Evaluating the Next Generation Sidecar-less Kubernetes Service Mesh: Ambient Mesh

Anupam Saha
School of Enterprise Computing and Digital Transformation, TU Dublin, Ireland
X00193210@myTUDublin.ie

Introduction

A service mesh is a dedicated infrastructure layer in the Kubernetes cluster to make service-to-service communication safer, faster, and more reliable. Typically, service meshes are built with a two-tier architecture, housing a control plane for mesh configuration and a data plane to provide mesh functionalities. The data plane consists of a sidecar container that attaches itself to a microservice pod, the smallest deployable unit in Kubernetes. Though this sidecar container architecture reduces a lot of burden on microservice source code, it also increases the overall compute resource usage of pods. To counter this, a new data plane architecture called ambient mesh was born. Instead of running a sidecar container in each microservice pod, ambient mesh implements the data plane by deploying a single proxy per node. This utilizes the Linux eBPF technology and brings a lot of excitements in service mesh space. Ambient mesh was originally developed by Solo and Google, but now it is part of the open-source project Istio. This research explores the ambient mode that comes with the newer version of Istio to evaluate its compute resource usage and operational complexities.

Objectives

RQ1. Does Istio ambient mode consumes less compute resources than sidecar mode?

RQ2. Does the sidecar-less architecture reduces operational complexities?

Motivation

With the release of ambient mesh alpha version as part of Istio, a massive improvement in compute resource consumption is reported over sidecar architecture in multiple gray literature, however, no academic research paper is available. While most of the service mesh products available today either use Envoy or in-house-developed sidecar proxies to determine the success factor, ambient mesh applies a completely different approach by leveraging Linux eBPF technology and separating layer 4 and layer 7 capabilities. While eBPF gives the ambient mesh to improve its resource efficiency, two separated layers allow Kubernetes administrators a different level of flexibility while deploying Istio in their environment. All of these bring an exciting opportunity to evaluate and publish an academic research report on ambient mesh.

