WHITE PAPER

Communications Service Providers
Resource Scaling

QCT* Demonstrates VNF Scaling Using Autoscale 2.0

Updated autoscaling framework for Intel® Rack Scale Design supports multiple workloads and quality of service integration. Quanta demonstrates autoscale closed loop operation using QCT NFV infrastructure with ONAP* and ASTRI vEPC.*

Taking network functions virtualization (NFV) to the next level of deployment means fully utilizing the ability to scale composable resources (compute, storage, network) to match changing network traffic levels or workload demands. To facilitate the rapid and automated scaling of these resources, and support increased demand for a data center service, Intel has introduced automated scaling frameworks for the Intel® Rack Scale Design (Intel® RSD) data center server architecture.

With Autoscale 1.0, Intel and Quanta,* an Intel® Network Builders ecosystem member, worked together to demonstrate a solution that could respond to a request to increase the resources for a virtual evolved packet core (vEPC) service in less than seven minutes.[1]

This white paper describes how the two companies worked together to demonstrate the Autoscale 2.0 framework, which adds multiple workload support, quality of service monitoring and management, resource-aware workload scheduling, and other capabilities. The companies demonstrated these features in the deployment of a vEPC from ASTRI,* a Hong Kong-based research and development center.

Autoscale 1.0 Framework

The Autoscale 1.0 platform for Intel RSD focused on developing the remote ability to scale composable resources (compute, storage, network) to match changing traffic levels or workload demands. This autoscaling framework utilizes Redfish*-compliant application programming interfaces (APIs) to automate service scale up and scale down, including the identification of the right hardware resources required for the application.

To create and adjust composed nodes, Intel RSD utilizes management software and agents that include the pod manager (PODM), the rack manager module (RMM) and the pooled system management engine (PSME). Every hardware resource in a rack is assigned a PSME that works with the RMM to report availability of resources. The PODM communicates with the PSMEs via the southbound DMTF Redfish* APIs to build the inventory of resources that is exposed to an orchestrator via northbound Redfish APIs. The orchestrator can then request the PODM to compose nodes based on these standard management APIs to meet the application requirements.

To facilitate autoscaling of services deployed on Intel RSD pods, the Autoscaling 1.0 framework includes the following software elements:

Table of Contents

INFRAm Watcher is a thread that polls for composed and assembled nodes and communicates with the PODM to maintain an inventory of these compute resources. The INFRAm Watcher also watches for key workload performance metrics.

The **resource autoscale daemon** (rASD) is a resource thread that serves as a control function, taking in requests for increased resources from the INFRAm Watcher and communicating them to the **workload autoscale daemon** (wASD). The wASD is composed of two parts: the wASD handle and workload daemons. The wASD handle instantiates the composition of a next-generation integrated core (NGIC) on a particular compute node based on its resource inventory. The wASD workload daemons are installed in each resource module to launch the new operating system and then provision the VNF on the NGIC to start the service.

Autoscale 2.0 Framework

With Autoscale 2.0, Intel builds on the features of Autoscale 1.0 to deliver resource-aware scheduling and quality of service (QoS) support on platform resources using Resource Management Daemon. In addition, gRPC has been integrated into the solution to deliver better performance for real-time applications.

Resource Management Daemon

Low latency and real-time workloads are sensitive to resource contention on a compute node. As the density of the software that runs on the platform increases, software components like VNFs, containers, or infrastructure components, like a virtual switch/router, often compete for hardware resources in situations such as network traffic bursts, and this can have significant impact on performance and service availability. To facilitate greater workload packing and to handle changes in the workload characteristics or priorities, finer-grain dynamic resource control at sub-second time granularity on the compute node is valuable.

Since VNFs use a shared platform, hardware partitioning/isolation can also be critical. Such partitioning enables each application to get the resources it needs when it needs them and not be adversely affected by other VNFs. This interference by other VNFs is sometimes called the noisy neighbor problem.

Orchestrators today don't have the ability to provide the above features on each compute node. This has led to the requirement of having a local node agent like Resource Management Daemon.

Resource Management Daemon has a holistic view of resources for the workloads deployed on a node and provides the following features:

- Provides finer-grain local control of resources through short-term monitoring on the local node and the ability to make dynamic adjustments to existing allocations to workloads consistent to user requests.
- Provides high-level abstraction of allocation requests of resources to workloads on the local platform.
- Associates QoS metadata with resource allocation requests to workloads.
- Provides hooks into the platform to address different interfaces for the same or multiple resources. This results in vendor or platform independence from control plane.

Resource Management Daemon in Autoscale 2.0 Architecture

The Resource Management Daemon can be applied to an appliance model or can work alongside orchestrators such as OpenStack* in the control plane. Autoscale 2.0 integrates the Resource Management Daemon to leverage QoS and resource-aware slicing for workloads on two platform resources: the last-level cache and memory bandwidth. The Autoscale 2.0 utilizes the Resource Management Daemon OpenStack integration framework to specifically manage OpenStack workloads.

