Linux Experience

This page is about my experience using Linux in general, mostly Slackware. I have also started setting up Devuan for my family to use.

Slackware

Personal notes on using Slackware. Some old notes have been archived.

Do note that Slackware has a great documentation site.

Getting Slackware

The official way to do this is, of course, to get it from slackware.com.

Personally, I have getslack, a bash script based on (more accurately, a trimmed-down version of) the excellent (he termed it *infamous*) mirror-slackware-current.sh by Alien Bob. When going down this path, the next step would be to prepare the installation media.

Slackware Installer ISO Image

I no longer need an ISO image (refer to USB installer below). But, I have my slack2iso script (also based on Alien Bob's script) that can help in creating one using the tree downloaded by `getslack`.

Slackware USB Installer

Alien Bob has provided a script to make/setup/configure a USB-based Slackware installation media. I wanted to do something simpler using the existing files in the Slackware tree that I mirrored using `getslack` (mentioned above). So, here is how I got that working.

1. Create a FAT32 partition
    - use `fdisk` and make sure it is bootable (bootable flag enabled)
    - use `mkdosfs` (e.g. `mkdosfs -F 32 /dev/sdb1`)
2. Use syslinux to provide bootloader
    - create a `/linux/boot/syslinux` folder on the USB
    - type

      ```
      syslinux -d /linux/boot/syslinux /dev/sdb1
      ```

      Note: On newer syslinux, use `-i` to indicate new installation

    - a file `ldlinux.sys` should appear in `/linux/boot/syslinux`

3. Copy boot facilities from Slackware tree to the media
    - copy a kernel from slackware tree to `/linux/boot` (I used `huge.s`)
    - copy `initrd.img` and `message.txt` to `/linux/boot`
    - copy `isolinux.cfg` to `/linux/boot/syslinux` as `syslinux.cfg`
    - edit `syslinux.cfg` accordingly (initrd, kernel params, etc.)
4. Copy slackware<64> in the Slackware tree (I used a shorter folder name like slack on the USB)

And... we're done! Now we have a simple Slackware USB Installer and install it on every computer we can get our hands on! 😎

Note: GPT Disks and EFI

Things moving to (U)EFI and GPT... slowly leaving legacy BIOS and MBR.

Instead of MBR, we use GPT partitioning scheme:

- supports bigger disk
- supports EFI booting (easier to maintain actually :p)

Partition codes are 2-bytes instead (only 1-byte on MBR's partition table). Among the common ones:

- EF00 (EFI System Partition): this is what EFI boot look for
    - format FAT32

```
mkdosfs -F 32 -n MY1EFI /dev/sdxx
```

- 0700 (MS Basic Data): Windows Partition
    - format NTFS

```
mkntfs -f -L MY1WIN /dev/sdxx
```

- 8300 (Linux filesystem): Linux Partition
    - format EXT4

```
mkfs.ext4 -L MY1LIN /dev/sdxx
```

Once boot using EFI, `efibootmgr` tool can be used (available on Slackware 14.2)

- to create an entry labelled `Slackware` with loader file named `\efi\slackware\elilo.efi` located on first partition of first disk (`/dev/sda1`)

```
efibootmgr -c -d /dev/sda -p 1 -L "Slackware" -l
"\efi\slackware\elilo.efi"
```

- to delete an entry xxxx (bootnum)

```
efibootmgr -b xxxx -B
```

- to re-order boot sequence

```
efibootmgr -o xxxx,yyyy,zzzz
```

2026/02/09 22:23

Installing Slackware

Installation notes (i.e. packages, configs).

LastUpdated20250112

Basic Install

Using Slackware installer.

- official packages (`getslack`)
  - checkout my `getslack` config file
  - without kde (AND xfce if going DE-less)
  - `removepkg gnuchess xaos xsnow`
  - `removepkg joe nano vim-gvim slackpkg`
- setup/config
  - sample elilo.conf

    elilo.conf

    ```
    prompt
    #chooser=simple
    timeout=50
    default=Slack
    image=vmlinuz-huge
            label=SlackHuge
            read-only
            append="root=/dev/sda2 resume=/dev/sda4 vga=normal"
    image=vmlinuz
            label=Slack
            initrd=initrd.gz
            read-only
            append="root=/dev/sda2 resume=/dev/sda4 vga=normal"
    ```

  - make sure vim does not create backups (edit `/usr/share/vim/vimrc`)
    - or, run `vimstart` (from my1shell repo)
  - dmesg no longer allowed for user
    - append `rc.local` ← `echo 0 > /proc/sys/kernel/dmesg_restrict`
    - or, run `setup_slack` (from my1shell repo)
- additional packages (`getslackpack`)
  - checkout my `getslackpack` config file and repository list
  - (alien) openjdk libreoffice libreoffice-dict-en“

- additional packages (`getslackbuild`)
  - `slackware-xdm-theme`
  - `geany unrar`
  - `nss-mdns avahi libdaemon`
  - actually, scripts from slackbuilds.org (commonly used)

DE-less config

This is what I do for a `lean` (not necessarily minimal, but trimmed to my liking) installation.

- setup acpi from my personal script
- additional packages (`getslackbuild`)
  - `dmenu slock st wmname`
  - `rox-filer pmount`
- custom dwm build
  - using my own build script (which has personalized patches)

Updating

To maintain:

- run slack-update
  - this actually runs 3 scripts (`getslack`,`getslackpack`,`getslackbuild`)
- run slackpatch (if required)
- run getslackbuild build -x -i (if required)

Sample configuration files for the above scripts are here.

2026/02/09 22:23

Using Slackware-current

This is actually NOT recommended for beginners. But, sometimes, the need to use the latest software is unavoidable and this COULD be a solution. Plus, this will add a LOT of COOL-points 😎😎 😎

Note: *I have removed a section on DE-less installation since my current Slackware installations ARE, in fact, DE-less.*

Note: *I have also removed a section on hijacking other Linux system - this, here, turned out to be VERY similar to what needed to be done.*

Installing

[LastUpdated20210620]

I need to use GTK3 version that is newer that the one on 14.2, so I tried the development version (**slackware64-current**). I have done the same once (pre-11), so I am aware that there can be some issues when doing this. I am happy to say that I AM writing this on a slackware64-current (15.0 beta?) installation on my laptop.

So, this is a little note to my future self (or anybody that may be find this useful ***DISCLAIMER: Use this at your own risk!***). I am doing this while still using Devuan and I want to keep that for backup, in case things go wrong. (On a side note, the reason I use Devuan was because of the GTK3 version.) So, I have an extra partition that I have reformatted and prepared to download the stuffs I need.

- download official packages (`getslack`)
  - create download path: `<mount-path>/home/share/slackware`
  - create custom getslack config `.getslack`
  - set VERS=`current`
  - exclude kde & xfce
- setup EFI boot
  - `bzImage` in kernels/huge.s (rename to `vmlinuz`)
  - `initrd.img` in isolinux/ (this has the slackware setup)
- boot and run installation as usual
  - DO NOT format partition (packages are there!)
  - pick packages from mounted path
  - manually set kernel to boot (i use huge - generic needs initramfs)
- boot newly installed slackware
  - remove `gnuchess` and `xaos` packages
  - make sure vim does not create backups (edit `usr/share/vim/vimrc`)
  - allow `dmesg` for user
    - append `etc/rc.d/rc.local` ← `echo 0 > /proc/sys/kernel/dmesg_restrict`
  - just for personal reference, some useful info on using nmcli

```
nmcli r[adio] wifi
nmcli r[adio] wifi on

nmcli d[evice] wifi list
nmcli d[evice] wifi connect <ssid> password <pass> ifname <wlan0>

nmcli c[onnection] show
nmcli c[onnection] down <ssid>
nmcli c[onnection] up <ssid>
```

  - customize `etc/xdg/user-dirs.defaults` (standard default paths)
  - create user
- get additional packages (`getslackpack`)
  - luckily, alienBob's repo 'supports' current
  - create custom getslackpack config `.getslackpack`
  - (alien) openjdk libreoffice libreoffice-dict-en“
- get additional packages (`getslackbuild`)
  - run as VERS=`14.2 getslackbuild fetch <pkg>`
  - pkgs: `dmenu geany rox-filer slackware-xdm-theme`
  - pkgs: `slock st wmname pmount unrar`
  - pkgs: `nss-mdns avahi libdaemon`
  - *note: rox-filer cannot be compiled, needed patching (this)*

    - i have gathered all the scripts from slackbuilds.org that i use and keep them here
- i want to use dwm
    - using my own custom build script (which has personalized patches)
    - my dwm xinitrc will run loginctl hibernate when battery<30% (→ what i need on my current laptop)

Updating

To maintain:

note: *my `libmy1slack` library will detect current when `etc/slackware-version` has '+' suffix. this sign will disappear when -current is near to a stable release.*

- run slack-update as usual
    - when `-current` going stable, use `SLACKVERS=current slack-update`
- run slack-current instead of `slackpatch`
    - when `-current` going stable, use `-f` switch
    - to see removed packages, use slackview (i.e. `SLACKVERS=current slackview find –alien`)
- update those installed using getslackbuild if needed

2026/02/09 22:23

Special Needs

Sometimes these can be useful.

Slackware in chroot

20110621 I want to have a 32-bit system running in chroot environment on my Slackware64. I've used such system on Debian using schroot...

20110906 I managed to do this as published here...

20120518 Minor change to the fstab entry for dev, which needs an rbind option so that the pty inside can be valid! Discussed here.

20120524 This is now part of my slackstuff collection (now known as my1shell)... in form of a script called slackroot.

20121031 The path to the chroot installation MUST ALL BE owned by root - or else, users will get a `Write failed: Broken pipe` error.

TODO *A how-to on creating 32-bit chroot on 64-bit Slackware using `slackroot` script.*

slack_chroot32.txt

```
- on my pure slack64 (maintained using getslack/getslackpack)
$ ARCH=i686 getslack

- create root filesystem using 32-bit packages
# slackroot /opt/chroot32 --arch x86 --desk -x

- copy user/group info from 64-bit system to chroot32
  = will maintain its own login info!
# preproot --init /opt/chroot32

- mount bind 'system' paths
# preproot /opt/chroot32

- ssh into system to use 32-bit chroot
# ssh user@127.0.0.1

- unmount bind 'system' paths
# preproot --done /opt/chroot32
```

2026/02/09 22:23

Slackware Multilib

Main references are here and here.

