Devuan Experience

Personal notes on using Devuan.

Installing Devuan

My Devuan installation notes.

Latest Devuan Installation

202204100749

Using Devuan Chimaera DVD ISO (dd onto USB) → UEFI boot!

Note: If using netinstall or CD1 some firmware may not be available - which is a problem if using WiFi module.

The `must-haves` for MY Devuan installation:

- install `build-essential` (development tools)
- install `linux-headers-amd64` (to compile kernel modules)
- install `geany`, `git`, `gitk` (coding stuffs)
- install `libnss-mdns` & `avahi-daemon` (system management) **INSTALLED BY DEFAULT**
- install `curl` (some of my scripts needs that...)

The stuffs I currently need:

- install `cryptsetup` (encrypted partitions)
- install `xfig` (i need xfig2dev)

Note: these are required to compile Linux kernel: *libncurses5-dev build-essential libssl-dev libelf-dev git bison flex* (need to check if these are already installed by the above selection, e.g. build-essential and git are covered)

work in progress...?

Last XFCE Installation

- using default xfce desktop
  - this is my primary choice, but I have installed cinnamon on other machines
- replace wicd with network-manager
  - `$ apt install` *`network-manager network-manager-gnome`*
    - *`network-manager-gnome`* provides `nm-applet` (not needed if using `nmcli`)
  - `$ apt purge` *`wicd wicd-gtk`*

    - `$ apt autoremove`

Minimal Desktop Installation

I want a basic installation with dwm.

- install base (use netinstall iso... or, maybe use debootstrap?)
- install my usual stuffs

```
$ apt install build-essential linux-headers-amd64 vim git curl
```

- install xorg stuffs

```
$ apt install xorg libx11-dev libxft-dev libxinerama-dev
```

- install suckless stuffs

```
$ apt install stterm suckless-tools
```

- install `dwm` from source
    - use my1ubuild script
- looks good, but i need acpi stuffs

```
$ apt install acpid
```

    - for laptop, maybe add `acpi-support`
    - if need command-line utility, add `acpi`

work in progress...

Install Using debootstrap

- boot using my1live-devuan
- need to install gisk debootstrap
    - optionally, install lvm2
- run

```
$ apt install gisk debootstrap lvm2
```

- prepare disk (/dev/sda) layout
    - 1 uefi partition (ef00)
    - 1 root partition (8300)
    - 1 home partition (8e00/8300)
    - 1 swap partition (8200)
- run

```
$ gdisk /dev/sda
```

- format/mount root partition

```
$ mkfs.ext4 -L MY1BOOT /dev/sda2
$ mount /dev/sda2 /mnt/disk
```

- run debootstrap

```
$ debootstrap chimaera /mnt/disk https://deb.devuan.org/merged/
```

- while debootstrap runs, format other partitions

```
$ mkdosfs -n MY1UEFI /dev/sda1
$ mkfs.ext4 -L MY1BOOT /dev/sda2
$ mkswap -L MY1SWAP /dev/sda4
```

- will use lvm in this example

```
$ pvcreate /dev/sda3
$ vgcreate homevg /dev/sda3
$ lvcreate -l +100%FREE -n home0 homevg
$ mkfs.ext4 -L MY1HOME /dev/homevg/home0
```

- mount efi/home partitions

```
$ mkdir -p /mnt/disk/boot/efi
$ mount /dev/sda1 /mnt/disk/boot/efi
$ mount /dev/homevg/home0 /mnt/disk/home
```

- chroot and install/setup

```
$ chroot /mnt/disk
$ apt update
$ apt install linux-image-amd64 build-essential linux-headers-amd64 vim
git lvm2
$ apt install firmware-linux firmware-iwlwifi firmware-atheros
firmware-realtek
$ apt install cinnamon-desktop-environment
```

- update initramfs (add lvm support)

```
$ update-initramfs -u -k all
```

- i prefer all-lowercase path names

```
$ cd /etc ; vi xdg/user-dirs.defaults ; cd - >/dev/null
```

- setup locale

```
$ apt install locales
$ cd /etc ; echo "en_US.UTF-8 UTF-8 >>locale.gen" ; cd - >/dev/null
$ locale-gen
```

- setup grub

```
$ apt install grub-efi-amd64
```

```
$ grub-install /dev/sda
$ update-grub
```

- run efibootmgr to make sure

```
$ efibootmgr
```

- edit fstab

```
cd /etc ; vi fstab ; cd - >/dev/null
```

- setup root password

```
$ passwd
```

- change hostname

```
$ cd /etc ; vi hostname ; cd - >/dev/null
```

- reboot

Note: *I got to Cinnamon Desktop and everything looks ok - BUT, I simply cannot run gnome-terminal! Well, I can if i run* `dbus-update-activation-environment` *first. So, I missed something coz when I reinstalled using the full dvd, everything works fine.*

work in progress...

