



# Data Service Server User's Manual

V1.0.0

## Revision History

Date	Version	Author	Reviewer	Description
2019-03-19	1.0.0	Zach Chih	Zach Chih	First version released

Confidential

# Table of Contents

<b>1</b>	<b>Preface.....</b>	<b>4</b>
<b>2</b>	<b>Introduction.....</b>	<b>4</b>
<b>3</b>	<b>Architecture.....</b>	<b>6</b>
<b>4</b>	<b>Components.....</b>	<b>7</b>
<b>5</b>	<b>Network Topology.....</b>	<b>8</b>
<b>6</b>	<b>Get Started .....</b>	<b>8</b>
6.1	<b>Basic Knowledge of Resources of K8S .....</b>	<b>9</b>
6.1.1	Pods .....	10
6.1.2	Services.....	10
6.1.3	ReplicaSets.....	10
6.1.4	Deployments.....	11
6.2	<b>Installing kubectl.....</b>	<b>11</b>
6.3	<b>Setting the Configuration for Authentication .....</b>	<b>11</b>
6.4	<b>Commands.....</b>	<b>14</b>
6.4.1	GET .....	15
6.4.2	DESCRIBE .....	15
6.4.3	LOGS.....	16
6.5	<b>Viewing the Nginx Resources .....</b>	<b>16</b>
6.6	<b>Viewing the DNS Resources .....</b>	<b>17</b>
6.7	<b>Viewing the Endpoints (Ingress).....</b>	<b>18</b>
6.8	<b>Setting the DNS .....</b>	<b>18</b>
6.8.1	Windows .....	19
6.8.2	Linux .....	24
6.9	<b>Accessing the Endpoints .....</b>	<b>25</b>

6.9.1	EdgeSense .....	25
6.9.2	Prometheus .....	25
6.9.3	Grafana.....	26
<b>7</b>	<b>FAQ .....</b>	<b>27</b>
<b>7.1</b>	<b>Accessing .....</b>	<b>27</b>
7.1.1	Why can't I use kubectl to access the resources on Data Service Server?.....	27
7.1.2	Why can't I access the services by the endpoints in ingresses?.....	27
7.1.3	How should I set the URLs for dashboard and S3 compatible storage in EdgeSense?.....	27
<b>7.2</b>	<b>Authentication .....</b>	<b>28</b>
7.2.1	Why do I keep getting an error message "error: You must be logged in to the server (Unauthorized)"when I use kubectl? .....	28
7.2.2	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? .....	28
7.2.3	Why can't I do more actions than just viewing on the resources of Data Service Server? .....	29
	The role for you is just a viewer and the action authorized for the viewer is only "viewing." .....	29
<b>7.3</b>	<b>Unhealthy Pods .....</b>	<b>29</b>
7.3.1	Why do I get the statuses of pods which are not <i>Running</i> but <i>Error</i> or others? .....	29
7.3.2	How can I do to those unhealthy pods? .....	29
<b>8</b>	<b>Appendix .....</b>	<b>29</b>
<b>8.1</b>	<b>Ports .....</b>	<b>29</b>
<b>9</b>	<b>References .....</b>	<b>31</b>

# 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

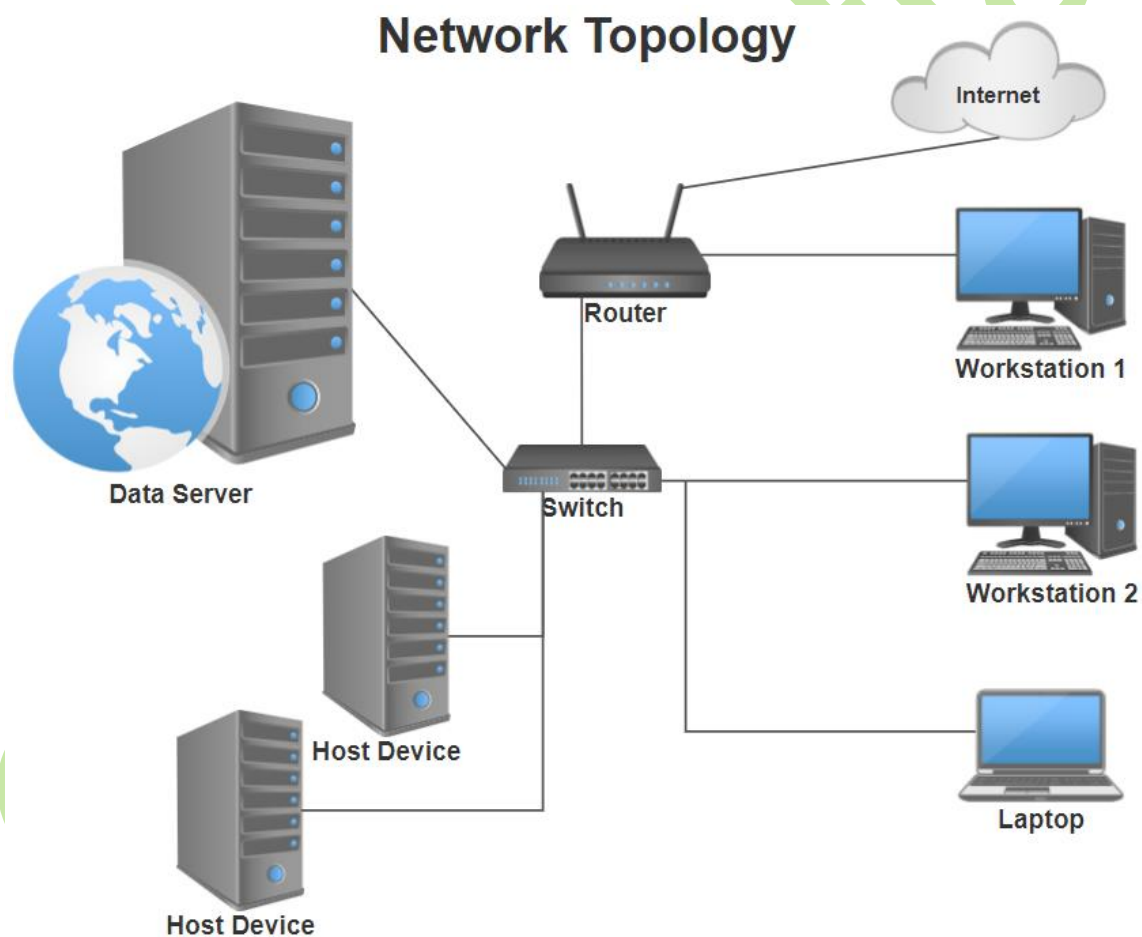
Kubernetes supports multiple virtual clusters backed by the same physical cluster. These virtual clusters are called namespaces. Namespaces are intended for use in environments with many users spread across multiple teams, or projects. [4] In Data Service Server, there're three namespaces — default, network and monitoring. 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
	RabbitMQ
	Postgres
	Mongo
	Minio
network	Nginx
	CoreDNS
monitoring	Prometheus
	Grafana
	Postgres-Exporter
	RabbitMQ-Exporter



