Analyzing Our Docker Compose Infrastructure Requirements

Posted At : April 27, 2018 5:18 PM | Posted By : Cutter
Related Categories: Development, Application Setup, Docker

> This multi-part series goes in depth in converting this site infrastructure to a containerized setup with Docker Compose. See the bottom of this post for other posts in the series.

So, before we continue I think it's important to layout some of the next steps in what it is I wanted/needed to accomplish. I'm using **Docker Compose** to define my infrastructure. I started with the database, as that will be used for multiple sites, so that was a no brainer.

But what's next? Well, first let me look at some of my requirements.

- Database (check)
- ColdFusion Rendering Engine (for this blog) [**Lucee**]
- Multi Context CF Setup (blog and os project sites/pages)
- Web Server
- New Photography Site (?)
- Secure Sites with SSL for Google
- Auto Backup to S3 (?)

Yeah, I set some stretch goals in there too. But, it's what I wanted, so I got to work.

In my initial implementation on **Digital Ocean** I used the default **lucee4-nginx** container. **Nginx** is a nice, easily configurable web server. And, it worked great for a year, up until Digital Ocean restarted my Droplet while running some necessary security maintenance on their infrastructure. Suddenly, nothing worked.

Whoops.

OK, so this was the first thing I had to figure out. Turned out to be a relatively easy thing to track down. I was using the "latest" container. Lucee updated the lucee4-nginx container version of Tomcat. There were changes to the container's internal pathing that no longer jived with the various settings files I had, so I just had to resolve the pathing issues to get it all straight. I also took the opportunity to go ahead and switch to Lucee 5.2.

Now I was back up and running on my initial configuration, but (as you can see in the list above) I had some new goals I wanted to accomplish. So I sat down and started

looking over my other requirements to figure out exactly what I needed. One of the first things I looked into was the SSL certs. I could buy expensive wildcard domain certs, but this is a blog. It creates no direct income. Luckily there's **LetsEncrypt**. LetsEncrypt is a great little project working to secure the internet, creating a free, automated and open Certificate Authority to distribute, configure and manage SSL certs.

Long story short, my investigation of all of my requirements made me realize that I needed to decouple Lucee from Nginx, putting each in it's own separate container. I'm going to use Nginx as a reverse proxy to multiple containers/services, so decoupling makes the most sense. I'm still keeping things small, because this is all out of pocket, but one of the advantages of Docker Compose is I **can** define multiple small containers, each handling it's own defined responsibility. In the end it comes down to this.

In the end our containers will look something like this:

- MariaDb (check)
- Lucee 5.2 (3 sites)
- Other (photo site, possibly **Ghost**)
- Nginx
- Docker-Gen (template generator, dependency for...)
- LetsEncrypt
- Backup (TBD)

Everyone's configuration changes over time, and this is what I came up with after my latest analysis of my requirements. I've already gone through multiple rounds of attacking each different requirement, and probably haven't finalized yet, but next post we'll step in again and setup our Nginx container and start some configuration.