ECE 1779 Assignment2

Junhong Chen

1

Google implemented the Borg system to manage clusters of servers/cells, identify the basic three functionalities that Borg accomplishes. Explain how separation of duties/isolation is accomplished in Borg. Identify the main components of Borg and describe how these components define its architecture.

Resource Management: Borg efficiently allocates and manages computing resources (such as CPU, memory, and storage) across a large number of servers in a data center. It ensures that applications receive the necessary resources to run effectively while maximizing resource utilization across the cluster.

Job Scheduling: Borg schedules tasks and jobs submitted by users or applications onto the available resources in the cluster. It considers factors such as resource requirements, constraints, priorities, and dependencies to optimize the allocation and execution of jobs, ensuring efficient resource utilization and meeting performance objectives.

Fault Tolerance and Reliability: Borg provides mechanisms for handling failures and maintaining service availability in the face of hardware failures, network issues, or other types of disruptions. It automatically detects and responds to failures by rescheduling tasks, reallocating resources, and maintaining service redundancy to minimize downtime and ensure high availability.

Borg achieves separation of duties and isolation through several mechanisms:
Containerization: Borg uses lightweight containerization to isolate individual tasks or jobs from each other. Each task runs within its own container, providing process-level isolation and ensuring that tasks cannot interfere with or access resources belonging to other tasks.

Resource Quotas and Limits: Borg enforces resource quotas and limits for each job or task, restricting the amount of CPU, memory, and other resources that a task can consume. This prevents individual tasks from monopolizing resources and ensures fair resource allocation across multiple users and applications.

Security Policies: Borg implements security policies and access controls to restrict access to sensitive resources and data. It provides authentication and authorization mechanisms to ensure that only authorized users and applications can access and manipulate resources within the cluster.

Main component and how they define Borg's architecture:
Borgmaster: it is logically a single process but is replicated 5 times to prevent the problem of single failure. Its job is to handle all operations and change the cell's state, such as submitting a job or terminating a task on a machine. If there is an outage, it can recover and re-synchronize its state from other replicas that are up-to-date.
Scheduler: When a job is submitted, the Borgmaster will record it to the **Paxos store** persistently and add the job's tasks to the pending queue. This queue will be scanned asynchronously by a scheduler which is responsible to assign these tasks to difference machines. The scheduling algorithm has two parts: to find machines where the task can run, and to pick one of these feasible machines by scoring.
Borglet: It is a local Borg agent on every machine in a cell. It is responsible for starting and stopping tasks, managing local resources to run the tasks and reporting the state of the machine to the Borgmaster and other monitoring system. Each Borgmaster replica runs a stateless link shard to communicate with some of the Borglets.

2.

Kubernetes is a portable, extensible, open source platform for managing containerized workloads and services, that facilitates both declarative configuration and automation. It has a large, rapidly growing ecosystem. Kubernetes services, support, and tools are widely available. - The name Kubernetes originates from Greek, meaning helmsman or pilot. K8s as an abbreviation results from counting the eight letters between the "K" and the "s". Google open-sourced the Kubernetes project in 2014. Kubernetes combines over 15 years of Google's experience running production workloads at scale with best-of-breed ideas and practices from the community.

2.1 Identify when would you deploy Kubernetes.

I would deploy Kubernetes if I have a complex application consisting of multiple microservices or containers that need to be deployed, managed, maintained, and scaled efficiently.

2.2 Identify the main components of Kubernetes.

Master components:

API Server: It exposes the Kubernetes API and serves as the front end for the Kubernetes control plane. It validates and processes REST operations, and it's the primary interaction point for managing Kubernetes objects.

Controller Manager: A set of controllers that manage various aspects of the cluster's state, such as node management, replication control, endpoints, and

namespace lifecycle. When node is down, controller manager will detect this failure and conduct the recovery process.

Scheduler: The component responsible for scheduling pods (groups of containers) onto nodes in the cluster. It considers factors like resource requirements, node capacity, and scheduling constraints when making scheduling decisions.

etcd: A distributed key-value store that serves as the **cluster's persistent storage** for all cluster data, including configuration settings, cluster state, and metadata.

Node components:

Kubelet: The primary node agent responsible for managing pods on the node. It ensures that containers are running in the pods, monitors their health, and reports back to the master.

Kube-proxy: A network proxy that runs on each node and maintains network rules required to implement Kubernetes services, such as load balancing and service discovery.

