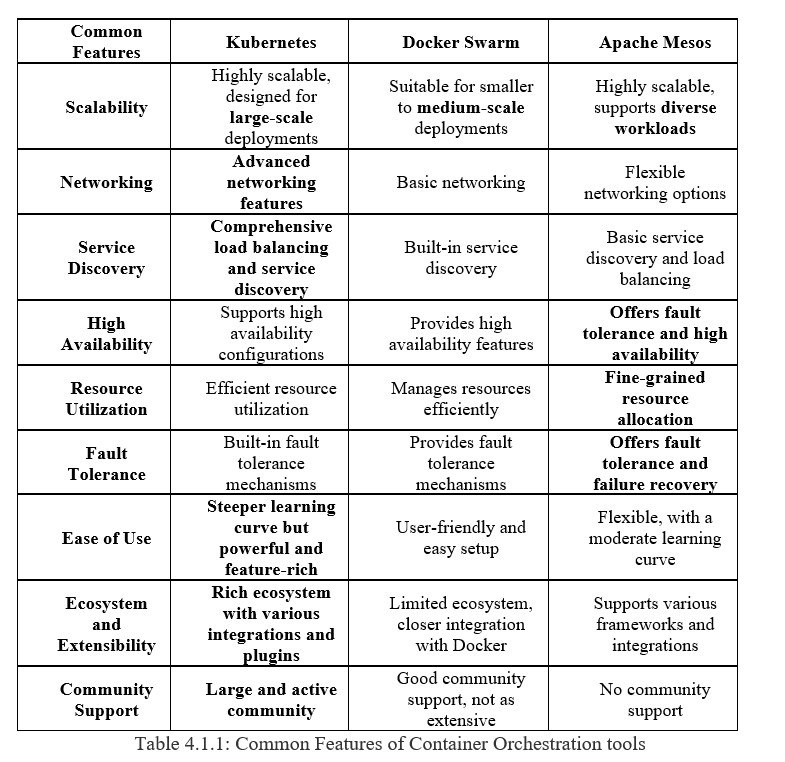
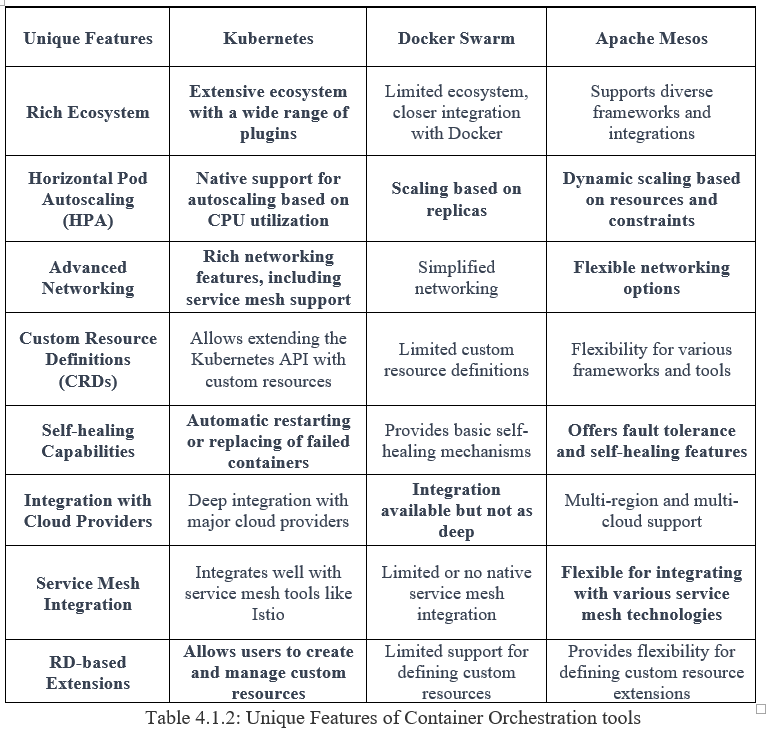
Common features:



Unique Features:



### Test Results output

All our quantitative metrics measured are represented as graphs and discussed below.

### Graph 4.3.1: Provisioning time of the orchestration tools

### 4.3.2 Provisioning time of the application with different complexity:

*In a container orchestration tool denotes the period required to set up, configure, and deploy an application with different complexity within the orchestration-environment.*

### Graph 4.3.2: Provisioning time of applications with different complexity

### 4.3.3 CPU usage with different complexity: Resource utilization

### Graph 4.3.3a: CPU usage with different complexity

### Graph 4.3.3b: Memory usage with different complexity

### 4.3.4 Failover Time of the containers:

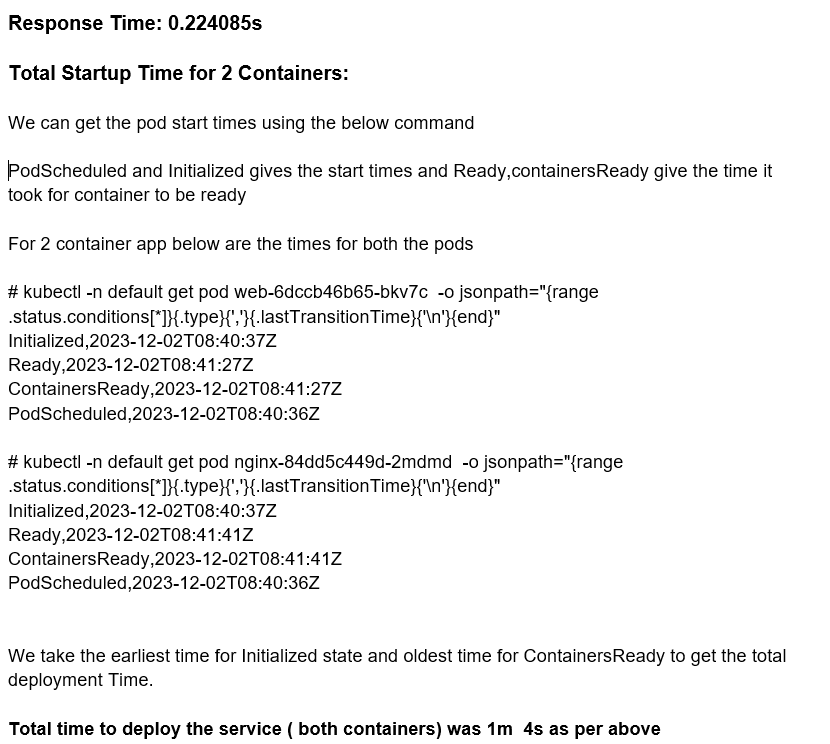
*Failover time in container orchestration is the period required for an orchestrator to identify a system failure and then restore services by diverting or restoring them to functional resources or nodes.*

### Graph 4.3.4: Failover Time - Container Failure.

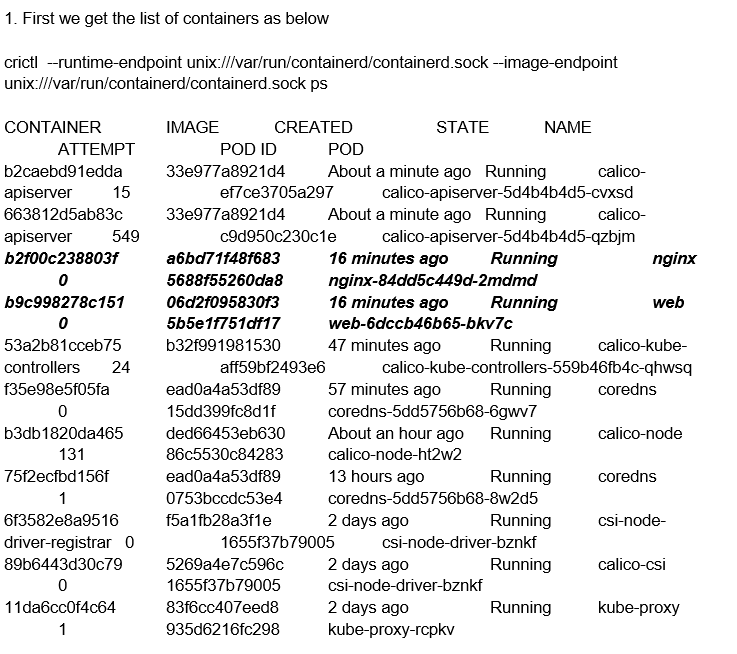
### 4.3.5 Response Time:

*Is the time it takes for an application or service deployed on an orchestration tool in our case to react to a specific request and generate a respond.*

### Graph 4.3.5: Response Time

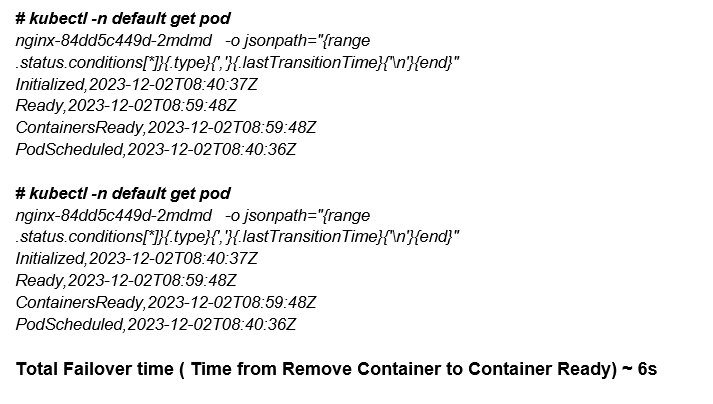
Adding execution results for Kubernetes below. Similar results of all 3 scenarios with different complexities, all the code snippets, commands used to fetch results, installation guide, chose n application repositories deployed on Apache Mesos, Docker Swarm and Kubernetes are all found in the link provided below. 