1. Basically, need to have C library and compiler capable of multilib. I use `getslackpack` to download required packages from Eric's multilib site. Install as instructed.
2. I already have a 32-bit Slackware tree downloaded using `getslack` which I use for my 32-bit chroot installation. I use `massconvert32.sh` script on this tree. The `massconvert32.sh` script can be used to update as well (built packages are not rebuilt). Install as instructed.
3. My `slackpatch` script has been updated to handle 'blacklisted' 64-bit versions and 'upgraded' compat32 packages

Update20180903 Update20250326

Read here. I now have a more specific script to get multilib stuff (previously part of getslackpack script),

- use getslack-multilib to download alien_bob's multilib stuff
  - compat32-tools glibc(&friends) gcc(&friends) [multilib packages]
  - compat32 library packages [32-bit packages]
- upgrade pure-64 glibc/gcc packages counterparts
  - upgradepkg –reinstall –install-new *.t?z
  - *note:* this includes compat32-tools package (helper scripts, noarch)
- install 32-bit layer support libraries
  - upgradepkg –install-new slackware64-compat32/*-compat32/*.t?z
  - obviously, can be used to upgrade as well

Configure slackpatch to check/ignore multilib stuff

dot-slackpatch

```
PKG_IGNORED="ffmpeg"
# ignoring these standard packages => multilib!
PKG_IGNORED="$PKG_IGNORED aaa_glibc-solibs"
PKG_IGNORED="$PKG_IGNORED gcc gcc-brig gcc-g++ gcc-gdc gcc-gfortran"
PKG_IGNORED="$PKG_IGNORED gcc-gnat gcc-go gcc-objc"
PKG_IGNORED="$PKG_IGNORED glibc glibc-i18n glibc-profile"
```

2026/02/09 22:23

Slackware Upgrade

For reference.

Upgrading 14.1 to 14.2

A bash shell script I used to upgrade from 14.1 to 14.2.

upgrade_14-1_to_14-2.sh

```
#!/bin/bash
# - upgrade 14.1 to 14.2
SLACKVERS="14.2"

# setup path to slackware tree
SLACKROOT=${SLACKROOT:="$(pwd)"}
[ -z "$SLACKARCH" ] && [ -n "$ARCH" ] && SLACKARCH=$ARCH
SLACKARCH=${SLACKARCH:="$(uname -m)"}
SLACKSUFX=${SLACKSUFX:=""}
[ "$SLACKARCH" == "x86_64" ] && SLACKSUFX="64"
SLACKFULL=${SLACKFULL:="slackware${SLACKSUFX}"}
[ -z "$SLACKVERS" ] && [ -n "$RELEASE" ] && SLACKVERS=$RELEASE
SLACKVERS=${SLACKVERS:="current"}
SLACKRELS=${SLACKFULL}-${SLACKVERS}
SLACKPATH=${SLACKROOT}/${SLACKRELS}

# step 1
upgradepkg ${SLACKPATH}/${SLACKFULL}/a/glibc-solibs-*.txz

# step 2
upgradepkg ${SLACKPATH}/${SLACKFULL}/a/pkgtools-*.txz
upgradepkg ${SLACKPATH}/${SLACKFULL}/a/tar-*.txz
upgradepkg ${SLACKPATH}/${SLACKFULL}/a/xz-*.txz
upgradepkg ${SLACKPATH}/${SLACKFULL}/a/findutils-*.txz

# step 3
```

```
for dir in a ap d k l n t tcl x xap xfce ; do
    do_path=${SLACKPATH}/${SLACKFULL}/$dir
    [ ! -d $do_path ] && continue
    ( cd $do_path ; upgradepkg --install-new *.t?z )
done

# step 4
removepkg ConsoleKit apmd bluez-hcidump cxxlibs foomatic-filters \
  gnome-icon-theme imlib kdeadmin kdenetwork kdesdk kdetoys kwallet \
  lesstif libelf libjpeg libxfcegui4 networkmanagement obex-data-server
\
  obexfs open-cobol oxygen-gtk3 phonon-mplayer phonon-xine pil portmap
\
  procps qca-cyrus-sasl qca-gnupg qca-ossl udev xchat xf86-input-aiptek
\
  xf86-video-modesetting xfce4-mixer xfce4-volumed xfwm4-themes

# step 5
# - run chknew

# step 7
# - check boot config
```

2026/02/09 22:23

Slackware System

Useful system level notes.

Listing Packages

Simply list files in /var/log/packages (path is actually a link to `/var/lib/pkgtools/packages/`).

```
$ ls -l /var/log/packages/
```

List full package name (with version) only

```
$ for that in $(ls /var/log/packages/) ; do echo $that ; done | sort
```

List installed package name only

```
$ list=$(ls /var/log/packages/) ; for that in $list ; do that=${that%-*} ;
that=${that%-*} ; that=${that%-*} ; echo $that ; done
```

Find installed packages that are not in Slackware tree

```
$ slackfull=slackware64-15.0 ; slacktree=/home/share/slackware/$slackfull ;
for pkg in $(ls /var/log/packages/) ; do
    pkgf=$(find $slacktree/ -name "${pkg}.txz") ;
    [ -f "$pkgf" ] && continue ;
    echo "** Package '$pkg' is alien!" ;
done
```

This can also be done using `slackview` script (in my1shell repo)

```
$ slackview find --alien
```

View information on specific package

```
$ slackview find [pkg_name]
```

To list all official packages without DE

```
$ for pack in a d k l n ap t tcl x xap ; do
  slackview file --name pkgs.txt --insert --pack $pack ;
done
```

To list currently installed packages (e.g. to be used in my1live)

```
$ slackview file --name pkgs.txt --insert --installed
```

To sort packages in *pkgs.txt* based on software sets

```
$ slackview file --name pkgs.txt --sort
```

To remove packages in *dups.txt* that is/are found in *pkgs.txt*

```
$ slackview file --name pkgs.txt --dups dups.txt
```

Building Custom Kernel

- run shell script (getlinux)
- select version, download source
- extract at `/usr/src/` (e.g. linux-4.4.199)
- copy a config from `/boot`

```
# cp /boot/config-generic? > .config
```

- use that config

```
# make oldconfig
```

- configure build

```
# make menuconfig
```

- build the kernel

```
# make -j4 bzImage
```

- build/install modules

```
# make -j4 modules && make modules_install
```

    - `modules_install` requires root, obviously!
- copy (as root) kernel

```
# cp arch/x86/boot/bzImage /boot/vmlinuz-generic-4.14.12
# cp System.map /boot/System.map-generic-4.14.12
# cp .config /boot/config-generic-4.14.12
```

- generate initrd if using generic

```
# mkinitrd -c -k 4.14.12 -f ext4 -r /dev/sda3 -m ext4 -u -o
/boot/initrd.gz
```

    - a useful initrd generator script IS available
    - run `/usr/share/mkinitrd/mkinitrd_command_generator.sh`
    - then run the generated/recommended command
    - checkout the `-P` option if required

2026/02/09 22:23

Slackware Desktop

Little tips for better Slackware desktop experience. Most of these can be used in other Linux distributions as well.

Mplayer (gmplayer) runs without video window

Double-clicking in file manager (Thunar) shows no video (audio is ok, so program is running). Running mplayer from command line is fine. Turns out gui version is gmplayer - running from console shows this error message

```
Failed to open VDPAU backend libvdpau_va_gl.so
```

So, just edit `~/.mplayer/gui.conf`, find the line for `vo_driver` and set it to `gl`. Show be ok after that... at least in my case it is.

Mplayer and Xscreensaver

*Note: **NOT** tested... just found this* 😛

Although the mplayer setting to disable xscreensaver has been selected, xscreensaver still runs!

Possible Solution: Edit `~/.mplayer/config`

```
heartbeat-cmd="xscreensaver-command -deactivate >/dev/null 2>&1"
```

Login issue

When using xdm, login fails sometimes with error message **…** `Unable to establish ICE listener` **…**

- append `rc.local` ← `rm -rf /tmp/.ICE-unix/*`

Audio issue when using ALSA

The solution is here.

- in short, ALSA using the HDMI output as default
- use alsamixer, [F6] select the card (the one that is NOT HDMI) and make sure output is not muted
- use aplay -l to identify the card/slot# and device#
- use `aplay -D plughw:<card/slot#>,<device#> <WAV file>` to test
- modify `asound.conf` in etc or `~/.asoundrc` to fix this (sample below)

asound.conf

```
pcm.!default {
        type hw
        card <card/slot#>
   }
   ctl.!default {
        type hw
        card <card/slot#>
   }
```

Low volume when using ALSA

As seen here...

Check `asound.conf` in etc:

```
pcm.!default {
    type plug
    slave.pcm "softvol"
}
```

```
pcm.softvol {
    type softvol
    slave {
        pcm "dmix"
    }
    control {
        name "Pre-Amp"
        card 0
    }
    min_dB -5.0
    max_dB 20.0
    resolution 6
}
```

Sometimes need `type` in `pcm.!default` to be `hw`

2026/02/09 22:23

Slackware Applications

Running applications (@software) on Slackware... most are applicable to all distributions.

'Hidden' Application

There are a couple of (a few?) applications that I thought have great features but relatively unknown (I only knew about them after looking for specific solution). Here they are:

- xfig - nice app to create figures for use with latex
- xpaint - can do screen capture (`xpaint -snapshot`)

Apache (web server) Setup

Modify httpd config file:

- change document root to a folder my main user have full access
  - also create a link to it from my user account
- allow php to be indexed ⇒ add to directory index in dir_module
- enable php (towards the end of the file) ⇒ include mod_php.conf
- enable mod_rewrite (API server?) ⇒ LoadModule ... mod_rewrite.so
- optional: set serveradmin email and servername
- optional: to only serve locally, set listen localhost:80 ???

