Neustar, Inc. is an American technology company that provides real-time information and analytics for the Internet, telecommunications, entertainment, and marketing industries; their portfolio also includes implementing clearinghouse and directory services to the global communications and Internet industries.
The Challenge

Neustar’s UltraDNS team was experiencing problems releasing new versions of their software on schedule and meeting their reliability Service Level Agreements (SLA). The difficulties Neustar encountered were related to the combination of having multiple target environments and a lack of automation. The IT leadership team commissioned a project to implement a DevOps pipeline that automated testing and Amazon Web Service (AWS) cloud-based code deployments. The team was asked to complete the project in 3 months and to approach 100% automation. The expected results included improved code quality and release timeliness.

The Solution

- Idexcel’s team created a deployment pipeline utilizing Amazon Machine Images (AMI), Jenkins and Terraform.
- Configured GIT webhooks to trigger application specific Jenkins job whenever the source code is checked in to GIT; this compiles the code, runs the static code analysis, generates the code coverage report, produces the documentation, and uploads the build artifacts to Amazon S3. If the Jenkins build succeeds, it then triggers an application infrastructure Jenkins job.
- Application infrastructure Jenkins job creates the live and canary load balancers in a configured AWS account for a specified version. If the load balancers already exist in the AWS account for a specified version, it skips creating ELBs. If this Jenkins build succeeds, it then triggers Packer pipeline Jenkins job.
- The Packer pipeline job creates an application-specific Amazon Machine Image (AMI) from the UltraDNS base AMI using Packer to run Puppet. If the AMI is created successfully, it then triggers the Server Group Pipeline Jenkins job.
- The Server Group Pipeline Jenkins job establishes the auto scaling group and attaches it to canary ELB; it then runs AWS Lambda sanity tests against the canary ELB. If the sanity tests pass it then appends the auto scaling group to live load balancer.
- The new software is live at this point; it runs the regression Jenkins job against the live load balancer. If the regression fails, it then attaches the old auto scaling group to the live load balance to rollback the install. If the regression is successful, it triggers another Jenkins build to clean-up the old AWS stack.
- Idexcel has also created an Amazon Cloudwatch dashboard to monitor the health of the application. If there are any hardware or software issues, it triggers Amazon Simple Notification Service (SNS), which conveys a notification to the customer support team and on-call resources.

The Benefits

Idexcel was able to complete the project in three months. Neustar and their UltraDNS team now deploy their software to production on a daily basis using this pipeline without worrying about questionable software quality, application downtime, or deployment errors. The UltraDNS development and operations team was able to utilize the code pipeline to reduce the number of defects by 30% and increase the cadence of their release schedule by more than 40% per month.