White Paper

Communications Service Providers
5G Multi-Access Edge Computing

Wireline Access Evolution and 5G Fixed-Mobile Convergence

Authors

Padraig Connolly,
Intel Platform Application Engineer

Jasvinder Singh,
Intel Software Architect

Eoin Walsh,
Intel Platform Solution Architect

Andrew Duignan,
Intel Platform Solution Architect

Table of Contents

Executive Summary

With the advent of both fixed-mobile convergence (FMC) and Cloud-Native Network Functions (CNFs), Intel has worked hard on engraining both of these concepts as the core fundamentals of our wireline reference package. This paper puts forward a cloud-native architecture for the Access Gateway Function (AGF) that complies with FMC standards set out by the Broadband Forum (BBF) and 3rd Generation Partnership Project (3GPP) in SD-407[1] and Release 17.

This paper is separated into four main sections:

- **Introduction –** Here we introduce the concept of FMC, the architecture of the AGF, and how it can be built on a highly optimized software platform.
- **AGF Application Development –** In this part of the paper we give you a deep level view of the vAGF pipeline that was built using FD.io's Vector Packet Processing[2] (VPP) technology.
- **AGF Deployment Strategy –** This section shows the cloud-native side to the AGF deployment using technologies such as Kubernetes[3] (K8s) for orchestration, Single Root I/O Virtualization (SR-IOV) for high throughput I/O, and Device Config Function[4] (DCF)/Dynamic Device Personalization[5] (DDP) for accelerated packet distribution. We also mention the future technologies that Intel is working on with partner OEMs, OSVs, and ISVs to ensure that cloud-native networking achieves high performance and durability.
- **Experimental Results –** Finally to show the true performance and scaling (up and down) of the architecture put forward in this paper, we discuss the benchmarking results of the AGF running on 3rd Gen Intel® Xeon® Gold 6338N processors and Intel® Ethernet Network Adapter E810-2CQDA2 Network Interface Cards.

Introduction

A digital transformation is happening in the telecommunications industry. It follows the widespread realization that 5G is much more than just a technological evolution but an industrial shift in how services and content are consumed and delivered. As 5G begins its ramp from initial rollouts to the broad market, the traffic volumes, use cases, and network capacity come sharply into focus. Many operators have been preparing by improving and future-proofing their fiber infrastructures. This is acknowledgement that 5G is as much about the physical and transport delivery network as it is about the wireless spectrum.

This new network evolution phase aligns with the extensive work jointly proposed by the Broadband Forum and 3rd Generation Partnership Project (3GPP) in SD-407[1] to present networking to the user as a single pane of glass. This new approach is a transition point in the telecommunications industry, allowing consumers to seamlessly access services and content from both fixed and mobile networks. The standards partnership, commonly known as "Fixed-Mobile Convergence Alliance" (FMCA), looks at how this can be achieved and charts the technical requirements needed to make it a reality. The concept of interworking both mobile and fixed networks is not new. Solutions like Multi-Path TCP and Hybrid Access Gateways (TR-348)[2] have been around for years and have more recently been refined into Access Traffic Steering, Switching, and Splitting (ATSSS). This paper looks at the options proposed by the standards from a wireless point of view and goes deeper on the Access Gateway Function (AGF) and how it can be optimally architected, constructed, and deployed for the cloud paradigm. Further in this paper, we demonstrate that it is possible to achieve 632 Gbps of zero packet loss throughput on one host by using the principles and guidelines set forth by the standards alongside a state of the art VPP pipeline. Intel looks forward to working with its partners to bring high performance and cloud native FMC to the forefront of telco compute in 2022 and beyond.

Digitalization in the era of 5G

The previous solutions for combining mobile and wireless traffic always maintained two separate networks that "interworked." Broadband Forum's technical paper TR-470[7] presents a flattening of the network toward a common core and control plane. This is not just logically simpler but reduces the network operator's capital and operational expenditure through node, infrastructure, and management reductions as well streamlining their service delivery medium.

Figure 1 shows the varying convergence scenarios as defined in TR-470:

- Broadband Network Gateway (BNG) and Fixed Mobile Interworking Function (FMIF)
- Access Gateway Function (AGF) adaptive/integration direct mode
- AGF and User Plane Function (UPF) combined

Working alongside members of the Broadband Forum and industry partners, it was clear that the AGF was the first key step toward exploring FMC. Based on the maturity of the specifications and with a reference virtual broadband network gateway (vBNG) in hand, Intel added AGF functionality to the reference as defined in the standards and made it possible to benchmark a containerized AGF.

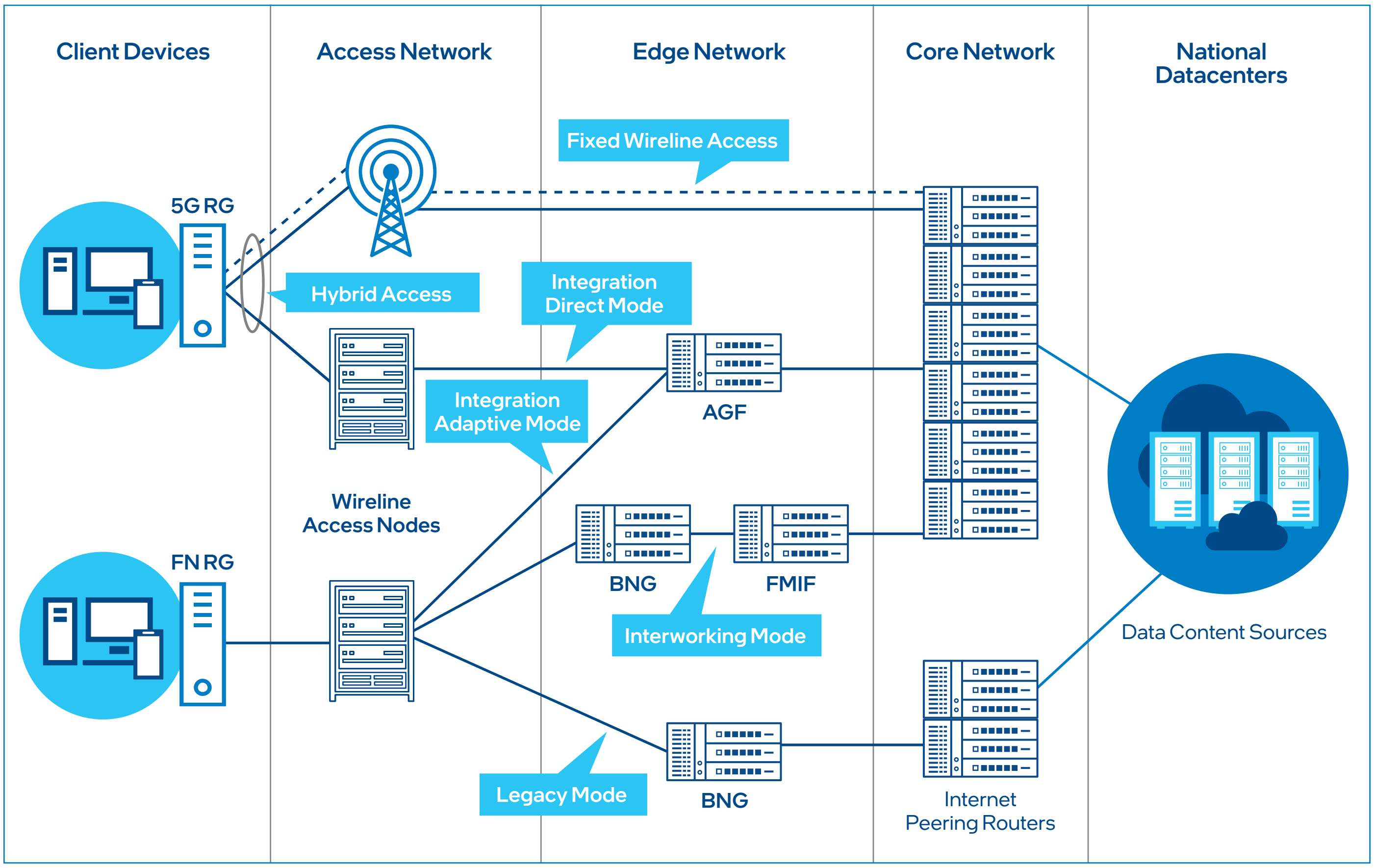


Figure 1. Varying convergence scenarios

AGF Architecture Approach

As network operators begin to evolve their infrastructures and software for 5G services, the architecture and design principles previously proposed for BNG workloads will remain valid for the AGF. This includes the CUPS (control and user plane separation) model, which decouples control and the user plane for flexible capacity expansion without restrictions from either plane. The AGF CUPS architecture includes a centralized AGF control plane (AGF-CP) running as a virtual network function interacting with multiple instances that may be more distributed than the AGF-CP to meet the bandwidth and latency requirements of customer services. In this model, there is a 1:N association between the AGF-CP and the AGF User Planes (AGF-UPs). The AGF-CP uses three interfaces to communicate with each AGF-UP, namely Control Packet Redirect interface (CPRi), State Control interface, and Management interface as specified by the Broadband Forum.[9]

There are two major advantages in deploying a standalone AGF. First, it allows an existing 5G network to connect seamlessly and offer broadband access services. Second, the AGF can sit in a remote edge location and offer breakout services to that location, which will be important for the provision of edge fibre accessed services.

The wireline AGF supports subscribers over fixed, fixed-wireless, or hybrid access, as shown in Figure 2. For subscribers connected through legacy Fixed Network Residential Gateway (FN-RG), which has no 5G capability and does not interact directly with the 5G core, the AGF-CP provides an N1/N2 signaling interface, and each AGF-UP provides an N3 interface toward the 5G network. In this case, the AGF terminates the legacy wireline access interface on one of its sides and implements the 5G core interfaces N1/N2/N3 on its other side.

In addition to supporting the FN-RG, the AGF enables the 5G core to serve residential gateways that are modified to support 5G protocols and procedures (5G-RG). The 5G-RG will have either a wireless interface (like a fixed wireless access deployment), the wireline access interface, or both to ensure higher throughput and reliability. Regardless of the interface in use, 5G-RG supports the 3GPP N1 reference point, which is used to pass NAS (Non-Access Stratum) signaling between it and the 5G core.

AGF Development

AGF Control Plane

The AGF-CP is a central part of the cloud-based AGF solution. For fixed-access subscribers, the AGF-CP contains several control function modules, including one for completing authentication, authorization, and accounting (AAA) management functions. It also contains functions for requesting IP address assignment from 5G core and subscriber management (registration and deregistration). Some of these modules are responsible for the connection with outside subsystems, such as 5G core and other management systems. All the AGF-CP functions are implemented as virtual network functions and are hosted on a centralized Intel® Xeon® platform.

The reference implementation doesn't consider the case where control packets are intercepted at the steering function and steer towards the AGF-CP. Therefore, control plane packets are handled in-band with data plane packets at the server machine running AGF-UP instances. As shown in Figure 3, the Intel® Ethernet Controller E810[10] filters the control plane traffic by subscriber (de-) registration, session setup/termination, etc., and forwards it to separate virtual functions and queues, thus freeing the AGF-UP instances from this task.

For recognizing control plane packets, the Dynamic Device Personalization[5] (DDP) profile is applied on the network interface controller (NIC), and switching rules are programmed on the NIC filters. The AGF-CP and AGF-UP use the Control Packet Redirect Interface (CPRI) to exchange control packets, such as VXLAN, etc., over the tunnel. In addition to control plane traffic, the AGF-CP uses Packet Forwarding Control Protocol (PFCP) over the State Control Interface (SCI) to install forwarding rules and related states for the subscriber traffic on the AG-UP instances. In addition to CPRI and SCI interfaces, Management Interface (MI) enables the control plane to push configuration and retrieve operational state and status to and from the AGF UPs.

