Developer Series

Join our webinar series to expand your developer skills!

Security Tuesdays
Data Science and AI Wednesdays
Cloud Native and Red Hat OpenShift Thursdays

meetup.com/IBM-Cloud-MEA

IBM **Developer**

IBM

Introduction to Kubernetes Security

Asna Javed
Lead Developer Advocate, Pakistan

Naiyarah Hussain
Lead Developer Advocate, UAE

IBM Developer

IBM

Let’s get started

- Code Lab: https://securecube.gitbook.io/securecube/
- Sign up for your IBM Cloud account – https://ibm.biz/kubesec
- Get your free Kubernetes Cluster at:

  URL: https://kubesec.mybluemix.net

  Key: oslab

Agenda

Security in Microservices

Causes for Concern

Approaching Kubernetes Security

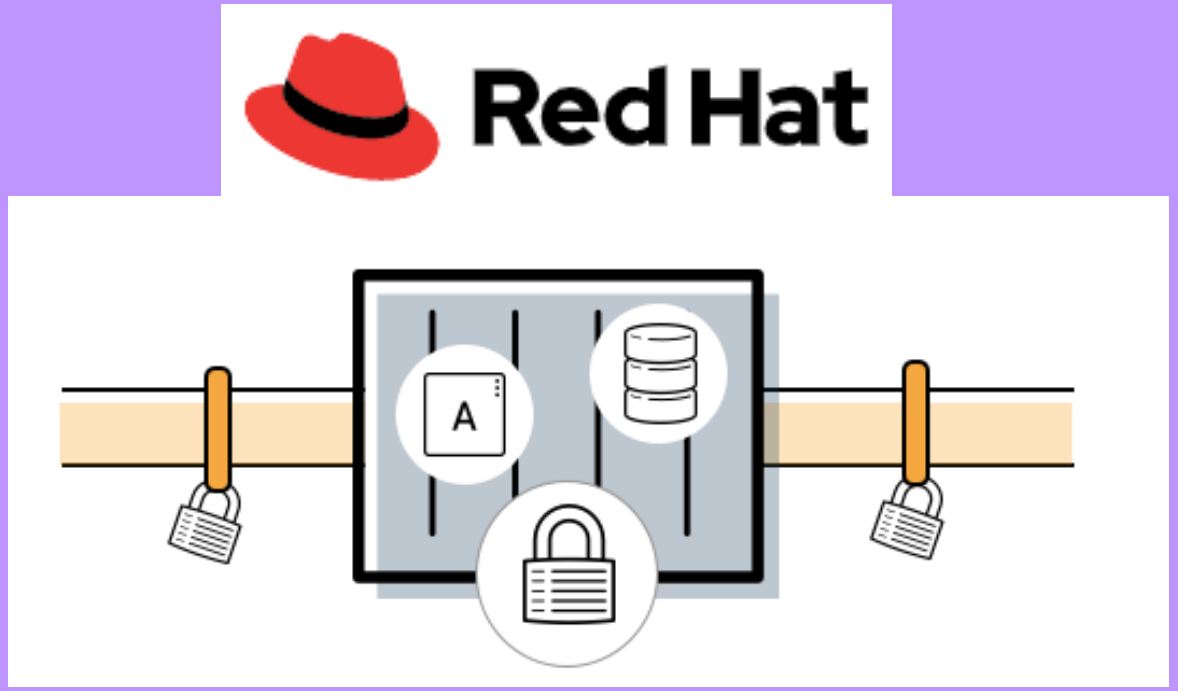
Important Components of Kubernetes Security

attack vectors of the Kubernetes architecture

Demo & Code Lab with FREE Clusters

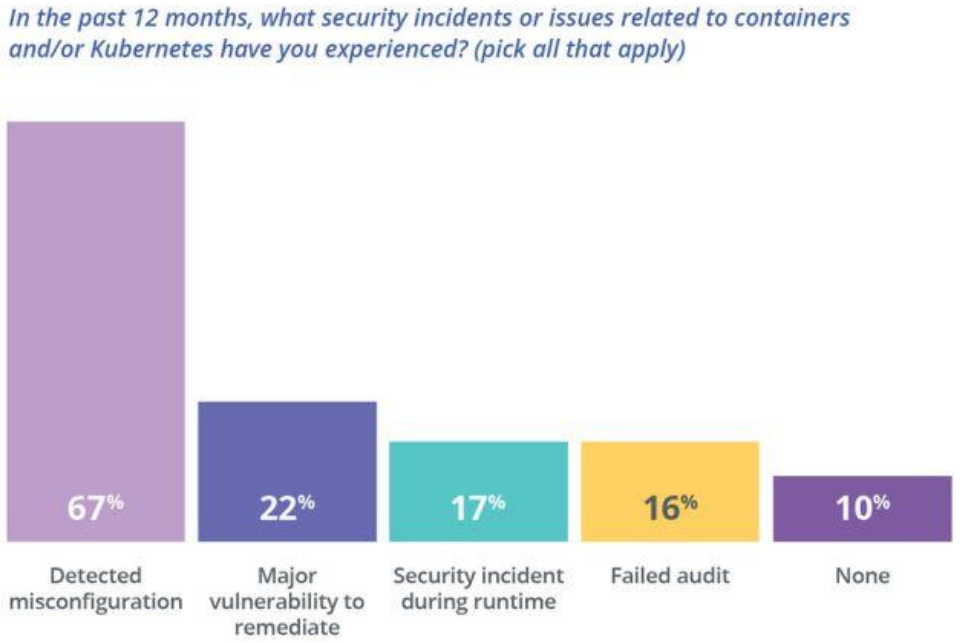
Microservices Security

When exposing an application to external clients, in a Microservices setting, there is a bit of concern regarding the security measurements.

