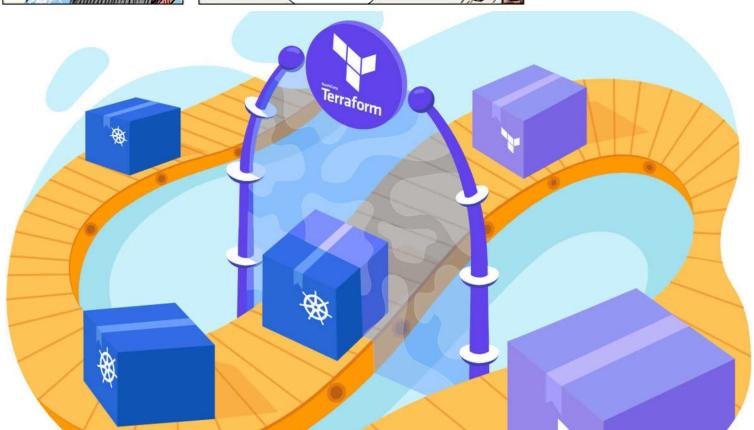
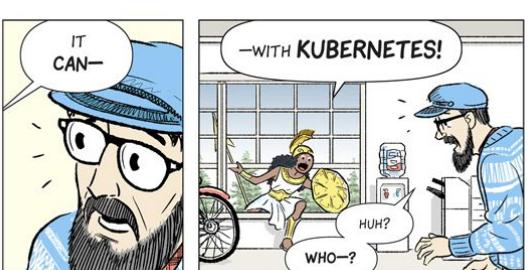
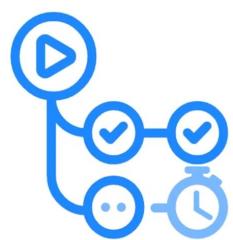
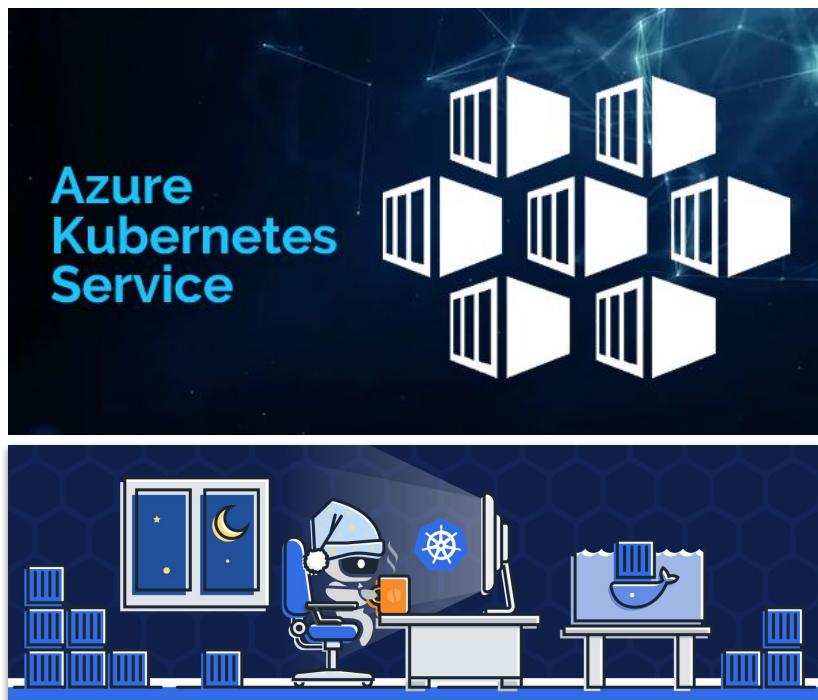


(Praparn Lueangphoonalap)

AKS LifeCycle with GitHub Action



kubernetes
by Google



Agenda

- Rising of cloud-manage for Kubernetes
- GitHub Action for any XKS
 - Purpose of Project
 - Architecture Design
- AKS Cluster
 - Credential as prerequisites
 - Feature on project
- Demo session
- Q&A



kubernetes
by Google

Git Resource

- <https://github.com/praparn/github-action-x-any-xks>

The screenshot shows a GitHub repository page with the following details:

- Repository Name:** praparn / github-action-x-any-xks (Public)
- Code Tab:** Selected
- Branches:** main (1 branch)
- Tags:** 0 tags
- Commits:** 134 commits (by praparn update readme)
- Recent Commits:**
 - .github/workflows/DestroyCluster (8 days ago)
 - azure-aks/Remove unuse elements (7 days ago)
 - img/Remove unuse elements (7 days ago)
 - .gitignore/Create Cluster (8 days ago)
 - LICENSE/Initial commit (10 days ago)
 - README.md/update readme (6 days ago)
- README.md Content:**

GitHub Action for Any XKS

This repository is background to leverage feature of github action and terraform technology for manage in lifecycle of Kubernetes on cloud platform and minimize effort to all contributed for operate Kubernetes cluster (Any XKS). And will continue update

Remark

 - All credential in this project was designed to store on "secrets" on github. If you new with this. Please kindly following [GitHub Secrets](#)
 - As project are coverage multiple cloud provider. So when you fork/clone this repository in action. You can consider to remove unused cloud provider's folder
- About:** Github Action for create any Kubernetes on cloud provider
 - Readme
 - 0 stars
 - 1 watching
 - 1 fork
- Releases:** No releases published [Create a new release](#)
- Packages:** No packages published [Publish your first package](#)
- Environments:** 1
 - production (Failure)
- Languages:** (progress bar)

K8S Lifecycle with Github Action



kubernetes
by Google

Reference

- [Azure AKS Cluster](#)
- [AWS EKS Cluster](#)
- [Google GKE Cluster](#)
- [Huawei Cloud Container Service](#)
- [GitHub Action with Terraform](#)
- [Terraform AKS](#)
- Etc.

