Intent-based Scheduling Approach for Kubernetes-Oriented Cloud Systems

Ta Phuong Bac, YoungHan Kim*,
Soongsil University

bactp@dcn.ssu.ac.kr, younghak@ssu.ac.kr*

Abstract

Kubernetes has evolved into a well-liked solution for managing containerized workloads at scale, which uses a scheduler to decide which node is the best candidate to host a given task. Although flexible, the default Kubernetes scheduler must adapt to new applications, such as machine/deep learning workloads and network function workloads in the telco domain. Due to this, several proposals for custom Kubernetes schedulers have been made. However, these solutions still face many difficulties in automated deployment and quality assurance management of workloads based on the initial user specification. Therefore, this paper proposes a new approach to the scheduling problem in Kubernetes-Oriented Cloud Systems. We present the Intent-based Scheduling approach for automatic workload placement, configuration, and deployment. Ensures to keep user workloads always in the initial desired state and easily change with the conditions of the environment

Ⅰ. INTRODUCTION

The growth of containers has changed how users conceive software applications' development, deployment, and maintenance. From the cloud service providers, a CaaS platform creates an abstraction layer that includes a container orchestration engine, typically based on Kubernetes (K8s). K8s is a leading container orchestration platform widely used in cloud management due to its outstanding capabilities. In particular, the task of assigning physical resources to containers is committed by the scheduler, which focuses on improving resource utilization and satisfying users' application requirements. However, as a rule, designing efficient container scheduling techniques in Kubernetes is still an open issue[1]. Due to this, several proposals for custom Kubernetes schedulers have been made, and most of them employ one of the following Kubernetes-native mechanisms[2]to perform their custom scheduling behavior.

- Annotation and Labels: can be used to add custom information to Pods or Nodes for enforcing certain constraints in the scheduling logic.
- Extent resources: advertising node-level resource
- Custom Resource Definitions: creating a new API object for scheduling tasks, such as scheduling a group of pods together in co-scheduling.
- DevicePlugin: when scheduler has to deal with vendor-specific hardware setups such as GPU or network devices.

However, these solutions still face many difficulties in automated deployment and quality assurance management of workloads based on the initial user specification. For example, complicated configurations are implemented manually, an obstacle in enterprise service management. In addition, current custom scheduling methods in k8s lack the application's "desired state" to guarantee initial user specification and day2-reconfiguration.

Therefore, in this study, we propose a new approach to the scheduling problem, an Intent-based [3] scheduling approach for Kubernetes-oriented cloud systems. The intent-based scheduling approach can simplify enterprises' configuration and deployment service workload, which aims to automate the deployment configuration by translating a high-level and abstract service request into a detailed policy description. In this paper, we consider the scheduling problem defined as *"Placement Intent."* Placement intents include tasks related to determining where to deploy user workloads as deployments in a specific or several k8s clusters. Along with that is the guarantee of satisfying the initial conditions of the user for the particular application. For instance, *PlacementIntent* is "*I want to run workload A (machine learning) in a cluster, with a GPU available, and it needs at least x amount of CPU/Memory available for serving.*"

Integration with Intent-based management that enables the human-readable configuration file and automated configuration to reduce human invention on the system. Through a system of Controller and Intents that can handle complex structure and placement applications on k8s cluster in a geo-distributed environment. The following section demonstrates the details of our approach.

Ⅱ. INTENT-BASED SCHEDULING APPROACH

This section presents the architecture for our proposed intent-based scheduling approach. The system's main components include Scheduling Intent Controller, Placement Controller, and a *PlacementIntent* Custom Resource in the k8s cluster.