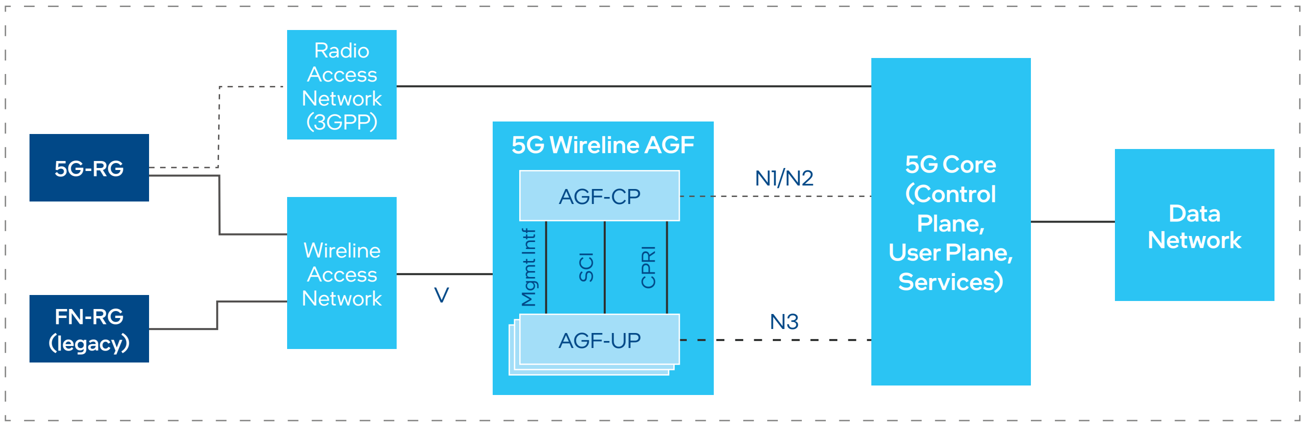


Figure 2. Wireline Converged Access onto 5G core via AGF

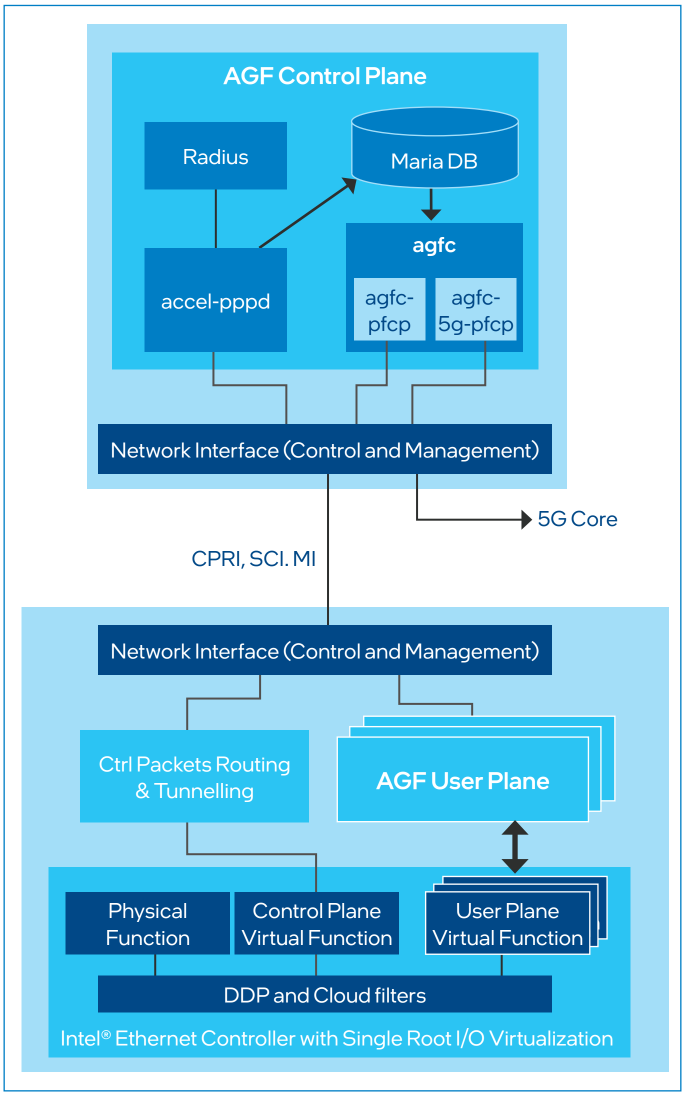

AGF Control Plane
Radius
Maria DB
agfc
accel-pppd
agfc-pfcp
agfc-5g-pfcp
Network Interface (Control and Management)
5G Core
CPRI, SCI. MI
Network Interface (Control and Management)
Ctrl Packets Routing & Tunnelling
AGF User Plane
Physical Function
Control Plane Virtual Function
User Plane Virtual Function
DDP and Cloud filters
Intel® Ethernet Controller with Single Root I/O Virtualization

Figure 3. AGF Control Plane Integration with User Plane

AGF User Plane

The AGF-UP routes the traffic between the subscribers connected to the wireline access network and the 5G core network of communication service providers. It then implements functions such as policy rule enforcement as per service level agreements (SLAs), packet classification, header processing, transport encapsulation/decapsulation, hierarchical quality of service traffic management, and more. Its functional complexity depends upon the service provider requirements.

AGF-UP functionality in the reference implementation uses separate uplink and downlink packet processing pipelines. Each pipeline applies a set of functions to each packet that enters the pipeline. As shown in Figure 4, the uplink packet processing pipeline handles the packets flowing from the subscriber access network to the ISP 5G core network, while the downlink pipeline deals with the packets running from the core network to the access network. The average packet size of upstream traffic is typically smaller than for downstream, and the amount of upstream traffic is normally five to eight times less than downstream traffic. In recent years, the traffic gap between upstream and downstream has reduced significantly due to increased use of applications like Instagram, Snapchat, TikTok, etc.

The reference AGF-UP is developed using the performance-optimized Vector Packet Processing (VPP) framework,[2] which leverages Data Plane Development Kit (DPDK) drivers[11] to take advantage of high speed I/O. The VPP is designed around an extensible and modular packet-processing graph architecture in which each independent graph node does limited packet processing on a vector of packets. The VPP framework allows application developers to plug in new graph nodes without changing core or kernel code to build customized packet processing solutions. The AGF uplink and downlink pipelines are implemented as separate packet processing graphs around a number of nodes where each graph node implements a specific packet processing function.

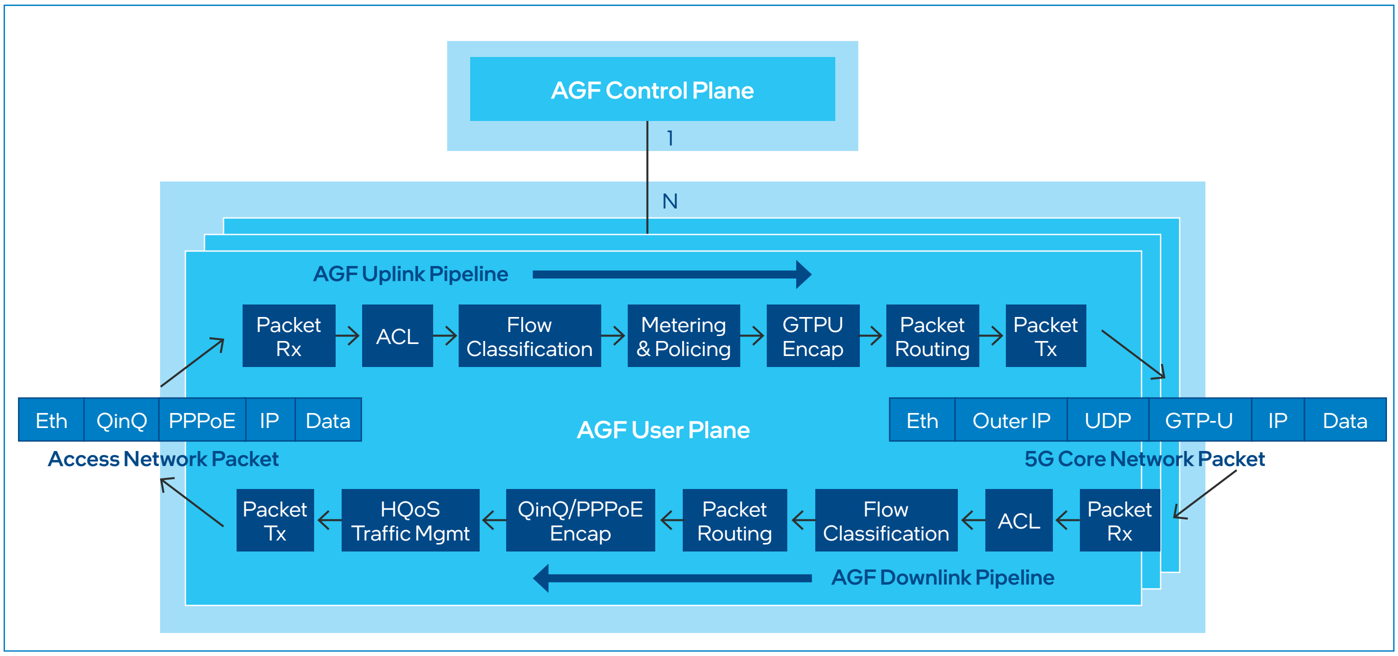


Figure 4. AGF User Plane Packet Processing Pipeline

Uplink Packet Processing Pipeline

The reference implementation of the AGF uplink packet processing pipeline consists of the following functions constructed as a directed graph of VPP nodes, as shown in Figure 4.

Packet Rx (Receive): Packets from the access network are received from the NIC ports using DPDK PMD drivers and sent to the next stage to begin packet processing.

Access Control List (ACL): This stage employs an access control list (ACL) table to implement firewall policies, such as block rules, on the incoming traffic. A table lookup operation is performed on each received packet, and in case of rule match, the packet is dropped.

Flow Classification: Exact-match classification is performed on the 5-tuple header fields of the input packets (source and destination IP addresses, source and destination UDP/TCP ports, and transport layer protocol ID) to identify the session. The session information is stored as packet metadata to be used later in the pipeline, and access network encapsulations (QinQ + PPPoE header) are stripped off the packets.

Metering and Policing: This function meters the subscriber traffic flows to determine compliance with a service contract and applies traffic policing to enforce the contract. As a result, packets that conform to a specified rate are sent to the next stage of the pipeline while packets that violate the rate are dropped.

GTP-U Encapsulation and Routing: Packets are encapsulated with a GPRS Tunnelling Protocol User Plane (GTP-U) header at this stage and routed to the 5G core network through the correct network interface port.[12]

Packet Tx (Transmit): With the help of DPDK poll mode drivers, packets are transmitted out of the system through the NIC ports connected to the 5G core network.

Downlink Packet Processing Pipeline

The reference implementation of the AGF downlink packet processing pipeline consists of the following functions constructed as a directed graph of VPP nodes, as shown in Figure 4.

Packet Rx (Receive): Packets from the 5G core network are received from the NIC ports using DPDK PMD drivers and sent to next stage to begin packet processing.

Access Control List (ACL): This stage employs an ACL table to implement firewall policies, such as allow rules, on the incoming traffic. A table lookup operation is performed on each received packet, and in case of rule match, the packet proceeds to the next stage.

Flow Classification: The 5G core network encapsulations (IP + GTPU Header) are stripped off the packets and exact-match classification is performed on the 5-tuple header fields (source and destination IP addresses, source and destination UDP/TCP ports, and transport layer protocol ID) to identify the session. The session information is stored as packet metadata to be used later in the pipeline.

QinQ Encapsulation and Routing: Packets are encapsulated with a QinQ (IEEE 802.1ad) header and routed to the access network through the correct network interface port.

Hierarchical QoS Traffic Management: Each packet runs through a hierarchical QoS (HQoS) scheduler to ensure that thousands of subscribers can get the desired broadband capacity as per the service contract. It supports scalable five-level hierarchical construction of traffic shapers and schedulers to allow fine-grain traffic control compared to a traditional single-level QoS scheduler.