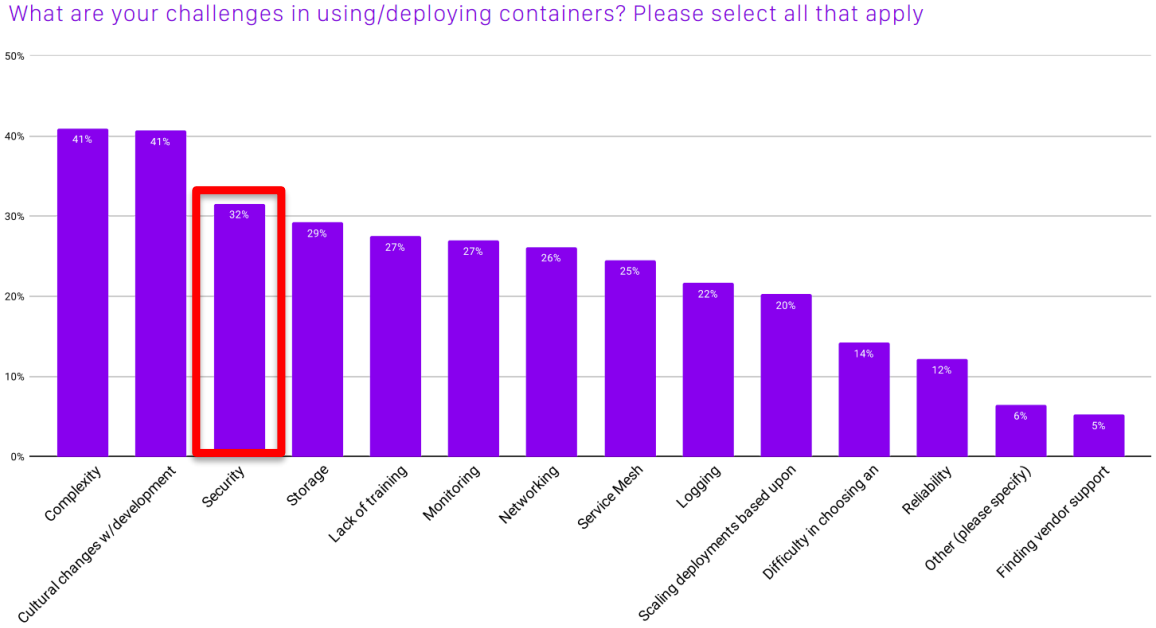


Causes for Concern

- Human error causes majority of security incidents in Kubernetes (StackRox, 2020)
- Security is consistently a significant challenge (CNCF, 2020)
- Recent incidents

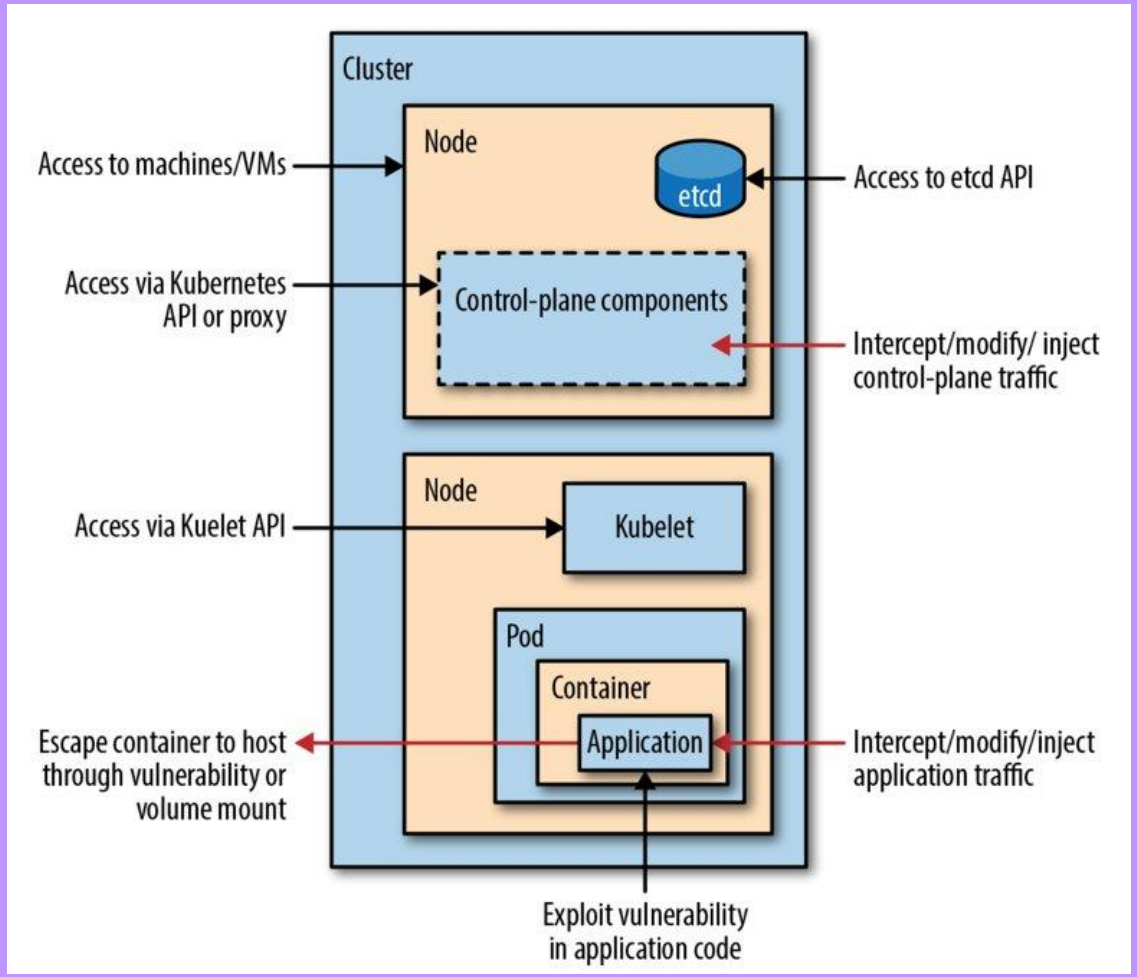


https://www.stackrox.com/kubernetes-adoption-security-and-market-share-for-containers/



https://www.cncf.io/wp-content/uploads/2020/11/CNCF_Survey_Report_2020.pdf

Approaching Kubernetes Security



https://www.oreilly.com/library/view/kubernetes-security/9781492039075/ch01.html

Remember your Security Principles

Defense in Depth

Preferable to have *several layers of defense* against attacks on your Kubernetes cluster.

If you're relying on a single defensive measure, attackers might find their way around it

Least Privilege

The principle of least privilege tells us to restrict access so that different components can access only the information and resources they need to operate correctly.

In the event of a component being compromised, an attacker can reach only the subset of information and resources available to that component. This limits the "blast radius" of the attack

Limit the Attack Surface

The attack surface is the set of all possible ways a system can be attacked.

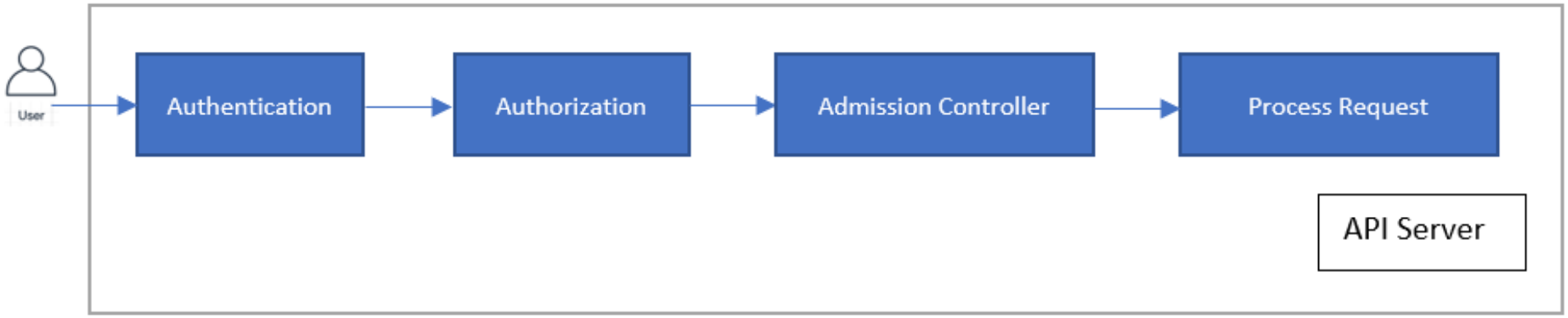
The more complex the system, the bigger the attack surface, and therefore the more likely it is that an attacker will find a way in.

