

Table of Contents

Table of Contents

FreeBSD 从入门到跑路	1. 1
----------------	------

第〇章 FreeBSD 中文社区

第一节 FreeBSD FAQ	2. 1
第二节 FreeBSD 中文社区（CFC）发展规划	2. 2
第三节 FreeBSD 的不足之处	2. 3

第一章 走近 FreeBSD

第一节 什么是 UNIX	3.1
第二节 什么是 Unix-like	3.2
第三节 什么是 Linux	3.3
第四节 FreeBSD 与其他操作系统	3.4
第五节 为什么要使用 FreeBSD	3.5
第六节 所谓开源哲学	3.6
第七节 其他 BSD 简介	3.7
第八节 Linux 用户迁移指北	3.8
第九节 参考资料与贡献者名单	3.9
第十节 编撰说明	3.10

第二章 安装 FreeBSD

第〇节 图解安装	4.1
第一节 三种虚拟机与 FreeBSD 版本比较	4.2
第二节 FreeBSD 13.0 安装——基于 Virtual Box	4.3
第三节 FreeBSD 13.0 安装——基于 Vmware Workstation Pro 16	4.4
第四节 腾讯云轻量云及其他服务器 dd 安装 FreeBSD	4.5
第五节 手动安装 FreeBSD	4.6
第六节 ee 用法及网络配置	4.7
第七节 常用软件 与 SSH 配置	4.8
第八节 物理机安装与硬件选配	4.9
第九节 物理机下显卡的配置	4.10
第十节 物理机下触摸板的设置	4.11
第十一节 物理机声卡与网卡设置	4.12
第十二节 打印机的安装	4.13
第十三节 无线蓝牙鼠标的设置	4.14

第三章 软件源及包管理器

第〇节 包管理器概述	5. 1
第一节 FreeBSD 镜像站现状	5. 2
第二节 FreeBSD 换源方式	5. 3
第三节 gitup 的用法	5. 4
第四节 软件包管理器 pkg 的用法	5. 5
第五节 通过源代码 ports 方式安装软件	5. 6
第六节 通过 DVD 安装软件	5. 7

第四章 桌面安装

第〇节 概述	6.1
第一节 安装 Xorg	6.2
第二节 安装 KDE 5	6.3
第三节 安装 Gnome	6.4
第四节 安装 Mate	6.5
第五节 安装 Xfce	6.6
第六节 安装 Cinnamon	6.7
第七节 安装 Lumina	6.8
第八节 root 登录桌面	6.9
第九节 主题与美化	6.10
第十节 远程桌面管理	6.11
第十一节 安装 Wayland (可选)	6.12

第五章 输入法及常用软件

第一节 Fcitx 输入法框架	7.1
第二节 Ibus 输入法框架	7.2
第三节 五笔输入法	7.3
第四节 Firefox 与 Chromium 安装	7.4
第五节 Linux 兼容层	7.5
第六节 安装 金山 WPS	7.6
第七节 安装 QQ	7.7
第八节 更换字体	7.8
第九节 wine	7.9
第十节 压缩与解压	7.10

第六章 文件系统与磁盘管理

第一节 UFS	8.1
第二节 ZFS	8.2
第三节 磁盘扩容	8.3
第四节 NTFS 的挂载	8.4
第五节 SWAP 交换分区的设置	8.5
第六节 Ext 2/3/4 等文件系统	8.6

第七章 VPN与代理

第一节 HTTP 代理	9.1
第二节 V2ray	9.2
第三节 Clash	9.3
第四节 OpenVPN	9.4
第五节 Wireguard	9.5

第八章 用户与权限

第一节 sudo	10.1
第二节 添加用户	10.2
第三节 用户组	10.3
第四节 用户权限	10.4

第九章 Jail

第一节 jail 与 docker 的比较	11.1
第二节 jail 相关术语	11.2
第三节 jail 配置	11.3
第四节 jail 更新	11.4
第五节 使用 ezjail 管理 jail	11.5

第十章 虚拟化

第一节 虚拟化简介	12. 1
第二节 安装 Virtual Box	12. 2
第三节 安装 bhyve	12. 3
第四节 使用 cbsd 管理 bhyve	12. 4
第五节 使用 bhyve 安装 Windows 11	12. 5
第六节 安装 XEN	12. 6
第七节 使用 XEN 安装 Windows 11	12. 7

第十一章 更新与升级 FreeBSD

第一节 通过 freebsd-update 更新	13.1
第二节 通过源代码更新	13.2
第三节 批量部署	13.3

第十二章 GEOM 存储框架

第一节 概述	14. 1
第二节 RAID 0	14. 2
第三节 RAID 1	14. 3
第四节 RAID 3	14. 4
第五节 软 RAID 配置	14. 5
第六节 GEOM Gate Network	14. 6
第七节 磁盘装置标签	14. 7
第八节 UFS Journaling 与 GEOM	14. 8
第九节 ZFS 磁盘加解密	14. 9

第十三章 DTrace

第一节 概述	15.1
第二节 开启 DTrace	15.2
第三节 使用 DTrace	15.3

第十四章 网络管理

第一节 PPP 拨号	16. 1
第二节 WIFI	16. 2
第三节 USB RNDIS (USB 网络共享)	16. 3
第四节 蓝牙	16. 4
第五节 IPv6	16. 5
第六节 CARP	16. 6
第七节 VLAN	16. 7
第八节 TCP BBR	16. 8

第十五章 FreeBSD 防火墙

第一节 网络参数配置命令	17.1
第二节 PF	17.2
第三节 IPFW	17.3
第四节 IPFILTER (IPF)	17.4

第十六章 服务器

第一节 FTP 服务器	18. 1
第二节 DHCP 服务器	18. 2
第三节 Node.js 相关	18. 3
第四节 DNS 服务器	18. 4
第五节 NIS 服务器	18. 5
第六节 Postfix 服务器	18. 6
第七节 Samba 服务器	18. 7
第八节 NFS 服务器	18. 8
第九节 iSCSI	18. 9
第十节 Webmin	18. 10
第十一节 rsync 同步服务	18. 11
第十二节 时间服务	18. 12
第十三节 Wildfly	18. 13

第十七章 网络服务器

第一节 Apache	19.1
第二节 Nginx	19.2
第三节 PHP 8.X	19.3
第四节 MySQL 5.X	19.4
第五节 MySQL 8.X	19.5
第六节 Typecho	19.6
第七节 SSL 配置	19.7
第八节 PostgreSQL 与 pgAdmin4	19.8

第十八章 树莓派与嵌入式

第一节 树莓派简介	20. 1
第二节 系统安装	20. 2
第三节 使用配置	20. 3
第四节 USB 网卡与 WIFI	20. 4
第五节 RISC-V	20. 5

第十九章 文学故事

第一节 我与 FreeBSD 的故事	21. 1
第二节 FreeBSD 与猫 ——选择1%的生活	21. 2
第三节 Linux 与苦难哲学	21. 3
第四节 从一个想法看 FreeBSD 是商业化还是学院派	21. 4
第五节 Linux 社区已经成为了一个肮脏的泥潭	21. 5
第六节 Linux 败局已定——驳 FreeBSD 大败局	21. 6

第二十章 娱乐与教育

第一节 游戏	22. 1
第二节 音视频播放器	22. 2
第三节 音视频剪辑	22. 3
第四节 教育	22. 4
第五节 科研与专业工具	22. 5

第二十一章 内核

第一节 获取 FreeBSD 内核源码	23. 1
第二节 修改内核源码	23. 2
第三节 编译内核	23. 3
第四节 内核分析	23. 4

第二十二章 编程与开发

第一节 如何报告 Bug	24. 1
第二节 如何提交一个软件包	24. 2
第三节 如何参与 FreeBSD 协作	24. 3
第四节 C/C++ 环境的配置	24. 4
第五节 Java 环境的配置	24. 5
第六节 QT 环境的配置	24. 6
第七节 Python 与 VScode	24. 7
第八节 Rust/Go 环境的配置	24. 8
第九节 Csh 与其他 Shell	24. 9
第十节 通过 IDA 7 调试 FreeBSD	24. 10
第十一节 Git	24. 11

第二十三章 引导与恢复

第一节 恢复模式与密码重置	25. 1
第二节 FreeBSD 多硬盘 EFI 引导统一	25. 2
第三节 FreeBSD 中文 TTY 控制台	25. 3
第四节 引导界面	25. 4
第五节 Grub 及其他引导	25. 5

第二十四章 FreeBSD 特色

第一节 BSD INIT 管理服务	26. 1
第二节 FreeBSD 目录结构	26. 2
第三节 bsdinstall 与 bsdconfig	26. 3
第四节 禁用 Sendmail	26. 4
第五节 利用脚本自动生成 BSDlibc 库文本	26. 5
第六节 BSD 风格的 make/grep/sed/awk	26. 6

第二十五章 系统设计与分析

第一节 FreeBSD 设计概要	27.1
第二节 内核	27.2
第三节 进程	27.3
第四节 内存管理	27.4
第五节 安全	27.5
第六节 I/O 系统	27.6

第二十六章 OpenBSD

第〇节 概述	28. 1
第一节 安装	28. 2
第二节 配置	28. 3
第三节 换源	28. 4
第四节 包管理器	28. 5
第五节 桌面与其他软件	28. 6

第二十七章 NetBSD

第〇节 概述	29. 1
第一节 安装与配置	29. 2
第二节 换源与包管理器	29. 3
第三节 桌面与其他软件	29. 4

第二十八章 DragonFlyBSD

第〇节 概述	30. 1
第一节 安装与配置	30. 2

第二十九章 桌面高级进阶

第〇节 窗口管理器与桌面的区别与联系	31. 1
第一节 安装 i3wm	31. 2
第二节 安装 CDE	31. 3
第三节 安装 Awesome	31. 4
第三节 安装 FVWM	31. 5

FreeBSD 从入门到跑路

FreeBSD 中文社区 (CFC)

概述

[项目预览——如果你认为拼音显得奇怪，那么你应该点击此处。](#)

请注意，PDF 等电子书版本仅供参考，且存在这样或那样的问题，应该以在线版本为主。

本书定位

我们的目标并非是 Handbook 的翻译，而是编写一本类似于《鸟哥的 Linux 私房菜：基础学习篇》+《鸟哥的 Linux 私房菜：服务器架设篇》二合一的基于 FreeBSD 的教程。也就说我们是 Handbook 的超集。

编辑指南概要

我们欢迎所有支持 FreeBSD 的人进行编写，并会将您添加到贡献者名单当中。

[详细的编辑指南，点击此处。](#)

前言

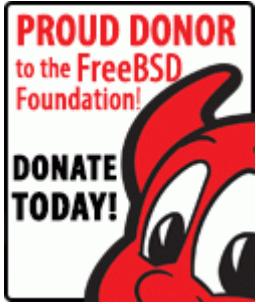
FreeBSD 从入门到跑路

本书诞生于 2021 年 12 月 19 日。编写目的是为了促进 FreeBSD 的中国化与世界化。编写内容为 FreeBSD 的基础与高阶知识。对于章节安排，如果你有一定的 UNIX 基础可以跳过第一章，如果你对 FreeBSD 有一定认识，欢迎你加入我们一起编写本书，贡献自己的力量。

内容提要

本书是由道长发起，并由 FreeBSD QQ 群 905149943 的一些群成员参与编写的《FreeBSD 从入门到跑路》。我们尝试从 0 开始，带领普通人走进 FreeBSD 世界，充分参考了 FreeBSD Handbook，构建了一个完整、科学的目录体系。本书不是一个教程的大杂烩亦或者是大集合，而是为了构建一个自成体系的一本开源书籍。全书共分三十章，既强调了学习 FreeBSD 的必要基础也提供了内核设计与实现等专业性较强的教程。本书可作为高等学校“FreeBSD 操作系统”课程的本科生教材，同时也适合相关专业研究生或计算机技术人员参考阅读。

开源维护与捐赠



[点此捐赠 FreeBSD 基金会。](#)

为了能够更好地维护本书，我们采用了 Gitbook 平台来进行协作。对于无法直接从 GitBook 导出为 PDF 的问题（我们提供了 PDF 的参考版本于 release），我们深感抱歉，我们目前没有财力为其提供所需要的企业版本（15 刀一个月）。如果您想为我们提供捐助，请加入我们的 [TG 群](#) 或者 QQ 群 905149943。 如果您也想参与编写，具体请参考 [WIKI](#)，关于贡献者名单请参考 [第一章 第九节](#)。

捐赠者：

【FreeBSD 2022 捐赠名单】

<https://docs.qq.com/doc/DSXZ1Q1J0enRzUkp4>

激励计划

【FreeBSD 中文社区 2022 教程 激励计划】

<https://docs.qq.com/doc/DSUJsUFBHTnVWQmtS>

意见反馈

由于编写者水平所限，书中缺点和谬误之处自不可免，希望同志们随时提出批评意见，以便修正。您可以利用 Github 的各种交互功能与我们联系：提交 Issue、Pull request 或者加入 QQ 群或 TG 群直接联系（yklaxds AT gmail DOT com）等。

TODO / Wishlist

后续还有很多需要完善的工作，包括不限于：

- FreeBSD 14 统一 shell 为 sh，教程需要针对其进行统一
- 整理和上传配置文件和环境
- 对教程的格式目录进行优化调整
- 积极对外宣传并寻求正式出版

许可证

本书采用 [BSD-3-Clause License](#) 许可证开源。我们在编写过程吸收了一些现有的研究成果，在此表示感谢。引用本书内容时，请务必留下我们的原地址——<https://book.freebsdcn.org> 及署名——FreeBSD 中文社区（CFC）。

其他

[FreeBSD Handbook 2022 中文翻译项目](#)

微信公众号：freebsdzh（扫码关注）



关于

FreeBSD 中文社区的愿景

我们成立于 2018年3月17日，由贴吧——FreeBSD 吧发展到了 QQ 群（主群 905149943），Telegram 群，乃至于微信群。

我们的成员具有非常大的广泛性和普遍性，能够代表绝大多数 FreeBSD 用户的平均水平：可能根本没有听说过何为 FreeBSD，但这并不影响我们的交流与沟通。也许有人觉得这是浪费时间，但是没有新生力量的培养，何来 FreeBSD 的明天呢？谁不知道新人可能有很多坏习惯呢。

同鲁迅先生说的那样，但愿每个人都是一束光，照亮 FreeBSD 在中国大陆地区前进的光荣的荆棘路。也希望，你可以加入我们，共同组成漫天星光亦或者是星星之火。

无穷的远方，无数的人们，都和我有关。

我曾无数次眺望远山，想要找到一汪清泉，天总是不随人愿，还是没有找到。

我是谁，我们是谁？这个问题永远也不会有结果。

我们选择 FreeBSD，是因为想选择一个清晰、明了、可靠、稳固的一个操作系统在工作上给我们带来收益以及在生活中给我们带来乐趣。当然 FreeBSD 还存在很多问题，有待大家积极发现、探讨、完善，社会在进步，技术在进步，热情丝毫不减在持续，未来越来越美好。

社区文档

【FreeBSD Q&A】

<https://docs.qq.com/doc/DSVhuc1hqZn1yR2Na>

教程征集计划——【FreeBSD 中文社区（CFC）2022 教程 激励计划】

<https://docs.qq.com/doc/DSUJsUFBHTnVWQmtS>

【FreeBSD 中文社区（CFC）2022 捐赠名单】

<https://docs.qq.com/doc/DSXZ1Q1J0enRzUkp4>

【2022 FreeBSD 中文社区（CFC）文档翻译须知】

<https://docs.qq.com/doc/DSUtxYmVwU29EdGVn>

【FreeBSD 中文社区（CFC）联络名单】

<https://docs.qq.com/doc/DSU1iWGh5ZU1XTFF0>

【FreeBSD 中文社区（CFC）发展规划】

<https://docs.qq.com/doc/DSUd5Q2tYS1ZWUmpj>

黑名单

见 FreeBSD 中文社区（CFC）黑名单。

第一节 FreeBSD FAQ

问题	答案
1、怎么在 VM 虚拟机上安装 FreeBSD（安装文字、视频、图解教程）？	①文字 https://book.freebsdcn.org/di-er-zan-zhuang-freebsd/di-san-jie-freebsd-13.0-zhuang-ji-yu-vmware-workstation-pro-16 ② https://www.bilibili.com/video/BV14i4y137 ③图解 https://book.freebsdcn.org/di-er-zan-zhuang-freebsd/di-ling-jie-tu-jie-an-zhi ④图解 https://handbook.freebsdcn.org/di-zhang-an-zhuang-freebsd/2.4.-kai-shi-an-zhi
2、怎么安装显卡驱动？	①VB虚拟机 https://book.freebsdcn.org/di-zhang-an-zhuang-freebsd/di-er-jie-freebsd-zhuang-ji-yu-virtual-box ②VM虚拟机 https://book.freebsdcn.org/di-er-zhang-zhuang-freebsd/di-san-jie-freebsd-13.0-zhuang-ji-yu-vmware-workstation-pro-16 ③机（英特尔核显、AMD\N卡） https://book.freebsdcn.org/di-er-zhang-zhuang-freebsd/di-jiu-jie-wu-li-ji-xia-ka-de-pei-zhi
3、支持Wayland？	不支持（gnome会卡在GDM、kde5会闪退）
4、怎么安装hald服务	现在不需要这个服务了。 https://book.freebsdcn.org/di-si-zhang-zhian-an-zhuang/di-er-jie-an-zhuang-kde
5、使用什么工具刻录FreeBSD？	rufus。不建议用其他软件。其他系统自己
6、硬件支持情况	去 https://bsd-hardware.info/?d=FreeBSD
7、选择ZFS 还是 UFS	内存大于等于 4G 用 ZFS，否则 UFS
8、安装了 KDE 5 桌面进不去，一直闪退	检查左下角是不是 X11。wayland 是进不去的。 机看不见左下角的去看问题 2

问题	答案
9、root 登录不进去 KDE5	https://book.freebsdcn.org/di-si-zhang-zmian-an-zhuang/di-liu-jie-root-deng-lu-zmian
10、FreeBSD 找不到 gcc wget.....	https://book.freebsdcn.org/di-yi-zhang-zjin-freebsd/di-ba-jie-linux-yong-hu-qianzhi-bei
11、FreeBSD 怎么安装软件	①pkg install xxxx ②cd /usr/ports/XXX && BATCH=yes install
12、FreeBSD 怎么安装 mysql/postgresql	① https://book.freebsdcn.org/di-shi-qi-zlwang-luo-fu-wu-qi/di-si-jie-mysql-5.7 ② https://book.freebsdcn.org/di-shi-qi-zlwang-luo-fu-wu-qi/di-wu-jie-mysql-8.0 ③ https://book.freebsdcn.org/di-shi-qi-zlwang-luo-fu-wu-qi/di-ba-jie-postgresql-pgadmin4
13、怎么升级系统	freebsd-update fetch install
14、什么是苦难哲学	https://book.freebsdcn.org/di-shi-jiu-zhwen-xue-gu-shi/di-san-jie-linux-yu-ku-nanxue
15、怎么安装输入法	这个问题和你的桌面、shell、用户都有关系。 细看书或在群内询问。 ①fcitx https://book.freebsdcn.org/di-wu-zhang-shfa-ji-chang-yong-ruan-jian/di-yi-jie-fcishu-ru-fa-kuang-jia ②ibus https://book.freebsdcn.org/di-wu-zhang-shfa-ji-chang-yong-ruan-jian/di-er-jie-ibusru-fa-kuang-jia ③五笔输入法 https://book.freebsdcn.org/di-wu-zhang-shfa-ji-chang-yong-ruan-jian/di-san-jie-qishu-ru-fa-kuang-jia

问题	答案
16、怎么配置 WIFI?	先查去 https://bsd-hardware.info/?d=FreeB 查询看支不支持；如果安装的时候不支持，那就是不支持。AX200/201之类的目前在测试。预计13.1会支持。支持的话看 https://book.freebsdcn.org/di-si-zhang-wang-luo-guan-li/di-er-jie-
17、怎么启动/关闭/重启服务	service xxx start、 service xxx stop 、 service xxx restart
18、关于FreeBSD 的书	看 QQ 群 905149943 文件
19、如何使用 Linux软件	建议构建一个ubuntu 20.04 兼容层（实质上是jail）① https://book.freebsdcn.org/di-zhang-shu-su-fa-ji-chang-yong-ruan-jian/di-jie-linux-jian-rong-ceng ② https://handbook.freebsdcn.org/di-10-zhi-linux-er-jin-zhi-jian-rong-ceng/10.4.-syong-debootstrap8-gou-jian-debian-ubuntu-ben-xi-tong ③ https://wiki.freebsd.org/LinuxJails ④ https://wiki.freebsd.org/LinuxApps
20、怎么安装QQ?	https://book.freebsdcn.org/di-wu-zhang-shu-ji-chang-yong-ruan-jian/di-qi-jie-a-zhuang-qq
21、vmware 虚拟机屏幕缩放不了	去下载群文件的 vmware https://book.freebsdcn.org/di-er-zhang-zhuang-freebsd/di-san-jie-freebsd-13.0-zhuang-ji-yu-vmware-workstation-pro-1
22、有没有中文翻译的、最新的 handbook?	https://handbook.freebsdcn.org/
23、在哪下载 FreeBSD	https://freebsd.org

问题	答案
24、我该下载哪个镜像？	选择最新的 release 版本，虚拟机选择iso， 选择 img 结尾的，别选 bootonly。 https://handbook.freebsdcn.org/di-2-zhang-zhuang-freebsd/2.3.-an-zhuang-qian-de-zhi-bei-gong-zuo
25、FreeBSD release current stable 有什么区别？	① https://book.freebsdcn.org/di-er-zhang-zhuang-freebsd/di-yi-jie-san-zhong-xu-ni-yu-freebsd-ban-ben-bi-jiao ② https://handbook.freebsdcn.org/di-24-zheng-xin-yu-sheng-ji-freebsd/24.4.-zhui-zai-kai-fa-fen-zhi
26、网络不通，ping不通、unkown host	输入ifconfig看看有没有网卡，没有去 https://bsd-hardware.info/?d=FreeBSD 查一 不是支持（虚拟机不用看）。支持的话一般问是有 DNS。① https://book.freebsdcn.org/di-zhang-an-zhuang-freebsd/di-liu-jie-ee-ying-ji-wang-luo-pei-zhi ② https://www.bilibili.com/video/BV14i4y1zw
27、root 登录 ssh ?	①视频 https://www.bilibili.com/video/BV14i4y137 ②文字 https://book.freebsdcn.org/di-er-zan-zhuang-freebsd/di-qi-jie-chang-yong-rong-jian-yu-ssh-pei-zhi
28、怎么看 FreeBSD 有没有一个软件？	①pkg -o search xxx ②打开搜索引擎（谷歌bing）搜关键词 freebsd xxx ports 一般有这个站 https://www.freshports.org/ 直接去里面是了，建议加个书签
29、支不支持 docker	暂不支持。包破损了，有能力者可修复 https://wiki.freebsd.org/Docker
30、FreeBSD 的 C 库文档在哪？	群文件有手册

问题	答案
31、怎么捐款?	①FreeBSD 基金会： https://freebsdfoundation.org/donate/ (如为支付问题无法完成, 我们可以代为捐献) ②们: 联系道长 (@道长)
32、tg群是多少?	@freebsdba
33、怎样成为群管理?	积极参与社区活动
34、怎么使用手机共享网络给 FreeBSD?	https://book.freebsdcn.org/di-shi-si-zha-wang-luo-guan-li/di-san-jie-usb-rndis-u-wang-luo-gong-xiang
35、目前我可以参加哪些社区项目?	①完善教程 https://github.com/FreeBSD-Ask 译文档 handbook 等, 请@道长
36、这个列表上没有我的问题?	先看书: https://book.freebsdcn.org 看了解了再去问
37、怎么换源?	https://book.freebsdcn.org/di-san-zhang-1-jian-yuan-ji-bao-guan-li-qi/di-er-jie-fre-huan-yuan-fang-shi
38、我有一个腾讯云服务器, 是轻量云, 怎么安装 FreeBSD?	https://book.freebsdcn.org/di-er-zhang-zhuang-freebsd/di-si-jie-teng-xun-yun-qiliang-yun-ji-qi-ta-fu-wu-qi-dd-an-zhuai-freebsd
39、用户组信息放在那个文件里?	https://handbook.freebsdcn.org/di-3-zha-freebsd-ji-chu/3.3.-yong-hu-he-ji-ben-zhu-hu-guan-li
40、在安装中应该选用哪个镜像站?	请不要问这类问题, 那是因为你的错误操作导致不要选用带有 bootonly 字样的镜像文件, 除非知道自己在干什么。 https://book.freebsdcn.org/di-er-zhang-zhuang-freebsd/di-ba-jie-wu-li-ji-an-zhu-yu-ying-jian-xuan-pei

问题	答案
41、FreeBSD 中文社区为什么要推广和发展FreeBSD，有何目的？	我们致力于： 使用 FreeBSD 发展中国的计算 业，提高广大人民群众的计算机水平，使中国自 机水平位列世界前沿。增进人们的精神文化建设 力中国信息化、现代化建设；增强以大学生为 的群体其对开源项目的参与度，促进开源事业在 乃至于亚洲地区的发展；
42、root mount waiting for usbus 0	可以把 <code>hw.usb.no_boot_wait=1</code> 追加 到 <code>/boot/loader.conf</code> 里，就不会出现这种信息
43、DVD、ISO、 IMG、BOOonly都有 什么区别？	https://handbook.freebsdcn.org/di-2-zhang-zhuang-freebsd/2.3.-an-zhuang-qian-de-zhi-bei-gong-zuo 虚拟机选择iso，（无用还会造 错误，比如使用其自带软件安装的gnome无法启 物理机选择 img 结尾的。无论何时都不应该 bootonly，除非云主机安装。

第二节 FreeBSD 中文社区 (CFC) 发展规划

短期规划（2022–2023）【进行中……】：

- 主力建设微信群（①和②……），扩充 QQ 群，以 Telegram 群为辅；
- 完成《FreeBSD 从入门到跑路》；
- 完成 FreeBSD 手册简体中文翻译；
- 继续发展微信公众号——FreeBSD 中文；
- 继续募集捐款（目标 还需约 ¥800）；
- 建设维护 freebsdcn.org
- 初步形成 FreeBSD 中文社区核心团队以及管理组；
- 捐赠 FreeBSD 基金会；

中期规划（2023-2025）：

- 根据《FreeBSD 从入门到跑路》录制视频上传 Bilibili;
- 在国内大学中广泛推广宣传 FreeBSD，并带领其形成校内社团/开源组织；
- 使用虚拟 VUP 宣传 FreeBSD；
- 维护审阅 FreeBSD 手册与《FreeBSD 从入门到跑路》；
- 完全形成 FreeBSD 中文社区核心团队、翻译组、以及管理组等；
- 继续募集捐款（目标 ¥3000）；
- 培养人才——集中授课；
- 孵化项目，成立公司；
- 为 FreeBSD 项目贡献文档；
- 建立 FreeBSD 中文社区自己的项目——使用 FreeBSD；
- 完成 FreeBSD 其他手册的翻译；
- 捐赠 FreeBSD 基金会；

长期规划（2025-2030）：

- 出版 FreeBSD 手册与《FreeBSD 从入门到跑路》；
- 募集捐款（目标 ¥100000）；
- 继续为 FreeBSD 项目贡献源代码；
- 有成员成为 FreeBSD 项目 Committer；
- 捐赠 FreeBSD 基金会；
- 解决镜像站问题；

最高纲领（2030-2040）：

- 有成员进入 FreeBSD 项目 核心团队；
- 使用 FreeBSD 发展中国的计算机事业，提高广大人民群众的计算机水平，使中国的计算机水平位列世界前沿。增进人们的精神文化建设，助力中国信息化、现代化建设；
- 增强以大学生为主体的群体其对开源项目的参与度，促进开源事业在中国乃至于亚洲地区的发展；
- 成立 FreeBSD 中国基金会或全国性社团组织；
- 募集捐款；
- 捐赠 FreeBSD 基金会；

第三节 FreeBSD 的不足之处

- FreeBSD 没有为用户提供一个带 GUI 的基本系统，甚至显卡驱动都需要自己通过 port 编译安装；
- FreeBSD 的驱动很差，直到最近也不能完美地支持 WIFI 6 的网卡，比如 AX210；
- FreeBSD 的开发者非常少，这意味着你的 bug 可能很久都无法得到解决，软件包也不能像 ARCH 那样时刻保持最新版；
- FreeBSD 的资料相对较少，中文资料更少，简体中文资料几乎为 0；
- 由于 systemd 不兼容 Linux 以外的操作系统，导致很多软件比如 NetworkManager 无法移植，桌面环境的组件也无法完善；
- FreeBSD 的菊苣们比 Linux 的还要更加高傲，他们不在意你到底会不会换源会不会设置代理，需不需要境内的官方镜像站；
- 由于 FreeBSD 项目的基本目标和设计问题，FreeBSD 基本系统不包含一般 Linux 中常用的一些软件和命令，比如没有 `lspci` , `free` 。有些可以自己安装，有些则不行；
- FreeBSD 的两个文件系统都只能扩大不能缩小；
- FreeBSD 缺乏上层应用软件设计，即使底层有类似 docker 的技术也没能发展起来；

第一节 什么是 UNIX

从前是一个操作系统。最后由 C 语言改写产生。——AT&T 公司

现在是一个 标准规范和商业商标。更是一种哲学思想，软件工程原则。

查询网址：<http://www.opengroup.org/openbrand/register>

我们可以知道认证 UNIX 需要：

1. 符合单一 UNIX 规范
2. 交钱认证

The screenshot shows the homepage of The Open Group's UNIX Certified Products register. At the top, there's a dark header bar with the 'THE Open GROUP' logo on the left and navigation links for 'UNIX®', 'Open Brand', 'Product Standard', 'Company', and 'Recent' on the right. Below the header is a large blue banner with the text 'UNIX® Certified Products'. Underneath the banner, there's a logo for 'The Open Brand Register' and the text 'The Open Group official register of UNIX Certified Products'. A 'Learn More' button is visible. At the bottom of the page, there's a section titled 'For details of the certification click the product links' followed by a list of certified products:

- Apple Inc.: macOS version 11.0 Big Sur on Apple silicon-based Mac computers
- Apple Inc.: macOS version 11.0 Big Sur on Intel-based Mac computers
- IBM Corporation: AIX version 7, at 7.2 TL5 (or later) on systems using CHRP system architecture with POWER™ processors
- IBM Corporation: AIX version 7, at either 7.1 TL5 (or later) or 7.2 TL2 (or later) on systems using CHRP system architecture with POWER™ processors

At the very bottom, there's a footer with the URL 'https://www.opengroup.org' and the text 'Cemprus LLC: DNCP Series running FTX Release 3'.

以下为详细说明：

1. Unix 的前身

1964 年麻省理工学院推出的 CTSS（兼容分时系统），是当时最有创造性的操作系统，有了 CTSS 这种高效的操作系统，麻省理工学院的研究人员决定做一个更好的版本。他们开始设计 Multics 系统。Multics 意思是多路复用信息和计算服务。

Multics 意图创造强悍的新软件和比肩 IBM 7094 功能更丰富的新硬件，麻省理工学院邀请了两家公司来帮忙。美国通用电气公司负责设计及生产有全新硬件特性、能更好地支撑分时及多用户体系的计算机，贝尔实验室在计算机发展早期就开发了自己的操作系统，因此麻省理工邀请了贝尔实验室共同开发 Multics。

最终 Multics 的开发陷入了困境，Multics 设计了大量的程序及功能，经常塞入很多不同的东西进去，导致系统过于复杂。1969 年，由于在贝尔实验室看来作为一套信息处理工具，它已经无法为实验室提供计算服务的目标，它的设计太昂贵了，于是在同年 4 月，贝尔实验室退出 Multics 项目，只剩麻省理工和美国通用电器公司继续开发。

2. “UNICS”

贝尔实验室退出 Multics 开发项目后，项目组成员肯·汤普逊（Kenneth Lane Thompson）找到一台 DEC PDP-7 型计算机，这台计算机性能不算强大，只有 4KB 内存，但是图形界面比较美观，汤普逊用他写了个太空游戏（Space Travel），PDP-7 有个问题就是磁盘转速远远低于计算机的读写速度，为了解决这个问题，汤普逊写了磁盘调度算法来提高磁盘总吞吐量。

如何测试这个新的算法？需要往磁盘上装载数据，汤普逊需要写一个批量写数据的程序。

他需要写三个程序，每周写一个：创建代码的编辑器，将代码转换为 PDP-7 能运行的机器语言汇编器，再加“内核的外层——操作系统就完成了”。

新的 PDP-7 操作系统编写没多时，汤普逊和几个同事讨论，当时新系统还没有名字，当时它被命名为“UNICS”，UNICS 最后改名为 UNIX，这个名字更加方便记忆。

第二节 什么是 Unix-like

Unix-like 即类 Unix，也即一切基于 UNIX 的操作系统，基本遵守 POSIX 规范，而没有获得第一节中所说的 UNIX 的认证。

也就是说，除了 Windows，基本上世界上大多数操作系统都被叫做 Unix-like，其中就包括 Linux 和 FreeBSD。

第三节 什么是 Linux

狭义 Linux 是内核；

Linux 与 GNU/Linux; GNU项目 1984; Linux kernel 1990;

GNU 软件 + Linux 内核 = GNU/Linux 发行版 = Ubuntu、RHEL、Deepin、OpenSUSE……

Gentoo (stage1) 或 LFS;

第四节 FreeBSD 与其他操作系统

什么是 FreeBSD?

FreeBSD 不是 Linux，不是国产操作系统，不兼容 Systemd，不能吃鸡，亦不是 UNIX。目前在 BSD 系中，FreeBSD 的用户是最多的。一些 Linux 下的软件基本上在 FreeBSD 中都能够被找到，即使找不到的也可以通过 CentOS 兼容层运行，你也可以自己通过 debootstrap 构建一个 debian 或者 ubuntu 的 / 系统。

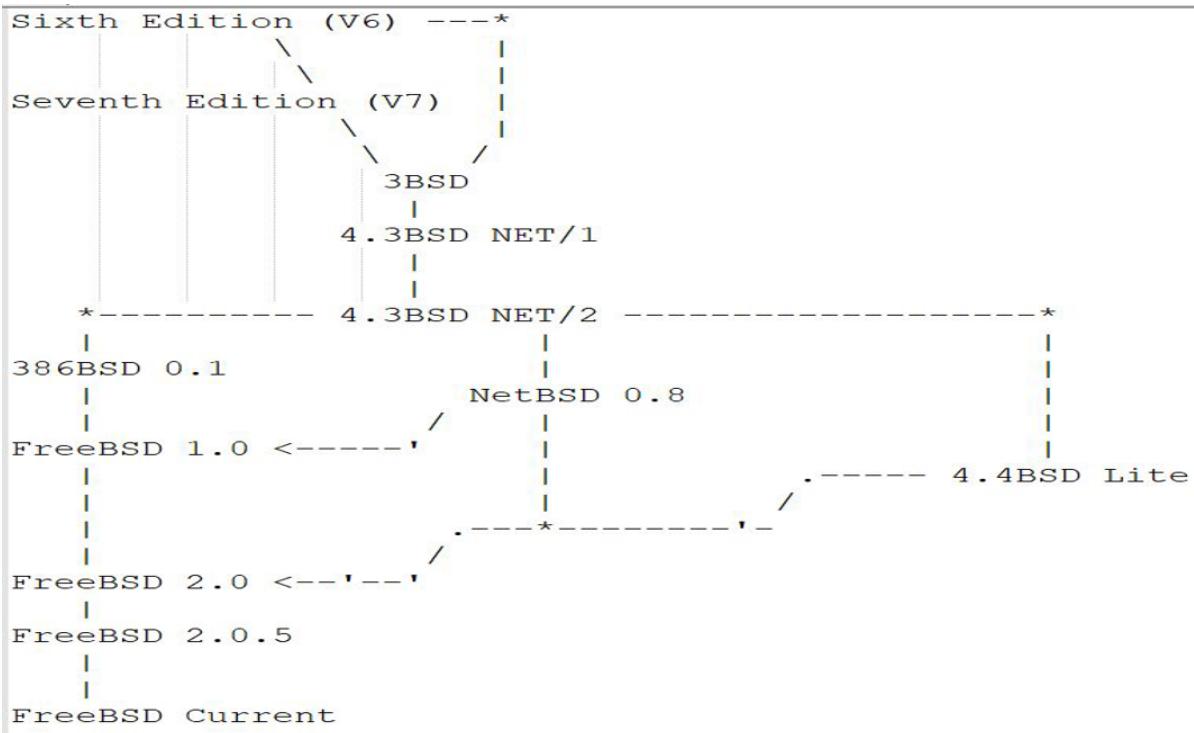


FreeBSD 不是Linux，亦不是UNIX。

UNIX -> Networking Release 1 -> Networking Release 2 -> 386BSD -> FreeBSD 1.0

386BSD -> 诉讼（1991-1994） -> 4.4 BSD-Lite -> FreeBSD 2.0

Linus “I have never even checked 386BSD out; when I started on Linux it was available”



FreeBSD or Others

1. Linux

首先大概许多人是从 Linux 跑过来的，这样说我也没什么统计依据，不过姑且这样说罢。如果你发现在哪本书是举例提到 FreeBSD 是一种 Linux 发行版，那么我个人是不建议你继续看下去的，这属于误人子弟，我也曾在某些慕课网站上看到过类似行为。

严格来说 Linux 是指 Linux kernel，只是个内核而非操作系统，而 FreeBSD 是个操作系统。FreeBSD 采用 BSD 授权许可（见 https://www.freebsd.org/zh_CN/copyright/freebsd-license.html ）。FreeBSD 驱动方面一直是个大 Bug，不如 Linux 。

1. macOS & iOS

macOS & iOS 在一定程度上来说，都基于 FreeBSD 。可见 FreeBSD 的 GUI 并不是搞不好，只是 Xorg 和开发方向有问题。

首先 macOS 和 iOS 某种程度上都基于 FreeBSD 。但是这时候就要说易用性了，FreeBSD 和 Linux 还都是那套 Xorg ，很明显不行。但是本着你行你上的观点我也上不去……图形界面才是第一 x3 。

到底是苹果成就了 macOS、iOS 还是反过来 二者成就了苹果呢？举例来说，买 Mac 装 Windows 当然这是个人喜好，没有任何值得批评的地方。假设 iOS 预装 Android （这么举例可能不恰当），相当一部分纯果粉应该是接受不了的。

生态环境。这个见 Windows Phone 。那么为什么选择 Apple 就不是 1% 的生活了？成功的商业化运作起着很大的用处。就像在这个贴吧里总有人看我不爽但又骂不过我一样，逞得口舌之利都不如我。FreeBSD 在大陆镜像站都没有，甚至因为 free 这个英文单词连官网都被电信屏蔽过。这个生态环境相比可知了。而且现在 UNIX 认证很宽容，所谓什么血统那是扯淡。好不好用自己心里没数吗？资本家之所以是资本家就在于产出再投入。对于这里而言，苹果的软件多就是因为用的人多。这个初期是怎么积累的？ FreeBSD 一场官司，初期就没有得到很好的发展，不然就没有 Linux 了，这话是 linus 说的。

国民素质有待提高。这个不是看不起嘲讽，这是客观事实。很多大学生甚至不知道什么是 Android，还有人说万物基于 MIUI 。这和术业有专攻这句话已经完全无关了。当然不是说用水果就是素质低，这么理解的人语文有毛病。

水果摆脱了开源界所谓的苦难哲学。

1. Microsoft Windows

微软非常重视用户体验，而一些社区可能完全忽视了这一点。直接的结果就是需要自己动手解决的地方略多。有人认为 Windows 简单因为都是图形化界面。事实上这是一种非常错误的说法，Windows 非常复杂。举例来说，你精通注册表否？知道每个选项什么意思吗？

至于安全性，很多人认为 UNIX-like 不需要杀毒软件，但是事实上这种观点是不正确的，当你发现自己中毒的时候，已经成为了病毒的培养基。但是目前来说，FreeBSD 远比 Windows 安全。

至于游戏什么的，已知 Steam 运行正常，运行 Minecraft 这种 java 软件也没毛病。

第五节 为什么要使用 FreeBSD

选择 FreeBSD 的一般原因

从道家来讲，你爱选不选，太长不看，不用？左转 Linux，Windows 吧，不谢。

从佛教来说，因为缘分。万物缘起性空，我们有缘相聚，又会者定离。万般诸相皆如此。

从基督教来讲，这是主的指引。就像出埃及记一样，你看上去是自己的选择，实在上都是主的安排。

从辩证唯物主义来讲，是因为联系。FreeBSD 是 UNIX 直接后裔，而 Linux 只是仿制品，而很多协议又离不开 UNIX，所以你注定了要来到这里。

按照我个人观点而言，追求软件的稳定和新，既要有二进制源，又要能编译安装。除了 FreeBSD 之外我找不到 Linux 系统。

BSD 三则授权协议：并允许自由分发。GPL 与 BSD 协议，究竟何者是真正的自由？

远离碎片化的 Linux 发行版，使得选择困难症用户免受痛苦。

BSD 是一个完整的 OS，而不是内核。内核和基本系统作为一个项目来整体维护。

Linux 社区已经成为一个肮脏的泥潭，无论是内核开发还是用户群组。——见 第十九章第五节

选择 FreeBSD 的技术性原因

系统配置文件与第三方软件配置文件分离。/etc 与 /usr/local/etc 等——见 第二十四章第二节。

文档齐全，所有涉及一般性的问题 Handbook 手册都有记述。

安全漏洞相比于 linux 较少。

接近 2.5 年的版本发布周期，5 年的维护周期赋予了 FreeBSD 稳定性。

通过 BSD 的 Ports 可以编译安装软件，进行自由配置。

ZFS 文件系统可以被配置为 root 分区。ZFS 被誉为最强大的文件系统。

Jail 与 byhve 虚拟化，不必配置底层虚拟化，节约系统资源。

传统的 BSD INIT 引导，使你免受 systemd 迫害。

DTrace 框架与 GEOM 存储框架。

Linux CentOS 二进制兼容层，可运行 Linux 软件，只要其支持 CentOS。且软件运行速度快于 Linux。

安全事件审计。

第六节 所谓开源哲学

我想说的，正如庄子在《南华经》中所言：“吾生也有涯，而知也无涯。以有涯随无涯，殆已！”所以我不强调所谓的终身学习观念。人能弘道，非道弘人。知识都是自己学的，即使可以像恐怖如斯的渡劫强者向他人醍醐灌顶传输功法，所受之人亦不能穷尽所有道法。人的生命短暂，整个人类的生命对于宇宙来说又何其短暂！不明白这个道理，永远只能被剥削。

如果困难是财富，那么在 FreeBSD 就是这样一个充满财富的合集。如果苦难是一种哲学，那么这种哲学的别名叫做 FreeBSD 哲学。

提到开源二字，首先人们会想到 GNU 计划，其次比如 FreeBSD 此类计划。

有很多人讽刺 Microsoft，说 Microsoft Windows 上运行的 IDE 垃圾，隐藏了引擎盖下的细节，一按下去“预处理，编译，汇编，链接”四步就都完成了。此时便会有人出来，说我们用 Linux 吧，再安个 GCC，用 VIM 写代码，用 GDB 调试。用 Windows 多垃圾啊，你入 Linux 啊！

从此，误入尘网中，一去三十年。从 Ubuntu 到 Gentoo，发行版换了几个，却没有达成初心。

我们都知道，先有键盘，后有鼠标，现在，你省下了买鼠标的钱。把 Xorg 删掉，你说桌面占用内存；把 Windows 删掉，你说节约硬盘空间。记住了 VIM 几千个指令，你发现，还是记事本好用。

苦难由此而生。

我不觉得在 TTY 下加载出 Bilibili 的 HTML 播放器有任何值得称赞的原因。也看不到使用 xfce 桌面系统，它哪里优越于 macOS 或者 Windows 的图形界面。但是一些人仍然一如既往的展现出自己无处不在的优越性。

开源不是乌托邦，意味着 Free 。这意味着免除一切责任。只能依靠自己。

我想不出来，为什么我们走进了青铜时代，又要回归石器时代。你说为了开发效率，为了节约硬件成本，为了节约正版软件费用。我说，现在的设备，即使是嵌入式也不再用汇编进行开发，而使用 C 语言；现在的笔记本，内存标配提升到了 8G ；而正版与否，大家心里都清楚，对于开源，也不是随意商用。

自由，轻量化，安全与稳定性似乎是开源的代名词。其实不然，自由并不是给你代码让你自己修改并编译，花上几个小时。你说自己编译的软件运行效率高，却拿不出任何论文作为证据。

开源哲学，号称互帮互助。著名 IRC 频道中，我很遗憾，没有看到这一点。对于国内论坛，社区，各种乱七八糟的 log 贴上去，你只能得到嘲讽，就像是多年的丑媳熬成了恶婆婆“什么 https，你懂 openssh 吗？你知道证书是什么东西吗？不知道你问个…”以此循环往复，我更是不必多说，还有某协会在为自己的前会长打广告。说到底，需要的不是知识，都是钱和肆意嘲讽他人的资本。

即使是对服务器，大部分人使用 CentOS，我看不出它哪里比 Scientific Linux 好。性质都是一样的。只有真正明白的人才会知道，盲目的从众，缺乏理性认识就深入一个东西，是多么的无知。

在图形界面盛行的今天，我们不应该开历史的倒车。也不能让 OSS only for server 。说 FreeBSD 不行的，可能没用过 iOS；说 Linux 垃圾的，可能没用过 Android 。

带着苦难哲学的人，犹如套在袋子里的人。

在有人把 FreeBSD 当作 Linux 的今天，本书的目的在于弘道。很多人抱怨，我们的传统文化在钢筋水泥中逐渐死亡。这当中有很多传统技艺功法失传是由于一些所谓封建的观点而造成的，比如传男不传女，教会徒弟饿死师父，概不外传等等。认为应该公开真本领，真本事，真技法。

真本领、真本事、真技法这三真，我叫他道法。无论是否简单，都是道法， $1+1=2$ 也是道法，如何证明 $1+1=2$ 也是道法。从这个方面来看，我们一切所知皆为道法。但是为何冠名以道，下章再议。

我认为道法不能轻传。

轻传并非指完全不传，而是指在条件的情况下完整的传承下去。一言以蔽之，在客观上免费获取知识的地方都是骗人的地方。在这里，比尔盖茨一定十分赞同这个观点。

举众所周知的例子来说，西游记中唐僧师徒五人前往西天取经。彼时东方无经文吗？还真没有。历经故意设置的九九劫数不只是他们的宿命，更因为道法不能轻传，一本经文通晓，已是难得的高僧，更妄论十本，百本。

在道教而言，不是所有人都能念经文的，因为没有道法，普通人不知道应该避讳哪些字，哪些是道士能念的，哪些居士不能念的，念时往何方向，掐何种手印，从哪到哪，该念几遍，何日禁忌。而这些只有

师父会告诉你，别人不会告诉你。

可以见得，佛道两家，都清醒的并做到了这一点：不轻传道法。并非单纯倡导宗教上的不轻传道法，而是说这两家的认识比较深刻，而且实行的较为正确。很多玄幻小说作家也认识到了不轻传道法这种观点。前辈将各种典藏都收于大山，留待有缘之人。原因有二，险地少人，非真有缘者不可来；取物磨难，非易得之物。北方有句谚语“听人劝，吃饱饭”，经验丰富的家里的老人会告诉你很多事情，不一定对，也不一定错，但一定的是能吃饱，不能吃好。诗人说路多歧路，歧路亡羊，不必告诉青年朋友们此路不通，尽可让他们头破血流，这才是青春，即使因此失去了生命。缘由就是你说了也没有人会去听，你说的的的确确是道法，也传了出去，但是产生什么效用了没有？并没有。唠唠叨叨是可以停止了，没有任何作用。

现代社会倡导知识分享，但是我可以看到核心期刊，专业学术论文没有人分享的。那么分享的究竟是什么东西？是华盛顿砍树这种假故事还是方便面是垃圾食品这种谣言？我所看到的慕课不过是把书上的内容念了一遍而已，书的质量差，这种慕课更差。这不是在于人才，国外一些大学的慕课水平如何？不错。为何？因为那根本不是慕课，就是将课堂录制了下来，仅此而已。人们看到免费的总要去占便宜，不知道被消费的是自己还有额外的机会成本即时间。因此我断言，免费的慕课是做不好的。即使课好，也不可能有多大的影响力。正如上文所言，你把道法传出去了，你就不管接收者，也管不着接收者了。这就是轻传道法的弊端所在，浪费你我时间金钱和感情。

如《理想国》一书所述，道法必有一种途径进行传承。如果传承不再，那也是必然规律，不可强求。即使费力的保留了下来，也定要当历史的吊车尾。

人人可为师，非人人可得道。一件事情是收费的，用金钱衡量了价值。对于普通人，恐怕没有有人说 Linux 比 Windows 好用。这里的普通人也包括没有接触过计算机教育的人。所以没人向一般用户群体推荐使用 Linux，除非他别有用心。商人可能市侩，但是也是为了道法的传承。如果物品免费，就不会再有人去做。由此也是版权专利的由来。使人失去了欲望是一件可怕的事情。现在的社区正面临这种困境。

很多专业书籍为了牟利不择手段，什么多少天精通 C++，一把年纪出书误人子弟。不是为了传道，而是为了牟利或自私的普及知识。这种书只能打击学习者的积极性。说是看来能够短时间普及文化，但是在长期看来营造了一个错误的环境和知识氛围，造成与他人更大的差距。

传道没有这么轻松，人人可以传道，但是传的道不一定是自己自以为可以传的道。

活在梦里，醒来却发现仍不知道什么是疼。未来的结局早已经注定，可仍要懦弱的挣扎偷生。

就像在物理机中运行的虚拟机中运行的虚拟机一样，总以为自己是真的。可悲的一生被设计好了，无论是国家机器把我们设置成热爱祖国热爱人民的学生，还是我们自己不愿意做圆上的一个切点。古今皆有之，把大器晚成作为自己一事无成的理由，来宽慰自己，勉励自己，并相信自己可以有所作为，经天纬地。仿佛很多人看破了，看透了，活的敞亮了，无拘无束了。向天一笑，“看的清澈又有何用，终归是自己为难自己。”可叹浮云长涨长消，潮水潮起潮落；可悲雾霾时有时无，而无路可走。

年轻的时候像茶水溢出了杯具，悲剧一事无成；时间长了终归不如洒掉全部。

不是他物磨平了我们的棱角，正是自己磨平了自己。物遇不平则鸣，君子不器，不愿意成为他人利用的对象，可是这种种却都是赤裸裸的血腥贪婪暴力，都改变了这些。天下大事绝非偶然，与其沉瀣一气不如寄情山水。

大器不成，天意难违。按照唯物辩证法的观点是不正确的。但是我却以为他是正确的，我并不会为某个主义牺牲自己，万一他是错的呢？我没有机会去验证，而那些验证的人得到的也只是个未知数，充满了不可知性。从上述观点看来，大器虽晚成，经天亦纬地。

第七节 其他 BSD 简介

OpenBSD 侧重于安全，号称是世界上最安全的操作系统（想起了德国 2014 年的电影 *我是谁：没有绝对安全的系统*）； WiFi 开发的很积极，已经支持了 intel AX200；但是软件包较少，较陈旧，比如 KDE 才 3.5；为了安全舍弃了 sudo 和 linux 兼容层，对于 N 卡的支持也很是问题；一般被用作防火墙的开发。

FreeBSD 是开发者最多用户最多、软件包最多的，有 ZFS 和 Linux 兼容层；

NetBSD 架构支持多，但普遍支持的不好，因为一个开发者同时维护好几个架构，开发者数量不足；

DragonFlyBSD（蜻蜓 BSD）自带 i915 显卡驱动，但是文档滞后，架构支持的也不多。

第八节 Linux 用户迁移指北

由于 GNU 开源运动的大规模开展，大部分人对于开源的理解被囿于 GPL 与 Linux，无法走进 FreeBSD 这片 BSD 世界，主要表现为某些自诩为 **开源社团** 的组织，其实际上的、具体的活动并没有超越 Linux 的基本边界。为此特撰写本文。

关于二者的对比在前文已经写过了，不再赘述，也毫无意义。再强调 FreeBSD 的优越性也是啰嗦的废话而已。与其强调 FreeBSD，不如走进 Linux，真实的 Linux。

以下言论可能某些人可能会感到冒犯，但这就是真实：

各大 GNU/LInux 发行版对比

Ubuntu

Ubuntu 是著名的内部错误发行版。有些人为此争辩“那是 Ubuntu 太谦虚了，他把不属于自己的报错也揽到自己身上”，但无可辩驳的是 Ubuntu 基于 Debian 的 SID 版本，本身稳定性是没有保证的。

Fedora

Fedora 俗称“地沟油”，是基于 RHEL 的上游系统，我更喜欢称其为小白鼠发行版，其发行的根本目的是为了测试 RHEL 系统的新设计和新架构，待稳定后迁移到 RHEL。稳定性可见一斑。

CentOS/Rocky Linux/RHEL

目前，CentOS 已经不再是以往的基于 RHEL 源代码构建的操作系统，而是 RHEL 的中游测试系统，和 Fedora 差不多了。其替代品五花八门，甚至还有取得了 UNIX 认证的所谓 欧拉系统。但是我认为 Rocky Linux 更加有前景。这些系统在服务器上被广泛部署，具体缺点就是以牺牲软件的“新”来换取“稳”，软件版本非常陈旧。

Debian

Debian 俗称“大便”。有个很奇怪的事情，设置了 root 密码就不会安装 sudo。Debian 的软件包也不甚更新。（此处仅指 stable）。

OpenSUSE

完整的 OpenSUSE 安装后系统非常的卡顿，据说是 btrfs 文件系统的某个特性，也许卡就是特性之一吧。OpenSUSE 俗称大蜥蜴。他所做的最搞笑的一件事是他的版本号，为了纪念英国作家道格拉斯·亚当斯在《银河系漫游指南》中写到的这个数字“42”，（被称作“生命、宇宙以及任何事情的终极答案 the answer to life, the universe and everything”），OpenSUSE 把版本号从 13 跳到了 42，然后又从 42 回到了 15。然后搞笑的一件事是，从 42 到 15 应该是升级的过程，但是 42 的版本号比 15 大，于是你到了 15 再升级就会再反向升级到 42。那么现在问题来了，到了 41 再升级是到 42 还是 43 呢？

Gentoo

Gentoo 俗称“元发行版”。一切软件都要通过 编译 的方式进行安装。其缺点也很明显，如果一个程序编译不过去就无法安装了，实际上这种软件非常多。有人会抬杠说 Gentoo 有二进制安装方式，但那也需要自己先本地构建，并没有统一的官方二进制源。一旦你一段时间不更新，Gentoo 会告诉你什么叫做 循环依赖。而且Gentoo 难以大规模部署，也难以在服务器上部署。另外 Gentoo 的 portage（包管理器）是python 语言编写的，这直接导致计算依赖的时间的延迟：在树莓派 4 上，安装 KDE 5 往往要计算几个小时……

简而言之，Gentoo 用自己的哲学捆绑了用户，简单问题复杂化，自己折磨自己；USE 过于复杂，对于一些常用软件，都经常出现循环依赖问题，破坏系统稳定性，软件安装升级卸载困难。

Deepin/UOS/中标麒麟

UOS 和 Deepin 的关系就好比 RHEL 之于 Fedora。本质上是一种东西。Deepin 系统似乎从不进行软件测试，直接就将更新包推送给用户，最直接结果就是他知道更新会导致系统崩溃也不撤回当次更新，而是在官网一个小角落里写个帖子和你讲怎么修复？这是令人迷惑的思路和解决方案。Deepin 这个系统仅仅是复制文件就会导致桌面卡死，无法理解他的系统是怎么做出来的，难道他自己的开发者不需要复制文件吗？UOS 个人桌面版就是让别人参与小白鼠测试，居然还需要注册，无话可说。

对于中标麒麟这类所谓国产系统无话可说。

Arch Linux/Manjaro

Arch Linux 俗称“邪教、洗发水”。这是我所见过的一个最不稳定的 Linux 发行版，也是个人计算机上用户人数最多的。我难以理解为什么有这么多人选择如此不稳定的一个操作系统。你安装的软件越多，挂的越快。有人会说这是你不看软件发行注记的后果，此言差矣。一个需要看发行注记才能更新的系统，本身就是有问题。Arch Linux 唯一优点就是软件新。似乎随处可见的就是 Arch Linux。Arch Linux 似乎是与苦难哲学挂钩的，关于这一点，请参看 [第十九章-第三节](#)。

FreeBSD 与 Linux 不同之处

- FreeBSD 仍然使用传统的 INIT 引导，而非 systemd；
- FreeBSD root 用户 shell 默认是 csh，而不是 bash；
- FreeBSD 基本系统几乎不包含任何非 BSD 协议的软件，并致力于去 GNU 化（这意味着基本系统不使用 Glibc、GCC 等软件），见

<https://wiki.freebsd.org/GPLinBase>

- FreeBSD 的用户配置文件和系统配置文件严格分离，即内核和基本系统与第三方应用程序是完全分离的；
- FreeBSD 项目是作为一个完整的操作系统维护的，而非内核与 userland 单独维护；也就是说如果你要使用 FreeBSD，那么就只有一个 FreeBSD 可选；
- FreeBSD 没有 free 命令也不支持安装这个包（FreeBSD 早就不使用 procfs 了），FreeBSD 基本系统自带的文本编辑器有 ee 和 vi（不是软链接到 vim 的 vi，是真实的 vi），没有预装 wget，而是 fetch。