Packet Tx (Transmit): With the help of DPDK poll mode drivers, packets are transmitted out of the system through the NIC ports connected to the access network.

AGF User Plane Management

AGF User Plane management refers to the functions and processes used for programming forwarding rules on an AGF-UP by interfacing with the AGF-CP. As shown in Figure 5, the management plane functions in the AGF reference implementation provide a standardized northbound interface (PFCP interface, etc.) to the AGF-CP, and southbound interfaces (VPP binary API interface) to the AGF-UP.

The AGF PFCP agent translates the messages from the AGF-CP into AGF-UP specific command format (VPP in this case), and subsequently stores them in etcd (a key-value database). The etcd data store is chosen to provide a reliable way of storing information that needs to be accessed by a distributed system.

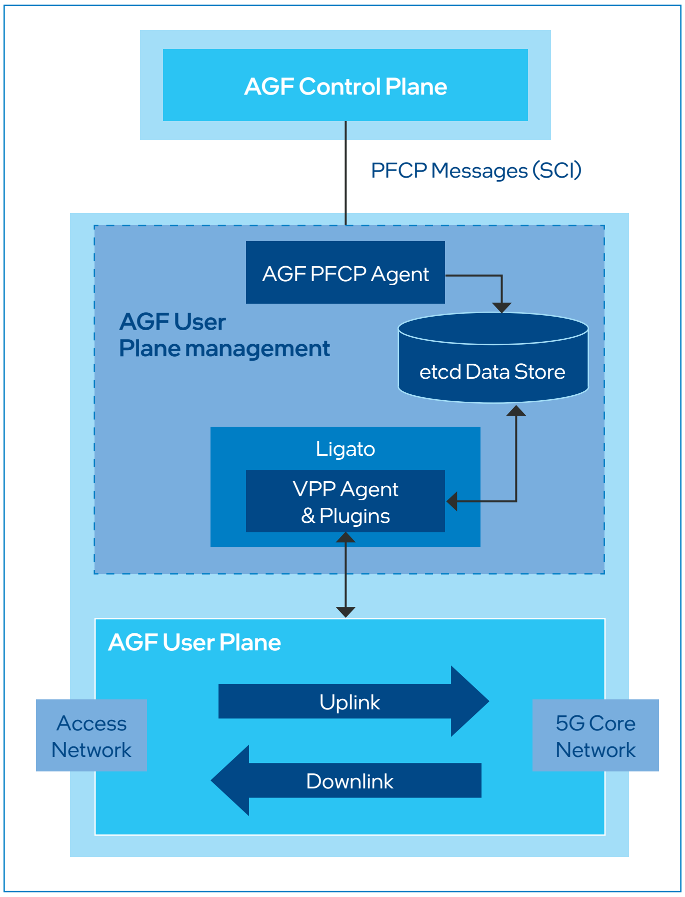


Figure 5. AGF User Plane Management

Ligato offers infrastructure and agents to control and manage cloud-native network functions and provides a VPP agent for programming a VPP-based AGF-UP. The VPP agent enables configuration and monitoring services for the VPP data plane, as well as several plugins to control and manage the VPP data plane nodes constituting the AGF uplink and downlink pipelines. The VPP agent synchronizes with the etcd data store for receiving traffic-forwarding rules and passes them to management plugins for configuring the rules on VPP data nodes of the AGF-UP uplink and downlink pipelines via the VPP binary interface. All the telemetry messages reported by the user plane nodes are exposed through Ligato to the infrastructure layer.

AGF Deployment Strategy

The AGF deployment strategy fully embraces the architectural and operational principles of cloud-native networking where a singular data plane networking entity in the stack is called a Cloud-Native Network Function (CNF). A deployed CNF generally should follow these conventions:

- **Highly Performant:** The CNF must take advantage of Enhanced Platform Awareness (EPA) features to ensure low latency and high throughput.
- **Agile Placement:** The CNF must allow for flexible placement to deploy on any EPA feature-ready platform and must be generic to the underlying EPA infrastructure.
- **In Service Lifecycle Management:** Using automatic telemetry-aware controllers, the CNF must ensure that it can scale resources under increasing workloads and retract resources under decreasing workloads.
- **Highly Available:** The CNF must be highly available and fault-tolerant to meet the service level agreement of near-zero downtime. The CNF must also utilize the HA schema to maintain interfaces for quick and simple service upgrades without affecting the service it provides.
- **Observability:** The CNF must ensure that all network and performance metrics of workloads are exposed through an easy-to-consume platform, allowing for rapid network debugging and modification.

As is common with many cloud deployments, the AGF uses the Kubernetes container-orchestration engine to ensure all the CNF conventions above are met.

To ensure the methodology of the CUPS architecture and simplify the management of each infrastructure-agnostic microservice, the AGF stack makes a macro level segmentation between the CNF infrastructure and the AGF application. Within the AGF application segment, the deployment is again segregated into functional sections based on their role in the CUPS architecture: the control plane, the user plane management, and the user plane. All these segments including the infrastructure will be discussed below with a focus on how they are deployed.

From the perspective of a network operations engineer, the AGF cluster is deployed and maintained from a secure admin portal. This remote configuration is enabled through the use of Helm Charts.[13] For Day 0 app deployment, Helm Charts are rolled out on top of the requested edge cloud infrastructure. The network operations engineer is also able to monitor and change lifecycle elements of the AGF cluster using Helm Charts and through user-friendly interfaces.

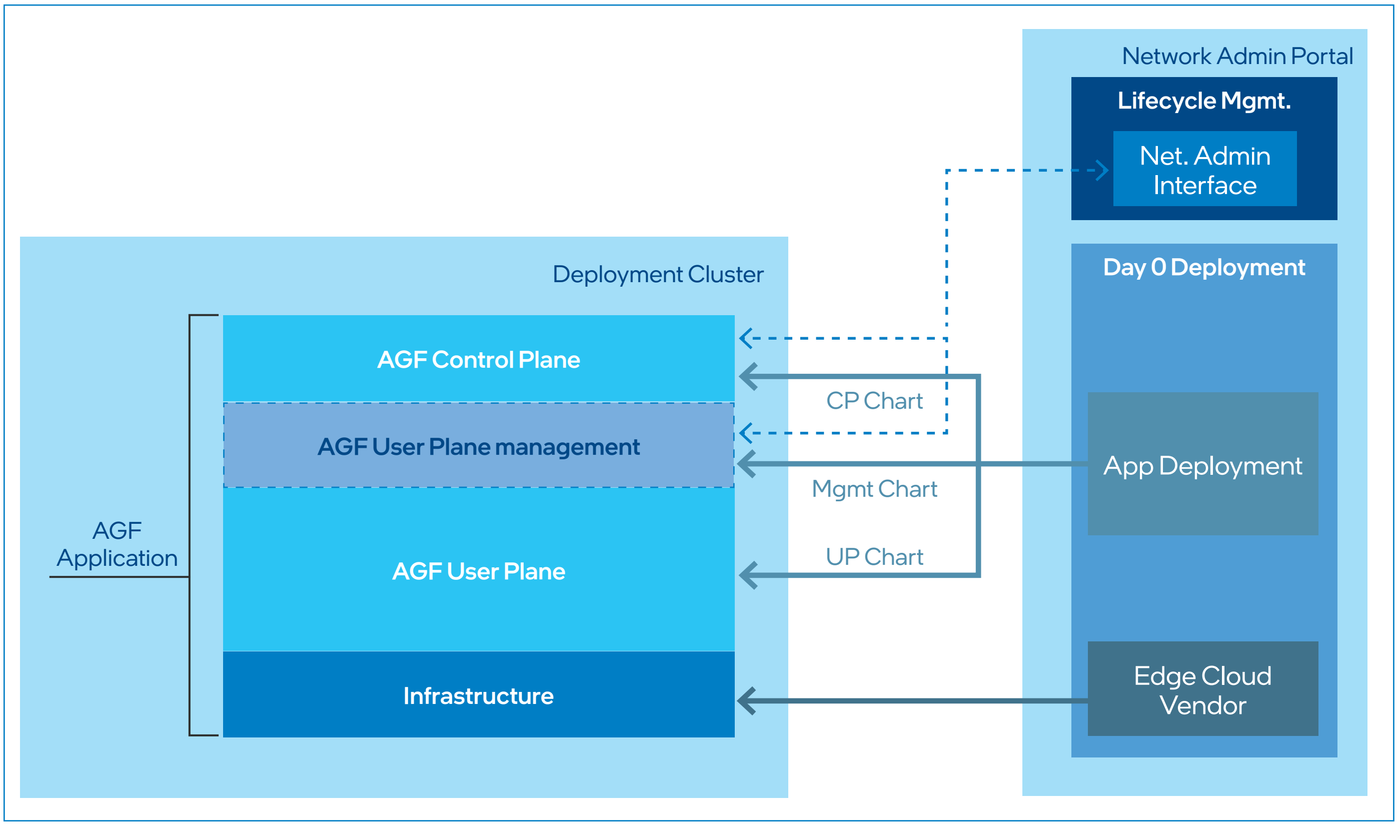


Figure 6. High Level View of AGF Deployment

The steps of deployment for the AGF cluster are as follows:

1. The network operations engineer requests the cloud platform infrastructure that will host the AGF cluster using an edge cloud of their choice. In the AGF reference application, the default infrastructure deployment is handled by the Container Bare Metal Reference Architecture (BMRA)[14]. Note that by design, the AGF application is agnostic to the underlying infrastructure and thus does not require a specific infrastructure deployment platform such as the BMRA.
2. Once the infrastructure is in a ready state, the network operations engineer will deploy each of the application plane charts.
3. Once all the application elements are up and running, the network operations engineer will proceed to configure the transport network on the 5G core and access side to route subscriber traffic to their assigned AGF instance while monitoring application and infrastructure metrics to ensure that peak performance is in line with the SLA.

The end state of this deployment can be seen below in Figure 7.

Infrastructure

When migrating from a Virtual Network Function (VNF) deployment to a CNF deployment, the infrastructure is generally a key differing component, and this is no different for the AGF in this scenario. The infrastructure encapsulates three components of the AGF deployment: orchestration, platform resource allocation and management, and cluster networking. All of these will be discussed in detail below. The BMRA is the default infrastructure component for the AGF as it contains an Intel architecture-optimized reference profile dedicated to the AGF/BNG deployment called "remote_fp." The BMRA represents a baseline configuration of components that are optimized to achieve maximum system performance mainly for CNF use cases when running on Intel® Xeon® Scalable processors.