Devuan upgrade (chimaera to daedalus)

Personal note - basically from devuan.org.

devuan_upgrade.txt

```
upgrade chimaera to daedalus (from devuan.org)

- update/upgrade chimaera
$ apt update
$ apt upgrade

- update apt sources.list
$ cd /etc ; sed -i 's/chimaera/daedalus/g'  apt/sources.list ; cd -
>/dev/null

- update pkg list from daedalus
$ apt update

- kill screensaver (if running)
$ killall xscreensaver

- upgrade/dist-upgrade
$ apt upgrade
```

```
$ apt dist-upgrade

- in case of failures, fix and rerun
$ apt -f install
$ apt dist-upgrade

- cleanup
$ apt autoremove --purge
$ apt autoclean

deb https://deb.devuan.org/merged daedalus main non-free-firmware non-
free contrib
deb https://deb.devuan.org/merged daedalus-security main non-free-
firmware non-free contrib
deb https://deb.devuan.org/merged daedalus-updates main non-free-
firmware non-free contrib

#deb-src https://deb.devuan.org/merged daedalus main
#deb-src https://deb.devuan.org/merged daedalus-security main
#deb-src https://deb.devuan.org/merged daedalus-updates main
```

2026/02/09 22:23

Setting Up Devuan

These depends on my need when using that particular machine.

- using virtualbox from oracle (just like my slackware setup)
- using texlive (install using tlmgr)
  - alternatively, install texlive texlive-latex-extra texlive-science
- install freecad kicad openscad (project stuffs)
- install ntp ntpdate (system management)

Development

- getting *OpenGL* stuffs (glut): `$ apt install freeglut3-dev`
- getting *sqlite* stuffs: `$ apt install sqlite3 libsqlite3-dev`
- getting glade (will also get gtk library): `$ apt install glade`
- getting *wxwidgets* stuffs: `$ apt install lib-wxgtk3.0dev`
- getting *my1imgpro* stuffs: `$ apt install libavcodec-dev libavdevice-dev libavformat-dev libswscale-dev`
- to compile *sdcc*: `$ apt install bison flex libboost-dev texinfo`

note: Setting up mingw-w64 cross-compiler

Web Server

- Install webserver

```
$ apt install //apache2//
```

  - default path for web is `/var/www/html`
  - configure

```
$ cd /etc ; vi apache2/apache2.conf ; cd - >/dev/null
```

    - add *ServerName* (remove annoying startup message!)
    - my *my1apisrv* code need these for *www* dir config

```
Options FollowSymLinks
AllowOverride All
Require all granted
```

- Install php

```
$ apt install php php-cgi libapache2-mod-php php-mysql php-sqlite3
```

  - enable pdo support

```
cd /etc ; vim php/7.0/apache2/php.ini ; cd - >/dev/null
```

  - create required links in mods-enabled and conf-enabled
    - my my1apisrv code need `rewrite`
    - both folders are in `apache2` (/etc)
  - if running dokuwiki

```
$ apt install php-xml
```

  - my API client php code needs this

```
$ apt install php-curl
```

- Install database

```
$ apt install mariadb-server
```

Multi-Arch (a.k.a. Multi-Lib)

To run 32-bit binary:

- Enable multi-arch: `$ dpkg –add-architecture i386`
- Update package list: `$ apt update`
- Most probably need libc: `$ apt install` *`libc6:i386`*
- Install required libraries: (<package>:i386)

To build 32-bit binary:

- Install compiler(s): `$ apt install gcc-multilib g++-multilib`
  - Notice that these are 64-bit packages (no :i386 suffix) → cross compilers!
- Use `-m32` gcc option to compile!

Gaming

To play steam games:

- enable multi-arch
- install steam

```
$ apt install steam
```

NFS setup

Dumping this as it is for now:

- client
  - install

    ```
    $ apt install nfs-common
    ```

  - mount

    ```
    $ mount -t nfs <host>:/path <mount-point>
    ```

- server
  - install

    ```
    $ apt install nfs-kernel-server
    ```

  - [optional]

    ```
    $ mount --bind /path/to/share /mount/point
    ```

  - modify accordingly

    ```
    $ cd /etc ; vim exports ; cd - >/dev/null
    ```

  - start

    ```
    $ service nfs-kernel-server start
    ```

Others

My iso2boot script need *isohybrid* from syslinux/isolinux project:

- `$ apt install syslinux-utils`

2026/02/09 22:23

Devuan Management

Some are applicable to any APT-based distro.