## 5 Network Topology

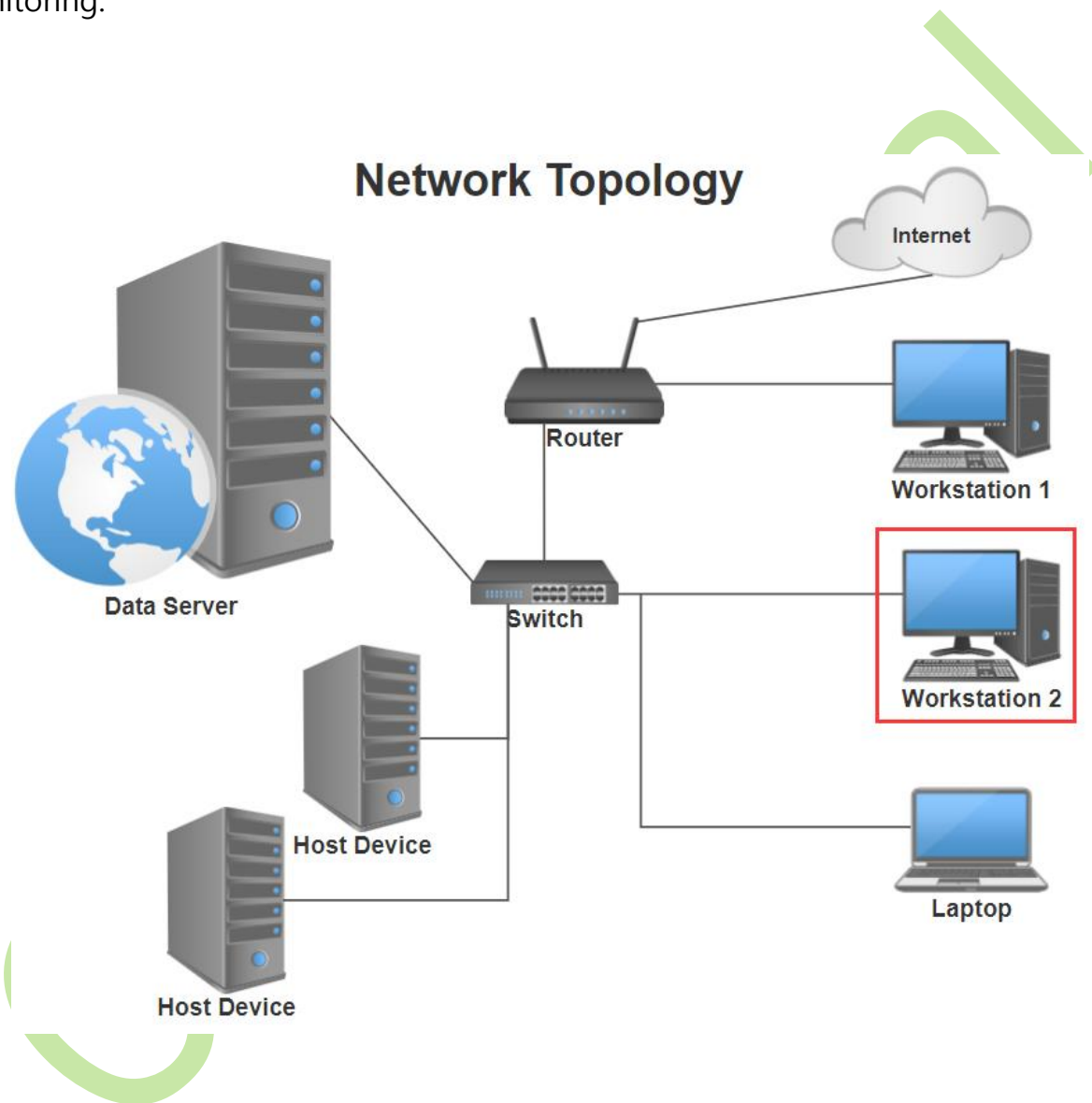
The figure is an example network topology. Your workstations or laptops might be able to connect to Data Service Server through a router or a switch. And some of the host devices might connect to Data Service Server through the switch. The topology might be different from different users' working environments.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



## 6 Get Started

To have a better picture on what we are going to do, let's assume you're using the Workstation 2 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 the configuration of authentication. Finally, you would know how to access the web services for management and monitoring.



## 6.1 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.

### 6.1.1 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. [5]

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. [5]

### 6.1.2 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. [6] By using Kubernetes Service, you could define the traffic routes to the backend services on your pods.

### 6.1.3 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.

#### 6.1.4 Deployments

A Deployment controller provides declarative updates for Pods and ReplicaSets. [7]

### 6.2 Installing kubectl

Kubectl is a command line interface for running commands against Kubernetes clusters.

[8] For details about each command, including all the supported flags and subcommands, see the kubectl reference documentation. For installation instructions see installing kubectl. [9]

### 6.3 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.

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

Here' s the content in config:

```
apiVersion: v1
clusters:
- cluster:
    server: <server-ip>
    name: cluster.local
contexts:
- context:
    cluster: cluster.local
    user: viewer
    name: viewer-ctx
current-context: viewer-ctx
kind: Config
preferences: {}
users:
- name: viewer
  user:
    password: viewer
    username: viewer
```

First, copy the config file and ca.crt into the folder `$HOME/.kube`. Then use the command below to set the certificate for the configuration.

```
$ kubectl config set-cluster kubernetes --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: viewer
  name: viewer-ctx
current-context: viewer-ctx
kind: Config
preferences: {}
users:
- name: viewer
  user:
    password: viewer
    username: viewer
```

Then use the command below to check your configuration:

```
$ kubectl get all
```

```

root@k8s-m1:~/k8e# kubectl get all
NAME                                READY    STATUS    RESTARTS   AGE
pod/es-minio-659f6ff48f-n6jrh      1/1      Running   0           1d
pod/es-mongo-68d557b46f-5f9sw      1/1      Running   0           8d
pod/es-postgres-84f54b9c7f-l46nn    1/1      Running   0           8d
pod/es-rabbitmq-8866746df-k5vkf     1/1      Running   0           7d
pod/es-rmm-ota-d9f48647f-42gg5      1/1      Running   0           1d
pod/es-rmm-portal-d64cd5db8-56t9v   1/1      Running   0           1d
pod/es-rmm-worker-5494cb78b4-kfb62  1/1      Running   0           1d

NAME                                AGE    TYPE    CLUSTER-IP    EXTERNAL-IP    PORT(S)
service/es-minio                    1d     ClusterIP  10.233.56.178  <none>         9000/TCP
service/es-mongo                    8d     ClusterIP  10.233.58.28   <none>         27017/TCP
service/es-postgres                 8d     ClusterIP  10.233.16.240  <none>         5432/TCP
service/es-rabbitmq                 7d     ClusterIP  10.233.28.192  <none>         4369/TCP,5672/TCP,1883/TCP,15672/TCP,5671/TCP,8883/TCP
service/es-rmm-portal               1d     ClusterIP  10.233.33.97   <none>         8080/TCP
service/kubernetes                  33d    ClusterIP  10.233.0.1     <none>         443/TCP
service/rmm-iothub-bridge-service   9d     NodePort   10.233.47.179  <none>         1885:30556/TCP

NAME                                DESIRED  CURRENT  UP-TO-DATE  AVAILABLE  AGE
deployment.apps/es-minio            1         1         1             1           1d
deployment.apps/es-mongo            1         1         1             1           8d
deployment.apps/es-postgres         1         1         1             1           8d
deployment.apps/es-rabbitmq         1         1         1             1           7d
deployment.apps/es-rmm-ota          1         1         1             1           1d
deployment.apps/es-rmm-portal       1         1         1             1           1d
deployment.apps/es-rmm-worker       1         1         1             1           1d

NAME                                DESIRED  CURRENT  READY  AGE
replicaset.apps/es-minio-659f6ff48f 1         1         1       1d
replicaset.apps/es-mongo-68d557b46f 1         1         1       8d
replicaset.apps/es-postgres-84f54b9c7f 1         1         1       8d
replicaset.apps/es-rabbitmq-8866746df 1         1         1       7d
replicaset.apps/es-rmm-ota-d9f48647f 1         1         1       1d
replicaset.apps/es-rmm-portal-d64cd5db8 1         1         1       1d
replicaset.apps/es-rmm-worker-5494cb78b4 1         1         1       1d

