Performance Analysis of Metric-Based Scaling in Kubernetes Environments

Focusing on Approaches Based on CPU Utilization and Custom Metrics

Department of Software Convergence, Jincheol Jung

Contents

Background

1. Need for Efficient Resource Management in Cloud Environments

- *The necessity of automated scaling to improve resource management in cloud environments*

2. Limitations of CPU/Memory-Based Scaling

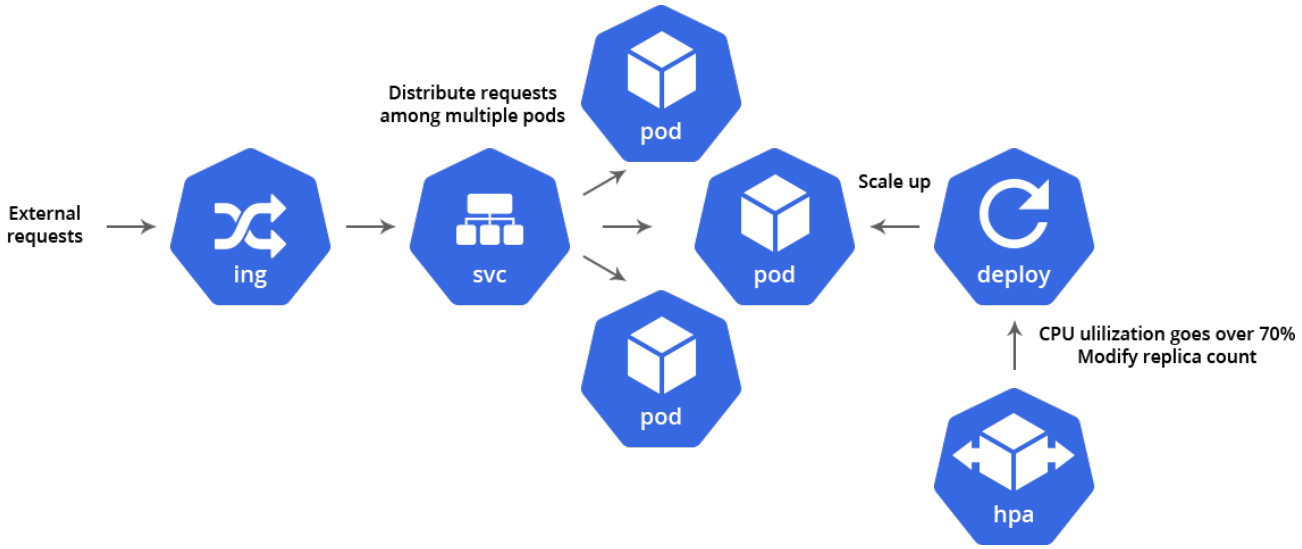
- *CPU/Memory-based scaling may not accurately reflect the actual load on the application, potentially leading to performance limitations*

3. Potential for More Efficient Scaling

- *By using metrics tailored to the specific needs of the application, more accurate and efficient scaling can be achieved*

