# NixOS: The Purely Functional Linux Distribution

Eelco Dolstra
LogicBlox

21 November 2013

# So what was wrong with the previous 298 distributions?

# So what was wrong with the previous 298 distributions?

- Upgrading is risky business
  - No way to roll back
  - Not atomic / transactional

# So what was wrong with the previous 298 distributions?

- Upgrading is risky business
  - No way to roll back
  - Not atomic / transactional
- Not declarative
  - Configuration is result of automated/manual changes over time
  - Therefore hard to reproduce

# So what was wrong with the previous 298 distributions?

- Upgrading is risky business
  - No way to roll back
  - Not atomic / transactional
- Not declarative
  - Configuration is result of automated/manual changes over time
  - Therefore hard to reproduce
- Package management limitations
  - Hard to have multiple versions of a package
  - Need to be root to install packages
  - Upgrading one package may break others (“DLL hell”)

# Analysis

Conventional Linux distributions have an
**imperative** (stateful) approach to configuration / package management.

```
/
├─ bin
│  ├─ sh
│  └─ ls
├─ lib
│  └─ libc.so.6
├─ etc
│  ├─ resolv.conf
│  ├─ passwd
│  └─ crontab
├─ usr
│  └─ bin
│     ├─ firefox
│     ├─ ssh
│     └─ ooffice
└─ ...
```

# Analysis

Conventional Linux distributions have an
**imperative** (stateful) approach to configuration / package management.

```
/
├ bin
│ ├ sh
│ └ ls
├ lib
│ └ libc.so.6
├ etc
│ ├ resolv.conf
│ ├ passwd
│ └ crontab
├ usr
│ └ bin
│   ├ firefox
│   ├ ssh
│   └ ooffice
└ ...
```

## Examples

- Package managers like RPM perform *destructive updates* to the filesystem while doing upgrades.
- Packages have post-install scripts that scribble all over `/etc`.
- Sysadmins make manual changes to config files in `/etc`.

# Enter NixOS

# NixOS has a purely functional model

(Based on Nix, the purely functional package manager.)
This means:

- Packages, config files and other static parts of the system are built by *pure functions*.
- They are *immutable* after they have been built.

# Consequences

- All static parts are stored under `/nix/store`;
  (almost) no `/bin`, `/lib`, `/usr`, ...
- Upgrades are non-destructive; can roll back.
- Upgrades are atomic.
- Stateless: upgrading equivalent to reinstalling from scratch.
- Deterministic: can easily reproduce a configuration on another machine.

# How it works: the Nix store

Nix stores all packages in isolation from each other:

`/nix/store/rpdqxnilb0cg...-firefox-25.0`

Paths contain a 160-bit **cryptographic hash** of **all** inputs used to build the package:

- Sources
- Libraries
- Compilers
- Build scripts
- ...

```
/nix/store
├─ l9w6773m1msy...-openssh-4.6
│  ├─ bin
│  │  └─ ssh
│  └─ sbin
│     └─ sshd
├─ smkabrbibqv7...-openssl-0.9.
│  └─ lib
│     └─ libssl.so.0.9.8
├─ c6jbqm2mc0a7...-zlib-1.2.3
│  └─ lib
│     └─ libz.so.1.2.3
└─ im276akmsrhv...-glibc-2.5
   └─ lib
      └─ libc.so.6
```

# NixOS is declarative

## /etc/nixos/configuration.nix

```
{
  boot.loader.grub.bootDevice = "/dev/sda";
  fileSystems."/".device = "/dev/sda1";
  swapDevices = [ { device = "/dev/sdb1"; } ];
  services.sshd.enable = true;
  services.sshd.forwardX11 = true;
}
```

# NixOS is declarative

To enable Apache, add this to `configuration.nix`:

```
services.httpd.enable = true;
services.httpd.documentRoot = "/bla";
```

# NixOS is declarative

To enable Apache, add this to `configuration.nix`:

```
services.httpd.enable = true;
services.httpd.documentRoot = "/bla";
```

Or to enable X11/KDE:

```
services.xserver.enable = true;
services.xserver.displayManager.kde4.enable = true;
```

# NixOS is declarative

To enable Apache, add this to `configuration.nix`:

```
services.httpd.enable = true;
services.httpd.documentRoot = "/bla";
```

Or to enable X11/KDE:

```
services.xserver.enable = true;
services.xserver.displayManager.kde4.enable = true;
```

Or to make Firefox available to users:

```
environment.systemPackages = [ pkgs.firefox ];
```

# NixOS is declarative

To enable Apache, add this to `configuration.nix`:

```
services.httpd.enable = true;
services.httpd.documentRoot = "/bla";
```

Or to enable X11/KDE:

```
services.xserver.enable = true;
services.xserver.displayManager.kde4.enable = true;
```

Or to make Firefox available to users:

```
environment.systemPackages = [ pkgs.firefox ];
```