Kubernetes startup-time nginx-go



Kubernetes failover-time nginx-go

Then we capture the Container ready time in Kubernetes after killing them using the above command. From the below command we take the latest ContainersReady time as highlighted in this case



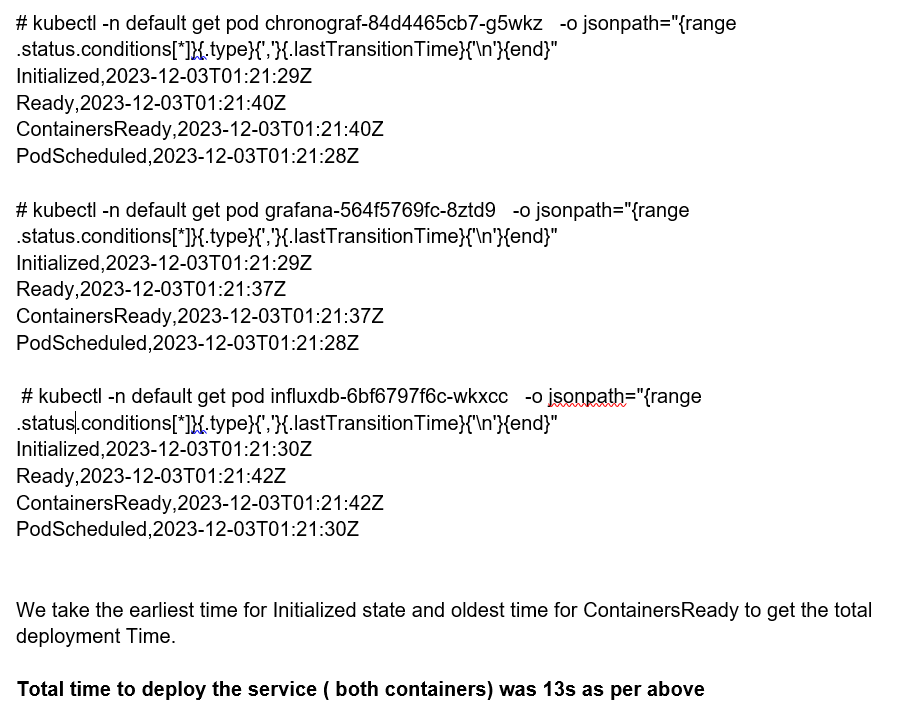
Kubernetes failover-start nginx-go

****

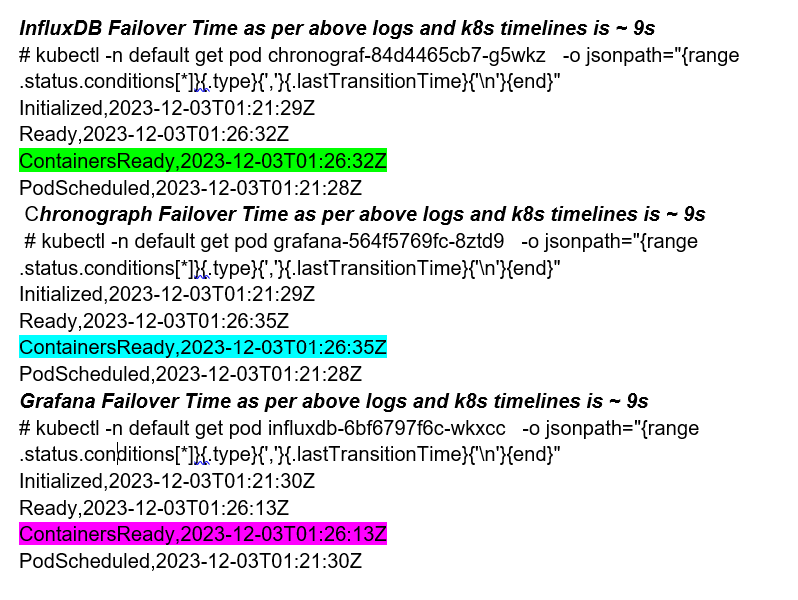
Kubernetes CPU and memory usage during load test nginx-go