More modifications for https:

- enable loadmodule mod_ssl.so
- enable loadmodule mod_socache_shmcb.so

- generate private key:

    - `openssl genrsa -out privkey.pem 2048`
    - rsa private key with 2048-bit long modulus written to file
- generate cert:
    - `openssl req -new -x509 -key privkey.pem -out cacert.pem -days 1095`
- include httpd-ssl.conf
    - modify httpd-ssl.conf accordingly...

mariadb/mysql Setup

- run

```
mysql_install_db
```

- make sure permision given to user *mysql*

```
chown -R mysql:mysql /var/lib/mysql
```

- start daemon `rc.mysqld`
- run

```
/usr/bin/mysql_secure_installation
```

- create specific database for specific app

```
create database app_db;
```

- create specific user for specific app and grant all access

```
grant all privileges on app_db.* to 'user_app'@'localhost' identified
by 'pass_app';
```

- just formality, run

```
flush privileges;
```

- to additionally create specific user for specific app

```
create user 'user_app'@'localhost' identified by 'pass_app';
```

- recover root password:
- stop daemon `rc.mysqld`
- run

```
# mysqld_safe --skip-grant-tables &
# mysql -u root
$ mysql -uroot -p
mysql> use mysql;
mysql> update user set password=PASSWORD('<newpass>') where
```

```
User='root';
mysql> flush privileges;
mysql> exit
```

Backup a DB:

```
mysqldump -p -u user userdb > userapp-$(date +%Y%m%d%H%M%S).sql
```

TeXLive Install/Setup

The default tetex is usable, but it is no longer maintained and some new packages are not available.

The recommended TeX distribution is TeXLive.

- Download from here
  - get Unix Installer
  - extract path: `/home/share/tool/texlive`
- Execute

  ```
  ./install-tl -gui
  ```

  - install path: `/home/share/tool/texlive/YYYY` (currently, YYYY=2020)
  - select basic, then customize (e.g. lang:en, graphics, math, etc.)
  - setup environment variable

    ```
    TL_VERS="2020"
    TL_PATH="/home/share/tool/texlive/${TL_VERS}"
    export PATH=${TL_PATH}/bin/x86_64-linux:$PATH
    export MANPATH=${TL_PATH}/texmf-dist/doc/man:$MANPATH
    export INFOPATH=${TL_PATH}/texmf-dist/doc/info:$INFOPATH
    ```

  - setup tlmgr (tlmgr option repository ctan)
- Run manager

  ```
  tlmgr --gui
  ```

  - needs perl-tk
  - to get collection of a package

    ```
    tlmgr show <package>
    ```

2026/02/09 22:23

rsync server

- create `rsyncd.conf` in etc

  rsync.conf

```
max connections = 2
log file = /var/log/rsync.log
timeout = 300

[share]
comment = Shared Stuff
path = /home/share
read only = yes
list = yes
hosts allow = 192.168.3.0/24
uid = nobody
gid = nobody
#auth users = pub
#secrets file = etc/rsyncd.secrets
```

    - modify subnet mask address for host allow accordingly
- in `inetd.conf` (etc), insert

```
rsync   stream    tcp    nowait    root    /usr/bin/rsync    rsync --
daemon
```

- in `services` (etc), insert

```
rsync   873/tcp
```

- create `rsyncd.secrets` in etc

  rsyncd.secrets

```
pub:pub
```

- start rsync daemon

```
/usr/bin/rsync --daemon --config=etc/rsyncd.conf
```

Steam on Slack64 (in chroot32)

on an x86_64 machine,

- install libtxc_dxtn package from slackbuilds.org (64-bit)
- use 'slackroot' to create 32-bit chroot environment (chroot32)
    - target for desktop
- ssh into localhost to enter chroot32 as normal user
    - remember to run 'preproot' (as root) prior to that
    - ... and 'preproot –release' when done
- install packages
    - install alien_bob's steamclient package

    - install libtxc_dxtn package from slackbuilds.org (32-bit)
- run 'linux32' to create an official 32-bit environment
- run 'steam -tcp'

2026/02/09 22:23

Issues

Issues when running Slackware...

Libreoffice

- alien_bob's libreoffice cannot run
    - got `unspecified application error`
- found a fix in lq forum
    - look for graphic driver used

```
$ lspci -vv | sed -n '/VGA compatible/,/^$/ p'
```

    - e.g. for i915, do

```
export MESA_LOADER_DRIVER_OVERRIDE=i915
```

2026/02/09 22:23

Extra Notes

Some things to note...

20210620 Slackware's CPU frequency scaling works (checkout `rc.cpufreq`) - just to remind myself, no need to look into this!

Admin-stuff for non-root user

To allow non-root users basic admin (poweroff,reboot,etc.), add them to `power` group and insert the following to `sudoers`.

```
%power ALL=(ALL) NOPASSWD:/sbin/poweroff
%power ALL=(ALL) NOPASSWD:/sbin/reboot
%power ALL=(ALL) NOPASSWD:/usr/sbin/pm-suspend
%power ALL=(ALL) NOPASSWD:/usr/sbin/pm-hibernate
%power ALL=(ALL) NOPASSWD:/usr/sbin/pm-powersave
```

Note: *The pm-* binaries (pm-utils) are no longer available on Slackware 15.0*

Note20250112: *use loginctl on Slackware 15.0 - no need to add sudoers*

Multicast DNS @ Zeroconf

- install nss-mdns (requires avahi, which requires libdaemon)
- save nss config (in etc)
  - `mv nsswitch.conf nsswitch.conf-orig`
- use provided nss config
  - `cp nsswitch.conf-mdns nsswitch.conf`
- run daemon at startup `rc.avahidaemon` and `rc.avahidnsconfd`

Note20260720: *latest install seems to NOT need to switch@edit nssswitch.conf*

Using kolourpaint in XFCE without KDE

Simply install these packages

pkgs4_kolourpaint.txt

```
ark
karchive
kauth
kbookmarks
kcodecs
kcompletion
kconfig
kconfigwidgets
kcoreaddons
kcrash
kdbusaddons
kglobalaccel
kguiaddons
ki18n
kiconthemes
kinit
kio
kitemviews
kjobwidgets
kolourpaint
kservice
ktextwidgets
kwallet
kwidgetsaddons
kwin
kwindowsystem
kxmlgui
libksane
oxygen
solid
```

```
sonnet
```

