HashiCorp
Packer

Using HCP Packer with Terraform Cloud

Agenda

01

Packer OSS & HCP Packer Overview

Packer

The standard for infrastructure automation to **build, govern** and **manage** any image for any cloud

Images as Code

Customization

Ansible
Bash
Powershell
...

Cloud

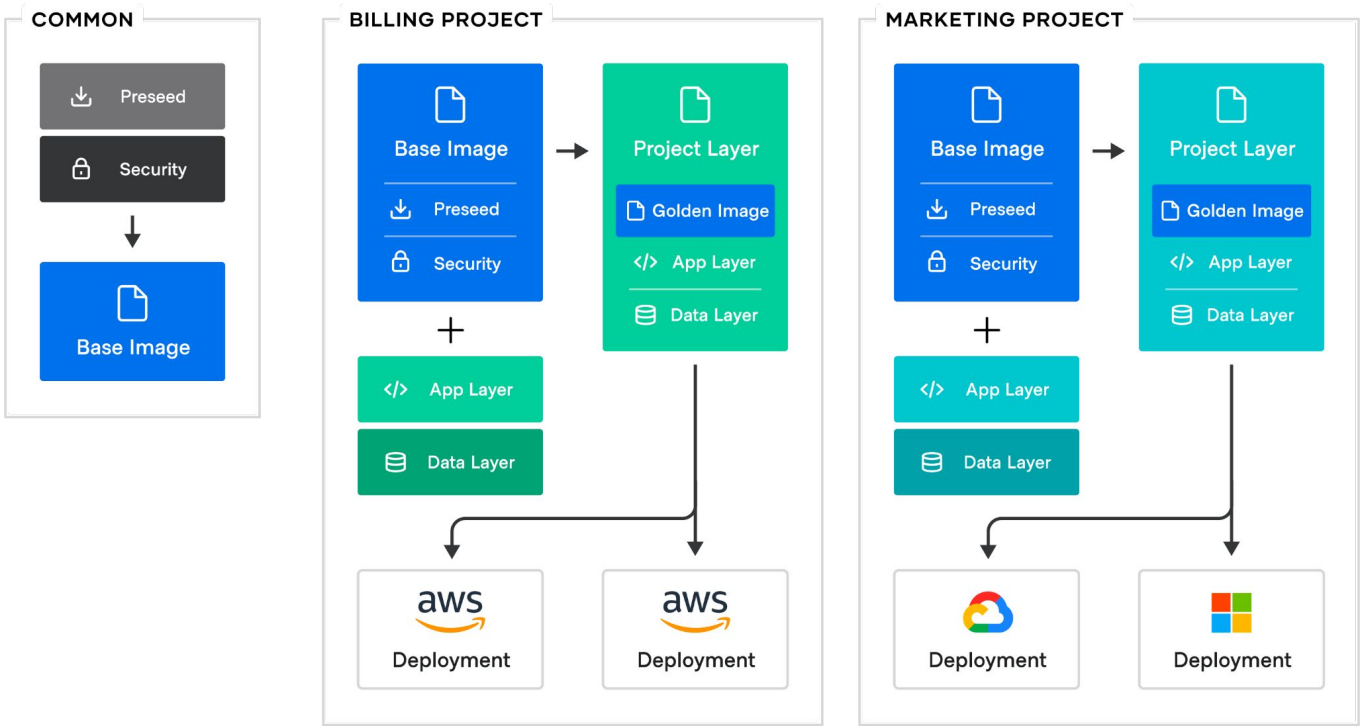
AWS
Azure
GCP
...

Private DC

VMware
VirtualBox
Vagrant
...

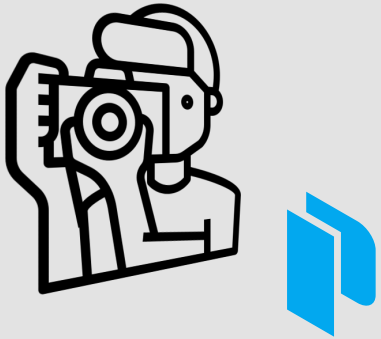
What is HCP Packer?

Codification = Standardization and Automation



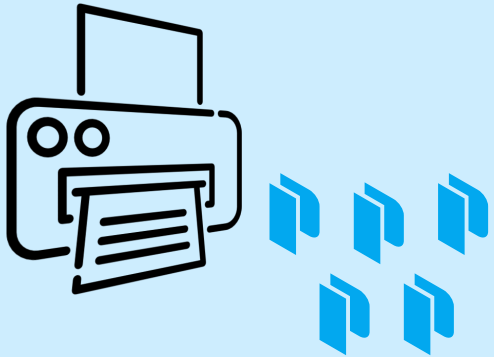
Packer

- Create/build your image with all the specific requirements your organization needs to meet
- Maintains the image by making updates when necessary