Then run:

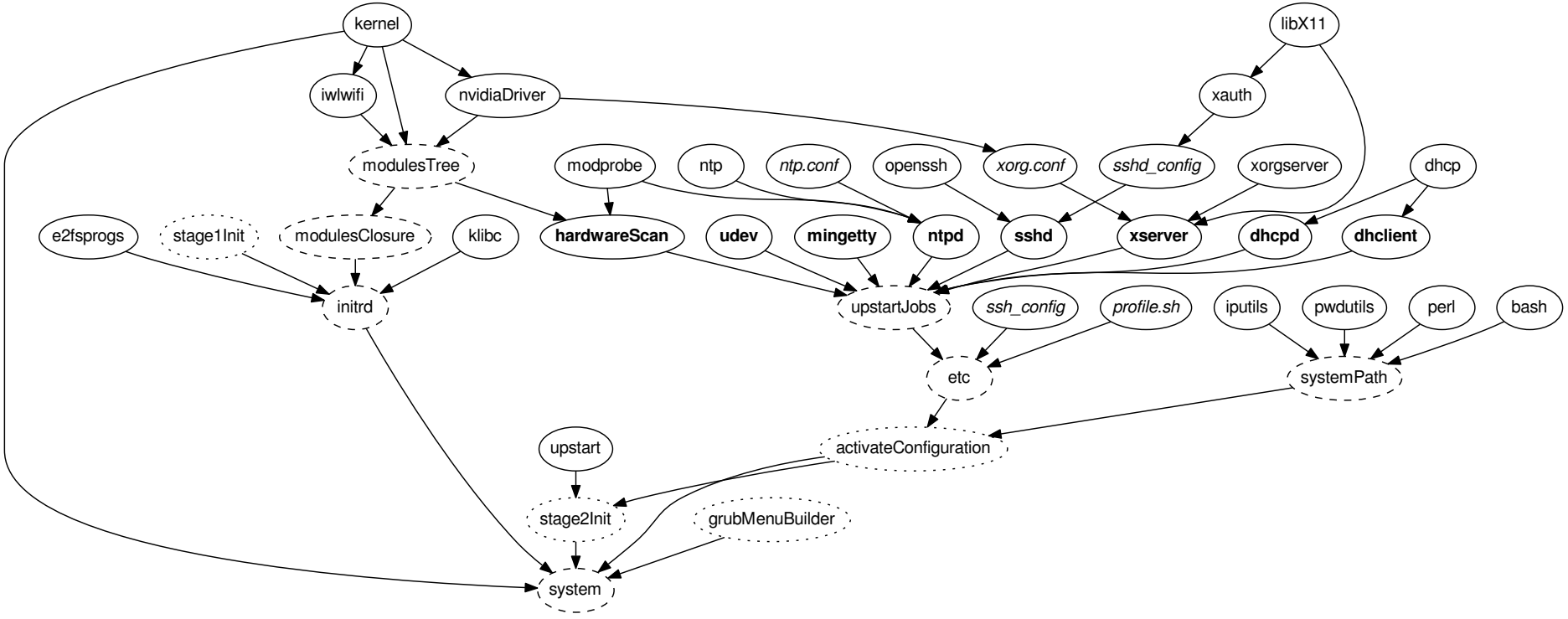
```
$ nixos-rebuild switch
```

# Reproducibility

`nixos-rebuild switch` invokes Nix to build or download the entire system configuration: packages, boot scripts, config files, ...