命令替代/软件替代

因为 Linux 广泛使用的也是 GNU 工具，因此只要理论上不是依赖于特定的 Linux 函数库，该工具都可以在 FreeBSD 上运行。

Linux 命令/GNU 软件	FreeBSD 命令/BSD 软件	需要安装的包（安装方式）	作用说明	额外说明/
lsusb	lsusb	pkg install usbutils	显示 USB 信息	粗略地可以 /var/run/
lspci	lspci	pkg install pciutils	显示 主板 信息	粗略地可以 /var/run/
lsblk	lsblk	pkg install lsblk	显示 磁盘 使用 情况	/
free	freecolor	pkg install freecolor	显示 内存 使用 情况	FreeBSD 供 free 命 其依赖 Li 包 procs 才 是呢，Free 不使用 pro 如实在需要 以用 <a href="https://git
keck/free">https://git keck/free 命令是 vi
lscpu	lscpu	pkg install lscpu	显示 处理器信 息	/
glibc	bsdlibc	/	C 库	/

Linux 命令/GNU 软件	FreeBSD 命令/BSD 软件	需要安装的包（安装方式）	作用说明	额外说明/
GCC	LLVM + Clang	/	编译器、编译链工具	非要用也要 install
vim	vim	pkg install vim	文本编辑器	FreeBSD 的 vi 并不到 vim，而是的 v
wget	wget	pkg install wget	下载器	系统默认的是 fe
bash	bash	pkg install bash	shell	系统默认 shell 是 csh 导致配置输入变量时遇到可能会无法模拟
NetworkManager	networkmgr	pkg install networkmgr	网络连接工具	NetworkManager 赖 systemd 接移

第九节 参考资料与贡献者名单

参考书目

相关书籍：《Absolute FreeBSD 3rd》、《FreeBSD操作系统设计与实现（原书第2版）》。旧的变化也不是很大。不像 linux 有这么多入门书籍，什么 《XX 秒精通 Linux》，《Linux XX 学》……当然上边这些书，学 Linux 的也尽量别看，质量太差。由于历史上的原因，看 UNIX 相关书籍即可。



[在线试读地址](#)

贡献者名单（以下排名不分前后）

凌莞

星不萌

雨天

柳离枝

peiyafei

ykla

艳阳天

X-Ray

fanyang1997

orzyyyy

Rintim

tomblackwhite

isNijikawa

星不萌

qinghecyn

清热解毒口服液

墨子

Jack

兜率

杭永聪

Shengyun

仰望天空

魔王酱

极品盗号

blu10ph

livrth

注意：如果缺少了您的信息或者不想被列出，请发起 issue。

第十节 编撰说明

pkg 与 ports

因为 FreeBSD 有两种安装软件的方式（但并非所有软件都支持 pkg 的安装方式），因此为了方便，在本跑路教程中可能不会同时出现两种方式的安装说明。但希望大家明白，只是为了方便，而并非不能使用 ports 或者 pkg 进行安装或必须使用二者其一进行安装。

本书中命令前的符号含义

代表 `root` 下的操作，同 `sudo`。

\$ 代表一般用户权限

对用户的要求

以高等院校计算机科学与技术学科一般本科毕业生所能达到的及格或及格以上水平为编写难度基准。如未能达到要求，请自行学习。

本书定位

我们的目标并非是 Handbook 的翻译，而是编写一本类似于《鸟哥的 Linux 私房菜：基础学习篇》+《鸟哥的 Linux 私房菜：服务器架设篇》二合一的基于 FreeBSD 的教程。也就说我们是 Handbook 的超集。

第〇节 图解安装

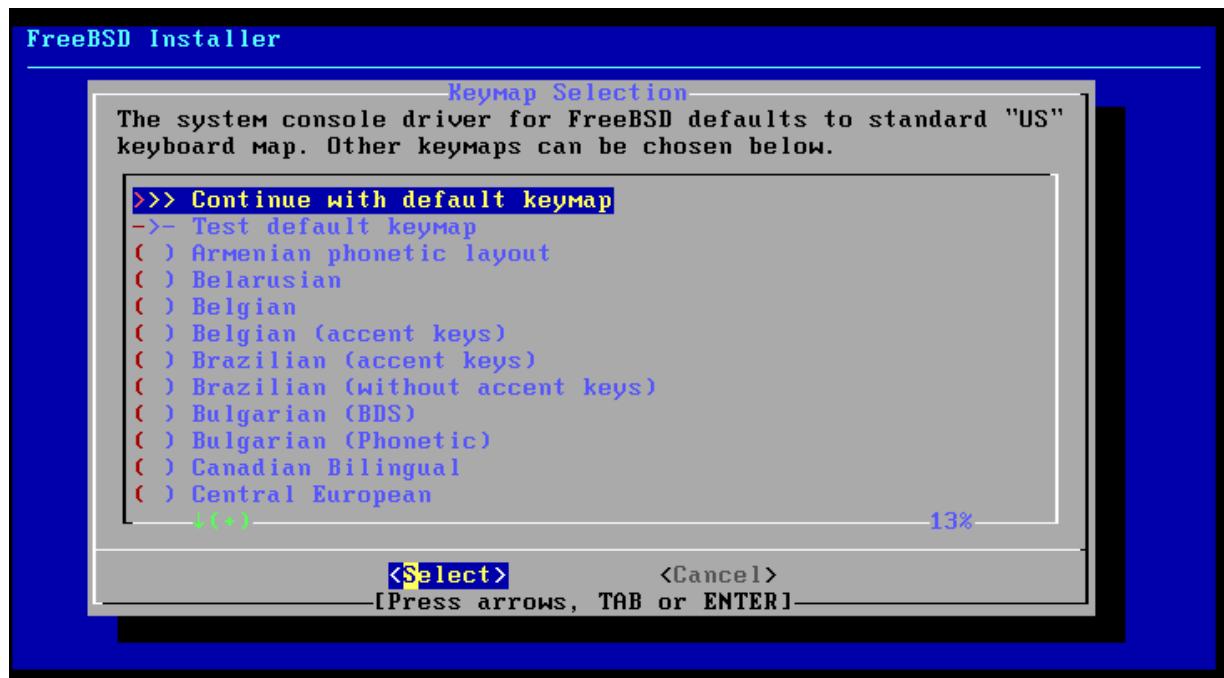


推荐等待十秒即可进入，也可以直接回车进入。

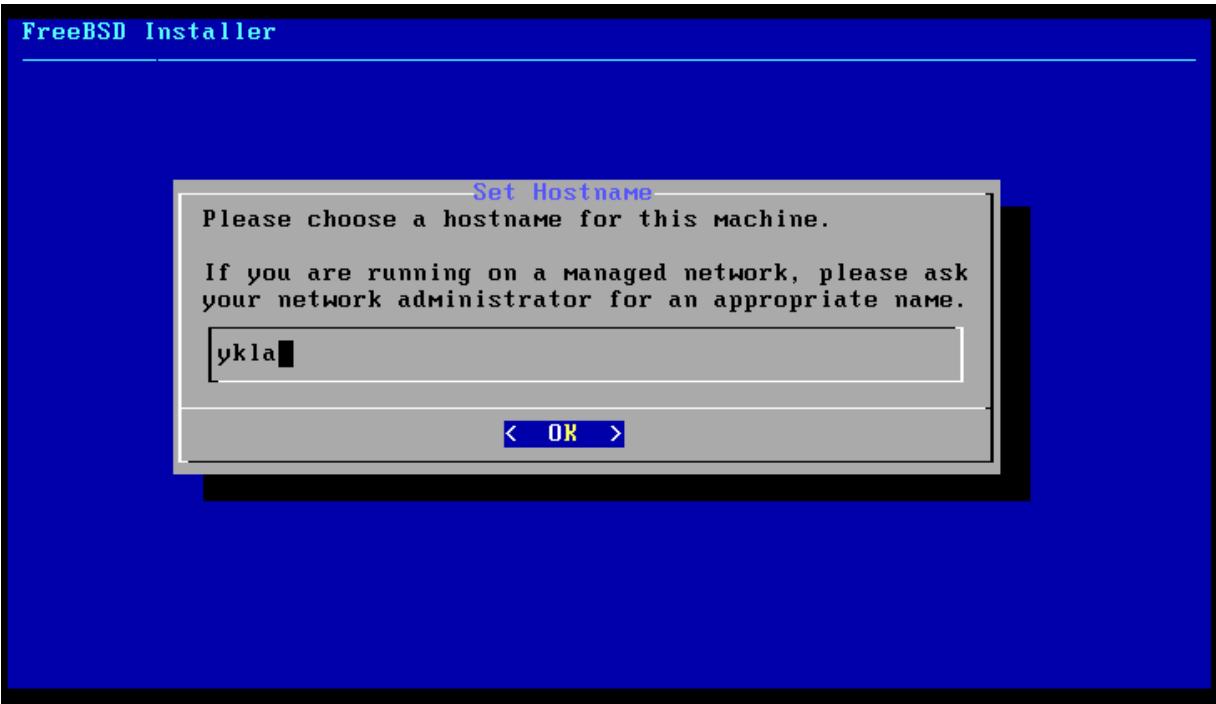
选项	解释
ACPI Support	ACPI 支持
Safe Mode	安全模式
Single User	单用户模式
Verbose	啰嗦模式，显示更多输出



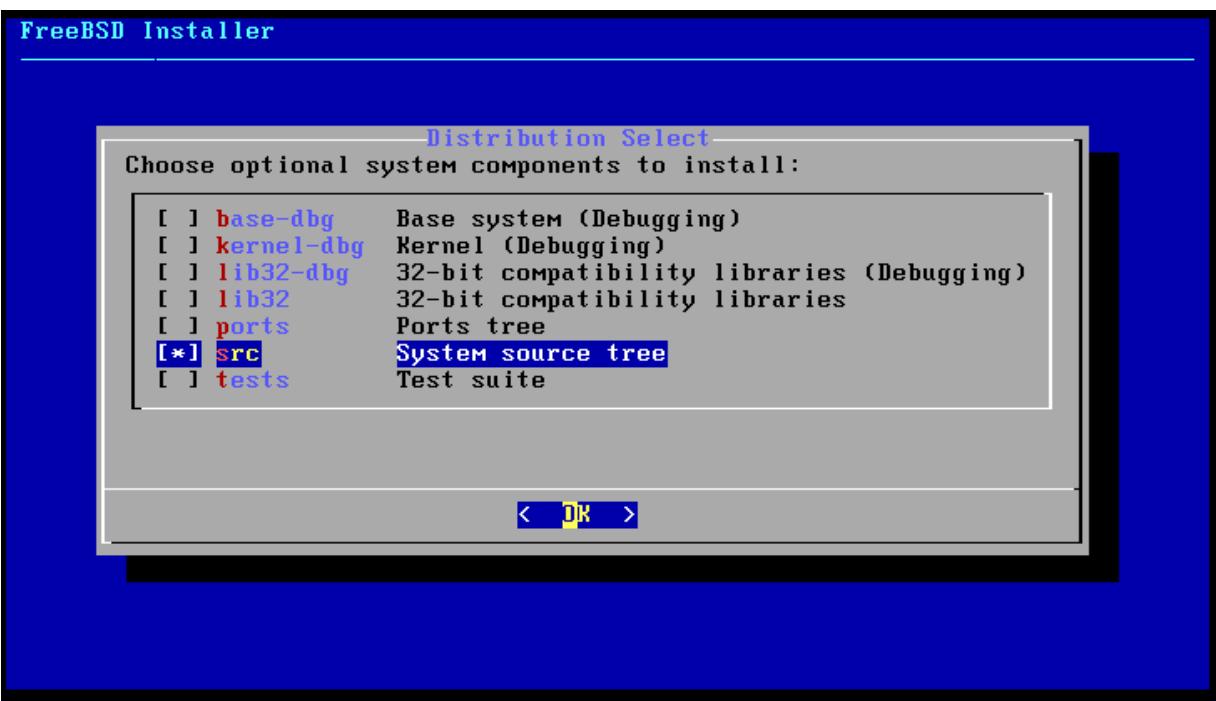
选择 `install`，按下 回车键 进行安装。



这里是设置键盘，直接回车即可。

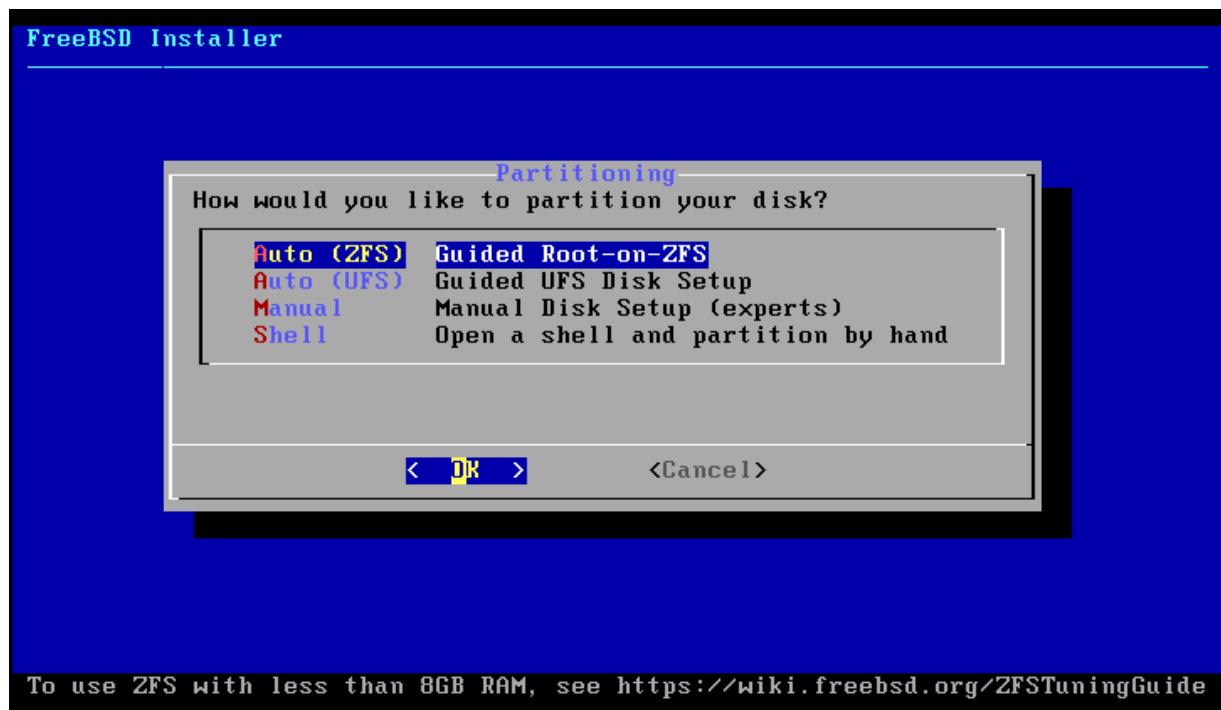


此处是设置主机名。

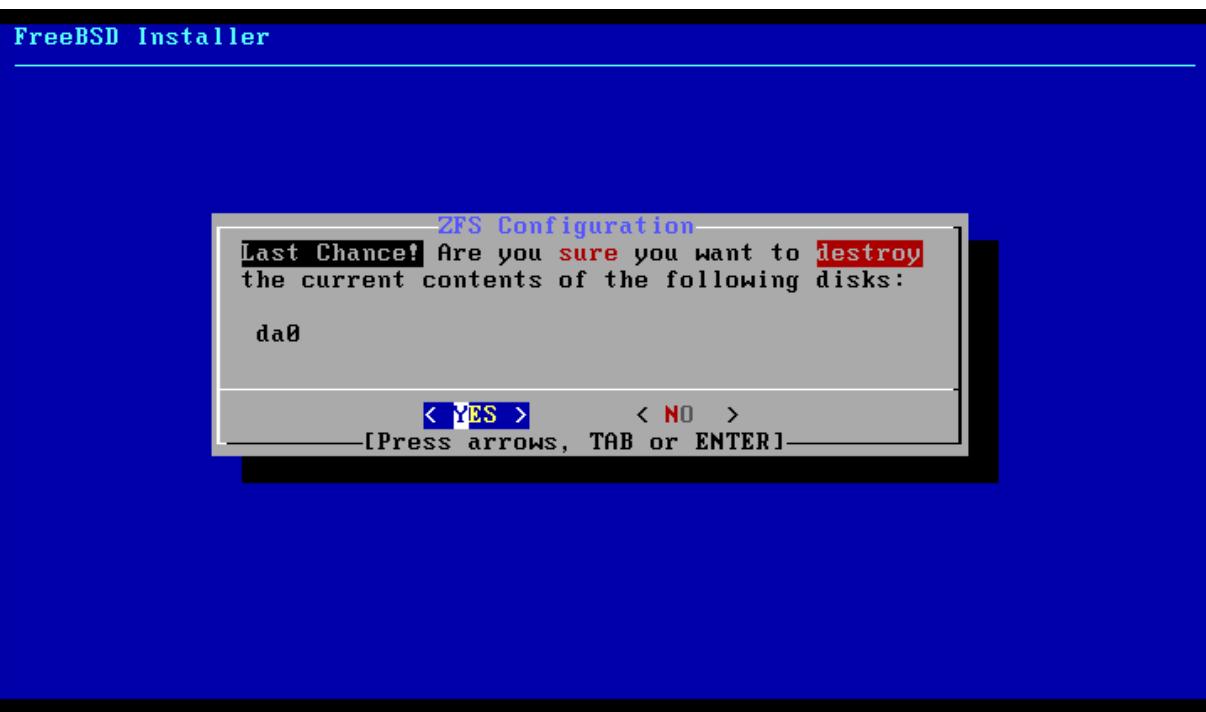
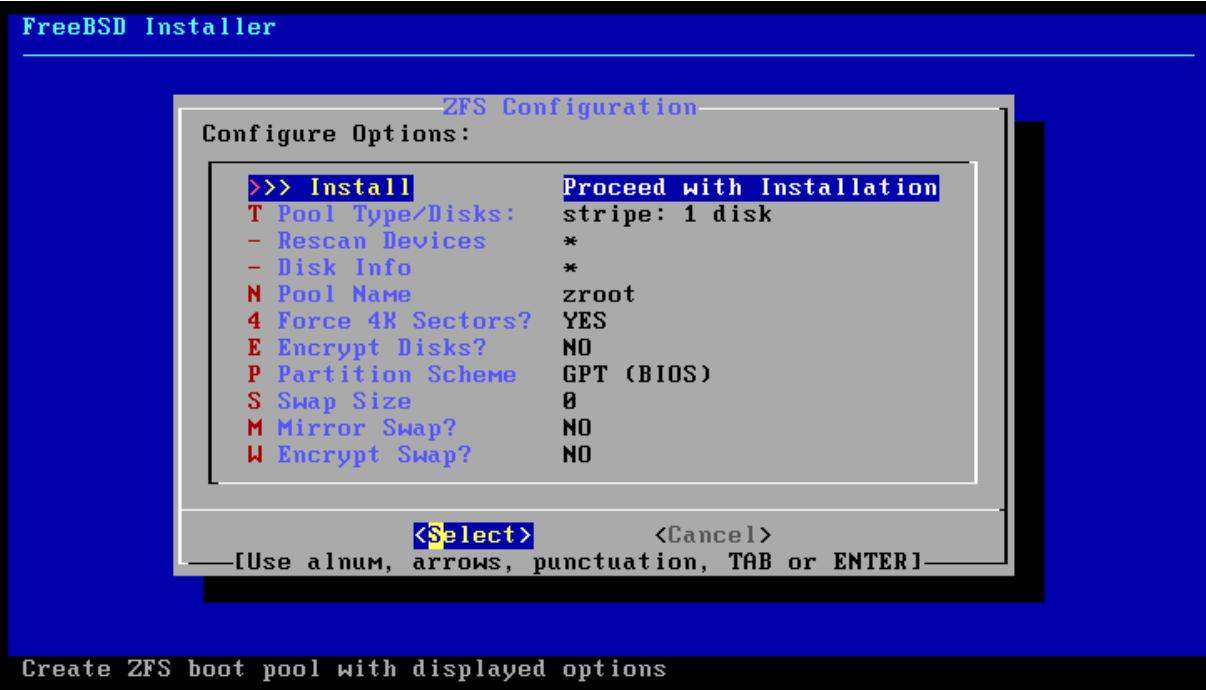


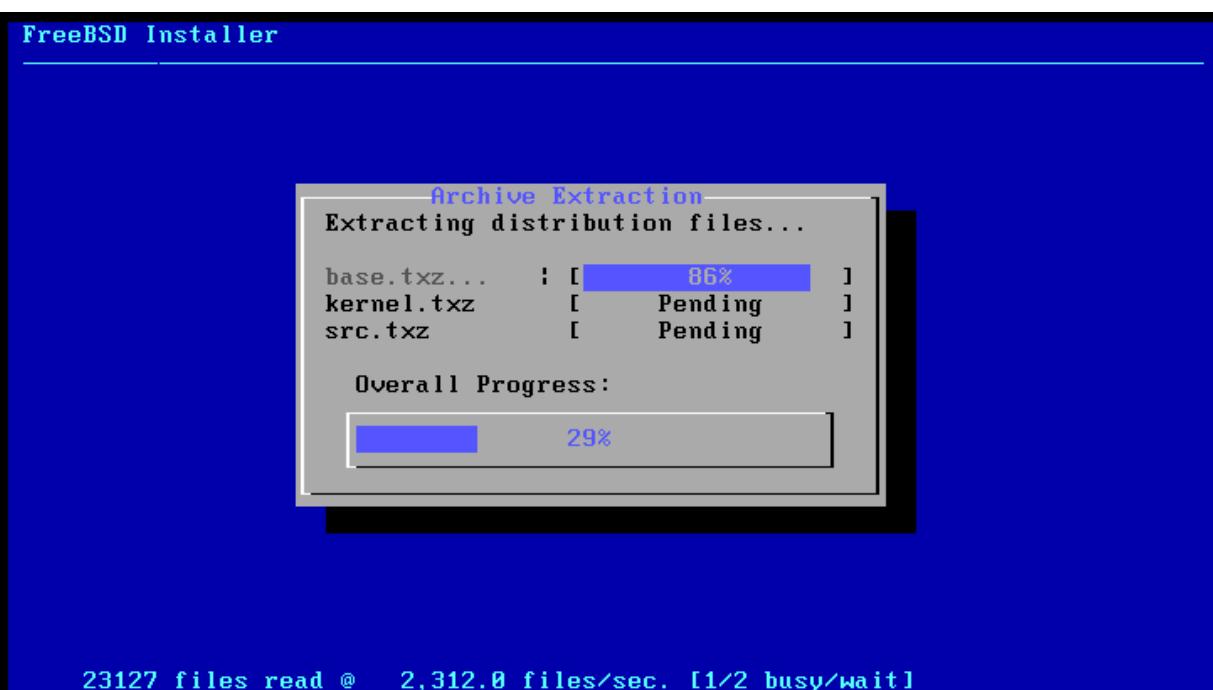
推荐：只选 `src` 以及 `lib32`。即使选了 `ports` 也不会安装的，还是空的。

选项	解释
base- dbg	基础工具，如 cat、ls 等，并激活调试符号
kernel- dbg	内核和模块的调试符号被激活
lib32- dbg	用于在激活调试符号的 64 位版本的 FreeBSD 上运行 32 位应用程序的兼容库
lib32	用于在 64 位版本的 FreeBSD 上运行 32 位应用程序的兼容库
ports	ports
src	系统源代码
tests	测试工具



推荐：文件分区详解在第 6 章。这里推荐选择 auto ZFS/UFS



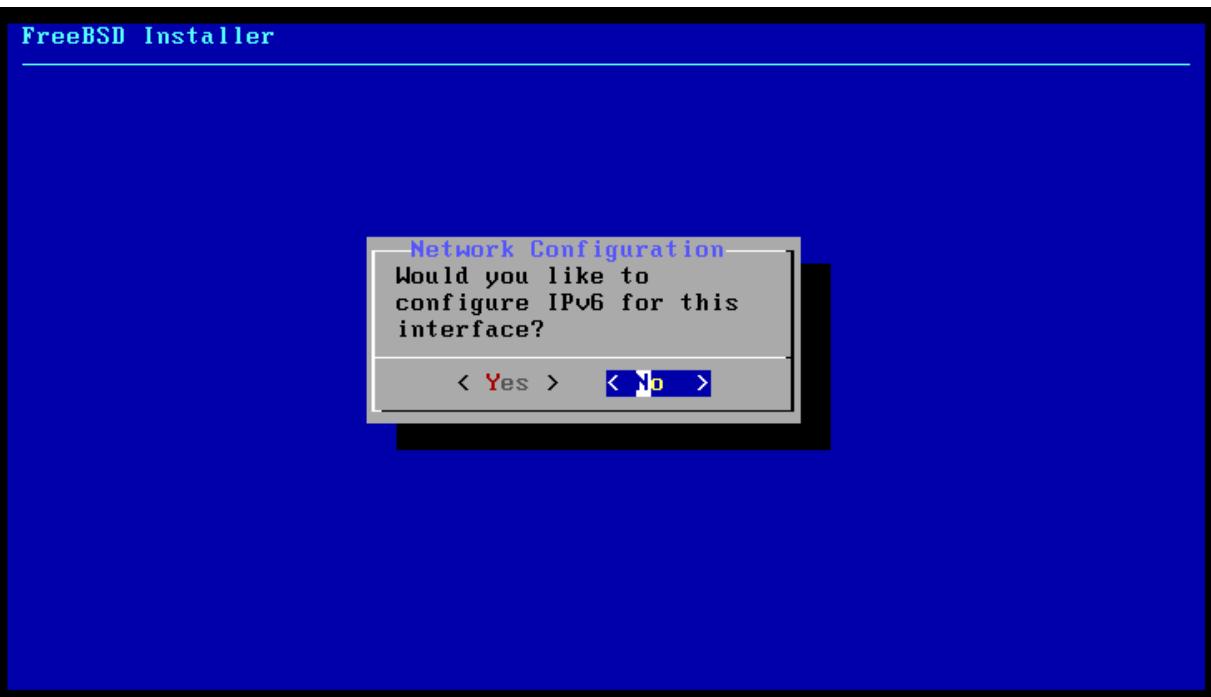


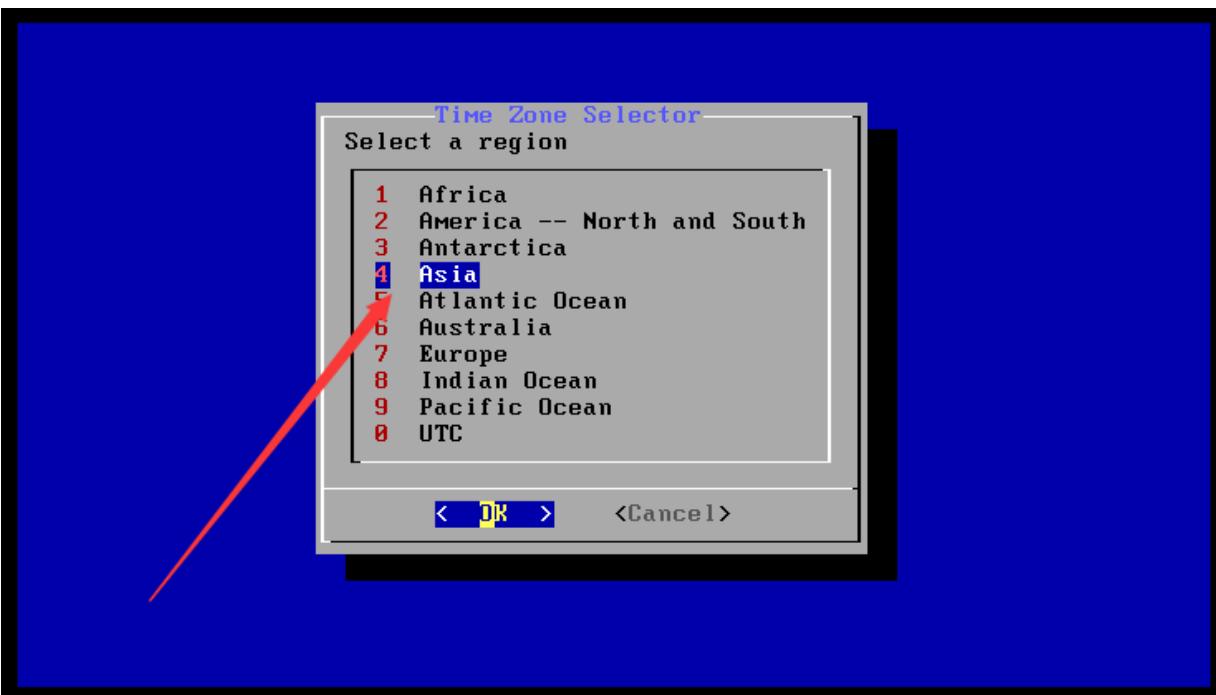
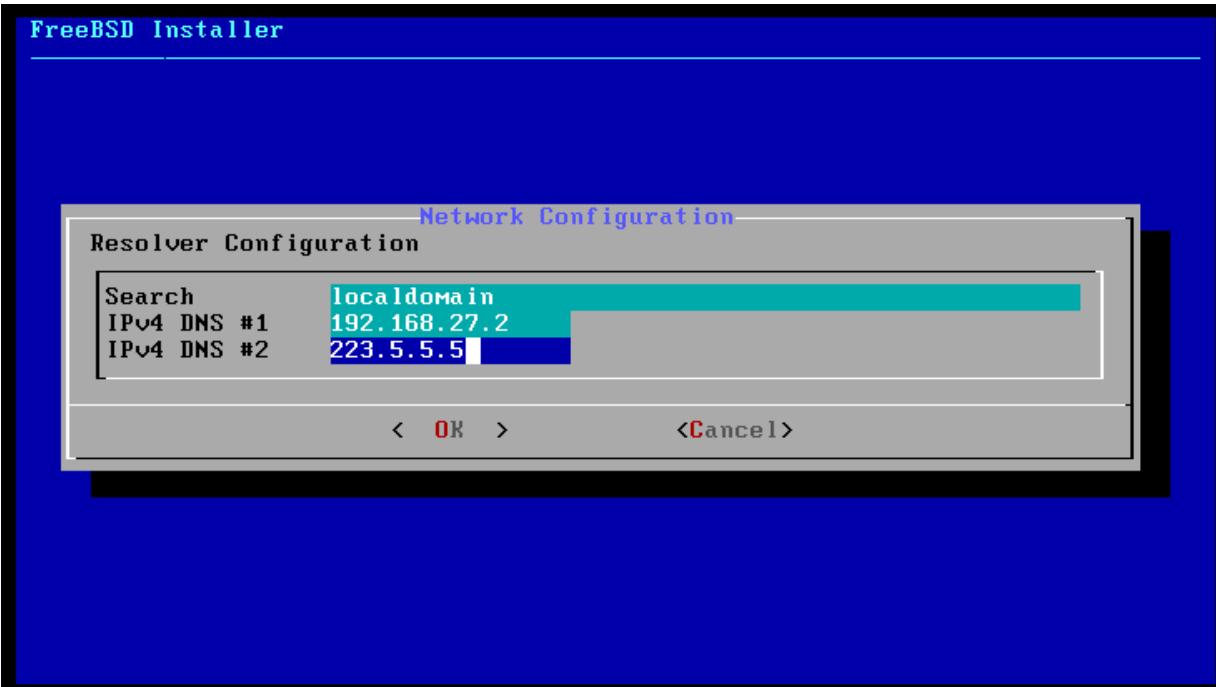
```
FreeBSD Installer
=====
Please select a password for the system management account (root):
Typed characters will not be visible.
Changing local password for root
New Password:■
```

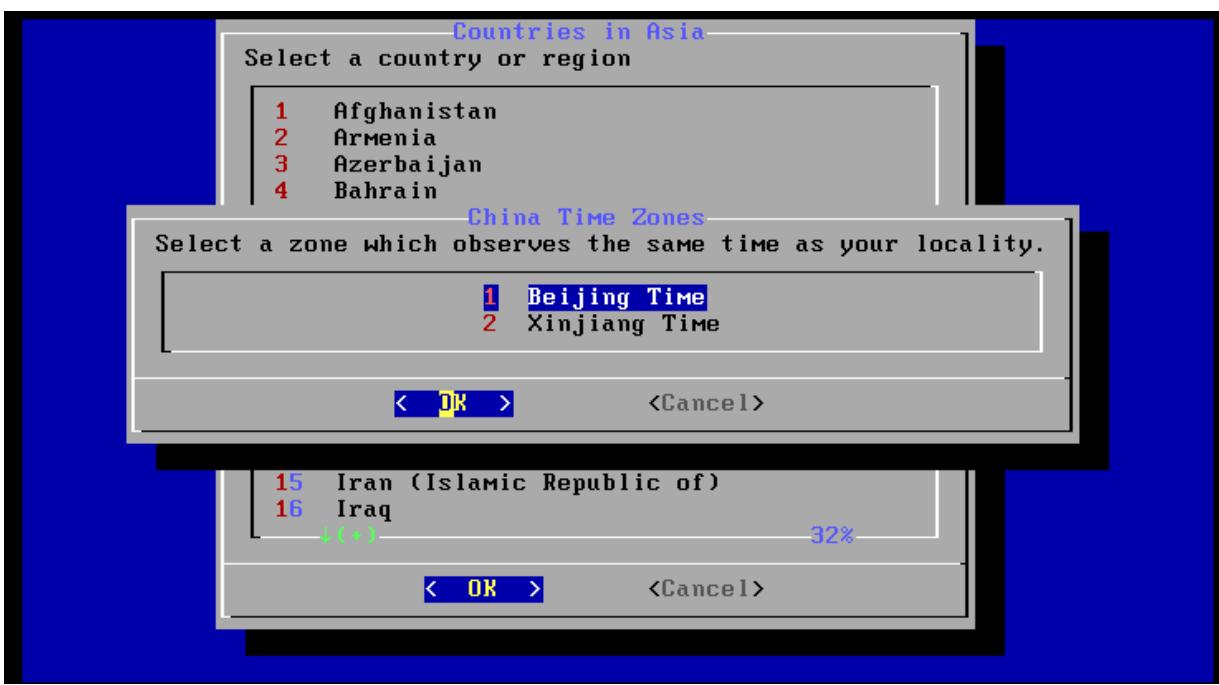
FreeBSD Installer

```
Network Configuration
Please select a network interface to configure:
[em0 Intel(R) PRO/1000 Network Connection]
< [OK] >      <Cancel>
```

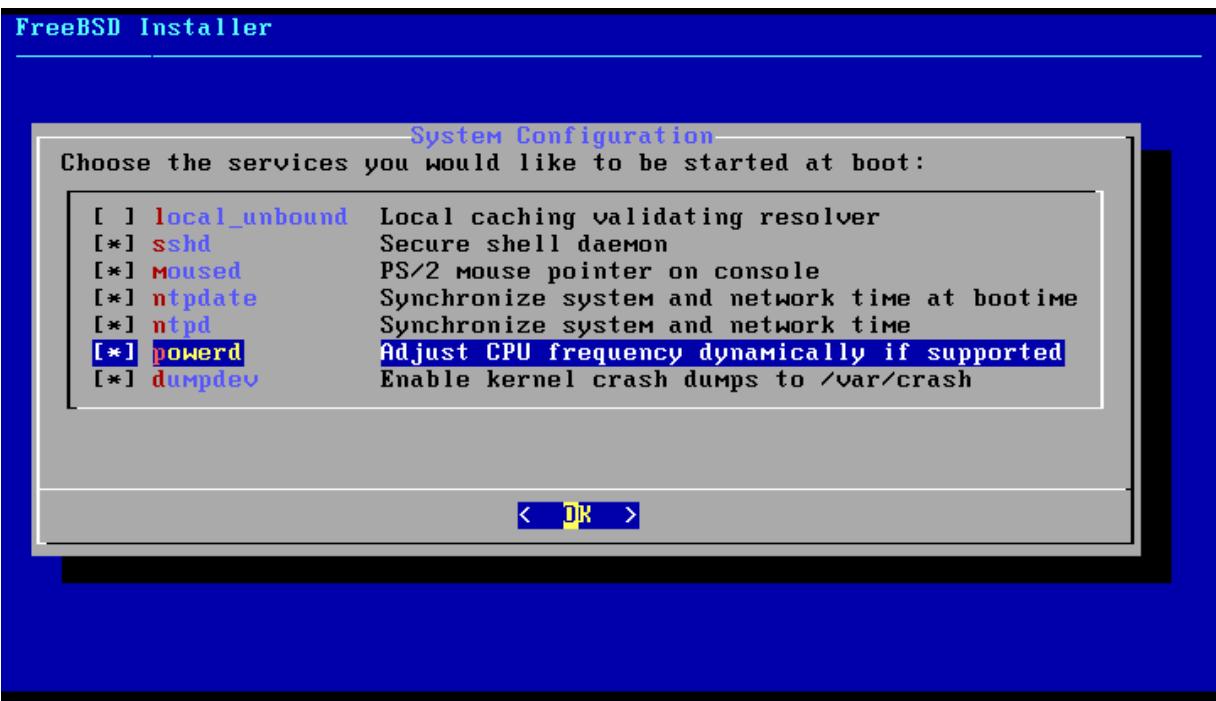












建议不选 `local_unbound`，会影响 DNS，见

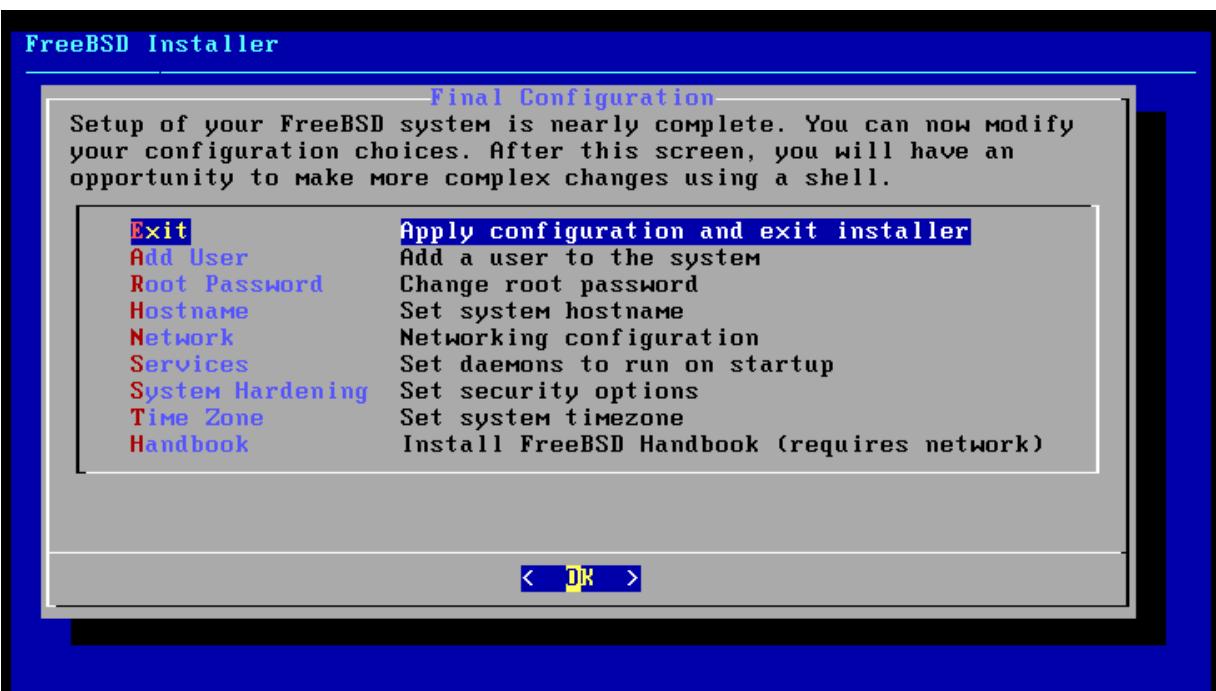
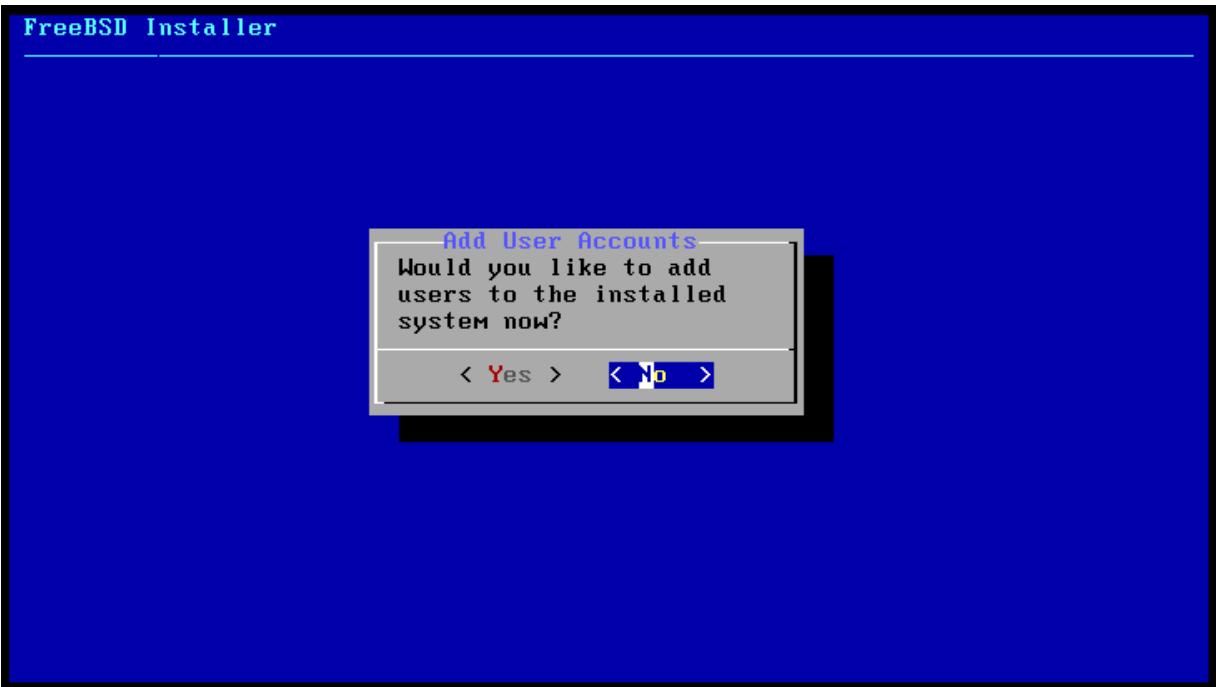
https://bugs.freebsd.org/bugzilla/show_bug.cgi?id=262290。

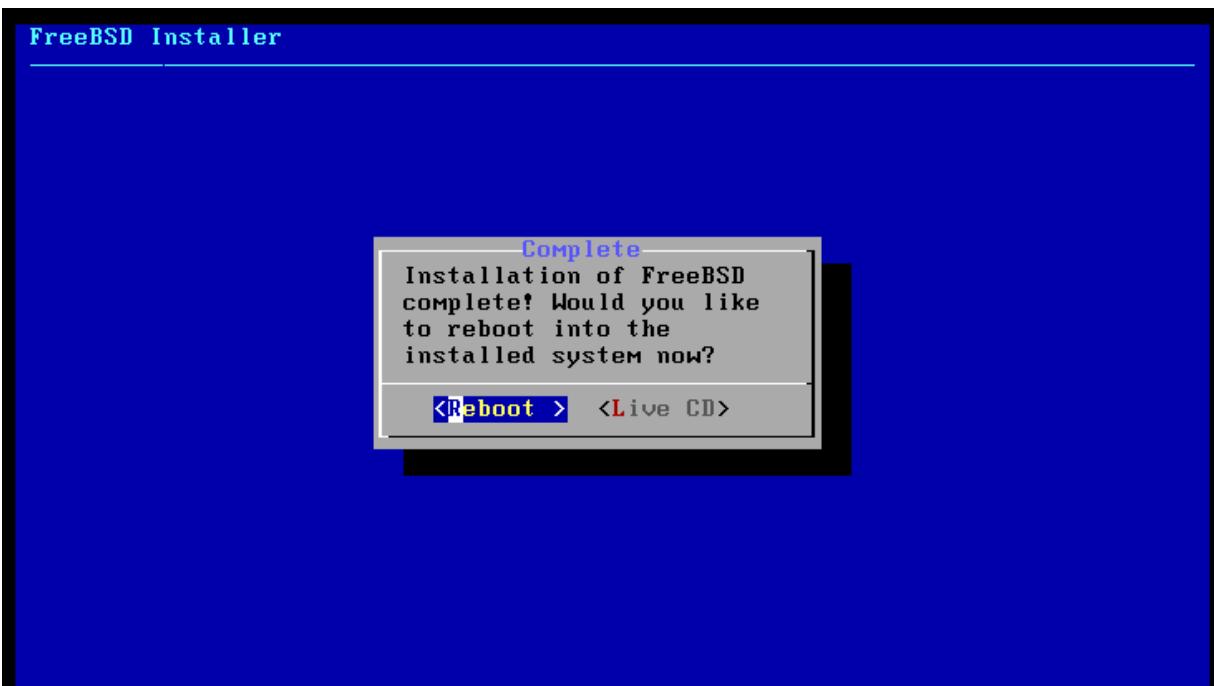
虚拟机不需要选 `powerd`。

选项	解释
local_unbound	启用 DNS 本地非绑定。有必要记住，这是基础系统的非绑定，只用于作为本地缓存转发解析器使用。
sshd	开启 ssh 服务
moused	在 tty 界面显示鼠标
ntpdate	启用启动时的自动时钟同步功能。
ntpd	用于自动时钟同步的网络时间协议（NTP）守护程序。
powerd	电源管理
dumpdev	启用崩溃转储，用于调试系统



推荐选择：这里是安全增强选择，应该选择 `disable_sendmail`，如果不禁止这个服务会使你在每次开机的时候卡上几分钟，而且这个服务本身没什么用，发邮件用的。





第一节 三种虚拟机与 FreeBSD 版本比较

FreeBSD 版本比较

已知 FreeBSD 有以下版本：rc、beta、release、current、stable。

release 是绝对的 stable，而 stable 和 current 都是开发分支，不太稳定。

current 相对稳定后会推送到 stable，但是不保证 stable 没有大的 Bug，只是确保其 ABI 兼容。

FreeBSD 版本选择

其中 rc 和 beta 都是测试版本；

日常使用应该选择 release 版本，当有多个 release 版本时，应该选择最新的一个；

如果硬件比较新或者需要测试 ax200 网卡，应该选择 current 版本，是滚动开发版。

注意：只有 rc、beta 和 release 才能使用 freebsd-update 命令更新系统（[且是一级架构](#)），其余系统均需要通过源代码编译的方式更新系统。

三种虚拟机比较

Virtual Box 与 VMware Workstation Pro

个人计算机上常用的虚拟机有两种，一是 Virtual Box，另一个是 VMware Workstation Pro。

一般来说，在 Windows 系统上建议使用 VMware Workstation Pro（以下简称 VM），在 Linux 系统上建议使用 Virtual Box（以下简称 VB）。

VM 是闭源的由商业公司提供的，是需要付费的，可用免费试用，也有免费版本 *VMware Workstation Player*；VB 是 Oracle 公司的开源产物，是免费的。

就个人而已，VM 在实际使用中 Bug 会比 VB 少一些：VB 会有一些奇奇怪怪的问题（详见 VB 章节），且很花时间去排除解决。但是为了给与大家更多自由，我们将两种虚拟机的安装使用方法都提供给大家。

Hyper-V

Hyper-V 是 Windows 开发的虚拟机，分为 Gen 1 和 Gen 2。

Gen 1 和 Gen 2 区别如下：

Hyper-V 代数	硬盘	启动引导
Gen 1	IDE + SCSI	仅 MBR
Gen 2	仅 SCSI	仅 UEFI + 安全启动支持 + PXE 支持

FreeBSD 目前（截止到 2022-1-28）尚且不能在 Hyper-V 上正常运行鼠标或键盘，因为鼠标没有驱动。不建议使用，具体支持情况如下表。

系统快速创建的为 Gen 2。

Hyper-V 代数	FreeBSD 版本	鼠标	键盘	备注
Gen 1	13.0	支持	不支持	/
Gen 2	13.0	不支持	支持	需要修改参数 sysctl kern.evdev.rcpt_mask=6
Gen 2	14.0-2022-1-27	不支持	支持	/

第二节 FreeBSD 13.0 安装—— 基于 Virtual Box

注意：不推荐新手使用此虚拟机，因为需要踩的坑比较多。

VirtualBox 下载

点击 [download](#) 即可下载：

<https://www.virtualbox.org>

VirtualBox 虚拟机 FreeBSD 配置

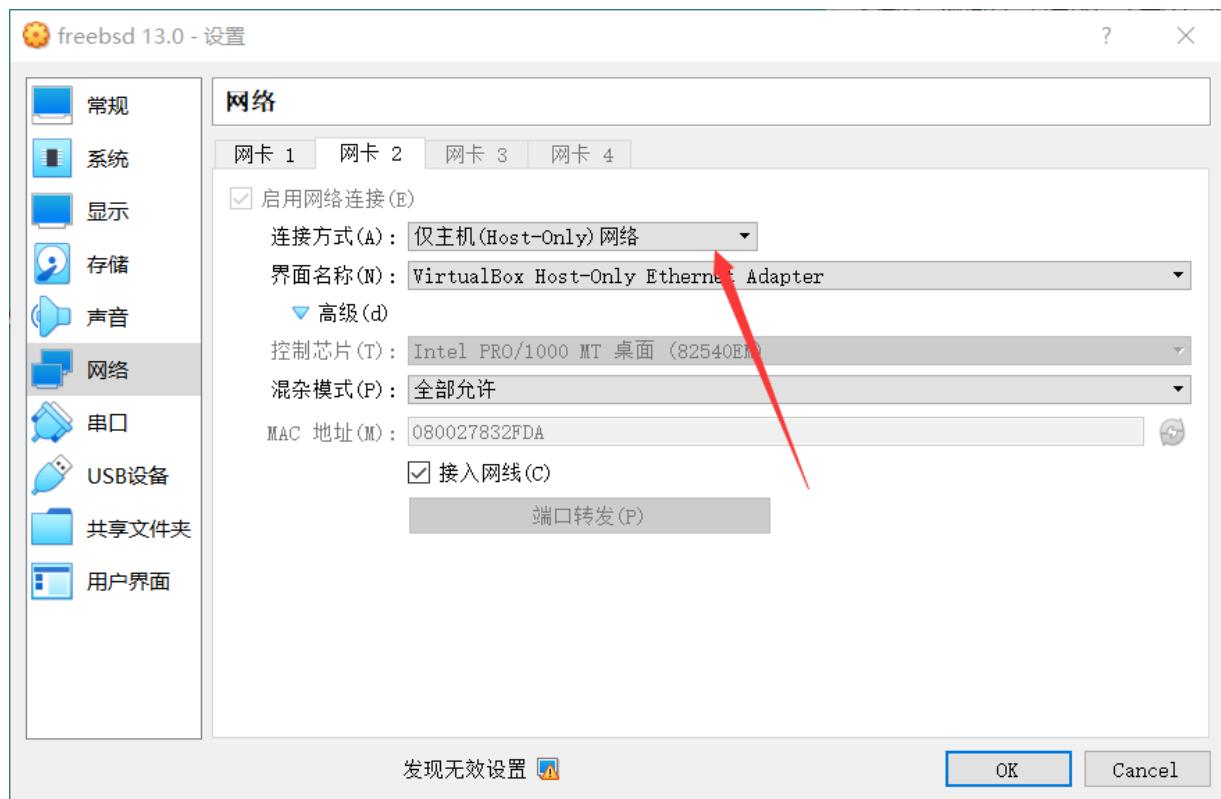
安装设置

安装完成后请手动关机，卸载或删除安装光盘，否则还会进入安装界面。

网络设置

网络设置比较复杂，为了达到使用宿主机（如 Windows10）控制虚拟机里的 FreeBSD 系统的目的，需要设置两块网卡——一块是NAT网络模式的网卡用来上网、另一块是仅主机模式的网卡用来互通宿主机。如图所示：





使用命令 `# ifconfig` 看一下，如果第二块网卡 `em1` 没有获取到 ip 地址，请手动 DHCP 获取一下：`# dhclient em1` 即可(为了长期生效可在 `/etc/rc.conf` 中加入 `ifconfig_em1="DHCP"`)

如果没有网络（互联网）请设置 DNS 为 `223.5.5.5`。如果不会，请看本章第四节。

安装增强工具

```
# pkg install virtualbox-ose-additions
```

xorg 可以自动识别驱动，不需要手动配置

`/usr/local/etc/X11/xorg.conf` (经过测试手动配置反而更卡，点一下要用 5 秒钟动一次……)。

显卡控制器用 `VBoxSVGA` 即可。

编辑 `# ee etc/rc.conf`，增加以下内容：

```
vboxguest_enable="YES"  
vboxservice_enable="YES"
```

启动服务，调整权限：

```
# service vboxguest restart # 可能会提示找不到模块，但是不影响使用  
# service vboxservice restart  
# pw groupmod wheel -m <yourname> # sudo 权限  
# pw groupmod opt -m <yourname> # 开机重启 权限
```

第三节 FreeBSD 13.0 安装—— 基于 Vmware Workstation Pro 16

视频教程（一共4节，完整版本请点击去 bilibili 观看）

<https://www.bilibili.com/video/BV14i4y137mh>

Release 正式版 镜像下载地址：

<https://download.freebsd.org/ftp/releases/amd64/amd64/ISO-IMAGES/13.0/FreeBSD-13.0-RELEASE-amd64-dis1.iso>

Current 测试版（仅限专业用户，对于该版本来说，无法启动，环境变量错误都是正常的事情！） 镜像下载地址（北京交通大学开源镜像站）：<https://mirror.bjtu.edu.cn/freebsd/snapshots/ISO-IMAGES/14.0/>

FreeBSD 旧版本下载地址：<http://ftp-archive.freebsd.org/pub/FreeBSD-Archive/old-releases/amd64/ISO-IMAGES/>

VMware Workstation Pro 下载

Vmware 16.2.2 build-19200509 目前无法缩放屏幕，已经报告 bug：

<https://gitlab.freedesktop.org/xorg/app/appres/-/issues/1>、https://bugs.freebsd.org/bugzilla/show_bug.cgi?id=262118 正常版本可至 QQ 群 905149943 群文件下载

VMware Workstation Pro 是免费试用下载的，请勿从第三方站点下载，否则会造成一些苦难哲学的后果。点击 Download NOW 即可。左边是 Windows 系统使用，右侧是 Linux 系统使用。该软件虽是收费的，但是授权码并不难获得。

<https://www.vmware.com/products/workstation-pro/workstation-pro-evaluation.html>

VMware Workstation 16 Player 下载

VMware Workstation 16 Player 是个人免费使用的，你也可以选择此版本。

<https://www.vmware.com/products/workstation-player/workstation-player-evaluation.html>

网络设置

请使用 NAT 模式，如果不能与宿主机（物理机）互通，请打开 VMware 编辑-虚拟网络管理器，移除第一项“桥接”。移除后重启虚拟机应该就可以了。

如果没有网络请设置 DNS 为 223.5.5.5。请看本章第四节。

显卡驱动以及虚拟机增强工具

显卡驱动

Vmware 16.2.2 build-19200509 目前无法缩放屏幕，已经报告 bug：

<https://gitlab.freedesktop.org/xorg/app/appres/-/issues/1>、https://bugs.freebsd.org/bugzilla/show_bug.cgi?id=262118 正常版本可至群文件下载

VMware 自动缩放屏幕请安装显卡驱动，即：

```
# pkg install xf86-video-vmware
```

wayland 下也需要安装该驱动。

如果屏幕显示不正常（过大），请尝试：编辑虚拟机设置——>硬件、设备——>显示器——>监视器、指定监视器设置——>任意监视器的最大分辨率，设置为主机的分辨率或者略低于主机分辨率均可。

虚拟机增强工具

如果没有桌面：

```
# pkg install open-vm-tools-nox11
```

如果有桌面

```
# pkg install open-vm-tools
```

具体配置

将下面几行加入 `/etc/rc.conf`

```
vmware_guest_vmblock_enable="YES"
vmware_guest_vmhgfs_enable="YES"
vmware_guest_vmmemctl_enable="YES"
vmware_guest_vmxnet_enable="YES"
vmware_guestd_enable="YES"
```

编辑 `/boot/loader.conf`

写入

```
fusefs_load="YES"
```

共享文件夹

请先安装虚拟机增强工具。

```
# mount -t .host:/ /mnt/hgfs
```

查看共享文件夹

```
# ls /mnt/hgfs
```

注意：由于 BUG，FreeBSD 11/12 可能在 VmMare 的 UEFI 环境下无法启动。经测试 13.0 正常启动。

第四节 腾讯云轻量云及其他服务 器 dd 安装 FreeBSD

阿里云用户请注意，目前不支持从 12.1 升级到任一版本，因为
<https://reviews.freebsd.org/D27262>

忠告：如果你还不懂什么是 `dd`，不建议食用本文。这一切超越了您的动手能力和知识储备。

此外，不再受安全支持的版本如 `9.2`，请参考本文并结合手动安装 FreeBSD 章节操作。

视频教程

<http://b23.tv/zcfHa4K>

文字教程

腾讯云轻量云以及阿里云等机器都没有 FreeBSD 系统的支持，只能通过特殊的的方法自己暴力安装。请注意数据安全，以下教程有一定危险性和要求你有一定的动手能力。

他是一個服務器面板里沒有 FreeBSD 鏏像 IDC，所以要用奇怪的方法來安裝了。因為 FreeBSD 和 Linux 的內核不通用，可執行文件也不通用，所以無法通過 chroot 再刪掉源系統的方法安裝。安裝的方法是先在內存盤中啟動 FreeBSD 系統，也就是 mfsBSD，再格式化硬盤安裝新系統。mfsBSD 是一個完全載入內存的 FreeBSD 系統，類似於 Windows 中的 PE。

