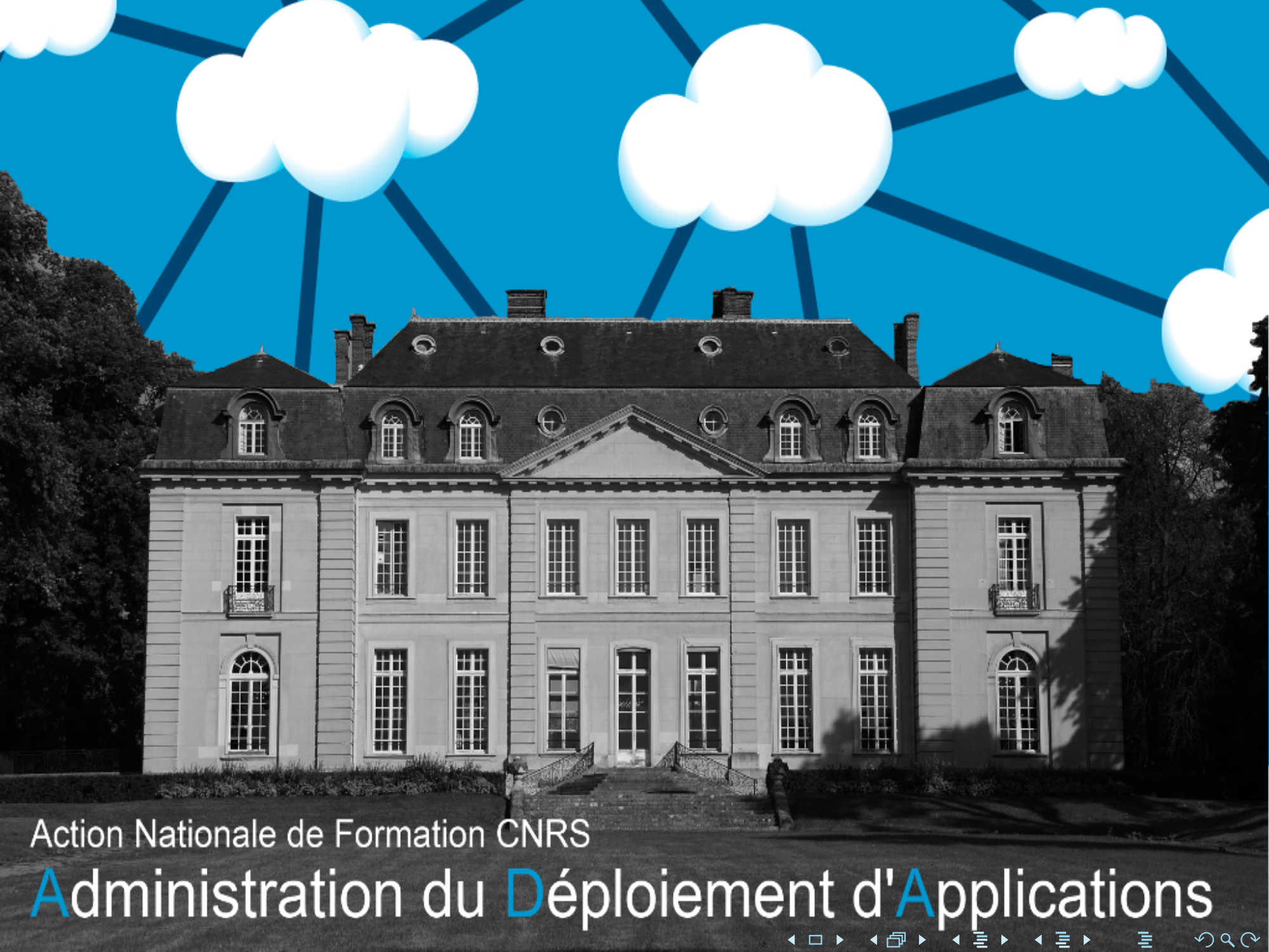

Action Nationale de Formation CNRS
Administration du Déploiement d'Applications

Infrastructure as Code: Préparation d'image serveur

Tristan Le Toullec

CNRS/LOPS

01 Octobre 2019

Outline

Types d'outils

Outils pour la configuration d'un système et le déploiement d'une application.