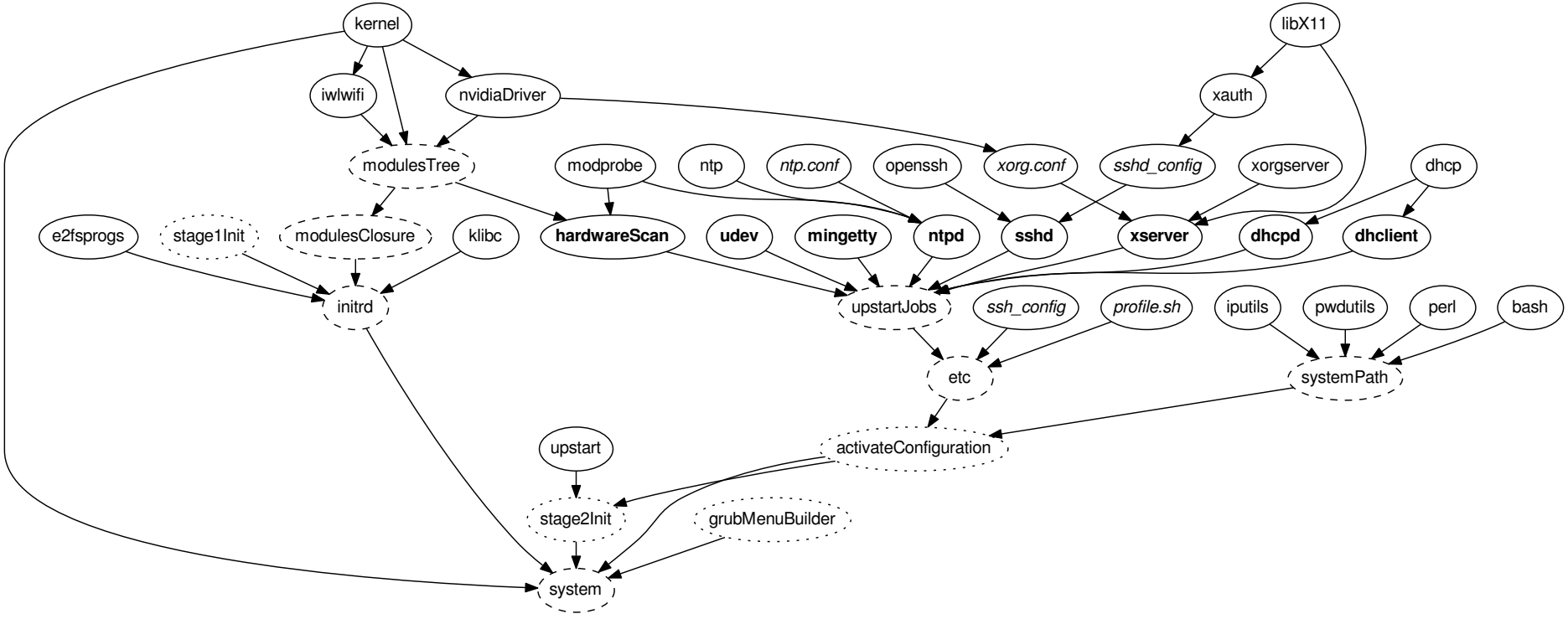
# Reproducibility

`nixos-rebuild switch` invokes Nix to build or download the entire system configuration: packages, boot scripts, config files, ...



# Reproducibility

`nixos-rebuild switch` invokes Nix to build or download the entire system configuration: packages, boot scripts, config files, ...




Since this is purely functional, configurations can always be reproduced.

