Packer
by Hashicorp

Agenda

- Packer – aperçu
- Avantages
- Cas d'utilisation
- Plateformes prises en charge
- Workflow - Création, provisionnement et post-traitement
- Lab

Packer

- Packer est un outil permettant de créer des images de machines et de conteneurs pour plusieurs plates-formes à partir d'une configuration source unique.
- Une image machine est une unité statique unique qui contient un système d'exploitation préconfiguré et des logiciels installés, utilisée pour créer rapidement de nouvelles machines en cours d'exécution.
- Packer crée uniquement des images. Il ne tente en aucun cas de les gérer.
- Une fois construites, c'est à vous de les lancer ou de les détruire.

Avantages d'utilisation

Déploiement d'infrastructure rapide : les images Packer vous permettent de lancer des machines entièrement provisionnées et configurées en quelques secondes. Cela profite non seulement à la production, mais également au développement.

Portabilité multi-fournisseurs : Packer crée des images identiques pour plusieurs plates-formes, vous pouvez exécuter la production dans AWS, la préparation/le contrôle qualité dans un cloud privé comme OpenStack et le développement dans des solutions de virtualisation de bureau telles que VMware ou VirtualBox.
Chaque environnement exécute une image machine identique, offrant une portabilité ultime.

Stabilité améliorée : Packer installe et configure tous les logiciels d'une machine au moment de la création de l'image. S'il y a des bugs dans ces scripts, ils seront détectés tôt, plutôt plus tard au lancement d'une machine.

Meilleure testabilité : Une fois qu'une image de machine est créée, cette image de machine peut être rapidement lancée et testée pour vérifier le bon fonctionnement. Si tel est le cas, vous pouvez être sûr que toutes les autres machines lancées à partir de cette image fonctionneront correctement.

Use cases

Livraison continue – Packer dans le pipeline de livraison continue peut être utilisé pour générer de nouvelles images de machine pour plusieurs plates-formes à chaque modification apportée à Chef/Puppet/Ansible. Dans ce pipeline, les images créées peuvent ensuite être lancées et testées, vérifiant ainsi le fonctionnement des modifications de l'infrastructure. Si les tests réussissent, vous pouvez être sûr que l'image fonctionnera une fois déployée. Cela apporte de la stabilité et testabilité aux changements d'infrastructure.

Parité Dev/Prod – Packer permet de maintenir le développement, la préparation et la production aussi similaires que possible. Packer peut être utilisé pour générer des images pour plusieurs plates-formes en même temps. Ainsi, si vous utilisez AWS pour la production et Docker pour le développement, vous pouvez générer à la fois une AMI et une machine VMware à l'aide de Packer en même temps à partir du même modèle. Mélangez.

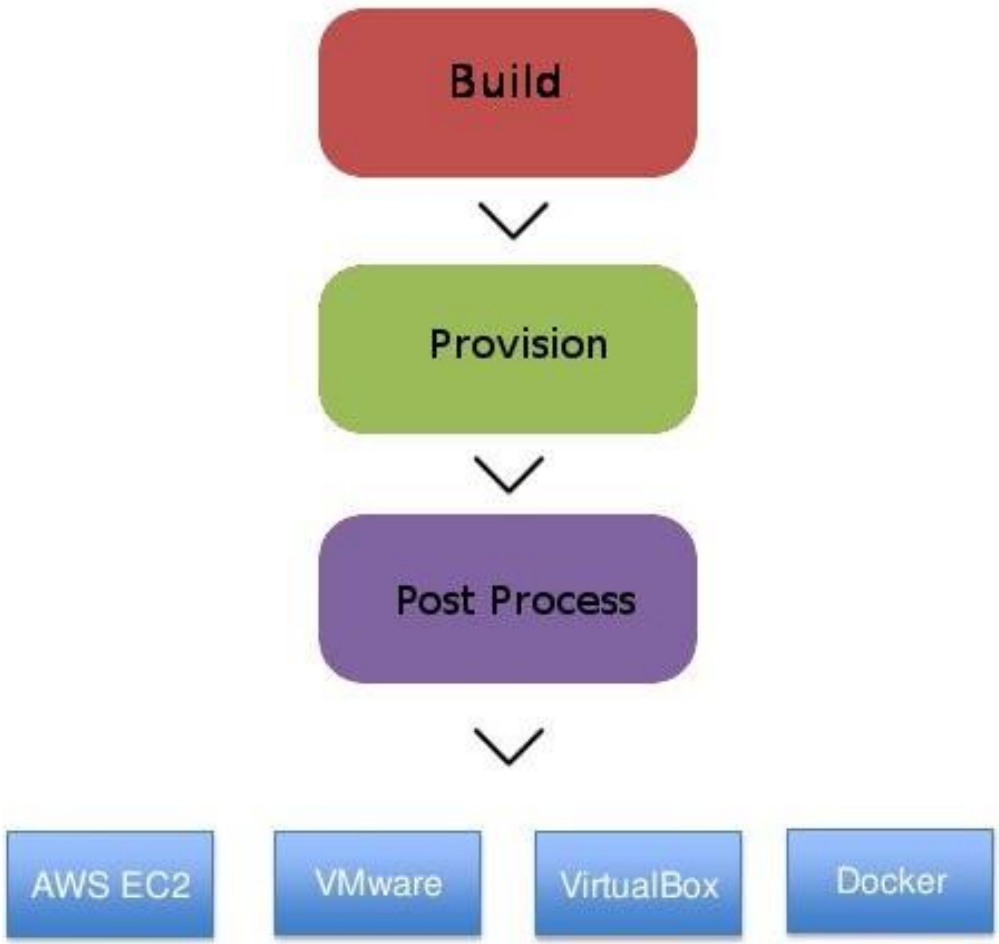
Création d'appareils/démonstrations - Étant donné que Packer crée des images cohérentes pour plusieurs plates-formes en parallèle, il est parfait pour créer des démonstrations d'appareils et de produits jetables. Au fur et à mesure que votre logiciel évolue, vous pouvez créer automatiquement des appliances avec le logiciel préinstallé.

