



UbuCon Korea 2023

퍼블릭 클라우드에서 Ubuntu Image 버전 관리

박준상 (JS Park, jsp@hashicorp.com)



- 클라우드 환경에서 사용되는 가상 서버 이미지(Ubuntu Image) 관리 방안을 다음 솔루션을 사용하여 알아보니다.
 - 가상 서버 이미지 생성 자동화, 가상 서버 이미지 아티팩트 저장소, 클라우드 인프라 생명 주기 관리
- 위 도구들을 사용, 클라우드 환경에서 **Ubuntu Image**를 복수의 클라우드 환경에서 사용하는 방법을 실습합니다.

목차



- Who Am I
- #1. 클라우드 서버 이미지 생명 주기 관리
- #2. 서버 이미지와 인프라 생명 주기 관리 연계
- #3. 실습

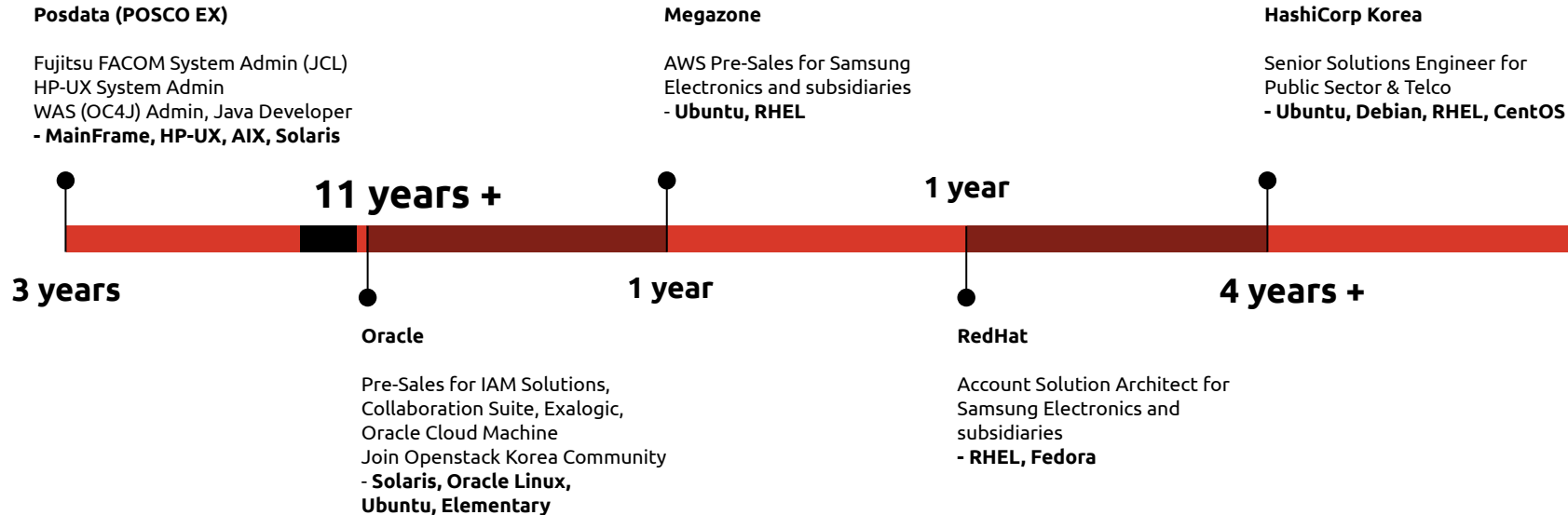


Who Am I

메인 프레임에서 클라우드 까지



Ops로 시작해 Dev를 만난 20년차 Pre-Sales 엔지니어





#1. 클라우드 서버 이미지 생명 주기 관리

수 작업 기반, 개별 클라우드 이미지 관리



AMI

Image artifact on AWS



VM Image

Image artifact on Azure



Machine image

Image artifact on VMware



Virtual Machine image

Image artifact on Openstack



Packer

모든 클라우드의 모든 이미지를 구축, 거버넌스 및
관리하기 위한 인프라 자동화 표준



코드형 이미지 (Images as Code)

고객화

Ansible
Bash
Powershell
...

다양한 클라우드

AWS
Azure
GCP
...

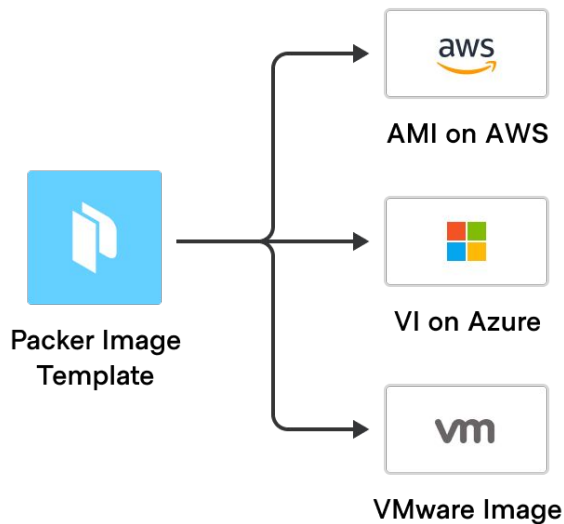
프라이빗 DC

VMware
VirtualBox
Vagrant
...

Packer, 단일 워크 플로우 기반 클라우드 이미지 통합 관리



- **사용 편의성**
HCL을 사용, Terraform처럼 손쉬운 연동.
- **확장성**
사용 환경과 무관하게 단일 언어로 이미지 관리.
타 구성 관리 도구와 연계.
- **Community 기반**
기존 커뮤니티 템플릿을 활용 및 직접 작성.



Packer, 핵심 기능



1 Builder

다양한 클라우드 및 온프리미엄
인프라에 대한 이미지 생성을 담당



2 Provisioner

기본 제공 소프트웨어 및 타사 소프트웨어를
사용, 이미지를 설치하고 구성할 수 있도록
시스템을 준비



3 Post-Processor

이미지 생성 후 작업 실행.
아티팩트를 업로드하거나 다시 패키징.



4 Data Source

구성 시 외부 데이터 사용.



서버 이미지 관리 표준화를 위한 사전 준비 사항은?



어떤 것이 "골드" 이미지이고 어떤 버전을 사용해야 하는가?



이미지 하드닝, 보안 및 규제 준수를 위한
이상적인 워크플로우는?
클라우드 전반에 걸쳐 복제할 수 있는가?

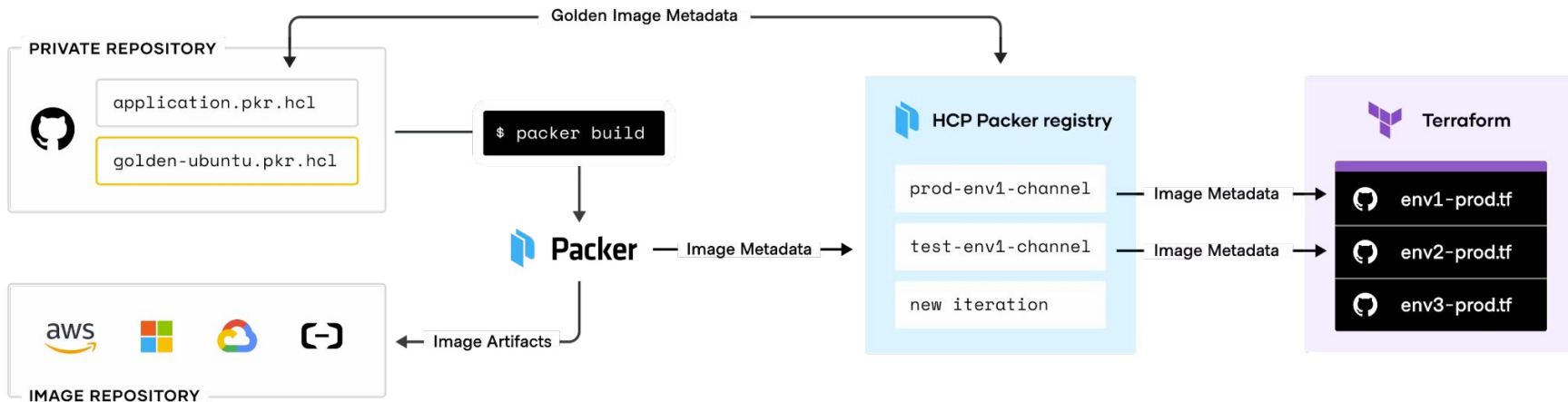


현재 프로비저닝 워크플로우에 이 프로세스를 통합할 수 있는가?

HCP Packer, 코드 기반 표준화를 통한 자동화



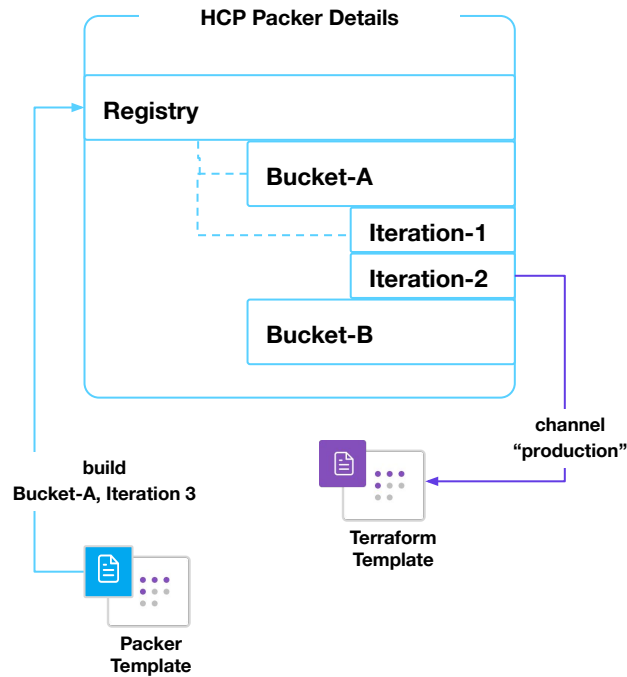
- **HCP Packer**는 프로비저닝 파이프라인에서 이미지 업데이트를 표준화, 보안 및 자동화하기 위한 멀티 클라우드 이미지 레지스트리입니다.
- 멀티 클라우드 골든 이미지 파이프 라인의 토대를 제공



HCP Packer 구성 요소



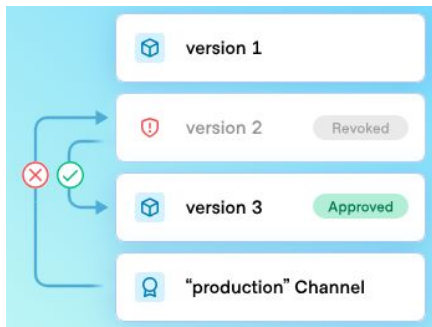
- **Registry** : 이미지 관련 메타 데이터 저장 서비스. 모든 이미지 버킷 조회 가능
- **Bucket** : 단일 **Packer 템플릿** 이미지 메타데이터 저장소
- **Iteration** : 성공한 개별 **Packer** 이미지 빌드 정보를 저장. (=버전).
- **Channel** : 버킷 내 사용하고자 하는 이미지의 기본 버전(**Iteration**)을 지정. Packer 템플릿과 Terraform Template에서 사용 가능. 버전 관리 표준화.



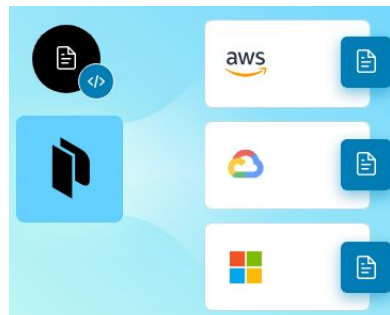
HCP Packer, 주요 쓰임새



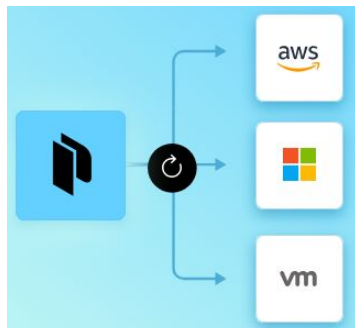
이미지 보안



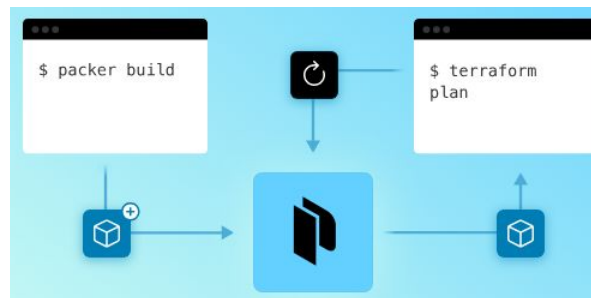
배포 시간 단축



클라우드 간 이미지 업데이트



테라폼과 연계

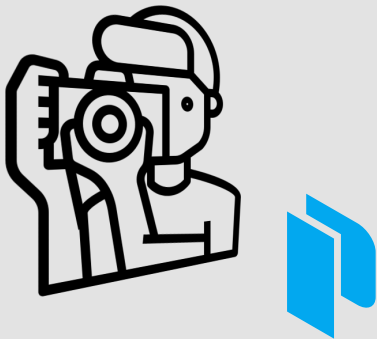


Packer vs. HCP Packer



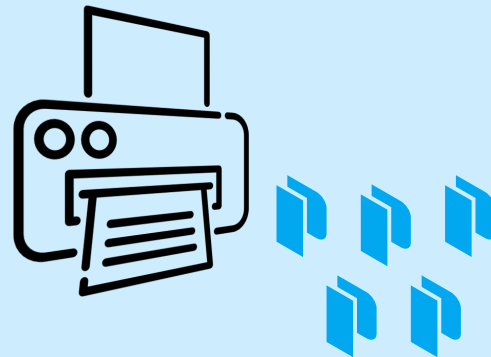
Packer

- 조직에서 준수해야 할 요구 사항을 충족하는 이미지 생성/구축
- 필요 시 업데이트를 통한 이미지 유지 관리



HCP Packer

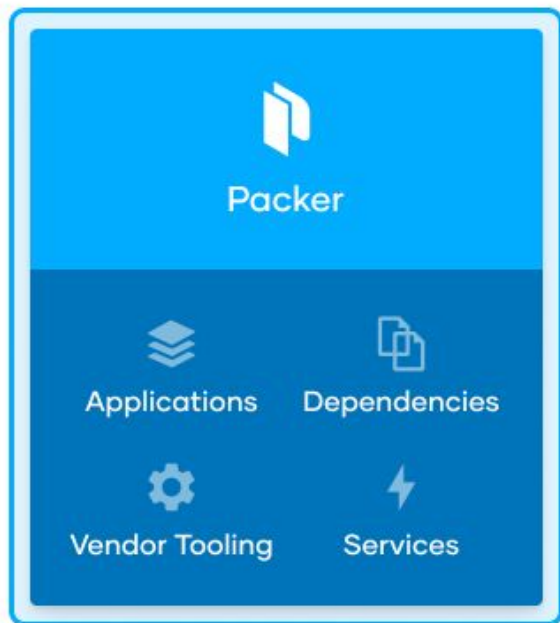
- Packer로 빌드한 이미지 메타 데이터 저장
- 단일 소스 템플릿을 사용, 여러 플랫폼에 동일한 머신 이미지의 빠른 배포



