Intercloud: Are You Moving Applications or Architectures?

Lori MacVittie, 2011-27-06

The former is easy. The latter? Not so much.

In the many, many – really, many – posts I've penned regarding cloud computing , and in particular the notion of Intercloud, I've struggled to come up with a way to simply articulate the problem inherent in current migratory and, for that matter, interoperability models. Recently I found the word I had long been groping for: **architecture**.

Efforts from various working groups, standards bodies and even individual vendors still remain focused on an application; a packaged up application with a sprinkling of meta-data designed to make a migration from data center A to data center B less fraught with potential disaster. But therein lies the continuing problem – it is focused on the application as a discrete entity, with very little consideration for the architecture that enables it, delivers it and supports it.

The underlying difficulty is not just that most providers simply don't offer the services necessary to replicate the infrastructure necessary, it's that there's an inherent reliance on network topology built into those services in the first place. That dependency makes it a non-trivial task to move even the simplest of architectures from one location to another, and introduces even more complexity when factoring in the dynamism inherent in cloud computing environments.

> "Virtualization will radically change how you secure and manage your computing environment," Gartner analyst Neil MacDonald said this week at the annual Gartner Security and Risk Management Summit. "**Workloads are more mobile, and more difficult to secure. It breaks the security policies tied to physical location. We need security policies independent of network topology.**" [emphasis added]
>
> -- Gartner: New security demands arising for virtualization, cloud computing, InfoWorld

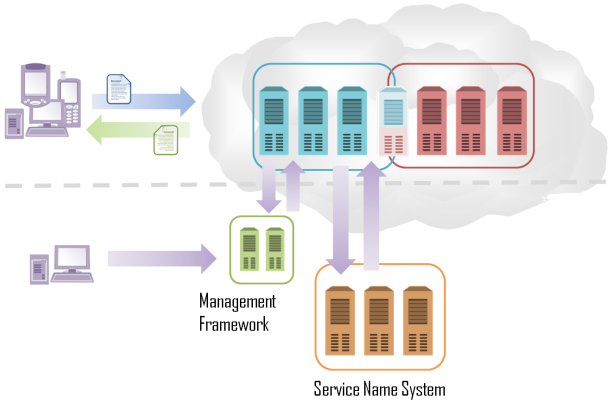
We could – and probably should – expand that statement to say, "We need application delivery policies independent of network topology", where "application delivery policies" include security but also encompass access management, load balancing, acceleration and optimization profiles. We desperately need to decouple architecture and services from network topology if we're to ever evolve to a truly dynamic and mobile data center – to realize IT as a Service.

JIT DEPLOYMENT

The best solution we have, now, is scripting. Scripting usually involves devops who use agile development technologies to create scripts that essentially reconfigure the entire environment "just in time" for actual deployment. This includes things like reconfiguring IP address dependencies. Once the environment is "up", scripts can be utilized to insert and update the appropriate policies that ultimately define the architecture's topology. Scripting performs some other environment-dependent settings, as well, but most important perhaps is getting those IP addresses re-linked such that traffic flows in the expected topology from one end to the other.

But in many ways this can be as frustrating as waiting for a neural network to converge as dependencies can actually inhibit an image from achieving full operational status. If you've ever booted a web server that relies upon an NFS or SMB mount on another machine for its file system, you know that if the server upon which the file system resides is not booted that it can cause excessive delays in boot time for the web server as well as rendering it inoperable – a.k.a. unavailable. That's a simple problem to fix – unlike some configurations and policies that rely heavily upon having the IP address of other interconnected systems.

These are well-known problems with not-so-best-practices solutions today. This complicates the process of moving an "application" from one location to another because you aren't moving just an application, you're moving an architecture.



Relying upon "the cloud" to provide the same infrastructure services is a gambling proposition. Even if the same services are available you still run the (very high) risk of a less-than-stellar migratory experience due to the differences in provisioning methods. The APIs used to provision ELB are not the same as those used to provision a load balancing service in another provider's environment, and certainly aren't the same as what you might be using in your own data center. That makes migration a several layer process, with logically moving images just the beginning of a long process that may take weeks or months to straighten out.

SNS (SERVICE NAME SYSTEM)

We really do need to break free from the IP-address chains that bind architectures today. The design of a stateless infrastructure is one way to achieve that, but certainly there are other means by which we can make this process a smoother one. DNS effectively provides that layer of abstraction for the network. One can query a domain name at any time and even if the domain moves from IP to IP, we are still able to find it. In fact we leverage that dynamism every day to provide services like Global Application Delivery in our quest for multi-site resilience. We rely upon that dynamism as the primary means by which disaster recovery processes actually work as expected in the event of a disaster. We need something similar to DNS for services; something that's universal and ubiquitous and allows configurations to reference services by name and not IP address.

We need a service registry, a **service name system** if you will, in which we can define for each environment a set of services available and the means by which they can be integrated. Rather than relying upon scripts to reconfigure components and services, a component would need only learn the location of the SNS service and from there could determine – based on service names – the location of dependent services and components without requiring additional reconfiguration or deployment of policies.

A more dynamic, service-oriented system that decouples IP from services would enable greater mobility not only across environments but within environments, enabling higher levels of resiliency in the event of inevitable failure. Purpose-built cloud services often already take this into consideration, but many of the infrastructure components – regardless of form-factor – do not. This means moving an architecture from the data center to a cloud computing environment is a nearly Herculean task today, requiring sacrifice of operationally critical services in exchange for cheaper compute and faster provisioning times.

If we're serious about moving toward IT as a Service – whether that leverages public or private cloud computing models – then we need to get serious about how to address the interdependencies inherent in enterprise infrastructure architecture that make such a goal more difficult to reach. Services will not empower migratory cloud computing behavior unless they are unchained from the network topology. That's true for security, and it's true for other application delivery concerns as well.

A core principle of a service-oriented anything is de –coupling interface from implementation. We need to apply that core principle to infrastructure architecture in order to move forward – and outward.

- It's Called Cloud Computing not Cheap Computing
- Operational Risk Comprises More Than Just Security
- Data Center Feng Shui: Reliability is not the Absence of Failure
- About that 'Unassailable Economic Argument' for Public Cloud Computing
- IT as a Service: A Stateless Infrastructure Architecture Model
- Challenging the Firewall Data Center Dogma
- Cloud-Tiered Architectural Models are Bad Except When They Aren't
- Cloud Chemistry 101
- You Can't Have IT as a Service Until IT Has Infrastructure as a Service

F5 Networks, Inc. | 401 Elliot Avenue West, Seattle, WA 98119 | 888-882-4447 | f5.com

F5 Networks, Inc.
Corporate Headquarters
info@f5.com

F5 Networks
Asia-Pacific
apacinfo@f5.com

F5 Networks Ltd.
Europe/Middle-East/Africa
emeainfo@f5.com

F5 Networks
Japan K.K.
f5j-info@f5.com