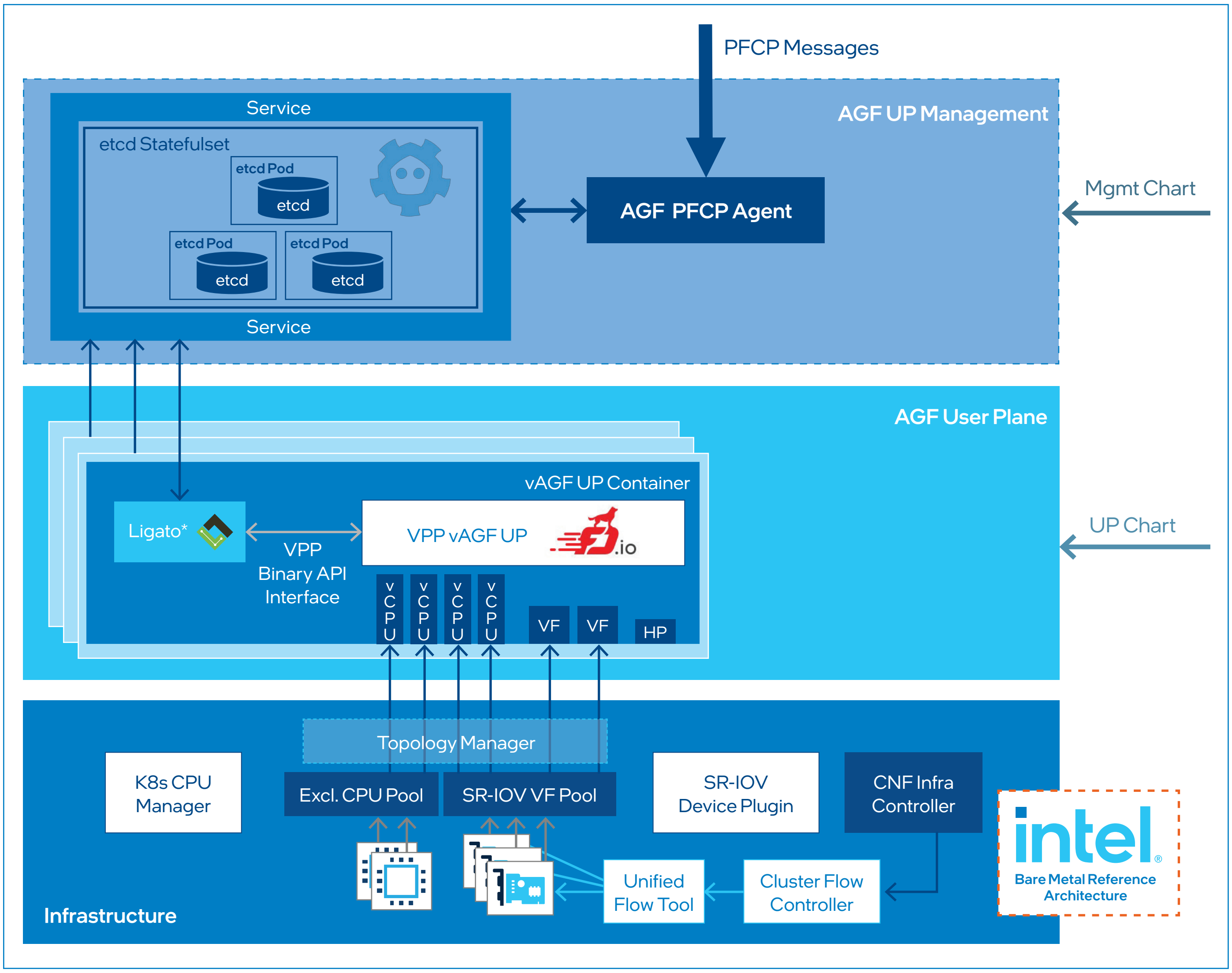


Figure 7. Detailed View of AGF User Plane Deployment

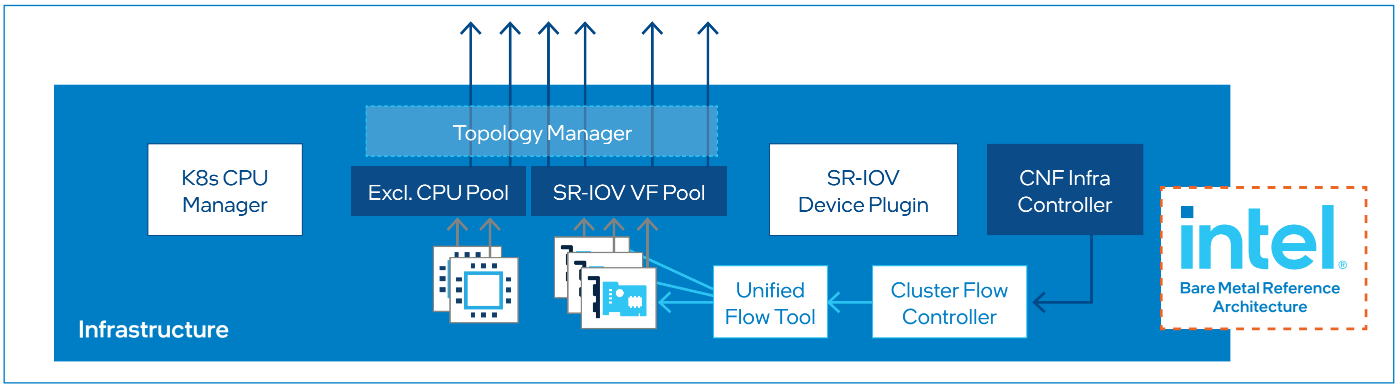


Figure 8. User Plane Infrastructure

Orchestration

Ultimately all the responsibility for running, maintaining, and updating each microservice in the AGF deployment lies with Kubernetes (K8s). When the pod goes down, it is expected that K8s will bring the pod or service back up again. Also, if a pod container image is updated through the Helm upgrade procedure, K8s is expected to perform a clean version migration.

Hardware Management

In general, hardware resources in non-telco cloud deployments are not a major concern for application vendors, but when deploying a CNF such as the AGF-UP, it is vital that it gets the required Enhanced Platform Awareness (EPA) resources to ensure high performance and low latency data plane processing. This is necessary to ensure that subscriber SLAs are met by the user plane vendor. EPA represents a methodology for targeting intelligent platform capability, configuration, and capacity consumption. EPA orchestration systems have the ultimate goal of delivering improved and deterministic application performance through a cloud-friendly manner.

A single instance of the AGF-UP requires three hardware resources from the infrastructure:

- 4 vCPUs from the exclusive K8s CPU Manager pool
    - 1 vCPU is used for uplink data plane processing
    - 2 vCPUs (sibling hyperthreads) are used for downlink data plane processing
    - 1 vCPU is used for command line interface (CLI) and user plane management
- 2 SR-IOV virtual functions (uplink and downlink)
- 512 2-MB Hugepages

The above resources are allocated by K8s in a cloud-friendly, portable manner. The following sections give a detailed overview of the K8s mechanisms used by the AGF to achieve an optimum configuration.

K8s CPU Manager

The K8s native CPU Manager is used to allocate exclusive paired vCPUs (sibling hyperthreads) to the AGF-UP. The CPU Manager uses a combination of isolcpus and the Kubelet “reserved-cpus” flag to safely allocate exclusive isolated vCPUs to the vAGF application. Figure 9 below shows how this is achieved for the AGF deployment for a single node configuration.

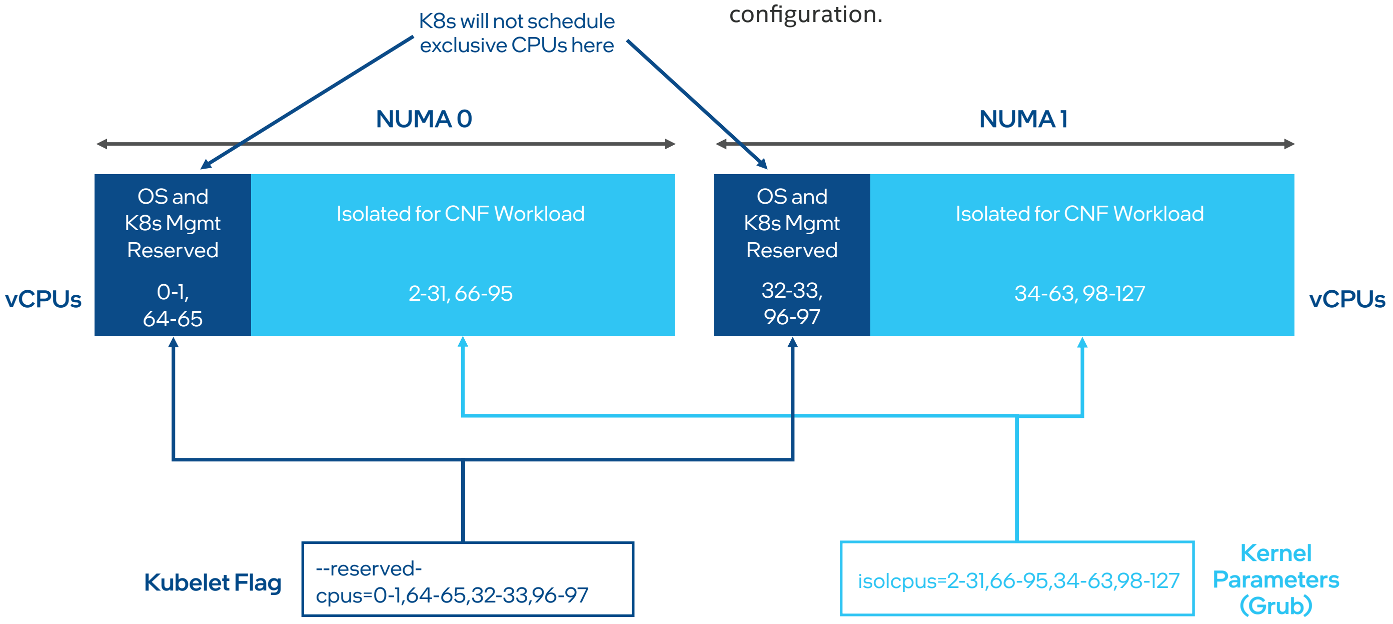

K8s will not schedule exclusive CPUs here
NUMA 0
NUMA 1
vCPUs
OS and K8s Mgmt Reserved
0-1, 64-65
Isolated for CNF Workload
2-31, 66-95
OS and K8s Mgmt Reserved
32-33, 96-97
Isolated for CNF Workload
34-63, 98-127
vCPUs
Kubelet Flag
--reserved-cpus=0-1,64-65,32-33,96-97
isolcpus=2-31,66-95,34-63,98-127
Kernel Parameters (Grub)

Figure 9. AGF-UP Single Node CPU Allocation

Sibling vCPUs are required in the vAGF application because the downlink pipeline needs two vCPUs to meet the expected subscriber performance SLA. These two vCPUs must reside on the same physical core to ensure rapid memory exchange (two vCPUs on the same physical core share Last Level Cache (LLC)). The first sibling vCPU allocated to downlink for packet processing runs the DPDK HQoS scheduling node. Networking blocks (RX, ACL, Flow Classification, QinQ Encapsulation and Routing) are run on the second sibling vCPU. Note that HQoS is the last node ran on the packet before TX; the vCPU it is placed on does not have any effect on where it runs in the downlink pipeline.

K8s SR-IOV Device Plugin

To dynamically allocate SR-IOV Virtual Functions to each vAGF pod and other AGF components, the K8s SR-IOV Device Plugin[16] (SR-IOV DP) is used. This works by building pools of SR-IOV devices that can then be fed through the K8s Device Manager as a requestable resource in the vAGF PodSpec. For the AGF deployment, four SR-IOV Virtual Function (VF) pools are created, one for Device Config Function[4] (DCF), one for Control Packet Redirect interface, one for vAGF-UP uplink interfaces and one for vAGF-UP downlink instances.

K8s Topology Manager

One important K8s EPA feature that the vAGF deployment requires is the K8s Topology Manager. This component ensures that the vCPUs and the SR-IOV VF devices assigned to a pod are NUMA-aligned, ensuring low latency and maximum hardware performance for the CNF.

CNF Infrastructure Controller

The CNF Infrastructure Controller is the main interface of the AGF application into the infrastructure stack. Using runtime K8s custom resources for configuration, the controller talks to K8s sub-operators that are responsible for their own assigned hardware. In the current AGF deployment setup, the infrastructure controller simply writes rules to the Cluster Flow Controller (explained below). However, the scope of this controller will be expanded in the future to configure more platform resources for the AGF application and integrate further into the K8s ecosystem.

Device Config Function

The AGF uses the SR-IOV eSwitch and DDP functionality on the Intel Ethernet Controller E810 to complement the CUPS architecture. The AGF uses traffic steering on the NIC to perform two macro level functions:

- Route control plane discovery and session packets directly to the control plane (VF1) without the cost of going through the data plane (CPRi).
- Route user plane packets to the correct vAGF instance running on an SR-IOV VF on that Physical Function (PF) (physical port).

To program these rules, the AGF uses the Device Config Function (DCF), available on the Intel Ethernet Controller E810.[10] With DCF, DPDK RTE_FLOW DDP rules can be sent through a trusted VF that uses the SR-IOV mailbox to communicate with a PF bound to a kernel device driver, which ultimately programs different hardware offload elements on the Intel Ethernet Controller E810.

