



# NixOS

Linux dentro de 10 años

Manuel Palenzuela Merino  
[@github.com/Baitinq](https://github.com/Baitinq)

XXXV Jornadas Técnicas del GUL  
Abril 2023  
[gul.uc3m.es](http://gul.uc3m.es)

# Antes de empezar

1. Podeis preguntar dudas en cualquier momento
2. Cuantas mas preguntas mejor :))
3. Tambien se aceptan sugerencias durante la charla (mas alto, explicar algo mas a fondo, etc)
4. <https://github.com/Baitinq/nixos-presentation>

# Tabla de contenidos

## Distribuciones de Linux "tradicionales"

Problemas

## Que es Nix?

## Nix (Lenguaje)

Derivaciones

## Nixpkgs

## Nix (Package Manager)

## NixOS

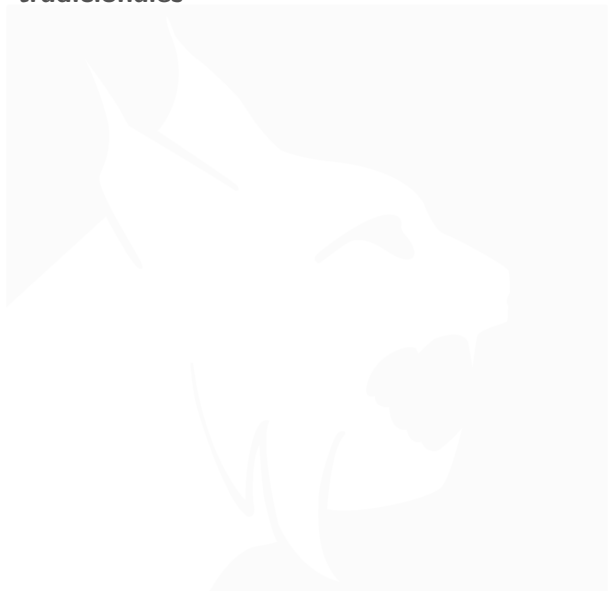
Highlights

Configuraciones

## El futuro de Linux

Mas informacion

## Preguntas



# Distribuciones de Linux "tradicionales"

Ej:

- ▶ Arch Linux
- ▶ Debian
- ▶ Fedora
- ▶ Ubuntu

Que son:

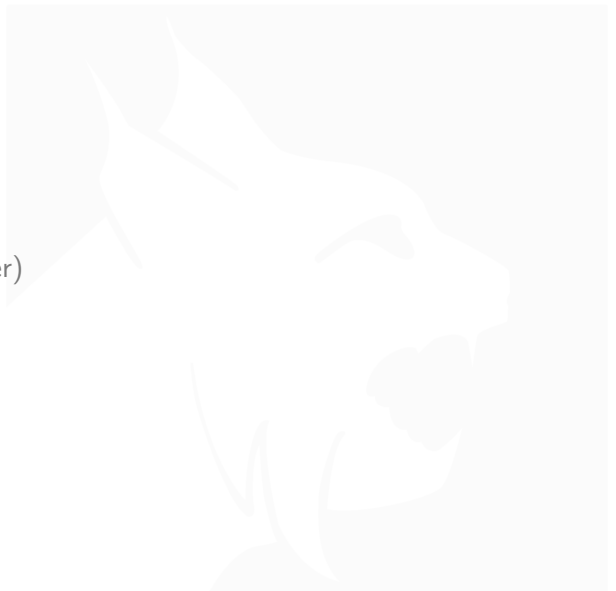
1. Package manager
2. Repositorios
3. Filesystem structure
4. Etc?

# Problemas de distros "tradicionales"

- ▶ Actualizaciones/cambios de configuración modifican destructivamente el estado del sistema (sobrescribiendo archivos en secuencia - **inconsistencia temporal**)
- ▶ Diferentes versiones de un binario
- ▶ Conflictos de paquetes
- ▶ No hay rollbacks
- ▶ No hay gestión de la configuración (dotfiles)
- ▶ **NO HAY DETERMINISMO**

# Que es Nix?

- I Nix (lenguaje)
- II Nixpkgs
- III Nix (package manager)



# Nix (Lenguaje)

- ▶ Es un DSL (no un GPL!)
- ▶ Usado para describir **derivaciones**
- ▶ Dynamically typed (por el momento)
- ▶ Lazy
- ▶ Funcional (no hay side-effects)
- ▶ Turing completo



# Nix (lenguaje)

```
let
  nombre = "GUL";
  saludar = x: "hola " + x + "!";
in {
  x = saludar nombre;
  z = 1;
}
```

```
nix-repl> { x = "hola GUL!"; z = 1; }
```



