

Use GitOps for cluster configuration

For use with Azure Arc enabled Kubernetes Demo Guides

By [Philippe Beraud](#), Microsoft France

This walkthrough is extrapolated from [Use GitOps for an Azure Arc-enabled configuration \(Preview\)](#) and [Use GitOps with Helm for an Azure Arc-enabled cluster configuration \(Preview\)](#).

This information pertains to the Public Preview of Azure Arc enabled Kubernetes.

Note For additional information, see [What is Azure Arc-enabled Kubernetes \(Preview\)](#).

To setup the GitOps configuration, you will need access to an Azure subscription, at least one (local) Kubernetes cluster not running in Azure, and a Git repo where the cluster configuration to apply resides.

You will use here the sample cluster configuration from [https://github.com/slack\(cluster-config\)](https://github.com/slack(cluster-config)), courtesy of [Jason Hansen](#), or its evolution, the now official sample for Azure Arc enabled Kubernetes: <https://github.com/Azure/arc-k8s-demo>.

A “word” about the GitOps workflow

This architecture uses a GitOps workflow to configure the cluster and deploy applications.

Note Since its inception in 2017 by [Weaveworks](#), GitOps has caused quite some fuss on Twitter and KubeCon. As such, GitOps is a way of implementing Continuous Deployment (CD) for cloud-native applications. It focuses on a developer-centric experience when operating infrastructure, by using tools developers are already familiar with, including Git and Continuous Deployment (CD) tools.

For more information, see [What you need to know - Guide To GitOps](#). See also [GitOps](#).

Configuration is described declaratively and stored in Git. An agent watches the Git repo for changes and applies them.

The connection between your Kubernetes cluster and one or more git repositories (for multi-tenancy and separation of duties) is tracked in Azure Resource Manager (ARM) as a `sourceControlConfiguration` extension resource. The `sourceControlConfiguration` resource properties represents where and how Kubernetes resources should flow from Git to your cluster.

The Azure Arc for Kubernetes `config-agent` running in your cluster is responsible for watching for new or updated `sourceControlConfiguration` resources and orchestrates adding, updating, or removing the git repo links automatically.

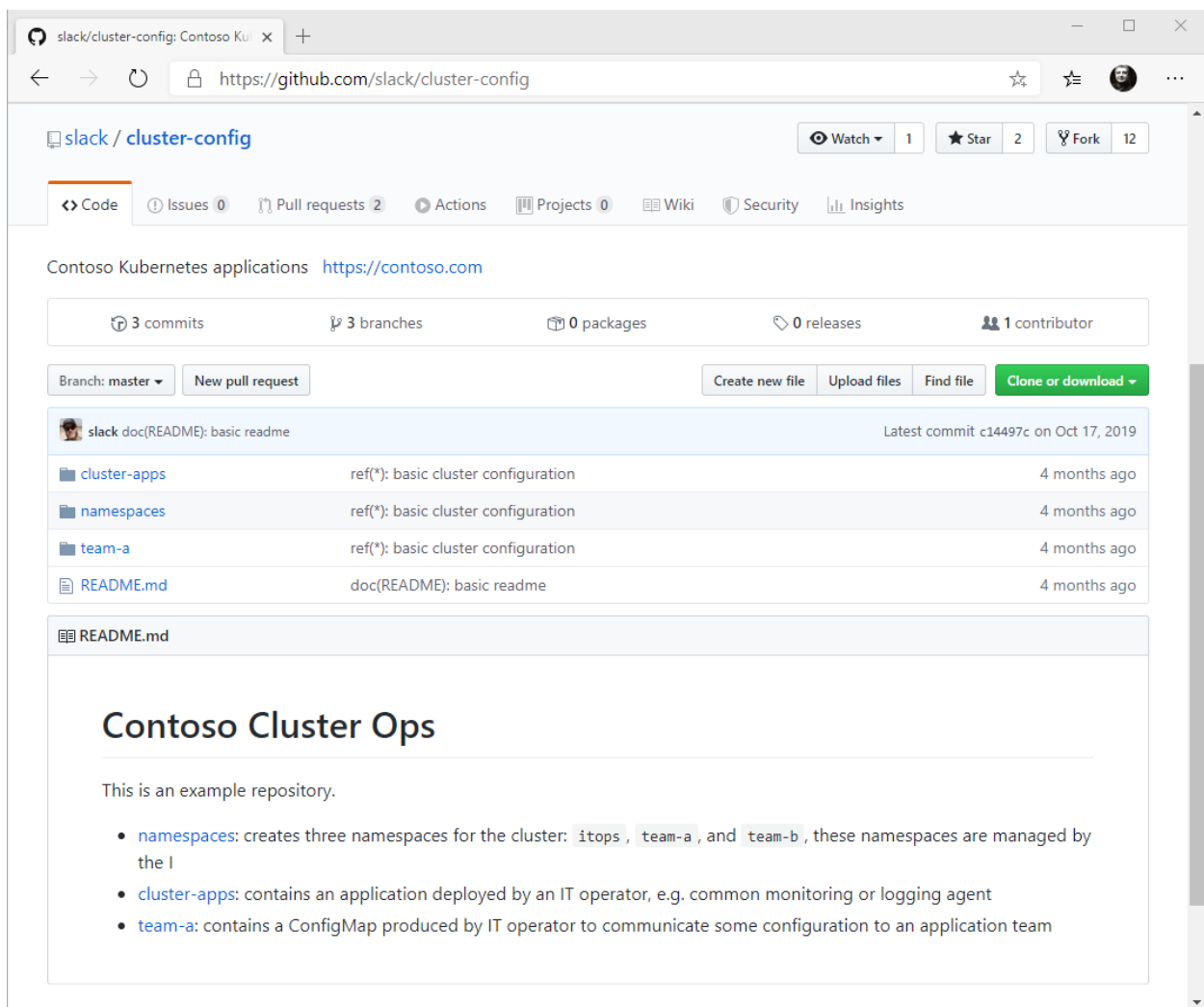
The same patterns can be used to manage a larger collection of clusters, which may be deployed across heterogenous environments in multi-clouds, in the edge, and/or on-premises. For example, you may have one repository that defines baseline configuration for your organization and apply that to tens of Kubernetes clusters at once.

The git repository can contain any valid Kubernetes resources including Namespaces, ConfigMaps, Deployments, DaemonSets, etc. A common set of scenarios include defining a baseline configuration for your organization, which might include common RBAC roles and bindings, monitoring or logging agents, or cluster-wide services.

This guide will walk you through applying a set of configurations with cluster-admin scope.

Creating a cluster configuration

For the sake of this illustration, and as stated above, you will use the following example cluster configuration repository: [https://github.com/slack\(cluster-config](https://github.com/slack(cluster-config) (you can use instead the new repository: <https://github.com/Azure/arc-k8s-demo>.)



The screenshot shows the GitHub repository page for 'slack / cluster-config'. The repository has 3 commits, 3 branches, 0 packages, 0 releases, and 1 contributor. The README file is selected, showing the following content:

Contoso Cluster Ops

This is an example repository.

- namespaces:** creates three namespaces for the cluster: `itops`, `team-a`, and `team-b`, these namespaces are managed by the I
- cluster-apps:** contains an application deployed by an IT operator, e.g. common monitoring or logging agent
- team-a:** contains a ConfigMap produced by IT operator to communicate some configuration to an application team

This example repository is structured around the persona of a cluster operator who would like to provision a few namespaces, deploy a common workload, and provide some team-specific configuration.

Using this repository will result in creating the following resources on your Kubernetes cluster:

- Namespaces: `cluster-config`, `team-a`, `team-b`

- Deployment: cluster-config/podinfo
- ConfigMap: team-a/endpoints

Using Azure CLI

Let's see how to use the Azure CLI extension for k8sconfiguration. You will link your connected cluster to the above example git repository.

You will give this configuration a name, for example, cluster-config in our illustration, instruct the agent to deploy the operator in the cluster-config namespace, and give the operator cluster-admin permissions.

For a connected cluster, we assume that you previously performed all the steps outlined in the document [Prepare your Kubernetes environment for Azure](#). In particular, Azure CLI is installed and the k8sconfiguration CLI extension is properly registered.

Proceed with the following steps:

1. Open a CLI.
2. Connect to Azure CLI if you haven't already done so:

```
$ az login
```

When asked for, open a browser session on your Windows 10 host machine and navigate to at <https://aka.ms/devicelogin>.