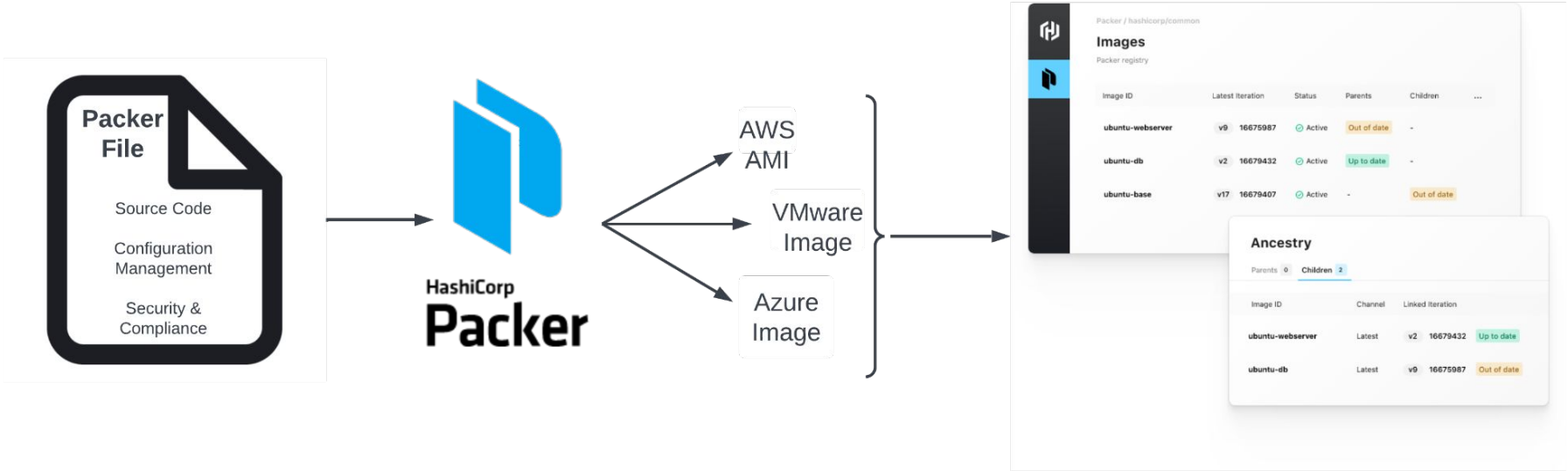


HCP Packer

- Store the metadata of your image after being built with Packer
- Quickly deploy identical machine images for multiple platforms from a single source template



HCP Packer Workflow



Organizations Must Still Define:

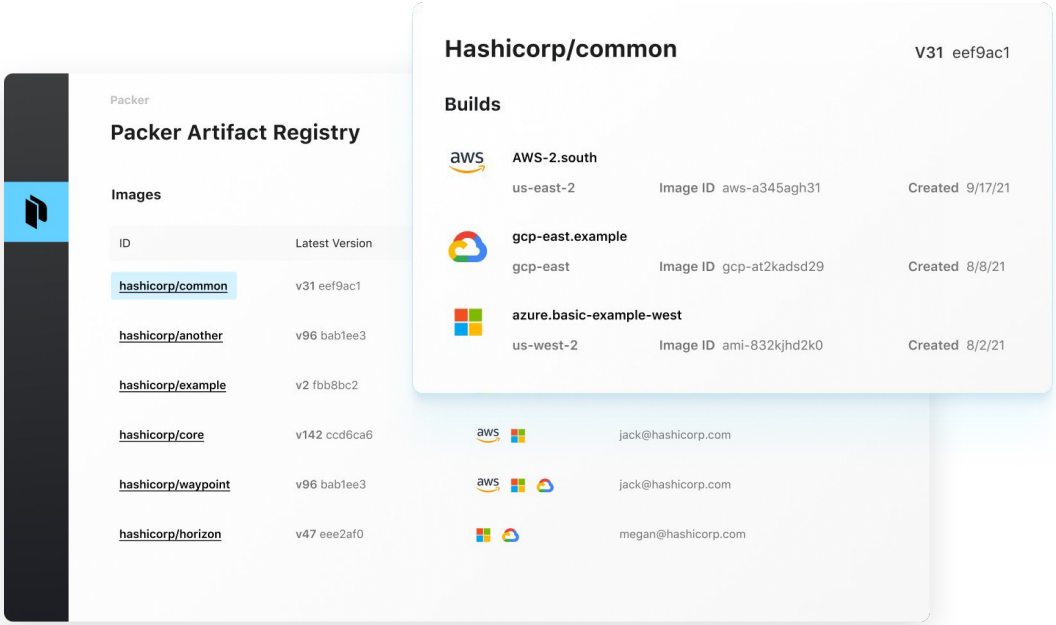
Which are the “golden” images, and what version should they use?

What is the ideal workflow to ensure images are hardened, secure, and compliant and can be replicated across clouds and private data centers?

Can this process be integrated into our current provisioning workflows?

Golden Image Pipeline

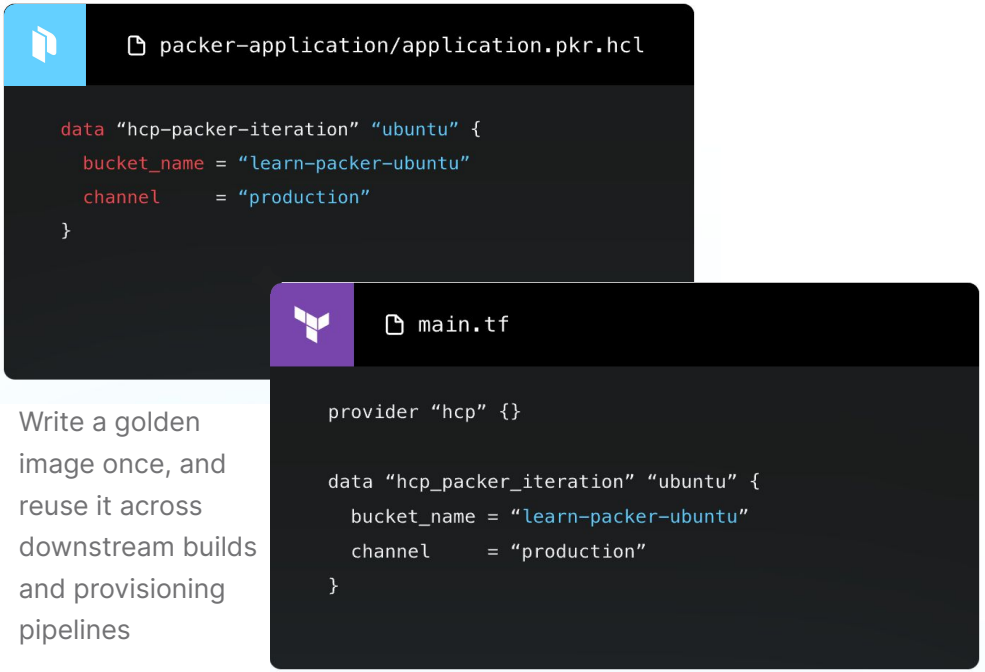
- HCP Packer is a multi-cloud image registry
- It uses metadata from your existing images to track iterations
- Builds artifacts across downstream builds and provisioning pipelines
- **It forms the foundation of a multi-cloud golden image pipeline**