# Nix (lenguaje) - Derivaciones

```
nix-repl> d = derivation { name = "myname"; builder = "mybuilder";  
system = "mysystem"; }  
nix-repl> d  
derivation /nix/store/z3hhlxbckx4g3n9sw91nnvlkjvyw754p-myname.drv
```

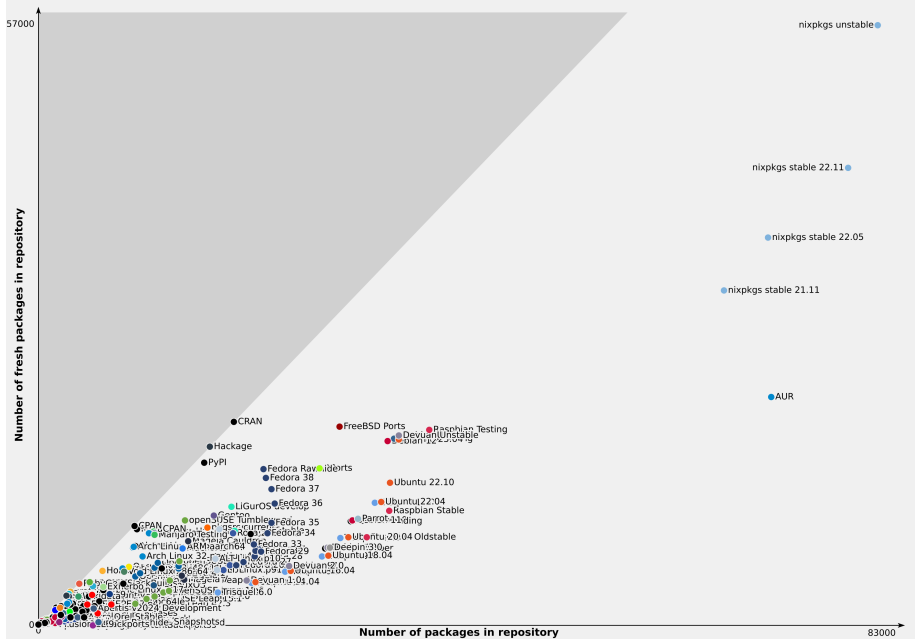
## hello.nix

```
with import <nixpkgs> {};  
stdenv.mkDerivation {  
  name = "hello";  
  src = ./hello-2.10.tar.gz;  
}
```

```
$ nix-build hello.nix  
/nix/store/6y0mzdarm5qxfafvn2zm9nr01d1j0a72-hello  
$ /nix/store/6y0mzdarm5qxfafvn2zm9nr01d1j0a72-hello/bin/hello  
Hello, world!
```

# Nixpkgs (the Nix packages collection)

- ▶ Repositorio en GitHub (Licencia FOSS)
- ▶ Contiene definiciones de paquetes (derivations)
- ▶ Tambien contiene tests, librerias, etc.
- ▶ Ramas diferentes (rolling: master, stable: release-YY.MM)
- ▶ Compilado y testado por Hydra (+=y subido al cache binario)
  
- ▶ Numero de paquetes  $\Rightarrow$  nixpkgs-unstable: **82918** vs AUR: **72357**
- ▶ Puesto numero **12** en lenguajes mas populares en GitHub (en cuanto a PRs)



## aircrack-ng.nix

```
{ lib, stdenv, fetchurl, libpcap, openssl, zlib, wireless_tools
, iw, ethtool, pciutils, libnl, usbutils }:
```

```
stdenv.mkDerivation {
  pname = "aircrack-ng";
  version = "1.7";

  src = fetchurl {
    url = "https://download.aircrack-ng.org/aircrack-ng-${version}.tar.gz";
    sha256 = "1hsq1gwmafka4bahs6rc8p98yi542h9a502h64bjlygpr3ih99q5";
  };

  buildInputs = [ libpcap openssl zlib libnl iw ethtool pciutils ];

  [...]

  meta = with lib; {
    description = "Wireless encryption cracking tools";
    homepage = "http://www.aircrack-ng.org/";
    license = licenses.gpl2Plus;
    maintainers = with maintainers; [ ];
    platforms = platforms.linux;
  };
}
```

# Nix (package manager)

- ▶ Un gestor de paquetes puramente funcional (y determinístico)
- ▶ Instalación de paquetes por usuarios sin privilegios
- ▶ Guarda los paquetes en la Nix store (/nix/store por defecto)
- ▶ Cada paquete tiene su propio UUID (/nix/store/\$hash\$version\$name)
- ▶ Permite instalar diferentes versiones de un mismo paquete
- ▶ Elimina totalmente problemas de dependencias
- ▶ Actualizaciones y rollbacks **atómicos**
- ▶ Instalable en cualquier distro + MacOS (homebrew replacement?)

