Linux Academy

Live! Lab

CodePipeline Basics

Contents

Lab Connection Information

- Labs may take up to five minutes to build
- Access to an AWS Console is provided on the Live! Lab page, along with your login credentials
- Ensure you are using the N. Virginia region
- Labs will automatically end once the alloted amount of time finishes

Related Courses

Manage and Deploy Code with AWS Developer Tools

Related Videos

What is CodePipeline?

Setup and Configuration

Creating a Simple Pipeline

Need Help?

Linux Academy Community

... and you can always send in a support ticket on our website to talk to an instructor!

CodePipeline Basics builds upon the preview CodeCommit and CodeDeploy labs, taking our previous results and configuring a simple two-state pipeline for release. With the basics down, we then expand our pipeline to include approval and production release stages.

Simple Two-Stage Pipeline

Navigate to the CodePipeline Dashboard, and press **Get started**. Provide a name for the pipeline. We named ours *LiveLabPipeline*. **Next step**.

For **Source Provider**, choose *AWS CodeCommit*, selecting the *LiveLabRepo* "created" in the previous lab, and the *master* branch. **Next step**.

We are not using a **Build provider**. Select *No Build*. **Next step**.

Our **Deployment provider** is *AWS CodeDeploy*, with an **Application name** of *LiveLabApplication* and **Deployment group** of *LiveLabDeploymentGroup*. **Next step**.

Next is **Service Role**. The **Role name** contains your Linux Academy username and the phrase *CodePiplineServiceRole*. **Next step**.

Review the information; then press **Create pipeline**. After the initial pipeline creation, the code automatically runs for the first time. This may take a few minutes to finish. Once finished, our files are installed on the EC2 instance within our code deployment group.

Navigate to the **EC2 Dashboard** and retrieve the IP address for the **LiveLabCodeDeployInstance**. SSH into the machine using the username *linuxacademy* and password *123456*:

```
user@workstation~ :: ssh linuxacademy@52.90.29.31
```

Based on the appspec file conditions we previously defined, our files are installed in the *livelab* directory of the linuxacademy user's home directory. Once confirmed, we now know our initial CodePipeline run has been successful. We can now progress to a more complicated, four-stage pipeline.

Create Resources for the Four-Stage Pipeline

Before we can create a more complicated four-stage pipeline with a production server and "allow" action, we need to create the SNS topic, new deployment group and production server.

Deploy a New Instance

Because our *LiveLabCodeDeployInstance* is already set up to work with CodeDeploy, we can make an Amazon Machine Image of this instance. This is possible from the **EC2 Dashboard**. Select the

LiveLabCodeDeploy instance, and under the **Actions** drop-down, go to **Images**, **Create Image**.

For the **Image name** we used *CodeDeployConfigured*, with a **description** of *configured for CodeDeploy*. The rest can be left with default settings. **Create Image**. Navigate to the **AMIs** page of the EC2 Dashboard to view image creation. It may take a few minutes to finish.

We now want to **Launch** this new AMI. Select *t2.micro* for the instance type. **Next: Configure Instance Details**.

We want to ensure that the instance has a public IP, so select *Enable* for **Auto-Assign Public IP**. For **IAM Role**, choose the same *CodeDeployInstance* role used earlier in the lab. **Next: Add Storage**, which can be left with default settings. **Next: Tag Instance**.

For the **Name** tag, we provided the value *Production*. **Next: Configure Security Group**. This, too, can be left with the default settings. **Review and Launch**. **Launch**.

We do need to **Create a new key pair** for this server. We called ours *newkeypair*. **Download Key Pair**, then **Launch Instances**. As before, this may take a few minutes to complete.

Create New Deployment Group

Because our current deployment group is working with our *LiveLabDeployInstance* -- which is working as our testing instance, we need to create an additional deployment group to work with our newly-created production server. Navigate to the **CodeDeploy** console. Select our **LiveLabApplication** and click *Create deployment group*. For **Deployment Group Name**, we choose *ProductionDeploymentGroup*, with an *Amazon EC2* tag type, a **Key** of *Name* and a **Value** of *Production*. Use the *CodeDeploy* **Service Role ARN** that contains your Linux Academy username. **Create Deployment Group**.

Create an SNS Topic

The SNS topic is used notify administrators that a build has been pushed and there are changes and tests waiting for review. Open the **SNS Dashboard**. Press **Get Started**, then **Create a Topic**. Input a **Topic Name**; we used *LiveLabPipelineApproval*. **Create Topic**.

Select **Create subscription** and set the **Protocol** to *email*. Put in any email you like, but ensure it is one you can check. **Create subscriotion**. Log in to your email address to confirm the subscription.

Four-Stage Pipeline

With our required services set up and properly provisioned, we can now go about creating the four-stage pipeline. From the **CodePipline Dashboard** return to the previous **LiveLabPipeline** created earlier. We are going to build upon this pipeline.

Press **Edit** and navigate to the options under the *Beta* section, press **+ Stage**. Give it the name *Approval*, and

from the **Action** menu select *Approval*. Set the **Name** to *TestApproval* and the **Approval type** to *Manual Approval*. From here, we need to select the **SNS topic ARN** from the SNS alert configured in previous steps. **Add Action**.

Now, add a final stage to work as the production environment. Press the **+ Stage** button below the Approval action. We named ours *ProductionDeployment* **Set the** Action** to *Deploy*, then give it the **Production name** *ProductionDeploy*. The **Development provider** is *AWS CodeDeploy*, with the **Application name** being *LiveLabApplication*. The **DeploymentGroup** is the newly-created *ProductionDeploymentGroup*, and the **Input artifact #1** is the only available option, *MyApp*. **Add Action**. **Save Pipeline Changes**.

To test our new settings, press **Release change** to trigger another run through the pipeline. We know the first changes run successfully, but when the pipeline reaches the *Approval* stage, an email is sent out to the email defined in the SNS steps, issuing an alert about the pending release.

During this period, generally, the application is reviewed by QA or beta testers, or by other departments. Since we are working from a lab, however, select **Review** and add an arbitrary description, then press **Approve**. Once more, a few minutes are needed before the deploy to *Production* finishes.

You can verify the files are on the new server by logging into the server using the key pair created in the previous section. Your four-stage pipeline is now complete!