

Spinnaker 설치하기

이 Spinnaker 설치 가이드 문서는, Spinnaker Version 1.5.x를 대상으로 합니다.

- 1.VPC 등 기본 환경 구축 (10mins)
 - 1-1 Cloud Formation 생성 전 사전 준비
 - 1-2 Cloud Formation Template 다운로드
 - 1-3 Cloud Formation으로 환경 생성하기
- 2. Ansible 및 Jenkins 설치 (15mins)
 - 2-1 Bastion 접속하기 (Windows)
 - PuTTY를 사용하여 Windows에서 Linux 인스턴스에 연결
 - PuTTY를 이용해 Auth Private Key forwarding 하기
 - 2-1 Bastion 접속하기 (Linux & MacOS)
 - 2-2 Ansible / Git / Jenkins 설치하기
 - Jenkins 접속
 - Jenkins 계정 생성
 - Spinnaker가 Jenkins에 접속할 수 있도록 준비하기
- 3. Spinnaker 설치하기 (25 mins)
 - 3-1 Halyard 설치하기 (7 mins)
 - 3-2 Spinnaker가 사용할 Persistent 설정하기
 - User의 Access Key 생성
 - Spinnaker Persistent용 Bucket 설정
 - 3-3 Provider 및 계정 설정
 - 3-4 Spinnaker 설치 하기 (10 Mins)
 - 3-4 Spinnaker 를 외부에서 접근 가능하도록 설정하기
 - 내 PC의 hosts파일 설정 변경하기
 - Public IP 확인
 - Linux & Mac
 - Windows
 - 3-6 Halyard를 이용하여 Jenkins 서버 연동 추가하기
 - 3-7 Spinnaker 서버에 Ansible 설치
 - 3-8 Spinnaker 서버에 Packer Template 추가하기
 - 3-9 내 브라우저에서 접속 하기 & 정상 확인
- 4. Spinnaker Pipeline 구성하기
 - 4-1 Application 생성
 - 4-2 Load Balancer 생성
 - 4-3 Pipeline 생성하기
 - 4-4 Stage 작성하기 - Jenkins
 - 4-5 Stage 작성하기 - Bake
 - Package
 - Template File Name
 - Base OS
 - 4-6 Stage 작성하기 - Deploy
 - Add server group
 - 기본 설정
 - Load Balancer와 Security Group 할당
 - Instance Type 결정
 - 추가 설정
 - Pipeline 실행해보기
 - 4-7 Stage 작성하기 - Enable Server Group
 - 4-8 Stage 작성하기 - Disable Server Group

1.VPC 등 기본 환경 구축 (10mins)

CloudFormation Template을 이용해 Spinnaker 설치 이전 기본적인 환경 구축을 목표로 합니다.

이 CloudFormation에는 VPC, Subnet, InternetGateway와 Bastion Host, Jenkins 용 Instance (미설치상태), Spinnaker용 Instance(미설치상태)를 포함합니다.