```

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

## 6.4 Commands

This section would introduce some basic functions of kubectl. There're more commands you could find out. [8]

```
$ 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
```

#### 6.4.1 GET

**get** — List one or more resources. For more info, visit the website. [8]

```
# 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
```

If no specified namespace, the namespace default is in.

#### 6.4.2 DESCRIBE

**describe** — Display detailed state of one or more resources, including the uninitialized ones by default. For more info, visit the website. [8]

```
# 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
```

### 6.4.3 LOGS

`logs` — Print the logs for a container in a pod. For more info, visit the website. [8]

```
# 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>
```

## 6.5 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. [10] An Ingress Controller is a daemon, deployed as a Kubernetes Pod, that watches the apiserver's /ingresses endpoint for updates to the Ingress resource. [11] 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:

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

```
root@k8s-m1:~# kubectl get all -n network
NAME                                READY   STATUS    RESTARTS   AGE
pod/coredns-coredns-77447c99fd-nfcft 1/1     Running   0           1d
pod/nginx-ingress-controller-56f4b49cd5-djkt7 1/1     Running   0           1d
pod/nginx-ingress-default-backend-5fcccb776-nrl46 1/1     Running   0           1d

NAME                                TYPE          CLUSTER-IP    EXTERNAL-IP  PORT(S)          AGE
service/coredns-coredns             NodePort      10.233.63.196 <none>       53:53/UDP        1d
service/nginx-ingress-controller     NodePort      10.233.17.114 <none>       80:80/TCP,443:443/TCP,15672:15672/TCP,1883:1883/TCP,27017:27017/TCP,3000:3000/TCP,5432:5432/TCP,8080:8080/TCP,8883:8883/TCP,9000:9000/TCP,9090:9090/TCP,9187:9187/TCP,9419:9419/TCP 1d
service/nginx-ingress-default-backend ClusterIP      10.233.41.91  <none>       80/TCP           1d

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

NAME                                DESIRED   CURRENT   READY   AGE
replicaset.apps/coredns-coredns-77447c99fd 1         1         1       1d
replicaset.apps/nginx-ingress-controller-56f4b49cd5 1         1         1       1d
replicaset.apps/nginx-ingress-default-backend-5fcccb776 1         1         1       1d
```

## 6.6 Viewing the DNS Resources

The DNS Server in Data Service Server is CoreDNS. Use the command below to view the resources related to coredns in namespace network:

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

```
root@k8s-m1:~# kubectl get all -n network
NAME                                READY   STATUS    RESTARTS   AGE
pod/coredns-coredns-77447c99fd-nfcft 1/1     Running   0           1d
pod/nginx-ingress-controller-56f4b49cd5-djkt7 1/1     Running   0           1d
pod/nginx-ingress-default-backend-5fcccb776-nrl46 1/1     Running   0           1d

NAME                                TYPE          CLUSTER-IP    EXTERNAL-IP  PORT(S)          AGE
service/coredns-coredns             NodePort      10.233.63.196 <none>       53:53/UDP        1d
service/nginx-ingress-controller     NodePort      10.233.17.114 <none>       80:80/TCP,443:443/TCP,15672:15672/TCP,1883:1883/TCP,27017:27017/TCP,3000:3000/TCP,5432:5432/TCP,8080:8080/TCP,8883:8883/TCP,9000:9000/TCP,9090:9090/TCP,9187:9187/TCP,9419:9419/TCP 1d
service/nginx-ingress-default-backend ClusterIP      10.233.41.91  <none>       80/TCP           1d

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

NAME                                DESIRED   CURRENT   READY   AGE
replicaset.apps/coredns-coredns-77447c99fd 1         1         1       1d
replicaset.apps/nginx-ingress-controller-56f4b49cd5 1         1         1       1d
replicaset.apps/nginx-ingress-default-backend-5fcccb776 1         1         1       1d
```

## 6.7 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. [12]

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-m1:~/kube# kubectl get ingress
NAME          HOSTS                                     ADDRESS          PORTS          AGE
es-ingress    portal.rmm.example.com,portal.rabbitmq.example.com,portal.minio.example.com  80              7d
```

```
$ 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.

## 6.8 Setting the DNS

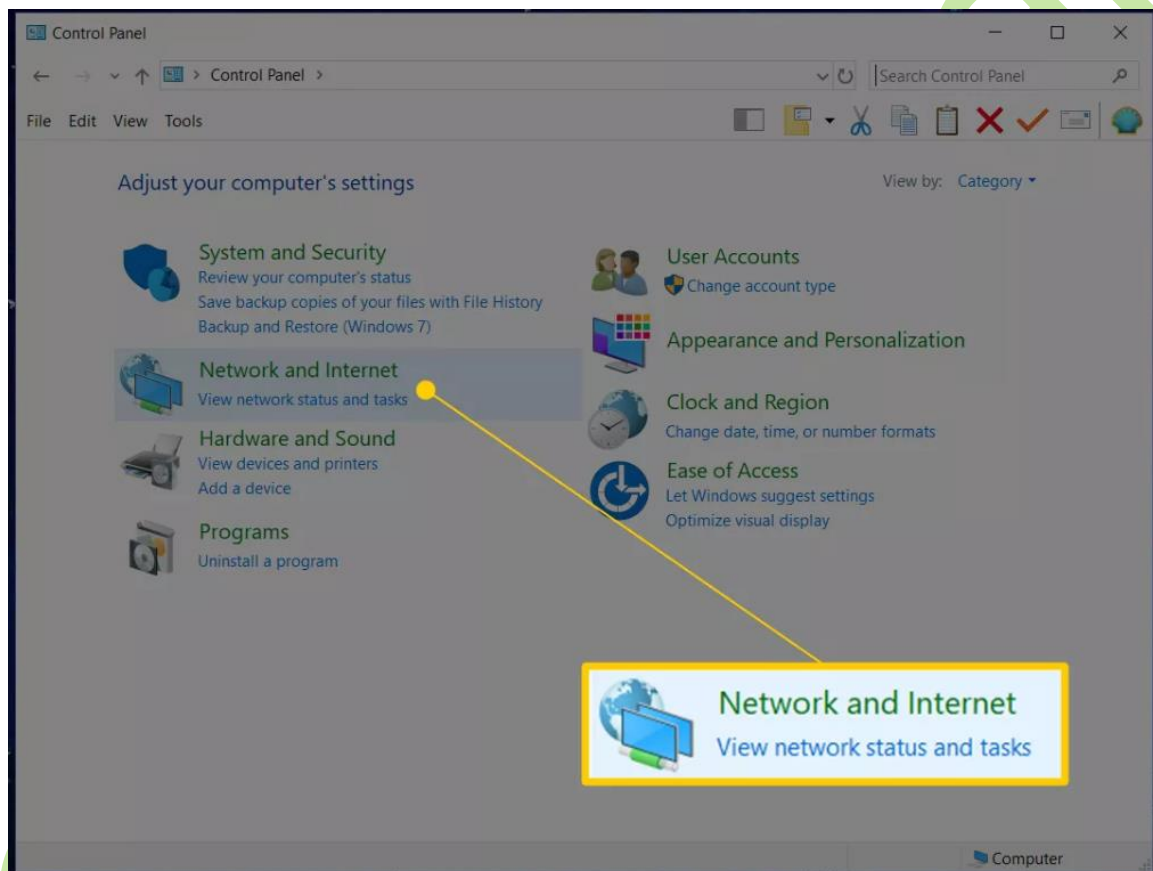
Before you try to connect to these services by these endpoints, you should set your DNS IP to the IP Data Service Server locates at. This is because the URL translation is

