What is Nix?

… and Nixpkgs, NixOS, nix-env, nix-shell, flakes, …

stick@satoshilabs.com

SATOSHILABS

TREZOR

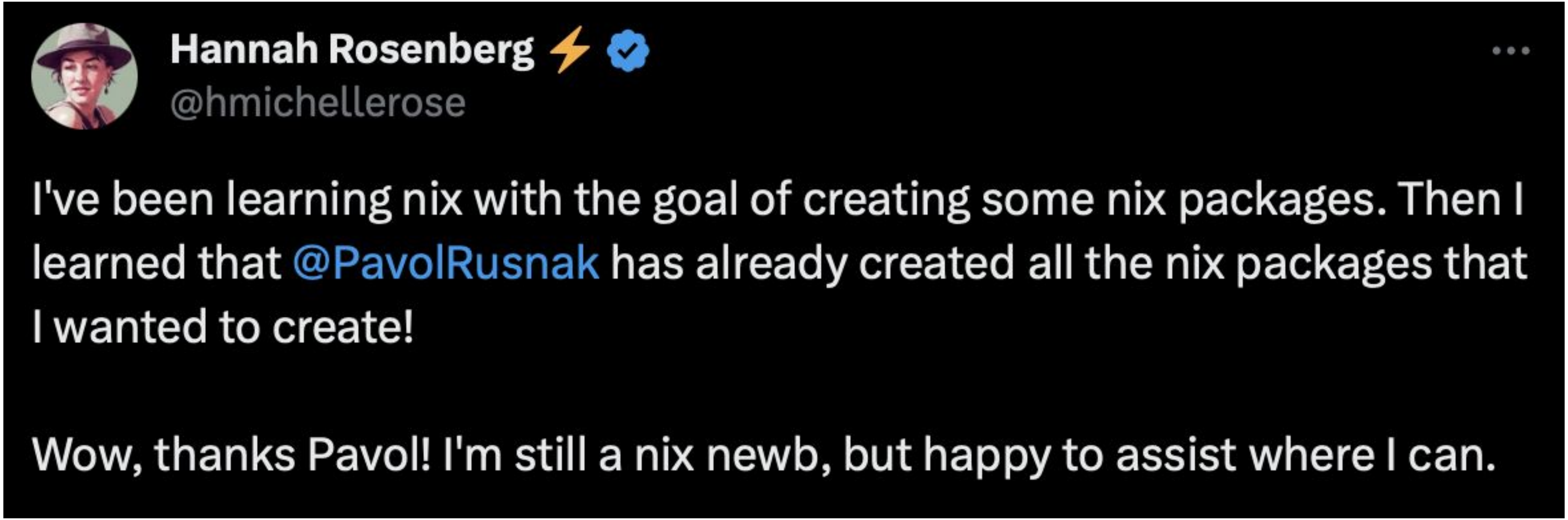
+1nvity

tropic square

vexl

SatoshiLabs ❤️ Nix

- SatoshiLabs server infrastructure
- Trezor server infrastructure
- Invity server infrastructure
- Trezor hardware wallet firmware reproducible builds
- company laptops (NixOS, Ubuntu and macOS)



Hannah Rosenberg

@hmichellerose

I've been learning nix with the goal of creating some nix packages. Then I learned that @PavolRusnak has already created all the nix packages that I wanted to create!

Wow, thanks Pavol! I'm still a nix newb, but happy to assist where I can.

Nix: What?

Nix

- purely functional package manager
    - custom lazy functional programming language

Nix

does not follow FHS (Filesystem Hierarchy Standard)

- executables in `/bin`, `/usr/bin`
- libraries in `/lib`, `/usr/lib`

packages stored in `/nix/store`

- `/nix/store/b6gvzjyb2pg0kjfwrjmg1vfhh54ad73z-firefox-33.1/{bin,lib}`
- `b6gv...d73z` is a cryptographic hash of the package’s build dependency graph
- hardcoded `rpaths` in binaries

Nix

- complete dependencies
- multi-user support
- multiple versions of the same package

