

## **Step by Step PlatformOne BigBang DSOP Deployment**

Deploying Big Bang is a two stage process.

- Stage 1: Deployment of an RKE2 cluster (**DSOP-RKE2**)
- Stage 2: Deployment of Big Bang on the RKE2 cluster created in stage 1 (**DSOP-ENVIRONMENT**)

Mandatory tools and accounts access required for P1 Bigbang DSOP deployment.

### **1. Accounts Access:**

<b>Azure DevOps</b>	<a href="https://azure.microsoft.com/en-us/services/devops/">https://azure.microsoft.com/en-us/services/devops/</a>
<b>Iron Bank Account</b>	<b>Existing Login:</b> <a href="https://ironbank.dso.mil/repomap/products?page=1&amp;sort=1">https://ironbank.dso.mil/repomap/products?page=1&amp;sort=1</a> <b>New Registration:</b> <a href="https://login.dso.mil/auth/realms/baby-yoda/protocol/openid-connect/registrations?client_id=account&amp;response_type=code">https://login.dso.mil/auth/realms/baby-yoda/protocol/openid-connect/registrations?client_id=account&amp;response_type=code</a>
<b>GitHub</b>	<a href="https://github.com/">https://github.com/</a>
<b>Azure Portal</b>	Azure subscription with full access. Should be able to create resources as an Administrator. <a href="https://portal.azure.us/#home">https://portal.azure.us/#home</a>

## 2. Required Tools:

The following commands to install on **Windows PC / WLS 2 / Ubuntu 18.04 LTS**

Tools scripts can be found: <https://github.com/benc-uk/tools-install>

<b>Bash</b>	(Linux / WSL2 / MacOS - Terminal)
<b>Terraform</b>	<pre>sudo apt-get update wget https://releases.hashicorp.com/terraform/1.0.1/terraform_1.0.1_linux_amd64.zip sudo apt-get install zip -y unzip terraform*.zip sudo mv terraform /usr/local/bin terraform -version</pre>
<b>Chocolatey (Optional)</b>	<p><a href="https://chocolatey.org/install">https://chocolatey.org/install</a></p> <p><a href="https://www.educba.com/linux-jq/">https://www.educba.com/linux-jq/</a></p> <p>Install JQ: (JQ is a lightweight and flexible command-line JSON processor)</p> <ul style="list-style-type: none"><li>◦ <b>chocolatey install jq</b></li></ul>
<b>JQ</b>	<p><b>Below commands, install JQ:</b></p> <pre>sudo apt-get update sudo apt install jq jq --version</pre>
<b>Kubectl</b> >= 1.21.0	<pre>curl -LO "https://dl.k8s.io/release/\$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"  curl -LO "https://dl.k8s.io/\$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl.sha256"  echo "\$(cat kubectl.sha256) kubectl"   sha256sum --check  sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl</pre>

	<u><code>sudo chmod +x kubectl</code></u> <u><code>sudo mkdir -p ~/.local/bin/kubectl</code></u> <u><code>sudo mv ./kubectl ~/.local/bin/kubectl</code></u> <u># and then add ~/.local/bin/kubectl to \$PATH</u> <u><code>kubectl version --client</code></u>
<b>Azure CLI</b>	<code>curl -sL <a href="https://aka.ms/InstallAzureCLIDeb">https://aka.ms/InstallAzureCLIDeb</a>   sudo bash</code> <code>az version</code>
<b>gpg</b>	<code>sudo apt-get install -y gpg</code> <code>gpg --version</code>
<b>Sops</b>	Download the latest sops version: <b>sops 3.7.1</b> <code>wget <a href="https://github.com/mozilla/sops/releases/download/v3.7.1/sops_3.7.1_amd64.deb">https://github.com/mozilla/sops/releases/download/v3.7.1/sops_3.7.1_amd64.deb</a></code>  <code>sudo dpkg -i sops_3.7.1_amd64.deb</code>  <code>sops --version</code> sops 3.7.1 (latest)     (OUTPUT)
<b>Kustomize</b>	<code>sudo curl -s "https://raw.githubusercontent.com/kubernetes-sigs/kustomize/master/hack/install_kustomize.sh"   bash</code>  <code>sudo mv kustomize /usr/local/bin</code> <code>kustomize version</code>

