Využití Nix/NixOps pro průběžnou integraci a nasazení software při vývoji

Tomáš Vlk
vedoucí práce: RNDr. Marek Rychlý, Ph.D.

BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

vlktomas/nix-examples

Motivace

Pro **spolehlivé otestování** a **nasazení** aplikace je potřeba eliminovat problémy způsobené odlišným prostředím. Drobné odchylky mezi prostředími mohou způsobit, v lepším případě chybu při zpracovávání CI/CD serverem, ale v horším případě nefunkční nasazenou aplikaci na produkčním serveru. Problémy spojené s prostředím se často řeší pomocí **kontejnerizace** aplikací.

Používání kontejnerů ale není ideální. Ve spoustě případů není potřeba aplikace v systému kompletně izolovat do samostatných kontejnerů. **Při použití Nix/NixOps** technologií jsou problémy způsobené odlišným prostředím prakticky eliminovány **bez nutnosti kontejnerizace**. Díky čistě funkcionálnímu jazyku je implementace CI/CD velmi jednoduchá a celý proces je navíc **reprodukovatelný**.

Co je Nix

Nix – **funkcionální / source based / bezstavový** správce balíčků.

- reprodukovatelnost
- atomičnost instalace
- explicitní závislosti

NixOS – linuxová distribuce

- deklarativní popis celého systému
- modulární, reprodukovatelný

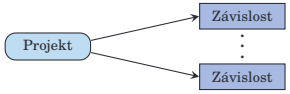
NixOps – IaC

- deklarativní popis celé infrastruktury

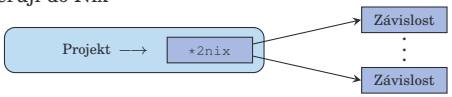
Explicitní závislosti mezi balíčky

Problém dodání závislostí

1 závislosti jsou v Nix



2 závislosti se generují do Nix



3 všechny závislosti jako jeden balíček

Výstupem práce je sada příkladů použití Nix/NixOps

Nástroj pro správu závislostí	1	2	3
Autotools	✓		
Go modules			✓
Cabal	✓	✓	
Maven			✓
Gradle			✓
NPM		✓	✓
Composer			✓
Pip	✓		

Vlastnosti použití Nix/NixOps pro CI/CD

Vývoj

+ nejsou opomenuty závislosti
+ stejný jazyk pro všechno
+ Nix vs Docker vs Vagrant

Sestavení

+ reprodukovatelnost
+ možnost různých variant
– v některých případech závislosti jako jeden balíček

Testování

+ žádný rozdíl mezi test/prod
+ testovací framework
– ale pouze pro NixOS

Release

+ různé instalační soubory
+ minimální image
+ cross-compilation
– pomalé pro deb a rpm

Nasazení

+ testování nasazení
+ spolehlivé, bezpečné
+ možnost rollbacku

Integrace Nix s klasickými CI/CD nástroji

+ jeden formát zápisu pipeline
+ reprodukovatelnost
– problém s virtuálními stroji

Nix – správce balíčků

```
{ stdenv, fetchurl, someDependency }:

stdenv.mkDerivation rec {
  pname = "example";
  version = "1.0";

  src = fetchurl {
    url =
  "https://example.org/${pname}-${version}";
    sha256 = "0ssi1wpafc...7c9lng89nd";
  };

  buildInputs = [ someDependency ];

  buildPhase = ''
    gcc example.c -o example
  '';
}
```

NixOS – linuxová distribuce

```
{ config, pkgs, ... }: {

  users.users.tom = {
    home = "/home/tom";
  };


  fileSystems."/mnt" = {
    fsType = "ext4";
    device = "/dev/sda1";
  };

  services.openssh.enable = true;

  environment.systemPackages = with pkgs; [
    vim
  ];
}
```

NixOps – IaC

```
{
  webserver = {
    services.httpd.enable = true;
    services.httpd.virtualHosts = {
      "example.org" = {
        documentRoot = "/data";
    };
    fileSystems."/data" = {
      fsType = "nfs4";
      device = "fileserver:/";
    };
  };

  fileserver = {
    services.nfs.server.enable = true;
    services.nfs.server.exports = "...";
  };
}
```