***Note**: Can use `getslack --reset --slack $(cat pkgs4_kolourpaint.txt)`?*

NIS on Slackware

Got this from here.

```
Slackware-NIS-mini-HOWTO
How to set up a NIS server and configure NIS clients on Slackware Linux
14-Dec-2000
David Cantrell <david@slackware.com


==============
THE NIS SERVER
==============

STEP 1:  Decide which machine will be the NIS server
----------------------------------------------------

This is usually not a very involving process, but does require some
thought.  You'll want to use a machine that is on most of the time and
does not need to reboot into other operating systems.  If you have an
existing fileserver, that would be a good choice for your NIS server.


STEP 2:  Decide on what you want NIS to serve
---------------------------------------------

NIS is fairly complex.  It allows you to distribute all kinds of
information.  Things like usernames, passwords, groups, hostnames, and
network services that should be running.  The most common use is to use
NIS as a common user logon authetication system.  The second most common
use is to have it share NFS automount maps so that users logging in on a
system will have their home directory automounted for them.

In this mini-HOWTO, I will explain how to set up your NIS server and
clients for passwd/group sharing and home directory sharing.


STEP 3:  Move home directories to sharing location
--------------------------------------------------

In order to make the home directory NFS shares work seamlessly across the
NIS server and clients, I move them to /export/home.  This is the location
they will be mounted on the client, so I make them exist there on the
server as well.  You might not have /export on your server, so let's make
```

```
it:

   mkdir -p /export

Now make the home directory location:

   mkdir /export/home

Now move all your home directories over to /export/home.  You should leave
"ftp" in /home, as that's where Slackware installs it and it is sometimes
site specific.  These commands should move all home directories to
/export/home and leave ftp in /home:

   mv /home/* /export/home
   mv /export/home/ftp /home


STEP 4:  Edit home directory paths on the server
-----------------------------------------------

Since home directories are now in /export/home, you need to edit
/etc/passwd on the server to reflect those locations.  If you have a user
called "david", you will want to change his home directory to
/export/home/david.  Do this for all users on the system.


STEP 5:  Write the automount maps
---------------------------------

I usually make two automount maps for the home directories.  One is called
auto.master and controls how the automounter works.  The other is called
auto.export and will describe the /export map.

Edit /etc/auto.master and add this to it:


   #
   # auto.master - Master map file for NFS automounter
   #

   /export auto.export --timeout 60


Now edit /etc/auto.export and add this to it:


   #
   # auto.export - NFS automounter configuration file
   #

   # Home directories
```

```
   home -fstype=nfs <NFS/NIS server>:/export/home


Of course, replace <NFS/NIS server> with the hostname of your NFS/NIS
server.


STEP 6:  Edit the NFS exports file on the server
------------------------------------------------

Now we need to make sure that /export is shared from the server.  Edit
/etc/exports and make sure this line is present:


   /export (rw,no_root_squash)


Save the file and restart the NFS server daemons:

   killall -HUP rpc.nfsd rpc.mountd


STEP 7:  Pick a name for your NIS domain
----------------------------------------

Each NIS domain needs to have a unique name.  Do not use the same name as
your DNS domain.  This should be something different.  I will use "mynis"
in this example.

Edit /etc/defaultdomain and add the name of your NIS domain:


   mynis


Save and exit.  The name of your NIS domain shoulw be the only thing in
that file.  Once you've done that, set the NIS domain with this command:

   nisdomainname `cat /etc/defaultdomain`


STEP 8:  Edit /var/yp/Makefile
------------------------------

The NIS server has a Makefile that controls what maps are served out by
NIS.  Each time you make a change (such as adding a user), you should go
into /var/yp and type "make" to update the NIS information.

In this file we need to edit the all: target and specify the stuff we want
to share.  Here's what your all: target should look like:
```

```
   all: passwd group hosts rpc services netid protocols netgrp mail \
        shadow auto.master auto.export

Now find the line that looks like this:

   AUTO_LOCAL  = $(YPSRCDIR)/auto.local

We want to add this line just after that line:

   AUTO_EXPORT = $(YPSRCDIR)/auto.export

Now find the auto.local: target.  We need to add this target just after
that target:

   auto.export: $(AUTO_EXPORT) $(YPDIR)/Makefile
    @echo "Updating $@..."
      -@sed -e "/^#/d" -e s/#.*$$// $(AUTO_EXPORT) | $(DBLOAD) \
            -i $(AUTO_EXPORT) -o $(YPMAPDIR)/$@ - $@
      -@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@

Save the Makefile and exit.


STEP 9:  Edit /var/yp/securenets
--------------------------------

Edit /var/yp/securenets and change the "0.0.0.0   0.0.0.0" line to
something like this:

   255.255.255.0   192.168.1.0

You should specify your network address and submask there, this is only
an example.  This example restricts NIS access to any machine on the
192.168.1 class C subnet.


STEP 10:  Create an /etc/ypserv.conf file
-----------------------------------------

The only line you need in this file is:

   dns: yes

This line tells ypserv to resolve hostnames using a DNS server instead of
relying on NIS hostname maps (which you probably aren't using).  Save and
exit.


STEP 11:  Start ypbind
----------------------
```

```
Start ypbind:

   /usr/sbin/ypbind

Use ps to make sure it started and is running properly.  You will probably
see a master and slave process for it.


STEP 12:  Start ypserv
----------------------

Start ypserv:

   /usr/sbin/ypserv

Use ps to make sure it started and is running properly.


STEP 13:  Initialize the NIS maps
---------------------------------

We need to initialize the NIS maps with this command:

   /usr/lib/yp/ypinit -m

Fill in the blanks for all machines on your network that will be acting as
NIS servers.  At this point, you probably want to fire up rpc.yppasswdd to
make sure that users on NIS clients can run chfn and passwd.


STEP 14:  Done, making sure NIS stuff loads at bootup
-----------------------------------------------------

Your NIS server is now up and running.  You should edit /etc/rc.d/rc.inet2
and make sure the following block is uncommented:

   # Setting up NIS:
   # (NOTE: For detailed information about setting up NIS, see the
documentation
   # in /usr/doc/yp-tools, /usr/doc/ypbind, and /usr/doc/ypserv)
   #
   # First, we must set the NIS domainname.  NOTE: this is not
   # necessarily the same as your DNS domainname, set in
   # /etc/resolv.conf!  The NIS domainname is the name of a domain
   # served by your NIS server.

   if [ -r /etc/defaultdomain ]; then
     nisdomainname `cat /etc/defaultdomain`
   fi

   # Then, we start up ypbind.  It will use broadcast to find a server.
```

```
   if [ -d /var/yp ] ; then
     echo -n " ypbind"
     ${NET}/ypbind
   fi

   # If you are the NIS master server for the NIS domain, then
   # you must run rpc.yppasswdd, which is the RPC server that
   # lets users change their passwords.

   if [ -x ${NET}/rpc.yppasswdd ]; then
     echo -n " yppasswdd"
     ${NET}/rpc.yppasswdd
   fi

It's probably adviseable to run rpc.yppasswdd with this line:

   ${NET}/rpc.yppasswdd -e chsh -e chfn

This ensures that chfn and chsh work.  You also need to add this after the
ypbind if block:

   if [ -x ${NET}/ypserv ]; then
     echo -n " ypserv"
     ${NET}/ypserv
   fi

The ypserv program needs to run on the NIS server.



==============
THE NIS CLIENT
==============

STEP 1:  Set the NIS domain
---------------------------

Edit /etc/defaultdomain just like you did on the server and enter the name
of your NIS domain.


STEP 2:  Edit /etc/yp.conf
--------------------------

Open /etc/yp.conf in a text editor and make sure this line is present:

   ypserver <IP or hostname of the NIS server>

Specify either the IP address or hostname of your NIS server.  If you
don't have working hostnames, you must use the IP address here.
```

```
STEP 3:  Set the NIS domain
---------------------------

Set the NIS domain by typing this command:

   nisdomainname `cat /etc/defaultdomain`


STEP 4:  Run ypbind
-------------------

Start ypbind with this command:

   /usr/sbin/ypbind


STEP 5:  Test the NIS connection
--------------------------------

See if the NIS server is working with this command:

   ypcat passwd.byname

If you see the /etc/passwd entries from the NIS server, you're good to go.
Otherwise, go back to the NIS server section and see if you missed any
steps.


STEP 6:  Edit /etc/nsswitch.conf
--------------------------------

You need to edit /etc/nsswitch.conf and tell it to use NIS for passwd,
shadow, and group.  Change those entries to:

   db files nis

Change the automount entry to:

   files nis


STEP 7:  Enable NIS logins in /etc/passwd and /etc/group
--------------------------------------------------------

Edit /etc/passwd and add this line to the end:

   +:::::

Edit /etc/group and add this line to the end:
```

```
   +:::


STEP 8:  Create the /export directory for home directory mounting
-----------------------------------------------------------------

Make sure you have that directory:

   mkdir -p /export


STEP 9:  Create an rc.autofs script
-----------------------------------

The included rc.autofs script should be moved into /etc/rc.d.  Edit
rc.local and add this line:

   /etc/rc.d/rc.autofs


STEP 10:  Done, enable NIS stuff at bootup
------------------------------------------

Your NIS client is now configured.  Edit /etc/rc.d/rc.inet2 and uncomment
this block:

   # Setting up NIS:
   # (NOTE: For detailed information about setting up NIS, see the
documentation
   # in /usr/doc/yp-tools, /usr/doc/ypbind, and /usr/doc/ypserv)
   #
   # First, we must set the NIS domainname.  NOTE: this is not
   # necessarily the same as your DNS domainname, set in
   # /etc/resolv.conf!  The NIS domainname is the name of a domain
   # served by your NIS server.

   if [ -r /etc/defaultdomain ]; then
     nisdomainname `cat /etc/defaultdomain`
   fi

   # Then, we start up ypbind.  It will use broadcast to find a server.

   if [ -d /var/yp ] ; then
     echo -n " ypbind"
     ${NET}/ypbind
   fi
```

2026/02/09 22:23
2026/02/09 22:23

Linux Notes (System)

Useful notes on Linux systems.

Linux Kernel

Kernel-related stuffs.

Intel Motherboard

On startup, got a kernel warning regarding a `firmware bug, TSC_DEADLINE disabled…` or something similar. The required 'fix' is to install `intel-microcode` package.

On **Devuan**, install the 'parent' package - `firmware-linux` (which covers both intel-microcode and amd64-microcode)

```
# apt install firmware-linux
```

Initramfs will be auto regenerated.

On **Slackware**, install `intel-microcode` using script from Slackbuilds.org. Create a new initramfs (initrd) using mknitrd (with `-P /boot/intel-ucode.cpio`).

Kernel Error Message 20210328

After an upgrade on Devuan, I got

```
pstore: crypto_comp_decompress failed, ret = -22!
pstore: decompression failed: -22
```

kernel message after rebooting.

Fixed it by deleting everything in `/sys/fs/pstore`.

Linux LVM

I have some experience playing with AIX and I am very fascinated by its LVM implementation (which is a MUST-USE). I also found out that Linux also has a slightly different LVM implementation - and obviously optional. I tried it on my home machine and I want to put my personal note here. I will simply put it as it is for now… and maybe discuss with a little more detail later.

linux_lvm_notes.txt

```
* DISCLAIMER: this is my personal note. use at your own risk.

* My home machine is running Devuan
- note: mount | grep -v -E "/sys|tmpfs|proc|devpts|gvfs"
- current disk layout on /dev/sda (GPT):
  = p1: UEFI partition (fat32) 127M (1M offset)
  = p2: Root partition (ext4) 30G
  = p3: Home partition (ext4) ~898G
  = p4: Swap partition (swap) 4G

@ TASK: Create a new lv from slice of p3
- make sure basic lvm terms/concepts are covered
  = volume group (vg)
  = physical volume (pv) [aix:pv MUST BE a disk]
  = physical element/extent (pe) [aix:pp=physical partition]
  = logical volume (lv)
  = logical element/extent (le) [aix:lp=logical partition]
- need to unmount /home
  = login as root and make sure no user logged in
- target disk layout:
  = p1: UEFI partition (fat32) 127M (1M offset)
  = p2: Root partition (ext4) 30G
  = p3: Home partition (ext4) 600G
  = p4: Xtra partition (lvm) ~298G
  = p5: Swap partition (swap) 4G
- current p3 content is <500G

* repartition disk to create space for new lv
# umount /home
# e2fsck -f /dev/sda3
# resize2fs -p /dev/sda3 500G
# swapoff /dev/sda4
# gdisk /dev/sda
> delete /dev/sda4
> note /dev/sda3 starting sector
> delete /dev/sda3
> create new /dev/sda3 (linux:8300)
  = make sure use same starting sector
  = specify size +600G
> create new /dev/sda4 (linux lvm:8e00)
  = default starting sector
  = specify -4G to leave space for swap
> create new /dev/sda5 (swap:8200)
  = should have 4G remaining
> exit
* reclaim /dev/sda3 space (600G)
# resize2fs -p /dev/sda3
* edit /etc/fstab accordingly
  - e.g. specify new uuid for swap & /dev/sda3
* restore swap
# swapon -a
```

```
- OR, 'swapon /dev/sda5'

* install lvm
- using lvm2 implementation (uses device mapper)
# apt install lvm2

* create pv
# pvcreate /dev/sda4
* create volume group vg
# vgcreate xtravg /dev/sda4
* playing around (create/remove lv)
# lvcreate -L 100G -n store xtravg
# lvcreate -l +100%FREE -n movie xtravg
# lvremove movie
# lvremove xtravg/store

* create single lv on vg
# lvcreate -l +100%FREE -n share xtravg
# vgdisplay xtravg
* create ext4 fs on lv
# mkfs.ext4 -L LVMSHARE /dev/xtravg/share
* create a mountpoint for the new lv
# mkdir /ext
* check entry in /dev/mapper/
  = should have xtravg-share
* add entry in fstab
  = e.g. /dev/mapper/xtravg-share /ext ext4 defaults 0 2
* manually mount
# mount /ext

@ TASK: Add new disk to existing lv
- got a new disk, using whole disk (1TB)
- extend existing lv

# pvcreate /dev/sdb
# vgdisplay xtravg
# vgextend xtravg /dev/sdb
# vgdisplay xtravg
* use all newly available space
# lvextend -l +100%FREE /dev/xtravg/share

@ TASK: Remove PV from VG
- need to use original /dev/sda4 for something else
- note: pvs -o+pv_used
- PV is in use, so need to free up

# umount /ext
# e2fsck -f /dev/xtravg/share
# resize2fs -p /dev/xtravg/share 800G
# lvreduce -L 850G /dev/xtravg/share
* reclaim space
```

```
# resize2fs -p /dev/xtravg/share
* check usage
# pvs -o+pv_used
- OR, pvdisplay
* IF any PEs still in use
# pvmove /dev/sda4
* remove pv from vg
# vgreduce xtravg /dev/sda4
* remove pv from system (no longer id as pv)
# pvremove /dev/sda4
* reclaim all lv space
# lvextend -l +100%FREE /dev/xtravg/share
# resize2fs -p /dev/xtravg/share
```

Useful Commands

Check physical disks:

```
# pvscan
```

List volume groups / logical volumes:

```
# vgs
# lvs
```

Moving VG to another machine

On current machine, assuming volume group `myvg` mounted at `/extra`.

