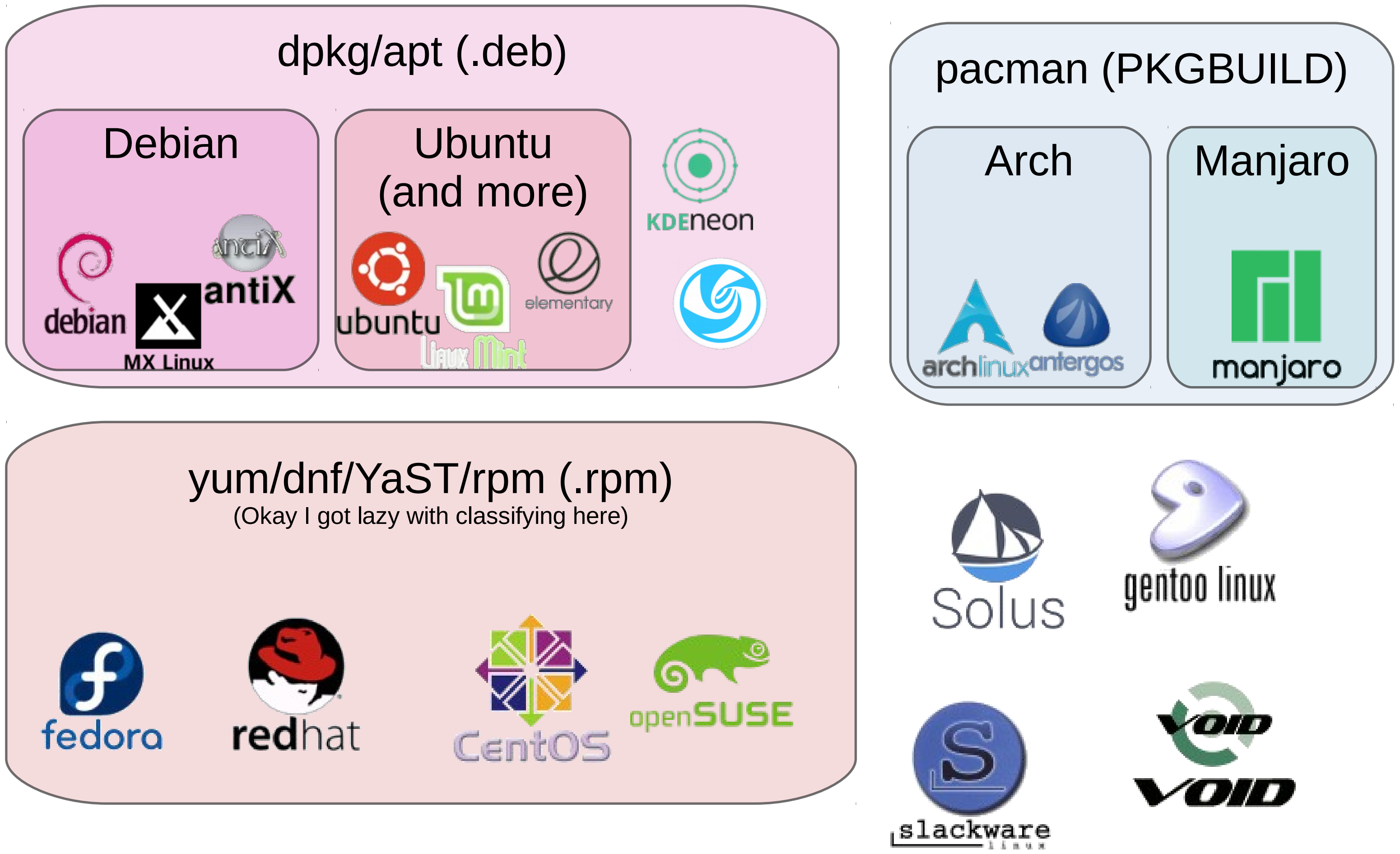
Package Management & Cross-Distro Packages

Package Managers

(I tried my best on accuracy, but don’t quote me)



What’s wrong

- Dependency hell
  - Deleting libraries that packages didn’t say they needed
- Bitrot: Packages leave behind cruft when uninstalled
- Impossible for program developers to test for
  - Which package manager, which versions, what other differences
- Package managers update, developers can’t

Alternatives?

