Installing OpenBSD 6.1 on your laptop is really hard (not)

I used the steps below to install OpenBSD, add the xfce4 desktop and to set up a graphical log-in on my Thinkpad X200 laptop.

Each step starts with a link to the relevant OpenBSD FAQ pages so you can cross check my suggestions. I recommend that you print this page out and **read the whole page** before pressing on with the installation.

Step 0: Install OpenBSD according to the instructions in FAQ

[top [#top]]

Background reading: FAQ 4: The OpenBSD installation guide [http://www.openbsd.org/faq/faq4.html] .

I usually plan to install a new kind of system *twice*. The first install is just to see how things work, and to make mistakes. I accept all the usual default settings. For instance, I select `[W]hole` disk and `[A]uto` partition settings at the appropriate point in the OpenBSD installation. I then install again, this time with the customised settings that I want, as I know what responses the system will give.

The steps below assume that you have successfully installed the base OpenBSD system from the USB stick installer (`install61.fs`) or the CD-ROM image (`install61.iso`).

Step 1: Connect to wifi

[top [#top]]

Background reading: OpenBSD FAQ 6.2.1 [http://www.openbsd.org/faq/faq6.html#Setup.if] , 6.13 [http://www.openbsd.org/faq/faq6.html#Wireless] .

I have a Netgear USB wifi stick that has a free/libre driver that does not require firmware. On OpenBSD, each kind of wifi device has a different driver name, like `iwn0` for Intel 5900 cards and `urtw0` for the Netgear. Use the `ifconfig -a` command to find the name of the driver for your particular card. Below is a typical dialogue for starting up an encrypted (wpa-personal) wifi connection.

```
$ su -l
# ifconfig -a # shows a list of all the interfaces
# ifconfig urtw0 up
# ifconfig urtw0 scan
# ifconfig urtw0 nwid connection_name wpakey password wpaprotos wpa1,wpa2
# dhclient urtw0
DHCPREQUEST on urtw0 to 255.255.255.255 # lots more output
```

Starting with OpenBSD 6.1, you need to use the `wpaprotos` option with argument `wpa1` to enable connections using `wpa1` encryption because of security issues with the older protocol.

You have to repeat these commands each time you connect to wifi, including on resume from suspend. There are various scripts available that automate reconnection. A very simple example is shown in Step 1a below.

You may have to install firmware for some wifi cards [http://firmware.openbsd.org/firmware/] using the `fw_update` command, e.g. Intel cards. A wired connection to the router will enable you to connect to the mirror to get the firmware for your particular card. On my Thinkpad, the ethernet driver is called `em0` and connecting to a wired connection is just one command…

```
# dhclient em0
DHCPDISCOVER on em0 - interval 3
DHCPOFFER from 192.168.0.1 (00:1b:2f:42:41:42)
DHCPREQUEST on em0 to 255.255.255.255
DHCPACK from 192.168.0.1 (00:1b:2f:42:41:42)
```

```
bound to 192.168.0.4 -- renewal in 43200 seconds.
#
```

Step 2: Set up a package mirror and install a package

[top [#top]]

Background reading: OpenBSD FAQ 15.2 [http://www.openbsd.org/faq/faq15.html#PkgMgmt] .

Precompiled binaries for application software that is not part of the OpenBSD base are called 'packages' and they are available from your local OpenBSD mirror [http://www.openbsd.org/ftp.html] . You use the `pkg_add` command as root to install packages. The `pkg_add` command reads the URL of the package mirror from the `/etc/installurl` file.

If you installed the OpenBSD package sets from the Internet, you will already have the`/etc/installurl` file in place and you can go to step 3. If, like me, you prefer to install OpenBSD from the `install.iso` or `install.fs` images, you will need to create the `/etc/installurl` file as below…

```
$ cat /etc/installurl
https://www.mirrorservice.org/pub/OpenBSD
```

The `pkg_add` command will append the version and machine architecture directories from the URL. Don't add a trailing slash or any version number or machine architecture after the 'OpenBSD' part of the URL.