Important Concepts of Kubernetes Security

1- Authentication

2- Authorization

3- Admission Controllers



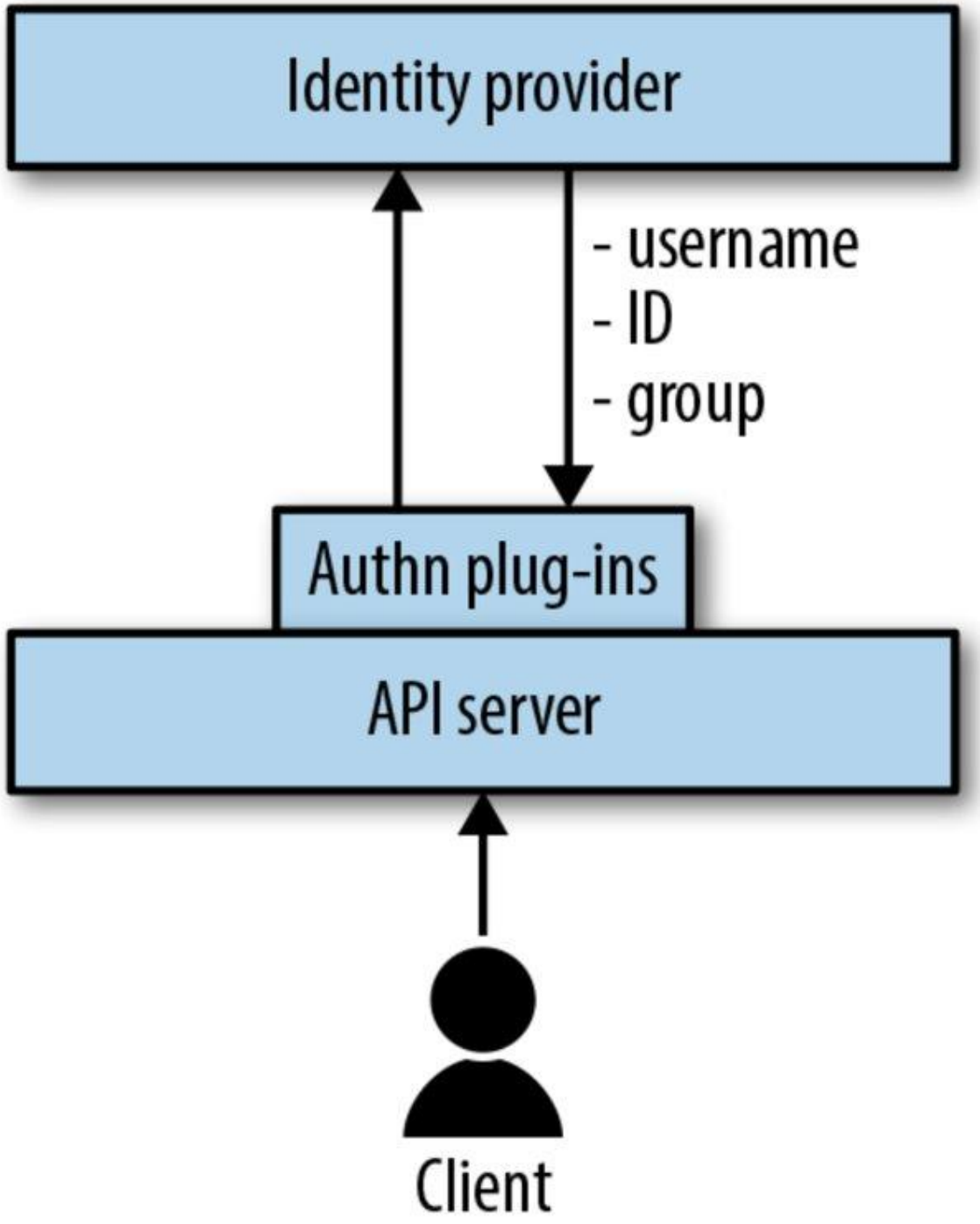
Kubernetes Authentication

All requests in Kubernetes originate from

a) external users, b) service accounts, c) Kubernetes components.

Kubernetes uses one or more of these authentication strategies:

- **Client certificates**
- **Static Tokens**
- **Basic authentication**
- **Service account tokens**
- **Webhook tokens**



Kubernetes Authentication – Client Certificates

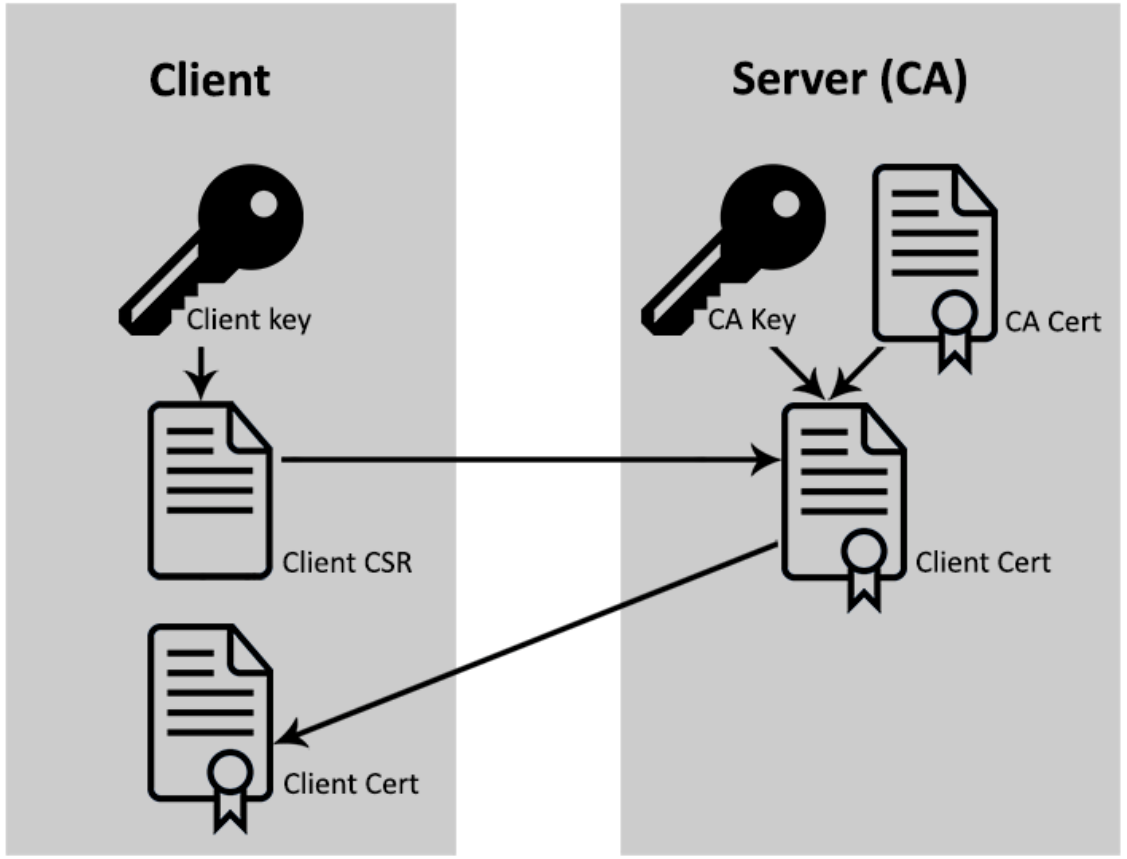
Kubernetes uses one or more of these authentication strategies