- Option 1: Bundles: Author packages everything needed to run together into one bundle

snapcraft

Bundles

- Pros:
  - Author can test and deploy all the libraries they use
  - Applications in a single file
  - Automatic updates (Snap & Flatpak)
- Cons:
  - Author must update and maintain all the libraries they use
  - Applications in a large file

Basic mechanics

- Bundle as many dependencies as you want
- To run:
  - Mount the archive
  - Potentially sandbox it to the archive + additional chosen directories
  - Run the program based on those paths
- Might have their own dependencies and layers
  - e.g. OSTree

Comparison of Bundles

	Snapcraft	Flatpak	AppImage
How to run	Install, use priveleged daemon	Install, use priveleged daemon	Run as any user. Optional daemon
Supported by	Canonical	Red Hat (& Fedora Project)	Community only
Repositories	Curated store owned by Canonical (hardcoded)	Multiple repositories, Free to host own	No official repositories
Bundling	Single bundle with sandbox metadata, Base snaps now	OSTree Layers ala Docker (package management again!)	Single bundle
Sandboxing	Always AppArmor	Always Bubblewrap	Must supply own
Automatic Updates	Yes	Yes	No
Run without desktop?	Yes	No	Yes
Endorsed by Linus Torvalds	No	No	Yes

(See https://github.com/AppImage/AppImageKit/wiki/Similar-projects for a detailed if biased/old comparison)

Alternatives?

- Option 2: Package management, but do it right and make it distro-independent

Nix

Guix

Nix & Guix (Overview)

- Fully track **all** dependencies
- Never overwrite
  - Can never break working packages, absolutely zero dependency hell
- Reproducible, system-independent packages
- Available on all Linux distributions, as well as many other operating systems

Nix & Guix (Overview)

- Fixes the problems:
  - Dependencies all tracked
  - No bitrot (garbage collection, everything in store)
  - Developers can release default.nix files, and can even pin to specific versions/ check different nixpkgs versions.
- But also: breaks expectations, so applications need to be patched

