Other Useful Tips

Show detailed error messages

You can always try to add `--show-trace --print-build-logs --verbose` to the `nixos-rebuild` command to get the detailed error message if you encounter any errors during the deployment. e.g.

```bash
cd /etc/nixos
sudo nixos-rebuild switch --flake .#myhost --show-trace --print-build-logs --ver

# A more concise version
sudo nixos-rebuild switch --flake .#myhost --show-trace -L -v
```

Managing the Configuration with Git

NixOS configuration, being a set of text files, is well-suited for version control with Git. This allows easy rollback to a previous version in case of issues.

> NOTE: When using Git, Nix ignores all files that are not tracked by Git. If you encounter an error in Nix stating that a particular file is not found, it may be because you haven't `git add` ed it.

By default, NixOS places the configuration in `/etc/nixos`, which requires root permissions for modification, making it inconvenient for daily use. Thankfully, Flakes can help solve this problem by allowing you to place your flake anywhere you prefer.

For example, you can place your flake in `~/nixos-config` and create a symbolic link in `/etc/nixos` as follows:

```shell
sudo mv /etc/nixos /etc/nixos.bak  # Backup the original configuration
sudo ln -s ~/nixos-config /etc/nixos
```

```
# Deploy the flake.nix located at the default location (/etc/nixos)
sudo nixos-rebuild switch
```

This way, you can use Git to manage the configuration in `~/nixos-config`. The configuration can be modified with regular user-level permissions and does not require root ownership.

Another approach is to delete `/etc/nixos` directly and specify the configuration file path each time you deploy it:

shell

```
sudo mv /etc/nixos /etc/nixos.bak
cd ~/nixos-config

# `--flake .#my-nixos` deploys the flake.nix located in
# the current directory, and the nixosConfiguration's name is `my-nixos`
sudo nixos-rebuild switch --flake .#my-nixos
```

Choose the method that suits you best. Afterward, system rollback becomes simple. Just switch to the previous commit and deploy it:

shell

```
cd ~/nixos-config
# Switch to the previous commit
git checkout HEAD^1
# Deploy the flake.nix located in the current directory,
# with the nixosConfiguration's name `my-nixos`
sudo nixos-rebuild switch --flake .#my-nixos
```

More advanced Git operations are not covered here, but in general, rollback can be performed directly using Git. Only in cases of complete system crashes would you need to restart into the bootloader and boot the system from a previous historical version.

Viewing and Deleting Historical Data

As mentioned earlier, each NixOS deployment creates a new version, and all versions are added to the system's boot options. In addition to restarting the computer, you can query all available historical versions using the following command:

```shell
nix profile history --profile /nix/var/nix/profiles/system
```

To clean up historical versions and free up storage space, use the following command:

```shell
# Delete all historical versions older than 7 days
sudo nix profile wipe-history --older-than 7d --profile /nix/var/nix/profiles/sy

# Wiping history won't garbage collect the unused packages, you need to run the
sudo nix-collect-garbage --delete-old

# Due to the following issue, you need to run the gc command as per user to dele
# https://github.com/NixOS/nix/issues/8508
nix-collect-garbage --delete-old
```

Why some packages are installed?

To find out why a package is installed, you can use the following command:

1. Enter a shell with `nix-tree` & `rg` available: `nix shell nixpkgs#nix-tree nixpkgs#ripgrep`
2. `nix-store --gc --print-roots | rg -v '/proc/' | rg -Po '(?<= -> ).*' | xargs -o nix-tree`
3. `/<package-name>` to find the package you want to check.
4. `w` to show the package is depended by which packages, and the full dependency chain.

Reducing Disk Usage

The following configuration can be added to your NixOS configuration to help reduce disk usage:

```nix
{ lib, pkgs, ... }:

```

```nix
{
  # ...

  # Limit the number of generations to keep
  boot.loader.systemd-boot.configurationLimit = 10;
  # boot.loader.grub.configurationLimit = 10;

  # Perform garbage collection weekly to maintain low disk usage
  nix.gc = {
    automatic = true;
    dates = "weekly";
    options = "--delete-older-than 1w";
  };

  # Optimize storage
  # You can also manually optimize the store via:
  #    nix-store --optimise
  # Refer to the following link for more details:
  # https://nixos.org/manual/nix/stable/command-ref/conf-file.html#conf-auto-opt
  nix.settings.auto-optimise-store = true;
}
```

By incorporating this configuration, you can better manage and optimize the disk usage of your NixOS system.

Loading comments...