- a. Enter the authorization code displayed in your Bash terminal console.
 - b. Sign in with your account credentials in the browser.
2. If you have multiple Azure subscriptions, select the subscription in which your cluster has been onboarded.

```
$ az account set -s <subscription_id>
```

Where <subscription_id> is the ID of your subscription retrieve from the Azure portal.

3. Use the Azure CLI extension for k8sconfiguration - Specify your cluster name in lieu of AzureArcMinikubeK8s1, do the same for your actual resource group -:

```
$ CLUSTER_NAME=AzureArcMinikubeK8s1
$ GROUP_NAME=AzureArcK8sTest
$ az k8sconfiguration create \
  --name cluster-config \
  --resource-group ${GROUP_NAME} \
  --cluster-name ${CLUSTER_NAME} \
  --operator-instance-name cluster-config \
  --operator-namespace cluster-config \
  --repository-url git://github.com/slack/cluster-config.git \
  --enable-helm-operator true \
  --helm-operator-chart-version 0.3.0
```

Command group 'k8sconfiguration' is in preview. It may be changed/removed in a future release.

```
{
  "complianceStatus": {
    "complianceState": "Pending",
```

```
"lastConfigApplied": "0001-01-01T00:00:00",
"message": "{\\"OperatorMessage\\":null,\\"ClusterState\\":null}",
"messageLevel": "3"
},
"enableHelmOperator": "True",
"helmOperatorProperties": {
  "chartValues": "",
  "chartVersion": "0.3.0"
},
"id": "/subscriptions/2f291b88-9ebf-4d90-836f-ae532edaa1b0/resourceGroups/AzureArcK8sTest/providers/Microsoft.Kubernetes/connectedClusters/AzureArcMinikubeK8s1/providers/Microsoft.KubernetesConfiguration/sourceControlConfigurations/cluster-config",
"name": "cluster-config",
"operatorInstanceName": "cluster-config",
"operatorNamespace": "cluster-config",
"operatorParams": "--git-readonly",
"operatorScope": "cluster",
"operatorType": "Flux",
"provisioningState": "Succeeded",
"repositoryPublicKey": "",
"repositoryUrl": "git://github.com/slack/cluster-config.git",
"resourceGroup": "AzureArcK8sTest",
"type": "Microsoft.KubernetesConfiguration/sourceControlConfigurations"
}
```

```

philber@PHILBER001: ~/azure-arc-kubernetes-preview
philber@PHILBER001:~/azure-arc-kubernetes-preview$ CLUSTER_NAME=AzureArcMinikubeK8s1
philber@PHILBER001:~/azure-arc-kubernetes-preview$ GROUP_NAME=AzureArcK8sTest
philber@PHILBER001:~/azure-arc-kubernetes-preview$ az k8sconfiguration create \
> --name cluster-config \
> --resource-group ${GROUP_NAME} \
> --cluster-name ${CLUSTER_NAME} \
> --operator-instance-name cluster-config \
--operator-namespace cluster-config \
> --repository-url git://github.com/slack/ccluster-config.git \
> --enable-helm-operator true \
> --helm-operator-chart-version 0.3.0
Command group 'k8sconfiguration' is in preview. It may be changed/removed in a future release.
{
  "complianceStatus": {
    "complianceState": "Pending",
    "lastConfigApplied": "0001-01-01T00:00:00",
    "message": "{\\\"OperatorMessage\\\":null,\\\"ClusterState\\\":null}",
    "messageLevel": "3"
  },
  "enableHelmOperator": "True",
  "helmOperatorProperties": {
    "chartValues": "",
    "chartVersion": "0.3.0"
  },
  "id": "/subscriptions/2f291b88-9ebf-4d90-836f-ae532edaa1b0/resourceGroups/AzureArcK8sTest/providers/Microsoft.KubernetesConfiguration/sourceControlConfigurations/cluster-config",
  "name": "cluster-config",
  "operatorInstanceName": "cluster-config",
  "operatorNamespace": "cluster-config",
  "operatorParams": "--git-readonly",
  "operatorScope": "cluster",
  "operatorType": "Flux",
  "provisioningState": "Succeeded",
  "repositoryPublicKey": "",
  "repositoryUrl": "git://github.com/slack/ccluster-config.git",
  "resourceGroup": "AzureArcK8sTest",
  "type": "Microsoft.KubernetesConfiguration/sourceControlConfigurations"
}
philber@PHILBER001:~/azure-arc-kubernetes-preview$

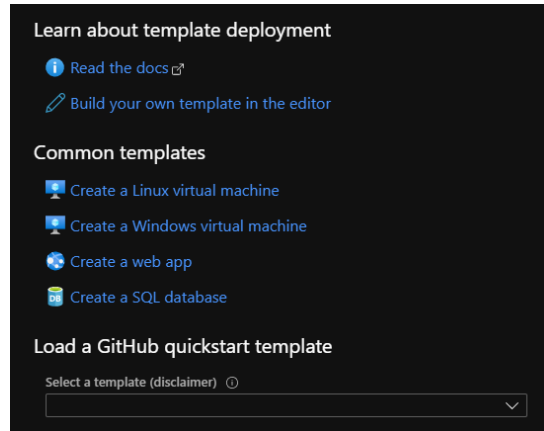
```

Using ARM template

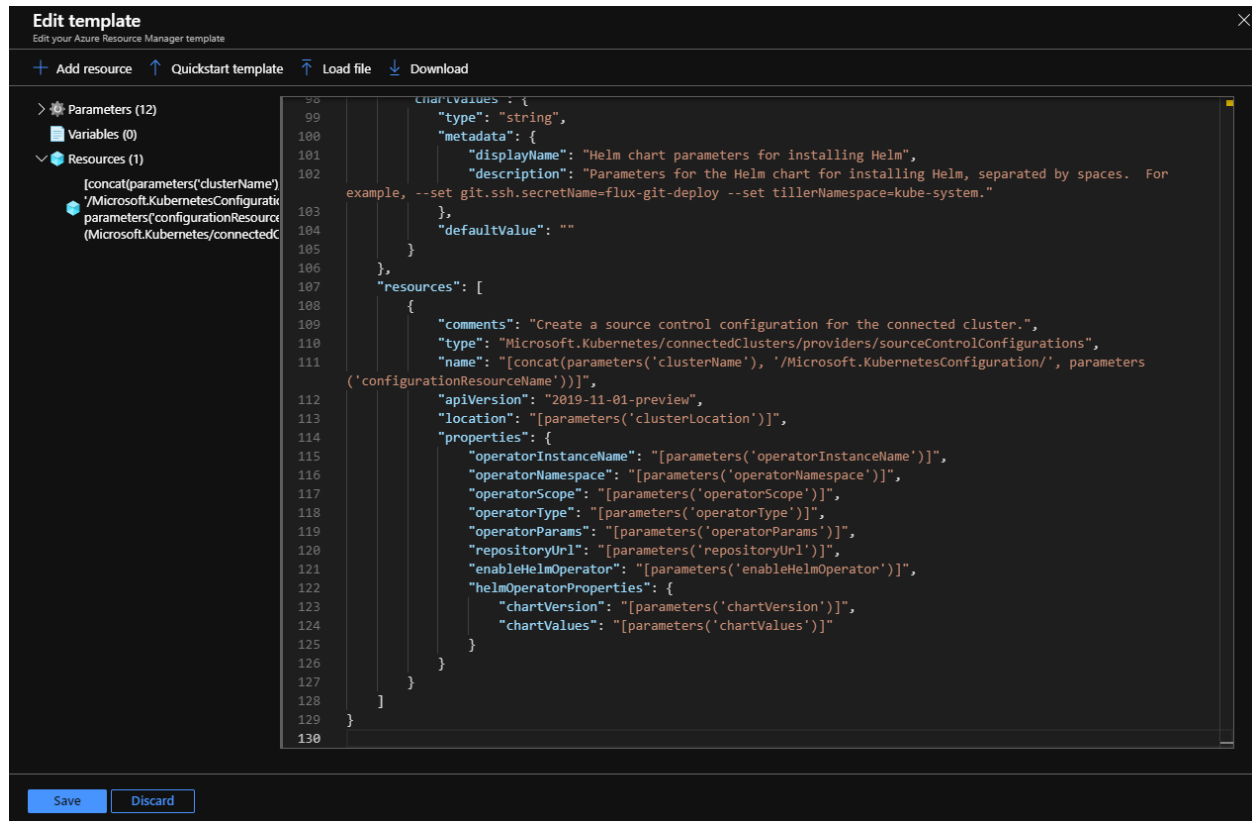
To automate the creation of sourceControlConfiguration in a connected cluster, you can deploy the [example ARM template](#), see ARM template for an Azure Arc enabled cluster.