****

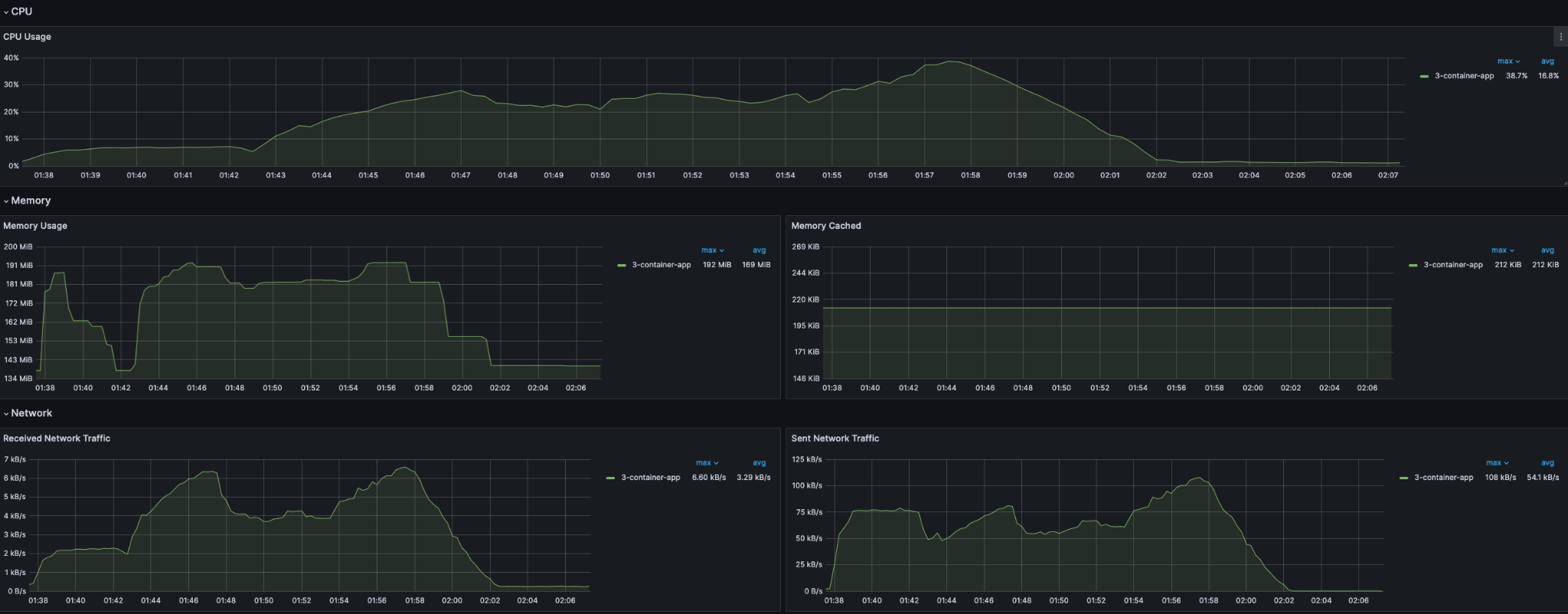
Kubernetes Network Traffic nginx-go



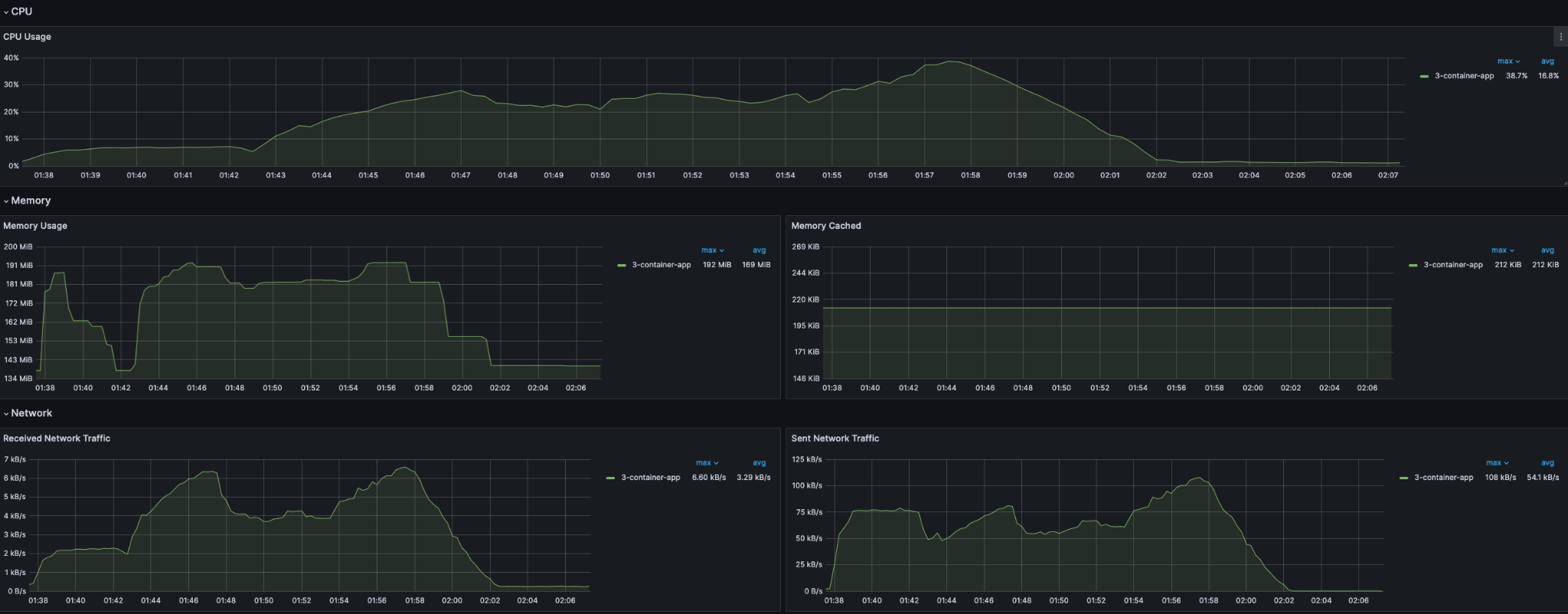
Kubernetes startup-time **influxdb-grafana-chrono**



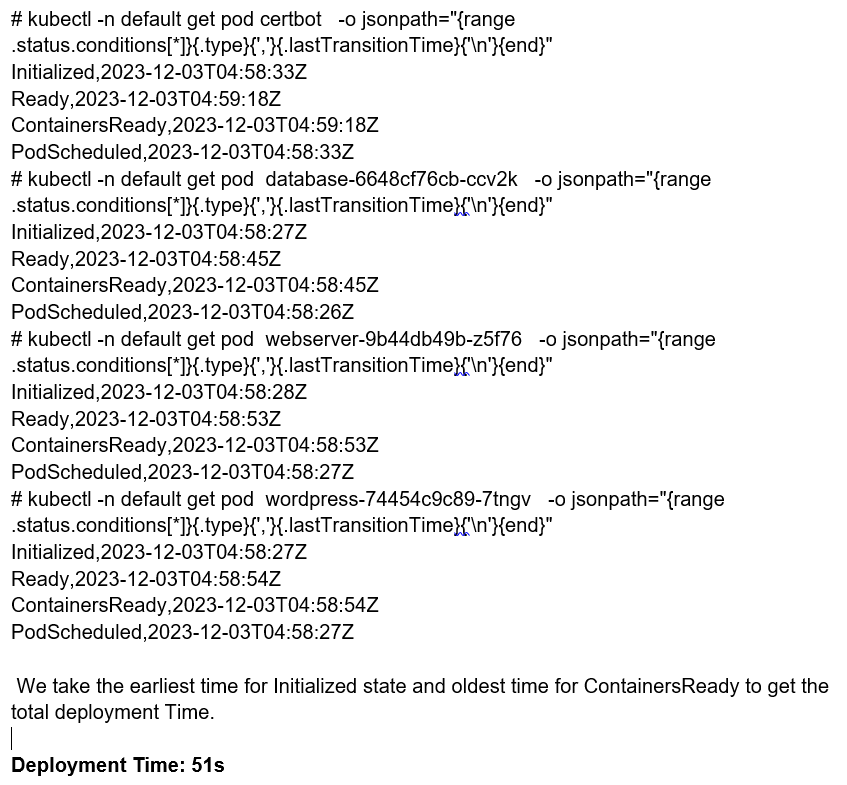
Kubernetes failover-time **influxdb-grafana-chrono**



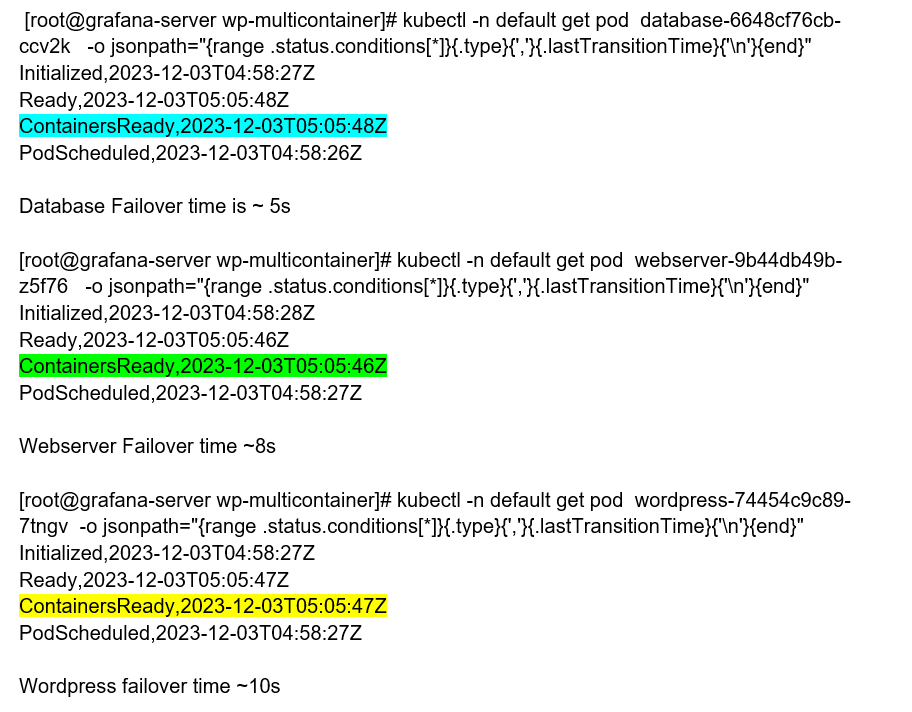
Kubernetes CPU and Memory (**influxdb-grafana-chrono**)



Kubernetes Network Traffic (**influxdb-grafana-chrono**)



Kubernetes startup-time Wordpress



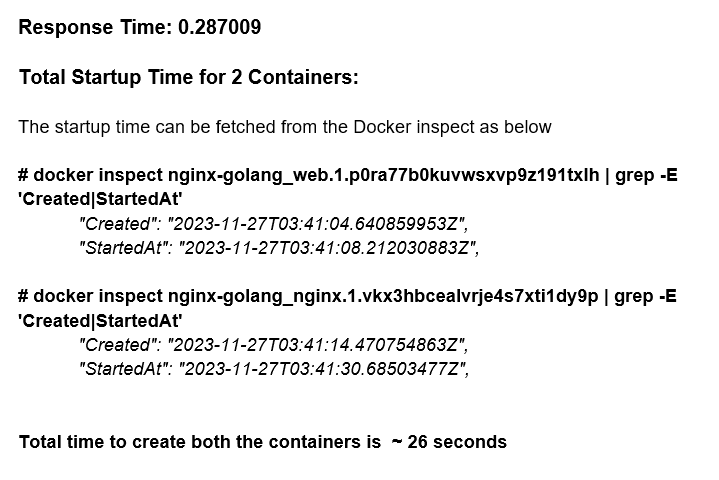
Kubernetes failover-time WordPress



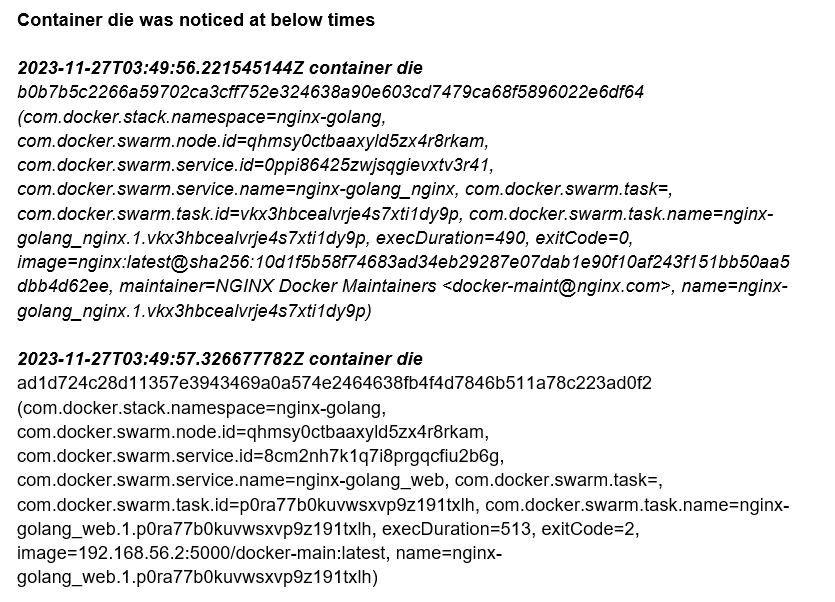
Kubernetes CPU and Memory (**WordPress**)



