Séminaire recherche reproductible
25/11/2021 - Batiment IMAG

Le gestionnaire de paquets NIX

GRICAD - Equipe systèmes
Bruno Bzeznik

November 25, 2021

GRENOBLE ALPES RECHERCHE
INFRASTRUCTURE DE CALCUL INTENSIF ET DE DONNÉES

Sommaire

Le gestionnaire de paquets NIX

Vue d'ensemble
The Purely Functional Package Manager

Nix

- Gestionnaire de paquets fonctionnel
- Fiable & reproductible
- Dispo sur Linux & MAC OS
- Les utilisateurs peuvent créer/installer un paquet sans passer `root`

Nixpkgs

- plus de 80000 paquets
- Pur: pas de dependances en dehors du "Nix store"

NixOS

The Purely Functional Linux Distribution

Vue d'ensemble
The Purely Functional Package Manager

Nix

- Gestionnaire de paquets fonctionnel → Pas d'effet de bord
- Fiable & reproductible (Expérimentation, recherche)
- Dispo sur Linux & MAC OS → Un même paquet pour Tier0/1/2/3
- Les utilisateurs peuvent créer/installer un paquet sans passer `root` → Facilité de la personnalisation d'env.

Nixpkgs

- plus de 80000 paquets (Communauté)
- Pur: pas de dependances en dehors du "Nix store"

NixOS

The Purely Functional Linux Distribution → Pour aller plus loin

Le store et les hash

Stockage des paquets

Unique répertoire `/nix/store`

→ Pas de pollution de l'arborescence du système

Le store et les hash

Stockage des paquets

Unique répertoire `/nix/store`

→ Pas de pollution de l'arborescence du système

Identification des paquets via hash

Construction du paquet → unique sous-répertoire

`/nix/store/an9dli66ng2jzvqf13b2i230mm9fq7qk-cdo-1.7.2`

$$Hash = mix(sources + deps + flags, ...)$$

Compilation + nouvelle option → nouveau hash
`/nix/store/srf6grrfy9vkc9fsplk8xk292lm8jvz5-cdo-1.7.2`

→ Conservation de l'arbre des dépendances,
→ Unicité du paquet

Environnement utilisateur

- `PATH=~/.nix-profile/bin:$PATH`
- Installation d'un paquet = Création de liens dans le profil de l'utilisateur stocké dans son home
- Commande `nix-env` pour gérer les opérations sur l'environnement (`nix-env -i <paquet>` pour installer un paquet)

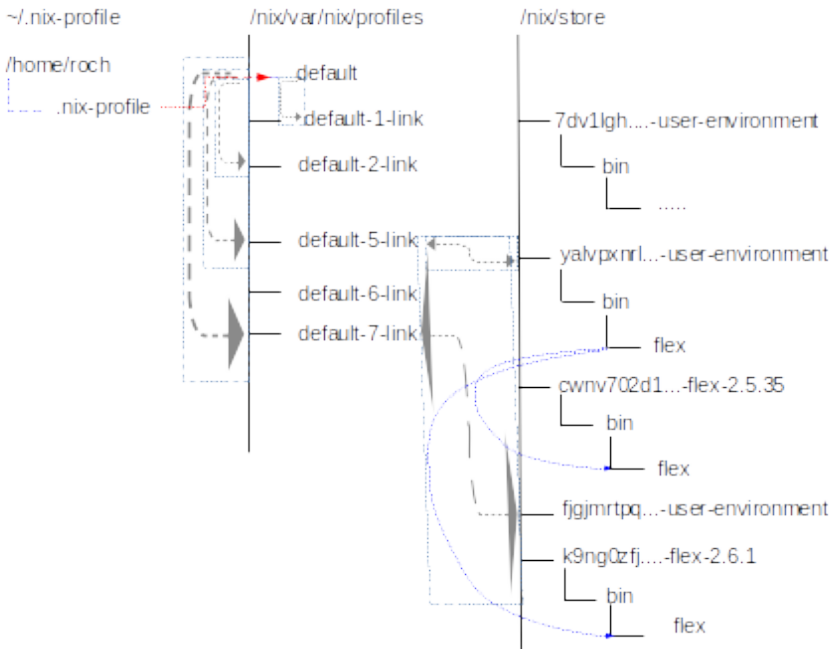
```
~$ nix-env -i hello

~$ readlink ‘which hello‘
/nix/store/3dlqv87hrrfjynj0brbn4h71g4g4g89z-hello-2.10/bin/hello

~$ ldd /nix/store/3dlqv87hrrfjynj0brbn4h71g4g4g89z-hello-2.10/bin/hello
        linux-vdso.so.1 (0x00007ffe5d1b6000)
        libc.so.6 =>
/nix/store/q3wx1gab2ysnk5nyvyyg56ana2v4r2ar-glibc-2.24/lib/libc.so.6
(0x00007f3d90bc9000)
/nix/store/q3wx1gab2ysnk5nyvyyg56ana2v4r2ar-glibc-2.24/lib/ld-linux-x86-64.so.2
(0x00007f3d90f67000)
```