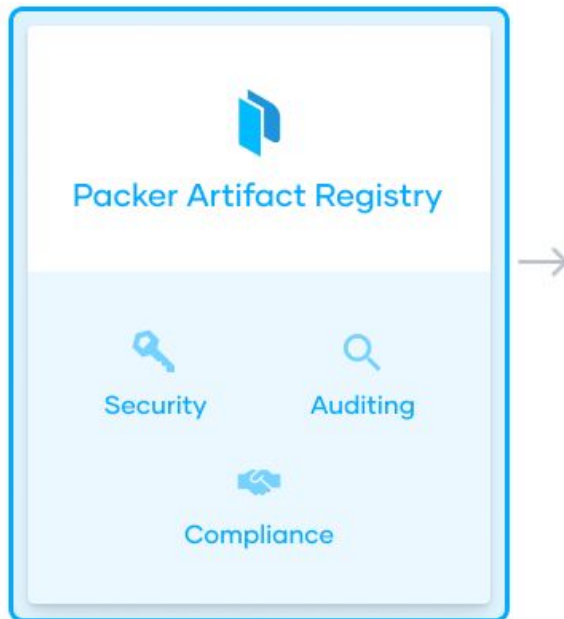


#2. 서버 이미지와 인프라 생명 주기 관리 연계

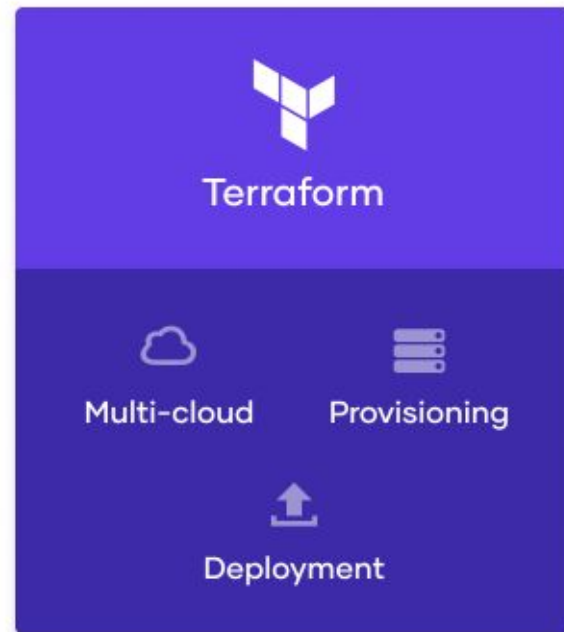
서버 이미지 배포 파이프 라인



Packer 커뮤니티 버전
서버 이미지 아티팩트 생성



HCP Packer
다양한 클라우드 환경에서 서버 이미지 생성, 추적 관리 및 거버넌스 과정의 복잡도 완화



Terraform(*)
서버 이미지 아티팩트를 이용, 자원 생명 주기 관리(배포/변경/삭제)

테라폼, 인프라 생명 주기 관리 자동화

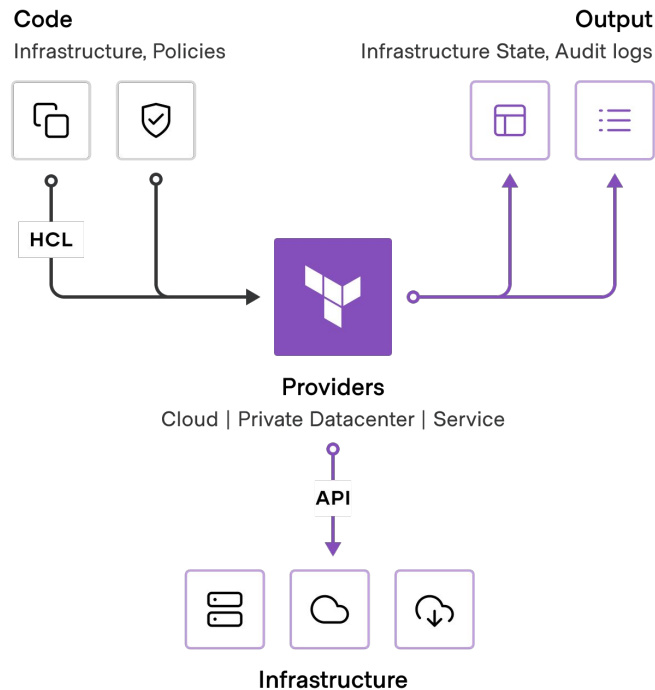


코드형 인프라(IaC)를 통한 클라우드 인프라 구성, 규제 준수 및 관리 자동화 솔루션

- ☑ **클라우드 관리 및 규정 적용**
단일 Workflow를 통해 인프라 프로비저닝과 Compliance 관리를 실현
- ☑ **인프라 셀프 서비스 기반**
사용자(개발자)가 직접 승인된 인프라 모듈 라이브러리를 사용, 시의적절하게 인프라를 쉽게 프로비저닝 할 수 있도록 지원
- ☑ **코드 기반 서비스 재해 복구 (DR)**
추가 비용없는 RTO 준수 및 빠른 장애 복구

☑ Business benefit

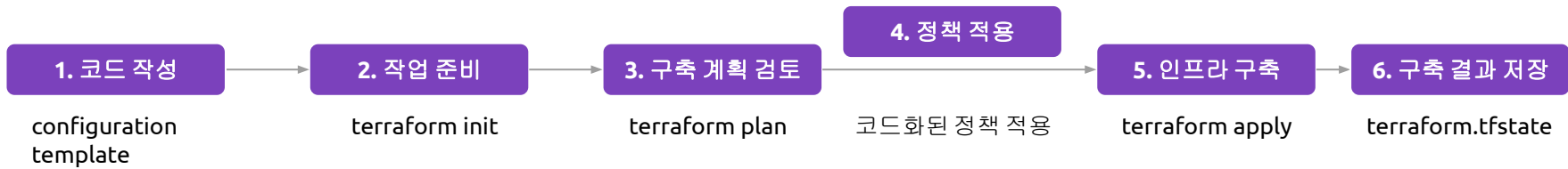
- 자동화를 통한 장애 최소화 / 운영 효율성 향상
- 개발자 생산성 향상 / Time-to-market
- 자원의 효율적인 활용을 통한 비용 절감



테라폼 엔터프라이즈(설치형, 관리형) 동작 방식



GUI/API 또는 CLI 사용



Terraform Cloud + HCP Packer



개선된 클라우드 기반 서버 이미지 관리 방안 제공

HCP Packer

- **Build** 서버 이미지 생성
- **Publish** 메타 데이터 저장
- **Promote** 채널 기반 관리
- **Revoke** 원복 및 삭제
- **View** 참조 및 이력 조회



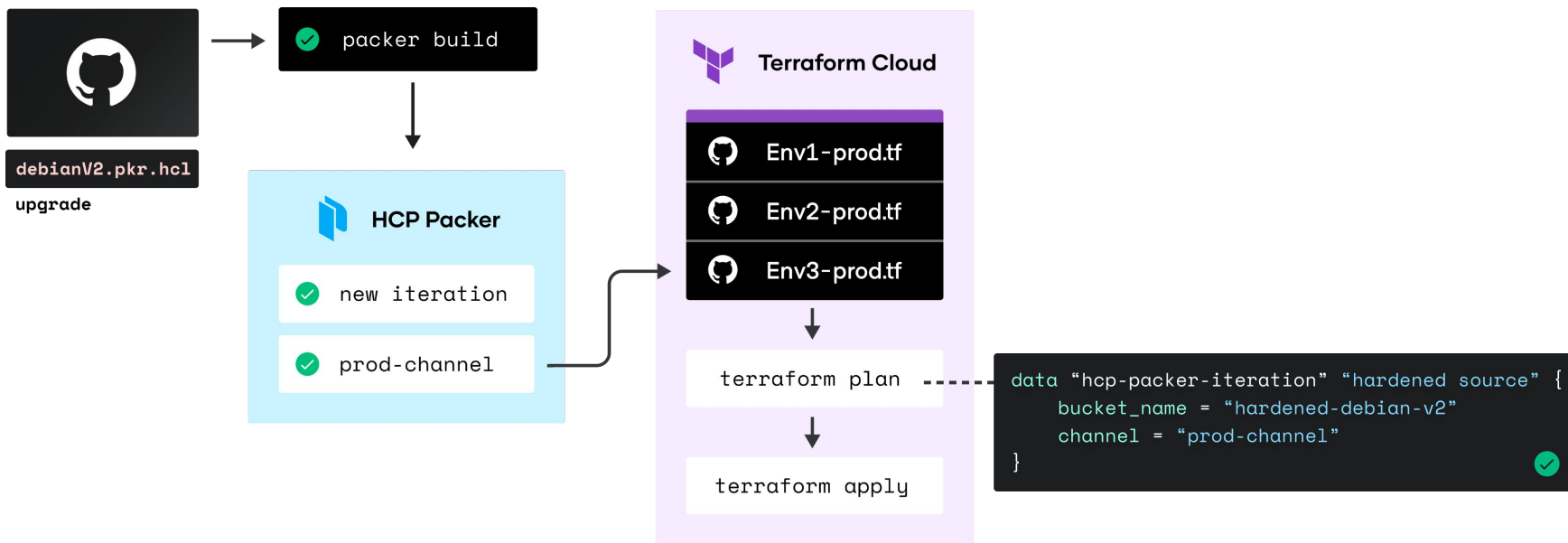
Terraform Cloud

- **Discover** 대상 이미지 검색
Data source
- **Provision** 인프라 배포
- **Validate** 이미지 상태 검증

Terraform Cloud + HCP Packer



통합된 서버 이미지 생성, 배포 워크 플로우 제공

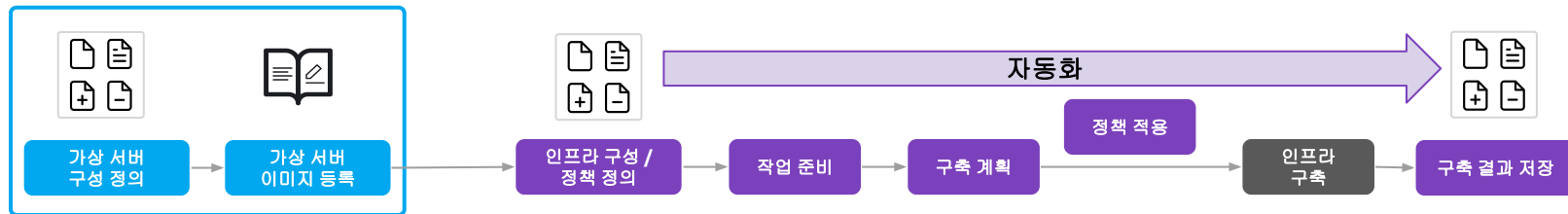


기대 효과



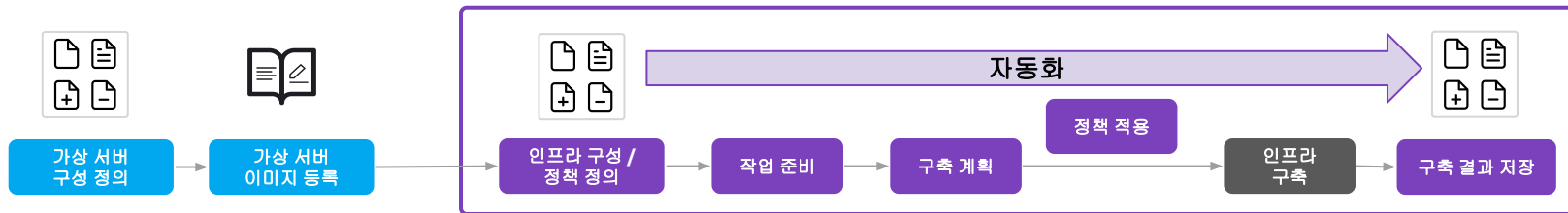
- 표준화된 인프라 형상 관리
- 인프라 관리 작업 방식 표준화
- 운영 표준 및 정책 적용 자동화