## Ejemplos:

```
$ nix-env -i aircrack-ng
$ nix run vim
$ nix run "github:example/repo#package"
$ nix shell blender-bin#blender_2_83
```

Integra profundamente Nix (los 3 tipos) en una distribución de Linux!



# NixOS Highlights
















- ▶ Actualizaciones atómicas (software + configuración)
- ▶ Actualizaciones fiables (y rollbacks - configuración vinculada a la versión de software correcta + recargas/reinicios del servicio)
- ▶ Rollbacks
- ▶ Reproducible
- ▶ Configuración del OS **declarativa**
  - ▶ TODO tu sistema operativo en un repositorio git
- ▶ **DETERMINISMO**

# Mi Configuración

🔗 master ▾

Go to file

Code ▾

 <b>Baiting Update</b> ...		4 days ago	👁️ 415
 .git-crypt	Add 1 git-crypt collaborator		9 months ago
 Documentation	Misc: Update README and documentation		last month
 dotfiles	Dotfiles: Generalise weather location in ...		2 months ago
 hardware	Hardware: Laptop: Use latest kernel		2 months ago
 hosts	Home: Move gitcookies to netrc		last week
 modules	Update		3 months ago
 overlays	Update		last month
 packages	Emacs: Use lsp-bridge		2 months ago
 secrets	Secrets: Wireguard: Add another peer		4 days ago
 .gitattributes	Misc: Encrypt flake.lock		7 months ago
 LICENSE	Initial commit		10 months ago
 README.md	Misc: Update README and documentation		last month
 flake.lock	Update		4 days ago
 flake.nix	Flake: Set stateVersion in the hardware set		2 months ago

## About

My Personal Nix/NixOS Configuration.

[dotfiles](#) [nix](#) [nixos](#)  
[hacktoberfest](#) [nixos-configuration](#)  
[nix-flake](#) [nixos-config](#) [nix-config](#)

📖 Readme

📄 BSD-2-Clause license

☆ 49 stars

👁️ 1 watching

🔗 1 fork

Report repository

## Releases

No releases published

## Packages

No packages published



# Configuración mínima de NixOS

## configuration.nix

```
{  
  boot.loader.grub.device = "/dev/sda";  
  
  fileSystems."/".device = "/dev/sda1";  
  
  services.sshd.enable = true;  
}
```

```
$ nixos-rebuild switch -I nixos-config=configuration.nix  
$ nixos-rebuild switch --rollback  
$ nixos-rebuild test -I nixos-config=configuration.nix  
$ nixos-rebuild build-vm -I nixos-config=configuration.nix
```

# Configuraciones NixOS

## configuration.nix

```
{ config, pkgs, ... } :
{
  imports = [ ./hardware.nix ];

  environment.systemPackages = with pkgs; [ thunderbird firefox vim ];

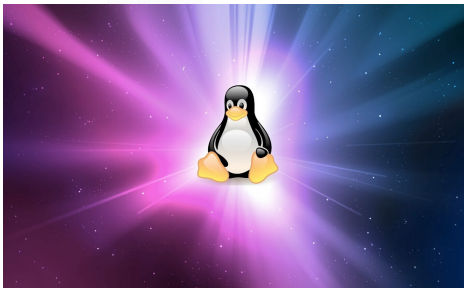
  users.users."alice" = {
    isNormalUser = true;
    home = "/home/alice";
    extraGroups = [ "wheel" "networkmanager" ];
  };

  services.xserver = {
    windowManager.i3.enable = true;
    displayManager.sddm.enable = true;
    videoDrivers = [ "nvidia" ];
  };

  networking.firewall.enable = true;
}
```

# El futuro de Linux

- ▶ NixOS tiene problemas
- ▶ Alternativas:
  - ▶ Fedora Silverblue?
  - ▶ Seguir como estamos?



# Mas informacion

- ▶ NixOS modules
  - ▶ Overriding
    - ▶ Flakes
    - ▶ Etc..

- ▶ NixOS website
- ▶ Nix Manuals
- ▶ **Nix Pills**
- ▶ Learn X in Y minutes, where  $X=nix$



# Gracias por escuchar!



- ▶ Preguntas
- ▶ Feedback

<https://github.com/Baitingq/nixos-config>