A fresh OpenBSD install has two command line editors, `vi` and `mg`. I'm not a real hacker so I use `echo` and redirection. As root…

```
$ su -l
# echo "https://www.mirrorservice.org/pub/OpenBSD" >> /etc/installurl
# exit
```

To install applications, you need to become root and run `pkg_add`.

```
$ su -l
# pkg_add nano
quirks-2.114 signed on 2015-08-09T15:30:39Z
nano-2.4.2: ok
# exit
$
```

Once the command returns, exit root and try editing a text file with nano.

Step 3: Install the xfce4 desktop environment and some applications

[top [#top]]

The packages below will provide the xfce4 desktop, a Web browser/email client and a pdf viewer.

```
# pkg_add -v consolekit2 xfce xfce-extras evince seamonkey xscreensaver
```

The `consolekit2` package is needed to allow the user to log out of xfce4 without using terminal commands. ConsoleKit essentially wraps xfce4 in a session with some extra permissions.

Notice that `pkg_add` will stop when it reaches the document reader Evince and offer you a choice of two versions of the package, each compiled with different configuratons…

```
# pkg_add evince
quirks-2.114 signed on 2015-08-09T15:30:39Z
Ambiguous: choose package for evince
a       0:
        1: evince-3.16.1p0
        2: evince-3.16.1p0-light
Your choice: 2
```

Because Evince is part of the Gnome Desktop suite of programs, choosing option 1 will pull in a large number of Gnome libraries, including part of Nautilus the Gnome file manager. Option 2 has been provided by the packager for those of us who wish to use Evince to read pdf files with a different desktop or window manager. The functionality appears to be similar, just less dependencies on other parts of Gnome.

Firefox is a much more popular choice for the Web browser, but I think I prefer **Seamonkey**. I run Seamonkey with the `no-script` plugin and with options set in a very conservative way. This reduces the load on the processor, and keeps me safe on the Web. The venerable `xscreensaver` provides desktop blanking and locking when you step away from your machine.

Some of the more complex packages - especially those that install daemons - come with `readme` files installed to `/usr/local/share/doc/pkg-readmes/`. It is best to skim these for pointers to configuration.

Don't reboot yet. You need to configure the graphical login and set up some daemons. See step 4 below…

Step 4: Use `/etc/rc.conf.local` to enable apmd and graphical log-in

[top [#top]]

Background reading: Comparison of Desktop Environments [https://en.wikipedia.org/wiki/Comparison_of_X_Window_System_desktop_environments] , ConsoleKit Github readme with definitions [https://github.com/ConsoleKit2/ConsoleKit2] , xenodm man page [http://man.openbsd.org/OpenBSD-6.1/xenodm] and the package_readme for consolekit2 at `/usr/local/share/doc/pkg-readmes/consolekit2-1.0.2p1` .

As root add some lines to `/etc/rc.conf.local` to enable power management (`apmd`) so that you can use Fn-F4 to suspend your thinkpad, and to enable the graphical log-in manager `xenodm`. Xenodm is an OpenBSD fork of the venerable `xdm`.

```
# nano /etc/rc.conf.local
multicast_host=YES       # Some avahi shenanigans
apmd_flags="-A"          # Laptop power saving
xenodm_flags=""          # Starts xenodm graphical login
pkg_scripts="messagebus" # Enables dbus/ConsoleKit stuff
```

Then **as user** add an `.xsession` file with a line that will start consolekit so that you can shutdown &c from within xfce4.

```
$ cat .xsession
exec ck-launch-session startxfce4
```

Reboot and you'll get the xenodm login greeter. When you log in, Xfce4 will ask you to specify a layout, and then show you the desktop. One unusual feature is the X console window showing on Desktop 1 - it looks like a small terminal window. The X console will spit out messages when you plug in e.g. a USB stick. Shutdown, suspend and restart should work from the xfce4 Logout menu item - check they are not greyed out and that they work.