Standardizing the Workflow

Bridging the Gap Between Image Factories and Image Deployments

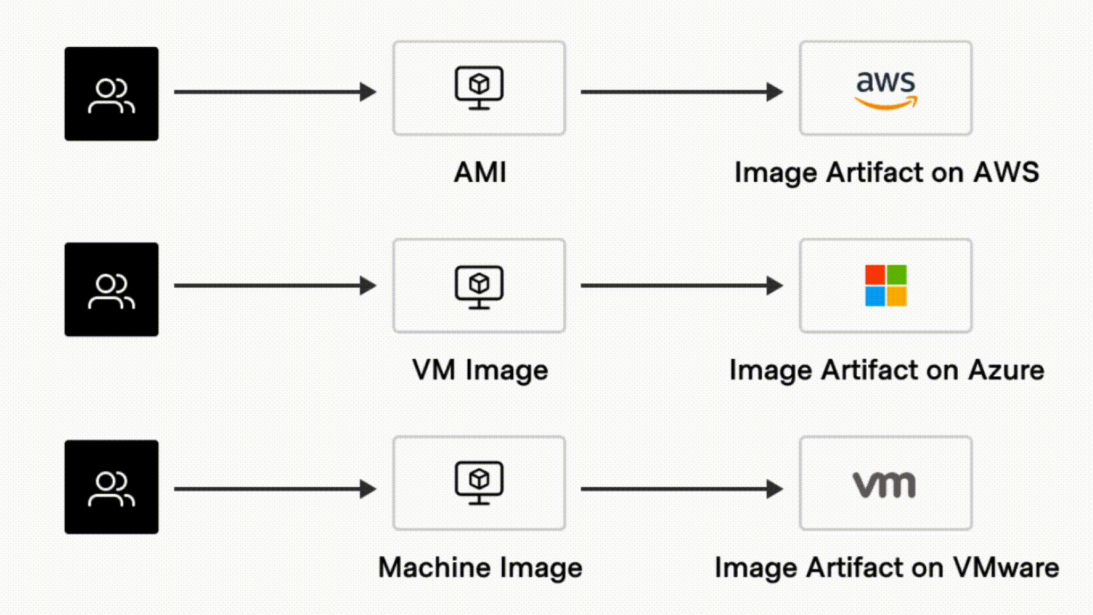
- **Gain visibility** into a base image's downstream builds and associated provisioning pipelines
- **Automate updates** by triggering a build or an apply
- **Create a compliant workflow** for images across all of your infrastructure



Write a golden image once, and reuse it across downstream builds and provisioning pipelines

The Shift to HCP Packer

- **Easy to read and write:** Leverages HCL2, making it easier to use alongside Terraform
- **Extensible:** Use one language to manage images across clouds and integrate with other configuration tools
- **Open source:** Use existing community templates, or write your own



Use Cases

Automated Machine Images

Golden Image Pipeline

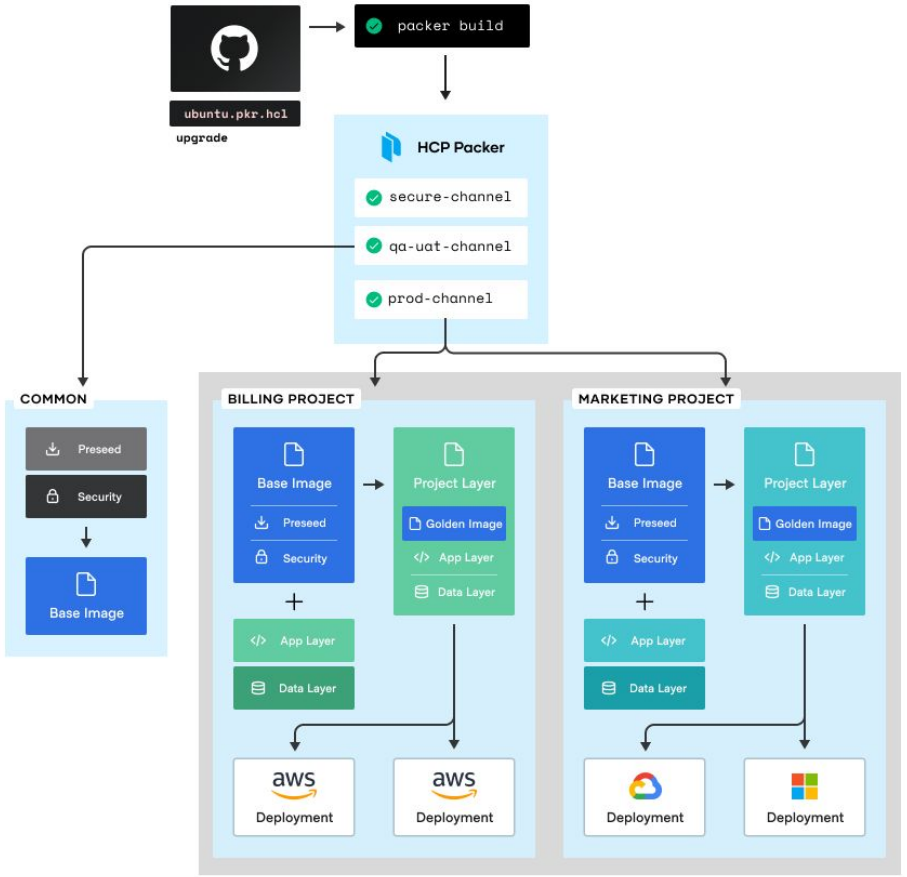
Image Governance & Compliance

Unified Workflow with Terraform

Feature:

Release Management

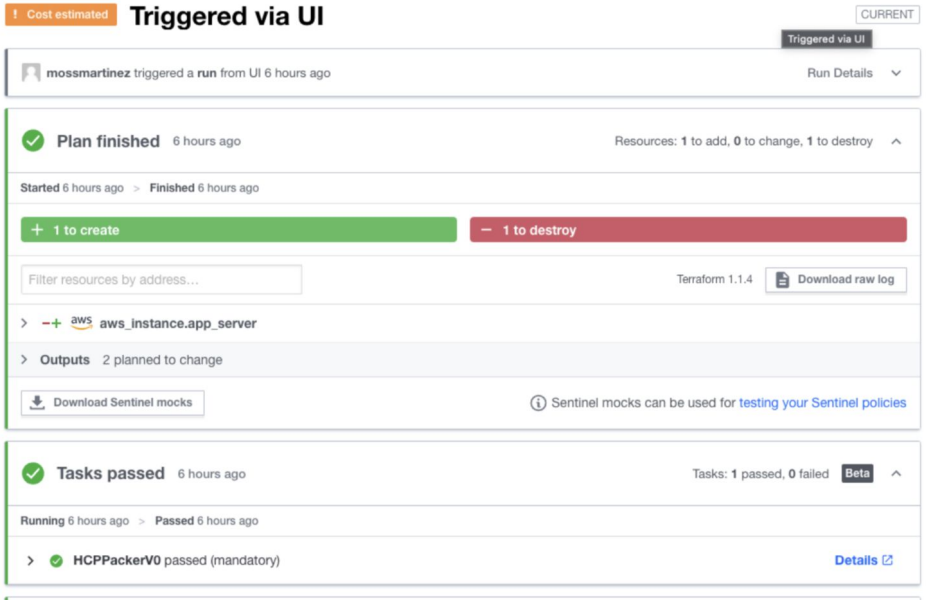
- **Image channels** assign iterations to human-readable names that consumers can reference in Packer templates or Terraform configurations
- **Standardize versions** by declaring a preferred iteration of an image
- **Empower other teams** to retrieve the latest image version from their desired channel without changing their configurations



Feature:

Image Compliance Checks

- **Ensure images are compliant** by integrating HCP Packer with Terraform Cloud using run tasks
- **Reduce risks** by scanning all Terraform configurations for revoked images



Advantages of Packer

1. Rapid infrastructure deployment
2. Multi-provider portability
3. Improved stability
4. Greater testability

Advantages of HCP Packer

1. Visibility
2. Governance & Controls
3. Unified Workflows

02

Creating a Golden Image

Packer Templates

- The `packer {}` block contains Packer settings, including specifying a required Packer version
- The `source` block configures a specific builder plugin, which is then invoked by a `build` block
- The `build` block defines what Packer should do with the image after it launches

```
packer {
  required_plugins {
    ...
  }
}

source "<BUILDER_TYPE>" "<UNIQUE_NAME>" {
  ...
}

build {
  ...
}
```

Packer Templates

- Build an Image

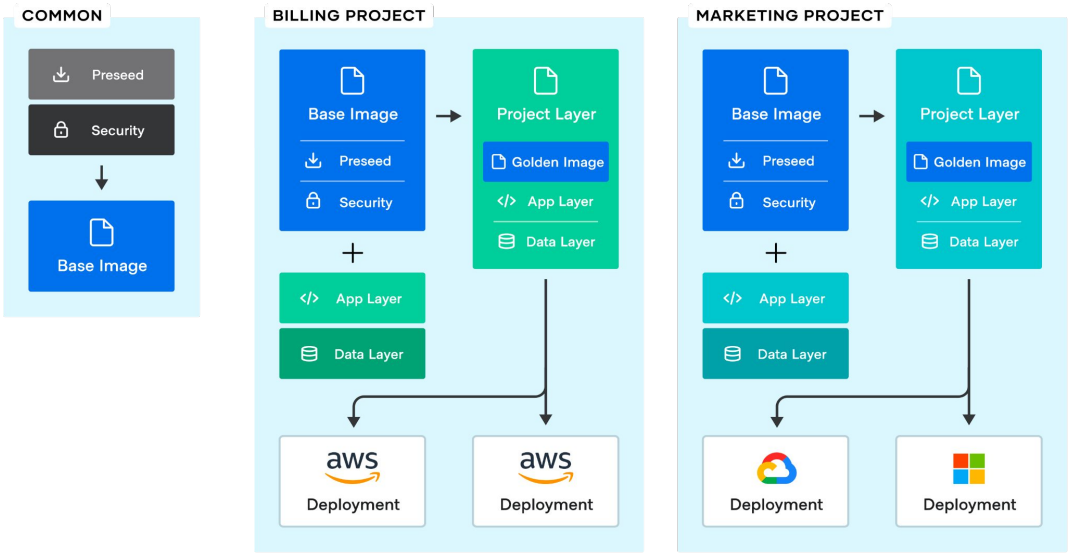
```
packer {
  required_plugins {
    amazon = {
      version = ">= 0.0.2"
      source  = "github.com/hashicorp/amazon"
    }
  }
}

source "amazon-ebs" "ubuntu" {
  ami_name      = "learn-packer-linux-aws"
  instance_type = "t2.micro"
  region        = "us-west-2"
  source_ami_filter {
    filters = {
      name                = "ubuntu/images/*ubuntu-xenial-16.04-amd64-server-*"
      root-device-type    = "ebs"
      virtualization-type = "hvm"
    }
    most_recent = true
    owners      = ["099720109477"]
  }
  ssh_username = "ubuntu"
}

build {
  name    = "learn-packer"
  sources = [
    "source.amazon-ebs.ubuntu"
  ]
}
```

