Data Service Server Admin’ s Manual

V1.0.3

ADVANTECH WISE-PaaS IoT Edge Intelligence

Revision History

Date	Version	Author	Reviewer	Description
2019-03-19	1.0.0	Zach Chih	Zach Chih	First version released
2019-05-28	1.0.1	Zach Chih	Zach Chih	Access Methods/Knowledge
2019-06-13	1.0.2	Zach Chih	Zach Chih	Helm/Kubectl
2019-06-21	1.0.3	Zach Chih	Zach Chih	Kubernetes Dashboard

Table of Contents

1 Preface

This manual is intended for new users with little or no experience using the Data Service Server. The goal of this document is to give a broad overview of the main functions of Data Service Server and some basic instructions on how to view and access the resources. This document will concentrate on demonstrating interaction with Data Service Server using the kubectl which is a command line tool used to communicate with Kubernetes.

Every effort has been made to ensure that this document is an accurate representation of the functionality of Data Service Server. As with every software application, development continues after the documentation has gone to press so small inconsistencies may occur. We would appreciate any feedback on this manual.

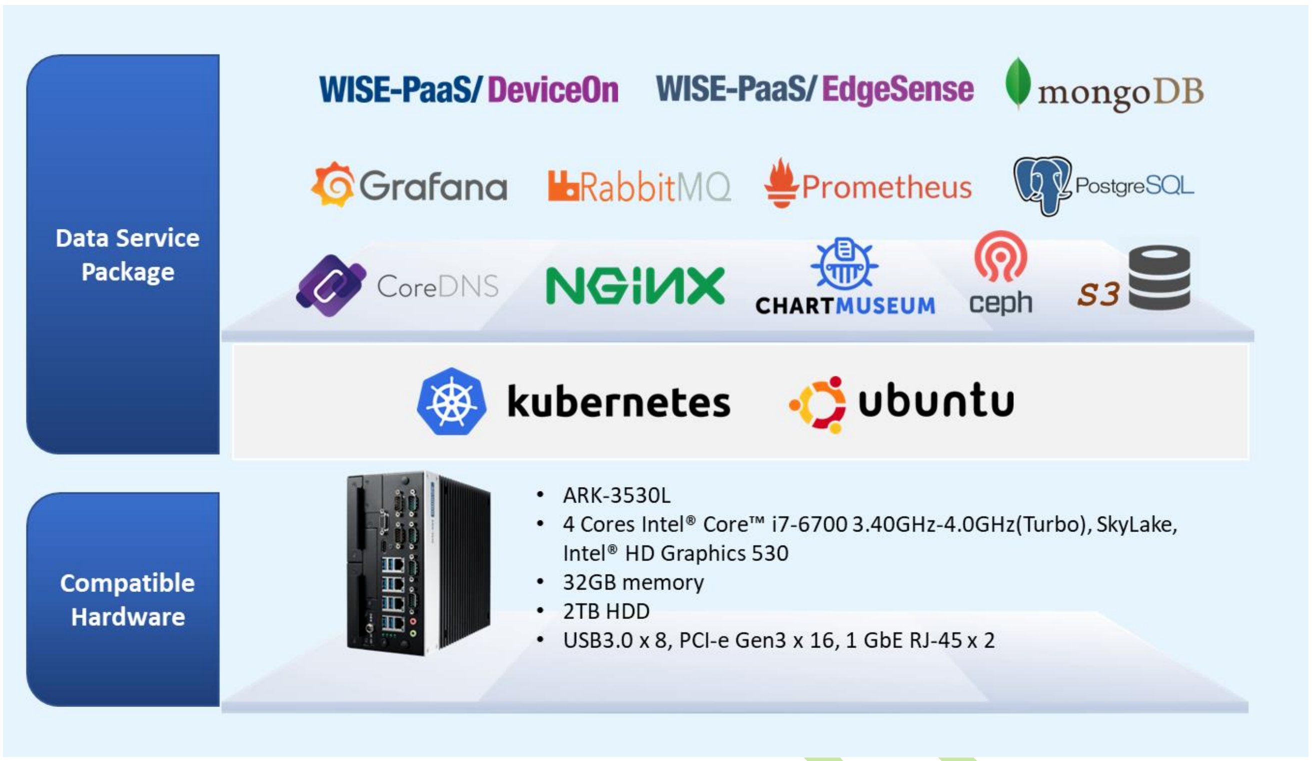
2 Introduction

If you’ re wondering what Data Service Server is, why you might want one and what it can do for you, then read on – you’ re about to find out.

At the simplest level, Data Service Server is three things: a data streaming, a data collection system and a monitoring system combined. Data streaming is a process in which every sensor data from devices could be upstream and data relating to controlling commands could be downstream. These two ways are the main communication channel by which end users could interact with their target devices. The interaction might be active or passive depending on your scenarios. The data collection process is in the middle of the whole streaming. Data Service Server would collect all the sensor data that users want and provide a user-friendly dashboard for user to manage and monitor the target devices. And Data Service Server has a backend monitoring system which is very powerful. The monitoring system could help users easily get to know the conditions and statuses of the components on the Data Service Server.

3 Architecture

Here’ s the architecture of Data Service Server. All of containerized applications are running based on a PaaS, which consists of kubernetes, docker and ceph. Kubernetes (k8s) is an open-source system for automating deployment, scaling, and management of containerized applications. [1] Docker is a computer program that performs operating-system-level virtualization, also known as "containerization". [2] Ceph is a free-software storage platform, implements object storage on a single distributed computer cluster, and provides interfaces for object-, block- and file-level storage. [3] By leveraging the power of these services behind the scenes, Data Service Server could be based on this robust and easily manageable ecosystem to provide preeminent data services for your devices.



4 Components

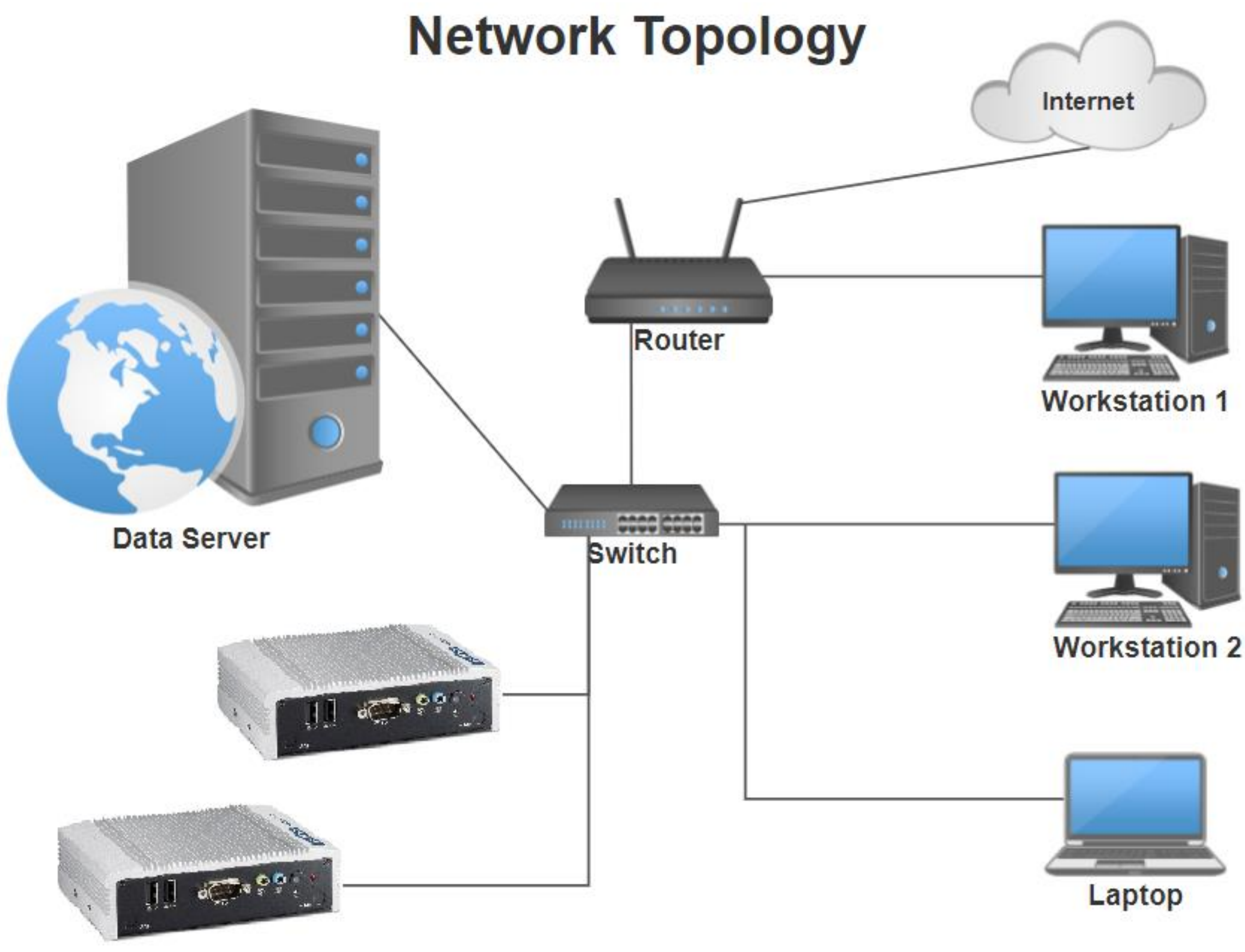
Kubernetes supports multiple virtual clusters backed by the same physical cluster. These virtual clusters are called namespaces. Namespaces are intended for use in environments with many users spread across multiple teams, or projects. [4] In Data Service Server, there’ re namespaces — default, network and monitoring, etc. The main components are in default namespace. The networking-related components are in network namespace. And the components used to keep an eye on the Data Service Server are in monitoring namespace.

Namespaces	Components
default	EdgeSense/DeviceOn
	RabbitMQ
	Postgres

	Mongo
	Minio
network	Nginx
	CoreDNS
monitoring	Prometheus
	Grafana
	Postgres-Exporter
	RabbitMQ-Exporter
Repository	ChartMuseum

5 Network Topology

The figure is an example network topology. Your workstations or laptops might be able to connect to Data Service Server through a router or a switch. And some of the host devices might connect to Data Service Server through the switch. The topology might be different from different users' working environments. **Just make sure your connections to Data Service Server work well.**



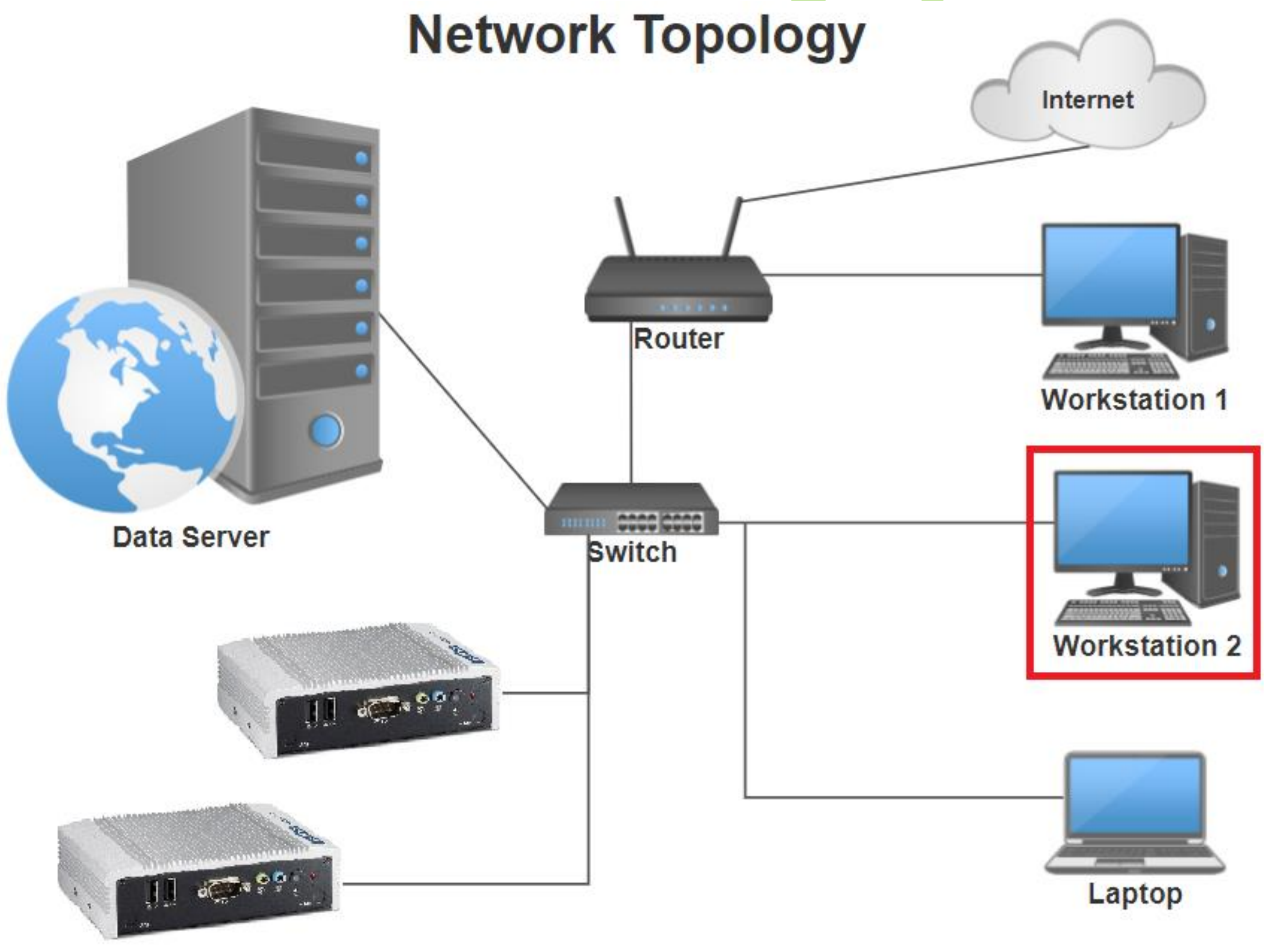
6 Starting Points

Data Service Server has two starting points to access and manage all the resources and applications running on itself. You could get started from your own PC or from Data Service Server.

6.1 Get Started from your PC

To have a better picture on what we are going to do, let’ s assume you’ re using the Workstation 2 as your PC and your connection to Data Service Server works well.

After setting up a good starting point, let’ s get started to set an environment to explore the resources on the Data Service Server by a configuration of authentication. Finally, you would know how to access the web services for management and monitoring.



6.1.1 Installing kubectl

Kubectl is a command line interface for running commands against Kubernetes clusters. [5] For details about each command, including all the supported flags and subcommands, see the kubectl reference documentation. For installation instructions see installing kubectl. [6] Installing kubectl in your PC is the first.

6.1.2 Logging in Data Service Server as Admin

To get a configuration and a certificate for authentication in the next section, as a user, you could login by Secure Shell (ssh) with default superuser (root). [7] The default username and password is root/r00tadvant1cH

6.1.3 Setting the Configuration for Authentication

Before exploring the resources on Data Service Server, you should set an environment with a configuration for authentication. There are two files you would need for the configuration. These files are in the following paths on the Data Service Server.

- **config — /root/config — the configuration file for authentication**
- **ca.pem — /root/ca — the certificate for accessing the Data Service Server**

Here’ s the content in config:

```
clusters:
- cluster:
    server: <server-ip>
  name: cluster.local
contexts:
- context:
    cluster: cluster.local
    user: root
  name: root-ctx
current-context: root-ctx
kind: Config
preferences: {}
users:
- name: root
  user:
    password: r00tadvant1cH
    username: root
```

First, copy the config file and ca.crt from Data Service Server into your computer and put them in the director `$HOME/.kube`. Then use the command below to set the certificate for the configuration.