To do so, proceed with the following steps.

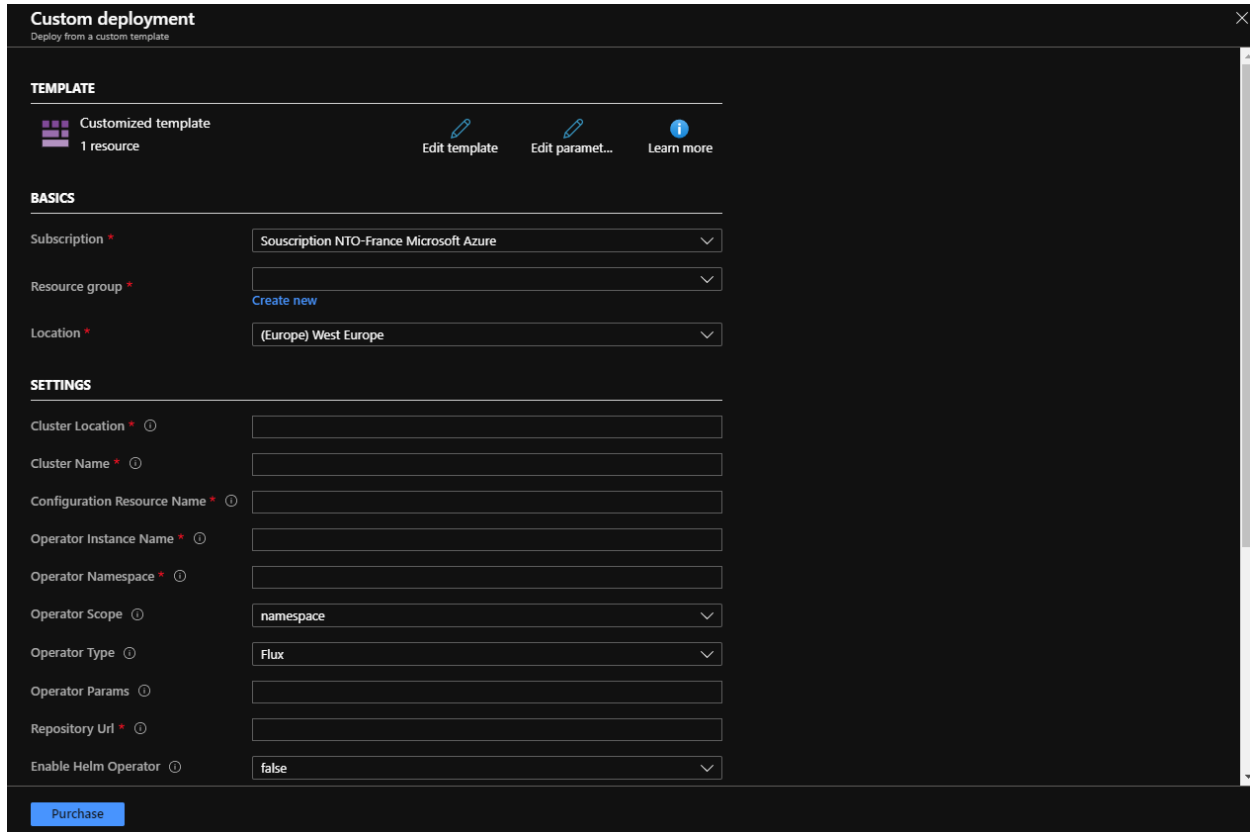
1. Open a browser session and navigate to the Azure portal at <https://portal.azure.com>.
2. Sign-in with your credentials.
3. In the portal search box (top center), type "deployment".
4. In the search results Services section, click on **Deploy a custom template**.



5. Click **Build your own template in the editor**.
6. In the edit box, delete the default content and copy/ARM template for an Azure Arc enabled clusterAppendix A. Examples of ARM template.



7. Click **Save**.
8. You'll be presented with a form for entering parameter values:



Custom deployment
Deploy from a custom template

TEMPLATE

Customized template
1 resource

Edit template Edit paramet... Learn more

BASICS

Subscription * Souscription NTO-France Microsoft Azure

Resource group * Create new

Location * (Europe) West Europe

SETTINGS

Cluster Location *

Cluster Name *

Configuration Resource Name *

Operator Instance Name *

Operator Namespace *

Operator Scope namespace

Operator Type Flux

Operator Params

Repository Url *

Enable Helm Operator false

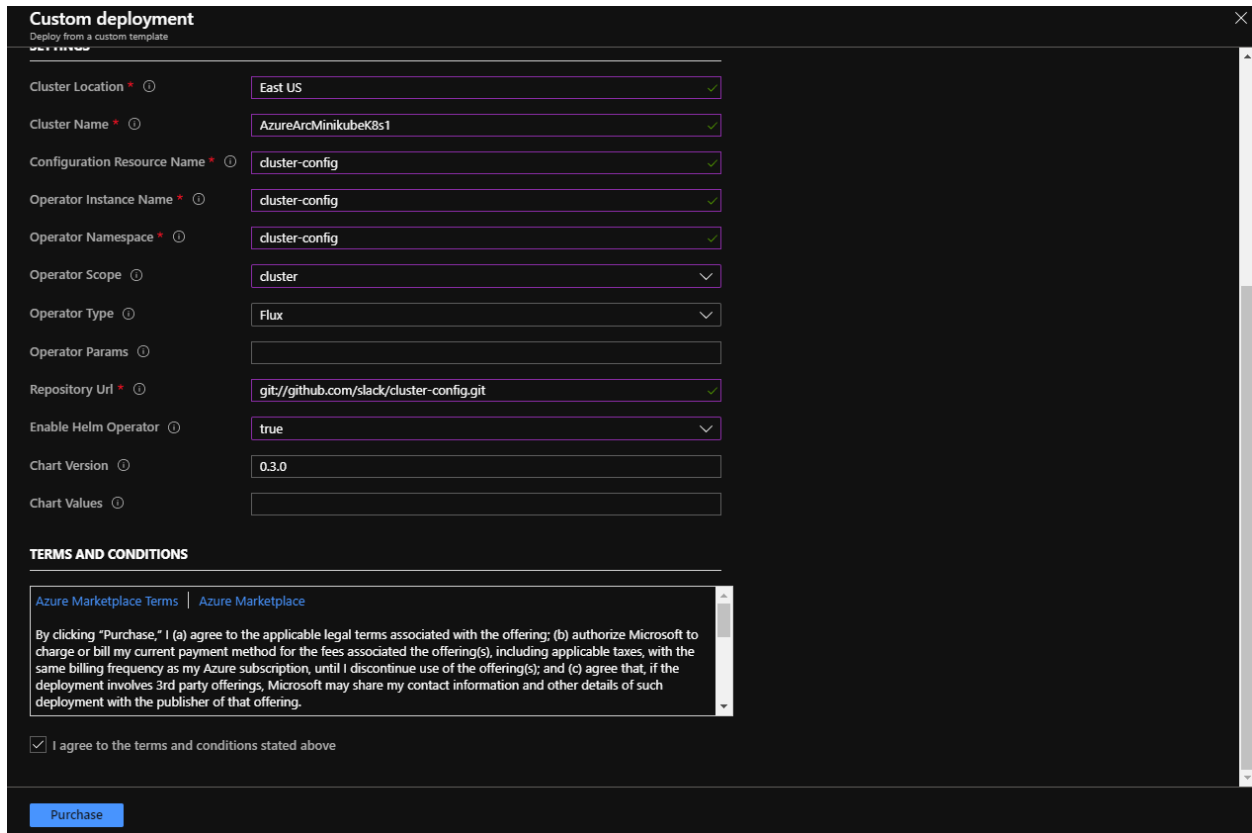
Purchase

- First select the **Resource group** where the connected cluster is located. For example, AzureArcK8sTest in our illustration.
- Enter the **Cluster Location** and the **Cluster Name** of the connectedCluster where the sourceControlConfiguration will be created. For example, "East US", and "AzureArcMinikubeK8s1" in our illustration.
- Enter the **Configuration Resource Name** for the sourceControlConfiguration - This will be the resource name in Azure -. For example, cluster-config in our illustration.
- Enter the **Operator Instance Name** - This is the operator name in the cluster -. For example, cluster-config in our illustration.
- Enter the **Operator Namespace** where the operator will be deployed in the cluster. For example, cluster-config in our illustration.
- Enter the **Operator Scope** of influence for the operator: cluster gives the operator permission to make changes throughout the cluster; namespace gives the operator permission to make changes only in the namespace. For example, **cluster** in our illustration.