Build a Golden Image

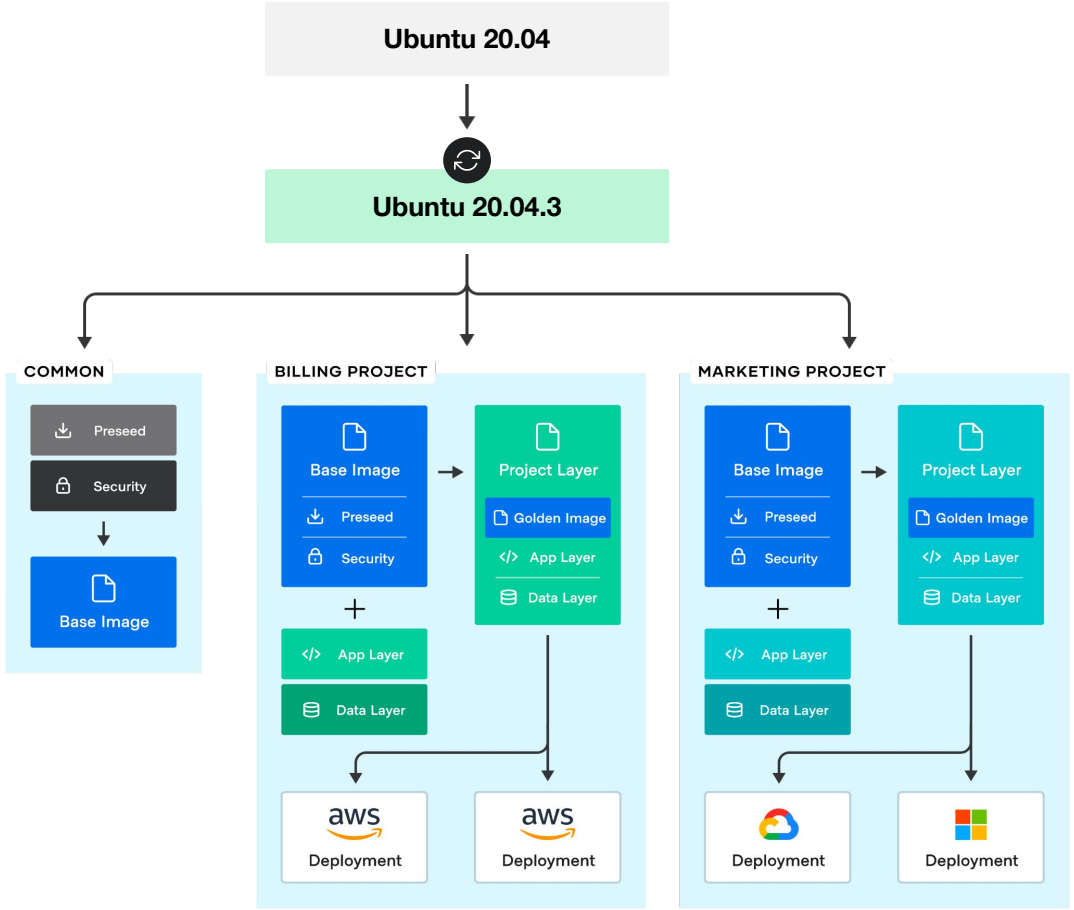
```
$ packer build aws-ubuntu.pkr.hcl

learn-packer.amazon-ebs.ubuntu: output will be in this color.

==> learn-packer.amazon-ebs.ubuntu: Prevalidating any provided VPC information

==> learn-packer.amazon-ebs.ubuntu: Prevalidating AMI Name: learn-packer-linux-aws

   learn-packer.amazon-ebs.ubuntu: Found Image ID: ami-0dd273d94ed0540c0

==> learn-packer.amazon-ebs.ubuntu: Creating temporary keypair:
packer_608a6435-e0b2-c633-95f0-bf168f01e891

==> learn-packer.amazon-ebs.ubuntu: Creating temporary security group for this
instance: packer_608a6437-6b48-288e-6d5e-c085366a5541

==> learn-packer.amazon-ebs.ubuntu: Authorizing access to port 22 from [0.0.0.0/0]
in the temporary security groups...

...

==> Wait completed after 5 minutes 19 seconds

==> Builds finished. The artifacts of the successful builds are:
--> learn-packer.amazon-ebs.ubuntu: AMIs were created:
us-east-2: ami-0d9356dfdbea0d50b


--> learn-packer.amazon-ebs.ubuntu: Published metadata to HCP Packer registry
packer/learn-packer/iterations/01FFJD3GDKRBW4CZ314EZ467VY
```