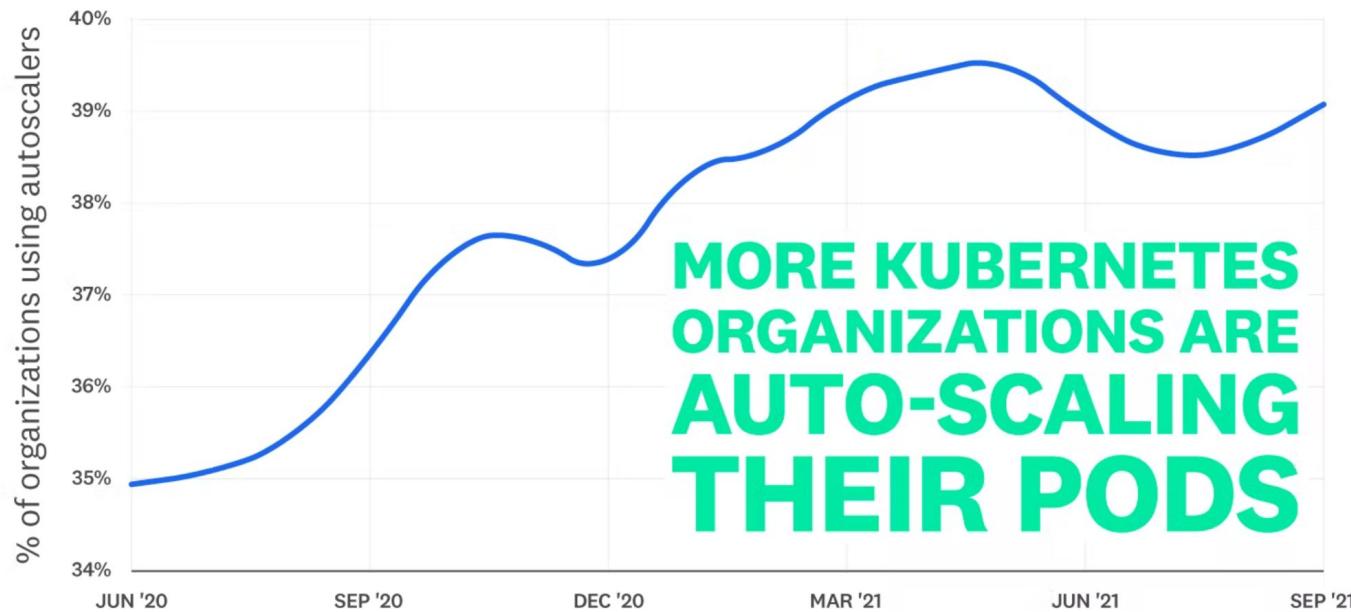
Rising of cloud-manage for Kubernetes



kubernetes
by Google

10 TRENDS IN REAL-WORLD CONTAINER USE

Auto-scaling Usage Over Time



Source: Datadog

Ref: <https://www.datadoghq.com/container-report/>

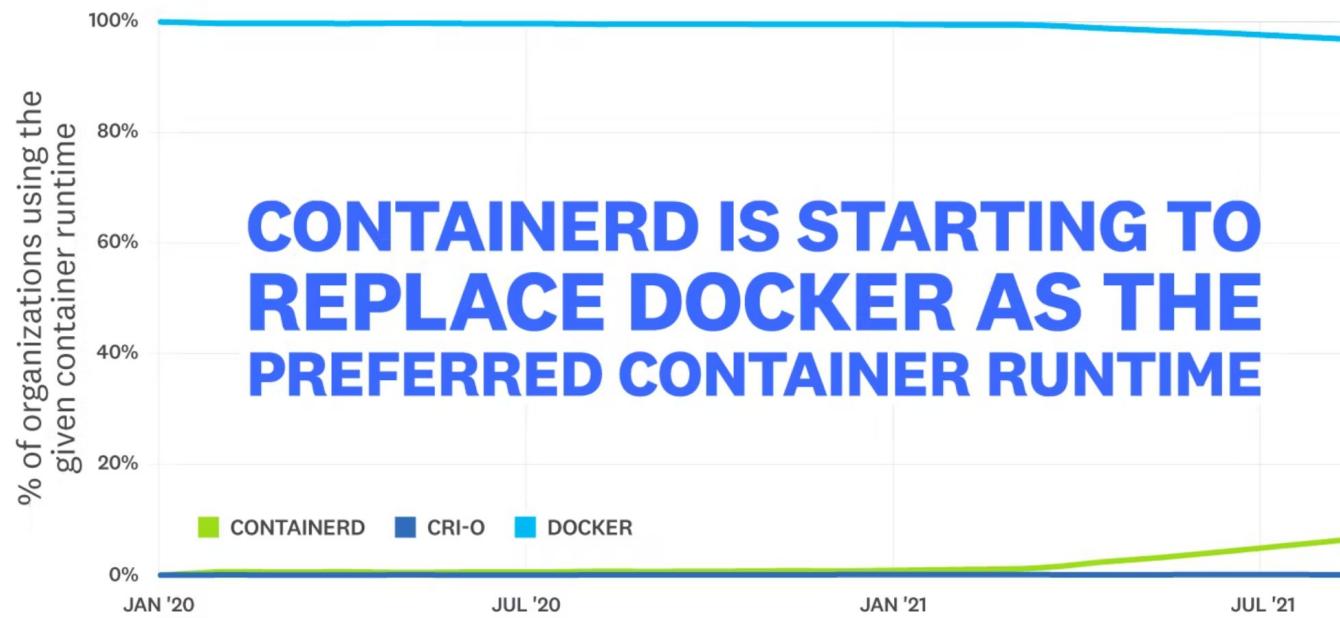
K8S Lifecycle with Github Action



kubernetes
by Google

10 TRENDS IN REAL-WORLD CONTAINER USE

Docker, Containerd, and CRI-O Usage



**CONTAINERD IS STARTING TO
REPLACE DOCKER AS THE
PREFERRED CONTAINER RUNTIME**

Source: Datadog

Ref: <https://www.datadoghq.com/container-report/>

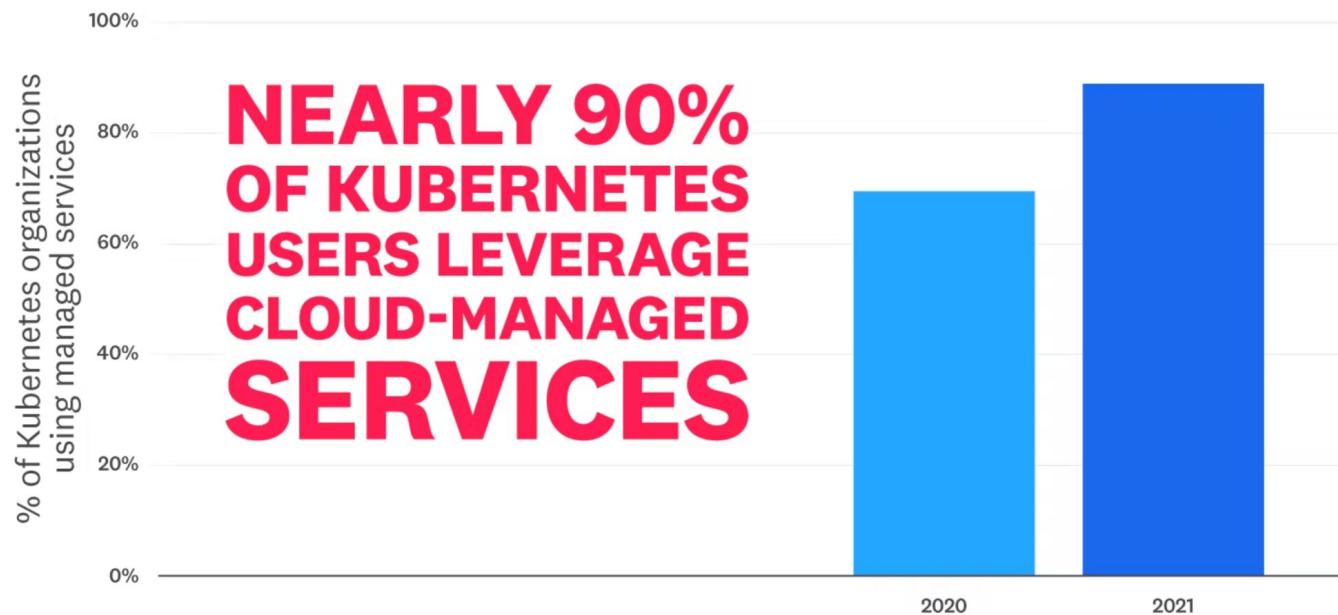
K8S Lifecycle with Github Action



kubernetes
by Google

10 TRENDS IN REAL-WORLD CONTAINER USE

Managed Kubernetes Service Adoption



Source: Datadog

Ref: <https://www.datadoghq.com/container-report/>

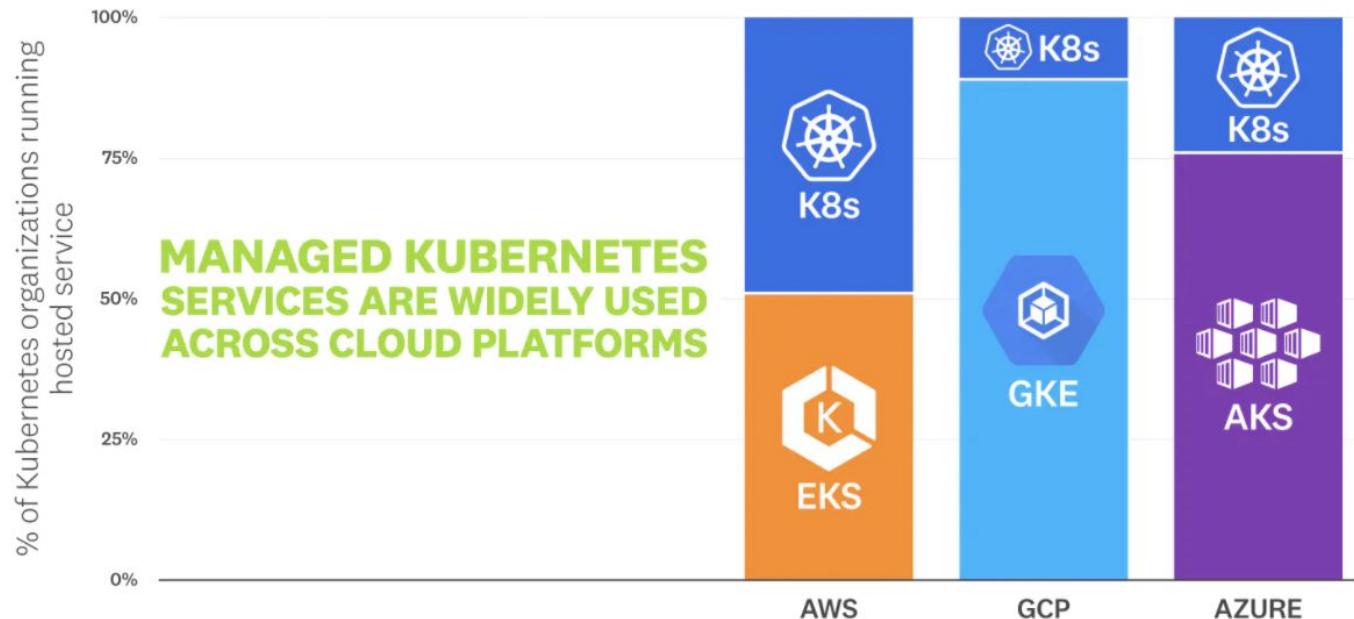
K8S Lifecycle with Github Action



kubernetes
by Google

10 TRENDS IN REAL-WORLD CONTAINER USE

Usage of Managed Kubernetes Services by Cloud Platform



Source: Datadog

Ref: <https://www.datadoghq.com/container-report/>

K8S Lifecycle with Github Action



kubernetes
by Google

GitHub Action for Any XKS



kubernetes
by Google

Purpose of Project

- Kubernetes on cloud provider (Kubernetes as service) is on every where.
- Each cloud provider have specific architecture and solution for manage their cloud provider
- This make complicate for implementor /developer/ devops/etc for handle multiple skill-set and lifecycle...
 - Create/Destroy cluster on cloud environment
 - Hardening / Tuning cluster
 - Upgrade Kubernetes version
 - Network plugin/policy
 - Network range for pods/cluster
 - Sysctl
 - Integrated with facilities on cloud
 - Scaling workload (Increase/Decrease/Autoscale etc)
 - etc.



