



AWS Builders Korea

EC2 Image Builder / CloudWatch Dashboard
Hands-on Lab

October 5, 2022

1. EC2 Image Builder 실습 개요

이번 실습은 AWS workshop studio(EC2 Image Builder Workshop)의 내용을 일부 수정해서 만든 문서입니다. 원문 워크샵은 웹으로도 접근이 가능하니 참조 바랍니다.

<https://ec2-image-builder.workshop.aws/>

EC2 Image Builder 는 스크립트를 작성하지 않고도 OS 이미지를 쉽게 구축하고, 커스터마이징 및 배포를 할 수 있는 **완전 관리형 서비스** 입니다. 이미지 파이프라인을 통해 OS 이미지를 커스터마이징 하는 프로세스를 정의할 수 있습니다. 이미지 레시피, 인프라 구성, 배포 및 테스트 설정으로 구성 됩니다.

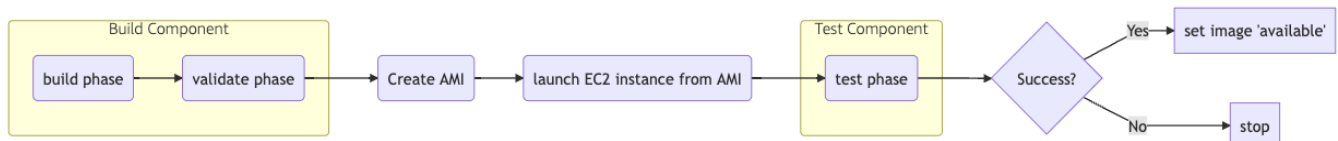
(1-1) 실습 목적

이 워크샵을 통해 EC2 Image Builder 의 기본 개념과 주요 구성 요소들을 이해하고 Amazon EC2 관리에 활용할 수 있는 유용한 도구들을 직접 테스트 해 봅니다.

(1-2) 다루는 서비스

- Amazon Identity and Access Management (IAM)
- Amazon EC2 Image Builder
- Amazon EC2
- Amazon CloudWatch

(1-3) 실습 아키텍처



(1-4) 실습 범위

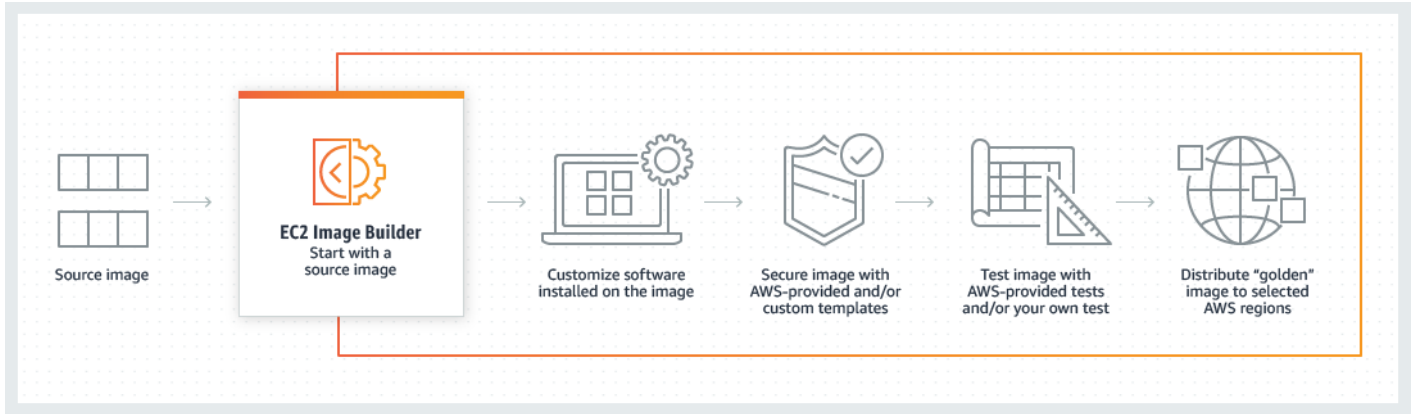
- **Phase 1** : Image Pipeline 생성
- **Phase 1** : Image recipe 선택 및 Components 추가
- **Phase 1** : Infrastructure 및 Distribution 구성
- **Phase 1** : Image Pipeline 실행
- **Phase 1** : [옵션] Component Document 확인 / CloudWatch 에서 로그 확인
- **Phase 2** : 사용자 지정 Components 를 사용하여 LAMP stack AMI 생성
- **Phase 2** : 생성된 AMI 로 EC2 instances launch
- **Phase 2** : 실행 중인 EC2 instance metrics 로 CloudWatch Dashboard 생성

(1-5) Amazon EC2 와 Amazon EC2 Image Builder

EC2 Image Builder 는 AWS 또는 온프레미스에서 사용하기 위해 가상 머신 및 컨테이너 이미지의 구축, 테스트 및 배포를 간소화 합니다. EC2 Image Builder 는 간단한 UI, 기본 자동화 및 AWS 제공 보안 설정을 통해 이미지를 최신 상태로 유지하고 보안을 강화하는 노력을 크게 줄이고 있습니다. EC2 Image Builder 를 사용하면 EC2 instance 를 launching 하기 위한 이미지를 업데이트하기 위해서 수동 작업 단계가 필요하지 않으며 자체적인 자동화 파이프라인을 구축하지 않아도 됩니다.

EC2 Image Builder 는 이미지를 생성, 저장 및 공유하는 데 사용되는 기본 AWS 리소스 비용 외에는 모두 무료로 제공됩니다.

EC2 Image Builder 로 AWS 콘솔에서 직관적으로 자동화된 파이프라인을 생성하여 AWS 및 온프레미스에서 사용할 수 있는 호환 Linux 및 Windows Server 이미지를 만들 수 있습니다. 소프트웨어 업데이트가 제공되면 이미지 빌더는 자동으로 새 이미지를 생성하고 테스트를 실행한 후 예약된 AWS 리전으로 배포 됩니다.



실습은 가장 먼저 AWS 콘솔에서 EC2 Image Builder 를 통해 Windows Server 2019 이미지를 생성하는 이미지 파이프라인을 만들 것 입니다. 이때 파이프라인에서는 Powershell 과 같은 build component 와 AMI 생성 후 이미지를 테스트 하기 위한 Test Component 를 추가하여 파이프라인을 구성하여 실행하고 로그를 확인해 볼 것 입니다.

그리고 사용자 정의 Component 로 Linux 이미지를 만드는 이미지 파이프라인을 새롭게 만들고, 여기서 생성된 AMI 로 EC2 instance 를 실행한 뒤, CloudWatch 로 모니터링하는 것 까지가 이번 HOL 의 범위 입니다.

2. 실습 환경 구성

이미 AWS 계정을 가지고 있다면 바로 이 실습의 가이드를 따라 진행할 수 있으나, 계정이 없다면 먼저 AWS 계정을 만들어야 합니다. 만약 임시 계정을 받으셨다면 (2-2) AWS 이벤트용 섹션을 참고하시기 바랍니다. AWS 계정 및 활성화하기 위해서는 [링크](#)를 참조하시기 바랍니다.

- 실습은 us-east-1 (N.Virginia)에서 진행하며, ap-northeast-2(Seoul) 등 EC2 Image Builder 가 지원되는 다른 리전에서도 가능 합니다.

(2-1) IAM 사용자

AWS 계정을 생성했지만 직접 IAM 사용자를 생성하지 않은 경우, IAM 콘솔을 사용하여 IAM 사용자를 생성할 수 있습니다. 다음 스텝에 따라 Administrator 를 생성 합니다. 이미 관리자 사용자가 있다면, 다음 IAM 사용자 생성 작업을 건너뛵니다.

1. AWS 계정 이메일 주소와 비밀번호를 사용하여 AWS 계정의 **Root** 사용자로 [IAM](#)에 로그인 합니다.
2. IAM 콘솔 왼쪽 메뉴 패널에서 **Users** 를 선택한 다음 **Add user** 를 클릭합니다.
3. **User name** 은 **Administrator** 로 입력합니다.
4. **AWS Management Console access** 체크박스를 선택하고, **Custom password** 를 선택한 다음 비밀번호를 입력합니다. **Require password reset** 체크박스는 해제 합니다.

5. Next: Permissions 을 클릭합니다.

Add user 1 2 3 4 5

Set user details

You can add multiple users at once with the same access type and permissions. [Learn more](#)

User name* Administrator

[Add another user](#)

Select AWS access type

Select how these users will primarily access AWS. If you choose only programmatic access, it does NOT prevent users from accessing the console using an assumed role. Access keys and autogenerated passwords are provided in the last step. [Learn more](#)

Select AWS credential type* ☐ Access key - Programmatic access
Enables an access key ID and secret access key for the AWS API, CLI, SDK, and other development tools.

☒ Password - AWS Management Console access
Enables a password that allows users to sign-in to the AWS Management Console.

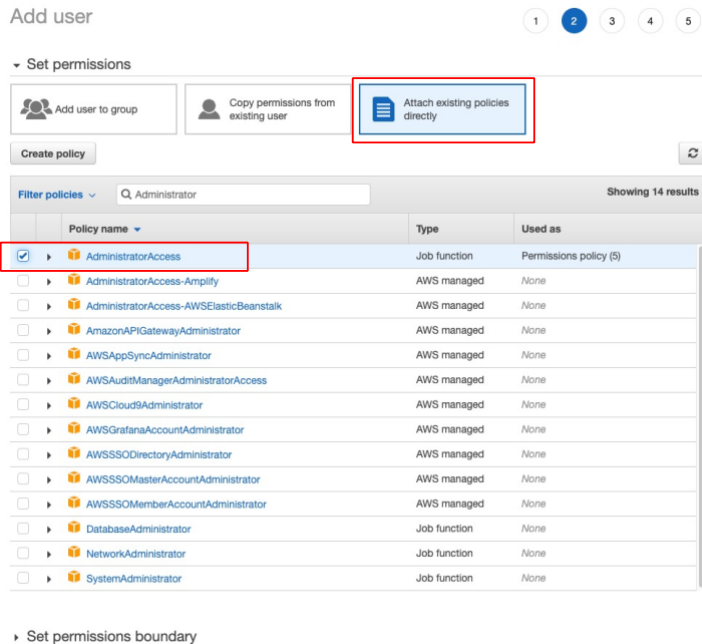
Console password* ☐ Autogenerated password
☒ Custom password

☐ Show password

Require password reset ☐ User must create a new password at next sign-in
Users automatically get the [IAMUserChangePassword](#) policy to allow them to change their own password.

* Required Cancel Next: Permissions

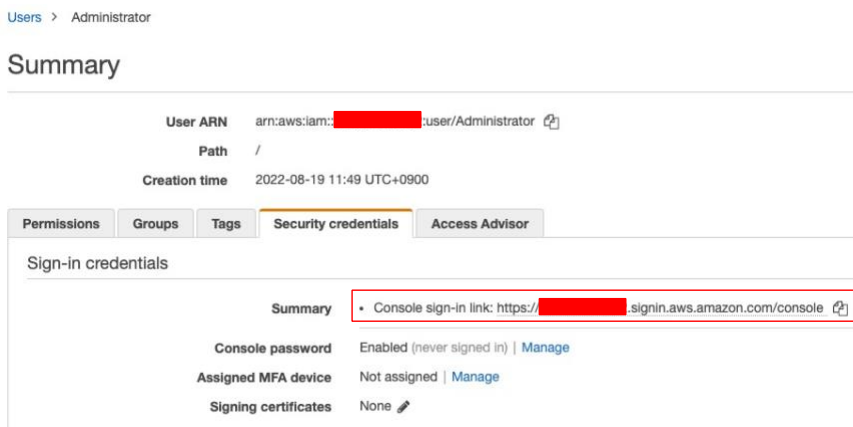
6. **Attach existing policies directly** 를 선택하고 **AdministratorAccess** 정책에 체크박스를 선택하고 Next: Tags 를 클릭합니다.



7. **Review** 화면에서 다시 확인 후 **Create user** 버튼을 클릭 합니다.

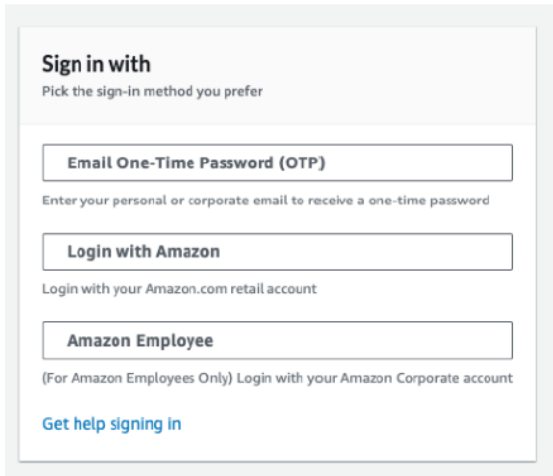
8. 아래와 같이 직접 만든 User 를 클릭 후 Summary > Security credentials 탭을 보면 직접 접근할 수 있는 URL 이 보입니다. Root 사용자를 로그아웃하고 새로 생성한 Administrator 사용자로 로그인 합니다.

(IAM user name: Administrator, Password: 직접 설정한 패스워드)



(2-2) AWS 이벤트용 계정

1. 임시 계정을 요청하셔서 링크를 받으신 분들은 접속 시 아래와 같은 화면이 보입니다. 여기서 **Email One-Time Password (OTP)**를 클릭합니다.