Methodology

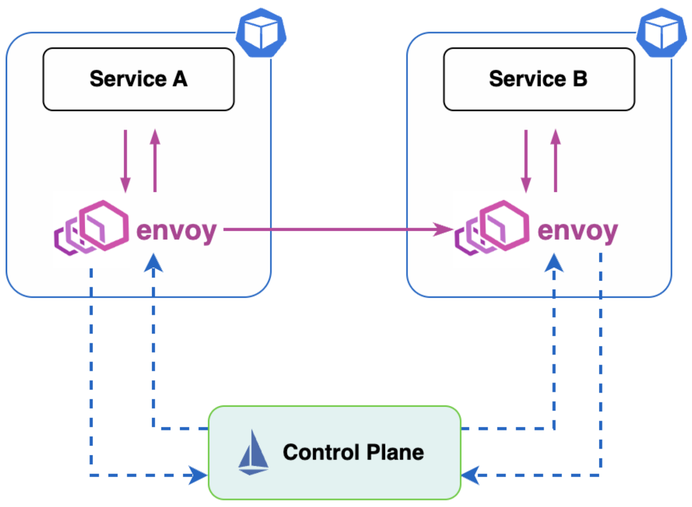


Figure 1: Istio Sidecar Mode Design

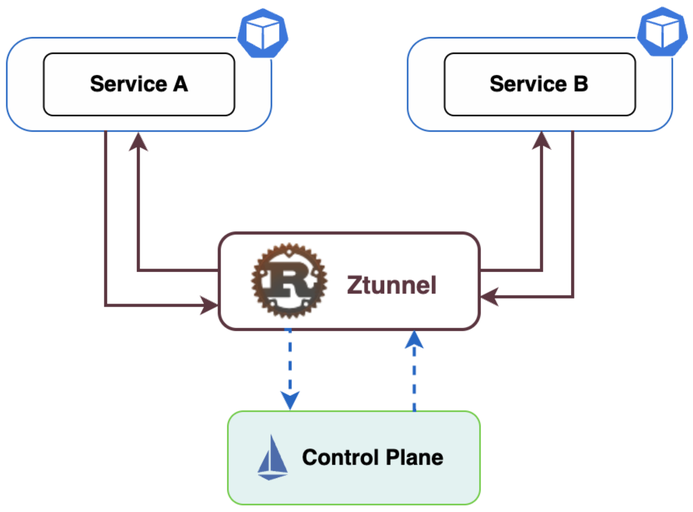
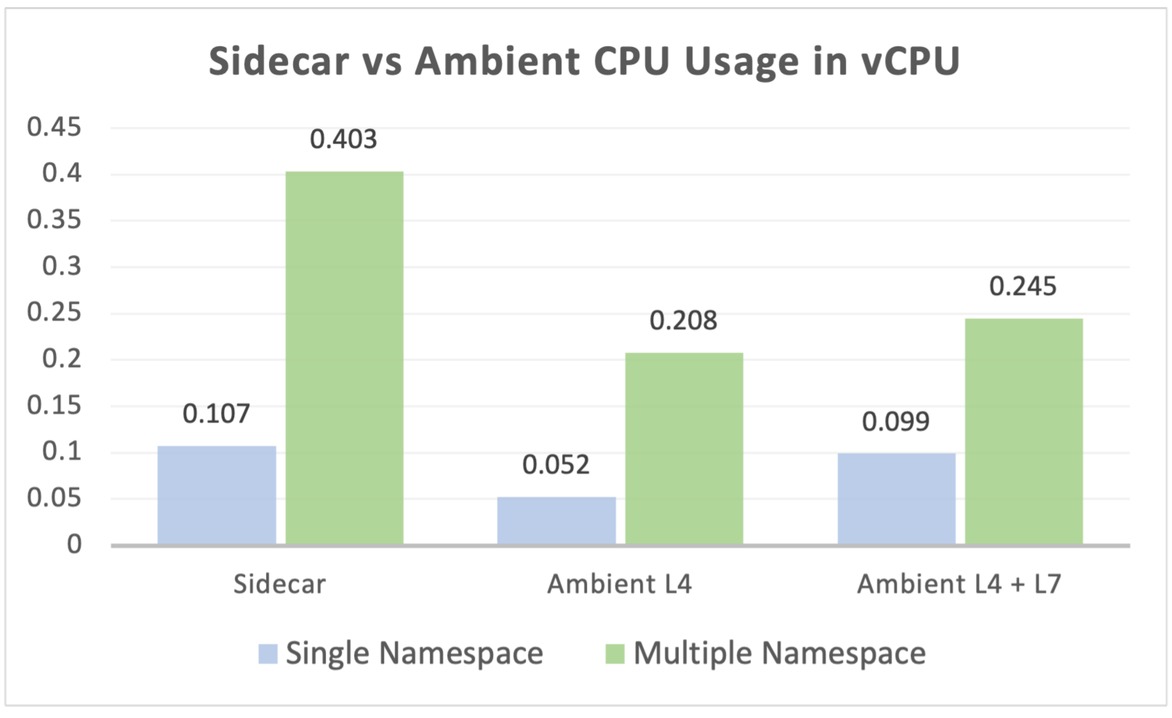
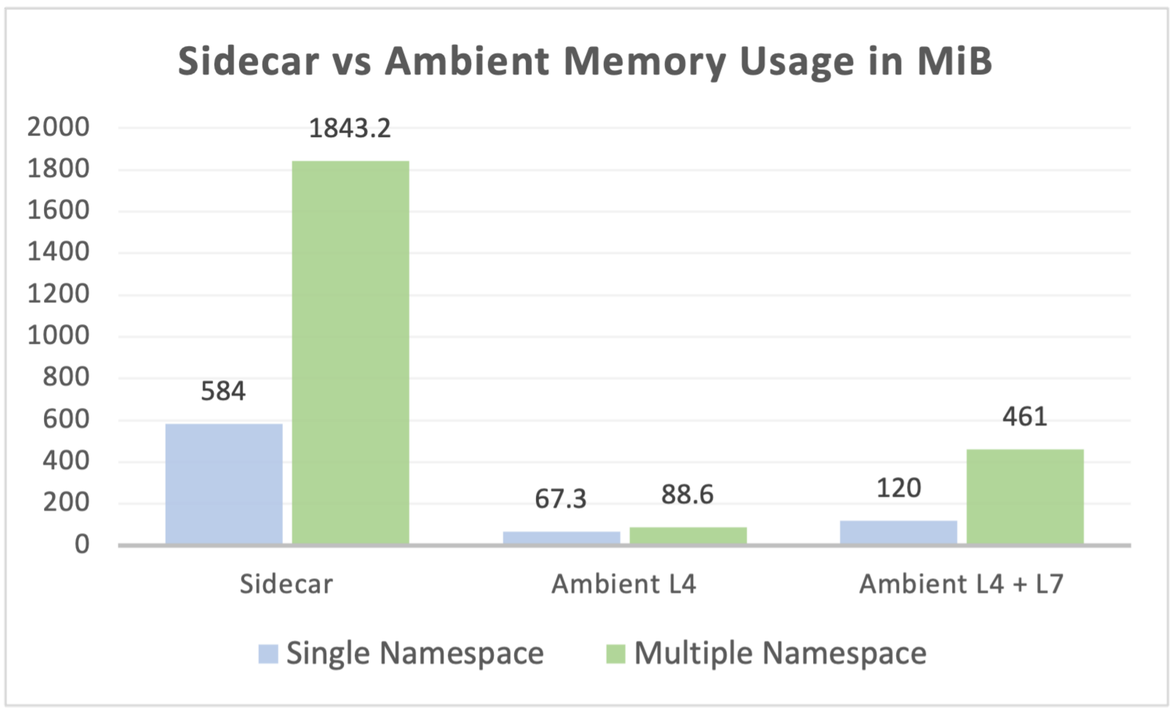