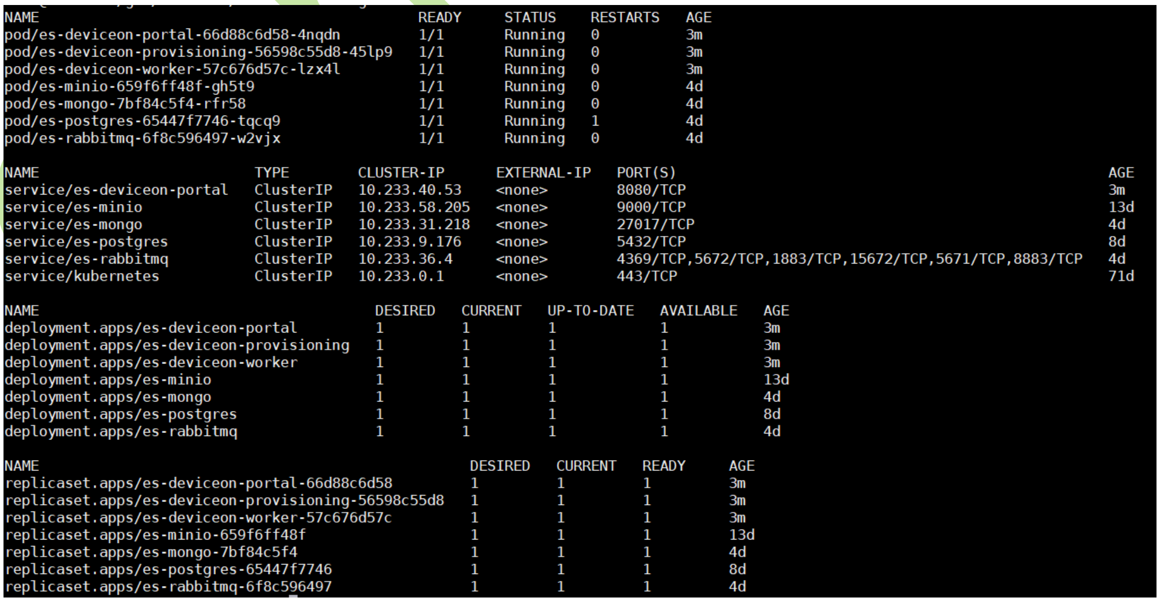
```
$ kubectl config set-cluster cluster.local --certificate-authority=<ca-path>
```

After the command, you would see the content of config like this:

```
apiVersion: v1
clusters:
- cluster:
    certificate-authority: <ca-path>
    server: <server-ip>
  name: cluster.local
contexts:
- context:
    cluster: cluster.local
    user: root
  name: root-ctx
current-context: root-ctx
kind: Config
preferences: {}
users:
- name: root
  user:
    password: r00tadvant1cH
    username: root
```

Then use the command below to check your configuration:

```
$ kubectl get all
```



```
NAME                                           READY   STATUS    RESTARTS   AGE
pod/es-deviceon-portal-66d88c6d58-4nqdn        1/1     Running   0          3m
pod/es-deviceon-provisioning-56598c55d8-45lp9  1/1     Running   0          3m
pod/es-deviceon-worker-57c676d57c-lzx4l        1/1     Running   0          3m
pod/es-minio-659f6ff48f-gh5t9                  1/1     Running   0          4d
pod/es-mongo-7bf84c5f4-rfr58                   1/1     Running   0          4d
pod/es-postgres-65447f7746-tqcq9               1/1     Running   1          4d
pod/es-rabbitmq-6f8c596497-w2vjx               1/1     Running   0          4d

NAME                          TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)                                                 AGE
service/es-deviceon-portal    ClusterIP   10.233.40.53    <none>        8080/TCP                                                3m
service/es-minio              ClusterIP   10.233.58.205   <none>        9000/TCP                                                13d
service/es-mongo              ClusterIP   10.233.31.218   <none>        27017/TCP                                               4d
service/es-postgres           ClusterIP   10.233.9.176    <none>        5432/TCP                                                8d
service/es-rabbitmq           ClusterIP   10.233.36.4     <none>        4369/TCP,5672/TCP,1883/TCP,15672/TCP,5671/TCP,8883/TCP  4d
service/kubernetes            ClusterIP   10.233.0.1      <none>        443/TCP                                                 71d

NAME                                       DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/es-deviceon-portal         1         1         1            1           3m
deployment.apps/es-deviceon-provisioning   1         1         1            1           3m
deployment.apps/es-deviceon-worker         1         1         1            1           3m
deployment.apps/es-minio                   1         1         1            1           13d
deployment.apps/es-mongo                   1         1         1            1           4d
deployment.apps/es-postgres                1         1         1            1           8d
deployment.apps/es-rabbitmq                1         1         1            1           4d

NAME                                                 DESIRED   CURRENT   READY   AGE
replicaset.apps/es-deviceon-portal-66d88c6d58        1         1         1       3m
replicaset.apps/es-deviceon-provisioning-56598c55d8  1         1         1       3m
replicaset.apps/es-deviceon-worker-57c676d57c        1         1         1       3m
replicaset.apps/es-minio-659f6ff48f                  1         1         1       13d
replicaset.apps/es-mongo-7bf84c5f4                   1         1         1       4d
replicaset.apps/es-postgres-65447f7746               1         1         1       8d
replicaset.apps/es-rabbitmq-6f8c596497               1         1         1       4d
```

If you could successfully get the resources on the Data Service Server, it means you already pass the authentication.

6.2 Get Started from Data Service Server

As a user, you could log in Data Service Server by remote desktop with default user (root). The default username and password is root/r00tadvant1cH

On the Data Service Server, you could skip section 6.1.1/6.1.2/6.1.3 because the kubectl and the configuration for authentication are pre-settled.

7 Accessing Web Services

Data Service Server provides two ways for you to access the web services like EdgeSense, DeviceOn and Grafana, etc. You could access the services directly through IP w/ Port or through DNS.

7.1 Through IP w/ Port

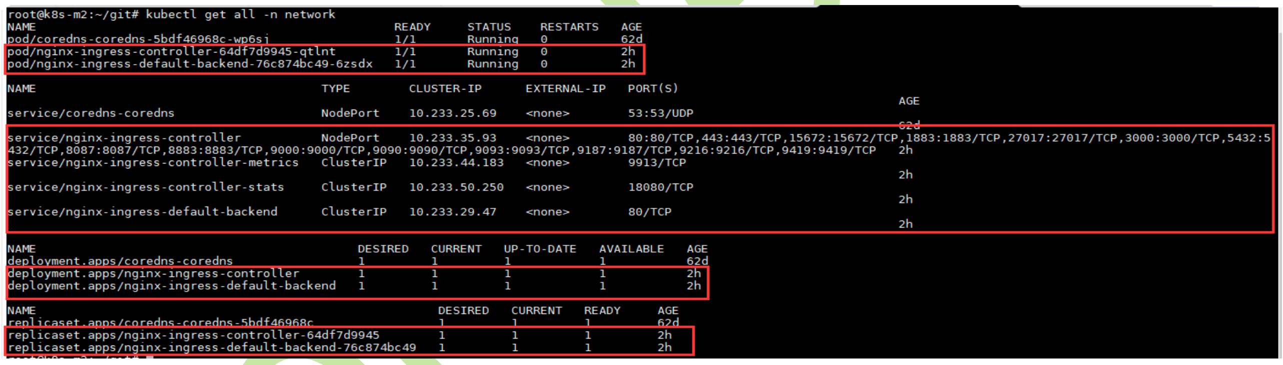
You could access the resources directly through IP with Port. This section takes DeviceOn as the example. The other resources could be accessed by the same way.

7.1.1 Viewing the Nginx Resources

Nginx [engine x] is an HTTP and reverse proxy server, a mail proxy server, and a generic TCP/UDP proxy server. [8] An Ingress Controller is a daemon, deployed as a Kubernetes Pod, that watches the apiserver's /ingresses endpoint for updates to the Ingress resource. [9] Its job is to satisfy requests for Ingresses. The section is to view the resources of nginx-ingress controller.

The command below would show the resources related to nginx in the namespace network and check the statuses are running:

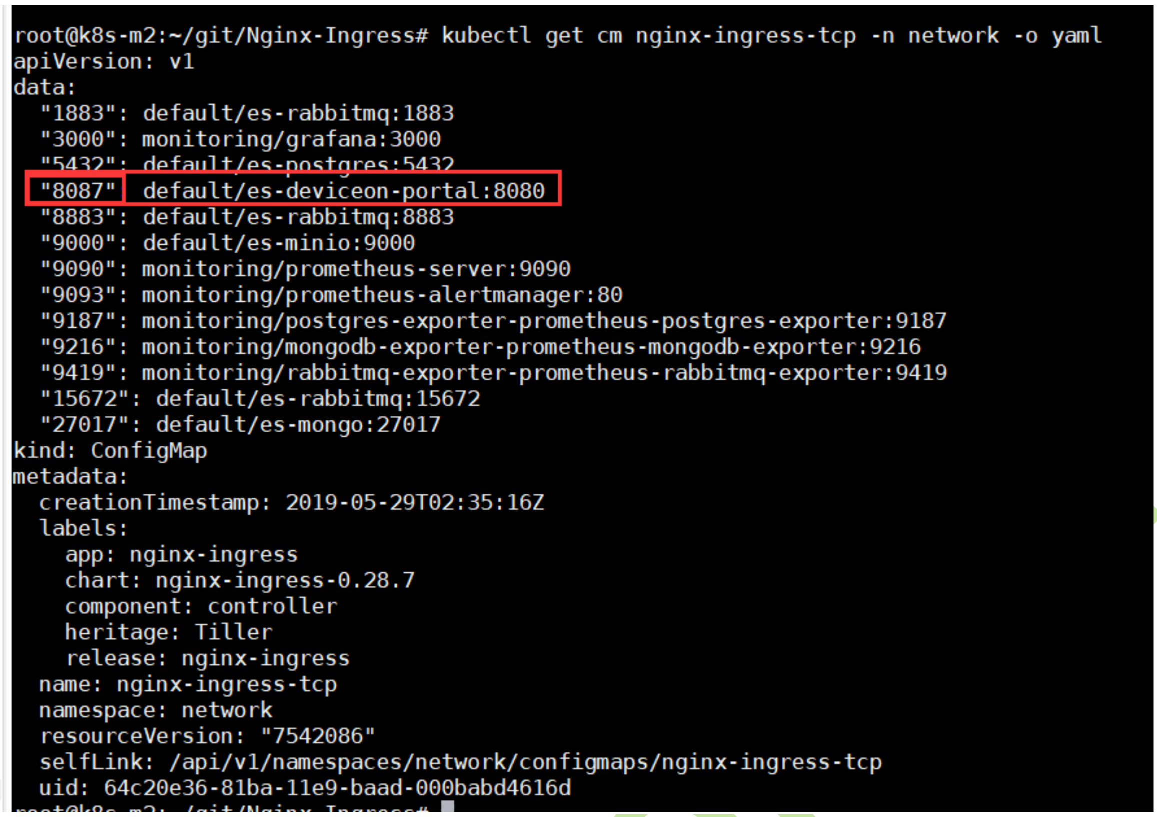
```
$ kubectl get all -n network
```



```
root@k8s-m2:~/git# kubectl get all -n network
NAME                                             READY     STATUS    RESTARTS   AGE
pod/coredns-coredns-5bdf46968c-wp6sj             1/1       Running   0          62d
pod/nginx-ingress-controller-64df7d9945-qtlnt    1/1       Running   0          2h
pod/nginx-ingress-default-backend-76c874bc49-6zsdx   1/1   Running   0          2h

NAME                                     TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)
                                                                                                                     AGE
service/coredns-coredns                  NodePort    10.233.25.69    <none>        53:53/UDP
                                                                                                                     62d
service/nginx-ingress-controller         NodePort    10.233.35.93    <none>        80:80/TCP,443:443/TCP,15672:15672/TCP,1883:1883/TCP,27017:27017/TCP,3000:3000/TCP,5432:5
432/TCP,8087:8087/TCP,8883:8883/TCP,9000:9000/TCP,9090:9090/TCP,9093:9093/TCP,9187:9187/TCP,9216:9216/TCP,9419:9419/TCP   2h
service/nginx-ingress-controller-metrics ClusterIP   10.233.44.183   <none>        9913/TCP
                                                                                                                     2h
service/nginx-ingress-controller-stats   ClusterIP   10.233.50.250   <none>        18080/TCP
                                                                                                                     2h
service/nginx-ingress-default-backend    ClusterIP   10.233.29.47    <none>        80/TCP
                                                                                                                     2h

NAME                                            DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/coredns-coredns                 1         1         1            1           62d
deployment.apps/nginx-ingress-controller        1         1         1            1           2h
deployment.apps/nginx-ingress-default-backend   1         1         1            1           2h

NAME                                                   DESIRED   CURRENT   READY     AGE
replicaset.apps/coredns-coredns-5bdf46968c             1         1         1         62d
replicaset.apps/nginx-ingress-controller-64df7d9945    1         1         1         2h
replicaset.apps/nginx-ingress-default-backend-76c874bc49   1     1         1         2h
```

Then use the command below to check the exposed port of DeviceOn:

```
$ kubectl get cm nginx-ingress-tcp -n network -o yaml
```



```
root@k8s-m2:~/git/Nginx-Ingress# kubectl get cm nginx-ingress-tcp -n network -o yaml
apiVersion: v1
data:
  "1883": default/es-rabbitmq:1883
  "3000": monitoring/grafana:3000
  "5432": default/es-postgres:5432
  "8087": default/es-deviceon-portal:8080
  "8883": default/es-rabbitmq:8883
  "9000": default/es-minio:9000
  "9090": monitoring/prometheus-server:9090
  "9093": monitoring/prometheus-alertmanager:80
  "9187": monitoring/postgres-exporter-prometheus-postgres-exporter:9187
  "9216": monitoring/mongodb-exporter-prometheus-mongodb-exporter:9216
  "9419": monitoring/rabbitmq-exporter-prometheus-rabbitmq-exporter:9419
  "15672": default/es-rabbitmq:15672
  "27017": default/es-mongo:27017
kind: ConfigMap
metadata:
  creationTimestamp: 2019-05-29T02:35:16Z
  labels:
    app: nginx-ingress
    chart: nginx-ingress-0.28.7
    component: controller
    heritage: Tiller
    release: nginx-ingress
  name: nginx-ingress-tcp
  namespace: network
  resourceVersion: "7542086"
  selfLink: /api/v1/namespaces/network/configmaps/nginx-ingress-tcp
  uid: 64c20e36-81ba-11e9-baad-000babd4616d
```

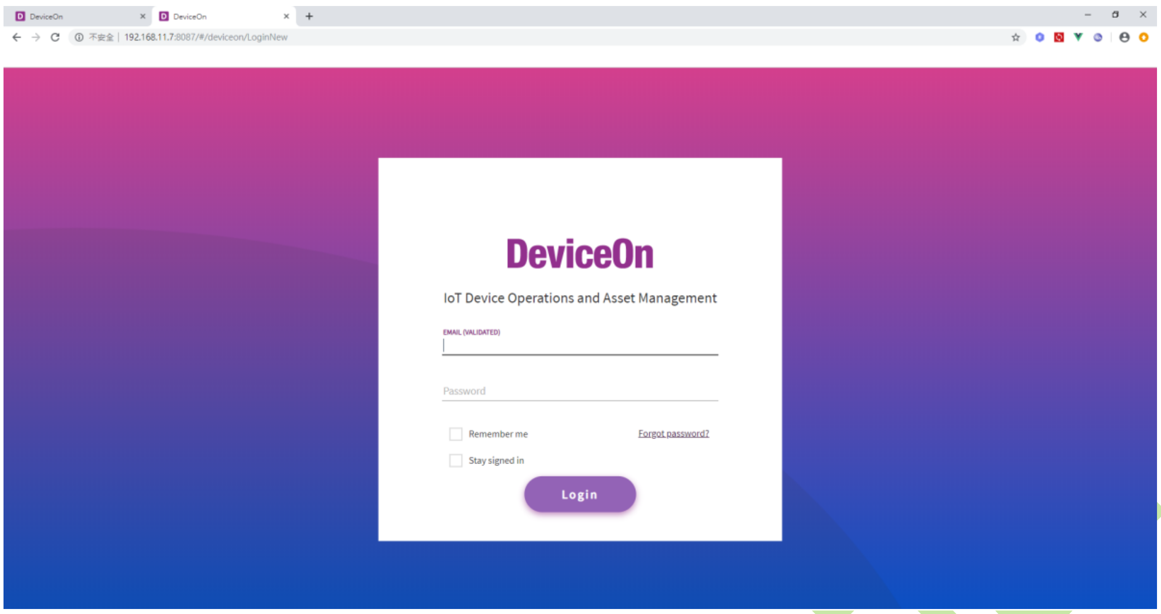
We could see the exposed port for DeviceOn is 8087. (8080 on the right is internal port)

You should use exposed port on the left, instead.