#1. 표준화된 인프라 형상 관리



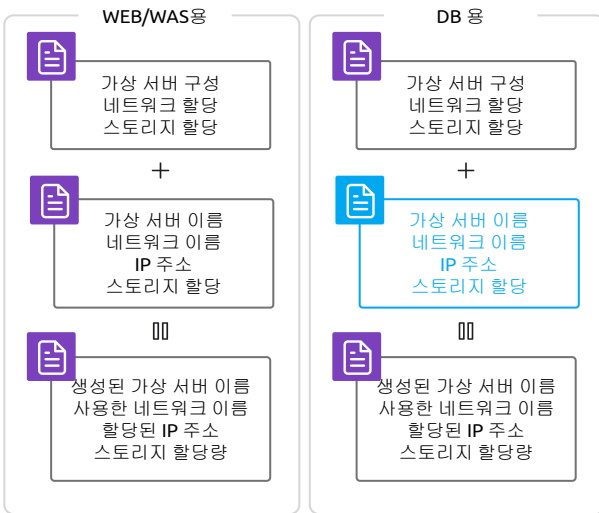
- 조직별/업무별 가상 서버 형상을 코드로 표준화
 - 필요한 소프트웨어 설치, 보안 조치 등의 작업을 가상 서버 배포 전 완료
 - 보안 및 OS 패치처럼 이미지 변경 시, 공통 이미지를 변경 후 적용
- 구성이 상이하여 발생할 수 있는 서비스 중단 및 장애 예방

#2. 인프라 관리 작업 방식 표준화



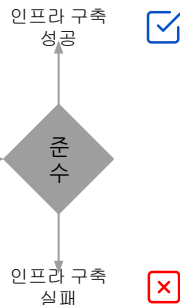
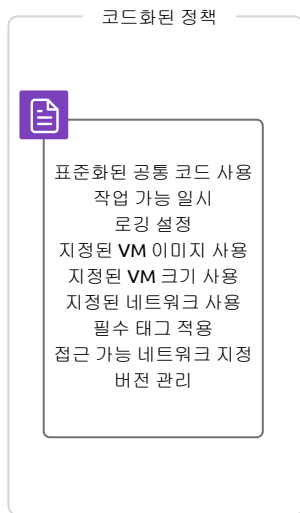
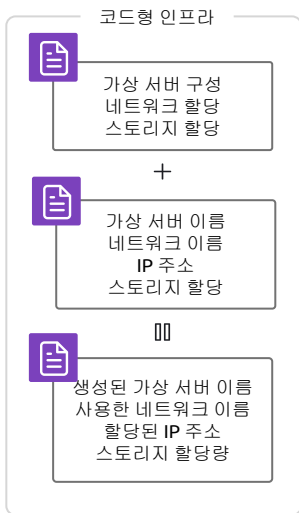
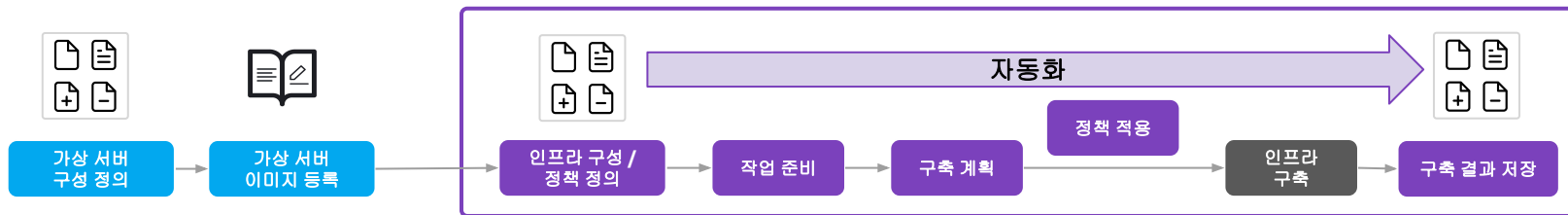
인프라 구축 작업
 전체 코드의 **90% 이상**
 → 인프라 규모와 무관하게
 재사용성이 높음

변수값 지정
 환경 별로 다른 값을 변수화
 전체 코드의 **10% 이내**



- 인프라 구성을 모두 코드로 관리.
- 수작업 최소화
 → 구성 오류로 인한 서비스 장애 최소화
- 단순 반복 작업 코드화
 → 작업 시간 및 인건비 절감

#3. 운영 표준 및 정책 적용 자동화

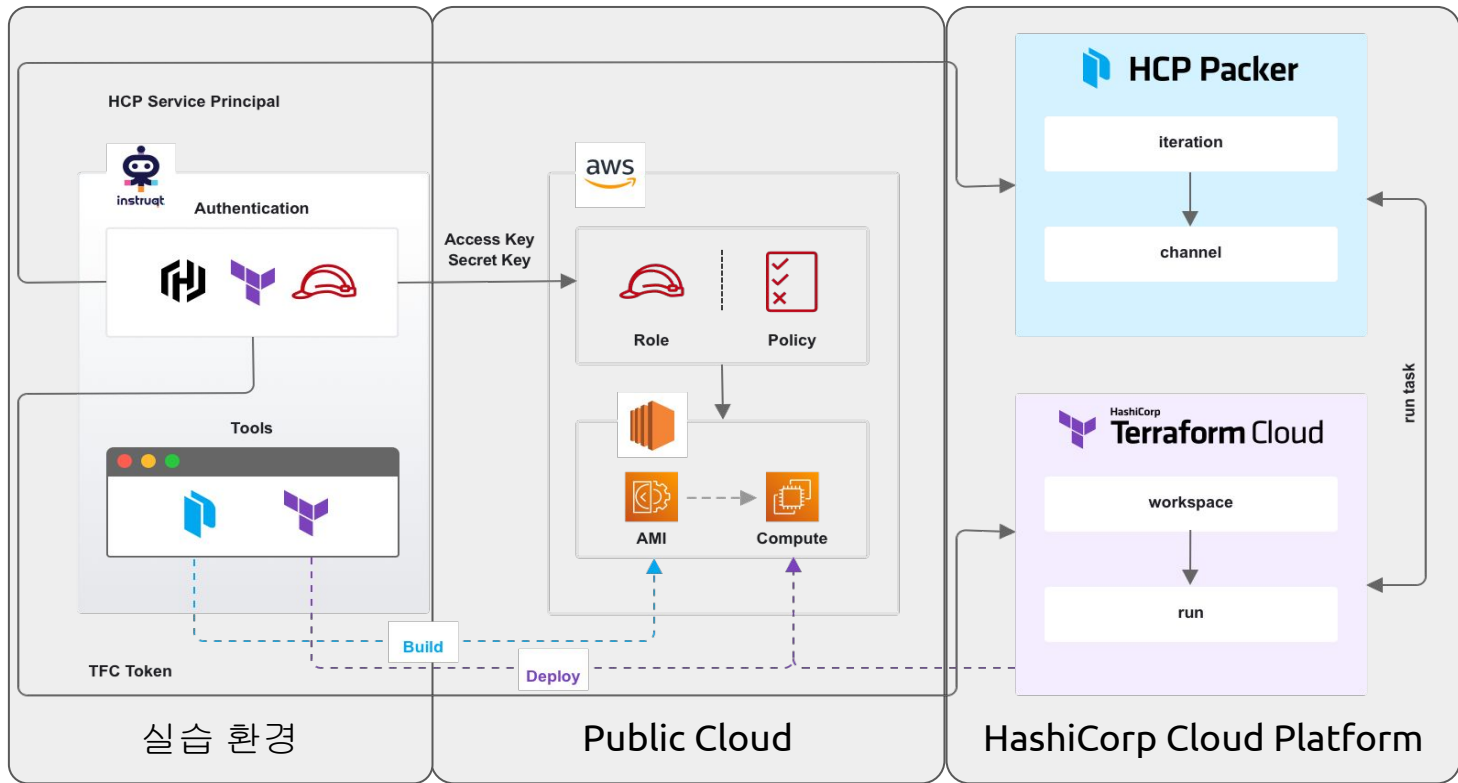


- 인프라 구성 시 보안 적용 강화
- 인프라 배포 시 코드화된 정책을 적용
→ 보안 표준 및 운용 표준에 부합하는 경우에만 인프라를 구성.



#3. 실습

실습 아키텍처





1	Runbook을 패커 템플릿으로	<ul style="list-style-type: none">• 작업 절차가 정리된 Runbook을 패커 템플릿화• 이미지 생성 작업 시간 단축• 작업 표준화로 오류 최소화
2	골든 이미지 기반 배포 (AWS → Azure)	<ul style="list-style-type: none">• 골든 이미지를 AWS AMI로 저장• 저장된 이미지를 사용, AWS에 가상서버 배포• 이미지를 재사용, Azure로 서비스 확장
3	실습 환경 정리	<ul style="list-style-type: none">• CSP 내 가상 서버 삭제• 필요 시 HCP 내 환경 삭제 및 정리• Shadow IT 예방

#0. 사전 준비 작업



1. 실습 환경 계정 생성 및 준비
2. HashiCorp Cloud Platform 계정 생성 및 환경 준비
3. Terraform Cloud 계정 생성 및 환경 준비

#0. 사전 준비 작업



1. 실습 환경 계정 생성 및 준비
2. HashiCorp Cloud Platform 계정 생성 및 환경 준비
3. Terraform Cloud 계정 생성 및 환경 준비



#0. 사전 준비 작업

1. 실습환경 계정 생성 및 준비

- 실습 환경 계정 생성

instruqt

Create account Login SSO

Google Github Twitter

Or

Full name
Full name

Email
Email

Password
Password

Create account

Instruqt's terms of services apply. By signing up you agree to receive our regular updates in accordance with our privacy policy.



<https://play.instruqt.com/signup>



#0. 사전 준비 작업

1. 실습환경 계정생성 및 준비

- 실습 환경 준비 : 다음 URL에 접속, 실습 환경 사용을 위한 동의서 제출

The image shows a QR code on the left and a screenshot of a HashiCorp content page on the right. The page displays 'KR-UbuCon-Packer' content and a registration form. The form includes fields for First name, Email address, Company name, Phone number, Job title, and Job level. A modal window is overlaid on the form, titled 'To access this content, please leave your details.' and 'Accept Hashicorp terms'. The modal contains a 'Download' button for the 'Evaluation Agreement (Instructq)' and a 'Submit and access' button. The page also shows 'Available content (1)' with 'Path to Packer' listed.

<https://play.instructq.com/hashicorp/invite/4njvh83uzqo7>



#0. 사전 준비 작업

1. 실습환경 계정생성 및 준비

- 실습 환경 준비 : 'Start' 버튼 클릭

KR-UbuCon-Packer
UbuCon Korea 2023 클라우드 이미지 관리 방안 실습입니다.

Available content (2) Share

- Path to Packer**
Explore the Path to Packer by using HCP Packer and Terraform Cloud View details Start
- machine image 관리 - Packer, HCP Packer** Start

#0. 사전 준비 작업

1. 실습환경 계정생성 및 준비



- 실습 환경 준비 :

A screenshot of the 'Path to Packer' interactive learning interface. The interface is dark-themed and features a central 3D graphic of a box labeled 'HashiCorp Cloud Platform' with a blue square on top. To the right of the graphic, the text reads 'Path to Packer' and 'Explore the Path to Packer by using HCP Packer and Terraform Cloud'. The interface includes a 'Share' button, a 'Progress' indicator, an 'Exit' button, and a 'Notify me' button. At the bottom, it says 'Patience is a virtue, and you're a virtuous user!' and 'Estimated time left: 02:56'.

