January 2024

TECHNIQUES TO FACILITATE INTELLIGENT MEC APPLICATION DEPLOYMENT FOR PRIVATE 5G ENTERPRISE NETWORKS

Robert Barton

Matthias Falkner

Indermeet Gandhi

Jerome Henry

Vinay Saini

TECHNIQUES TO FACILITATE INTELLIGENT MEC APPLICATION DEPLOYMENT FOR PRIVATE 5G ENTERPRISE NETWORKS

AUTHORS:
Robert Barton
Matthias Falkner
Indermeet Gandhi
Jerome Henry
Vinay Saini

ABSTRACT

Private 5G networks are emerging that allow a cellular network to be deployed and operated as an extension of an enterprise network. The deployment of such a private 5G cellular network typically involves the integration of cellular network components with exiting elements of an enterprise network, such as using the existing network to implement fronthaul, midhaul, and backhaul network connectivity. A key advantage of 5G cellular connectivity is the ability to offer edge compute resources to user equipment through the use of multi-access edge compute (MEC) deployments. However, in an enterprise network, edge compute resources may be scarce. In order to address such issues, novel techniques are proposed herein that provide for a MEC deployment controller that enables the deployment of enterprise edge applications to a radio access network of an enterprise network.

DETAILED DESCRIPTION

Private 5G networks are providing new opportunities to extend enterprise networks to facilitate the delivery of enterprise services to users through cellular network connectivity. Further, the deployment of edge compute resources in multi-access edge compute (MEC) deployments provides for the ability to implement enterprise applications and services closer to/within radio access networks (RANs), thereby offering service level improvements (e.g., latency, throughput, etc.) over applications and services that may be implemented in a centrally within an enterprise network.

In a public cellular network, a service provider may deploy compute clusters (MEC resources) throughout a RAN, such as anywhere from a centralized unit (CU) of the RAN to a distributed unit (DU) of the RAN, or anywhere in between. However, in an enterprise

network, edge compute resources may be scarce. While they may exist in limited availability on edge switches and/or on industrial switching and compute platforms, the performance of such enterprise resources is often limited, making their use difficult for enterprise application deployment.

In order to address such issues, a MEC deployment controller, referred to as a Smart Edge Application (SEA) MEC controller, is proposed herein that can enable the deployment of enterprise edge applications to a RAN of an enterprise network. The SEA MEC controller may be similar to a lightweight K8s®/K3s® orchestrator, but with added intelligence that may facilitate the deployment of MEC applications (apps) within a private 5G network using enterprise network resources or stated differently, may deploy application containers to compute resources that reside in an enterprise network (e.g., switches, routers, etc.). Kubernetes®, K8s®, and K3s® are registered trademarks of the Linux Foundation.

For an implementation involving the SEA MEC controller, an administrator can load application containers to the controller, along with descriptors for each application. Various applications that may deployed may include edge firewall apps, Internet of Things (IoT) brokers/data processors, industrial security apps, telemetry/full-stack observability (FSO)/performance analysis apps, and/or any other containerized edge applications.

Once loaded with applications to be installed, the SEA MEC controller can assemble an inventory of available compute resources within the enterprise network. This may include compute resources on a switch (e.g., central processor unit (CPU) resources, memory resources, storage resources, etc.), industrial compute servers (e.g., International Joint Conference on Knowledge Discover, Knowledge Engineering, and Knowledge Management (IC3K) resources), or any other resources in the enterprise network. The SEA MEC controller can determine an inventory of the resources and their compute capabilities found on each device on the network using a network inventory system, a network management application, or the like.

5G endpoints are assumed to want to consume the edge applications in accordance with a certain service level agreement (SLA)/Quality of Experience (QoE). An administrator can create policies to reflect the consumption demands from the 5G endpoints. Such policies may include assigning specific edge applications to user

equipment (UEs), groups of UEs, per-Data Network Name (DNN), 5G radio nodes (e.g., gNBs), or to all the endpoints in a RAN. Such policies may allow the SEA MEC controller to know the general proximity where to deploy a specific MEC app.

Following policy definition, the SEA MEC controller can create a mapping of applications to available edge compute resources (mapping where the applications will be deployed). This mapping is intended to provide optimal resource targets for the edge apps used by the 5G network under a set of conditions. In some cases, the edge app may also involve deploying a mini-user plane function (mini-UPF) in order to offload traffic to the application. A mini-UPF may be needed in scenarios in which a default UPF is provided deep within an enterprise network, but an application is to be installed at an edge resource. Thus, application requirements should also be considered when compute resources are examined (e.g., determining when an app plus UPF functionality may need to be deployed on an edge compute resource).