Step 5: Use `/etc/doas.conf` to allow user mounting of an external USB stick

[top [#top]]

Background reading: OpenBSD FAQ sections 10 (doas) [https://www.openbsd.org/faq/faq10.html#doas] , 14 (File Systems Intro) [] as well as `man doas` and `man mount`.

You must use `doas` and a few lines in `/etc/doas.conf` to allow user mounting of USB sticks. My `/etc/doas.conf` file looks like this…

```
$ cat /etc/doas.conf
# http://daemonforums.org/showthread.php?t=9774
permit nopass keith as root cmd mount
permit nopass keith as root cmd umount
```

Once OpenBSD sources the `doas.conf` file, you can mount and unmount(say) an external USB thumb drive formatted to VFAT like this…

```
doas mount /dev/sd1i /home/keith/usb    # mounts my USB on ~/usb
doas umount /dev/sd1i                   # un-mounts the drive
```

I knew that my USB stick corresponded to the `/dev/sd1i` device because I ran the `dmesg` command after plugging the USB stick in and waiting a few seconds. The device will be listed in the last few lines of the dmesg output something like this…

```
umass0 at uhub0 port 2 configuration 1 interface 0 "Kingston DataTraveler 112" rev 2.00/1.00 addr 3
umass0: using SCSI over Bulk-Only
scsibus4 at umass0: 2 targets, initiator 0
sd1 at scsibus4 targ 1 lun 0:  SCSI2 0/direct removable serial.0951162aFCC127195547
sd1: 14762MB, 512 bytes/sector, 30233588 sectors
```

Once mounted, you can use a graphical file manager like Thunar to copy and paste files to and from your storage stick. You can't unmount the USB stick from Thunar, remember to use the `umount /dev/sd1i` command before removing the USB stick.

Step 6: Use `xfce4-mount-plugin` and an `/etc/fstab` entry to allow graphical mount/unmount of a USB thumb drive

[top [#top]]

Background reading: xfce4-mount-plugin page on the Xfce Web site [http://goodies.xfce.org/projects/panel-plugins/xfce4-mount-plugin] .

A note on how disks get numbered: My laptop has SATA hard drive as its fixed disc, and that device will appear as `/dev/sd0` to OpenBSD. If I install from a CD-ROM and *don't* use full disk encryption, the first USB stick I plug in will appear as `/dev/sd1`. If I *do* use hard drive encryption, OpenBSD will be using a softraid disc that will appear as `/dev/sd1`, and the first USB stick that I plug in will appear as `/dev/sd2`. The safest thing to do when following the instructions in this step is to run the `mount` command without any arguments. That gives you a list of what is mounted where…

```
$ mount
# example from OpenBSD with hard drive encryption
# and installed to a softraid disc at /dev/sd1
/dev/sd1a on / type ffs (local)
/dev/sd1k on /home type ffs (local, nodev, nosuid)
/dev/sd1d on /tmp type ffs (local, nodev, nosuid)
/dev/sd1f on /usr type ffs (local, nodev)
/dev/sd1g on /usr/X11R6 type ffs (local, nodev)
/dev/sd1h on /usr/local type ffs (local, nodev, wxallowed)
/dev/sd1j on /usr/obj type ffs (local, nodev, nosuid)
/dev/sd1i on /usr/src type ffs (local, nodev, nosuid)
/dev/sd1e on /var type ffs (local, nodev, nosuid)
/dev/sd2i on /home/keith/usb type msdos (local, uid=1000, gid=1000)
```

The instructions below reflect an OpenBSD installation on `sd0` with a USB stick that will appear at `sd1`.

I **install the `xfce4-mount` package using `pkg_add`**, and then **add an to the XFCE4 panel** by right-clicking on the panel and selecting **Panel | Add New Items** and searching for mount.