Unmount

```
# umount /extra
```

Deactivate VG

```
# vgchange -an myvg
```

Export

```
# vgexport myvg
```

On to the target machine, assuming mount point is also at `/extra`.

Import

```
# vgimport myvg
```

Activate VG

```
# vgchange -ay myvg
```

Mount

```
# mount /dev/mapper/myvg /extra
```

Linux AUFS

Dumped from my personal note on making AUFS-patched Linux kernel.

linux_aufs_notes.txt

```
---------------------------------
How I built an aufs-patched kernel
---------------------------------

- get kernel source
  = got 5.4.26 (latest longterm atm)
  $ getlinux 5.4.26 -x
- prepare kernel source
  = extract
  $ tar xf linux-5.4.26.tar.xz
  = actually, i got 5.4.25 like a day before 5.4.26 is release
  = got patch-5.4.25-26 and apply
  $ cd linux-5.4.25
  $ patch -p1 < ~/temp/patch-5.4.25-26
  = renamed the path as linux-5.4.26

- get the required aufsX
  = need aufs5 because i chose kernel v5
  $ git clone git://github.com/sfjro/aufs5-standalone.git
- prepare aufs5
  $ git checkout -b aufs5.4 origin/aufs5.4

- get aufs-util (for later use)
  $ git clone git://git.code.sf.net/p/aufs/aufs-util
- prepare aufs-util
  = aufs-util does not have exact kernel version (get closest lower)
  $ git checkout -b aufs-5.0 origin/aufs5.0

- apply patches in aufs5-standalone into linux
  = not all patches are required
  = refer to README in aufs5-standalone
  = i use method 1
  $ cd linux-5.4.26
  $ patch -p1 < ../aufs5-standalone/aufs5-kbuild.patch
  $ patch -p1 < ../aufs5-standalone/aufs5-base.patch
```

```
  $ patch -p1 < ../aufs5-standalone/aufs5-mmap.patch
  $ patch -p1 < ../aufs5-standalone/aufs5-standalone.patch
  = OR (in a single line command)
  $ for a in kbuild base mmap standalone ; do patch -p1 < ../aufs5-
standalone/aufs5-$a.patch ; done
- need to copy some files
  = still in linux-5.4.26
  $ cp -av ../aufs5-standalone/Documentation/* Documentation/
  $ cp -av ../aufs5-standalone/fs/* fs/
  $ cp -av ../aufs5-standalone/include/uapi/linux/aufs_type.h
include/uapi/linux/
- repack this for easy reuse
  = rename path as linux-5.4.26_aufs
  $ cd .. ; mv linux-5.4.26 linux-5.4.26_aufs
  $ cd linux-5.4.26_aufs
  $ packthis

- prepare to build
  = unpack to /usr/src
  $ cd /usr/src ; tar xf linux-5.4.26_aufs.tar.xz
- configure kernel
  $ cd linux-5.4.26_aufs
  = use make_kernel
  $ make_kernel -i -c
  = that will run make menuconfig
  = select aufs as module
- build kernel
  = use make_kernel
  $ make_kernel -x
  = wait...

- pack kernel (slackware)
  = use pack_kernel (duh!)
  $ pack_kernel --all
  = by default package will be at $HOME/temp
```

2026/02/09 22:23

Linux Desktop Environment

Some usage tips and fixes for Linux Desktop Environment(s)

JWM stuff

dumped...

jwm_key_mask.txt

```
jwm mask modifier
- mod key map can be checked using xmodmap -pm
```

My last used config - requires xdgmenumaker (getslackbuild)

dot-jwmrc

```
<?xml version="1.0"?>
<JWM>
    <RootMenu onroot="3">
        <Program icon="terminal.png" label="Terminal">st -g
110x35</Program>
        <Program icon="file-manager.png" label="File
Manager">rox</Program>
        <Program icon="text-editor.png"
label="Editor">geany_space</Program>
        <Program icon="browser.png" label="Browser">firefox</Program>
        <Separator/>
        <Program icon="drive-harddisk.png" label="Mount
Tool">my1pmount-gtk</Program>
        <Separator/>
        <Include>exec: xdgmenumaker -n -i -f jwm</Include>
        <Separator/>
        <Program icon="reload3.png" label="Reload Menu">jwm -
reload</Program>
        <Restart icon="reload.png" label="Restart JWM"/>
        <Exit icon="exit.png" label="Exit JWM"/>
        <Separator/>
        <Program icon="redo.png" label="Reboot">loginctl
reboot</Program>
        <Program icon="system-shutdown.png" label="Shutdown">loginctl
poweroff</Program>
    </RootMenu>
    <Group>
        <Name>st</Name>
        <Option>cascaded</Option>
    </Group>
    <Tray x="0" y="0" autohide="off">
        <TrayButton label="MY1">root:3</TrayButton>
        <Spacer width="2"/>
        <TrayButton label="Desktop">showdesktop</TrayButton>
        <Spacer width="2"/>
        <Pager labeled="true"/>
        <TaskList maxwidth="256"/>
        <Dock/>
        <Clock format="%Y%m%d %H:%M"></Clock>
    </Tray>
    <WindowStyle>
```

```
        <Font>Sans-10:bold</Font>
        <Width>4</Width>
        <Height>21</Height>
        <Corner>3</Corner>
        <Foreground>#FFFFFF</Foreground>
        <Background>#555555</Background>
        <Outline>#000000</Outline>
        <Opacity>0.5</Opacity>
        <Active>
            <Foreground>#FFFFFF</Foreground>
            <Background>#0077CC</Background>
            <Outline>#000000</Outline>
            <Opacity>1.0</Opacity>
        </Active>
    </WindowStyle>
    <TrayStyle group="true" list="desktop">
        <Font>Sans-10</Font>
        <Background>#333333</Background>
        <Foreground>#FFFFFF</Foreground>
        <Outline>#000000</Outline>
        <Opacity>0.75</Opacity>
    </TrayStyle>
    <TaskListStyle>
        <Font>Sans-10</Font>
        <Active>
            <Foreground>#FFFFFF</Foreground>
            <Background>#555555</Background>
        </Active>
        <Foreground>#FFFFFF</Foreground>
        <Background>#333333</Background>
    </TaskListStyle>
    <PagerStyle>
        <Outline>#000000</Outline>
        <Foreground>#555555</Foreground>
        <Background>#333333</Background>
        <Text>#FFFFFF</Text>
        <Active>
            <Foreground>#0077CC</Foreground>
            <Background>#004488</Background>
        </Active>
    </PagerStyle>
    <MenuStyle>
        <Font>Sans-10</Font>
        <Foreground>#FFFFFF</Foreground>
        <Background>#333333</Background>
        <Outline>#000000</Outline>
        <Active>
            <Foreground>#FFFFFF</Foreground>
            <Background>#0077CC</Background>
        </Active>
        <Opacity>0.85</Opacity>
```

```
    </MenuStyle>
    <PopupStyle>
        <Font>Sans-10</Font>
        <Foreground>#000000</Foreground>
        <Background>#999999</Background>
    </PopupStyle>
    <IconPath>/usr/share/icons/hicolor/16x16/apps</IconPath>
    <IconPath>/usr/share/icons/Tango/16x16/apps</IconPath>
    <IconPath>/usr/share/icons/Tango/16x16/actions</IconPath>
    <IconPath>/usr/share/icons/Tango/16x16/devices</IconPath>
    <IconPath>/usr/share/icons/Tango/16x16/categories</IconPath>
    <IconPath>/usr/share/icons/Tango/16x16/mimetypes</IconPath>
    <IconPath>/usr/share/icons/Adwaita/16x16/apps</IconPath>
    <IconPath>/usr/share/icons/Adwaita/16x16/actions</IconPath>
    <IconPath>/usr/share/icons/Adwaita/16x16/categories</IconPath>
    <IconPath>/usr/share/pixmaps</IconPath>
    <IconPath>/usr/share/jwm</IconPath>
    <Desktops width="2" height="1">
        <!--<Background type="solid">#111111</Background>-->
        <Background
type="scale">/usr/share/jwm/slackwall.png</Background>
    </Desktops>
    <!-- Double click speed (in milliseconds) -->
    <DoubleClickSpeed>400</DoubleClickSpeed>
    <!-- Double click delta (in pixels) -->
    <DoubleClickDelta>2</DoubleClickDelta>
    <!-- The focus model (sloppy or click) -->
    <FocusModel>click</FocusModel>
    <!-- The snap mode (none, screen, or border) -->
    <SnapMode distance="10">border</SnapMode>
    <!-- The move mode (outline or opaque) -->
    <MoveMode>opaque</MoveMode>
    <!-- The resize mode (outline or opaque) -->
    <ResizeMode>opaque</ResizeMode>
    <!-- Key bindings -->
    <Key key="Up">up</Key>
    <Key key="Down">down</Key>
    <Key key="Right">right</Key>
    <Key key="Left">left</Key>
    <Key key="h">left</Key>
    <Key key="j">down</Key>
    <Key key="k">up</Key>
    <Key key="l">right</Key>
    <Key key="Return">select</Key>
    <Key key="Escape">escape</Key>
    <Key mask="A" key="p">exec:dmenu_run -b</Key>
    <Key mask="A" key="Tab">nextstacked</Key>
    <Key mask="A" key="F4">close</Key>
    <Key mask="A" key="#">desktop#</Key>
    <Key mask="A" key="F1">root:3</Key>
    <Key mask="A" key="F2">window</Key>
```

```
    <Key mask="A" key="F5">move</Key>
    <Key mask="A" key="F6">resize</Key>
    <Key mask="A" key="F10">maximize</Key>
    <Key mask="A" key="F11">fullscreen</Key>
    <Key mask="A" key="Right">rdesktop</Key>
    <Key mask="A" key="Left">ldesktop</Key>
    <Key mask="A" key="Up">udesktop</Key>
    <Key mask="A" key="Down">ddesktop</Key>
    <Key mask="CA" key="Up">maxtop</Key>
    <Key mask="CA" key="Down">maxbottom</Key>
    <Key mask="CA" key="Right">maxright</Key>
    <Key mask="CA" key="Left">maxleft</Key>
    <Key mask="C4" key="v">maxv</Key>
    <Key mask="C4" key="h">maxh</Key>
    <Key mask="CA" key="l">exec:xlock -mode matrix</Key>
    <Key mask="CA" key="m">exec:my1pmount-gtk</Key>
    <Key mask="CA" key="plus">exec:volume +</Key>
    <Key mask="CA" key="minus">exec:volume -</Key>
    <Key mask="AS" key="Return">exec:st -g 110x35</Key>
    <StartupCommand>nm-applet</StartupCommand>
    <SnapMode>none</SnapMode>
</JWM>
```

