From Virtual Appliance to Cloud Native VNF

Requirements for a Cloud Native Network Function

By: GigaSpaces Research, Cloudify Team
Contributors: Nati Shalom, Yoram Weinreb

Table of Contents

Introduction

Many of the existing network functions, such as routers, firewalls, load balancers and such, have undergone the initial transition from a physical appliance to a virtual appliance. That transition required mostly performance optimization to accommodate the additional I/O overhead of the hypervisor and some configuration changes to accommodate the fact that a VM can be more dynamic in nature.

This shift to NFV, which is basically a cloud-based data center, has revolutionized the way network functions can be delivered. The transition to a Cloud Native world is considered far more disruptive as it touches changes in both the architecture, to accommodate hyper-scale and multi-tenancy, as well as the business model, which needs to be more consumption based, rather than fixed.

This paper will dive into the main requirements that differentiate a cloud native network function from the traditional network function, and, after making the leap from non-virtualized to virtualized network functions, what is then required to achieve cloud-native capabilities, along with the challenges and benefits of this transition.

Read More: NFV and What it Means to You

Requirements for a Cloud Native Network Function

The main requirements that differentiate a cloud native network function from the traditional network function are centered around ***self-management, and scale***.

A cloud native VNF also needs to fit within a DevOps environment and expose APIs to control all of the aspects of the VNF by other software, and not just by a human operator. That includes DevOps processes to allow continuous upgrades of the service without any downtime.

Below, listed in more detail, are the primary characteristics that are expected from a Cloud Native VNF:

Self-Management

- Built-in automation for the installation and configuration of the VNF
- Multi-cloud support (at the very least for the most common clouds - VMware, OpenStack, AWS, Azure, GCP)
- Container support can be seen as lightweight VMs, as well as a new packaging model.
- KPI monitoring - Built-in monitoring becomes a critical piece to detect failure, performance and capacity issues.
- Analytics - Provide insight into the service behavior and SLAs as well as a means to trigger actionable insights in the case of failure or scaling issues.
- Self-healing (aka remediation) - This should enable the handling of common failure use cases without any human intervention or with light intervention in cases where there's a need to control the recovery process.
- Auto-Scaling - Enables the increase of capacity of a network function in a fully automated fashion or with light intervention
- Policy Management - Provides a more generic way to trigger workflows based on insights from the network function behavior metrics.

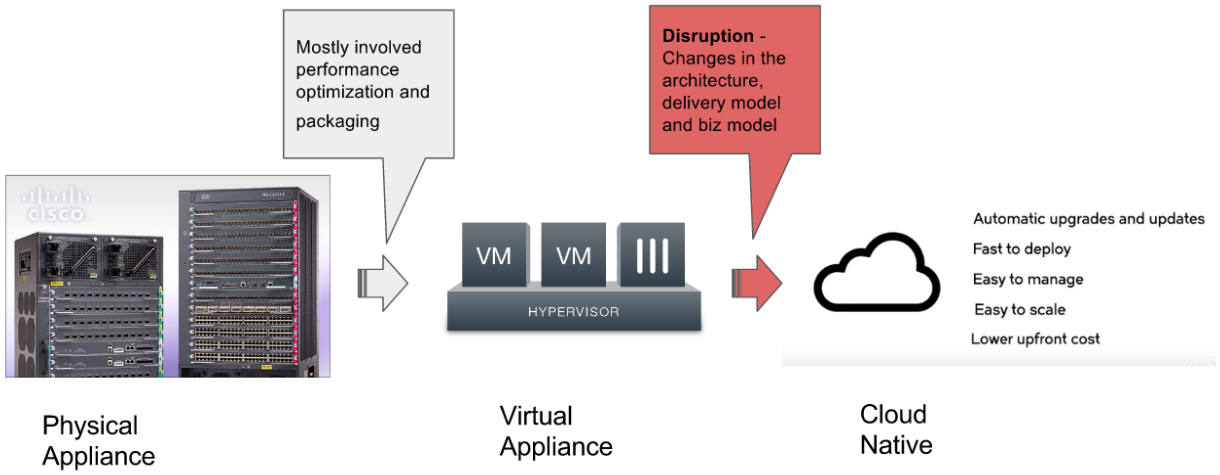
Multi-Tenancy

Multi-tenancy comes with a direct correlation to cost. By allowing multiple customers/users to share the same VNFs we can reduce the cost per customer significantly.