Plateformes Supportées

- Amazon EC2 (AMI)
- CloudStack
- OpenStack
- DigitalOcean
- Docker
- Google Compute Engine
- Parallels
- QEMU
- VirtualBox (OVF)
- VMware (VMX)

Vous pouvez ajouter le support à n'importe quelle plate-forme en étendant Packer par des plugins.

Workflow


Build
Provision
Post Process
AWS EC2
VMware
VirtualBox
Docker

Workflow - terminologie

- **Templates:** Fichiers JSON contenant les informations de build
- **Builders:** Configuration de build spécifique à la plate-forme
- **Provisioners:** Outils qui installent les logiciels après l'installation initiale du système d'exploitation
- **Post-processors:** Actions à effectuer après que l'image a été construite

Workflow - Build

```
{
 "variables": {
  "aws_access_key": "{{env `AWS_ACCESS_KEY`}}",
  "aws_secret_key": "{{env `AWS_SECRET_KEY`}}"
 },
 "builders": [{
  "type": "amazon-ebs",
  "access_key": "{{user `aws_access_key`}}",
  "secret_key": "{{user `aws_secret_key`}}",
  "region": "us-east-1",
  "source_ami": "ami-fce3c696",
  "instance_type": "t2.micro",
  "ssh_username": "admin",
  "ami_name": "yourApp  {{timestamp}}"
 }]
}
```

Shell> packer validate packer.json

Workflow - Build & Variables

- Via Command line using **-var** flag

  >packer build
  -var 'aws_access_key=Secret
  -var 'aws_secret_key=Secret2
  packer.json

- Via File with **-var-file** flag

```
{
 "aws_access_key": "accessKey",
 "aws_secret_key": "secretKey"
}
```

  >packer build -var-file=variables.json packer.json

  L'option var-file peut être spécifiée plusieurs fois et les variables de plusieurs fichiers seront lues et appliquées. La combinaison des options -var et -var-file fonctionne également comme prévu.

Workflow - Provision

```
{
 "variables": ["..."],
 "builders": ["..."],
 "provisioners": [{
 "type": "shell",
 "inline": [
   "sleep 30",      ##Attendre SSH
   "sudo apt-get update",
   "sudo apt-get install -y redis-server"
  ]
 },{
  "type": "shell",
  "script":  "./scripts/install-java.sh",
}]
}
```

Autres - Remote shell, File uploads, Ansible (local & remote), Chef, Puppet, Salt, PowerShell etc.

Workflow - Post Process

```
{
 "variables": "...",
 "builders": ["..."],
 "provisioners": ["..."],
 "post-processors": [
  {
   "type": "compress",
   "format": "tar.gz"
  }
 ]
}
```

Autres - Amazon Import, CheckSum, Docker Push/Tag/Save, Google Compute Export, Vagrant, vSphere etc.

Lab

Netographie

Références :

1. https://www.packer.io
2. https://www.packer.io/docs/extend/builder.html
3. https://www.packer.io/docs/builders/virtualbox-iso.html
4. https://www.packer.io/docs/builders/amazon.html
5. https://www.packer.io/docs/builders/docker.html
6. https://www.packer.io/intro/getting-started/build-image.html
7. https://www.packer.io/intro/getting-started/provision.html
8. https://www.packer.io/intro/getting-started/parallel-builds.html
9. https://www.packer.io/docs/templates/user-variables.html
10. https://www.packer.io/docs/templates/post-processors.html
11. https://github.com/chef/bento
12. http://blog.deimos.fr/2015/01/16/packer-build-multiple-images-easily/
13. http://kappataumu.com/articles/creating-an-Ubuntu-VM-with-packer.html