Powered by **instruct**

Share Progress

Path to Packer
Explore the Path to Packer by using
HCP Packer and Terraform Cloud

Exit

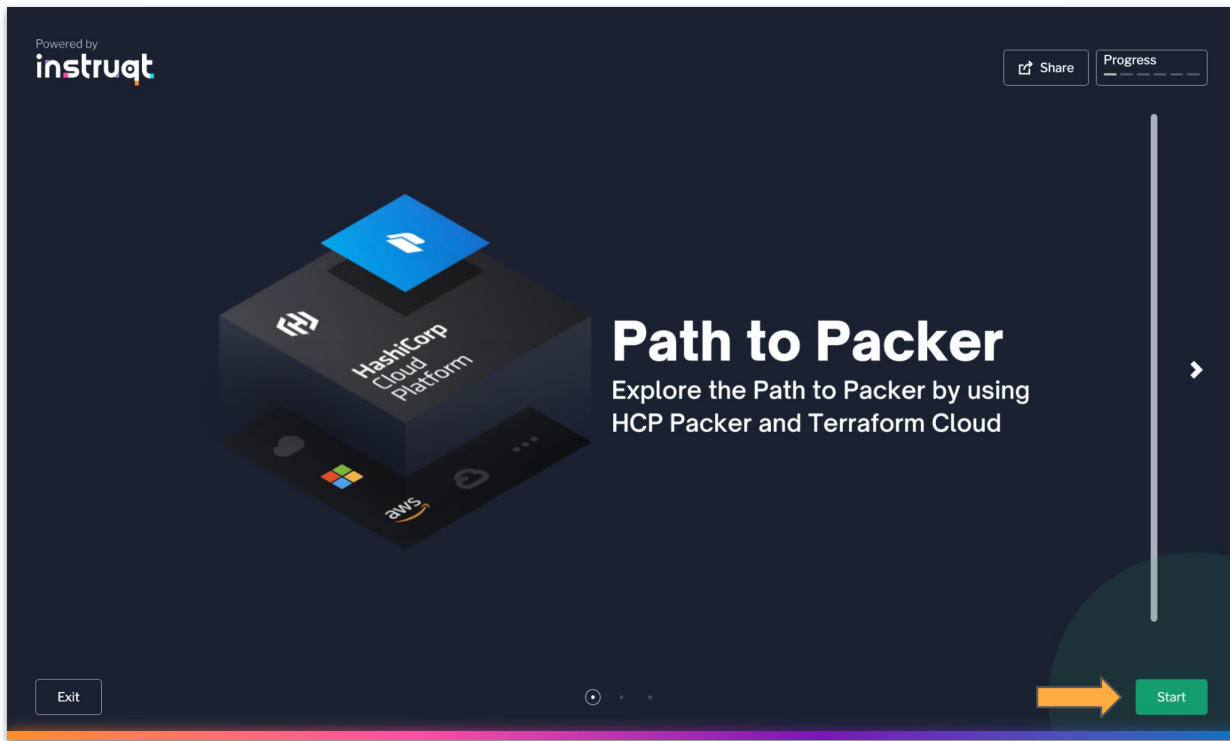
Patience is a virtue, and you're a virtuous user!
Estimated time left: 02:56 - [Notify me](#)

#0. 사전 준비 작업

1. 실습환경 계정생성 및 준비



- 실습 환경 준비 완료 : “Start” 버튼이 활성화 (잠시 대기). 브라우저 다른 탭/창 열기



#0. 사전 준비 작업



1. 실습 환경 계정 생성 및 준비
2. HashiCorp Cloud Platform 계정 생성 및 환경 준비
3. Terraform Cloud 계정 생성 및 환경 준비

#0. 사전 준비 작업

2.HashiCorp Cloud Platform 계정 생성 및 환경 준비



- HashiCorp Cloud Platform 계정 생성

The image shows a sequence of three overlapping screenshots from the HashiCorp Cloud Platform sign-up process. The first screenshot on the left is the landing page with the text "Get started in minutes with our cloud products" and buttons for "Create an account" and "Sign in". The middle screenshot shows the "Create Your Account" form with fields for "Email address" and "Continue". The rightmost screenshot is a "Service agreement" modal with checkboxes for "I agree to the terms of use" and "I accept the privacy policy", an "Email opt-in" section, and a "Continue" button. Overlaid on the right is a "Email verification sent" notification box with the text "We've sent you a link to verify your email address" and buttons for "Resend email verification" and "Wrong email? Start over".

<https://portal.cloud.hashicorp.com/sign-up>

#0. 사전 준비 작업

2.HashiCorp Cloud Platform 계정 생성 및 환경 준비



- 이메일 인증 링크를 클릭하여 접속

Create organization

A HashiCorp Cloud Platform organization allows you to deploy and manage HCP services and invite collaborators.

Organization name
Names can contain letters, numbers, spaces, and dashes. They also must start with a letter and contain 3-36 characters.

Country
Select your organization's country. This is used to ensure we provide accurate pricing information and can be adjusted when you add a payment method.

[Create organization](#)

Account summary

Payment methods

jsp-ubucon-org

This month's summary

Usage
\$0.00
Sep 3, 2023

\$ Trial

Payment method

Remaining credits
\$50.00

[Add credit card](#)
To continue usage once your credits have been used, please add a payment method.

[Add credit card](#) →

Usage since Sep 3, 2023

Last updated Sep 3, 2023, 12:00:00 AM
[Billing documentation](#)

Expand all

jsp-ubucon-project	\$0.00
--------------------	--------

Resources will appear here once there is usage.



#0. 사전 준비 작업

2.HashiCorp Cloud Platform(HCP) 계정 생성 및 환경 준비

- HashiCorp Packer Registry 생성 : HCP 내 프로젝트 접속 > Packer 선택 > 'Create a free registry'

클릭

The screenshot shows the HCP interface with a dark sidebar on the left and a main content area on the right. The sidebar contains a list of services: Boundary, Consul, Packer, Vault, Vault Secrets (Beta), Terraform, Vagrant, and Waypoint (Beta). The 'Packer' service is highlighted with a grey background and an orange arrow pointing to it. Another orange arrow points to the 'jsp-ubucon-project' entry in the sidebar. In the main content area, the breadcrumb path is 'jsp-ubucon-org / jsp-ubucon-project / Packer'. Below the breadcrumb, the title 'HCP Packer' is displayed, followed by a description: 'Bridge the gap between image creation and deployment with image management workflows for development and security teams.' A blue button labeled '+ Create a free registry' is prominently displayed. Below this, there are three sections: 'Simplified lifecycle management' (describing how the registry reduces complexity), 'Centralized governance' (describing how to define release channels), and 'Supports your current workflows' (describing how to use existing workflows). To the right of these sections is a diagram showing a workflow: a terminal window with '\$ packer build' feeds into the 'HCP Packer registry' box, which then feeds into another terminal window with '\$ terraform plan'. A 'Fetch Latest' arrow points from the registry back to the '\$ packer build' terminal. Below the diagram is a 'Learn more' section with a link: 'Push Image Metadata to the HCP Packer Registry'.

#0. 사전 준비 작업

2.HashiCorp Cloud Platform(HCP) 계정 생성 및 환경 준비



- Image Registry 생성 완료

A screenshot of the HashiCorp Cloud Platform (HCP) interface. On the left is a dark sidebar with the HashiCorp logo at the top, followed by a search bar containing 'NO', a 'Back to HCP Overview' link, and a menu with 'Packer', 'Images' (highlighted with an orange arrow), 'Settings', and 'Pricing'. The main content area shows the breadcrumb 'jsp-ubucon-org / jsp-ubucon-project / Packer / Images' and the title 'Images'. Below the title is the text 'Packer registry' and two buttons: 'Integrate with Terraform Cloud' and 'Manage'. A section titled 'No image buckets' contains the text 'Re-run `packer build` after configuring Packer to use your HCP Packer Registry.' and a blue link 'Push Image Metadata to the HCP Packer Registry' with a plus icon. The footer includes 'Platform Fully Operational', 'Changelog', 'Support', 'Terms', 'Privacy', 'Security', and '©2023 HashiCorp'.

#0. 사전 준비 작업

2.HashiCorp Cloud Platform(HCP) 계정 생성 및 환경 준비



- Service Principal 생성 : 'Back to HCP Overview' 클릭

A screenshot of the HashiCorp Cloud Platform (HCP) interface. On the left is a dark sidebar with the HashiCorp logo at the top, followed by a search bar containing 'NO', and a list of navigation items: '< Back to HCP Overview', 'Packer', 'Images', 'Settings', and 'Pricing'. An orange arrow points to the '< Back to HCP Overview' link. The main content area shows the breadcrumb 'jsp-ubucon-org / jsp-ubucon-project / Packer / Images' and the title 'Images'. Below the title is a 'Packer registry' section with a dropdown menu set to 'Integrate with Terraform Cloud' and a blue 'Manage' button. The main content area displays 'No image buckets' with instructions to re-run 'packer build' after configuring Packer to use the HCP Packer Registry. A blue link 'Push Image Metadata to the HCP Packer Registry' with a plus icon is also visible. The footer contains the status 'Platform Fully Operational', links for 'Changelog', 'Support', 'Terms', 'Privacy', and 'Security', and the copyright notice '©2023 HashiCorp'.



#0. 사전 준비 작업

2.HashiCorp Cloud Platform(HCP) 계정 생성 및 환경 준비

- Service Principal (Org Level) 생성 : "Org 선택 후" 'Access Control (IAM)' 선택

The screenshot displays the HashiCorp Cloud Platform (HCP) interface for the organization 'jsp-ubucon-org'. The left sidebar contains navigation links: 'jsp-ubucon-org', 'Projects', 'Access control (IAM)', 'Billing', and 'Organization settings'. Two orange arrows highlight the 'jsp-ubucon-org' link and the 'Access control (IAM)' link. The main content area shows the organization dashboard with the following sections:

- Billing summary**: Shows usage of \$0.00 and a notification to add a credit card. The notification states: "Add credit card. You have \$50.00 credits. To continue using paid HCP services once these are depleted, please add a payment."
- Principals**: Shows users invited to the organization and Service Principals created at the organization scope.
- Projects**: Shows 1 active project. The text below states: "Projects created in the organization to deploy and manage HCP resources with segmented access."



#0. 사전 준비 작업

2.HashiCorp Cloud Platform(HCP) 계정 생성 및 환경 준비

- Service Principal (Org Level) 생성 : 'Service Principals' 선택

The screenshot displays the HashiCorp Cloud Platform (HCP) interface. On the left, a dark sidebar contains a navigation menu with the following items: 'Back to Dashboard', 'Access control (IAM)', 'Users', 'Groups', 'Pending invites', and 'Service principals'. An orange arrow points to the 'Service principals' item. The main content area is white and shows the breadcrumb 'jsp-ubucon-org / Access control (IAM) / Service principals', the title 'Service principals', a blue '+ Create service principal' button, and a message 'No service principals found' with a link to '+ Create a service principal'.

#0. 사전 준비 작업

2.HashiCorp Cloud Platform(HCP) 계정 생성 및 환경 준비



- Service Principal (Org Level) 생성 : '+ Create service principal' 선택

The screenshot shows the HashiCorp Cloud Platform (HCP) interface for managing Service Principals. On the left is a dark sidebar with the HashiCorp logo and navigation options: 'Back to Dashboard', 'Access control (IAM)', 'Users', 'Groups', 'Pending invites', and 'Service principals' (which is highlighted). The main content area is titled 'Service principals' and shows a breadcrumb path: 'jsp-ubucon-org / Access control (IAM) / Service principals'. A blue button labeled '+ Create service principal' is prominently displayed, with an orange arrow pointing to it from the sidebar. Below this button, the text reads 'No service principals found' followed by an explanatory paragraph: 'After creating a service principal, it will appear here for you to manage. Click the link below to get started.' A blue link '+ Create a service principal' is provided at the bottom of the main content area. The footer contains the status 'Platform Fully Operational', links for 'Changelog', 'Support', 'Terms', 'Privacy', and 'Security', and the copyright notice '©2023 HashiCorp'.

#0. 사전 준비 작업

2.HashiCorp Cloud Platform(HCP) 계정 생성 및 환경 준비



- Service Principal (Org Level) 생성 : 'Role'은 'Contributor'로, 이름은 동일하거나 원하는 이름 입력 후 생성

A screenshot of the HashiCorp Cloud Platform (HCP) interface showing the 'Create service principal' page. The page is titled 'Create service principal' and is part of the 'Service principals' section under 'Access control (IAM)'. The breadcrumb path is 'jsp-ubucon-org / Access control (IAM) / Service principals / Create service principal'. The form includes a 'Service principal name' field with the value 'ubuconf-demo', a 'Role' dropdown menu with 'Contributor' selected, and a 'Create service principal' button. The 'Service principal name' field is marked as 'Required'. The 'Role' dropdown is also marked as 'Required'. Below the role selection, there is a description: 'Can create and manage all types of resources but can't grant access to others.' At the bottom of the form, there are two buttons: 'Create service principal' (blue) and 'Cancel' (white). The left sidebar shows the navigation menu with 'Service principals' highlighted. Three orange arrows point from the sidebar to the form fields: one to the 'Service principal name' field, one to the 'Role' dropdown, and one to the 'Create service principal' button. The footer of the page includes the text 'Platform Fully Operational Changelog Support Terms Privacy Security' and the HashiCorp logo with the copyright notice '©2023 HashiCorp'.

#0. 사전 준비 작업

2.HashiCorp Cloud Platform(HCP) 계정 생성 및 환경 준비



- Service Principal 생성 : 'Create service principal key' 클릭하여 key 생성

A screenshot of the HashiCorp Cloud Platform (HCP) interface. On the left is a dark sidebar with the HashiCorp logo and navigation links: 'Back to Dashboard', 'Access control (IAM)', 'Users', 'Groups', 'Pending invites', and 'Service principals' (which is highlighted). The main content area shows the details for a service principal named 'ubuconf-demo'. At the top right of this area is a 'Manage' button. Below it is a table with two columns: 'ID' and 'Role'. The 'ID' column contains the value 'ubuconf-demo-238580@27801bbe-6cf1-4661-b073-d4af157a4c1b' with a copy icon. The 'Role' column contains the value 'Contributor'. Below the table, it shows 'Created' as '39 seconds ago'. A horizontal line separates this section from the 'Keys' section below. The 'Keys' section has the heading 'Keys' and a message: 'No keys found. Create a key to allow API access to your HashiCorp Cloud Platform account.' At the bottom of this section is a blue button labeled 'Create service principal key' with an orange arrow pointing to it from the left.

