A Scheduler for Resource Allocation in Cloud-Edge Continuum

Oruç Berat Turan
oruc.berat@marun.edu.tr

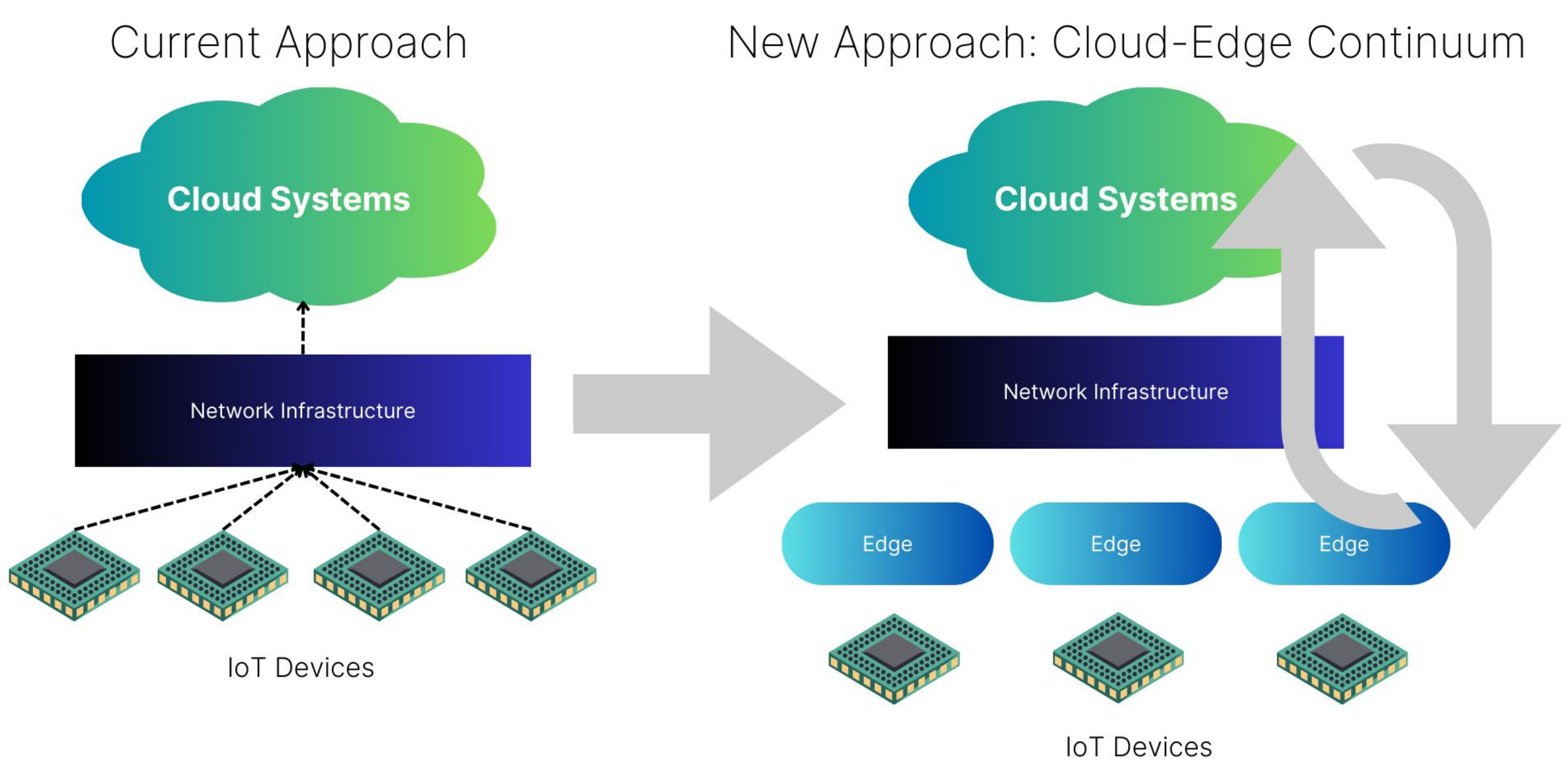
Anılcan Erciyes
anilcanerciyes@marun.edu.tr

Mehmet Akif Gülmüş
makifgulmus@gmail.com

VeNIT Lab
Vehicular Networking and Intelligent Transportation Systems Research Laboratory