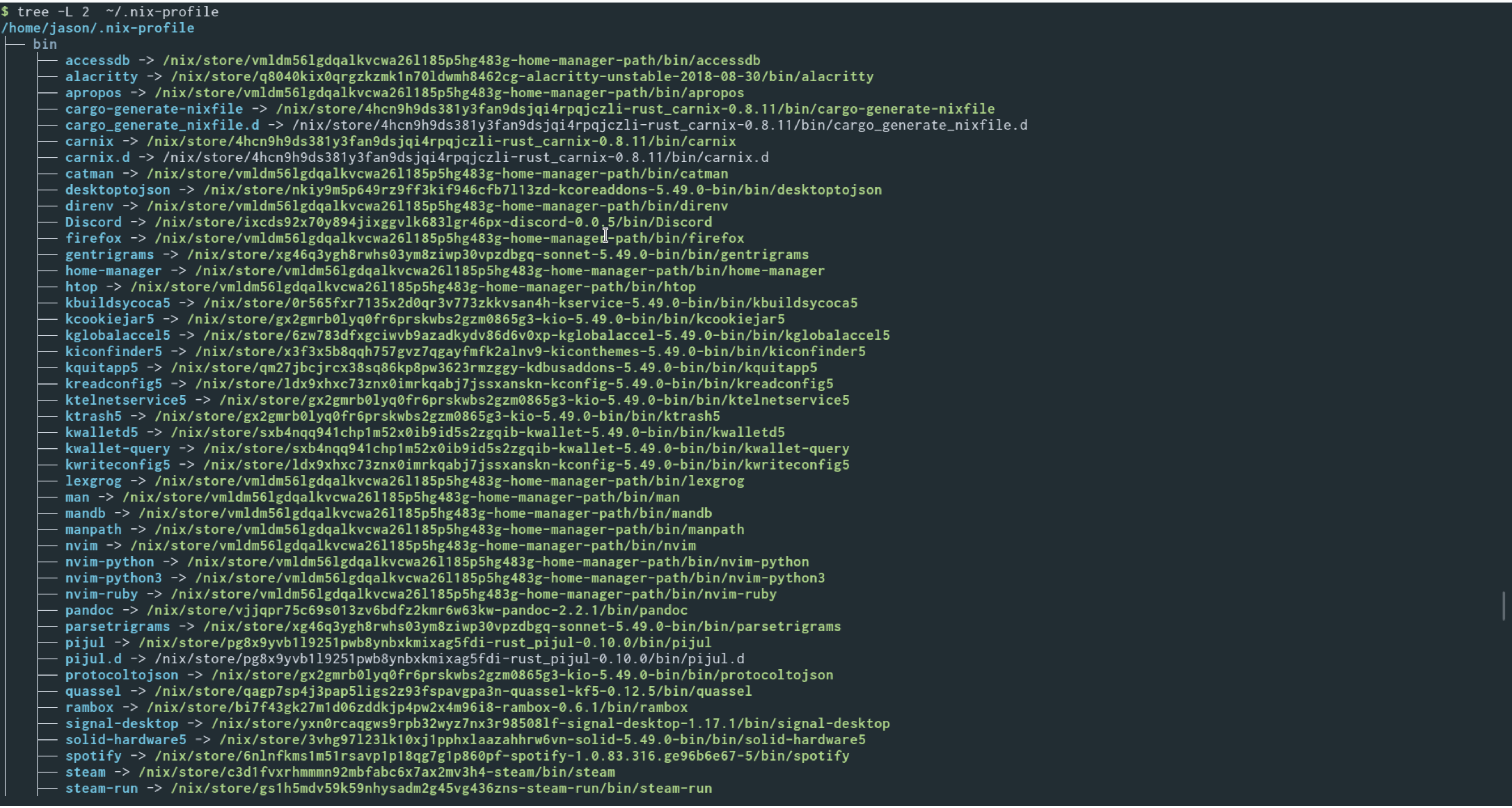
Why is this cool?

- No dependency hell, system-independent
- A lot of cool new features
  - Rootless installs
  - Install a package for the duration of a shell
  - Packages are expressions, not files
  - Bit-for-bit identical dev environments
  - Cache distributes binaries, can still patch your sources Gentoo-style, build with musl, etc