Container Runtime: The software responsible for running containers on the node, such as *Docker*, containerd, or cri-o.

3.

m. From the YAML file, observer the number of replicas; how many replicas where specified?

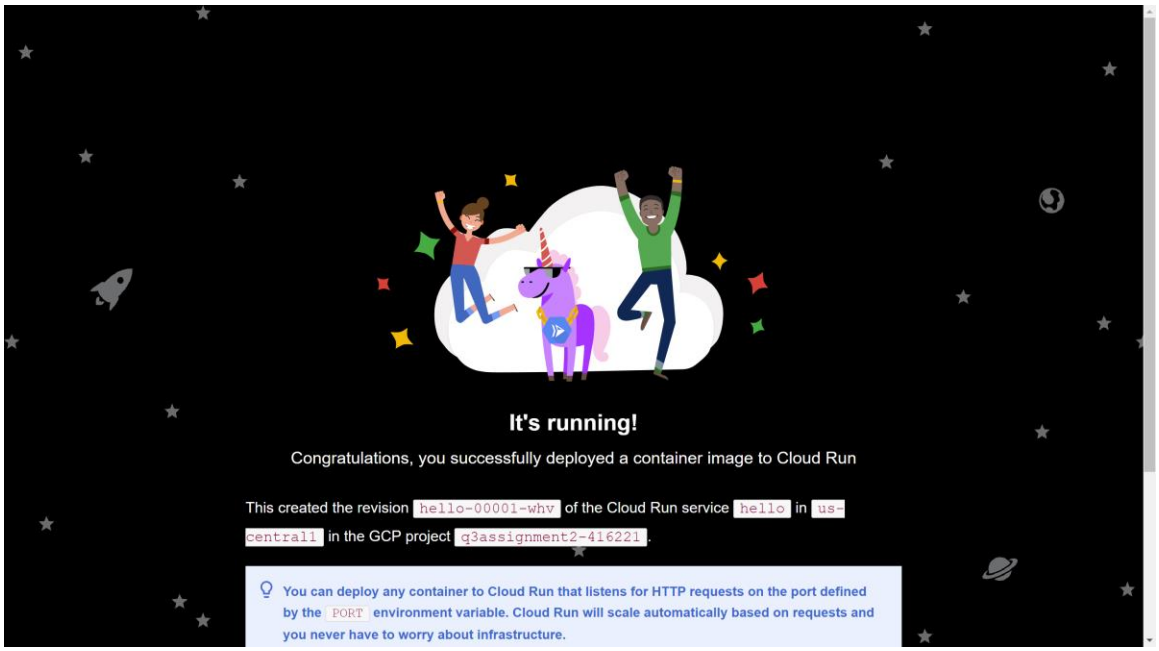
There is 1 replica.

n. What is the URL specified for consuming the service.

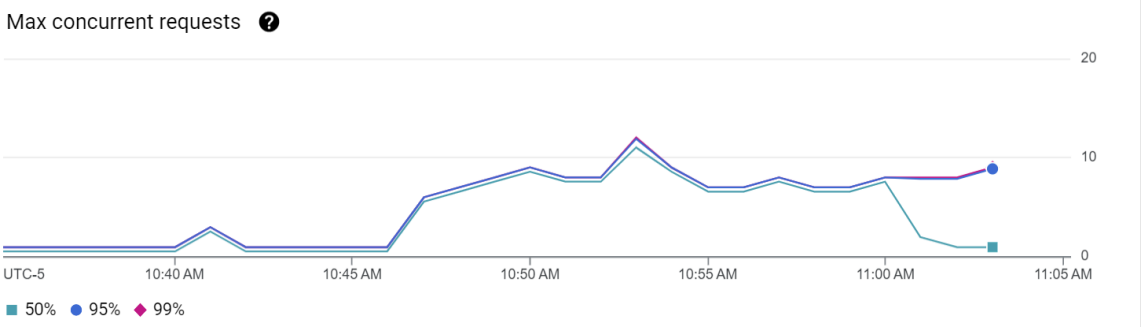
https://hello-lk7ohbvv6a-uc.a.run.app

o. What is the result of parsing the URL in the browser.

This page shows up, indicating that the container is running on the server.

