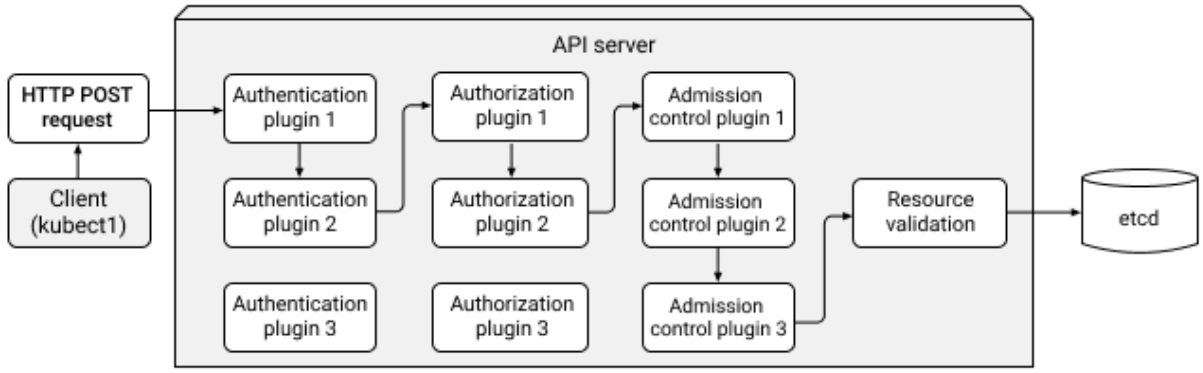
K8S 36: Kubernetes. Advanced. Authentication (authn) и Authorization (authz)

Описание:

Для доступа к ресурсам kubernetes применяется три шага:

- authentication — осуществляет логин для пользователя,
- authorization — авторизует доступ к ресурсу,
- admission control — может изменить или проверить объект на основе дополнительных правил (например, таких как квоты).



Authentication

В kubernetes нет такого объекта, как пользователь или пароль. Но и без этого kubernetes может использовать имена пользователя для доступа к объектам. Есть два типа пользователей: люди и служебные пользователи.

- Люди — это пользователи вне kubernetes, которые получают к нему доступ, авторизуясь через сертификат, куку и подобное.
- Служебные пользователи — Service Accounts — это пользователи внутри кластера. Большинство из них создается автоматически при разворачивании кластера, но мы можем сделать дополнительные в каждом namespace. При определенной настройке kubernetes может обслуживать запросы и от анонимных пользователей.

Типы авторизации для внешних пользователей (могут активироваться при помощи передачи параметров запуска к Kubernetes API Server):

- client certificate — для включения авторизации по сертификатам нужно добавить в параметры запуска API сервера --client-ca-file=FILE. CA в этом файле будут валидировать клиентские сертификаты.
- static token file — для включения авторизации по сертификатам нужно добавить в параметры запуска API сервера --token-auth-file=FILE. В этом файле будут

записаны токены и их нельзя будет изменить без перезапуска API сервера. Также срок их действия не будет ограничен.

- bootstrap token — используется для установки нового кластера, токен генерируется автоматически.
- static password file — --basic-auth-file=SOMEFILE, содержит пользователей и пароли. Изменить без рестарта API сервера нельзя, срок действия не ограничен.
- service account token — это автоматически генерируемый токен для служебного пользователя. Он добавляется к pod через serviceaccount admissioncontroller и позволяет взаимодействовать с kubernetes изнутри pod.
- openid connect token — oauth2 авторизация, примеры — ActiveDirectory, Google, Facebook. Переносит процесс авторизации на внешний сервис.
- webhook token auth — переносит верификацию токенов на внешний сервис.
- keystone password — --experimental-keystone-url=<AuthURL> -- авторизация через альфа-версию keystone.
- auth proxy — авторизация через прокси, который реализует свою дополнительную логику.

Одновременно можно включить несколько видов авторизации. После прохождения аутентификации можно взаимодействовать с API kubernetes. В дело вступают авторизаторы.

Authorization

Это модули, которые запрещают или разрешают определенное действие. Примеры:

- Node Authorize — специальный авторизатор. Служит для проверки запросов, отправленных с kubelet рабочих узлов (https://kubernetes.io/docs/reference/access-authn-authz/node/).
- ABAC (Attribute based control authorizer) — контроль доступа на основе атрибутов. Устаревший авторизатор, редко используется. Включается через --authorization-mode=ABAC --authorization-policy-file=PolicyFile.json Пример:

```
{
  "apiVersion": "abac.authorization.kubernetes.io/v1beta1",
  "kind": "Policy",
  "spec": {
    "user": "nkhare",
    "namespace": "lfs158",
    "resource": "pods",
    "readonly": true
  }
}
```

-
- WebHookAuthorizer — авторизатор через вебхук. Делает запрос на внешний сервис. Очень простой, принимает ответ только true или false. Включается --authorization-webhook-config-file=SOME_FILENAME.

- RBAC (Role based authorizer) — доступ с разделением по ролям. В данный момент — основной используемый авторизатор по умолчанию. Включается через --authorization-mode=RBAC.

Admission control

Данные опции позволяют обрабатывать запросы, когда они, казалось бы, прошли этап аутентификации и авторизации. Как правило, они используются для того, чтобы проверить.

Есть 2 типа admission controller:

- валидирующие (validating) — подтверждают, что переданные данные верны и валидны;
- изменяющие (mutating) — изменяют входные данные.

На самом деле, любой контроллер может выполнять обе роли, если это требуется. И, соответственно, одновременно может быть включено несколько контроллеров и, если на любом из них проваливается запрос, то весь запрос считается провальным.

Для включения конкретных Admission Controller требуется передать параметр --enable-admission-plugins=X,Y,.. в параметры Kubernetes API Server, где X, Y - это имена требуемых контроллеров и разделяться они должны запятой без пробелов до и после.

Для отключения же используется опция --disable-admission-plugins с таким же синтаксисом.

Если говорить о примерах, то LimitRanger отвечает за то, чтобы объекты типа LimitRange обрабатывались, а PersistentVolumeClaimResize отвечает за возможность расширять PVC, как мы обсуждали ранее.

Полезные ссылки:

- Controlling Access to the Kubernetes API (official docs)
- Authenticating (official docs)
- Authorization Overview (official docs)
- Using Admission Controllers (official docs)
- Securing the Configuration of Kubernetes Cluster Components

Задание:

1. Включите Admission controller AlwaysPullImages в вашем Minikube (команды и вывод сохраните).
2. Найдите, исходя из параметров Kubernetes API Server вашего Minikube кластера, список включенных модулей Authorization и Admission controllers.
3. На проверку отправьте список включенных модулей Authorization и Admission controllers, описание, как вы их получили, а также сохраненные команды и выводы.