By default, `xfce4-mount-plugin` lists all the devices including the default local hard drive including all the partitions on `sd0`. I can set options to prevent that and to use a custom mount command. Right click over the xfce4-mount icon and **select Properties | File Systems** tab. I just added the pattern `/dev/sd0*` to the Exclude specified file systems textbox so my local drive was not listed.

I then once again right-clicked on the xfce4-mount icon, and selected **Preferences | Commands** and wrote the following in the Custom Commands textboxes, after ensuring that the Custom Commands checkbox was ticked…

```
doas mount %m
doas umount %m
```

Now to ensure that a USB stick is *listed* in the xfce4-mount-popup list, you have to add a line for the device to `/etc/fstab`. My extra line looks like this (adapted from the examples in `man fstab`…

```
/dev/sd1i /home/keith/usb msdos rw,noauto 0 0
```

Using an `fstab` entry like this means that only one USB thumb drive will be listed and available with mouse clicks. If you are in the habit of using several USB thumb drives then just experiment with different lines and mountpoints

Thunar has volume management enabled by default, so mounted drives will be listed on the left hand side of the file window with an 'eject' icon next to each device. Attempting to eject a mounted drive by clicking on the eject icon will give an error message and has no effect. To avoid these error messages, I have unticked Enable Volume Management in the Thunar Preferences. The USB devices are still listed but with no eject icon next to them.

Step 7: Set up updates

[top [#top]]

Background reading: pages about following the `-stable` branch [http://www.openbsd.org/stable.html] or following the `-current` branch [http://www.bsdnow.tv/tutorials/stable-current-obsd] , and the OpenBSD 6.1 Errata [https://www.openbsd.org/errata61.html] page.

The `syspatch` command provides binary updates to the core system. Running the `syspatch` command without arguments while connected to the Internet resulted in the first four errata for OpenBSD 6.1 [] being installed as you can see in the transcript below.

```
$ su
Password:
# syspatch
Get/Verify syspatch61-001_dhcpd.tgz 100% |**************| 71730       00:00
Installing patch 001_dhcpd
Get/Verify syspatch61-002_vmmfpu.tgz 100% |*************|  9377 KB    00:39
Installing patch 002_vmmfpu
Get/Verify syspatch61-003_libress... 100% |*************| 11391 KB    00:51
Installing patch 003_libressl
Get/Verify syspatch61-004_softrai... 100% |*************|  9356 KB    00:41
Installing patch 004_softraid_concat
```

M:Tier sponsors OpenBSD and has provided `binpatches` for the packages in the stable release for use together with the `openup` script. Their update page [https://stable.mtier.org/] has been updated for OpenBSD 6.1, and they recommend using `openup` to upgrade packages that you have installed in addition to the base system.

Step 8: Read

[top [#top]]

Read the man pages and the package readmes. Putting it all together can be difficult at first as the documents reference each other, but it gets familiar with experience and experimentation like most things.

Step 0a: Install OpenBSD with whole drive encryption (advanced)

[top [#top]]

Background reading: OpenBSD FAQ sections 14.1 [http://www.openbsd.org/faq/faq14.html#intro] , 14.2 [http://www.openbsd.org/faq/faq14.html#fdisk] and 14.3 [http://www.openbsd.org/faq/faq14.html#disklabel] . It is probably better to try this *after* you have worked through a default install and become familiar with fdisk and disklabel.

I like to use an encrypted hard drive just in case I leave this laptop on the bus or it gets stolen by a petty thief who does not realise how old the laptop is.

OpenBSD provides encryption through its bioctl [http://www.openbsd.org/cgi-bin/man.cgi/OpenBSD-5.8/man8/bioctl.8?query=bioctl&arch=i386&manpath=OpenBSD-5.8] RAID management interface. Essentially you create an encrypted softraid device that looks like another disk to the OpenBSD system and then create the file system partitions within that device. My recipe is adapted from a tutorial by David Crupmton [http://davidcrumpton.blogspot.co.uk/2013/11/openbsd-54-full-disk-encryption.html] .