As of this writing, the **Operator Type** currently is restricted to [flux](#), the [GitOps Kubernetes operator](#).

- Enter any [Operator params](#) you want to pass through to the new flux instance.
- Enter the **Repository Url** of the Git repo that the operator will monitor for Kubernetes manifests. For example, [git://github.com/slack/cluster-config.git](#) in our illustration.

- i. Indicate if you want the flux Helm operator installed or not. Select true for **Enable Helm Operator**.
 - j. If so, for the Helm operator, indicate:
 - The [Chart Version](#) to use for installation (default is 0.3.0). Stay with the default here for the sake of the illustration.
 - Any [Chart Values](#) you want to pass through to the new instance. Leave the filed blank.
9. Agree to the **TERMS AND CONDITIONS**. Check **I agree to the terms and conditions stated above** -.



Custom deployment
Deploy from a custom template

Cluster Location * ⓘ East US ✓

Cluster Name * ⓘ AzureArcMinikubeK8s1 ✓

Configuration Resource Name * ⓘ cluster-config ✓

Operator Instance Name * ⓘ cluster-config ✓

Operator Namespace * ⓘ cluster-config ✓

Operator Scope ⓘ cluster ▼

Operator Type ⓘ Flux ▼

Operator Params ⓘ

Repository Url * ⓘ git://github.com/slack/cluster-config.git ✓

Enable Helm Operator ⓘ true ▼

Chart Version ⓘ 0.3.0

Chart Values ⓘ

TERMS AND CONDITIONS

[Azure Marketplace Terms](#) | [Azure Marketplace](#)

By clicking "Purchase," I (a) agree to the applicable legal terms associated with the offering; (b) authorize Microsoft to charge or bill my current payment method for the fees associated the offering(s), including applicable taxes, with the same billing frequency as my Azure subscription, until I discontinue use of the offering(s); and (c) agree that, if the deployment involves 3rd party offerings, Microsoft may share my contact information and other details of such deployment with the publisher of that offering.

☒ I agree to the terms and conditions stated above

Purchase

10. Click **Purchase** - It's free :-). The template deployment will be started.

When it completes you can navigate to the connected cluster resource and validate the new configuration.

After validating that the ARM templates works for you, you can start using it in your automated infrastructure deployments.

Note You can also install a sourceControlConfiguration in an [Azure Kubernetes Service \(AKS\)](#) cluster. You will need to use instead the [example ARM template for deploying configuration to AKS cluster](#) (see ARM template for an AKS cluster), and then follow the steps above.

Validating the sourceControlConfiguration for your configuration

Proceed with the following steps:

1. Open a CLI
2. Connect to Azure CLI if you haven't already done so.

```
$ az login
```

When asked for, open a browser session on your Windows 10 host machine and navigate to at <https://aka.ms/devicelogin>.

- a. Enter the authorization code displayed in your Bash terminal console.
 - b. Sign in with your account credentials in the browser.
3. If you have multiple Azure subscriptions, select the subscription in which your cluster has been onboarded.

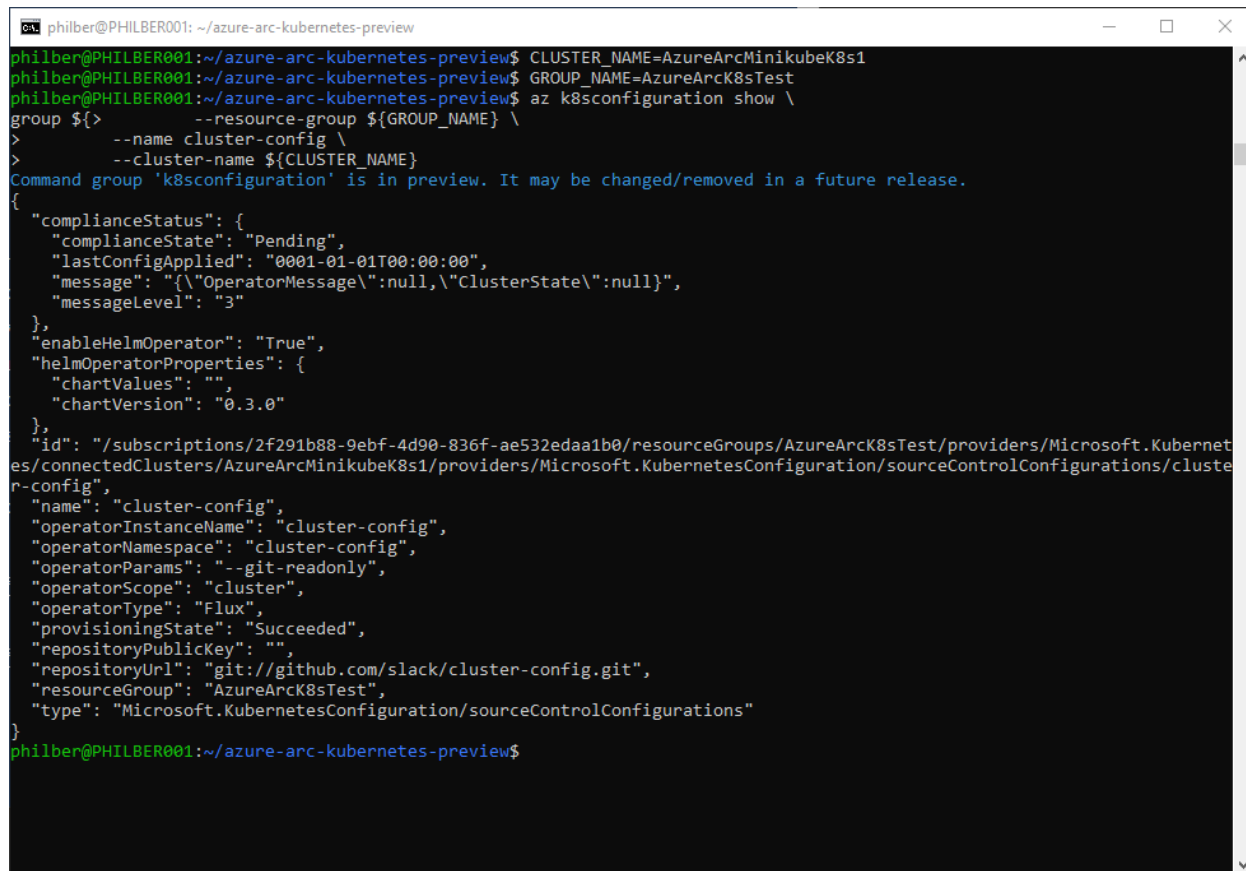
```
$ az account set -s <subscription_id>
```

Where <subscription_id> is the ID of your subscription retrieve from the Azure portal.

