Properly Setting Up Ingress Objects with Security Control

Here's a step-by-step guide on setting up Ingress objects with security control in your Kubernetes cluster:

1. Prerequisites:

•Ensure you have a running Kubernetes cluster and the kubectl command-line tool configured to interact with your cluster.

•Have a TLS certificate and key pair ready for secure communication (optional, but highly recommended). You can obtain these from a Certificate Authority (CA) or generate them for testing purposes using tools like openssl.

2. Create a Secret (Optional):

If you're using a TLS certificate and key pair, create a Kubernetes Secret to store them securely:

Terminal

```
***create a pod
kubectl get nodes
kubectl run caddy --image caddy
kubectl get pod

***create a services
kubectl expose pod caddy --name caddy-svc --port 80
kubectl describe svc caddy-svc


***now create an ingress without TLS
vim ingress.yaml
```

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: simple-ingress
spec:
  rules:
  - host: usamaaslamgill.com
    http:
      paths:
      - pathType: Prefix
        path: "/"
        backend:
          service:
            name: caddy-svc
            port:
              number: 80
```

```
kubectl apply -f ingress.yaml

kubectl describe ingress


*** add domain to hosts file
kubectl get nodes -o wide
minikube ip
```

```
echo $(minikube ip) usamaaslamgill.com | sudo tee -a /etc/hosts
cat /etc/hosts

*** create Certificate TLS key pair
openssl req -x509 -newkey rsa:4096 -sha256 -nodes -keyout tls.key -out tls.crt -
subj "/CN=usamaaslamgill.com" -days 365
ll


***Create secret with TLS data
kubectl create secret tls sec-usamaaslamgill --cert=tls.crt --key=tls.key
kubectl get secret

vim ingress.yaml
```

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: simple-ingress
spec:
  tls:
  - hosts:
      - usamaaslamgill.com
    secretName: sec-usamaaslamgill
  rules:
  - host: usamaaslamgill.com
    http:
      paths:
      - pathType: Prefix
        path: "/"
        backend:
          service:
            name: caddy-svc
            port:
              number: 80
```

```
kubectl apply -f ingress.yaml
kubectl describe ingress

***Test and Verify
curl --cacert tls.crt https://usamaaslamgill.com
```

Explanation of the YAML:

•apiVersion: Specifies the Kubernetes API version used for this resource.
•kind: Indicates the type of resource, which is Ingress in this case.
•metadata: Defines metadata about the Ingress object, including its name.
•spec: The Ingress specification, where we configure the following:
•rules: An array of rules defining how to route traffic.
•host: The hostname or wildcard domain for the Ingress.
•http: Specifies routing configuration for HTTP traffic.
•paths: An array of path definitions for routing within the Ingress.
•path: The path prefix for incoming requests.
•pathType: Optional, specifies how to match the path (e.g., Prefix, Exact). Defaults to Prefix.
•backend: Defines the service and port to route traffic to.

•Tls (Optional): References the Secret containing the TLS certificate and key if using HTTPS.

4. Apply the Ingress:

Bash

```
kubectl apply -f my-ingress.yaml
```

Use code with caution.

5. Verify Ingress:

Once the Ingress is applied, you can verify its functionality and security features:

•**Access the application:** Try accessing your application using the specified hostname (e.g., my-app.example.com) in your web browser.

•**Verify TLS (if enabled):** If you used TLS, your browser should indicate a secure connection (https) and display the certificate details.

Additional Considerations:

•**Authentication:** Consider implementing additional security measures like basic authentication or authorization mechanisms (e.g., RBAC) to restrict access to your application through the Ingress.

•**Load Balancing:** Ingress can be used with a load balancer to distribute traffic across multiple instances of your application.

•**Advanced Routing:** Explore advanced Ingress features like annotations for path-based routing, rewrites, and service discovery options.

Remember, this is a basic approach, and the specific configuration will vary based on your requirements and chosen security mechanisms. Refer to the official Kubernetes Ingress documentation for further details and advanced configuration options: https://kubernetes.io/docs/concepts/services-networking/ingress/