NixOS ❄: tmpfs as home

Table of Contents

This is a follow-up post for my earlier post: NixOS ❄: tmpfs as root.

When you start to go down the route of setting up a “pure” system that is as clean as you want it to be on each boot. You may start with the lazy route of using a persistent partition for your home directory. But it never feels quite right. But initially it's so convenient to choose this path.

So if you have gone through the first steps to get to a state of having a “pure” system, but didn't go “all in” and left your entire home directory as a persistent directory. This post will explain how to take it to the next level of purity.

Part 1 - Home Manager

The first thing you want to do unless you already done this is to start using Home Manager to manage the home directory. It's an excellent way to configure many programs as the user through a nix configuration file.

However I would advice to use the NixOS module that home manager provides instead of using home manager directly. This way the updates to home manager is done as part of the system generations, as well as rollbacks. The set-up for using the module is fairly straightforward and simple.

Sample module that can be imported into `configuration.nix` using `imports`:

Nix
```
{ pkgs, ... }:
```

```
let
  home-manager = builtins.fetchTarball {
    url = "https://github.com/nix-community/home-manager/archive/maste
  };
in {
  # Import the home-manager module.
  imports = [ "${home-manager}/nixos" ];

  # Home manager configuration for user jane.
  home-manager.users.jane = { pkgs, ... }: {
    programs.home-manager.enable = true;

    # In here you can do settings for home-manager, these are listed i
    # manual available in: $ man home-configuration.nix

    # In here you can use the "home.file" features of home manager to
    # certain files to certain content. However, that's not the same a
    # linking it to persistent storage since it will copy the file to
    # store where it will become read only. That may be perfectly fine
    # certain files. For other paths (such as "~/Downloads/"), it may
    # quite unsuitable.

    # Example: Enable network manager applet for this user:
    services.network-manager-applet.enable = true;
  };
}
```

To make this even better, talyz spent quite some time to implement modules to help doing persistence handling better – both in NixOS, but also as a Home Manager module. These modules became the impermanence community project.

Part 2 - Persistence module for Home Manager

Based on the previous example we're going to extend it to use the home manager module.

So we're going to extend the code sample from above:

Nix
```
{ pkgs, ... }:
```

```
let
  home-manager = builtins.fetchTarball {
    url = "https://github.com/nix-community/home-manager/archive/maste
  };
  # New: Load the module
  impermanence = builtins.fetchTarball {
    url =
      "https://github.com/nix-community/impermanence/archive/master.ta
  };
in {
  # Import the home-manager module.
  imports = [ "${home-manager}/nixos" ];

  # Home manager configuration for user jane.
  home-manager.users.jane = { pkgs, ... }: {
    # New: Import a persistence module for home-manager.
    imports = [ "${impermanence}/home-manager.nix" ];

    programs.home-manager.enable = true;

    # New: Now we can use the "home.persistence" module, here's an exa
    home.persistence."/nix/persist/home/jane" = {
      directories = [ ".ssh" "Downloads" ];
      files = [ ".bash_history" ];
    };
  };
}
```

The `home.persistence` module allows to easily select which files and directories you want to have symlinked from your home directory to the persistent storage selected.

This also allows to have different storage for different files in your home directory. Say that you have different needs to backup certain files but that you still want persistence. Then you can put the ones you want to backup in a certain directory and the rest in another place.

Part 3 - Persistence module for NixOS

This example nix file will import the persistence module for nixos and create some symlinks for files in `etc` that may be useful to have persistence on, as well making bind mounts for certain directories.

```nix
{ ... }:

let
  impermanence = builtins.fetchTarball {
    url =
      "https://github.com/nix-community/impermanence/archive/master.ta
  };
in {
  # Import the nixos module.
  imports = [ "${impermanence}/nixos.nix" ];

  environment.persistence."/nix/persist" = {
    directories = [
      "/etc/nixos"
      "/etc/NetworkManager/system-connections"
    ];
    files = [
      "/etc/machine-id"
      "/etc/ssh/ssh_host_rsa_key"
      "/etc/ssh/ssh_host_rsa_key.pub"
      "/etc/ssh/ssh_host_ed25519_key"
      "/etc/ssh/ssh_host_ed25519_key.pub"
    ];
  };
}
```

Date: 2020-06-27 Sat 22:30
Author: Elis Hirwing <elis@hirwing.se>
Created: 2021-05-23 Sun 15:02