DWM stuff

dumped...

dwm_hotkeys.txt

```
Mod = Alt

[Shift]+[Mod]+[Enter]   - launch terminal.
[Mod]+[b]               - show/hide bar.
[Mod]+[p]               - dmenu
[Mod]+[Enter]           - swap master with top stack
[Mod] + [j / k]         - focus on next/previous window in current tag.
[Mod] + [h / l]         - increases / decreases master size.

[Mod]+[2]               - moves your focus to tag 2.
[Shift]+[Mod]+[2]       - move active window to the 2 tag.

[Mod] + [i / d]         - increases / decreases number of windows on
master
[Mod] + [, / .]         - move focus between screens (multi monitor
setup)
[Shift]+[Mod]+[, / .]   - move active window to different screen.

[Mod]+[0]               - view all windows on screen.
```

```
[Shift]+[Mod]+[0]       - make focused window appear on all tags.
[Shift]+[Mod]+[c]       - kill active window.
[Shift]+[Mod]+[q]       - quit dwm cleanly.

[Mod]+[t]               - tiling mode
[Mod]+[f]               - floating mode
[Mod]+[m]               - monocle mod

[Mod]+[R M B]           - to resize the floating window.
[Mod]+[L M B]           - to move the floating window around.
[Mod]+[Space]           - toggles to the previous layout mode.
[Mod]+[Shift]+[Space]   - to make an individual window float.
[Mod]+[M M B]           - to make an individual window un-float.
```

st_hotkeys.txt

```
[Shift]+[Mod]+[PageUp]  - increase font size
[Shift]+[Mod]+[PageDn]  - decrease font size
[Shift]+[Mod]+[Home]    - reset font size
```

Generic Stuff

Applicable to most (if not all) DE/WM.

Geany on multiple workspaces

geany_space

```
#!/bin/bash
# from:
https://unix.stackexchange.com/questions/246827/geany-open-a-new-instan
ce-per-workspace-when-opening-a-file-in-that-workspace

socket=$(xprop -root _NET_CURRENT_DESKTOP)
socket=${socket##* }

if [ "$socket" ] ; then
    if [ "$DISPLAY" ] ; then
        socket="${DISPLAY%.*}-$socket"
        socket=${socket#*:}
    else
        socket="NODISPLAY-$socket"
    fi
    exec geany --socket-file "/tmp/geany_socket_$socket" "$@"
else
```

```
    exec geany "$@"
fi
```

geany_space.desktop

```
[Desktop Entry]
Type=Application
Version=1.0
Name=Geany (Workspace)
Comment=Geany wrapper to detect workspace
Exec=geany_space %F
Icon=geany
Terminal=false
Categories=Development;IDE;
MimeType=text/plain;text/x-chdr;text/x-csrc;text/x-c++hdr;text/x-
c++src;text/x-java;text/x-dsrc;text/x-pascal;text/x-perl;text/x-
python;application/x-php;application/x-httpd-php3;application/x-httpd-
php4;application/x-httpd-
php5;application/xml;text/html;text/css;text/x-sql;text/x-diff;
StartupNotify=true
Keywords=Text;Editor;
```

Useful Desktop Files

The *.desktop files provide menu entry for applications.

arduino-ide.desktop

```
[Desktop Entry]
Type=Application
Version=1.0
Name=Arduino IDE (App-Image)
Comment=Arduino IDE
Exec=/home/share/appx/app-image/arduino-ide %F
#Icon=
Terminal=false
Categories=Development;IDE;
#MimeType=text/plain;text/x-chdr;text/x-csrc;text/x-c++hdr;text/x-
c++src;text/x-java;text/x-dsrc;text/x-pascal;text/x-perl;text/x-
python;application/x-php;application/x-httpd-php3;application/x-httpd-
php4;application/x-httpd-
php5;application/xml;text/html;text/css;text/x-sql;text/x-diff;
StartupNotify=true
#Keywords=Text;Editor;
```

android-studio.desktop

```
[Desktop Entry]
Version=1.0
Type=Application
Name=Android Studio
Comment=
Exec=env JAVA_HOME=/home/share/appx/android-studio/jbr
/home/share/appx/android-studio/bin/studio.sh
Icon=/home/share/appx/android-studio/bin/studio.png
Path=/home/share/appx/android-studio/bin
Categories=Development;IDE;
Terminal=false
StartupWMClass=jetbrains-studio
StartupNotify=true
```

xfig.desktop

```
[Desktop Entry]
Encoding=UTF-8
Version=1.0
Type=Application
NoDisplay=true
Exec=/usr/bin/xfig -nowrite_bak %f
Name=xfig -nowrite_bak
Comment=Custom definition for xfig -nowrite_bak
```

Using xdg-mime

Dumped

```
- get mime type of <filename>
$ xdg-mime query filetype <filename>

- get currently set default app for a <mime/type>
$ xdg-mime query default <mime/type>

- set a new default app for a <mime/type>
$ xdg-mime default app.desktop <mime/type>
```

User Applications

Some specific use-cases (for my personal reference)

- Video subtitle (*.srt) editor - gaupol (apt,slackbuilds)

XFCE: Slackware64-15.0

Icons for extra/unused partitions on my hard disk are shown on XFCE desktop. I DO NOT want this!

- Create a udev rule in `udev/rules.d` path

  10-hdpart.rules

  ```
  KERNEL=="sda3", ENV{UDISKS_IGNORE}="1"
  KERNEL=="sda4", ENV{UDISKS_IGNORE}="1"
  ```

  - add more lines as required
- Reload udev rules

```
udevadm control --reload-rules && udevadm trigger
```

XFCE: Clock Panel

Custom Digital Clock settings on XFCE

```
<span size='x-small'>%Y%m%d%n</span><b>%H:%M:%S</b>
```

```
<span size='x-small'><i>%Y-%m-%d</i>%n</span><span
size='small'><b>%T</b></span>
```

```
<span size='small'><b>%Y%m%d</b></span>  %H:%M:%S
```

XFCE: Terminal Shortcut for Thunar

Note20230501 No longer needed? Simply go `Edit>Configure custom actions…` and set it there.

There is already a User Custom Action (UCA) defined to 'Open Terminal Here' - so we just need to add a shortcut.

- Find that UCA id from `~/.config/Thunar/uca.xml`
- Locate that id in `~/.config/Thunar/accels.scm`
  - uncomment and add shortcut key, e.g. F4

XFCE: Quake-Style Terminal

XFCE4 terminal can actually run in a drop-down mode (Quake-style):

```
xfce4-terminal --drop-down --hide-menubar
```

Use keyboard settings (Settings→Keyboard): Create new entry in 'Application Shortcuts' tab with the previous command, with <F12> as trigger key.

XFCE: Remove Suspend/Hibernate Button

```
xfconf-query -c xfce4-session -np '/shutdown/ShowSuspend' -t 'bool' -s
'false'
xfconf-query -c xfce4-session -np '/shutdown/ShowHibernate' -t 'bool' -s
'false'
xfconf-query -c xfce4-session -np '/shutdown/ShowHybridSleep' -t 'bool' -s
'false'
```

Sometimes this is also annoying?

```
xfconf-query -c xfce4-session -np '/shutdown/ShowSwitchUser' -t 'bool' -s
'false'
```

XFCE: Multi-User

***Note**: Latest version should be here*

Copying XFCE settings for other users:

clone-xfceconf

```
#!/bin/bash

DSTUSER=$1
[ -z "$DSTUSER" ] && echo "** Missing target username!" && exit 1

SRCUSER=$(pwd)
SRCTEST=$(dirname $SRCUSER)

while [ 1 ] ; do
[ "$SRCTEST" == "/" ] &&
    echo "** Cannot find /home in current path!" && exit 1
[ "$SRCTEST" == "/home" ] && break;
SRCUSER=$(basename $SRCTEST)
SRCTEST=$(dirname $SRCTEST)
done
echo "-- Using user '$SRCUSER' as source!"

DSTHOME="/home/$DSTUSER"
SRCHOME="/home/$SRCUSER"
```

```
[ "$DSTUSER" == "$SRCUSER" ] &&
    echo "** Target ($DSTUSER) same as source ($SRCUSER)?" && exit 1

[ ! -d "$DSTHOME" ] &&
    echo "** Cannot find target user path '$DSTHOME'!" && exit 1

[ ! -d "$SRCHOME" ] &&
    echo "** Cannot find source user path '$SRCHOME'!" && exit 1

DST_GRP=$(groups $DSTUSER | sed 's/.* : \([^[:space:]]*\).*$/\1/')
echo "-- Target:{$DSTUSER:$DST_GRP}"
[ $UID -ne 0 ] && echo -e "\n** Must run as root!\n" && exit 1

CHKCONF="Thunar geany xfce4"
for conf in $CHKCONF ; do
    SRCCONF="$SRCHOME/.config/$conf"
    [ ! -d "$SRCCONF" ] && continue
    DSTCONF="$DSTHOME/.config/$conf"
    if [ -d "$DSTCONF" ] ; then
        echo -n "-- Deleting $DSTCONF... "
        rm -rf $DSTCONF
        [ $? -ne 0 ] && echo -n "**error0** "
        echo "done."
    fi
    echo -n "-- Copying $SRCCONF as $DSTCONF... "
    cp -a $SRCCONF $(dirname $DSTCONF)
    [ $? -ne 0 ] && echo -n "**error1** "
    chown -R $DSTUSER:$DST_GRP $DSTCONF
    [ $? -ne 0 ] && echo -n "**error2** "
    echo "done."
done
```