3. Validate that the sourceControlConfiguration was successfully created- Specify your cluster name in lieu of AzureArcMinikubeK8s1, do the same for your actual resource group -:

```
$ CLUSTER_NAME=AzureArcMinikubeK8s1
$ GROUP_NAME=AzureArcK8sTest
$ az k8sconfiguration show \
  --resource-group ${GROUP_NAME} \
  --name cluster-config \
  --cluster-name ${CLUSTER_NAME}
Command group 'k8sconfiguration' is in preview. It may be changed/removed in a future release.
{
  "complianceStatus": {
    "complianceState": "Pending",
    "lastConfigApplied": "0001-01-01T00:00:00",
    "message": "{\"OperatorMessage\":null,\"ClusterState\":null}",
    "messageLevel": "3"
  },
  "enableHelmOperator": "True",
  "helmOperatorProperties": {
    "chartValues": "",
    "chartVersion": "0.3.0"
  },
  "id": "/subscriptions/2f291b88-9ebf-4d90-836f-ae532eda1b0/resourceGroups/AzureArcK8sTest/providers/Microsoft.Kubernetes/connectedClusters/AzureArcMinikubeK8s1/providers/Microsoft.KubernetesConfiguration/sourceControlConfigurations/cluster-config",
  "name": "cluster-config",
  "operatorInstanceName": "cluster-config",
  "operatorNamespace": "cluster-config",
}
```

```
{
  "operatorParams": "--git-readonly",
  "operatorScope": "cluster",
  "operatorType": "Flux",
  "provisioningState": "Succeeded",
  "repositoryPublicKey": "",
  "repositoryUrl": "git://github.com/slack/cluster-config.git",
  "resourceGroup": "AzureArcK8sTest",
  "type": "Microsoft.KubernetesConfiguration/sourceControlConfigurations"
}
```



```
philber@PHILBER001: ~/azure-arc-kubernetes-preview
philber@PHILBER001:~/azure-arc-kubernetes-preview$ CLUSTER_NAME=AzureArcMinikubeK8s1
philber@PHILBER001:~/azure-arc-kubernetes-preview$ GROUP_NAME=AzureArcK8sTest
philber@PHILBER001:~/azure-arc-kubernetes-preview$ az k8sconfiguration show \
group ${>
  --resource-group ${GROUP_NAME} \
>
  --name cluster-config \
>
  --cluster-name ${CLUSTER_NAME}
Command group 'k8sconfiguration' is in preview. It may be changed/removed in a future release.
{
  "complianceStatus": {
    "complianceState": "Pending",
    "lastConfigApplied": "0001-01-01T00:00:00",
    "message": "{\"OperatorMessage\":null,\"ClusterState\":null}",
    "messageLevel": "3"
  },
  "enableHelmOperator": "True",
  "helmOperatorProperties": {
    "chartValues": "",
    "chartVersion": "0.3.0"
  },
  "id": "/subscriptions/2f291b88-9ebf-4d90-836f-ae532edaa1b0/resourceGroups/AzureArcK8sTest/providers/Microsoft.Kubernet
es/connectedClusters/AzureArcMinikubeK8s1/providers/Microsoft.KubernetesConfiguration/sourceControlConfigurations/cluste
r-config",
  "name": "cluster-config",
  "operatorInstanceName": "cluster-config",
  "operatorNamespace": "cluster-config",
  "operatorParams": "--git-readonly",
  "operatorScope": "cluster",
  "operatorType": "Flux",
  "provisioningState": "Succeeded",
  "repositoryPublicKey": "",
  "repositoryUrl": "git://github.com/slack/cluster-config.git",
  "resourceGroup": "AzureArcK8sTest",
  "type": "Microsoft.KubernetesConfiguration/sourceControlConfigurations"
}
philber@PHILBER001:~/azure-arc-kubernetes-preview$
```

Please note that the sourceControlConfiguration resource is updated with compliance status, messages, and debugging information.

When the sourceControlConfiguration is created, a few things happen under the hood:

1. The Azure Arc config-agent monitors Azure Resource Manager (ARM) for new or updated configurations (Microsoft.KubernetesConfiguration/sourceControlConfiguration)
2. config-agent notices the new Pending configuration
3. config-agent reads the configuration properties and prepares to deploy a managed instance of flux
 - a. config-agent creates the destination namespace
 - b. config-agent prepares a Kubernetes Service Account with the appropriate permission (cluster or namespace scope)
 - c. config-agent generates a deploy key
 - d. config-agent deploys an instance of flux

4. config-agent reports status back to the sourceControlConfiguration

While the provisioning process happens, the sourceControlConfiguration will move through a few state changes. Monitor progress with the `az k8sconfiguration show ...` command above:

1. complianceStatus -> Pending: represents the initial and in-progress states
2. complianceStatus -> Compliant: config-agent was able to successfully configure the cluster and deploy flux without error
3. complianceStatus -> Noncompliant: config-agent encountered an error deploying flux, details should be available in complianceStatus.message response body

Validating the Kubernetes configuration

After config-agent has installed the flux instance, resources held in the git repository should begin to flow to the cluster.

Proceed with the following steps:

1. Open a CLI.
2. Check to see that the namespaces, deployments, and resources have been created:

```
$ kubectl get ns --show-labels
NAME                STATUS   AGE    LABELS
azure-arc            Active   144m   <none>
cluster-config       Active   14m    <none>
dapr-system          Active   6d6h   <none>
default              Active   7d2h   <none>
itops                Active   12m    fluxcd.io/sync-gc-
mark=sha256.9oYk8yEsRwWkR09n8eJCRNafckASgghAsUWgXWEQ9es,name=itops
kube-node-lease      Active   7d2h   <none>
kube-public          Active   7d2h   <none>
kube-system          Active   7d2h   <none>
kubernetes-dashboard Active   7d1h   <none>
addonmanager.kubernetes.io/mode=Reconcile,kubernetes.io/minikube-addons=dashboard
team-a               Active   12m    fluxcd.io/sync-gc-
mark=sha256.CS5boSi8kg_vyxfAeu7Das5harSy1i0gc2fodD7YDqA,name=team-a
team-b               Active   12m    fluxcd.io/sync-gc-
mark=sha256.vF36thDIFnDDI2VEttBp5jgdxvEuaLmm7yT_cuA2UEw,name=team-b
```

You can see that team-a, team-b, itops, and cluster-config namespaces have been created.

3. The flux operator has been deployed to cluster-config namespace, as directed by your sourceControlConfig:

```
$ kubectl -n cluster-config get deploy -o wide
NAME                READY   UP-TO-DATE   AVAILABLE   AGE    flux
CONTAINERS          IMAGES
cluster-config      1/1     1             1           13m
docker.io/fluxcd/flux:1.18.0
cluster-config-helm-cluster-config-helm-operator 0/1     1             0           12m
flux-helm-operator  docker.io/fluxcd/helm-operator:1.0.0-rc4 app=helm-
operator,release=cluster-config-helm-cluster-config
```

memcached		1/1	1	1	13m
memcached	memcached:1.5.15			name=memcached	

```

philber@PHILBER001: ~/azure-arc-kubernetes-preview
philber@PHILBER001:~/azure-arc-kubernetes-preview$ kubectl get ns --show-labels
NAME                STATUS   AGE    LABELS
azure-arc            Active   144m   <none>
cluster-config       Active   14m    <none>
dapr-system          Active   6d6h   <none>
default              Active   7d2h   <none>
itops                Active   12m    fluxcd.io/sync-gc-mark=sha256.9oYk8yEsRwWkR09n8eJCRNafckASgghAsUwGXWEQ9es,name=itops
ops                  Active   7d2h   <none>
kube-node-lease      Active   7d2h   <none>
kube-public          Active   7d2h   <none>
kube-system          Active   7d2h   <none>
kubernetes-dashboard Active   7d1h   addonmanager.kubernetes.io/mode=Reconcile,kubernetes.io/minikube-addons=dashboard
team-a               Active   12m    fluxcd.io/sync-gc-mark=sha256.CS5boSi8kg_vyxfAeu7Das5harSy1i0gc2fodD7YDqA,name=team-a
am-a                 Active   12m    fluxcd.io/sync-gc-mark=sha256.vF36thDIFnDDI2VEtt8p5jgdxvEuaLmm7yT_cuA2UEw,name=team-a
am-b                 Active   12m    fluxcd.io/sync-gc-mark=sha256.vF36thDIFnDDI2VEtt8p5jgdxvEuaLmm7yT_cuA2UEw,name=team-a
philber@PHILBER001:~/azure-arc-kubernetes-preview$ kubectl -n cluster-config get deploy -o wide
NAME                READY   UP-TO-DATE   AVAILABLE   AGE    CONTAINERS      IMAGES
cluster-config      1/1     1             1           13m    flux             docker.io/flux:1.18.0
cluster-config-helm-cluster-config-helm-operator  0/1     1             0           12m    flux-helm-operator docker.io/fluxcd/helm-operator:1.0.0-rc4
memcached            1/1     1             1           13m    memcached        memcached:1.5.15
philber@PHILBER001:~/azure-arc-kubernetes-preview$

```

4. Optionnaly, now explore the other resources deployed as part of the configuration repository:

```

$ kubectl -n team-a get cm -o yaml
apiVersion: v1
items:
- apiVersion: v1
  data:
    logs: https://logs.endpoint.internal.corpnet
    metrics: https://metrics.endpoint.internal.corpnet/v2
  kind: ConfigMap
  metadata:
    annotations:
      fluxcd.io/sync-checksum: 9c10853c3e6007352ff98532586a2416183c063f
      kubectl.kubernetes.io/last-applied-configuration: |
        {"apiVersion":"v1","data":{"logs":"https://logs.endpoint.internal.corpnet","metrics":"https://metrics.endpoint.internal.corpnet/v2"},"kind":"ConfigMap","metadata":{"annotations":{"fluxcd.io/sync-checksum":"9c10853c3e6007352ff98532586a2416183c063f"},"labels":{"fluxcd.io/sync-gc-mark":"sha256.8z7gJc5Y40YRobz63Uc-xT5UHEQH_2X-RHLbw60DSSM"},"name":"endpoints","namespace":"team-a"}}
        creationTimestamp: "2020-02-20T15:31:21Z"
    labels:
      fluxcd.io/sync-gc-mark: sha256.8z7gJc5Y40YRobz63Uc-xT5UHEQH_2X-RHLbw60DSSM
    name: endpoints
    namespace: team-a
    resourceVersion: "81064"
    selfLink: /api/v1/namespaces/team-a/configmaps/endpoints
    uid: 09660c97-64de-425f-90c7-005cd09d00e8
  kind: List