#0. 사전 준비 작업

2.HashiCorp Cloud Platform(HCP) 계정 생성 및 환경 준비



- Service Principal 생성 : 'Client ID' 와 'Client Secret' 정보 화면 출력, 해당 정보 복사 후 별도 보관 후 'Close'

A screenshot of the HashiCorp Cloud Platform (HCP) interface. The main window shows the 'Service principals' page for a user named 'ubunconf-demo'. A modal dialog box titled 'Save your key' is open in the center. The dialog contains the following information:

- A message: 'Your key was successfully generated.'
- A section for 'Client ID' with the value 'Vk3A...' followed by a text input field containing '2258xw'. An orange arrow points to a copy icon.
- A warning icon and text: 'Be sure to copy the client secret. You will not be able to retrieve it later.'
- A section for 'Client secret' with a long, partially obscured text input field ending in 'T6B-'. An orange arrow points to a copy icon.
- A 'Close' button at the bottom left of the dialog.

At the bottom of the screenshot, a green notification banner says 'Key generated' with a checkmark icon. The background interface includes a sidebar with navigation options like 'Back to Dashboard', 'Access control (IAM)', 'Users', 'Groups', 'Pending invites', and 'Service principals'. The top right of the main window has a 'Manage' dropdown menu.



#0. 사전 준비 작업

2.HashiCorp Cloud Platform(HCP) 계정 생성 및 환경 준비

- Org ID, Project ID 확인 : Project 선택 시, URL 창 에서 해당 정보 복사

'Orgs/Org ID/projects/Project ID'

portal.cloud.hashicorp.com/orgs/27801bbe-6cf1-4661-b073-d4af157a4c1b/projects/352160a5-3900-4d0f-97a...

jsp-ubucon-org / Projects / jsp-ubucon-project

jsp-ubucon-project

Project dashboard
An overview of all resources in the project. [Learn more](#)

Manage project

Active resources

2 active resources

There are 2 active resources inside this project.

[View active resources](#)

Billing summary

To view billing information across projects, go to the organization.

[View organization billing summary](#)



#0. 사전 준비 작업

2.HashiCorp Cloud Platform(HCP) 계정 생성 및 환경 준비

- HCP 필요 정보 확인 완료 : 복사 후 별도 보관

The screenshot shows the 'instruct' interface with a dark header bar containing 'Explainer', 'Terminal', and 'Cloud Credentials' tabs. The main content area is titled 'HCP Requirements' and includes a descriptive paragraph: 'To describe the relationship between an Amazon Machine Image (AMI) and HashiCorp Packer we need access to your HCP organization.' To the right, a form with an orange border contains four input fields: 'HCP Organization ID', 'HCP Project ID', 'HCP Client ID', and 'HCP Client Secret'. The 'HCP Organization ID' and 'HCP Client ID' fields have small square icons with arrows pointing to the top-right corner. A blue 'Save' button is located at the bottom of the form. Navigation arrows are visible at the bottom of the interface.

#0. 사전 준비 작업



1. 실습 환경 계정 생성 및 준비
2. HashiCorp Cloud Platform 계정 생성 및 환경 준비
3. Terraform Cloud 계정 생성 및 환경 준비

#0. 사전 준비 작업

2. Terraform Cloud 계정 생성 및 환경 준비



- Terraform Cloud 계정 생성 : 'Terraform' 선택 후 'Continue to Terraform Cloud' 선택

The screenshot shows the HCP interface. On the left is a dark sidebar with a navigation menu. The 'Terraform' option is highlighted with a grey background and a blue icon, and an orange arrow points to it from the left. The main content area is titled 'Terraform' and shows the breadcrumb 'jsp-ubucon-org / jsp-ubucon-project / Terraform'. Below the title, there is a section for 'Terraform Cloud' with a description and a 'Continue to Terraform Cloud' link. An orange arrow points from the 'Terraform' menu item to this link. Below this, there is a section for 'HashiCorp Cloud Platform Terraform Provider' with a description and a 'HashiCorp Cloud Platform Provider' link.

jsp-ubucon-org / jsp-ubucon-project / Terraform

Terraform

Terraform Cloud
Terraform Cloud is HashiCorp's managed Terraform offering that eliminates the need for unnecessary tooling and documentation to use Terraform in production.

Terraform Cloud runs on its own platform and allows you to provision infrastructure securely and reliably in the cloud.

[Learn more](#)

[Continue to Terraform Cloud](#)

HashiCorp Cloud Platform Terraform Provider
Write, plan, and apply your HCP services and networks with the Terraform Provider. Integrate faster with your applications and infrastructure.

[HashiCorp Cloud Platform Provider](#)

#0. 사전 준비 작업


2. Terraform Cloud 계정 생성 및 환경 준비



- Terraform Cloud 계정 생성 : 'Continue with HCP Account' 선택

HashiCorp
Terraform Cloud

Sign in to Terraform Cloud

 Continue with HCP account

OR

Username or email

Password

[Forgot password?](#)

Sign in

[Sign in with SSO.](#)

Need to sign up? Create your [free account](#).

View [Terraform Offerings](#) to find out which one is right

[Privacy - Terms](#)



#0. 사전 준비 작업


2. Terraform Cloud 계정 생성 및 환경 준비

- Terraform Cloud 계정 : 'Continue' 선택

Create a new HCP-linked account

Your HCP Account is your HashiCorp Cloud Platform login and the only credentials you need to access every HashiCorp product. From the HashiCorp Learn and community platforms, to Terraform Cloud and the Terraform Registry. Log in once and stay logged in.

i Some features, like two-factor authentication, are managed from the HashiCorp Cloud Platform.



#0. 사전 준비 작업

2. Terraform Cloud 계정 생성 및 환경 준비



- Terraform Cloud 환경 준비 : '이름약자3자-ubucon-org'를 조직명으로 설정 후, 'Create organization' 선택

A screenshot of the Terraform Cloud web interface showing the 'Create a new organization' form. The form is titled 'Create a new organization' and includes a search bar, a help icon, and a user profile icon in the top left. The main content area contains the following text and fields:

Organizations / New

Create a new organization

Organizations are privately shared spaces for teams to collaborate on infrastructure. [Learn more](#) about organizations in Terraform Cloud.

Organization name

Organization names must be unique and can only include numbers, letters, underscores (_), and hyphens (-).

Email address

The organization email is used for any future notifications, such as billing alerts, and the organization avatar, via [gravatar.com](#).

Create organization

An orange arrow points to the 'Create organization' button.

#0. 사전 준비 작업

2. Terraform Cloud 계정 생성 및 환경 준비



- Terraform Cloud 환경 준비 : 'CLI-driven Workflow' 선택

A screenshot of the Terraform Cloud workspace creation wizard. The interface is divided into a dark sidebar on the left and a main content area on the right. The sidebar contains navigation options: 'Manage' (with sub-items 'Projects & workspaces', 'Registry', 'Usage', 'Settings'), 'Visibility' (with 'Explorer Beta'), and 'Cloud Platform' (with 'HashiCorp Cloud Platform'). The main content area shows a progress bar with four steps: '1 Choose Type', '2 Connect to VCS', '3 Choose a repository', and '4 Configure settings'. The 'Choose your workflow' section is active, displaying three options: 'Version control workflow' (Most common), 'CLI-driven workflow' (highlighted with a grey background and an orange arrow pointing to it), and 'API-driven workflow'. Each option includes a brief description and a 'Learn More' link.

#0. 사전 준비 작업

2. Terraform Cloud 계정 생성 및 환경 준비



- Terraform Cloud 환경 준비 : 'path-to-packer'로 Workspace 이름 지정 후, 'Create Workspace' 선택

A screenshot of the Terraform Cloud web interface showing the 'Name Workspace' form. The left sidebar is dark with navigation options like 'Manage', 'Projects & workspaces', 'Registry', 'Usage', 'Settings', 'Visibility', 'Explorer Beta', and 'Cloud Platform'. The main content area is white and contains the form fields: 'Workspace Name' (with 'path-to-packer' entered), 'Project' (with 'Default Project' selected), and 'Description' (with a placeholder 'Workspace description'). At the bottom, there are two buttons: 'Create workspace' (highlighted with an orange arrow) and 'Cancel'.

Name Workspace

Workspace Name

path-to-packer

The name of your workspace is unique and used in tools, routing, and UI. Dashes, underscores, and alphanumeric characters are permitted. [Learn more about naming workspaces](#)

Project

Default Project

Every workspace must belong to a single project. Projects must be named uniquely within an organization. Workspaces may be moved between projects at any time from the workspace list or settings. [Learn more about projects](#)

Description

Optional

Workspace description

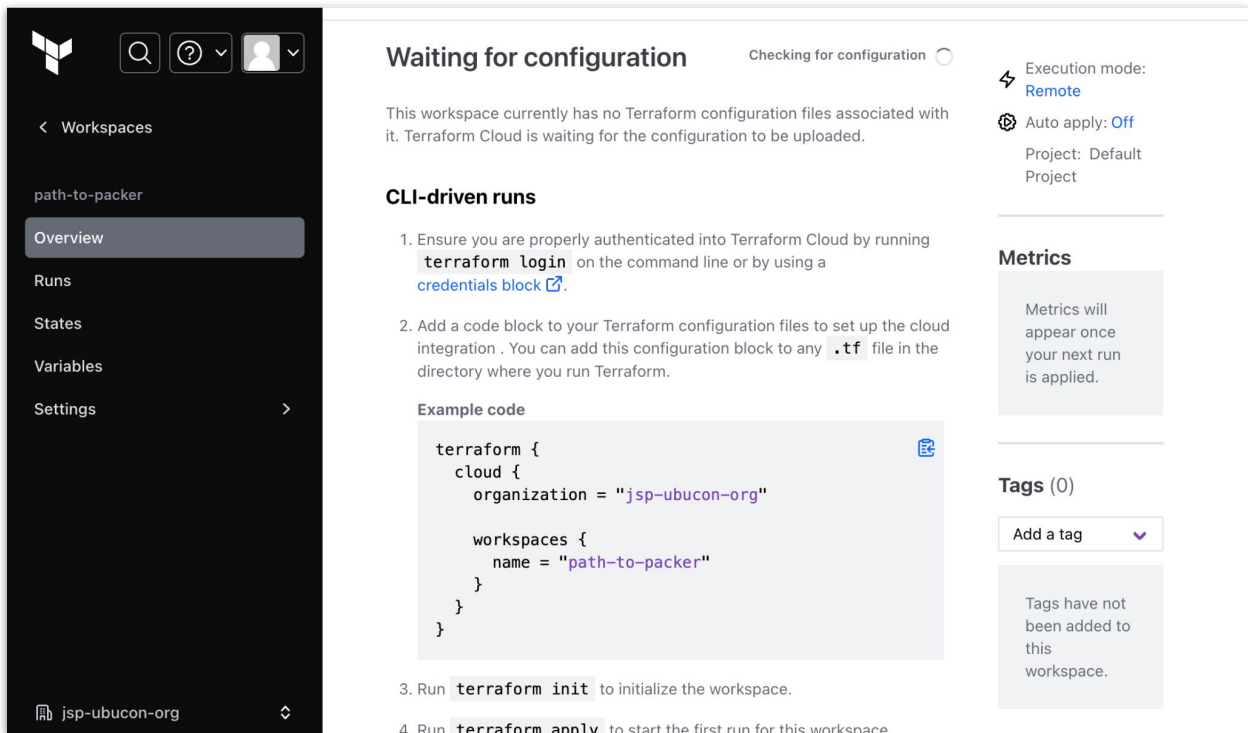
Create workspace Cancel

#0. 사전 준비 작업

2. Terraform Cloud 계정 생성 및 환경 준비



- Terraform Cloud 환경 준비 : 아래와 같이 **Workspace** 생성 완료



The screenshot shows the Terraform Cloud interface for a workspace named 'path-to-packer'. The left sidebar contains navigation options: Overview (selected), Runs, States, Variables, and Settings. The main content area is titled 'Waiting for configuration' and indicates that Terraform Cloud is waiting for configuration files. It provides instructions for CLI-driven runs, including authentication and adding code blocks to the configuration files. An example code block is shown with the following Terraform configuration:

```
terraform {
  cloud {
    organization = "jsp-ubucon-org"

    workspaces {
      name = "path-to-packer"
    }
  }
}
```

On the right side, there are settings for 'Execution mode' (Remote), 'Auto apply' (Off), and 'Project' (Default Project). Below these are sections for 'Metrics' and 'Tags (0)'. The 'Metrics' section states that metrics will appear once the next run is applied. The 'Tags' section has an 'Add a tag' button and a message stating that tags have not been added to this workspace.

#0. 사전 준비 작업

2. Terraform Cloud 계정 생성 및 환경 준비



- Terraform Cloud 환경 준비 : 좌측 상단 사람 모양 클릭 후 'User Settings' 선택

A screenshot of the Terraform Cloud web interface. On the left, a dark sidebar contains navigation options: Workspaces, path-to-packer, Overview (highlighted), Runs, States, Variables, and Settings. The main content area shows the 'path-to-packer' workspace overview. At the top left of the main area, a user profile icon is clicked, opening a dropdown menu with options: 'Signed in as' (with a username field), 'User Settings', and 'Sign out'. An orange arrow points from the user icon to the 'User Settings' option. The main content area displays 'path-to-packer' workspace details, including 'Resources: 0' and 'Terraform version: 1.5.6'. Below this, it shows 'Updated 4 minutes ago' and 'Unlocked' status. A section titled 'Waiting for configuration' indicates that the workspace is currently without Terraform configuration files. To the right, there are settings for 'Execution mode: Remote', 'Auto apply: Off', and 'Project: Default Project'. At the bottom, there are instructions for 'CLI-driven runs' and a 'Metrics' section.