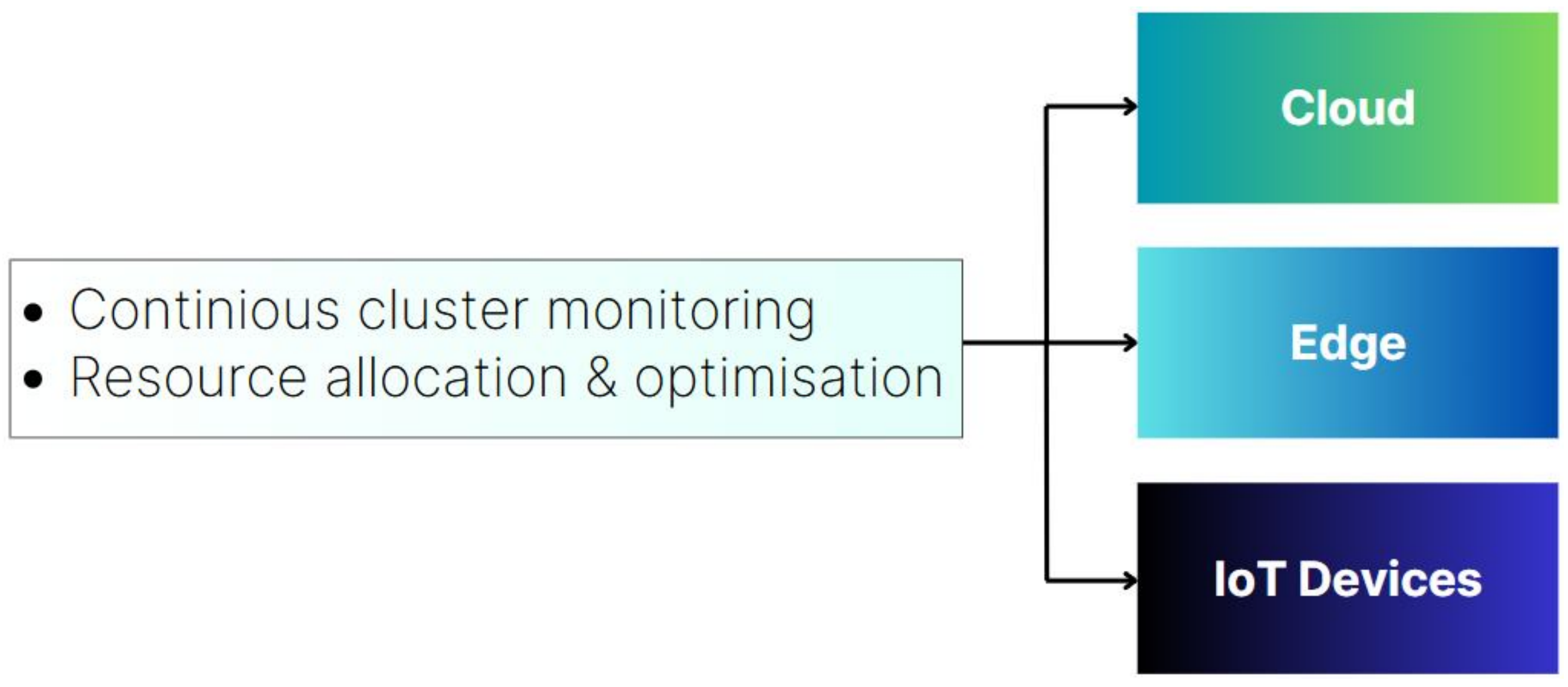
Advisor: Assoc. Prof. Müjdat Soytürk

Introduction



The Edge-Cloud Continuum introduces a novel approach to computing. Instead of solely relying on central servers, which can increase latency, now the data can also be processed closer to the source at the edge level. This enhances performance, privacy and flexibility, since processes can be selected to run on the cloud or at the edge, depending on current needs and requirements of the system. Such adaptable approach improves efficiency and responsiveness, making it ideal for applications requiring real-time data processing, such as IoT devices [1].

Solution Methodology



- Continious cluster monitoring
- Resource allocation & optimisation

1 - Employed **KubeEdge** to set up a physical edge computing cluster, located in Dragos campus, which comprises a control plane and several connected edge devices over a LAN switch [3].

