`system-manager`

unleashing nix on (almost) any distro

Ramses de Norre

01-02-2025

Who am I?

- Ramses, `r-vdp` online

Who am I?

- Ramses, `r-vdp` online
- Theoretical high-energy physicist in a distant past

Who am I?

- Ramses, `r-vdp` online
- Theoretical high-energy physicist in a distant past
- Figured out that there is absolutely no job market for that 🤷

Who am I?

- Ramses, `r-vdp` online
- Theoretical high-energy physicist in a distant past
- Figured out that there is absolutely no job market for that 🤷
- Full stack Java/Scala developer

Who am I?

- Ramses, `r-vdp` online
- Theoretical high-energy physicist in a distant past
- Figured out that there is absolutely no job market for that 🤷
- Full stack Java/Scala developer
- Tech lead/security engineer at an INGO

Who am I?

- Ramses, `r-vdp` online
- Theoretical high-energy physicist in a distant past
- Figured out that there is absolutely no job market for that 🤷
- Full stack Java/Scala developer
- Tech lead/security engineer at an INGO
- Independent software engineer with Numtide

Who am I?

- Ramses, `r-vdp` online
- Theoretical high-energy physicist in a distant past
- Figured out that there is absolutely no job market for that 🤷
- Full stack Java/Scala developer
- Tech lead/security engineer at an INGO
- Independent software engineer with Numtide
- Using Nix since 2017

Who am I?

- Ramses, `r-vdp` online
- Theoretical high-energy physicist in a distant past
- Figured out that there is absolutely no job market for that 🤷
- Full stack Java/Scala developer
- Tech lead/security engineer at an INGO
- Independent software engineer with Numtide
- Using Nix since 2017
- **Love working on FOSS**

Who am I?

- Ramses, `r-vdp` online
- Theoretical high-energy physicist in a distant past
- Figured out that there is absolutely no job market for that 🤷
- Full stack Java/Scala developer
- Tech lead/security engineer at an INGO
- Independent software engineer with Numtide
- Using Nix since 2017
- **Love working on FOSS**

Matrix: @rvdp:infosec.exchange

Outline

`system-manager`

- What problem are we trying to solve?
- What did we come up with?
- (Current) limitations
- What's next?

The problem

NixOS is great, but it's kind of all or nothing

The problem

NixOS is great, but it's kind of all or nothing

`home-manager` fills the gap for declarative home environments

The problem

NixOS is great, but it's kind of all or nothing

`home-manager` fills the gap for declarative home environments

Can we have nix manage system-level configuration on non-NixOS systems?

Potential options

- systemd portable services

Potential options

- systemd portable services
- NixOS containers (systemd-nspawn)

Potential options

- systemd portable services
- NixOS containers (systemd-nspawn)
- custom switch-to-configuration script (cf. home-manager)

Potential options

- systemd portable services
- NixOS containers (systemd-nspawn)
- **custom switch-to-configuration script (cf. home-manager)**

Advantages

- Well understood mechanism (we mostly copy NixOS/`home-manager`)
- Native systemd services (no containers, no images, no networking shenanigans)
- Small number of fundamental operations that need to be supported

Advantages

- Well understood mechanism (we mostly copy NixOS/`home-manager`)
- Native systemd services (no containers, no images, no networking shenanigans)
- Small number of fundamental operations that need to be supported
  - Create files in `/etc`
  - systemd-tmpfiles
  - start/restart/stop systemd services

Advantages

- Well understood mechanism (we mostly copy NixOS/`home-manager`)
- Native systemd services (no containers, no images, no networking shenanigans)
- Small number of fundamental operations that need to be supported
  - Create files in `/etc`
  - systemd-tmpfiles
  - start/restart/stop systemd services

Some things are missing:

- Users and groups
- Filesystems
- Networking
- Global systemd settings
- …

Disadvantages

The implementation needs to be

- written from scratch
- maintained

Disadvantages

The implementation needs to be

- written from scratch
- maintained

NixOS modules

- have a lot of interdependencies
- assume a lot of things

Disadvantages

The implementation needs to be

- written from scratch
- maintained

NixOS modules

- have a lot of interdependencies
- assume a lot of things

And also

- we cannot easily support everything that can be done on NixOS without clashing with the embedding distro
- we share the space with the embedding distro

What does it do?

You need to have nix available, running in daemon mode

```
nix run github:numtide/system-manager -- --flake '...'
```

What does it do?

