KPLABS Course

Certified Kubernetes Administrator

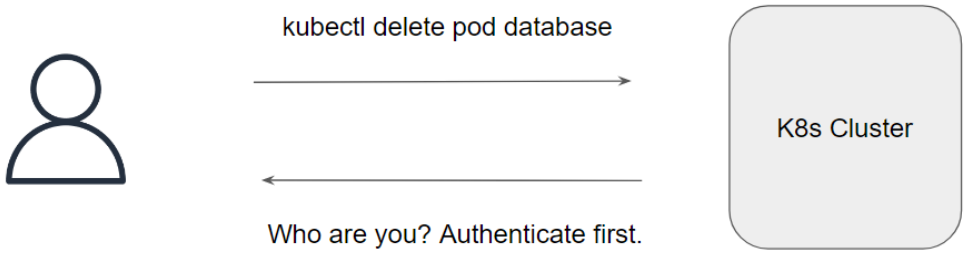
Security

ISSUED BY
Zeal Vora

REPRESENTATIVE
instructors@kplabs.in

Module 1: Authentication

Authentication is the process or action of verifying the identity of a user or process.

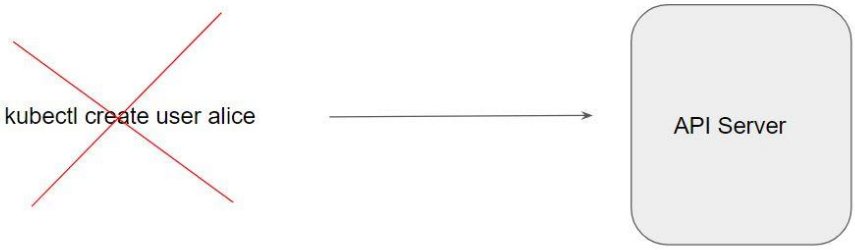


Kubernetes Clusters have two categories of users:

- Normal Users.
- Service Accounts

Kubernetes does not have objects which represent normal user accounts.

Kubernetes does not manage the user accounts natively.



There are multiple ways in which we can authenticate. Some of these include:

- Usernames / Passwords.

- Client Certificates
- Bearer Tokens

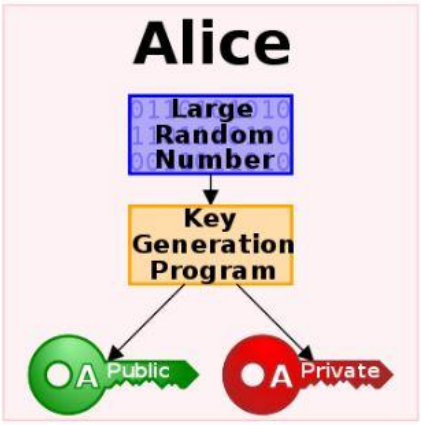
Module 2: Asymmetric Key Encryption

Asymmetric cryptography uses public and private keys to encrypt and decrypt data.

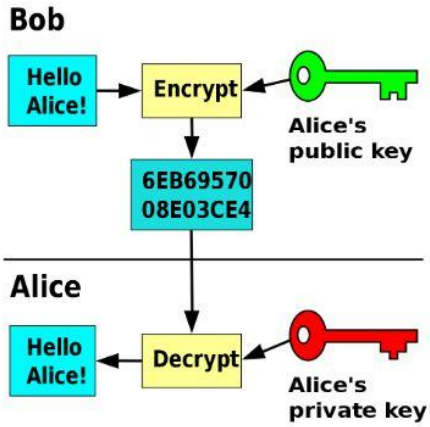
One key in the pair can be shared with everyone; it is called the public key. The other key in the pair is kept secret; it is called the private key.

Either of the keys can be used to encrypt a message; the opposite key from the one used to encrypt the message is used for decryption.