1-1 Cloud Formation 생성 전 사전 준비

AWS Console 로그인 > EC2 Console > Network & Security > Key Pairs > Create Key Pair

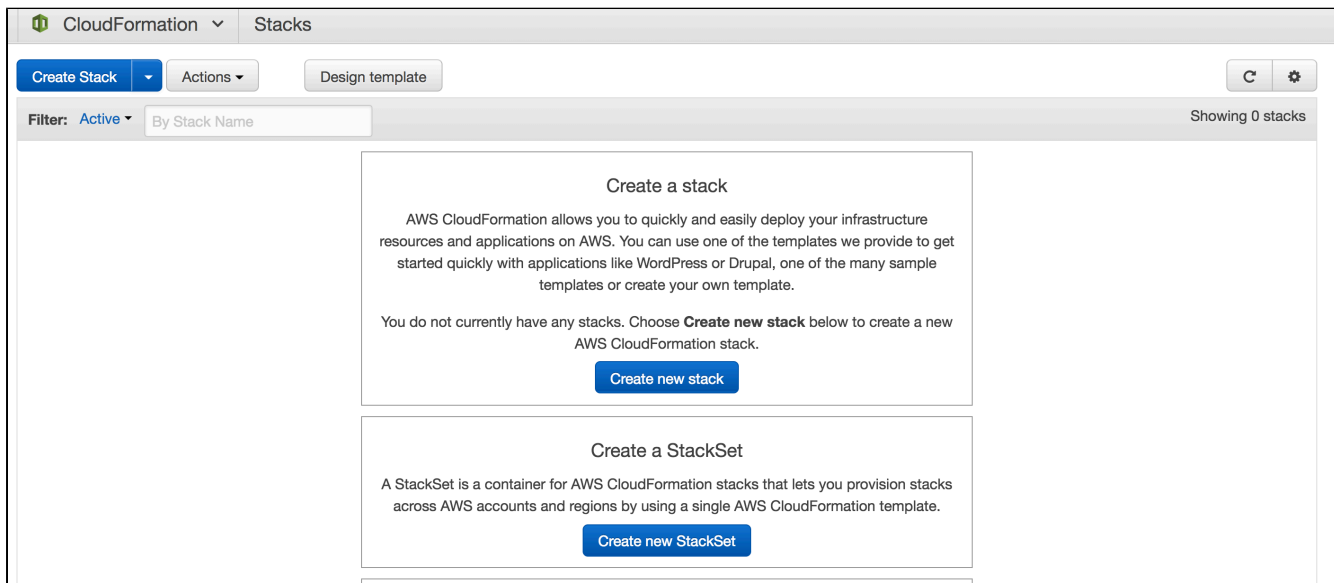
Key Pair Name: spinnaker

1-2 Cloud Formation Template 다운로드

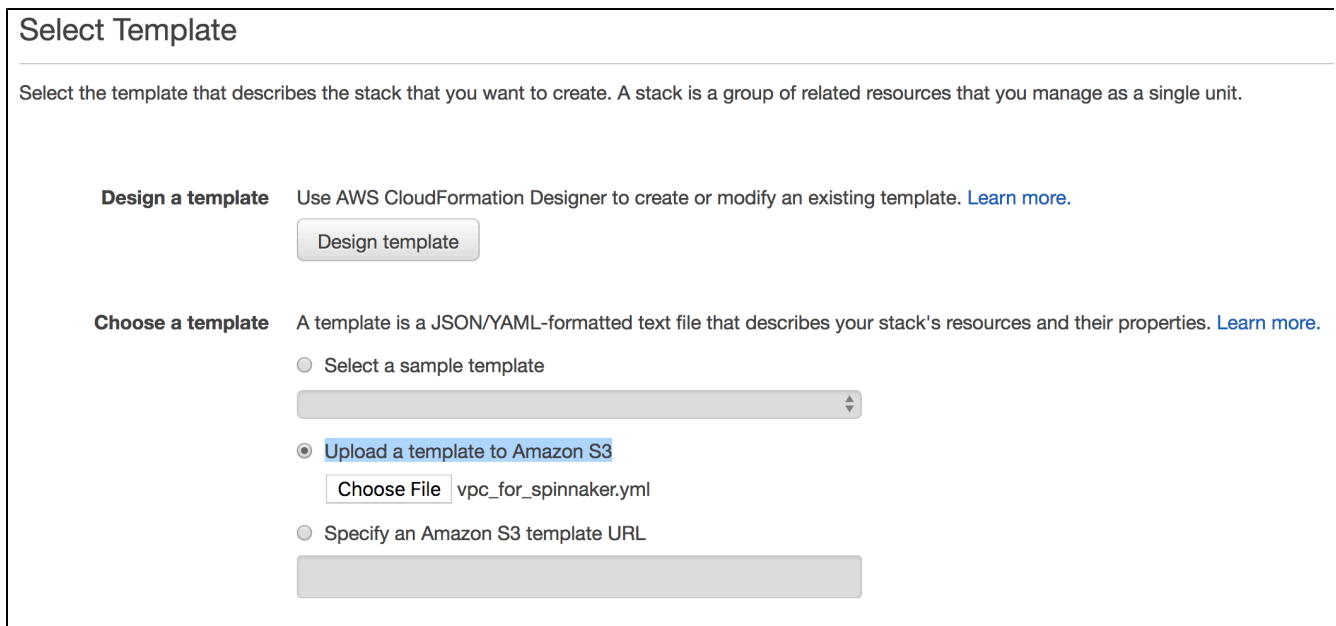
CloudFormation 다운로드 하기: <https://github.com/stevenshim/MyCloudFormation.git>

1-3 Cloud Formation으로 환경 생성하기

CloudFormation Console > Create New Stack 클릭



Choose a template > Upload a template to Amazon S3 > Choose File 선택하여 위에서 다운로드 받은 yaml파일을 지정하여 진행



Specify Details 화면에서 아래와 같이 입력.

- Stack Name: AWSKRUG
- ClusterStack: Spinnaker
- KeyName에는 1-1에서 생성한 KeyName사용
- SSHLocation은 현재 내 IP Address를 입력 (내 IP 조회하기: <https://www.whatismyip.com/>)

Specify Details

Specify a stack name and parameter values. You can use or change the default parameter values, which are defined in the AWS CloudFormation template. [Learn more.](#)

Stack name

Parameters

ClusterStack This value will be used for Resources Name.

KeyName Name of an existing EC2 KeyPair to enable SSH access to the instance

SSHLocation The IP address range that can be used to connect using SSH to the Amazon EC2 instances.

다음 화면 진행 > 추가 설정 없이 Create 진행 (!!! Capabilities에 체크박스 체크 필수)

No rollback triggers provided

Advanced

Notification

Termination Protection Disabled

Timeout none

Rollback on failure Yes

Capabilities

i The following resource(s) require capabilities: [AWS::IAM::Policy, AWS::IAM::Role]

This template contains Identity and Access Management (IAM) resources. Check that you want to create each of these resources and that they have the minimum required permissions. In addition, they have custom names. Check that the custom names are unique within your AWS account. [Learn more.](#)

☒ I acknowledge that AWS CloudFormation might create IAM resources with custom names.

[Quick Create Stack](#) (Create stacks similar to this one, with most details auto-populated)

[Cancel](#) [Previous](#) [Create](#)

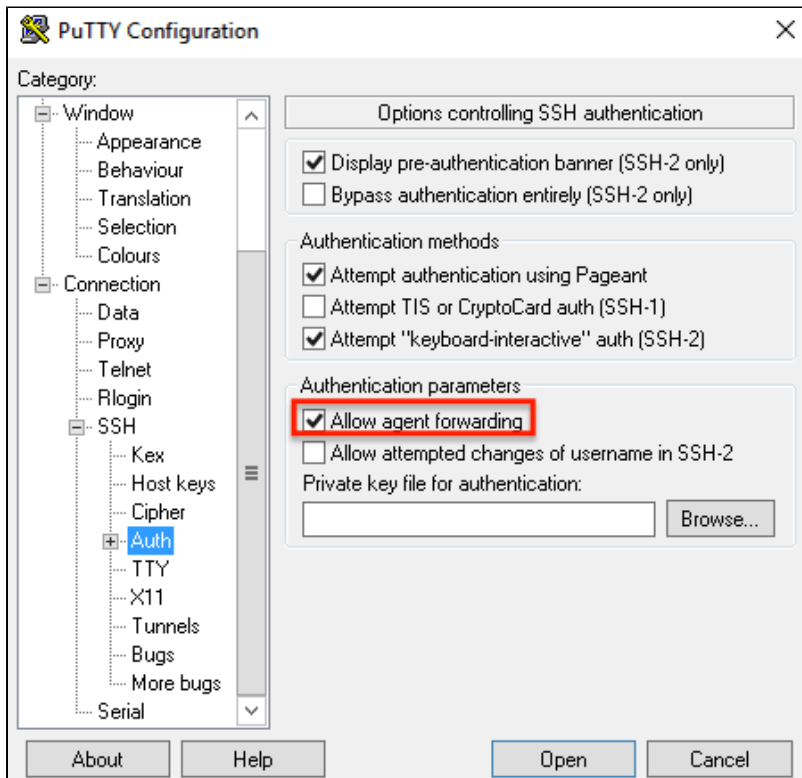
2. Ansible 및 Jenkins 설치 (15mins)