For More Information

Resource Management Daemon

Resource Management Daemon currently supports two platform resources from the Intel® Resource Director Technology (Intel® RDT): last-level cache (LLC) using Cache Allocation Technology (CAT) and Memory Bandwidth Allocation (MBA), currently available on selected CPU SKUs. For more information, visit https://www.intel.com/content/www/us/en/architecture-and-technology/resource-director-technology.html and https://github.com/intel/rmd.

QCT co-developed the MBA features into the Resource Management Daemon.

Intel® Resource Director Technology (Intel® RDT)

Read "Deterministic Network Functions Virtualization with Intel® Resource Director Technology" for more information on Intel® RDT and its use to achieve service assurance, predictable latency, and improved throughput for NFV deployments: https://builders.intel.com/docs/networkbuilders/deterministic_network_functions_virtualization_with_Intel_Resource_Director_Technology.pdf.

Resource Management Daemon OpenStack* Integration Framework

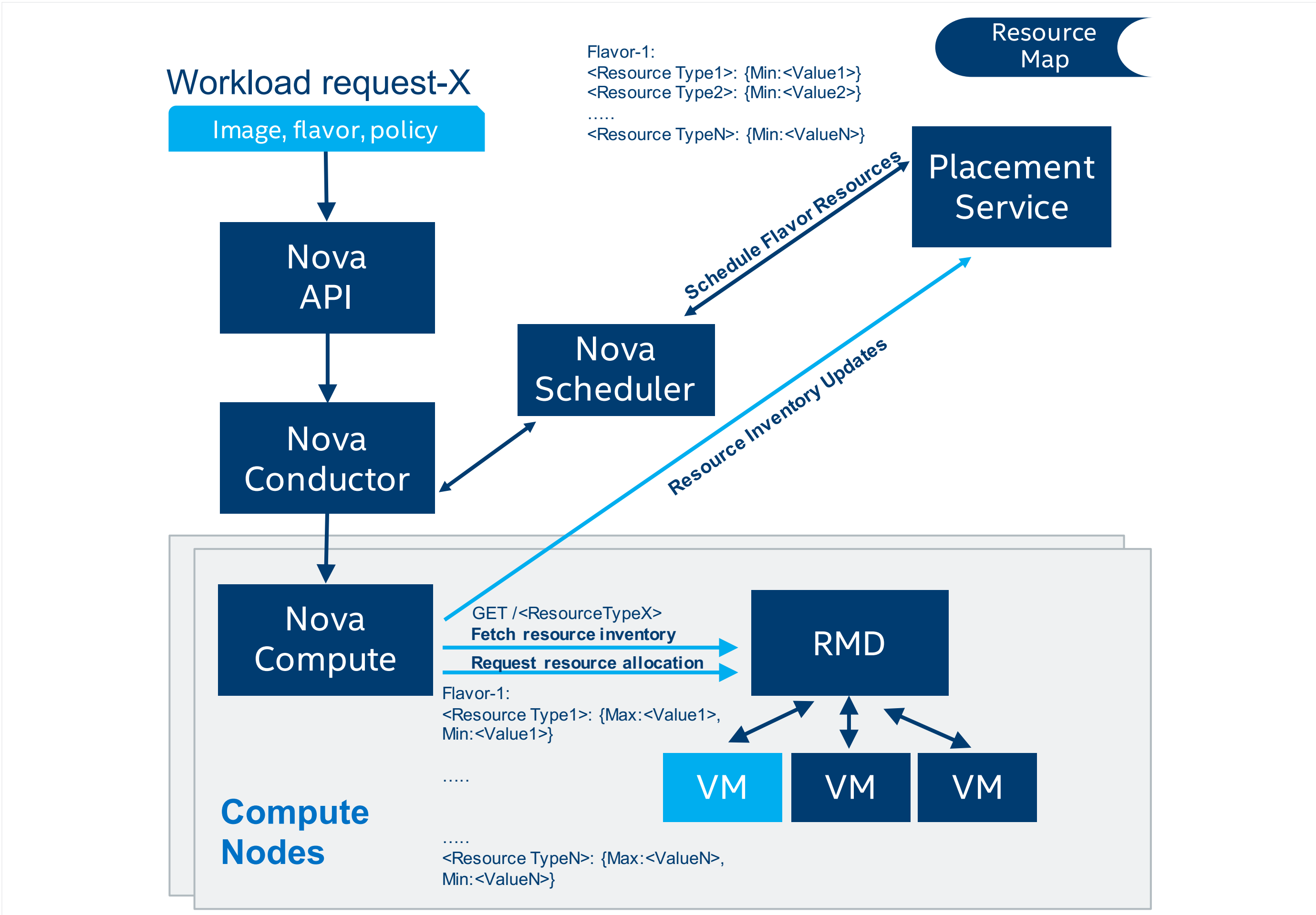

Figure 1. OpenStack Resource Management Daemon integration.

OpenStack's interaction with the physical node is limited to characterizing nodes and allocating resources to a virtual machine (VM) to improve the placement of the VM. Runtime handling of node specifications is not part of the core functionality in the OpenStack Nova* compute component. Acting as an external component to OpenStack, Resource Management Daemon offers the functionality to manage different node capabilities and the ability to configure resources using node policies offered by sys-admin and VNF policies from the VNF manager (VNFM).

