PLAYSTATION AND KUBERNETES: HOW TO SOLVE A PROBLEM LIKE REAL-TIME

Joseph Irving

LEVEL SELECT:

TUTORIAL: REAL-TIME GAMSERVERS

LEVEL 1: RUNNING IN KUBERNETES

LEVEL 2: AUTOSCALING

LEVEL 3: MULTI-REGION

LEVEL SELECT:

TUTORIAL: REAL-TIME GAMSERVERS

LEVEL 1: RUNNING IN KUBERNETES

LEVEL 2: AUTOSCALING

LEVEL 3: MULTI-REGION

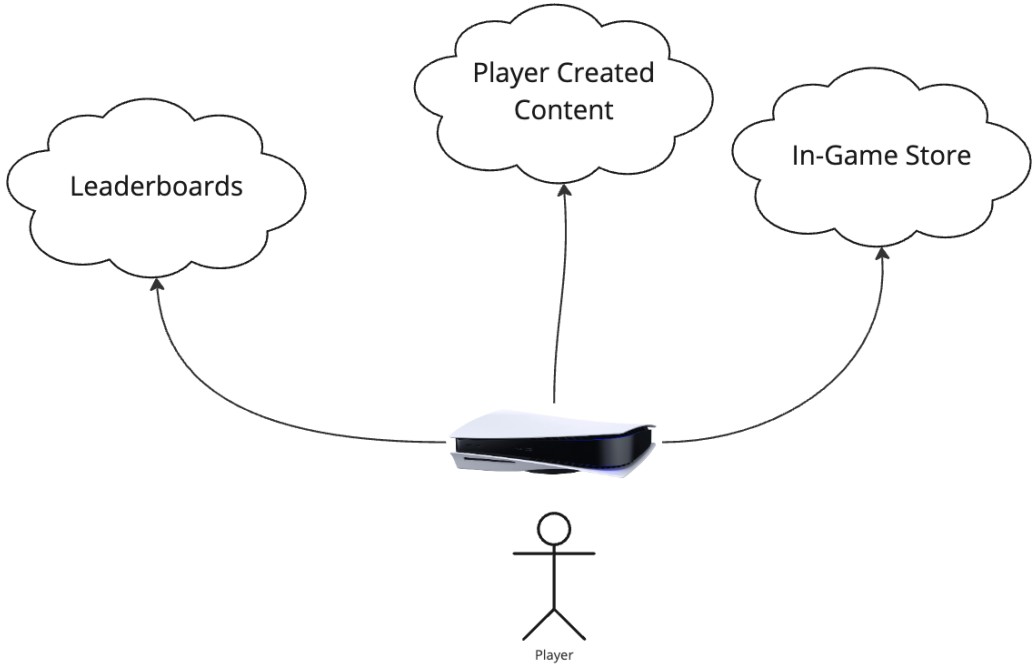
What isn’t Real-time?

Async

Request Based

HTTPS/GRPC

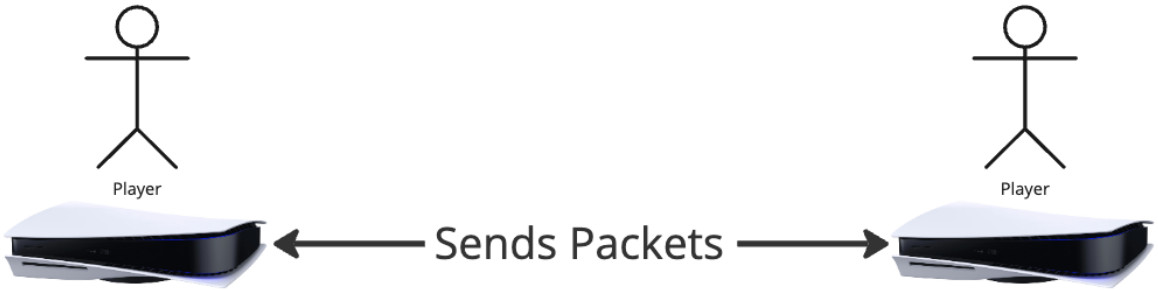
Not Latency Sensitive



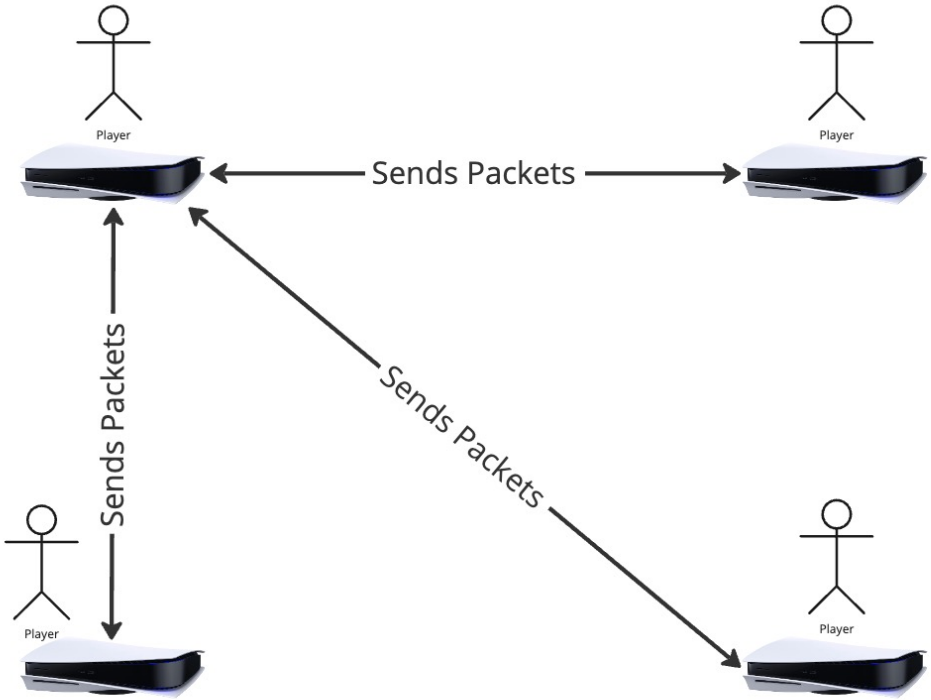
Real-time


Codelwave
<PALS FOR LIFE>
Madtoker
<PALS FOR LIFE>
Juron
<PALS FOR LIFE>
Forekin
<PALS FOR LIFE>
<AFK>Leeroy
<PALS FOR LIFE>