How it works

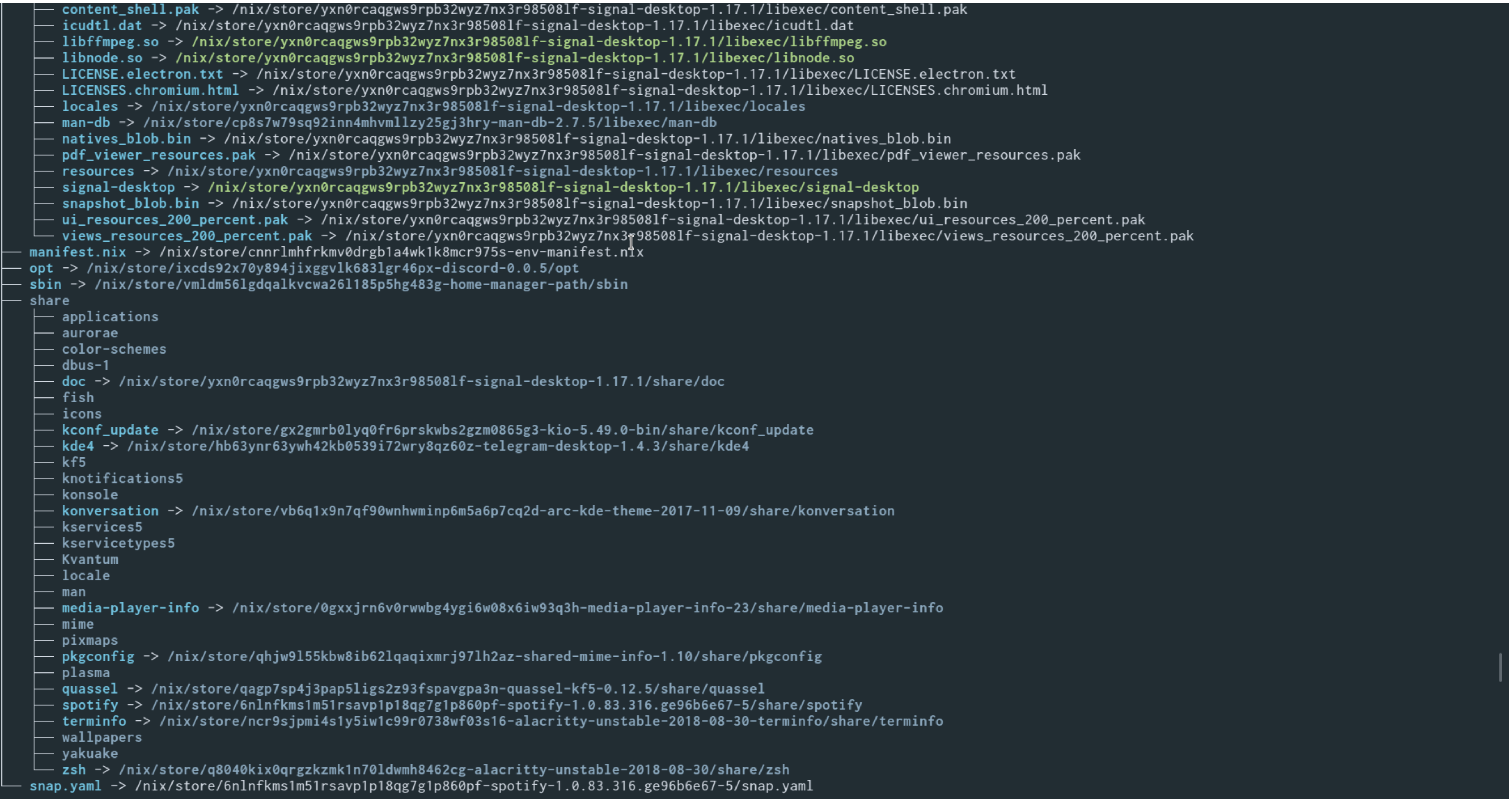
- When a derivation (package) is built, give it a unique name (hash of inputs)
- To change installed packages, link them into a profile (~/.nix-profile or /run/current-system/sw)
- Always use absolute paths, produce files in fixed format
  - Packages are just directories with /bin, /share, /etc, etc.
- Yes this means we patch binaries