# Rollbacks are easy

To undo the last upgrade:

```
$ nixos-rebuild switch --rollback
```

# Rollbacks are easy

To undo the last upgrade:

```
$ nixos-rebuild switch --rollback
```

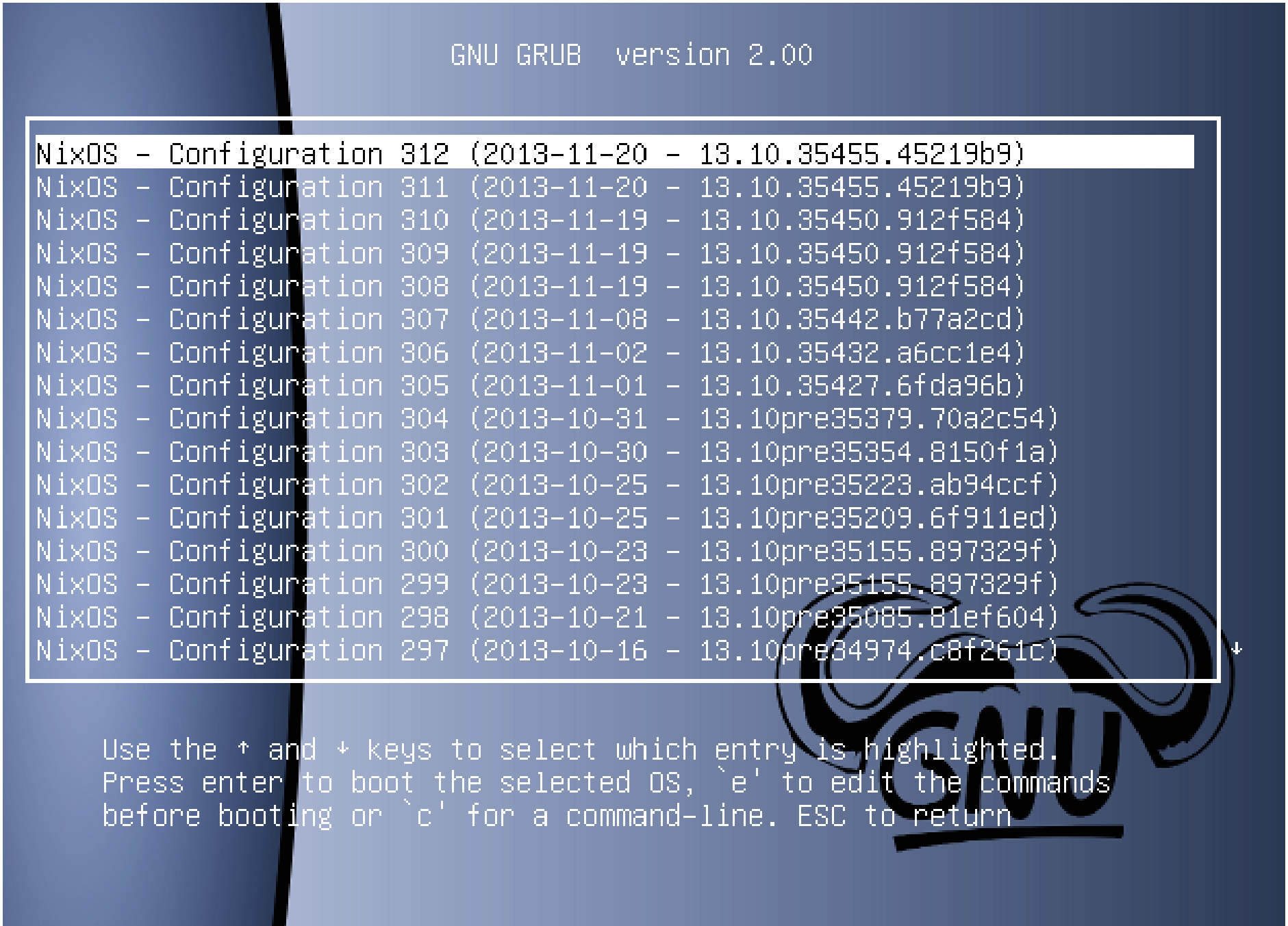
Alternatively, revert `configuration.nix`, e.g. remove

```
services.httpd.enable = true;
services.httpd.documentRoot = "/bla";
```