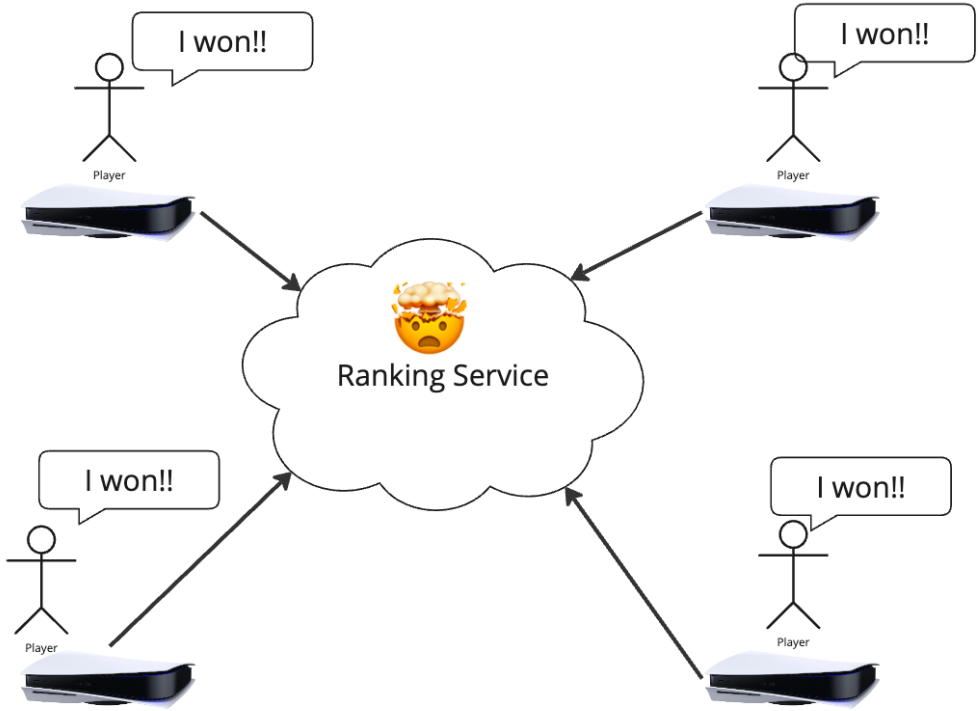
Peer to Peer



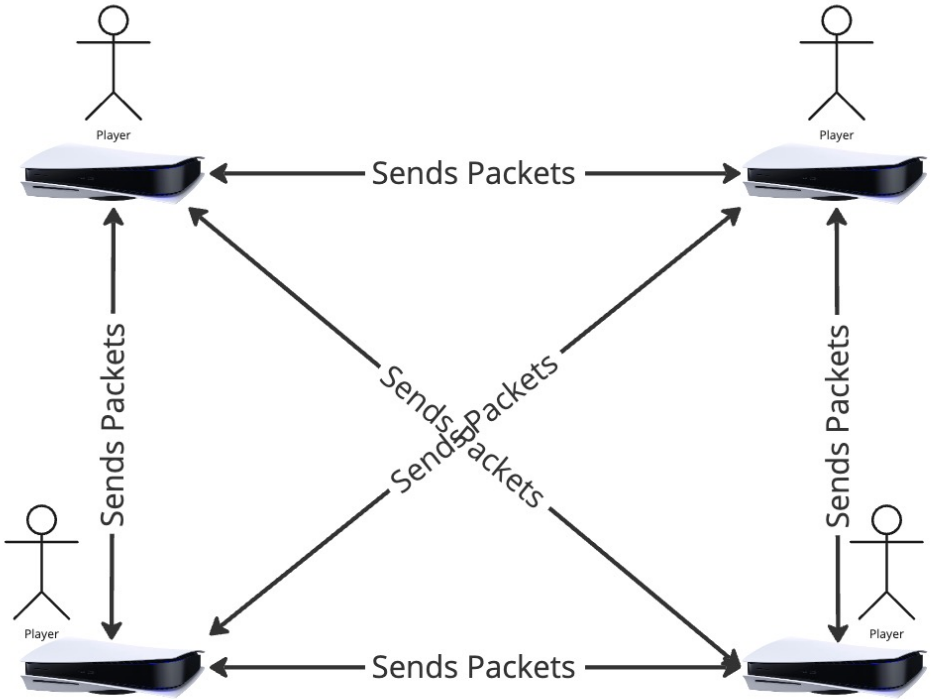
Peer to Peer


Player
Player
Sends Packets
Sends Packets
Sends Packets
Player
Player

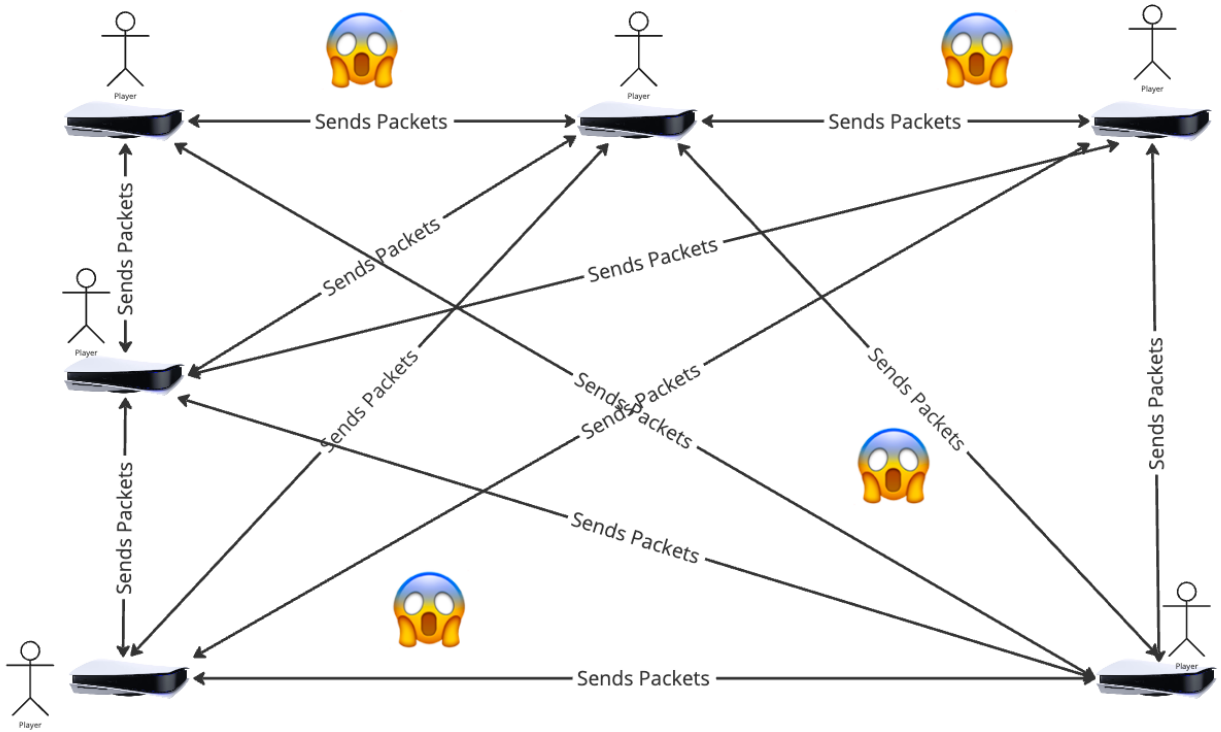
Peer to Peer


I won!!
Player
I won!!
Player
Ranking Service
I won!!
Player
I won!!
Player

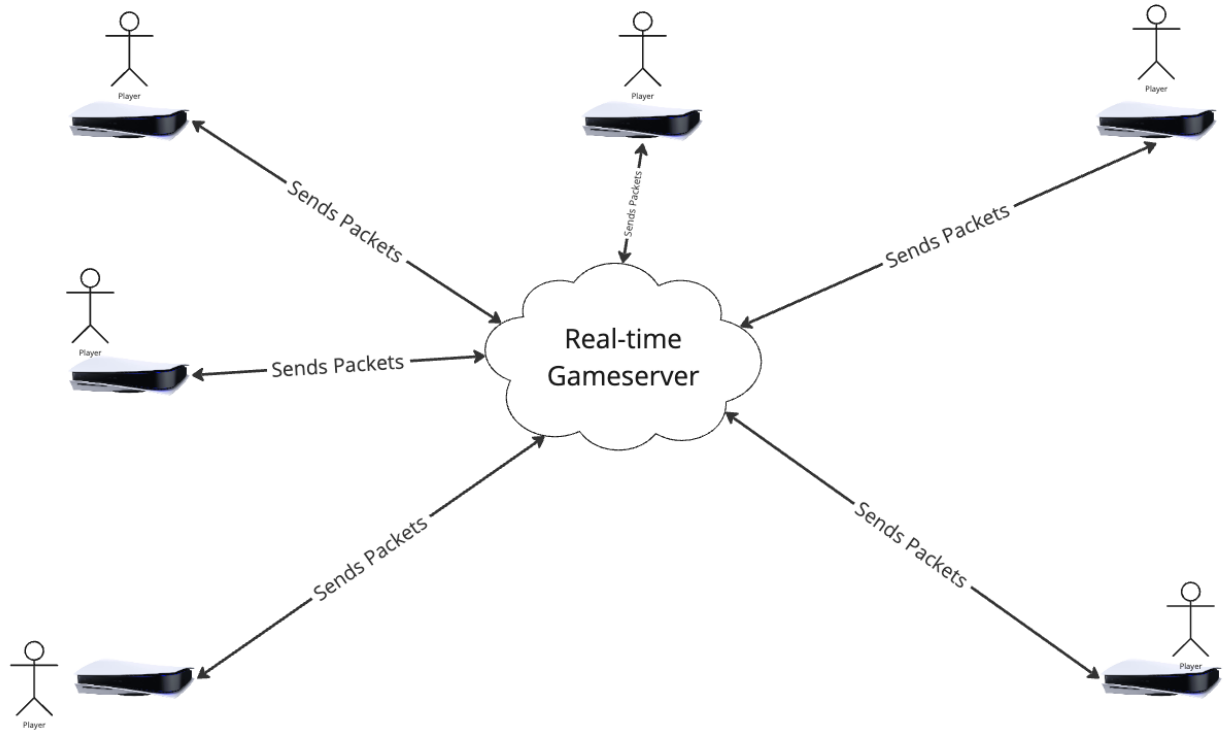
Peer to Peer


Player
Player
Sends Packets
Sends Packets
Sends Packets
Sends Packets
Sends Packets
Player
Player
Sends Packets