being handled by the CoreDNS which the Data Service Server provides. The steps of setting the DNS Server are as follows. [13]

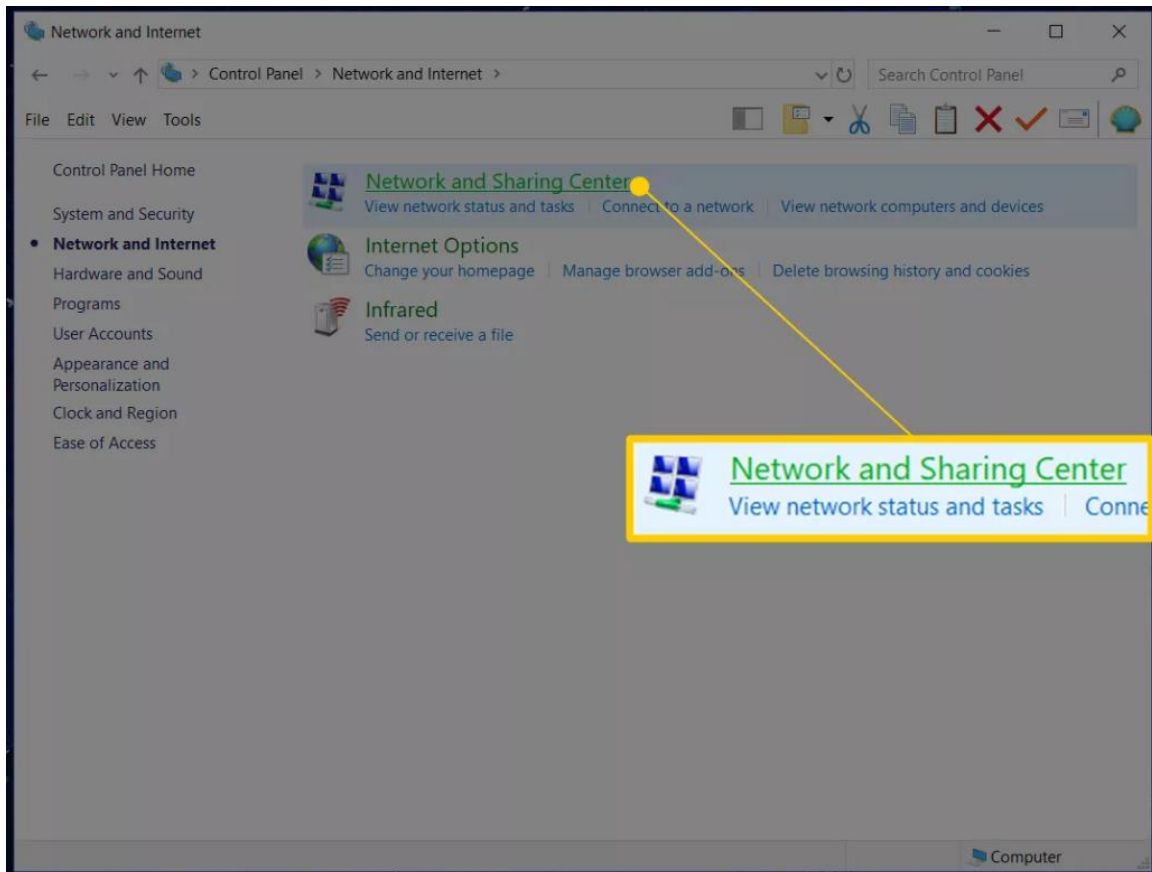
### 6.8.1 Windows

01. Open Control Panel.

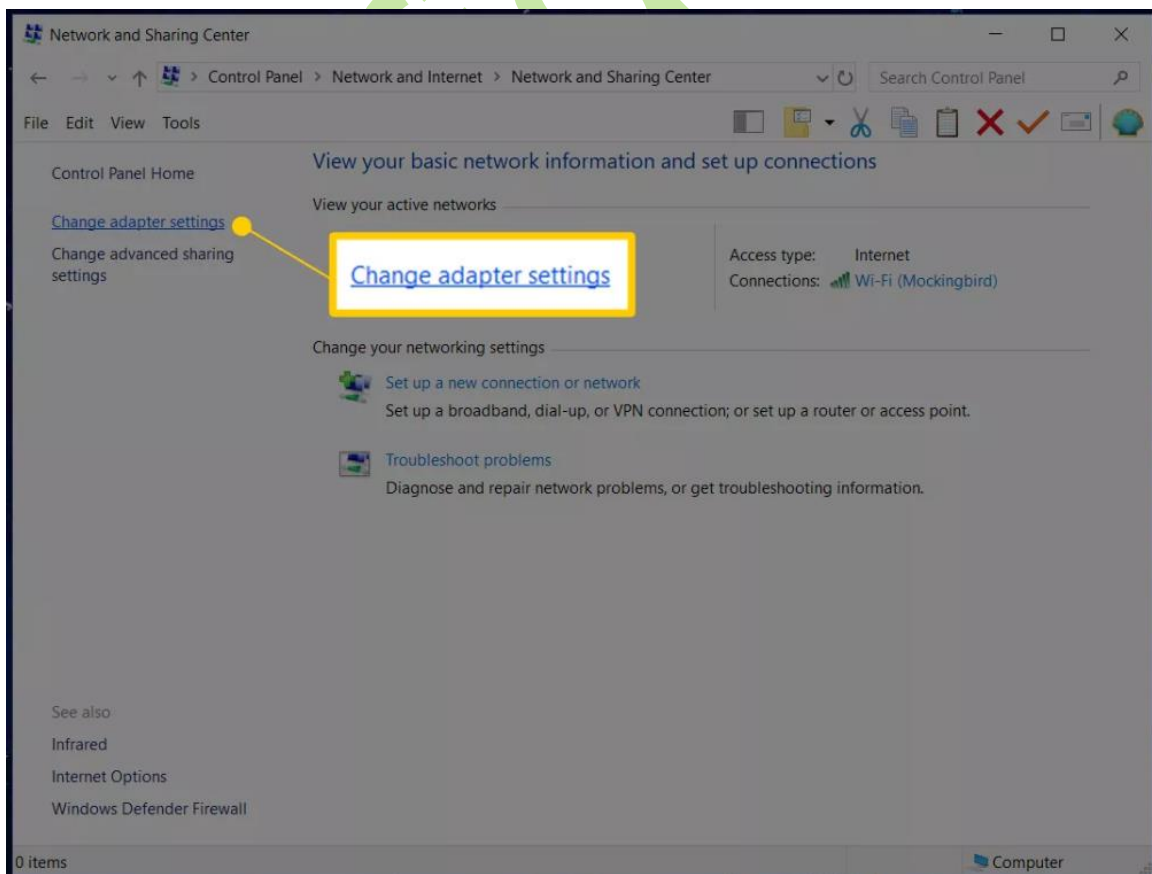
02. Once in Control Panel, touch or click **Network and Internet**.