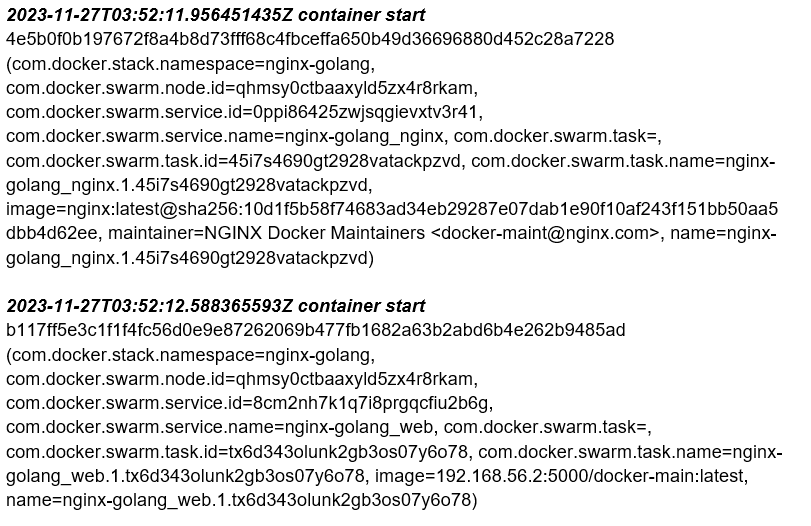
Kubernetes Network Traffic (**WordPress**)