and run

```
$ nixos-rebuild switch
```

# Rollbacks are really easy



# Testing a configuration is painless

To test a new configuration without making it the boot default:

```
$ nixos-rebuild test
```

# Testing a configuration is painless

To test a new configuration without making it the boot default:

```
$ nixos-rebuild test
```

To build a VM containing the new configuration:

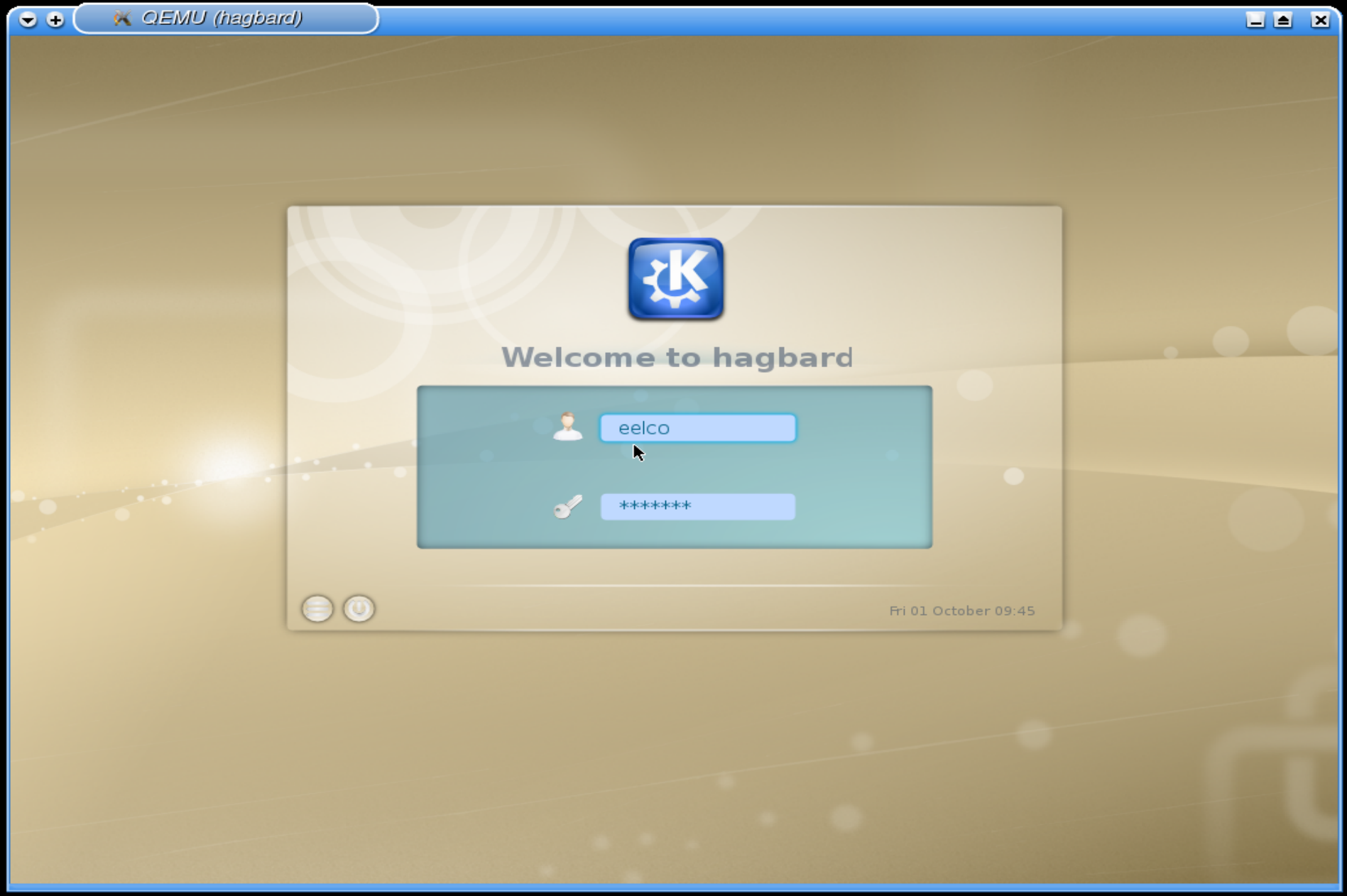
```
$ nixos-rebuild build-vm
```

```
[eelco@hagbard:~/Dev/nixos]$ nixos-rebuild build-vm --fast
building the system configuration...
trace: Obsolete option `services.sshd.enable' is defined instead of `services.openssh.enable'.

Done.  The virtual machine can be started by running ./result/bin/run-hagbard-vm.

[eelco@hagbard:~/Dev/nixos]$ ./result/bin/run-hagbard-vm
```