Purpose of Project

- For our project have main purpose for reduce this complicate for each cloud provider and make all contribute to leverage any “**XKS**” with simple standard
- Project is integrated with “**Terraform**” framework for operate IaC (infrastructure as code) and make same standard on project
- For make it automation part. To make this project help for provision kubernetes cluster automatic. We choose “**Github Action**” as build-in on repository. So this will effort less for operation
- All credential was keep in “**GitHub Secret**” and run via github action. So we not leak any credential to outside



Purpose of Project

- Project will target for handle all Kubernetes platform in cloud provider



Azure Kubernetes Service (AKS)



Google Kubernetes Engine



Amazon EKS

Alibaba Cloud > Products > Container Service for Kubernetes

Alibaba Cloud Container Service for Kubernetes (ACK)

A Kubernetes-based service that ensures high efficiency for enterprises by running containerized applications on the cloud

[Activate Now](#)

[Contact Sales](#)

[Console](#)



HUAWEI CLOUD

Activities Products Solutions Pricing Documentation Marketplace Partners Support About Us

Cloud Container Engine

Cloud Container Engine (CCE) is a fully managed Kubernetes service for you to deploy, manage, and scale containerized applications. CCE supports Kubernetes ecosystem and tooling, allowing you to easily set up a container runtime environment in the cloud.

K8S Lifecycle with Github Action



kubernetes
by Google

Purpose of Project

- Contributor can clone repository (to private repository) and configure properties that need (include credential). After that just push code to your repository and tag “xxxx” and... done !!!

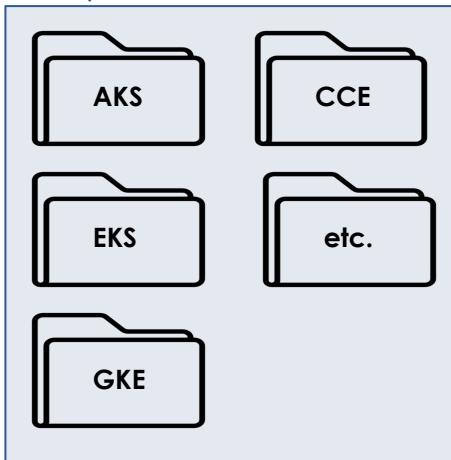


Purpose of Project

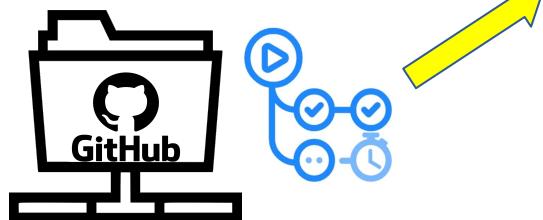


<https://github.com/praparn/github-action-x-any-xks>
Master Repo

Step1: Git clone
(private repository)



Step4: Github Action will
run terraform to create K8S



Step2: Configure
properties and Credential



Step3: Commit and
Push with specific
“Tags”



Step5: Cluster was
created



kubernetes



Developer/
Application
Owner

Step6: Access
cluster and operate

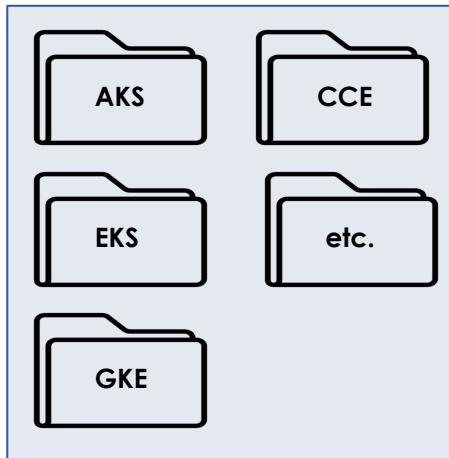


kubernetes

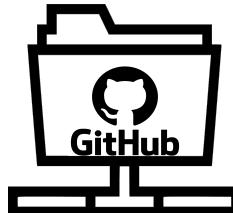
by Google

K8S Lifecycle with Github Action

Architecture Design



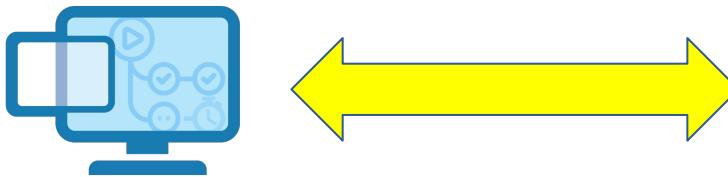
GitHub Secret
<credential>.....



Runner will active with “Tag”

- “xxx-init-env***” (create state file location)
- “xxx-cluster-create***” (create cluster)
- “xxx-cluster-modify***” (modify cluster)
- “xxx-cluster-destroy***” (destroy cluster)
- “xxx-destroy-env***” (destroy state file location)

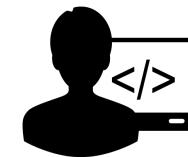
K8S Lifecycle with Github Action



- Step1:** Create space for housing terraform “state file”
Step2: Terraform was configure remote state file as create
Step3: Run terraform for create/modify/delete cluster as design