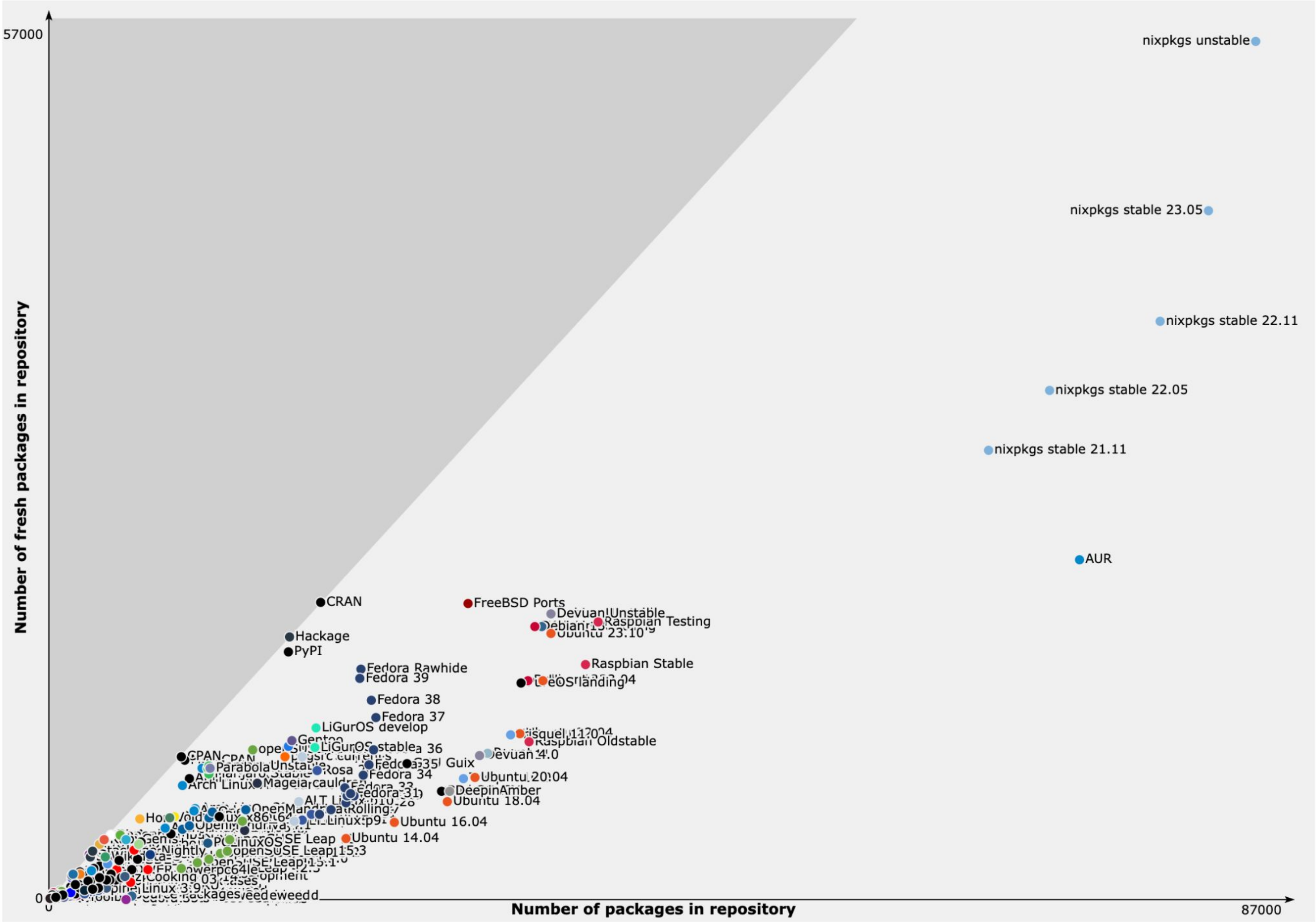
nix-channel

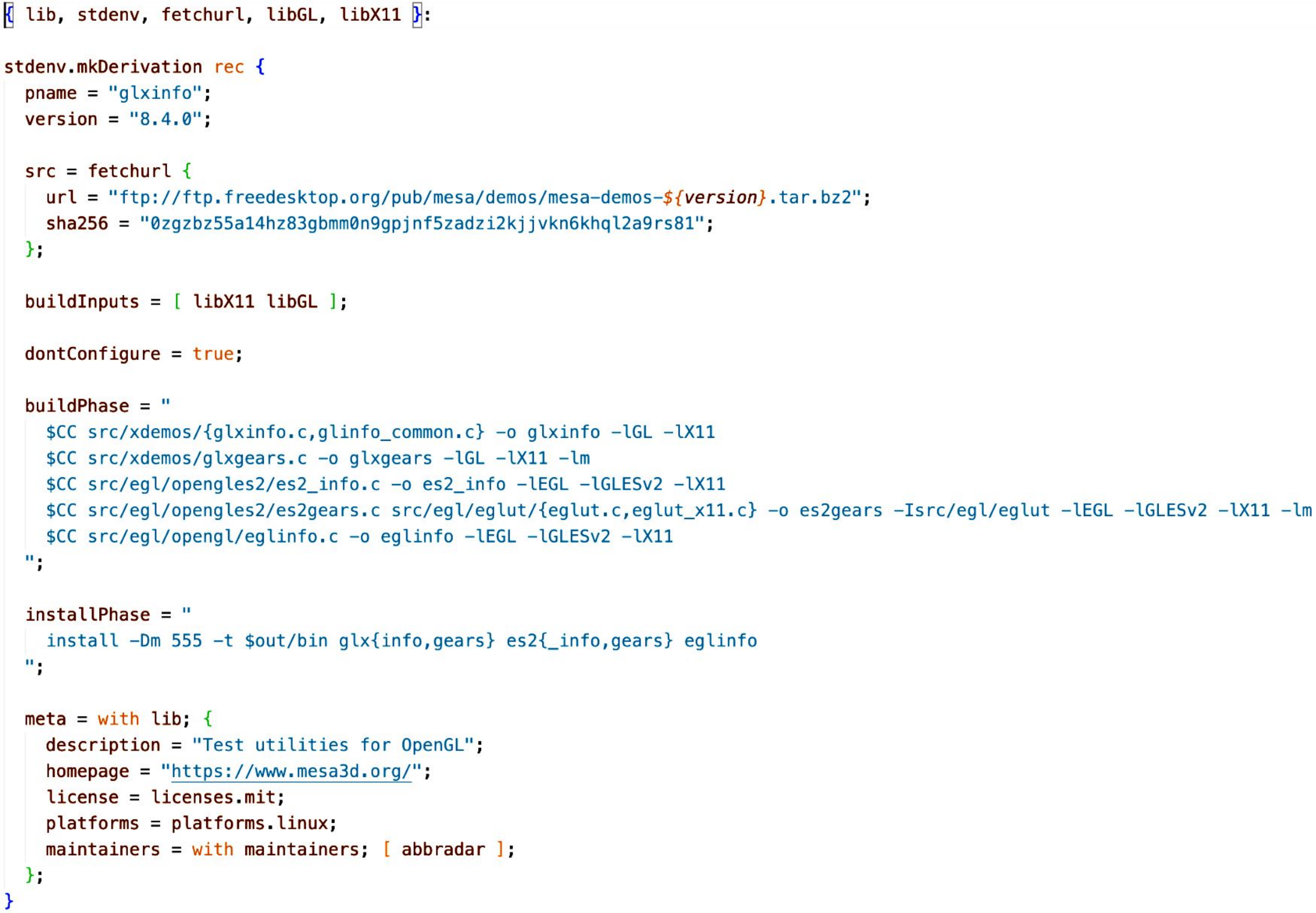
```
$ nix-channel --list
nixpkgs https://nixos.org/channels/nixpkgs-unstable
```

nixpkgs
intermezzo

Nixpkgs

- Nix Packages
- set of Nix expressions packaging software
- the world's largest software collection
- not only Linux
    - but also macOS, *BSD, Windows, etc.