2-1 Bastion 접속하기 (Windows)

PuTTY를 사용하여 Windows에서 Linux 인스턴스에 연결

https://docs.aws.amazon.com/ko_kr/AWSEC2/latest/UserGuide/putty.html

PuTTY를 이용해 Auth Private Key forwarding 하기



2-1 Bastion 접속하기 (Linux & MacOS)

```
## SSH pem Key Forwarding
Local> ssh-add spinnaker.pem

## {BASTION_PUBLIC_IP} Bastion Instance Public IP 53.10.20.30
Local> ssh -A ec2-user@{BASTION_PUBLIC_IP}

## Forwarding
Bastion:~/> ssh-add -L
```

2-2 Ansible / Git / Jenkins 설치하기

```
Bastion:~/> sudo yum update -y

## git
Bastion:~/> sudo yum install git -y

## pip ansible
Bastion:~/> sudo pip install ansible

## Jenkins ansible script
Bastion:~/> git clone https://github.com/stevenshim/ansible-scripts.git

Bastion:~/> cd ansible-scripts

## {YOUR_JENKINS_PRIVATE_IP} Jenkins Instance Private IP .
## !!! IP , ( ) .
## ex) ansible-playbook -i "10.100.1.43," playbooks/jenkins.yml

## Jenkins
Bastion:~/ansible-scripts/> ansible-playbook -i
"{YOUR_JENKINS_PRIVATE_IP}," playbooks/jenkins.yml

## Jenkins
## ex) > ssh 10.100.1.43
Jenkins:~/> ssh {YOUR_JENKINS_PRIVATE_IP}

## Jenkins Password ( . )
Jenkins:~/> sudo cat /data/jenkins/secrets/initialAdminPassword

## Jenkin ssh
Jenkins:~/> exit
```

Jenkins 접속

http://{YOUR_JENKINS_SERVER_IP}:8080/

접속하여 위에서 복사한 password 입력 후 진행

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log ([not sure where to find it?](#)) and this file on the server:

`/data/jenkins/secrets/initialAdminPassword`

Please copy the password from either location and paste it below.

Administrator password

.....

다음 화면에서 Install Suggested Plugin 선택 후 진행 (약 4~5분 소요)

Jenkins 계정 생성

계정정보는 아래와 같이 생성하여 Jenkins 설치 완료

- Username: admin
- Password: admin
- Confirm password: admin
- Full Name: admin
- Email: your email.

Create First Admin User

Username:

Password:

Confirm password:

Full name:

E-mail address:

Spinnaker가 Jenkins에 접속할 수 있도록 준비하기

Jenkins admin 계정으로 로그인 후 아래 메뉴로 이동.

[Manage Jenkins](#) >

[Configure Global Security](#)

화면 중간에 CSRF Protection 에 체크 설정 제거

CSRF Protection

☐ Prevent Cross Site Request Forgery exploits

CSRF를 활성화 할 경우

Spinnaker 1.5.x 버전 이하 버전에 CSRF설정 기능이 정상 동작 하지 않는것 같아 추가 확인이 필요합니다.

Spinnaker 1.6.1 버전에 CSRF설정 기능이 포함되어 있으나, Halyard 명령어가 아직 지원하지 못함. (2018년 2월중에 pull request 진행중으로 알고 있음)

Halyard가 지원하지 않아도 Spinnaker 1.6.1 버전에서 CSRF설정을 하고 싶은 경우

/opt/spinnaker/config/igor.yml을 /opt/spinnaker/config/igor-local.yml로 복사하여 아래와 같이 csrf 설정을 true로 바꿔야 함.

```
jenkins:
  enabled: true
  masters:
    - name: <jenkins master name>
      address: http://<jenkins ip>/jenkins
      username: <jenkins admin user>
      password: <admin password>
      csrf: true
```

3. Spinnaker 설치하기 (25 mins)

3-1 Halyard 설치하기 (7 mins)

Halyard는 Spinnaker의 Microservice중 하나로, Spinnaker Configuration을 수행하고, Spinnaker 의 나머지 Microservice를 배포하는데 사용함.

```
## Spinnaker SSH
## ex) > ssh ubuntu@10.200.100.10
Bastion:~/ansible-scripts/> ssh ubuntu@{SPINNAKER_SERVER_PRIVATE_IP}

## Halyard
ubuntu-Spinnaker:~/> sudo apt-get update

ubuntu-Spinnaker:~/> curl -O
https://raw.githubusercontent.com/spinnaker/halyard/master/install/debian/InstallHalyard.sh

ubuntu-Spinnaker:~/> sudo bash InstallHalyard.sh

ubuntu-Spinnaker:~/> sudo update-halyard

ubuntu-Spinnaker:~/> hal -v
```

3-2 Spinnaker가 사용할 Persistent 설정하기

Spinnaker는 Persistent Storage로 S3, Redis, Minio, Google Cloud Storage, Azure Storage 중 하나를 사용할 수 있다.
Persistent Storage는 Front50에서 Pipeline 정보나 Application 정보 등을 저장하는 DB역할을 한다.
예전 버전의 Spinnaker는 Cassandra를 이용했는데 이제 S3와 같은 Object Storage를 사용하고 있다.