```

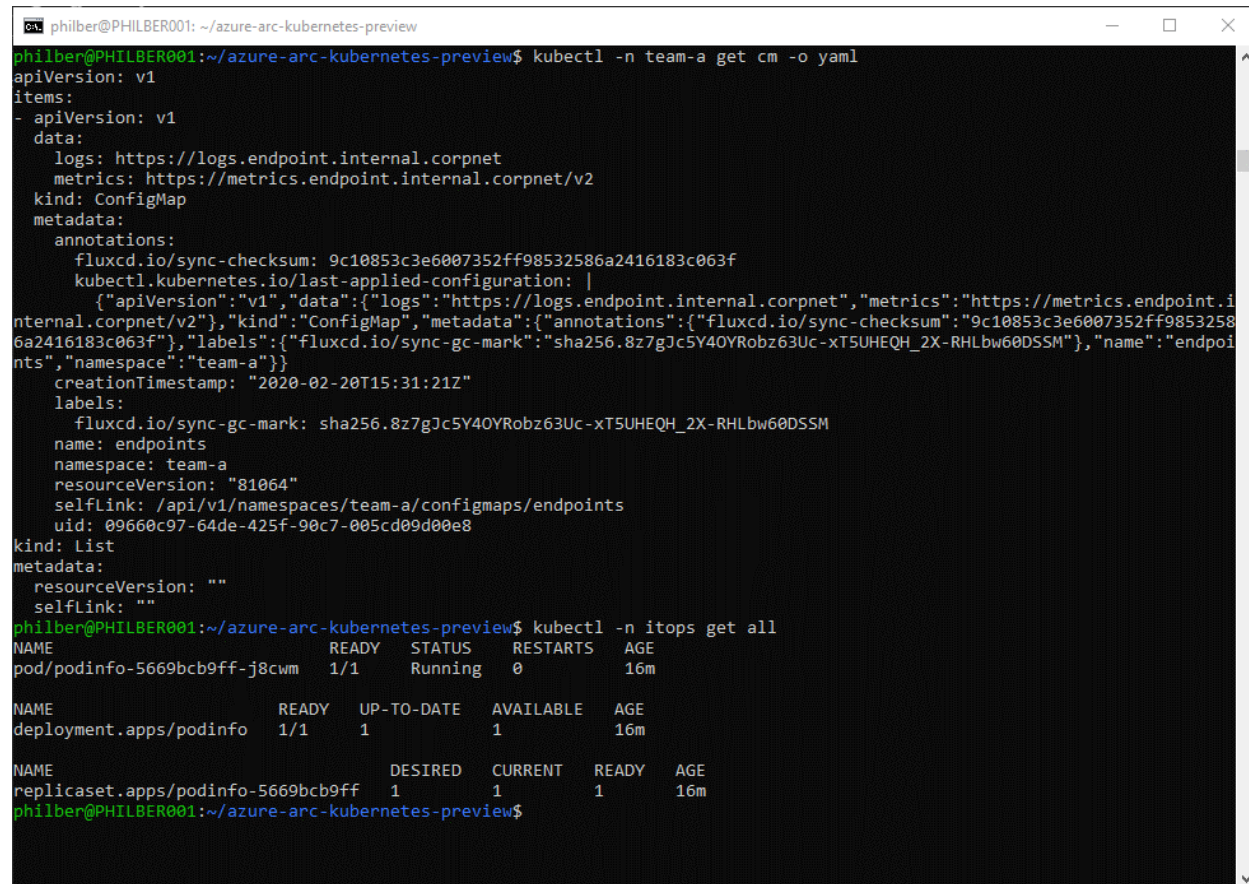
```

metadata:
  resourceVersion: ""
  selfLink: ""
$ kubectl -n itops get all
NAME                                READY   STATUS    RESTARTS   AGE
pod/podinfo-5669bcb9ff-j8cwm       1/1     Running   0           16m

NAME                                READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/podinfo            1/1     1             1           16m

NAME                                DESIRED   CURRENT   READY   AGE
replicaset.apps/podinfo-5669bcb9ff 1         1         1       16m

```



```

philber@PHILBER001: ~/azure-arc-kubernetes-preview
philber@PHILBER001:~/azure-arc-kubernetes-preview$ kubectl -n team-a get cm -o yaml
apiVersion: v1
items:
- apiVersion: v1
  data:
    logs: https://logs.endpoint.internal.corpnet
    metrics: https://metrics.endpoint.internal.corpnet/v2
  kind: ConfigMap
  metadata:
    annotations:
      fluxcd.io/sync-checksum: 9c10853c3e6007352ff98532586a2416183c063f
      kubectl.kubernetes.io/last-applied-configuration: |
        {"apiVersion":"v1","data":{"logs":"https://logs.endpoint.internal.corpnet","metrics":"https://metrics.endpoint.i
        nternal.corpnet/v2"},"kind":"ConfigMap","metadata":{"annotations":{"fluxcd.io/sync-checksum":"9c10853c3e6007352ff9853258
        6a2416183c063f"},"labels":{"fluxcd.io/sync-gc-mark":"sha256.8z7gJc5Y40YRobz63Uc-xT5UHEQH_2X-RHLbw60DSSM"},"name":"endpoi
        nts","namespace":"team-a"}}}
      creationTimestamp: "2020-02-20T15:31:21Z"
    labels:
      fluxcd.io/sync-gc-mark: sha256.8z7gJc5Y40YRobz63Uc-xT5UHEQH_2X-RHLbw60DSSM
    name: endpoints
    namespace: team-a
    resourceVersion: "81064"
    selfLink: /api/v1/namespaces/team-a/configmaps/endpoints
    uid: 09660c97-64de-425f-90c7-005cd09d00e8
  kind: List
  metadata:
    resourceVersion: ""
    selfLink: ""
philber@PHILBER001:~/azure-arc-kubernetes-preview$ kubectl -n itops get all
NAME                                READY   STATUS    RESTARTS   AGE
pod/podinfo-5669bcb9ff-j8cwm       1/1     Running   0           16m

NAME                                READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/podinfo            1/1     1             1           16m

NAME                                DESIRED   CURRENT   READY   AGE
replicaset.apps/podinfo-5669bcb9ff 1         1         1       16m
philber@PHILBER001:~/azure-arc-kubernetes-preview$

```

Doing further exploration

With the above understanding, you are now fully equipped to explore other configuration.

What about deploying [Rudr](#), i.e. a Kubernetes implementation of the [Open Application Model \(OAM\)](#) specification.

Note For more information, see [Understanding and leveraging the Open Application Model \(OAM\) and Rudr – A starter guide for developers and others](#). This whitepaper is part of the series of guides [New perspectives](#)

[for cloud-native applications with the Open Application Model \(OAM\), and the Distributed Application Runtime \(Dapr\).](#)

For the sake of this illustration, you will use another example configuration repository: <https://github.com/slack/rudr-gitops>.

You will give this configuration a name, for example, `cluster-config` in our illustration, instruct the agent to deploy the operator in the `cluster-config` namespace, and give the operator `cluster-admin` permissions.

To create the Rudr configuration, proceed with the following steps:

1. Open a CLI.
2. Connect to Azure CLI if you haven't already done so:

```
$ az login
```

When asked for, open a browser session on your Windows 10 host machine and navigate to <https://aka.ms/devicelogin>.

- a. Enter the authorization code displayed in your Bash terminal console.
- b. Sign in with your account credentials in the browser.
4. If you have multiple Azure subscriptions, select the subscription in which your cluster has been onboarded.

```
$ az account set -s <subscription_id>
```

Where `<subscription_id>` is the ID of your subscription retrieve from the Azure portal.

