TIER Campus Success Program Case Study:University of Illinois Urbana-Champaign

Exectuive summary

Integrating the available TIER packaging with our new AWS infrastructure and DevOps pipeline as part of a cloud-first initiative helped us deploy a new instance of Grouper as well as a migration of Shibboleth.

TIER Feature Supported

Shibboleth and Grouper TIER Docker Containers

Collaborators

- Erik Coleman
- Keith Wessel (Illinois IAM team)
- Liam Hoekenga (Michigan)
- Chris Hyzer
- Shilen Patel (Grouper)
- Scott Cantor (Shibboleth)
- Chris Hubing
- Paul Caskey (Internet2)

Community resources:

CSP Grouper Deployment Enhancement Working Group, grouper-users mailing list, users@shibboleth.net mailing list. Grouper Wiki, Grouper Deployment Guide, Grouper Training,

The Project

Our Cloud First Initiative aims to migrate many campus IT services to the cloud, initially using the AWS platform, and following a newly-developed continuous integration/continuous development (CI/CD) pipeline developed internally to shift the paradigm from ad-hoc server deployment to one of “infrastructure as code” with a DevOps mentality. The primary technology being used is Amazon's Elastic Container Service and Fargate, which is Amazon's version of Docker, making the TIER-packages perfect for our needs.

The Problem

Our Shibboleth IdP is currently running in-house and was identified as one of our first services to move to Amazon AWS as part of our cloud-first initiative. The plan is to replace our in-house IdP nodes with additional cloud nodes. Dockerized versions of the IdP existed before TIER, but they require a significant amount of tweaking to run in the Fargate environment.

We also have a need to deploy our first production instance of Grouper, branded as Authorization Manager. Since it is a brand-new installation, we plan to deploy this to the cloud from the start and hope to have it in production later this fall. As with Shibboleth, there were Dockerized versions of the Grouper components before TIER, but they aren’t drop-in ready for Fargate.

The Solution

The solution is based largely on leveraging the TIER packages. We used a layered approach, taking the base TIER images and layering additional changes on top of the image, creating our own “custom base” image. These extra layers help reconfigure Apache, Tomcat and Shibboleth to not use SSL (now terminated on the AWS Application Load Balancer) and add Web UI customizations. From there, we modified the configuration files as appropriate for our application, and pushed the images into Amazon Elastic Container Registry (ECR).

To launch the images, we needed to build the necessary AWS infrastructure. Our Cloud First Team crafted reusable Terraform code that could very easily deploy all the necessary infrastructure components like Elastic Container Service Tasks and Clusters, Application Load Balancers, Security Groups, Roles, even DNS entries-- all in pre-built Virtual Private Clouds (VPCs), one for test/staging and one for production. As owners of the IAM services, the only customizing needed for Terraform was to specify the parameters and quantities of the AWS objects. Most of the code was reusable between both the Grouper and Shibboleth deployments.

To automate the regular updating of images and infrastructure, we are leveraging Drone (https://drone.io) to create the continuous integration and continuous development (CI/CD) pipeline. With our AWS authentication behind Shibboleth, we created a new AWS CLI plugin that leverages the Shibboleth ECP for command-line Shib login, complete with support for Two-Factor Authentication (2FA). A diagram of our CI/CD pipeline is below:

blocked URL

The Result

The end-result was a fully cloud-hosted implementation of Grouper and Shibboleth in AWS. For Grouper, at least one container was deployed for each role (UI, web service, daemon/loader), all talking to a back-end RDS (MariaDB) instance for the Grouper database. The use of Application Load Balancers (ALB) allowed us to terminate SSL at the ALB, as well as permit the expansion of the server roles through the use of auto-scaling, achieved by changing the container quantity in the Terraform code.