You need to have nix available, running in daemon mode

```
nix run github:numtide/system-manager -- --flake '...'
```

By default, we:

- Run `systemd-tmpfiles` (scoped to system-manager config)
- Create `/run/system-manager/sw`, containing installed packages
- Create `/etc/profile` drop-in to configure `$PATH`
- Create `system-manager.target`, pulled in by `default.target`, to initialise system-manager on reboot
- All other services, targets, etc. get pulled in by `system-manager.target`
- Register a profile and a GC root

Result

- `/nix/store`
- Symlinks in /etc
- Systemd services
- Nix' "static linking" through `/nix/store` references (you can use `${ ... }`
- Stuff in `$PATH`

Can I use this today?

Yes!

Can I use this today?

Yes! But...

- It is definitely still work in progress
- Not much documentation
- We need more modules

Can I use this today?

Yes! But...

- It is definitely still work in progress
- Not much documentation
- We need more modules

But it works!

Potential improvements

- Mount `/etc` using an overlay, similar to the experimental feature in NixOS
- Figure out a way to manage **some** users/groups, not all
- Figure out how to reuse modules from nixpkgs, if possible?

Testing across different distros

- We re-use the NixOS testing driver
- Load different images in the VM
- Same test instrumentation

Testing across different distros

```
copying 2 paths...
copying path '/nix/store/65szxgdhxvc8ri52irxhf1j69g1ir8zn-test-script' to 'ssh-ng://ramses@bld1.numtide.com'...
copying path '/nix/store/k3c4bpnhzynipaw2g92rsdw41kvfqicp-run-vm-vm' to 'ssh-ng://ramses@bld1.numtide.com'...
building '/nix/store/9pmadjzw0c94qdfhiqk2nfwhsp6zsb5w-vm-test.drv'...
vm-test> Machine state will be reset. To keep it, pass --keep-vm-state
vm-test> start all VLans
vm-test> start vlan
vm-test> running vlan (pid 9; ctl /build/vde1.ctl)
vm-test> (finished: start all VLans, in 0.00 seconds)
vm-test> Test will time out and terminate in None seconds
vm-test> run the VM test script
vm-test> additionally exposed symbols:
vm-test>     vm,
vm-test>     vlan1,
vm-test>     start_all, test_script, machines, vlans, driver, log, os, create_machine, subtest, run_tests, join_all, retry, serial_stdout_off, serial_stdout_on, polling_condition, Machine
vm-test> start all VMs
vm-test> vm: starting vm
vm-test> vm # /build /build/vm-state-vm
vm-test> vm: QEMU running (pid 11)
vm-test> vm # /build/vm-state-vm
vm-test> (finished: start all VMs, in 0.10 seconds)
vm-test> vm: waiting for unit default.target
vm-test> vm: waiting for the VM to finish booting
vm-test> vm # cSeaBIOS (version rel-1.16.3-0-ga6ed6b701f0a-prebuilt.qemu.org)
vm-test> vm #
vm-test> vm #
vm-test> vm # iPXE (http://ipxe.org) 00:04.0 CA00 PCI2.10 PnP PMM+3EFD0CE0+3EF30CE0 CA00
vm-test> vm # Press Ctrl-B to configure iPXE (PCI 00:04.0)...
vm-test> vm #
vm-test> vm #
vm-test> vm # Booting from Hard Disk...
vm-test> vm # [    0.000000] Linux version 6.5.0-44-generic (buildd@lcy02-amd64-014) (x86_64-linux-gnu-gcc-13 (Ubuntu 13.2.0-4ubuntu3) 13.2.0, GU ld (GNU Binutils for Ubuntu) 2.41) #44-Ubuntu SMP PREEMPT_DYNAM
IC Fri Jun  7 15:10:09 UTC 2024 (Ubuntu 6.5.0-44.44-generic 6.5.13)
vm-test> vm # [    0.000000] Command line: BOOT_IMAGE=/vmlinuz-6.5.0-44-generic root=LABEL=cloudimg-rootfs ro console=tty1 console=ttyS0
vm-test> vm # [    0.000000] KERNEL supported cpus:
vm-test> vm # [    0.000000]   Intel GenuineIntel
vm-test> vm # [    0.000000]   AMD AuthenticAMD
vm-test> vm # [    0.000000]   Hygon HygonGenuine
vm-test> vm # [    0.000000]   Centaur CentaurHauls
vm-test> vm # [    0.000000]   zhaoxin   Shanghai
vm-test> vm # [    0.000000] x86/split lock detection: #DB: warning on user-space bus_locks
vm-test> vm # [    0.000000] BIOS-provided physical RAM map:
vm-test> vm # [    0.000000] BIOS-e820: [mem 0x0000000000000000-0x000000000009fbff] usable
```

Testing across different distros

- We re-use the NixOS testing driver
- Load different images in the VM
- Same test instrumentation
- **https://github.com/numtide/nix-vm-test/** (kudos to @picnoir)

Repo

Want to try it out?

https://github.com/numtide/system-manager