This model is achieved through interactions between the virt driver in OpenStack Nova compute, which is responsible for provisioning the VM using Resource Management Daemon. The virt driver queries the Resource Management Daemon for capabilities and the inventory of the platform resources it supports. This information aids the scheduler to determine the appropriate compute node that has the capacity to host resources requested by the user and supported by Resource Management Daemon. User requests are handled the traditional way from the OpenStack control plane but will now include details for resource requests to Resource Management Daemon. Upon initialization of a VM, the virt driver interprets the user request and, in turn, makes a request from the Resource Management Daemon for allocation of resources from its inventory. Local fine tuning of platform resources via any dynamic re-allocations done by Resource Management Daemon must also guarantee the minimum resource allocation amounts allotted to the VM.

Intel in collaboration with QCT among others co-authored the design specification for this framework.

gRPC

To respond to any traffic change event in real time, the management and network orchestration (MANO) software has to detect and send signals to the cluster controllers to start a new VNF instance before existing instances get exhausted. To address this challenge in Autoscale 2.0, Intel introduced gRPC remote procedure call. gRPC communicates complex command signals to the compute platform. It helps facilitate communication between the MANO software and cluster nodes because it is a well-versed and low-latency communication channel for signaling between traffic event trigger and the start of a new VNF instance. The previous generation of autoscaling prototype used AF_INET-based socket communication between its rASD and wASD components, which was limited by the types of command signals and VNF workload details as it was using simple sockets in a complex way.

Autoscale 2.0 in Operation

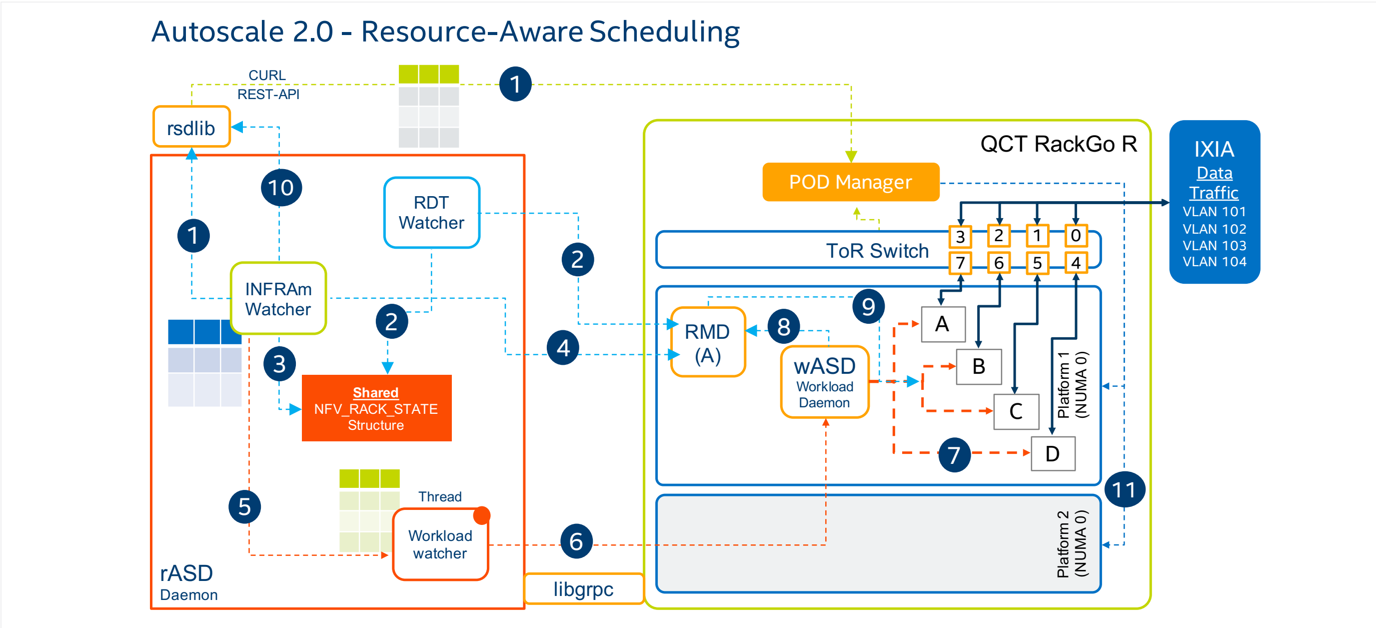

Figure 2. Resource-aware scheduling information flow.*

The Autoscale 2.0 architecture proposes flows to schedule and scale VNF workloads on a rack-based network traffic telemetry and traffic threshold defined per workload. Resource Management Daemon is deployed on every new platform provisioned by the PODM on the rack. Resource Management Daemon enables QoS on cache and memory bandwidth while Intel RDT monitors the resources of workloads by efficiently slicing and allocating platform resources.