Les profils

Profile

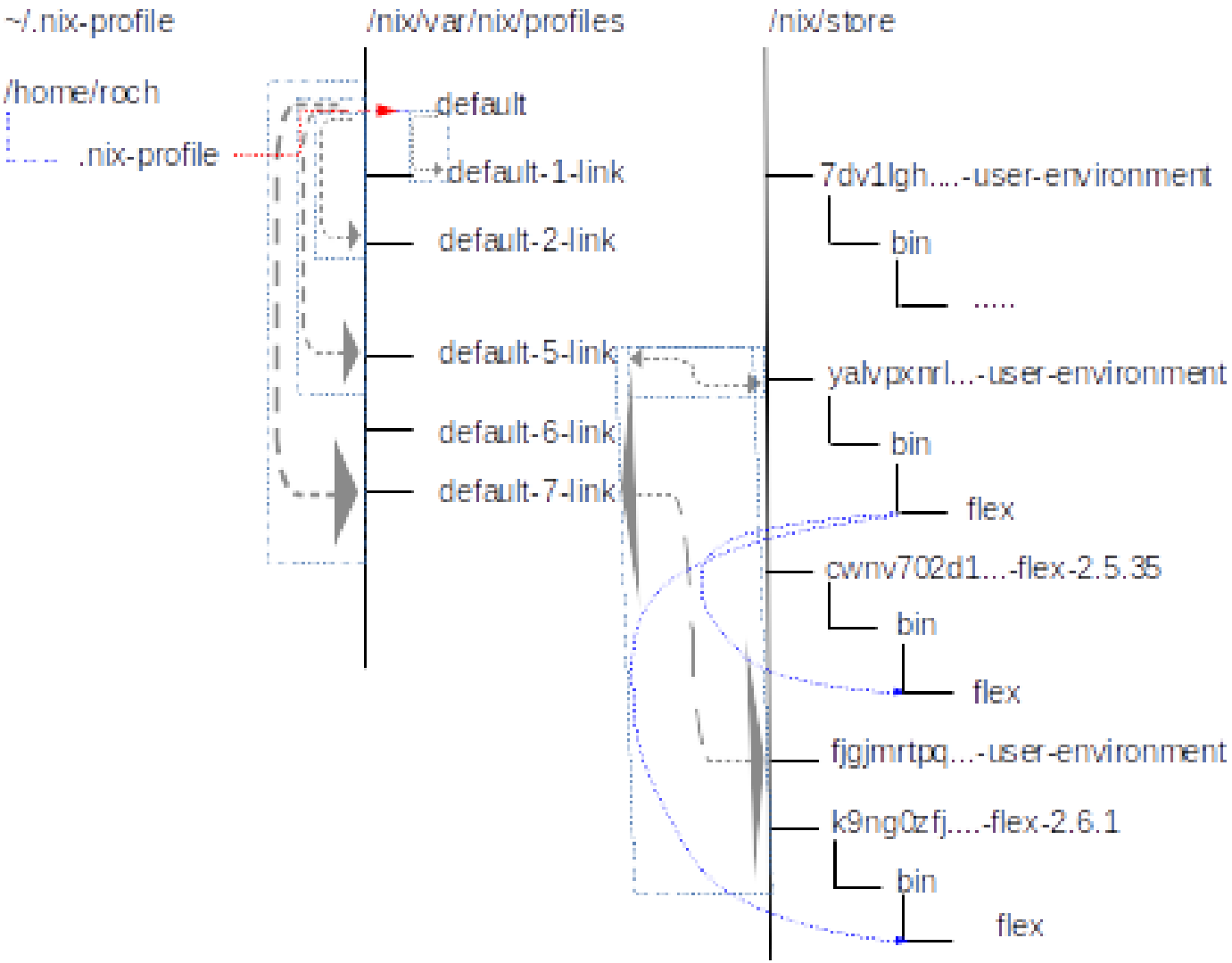
Environnement défini par un **ensemble de liens symboliques** dans le répertoire personnel de l'utilisateur, vers les paquets utilisés dans le store.

```
nix-env --switch-profile $NIX_USER_PROFILE_DIR/new_profile
```