The steps below assume that you are installing onto the whole of the single hard drive using a SATA interface and that OpenBSD sees the physical hard drive as `sd0` and the USB stick I booted from as `sd1`.

1. Boot from the installer and select `[S]hell` to get the root prompt #
2. `# fdisk -iy sd0` initialises the disk
3. `# disklabel -E sd0` enters the partition editor. Once in the disklabel command prompt >…
   - type `?` to see a list of the commands within disklabel (I find this reassuring)
   - type `a` and return to create a partition
   - type `a` and return to create the sd0a partition
   - accept the default start [64] and end [size of disk] as the size
   - type `RAID` as the partition type
   - type `w` to write the partition
   - type `q` to quit the disklabel program and return to the root prompt #
4. `# bioctl -c C -l /dev/sd0a softraid0` to create an encrypted RAID device. Enter a strong passphrase at the prompt, and then enter the phrase again to check the typing. A look at the bioctl man page will clarify the option letters and the result is an encrypted RAID device that looks like a disk to the disklabel program. The encrypted device is identified as sd2 on my system because sd1 is the USB stick that I booted from and sd0 is the hard drive in the laptop.
5. `# exit` to return to the OpenBSD installer
6. Work through the installer steps until the "`Available disks are: sd0 sd1 sd2`" question is reached. Select the softraid device `sd2` and specify `[W]hole` disk. I got a warning after this step: "`MBR is not showing a valid signature, ignoring it`" but everything seems to be working.
7. I selected `[A]uto` partition layout and got the usual half dozen partitions within the softraid device sd2.
8. Complete the rest of the installer steps and reboot into the new installation
9. Type in the passphrase at the prompt just after the kernel loads and you should see the usual default OpenBSD boot dialogue
10. You may see a boot message something like "`softraid is roaming used to be sd2 now sd1 using UUID.a`" where UUID is some long random disk identifier. The USB stick I installed from is no longer connected and so isn't using the sd1 identifier, so that identifier is allocated to the encrypted softraid device.

The swap partition is within the encrypted softraid device so we can disable OpenBSD's default encryption of the swap partition. You can do that by copying the sysctl.conf file from /etc/examples/sysctl.conf to /etc/sysctl.conf and uncommenting the line that reads `vm.swapencrypt.enable=0`. My commands looked like this…

```
# cp /etc/examples/sysctl.conf /etc/sysctl.conf
# nano /etc/sysctl.conf
vm.swapencrypt.enable=0 # uncomment this line or just add this line to empty sysctl.conf file
# cat /etc/sysctl.conf | grep vm.s
vm.swapencrypt.enable=0 # 0=Do not encrypt pages that go to swap
```

Step 1a: Simple wifi script

[top [#top]]

There is no graphical wifi manager available on OpenBSD. Many users have written fancy scripts that will automatically reconnect and/or list the strongest available wifi signals. I like this little script that runs from my user account and simply automates the typing in of the `ifconfig` commands. None of the network related configuration files are changed so I can always fall back on the `ifconfig` commands in a new place. The script itself also serves as a reminder of the syntax of the commands.

```
$ cat bin/wifi
#!/bin/sh
# adapted from http://marc.info/?l=openbsd-tech&m=146490607627340&w=2

if [[ $1 == "home" ]]; then
        doas ifconfig iwn0 nwid home_wifi_name wpa wpakey home_wifi_password wpaprotos wpa1,wpa2
        doas dhclient iwn0
fi

if [[ $1 == "blackberry" ]]; then
        doas ifconfig iwn0 nwid phone_hotspot_name wpa wpakey phone_hotspot_password
        doas dhclient iwn0
```

```
fi
```

The script requires the following lines to be added to `/etc/doas.conf`.

```
permit nopass keith as root cmd ifconfig
permit nopass keith as root cmd dhclient
```

Keith Burnett, 6th May 2017: added `syspatch` command output.