Using X509 **Certificate Authority** (**CA**) certificates is the most common authentication strategy

It can be enabled by passing **--client-ca-file=file_path** to the server

The file passed to the API server has a list of CAs, which creates and validates client certificates in the cluster.

kube-apiserver --advertise-address=192.168.99.104 --allow-privileged=true --authorization-mode=Node,RBAC --client-ca-file=/var/lib/minikube/certs/ca.crt



https://blog.codecentric.de/en/2018/08/x-509-client-certificates-with-spring-security/

Kubernetes Authentication – Static Tokens

The API server uses a static file to read the bearer tokens.

This static file is passed to the API server using **--token-auth-file=<path>**

The token file is a comma-separated file consisting of **secret**, **user**, **uid**, **group1**, and **group2**

The token is passed as an HTTP header in the request:

Authorization: Bearer 66e6a781-09cb-4e7e-8e13-34d78cb0dab6

The tokens persist indefinitely, and the API server needs to be restarted to update the tokens. This is *not* a recommended authentication strategy.

These tokens can be easily compromised if the attacker is able to spawn a malicious pod in a cluster.

Once compromised, the only way to generate a new token is to restart the API server.

Kubernetes Authentication – Basic Authentication

Kubernetes also supports basic authentication

This can be enabled by using **basic-auth-file=<path>**

The authentication credentials are stored in a CSV file as **password**, **user**, **uid**, **group1**, and **group2**

The username and password are passed as an authentication header in the request:

Authentication: Basic base64(user:password)

Basic authentication should not be used in production clusters

Kubernetes Authentication - Service Account Tokens

The service account authenticator is automatically enabled

Service accounts are created by the **kube-apiserver** and are associated with the pods.

It verifies signed bearer tokens

The signing key is specified using **--service-account-key-file**. If this value is unspecified, the Kube API server's private key is used

To create a service account test, you can use the following:

kubectl create serviceaccount test

The service account has associated secrets, which includes the CA of the API server and a signed token

kubectl get serviceaccounts <service account name> -o yaml

kubectl get secret <secret name> -o yaml

Kubernetes Authentication – Webhook Tokens

Kubernetes makes a call to a REST API outside the cluster to determine the user's identity

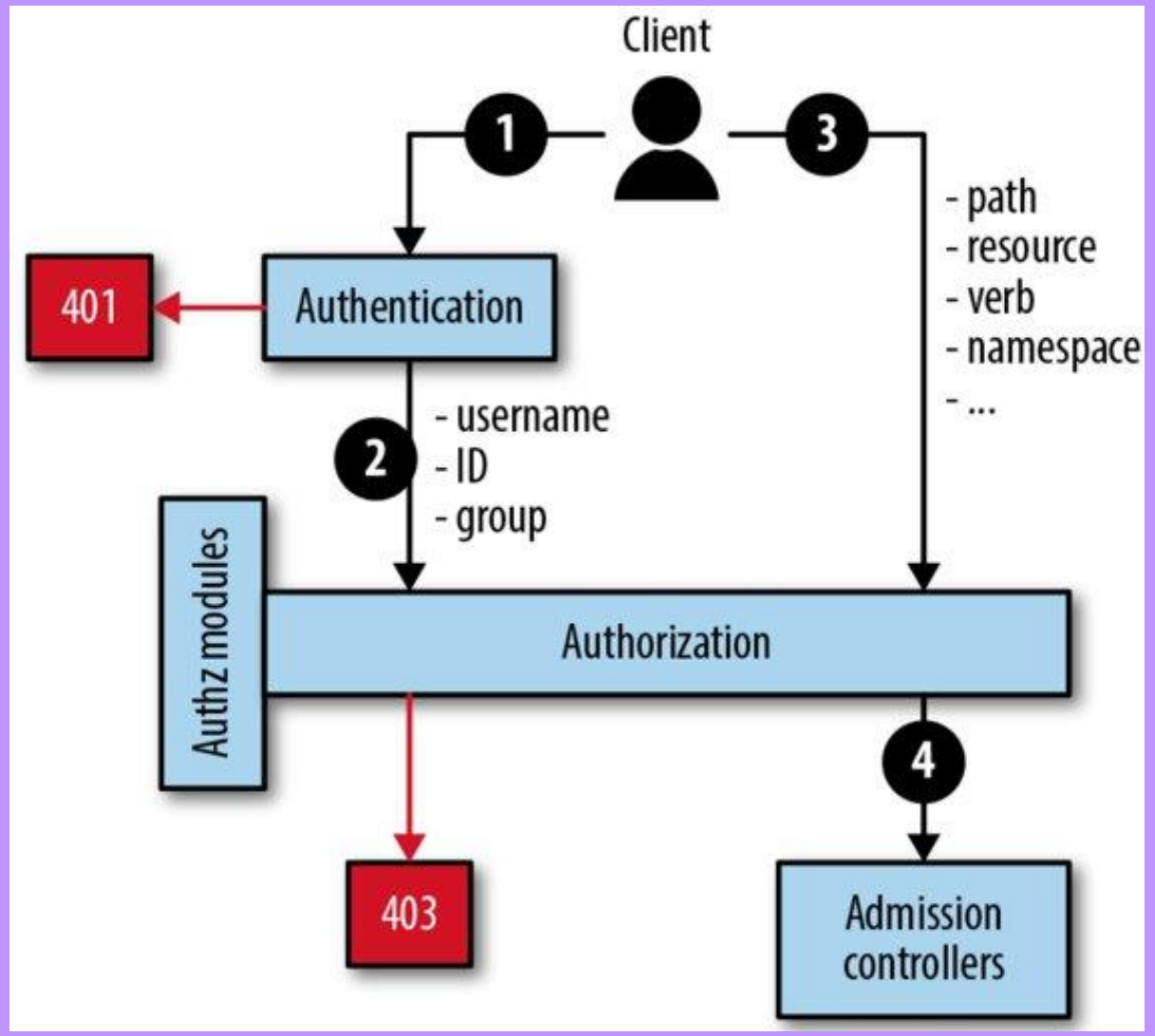
Webhook mode for authentication can be enabled by passing **--authorization-webhook-config-file=<path>** to the API server

```
clusters:
  - name: name-of-remote-authn-service
    cluster:
      certificate-authority: /path/to/ca.pem
      server: https://authn.example.com/authenticate
```

Kubernetes Authorization

Once the origin of the request is identified, active authorization modules evaluate the attributes of the request against the authorization policies of the user to allow or deny a request.