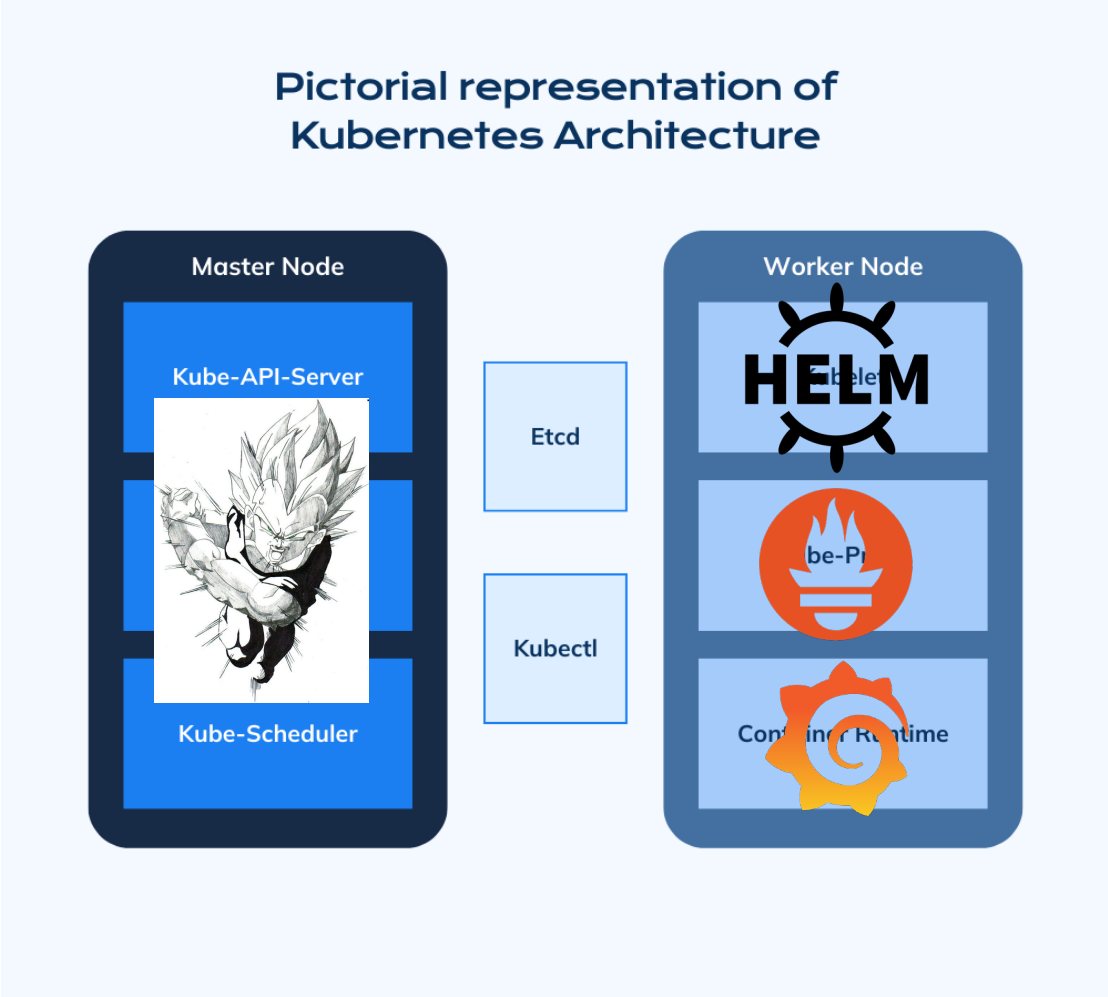
Objectives

Propose a customized scaling solution for applications in dynamic environments by comparing and analyzing scaling approaches based on CPU Utilization Metrics and Custom Metrics focused on network traffic