#0. 사전 준비 작업

2. Terraform Cloud 계정 생성 및 환경 준비



- Terraform Cloud 환경 준비 : Tokens 선택 후 'Create an API token' 선택

The screenshot shows the Terraform Cloud user interface. On the left is a dark sidebar with navigation options: User Settings, Profile, Sessions, Organizations, Password, Two Factor Authentication, SSO, and Tokens. An orange arrow points to the 'Tokens' option. The main content area is titled 'Settings / Tokens' and contains the following text: 'Tokens', 'Your API tokens can be used to access the Terraform Cloud API and perform all the actions your user account is entitled to. For more information, see the [user API tokens documentation](#).', and 'Treat these tokens like passwords, as they can be used to access your account without a username, password, or two-factor authentication.' Below this text is a purple button labeled 'Create an API token', with an orange arrow pointing to it. Further down, there is a section for 'Github App OAuth Token' with a blue button labeled 'Create a GitHub App token'. At the bottom of the page, there is a footer with the HashiCorp logo, copyright information '© 2023 HashiCorp, Inc.', and links for 'Support', 'Terms', 'Privacy', and 'Security'. A small 'Privacy - Terms' icon is also visible in the bottom right corner.

#0. 사전 준비 작업

2. Terraform Cloud 계정 생성 및 환경 준비



- Terraform Cloud 환경 준비 : 'ubucon-demo' 입력 후 , 'Generate token' 선택

A screenshot of the Terraform Cloud interface. The main page is titled 'Tokens' and is part of the 'Settings' section. A modal window titled 'Creating a user token' is open in the foreground. The modal has a 'Description' field with the text 'ubucon-demo' entered, highlighted by a blue border and an orange arrow. Below the description is an 'Expiration' dropdown menu set to '30 days', with a note that the token will expire on October 5th, 2023. At the bottom of the modal, there are two buttons: 'Generate token' (highlighted with an orange arrow) and 'Cancel'. The background shows the 'Tokens' page with a search bar, a user profile icon, and a sidebar with navigation options like 'User Settings', 'Profile', 'Sessions', 'Organizations', 'Password', 'Two Factor Authentication', 'SSO', and 'Tokens'. At the bottom of the page, there is a footer with copyright information for HashiCorp, Inc. and links for 'Support', 'Terms', 'Privacy', and 'Security'.

Settings / Tokens

Tokens

Creating a user token

Description Required
To help you identify this token later.

ubucon-demo

Expiration
30 days This token will expire October 5th, 2023

Generate token Cancel

Create a GitHub App token

Choose an organization

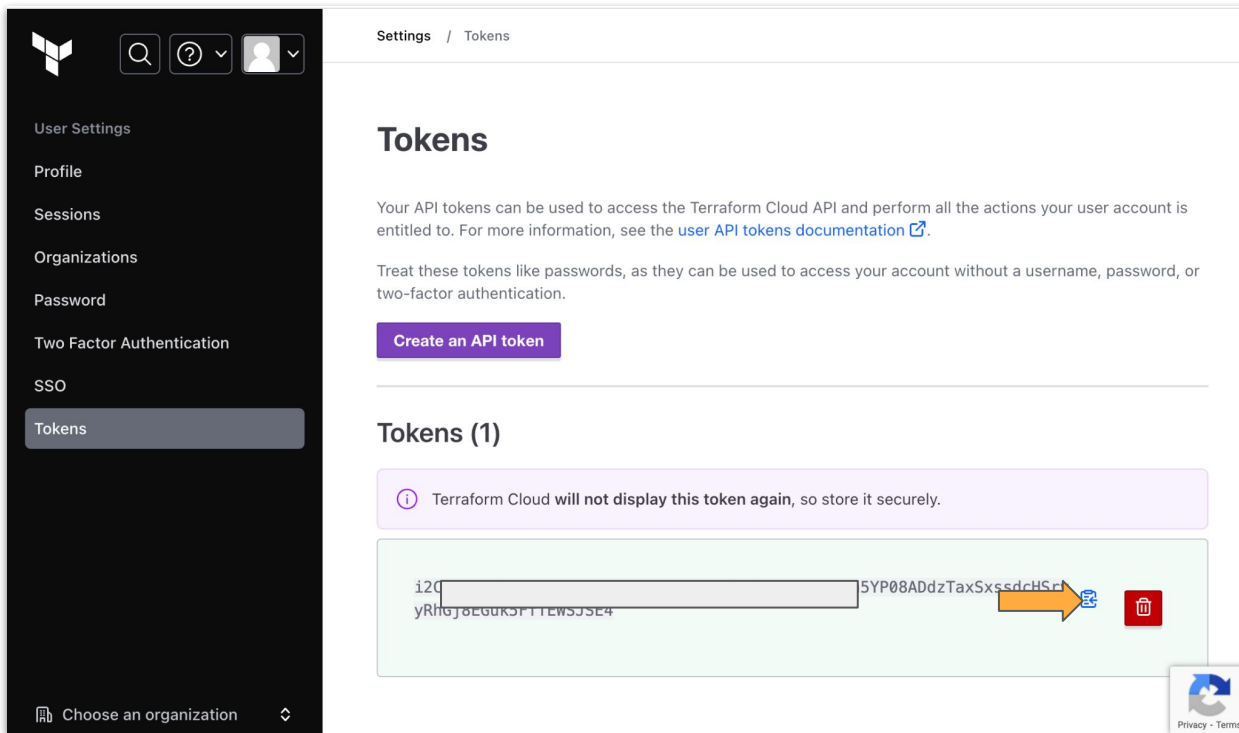
© 2023 HashiCorp, Inc. [Support](#) [Terms](#) [Privacy](#) [Security](#)

#0. 사전 준비 작업

2. Terraform Cloud 계정 생성 및 환경 준비



- Terraform Cloud 환경 준비 : 생성된 토큰 복사 후 별도 보관

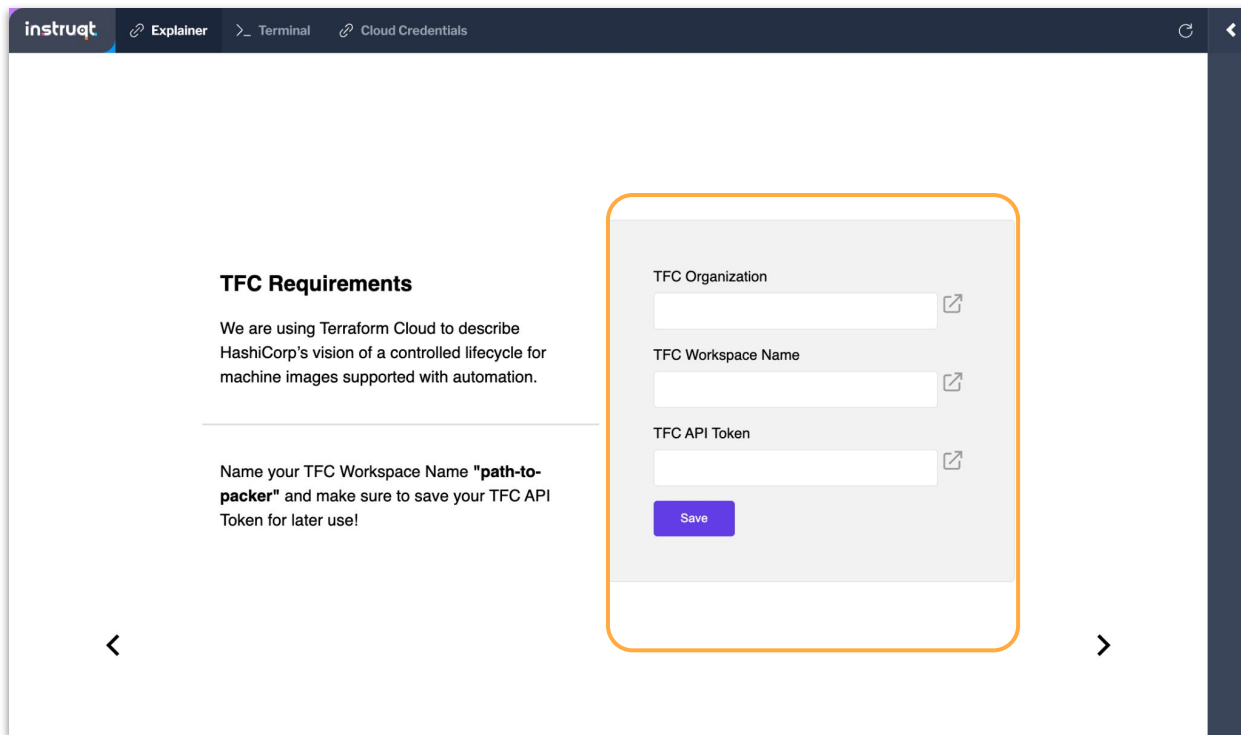
A screenshot of the Terraform Cloud user interface. On the left is a dark sidebar with navigation options: User Settings, Profile, Sessions, Organizations, Password, Two Factor Authentication, SSO, and Tokens (which is highlighted). The main content area is titled 'Settings / Tokens' and 'Tokens'. It contains a purple 'Create an API token' button. Below that, a light purple notification box states: 'Terraform Cloud will not display this token again, so store it securely.' Underneath, a green box displays a long alphanumeric token string: 'i2Q...5YP08ADdzTaxSxssdcH5r...' with a redacted portion in the middle. An orange arrow points to the right end of the token, and a red trash icon is to its right. At the bottom left of the page, there is a 'Choose an organization' dropdown menu, and at the bottom right, there is a 'Privacy - Terms' link.

#0. 사전 준비 작업

2. Terraform Cloud 계정 생성 및 환경 준비



- Terraform Cloud 환경 준비 완료

A screenshot of the 'instruct' web interface for Terraform Cloud setup. The interface has a dark blue header with the 'instruct' logo and navigation links for 'Explainer', 'Terminal', and 'Cloud Credentials'. The main content area is white and contains a section titled 'TFC Requirements'. Below the title, there is explanatory text: 'We are using Terraform Cloud to describe HashiCorp's vision of a controlled lifecycle for machine images supported with automation.' A horizontal line separates this text from a note: 'Name your TFC Workspace Name "path-to-packer" and make sure to save your TFC API Token for later use!'. To the right of the text is a form with three input fields: 'TFC Organization', 'TFC Workspace Name', and 'TFC API Token', each with a copy icon to its right. A purple 'Save' button is located below the 'TFC API Token' field. The entire form area is highlighted with an orange border. Navigation arrows are visible at the bottom of the interface.



1	Runbook을 패커 템플릿으로	<ul style="list-style-type: none">• 작업 절차가 정리된 Runbook을 패커 템플릿화• 이미지 생성 작업 시간 단축• 작업 표준화로 오류 최소화
2	골든 이미지 기반 배포 (AWS → Azure)	<ul style="list-style-type: none">• 골든 이미지를 AWS AMI로 저장• 저장된 이미지를 사용, AWS에 가상서버 배포• 이미지를 재사용, Azure로 서비스 확장
3	실습 환경 정리	<ul style="list-style-type: none">• CSP 내 가상 서버 삭제• 필요 시 HCP 내 환경 삭제 및 정리• Shadow IT 예방



1	Runbook을 패커 템플릿으로	<ul style="list-style-type: none">● 작업 절차가 정리된 Runbook을 패커 템플릿화● 이미지 생성 작업 시간 단축● 작업 표준화로 오류 최소화
2	골든 이미지 기반 배포 (AWS → Azure)	<ul style="list-style-type: none">● 골든 이미지를 AWS AMI로 저장● 저장된 이미지를 사용, AWS에 가상서버 배포● 이미지를 재사용, Azure로 서비스 확장
3	실습 환경 정리	<ul style="list-style-type: none">● CSP 내 가상 서버 삭제● 필요 시 HCP 내 환경 삭제 및 정리● Shadow IT 예방

#1. Runbook을 패커 템플릿으로



- 실습환경으로 복귀!

The screenshot shows a dark-themed interactive tutorial interface. At the top left, it says "Powered by instruct". In the top right, there are "Share" and "Progress" buttons. The main content features a 3D isometric illustration of a box labeled "HashiCorp Cloud Platform" with a blue square on top containing the HashiCorp logo. Below the box are icons for various cloud providers: Microsoft Azure, Google Cloud, and AWS. To the right of the illustration, the text reads "Path to Packer" in large white font, followed by "Explore the Path to Packer by using HCP Packer and Terraform Cloud" in a smaller white font. A vertical white line on the right side of the screen indicates the current position in the tutorial, with a right-pointing arrow next to it. At the bottom left is an "Exit" button, and at the bottom right is a green "Start" button. A small navigation bar with a play button and dots is visible at the very bottom center.

