Host a DevOps exam using NixOS

Deploy a Bastion host and a KVM virtual machine per student in order for them to deploy a website which will be auto-validated.

Yvan SRAKA, Ryan LAHFA

Host a DevOps exam using NixOS

- DevOps students have to test their skills in a real-world exam, what's better than scripting it using NixOps in order to reuse it?
- They will be provided with their own user account on the Bastion and access to a jump user account, WireGuard will be auto-configured on the host to enable access to their KVM host directly.
- The main point is controlling the KVM guests "declaratively" and giving choice to the students to choose their guest OS: Debian or NixOS.
- We will see how we can easily generate Nix expression from a scripting language, e.g. Python, and feed it to a NixOps deployment.

Disclaimer: *all this is incomplete and experimental!*

This was tried during a real exam this summer, but it's very alpha-quality and had to get a lot of last-minute fixes due to external issues (e.g. disks dying…)

Git repository

The full code can be found here[1], it's still a bit messy...

`\nix` folder contains files on which will focus this talk:

- `student-setup.nix` is generated by `generate-setup.py` (that rely on `nix-expr.py`) from students data.
- `deploy.sh` push on our setup `exam.nix`, `student.nix`, `kvm-guests.nix` and `student-setup.nix`

[1] https://git.newtype.fr/yvan/devops-exam-model

deploy.sh

```
##!/usr/bin/env bash

echo "[+] Regenerating of the setup"
python3 generate_setup.py

echo "[+] Sending Nix files"
rsync --inplace --temp-dir=/tmp -avPz *.nix \
    yvan@bastion:/etc/nixos/

echo "[+] Rebuilding of the exam machine"
ssh -t yvan@bastion "sudo nixos-rebuild switch"
```

generate_setup.py *(subset of)*

```
    def students(csv_filename):
        with open(csv_filename, 'r', newline='') as csvfile:
            ereader = csv.reader(csvfile, delimiter=';')
            next(ereader) # exhaust header.
            for index, student in enumerate(ereader):
                name, surname, email, username = student
                wg_pubkey, wg_privkey = wireguard_parameters()
                keys = read_keys(username)
                yield {
                    "surname": surname,
                    "name": name,
                    "email": email,
                    "username": username,
                    "keys": keys,
                    "wireguardPublicKey": wg_pubkey,
                    "wireguardPrivateKey": wg_privkey,
                }
```

student-setup.nix *(generated)*

```
  ltorvalds = {
      email = "torvalds@linux-foundation.org";
      guestOperatingSystem = "debian";
      index = 42;
      keys = [
        "ssh-ed25519 ... torvalds@linux-foundation.org"
      ];
      name = "Linus";
      surname = "Torvalds";
      username = "ltorvalds";
      wireguardPrivateKey = "0ECE2Js+RkxQVTyJ9BvZB0DjpEGnWMy
      wireguardPublicKey = "3o9Dhmrrql/5PZEhi5kS+Fob1m8rN70S
    };
  rstallman = { ... }
```

student.nix *(subset of)*

```
  mkGuest = name: student: {
    memory = "1G";
    netDevice = "tap${toString student.index}";
    vncDisplay = "localhost:${toString student.index}";
    operatingSystem = student.guestOperatingSystem;
  };

  services.kvmGuests = {
    enable = true;
    guests = mapAttrs mkGuest cfg.students;
  };

  # Create users for each student + management/jump accounts
  users.users = (mapAttrs mkBastionUser cfg.students) // ({
    jump = mkJumpUser cfg.students;
    admin = adminUser;
  });
```

Q/A