- **Nombre illimité et cohabitation** de profiles
- **Suppression, mise à jour d'un profile**
- **Rollbacks** et historique des générations

Les profils


~/.nix-profile
/nix/var/nix/profiles
/nix/store
/home/roch
.nix-profile
default
default-1-link
default-2-link
default-5-link
default-6-link
default-7-link
7dv1lgh....-user-environment
bin
.....
yalvpxnrl...-user-environment
bin
flex
cwnv702d1...-flex-2.5.35
bin
flex
fjgjmrtpq...-user-environment
k9ng0zfj....-flex-2.6.1
bin
flex

Le garbage collector

- Lancé par l'administrateur,il va eviter le gonflement du store
- Les profils des utilisateurs permettent au garbage collector de savoir ce qui est actuellement utilisé ou pas

Attention

Pour que les paquets d'un profil ne soient pas collectés (effacés), le profil doit bien être créé dans le répertoire pointé par la variable **$NIX_USER_PROFILE_DIR**

Channels

channel

Une channel correspond à une version du dépot de Nixpkgs
Ex. : `nixos-21.05` , `nixos-20.09` , `nixpkgs-unstable`
On utilise la variable NIX_PATH pour spécifier une ou plusieurs channels.

Exemple: Installation du paquet `openmpi` en spécifiant une channel différente

```
$ NIX_PATH="nixpkgs=channel:nixos-20.03" nix-env -i -A openmpi
installing 'openmpi-4.0.2'
$ NIX_PATH="nixpkgs=channel:nixos-21.05" nix-env -i -A openmpi
replacing old 'openmpi-4.0.2'
installing 'openmpi-4.1.1'
```

Liste des channels: `https://channels.nix.gsc.io/`

Binary cache

`binary cache`

1. Les paquets pré-compilés sont récupérés depuis un `binary-cache`
2. Si le paquet n'est pas dans un cache, il est automatiquement construit
3. Même chose pour toutes les dépendances d'un paquet

Les Attributs

Un même paquet peut comporter plusieurs attributs. Ex avec `gromacs`

```
$ nix search gromacs
* nixpkgs.gromacs (gromacs)
Molecular dynamics software package
* nixpkgs.gromacsDouble (gromacs)
Molecular dynamics software package
* nixpkgs.gromacsDoubleMpi (gromacs)
Molecular dynamics software package
* nixpkgs.gromacsMpi (gromacs)
Molecular dynamics software package
```

`attribut`

C'est pourquoi on fait généralement les installation en utilisant l'option `-A`:

`nix-env -i -A gromacsMpi`

NUR

Nix User Repository

NUR permet aux utilisateurs d'avoir leur propre dépot de paquets, hors des paquets officiels nixpkgs.
GRICAD a son dépot NUR:
`https://github.com/Gricad/nur-packages`

Exemple:

```
$ nix-env -i -A nur.repos.gricad.osu-micro-benchmarks
```

Une particularité du dépot NUR de GRICAD est qu'il fournit des paquets des **compilateurs Intel** qui sont sous Licence et donc non disponibles publiquement. Seule la définition des paquets y est disponible et l'installation des paquets binaires n'est possibles que depuis les infrastructures Gricad, qui sont connectées sur un binary-cache local.

