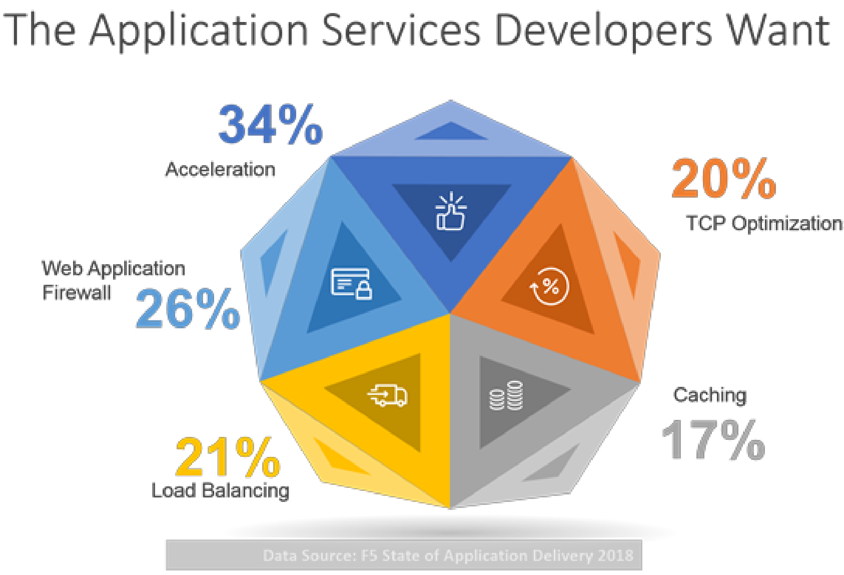
Why Infrastructure Matters to Developers using Containers

Lori MacVittie, 2018-04-09

Today, one of the most frequently heard questions in the halls of an infrastructure provider is how to explain the value of that infrastructure to the developer community. Trouble is, most of the benefits of infrastructure are garnered after delivery to production. None of these really provide value directly to the developer, at least not in a meaningful way that impacts their daily routine.

The extra-development benefits are something of which developers are certainly aware. In our 2018 State of Application Delivery we had a small percentage of developers represented. But that percentage spoke loudly on the topic of the application services they wanted to deploy. Some of those application services - load balancing, caching, and acceleration - are just as often deployed as part of the application itself as they are as infrastructure. TCP optimizations and WAF are almost always infrastructure services, deployed up stream (in the data path) in front of applications (whether monolith or microservices).