My profile



```
$ tree -L 2  ~/.nix-profile
/home/jason/.nix-profile
├── bin
│   ├── accessdb -> /nix/store/vmldm56lgdqalkvcwa26l185p5hg483g-home-manager-path/bin/accessdb
│   ├── alacritty -> /nix/store/q8040kix0qrgzkzmk1n70ldwmh8462cg-alacritty-unstable-2018-08-30/bin/alacritty
│   ├── apropos -> /nix/store/vmldm56lgdqalkvcwa26l185p5hg483g-home-manager-path/bin/apropos
│   ├── cargo-generate-nixfile -> /nix/store/4hcn9h9ds381y3fan9dsjqi4rpqjczli-rust_carnix-0.8.11/bin/cargo-generate-nixfile
│   ├── cargo_generate_nixfile.d -> /nix/store/4hcn9h9ds381y3fan9dsjqi4rpqjczli-rust_carnix-0.8.11/bin/cargo_generate_nixfile.d
│   ├── carnix -> /nix/store/4hcn9h9ds381y3fan9dsjqi4rpqjczli-rust_carnix-0.8.11/bin/carnix
│   ├── carnix.d -> /nix/store/4hcn9h9ds381y3fan9dsjqi4rpqjczli-rust_carnix-0.8.11/bin/carnix.d
│   ├── catman -> /nix/store/vmldm56lgdqalkvcwa26l185p5hg483g-home-manager-path/bin/catman
│   ├── desktoptojson -> /nix/store/nkiy9m5p649rz9ff3kif946cfb7l13zd-kcoreaddons-5.49.0-bin/bin/desktoptojson
│   ├── direnv -> /nix/store/vmldm56lgdqalkvcwa26l185p5hg483g-home-manager-path/bin/direnv
│   ├── Discord -> /nix/store/ixcds92x70y894jixggvlk683lgr46px-discord-0.0.5/bin/Discord
│   ├── firefox -> /nix/store/vmldm56lgdqalkvcwa26l185p5hg483g-home-manager-path/bin/firefox
│   ├── gentrigrams -> /nix/store/xg46q3ygh8rwhs03ym8ziwp30vpzdbgq-sonnet-5.49.0-bin/bin/gentrigrams
│   ├── home-manager -> /nix/store/vmldm56lgdqalkvcwa26l185p5hg483g-home-manager-path/bin/home-manager
│   ├── htop -> /nix/store/vmldm56lgdqalkvcwa26l185p5hg483g-home-manager-path/bin/htop
│   ├── kbuildsycoca5 -> /nix/store/0r565fxr7135x2d0qr3v773zkkvsan4h-kservice-5.49.0-bin/bin/kbuildsycoca5
│   ├── kcookiejar5 -> /nix/store/gx2gmrb0lyq0fr6prskwbs2gzm0865g3-kio-5.49.0-bin/bin/kcookiejar5
│   ├── kglobalaccel5 -> /nix/store/6zw783dfxgciwvb9azadkydv86d6v0xp-kglobalaccel-5.49.0-bin/bin/kglobalaccel5
│   ├── kiconfinder5 -> /nix/store/x3f3x5b8qqh757gvz7qgayfmfk2alnv9-kiconthemes-5.49.0-bin/bin/kiconfinder5
│   ├── kquitapp5 -> /nix/store/qm27jbcjrcx38sq86kp8pw3623rmzggy-kdbusaddons-5.49.0-bin/bin/kquitapp5
│   ├── kreadconfig5 -> /nix/store/ldx9xhxc73znx0imrkqabj7jssxanskn-kconfig-5.49.0-bin/bin/kreadconfig5
│   ├── ktelnetservice5 -> /nix/store/gx2gmrb0lyq0fr6prskwbs2gzm0865g3-kio-5.49.0-bin/bin/ktelnetservice5