For each application managed by the SEA MEC controller, an edge preference attribute can be assigned by the administrator indicating a preference regarding the proximity between each application and the UE that are to consume the application. For example, certain apps such as digital experience monitoring apps may need to be deployed as close to the extreme edge as possible in order to support network telemetry characteristics. Other apps, such as IoT edge intelligence apps (e.g., a Message Queuing Telemetry Transport (MQTT) message broker) have less dependency and, thus, may be deployed at a resource higher up in the hierarchy (perhaps in the midhaul or backhaul areas of a network). In this manner, apps can be provided an edge preference attribute such as: (1) Extreme Edge Mandatory, (2) Extreme Edge Preferred, (3) Extreme Edge Best Effort, or variations thereof.

Additionally, apps can be assigned a dependency index, for example, indicating that some apps are to be used before other ones in a logical order of operations. A dependency map can be used influence where the compute resources are deployed, how apps are linked, and how close to the edge each is situated. Such a dependency map may be particularly useful for situations in which network functions have an obvious service-chain type of dependency.

Once the various policy information and mappings are completed, the SEA MEC controller can begin orchestrating apps to the available compute resources and, if needed, providing a UPF instance to front-end the apps. Accordingly, apps can be deployed according to a hierarchy of available resources according to their edge preference attribute, dependency requirement, and desired latency objective, thereby facilitating a proper hierarchy of edge applications to be deployed. Once deployed, apps that have a dependency on each other may require a second network connection such that the SEA MEC controller can connect them using a service mesh (e.g., Istio) that allows direct communication between the deployed applications.

In an illustrative example of techniques proposed herein involving the SEA MEC controller, consider that some apps may not need to be deployed at the extreme edge of a network (e.g., they may have a lower edge preference ranking that may be trumped by another application that must be run at the edge, such as a security app). On other hand, some apps do not have requirement to be directly at the edge (such as an MQTT message broker) and, thus, may be deployed deeper in the network. In accordance with the techniques proposed herein, the SEA MEC controller may facilitate a logical framework to identify an app's edge priority/preference and inter-app dependency, which can then used by the orchestrator to deploy the apps appropriately. Thus, a framework is provided that facilitates deploying appropriate applications at appropriate network locations and, when app dependencies exist, ensuring that the apps are deployed to the edge in a hierarchical manner that reflects how the applications need to function.

By considering different application requirements, such as latency and priority, as objective functions, which directly play a role in deciding which apps get deployed on a constrained set of resources, the SEA MEC controller can efficiently deploy apps throughout a MEC network. For example, if an app has a higher edge priority than another app, but edge resources are very limited, the controller can determine to deploy the app with a higher priority vs. a competing app that may still be accessible in the cloud.

Such prioritization of edge app deployment in a constrained environment and mapping deployment to a physical topology, along with the logical dependency mapping are novel enhancements to the capabilities of conventional network controllers.

Accordingly, the multifaceted approach to edge app deployment involving the SEA MEC controller may involve several operations including creating a mapping of applications to available edge compute nodes based on the edge resources, app requirements, app dependencies, and edge preference. Descriptors can be attached to apps to help the controller interpret how close an app is to be deployed to an edge resource and an edge preference identifier can be defined for apps, allowing the controller to provision apps with a higher edge preference closer to the edge and those with a lower preference deeper in the network. In some instances, an app dependency index or dependency hierarchy can be mapped to physical compute nodes, allowing the controller to deploy the apps is a similar fashion. Thus, the controller can effectively create a new edge compute topology model based on values derived from the (1) edge preference (2) app descriptor, and (3) app dependency index. Finally, the controller can facilitate the orchestration of network components that reflect the edge app dependency hierarchy and, if needed, facilitate a service mesh to facilitate connections between dependent applications.

In summary, an SEA MEC controller can be provided in accordance with this proposal in order to deploy light-weight edge compute software/applications to constrained compute nodes (such as running on an X86 or ARM processor in an edge switch, router, or other nearby resource). The controller may provide a logical mechanism through which edge apps can be deployed considering their dependency on each other and affinity to how close to the edge they should be deployed; thus, providing is a new logic / intelligence layer that helps deploy edge applications to appropriate edge nodes in an appropriate hierarchy. Rather than deploying apps haphazardly through MEC networks, the dependency logic as discussed herein can enable the SEA MEC controller to facilitate an accurate deployment of applications within MEC networks.