Each request passes through the authorization module sequentially and if any module provides a decision to allow or deny, it is automatically accepted or denied.



Kubernetes Authorization – Request Attributes

Authorization modules parse a set of attributes in a request to determine whether the request should be parsed, allowed, or denied.

Following are the Request attributes -

User: The originator of the request. This is validated during authentication

Group: The group that the user belongs to. This is provided in the authentication layer

API: The destination of the request

Request verb: The type of request, which can be **GET**, **CREATE**, **PATCH**, **DELETE**, and more

Resource: The ID or name of the resource being accessed

Namespace: The namespace of the resource being accessed.

Request path: If the request is for a non-resource endpoint, the path is used to check whether the user is allowed to access the endpoint. This is true for the **api** and **healthz** endpoints

Kubernetes Authorization – Nodes

Node authorization mode grants permissions to kubelets to access services, endpoints, nodes, pods, secrets, and persistent volumes for a node.

The kubelet is identified as part of the **system:nodes** group with a username of **system:node:<name>** to be authorized by the node authorizer. This mode is enabled by default in Kubernetes.

The API server uses the **--authorization-mode=Node** flag to use the node authorization module

Kubernetes Authorization – ABAC

With ABAC, requests are allowed by validating policies against the attributes of the request

ABAC authorization mode can be enabled by using **--authorization-policy-file=<path>** and **--authorization-mode=ABAC** with the API server

The policies include a JSON object per line. Each policy consists of the following:

Version: The API version for the policy format.

Kind: The **Policy** string is used for policies.

Spec: This includes the user, group, and resource properties, such as **apiGroup**, **namespace**, and **nonResourcePath** (such as **/version**, **/apis**, **readonly**) to allow requests that don't modify the resource.

Example:

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

This policy allows a kubelet to read any pods. ABAC is difficult to configure and maintain. It is not recommended that you use ABAC in production environments

Kubernetes Authorization – Webhooks

Similar to webhook mode for authentication, webhook mode for authorization uses a remote API server to check user permissions.

Webhook mode can be enabled by using **--authorization-webhook-config-file=<path>**

Example –

clusters:

- name: authz_service

  cluster:

    certificate-authority: ca.pem

    server: https://authz.remote/

Kubernetes Authorization – RBAC

With RBAC, access to resources is regulated using roles assigned to users.

To enable RBAC, start the API server with **--authorization-mode=RBAC**

```
kind: Role
apiVersion: rbac.authorization.k8s.io/v1beta1
metadata:
  namespace: default
  name: deployment-manager
rules:
- apiGroups: [""]
  resources: ["pods"]
  verbs: ["get", "list", "watch", "create", "update", "patch", "delete"]
```

```
kind: RoleBinding
apiVersion: rbac.authorization.k8s.io/v1beta1
metadata:
  name: binding
  namespace: default
subjects:
- kind: User
  name: employee
  apiGroup: ""
roleRef:
  kind: Role
  name: deployment-manager
  apiGroup: ""
```

Admission Controllers

Admission controllers are modules that intercept requests to the API server after the request is authenticated and authorized

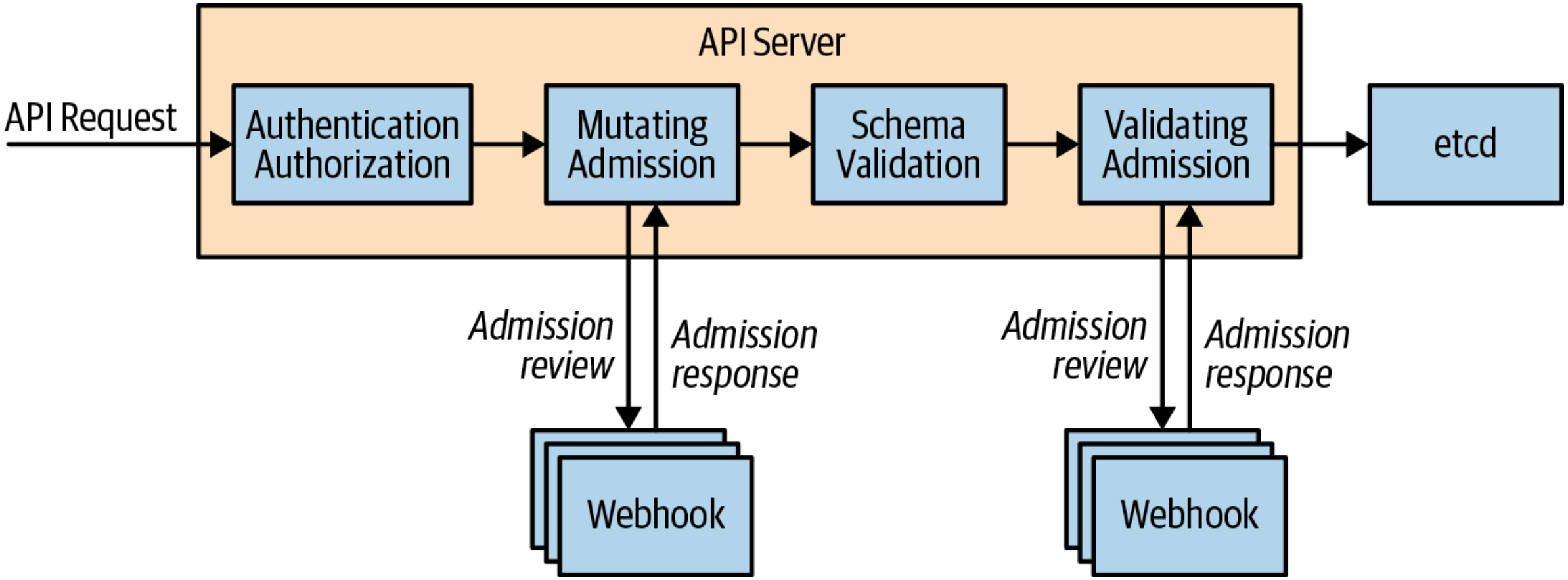
The controllers validate and mutate the request before modifying the state of the objects in the cluster. A controller can be both mutating and validating. If any of the controllers reject the request, the request is dropped immediately and an error is returned to the user so that the request will not be processed.

