Conda on NixOS

Jaakko Luttinen — 2017-05-27 16:36 — 3 Comments

We've been using Conda (https://conda.io/) (also known as Anaconda[1] or Miniconda[2]) to manage our Python environments at Leanheat (http://leanheat.com/). However, Conda doesn't work on NixOS (https://nixos.org/), a Linux distribution I recently started using. After some studying, I found a way to install and use Conda on NixOS, which I thought I'd share here.

NixOS is quite different from other Linux distributions: the system (including applications and services) is configured by writing a specification which is then "activated". This has a lot of benefits in system maintenance. One of the main differences in NixOS to traditional Linux distributions is the lack of many standard directories such as `/usr/bin` and `/usr/lib`. Instead, all packages are kept in a separate directory called Nix store and environments are created by adding the selected binaries to the search path. Thus, it is possible to have different versions of packages in Nix store and use different sets of packages in different environments.

However, using third-party binaries such as Conda on NixOS turns out to be non-trivial because the binaries expect to find the dynamic linker in `/lib64/ld-linux-x86-64.so.2`, which doesn't exist on NixOS. It is relatively straightforward to fix this kind of issues by using PatchELF (https://nixos.org/patchelf.html) to patch the binaries. This approach usually works with third-party binaries but it becomes a mess with Conda because it is a package manager which installs and upgrades packages on its own. One would need to patch all the packages that Conda installs and I'm not sure there is an easy way to achieve that.

Fortunately, there is another way to use Conda on NixOS: instead of patching the binaries, create an environment which is FHS compatible so that it has the standard directories for system libraries and binaries. This solution was suggested for Steam in a Sander van der Burg's blog post (http://sandervanderburg.blogspot.fi/2013/09/composing-fhs-compatible-chroot.html). I recommend you read that post if you are interested in more details. FHS compatible environments can be created on NixOS with a function `buildFHSUserEnv`. I wrote a nix expression defining an environment that enables Conda in `~/.conda-shell.nix`:

```
{ pkgs ? import <nixpkgs> {} }:

let

  # Conda installs it's packages and environments under this directory
  installationPath = "~/.conda";

  # Downloaded Miniconda installer
  minicondaScript = pkgs.stdenv.mkDerivation rec {
    name = "miniconda-${version}";
    version = "4.3.11";
    src = pkgs.fetchurl {
      url = "https://repo.continuum.io/miniconda/Miniconda3-${version}-Linux-x86_64.sh";
      sha256 = "1f2g8x1nh8xwcdh09xcra8vd15xqb5crjmpvmc2xza3ggg771zmr";
    };
    # Nothing to unpack.
    unpackPhase = "true";
    # Rename the file so it's easier to use. The file needs to have .sh ending
    # because the installation script does some checks based on that assumption.
    # However, don't add it under $out/bin/ becase we don't really want to use
    # it within our environment. It is called by "conda-install" defined below.
    installPhase = ''
      mkdir -p $out
      cp $src $out/miniconda.sh
    '';
    # Add executable mode here after the fixup phase so that no patching will be
    # done by nix because we want to use this miniconda installer in the FHS
    # user env.
    fixupPhase = ''
      chmod +x $out/miniconda.sh
    '';
  };

  # Wrap miniconda installer so that it is non-interactive and installs into the
  # path specified by installationPath
  conda = pkgs.runCommand "conda-install"
    { buildInputs = [ pkgs.makeWrapper minicondaScript ]; }
    ''
      mkdir -p $out/bin
      makeWrapper                   \
        ${minicondaScript}/miniconda.sh     \
        $out/bin/conda-install          \
        --add-flags "-p ${installationPath}" \
        --add-flags "-b"
    '';

in
(
  pkgs.buildFHSUserEnv {
    name = "conda";
    targetPkgs = pkgs: (
```

```
      with pkgs; [

        conda

        # Add here libraries that Conda packages require but aren't provided by
        # Conda because it assumes that the system has them.
        #
        # For instance, for IPython, these can be found using:
        # `LD_DEBUG=libs ipython --pylab`
        xorg.libSM
        xorg.libICE
        xorg.libXrender
        libselinux

        # Just in case one installs a package with pip instead of conda and pip
        # needs to compile some C sources
        gcc

        # Add any other packages here, for instance:
        emacs
        git

      ]
    );
    profile = ''
    # Add conda to PATH
    export PATH=${installationPath}/bin:$PATH
    # Paths for gcc if compiling some C sources with pip
    export NIX_CFLAGS_COMPILE="-I${installationPath}/include"
    export NIX_CFLAGS_LINK="-L${installationPath}lib"
    # Some other required environment variables
    export FONTCONFIG_FILE=/etc/fonts/fonts.conf
    export QTCOMPOSE=${pkgs.xorg.libX11}/share/X11/locale
  '';
  }
).env
```

This environment can be activated with `nix-shell ~/.conda-shell.nix`. It adds an executable `conda-install` which should be run in order to install Miniconda under `~/.conda`. After running `conda-install`, Conda can be used normally. For instance, create a new environment with `conda create -n myenv ipython numpy` and activate an environment with `source activate myenv`. The "Conda shell" can be closed by typing simply `exit`.

To make the activation of the environment even simpler in Bash, I added to `~/.bashrc`:

```
function conda-shell {
    nix-shell ~/.conda-shell.nix
}
```

Now the environment can be activated by writing `conda-shell`.

Some random thoughts:

- Although I think Nix is in many ways better than Conda, I still feel Conda makes it easier to install specific versions of (Python) packages and to maintain identical setups among non-NixOS systems. Also, if a team uses Conda, it is a bit difficult to make the others use Nix so I just needed to find a way to use Conda on

NixOS.

- I didn't want the activation of the Conda shell environment to have any side effects on the system so I didn't install Miniconda automatically during the activation but added an explicit installer instead.
- If I want to use the Conda environments in Emacs, I need to launch Emacs from the "Conda shell" in terminal. Thus, I added Emacs to the list of installed packages in Conda shell.

As I'm new to NixOS, there must be many possible improvements to this solution. I'd be glad to hear about any such ideas. I'd be interested to know if this could be modified for inclusion in `nixpkgs`, thus making Conda easily accessible to all NixOS users.

Updated 2017-05-27: Added some libraries and environment variables to the configuration in order to support Spyder.

---

1. Anaconda is a comprehensive Conda installer containing Python, Conda and a large number of other packages. ↩
2. Miniconda is a minimal Conda installer containing Python, Conda and their dependencies. ↩

Conda **NixOS**