Software-Defined

The VNF needs to be controlled by other software, not just the operator, for the purpose of service chaining and continuous updates and deployment.

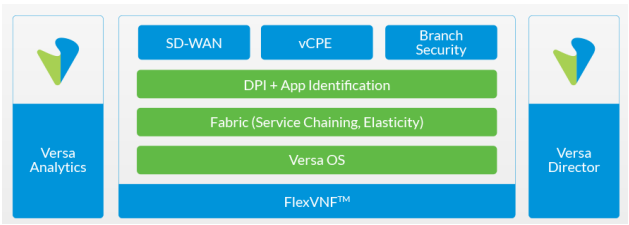
The Cloud Native Disruption

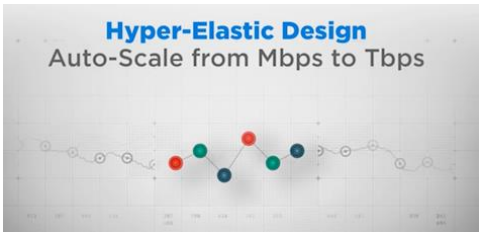


The Cloud Native Disruption

As already mentioned, where many existing network functions have gone through the first transition from a physical appliance to a virtual appliance, the transition to a Cloud Native world is considered far more disruptive, and, on top of that, each network provider needs to support multiple cloud environments such as OpenStack, VMware, AWS and Azure as well as bare metal.

A new generation of Cloud Native VNFs, such as those offered by Versa Networks or Ruckus Wireless, are great examples of the networking world embracing cloud technology to deliver complex networking services faster and cheaper.





Versa Networks comes with a self-managed VNF comprised of a built-in analytics, and orchestration

Ruckus SaaS enabled VNF

Ruckus Wireless' SaaS enabled VNF comes not only with built-in self-management, but its Wi-Fi controller has been delivered as a cloud service as well.

The common thing to note about both examples is, that, as we move to a ***virtual*** network function, the management piece, which formerly with a traditional network function was just an enabler that provided web-based configuration and control over the network function for the most part, has now become a central piece in the architecture that handles not just the configuration but the deployment, scaling, upgrades, and essentially the entire service's lifecycle.

The move to a container-based architecture imposes additional challenges in a number of areas:

- Packaging - They're mostly packaged as single tenant VMs
- Management - Cluster management, auto-scaling, and self-healing require a lot of customization which quite often involves manual intervention
- Cloud Support - Containers do not come with built-in cloud support. Therefore, deploying the network function on each target cloud requires custom integration and support.
- VNF Support - Most of the existing network functions are not packaged into containers, and, since containers are still going through maturity cycles, they aren't ready to be the only delivery model
- Licensing Model - Most of the existing VNFs come with fixed license models that do not fit well with consumption-based business models of cloud native applications.

Adding Cloud Native Support for Existing and New VNFs

When we examine the requirements for Cloud Native VNFs, it becomes clear that majority of these can be made generic and are not VNF-specific. Such requirements include automation, auto-scaling, self-healing and can be handled through any generic NFV orchestrator.

Obviously, there are also things that requires changes at the core VNF layer, but, since they tend to be implementation-specific, they will not be covered in this paper.

The Challenges with Existing NFV Orchestration

While many of the requirements are generic, many of the existing NFV orchestrators that could potentially be used to cover the most common use cases were built to run as data center services, not embeddable services that can be integrated as part of the VNF implementation and white labeled. In addition, these orchestrators often consist of a fairly heavy software stack, as in the case of Amdocs or Netcracker, just to name a couple. So, any attempt to include them as part of the VNF distribution would result in huge complexity.