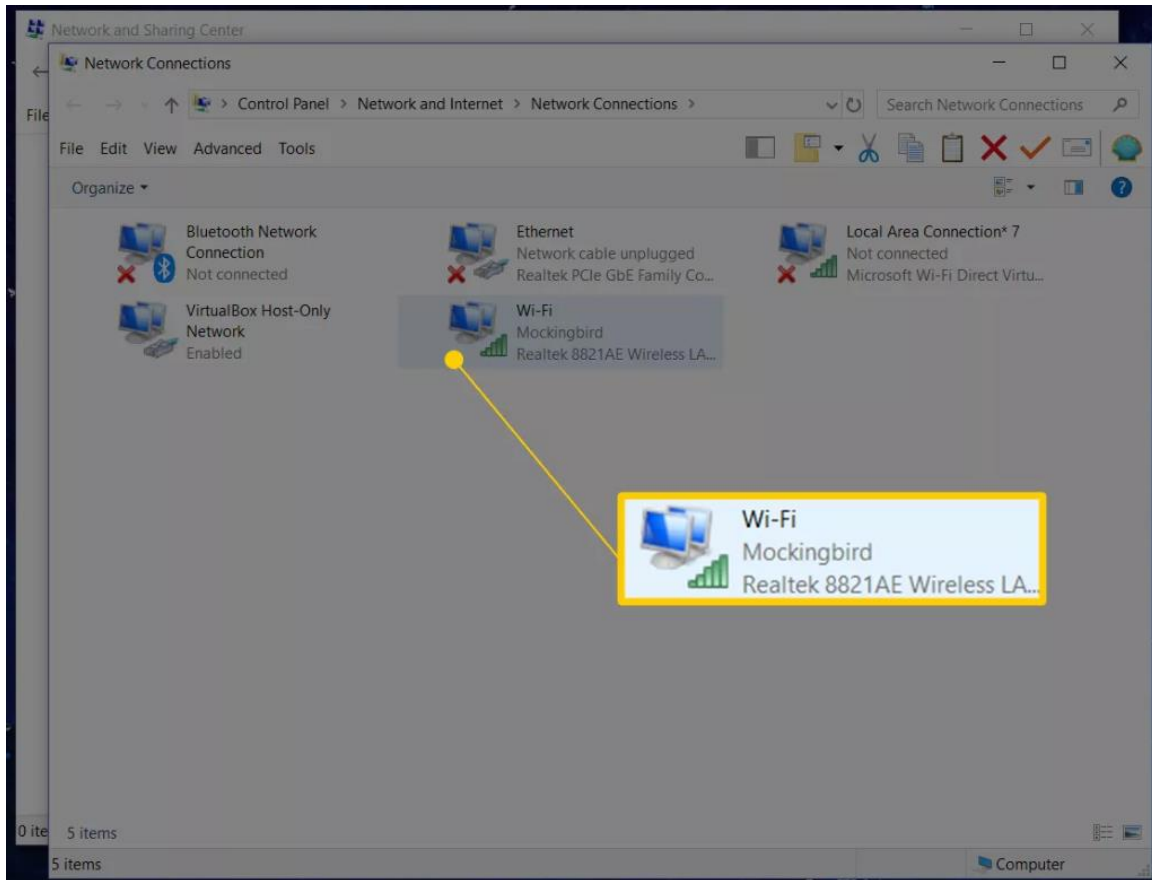
03. In the Network and Internet window that's now open, click or touch **Network and Sharing Center** to open that applet.



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



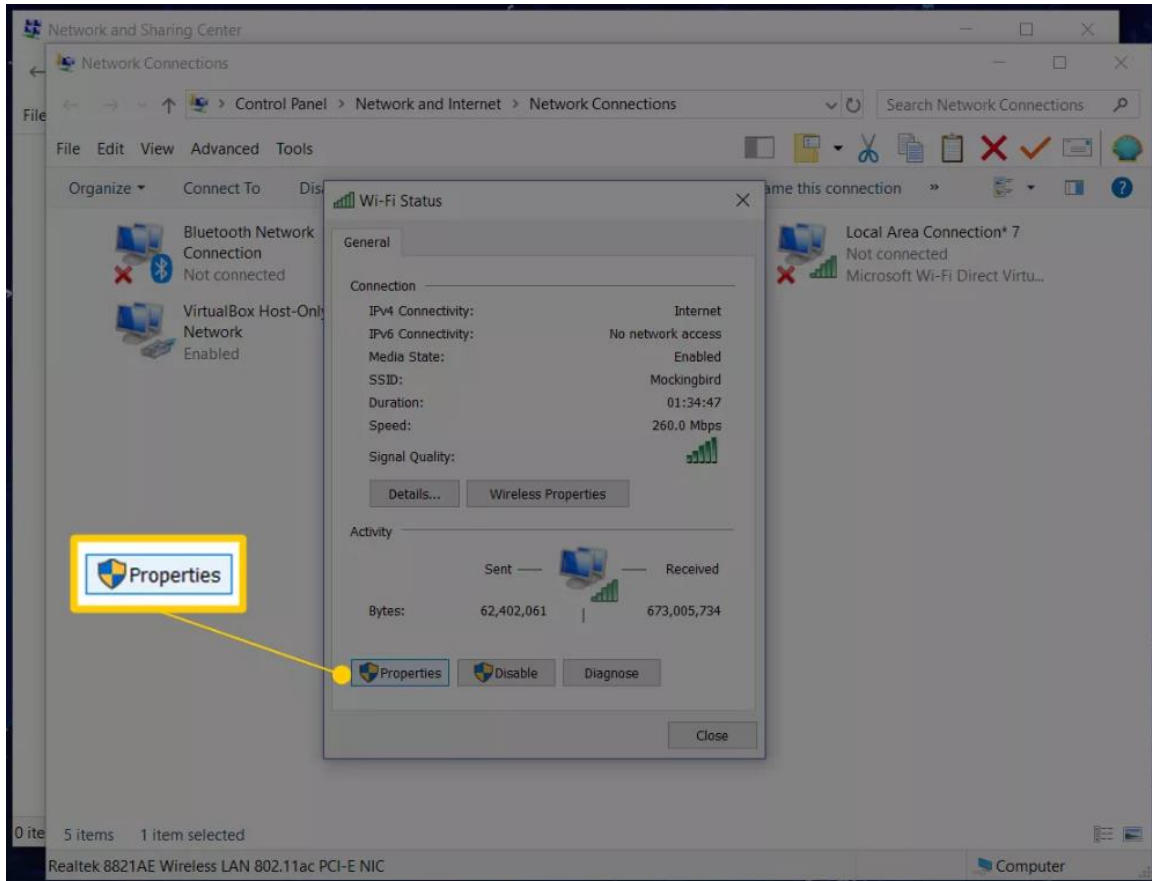
05. From this new Network Connections screen, locate the network connection that you want to change the DNS servers for.