In the case of Shibboleth, the cloud-hosted IDP nodes were integrated into the existing on-premise IDP infrastructure as a sort of "hybrid" configuration, splitting the load between on-premise VMs and cloud containers through the use of a Route 53-hosted Global Server Load Balancer (GSLB). Eventually, the on-premise instances can be phased out, to move toward a fully 100% cloud-hosted model for the IDP. We also plan to divide up IdP nodes among different AWS regions for further stability. Finally, we plan to autoscale the IdP cluster size based on load, reducing cost associated with idle CPUs when usage is low.

For both services, we needed to incorporate the service logs into our central SIEM, which uses Splunk. We built a Lambda function to collect CloudWatch logs and push those into a Splunk HTTP Event Collector (HEC).

Lessons Learned

During the course of this initiative, we've been learning the TIER packaging methodologies while also learning about Docker and AWS infrastructure. We've learned a number of valuable lessons that we can share to make this type of project easier for others:

- Pick an environment, and commit: the ever-changing AWS features make goals a moving target. We originally targeted using Elastic Beanstalk, but ECS Fargate came along and we rebuilt our pipeline to use it instead, which cost us time.
- The learning curve for cloud infrastructure as code-- Terraform and Dockerfiles is significant. Find a friend and some good on-line resources.
- Build a hierarchy of reuseable Terraform modules. You'll then have a set of tinker toys that will greatly speed up deployments of new services as well as changes to underlying components.
- When moving to containerized services, step back and reevaluate everything, striving for one process per container. Those cron jobs that you've been running for the past ten years might be better accomplished with new or interesting external functionality, for instance.
- Consider all of the pieces of infrastructure you have available before you architect a solution, for instance AWS Secret Manager for data sealer key rotation, instead of Cron running inside an image.
- Optimize your Grouper database regularly. You'll save yourself a lot of waiting and frustration.
- When moving to cloud, consider how administrative console access is done and make a game plan (i.e., Grouper Shell). Similar to mentioned above, when making a shift like this, you need to step back and re-evaluate your current practices. For us, it was how to handle lack of console access in AWS, but any new architecture can present different but similar challenges.

Resources

- Amazon Web Services ECS Fargate
- Terraform by HashiCorp, Inc.
- Python Module for AWS CLI login using Shibboleth ECP
- TIER Grouper Docker Images
- TIER Shibboleth IDP Docker Images

Conclusions

TIER has worked to package common software components in standards-based packaging, rolling in preconfigured best practices. This effort has been a huge help to our Cloud First initiative, speeding up our work from months to weeks. In addition, the community collaboration of the Campus Success Program has allowed participants to synergize, teaching each other and group brainstorming solutions. We've helped others solve problems that we've already solved for ourselves, and they've done the same for us. Illinois is very close to accomplishing our goals. In the process, we've been able to produce components that other institutions can reuse to speed up their own cloud deployments.

About The University of Illinois at Urbana-Champaign

The University of Illinois at Urbana-Champaign is one of the original 37 public land-grant institutions created after President Abraham Lincoln signed the Morrill Act in 1862. As of fall, 2016, we had an undergrad enrollment of 33,467 students from all 50 states and an additional 5,537 international undergraduate students from 82 countries. We also had 11,413 students in our graduate and professional programs. We have approximately 2,700 faculty members and 8,000 staff members. Our students can earn degrees in more than 150 academic programs in our fifteen colleges, including professional colleges for law and veterinary medicine.

Our library is one of the largest public university libraries in the world with a collection of more than 24 million items. We have twenty Area studies libraries, including one of the largest engineering libraries in the country, a state-of-the-art agricultural library, and a world-renowned rare book and manuscript library.

The University of Illinois at Urbana-Champaign is a major research institution. In fiscal year 2016, we had research and development expenditures of $620 million. We are the top recipient among all universities in National Science Foundation award funding for six years running.

Such an extensive community of learning and research leads to increased needs for federation, international multi-institution collaboration, and scalable authorization mechanisms. In addition, in our current tight economic climate, we must do all of this efficiently.

(Source: Facts about University of Illinois at Urbana-Champaign, http://illinois.edu/about/facts.html)