno1
iface eno1 inet dhcp
iface eno1 inet6 auto
```

3.4. NetworkManager

编写 dispatcher 可以实现自动切换有线,无线连接,详见 NetworkManager: automatically switch between Ethernet and Wi-Fi.

dnsmasq 可以作为 NetworkManager 的本地 DNS 缓存服务器.且可以为不同的域名选择不同的 DNS 服务器:

```
# /etc/NetworkManager/dnsmasq.d/server.conf
server=/tsinghua.edu.cn/166.111.8.28
server=/tsinghua.edu.cn/2402:f000:1:801::8:28
server=/cn/101.6.6.6
server=101.101.101.101
server=2001:de4::101
```

```
# /etc/NetworkManager/NetworkManager.conf
[main]
dns=dnsmasq
```

3.5. 设备管理

rfkill 可以启用/禁用 WIFI,蓝牙在内的无线设备.例如:

```Shell
rfkill # list wireless devices
rfkill unblock bluetooth
```

3.6. 网络测绘

参考进行网络测绘的方法与挑战 | Ajax's Blog.

4. 启动引导

4.1. GRUB

从 grub 命令行中引导系统.

/分区为 btrfs 的例子:

```Shell
grub> ls
grub> set root=(hd0,gpt2)
grub> linux /@rootfs/boot/vmlinuz-6.1.0-30-amd64 root=/dev/sda2 rw rootflags=subvol=@rootfs
grub> initrd /@rootfs/boot/initrd.img-6.1.0-30-amd64
grub> boot
```

/分区为 ext4 的例子:

```Shell
grub> ls
grub> set root=(hd0,gpt2)
grub> linux /boot/vmlinuz-6.1.0-30-amd64 root=/dev/sda2
grub> initrd /boot/initrd.img-6.1.0-30-amd64
grub> boot
```

5. 软件分发

5.1. 手册页

scdoc 自定义了与 Markdown 相近的语法,可以用来生成 man 手册页:

```Shell
scdoc < foot.1.scd > foot.1
gzip foot.1
man ./foot.1.gz
```

5.2. Shell 补全

5.2.1. fish

fish_update_completions 可以从系统 man pages 生成补全.

5.3. Arch Linux 打包

namcap 可以方便地检查 PKGBUILD 和打好的包当中出现的错误,以及所依赖的动态库.

5.4. Debian 打包

参考:

- Debian Policy Manual
- Debian 维护者指南
- Debian 打包教程

Debian 打包体系随时间演化,在 Debian Trends 可以看到不同源码格式,打包工具的使用比例.

我构建了一个 debian 的 docker 环境,adamanteye/images/debian-builder,用来完成在干净系统下的打包.

一般的打包流程:

```Shell
tar xaf example-0.1.0.tar.gz
cd example-0.1.0
debmake -b':sh' -x1   # 选一个模板
vim debian/control    # 以及其他文件
rm -rf debian/patches # 以及其他用不到的文件
debuild
```

6. 规范

6.1. 时间日期

查看 strftime(3) 了解可用的格式化选项,例如 2025 03 05 对应 %Y %m %d.

7. Shell 技巧

7.1. readline 键位

- Ctrl A 跳到行首
- Ctrl E 跳到行尾
- Ctrl U 删除光标位置到行首的所有内容
- Ctrl K 删除光标位置到行尾的所有内容
- Ctrl W 删除光标位置前一个单词
- Ctrl Y 粘贴最后一次删除的内容
- Ctrl L 清屏
- Ctrl F 向前移动一个字符
- Ctrl B 向后移动一个字符
- Alt F 向前移动一个单词
- Alt B 向后移动一个单词
- Ctrl R 启动反向搜索历史命令
- Ctrl S 启动正向搜索历史命令
- Ctrl G 取消搜索

常见 shell 和 emacs 都支持这些键位.

7.2. 彩色输出

以下是一些可启用彩色输出的命令:

```shell
alias ls='ls --color=auto'
alias ip='ip --color=auto'
alias grep='grep --color=auto'
```

使用 bat 来代替 cat 和 less,可以无感知实现大多数语言的语法高亮:

```shell
alias cat='bat --style=plain --paging=never'
alias less='bat  --style=plain'
```

不过, bat 作为 MANPAGER 后会丢失下划线格式:

```shell
export MANPAGER="sh -c 'col -bx | bat -l man -p'"
```

作为对比,查看依靠 less 的彩色化输出:

```shell
export MANPAGER="less -M -R -i --use-color -Dd+R -Du+B -DHkC -j5"
```

7.3. 实时输出

许多命令提供-f/--follow 选项,例如:

```shell
sudo tail -f /var/log/xray/access.log
```

```shell
docker compose logs -f
```

```shell
dmesg -w
```

此外,可以利用 watch 以固定间隔更新输出:

```shell
watch -n 3 "ps aux | grep node"
```

7.4. 别名

用 alias 为命令创建别名:

```shell
alias hx=helix
```

对于 git 子命令,可以设置 git 别名:

```
# ~/.gitconfig
[alias]
    ss = status
    kmt = commit
    chk = checkout
    br = branch
    ps = push
```

随后可以用 git kmt 代替 git commit.

7.5. 定时任务

crontab guru 可以在线编辑验证 crontab 语法.

7.6. ssh

配置 ssh 转发可以在服务器上使用本地的私钥

```
Host foo
  ForwardAgent yes
```

如果端口被阻断,需要配置网络代理以通过代理服务器访问 ssh 服务器:

```
Host github.com
  Hostname ssh.github.com
  Port 443
  ProxyCommand nc -X connect -x [::1]:10801 %h %p
```

Arch Linux 的 openbsd-netcat 提供了 nc 命令.

7.7. 临时文件

创建临时文件或临时文件夹:

```shell
mktemp example.XXXXXXXX
```

7.8. 文本处理

grep, sed, awk 是最常使用的文本处理程序.

```shell
cat /usr/share/fortune/chinese | sed 's/\[[^m]*m//g' > chinese-without-color
```

8. 实用程序

8.1. Git

8.1.1. Hooks

commit-msg 在提交信息编辑完成后,最终提交前执行. 可以验证或修改最终的提交信息.

prepare-commit-msg 在生成提交信息后,打开编辑器前执行.在提交时增加额外信息.

8.2. 桌面环境

之前用 hyprland,发现依赖太重,于是切换到 niri. 此外 niri 的标签页交互很舒适.

此外之前也用过很久的 KDE,同样因为太笨重而切换了平铺式桌面管理器.

8.3. 编辑器

helix-editor/helix 在绝大多数发行版都已经得到了支持,但是 debian 尚且没有打包.

8.4. 终端与 Shell

foot 是轻量的 wayland 终端,支持 img2sixel.

fish 相比 zsh 速度更快,且 4.0 版本已经成功用 Rust 重写.

8.5. 浏览器

firefox 与 zotero 都可以配置多个 profile,从而实现插件,设置,书签等的隔离:

```shell
firefox --profile "$HOME.mozilla/firefox/crawler/"
```

通过改变浏览器 UA,可以绕开相当多的付费墙. 例如 firefox 在 about:config 选项卡中,设置 general.useragent.override 为:

```
Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)
```

经过测试,这对端传媒有效.可以为 google bot UA[1] 配置与日常使用隔离的 profile.

查看当前的浏览器指纹: What browser?

8.6. 文件管理器

sxyazi/yazi 是使用 Rust 编写的命令行文件管理器.

8.7. 文档手册

tealdeer 提供了相当好的 tldr 体验,可以查看大部分命令的简短文档.

8.8. 输入法

aur/fcitx5-pinyin-sougou-dict-git 提供了搜狗词库.

8.9. 邮件客户端

thunderbird 几乎开箱即用,不过一些高级用户会选择 neomutt/mutt,两者配置基本兼容.

对于 mutt,既可以用自带的 IMAP 协议,也可以使用外部收取程序,如 offlineimap 以及 msmtp.

参考配置教程:

- Mutt: 阅读邮件列表 | FancySeeker
- Mutt 配置: 一种实践的部署方式
- OfflineIMAP examples

8.10. tmux

- 左右布局: Ctrl B, %
- 向左切换布局: Ctrl B, 方向左

8.11. 桌面通知

libnotify 提供了以下命令:

```shell
notify-send -t 5000 "Your Title or App Name" "a message that will be displayed for 5000ms"
```

如果还想播放声音,可以使用 libpulse 提供的 paplay:

```shell
paplay /usr/share/sounds/freedesktop/stereo/message.oga
```

[1] 查看其他浏览器 UA: List of User Agents

注意音频由 sound-theme-freedesktop 提供.

8.12. RSS 订阅

如果熟悉 mutt,对 newsboat 应当同样很有好感.

8.13. 排版软件

typst 可以替代 LaTeX.

8.13.1. 字体

OpenType 字体有一系列 feature 可以启用,参考 Text Function - Typst Documentation 以及 Registered features (OpenType 1.9.1) - Typography | Microsoft Learn. 例如:

```typst
#let smcp(it) = {
  set text(features: ("smcp",))
  it
}
```

8.13.2. 配色

- 猫布奇诺调色盘
- OKLCH Color Picker & Converter

8.13.3. 排版原则

- Butterick's Practical Typography

8.14. 配置管理

GNU stow 利用软链接集中地管理配置文件,可以配合 git 进行版本控制和备份.

我自己的配置文件管理在 adamanteye/dotfiles.

8.15. 待办管理

Taskwarrior 功能丰富,更新到 3.0 版本后改变了远程同步的方式,可以自己托管远程同步服务.

Taskwarrior 创建循环任务:

```shell
task recur:2d due:eod add 吃山楂片
```

8.16. 密码管理

pass 基于 gpg,在本地管理密码,并且支持用 git 进行版本控制,可以一同配置 GitHub 远程仓库来备份.

8.17. 音视频处理

使用 ffmpeg 连接视频:

```shell
ffmpeg -f concat -i filelist.txt -c copy out.mp4
```

其中 filelist.txt 的内容为:

```
file '1.mp4'
file '2.mp4'
```

图形化剪辑软件 shotcut 使用 qt6,界面现代.

8.18. PDF

poppler 包提供了 pdfseparate 以及 pdfunite 两个包,可拆分合并指定页范围的 PDF 文件.

8.19. 远程文件系统

sshfs 可以通过 ssh 连接挂载远程文件系统.

8.20. 逆向

Hash 推荐我用 binaryninja-free.

8.21. 资源监控

8.21.1. 文件系统

ncdu 是采用 ncurses 界面的磁盘占用统计工具,比 du 命令更好用,带有彩色模式:

```
# /etc/ncdu.conf
--color dark-bg
```

使用 fio 测试顺序读写,随机读写等情景下的性能:

```shell
fio -filename=/home/adamanteye/test -direct=1 -iodepth 1 -thread -rw=randrw -bs=4k -size=2G -numjobs=5 -runtime=10 -group_reporting -name=mytest | tee randrw.log
```

8.21.2. CPU 信息

```shell
cat /proc/cpuinfo
```

8.21.3. 温度监控

读取硬盘温度,使用 smartctl:

```shell
sudo smartctl --all /dev/sda
```

读取 CPU 温度,使用 lm-sensors:

```shell
sudo sensors-detect  # 初次配置 lm-sensors
```

```
2 sensors                # 按照配置读取温度
```

8.21.4. 风扇控制

参考 jhatler/ipmi-fanctrl-dell_r630.sh:

```
1 sudo ipmitool raw 0x30 0x30 0x01 0x00
  # enable manual fan control
2 sudo ipmitool raw 0x30 0x30 0x02 0xff 0x14  #
  set fan speed to 20%
```

Shell

8.21.5. 功率监控

```
1 ipmitool dcmi power reading
```

Shell

9. 包管理

体验过 pacman, apt, emerge,其中还是 pacman 的体验最好(毕竟是功能最简陋的).

Repology 列出了常见发行版上的打包情况,不过没有 gentoo 的.

9.1. pacman

列出所有显式安装的包:

```
1 pacman -Qe
```

Shell

列出所有悬垂包:

```
1 pacman -Qtdq
```

Shell

更新文件数据库:

```
1 pacman -Fy
```

Shell

寻找包含指定文件名的包:

```
1 pacman -F tldr
```

Shell

9.2. AUR

之前使用 Juger/yay,现在我迁移到了 Morganamilo/paru.

9.3. dpkg & apt

解压 deb 包:

```
1 ar x example_0.1.0-1_all.deb
```

Shell

列出文件:

```
1 dpkg-deb -c example_0.1.0-1_all.deb
```

Shell

显示元信息:

```
1 dpkg-deb -I example_0.1.0-1_all.deb
```

Shell

10. 守护程序

10.1. macOS

macOS 所使用的守护进程管理是 launchd,管理系统级或用户级的守护程序.

要编写<LABEL>.plist 文件,参考 macOS daemons and agents 以及 Creating Launch Daemons and Agents.

11. 日志

11.1. systemd

清除 10 天前的所有日志:

```
1 sudo journalctl --vacuum-time 10d
```

Shell

输出特定服务的日志:

```
1 sudo journalctl --unit github-
  actions-runner
```

Shell

注意希望使用如果非 sudo 身份查看日志,需要在 systemd-journal 用户组当中.

12. 服务器

12.1. Dell Power Edge R630

我这台 R630 上装的阵列卡是 H330(小卡),可以在 BIOS 里面改成 HBA 模式,即硬盘直通.

双路 E5-2680 处理器,2 条 32GB 内存,外加 4 个 2.5 英寸硬盘在开机空载时的耗电量大约为 161W.仅主板通电的功率为 10W 左右.

12.2. 文件系统

12.2.1. ZFS

zpool history 可以查看操作历史,包括 zpool create, zpool add 等的记录.前提是 pool 还在,如果已经被 zpool destroy 了,相应的历史都会删除.