│   ├── ktrash5 -> /nix/store/gx2gmrb0lyq0fr6prskwbs2gzm0865g3-kio-5.49.0-bin/bin/ktrash5
│   ├── kwalletd5 -> /nix/store/sxb4nqq941chp1m52x0ib9id5s2zgqib-kwallet-5.49.0-bin/bin/kwalletd5
│   ├── kwallet-query -> /nix/store/sxb4nqq941chp1m52x0ib9id5s2zgqib-kwallet-5.49.0-bin/bin/kwallet-query
│   ├── kwriteconfig5 -> /nix/store/ldx9xhxc73znx0imrkqabj7jssxanskn-kconfig-5.49.0-bin/bin/kwriteconfig5
│   ├── lexgrog -> /nix/store/vmldm56lgdqalkvcwa26l185p5hg483g-home-manager-path/bin/lexgrog
│   ├── man -> /nix/store/vmldm56lgdqalkvcwa26l185p5hg483g-home-manager-path/bin/man
│   ├── mandb -> /nix/store/vmldm56lgdqalkvcwa26l185p5hg483g-home-manager-path/bin/mandb
│   ├── manpath -> /nix/store/vmldm56lgdqalkvcwa26l185p5hg483g-home-manager-path/bin/manpath
│   ├── nvim -> /nix/store/vmldm56lgdqalkvcwa26l185p5hg483g-home-manager-path/bin/nvim
│   ├── nvim-python -> /nix/store/vmldm56lgdqalkvcwa26l185p5hg483g-home-manager-path/bin/nvim-python
│   ├── nvim-python3 -> /nix/store/vmldm56lgdqalkvcwa26l185p5hg483g-home-manager-path/bin/nvim-python3
│   ├── nvim-ruby -> /nix/store/vmldm56lgdqalkvcwa26l185p5hg483g-home-manager-path/bin/nvim-ruby
│   ├── pandoc -> /nix/store/vjjqpr75c69s013zv6bdfz2kmr6w63kw-pandoc-2.2.1/bin/pandoc
│   ├── parsetrigrams -> /nix/store/xg46q3ygh8rwhs03ym8ziwp30vpzdbgq-sonnet-5.49.0-bin/bin/parsetrigrams
│   ├── pijul -> /nix/store/pg8x9yvb1l9251pwb8ynbxkmixag5fdi-rust_pijul-0.10.0/bin/pijul
│   ├── pijul.d -> /nix/store/pg8x9yvb1l9251pwb8ynbxkmixag5fdi-rust_pijul-0.10.0/bin/pijul.d
│   ├── protocoltojson -> /nix/store/gx2gmrb0lyq0fr6prskwbs2gzm0865g3-kio-5.49.0-bin/bin/protocoltojson
│   ├── quassel -> /nix/store/qagp7sp4j3pap5ligs2z93fspavgpa3n-quassel-kf5-0.12.5/bin/quassel
│   ├── rambox -> /nix/store/bi7f43gk27m1d06zddkjp4pw2x4m96i8-rambox-0.6.1/bin/rambox
│   ├── signal-desktop -> /nix/store/yxn0rcaqgws9rpb32wyz7nx3r98508lf-signal-desktop-1.17.1/bin/signal-desktop
│   ├── solid-hardware5 -> /nix/store/3vhg97l23lk10xj1pphxlaazahhrw6vn-solid-5.49.0-bin/bin/solid-hardware5
│   ├── spotify -> /nix/store/6nlnfkms1m51rsavp1p18qg7g1p860pf-spotify-1.0.83.316.ge96b6e67-5/bin/spotify
│   ├── steam -> /nix/store/c3d1fvxrhmmmn92mbfabc6x7ax2mv3h4-steam/bin/steam
│   ├── steam-run -> /nix/store/gs1h5mdv59k59nhysadm2g45vg436zns-steam-run/bin/steam-run
```