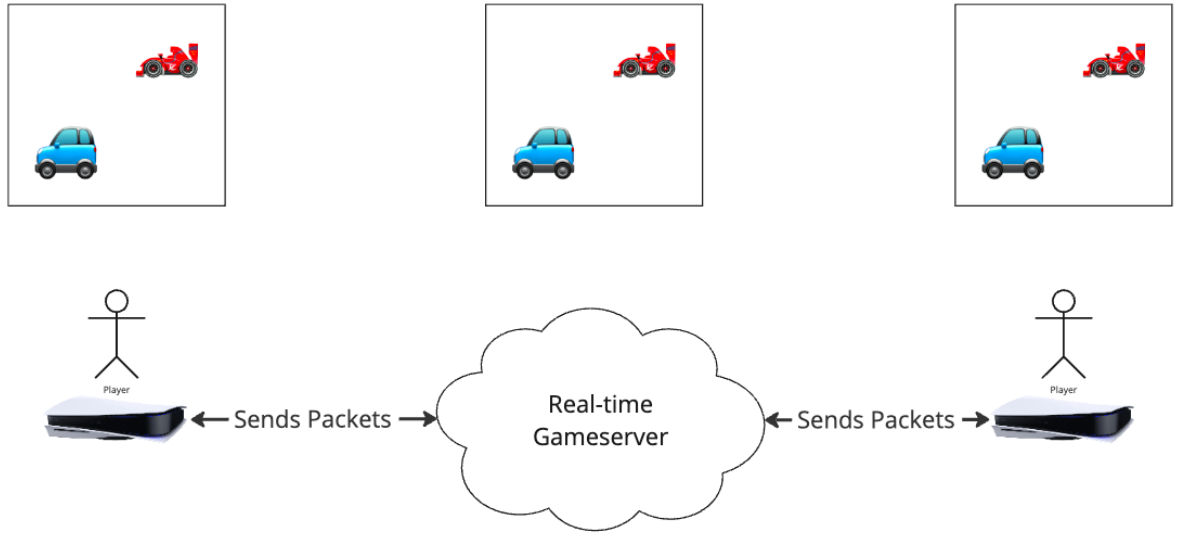
Peer to Peer


Player
Player
Player
Player
Player
Player
Sends Packets
Sends Packets
Sends Packets
Sends Packets
Sends Packets
Sends Packets
Sends Packets
Sends Packets
Sends Packets
Sends Packets
Sends Packets
Sends Packets