- **Script de configuration**
  - shell, python
- **Configuration Management Tools**
  - Ansible, Puppet, Chef
- **Templating tools**
  - Packer, docker, buildha
- **Provisionning**
  - Terraform, docker, Ansible , foreman

Méthodes de configuration

Scriptée

Non idempotent
Gestion de l'échec
Lecture difficle
Configuration drift

Grande latitude

CMT (Puppet. . .)

Configuration drift

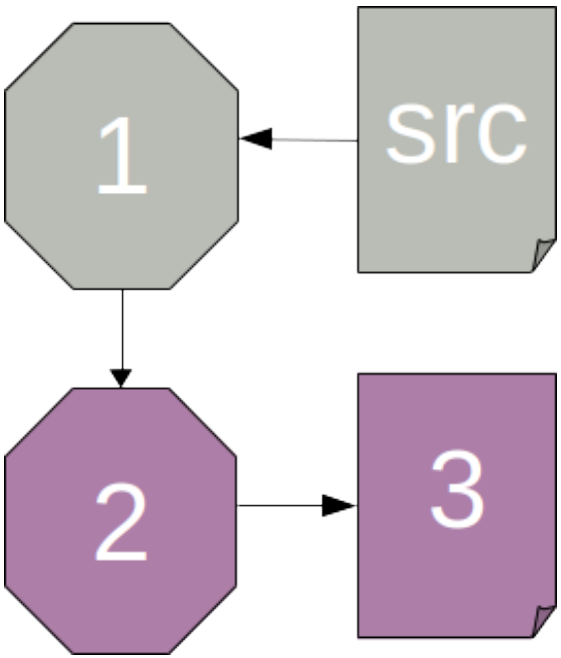
Idempotent
Lecture facile
Gestion de l'échec

Templating tools

Immutabilité

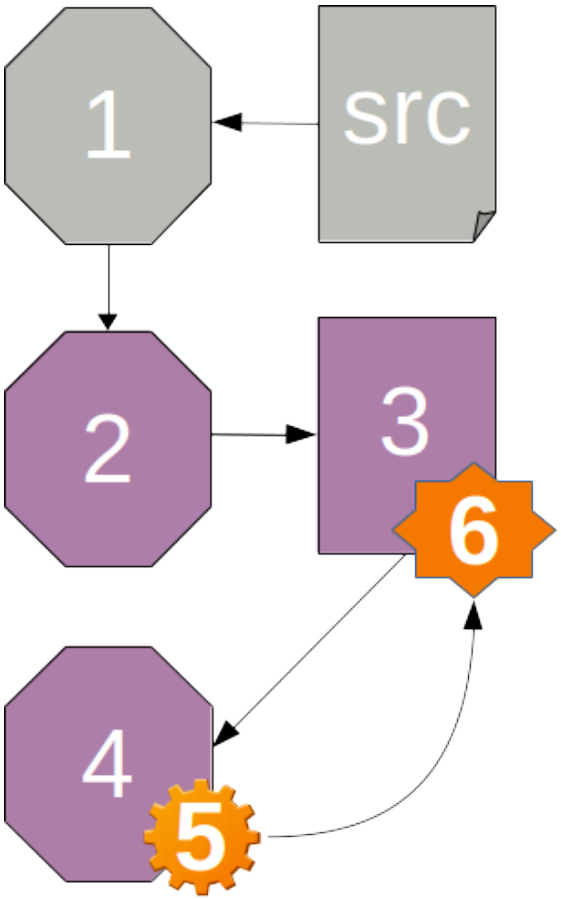
Immutabilité
Lecture facile
Gestion de l'échec