kubernetes

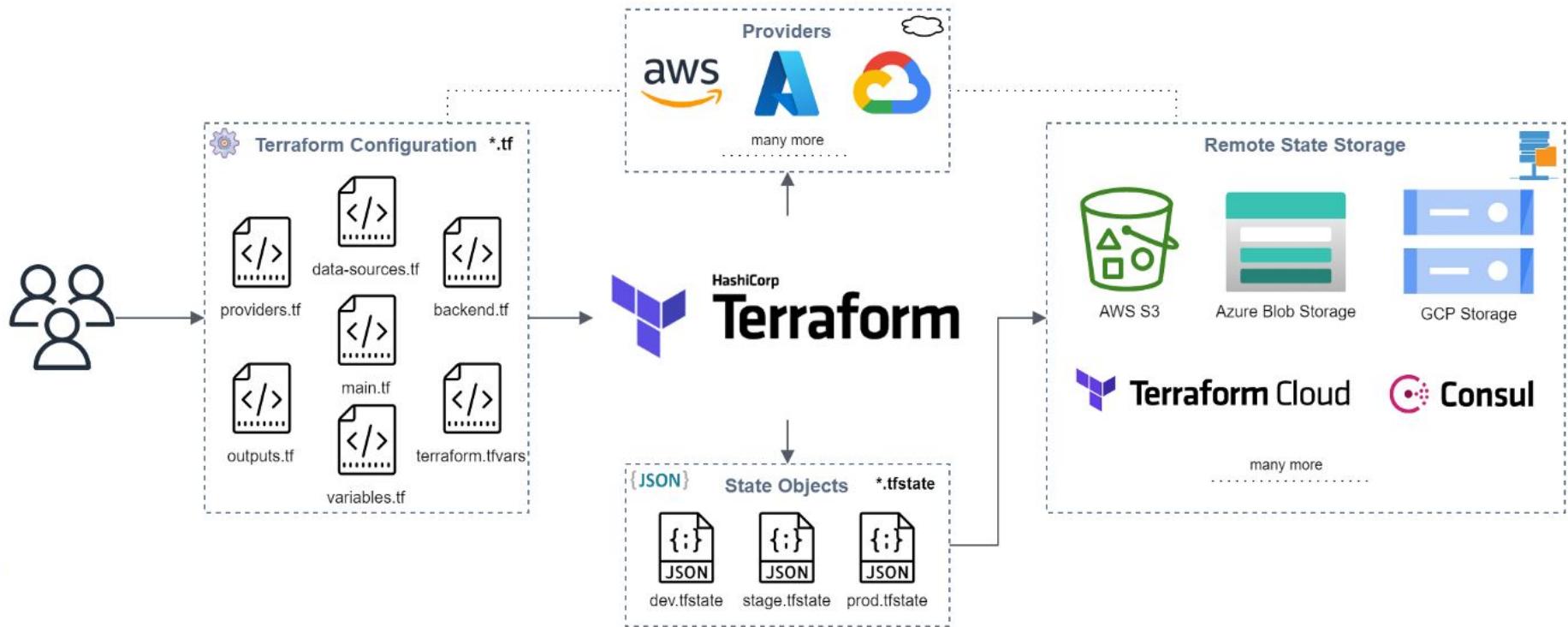


Developer/
Application
Owner



kubernetes
by Google

Architecture Design



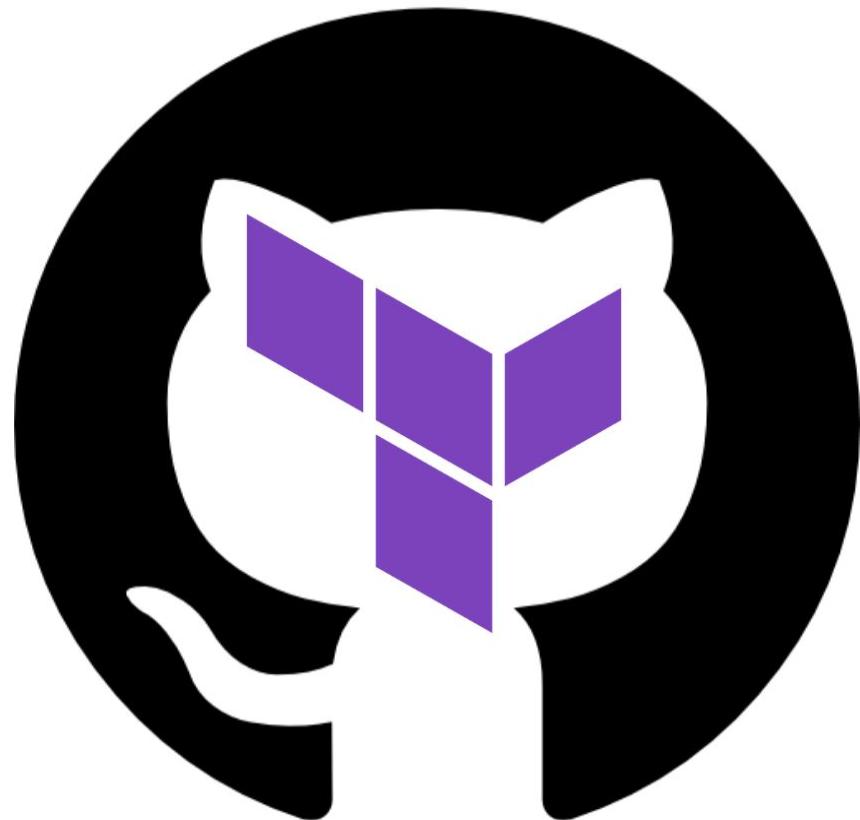
Ref:<https://medium.com/devops-mojo/terraform-remote-states-overview-what-is-terraform-remote-state-storage-introduction-936223a0e9d0>

K8S Lifecycle with Github Action



kubernetes
by Google

AKS Cluster



kubernetes
by Google

Credential as prerequisites

- {AZURE_CREDENTIALS}: Store output in JSON format of your service principle. If you not yet to create service principal. Please follow this KB Azure Service Principal

```
{  
  "clientId": "<GUID>",  
  "clientSecret": "<GUID>",  
  "subscriptionId": "<GUID>",  
  "tenantId": "<GUID>",  
  (...)  
}
```

- {AZURE_CLIENT_ID}: Input client id (You can check this from "{AZURE_CREDENTIALS}")
- {AZURE_CLIENT_SECRET}: Input client secret id (You can check this from "{AZURE_CREDENTIALS}")



Credential as prerequisites

- {AZURE_SUBSCRIPTION_ID}: Input subscription id (You can check this from "{AZURE_CREDENTIALS}")
- {AZURE_TENANT_ID}: Input tenant id (You can check this from "{AZURE_CREDENTIALS}")
- {AZURE_REGION}: Input your region on portal.
Ex:"eastasia" Region code
- {AZURE_RESOURCEGROUP}: Input your resource group name for create other elements
- {AZURE_STORAGEACCOUNT}: Input your storage account name for keep terraform state on portal.
Remark: Storage account name must be between 3 and 24 characters in length and use numbers and lower-case letters only

Credential as prerequisites

- {AZURE_CLUSTERNAME}: Input your AKS cluster name

Actions secrets		
New repository secret		
Secrets are environment variables that are encrypted . Anyone with collaborator access to this repository can use these secrets for Actions.		
Secrets are not passed to workflows that are triggered by a pull request from a fork. Learn more .		
 AZURE_CLIENT_ID	Updated yesterday	Update Remove
 AZURE_CLIENT_SECRET	Updated yesterday	Update Remove
 AZURE_CLUSTERNAME	Updated 13 hours ago	Update Remove
 AZURE_CREDENTIALS	Updated 2 days ago	Update Remove
 AZURE_REGION	Updated 2 days ago	Update Remove
 AZURE_RESOURCEGROUP	Updated 2 days ago	Update Remove
 AZURE_STORAGEACCOUNT	Updated 2 days ago	Update Remove
 AZURE_SUBSCRIPTION_ID	Updated yesterday	Update Remove
 AZURE_TENANT_ID	Updated yesterday	Update Remove