我們需要下載 img 格式的 mfsBSD 鏏像，我也不知道 mfsBSD 的服務器在國內連接如何，所以就把文件先傳到了另一台國內的文件服務器上。

為什麼不能直接 dd？（錯誤示範，僅供說明，請勿執行）

我試了在正常的 Linux 系統內直接把 mfsBSD 的 img dd 到硬盤里，重啟之後雖然正常加載 bootloader，但是可能因為系統又對硬盤進行了寫入而無法正常挂載內存盤。

```
# wget https://mfsbsd.vx.sk/files/images/13/amd64/mfsbsd-se-13.0-  
RELEASE-amd64.img -O- | dd of=/dev/vda
```

这里的 | 是管道的意思，将上一个命令的标准输出作为下一个命令的标准输入。 -o- 指把文件下载输出到标准输出，而 dd 没有指定 if 时会自动从标准输入读取内容。

```
cd0: 141MB (72276 2048 byte sectors)
GEOM: vtbd0: the secondary GPT header is not in the last LBA.
Root mount waiting for: usbus0
uhub0: 2 ports with 2 removable, self powered
mountroot: waiting for device /dev/md0...
Mounting from ufs:/dev/md0 failed with error 19.

Loader variables:
vfs.root.mountfrom=ufs:/dev/md0

Manual root filesystem specification:
<fstype>:<device> [options]
    Mount <device> using filesystem <fstype>
    and with the specified (optional) option list.

eg. ufs:/dev/da0s1a
    zfs:zroot/ROOT/default
    cd9660:/dev/cd0 ro
        (which is equivalent to: mount -t cd9660 -o ro /dev/cd0 /)

?
List valid disk boot devices
.
Yield 1 second (for background tasks)
<empty line> Abort manual input

mountroot> █
```

真正的 mfsBSD 启动方法

就是因为刚才说的问题，而且 FreeBSD 和一般的 Linux 是不同的生态，我们需要先进入一个 Linux 的内存盘，再在内存中运行的 Linux 里将 mfsBSD 写入硬盘。

就在 mfsBSD 下载位置的下方，有一个 [mfsLinux](#)，就是我们可以用的工具。由于它只有 ISO 格式，没法直接放在当前环境下启动，而它说自己是纯 initrd 类型的，我们就把启动它的 initrd 和内核提取出来，放在硬盘里手动启动。

我们知道在一般的 Linux 系统中，initrd 是一个打包成内存盘的微型但完整的 Linux 根目录，里面有一些比如说加载驱动，挂载硬盘，以及启动初始化程序的必要数据。开机时内核与 initrd 被 Bootloader 加载，initrd 中的脚本进行启动的准备工作并运行硬盘里的初始化程序。

我们先把从那个 ISO 提取出来的内核和 initrd 文件放在根目录比如说 qwq 这个文件夹下，然后重启机器进入 GRUB 的命令行界面（可以在倒计时的时候按 `e` 进入编辑模式，删掉所有 `linux`、`initrd` 行原有内容，写完后按 `Ctrl x` 即可加载），手动启动指定的内核和 initrd（可以用 `Tab` 键补全路径）。

```
linux (hd0,msdos1)/qwq/vmlinuz
initrd (hd0,msdos1)/qwq/initramfs.igz
boot
```

```
GNU GRUB version 2.02+dfsg1-20+deb10u2

Minimal BASH-like line editing is supported. For the first word, TAB lists possible
command completions. Anywhere else TAB lists possible device or file completions. ESC
at any time exits.

grub> ls
(hd0) (hd0,msdos1)
grub> ls (hd0,msdos1)
Partition hd0,msdos1: Filesystem type ext* - Last modification time 2021-07-25 05:44:38
Sunday, UUID 28507de7-6bf8-4229-b994-4169fdb432e - Partition start at 1024KiB - Total size
52427759.5KiB
grub> ls (hd0,msdos1)/
lost+found/ etc/ media/ vmlinuz.old var/ bin usr/ sbin lib lib32 lib64 libx32 boot/ dev/ home/
proc/ root/ run/ sys/ tmp/ mnt/ srv/ opt/ qwq/ data/ initrd.img.old vmlinuz initrd.img
initramfs.igz
grub> ls (hd0,msdos1)/qwq
initramfs.igz vmlinuz
grub> linux (hd0,msdos1)/qwq/vmlinuz
grub> initrd (hd0,msdos1)/qwq/initramfs.igz
grub> _
```

这个特制的 initrd 启动之后并没有加载本地的系统，而是自己连接了网络并打开 ssh 服务器。于是我们就获得了一个运行在内存中的 Linux 系统。

这个时候服务器应该就可以被 ssh 连接上了，并且可以安全的格式化硬盘。

mfsBSD 和 mfsLinux 镜像的 root 密码默认是 `mfsroot`

```
# cd /tmp
# wget https://mfsbsd.vx.sk/files/images/13/amd64/mfsbsd-se-13.0-
RELEASE-amd64.img
# dd if=mfsbsd-se-13.0-RELEASE-amd64.img of=/dev/vda
# reboot
```

提示：建议在此处使用服务器的“快照”功能对服务器进行备份，以防以下教程操作失误重来耽误时间。

安装 FreeBSD

等待系统初始化完成之后就可以 ssh 上去了，不过还需要一个步骤才能正常安装（不然走完安装向导会报错）

我们还需要下载 FreeBSD 的安装清单文件。

```
# mkdir -p /usr/freebsd-dist
# cd /usr/freebsd-dist
# fetch http://ftp.freebsd.org/pub/FreeBSD/releases/amd64/13.0-
RELEASE/MANIFEST
```

最后执行 `# bsdinstall` 进行正常的安装即可（最好使用自动 ufs 分区）。请注意大多数服务器如本文的示例腾讯云轻量云，是不支持 UEFI 的，仍然使用传统的 BIOS；另外请使用 ufs, zfs 安装时会出错。

第五节 手动安装 FreeBSD

第六节 ee 用法及网络配置

概述

ee 用法

ee 的用法比 nano 还要简单许多。是系统自带的文本编辑器。

比如

```
# ee a.txt
```

按 ESC 键，会显示提示框，按两次回车即可保存。

网络配置

先 `ifconfig` 看看有没有网卡，没有那就不属于本节的范围之内了。
请注意 `lo0` 并不是真实网卡，如果你只能看到这个说明你网卡没有被正确驱动。

示例输出：

```
root@ykla:~ # ifconfig
genet0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0
mtu 1500

options=68000b<RXCSUM,TXCSUM,VLAN_MTU,LINKSTATE,RXCSUM_IPV6,TXCSUM_IP
V6>
    ether dc:a6:1a:2e:f4:4t
    inet 192.168.123.157 netmask 0xffffffff00 broadcast 192.168.123.255
    media: Ethernet autoselect (1000baseT <full-duplex>)
    status: active
    nd6 options=29<PERFORMNUD,IFDISABLED,AUTO_LINKLOCAL>
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> metric 0 mtu 16384
options=680003<RXCSUM,TXCSUM,LINKSTATE,RXCSUM_IPV6,TXCSUM_IPV6>
inet6 ::1 prefixlen 128
inet6 fe80::1%lo0 prefixlen 64 scopeid 0x2
inet 127.0.0.1 netmask 0xff000000
groups: lo
nd6 options=21<PERFORMNUD,AUTO_LINKLOCAL>
root@ykla:~ #
```

以下内容同时适用于虚拟机和物理机。

默认情况下，FreeBSD 是无法联网的，因为没有配置 DNS。

```
# ee /etc/resolv.conf
```

清空里面原有内容。添加以下内容。

```
nameserver 223.5.5.5 #阿里 DNS, 下同
nameserver 223.6.6.6
nameserver 8.8.8.8 #谷歌 DNS, 境外设备专用
```

之后重启一下网络配置

```
# /etc/netstart restart
```

尝试 ping 一下 163.com。 (按下ctrl + C 可中断)

示例输出：

```
root@ykla:~ # ping 163.com
PING 163.com (123.58.180.7): 56 data bytes
64 bytes from 123.58.180.7: icmp_seq=0 ttl=55 time=30.617 ms
64 bytes from 123.58.180.7: icmp_seq=1 ttl=55 time=30.608 ms
64 bytes from 123.58.180.7: icmp_seq=2 ttl=55 time=30.633 ms
^C
--- 163.com ping statistics ---
3 packets transmitted, 3 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 30.608/30.619/30.633/0.010 ms
root@ykla:~ #
```

网络联通。

详细用法

编辑后按 `ESC` 会弹出提示框，输入 `a` 保存；

- \ 或 [键 显示主选单。
- o 输入 ASCII code，例如输入 65 就会显示 A。
- u 跳到档案结尾。
- t 跳到档案开头。
- c 输入指令。在按了 `Ctrl+c` 后，上方选单会出现命令说明，例如您可以直接输入数字，表示将光标移到某一行。
- y 搜寻。按了 `Ctrl+y` 之后，你可以输入欲搜寻的字符串。如果要搜寻下一个该字符串，只要再按 `Ctrl+x` 即可。预设的搜寻是不分大小写的，如果要区分大小写，您可以按 `Ctrl+c` 并输入 `case` 即可。如果要取消只要再按 `Ctrl+c` 并输入 `nocase`。
- a 跳到行首。
- e 跳到行尾。
- d 删除光标所在位置的字符。
- j 贴上上一次所删除的字符。
- k 删除光标所在位置的一整行。
- l 贴上上一次删除的一整行内容。
- w 删除一个字。
- r 贴上上一次所删除的字。
- p 将光标移到上一行。
- n 将光标移到下一行。
- b 将光标移到上一个字，和方向键左键一样。

- f 将光标移到下一个字，和方向键右键一样。
- g 下一页。
- v 上一页。
- z 移到下一个字。
- 离开 ee。如果文件有修改过，它会问您是否要储存文件。

第七节 常用软件与 SSH 配置

WinSCP 下载

WinSCP 是对 `scp` 命令的图形化封装，同时支持 FTP 等多种协议。可以快捷的传输文件与 Windows 系统和 Linux 或 BSD 之间。

下载地址：

<https://winscp.net/eng/download.php>

Xshell 下载

Xshell 是 Windows 平台上的强大的 shell 工具，不建议使用苦难哲学的 putty 。

下载地址（输入用户名和邮件即可）：

<https://www.netsarang.com/zh/free-for-home-school>

配置 SSH

允许 root ssh

```
# ee /etc/ssh/sshd_config      # (删去前边的 #, 并将 yes 或 no 修改为如下)
PermitRootLogin yes          #允许 root 登录
PasswordAuthentication yes    # (可选) 设置是否使用普通密码验证, 如果不设置此参数则使用 PAM 认证登录, 安全性更高
```

提示：删去前边的 `#` 是什么意思？`#` 在 UNIX 当中一般是起到一个注释作用，相当于 C 语言里面的 `//`。意味着后边的文字只起到说明作用，不起实际作用。

开启 SSH 服务

```
# service sshd restart
```

如果提示找不到 `sshd`，请执行下一命令：

```
# sysrc sshd_enable="YES"
```

然后再

```
# service sshd restart
```

保持 SSH 在线

服务端设置：

编辑 `# ee /etc/ssh/sshd_config`，调整 `ClientAlive` 的设置：

```
ClientAliveInterval 10  
ClientAliveCountMax 3
```

10 秒给客户端发一次检测，客户端如果 3 次都不回应，则认为客户端已断开连接。

`ClientAliveInterval` 默认是 `0`，表示禁用检测。

客户端设置：

全局用户生效：`# ee /etc/ssh/ssh_config`，仅对当前用户生效：`~/.ssh/config`。

```
Host *  
ServerAliveInterval 10  
ServerAliveCountMax 3
```

或者在连接的时候使用 `-o` 指定参数：

```
# ssh user@server -p 22 -o ServerAliveInterval=10 -o  
ServerAliveCountMax=3
```

客户端和服务端任一开启检测即可。

SSH 密钥登录

生成密钥

```
# ssh-keygen
```

OpenSSH 7.0 及以上版本默认禁用了 ssh-dss(DSA) 公钥算法。
FreeBSD 13.0 采用 OpenSSH_7.9。因此使用默认值即可。

```
root@ykla:~ # ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa): #此处回车
Created directory '/root/.ssh'.
Enter passphrase (empty for no passphrase):      #此处输入密码（为了安全
建议设置密码）
Enter same passphrase again:      #此处重复输入密码
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:MkcEjGhWCv6P/8y62JfbpEws9OnRN1W0adxmpceNny8 root@ykla
The key's randomart image is:
+---[RSA 2048]---+
| . o.o...     .. |
| ..+.. .     o+* |
| +. .       o*B |
| . .       o=. |
| . .o S     . . |
| + o+o     . . |
| . o *.o o   E . |
| + Bo= . . . |
| . ==O..     |
+---[SHA256]---+
root@ykla:~ #
```

配置密钥

检查权限（默认创建的权限如下）：

```
drwx----- 2 root  wheel  512 Mar 22 18:27 /root/.ssh #权限为 700  
-rw----- 1 root  wheel  1856 Mar 22 18:27 /root/.ssh/id_rsa #私  
钥, 权限为 600  
-rw-r--r-- 1 root  wheel  391 Mar 22 18:27 /root/.ssh/id_rsa.pub #公  
钥, 权限为 644
```

生成验证公钥：

```
# cat /root/.ssh/id_rsa.pub >> /root/.ssh/authorized_keys  
-rw-r--r-- 1 root  wheel  391 Mar 22 18:39  
/root/.ssh/authorized_keys #检查权限 644
```

使用 winscp 把私钥和公钥保存到本地后，删除服务器上的多余文件：

```
# rm /root/.ssh/id_rsa*
```

修改 `/etc/ssh/sshd_config`

```
# ee /etc/ssh/sshd_config
```

修改配置如下（删去前边的 #，并将 yes 或 no 修改为如下）：

```
PermitRootLogin yes          #允许 ROOT 用户直接登陆系  
统  
AuthorizedKeysFile      .ssh/authorized_keys #修改使用用户目录下密钥文  
件， 默认已经正确配置， 请检查  
PasswordAuthentication no    #不允许用户使用密码方式登  
录  
ChallengeResponseAuthentication no #禁止密码登录验证  
PermitEmptyPasswords no      #禁止空密码的用户进行登录
```

重启服务

```
# service sshd restart
```

使用 xshell 登录即可， 输入密钥密码， 导入私钥 `id_rsa`， 即可登
录。

| 如果使用其他 ssh 软件无法登陆请自行转换密钥格式。

第八节 物理机安装与硬件选配

物理机安装添加

注意：请不要问我在安装中应该选用哪个镜像站这类问题，那是因为你的错误操作导致的。不要选用带有 `bootonly` 字样的镜像文件，除非你知道自己在干什么。

使用 光盘 安装应选用 `iso` 结尾的镜像。

使用 U盘 安装应该选用 `img` 结尾的镜像，例如

<https://download.freebsd.org/ftp/releases/amd64/amd64/ISO-IMAGES/13.0/FreeBSD-13.0-RELEASE-amd64-memstick.img>

FreeBSD 所有安装介质包括不限于虚拟机文件都没有提供图形界面，需要自行安装。

注意：如果虚拟机要使用 UEFI，必须使用 FreeBSD 13.0 及以上，否则启动会花屏。

刻录工具 Windows 应该选用 Rufus，Linux 直接使用 `dd` 命令即可。

<https://rufus.ie/zh>

注意：不建议使用 Handbook 推荐的 win32diskimager，有时会出现校验码错误的情况（实际上文件校验码正常）。也不要使用 vventory 引导实体机安装，有时会报错找不到安装文件。

硬件选配（以下硬件均正常运行）

更多硬件请参考：

<https://bsd-hardware.info/?d=FreeBSD>

1. 小米笔记本 12.5 一代：处理器 6Y30、显卡 HD515、WIFI intel 8260AC、声卡 ALC 233（实际上是 235）、硬盘 NVME INTEL 600P。
2. 联想 G400：处理器 i3-3110M/i5-3230M、显卡 HD4000、WIFI intel N135（联想 G400 网卡白名单支持三种网卡，如果是博通 BCM43142 建议更换为 N135，FUR 料号：04W3783）。

网卡推荐

以下无利益关系。

类型	品牌/型号	芯片组/参数	售价(¥)
USB 无线网卡	COMFAST CF-WU810N	RTL8188EUS 150M 2.4G	20
USB 以太网卡	绿联 USB 百兆网卡 CR110	AX88772A 100M	40
USB 以太网卡	绿联 USB 千兆网卡 CM209	AX88179A 1000M	79
USB 以太网卡	绿联 USB 2.5G 网卡 CM275	RTL8156 2.5G	189
Type-C 以太网卡	绿联 Type-C 转百兆网卡 30287	AX88772A 100M	59
Type-C 以太网卡	绿联 Type-C 转千兆网卡 CM199	AX88179A 1000M	99
Type-C 以太网卡	绿联 Type-C 转 2.5G 网卡	RTL8156 2.5G	199

第九节 物理机下显卡的配置

FreeBSD 已从 Linux 移植了显卡驱动，理论上，A 卡 N 卡均在 AMD64 架构上正常运行。

支持情况

对于 FreeBSD 11，支持情况同 Linux 内核 4.11；

对于 FreeBSD 12，支持情况同 Linux 内核 4.16；

对于 FreeBSD 13/14-current，编译使用 `drm-devel-kmod`，支持情况同 Linux 5.7.19，可以支持 Intel 第十二代处理器，AMD 可支持 R7 4750U。

详细情况可以看

<https://wiki.freebsd.org/Graphics>

英特尔核显 / AMD 独显

安装驱动

注意，如果要通过 `ports` 安装提示需要源码，请见第二十一章。

- FreeBSD 12.0: `#pkg install drm-fbsd12.0-kmod`

注意：除了 12.0，对于任意 12.X 均应该安装 `drm-fbsd12.0-kmod`，但应该使用 `port` 在本地重新构建而不应该使用 `pkg` 进行安装，否则不会正常运行。

- FreeBSD 13: `# cd /usr/ports/graphics/drm-devel-kmod/ && make BATCH=yes install clean`

FreeBSD 13.0 也可以使用二进制包进行安装 `# pkg install drm-fbsd13-kmod`，配置方法同“加载显卡”。13.X 需要通过 `ports` 编译安装，但是不如直接编译 `drm-devel-kmod`。因为后者支持的显卡更多。

- FreeBSD 14: `# cd /usr/ports/graphics/drm-devel-kmod/ && make BATCH=yes install clean`

故障排除：如果提示 `/usr/ports/xxx no such xxx` 找不到路径，请先获取 `portsnap`: `portsnap fetch extract`。`portsnap` 换源问题请看 第三章 第二节。

加载显卡

打开 `/etc/rc.conf` :

- 如果为 intel 核芯显卡, 添加 `kld_list="i915kms"`
- 如果为 HD7000 以后的 AMD 显卡, 添加 `kld_list="amdgpu"` (大部分人应该使用这个, 如果没用再去使用 `radeonkms`)
- 如果为 HD7000 以前的 AMD 显卡, 添加 `kld_list="radeonkms"` (这是十余年前的显卡了)

视频硬解

```
# pkg install xf86-video-intel libva-intel-driver
```

亮度调节

通用

一般计算机:

```
# sysrc -f /boot/loader.conf acpi_video="YES"
```

对于 Thinkpad:

```
# sysrc -f /boot/loader.conf acpi_ibm_load="YES"
# sysrc -f /boot/loader.conf acpi_video="YES"
```

仅限 FreeBSD 13

```
# backlight decr 20 #降低 20% 亮度
```

英特尔

```
# pkg install intel-backlight  
# intel-backlight 80 #调整为 80% 亮度
```

英伟达显卡

```
# pkg install nvidia-driver nvidia-settings nvidia-xconfig #安装几个  
nvidia 相关的包  
# sysrc kld_list+="nvidia-modeset" #配置驱动  
# reboot #重启
```

这时候应该已经可以驱动显卡了。

```
# 查看驱动信息  
$ nvidia-smi
```

如果发现系统没有使用 nvidia 驱动需要自动生成配置文件：

```
# Xorg -configure #生成配置文件。注意，该步骤不是必要！  
# cp /root/xorg.conf.new /etc/X11/xorg.conf
```

然后重新启动就可以发现正常使用 nvidia 驱动了

注意： 默认情况下，通过 pkg 安装的 nvidia-driver 是包含 Linux 兼容层支持的，如果要使用 Linux 软件，需要执行以下命令，（实际上使用 linux 兼容层，以下命令是必须的。） 如果不需要使用 Linux 兼容层，则不需要执行。

```
# sysrc linux_enable="YES"
```

当然如果使用官方的 pkg 包，安装好驱动重启后：

```
$ kldstat
```

会发现系统自动加载了 `linux.ko` 模块。如果觉得太臃肿，不需要 Linux 兼容层 可以自己通过 ports 编译 `nvidia-driver`，去掉 `linux compatibility support`。

第十节 物理机下触摸板的设置

第十一节 声卡与网卡设置

FreeBSD 声卡

声音设置

先加载声卡驱动:

```
# sysrc snd_hda="YES"
```

然后重启。

用以下命令查看当前声卡设备

```
$ cat /dev/sndstat
Installed devices:
pcm0: <NVIDIA (0x0083) (HDMI/DP 8ch)> (play)
pcm1: <NVIDIA (0x0083) (HDMI/DP 8ch)> (play)
pcm2: <NVIDIA (0x0083) (HDMI/DP 8ch)> (play)
pcm3: <NVIDIA (0x0083) (HDMI/DP 8ch)> (play)
pcm4: <Realtek ALC892 (Rear Analog 5.1/2.0)> (play/rec) default
pcm5: <Realtek ALC892 (Front Analog)> (play/rec)
pcm6: <Realtek ALC892 (Rear Digital)> (play)
No devices installed from userspace.
```

后面带有 default 是 oss 默认设备。如果软件的音频使用的 oss 且输出是默认的，音频就会从这个设备输出。

FreeBSD 大部分软件的音频输出驱动为 oss。有些默认是 pulseaudio(比如 firefox) , 这些软件的设置看最后的提示。

下列命令可以修改输出的设备。最后的数字是对应的pcm后面的数字。

```
$ sysctl hw.snd.default_unit=5
```

这里推荐几个 oss mixer:

GUI环境	名称
kde5	audio/dsbmixer
gtk	audio/gtk-mixer
非图形化	audio/mixertui

提示

但是 oss 有些缺点，使用 `obs-studio` 无法录制 oss 输出。只能录制 oss 输入。看官方论坛里，可以 `virtual_oss` 模拟一个设备实现。

但是 `obs-studio` 可以录 pulseaudio 输出的音频。

所以有些软件可以使用 pulseaudio 作为输出。使用 pulseaudio 的软件的音频输出, 不受上面的命令控制音频输出设备。pulseaudio 会根据自己的设置把音频送到对应设备，所以需要使用 pulseaudio 混音器控制。

在 kde5 下面自带的音频控制器，切换设备就是控制的 pulseaudio。

官方打包好的多媒体软件有些是支持 pulseaudio 但是这些软件中的大部分对应的编译选项没有打开。如果需要录制软件的音频输出，可以自行打开 ports 的编译选项自己编译。在软件中设置 pulseaudio 作为音频驱动输出就可以了

第十二节 打印机的安装

本过程环境使用的是 **KDE5 桌面系统及 HP LaserJet Pro MFP M126nw 多功能激光打印机**（如果是其它型号的惠普打印机需在添加打印机时能找到对应的型号的驱动就能使用），并且已连入局域网实现网络打印。

安装CUPS(通用 Unix 打印系统)

```
# pkg install cups
```

或

```
# cd /usr/ports/print/cups  
# make install clean
```

请在界面中选中 `x11`，此选项可在 kde 5 桌面系统中生成添加和配置打印机的应用。

添加服务

```
# sysrc cupsd_enable="YES"
```

完成后启动 cups 服务，执行如下命令

```
# service cupsd restart
```

安装打印机驱动

```
# pkg install hplib
```

添加打印机

在浏览器中输入 `http://localhost:631` , 该地址为该打印机的管理页面

点击顶部菜单的 `Administration` 进行添加打印机

或者在 kde 5 的系统设置->打印机中添加

第十三节 无线蓝牙鼠标的设置

本文以 FreeBSD 13.0 为基准，使用罗技 m337。

```
# sysrc hcsecd_enable="YES"
# sysrc bthidd_enable="YES"
# service hcsecd start
# service bthidd start
```

使用 `bluetooth-config` 工具添加蓝牙设备即可。

蓝牙鼠标调到配对模式，运行 `# bluetooth-config scan`，按提示信息进行添加：

```
# bluetooth-config scan
Scanning for new Bluetooth devices (Attempt 1 of 5) ... done.
Found 1 new bluetooth device (now scanning for names):
[ 1] 34:88:5d:12:34:56 "Bluetooth Mouse M336/M337/M535" (Logitech-M337)
Select device to pair with [1, or 0 to rescan]: 1

This device provides human interface device services.
Set it up? [yes]:
```

注意： logitech m337 配对连接后会自动断开。解决方案：删除 `/var/db/bthidd.hids` 文件中对应鼠标
的 `bd_addr` 行 `xx:xx:xx:xx:xx:xx`，重启 bthidd 服务 `# service bthidd restart`

第〇节 包管理器概述

FreeBSD 包管理器设计理念

熟悉 Linux 的人也许会发现，FreeBSD 的包管理方案实际上大约等于以下两大 Linux 发行版包管理器的完美合体：

Arch Linux：Pacman，对应 Pkg（同样秉持 KISS 的理念）

Gentoo Linux：Portage，对应 Ports（Portage 本身就是 Ports 的仿制品）

第一节 FreeBSD 镜像站现状

现状

主要问题在于官方无论如何也不开放 rsync 且不接受镜像站的官方二级镜像申请。

多次联系均无二次联系，如邮件列表，大概五次，其中三次回应，两次无回应。其主要回复内容为“深表歉意，但台湾地区已有镜像”。并未直接说明如何镜像，此外特别向科大 Linux 协会(其中其他镜像站并未理会，如清华大学 TUNA 协会)申请镜像，对方提到，FreeBSD 也是无人回应。中国大陆目前没有 FreeBSD 官方镜像站。

如有朋友们能够联系 FreeBSD 官方，还望早日开放镜像，1ftp 不能解决问题。此外，Kernel 或者 Base system 源码 SVN 速度更加感人，除非安装系统的时候安装源码，否则……国内网络环境如此，提升速度采取代理方式也是基本功，但是，不能够要求每个人都一样，提供便捷的网络服务，方便更多人的使用，才是发展 FreeBSD 的核心要义。

请朋友们注意这一点，镜像站是基础设施，就像那句话，“要想富，先修路”，如果通往 FreeBSD 的康庄大道不通，那就全是荆棘的小道。

在此号召能够联系到 FreeBSD 官方的朋友们，首先解决这一基本问题。

目前开放的非官方 issue 镜像申请:

USTC:

<https://github.com/ustclug/mirrorrequest/issues/172>

<https://github.com/ustclug/mirrorrequest/issues/171>

目前已经关闭的非官方 issue 镜像申请:

TUNA:

<https://github.com/tuna/issues/issues/16>

官方给出的镜像站基本要求

- 服务器的 root 权限，这一点上国内的大学开源镜像站不会给与；
- IPv6 及 CN2 网络——国内也很缺乏；
- BGP 网络；
- 足够的存储空间（约50TB）和 1G 带宽；
- 上述计算机 5 台。
- 备案问题——需要专门公司组织才能给 cn.FreeBSD.org 备案；
- 还有最大的问题，没有钱；

非官方镜像站

FreeBSD 在中国大陆境内没有非官方镜像站；

FreeBSD 目前在大陆非官方镜像站有若干个（详见第二节。）：

USTC

<https://mirrors.ustc.edu.cn>

仅 pkg ports

ChinaFreeBSD

<http://chinafreebsd.cn/article/chinafreebsd-resouce>

或者 freebsd.cn

北京交通大学自由与开源镜像站（四类源

mirror.bjtu.edu.cn

联系方式：

<https://t.me/bjtmirror>

网易 163 镜像站（仅 ports 源

FreeBSD 官方联系方式：

freebsd-hubs@freebsd.org

第二节 FreeBSD 换源方式

通知：

bjtu 北京交通大学开源镜像站镜像站故障。请切换其他源，例如 freebsd.cn/ustc/nju/163。

原因是学校要求保证网课，所以出口限速了100Mbps。现在该镜像站每天都很拥挤。

预计修复：放暑假，具体未知。

FreeBSD 有四类源，pkg、ports、portsnap、update。

对于失去安全支持的版本，请参考最后一节

本文对于一个源列出了多个镜像站，无需全部配置，只需选择其一即可。推荐使用北京交通大学自由与开源软件镜像站，因为是反向代理的方式镜像的，可以保持与上游同步。

目前境内没有官方镜像站，以下均为非官方镜像站。

pkg 源: pkg 源提供二进制安装包

pkg 的下载路径是 `/var/cache/pkg/`

FreeBSD 中 pkg 源分为系统级和用户级两个源。不建议直接修改 `/etc/pkg/FreeBSD.conf`，因为该文件会随着基本系统的更新而发生改变。

创建用户级源目录：

```
# mkdir -p /usr/local/etc/pkg/repos
```

北京交通大学自由与开源软件镜像站

创建用户级源文件：

```
# ee /usr/local/etc/pkg/repos/bjtu.conf
```

写入以下内容：

```
bjtu: {
    url: "pkg+http://mirror.bjtu.edu.cn/reverse/freebsd-
        pkg/${ABI}/quarterly",
    mirror_type: "srv",
    signature_type: "none",
    fingerprints: "/usr/share/keys/pkg",
    enabled: yes
}
FreeBSD: { enabled: no }
```

故障排除

若要获取滚动更新的包, 请将 `quarterly` 修改为 `latest`。请注意,
`CURRENT` 版本只有 `latest`。

若要使用 `https`, 请先安装 `security/ca_root_nss`, 并将 `http` 修改
为 `https`, 最后使用命令 `# pkg update -f` 刷新缓存即可, 下同。

网易开源镜像站

创建用户级源文件:

```
# ee /usr/local/etc/pkg/repos/163.conf
```

写入以下内容:

```
163: {
url: "pkg+http://mirrors.163.com/freebsd-pkg/${ABI}/quarterly",
mirror_type: "srv",
signature_type: "none",
fingerprints: "/usr/share/keys/pkg",
enabled: yes
}
FreeBSD: { enabled: no }
```

中国科学技术大学开源软件镜像站

创建用户级源文件：

```
# ee /usr/local/etc/pkg/repos/ustc.conf
```

写入以下内容：

```
ustc: {
url: "pkg+http://mirrors.ustc.edu.cn/freebsd-pkg/${ABI}/quarterly",
mirror_type: "srv",
signature_type: "none",
fingerprints: "/usr/share/keys/pkg",
enabled: yes
}
FreeBSD: { enabled: no }
```

南京大学开源镜像站

```
# ee /usr/local/etc/pkg/repos/nju.conf
```

写入以下内容：

```
nju: {  
    url: "pkg+http://mirrors.nju.edu.cn/freebsd-pkg/${ABI}/quarterly",  
    mirror_type: "srv",  
    signature_type: "none",  
    fingerprints: "/usr/share/keys/pkg",  
    enabled: yes  
}  
FreeBSD: { enabled: no }
```

FreeBSD.cn(非官方, 下同)

```
# ee /usr/local/etc/pkg/repos/freebsdcn.conf
```

写入以下内容:

```
freebsdcn: {
    url: "pkg+http://pkg.freebsd.cn/${ABI}/quarterly",
    mirror_type: "srv",
    signature_type: "none",
    fingerprints: "/usr/share/keys/pkg",
    enabled: yes
}
FreeBSD: { enabled: no }
```

ports 源：提供源码方式安装软件的包管理器

ports 下载路径是 `/usr/ports/distfiles`

北京交通大学自由与开源软件镜像站

创建或修改文件 `# ee /etc/make.conf`：

写入以下内容：

```
MASTER_SITE_OVERRIDE?=http://mirror.bjtu.edu.cn/reverse/freebsd-
pkg/ports-distfiles/
```

网易开源镜像站

创建或修改文件 `# ee /etc/make.conf`：

写入以下内容：

```
MASTER_SITE_OVERRIDE?=http://mirrors.163.com/freebsd-ports/distfiles/
```

中国科学技术大学开源软件镜像站

创建或修改文件 `# ee /etc/make.conf`：

写入以下内容：

```
MASTER_SITE_OVERRIDE?=http://mirrors.ustc.edu.cn/freebsd-
ports/distfiles/
```

FreeBSD.cn

创建或修改文件 `# ee /etc/make.conf` :

写入以下内容:

```
MASTER_SITE_OVERRIDE?=http://ports.freebsd.cn/ports-distfiles/
```

portsnap 源：打包的 ports 文件

北京交通大学自由与开源软件镜像站

编辑portsnap配置文件 `# ee /etc/portsnap.conf` :

将 `SERVERNAME=portsnap.FreeBSD.org` 修改为 `SERVERNAME=freebsd-portsnap.mirror.bjtulug.org`

获取portsnap更新

```
# portsnap fetch extract
```

故障排除

```
Snapshot appears to have been created more than one day into the
future!
(Is the system clock correct?)
Cowardly refusing to proceed any further.
```

需要同步时间。

```
ntpdate ntp.api.bz
```

FreeBSD. cn

编辑portsnap配置文件 `# ee /etc/portsnap.conf` :

将 SERVERNAME=portsnap.FreeBSD.org 修改为

SERVERNAME=portsnap.FreeBSD.cn

freebsd-update 源: 提供基本系统更新

注意: 只有一级架构的 release 版本才提供该源。也就是说 current 和 stable 是没有的。 关于架构的支持等级说明请看:

<https://www.freebsd.org/platforms>

北京交通大学自由与开源软件镜像站

编辑 `# ee /etc/freebsd-update.conf` 文件:

将 `ServerName update.FreeBSD.org` 修改为 `ServerName freebsd-update.mirror.bjtulug.org`

例: 从 FreeBSD 12 升级到 13.0

```
# freebsd-update -r 13.0-RELEASE upgrade
```

FreeBSD.cn

编辑 `# ee /etc/freebsd-update.conf` 文件:

将 `ServerName update.FreeBSD.org` 修改为 `ServerName update.FreeBSD.cn`

不受安全支持的版本（请酌情使用）

不受安全支持的版本也是可以使用二进制源的。

以下，以 FreeBSD 9.2 为例：

首先切换成可以用的二进制源

```
# setenv PACKAGESITE http://ftp-archive.freebsd.org/pub/FreeBSD-
Archive/ports/amd64/packages-9.2-release/Latest
```

如果 shell 不是 csh，那么：

```
# export PACKAGESITE=http://ftp-archive.freebsd.org/pub/FreeBSD-
Archive/ports/amd64/packages-9.2-release/Latest
```

安装示例：现在安装 bsdinfo。

```
root@ykla:~ # pkg_add -r bsdinfo
Fetching http://ftp-archive.freebsd.org/pub/FreeBSD-
Archive/ports/amd64/packages-9.2-release/Latest/bsdinfo.tbz... Done.
```

pkg 是不可用的，会提示找不到 digests.txz 和 repo.txz，因为当时 pkgng 还没有被官方所支持，仍然仅支持使用 pkg_* 命令。

第三节 gitup 的用法

在 FreeBSD 13.0，FreeBSD 官方准备将 `portsnap` 移除（但仍可使用），转而使用 `gitup`，换用 git 方式获取系统源代码和 ports 打包套件。

```
# pkg install gitup #安装 gitup
# gitup ports #获取 ports
# gitup release #获取 release 版本的源代码
```

故障排除：速度太慢

设置 HTTP 代理

`gitup` 的代理不取决于系统代理，而是由其配置文件单独决定。

```
# ee /usr/local/etc/gitup.conf
```

示例（先删去前边的 # 再改）

```
"proxy_host" : "192.168.27.1",
"proxy_port" : 7890,
```

第四节 软件包管理器 pkg 的用法

请注意：pkg 只能管理第三方软件包，并不能起到升级系统，获取安全更新的作用。这是因为 FreeBSD 项目是把内核与用户空间作为一个整体来进行维护的，而不是像 Linux 那样 linus torvalds 负责维护内核，各个发行版的人负责维护 GNU 工具（他们这些软件实际上被设计为单个软件包，因此可以用包管理器更新与升级系统）。FreeBSD 使用 `freebsd-update` 来升级系统，获取安全补丁。

如何用 pkg 安装软件

装上系统默认没有 pkg，先获取 pkg:

```
# pkg 回车即可输入 y 确认下载
```

pkg 使用 https，先安装 ssl 证书:

```
# pkg install ca_root_nss
```

然后把 repo.conf 里的 pkg+http 改成 pkg+https 即可。

最后刷新 pkg 数据库:

```
# pkg update -f
```

安装 python 3:

```
# pkg install python
```

pkg 升级:

```
# pkg upgrade
```

错误: You must upgrade the ports-mgmt/pkg port first

解决：

```
# cd /usr/ports/ports-mgmt/pkg  
# make deinstall reinstall
```

如何卸载软件

直接使用 `pkg delete` 会破坏正常的依赖关系，应该尽量避免使用（ports 的 `make deinstall` 也一样），转而使用 `pkg-rmleaf` 命令，该命令属于的软件需要自行安装：

```
# pkg install pkg-rmleaf
```

故障排除

FreeBSD pkg 安装软件时出现创建用户失败解决

问题示例：

```
[1/1] Installing package...
====> Creating groups.
Creating group 'package' with gid '000'.
====> Creating users
Creating user 'package' with uid '000'.
pw: user 'package' disappeared during update
pkg: PRE-INSTALL script failed
```

问题解析：数据库未同步

问题解决：

```
# /usr/sbin/pwd_mkdb -p /etc/master.passwd
```

Shared object "x. so. x" not found, required by
"xxx"

出现该问题一般是由于 ABI 破坏，更新即可。

```
# pkg install bsdadminscripts
# pkg_libchk
# port-rebuild
```

Newer FreeBSD version for package pkg

问题示例：

```
Neuer FreeBSD version for package pkg:  
To ignore this error set IGNORE_OSVERSION=yes  
- package: 1402843  
- running kernel: 1400042  
Ignore the mismatch and continue? [y/N]:
```

这通常发生在失去安全支持的或者在 Current 版本的系统上，不影响使用，输入 `y` 即可。

如果想要从根源上解决，需要自己卸载 pkg，从 ports 安装 `ports-mgmt/pkg`；或者从源代码更新整个系统。

如果只是不想看到这个提示只需要按照提示将 `IGNORE_OSVERSION=yes` 写到 `/etc/make.conf` 里面（没有就新建）。

第五节 通过源代码 port 方式安装软件

FreeBSD ports 基本用法（仅限 FreeBSD 13.0 以前，不含 13.0）

首先获取 portsnap

```
# portsnap fetch extract
```

使用 whereis 查询软件地址

如 # whereis python

输出 python: /usr/ports/lang/python

如何安装 python3:

```
# cd /usr/ports/lang/python
# make BATCH=yes clean
```

其中 BATCH=yes 的意思是使用默认配置

如何使用多线程下载:

```
# pkg install axel #下载多线程下载工具#
```

新建或者编辑 `# ee /etc/make.conf` 文件，写入以下两行：

```
FETCH_CMD=axel  
FETCH_BEFORE_ARGS= -n 10 -a  
FETCH_AFTER_ARGS=  
DISABLE_SIZE=yes
```

进阶

如果不选择 `BATCH=yes` 的方法手动配置依赖：

看看 python 的 ports 在哪：

```
# whereis python  
# python: /usr/ports/lang/python
```

安装python3：

```
# cd /usr/ports/lang/python
```

如何设置全部所需的依赖：

```
# make config-recursive
```

如何一次性下载所有需要的软件包：

```
# make BATCH=yes fetch-recursive
```

升级 ports collection

```
# portsnap fetch extract
```

ports 编译的软件也可以转换为 pkg 包

```
# pkg create nginx
```

FreeBSD 包升级管理工具

首先更新 Ports 树

```
# portsnap fetch update
```

然后列出过时 Ports 组件

```
# pkg_version -l '<'
```

下边分别列出 2 种 FreeBSD 手册中提及的升级工具：

一、portupgrade

```
# cd /usr/ports/ports-mgmt/portupgrade && make install clean
# portupgrade -ai #自动升级所有软件
# portupgrade -R screen #升级单个软件
```

二、portmaster (推荐)

```
# cd /usr/ports/ports-mgmt/portmaster && make install clean
# portmaster -ai #自动升级所有软件
# portmaster screen #升级单个软件
# portmaster -a -m "BATCH=yes" #或者-D -G -no-confirm 都可以免除确认
```

FreeBSD ports 多线程编译

Linux 如 gentoo上一般是直接 `-jx` 或者 `jx+1`，`x` 为核心数。

FreeBSD ports 多线程编译

```
FORCE_MAKE_JOBS=yes  
MAKE_JOBS_NUMBER=4
```

写入 `/etc/make.conf` 没有就新建。

`4` 是处理器核心数，不知道就别改。

第六节 通过 DVD 安装软件

第〇节 概述

所有 FreeBSD 安装介质默认均不包含图形界面，需要手动安装。
请勿使用 `sysutils/desktop-installer`，会引发不必要的错误和问题。

本章内容并非是让大家把所有的桌面都安装一遍，而是尽可能多地提供选择。

安装桌面的基本步骤是：① 安装显卡驱动 -> ② 安装 Xorg/Wayland -> ③ 安装 KDE5/Gnome3/Xfce/MATE -> ④ 安装显示管理器 sddm/lightdm/slim -> ⑤ 安装输入法等软件

其中，Gnome3 可省略第四步，因为其显示管理器 gdm 早在第二步就进行了安装。

目前支持 Wayland 的桌面经测试只有 Gnome3，还是建议使用 Xorg。

显示管理器推荐搭配是：

KDE5 + sddm

Xfce/Mate + lightdm

Slim 由于作者早在 2013 年就停止了开发，不推荐使用，会产生一些奇怪的 bug（比如 fcitx5 用不了，加载不了 dbus）。

输入法框架目前推荐使用 fcitx（对于 KDE 5 桌面）、ibus（对于其他基于 GTK 的桌面，如 gnome、xfce、mate……）。请勿使用 scim，作者早就跑路（大概已经距今 16 年了）。对于不同的 SHELL，环境变量的配置方法是不一样的，FreeBSD 默认使用 `csh`，而且不同桌面加载环境变量的方法也是不一样的，所以针对不同桌面，不同 SHELL 的配置方法是不一样的，具体方法请看具体桌面。

第一节 安装 Xorg

安装xorg

可选包：

xorg 完整包： xorg

xorg 最小化包： org-minimal

安装

通过pkg安装

```
# pkg install xorg
```

通过ports安装

```
# cd /usr/ports/x11/xorg
# make install clean
```

第二节 安装 KDE 5

以下教程适用于 shell 为 csh/tcsh 的用户。

首先看看现在自己的 shell 是不是 csh/tcsh

```
# echo $0
```

如果是 csh/tcsh 其中之一，请继续。

安装

```
# pkg install xorg sddm kde5 wqy-fonts xdg-user-dirs
```

上面的命令分别安装了桌面、窗口管理器和中文字体以及创建用户目录的工具。

配置

```
# ee /etc/fstab
```

添加内容如下：

proc	/proc	procfs	rw	0	0
------	-------	--------	----	---	---

添加 proc 挂载这一步是非常必要的，如果不添加会导致桌面服务无法正常运行，部分组件无法加载！

然后

```
sysrc dbus_enable="YES"
sysrc sddm_enable="YES"
```

然后

```
# echo "exec ck-launch-session startplasma-x11" > ~/.xinitrc
```

如果你在 root 下已经执行过了，那么新用户仍要再执行一次才能正常使用（无需 root 权限或 sudo 等） `startx`。

提示：hal 已经被删除。不需要再添加`hal_enable="YES"`，见：

<https://www.freshports.org/sysutils/hal>

注意：如果 sddm 登录闪退到登录界面，请检查左下角是不是 plasma-X11，闪退的一般都是 Wayland！因为目前 FreeBSD 上的 KDE 5 尚不支持 Wayland。

中文化

点击开始-> System Settings -> Regional Settings 在 Language 项的 Available Languages 栏中找到 “简体中文” 单击 > 将其加到 Preferred Languages 栏中，然后单击 Apply 按钮；再到 Formats 项，将 Region 文本框中的内容修改为 “中国-简体中文(zh-CN)”，单击 Apply 按钮，logout（注销）后重新登录，此时系统语言将变为中文。

第三节 安装 Gnome

注意：截止到 2022-3-21，由于版本号的迁移，`Gnome` 在不同版本的 `pkg` 源里的存在形式也是不一样的：在 `quarterly` 中，为 `x11/gnome3`（版本 3.36）；在 `latest` 中，为 `x11/gnome`（版本 41），见

<https://www.freshports.org/x11/gnome/>

安装

注意：以下安装二选一。如果你分不清 quarterly 和 latest 源的区别请看 第三章 第二节。这里我建议使用 latest 源，因为可以保持最新。

latest 源 (gnome 41)

```
# pkg install xorg gnome wqy-fonts xdg-user-dirs
```

解释：|包|用途| |:---:|:---:| |xorg|X11| |gnome3|Gnome3 主程序| |wqy-fonts|文泉驿中文开源字体| |xdg-user-dirs|用于创建用户家目录的子目录|

quarterly 源 (gnome 3.36)

```
# pkg install xorg gnome3 wqy-fonts xdg-user-dirs
```

配置

```
# ee /etc/fstab
```

添加内容如下：

```
proc /proc procfs rw 0 0
```

配置启动项：

```
# sysrc dbus_enable="YES"
# sysrc gdm_enable="YES"
# sysrc gnome_enable="YES"
```

输入以下命令：

```
% echo "/usr/local/bin/gnome-session" > ~/.xinitrc
```

中文化 GNOME

本小节用户 shell 应该是默认的 sh，判断方法见下一小节“安装输入法”（root 用户默认 shell 是 csh，无法适用）。

```
# cd /usr/local/etc/gdm && ee locale.conf
```

添加以下内容

```
LANG="zh_CN.UTF-8"
LC_CTYPE="zh_CN.UTF-8"
LC_MESSAGES="zh_CN.UTF-8"
LC_ALL="zh_CN.UTF-8"
```

安装输入法

以下 `ibus` 、 `fcitx5` 二选一即可。

ibus

gnome 捆绑的输入法面板是 `ibus`。

```
# pkg install zh-ibus-libpinyin (安装好运行初始化命令 ibus-setup )
```

fcitx 5

首先看看现在自己的 shell 是不是 `sh`，`bash`，`zsh`：

```
# echo $0
```

如果是 `sh`，`bash`，`zsh` 其中之一，请继续；如果不是，请参考第五章第一节。

安装 `fcitx5`：

```
# pkg install fcitx5 fcitx5-qt fcitx5-gtk fcitx5-configtool zh-fcitx5-chinese-addons
```

打开或新建文件 `~/.xprofile`，写入：

```
export GTK_IM_MODULE=fcitx
export QT_IM_MODULE=fcitx
export XMODIFIERS=@im=fcitx
```

参考：以下是该文件的一个示例：

```
# $FreeBSD$  
#  
# .profile - Bourne Shell startup script for login shells  
#  
# see also sh(1), environ(7).  
#  
  
# These are normally set through /etc/login.conf. You may override  
them here  
# if wanted.  
#  
PATH=/sbin:/bin:/usr/sbin:/usr/bin:/usr/local/sbin:/usr/local/bin:$HO  
ME/bin; export PATH  
  
# Setting TERM is normally done through /etc/ttys. Do only override  
# if you're sure that you'll never log in via telnet or xterm or a  
# serial line.  
# TERM=xterm;      export TERM  
  
EDITOR=vi;          export EDITOR  
PAGER=less;         export PAGER  
  
# set ENV to a file invoked each time sh is started for interactive  
use.  
ENV=$HOME/.shrc; export ENV  
  
# Let sh(1) know it's at home, despite /home being a symlink.  
if [ "$PWD" != "$HOME" ] && [ "$PWD" -ef "$HOME" ] ; then cd ; fi  
  
# Query terminal size; useful for serial lines.  
if [ -x /usr/bin/resizewin ] ; then /usr/bin/resizewin -z ; fi  
  
# Display a random cookie on each login.  
if [ -x /usr/bin/fortune ] ; then /usr/bin/fortune freebsd-tips ; fi
```

```
export GTK_IM_MODULE=fcitx
export QT_IM_MODULE=fcitx
export XMODIFIERS=@im=fcitx
```

提示：如果要显示 fcitx 输入法面板，需要安装 gnome 插件 TopIconsFix，请勿安装 AppIndicator and KStatusNotifierItem Support，已知该插件与 fcitx5 相冲突，会造成输入法卡死。

该插件需要通过火狐浏览器进行安装：

```
# pkg install -y firefox chrome-gnome-shell
```

打开网站 <https://extensions.gnome.org/extension/1674/topiconsfix/> 即可安装插件。

终端显示中文(文件用户根目录)

```
# ee .cshrc
```

添加以下内容

```
setenv LANG zh_CN.UTF-8  
setenv LC_CTYPE zh_CN.UTF-8  
setenv LC_ALL zh_CN.UTF-8
```

优化系统

```
# pkg install gnome-tweaks
```

卸载自带游戏（可选）

```
# pkg delete gnome-2048 gnome-klotski gnome-tetravex gnome-mines  
gnome-taquin gnome-sudoku gnome-robots gnome-nibbles lightsoff tali  
quadrapassel swell-foop gnome-mahjongg five-or-more iagno aisleriot  
four-in-a-row
```

第四节 安装 Mate 桌面

以下教程适用于 shell 为 bash/sh/zsh 的用户。

首先看看现在自己的 shell 是不是 sh , bash , zsh :

```
# echo $0
```

如果是 sh , bash , zsh 其中之一, 请继续;

安装开始（主要程序）

```
# pkg install mate xorg wqy-fonts lightdm lightdm-gtk-greeter
```

```
# sysrc moused_enable="YES"
# sysrc dbus_enable="YES"
```

安装登陆管理器

Slim 和 Lightdm 任选其一，因为前者已经停止开发，故推荐使用 Lightdm。

安装 Lightdm

- `# pkg install lightdm lightdm-gtk-greeter`
- 在 `/etc/rc.conf` 中加入一行：`lightdm_enable="YES"`
- 在主目录 `.xinitrc` 文件内加入下面的行：

```
exec mate-session
```

显示中文桌面环境

默认是 csh，在 `.cshrc` 中添加如下内容：

```
setenv LANG zh_CN.UTF-8  
setenv LC_CTYPE zh_CN.UTF-8
```

输入法

```
# pkg install zh-ibus-libpinyin (安装好运行初始化命令 ibus-
setup )
```

设置输入法变量：

```
# ee .xinitrc
```

文件添加以下内容

```
export GTK_IM_MODULE=ibus
export XMODIFIERS=@im=ibus
export QT_IM_MODULE=ibus
ibus &
```

第五节 安装 Xfce

安装 xfce4

以下教程适用于 shell 为 bash/sh/zsh 的用户。

首先看看现在自己的 shell 是不是 sh , bash , zsh :

```
# echo $0
```

如果是 sh , bash , zsh 其中之一, 请继续;

通过 pkg 安装

```
# pkg install xorg lightdm lightdm-gtk-greeter xfce wqy-fonts
```

或

通过 ports 安装

```
# cd /usr/ports/x11-wm/xfce4  
# make install clean
```

启用 xfce

```
# echo "/usr/local/etc/xdg/xfce4/xinitrc" > ~/.xinitrc
```

或者

```
# echo "/usr/local/etc/xdg/xfce4/xinitrc" > ~/.xsession
```

根据条件使用

启动服务

```
# sysrc dbus_enable="YES"
# sysrc lightdm_enable="YES"
```

设置中文显示

在 `.xinitrc` 添加以下内容（但要在最前面才正常启用）
`export LANG=zh_CN.UTF-8`

可选配置

输入法

请检查自己的shell是不是 `sh`、`bash`、`zsh` 其中之一。

```
# echo $0
```

如果是以上三个 SHELL 之一，请继续，如果不是请参考第五章第一节：

```
# pkg install zh-fcith zh-fcith-configtool fcith-qt5 fcith-m17n zh-fcith-libpinyin
```

配置文件：

```
# ee ~/.xinitrc
```

在该文件中添加以下内容：

```
export XMODIFIERS="@im=fcith"
export XIM_PROGRAM="fcith"
export GTK_IM_MODULE="fcith"
fcith &
```

全局菜单（可选）

```
# pkg install xfce4-appmenu-plugin appmenu-gtk-module appmenu-registrar
$ xfconf-query -c xsettings -p /Gtk/ShellShowsMenubar -n -t bool -s true
$ xfconf-query -c xsettings -p /Gtk/ShellShowsAppmenu -n -t bool -s true
$ xfconf-query -c xsettings -p /Gtk/Modules -n -t string -s "appmenu-gtk-module"
```

故障排除

xfce 普通用户关机按钮灰色解决方案

```
# chown -R polkitd /usr/local/etc/polkit-1
```

即可解决 xfce4 普通用户关机按钮灰色的问题

FreeBSD 的 xfce 终端动态标题不显示问题

tcsh 配置：

```
home 目录创建 .tcshrc ,
```

写入以下配置

```
alias h history 25 alias j jobs -l alias la ls -aF alias lf ls -FA alias
1l ls -lAF setenv EDITOR vi setenv PAGER less switch ($TERM) case
"xterm*": set prompt="%{033]0;[%~007%}%" set filec set history = 1000
set savehist = (1000 merge) set autolist = ambiguous # Use history to aid
expansion set autoexpand set autorehash breaksw default: set prompt="%#"
breaksw endsw
```

第六节 安装 Cinnamon

以下教程适用于 shell 为 bash/sh/zsh 的用户。

首先看看现在自己的 shell 是不是 sh , bash , zsh :

```
# echo $0
```

如果是 sh , bash , zsh 其中之一, 请继续;

安装

```
# pkg install xorg lightdm lightdm-gtk-greeter cinnamon wqy-fonts
```

配置

```
# ee ~/.xinitrc
```

添加：

```
exec cinnamon-session
```

```
# ee /etc/fstab
```

添加：

```
proc /proc procfs rw 0 0
```

添加启动项：

```
# sysrc dbus_enable="YES"
# sysrc lightdm_enable="YES"
```

中文化

编辑 `etc/login.conf` :

找到 `default:\` 这一段，把 `:lang=C.UTF-8` 修改为 `:lang=zh_CN.UTF-8`。

刷新数据库：`# cap_mkdb /etc/login.conf`

第七节 安装 Lumin

安装

```
# pkg install lumina xorg  lightdm lightdm-gtk-greeter wqy-fonts
```

配置

```
# sysrc dbus_enable="YES"
# sysrc lightdm_enable="YES"
```

```
# ee ~/.xinitrc
```

添加：

```
exec lumina-desktop
```

中文化

设置完毕还是英语。原因未知，如果您知道，请提交 issue 或者 pull。

Desktop Settings ——> Localization ——> 全部调整为“简体中文”，然后“save”，退出登录，重启系统。

第六节 root 登录桌面

警告：鉴于部分用户希望 root 登录桌面，为贯彻自由精神撰写本章节。请注意 root 账户拥有最高权限，失误使用 root 账户可能破坏系统，因此用其登录图形界面存在极高的安全风险。以下内容请谨慎操作，风险自负。

lightdm

安装与配置：

```
# pkg install lightdm-gtk-greeter lightdm
```

首先设置启动服务：

```
# sysrc lightdm_enable="YES"
```

然后修改配置文件：

- 编辑 `# ee /usr/local/etc/lightdm/lightdm.conf` :

往下拉，找到 `greeter-show-manual-login=true` 移除前面的 `#`。该行会多次出现，第一次出现是为你介绍，请勿修改，而应该继续往下拉。

- 编辑 `# ee /usr/local/etc/pam.d/lightdm` :

注释 `account requisite pam_securetty.so` 这一行（往最前面加 `#`）

重启服务

```
# service lightdm restart
```

即可。

sddm

安装

```
# pkg install sddm  
# sysrc sddm_enable="YES"
```

更改 `/usr/local/etc/pam.d/sddm` 文件：

把 `include` 之后的 `login`，替换成 `system`，一共4个。

重启服务

```
# service sddm restart
```

之后就可以 root 登录 sddm了！

注意 sddm 左下角选项不能为 Wayland，应该是 Plasma-X11，目前 KDE 5 不支持 wayland，选错无法登陆！

再次警告：root 账户拥有最高权限，失误使用 root 账户可能破坏系统，因此用其登录图形界面存在极高的安全风险。

第七节 主题与美化

FreeBSD 安装桌面环境后，同其它 类Unix 系统一样，很多时候扑面而来的，是简单朴素的色调。这种未加修饰的设定，可能一时会让新用户无法接受。其实它背后的逻辑也很简单：一来是为了系统的稳定，二来众口难调，桌面系统的美化，涉及个人的审美。为了系统的美观，我们在这一节学习添加 主题 和 图标。

安装软件包

- [] 新手任务：从以下软件包中，各选一款主题和图标来安装。

注：本节仅涉及了 **GTK** 库的桌面主题，囊括了 **Gnome**、**Xfce**、**MATE**、**Cinnamon** 和 **LXDE** 等桌面环境。

以下仅收录了部分图标和主题，想要获取更多资源，可访问 [FreshPorts](#)。

主题

- matcha 主题：`# pkg install matcha-gtk-themes`
- Qogir 主题：`# pkg install qogir-gtk-themes`
- Pop 主题：`# pkg install pop-gtk-themes`
- Flat 主题：`# pkg install flat-remix-gtk-themes`
- Numix 主题：`# pkg install numix-gtk-theme`
- Sierra 主题：`# pkg install sierra-gtk-themes`
- Yaru 主题：`# pkg install yaru-gtk-themes`
- Canta 主题：`# pkg install canta-gtk-themes`

图标

- papirus 图标：`# pkg install papirus-icon-theme`
- Qogir 图标：`# pkg install qogir-icon-themes`
- Pop 图标：`# pkg install pop-icon-theme`