57000
0
0
87000
Number of fresh packages in repository
Number of packages in repository
nixpkgs unstable
nixpkgs stable 23.05
nixpkgs stable 22.11
nixpkgs stable 22.05
nixpkgs stable 21.11
AUR
CRAN
FreeBSD Ports
Devuan Unstable
Raspbian Testing
Ubuntu 23.10
Hackage
PyPI
Fedora Rawhide
Fedora 39
Raspbian Stable
Fedora 38
Fedora 37
LiGurOS develop
Raspbian Oldstable
Gentoo
LiGurOS stable
Devuan 4.0
CPAN
Parabola
Rosa
Fedora 34
Guix
Ubuntu 20.04
Arch Linux
Mageia cauldron
Fedora 31
DeepinAmber
Ubuntu 18.04
ALT Linux
OpenMandriva
Ubuntu 16.04
Ubuntu 14.04
openSUSE Leap 15.3
openSUSE Leap 15.1
PCLinuxOS
Wikidata
Nightly
Alpine Linux 3.9



```
{ lib, stdenv, fetchurl, libGL, libX11 }:

stdenv.mkDerivation rec {
  pname = "glxinfo";
  version = "8.4.0";

  src = fetchurl {
    url = "ftp://ftp.freedesktop.org/pub/mesa/demos/mesa-demos-${version}.tar.bz2";
    sha256 = "0zgzbz55a14hz83gbmm0n9gpjnf5zadzi2kjjvkn6khql2a9rs81";
  };

  buildInputs = [ libX11 libGL ];

  dontConfigure = true;

  buildPhase = "
    $CC src/xdemos/{glxinfo.c,glinfo_common.c} -o glxinfo -lGL -lX11
    $CC src/xdemos/glxgears.c -o glxgears -lGL -lX11 -lm
    $CC src/egl/opengles2/es2_info.c -o es2_info -lEGL -lGLESv2 -lX11
    $CC src/egl/opengles2/es2gears.c src/egl/eglut/{eglut.c,eglut_x11.c} -o es2gears -Isrc/egl/eglut -lEGL -lGLESv2 -lX11 -lm
    $CC src/egl/opengl/eglinfo.c -o eglinfo -lEGL -lGLESv2 -lX11
  ";

  installPhase = "
    install -Dm 555 -t $out/bin glx{info,gears} es2{_info,gears} eglinfo
  ";

  meta = with lib; {
    description = "Test utilities for OpenGL";
    homepage = "https://www.mesa3d.org/";
    license = licenses.mit;
    platforms = platforms.linux;
    maintainers = with maintainers; [ abbradar ];
  };
}
```

end of nixpkgs intermezzo

nix-env

```
$ nix-channel --list
nixpkgs https://nixos.org/channels/nixpkgs-unstable

$ nix-env -q
nix-2.11.0
nss-cacert-3.83
```

nix-env

```
$ nix-channel --list
nixpkgs https://nixos.org/channels/nixpkgs-unstable

$ nix-env -q
nix-2.11.0
nss-cacert-3.83

$ nix-env -iA nixpkgs.fish
installing 'fish-3.5.1'
copying path '/nix/store/84qh7r0j8vi81d3fypf6c9ck733saxj2-fish-3.5.1' from 'https://cache.nixos.org'...
building '/nix/store/8xrals5djwzpb1c5rfbz1a12k2xjp089-user-environment.drv'...
```

all packages built by hydra.nixos.org (Nix CI)

are cached on cache.nixos.org

(forever*) 🚀

nix-env

```
$ nix-channel --list
nixpkgs https://nixos.org/channels/nixpkgs-unstable

$ nix-env -q
nix-2.11.0
nss-cacert-3.83

$ nix-env -iA nixpkgs.fish
installing 'fish-3.5.1'
copying path '/nix/store/84qh7r0j8vi81d3fypf6c9ck733saxj2-fish-3.5.1' from 'https://cache.nixos.org'...
building '/nix/store/8xrals5djwzpb1c5rfbz1a12k2xjp089-user-environment.drv'...

$ which fish
/Users/stick/.nix-profile/bin/fish
```

nix-env

```
$ nix-channel --list
nixpkgs https://nixos.org/channels/nixpkgs-unstable

$ nix-env -q
nix-2.11.0
nss-cacert-3.83

$ nix-env -iA nixpkgs.fish
installing 'fish-3.5.1'
copying path '/nix/store/84qh7r0j8vi81d3fypf6c9ck733saxj2-fish-3.5.1' from 'https://cache.nixos.org'...
building '/nix/store/8xrals5djwzpb1c5rfbz1a12k2xjp089-user-environment.drv'...

$ which fish
/Users/stick/.nix-profile/bin/fish

$ echo $PATH
/Users/stick/.nix-profile/bin:...
```

nix-env

```
$ nix-channel --list
nixpkgs https://nixos.org/channels/nixpkgs-unstable

$ nix-env -q
nix-2.11.0
nss-cacert-3.83

$ nix-env -iA nixpkgs.fish
installing 'fish-3.5.1'
copying path '/nix/store/84qh7r0j8vi81d3fypf6c9ck733saxj2-fish-3.5.1' from 'https://cache.nixos.org'...
building '/nix/store/8xrals5djwzpb1c5rfbz1a12k2xjp089-user-environment.drv'...

$ which fish
/Users/stick/.nix-profile/bin/fish

$ echo $PATH
/Users/stick/.nix-profile/bin:...

$ fish
🐟
```

nix-env

```
$ ls -l /Users/stick/.nix-profile

… -> /nix/var/nix/profiles/per-user/stick/profile
```

nix-env

```
$ ls -l /Users/stick/.nix-profile

… -> /nix/var/nix/profiles/per-user/stick/profile

$ ls -l /nix/var/nix/profiles/per-user/stick/profile

… -> profile-22-link
```