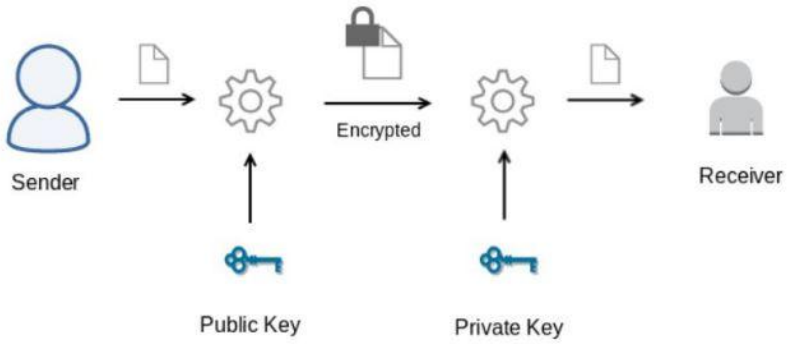
Step 1: Generation of Keys



Step 2: Encryption and Decryption



High-Level Steps:



Use-Case of Asymmetric Key Encryption Steps:

Step 1:

User zeal wants to log in to the server. Since the server uses a public key authentication, instead of taking the password from the user, the server will verify if the User claiming to be zeal actually holds the right private key.

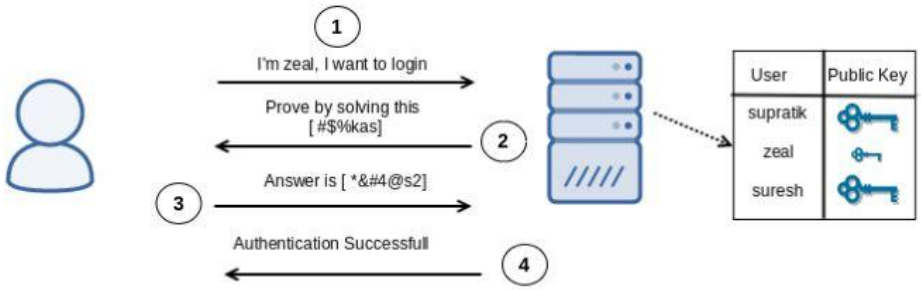
Step 2:
The server creates a simple challenge, 2+3=? and encrypts this challenge with the Public Key of the User and sends it back to the User. The challenge is sent in an encrypted format.

Step 3:

Since the user zeal holds the associated private key, he will be able to decrypt the message and compute the answer, which would be 5. Then, he will encrypt the message with the private key and send it back to the server.

Step 4:

The server decrypts the message with the user's Public Key and checks if the answer is correct. If yes, then the server will send an Authentication Successful message and the user will be able to log in.



Because of the advantage that it offers, Asymmetric key encryption is used by variety of protocols.

Some of these include:

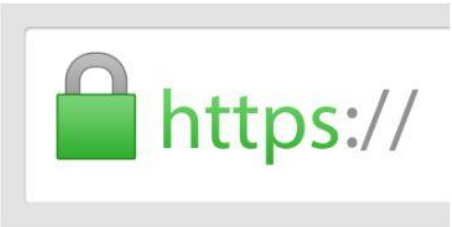
- PGP
- SSH
- Bitcoin
- TLS
- S/MIME

Module 3: Understanding SSL/TLS

HTTPS is an extension of HTTP.

In HTTPS, the communication is encrypted using Transport Layer Security (TLS)

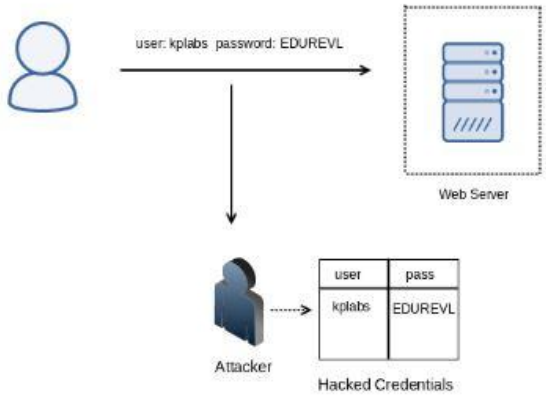
The protocol is therefore also often referred to as HTTP over TLS or HTTP over SSL.