- Flat 图标: # pkg install flat-remix-icon-themes
- Numix 图标: # pkg install numix-icon-theme
- Numix 圆形图标: # pkg install numix-icon-theme-circle
- Yaru 图标: # pkg install yaru-icon-theme
- Canta 图标: # pkg install canta-icon-theme

终端模式安装

提示：新接触 Unix 的用户可略过本节

- [] 高阶任务：为 KDE 或 Gnome 安装一款仿 MacOS 系统样式的主题和图标。
- [x] 提前任务1 安装 bash: `# pkg install bash`
- [x] 提前任务2 安装 plank: `# pkg install plank`
- [x] 提前任务3 安装 git: `# pkg install git`

KDE 主题美化

我们要安装的是 [WhiteSur](#) 主题。

1. 下载主题源码包: `git clone https://github.com/vinceliuice/WhiteSur-kde`
2. 进入主题包目录: `cd WhiteSur-kde`
3. 修改 shebang: `ee install.sh`，修改第一行为
`#!/usr/local/bin/bash`，然后保存。
4. 执行安装: `bash install.sh`

Gnome 主题美化

同样我们要安装的是 [WhiteSur](#) 主题。

1. 下载主题源码包: `git clone https://github.com/vinceliuice/WhiteSur-gtk-theme`

2. 进入主题包目录: `cd WhiteSur-gtk-theme`
3. 修改 shebang: `ee install.sh`, 修改第一行为
`#!/usr/local/bin/bash`, 然后保存。
4. 执行安装: `bash install.sh`

图标

1. 下载图标: `git clone https://github.com/vinceliuice/WhiteSur-icon-theme`
2. 进入软件目录: `cd WhiteSur-icon-theme`
3. 修改 shebang: `ee install.sh`, 修改第一行为
`#!/usr/local/bin/bash`, 然后保存。
4. 执行安装: `bash install.sh`

光标

1. 下载光标: `git clone https://github.com/vinceliuice/McMojave-cursors`
2. 进入软件目录: `cd McMojave-cursors`
3. 修改 shebang: `ee install.sh`, 修改第一行为
`#!/usr/local/bin/bash`, 然后保存。
4. 执行安装: `bash install.sh`

背景图片

[下载地址](#)

课后练习

试着按照下面的步骤，在终端安装 [Papirus 图标](#):

```
git clone https://github.com/PapirusDevelopmentTeam/papirus-icon-theme  
cd papirus-icon-theme  
.install.sh
```

第八节 远程桌面管理

VNC

启用 VNC 服务

FreeBSD 操作系统的 VNC 服务可以使用 TigerVNC Server，在终端下执行命令进行安装。

```
# pkg install -y tigervnc-server
```

安装之后，还要做一些设置。

1. 在终端执行命令 `vncpasswd`，设置访问密码。

2. 创建 `~/.vnc/xstartup` 文件，内容如下：

```
#!/bin/sh
unset SESSION_MANAGER
unset DBUS_SESSION_BUS_ADDRESS
[ -x /etc/X11/xinit/xinitrc ] && exec /etc/X11/xinit/xinitrc
[ -f /etc/X11/xinit/xinitrc ] && exec sh /etc/X11/xinit/xinitrc
xsetroot -solid grey
$command &
```

注意: `$command` 需要替换, 请注意保留 `&`, 在不同桌面下需要替换, Gnome 用 `gnome-session`, KDE 用 `startplasma-x11`, MATE 用 `mate-session`, Xfce 用 `xfce4-session`。

保存后执行命令 `# chmod 755 ~/.vnc/xstartup`。

1. 接下来在终端执行命令 `vncserver` 或 `vncserver :1`。

其中“`:1`”相当于 `DISPLAY=:1`, 即指定桌面显示的通信端口为 `1`, 对应 VNC 服务的通信端口为 `5901`。尽管桌面显示通信端口是从 `0` 开始, 但该端口已被当前桌面占用, 因此 VNC 服务默认端口虽为 `5900`, 但实际执行往往从 `5901` 开始。

如果启动服务时不指定通信端口, 则系统根据使用情况自动指定。

关闭服务请用命令 `# vncserver -kill :1`, 这里通信端口必须指定。

1. 如果启用了防火墙, 那么此时还需要开通防火墙, 以 `ipfw` 为例, 在终端输入命令:

```
# ipfw add allow tcp from any to me 5900-5910 in keep-state
```

上行命令表示开通 `5900-5910` 的端口, 即 `DISPLAY` 的 `0-10` 端口, 通常情况下, 即便需要开启很多桌面, `10` 个端口也足够了。最后别忘了将指令加入规则集文件, 否则操作系统重启后会丢失。

XRDP

这里介绍一款软件 rdesktop。 安装命令：

```
# pkg install -y rdesktop
```

但 rdesktop 安装后不会在系统中生成菜单，因此要在终端输入命令：

```
# rdesktop windows 设备 ip
```

首次登陆设备会有安全提示，输入 `yes`，回车后远程桌面窗口就会弹出：

第九节 安装 Wayland (可选)

Wayland 是 Xorg 的替代品，使用上的不同就是相比 xorg 而已，前者界面渲染效果更好，更加流畅；触控板效果更加接近 macOS。

FreeBSD已经正式支持wayland，版本是 1.20。注意：请使用 latest 源安装，因为是最新提交的包。经过测试目前暂不支持任何桌面。

第一节 Fcith 输入法框架

fcitx 5 相比前一代，增加了对 Wayland 的支持，据说更加流畅。

注意，在 FreeBSD-14.0-Current 中会出现许多不可预料的奇怪的 bug (fcitx5 诊断信息英文乱码，输入法显示出奇怪的汉字，Fcitx5-qt5 环境不能正常加载……)，如果条件允许应该在 FreeBSD-Release 中参考使用本文。

Fcith 4. X

注意：该教程仅在 KDE 5/csh 下测试通过。

```
# pkg install zh-fcith zh-fcith-configtool fcith-qt5 fcith-m17n zh-fcith-libpinyin
```

在 `.cshrc` 和 `/etc/csh.cshrc` 中添加如下配置，此配置可以解决部分窗口 fcith 无效的问题。

```
setenv QT4_IM_MODULE fcith
setenv GTK_IM_MODULE fcith
setenv QT_IM_MODULE fcith
setenv GTK2_IM_MODULE fcith
setenv GTK3_IM_MODULE fcith
setenv XMODIFIERS @im=fcith
```

在 `.cshrc` 和 `/etc/csh.cshrc` 中添加下面两行配置可以解决终端无法输入中文和无法显示中文的问题。

```
setenv LANG zh_CN.UTF-8
setenv MM_CHARSET zh_CN.UTF-8
```

接Fcith 输入法补充：

```
#要想终端不乱码还需要添加:  
setenv LANG zh_CN.UTF-8  
setenv LC_CTYPE zh_CN.UTF-8  
setenv LC_ALL zh_CN.UTF-8
```

自动启动:

```
# cp /usr/local/share/applications/fcitx.desktop ~/.config/autostart/
```

Fcitx 5.X

注意：该教程仅在 KDE 5/csh 下测试通过。

```
# pkg install fcitx5 fcitx5-qt fcitx5-gtk fcitx5-configtool zh-fcitx5-rime zh-rime-essay zh-fcitx5-chinese-addons
```

也可通过 ports 安装。环境变量取决于你的窗口管理器和桌面以及 shell 。经测试不支持 slim，可能是配置问题。sddm 可用。

自动启动：

```
# cp /usr/local/share/applications/org.fcitx.Fcitx5.desktop  
~/.config/autostart/
```

在 `.cshrc` 和 `/etc/csh.cshrc` 中进行如下配置，此配置可以解决部分窗口 fcitx 无效以及无法输入显示中文的问题。

```
setenv QT4_IM_MODULE fcitx  
setenv GTK_IM_MODULE fcitx  
setenv QT_IM_MODULE fcitx  
setenv GTK2_IM_MODULE fcitx  
setenv GTK3_IM_MODULE fcitx  
setenv XMODIFIERS @im=fcitx  
setenv LANG zh_CN.UTF-8  
setenv MM_CHARSET zh_CN.UTF-8
```

在 root 用户下 rime 不会自动被添加到输入法，需要手动添加完成初始化（程序里找到 fcitx 配置工具，添加 rime 输入法即可）！对于普通用户如果未生效，请检查自己的 shell，应该是 csh，如果不

是请将该用户加入 wheel 组。对于其他 shell 请自行更正为对应 shell 的环境变量。

SLIM 窗口下会提示 IBUS 找不到……疑似bug。

普通用户设置

普通用户的默认 shell 一般不是 `csh`，为了方便配置，需要把默认 shell 改成 `csh`。然后其余配置方法同上所述。

先看看现在的 shell 是什么：`# echo $0`，如果输出不是 `csh`，尝试修改成 `csh`：

```
# chsh -s /bin/csh
```

退出当前账号，重新登录，查看 shell 是否变为 `csh`：

```
# echo $0
```

如果输出 `csh`，代表配置成功。然后其余环境变量配置方法同上所述。

提示：如果不使用 `csh`，把 `setenv` 等环境变量改为 `export` 形式亦可，例如 `export GTK_IM_MODULE=fcitx`。

故障排除

遇到问题，请先运行 `fcitx` 故障诊断，但是该输出仅对 `bash` 做了环境变量的配置。也就是说他输出的环境变量仅适用于 `bash`、`sh` 和 `zsh` 等 SHELL，而不适用于 `csh`。于 `csh` 的环境变量配置需要参考上文。

如果提示 `bash` 字样且无法输出诊断信息，则需要先安装 `bash`：
`# pkg install bash`

fcitx 4.x

```
# fcitx-diagnose
```

对于 `fcitx5.x` 来说，找不到 `GTK 4` 的支持是正常的。

fcitx 5.x

```
# fcitx5-diagnose
```

对于 `fcitx5.x` 来说，找不到 `fcitx qt 4` 的支持是正常的。

第二节 Ibus 输入法框架

注意：该教程仅在 XFCE 桌面下测试通过。不适用于KDE5。 ibus
输入法框架配置 `.xinitrc` 中增加

```
XIM=ibus;export XIM
GTK_IM_MODULE=ibus;export GTK_IM_MODULE
QT_IM_MODULE=xim; export QT_IM_MODULE
XMODIFIERS='@im=ibus'; export XMODIFIERS
XIM_PROGRAM="ibus-daemon"; export XIM_PROGRAM
XIM_ARGS="--daemonize -xim"; export XIM_ARGS
```

`.cshrc` 中增加

```
setenv LANG zh_CN.UTF-8
setenv LC_CTYPE zh_CN.UTF-8
setenv LC_ALL zh_CN.UTF-8
setenv XMODIFIERS @im=ibus
```

`.profile` 中添加

```
export LC_ALL=zh_CN.UTF-8
```

第三节 五笔输入法

FreeBSD 使用 98 五笔输入法教程

注意：以下全部教程可能仅适用于 GNOME 桌面

以下教程二选一。

ibus

```
# pkg install zh-ibus-rime
```

配置

环境变量配置：安装好运行初始化命令 `ibus-setup`，将 98 五笔码表（`wubi86.dict.yaml`、`wubi86.schema.yaml`）复制到 `/usr/local/share/rime-date` 目录下，修改 `rime-date` 目录下 `default.yaml` 文件：

打开 `default.yaml` 找到 `schema_lis`：

下面第一行添加 `- schema: wubi98` 保存退出重新加载 `ibus` 输入法即可。

98 五笔码表 下载地址

<https://gitee.com/ykla/free-bsd-98wubi-tables/tree/master>

fcitx5

请先看第四章 桌面安装——第三节安装Gnome3。

首先下载所需文件: <https://github.com/FreeBSD-Ask/98-input>

把 `98wbx.conf` 文件复制到

`/usr/local/share/fcitx5/inputmethod/` (`inputmethod` 目录) 下面 把
`fcitx-98wubi.png` 和 `org.fcitx.Fcitx5.fcitx-98wubi.png` 图标复制到
`/usr/local/share/icons/hicolor/48x48/apps/` (`apps`目录) 下面 把
`98wbx.main.dict` 词库放到 `/usr/local/share/libime/` (`libime` 目录)
下面 重启 `fcitx5`，在 `fcitx5-configtool` 起用98五笔即可

提示: 王码 98 五笔生成 `.dict` 库方法, 直接用下面命令生成:

```
$ libime_tabledict 98wbx.txt 98wbx.main.dict
```

第四节 Firefox 与 Chromium 安装

火狐浏览器

安装普通版本：

```
# pkg install firefox
```

或者

```
# cd /usr/ports/www/firefox  
# make install clean
```

安装长期支持版本：

```
pkg install firefox-esr
```

或者

```
#cd /usr/ports/www/firefox-esr/ && make install clean
```

Chromium

```
# pkg install chromium
```

或者

```
# cd /usr/ports/www/chromium  
# make install clean
```

第五节 Linux 兼容层

注意：一个常见误解就是把 FreeBSD 的 Linux 兼容层当做 Wine，认为这样做会降低软件的运行效率。实际情况是不仅不会慢，而且速度还会比在 Linux 中更快，运行效率更高。

系统自带；

以下参考

<https://handbook.freebsdcn.org/di-10-zhang-linux-er-jin-zhi-jian-rong-ceng/10.1.-gai-shu>

开启服务

```
# sysrc linux_enable="YES"
# sysrc kld_list+="linux linux64"
# kldload linux64
# pkg install emulators/linux-c7 dbus
# service linux start
# sysrc dbus_enable="YES"
# service dbus start
# dbus-uuidgen > /compat/linux/etc/machine-id
# reboot
```

配置 fstab

以下写入 `/etc/fstab` :

```
linprocfs  /compat/linux/proc    linprocfs    rw    0    0
linsysfs   /compat/linux/sys    linsysfs    rw    0    0
tmpfs      /compat/linux/dev/shm  tmpfs       rw,mode=1777  0    0
```

检查挂载有无报错：

```
# mount -al
```

```
# reboot
```

自己构建 Ubuntu 兼容层

以下教程仅在 FreeBSD 13.0 测试通过。构建的是 Ubuntu 20.04 LTS (18.04 亦可)。兼容层使用技术实际上是 Linux jail，并非 chroot。

需要先按照“系统自带”的方法配置好原生的 CentOS 兼容层。

更多其他系统请看 `/usr/local/share/debootstrap/scripts/`

将 `nullfs_load="YES"` 写入 `/boot/loader.conf`

开始构建

```
# pkg install debootstrap  
# debootstrap focal /compat/ubuntu http://mirror.bjtu.edu.cn/ubuntu/  
# reboot
```

挂载文件系统

将以下行写入 `/etc/fstab`：

# Device	Mountpoint	FStype	Options
Dump			
devfs	/compat/ubuntu/dev	devfs	rw,late
0	0		
tmpfs	/compat/ubuntu/dev/shm	tmpfs	
	rw,late,size=1g,mode=1777	0	0
fdescfs	/compat/ubuntu/dev/fd	fdescfs	
	rw,late,linrdlnk	0	0
linprocfs	/compat/ubuntu/proc	linprocfs	rw,late
0	0		
linskyfs	/compat/ubuntu/sys	linskyfs	rw,late
0	0		
/tmp	/compat/ubuntu/tmp	nullfs	rw,late
0	0		
/home	/compat/ubuntu/home	nullfs	rw,late
0	0		

检查挂载有无报错：

```
# mount -al
```

如果提示没有home文件夹，请新建：

```
# mkdir /home
```

Jail

首先 chroot 进去 Ubuntu，移除会报错的软件：

```
# chroot /compat/ubuntu /bin/bash  
# apt remove rsyslog # 此时已经位于 Ubuntu 兼容层了。
```

换源

在卸载 rsyslog 之后，换源，由于 SSL 证书没有更新，所以还不能用 https：

```
# ee /compat/ubuntu/etc/apt/sources.list #此时处于 FreeBSD 系统！因为  
Ubuntu 兼容层还没有文本编辑器。
```

写入：

```
deb http://mirror.bjtu.edu.cn/ubuntu/ focal main restricted universe  
multiverse  
deb-src http://mirror.bjtu.edu.cn/ubuntu/ focal main restricted  
universe multiverse  
deb http://mirror.bjtu.edu.cn/ubuntu/ focal-updates main restricted  
universe multiverse  
deb-src http://mirror.bjtu.edu.cn/ubuntu/ focal-updates main  
restricted universe multiverse  
deb http://mirror.bjtu.edu.cn/ubuntu/ focal-backports main restricted  
universe multiverse  
deb-src http://mirror.bjtu.edu.cn/ubuntu/ focal-backports main  
restricted universe multiverse  
deb http://mirror.bjtu.edu.cn/ubuntu/ focal-security main restricted  
universe multiverse  
deb-src http://mirror.bjtu.edu.cn/ubuntu/ focal-security main  
restricted universe multiverse
```

进入 Ubuntu 兼容层，开始更新系统，安装常用软件：

```
# apt update && apt upgrade && apt install nano wget # 此时已经位于  
Ubuntu 兼容层了。
```

运行 X11 软件

```
# xhost +local: #此时处于 FreeBSD 系统!
```

示例：运行 Chrome

```
# wget https://dl.google.com/linux/direct/google-chrome-  
stable_current_amd64.deb # 无需代理软件，可以直连。此时已经位于 Ubuntu  
兼容层了。  
# apt install ./google-chrome-stable_current_amd64.deb # 此时已经位于  
Ubuntu 兼容层了。
```

```
# /usr/bin/google-chrome-stable --no-sandbox --no-zygote --in-  
process-gpu # 此时已经位于 Ubuntu 兼容层了。
```

Systemd 不可用，但可以用 `server xxx start`。其他更多可以运行的软件见 <https://wiki.freebsd.org/LinuxApps>。

参考文献 <https://wiki.freebsd.org/LinuxJails>、
<https://handbook.freebsdcn.org/di-10-zhang-linux-er-jin-zhi-jian-rong-ceng/10.4.-shi-yong-debootstrap8-gou-jian-debian-ubuntu-ji-ben-xi-tong>。

类似的方法可以构建 Debian、Arch 兼容层（经测试会提示 内核太老，旧版本则强制升级无法使用）。Gentoo 兼容层则提示 bash so 文件错误，即使静态编译了 zsh。

导入过 <https://github.com/zq1997/deepin-wine> 源以安装 deepin-qq, deepin-wechat 等软件，但都提示段错误。所有 Wine 程序都无法正常运行。如果你能解决这个问题，请提出 issue 或者 pull。

第六节 安装 金山 WPS

金山 WPS 提供两个版本一是国际版，一是国内版本。国际版无中文支持。

二者的使用都需要先安装 Linux 兼容层，见 第六节。字体问题见 第八节

1、国内版

linux-wps-office-zh_CN

安装，目前只能通过 ports 安装：

```
# cd /usr/ports/chinese/linux-wps-office-zh_CN/ && make install clean  
#如要默认请添加 BATCH=yes
```

2、国际版

linux-wps-office

请注意：国际版的服务器在境外，境内下载速度非常慢，请参考 第七章 第一节

安装，目前只能通过 ports 安装：

```
# cd /usr/ports/editors/linux-wps-office/ && make install clean #如要  
默认请添加 BATCH=yes
```

第七节 安装 QQ

介绍

FreeBSD 安装 Linux QQ 方法

安装 Linux 兼容层：

请先安装 Linux 兼容层，具体请看 第五章 第五节。

```
# pkg install linux-c7-gtk2 linux-c7-libxkbcommon
```

下载 Linux QQ

```
# mkdir /home/work
# fetch https://down.qq.com/qqweb/LinuxQQ/linuxqq_2.0.0-b2-
1089_x86_64.rpm
```

提示：后续如果版本更新请自行前往

<https://im.qq.com/linuxqq/download.html>

手动下载。

安装 Linux QQ：

```
# pkg install archivers/rpm4
# cd /compat/linux
# rpm2cpio < /home/work/linuxqq_2.0.0-b2-1089_x86_64.rpm | cpio -id
```

下载并安装 Linux QQ 所需依赖

由于未知原因，安装的 Linux QQ 无法输入，需要安装以下依赖才可以输入文字，但是只摸索了 Fcith 输入法框架下的依赖。

```
# cd /home/work
# fetch http://mirror.centos.org/centos/7/os/x86_64/Packages/gtk2-
immodule-xim-2.24.31-1.el7.x86_64.rpm
# fetch https://download-
ib01.fedoraproject.org/pub/epel/7/x86_64/Packages/f/fcitx-gtk2-
4.2.9.6-1.el7.x86_64.rpm
# fetch https://download-
ib01.fedoraproject.org/pub/epel/7/x86_64/Packages/f/fcitx-4.2.9.6-
1.el7.x86_64.rpm
# fetch https://download-
ib01.fedoraproject.org/pub/epel/7/x86_64/Packages/f/fcitx-libs-
4.2.9.6-1.el7.x86_64.rpm
```

然后分别安装以上 4 个包：

```
# cd /compat/linux
# rpm2cpio < /home/work/gtk2-immodule-xim-2.24.31-1.el7.x86_64.rpm | 
cpio -id
# rpm2cpio < /home/work/fcitx-gtk2-4.2.9.6-1.el7.x86_64.rpm | cpio - 
id
# rpm2cpio < /home/work/fcitx-4.2.9.6-1.el7.x86_64.rpm | cpio -id
# rpm2cpio < /home/work/fcitx-libs-4.2.9.6-1.el7.x86_64.rpm | cpio - 
id
```

注意：为了方便境内 FreeBSD 用户，可以使用境内的 gitee 同步下载以上 4 个文件；

```
# cd /home/work  
# pkg install git  
# git clone https://gitee.com/ykla/Linux-QQ.git
```

其余步骤自行参考。 境外用户可以使用 Github：

<https://github.com/ykla/FreeBSD-Linux-QQ>

刷新 gtk 缓存

```
# /compat/linux/usr/bin/gtk-query-immodules-2.0-64 --update-cache
```

运行 Linux QQ

```
$ /compat/linux/usr/local/bin/qq
```

第八节 更换字体

首先提取 `C:\Windows\Fonts` 里所有的 `.ttf` 和 `.ttc` 字体文件。

为新字体新建一个目录便于管理：

```
# mkdir -p /usr/local/share/fonts/WindowsFonts
```

将字体文件复制进 `WindowsFonts` 即可。

```
# chmod -R +755 /usr/local/share/fonts/WindowsFonts #刷新权限, 然后  
# fc-cache
```

即可。

第九节 wine

第十节 压缩与解压

zip

安装 zip 压缩文件 # pkg install zip unzip

```
# zip test.zip test # 打包 zip 格式文件
```

```
# unzip test.zip # 释放 zip 格式文件
```

tar/xz

```
# tar -cvf test.tar test # 打包 tar 格式文件

# tar -xvf test.tar # 释放 tar 格式文件

# tar -zcvf test.tar.gz test # 打包 gzip 格式文件

# tar -zxvf test.tar.gz # 释放 gzip 格式文件

# tar -jcvf test.tar.bz2 test # 打包 bzip2 格式文件

# tar -jxvf test.tar.bz2 # 释放 bzip2 格式文件

# tar -Jcvf test.tar.xz test # 打包 xz 格式文件

# tar -Jxvf test.tar.xz # 释放 xz 格式文件

# xz -z -k test.tar # 打包 xz 格式文件, 如不加-k 参数, 命令执行完原文件将被删除

# xz -d -k test.tar.xz # 释放 xz 格式文件, 如不加-k 参数, 命令执行完 xz 文件将被删除
```

7z/7za

FreeBSD 操作系统下，7z 和 7za 命令均通过 `# pkg install -y 7zip` 获取。

示例如下：

```
# 7z a test.7z test # 7z 打包文件  
# 7z x test.7z # 7z 释放文件  
# 7za a test.7z test # 7za 打包文件  
# 7za x test.7z # 7za 释放文件
```

第一节 UFS

第二节 ZFS

使用建议

- 最好不要在内存少于 1G 的机器上使用 ZFS。因为 ZFS 大约 每 1TB 硬盘空间需要消耗 1GB 内存。
- 最好不要在非 64 位系统上使用 ZFS。
- 为了提高机械硬盘随机读能力，可设置
`vfs.zfs.prefetch_disable=1`。
- 为了避免 ZFS 吃掉太多内存，最好要设置
`vfs.zfs.arc_max="?"`，例如：1024 M。
- 如果要复制某个文件系统，可以用 `zfs send/recv`，这样还能通过 ssh 跨网络传输。
- 推荐使用固态硬盘，使用 SSD 可以改善 ZFS 随机读能力，并且 ZFS 这种写时复制的文件系统也有益于 SSD 寿命。

注意事项

- ZFS 并不使用 `/etc/fstab`

第三节 磁盘扩容

扩容方法

1. `gpart show`

```
# gpart show
=> 63 209715137 vtbd0 MBR (100G)
 63 1 - free - (512B)
 64 62914496 ***1*** freebsd [active] (30G)
 62914560 146800640 - free - (70G)
```

查看系统盘大小只有 30G，显示 1 只有这一个盘。

1. 执行扩容命令，`vtbd0` 从 `gpart show` 执行后查看

```
# gpart resize -i 1 vtbd0
vtbd0s1 resized
```

1. 启动 `growfs` 服务，自动完成扩展

```
# service growfs onestart

Growing root partition to fill device
vtbd0s1 resized
gpart: arg0 'ufsid/62b5826d': Invalid argument
super-block backups (for fsck_ffs -b #) at:
64112192 65394432 66676672 67958912 69241152 70523392
```

1. 用 `df -h` 命令查看结果。

```
# df -h
Filesystem Size Used Avail Capacity Mounted on
/dev/ufsid/62b5826d 97G 15G 75G 16% /
devfs 1.0K 1.0K 0B 100% /dev
```

第四节 NTFS 的挂载

1. 安裝 ntfs-3g 软件 `# pkg install sysutils/fusefs-ntfs`
2. 把你的 ntfs 格式的硬盘或 U 盘插入计算机。这时候你会看到它的设备名，例如 `da0`。
3. 修改 `rc.conf`

```
sysrc kld_list+="fusefs"
```

1. 修改 fstab 自动挂载

为了开机自动挂载，修改添加

```
# ee /etc/fstab
/dev/da0s1  /media/NTFS  ntfs  rw,mount_prog=/usr/local/bin/ntfs-
3g,late  0  0
```

或者，手动挂载

```
# ntfs-3g  /dev/da0s1  /media/NTFS  -o
rw,uid=1000,gid=1000,umask=0`
```

如果不知道哪个磁盘分区是 NTFS，可以用命令来查看

```
# fstyp /dev/da0s1
```

详细参数见 [ntfs-3g manpage](#)。

第五节 SWAP 交换分区的设置

如果在安装系统的时候并未设置 swap 即交换分区，那么后期只能通过 dd 一个交换分区的文件来实现了。因为无论是 UFS 还是 ZFS 都是不支持缩小文件分区的……

dd 一个 大小为 1GB 的 swap 文件

```
# dd if=/dev/zero of=/usr/swap0 bs=1M count=1024
```

设置权限为 600，即只有拥有者有读写权限。

```
# chmod 0600 /usr/swap0
```

如果要立即使用

```
# mdconfig -a -t vnode -f /usr/swap0 -u 0 && swapon /dev/md0
```

为了重启后仍然有效，还需要往 `/etc/rc.conf` 中加入

```
swapfile="/usr/swap0"
```

第六节 Ext 2/3/4 等文件系统

EXT 文件系统

请注意：这里应该安装 `fusefs-ext2`（同时支持EXT2/3/4）而非 `fusefs-ext4fuse`，因为后者是只读且被废弃的。

- 安装 `fusefs-ext2`

```
pkg install fusefs-ext2
```

- 加载

打开 `/etc/rc.conf`，在 `kld_list` 一栏里添加 `ext2fs`，结果可能如 `kld_list="ext2fs i915kms"`

- 重启后，挂载。

对于用户名为 `XiaoMing` 的账号，可如下操作：

```
$ cd ~  
$ mkdir media  
$ cd media  
$ mkdir first  
# mount -t ext2fs /dev/da0sX /home/XiaoMing/media/first/
```

提示：上式不一定是 `da0sX` (X 为对应的阿拉伯数字)，可通过
`# gpart list` 命令查看硬盘名。

- 卸载硬盘

```
# umount /home/XiaoMing/media/first/
```

Brtfs/XFS 文件系统

未经测试

```
# pkg install fusefs-lkl
```

第一节 HTTP 代理

示例：V2ray 或 clash 开启允许局域网连接。然后按照具体配置如下：

```
# setenv http_proxy http://192.168.X.X:10809
```

取消：

```
# unsetenv http_proxy
```

第二节 V2ray

第三节 Clash

第四节 OpenVPN

第五节 Wireguard

WireGuard 是一个极其简单而又快速的现代 VPN，采用了最先进的加密技术。它的目标是比 IPsec 更快、更简单、更精简、更有用，同时避免了大量的头痛问题。它打算比 OpenVPN 的性能要好得多。

WireGuard 被设计成一个通用的 VPN，可以在嵌入式接口和超级计算机上运行，适合于许多不同的情况。它最初是为 Linux 内核发布的，现在是跨平台的（Windows、macOS、BSD、iOS、Android），可广泛部署。它目前正在大力发展，但已经被认为是业内最安全、最容易使用和最简单的 VPN 解决方案。

推荐 FreeBSD 13，安装：

```
# pkg install wireguard
```

第一节 sudo

安装

系统默认不提供 `sudo` 命令，需要使用 `root` 用户自行安装：

```
# pkg install sudo
```

sudo 免密码

在 `/usr/local/etc/sudoers.d/` 下新建两个文件 `username` (需要免密码的用户) `admin` 和 `wheel` :

`username` 内容如下:

```
%admin ALL=(ALL) ALL
```

`wheel` 内容如下, 多加一行 `NOPASSWD:` , 使用 `sudo` 时不需要输入密码:

```
%wheel ALL=(ALL) NOPASSWD:ALL
```

第二节 添加用户

创建一个普通用户（用户名 `ykla`），并默认添加到 `video` 分组：

```
# adduser -g video -s sh -w yes  
# Username: ykla
```

第三节 用户组

创建一个 `admin` 分组，并添加 `ykla` 和 `root` 两位用户：

```
# pw groupadd admin  
# pw groupmod admin -m ykla root
```

创建一个 `wheel` 分组，只添加 `root` 用户：

```
# pw groupadd wheel  
# pw groupmod wheel -m root
```

从 `admin` 组里移除用户 `ykla`：

```
# pw groupmod admin -d ykla
```

删除 `admin` 用户组：

```
# pw groupdel admin
```

`admin` 和 `wheel` 权限的区别：

- `admin`，具有管理系统的权限（`sudo` 的默认配置如此），可以使用 `sudo` 命令。

- `wheel`，超级管理员权限，可以任意修改系统（该名称来源于俚语 `big wheel`，意为大人物）。

用 `pw` 命令管理用户和组操作系统用户，或许叫账号更恰当，是操作人员登陆操作系统的凭证，如前面的自建用户；

而操作系统组，也可以简单理解为角色，限制了操作系统用户的权限。

在 FreeBSD 中，用户和组统一用 pw 命令管理，下面介绍几个常用的 pw 子命令：

1. useradd 命令，用于新建用户，常用参数：

-u，指定 uid，不指定则由操作系统根据已存在的 `uid` 自动生成。

在某些情况下，例如从测试场推送文件到生产环境，由于 `uid` 不同，可能导致系统认为文件归属不同用户，从而导致错误，这种情况事先指定统一的 `uid` 会好些。

-c，注释，也可以理解为用户全名，登录桌面后显示的用户名即注释

-d，指定主目录，不指定则默认 `/home/新建用户名`，如 `/home/ykla`

-g，指定起始组，或者说指定主要角色，如果不指定则默认为与新建用户同名的组

-G，指定附加组，不指定就没有

-m，指定创建用户同时创建主目录，不指定则默认不创建主目录

-s，指定用户登陆后的 shell 环境，不指定则使用 `/bin/sh`，FreeBSD 建议指定 `/bin/csh`，如果想禁止用户登陆，可以指定 `/usr/sbin/nologin`

-h，指定输入设备以非交互状况下设置密码，通常设置参数为 0，代表 `stdin`，即系统的标准输入

示例：

```
# pw useradd test1 #创建用户 test1, uid 系统默认, test1 组, 登陆环境/bin/sh, 主目录未创建
# pw useradd test2 -u 1200 -m -d /tmp/test -g test1 -G wheel -s csh -
c test2 #创建用户 test2, uid 为 1200, 创建主目录, 主目录为/tmp/test,
test1 组, 有管理员权限, 登陆环境/bin/csh, 全名 test2
# echo password | pw useradd test3 -h 0 #创建用户 test3, 同时设置密码为
password
```

1. `usermod` 命令, 用于修改用户信息, 常用参数:

`-l`, 为用户改名 其他参数参考 `useradd` 子命令。

示例:

```
# pw usermod test1 -G wheel #为用户 test1 增加管理员权限
# pw usermod test1 -l myuser #用户 test1 改名为 myuser
# echo password | pw usermod test2 -h 0 #修改用户 test2 密码为
password
```

1. `userdel` 命令, 用于删除用户, 常用参数:

`-r`, 删 除 用户 同时 删 除 用户 主 目 录 及 所 有 相 关 信 息, 不 使用 该 参数
则 信 息 保 留, 仅 删 除 用户

示例:

```
# pw userdel test2 -r
```

1. `usershow` 命令, 用于显示用户信息,

示例:

```
# pw usershow test2
```

1. `usernext` 命令, 返回下一个可用的 uid,

示例:

```
# pw usernext
```

1. `lock` 命令, 锁定账号, 锁定后账号无法登录使用,

示例:

```
# pw lock test2
```

1. `unlock` 命令, 解锁账号, 解锁后账号可以正常使用,

示例:

```
# pw unlock test2
```

1. `groupadd` 命令, 用于新建组, 常用参数:

-g, 指定 `gid`, 不指定则由操作系统根据已存在的 `gid` 自动生成

-M, 指定组成员列表, 多个用户用逗号隔开

示例:

```
# pw groupadd test -g 1200 #创建组 test, gid 为 1200, 注意, gid 与 uid  
不是一回事  
# pw groupadd test5 -M test1,test2 #创建组 test5, 成员有 test1 和  
test2
```

1. `groupmod` 命令, 用于修改组信息, 常用参数:

`-g`, 指定新的 `gid`

`-l`, 为组改名

`-M`, 替换现有组成员列表, 多个用逗号隔开

`-m`, 为现有组成员列表增加新的成员

其他参数参考 `groupadd` 命令。

示例:

```
# pw groupmod test -g 1300 #修改 test 组的 gid 为 1300  
# pw groupmod test -l mygroup 组 test 改名为 mygroup  
# pw groupmod test5 -M test1 #设置组 test5 的成员为 test1  
# pw groupmod test5 -m test3 #为组 test5 增加成员 test3
```

1. `groupdel` 命令, 用于删除组,

示例:

```
# pw groupdel mygroup
```

1. `groupshow` 命令, 用于显示组信息,

示例：

```
# pw groupshow test
```

1. `groupnext` 命令，返回下一个可用的 `gid`，

示例：

```
# pw groupnext
```

其他用户管理命令

1. `adduser` 命令，用于新建用户，与 `pw` 相比，`useradd` 的区别在于该命令是交互式的，安装操作系统时自建的用户，就是基于该命令创建的。
2. `rmuser` 命令，用于删除用户，与 `adduser` 命令一样，也是交互式的。不过该命令带 `-y` 参数，并允许列出用户列表，

示例：

```
# rmuser -y test1 test2 #同时删除用户 test1 和 test2,
```

`-y` 参数用于省略询问步骤

1. `chpass` 命令，以 `vi` 编辑器方式打开并修改指定用户信息，如不指定用户则默认为当前用户。

常用参数： `-s`，用于登录环境

示例：

```
# chpass -s csh test1 #更换用户 test1 的登陆环境为/bin/csh  
# chpass #以 vi 方式打开当前用户信息进行修改 d.passwd 命令，修改用户密码，如不指定用户则默认为当前用户。
```

示例：`passwd` 用户 #回车后根据系统提示设置用户密码

1. `id` 命令，查看用户 `id` 信息，包括用户 `id` 和起始组 `id`，

示例：

```
# id # 查看当前用户 id 信息  
# id test1 # 查看用户 test1 的 id 信息
```

1. `whoami` 命令，查看当前用户是谁。
2. `who` 命令，查看当前用户登陆信息。

第四节 用户权限

FreeBSD 文件访问权限可以用 10 个标志位来说明，10 个标志位由 4 部分组成：

第一部分是第 1 位，用 d 表示目录，- 表示普通文件，l 表示链接，b 表示块设备文件，p 表示管道文件，c 表示字符设备文件，s 表示套接字文件；

第二部分是第 2-4 位，用于标识文件所属用户对文件的访问权限，用 `rwx` 表示读、写、执行权限，没有权限就写成-；

第三部分是第 5-7 位，用于标识文件所属组成员对文件的访问权限，标识参考第二部分。

第四部分是第 8-10 位，用于标识文件其他用户对文件的访问权限，标识参考第二部分。

读、写、执行除了用 `rwx` 表示，也可以对应成数字 4、2、1，没有权限用 0，每部分的数字相加后，组合在一起，就是 3 位数字的表示方式。

如：

字符标识 权限	数字 标识 权限	说明
- rwxrwxrwx	777	所有人都可以读、写、执行的普通文件
drw----- -	600	目录，只有所属用户可以读、写
-rwxr-xr-x x	755	普通文件，所属用户有读、写、执行权限，同组用户和其他用户只能读取或执行

1. ls 命令，列出文件，常用参数：

-l，列表显示文件信息，信息中包含权限、属主、文件大小、修改日期和时间等信息，也可以直接用 ll 命令

-a，隐藏文件也要显示

示例：

```
# ls -l -a #列表显示包括隐藏文件在内的所有文件
# ll -a #与上例效果相同
# ls -l /tmp/a.log #列表显示/tmp/a.log 文件
```

1. chmod 命令，修改文件访问权限，有两种操作方式。

一种是操作符方式，如：

chmod a+x t.sh 在“a+x”中：

第一位表示操作对象， u 是所属用户， g 是所属组， o 是其他用户， a 则表示所有用户，不写按系统默认操作；

第二位是操作符，+是添加权限，-是减少权限；

第三位是 权限模式，`rwx` 分别表示读、写、执行，还有 `s` 表示文件执行时把进程属主或组置成与文件属主或组一致，这种方式使用起来比较直观。

另一种是数字方式，如：`chmod 750 t.sh`

其中 7 表示所属用户拥有读、写、执行的权限，同组用户有读和执行的权限，其他用户则没有任何权限，这种方式使用起来比较方便。

然后 `chmod` 还有一个重要的参数：

`-R`，递归赋权

示例：

```
# chmod -R 777 /tmp #/tmp 目录下所有文件将允许任何用户读、写、执行  
# chmod -R a+rwx /tmp #/tmp 目录下所有文件将允许任何用户读、写、执行
```

1. `chown` 命令，修改文件的属主，包括所属用户和所属组。

常用参数：

`-R`，递归赋权

示例：

```
# chown test1 t.sh #修改 t.sh 属主为用户 test1  
# chown test1:test t.sh #修改 t.sh 属主为用户 test1、组 test  
# chown -R test1:test /tmp #修改/tmp 目录下所有文件的属主为用户 test1、  
组 test
```


第一节 jail 与 docker 的比较

[第一章 第五节](#) 提及过 “Jail 与 byhve 虚拟化，不必配置底层虚拟化，节约系统资源。”

第二节 jail 相关术语

第三节 jail 配置

创建jail目录

放入基本系统

方案一

```
# make buildworld #编译基本系统  
# make installworld DESTDIR=/usr/jail/ #安装到 jail  
# make distribution DESTDIR=/usr/jail/ #或者
```

方案二

下载 base.txz 或者从 iso 提取 baes.txz，然后解压到 jail

```
# tar -xvf base.txz -C /usr/jail/
```

挂载 devfs 文件系统。(不是必须)

```
# mount -t devfs devfs /usr/jail/dev
```

写入 `rc.conf`

```
# sysrc jail_enable="YES"
```

创建 `jail.conf` 文件(可以写进 `rc.conf` 但这样便于管理)

```
www {  
host.hostname =www.example.org; # 主机名  
ip4.addr = 192.168.0.10; # IP 地址  
path ="/usr/jail"; # jail位置  
devfs_ruleset = "www_ruleset"; # devfs ruleset  
mount.devfs; # 挂载 devfs文件系统到jail  
exec.start = "/bin/sh /etc/rc"; # 启动命令  
exec.stop = "/bin/sh /etc/rc.shutdown"; # 关闭命令  
}
```

管理

jails 查看在线 jail 信息列表

JID	IP Address	Hostname	Path
3	192.168.0.10	www	/usr/jail/www

中英对照

英语	中文
JID	jail ID
IP Address	IP地址
Hostname	主机名
Path	Jail 路径

启动与停止jail

```
# service jail start www  
# service jail stop www
```

登录jail

```
# jexec 1 tcsh
```

干净关闭jail

```
# jexec 3 /etc/rc.shutdown
```

升级jail

```
# freebsd-update -b /here/is/the/jail fetch  
# freebsd-update -b /here/is/the/jail install
```

ping与网络

开启ping

写入 `/etc/jail.conf`

```
allow.raw_sockets=1;
allow.sysvipc=1;
```

网络

创建 `/etc/resolv.conf`，并编辑

```
search lan
nameserver 119.29.29.29
nameserver 182.254.116.116
nameserver 114.114.114.114
nameserver 223.5.5.5
nameserver 223.6.6.6
#不要写路由器地址
```

创建 jail 目录

创建4个 分别是模板 骨架 数据 项目

创建模板目录

```
# mkdir -p /jail/j1  
#然后放入基本目录，上边说过不再写
```

创建骨架目录

```
# mkdir -p /jail/j2  
#移动目录 etc usr tmp var root
```

```
# cd /jail/j2/ # 注意目录  
# mv /jail/j1/etc ./etc  
# mv /jail/j1/tmp ./tmp  
# mv /jail/j1/var ./var  
# mv /jail/j1/root ./root
```

创建数据目录

就是复制一份骨架给他用

```
# cp -R /jail/j2/ /jail/js/www/
```

创建项目目录

```
# mkdir -p /jail/www/
```

建立链接

```
# cd /jail/j1 #cd 到模板目录  
# mkdir -p jusr #创建用来做链接数据的目录  
# ln -s jusr/etc etc  
# ln -s jusr/home home  
# ln -s jusr/root root  
# ln -s jusr/usr usr  
# ln -s jusr/tmp tmp  
# ln -s jusr/var var  
#链接目录，注意链接的目录
```

创建fstab

```
#ee /jail/www.fstab  
#将公共只读系统挂载到项目目录  
/jail/j1/ /jail/www/ nullfs ro 0 0  
#将项目数据目录挂载到项目目录  
/jail/js/www/ /jail/www/jusr/ nullfs ro 0 0
```

创建fstab

```
# ee /jail/www.fstab
#将公共只读系统挂载到项目目录
/jail/j1/ /jail/www/ mullfs ro 0 0
#将项目数据目录挂载到项目目录
/jail/js/www/ /jail/www/jusr/ mullfs ro 0 0
awk
```

写入 jail.conf

```
#全局部分

exec.start = "/bin/sh /etc/rc";
exec.stop = "/bin/sh /etc/rc.shutdown";
exec.clean;
mount.devfs;
allow.raw_sockets = 1;
allow.sysvipc = 1;
```

#网关 没用就不写

```
interface = "网卡地址"; #主机名也可以用变量代替
```

```
hostname = "$name.domain.local";
#jail 位置, 也可以用变量
path = "/jail/$name";
```

#ip地址

```
ip4.addr = 192.168.1.$ip;
```

#fstab位置

```
mount.fstab = /jail/www.fstab;
www {
$ip=2
#不使用fstab, 使用
#mount.fstab = "";
```

```
#替换全局  
}
```

删除文件没有权限

```
# chflags -R noschg directory
```

第四节 jail 更新

第五节 使用 ezjail 管理 jail

第一节 虚拟化简介

第二节 安装 Virtual Box

第三节 安装 bhyve

第四节 使用 cbsd 管理 bhyve

第五节 使用 bhyve 安装 Windows 11

第六节 安装 XEN

第七节 使用 XEN 安装 Windows 11

第一节 通过 freebsd-update 更新

注意：只有一级架构的 release 版本才提供该源。也就是说 current 和 stable 是没有的。关于架构的支持等级说明请看：
<https://www.freebsd.org/platforms>

前排提示：阿里云用户请注意，目前不支持从 12.1 升级到任一版本，因为 <https://reviews.freebsd.org/D27262>

更新系统

FreeBSD 提供了实用工具 `freebsd-update` 来安装系统更新，包括升级到大版本。

常规的安全更新：

```
# freebsd-update fetch  
# freebsd-update install
```

小版本或者大版本更新，`13.0` 是要更新到的版本号：

```
# freebsd-update upgrade -r 13.0-RELEASE  
# freebsd-update install
```

安装后需要重启系统：

```
# reboot
```

然后再继续完成安装：

```
# freebsd-update install
```

故障排除

FreeBSD 升级出错，没有 ntp 用户

终端执行命令

```
# pw groupadd ntpd -g 123
# pw useradd ntpd -u 123 -g ntpd -h - -d /var/db/ntp -s
/usr/sbin/nologin -c "NTP Daemon"
```

第二节 通过源代码更新

第三节 批量部署

第一节 概述

FreeBSD中的磁盘加密功能

GBDE (基于 GEOM 的磁盘加密)

FreeBSD 5, 2003

Poul-Henning Kamp

内核中的 GEOM 模块 gbde(4)

用户空间工具 gbde(8)

创建后缀为 .bde 的新设备

GELI (GEOM eli)

FreeBSD 6, 2005

Paweł Jakub Dawidek

内核中的 GEOM 模块

用户空间工具 geli(8)

创建后缀为 .eli 的新设备

在扇区级操作

创建新的设备以允许对数据进行纯文本访问

GEOM 框架

访问存储层的标准化方式

FreeBSD 5, 2003

Poul-Henning Kamp

GEOM 类的集合

类可以以任何顺序自由堆叠

I/O 请求转换的抽象化

变换：条带化、镜像、分区、加密

提供者和消费者

自动发现

GBDE

主密钥（2048 个随机位）位于 GEOM 设备的一个随机位置在 GEOM 设备上，其位置存储在一个锁文件中。

锁定文件使用用户密码进行加密，并且应该 应单独存储

最多可以有 4 个独立的用户秘密(锁定扇区)

每个扇区使用 **AES-CBC-128** 和一个随机的扇区密钥进行加密。扇区密钥

扇区密钥使用从主密钥和扇区号中提取的密钥进行加密。扇区密钥使用由主密钥和扇区编号衍生的密钥进行加密

存储每个扇区密钥的磁盘空间开销

非原子性的磁盘更新，因为扇区密钥是与数据分开存储的 因为扇区密钥与数据分开存储

不支持在/文件系统中安装加密的设备系统中的加密设备

GELI

简单的扇区对扇区加密

为了对扇区进行对称加密，选择一个随机的主密钥

主密钥使用用户密钥进行加密，并存储在 GEOM 设备的最后一个扇区中

主密钥的最多两个加密副本可以存储在扇区中

用户密钥由最多两个部分组成：一个用户口令和一个密钥文件

口令使用 PKCS #5：基于密码的密码学规范 2.0 (Password-Based Cryptography Specification 2.0) 来加强

加密技术规范 2.0 (RFC 2898)

可以对数据完整性进行验证

由于利用了 crypto(9) 框架，自动利用硬件加速加密操作的优势

支持多种加密算法（AES-XTS, AES-CBS, Blowfish-CBC, Camellia-CBC, 3DES-CBC）和不同的密钥长度。不同的密钥长度

允许在/文件系统中挂载加密的设备

自 FreeBSD 11 支持从加密的分区启动

GEOM 模块化磁盘变换框架及其他磁盘管理常用命令：

fdisk -s /dev/da0 #打印磁盘对象汇总信息。其中/dev/da0 即磁盘对象，表示本机的第一块硬盘，如果不写 默认显示启动盘信息。还可以写成分片或分区，如/dev/da0s1 和/dev/da0s1a，其中硬盘用 da 表示，从 0 起算，分片用 s 表示，从 1 起算，分区则用字母 a-h 表示，/dev/da0s1a 即表示第一块硬盘第一个分片的第一个分区，这是 MBR 的表示方法。GPT 由于没有分片的概念，直接就是分区，因此用 p 表示分区，从 1 起算，/dev/da0p1 即表示第一块硬盘的第一个分区

```
# dd if=/dev/zero of=/dev/da1 bs=1k count=1 #清理磁盘信息

# fdisk -BI /dev/da1 #初始化磁盘，默认 MBR 模式 bslabel -w /dev/da1s1
#写入 bslabel

# bslabel -e /dev/da1s1 #用 vi 编辑器编辑

# bslabel geom -t #树状结构显示磁盘对象关系

# geom disk lsit #列表显示已使用的物理磁盘

# geom disk status #显示已使用的物理磁盘状态信息

# gpart list | geom part list #列表显示已创建的分片和分区

# gpart status | geom part status #显示已创建的分片和分区状态信息

# gpart show /dev/da1 #显示已使用的硬盘信息

# gpart create -s GPT /dev/da1 #为磁盘/dev/da1 创建分区表，本例为 GPT
模式，还可以设置 MBR、APM、BSD、BSD64、LDM、VTOC8

# gpart add -b 64 -s 2048m -t freebsd-ufs -i 2 -l root0 /dev/da1 #在
磁盘/dev/da1 上创建新分区。-b 表示起始位 置；-s 表示分配空间；-t 为分区格
式，分片时可以用 freebsd，还有 freebsd-boot、freebsd-swap、freebsdzfs
等类型；-i 表示索引，本例为 2，即新分区名为/dev/da1p2；-l 为标签 newfs
/dev/da1p2 #格式化分区
```

```
# gpart modify -i 2 -t freebsd-zfs -l myroot /dev/da1 #在磁盘/dev/da1  
上修改索引为 2 的分区，分区格式和标签均可修改
```

```
# gpart resize -i 2 -s 4g /dev/da1 #在磁盘/dev/da1 上调整索引为 2 的分  
区大小，单位可以用 k、m、g、t。注意，如果要缩小分区，则分区不能处于使用状  
态，这意味着系统分区默认情况下无法缩小；如果要扩大 分区，则分区后面必须是空  
闲空间，而不能有其他分区，这意味着系统分区默认情况下也无法扩展。因 此在创建  
FreeBSD 虚机时，应充份考虑可能使用系统盘的情况，或尽量避免使用系统盘
```

```
# gpart bootcode -b /boot/mbr /dev/da1 #写入启动代码，常用的还  
有/boot/gptboot 和/boot/boot
```

```
# gpart set -a active -i 1 /dev/da0 #设置活动分片。分区表为 MBR 时，  
bsdinstall 和 sade 会自动把新建的分片设置为活动分片，从而导致操作系统重启  
时无法正确加载启动分区，故需要重设
```

```
# gpart delete -i 2 /dev/da1 #在磁盘/dev/da1 上删除索引为 2 的分区
```

```
# gpart destroy -F /dev/da1 #销毁磁盘/dev/da1 上的信息，-F 参数表示强制
```

```
# mount /dev/da1p1 /data #将分区/dev/da1p1 挂载到/data 目录，挂载后注意  
用 chown 命令设置归属，若希望重启后自动挂载，请在终端执行命令：
```

```
# printf "/dev/da1p1t/datattufstrwt0t0n" >> /etc/fstab  
# umount /data #卸载/data 目录上的挂载
```

下面再给出四组示例，谨供参考：

```
#1.MBR 在系统盘扩展分片后新建分区(假设已为系统盘增加 50G 磁盘空间)
```

```
# gpart resize -i 1 -s 149g /dev/da0 #调整分片/dev/da0s1 的空间为  
149G。尽管磁盘的大小为 150G，但由于技术原因，实际可使用的空间并没有那么多
```

```
# gpart add -t freebsd-ufs /dev/da0s1 #在分片/dev/da0s1 上添加分区，类  
型 freebsd-ufs。不指定-s 参数时，表示将 剩余空间都分配给该分区
```

```
# newfs /dev/da0s1d #格式化新分区。这里注意新分区名称，由于 a 是启动分  
区，b 是 swap 分区，c 已经被分 片本身占用，因此新分区默认分配为 d
```

```
# mkdir /data
```

```
# mount /dev/da0s1d /data
```

```
# printf "/dev/da0s1d /data ufs rw 2 2" >> /etc/fstab
```

```
#2.MBR 在系统盘新建分片后再建分区(假设已为系统盘增加 50G 磁盘空间)
```

```
# gpart add -t freebsd /dev/da0 #在次跑/dev/da0 上添加分片，类型  
freebsd。不指定-s 参数时，表示将剩余空间都 分配给该分片
```

```
# gpart create -s BSD /dev/da0s2 #设置分片生效 gpart add -t freebsd-  
ufs /dev/da0s2 #在分片/dev/da0s2 上添加分区，类型 freebsd-ufs。不指定-s  
参数时，表示将 剩余空间都分配给该分区
```

```
# newfs /dev/da0s2a #格式化新分区。由于当前分区是当前分片上的第一个分区，  
因此系统默认分配为 a
```

```
# gpart set -a active -i 1 /dev/da0 #设置活动分片。若用 bsdinstall 或  
sade 创建新分片，则此步骤为必须
```

```
# mkdir /data mount /dev/da0s2a /data
```

```
# printf "/dev/da0s2a /data ufs rw 2 2" >> /etc/fstab
```

```
#3.GPT 在系统盘新建分区(假设已为系统盘增加 50G 磁盘空间)
```

```
# gpart add -t freebsd-ufs /dev/da0 #在磁盘/dev/da0 上添加分区，GPT 中  
没有分片的概念
```

```
# newfs /dev/da0p4 #格式化新分区。这里注意新分区名称，p1 是 boot 分区，p2  
是系统分区，p3 是 swap 分区，因此新分区默认为 p4
```

```
# mkdir /data mount /dev/da0p4 /data
```

```
# printf "/dev/da0p4t/datattufstrwt2t2n" >> /etc/fstab
```

```
#4.GPT 创建数据分区
```

```
# gpart create -s GPT /dev/da1 #为磁盘/dev/da1 设置分区表。若想用 MBR  
分区，则将-s 参数的值改为 MBR
```

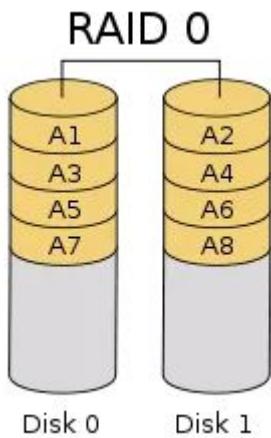
```
# gpart add -t freebsd-ufs /dev/da1 #在磁盘/dev/da1 上添加分区，类型  
freebsd-ufs
```

```
# newfs /dev/da1p1 #格式化新分区。由于当前分区是当前分片上的第一个分区，因  
此系统默认分配为 p1
```

```
# mkdir /data mount /dev/da1p1 /data
```

```
# printf "/dev/da1p1t/datattufstrwt2t2n" >> /etc/fstab
```

第二节 RAID 0



实现方式：RAID 0 最简单的实现方式就是把多块同样的硬盘串联在一起创建一个大的逻辑硬盘。最大优点就是可以整倍的提高硬盘的容量。如使用了三块 10T 的硬盘组建成 RAID 0 模式，那么磁盘容量就会是 30 TB。

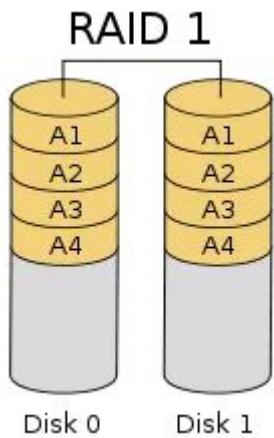
RAID 0数据恢复：由于不提供数据冗余保护，阵列中某一磁盘发生故障，将导致其中数据丢失，无法恢复。

应用场景：RAID 0 一般适用于对性能要求严格但对数据安全性和可靠性不高的应用，如视频、音频存储、临时数据缓存空间等。

最少硬盘数：创建 RAID 0最少需要 2 块硬盘

可用容量：磁盘空间利用率为 100%

第三节 RAID 1



实现方式： RAID 1 使用两组相同的磁盘系统互做镜像，在主硬盘上存放数据的同时也在镜像硬盘上写一样的数据。当主硬盘损坏时，镜像硬盘则代替主硬盘工作。

RAID1数据恢复： 如任一磁盘发生损坏，可以马上从镜像磁盘进行数据恢复。如：上图 `Disk0` 损坏导致数据丢失，我们可以用新盘替换故障盘，读取 `Disk1` 的数据，将其复制到新盘上，从而实现了数据的恢复。

应用场景： RAID 1 应用于对顺序读写性能要求高以及对数据保护极为重视的应用。如：服务器、数据库存储领域。

最少硬盘数： 创建 RAID 1 最少需要2块硬盘

可用容量： 实际可用的硬盘为总硬盘数量的一半

第四节 RAID 3

第四节

第五节 软 RAID 配置

第六节 GEOM Gate Network

第七节 磁盘装置标签

第八节 UFS Journaling 与GEOM

第九节 ZFS 磁盘加解密

概述

如果在安装的时候选择了 ZFS 磁盘加密，那么如何挂载该磁盘呢？

磁盘结构（FreeBSD 11 以后）

分区类型	挂载点	设备
freebsd-boot /EFI		/dev/ada0p1
freebsd-zfs	/	/dev/ada0p2/、/dev/ada0p2.eli
freebsd-swap		/dev/ada0p3、/dev/ada0p3.eli