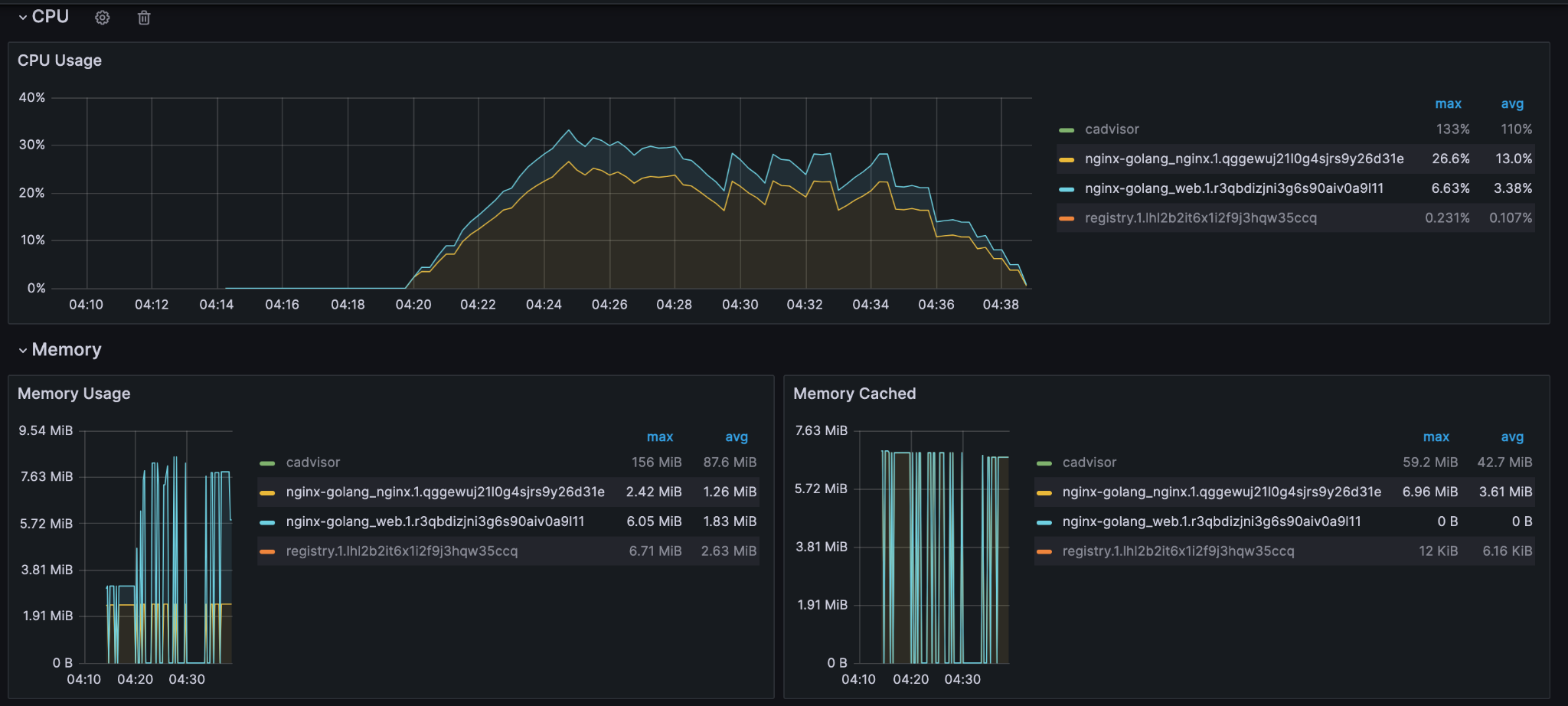
Docker startup-time nginx-go



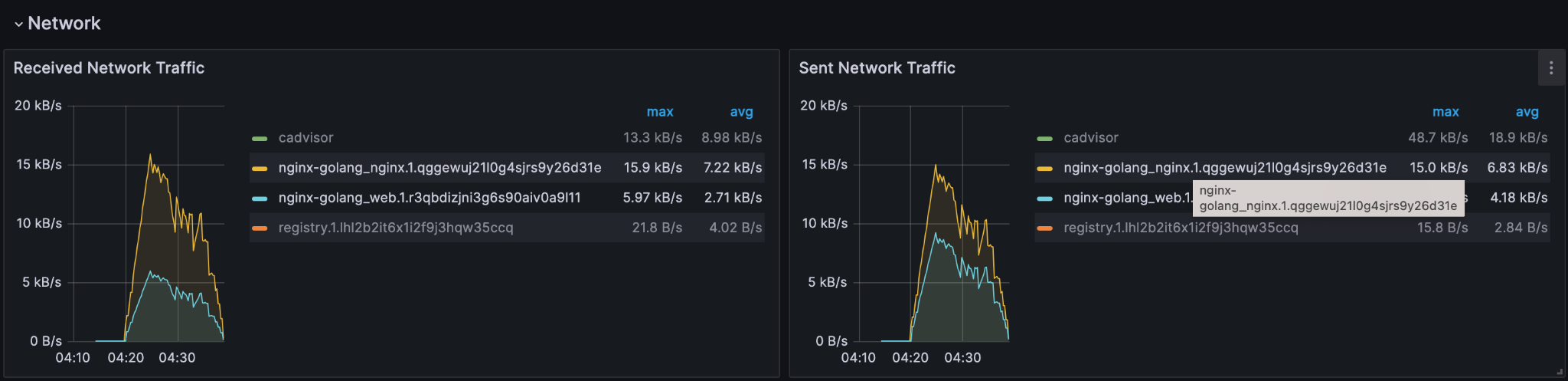
Docker failover-time nginx-go



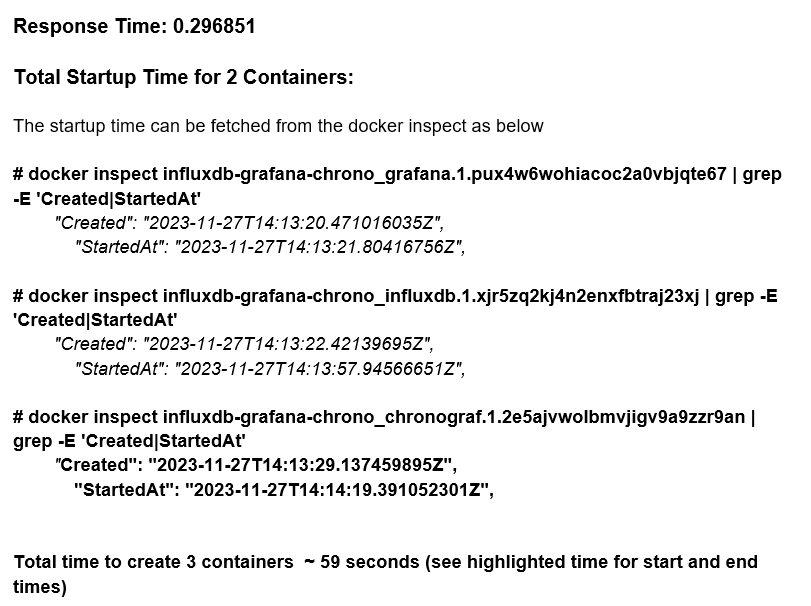
Docker failover-start nginx-go



Docker CPU and memory usage during load test nginx-go



Docker Network Traffic nginx-go



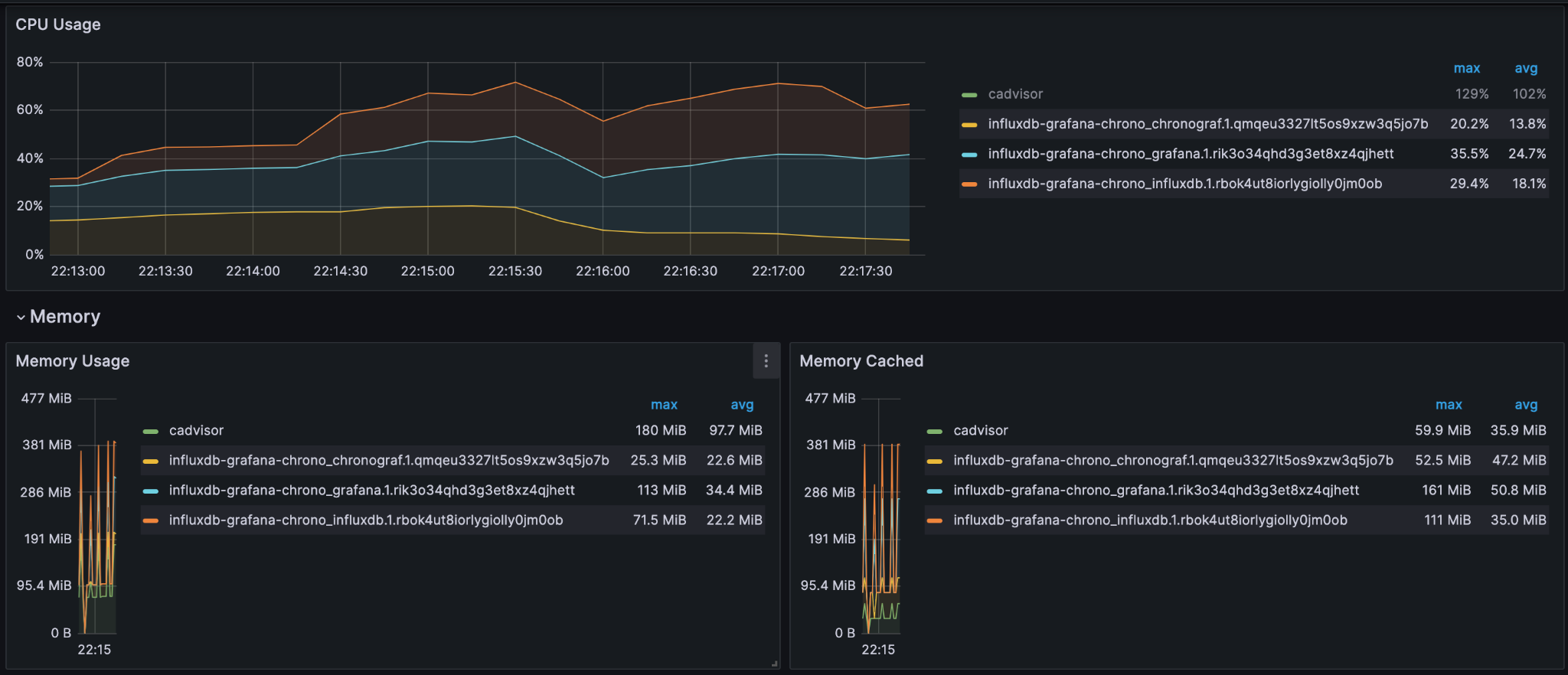
Docker startup-time **influxdb-grafana-chrono**



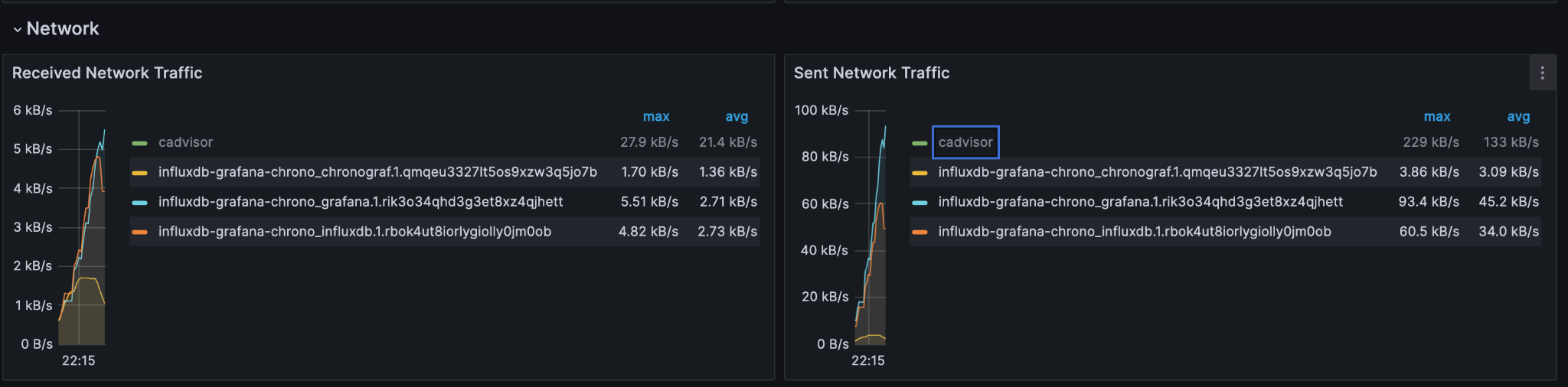
Docker failover-time **influxdb-grafana-chrono**



Docker failover-start **influxdb-grafana-chrono**



Docker CPU and Memory Usage with 1000 calls (**influxdb-grafana-chrono**)

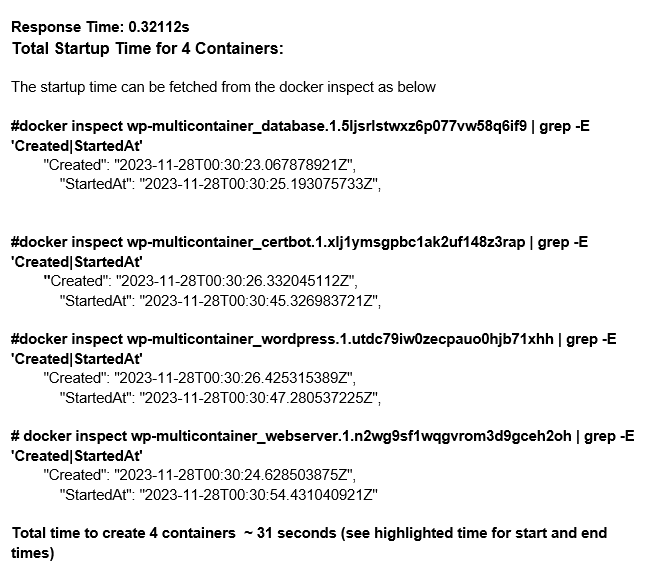
****

Docker Network Traffic with 1000 calls (**influxdb-grafana-chrono**)

To deploy the service with the above compose file, we use the below command

**docker stack deploy --compose-file compose.yaml wp-multicontainer**

Note: Docker swarm handles all the load balancing and DNS of services across the nodes.