nix-env

```
$ ls -l /Users/stick/.nix-profile

… -> /nix/var/nix/profiles/per-user/stick/profile

$ ls -l /nix/var/nix/profiles/per-user/stick/profile

… -> profile-22-link

$ ls -l /nix/var/nix/profiles/per-user/stick/profile-22-link

… -> /nix/store/jamx66gjjpx1i73ha7sqmhxasxy1qgp7-user-environment
```

nix-env

```
$ ls -l /Users/stick/.nix-profile

… -> /nix/var/nix/profiles/per-user/stick/profile

$ ls -l /nix/var/nix/profiles/per-user/stick/profile

… -> profile-22-link

$ ls -l /nix/var/nix/profiles/per-user/stick/profile-22-link

… -> /nix/store/jamx66gjjpx1i73ha7sqmhxasxy1qgp7-user-environment

$ ls -l /nix/store/jamx66gjjpx1i73ha7sqmhxasxy1qgp7-user-environment/bin

… fish -> /nix/store/pldynq1amrmvsdyqlvv87d08srkc8i52-fish-3.6.1/bin/fish
… nix -> /nix/store/0cllbk3shzz2yqypw72nvh6z7fj224wb-nix-2.18.0/bin/nix
… nix-build -> /nix/store/0cllbk3shzz2yqypw72nvh6z7fj224wb-nix-2.18.0/bin/nix-build
… nix-channel -> /nix/store/0cllbk3shzz2yqypw72nvh6z7fj224wb-nix-2.18.0/bin/nix-channel
… nix-env -> /nix/store/0cllbk3shzz2yqypw72nvh6z7fj224wb-nix-2.18.0/bin/nix-env
… nix-hash -> /nix/store/0cllbk3shzz2yqypw72nvh6z7fj224wb-nix-2.18.0/bin/nix-hash
… nix-shell -> /nix/store/0cllbk3shzz2yqypw72nvh6z7fj224wb-nix-2.18.0/bin/nix-shell
```

SYMLINK
MAGIC

Nix

- atomic upgrades and rollbacks
  - `profile-22-link`
  - `profile-23-link`
  - etc.
- garbage collection
  - `nix-collect-garbage`
  - removes all unreachable stuff from `/nix/store`

nix-env

```
$ nix-channel --list
nixpkgs https://nixos.org/channels/nixpkgs-unstable

$ nix-env -q
nix-2.11.0
nss-cacert-3.83

$ nix-env -iA nixpkgs.fish
installing 'fish-3.5.1'
copying path '/nix/store/84qh7r0j8vi81d3fypf6c9ck733saxj2-fish-3.5.1' from 'https://cache.nixos.org'...
building '/nix/store/8xrals5djwzpb1c5rfbz1a12k2xjp089-user-environment.drv'...

$ which fish
/Users/stick/.nix-profile/bin/fish

$ echo $PATH
/Users/stick/.nix-profile/bin:...

$ fish
🐟
```

nix-shell

```
$ nix-shell -p fish xz

> echo $PATH

/nix/store/pldynq1amrmvsdyqlvv87d08srkc8i52-fish-3.6.1/bin:/nix/store/nws79v3a
sz8a9cyyn8f76kwykjr6lhkb-xz-5.4.4-bin/bin:...
```

shell.nix

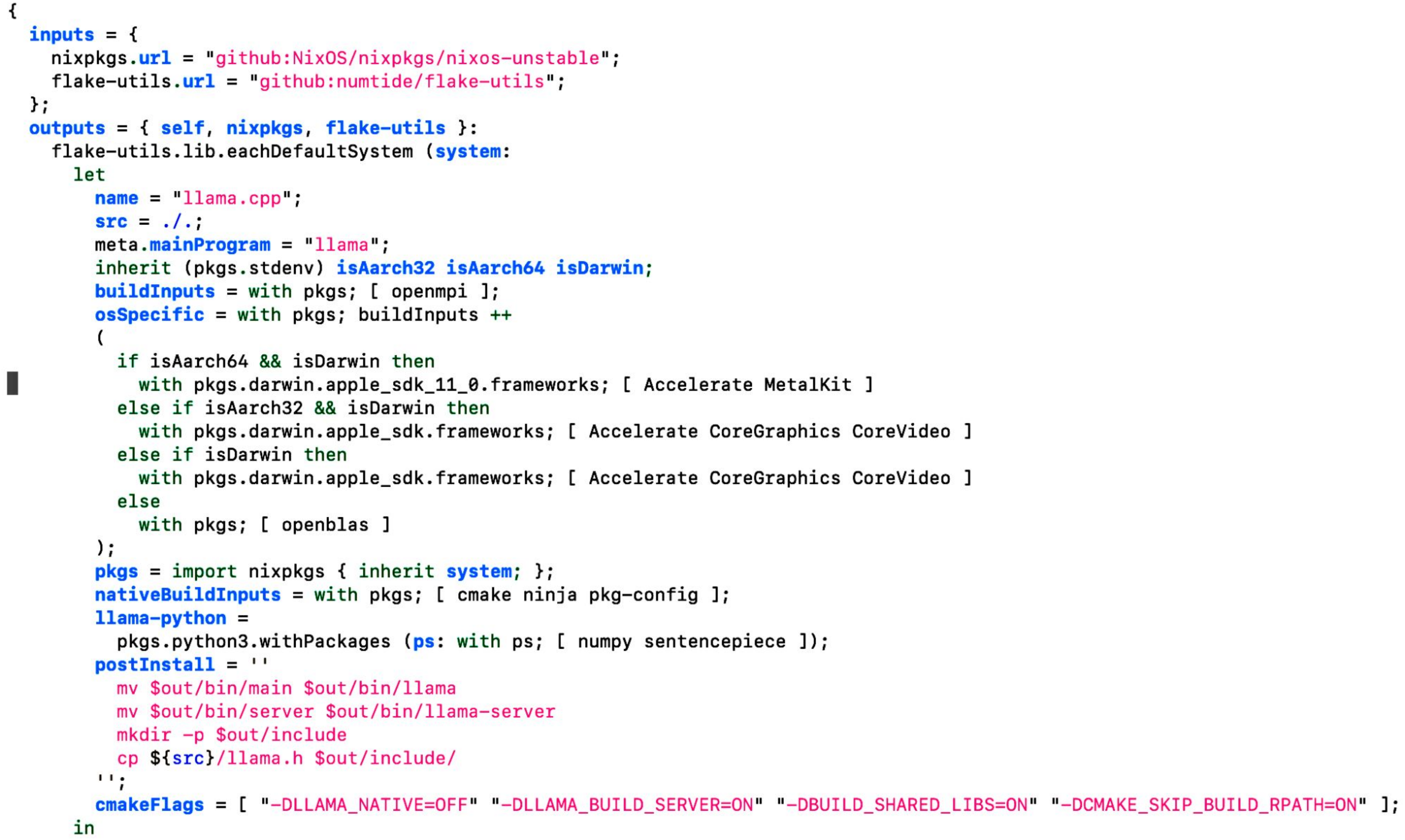
```
with import <nixpkgs> {};
mkShell {
  buildInputs = [
    bzip2
    go
    lz4
    pkg-config
    rocksdb
    snappy
    zeromq
    zlib
  ];
  shellHook = ''
    export CGO_LDFLAGS="-lrocksdb -lz -lbz2 -lsnappy -llz4 -lm -lstdc++"
  '';
}
```