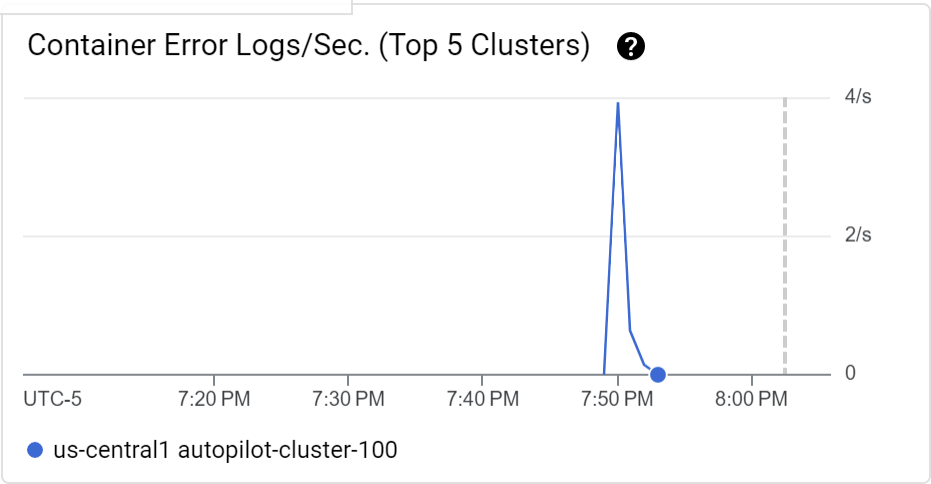


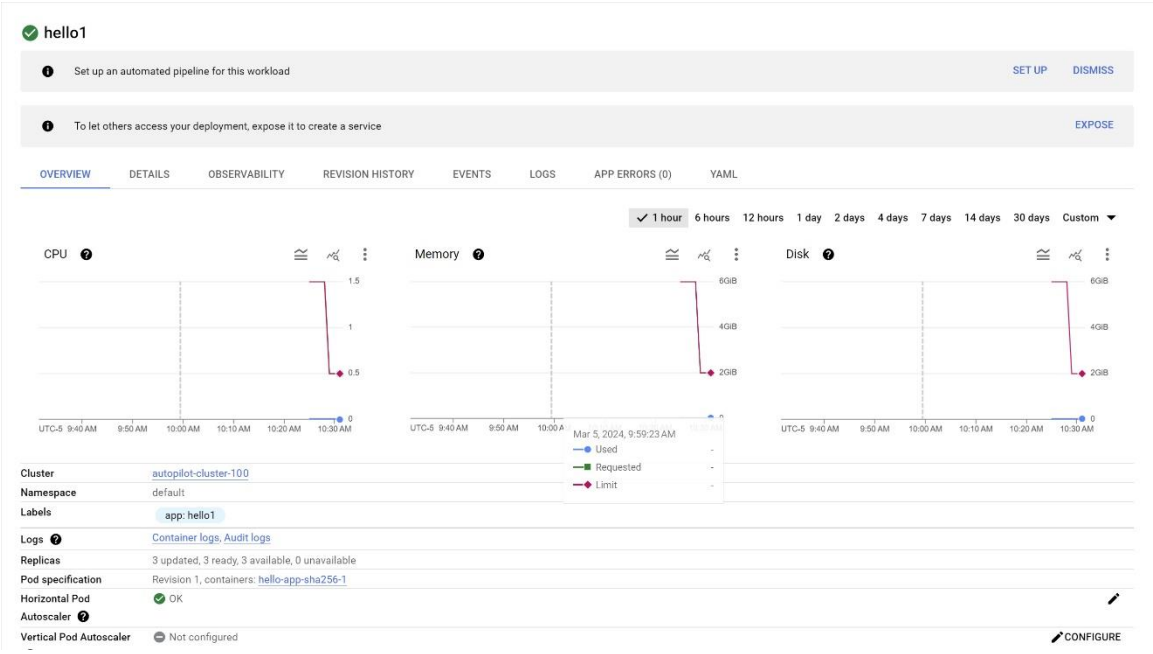
r. What is the analytical summary output of the Container Instance Count and container CPU Utilization?