https://www.virtuozzo.com/company/blog/scaling-kubernetes/

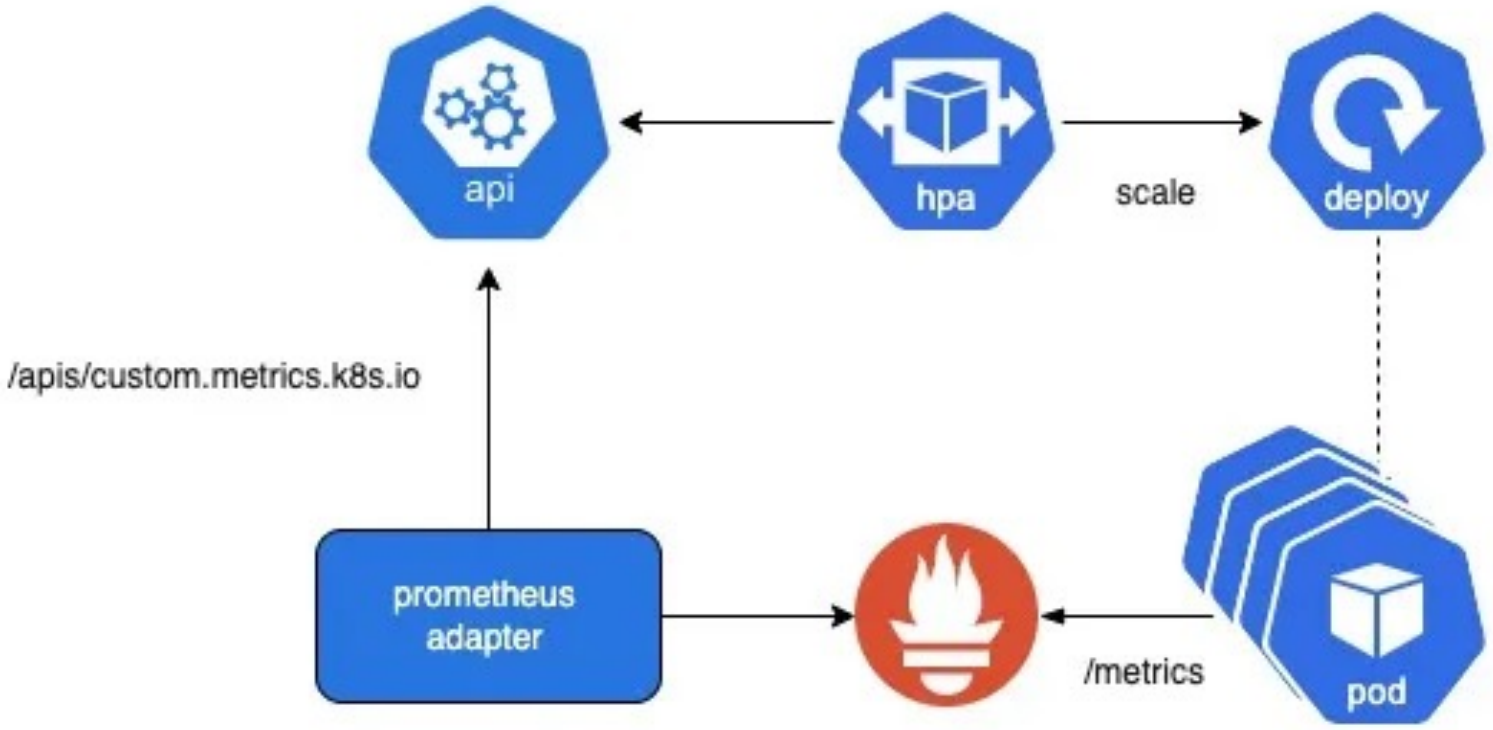
Methodology



https://www.squadcast.com/blog/kubernetes-simplified-understanding-its-inner-workings