Feature on Project

- **Full life-cycle** for AKS cluster with gitHub action
- **Init-Environment:**
 - Create “resource group” for reference of all element in Azure portal
 - Create “storage account” for housing file
 - Create “tfstate” on storage account
- **Init-Cluster:**
 - Create “Log analytic”
 - Create “Virtual network” and “Subnet”
 - Create “EKS” cluster with custom configuration
 - Create “Ingress Application Gateway” for application
 - Export credential of Kubernetes to file “aks-config” and commit back to git repository
 - Deploy demo application for test cluster



Feature on Project

- **Modify-Cluster:**
 - Upgrade Kubernetes version
 - Increase/Decrease worker node
 - Cluster autoscaling feature
 - Custom configuration
 - etc.
- **Destroy-Cluster:**
 - Delete AKS and resource related
- **Destroy-Environment:**
 - Delete tfstate and blob storage
 - Delete resource group



Feature on Project (Custom Config)

```
#####
#     General Properties      #
#####

variable region {
    default  = "eastasia"
    description = "Region for housing EKS (https://azuretracks.com/2021/04/current-azure-region-names-reference/)"
}

variable location {
    default = "East Asia"
    description = "Full name of region for housing EKS (https://azuretracks.com/2021/04/current-azure-region-names-reference/)"
}

variable environment {
    default = "Development"
    description = "Environment to deploy (Development, UAT, Production)"
}

variable ssh_public_key {
    default = "~/.ssh/id_rsa.pub"
    description = "SSH public key location"
}

variable network_space {
    default = "10.255.0.0/16"
    description = "Total network cidr for cluster"
}

variable network_space_aks {
    default = "10.255.0.0/20"
    description = "Total network cidr for AKS cluster itself (default: 10.255.0.0 - 10.255.15.254 at 4094 hosts)"
}

variable network_space_aksapp {
    default = "10.255.16.0/20"
    description = "Total network cidr for AKS cluster itself (default: 10.255.16.0 - 10.255.31.254 at 4094 hosts)"
}
```



Feature on Project (Custom Config)

```
#####
#     AKS Properties      #
#####

variable cluster_public {
  default = true
  description = "AKS cluster enable access from public or not (true/false)"
}

variable default_node_pool {
  default = "aksnodepool"
  description = "AKS node pool name (Avoid '-')"
}

variable cluster_version {
  default = "1.21.9"
  description = "AKS K8S version (1.21.9/1.22.4/1.22.6)"
}

variable cluster_autoscale {
  default = false
  description = "AKS autoscale"
}

variable cluster_autoscale_max {
  default = null
  description = "AKS maximum autoscale (If not autoscale. This need to be null. (1 - 1000))"
}

variable cluster_autoscale_min {
  default = null
  description = "AKS minimum autoscale (If not autoscale. This need to be null). (1 - 1000)"
}

variable aks_zone {
  default = ["1", "2", "3"]
  description = "AKS available on which zone of Azure Cloud"
}
```

```
variable cluster_httprouting {
  default = false
  description = "AKS enable http routing (true/false). (Recommend enable on DEV environment)"
}

variable "worker_count" {
  default = 3
  description = "AKS total worker node 1 - 1000"
}

variable "worker_os" {
  default = "Ubuntu"
  description = "AKS worker os (Ubuntu/CBLMariner)"
}

variable "worker_maxpods" {
  default = 30
  description = "AKS maximum pods per node #Maximum pods per node is configurable up to 250"
}

variable "worker_encrypt" {
  default = false
  description = "AKS worker node encryption (true/false)"
}

variable "worker_type" {
  default = "VirtualMachineScaleSets"
  description = "AKS type of worker (AvailabilitySet/VirtualMachineScaleSets)"
}

variable "worker_disk_type" {
  default = "Managed"
  description = "AKS worker disk type (Ephemeral/Managed)"
}

variable "worker_size" {
  default = "Standard_D2_v2"
  description = "AKS vmware size for worker node"
}
```



Feature on Project (Custom Config)

```
variable "worker_disksize" {
  default = "30"
  description = "AKS worker disk size (GB)"
}

variable "worker_sysctl_vmswappiness" {
  default = 0
  description = "AKS WORKER TUNNING for vm_swappiness (0 - 100)"
}

variable "worker_sysctl_vmvfscachepressure" {
  default = 50
  description = "AKS WORKER TUNNING for vm_vfs_cache_pressure (0 - 100)"
}

variable "worker_sysctl_fsfilemax" {
  default = 12000500
  description = "AKS worker tuning for fs_file_max (8192 - 12000500)"
}

variable "worker_sysctl_kernelthreadsmax" {
  default = 513785
  description = "AKS worker tuning for kernel_threads_max (20 - 513785)"
}

variable "worker_sysctl_netcoreremdef" {
  default = 134217728
  description = "AKS worker tuning for net_core_rmem_default (212992 - 134217728)"
}

variable "worker_sysctl_netcoreremmax" {
  default = 134217728
  description = "AKS worker tuning for net_core_rmem_max (212992 - 134217728)"
}

variable "worker_sysctl_netcorewmemdef" {
  default = 134217728
  description = "AKS worker tuning for net_core_wmem_default (212992 - 134217728)"}
```

```
variable "worker_sysctl_netcorewmemmax" {
  default = 134217728
  description = "AKS worker tuning for net_core_wmem_max (212992 - 134217728)"
}

variable "worker_sysctl_netipv4tcpfintimeout" {
  default = 15
  description = "AKS worker tuning for net_ipv4_tcp_fin_timeout (5 - 120)"
}

variable "worker_sysctl_netipv4tcpkeepaliveprobes" {
  default = 9
  description = "AKS worker tuning for net_ipv4_tcp_keepalive_probes (1 - 15)"
}

variable "worker_sysctl_netipv4tcpkeepalivetime" {
  default = 7200
  description = "AKS worker tuning for net_ipv4_tcp_keepalive_time (30 - 432000)"
}

variable "worker_sysctlnetipv4tcptwreuse" {
  default = true
  description = "AKS worker tuning for net_ipv4_tcp_tw_reuse (boolean)"
}

#https://docs.microsoft.com/en-us/azure/aks/custom-node-configuration

variable "kubelet_logmaxline" {
  default = 200
  description = "AKS kubelet tuning for max container logs (> 2 line)"
}

variable "kubelet_logmaxmb" {
  default = 200
  description = "AKS kubelet tuning for max container logs before rotated (MB)"
}

variable "kubelet_imagegchighthreshold" {
  default = 60
  description = "AKS kubelet high threshold for start GC image (0% - 100%)"
```