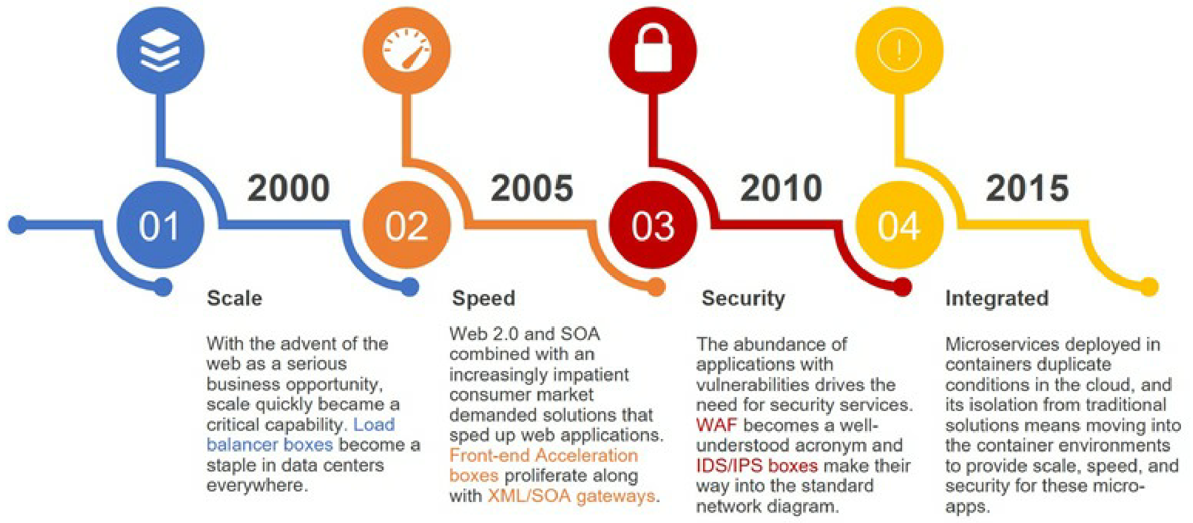


There’s value in all these application services. Reduction of risk, improved performance, scalability. But these are application and business benefits; they benefit developers post-delivery of the application to production. It’s hard to find - let alone articulate - benefits of infrastructure pre-delivery, i.e. as part of the development lifecycle.

But as we continue to embrace containers and microservices, the value in infrastructure pre- and post-delivery becomes more obvious.

Like most emerging technologies, the early days of a new application architecture bring with it very little infrastructure. It might be surprising for developers to learn that application architecture significantly shapes the application services infrastructure. From the move to three-tier, web-based apps came scalability (load balancing). From the adoption of Web 2.0 with its responsive presentation layer came front-end acceleration infrastructure. With the advent of mobile and increasing digitization of every industry we saw infrastructure react with security services like WAF, DDoS and bot defense.

The Impact of Apps on the Network



01 2000 **Scale**

With the advent of the web as a serious business opportunity, scale quickly became a critical capability. Load balancer boxes become a staple in data centers everywhere.

02 2005 **Speed**

Web 2.0 and SOA combined with an increasingly impatient consumer market demanded solutions that sped up web applications. Front-end Acceleration boxes proliferate along with XML/SOA gateways.

03 2010 **Security**

The abundance of applications with vulnerabilities drives the need for security services. WAF becomes a well-understood acronym and IDS/IPS boxes make their way into the standard network diagram.

04 2015 **Integrated**

Microservices deployed in containers duplicate conditions in the cloud, and its isolation from traditional solutions means moving into the container environments to provide scale, speed, and security for these micro-apps.

So what we see happening right now is developers are codifying infrastructure service capabilities in their applications. Developers are putting into code what has traditionally been the responsibility of upstream services. Retries from load balancers. mTLS from the platforms or proxies. Access control to restrict communication to legitimate clients.

These are infrastructure responsibilities that, because of the nascent nature of container frameworks and the rapid adoption rates, have been assumed by developers.

But as has always been the case, that's changing. Just as previous app architectures have driven responses in the network infrastructure, so are containers and microservices. Only this time the changes aren't coming in the form of a new box. What's happening now is the move to integrate the application services developers need into the container environment. That's where service mesh comes in and offers up real, quantifiable value directly for developers.

As explained by Andrew Jenkins, Lead Architect at Aspen Mesh in an interview with Linux.com, "It's remarkable how easy it is to start making a web service today. You can fit the code in a tweet. This isn't a real web service, though. To make it resilient and scalable you need to add some stuff to the data plane of the app. It needs to do TLS, and it needs to retry failures, and it needs to only accept requests from this service but not that one, and it needs to check the user's authentication, and so on. A service mesh can help you get that data plane functionality without having to add code to the app."

The value pre-delivery is in the reduction in scope and elimination of repetitive but necessary code to handle basic security and scale within a container environment. A service mesh has a wide variety of cool capabilities, but its benefits remain largely operational - observability, accountability, scale. The most significant value to a developer is in the elimination of code (and thus reduction of technical and architectural debt).

> Employing a service mesh to scale, secure, and observe apps deployed in container environments relieves the burden of writing code that should be handled by the infrastructure - but until recently has been delegated to developers. A service mesh is one way to relieve the responsibility on developers and give them back precious time they could be using to develop services and apps that in turn deliver value to the business.

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