Embedding a third-party orchestrator also requires business alignment with the VNF provider's business model. The challenges with some NFV orchestrators such as the one delivered by Cisco or Juniper Networks also come with built-in conflicts of interest with other network service providers. Others come with a fairly heavy cost structure that limit the cost margin for a particular VNF provider.

These business misalignments can be a much bigger challenge than the technical ones, see: Where AT&T Leads, Cisco Cannot Follow. (Interestingly enough by the time of writing these lines it been announced that Cisco Is planning a layoff of 14,00 employees!)

Cloudify & ARIA: Open and Embeddable NFV Orchestration

Cloudify was born as an open-source orchestration framework from the get-go. It was originally embedded with the Alcatel CloudBand orchestration platform, and that influenced the design for embeddability, and the OEM model, with the introduction of the third generation of Cloudify.

ARIA was born out of Cloudify and designed to provide a very lightweight and simple TOSCA orchestration engine that is open source and open governance under the Apache Software Foundation.

VNF providers can choose either ARIA or Cloudify depending on the value they intend to gain from cloud native capabilities.

ARIA: Lightweight Orchestration Engine

VNF providers should consider ARIA to gain TOSCA support, automation of deployment and configuration across multiple clouds such as OpenStack, VMware, or any other cloud, as well as portability between different container-based architectures such as Docker Swarm or Kubernetes.

Cloudify Manager: Continuous Management, Monitoring and Security Support

Cloudify includes ARIA as its core orchestration engine, so by definition, it inherits all of ARIA's capabilities in addition to the capabilities that are provided by Cloudify Manager.

VNF providers should consider Cloudify in cases where they would want to gain more advanced management, ongoing automation and security such as:

- Self-healing
- Auto-scaling
- Monitoring
- Policy Management
- Security and Multi-Tenancy

Multi-Tenancy

In a cloud native world, multi-tenancy is required in order to maximize the utilization of a VNF service across multiple users and organizations, thus reducing cost per user. To do that, multi-tenancy allows individual users to get a "slice" of a VNF as if it was dedicated to them, by ensuring complete isolation with other users.

The challenge with multi-tenancy is that achieving fine-grained multi-tenancy often requires changes to the architecture and code of the existing VNF.

Albeit, there are ways in which we can still achieve cross-grain multi-tenancy by assigning each tenant to a container or an infrastructure tenant, and, with that, achieve maximum isolation and a high enough degree of utilization to meet the initial demand before changes are needed to be made to the VNF architecture and code base.

Examples of Cloud Native VNFs Using Cloudify/ARIA

There are already a number of VNFs using Cloudify or ARIA for cloud enablement. These include Bind9, Metaswitch Clearwater, Fortigate, Brocade, Ruckus, and Athonet.

A good reference is the demo below, created by Metaswitch that illustrates how they were able to integrate their IMS, SBC and turn them into cloud native VNFs.

MetaSwitch Cloud Native VNF

Demystifying the Confusion Between Carrier-generic NFV Orchestration and VNF Orchestration