My profile



```
│   ├── content_shell.pak -> /nix/store/yxn0rcaqgws9rpb32wyz7nx3r98508lf-signal-desktop-1.17.1/libexec/content_shell.pak
│   ├── icudtl.dat -> /nix/store/yxn0rcaqgws9rpb32wyz7nx3r98508lf-signal-desktop-1.17.1/libexec/icudtl.dat
│   ├── libffmpeg.so -> /nix/store/yxn0rcaqgws9rpb32wyz7nx3r98508lf-signal-desktop-1.17.1/libexec/libffmpeg.so
│   ├── libnode.so -> /nix/store/yxn0rcaqgws9rpb32wyz7nx3r98508lf-signal-desktop-1.17.1/libexec/libnode.so
│   ├── LICENSE.electron.txt -> /nix/store/yxn0rcaqgws9rpb32wyz7nx3r98508lf-signal-desktop-1.17.1/libexec/LICENSE.electron.txt
│   ├── LICENSES.chromium.html -> /nix/store/yxn0rcaqgws9rpb32wyz7nx3r98508lf-signal-desktop-1.17.1/libexec/LICENSES.chromium.html
│   ├── locales -> /nix/store/yxn0rcaqgws9rpb32wyz7nx3r98508lf-signal-desktop-1.17.1/libexec/locales
│   ├── man-db -> /nix/store/cp8s7w79sq92inn4mhvmllzy25gj3hry-man-db-2.7.5/libexec/man-db
│   ├── natives_blob.bin -> /nix/store/yxn0rcaqgws9rpb32wyz7nx3r98508lf-signal-desktop-1.17.1/libexec/natives_blob.bin
│   ├── pdf_viewer_resources.pak -> /nix/store/yxn0rcaqgws9rpb32wyz7nx3r98508lf-signal-desktop-1.17.1/libexec/pdf_viewer_resources.pak
│   ├── resources -> /nix/store/yxn0rcaqgws9rpb32wyz7nx3r98508lf-signal-desktop-1.17.1/libexec/resources
│   ├── signal-desktop -> /nix/store/yxn0rcaqgws9rpb32wyz7nx3r98508lf-signal-desktop-1.17.1/libexec/signal-desktop
│   ├── snapshot_blob.bin -> /nix/store/yxn0rcaqgws9rpb32wyz7nx3r98508lf-signal-desktop-1.17.1/libexec/snapshot_blob.bin
│   ├── ui_resources_200_percent.pak -> /nix/store/yxn0rcaqgws9rpb32wyz7nx3r98508lf-signal-desktop-1.17.1/libexec/ui_resources_200_percent.pak
│   └── views_resources_200_percent.pak -> /nix/store/yxn0rcaqgws9rpb32wyz7nx3r98508lf-signal-desktop-1.17.1/libexec/views_resources_200_percent.pak
├── manifest.nix -> /nix/store/cnnrlmhfrkmv0drgb1a4wk1k8mcr975s-env-manifest.nix
├── opt -> /nix/store/ixcds92x70y894jixggvlk683lgr46px-discord-0.0.5/opt
├── sbin -> /nix/store/vmldm56lgdqalkvcwa26l185p5hg483g-home-manager-path/sbin
├── share
│   ├── applications
│   ├── aurorae
│   ├── color-schemes
│   ├── dbus-1
│   ├── doc -> /nix/store/yxn0rcaqgws9rpb32wyz7nx3r98508lf-signal-desktop-1.17.1/share/doc
│   ├── fish
│   ├── icons
│   ├── kconf_update -> /nix/store/gx2gmrb0lyq0fr6prskwbs2gzm0865g3-kio-5.49.0-bin/share/kconf_update
│   ├── kde4 -> /nix/store/hb63ynr63ywh42kb0539i72wry8qz60z-telegram-desktop-1.4.3/share/kde4
│   ├── kf5
│   ├── knotifications5
│   ├── konsole
│   ├── konversation -> /nix/store/vb6q1x9n7qf90wnhwminp6m5a6p7cq2d-arc-kde-theme-2017-11-09/share/konversation
│   ├── kservices5
│   ├── kservicetypes5
│   ├── Kvantum
│   ├── locale
│   ├── man
│   ├── media-player-info -> /nix/store/0gxxjrn6v0rwwbg4ygi6w08x6iw93q3h-media-player-info-23/share/media-player-info
│   ├── mime
│   ├── pixmaps
│   ├── pkgconfig -> /nix/store/qhjw9l55kbw8ib62lqaqixmrj97lh2az-shared-mime-info-1.10/share/pkgconfig
│   ├── plasma
│   ├── quassel -> /nix/store/qagp7sp4j3pap5ligs2z93fspavgpa3n-quassel-kf5-0.12.5/share/quassel
│   ├── spotify -> /nix/store/6nlnfkms1m51rsavp1p18qg7g1p860pf-spotify-1.0.83.316.ge96b6e67-5/share/spotify
│   ├── terminfo -> /nix/store/ncr9sjpmi4s1y5iw1c99r0738wf03s16-alacritty-unstable-2018-08-30-terminfo/share/terminfo
│   ├── wallpapers
│   ├── yakuake
│   └── zsh -> /nix/store/q8040kix0qrgzkzmk1n70ldwmh8462cg-alacritty-unstable-2018-08-30/share/zsh
└── snap.yaml -> /nix/store/6nlnfkms1m51rsavp1p18qg7g1p860pf-spotify-1.0.83.316.ge96b6e67-5/snap.yaml
```