Cinnamon: Terminal Shortcut for Nemo

Shortcut for nemo file manager

- `$HOME/.gnome2/accels/nemo`

```
(gtk_accel_path "<Actions>/DirViewActions/OpenInTerminal" "F4")
```

Cinnamon: Shortcuts

***Note**: Super key is the Windows key*

shortcuts.txt

```
Ctrl+Alt+Down: Show the window selection screen
Ctrl+Alt+Up: Show the workspace selection screen
Ctrl+Alt+End: Shut down dialog
Super+D: Show the desktop
Alt+F2: Run dialog
Alt+F8: Resize window
Alt+F7: Move window
Super+Left: Push tile left
Super+Right: Push tile right
Super+Up: Push tile up
Super+Down: Push tile down
Ctrl+Super+Left: Push snap left +
Ctrl+Super+Right: Push snap right
Ctrl+Super+Down: Push snap down
Shift+Ctrl+Alt+Left: Move window to left workspace
Shift+Ctrl+Alt+Right: Move window to right workspace
Super+Shift+Left: Move window to left monitor
Super+Shift+Right: Move window to right monitor
Super+Shift+Up: Move window to up monitor
Super+Shift+Down: Move window to down monitor
Alt+F5: Un-maximize window
Alt+F4: Minimize window
Alt+Space: Activate window menu
Alt+F10: Toggle maximization state
```

shortcuts_check.txt

```
Super+P: Re-detect display devices
```

Cinnamon: Disable Recent Files

Option is in `Menu > Preferences > Privacy`

2026/02/09 22:23

Linux Notes (User?)

Some useful things to know in Linux.

DNS Privacy

Dumped

dns_privacy.txt

```
public/free dns:
dns.quad9.net
9.9.9.9
149.112.112.112
2620:fe::fe
2620:fe::9

Recursive DNS Server Addresses and Features - Service based
configuration:

Recommended: Malware Blocking, DNSSEC Validation (this is the most
typical configuration)

IPv4
9.9.9.9
149.112.112.112

IPv6
2620:fe::fe
2620:fe::9

HTTPS
https://dns.quad9.net/dns-query

TLS
tls://dns.quad9.net

Secured w/ECS: Malware blocking, DNSSEC Validation, ECS enabled

IPv4
9.9.9.11
149.112.112.11

IPv6
2620:fe::11
2620:fe::fe:11

HTTPS
https://dns11.quad9.net/dns-query

TLS
tls://dns11.quad9.net

Unsecured: No Malware blocking, no DNSSEC validation (for experts
only!)

IPv4
9.9.9.10
149.112.112.10
```

```
IPv6
2620:fe::10
2620:fe::fe:10

HTTPS
https://dns10.quad9.net/dns-query

TLS
tls://dns10.quad9.net
```

Code Debugging

Note on using valgrind...

```
$ valgrind --leak-check=full --track-origins=yes --log-file=valgrind.rpt \
    ./my1asm85 ../my1asm85/asm/monitor85.asm

$ valgrind --tool=memcheck --leak-check=full --track-origins=yes -s \
    ./my1asm85 ../my1asm85/asm/monitor85.asm
```

User Admin

To disable user login

```
# passwd -l username
```

To re-enable user login

```
# passwd -u username
```

OR,

To disable user login

```
# usermod -L -e 1 username
```

To re-enable user login

```
# usermod -U -e "" username
```

I prefer/use the first one.

Avahi/mDNS Stuff

List all hosts in *.local

```
$ avahi-browse -atr | grep hostname | sort -u
```

Terminal Display

To display extended ASCII (which is a non-standard), use `Western-IBM850` encoding (instead of the currently common UTF-8). The extended ASCII is actually great in drawing boxes in text mode.

Manipulating PDF

To merge multiple PDFs (using pdftk):

```
pdftk $(ls *.pdf | sort) cat output ../output.pdf
```

Use xournal (or, maybe, xournalpp) to mark/sign - used export PDF to recreate PDF. If the created PDF is big (tends to be), import from LibreOffice Draw and re-export PDF.

Cleanup Stuff

Just found out about `/proc/sys/vm/drop_caches` kernel interface today. Need to look at this some time...

```
* clear page cache {flag:0x01}
# echo 1 > /proc/sys/vm/drop_caches
* clear dentries and inodes {flag:0x02}
# echo 2 > /proc/sys/vm/drop_caches
* can request simultaneously {flag:0x02|0x01}
# echo 3 > /proc/sys/vm/drop_caches
```

Setup User Development Space (Isolated?)

Got this info from somewhere online - I just put it into a script.

Note: *Create access to an IPad... requires ifuse and libimobiledevice-utils packages (tested on Devuan).*

setup_dev_env

```
#!/bin/bash
```

```
# setup_dev_env
# - got this info from internet, i just put it in a script
# - should source this file :p

WORK_PATH=`pwd`
[ -d "$1" ] && WORK_PATH=`cd $1;pwd`
echo "-- Setup environment in ${WORK_PATH}"

MAKE_PATH=0
add_path()
{
    [ -d "$1" ] && PATH="$1":$PATH && MAKE_PATH=1
}
add_path ${WORK_PATH}/bin
add_path ${WORK_PATH}/sbin
add_path ${WORK_PATH}/usr/bin
add_path ${WORK_PATH}/usr/sbin
[ $MAKE_PATH -eq 0 ] &&
    echo "** No binary path found!" && exit 1
export PATH

MAKE_PATH=0
add_ldlib_path()
{
    if [ -d "$1" ] ; then
        [ -z "$LD_LIBRARY_PATH" ] &&
            LD_LIBRARY_PATH="$1" ||
LD_LIBRARY_PATH="$1":$LD_LIBRARY_PATH
        MAKE_PATH=1
    fi
}
add_ldlib_path ${WORK_PATH}/lib
add_ldlib_path ${WORK_PATH}/usr/lib
[ $MAKE_PATH -ne 0 ] && export LD_LIBRARY_PATH

MAKE_PATH=0
add_c_path()
{
    if [ -d "$1" ] ; then
        [ -z "$CPATH" ] &&
            CPATH="$1" || CPATH="$1":$CPATH
        MAKE_PATH=1
    fi
}
add_c_path ${WORK_PATH}/include
add_c_path ${WORK_PATH}/usr/include
[ $MAKE_PATH -ne 0 ] && export CPATH

MAKE_PATH=0
add_man_path()
```

```
{
    if [ -d "$1" ] ; then
        [ -z "$MANPATH" ] &&
            MANPATH="$1" || MANPATH="$1":$MANPATH
        MAKE_PATH=1
    fi
}
add_man_path ${WORK_PATH}/share/man
add_man_path ${WORK_PATH}/usr/share/man
[ $MAKE_PATH -ne 0 ] && export MANPATH

MAKE_PATH=0
add_pkgcfg_path()
{
    if [ -d "$1" ] ; then
        [ -z "$PKG_CONFIG_PATH" ] &&
            PKG_CONFIG_PATH="$1" ||
PKG_CONFIG_PATH="$1":$PKG_CONFIG_PATH
        MAKE_PATH=1
    fi
}
add_pkgcfg_path ${WORK_PATH}/lib/pkgconfig
add_pkgcfg_path ${WORK_PATH}/usr/lib/pkgconfig
[ $MAKE_PATH -ne 0 ] && export PKG_CONFIG_PATH
```

Support for RTL8723DE WiFi Module

I have Slackware 14.2 on my HP laptop, which has an RTL8723DE wifi module hardware. This module is still not supported in the mainstream kernel, but the driver is available at https://github.com/lwfinger/rtlwifi_new.git.

- get source, go extended

```
$ git clone https://github.com/lwfinger/rtlwifi_new.git
$ cd rtlwifi_new
$ git checkout extended
```

- to install (as root)

```
$ make install
$ modprobe rtl8723de
```

- to uninstall module (e.g. for kernel update)

```
$ rmmod rtl8723de
$ make uninstall
```

- to get better signal strength, create file '/etc/modprobe.d/rtl8723de.conf' with:

```
options rtl8723de ant_sel=2
```

Update20200824 *That repo is no longer available. Using rtw88 instead (module is now named rtw88_8723de). Just to note that the rtw88 code is now in mainline kernel starting 5.2*

KVM Stuff

Somehow, `/dev/kvm` is missing?? So, just create this udev rule:

65-kvm.rules

```
KERNEL=="kvm", GROUP="users", MODE="0660"
KERNEL=="vhost-net", GROUP="users", MODE="0660"
```

Text-manipulation

I mostly use `sed` / `grep` / `cut` for this. But I have found the need to use `tr` at times.

For example, to filter out non-ASCII characters

```
tr -cd '\000-\177'
```

Removing ldlinux.sys

When installing using extlinux, the file ldlinux.sys is created and cannot be removed even by root! The command `chattr` can be used to remove the flag that protects that particular file

```
# chattr -i ldlinux.sys
```

To check/confirm, run

```
# lsattr ldlinux.sys
```

After that, the file can be removed by root as usual.