Below is an explanation of the data flows specified by the Autoscale 2.0 architecture that are also shown in Figure 2:

1. INFRAm Watcher queries for port virtual LAN statistics from the top of rack switch using rsdlib.
2. The Intel RDT watcher queries Intel RDT inventory information from the local Resource Management Daemon deployment on each platform in the rack. Intel RDT inventory information shows available capacity to allocate LLC or memory bandwidth and is retained at the rack level in NFV_RACK_STATE structure.
3. NFV_RACK_STATE structure is updated with platform odataID.
4. The desired platform is determined based on the Intel RDT capacities reported in the NFV_RACK_STATE structure. Before scheduling the workload on a platform, the Resource Management Daemon hospitality score for Intel RDT allocations is verified against the local Resource Management Daemon deployment on the target platform. The hospitality score is a litmus test that verifies if the workload's Intel RDT resource requirements can be met by the target resource.
5. Now, the workload watcher starts, along with the respective gRPC socket. As part of this process, a table is shared with platform ID, gRPC socket ID, wASD ID, and odataID.
6. Workload watcher starts the gRPC socket using libproto with the respective wASD.
7. wASD starts the workload type/instance based on the VLAN ID and the threshold throughput is defined for each workload type (A, B, etc.).
8. wASD (virsh/Docker* local process start/stop) makes the local call to Resource Management Daemon for a PID/Core info exchange.
9. Resource Management Daemon enforcement for Intel RDT resources is completed for a workload represented by PID/core ID-based on a policy describing the desired Intel RDT resource requirements.
10. After the periodic check of the resources to deploy the new workload, INFRAm starts new platform through rsdlib.
11. PODM starts a new platform.

Autoscale vEPC Demonstration

To demonstrate these new autoscale features, Intel and Quanta developed a demonstration in an Open Network Automation Platform (ONAP)* environment using ASTRI vEPC as the workload with Red Hat* OpenStack 13 providing NFV infrastructure (NFVI) services. This was a closed loop automation framework deployment with ONAP-compliant telemetry system based on collectd.

ONAP provides a comprehensive, open source platform for the designing, creating, orchestrating, and handling of VNFs. ONAP orchestration manages the full lifecycle of a VNF and delivers real-time, policy-driven orchestration and automation that allows developers to rapidly automate new services from service instantiation to service teardown.

For better management, the QCT NFVI was integrated with ONAP as the MANO solution. For stability, the demonstration was developed on ONAP Beijing version.

The two VNFs that were used in the demonstration were:

- **ASTRI vEPC VNF** is a 3GPP*-compliant vEPC with all of the standards-defined functionality, including a mobility management entity (MME), packet gateway (PGW), serving gateway (SGW), and system management. The vEPC utilizes the open source Data Plane Development Kit (DPDK) software library for performance. The system features a control and user plane separation (CUPS) architecture, which implements separate control plane and data plane VMs. This allows independent scaling of each VM to support user needs.
- **QCT Telemetry VNF** leverages a customized version of collectd as a memory and compute data collection agent for providing all platform telemetry for service orchestration and monitoring/analytics systems. Collectd gathers, transfer,s and stores performance data of computers and network equipment. QCT Telemetry is a major part of the QCT service assurance solution used to automate the performance and visibility for OpenStack-based NFVI and VNF applications.

As seen in Figure 3, the hardware configuration for the demonstration consisted of Quanta QCT Rackgo R* servers and a 10 GbE top of rack (TOR) switch.

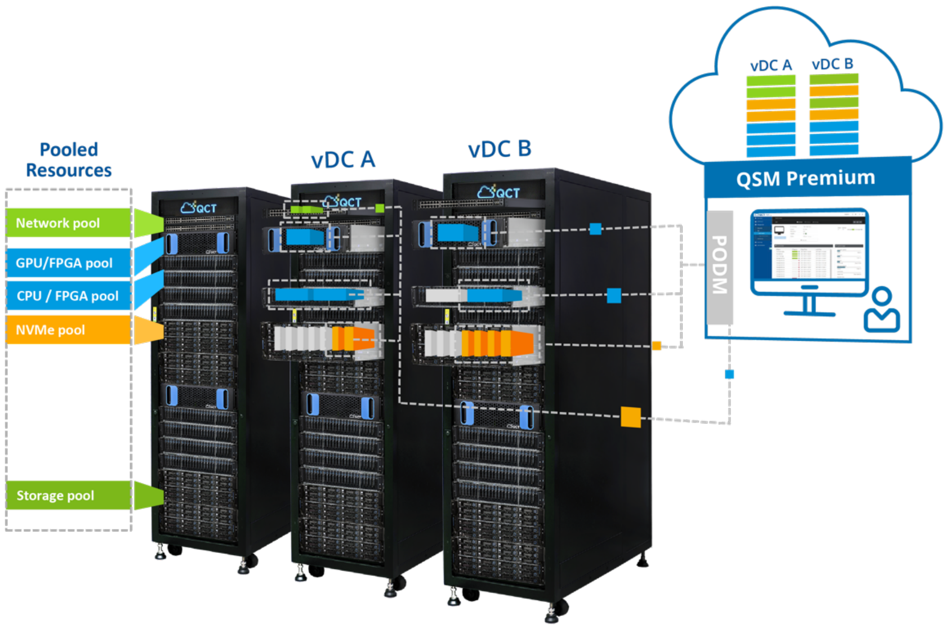

Figure 3. Quanta Rackgo R system based on Intel RSD.[2]

Management of the Rackgo R systems provides all network, compute, and storage resource configurations through a single management screen, simplifying administration. As virtual environments diversify and become more complex, Rackgo R provides a solution to manage heterogeneous virtual environments with its own complete resource composition and control solution.