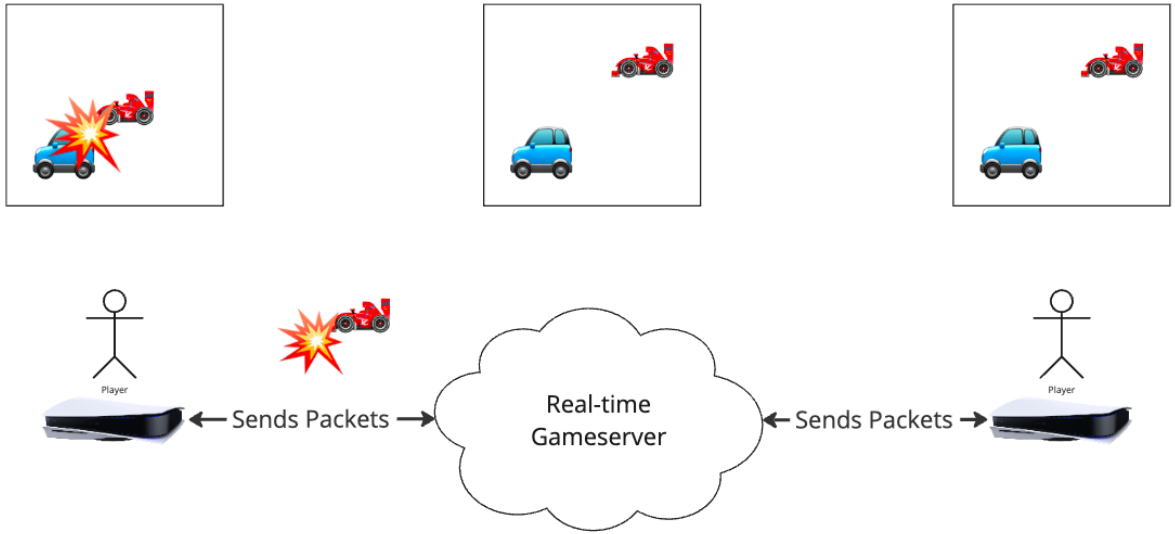
Dedicated Game Server



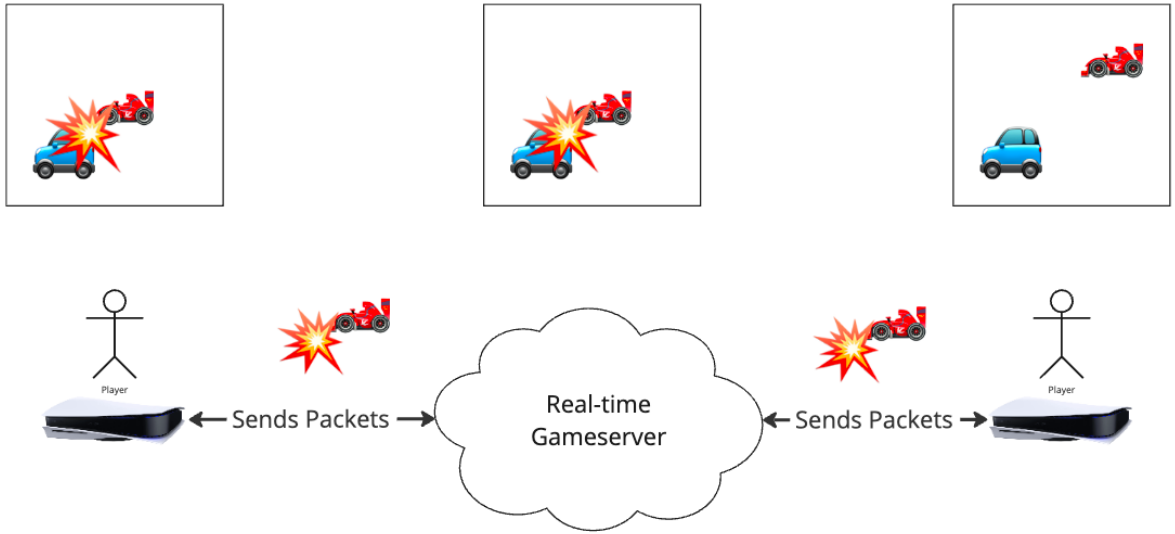
Dedicated Game Server



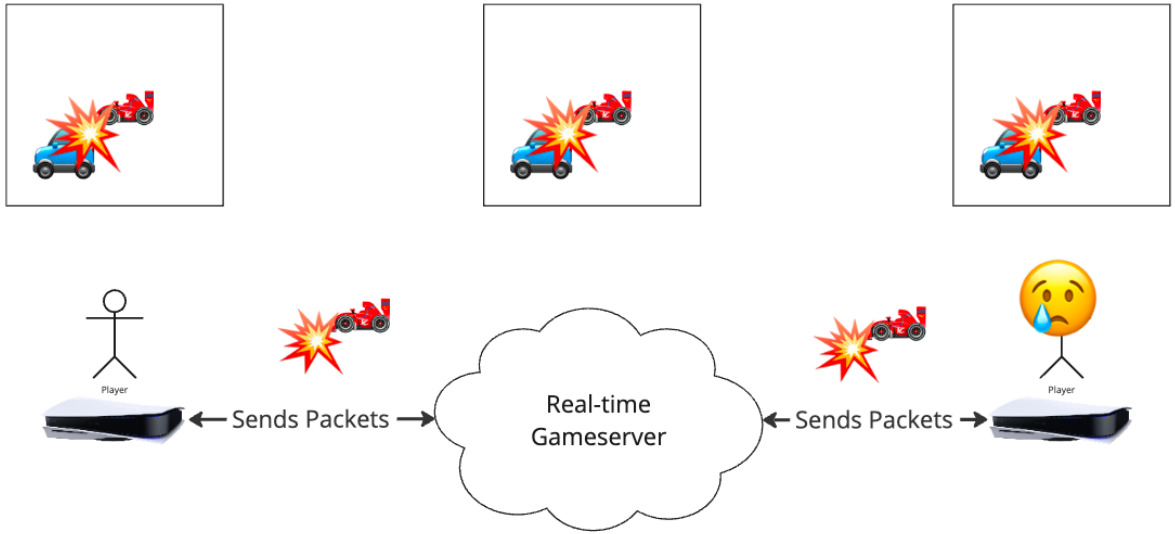
Dedicated Game Server



Dedicated Game Server



Dedicated Game Server



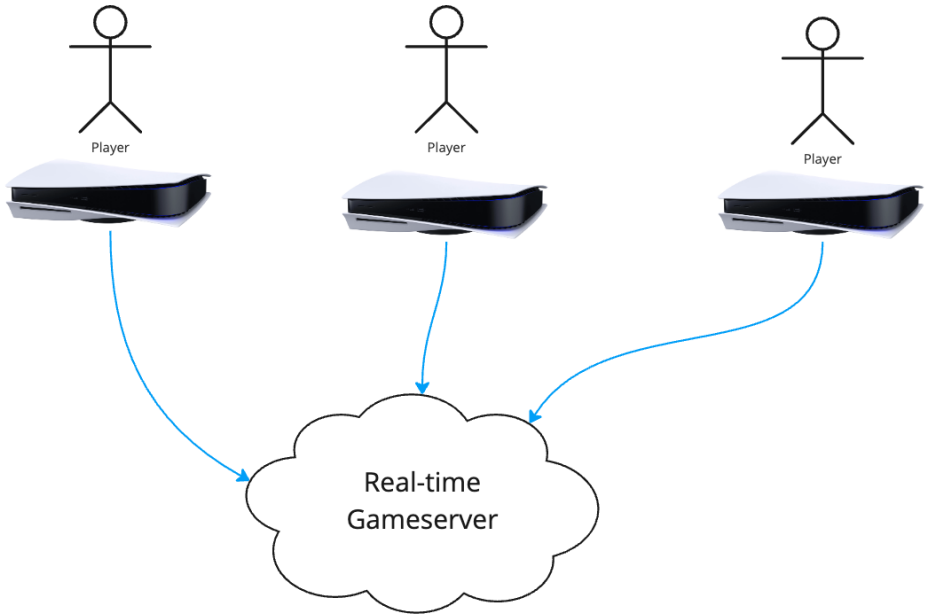
Real-time

Moment to moment

UDP

Persistent Connection

Latency Sensitive



LEVEL SELECT:

TUTORIAL: REAL-TIME GAMSERVERS

LEVEL 1: RUNNING IN KUBERNETES

LEVEL 2: AUTOSCALING

LEVEL 3: MULTI-REGION

Agones

Open Source Project for running game servers in Kubernetes

Made by Google

googleforgames/agones

Pod Termination

Cluster Autoscaler

Deployment Rollouts

GameServers

```yaml
apiVersion: agones.dev/v1
kind: GameServer
metadata:
  name: us-west-2-game-servers-p2qvj-6tj9k
  namespace: dev-game-server
spec:
  container: simple-game-server
  ports:
  - container: simple-game-server
    containerPort: 7654
    hostPort: 7025
    portPolicy: Dynamic
    protocol: UDP
  scheduling: Packed
  template:
    spec:
      containers:
      - image: gcr.io/agones-images/simple-game-server:0.13
        name: simple-game-server
```

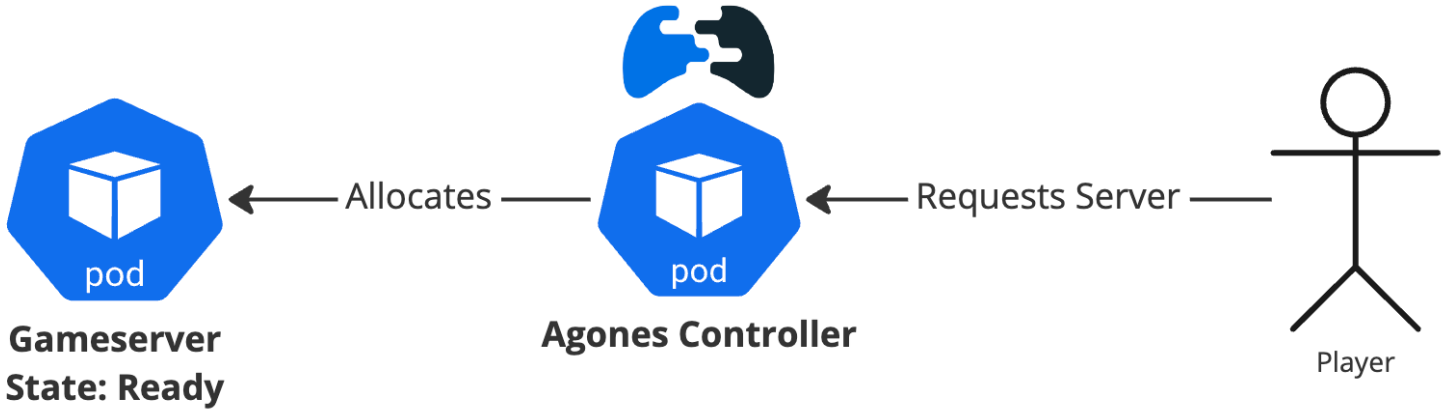
```
kubectl get gameserver
NAME                                  STATE   ADDRESS                                          PORT   NODE                                          AGE
us-west-2-game-servers-p2qvj-669m8    Ready   ec2-35-89-190-116.us-west-2.compute.amazonaws.com   7140   ip-10-0-6-88.us-west-2.compute.internal   3d18h
us-west-2-game-servers-p2qvj-6tj9k    Ready   ec2-35-89-190-116.us-west-2.compute.amazonaws.com   7025   ip-10-0-6-88.us-west-2.compute.internal   3d18h
```

GameServers

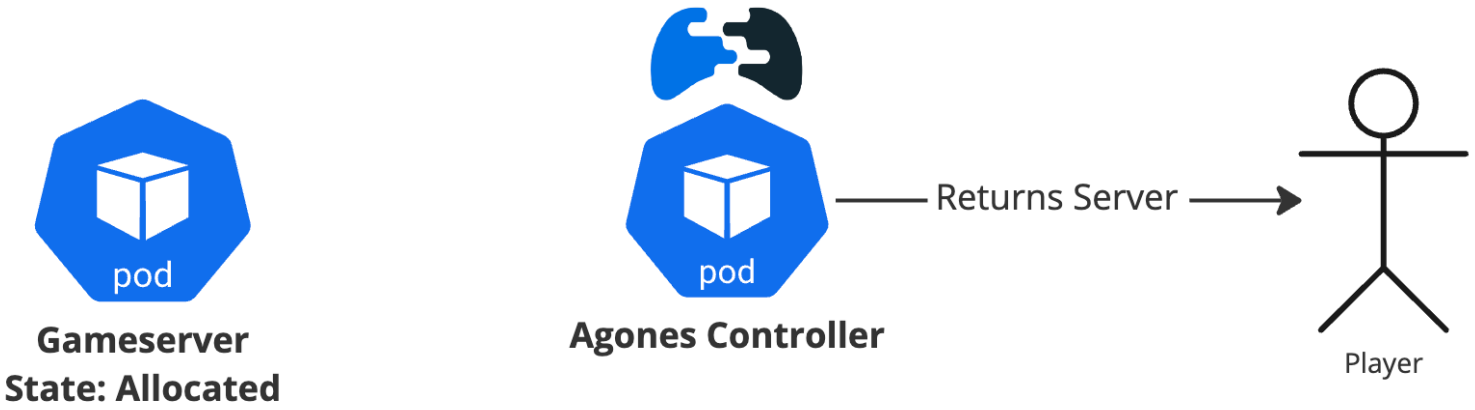
```yaml
apiVersion: agones.dev/v1
kind: GameServer
metadata:
  name: us-west-2-game-servers-p2qvj-6tj9k
  namespace: dev-game-server
spec:
  container: simple-game-server
  ports:
  - container: simple-game-server
    containerPort: 7654
    hostPort: 7025
    portPolicy: Dynamic
    protocol: UDP
  scheduling: Packed
  template:
    spec:
      containers:
      - image: gcr.io/agones-images/simple-game-server:0.13
        name: simple-game-server
```

```
kubectl get gameserver
NAME                                  STATE   ADDRESS                                          PORT   NODE                                          AGE
us-west-2-game-servers-p2qvj-669m8    Ready   ec2-35-89-190-116.us-west-2.compute.amazonaws.com   7140   ip-10-0-6-88.us-west-2.compute.internal   3d18h
us-west-2-game-servers-p2qvj-6tj9k    Ready   ec2-35-89-190-116.us-west-2.compute.amazonaws.com   7025   ip-10-0-6-88.us-west-2.compute.internal   3d18h
```

GameServers



GameServers



GameServer Allocation

```yaml
apiVersion: "allocation.agones.dev/v1"
kind: GameServerAllocation
spec:
  selectors:
    - matchLabels:
        agones.dev/fleet: my-fleet
      players:
        minAvailable: 10
        maxAvailable: 20
```

GameServer Allocation

```yaml
apiVersion: "allocation.agones.dev/v1"
kind: GameServerAllocation
spec:
  selectors:
    - matchLabels:
        agones.dev/fleet: my-fleet
      players:
        minAvailable: 10
        maxAvailable: 20
```