Docker startup-time Wordpress

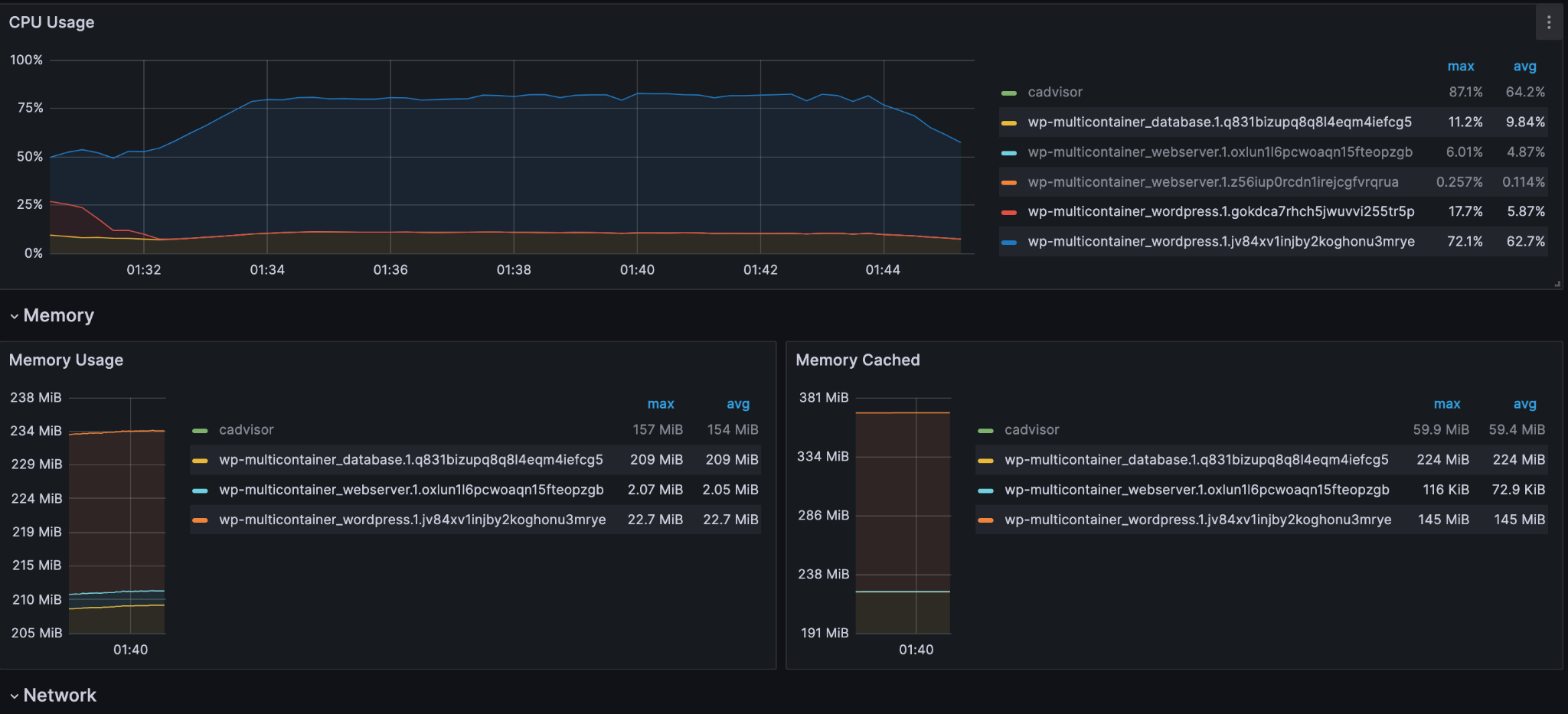




Docker failover-time Wordpress



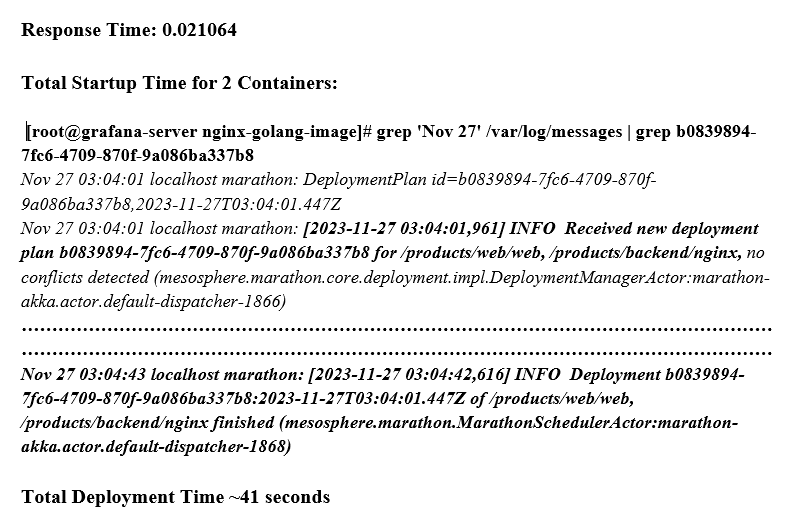
Docker failover-start Wordpress



Docker CPU and memory usage during load test (Wordpress)

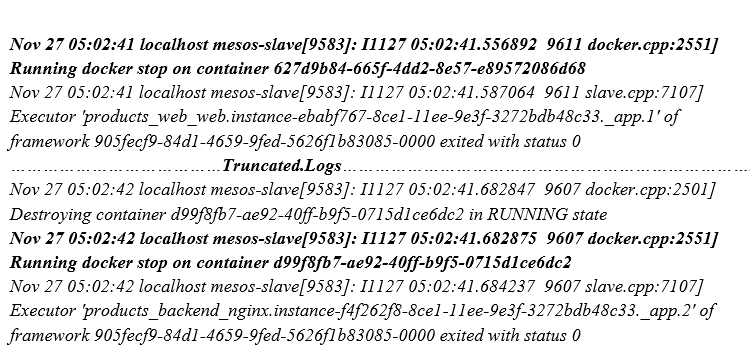


Network Traffic (Wordpress)



Mesos startup-time nginx-go

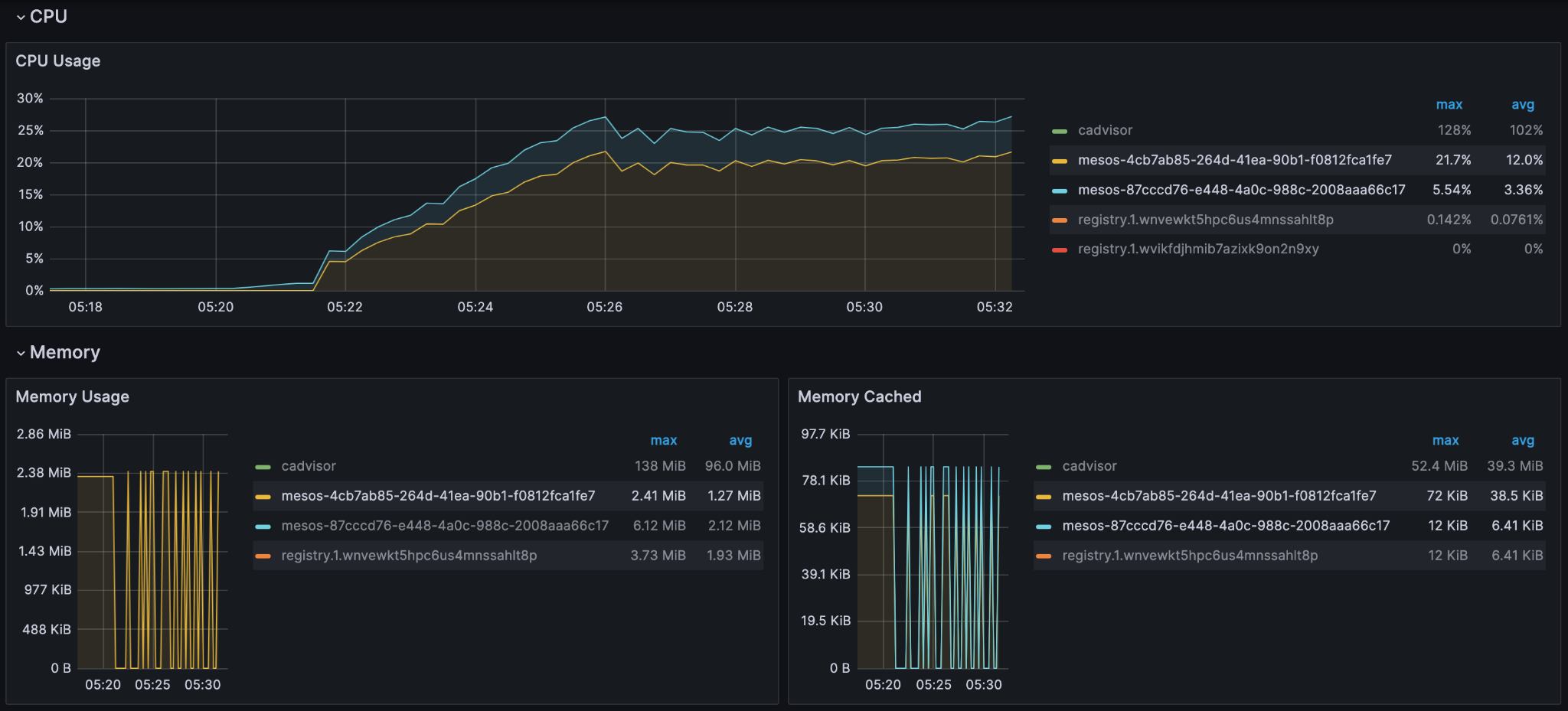
***#docker stop mesos-d99f8fb7-ae92-40ff-b9f5-0715d1ce6dc2 mesos-627d9b84-665f-4dd2-8e57-e89572086d68***



