Kubernetes and Cloud Native Associate (KCNA)

Prep Exam Note:

curiculum:

Summary:

01 Kubernetes Fundamentals:

Kubernetes Resources:

- Resource : An object of a certain type in the Kuberntes API
- You can list all available resource types in the cluster with the command kubectl api-resources.
- You can create your own custom resources using CustomResourceDefinition.
- Get documentation for a resource type using kubectl explain.
- Use an init container to run a task befor a Pod's main container start up.

Pod-management Resources

- ReplicaSet : Ensure a given number of replica pods are runiing at any given time.

- Deployment : provide declarative update for ReplicaSets and Pods. Greate for scaling stateless application. Use the default RollingUpdate deployment strategy for zero-downtime deployments.
- Imperative command to create a Deployment: kubectl create deploy --image= --replicas=
- StatefulSet : is similar to deployment, but for stateful application. Replica Pods have a sticky identity. Required to manually create a headless Service.
- DaemonSets: Dynamically runs a replica Pod on each Node, or just some Nodes.
- Job : Runs a containerized task to completion, automatically rerries if it fails.
- CronJob : Runs jobs repeatedly according to a schedule

Kuberenetes Architecture

- Worker Node : Responsible for running container workloads.
- Control Plane: Set of components that manage the cluster.
- etcd : Controle plane component / Distributed object storage used by the API Server
- Scheduler : Control plane component / Assigns new Pods to an approriate worker Node.
- Controller manager: Control plane component / Bundles serval contrllers, each of which provides some cluster functionality.
- Cloud Controller Manager: Control plane component/ Bundles controllers that interact with cloud providers APIs.
- kube-proxy: WorkerNode component / manages local routing rules on the Node to route network traffic to Pods.
- Kubelet: WorkerNode component / kubernetes agent that works with the contaienr runtime to run containers on the Node.

- Container Runtime: WorkerNode component / software that runs containers, such as containerd or CRI-O
- Kubernetes CRI : standard interface for container runtimes. any runtime that implements CRI should work with kubernetes.

Kubernetes API

- Kubernetes API: HTTP interface for kubernetes. Users, kubernetes components, and external components use it to communicate.
- The API uses REST or RESTful objects to represent and change state.
- Formats like YAML and JSON can be used yo communicate with the API, but they are ultimately considered sim;ply REST objects when persisted/stored.

Containers

- Linux control Groups are used to provide container isolation.
- You can have more than one container per Pod.
- A Dockerfile is a text file that contains commands used to build an image.

Scheduling

- Scheduling is the process of assigning a Pod to a Node.
- Scheduling occurs when a new Pod is created and has not yet been scheduled.
- Scheduler takes into account things like resource requierments, Pod affinity, and taint/tolerations when selecting a Node.

02 Container Orchestration:

Orchestration Fundamentals

- Orchetration: means using automation to take on the work of running and managing containerized workloads.
- Orchestration includes things like assigning containers to a server and spinning them up on that server, or restarting a broken container.

Container Runtime

- Container Runtime: A piece of software responsible for actually running containers.
- Container Runtime Interface (CRI) - Standard protocol for communication between kubelet and the container runtime.
- CRI-O and Containerd - Two examples of container runtimes that support the CRI standard and therefore work with kubernetes.

Kubernetes Securiy

- 4Cs of Cloud Native Security: Cloud,Clusters,Containers and Code
- Client Certificates: API Server uses a signed X509 client certificate to authenticate a user.
- OpenID Connect - Uses a JSON Web Token (JWT) signed by an external identity provider to authenticate a user.
- OPA GateKeeper - You can create policies to limit what can be done in your cluster. GateKeeper validates incomming requiests to the API according to your policies.
- The OPA GateKeeper tool exists outside of Kubernetes and can be used in other contexts.

Kubernetes Networking

- Kubernetes uses a virtual cluster network to allow containers to communicate transparentlu regardless of which Node they are on.
- The ckuster Domain Name Server (DNS) allows containers to discover Services by hostname.
- Network policies control what network traffic is allowed in the cluster network and determine whether Pods are isolated or non-isolated.
- Pods are non-isolated by defaut. All network traffic is allowed.
- If any Network Policy selects a Pod, the Pod is isolated. Only traffic allowed by any Network Policy that selects the Pod is allowed.
- Isolation is treated seprately for incomming (ingress) traffic and outgoing (egress) traffic.

