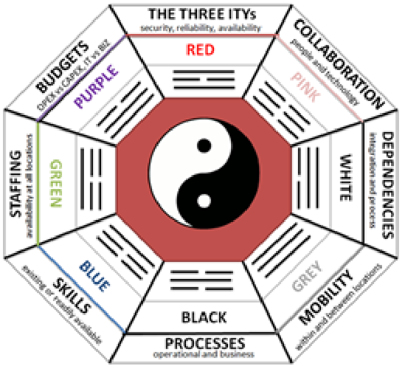
Data Center Feng Shui: Normalizing Phased Deployment with Virtualized Network Appliances

Lori MacVittie, 2010-16-08

Normalizing deployment environments from dev through production can eliminate issues earlier in the application lifecycle, speed time to market, and gives devops the means by which their emerging discipline can mature with less risk.

One of the big “trends” in cloud computing is to use a public cloud as an alternative environment for development and test. On the surface, this makes sense and is certainly a cost effective means of managing the highly variable environment that is development. But unless you can actually duplicate the production environment in a public cloud, the benefits might be offset by the challenges of moving through the rest of the application lifecycle.



NORMALIZATION LEADS to GREATER EFFICIENCIES

One of the reasons developers don’t have an exact duplicate of the production environment is cost. Configuration aside, the cost of the hardware and software duplication across a phased deployment environment is simply too high for most organizations. Thus, developers are essentially creating applications in a vacuum. This means as they move through the application deployment phases they are constantly barraged with new and, shall we say, interesting situations caused or exposed by differences in the network and application delivery network.

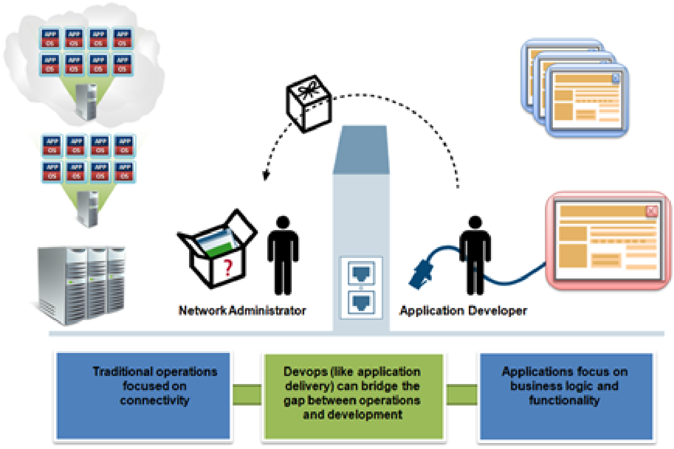
Example: One of the most common problems that occurs when moving an application into a scalable production environment revolves around persistence (stickiness). Developers, not having the benefit of testing their creation in a load balanced environment, may not be aware of the impact of a Load balancer on maintaining the session state of their application. A load balancer, unless specifically instructed to do so, does not care about session state. This is also true, in case you were thinking of avoiding this by going “public” cloud, in a public cloud. It’s strictly a configuration thing, but it’s a thing that is often overlooked. This causes problems when developers or customers start testing the application and discover it’s acting “wonky”. Depending on the configuration of the load balancer, this wonkiness (yes, that is a technical term, thank you very much) can manifest in myriad ways and it can take precious time to pinpoint the problem and implement the proper solution. The solution should be trivial (persistence/sticky sessions based on a session id that should be automatically generated and inserted into the HTTP headers by the application server platform) but may not be. In the event of the latter it may take time to find the right unique key upon which to persist sessions and in some few cases may require a return to development to modify the application appropriately.

This is all lost time and, because of the way in which IT works, lost money. It’s also possibly lost opportunity and mindshare if the application is part of an organization’s competitive advantage.

Now, assume that the developer had a mirror image of the production environment. S/He could be developing in the target environment from the start. These little production deployment “gotchas” that can creep up will be discovered early on as the application is being tested for accuracy of execution, and thus time lost to troubleshooting and testing in production is effectively offset by what is a more agile methodology.

DEVELOPING DEVOPS as a DISCIPLINE

Additionally developers can begin to experiment with other infrastructure services that may be available but were heretofore unknown (and therefore untrusted). If a developer can interact with infrastructure services in development, testing and playing with the services to determine which ones are beneficial and which ones may not, they can develop a more holistic approach to application delivery and control the way in which the network interacts with their application.


Network Administrator
Application Developer
Traditional operations focused on connectivity
Devops (like application delivery) can bridge the gap between operations and development
Applications focus on business logic and functionality

That's a boon for the operations and network teams, too, as they are usually unfamiliar with the application and must take time to learn its nuances and quirks and adjust/fine-tune the network and application delivery network to meet the needs of the application. If the developer has already performed these tasks, the only thing left for the ops and network teams is to implement and verify the configuration. If the two networks – production and virtual production – are in synch this should eliminate the additional time necessary and make the deployment phase of the application lifecycle less painful.

If not developers, ops, or network teams, then devops can certainly benefit from a "dev" environment themselves in which they can hone their skills and develop the emerging discipline that is devops. Devops requires integration and development of automation systems that include infrastructure which means devops will need the means to develop those systems, scripts, and applications used to integrate infrastructure into the operational management in production environments. Like developers, this is an iterative and ongoing process that probably shouldn't use production as an experimental environment. Thus, devops, too, will increasingly find a phased and normalized (commoditized) deployment approach a benefit to developing their libraries and skills.

This assumes the use of virtual network appliances (VNA) in the development environment. Unfortunately the vast majority of hardware-only solutions are not available as VNAs today which makes a perfect mirrored copy of production at this time unrealistic. But for those pieces of the infrastructure that ***are*** available as a VNA, it should be an option to deploy them as a copy of production as the means to enable developers to better understand the relationship between their application and the infrastructure required to deliver and secure it. Infrastructure services that most directly impact the application – load balancers, caches, application acceleration, and web application firewall – should be mirrored into development for use by developers as often as possible because it is most likely that they will be the cause of some production-level error or behavioral quirk that needs to be addressed.

The bad news is that if there are few VNAs with which to mirror the production environment there are even fewer that can be/are available in a public cloud environment. That means that the cost-savings associated with developing "in the cloud" may be offset by the continuation of a decades old practice which results in little more than a game of "throw the application over the network wall."

Related Posts

- All Data Center Feng Shui posts on DevCentral
- Sessions and Cookies and Persistence, oh my!
- Cloud Computing: Is Your Cloud Sticky? It Should Be

from tag hardware

- Cloud Lets You Throw More Hardware at the Problem Faster
- Don't Throw the Baby out with the Bath Water
- When Did Specialized Hardware Become a Dirty Word?
- IT Myths and Legends: No One Understands Our Legacy Software
- Virtual Network Infrastructure: Virtually Good Enough?
- 64 Things Every Geek Should Know - LaptopLogic.com
- Hardware Acceleration Critical Component for Cost-Conscious Data Centers

from tag application

- 3 Really good reasons you should use TCP multiplexing
- Packets, shmackets. What about the apps?
- The Unpossible Task of Eliminating Risk

- The third greatest (useful) hack in the history of the Web
- 4 things you can do in your code now to make it more scalable later

(more..)

del.icio.us Tags: MacVittie,F5,data center feng shui,virtualization,virtual network appliance,hardware,architecture,application,lifecycle,cloud computing,devops

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