GameServer Allocation

```yaml
apiVersion: "allocation.agones.dev/v1"
kind: GameServerAllocation
metadata:
  name: my-allocation
spec:
  selectors:
    - matchLabels:
        agones.dev/fleet: my-fleet
      players:
        minAvailable: 10
        maxAvailable: 20
status:
  address: ec2-54-214-163-180.us-west-2.compute.amazonaws.com
  gameServerName: simple-game-server-ngtq6-ctgjc
  nodeName: ip-10-0-1-113.us-west-2.compute.internal
  ports:
  - name: default
    port: 7044
  state: Allocated
```

GameServer Allocation

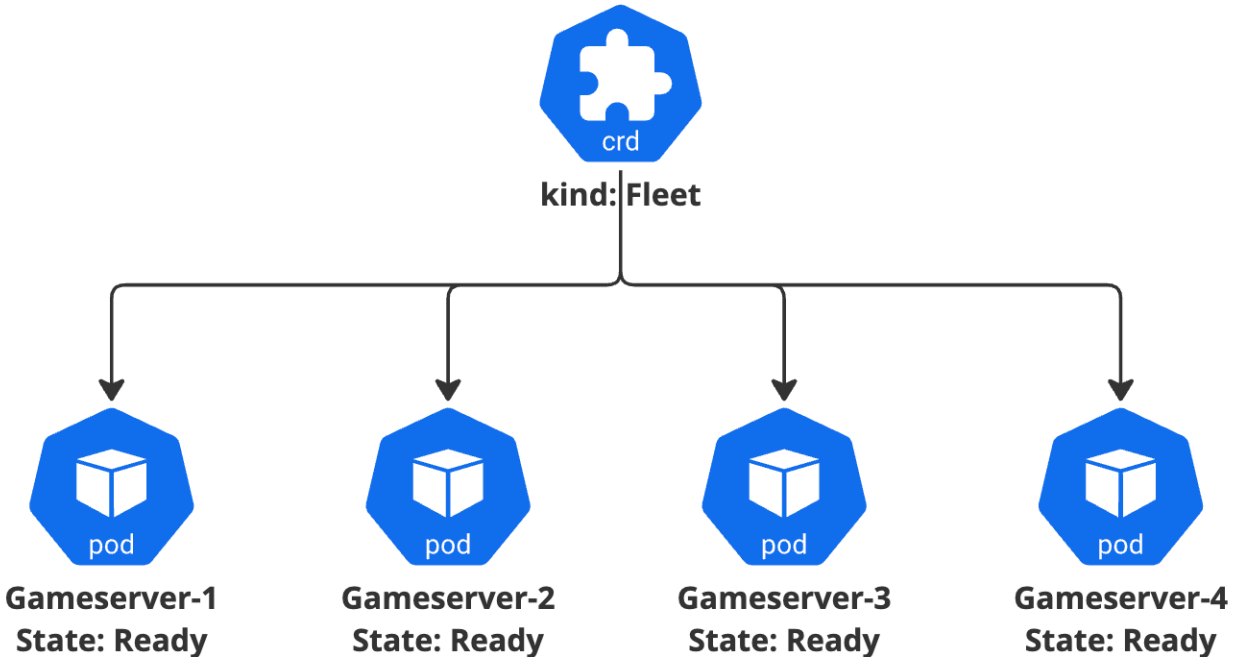
```yaml
apiVersion: "allocation.agones.dev/v1"
kind: GameServerAllocation
metadata:
  name: my-allocation
spec:
  selectors:
    - matchLabels:
        agones.dev/fleet: my-fleet
      players:
        minAvailable: 10
        maxAvailable: 20
status:
  address: ec2-54-214-163-180.us-west-2.compute.amazonaws.com
  gameServerName: simple-game-server-ngtq6-ctgjc
  nodeName: ip-10-0-1-113.us-west-2.compute.internal
  ports:
  - name: default
    port: 7044
  state: Allocated
```

Fleets

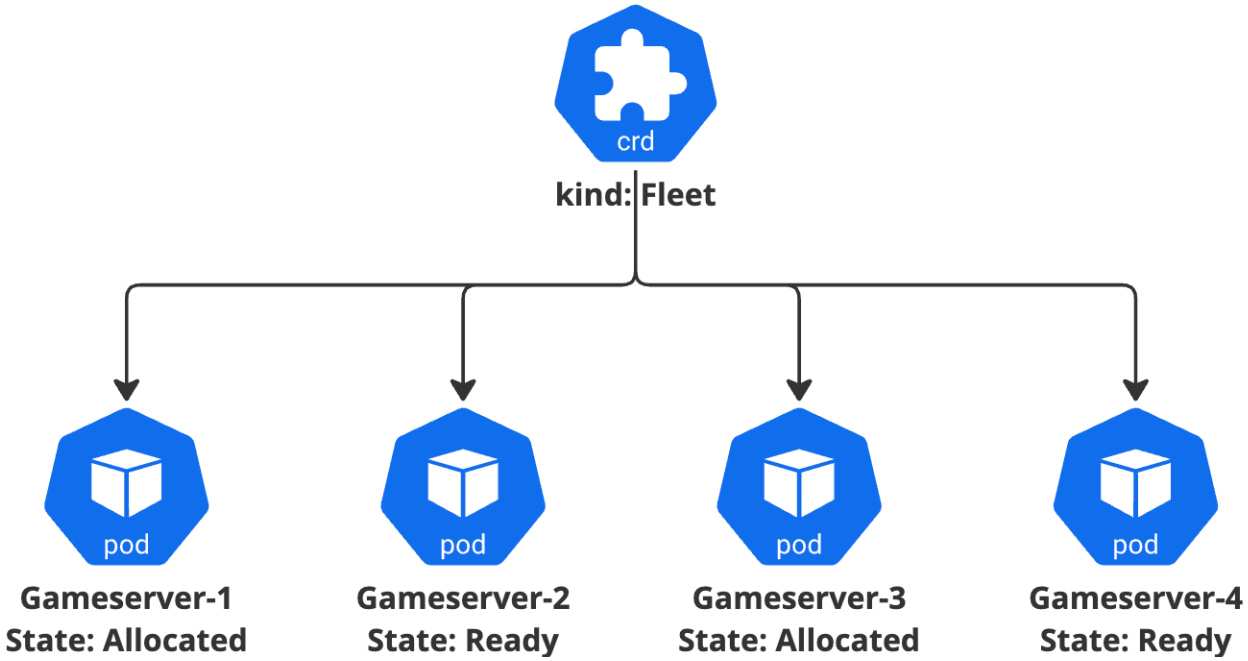
```yaml
apiVersion: agones.dev/v1
kind: Fleet
  name: us-west-2-game-servers
  namespace: dev-game-server
spec:
  replicas: 2
  scheduling: Packed
  template:
    spec:
      ports:
      - containerPort: 7654
        name: default
      template:
        spec:
          containers:
          - image: gcr.io/agones-images/simple-game-server:0.13
            name: simple-game-server
```

```
kubectl get fleets
NAME                   SCHEDULING   DESIRED   CURRENT   ALLOCATED   READY   AGE
us-west-2-game-servers   Packed       2         2         0           2       3d22h
```

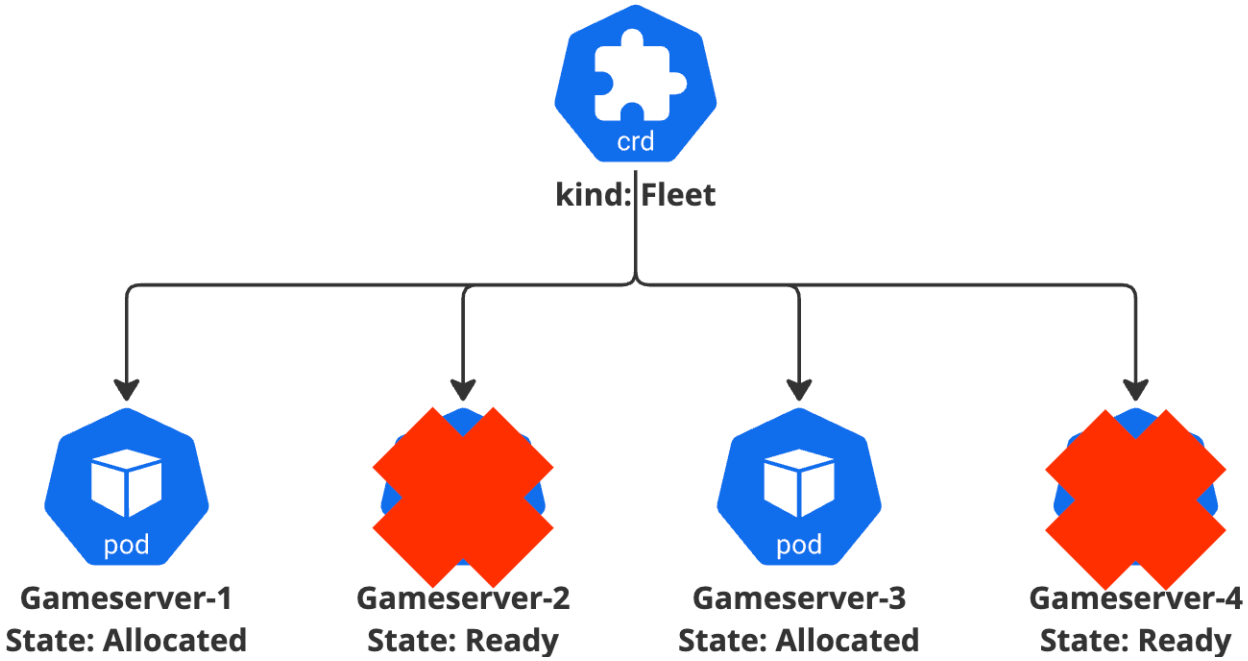
Fleets


crd
kind: Fleet
pod
pod
pod
pod
Gameserver-1
State: Ready
Gameserver-2
State: Ready
Gameserver-3
State: Ready
Gameserver-4
State: Ready