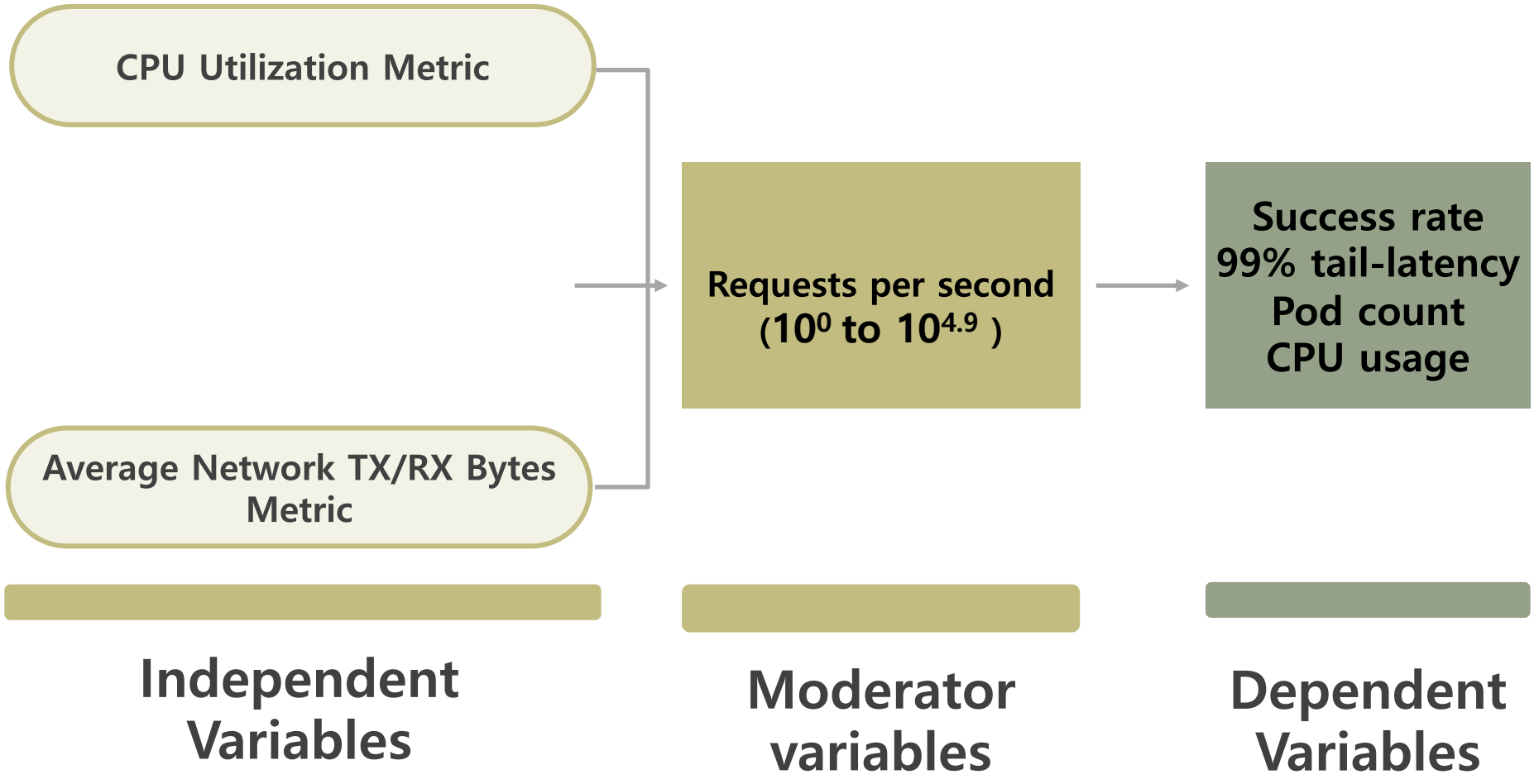
Methodology

Worker Node



https://chaitu-kopparthi.medium.com/scaling-kubernetes-workloads-using-custom-prometheus-metrics-1eb64b23919e

Methodology



Methodology

irate()

`irate(v range-vector)` calculates the per-second instant rate of increase of the time series in the range vector. This is based on the last two data points. Breaks in monotonicity (such as counter resets due to target restarts) are automatically adjusted for.

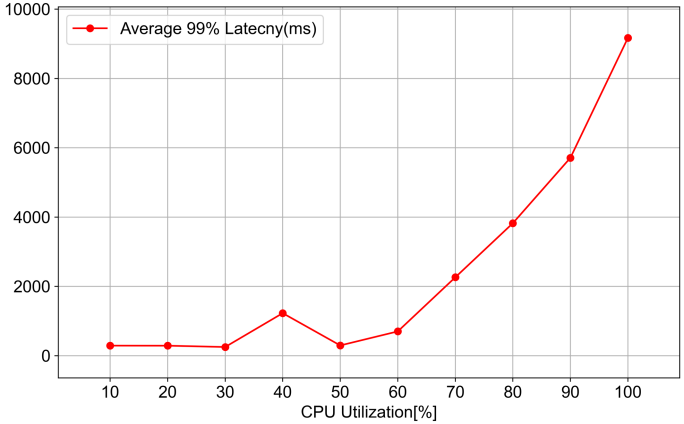
The following example expression returns the per-second rate of HTTP requests looking up to 5 minutes back for the two most recent data points, per time series in the range vector:

```
irate(http_requests_total{job="api-server"}[5m])
```

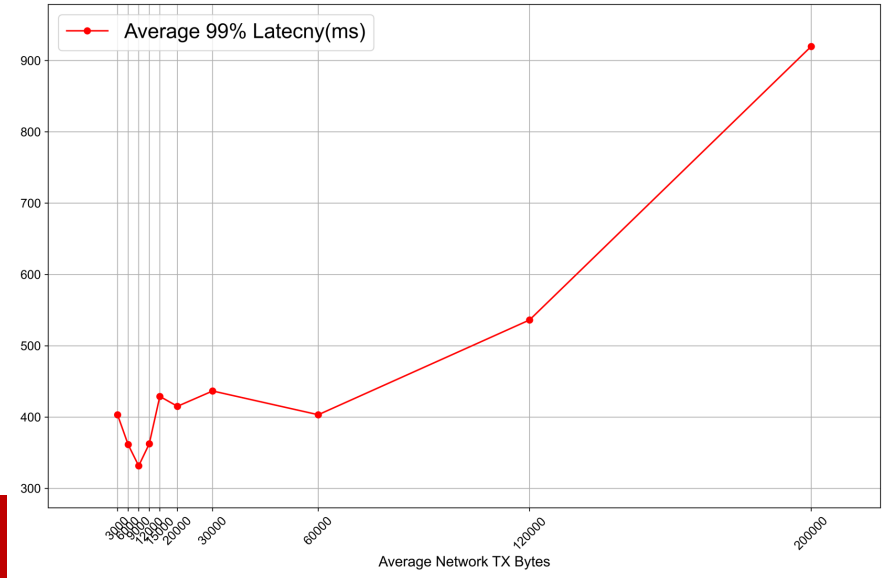
https://prometheus.io/docs/prometheus/latest/querying/functions/

