Restricting Access to the Kubernetes API Server (Cluster Hardening)

Securing your Kubernetes cluster hinges on restricting access to the Kubernetes API server, the central control plane component. a step-by-step approach to achieve this:

1. Utilize Authentication Mechanisms:

- Implement mechanisms to verify the identity of users and service accounts attempting to access the API server.
- Two common options are:
- **Client certificates:** Issue client certificates to authorized users and services, and configure the API server to require them for access.
- **API tokens:** Generate short-lived API tokens for specific users or services, offering granular control over access duration and permissions.

2. Client Certificate Authentication (Example):

- **Generate a Certificate Authority (CA):** Use openssl or a dedicated CA tool to generate a CA, its private key, and a server certificate for the API server.
- **Create client certificates:** Generate client certificates and private keys for authorized users and service accounts.
- **Configure the API server:** Edit the API server configuration file (e.g., kube-apiserver.yaml) to:
- Set --client-ca-file to the path of the CA certificate.
- Set --request-header-allowed-names and --request-header-client-ca-file based on your configuration.
- **Distribute client certificates:** Securely distribute client certificates and private keys to authorized users and service accounts.

3. API Token Authentication (Example):

- **Enable API token generation:** In your cluster configuration, enable the TokenRequest API group for the authentication API server.
- **Generate API tokens:** Use kubectl commands like token-request to create tokens with specific permissions and expiration times for authorized users or services.

- **Configure clients:** Update client tools (e.g., kubectl) to use the generated API token for authentication when interacting with the API server.

4. Implement Authorization (RBAC):

- Enforce Role-Based Access Control (RBAC) to define the level of access granted to users and service accounts even after successful authentication.
- Create roles, cluster roles, rolebindings, and clusterrolebindings to define access permissions for various resources and actions within the cluster.

5. Additional Considerations:

- **Limit API server exposure:** Restrict access to the API server by binding it to a specific IP address or using a network policy to limit access only from authorized sources.
- **Monitor API server activity:** Implement monitoring tools to track API server activity and identify any suspicious or unauthorized access attempts.
- **Minimize use of default service accounts:** Disable or restrict the use of default service accounts assigned to nodes or pods, as they might grant excessive permissions.
- **Keep Kubernetes and its components updated:** Regularly update Kubernetes and its components to address potential vulnerabilities related to authentication and authorization mechanisms.

By implementing these steps and maintaining a vigilant security posture, you can significantly restrict unauthorized access to your Kubernetes API server and enhance the overall security of your cluster.

Always Consult the official Kubernetes documentation:
https://kubernetes.io/docs/reference/access-authn-authz/authentication/

DEMO

```
kubectl get nodes
kubectl get pods -A
kubectl proxy --address='0.0.0.0' --accept-hosts='^*$'
#browser
localhost:8001/
```

```
kubectl create namespace test
kubectl run web --image nginx -n test
kubectl get pods -n test
```

#browser
localhost:8001/api/v1/namespaces/test/pods

```
kubectl config view
```

#how to secure api server?
#avoid anonymous request to API server

```
curl https://192.168.56.10:6443 -k

kubectl get nodes -o wide
```

#default http server comes with 2 different ports

# 8080 port is not secure so implement 6443 port it will apply 3 levels of checks
# so we will use bind address
# I want to use secure but I don’t want 6443 I want t different port

```
kubectl describe pod kube-api-server-master-node -n kube-system
```

goto master node

```
cd /etc/kubernetes/manifests
ll
sudo cat kube-apiserver.yaml
```

DISABLE ANONYMOUS REQUESTS

```
--anonymous-auth=true    #that means its enable
```