Sign in with
Pick the sign-in method you prefer

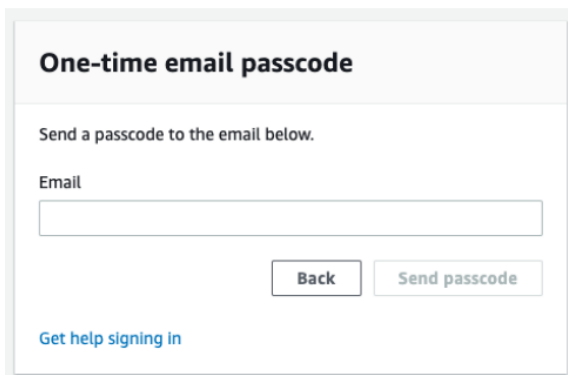
Email One-Time Password (OTP)
Enter your personal or corporate email to receive a one-time password

Login with Amazon
Login with your Amazon.com retail account

Amazon Employee
(For Amazon Employees Only) Login with your Amazon Corporate account

[Get help signing in](#)

2. 아래 화면이 나오면 본인의 이메일 계정을 입력하시고 **Send passcode** 버튼을 누릅니다.



One-time email passcode

Send a passcode to the email below.

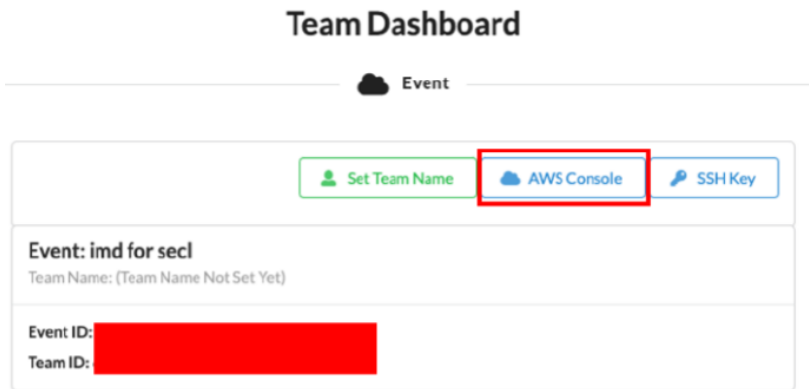
Email

[Back](#) [Send passcode](#)

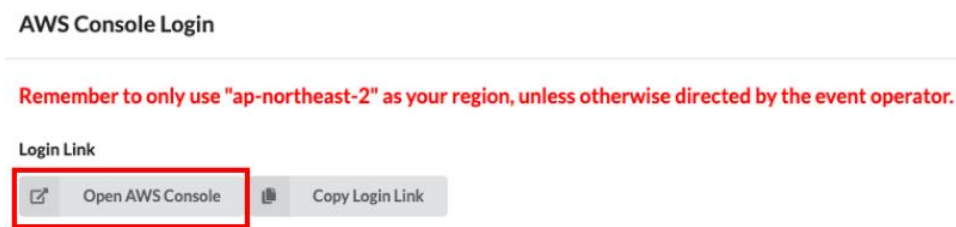
[Get help signing in](#)

3. 이메일을 통해 passcode 를 받으면 입력 후 **Sign in** 을 클릭합니다.

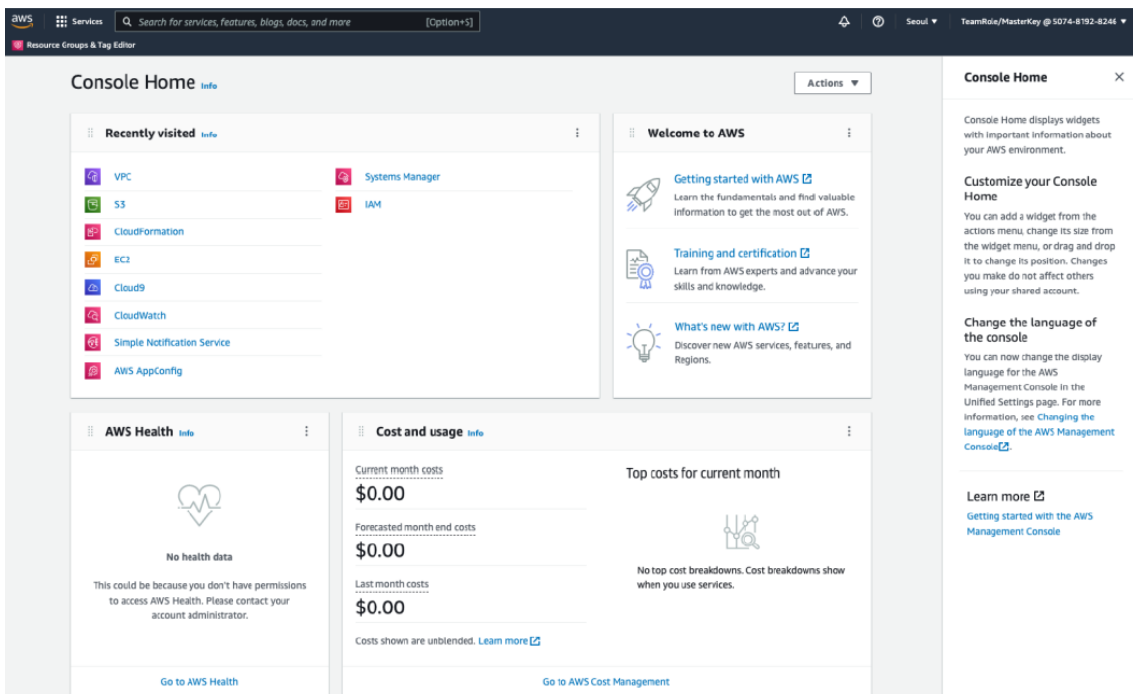
4. 이후 아래 화면이 보이면 **AWS Console** 을 클릭합니다.



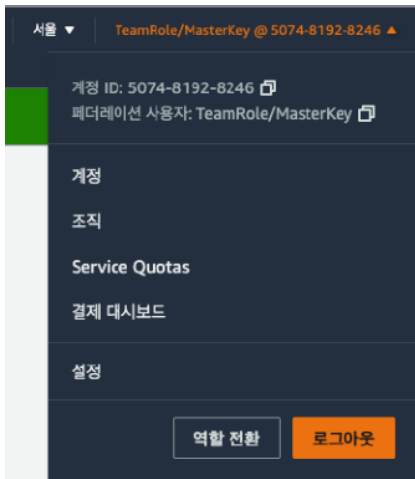
5. 이후 아래 화면에서는 Open AWS Console 을 클릭합니다.



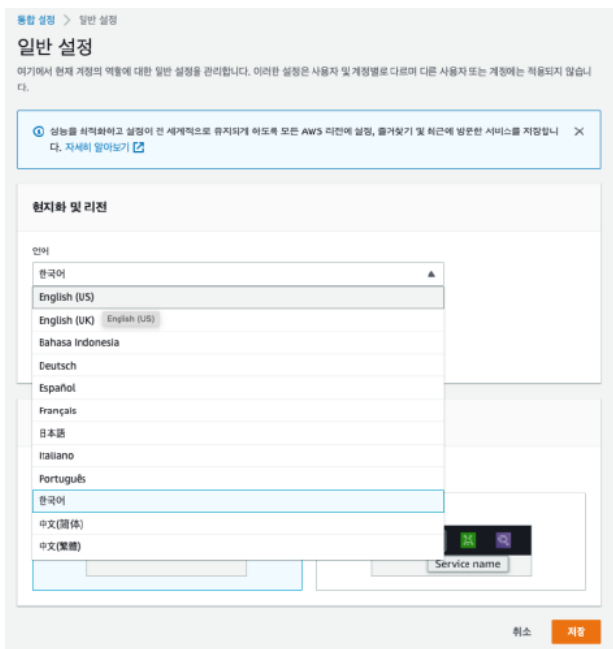
6. 아래와 같이 AWS Console 초기 화면이 보입니다.



7. 실습은 영문으로 진행되므로 필요에 따라 언어 설정을 변경합니다.
8. 기본 설정이 한글로 되어 있다면 우측 상단 계정을 클릭 후 설정을 클릭합니다.



9. 이후 언어 설정에서 영어를 선택하신 후 저장 버튼을 클릭합니다.



10. 이제 설정이 마무리 되었고 이후 단계부터 본격적으로 실습을 시작합니다.

3. EC2 Instance Profile

EC2 Image Builder 는 EC2 Instance 를 사용하여 AMI 를 생성하므로 EC2 Instance Profile 을 사용해야 합니다. Role 을 사용할 때 장기 자격 증명(예: user name 과 password 또는 access key)을 EC2 Instance 에 배포할 필요가 없습니다. 대신 Role 은 애플리케이션이 다른 AWS 리소스를 호출할 때 사용할 수 있는 임시 권한을 제공 합니다.

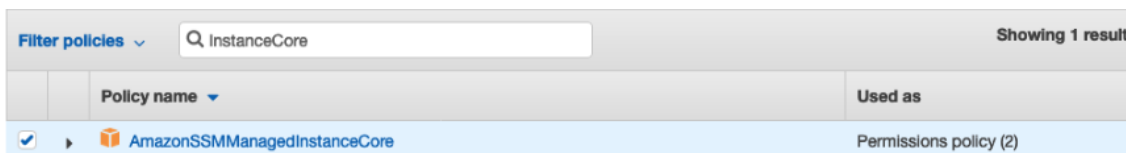
인스턴스 프로파일과 연결하는 IAM 역할에는 이미지에 포함된 Build 및 Test Component 를 실행할 수 있는 권한이 있어야 합니다. 예를 들어 s3 에서 파일을 다운로드 해야 하는 경우 Instance Profile 에 다운로드 권한을 부여해야 합니다.

따라서 가정 먼저 해야 할 작업은 **EC2InstanceProfileForImageBuilder** 와

AmazonSSMManagedInstanceCore 라는 IAM Role Policy 를 사용하도록 Role 을 만듭니다.

(3-1) Create the role

1. 이전 단계에서 만든 **Administrator** 사용자로 [IAM](#) 에 로그인 합니다.
2. IAM 콘솔 왼쪽 메뉴 패널에서 **Roles** 를 선택한 다음 **Create Role** 을 클릭 합니다.
3. Select trusted entity 에서 '**AWS service**'를 선택하고, 아래 Use case 에서는 '**EC2**' 를 선택하고 Next 를 클릭 합니다.
4. Attach permissions policies 화면에서 검색창에서 **AmazonSSMManagedInstanceCore** 를 검색하고 선택 합니다. 그리고 다시 검색창에서 **EC2InstanceProfileForImageBuilder** 를 검색하고 선택 합니다. (총 2 개의 Policy 를 선택 하였습니다) 그리고 Next 를 클릭 합니다.

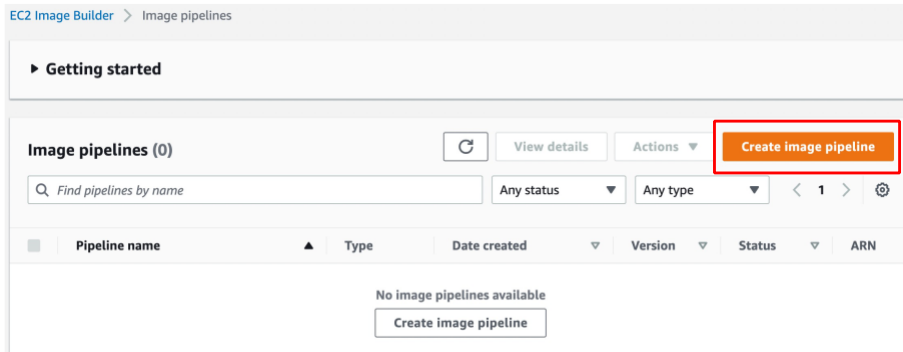


5. Review 화면에서 **Role Name** 에 **TeamRole-builder** 이라고 입력 합니다.
그리고 맨 아래 **Create role** 버튼을 클릭 합니다.

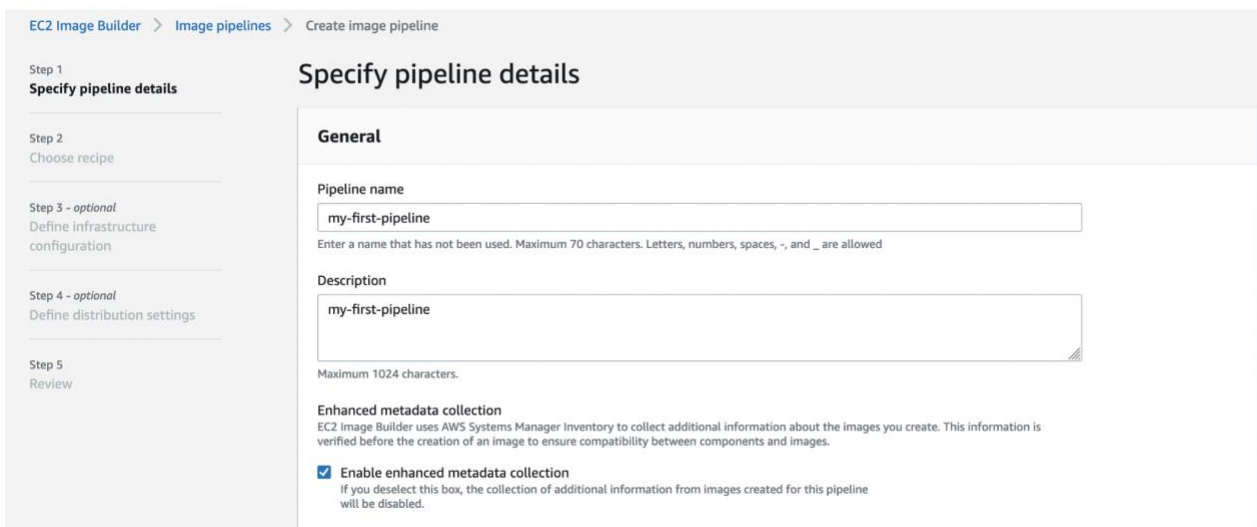
4. Create EC2 Image Builder Pipeline #1

(4-1) Creating an Image Pipeline

1. [EC2 Image Builder Console](#) 로 이동 합니다.
2. **Create image pipeline** 버튼을 클릭 합니다.