7.1.2 Accessing Web Services

Open the browser to access DeviceOn.

IP Access — http://<DataServer-IP>:8087



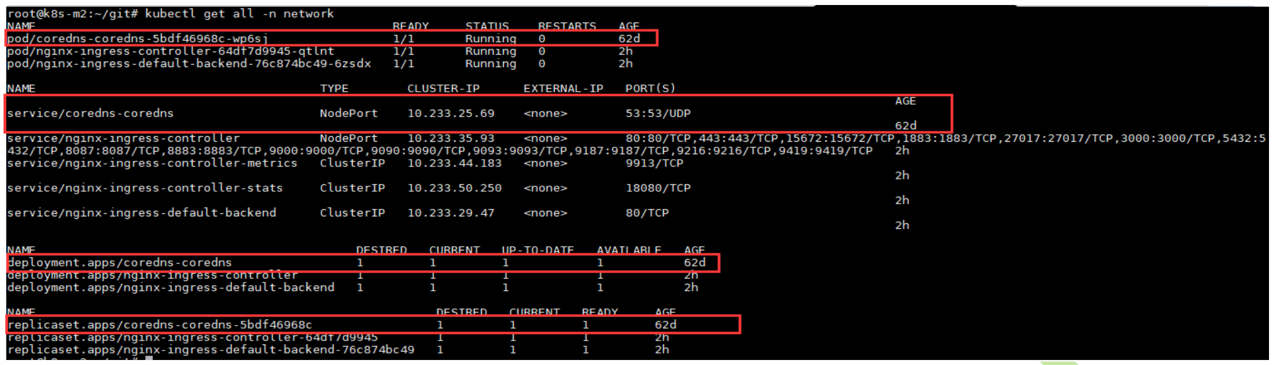
7.2 Through DNS

The other way to access the resources is through DNS. This section takes DeviceOn as the example. Still, make sure nginx is healthy because it' s the starting point for any access methods. The other resources could be accessed by the same way.

7.2.1 Viewing the DNS Resources

The DNS Sever in Data Service Server is CoreDNS. Use the command below to view the resources related to coredns in namespace network and check the statuses are running:

```
$ kubectl get all -n network
```



```
root@k8s-m2:~/git# kubectl get all -n network
NAME                                            READY     STATUS    RESTARTS   AGE
pod/coredns-coredns-5bdf46968c-wp6sj            1/1       Running   0          62d
pod/nginx-ingress-controller-64df7d9945-qtlnt   1/1       Running   0          2h
pod/nginx-ingress-default-backend-76c874bc49-6zsdx   1/1  Running   0          2h

NAME                                      TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)
                                                                                                                              AGE
service/coredns-coredns                   NodePort    10.233.25.69    <none>        53:53/UDP
                                                                                                                              62d
service/nginx-ingress-controller          NodePort    10.233.35.93    <none>        80:80/TCP,443:443/TCP,15672:15672/TCP,1883:1883/TCP,27017:27017/TCP,3000:3000/TCP,5432:5
432/TCP,8087:8087/TCP,8883:8883/TCP,9000:9000/TCP,9090:9090/TCP,9093:9093/TCP,9187:9187/TCP,9216:9216/TCP,9419:9419/TCP   2h
service/nginx-ingress-controller-metrics  ClusterIP   10.233.44.183   <none>        9913/TCP
                                                                                                                              2h
service/nginx-ingress-controller-stats    ClusterIP   10.233.50.250   <none>        18080/TCP
                                                                                                                              2h
service/nginx-ingress-default-backend     ClusterIP   10.233.29.47    <none>        80/TCP
                                                                                                                              2h

NAME                                           DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/coredns-coredns                1         1         1            1           62d
deployment.apps/nginx-ingress-controller       1         1         1            1           2h
deployment.apps/nginx-ingress-default-backend  1         1         1            1           2h

NAME                                                     DESIRED   CURRENT   READY     AGE
replicaset.apps/coredns-coredns-5bdf46968c               1         1         1         62d
replicaset.apps/nginx-ingress-controller-64df7d9945      1         1         1         2h
replicaset.apps/nginx-ingress-default-backend-76c874bc49 1         1         1         2h
```

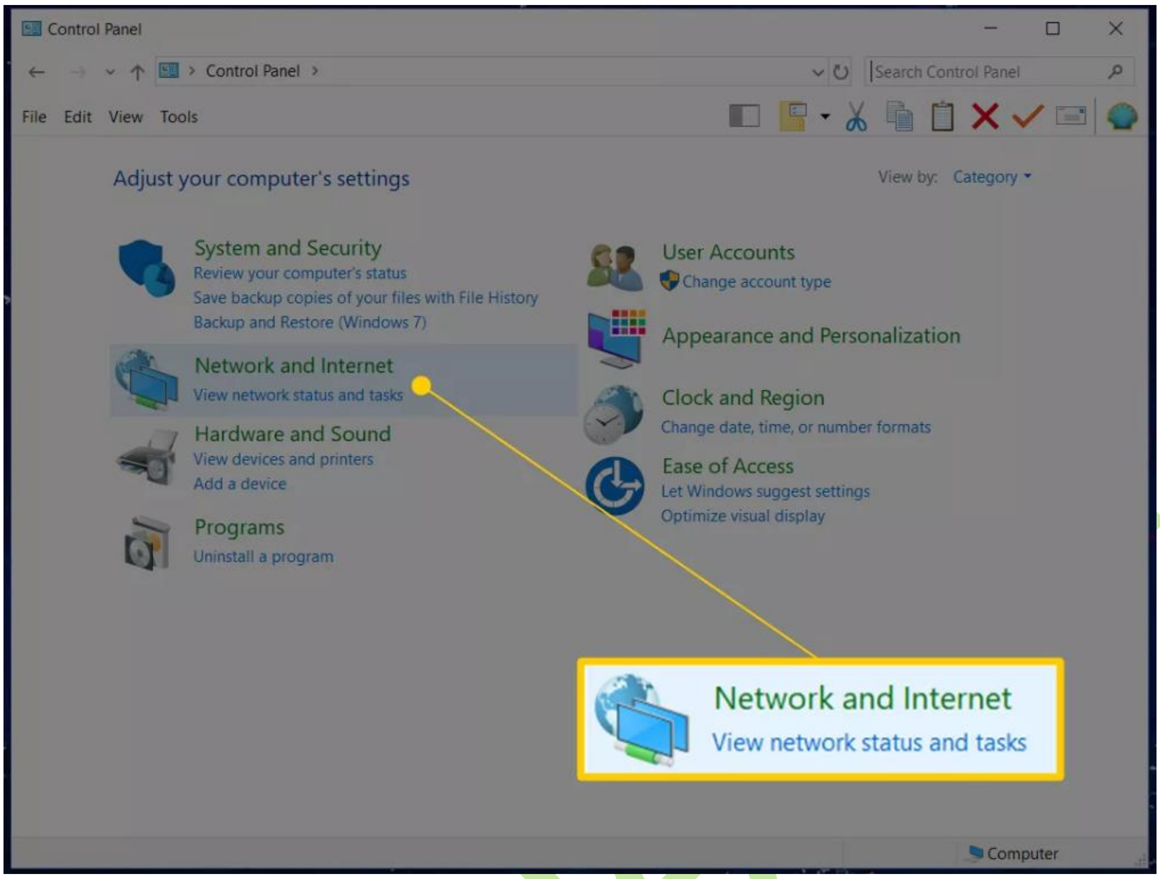
7.2.2 Setting the DNS

Before you try to connect to these services by these endpoints, you should set your DNS IP to the IP Data Service Server locates at. This is because the URL translation is being handled by the CoreDNS which the Data Service Server provides. The steps of setting the DNS Server are as follows. [10]

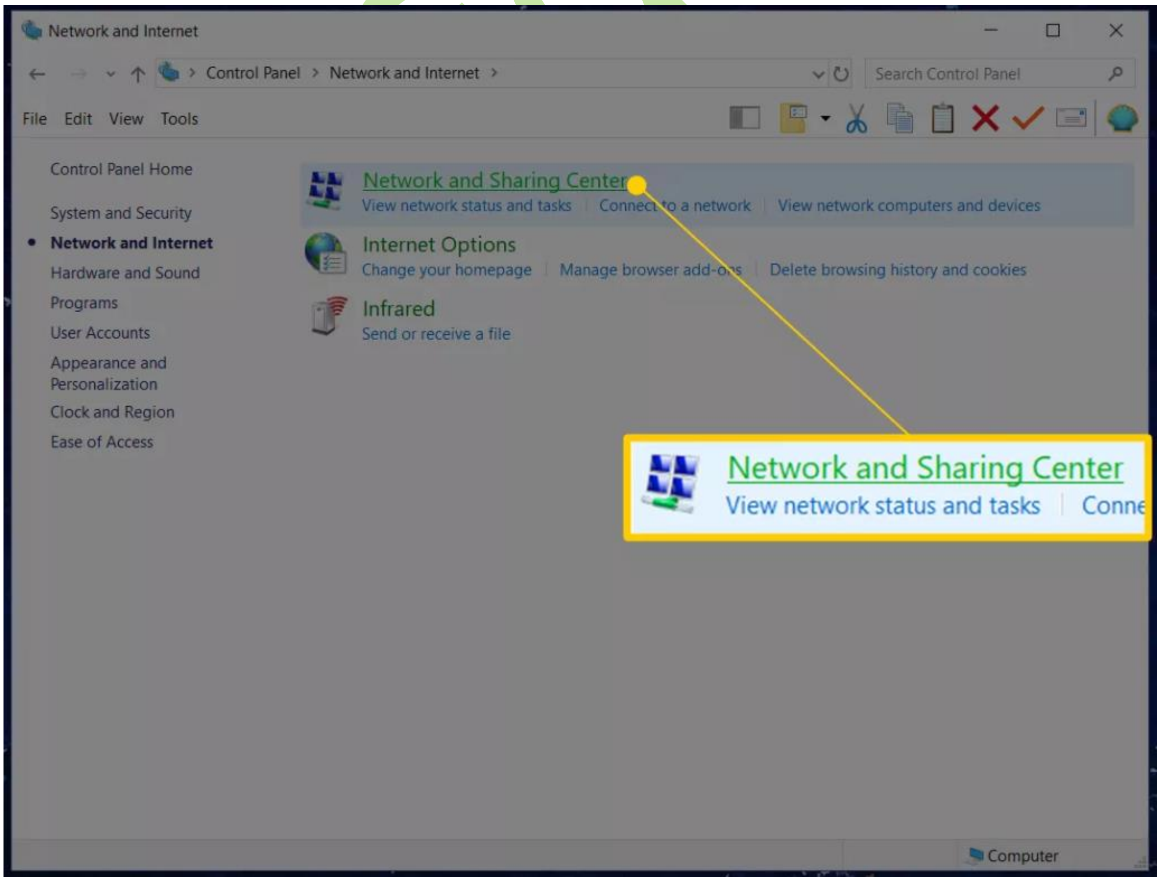
7.2.2.1 Windows

01. Open Control Panel.

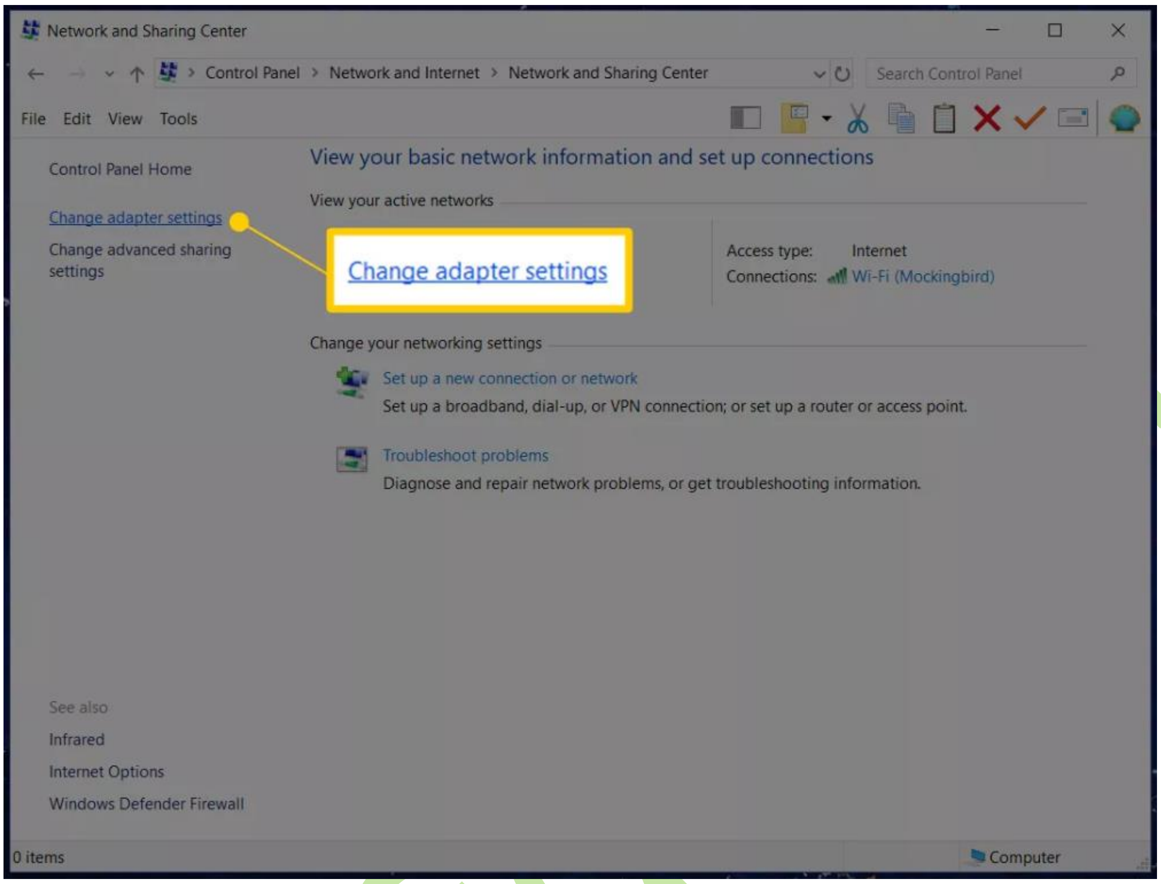
02. Once in Control Panel, touch or click **Network and Internet**.