2 - **Emulator** programmes are created for generating artificial workloads to simulate real-world conditions. In this regard, **stress testing** tools (i.e., stress-ng) are used. These programmes are then containerized with **Docker**, to be deployed on various nodes within the edge computing cluster with flexible and configurable parameters.

3 - To monitor resource usage metrics across the nodes of the cluster, **Prometheus** and **Grafana** are used [4]. The Prometheus server, integrated with the KubeEdge cluster, collects and exposes detailed performance data, which is also accessed by the **Python**-based scheduler script.

3 - The **scheduler** operates by periodically assessing the current resource utilization of each node. If the script detects that any node exceeds predefined workload thresholds (i.e., for CPU or memory usage), it initiates a process to **redistribute the workload** by redeploying certain pods to alternative machines within the edge computing cluster to prevent performance degradation & overload [5].

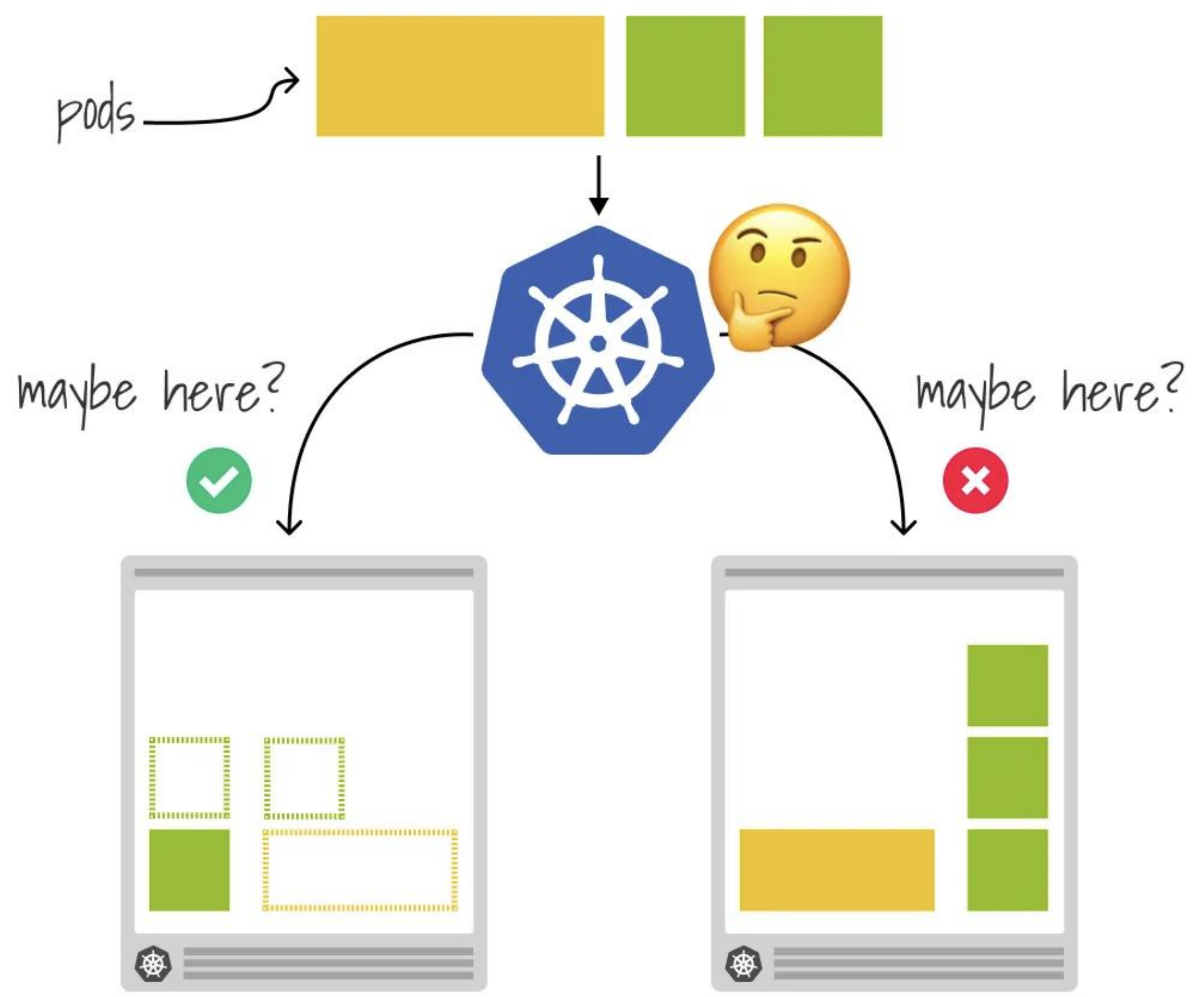
5 - Various **test cases** and scenarios are run to observe how system stability is maintained, enhancing the overall efficiency and reliability of the edge computing environment. Clear observations on how the scheduler can decide to transfer a pod to another node (depending on the current workload of the system) is made. Overall efficiency and balance is observed to be improved.