Templating simple



1. Créer une ressource (instance, container)
2. Configuration
3. Sauvegarder comme modèle

Templating type CI/CD



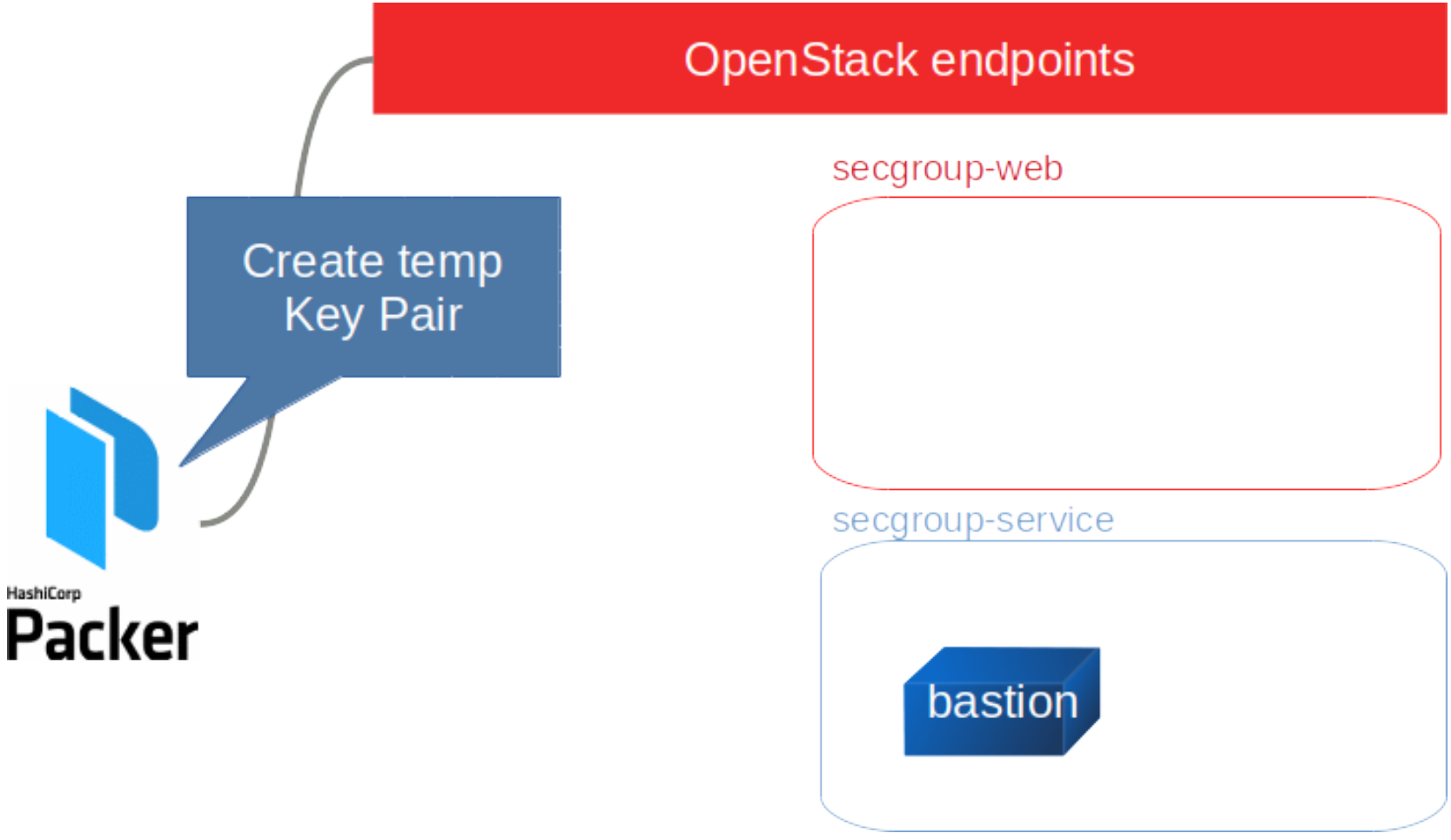
1. Créer une ressource (instance, container)
2. Configuration
3. Sauvegarder comme modèle
4. Instancier le modèle
5. Tester l'instance
6. Tagguer le modèle