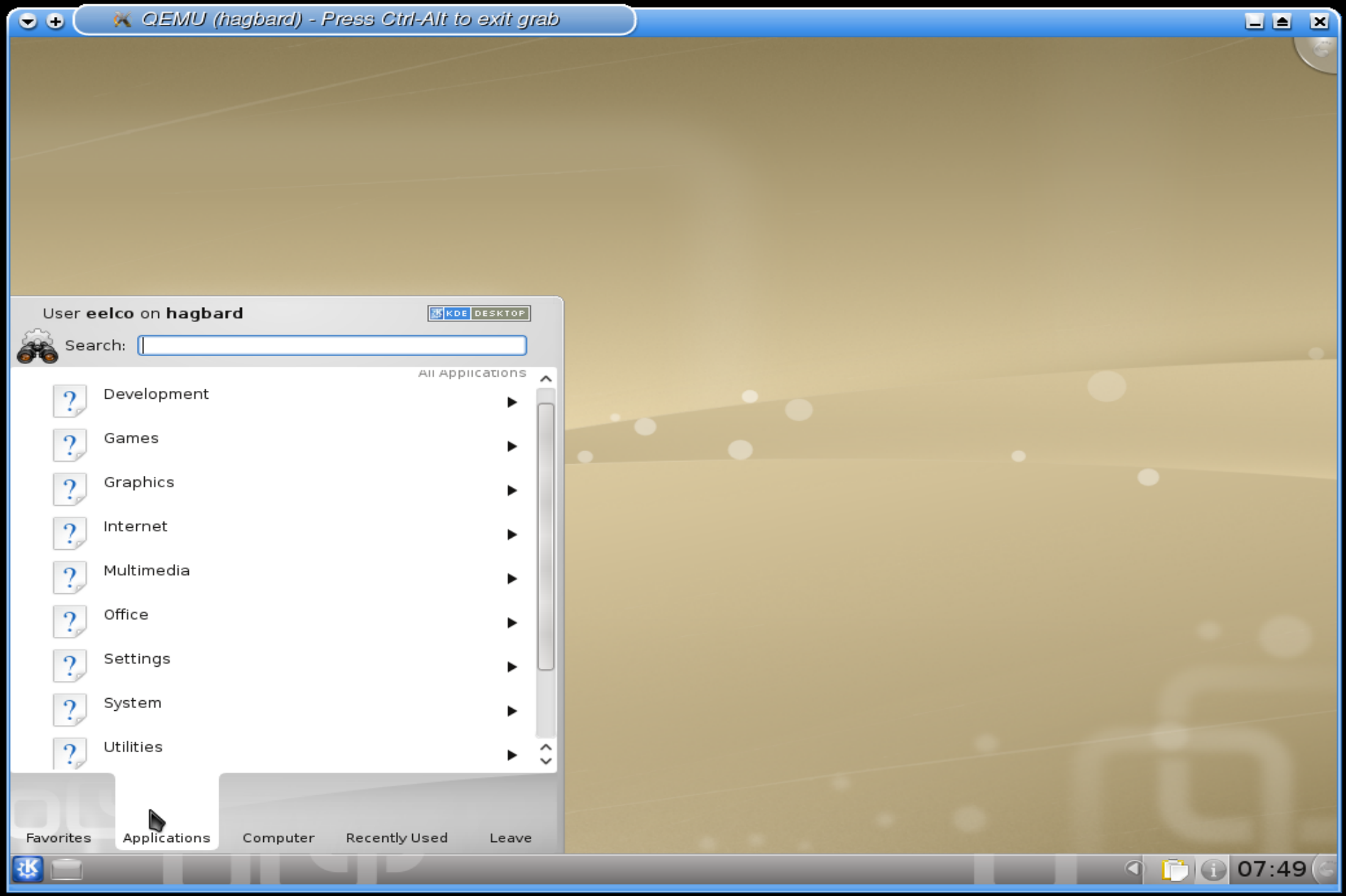
QEMU (hagbard)

```
[    0.659344] virtio-pci 0000:00:03.0: PCI INT A -> Link[LNKC] -> GSI 11 (level, high) -> IRQ 11
[    0.671629] ACPI: PCI Interrupt Link [LNKD] enabled at IRQ 10
[    0.672124] virtio-pci 0000:00:04.0: PCI INT A -> Link[LNKD] -> GSI 10 (level, high) -> IRQ 10
[    0.679294]  vda: unknown partition table
[    0.682585] scsi0 : ata_piix
[    0.684051] scsi1 : ata_piix
[    0.698057] ata1: PATA max MWDMA2 cmd 0x1f0 ctl 0x3f6 bmdma 0xc000 irq 14
[    0.699059] ata2: PATA max MWDMA2 cmd 0x170 ctl 0x376 bmdma 0xc008 irq 15
[    0.852299] ata2.00: ATAPI: QEMU DVD-ROM, 0.12.5, max UDMA/100
[    0.855273] ata2.00: configured for MWDMA2
[    0.857502] scsi 1:0:0:0: CD-ROM            QEMU     QEMU DVD-ROM     0.12 PQ: 0 ANSI: 5
[    0.870604] sr0: scsi3-mmc drive: 4x/4x xa/form2 tray
[    0.871193] Uniform CD-ROM driver Revision: 3.20
starting device mapper and LVM...
  Reading all physical volumes.  This may take a while...
  No volume groups found
  No volume groups found
[    0.944976] Switching to clocksource hpet
failed to resume...
mounting /dev/vda on /...
fsck from util-linux-ng 2.17.2
[/nix/store/6vn41a0b98gz18wynsbamzjxwby4yjgn-extra-utils/bin/fsck.ext3 (1) -- /dev/vda] fsck.ext3
-a -C0 /dev/vda
/dev/vda: recovering journal
/dev/vda: clean, 1322/32768 files, 21486/131072 blocks
[    1.007550] kjournald starting.  Commit interval 5 seconds
[    1.019665] EXT3 FS on vda, internal journal
[    1.020993] EXT3-fs: mounted filesystem with writeback data mode.
mounting //10.0.2.4/qemu on /hostfs...
[    1.032796] Slow work thread pool: Starting up
[    1.033430] Slow work thread pool: Ready
mounting /mnt-root/hostfs/nix/store on /nix/store...

<<< NixOS Stage 2 >>>

running activation script...
setting up /etc...
```

```
[eelco@hagbard:~/Dev/nixos]$ nixos-rebuild build-vm --fast
building the system configuration...
trace: Obsolete option `services.sshd.enable' is defined instead of `services.openssh.enable'.

Done.  The virtual machine can be started by running ./result/bin/run-hagbard-vm.

[eelco@hagbard:~/Dev/nixos]$ ./result/bin/run-hagbard-vm
```



```
[eelco@hagbard:~/Dev/nixos]$ nixos-rebuild build-vm --fast
building the system configuration...
trace: Obsolete option `services.sshd.enable' is defined instead of `services.openssh.enable'.

Done.  The virtual machine can be started by running ./result/bin/run-hagbard-vm.

[eelco@hagbard:~/Dev/nixos]$ ./result/bin/run-hagbard-vm
```