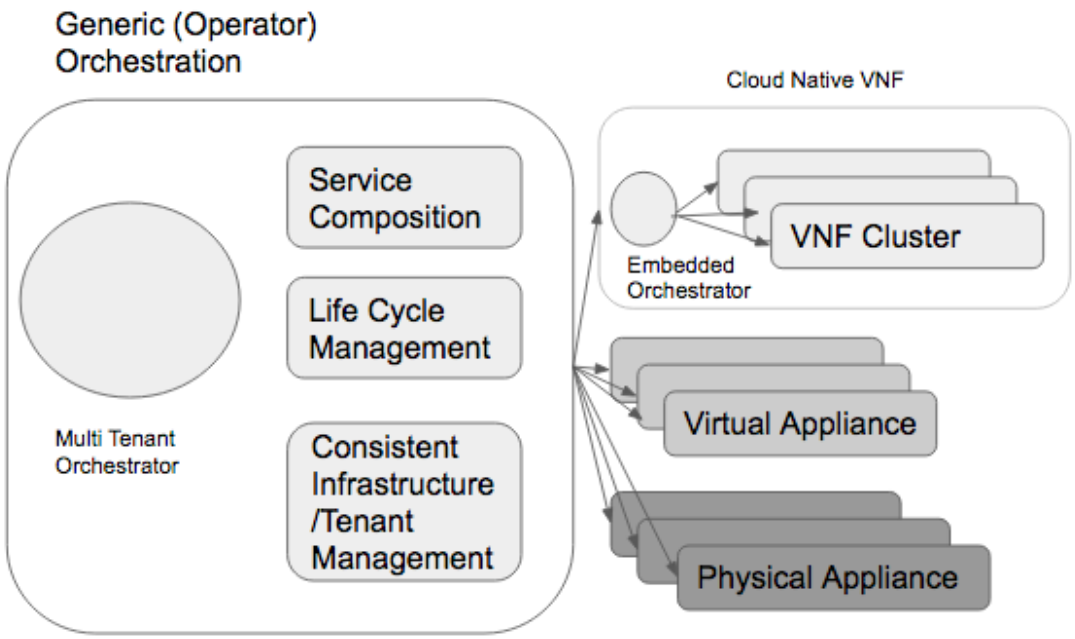
VNF providers can't dictate what the operator's orchestrator of choice should be, and therefore often need to support an ecosystem of NFV orchestration tools as well as integrate with them in the operator's data center.

Which begs the question – Does the use of an embedded orchestrator conflict with that of the operator's orchestration tool?

The experience with this new generation of Cloud Native VNFs, as mentioned above, is that quite often advanced VNFs are expected to take control over their own resources. This is because the complexity of exposing their intimate processes to the operator's orchestrator for handling methods such as scaling or remediation makes it rather unsustainable, and could be easily broken between new release cycles of the VNF service.

In this context, it is expected that the operator's orchestrator only interact with the northbound API exposed by the Cloud Native VNF, and not with each underlying resource provided by the VNF. The VNF is expected to take care of its internal resources provisioning and configuration, as well as self-healing and auto-scaling.

With that in mind, it becomes clear that an embedded VNF orchestrator doesn't conflict with the operator's orchestrator, it only makes the process of integrating the VNF into the operator's orchestrator simpler.



Operator vs VNF provider Orchestration

Interestingly enough this realization is now also part of the new ETSI spec – NFV MANO. (Page 12)

Final Notes

Taking your existing VNF and making it Cloud Native, or even building Cloud Native VNFs from scratch, is a complex project.

Cloud Native VNF providers that first entered this space had no other choice but to build their own proprietary orchestrator to meet these requirements. Since then new, generic open source orchestration tools have emerged and can provide new vendors a simpler and faster way to build similar capabilities with a significantly shorter time to market. Cloudify and ARIA were designed specifically to provide an embeddable and lightweight orchestrator to fill that void.

In addition to all that, the use of standards such as TOSCA over proprietary orchestration makes the integration, and the management of VNFs, with the carrier generic orchestrator smoother.

Where Do I Go From Here?

This NFV Lab on Demand provides an easy to use lab environment that allows you to run the Clearwater IMS, as well as a hands-on experience to learn how you can turn an existing VNF into a cloud native service in a few simple steps.

GIGASPACES

www.getcloudify.org

GigaSpaces Offices Worldwide

US East Coast Office, New York
Tel: +1-646-421-2830

US West Coast Office, San Jose
Tel: +1-408-816-1740

International Office, Tel Aviv
Tel: +972-9-952-6751

Europe Office, London
Tel: +44-207-117-0213

Asia Pacific Office, Hong Kong
Tel: +852-37198212