FABRIC Visioning White Paper
2-14-2020

Network Virtualization Architecture

We all hear of virtual machines, virtual circuits, virtual local area networks, virtual memory, virtual routing and forwarding, virtual private networks and the list goes on. Technopedia.com lists 130 technical terms prefaced as "virtual" things. Individually these different concepts have shown to be useful in their own ways, within defined scopes. But what is the common thread connecting them all? In general, it is difficult to identify a clear common denominator other than the fact that they are simply software constructs of functionalities that used to require dedicated hardware.

In this white paper we assert that *virtualization* is much more than simply a software option of past or existing, hardware centric functionality implementations. We assert that virtualization can deliver much more power and flexibility if it were formalized as the core concept of a complete cyber-infrastructure architecture, a design principle enabling the deployment of dynamic CI environments, or new and innovative services or applications. The transition from simply "virtualizing" individual functionalities to virtualization-centric architecture and design leads to significantly optimized, flexible cyber-infrastructures.

A Virtualization Architecture abstracts a function or service, it defines those functions or services in terms of their *behavior*, not in terms of their physical implementation. In this manner, the virtual object is no longer tied to or dependent upon specific physical hardware or technology but rather the other way – the infrastructure is chosen to produce the behavior defined for the service. The virtual object – or objects – can be manipulated in a *conceptual plane* to easily design advanced and innovative new services without being constrained by specific hardware (or lack of hardware). Even when the abstracted function or component may be modeled to mimic a piece of conventional hardware or physical infrastructure, a virtualized version remains technology agnostic and can therefore be more efficiently implemented or mapped into the infrastructure and managed in operations.

A well bounded object model constrains the behavior of virtual objects to prevent the activities in one virtual scope from interfering in the functioning of other virtual objects. This is the "Vegas Rule" – what happens in a virtual object, stays in that virtual object. Insulated and isolated objects enable deterministic and predictable performance – repeatable and secure performance. Predicted performance, or performance expectations, as compared to actual measured performance, is necessary to do analysis and evaluation of experiments.

The Vegas rule is not all we need. Some virtual objects *do* need to interact with other objects, or with existing conventional infrastructure or services. How do we model this interaction and still support deterministic performance? A Virtualization Architecture must address how virtual objects work together. How do virtual objects exchange information? How do we do this and preserve isolation and predictability? How do virtual objects interact in order to create a *combined* predictable functionality? What is the algebra that allows us to compose services with a deterministic and predictable performance and map these complex composite objects efficiently into the infrastructure?

The paper asserts that the "network", with the introduction of compute, storage, and user defined switching, is evolving into a general purpose infrastructure that simply specializes in data transport and switching. That is, a viable network virtualization architecture is also a more comprehensive cyber-infrastructure virtualization architecture. Indeed, we assert that even CI components such as sensors or instruments (e.g. radio telescopes) and non-traditional CI components such as licensed cellular spectrum can be virtualized and manipulated in the concept plane as easily and as effectively as other components.

Virtualization does not mean emulation or simulation. Virtualization is a process by which a defined user resource or service is mapped onto the infrastructure to produce the defined behavior. This may include emulation or simulation in some cases, but in most environments today the virtual object is mapped to hardware and runs natively on that hardware at line rate. Virtual machines are typically running natively on actual CPUs hardware. Virtual circuits run an full line rate. Virtual switching – even for advanced technologies such as OpenFlow- now have virtualization features implemented in the fast path so even “virtual” OpenFlow switches are able to run at native hardware performance (100Gbps) and can easily and quickly be mapped from one set of port assignments to another. We believe these features can be replicated in emerging switching technologies as well (e.g. P4 and FPGAs)

Envisioned features in FABRIC such as QoS and fine-grained control of VM mapping to CPUs and cores are capabilities that are prerequisites for isolation and insulation. The ability to reprogram microcode in NICs or to access pipelines in advanced P4 switches to implement virtualization support features in the fast path is also necessary. High performance storage together with high-speed network transport is necessary to support operational requirements such as virtual resource migration and grooming especially for complex composite virtual environments.

Making virtualization a central architectural principle is an enabler a larger scope concepts such as composite virtual objects and composable services which also need formalization. The development of these layers requires an appropriate facility where they can be deployed and incrementally scaled up.

The authors of this white paper have been focused on network virtualization and dynamic network services and application specific networks for many years. There is now working open source code that can demonstrate the foundations of virtualization described here. We would like to deploy this basic generic virtualization service (GVS) software across FABRIC as an open experiment, offering it as one possible service capability to interested FABRIC users and also continue its development and refinement of features and principles.

Jerry Sobieski
Chief Research Officer, NORDUNet; and George Mason University

Susanne Naegele-Jackson
Erlangen Regional Computing Center (RRZE), Friedrich-Alexander University of Erlangen-Nuremberg (FAU)

Ciprian Popoviciu
East Carolina University

Bijan Jabbari
George Mason University