很简单，也不需要密钥。

执行命令 `# geli attach /dev/ada0p3`

然后输入正确的密码即可通过 `zfs mount zroot/ROOT/default` 命令导入磁盘。

使用 GELI 加密 ZFS 卷

```
# 创建一个块设备
# zfs create -V 256M zroot/test
# 创建一个随机生成的、4K 大小的 key
# dd if=/dev/random of=/tmp/test.key bs=4k count=1
# 初始化并加载加密磁盘
# geli init -K /tmp/test.key /dev/zvol/zroot/test
# geli attach -k /tmp/test.key /dev/zvol/zroot/test
# 发现一个新设备
# ls /dev/zvol/zroot/test.eli
# 我们可以在该设备上创建一个新的文件分区
# zpool create -m /tmp/ztest ztest /dev/zvol/zroot/test.eli
```

GELI 数据备份和恢复

```
# 备份 GELT 数据
# geli backup /dev/zvol/zroot/test /tmp/test.eli
# 清空 GELT 数据
# geli clear /dev/zvol/zroot/test
# GELI尝试挂载 GELT设备，但无法做到，因为找不到他的 GELT 数据
# geli attach -k /tmp/test.key /dev/zvol/zroot/test
# 恢复 GELT 数据
# geli restore /tmp/test.eli /dev/zvol/zroot/test
# 现在我们可以挂载设备并导入池了
# geli attach -k /tmp/test.key /dev/zvol/zroot/test
# zpool import
```

调整 GELI 磁盘大小

```
# 调整 ZFS 卷
# zfs set volsize=512M zroot/test
# 现在还不能挂载 GELT 设备，因为 GELT 找不到数据
# geli attach /dev/zvol/zroot/test
# 我们需要告诉 GELT 以前设备的存储大小
# geli resize -s 256M /dev/zvol/zroot/test
# 现在我们可以挂载设备并导入池了
# geli attach -k /tmp/test.key /dev/zvol/zroot/test
# zpool import
```

第一节 概述

DTrace 是动态追踪技术的鼻祖，源自 SUN 公司的 Solaris 系统，提供了高级性能分析和调试功能，它的源代码采用 CDDL 许可证。

第二节 开启 DTrace

FreeBSD 11 以后

```
# kldload dtraceall
```

即可开启

对于 FreeBSD 9/10

FreeBSD 内核默认没有开启 DTrace 这项功能。要开启本功能必须加入参数重新编译内核。

建议先阅读第二十章 内核。

编辑内核配置文件加入：

```
options      KDTRACE_HOOKS
options      DDB_CTF
makeoptions DEBUG=-g
makeoptions WITH_CTF=1
```

如果是64 位操作系统：

```
options      KDTRACE_FRAME
```

第三节 使用 DTrace

首先安装 ksh93

```
# pkg install shells/ksh93
```

接下来安装 dtrace-toolkit

```
# pkg install sysutils/dtrace-toolkit
```

第一节 PPP 拨号

第二节 WiFi

英特尔 WIFI 5/6 芯片（AC 8265、AC 9260、AC 9560、AX200、AX201、AX210）驱动见最后一部分。

注意：如果安装 FreeBSD 的时候就不能识别出无线网卡，那么就是不支持你的无线网卡。请忽略下文。

一般网卡驱动

首先运行 `# ifconfig`，看看能不能找到你的网卡，如果能，那么你可以跳过本节了。

运行 `# sysctl net.wlan.devices`，他会告诉你的无线网卡驱动，如果冒号输出后边没有东西，那就是识别不了。请更换无线网卡。

编辑 `/boot/loader.conf` 文件

加入：

```
if_urtn_load="YES"
legal.realtek.license_ack=1
```

注意：这里只是示例，请添加自己所需的驱动（看 `# sysctl net.wlan.devices` 的输出）。

接下来，创建 `wlan0`

```
# ifconfig wlan0 create wlandev at0
```

`at0` 是你的网卡，具体看自己的 `# sysctl net.wlan.devices` 输出，该命令是临时的，若需要永久开机生效，在 `rc.conf` 中，加入：

```
# wlans_ath0 ="wlan0"
```

扫描 WiFi:

```
# ifconfig wlan0 up scan  
# ifconfig wlan0 ssid abc
```

连接 WiFi :

```
# dhclient wlan0
```

获取地址

连接加密网络

创建 `/etc/wpa_supplicant.conf` :

```
network={  
    ssid="WIFI 名字 (SSID)"  
    psk="WIFI 密码"  
}
```

在 `rc.conf` 里面加入

```
# ifconfig_wlan0="WPA SYNCDHCP"
```

然后重启电脑（因为命令有点问题，只能重启让 `rc.conf` 生效）

```
# wpa_supplicant -B -i wlan0 -c /etc/wpa_supplicant.conf  
# wpa_cli -i wlan0 scan // 搜索附近wifi网络  
# wpa_cli -i wlan0 scan_result // 打印搜索wifi网络结果  
# wpa_cli -i wlan0 add_network // 添加一个网络连接  
# wpa_cli -i wlan0 remove_network 1 // 删除一个网络连接  
# wpa_cli -i wlan0 set_network 0 ssid “name”  
# wpa_cli -i wlan0 set_network 0 psk “psk”  
# wpa_cli -i wlan0 enable_network 0
```

保存连接

```
# wpa_cli -i wlan0 save_config
```

断开连接

```
# wpa_cli -i wlan0 disable_network 0
```

连接已有的连接

```
# wpa_cli -i wlan0 list_network 列举所有保存的连接  
# wpa_cli -i wlan0 select_network 0 连接第1个保存的连接  
# wpa_cli -i wlan0 enable_network 0 使能第1个保存的连接
```

断开 WiFi

```
# ifconfig wlan0 down
```

附配置详情: <https://segmentfault.com/a/1190000011579147>

wpa 验证，静态 ip

```
# ifconfig_wlan0="WPA inet 192.168.1.100 netmask 255.255.255.0"  
  
# ifconfig wlan0 inet 192.168.0.100 netmask 255.255.255.0
```

开启无线 ap，先确认下你的网卡是否支持 hostap

```
# ifconfig wlan0 list caps
```

先销毁

```
# ifconfig wlan0 destroy
```

再创建

```
# ifconfig wlan0 create wlandev ath0 wlanmode hostap  
  
# ifconfig wlan0 inet 192.168.0.1 netmask 255.255.255.0 ssid  
freebsdap mode 11g channel 1
```

如果连不上或者搜不到调试信道，尝试将 WiFi 区域码选 Japan
，然后选 China

简单版本

以螃蟹卡为例：

>

```
# ee /boot/loader.conf
```

加入

```
rtwn_usb_load="YES"  
legal.realtek.license_ack=1
```

在 `/etc/rc.conf` 中写入

```
wlans_rtwn0="wlan0"  
ifconfig_wlan0="WPA DHCP"
```

执行:

```
# /etc/rc.d/netif start
```

英特尔 WIFI 5/6 芯片

以下部分仅适用于 FreeBSD 13.1、14.0。

iwlwifi 驱动适用于 AC 8265、AC 9260、AC 9560、AX200、AX201、AX210 以及旧的 iwm 驱动所包含的网卡，见

<https://www.intel.cn/content/www/cn/zh/support/articles/00005511/wireless.html>。iwm(4) 的原有网卡若想使用该驱动，请参考

https://wiki.freebsd.org/WiFi/Iwlwifi#I_have_an_iwm_284.29_supported_device。

该驱动仍不完善，不会自动加载（可以用命令 `# sysctl net.wlan.devices` 查看有无加载，如果没有显示 `iwlwifi0` 就没有加载），每次都需要手动加载。

将以下部分写入 `/etc/rc.conf`：

```
kld_list="if_iwlwifi"
wlans_iwlwifi0="wlan0"
wlandebug_wlan0="+state +crypto +node +auth +assoc +dot1xsm +wpa"
ifconfig_wlan0="WPA SYNC DHCP"
```

创建 `/etc/wpa_supplicant.conf`：

```
network={  
    ssid="WIFI 名字 (SSID)"  
    psk="WIFI 密码"  
}
```

先加载驱动看一下：

```
# kldload if_iwlwifi
```

注意：以下部分每次开机都要执行一次方能联网。

创建并链接：

```
# ifconfig wlan0 create wlandev iwlwifi0  
# /etc/rc.d/netif start wlan0
```

故障排除：<https://wiki.freebsd.org/WiFi/Iwlwifi>

第三节 USB RNDIS (USB 网络共享)

该教程在小米手机上测试通过，理论上同时支持 Android 和 iOS。

首先加载内核模块：

```
# kldload if_urndis #安卓 Android  
# kldload if_ipheth #苹果 iOS  
# kldload if_cdce #其他设备
```

启动时开机加载：写入到

```
if_urndis_load="YES"  
if_cdce_load="YES"  
if_ipheth_load="YES"
```

第四节 蓝牙

第五节 IPv6

第六节 CARP

第七节 VLAN

第八节 TCP BBR

TCP BBR 是一种 Google 开发的拥塞控制算法。作用有两个：

1. 在有一定丢包率的网络链路上充分利用带宽。
2. 降低网络链路上的 buffer 占用率，从而降低延迟。

一般来说，如果你使用了代理软件，建议开启 TCP BBR 功能，在速度和稳定性上会有十分显著的作用。该项目在 FreeBSD 中，由 Netflix 团队协助开发。最低系统版本支持：r363032，也即推荐 FreeBSD 13.0。

修改内核配置

```
# cd /usr/src/sys/amd64/conf
```

如果安装FreeBSD时没有选择安装内核源码，建议阅读 第二十章。

```
# cp GENERIC GENERIC-bbr  
# ee GENERIC-bbr
```

调整配置，修改 `ident` 的值为 `GENERIC-bbr`，在 `ident` 这一项下面加入以下项目：

```
options TCPHPTS  
options RATELIMIT  
makeoptions WITH_EXTRA_TCP_STACKS=1
```

新建 `/etc/src.conf`，内容为：

```
KERNCONF=GENERIC-bbr  
MALLOC_PRODUCTON=yes
```

编译并安装内核

```
# /usr/sbin/config GENERIC-bbr  
# cd ../compile/Generic-bbr  
# make cleandepend && make depend  
# make -jN+1
```

其中 `N` 建议为 `CPU` 核心数。

```
# make install
```

安装内核，完成后重启使用新内核。

```
# uname -a
```

如果显示出 `GENERIC-bbr`，则表示 TCP BBR 内核编译并安装成功。

配置和加载 BBR 模块

```
# sysrc kld_list+="tcp_rack tcp_bbr"
```

启动时加载 BBR 模块。

```
# echo 'net.inet.tcp.functions_default=bbr' >> /etc/sysctl.conf
```

设置默认使用 BBR，重启。

```
# sysctl net.inet.tcp.functions_default
```

如果结果是 `net.inet.tcp.functions_default: bbr`，则启用 TCP BBR 成功。

注意： 故障排除等事宜请参考：

https://github.com/netflix/tcplog_dumper

第一节 网络参数配置命令

系统设置工具 bsdconfig

`bsdconfig` 是 FreeBSD 提供的系统配置实用工具，是个 Shell 界面。

安全的操作 rc 文件

`sysrc` 是 FreeBSD 提供的 rc 文件实用工具，代替手动编辑 `rc.conf`。

作为网关服务器

打开 IP 转发功能：

```
# sysrc gateway_enable="YES"
# sysrc ipv6_gateway_enable="YES"
```

打开防火墙，开启 NAT：

```
# sysrc firewall_enable="YES"
# sysrc firewall_script="/etc/ipfw.rules"
# sysrc firewall_nat_enable="YES"
```

设置默认接受连接： # ee /boot/loader.conf

```
net.inet.ip.fw.default_to_accept=1
```

手动设置 resolv.conf

手动编辑 resolv.conf 后，重启系统又会被重置，因为 DHCP 会重写这个文件。

防止 `resolvconf` 服务覆盖 resolv.conf: `# ee /etc/resolvconf.conf`

```
resolv_conf="/dev/null"
```

再编辑 resolv.conf 就可以了。

查看网卡速率

每 1 秒刷新一次:

```
# systat -if 1
```

第二节 PF

OpenBSD Packet Filter(PF) 是一款自 OpenBSD 移植来的防火墙，提供了大量功能，包括 ALTQ (Alternate Queuing，交错队列)。

如需启用，可以在终端执行命令：

```
# cp /usr/share/examples/pf/pf.conf /etc #复制示例文件作为默认配置规则集文件，否则 pf 无法启动
# service pf enable #设置 pf 开机启动，也可以通过 bsdconfig 设置 pf_enable
# service pf start #启动 pf
```

pf 的管理命令为 `pfctl`，常用操作示例如下：

```
# pfctl -e #启动 pf, 相当于 service pf start

# pfctl -d #停止 pf, 相当于 service pf stop

# pfctl -f /etc/pf.conf #加载规则集文件中的规则

# pfctl -nf /etc/pf.conf #解析规则, 但不加载。-f 参数还可以与其他参数配合, 如 -N 表示只载入 NAT 规则, -R 表示只载入过滤规则, -A 只载入队列规则, -O 只载入选项规则

# pfctl -s all #查看 pf 所有对象信息, 如果想查看特定对象信息, 可以用 nat、queue、rules、Anchors、states、Sources、info、Running、labels、timeouts、memory、Tables、osfp、Interfaces 替换 all

# pfctl -F all #清理 pf 所有规则,
```

如果想查看特定规则, 可以用 nat、queue、rules、states、Sources、info、Tables、osfp 替换 all。

不过以上操作并没有对规则的管理, 因此还需要修改规则集文件, 常用示例如下:

```
# scrub in all #整理所有输入的数据
```

```
block all #拒绝所有访问。
```

ipfilter #是默认明示禁止的防火墙，因此需要通过此规则禁止所有访问。其中 **block** 是动作，**out** 表示拒绝，**pass** 表示通过；**all** 是 **from any to any** 的简写，表示从源地址到目标地址，地址通常用网段(如 **192.168.1.0/24**)或 IP 地址(如 **192.168.1.100**)，**any** 是特殊词，表示任何地址；此外，当规则同时适用于输入 **in** 和输出 **out** 时，可以省略关键字，因此本条规则同时适用于输入输出

```
pass quick on lo0 all #放开回环接口的访问权限，回环接口不对外部。quick 关键字表示若规则匹配，就停止执行，不会再执行后续规则
```

pass in quick proto tcp from any to 192.168.1.184 port 80 #增加 TCP 协议访问 80 端口的规则，允许任何设备以 TCP 协议访问本机 80 端口。其中 **proto tcp** 是访问协议，常用值有 **tcp**、**udp**、**icmp**、**icmp6**；**port = 80** 是端口，写在目标地址之后为目标 端口，源地址之后未写，表示从源地址的任何端口发起访问

```
pass out quick proto tcp from 192.168.1.184 port 80 to any #允许回显信息给任何访问的设备
```

```
rdr pass on em0 inet proto tcp from any to 192.168.1.184 port 80 -> 192.168.1.166 port 8080 #增加 80 端口到 8080 端口流量转发的规则，由于测试机只有一块网卡，因此转发仅限本机
```

pass quick inet proto icmp all icmp-type 8 code 0 #允许本机与外部设备互 ping。其中 **icmp-type 8** 是查询请求，**code** 表示返回码为 0

```
pass out quick inet proto icmp from 192.168.1.184 to any icmp-type 11 code 0 #允许 traceroute 命令以 ICMP 协议执行
```

```
pass out quick proto udp from 192.168.1.184 to any port 33434 <> 34500 #traceroute 默认协议 UDP，端口号从 33434 开始，每转发一次端口号加 1
```

下面根据我的操作系统整理规则集文件 `/etc/pf.conf` 如下：

```
#流量整形 scrub in all #转发规则

rdr pass on em0 inet proto tcp from any to 192.168.1.184 port 8080 ->
192.168.1.184 port 80 #注意规则次序，根据 pf.conf 规则，转发规则应位于过滤规则之前，相关内容请参考帮助 #过滤规则

block all pass quick on lo0 all #设置任何设备可以访问服务器的 22、80、
443、4200、10000 端口

pass in quick proto tcp from any to 192.168.1.184 port {
22,80,443,4200,10000 }

pass out quick proto tcp from 192.168.1.184 port {
22,80,443,4200,10000 } to any

pass out quick proto tcp from 192.168.1.184 to any port { 80,443 }
keep state #设置服务器访问任何网络设备 的 80、443 端口

pass out quick proto udp from any to any port 53 keep state #设置访问
DNS 服务器

pass out quick proto udp from any to any port 67 keep state #设置访问
DHCP 服务器

pass quick inet proto icmp all icmp-type 8 code 0

pass out quick inet proto icmp from 192.168.1.184 to any icmp-type 11
code 0

pass out quick proto udp from 192.168.1.184 to any port 33434 ><
34500 保存文件，接下来在终端执行命令：

# pfctl -Fa -f /etc/pf.conf #加载规则集文件中的规则 就可以看到效果了。
```

第三节 IPFW

(一) 介绍说明:

IPFIREWALL (IPFW) 是一个由 FreeBSD 发起的防火墙应用软件，它由 FreeBSD 的志愿者成员编写和维护。

在 FreeBSD 12 中，ipfw 已经默认被编译进内核了，它默认会有一条规则，规则号为 65536，是不可以删除的，这条规则会把所有流量都切断，所以还没配置好之前，千万不要随意启动 ipfw，否则就会面临无法连上远程 FreeBSD 的问题。

图形化配置工具：

```
#pkg install fwbuilder
```

(二) 配置ipfw:

1. 执行以下命令:

```
# sysrc firewall_enable="YES"  # 允许防火墙开机自启
# sysrc firewall_type="open"  # 让系统把流量通过, 这样就可以使用防火墙
# sysrc firewall_script="/etc/ipfw.rules"  # 制定ipfw规则的路径, 我们待会儿在这里编辑规则
# sysrc firewall_logging="YES"  # 这样ipfw就可以打日志
# sysrc firewall_logif="YES"  # 把日志打到 `ipfw0` 这个设备里
```

1. 编辑 `/etc/ipfw.rules` 文件:

```
# ee /etc/ipfw.rules

IPF="ipfw -q add"
ipfw -q -f flush

# loopback
$IPF 10 allow all from any to any via lo0
$IPF 20 deny all from any to 127.0.0.0/8
$IPF 30 deny all from 127.0.0.0/8 to any
$IPF 40 deny tcp from any to any frag

# statefull
$IPF 50 check-state
$IPF 60 allow tcp from any to any established
$IPF 70 allow all from any to any out keep-state
$IPF 80 allow icmp from any to any

# open port for ssh
$IPF 110 allow tcp from any to any 22 out
$IPF 120 allow tcp from any to any 22 in

# open port for samba
$IPF 130 allow tcp from any to any 139 out
$IPF 140 allow tcp from any to any 139 in
$IPF 150 allow tcp from any to any 445 out
$IPF 160 allow tcp from any to any 445 in
$IPF 170 allow udp from any to any 137 out
$IPF 180 allow udp from any to any 137 in
$IPF 190 allow udp from any to any 138 out
$IPF 200 allow udp from any to any 138 in

# deny and log everything
$IPF 500 deny log all from any to any
```

额外说明： samba 开放 tcp/139, 445 端口， udp/137, 138 端口

1. 启动 ipfw:

```
# service ipfw start

Firewall rules loaded.
Firewall logging enabled.
ifconfig: interface ipfw0 already exists
Firewall logging pseudo-interface (ipfw0) created.
```

1. 查看 ipfw 状态:

```
# service ipfw status

ipfw is enabled
```

1. 查看 ipfw 规则条目

```
# ipfw list

00010 allow ip from any to any via lo0
00020 deny ip from any to 127.0.0.0/8
00030 deny ip from 127.0.0.0/8 to any
00040 deny tcp from any to any frag
00050 check-state :default
00060 allow tcp from any to any established
00070 allow ip from any to any out keep-state :default
00080 allow icmp from any to any
00110 allow tcp from any to any 22 out
00120 allow tcp from any to any 22 in
00500 deny log ip from any to any
65535 deny ip from any to any
```

第四节 IPFILTER (IPF)

IPF是一款开源软件，作者 Darren Reed。

如果想启用 ipf，可以执行以下命令：

```
#ipfilter

# cp /usr/share/examples/ipfilter/ipf.conf.sample /etc/ipf.rules #复制示例文件作为默认配置规则集文件，否则 ipfilter 启动后会没有规则。示例文件自带的规则不会影响使用

# service ipfilter enable #设置 ipfilter 开机启动，也可以通过 bsdconfig 设置 ipfilter_enable

# service ipfilter start #启动 ipfilter #ipnat sudo cp /usr/share/examples/ipfilter/ipnat.conf.sample /etc/ipnat.rules #复制示例文件作为默认配置规则集文件，否则 ipnat 无法启动

# service ipnat enable #设置 ipnat 开机启动，也可以通过 bsdconfig 设置 ipnat_enable

# service ipnat start #启动 ipnat。
```

注意，ipfilter 服务重启后，ipnat 也需要重启。

ipf 的管理命令主要用 ipf、ipfstat 和 ipnat，常用操作示例如下：

```
# ipf -E #启动 ipfilter, 相当于 service ipfilter start

# ipf -D #停止 ipfilter, 相当于 server ipfilter stop

# ipf -f /etc/ipf.rules #加载规则集文件中的规则 ipfstat #查看所有规则

# ipfstat -iohn #查看规则, i 表示输入规则, o 表示输出规则, h 表示通过该规则的流量, n 表示记录编号

# ipfstat -t #进入监控模式, 按 Q 退出

# ipf -Fa #清理已加载的规则

# ipnat -f /etc/ipnat.rules #加载规则集文件中的 nat 规则

# ipnat -s #汇总并显示 nat 状态 ipnat -lh #列表显示 nat 规则, 加 h 表示同时显示通过该规则的流量

# ipnat -CF #清理已加载的 nat 规则 不过以上操作并没有对规则的管理, 因此还需要修改规则集文件, 常用示例如下: block all # #拒绝所有访问。ipfilter 是默认明示禁止的防火墙, 因此需要通过下列规则禁止所有访问 block in all #block 是动作, block 表示拒绝, pass 表示通过; in 为数据方向, in 为入, out 为出, 在 ipfilter 里数据方向是必须的; all 是 from any to any 的简写, 表示从源地址到目标地址, 地址通常用网段(如 192.168.1.0/24)或 IP 地址(如 192.168.1.100), any 是特殊词, 表示任何地址

block out all #放开回环接口的访问权限, 回环接口不对外部

pass in quick on lo0 all #quick 关键字表示若规则匹配, 就停止执行, 不会再执行后续规则

pass out quick on lo0 all #增加 TCP 协议访问 80 端口的规则

pass in quick proto tcp from any to 192.168.1.184 port = 80 #允许任何设备以 TCP 协议访问本机 80 端口。 其中 proto tcp 是访问协议, 常用值有
```

`tcp`、`udp`、`tcp/udp`、`icmp`，不写则表示支持所有协议；`port = 80` 是 端口，写在目标地址之后为目标端口，源地址之后未写，表示从源地址的任何端口发起访问

```
pass out quick proto tcp from 192.168.1.184 to any #允许回显信息给任何访问的设备 #增加 80 端口到 8080 端口流量转发的规则
```

```
pass in quick proto tcp from any to 192.168.1.184 port = 80 #首先放开 ipfilter 的访问限制 rdr em0 192.168.1.184 port 80 -> 192.168.1.184 port 8080 #由于测试机只有一块网卡，因此转发仅限本机
```

```
pass out quick proto icmp from 192.168.1.184 to any icmp-type 8 keep state #允许本机 ping 任何外部设备。 其中 ICMP type 8 是查询请求；keep state 表示维持状态。如与下例合并，会完全放开 ping 的功能
```

```
pass in quick proto icmp from any to 192.168.1.184 icmp-type 8 keep state #允许任何外部设备 ping 本机
```

```
pass out quick proto icmp from 192.168.1.184 to any icmp-type 0 #允许 traceroute 命令以 ICMP 协议执行
```

```
pass out quick proto udp from 192.168.1.184 to any port 33434 <> 34500 keep state #traceroute 默认协议 UDP， 端口号从 33434 开始，每转发一次端口号加 1
```

下面根据我的操作系统整理规则集文件 `/etc/ipf.rules` 如下：

```
block in all

block out all

pass in quick on lo0 all

pass out quick on lo0 all #设置任何设备可以访问服务器的 22、80、443、
4200、10000 端口

pass in quick proto tcp from any to 192.168.1.184 port = {
22,80,443,4200,10000 }

pass out quick proto tcp from 192.168.1.184 port = {
22,80,443,4200,10000 } to any pass out quick proto tcp from
192.168.1.184 to any port = { 80,443 } keep state #设置服务器访问任何网
络设备的 80、443 端口

pass out quick proto udp from any to any port = 53 keep state #设置访
问 DNS 服务器

pass out quick proto udp from any to any port = 67 keep state #设置访
问 DHCP 服务器

pass out quick proto icmp from 192.168.1.184 to any icmp-type 8 keep
state

pass in quick proto icmp from any to 192.168.1.184 icmp-type 8 keep
state

pass out quick proto icmp from 192.168.1.184 to any icmp-type 0

pass out quick proto udp from 192.168.1.184 to any port 33434 ><
34500 keep state
```

```
pass in quick proto tcp from any to 192.168.1.184 port = 8080 #数据转发前要放开相应端口
```

然后再整理 NAT 规则集文件 `/etc/ipnat.rules` 如下：

```
# rdr em0 192.168.1.184 port 8080 -> 192.168.1.184 port 80 #设置本机  
8080 到 80 端口的映射
```

保存文件，接下来在终端执行命令：

```
# ipf -Fa -f /etc/ipf.rules #加载规则集文件中的规则  
  
# ipnat -CF -f /etc/ipnat.rules #加载规则集文件中的 nat 规则就可以看到效果了。
```

第一节 FTP 服务器

FTP 意为文件传输协议。使用 FTP 服务搭建服务器可以快速传输文件。

pure-ftpd（以 MySQL 支持为例）

RFC 2640 的支持已经被移除，所以 Windows 下的 FTP 文件会乱码，见 <https://www.pureftpd.org/project/pure-ftpd/news/> 无法解决，同时不建议把 Windows 的系统编码改为 UTF8，会造成更多乱码的发生，比如 zip 文件。

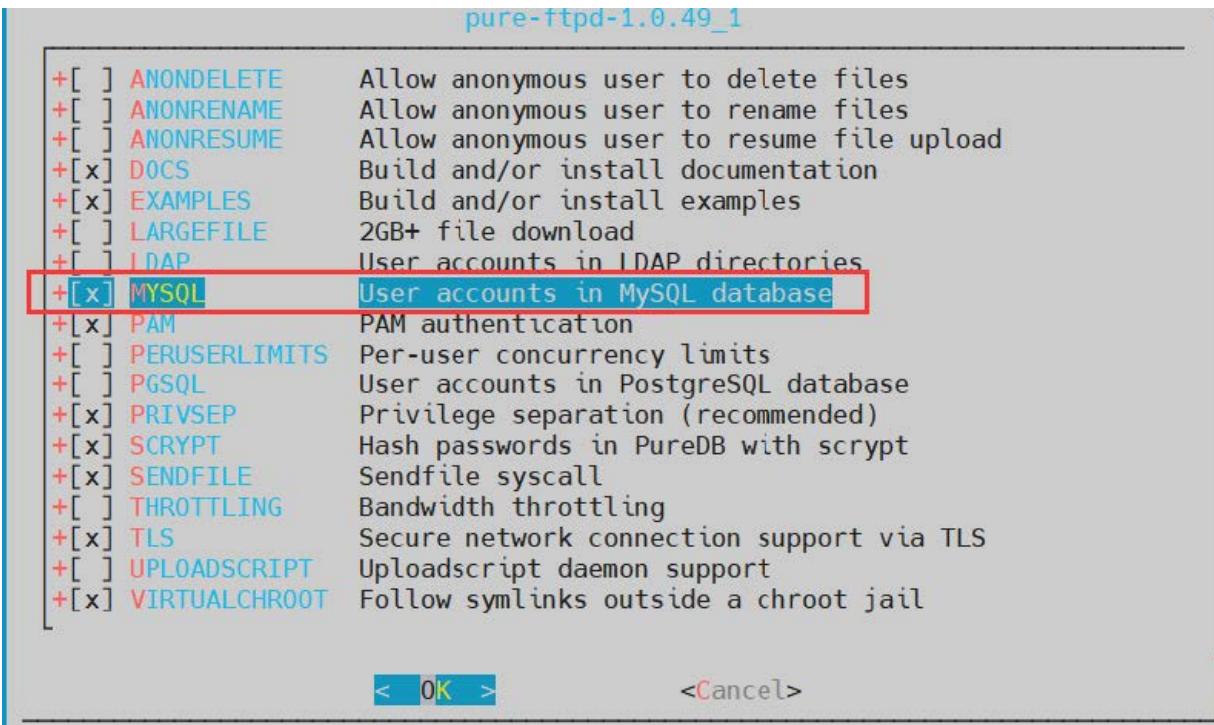
注意：本示例以 mysql 5.x 为例。

安装

由于 pkg 包不带有数据库支持功能，所以需要通过 ports 来安装该软件：

```
# /usr/ports/ftp/pure-ftpd  
# make config-recursive
```

选中 mysql，其余保持默认选项回车即可：



```
# make install clean
```

注意：关于 mysql 的基本设置请看 第十七章

请自行安装 mysql，理论上兼容 mysql 5.x、8.x

配置 /usr/local/etc/pure-ftpd.conf 文件

生成配置文件：

```
# cp /usr/local/etc/pure-ftpd.conf.sample /usr/local/etc/pure-
ftpd.conf
# cp /usr/local/etc/pureftpd-mysql.conf.sample
/usr/local/etc/pureftpd-mysql.conf
```

编辑配置文件并增加 mysql 的支持:

```
#兼容 ie 等非正规化的 ftp 客户端

BrokenClientsCompatibility yes

# 被动连接响应的端口范围。
PassivePortRange 30000 50000

# 认证用户允许登陆的最小组 ID (UID) 。
MinUID 2000

# 仅允许认证用户进行 FXP 传输。
AllowUserFXP yes

# 用户主目录不存在的话，自动创建。
CreateHomeDir yes

# MySQL configuration file (see README.MySQL)

MySQLConfigFile /usr/local/etc/pureftpd-mysql.conf
```

配置 mysql

创建数据库

```
create database pureftp;
use pureftp;
DROP TABLE IF EXISTS `users`;
CREATE TABLE `users` (
  `User` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_bin NOT NULL,
  `Password` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_bin NOT NULL,
  `Uid` int(11) NOT NULL DEFAULT -1 COMMENT '用户ID',
  `Gid` int(11) NOT NULL DEFAULT -1 COMMENT '用户组ID',
  `Dir` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_bin NOT NULL,
  `quotafiles` int(255) NULL DEFAULT 500,
  `quotasize` int(255) NULL DEFAULT 30,
  `ulbandwidth` int(255) NULL DEFAULT 80,
  `dlbandwidth` int(255) NULL DEFAULT 80,
  `ipaddress` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci NULL DEFAULT'*',
  `comment` int(255) NULL DEFAULT NULL,
  `status` tinyint(4) NULL DEFAULT 1,
  `ulratio` int(255) NULL DEFAULT 1,
  `dlratio` int(255) NULL DEFAULT 1,
  PRIMARY KEY (`User`) USING BTREE
) ENGINE = InnoDB CHARACTER SET = utf8mb4 COLLATE = utf8mb4_general_ci ROW_FORMAT = Dynamic;
INSERT INTO `users` VALUES ('demo', 'demo&2022*', 2002, 2000, '/home/www/demo', 500, 30, 80, 80, '*', NULL, NULL, 1, 1);
```

创建登录数据库用户及设置密码

```
grant select,insert,update,delete on pureftp.* to pftp@localhost
identified by "Ab123456&";
```

配置 /usr/local/etc/pureftpd-mysql.conf

```
#####
# #
# Sample Pure-FTPd Mysql configuration file. #
# See README.MySQL for explanations. #
# #
#####

# Optional : MySQL server name or IP. Don't define this for unix
# sockets.

# MYSQLServer 127.0.0.1
MYSQLServer localhost

# Optional : MySQL port. Don't define this if a local unix socket is
# used.

MYSQLPort 3306

# Optional : define the location of mysql.sock if the server runs on
# this host.

MYSQLSocket /var/run/mysqld/mysqld.sock

# Mandatory : user to bind the server as.

MYSQLUser pftp

# Mandatory : user password. You must have a password.

MYSQLPassword Ab123456&
```

```
# Mandatory : database to open.

MYSQLDatabase pureftpd


# Mandatory : how passwords are stored
# Valid values are : "cleartext", "argon2", "scrypt", "crypt",
"sha1", "md5",password" and "any"

# ("password" = MySQL password() function, which is
sha1(sha1(password)))

#MySQLCrypt scrypt
MySQLCrypt cleartext


# In the following directives, parts of the strings are replaced at
# run-time before performing queries :
#
# \L is replaced by the login of the user trying to authenticate.
# \I is replaced by the IP address the user connected to.
# \P is replaced by the port number the user connected to.
# \R is replaced by the IP address the user connected from.
# \D is replaced by the remote IP address, as a long decimal number.
#
# Very complex queries can be performed using these substitution
strings,
# especially for virtual hosting.

# Query to execute in order to fetch the password

MySQLGetPW SELECT Password FROM users WHERE User='\L'
```

```
# Query to execute in order to fetch the system user name or uid
MYSQLGetUID SELECT Uid FROM users WHERE User='\\L'

# Optional : default UID - if set this overrides MYSQLGetUID

MySQLDefaultUID 2000

# Query to execute in order to fetch the system user group or gid

MYSQLGetGID SELECT Gid FROM users WHERE User='\\L'

# Optional : default GID - if set this overrides MYSQLGetGID

MySQLDefaultGID 2000

# Query to execute in order to fetch the home directory

MYSQLGetDir SELECT Dir FROM users WHERE User='\\L'

# Optional : query to get the maximal number of files
# Pure-FTPd must have been compiled with virtual quotas support.

# MySQLGetQTAFS SELECT QuotaFiles FROM users WHERE User='\\L'

# Optional : query to get the maximal disk usage (virtual quotas)
# The number should be in Megabytes.
# Pure-FTPd must have been compiled with virtual quotas support.

# MySQLGetQTASZ SELECT QuotaSize FROM users WHERE User='\\L'
```

```
# Optional : ratios. The server has to be compiled with ratio
support.

# MySQLGetRatioUL SELECT ULRatio FROM users WHERE User='\L'
# MySQLGetRatioDL SELECT DLRatio FROM users WHERE User='\L'

# Optional : bandwidth throttling.
# The server has to be compiled with throttling support.
# Values are in KB/s .

# MySQLGetBandwidthUL SELECT ULBandwidth FROM users WHERE User='\L'
# MySQLGetBandwidthDL SELECT DLBandwidth FROM users WHERE User='\L'

# Enable ~ expansion. NEVER ENABLE THIS BLINDLY UNLESS :
# 1) You know what you are doing.
# 2) Real and virtual users match.

# MySQLForceTildeExpansion 1

# If you're using a transactionnal storage engine, you can enable SQL
# transactions to avoid races. Leave this commented if you are using
# the
# traditional MyIsam engine.

# MySQLTransactions On
```

添加 ftp 组和用户

```
# pw groupadd ftpgroup -g 2000  
# pw useradd ftpuser -u 2001 -g 2000
```

或

```
# pw useradd ftpuser -u 2001 -g 2000 -s /sbin/nologin -w no -d  
/home/vftp -c "VirtualUser Pure-FTPd" -m
```

配置 FTP 目录

```
# mkdir /home/www/pureftp  
# chown -R ftpuser /home/www/  
# chgrp -R ftpgroup /home/www/
```

服务操作

```
# sysrc pureftpd_enable="YES"  
# service pure-ftpd start #启动服务器  
# service pure-ftpd stop #停止服务  
# service pure-ftpd restart #重启服务  
  
## proftpd (以 mysql 支持为例)
```

安装 proftpd（以 mysql 支持为例）

```
# pkg install proftpd proftpd-mod_sql_mysql
```

编辑配置文件 /usr/local/etc/proftpd.conf

```
# cat /usr/local/etc/proftpd.conf
ServerName "Test Ftp Server"
ServerType standalone
DefaultServer on
ServerIdent on "FTP Server ready"
DeferWelcome off
Port 21
Umask 022
TimeoutLogin 300
TimeoutIdle 36000
TimeoutNoTransfer 36000
TimeoutStalled 36000
TimeoutSession 0
User proftpd
Group proftpd
MaxInstances 100
MaxClientsPerHost 100
AllowRetrieveRestart on
AllowStoreRestart on
AllowOverwrite on
AllowOverride off
RootLogin off
IdentLookups off
UseReverseDNS off
DenyFilter \*.*/
TimesGMT off
DefaultRoot ~
#RLimitCPU 1200 1200
RLimitMemory 256M 256M
RLimitOpenFiles 1024 1024
PassivePorts 50000 60000
LogFormat default "%h %l %u %t \"%r\" %s %b"
LogFormat auth "%v [%P] %h %t \"%r\" %s"
LogFormat write "%h %l %u %t \"%r\" %s %b"
SystemLog /var/log/proftpd/proftpd.log
```

```
TransferLog /var/log/proftpd/xfer.log
ExtendedLog /var/log/proftpd/access.log WRITE,READ write
ExtendedLog /var/log/proftpd/auth.log AUTH auth
LoadModule mod_sql.c
LoadModule mod_sql_mysql.c
<Global>
    SQLConnectInfo proftpd@localhost proftpd proftpd_password
    SQLAuthTypes Crypt
    SQLUserInfo users username password uid gid homedir NULL
    SQLDefaultGID 2000
    SQLDefaultUID 2000
    RequireValidShell off
    SQLAuthenticate users*
    SQLLogFile /var/log/proftpd.log
    SQLNamedQuery getcount SELECT "count, username from users where
username='%u'"
    SQLNamedQuery updatecount UPDATE "count=count+1 WHERE
username='%u'" users
    SQLShowInfo PASS "230" "You've logged on %{getcount} times, %u"
    SQLLog PASS updatecount
    SQLLog DELE,RETR,STOR, log_work
    SQLNamedQuery log_work FREEFORM "\
INSERT INTO worklog (\ \
user_name,\ \
file_and_path,\ \
bytes,\ \
send_time,\ \
client_ip,\ \
client_name,\ \
client_command) \ \
VALUES('%u','%f','%b','%T','%a','%h','%m')"
</Global>
```

我们在设置中指定服务器将在主动模式下在端口 21 上工作，在被动模式下在 50000–60000 范围内工作。这些端口应该在防火墙中打开。对于 PF，这是通过以下规则完成的：

```
pass in quick on $ext_if proto tcp from any to $ext_if port { 21,  
50000:60000 }
```

创建用户

出于安全目的，我们将以非 root 用户身份运行 Proftpd。因此，我们将创建此用户：

```
# adduser
Username: proftpd
Full name: FTP User
Uid (Leave empty for default):
Login group [proftpd]:
Login group is proftpd. Invite proftpd into other groups? []:
Login class [default]:
Shell (sh csh tcsh bash nologin) [sh]: nologin
Home directory [/home/proftpd]:
Home directory permissions (Leave empty for default):
Use password-based authentication? [yes]: no
Lock out the account after creation? [no]:
Username : proftpd
Password : <disabled>
Full Name : FTP User
Uid : 2000
Class :
Groups : proftpd
Home : /home/proftpd
Shell : /usr/sbin/nologin
Locked : no
OK? (yes/no): yes
adduser: INFO: Successfully added (proftpd) to the user database.
Add another user? (yes/no): no
Goodbye!
```

现在已经创建了自己的 proftpd 用户和组 ID。因此，在添加 ftp 用户时，您将使用它。您可以通过以下方式确定 UID：

```
# cat /etc/passwd | grep proftpd
proftpd:*:2000:2000:FTP User:/home/proftpd:/usr/sbin/nologin
```

日志相关

创建一个目录来存储 FTP 服务器的日志：

```
# mkdir /var/log/proftpd
```

创建一个 MySQL 数据库和一个对创建的数据库具有完全访问权限的用户：

```
CREATE DATABASE `proftpd` CHARACTER SET utf8 COLLATE utf8_general_ci;
```

创建数据库用户和密码(授权 proftpd 数据库)：

```
grant select,insert,update,delete on proftpd.* to pftp@localhost  
identified by "123456";  
FLUSH PRIVILEGES; 立即生效权限
```

或

```
grant select,insert,update,delete on *.* to pftp@"localhost"  
Identified by "123456";
```

创建数据量：

```
DROP TABLE IF EXISTS users;
CREATE TABLE `users` (
    `username` varchar(30) NOT NULL DEFAULT '',
    `descr` text CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci NOT
NULL,
    `password` varchar(30) NOT NULL DEFAULT '',
    `uid` int(11) DEFAULT NULL,
    `gid` int(11) DEFAULT NULL,
    `homedir` varchar(255) DEFAULT NULL,
    `shell` varchar(255) DEFAULT NULL,
    `count` int(11) NOT NULL DEFAULT '0',
    UNIQUE KEY `username` (`username`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8mb4;
DROP TABLE IF EXISTS worklog;
CREATE TABLE worklog (
    id bigint(20) UNSIGNED NOT NULL AUTO_INCREMENT,
    date timestamp(0) NULL DEFAULT CURRENT_TIMESTAMP(0),
    user_name varchar(20) CHARACTER SET utf8mb4 COLLATE
utf8mb4_general_ci NULL DEFAULT NULL,
    file_and_path varchar(1024) CHARACTER SET utf8mb4 COLLATE
utf8mb4_general_ci NULL,
    bytes bigint(20) NULL DEFAULT NULL,
    send_time varchar(9) CHARACTER SET utf8mb4 COLLATE
utf8mb4_general_ci NULL DEFAULT NULL,
    client_ip varchar(15) CHARACTER SET utf8mb4 COLLATE
utf8mb4_general_ci NULL DEFAULT NULL,
    client_name text CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci
NULL,
    client_command varchar(5) CHARACTER SET utf8mb4 COLLATE
utf8mb4_general_ci NULL DEFAULT NULL,
    PRIMARY KEY (id) USING BTREE,
    UNIQUE INDEX id(id) USING BTREE
) ENGINE = MyISAM CHARACTER SET = utf8mb4 COLLATE =
utf8mb4_general_ci ROW_FORMAT = DYNAMIC;
```

创建一个目录和一个测试 FTP 用户，将创建的目录指定为用户目录：

```
# mkdir -p /home/www/ftp
# chown -R proftpd:proftpd /home/www/ftp
# mysql -u proftpd -p
INSERT INTO `proftpd`.`users` (`username` , `descr` , `password` ,
`uid` , `gid` , `homedir` , `shell` , `count` ) VALUES ('test', 'Test
user', ENCRYPT('FTPpassword_here') , '2000', '2000',
'/home/www/ftp', NULL , '0' );

Query OK, 1 row affected, 1 warning (0.02 sec)
```

服务操作

```
# sysrc proftpd_enable="YES"

# service proftpd start #启动服务器

# service proftpd stop #停止服务

# service proftpd restart #重启服务
```

连接到 FTP 服务器

简单示例：

```
# telnet localhost 21
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
220 FTP Server ready
quit
221 Goodbye.
```

使用 `ftp` 命令可以快速连接到 FTP 服务器。

用法：

```
ftp [选项] [URL]
```

选项：

`-4` 强制使用 IPv4 协议连接

`-6` 强制使用 IPv6 协议连接

`-a` 使用匿名登录

`-q [quittime]` 在设定时间后连接失败则自动放弃连接

`-r [wait]` 每隔 `wait` 秒发送一次连接请求

-A 强制使用主动模式

-d 开启调试模式

-v 开启啰嗦模式

-V 关闭啰嗦模式

登录后的命令：

account [passwd] 提交补充密码

append [local-file] [remote-file] 以 **remote-file** 为文件名向服务器上传本地文件 **local-file**

ascii 将FTP文件传送类型设置为 ASCII 模式

bell 在文件传送完后发出提示音

bye 结束与服务器的会话

cd 切换目录

cdup 退回父目录

delete 删除文件

dir 显示该目录下的文件及文件夹

features 显示该服务器支持的功能

get remote-file 下载服务器上的 **remote-file**

第二节 DHCP 服务器

第三节 Nodejs 相关

在 FreeBSD 13 上的安装

nodejs 依赖 `/lib/libcrypto.so.111` 的某个特定版本，而这意味着如果你需要在 FreeBSD 上使用 NodeJS，你必须留意 FreeBSD 本身的版本，尤其是当你的 pkg 配置使用了 latest 源时。

一般而言，如果想要在 FreeBSD 13.0 上 安装 node+yarn，请这么做：

```
# freebsd-update fetch install #必须先更新基本系统  
# pkg install yarn #会自动安装对应版本的 nodejs
```

如果你跳过了 FreeBSD 的升级直接安装软件，那么在 FreeBSD 13 上，你将会遇到以下错误：

```
# pkg install yarn  
% node  
ld-elf.so.1: /lib/libcrypto.so.111: version OPENSSL_1_1_1e  
required by /usr/local/bin/node not found
```

第四节 DNS 服务器

第五节 NIS 服务器

第六节 Postfix 服务器

环境: freebsd 11

设置 samba 为独立服务器

安装 samba

```
# pkg install samba413
```

配置 samba

1. 打开/etc/rc.conf

```
# ee /etc/rc.conf
```

1. 在 /etc/rc.conf 最后加入如下，并保存：

```
nmbd_enable="YES"
winbindd_enable="YES"
samba_enable="YES"
samba_server_enable="YES"
```

1. 创建 /usr/local/etc/smb4.conf，添加如下内容并保存

```
#vi /usr/local/etc/smb4.conf

[root]
comment = root's stuff
path = /root
public = no
browseable = yes
writable = yes
printable = no
create mask = 0755
```

1. 创建 samba root 用户：

```
# smbpasswd -a root
```

1. 进入 /usr/local/etc

```
# cd /usr/local/etc
```

1. 再执行

```
# service samba_server start //启动命令
```

或

```
# service samba_server restart //重启命令
```

1. 查看 samba 状态:

```
# service samba_server status
```

1. 在 windows 下利用 192.168.X.X 访问共享文件夹（以实际 IP 为准，Windows 需要先开启 SMB 1.0 支持）

192.168.X.X

将Samba设置为域成员

环境: freebsd 12

配置静态IP地址

使用如下命令配置：

```
bsdconfig
```

配置主机名

```
# ee /etc/rc.conf  
  
hostname="fb"
```

配置 DNS

```
# ee /etc/resolv.conf

# Generated by resolvconf
search SVROS.COM          //设置域控制器域名
# nameserver 192.168.253.2

nameserver 192.168.253.130 //设置域控制器IP地址
nameserver 223.5.5.5
nameserver 127.0.0.1
options edns0
```

修改 /etc/sysctl.conf

```
# echo "kern.maxfiles=25600" >> /etc/sysctl.conf  
# echo "kern.maxfilesperproc=16384" >> /etc/sysctl.conf  
# echo "net.inet.tcp.sendspace=65536" >> /etc/sysctl.conf  
# echo "net.inet.tcp.recvspace=65536" >> /etc/sysctl.conf
```

创建 /etc/krb5.conf

```
[libdefaults]
    default_realm = SVROS.COM //设置域名
    dns_lookup_realm = true
    dns_lookup_kdc = true
    ticket_lifetime = 24h
    renew_lifetime = 7d
    forwardable = yes
```

修改 /etc/nsswitch.conf

```
# sed -i -e "s/^passwd:.*/passwd: files winbind/" /etc/nsswitch.conf  
# sed -i -e "s/^group:.*/group: files winbind/" /etc/nsswitch.conf
```

创建 /usr/local/etc/smb4.conf

```
[global]
    workgroup = SVROS
    server string = Samba Server Version %v
    security = ads
    realm = SVROS.COM
    domain master = no
    local master = no
    preferred master = no
    socket options = TCP_NODELAY IPTOS_LOWDELAY SO_RCVBUF=131072
    SO_SNDBUF=131072
    use sendfile = true

    idmap config * : backend = tdb
    idmap config * : range = 100000-299999
    idmap config SVROS : backend = rid
    idmap config SVROS : range = 10000-99999
    winbind separator = +
    winbind enum users = yes
    winbind enum groups = yes
    winbind use default domain = yes
    winbind nested groups = yes
    winbind refresh tickets = yes
    template homedir = /home/%D/%U
    template shell = /bin/false

    client use spnego = yes
    client ntlmv2 auth = yes
    encrypt passwords = yes
    restrict anonymous = 2
    log file = /var/log/samba4/log.%m
    max log size = 50

#===== Share Definitions =====
```

```
[testshare]
comment = Test share
path = /samba/testshare
read only = no
force group = "Domain Users"
directory mode = 0770
force directory mode = 0770
create mode = 0660
force create mode = 0660
```

上面【testshare】最后两行内容实际使用权限优化（可选）

```
create mode = 0750
force create mode = 0750
```

将 samba 加入到域

```
net ads join --no-dns-updates -U administrator
net ads testjoin
# Should report "Join is OK"
# On your DC, open the DNS MMC and add an "A" entry for your BSD
server so clients can find it
```

使 samba 启动并设置为开机自启动

```
# echo "samba_server_enable=YES" >> /etc/rc.conf  
# echo "winbindd_enable=YES" >> /etc/rc.conf  
# service samba_server start
```

测试 Kerberos

```
kinit administrator
# Enter domain admin password, it should return to the prompt with no
errors

klist
# Credentials cache: FILE:/tmp/krb5cc_0
#   Principal: administrator@SVROS.COM
#
# Issued           Expires           Principal
# Dec  6 10:15:39 2021  Feb  4 20:15:39 2021  krbtgt
```

测试 Winbind

```
wbinfo -u  
# Should return domain users

wbinfo -g  
# Should return domain groups

getent passwd  
# Should return domain users at the end of the list with 10000+ UIDs

getent group  
# Should return domain groups at the end of the list with 10000+ GIDs
```

如果 wbinfo 命令显示报错，请执行命令

```
# service samba_server restart
```

创建共享文件夹

```
# mkdir -p /samba/testshare  
# chown "administrator":"domain users" /samba/testshare  
# chmod 0770 /samba/testshare
```

如果只允许属主可读可写，属组只允许读，用以下命令设置

```
# chmod 0750 /samba/testshare
```

如果只允许属主可读可写，属组和其他均不可读写，用以下命令设置

```
# chmod -R 0700 /samba/testshare
```

第八节 NFS 服务器

第九节 iSCSI

第十节 Webmin

```
# pkg install webmin #安装 webmin  
# /usr/local/lib/webmin/setup.sh #启动 webmin安装向导, SSL需要配置开启。
```

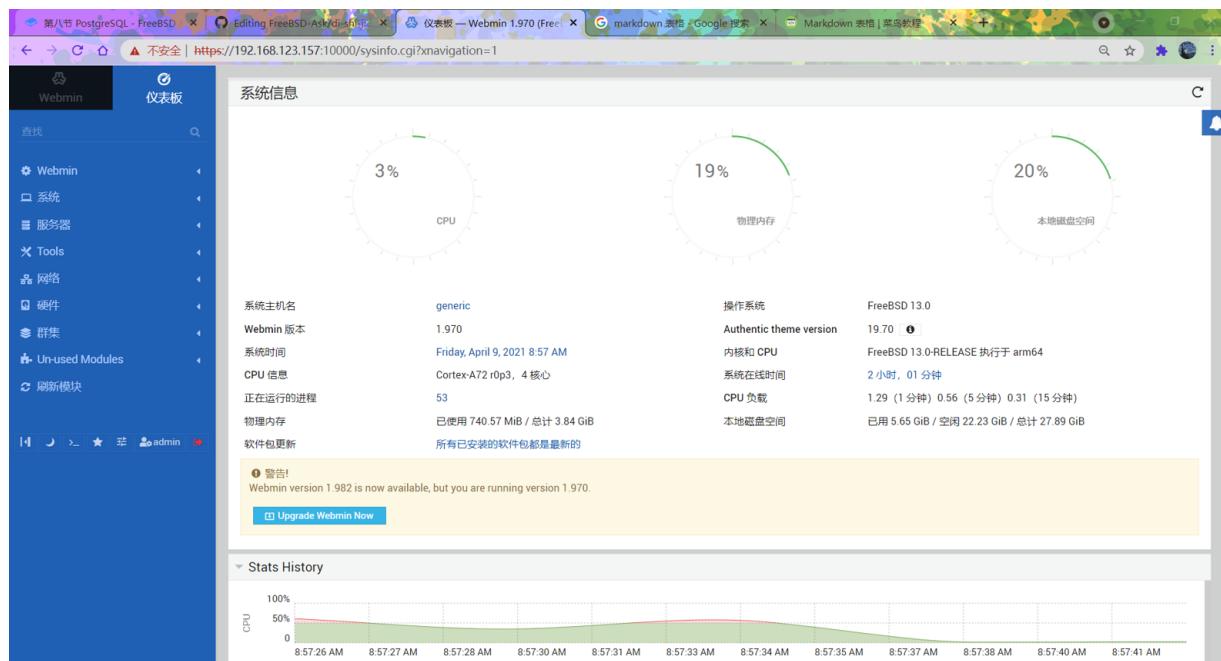
```
# sysrc webmin_enable="YES"  
# service webmin start
```

下面打开浏览器输入 `https://Ip:10000` , 如

`https://192.168.123.157:10000`

回车后如浏览器提示不安全，选择“继续前往”即可，之后会出现 Webmin 登录界面。

这是管理控制台。在文本框中输入 `admin` 和密码，点击 `Sign In`，登录进入控制台。

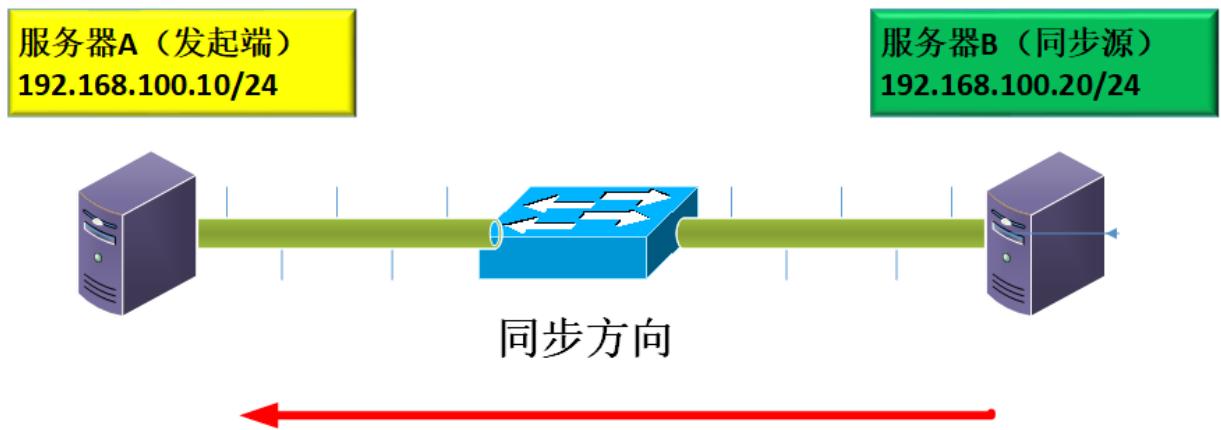


切换中文：

Webmin → Change Language and Theme，在 **Change Language and Theme** 的 **Webmin UI language** 字段中，先选择 **Personal choice** 再选择 **Simplified Chinese(ZH_CN.UTF8)** 点击 **Make Changes** 按钮，然后点击菜单 →Dashboard 控制台会刷新为中文界面。

第十一节 rsync 同步服务

Rsync备份服务器架构图



环境介绍

服务器 A、服务器 B 均为 FreeBSD-12.2-RELEASE-amd64

服务器 A（发起端）： 192.168.100.10/24

服务器 B（同步源）： 192.168.100.20/24

需求：实现服务器B的数据同步到服务器A上

服务器B（同步源）配置

安装 rsync 软件包

```
# pkg install -y rsync
```

查询已安装 rsync 软件包的信息

```
# pkg info | grep rsync  
rsync-3.2.3
```

新建需要备份的文件夹 **test**，并且设置其属主为 **root**，以及在其内部新建测试文件

```
# mkdir test  
# chown root /home/test/  
# touch txt001 /home/test/
```

编辑 **rsyncd.conf** 文件

```
# ee /usr/local/etc/rsync/rsyncd.conf

uid = root //服务端操作系统的用户
gid = wheel //服务端操作系统的用户的组
use chroot = yes //禁锢在源目录
address = 192.168.100.20 //监听地址
port 873 //用于通信的TCP端口，缺省是873
log file = /var/log/rsyncd.log //日志文件位置
pid file = /var/run/rsyncd.pid //存档进程ID的文件位置
hosts allow = 192.168.100.0/24 //允许访问的客户机地址

[testcom] //共享模块名称，自定义的名称，不一定要与同步目录相同
path = /home/test //同步的目录名，必须是uid参数指定的用户和gid参数指定的组
comment = testcombackup //模块说明文字
read only = yes //是否为只读
dont compress = *.gz *.tgz *.zip *.z *.Z *.rpm *.deb *.bz2 //同步时不再压缩的文件类型

auth users = root //授权账户
secrets file = /etc/rsyncd_users.db //定义rsync客户端用户认证的密码文件
```

创建授权备份账户认证的密码文件

```
# ee /etc/rsyncd_users.db

root:12345678 //格式：授权账户用户名：密码
```

修改数据文件权限

```
# chmod 600 /etc/rsyncd_users.db
```

rsync 的服务名是 rsyncd，启动 rsync 服务程序

```
# rsync --daemon //启动服务  
# sysnc rsyncd_enable="YES" //设置开机自启动  
# /usr/local/etc/rc.d/rsyncd start //启动服务
```

查看 rsync 运行端口号

```
# sockstat | grep rsync  
root      rsync      1185  4  dgram   -> /var/run/logpriv  
root      rsync      1185  5  tcp4    192.168.100.20:873    *:*
```

防火墙放行 rsync 服务

```
# ee /etc/ipfw.rules

IPF="ipfw -q add"
ipfw -q -f flush

#loopback
$IPF 10 allow all from any to any via lo0
$IPF 20 deny all from any to 127.0.0.0/8
$IPF 30 deny all from 127.0.0.0/8 to any
$IPF 40 deny tcp from any to any frag

# statefull
$IPF 50 check-state
$IPF 60 allow tcp from any to any established
$IPF 70 allow all from any to any out keep-state
$IPF 80 allow icmp from any to any

# open port for ssh
$IPF 110 allow tcp from any to any 22 out
$IPF 120 allow tcp from any to any 22 in

# open port for rsync
$IPF 130 allow tcp from any to any 873 in

# deny and log everything
$IPF 500 deny log all from any to any
```

服务器 A (发起端) 配置

创建本地文件夹 **/home/testBackUp/** 并设置好相关权限

```
# mkdir testBackUp  
# chown root:root testBackUp
```

发起端访问同步源，将文件下载到本地
/home/testBackUp/ 下载目录下，需要人机交互
手动输入密码

```
# rsync -avz root@192.168.100.20::testcom /home/testBackUp
```

查看同步情况

```
# ls -l /home/testBackUp/  
total 0  
-rw-r--r-- 1 root root 0 Feb  2 22:27 txt001
```

第十二节 时间服务

- 配置时区
- 同步时间

选择正确时区

安装操作系统的时候选择正确的时区

配置时区

```
# cp /usr/share/zoneinfo/Asia/Shanghai /etc/localtime
```

配置同步时间

设置 ntp 服务可用

```
# sysrc ntpd_enable="YES"
```

设置 ntp 服务开机时候启动

```
# sysrc ntpd_sync_on_start="YES"
```

编辑 **ntp.conf**文件

```
# ee /etc/ntp.conf
```

添加附加时钟服务器

```
server time.windows.com
server 0.cn.pool.ntp.org
server 1.cn.pool.ntp.org
server 2.cn.pool.ntp.org
server 3.cn.pool.ntp.org
```

设置 ntp 服务开机自启动（上面已设置，此处可选）

```
# /etc/rc.d/ntp enable  
或  
# service ntpd enable
```

启动 ntp 服务

```
# /etc/rc.d/ntp start  
或  
# service ntpd start
```

重启 ntp 服务

```
# /etc/rc.d/ntp restart  
或  
# service ntpd restart
```

显示当前时间

```
# date
```

手动同步时间（可选）

```
# ntpdate time.windows.com
```

第十三节 Wildfly

第一节 Apache

安装

以下二选一

```
# cd /usr/ports/www/apache24/ && make install clean
```

或者

```
# pkg install apache24
```

启动服务

```
# 添加服务开机自启  
# sysrc apache24_enable=YES  
  
# 启动服务  
# service apache24 start  
  
# 查看状态  
# service apache24 status
```

按理来说，apache 服务已经启动了，现在可以打开网址 `localhost` 看一下。

第二节 Nginx

安装

- pkg: # pkg install nginx

或

- ports: # cd /usr/ports/www/nginx/ && make install clean

查找相关的软件包

- ports: \$ ls /usr/ports/www/ | grep nginx
- pkg: \$ pkg search -o nginx

配置

配置教程可参阅 [官方文档](#) 与 [官方百科](#)。

本文仅简单说明 FreeBSD 中如何启动 Nginx 及 Nginx 的配置文件。

启动

```
# sysrc nginx_enable=YES  
# service nginx start
```

你可以通过 `$ sockstat -4 | grep nginx` 检查 nginx 是否启动并正常运行。

配置文件

FreeBSD中，Nginx 的配置文件位于 `/usr/local/etc/nginx/` 中，而主要的配置文件则在 `/usr/local/etc/nginx/nginx.conf`

默认配置中 Nginx 的根目录为 `/usr/local/www/nginx/`，如果需要更改目录位置，请将 `/usr/local/etc/nginx/nginx.conf` 中的

```
root /usr/local/www/nginx;
```

改成你想要的目录位置，例如 `root /path/to/new/webroot;`

示例文件（Nginx + Typecho 伪静态
+ SSL）

```
#user    nobody;
worker_processes  1;

# This default error log path is compiled-in to make sure
configuration parsing
# errors are logged somewhere, especially during unattended boot when
stderr
# isn't normally logged anywhere. This path will be touched on every
nginx
# start regardless of error log location configured here. See
# https://trac.nginx.org/nginx/ticket/147 for more info.
#
#error_log  /var/log/nginx/error.log;
#
#pid        logs/nginx.pid;

events {
    worker_connections  1024;
}

http {
    include      mime.types;
    default_type application/octet-stream;

    #log_format  main  '$remote_addr - $remote_user [$time_local]'
"$request"
    #
    #                      '$status $body_bytes_sent "$http_referer" '
    #                      '"$http_user_agent" "$http_x_forwarded_for"'; 

    #access_log  logs/access.log  main;

    sendfile      on;
```

```
#tcp_nopush      on;

#keepalive_timeout  0;
keepalive_timeout  65;

#gzip  on;

server {
    listen      80;
    server_name localhost;

    #charset koi8-r;

    #access_log  logs/host.access.log  main;

    location / {
        root   /usr/local/www/nginx;
        index index.html index.htm index.php;
        if (-f $request_filename/index.html){
            rewrite (.*) $1/index.html break;
        }
        if (-f $request_filename/index.php){
            rewrite (.*) $1/index.php;
        }
        if (!-f $request_filename){
            rewrite (.*) /index.php; }
    }

location ~ \.php(/.*)*$ {
        root           /usr/local/www/nginx;
        fastcgi_pass  127.0.0.1:9000;
        fastcgi_index index.php;
        fastcgi_param SCRIPT_FILENAME $request_filename;
        include        fastcgi_params;
    }
```

```
#location ~ .*\.php(.*\*$) {
    #           include fastcgi_params;
    #           fastcgi_pass 127.0.0.1:9000;
    #       }

#error_page 404 /404.html;

# redirect server error pages to the static page /50x.html
#
error_page 500 502 503 504 /50x.html;
location = /50x.html {
    root /usr/local/www/nginx-dist;
}

# proxy the PHP scripts to Apache listening on 127.0.0.1:80
#
#location ~ .php$ {
#    proxy_pass http://127.0.0.1;
#}

# pass the PHP scripts to FastCGI server listening on
127.0.0.1:9000
#
location ~ .*\.php(.*\*$) {
    root /usr/local/www/nginx;
    fastcgi_pass 127.0.0.1:9000;
    fastcgi_index index.php;
    fastcgi_param SCRIPT_FILENAME $request_filename;
    include fastcgi_params;
}
```

```
# deny access to .htaccess files, if Apache's document root
# concurs with nginx's one
#
#location ~ /\.ht {
#    deny all;
#}
}

# another virtual host using mix of IP-, name-, and port-based
configuration
#
#server {
#    listen      8000;
#    listen      somename:8080;
#    server_name somename alias another.alias;

#    location / {
#        root  html;
#        index index.html index.htm;
#    }
#}

# HTTPS server
#
server {
    listen      443;
    server_name localhost;

    ssl_certificate      /usr/local/etc/nginx/fbxs.crt;
    ssl_certificate_key  /usr/local/etc/nginx/fbxs.key;

    ssl on;
    ssl_certificate fbxs.crt;
    ssl_certificate_key fbxs.key;
```

```
ssl_session_timeout 5m;
ssl_protocols TLSv1 TLSv1.1 TLSv1.2; #按照这个协议配置
ssl_ciphers ECDHE-RSA-AES128-GCM-
SHA256:HIGH:!aNULL:!MD5:!RC4:!DHE;#按照这个套件配置
ssl_prefer_server_ciphers on;
location ~ .*\.php(/.*)*$ {
    root          /usr/local/www/nginx-dist;
    fastcgi_pass  127.0.0.1:9000;
    fastcgi_index index.php;
    fastcgi_param SCRIPT_FILENAME $request_filename;
    include        fastcgi_params;
}
location / {
    root    /usr/local/www/nginx-dist;
    index  index.php;
}
}

}
```

第三节 PHP 8.X

第四节 MySQL 5.X

mysql 5.5 (请参考 5.6) /5.6

安装

pkg、ports 安装二选一

```
# pkg install mysql56-server  
# 或者  
# cd /usr/ports/databases/mysql56-server/ && make install clean
```

启动服务

```
# sysrc mysql_enable=YES  
# service mysql-server start
```

配置

```
# mysql_secure_installation
```

输出：

```
root@ykla:~ # mysql_secure_installation
```

NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MySQL SERVERS IN PRODUCTION USE! PLEASE READ EACH STEP CAREFULLY!