Control Panel
Control Panel
Search Control Panel
File Edit View Tools
Adjust your computer's settings
View by: Category
System and Security
Review your computer's status
Save backup copies of your files with File History
Backup and Restore (Windows 7)
Network and Internet
View network status and tasks
Hardware and Sound
View devices and printers
Add a device
Programs
Uninstall a program
User Accounts
Change account type
Appearance and Personalization
Clock and Region
Change date, time, or number formats
Ease of Access
Let Windows suggest settings
Optimize visual display
Network and Internet
View network status and tasks
Computer

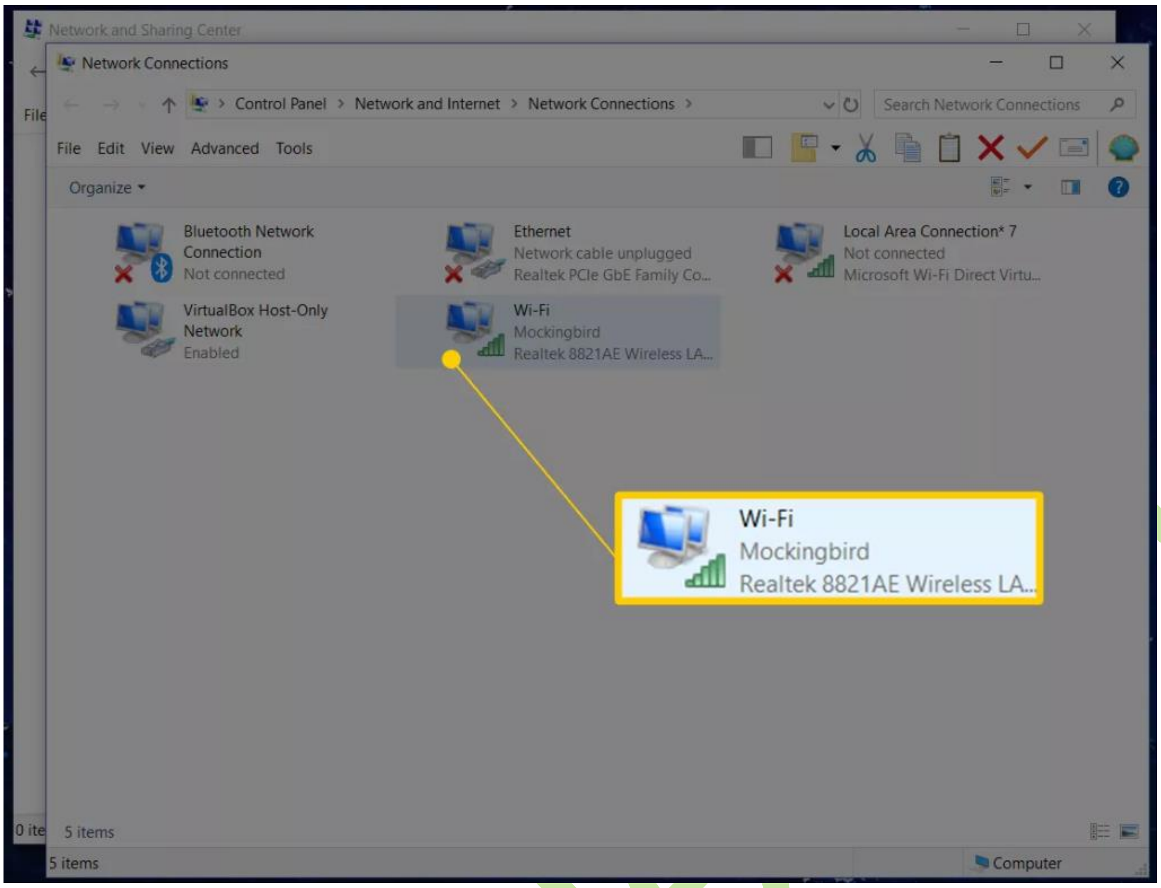
03. In the Network and Internet window that's now open, click or touch **Network and Sharing Center** to open that applet.



04. Now that the Network and Sharing Center window is open, click or touch **Change adapter settings**, located in the left margin.

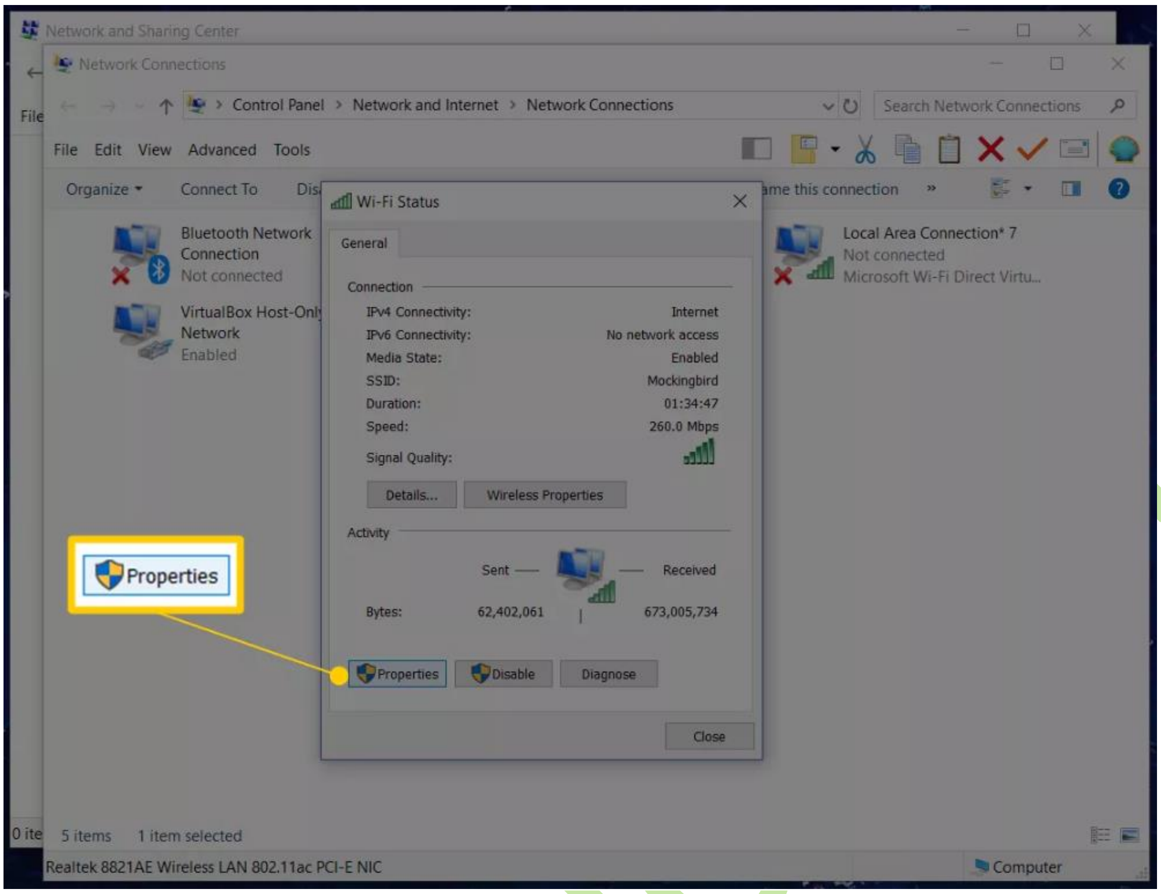

Network and Sharing Center
Control Panel > Network and Internet > Network and Sharing Center
Search Control Panel
File Edit View Tools
Control Panel Home
Change adapter settings
Change advanced sharing settings
View your basic network information and set up connections
View your active networks
Change adapter settings
Access type: Internet
Connections: Wi-Fi (Mockingbird)
Change your networking settings
Set up a new connection or network
Set up a broadband, dial-up, or VPN connection; or set up a router or access point.
Troubleshoot problems
Diagnose and repair network problems, or get troubleshooting information.
See also
Infrared
Internet Options
Windows Defender Firewall
0 items
Computer

05. From this new Network Connections screen, locate the network connection that you want to change the DNS servers for.


Network and Sharing Center
Network Connections
Control Panel > Network and Internet > Network Connections
Search Network Connections
File Edit View Advanced Tools
Organize
Bluetooth Network Connection
Not connected
Ethernet
Network cable unplugged
Realtek PCIe GbE Family Co...
Local Area Connection* 7
Not connected
Microsoft Wi-Fi Direct Virtu...
VirtualBox Host-Only Network
Enabled
Wi-Fi
Mockingbird
Realtek 8821AE Wireless LA...
5 items
Computer

06. Open the network connection you want to change the DNS servers for by double-clicking or double-tapping on its icon.

07. Tap or click **Properties** on the connection's Status window that's now open.


Network and Sharing Center
Network Connections
Control Panel > Network and Internet > Network Connections
Search Network Connections
File Edit View Advanced Tools
Organize Connect To
Bluetooth Network Connection
Not connected
VirtualBox Host-Only Network
Enabled
Local Area Connection* 7
Not connected
Microsoft Wi-Fi Direct Virtu...
Properties
5 items 1 item selected
Realtek 8821AE Wireless LAN 802.11ac PCI-E NIC
Computer

Wi-Fi Status

General

Connection

IPv4 Connectivity:	Internet
IPv6 Connectivity:	No network access
Media State:	Enabled
SSID:	Mockingbird
Duration:	01:34:47
Speed:	260.0 Mbps
Signal Quality:	

Details... Wireless Properties

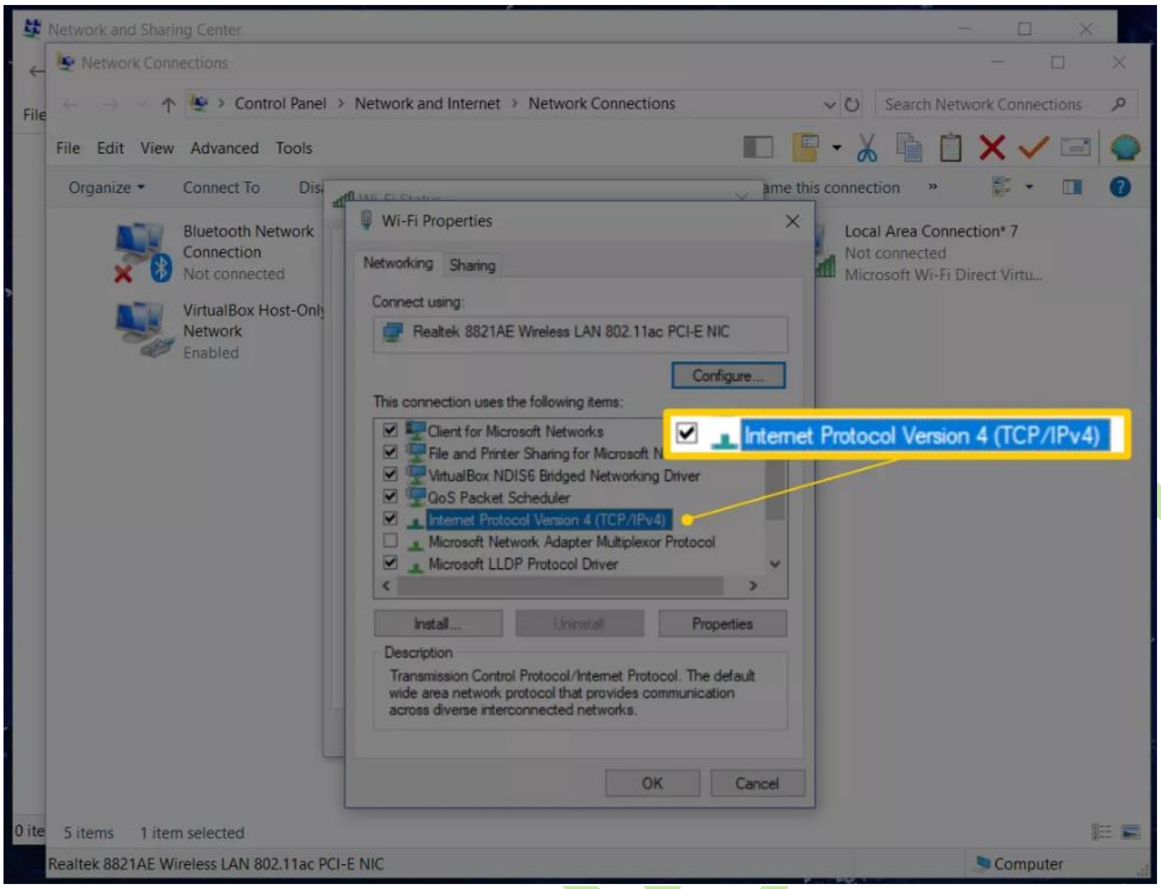
Activity

	Sent	Received
Bytes:	62,402,061	673,005,734

Properties Disable Diagnose

Close

08. On the connection's Properties window that appeared, scroll down in the *This connection uses the following items*: list and click or tap **Internet Protocol Version 4 (TCP/IPv4)** or **Internet Protocol (TCP/IP)** to select the IPv4 option to change the IPv4 DNS server settings.


Network and Sharing Center
Network Connections
Control Panel > Network and Internet > Network Connections
Search Network Connections
File Edit View Advanced Tools
Organize Connect To
Bluetooth Network Connection
Not connected
VirtualBox Host-Only Network
Enabled
Local Area Connection* 7
Not connected
Microsoft Wi-Fi Direct Virtu...
Wi-Fi Properties
Networking Sharing
Install... Uninstall Properties
OK Cancel
5 items 1 item selected
Computer

Connect using:

Realtek 8821AE Wireless LAN 802.11ac PCI-E NIC

Configure...

This connection uses the following items:

- [✓] Client for Microsoft Networks
- [✓] File and Printer Sharing for Microsoft N
- [✓] VirtualBox NDIS6 Bridged Networking Driver
- [✓] QoS Packet Scheduler
- [✓] Internet Protocol Version 4 (TCP/IPv4)
- [ ] Microsoft Network Adapter Multiplexor Protocol
- [✓] Microsoft LLDP Protocol Driver

Description

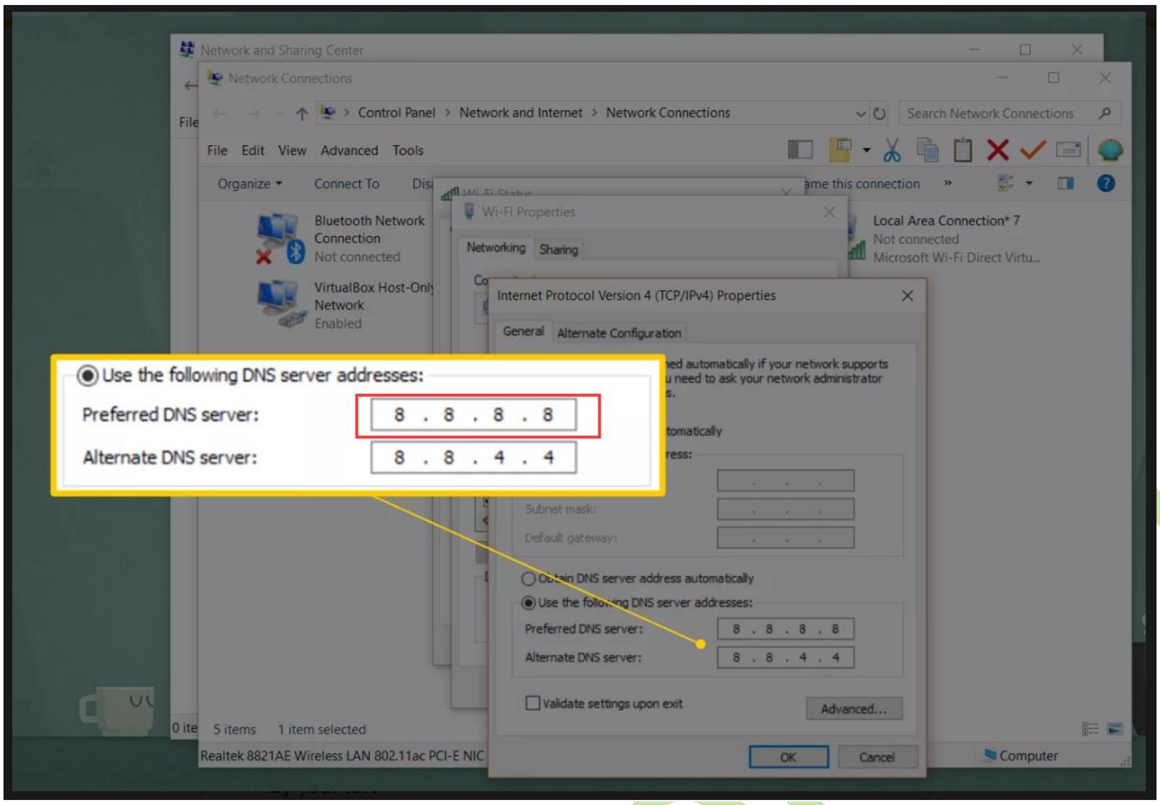
Transmission Control Protocol/Internet Protocol. The default wide area network protocol that provides communication across diverse interconnected networks.