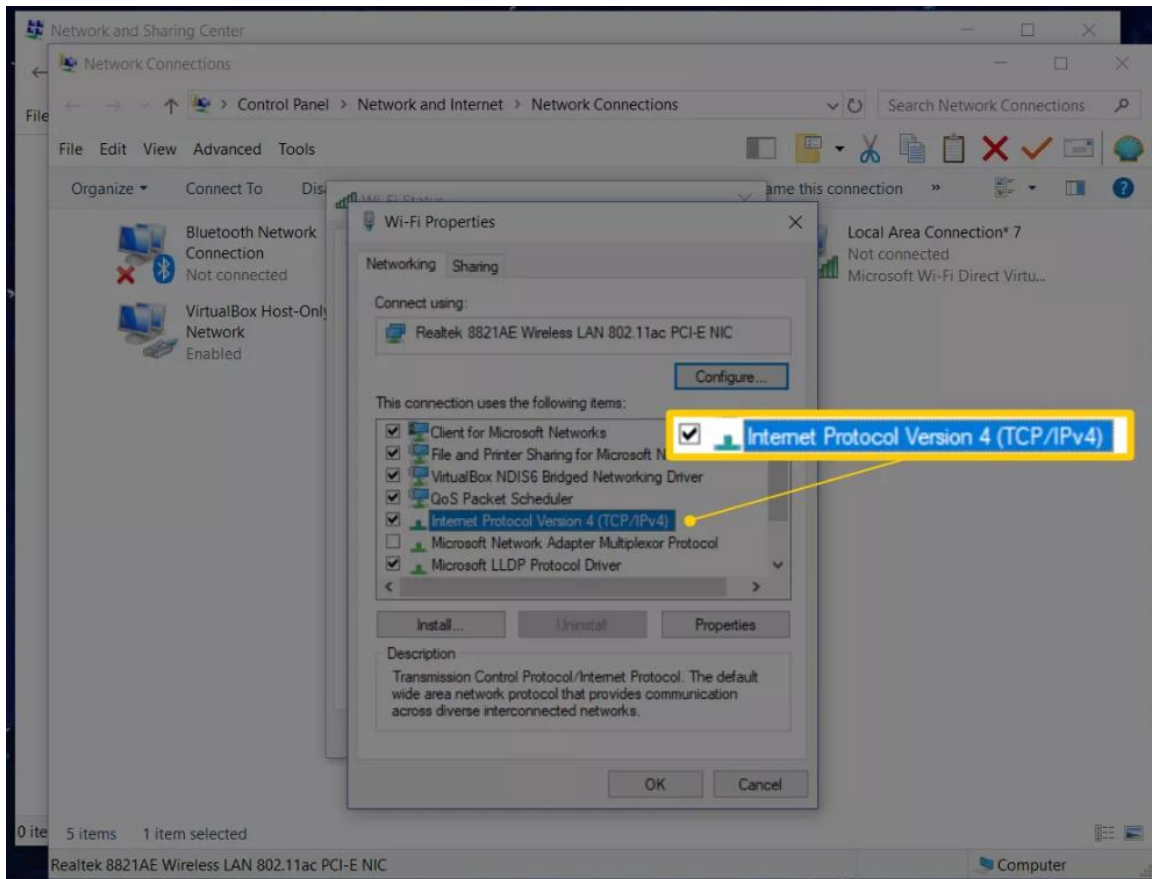
06. Open the network connection you want to change the DNS servers for by double-clicking or double-tapping on its icon.

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





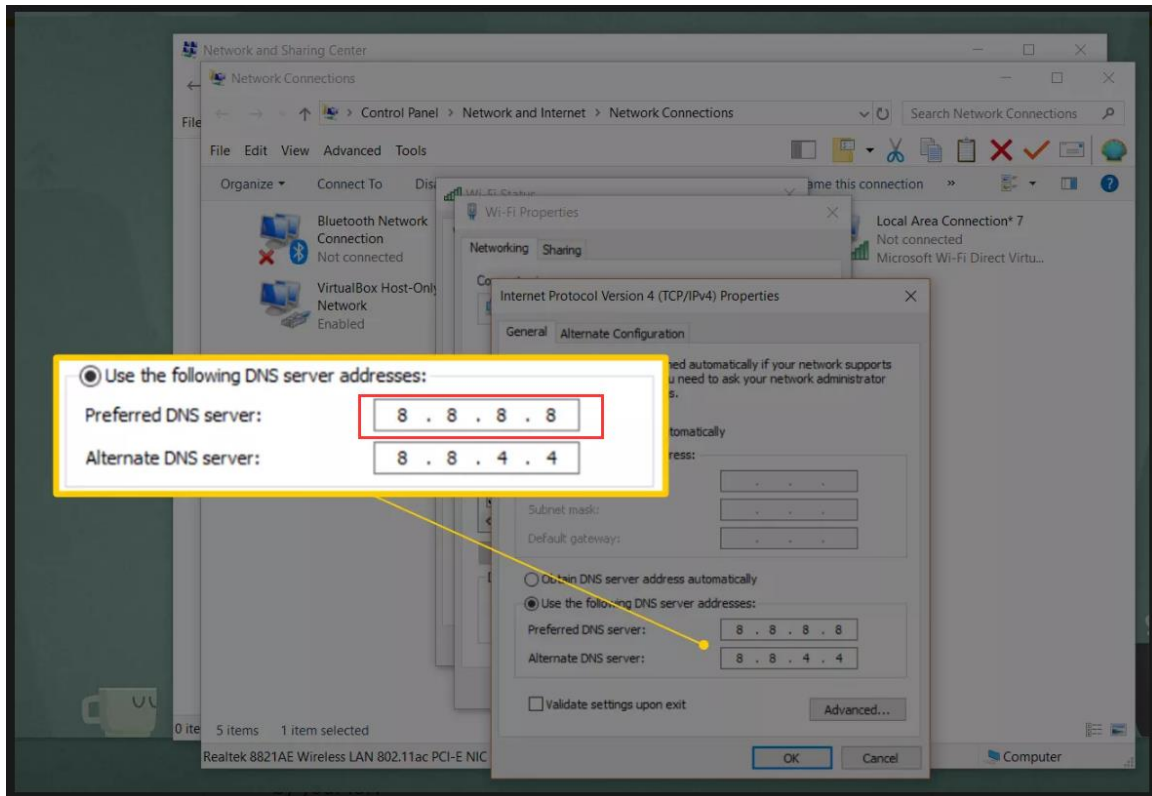
08. On the connection's Properties window that appeared, scroll down in the *This connection uses the following items:* list and click or tap **Internet Protocol Version 4 (TCP/IPv4)** or **Internet Protocol (TCP/IP)** to select the IPv4 option to change the IPv4 DNS server settings.