A propos de Packer

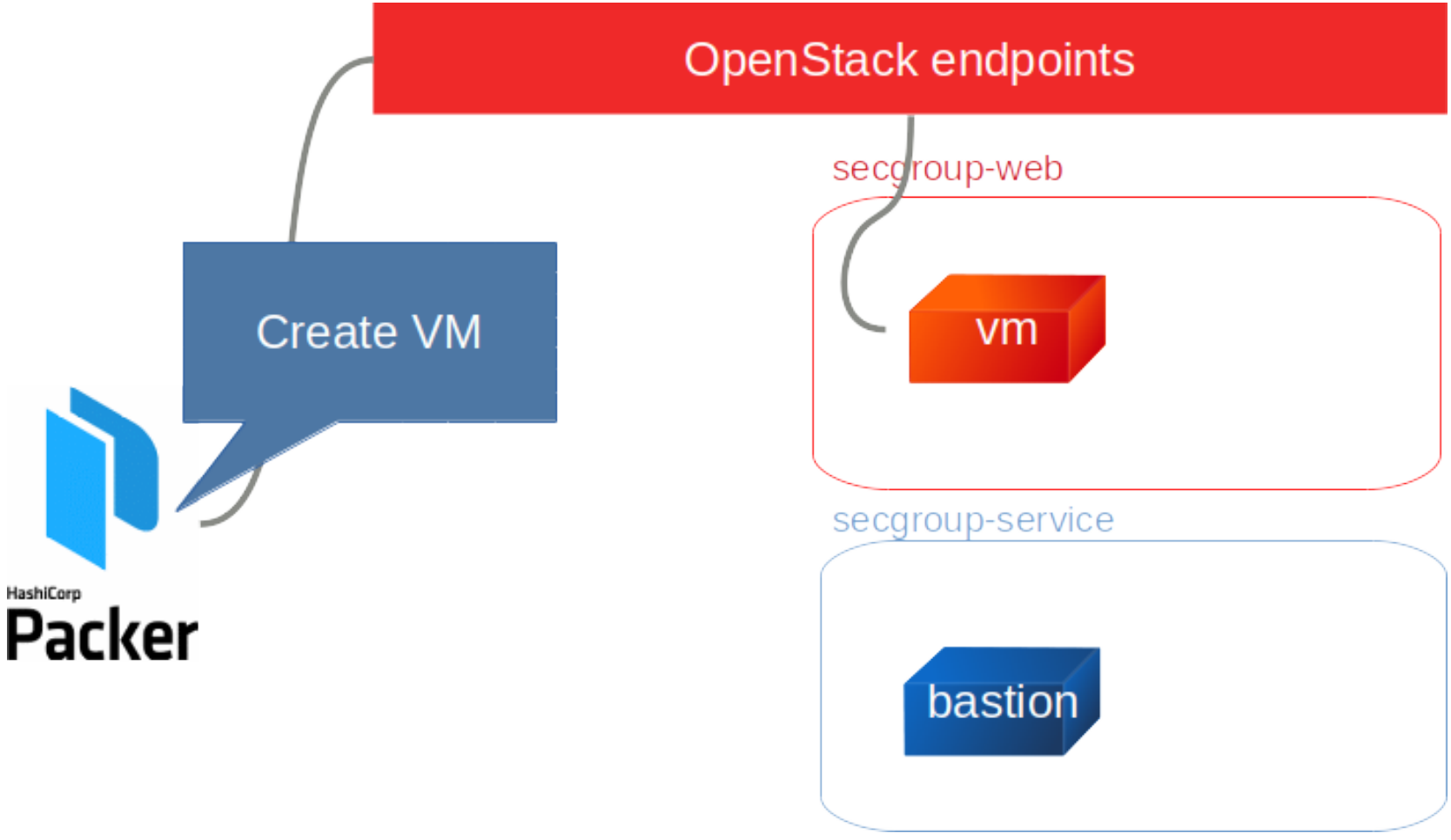
Outils open-source, développé par la société Hashicorp. Permet de générer des images de machines à partir d'une seule configuration, pour plusieurs cibles.

- Builders : Chargé de des opérations
- Builds : Construction d'une image
- Provisionners : Actions de configuration
- Post-processors : Suite du build

