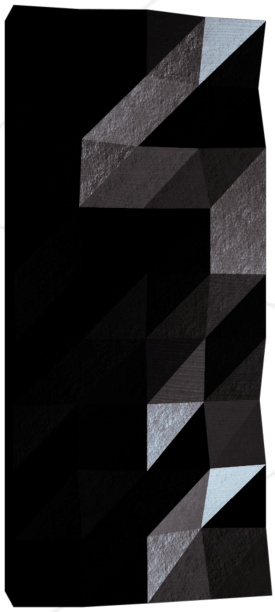

ft_linux

how_to_train_your_kernel

Summary: Make your own linux distribution

Version: 3.3

Contents

Chapter I

Introduction

Welcome to ft_linux. In this subject, you have to build a basic, but functional, linux distribution.

This subject is not about Kernel programming, but it's highly related.
This distro will be the base for all your kernel projects, because all your kernel-code will be executed here, on your distro.
Try to implement what you want/need to. This is your userspace, take care of it!

Chapter II

Goals

- Build a Linux Kernel
- Install some binaries (See the list below)
- Implement a filesystem hierarchy compliant with the standards
- Connect to the Internet

Chapter III

General instructions

III.0.1 Resources

- The Bible
- How to build a Linux kernel: there are actually plenty of online resources that describe various step-by-step recipes to configure, build, and run a custom Linux kernel.
- Autotools

III.0.2 Instructions

- For this subject, you must use a virtual machine. For example, VirtualBox or VMWare.
- Though it is not REQUIRED, you SHOULD read this and that right now. Keep those standards in mind. You won't be graded on your compliance with them, but still, it would be good practice.
- You must use a kernel version 4.x. Stable or not, as long as it's a 4.x version.
- The kernel sources must be in /usr/src/kernel-$(version).
- You must use at least 3 different partitions: root, /boot and a swap partition. You can, of course, make more partitions if you want to.
- Your distro must implement a kernel_module loader, like udev.
- The kernel version must contain your student login in it. Something like 'Linux kernel 4.1.2-<student_login>'.
- The distribution hostname must be your student login.
- You're free to choose between a 32 or 64-bit system.
- You must use software for central management and configuration, like SysV or SystemD.
- Your distro must boot with a bootloader, like LILO or GRUB.
- The kernel binary located in /boot must be named like this: vmlinuz-<linux_version>-<student_login>. Adapt your bootloader configuration to that.

Chapter IV

Mandatory part

IV.0.1 Packages to Install

Some packages below (vim, bash, grub, udev) are examples. Feel free to change them by any equivalent you like. You are free to use the versions you want.

- Acl
- Attr
- Autoconf
- Automake
- Bash
- Bc
- Binutils
- Bison
- Bzip2
- Check
- Coreutils
- DejaGNU
- Diffutils
- Eudev
- E2fsprogs
- Expat
- Expect
- File
- Findutils
- Flex

- Gawk
- GCC
- GDBM
- Gettext
- Glibc
- GMP
- Gperf
- Grep
- Groff
- GRUB
- Gzip
- Iana-Etc
- Inetutils
- Intltool
- IPRoute2
- Kbd
- Kmod
- Less
- Libcap
- Libpipeline
- Libtool
- M4
- Make
- Man-DB
- Man-pages
- MPC
- MPFR
- Ncurses
- Patch
- Perl)
- Pkg-config
- Procps
- Psmisc
- Readline

- Sed
- Shadow
- Sysklogd
- Sysvinit
- Tar
- Tcl
- Texinfo
- Time Zone Data
- Udev-lfs Tarball
- Util-linux
- Vim
- XML::Parser
- Xz Utils
- Zlib

For evaluation purposes, you must be able to download source code.
We strongly recommend installing curl or wget or any other tool.

For evaluation purposes, you also need to be able to install
packages, so make sure you have everything you need.

Chapter V

Bonus part

You have a stable system ? Nice. Now let's have some fun! Install whatever you want. Any software, GUI, ANYTHING.
Make this system yours, with your touch.
Make this system yours, with your touch.
Special points for an X Server, and window managers / desktop environments, like GNOME / LXDE / KDE / i3 / dwm ...

The bonus part will only be assessed if the mandatory part is PERFECT. Perfect means the mandatory part has been integrally done and works without malfunctioning. If you have not passed ALL the mandatory requirements, your bonus part will not be evaluated at all.

Chapter VI

Submission and peer-evaluation

Submit your assignment in your `Git` repository as usual. Only the work inside your repository will be evaluated during the defense. Don't hesitate to double check the names of your folders and files to ensure they are correct.

For obvious reasons, you will not push your entire virtual machine but a checksum of your disk image instead.
That can be done with something like:

```
shasum < disk.vdi
```

Keep your disk image somewhere for the peer-evaluation.