NixOS

- Tl;dr Nix works for packages, why not make the entire system a package?
  - System configuration version chosen at boot,
  - Get all the same benefits, can send system configurations over network, isolation
  - Instant, nearly* atomic switches
    - E.g. can shut down during update

*Services need to be restarted and this may take some time

NixOS

- Downsides:
  - Can’t run binaries from the internet without patching
    - There are binaries in npm/maven/etc...
    - In Nixpkgs we use a tool called patchelf to fix them up
    - Still have snap/flatpak/appimage though
  - Still need /usr/bin/env and /bin/sh to make shebangs work reasonably well

What about Guix?

- Some people wanted in on Nix but it wasn’t free enough for them
  - Uses exclusively (and I mean it) Guile Scheme
  - Only free software(/firmware for GuixSD) https://www.gnu.org/distros/free-distros.en.html
  - Much younger than Nix, so generally fewer packages

Questions?

Low-level comparison of Nix and Guix

	Nix(OS)	Guix(SD)
Package language	Nix, Bash	Guile Scheme
Implementation language	C++	Guile Scheme (again)
Freeness (GNU-style)	Optionally free (`allowUnfree = true;`)	Fully free (no nonfree packages)
Supported Environments	Officially: Linux, Darwin (MacOS) Somewhat: Cygwin, BSD, Android, iPhone, RaspPi, Solaris NixOS uses Linux	Linux, GNU Hurd GuixSD uses Linux-libre, someday Hurd too https://www.gnu.org/software/guix/blog/2015/porting-guix-and-guixsd/
System Daemon	systemd	GNU Shephard (previously known as GNU dmd)
License	MIT/X11	GPLv3
Store location	/nix	/gnu

Other random nix details: Tips

- Manuals:
  https://nixos.org/nix/manual/
  https://nixos.org/nixos/manual/
- Can search packages/options:
  https://nixos.org/nixos/packages.html
  (or use `nix search` from a terminal)
  https://nixos.org/nixos/options.html
- Irc: #nixos for both nixpkgs & nixos. They’re quite helpful
- Arch wiki is very useful if you can translate

Other random nix details: Release structure for nixpkgs

- Two serious options and some foot-shooting options:
  - Nixpkgs stable – biyearly releases, tested automatically, definitely binary distribution
  - Nixpkgs unstable – rolling, tested automatically, could be source or binary
  - The git repo – no tests, usually from source
  - Somebody else’s unmerged branch. Sometimes you really want Pantheon DE

Other random nix details: Tips

- In my daily usage, I use a combination of desktop applications in my environment / system
  - Mixed repositories: stable + unstable + random other sources (anyone with the right files can give you packages)
  - System is on stable

Other random nix details: Workflow/Philosophy for Devs

- It’s not good practice to keep development tools in your global environment:
  - (e.g. g++ is any version, there’s no g++-8, no python2/python3, so pick them when you need them)
- I use direnv w/ nix-shell, so in a project directory it pulls in everything automatically
  - Aliases make this very fast