shell.nix

```
let nixpkgs = import (builtins.fetchTarball {
    url = "https://github.com/NixOS/nixpkgs/archive/
           c58e6fbf258df1572b535ac1868ec42faf7675dd.tar.gz";
    sha256 = "18pna0yinvdprhhcmhyanlgrmgf81nwpc0j2z9fy9mc8cqkx3937";
  })

in

with nixpkgs {};

mkShell {

...

}
```

Flakes

```
flake.nix
```



```
{
  inputs = {
    nixpkgs.url = "github:NixOS/nixpkgs/nixos-unstable";
    flake-utils.url = "github:numtide/flake-utils";
  };
  outputs = { self, nixpkgs, flake-utils }:
    flake-utils.lib.eachDefaultSystem (system:
      let
        name = "llama.cpp";
        src = ./.;
        meta.mainProgram = "llama";
        inherit (pkgs.stdenv) isAarch32 isAarch64 isDarwin;
        buildInputs = with pkgs; [ openmpi ];
        osSpecific = with pkgs; buildInputs ++
        (
          if isAarch64 && isDarwin then
            with pkgs.darwin.apple_sdk_11_0.frameworks; [ Accelerate MetalKit ]
          else if isAarch32 && isDarwin then
            with pkgs.darwin.apple_sdk.frameworks; [ Accelerate CoreGraphics CoreVideo ]
          else if isDarwin then
            with pkgs.darwin.apple_sdk.frameworks; [ Accelerate CoreGraphics CoreVideo ]
          else
            with pkgs; [ openblas ]
        );
        pkgs = import nixpkgs { inherit system; };
        nativeBuildInputs = with pkgs; [ cmake ninja pkg-config ];
        llama-python =
          pkgs.python3.withPackages (ps: with ps; [ numpy sentencepiece ]);
        postInstall = ''
          mv $out/bin/main $out/bin/llama
          mv $out/bin/server $out/bin/llama-server
          mkdir -p $out/include
          cp ${src}/llama.h $out/include/
        '';
        cmakeFlags = [ "-DLLAMA_NATIVE=OFF" "-DLLAMA_BUILD_SERVER=ON" "-DBUILD_SHARED_LIBS=ON" "-DCMAKE_SKIP_BUILD_RPATH=ON" ];
      in
```