#1. Runbook을 패커 템플릿으로



- 실습 환경 준비 : 실습환경에 저장된 정보 입력 (HCP Org ID, Project ID, Client ID/Secret). **Tab**키로 이동

The screenshot displays the 'instruct' application interface. The main content area shows the 'HCP Requirements' section with a form for entering credentials. The form fields are:

- HCP Organization ID: 27801bbe-6cf1-4661-
- HCP Project ID: 35216085-3900-4d0f-
- HCP Client ID: 31T3NjoKRicjVF6p0c
- HCP Client Secret: M_x_03vtugAPEJUD

A green 'Save' button is located below the form. An orange box highlights the form fields, and an orange arrow points to the 'Save' button. Another orange arrow points to the right, indicating navigation. The sidebar on the right contains a 'Progress' indicator (49m) and a list of steps:

- Introduction
- #1 - Finding HCP Credentials
- #2 - Create TFC Workspace
- #3 - Validate HCP Credentials
- #4 - Validate TFC Credentials
- #5 - Validate AWS Credentials

At the bottom of the sidebar, there are 'Exit' and 'Check' buttons.

#1. Runbook을 패커 템플릿으로



- 실습 환경 준비 : 실습환경에 저장된 정보 입력 (Org Name, Workspace Name, Token). **Tab**키로 이동

The screenshot shows the 'instruct' application interface. The main content area displays 'TFC Requirements' with instructions on how to use Terraform Cloud. A form is overlaid on the screen, containing the following fields:

- TFC Organization: jsp-ubucon-org
- TFC Workspace Name: path-to-packer
- TFC API Token: lzCs9...ZcK0Hy9Q.atlasv1.aD15fQuVoU3gd0gYC

A blue 'Save' button is located below the form. The right sidebar shows a progress indicator and a list of steps: Introduction, #1 - Finding HCP Credentials, #2 - Create TFC Workspace, #3 - Validate HCP Credentials, #4 - Validate TFC Credentials, and #5 - Validate AWS Credentials. At the bottom, there are 'Exit' and 'Check' buttons.

#1. Run Book을 패커 템플릿으로

2. Terraform Cloud 계정 생성 및 환경 준



- 실습 환경 준비 : 설정된 정보 확인. #1 ~ #5까지 Task 수행 후 성공 시 'Check' 선택하여 다음 단계로 이동

The screenshot shows the Instruct terminal interface. On the left is a terminal window with the prompt `root@cloud-client:~#`. On the right is a task guide panel. The panel has a 'Progress' indicator at the top right showing '44m'. The main content is titled '#1 - Finding HCP Credentials' and includes instructions on how to find the HCP Organization ID and Project ID. A code block shows a URL template: `https://portal.cloud.hashicorp.com/orgs/XXXXXXXX-XXXX-XXXX-XXXXXXXXXX/projects/XXXXXXXX-XXXX-XXXX-XXXXXXXXXX`, with 'Organization ID' and 'Project ID' labels. Below the code block, there are instructions for finding the HCP client ID and secret, and a note for first-time users to create a free registry. At the bottom of the panel, there are 'Exit' and 'Check' buttons.

#1. Runbook을 패커 템플릿으로



- 실습 환경 준비 : #1 HCP 정보 확인, #2 Terraform Cloud 구성 확인 단계

The screenshot displays the 'instruct' application interface. On the left, a terminal window shows the prompt 'root@cloud-client:~#'. The top navigation bar includes 'Explainer', 'Terminal', and 'Cloud Credentials' tabs. On the right, a 'Progress' sidebar is visible, showing the current step: '#2 - Create TFC Workspace'. The sidebar contains three numbered instructions:

- 2.1 - Go into your Terraform Cloud account to create an **organization**. Name that organization anything you want.
For first time Terraform Users: choose "Start from scratch" when choosing your setup workflow.
- 2.2 - In your organization that you just created, create a new **workspace** and choose the **CLI-driven workflow**. Name your workspace `path-to-packer`.
- 2.3 - Once you have successfully created your workspace, go into your workspace **settings**. Click **General**. Under **Execution Mode**, make sure you are in **Local**.

Additional text in the sidebar includes: 'Make sure to save your settings before continuing on!', 'Now going back to the Explainer tab, input your organization and your workspace.', and 'To find your TFC API Token, follow this link'. At the bottom of the sidebar, there are 'Exit' and 'Check' buttons.

#1. Runbook을 패커 템플릿으로



- 실습 환경 준비 : #4 Terraform Cloud 정보 확인, Terminal 탭에서 아래와 같이 실행

The screenshot shows the 'instruct' terminal interface. The main terminal window displays the following commands and output:

```
root@cloud-client:~# curl -k -s \
--header "Authorization: Bearer $TFE_TOKEN" \
--header "Content-Type: application/vnd.api+json" \
--request GET $TFC_API_ACCT_DETAILS \
| jq -r '.data.attributes.email'
[redacted]@il.com
root@cloud-client:~# curl -k -s \
--header "Authorization: Bearer $TFE_TOKEN" \
--header "Content-Type: application/vnd.api+json" \
--request GET $TFC_API_SHOW_WORKSPACE \
| jq '.data.relationships.organization'
{
  "data": {
    "id": "jsp-ubucon-org",
    "type": "organizations"
  }
}
root@cloud-client:~#
```

The right sidebar shows the progress of the task:

- Progress: 1h, 56m
- Task: #4 - Validate TFC Credentials
- Instruction: Now, you can authenticate with TFC using your TFC TOKEN and confirm your identity:
- Code Block:

```
curl -k -s \
--header "Authorization: Bearer $TFE_TOKEN" \
--header "Content-Type: application/vnd.api+json" \
--request GET $TFC_API_ACCT_DETAILS \
| jq -r '.data.attributes.email'
```
- Instruction: Check the status of your intended TFC Workspace:
- Code Block:

```
curl -k -s \
--header "Authorization: Bearer $TFE_TOKEN" \
--header "Content-Type: application/vnd.api+json" \
--request GET $TFC_API_SHOW_WORKSPACE \
| jq '.data.relationships.organization'
```
- Task: #5 - Validate AWS Credentials
- Buttons: Exit, Check

#1. Runbook을 패커 템플릿으로



- 실습 환경 준비 : #5 AWS 정보 확인, Terminal 탭에서 아래와 같이 실행. 정상 종료 시 'Check' 선택

The screenshot displays the Instruct platform interface. On the left, a terminal window shows the following commands and outputs:

```
--header "Content-Type: application/vnd.api+json" \  
$TFC_API_SHOW_WORKSPACE \  
| jq '.data.relationships.organization' \  
{ \  
  "data": { \  
    "id": "jss-ubucon-org", \  
    "type": "organizations" \  
  } \  
} \  
root@cloud-client:~# aws sts get-caller-identity \  
{ \  
  "UserId": "AIDA6IEN6V7B3QUW3GQKM", \  
  "Account": "979550252995", \  
  "Arn": "arn:aws:iam::979550252995:user/jm0186mg57s9zq2t" \  
} \  
root@cloud-client:~# aws configure get profile.default.region \  
us-east-2 \  
root@cloud-client:~# aws ec2 describe-vpcs \  
{ \  
  "Vpcs": [ \  
    { \  
      "CidrBlock": "172.31.0.0/16", \  
      "DhcpOptionsId": "dopt-0723c4f9ce8589e91", \  
      "State": "available", \  
      "VpcId": "vpc-0d510197b53a9ad80", \  
      "OwnerId": "979550252995", \  
      "InstanceTenancy": "default", \  
      "CidrBlockAssociationSet": [ \  
        { \  
          "AssociationId": "vpc-cidr-assoc-0b98367e0d2c8a7f0", \  
          "CidrBlock": "172.31.0.0/16", \  
          "CidrBlockState": { \  
            "State": "associated" \  
          } \  
        } \  
      ], \  
      "IsDefault": true \  
    } \  
  ] \  
} \  
root@cloud-client:~#
```

On the right, the progress sidebar shows the current step: #5 - Validate AWS Credentials. Below the title, there is explanatory text: "Your AWS Credentials are part of the Instruct platform. To see what these credentials are, use the following commands." This is followed by three code blocks containing the commands: `aws sts get-caller-identity`, `aws configure get profile.default.region`, and `aws ec2 describe-vpcs`. At the bottom of the sidebar, there are two buttons: "Exit" and "Check". An orange arrow points to the "Check" button, indicating that it should be selected upon successful completion of the tasks.



#1. Runbook을 패커 템플릿으로

1. Converting your runbook to a Packer Template

- RunBook 확인 : 수작업으로 진행하는 AWS 인스턴스 배포 및 구성 작업 확인

The screenshot shows the Instruct interface with a dark theme. The main content area displays a runbook titled "Downloading and Configuring NGINX". The runbook includes a list of steps with terminal commands and instructions. An orange arrow points to the "Runbook" tab in the top navigation bar. The sidebar on the right contains a "Goals" section with three bullet points, a progress indicator for "1 - Converting your runbook to a Packer template", and a "Check" button at the bottom.

Downloading and Configuring NGINX

- In the new user directory, download NGINX
 - Run these commands. When complete, type in the command quit

```
sudo -s
nginx=stable # use nginx=development for latest development version
add-apt-repository ppa:nginx/nginx
apt update
apt install nginx
```
 - Verify the installation with `nginx -v`
 - It'll display the software version nginx version: `nginx/1.18.0` (Ubuntu)
- Enable and start the NGINX landing page
 - Start by checking the status where it will display `active(running)`

```
sudo systemctl status nginx
```

 - If NGINX is not running, run the following `sudo systemctl start nginx`
 - Load when the system starts

```
sudo systemctl enable nginx
```
 - Allow NGINX traffic and grant access to the firewall

```
sudo ufw app list
sudo ufw allow 'nginx full'
sudo ufw reload
```
- Deploying the NGINX Page
 - Open a new web browser with the IP address from the Connect instance page
 - It should look like this, where the localhost is the IP address

Goals:

- Turn Dunder Mifflin's runbook into a Packer template
- Adding code to the `ubuntu_base.pkr.hcl` template
- Understand how a Packer file is formatted and the mapping between the runbook and the code.

1 - Converting your runbook to a Packer template

Use the Dunder Mifflin runbook to build a Packer template and associate the new build in the HCP Packer Registry. This runbook can be found under the `Runbook` tab.

You will be given blocks of code to insert into the `ubuntu_base.pkr.hcl` file to create your Packer template.

NOTE: You will often see `var. {VARIABLE_NAME}`. This indicates that you are referencing a variable defined in the `variables.pkr.hcl` file.

2 - Defining your Data Source block

3 - Defining your Source block

Exit Check

#1. Runbook을 패커 템플릿으로

2. Defining your Data Source Block



- ubuntu_base.pkr.hcl 수정 : 'Packer' 탭에서 수행. Data Source 블록 추가

The screenshot shows the Instruct IDE interface. The main editor displays the Packer configuration file `ubuntu_base.pkr.hcl`. The code is as follows:

```
1 packer {
2   required_plugins {
3     amazon = {
4       version = ">= 1.0.1"
5       source  = "github.com/hashicorp/amazon"
6     }
7   }
8 }
9
10 data "amazon-ami" "ubuntu-server-east" {
11   region = var.region
12   filters = {
13     name           = var.image_name
14     root-device-type = "ebs"
15     virtualization-type = "hvm"
16   }
17   most_recent = true
18   owners      = ["099720109477"]
19 }
20
21 build {
22 }
23 }
```

The code block starting at line 9 is highlighted with a yellow box. The sidebar on the right shows a progress bar and the following text:

2 - Defining your Data Source block

Under the `packer` tab, edit the template file `ubuntu_base.pkr.hcl` and include the following code stanza (start on line 9).

```
data "amazon-ami" "ubuntu-server-east" {
  region = var.region
  filters = {
    name           = var.image_name
    root-device-type = "ebs"
    virtualization-type = "hvm"
  }
  most_recent = true
  owners      = ["099720109477"]
}
```

A data block will request Packer to read from a given data source (`"amazon-ami"`) and export the results under the given local name (`"ubuntu-server-east"`).

Adding this chunk of code chooses the region you wish to set up your instance in.

3 - Define your Source block

Buttons for `Exit` and `Check` are visible at the bottom of the sidebar.

#1. Runbook을 패커 템플릿으로

3. Defining your Source Block



- ubuntu_base.pkr.hcl 수정 : Source 블록 추가

The screenshot shows the instruct IDE interface. The main editor displays the Packer configuration file `ubuntu_base.pkr.hcl`. The configuration includes a `packer` block with `required_plugins` for Amazon, and a `data` block for an Amazon AMI. A new `source` block for "amazon-ecs" is being added, highlighted with a yellow box. The sidebar on the right shows the runbook progress, with the current step being "3 - Define your Source block". The runbook text explains that this block defines the builder type and the unique name for the source.

```
1 packer {
2   required_plugins {
3     amazon = {
4       version = ">= 1.0.1"
5       source  = "github.com/hashicorp/amazon"
6     }
7   }
8 }
9 data "amazon-ami" "ubuntu-server-east" {
10  region = var.region
11  filters = {
12    name           = var.image_name
13    root-device-type = "ebs"
14    virtualization-type = "hvm"
15  }
16  most_recent = true
17  owners      = ["099720109477"]
18 }
19 source "amazon-ecs" "ubuntu-server-east" {
20  region           = var.region
21  source_ami       = data.amazon-ami.ubuntu-server-east.id
22  instance_type    = "t2.small"
23  ssh_username     = "ubuntu"
24  ssh_agent_auth   = false
25  ami_name         = "path-to-packer{{timestamp}}_v${var.version}"
26  tags             = var.aws_tags
27 }
28 build {
29 }
30 }
31
```

2 - Defining your Data Source block

3 - Define your Source block

In the same template file, `ubuntu_base.pkr.hcl`, insert the following code snippet on line 19:

```
source "amazon-ecs" "ubuntu-server-east" {
  region           = var.region
  source_ami       = data.amazon-ami.ubuntu-server-east.id
  instance_type    = "t2.small"
  ssh_username     = "ubuntu"
  ssh_agent_auth   = false
  ami_name         = "path-to-packer{{timestamp}}_v${var.version}"
  tags             = var.aws_tags
}
```

This block of code defines your builder type (`"amazon-ecs"`) and the unique name or identifier you want to give the source (`"ubuntu-server-east"`).