**Stage 1: Deployment of an RKE2 cluster (DSOP-RKE2)**

**Azure DevOps Repository:** [https://azure-ecosystem.visualstudio.com/Azure%20Gov%20Engineering/\\_git/dsop-rke2](https://azure-ecosystem.visualstudio.com/Azure%20Gov%20Engineering/_git/dsop-rke2)  
OR

**GitHub:** <https://github.com/cheruvu1/dsop-rke2>

**Following tools required for the dsop-rke2:**

**Note:** Following commands are compatible with **Ubuntu Linux System [Windows PC / WLS 2 / Ubuntu 18.04 LTS]**

**Follow the below steps to install dsop-rke2:**

Step1	<p><b>Azure DevOps Repository:</b></p> <p>git clone <a href="https://azure-ecosystem.visualstudio.com/Azure%20Gov%20Engineering/_git/dsop-rke2">https://azure-ecosystem.visualstudio.com/Azure%20Gov%20Engineering/_git/dsop-rke2</a> OR</p> <p><b>GitHub Repository:</b></p> <p>git clone <a href="https://github.com/cheruvu1/dsop-rke2">https://github.com/cheruvu1/dsop-rke2</a> code . (Open the dsop-rke2 in Visual Studio Code or preferred IDE)</p> <p><b>For VNET Customization, use the following Repository:</b></p> <p>git clone <a href="https://github.com/cheruvu1/dsop-rke2-vnet-customization.git">https://github.com/cheruvu1/dsop-rke2-vnet-customization.git</a> code . (Open the dsop-rke2 in Visual Studio Code or preferred IDE)</p>
Step2	<p>cd example (GoTo Example folder) copy `terraform.tfvars.sample` to `terraform.tfvars`</p> <p>Line Number 2: Change `cluster_name` and other settings, but most can be left as the defaults</p>



Step3	<p><b>Login to Azure Portal using Command line:</b></p> <pre>az cloud set --name AzureUSGovernment (Switch to Azure US Government, if pointing to Azure Commercial) az cloud list --output table az login</pre> <p><b>Run Terraform commands:</b></p> <pre>sudo terraform init  sudo terraform apply -auto-approve</pre>
Step4	<p><b>Folder:</b> /dsop-rke2/example</p> <pre>terraform output -raw kv_name</pre> <pre>KV_NAME=\${1:-\$(terraform output -raw kv_name)} echo \$KV_NAME</pre> <pre>source ../scripts/fetch-kubeconfig.sh FILE=\$(realpath rke2.kubeconfig)</pre> <pre>echo \$FILE az keyvault secret show --name kubeconfig --vault-name \$KV_NAME   jq -r '.value' &gt; \$FILE</pre> <pre>export KUBECONFIG=\$PWD/rke2.kubeconfig echo \$KUBECONFIG</pre>
Step5	<p>Download the Private Key Copy the key vault name: <b>rke2-lmco-example-a06</b> (replace below line)</p> <pre>az keyvault secret show --name node-key --vault-name <b>rke2-lmco-example-a06</b>   jq -r '.value' &gt; rke2.priv_key</pre>

	cat rke2.priv_key
Step6	<del>Execute the following command from terraform state folder..</del>  source ../scripts/fetch-kubeconfig.sh (script creates the file --> rke2.kubeconfig)
Step7 – Is needed to ssh to the server.	Sudo chmod 400 rke2.priv_key  Go To Azure Portal Console Open the resource group: rke2-lmco-example Open type Public IP resource: rke2-lmco-example-wyf-pip (name may change) ssh rke2@52.227.192.136 -p 5001 -i rke2.priv_key (Optional)  kubectl get nodes kubectl get nodes -A
Final step	This concludes the RKE2 cluster deploy.