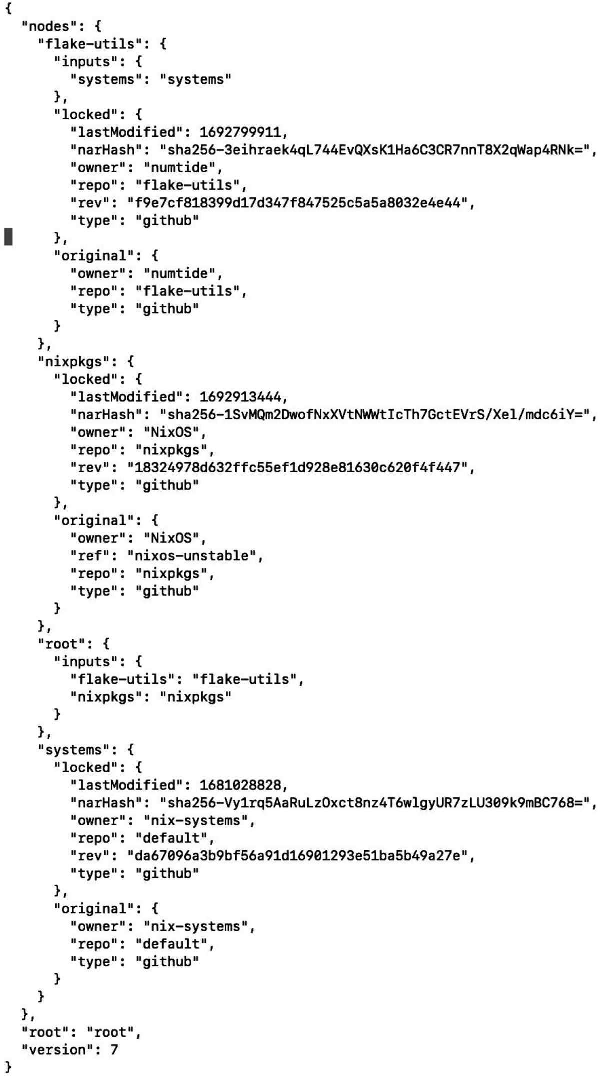
Flakes

`flake.nix`

`flake.lock`

Flakes

```
nix flake init

nix flake lock

nix flake update
```

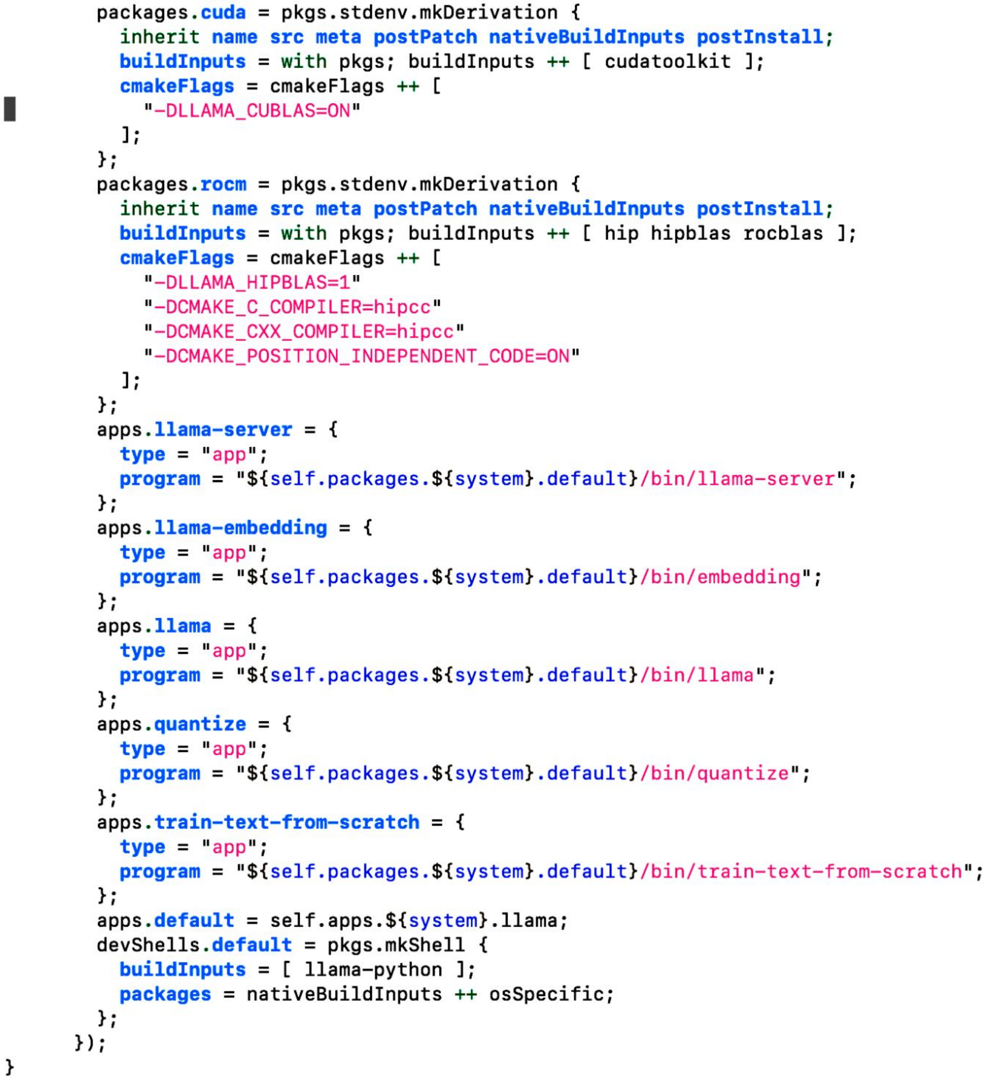


```
{
  "nodes": {
    "flake-utils": {
      "inputs": {
        "systems": "systems"
      },
      "locked": {
        "lastModified": 1692799911,
        "narHash": "sha256-3eihraek4qL744EvQXsK1Ha6C3CR7nnT8X2qWap4RNk=",
        "owner": "numtide",
        "repo": "flake-utils",
        "rev": "f9e7cf818399d17d347f847525c5a5a8032e4e44",
        "type": "github"
      },
      "original": {
        "owner": "numtide",
        "repo": "flake-utils",
        "type": "github"
      }
    },
    "nixpkgs": {
      "locked": {
        "lastModified": 1692913444,
        "narHash": "sha256-1SvMQm2DwofNxXVtNWWtIcTh7GctEVrS/Xel/mdc6iY=",
        "owner": "NixOS",
        "repo": "nixpkgs",
        "rev": "18324978d632ffc55ef1d928e81630c620f4f447",
        "type": "github"
      },
      "original": {
        "owner": "NixOS",
        "ref": "nixos-unstable",
        "repo": "nixpkgs",
        "type": "github"
      }
    },
    "root": {
      "inputs": {
        "flake-utils": "flake-utils",
        "nixpkgs": "nixpkgs"
      }
    },
    "systems": {
      "locked": {
        "lastModified": 1681028828,
        "narHash": "sha256-Vy1rq5AaRuLzOxct8nz4T6wlgyUR7zLU309k9mBC768=",
        "owner": "nix-systems",
        "repo": "default",
        "rev": "da67096a3b9bf56a91d16901293e51ba5b49a27e",
        "type": "github"
      },
      "original": {
        "owner": "nix-systems",
        "repo": "default",
        "type": "github"
      }
    }
  },
  "root": "root",
  "version": 7
}
```