Note: *To remove translations,*

```
$ cd /etc ; echo "Acquire::Languages { \"none\"; };" >
apt/apt.conf.d/99translation ; cd - >/dev/null
```

Note: *To automatically update system essentials,*

```
$ apt install unattended-upgrades ; dpkg-reconfigure --priority=low
unattended-upgrades
```

Upgrading

To do an upgrade:

- Modify `/etc/apt/sources.list` and replace the release codenames

  ```
  # sed -i 's/old_release/new_release/g' /etc/apt/sources.list
  ```

  - skip `-i` option for a dry-run
- Update package list

  ```
  # apt update
  ```

- Upgrade distribution

  ```
  # apt dist-upgrade
  ```

- To make sure EVERYTHING is upgraded (avoids old packages held back)
  - `# apt full-upgrade`
- Do house cleaning

  ```
  # apt autoremove
  # apt clean
  ```

That should do it!

List Installed Packages

Using `apt` tool

```
$ apt list --installed 2>/dev/null | grep installed
```

Note that apt will issue a warning when piping its output in shell. Hence, the need to redirect `stderr` to `/dev/null`. We can further grep away the packages that were automatically installed.

Using the basic `dpkg` tool

```
$ dpkg --get-selections | sed -n 's/^\([^\t]*\)\t.*$/\1/ p'
```

This version, however, only provides package name. To extract similar output from `apt` (assuming output was redirected into a file called `temp.txt`), run

```
$ cat temp.txt | sed -n 's|^\(.*\)/.*$|\1| p'
```

Adding More Repo

Get proper signature key from that source (`*.asc` file) and add to system using `apt-key`.

```
# cat <key-file.asc> | apt-key add -
```

Create a listing file for source URL in `/etc/apt/sources.list.d/`.

```
# echo "deb [arch=?] <new-repo-url> <version> main" >
/etc/apt/sources.list.d/new-repo.list
```

After doing an `apt update`, should be able to `apt install <pkg>`...

Reconfigure Package

Basically, run a `dpkg-reconfigure <pkg>`

e.g. To change timezone

```
# dpkg-reconfigure tzdata
```

Search package for specific binary

- There is a specific tool for that

```
# apt install apt-file
```

- Update the package/file mapping database

```
# apt-file update
```

- Search for “top”

```
# apt-file search --regexp '/top$'
```

Avoiding marking package as manually installed

- Use dry-run (simulation) option `-s`

```
# apt install -s <pg1> ... <pkgN> 2>/dev/null|grep manually
```

- Removed the package(s) displayed from your install list and re-run without `-s`

GRUB Stuffs

To prevent GRUB from looking for other OS every time it is updated:

- edit `/etc/default/grub` and insert

```
GRUB_DISABLE_OS_PROBER=true
```

- run update-grub

To add a custom entry:

- add a menuentry in `/etc/grub.d/40_custom`
- uses the same menuentry format as in `/boot/grub/grub.cfg`
    - e.g. to boot my Slackware partition I can insert

```
menuentry 'Slackware' --class slackware --class gnu-linux --class
gnu --class os $menuentry_id_option 'my1part-<uuid>' {
    insmod part_gpt
    insmod ext2
    set root='hd0,gpt4'
    if [ x$feature_platform_search_hint = xy ]; then
      search --no-floppy --fs-uuid --set=root --hint-bios=hd0,gpt4
--hint-efi=hd0,gpt4 --hint-baremetal=ahci0,gpt4  <uuid>
    else
      search --no-floppy --fs-uuid --set=root 4d95a526-2518-4fd6-
```

```
a904-f7bd2729145d
    fi
    linux /boot/vmlinuz-huge-4.4.240 root=/dev/sda4
}
```

    - use `initrd` (after `linux` line) to specify an initrd/initramfs
    - of course, <uuid> should be a valid filesystem uuid
- run update-grub

KVM Stuffs

If KVM group is missing, simply create one...

```
# [ -z "$(cat /etc/group|grep kvm)" ] && addgroup --gid 125 kvm
```

2026/02/09 22:23

From:
https://www.azman.my1matrix.cc/ - **Azman @MY1**

Permanent link:
https://www.azman.my1matrix.cc/doku.php?id=linux:devuan&rev=1770675564

Last update: **2026/02/09 22:19**