3. General 섹션에서 Pipeline name 을 입력 합니다. 여기서는 **my-first-pipeline** 이라고 입력합니다. 아래에 Enhanced metadata collection 부분은 기본적으로 설정되어 있습니다. AWS Systems Manager 인벤토리를 사용하여 생성한 이미지에 대한 추가 정보를 수집합니다. 이 정보는 구성 요소와 기본 이미지 간의 호환성을 보장하기 위해 이미지를 생성하기 전에 확인됩니다. 그리고 아래에 Build schedule 섹션에서 이미지를 빌드한 간격을 구성할 수 있습니다. 여기서는 기본 일정을 유지합니다.

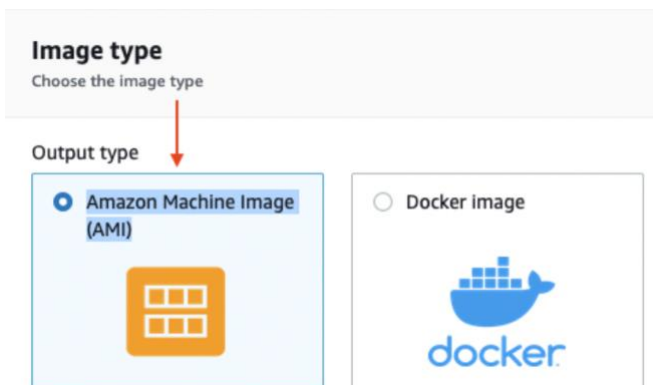


4. 페이지 맨 아래 **Next** 버튼을 클릭합니다.

(4-2) Choose Recipe

Image Recipe 는 output image 에 대해 원하는 구성을 만들기 위해 소스 이미지에 적용할 구성 요소를 정의하는 부분 입니다. 레시피를 만든 후에는 수정할 수 없습니다. 구성 요소를 변경하려면 새 버전을 만들어야 합니다.

1. Recipe 의 Configuration options 에서 **Create new recipe** 를 선택 합니다.
2. Image type 에서 output type 은 **Amazon Machine Image(AMI)**를 선택 합니다.



3. General 에서 레시피에 대한 Name, Version 및 Description(선택사항)을 입력하세요.
 - Name : **my-first-recipe**
 - Version : **1.0.0**
 - Description : **My first image recipe**

4. Base image 에서 **Select managed images** 를 선택 하고, Image OS 는 **Windows** 를 선택합니다.
5. Image origin 은 **Quick start (Amazon-managed)**를 선택합니다.
6. Image name 에서 **Windows Server 2019 English Full Base x86** 을 선택하세요.
7. Auto-versioning options 에서는 **Use latest available OS version** 을 선택합니다.

Base image [Info](#)

Select image
You can select the source image from a list of Image Builder managed images or Amazon Machine Images (AMIs) that your account has access to.

☒ **Select managed images**
Image Builder managed images created by you, shared with you, or provided by AWS.

☐ **Enter custom AMI ID**
The AWS Systems Manager Agent (SSM Agent) must be pre-installed in the selected AMI.

Image Operating System (OS)
Image Builder supports Amazon Linux, Windows, Ubuntu, CentOS, RHEL, and SLES.

☐ **Amazon Linux**
Amazon Linux 2


☒ **Windows**
Windows Server 1909, 2004, 2012R2, 2016, and 2019


☐ **Ubuntu**
Ubuntu 16, 18 and 20


☐ **CentOS**
CentOS 7 and 8


☐ **Red Hat Enterprise Linux (RHEL)**
RHEL 7 and 8


☐ **SUSE Linux Enterprise Server (SLES)**
SLES 12 and 15


Image origin
Choose the image to configure from a list of previously created pipeline images, images shared with you or a quick start list to help you get started. You could also enter a custom AMI ID to define the source image.

☒ **Quick start (Amazon-managed)**
Amazon curated images to help you get started

☐ **Images owned by me**
Images you created with Image Builder

☐ **Images shared with me**
Images shared with this account

Image name
Choose an image based on the above selections.

Windows Server 2019 English Full Base x86
Owner: Amazon OS: Windows

Auto-versioning options
Choose the OS version that the pipeline can automate for future builds.

☒ **Use latest available OS version**

☐ Use selected OS version

☐ Specify OS version

- Instance configuration 에 User data 부분에 이미지가 실행될 때 구성 스크립트를 실행할 사용자 데이터를 지정할 수 있습니다. 이번 실습에서는 입력하지 않습니다.
- Working directory 섹션에 Working directory path 는 기본적으로 c:/로 설정되어 있습니다. 빌드 및 테스트 워크플로 중에 사용할 작업 디렉토리를 의미합니다. 그대로 둡니다.

(4-3) Adding Components

이제 AMI 에 Component 를 추가 합니다. 여기서는 AWS 에서 제공하는 Component 를 사용합니다. 나중 실습에서는 우리가 직접 Component 를 만들어서 구축하는 것을 해보겠습니다.

- Components 섹션의 Step 1: Choose build components to produce the desired output AMI 부분에서 검색창에 **powershell** 을 입력합니다.
- 검색 결과로 나온 항목 중 **powershell-windows** 를 선택합니다.

Components info

Components are software scripts that define the custom configuration for an image. Components cannot be modified or replaced after a recipe is created. Automatic version choices are provided for each component. A maximum of 20 components (including build and test) can be applied to a recipe.

Step 1: Choose build components to produce the desired output AMI

Build components are software scripts that define a sequence of steps for downloading, installing, and configuring software packages. They also define validation steps.

Build components - Windows (25)

Search: powershell | Filter: Amazon-managed | Page: 1

Name	Description
<input type="checkbox"/> powershell-core-windows	Performs a default installation of PowerShell Core 6.2.3 from the https://github.com/PowerShell/PowerShell repository. Owner: ARN Amazon: arn:aws:imagebuilder:eu-west-1:aws:component/powershell-core-windows/x.x.x
<input checked="" type="checkbox"/> powershell-windows	Installs PowerShell version 7.0.6 from the GitHub repository at https://github.com/PowerShell/PowerShell . Owner: ARN Amazon: arn:aws:imagebuilder:eu-west-1:aws:component/powershell-windows/x.x.x

Selected components (1)

Expand the component to view versioning options. To sort the build sequence, drag the components up and down.

Sequence

1 powershell-windows Owner: Amazon

Versioning options

☒ Use latest available component version

☐ Specify component version

3. 이어서 테스트 구성요소도 추가해보겠습니다. 인스턴스가 부팅되고 Password 를 생성할 수 있는지 확인하는 테스트를 추가해보겠습니다.

Step 2: *Optional* Select tests to verify the output AMI (post-build) 에서 검색창에 **simple** 을 입력합니다.

4. 검색 결과로 나온 항목 중 **simple-boot-test-windows** 를 선택합니다.

Step 2: *Optional* Select tests to verify the output AMI (post-build)

Test components are a sequence of steps used to verify that the output image built by your image pipeline is functioning as expected.

Test components - Windows (8)

Search: simple | Filter: Amazon-managed

Name	Description
simple-boot-test-windows	Executes a simple boot test.

Selected components (1)

Expand the component to view versioning options and input parameters. To sort the build sequence, drag the components up and down.

Sequence

Component (drag the component up or down to change the sequence)

1 simple-boot-test-windows Owner: Amazon

Versioning options

- ☒ Use latest available component version
- ☐ Specify component version

5. 다음으로 Storage (volumes)에서 선택적으로 파이프라인에 대한 스토리지 장치 설정을 지정할 수 있습니다. 볼륨 유형, KMS 키, 크기, 종료시 EBS 볼륨을 종료할지 여부 파악, IOPS 등을 선택적으로 지정할 수 있으며, 여기서는 그대로 둡니다.

6. **Next** 버튼을 클릭 합니다.

(4-4) Infrastructure Configuration

마지막으로 인프라 구성을 설정해야 합니다. 여기서는 앞에서 미리 생성한 IAM role 인 TeamRole-builder 을 사용하여 CloudWatch 에 로그를 보내거나 s3 에서 파일을 가져오는 것과 같은 빌드 인스턴스 권한을 부여할 수 있습니다.

1. Configuration options 에서 **Create a new Infrastructure configuration** 을 선택 합니다.
2. General 섹션에서 Name 은 **my-first-infraconfig** 라고 지정합니다.
3. IAM role 에서는 앞에서 생성한 **TeamRole-builder** 을 선택합니다.

Infrastructure configuration

Configuration options

☐ Create infrastructure configuration using service defaults

☐ Use existing infrastructure configuration

☒ Create a new infrastructure configuration

General

Enter a name and description for this configuration.

Name

my-first-infraconfig

Maximum of 128 characters. Letters, numbers, spaces, -, and _ are allowed

Description - optional

Maximum of 1024 characters

IAM role [Info](#)

Select a role to associate with the instance profile. This role defines what permissions the instances launched by EC2 Image Builder will have in your account. These permissions are used to download and execute your components, upload logs to CloudWatch, and perform any additional actions specified in your selected components.

TeamRole ▼

↺

[Create new role](#) [↗](#)

4. AWS infrastructure 섹션에서 Instance type 은 **c5.large** 를 선택합니다.

AWS infrastructure
Service-specific defaults will be applied if you do not select values.

Instance type [Info](#)
Select one or more instance types to customize your image.

Choose one or more instance types ▼

c5.large X

SNS topic [Info](#)
Select an SNS topic to receive notifications and alerts from EC2 Image Builder

Choose SNS topic ▼

[Create SNS topic](#)

▶ **VPC, subnet and security groups**
Specify advanced settings to launch the instance that will customize your image.

▶ **Troubleshooting settings** [Info](#)
Specify settings to troubleshoot issues with building your image.

▶ **Instance metadata settings**
The instance metadata options which apply to the HTTP requests that pipeline builds use to launch EC2 build and test instances.

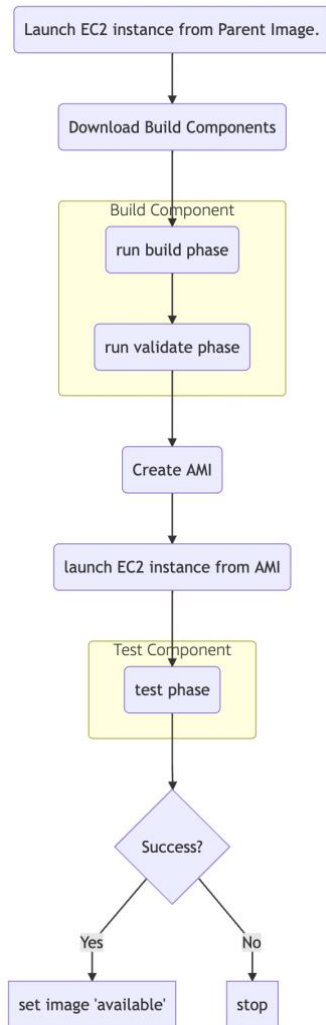
5. 맨 아래 **Next** 버튼을 클릭 합니다.

(4-5) Distribution Configuration

배포 설정에는 암호화를 위한 특정 리전 설정, 시작 권한, output AMI 를 시작할 수 있는 계정, output AMI 이름 및 라이선스 구성이 포함됩니다. 이번 실습에서는 이미지를 계정에 비공개로 유지하므로 따로 설정을 변경할 필요가 없습니다.

1. Define distribution settings – *optional*에서는 맨 아래 **Next** 버튼을 클릭합니다.
2. 화면 아래 **Create pipeline** 버튼을 클릭 합니다.

지금까지 생성한 파이프라인의 흐름을 리뷰해보겠습니다.



(4-6) Run the Pipeline

이제 생성된 Pipeline 을 실행해보겠습니다.

1. [EC2 Image Builder Console](#) 로 이동하고 왼쪽 패널에서 Image pipelines 를 선택 합니다.
2. Image pipelines 섹션에서 우리가 만든 **my-first-pipeline** 을 선택 합니다.

3. Actions 버튼을 클릭하고 **Run pipeline** 을 클릭 합니다.

The screenshot shows the AWS CloudFormation console for a pipeline named 'my-first-pipeline'. The 'Actions' menu is open, showing options: 'Delete', 'Run pipeline', 'Disable pipeline', and 'Edit pipeline'. The 'Run pipeline' option is highlighted. Below the menu, the 'Summary' section displays the following details:

Summary			
Description	Date created	Date of last run	Date of next run
-	May 11, 2021 10:50 AM	-	2021-05-17T09:00:00.000Z
IAM role	Build schedule (UTC)	Status	Enhanced metadata collection
TeamRole	cron(0 9 ? * mon)	Enabled	Enabled

4. 다시 Image pipelines 페이지로 돌아와서 my-first-pipeline 을 클릭 합니다.
여기서 Output images 에서 Status 가 Pending 또는 Building 상태임을 확인합니다.

The screenshot shows the 'Output images' section for the 'my-first-pipeline'. It displays a table with the following information:

Version	Type	Date created	Status
1.0.0/1	AMI	Jun 14, 2021 5:47 PM	Building

5. [EC2 Console](#) 화면으로 이동합니다.
6. 여기서 이미지를 빌드하는데 사용되는 c5.large 타입의 EC2 인스턴스를 확인할 수 있습니다.