Golden Image Pipeline

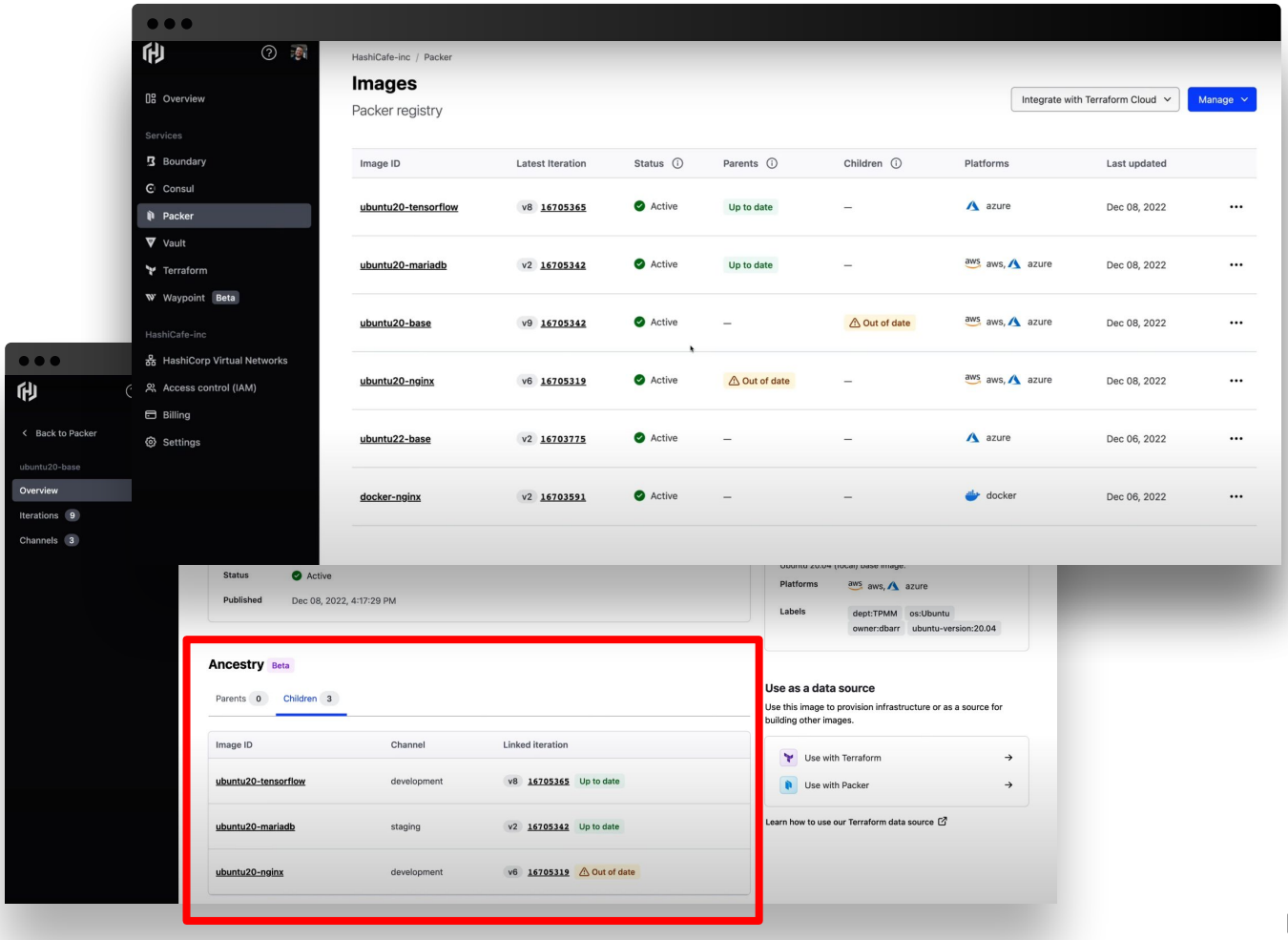
Golden Image: an image on top of which developers can build applications, letting them focus on the application itself instead of system dependencies and patches



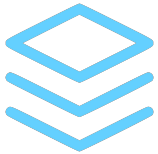
Updating your Golden Image



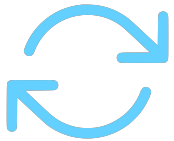
Golden Image Ancestry



Manage the Lifecycle of a Packer Image

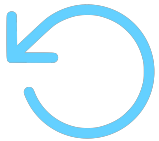


Create multiple versions of the same image



Release new image versions

Scheduling a revocation of an image



Use channel rollback

Best Practices

Variables

- Provide the type in your variable definition
- If no type is set, Packer assumes it will be of type string
- Set a default value so you don't have to at run time
- Do NOT save sensitive information to a VCS

```
variable "weekday" {}

variable "sudo_password" {
      type      = string
      default   = "mypassword"
      sensitive = true
}

variable "flavor" {
      type    = string
      default = "strawberry"
}

variable "exit_codes" {
      type    = list(number)
      default = [0]
}

locals {
      ice_cream_flavor = "${var.flavor}-ice-cream"
      foo              = "bar"
}
```

Best Practices

Bucket Labels & Build Labels

- Bucket labels help identify characteristics common to a set of images
- HCP Packer applies custom bucket labels to an entire image bucket
- Build labels are a immutable set of labels based on the Packer template at the time of build

```
"bucket-labels": {
     "team" = "amazon-development",
     "os"   = "Ubuntu"
}

build_labels = {
      "build-time"   = timestamp(),
      "build-source" = basename(path.cwd)
    }
```

Best Practices

Parallel Builds

1. Create a source
2. Add the source to the sources array in the build block

```
source "docker" "ubuntu" {
     image  = "ubuntu:xenial"
     commit = true
}

source "docker" "ubuntu-bionic" {
     image  = "ubuntu:bionic"
     commit = true
}

build {
     name    = "learn-packer"
     sources = [
          "sources.docker.ubuntu",
          "sources.docker.ubuntu-bionic"
     ]
}
```

03

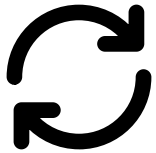
Unified Workflow with Terraform Cloud

Terraform Cloud + HCP Packer

Better together

HCP Packer

- **Build** images
- **Publish** metadata
- **Release** via channels
- **Revoke** and rollback
- **View** ancestry & history

