Nix development environments

For Rust enabled Linux kernels

Fiona Behrens <me@kloenk.dev>

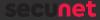
secunet Security Networks AG Division eHealth, working student

September 8, 2024



Summary

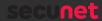
- 1 What is Nix
- 2 Development Shell
- 3 Direnv
- 4 Other features



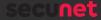
What is Nix



- A language for declarative & reproducible package management [2]
- Build steps defined in a derivation
- Package named after hash ⇒ can be used for substitution
- NixOS uses nix, but nix runs on all linux distros (+mac)
- Usually builds in a new environment ⇒ no incremental builds
- Lix[3] is a recent fork which offers some newer UI and a more "correct" implementation



- A language for declarative & reproducible package management [2]
- Build steps defined in a derivation
- Package named after hash \Rightarrow can be used for substitution
- NixOS uses nix, but nix runs on all linux distros (+mac)
- Usually builds in a new environment ⇒ no incremental builds
- Lix[3] is a recent fork which offers some newer UI and a more "correct" implementation



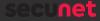
- A language for declarative & reproducible package management [2]
- Build steps defined in a derivation
- Package named after hash ⇒ can be used for substitution
- NixOS uses nix, but nix runs on all linux distros (+mac)
- Usually builds in a new environment ⇒ no incremental builds
- Lix[3] is a recent fork which offers some newer UI and a more "correct" implementation



- A language for declarative & reproducible package management [2]
- Build steps defined in a derivation
- Package named after hash ⇒ can be used for substitution
- NixOS uses nix, but nix runs on all linux distros (+mac)
- Usually builds in a new environment \Rightarrow no incremental builds
- Lix[3] is a recent fork which offers some newer UI and a more "correct" implementation



- A language for declarative & reproducible package management [2]
- Build steps defined in a derivation
- Package named after hash ⇒ can be used for substitution
- NixOS uses nix, but nix runs on all linux distros (+mac)
- Usually builds in a new environment ⇒ no incremental builds
- Lix[3] is a recent fork which offers some newer UI and a more "correct" implementation



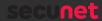
nixpkgs

- nixpkgs[4] as the main collection of packages
- stable release every 6 months (with rolling release unstable)
- Only has the most recent versions of most packages
- can be extended by third party overlays
- oxalica[5] provides more versions of rust



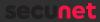
nixpkgs

- nixpkgs[4] as the main collection of packages
- stable release every 6 months (with rolling release unstable)
- Only has the most recent versions of most packages
- can be extended by third party overlays
- oxalica[5] provides more versions of rust



nixpkgs

- nixpkgs[4] as the main collection of packages
- stable release every 6 months (with rolling release unstable)
- Only has the most recent versions of most packages
- can be extended by third party overlays
- oxalica[5] provides more versions of rust



- An "Experimental" way to manage nix code dependencies
- Manages nix inputs
- povides eval caching



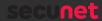
- An "Experimental" way to manage nix code dependencies
- Manages nix inputs
- povides eval caching



- An "Experimental" way to manage nix code dependencies
- Manages nix inputs
- povides eval caching



```
description = "Flake description";
inputs.nixpkgs.url = "github:nixos/nixpkgs/master";
outputs = { self, nixpkgs }: {
   ...
};
```



Development Shell



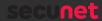
nix develop

nix develop nixpkgs#linux

- Does not provide development dependencies
- Does only support rust if the kernel config has rust enabled

"pkgs.mkShell"

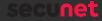
- Helper function creates a development environment
- Accepts "inputsFrom" to pick dependencies from a package



nix develop

nix develop nixpkgs#linux

- Does not provide development dependencies
- Does only support rust if the kernel config has rust enabled "pkgs.mkShell"
 - Helper function creates a development environment
 - Accepts "inputsFrom" to pick dependencies from a package



nix develop

```
mkShell {
  name = "linux";
  inputsFrom = [ linux ];
  nativeBuildInputs = [
    pkg-config
    ncurses
    (rust-bin.stable."1.78.0".default.override {
      extensions = [ "rust-src" ];
    })
    rust-bindgen
  hardeningDisable = [ ... ];
```

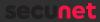
Direnv



direnv

Loads an environment from the file ".envrc" [1]

use flake /home/kloenk/Developer/nix/nixfiles#kernel



direnv

Loads an environment from the file ".envrc" [1]

use flake /home/kloenk/Developer/nix/nixfiles#kernel



Other features



NixOS VM tests

- Build a NixOS system
- Test in qemu using a python test driver
- Build in a sandbox without networking
- Usually makes a mrproper build
- Not really designed for kernel development



NixOS VM tests

- Build a NixOS system
- Test in qemu using a python test driver
- Build in a sandbox without networking
- Usually makes a mrproper build
- Not really designed for kernel development



NixOS VM tests

- Build a NixOS system
- Test in qemu using a python test driver
- Build in a sandbox without networking
- Usually makes a mrproper build
- Not really designed for kernel development



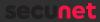
VM images

- Build full iso images ready to boot
- Build initrds for direkt kernel boot using qemu
- Includes kernel modules



VM images

- Build full iso images ready to boot
- Build initrds for direkt kernel boot using qemu
- Includes kernel modules



References

- direnv unclutter your .profile. https://direnv.net/.
- Dolstra, E. The Purely Functional Software Deployment Model.
 Undefined/Unknown. Doctoral thesis 1 (Research UU / Graduation UU) (Utrecht University, Jan. 2006). ISBN: 90-393-4130-3.
- Lix is a modern, delicious implementation of the Nix package manager, focused on correctness, usability, and growth. https://lix.systems/.
- Nix Packages collection & NixOS. https://github.com/nixOS/nixpkgs/.
- 5. oxalica's rust overlay.
 https://github.com/oxalica/rust-overlay



nix-collect-garbage -d