Admission controllers can be enabled by using the **--enable-admission-plugins** flag

Important Admission controllers:

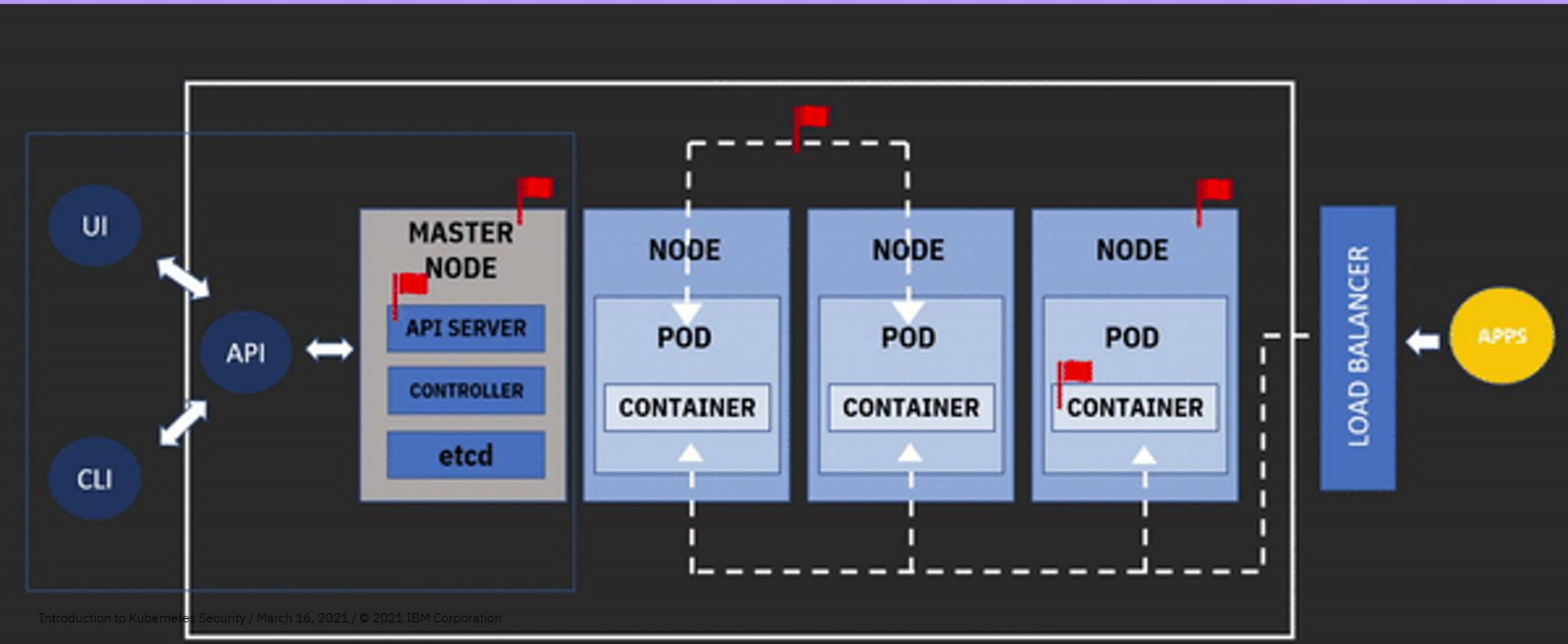
- AlwaysAdmit
- AlwaysPullImages
- EventRateLimit
- LimitRanger
- NodeRestriction
- PersistentVolumeClaimResize
- PodSecurityPolicy
- SecurityContextDeny
- ServiceAccount

Admission Controlled & Authorization in Action

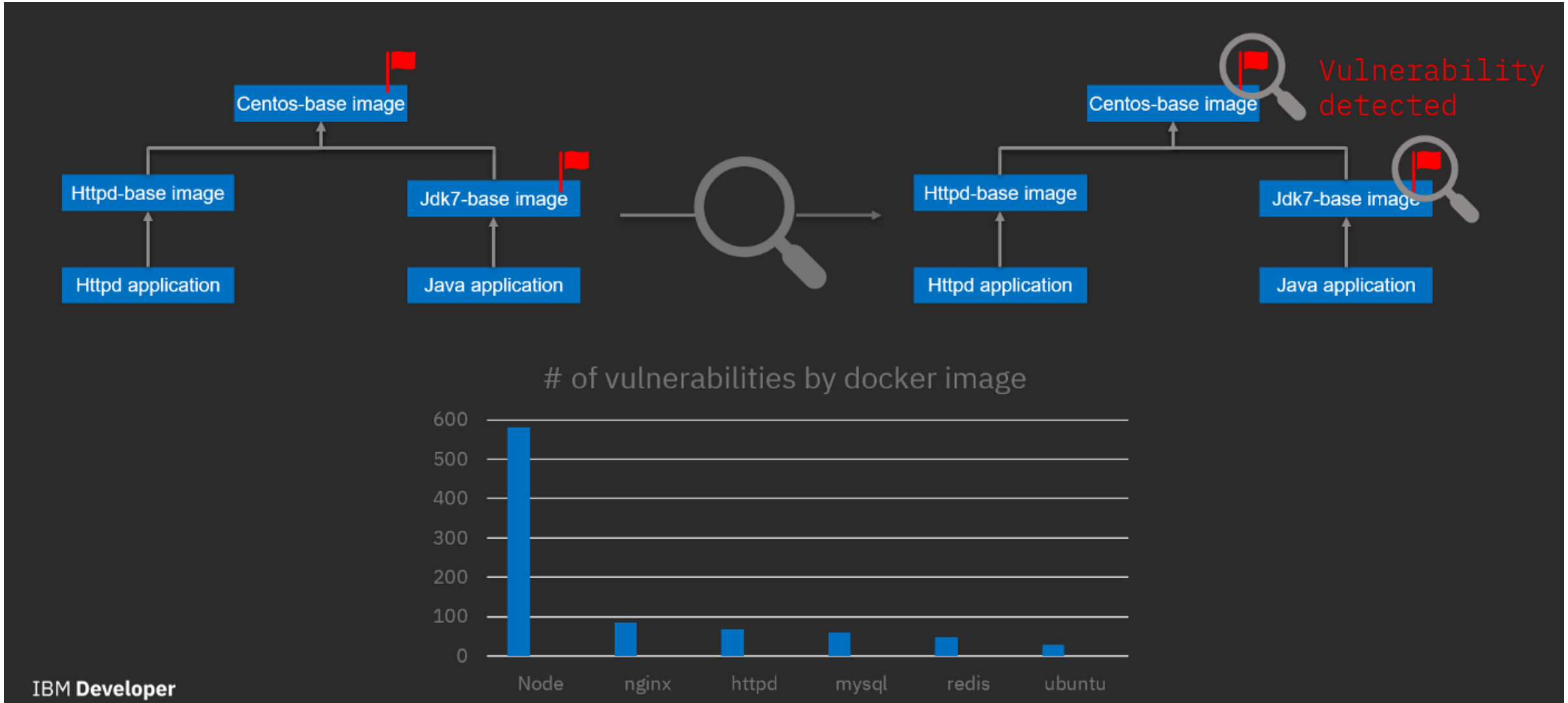


https://www.oreilly.com/library/view/kubernetes-best-practices/9781492056461/ch17.html