Two infrastructure tools are used to expose DCF in a cloud-friendly manner:

- **Unified Flow Tool –** In the AGF, there is a Unified Flow Tool (UFT) instance deployed per worker node. The UFT has a northbound gRPC API interface for programming DCF rules through a respective trusted VF for a desired PF.
- **Cluster Flow Controller –** This tool offers the ability to program the equivalent UFT instance on its respective host. The Cluster Flow Controller uses a K8s custom resource definition (CRD) for its northbound programming interface.

In the future, both of these tools will be incorporated into the Ethernet Operator which will be discussed further in future papers.

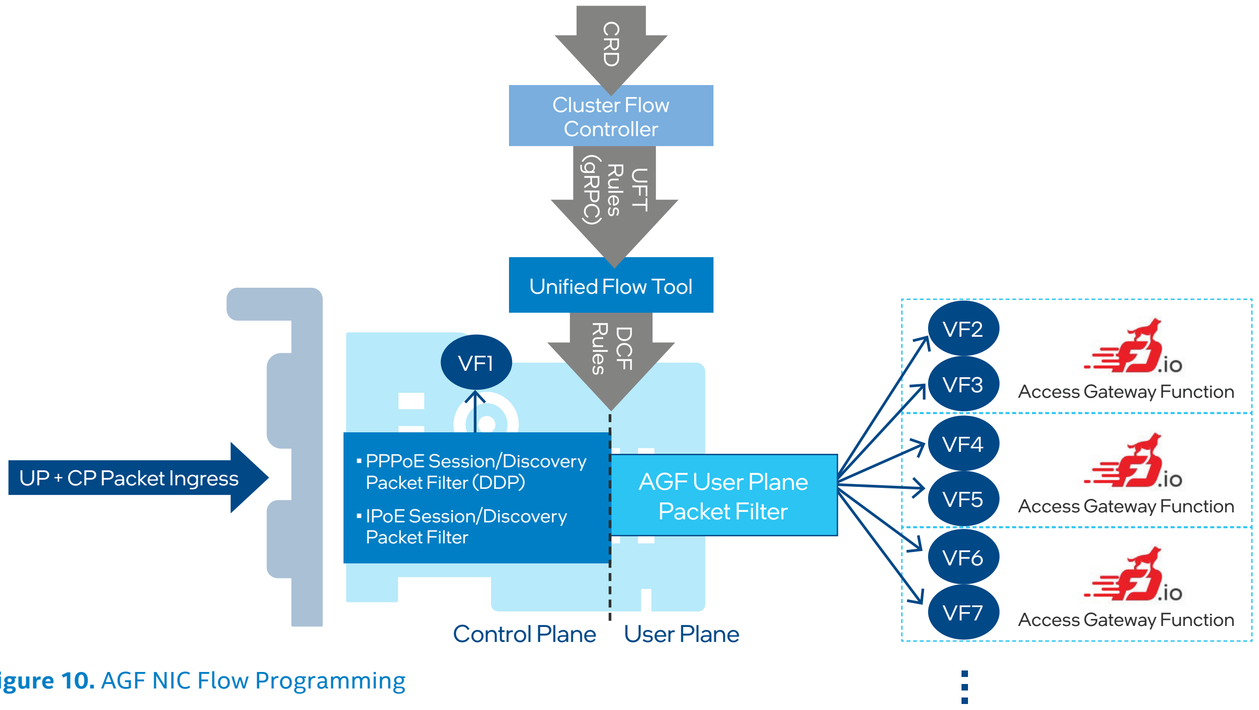


Figure 10. AGF NIC Flow Programming

Dynamic Device Personalization

Under the default configuration (default DDP package), the Intel Ethernet Controller E810[10] can support traffic steering on packet headers that are common to most networking domains. To support custom packet types, a Dynamic Device Personalization (DDP)[5] package can be loaded for immediate use without reloading the Ethernet controller NVM image.

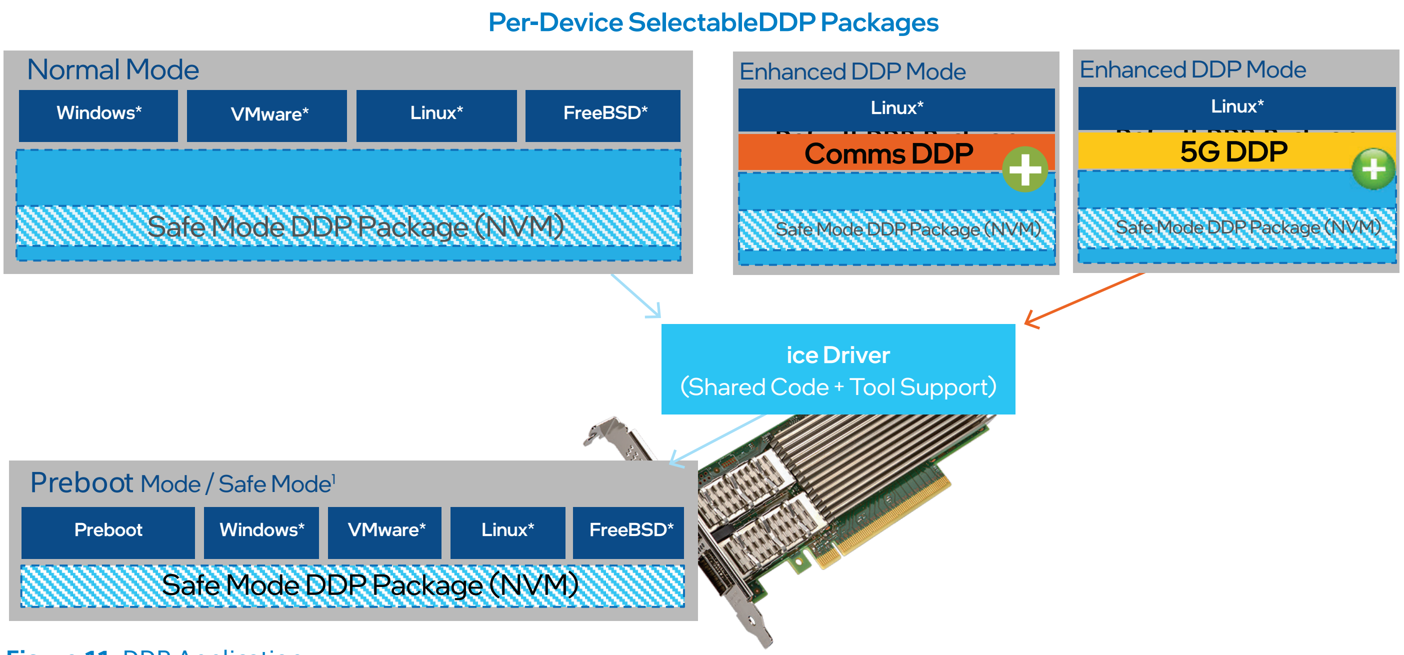

Per-Device SelectableDDP Packages
Normal Mode
Windows*
VMware*
Linux*
FreeBSD*
Safe Mode DDP Package (NVM)
Enhanced DDP Mode
Linux*
Comms DDP
Safe Mode DDP Package (NVM)
Enhanced DDP Mode
Linux*
5G DDP
Safe Mode DDP Package (NVM)
ice Driver
(Shared Code + Tool Support)
Preboot Mode / Safe Mode1
Preboot
Windows*
VMware*
Linux*
FreeBSD*
Safe Mode DDP Package (NVM)

Figure 11. DDP Application

In the context of the AGF deployment, the Telecommunication (Comms) Dynamic Device Personalization Package is used. Once added, this package allows the Ethernet controller to steer traffic based on PPPoE header fields, thus supporting control plane offloading described in the Device Config Function. Figure 12 provides an example of what the AGF can achieve with NIC traffic steering and DDP:

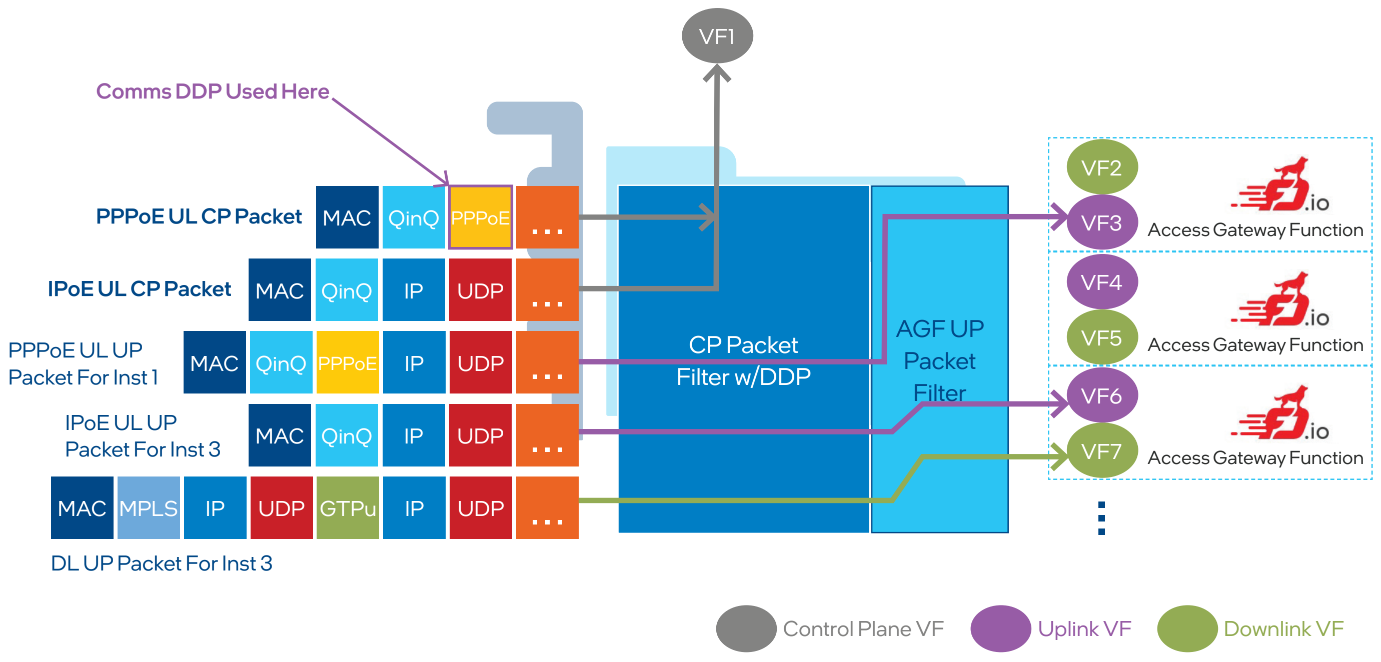


Figure 12. AGF Traffic Steering with DDP

Cluster Network

The AGF deployment uses Flannel for its overlay network infrastructure. The overlay network is the component that provides network plumbing between all elements in a K8s cluster. Using a mixture of Linux network bridges and VXLANs, Flannel provides an inter-service networking component for bare metal deployments that is easy to install and maintain. Along with Flannel, CoreDNS is used to assign all service interfaces with an addressable hostname. This greatly simplifies deployments as each service in the AGF cluster can automatically initialize without the need to manually enter service IP addresses for connection initialization. The network mesh is used in the AGF to connect the following components together:

- vAGF-UP Pod ↔ etcd
- etcd↔PFCP Agent
- PFCP Agent ↔ Control Plane (Depending on location of CP relative to UP cluster)
- Network Controller ↔ CNF Infrastructure Controller
- CNF Infrastructure Controller ↔ Cluster Flow Controller
- Cluster Flow Controller ↔ Unified Flow Tool

Figure 13 below gives an overview of how Flannel is used to connect each component in the AGF deployment. In the future, it is planned for all Control Plane and Infrastructure traffic to run over a Service Mesh to provide more security and observability of management level traffic.