Problem Statement

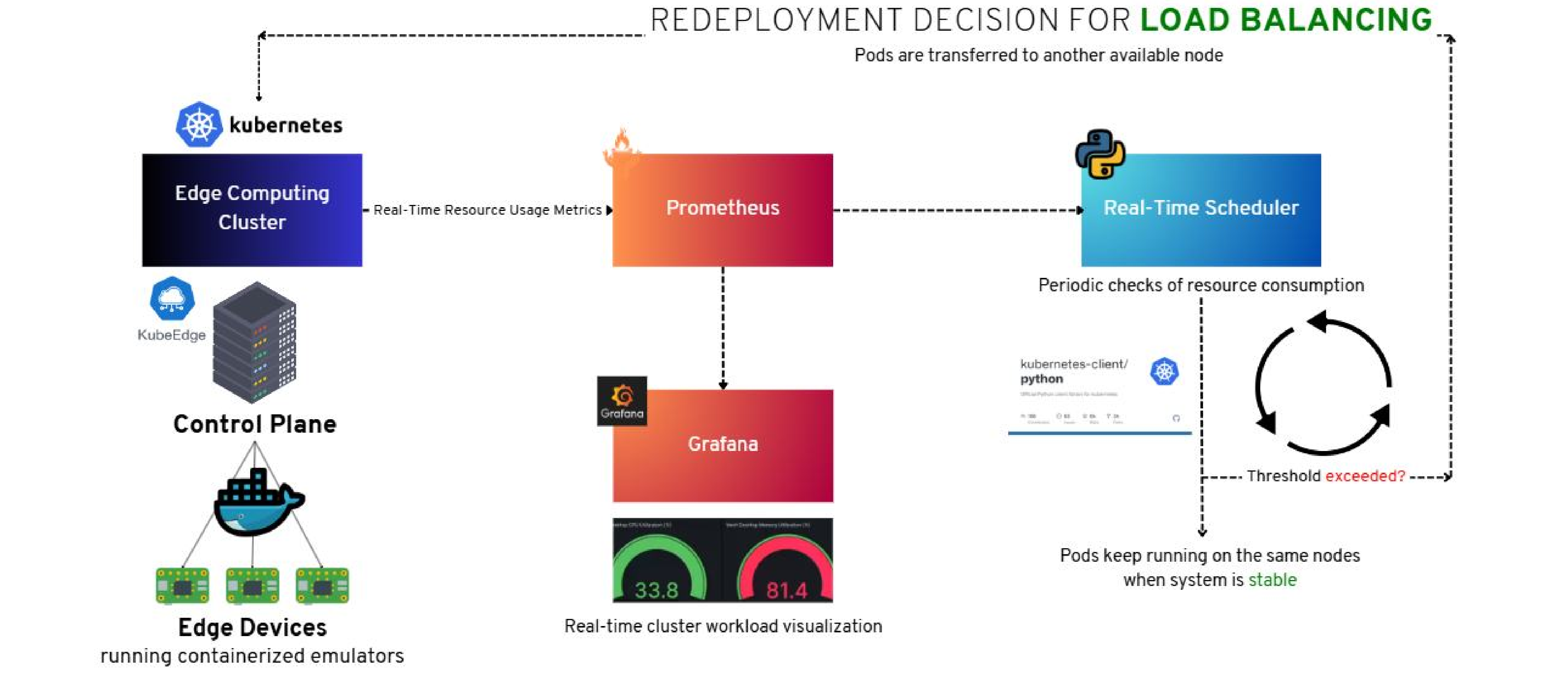
While the Edge-Cloud Continuum offers numerous benefits, efficient load balancing mechanisms are much needed in this context.