**Stage 2:** Deployment of Big Bang on the RKE2 cluster created in stage 1 (**DSOP-ENVIRONMENT**)

**1) DSOP-ENVIRONMENT Repo:**

**Azure DevOps Repository:** [https://azure-ecosystem.visualstudio.com/Azure%20Gov%20Engineering/\\_git/dsop-environment](https://azure-ecosystem.visualstudio.com/Azure%20Gov%20Engineering/_git/dsop-environment)  
OR

**GitHub:** <https://github.com/cheruvu1/dsop-environment>

## 2) Setup Instructions:

**Option 1: Readme file contains step by step instructions for the PlatformOne DSOP installation...**

[https://azure-ecosystem.visualstudio.com/Azure%20Gov%20Engineering/\\_git/dsop-environment?path=/readme.md](https://azure-ecosystem.visualstudio.com/Azure%20Gov%20Engineering/_git/dsop-environment?path=/readme.md)

**Option 2: Follow below steps: PlatformOne BigBang Environment Setup:**

This is a set of manual pre-req steps that has to be done, and can't realistically be scripted

### Set Up Git Repo:

<b>Git Clone</b>	<b>Azure DevOps Repository:</b> Clone this repo to your machine, you can use your personal Azure AD account to do this. sudo git clone <a href="https://azure-ecosystem.visualstudio.com/Azure Gov Engineering/_git/dsop-environment">https://azure-ecosystem.visualstudio.com/Azure Gov Engineering/_git/dsop-environment</a> OR <b>GitHub Repository:</b> sudo git clone <a href="https://github.com/cheruvu1/dsop-environment">https://github.com/cheruvu1/dsop-environment</a>
	Create a new branch and name it, a suggestion is to place env/ as a prefix in front of the branch name, e.g. env/dbowie , to identify each developer's own environment branch
<b>GitOps need your own branch.</b> Push branch	Push branch back to remote origin so it is tracked, e.g. git push --set-upstream origin {branch-name}  cd dsop-environment git checkout -b env/bha origin/main



	<pre>git branch -v git push --set-upstream origin env/bha</pre>
Generate Git Credentials	<p><b>Azure DevOps:</b></p> <p>Create a set of credentials to clone the repo, these will be used by Flux, you cannot use your Azure AD account or credentials. From <a href="#">the Azure DevOps page for this repo</a></p> <ul style="list-style-type: none"> <li>○ Click 'Clone' button again</li> <li>○ Click 'Generate Git Credentials' button</li> <li>○ Make a note of the username and password generated, they are needed for secrets.sh</li> </ul> <p><b>GitHub:</b></p> <p>From your GitHub account,</p> <p>go to Settings =&gt; Developer Settings =&gt; Personal Access Token =&gt; Generate New Token (Give your password) =&gt; Fill-up the form =&gt; click Generate token =&gt; Copy the generated Token</p>

### Generate wildcard certificate for your domain

Self signed Certificate	<p>A certificate for non-production environments can be generated by executing the following steps:</p> <p><b>Folder Location: dsop-environment</b></p> <p>HOSTNAME=bigbang.dev</p>
-------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

	<pre>./scripts/create-root-cert.sh  ./scripts/create-domain-cert.sh \$HOSTNAME  ISTIO_GW_CRT=\$(cat \$HOSTNAME.crt   base64 -w0)  echo \$ISTIO_GW_CRT  ISTIO_GW_KEY=\$(cat \$HOSTNAME.key   base64 -w0)  echo \$ISTIO_GW_KEY</pre>
<b>Key Vault stored certificate</b>	<pre>export ISTIO_GW_CRT="&lt;certificate id in keyvault&gt;" (copy the output from echo \$ISTIO_GW_CRT) export ISTIO_GW_KEY="&lt;certificate id in keyvault&gt;" (copy the output from echo \$ISTIO_GW_KEY)</pre> <p><b>If the scripts already executed once,</b></p> <p>If your certificate is stored already as secrets in keyvault set ISTIO_GW_CRT and ISTIO_GW_KEY to the keyvault id of those secrets in secrets.sh</p> <p>If your certificate is stored already as secrets in keyvault set USE_KEYVAULT_CERT to true on deploy-vars.sh</p> <p><i>Changing certificate</i></p> <p>If your certificate was changed change the value in secrets.sh and deploy-vars.sh then execute update-certs.sh</p>