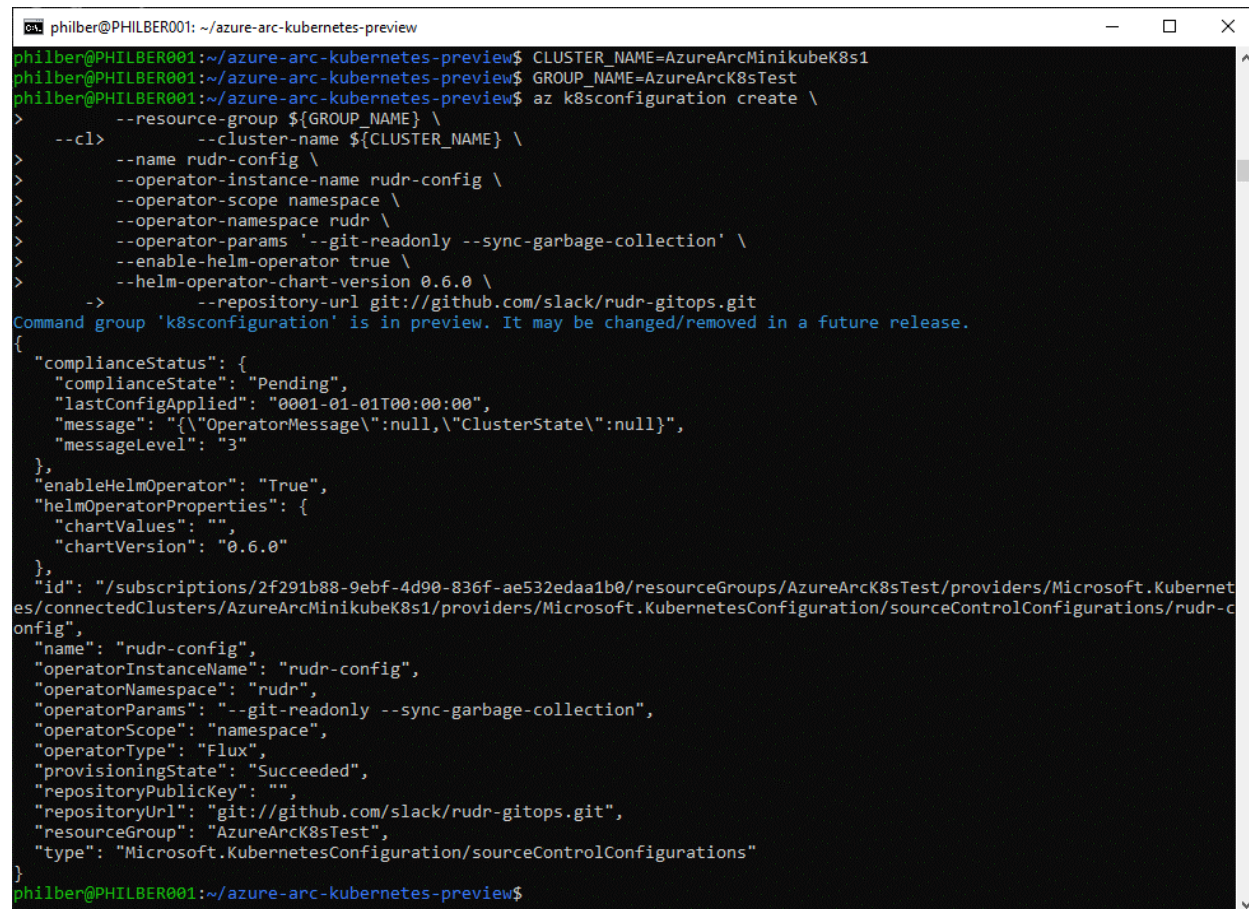
3. Use as you did above the Azure CLI extension for `k8sconfiguration` - Specify your cluster name in lieu of `AzureArcMinikubeK8s1`, do the same for your actual resource group -:

```
$ CLUSTER_NAME=AzureArcMinikubeK8s1
$ GROUP_NAME=AzureArcK8sTest
$ az k8sconfiguration create \
  --resource-group ${GROUP_NAME} \
  --cluster-name ${CLUSTER_NAME} \
  --name rudr-config \
  --operator-instance-name rudr-config \
  --operator-scope namespace \
  --operator-namespace rudr \
  --operator-params '--git-readonly --sync-garbage-collection' \
  --enable-helm-operator true \
  --helm-operator-chart-version 0.6.0 \
  --repository-url git://github.com/slack/rudr-gitops.git
```

Command group 'k8sconfiguration' is in preview. It may be changed/removed in a future release.

```
{
  "complianceStatus": {
    "complianceState": "Pending",
    "lastConfigApplied": "0001-01-01T00:00:00",
    "message": "{\n\"OperatorMessage\":null,\n\"ClusterState\":null}",
    "messageLevel": "3"
  },
  "enableHelmOperator": "True",
```

```
{
  "helmOperatorProperties": {
    "chartValues": "",
    "chartVersion": "0.6.0"
  },
  "id": "/subscriptions/2f291b88-9ebf-4d90-836f-ae532edaa1b0/resourceGroups/AzureArcK8sTest/providers/Microsoft.Kubernetes/connectedClusters/AzureArcMinikubeK8s1/providers/Microsoft.KubernetesConfiguration/sourceControlConfigurations/rudr-config",
  "name": "rudr-config",
  "operatorInstanceName": "rudr-config",
  "operatorNamespace": "rudr",
  "operatorParams": "--git-readonly --sync-garbage-collection",
  "operatorScope": "namespace",
  "operatorType": "Flux",
  "provisioningState": "Succeeded",
  "repositoryPublicKey": "",
  "repositoryUrl": "git://github.com/slack/rudr-gitops.git",
  "resourceGroup": "AzureArcK8sTest",
  "type": "Microsoft.KubernetesConfiguration/sourceControlConfigurations"
}
```



```
philber@PHILBER001: ~/azure-arc-kubernetes-preview
philber@PHILBER001:~/azure-arc-kubernetes-preview$ CLUSTER_NAME=AzureArcMinikubeK8s1
philber@PHILBER001:~/azure-arc-kubernetes-preview$ GROUP_NAME=AzureArcK8sTest
philber@PHILBER001:~/azure-arc-kubernetes-preview$ az k8sconfiguration create \
> --resource-group ${GROUP_NAME} \
> --cld> --cluster-name ${CLUSTER_NAME} \
> --name rudr-config \
> --operator-instance-name rudr-config \
> --operator-scope namespace \
> --operator-namespace rudr \
> --operator-params '--git-readonly --sync-garbage-collection' \
> --enable-helm-operator true \
> --helm-operator-chart-version 0.6.0 \
> --repository-url git://github.com/slack/rudr-gitops.git
Command group 'k8sconfiguration' is in preview. It may be changed/removed in a future release.
{
  "complianceStatus": {
    "complianceState": "Pending",
    "lastConfigApplied": "0001-01-01T00:00:00",
    "message": "{\"OperatorMessage\":null,\"ClusterState\":null}",
    "messageLevel": "3"
  },
  "enableHelmOperator": "True",
  "helmOperatorProperties": {
    "chartValues": "",
    "chartVersion": "0.6.0"
  },
  "id": "/subscriptions/2f291b88-9ebf-4d90-836f-ae532edaa1b0/resourceGroups/AzureArcK8sTest/providers/Microsoft.Kubernetes/connectedClusters/AzureArcMinikubeK8s1/providers/Microsoft.KubernetesConfiguration/sourceControlConfigurations/rudr-config",
  "name": "rudr-config",
  "operatorInstanceName": "rudr-config",
  "operatorNamespace": "rudr",
  "operatorParams": "--git-readonly --sync-garbage-collection",
  "operatorScope": "namespace",
  "operatorType": "Flux",
  "provisioningState": "Succeeded",
  "repositoryPublicKey": "",
  "repositoryUrl": "git://github.com/slack/rudr-gitops.git",
  "resourceGroup": "AzureArcK8sTest",
  "type": "Microsoft.KubernetesConfiguration/sourceControlConfigurations"
}
```

Once done, to remove the above configuration, run the following commands:

```
$ CLUSTER_NAME=AzureArcMinikubeK8s1
$ GROUP_NAME=AzureArcK8sTest
$ az k8sconfiguration delete
```

```
--resource-group ${GROUP_NAME} \  
--cluster-name ${CLUSTER_NAME} \  
--name rudr-config
```

This concludes this walkthrough.