User의 Access Key 생성

AWS Console 로 이동 > IAM > Users > Spinn-user 선택 (이 User는 위 CloudFormation Template을 이용해 미리 생성해놓은 User이다.)
위 User선택 후 Security Credential > Create Access Key 클릭하여 Access Key 발급 (csv파일 Key 다운로드)

Permissions
Groups (0)
Security credentials
Access Advisor

Sign-in credentials

Console password	Disabled Manage password
Console login link	N/A
Last login	None
Assigned MFA device	No
Signing certificates	None

Access keys

Use access keys to make secure REST or HTTP Query protocol requests to AWS service APIs. For your prod frequent key rotation. [Learn more](#)

Create access key

Spinnaker Persistent용 Bucket 설정

```
##      prompt secret-access-key .
ubuntu-Spinnaker:~/> hal config storage s3 edit --region ap-northeast-2
--access-key-id ACCESS_KEY --secret-access-key
```

```
## Persistent S3
ubuntu-Spinnaker:~/> hal config storage edit --type s3
```

3-3 Provider 및 계정 설정


```
## account id AWS ( : 954246081234)
## role/SpinnakerManagedRole CloudFormation .
ubuntu-Spinnaker:~/> hal config provider aws account add spin-awskrug
--account-id {YOUR_AWS_ACCOUNT_ID} --assume-role
role/SpinnakerManagedRole --regions ap-northeast-2

##
ubuntu-Spinnaker:~/> hal config provider aws account list

##
ubuntu-Spinnaker:~/> hal config provider aws account get spin-awskrug
#####
AwsAccount(defaultKeyPair=null, edda=null, discovery=null,
accountId=954256781234,
regions=[AwsProvider.AwsRegion(name=ap-northeast-2)],
assumeRole=role/SpinnakerManagedRole, sessionName=null)

## provider
ubuntu-Spinnaker:~/> hal config provider aws enable
```

3-4 Spinnaker 설치 하기 (10 Mins)

2018년 2월 1.6.0 버전에 이슈 확인.

<https://github.com/spinnaker/orca/pull/2005>

Orca PackageInfo 클래스에 이슈 - 최초 설치 이후에 bake단계에서 이전 Package정보가 없으면 NullPointerException 발생함.

Halyard를 이용해 설정한 값을 가지고 각 Microservice를 배포하는 단계.

```
##
ubuntu-Spinnaker:~/> hal version list

##
ubuntu-Spinnaker:~/> hal config version edit --version 1.5.4

## Spinnaker ( 7~8 )
ubuntu-Spinnaker:~/> sudo hal deploy apply
```

3-4 Spinnaker 를 외부에서 접근 가능하도록 설정하기

Spinnaker는 인증 기능을 활성화 하지 않으면, 외부에서 접근 가능하지 못함.

만약 인증이 필요 없고, Private network에서만 접근 가능한 환경이라면 아래와 같이 설정하여 외부에 오픈함.

```
# ubuntu .
# deck gate override .
# deck apache .
ubuntu-Spinnaker:~/> echo "host: 0.0.0.0" | tee \
    ~/.hal/default/service-settings/gate.yml \
    ~/.hal/default/service-settings/deck.yml

# security override spinnaker.mydomain.org .
ubuntu-Spinnaker:~/> hal config security ui edit \
    --override-base-url http://spinnaker.mydomain.org:9000

# security override spinnaker.mydomain.org .
ubuntu-Spinnaker:~/> hal config security api edit \
    --override-base-url http://spinnaker.mydomain.org:8084
```

내 PC의 hosts파일 설정 변경하기

직접 운영중인 DNS가 있을 경우 원하는 domain 레코드를 등록으로 해도 됩니다.

Public IP 확인

AWS Consoel > EC2 > Instances > Spinnaker-AWSKRUG 선택 후 Public IP확인

search : vpc-2749344f Add filter						
<input type="checkbox"/>	Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks
<input checked="" type="checkbox"/>	Spinnaker-AWSKRUG	i-086622fdaa1a57d81	t2.large	ap-northeast-2a	running	2/2 checks passed
<input type="checkbox"/>	Jenkins	i-04152690b9caa666c	t2.micro	ap-northeast-2a	running	2/2 checks passed
<input type="checkbox"/>	BastionHost	i-0e44810ab0e7c10d1	t2.nano	ap-northeast-2a	running	2/2 checks passed

Instance: i-086622fdaa1a57d81 (Spinnaker-AWSKRUG)		Public DNS: ec2-13-124-8-112.ap-northeast-2.compute.amazonaws.com	
Description	Status Checks	Monitoring	Tags
Instance ID	i-086622fdaa1a57d81	Public DNS (IPv4)	ec2-13-124-8-112.ap-northeast-2.compute.amazonaws.com
Instance state	running	IPv4 Public IP	13.124.8.112

Linux & Mac

```
# /etc/hosts
# ) 53.23.23.23 spinnaker.mydomain.org

SPINNAKER_PUBLIC_IP spinnaker.mydomain.org
```

Windows

```
# C:\windows\system32\drivers\etc\hosts
# ) 53.23.23.23 spinnaker.mydomain.org

SPINNAKER_PUBLIC_IP spinnaker.mydomain.org
```