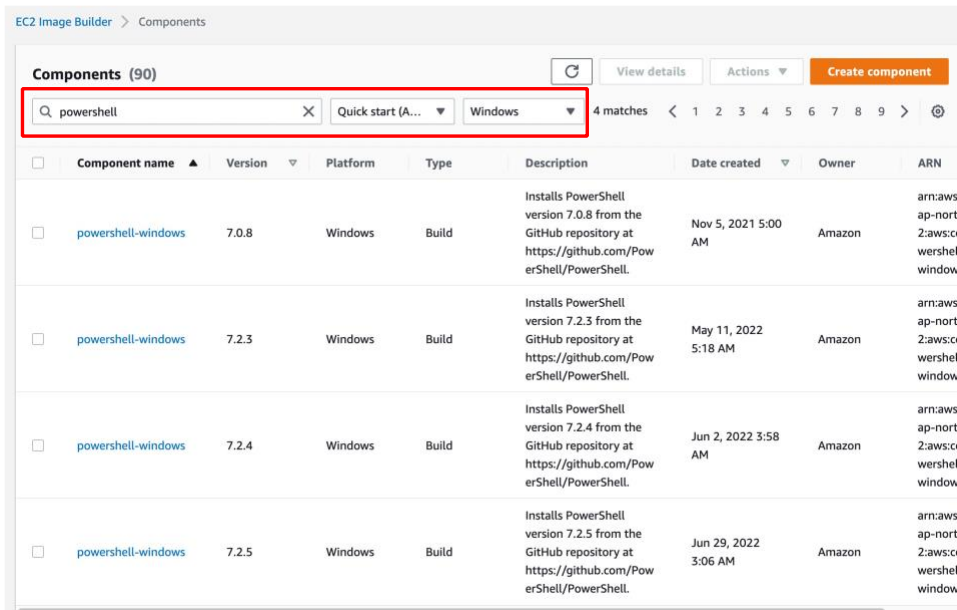
The screenshot shows the AWS EC2 console 'Instances (1)' page. It displays a table with the following information:

Name	Instance ID	Instance state	Instance type
Build instance for my-first-recipe	i-0d6b37fdd2d1c2267	Running	c5.large

(4-7) 추가 확인#1 : DOCUMENT

EC2 Image Builder 에서 Pipeline 을 만들 때 추가했던 Component 에 대해 자세히 살펴보겠습니다.

1. [EC2 Image Builder Console](#) 로 이동하고 왼쪽 패널에서 Components 를 선택 합니다.
2. 검색창 옆에 있는 Owned by me 부분을 **Quick start (Amazon-managed)**로 바꿔 선택하고, [EC2 Image Builder Console](#) 로 이동하고 왼쪽 패널에서 Image pipelines 를 선택 합니다. 그리고 검색창에서 **powershell** 을 입력합니다.



3. 앞서 Pipeline 에서 사용한 것과 동일한 버전(7.2.5)을 클릭 합니다.
4. 다음과 같은 패턴의 DOCUMENT 를 확인할 수 있습니다.

이 Component 는 Build 및 Validation 단계, 여러 단계로 구성되며 모든 단계는 하나 이상의 Powershell 명령을 실행합니다.

```
phases:
- name: build
  steps:
  - name: InstallerUri
    action: ExecutePowerShell
    onFailure: Abort
    timeoutSeconds: 5
```

inputs:

commands:

EC2 Image Builder > Components > powershell-windows

powershell-windows Delete Create new version

Summary

Version 7.2.5	Type Build	Image OS Windows	Compatible OS Versions -
KMS key -	Date created Jun 29, 2022 3:06 AM	Owner Amazon	Description Installs PowerShell version 7.2.5 from the GitHub repository at https://github.com/PowerShell/PowerShell .
Change description Updated for PowerShell version 7.2.5.		ARN arn:aws:imagebuilder:ap-northeast-2:aws:component/powershell-windows/7.2.5/1	

Content

```

13 # SUFFIXES OR THE USE OF OTHER SUFFIXES IN THE SUFFIXES.
16 name: PowerShell 7.2.5
17 description: 'Installs PowerShell version 7.2.5 from the GitHub repository at https://github.com/PowerShell/PowerShell.'
18 schemaVersion: 1.0
19 constants:
20 - PowerShellVersion:
21   type: string
22   value: '7.2.5'
23 - InstallerChecksum:
24   type: string
25   value: '632A69E446F96A83A1EE0AA1647C5970DF7B5936BDF27DF3CF18E6C63C21198'
26 - ExitWithFailure:
27   type: string
28   value: '[System.Environment]::Exit(1)'
29 phases:
30 - name: build
31   steps:
32     - name: InstallerUri
33       action: ExecutePowerShell
34       onFailure: Abort
35       timeoutSeconds: 60
36       inputs:
37         commands:
38           - $releases = 'https://github.com/PowerShell/PowerShell/releases'
39           - $arch = 'x64'
40           - Write-Host "$releases/download/v[{{ PowerShellVersion }}/PowerShell-{{ PowerShellVersion }}-win-$arch.msi"
41     - name: InstallerDestination
42       action: ExecutePowerShell
43       onFailure: Abort
44       timeoutSeconds: 60
45       inputs:
46         commands:
47           - $fileName = '{{ build.InstallerUri.outputs.stdout }}'.Split('/')[-1]
48           - Join-Path -Path $env:TEMP -ChildPath $fileName
49     - name: InstallerDownload
50       action: WebDownload

```

(4-8) 추가 확인#2 : LOGGING

이제 EC2 Image Builder 에서 수행한 Pipeline 의 로그를 확인해보겠습니다.

기본적으로 로그는 **CloudWatch** 로그로 전송됩니다. 선택적으로 s3 에 기록할 수도 있습니다.

1. [CloudWatch 콘솔](#)로 이동 합니다.
2. 왼쪽 패널에서 **Log groups** 를 클릭 합니다.

3. 표시되는 로그 그룹에서 **/aws/imagebuilder/my-first-recipe** 을 클릭 합니다.
그리고 Log streams **1.0.0/1** 을 클릭 합니다.

CloudWatch > Log groups > /aws/imagebuilder/my-first-image-recipe > 1.0.0/1

Log events
You can use the filter bar below to search for and match terms, phrases, or values in your log events. [Learn more about filter patterns](#)

Filter events

Clear 1m 30m 1h 12h Custom

Timestamp	Message
	There are older events to load. Load more
2022-04-28T18:56:31.485+02:00	Setup: STARTED SETUP
2022-04-28T18:56:31.492+02:00	Setup: Document arn:aws:imagebuilder:eu-central-1:454249771573:component/powershell-windows/7.2.2/1 is an EC2 Image Builder Component Version Arn
2022-04-28T18:56:31.611+02:00	OW Logs Monitor: (File C:\Program Files\Amazon\TaskOrchestratorAndExecutor\TOE_2022-04-28_16-56-31_UTC-0_275522a6-c714-11ec-97c3-0a98a76e9bc6/console.log) Th...
2022-04-28T18:56:32.387+02:00	Setup: Document arn:aws:imagebuilder:eu-central-1:454249771573:component/simple-boot-test-windows/1.0.0/1 is an EC2 Image Builder Component Version Arn
2022-04-28T18:56:32.789+02:00	Document arn:aws:imagebuilder:eu-central-1:454249771573:component/powershell-windows/7.2.2/1
2022-04-28T18:56:32.789+02:00	Phase build
2022-04-28T18:56:32.789+02:00	Step InstallIrdi
2022-04-28T18:56:32.789+02:00	ExecutePowerShell: STARTED EXECUTION
2022-04-28T18:56:32.789+02:00	ExecutePowerShell: Document input parameter: commands
2022-04-28T18:56:32.789+02:00	ExecutePowerShell: Created temporary directory: C:\Windows\TEMP\AWSSTOE1086952246
2022-04-28T18:56:32.789+02:00	ExecutePowerShell: Created temporary script file at: C:\Windows\TEMP\AWSSTOE1086952246\script-738985437.ps1
2022-04-28T18:56:32.789+02:00	CmdExecution: Preparing object for command C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe with arguments [-InputFormat None -Noninteractive -NoPro...
2022-04-28T18:56:32.789+02:00	CmdExecution: Starting execution of command with arguments [C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -InputFormat None -Noninteractive -NoPr...
2022-04-28T18:56:33.335+02:00	Stdout: https://github.com/PowerShell/PowerShell/releases/download/v7.2.2/PowerShell-7.2.2-win-x64.msi
2022-04-28T18:56:33.336+02:00	CmdExecution: Terminating read operation on STDOUT pipe - EOF
2022-04-28T18:56:33.367+02:00	CmdExecution: Command execution has been completed
2022-04-28T18:56:33.367+02:00	CmdExecution: Command execution was completed successfully
2022-04-28T18:56:33.367+02:00	CmdExecution: Stderr:
2022-04-28T18:56:33.367+02:00	CmdExecution: ExitCode 0

위와 비슷한 로그를 확인하실 수 있습니다. 몇가지 순서대로 확인을 해보자면,

- 선택한 문서가 다운 됨 (~component/powershell-windows/7.2.2/1)
- 다음으로 Build 단계가 호출 됨
- Step 시간순으로 처리가 됨
- CmdExecution 성공 후, 종료코드 ExitCode (0)가 표시됩니다.

[TIP] 종료 코드를 확인하는 것은 중요합니다!

Command 로 종료되지 않는 오류가 발생하면 ExitCode 는 0 이 되지만, 파이프라인은 계속 실행 됩니다.

(4-9) CLEAN-UP

AMI 이미지가 성공적으로 생성되었는지 확인해 봅니다.

[EC2 Image Builder 콘솔의 왼쪽 패널의 Images](#) 에서 확인할 수 있습니다.

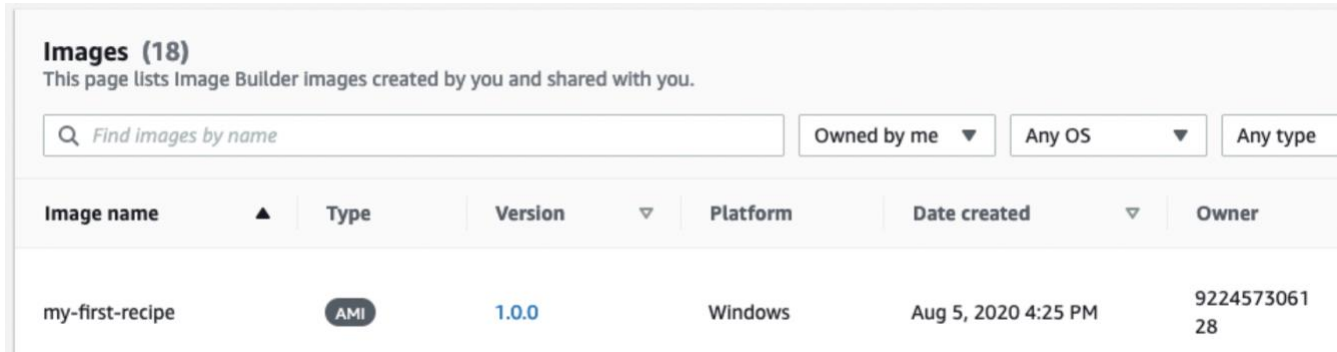
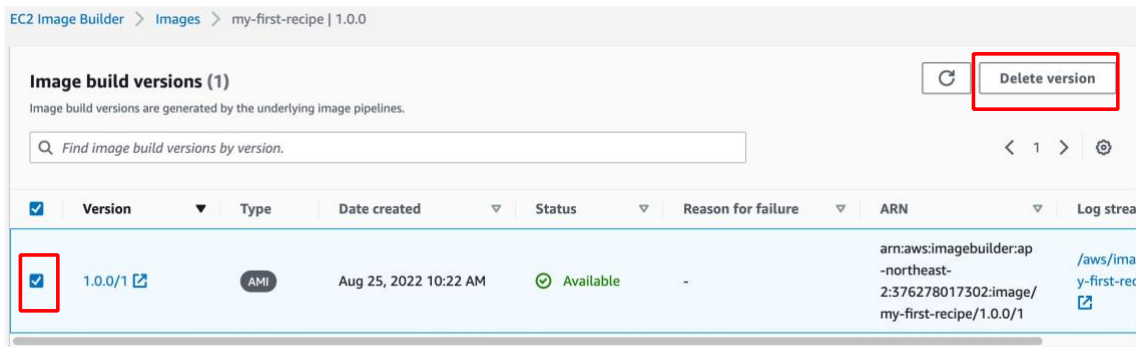


Image name	Type	Version	Platform	Date created	Owner
my-first-recipe	AMI	1.0.0	Windows	Aug 5, 2020 4:25 PM	922457306128

이제 불필요한 과금을 막기 위해 앞에서 만들었던 리소스들을 정리하겠습니다.

[EC2 Image Builder – Images]

1. [EC2 Image Builder 콘솔의 왼쪽 패널의 Images](#) 에서 **Version** (1.0.0)을 클릭 합니다.
2. Image build versions 에 표시되는 이미지 **Version** 을 선택하고 우측 상단의 **Delete version** 버튼을 클릭 합니다.



Version	Type	Date created	Status	Reason for failure	ARN	Log stream
1.0.0/1	AMI	Aug 25, 2022 10:22 AM	Available	-	arn:aws:imagebuilder:ap-northeast-2:376278017302:image/my-first-recipe/1.0.0/1	/aws/ima-y-first-rec

3. Confirm 을 위해 **Delete** 라고 타이핑 하고 **Delete** 버튼을 클릭 합니다.

Delete images

X

The following selected images will be deleted:

- my-first-recipe, 1.0.0/1, arn:aws:imagebuilder:ap-northeast-2:376278017302:image/my-first-recipe/1.0.0/1

To confirm deletion, type Delete in the field.

Delete

Cancel

Delete

[EC2 Console – AMI, Snapshot]

1. [EC2 콘솔](#)의 왼쪽 패널의 **AMIs** 에서 파이프라인으로 만들어진 AMI 를 선택 합니다.
2. AMI ID 를 복사 합니다.