Realtek 8821AE Wireless LAN 802.11ac PCI-E NIC

09. Tap or click **Properties** below the list.

10. Choose the **Use the following DNS server addresses**: radio button at the bottom of the Internet Protocol Properties window.

11. In the spaces provided, enter the IP address for the **CoreDNS server**.


Network and Sharing Center
Network Connections
Control Panel > Network and Internet > Network Connections
Search Network Connections
File Edit View Advanced Tools
Organize
Connect To
Bluetooth Network Connection
Not connected
VirtualBox Host-Only Network
Enabled
Local Area Connection* 7
Not connected
Microsoft Wi-Fi Direct Virtu...
Wi-Fi Properties
Networking
Sharing
Internet Protocol Version 4 (TCP/IPv4) Properties
General
Alternate Configuration
Subnet mask:
Default gateway:
Obtain DNS server address automatically
Validate settings upon exit
Advanced...
OK
Cancel
5 items
1 item selected
Realtek 8821AE Wireless LAN 802.11ac PCI-E NIC
Computer

Use the following DNS server addresses:

Preferred DNS server: 8 . 8 . 8 . 8

Alternate DNS server: 8 . 8 . 4 . 4

12. Tap or click **OK**.

The DNS server change takes place immediately. You can now close any *Properties*, *Status*, *Network Connections*, or *Control Panel* windows that are open.

13. Then use $ nslookup <URL> to check if the URL translation is proper or not.

7.2.2.2 Linux

In linux, you should just use $ sudo vi /etc/resolv.conf to add "options rotate" and "nameserver <DataServer-IP>" to the first and the second line then save the file. Then you could use $ nslookup <URL> to check if the URL translation is proper or not.

7.2.3 Viewing the Endpoints (Ingress)

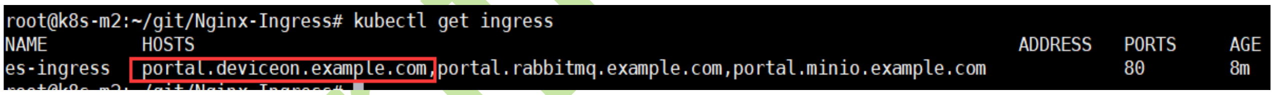
After viewing the resources related to networking in namespace network, what you must want to know is what the endpoints of web services are. Before that, you should know what ingress is.

Ingress, added in Kubernetes v1.1, exposes HTTP and HTTPS routes from outside the cluster to services within the cluster. Traffic routing is controlled by rules defined on the Ingress resource. [11]

Now, you could view the endpoints of services routing by the ingresses. There' re two ingresses, one in namespace default, the other in namespace network.

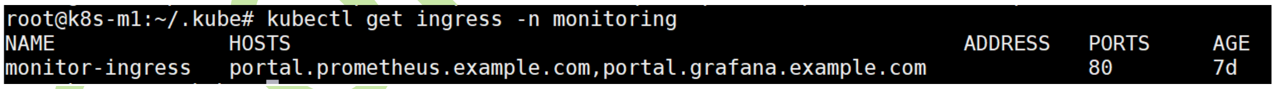
Type the command:

```
$ kubectl get ingress
```



```
root@k8s-m2:~/git/Nginx-Ingress# kubectl get ingress
NAME         HOSTS                                                                          ADDRESS   PORTS    AGE
es-ingress   portal.deviceon.example.com,portal.rabbitmq.example.com,portal.minio.example.com           80       8m
```

```
$ kubectl get ingress -n monitoring
```



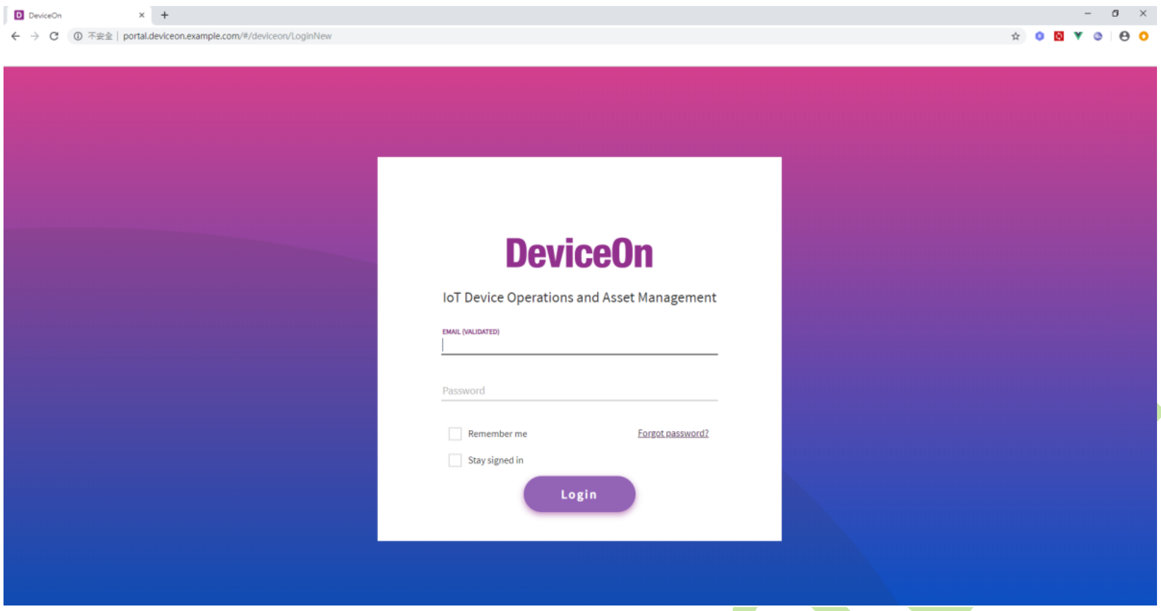
```
root@k8s-m1:~/.kube# kubectl get ingress -n monitoring
NAME              HOSTS                                                        ADDRESS    PORTS    AGE
monitor-ingress   portal.prometheus.example.com,portal.grafana.example.com               80       7d
```

The examples here are the endpoints based on domain of .example.com. You might see the domain different from this one. The URL for DeviceOn is portal.deviceon.example.com in this example.

7.2.4 Accessing Web Services

Open the browser to access DeviceOn by the endpoint.

Endpoint — http://portal.deviceon.example.com


DeviceOn
portal.deviceon.example.com/#/deviceon/LoginNew
DeviceOn
IoT Device Operations and Asset Management
EMAIL (VALIDATED)
Password
Remember me
Forgot password?
Stay signed in
Login

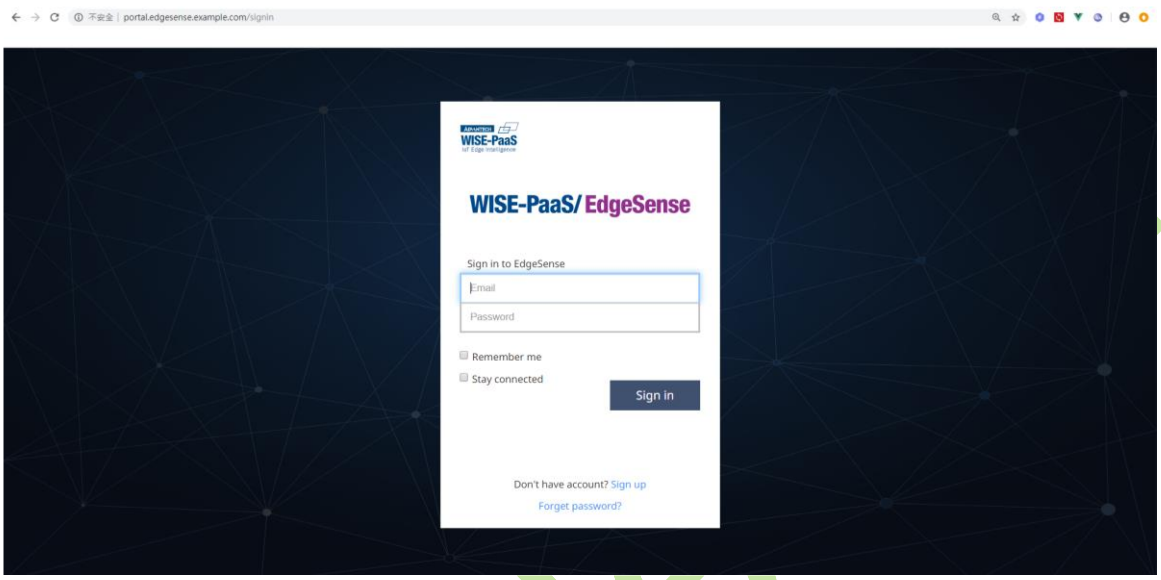
7.3 Summary: Access Methods

Here’ re the summary for both access methods: IP and DNS. After having the ports or endpoints of the services, you could easily access them by typing the ports and endpoints in the browser. For endpoints, the examples here are based on domain of .example.com. In your case, the domain might be different. You should get your endpoints by the previous steps. To get more info about these services, you could refer to the manuals for EdgeSense, DeviceOn and monitoring system. Please make sure what applications your Data Service Server includes.

7.3.1 EdgeSense

Endpoint — http://portal.edgesense.example.com

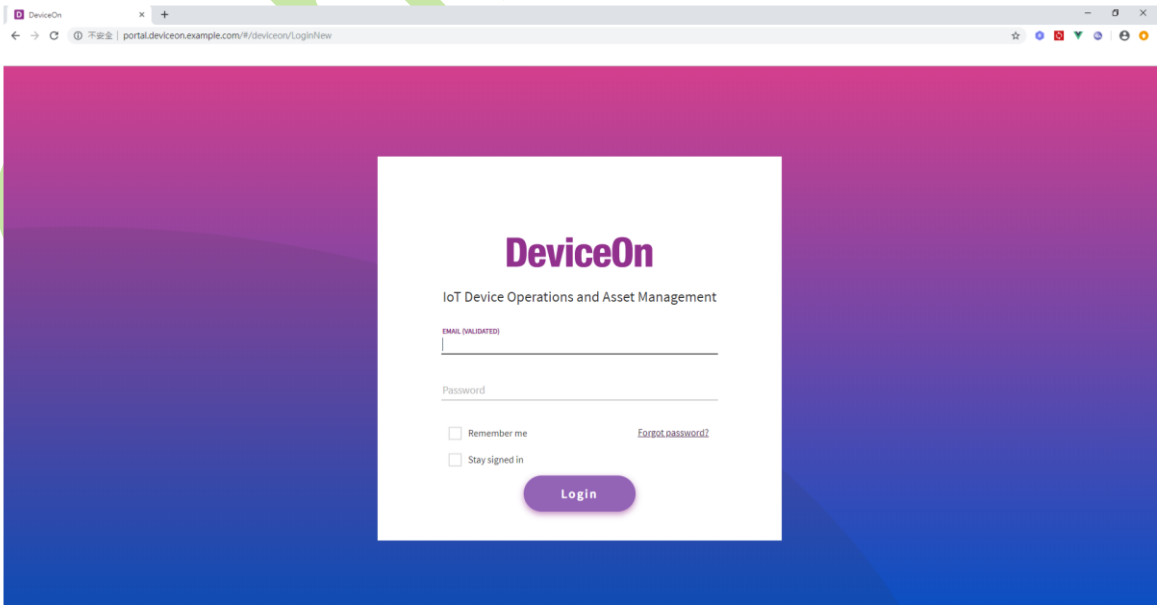
IP Access — http://<DataServer-IP>:8087



7.3.2 DeviceOn

Endpoint — http://portal.deviceon.example.com

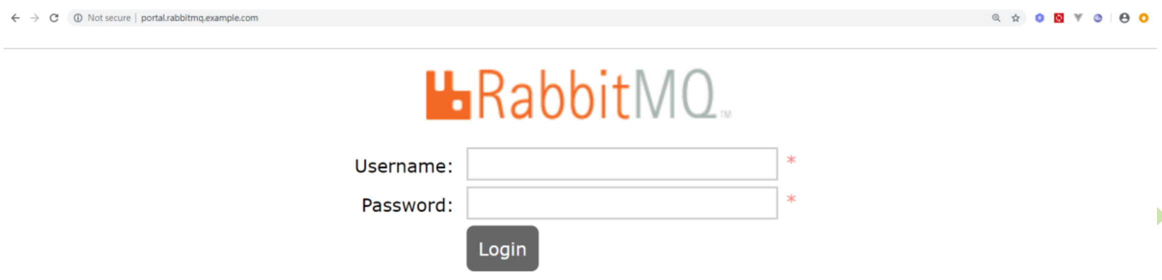
IP Access — http://<DataServer-IP>:8087


DeviceOn
portal.deviceon.example.com/#/deviceon/LoginNew
DeviceOn
IoT Device Operations and Asset Management
EMAIL (VALIDATED)
Password
Remember me
Forgot password?
Stay signed in
Login

7.3.3 RabbitMQ

Endpoint — http://portal.rabbitmq.example.com

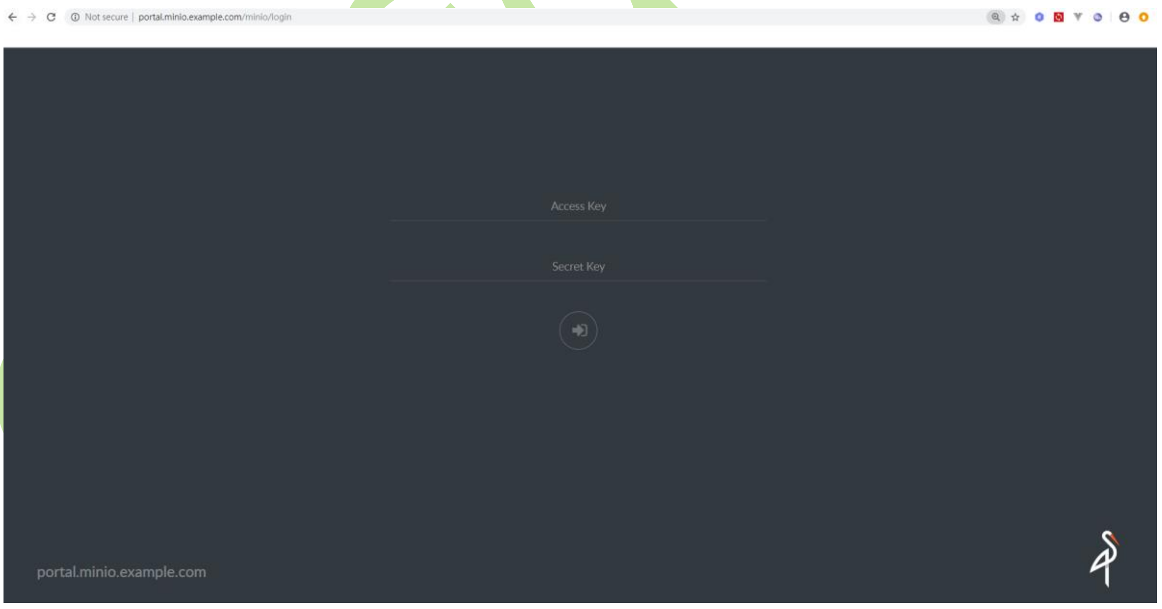
IP Access — http://<DataServer-IP>:15672