Kubectl scale fleet –replicas=1



Kubectl scale fleet –replicas=1



LEVEL SELECT:

TUTORIAL: REAL-TIME GAMSERVERS

LEVEL 1: RUNNING IN KUBERNETES

LEVEL 2: AUTOSCALING

LEVEL 3: MULTI-REGION

Autoscaling

HPA based on CPU/Memory doesn’t make sense with gameservers

Autoscaling needs to consider the state of the gameservers

Fleet Autoscaler

```yaml
apiVersion: "autoscaling.agones.dev/v1"
kind: FleetAutoscaler
metadata:
  name: fleet-autoscaler-example
spec:
  fleetName: fleet-example
  policy:
    type: Buffer
    buffer:
      bufferSize: 5
      minReplicas: 10
      maxReplicas: 20
```

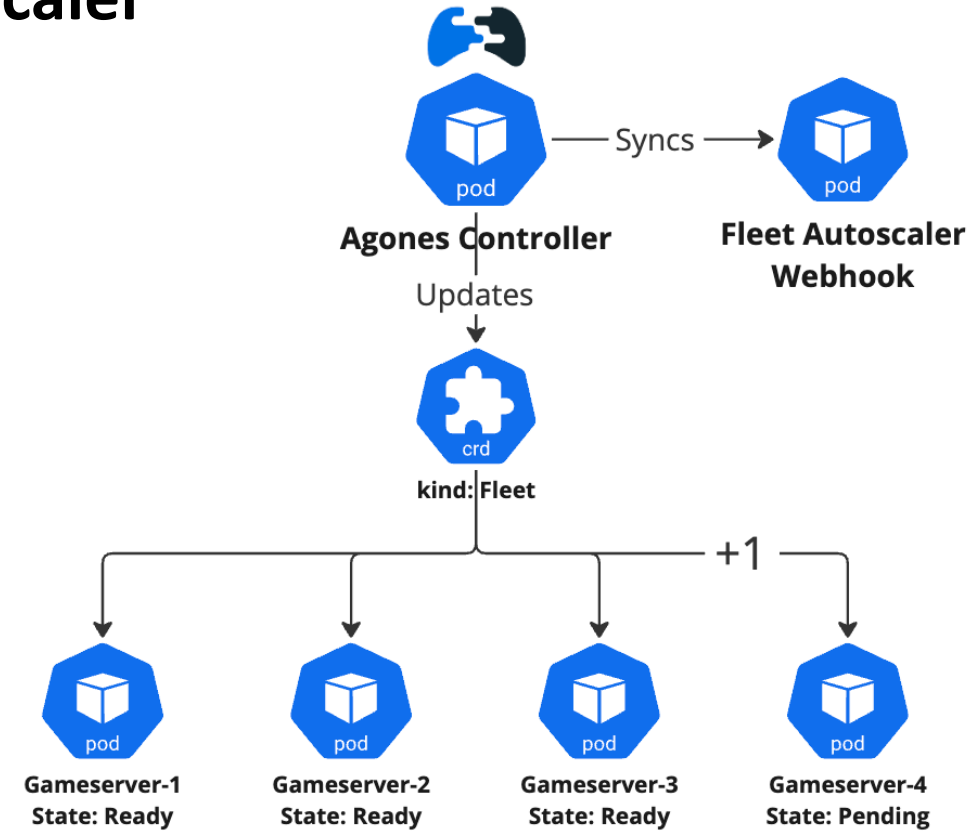
Fleet Autoscaler

```yaml
apiVersion: "autoscaling.agones.dev/v1"
kind: FleetAutoscaler
metadata:
  name: fleet-autoscaler-example
spec:
  fleetName: fleet-example
  policy:
    type: Buffer
    buffer:
      bufferSize: 5
      minReplicas: 10
      maxReplicas: 20
```

Fleet Autoscaler

```yaml
apiVersion: "autoscaling.agones.dev/v1"
kind: FleetAutoscaler
metadata:
  name: fleet-autoscaler-example
spec:
  fleetName: fleet-example
  policy:
    type: Webhook
    webhook:
      service:
        name: autoscaler-webhook-service
        namespace: default
        path: scale
```

Fleet Autoscaler



LEVEL SELECT:

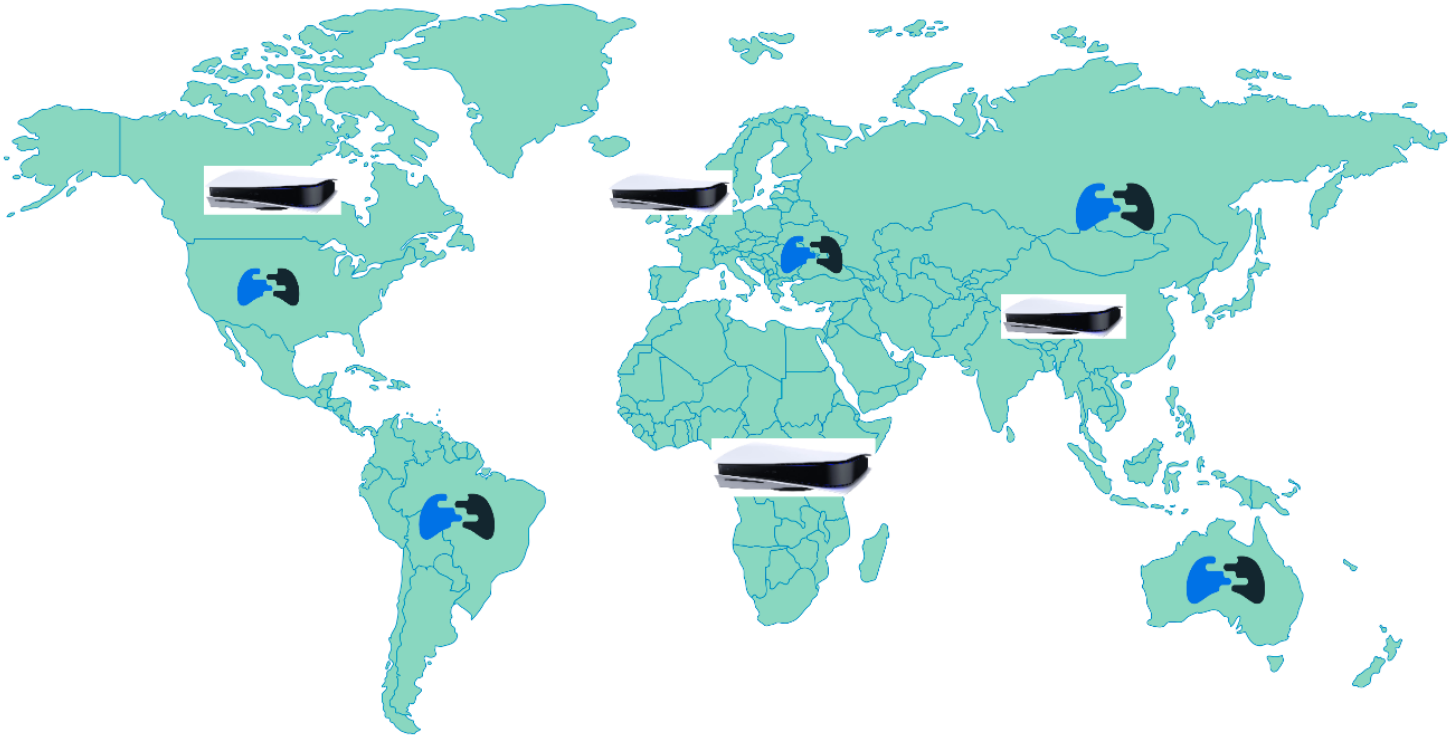
TUTORIAL: REAL-TIME GAMSERVERS

LEVEL 1: RUNNING IN KUBERNETES

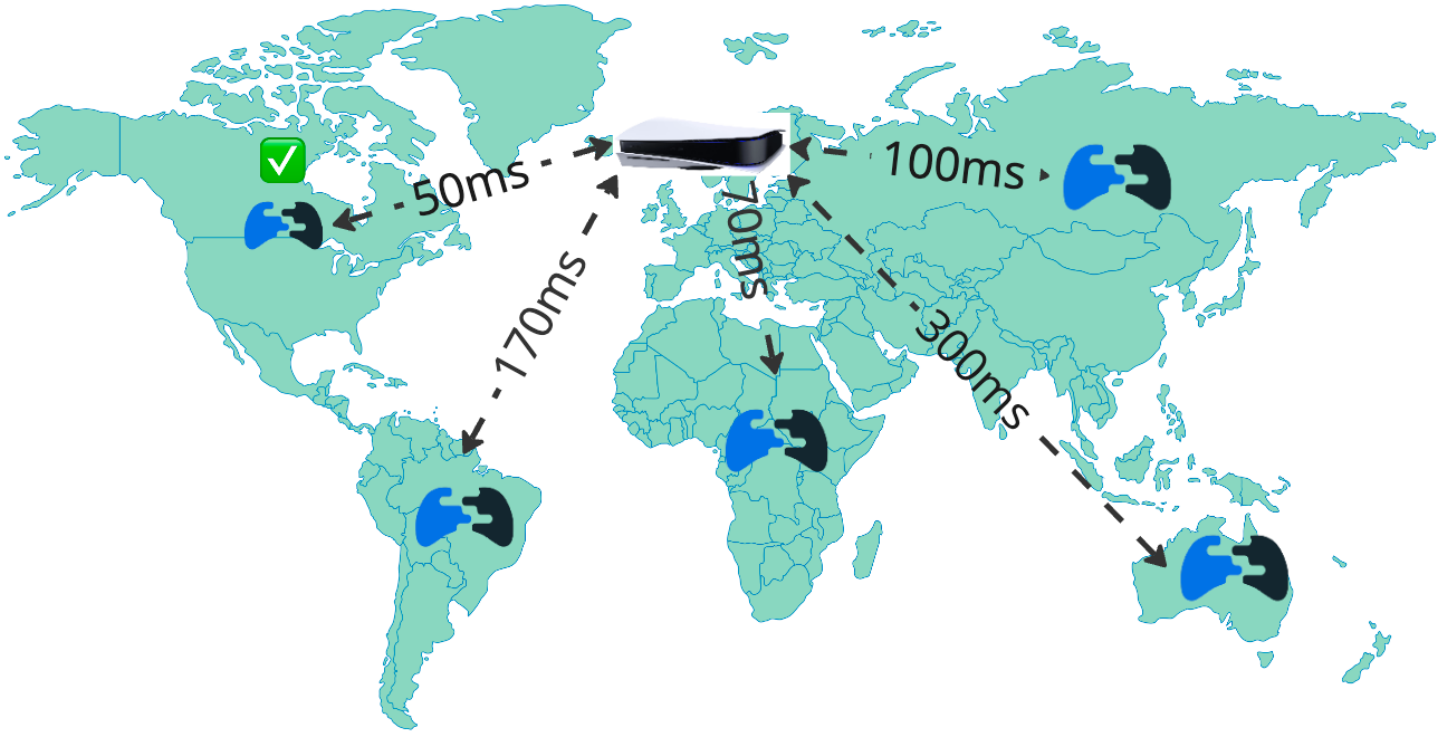
LEVEL 2: AUTOSCALING

LEVEL 3: MULTI-REGION

Multi-Region Agones



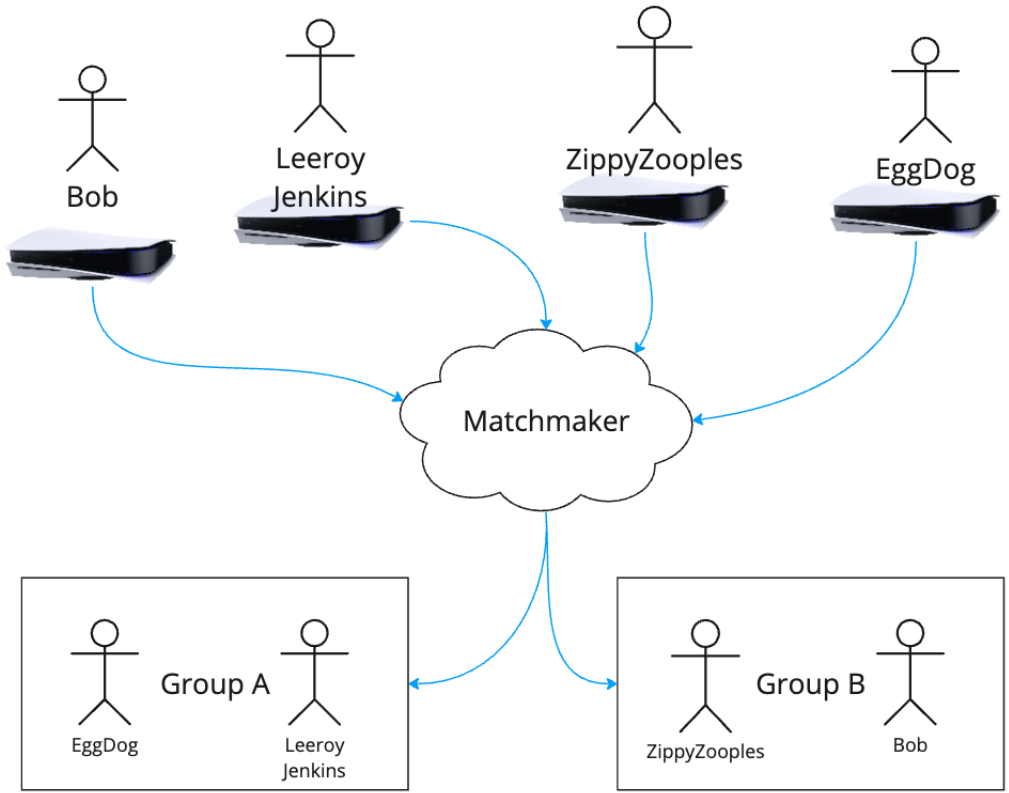
Multi-Region Agones



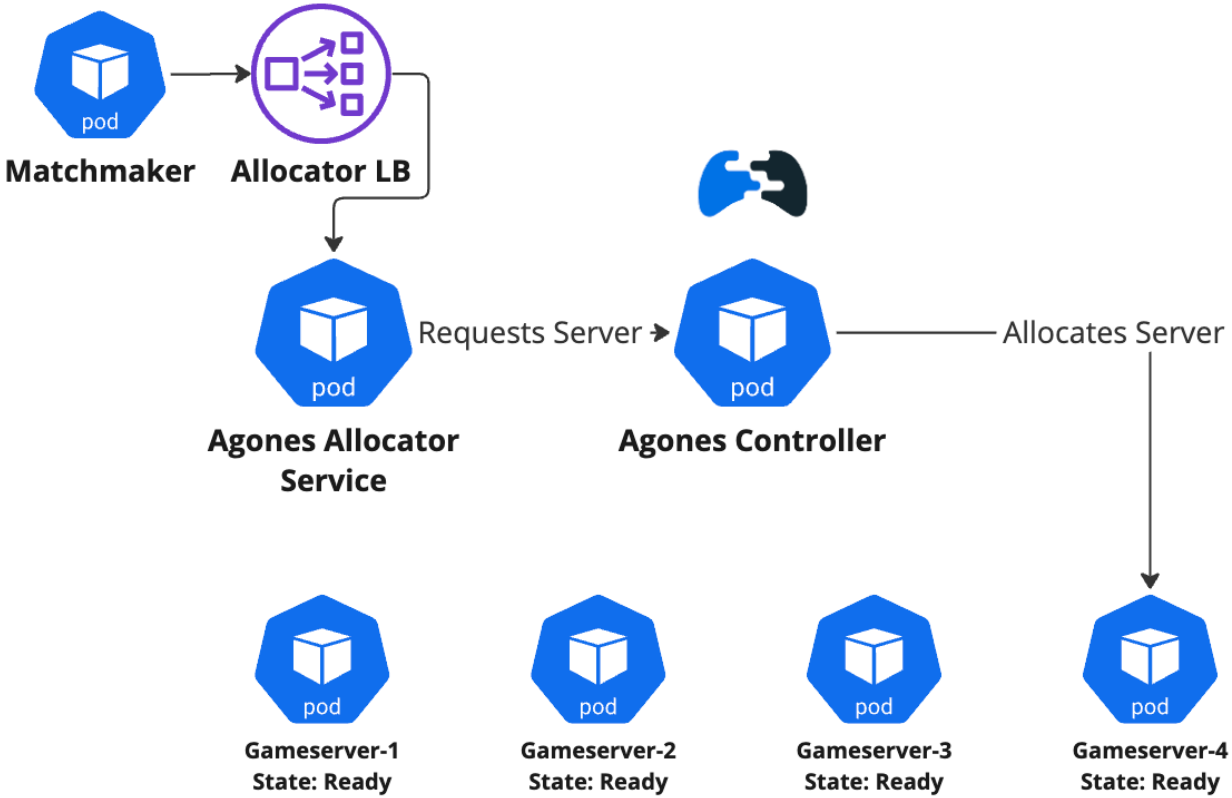
Matchmaker


Madtoker
<PALS FOR LIFE>
Juron
<PALS FOR LIFE>
Forekin
<PALS FOR LIFE>
<AFK>Leeroy
<PALS FOR LIFE>