Scenario 1: MITM Attacks

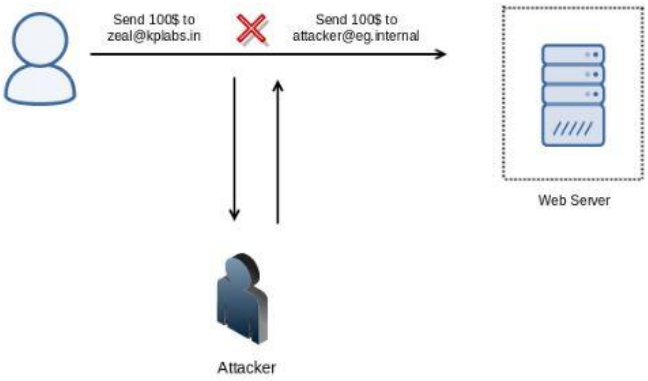
User is sending their username and password in plaintext to a Web Server for authentication over a network.

There is an Attacker sitting between them doing a MITM attack and storing all the credentials he finds over the network to a file:



Scenario 2: MITM & Integrity Attacks

Attacker changing the payment details while packets are in transit.



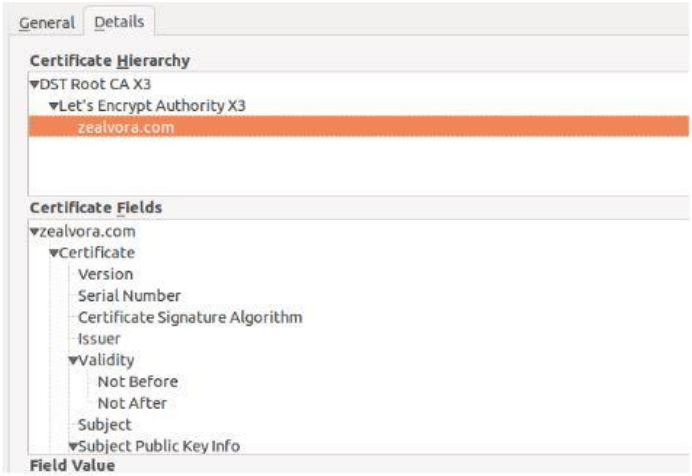
Introduction to SSL/TLS

To avoid the previous two scenarios (and many more), various cryptographic standards were clubbed together to establish secure communication over an untrusted network and they were known as SSL/TLS.

Protocol	Year
SSL 2.0	1995
SSL 3.0	1996
TLS 1.0	1999
TLS 1.1	2006
TLS 1.2	2008
TLS 1.3	2018

Every website has a certificate (like a passport that is issued by a trusted entity).

The certificate has a lot of details like the domain name it is valid for, the public key, validity, and others.



The browser (clients) verifies if it trusts the certificate issuer.

It will verify all the details of the certificate.

It will take the public key and initiate a negotiation.

Asymmetric key encryption is used to generate a new temporary

Symmetric key which will be used for secure communication.

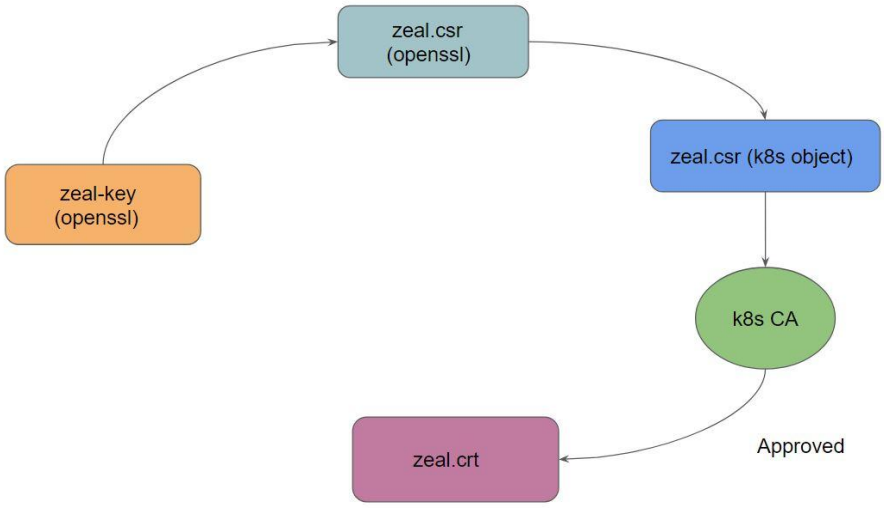
The following snapshot shows sample web-server configuration:

```
server {
       listen         80;
       server_name    zealvora.com;
       return         301 https://$server_name$request_uri;
}


server {
  server_name zealvora.com;
  listen 443 default ssl;
  server_name zealvora.com;
  ssl_certificate /etc/letsencrypt/archive/zealvora.com/fullchain1.pem;
  ssl_certificate_key /etc/letsencrypt/archive/zealvora.com/privkey1.pem;

  location / {
    root /websites/zealvora/;
    include location-php;
    index index.php;
 }
 location ~ /.well-known {
        allow all;
 }
}
```