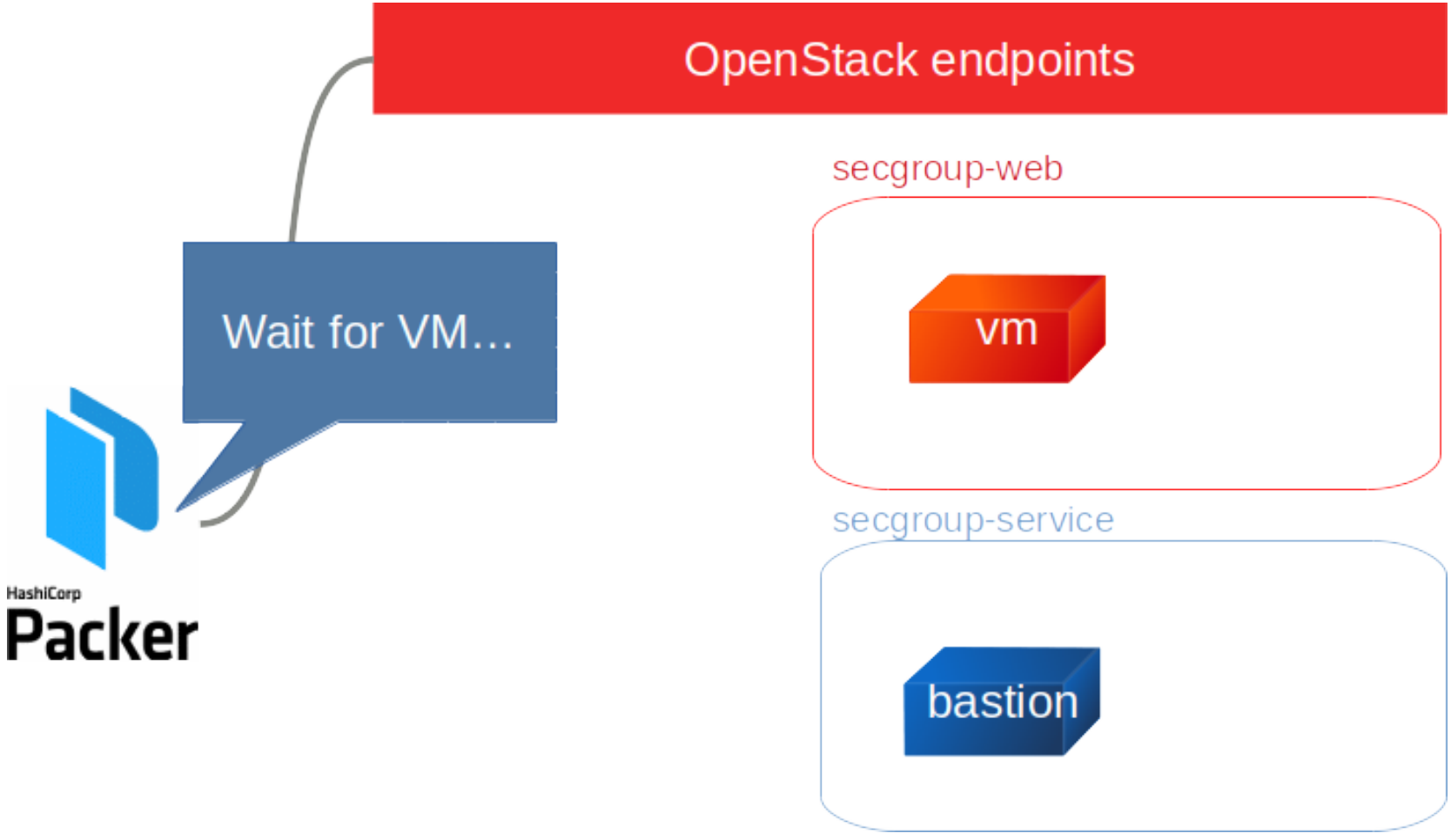
Cinématique Packer : Création Keypair



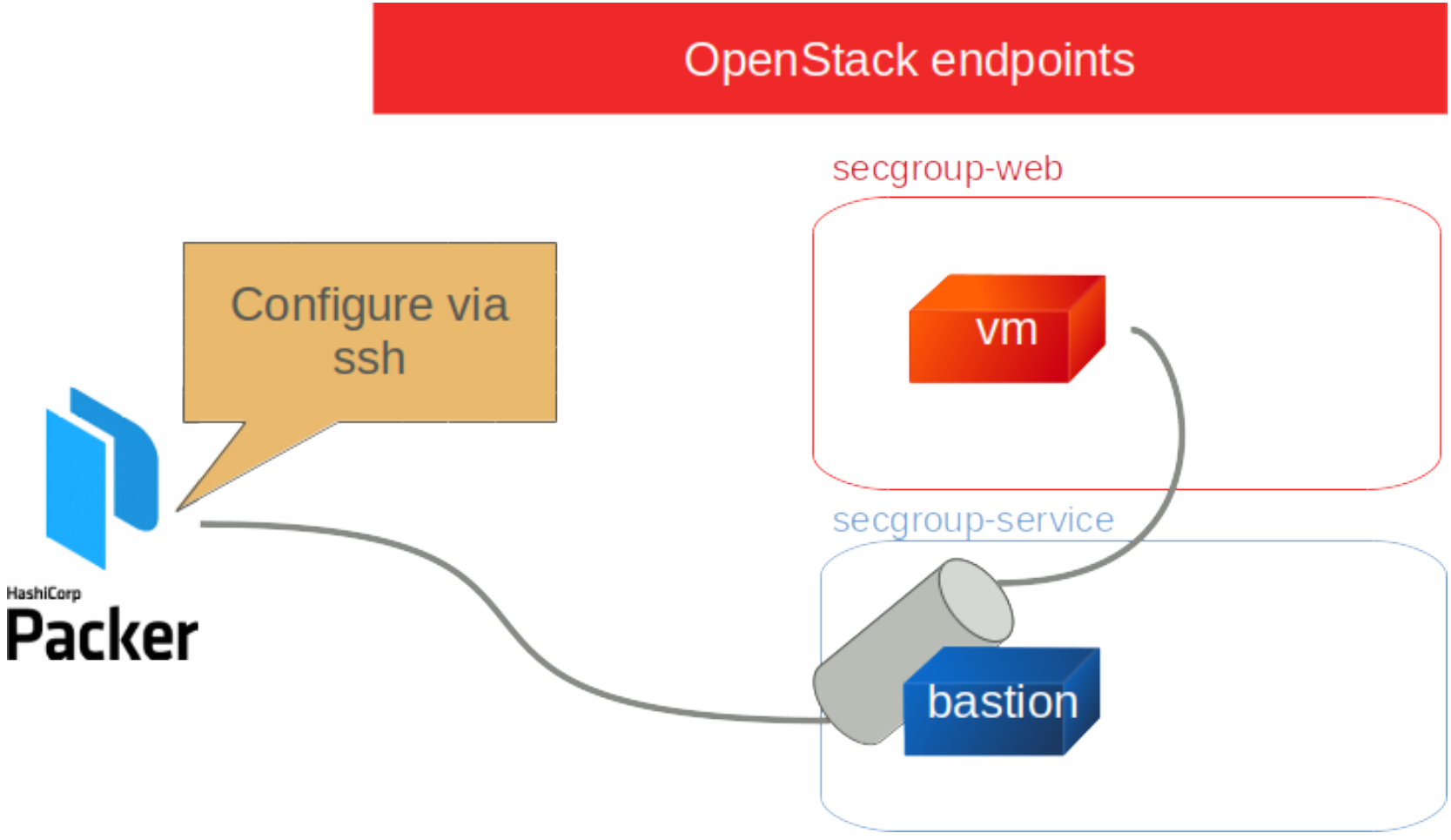
Cinématique Packer : Création instance



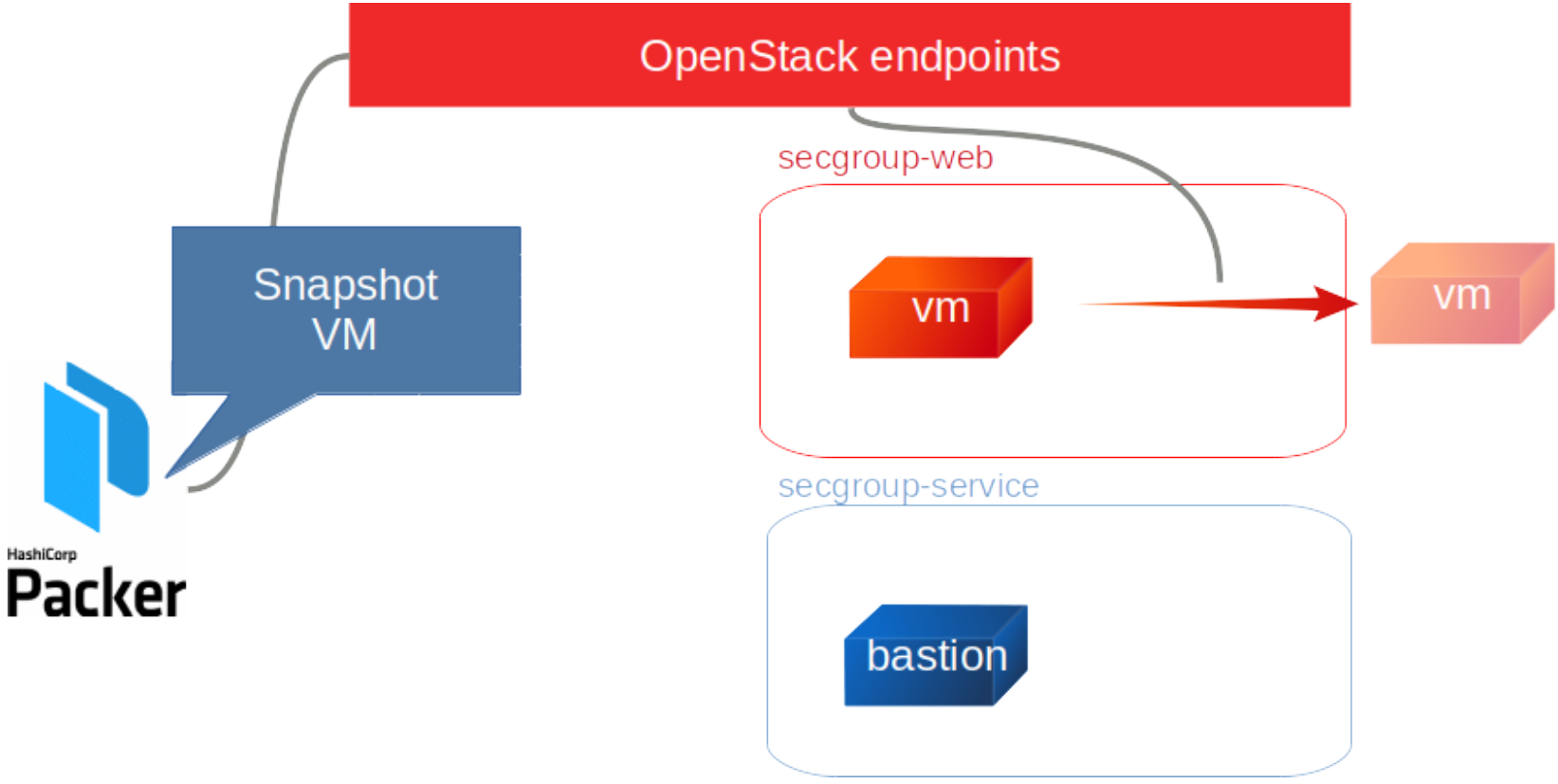
Cinématique Packer : Attendre l'instance



Cinématique Packer : Provisionner



Cinématique Packer : Création snapshot



Packer usage

Pour lancer un build...

```
packer build mon_template.json
```

Les options **validate** et **inspect** sont également souvent utiles.

Template packer : structure du fichier json

- variables : dictionnaire
- builders : liste
- provisionners : liste
- post-processor : liste

```
{
  "variables": {
  },
  "builders": [
   ],
  "provisioners": [
   ],
  "post-processors": [
  ]
}
```

Template packer : les variables

```
"variables" : {
 "vaportag": "1.0",
 "bastionpassword": "{{env `OS_PASSWORD`}}",
 "hello": "{{ consul_key `my_appli/data/hello` }}",
 "secret": "{{ vault `secrets/hello` `foo`}}",
 "imagename": "app-vapor-myname-{{isotime \"2006-01-02_0304\" }}
}
```

Par la suite, dans le template, ces variables seront utilisées sous la forme.

```
"version2deploy": "{{user `vaportag`}}"
```

Template packer : le builder

```
"builders": [
{
  "type": "openstack",
  "ssh_username": "ubuntu",
  "image_name": "vapormap-destimg-user-tletou",
  "source_image_name": "ada-ubuntu.18.04-raw",
  "networks": ["000-0000000-00"],
  "flavor": "m1.small",
  "ssh_bastion_host": "134.158.74.151",
  "ssh_bastion_username": "{{ user `bastionuser` }}",
  "ssh_bastion_password": "{{ user `bastionpassword` }}"
} ]
```