Figure 2: Istio Ambient Mode Design

A research platform is built to investigate the compute resource consumption by Istio sidecar and ambient modes after exploring the ambient mesh architecture. In sidecar mode, a tight-coupled architecture is seen where envoy proxies are attached to each running service pod, as shown in Figure 1, whereas ambient mode deploys a single Ztunnel proxy per node, as shown in Figure 2. The research is based around the principles of having multiple test executions with Ztunnel and Waypoint proxies engaged in ambient mode and Envoy proxies attached to each pod where a demo application, Bank-of-Anthos (BOA), is deployed to a Kubernetes cluster. To follow an enterprise-grade deployment model, BOA is deployed in single and multiple Kubernetes namespaces to explore the intensive nature of Istio in both modes. Google Cloud Platform is used as a cloud provider for the research, where Kubernetes clusters are provisioned using Terraform and Istio is installed using Istioctl and Istio operator. To engage Istio in resource-intensive filtering, 10-minute load testing is performed on BOA for each test session, along with layer 4 and layer 7 traffic filtering. Test results are captured from the Grafana dashboard by applying Prometheus metrics for CPU and memory. To measure operational complexity, Istio is upgraded using a blue-green deployment strategy to check whether a pod restart is required to apply the new version of Istio to the running microservices.

Research Findings





Conclusions and Future Work

While the sidecar-less model of ambient mesh brings some significant improvements in compute resource consumption and operational excellence, the sidecar model of Istio is well established and widely used today. In near future when a stable version of ambient mesh is released, this scenario shall be changed and ambient mode may become the default installation mode for Istioctl. Leveraging Linux eBPF technology, ambient mesh brings a lot of opportunity in service mesh space for future research. Some of which can be categorized as Istio's network latency due to Ztunnel and Waypoint proxies distributed nature and supporting Kubernetes Jobs or server-send-first protocols features which are currently not supported by Istio sidecar mode.

QR Code for Recording