

# 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 different 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, observe the number of replicas; how many replicas were specified?

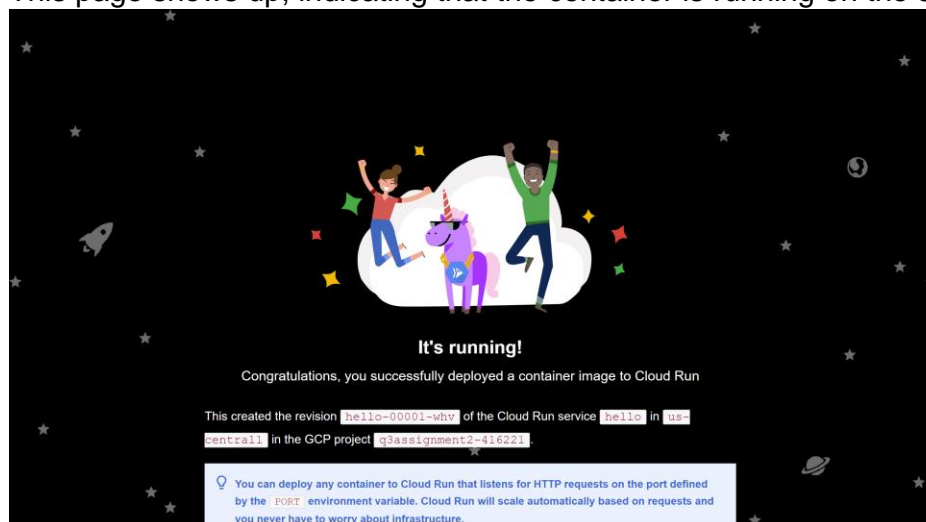
There is 1 replica.

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

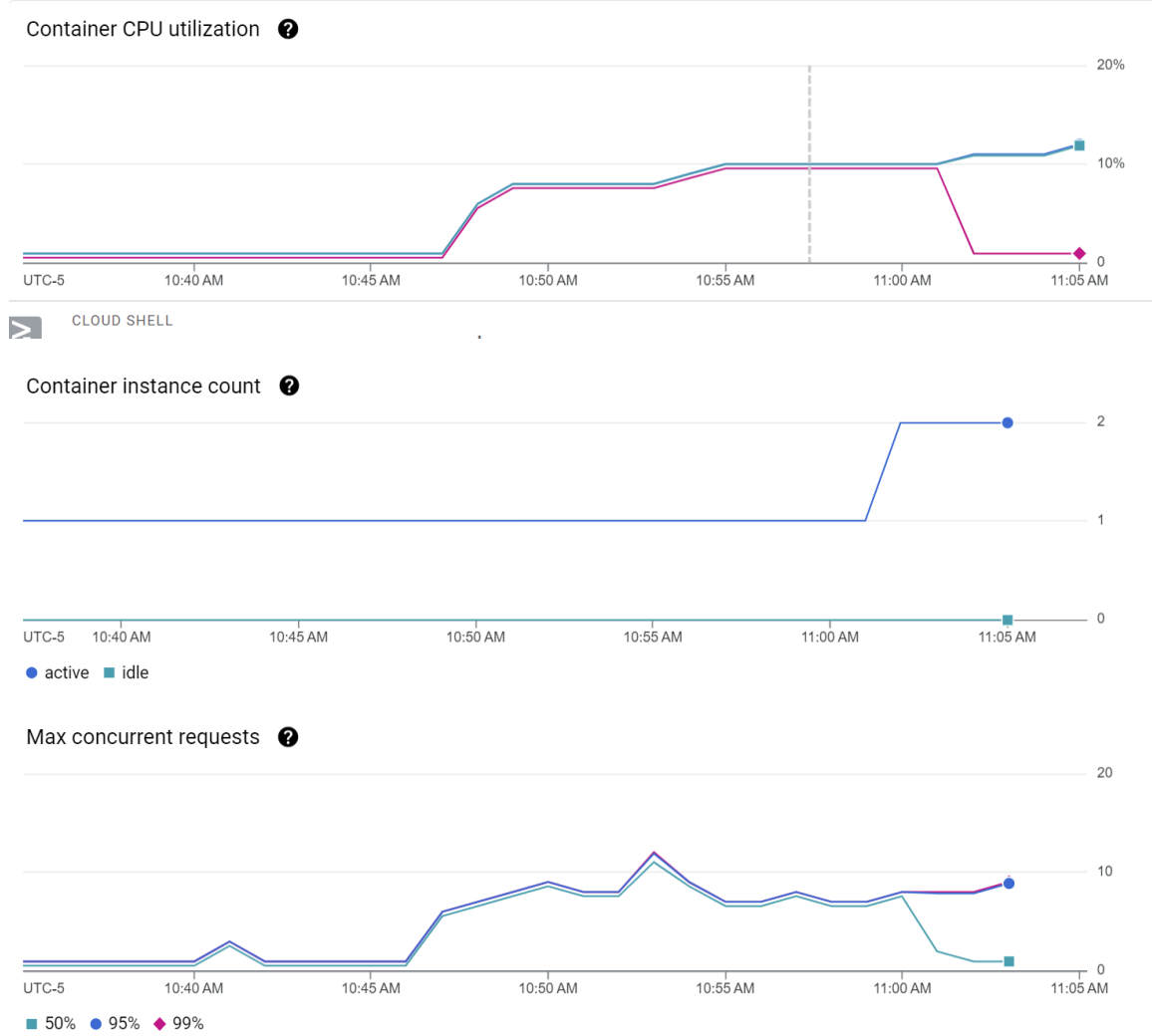
<https://hello-lk7ohbvv6a-uc.a.run.app>

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.