Flakes

```
nix run
```

```
nix develop
```



```
        packages.cuda = pkgs.stdenv.mkDerivation {
          inherit name src meta postPatch nativeBuildInputs postInstall;
          buildInputs = with pkgs; buildInputs ++ [ cudatoolkit ];
          cmakeFlags = cmakeFlags ++ [
            "-DLLAMA_CUBLAS=ON"
          ];
        };
        packages.rocm = pkgs.stdenv.mkDerivation {
          inherit name src meta postPatch nativeBuildInputs postInstall;
          buildInputs = with pkgs; buildInputs ++ [ hip hipblas rocblas ];
          cmakeFlags = cmakeFlags ++ [
            "-DLLAMA_HIPBLAS=1"
            "-DCMAKE_C_COMPILER=hipcc"
            "-DCMAKE_CXX_COMPILER=hipcc"
            "-DCMAKE_POSITION_INDEPENDENT_CODE=ON"
          ];
        };
        apps.llama-server = {
          type = "app";
          program = "${self.packages.${system}.default}/bin/llama-server";
        };
        apps.llama-embedding = {
          type = "app";
          program = "${self.packages.${system}.default}/bin/embedding";
        };
        apps.llama = {
          type = "app";
          program = "${self.packages.${system}.default}/bin/llama";
        };
        apps.quantize = {
          type = "app";
          program = "${self.packages.${system}.default}/bin/quantize";
        };
        apps.train-text-from-scratch = {
          type = "app";
          program = "${self.packages.${system}.default}/bin/train-text-from-scratch";
        };
        apps.default = self.apps.${system}.llama;
        devShells.default = pkgs.mkShell {
          buildInputs = [ llama-python ];
          packages = nativeBuildInputs ++ osSpecific;
        };
      });
}
```

Flakes

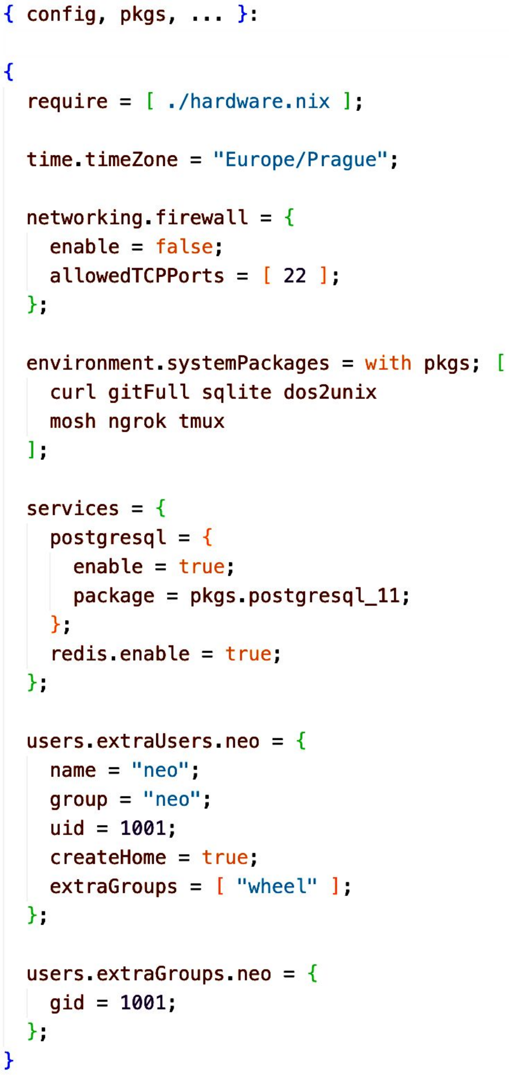
```
nix run

nix run .

nix run .#llama

nix run .#quantize
```

Flakes

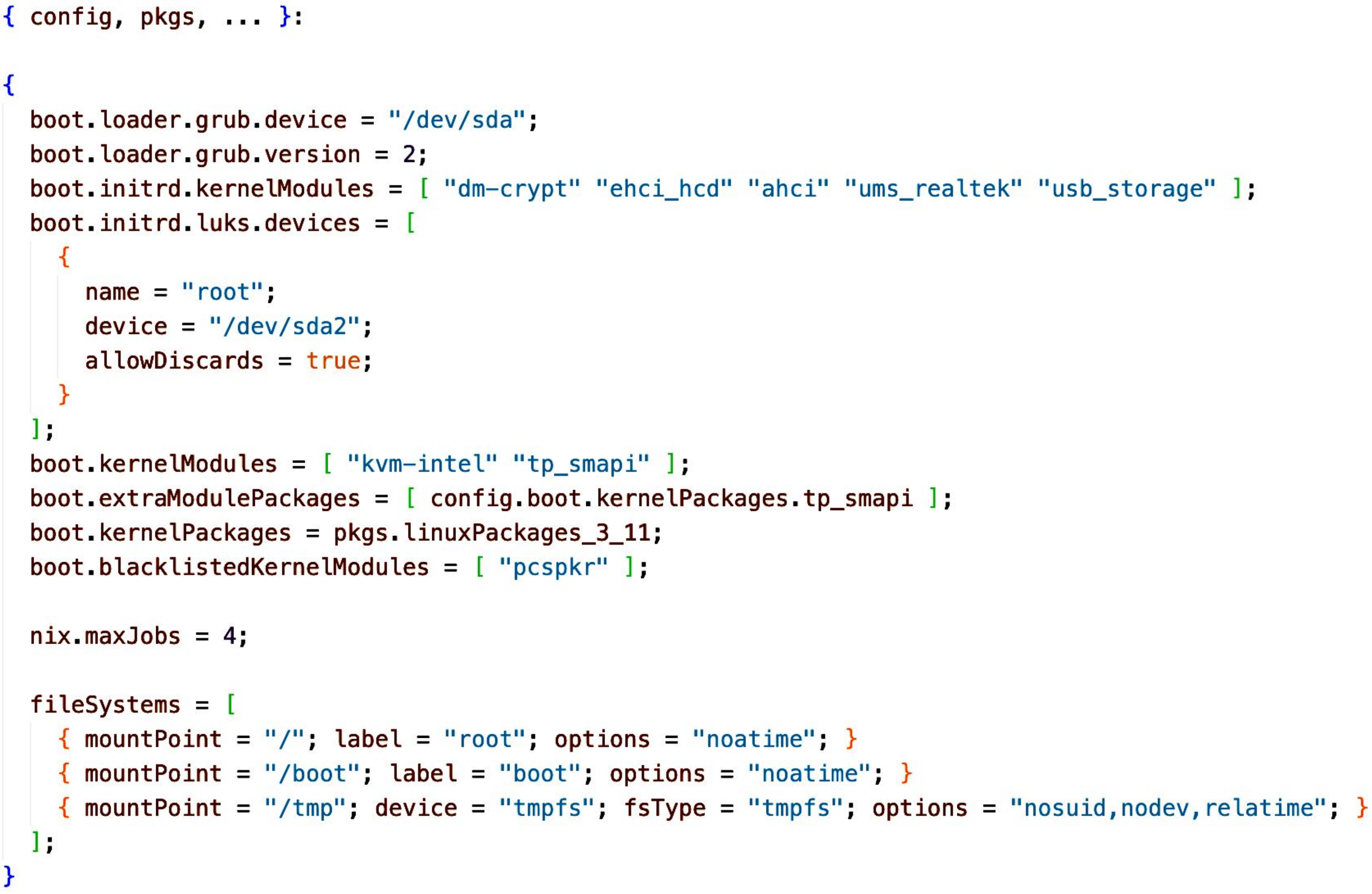
```
nix run github:ggerganov/llama.cpp

nix run github:ggerganov/llama.cpp#llama

nix run github:ggerganov/llama.cpp#quantize
```