Feature on Project (Custom Config)

```
variable "kubelet_imagegc_lowthreshold" {
  default = 10
  description = "AKS kubelet low threshold for start GC image (0% - 100%)"
}

variable "network_policy" {
  default = "calico"
  description = "AKS network policy for use in cluster. (calico/azure)"
}

variable "network_plugin" {
  default = "kubenet"
  description = "AKS network plugin for cluster. (kubenet/azure)"
}

#https://docs.microsoft.com/en-us/azure/aks/custom-node-configuration

variable "load_balancer_sku" {
  default = "standard"
  description = "AKS SKU for load balancer. (basic/standard)"
}

variable "load_balancer_profile" {
  default = "standard"
  description = "AKS SKU for load balancer profile. (basic/standard)"
}

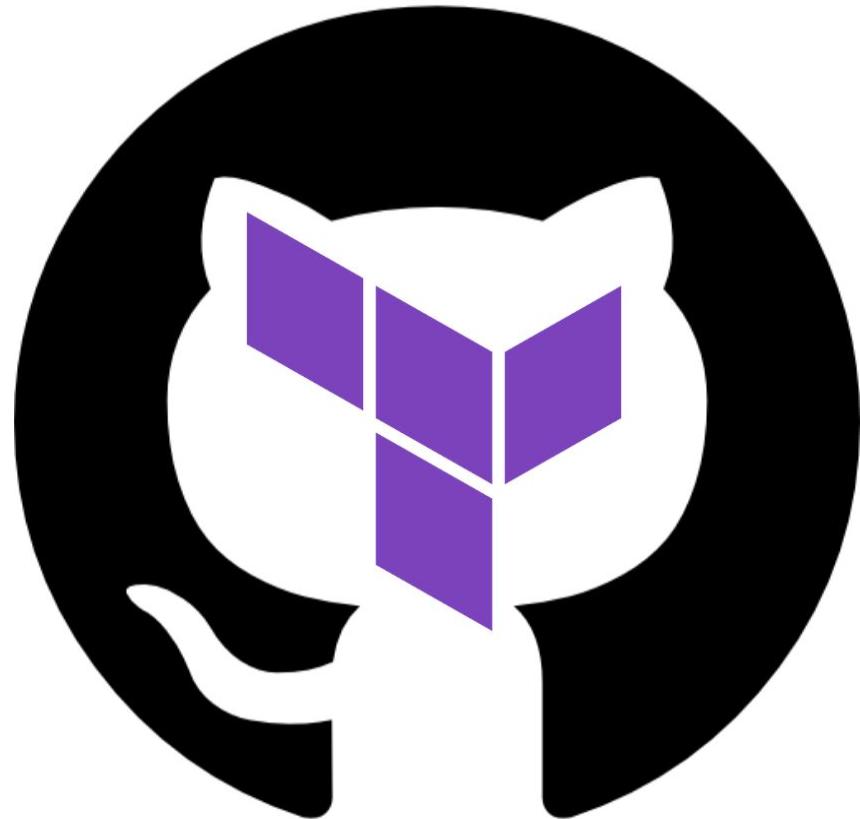
variable log_analytics_workspace_name {
  default = "template-log-aks-lab-2022"
  description = "Log workspace name. This will be add suffix with random number automatically"
}

variable log_analytics_workspace_sku {
  default = "PerGB2018"
  description = "SKU for pricing of log analytics on Azure (https://azure.microsoft.com/en-us/pricing/details/monitor/)"
}

# AKA Properties #
```



Demo Session



K8S Lifecycle with Github Action



kubernetes
by Google

Init Environment

```
github-action-x-any-xks-lab — Terminal MAC Pro — bash — 168x21
praparns-MacBook-Pro:github-action-x-any-xks-lab praparn$ echo "aks-init-env-20220416091138" > ./azure-aks/result/result-env-init
praparns-MacBook-Pro:github-action-x-any-xks-lab praparn$ git add -A
praparns-MacBook-Pro:github-action-x-any-xks-lab praparn$ git commit -m "AKS Cluster Init Environment"
[main 6aab15a] AKS Cluster Init Environment
 1 file changed, 1 insertion(+), 1 deletion(-)
praparns-MacBook-Pro:github-action-x-any-xks-lab praparn$ git tag aks-init-env-20220416091138 -m "aks-init-env-20220416091138"
praparns-MacBook-Pro:github-action-x-any-xks-lab praparn$ git push --atomic origin main aks-init-env-20220416091138
Enumerating objects: 10, done.
Counting objects: 100% (10/10), done.
Delta compression using up to 8 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (6/6), 605 bytes | 605.00 KiB/s, done.
Total 6 (delta 3), reused 0 (delta 0)
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To https://github.com/praparn/github-action-x-any-xks-lab.git
 5ac62f9..6aab15a  main -> main
 * [new tag]      aks-init-env-20220416091138 -> aks-init-env-20220416091138
praparns-MacBook-Pro:github-action-x-any-xks-lab praparn$ █
```

1. Clone the repo

```
git clone https://github.com/praparn/github-action-x-any-xks.git
```

2. Setup "secrets" elements as explain in Prerequisites

3. Create Azure environment by commit and tag "aks-init-env"

```
git pull
git tag #check tag duplicate
echo "aks-init-env-yyyymmddhhmmss" > ./azure-aks/result/result-env-init
git add -A
git commit -m "Any command that you need"
git tag aks-init-env-yyyymmddhhmmss -m "aks-init-env-yyyymmddhhmmss"
git push --atomic origin <branch name> aks-init-env-yyyymmddhhmmss
```

4. Check progress on tab "action" until it finished. (Optional: Verify result on web console/cli for double check)

Init Environment AKS initial basic element on Azure Portal #16

Re-run all jobs ...