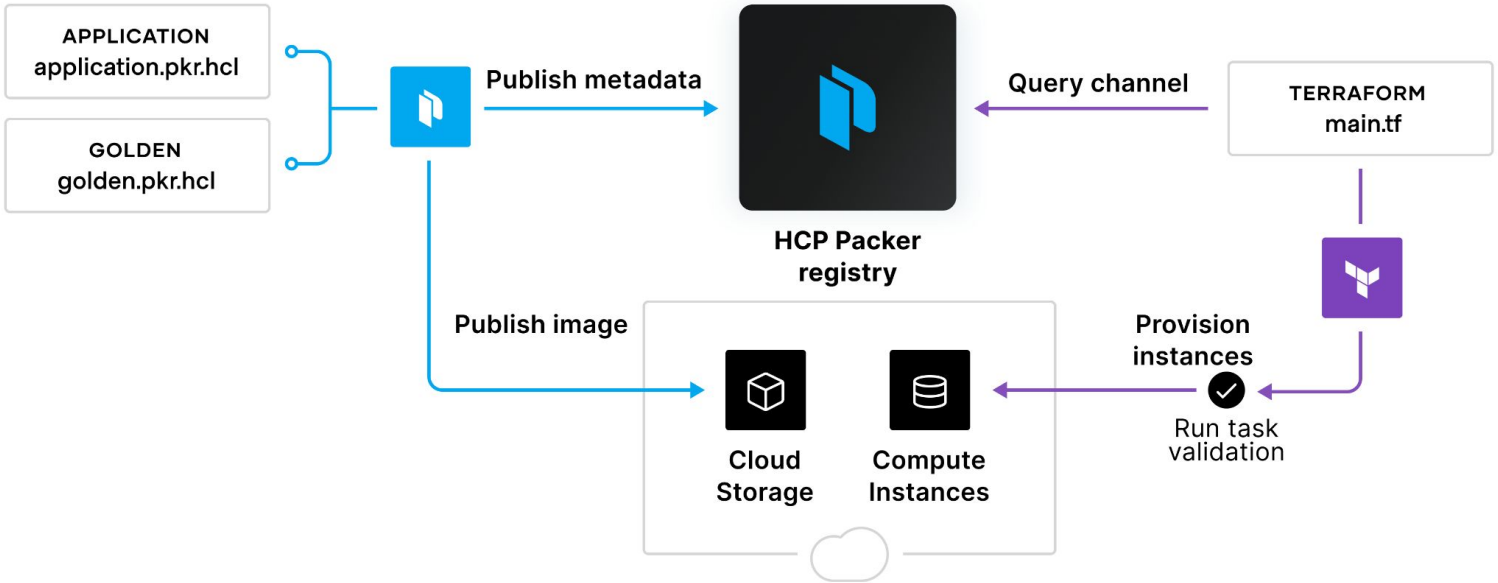


Terraform Cloud

- **Fetch** images
  Data source
- **Provision** infrastructure
- **Validate** image status
  Run tasks, continuous validation