Application

From a telco perspective, network applications can be implemented differently for different requirements. However, the general principles that apply to a typical CNF deployment apply to the AGF application deployment as well. These principles generally encapsulate how the CNF interacts with its environment, for example, how EPA features are allocated and how the northbound user plane management interfaces are used for programming user plane rules.

There are three functional components to the AGF application deployment: AGF Control Plane, AGF User Plane, and AGF Management. The deployment of each will be described below.

AGF Control Plane

The control plane used in the AGF deployment is built by Berlin Institute of Software Defined Networks (BISDN). This follows the same micro service architecture as the other components in the AGF deployment in which each element is deployed as a container entity. For the AGF deployment, the AGF-CP may be deployed on the same cluster as the AGF User Plane management and AGF-UP or in a separate remote cluster for commanding multiple AGF-UP clusters. If deployed in the same cluster, the control plane uses the same container network interfaces as the other AGF components. If the network operations engineer requires the AGF-CP to be placed in a remote cluster, a K8s ingress controller would be set up on the user plane cluster to ensure external AGF -CP access is regulated and load-balanced for the UP PFCP agent to receive and parse messages.

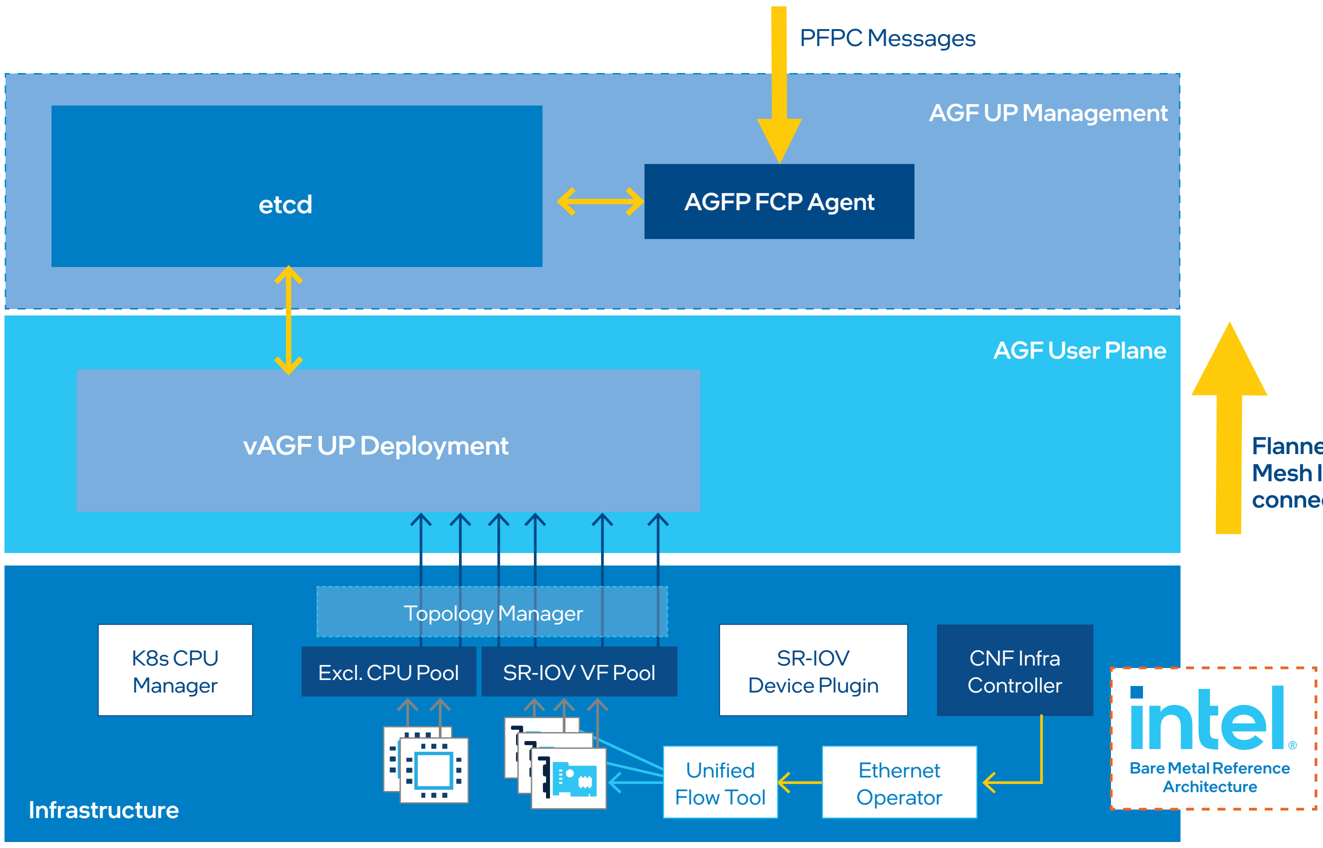


Figure 13. Cluster Network Overlay Mesh

AGF User Plane Management

From a deployment perspective, the AGF-UP management section provides three functions:

1. Store the current and recent states of the AGF-UP instances in a highly available etcd database.
2. Northbound control plane PFCP comms interfacing.
3. Southbound interface for programming of forwarding rules to the AGF-UP through the Ligato VPP agent.

As can be seen in Figure 14, the central part of the AGF-UP management is the highly available etcd database. The etcd data store offers an extremely reliable configuration state database allowing for easy application component recovery.

The AGF PFCP agent is deployed in its own pod. Its main function is to convert the PFCP messages from the AGF-CP in a form compatible with the AGF-UP and enter this data into the etcd database cluster.

At initialization, each Ligato instance is provided with the etcd service DNS name for immediate static etcd discovery. Once the connection is established, the Ligato instance will submit its unique instance settings and entries through its northbound interface to the etcd database. The Ligato instance will also watch for any CP rule setting changes inserted by the PFCP agent and will proceed to insert the respective rules into the vAGF-UP VPP application pipeline.

AGF User Plane

The vAGF-UP pod is the simplest component of the AGF User Plane deployment, containing a single entity called the "vAGF-UP Container." Within this container is the VPP vAGF uplink/downlink packet processing pipelines and its respective VPP agent for programming control plane rules from its northbound interface (Figure 15).

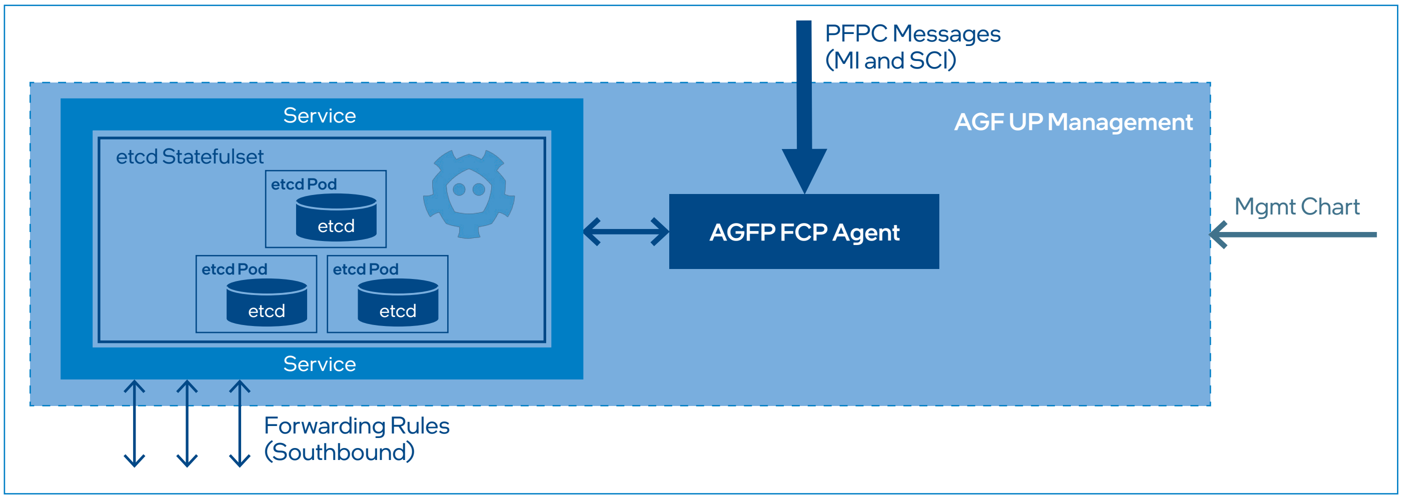


Figure 14. Management Plane Deployment

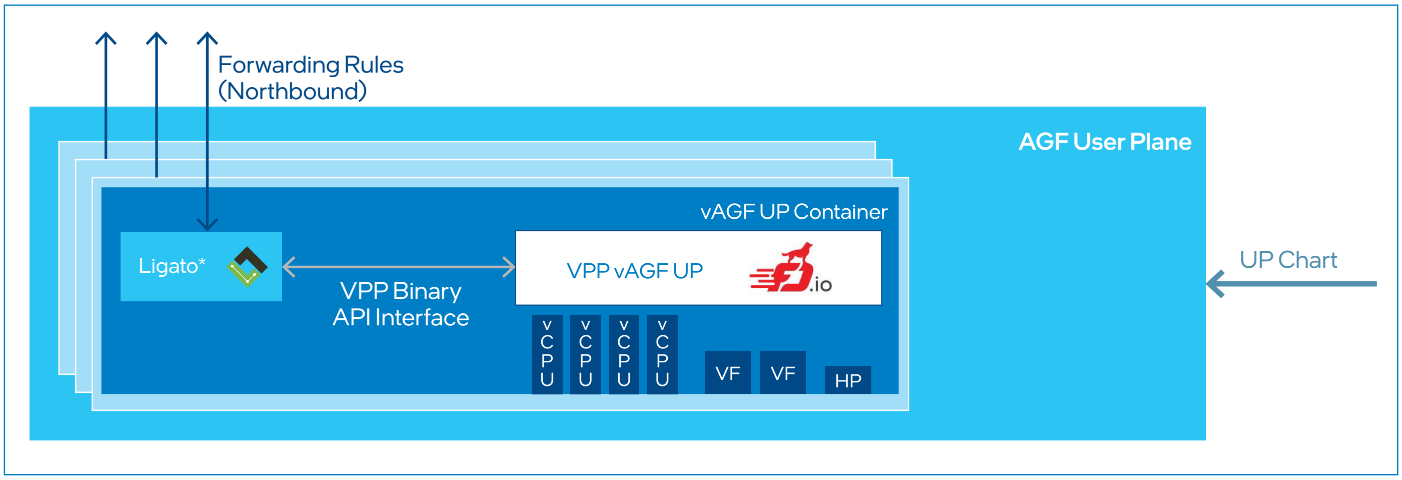


Figure 15. User Plane Deployment

** Ligato is seen logically as an AGF-UP management object. From a deployment perspective, it resides in the AGF User Plane.*

The vAGF-UP container image is packaged using a set Dockerfile from the Docker container toolset. As part of the Docker build process, all the VPP and Ligato libraries are built and pre-configured for initialization when the pod is brought up. At initialization, the Ligato VPP agent connects to the VPP vAGF-UP CLI using a socket interface through the Ligato GoVPP core. Multiple Ligato plugins can connect to the VPP instance through their own channel using the GoVPP Mux plugin.

Finally, the vAGF-UP pod is deployed and scaled out using the AGF-UP Helm Charts.[13] This allows the network operations engineer to configure aspects of the AGF-UP deployment including the number of instances, current vAGF container image, etc.

Application Overview

The final high level overview of the deployment model includes a single AGF control plane instance commanding multiple child user plane instances all programmed through the Ligato interface described above. Figure 16 gives a view of what this will look like.

Experimental Results

The Intel reference AGF application is benchmarked to understand how the architecture put forward in this paper works in a real context. To get a more comprehensive view of Intel compute, two tests are run, a symmetric traffic workload and an asymmetric traffic workload. The differences between these two tests are described further on in this section. The traffic benchmarking methodology used is RFC2544[17] throughput testing.