7.3.4 Minio

Endpoint — http://portal.minio.example.com

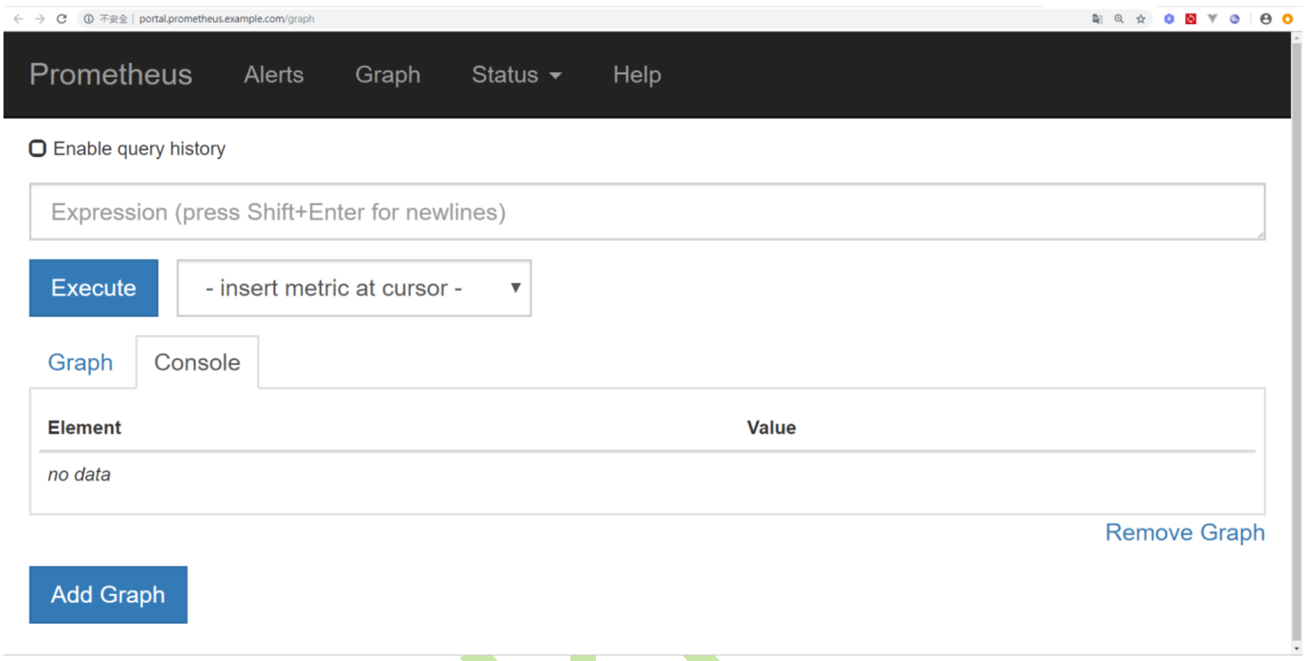
IP Access — http://<DataServer-IP>:9000



7.3.5 Prometheus

Endpoint — http://portal.prometheus.example.com

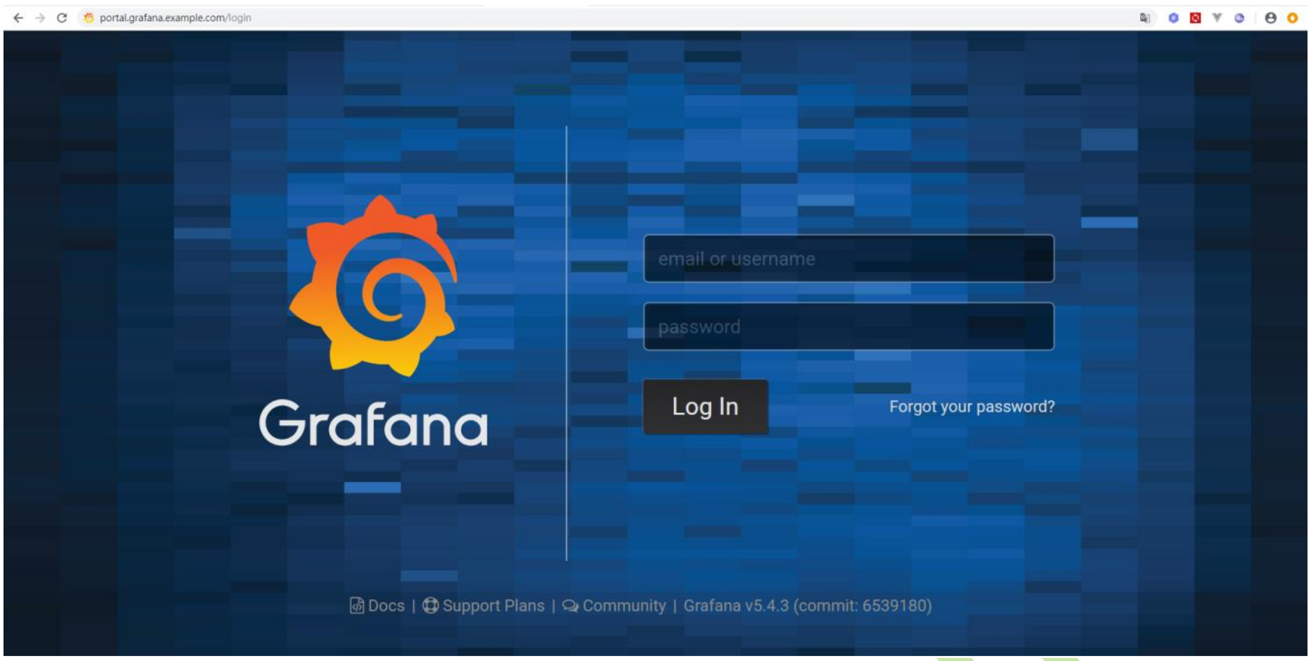
IP Access — http://<DataServer-IP>:9090



7.3.6 Grafana

Endpoint — http://portal.grafana.example.com

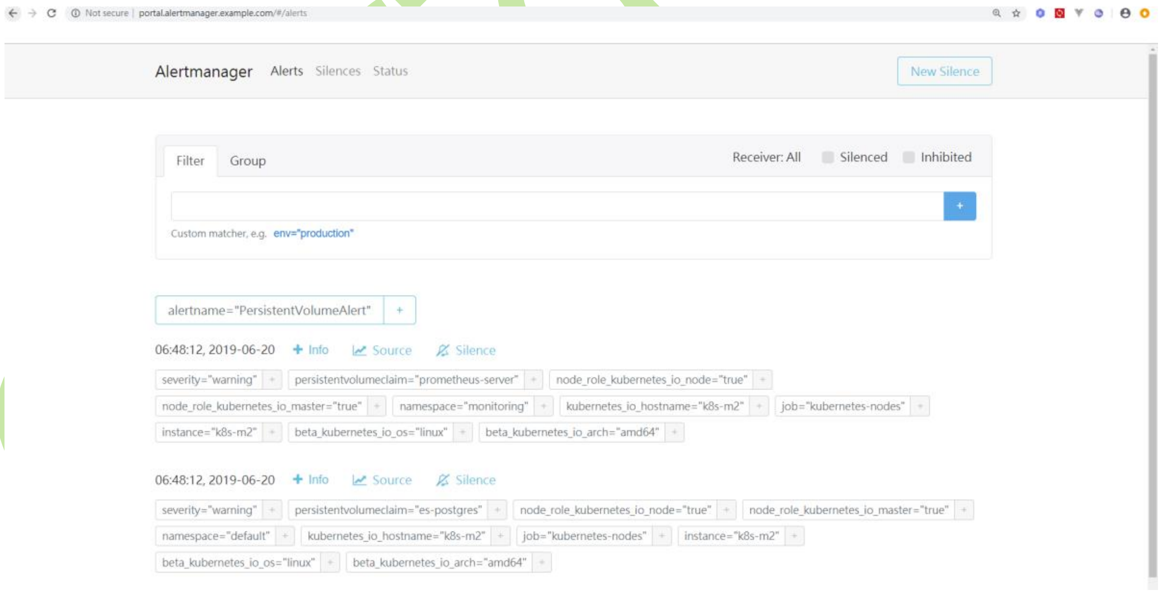
IP Access — http://<DataServer-IP>:3000


portal.grafana.example.com/login
Grafana
email or username
password
Log In
Forgot your password?
Docs | Support Plans | Community | Grafana v5.4.3 (commit: 6539180)

7.3.7 AlertManager

Endpoint — http://portal.alertmanager.example.com

IP Access — http://<DataServer-IP>:9093



Custom matcher, e.g. env="production"

alertname="PersistentVolumeAlert"

06:48:12, 2019-06-20 Info Source Silence

severity="warning" persistentvolumeclaim="prometheus-server" node_role_kubernetes_io_node="true" node_role_kubernetes_io_master="true" namespace="monitoring" kubernetes_io_hostname="k8s-m2" job="kubernetes-nodes" instance="k8s-m2" beta_kubernetes_io_os="linux" beta_kubernetes_io_arch="amd64"

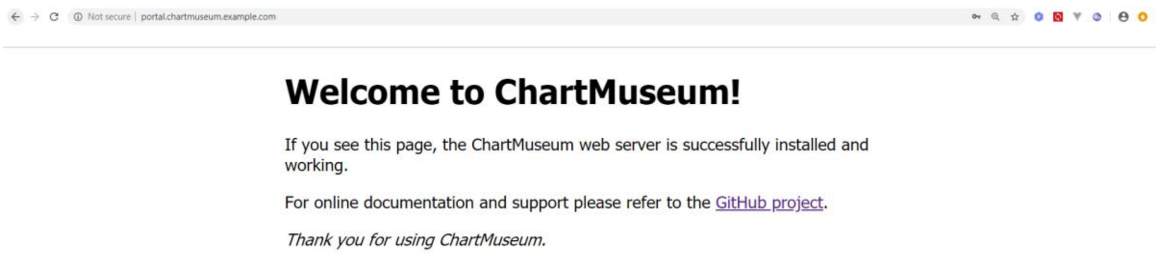
06:48:12, 2019-06-20 Info Source Silence

severity="warning" persistentvolumeclaim="es-postgres" node_role_kubernetes_io_node="true" node_role_kubernetes_io_master="true" namespace="default" kubernetes_io_hostname="k8s-m2" job="kubernetes-nodes" instance="k8s-m2" beta_kubernetes_io_os="linux" beta_kubernetes_io_arch="amd64"

7.3.8 ChartMuseum

Endpoint — http://portal.chartmuseum.example.com

IP Access — http://<DataServer-IP>:8081


Not secure | portal.chartmuseum.example.com

Welcome to ChartMuseum!

If you see this page, the ChartMuseum web server is successfully installed and working.

For online documentation and support please refer to the GitHub project.

Thank you for using ChartMuseum.

8 Kubernetes Dashboard

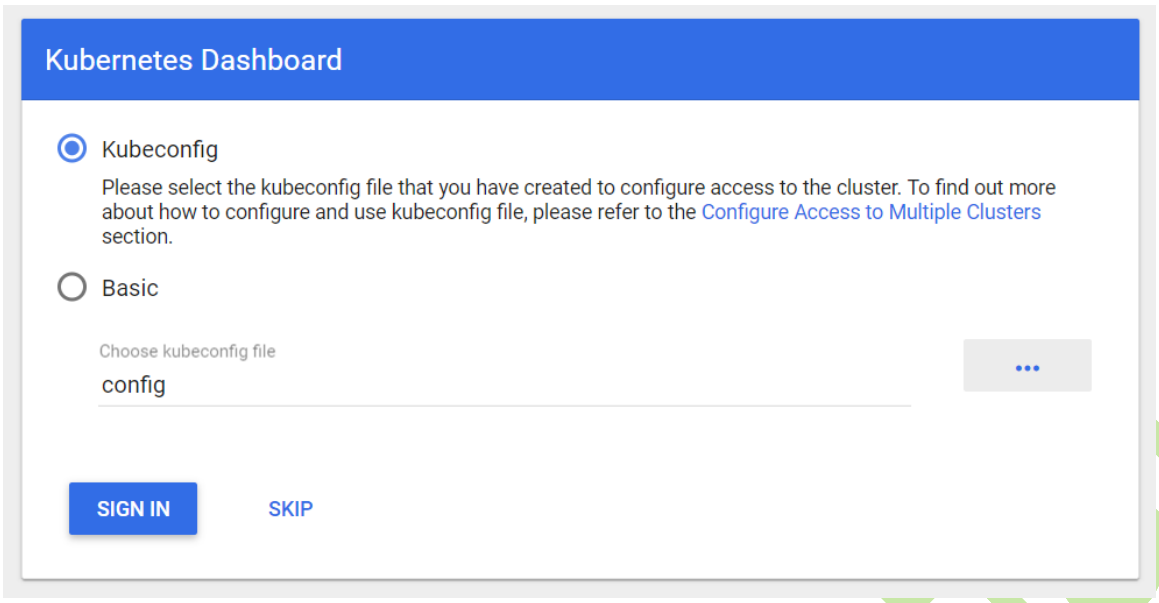
Kubernetes Dashboard is an overview for management of the whole cluster. You could view all the resources on the Data Serivce Server.

8.1 Login

To access the dashboard, use the URL as below:

https://<DataServer-IP>:6443/api/v1/namespaces/kube-system/services/https:kubernetes-dashboard:/proxy/#!/login

Then you could use Kubeconfig to choose the config file in `$HOME/.kube` to login.



Kubernetes Dashboard

Kubeconfig [✓]

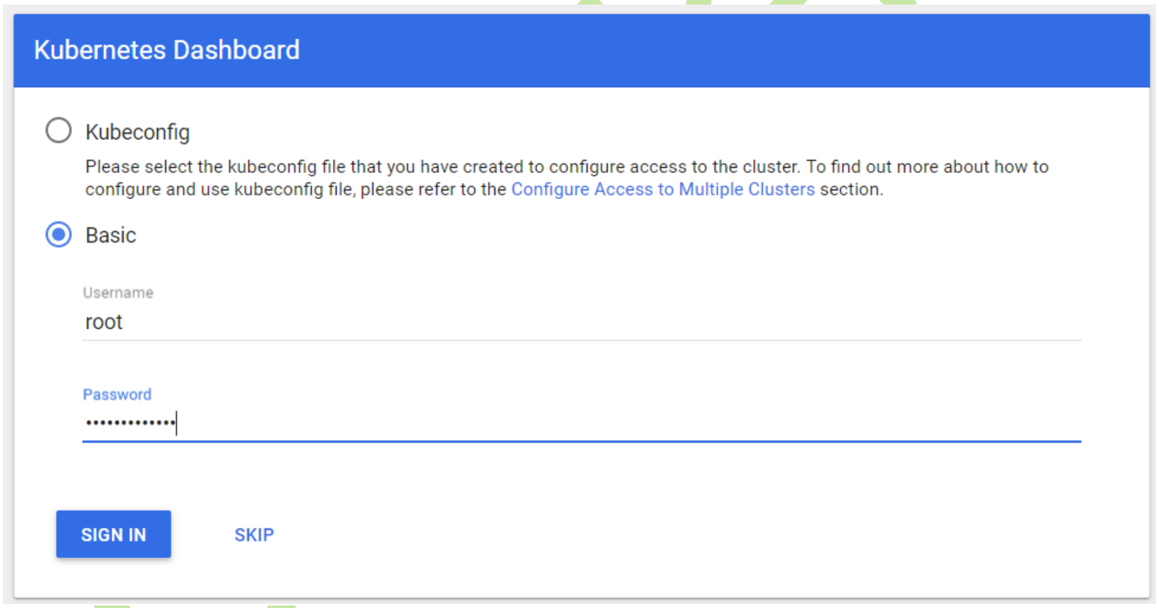
Please select the kubeconfig file that you have created to configure access to the cluster. To find out more about how to configure and use kubeconfig file, please refer to the Configure Access to Multiple Clusters section.