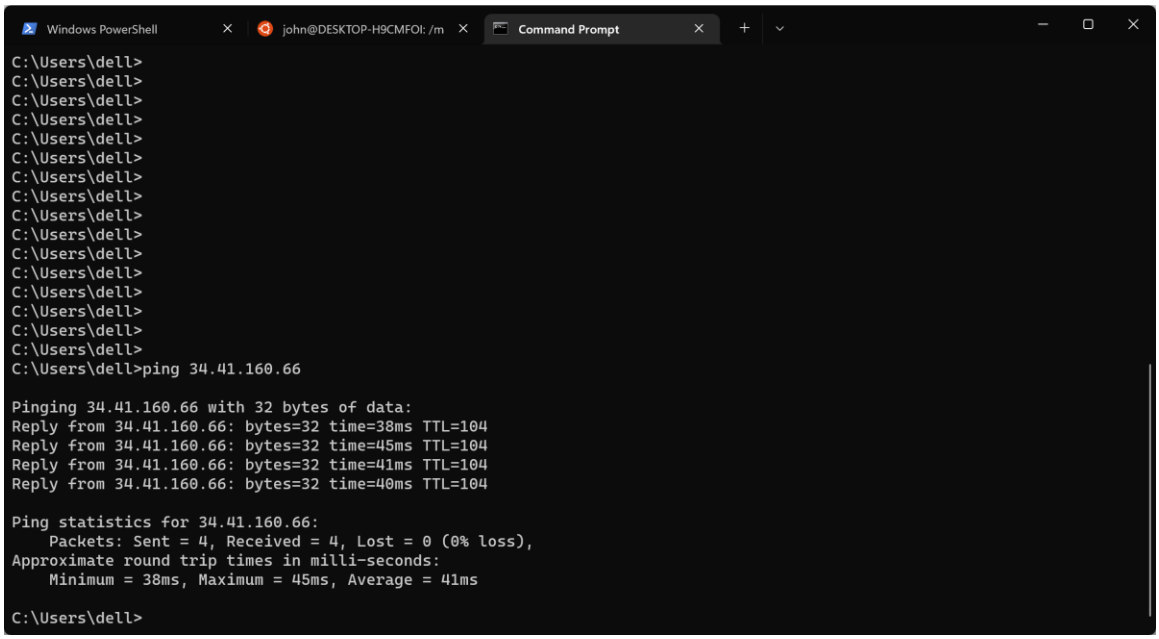


4.

b.Capture your version of the Container Error Logs/Second chart and paste in your report.






Windows PowerShell
john@DESKTOP-H9CMFOI: /m
Command Prompt
C:\Users\dell>
C:\Users\dell>
C:\Users\dell>
C:\Users\dell>
C:\Users\dell>
C:\Users\dell>
C:\Users\dell>
C:\Users\dell>
C:\Users\dell>
C:\Users\dell>
C:\Users\dell>
C:\Users\dell>
C:\Users\dell>
C:\Users\dell>
C:\Users\dell>
C:\Users\dell>
C:\Users\dell>ping 34.41.160.66
Pinging 34.41.160.66 with 32 bytes of data:
Reply from 34.41.160.66: bytes=32 time=38ms TTL=104
Reply from 34.41.160.66: bytes=32 time=45ms TTL=104
Reply from 34.41.160.66: bytes=32 time=41ms TTL=104
Reply from 34.41.160.66: bytes=32 time=40ms TTL=104
Ping statistics for 34.41.160.66:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 38ms, Maximum = 45ms, Average = 41ms
C:\Users\dell>


requests:
cpu: 500m
ephemeral-storage: 1Gi
memory: 2Gi
securityContext:
capabilities:
drop:
- NET_RAW
terminationMessagePath: /dev/termination-log
terminationMessagePolicy: File
dnsPolicy: ClusterFirst
restartPolicy: Always
schedulerName: default-scheduler
securityContext:
seccompProfile:
type: RuntimeDefault
terminationGracePeriodSeconds: 30
tolerations:
- effect: NoSchedule
key: kubernetes.io/arch
operator: Equal
value: amd64
status:
availableReplicas: 1
conditions:
- lastTransitionTime: "2024-03-05T15:24:12Z"
lastUpdateTime: "2024-03-05T15:24:12Z"
message: Deployment has minimum availability.
reason: MinimumReplicasAvailable
status: "True"
type: Available
- lastTransitionTime: "2024-03-05T15:22:28Z"
lastUpdateTime: "2024-03-05T15:24:12Z"
message: ReplicaSet "hello1-89d8cbd" has successfully progressed.
reason: NewReplicaSetAvailable
status: "True"
type: Progressing
observedGeneration: 2
readyReplicas: 1
replicas: 1