In order to log into MySQL to secure it, we'll need the current password for the root user. If you've just installed MySQL, and you haven't set the root password yet, the password will be blank, so you should just press enter here.

Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password ensures that nobody can log into the MySQL root user without the proper authorisation.

Set root password? [Y/n] y
New password:
Re-enter new password:
Password updated successfully!
Reloading privilege tables..
... Success!

By default, a MySQL installation has an anonymous user, allowing anyone to log into MySQL without having to have a user account created for them. This is intended only for testing, and to make the installation go a bit smoother. You should remove them before moving into a production environment.

Remove anonymous users? [Y/n] y

```
... Success!
```

Normally, root should only be allowed to connect from 'localhost'.

This

ensures that someone cannot guess at the root password from the network.

```
Disallow root login remotely? [Y/n] n
```

... skipping.

By default, MySQL comes with a database named 'test' that anyone can access. This is also intended only for testing, and should be removed

before moving into a production environment.

```
Remove test database and access to it? [Y/n] n
```

... skipping.

Reloading the privilege tables will ensure that all changes made so far

will take effect immediately.

```
Reload privilege tables now? [Y/n]
```

... Success!

All done! If you've completed all of the above steps, your MySQL installation should now be secure.

Thanks for using MySQL!

Cleaning up...

使用

登录

```
# mysql -u root -p
```

示例输出

```
root@ykla:~ # mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or g.
Your MySQL connection id is 12
Server version: 5.6.51 Source distribution

Copyright (c) 2000, 2021, Oracle and/or its affiliates. All rights
reserved.
```

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or 'h' for help. Type 'c' to clear the current input
statement.

```
mysql> show databases;
+-----+
| Database      |
+-----+
| information_schema |
| mysql          |
| performance_schema |
| test           |
+-----+
4 rows in set (0.00 sec)
```

```
mysql>
```

mysql 5.7

注意：如果是从旧版本升级，请先执行 `mysql_upgrade`

安装

以下二选一

```
# pkg install mysql57-server  
# 或  
# cd /usr/ports/databases/mysql57-server/ && make install clean
```

启动服务

```
# sysrc mysql_enable=YES  
# service mysql-server start
```

示例输出(可以看到密码在 `/root/.mysql_secret` 文件夹下，是
`q(<p2ZZ>1X/:`

```
root@ykla:~ # sysrc mysql_enable=YES
mysql_enable: -> YES
root@ykla:~ # service mysql-server start
Starting mysql.
root@ykla:~ # cat /root/.mysql_secret
# Password set for user 'root@localhost' at 2021-12-13 00:21:02
q(<p2ZZ>1X/:
root@ykla:~ #
```

尝试登录

登录出现报错，提示需要修改密码。

```
root@ykla:~ # mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or g.
Your MySQL connection id is 2
Server version: 5.7.36-log

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or 'h' for help. Type 'c' to clear the current input
statement.

root@localhost [(none)]> show databases;
ERROR 1820 (HY000): You must reset your password using ALTER USER
statement before executing this statement.
root@localhost [(none)]>
```

修改密码

将现在的密码修改为 `your_new_password` , 然后刷新权限。

```
root@localhost [(none)]> SET PASSWORD =
PASSWORD('your_new_password');
Query OK, 0 rows affected, 1 warning (0.00 sec)
root@localhost [(none)]> flush privileges;
Query OK, 0 rows affected (0.00 sec)
```

正常登录

```
root@ykla:~ # mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or g.
Your MySQL connection id is 3
Server version: 5.7.36-log Source distribution
```

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

Type 'help;' or 'h' for help. Type 'c' to clear the current input statement.

```
root@localhost [(none)]> show databases;
+-----+
| Database      |
+-----+
| information_schema |
| mysql          |
| performance_schema |
| sys            |
+-----+
4 rows in set (0.00 sec)
```

```
root@localhost [(none)]>
```

第五节 MySQL 8.X

mysql 80

安装（以下二选一）

```
# pkg install mysql80-server
```

或者

```
# cd /usr/ports/databases/mysql80-server/ && make install clean
```

启动服务

```
# sysrc mysql_enable=YES  
# service mysql-server start
```

登录

mysql 8.0 默认密码是空，直接回车即可。

```
root@ykla:~ # mysql -uroot -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or g.
Your MySQL connection id is 8
Server version: 8.0.27 Source distribution

Copyright (c) 2000, 2021, Oracle and/or its affiliates.
```

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

Type 'help;' or 'h' for help. Type 'c' to clear the current input statement.

```
root@localhost [(none)]>
```

修改密码

设置密码为 `z`，然后刷新权限。

```
root@localhost [(none)]> alter user 'root'@'localhost' identified by
'z';
Query OK, 0 rows affected (0.02 sec)

root@localhost [(none)]> flush privileges;
Query OK, 0 rows affected (0.00 sec)
```

重新登录：

```
root@localhost [(none)]> quit;
Bye
root@ykla:~ # mysql -uroot -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or g.
Your MySQL connection id is 9
Server version: 8.0.27 Source distribution
```

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

Type 'help;' or 'h' for help. Type 'c' to clear the current input statement.

```
root@localhost [(none)]> show databases;
+-----+
| Database      |
+-----+
| information_schema |
| mysql          |
| performance_schema |
| sys            |
+-----+
4 rows in set (0.01 sec)
```

```
root@localhost [(none)]>
```

第六节 Typecho

第七节 SSL 配置

传输层安全性协议（英语：Transport Layer Security，缩写：TLS）及其前身安全套接层（英语：Secure Sockets Layer，缩写：SSL）是一种安全协议，目的是为互联网通信提供安全及数据完整性保障。网景公司（Netscape）在1994年推出首版网页浏览器—网景导航者时，推出 HTTPS 协议，以 SSL 进行加密，这是 SSL 的起源。IETF 将 SSL 进行标准化，1999年公布 TLS 1.0 标准文件（RFC 2246）。随后又公布 TLS 1.1（RFC 4346，2006 年）、TLS 1.2（RFC 5246，2008年）和 TLS 1.3（RFC 8446，2018年）。在浏览器、电子邮件、即时通信、VoIP、网络传真等应用程序中，广泛使用这个协议。目前已成为互联网上保密通信的工业标准。

第八节 PostgreSQL 与 pgAdmin4

PostgreSQL

PostgreSQL 是一款自由的对象-关系型数据库，最早发布于 1989 年 6 月。在 FreeBSD 上，提供了 9.6、10、11、12、13、14 共计 6 个大版本可选。

postgresql 安装示例，6个版本都如此。

安装：二选一

```
# pkg install -y postgresql96-server
```

或者

```
cd /usr/ports/databases/postgresql96-server/ && make install clean
```

加入启动项

```
# sysrc postgresql_enable=YES
```

初始化数据库

```
/usr/local/etc/rc.d/postgresql initdb
```

示例输出：

```
root@ykla:~ # /usr/local/etc/rc.d/postgresql initdb
The files belonging to this database system will be owned by user
"postgres".
This user must also own the server process.
```

The database cluster will be initialized with locales

```
COLLATE: C
CTYPE: C.UTF-8
MESSAGES: C.UTF-8
MONETARY: C.UTF-8
NUMERIC: C.UTF-8
TIME: C.UTF-8
```

The default text search configuration will be set to "english".

Data page checksums are disabled.

```
creating directory /var/db/postgres/data96 ... ok
creating subdirectories ... ok
selecting default max_connections ... 100
selecting default shared_buffers ... 128MB
selecting default timezone ... PRC
selecting dynamic shared memory implementation ... posix
creating configuration files ... ok
running bootstrap script ... ok
performing post-bootstrap initialization ... ok
syncing data to disk ... ok
```

```
WARNING: enabling "trust" authentication for local connections
You can change this by editing pg_hba.conf or using the option -A, or
--auth-local and --auth-host, the next time you run initdb.
```

Success. You can now start the database server using:

```
/usr/local/bin/pg_ctl -D /var/db/postgres/data96 -l logfile start
```

```
root@ykla:~ #
```

登录使用

Postgresql 默认是没有 root 用户的，需要使用其创建的 postgres 用户登录。

示例输出：

```
root@ykla:~ # psql
psql: FATAL:  role "root" does not exist
```

正确用法：

```
#切换用户
root@ykla:~ # su - postgres

#启动服务
$ /usr/local/bin/pg_ctl -D /var/db/postgres/data96 -l logfile start

#创建新用户 ykla，并设置密码
$ createuser -sdrP ykla
Enter password for new role:
Enter it again:
$

#创建数据库
$ createdb new_db
#登录进数据库并将数据库权限赋予用户 ykla。
$ psql
psql (9.6.24)
Type "help" for help.

postgres=# ALTER USER ykla WITH ENCRYPTED PASSWORD 'password';
ALTER ROLE
postgres=#
postgres=# GRANT ALL PRIVILEGES ON DATABASE new_db TO ykla;
GRANT
#退出数据库
postgres=# q
$exit
root@ykla:~ #
```

安装 pgAdmin4

以下教程以 FreeBSD 13.0 为基准。

pgAdmin4 是用于管理 PostgreSQL 数据库服务器的最流行的开源应用程序。pgAdmin4 提供功能丰富的图形用户界面，轻松管理数据库。它是用 Python 和 Javascript / jQuery 编写的。它可以在多种环境中使用，如 Linux, Windows, Unix，可在桌面和服务器模式下使用。

注意：在安装 pgAdmin4 前先行安装 PostgreSQL 数据库，否则安装 pgAdmin4 会失败。

pgAdmin4 需要在 python 环境下运行，并且安装时要通过 python 的 pip 进行安装，所以先安装 python。本文用的默认版本是 Python3.8，请注意，FreeBSD 13 系统上默认没有 python 环境。可通过以下命令查看：

```
# python  
python: Command not found #说明当前没有 python 命令
```

安装 Python 及 pip

```
# pkg install python
```

pip 是 Python 包的包管理器。它用于安装和管理 Python 包和依赖包的关系。

virtualenv 用来建立一个虚拟的 python 环境，一个专属于项目的 python 环境。

本文实际安装过程中是通过 virtualenv 创建独立的 Python 环境来安装 pgAdmin4。

从 py38-pip 包安装 pip:

```
# pkg install py38-pip
```

安装配置 virtualenv

使用 virtualenv 创建独立的 Python 环境。Virtualenv 会创建一个自己的 Python 安装的环境，它不支持具有全局或另一个虚拟环境的库。运行以下命令来安装 Virtualenv。

```
# pkg install py38-virtualenv (本次安装 python 版本是 3.8 故使用  
py38)
```

通过运行以下命令为 pgAdmin4 创建虚拟环境

```
# virtualenv-3.8 pgadmin4
```

如果创建完成则有如下显示，在 root 用户的根目录下生成了一个名为 pgadmin4 的虚拟环境。

```
Using base prefix '/usr/local'  
New python executable in /home/vagrant/pgadmin4/bin/python3.8  
Also creating executable in /home/vagrant/pgadmin4/bin/python  
Installing setuptools, pip, wheel...done.  
done.
```

安装 sqlite3

```
#pkg install py38-sqlite3
```

激活创建的虚拟环境。

```
#source pgadmin4/bin/activate.csh
```

你会看到 shell 已经变为 (pgadmin4) (以下操作均在该 shell 下进行)

实例如下：

```
(pgadmin4) root@ykla:~ #
```

安装 pgAdmin4:

现在 pip 源一律要求使用 https, 由于缺少 SSL 证书还需要安装。

```
(pgadmin4) root@ykla:~# pkg install ca_root_nss
```

然后对 pip 进行换源，此处使用清华源：

```
pip config set global.index-url  
https://pypi.tuna.tsinghua.edu.cn/simple
```

其中有依赖 openjpeg，先进行安装

```
(pgadmin4) root@ykla:~# pkg install openjpeg
```

如果报错：

```
WARNING: Retrying (Retry(total=3, connect=None, read=None,  
redirect=None, status=None)) after connection broken by  
'SSLError(SSLCertVerificationError(1, '[SSL:  
CERTIFICATE_VERIFY_FAILED] certificate verify failed: certificate is  
not yet valid (_ssl.c:1136)'))': /simple/pgadmin4/
```

是由于时间不正确引发的，先同步时间：

```
ntpdate ntp.api.bz  
(pgadmin4) root@ykla:~# ntpdate ntp.api.bz  
17 Dec 16:35:36 ntpdate[1453]: step time server 114.118.7.161 offset  
+401965.911037 sec
```

然后再安装 pgAdmin4 及其依赖环境 rust：

```
(pgadmin4) root@ykla:~# pkg install rust  
(pgadmin4) root@ykla:~# pip install pgadmin4
```

注意：如果内存不足（小于 4GB）且没有 swap，会提示 killed，如出现该问题请先添加一块 swap。

配置并运行 pgAdmin4

安装完成后为 pgAdmin4 创建配置文件，复制 pgAdmin4 配置文件：

```
(pgadmin4) root@ykla:~# cp ./pgadmin4/lib/python3.8/site-packages/pgadmin4/config.py ./pgadmin4/lib/python3.8/site-packages/pgadmin4/config_local.py
```

使用 ee 编辑器编辑配置文件的本地副本。

```
(pgadmin4) root@ykla:~# ee ./pgadmin4/lib/python3.8/site-packages/pgadmin4/config_local.py
```

找到 `DEFAULT_SERVER` 将默认服务器侦听地址更改为 `0.0.0.0`。找到 `DEFAULT_SERVER_PORT` 可改应用程序监听的端口。

实例如下：

```
DEFAULT_SERVER = '0.0.0.0'  
DEFAULT_SERVER_PORT = 5050
```

手动创建软件目录：

```
(pgadmin4) root@ykla:~# mkdir -p /var/lib/pgadmin  
(pgadmin4) root@ykla:~# mkdir /var/log/pgadmin
```

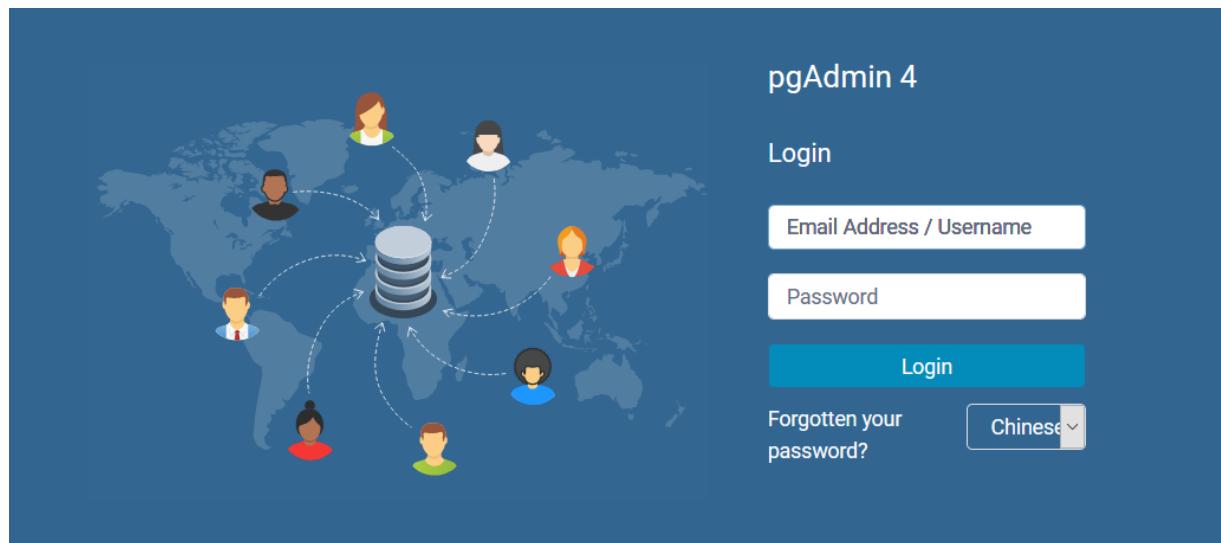
配置文件编辑完成后执行以下命令来初始化账号和登录密码。

```
(pgadmin4) root@ykla:~# pgadmin4
```

显示实例如下：

```
NOTE: Configuring authentication for SERVER mode.  
Enter the email address and password to use for the initial pgAdmin  
user account:  
Email address: your_email          //输入你的邮件地址  
Password: your_new_password      //输入你的登录密码  
Retype password:                 //再次输入密码  
Starting pgAdmin 4. Please navigate to http://0.0.0.0:5050 in your  
browser.
```

现在我们已经安装并运行了 pgAdmin4，并可以通过 <http://ip:5050> 访问 Web 控制面板：



The screenshot shows the pgAdmin 4 welcome page. At the top, there are three browser tabs: 'pgadmin4 The headers or lib...', 'find命令全焉查找~Google 搜索' (which is a search result for 'find command'), and 'pgAdmin 4'. The main window has a dark blue header bar with the pgAdmin logo and a user profile picture. Below the header is a navigation menu with '文件' (File), '对象' (Object), '工具' (Tools), and '帮助' (Help). A 'Servers' dropdown is open, showing one entry: '1'. The main content area is titled '欢迎' (Welcome) and features the pgAdmin logo and the text 'Management Tools for PostgreSQL'. It highlights '特性丰富 | 最强的PostgreSQL | 开源' (Rich features | Strongest PostgreSQL | Open source). Below this, a section titled '快速链接' (Quick Links) includes icons for '添加服务器' (Add Server) and '配置 pgAdmin' (Configure pgAdmin). The '入门指南' (Getting Started) section contains four links: 'PostgreSQL 文档' (PostgreSQL Documentation), 'pgAdmin 主页' (pgAdmin Home), 'PostgreSQL 星球' (PostgreSQL Planet), and '社区支持' (Community Support).

This screenshot shows the pgAdmin 4 interface with a database tree on the left. The tree is expanded to show two databases: 'new_db' and 'postgres'. Under 'new_db', there are several objects: '事件触发器' (Event Triggers), '外部数据封装器' (External Data Wrappers), '强制转换' (Type Casting), '扩展' (Extensions), '架构' (Schemas), '目录' (Directories), and '语言' (Languages). Under 'postgres', there are similar objects: '事件触发器' (Event Triggers), '外部数据封装器' (External Data Wrappers), '强制转换' (Type Casting), '扩展' (Extensions), '架构' (Schemas), '目录' (Directories), and '语言' (Languages). At the top, there are three browser tabs: 'pgadmin4 The headers or lib...', '进入 PostgreSQL - FreeBSD' (which is a search result for 'PostgreSQL'), and 'pgAdmin 4'. The main window has a dark blue header bar with the pgAdmin logo and a user profile picture. Below the header is a navigation menu with '文件' (File), '对象' (Object), '工具' (Tools), and '帮助' (Help). A 'Servers' dropdown is open, showing one entry: '1'. The main content area is identical to the welcome page, featuring the pgAdmin logo and the text 'Management Tools for PostgreSQL'. It highlights '特性丰富 | 最强的PostgreSQL | 开源' (Rich features | Strongest PostgreSQL | Open source). Below this, a section titled '快速链接' (Quick Links) includes icons for '添加服务器' (Add Server) and '配置 pgAdmin' (Configure pgAdmin). The '入门指南' (Getting Started) section contains four links: 'PostgreSQL 文档' (PostgreSQL Documentation), 'pgAdmin 主页' (pgAdmin Home), 'PostgreSQL 星球' (PostgreSQL Planet), and '社区支持' (Community Support).

保持 pgAdmin4 后台运行

如果服务关闭下次要运行时需使用 pgadmin4 的安装用户（此处是 `root`）进入根目录，执行如下命令：

```
root@ykla:~# source pgadmin4/bin/activate.csh  
(pgadmin4) root@ykla:~# pgadmin4 &
```

提示： `&` 表示后台运行

服务启动后在当前界面中输入 `&` 按回车键，可切换至前台命令行，让服务程序在后台运行。

升级 pgAdmin4

本文测试如果直接使用 pip 升级后还是提示旧版本。

pgadmin4 更新频率较高，如需升级要先删除原有用 virtualenv 创建的 pgadmin4 目录然后用安装用户再次执行如下指令：

```
root@ykla:~# virtualenv-3.8 pgadmin4
```

虚拟目录创建完成后激活

```
root@ykla:~# source pgadmin4/bin/activate.csh
```

激活后不要开启服务，直接执行升级

```
(pgadmin4) root@ykla:~# pip install --upgrade pgadmin4
```

完成升级后启动服务

```
(pgadmin4) root@ykla:~# pgadmin4
```

登录帐号和密码还是原来的(登陆后再无更新提示，查看版本已是为最新)。

第一节 树莓派简介

树莓派是什么，相信凡是关注了我们的人都不会不知道，但是介于非专业人员需要在此做简要介绍。我们的安卓手机，大部分的 ARM 芯片的，即使歌曲 华为美 所谓的“中国芯”也是来源于 ARM 的专利授权，没了授权，什么芯也不是。

而树莓派就是一块使用 ARM 芯片的开发板，就是一块接口丰富（HDMI 、 I2C 、 USB X 4 、 POE 模块等等）的电路板，相当于配置较低的手机，手机能做的事情，树莓派都能做。一般用于嵌入式开发，比如机器人，路由器和监控等等。

FreeBSD 对架构的支持是按照等级划分的， ARM 属于二级架构（FreeBSD 13 升级为一级架构对待），所以软件支持上不如 AMD64，一些软件无法通过 ports 以源码的形式进行编译，比如编译 X11/xorg 就会遇到很多错误（已经报告 bug 列表）。

第二节 系统安装

自树莓派 3B+ 开始，无需任何改动，系统即可从 U 盘启动，我测试了 FreeBSD12/13 都是支持的，但是速度非常慢，一方面树莓派使用 USB2.0 极大的限制的总线速度，另一方面可能是玄学问题。（我人丑？我用的是东芝（ TOSHIBA ） 64GB USB3.0 U 盘 U364 高速迷你车载 U 盘）。

因此不建议使用 U 盘启动，慢的我要打年年猫，年年猫是谁？是一只调皮的狸花猫而已。

我们所有要准备的有树莓派 4 板子一块，网线一段，存储卡一枚。从 FreeBSD.org 下载适用于树莓派 4 的镜像：

<https://download.freebsd.org/ftp/releases/arm64/aarch64/ISO-IMAGES/13.0/FreeBSD-13.0-RELEASE-arm64-aarch64-RPI.img.xz>

下载后解压缩。使用 rufus 刻录。插入网线，将存储卡插入树莓派，通电等待约五分钟，查看路由器后台获取 IP 。

注意： 刻录完需要挂载 FAT 分区 替换里面的所有文件，否则会启动花屏，替换的文件路径为：

<https://github.com/FreeBSD-Ask/FreeBSD-rpi4-firmware>

使用 XShell 即可登录树莓派。用户名密码均为 `freebsd` 。root 需要登录后输入 `su` ，密码为 `root` 。

可通过更改 `/etc/ssh/sshd_config` 文件来开启 root 账户的 ssh 远程登录权限。

方法：

编辑 `/etc/sh/sshd_config` （注意是 sshd 不是 ssh ! 这是两个文件），修改或者加入

```
PermitRootLogin yes #允许 root 登录  
PasswordAuthentication yes # 设置是否使用口令验证。
```

（也可以把对应行前面的注释 # 去掉，注意 `PermitRootLogin` 一行默认是 no，去掉后要改成 yes 。即 `PermitRootLogin yes` ）。

然后重启服务：

```
# server sshd restart
```

然后就是时间设置问题，树莓派没有板载的纽扣电池确保 CMOS 时钟准确。所以完全依靠 NTP 服务来校正时间，如果时间不准确，将影响很多服务的运行，比如无法执行 portsnap fetch 命令。

方法很简单：

在 `/etc/rc.conf` 中加入

```
ntp_enable="YES"  
ntpdate_enable="YES"  
ntpdate_program="ntpdate"  
ntpdate_flags="0.cn.pool.ntp.org"
```

然后开启时间服务器：

```
# service ntpdate start
```

输入 `# date` 查看时间，完成校时。我国使用 UTC+8 北京时，虽然不更改不会影响软件使用，但看起来不方便，可通过 `# bsdconfig` 命令将地区调整到亚洲 /中国 /上海。

树莓派应该会自动接通互联网，所以不必考虑联网问题。

CPU 频率调整（600MHZ 到 1500MHZ）：

```
# sysrc powerd_enable="YES"
# service powerd restart
```

第三节 使用配置

第四节 USB 网卡与 WIFI

树莓派 3B+ 是目前配置比较高的型号，由于 FreeBSD 的 SDIO 驱动并未完全支持，所以板载的蓝牙和 WIFI 均无法使用（可购买无线网卡，推荐 16 块钱的 COMFAST CF-WU810N ）。

如果你购买了上述无线网卡，想实现开机自动连接 wifi 的功能，那也非常简单。

方法： `/boot/loader.conf` 中写入

```
rtwn_usb_load="YES"
legal.realtek.license_ack=1
```

在 `/etc/rc.conf` 中写入

```
wlans_rtwn0="wlan0"
ifconfig_wlan0="WPA DHCP"
```

注意在 `/etc/wpa_supplicant.conf` 文件中（没有就自己通过 `touch` 命令新建一个）写入

```
network={ ssid="wifi 名字，别搞什么中文" psk="密码" }
```

保存重启即可。能够实现开机自动连接 wifi 。

第五节 RISC-V

第一节 我与 FreeBSD 的故事

我与FreeBSD 的故事之一

记得还是那些无聊的日子，群里有网友称 Linux 只能玩 WPS，我表示质疑，并通过百度这个搜索引擎搜索到了 Ubuntu Kylin，即由湖南的国防科技大学 与 Ubuntu 社区合作并由其主导的 UbuntKylin 麒麟项目。

说到麒麟，本是祥瑞，就和龙一样具有中国特色，但是沦为资本的玩物，我深表遗憾。当时我按照其论坛的优盘刻录方法，用百度下载了软碟通，让我家人买了个 4G 的，USB 2.0 的金士顿优盘。刻录得非常慢。

我用的笔记本是联想 G400，i3-3110M，4G 内存，500G 机械硬盘，显卡就更是亮机卡 HD 4000+AMD 8570M。我很快就进入了安装界面，很不幸我的无线网卡是博通的 BCM43142，并不开源。但幸运的是 Ubuntu 内置了许多专有驱动，因此我得以摆脱绕成毛线团的廉价网线。此前我只有过 ghost 安装 Windows 的经历，并没有接触过任何其他操作系统。

进入了 Ubuntu Kylin，我并没有感到多么的惊奇，只是感觉有些反人类，窗口部件都和 Windows 相反，在左边而非右边。Unity 非常耗费系统资源，我这种配置经常卡顿，我开始安装其他软件，我证明了一件事，Linux 不止 WPS。此后我觉得有必要深入的研究一下，这是我

对自由的渴望，任何可以 hack 的机器我都愿意研究一二，Android Root，Kindle 刷安卓，苹果越狱，学习机破解安软件。我感觉这也许也是一种控制欲望，但实际上不是，我并非 root 敢死队，也许是 chrome 和 Ubuntu 的桌面对我的压制吧。

我并没有游戏为什么不能运行在 Linux 上的困扰，因为我并不偏好任何游戏。或许这种配置，根本运行不了任何值得去玩的游戏。

我一直在研究 Linux 的各个方面，但是对服务器方面缺乏研究，我只有理论，而缺乏实践，后来吃了很多苦。我这几天看到鸟哥的 Linux 服务器篇才发现，我这些年所苦学的，里面尽有。不得不喟叹，听人劝吃饱饭。因为我只学了第一本基础篇，我不相信只有这一本书能看，去了当地最大的图书批发市场，去找《The C language Program》和其答案，但是很遗憾，并没有。这么多年过去了，我曾经考了五次 NCRE，均未通过。于是开始思索什么东西。Ubuntu Kylin 非常的不稳定，于是我开始装原版 Ubuntu，然后我发现错怪了 kylin，Ubuntu 的确如此。后来几乎把所有发行版都装过并用过一段时间。

我极其厌恶社区一些人自大狂妄的嘴角，所以很少发言。我至今不懂的一个问题是，为什么用差劲的 CentOS 而不是 Scientific Linux，两者都是 RHEL 代码的重构发行版。后者远远优于 CentOS，甚至与 RHEL 互换源都没有问题，更妄论开发者水平，后者更多地是科研院所或者大学。这就是所谓盲目从众？大家抱着一种目的去学 Linux，总是想要学什么内核源码剖析，组成原理这类理论性更强的东西，或者针对的考红帽认证。可见的，每当大学开学，新学期，贴吧社区就会涌入很多新人，提出各种有趣的问题。而一些所谓的大佬只会不断地禁言。

我使用的博通网卡让我受了很多本不存在的困扰，所以我现在到哪都要喊一句“傻逼博通”，每次编译安装都是非常麻烦的，有很多代码要改，查询各种手册。而反观英特尔就非常方便，直接免驱。后来我查到联想 G400，或者说笔记本电脑都有一种叫白名单的玩意，我不愿承担刷 BIOS 的风险，于是和别人换了别的网卡，这款电脑同型号的有使用英特尔网卡。

因此现在我看到树莓派也用的博通我就气愤不已，不开源往开源产品里插什么手，徒增麻烦。“傻逼博通” x3。

我与FreeBSD 的故事之二

那些人的丑恶嘴脸使我发笑，我愈发远离所谓的社区与论坛。电视剧《武林外传》说的好：有的地方就有江湖，江湖从未走远，从未改变。社区中的冲突很少是技术层面的，按照老话说睿智的人很少发表自己的见解，只是默默的围观，而对于所谓的技术群就更多是有这群睿智的人们在围观。有技术的人会在群里？有个群友说的好，不怕和你谈技术，就怕你根本没技术还说要对谈技术的人说——少说废话。

技术要转化为生产力才能体现其价值这句话是这些人实用主义特征的彻底表现。有人和我说沉下心来研究一个系统，学一门编程语言而不是天天装系统。我非常不认同这种想法，这就是他们默守陈规的根源所在。系统是什么生产力吗？我和以往的态度没有任何变化，没有，不是，这只是苦难哲学，自己折磨自己，放着好用的不用，用所谓自由开放来给自己设了个圈子，就像二维世界的纸片人，再也跳不出去。

对于我个人而言，只要求系统满足两点，稳定与求新。大部分操作系统都不符合我这个要求，单单一个稳定，就把 Ubuntu 及其衍生版本除了 Mint linux 刷了下去，而求新，Archlinux 在我这里三天就要崩溃。我最终还是选择了 Gentoo Linux。我认为这是一个自然选择的过程，任何真正喜欢这些的人都会从 Ubuntu 到 RHEL 再经历些别的，最后待在 Gentoo，很多人使用 Linux 只是迫于无奈，其他根本无所谓喜不喜欢，他们认为这是孩子幼稚的想法，但究竟谁更幼稚呢？

我看到实验楼的 Linux 基础课程把 “FreeBSD” 说成是 Linux 的一个发行版，我联系他们，对方称会严肃批评制作课程的老师，并称自己也是 FreeBSD 的爱好者，这已经不是错误了，这是常识问题，由此我怀疑其课程质量，再也没有使用过实验楼了。

提到编程，我只会 C 语言和 JAVA，会，是指能写“你好，世界！”这种代码。我非常厌恶 VC++6.0. 也许不是微软的错误，是教育制度有问题，更多的是人有问题，还有各种 $a^{++} + ++a$ 的荒谬题目，误人子弟。我直接选择放弃，如果不能得到真理。机械工业出版社的经典大理石丛书没什么问题，就是没有多大意义。实际工作中，没有人会用到编译原理和算法导论这些东西。底层编程人员要做的只是搬运而非创造或者科研。要认清这一点很难。

当初我想创建一个学生社团，但是根本做不到。一是没人，二是没钱。试问连网都上不去的学校如何满足社团存在的条件呢？而做不到的事情自然没有意义来说什么东南西北。很多尝试都是失败的，这是很正常的事情，只能说明这件事发展还不到这个阶段，或者选择的环境不符合，而我的失败很明显属于这个。同心而同德，我想很难。也许根本不可能，我没有任何支援。

生命中复杂困难的事情多的是，我对这一点认识的很深刻，因此完全没有找什么借口，这是没有用的，至于鸡汤文学的什么为成功找方法的鬼话也是不可信的。在这以前，我只是想要找能说得上话的人，就像刘震云那本书《一句顶一万句》表达的思想一样，人这一辈子，能有个说得上话的人是非常幸运的。而我则更像里面的那个老传教士，在一个普通院校传递我的火炬。但是非常不幸，没有一个人是能够“说得上话”的。

FreeBSD 到了今天，除了当今形势，和社区的基础有着重要影响，我以前就说了，看你的目标人群是谁，如果外国人或者程序员，何必翻译 Handbook？这都不懂你没有必要继续下去了。但明显我不是这么想的，这也是为什么在今天我们 FreeBSD 式微的原因所在，有人都开始搞内核了，有人还不会装系统，虽然 FreeBSD 全都是点下一步就能装上。而我并不在意那群会内核的，而非开发者的人，我更多地在意什么都不会的人。

我第一次、第二次、第三次都没有能够顺利安装 FreeBSD。因为我用的是联想 G400，UEFI 下我根本无法安装，会花屏，但是与显存无关，至今为止，我也只是挂了个 Bug 报告，无人回复。因此我失去了最开始接触 FreeBSD 的机会，坠入了 Linux 的深渊。

我与FreeBSD 的故事之三

联想 G400 是我在国美电器线下买的笔记本。我什么也不懂，就随便买了，不随便也不行，谁都知道只要不是那种特别的奸商，基本上货物都是符合价值决定价格这个基本的经济学规律的。所以没钱就失去了选择的自由。到了现在，我还是没有这种自由，就算整个世界，有自由的人也不是很多。因此我现在还在受着 4G 内存的折磨。我想的很多，也做了很多；但失败的越多，想的就更多，做就显得弥足珍贵。

联想 G400 的 CMOS 是有问题的，在某个版本的 Kali 上，加载 UEFI 启动，会导致主板损毁。而联想的 BIOS 更新，也只在美国官网上才有，而台湾，香港地区则也是转跳过去下载。似乎是解决了这个问题，但耽误了我几乎几个月的时间。

联想为电脑预置了 Linpus Linux ，我的 G400 一开始就被国美电器的人给 Ghost 了。我想趁机恢复，因为我看到，软件保修一年。很久之后，联想致电我称电脑修好了，但是无法恢复 Linux，他们不知道那是什么东西。我发邮件给 Linpus，但是对方总是没有回应。直到半年后我才又想起了这件事，遂又发邮件。这次有个好像是客服的回答了我的一些问题，比如系统的稳定性(基于 fedora 使我不得不置疑)，桌面的帮助图标如何删除等等。但是还是没有回答我，如何恢复预置系统。直至一年后，我又从 QQ 邮件收件箱翻出来了这些邮件，再次联系，按照桃花源记的说法，“后遂无问津者”。

后来我才从一些所谓大佬的只言片语中才了解到什么美帝联想，目前我已经彻底拉黑了 Lenovo。不过总的来说，还是贫穷害了我。4G 内存连 PS 都打不开，更别说什么了。

很多人觉得我是幸运的，至少还有电脑，但是更多地是误解与荒谬。我想起来一个笑话，一个公司招聘，在员工福利那标注“为所有苹果开发工程师提供 iMac”，我就想笑了，开发苹果不用苹果的设备用什么，Windows 吗？难不成是黑苹果？很明显这种公司是有坑的。不用计算机也能学习计算机，这么想的人脑子大概也是有了坑。这种类比推理在这里还是很好用的。

我在思考一个简单的问题，不是我的时间都去哪了，而是我的童年在哪里？或者更深切地说，我的青春在哪里？似乎是各种试卷，各种模拟考试，也许还有各种补课，从来如此就是不对的，但是我们的抗争是无用的。其制度就决定了一切，再难动迁。而被牺牲的，就成为所谓社会的螺丝钉，被环境的锤子砸进去，看谁都像钉子，只要和自己不一样就要把他锤进去。而真正享乐的人儿呢？恐怕我真的是个被牺牲的人。

我从未觉得孤独，虽然我一贯独行，伟大的精神是支撑我的动力所在。为此我有非常严重的精神疾病，医生为我开了500多的药，但是我最终没有拿着处方笺去药房取。很多的，包容的社会，是我所希望的，但是非常遗憾，我们得不到任何外在事物对我的任何的支持。此时我感觉的不是孤独，也不是绝望，而是欣喜。因为绝望到了顶点，其高峰往上就是如此。

也许就是对自由的追求，使我离开了 Linux，走向了 BSD。Linux 带给我很多的只是苦难哲学，而非什么内核技术分析，协议实验，服务安装，红帽 RHCA 这类东西。人永远不是工具，绝对不是，不能是螺

丝钉，否则就不是人。人存在的精神全在人的价值上，如果自我认同自己有工具属性，那就和那些人一样了：碌碌无为，用繁重的工作和社交灌满自己的日程表，拒绝任何思考，还说自己很忙，不要打扰他，并嘲笑有理想的人劝其不要想太多，多做实事，勿空谈。人非工具。操作系统的工具属性是与生俱来的，所以 OS 是不是一种工具？

要讨论什么狗屁技术吗？那些机械工业出版社的黑砖头多的是。什么是技术？炫耀还是什么？我不是很懂。至于学习？恐怕还不是时候。预计要将手册翻译完毕才可以。那些又没有人看也不具有现实意义，我并不想把所有人当作工具来使用，当作鸭子来灌输知识，永远不要祈求他人和你一样，那不是人道主义的，也不现实。

我与FreeBSD 的故事之四

FreeBSD 已经 28 岁了(2021)。但是图形化上仍然很差劲，开源界都这样。按照软件工程的说法就是缺乏用户需求分析。就像一些人说的，你就天天写代码，不要管有什么用。在这种思想的指导下，更多地垃圾软件和不友好的软件被制造了出来，有的是造轮子。上层理论已经几乎独立于底层，如果说你能从 C 的指针和数组中看到汇编的影子，那么今天已经没有了。所以我并不认为内核分析对我们使用 FreeBSD 有多少帮助，如果我并不是开发者。毕竟现在是市场经济，任何衡量价值的方法都是价格。又把人视为工具，如果老板挽留你，很大程度上是因为你是他所有可选项中成本最低的那个。

现在有些人是民粹主义，我一直说人民邮电出版社耽误了大陆计算机事业的发展。众所周知入门 Linux 就那一本书 - 鸟哥的 Linux 私房菜。这书四年前就该出来，一直推到了现在。但是拿到新版书，第四版的我非常遗憾。书的封面写着“基于 Linux 7.x”，我不由怀疑其质量了，Linux 版本号还没有到 6，这就 7 了？译者称已经返厂重印，并看不到任何歉意地称可以当错版人民币这种东西来收藏。而有人说这是鸟哥的政治立场有问题却又拿不出来任何证据，这就是人云亦云。

自学有个问题，就是自己以为自己会了，其实不会。或者与所谓主流教育考核方式不符。很多人把自然科学的东西往社会科学里套，类比推理，这也是我最厌恶的一个地方。很多民粹主义就如此。很多人过

于强调个人能动性，完全不在意恶劣的环境，这也是一种民粹主义体现，认为人多就代表真理。凡是不这样想的就被扣上帽子。认为是个人不努力，这完全是资本家在讽刺一个乞丐那么饿为什么不吃肉。

我不知道问题出在哪里，我也不考虑有没有救，这不是麻木，而是无奈。也无可批判什么。

回到网络互助上，我不知道如何对待，我现在更多地是围观，做个“智者”（见故事三）。我受到了很多伤害，我曾经在互联网上无偿的帮人安装操作系统和一些故障排除。但是实际上我什么也没有得到，没有人认为这些简单的事情对学术研究有什么意义。我只是浪费时间而已，而那群智者，一旦当我无用时，就忽视我。仿佛我是一个工具。这种事情发生的太多了，不只是少数人，我就自己安慰自己，我知道丛飞的故事，也能直面他的想法。当初的目的不是为了得到什么，但是为什么会无视呢？我以前说过，免费的东西往往如此。一个东西必须付费才能体现其价值，即使只收一分钱，所以我明白了一些商品为什么要付费试用。这些是国内经济学课本上所没有的。一言以蔽之，FreeBSD 的价值就是由于缺乏像 Linux 那样的商业化而失败。我也是个“智者”，就在人群中潜水，从不发表自己的意见，就默默地围观。

“智者”太多了，这个组织就死光了。而理由就更简单了 - “自己很忙，在上班，没空”。好像自己比美国总统特朗普还忙上几分。说白了，就是不能做到同心同德。Linux 的群里我见过不少这些人，大家都说他是大佬，其实对于群体来说除了有个响亮的名头外，对团体没什么别的用处，对外也根本起不到什么宣传之类间接用处。人们只是简单的知道，群里这个谁是个大佬，然后呢？指望他们来答疑解惑吗？我想多半会“很忙”。是啊，大佬那么忙，怎么能解决自己如此

简单的问题呢？更多地是嘲讽。所以我的推论就是大佬无用，小白无用，这是个可怕的结果。最终形成的可能更像大学的社团，懂，但不是特别懂。

我曾经在四个“爱好者”相关组织里待过。两个网络，两个现实。所以我现在彻底不抱有幻想。团队的领导者告诉我说要开始论坛，我就去投稿。最终还是人出了问题，一群人天天游戏，怎么指望去为团队做什么贡献，既然不愿牺牲一些，那团队还是解散了最好。制度规范，有哲学理论指导，经费满足，市场需求，那明显是人出了问题。中国的教育制度使得我找不到一个人符合我的需要，是的，偌大的中国竟一个也没有。可见若想让一个民族消亡，教育是最好的方法。我找不到任何有梦想于团队的人，而不是电子竞技。

群里的人越多，就越接近社会平均水平。知乎，豆瓣，B 站是自己拉低了自己的用户氛围和核心价值观。这是为了流量变现。否则看看 Acfun 和 Ofo。这是时代的悲哀，不仅仅是我个人的悲哀。我的悲哀不能让人感觉什么，下一秒读者点击屏幕一侧的 x，然后去做什么就会自然地忘记这一切。而你所忘记的，是别人半生抑郁的痛楚。而有一天，你也会被忘记。但这没什么。

所以当我得知“琴台”就在武汉时，我即前去。妄图看到高山流水般的知音，知音难觅，看到的只是什么“第一届古琴等级评定”，又是圈钱的东西。简直侮辱了此地。还不如让陶渊明的桃花源永远活在学术论文和课本之中。我知道了，I have no best friend.