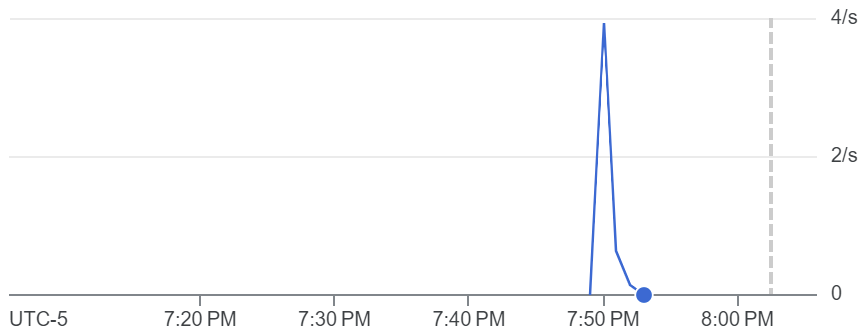
r. What is the analytical summary output of the Container Instance Count and container CPU Utilization?



4.

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

## Container Error Logs/Sec. (Top 5 Clusters) ?



● us-central1 autopilot-cluster-100

### hello1

1 Set up an automated pipeline for this workload

SET UP DISMISS

2 To let others access your deployment, expose it to create a service

EXPOSE

#### OVERVIEW

#### DETAILS

#### OBSERVABILITY

#### REVISION HISTORY

#### EVENTS

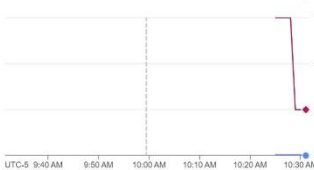
#### LOGS

#### APP ERRORS (0)

#### YAML

✓ 1 hour 6 hours 12 hours 1 day 2 days 4 days 7 days 14 days 30 days Custom

#### CPU ?



#### Memory ?



#### Disk ?



Cluster autopilot-cluster-100  
Namespace default  
Labels app: hello1  
Logs Container logs, Audit logs  
Replicas 3 updated, 3 ready, 3 available, 0 unavailable  
Pod specification Revision 1, containers: hello-app-sha256-1  
Horizontal Pod Autoscaler OK  
Vertical Pod Autoscaler Not configured

CONFIGURE

```
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>
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>
```

### 3.6

```
chen763060065@cloudshell:~ (q3assignment2-416221)$ gcloud artifacts repositories list
Listing items under project q3assignment2-416221, across all locations.

ARTIFACT_REGISTRY

REPOSITORY: hello-repo
FORMAT: DOCKER
MODE: STANDARD_REPOSITORY
DESCRIPTION: Docker repository
LOCATION: us-west1
LABELS:
ENCRYPTION: Google-managed key
CREATE_TIME: 2024-03-05T15:01:48
UPDATE_TIME: 2024-03-05T15:20:53
SIZE (MB): 12.193
chen763060065@cloudshell:~ (q3assignment2-416221)$
```

d. Refer to the YAML deployment configuration file, what are the minimum and maximum replicas values?

```
spec:
  scaleTargetRef:
    kind: "Deployment"
    name: "hello1"
    apiVersion: "apps/v1"
  minReplicas: 1
  maxReplicas: 5
  metrics:
```

f.

UTC-48:00 PM8:10 PM8:20 PM8:30 PM8:40 PM0

UTC-48:00 PM8:10 PM8:20 PM8:30 PM8:40 PM0

UTC-48:00 PM8:10 PM8:20 PM8:30 PM8:40 PM0

Cluster

autopilot-cluster-101

Namespace

default

Labels

app: hello1

Logs

Container logs, Audit logs

Replicas

1 updated, 1 ready, 1 available, 0 unavailable

Pod specification

Revision 1, containers: [hello-app-sha256-1](#)

Horizontal Pod Autoscaler

OK

Vertical Pod Autoscaler

Not configured

CONFIGURE

Active revisions

Revision	Name	Status	Summary	Created on	Pods running/Pods total
1	<a href="#">hello1-89d8cbd</a>	OK	hello-app-sha256-1: us-west1-docker.pkg.dev/q3assignment2-416221/hello-repo/hello-app@sha256:1362056cfaae989fd9439ca383b75935bd2b46b16f0489a43a4fa28f9535cec4	Mar 5, 2024, 10:22:28 AM	1/1

Managed pods

Revision	Name	Status	Restarts	Created on
1	<a href="#">hello1-89d8cbd-vmmnr</a>	Running	0	Mar 5, 2024, 10:22:29 AM

Exposing services

Name	Type	Endpoints
<a href="#">hello1-service</a>	Load balancer	<a href="#">34.41.160.66:80</a>