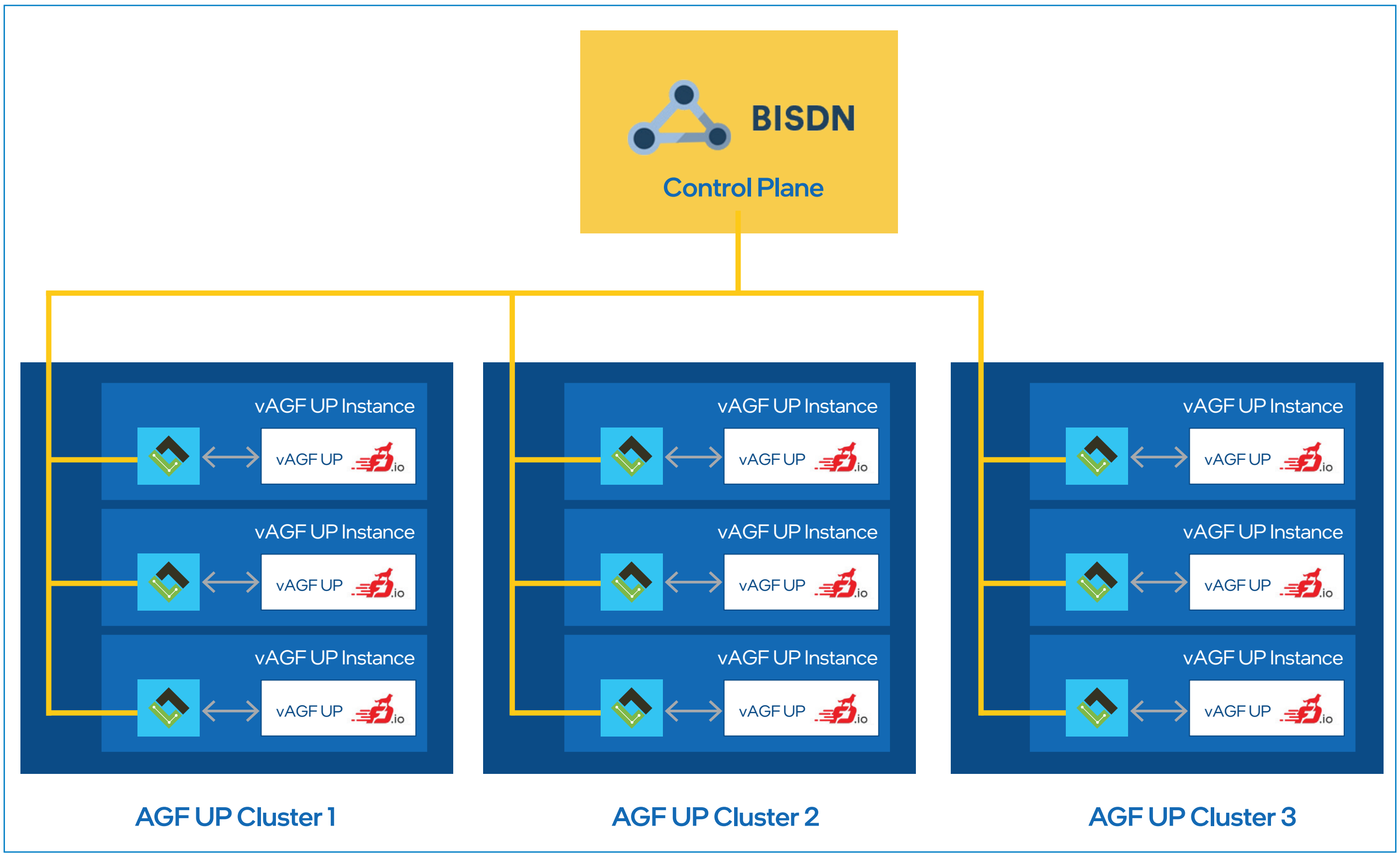


Figure 16. High Level AGF Deployment Overview

Test Details

The following table provides more key information points about the benchmarking setup:

Table 1: Benchmarking Configuration Details

Configuration Name			Configuration Setting
Hardware	Server	Name	M50CYP2SB2U (K88091-301)
		BIOS	20.P30 (uCode: 0xD0002C10)
	CPU	Name	3rd Gen Intel® Xeon® Gold 6338N processors (x2) 2.2 GHz, 32 cores
		OS	Ubuntu 20.04.3 LTS (5.4.0-81-generic)
		# Isolated CPUS	30
		Hyperthreading	On
		Turbo-boost	Off
	NIC	Name	Intel® Ethernet Network Adapter E810-2CQDA2 (x4)
		FW	3.00
		Driver	1.6.7
	Memory	Name	9965600-023.A00G Kingston 16 GB (x16) (2133 MHz)
		Hugepages	104 GB of 2M Hugepages
	Tester		Keysight Novus QSFP28 100GE
Application	Name		virtual Access Gateway Function
	Version		21.03
	VPP Version		21.01
	DPDK Version		20.11

Core Layout

When all 30 instances of the AGF were deployed on the device under test, the following is the core and I/O layout:

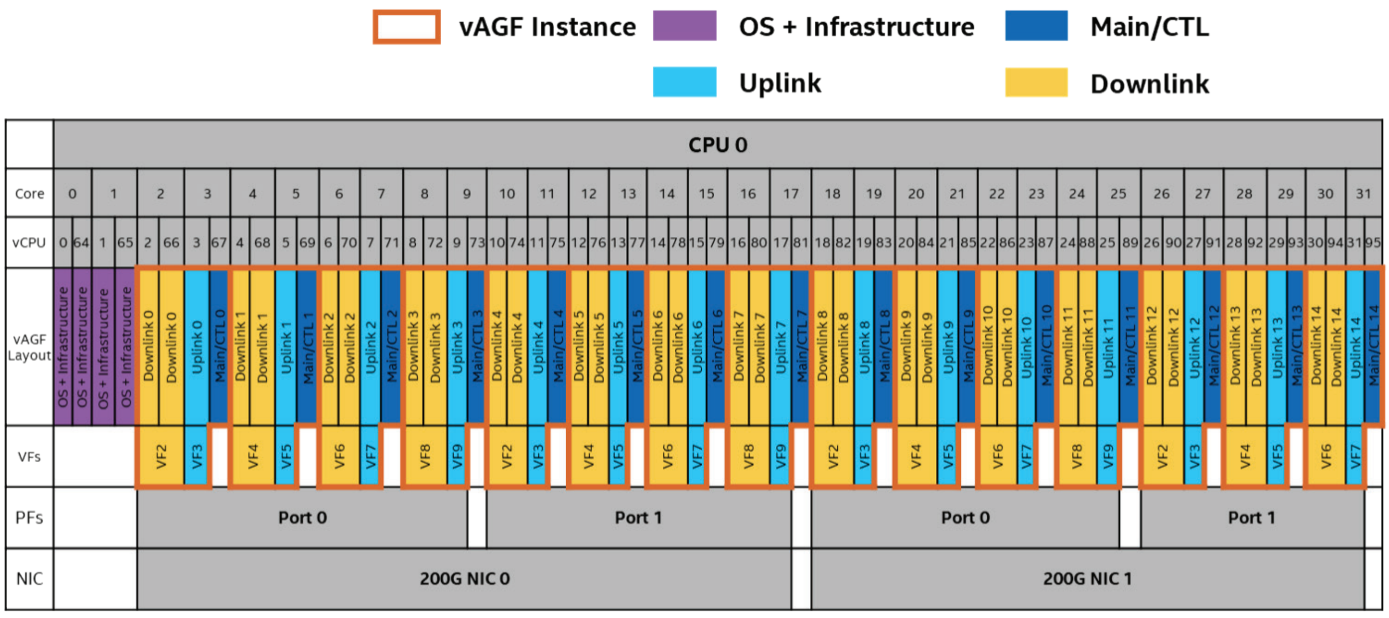


Figure 17. Application deployment on CPU0 (NIC 0 + NIC 1)

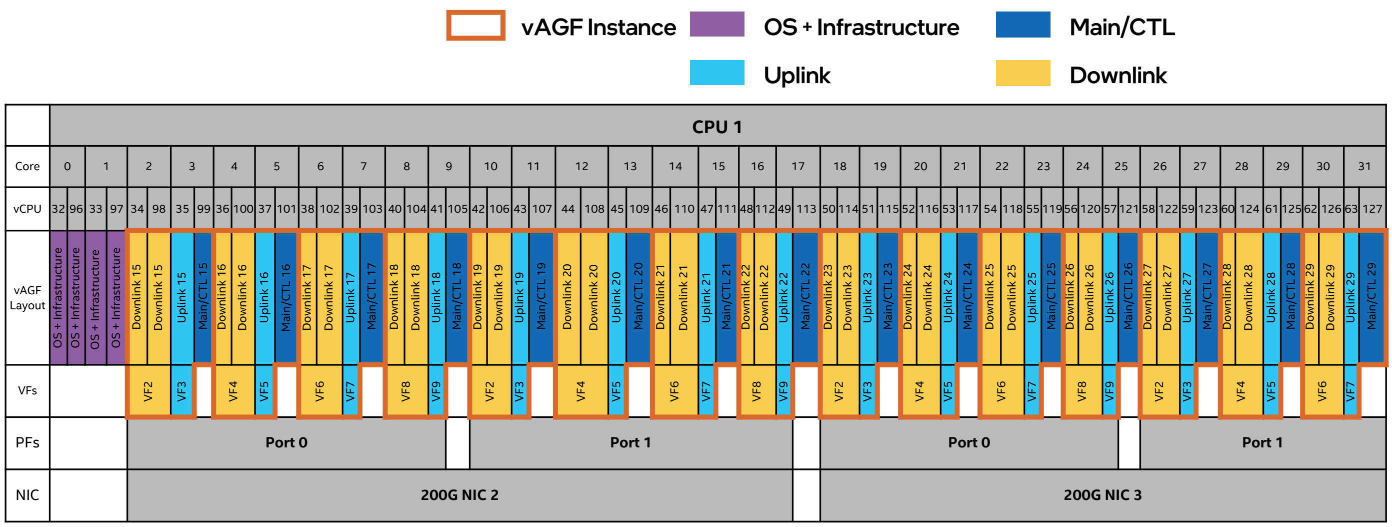


Figure 18. Application Deployment on CPU1 (NIC 2 + NIC 3)

As described in the configuration section, each Intel Ethernet E810-2CQDA2 port has 4 vAGF instances running on it, each served with 25 Gbps.

It can be noted in both Figure 17 and Figure 18 that on the final port of each CPU, there are only three instances. The reason behind this is because there are no remaining isolated cores on the processor and thus only three of a possible four instances can be deployed on this port.

As alluded to in Figure 10 and its description, for every Intel Ethernet E810 port, the first two VFs are used for infrastructure and control plane purposes respectively. Thus, for every physical port, the user plane VFs start at VF2.

Finally, the following are some notes on the test setup:

- The vAGF instances were deployed with Docker, but all resource allocation was done in a similar manner to K8s. For example, the K8s CPU manager does not allow containers to share VPP main vCPUs and thus this restraint was applied to the benchmarked containers.
- All data plane VFs were bound to the igb_uio driver to expose them to DPDK in user space.
- All DIMMs used during testing were 2133 MHz.