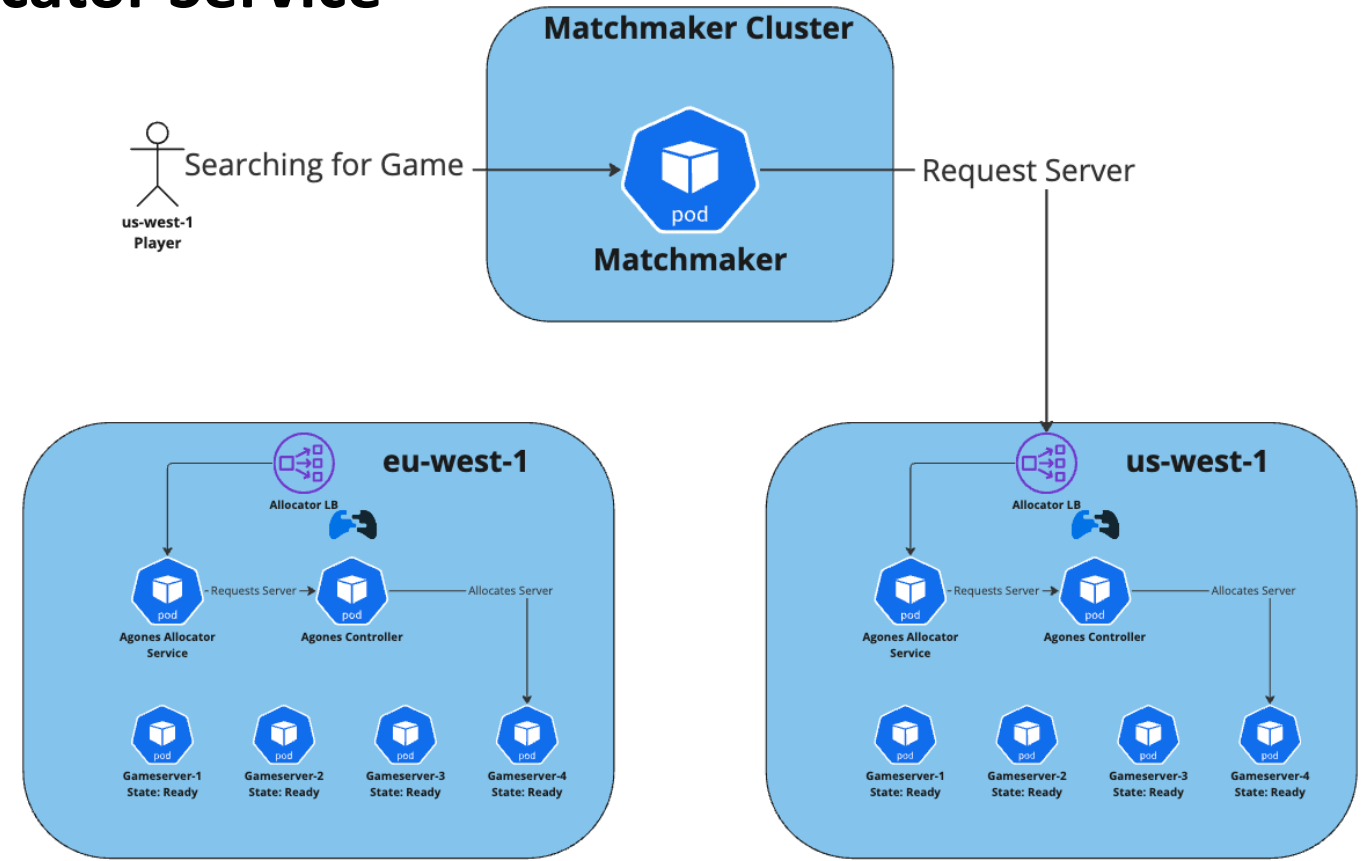
Matchmaker


Bob
Leeroy Jenkins
ZippyZooples
EggDog
Matchmaker
Group A
EggDog
Leeroy Jenkins
Group B
ZippyZooples
Bob

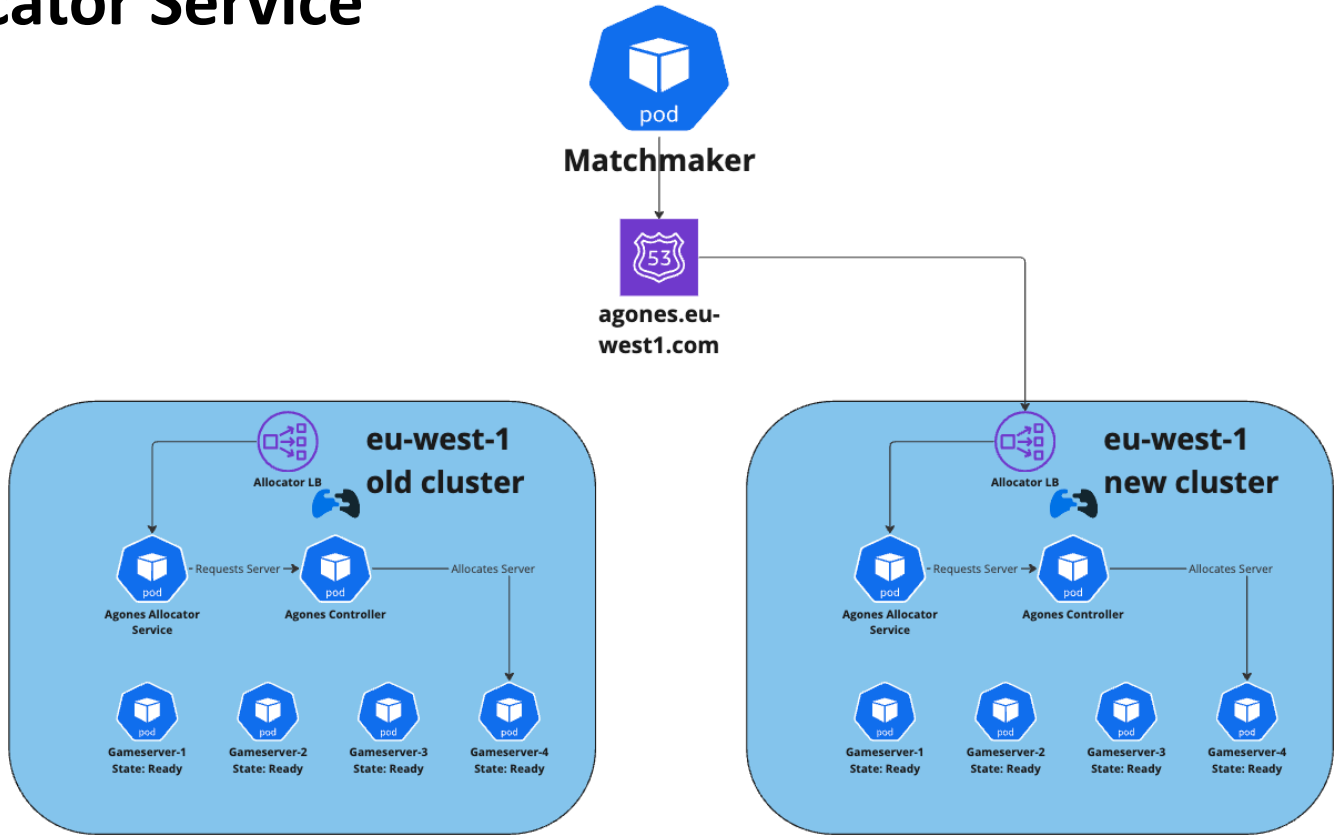
Allocator Service



Allocator Service



Allocator Service



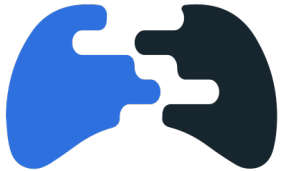
To Conclude…

Fleets and Gameservers

Fleet Autoscalers

Allocator Service

Multi-Region



agones

Scan the QR Code to leave feedback and see slides!