The selection of nodes for deploying new pods/ processes must prioritize efficiency to ensure overall system stability, a well-balanced load distribution, and low latency for applications [2].

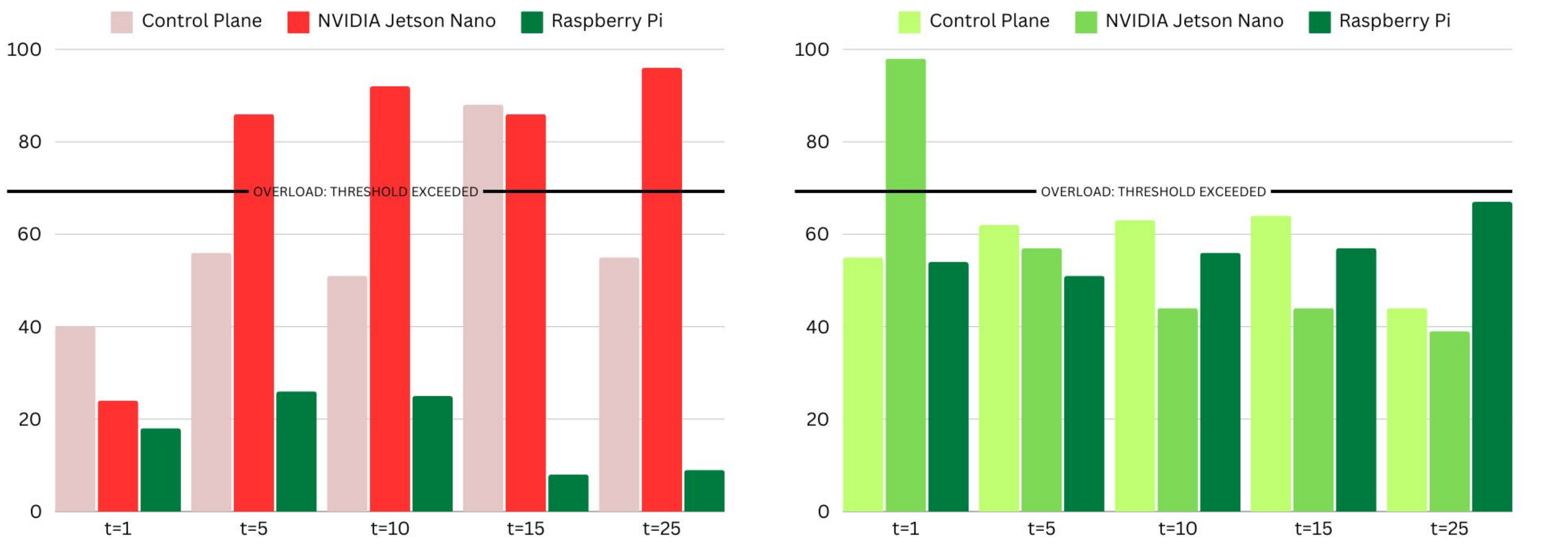
Therefore, deciding on which node to deploy an incoming process is crucial for maintaining the actual objectives and to fully leverage the advantages of a joint edge-cloud system.



High-Level System Architecture


REDEPLOYMENT DECISION FOR LOAD BALANCING
Pods are transferred to another available node
kubernetes
Edge Computing Cluster
Real-Time Resource Usage Metrics
Prometheus
Real-Time Scheduler
Periodic checks of resource consumption
KubeEdge
Control Plane
Grafana
kubernetes-client/ python
Threshold exceeded?
33.8
81.4
Pods keep running on the same nodes when system is stable
Edge Devices
running containerized emulators
Real-time cluster workload visualization