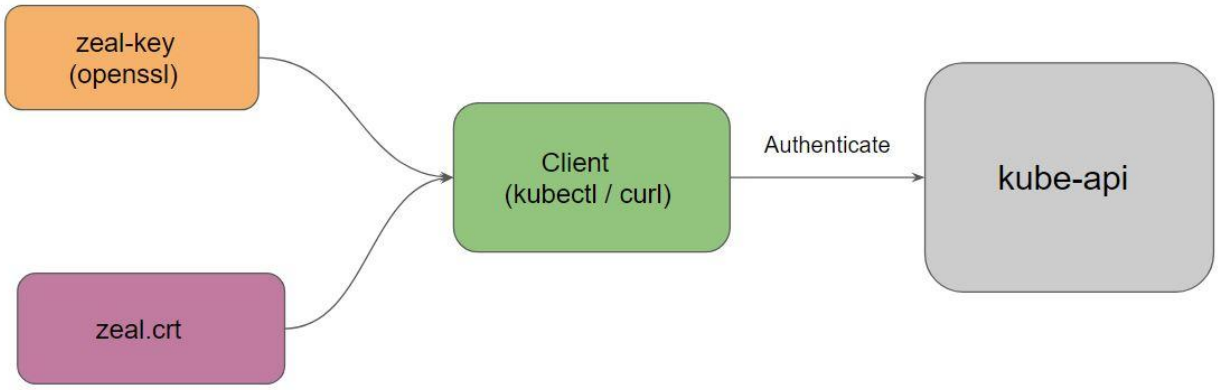
Module 4: Creating TLS Certificate for Authentication

Our goal is to set up authentication based on X509 Client certificates.

User A should be able to authenticate with Kubernetes Cluster with his certificates.

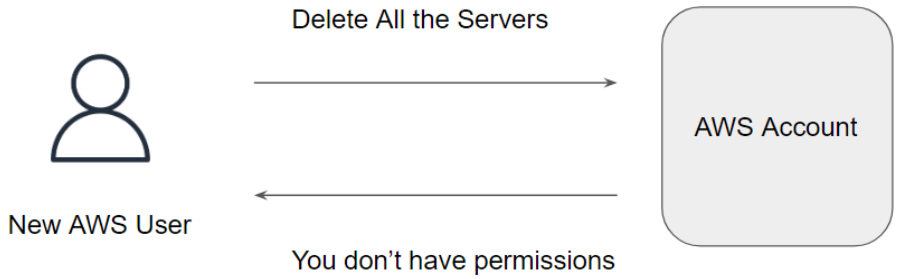


High-Level Authentication Steps:



Module 5: Authorization

Authorization is the process of giving someone permission to perform some operation.



By default, the user does not have any permission granted to them.

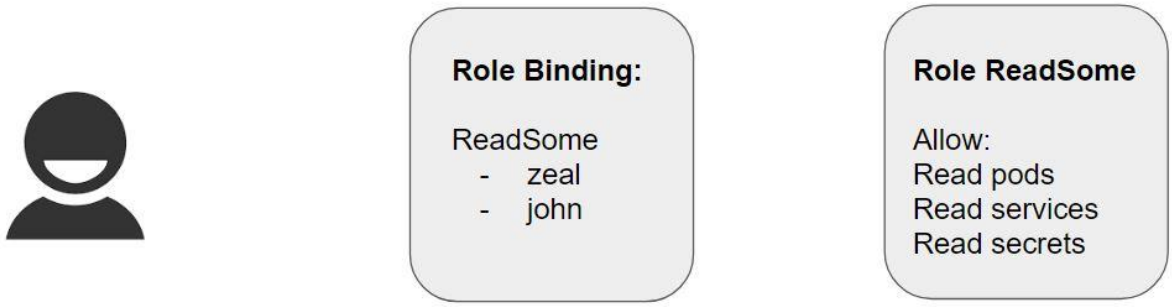
Kubernetes provides multiple authorization modules like:

- AlwaysAllow
- AlwaysDeny
- Attribute-Based Access Control (ABAC)

- Role-based access control (RBAC)
- Node
- WebHook

A role contains rules that represent a set of permissions

A role binding grants the permissions defined in a role to a user or set of users.



Module 6: API Groups, Resources and Verbs

6.1 Overview of API Groups

API Groups makes it easier to extend the Kubernetes API.

Currently there are several groups available:

The core group, often referred to as the legacy group, is at the REST path /api/v1

The named groups are at REST path /apis/$GROUP_NAME/$VERSION

We have flexibility to enable/disable a specific API Groups in API server.

6.2 Resources and Verbs

Within a specific API Group, there are multiple resources available.

Example: Pods, namespaces, nodes,

Verbs are generally the type of requests that can be sent to a resource.

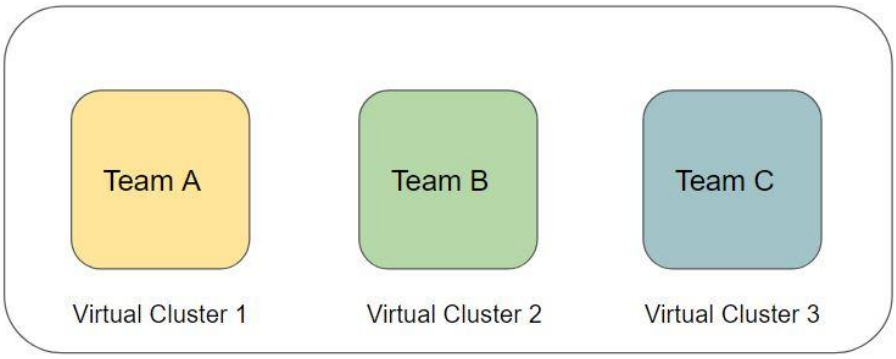
Example Verbs:

Get, list, create, update, patch, watch and others.

Module 7: Access Control based on Namespace

Example Use-Case:

You provide all the users within Team A full access on “Pods” resources [only for Team A NS]



Important Pointer:

A Role can only be used to grant access to resources within a single namespace.

There can be multiple role with the same name in two different namespaces.

Module 8: ClusterRole and ClusterRoleBinding

A Role can only be used to grant access to resources within a single namespace.

A ClusterRole can be used to grant the same permissions as a Role, but because they are cluster-scoped, they can also be used to grant access to:

cluster-scoped resources (like nodes)
namespaced resources (like pods) across all namespaces

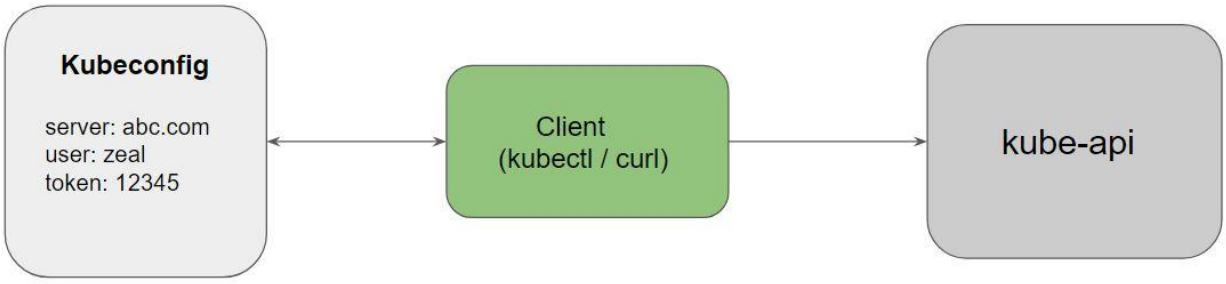
Important Pointers:

A RoleBinding may also reference a ClusterRole to grant the permissions to namespaced resources defined in the ClusterRole within the RoleBinding's namespace

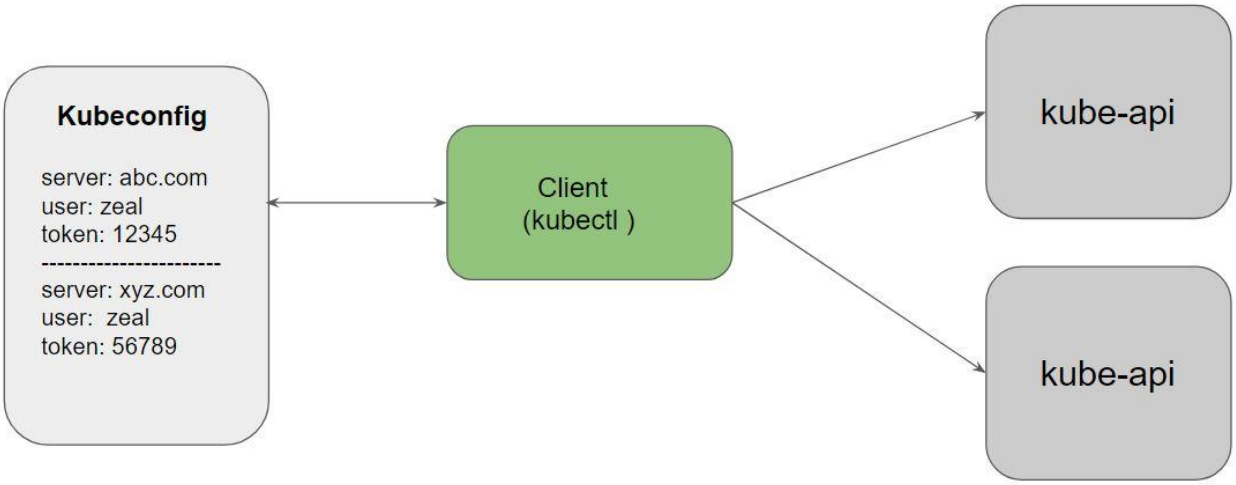
This allows administrator to have set of central policy which can be attached via RoleBinding so it is applicable at a per-namespace level.

Module 9: Understanding Kubeconfig

Kubectl command uses kubeconfig files to find the information about the cluster, username, authentication mechanisms and others.

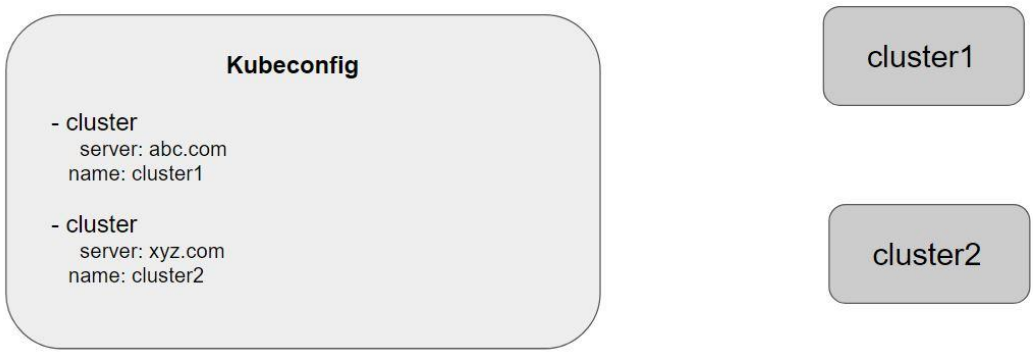


Kubeconfig file can also have details associated with multiple kubernetes clusters.