其实靠人而不是其他的东西，国立清华大学的校长梅贻琦就说过，“大学之大，非大楼之大，乃大师之大。”这就和我刚才冲突了，引用名人名言在科学上不能为证明增添一分信度，我只是将其作为我的引证。我想说群里不是不需要大佬，而是需要真正的大佬，不是睿智

的大佬。否则即使你聚集的人再多也没有用，教化那是教育部的事情，我并不在行。按照管理学说法，组织里存在非正式组织，就是俗话说的“圈子”。这也是一种阶级划分方法。我不是很赞同。但是不可否认其存在。对于饭都吃不上的人，怎么研究什么 FreeBSD，乔布斯和丹尼斯·里奇几乎同周去世，但是我当时看到的只是乔布斯，还有人写了本售价极高的乔布斯传。大师的逝世却无人纪念，这没什么，圈子不同。我只觉得也该有本里奇传，那是他应得的，我们欠他一本传记。如果你现在也不知道他是谁，请点击屏幕一角的“X”。

很多人说 Linux 下没有病毒，有也是双系统 Windows 感染的。但实际上不是如此，只是利益不够大而已。当你发觉系统不正常的时候，它就已经成为了病毒的培养基。我今天预见，未来 Linux 平台的木马病毒绝不会比 Windows 少多少，甚至 macOS 也是。没有绝对安全的操作系统。OpenBSD 都被曝有后门，人是不可靠的，妄图以人力替代机器来提高计算机安全，这是一种可笑的想法。缺乏网络安全意识，也是国内缺德企业大数据分析存在的理论论证。各种数据随便你收集，反正用户不在意。某毒甚至称用户愿意为了方便而牺牲个人隐私。不愧是“毒”。令人痛心。

我与FreeBSD 的故事之五

很久之后，我在 FreeBSD 发帖问 G400 花屏的问题，那时还有三四个人给予了一些回复。但是无用，FreeBSD 官方论坛也是。

那时候应该是 10.3 我记得。10.3 还没有发布。处于测试阶段。我以为是驱动问题，当时能用的只有百毒，所以得不到任何有用的信息，完全是浪费时间。

中文用户的互联网圈子，各种教程都是你抄我的，我抄你的，这没什么，但是一旦有某个 SB 进行改动，就全变了。CSDN，博客园就是这种圈子。比如 Windows 上的 MySQL 数据库 5.7 的 my.ini 文件，data 目录是自己指定的，本身安装路径是没有 data 目录的。一些脑残教程就会告诉你手动新建，但是这样反而会提示你 data 存在，无法初始化。这就很有意思，A 说 data 目录不用管，B 说要新建。如果有人信了 B 的鬼话，而且还不看错误输出，那他恐怕就要浪费更多地时间。当然后话不说，往 Windows 上装这种东西本身就是浪费时间的行为。对于我在 Linux 这些年的体验来说，这样的 SB 教程不在少数。我甚至觉得明天可以写篇文章专门论述怎样识别文章的可信度。

FreeBSD 这里的教程比较少，大多数没什么坑，也可能是基数问题导致的，用户太少了。大部分通过百毒搜到的，都是什么系统安装，还在用 pkg_add 这种十年前的命令。再深入就是什么 fnmp (freebsd, nginx, MySQL, php) 这种东西的配置。

提供 FreeBSD 镜像的云服务器提供商从世界范围内来看也不是很多，阿里云的 FreeBSD 那时候是 10.1，pkg 根本用不了，说是 Bug，要升级到 10.3，然后按照百毒的搜索，看了各种垃圾教程，总算是升了上去。但是提示我 pkg 有个 .so 文件找不到。我通过百毒搜不到任何信息。只能重装系统，什么也不做，直接运行 freebsd-update 命令，这样才可以。现在大概知道了什么 pkg-static 命令可以调整软链接。我也没有试过。我感觉很悲哀，使用百毒简直是浪费我的生命。而其余根本毫无选择的余地可言。无异于慢性自杀。

FreeBSD 无法使 G400 正常工作，我开始安装虚拟机。在虚拟机中第一次认识了 FreeBSD。安装界面和 Debian 差不多，是那种蓝底的老旧风格。FreeBSD 的安装极其简单，全部下一步都没问题。这连 XP 都做不到，XP 写不进去优盘，如果你用软碟通，我现在也不知道怎么纯净的把他写进去，不过也没有研究的必要了，我从不使用 XP 和 win7。

装好之后也没什么可多说的，就是感觉没有 grub 不好切换操作系统，影响将来物理机安装。tty 也就那样，黑底白字。直接 pkg 开始安什么 gnome。速度很慢很慢(如果有人可以联系 FreeBSD 官方，请告之中科大可提供镜像，详见历史推送“镜像站”)，绝不超过 20kib/s，也不能挂着让他下载，会 timeout 的……我只能盯着让他下完 500 多个包。这里我发现一个事情，无论什么设备，什么操作系统，什么软件，只要你盯着他他就下的快，不会断，你离开不看他他就慢，会断开。按理说这是不以人的意志为转移的，但现在有点像薛定谔的猫。不知道大家有没有这种经历？我想和后台前台服务无关，我还控制变量过。

说实话我并没有觉得 Linux 和 FreeBSD 有什么区别。如果装个 bash，那 shell 脚本大都是通的。

我百思不得其解的是没有 free 命令，也没有 ls pci 命令。按理说这种命令不是 bash 内置的，就是缩写变量，或者哪个软件包提供的。ls pci 属于 pciutils 这个包，用这个看设备信息不比 dmesg 方便？这不是自己折磨自己的苦难哲学吗？我是极其反对的。类似的包还有 usbutils(ls usb 命令)。

free 命令更加让我疑惑，因为我个人查不到他属于哪种类型的命令，后贴吧有人告知这个命令属于一个包：procps
(<https://gitlab.com/procps-ng/procps>) 这个包，授权 GPL v2。

free 命令读取 procfs 信息，但是 FreeBSD 早就弃用了这个伪文件系统，因此推论无法使用free 是这个原因。我以后还会关注这个问题。因为 FreeBSD 原生的 vmstat 太难用了。远不如 free 直观。

坚决反对苦难哲学。

我与FreeBSD 的故事之六——我在开源社区这些年

小说的六要素：时间、地点、人物。 起因、经过、结果。其中“人物”是小说的核心。没有人物，一切皆空，动物小说中的动物也算是人物。说起我与FreeBSD，那就不得不谈谈这七八年来我在开源社区见过的种种。所谓开源社区其实是个泛泛的概念，并不单指某一社区、论坛、协会、贴吧亦或者是各种 QQ 群、TG 群、微博群……数不胜数。

之前说过小圈子有小圈子的好，但是坏处也很明显，外人几乎融入不进去，缺乏包容性和开放性。当中让我感觉最难融入的就是一些以 USTC 协会，Gentoo 群为代表的 TG 群。

USTC 协会把加群的链接放到了一个域名的 TXT 记录上，这倒不是很难找，随便找个网页反查一下就出来了。他们的群名和二十四节气是结合起来的，类似“USTC 谷雨”。这使我当时想起了中医。感觉是在卖大力丸呢。进群第一件事我就问“有没有人在用 FreeBSD？”这时候就有几个“大爷”出来了，“直接说什么事”、有个开源输入法的开发者说道，“他不礼貌，大家不要回复他”，“不要理睬这些问题”，搞得好像我在骂人一样，我仅仅单纯问有没有人在用 FreeBSD 就是在骂人，那我要发一句“你是傻逼”岂不是在夸他？我现在就要发一句……“你是傻逼”X3！这个群真是很有意思，我诘问他我怎么就没有礼貌了，我就问问有没有人用 FreeBSD 都不行吗？还有人给我丢出了提问的艺术。我当时就退群了，最近加群我又问了一遍“有没有人用 FreeBSD？”，无人回复，倒不是没人理我，只是单纯死群而

已，没人说话了，是了，除了当初那个说我没礼貌的人，谁敢说话呢，随便就给别人扣帽子，谁才是没有礼貌的人呢？这种群不“死”，那简直是没有天理。

Gentoo 的 TG 群就更有意思了，他们的管理员各自称呼对方为“大佬”，“巨佬”，各聊各的，仿佛与世隔绝在“华山论剑”，颇有一种英雄迟暮的感觉！这个圈子不是你想不想融进去的问题，是你他们根本不会理你，你是谁？海外留学生吗？不是，那你就别说话了。是的，我一直不敢在那个群里说话。

有人和我说圈子不同，不要硬融。群唯一目的就是扯。类似上面几个扯淡的群还有 Linux 中国的几个 QQ 群，我在百毒无果后加了个 Linux 群去问，SSL 证书如何配置。就有一个管理员上来咄咄逼人，问我“你懂什么是 SSL 协议吗，你知道 HTTPS 底层加密原理是怎么实现的吗？”吓得我赶紧退群，原来搞 Nginx 的 SSL 配置还得学习密码学原理。我寻思着我现在也不会他说的那些，但是丝毫不影响我配置 SSL 证书。你以为他很懂，其实他根本就不懂。还有几个群叫什么“水立方”来着……

都说搞技术的 30 以后就没人要了……我似乎加了个中老年技术群？是一个关于开源自制内核的微信群。是的，中老年才用微信，这就是我的偏见。在我看来微信的反人类设计并不比 Unity 这个 Ubuntu 当初的默认桌面少上多少。都说中老年了，你以为他们会在群里和你讨论什么同步异步这种蠢货问题吗？人家关心的是天下事！天天在群里造谣也是司空见惯的一件事了。这个圈子不是你能不能融进去，而是你根本不想融进去。天天东家长，西家短的。真是哀怨啊！

关于 FreeBSD 的群组实在不是太多，你在 QQ 上搜索，看着想加进去的也就那么三四个。但是方向走错了又不回头，指望地球是 O 的在绕回来吗？关于我在 FreeBSD 群的一些事情是下文要说的。

我与FreeBSD 的故事之七——我与委员长那些年

我和委员长是在那个 FreeBSD 的 QQ 群里遇见的：那时天空还有些雾霾，晴朗的日子实在算不上太多。用一句诗圣杜甫的诗歌来说那就是“落花时节又逢君”。

委员长是在加拿大的 University of Waterloo 留学，University of Waterloo 的 Co-operative project 使委员长得以名副其实。当时还不怎么羡慕委员长，现在再羡慕也已然没有丝毫用处了。

委员长是 doc committer，负责文档的修订翻译等工作。所以委员长在 QQ 群里招兵买马翻译 Handbook 手册就是一件理所当然的事情。但是往往事与愿违，委员长搞了一套对翻译者比较复杂的翻译流程，大体来说就是先在 Github 创建一个项目，然后通过 Issue 认领自己翻译部分，最后通过 pull 申请合并翻译后的po 翻译文本。对于一个经验丰富的程序员来说这算不上难事，但是对于一般人来说同步到本地都有点费劲，因为 Github 直接下是 404 的，有的地区甚至 <http://github.com> 都打不开，其中原因大家都知道。

于是我的朋友毛狗子说自己不会 git，直接下载 po 文档，翻译完再发 QQ 给委员长不行吗？委员长当即表示可以，于是毛狗子开始了他的翻译之路……

类似的，我也开始我的翻译之路，我翻译的好像是 Xorg 那一章。但是编辑器成为一个难题，所以我最终翻译得比较艰难，由于给电脑换系统而忘了保存，我丢了我翻译好的文档。我本想继续翻译，但是委

委员长的 Github Issue 上，对我的标记是 Abandon，我就没有继续了……

毛狗子也把他的翻译用 QQ 发给了委员长。

过了很久……大概半年，我都以为他们翻译好了 FreeBSD Handbook，但是委员长把整个项目都 Abandon 遗弃了。毛狗子发 QQ 问委员长，委员长说 github 上没有毛狗子的 pull，是的，委员长和我一样都丢了 po 文件，不同的是，我丢的是我的，委员长丢的是毛狗子的。

多年以后我想重开FreeBSD Handbook 手册翻译工作，但这必须要先从一个编辑器说起……

Po 在 Windows 上只有一个能用的编辑器，叫做 Poedit，这个作者叫做 Vslavik，要不是我没学过英语骂人，我定好好教育一下他什么是什么做人。Poedit 虽然是开源的，但是他的基本功能都是需要付费的，而且还是很贵。我在群里众筹买了高级会员，本来打算用 Visa 信用卡支付，一看有 Alipay……久久无语……我提现了半天不说，还被 WeChat 扣了一笔巨额手续费。

我用 Poedit 预翻译，就是机器翻译，他不会保存机器翻译的结果，于是每次打开都需要预翻译，久而久之我突然就无法预翻译了，我去找 Vslavik，他说我已经预翻译了几千万字了，凭借他的软件赚了很多钱之类的话，我说这是你软件是 bug 不是我的事，我是为开源软件文档翻译的，没有任何收入。但是 Vslavik 非说我违反了用户协议，不给我解，这我就要骂人了，英语骂人不会，然后也就那样没说什么。委员长不去指责 Vslavik 反而过来唏嘘我，令我十分汗颜。

然后我发现我的翻译都 make 编译不了，po 这个文档对格式要求太严格了，一个个改是不可能的只能 rm -rf。

2020年的时候我听说 FreeBSD Handbook 有了新的在线翻译网站
<https://translate-dev.freebsd.org/> 我提交了申请，有人用中文和我说欢迎。我打开后台一看翻译进度 99%，好家伙，谁翻译的？不知道，都 99% 还要我干嘛，我就发邮件给那个欢迎我的人，现在半年过去了也没有收到任何回复，我把客气当真了？是我太实在吗。但是现在 Handbook 仍然如初……

我很久没有见过委员长了，再次借用杜甫先生的几句诗歌吧……何时一尊酒，重与细论文。

第二节 FreeBSD 与猫 ——选择 1% 的生活

很多人觉得 FreeBSD 就是一个操作系统，作为桌面来说又极其糟糕。事实不能否认，FreeBSD 确实如此。FreeBSD 的小恶魔是其标志物，代表守护进程及其 fork。

说实话按照我的审美这个简直丑到一定境界。所以我们的宣传图标都不使用这个，而用更加美观的替代图。

昨天有人问我公众号有什么用，能赚到钱吗，既然不能广泛推广，又赚不到钱，那有什么用？这其实是每个人的需求层次不同所造成的价值观上的差异，我们不必理会。以前的文章段落《务实与务虚》中我已经说的非常明白了，不再赘述。真是让人头痛。那古老的话语，志不同不相为友，道不同不相为谋，一遍遍的催人深省。

FreeBSD 是什么东西？我们都知道 iPhone 用的系统是 iOS，MacBook Pro 用的是 macOS，我们一般的手机是 Android，电脑是 Windows xp/7/8/8.1/10。可能计算机专业的学生还接触过 Linux。FreeBSD 是和他们同级别的一个操作系统。这样解释可能才会有更多人知道 FreeBSD 是什么吧。但是 FreeBSD 很少直接安装在我们的手机上，更多地是嵌入式设备，类似路由器，或者服务器上。

而我们知道硬件需要工作就必须有驱动的支持，驱动是链接软件和硬件的桥梁。比方说玩游戏就必须根据显卡装驱动，更新驱动。而由于版权和厂商支持问题，FreeBSD 只支持很少的硬件设备，或者说支持

的性能非常差劲。

这是一个死循环，和目前所谓的国产操作系统差不多，缺乏生态环境：用户不用 ——> 没必要开发 ——> 没有支持，用户怎么用？ ——> 用别的去 ——> 用户不用

其实这里就能看出一个企业的企业道德和商业方面的预见性，国内企业大多眼界狭隘，缺乏道德，令人痛心。某企鹅曾经支持过几个月的 Linux，后来就人去楼空，空留一个 beta 版本。某娘不说自己更新进度，反而说是用户自己为什么选择了 1% 的生活？而反观国外企业的软件大多全平台支持，也不存在系统歧视问题。

选择是自由的，无可指摘。回过头来很多人疑问了，为什么要用 FreeBSD？老生常谈，爱用不用。

那么和猫有什么关系？

不知道是谁家的猫，不过看上去没有人喂养它，就随便给了点馒头。之后怕它跑了，放在笼子里养。不过吃饱了它也从没有有过要溜出来的想法。

放出来的时候它已经从黑猫变成了花猫，大部分是黄白相间，额头一部分是黑色的。我们从来不给动物起什么名字，因为我们不是很懂，这有什么意义。

这猫大概是我见过最好的一只猫了，因为我现在只记得它，连着狗和兔子什么的我也记不清了。只记得一天半夜醒来出去看到有个刺猬，就直接拿了個塑料桶扣在了土地上，想着第二天再说。第二天想起来，应该往桶上放几块红砖的。自此就想着，为什么没放块砖压着

呢？天遂人愿，又一只刺猬企图穿越过道，我抓起团成团的刺猬，放到了钢板上，又扣了桶，加了三两块砖。这回终究它没有跑，刺猬的鼻子是最可爱的了。然而这个物种似乎不喜欢寄养，就放了它。

猫都是喜欢睡觉的，它睡得很长。有时候还是一只猫好。

它大概已经有好几岁了。它与狗唯一的不同就是它能够逃离院子对它的束缚。它坐在煤火炉子上，毛差点被点着；它趴在电暖器前，胡子都焦了。大概它的鼻子不是被我捏的变了色。

猫一般不咬人，它只用爪子挠人。半驯化的动物大抵如此吧。它很少喵喵的叫，也经常把死老鼠放在床底下。我才知道，这只猫是吃老鼠的。我一直想把猫养的胖一点，抱着好一些，最起码要和兔子一样，无奈它爱晒太阳，又爱飞檐走壁。

它看到我，不是和狗一样要舔我，只是睁开眼睛，然后闭上眼睛继续安眠。

这条街道都拆迁的差不多了，留下了一半废墟和一半卖小吃的摊点。也要拆到这里了。这是租的房子，我只记得要把它关在屋内。白天却找不到它了。自此以后再也没见到过这只猫，也不曾养过猫了。

第三节 Linux 与苦难哲学

在许多狂热的 FreeBSD 粉丝里，他们甚至不允许别人把 FreeBSD 写作 freebsd，要和你强调，F 和 BSD 都是大写的。还说这是什么尊重之类的东西。大抵和孔乙己的茴香豆的茴的有四种写法一样吧：

“FreeBSD 拼写有四样写法，你知道么？”，“不能写罢？……我教给你，记

着！‘FreeBSD’、‘freeBSD’、‘Freebsd’、‘freebsd’这些字应该记着。以后做 FreeBSD 管理员的时候，写文档要用。”

我觉得这是一种病，用自己的要求规范别人，起码和孔乙己是一样的。

狂热粉一到店，所有喝酒的人便都看着他笑，有的叫道，“狂热粉，你脸上又添上新伤疤了！”他不回答，对柜里说，“温两碗酒，要一碟茴香豆。”便排出九文大钱。他们又故意的高声嚷道，“你一定又把 FreeBSD 这个单词小写了！”狂热粉睁大眼睛说，“你怎么这样凭空污人清白……”“什么清白？我前天亲眼见你在群里用小写拼出了 freebsd，被群主和管理员吊着十循。”狂热粉便涨红了脸，额上的青筋条条绽出，争辩道，“小写不能算错……特殊的表达方式！……读书人的事，能算错么？”接连便是难懂的话，什么“人非圣贤，孰能无错”，什么“特殊语法”之类，引得众人都哄笑起来：店内外充满了快活的空气。

讨论了这些可笑的大小写问题，再来说说以前经常说的苦难哲学。

能用 Windows 在几秒钟内完成的工作，非要在 Linux 上瞎折腾，QQ 是腾讯公司软件，嘴上说着 GNU 精神，手上开始装 virtual box 或者 wine crossover。这算哪门子 free？这就是苦难哲学，你知道离不开 QQ，那就无法选择 Linux。当然每个人选择百分之几的生活都是自由的，只是，不累吗？这不是自由，是束缚。都到 Linux 了，还是依赖于 QQ。这是苦难哲学彻头彻尾的体现。具体表现就是自己折磨自己，重复造轮子，不尊重现实。那些说着不要管是什么，先写个项目的人，都是这种人，造成了更大的悲哀。是不尊重软件工程的表现，完全无视用户与市场需求。写出来的东西别人怎么用，怎么看？吹嘘什么命令行比图形化好，说什么开发周期，加几个框框就能影响开发周期了？无不是受苦难哲学的影响，仿佛多经过一些步骤在类 UNIX 上实现了和 Windows 类似的功能是多么牛逼的一件事。有人连 fcitx 和 rime 都分不清就和我说 ibus 好，真是有趣的苦难哲学呢。

如何定义苦难哲学呢？

造轮子（无意义重复前人工作），忽视已有软件开发原理；

明明能用 A 完成非要用什么意义价值观这种玄而又玄的狗屁原则问题这种东西来捆绑自己用 B 完成；

鄙视一切用鼠标的软件，去背什么 vi 键盘图；

那么回过头来，用 WP 手机的用户是否也是苦难哲学的受害者？使用 Linux 桌面，FreeBSD 桌面的用户是否也是苦难哲学的践行者？

在某种意义上来说，确认如此。一些人非要在 Linux 下学习 C 语言，说什么更清楚的了解 IDE 的工作方法，说什么预处理编译汇编链接在 IDE 下不直观，非要折腾自己用什么 GCC。GCC 好用吗？对于初

学者来说有什么用？那些是编译原理课程所需要的，而不是 C 语言。作为一门编程语言课程，绝不能无限的扩展其课程内容，况且目的不同，什么指针这种东西在高级语言里是几乎看不到的，吹嘘什么更理解指针与数组的实现原理是苦难哲学的表现，无异于屠龙术。我还是那句话，编译原理和算法导论没有用，大部分企业公司用不到，要对自己有一个定位，是搬砖的就不需要学习量子力学。除非仅仅出于爱好或者学术目的。

用户需求决定了软件的开发方向，而缺乏商业支持的开源产品往往无视之。用 GPL 协议捆绑用户也是一种苦难哲学。按照这个观点，用 Windows 就不是苦难哲学了？不然，能用 Linux 很快完成的工作，为什么非要用 Windows？到底这种东西是不是工具，我认为人非工具，而这种该是工具的则一定是工具。

我尊重那些将其珍视为亲人朋友而非工具的用户，但是请务必意识到，人是具有社会属性的，请多多关心他人，无论是谁。

我不止一次的看到这句话“多谈技术，勿水”，有时候则是其变种“你写过什么项目吗？”，“这里用什么可以实现”，“试编程，完成……”这种句子。有时候甚至是英文，是中英混杂的。我觉得这破坏了语言的纯洁性。

因此很多计算机行业或者爱好者的圈子都有专门的“水群”，还有个组织的名字更加有趣，叫做“水立方”来“水”。他们认为的“水”不完全是与计算机或者其主题无关的东西，有时候只是随便发发以显示自己的权威。

“如果你不按照我的方式去做，我们就没有交流的前提存在。”这就是他们的观点。“闻道有先后，术业有专攻”，韩愈的《师说》就阐述了这个道理，我们和他的不同之处无非就是时间罢了。这些和天赋什么东西完全无关。这并不比造火箭那样复杂。所以我不认为这种规则，双方平等对话的规则是建立在遵守一方先有规定的基础上。现在计算机界中引入了这样一个政治词汇“政治正确”就是为了反对这种固有规则。

看起来有些荒谬，技术和这些有什么关系？关我屁事？是的，和我以前说的那样，总是认为这些东西是无用的，技术强就是最厉害的。这是缺乏人文关怀的体现。这些无用之物不是因为其真正无用，而是太有用以至于不知道怎么用。从而认为其是玄学。联系是普遍的。

Linus 说自己没变，是 Linux 社区脏了。所谓“强者制定规则，弱者只能遵守”，但是强者一开始就最强？强者的存在意义是什么？我认为强者有更大的责任去发展创新。否则便与社会达尔文主义无二——“优胜劣汰，适者生存”。这是极其不正确的，更是不尊重科学精神的行为。自然科学的结论怎么能够不经过验证，证伪就按到社会科学领域头上？倘若都是“竞争”，那为何要构建什么“和谐社会”，诺贝尔和平奖为什么要颁给“为促进民族国家团结友好、取消或裁减军备以及为和平会议的组织和宣传尽到最大努力或做出最大贡献的人”？直接打不就好了，你自己说的，谁强谁上。计算机行业也很简单，一样照这个套路来，谁技术高谁钱最多，谁说了算，谁在社会上受尊敬。显然不是，有谁记得和乔布斯同月去世的丹尼斯·里奇呢？每个人的选择都是自由的，你完全可以退出那些你不想也没有能力改变其规定的组织。就好比 Funtoo Linux 之于 Gentoo Linux 社区。

我极力避免使我的组织成为这种东西。我给人们更多选择，也坚持原则。

回过头来，”水”是什么意思。我们都知道这个道理：“一个团体或组织里的人越多，其水平就越接近社会平均水平。”好比知乎豆瓣乃至 B 站都遇到过商业化的困局。现在的用户几乎根本给不了他们多大的商业价值，而扩招则会降低其专业水准和用户氛围。那既然你允许别人加入，又要不影响这种东西，如何做到？除非对方水平更高，如此其加入能得到什么有意义的东西？所以我们更喜欢新人。而非固执己见的老程序员，甚至还要你叫他“叔叔”。这明显不是一个圈子的人。你和他谈生活，他和你谈年龄，你和他谈技术，他和你谈经验，你和他谈项目，他和你谈工资。我们必须有这种觉悟，谈技术要有拿得出的东西，但不必奇货可居。

有人把这些都扯开了，认为完全靠“人”。此言差矣，封建制度之所以维持这么久，就在于这么一套制度或者几套制度。①中央集权 ②专营制度 ③科举制度 ④思想专制。这是完全靠人的吗？西方其法治才是源远流长。没有任何人能够脱离生活谈技术。现在几乎没有那种所谓“追求真理”的人了。也没有颜回那种有所得乐而忘忧的人了。不还是为了生存？我不懂如今大谈区块链，大数据，人工智能的人和古代那种大谈孟子治国，孔子育人的老儒有何区别。怕也是一个孔乙己。“FreeBSD”中的“F”和”BSD”都要大写，不然就有四种写法。你应该学会罢，将来做运维的时候要用到。

有些入门者也很搞笑，也谈“勿水”，自己进来本来就是“水”的过程。人越多，专业组织水平越低。我力图改变这种局面。还是老话，同心同德，大家想的不一样，就没必要浪费时间了。有人还为新手写

了本书，叫“提问的艺术”，或者直接说 read fuck。诚然我承认不是每个人都和孔子一样有教无类，是个教育家。但是我还是愿意给出更多帮助，因为，曾经每个人都如此。

最后奉劝读者，勿用百毒，善用 Google、Duckduckgo 等。这也是你能看懂本文的前提所在，否则你不可能明确地理解我的意思。我承认读者自己的阅读理解是对作品的再次创作，但此处请务必不要。

什么是苦难哲学，这个词不是我生造出来的，而是切实存在的，在学界这么多年我看到很多充斥着苦难哲学的地方以及构成这些地方的成员和他们所开发的软件。

软件工程，是将软件开发规范化，流程化以提高软件开发效率的一种工业方法。

软件工程第一步就是用户需求分析，大多数苦难的来源就是因为无视这一步，总是认为“写”代码是最重要的，而“算法”更是其吹捧的重中之重，所以写文档，用户需求分析什么的，和自己没多大关系。自己只管写代码就好，有没有人用？造轮子？那都不是我所需要考虑的问题，无视科学，不尊重科学是苦难哲学产生的重要根源。

有人认为自己是做内核开发的就很牛逼，但其实不过尔尔。

连 Linux 内核不是完全开源（见 GNU Linux-libre 项目，一个完全开源的 linux 内核）这一事实都不知道，还认为内核完全都是开源的，还能说出“不开源那我是怎么编译”这种胡话，真是令人忍俊不禁。这些骄傲的人儿不在少数，我在 USTC 和 TUNA 都见过很多，或者说都是这种人。对此我不予置评。都是被苦难哲学毒害无法自拔的人们啊。

开源软件的质量通常很差，存在各种问题，甚至是相当易于发现的问题。就拿图形化界面的 ZenmapGUI 来说，至少存在两个重要 bug。一是经常性地在输入框无法输入任何文字，且与输入法键盘无关（问题在多台计算机均复现）；二是当切换扫描窗口时，扫描输入的日志会被清空，找不到日志。所以我很好奇这些人究竟自己用没用过自己开发出来的软件，因为凡是用过的，都有这种问题。这难道说不上是一种苦难吗？（已经报告 bug 但未回应）原因在于人力和资金问题，开发者水平参差不齐，能力有限，二是本身有本职工作，无法分心顾忌太多。

这种简单的 bug 随处可见，比如安装 Debian 时，如果你创建了普通用户，那么你使用 su 命令或者 sudo su 命令都不会成功，因为其并未改动 sudo 的配置文件（应加入 ALL=(ALL) 一行）。虽然只是一行只差，但足够困扰新手了，反观 windows 绝不会有这种问题，蓝屏了重启你多半是能够开机的，但 kernel panic 怎么办呢？我想重启是无用功。是技术问题吗？是其根本不关心这些问题。

所以产生了 FreeBSD Handbook 文档要不要翻译，有什么意义的问题，当然缺乏人手各种推诿是主要原因。会的不用翻译，不会的翻译了也没用，还是不会。

计算机哲学目前没有一个完全大一统的理论体系。计算主义是一大主流，源自毕达哥拉斯学派，其认为数是万物本原，数是和谐统一美的东西，构成了世界。

黑客帝国就蕴含了计算主义：如何证明我们真实存在，而不是计算机模拟的数字信号？

苦难哲学看上去只是一种行为方式，如在 Windows 平台安装 VIM，并声称其比记事本强一万倍。但不然，苦难哲学只是通过这些行为表现出来，其根源还需要进一步批判思索。

许多开源小将不懂何为开源却大谈开源。

“你写小说为什么不用 LaTeX，为什么用 Word？”

“你凭什么说 LibreOffice 不兼容 Word，你怎么不说 MS Word 不兼容 LibreOffice？”

“自由与软件基金会的 ‘Free’ 的意思就是免费，不要钱。”

“为什么你不用 GIMP，去用 PS。”

以上种种就是我看到的一系列笑话。他自己说的那些东西有可用性吗？

小学生说 libreOffice 好用，于是我去安装了一个，先不提反人类的界面设计，就说他 10 分钟崩溃了 3 次，连崩溃前的编辑记录都没有，“这玩意也是人能用的？”我不禁感叹道。他甚至没有小一小二四号字，只有 12345678 这种东西。这个东西首行缩进的设置都够呛，都是厘米以单位的，而且首行缩进不随着段落文本的调整而调整。这无疑贯彻了开源哲学的反人类。

小学生说 LaTeX 好用，我看了半天也没发现对于一个文字工作者来说这东西有什么好用的。

小学生说 GIMP 比 PS 好用则更是无中生有。

小学生说 Free 是免费就更加是谬论，Free 是自由而非免费，也就是说收费开源是被允许的，微软的开源模式就是这样的。微软事实上并不是闭源的，它是有条件的开源的，条件是购买一定数量的 Windows 副本，而且有正当理由。还有人说 Windows 是闭源的，说明这个人不仅不懂 Windows，更不懂什么是开源。自己不信可以看看

<https://www.microsoft.com/en-us/sharedsource/enterprise-source-licensing-program.aspx>

说他是小学生都是侮辱了小学生，这就是一个脑瘫罢了。这就是苦难哲学简称闲的没事干，在那造轮子。

这种人比那些巨盲更加令人感到恶心，他不仅没有技术，还在那里半瓶子水晃荡。

他举例 Windows 似乎是一无是处，事实上是这样吗？

当然不是。

举例 NTFS 文件系统，这个文件系统是 Linux 一辈子都做不到的。这个文件系统的稳定性远远超过所谓的什么 xfs ext，更不要 btrfs 了，那就是一个笑话。至于 zfs 这种文件系统更是好笑，设计上只能扩大文件分区，无法缩小。

NTFS 无论你怎么意外断电他都能顺利开机，反观 Ext 这种东西根本没有可靠性可言。

最令他们忽视的是 Windows 的兼容性。

2021 年代的 Windows 11，甚至还可以运行 26 年前 windows 95 程序，而且更甚至地是不需要重新编译源码。

这是任何一个 Linux 系统都做不到的，他们甚至连几天前的程序都不能够兼容。

他们的程序会依赖特定的 C 库，依赖特定的内核版本。而这些都是改不了的，除非你的 Gentoo。就算是 Gentoo 也不是能够任意选择的。Linux 程序也有兼容性？这就是一个笑话罢了。

还有人说 Linux 软件不需要兼容性，哈哈哈哈，看看这些开源小将多么荒谬吧！

不要提及 Macos，那是苹果战略性的不兼容，而且他想兼容就可以兼容，老的软件有老的行为，新软件有新行为。

Windows 的图形界面的稳定性是 Linux 一百年也达不到的。Xorg 和他所谓的替代品 wayland 就是两座屎山。

开源软件根本上就是违反软件工程的，因为其第一步就没有进行用户需求设计，他们才懒得管用户到底看不看得懂 LibreOffice 那一串二十几个没有介绍的图标是什么作用的，他们才懒得管你到底会不会编译程序。

这些小学生连以上这几点常识都没有就出来半瓶子晃荡，叮铃咣当乱响。

第四节 从一个想法看 FreeBSD 是商业化还是学院派

在某知名计算机网络论坛上我看到一个帖子，说自己想根据 FreeBSD 做一个移动的终端操作系统，就像安卓，苹果的 IOS 一样的。

逆向思维当初开发安卓的时候不可能没有考虑过 FreeBSD，因为无论从代码质量还是 BSD 协议来看，FreeBSD 都优于 Linux，这是一部分人的看法；还有人认为 FreeBSD 做出来最多和安卓一样，首先驱动问题就解决不了；还有人从技术方面入手，称 xorg 阻碍了其发展，造轮子非常困难，安全方面也值得考虑，虽然去掉了虚拟机，性能会有所提高，但会 FreeBSD 的开发者少之又少；更多地人劝其脚踏实地，这个根本赚不到什么钱。

但是根据我的理解 FreeBSD 主要是因为缺乏大的商业公司对其进行服务支持。因为我们都知道开源产品一般是靠出售服务而不是软件本身来盈利。根据 GPL 协议，有源代码提供，重新编译一下就可以了，就如同 RHEL 与 CentOS, Scientific Linux 的关系。所以久而久之变成了恶性循环。但是 FreeBSD 用户群中存在着相当大的一部分人认为是 BSD 协议阻止了其发展，因为根据二则协议，修改过的产品可以不开源，收费。但是这些想法也是不正确的，具体看苹果与 FreeBSD，以及 handbook 中众多使用 FreeBSD 服务的商业巨头，就知道回溯源代码更加能够节约企业的经济成本，是相互促进的关系。

FreeBSD 对 ARM 的支持很差劲，现在移动终端不就 arm 指令集吗？难到还能是 MIPS ? 说到这里我还是真的很希望 FreeBSD 能够支持龙芯处理器的。因为毕竟 MIPS 目前应用的不如 arm 广泛，机遇更多一些。FreeBSD 的嵌入式开发绝不是阉割内核，加几个打电话，发短信的软件就可以的，难度接近从 0 开始。

其实不只是 FreeBSD 的嵌入式开发，Linux 的进程也大抵相当，Ubuntu Phone 、FireFox OS 、Windows Phone 无不从其 PC 端迁移到嵌入式设备的失败产品。

其实更多的人们在强调风险，规避风险。有人在我国一所著名大学毕业典礼上发言称，要相信社会上存在公平与正义，存在真正的学术。可能真的有人信了，但给他的终将是多年以后聚在一起碰杯的声音。这片土地真正缺乏的是她所说的那张种子，从来都没有。同样的，独行而无友，是一种最深层次上的孤独。

我们真心祝福那些有梦想并愿意去做的人，也同情那些受于现实桎梏无法前行的普通学生，但更祈求你，每个个体，更看重自己的价值，开心就好。

至于 FreeBSD，我始终是当做工具来看待，工具属性是其基本属性，也许以前是玩具，是别的什么，但是我更加看重人的价值。看到做嵌入式，就在等树莓派 4，仅此而已，博通的处理器和无线设备就非常烦人，根本不开源，还要做这种开源设备，导致驱动问题的发生。

第五节 Linux 社区已经成为了一个肮脏的泥潭

标题这句话是 Linux 内核首席维护者、Linux 之父 Linus Torvalds 所说的。原话是 “The Linux community is now a dirty quagmire”。

这句话不仅仅体现在内核开发中，而是体现在 Linux 的各个阶段，包括不限于国内的各种 Linux 社区社团组织。本文主要讨论的就是后者。至于 Linux 内核、systemd、Code of Conduct 那些乱七八糟的争议问题，更是泥潭一滩。

不得不承认，国内大多数 Linux 社团，有一个算一个，几乎都被所谓的巨兽所支配。看着花里胡哨的，漂亮，居然把自己的 Linux 社区打造成了一个只有巨兽们说话，没有一点实际作用的，一个虚幻的组织。你看你的群里有新人敢说话吗？

什么是巨兽，我不做解释，懂的都懂。

我们都希望的 Linux 社区是这样的：“我们都知道新人的确很菜，也喜欢抱怨，并且带有浓厚的 Windows 习惯，但既然在这里询问，我们就应该有责任帮助他们解决问题，而不是直接泼冷水、简单的否定或发表对解决问题没有任何帮助的帖子。乐于分享，以人为本，这正是 Ubuntu 的精神所在。”

巨苣们是不是忘记这几句话了？巨苣们把所有人，所有问题都当做他们口中的“伸手党”，我不否认，一些问题是愚蠢的，比如 apt 怎么卸载软件。因为此类问题往往通过搜索引擎就能解决。

但事实上呢？

“请问各位大佬，HTTPS 证书应该怎么配置呢，我查过了搜索引擎都是不正确的。”

“你懂 HTTP 原理吗？”

——以上是在 Linux.cn 的官方 QQ 群组亲自体会的，并且还是个管理员回复的。

“请问有人用过 FreeBSD 吗？”

“你想问什么，有屁就放”，“大家不要搭理他他不礼貌”

“我就是问问又没有人用，怎么还和礼貌扯上问题了？”

——以上发生在 USTC 的 TG 群组里。

“巨苣，你的论文写完了吗”

“我还没写完，巨佬”

“巨苣好厉害啊！”

——以上发生在 USTC QQ 群组里。

“不会翻墙的人都是畜生，他们连猪狗都不如。他们只会吃屎”

——以上发生在 Gentoo 中文社区 QQ 群群公告。

“别问了，你看有人搭理你吗？”

——以上发生在 RockyLinux 中文群组 QQ 群里。

ipxe isn't a priority at the moment. We need many more mirrors around the world in more countries for that to be useful. Considering most people don't network boot from their personal notebooks/desktops, I call bullshit. Network booting is typical in data centers and large enterprises. omg bro you're trying so hard to put your malware inside artix. You can't audit it. You and your images are not part of our infrastructure. Which means we have no control over it and therefore cannot stand behind it. As long as you don't share it here, you're are fine. It is unofficial and potentially very dangerous. i see, i think they force me to talk about bedrock. Again, your response has nothing to do with the conversation... your translator is pure shit. What is the point having those in the repo?, one cannot use them.

——以上发生在 telegram @artixlinux 群组。

有人说孤证不立，那以上 6 个例子够不够呢？并且可以看出，和社区成立时间以及人数都是没有关系的，RockyLinux 群组成立不到一年，人数不过百。

那么我要问一问，到底谁才是真正的畜生呢？显而易见，这些巨兽都不是什么好东西。有问题他不在，一开始吹牛逼他就出来了。大家你好我好，互吹大佬，互捧巨兽。

作为一个管理员在群里的唯一作用就是打击和吹嘘。这就是他们唯一的作用。自己会这些显得自己牛逼的很啊！你们都不会的干活！就我会，你这个小垃圾！

那试问你建群的目的是什么？为了方便你们各位大佬互称“巨佬”？那些真正牛的大人物会在一个群里刷自己的存在感？你不帮助别人可以闭嘴，给好人留个位置就可以了。别浪费了好人的位置。

正是这些真正的垃圾，脏了 Linux 社区。污秽了 UNIX 哲学。

只知道吹嘘，对社区毫无贡献可言，笔者好歹还翻译了 gentoo wiki，这些互称巨佬的巨兽做了什么呢？打击新人，外加吹嘘自己的“傲骨”？

这些所谓的巨佬颇有多年丑媳熬成婆的模样，当自己当了婆婆就开始疯狂压榨儿媳。真是好笑呢。

按照沈腾的小品来说就是“你不帮忙就给好人让个位置，你也不是多么牛逼，地球缺了你就转不动了”。

他就一点点小小的权力，偏要把他发挥到极限。正是这些“畜生”长期把持 Linux 境内组织的上层，才导致了今天的式微。他们几乎把全国的 Linux 组织瓜分干净，以至于新人进去好像进了狼窝一样。

同时我要劝告那些还想加入某某 Linux 组织的新人，这些巨兽一般都是垄断了最优秀的教育资源，编写了一堆造轮子的没有用的程序，群主为了笼络这些“畜生”，授予其管理，实际上群里根本没有别人说话的位置，是“畜生”与发起人共治 Linux 社群。

他们无恶不作，不仅仅是毫无贡献，更是迫害新人，杀人，司空见惯！用他们那把隐形的锋利地刀割裂了 Linux 社群与新人。

你想反抗？不行！你要么被逼叫他巨佬巨苣，被其侮辱，要么自己自力更生，自己学习。他们的特权就到这个地步，就嚣张跋扈到这个地步。

大量的巨苣占领了 Linux 社群，社区。这整个 Linux 开源界都已经被他们污染了。没救了。学问很大，但是无恶不作，对社区毫无贡献。

这些所谓自称的 Linux 国内的社区的管理员们，又何尝不是？他们看不起没有梯子，不会翻墙的人，也看不起那些使用 Ubuntu 的人，形成了一整套完整的鄙视链。

我至今也没有弄懂这些巨苣的苦难哲学，他们喜欢造轮子忽悠那些新人，不去装 KDE，Gnome 这类完整的桌面环境，而去搞什么 fvwm 这些窗口管理器。硬生生地自己把使用难度提高，对其而言，使用体验真的有所提升吗？我看不见得，苦难哲学而已。

巨苣的代表行为就是他们到处忽悠别人去安装 ArchLinux。我也至今不明白，一个稳定性还不如 Ubuntu 的系统有什么可吹嘘的呢？但是这些所谓巨苣们就喜欢这些臭名昭著的东西。

似乎他们是掌握了天堂钥匙的人，掌握了整个世界，所以他们控制了整个 Linux 群组也不新鲜。你不跪在他面前口呼巨佬，那就是你的错！

那么我们想强调什么呢？一味地强调与 Linux 世界对立吗？这些人只在 Linux 社区出现吗？正好相反。也即其实不然。大家可以对比下远景论坛和国外的黑苹果论坛。

事实上，这些巨苣无处不在，只是 Linux 社区的现状比较严重而已。

对于笔者自己而已，正是这些在 FreeBSD 群组里打着管理员名头的巨
酋才使得笔者自己来创建一个社区，一个自己主导的社区。

一个人，天天在某个 FreeBSD 群里发内核分析的片段话语，他也不是
为了分享自己的教程惠及他人，也发的不完整：似乎就像是一个大学
生在小学生的 QQ 群里发了一道微分几何的题目。他并不是为了教会
你高等数学，而是为了显示自己高贵的身份罢了。

巨酋无处不在并且已经渗透了整个计算机世界，我们应该做些什么，
又能够做些什么呢？

第六节 Linux 败局已定——驳 FreeBSD 大败局

看过我上一篇文章《Linux 社区已经成为了一个肮脏的泥潭》以及上一篇文章《Linux 与苦难哲学》的读者都知道，Linux 从开发者到社区乃至用户都是充斥着难以言表的一种骄傲。仿佛自己是真正的 UNIX 后裔，是开源界的唯一真理，谁拥有 Linux，谁会 Linux，谁就掌握话语权。

如此过了二十余年，物是人非，当初的 Linux 走到了今天这个地步，无论从何种意义上来说，都是一种错位。

王垠过去写了一篇 Linux 劝进文——完全用 Linux 工作，后又发了一篇 Linux 劝退文——谈 Linux，Windows 和 Mac。人的认识是一个螺旋式上升的过程，当初的错位必将在日后得到修正。Linux 也如是。

不断的有 Arch Linux 用户到处宣传 Arch Linux 的优美。是啊，如此优美的一个系统！Nano，Vi 都依赖 Glibc，不更新就意味着你用不了这些再基础不过的基础软件，强制性地要求你更新，他们才不在乎更新会不会导致你的系统挂掉——反正我又不能给你蓝屏。我不知道这些 Arch Linux 劝进文的作者多年后会不会像王垠一样，能够坦诚的承认自己的错位。我也不知道这些 Linux 劝进文的作者多年后会不会像王垠一样，能够坦诚的承认自己的错位。

画龙画虎难画骨，譬如皇帝的新衣。以模仿开始，无论怎么看，都注定了 Linux 败局已定。

01 成也 GPL , 败也 GPL

GPL 许可证给人们带来最多的感受绝不是“free”，而是“传染”、“感染”，似乎 GPL 是一场瘟疫，传到哪里都会带来灾难。而且似乎对其误读的人不在少数，GPL 之“free”，不是免费，而是更接近“libre”。对于 GPL 来说也并不意味着你的代码会被 100% 开源下去，得到回馈。著名的例子就是 Boox——一家使用了 GPL 代码却没有开源的 Eink 生态链公司。

FreeBSD之所以要与 GPL 划清界限是因为 GPL 阻碍了FreeBSD的发展目标——One of the ongoing goals for the FreeBSD base system is a migration to modern, copyfree or at least more permissively licensed components. FreeBSD 基础系统的一个持续目标是迁移到现代的、宽松授权条款或至少是更多许可的组件。

如果说 BSDL 会被大型公司滥用，那么 Anti-996L 才是最佳许可证，而绝不是什么 GPL 。因为任何许可证都会被滥用（除了 Anti-996L ）。Anti-996L 才是世界上最好，最优秀的许可证。保证了你的代码不会被任何公司滥用——因为没有公司敢用。

Apple公司复用了大量的 FreeBSD 代码，同时也为其提供了资金以开发 LLVM，类似的，还有英特尔和 Netflix。

被“利用”不一定是一件坏事。毕竟只有没有任何价值的东西才不会被利用，只有没有任何用处的“hello world”代码才不会被滥用。也就成了庄子所说的“无用之用，是为大用”。

尽管我们看到了许多开发者由于项目被滥用而删库跑路，但是这件事也是双向的。开源代码开发者往他们的代码里塞圣诞节彩蛋，商业公司及“利用”他们的人就要承担这一代价——无论是被开除还是被质问到你这个按钮到底被谁吃掉了。

得益于 GPL，一个 CentOS 倒下去，千千万万个 XXOS 站起来，肉眼可见，从此以后的教程将更加地复杂，更加地不具备普适性。此前写文章抨击 CentOS 的 Si Feng 说过“CentOS：永远有多远就离它多远”——如果一定要为服务器挑选 Linux 发行版（假设不考虑其他非 Linux 系统例如 FreeBSD）的话，首要的原则就是尽可能远离 CentOS。（<https://feng.si/posts/2019/07/centos-the-last-Linux-distro-you-should-ever-consider/>）当时还有很多人抨击他说他荒谬，现在看来谁荒谬还真是不一定。现在看看那篇文章下的回复，不知道斯言仍在，斯人何在？我现在也仿照着他，说一句话——“Linux：永远有多远就离它多远”

02 “Linux 真的是个好东西吗”

Linux 真的赶上了一个好时候，其实 Linux 最初和 GNU 是没有半毛钱关系的，GNU 计划一直排斥 Linux，他们想要有自己的操作系统或者说内核，他们管他叫做——GNU Hurd。这一项目至今仍在进行中……

Linux 发行版不过是一堆 GNU 工具+包管理器+ Linux 内核所构成的松散的操作系统（一般将前两者合称为“Userland”）。每个人各维护各自的，就像 Ubuntu 很少回馈上游社区一样，这也反证了 GPL 许可证的确不如 Anti-996L。

GNU Hurd 错位，Linux 占据了所有开发者的计算机，就像破窗效应那样，如果那些窗没修理好，可能将会有破坏者破坏更多的窗户。越来越多的开发者走向了 Linux 而非 GNU Hurd。可以说是“所有的好东西都给了 Linux”，那么反过来说集开发之大成的 Linux 是个好东西吗？看上去答案是肯定的，其实不然。

硬件上看，Linux 的 GPL 阻碍了硬件兼容性扩大（他们是怎么解决这个问题的呢？），仍然有许多用户认为 Linux 是完全开源的，他们认为——不开源我是怎么编译的呢？那么“Linux -libre”项目（<https://www.fsfla.org/ikiwiki/selibre/>）所删减的是什么呢？—— Linux , the kernel developed and distributed by Linus Torvalds et al, contains non-Free Software, i.e., software that does not respect your essential freedoms, and it induces you to install additional non-Free

Software that it doesn't contain. Even after allegedly moving all firmware to a separate project as of release 4.14, Linux so-called "sources" published by Mr Torvalds still contain non-Free firmware disguised as source code. Linux , 由 Linus Torvalds 等人开发和发布的内核，包含非自由软件，即不尊重你的基本自由的软件，它诱使你安装它不包含的，额外的非自由软件。即使据称从 4.14 版开始将所有固件转移到一个单独的项目， Linus 先生发布的 Linux 所谓的“源代码”仍然包含伪装成源代码的非自由固件。

软件上， Linux 的软件包数量的确很多，甚至比 Windows 的软件还要多，这在一定程度上有积极影响，但是其消极影响更甚。有很多的开发者并没有很好的维护他们的项目，因为他们的 GPL 并不起到任何保证作用，开发者也没有任何义务为软件提供任何保证。可见到的是，先有软件，后有用户需求设计（好吧，实际上压根没有这个东西），他们不在意用户是怎么想的，你想往一个IM软件里加一个截图功能，因为类似的 Windows IM 软件（比如 QQ，微信）有这个功能，而且这个功能很重要。但是开发者会和你说，你用其他的软件截图然后复制进来就可以了；或者说我们是 GPL ，你看得到源码，你自己去改就是了，你不会改那就是你自己的问题了，和我们无关。你可以随便报告 Bug，但是我们可以简单地说“No”。

安全上，OpenBSD 为了改进其安全性移除了 Linux 兼容层以及其常用的软件“sudo”。Linus 此前就抨击 OpenBSD 团队是一群自慰的猴子（OpenBSD crowd is a bunch of masturbating monkeys），认为他们是玩具。至今人们仍不知道那些年 SELinux 和 FBI 的关系（SELinux （Security-Enhanced Linux ）是由美国国家安全局（NSA）所开发）。嗯，没错，这也是一个 GPL 软件，“你觉得有问

题你可以自己去看源代码吗！并给我们指出来。”但是更多人不知道的是 Windows 的开源是没有意义的，几十 TB 的源代码，无论他是开源还是不开源对于何种意义来说都是一样的——你既看不懂，也看不完，等于没开源。

03 Linux 的商业布局

Linux 的用户或者说在有商业版发行下的，使用所谓“社区版、开源版”的 Linux 用户其实和他们所嘲讽的使用 Insider 的“小白鼠”没有什么本质上的区别。所谓 Linux 的商业布局就是让更多地用户接触到苦难哲学，接触到一个根本不完善不稳定的软件。

Wine 的商业版叫做 Crossover，境内由著名的“苏州思杰马克丁”所代理。Wine 是一款在类 UNIX 操作系统上模拟 Windows 程序的开源软件，使用起来可谓是苦难哲学到了顶峰。Crossover 则不然，他们以开源为测试模板积累了丰富的经验以改进自己的程序。类似的著名操作系统就是 Fedora——RHEL 的社区版，开源版。

这些 GPL 的软件随时都有闭源的风险，但是他们又说了，GPL 允许 Fork，于是出现了很多李逵的副本——李鬼。比如 Mysql 与 MariaDB、Rocky Linux 之于 CentOS、OpenJDK 与 OracleJDK……那么到底该用哪个呢？你不知道，因为 GPL 软件防止你滥用他。商业公司真的愿意和开源软件共荣共生吗？答案是否定的，他们只是没有更好的选择而已，君不见使用 LLVM 后的苹果公司怎么看待 GCC。认清这个现实的商业公司正越来越多。

Linux 的商业布局就是让用户免费帮自己测试软件，然后赚钱，“回馈”社区继续让用户帮自己免费测试软件，最后用户还会非常感谢这些公司为他们的开源事业做出的贡献。而用户得到的则是一个充斥着苦难哲学的软件，除此以外，别无长物。按照他们的说法，这也叫做“利用”。

04 “大独裁者”未必是好事

众所周知，Linux 的内核开发是由 Linus 一人独裁负责的。我们仍未知道何时独裁也成了一件值得称赞的事情了。或者在他们看来只要项目进展顺利，Linus 选好接班人，Linux 内核就可以像中国的秦始皇所畅想的那样——受命于天，既寿永昌。他们把不同意见的人都打成“左派”，给他们扣上帽子，没有人告诉你，那个年代已经过去了吗？

Linux 的社区和他们的开发风格也是一样的，充满了各种巨匠，巨匠们一言以定天下，其他人说了不算。最典型的就是贴吧，如果你想问 Linux 的开发模式是什么，他们的社区是什么——请看百度贴吧：有五六个自诩代表吧友实则毫无贡献的人，在奔走试图合纵连横试图推翻现有吧主；有一个自称贴吧是自己的私有物品，吧主万世万代都得是自己的小丑；还有一个固定的吧宠，但从来都认不清自己的身份，还在说别人哗众取宠；剩下的都是摸鱼躺平的围观群众。

Linux 社区如 Linus 所言，是一个肮脏的泥潭，任何想要和这些人辩论并试图驳倒他们的人都将会被其拉低智商到同一水平，并眼睁睁地看着他用丰富的经验打败你。

Linux 本人所骂走的开发者绝不在少数，他们的行为准则（CoC）制定了和没有制定是一样的，你是能开除 Linux 不让他贡献一行代码，还是你能让他只做事不说话？类似的 GNU 离开了 Richard Matthew Stallman 会怎样？就像现在这样吗？

05 Linux 不会再年青了

我们都知道，引用任何名人名言并不会为自己的论述增加一丝一毫的正确性。

前文所说到的 GNU Hurd 的确在设计上是一个非常优秀的微内核操作系统。但因为错失良机，使得更多开发者投入了 Linux 的泥潭之中。

Linux 被无数人包装成 GNU 计划的伟大产物， GPL 的伟大产物， Linus 独裁领导下的伟大产物。没有任何人可以以任何理由批评他，除了 Linus 本人。事实上这就是皇帝的新衣，你想掀开 Linux 光鲜亮丽的外衣去一窥内部却发现 Linux 外边根本没有穿着任何衣物。

在抨击 FreeBSD 社区以前，请看看泥潭般的 Linux 社区，这不是五十步笑百步，而是大哥莫说二弟。此处的读者可以移步之前的文章——《Linux 社区已经成为了一个肮脏的泥潭》。 Linux 社区的那些巨兽，那些搞苦难哲学的是最没有资格说别人的社区肮脏污秽不堪的人——他们天天说着 RTFM。关于 Linux 社区的种种一直无人敢说，无人敢言，任由他们这群巨兽去践踏，去污蔑，去破坏别的社区。

Linux 社区早就不像当年那样了——我们都知道新人的确很菜，也喜欢抱怨，并且带有浓厚的 Windows 习惯，但既然在这里询问，我们就应该有责任帮助他们解决问题，而不是直接泼冷水、简单的否定或发表对解决问题没有任何帮助的帖子。乐于分享，以人为本，这正是 Ubuntu 的精神所在。

他们会把那些问题叫做“日经”，然后将你移出他们的社区。当然你不得不承认有些人是不听劝的：你和他说四遍，发了各种形式的教程（既有图片又有视频还有文字），他说自己照着文档做了，你一看让他发照片。“这就是你做了的结果？配置文件一个字都没有改”他改了后正常运行了，也不会和你说一声谢谢，然后继续问一些“日经”问题，你再回他，他就会说你“指手画脚”，不会指导人只知道让他看文档，妨碍别人教他，然后还去 Linux 社区发帖说 FreeBSD 社区有问题，真是恶人先告状。但是他们往往会上报团取暖，觉得对方说的都对，是别人有问题，自己 100% 正确。然后弹冠相庆，互称兄弟。

你告诉他“lib32”兼容层很重要，后边安不上去，必须现在选。他会告诉你“老子就是不用垃圾的 32 位，老子就是要用纯净的 64 位系统，按不上是老子自己的事，你管不着”，“既然你都懂那还来问些什么呢？”这种人是没有任何资格抱怨 FreeBSD 社区的，他们连最低水平的素质都没有，他们只配在泥潭般的 Linux 社区待着，是 Linux 社区造成了他们这种畸形的人格的变态的心理——无论是提问者还是回答者，谁也看不上谁。

与 Linux 相关的荒谬事件更是数不胜数。一个 systemd 与 SysV init 的争论就搅得 Linux 界七荤八素。Systemd 试图统管一切——网络，日志，磁盘管理，引导，时钟，电源、Cron、区域语言……然后就造成了各大发行版的分裂，虽然已经很分裂了。更别说前文提及的 SELinux 这种东西是怎么进入 Linux 内核的了。红帽公司——他们引以为豪的 Linux 的商业布局正在一步一步地控制整个 Linux 的发展。Linux 的工具属性越来越强，那个曾经年青有趣的玩具般的 Linux 早已不在了。

06 屠龙者终成恶龙

Wer mit Ungeheuern kämpft, mag zusehn, dass er nicht dabei zum Ungeheuer wird. Und wenn du lange in einen Abgrund blickst, blickt der Abgrund auch in dich hinein. 与恶龙缠斗过久，自身也会成为恶龙；当你凝视深渊，深渊也在凝视你。

——Friedrich Wilhelm Nietzsche

Linux 打败了一系列的操作系统：GNU Hurd、FreeBSD、OpenBSD 以及其他一堆 BSD，在服务器领域击败了 Windows server、MacOS，在移动平台产生了 Android。

Linux 的发展看上去是 GPL 的胜利，是 Linus 独裁的胜利，也是这个开源社区的胜利，更是 Linux 商业布局的胜利。

事实果真如此吗？究竟是皇帝的新衣还是破窗理论呢？

新事物必将取代旧事物，万事万物的新陈代谢如此，Linux 也不会永远长久下去，终究有一天会有更优秀的操作系统脱胎于此，并取代他。这是事物运行发展的基本规律，无可避免，而这些人所做的只是为 Linux 延长寿命而已。看上去蓬勃发展的 Linux，其实早就垂垂老矣。

Linux 败局已定。

只是没人愿意揭开这皇帝的新衣罢了。

屠龙者终成恶龙—— Linux 和 GPL 正在阻碍着越来越多的新技术发展。他们逼迫人们耗费着大量的人力物力去维护一堆堪称“屎山”的代码——每错，我说的就是你们：GCC、Xorg 以及其所谓的替代品 Wayland。

07 结语

Linux 已经日薄西山了，尽早投入 GNU Hurd 的怀抱才是真正明智的选择，尽早离开 GPL 选择 Anti-996L 才是真正自由的选择。

当然，不管 Linux 是不是皇帝的新衣，有多么的苦难哲学，他都给这个世界带来了一种选择，时势造英雄， Linux 乘势而来，也必将乘势而终。

Linux 内核的版本号已经刷到了 5.17-rc5，那么也许还会有 6.17-rc5、7.17-rc5、8.17-rc5……

但是 Linux 败局已定，且无人能够拯救他。我们能做只是尽量去多使用他做一些事情，最后发挥一下 Linux 的余热。

第一节 游戏

经典游戏

五分钟游戏

注：斜体为暂未打包的游戏

	Gnome/GTK	KDE/Qt	备注
数独	gnome-sudoku	ksudoku	逻辑/益智游戏
数壹	hitori		逻辑/益智游戏
扫雷	gnome-mines	kmines	益智游戏
2048	gnome-2048	2048-qt	益智游戏
贪吃蛇	gnome-nibbles		休闲游戏
国际象棋	gnome-chess	knight	益智游戏
五子棋		bovo	益智游戏
拼图	gnome-taquin		益智游戏
配对拼图	gnome-tetravex		益智游戏
对对碰	gnome-mahjongg	kajongg	
俄罗斯方块	quadrapassel	kblocks	限时消除类游戏
机器人	gnome-robots		
黑白棋	iagno	kreversi	翻转棋
吃豆人		kapman	
华容道	gnome-klotski		
宝石迷阵	swell-foop	kdiamond	消除类游戏
快艇骰子	tali	kiriki	
四子棋	four-in-a-row	kfourinline	
炸弹人		granatier	
珠玑妙算	gnome-mastermind		逻辑/益智游戏
		klines	彩色线条游戏

	Gnome/GTK	KDE/Qt	备注
纸牌接龙	aisleriot		
	atomix	katomic	解谜游戏

更多逻辑/益智游戏请访问[网页谜题](#)，及[本地小游戏](#)。

经典开源游戏

Wine / PlayOnBSD 游戏

Renpy 游戏

`Renpy` 是一款视觉小说引擎，可以很方便地拿来制作互动视频游戏。由于游戏骨架是 `Python` 语言，因此可以很方便地移植到不同的系统平台上，如 Windows 及 Linux。

虽然 `Renpy` 暂时未对 `FreeBSD` 作系统适配，但是 `FreeBSD` 自己对其作了二次打包。如此一来，就可以在 FreeBSD 上畅玩互动游戏了么？显然不是！不过，我们可以作一番小小的尝试。

操作

1. 安装 `renpy`

```
# pkg install renpy
```

2. 下载游戏

这里以[心跳文学社！](#)为例，其它游戏也可同样操作。选择附有 Linux 版本的游戏解压。

3. 运行 `renpy`

在引擎界面左侧 `工程(Projects)` 处可以看到列出来刷新的游戏 `DDLC-1.1.1`，点击该游戏后，选择右下角的 `启动工程(Launch Project)` 即可加载游戏。

补充

- 游戏分发站: [itch](#)
- 尽量选择附有 Linux 版本的游戏

如果游戏仅支持 Windows 系统, 可通过 renpy 引擎打包 Linux 版本。

- rpa 文件解包: [unrpa](#)

```
python3 unrpa -mp "解包目录" "XXX.rpa"
```

- rpyc 文件解包: [unrpyc](#)

```
python3 unrpyc -c "XXX.rpyc"
```

这一步非必要, 仅为了方便翻译成其它语言。

Godot 游戏

其它

第二节 音视频播放器

第三节 音视频剪辑

音频剪辑

- Audacity : # pkg install audacity

视频剪辑

扣图

Unix 系统下相关软件有很多，这里我们简单介绍一下矢量制图程序 Inkscape 的使用方法

- [] 安装方式: `# pkg install inkscape`
- `Ctrl O` (拉丁字母) 打开图片
- 点击图片
- `Alt i` 改为矢量模式
- `Shift F6` 贝塞尔和直线模式
- `Ctrl A` 全选
- 选择 路径 --> 交集，实现扣图

想了解更多，可查阅 Inkscape [官方教程](#)

剪辑

- Olive 视频编辑器: `# pkg install olive-video-editor`

编辑字幕

- Aegisub: # pkg install aegisub

压缩字幕

- ffmpeg : # pkg install ffmpeg

```
$ ffmpeg -i 视频文件.mp4 -vf subtitles=对应字幕.ass 输出视频.mp4
```

第四节 教育

计算机科学

语言与翻译

代数学

简单的计算器 bc

安装命令： # pkg install gnubc

后缀表示法

注：本节内容参考了维基百科

从幼儿园接触数学开始，我们用的计算表达方式就是中缀表示法。如，要计算 3 和 4 的和，写作 3 + 4，操作符（本例是加号）处于操作数（即 3 和 4）的中间。

而在后缀表示法里，操作符置于操作数的后面。如上式改若用后缀表示法，则写作 3 4 + 。如果有多个操作符，操作符则置于第二个操作数的后面，所以常规中缀表示法的 3 - 4 + 5 在后缀表示法里写作 3 4 - 5 + 。

初看起来，后缀表示法十分反人类，然而这种方法更容易被计算机理解，可以利用堆栈结构减少计算机内存访问。

有一例中缀表达式 $5 + ((1 + 2) * 4) - 3$ ，改为后缀表示法为 $5\ 1\ 2\ +\ 4\ *\ +\ 3\ -$ 。下表给出了从左至右求值的过程，堆栈栏给出了中间值，用于跟踪算法。

输入	操作	堆栈	注释
5	入栈	5	
1	入栈	5, 1	
2	入栈	5, 1, 2	
+	加法运算	5, 3	1, 2 出栈，让结果 3 入栈
4	入栈	5, 3, 4	
*	乘法运算	5, 12	3, 4 出栈，让结果 12 入栈
+	加法运算	17	5, 12 出栈，让结果 17 入栈
3	入栈	17, 3	
-	减法运算	14	17, 3 出栈，让结果 14 入栈