Create Cluster

The image shows a network diagram with three nodes: C (green), F (purple), and B (blue). Node C is at the bottom left, node F is at the top right, and node B is at the bottom right. They are connected by curved lines of different colors: purple, green, and blue. The background features a screenshot of a Mac OS X terminal window titled "azure-aks — Terminal MAC Pro — bash — 100x7". The terminal displays the following command output:

```
[praparns-MacBook-Pro:azure-aks praparn$ kubectl get node --kubeconfig=./aks-config
NAME           STATUS   ROLES     AGE    VERSION
aks-aksnodepool-93106026-vmss000000  Ready    agent    160m   v1.21.9
[praparns-MacBook-Pro:azure-aks praparn$ kubectl get ing -n=management-ui --kubeconfig=aks-config
NAME          CLASS      HOSTS   ADDRESS        PORTS   AGE
management-ui-ingress <none>    *    20.187.168.169   80      4m20s
praparns-MacBook-Pro:azure-aks praparn$ ]
```

Modify Cluster

The screenshot shows a GitHub Actions log for a workflow named "modify-cluster". The log output is as follows:

```
Started 43s ago

104     ~ kubernetes_version          = "1.21.9" -> "1.22.4"
105     name                          = "***"
106     tags                           =
107         "Environment" = "Development"
108     }
109     # (20 unchanged attributes hidden)
110     ~ default_node_pool {
111         name                  = "aksnodepool"
112         ~ orchestrator_version = "1.21.9" -> "1.22.4"
113         tags                  = {}
114         # (20 unchanged attributes hidden)
115         # (2 unchanged blocks hidden)
116     }
117     # (5 unchanged blocks hidden)
118   }
119 Plan: 0 to add, 1 to change, 0 to destroy.
120 -----
121 Saved the plan to: tfplan
122 To perform exactly these actions, run the following command to apply:
123   terraform apply "tfplan"
124 Releasing state lock. This may take a few moments...
```

A dropdown menu titled "AKS update cluster configuration" is open, showing the following options:

- Run terraform apply -input=false tfplan
- /home/runner/work/_temp/e9cd3f43-4c6b-49c9-bcf8-c552e893ad2a/terraform-bin apply -input=false tfplan
- Acquiring state lock. This may take a few moments...

Below the dropdown, there are four other options:

- AKS export output
- Export configuration (KUBECONFIG)
- Commit and push result back
- Post Git checkout code



Destroy Cluster

The screenshot shows a GitHub Actions job named "destroy-cluster" running on a Mac. The job has started 27 seconds ago. It consists of several steps:

- Set up job
- Git checkout code
- Setup system configuration
- Load Terraform to operate
- AKS readiness verify
- AKS initialize
- AKS validation configuration step1
- AKS destroy farm (This step is currently active, showing log output)

The log output for the AKS destroy farm step includes the following entries:

```
17 Acquiring state lock. This may take a few moments...
18 random_id.log_analytics_workspace_name_suffix: Refreshing state... [id=7zzfkBbehlw]
19 azurerm_log_analytics_workspace.lab: Refreshing state...
[id=subscriptions/**/resourceGroups/**/providers/Microsoft.OperationalInsights/workspaces/**-wrkspc-17238899283619382876]
20 azurerm_virtual_network.virtual_network: Refreshing state...
[id=subscriptions/**/resourceGroups/**/providers/Microsoft.Network/virtualNetworks/**-vnet]
21 azurerm_subnet.aks_subnet: Refreshing state... [id=/subscriptions/**/resourceGroups/**/providers/Microsoft.Network/virtualNetworks/**-vnet/subnets/**-aksvnet]
22 azurerm_log_analytics_solution.lab: Refreshing state...
[id=/subscriptions/**/resourceGroups/**/providers/Microsoft.OperationsManagement/solutions/ContainerInsights(**-wrkspc-17238899283619382876)]
23 azurerm_kubernetes_cluster.k8s: Refreshing state...
[id=/subscriptions/**/resourceGroups/**/providers/Microsoft.ContainerService/managedClusters/**]
```

At the bottom of the job details, there are three options:

- Rollback all Propertie KUBECONFIG
- Commit and push result back
- Post Git checkout code



Destroy Environment

praparn / [github-action-x-any-xks](#) Private

Code Issues Pull requests Actions Projects Security Insights Settings

DestroyCluster AKS destroy cluster #13 Re-run all jobs ...

Summary

Jobs

destroy-cluster succeeded 4 minutes ago in 25s

Search logs

Step	Description	Duration
> ✓	Set up job	1s
> ✓	Git checkout code	1s
> ✓	Setup system configuration	0s
> ✓	Load Terraform to operate	0s
> ✓	AKS readiness verify	0s
> ✓	AKS initialize	7s
> ✓	AKS validation configuration step1	2s
> ✓	AKS destroy farm	9s
> ✓	Rollback all Propertie KUBECONFIG	0s
> ✓	Commit and push result back	0s
> ✓	Post Git checkout code	0s
> ✓	Complete job	0s

K8S Lifecycle with Github Action



kubernetes
by Google

Q&A

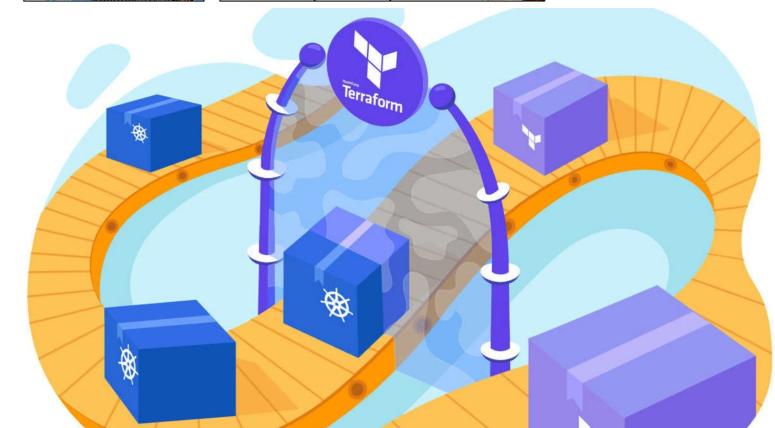
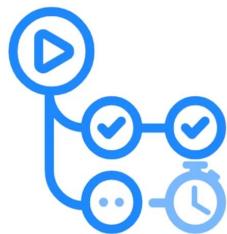


K8S Lifecycle with Github Action



kubernetes
by Google

Thanks You



kubernetes
by Google