Convert `configuration.nix` to be a flake

A problem with the default NixOS `configuration.nix` generated by the official installer is that it is not “pure” and thus not reproducible (see here), as it still uses a mutable Nix channel (which is generally discouraged). For this reason (among others), it is recommended to immediately switch to using Flakes for our NixOS configuration. Doing this is pretty simple. Just add a `flake.nix` file in `/etc/nixos`:

```
sudo nvim /etc/nixos/flake.nix
```

Add the following:

```
# /etc/nixos/flake.nix
{
  inputs = {
    # NOTE: Replace "nixos-23.11" with that which is in system.stateVersion of
    # configuration.nix. You can also use latter versions if you wish to
    # upgrade.
    nixpkgs.url = "github:NixOS/nixpkgs/nixos-23.11";
  };
  outputs = inputs@{ self, nixpkgs, ... }: {
    # NOTE: 'nixos' is the default hostname set by the installer
    nixosConfigurations.nixos = nixpkgs.lib.nixosSystem {
      # NOTE: Change this to aarch64-linux if you are on ARM
      system = "x86_64-linux";
      modules = [ ./configuration.nix ];
    };
  };
}
```

Make sure to change a couple of things in the above snippet:

- Replace `nixos-23.11` with the version from `system.stateVersion` in your `/etc/nixos/configuration.nix`. If you wish to upgrade right

away, you can also use latter versions, or use `nixos-unstable` for the bleeding edge.

- `x86_64-linux` should be `aarch64-linux` if you are on ARM

Now, `/etc/nixos` is technically a flake. We can "inspect" this flake using the `nix flake show` command:

```
$ nix flake show /etc/nixos
error: experimental Nix feature 'nix-command' is disabled; use '--extra-exper
```

Oops, what happened here? As flakes is a so-called "experimental" feature, you must manually enable it. We'll *temporarily* enable it for now, and then enable it *permanently* latter. The `--extra-experimental-features` flag can be used to enable experimental features. Let's try again:

```
$ nix --extra-experimental-features 'nix-command flakes' flake show /etc/nixos
warning: creating lock file '/etc/nixos/flake.lock'
error:
       … while updating the lock file of flake 'path:/etc/nixos?lastModified=1

       error: opening file '/etc/nixos/flake.lock': Permission denied
```

Progress, but we hit another error—Nix understandably cannot write to root-owned directory (it tries to create the `flake.lock` file). One way to resolve this is to move the whole configuration to our home directory, which would also prepare the ground for storing it in Git. We will do this in the next section.

> ⓘ `flake.lock`
>
> Nix commands automatically generate a (or update the) `flake.lock` file. This file contains the exacted pinned version of the inputs of the flake, which is important for reproducibility.

Move configuration to user directory

Move the entire `/etc/nixos` directory to your home directory and gain control of it:

```
$ sudo mv /etc/nixos ~/nixos-config && sudo chown -R $USER ~/nixos-config
```

Your configuration directory should now look like:

```
$ ls -l ~/nixos-config/
total 12
-rw-r--r-- 1 srid root 4001 Dec  9 16:03 configuration.nix
-rw-r--r-- 1 srid root  224 Dec  9 16:12 flake.nix
-rw-r--r-- 1 srid root 1317 Dec  9 15:43 hardware-configuration.nix
```

Now let's try `nix flake show` on it, and this time it should work:

```
$ cd ~/nixos-config
$ nix --extra-experimental-features 'nix-command flakes' flake show
warning: creating lock file '/home/srid/nixos-config/flake.lock'
path:/home/srid/nixos-config?lastModified=1702156518&narHash=sha256-nDtDyzk3f
└───nixosConfigurations
    └───nixos: NixOS configuration
```

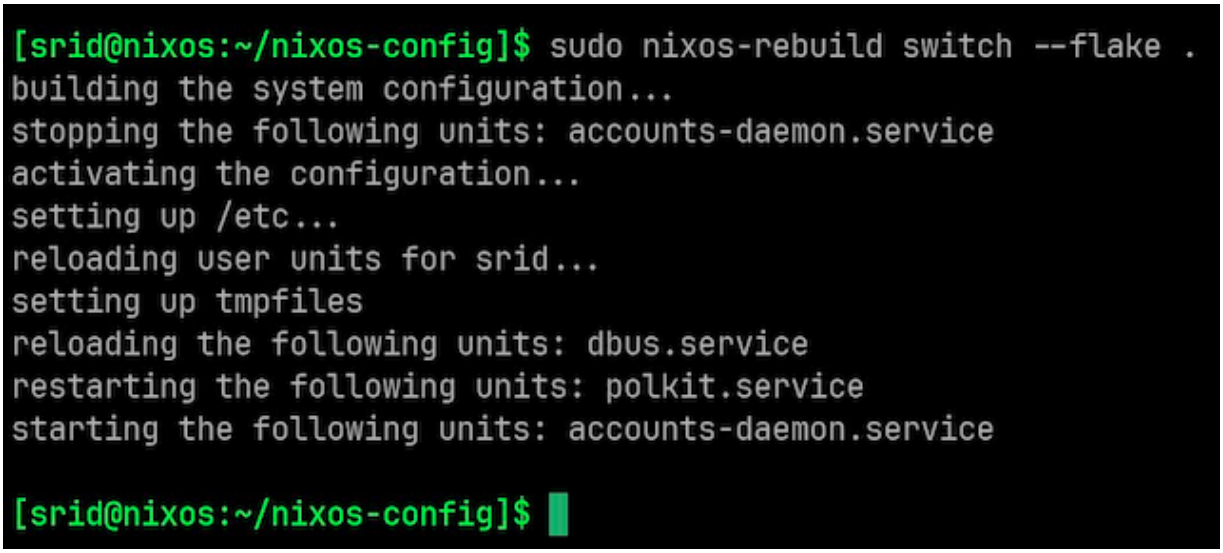
Voila! Incidentally, this flake has a single output, `nixosConfigurations.nixos`, which is the NixOS configuration itself.

> **More on Flakes**
>
> See Rapid Introduction to Nix for more information on flakes.

Once flake-ified, we can use the same command to activate the new configuration but we must additionally pass the `--flake` flag, viz.:

```
# The '.' is the path to the flake, which is current directory.
$ sudo nixos-rebuild switch --flake .
```

If everything went well, you should see something like this:



Excellent, now we have a flake-ified NixOS configuration that is pure and reproducible!

Links to this page

Install NixOS with `disko` disk partitioning

> Before we can utilize **`disko`** in our generated configuration, we will convert our configuration to a flake. This is a simple process of adding a **`flake.nix`** file in **`/mnt/etc/nixos`**:

> For details, see Convert **`configuration.nix`** to be a flake.

> Move configuration to home dir

Install NixOS with Flake configuration on Git

> Let's store our whole configuration in a Git repository.

> Convert **`configuration.nix`** to be a flake