Looking at the `runbook` tab, adding this block of code correlates to **Create a new instance by using the Amazon EC2** under **Creating an Ubuntu Server in AWS**. You can now tell that this block applies the

Exit Check

#1. Runbook을 패커 템플릿으로

4. Add your build Block



- 4.1 Associate your build with the HCP Packer Registry :

The screenshot shows the instruct IDE interface. The main editor displays a Packer build file named `ubuntu_base.pkr.hcl`. An orange arrow points to the `</> Packer` tab. The code in the editor includes a `data` block for an Amazon AMI and a `source` block for an EBS volume. A yellow box highlights the `build` block, which contains the `hcp_packer_registry` configuration. The progress panel on the right shows the current step: "4.1 Associate your build with the HCP Packer Registry." Below this, it provides instructions and a code snippet for the `hcp_packer_registry` block.

```
5     source = "github.com/hashicorp/amazon"
6   }
7 }
8 }
9 data "amazon-ami" "ubuntu-server-east" {
10   region = var.region
11   filters = {
12     name      = var.image_name
13     root-device-type = "ebs"
14     virtualization-type = "hvm"
15   }
16   most_recent = true
17   owners      = ["099720109477"]
18 }
19 source "amazon-ebs" "ubuntu-server-east" {
20   region      = var.region
21   source_ami  = data.amazon-ami.ubuntu-server-east.id
22   instance_type = "t2.small"
23   ssh_username = "ubuntu"
24   ssh_agent_auth = false
25   ami_name     = "path-to-packer{{timestamp}}_v${var.version}"
26   tags        = var.aws_tags
27 }
28 build {
29   hcp_packer_registry {
30     bucket_name = "path-to-packer-aws"
31     description = "Path to Packer Demo"
32     bucket_labels = var.aws_tags
33     build_labels = {
34       "build-time" = timestamp(),
35       "build-source" = basename(path.cwd)
36     }
37   }
38 }
```

3 - Define your Source block

4 - Add to your build block

4.1 Associate your build with the HCP Packer Registry.
You can do so by adding this block of code to the build block:

```
hcp_packer_registry {
  bucket_name = "path-to-packer-aws"
  description = "Path to Packer Demo"
  bucket_labels = var.aws_tags
  build_labels = {
    "build-time" = timestamp(),
    "build-source" = basename(path.cwd)
  }
}
```

The `bucket_name` will help you identify the image that shows up in your HCP Packer Registry.

By adding the `hcp_packer_registry` block, you are defining the image you would like to store in your HCP Packer account.

Exit Check

#1. Runbook을 패커 템플릿으로

4. Add your build Block



- 4.2 Continue adding to the 'build' block:

The screenshot shows the Instruct IDE interface. The main editor displays a Packer build file named `ubuntu_base.pkr.hcl`. The code includes a `data` block for an Amazon AMI, a `source` block for an EBS volume, and a `build` block. The `build` block contains an `hcp_packer_registry` block and a `sources` list. An orange arrow points to the `</> Packer` tab, and an orange box highlights the `sources` list in the `build` block.

```
8 }
9 data "amazon-ami" "ubuntu-server-east" {
10   region = var.region
11   filters = {
12     name           = var.image_name
13     root-device-type = "ebs"
14     virtualization-type = "hvm"
15   }
16   most_recent = true
17   owners      = ["099720109477"]
18 }
19 source "amazon-ebs" "ubuntu-server-east" {
20   region           = var.region
21   source_ami       = data.amazon-ami.ubuntu-server-east.id
22   instance_type    = "t2.small"
23   ssh_username     = "ubuntu"
24   ssh_agent_auth   = false
25   ami_name         = "path-to-packer{{timestamp}}_v${var.version}"
26   tags             = var.aws_tags
27 }
28 build {
29   hcp_packer_registry {
30     bucket_name = "path-to-packer-aws"
31     description = "Path to Packer Demo"
32     bucket_labels = var.aws_tags
33     build_labels = {
34       "build-time" = timestamp(),
35       "build-source" = basename(path.cwd)
36     }
37   }
38   sources = [
39     "source.amazon-ebs.ubuntu-server-east"
40 ]
```

By adding the `hcp_packer_registry` block, you are defining the image you would like to store in your HCP Packer account.

4.2 Continue adding to the `build` block by inserting this snippet of code:

```
sources = [
  "source.amazon-ebs.ubuntu-server-east"
]
```

This tells Packer to build the instance you have defined in your `source` block you created in #3

4.3 Add your provisioners (continuing in the `build` block):

A `file` provisioner uploads files to the machine built by Packer, then the `shell` provisioner will move those files to the proper place, set permissions, etc.

```
# Add startup script that will run path to p
path-to-packer-aws #612" {
```

Exit Check

#1. Runbook을 패커 템플릿으로

4. Add your build Block



- 4.3 Add your provisioner : "디스켓"모양 클릭하여 저장 후, "Check" 선택

The screenshot shows the instruct IDE interface. The main editor displays a Packer build file with the following content:

```
32 bucket_labels = var.aws_tags
33 build_labels = {
34   "build-time" = timestamp(),
35   "build-source" = basename(path.cwd)
36 }
37 }
38 sources = [
39   "source.amazon-eks.ubuntu-server-east"
40 ]
41
42 # Add startup script that will run path to packer on instance build
43 provisioner "file" {
44   source = "../production/setup-deps-path-to-packer.sh"
45   destination = "/tmp/setup-deps-path-to-packer.sh"
46 }
47
48 # Move temp files to actual destination
49 # Must use this method because their destinations are protected
50 provisioner "shell" {
51   inline = [
52     "sudo cp /tmp/setup-deps-path-to-packer.sh /var/lib/cloud/seed/
53   ]
54 }
55
56 provisioner "shell" {
57   inline = [
58     "sleep 30",
59     "sudo apt-get update",
60     "sudo apt-get upgrade -y",
61     "sudo apt-get install -y nginx",
62     "sudo systemctl start nginx",
63     "sudo apt-get install tree"
64 ]
65 }
```

On the right side, a panel titled "Progress" shows a progress bar and a timer for 1h, 36m. Below the progress bar, the text reads: "4.3 Add your provisioners (continuing in the build block):". Below this, there is a description: "A file provisioner uploads files to the machine built by Packer, then the shell provisioner will move those files to the proper place, set permissions, etc." Below the description, there is a code block with the following content:

```
# Add startup script that will run path to packer on instance build
provisioner "file" {
  source = "../production/setup-deps-path-to-packer.sh"
  destination = "/tmp/setup-deps-path-to-packer.sh"
}

# Move temp files to actual destination
# Must use this method because their destinations are protected
provisioner "shell" {
  inline = [
    "sudo cp /tmp/setup-deps-path-to-packer.sh /var/lib/cloud/seed/
  ]
}

provisioner "shell" {
  inline = [
    "sleep 30",
    "sudo apt-get update",
    "sudo apt-get upgrade -y",
    "sudo apt-get install -y nginx",
    "sudo systemctl start nginx",
    "sudo apt-get install tree",
  ]
}
```

At the bottom right of the IDE, there are two buttons: "Exit" and "Check". A yellow arrow points to the "Check" button.



1	Runbook을 패커 템플릿으로	<ul style="list-style-type: none">• 작업 절차가 정리된 Runbook을 패커 템플릿화• 이미지 생성 작업 시간 단축• 작업 표준화로 오류 최소화
2	골든 이미지 기반 배포 (AWS → Azure)	<ul style="list-style-type: none">• 골든 이미지를 AWS AMI로 저장• 저장된 이미지를 사용, AWS에 가상서버 배포• 이미지를 재사용, Azure로 서비스 확장
3	실습 환경 정리	<ul style="list-style-type: none">• CSP 내 가상 서버 삭제• 필요 시 HCP 내 환경 삭제 및 정리• Shadow IT 예방

#2. 골든 이미지 기반 배포 (AWS → Azure)




-

Powered by **instruct**

[Share](#) [Progress](#)

Now it is time to deploy images using HashiCorp Terraform from our HCP Packer registry! Let's create snapshots with our golden image.

A photograph of a woman wearing a white wedding veil, looking slightly to the side with a serious expression. She is pointing her index fingers upwards with both hands. The background is dimly lit, showing what appears to be a crib or bed with some items on it.

[Exit](#) [Start](#)



#2. 골든 이미지 기반 배포 (AWS)

1. Add your build Block

- #1 ~ #6까지 수행. #1 이미지 생성 시 약 4분 30초 ~ 5분 소요

Deploying an Amazon EC2 instance with Terraform

Once you have stored your golden image in HCP Packer and built your Amazon Machine Image (AMI) with Packer, you can use HashiCorp Terraform to deploy an Amazon EC2 instance. Using HCP Packer and Terraform in conjunction allows you to automate the process of deploying your image.

```
data "hcp_packer_iteration" "ubuntu" {
  bucket_name = var_hcp_bucket_ubuntu
  channel     = var_hcp_channel
}

data "hcp_packer_image" "ubuntu" {
  bucket_name = data.hcp_packer_iteration.ubuntu.bucket_name
  iteration_id = data.hcp_packer_iteration.ubuntu.id
  cloud_provider = "aws"
  region = var_region
}
```

Goals:

- Build an Amazon Machine Image (AMI) using Packer
- Recognize that without Packer, cloud practitioners build, track, and maintain their machine images manually
- With Packer and HCP Packer, cloud practitioners use a central registry to build, track and maintain the lifecycle of their machine images with automation

- 1 - Build a machine image with Packer
- 2 - Create an HCP Packer Channel
- 3 - Edit your TFC Organization and Workspace names
- 4 - Deploy with the AWS EC2 CLI
- 5 - Automate with HCP Packer
- 6 - Deploy a new machine image using Terraform

Machine images can be easily updated and released to react to emerging business, technology or security conditions.

#2. 골든 이미지 기반 배포 (Azure)

1. Add your build Block

- #1 ~ #5까지 수행

The screenshot shows the Instruct IDE interface. The main editor area displays the text "AWS vs. Azure" and explains that while images can be deployed in both AWS and Azure, variable names and syntax differ. It instructs the user to use the AWS Packer template as a guide to create an Azure Packer template. A progress bar on the right indicates the task is 1h, 35m long. The right sidebar shows a table of contents with five steps: #1 - Convert from AWS to Azure, #2 - Create and connect to your Azure account, #3 - Edit your Terraform and Packer files, #4 - Build your Packer image on Azure, and #5 - Use Terraform to deploy your image in Azure. The "Check" button is highlighted in green.

instruct Explorer Packer Terraform Terminal Credentials

Progress 1h, 35m

Introduction

Goals:

- Convert your Packer template used to deploy in AWS to a template you can deploy in Azure. You must do so because variable names and syntax differ between the two cloud providers.
- Build your image from the Azure Packer template
- Deploy your image using TFC in Azure

#1 - Convert from AWS to Azure

#2 - Create and connect to your Azure account

#3 - Edit your Terraform and Packer files

#4 - Build your Packer image on Azure

#5 - Use Terraform to deploy your image in Azure

Exit Check



1	Runbook을 패커 템플릿으로	<ul style="list-style-type: none">• 작업 절차가 정리된 Runbook을 패커 템플릿화• 이미지 생성 작업 시간 단축• 작업 표준화로 오류 최소화
2	골든 이미지 기반 배포 (AWS → Azure)	<ul style="list-style-type: none">• 골든 이미지를 AWS AMI로 저장• 저장된 이미지를 사용, AWS에 가상서버 배포• 이미지를 재사용, Azure로 서비스 확장
3	실습 환경 정리	<ul style="list-style-type: none">• CSP 내 가상 서버 삭제• 필요 시 HCP 내 환경 삭제 및 정리• Shadow IT 예방

#3. 실습 환경 정리




-

Powered by **instruat**

[Share](#) [Progress](#)

Once you are done using your image, you need to clean up your environment and destroy your infrastructure.

A photograph showing a person in a grey uniform using a mop to clean a spill on a grey floor. A blue bucket and a yellow caution sign are visible in the foreground.

Exit

Start

#3. 실습 환경 정리



- #1 ~ #8까지 수행.

The screenshot shows the Instruct interface. On the left is a terminal window with the prompt `root@cloud-client:~#`. On the right is a sidebar with a progress bar at the top showing "1h, 20m". The sidebar contains two sections:

- Introduction**: A text block explaining the cleanup steps, followed by a bulleted list:
 - Remove your HCP Bucket and associated Iterations and Channels
 - Remove your TFC Workspace
 - Remove any EC2 instances and AMIs under a specific account (be careful here)
 - Remove any instances in Azure
- Step by step**: A section with two steps:
 - STEP 1: Destroy the infrastructure you have built in AWS**: A code block containing the commands:

```
cd $ASSETS_HOME/terraform/production
terraform destroy -auto-approve
```
 - STEP 2: Type the following command to find the**

At the bottom of the sidebar are two buttons: "Exit" and "Next".



- 클라우드 환경, 가상 서버 이미지 (Ubuntu Image) 관리 방안 실습