```
1 zpool create -o ashift=12 oskar
  draid:2d \
```

Shell

```
/dev/disk/by-id/ata-TOSHIBA_MQ02ABF100_Z6NUPPCRT \
/dev/disk/by-id/scsi-35000cca02d7d75ec \
/dev/disk/by-id/scsi-35000cca02d7d78f8 \
-f  # in case they are of different sizes
```

dRAID 是 raidz(1-3)的封装,手册中给出的创建格式为:

```
draid[parity][:datad][:childrenc][:sparess]
...
In regards to I/O, performance is similar to raidz since, for any read, all D data disks must be accessed. Delivered random IOPS can be reasonably approximated as floor((N-S)/(D+P))*single_drive_IOPS.

A dRAID with N disks of size X, D data disks per redundancy group, P parity level, and S distributed hot spares can hold approximately (N-S)*(D/(D+P))*X bytes and can withstand P devices failing without losing data.
```

children 是所有盘(包括热备盘)的数量, parity 指定奇偶校验级别(1-3),默认为 1.

例如 draid2:7d:10c:1s 表示 10 个磁盘中设置 1 个热备盘,剩下 9 个磁盘为一个 group,含有 7 个数据盘以及 2 个校验盘.这可以承受 2 个磁盘的损坏,并在第一个损坏发生时立刻投入 1 个热备盘进行恢复(所以实际上能承受 3 个磁盘损坏而不丢失数据).

单个 group 相对于单块硬盘的吞吐量为:

$$\left\lfloor \frac{c-s}{d+p} \right\rfloor$$

其中c是 children 的数量.注意这里是理论值,实际上还受内存缓存,是否开启压缩等因素影响.

单个 group 相对于单块硬盘的存储量为:

$$\frac{d(c-s)}{d+p}$$

注意奇偶校验和热备实际上是分散在所有磁盘上的,这也是为什么不能创建后再修改热备盘的数量.

为 ZFS 文件系统添加 dataset:

```shell
sudo zfs create -o compression=zstd -o mountpoint=/home oskar/home
```

也可以改变挂载点

```shell
sudo zfs set mountpoint=/srv oskar/home
```

参考:

- zpoolconcepts(7)
- zpool-create(8)
- zfs-create(8)
- zfsprops(7)
- ZFS dRAID Basics and Configuration - Thomas-Krenn-Wiki-en
- Linux 文件大小浅谈
- What does the ZFS Metadata Special Device do? · openzfs/zfs · Discussion 14542
- 系统中的大多数文件有多大？ - Farseerfc 的小窝
- zpoolconcepts.7 — OpenZFS documentation
- OpenZFS dRAID - A Complete Guide - Resources - TrueNAS Community Forums

12.2.2. Btrfs

12.2.3. NFS

参考:

- NFSServerSetup - Debian Wiki
- NFSv4Howto - Community Help Wiki

在服务端上:

```shell
apt install nfs-kernel-server
sudo -e /etc/exports
```

写入以下内容:

```
/oskar/elisabeth 192.168.0.3(rw,sync,no_root_squash,no_subtree_che
```

在客戶端上:

```shell
apt install nfs-common
mount -t nfs4 -o proto=tcp,port=2049 heloise.adamanteye.cc:/oskar/elisabeth /srv
```

或者写入/etc/fstab:

```
heloise.adamanteye.cc:/oskar/elisabeth  /srv/
nfs4  _netdev,auto,defaults 0 0
```

12.3. docker

从含 Dockerfile 的路径构建镜像:

```
docker buildx build --tag
nvchecker:master .
```

Shell

交互式运行容器,设置代理,挂载本地路径:

```
docker run -it --rm --name debian --
network host -e HTTP_PROXY=http://[::
1]:10801 -v "$HOME/Documents/debian:/
home/debian:rw" -v "/etc/wgetrc:/etc/
wgetrc:ro" -e DEBFULLNAME -e DEBEMAIL
ghcr.io/adamanteye/debian-
builder:master
```

Shell

删除所有未使用镜像(不止 dangling):

```
docker image prune -a
```

Shell

12.4. 网络服务

12.4.1. nginx

启用 OSCP Stapling:

```
ssl_stapling on;
ssl_stapling_verify on;
```

12.4.2. caddy

mholt/caddy-webdav 为 caddy 扩展了 webdav 模块.

12.4.3. 压力测试

用 hatoo/oha: Ohayou(おはよう)产生 HTTP 流量.

12.5. 证书

acmesh-official/acme.sh 可以自动签发与更新证书.

将证书安装到指定位置:

```
./acme.sh --install-cert -d
'adamanteye.cc' \
    --fullchain-file /srv/cert/
all.adamanteye.cc.fullchain \
    --key-file /srv/cert/all.adamanteye.cc.key
```

Shell

12.5.1. 代理

代理使用者会有相当多的特征,可以依次进行探测[2].

12.5.2. Telegram Bot

首先阅读 From BotFather to 'Hello World',这里讲解了机器人开发的基础知识.

13. 内核

13.1. dkms

[2]参见 Avoiding Live VPN/Proxy Detection · Issue 445 · net4people/bbs