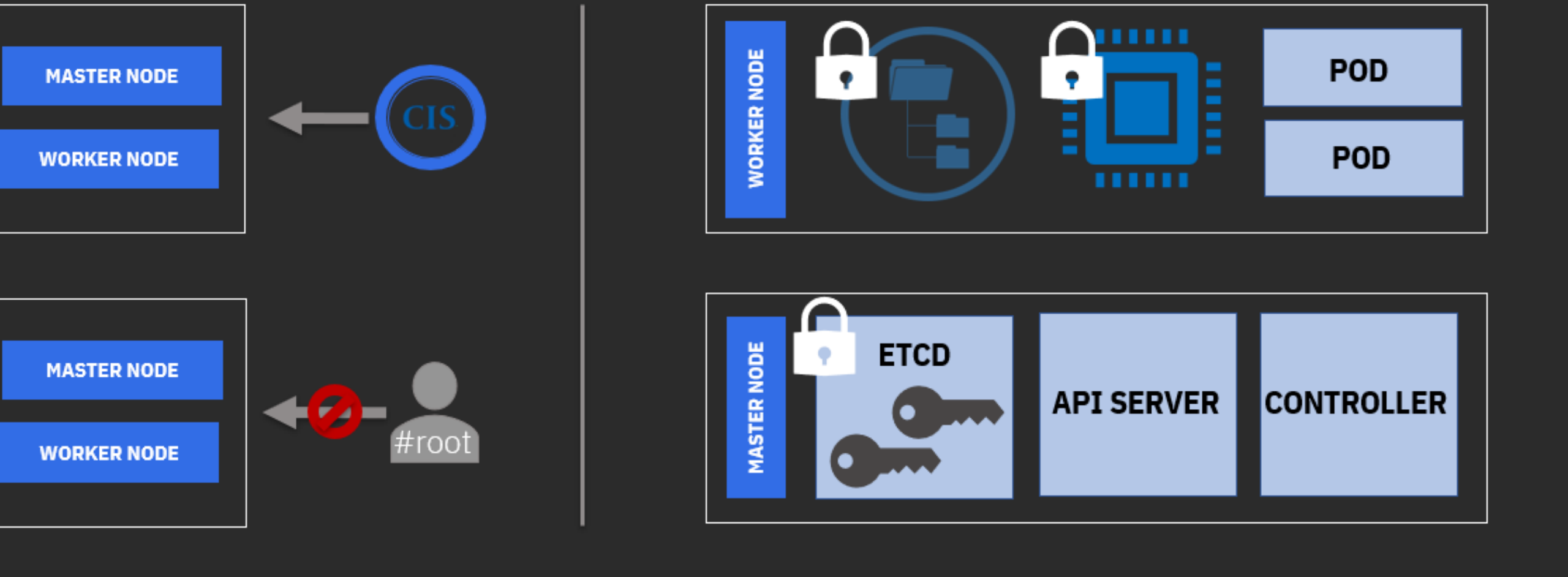
attack vectors of the Kubernetes architecture