3-6 Halyard를 이용하여 Jenkins 서버 연동 추가하기

Spinnaker와 CI서버 연동을 하려면 추가 설정을 해야 합니다.

```
ubuntu-Spinnaker:~/> hal config ci jenkins enable

## address Jenkins URL username password    jenkins user/password
## )
## ubuntu-Spinnaker:~/> hal config ci jenkins master add
my-jenkins-master --address http://10.200.100.90:8080 --username admin
--password
##      BaseURL Private IP      .
ubuntu-Spinnaker:~/> hal config ci jenkins master add my-jenkins-master \
--address $BASEURL \
--username $USERNAME \
--password
```

3-7 Spinnaker 서버에 Ansible 설치

아래 명령어는 Bastion Host 서버에서 수행합니다.
Ansible을 이용해 Spinnaker 서버에도 Ansible을 설치합니다.

```
## Bastion Host
## Ansible Spinnaker Ansible
## ) ansible-playbook -i "10.100.10.1," playbooks/ansible.yml

Bastion:~/> ansible-playbook -i "{YOUR_SPINNAKER_PRIVATE_IP},"
playbooks/ansible.yml
```

3-8 Spinnaker 서버에 Packer Template 추가하기

아래 명령어는 Bastion Host 서버에서 수행합니다.
Ansible을 이용해 Spinnaker 서버에 git, packer template, rosco 설정을 설치 및 변경합니다.

```
## Bastion Host
## Ansible Spinnaker git packer template
## ) ansible-playbook -i "10.100.10.1," playbooks/packer_template.yml

Bastion:~/> ansible-playbook -i "{YOUR_SPINNAKER_PRIVATE_IP},"
playbooks/packer_template.yml
```

```
## Spinnaker Spinnaker
ubuntu-Spinnaker:~/> sudo hal deploy apply

## !! . 2 !!
```

3-9 내 브라우저에서 접속 하기 & 정성 확인

!! 스피나커가 구동되기 까지 시간이 조금 걸립니다. !!

<http://spinnaker.mydomain.org:9000/> 접속해보기

상단 Applications 탭 > 우측 Actions > Create Application 선택

아래와 같이 팝업이 바로 정상적으로 출력되어야 함.

New Application

Name *

Enter an application name

Owner Email *

Enter an email address

Repo Type

Select Repo Type

Description

Enter a description

AWS Settings

☐ Prefer AMI Block Device Mappings

Instance Health

☐ Consider only cloud provider health when executing tasks
☐ Show health override option for each operation

Instance Port

80

Pipeline Behavior

☐ Enable restarting running pipelines

* Required

Cancel

Create

4. Spinnaker Pipeline 구성하기

4-1 Application 생성

Spinnaker에 접속(<http://spinnaker.mydomain.org:9000/>)해서 Application을 생성합니다.

상단 Applications > 우측 Actions > New Application 팝업 작성

New Application

Name * TestApp

Owner Email * steven.hj.shim@gmail.com

Repo Type Select Repo Type

Description Enter a description

AWS Settings ☒ Prefer AMI Block Device Mappings

Instance Health ☐ Consider only cloud provider health when executing tasks
☐ Show health override option for each operation

Instance Port 80

Pipeline Behavior ☐ Enable restarting running pipelines

* Required

Cancel Create

4-2 Load Balancer 생성

Spinnaker에서 ELB(Classic)을 만들어보도록 합니다.

Applications > 생성된 Application 선택 > Create Load Balancer 버튼 > Classic (Legacy)

- Account 선택
- Region 선택
- Stack 이름 자유롭게 기재
- VPC Subnet 선택
- Security Group은 CloudFormation으로 자동으로 생성된 SG 사용 (AWSKRUG-ServergroupElbSG)
- Listener 설정은 80 → 80
- Health Check 설정은 tcp ping 80

Location

Your load balancer will be named: testapp-front-elb ⓘ

Account ⓘ

spin-awskrug

Region ⓘ

ap-northeast-2

Stack ⓘ

front



Detail ⓘ

elb

Availability Zones ⓘ

Automatic Availability Zone Balancing:

Enabled

Server group will be available in:

- ap-northeast-2a
- ap-northeast-2c

VPC Subnet ⓘ

awskrug-test

Internal ⓘ

☐ Create an internal load balancer

Security Groups

Security Groups ⓘ

AWSKRUG-ServerGroupElbSG-6WINTCX4W55J (sg-1a85ac71)



Select...

Security groups last refreshed 2018-02-26 05:01:09 PST

If you're not finding a security group that was recently added, [click here](#) to refresh the list.

Listeners

External Protocol	External Port		Internal Protocol	Internal Port	
HTTP	80	→	HTTP	80	
<div> Add new port mapping </div>					

Health Check

Ping	Protocol	Port
	TCP	80

Advanced Settings

Timeout

5

Interval

10

Healthy Threshold

10

Unhealthy Threshold

2

Additional configuration options (idle timeout, cross-zone load balancing, session stickiness, access logs) are available via the AWS console.

4-3 Pipeline 생성하기

상단 PIPELINES 선택 후 > Create > Pipeline 이름 입력

PIPELINES

TestPipe

+

-

Group

Create New Pipeline

Type

Pipeline

Pipeline Name

Copy From

None

Cancel

Create

4-4 Stage 작성하기 - Jenkins

Jenkins Stage 작성하기.

The screenshot shows the Jenkins Pipeline configuration page for a pipeline named 'TestPipe'. At the top, there are buttons for 'Permalink', 'Create', 'Configure', and 'Pipeline Actions'. Below this is a progress bar showing the 'Configuration' stage and a '[new stage]' indicator. There are two buttons: 'Add stage' and 'Copy an existing stage'. Below the progress bar, there is a section for the '[new stage]'. It shows 'No stage type selected'. There are fields for 'Type', 'Stage Name', and 'Depends On'. The 'Type' dropdown is open, showing 'Jenkins' as the selected option, with a description 'Runs a Jenkins job'. There are also buttons for 'Remove stage' and 'Edit stage as JSON'.