Basic

Choose kubeconfig file: config

Or you could choose Basic method and type in username/password to login:

root/r00tadvant1cH



Kubernetes Dashboard

Kubeconfig

Please select the kubeconfig file that you have created to configure access to the cluster. To find out more about how to configure and use kubeconfig file, please refer to the Configure Access to Multiple Clusters section.

Basic [✓]

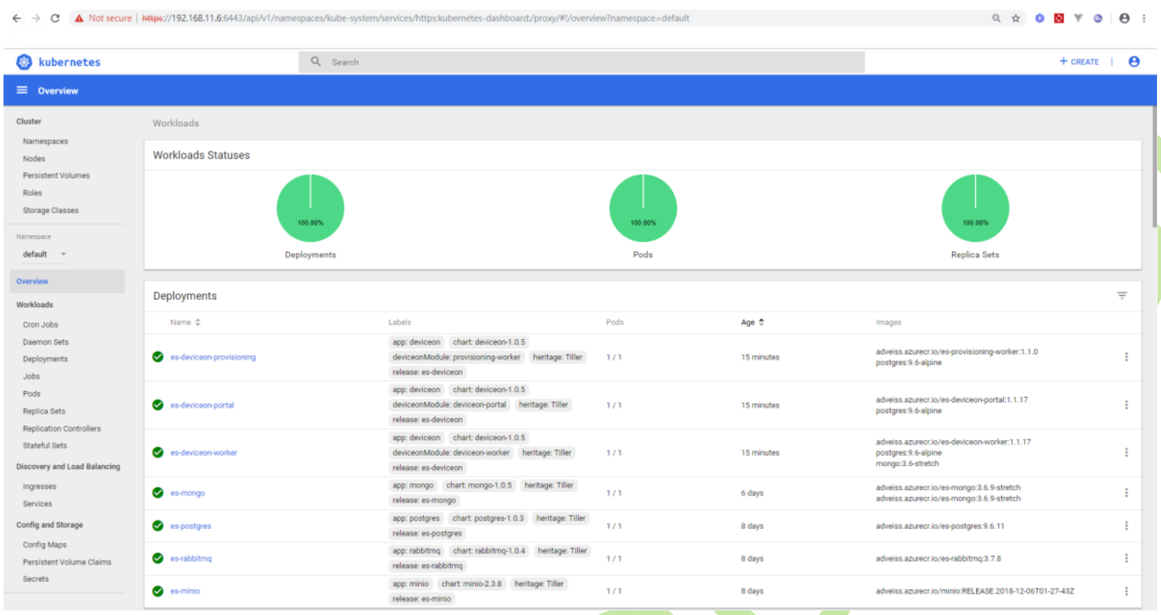
Username: root

Password: •••••••••••••

8.2 Viewing

After you login the dashboard, you could explore all the resources on the Data Service

Server.



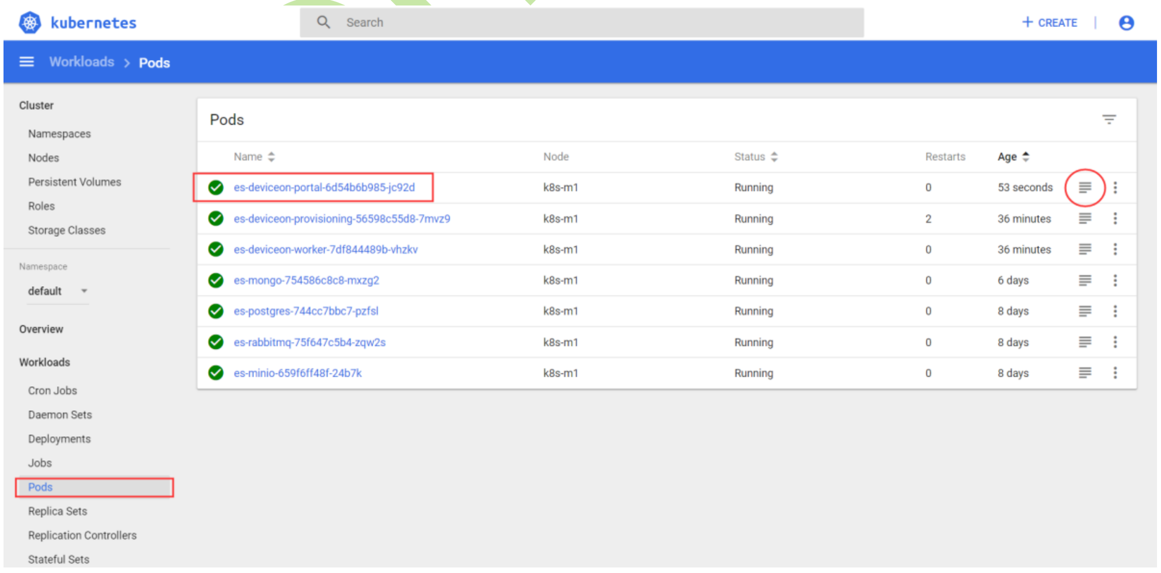
Deployments

Name	Labels	Pods	Age	Images
✓ es-deviceon-provisioning	app: deviceon, chart: deviceon-1.0.5, deviceonModule: provisioning-worker, heritage: Tiller, release: es-deviceon	1 / 1	15 minutes	adveiss.azurecr.io/es-provisioning-worker:1.1.0 postgres:9.6-alpine
✓ es-deviceon-portal	app: deviceon, chart: deviceon-1.0.5, deviceonModule: deviceon-portal, heritage: Tiller, release: es-deviceon	1 / 1	15 minutes	adveiss.azurecr.io/es-deviceon-portal:1.1.17 postgres:9.6-alpine
✓ es-deviceon-worker	app: deviceon, chart: deviceon-1.0.5, deviceonModule: deviceon-worker, heritage: Tiller, release: es-deviceon	1 / 1	15 minutes	adveiss.azurecr.io/es-deviceon-worker:1.1.17 postgres:9.6-alpine mongo:3.6-stretch
✓ es-mongo	app: mongo, chart: mongo-1.0.5, heritage: Tiller, release: es-mongo	1 / 1	6 days	adveiss.azurecr.io/es-mongo:3.6.9-stretch adveiss.azurecr.io/es-mongo:3.6.9-stretch
✓ es-postgres	app: postgres, chart: postgres-1.0.3, heritage: Tiller, release: es-postgres	1 / 1	8 days	adveiss.azurecr.io/es-postgres:9.6.11
✓ es-rabbitmq	app: rabbitmq, chart: rabbitmq-1.0.4, heritage: Tiller, release: es-rabbitmq	1 / 1	8 days	adveiss.azurecr.io/es-rabbitmq:3.7.8
✓ es-minio	app: minio, chart: minio-2.3.8, heritage: Tiller, release: es-minio	1 / 1	8 days	adveiss.azurecr.io/minio:RELEASE.2018-12-06T01-27-43Z

8.3 Logging

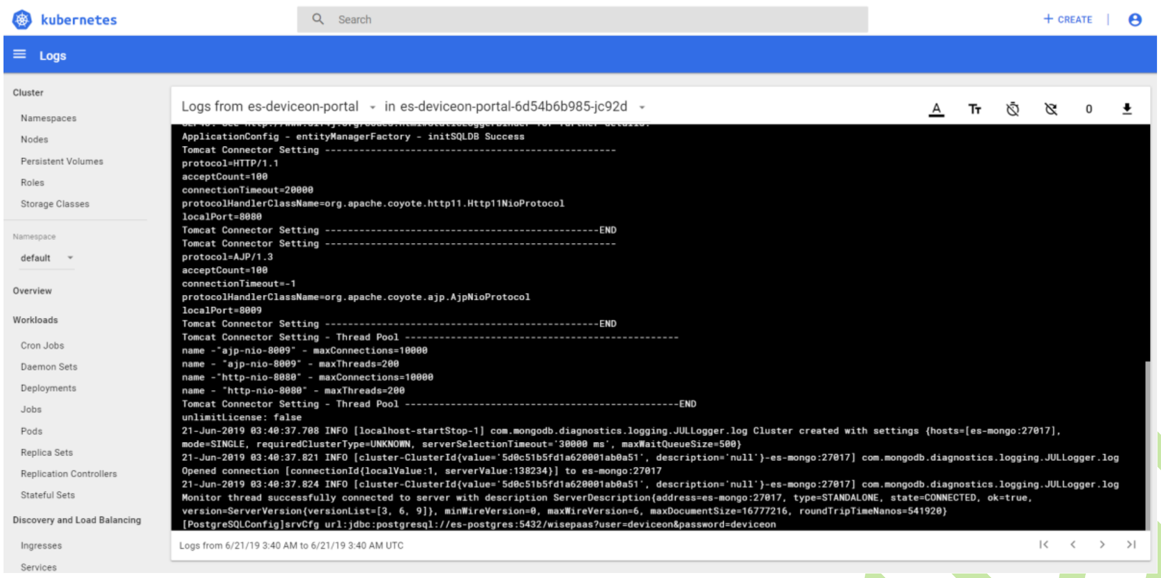
You could go to the Pods page and select the pod you want to get the logs.

The following takes deviceon portal as the example.



Pods

Name	Node	Status	Restarts	Age
✓ es-deviceon-portal-6d54b6b985-jc92d	k8s-m1	Running	0	53 seconds
✓ es-deviceon-provisioning-56598c55d8-7mvz9	k8s-m1	Running	2	36 minutes
✓ es-deviceon-worker-7df844489b-vhzkv	k8s-m1	Running	0	36 minutes
✓ es-mongo-754586c8c8-mxzg2	k8s-m1	Running	0	6 days
✓ es-postgres-744cc7bbc7-pzfsl	k8s-m1	Running	0	8 days
✓ es-rabbitmq-75f647c5b4-zqw2s	k8s-m1	Running	0	8 days
✓ es-minio-659f6ff48f-24b7k	k8s-m1	Running	0	8 days



Logs from es-deviceon-portal in es-deviceon-portal-6d54b6b985-jc92d

```
ApplicationConfig - entityManagerFactory - initSQLDB Success
Tomcat Connector Setting ---------------------------------------------------
protocol=HTTP/1.1
acceptCount=100
connectionTimeout=20000
protocolHandlerClassName=org.apache.coyote.http11.Http11NioProtocol
localPort=8080
Tomcat Connector Setting ------------------------------------------------END
Tomcat Connector Setting ---------------------------------------------------
protocol=AJP/1.3
acceptCount=100
connectionTimeout=-1
protocolHandlerClassName=org.apache.coyote.ajp.AjpNioProtocol
localPort=8009
Tomcat Connector Setting ------------------------------------------------END
Tomcat Connector Setting - Thread Pool ------------------------------------------------
name -"ajp-nio-8009" - maxConnections=10000
name - "ajp-nio-8009" - maxThreads=200
name -"http-nio-8080" - maxConnections=10000
name - "http-nio-8080" - maxThreads=200
Tomcat Connector Setting - Thread Pool ---------------------------------------------END
unlimitLicense: false
21-Jun-2019 03:40:37.708 INFO [localhost-startStop-1] com.mongodb.diagnostics.logging.JULLogger.log Cluster created with settings {hosts=[es-mongo:27017],
mode=SINGLE, requiredClusterType=UNKNOWN, serverSelectionTimeout='30000 ms', maxWaitQueueSize=500}
21-Jun-2019 03:40:37.821 INFO [cluster-ClusterId{value='5d0c51b5fd1a620001ab0a51', description='null'}-es-mongo:27017] com.mongodb.diagnostics.logging.JULLogger.log
Opened connection [connectionId{localValue:1, serverValue:138234}] to es-mongo:27017
21-Jun-2019 03:40:37.824 INFO [cluster-ClusterId{value='5d0c51b5fd1a620001ab0a51', description='null'}-es-mongo:27017] com.mongodb.diagnostics.logging.JULLogger.log
Monitor thread successfully connected to server with description ServerDescription{address=es-mongo:27017, type=STANDALONE, state=CONNECTED, ok=true,
version=ServerVersion{versionList=[3, 6, 9]}, minWireVersion=0, maxWireVersion=6, maxDocumentSize=16777216, roundTripTimeNanos=541920}
[PostgreSQLConfig]srvCfg url:jdbc:postgresql://es-postgres:5432/wisepaas?user=deviceon&password=deviceon
```

Logs from 6/21/19 3:40 AM to 6/21/19 3:40 AM UTC

9 Charts

Charts are packages of pre-configured Kubernetes resources. [12] Data Service Server takes all applications as charts by using helm, which is a package manager for kubernetes. The chapter would show you how to view the charts on the Data Service Server. Before diving deeper into it, you should make sure the authentication parts in section 6.1.1/6.1.2/6.1.3 are all complete because you need to be authenticated to do your management.

9.1 Installing Helm

Helm was installed in Data Service Server by default. If you want manage the resources from your own PC then you should continue or you could skip this section. Helm is a tool for managing Kubernetes charts. Charts are packages of pre-configured Kubernetes resources. [12] Helm uses a packaging format called charts. A chart is a collection of files that describe a related set of Kubernetes resources. A single chart might be used to deploy something simple, like a memcached pod, or something complex, like a full web app stack with HTTP servers, databases, caches, and so on. [13] Data Service Server takes all applications as charts by using helm. Please follow the instructions to install helm. [14]

Use Helm to: [12]

- Find and use popular software packaged as Helm charts to run in Kubernetes
- Share your own applications as Helm charts
- Create reproducible builds of your Kubernetes applications
- Intelligently manage your Kubernetes manifest files
- Manage releases of Helm packages

9.2 Viewing Charts

By using the command below to check the installed charts:

```
# list all of the releases
$ helm list
```

10 Disaster Recovery

Running services for a long period of time, unfortunately, you might encounter some disasters. We all like to live in a perfect world. However, things might not be on the right way as we expect they should be so we should be always prepared. Data Service Server has some ways for you to restore your applications when disasters come up.

10.1 Restoration

10.1.1 Restoring Applications

If some applications are not running properly, you could restore them from a directory (/etc/dss/charts) or from chartmuseum, the private chart repository. Before you start to restore an abnormal application, please use the uninstall scripts in each directories. Let’ s take deviceon as the abnormal example for our restoration.

If the applications belonging to deviceon are not working properly, which might be worker, portal or provisioning, you could use the commands below to restore them.

```
# Delete a chart of DeviceOn
$ cd /etc/dss/charts/DeviceOn
$ ./uninstall.sh

# Install a chart of DeviceOn from the directory
$ cd /etc/dss/charts/DeviceOn
$ ./install.sh

# Install a chart of DeviceOn from private chart repo
$ cd /etc/dss/charts/DeviceOn
$ ./installFromRepo.sh
```

11 Precautions

11.1 Shutdown/Reboot

Since suddenly shutdown might cause Prometheus server or ceph crashes, user needs to execute shutdown utility before shutdown and reboot. Although the crash rarely happens, executing shutdown utility is to play it safe.

1) Execute shutdown script by the following commands.

```
# Execute the shutdown script to make sure promethus & ceph are closed normally
$ cd /root/scripts/shutdown-bootup
$ sh -x k8s-shutdown.sh
```

WARNING: Please execute the script before you shutdown or reboot the Data Service Server. Otherwise, take the responsibility of any possible outcome.

12 Knowledge

12.1 Basic Knowledge of Resources of K8s

Before moving on, we need to have some basic knowledge of the resources of k8s. Here' re some main resources you are going to see.