Test 1 – Symmetric Traffic

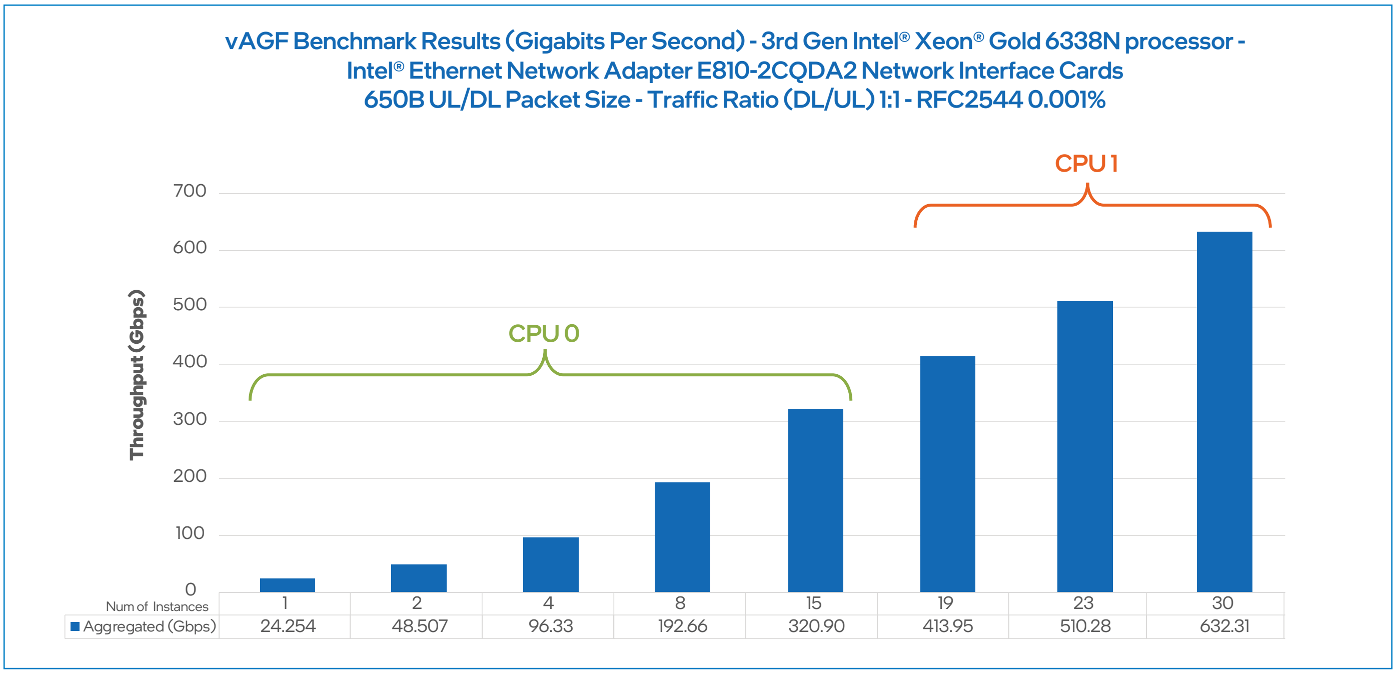
Under symmetric test conditions, each instance is allocated 25 Gbps (12.5 Gbps downlink max, 12.5 Gbps uplink max 1:1 ratio). RFC2544 throughput is executed on the downlink pipeline and uplink pipeline. If the RFC2544 test deems 12.5 Gbps to be too high for either downlink or uplink (i.e., >0.001 percent loss at this rate), it will back off to a lower throughput that matches or is below the required loss rate.

Symmetric traffic is used to show the true potential of the 3rd Gen Intel Xeon CPU. Under asymmetric test conditions (Test 2 below), the uplink pipeline is heavily under-utilized and thus provides a more complex representation of the performance per core. The symmetric test places the same test conditions (frame size + traffic ratio) on the uplink and downlink pipelines. From this it is easier to deduce the true processing power of an 3rd Gen Intel Xeon processor.

Table 2: Test Parameters for Symmetric Traffic Workload

Configuration Name		Configuration Setting
Test Parameters	Max # Instances	30
	Traffic / Instance	25 Gbps
	Traffic Ratio (DL:UL)	1:1
	Instances / NIC Port	4 (3 on final PF on each CPU)
	Subscribers / Instance	4096
	Frame Sizes (Tx from Tester)	DL: 650 bytes UL: 650 bytes
	RFC2544 Acceptable Frame Loss	0.001% of Line Rate
Tested by Intel on	23/09/2021	

Results:



Num of Instances	1	2	4	8	15	19	23	30
■ Aggregated (Gbps)	24.254	48.507	96.33	192.66	320.90	413.95	510.28	632.31

Figure 19. Intel® Xeon® Processor-Based Server (Dual Socket) RFC2544 Throughput Scale of the AGF Running A Symmetric Traffic Load (higher is better).

From the results in Figure 19, when running 30 instances of the vAGF with symmetric traffic on a Dual Socket Intel M50CYP Server hosting two 3rd Gen Intel® Xeon® Gold 6338N processors, one can achieve 632 Gbps of symmetric AGF traffic when running with RFC2544 throughput at 0.001 percent loss acceptance rate on both uplink and downlink.

Test 2 – Asymmetric Traffic

Under asymmetric test conditions, each instance is allocated 25 Gbps (22.25 Gbps downlink max, 2.75 Gbps uplink – 89:11 ratio). The uplink pipeline is run at a constant 2.75 Gbps and RFC2544 throughput is executed on the downlink pipeline only. If the RFC2544 test deems the 22.25 Gbps per instance too high (i.e., >0.001 percent loss), it will back off to a lower downlink throughput that matches or is below the required loss rate.

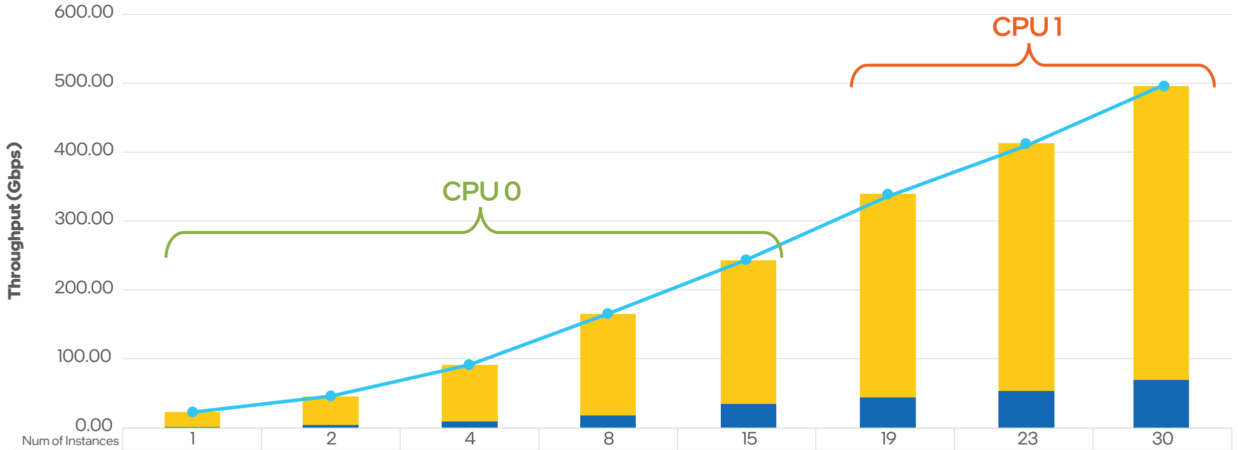
As described previously, the asymmetric workload can under-utilize the uplink vCPU, thus producing a lower performance per core result, but it is also a healthy signifier of how the AGF works under different traffic constraints without the need to change anything in the deployment. The AGF architecture tries to lend itself towards being flexible and agile so that it can fit many workload scenarios with minimal intervention.

Table 3: Test Parameters for Asymmetric Traffic Workload

Configuration Name		Configuration Setting
Test Parameters	Max # Instances	30
	Max Traffic / Instance	25 Gbps
	Traffic Ratio (DL:UL)	89:11
	Instances / NIC Port	4 (3 on final PF on each CPU)
	Subscribers / Instance	4096
	Frame Sizes (Tx from Tester)	DL: 504 bytes UL: 128 bytes
	RFC2544 Acceptable Frame Loss	0.001% of Line Rate
Tested by Intel on	14/09/2021	

Results:

vAGF Benchmark Results (Gigabits Per Second) - 3rd Gen Intel® Xeon® Gold 6338N processor - Intel® Ethernet Network Adapter E810-2CQDA2 Network Interface Cards 504B DL Packet Size / 128 UL Packet Size - Traffic Ratio (DL/UL) 89:11 - RF2544 0.001%



Num of Instances	1	2	4	8	15	19	23	30
🟨 Downlink	20.63	41.25	81.78	146.98	208.25	294.87	359.27	426.13
🟦 Uplink	2.29	4.58	9.17	18.42	34.93	44.05	53.29	69.80
● Aggregated	22.919	45.839	90.96	165.39	243.18	338.919	412.559	495.927

Figure 20. Intel® Xeon® Processor-Based Server (Dual Socket) Throughput Running Various vAGF Instances (higher is better).

From the results in Figure 20, when running 30 instances of the vAGF with asymmetric traffic on a Dual Socket Intel M50CYP Server hosting two 3rd Gen Intel® Xeon® Gold 6338N processors, one can achieve 495 Gbps of AGF asymmetric traffic when running with RFC2544 throughput at 0.001 percent loss acceptance rate on downlink only.

Conclusion/Summary

Technological advancements in 5G offer an opportunity for wireline-wireless convergence. Since most operators support both access technologies, a migration path to wireless and wireline convergence in the core will simplify provisioning and reduce capital and operations expenditures. 3GPP, the Broadband Forum, and CableLabs have collaborated on technical reports and specifications to define the services and systems required to support 5G wireless and wireline convergence. These reports specify the necessity of an Access Gateway Function (AGF) to deliver the functionality previously afforded by Broadband Remote Access Servers (BRASs) and Broadband Network Gateways (BNGs).

This paper presents the initial work on a software implementation and cloud-focused deployment of AGF on the Intel® Xeon® platform. The Container Bare Metal Reference Architecture (BMRA) ensures quick and scalable AGF deployment and helps in applying Intel's best known configurations to harness the power of servers with Intel® Xeon® processors and Intel® Ethernet Controllers for optimized telco data plane performance in line with industry expectations for BNG/AGF.

The future work will demonstrate specific technologies and software mechanisms running on the latest Intel® processors to help scale performance, life cycle management of AGF, intelligent power management policy, and cloud-native deployment.

[1] Study on 5G WWC (BBF SD-407) https://issues.broadband-forum.org/browse/CONTRIB-20777
[2] FDIO VPP https://fd.io/
[3] Kubernetes https://kubernetes.io/
[4] Device Config Function https://doc.dpdk.org/guides/nics/ice.html#device-config-function-dcf
[5] Dynamic Device Personalization https://www.Intel.com/content/www/us/en/architecture-and-technology/ethernet/dynamic-device-personalization-brief.html
[6] Hybrid Access Broadband Network Architecture (BBF TR-348) https://www.broadband-forum.org/download/TR-348.pdf
[7] 5G Wireless Wireline Convergence Architecture (TR-470) https://www.broadband-forum.org/technical/download/TR-470.pdf
[8] Re-Architecting the Broadband Network Gateway (BNG) in a Network Functions Virtualization (NFV) and Cloud Native World https://builders.intel.com/docs/networkbuilders/re-architecting-the-broadband-network-gateway-bng-in-a-network-functions-virtualization-nfv-and-cloud-native-world-1633374232.pdf
[9] BBF WT-458 AGF CUPS https://issues.broadband-forum.org/browse/CONTRIB-21163
[10] Intel® Ethernet Controller E810 (Formerly Columbiaville) https://www.intel.com/content/www/us/en/products/details/ethernet/800-network-adapters.html
[11] Data Plane development Kit https://www.dpdk.org
[12] BBF TR-456: AGF functional requirements https://www.broadband-forum.org/technical/download/TR-456.pdf
[13] Helm Charts https://helm.sh/docs/topics/charts/
[14] Container Bare Metal Reference Architecture https://networkbuilders.intel.com/solutionslibrary/container-bare-metal-for-2nd-3rd-generation-intel-xeon-scalable-processor
[15] Kubernetes CPU Manager https://kubernetes.io/docs/tasks/administer-cluster/cpu-management-policies/
[16] Kubernetes SR-IOV Device Plugin https://github.com/k8snetworkplumbingwg/sriov-network-device-plugin
[17] RFC2544 Testing Methodology https://datatracker.ietf.org/doc/html/rfc2544