With this platform, QCT designed the VNF deployment demonstration that is described in the steps below and is shown in Figure 4.

The closed loop automation steps of QCT vEPC use case were as follows:

- At the start of the demonstration, an ASTRI vEPC instance was running with just one control plane instance and one data plane instance.
- A traffic generator was used to simulate an increase in the number of subscribers and an increase in the data traffic flow.
- The vEPC data plane instance then reported the increased number of subscribers and the increasing traffic levels to the control plane.
- The control plane was polled by collectd, which sent the performance metrics to the ONAP data collection, analytics, and events (DCAE) engine via the VNF event stream (VES).
- The traffic generator continued to simulate more subscribers and traffic.
- When DCAE sensed the subscriber number, or bandwidth consumed, had reached the high critical threshold, DCAE sent this event information to ONAP's data movement as a platform (DMaaP) function.
- The Intel RSD policy engine then processed this event and sent the scale-out event notice to DMaaP.
- The ONAP application controller (APPC) processed this scale-out event and sent a command to the ONAP infrastructure controller for the scale-out operation.
- The infrastructure controller called an OpenStack API to launch a new data plane instance.
- The new vEPC data plane instance registered on the control plane instance and joined the EPC service.
- As the traffic generator continued to increase the number of subscribers, and the traffic levels from those subscribers, the same scaling operation was repeated.

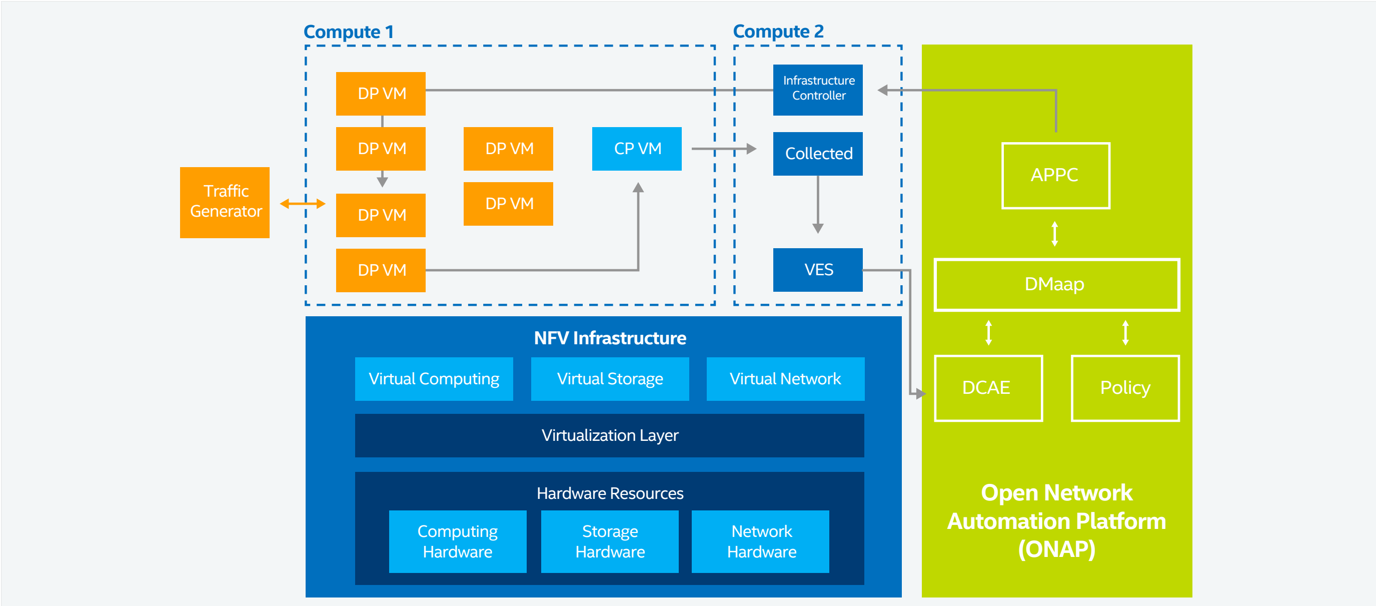

Figure 4. Autoscale closed loop work flow.

With its telemetry framework, QCT was able to rapidly create the data link from the physical infrastructure to the NFVI and core network VNF, to DCAE engine.

