Application Disaggregation & "Edgeification"
Do you have the freedom?

An Avesha Vision Whitepaper

"Do we need a purpose-built physical network in order to put your applications closer to the customer?" The answer from our perspective is "NO!"

Disaggregation is the term used to describe the stateless instantiation of resources on demand to support auto-scaling. When applied to applications, disaggregation is the parallel execution of micro-services and VMs to allow flexibility of deployment to the edge (what we call "edgeification") to enable *Application Intents* such as latency-controlled services, edge inferencing/acceleration, redundancy, resilience and governance — all needed to improve end user-experience and better customer experience.

Yet today, this flexibility is largely not realized because of the visibility and complexity of the underlay network state leading to an inertia towards disaggregation, emanating also from the fear of the unknown, involvement of various disciplines (full stack devs, netops, etc), lack of infrastructure availability and a pervasive footprint. The concept of an *Application Overlay* network that virtualizes the underlay — essentially making it disappear from the application view –is the underpinning of Avesha's technology called Application Intent Mesh (AIM). We strongly believe that this idea while simple in concept is powerful and can remove the inertia of disaggregation. This can free up application developers to double down on building business logic and harnessing application details.

Kubernetes introduced the concept of separating the application logic from the deployment details using the concept of a "side car". With application disaggregation, this pattern is further extended over the wide area. Disaggregation may thus be also defined as the logical devolution of an application service into autonomous deployments like how a database is sharded or a data lake has multiple synchronous backups for performance or resilience. To enable these parallel operations, data (packets) must be "multi-casted" to all disaggregated units at the same time, hence an application overlay is in this case conceptually an application-level multicast over a multi-path overlay.

Application Slice

Avesha Application Intent Mesh (AIM) provides support for disaggregation through its patent-pending "Application Slice" abstraction for a virtual service mesh over the wide area for application deployments. SNAP's application slice abstraction creates an overlay topology that virtualizes the underlay capability via the slice API allowing application deployments to specify edgeification needs declaratively using application semantics.

Examples of application disaggregation include: (1) a data lake with application-level multicast for updating all data posts; (2) synchronous backups at multiple locations — replicas for performance and redundancy; (3) a gaming application that needs all players in a multi-player game to get a normalized service for achieving fairness; (4) a medical inferencing application that needs real-time inferencing to assist a doctor performing a medical procedure, where proximity of the inferencing GPU is key to achieving the desired performance goals; and (5) a multinational enterprise that needs to enforce its governance in order to follow regulations through influencing the data movement on the underlay.

Proximity, scale, & policy

Application developers want to follow the "network is invisible" model — just like what Kubernetes offers. They do not want to know about networking — they are only interested in specifying basic data movement requirements such as reachability, discovery, low-latency, GPU, etc. In addition, they want to establish an application contract at a service interchange or service-to-service communication. We extend the application developers language for business-network separation, i.e., the Application Intent, through a YAML interface that allows for complex application rules to be expressed in a human-readable form. This declarative language enables automation, removes fuzziness in interpretation of inter-service contracts and allows application developers to focus on business logic while offloading the plumbing to Avesha.

How does the "invisible network" abstraction manifest in an application slice? — especially, when there are distance issues and therefore topology issues. Distance must not be hidden because distance brings with it rules about access to data, encryption and transport. Also, distance means that we need to face latency — i.e., fairness and SLA (for

acceptable performance). A distributed footprint also brings the challenge of disaggregated networks, each with its own characteristics. Simplification and offloading such challenges will accelerate application deployment velocity. Avesha helps achieve that goal for the enterprise.

Avesha Application Intent Mesh (AIM) uses Reinforcement Learning (RL) to select paths that meet these application intents in a manner that scales. Several factors are considered including bandwidth, latency, costs, resource availability, governance rules, multipath routing, load times, among others. The initial concern is getting data to the edge (i.e., latency, network utilization (cost)); next, topology (movement — collection/distribution and governance); and ultimately, functions on the edge (inferencing GPUs, or VNFs).

Conclusion: In this note, we describe the vision of application disaggregation as a mechanism to enable edgeification — a growing trend that needs a good abstraction to enable faster adoption of the wide availability of edge computing to support a range of applications from gaming to medical. We believe that this vision of making the network disappear through and overlay is the right approach for widespread and rapid adoption of this exciting transformation.