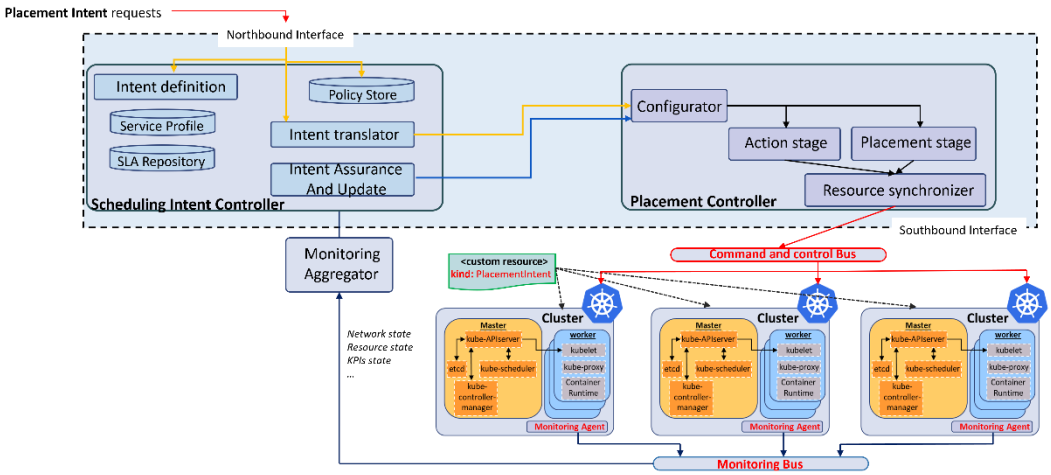

Figure 1 Reference Architecture of proposed approach

As depicted in Figure 1, we create *PlacementIntent* resource as a custom resource (CR) in Kubernetes responsible for automatically deploying and config the other resources (deployment, service, etc.) that are related to the service workload of enterprises, and following by the desired state generate from placement intent request. *PlacementIntent* CR continuously monitors and updates the state of service workload deployed for maintaining the desired state of intent through two controllers. Placement Controller and Scheduling Intent Controller are responsible for managing *PlacementIntent* CR, continuously watching the *PlacementIntent* CR's state to validate whether the target workload is in the desired state, as depicted in Figure 2.

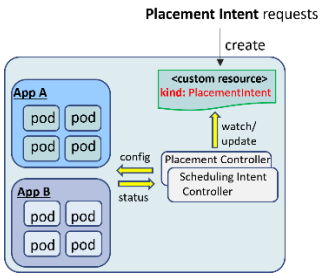

Figure 2 Architectural view of Controllers with PlacementIntent CR

The Scheduling Intent Controller acts as a handler for intent-related tasks, including defining and formatting the structures of placement intents through the Intent Definition. Information about the application, specific parameters, and policies in the system are stored in the Service Profile, SLA repository, and Policy Store. This information will serve the intent resolution process performed by the Intent Translator into metadata that is input to the Placement Controller configurator.

The Placement Controller plays a role in implementing and implementing requests and information from the Configurator for executing the initial intent (placement stage) and updating and reconfiguring the parameters of the related resources. (Action stage)

These processes are described as the beginning of implementing a new intent. After the *PlacementIntent* CR is successfully deployed along with other service workload resources, their state will be continuously monitored and updated to the Intent Assurance and Update. Intent Assurance will automatically update, adjust and reconfigure to the PlacementIntent CR if the state is unsatisfied.

IV. CONCLUSION

In this study, we present an intent-based scheduling approach for a Kubernetes-oriented cloud system. That support deploys the *PlacementIntent* as a custom resource in Kubernetes, responsible for handling complex configuration and placement needs through Placement Controller and Scheduling Intent Controller. By relying on the Intent-based approach, the application definition is solely based on a desired state configuration, which brings minimalism and ease in service management and operation to the enterprise. In the future, we will discuss the implementation strategy of the proposal and evaluation.

ACKNOWLEDGMENT

This work was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2022-0-01015, Development of Candidate Element Technology for Intelligent 6G Mobile Core Network).

REFERENCES

[1] Rejiba Z, Chamanara J. Custom scheduling in Kubernetes: A survey on common problems and solution approaches. ACM Computing Surveys. 2022 Dec 15;55(7):1-37.

[2] Carrión C. Kubernetes scheduling: Taxonomy, ongoing issues and challenges. ACM Computing Surveys (CSUR). 2022.

[3] Sharma Y, Bhamare D, Sastry N, Javadi B, Buyya R. SLA Management in Intent-Driven Service Management Systems: A Taxonomy and Future Directions. arXiv preprint arXiv:2208.01218. 2022 Aug 2.