Amazon Machine Images (AMIs) (1/1) info

Recycle Bin

EC2 Image Builder

Actions

Launch instance from AMI

Owned by me

Search

<input checked="" type="checkbox"/>	Name	AMI ID	AMI name	Source	Owner	Visit
<input checked="" type="checkbox"/>	-	ami-0a66a83aee8206f3b	my-first-recipe 2022-08-25T01-...	376278017302/my-first-recipe 2022-0-...	376278017302	Priv

AMI ID: ami-0a66a83aee8206f3b

Details

Permissions

Storage

Tags

AMI ID

ami-0a66a83aee8206f3b

Image type

machine

Platform details

Windows

Root device type

EBS

AMI name

my-first-recipe 2022-08-25T01-22-52.819Z

Owner account ID

376278017302

Architecture

x86_64

Usage operation

RunInstances:0002

3. Actions -> **Deregister AMI** 를 클릭 합니다. (팝업에서도 Deregister AMI 버튼을 클릭 합니다)

Amazon Machine Images (AMIs) (1/1) info

Recycle Bin

EC2 Image Builder

Actions

Launch instance from A

Owned by me

Search

<input checked="" type="checkbox"/>	Name	AMI ID	AMI name	Source
<input checked="" type="checkbox"/>	-	ami-0a66a83aee8206f3b	my-first-recipe 2022-08-25T01-...	376278017302/my-first-recipe

AMI ID: ami-0a66a83aee8206f3b

Details

Permissions

Storage

Tags

Copy AMI

Edit AMI permissions

Request Spot Instances

Manage tags

Deregister AMI

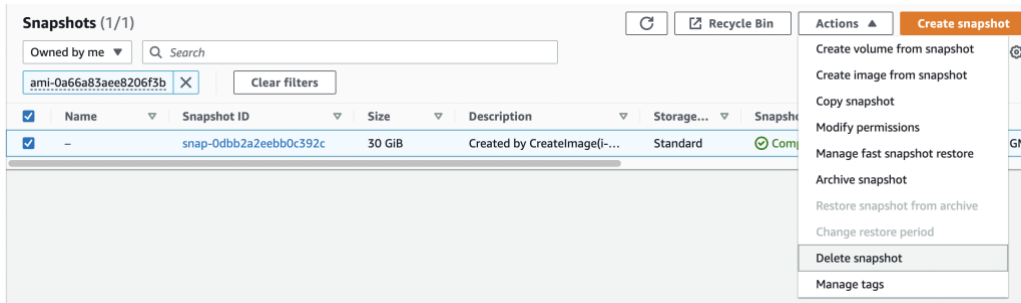
Change description

Manage image optimization

Manage AMI Deprecation

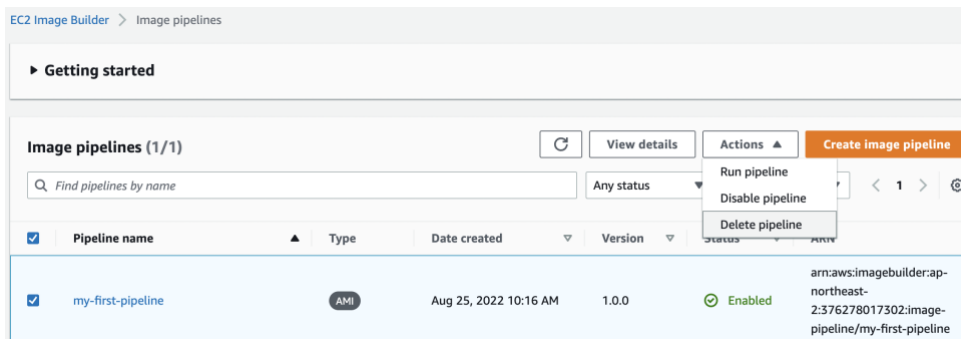
Register instance store-backed AMI

4. 왼쪽 패널의 Snapshots 으로 이동 합니다.
5. 검색창에 앞에서 복사한 AMI ID 를 붙여넣습니다.
6. Actions -> Delete snapshot -> Delete 를 클릭 합니다.



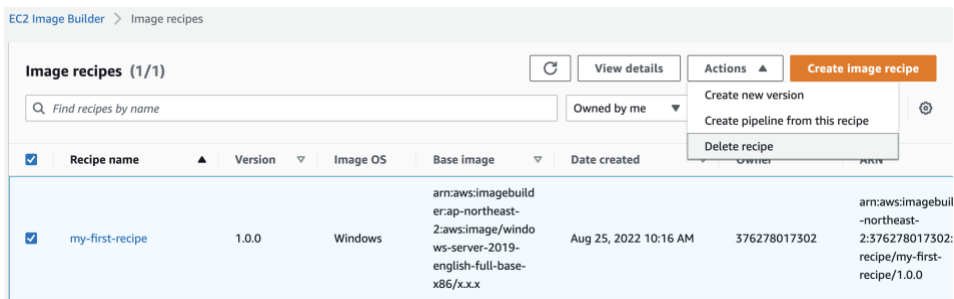
[EC2 Image Builder – Pipeline, Recipe, Distribution, Infrastructure configurations]

1. [EC2 Image Builder 콘솔](#)의 왼쪽 패널의 Image Pipelines 를 선택 합니다.
2. Pipeline 을 선택하고 Actions -> Delete 를 클릭 합니다.



3. Delete 라고 타이핑하고 Delete 버튼을 클릭 합니다.
4. [EC2 Image Builder 콘솔](#)의 왼쪽 패널의 Image recipes 를 선택 합니다.

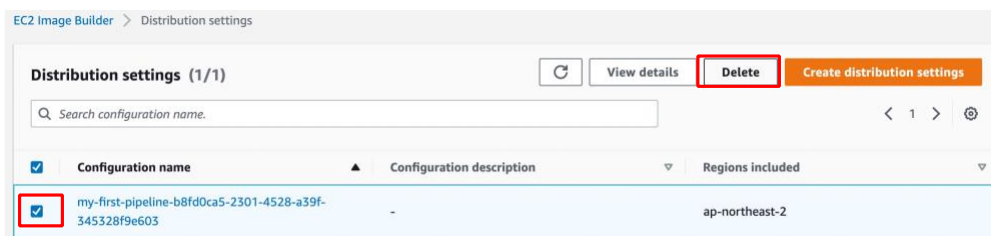
5. Recipe 를 선택하고 Actions -> Delete recipe 를 클릭 합니다.



6. Delete 라고 타이핑하고 Delete 버튼을 클릭 합니다.

7. EC2 Image Builder 콘솔의 왼쪽 패널의 Distribution settings 를 선택 합니다.

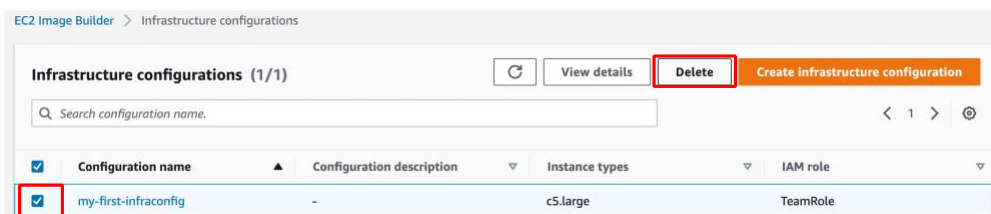
8. Configuration 을 선택하고 상단의 Delete 버튼을 클릭 합니다.



9. Delete 라고 타이핑하고 Delete 버튼을 클릭 합니다.

10. EC2 Image Builder 콘솔의 왼쪽 패널의 Infrastructure configuration 을 선택 합니다.

11. Configuration 을 선택하고 상단의 Delete 버튼을 클릭 합니다.



12. Delete 라고 타이핑하고 Delete 버튼을 클릭 합니다.

5. Create EC2 Image Builder Pipeline #2

이번 실습에서는 EC2 Image Builder 에서 Component 를 직접 만들고 Pipeline 을 만들어서 AMI 를 생성 후, 실제 EC2 instance 까지 launching 해 보겠습니다. 만들고자 하는 AMI 는 LAMP Stack 을 설치한 간단한 웹서버 AMI 입니다.

기본적인 흐름은 앞선 4. Create EC2 Image Builder Pipeline #1 과 같으나, Pipeline 을 만들기 전 사용자 Components 를 만드는 것부터 시작 합니다.

(5-1) Creating Components

1. [EC2 Image Builder Console](#) 로 이동 합니다.
그리고 왼쪽 패널에서 **Components** 를 선택 합니다.
2. **Create component** 버튼을 클릭 합니다.
3. Component type 섹션에서 Type 은 **Build** 를 선택 합니다.
4. Component details 섹션에서 각 정보는 다음과 같이 합니다.
 - Image operating system (OS) : **Linux** 선택
 - Compatible OS Versions : **Amazon Linux 2** 선택
 - Component name : **InstallLAMPstack**
 - Component version : **1.0.0**

Create component

Both build components and tests are part of a recipe. Build components contain software, settings, and configurations that are installed or applied during the process of building custom images. Tests are run after a custom image is built to validate functionality, security, performance, etc.

Component type

Type

Choose the component type.

☒ **Build**
Documents that define a sequence of steps for downloading, installing, and configuring software packages.

☐ **Test**
Tests to run on software packages.

Component details

Image operating system (OS)
Specify the OS the component is compatible with.

Linux

Compatible OS Versions
Specify the OS versions that this component is compatible with.

Amazon Linux 2

Component name
Maximum of 128 characters. Letters, numbers, spaces, -, and _ are allowed.

Install-LAMP-Stack

Description - optional
Maximum of 1024 characters.

KMS keys - optional
Specify the KMS key to encrypt the component with. Only customer managed keys that have an alias are shown.

Image Builder service key (enabled by default)

Component version
Format: major.minor.patch

1.0.0

Change description - optional
Details about changes made to the specific version of the component.

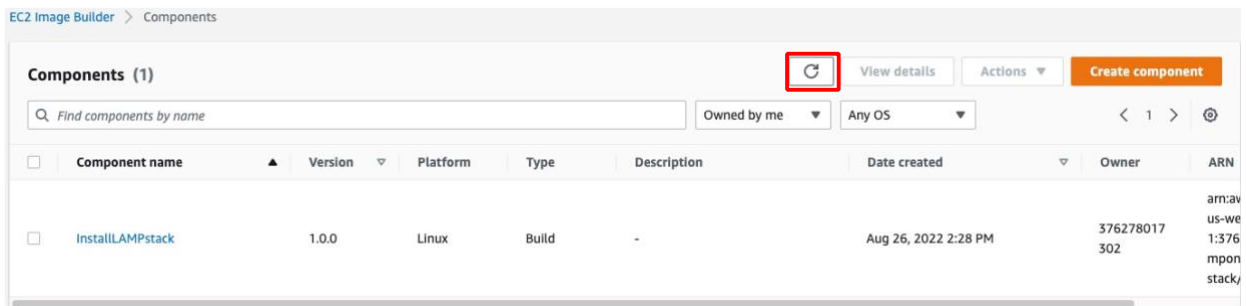
Maximum of 1024 characters.

5. Definition document 섹션에서 Define document content 를 선택하고, Content ([LINK](#))는 아래 내용을 넣습니다.

```
name: InstallLAMPstack
description: Install LAMP stack.
schemaVersion: 1.0

phases:
  - name: build
    steps:
      - name: InstallLAMPstack
        action: ExecuteBash
        inputs:
          commands:
            - amazon-linux-extras install -y lamp-mariadb10.2-php7.2 php7.2
            - yum -y install httpd php-mbstring
```

6. 화면 아래 **Create component** 버튼을 클릭 합니다.
7. 새로 고침 버튼을 클릭하면 방금 만든 Component 가 보입니다.



8. 2~6 단계를 반복하여, 3 개의 Components 를 더 만들어 봅니다.

추가로 만들 Components 의 이름은 다음과 같습니다.

- StartWebServer
- InstallWebPages
- InstallAWSSDK

9. StartWebServer component 생성
- component name : **StartWebServer**
 - Content ([LINK](#))

```
name: StartWebServer
description: Start the web server
schemaVersion: 1.0
```

phases:

- name: build

steps:

- name: StartWebServer

action: ExecuteBash

inputs:

commands:

- chkconfig httpd on
- systemctl start httpd

10. InstallWebPages component 생성

- component name : **InstallWebPages**

- Content ([LINK](#))

```
name: InstallWebPages
description: Install the web pages for this lab
schemaVersion: 1.0
```

phases:

- name: build

steps:

- name: InstallWebPages

action: ExecuteBash

inputs:

commands:

- cd /var/www/html;wget https://aws-joozero.s3.ap-northeast-2.amazonaws.com/immersion-day-app-php7.tar.gz
- cd /var/www/html;tar xvfz immersion-day-app-php7.tar.gz

11. InstallAWSSDK component 생성

- component name : **InstallAWSSDK**

- Content ([LINK](#))

```
name: InstallAWSSDK
description: Install the AWS SDK for PHP
schemaVersion: 1.0
```

phases:

- name: build

steps:

- name: InstallAWSSDK

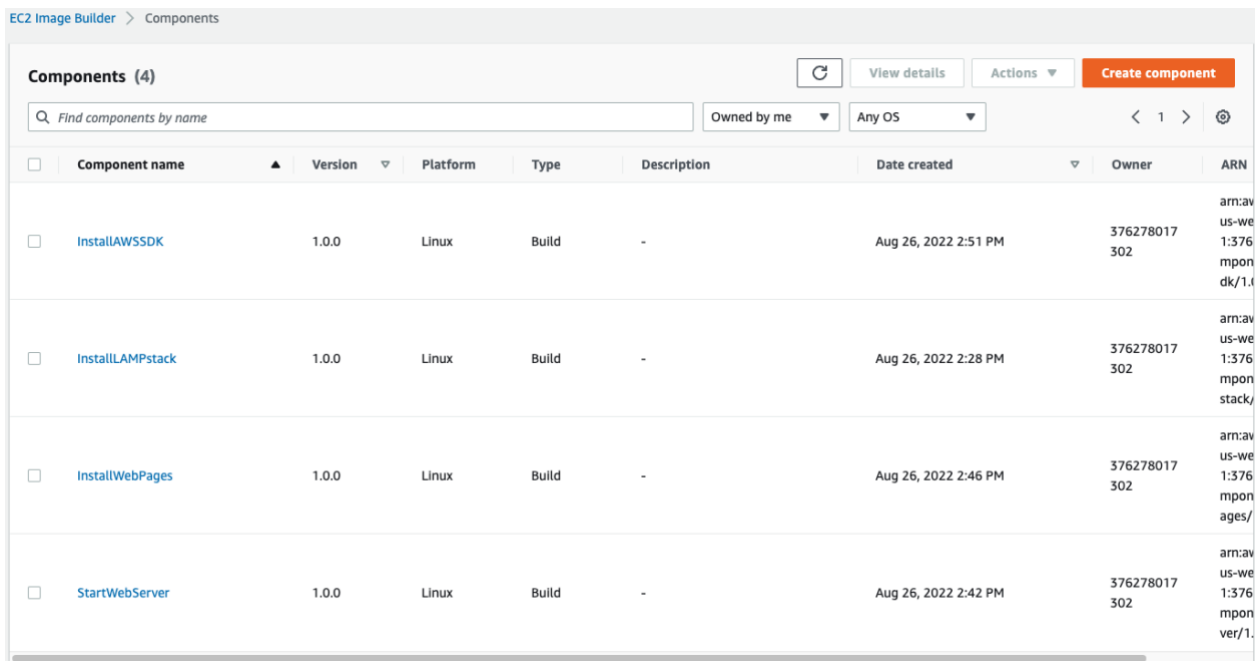
action: ExecuteBash

inputs:

commands:

- mkdir /var/www/html/vendor
- cd /var/www/html/vendor;wget https://docs.aws.amazon.com/aws-sdk-php/v3/download/aws.zip
- cd /var/www/html/vendor;unzip aws.zip

12. 새로 고침 버튼을 클릭하면 총 4 개의 Components 가 보입니다.