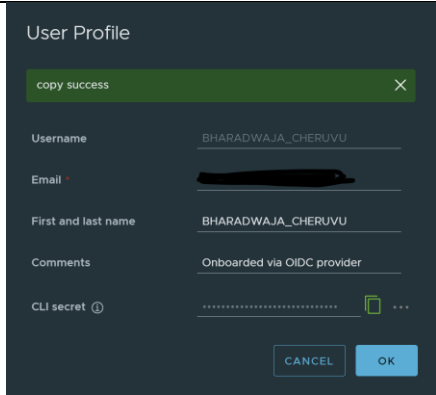
### Configure For GitOps

dev/bigbang.yaml file	<p>Update the dev/bigbang.yaml file,</p> <ol style="list-style-type: none"> <li>1. <b>LINE#14:</b> url: <a href="https://github.com/cheruvu1/dsop-environment">https://github.com/cheruvu1/dsop-environment</a></li> <li>2. <b>LINE#16:</b> place your own branch name where it has __CHANGE_ME__</li> </ol>
-----------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

	Example: branch: env/dbowie
save and commit your change	<pre>git add dev/bigbang.yaml git status git commit -m "updated dev/bigbang.yaml" git push  OR sudo git push --set-upstream origin env/lm</pre>

### Deploy:

Modify Scripts/secrets.sh file – Add credentials info.	<p>1. Folder: DSOP_ENVIRONMENT/Scripts.</p> <p>Copy <b>secrets.sh.sample</b> to <b>secrets.sh</b> and edit to with your own values and secrets as follows:</p> <ul style="list-style-type: none"> <li>• Set IRON_BANK_USER &amp; IRON_BANK_PAT with the Username and CLI secret from your User Profile on <a href="https://registry1.dso.mil">https://registry1.dso.mil</a> (After logging in click your username in the upper righthand corner).</li> <li>• Set AZDO_USER &amp; AZDO_PASSWORD with the credentials you generated in step 2</li> <li>• Set ISTIO_GW_CRT &amp; ISTIO_GW_KEY with the certificates from step 3.</li> </ul> <p>To get the IronBank credentials:</p> <p>Login: <a href="https://registry1.dso.mil/harbor/projects">https://registry1.dso.mil/harbor/projects</a></p> <p>IRON_BANK_USER = <b>Username</b>  IRON_BANK_PAT = <b>CLI Secret</b></p>
--------------------------------------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

	
Application specific changes, ex: NAMESPACE="bigbang"	<p><b>Folder: DSOP_ENVIRONMENT/Scripts</b></p> <p>Copy deploy-vars.sh.sample to deploy-vars.sh and configure as you wish</p>
Run the automated deployment script	<p>cd scripts</p> <p><b>./deploy.sh</b></p> <p>(Keep your Azure DevOps User ID and Password Handy, you might need, deployment will ask multiple times)</p> <p><b>Note:</b></p>
This script will carry out the following:	<ol style="list-style-type: none"> <li>1. One time creation of GPG keys and update to .sops.yaml if keys are found to exist, this step is skipped.</li> <li>2. Creation/update of secrets.enc.yaml and pushed with git</li> <li>3. <i>OPTIONAL: Deployment of AKS cluster.</i></li> <li>4. <i>OPTIONAL: Connection to AKS cluster for kubectl etc</i></li> <li>5. Creation of namespaces: bigbang and flux-system</li> <li>6. Creation of secrets: sops-gpg, private-registry &amp; private-git</li> <li>7. Deployment of Flux from the main bigbang repo which will be cloned and scripts/install_flux.sh run. This can be disabled by setting DEPLOY_FLUX=false.</li> </ol>