# Multi-user package management

Non-root users can install software:

```
alice$ nix-env -i firefox-25.0.1

alice$ firefox --version
Mozilla Firefox 25.0.1
```

# Multi-user package management

Non-root users can install software:

```
alice$ nix-env -i firefox-25.0.1

alice$ firefox --version
Mozilla Firefox 25.0.1
```

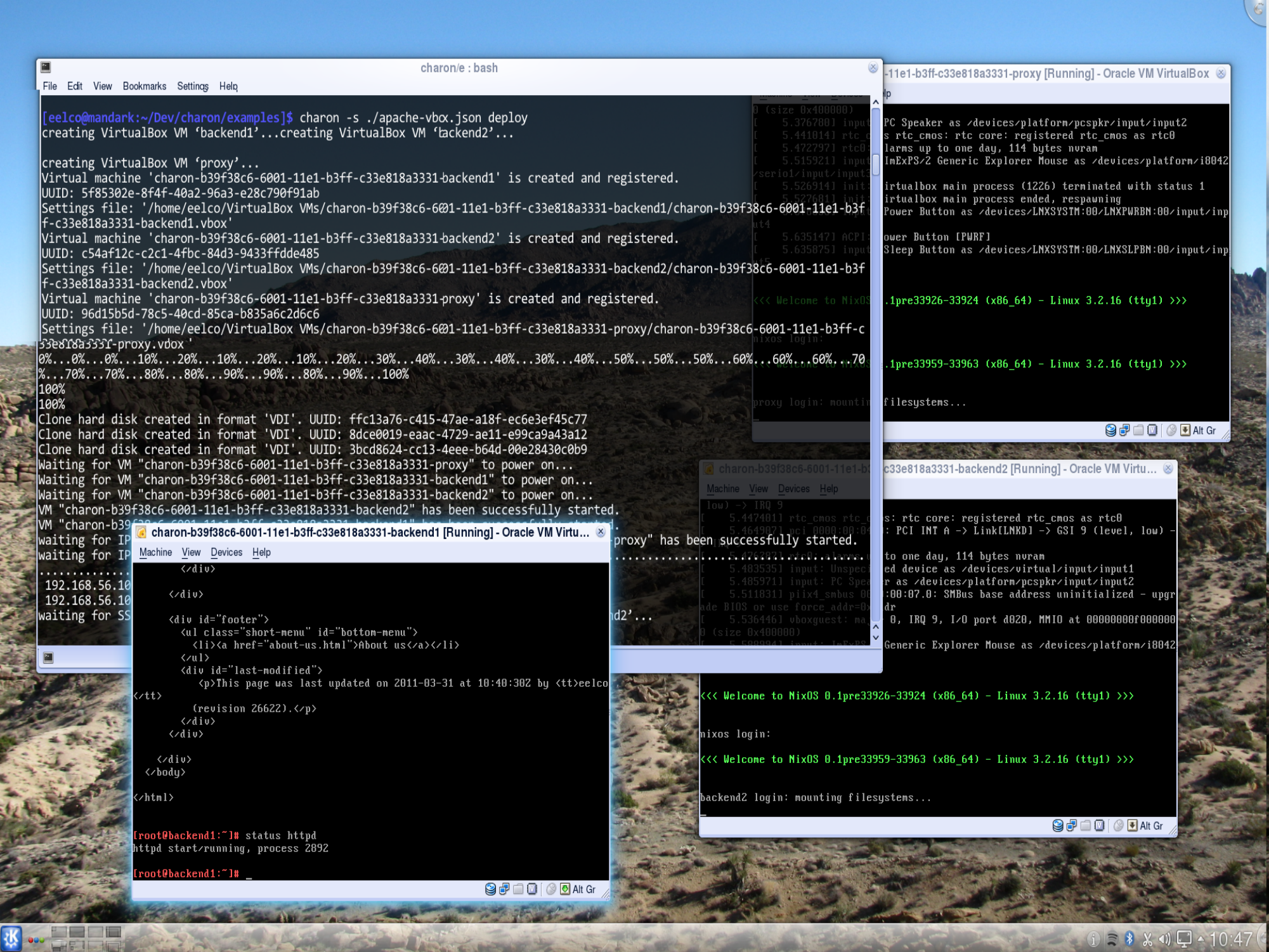
Without interfering with each other:

```
bob$ firefox
firefox: No such file or directory

bob$ nix-env -i firefox-3.6.27

bob$ firefox --version
Mozilla Firefox 3.6.27
```

# NixOps: Declarative Provisioning and Deployment

# NixOps extends the NixOS approach to networks of machines

```
{
  database =
    { deployment.targetEnv = "virtualbox";
      services.postgresql.enable = true;
    };

  webserver =
    { deployment.targetEnv = "virtualbox";
      services.httpd.enable = true;
      services.httpd.documentRoot = ...;
    };
}
```