The screenshot shows the 'Components' page in the EC2 Image Builder console. At the top, there's a header 'Components (4)' with a refresh button, 'View details', 'Actions', and a 'Create component' button. Below the header is a search bar 'Find components by name' and filters for 'Owned by me' and 'Any OS'. The main content is a table with the following columns: Component name, Version, Platform, Type, Description, Date created, Owner, and ARN. There are four components listed:

Component name	Version	Platform	Type	Description	Date created	Owner	ARN
InstallAWS SDK	1.0.0	Linux	Build	-	Aug 26, 2022 2:51 PM	376278017302	arn:aws:imagebuilder:us-west-1:376278017302:imagebuilder-component/1.0.0
InstallLAMPstack	1.0.0	Linux	Build	-	Aug 26, 2022 2:28 PM	376278017302	arn:aws:imagebuilder:us-west-1:376278017302:imagebuilder-component/stack/1.0.0
InstallWebPages	1.0.0	Linux	Build	-	Aug 26, 2022 2:46 PM	376278017302	arn:aws:imagebuilder:us-west-1:376278017302:imagebuilder-component/pages/1.0.0
StartWebServer	1.0.0	Linux	Build	-	Aug 26, 2022 2:42 PM	376278017302	arn:aws:imagebuilder:us-west-1:376278017302:imagebuilder-component/server/1.0.0

(5-2) Creating an Image Pipeline

1. [EC2 Image Builder Console](#) 에서 왼쪽 패널의 **Image pipelines** 로 이동 합니다.
2. **Create image pipeline** 버튼을 클릭 합니다.
3. General 섹션에서 Pipeline name 은 **my-second-pipeline** 으로 입력 합니다.
4. Build schedule 섹션에서 Schedule options 는 **Manual** 을 선택 합니다.
5. 화면 하단의 **Next** 버튼을 클릭 합니다.

(5-3) Choose Recipe

1. Recipe 섹션에서 Configuration options 는 **Create new recipe** 를 선택 합니다.
2. Image type 에서 output type 은 **Amazon Machine Image(AMI)**를 선택 합니다.
3. General 에서 레시피에 대한 Name, Version 및 Description(선택사항)을 입력하세요.
 - Name : **my-second-recipe**
 - Version : **1.0.0**
 - Description : **My second image recipe**
4. Base image 에서 **Select managed images** 를 선택 하고, Image OS 는 **Amazon Linux** 를 선택합니다.
5. Image origin 은 **Quick start (Amazon-managed)**를 선택합니다.
6. Image name 에서 **Amazon Linux 2 x86** 을 선택하세요.
7. Auto-versioning options 에서는 **Use latest available OS version** 을 선택합니다.

Base image Info

Select image
You can select the base image from a list of Image Builder managed images or Amazon Machine Images (AMIs) that your account has access to or import a virtual image.

☒ **Select managed images**
Image Builder managed images created by you, shared with you, or provided by AWS.
 ☐ Enter custom AMI ID
The AWS Systems Manager Agent (SSM Agent) must be pre-installed in the selected AMI.
 ☐ Import base image
Import from your VM into Image Builder and use the converted image as the base image in your recipe.

Image Operating System (OS)
Image Builder supports Amazon Linux, Windows, Ubuntu, CentOS, RHEL, and SLES.

☒ **Amazon Linux**
Amazon Linux 2
 ☐ Windows
Windows Server 2012R2, 2016, 2019, 2004, 20H2, and 2022
 ☐ Ubuntu
Ubuntu 16, 18 and 20

☐ CentOS
CentOS 7 and 8
 ☐ Red Hat Enterprise Linux (RHEL)
RHEL 7 and 8
 ☐ SUSE Linux Enterprise Server (SLES)
SLES 12 and 15

Image origin
Choose the image to configure from a list of previously created pipeline images, images shared with you or a quick start list to help you get started. You could also enter a custom AMI ID to define the base image.

☒ **Quick start (Amazon-managed)**
Amazon curated images to help you get started
 ☐ Images owned by me
Images you created with Image Builder
 ☐ Images shared with me
Images shared with this account

Image name
Choose an image based on the above selections.

Amazon Linux 2 x86
Owner: Amazon OS: Linux

Auto-versioning options
Choose the OS version that the pipeline can automate for future builds.

☒ **Use latest available OS version**
☐ Use selected OS version
 ☐ Specify OS version

(5-4) Adding Components

1. Components 섹션에서는 앞서 우리가 만든 4 개의 Components 를 사용할 것 입니다.
검색창 옆에 **Owned by me** 를 선택 합니다.
2. 4 개의 Components 를 다음의 순서대로 추가(선택) 합니다. (추가 후 순서 변경도 가능 합니다)
 - 1) InstallLAMPstack
 - 2) StartWebServer
 - 3) InstallWebPages
 - 4) InstallAWSSDK

Build components - Amazon Linux (4)

Find components by name Owned by me < 1 >

<input checked="" type="checkbox"/>	Name	Description
4 <input checked="" type="checkbox"/>	InstallAWSSDK	Owner: 376278017302 ARN: arn:aws:imagebuilder:us-west-1:376278017302:component/installawssdk/x.x.x
1 <input checked="" type="checkbox"/>	InstallLAMPstack	Owner: 376278017302 ARN: arn:aws:imagebuilder:us-west-1:376278017302:component/installlampstack/x.x.x
3 <input checked="" type="checkbox"/>	InstallWebPages	Owner: 376278017302 ARN: arn:aws:imagebuilder:us-west-1:376278017302:component/installwebpages/x.x.x
2 <input checked="" type="checkbox"/>	StartWebServer	Owner: 376278017302 ARN: arn:aws:imagebuilder:us-west-1:376278017302:component/startwebserver/x.x.x

Selected components (4)
Expand the component to view versioning options and input parameters. To sort the build sequence, drag the components up and down.

Sequence	Component (drag the component up or down to change the sequence)	Expand all
1	<div>InstallLAMPstack</div> <div>Versioning options</div> <div>Owner: 376278017302</div> <div>×</div>	
2	<div>StartWebServer</div> <div>Versioning options</div> <div>Owner: 376278017302</div> <div>×</div>	
3	<div>InstallWebPages</div> <div>Versioning options</div> <div>Owner: 376278017302</div> <div>×</div>	
4	<div>InstallAWSSDK</div> <div>Versioning options</div> <div>Owner: 376278017302</div> <div>×</div>	

3. 나머지 항목은 Default 로 두고, 화면 아래 **Next** 버튼을 클릭 합니다.

(5-5) Infrastructure Configuration

마지막으로 인프라 구성을 설정 합니다. 이번에도 IAM role 은 **TeamRole-builder** 을 사용하여 CloudWatch 에 로그를 보내거나 s3 에서 파일을 가져오는 것과 같은 빌드 인스턴스 권한을 부여합니다.

1. Configuration options 에서 **Create a new Infrastructure configuration** 을 선택 합니다.
2. General 섹션에서 Name 은 **my-second-infraconfig** 라고 지정합니다.
3. IAM role 에서는 앞에서 생성한 **TeamRole-builder** 을 선택합니다.
4. AWS infrastructure 섹션에서 Instance type 은 **c5.large** 를 선택합니다.
5. 맨 아래 **Next** 버튼을 클릭 합니다.

(5-6) Distribution Configuration

배포 설정에는 암호화를 위한 특정 리전 설정, 시작 권한, output AMI 를 시작할 수 있는 계정, output AMI 이름 및 라이선스 구성이 포함됩니다. 이번 실습에서는 이미지를 계정에 비공개로 유지하므로 따로 설정을 변경할 필요가 없습니다.

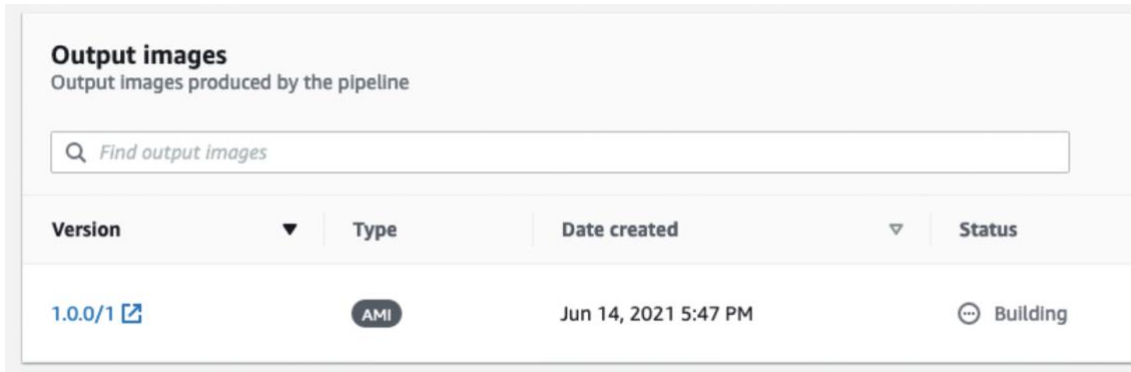
1. Define distribution settings – *optional* 에서는 맨 아래 **Next** 버튼을 클릭합니다.
2. 화면 아래 **Create pipeline** 버튼을 클릭 합니다.

(5-7) Run the Pipeline

이제 생성된 Pipeline 을 실행해보겠습니다.

1. [EC2 Image Builder Console](#) 로 이동하고 왼쪽 패널에서 Image pipelines 를 선택 합니다.

2. Image pipelines 섹션에서 방금 만든 **my-second-pipeline** 을 선택 합니다.
3. Actions 버튼을 클릭하고 **Run pipeline** 을 클릭 합니다.
4. 다시 Image pipelines 페이지로 돌아와서 my-second-pipeline 을 클릭 합니다.
여기서 Output images 에서 Status 가 Pending, Creating 또는 Building 상태임을 확인합니다.

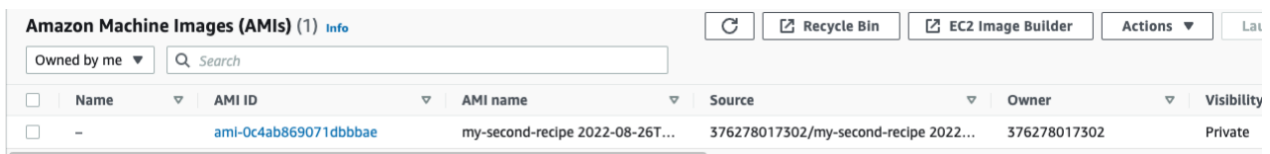


5. Pipeline 작업이 완료되면 Output images 의 버전 Status 가 Testing 으로 바뀝니다.
(지정한 Run Pipeline 작업이 완료되어 AMI 가 생성되는데 까지 약 18 분 정도 소요 됩니다)

(5-8) Launching EC2 instance using AMI

이제 Pipeline 으로 생성된 AMI 로 EC2 Instance 를 기동하겠습니다.

1. [EC2 Console](#) 화면으로 이동합니다. 그리고 왼쪽 패널의 AMIs 를 선택 합니다.
2. AMIs 에서 Pipeline 작업으로 생성된 AMI 를 확인할 수 있습니다.



3. 왼쪽 패널의 Instances 를 선택 합니다.
4. **Launch Instances** 버튼을 클릭 합니다.

5. Name 은 **web-server** 로 지정 합니다.

6. Application and OS Images (Amazon Machine Image) 섹션에서 **My AMIs** 를 선택 합니다.

그리고 **Owned by me** 를 선택하고 AMI 는 **우리가 만든 AMI** 를 선택 합니다.

▼ **Application and OS Images (Amazon Machine Image)** [Info](#)


An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

My AMIs

Quick Start

☒ Owned by me

☐ Shared with me



[Browse more AMIs](#)

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

my-second-recipe 2022-08-26T06-48-02.529Z

ami-0c4ab869071dbbbae

2022-08-26T07:01:47.000Z Virtualization: hvm ENA enabled: true Root device type: ebs

Description

My second image recipe

Architecture	AMI ID
x86_64	ami-0c4ab869071dbbbae

7. Instance type 은 **t2.micro** 를 선택 합니다.

8. Key pair name 은 **Proceed without a key pair (Not recommended)** 를 선택 합니다.

9. Network settings 에서는 **Allow HTTP traffic from the internet** 을 체크 합니다.

10. 나머지 항목은 default 로 두고, **Number of instances** 는 2 개로 입력한 뒤, **Launch instance** 버튼을 클릭 합니다.