	8. Removes network policies which block Flux being scraped 9. Deploys the dev/bigbang.yaml to the cluster 10. Validates the status of the deployment
Status of what was just deployed.	kubectl get gitrepositories,ks,hr -A kubectl get pods -A kubectl get nodes kubectl get vs -A (Show the Hosts Information) kubectl get all -n hello-world (Run this command, if hello-world deployed using GitOps)

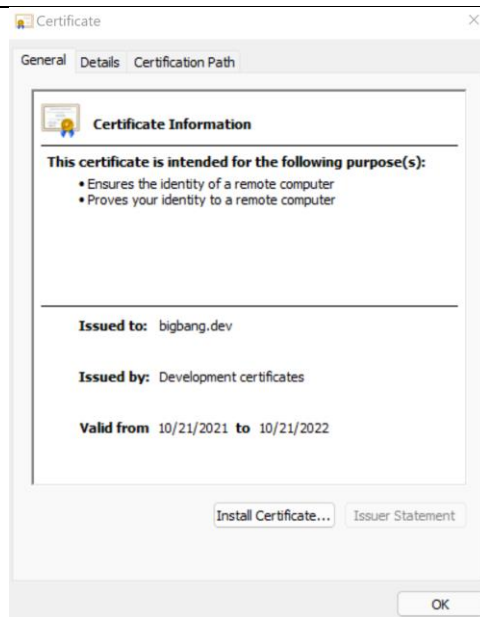
### Configure local domain to IP address mapping:

<b>IP address mapping</b>	In dev, when using a domain name not recorded in a DNS server, if we want to access the virtual services created by Bigbang, we can add the IP address - domain mapping to /etc/hosts running the following commands:
<b>Get IP Address</b>	Go To the folder: /mnt/c/Work/lmco/dsop-rke2/example ( <b>RKE2 setup location</b> ) <b># get istio gateway ip</b> ip=\$(kubectl -n istio-system get service istio-ingressgateway -o jsonpath='{.status.loadBalancer.ingress[0].ip}') <b># get domains</b> domains=\$(kubectl --kubeconfig rke2.kubeconfig get virtualservices -A -o jsonpath='{.items[*].spec.hosts[*]}')

	<pre>echo \$domains  # add entry in /etc/hosts  echo "\$ip \$domains"   sudo tee -a /etc/hosts</pre>
	<pre>Windows Hostfile Location C:\Windows\System32\drivers\etc 52.245.218.56 tracing.bigbang.dev 52.245.218.56 kiali.bigbang.dev 52.245.218.56 kibana.bigbang.dev 52.245.218.56 alertmanager.bigbang.dev 52.245.218.56 grafana.bigbang.dev 52.245.218.56 prometheus.bigbang.dev 52.245.218.56 twistlock.bigbang.dev 52.245.218.56 helloworld.bigbang.dev 52.245.218.56 currency-exchange.bigbang.dev</pre>

### Install the Certificates:

- ➔ Open the file bigbang.dev.cert & ca.cert from the location: \dsop-environment)
- ➔ Using Windows Explorer right mouse ➔ Open ➔



➔ Install Certificate button

## Welcome to the Certificate Import Wizard

This wizard helps you copy certificates, certificate trust lists, and certificate revocation lists from your disk to a certificate store.

A certificate, which is issued by a certification authority, is a confirmation of your identity and contains information used to protect data or to establish secure network connections. A certificate store is the system area where certificates are kept.

Store Location

☒ Current User

☐ Local Machine

To continue, click Next.

Next

Cancel





## Certificate Import Wizard

### Certificate Store

Certificate stores are system areas where certificates are kept.

Windows can automatically select a certificate store, or you can specify a location for the certificate.

☐ Automatically select the certificate store based on the type of certificate

☒ Place all certificates in the following store

Certificate store:

Trusted Root Certification Authorities

Browse...

Next

Cancel

← Certificate Import Wizard

Completing the Certificate Import Wizard

The certificate will be imported after you click Finish.

You have specified the following settings:

Certificate Store Selected	Automatically determined by the wizard
Content	Certificate

Finish

Cancel

Certificate Import Wizard

i

The import was successful.