여기서 Job 부분은 Jenkins에서 샘플 프로젝트 하나 생성해야 보입니다.

The screenshot shows the Jenkins Configuration page for the 'Jenkins' stage. It has fields for 'Type' (Jenkins), 'Stage Name' (Jenkins), and 'Depends On' (Select...). There are buttons for 'Remove stage' and 'Edit stage as JSON'. Below this is a section titled 'Jenkins Configuration'. It has fields for 'Master' (my-jenkins-master), 'Job' (TestAppBuild), and 'Property File' (empty). There are also checkboxes for 'Wait for results' (checked) and 'If build is unstable' (fail the stage selected, consider stage successful unselected). There are refresh icons next to the Master and Job fields.

4-5 Stage 작성하기 - Bake

Bake 단계는 AMI기반 Instance 단위 배포를 할 때, 배포에 사용할 AMI를 생성하는 단계입니다.

Package

Package는 설치할 package를 명시할 수 있습니다. 이는 Packer script에 전달되어 설치하는데 이용됩니다. 다만, 이게 꼭 필요한 부분은 아닐텐데 현재 버전의 Spinnaker에서는 필수로 입력하는 항목으로 사용합니다.

Package 설치에 Ansible을 통해서도 할 수 있으니, 활용하시기 따라 달라질 것 같습니다.

Template File Name

아래와 같이 Template File Name을 Override 할 경우, rosco.yml설정에 있는 packer 설정 파일을 override하게 됩니다.

즉, 기본 설정 /opt/rosco/config/packer/aws-eks.json이 아니라
변경된 설정 /opt/rosco/config/packer/aws-krug.json을 실행할 수 있도록 합니다.

이는 자신이 원하는 Packer 설정을 활용하고 싶을 때, 추가 packer 설정을 작성하고 override하여 사용하면 됩니다.

☒ **Show Advanced Options**

Template File Name ?

aws-krug.json

Extended Attributes ?

Key	Value	Actions
aws_associate_public_ip_address	false	Remove
aws_region	ap-northeast-2	Remove

이번 실습에서는 아래와 같이 packer가 provisioner로 Ansible을 사용하도록 준비하였습니다.

```
"builders": [{
  "type": "amazon-ebs",
  "access_key": "{{user `aws_access_key`}}",
  "secret_key": "{{user `aws_secret_key`}}",
  "subnet_id": "{{user `subnet_id`}}",
  "vpc_id": "{{user `aws_vpc_id`}}",
  "region": "{{user `aws_region`}}",
  "security_group_id": "{{user `security_group_id`}}",
  "ssh_username": "{{user `aws_ssh_username`}}",
  "ssh_pty": true,
  "instance_type": "{{user `aws_instance_type`}}",
  "source_ami": "{{user `aws_source_ami`}}",
  "ami_name": "{{user `aws_target_ami`}}",
  "associate_public_ip_address": "{{user `aws_associate_public_ip_address`}}",
  "ena_support": "{{user `aws_ena_support`}}",
  "tags": {
    "appversion": "{{user `appversion`}}",
    "build_host": "{{user `build_host`}}",
    "build_info_url": "{{user `build_info_url`}}"
  },
  "run_tags": {"Packages": "{{user `packages`}}"}
}],
"provisioners": [{
  "type": "ansible",
  "user": "ec2-user",
  "playbook_file": "/opt/rosco/config/packer/ansible_krug/playbooks/java.yml",
  "pause_before": "30s"
}]
```

Base OS

Ansible Script가 이미 rosco.yml (/opt/rosco/config/rosco.yml) 설정에 서울 리전의 Bakery Image(aws amazon linux)를 추가해 놓았습니다. Base OS는 미리 rosco.yml에 준비되지 않은 경우에 Bake단계가 정상적으로 진행되지 않습니다.

이상의 점은 Base AMI 및 각 Parameter들을 Override하더라도 Base OS에 먼저 적절한 Bakery 설정이 있지 않으면 동작하지 않습니다.
따라서, 좋은 실드 필요시 사용자 환경에 맞게 rosco.yml에 Bakery설정을 추가한 뒤 진행해야 합니다.

Bake 단계 기본 설정

Bake Configuration

Regions ☒ ap-northeast-2

Package

Base OS

VM Type ☒ HVM ☐ PV

Rebake ☒ Rebake image without regard to the status of any existing bake

☒ **Show Advanced Options**

Template File Name

Extended Attributes	Key	Value	Actions
	aws_associate_public_ip_address	<input type="text" value="false"/>	Remove
	aws_region	<input type="text" value="ap-northeast-2"/>	Remove
	aws_vpc_id	<input type="text" value="vpc-2749344f"/>	Remove
	security_group_id	<input type="text" value="sg-e684ad8d"/>	Remove
	subnet_id	<input type="text" value="subnet-f0692998"/>	Remove
<input type="button" value="+ Add Extended Attribute"/>			

Var File Name

Base AMI (실습에서는 아래 BaseAMI를 비워두고 진행합니다.)

!!!! 실습에서는 아래 BaseAMI를 비워두고 진행 합니다. !!!!