Terraform Cloud + HCP Packer

Integrated build & provisioning workflow



Automate Image Updates

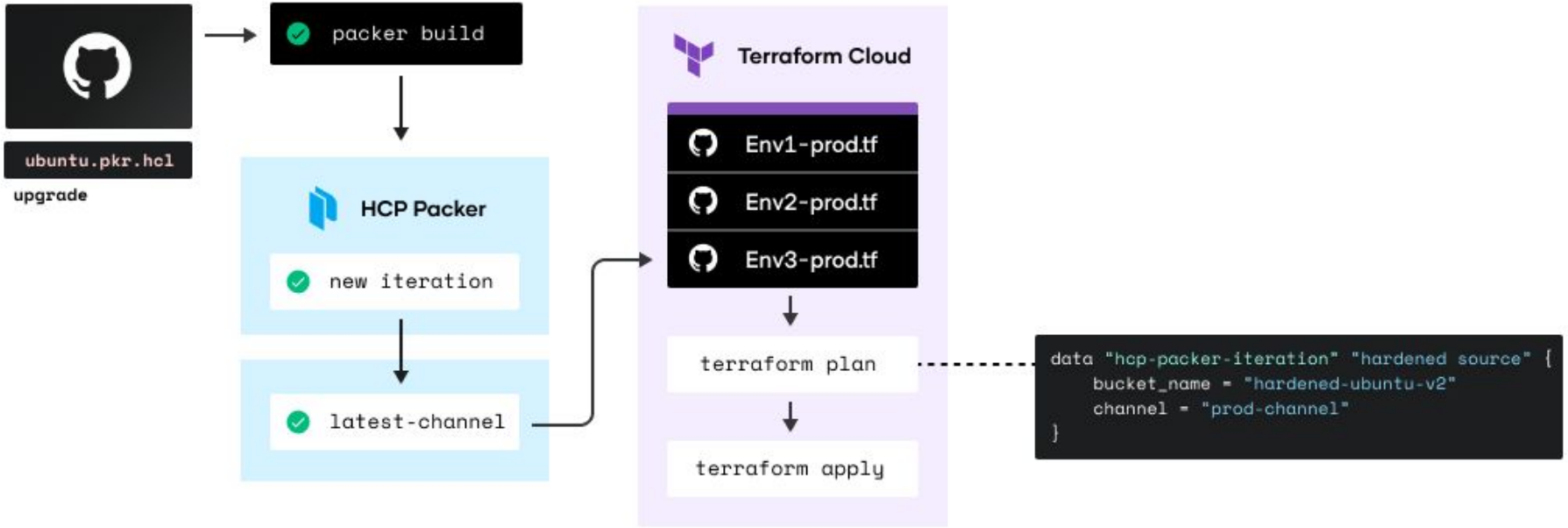


Image Lifecycle Management

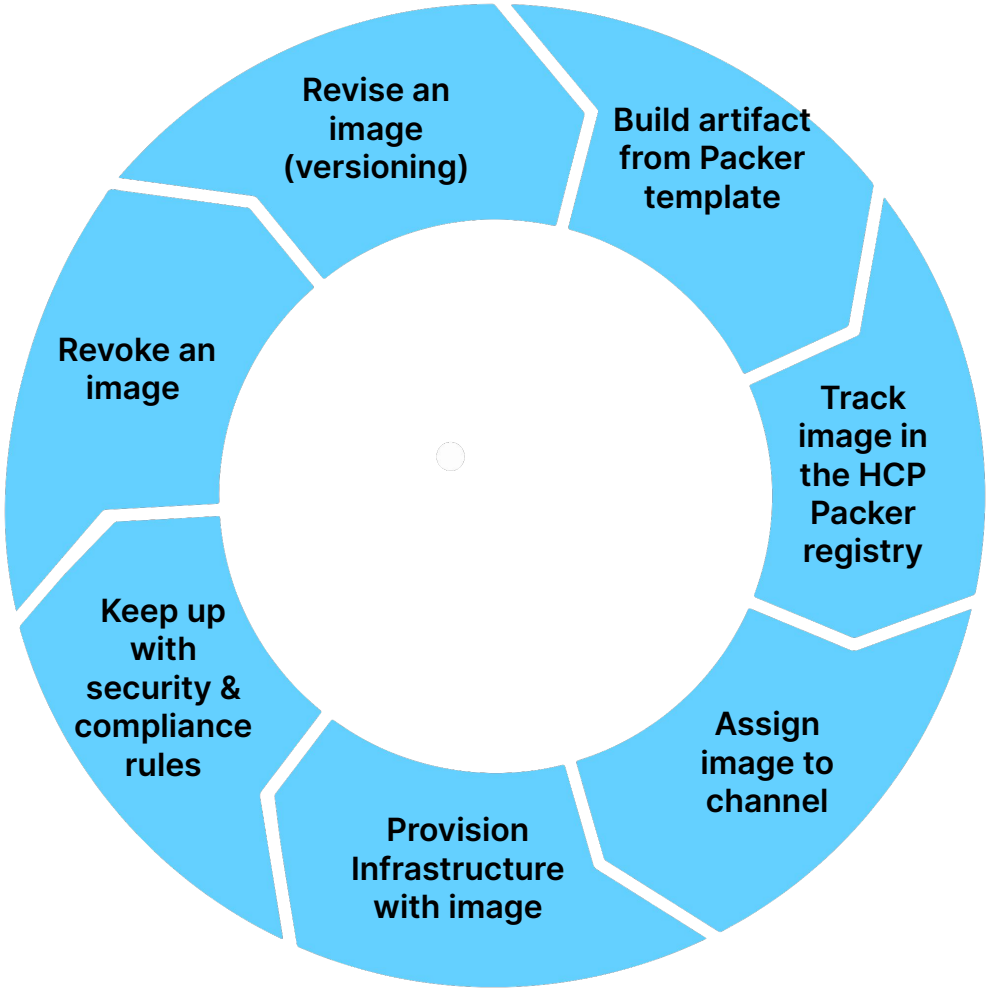
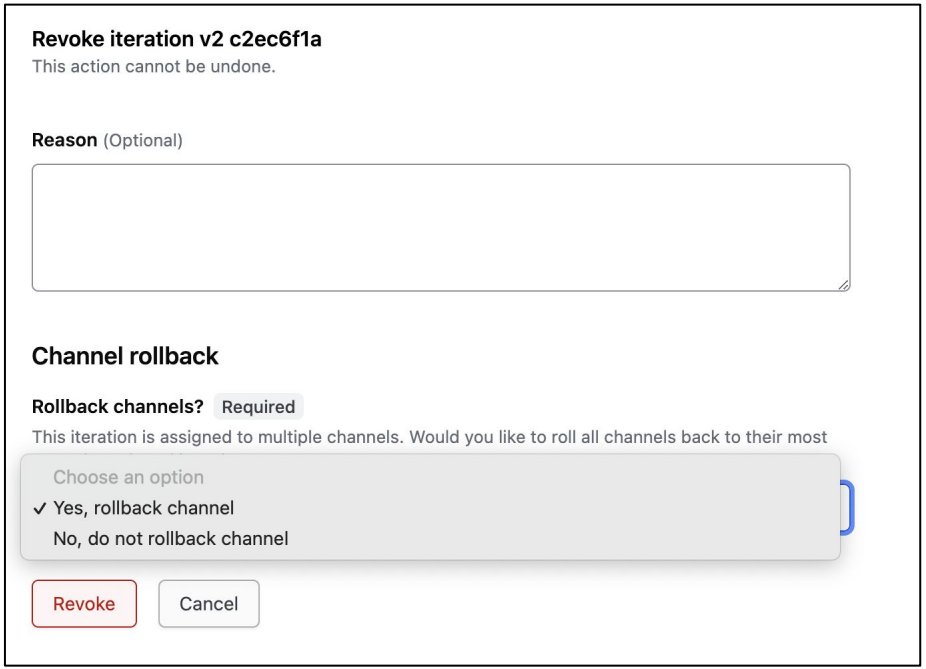
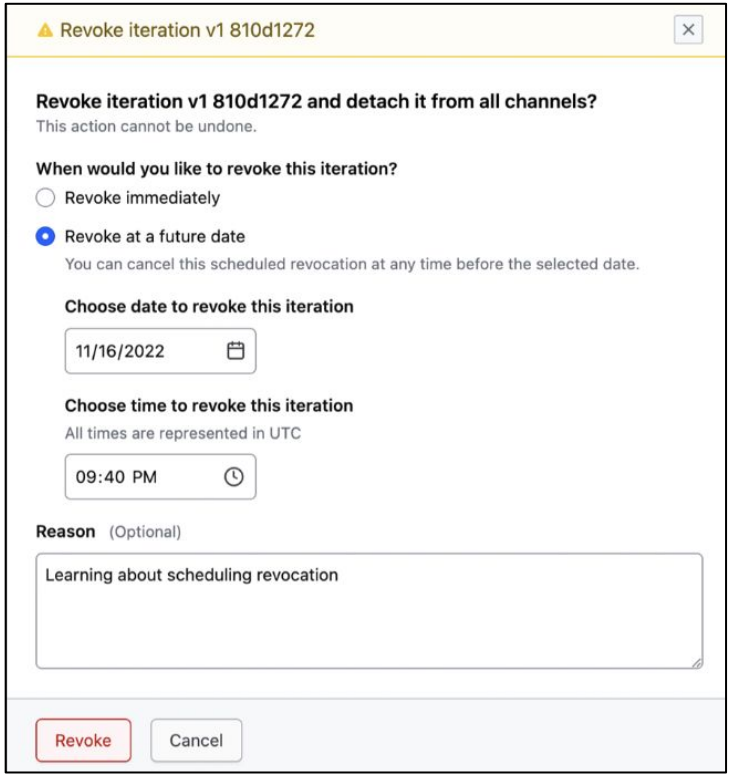


Image Revocation





Resources

- Intro to HCP Packer
- Mutable vs. Immutable Infrastructure
- Packer with AWS
- Getting Started with HCP Packer
- Fundamentals of HCP Packer
- Terraform Cloud Run Tasks for HCP Packer
- Channel Assignment History and Automated Rollback

Thank you

customer.success@hashicorp.com

www.hashicorp.com/customer-success