OK

**Note:** You might need to restart the pc, to take certificate changes effect.

## Test Bigbang deployment Using Browser:

How to get credentials:

[https://repo1.dso.mil/platform-one/big-bang/bigbang/-/blob/master/docs/guides/using\\_bigbang/default\\_credentials.md](https://repo1.dso.mil/platform-one/big-bang/bigbang/-/blob/master/docs/guides/using_bigbang/default_credentials.md)

Grafana	<a href="https://grafana.bigbang.dev/login">https://grafana.bigbang.dev/login</a>
Kiali	<a href="https://kiali.bigbang.dev/kiali">https://kiali.bigbang.dev/kiali</a>
Kibana	<a href="https://kibana.bigbang.dev/login?next=%2F">https://kibana.bigbang.dev/login?next=%2F</a>
TwistLock	<a href="https://twistlock.bigbang.dev/#!/login">https://twistlock.bigbang.dev/#!/login</a>
Prometheus alert manager	<a href="https://alertmanager.bigbang.dev/#!/alerts">https://alertmanager.bigbang.dev/#!/alerts</a>
Prometheus Graph	<a href="https://prometheus.bigbang.dev/graph">https://prometheus.bigbang.dev/graph</a>
Jaeger	<a href="https://tracing.bigbang.dev/search">https://tracing.bigbang.dev/search</a>
HelloWorld – GitOps Flux	<a href="https://helloworld.bigbang.dev/login">https://helloworld.bigbang.dev/login</a>

Grafana	User ID: admin / Password: prom-operator
Kiali	kubectrl get secret -n kiali   grep kiali-service-account-token   awk '{print \$1}'   xargs kubectrl get secret -n kiali -o go-template='{{.data.token   base64decode}}'
Kibana	User ID: elastic kubectrl get secrets -n logging logging-ek-es-elastic-user -o go-template='{{.data.elastic   base64decode}}'
TwistLock	<a href="https://twistlock.bigbang.dev/#!/login">https://twistlock.bigbang.dev/#!/login</a> Create Account, after login.
Prometheus alert manager	<a href="https://alertmanager.bigbang.dev/#!/alerts">https://alertmanager.bigbang.dev/#!/alerts</a>
Prometheus Graph	<a href="https://prometheus.bigbang.dev/graph">https://prometheus.bigbang.dev/graph</a>
Jaeger	<a href="https://tracing.bigbang.dev/search">https://tracing.bigbang.dev/search</a>
HelloWorld – GitOps Flux	<a href="https://helloworld.bigbang.dev/login">https://helloworld.bigbang.dev/login</a>

## Test Bigbang deployment Using Python Scripts :

Install the Python3	Go to → cd dsop-environment/tests  This test required, Python 3.8.10: sudo apt install python3.8-venv python3 --version
	Tests are written in python; in order to run them follow the steps below in the dsop-environment directory: /dsop-environment
1. Create virtual environment	sudo apt-get install python3-venv sudo apt-get install pip sudo /usr/bin/python3 -m venv ./venv
2. Activate environment	source ./venv/bin/activate
3.Install requirements	cd .. (Should be in the folder location: dsop-environment)  pip install -r requirements.txt
4.Run tests	pytest ./tests -v
5.Test Output	<pre>&gt; pytest ./tests -v  ===== test session starts ===== platform linux -- Python 3.8.10, pytest-6.2.4, py-1.10.0, pluggy-0.13.1 -- /mnt/c/Work/lmco/lmco/dsop-environment/venv/bin/python3 cachedir: .pytest_cache rootdir: /mnt/c/Work/lmco/lmco/dsop-environment collected 5 items  tests/test_bigbang_k8s_resources.py::test_namespaces_created PASSED [ 20%] tests/test_bigbang_k8s_resources.py::test_successful_pod_status PASSED [ 40%] tests/test_bigbang_k8s_resources.py::test_successful_deployment_status PASSED [ 60%] tests/test_bigbang_k8s_resources.py::test_virtual_services_deployed PASSED [ 80%] tests/test_bigbang_services.py::test_services_are_reachable PASSED [100%]</pre>