Base AMI는 위 Base OS에서 선택된 값을 Override합니다.

	security_group_id	<input type="text" value="sg-e684ad8d"/>	Remove
	subnet_id	<input type="text" value="subnet-f0692998"/>	Remove
<input type="button" value="+ Add Extended Attribute"/>			
Var File Name	<input type="text"/>		
Base Name	<input type="text"/>		
Base AMI	<input type="text" value="ami-3f248051"/>		
AMI Name	<input type="text"/>		

정확하게는 Spinnaker에서 {{user `aws_source_ami`}} 변수로 사용하는 부분이 Override됩니다.

아래와 같이 Packer 설정 파일(aws-krug.json) 에서 확인할 수 있습니다.

```
builders : List
  {
    "type": "amazon-ecs",
    "access_key": "{{user `aws_access_key`}}",
    "secret_key": "{{user `aws_secret_key`}}",
    "subnet_id": "{{user `aws_subnet_id`}}",
    "vpc_id": "{{user `aws_vpc_id`}}",
    "region": "{{user `aws_region`}}",
    "ssh_username": "{{user `aws_ssh_username`}}",
    "ssh_pty": true,
    "instance_type": "{{user `aws_instance_type`}}",
    "source_ami": "{{user `aws_source_ami`}}",
    "ami_name": "{{user `aws_target_ami`}}",
    "associate_public_ip_address": "{{user `aws_associate_public_ip_address`}}",
    "enable_support": "{{user `aws_enable_support`}}",
    "tags": {
      "appversion": "{{user `appversion`}}",
      "build_host": "{{user `build_host`}}",
      "build_info_url": "{{user `build_info_url`}}",
    },
    "run_tags": {"Packages": "{{user `packages`}}"}
  },
}
```

4-6 Stage 작성하기 - Deploy

Deploy 단계에서는 이전 Bake 단계에서 생성한 AMI를 이용하여 Instance를 생성합니다.

아래 Step과 같이 진행하게 된다면, 이 Deploy단계에서 생성된 Server Group은 ELB로부터 traffic을 받지 않는 상태로 Stage가 완료 됩니다.

Add server group

Type

Deploy

Remove stage

Stage Name

Deploy

Edit stage as JSON

Depends On

Bake

Select...

Deploy Configuration

Account	Cluster	Region	Strategy	Capacity	Actions
Add server group					

기본 설정

아래와 같이 Region, VPC 등을 선택 후 Stack, Detail 내용을 채웁니다.

Basic Settings

Account	spin-awskrug
Region	ap-northeast-2
VPC Subnet ?	awskrug-test
Stack ?	TestStack
Detail ?	Detail
Traffic ?	<input checked="" type="checkbox"/> Send client requests to new instances
Strategy ?	None

Your server group will be in the cluster:
testapp2-TestStack-Detail (new cluster)

Reason

(Optional) anything that might be helpful to explain the reason for this change; HTML is okay

Load Balancer와 Security Group 할당

이번에 Deploy될 Instance가 어떤 Load Balancer에 등록되도록 할지, Security Group은 무엇을 할당 받을지 설정합니다. 이는 곧 Launch Config 와 Auto Scaling Group에 반영되게 됩니다.

Load Balancers

Target Groups ?

No target groups found in the selected account/region/VPC

Classic Load Balancers ?

testapp2-PipeTest-ELB



Select...

Load balancers last refreshed 2018-02-27 02:52:02 PST

If you're not finding a target group or classic load balancer that was recently added, [click here](#) to refresh the list.

Security Groups

Security Groups

AWSKRUG-ServerGroupSG-JN8U3YFKAUI7 (sg-e684ad8d)



Select...

Security groups last refreshed 2018-02-27 02:52:48 PST

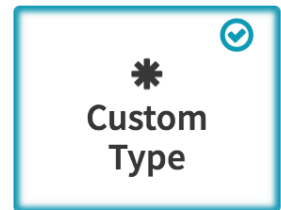
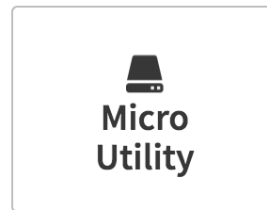
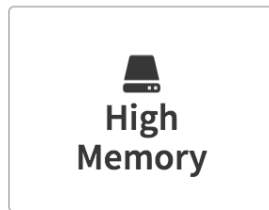
If you're not finding a security group that was recently added, [click here](#) to refresh the list.

Instance Type 결정

Launch Config에 사용할 EC2 Instance Type을 결정합니다.
또한 Auto Scaling Group에 시작시 몇대 생성할지도 같이 결정합니다.

Instance Type

This application is:



t2.micro

Instance types last refreshed 2018-02-27 02:52:02 PST

If you're not seeing an instance type you expect, [click here](#) to refresh the list.

Capacity

Sets min, max, and desired instance counts to the same value.

To allow true auto-scaling, use the [Advanced Mode](#).

Number of Instances


1

Consider deployment successful when percent of instances are healthy.

추가 설정

- Termination Policies: ASG에 사용할 Termination Policies
- Key Name: EC2에 사용할 Key pair 설정
- IAM Instance Profile: Instance에 Role 맵핑
- Scaling Processes
 - AddToLoadBalancer: 미체크시 ELB 에 포함되지 않아 traffic을 받지 않습니다. (중요)