Step 1: Understanding Clusters Field

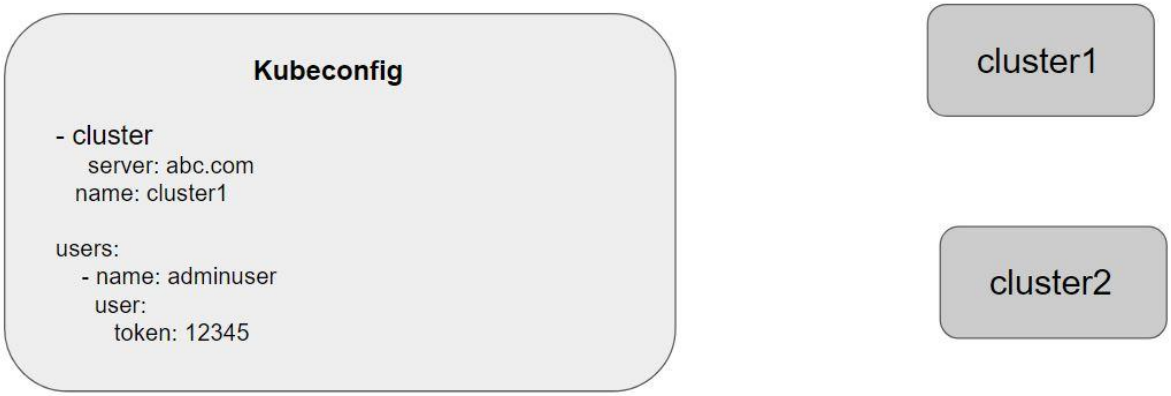
Cluster field has details related to the URL of your Kubernetes Cluster and it's associated information.



Step 2: Users Field

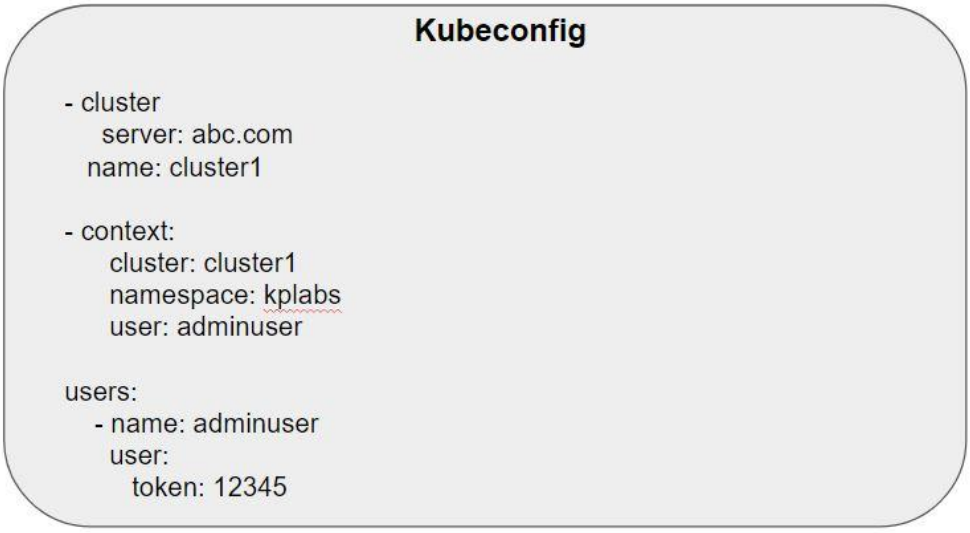
User field contains authentication specific information like username, passwords.

There can be different type of authentication mechanisms (user/pass, certificates, tokens etc



Step 3: Context Field

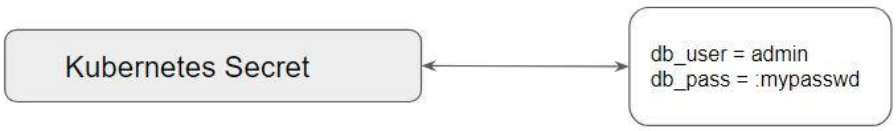
Context groups information related to cluster, user and namespace.



Module 10: Kubernetes Secrets

A Secret is an object that contains a small amount of sensitive data such as a password, a token, or a key.

Allows customers to store secrets centrally to reduce risk of exposure.
Stored in ETCD database.



Following is the syntax for creating a secret via CLI:

kubectl create secret [TYPE] [NAME] [DATA]

Elaborating Type:

i) Generic:
File (--from-file)
directory
literal value

ii) Docker Registry
iii) TLS

Join Our Discord Community

We invite you to join our Discord community, where you can interact with our support team for any course-based technical queries and connect with other students who are doing the same course.

Joining URL:

http://kplabs.in/chat