charon/e : bash
File Edit View Bookmarks Settings Help
[eelco@mandark:~/Dev/charon/examples]$ charon -s ./apache-vbox.json deploy
creating VirtualBox VM 'backend1'...creating VirtualBox VM 'backend2'...
creating VirtualBox VM 'proxy'...
Virtual machine 'charon-b39f38c6-6001-11e1-b3ff-c33e818a3331-backend1' is created and registered.
UUID: 5f85302e-8f4f-40a2-96a3-e28c790f91ab
Settings file: '/home/eelco/VirtualBox VMs/charon-b39f38c6-6001-11e1-b3ff-c33e818a3331-backend1/charon-b39f38c6-6001-11e1-b3ff-c33e818a3331-backend1.vbox'
Virtual machine 'charon-b39f38c6-6001-11e1-b3ff-c33e818a3331-backend2' is created and registered.
UUID: c54af12c-c2c1-4fbc-84d3-9433ffdde485
Settings file: '/home/eelco/VirtualBox VMs/charon-b39f38c6-6001-11e1-b3ff-c33e818a3331-backend2/charon-b39f38c6-6001-11e1-b3ff-c33e818a3331-backend2.vbox'
Virtual machine 'charon-b39f38c6-6001-11e1-b3ff-c33e818a3331-proxy' is created and registered.
UUID: 96d15b5d-78c5-40cd-85ca-b835a6c2d6c6
Settings file: '/home/eelco/VirtualBox VMs/charon-b39f38c6-6001-11e1-b3ff-c33e818a3331-proxy/charon-b39f38c6-6001-11e1-b3ff-c33e818a3331-proxy.vbox'
0%...0%...0%...10%...20%...10%...20%...10%...20%...30%...40%...30%...40%...30%...40%...50%...50%...50%...60%...60%...60%...70
%...70%...70%...80%...80%...90%...90%...80%...90%...100%
100%
100%
Clone hard disk created in format 'VDI'. UUID: ffc13a76-c415-47ae-a18f-ec6e3ef45c77
Clone hard disk created in format 'VDI'. UUID: 8dce0019-eaac-4729-ae11-e99ca9a43a12
Clone hard disk created in format 'VDI'. UUID: 3bcd8624-cc13-4eee-b64d-00e28430c0b9
Waiting for VM "charon-b39f38c6-6001-11e1-b3ff-c33e818a3331-proxy" to power on...
Waiting for VM "charon-b39f38c6-6001-11e1-b3ff-c33e818a3331-backend1" to power on...
Waiting for VM "charon-b39f38c6-6001-11e1-b3ff-c33e818a3331-backend2" to power on...
VM "charon-b39f38c6-6001-11e1-b3ff-c33e818a3331-backend2" has been successfully started.
VM "charon-b39f38c6-6001-11e1-b3ff-c33e818a3331-proxy" has been successfully started.
waiting for IP
waiting for IP
192.168.56.10
192.168.56.10
waiting for SS
charon-b39f38c6-6001-11e1-b3ff-c33e818a3331-backend1 [Running] - Oracle VM Virtu...
Machine View Devices Help

<ul class="short-menu" id="bottom-menu">
<li><a href="about-us.html">About us</a></li>
</ul>

This page was last updated on 2011-03-31 at 10:40:30Z by <tt>eelco</tt>
(revision 26622).

</body>
</html>
[root@backend1:~]# status httpd
httpd start/running, process 2892
[root@backend1:~]#
Alt Gr
-11e1-b3ff-c33e818a3331-proxy [Running] - Oracle VM VirtualBox
PC Speaker as /devices/platform/pcspkr/input/input2
s rtc_cmos: rtc core: registered rtc_cmos as rtc0
larms up to one day, 114 bytes nvram
ImExPS/2 Generic Explorer Mouse as /devices/platform/i8042
irtualbox main process (1226) terminated with status 1
irtualbox main process ended, respawning
Power Button as /devices/LNXSYSTM:00/LNXPWRBN:00/input/inp
ower Button [PWRF]
Sleep Button as /devices/LNXSYSTM:00/LNXSLPBN:00/input/inp
.1pre33926-33924 (x86_64) - Linux 3.2.16 (tty1) >>>
.1pre33959-33963 (x86_64) - Linux 3.2.16 (tty1) >>>
filesystems...
Alt Gr
c33e818a3331-backend2 [Running] - Oracle VM Virtu...
s: rtc core: registered rtc_cmos as rtc0
: PCI INT A -> Link[LNKD] -> GSI 9 (level, low) -
to one day, 114 bytes nvram
ed device as /devices/virtual/input/input1
r as /devices/platform/pcspkr/input/input2
:00:07.0: SMBus base address uninitialized - upgr
dr
0, IRQ 9, I/O port d020, MMIO at 00000000f000000
Generic Explorer Mouse as /devices/platform/i8042
<<< Welcome to NixOS 0.1pre33926-33924 (x86_64) - Linux 3.2.16 (tty1) >>>
nixos login:
<<< Welcome to NixOS 0.1pre33959-33963 (x86_64) - Linux 3.2.16 (tty1) >>>
backend2 login: mounting filesystems...
Alt Gr
10:47

# Deploying to EC2

```
{
  database =
    { deployment.targetEnv = "ec2";
      deployment.ec2.region = "us-east-1";
      services.postgresql.enable = true;
    };

  webserver =
    { deployment.targetEnv = "ec2";
      deployment.ec2.region = "eu-west-1";
      services.httpd.enable = true;
      services.httpd.documentRoot = ...;
    };
}
```

# NixOps — Deploying

```
$ nixops create -n foo ./network.nix
$ nixops deploy -d foo
```

This will:

- Create all machines
- Build/download all dependencies
- Upload them to the machines
- Activate any necessary services

To redeploy: just run `nixops deploy` again.

# Conclusion

## Why NixOS is awesome

- Completely declarative
- Atomic upgrades and rollbacks
- Reproducibility
- Multi-user package management
- NixOps extends this to networks and adds provisioning

## More information

- `http://nixos.org/`