Attack vectors of the Kubernetes architecture: Container image security

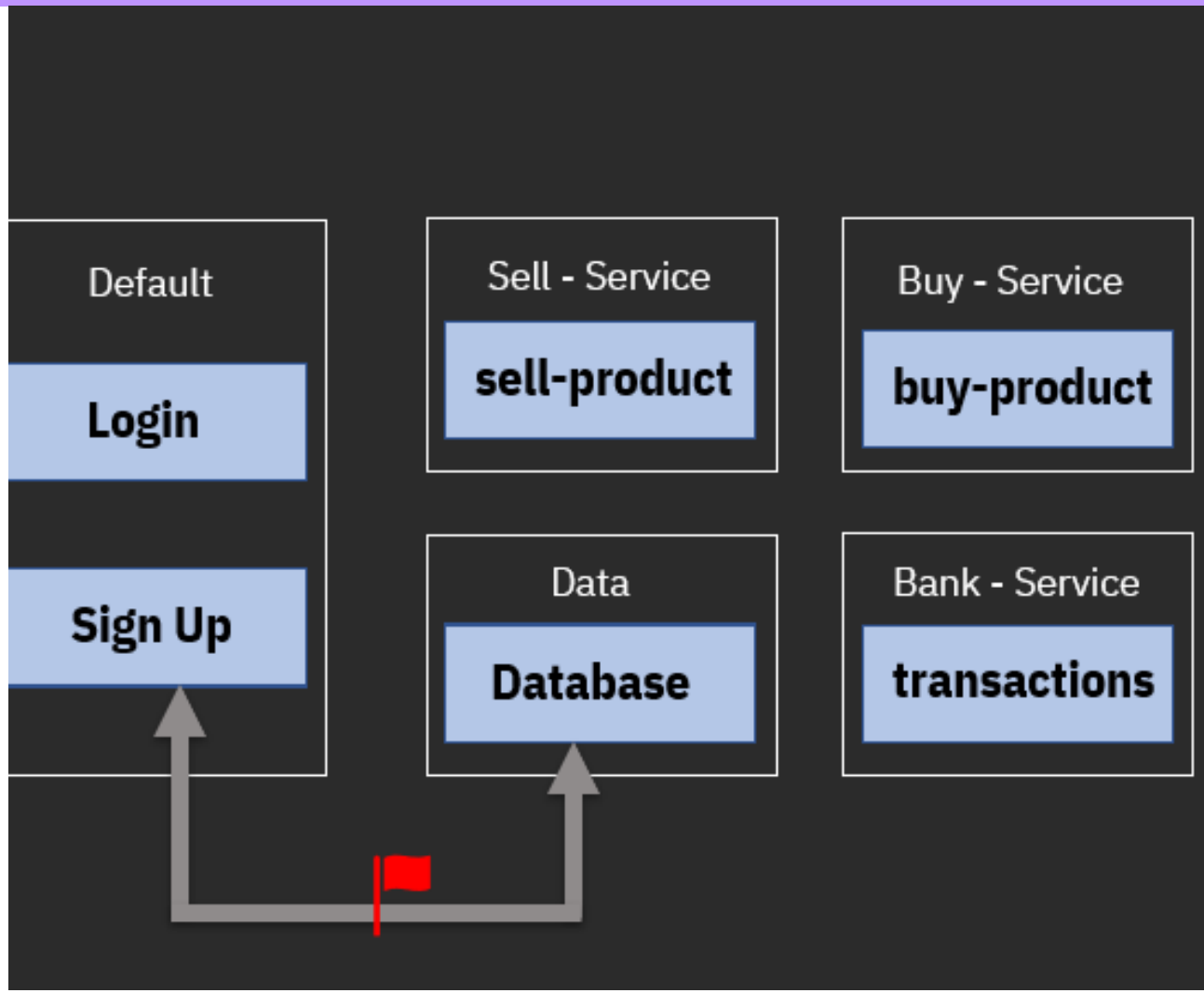


Attack vectors of the Kubernetes architecture: Master and worker node security



Attack vectors of the Kubernetes architecture: API server security

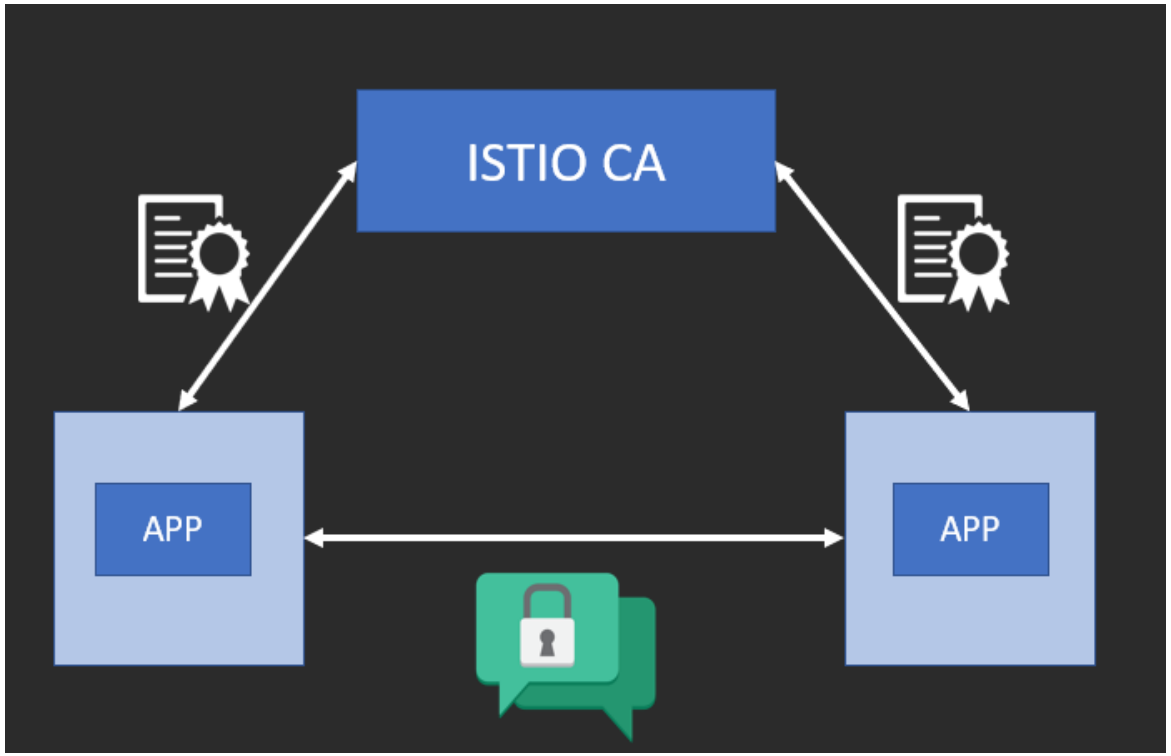
Attack vectors of the Kubernetes architecture: Network security



ID	FROM	TO	PORT	ACTION
1	12.12.12.12	5.6.2.4	443	Allow
2	9.3.4.5	66.3.4.3	80	Decline

ID	FROM	TO	PORT	ACTION
1	login	buy-product	443	Allow
2	login	database	3306	Decline

Attack vectors of the Kubernetes architecture: Network security



Istio is a service mesh that implements certifications and encryption for pod-to-pod communication using mutual Transport Layer Security (TLS),

- Code Lab
  https://securecube.gitbook.io/securecube/
- IBM Cloud Signup/Sign in – https://ibm.biz/kubesec

- Get your free Red Hat OpenShift Cluster at:
  URL: https://kubesec.mybluemix.net/
  Key: oslab

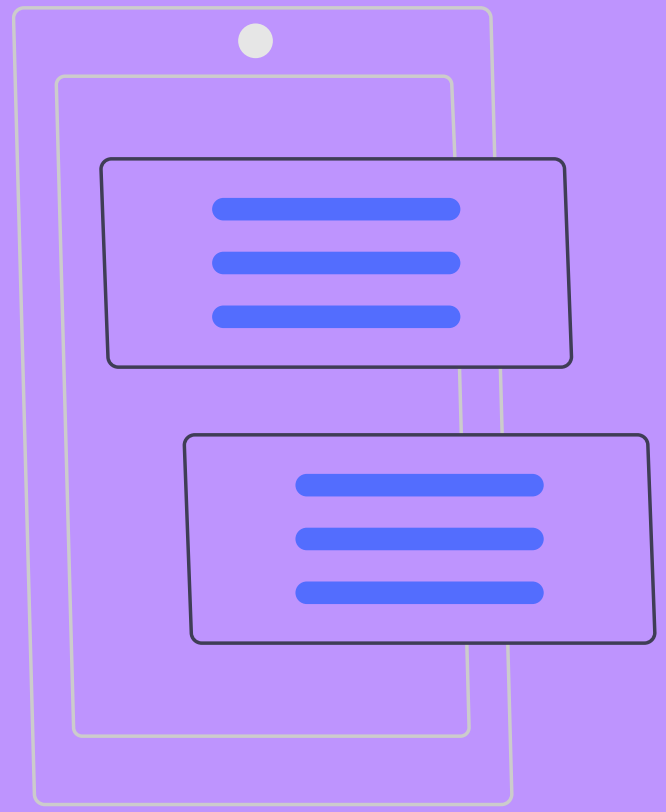
Summary

- We talked about microservices and how we can approach to Kubernetes security.
- We discussed important components of Kubernetes security in detail which includes authentication, authorization and Admission Controllers
- Different attack vectors of the Kubernetes architecture and how to avoid them; image security, master and worker node, API server and network security using Istio

- Xxx
- (image on the right is from DAM)

Survey

https://ibm.biz/kubesecsurvey



Resources

You can learn more using the following resources:

- Basics of Kubernetes Security
- A Journey to Kubernetes Security
- IBM Developer (developer.ibm.com)
- Red Hat courses (https://www.redhat.com/en/services/training/all-courses-exams)
- Meetup Page:
  https://www.meetup.com/IBM-Developer-Islamabad-Meetup/
  https://www.meetup.com/IBMKarachi/