Advanced Settings

Termination Policies	<div>Default </div> <div>Select...</div>
Key Name	<div>spinnaker ▼</div>
Ramdisk Id (optional)	<div></div>
IAM Instance Profile (optional)	<div>BaseIAMRole</div>
UserData (optional) ?	<div></div>
Instance Monitoring ?	<div><input type="checkbox"/> Enable Instance Monitoring</div>
EBS Optimized	<div><input type="checkbox"/> Optimize Instances for EBS</div>
Spot Instances Price (optional) ?	<div></div>
AMI Block Device Mappings	<div><div><input type="radio"/> Copy from current server group ?</div><div><input checked="" type="radio"/> Prefer AMI block device mappings ?</div><div><input type="radio"/> Defaults for selected instance type ?</div></div>
Associate Public IP Address	<div><div><input type="radio"/> Yes</div><div><input type="radio"/> No</div><div><input type="radio"/> Default</div></div>
Scaling Processes	<div><div><input checked="" type="checkbox"/> Launch ?</div><div><input checked="" type="checkbox"/> Terminate ?</div><div><input type="checkbox"/> AddToLoadBalancer ?</div><div><input type="checkbox"/> AlarmNotification ?</div><div><input type="checkbox"/> AZRebalance ?</div><div><input checked="" type="checkbox"/> HealthCheck ?</div><div><input checked="" type="checkbox"/> ReplaceUnhealthy ?</div><div><input checked="" type="checkbox"/> ScheduledActions ?</div></div>

Pipeline 실행해보기

지금까지의 설정으로 Pipeline을 한번 실행해 봅니다.

- Pipelines > Start Manual Execution

Pipelines Create Configure Start Manual Execution

+ - Group by Pipeline Show 10 executions per pipeline ☒ stage durations

SPIN-AWSKRUG **TestPipeline 1** Configure Start Manual Execution

MANUAL START
[anonymous]
a few seconds ago
[Details](#)

Status: **RUNNING** Duration: 00:59

Jenkins Build #4 **Bake** Deploy

STAGE DETAILS: BAKE
Duration: 00:58

Step	Started	Duration	Status
Bake in ap-northeast-2	2018-02-27 19:28:52 PST	00:58	RUNNING
Bake	2018-02-27 19:28:52 PST	00:58	RUNNING

BAKE IN AP-NORTHEAST-2

Bake Config **Task Status**

Image		Base OS	amazon-linux
Region	ap-northeast-2	VM Type	HVM
Package	testapp	Rebake	false
		Template	aws-krug.json

[View Bakery Details](#)

View Bakery Details 클릭 시, Packer 수행 상태를 볼 수 있음.

```
==> amazon-ebs: Prevalidating AMI Name...
amazon-ebs: Found Image ID: ami-863090e8
==> amazon-ebs: Creating temporary keypair: packer_5a9621f4-c991-f392-f208-91052a7a8713
==> amazon-ebs: Launching a source AWS instance...
==> amazon-ebs: Adding tags to source instance
amazon-ebs: Adding tag: "Packages": "testapp"
amazon-ebs: Adding tag: "Name": "Packer Builder"
amazon-ebs: Instance ID: i-062b69cc401544529
==> amazon-ebs: Waiting for instance (i-062b69cc401544529) to become ready...
==> amazon-ebs: Waiting for SSH to become available...
==> amazon-ebs: Connected to SSH!
==> amazon-ebs: Pausing 30s before the next provisioner...
==> amazon-ebs: Provisioning with Ansible...
==> amazon-ebs: Executing Ansible: ansible-playbook --extra-vars packer_build_name=amazon-ebs packer_builder_type=amazon-ebs
amazon-ebs:
amazon-ebs: PLAY [all] *****
amazon-ebs:
amazon-ebs: TASK [Gathering Facts] *****
amazon-ebs: ok: [default]
amazon-ebs:
amazon-ebs: TASK [httpd : install httpd] *****
amazon-ebs: changed: [default] => (item={u'package': u'httpd'})
amazon-ebs:
amazon-ebs: TASK [httpd : create directory httpd document root] *****
amazon-ebs: changed: [default]
amazon-ebs:
amazon-ebs: TASK [httpd : set apache virtual host configuration] *****
amazon-ebs: changed: [default]
amazon-ebs:
amazon-ebs: TASK [httpd : copy index file] *****
amazon-ebs: changed: [default]
amazon-ebs:
amazon-ebs: TASK [httpd : start httpd service] *****
amazon-ebs: changed: [default]
amazon-ebs:
amazon-ebs: PLAY RECAP *****
amazon-ebs: default                : ok=6    changed=5    unreachable=0    failed=0
amazon-ebs:
==> amazon-ebs: Stopping the source instance...
amazon-ebs: Stopping instance, attempt 1
==> amazon-ebs: Waiting for the instance to stop...
```

4-7 Stage 작성하기 - Enable Server Group

4-6 단계에서 생성된 Server Group을 ELB로부터 traffic을 받도록 설정합니다.

- Cluster: Server Group이 있는 Cluster 선택
- Target: Newest Server Group 은 Deploy단계에서 마지막 배포된 Server Group을 의미합니다.

이 단계가 진행되면 4-6에서 생성된 Server Group이 ELB에 포함되게 됩니다.

Type

Enable Server Group

Stage Name

Enable Server Group

Depends On

Deploy

Select...

Remove stage

Edit stage as JSON

Enable Server Group Configuration

Account

spin-awskrug

Regions

☒ ap-northeast-2

Cluster

testapp2-TestStack-Detail

Toggle for text input

Target

Newest Server Group

4-8 Stage 작성하기 - Disable Server Group

기존 버전의 Server Group을 ELB로부터 Traffic 받지 않도록 변경합니다.

- Target: Previous Server Group (이전 버전의 Server Group을 의미)

(* 만약 Server Group 이 하나밖에 없는 경우에는 이 단계가 예러로 끝나게 됩니다.)

Bake

Deploy

Enable Server Group

Disable Server Group

Add stage

Copy an existing stage

Type

Disable Server Group

Stage Name

Disable Server Group

Depends On

Enable Server Group

Select...

Remove stage

Edit stage as JSON

Disable Server Group Configuration

Account

spin-awskrug

Regions

☒ ap-northeast-2

Cluster

testapp2-TestStack-Detail

Toggle for text input

Target

Previous Server Group