Services

- Services: expose an application running on a set of replica Pods as a network service.
- Service Type OF ClusterIP: Expose internally within the cluster network.
- Service Type OF NodePort: Expose externally in a port on each Node.
- Service Tyype OF LoadBalancer: Expose using a cloud provider's load balancer.
- Service Tyype OF ExternalName: Provide a DNS name for an External service.
- Service Tyype OF Headless Service: A service with no cluster IP address.
- A service without a selector requires any endpoints to be manually created
- There are two main service discovery methods in Kubernetes: DNS and environment variables.

- An ingress exposes an application externally and routes traffic to a Service. It can also provide additional functionality like SSL termination.

Service MESH

- Service Mesh: a tool that manages communication between application components, often adding additional functionality like logging, tracing, or encryption.
- Two main components of a service mesh are the control plane and service proxy/data plane.
- SideCar: an additional container running in a Pod alongside the main container.
- Service Mesh interface (SMI): A standard interface for managing kubernetes sservice meshes that support the standard.

storage

- Volume: Provide external storage to your kubernetes containets to store application data.
- PersistentVolume: Define a dynamically-consumable storage resource.
- PersistentVolumeClaim: Binds to a PersistentVolume and allows you to mount the storage resrouce in a Pod.
- Rook: automates storage management with self-managing,self-scaling,self-healing storage services.
- PersistentVolune reclaime policies:
-
  - Retain: Reclaim manually.
-
  - Recycle: automatic reclamation via a simple data scrub
-
  - Delete: Underlying storage resource is deleted.

- Use immutable: true with ConfigMap and Secret to mark the data as unchangeable.
- By default, data stored in Secrets is no encrypted, just base64-encoded.

03 Cloud Native Architecture:

Cloud Native Architecture

- Cloud native technology is important because it removes roadblocks to innovation.

Autoscalling

- Vertical scaling: Adds resources to existing apps and servers.
- Horizontal Scaling: Adds additional replicas of apps and servers.
- Horizontal Pod Autoscaler: Adds/removes replica Pods based on real-time usage data.
- Cluster Autoscaler: Adds/removes cluster Nodes based on real-time usage data.

Serverless

- Serverless: A framework that allows developers to write and run code without worrying about things like servers,scaling,and operating system.

Community and Governance

- CNCF : The cloud native computing fundation.
- The CNCF: makes decisions through public discussion and voting.

Organizational Personas

- Organizational Personas are not necesserily singular individuals or job positions. They are roles that describe the work of managing cloud native applications.
- Site Relicability Engineets (SRE): Responsible for creating and maintaining SLOs, SLAs and SLIs.

Open Standards

- Open Container Initiative (OCI) - Organization that creates open standards for containers, such as image formats and runtimes.
- runc: is the reference implemetation for the OCI runtime-spec standard.

04 Cloud Native Observability

Telemetry and Observability

- Command to get container logs: kubectl logs pod_name container_name
- Trace: a set of related events across multiple components.
- Span : Data about a single component within a trace.

Prometheus Monitoring

- Prometheus: is a tool that allows you to collect metric data for applications and systems.
- Counter: a prometheus metric type; a single number that can only increase or be reset to zero.
- Gauge: a prometheus metric type; a single number that can both increase and decrease.

- Grafana: a tool that can be used to build useful visualization of Prometheus data.

Cost Management

- FinOps: Using observabilty to support automation and data-driven decisions to limit cloud costs.
- Tools like cluster autoscaler can help manage costs by scaling up the cluster only when needed,such as during a large batch job.

05 Cloud Native Application Delivery

Application Delivery Fundamentals

- Cloud native techniques and technologies support rapid innovation and reliability when delivering applications.

GitOps

- GitOps tools like Flux and ArgoCD monitor a Gir repository and apply changes to the cluster based upon what is in that repository.
- Flux is build on top of the GitOps Toolkit standard library.
- Flux and ArgoCD are both writen in Go.

CI/CD

- CI/CD: continuous integration and continuous delivery/deployment
- CI/CD is continuous because it includes things like frequent deployments, incremental changes, automated processes, and reliable rollbacks.