Results

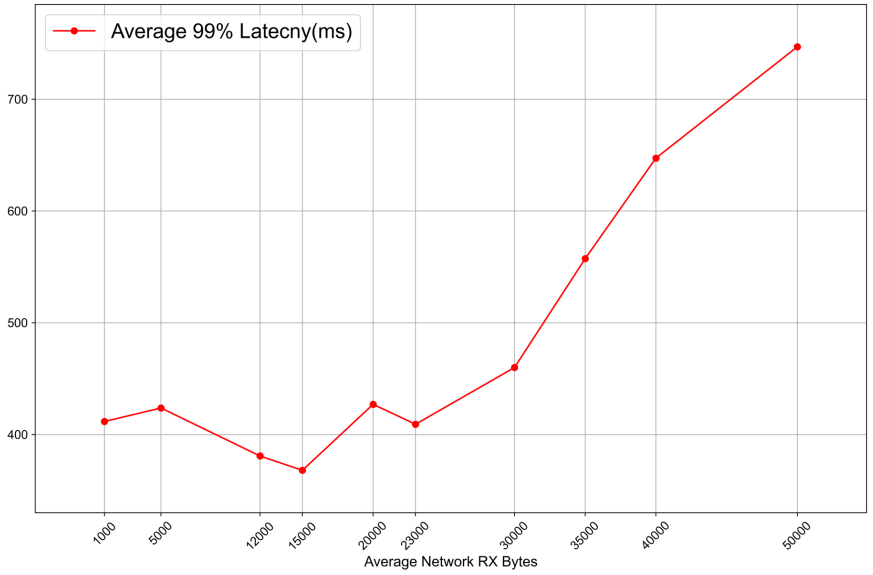
CPU Utilization Metric



Average Network TX Bytes

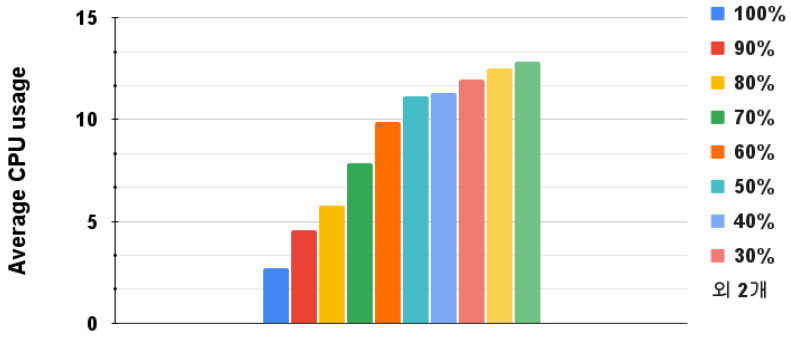


Average Network RX Bytes

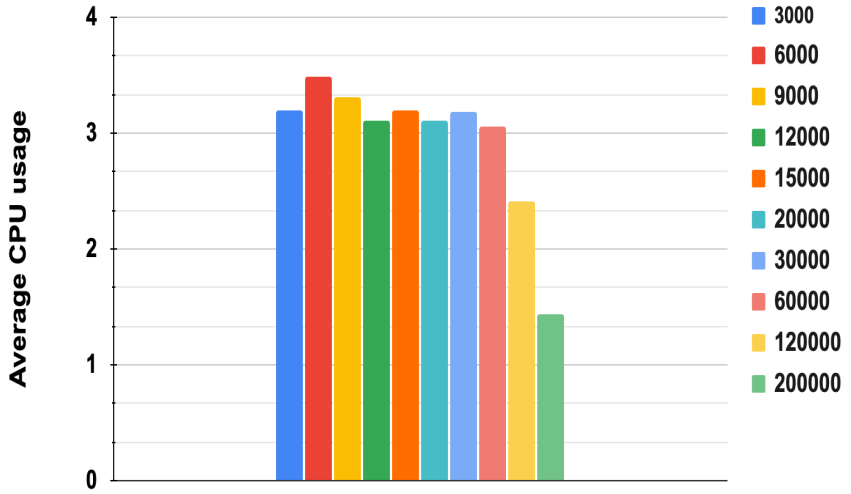


Results

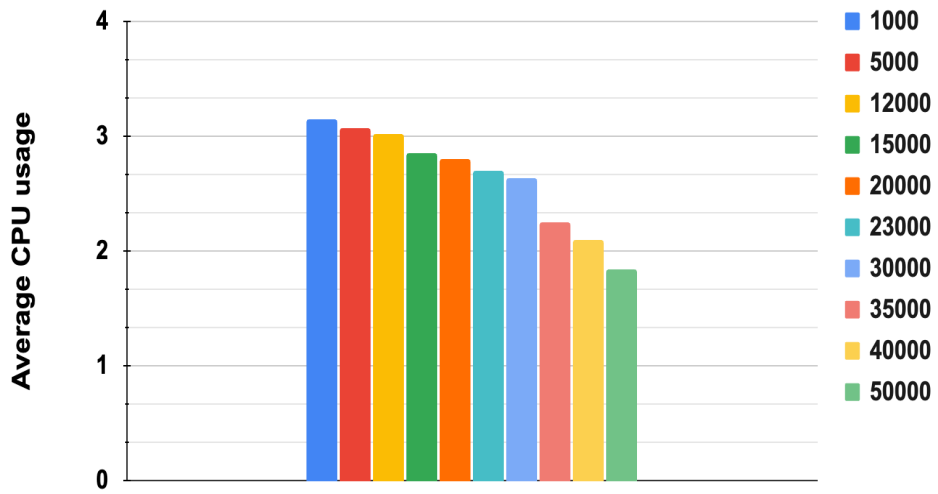
CPU Utilization Metric



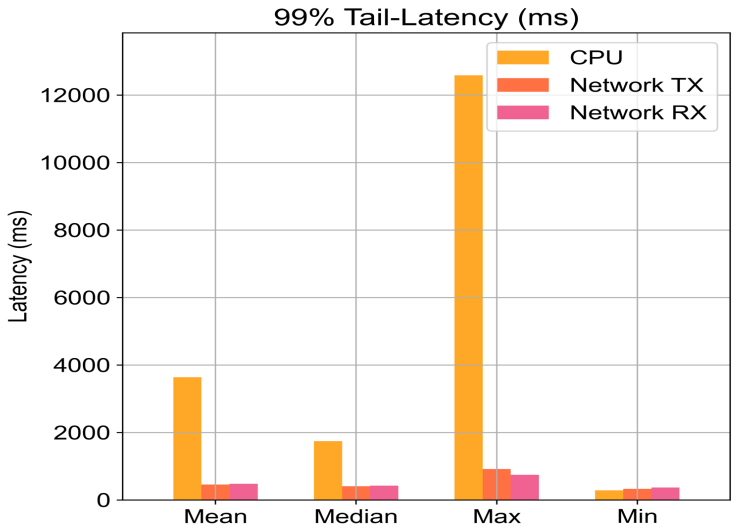
Average Network TX Bytes

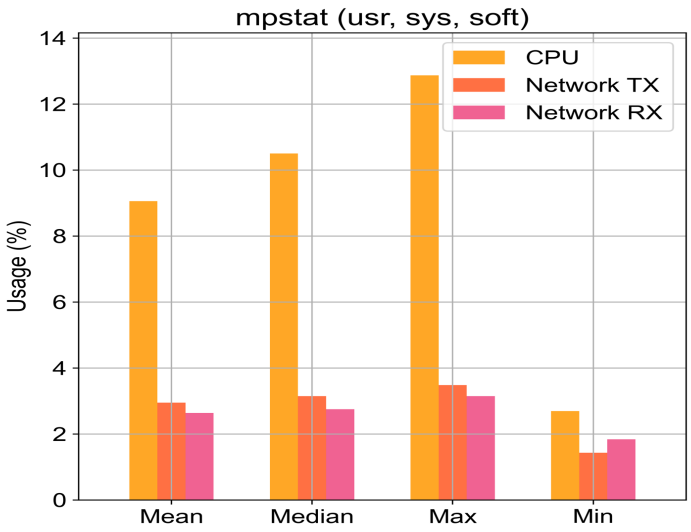


Average Network RX Bytes



Results





The average latency of the CPU usage metric is approximately 7.91 times higher than Network TX and 7.53 times higher than Network RX.
The maximum latency of the CPU usage metric is approximately 13.69 times higher than Network TX and 16.85 times higher than Network RX.

The average CPU usage of the CPU metric is approximately 3.07 times higher than Network TX and 3.43 times higher than Network RX.
The maximum CPU usage of the CPU metric is approximately 3.69 times higher than Network TX and 4.09 times higher than Network RX.

Conclusion

1. Application-Specific Scaling Solution: Custom metrics can provide a scaling solution that meets the specific demands of the application.

2. Performance Optimization: Contributes to providing stable and fast responses under dynamic workload conditions.

THANKS FOR YOUR ATTENTION