GTK3 Stuff

Running my GUI code compiled with GTK3 on Artix causes a runtime warning message

```
dbind-WARNING **: 11:12:11.208: Couldn't register with accessibility bus:
Did not receive a reply. Possible causes include: the remote application did
```

```
not send a reply, the message bus security policy blocked the reply, the
reply timeout expired, or the network connection was broken.
```

I found this simple fix somewhere in a forum - simply create an environment variable

```
export NO_AT_BRIDGE=1
```

To check screen dpi

```
$ xdpyinfo
$ xrdb -query
```

Optimize hard disk using hdparm

Note to self: CHECK THIS OUT! For example, on T23 laptop

```
$ hdparm -q -d1 -c3 -W1 -u1 -m 16 /dev/hda
$ hdparm -q -d1 -c3 -X66 /dev/hdc
```

Adding fonts in Linux

```
1. xorg.conf, "mkfontscale", "mkfontdir", "ttmkfdir" => old school, nobody
needs them?
2. /etc/fonts/*, ~/.fonts.conf, fc-cache, fc-list => the way to go
```

To strip binary/library files

```
$ strip --strip-debug /lib/*
$ strip --strip-unneeded /{,s}bin/*
```

Create a patch file

```
$ diff -Naur [file1] [file2] > file.patch
**Note**
  %%{{{@@ -X,Y +J,K @@}}}%% is a hunk where diff is
    X & J - starting line number
    Y & K - line counts
```

Script to create multiple users

create_users

```
#!/bin/bash

USERLIST="$1"

[ ! -f "$USERLIST" ] &&
    echo "Cannot access user list '$USERLIST'! Abort!" && exit 1

# assume format is user:pass
for userpass in `cat $USERLIST` ; do
    user=`echo $userpass | sed 's/\([^:]\):.*/\1/'`
    pass=`echo $userpass | sed 's/.*:\(.*\)/\1/'`
    what=`echo $userpass | sed 's/^\(#\)[^#]*/\1/'`
    [ "$what" == "#" ] && continue
    what=`cat /etc/passwd | grep "$user"`
    [ "$what" != "" ] && continue
    echo "User:[$user],Pass:[$pass]"
    echo "${user}:${pass}::100:User Account:/home/${user}:/bin/bash"
done
```

NTP (System Date/Time)

By default, date/time are set manually: e.g. as root,

```
ntpdate pool.ntp.org
```

If ntpdate is not available (being deprecated?), use ntp daemon:

```
ntpd -gq
```

(May have to stop running ntp daemon)

For Slackware, a startup script for NTP daemon is available `/etc/rc.d/rc.ntpd`, but do not forget to modify `/etc/ntp.conf` and uncomment (or add custom) NTP server information. To query, use

```
ntpq -p
```

To still use `ntpdate` while NTP daemon is running, use the `-d` switch,

```
ntpdate -d pool.ntp.org
```

Useful /proc Interface

To display active pertitions,

```
cat /proc/partitions
```

To display cpu information,

```
cat /proc/cpuinfo
```

To check for Intel-VT virtualization features,

```
grep --color vmx /proc/cpuinfo
```

To check for AMD-V virtualization features,

```
grep --color svm /proc/cpuinfo
```

***Note**: Some BIOS may disable this feature by default. Also, old Linux kernel may not support it either.*

To display memory information,

```
cat /proc/meminfo
```

To check shared libraries used by a program

```
cat /proc/<proc_id>/maps
```

Playing with LDAP

I want to have LDAP-based central authentication for my two new Slackware machines used for my GMC project. Things to do:

1. get LDAP server installed and running
    - Slackware only have openldap-client by default
    - I'm referring to here to do this
2. figure out how to setup the client side
    - I may need nss_ldap only - will check into this later

SSH X Forwarding

On server, edit `/etc/ssh/sshd_config`

```
AllowTcpForwarding yes
X11Forwarding yes
X11DisplayOffset 10
```

```
X11UseLocalhost yes
```

On client, edit `/etc/ssh/ssh_config`

```
ForwardAgent yes
ForwardX11 yes
```

To connect,

```
# allow remote access for display
ssh -Y user@host
export DISPLAY=localhost:10.0
# then, run any x-program
```

Display Access Control

Only the user on the main console gets control of display protocol. If we do `su`, the root user cannot use any GUI.

To disable and enable access control

```
$ xhost +
$ xhost -
```

Find and set DISPLAY value accordingly

```
$ echo $DISPLAY
$ export DISPLAY=:0.0
```

Remote VNC (Remote Desktop)

This enables us to have GUI access on a remote machine. I use screen because I want to keep the running terminal 'alive'.

Use virtual terminal, run `screen` on the remote machine

```
$ screen -S VNC
```

Start VNC server on a display port

```
$ vncserver :23
```

Note *~/.vnc/xstartup will be executed*

To stop the VNC server

```
$ vncserver -kill :23
```

Detach the virtual terminal by hitting *<CTRL+A>-<D>*. To resume the virtual terminal from any console

```
$ screen -r VNC
```

To tunnel through SSH, start an ssh session (with port forwarding [5900+DP] and going background)

```
ssh -L 5923:localhost:5923 -N -f user@remotehost
```

Connect as if the server is on localhost

```
vncviewer localhost:23
```

Note: *Desktop response becomes too slow for me! Not using this...*

Console Multi-Tasking

On Linux, multi-tasking is also available on console. To suspend a task, hit `ctrl+z`

To list all jobs **duh!**

```
$ jobs
```

To resume

```
$ fg %<num>
```

VirtualBox Stuffs

Rebuild VirtualBox kernel module

After a kernel upgrade... do a

```
# /etc/init.d/vboxdrv setup
```

Virtual Serial Port

We can actually use minicom to connect...

```
minicom -D unix\#/tmp/xxx
```

where /tmp/xxx is the host pipe 😎

Convert Image to Disk

```
VBoxManage convertfromraw -format VDI <file.img> <file.vdi>
```

Compact Disk Image

```
VBoxManage modifymedium --compact <file.vdi>
```

Create Disk Image

```
VBoxManage createmedium disk --filename <file.vdi> --size <megabytes>
```

Video4Linux Stuffs

```
v4l2-ctl --list-devices
v4l2-ctl -d 0 --list-formats-ext
v4l2-ctl -d 0 --list-ctrls

v4l2-ctl --device=0 --set-ctrl=?

v4l2-ctl -d 0 --set-fmt-video=width=1920,height=1080,pixelformat=YUYV
```

Mounting FreeBSD Partition (Slice)

Note *At the moment, only read-only access (no write permission)!*

```
# mount -t ufs -o ufstype=ufs2 /dev/<partition> <mount-path>
```

Backing up dokuwiki pages on github

dokuwiki2github.sh

```
#!/bin/bash

DOKUWIKI=${DOKUWIKI:="$HOME/public_html/dokuwiki"}
WIKIPAGE=${WIKIPAGE:="$DOKUWIKI/data/pages"}

cd ${WIKIPAGE} && git add . && git add -u &&
    git commit -a -m "Content update `date +'%H:%M %d/%m/%Y %Z'`" &&
    git push origin master

# setup a cron job for hourly update:
#
# 0 * * * * dokuwiki2github.sh
```

Linux kernel patch for CH341 USB2Serial

Tested this on Slackware 14.2.

Using patch found here, I modified it a bit for latest Linux kernel used in Slackware 14.2 (4.4.190)

To create this patch, I copied out `ch341.c` from kernel source to my user path. Make another copy as `ch341_patched.c`. Modify as required. Do,

```
$ diff -u ch341.c ch341_patched.c > linux_4.4.190_ch341.patch
```

I do not know if this last step is needed, but I modified the patch so that the first two lines have the same file name.

linux_4.4.190_ch341.patch

```
--- ch341.c   2019-10-24 15:02:10.039646991 +0800
+++ ch341.c   2019-10-24 15:06:14.244416258 +0800
@@ -358,6 +358,7 @@
    struct ch341_private *priv = usb_get_serial_port_data(port);
    unsigned baud_rate;
    unsigned long flags;
+ unsigned int par_flags;

    baud_rate = tty_get_baud_rate(tty);

@@ -371,6 +372,30 @@
     * (cflag & PARENB) : parity {NONE, EVEN, ODD}
     * (cflag & CSTOPB) : stop bits [1, 2]
     */
+ /* CH340 doesn't appear to support variable stop bits or data bits */
+ if (C_PARENB(tty)) {
+     if (C_PARODD(tty)) {
+         if (tty->termios.c_cflag & CMSPAR) {
+             dev_dbg(&port->dev, "parity = mark\n");
+             par_flags = 0xeb;
+         } else {
+             dev_dbg(&port->dev, "parity = odd\n");
+             par_flags = 0xcb;
+         }
+     } else {
+         if (tty->termios.c_cflag & CMSPAR) {
+             dev_dbg(&port->dev, "parity = space\n");
+             par_flags = 0xfb;
+         } else {
+             dev_dbg(&port->dev, "parity = even\n");
+             par_flags = 0xdb;
+         }
+     }
+ } else {
```

```
+     dev_dbg(&port->dev, "parity = none\n");
+ par_flags = 0xc3;
+ }
+ ch341_control_out(port->serial->dev, 0x9a, 0x2518, par_flags);

    spin_lock_irqsave(&priv->lock, flags);
    if (C_BAUD(tty) == B0)
```

Steps (as root) in terminal:

1. get to location

```
$ cd /usr/src/linux/drivers/usb/serial
```

2. patch `ch341.c`

```
$ patch < /path/to/linux_4.4.190_ch341.patch
```

   - `path/to` is the path where you save the above patch

3. get to top kernel source path

```
$ cd /usr/src/linux
```

4. compile module

```
$ make M=drivers/usb/serial modules
```

5. (OPTIONAL) remove previously loaded module

```
$ rmmod ch341
```

6. 'install' the module

```
$ cp drivers/usb/serial/ch341.ko /lib/modules/`uname -
r`/kernel/drivers/usb/serial/
```

2026/02/09 22:23

Linux Shell

Useful command-line stuff.

Raw Disk Data

Using dd

```
$ dd if=ifile of=ofile skip=offs count=size status=progress
```

where `skip` and `count` values are in sector counts.

dd_size.txt

```
- note on byte-size & sector size (for 512-bytes sector size)
01M:     1048576byte @ 000002048sect
01G:  1073741824     @ 002097152
10G: 10737418240     @ 020971520
50G: 53687091200     @ 104857600
80G: 85899345920     @ 167772160
```

File Management

Using `find`... dumped

```
$ find . -type d -empty -print
$ find . -type d -empty -delete

$ find -type f \( -name "*zip" -o -name "*tar" -o -name "*gz" \) -size +1M -
delete
$ find -type f \( -name "*zip" -o -name "*tar" -o -name "*gz" \) -size +1M -
exec rm {} +
```

2026/02/09 22:23

Devuan Installation

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

Permanent link:
https://azman.my1matrix.cc/doku.php?id=linux:linux&rev=1770522381

Last update: **2026/02/08 03:46**