The two key processes that were the focus of this use case were design time and run time. For the design time, the flow is illustrated as follows:

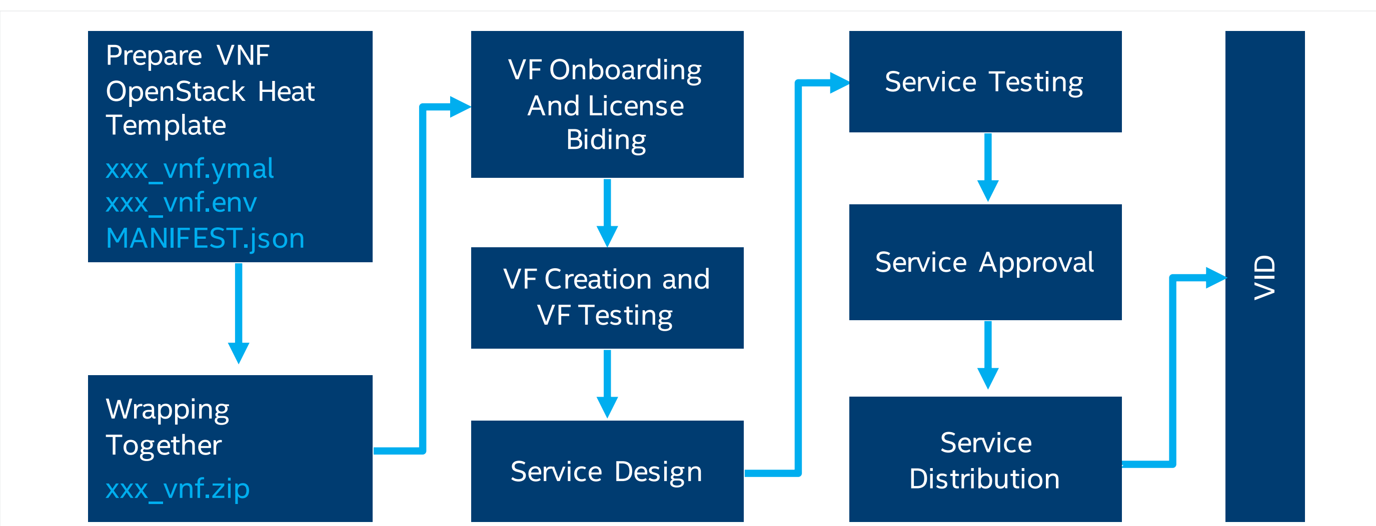

Figure 5. Auto-scaling closed loop automation—design time.

Below are the steps in the design time flow featured in Figure 5:

- The vEPC OpenStack Heat template is onboarded to ONAP through ONAP service design and creation (SDC).
- After onboarding the VNFs, VNF descriptions are created that are then used to create a virtual function component (VFC) for each of the VNFs.
- The VFC then composes the vEPC services.
- After the VFC is validated and distributed by the SDC, the vEPC service description is available in the ONAP virtual infrastructure deployment (VID) module.
- The vEPC service can be launched by an administrator with access to the VID module.

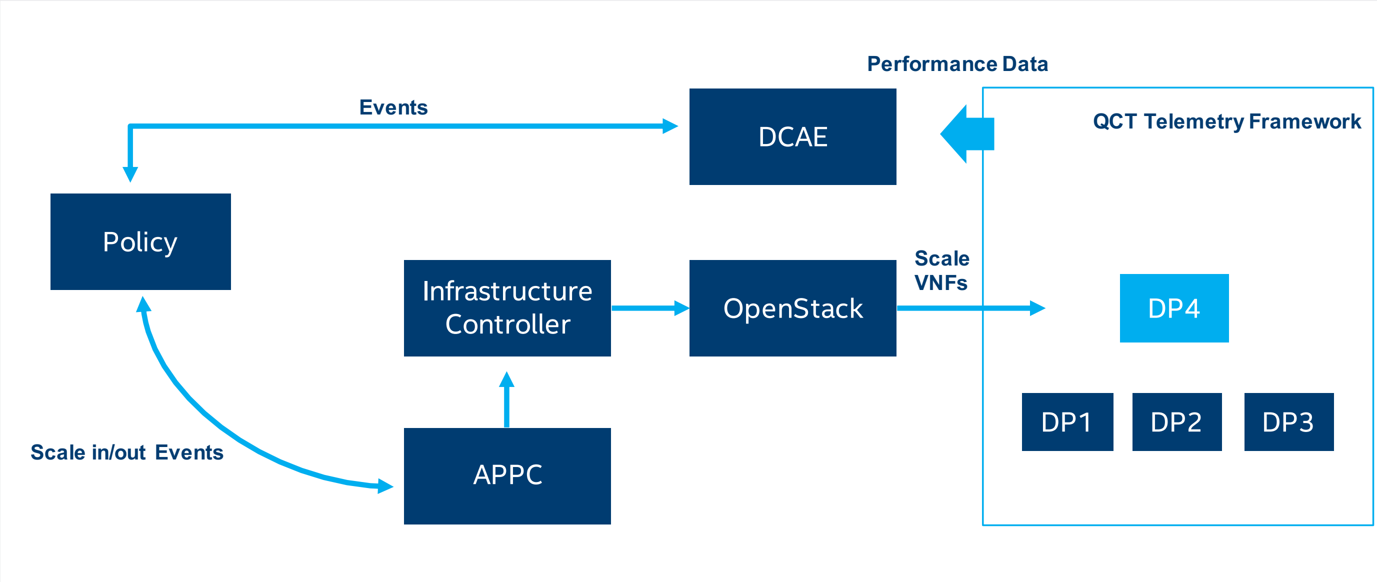

Figure 6. Auto-scaling closed loop automation—run time.