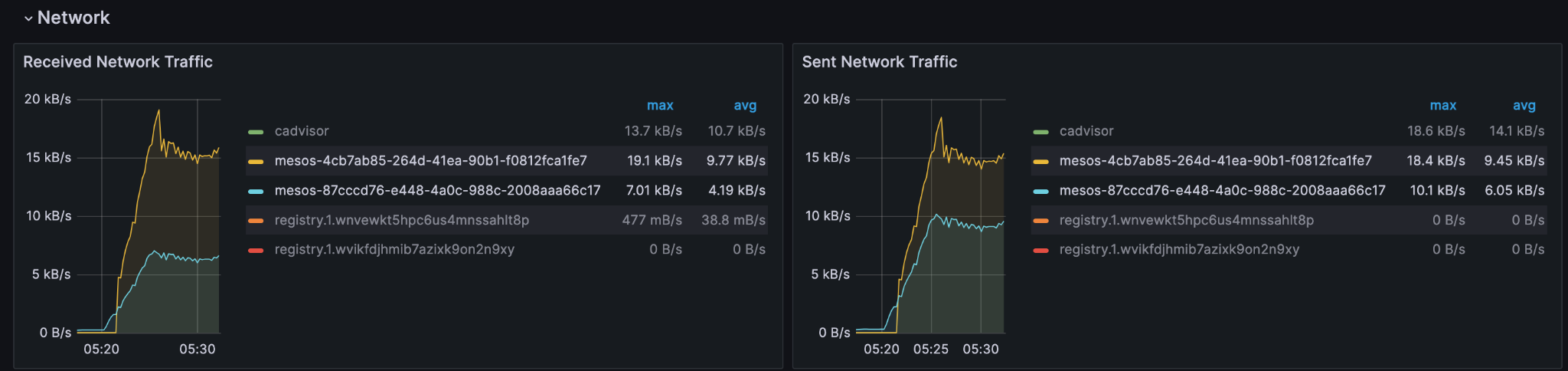
Docker failover-time nginx-go



Mesos failover-time nginx-go



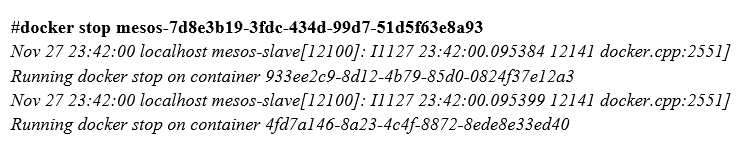
Mesos CPU and memory usage during load test nginx-go

**

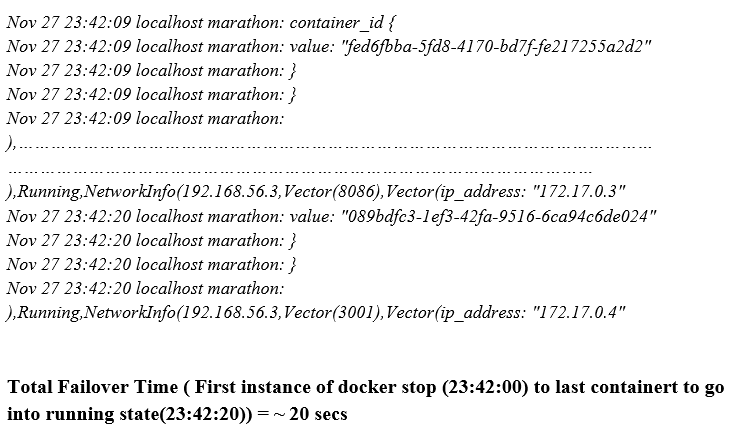
Mesos Network Traffic nginx-go



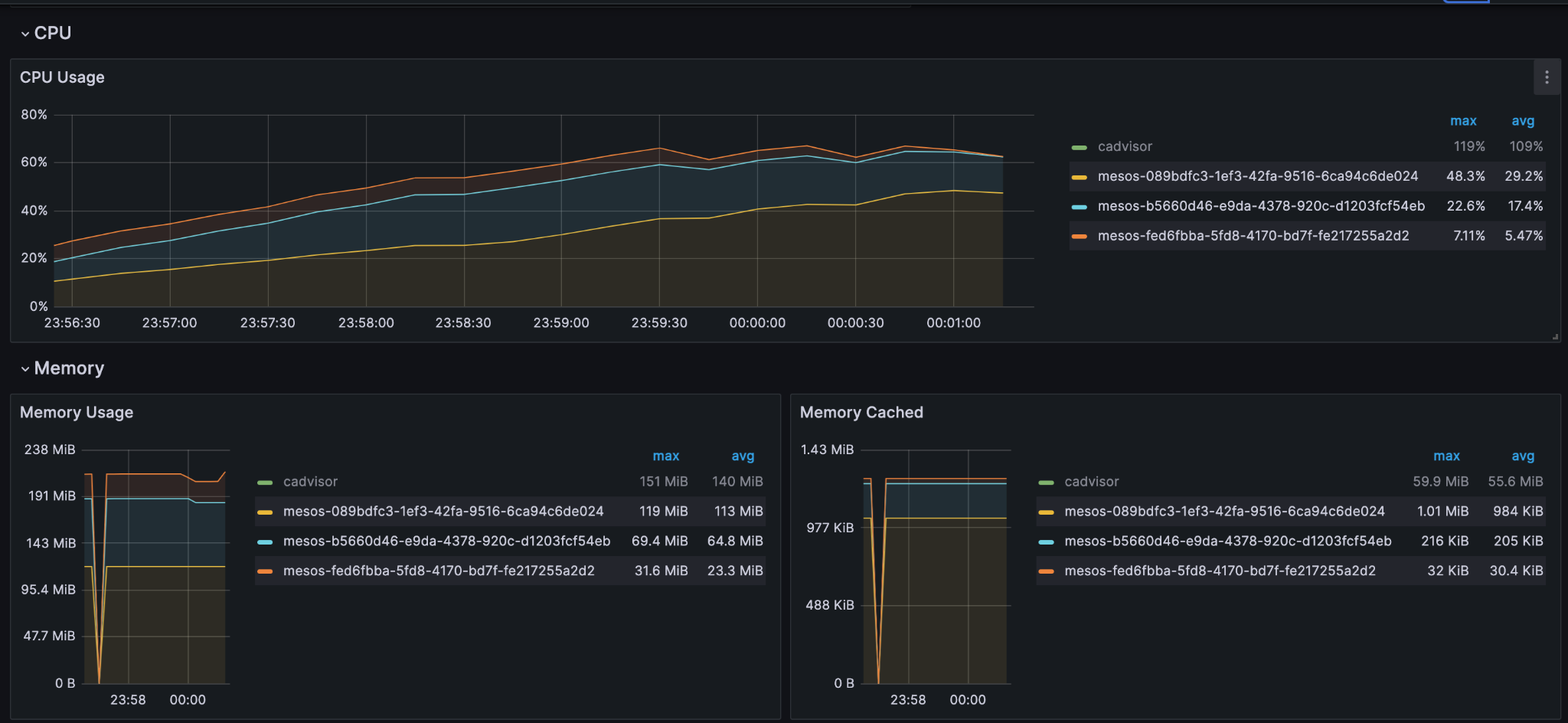
Mesos startup-time **influxdb-grafana-chrono**



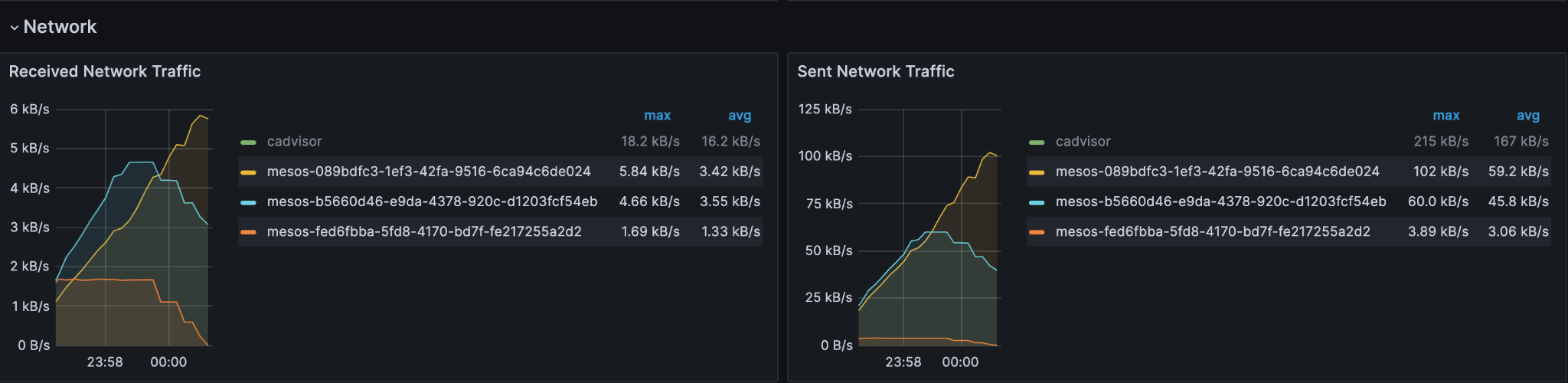
Mesos failover-time **influxdb-grafana-chrono**