12.1.1 Pods

A Pod is the basic building block of Kubernetes–the smallest and simplest unit in the Kubernetes object model that you create or deploy. A Pod represents a running process on your cluster. [15]

A Pod encapsulates an application container (or, in some cases, multiple containers), storage resources, a unique network IP, and options that govern how the container(s) should run. A Pod represents a unit of deployment: a single instance of an application in Kubernetes, which might consist of either a single container or a small number of containers that are tightly coupled and that share resources. [15]

To specify a memory limit, include resources:limits

In this exercise, you create a Pod that has one Container. The Container has a memory request of 100 MiB and a memory limit of 200 MiB. Here' s the configuration file for the Pod: [15]

```
apiVersion: v1
kind: Pod
metadata:
  name: memory-demo
  namespace: mem-example
spec:
  containers:
  - name: memory-demo-ctr
    image: polinux/stress
    resources:
      limits:
        memory: "200Mi"
      requests:
        memory: "100Mi"
    command: ["stress"]
    args: ["--vm", "1", "--vm-bytes", "150M",
"--vm-hang", "1"]
```

12.1.2 Services

A Kubernetes Service is an abstraction which defines a logical set of Pods and a policy by which to access them - sometimes called a micro-service. [16] By using Kubernetes Service, you could define the traffic routes to the backend services on your pods.

For example, suppose you have a set of Pods that each expose port 9376 and carry a label "app=MyApp".

```
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  selector:
    app: MyApp
  ports:
  - protocol: TCP
    port: 80
    targetPort: 9376
```

12.1.3 ReplicaSets

A ReplicaSet’ s purpose is to maintain a stable set of replica Pods running at any given time. As such, it is often used to guarantee the availability of a specified number of identical Pods. It ensures that a specified number of pod replicas are running at any given time. However, a Deployment is a higher-level concept that manages ReplicaSets and provides declarative updates to Pods along with a lot of other useful features. [17]

12.1.4 Deployments

A Deployment controller provides declarative updates for Pods and ReplicaSets. [18]

The following is an example of a Deployment. It creates a ReplicaSet to bring up three nginx Pods: [18]

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.7.9
        ports:
        - containerPort: 80
```

12.1.5 Secrets

Kubernetes secret objects let you store and manage sensitive information, such as passwords, OAuth tokens, and ssh keys. Putting this information in a secret is safer and more flexible than putting it verbatim in a Pod definition or in a container image. [19]

A Secret is an object that contains a small amount of sensitive data such as a password, a token, or a key. Such information might otherwise be put in a Pod specification or in an image; putting it in a Secret object allows for more control over how it is used, and reduces the risk of accidental exposure. [19]

For example, to store two strings in a Secret using the data field, convert them to base64 as follows: [19]

```
echo -n 'admin' | base64
YWRtaW4=
echo -n '1f2d1e2e67df' | base64
MWYyZDFlMmU2N2Rm
```

```
apiVersion: v1
kind: Secret
metadata:
  name: mysecret
type: Opaque
data:
  username: YWRtaW4=
  password: MWYyZDFlMmU2N2Rm
```

12.1.6 ConfigMaps

ConfigMaps allow you to decouple configuration artifacts from image content to keep containerized applications portable. [20]

The ConfigMap API resource holds key-value pairs of configuration data that can be consumed in pods or used to store configuration data for system components such as controllers. ConfigMap is similar to Secrets, but designed to more conveniently support working with strings that do not contain sensitive information. [21]

```
kind: ConfigMap
apiVersion: v1
metadata:
  creationTimestamp: 2016-02-8T19:14:8Z
  name: example-config
  namespace: default
data:
  example.property.1: hello
  example.property.2: world
  example.property.file: |-
    property.1=value-1
    property.2=value-2
    property.3=value-3
```

12.2 Commands of kubectl

This section would introduce some basic functions of kubectl. There’ re more commands you could find out. [5]

```
$ kubectl [command] [TYPE] [NAME] [flags]
```

where `command`, `TYPE`, `NAME`, and `flags` are:

`command` — Specifies the operation that you want to perform on one or more resources, for example `create`, `get`, `describe`, `delete`.

`TYPE` — Specifies the resource type. Resource types are case-insensitive and you can specify the singular, plural, or abbreviated forms. For example, the following commands produce the same output:

`NAME` — Specifies the name of the resource. Names are case-sensitive. If the name is omitted, details for all resources are displayed, for example `$ kubectl get pods`.

However, the authorized actions for the viewer in the config only include the following.

```
  $ kubectl get pod pod1
  $ kubectl get pods pod1
  $ kubectl get po pod1
```

12.2.1 GET

`get` — List one or more resources. For more info, visit the website. [5]

```
# List all resources.
$ kubectl get all -n <namespace>

# List all pods in plain-text output format.
$ kubectl get pods

# List all pods in plain-text output format and include additional information
(such as node name).
$ kubectl get pods -o wide

# Get details of a resource and output it as a yaml
$ kubectl get <resource> <name> -o yaml
$ kubectl get pod <name> -o yaml
```

If no specified namespace, the namespace default is in.

12.2.2 DESCRIBE

`describe` — Display detailed state of one or more resources, including the uninitialized ones by default. For more info, visit the website. [5]

```
# Display the details of the node with name <node-name>.
$ kubectl describe nodes <node-name>

# Display the details of the pod with name <pod-name>.
$ kubectl describe pods/<pod-name>

# Display the details of all the pods that are managed by the replication
controller named <rc-name>.
# Remember: Any pods that are created by the replication controller get
prefixed with the name of the replication controller.
$ kubectl describe pods <rc-name>

# Describe all pods, not including uninitialized ones
$ kubectl describe pods --include-uninitialized=false
```

12.2.3 LOGS

`logs` — Print the logs for a container in a pod. For more info, visit the website. [5]

```
# Return a snapshot of the logs from pod <pod-name>.
$ kubectl logs <pod-name>

# Start streaming the logs from pod <pod-name>. This is similar to the 'tail
-f' Linux command.
$ kubectl logs -f <pod-name>
```

12.2.4 DELETE

`delete` — Delete resources either from a file, stdin, or specifying label selectors, names, resource selectors, or resources. [5]

```
# Delete a pod with name <pod-name>.
$ kubectl delete pod/<pod-name>

# Delete a pod with name <pod-name> in <namespace>.
$ kubectl delete pod/<pod-name> -n <namespace>
```

WARNING: This action should be carefully taken because it's not reversible. Take the responsibility of your own action.

12.2.5 EDIT

`edit` — Edit and update the definition of one or more resources on the server by using the default editor. [5]

```
# Edit a service with name <service-name>.
$ kubectl edit service/<service-name>

# Edit a deployment with name <deployment-name>.
$ kubectl edit deployment.apps/<deployment-name>
```

WARNING: This action should be carefully taken. Take the responsibility of your own action.

12.3 Basic Knowledge of Helm

12.3.1 Purposes

Helm is a tool for managing Kubernetes packages called *charts*. Helm can do the following: [22]

- Create new charts from scratch
- Package charts into chart archive (tgz) files
- Interact with chart repositories where charts are stored
- Install and uninstall charts into an existing Kubernetes cluster
- Manage the release cycle of charts that have been installed with Helm

For Helm, there are three important concepts:

1. The *chart* is a bundle of information necessary to create an instance of a Kubernetes application.
2. The *config* contains configuration information that can be merged into a packaged chart to create a releasable object.
3. A *release* is a running instance of a *chart*, combined with a specific *config*.

12.3.2 Components

Helm has two major components: [22]

The Helm Client is a command-line client for end users. The client is responsible for the following domains:

- Local chart development
- Managing repositories
- Interacting with the Tiller server
  - Sending charts to be installed
  - Asking for information about releases
  - Requesting upgrading or uninstalling of existing releases

The Tiller Server is an in-cluster server that interacts with the Helm client, and interfaces with the Kubernetes API server. The server is responsible for the following:

- Listening for incoming requests from the Helm client
- Combining a chart and configuration to build a release
- Installing charts into Kubernetes, and then tracking the subsequent release

- Upgrading and uninstalling charts by interacting with Kubernetes

In a nutshell, the client is responsible for managing charts, and the server is responsible for managing releases.

12.3.3 Commands

Here' re some the useful commands you might use to control your charts. For more info, you could find them on the website of helm. [23]

12.3.3.1 INIT

This will install Tiller to your running Kubernetes cluster. It will also set up any necessary local configuration.

```
# install Tiller to your running Kubernetes cluster
$ helm init
```

12.3.3.2 INSTALL

This command installs a chart archive. The install argument must be a chart reference, a path to a packaged chart, a path to an unpacked chart directory or a URL.

There are five different ways you can express the chart you want to install:

1. By chart reference: helm install stable/mariadb
2. By path to a packaged chart: helm install ./nginx-1.2.3.tgz
3. By path to an unpacked chart directory: helm install ./nginx
4. By absolute URL: helm install https://example.com/charts/nginx-1.2.3.tgz
5. By chart reference and repo url: helm install –repo https://example.com/charts/ nginx

```
# install a chart archive
$ helm install [CHART] [flags]
```

12.3.3.3 LIST

This command lists all of the releases.

```
# list all of the releases
$ helm list
```

12.3.3.4 DELETE

This command takes a release name, and then deletes the release from Kubernetes. It removes all of the resources associated with the last release of the chart.

```
# delete a chart
$ helm delete [NAME] --purge
```

WARNING: This action should be carefully taken because it' s not reversible. Take the responsibility of your own action.

12.3.3.5 REPO

This command consists of multiple subcommands to interact with chart repositories.

It can be used to add, remove and list chart repositories.

```
# add repo
$ helm repo add [flags] [NAME] [URL]
# list repo
$ helm repo list
# update repo
$ helm repo update
# remove repo
$ helm repo remove [NAME]
```

13 FAQ

13.1 Accessing

13.1.1 Why can’ t I use kubectl to access the resources on Data Service Server?

First, you should make sure the Data Service Server is accessible in your network by using ping or whatever tools. If it’ s accessible, the problem might be the configuration for authentication. (13.2)

13.1.2 Why can’ t I access the services by the endpoints in ingresses?

First, you should make sure the Data Service Server is accessible in your network by using ping or whatever tools. Second, use the commands kubectl get, describe or logs (12.2) to check whether the networking services in namespace network are healthy or not. If the networking services are healthy, go check health condition of the pod of the service you want to access.

13.1.3 How should I set the URLs for dashboard and S3 compatible storage in DeviceOn/EdgeSense?

You could set the IP with ports for both of them. <DataServer-IP>:3000 for dashboard and <DataServer-IP>:9000 for S3 compatible storage. You shouldn’ t set the URLs in ingresses because they are serving for the public not for internal applications.

13.2 Authentication

13.2.1 Why do I keep getting an error message "error: You must be logged in to the server (Unauthorized)" when I use kubectl?

The error message might be from the wrong configuration of username/password in the config. The username/password for the viewer is **viewer/viewer.** You can find the instruction in **Setting the Configuration for Authentication** (6.1.2).

13.2.2 Why do I keep get an error message "Unable to connect to the server: dial tcp [::1]:8080: connectex: No connection could be made because the target machine actively refused it." when I use kubectl?

The error message might be from the wrong path or wrong configuration of the config. You can find the instruction in **Setting the Configuration for Authentication** (6.1.2).

13.3 Unhealthy Pods

13.3.1 Why do I get the statuses of pods which are not *Running* but *Error* or others?

The healthy status for a pod is *Running.* If you get a status other than *Running*, in which case it might be *Error* or whatever, it might means the pod is unhealthy. An unhealthy pod could make your services unavailable or not work normally.

13.3.2 How can I do to those unhealthy pods?

Normally, the kubernetes would restart unhealthy pods to try to make them healthy again. However, if you keep getting unhealthy statuses, please try to make an approach to the supports who have more authorized actions which might come in handy to solve your problems.

13.4 Environment Change

13.4.1 Could I change the IP address of the Data Service Server?

The answer to this is No. The IP setting is one of the core parts for most of the servers and Data Service Server is no exception. Changing the IP address might cause many unexpected problems. If you do need to change the IP address, please contact the technical support.

14 Appendix

14.1 Ports

Here's the list of ports of all applications. You could access these applications by using <DataServer-IP>:<Port>

Applications	Port
EdgeSensePortal/DeviceOnPortal	8087
RabbitMQ	15672 (Portal)
	1883 (MQTT)
	8883 (MQTT_SSL)
Postgres	5432

Mongo	27017
Minio	9000
Prometheus	9090
AlertManager	9093
Grafana	3000
ChartMuseum	8081

14.2 Credentials

Here' re the lists of credentials. You could access the applications by any tools as your want.

Applications	Username	Password
OS (Superuser)	root	r00tadvant1cH
Kubernetes Dashboard	root	r00tadvant1cH
EdgeSensePortal/DeviceOnPortal	root@advantech.com.tw	root
RabbitMQ	admin	advant1cH
Postgres	postgres	advant1cH
Mongo	admin	advant1cH
Minio	admin	advant1cH

Grafana	admin	advant1cH
ChartMuseum	admin	advant1cH

15 References

[1] kubernetes, "kubernetes," [Online]. Available: https://kubernetes.io/.

[2] docker, "docker," [Online]. Available: https://en.wikipedia.org/wiki/Docker_(software).

[3] ceph, "ceph," [Online]. Available: https://en.wikipedia.org/wiki/Ceph_(software).

[4] kubernetes, "namespaces," [Online]. Available: https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/.

[5] kubernetes, "kubectl," [Online]. Available: https://kubernetes.io/docs/reference/kubectl/overview/.

[6] kubernetes, "install kubectl," [Online]. Available: https://kubernetes.io/docs/tasks/tools/install-kubectl/.

[7] wiki, "ssh," [Online]. Available: https://en.wikipedia.org/wiki/Secure_Shell.

[8] nginx, "nginx," [Online]. Available: https://nginx.org/en/.

[9] kubernetes, "nginx-ingress-controller," [Online]. Available: https://github.com/kubernetes/ingress-nginx.

[10] T. Fisher, "How to Change DNS Servers in Windows," [Online]. Available: https://www.lifewire.com/how-to-change-dns-servers-in-windows-2626242.

[11] kubernetes, "ingress," [Online]. Available: https://kubernetes.io/docs/concepts/services-networking/ingress/.

[12] github, "helm," [Online]. Available: https://github.com/helm/helm.

[13] helm, "charts," [Online]. Available: https://helm.sh/docs/developing_charts/#charts.

[14] helm, "helm," [Online]. Available: https://helm.sh/docs/using_helm/#installing-helm.

[15] kubernetes, "pods," [Online]. Available: https://kubernetes.io/docs/concepts/workloads/pods/.

[16] kubernetes, "services," [Online]. Available:

] https://kubernetes.io/docs/concepts/services-networking/service/.

[17] kubernetes, "replicaset," [Online]. Available: https://kubernetes.io/docs/concepts/workloads/controllers/replicaset/.

[18] kubernetes, "deployments," [Online]. Available: https://kubernetes.io/docs/concepts/workloads/controllers/deployment/.

[19] kubernetes, "secrets," [Online]. Available: https://kubernetes.io/docs/concepts/configuration/secret/.

[20] kubernetes, "configmaps," [Online]. Available: https://kubernetes.io/docs/tasks/configure-pod-container/configure-pod-configmap/#before-you-begin.

[21] u. kubernetes, "configmaps," [Online]. Available: https://unofficial-kubernetes.readthedocs.io/en/latest/tasks/configure-pod-container/configmap/.

[22] helm, "architecture," [Online]. Available: https://helm.sh/docs/architecture/.

[23] helm, "commands," [Online]. Available: https://helm.sh/docs/helm/#helm.

[24] helm, "chart_template," [Online]. Available: https://helm.sh/docs/chart_template_guide/#getting-started-with-a-chart-template.

[25] wiki, "docker," [Online]. Available: https://en.wikipedia.org/wiki/Docker_(software).

[26] docker, "curriculum," [Online]. Available: https://docker-curriculum.com/.

[27] chartmuseum, "push," [Online]. Available: https://github.com/chartmuseum/helm-push.