Figure 6 shows the runtime message flow of the EPC use case.

- DCAE is used to receive the performance statistics from VNFs and analyze the system behaviors and resource availability of the EPC service.
- If any abnormal behavior is detected by the DCAE, it sends the event to the DMaaP to access the policy that determines the scale-in or to scale-out actions to be taken on the VNF.
- Once the policy has determined the action, the APPC begins to scale in or scale out VNF.
- The APPC then accepts the command from the policy and informs the QCT infrastructure controller of the course of action.
- The QCT infrastructure controller triggers the virtualized infrastructure manager (VIM) to launch or terminate the designated VNF.

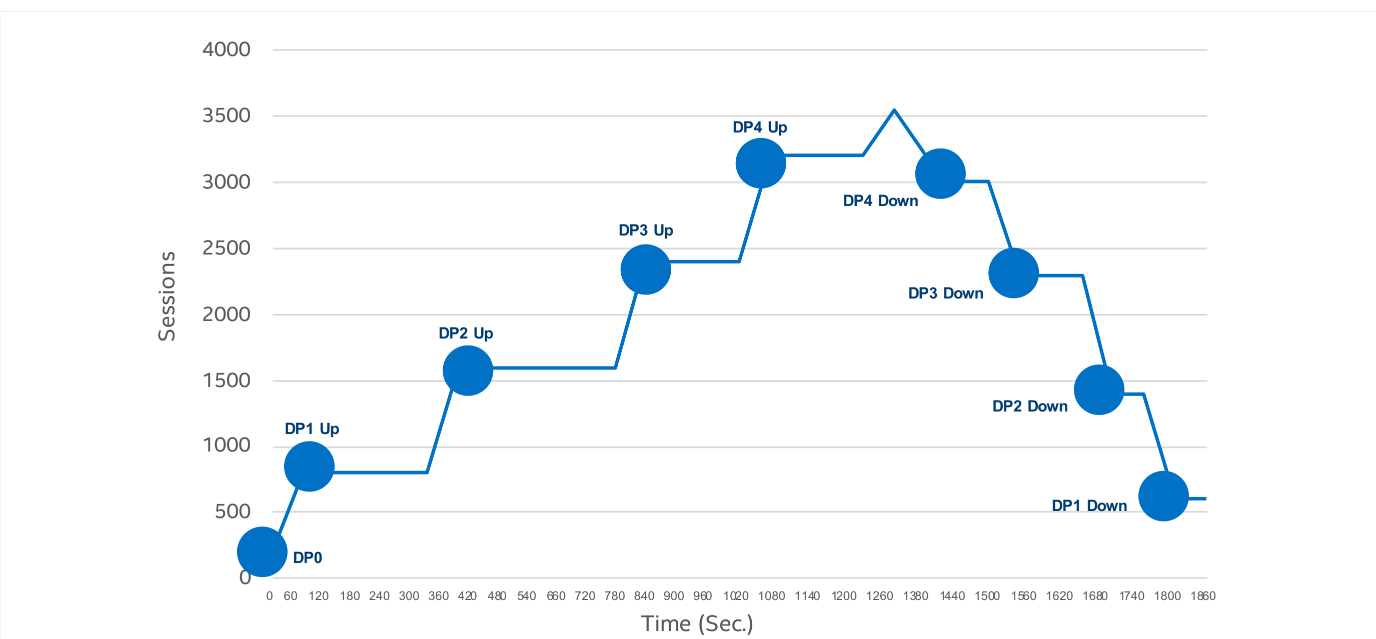

Figure 7. Number of data planes automatically added or deleted with every 800 sessions.[3]

Figure 7 shows how the autoscaling capability reacted to changes in user sessions and data traffic. The number of vEPC data planes increased or decreased as traffic grew or declined by 800 sessions.[3] The closed-loop demonstration was scaled from one data plane to four to serve 3,200 sessions. Also, when the session count decreased, ONAP scaled back the data plane services to release the NFV resources.

Next Steps

The work done to date on Autoscale 2.0 demonstrates that orchestrating VNF-based services through an integrated management system is an optimal way to automate application deployment in response to changes in user demand for those services. The key addition to this platform is a mechanism for constantly monitoring resource utilization and hardware and software events. This system statistic collection and analytics method was designed to be used to dispatch new applications and to be able to trigger application termination actions to keep the system healthy and efficient. Intel's and Quanta's work with Autoscale 2.0 and vEPC has shown the feasibility of this type of orchestration and closed-loop automation.

The focus of future work on autoscaling will revolve around expanding ONAP's role in automatically orchestrating VNFs through an integrated management system that better understands hardware and network resource availability. ONAP's DCAE functionality could be used to collect and analyze data from the infrastructure. Then, the service orchestrator and Autoscale 2.0 functions can ensure that orchestration policies are informed by this data. Other functions, such as platform for network data analytics (PNDA) and cask data application platform (CDAP), can be integrated into Autoscale 2.0 to discover abnormal events and alert ONAP in order to orchestrate a termination of problematic VNFs.