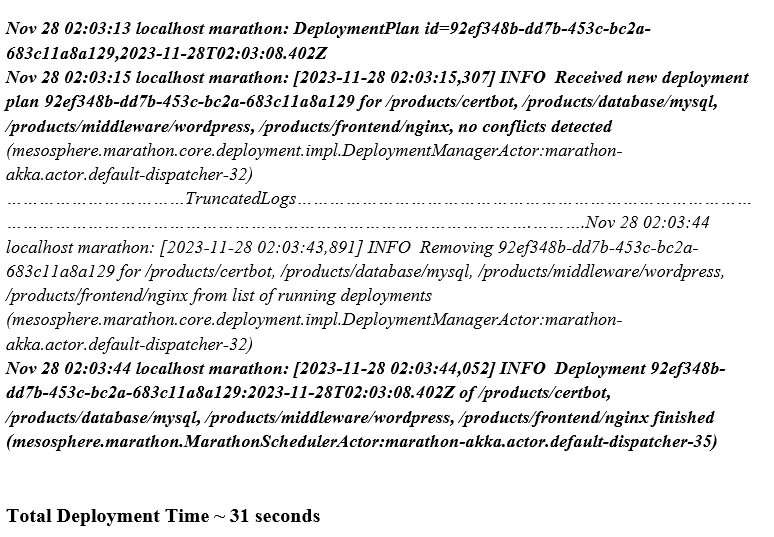
Mesos failover-start **influxdb-grafana-chrono**



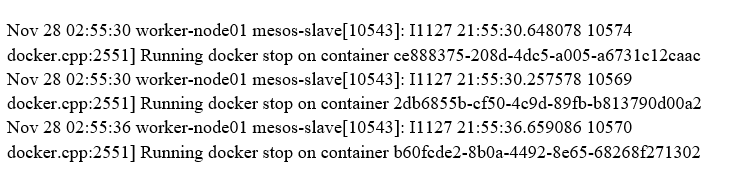
Mesos CPU and Memory Usage (**influxdb-grafana-chrono**)



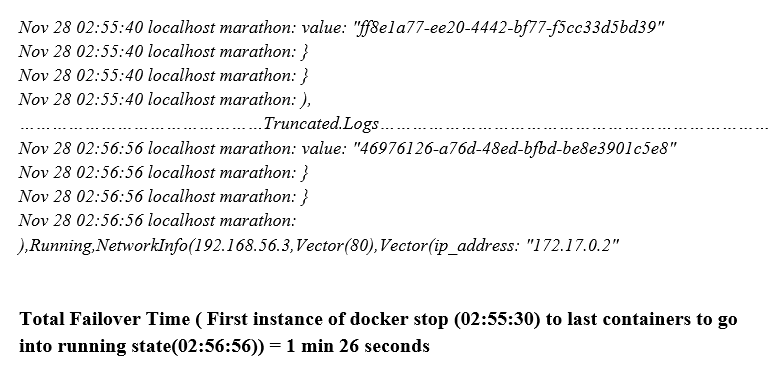
Mesos Network Traffic (**influxdb-grafana-chrono**)



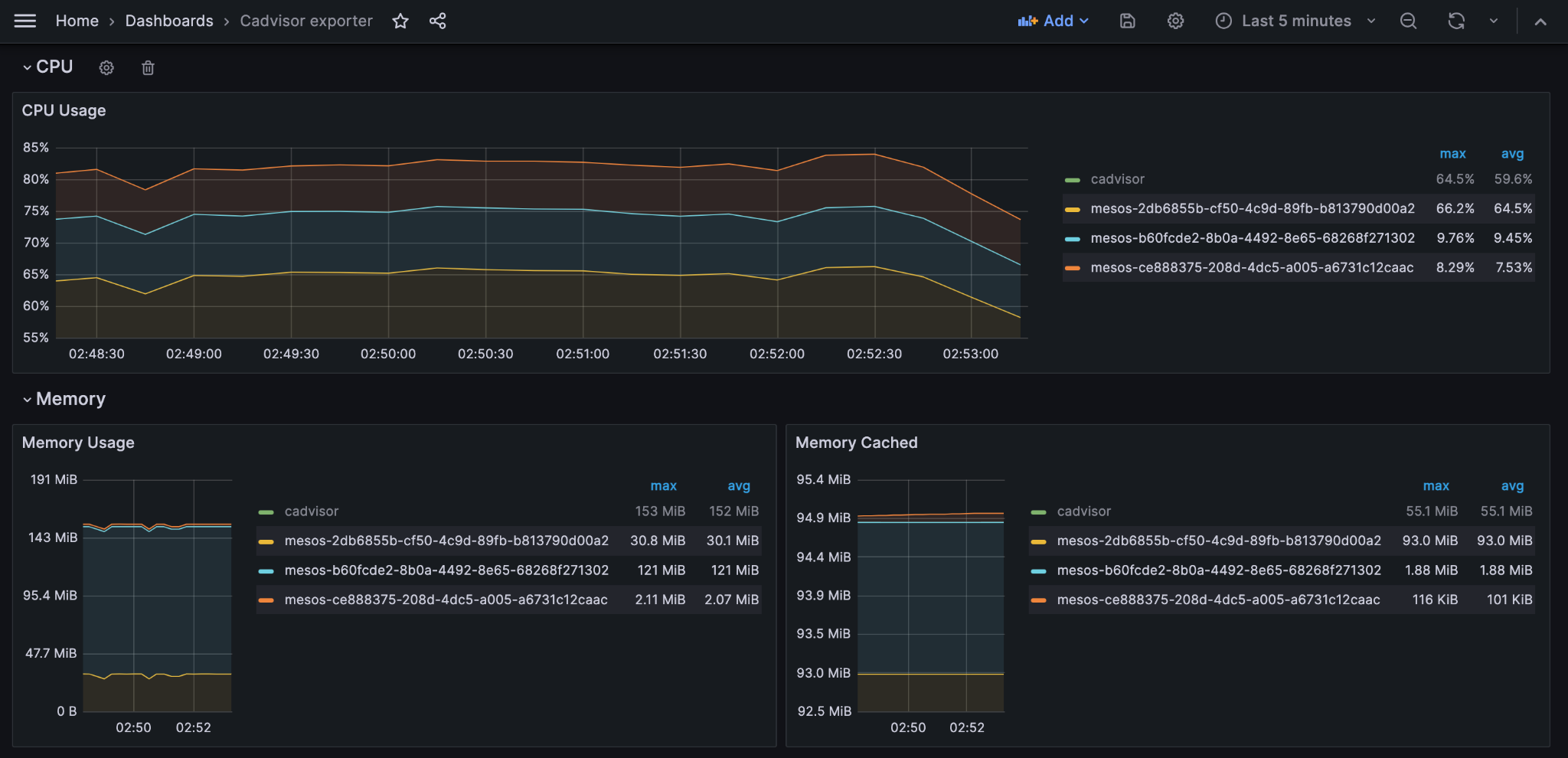
Mesos startup-time WordPress



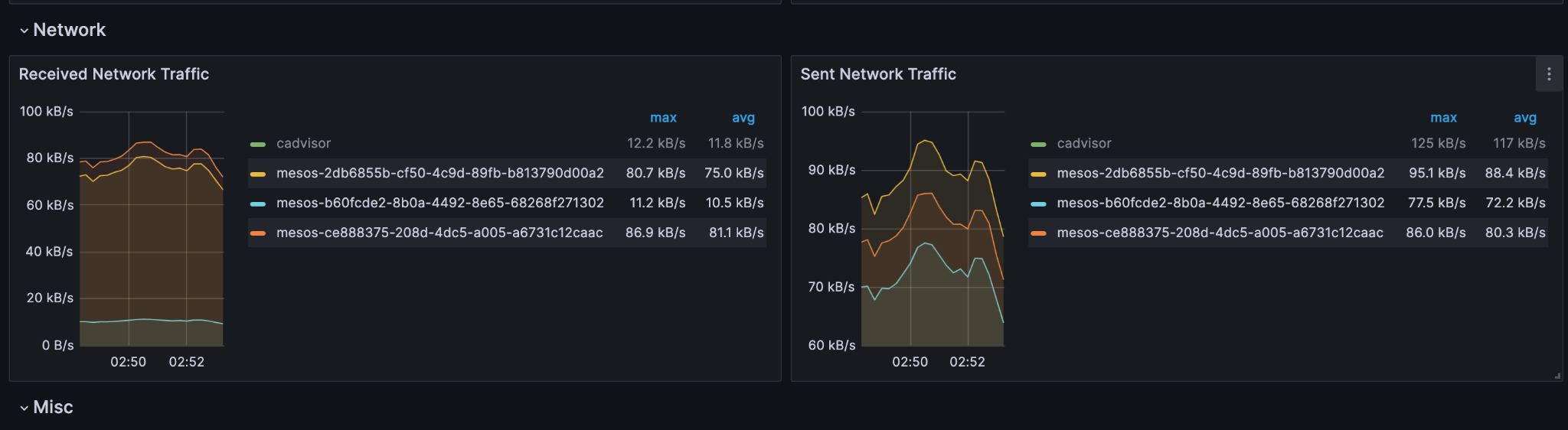
Mesos failover-time WordPress



Mesos failover-start WordPress



Mesos CPU and Memory (**WordPress**)



Mesos Network Traffic (**WordPress**)