计算完成时，栈内只有一个操作数，这就是表达式的结果 14 。

桌面计算器 dc

`dc` 是采用后缀表示法的开源计算器，传统上，采用中缀表示法的 `bc` 计算器程序是在 `dc` 之上实现的。

对于上例，我们试着用 `dc` 操作一番。终端输入 `dc -e '5 1 2 + 4 *`
`+ 3 - p'`，可得到 14 。问题解决。

几何学

几何绘图软件 geogebra

安装命令： # pkg install geogebra

线性规划

GLPK 安装: `# pkg install glpk`

线性规划里，我们经常会用到 单纯形法。在计算机层面，有很多软件可以解放我们的大脑，甚至你可以在 OFFICE 表格软件的“规划求解”里，得到很好的解答。这里我们简单介绍一下命令行免费线性规划软件包 GLPK 的使用方法。希望大家可以触类旁通，举一反三。

我们先来看一个例子，取自 Hamdy A. Tara 所著的 “Operations Research : An Introduction” 一书，例题 2.4-1：

有一家银行计划放贷，预计最高投放 1200 万。下表显示了相关数据。

贷款类型	利率	坏账率
个人	0.14	0.1
汽车	0.13	0.07
家用	0.12	0.03
农业	0.125	0.05
商业	0.1	0.02

坏账意味着不产生利润，本金也无法收回。为了同其它商业机构竞争，农业贷款和商业贷款之和不少于全部贷款的 40%；为了振兴房地产业，个人贷款、家用贷款和个人贷款三项总计中，其中的个人贷款占比不少于 50%；银行坏账率最高允许为全部贷款的 4%。

求解银行能获得最高利润的分配方式。

我们来逐步分析。先设个人、汽车、家用、农业及商业分别的贷款量为 x_1 、 x_2 、 x_3 、 x_4 以及 x_5 。

$$\text{利润} = 0.14 \cdot 0.9 x_1 + 0.13 \cdot 0.93 x_2 + 0.12 \cdot 0.97 x_3 + 0.125 \cdot 0.95 x_4 + 0.1 \cdot 0.98 x_5$$

$$\text{本金损失} = 0.1 x_1 + 0.07 x_2 + 0.03 x_3 + 0.05 x_4 + 0.02 x_5$$

$$\text{得目标函数 } \max z = \text{利润} - \text{本金损失};$$

接下来，读者可根据题意找出约束条件。

由以上的目标函数和约束条件，我们可以给出以下数学模型：

```
max z = 0.026 x1 + 0.0509 x2 + 0.0864 x3 + 0.06875 x4 + 0.78 x5

s.t. x1 + x2 + x3 + x4 + x5 <= 1200,
      x4 + x5 <= 0.4 (x1 + x2 + x3 + x4 + x5),
      x3 >= 0.5 (x1 + x2 + x3),
      0.1 x1 + 0.07 x2 + 0.03 x3 + 0.05 x4 + 0.02 x5 <= 0.04 (x1 + x2 +
      x3 + x4 + x5),
      xi >= 0 (i = 1, 2, 3, 4, 5)
```

新建一个文本文件，命名为 `example`，输入：

```
var x1 >= 0;
var x2 >= 0;
var x3 >= 0;
var x4 >= 0;
var x5 >= 0;

maximize z: 0.026*x1 + 0.0509*x2 + 0.0864*x3 + 0.06875*x4 + 0.78*x5;
s.t. C1: x1 + x2 + x3 + x4 + x5 <= 1200;
s.t. C2: x4 + x5 <= 0.4*(x1 + x2 + x3 + x4 + x5);
s.t. C3: x3 >= 0.5*(x1 + x2 + x3);
s.t. C4: 0.1*x1 + 0.07*x2 + 0.03*x3 + 0.05*x4 + 0.02*x5 <= 0.04*(x1 +
      x2 + x3 + x4 + x5);
end;
```

保存后，终端输入 `$ glpsol -m example -o example.out`，打开 `example.out` 文件可查看解答值为： $x_3 = 720$, $x_5 = 480$, 其余各项值为 0。

问题解决。

物理和化学

- 元素周期表 GPeriodic : # pkg install gperiodic

天文地理

- 星图软件 Stellarium: `# pkg install stellarium`
- Gnome 地图: `# pkg install gnome-maps`

艺术

音乐

- MIDI 编曲软件 LMMS : # pkg install lmms
- 制谱软件 MuseScore : # pkg install musescore

三维图像

- 三维图形图像软件 Blender : # pkg install blender

绘画

- Krita: # pkg install krita
- MyPaint: # pkg install mypaint

第五节 科研与专业工具

工具与软件

科学计算软件 GNU Octave: `# pkg install octave`

计算机代数系统 wxMaxima: `# pkg install wxmaxima`

Python 及其相关模块: `# pkg install python py38-scipy py38-pandas
py38-matplotlib py38-sympy`

统计学

分析学

统筹学

上一节我们使用 GLPK 求解线性规划问题，本节我们使用性能更加强大的 wxMaxima 来解决问题。以下代码仅供参考，想深入了解，可查看[官方文档](#)。

```
load("simplex")$  
maximize_lp(0.026 * x1 + 0.0509 * x2 + 0.0864 * x3 + 0.06875 * x4 +  
0.78 * x5,  
[x1 + x2 + x3 + x4 + x5 <= 1200,  
x4 + x5 <= 0.4 * (x1 + x2 + x3 + x4 + x5),  
x3 >= 0.5 * (x1 + x2 + x3),  
0.1 * x1 + 0.07 * x2 + 0.03 * x3 + 0.05 * x4 + 0.02 * x5 <= 0.04 *  
(x1 + x2 + x3 + x4 + x5)]),  
epsilon_lp = 0, nonnegative_lp = true;
```

软件给出的解答为： [436.608, [x5=480.0, x4=0, x3=720.0, x2=0, x1=0]]，可知与 GLPK 求得的答案相同。问题解决。

第一节 获取 FreeBSD 内核源码

FreeBSD 项目在 2021 年从 SVN 全面迁移到了 Git，即

<https://git.freebsd.org>

所以获取源代码的方式也产生了变化。

从 Git 获取源代码

```
# pkg install -y gitup
# gitup release # 具体版本需要参考当前 gitup 配置
https://github.com/johnmehr/gitup/blob/main/gitup.conf
# gitup current # 获取 current 源代码
```

从压缩包获取源代码(推荐)

该方法比较简单快捷。

以 FreeBSD 13.0 为例：

```
# fetch https://download.freebsd.org/ftp/releases/amd64/13.0-
RELEASE/src.txz
# tar xvzfp src.txz -C /
```

如果速度慢可以切换到
<https://mirror.bjtu.edu.cn/freebsd/releases/amd64/13.0-RELEASE/src.txz>

第二节 修改内核源码

第三节 编译内核

第四节 内核分析

第一节 如何报告 Bug

当你遇到问题时，你该怎么做？

首先，确定不是你自己的问题。

- 看看常见问题列表、文档手册什么的
- 善用搜索引擎！
- 搜索一下往期的邮件列表，看看有没有人提过类似的问题
- 问问邮件列表

如果你在刚升级的系统中遇到问题，看看 `/usr/src/UPDATING` 的说明，如果刚升级的第三方软件，看看 `/usr/ports/UPDATING` 和 `/usr/ports/CHANGES`。

然后，问问自己，这是 bug 吗？如果这个问题可以用问句来表示（例如“我怎么做……，哪里有……？”），那么请先去FreeBSD邮件列表问问。

如果这是 bug，确定你的版本是否受到支持，否则不会有人管的。

一定要确定这个 bug 是不是已经被修复了！

- 检查一下系统版本，是不是有更新和补丁
- 搜索 FreeBSD bug 管理系统，有人报告过类似的问题吗

如果没人报告过类似的问题，那么你就要写了

请别提交这样的问题：

- 某个应用有新版本（他们有自动通知！）
- 对于那些没人维护的软件，如果你没有补丁，只报告bug很可能没人理会（如果你想维护它，请提交你的请求）
- 如果你不能使问题复现，那么别人也很难解决
- 增加新功能（到 FreeBSD 邮件列表里去呼吁，但为什么不自己做一个！）

决定你该把问题报告给谁：

- 对于基本系统组件，文档和网站问题，请直接提交给 FreeBSD 开发者。
- 对于那些随系统发行，但由他人维护的软件，除非能确定这个问题与系统无关，否则也请提交给FreeBSD 开发者。
- 对于那些第三方软件，除非能确定这个问题与系统有关，否则请提交给软件开发者。

报告的书写提示：

- 不要把“Summary”栏空着，因为它是群发邮件的标题。
- “Summary”要有代表性，简明扼要。
- 如果你有补丁，一定要上传。

- 注明你的系统版本，电脑架构，若果是自己编译的软件，把编译设置写上。
- 如果问题可以复现，注明问题产生的条件。
- 如果是内核问题，请准备好你的：硬件配置，内核配置，是否开启调试选项，错误信息或者 `/var/log/messages`，声明你已经看过了 `/usr/src/UPDATING` 并没有解决问题（因为总有人会问），是否可以运行其他内核。
- 如果是第三方软件问题，请准备好你的：安装的软件，覆盖了 `bsd.port.mk` 的默认设置的所有环境变量，声明你已经看过了 `/usr/ports/UPDATING` 并没有解决问题（因为总有人会问）。
- 一个报告做一件事，不要报告一堆东西，除非它们有紧密的联系。
- 对有争议的问题要慎重，最好先讨论一下。
- 注意下排版格式，尤其是复制粘贴时。
- 记得备份一下报告，万一网不好，没提交上就惨了。
- 讲礼貌，大家几乎都是拿不到一分钱的志愿者，请多些理解。

现在，开始填表了！

注意，你的邮件是公开的，因此可能会收到骚扰，请做好防护或是使用公用邮箱！

- Summary：梗概，简明扼要。

- Severity: 严重程度，只影响我/影响某些人/影响许多人，不要过高评价
 - Category: 类别
 - kern 内核、库、内置驱动（手册 2-4），除了 USB 子系统、多线程库之外
 - bin 内置程序
 - ports 第三方程序
 - conf 配置文件或者启动脚本（手册5）
 - docs 文档或网站
 - usb USB 子系统
 - threads 多线程库
 - standards 标准遵循
 - （架构名称） 与架构有关
 - misc 其他，或者真的找不出来问题（最好先问问邮件列表）
 - Environment 环境，包括系统版本、程序版本、系统配置等
 - Description 问题描述，完整、详细。不要加上你对问题的猜测
-

提交之后，你会收到一则确认电子邮件，可以通过它来跟踪进展情况。保持关注，如果有人要求提供更多信息，记得及时回复，耐心等候，[坐和放宽](#)，相信你的问题会很快得到解决的！

祝你在 FreeBSD 大家庭中玩得愉快！

第二节 如何提交一个软件包

第三节 如何参与 FreeBSD 协作

第四节 C/C++ 环境的配置

第五节 Java 环境的配置

第六节 QT 环境的配置

第七节 Python 与 VScode

在这一节，我们将假设你已经成功设置好 `pkg` 工具。如果你还没有，请参考本书 第三章第四节 以进行设置。

Python

在 FreeBSD 中，不同的 Python 版本被分开包装。意思是说，Python3.8 和 Python3.11 属于不同的包。就好像 `11vm11` 和 `11vm13` 那样。

但是，就目前的 FreeBSD 13 来说，支持最完整的 Python 是 Python 3.8。`pip` 与 `numpy` 之类的工具只有针对 `py38` 的打包。这也就是 `python3` 这个虚包指向的是 `python38` 的原因。

所以想要安装Python，你需要这样做：

```
# pkg install python38 py38-pip
```

显式要求 `python38` 的原因是为了防止 `python3` 指向的那个 Python 未来发生变化以后，`pip` 却没有跟着进入新版本之类的事情发生。

当然你也可以通过：

```
# pkg install python311
```

之类的命令来要求一个更新的 Python 版本。但是请记住，最新版本的 Python 版本在 FreeBSD 上可能没有那么完全。

然而不管怎么说，这总好过某些 Linux 发行版，从来只维护一个 Python 版本，也不管这么做的后果是什么。

VS Code

在以前，FreeBSD 用户想要安装 VS Code 手动从 GitHub 上下载源码，自己配置 Electron 的编译环境，然后自己编译 VS Code。或者通过利用 Linux 兼容层的方式来直接运行 VS Code 的 RPM 包。

但是自从 VS Code 进入 ports tree，进而进入 pkg repo 以后，用户可以不用那么折腾了。直接：

```
# pkg install vscode
```

然后你就能写代码了。

需要注意的是，通过这种方式获取到的 VS Code 其实是 Code - OSS。Code - OSS 和 VS Code 的区别将不在这里展开陈述，有兴趣的人可以自己 [做做功课](#)。

这里只想提一下，目前已知微软的 Python/PyLance 插件，以及 LLVM 的 clangd 插件都能直接在 Code - OSS 运行，使用起来和 VS Code 没有任何区别。

微软的 Remote Development 在 Code - OSS 也没有任何问题。

设置同步服务看起来是不能用的。

第八节 Rust/Go 环境的配置

安装 Rust 语言

以下安装方式二选一

FreeBSD 打包

```
# pkg install rust
```

Rust 官方打包（不建议）

- 安装: `curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh`
- 升级: `cd ~/.cargo/bin/rustup update`
- 删 除: `./cargo/bin/rustup self uninstall`

安装成功后，输入 `rustc --version` 或 `cargo --version` 查看软件版本。

为美好的世界献上祝福

```
cd ~  
mkdir projects && cd projects  
cargo new greeting
```

```
cd greeting
```

```
ee src/main.rs
```

添加如下文本：

```
fn main() {
    println!("Hello, world!");
}
```

保存后，运行 `cargo run` 即可输出代码。

安装 Go 语言

以下安装方式二选一

FreeBSD 打包

```
# pkg install go
```

Golang 官方打包（不建议）：

- 下载二进制包：[下载地址](#)

选择 `goXXX.freebsd-amd64.tar.gz` 或 `goXXX.freebsd-386.tar.gz` 软件包。

- 解压二进制包：以 amd64 架构为例，终端代码如下

```
tar -C /usr/local -xzf  
https://golang.google.cn/dl/go1.18.1.freebsd-amd64.tar.gz
```

- 添加环境变量：

文本模式打开 `.profile`，添加一行 `export PATH=$PATH:/usr/local/go/bin`

安装成功后，输入 `go version` 查看软件版本。

为美好的世界献上祝福

新建名为 `helloWorld.go` 的文本，添加如下内容：

```
package main

import "fmt"

func main() {
    fmt.Println("Hello, World!")
}
```

保存后，在终端运行 `go run helloWorld.go` 即可输出代码。

第九节 Csh 与其他 Shell

FreeBSD csh shell 配置

在 `/etc/csh.cshrc` 里面加入：

```
alias ls ls -G
```

并重新登录

问：如何让 FreeBSD 的 csh 像 bash 那样按 Tab 列出列出无法补齐的候选文件？

答：标准的方法是按 `Ctrl+D`。

但如果一定要用 Tab 的话，在 `/etc/csh.cshrc` 中加入：`set autolist`

FreeBSD 如何让 csh 像 zsh 那样具有命令错误修正呢

比如你用 emacs 写 c 语言程序，但你输完 `emacs ma` 按 `Tab` 回车是，他会匹配所有 `ma` 开头的文件，而这个是忽略掉，也就是按 `Tab` 时不会在有你忽略的东西，对编程之类的友好，不用再匹配到二进制。`.o` 之类的文件，

```
set correct = cmd lz/usr/bin tcsh>ls /usr/bin (y|n|e|a)?
set fignore = (.o ~) emacs ma[^D] main.c main.c~ main.o emacs
ma[tab] emacs main.c
```

更换默认

警告：切换默认 Shell 会导致 恢复模式无法正常启动加载命令行环境。

例如切换到 zsh：

```
# pkg install zsh zsh-autosuggestions zsh-syntax-highlighting
# chsh -s /usr/local/bin/zsh
# touch ~/.zshrc
# ee ~/.zshrc #添加下面几行
source /usr/local/share/zsh-autosuggestions/zsh-autosuggestions.zsh
source /usr/local/share/zsh-syntax-highlighting/zsh-syntax-
highlighting.zsh
source ~/.p10k.zsh
```

切换到 bash：

```
# pkg install bash
# chsh -s /usr/local/bin/bash
# ee ~/.bash_profile
```

第十节 通过 IDA 7 调试 FreeBSD

注意：Windows 、IDA 、FreeBSD、FreeBSD 兼容层 均需要 64 位，否则可能无法正常使用。

首先在 Windwos 系统里 IDA 的安装路径里找到 dbgsrv 文件夹里的 `linux_server64` 文件。

复制到 FreeBSD 里，可以用 Winscp

那就复制 `linux_server64` 和你需要远程调试的文件 target（假设）到 `/root/reverse` 文件夹里（文件夹任意），给权限 `777`，并且运行 `linux_server64`。

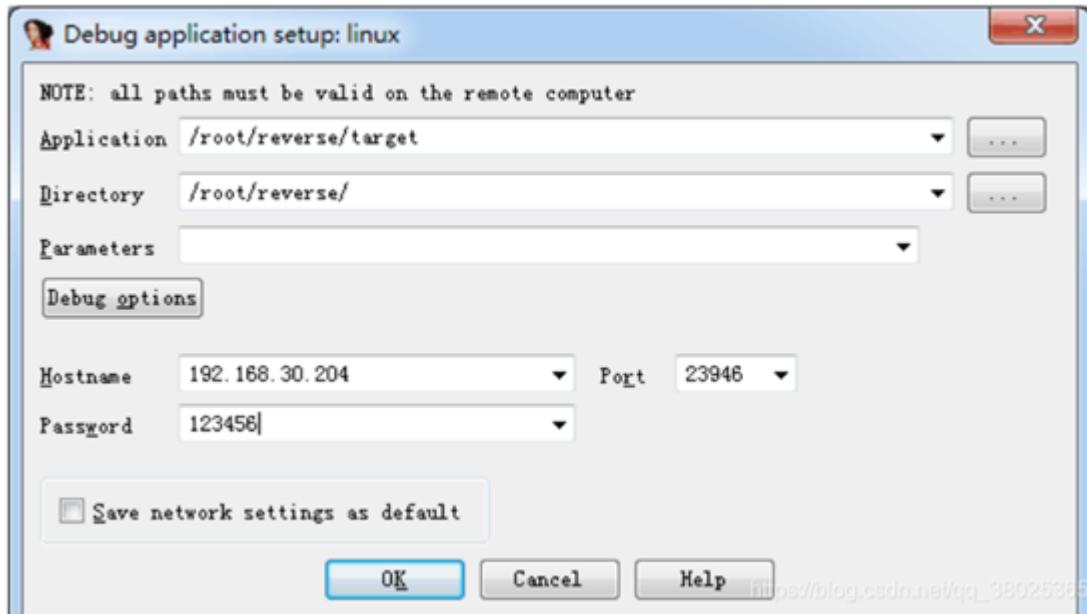
并且运行，参考红色方框。

```
root@FreeBSD:~ # ls
.cshrc          .k5login       .mysql_history  .profile      mt_server
.history        .login         .mysql_secret   linux_server64
root@FreeBSD:~ # ./linux_server64
IDA Linux 64-bit remote debug server(ST) v1.25. Hex-Rays (c) 2004-2018
Listening on 0.0.0.0:23946...
2021-11-25 17:09:01 [i] Accepting connection from 192.168.1.7...
Last login: Thu Nov 25 17:11:52 2021 from 192.168.1.7
FreeBSD 12.2-RELEASE r366954 GENERIC

Welcome to FreeBSD!

Release Notes, Errata: https://www.FreeBSD.org/releases/
```

请用 64位 的 IDA，按照如下截图操作。



第一个是要调试的文件在虚拟机里的位置。

第二个是要调试的文件在虚拟机里的路径

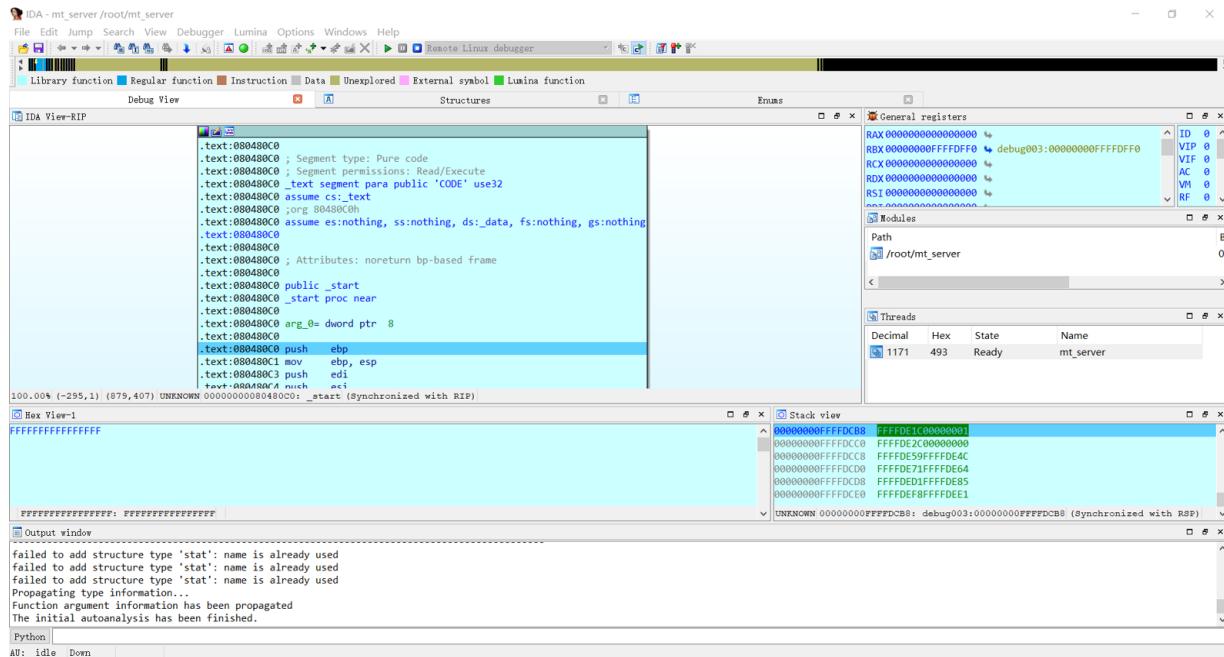
这里 target 就是具体要调试的文件。

第三个是要传递给 main 函数的参数，一般不写。

然后是 FreeBSD 系统的主机 ip 地址，监听的端口号和密码（即 SSH 密码，这里是 root 用户的，因为 `linux_server64` 文件运行在 root 用户下）

在 FreeBSD 系统终端 `ifconfig -a` 可以查看到自己的 ip 地址

成功结果如下：



第十一节 Git

第一节 恢复模式与密码重置

开机按 2 进入 `single user` 单用户模式。

UFS 文件系统

```
# mount -u /
# mount -a -t ufs
```

ZFS 文件系统

```
# mount -u  
# zfs mount -a
```

使用 U 盘设备

```
# mount /dev/adaXpN -o rw /mnt
```

x、N 的参数取决于具体设备。

第二节 FreeBSD 多硬盘 EFI 引导统一

何谓多硬盘 EFI 统一？就是说有两块硬盘，两块硬盘上分别都有EFI 分区，一个分区里是 FreeBSD，另一个是 Windows。现在只想保留一个分区，即想把EFI配置文件放到一块硬盘的EFI分区里统一管理。

设装有 Windows 的硬盘为 `ada0`，FreeBSD 的硬盘为 `nvd0`。

首先关闭 Windows 的快速启动启动：命令为 `powercfg /h off`，

然后关机重启进入 FreeBSD，创建挂载点 `mkdir /mnt/efi`。

检测 `ada0p1`（硬盘的第一个分区）是不是我们要挂载的 EFI 分区，输入命令 `fstyp /dev/ada0p1`，我的输出是 NTFS，可见不是我们想要的EFI 分区；`fstyp /dev/ada0p2`，输出 `msdosfs`，是我们的 Windows 磁盘上的 EFI 分区。

接下来挂载 `ada0` 磁盘上的EFI分区到 FreeBSD 的 `/mnt/efi`：
`# mount -t msdosfs /dev/ada0p2 /mnt/efi`

为FreeBSD 引导性创建 EFI 路径下的目录：
`# mkdir /mnt/efi/EFI/freebsd`

然后复制启动文件到该路径
`# cp /boot/boot1.efi /mnt/efi/EFI/freebsd/bootx64.efi`

最后生成启动项：
`# efibootmgr -c -l /mnt/efi/EFI/freebsd/bootx64.efi -L "FreeBSD niu pi"`

重启进入 Windows , 使用 easyuefi 激活 FreeBSD niu pi 这个启动项即可。如没有问题, 可使用 DiskGenius 删除 nvd0 磁盘的 EFI 分区文件。

第三节 FreeBSD 中文 TTY 控制台

FreeBSD 新型终端 VT，支持 `cjk`，所以丢个字体进去，就能显示中文了

1. 首先你没有改过控制台程序，使用的是默认的……
2. 最新版本，本说明是以 FreeBSD 12.1 release 为例。

字体格式为 `.fnt`

命令： `$ vidcontrol -f ABC.fnt`

系统提供了一个工具，可用于将 bdf, hex 转换为 fnt

```
vtfontcvt [ -h 高度 ] [ -v ] [ -w 宽度 ] [字体]
```

命令一般都是临时的，若要永久生效，将其加入 `rc.conf`

第四节 引导界面

开机时间调整为 2 秒

```
# ee /boot/loader.conf  
autoboot_delay="2"
```

第五节 Grub 及其他引导

注意，本书所有 `grub` 均为 `grub2`。

根据 FreeBSD 开发者介绍由于 `grub` 的 ZFS 模块不采用 BSD 提供的，而是自己开发的，导致 `grub` 无法直接引导 FreeBSD 的内核从而启动系统。只能采取 `chainload+1` 的方式间接引导。

```
menuentry "FreeBSD-13.0 Release" {
    set root='(hd0,gpt1)' # 请自己检查
    chainloader /boot/boot1.efi
}
```

第一节 System INIT 管理服务

基础

FreeBSD 使用 BSD INIT 管理系统服务。

- 启动一个服务: `# service XXX start`
- 停止一个服务: `# service XXX stop`
- 重启一个服务: `# service XXX restart`

出于安全性考虑，服务安装以后默认是禁用状态，以上命令是无法执行的，需要先开启服务：

```
# ee /etc/rc.conf
```

添加一行，`XXX_enable="YES"`，`XXX` 表示服务名称（这里只是举例，实际上可以是 `nginx samba` 等），这是固定格式：

```
XXX_enable="YES"
```

也可以用命令添加：

```
sysrc XXX_enable="YES"
```

服务所对应的脚本路径是：`#/usr/local/etc/rc.d/`

当然也可以直接调用 `/etc/rc.d/` 和 `/usr/local/etc/rc.d/` 下的那些脚本。

- `# /usr/local/etc/rc.d/XXX reload`
- `# /usr/local/etc/rc.d/XXX stop`

如果 `rc.conf` 中并没有启用某项服务，但想临时启动它，那么可以这样：

- `# service XXX onestart`
- `# service XXX onestop`

进阶

rc.conf 掌管着所有系统服务。与之相关的文件和路径如下：

1. 默认的配置位于 `/etc/default/rc.conf`。尽量不要对其进行修改。
2. 用户自定义的配置位于 `/etc/rc.conf`。例如，如果想让系统自动启动 ssh、ipfw、nginx 等服务，就要修改本文件。注意，如果某项配置与默认的配置有冲突，则以本文件为准。
3. 基系统的服务脚本位于 `/etc/rc.d/`。第三方应用的服务脚本位于 `/usr/local/etc/rc.d/`。当遇到问题时，通过查阅配置文件，找出问题所在。

`/etc/rc.conf` 常用配置文件

```
hostname="server.shuang.ca" #设定主机名
ifconfig_vtnet0="inet xxx.xxx.xxx.xxx netmask 255.255.255.0" #设定IP
地址, 其中vtnet0是网卡名称, 注意设置正确
defaultrouter="xxx.xxx.xxx.1" #网关地址
syslogd_enable="YES" #开启日志
syslogd_flags="-s -s" #禁止接收其他主机的日志
fsck_y_enable="YES" #开机自动fsck硬盘
enable_quotas="YES"
check_quotas="YES" #系统配额
clear_tmp_enable="YES" #开机自动清空/tmp
update_motd="NO" #禁用内核信息提示
icmp_drop_redirect="YES"
icmp_log_redirect="YES" #ICMP重定向
log_in_vain="YES" #记录每一个企图到关闭端口的连接
accounting_enable="YES" #系统审计功能
```

periodic.conf

FreeBSD 默认有一些周期执行的任务，它们是通过 `periodic` 命令执行的，由 `cron` 自动调用。与 `periodic` 有关的配置和路径如下：

1. 默认的配置位于 `/etc/defaults/periodic.conf`。
2. 用户自定义的配置位于 `/etc/periodic.conf`。
3. 基系统的任务脚本位于 `/etc/periodic/`。
4. 第三方应用的任务脚本位于 `/usr/local/etc/periodic/`。

以 `locate` 命令的所依赖的路径数据库 `/var/db/locate.database` 为例，

该数据库由 `/etc/periodic/weekly/310.locate` 这个脚本每周更新一次。

如果你要立即更新，也可以直接执行这个脚本。

其他配置文件

- crontab: cron 配置，位于 `/etc/crontab`，请参考 `man crontab`。
- syslog.conf: 系统日志配置，位于 `/etc/syslog.conf`，请参考 `man syslog.conf`。
- loader.conf: 系统启动配置，位于 `/boot/loader.conf`，请参考 `man loader.conf`。
- sysctl.conf: 内核参数配置，位于 `/etc/sysctl.conf`，请参考 `man sysctl.conf`。

第二节 FreeBSD 目录结构

FreeBSD 设计上属于学院派，条理清晰。

路径	简介
/bin	在单用户和多用户环境下的基本工具目录。
/sbin	在单用户和多用户环境下的存放系统程序和管理所需的基本实用目录。
/etc	系统的配置和脚本。
/usr/bin	存放系统应用软件。
/usr/sbin	存放系统后台程序 和 系统工具（由用户执行）。
/usr/libexec	存放系统实用或后台程序（从另外的程序启动执行）。
/tmp	临时文件。 /tmp 目录中的内容，一般不会在系统重新启动之后保留。
/var/log	存放各种系统日志。
/var/tmp	临时文件。 这些文件在系统重新启动时通常会保留， 除非 /var 是一个内存中的文件系统。
/var/run	用来存放 Pidfile。

值得注意地是，FreeBSD 现在并不使用 /proc 以及 procfs。

对于用户安装的程序，允许写入的目录是：

- /var/run
- /var/log

- /var/tmp
- /tmp

用户安装的程序都统一在 /usr/local 下，比如：

- /usr/local/bin
- /usr/local/sbin
- /usr/local/etc
- /usr/local/libexec

简而言之：系统使用 /usr，用户使用 /usr/local。这点与 Linux 截然不同，虽然后者理论设计也是如此，但是实际上很难做到。

更多信息，请参考官方的文档：

<https://docs.freebsd.org/en/books/handbook/dirstructure.html>

第三节 bsdinstall 与 bsdconfig

bsdinstall

`bsdinstall` 是安装界面的命令。

`bsdconfig`

`bsdconfig` 命令的界面可以进行许多有用的设置，比如网络设置，需要注意的是，该界面无法连接 WiFi，因为无法输入密码。

第四节 禁用 Sendmail

FreeBSD 系统中的 sendmail 一直默认启动，对于大多数人来说是无用的，这个可以在安装时禁止，详见安装说明。

编辑 `/etc/rc.conf`，加入以下几行：

```
sendmail_enable="NO"
sendmail_submit_enable="NO"
sendmail_outbound_enable="NO"
sendmail_msp_queue_enable="NO"
```

编辑 `/etc/periodic.conf`，加入以下几行，关闭某些 Sendmail 才会用到的设定。

```
daily_clean_hoststat_enable="NO"
daily_status_mail_rejects_enable="NO"
daily_status_include_submit_mailq="NO"
daily_submit_queuerun="NO"
```

第五节 利用脚本自动生成 BSD libc 库文本

首先安装依赖:

```
# pkg install netpbm groff ghostscript9-base
```

```

#!/bin/sh

pstarget="/tmp/$$.libcdoc.ps"
pdftarget="libcdoc.pdf"
pdftarget_noidx="/tmp/$$.${pdftarget}"
pdfindex="/tmp/$$.pdfindex.info"
index="/tmp/$$.index"
sorted_index="$index.sorted"
flist="/tmp/$$.flist"
tocin="/tmp/$$.toc.mdoc"
keywords="/tmp/$$.keywords"
mandir="/usr/share/man"
paths="$mandir/man2 $mandir/man3"
content_offset=0

mkidx()
{
    for i in `find $paths -name "*.gz"`; do
        if zgrep -q '.Lb libc' $i && zgrep -q '.Sh LIBRARY' $i; then
            for j in `gettitles $i`; do
                echo "$j:$i" >> $index
            done
        fi
    done
    cat $index | sort -n | uniq | awk -F: 'BEGIN { prev = "" } {
        if ($1 != prev) {
            print $0;
        }
        prev = $1;
    }' > $index.tmp
    mv $index.tmp $index

    for i in `cat $index`; do
        fname=`echo $i | cut -d: -f2`
        grep $fname $index | sort -n | awk -F: 'BEGIN {n = 0} {

```

```

        if (n++ > 0)
            printf ",";
        printf "%s", $1;
    }'
    echo ":$fname"
done | sort -n | uniq > $index.tmp
mv $index.tmp $index

currp=1
for i in `cat $index` ; do
    fname=`echo $i | cut -d: -f2`
    keywords=`echo $i | cut -d: -f1`
    nextp=`mandoc -T ps $fname|egrep '^%Pages: [0-9]+'|cut -d: -
f2`
    echo "$keywords:$currp:$fname" >> $index.tmp
    currp=`expr $currp + $nextp`
done
mv $index.tmp $index
for i in `cat $index | sed -E 's/(^[^:]+):.*$/\1' | tr ',' ' '`; do
    echo $i
done | sort -n > $keywords

for i in `cat $keywords` ; do
    page=`grep -w $i $index | tail -1 | cut -d: -f 2`
    echo $i:$page
done > $sorted_index
}

mkpsdoc()
{
    for i in `cat $index` ; do
        fname=`echo $i | cut -d: -f3`
        zcat $fname | sed -e 's/^.\Dd.*$/.Dd __PAGENO__/'
                           -e '/.Os.*/d' | mandoc -T ps >> $pstarget
    done
}

```

```

mktoc()
{
    echo ".XS 1" > $tocin
    echo "Table of Contents" >> $tocin
    for i in `cat $sorted_index` ; do
        keyword=`echo $i | cut -d: -f 1`
        page=`echo $i | cut -d: -f 2`
        page=`expr $content_offset + $page`
        printf ".XA $pagen$keywordn" >> $tocin
    done
    echo ".XE" >> $tocin
    echo ".PX" >> $tocin
}

get_content_offset()
{
    mktoc
    content_offset=`groff -T ps -ms $tocin | egrep '^%Pages: [0-9]+' |
        cut -d: -f2`
    content_offset=`expr $content_offset + 0`
}

prepend_toc()
{
    in=$1
    tmp=$in.tmp

    groff -T ps -ms $tocin > $tmp
    cat $in >> $tmp
    mv $tmp $in
}

mkpdfidx()
{
    printf "[/Page 1 /View [/XYZ null null null]] " > $pdfindex
}

```

```

printf "/Title (Table of Contents) /OUT pdfmarkn" >> $pdfindex

for i in `cat $sorted_index`; do
    kword=`echo $i | cut -d: -f 1`
    page=`echo $i | cut -d: -f 2`
    page=`expr $page + $content_offset`
    printf "[/Page $page /View "        >> $pdfindex
    printf "[/XYZ null null null] "     >> $pdfindex
    printf "/Title ($kword) /OUT pdfmarkn" >> $pdfindex
done
}

gettitles()
{
    zcat $1 | sed -n '/.Sh NAME/,/.Sh LIBRARY/p' |
        egrep '^.\Nm [^ ]+' | cut -d" " -f 2 | sort -n | uniq
}

mkidx
mkpsdoc
get_content_offset
mktoc
prepend_toc $pstarget
mkpdfidx

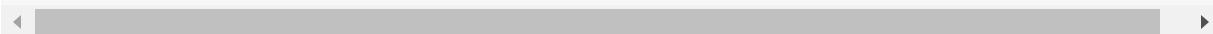
cat $pstarget | awk -v p=$content_offset '{
    if ($0 ~ /(__PAGENO__)/) {
        t = sprintf("(%s)", ++p);
        sub(__PAGENO__, t);
    }
    print $0;
}' > $pstarget.tmp

mv $pstarget.tmp $pstarget

ps2pdf $pstarget $pdftarget_noidx

```

```
gs -sDEVICE=pdfwrite -q -dBATCH -dNOPAUSE -sOutputFile=$pdftarget  
$pdfindex  
-f $pdftarget_noidx  
  
rm -f $tocin  
rm -f $pstarget  
rm -f $index  
rm -f $pdftarget_noidx  
rm -f $pdfindex  
rm -f $sorted_index
```



运行脚本即可在同路径文件夹下找到 PDF 文档。现成的文档请看：

<https://github.com/FreeBSD-Ask/BSelibc>

第六节 BSD 风格的 make/grep/sed/awk

FreeBSD 的 `make/grep/sed/awk` 与 GNU 那套有所不同。

第一节 FreeBSD 设计概要

第二节 内核

第三节 进程

第四节 内存管理

第五节 安全

第六节 I/O 系统

第〇节 概述

OpenBSD，也是一款类 Unix 计算机操作系统，诞生于1995年，由荷裔加拿大的程序员西奥·德·若特（Theo de Raadt）从 NetBSD 分支而来。采用了 LLVM/Clang 项目来构建系统，默认 shell 为 ksh，吉祥物是一只名为普菲（Puffy）的河豚。

相比于其它 BSD 系统，OpenBSD 的诉求倾向于安全级别。对于此，用户体验可能会见仁见智，毕竟这样的后果之一是软相对较少，不仅远远落后于 FreeBSD，甚至比起 NetBSD 也略逊一筹。不过好消息是，OpenBSD 以少量的人力和物力维护了 amd64/i386、arm64/armv7 及 riscv64 等诸多架构，是一款实实在在的通用操作系统。为了拓展桌面用户，也打包了 Gnome 、 KDE 和 XFCE 等桌面环境和 Blender 、 Firefox、Krita 及 libreoffice 一大批软件。相信在未来的发展道路上，OpenBSD 会逐步优化性能，更好地为全社会服务。

在中文互联网上，OpenBSD 被不少人熟知，概因一条让人感慨万千的新闻——在 2014 年，项目因欠缴电费，面临关停的风险。而后不少公司对其施以援手，其中包括国内的锤子科技。

第一节 安装

下载镜像

以 OpenBSD 7.1, AMD 64 架构为例，访问

<https://cdn.openbsd.org/pub/OpenBSD/7.1/amd64>

获取系统镜像。若是刻录 U 盘 安装，就下载 `installXX.img` ；若是虚拟机体验，请下载 `installXX.iso` 。（注：截止 OpenBSD 7.1 时，请不要使用 ventry 引导实体机安装。）

自定义安装

这里推荐大家使用 **自定义安装**，不要使用系统推荐的方式。为了安全考虑，OpenBSD 默认大量分区。新用户初次遇到，会一头雾水，极不适应。

推荐分区

对于尝鲜的朋友，假设有 32GB 的容量，推荐两个分区：`swap 2GB`，`/` 为剩下的全部容量。（UEFI 请自行创建一个 100~300M 大小的 FAT32 格式的 EFI 分区，或者使用已有分区，下同）

如果有 128GB 容量，推荐分区：`/ 32GB`，`swap 4GB`，`/home` 为剩下的全部容量。

对于更大的容量，可依自己喜好，进一步细化分区配置。

安装过程

```
Welcome to the OpenBSD/amd64 7.1 installation program.
```

```
(I)nstall, (U)pgrade or (S)hell? i
```

选择 `i` 进行安装

```
System hostname? (short form, e.g. 'foo') XiaoMing
```

系统主机名，可以选择一个字母少的，将来会显示 `XiaoMing.DHCP` 这样的主机名。

Available network interfaces are: em0 rtwn0.

Which one do you wish to configure? (or 'done') [em0]

这一步选择网络连接。为免去不必要麻烦，请尽量选择有线网络。可先输入 `?` ，详细了解网络名称后再选择。如本例中 `em0` 为有线网络，`rtwn0` 为无线网络。

后续配置直接 回车键 确认即可。

Password for root account? (will not echo)

设置 root 账号密码，输入后回车确认（密码不会在屏幕上显示）。

Password for root account? (again)

再次输入一遍 root 账号密码，回车键确认。

后续配置回车确认。

Do you want the X Window System to be started by xenodm?

[no] `no`

是否运行图形窗口界面，这一步选择 `no`，后续我们会安装自己需要的桌面环境。

Setup a user? (enter a lower-case loginname, or 'no')

[no] `XiaoMing`

设置一个用户名。

Full user name for XiaoMing?

用户全名，可随意输入。

| Password for XiaoMing account? (will not echo)

为该账号设置密码（密码不会显示在屏幕上）。

| Password for XiaoMing account? (again)

再次输入该用户名的密码。

后续配置回车确认。

| What timezone are you in? ('?' for list) [US/Eastern] ?

时区选择，找到 `Asia/Shanghai`

| Available disks are: sd0, sd1, sd2. Which one is the root disk? (or 'done') [sd0] ?

这一步是选择要将系统安装在哪一块硬盘。按 `?` 列出识别的所有硬盘。请务必记住所有的盘符：所要安装系统的盘符，以及我们的U盘盘符。然后输入需要安装的位置，如 `sd0`。

再次提醒：请确认好目标硬盘，否则悔之晚矣！

| Use DUIDs rather than device names in fstab? [yes]

回车确认默认选择。

| Use (W)hole disk, use the (O)penBSD area, or (E)dit the MBR? [whole]

是否选择全部硬盘空间，回车选择全部。

| Use (A)uto layout, (E)dit auto layout, or create (C)ustom layout? [a] `c`

这一步一定要选择 `c` , 即 `自定义设置` 。

`p m`

输入 `p m` 来显示硬盘。其它选项如下表:

代码	作用
<code>p m</code>	查看分区大小
<code>a</code>	增加分区
<code>d</code>	删除分区
<code>z</code>	删除全部分区
<code>q</code>	确认分区

以下假设一块 64 GB 硬盘, 分区为: / 20GB , swap 4GB , 然后剩下的空间全部划分给 /home 。

```
> a  
partition: [a]  
offset: [64]  
size: [xxxxxxxx] 20g  
Rounding size to cylinder (bbbb sectors): yyyy/yyyy  
FS type: [4.2BSD]  
mount point: [none] /  
Rounding size to bsize (w sectors): zzzzzzzz  
>
```

这里设置了 20GB 的 / 分区，阴影框为我们输入的设置，其余的皆为回车键默认选择。

```
> a  
partition: [b]  
offset: []  
size: [xxxxxxxxxx] 4g  
Rounding size to cylinder (w sectors): zzzzzz  
FS type: [swap]  
>
```

这里设置了 4GB 的 swap 分区，**阴影框** 为我们输入的设置，其余的皆为回车键默认选择。

```
> a  
partition: [d]  
offset: [aaaaaaaa]  
size: [ASDFGHJKL]  
FS type: [4.2BSD]  
mount point: [none] /home  
Rounding size to bsize (w sectors): zzzzzzzz  
>
```

注意 **size** 一栏里我们并未输入数值，而是直接回车，意味着上步余下的全部容量都给了该分区，即 **/home** 分区。

配置完毕，记得输入 **q** 确认。

```
> q
```

以上，分区完毕。

```
| Let's install the sets! Location of sets? (cd disk ftp  
| http or 'done') [cd] disk
```

软件地址，选择 **disk**。这里我们选择安装盘为软件地址。

```
| Is the disk partition already mounted? [yes] no
```

需要提示一点的是，系统询问是否已识别 U盘时，一定要选择否，以便我们再确认一遍 U盘位置。如不确定 U盘编号，可输入 `? 查` 看。

```
Select sets by entering a set name, a file name pattern or 'all'. De-select  
sets by prepending a '-' to the set name, name pattern or 'all'.  
Selected sets are labelled `'[X]'`  
  
[X] bsd          [X] etc71.tgz      [X] xbase71.tgz  
[X] xserv71.tgz  [X] bsd.rd       [X] comp71.tgz  
[X] xetc71.tgz   [X] bsd.mp       [X] man71.tgz  
[X] xshare71.tgz [X] base71.tgz    [X] game71.tgz  
[X] xfont71.tgz  
  
Set name(s)? (or 'abort' or 'done') [done] -game*
```

这里我们输入 `-game*` 来取消 `game71.tgz`，其它都勾选。

注：即使不使用桌面，也请勾选 `x11` 选项，否则部分软件可能无法正常运行。

```
Set name(s)? (or 'abort' or 'done') [done] ` -game*`  
  
[X] bsd          [X] etc71.tgz      [X] xbase71.tgz  
[X] xserv71.tgz  [X] bsd.rd       [X] comp71.tgz  
[X] xetc71.tgz   [X] bsd.mp       [X] man71.tgz  
[X] xshare71.tgz [X] base71.tgz    [ ] game71.tgz  
[X] xfont71.tgz  
  
Set name(s)? (or 'abort' or 'done') [done]
```

回车确认。

```
| Location of sets? (cd disk ftp http or 'done') [done]
```

继续回车确认。此后开始安装系统。约 5 分钟后，会出现如下提示：

```
CONGRATULATIONS! Your OpenBSD install has been successfully  
completed!
```

```
To boot the new system, enter 'reboot' at the command prompt.  
When you login to your new system the first time,  
please read your mail using the 'mail' command.
```

恭喜！系统已成功安装，重启后可进入系统。

第二节 配置

初次登录

获取驱动

第一次进入系统后，OpenBSD 会自动检测无线、显卡和声卡，并下载相关驱动。静等几分钟，待其自行更新。由于国外网站连接比较慢，如果等待时间过长，可 `Ctrl + c` 取消，待进入系统后运行 `# fw_update` 重新获取驱动。

桌面支持

上一节安装时，我们屏蔽了桌面选项。这一步我们重新开启。打开 `/etc/sysctl.conf`，添加一行 `machdep.allowaperture=2`。

修改软件源

打开 `/etc/installurl`，将默认源注释掉，改为
`https://mirrors.bfsu.edu.cn/OpenBSD`。此处我们选择了北外源，用户也可选择 [清华镜像源](#)、[阿里镜像源](#)、及[南京大学源](#) 等。

系统更新

普通账号获取权限

以 root 账号登录系统，而后新建 `/etc/doas.conf` 文本，打开 `doas.conf`，添加一行 `permit persist :wheel`

内核更新

内核更新: `# syspatch`

驱动升级: `# fw_update`

软件升级: `# pkg_add -u`

修改shell: `chsh`

示例: `# chsh -s /usr/local/bin/bash $USER`

软件管理

- 查找软件: `# pkg_info -Q foo`
- 安装软件: `# pkg_add foo`
- 升级软件: `# pkg_add -iu foo`

挂载可移动磁盘

新建挂载点

```
$ cd ~  
$ mkdir media  
$ cd media  
# mkdir first second third forth
```

查看盘符

使用 `dmesg` 命令来查看新插入的盘符，如格式为 fat32 的 U盘，可能在 OpenBSD 系统里盘符为 `sd1`。

检查分区

如插入的盘符为 `sd1`，则输入 `disklabel sd1` 查看分区情况。如下

```
#          size      offset  fstype [fsiz bsize   cpg]  
c:    60062500          0  unused  
i:    60062244        256   MSDOS
```

挂载

由上则可知分区为 `i`，使用以下命令挂载：

```
# mount /dev/sd1i /$USER/media/first , $USER 替换为当前用户名。
```

其它格式

OpenBSD 可挂载的外接硬盘格式有 NTFS、ext2/ext3 以及 CD 磁盘等，具体命令可参考如下：

```
# mount /dev/sd3i /$USER/media/first    # fat32
# mount /dev/sd2k /$USER/media/second   # NTFS
# mount /dev/sd1l /$USER/media/third    # ext2/ext3
# mount /dev/cd0a /$USER/media/forth   # CD
```

卸载磁盘

```
# umount /$USER/media/first
```

无线测试

OpenBSD 里的无线网络，配置文件通常是 `hostname.if` ，其中 `if` 为无线驱动名称+序号。如一台笔记本无线型号为 `rtl8188cu`，OpenBSD 下驱动为 `rtwn`，序号从 0 开始。为了让系统自动加载无线，可打开 `/etc/hostname.rtwn0` 文件，而后添加：

```
dhcp  
join 无线名称 wpakey 无线密码
```

保存后即可。

补遗

加载触摸板

打开 `/etc/wsconsctl.conf`，添加一行 `mouse.tp.tapping=1`。

加载多线程

打开 `/etc/sysctl.conf`，添加一行 `hw.smt=1`。

相关资料

- OpenBSD FAQ 推荐
- Absolute OpenBSD 补充

如果觉得 *Firefox* 运行缓慢，试试 *Epiphany* (*Gnome* 浏览器)。

第三节 换源

打开 `/etc/installurl`，注释掉官方源，改为如下国内源（任选其中一个）：

- 北外软件源：<https://mirrors.bfsu.edu.cn/OpenBSD>
- 清华大学软件源：<https://mirrors.tuna.tsinghua.edu.cn/OpenBSD>
- 上海交大软件源：<https://mirror.sjtu.edu.cn/OpenBSD>
- 南京大学软件源：<https://mirrors.nju.edu.cn/OpenBSD>
- 腾讯软件源：<https://mirrors.cloud.tencent.com/OpenBSD>
- 阿里云软件源：<https://mirrors.aliyun.com/pub/OpenBSD>

保存后即可。

第四节 包管理器

同其它 Unix-like 系统一样，OpenBSD 的软件安装主要有两种方式：采用官方预编译好的二进制包，以及通过源代码自己打包安装。这里我们推荐第一种方式安装。

二进制包

我们推荐以二进制包的方式来安装软件，以火狐浏览器为例：

- 安装软件 `pkg_add firefox`
- 删除软件 `pkg_delete firefox`
- 查询软件 `pkg_info -Q firefox`
- 升级软件 `pkg_add -iu firefox`

此外，全局的命令有：升级所有软件 `pkg_add -iu`；删除所有软件包缓冲 `pkg_delete -a`。

ports

[查询网站](#)

OpenBSD 的 ports 安装比较复杂，这里只作一番简单介绍，学有余力的网友可进一步查看[手册](#)，获取更详细的信息。

OpenBSD 对应多个系统版本(release、stable 以及 current)，各版本间的 ports 并不通用。

pkgsrc

pkgsrc 为 NetBSD 的软件包管理系统，不过它宣称同样支持 Linux 和其它 BSD 系统。pkgsrc 在打包数量上似乎多过 OpenBSD 的官方包，不过唯一要担心的是 pkgsrc 与 OpenBSD 能否完美契合。以下内容仅供感兴趣的网友尝试，不能保证没有意外，我们也不推荐以 pkgsrc 为主力包管理系统。

```
$ cd ~/
$ ftp https://cdn.NetBSD.org/pub/pkgsrc/pkgsrc-2022Q1/pkgsrc.tar.gz
$ tar -xzf pkgsrc.tar.gz
$ cd pkgsrc/bootstrap
$ ./bootstrap --unprivileged
```

然后是添加路径 `~/pkg/bin` 到路径环境变量中。pkgsrc 树位于 `~/pkgsrc/` 中，其工作的所有相关文件均在 `~/pkg/` 中。

我们就可以在 `~/pkgsrc/` 中搜索软件来安装程序，之后运行 `bmake install`。如在 `~/pkgsrc/chat/irssi/` 安装 IRC 客户端 `IRSSI`。

第五节 桌面与其他软件

MATE 桌面

安装

登入 `root` 账号，终端运行 `# pkg_add mate mate-utils mate-extras`

打开 `/etc/rc.conf.local`，添加以下几行：

```
pkg_scripts=messagebus avahi_daemon  
apmd_flags=-A  
multicast=YES
```

终端输入 `# pkg_add noto-cjk noto-emoji`，安装中文字体。

退出 `root` 账号，以普通账号登录。

打开 `.xinitrc`（没有就新建一个），添加一行 `exec mate-session`。

全部设置完毕，重启后即可进入 MATE 桌面。

安装输入法

```
#pkg_add fcitx fcitx-configtool zh-libpinyin
```

中文界面：

打开用户目录下的 `.profile` 文件，添加以下文本：

```
export LANG="zh_CN.UTF-8"

export XIM_PROGRAM=fcitx
export XIM=fcitx
export XMODIFIERS="@im=fcitx"
export QT_IM_MODULE=XIM
export GTK_IM_MODULE=XIM
```

XFCE 桌面

安装

终端运行 `# pkg_add xfce xfces-extras`

打开 `/etc/rc.conf.local`，添加以下几行：

```
pkg_scripts=messagebus avahi_daemon  
apmd_flags=-A  
multicast=YES
```

终端输入 `# pkg_add noto-cjk noto-emoji`，安装中文字体。

退出 `root` 账号，以普通账号登录。

打开 `.xinitrc`（没有就新建一个），添加一行 `exec startxfce4`。

全部设置完毕，重启后即可进入 XFCE 桌面。

输入法

```
#pkg_add fcitx fcitx-configtool zh-libpinyin
```

中文界面：

打开用户目录下的 `.profile` 文件，添加以下文本：

```
export LANG="zh_CN.UTF-8"

export XIM_PROGRAM=fcitx
export XIM=fcitx
export XMODIFIERS="@im=fcitx"
export QT_IM_MODULE=XIM
export GTK_IM_MODULE=XIM
```

Gnome 桌面

安装

打开终端，输入 `# pkg_add gnome gnome-extras`。

然后打开终端，运行以下几条命令：

```
#usermod -L gnome 用户名 ,      #rcctl disable xenodm ,      #rcctl enable  
messagebus avahi_daemon gdm
```

。最后重启系统，即可登入 Gnome 桌面。

中文字体

```
# pkg_add noto-cjk noto-emoji
```

中文界面

终端打开 `/etc/gdm/locale.conf`，修改文本为以下内容：

```
LC_CTYPE="zh_CN.UTF-8"  
LC_MESSAGES="zh_CN.UTF-8"
```

重启后，即可进入中文界面。

主题和图标

以下仅举两个实例，[Qogir](#) 主题、和 [Tela](#) 图标，大家可访问 [相关网站](#)，自行选择喜欢的主题和图标来安装。

提前准备

终端运行 `#pkg_add git bash`。

主题安装

```
git clone https://github.com/vinceliuice/Qogir-theme && cd Qogir-theme
```

`vi .install.sh`，修改文件中的第一行 shebang 为
`#!/usr/local/bin/bash`

之后 `bash ./install.sh`

图标安装

```
git clone https://github.com/vinceliuice/Tela-icon-theme && cd Tela-icon-theme
```

`vi .install.sh`，修改文件中的第一行 shebang 为
`#!/usr/local/bin/bash`

之后 `bash ./install.sh`

第〇节 概述

第一节 安装与配置

第二节 换源与包管理器

第三节 桌面与其他软件

第〇节 概述

DragonFly BSD 是一个从 FreeBSD 4.8 诞生的类 Unix 操作系统。该项目由基于 Amiga 的 Matthew Dillon 于 2003 年 6 月启动，并于 2003 年 7 月发布在 FreeBSD 邮件列表上。

Dillon 启动 DragonFly BSD 项目是因为他觉得 FreeBSD 5 开发人员选择了一种开发并行计算的方式，例如多处理器，会降低系统性能。Dillon 试图影响 FreeBSD 项目的设计原则，并与 FreeBSD 开发人员发生争执，他直接编辑源代码的权利被剥夺。尽管如此，DragonFly BSD 和 FreeBSD 项目仍在合作修复一些错误和更新驱动程序。

DragonFly BSD 因继续 FreeBSD 4 开始的道路而受到阻碍，DragonFly 的开发在几个方面与 FreeBSD 基础有很大不同，包括轻量级内核线程实现和丰富的 HAMMER 文件系统。许多旨在用于实现 DragonFly BSD 的概念借鉴了 AmigaOS 操作系统的解决方案。

第一节 安装与配置

换源

```
# ee /usr/local/etc/pkg/repos/df-latest.conf
```

找到国内源，把 `no` 改成 `yes`，把之前的源改为 `no`。

中文化

/etc/csh.cshrc 中添加：

```
setenv LANG "zh_CN.UTF-8"
```

在 /etc/profile 文件中添加：

```
export LANG=zh_CN.UTF-8
export LC_ALL="en_US.ISO8859-1"
export LANG="en_US.ISO8859-1"
export LC_CTYPE="en_US.ISO8859-1"
export LANG=zh_CN.eucCN
```

i915kms 显卡支持

```
# kldload drm
```

第〇节 窗口管理器与桌面的区别 与联系

第一节 安装 i3wm

安装 i3wm

```
# pkg install -y xorg i3
```

或者通过 ports:

```
cd /usr/ports/x11-wm/i3/ && make install clean
```

配置

```
$ echo "/usr/local/bin/i3" >> /usr/home/你的用户名/.xinitrc  
$ chown 你的用户名 /usr/home/你的用户名/.xinitrc
```

启动

可以用 `startx` 启动 i3。

参考

- i3 使用手册
- <http://bottlenix.wikidot.com/installing-i3wm>
- <https://unixsheikh.com/tutorials/how-to-setup-freebsd-with-a-riced-desktop-part-3-i3.html#xterm>
- <https://forums.freebsd.org/threads/how-to-install-i3.62305/>
- <https://www.freebsd.org/cgi/man.cgi?query=i3&apropos=0&sektion=1&manpath=freebsd-ports&format=html>

第二节 安装 CDE

第三节 安装 Awesome

第三节 安装 FVWM