Nix-shell

nix-shell

Permet d'entrer dans l'environnement de développement d'un paquet. Utile pour lancer une compilation simple (à éviter dans les cas plus complexes pour lesquels il vaut mieux créer un paquet). Cela peut aussi être utile pour lancer une commande d'un paquet sans l'installer dans son profil.

Ex:

```
$ nix-shell -p openmpi
[nix-shell:~]$ mpicc hello_mpi.c -o hello_mpi
[nix-shell:~]$ exit
$ nix-shell -p cowsay --run "cowsay hello"
```

```
 -------
< hello >
 -------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||
```

Dérivation, expression, paquet

Expression

Exemple d'**expression** définissant une **dérivation** permettant d'instancier un **paquet**

```
{ stdenv, fetchurl, cmake,
  singlePrec ? true,
  mpiEnabled ? false,
  fftw,
  openmpi
}:
stdenv.mkDerivation {
  name = "gromacs-4.6.7";

  src = fetchurl {
    url = "ftp://ftp.gromacs.org/pub/gromacs/gromacs-4.6.7.tar.gz";
    sha256 = "6afb1837e363192043de34b188ca3cf83db6bd189601f2001a1fc5b0b2a214d9";
  };
  buildInputs = [cmake fftw]
  ++ (stdenv.lib.optionals mpiEnabled [ openmpi ]);

  cmakeFlags = ''
    ${if singlePrec then "-DGMX_DOUBLE=OFF" else "-DGMX_DOUBLE=ON -DGMX_DEFAULT_SUFFIX=OFF"}
    ${if mpiEnabled then "-DGMX_MPI:BOOL=TRUE
                          -DGMX_CPU_ACCELERATION:STRING=SSE4.1
                          -DGMX_OPENMP:BOOL=TRUE
                          -DGMX_THREAD_MPI:BOOL=FALSE"
                    else "-DGMX_MPI:BOOL=FALSE" }
  '';
  meta = with stdenv.lib; {
    homepage    = "http://www.gromacs.org";
    license     = licenses.gpl2;
    description = "Molecular dynamics software package";
    platforms = platforms.unix;
  };
}
```

Flakes

Feature à venir et à suivre de près pour ce qui concerne la reproductibilité: possibilité de pinning d'un environnement sur un numéro de commit

`https://www.tweag.io/blog/2020-05-25-flakes`

Liens utiles

Biblio officielle

Nix: `https://nixos.org/learn.html`
Nix pillsl: `https://nixos.org/guides/nix-pills`

Références GRICAD

Doc: `https://gricad-doc.univ-grenoble-alpes.fr/hpc/softenv/nix/`

Autres

Tutoriel: `https://nix-tutorial.gitlabpages.inria.fr/nix-tutorial`
Discourse: `https://discourse.nixos.org/`
Nixpkgs sources: `https://github.com/NixOS/nixpkgs`

NIX SUR LES INFRASTRUCTURES GRICAD

Chargement de l'environnement Nix


```
$ source /applis/site/nix.sh
$ echo $NIX_PATH
nixpkgs=channel:nixos-21.05
```

Lors du premier chargement, la channel stable actuelle est fixée pour tous les chargements ultérieurs. Utiliser l'option `-help` pour changer de channel ou modifier votre `$NIX_PATH`

Démo et doc

- Démo: construisons un paquet!
- Doc / tuto: `https://gricad-doc.univ-grenoble-alpes.fr/hpc/softenv/nix/`