인스턴스가 완전히 기동되는데 까지 약 5 분 소요 됩니다.

11. 생성한 EC2 instance 중 하나를 클릭하면 상세 정보를 확인할 수 있습니다.

여기에서 Public IPv4 address(예: 100.100.100.100) 를 복사한뒤,

웹브라우저에서 **http://100.100.100.100:80** 처럼 입력하면 다음과 같은 웹사이트가 표시 됩니다.



LOAD TEST	RDS
Meta-Data	Value
InstanceId	<i>i-0ca40bdfd4a43a9c1</i>
Availability Zone	<i>us-west-1a</i>

Current CPU Load: 0%

나머지 인스턴스에 대해서도 웹페이지를 띄워 봅니다.

12. 2 개의 웹서버에서 표시되는 웹페이지에 **LOAD TEST** 를 각각 클릭 합니다.
(스크립트가 수행되며 CPU 사용률을 올리게 됩니다)

6. CloudWatch 모니터링

Amazon CloudWatch 는 AWS 클라우드 리소스 및 AWS 에서 실행하는 애플리케이션을 위한 모니터링 서비스 입니다. Amazon CloudWatch 를 사용하여 지표를 수집 및 추적하고, 로그 파일을 수집 및 모니터링하고, 알람을 설정할 수 있습니다. Amazon CloudWatch 는 Amazon EC2 인스턴스, Amazon DynamoDB 테이블, Amazon RDS DB 인스턴스와 같은 AWS 리소스 뿐만 아니라 애플리케이션 및 서비스에서 생성된 사용자 지정 지표 및 애플리케이션이 생성하는 모든 로그 파일을 모니터링 할 수 있습니다. Amazon CloudWatch 를 사용하여 리소스 사용률, 애플리케이션 성능 및 운영 상태에 대한 시스템 전반의 가시성을 확보할 수 있습니다. 이러한 인사이트를 사용하여 애플리케이션을 원활하게 실행하고 대응할 수 있습니다.

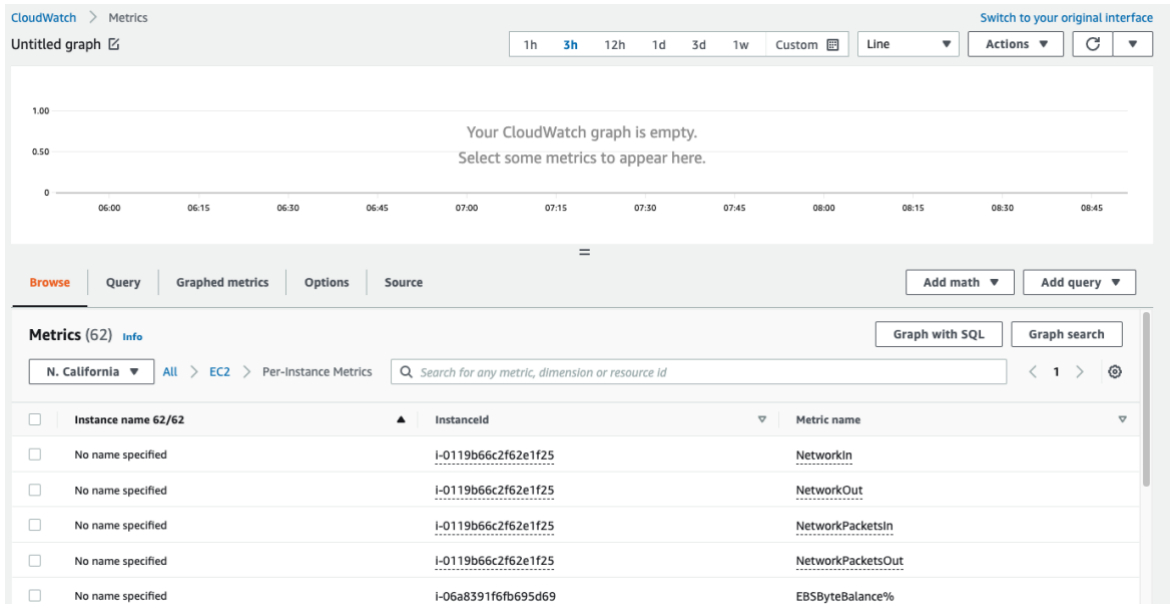
마지막으로 위에서 생성한 EC2 instance 를 CloudWatch 를 통해 간단히 모니터링 하는 방법을 확인해 보겠습니다.

(6-1) CloudWatch Metrics

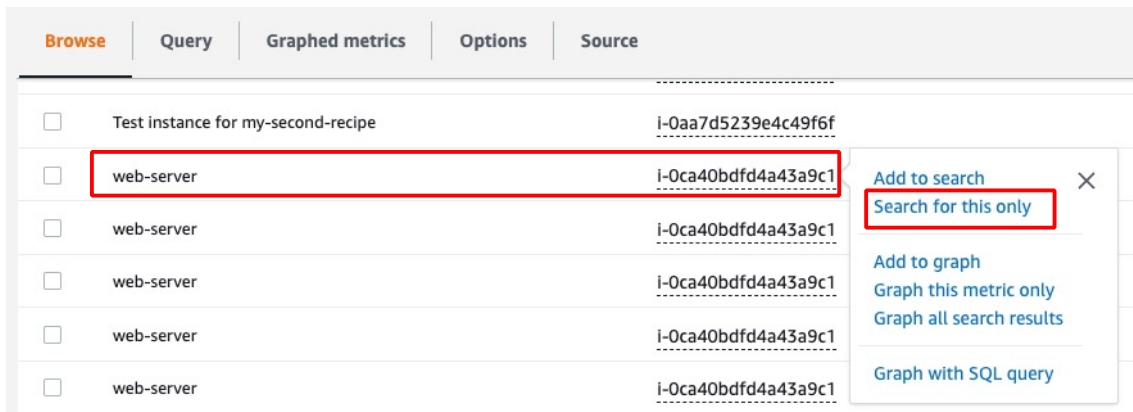
Metric 은 CloudWatch 의 기본 개념 입니다. Metric 은 CloudWatch 에 게시되는 시간 순서가 지정된 데이터 요소 집합을 나타냅니다. 예를 들어 특정 EC2 instance 의 CPU 사용량은 Amazon EC2 에서 제공하는 하나의 Metric 입니다.

Metric 의 각 데이터 요소에는 timestamp 가 있으며 (선택적으로) 측정 단위가 있습니다.

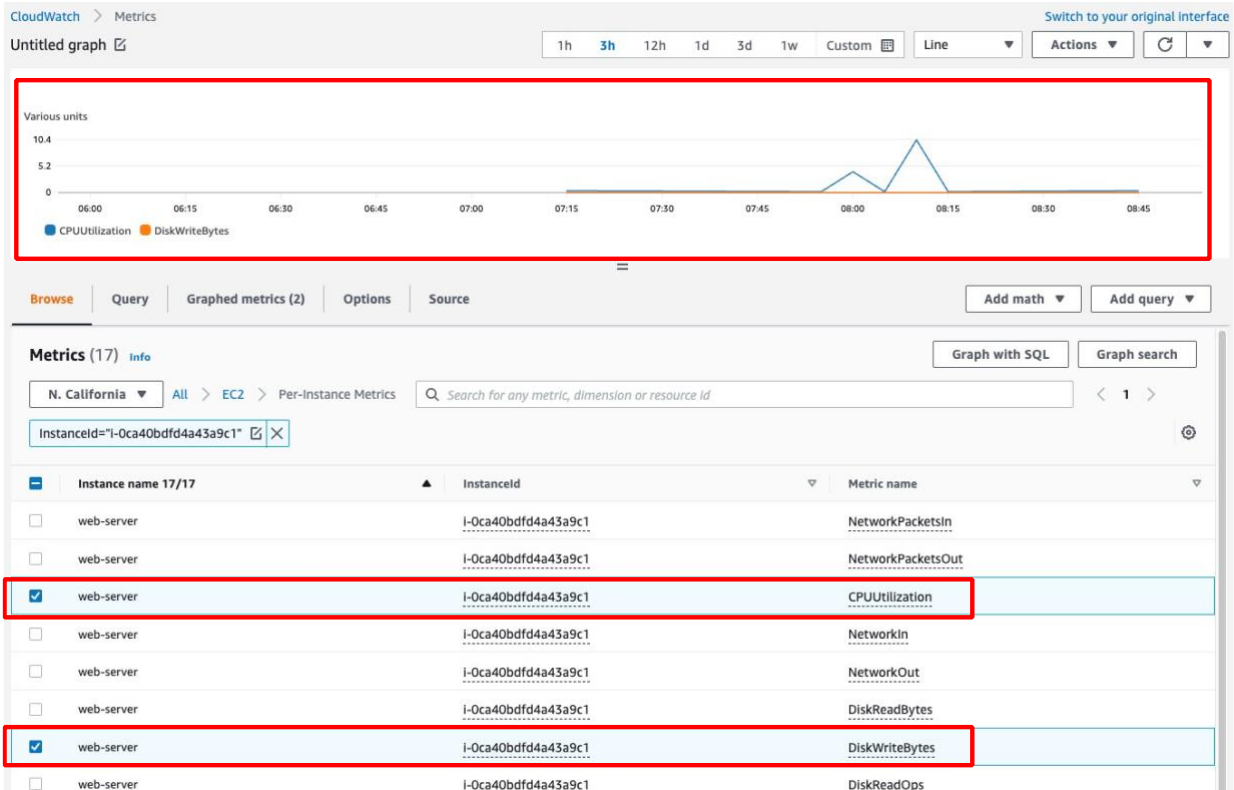
1. [CloudWatch 콘솔](#)로 이동 합니다.
2. 왼쪽 패널에서 Metrics → **All metrics** 를 선택 합니다.
3. **EC2** 를 클릭한 다음, **Per-Instance Metrics** 를 클릭 합니다.
그럼 다음과 같은 화면이 표시 됩니다.



4. 여기서는 web-server 라는 인스턴스에 대해서 모니터링 할 것이기 때문에 Browse 에서 아래에 보면 web-server 의 instance-id 가 보입니다. (web-server 2 개 중 하나를 선택합니다.) 이것을 선택하고 Search for this only 를 클릭 합니다.



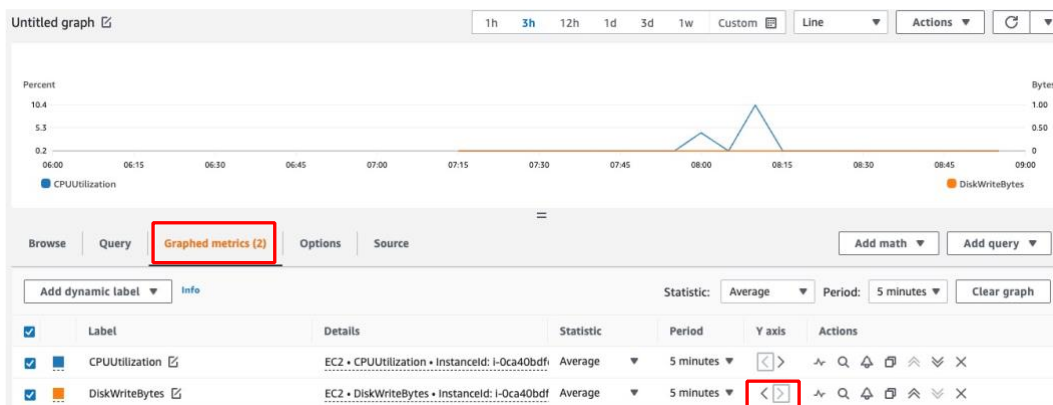
5. Web-server 인스턴스에 대한 metric 만 필터링 되어 확인이 됩니다.
6. 여기서 **CPUUtilization** 과 **DiskWriteBytes** metric 을 선택합니다.
화면 상단에 선택한 metric 에 대한 그래프가 표시되는 것을 볼 수 있습니다.



7. 그래프를 좀 더 보기 좋게 수정해보겠습니다.

Graphed metrics 탭을 클릭 합니다.

다음으로 **DiskWriteBytes** 를 오른쪽 Y 축으로 이동합니다.



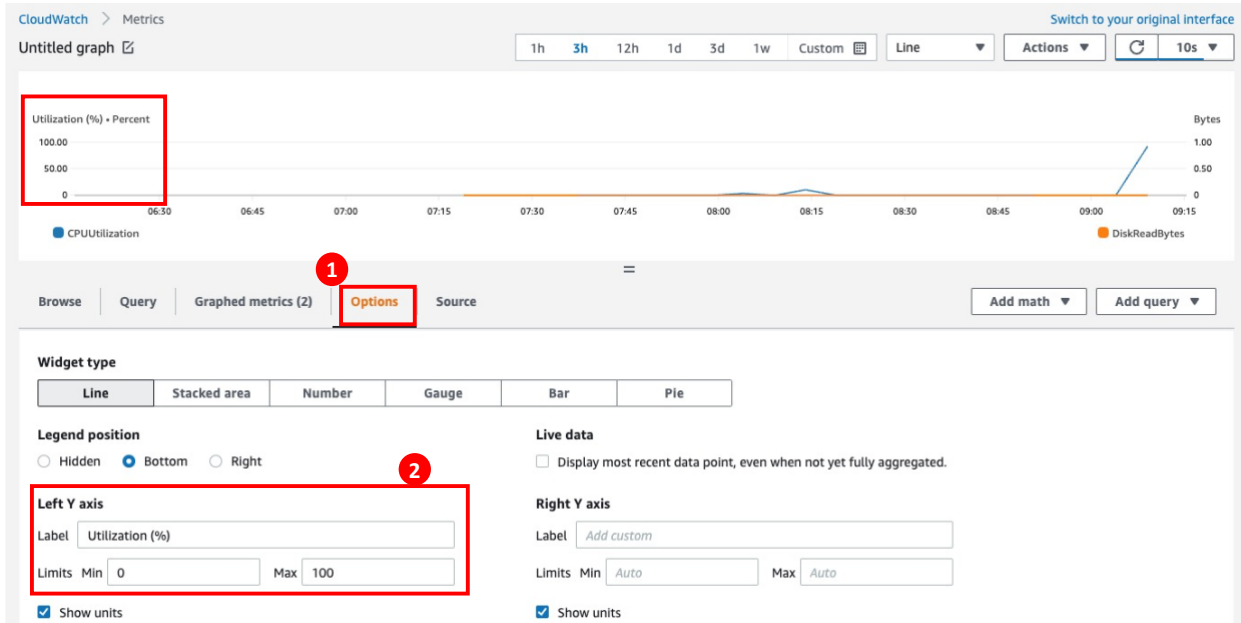
8. 그래프 왼쪽 축에는 Percent 가 있고, 오른쪽 축에는 Bytes 가 있습니다.