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.



## 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.

### 6.8.2 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.

## 6.9 Accessing the Endpoints

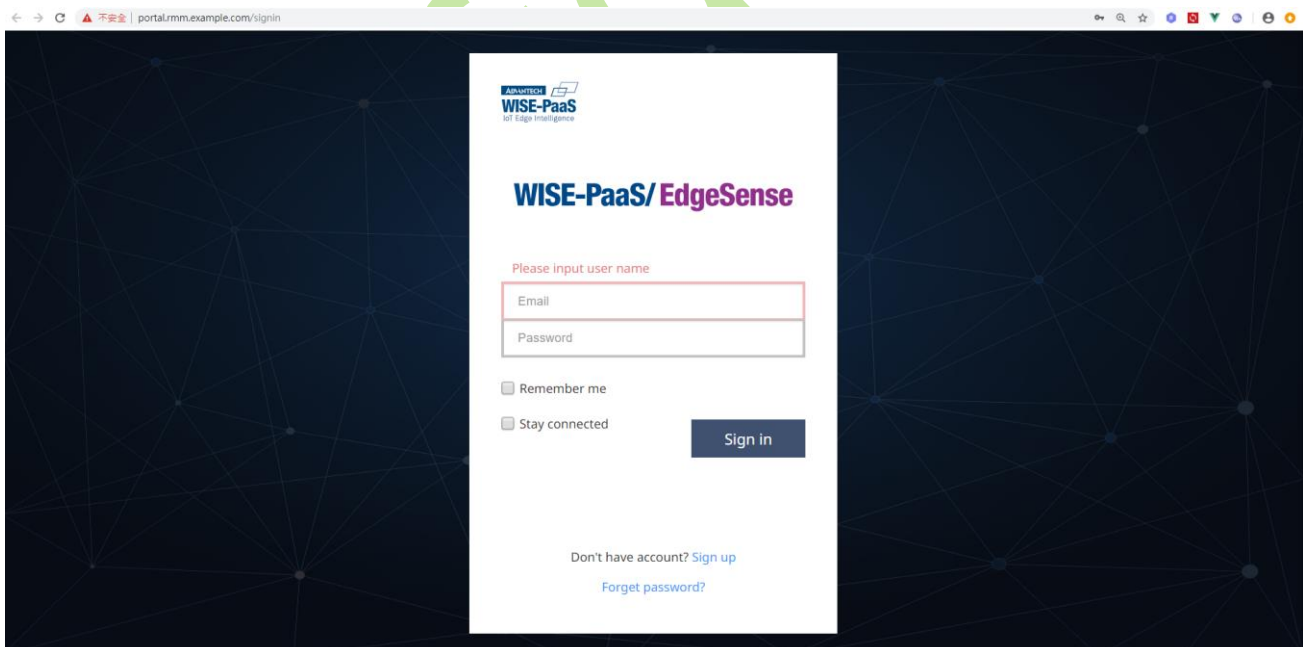
After having the endpoints of the services, you could easily access them by typing the endpoints in the browser. The examples here are based on domain of .example.com. In your case, the domain might be different, you should viewing your endpoints be the previous step.

To get more info about these services, you could refer to the manuals for EdgeSense and monitoring system.