In the future, it should be possible to design and implement the closed-loop automation scenario using this complete platform data to showcase an ONAP-based autoscaling framework that can automatically invoke application and hardware orchestration and configuration without any human intervention.

Conclusion

Autoscaling of VNFs on the Intel RSD platform is a big advancement for software-driven data center services, adding a significant increase in service agility through automation. Intel RSD is based on the concept of composable compute, storage, and memory resources. By developing the Autoscale 1.0 framework, Intel provided a way to automatically scale the amount of these composable resources dedicated to a workload to match changes in demand for that workload.

With this framework in place, Intel created Autoscale 2.0 by integrating Resource Management Daemon for resource-aware scheduling and QoS support and gRPC for support of real-time applications. To demonstrate how resource-aware autoscaling works, Intel teamed up with Quanta to apply the Intel RSD Autoscale framework to effectively scale vEPC workloads, using the Intel RSD-based QCT Rackgo R environment in ONAP. The companies are continuing to develop this technology so that CommSPs can eventually apply this technology and experience to their own network services to improve service deployment agility.

About Quanta Cloud Technology (QCT)

QCT is a global data center solution provider. It combines the efficiency of hyperscale hardware with infrastructure software from a diversity of industry leaders to solve next generation data center design and operation challenges. QCT serves cloud service providers, telecoms, and enterprises running public, hybrid, and private clouds. Product lines include hyperconverged and software-defined data center solutions as well as servers, storage, switches, and integrated racks with a diverse ecosystem of hardware component and software partners. QCT designs, manufactures, integrates, and services cutting-edge offerings via its own global network. The parent of QCT is Quanta Computer, Inc., a Fortune Global 500 corporation. http://www.QCT.io.

About Intel® Network Builders

Intel® Network Builders is an ecosystem of infrastructure, software, and technology vendors coming together with communications service providers and end users to accelerate the adoption of solutions based on network functions virtualization (NFV) and software defined networking (SDN) in telecommunications and data center networks. The program offers technical support, matchmaking, and co-marketing opportunities to help facilitate joint collaboration through to the trial and deployment of NFV and SDN solutions. Learn more at http://networkbuilders.intel.com.

ACRONYMS			
API	Application programming interface	DMaaP	Data movement as a platform
APPC	Application controller	DMTF	Association formerly known as the Distributed Management Task Force
AS 2.0	Autoscale 2.0	gRPC	Remote protocol call
CAT	Cache allocation technology	HTTPS	Hypertext transfer protocol secure
CDP	Code and data prioritization	RDT	Resource Directory Technology
CLOS	Class of service	LLC	Last level cache
CMT	Cache monitoring technology	MANO	Management and network orchestration
DCAE	Data collection, analytics, and events	MBA	Memory bandwidth allocation

MBM	Memory bandwidth monitoring	RMM	Rack manager module
NGIC	Next-generation integrated core	SDC	Service design and creation
ONAP	Open Network Automation Platform	SDNC	SDN controller
PAM	Pluggable authentication module	VCF	Virtual function component
PID	Process identification number	vEPC	Virtual evolved packet core
PODM	Pod manager	VID	Virtual infrastructure deployment
PSME	Pooled system resource manager	VM	Virtual machine
QoE	Quality of experience	VNF	Virtual network function
QoS	Quality of service	VNFM	Virtual network function manager
rASD	Rack manager module	wASD	Workload autoscale daemon

[1] Testing conducted by Intel as of February 2018. Configurations of device under test (DUT): QCT Rackgo R, Hyper-Threading, Intel® Virtualization Technology (Intel® VT) for Directed I/O (Intel® VT-D) SR-IOV enabled on Intel® Xeon® Gold 6152 processor 22C @2.10 GHz with Intel® Ethernet Converged Network Adapter X710-DA4 NICs (4x10 GbE ports); Host OS Ubuntu* 16.04.3 Linux* version: 4.4.0-87; Guest OS Ubuntu 16.04.3 Linux version: 4.4.0-104. Configurations of ng4T* traffic platform: Intel® Xeon® CPU E5-2695 v4 @ 2.10GHz; 16GB Micron* DDR4-2400 RDIMM, 4 channels; 2 DIMMs per channel, 128 GB total. For more information, see the white paper "QCT* Tests Autoscaling Capability of Intel® Rack Scale Design" at https://builders.intel.com/docs/networkbuilders/quanta-tests-autoscaling-capability-of-intel-rack-scale-design.pdf.

[2] Image courtesy of QCT.

[3] Testing conducted by QCT in October 2018. Configurations: QuantaGrid D52B-1U 2.5 Tiered (3PCIe) with Intel Xeon 6152 processors (22 core, 2.1GHz); 32GB 2666MHz DDR4 RAM and Intel SSD 480GB SATA and two-port Intel Ethernet Network Adapter XXV710-DA2 (PCIe 25G). Host operating system Ubuntu 16.04.3 and Astri EPC