Nix/NixOS & secrets

> Jakub Vokoun
>
> jakub.vokoun@wftech.cz

Motivace

- **deklarativní**, **reprodukovatelná**, **bezpečná** a **verzovaná** konfigurace systému
- zakládní prvky:
    - NixOS
    - Flakes
    - Home Manager
    - sops-nix

Nix 101

Nix

- jazyk
- CLI nástroj
- operační systém

Nix 101

Jazyk Nix

- dynamicky typovaný
- funkcionální
- lazy evaluation
- DSL (Domain Specific Language)

Nix 101

Flakes

> Nix flakes provide a standard way to write Nix expressions (and therefore packages) whose dependencies are version-pinned in a lock file, improving reproducibility of Nix installations. (https://nixos.wiki/wiki/Flakes)

- `flake.nix`
- `falke.lock`

Nix 101

Flakes

```
nix flake init
```

```
nix flake init --experimental-features 'nix-command flakes'
```

```
cd /etc/nixos
sudo nix flake init --template github:username/flake-starter-config
```

Nix 101

Flakes

```nix
{
  description = "Nixos config flake";
  inputs = { nixpkgs.url = "github:nixos/nixpkgs/nixos-unstable"; };
  outputs = { self, nixpkgs, ... }@inputs: {
    nixosConfigurations.default = nixpkgs.lib.nixosSystem {
      specialArgs = { inherit inputs; };
      modules = [
        ./configuration.nix
        #./my-module.nix
      ];
    };
  };
}
```

Nix 101

Flakes

```
sudo nixos-rebuild switch --flake /etc/nixos/#default
```

Nix 101

Flakes

```
nix flake info
```

```
nix flake metadata foo --json | jq .
```

```
nix flake metadata foo
```

```
nix flake update
```

SOPS 101

- Secrets OPerationS
- formáty: **YAML**, JSON, ENV, INI, BINARY
- šifruje pomocí: AWS KMS, GCP KMS, Azure Key Vault, PGP, **age**
- široké spektrum použití:
  - CLI
  - Terraform provider
  - k8s operátor
  - …
  - sops-nix

SOPS 101

```
nix-shell -p age sops
```

```
age-keygen -o age-key.txt
age-keygen -y age-key.txt
```

SOPS 101

```yaml
# .sops.yaml
keys:
  - &primary age13tl7p3xy6fwxgwfp5dtflpm7teag56mwy932xka4d2ujcfe9weusttlm9p
creation_rules:
  - path_regex: secrets.yaml$
    key_groups:
    - age:
      - *primary
```

```bash
export SOPS_AGE_KEY_FILE=$(pwd)/age-key.txt
sops edit secrets.yaml
```

sops-nix 101

```
# configuration.nix
{ pkgs, inputs, config, ... }: {
  imports = [ inputs.sops-nix.nixosModules.sops ];

  sops.defaultSopsFile = /home/user/repos/secrets/secrets.yaml;
  sops.defaultSopsFormat = "yaml";

  sops.age.keyFile = "/home/user/.config/sops/age/keys.txt";

  sops.secrets.example-key = { };
  sops.secrets."myservice/my_subdir/my_secret" = { owner = "sometestservice"; };
}
```

sops-nix 101

```
# flake.nix
{
  description = "System configuration flake";
  inputs = {
    nixpkgs.url = "github:nixos/nixpkgs/nixos-unstable";
    sops-nix.url = "github:Mic92/sops-nix";
  };
  outputs = { self, nixpkgs, ... }@inputs:
    let pkgs = nixpkgs.legacyPackages.x86_64-linux;
    in {
      nixosConfigurations = {
        default = nixpkgs.lib.nixosSystem {
          specialArgs = { inherit inputs; };
          modules = [ ./configuration.nix ];
        };
      };
    };
}
```

sops-nix 101

```nix
# flake.nix
{
  description = "System configuration flake";
  inputs = {
    nixpkgs.url = "github:nixos/nixpkgs/nixos-unstable";
    sops-nix.url = "github:Mic92/sops-nix";
    home-manager.url = "github:nix-community/home-manager";
  };
  outputs = { self, nixpkgs, home-manager, sops-nix, ... }@inputs:
    let pkgs = nixpkgs.legacyPackages.x86_64-linux;
    in {
      nixosConfigurations = {
        default = nixpkgs.lib.nixosSystem {
          specialArgs = { inherit inputs; };
          modules = [
            ./configuration.nix
            home-manager.nixosModules.default
            {
              home-manager.sharedModules = [ sops-nix.homeManagerModules.sops ];
              home-manager.useUserPackages = true;
              home-manager.users.vagrant = import ./home.nix;
            }
          ];
        };
        inputs.nixpkgs.follows = "nixpkgs";
      };
    };
}
```

sops-nix 101

```nix
# home.nix
{ inputs, lib, config, pkgs, ... }: {
  sops = {
    age.keyFile = "/home/vagrant/flake-config/keys.txt";

    defaultSopsFile = ./secrets.yaml;

    secrets.secret_api_key = {
      path = "${config.sops.defaultSymlinkPath}/secret_api_key";
    };
  };

  programs.bash = {
    enable = true;
    bashrcExtra = ''
      export SECRET_API_KEY="$(cat ${config.sops.secrets.secret_api_key.path})"
    '';
  };
}
```

sops-nix 101

```
let
  gh-wrapped = pkgs.writeShellScriptBin "gh" ''
    ${pkgs.gh}/bin/gh --token $(cat ${config.sops.secrets.github-token.path}) $@
  '';
in
environment.systemPackages = [
  gh-wrapped
];
```

sops-nix 101

```
{
  sops.templates."service.env" = {
    content = ''
      API_KEY=${config.sops.placeholder."service/api/key"}
      API_SECRET=${config.sops.placeholder."service/api/secret"}
    '';
  };
}
```

```
cat /run/secrets-rendered/service.env
API_KEY=my-api-key
API_SECRET=my-api-secret
```

Demo

Refrence

- https://nix-community.github.io/home-manager/
- https://github.com/nix-community/home-manager
- https://github.com/Mic92/sops-nix
- https://github.com/getsops/sops