NixOS

- a Linux distribution based on Nix
- not Not-invented-here syndrome
    - systemd, vanilla, patches → upstream
- not only package management
- but also system configuration
    - `/etc/nixos/configuration.nix`
    - `/etc/nixos/hardware.nix`



```
{ config, pkgs, ... }:

{
  require = [ ./hardware.nix ];

  time.timeZone = "Europe/Prague";

  networking.firewall = {
    enable = false;
    allowedTCPPorts = [ 22 ];
  };

  environment.systemPackages = with pkgs; [
    curl gitFull sqlite dos2unix
    mosh ngrok tmux
  ];

  services = {
    postgresql = {
      enable = true;
      package = pkgs.postgresql_11;
    };
    redis.enable = true;
  };

  users.extraUsers.neo = {
    name = "neo";
    group = "neo";
    uid = 1001;
    createHome = true;
    extraGroups = [ "wheel" ];
  };

  users.extraGroups.neo = {
    gid = 1001;
  };
}
```



```
{ config, pkgs, ... }:

{
  boot.loader.grub.device = "/dev/sda";
  boot.loader.grub.version = 2;
  boot.initrd.kernelModules = [ "dm-crypt" "ehci_hcd" "ahci" "ums_realtek" "usb_storage" ];
  boot.initrd.luks.devices = [
    {
      name = "root";
      device = "/dev/sda2";
      allowDiscards = true;
    }
  ];
  boot.kernelModules = [ "kvm-intel" "tp_smapi" ];
  boot.extraModulePackages = [ config.boot.kernelPackages.tp_smapi ];
  boot.kernelPackages = pkgs.linuxPackages_3_11;
  boot.blacklistedKernelModules = [ "pcspkr" ];

  nix.maxJobs = 4;

  fileSystems = [
    { mountPoint = "/"; label = "root"; options = "noatime"; }
    { mountPoint = "/boot"; label = "boot"; options = "noatime"; }
    { mountPoint = "/tmp"; device = "tmpfs"; fsType = "tmpfs"; options = "nosuid,nodev,relatime"; }
  ];
}
```

Morph

- NixOS deployment tool
- https://github.com/DBCDK/morph
- fancy wrapper around nix-build, nix-copy, nix-env
- health checks

```
let
  pkgs = import (import ../nixpkgs.nix) {};
in
{
  network =  {
    inherit pkgs;
    description = "simple hosts";
    ordering = {
      tags = [ "db" "web" ];
    };
  };

  "web01" = { config, pkgs, ... }: {
    deployment.tags = [ "web" ];

    boot.loader.systemd-boot.enable = true;

    services.nginx.enable = true;

    fileSystems = {
        "/" = { label = "nixos"; fsType = "ext4"; };
        "/boot" = { label = "boot"; fsType = "vfat"; };
    };
  };

  "db01" = { config, pkgs, ... }: {
    deployment.tags = [ "db" ];

    boot.loader.systemd-boot.enable = true;

    services.postgresql.enable = true;

    fileSystems = {
        "/" = { label = "nixos"; fsType = "ext4"; };
        "/boot" = { label = "boot"; fsType = "vfat"; };
    };
  };
}
```

Morph

- modules
    - {lnbits, lnd, bitcoind, …}
- roles
    - modules + secrets
- hardware
    - {ovirt, vps, physical, …}
- profiles
    - roles + health checks + hardware
- infra
    - [profile]

Other options for Nix deployment

- https://github.com/krebs/krops
- https://github.com/zhaofengli/colmena
- https://github.com/NixOS/nixops
    - can also do “physical” provisioning

Guix

- NixOS inspired GNU/Linux OS by the GNU project
- uses Guile Scheme instead of Nix
- https://guix.gnu.org
- https://bootstrappable.org

```
(define-public libssh2
  (package
   (name "libssh2")
   (version "1.10.0")
   (source (origin
            (method url-fetch)
            (uri (string-append
                   "https://www.libssh2.org/download/libssh2-"
                   version ".tar.gz"))
            (sha256
             (base32
              "0l8xwhhscvss7q007vpbkbv7jh9s43579rx2sf8lnfgd7l7yjr1d"))))
   (build-system gnu-build-system)
   ;; The installed libssh2.pc file does not include paths to libgcrypt and
   ;; zlib libraries, so we need to propagate the inputs.
   (propagated-inputs (list libgcrypt zlib))
   (arguments
    (list #:configure-flags #~'("--with-libgcrypt"
                                "--disable-static")))
   (synopsis "Client-side C library implementing the SSH2 protocol")
   (description
    "libssh2 is a library intended to allow software developers access to
the SSH-2 protocol in an easy-to-use self-contained package.  It can be built
into an application to perform many different tasks when communicating with
a server that supports the SSH-2 protocol.")
   (license license:bsd-3)
   (home-page "https://www.libssh2.org/")))
```

Resources

https://nixos.org/learn

https://nixos.org/guides/nix-pills/

https://nix.dev/

https://discourse.nixos.org/

https://github.com/nixos

Thank you!

stick@satoshilabs.com