Template packer : authentification du builder

Utilisation des variables d'environnement *OS_** ou déclaration des paramètres du cloud dans le template.

```
"builders": [
{
  "type": "openstack",
  "identity_endpoint": "https://keystone.lal.in2p3.fr:5000/v3",
  "tenant_name": "resinfo-anf-ada",
  "domain_name": "Default",
  "username": "{{ user `username` }}",
  "password": "{{ user `password` }}",
  "region": "{{ user `region` }}",
}
```

Template packer : type de provisionners

- Ansible local : Ansible est lancé sur l'ordinateur executant Packer
- Ansible remote : Ansible est lancé sur l'instance
- File : Déposer un fichier sur l'instance
- PowerShell : scripts pour système Windows
- Shell : ligne de commande ou scripts lancés sur l'instance
- ou encore ; Puppet, Salt, Chef, plus...

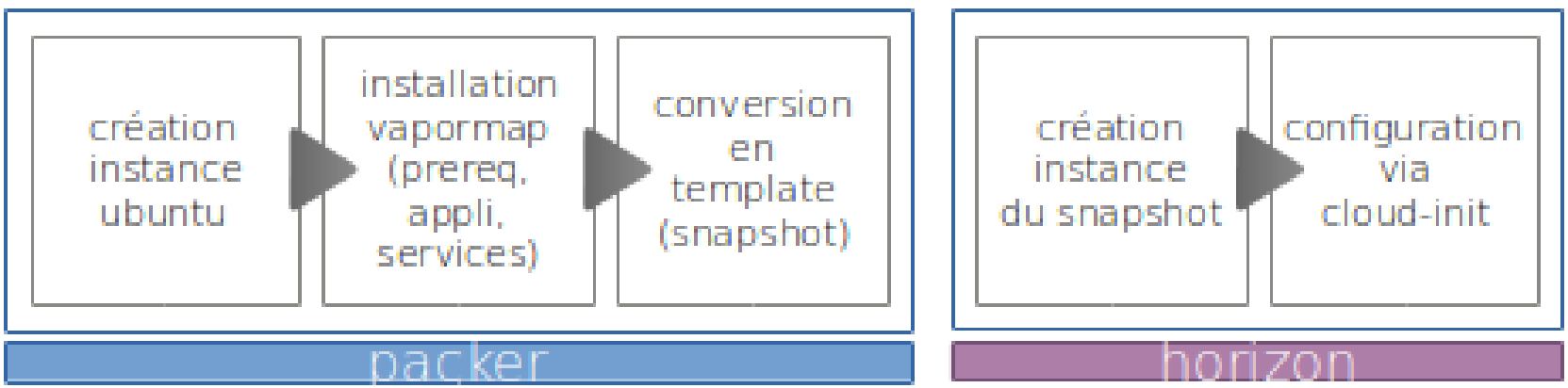
more : https ://www.packer.io/docs/provisioners/index.html

Template packer : déclaration des provisionners

```
"provisioners": [
        {
          "type": "file",
          "source": "app.tar.gz",
          "destination": "/tmp/app.tar.gz"
        },
        {
          "type": "shell",
          "inline": ["tar xfz /tmp/app.tar.gz"]
        }
]
```

Déroulement du TP

1. Préparation d'une image
   - Installation de VaporMap et nginx
   - Installation du service sans configuration MySQL (/etc/systemd/system/gunicorn.conf)
2. Instanciation via Horizon
   - configuration connexion MySQL par cloud-init



Les bons outils : packer et openstackclient

Pour réaliser le TP, vous aurez besoin de **Packer** et du client **openstack**.

1. Installation manuelle sur votre machine
2. **Utilisation d'une toolbox Docker**

Les bons outils : toolbox devops ada

La toolbox ada-devops-tbx est composée d'une image Docker stockée dans un registry et un script qui perme de lancer celle-ci.
Le script :

1. pull l'image docker
2. configure des variables d'environnement (ex : OS_*)
3. demande OS_USERNAME et OS_PASSWORD (si non renseignées)
4. lance le container en interactif
5. mappe le répertoire courant dans le container