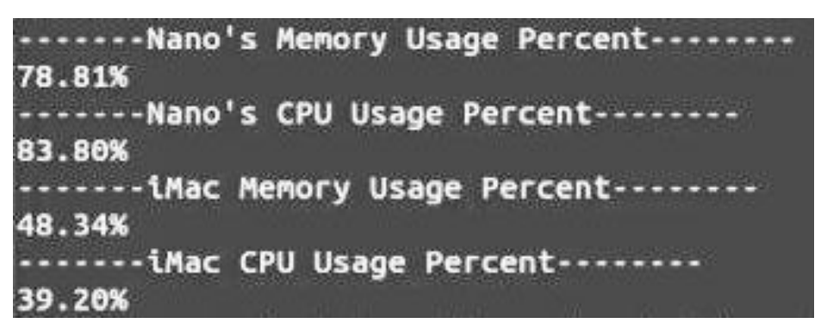
Experimental Results



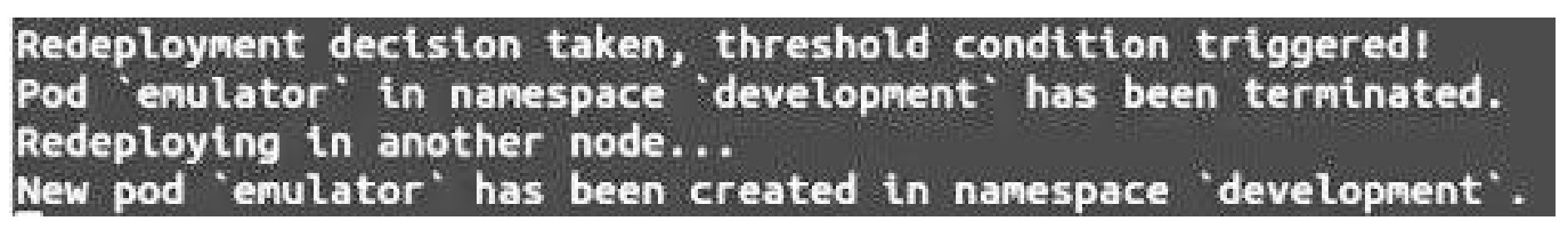
Charts: %CPU utilization on nodes, measured every 5 minutes

Without the scheduler
Imbalanced cluster workload
Poor computational efficiency
Application delays & latency
Higher risk of node failures

Scheduler added
Much better load distribution
Stable cluster performance
Constant, real-time optimisation
Pod redeployment capability



```
-------Nano's Memory Usage Percent--------
78.81%
-------Nano's CPU Usage Percent--------
83.80%
-------iMac Memory Usage Percent--------
48.34%
-------iMac CPU Usage Percent--------
39.20%
```



```
Redeployment decision taken, threshold condition triggered!
Pod `emulator` in namespace `development` has been terminated.
Redeploying in another node...
New pod `emulator` has been created in namespace `development`.
```

Conclusion

→ Our project successfully established a KubeEdge cluster to represent the edge-cloud continuum. Containerized stress testing applications were deployed on any desired machine for easily configurable workload generation.

→ A constant monitoring system was successfully built, and the resource consumption of each node was tracked and visualized in real-time for various metrics.

→ A threshold-based real-time scheduler were successfully implemented, enabling an optimized decision-making, with the ability to transfer a pod to more suitable node when an overload was detected. Throughout the tests, no single device in the model system bore more load than the predefined threshold of 70% resource capacity.

Technologies Used

kubernetes KubeEdge Prometheus Grafana docker

References

[1] Panagiotis Gkonis , Anastasios Giannopoulos , Panagiotis Trakadas, Xavi Masip-Bruin & Francesco D'Andria, A Survey on IoT-Edge-Cloud Continuum Systems: Status, Challenges, Use Cases, and Open Issues, MDPI, November 2023.
[2] Quang-Minh Nguyen, Linh-An Phan & Taehong Kim, Load-Balancing of Kubernetes-Based Edge Computing Infrastructure Using Resource Adaptive Proxy, MDPI, April 2022.
[3] Ying Xiong, Yulin Sun, Li Xing, Ying Huang, Extend Cloud to Edge with KubeEdge, IEEE, December 2018.
[4] Abhijeet Dhane, Atharva Joshi, Chinmay Borgaonkar, Manas Deshpande, Pravin Patil, A Survey on Deploying Prometheus and Grafana for Application Monitoring, JETIR, January 2024.
[5] Khaldoun Senjab, Sohail Abbas, Naveed Ahmed1, & Atta ur Rehman Khan, A survey of Kubernetes scheduling algorithms, SpringerOpen, June 2023.