CIMI Corporation

Trends in Cloud-Native Networking

A Special Report by Tom Nolle,
President of CIMI Corporation

If you ask a hundred enterprise CIOs what the most important trend in computing is, at least three-quarters of them will say "the cloud". If you ask that three-quarters what the most important cloud development trend is, at least three-quarters will say "cloud-native". I know, because I have asked, but just reading the trade rags will show even hardened skeptics that cloud-native is hot.

Networking is hot too. Roughly that same three-quarters of CIOs know that cloud computing requires some fundamental changes in their networking strategy, not the least because you can't put your own network hardware into public cloud infrastructure. How many CIOs know what cloud-native means to networking. Not three-quarters, or even half or one quarter. About ten percent even **think** they have the answer, and that's not acceptable…to CIOs themselves or to the networking or cloud industries.

Cloud-native refers to a model of building applications that focuses on realizing the full benefits of cloud computing, rather than just "migrating" legacy applications to the cloud. Those "full benefits" include component reuse, scalability under load, resiliency, ease of non-destructive redeployment after changes, agility in meeting new business requirements and opportunities, and management of development and operations costs. OK, all these would be good things to have, but the network implications of cloud-native could be significant, even crippling, if they're not addressed.

Let's start by looking at what "cloud-native" means in a technical sense. First, it's based on **container technology**. A container is a highly portable unit of application logic that carries with it all the instructions needed to install it on a resource, connect it, and parameterize it for operation. Containers are efficient and lightweight, and they're the future of application development.

Applications consisting of multiple containers require some form of "orchestration" for deployment, scaling, and replacement of elements, either because a new version is available or because the resource on which the container ran has failed. In the cloud-native world, **Kubernetes** has emerged as the de facto standard for orchestration, and thus the basis for cloud-native deployment.

Kubernetes is an orchestrator, a management tool. The leading edge of cloud-native technology is the third of our technical elements, the **service mesh**. A service mesh is a run-time framework in which containers that deploy can register their location so they can be found, scale with load balancing when workloads change, and communicate among themselves without regard for where everything has been put, or has moved. Without a service mesh, organizing application workflows in a cloud-native world would be difficult.

You can see that, first and foremost, cloud-native is **the cloud**. We already know, and have known from literally the first cloud computing services, that networking in the cloud is different. It has its own address space, its own addressing rules, its own management and connection practices. Every public cloud provider has responded to this with its own virtual networking model, because cloud-native is the ultimate in virtualization, and it demands the ultimate in virtual networking. That seems obvious, but it's not yet fully accepted.

Cloud-native takes the age-old "what it is versus where it is" debate in networking to a new level. Physical networks connect physical access points, and users and processes are addressed based on what physical places they happen to reside in. To make a "logical" entity addressable, you'd use a second-level of abstraction, like a URL, that decoded to an address. That way, changing the URL mapping could accommodate a movement or redeployment of a function. The problem in cloud-native is that

everything is virtual, everything is made up of a lot of separate, moving, scalable, parts, and so everything is constantly trying to figure out where everything else can be found.

All of this could be eliminated by shifting focus from "real" or "physical" networks to virtual networks. A virtual network connects logical elements not physical places. It can accommodate movement of users, rehosting of applications, and scaling of components. In short, it can do what everything in the cloud-native computing revolution is focused on creating and optimizing.

The problem for users is that all virtual networks aren't really as virtual as they need to be. We've had "virtual private networks" in the form of MPLS VPNs and virtual LANs for over a decade, but these have been little more than segments of "real" networks, still based on location rather than logical identity, and still rigidly tied to location so as to make scaling and redeployment complex. SD-WAN technology represents the first real step forward by the industry to free itself from the old paradigm.

Unfortunately, even most SD-WAN implementations fail the virtual-network test. Instead of assigning addresses to logical entities, they still assign them to physical resources, the places we've put those logical entities for the moment. There's obviously some key missing ingredient here, so what is it?

The answer is **session awareness**. Sessions are workflows between logical entities. Sure they have to be hosted or located somewhere at the moment, but with sessions you can identify the logical entity involved and preserve that knowledge when you route traffic. If you know who's in session, you know the logical entities and you can then track those entities if they move around. In addition, because we're now talking about logical entities that represent people and processes or databases, we also know what's important and not important, or even what's not a valid relationship at all.

Session awareness augments cloud virtual networking. In its best implementation by 128 Technology, cloud virtual networking is supported by adding an SD-WAN "proxy" or "sidecar" to a logical entity, meaning an application or component. This lets the cloud component participate fully in a true virtual network but remains compatible with the cloud's own virtual network, which it uses as a transport network.

This is exactly how service mesh technology works for cloud-native deployments. In other words, the sidecar approach is the accepted virtual network underpinning of the cloud-native community, a realistic path from legacy development practices where programmers have to build their own workflows and tools, to a future where a single framework describes how to build and then deploy and sustain a universe of cloud-native applications.

All of this starts with session awareness, though, because without that we can't take cloud-native concepts beyond the cloud and into the workplace. It won't end there, for the market or for 128 Technology. They're already exploring the potential of linking session-aware SD-WAN directly to Kubernetes virtual networking via the hook provided, and also integrating their cloud sidecar technology with service mesh sidecar implementations to optimize the tracking of scalable, movable, cloud-native assets. This is the sort of thing the industry, and in particular SD-WAN prospects, really need to see as they face their own, inevitable, transition to the cloud-native future.