Appendix A. Examples of ARM template

ARM template for an Azure Arc enabled cluster

```
{
  "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "parameters": {
    "clusterLocation": {
      "type": "string",
      "metadata": {
        "displayName": "Connected Cluster Location",
        "description": "The Azure location for the connected cluster. For example, East US."
      }
    },
    "clusterName": {
      "type": "string",
      "metadata": {
        "displayName": "Connected Cluster Name",
        "description": "The ARM name for the Microsoft.Kubernetes/connectedClusters resource that the configuration will be created on."
      }
    },
    "configurationResourceName": {
      "type": "string",
      "metadata": {
        "displayName": "Configuration Resource Name",
        "description": "The ARM name for the Microsoft.Kubernetes/connectedClusters/providers/sourceControlConfigurations resource that will be created."
      }
    },
    "operatorInstanceName": {
      "type": "string",
      "metadata": {
        "displayName": "Operator Instance Name",
        "description": "The instance name of the operator, used to identify the specific configuration. The instance name can contain up to 353 lower-case alphanumeric characters, hyphen, or period."
      }
    },
    "operatorNamespace": {
      "type": "string",
      "metadata": {
        "displayName": "Operator Namespace",
        "description": "The namespace in which the Config operator will be installed. The namespace can contain up to 353 lower-case alphanumeric characters, hyphen, or period."
      }
    },
    "operatorScope": {
      "type": "string",
      "metadata": {
        "displayName": "Operator Scope",
```

```

        "description": "Enter the scope of influence for the operator: 'cluster' gives
the operator permission to make changes throughout the cluster; 'namespace' gives the operator
permission to make changes only in the namespace."
    },
    "allowedValues": [
        "cluster",
        "namespace"
    ],
    "defaultValue": "namespace"
},
"operatorType": {
    "type": "string",
    "metadata": {
        "displayName": "Operator Type",
        "description": "The type of operator to install. Currently, 'Flux' is
supported."
    },
    "allowedValues": [
        "Flux"
    ],
    "defaultValue": "Flux"
},
"operatorParams": {
    "type": "string",
    "metadata": {
        "displayName": "Operator Parameters",
        "description": "Parameters to pass to the Flux operator, separated by spaces.
For example, --git-email someuser@users.noreply.github.com --git-path namespaces,workloads.
If the --git-email parameter is not included, then the --git-readonly flag will be set
automatically."
    },
    "defaultValue": ""
},
"repositoryUrl": {
    "type": "string",
    "metadata": {
        "displayName": "Repository Url",
        "description": "The Url of the source control repository."
    }
},
"enableHelmOperator": {
    "type": "string",
    "metadata": {
        "displayName": "Enable Helm",
        "description": "Indicate whether to enable Helm for this instance of Flux."
    },
    "allowedValues": [
        "true",
        "false"
    ],
    "defaultValue": "false"
},
"chartVersion": {
    "type": "string",
    "metadata": {
        "displayName": "Helm chart version for installing Helm",
        "description": "The version of the Helm chart for installing Helm. For
example, 0.3.0"
    },
    "defaultValue": "0.3.0"
},

```

```

    "chartValues": {
      "type": "string",
      "metadata": {
        "displayName": "Helm chart parameters for installing Helm",
        "description": "Parameters for the Helm chart for installing Helm, separated
by spaces. For example, --set git.ssh.secretName=flux-git-deploy --set tillerNamespace=kube-
system."
      },
      "defaultValue": ""
    },
  },
  "resources": [
    {
      "comments": "Create a source control configuration for the connected cluster.",
      "type":
"Microsoft.Kubernetes/connectedClusters/providers/sourceControlConfigurations",
      "name": "[concat(parameters('clusterName'), '/Microsoft.KubernetesConfiguration/',
parameters('configurationResourceName'))]",
      "apiVersion": "2019-11-01-preview",
      "location": "[parameters('clusterLocation')]",
      "properties": {
        "operatorInstanceName": "[parameters('operatorInstanceName')]",
        "operatorNamespace": "[parameters('operatorNamespace')]",
        "operatorScope": "[parameters('operatorScope')]",
        "operatorType": "[parameters('operatorType')]",
        "operatorParams": "[parameters('operatorParams')]",
        "repositoryUrl": "[parameters('repositoryUrl')]",
        "enableHelmOperator": "[parameters('enableHelmOperator')]",
        "helmOperatorProperties": {
          "chartVersion": "[parameters('chartVersion')]",
          "chartValues": "[parameters('chartValues')]"
        }
      }
    }
  ]
}

```

ARM template for an AKS cluster

```

{
  "$schema": "https://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "parameters": {
    "clusterLocation": {
      "type": "string",
      "metadata": {
        "displayName": "AKS Cluster Location",
        "description": "The Azure location for the AKS cluster. For example, East
US."
      }
    },
    "clusterName": {
      "type": "string",
      "metadata": {
        "displayName": "AKS Cluster Name",
        "description": "The ARM name for the
Microsoft.ContainerService/managedClusters resource that the configuration will be created
on."
      }
    }
  }
}

```

```

    },
    "configurationResourceName": {
      "type": "string",
      "metadata": {
        "displayName": "Configuration Resource Name",
        "description": "The ARM name for the
Microsoft.ContainerService/managedClusters/providers/sourceControlConfigurations resource that
will be created."
      }
    },
    "operatorInstanceName": {
      "type": "string",
      "metadata": {
        "displayName": "Operator Instance Name",
        "description": "The instance name of the operator, used to identify the
specific configuration. The instance name can contain up to 353 lower-case alphanumeric
characters, hyphen, or period."
      }
    },
    "operatorNamespace": {
      "type": "string",
      "metadata": {
        "displayName": "Operator Namespace",
        "description": "The namespace in which the Config operator will be installed.
The namespace can contain up to 353 lower-case alphanumeric characters, hyphen, or period."
      }
    },
    "operatorScope": {
      "type": "string",
      "metadata": {
        "displayName": "Operator Scope",
        "description": "Enter the scope of influence for the operator: 'cluster' gives
the operator permission to make changes throughout the cluster; 'namespace' gives the operator
permission to make changes only in the namespace."
      },
      "allowedValues": [
        "cluster",
        "namespace"
      ],
      "defaultValue": "namespace"
    },
    "operatorType": {
      "type": "string",
      "metadata": {
        "displayName": "Operator Type",
        "description": "The type of operator to install. Currently, 'Flux' is
supported."
      },
      "allowedValues": [
        "Flux"
      ],
      "defaultValue": "Flux"
    },
    "operatorParams": {
      "type": "string",
      "metadata": {
        "displayName": "Operator Parameters",
        "description": "Parameters to pass to the Flux operator, separated by spaces.
For example, --git-email someuser@users.noreply.github.com --git-path namespaces,workloads.

```

If the `--git-email` parameter is not included, then the `--git-readonly` flag will be set automatically."

```

    },
    "defaultValue": ""
  },
  "repositoryUrl": {
    "type": "string",
    "metadata": {
      "displayName": "Repository Url",
      "description": "The Url of the source control repository."
    }
  },
  "enableHelmOperator": {
    "type": "string",
    "metadata": {
      "displayName": "Enable Helm",
      "description": "Indicate whether to enable Helm for this instance of Flux."
    },
    "allowedValues": [
      "true",
      "false"
    ],
    "defaultValue": "false"
  },
  "chartVersion": {
    "type": "string",
    "metadata": {
      "displayName": "Helm chart version for installing Helm",
      "description": "The version of the Helm chart for installing Helm. For
example, 0.3.0"
    },
    "defaultValue": "0.3.0"
  },
  "chartValues": {
    "type": "string",
    "metadata": {
      "displayName": "Helm chart parameters for installing Helm",
      "description": "Parameters for the Helm chart for installing Helm, separated
by spaces. For example, --set git.ssh.secretName=flux-git-deploy --set tillerNamespace=kube-
system."
    },
    "defaultValue": ""
  },
  "resources": [
    {
      "comments": "Create a source control configuration for the AKS cluster.",
      "type":
"Microsoft.ContainerService/managedClusters/providers/sourceControlConfigurations",
      "name": "[concat(parameters('clusterName'), '/Microsoft.KubernetesConfiguration/',
parameters('configurationResourceName'))]",
      "apiVersion": "2019-11-01-preview",
      "location": "[parameters('clusterLocation')]",
      "properties": {
        "operatorInstanceName": "[parameters('operatorInstanceName')]",
        "operatorNamespace": "[parameters('operatorNamespace')]",
        "operatorScope": "[parameters('operatorScope')]",
        "operatorType": "[parameters('operatorType')]",
        "operatorParams": "[parameters('operatorParams')]",
        "repositoryUrl": "[parameters('repositoryUrl')]",
        "enableHelmOperator": "[parameters('enableHelmOperator')]",

```

```
    "helmOperatorProperties": {  
      "chartVersion": "[parameters('chartVersion')]",  
      "chartValues": "[parameters('chartValues')]"  
    }  
  }  
]  
}
```