### 6.9.1 EdgeSense

Endpoint — <http://portal.rmm.example.com>

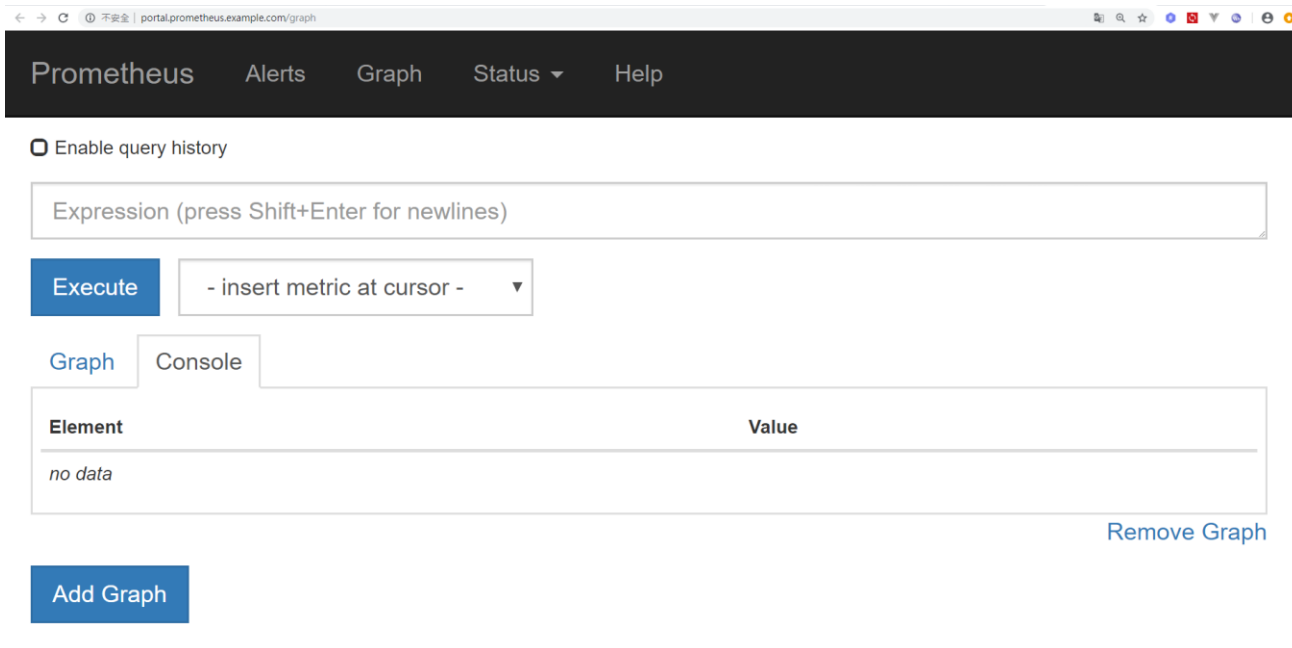
IP Access — `<DataServer-IP>:8087`



### 6.9.2 Prometheus

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

IP Access — **<DataServer-IP>:9090**



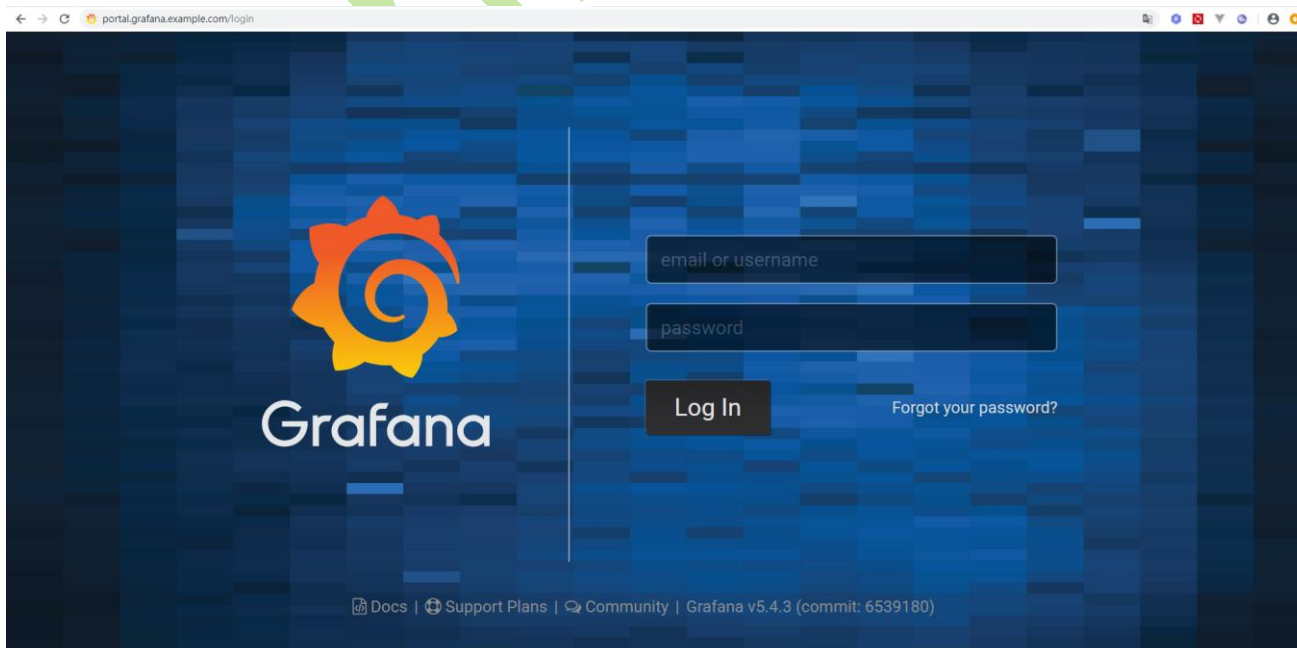
The screenshot shows the Prometheus web interface in a browser. The address bar displays 'portal.prometheus.example.com/graph'. The top navigation bar includes 'Prometheus', 'Alerts', 'Graph', 'Status', and 'Help'. Below the navigation bar, there is a checkbox for 'Enable query history'. A text input field for the query expression is present, with a hint 'Expression (press Shift+Enter for newlines)'. Below the input field is an 'Execute' button and a dropdown menu with the option '- insert metric at cursor -'. There are two tabs: 'Graph' (selected) and 'Console'. Below the tabs is a table with two columns: 'Element' and 'Value'. The table currently shows 'no data'. To the right of the table is a 'Remove Graph' link. At the bottom left is an 'Add Graph' button.

Element	Value
no data	

### 6.9.3 Grafana

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

IP Access — **<DataServer-IP>:3000**



The screenshot shows the Grafana login page in a browser. The address bar displays 'portal.grafana.example.com/login'. The page features the Grafana logo on the left. On the right, there is a login form with two input fields: 'email or username' and 'password'. Below these fields is a 'Log In' button and a link for 'Forgot your password?'. At the bottom of the page, there is a footer with links for 'Docs', 'Support Plans', and 'Community', followed by the text 'Grafana v5.4.3 (commit: 6539180)'.

## 7 FAQ

### 7.1 Accessing

#### 7.1.1 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. (7.2)

#### 7.1.2 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 (6.4) 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.

#### 7.1.3 How should I set the URLs for dashboard and S3 compatible storage in 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.

## 7.2 Authentication

7.2.1 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** (6.3).

7.2.2 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** (6.3).



### 7.2.3 Why can't I do more actions than just viewing on the resources of Data Service

Server?

The role for you is just a viewer and the action authorized for the viewer is only

"viewing."

## 7.3 Unhealthy Pods

### 7.3.1 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 mean the pod is unhealthy. An unhealthy pod could make your services unavailable or not work normally.

### 7.3.2 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.

## 8 Appendix

### 8.1 Ports

Here's the list of ports of all applications. You could access these applications by using

<DataServer-IP>:<Port>

Applications	Port
EdgeSensePortal	8087
	15672 (Portal)
RabbitMQ	1883 (MQTT)
	8883 (MQTT_SSL)
Postgres	5432
Mongo	27017
Minio	9000
Prometheus	9090
Grafana	3000

## 8.2 Access

### 8.2.1 Minio

To access Minio S3 Compatible, you should have accessKey and secretKey. You could use the command below to get them:

```
$ kubectl get cm es-minio -o yaml | grep 'accessKey\|secretKey'
```

```
root@k8s-m1:~# kubectl get cm es-minio -o yaml | grep 'accessKey|secretKey'
root@k8s-m1:~# kubectl get cm es-minio -o yaml | grep 'accessKey\|secretKey'
  "accessKey": "AKIAIOSFODNN7EXAMPLE",
  "secretKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY"
```

The default values:

accessKey - AKIAIOSFODNN7EXAMPLE

secretKey — wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

## 9 References

- [1] kubernetes, "kubernetes," [Online]. Available: <https://kubernetes.io/>.
- [2] docker, "docker," [Online]. Available: [https://en.wikipedia.org/wiki/Docker\\_\(software\)](https://en.wikipedia.org/wiki/Docker_(software)).
- [3] ceph, "ceph," [Online]. Available: [https://en.wikipedia.org/wiki/Ceph\\_\(software\)](https://en.wikipedia.org/wiki/Ceph_(software)).
- [4] kubernetes, "namespaces," [Online]. Available: <https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/>.
- [5] kubernetes, "pods," [Online]. Available: <https://kubernetes.io/docs/concepts/workloads/pods/>.
- [6] kubernetes, "services," [Online]. Available: <https://kubernetes.io/docs/concepts/services-networking/service/>.
- [7] kubernetes, "deployments," [Online]. Available: <https://kubernetes.io/docs/concepts/workloads/controllers/deployment/>.
- [8] kubernetes, "kubectl," [Online]. Available: <https://kubernetes.io/docs/reference/kubectl/overview/>.
- [9] kubernetes, "install kubectl," [Online]. Available: <https://kubernetes.io/docs/tasks/tools/install-kubectl/>.
- [10] nginx, "nginx," [Online]. Available: <https://nginx.org/en/>.
- [11] kubernetes, "nginx-ingress-controller," [Online]. Available: <https://github.com/kubernetes/ingress-nginx>.
- [12] kubernetes, "ingress," [Online]. Available: <https://kubernetes.io/docs/concepts/services-networking/ingress/>.
- [13] T. Fisher, "How to Change DNS Servers in Windows," [Online]. Available: <https://www.lifewire.com/how-to-change-dns-servers-in-windows-2626242>.
- [14] kubernetes, "replicaset," [Online]. Available: <https://kubernetes.io/docs/concepts/workloads/controllers/replicaset/>.