Options 탭으로 이동해 왼쪽 Y 축에 대한 Label 과 Limit 값을 지정해 봅시다.

- Left Y axis

Label : **Utilization (%)**

Limits Min : 0 Max : 100



9. 이제 모든 web-server 인스턴스의 CPU 사용률을 보여주는 그래프를 만들어 보겠습니다.
기존에 선택된 metric 은 모두 해제 합니다.

그리고 **Browse** 탭에서 Metric name 이 **CPUUtilization** 을 선택하고 **Search for this only** 를
클릭해서 **2 개의 web-server 의 CPUUtilization metric 을 선택** 합니다.

Browse Query Graphed metrics (2) Options Source

Add math Add query

Metrics (6) Info

N. California All > EC2 > Per-Instance Metrics

Search for any metric, dimension or resource id

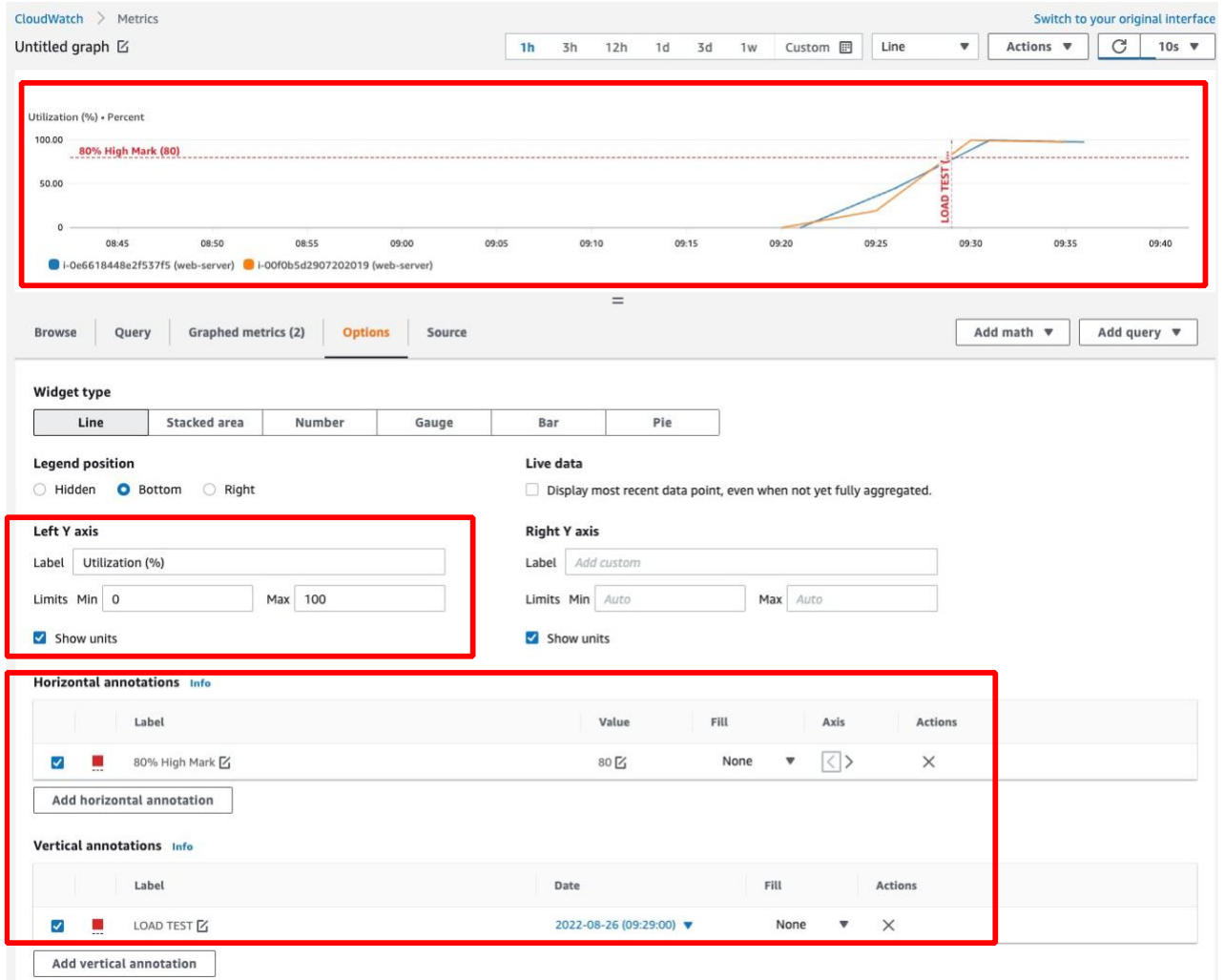
MetricName="CPUUtilization"

Instance name 6/6	Instanceid	Metric name
<input type="checkbox"/> No name specified	i-0119b66c2f62e1f25	CPUUtilization
<input type="checkbox"/> No name specified	i-06a8391f6fb695d69	CPUUtilization
<input type="checkbox"/> Build instance for my-second-recipe	i-078e41ef2613f80f6	CPUUtilization
<input type="checkbox"/> web-server	i-0ca40bdfd4a43a9c1	CPUUtilization • 1 model (Average)
<input checked="" type="checkbox"/> web-server	i-00f0b5d2907202019	CPUUtilization
<input checked="" type="checkbox"/> web-server	i-0e6618448e2f537f5	CPUUtilization

10. **Options** 탭으로 이동 합니다.

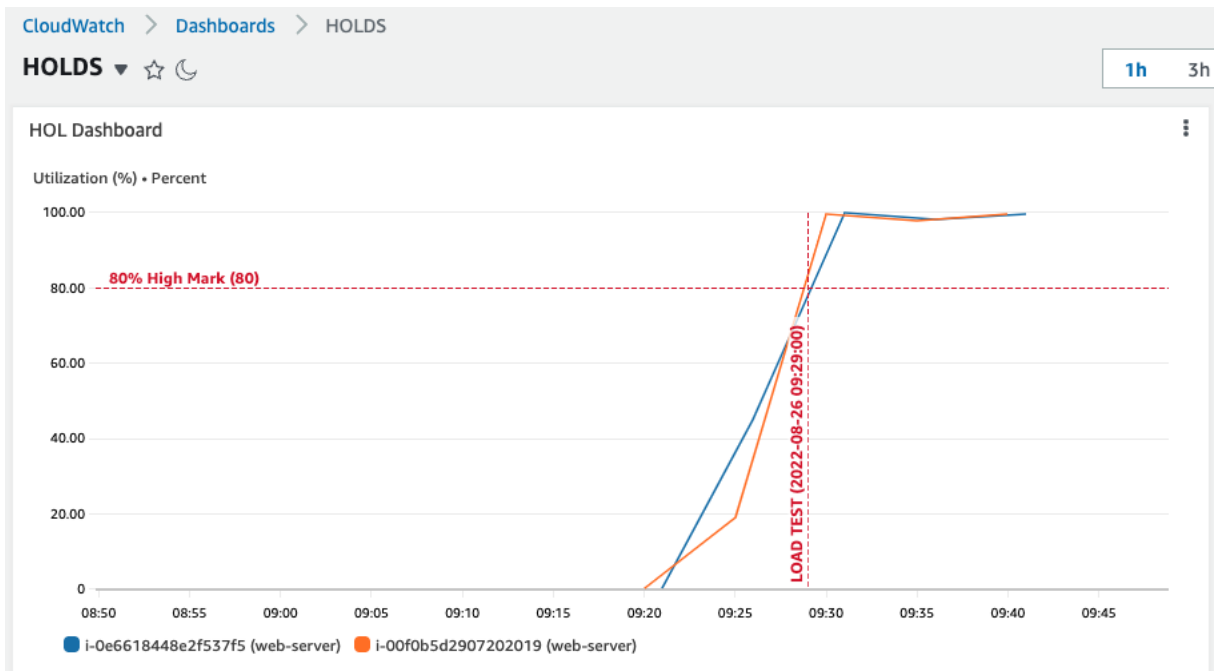
만약 그래프에 보이는 CPU 사용률이 낮다면 Left Y axis 의 Limit Max 값을 낮게 조정해 줍니다.

11. **Horizontal annotation** 의 Label 은 **80% High Mark** 로 지정하고, Value 는 **80** 으로 설정 합니다.
12. **Vertical annotation** 의 Label 은 **LOAD TEST** 로 지정하고, Date 는 CPU 사용률이 올라가는 시점을 선택합니다.



13. 대쉬보드 이름을 지정하고 Action 을 클릭한 뒤, **Add to dashboard** 를 선택 합니다.
팝업에서 **Create New** 버튼을 클릭하고 대쉬보드 이름(HOLDS)을 지정하고 **Create** 버튼을 클릭 합니다. 그리고 **Add to dashboard** 버튼을 클릭 합니다.

14. 이제 CloudWatch > Dashboards 에서 위에서 만든 대쉬보드(HOLDS)를 확인할 수 있습니다.



7. Clean-up

앞서 실습했던 “4. Create EC2 Image Builder Pipeline #1” 에서의 (4.10) CLEAN-UP 에서 CloudWatch dashboard 와 EC2 Instance 삭제가 추가되고 나머지는 반복되는 내용 입니다.

(7-1) CloudWatch - Dashboard 삭제

1. CloudWatch 콘솔의 왼쪽 패널에서 Dashboards 를 선택 합니다.
2. Custom dashboards 에서 앞서 만들었던 HOLDS 항목을 선택하고 Delete 버튼을 클릭 합니다.
3. Delete this dashboard 팝업에서 Delete 버튼을 클릭합니다.

(7-2) EC2 Instances 삭제

1. EC2 콘솔의 왼쪽 패널에서 Instances → Instances 를 선택 합니다.
2. 앞선 실습에서 생성한 인스턴스를 선택하고, Instance state → Terminate instance 를 클릭 합니다.

(7-3) EC2 Image Builder - Images 삭제

1. EC2 Image Builder 콘솔의 왼쪽 패널에서 Image 를 선택 합니다.
2. 실습에서 만든 Version 을 선택하고 우측 상단의 Delete version 을 클릭 합니다.
3. Confirm 을 위한 팝업에서 Delete 를 입력 후 Delete 버튼을 클릭 합니다.

(7-4) AMI / Snapshot 삭제

1. EC2 콘솔의 왼쪽 패널에서 AMIs 를 선택 합니다.
2. 파이프라인으로 만들었던 AMI 를 선택합니다.
3. Details 에서 AMI ID 를 복사합니다.
4. 화면 우측 상단에서 Actions → Deregister AMI 를 클릭 합니다.
5. EC2 콘솔 왼쪽 패널의 Snapshots 으로 이동 합니다.
6. 검색창에 앞에서 복사했던 AMI ID 를 붙여넣습니다.

7. 검색 결과로 나온 Snapshot 을 선택하고 Actions → Delete snapshot → Delete 를 클릭 합니다.

(7-5) EC2 Image Builder – Pipeline / Component / Recipe / Distribution / Infrastructure configurations 삭제

1. EC2 Image Builder 콘솔의 왼쪽 패널에서 Image Pipelines 를 선택 합니다.
2. 파이프라인을 선택하고 Actions → Delete 를 클릭 합니다.
(팝업이 뜨면 Delete 를 입력후 Delete 버튼을 클릭 합니다)
3. 왼쪽 패널에서 Components 를 선택 합니다.
4. 생성했던 4 개의 Components 를 선택하고 Actions → Delete component 를 클릭 합니다.
(팝업이 뜨면 Delete 를 입력후 Delete 버튼을 클릭 합니다)
5. 왼쪽 패널에서 Image recipes 를 선택 합니다.
6. 생성했던 Recipe 를 선택하고 Actions → Delete recipe 를 클릭 합니다.
(팝업이 뜨면 Delete 를 입력후 Delete 버튼을 클릭 합니다)
7. 왼쪽 패널에서 Distribution settings 를 선택 합니다.
8. Configuration 을 선택하고 Delete 버튼을 클릭 합니다.
(팝업이 뜨면 Delete 를 입력후 Delete 버튼을 클릭 합니다)
9. 왼쪽 패널에서 Infrastructure configuration 을 선택 합니다.
10. 생성했던 Configuration 을 선택하고 상단의 Delete 버튼을 클릭 합니다.
(팝업이 뜨면 Delete 를 입력후 Delete 버튼을 클릭 합니다)

(7-6) IAM role 삭제

1. IAM 콘솔의 왼쪽 패널에서 Roles 를 선택 합니다.
2. 검색창에서 TeamRole-builder 를 검색하고 선택합니다.
3. 화면 상단의 Delete 버튼을 클릭 합니다.
(팝업이 뜨면 Role 이름인 TeamRole-builder 를 입력하고 Delete 버튼을 클릭 합니다)