SUSE Blog published: Protect AWS CodePipeline CI/CD workflows with SUSE Security

Date:	2024-12-17
modified:	2024-12-17
tags:	SUSE, Blog, AWS, CodePipeline, Security, NeuVector
description:	I published a new SUSE Blog: Protect AWS CodePipeline CI/CD workflows with SUSE Security
category:	Code
slug:	suse-blog-published-protect-aws-codepipeline-ci-cd-workflows-with-suse-security-neuvector
Author:	Dominik Wombacher
lang:	en
transid:	suse-blog-published-protect-aws-codepipeline-ci-cd-workflows-with-suse-security-neuvector
Status:	published

This year I started as guest writer for the SUSE Blog. Today my latest Article was published: Protect AWS CodePipeline CI/CD workflows with SUSE Security (NeuVector) (Archive: [1], [2]).

I want to raise awareness about the importance of security inside your pipeline. The *classic* CI/CD Workflow is more focused on building code, performing unit- and / or integration tests and promoting the resulting artifacts into production. In general, there is nothing wrong with this approach. But there is a crucial part missing in: Security comes into the picture when your code already runs in production.

I also covered this topic in my SUSECON 24 session TUTORIAL 1156 NeuVector Integration into AWS CodePipeline CI CD Workflow, abstract:

> Learn in this session how NeuVector can help to secure your CI/CD Workflow on AWS CodePipeline. We demonstrate, how you ensure that only container images without a detected vulnerability move to the next stage in your Pipeline. And how you automatically block the deployment into production if an unexpected behavior at run-time was detected in your testing stage. We close the session with a brief overview of the NeuVector Prime listing on AWS Marketplace.

And my openSUSE Conference 2024 session NeuVector Integration into AWS CodePipeline CI/CD workflow, abstract:

> NeuVector is a open source container security platform. Key strengths are vulnerability and runtime scanning. I demonstrate in this talk how you ensure that only container images without a detected vulnerability move to the next stage in your Pipeline. How you define the baseline of allowed activities of your application. And how you can block the deployment into production if an unexpected behavior at runtime was detected in your testing stage. I'll use AWS CodePipeline, AWS CodeDeploy and AWS CloudFormation. The procedure is applicable to other toolset and Hybrid environments as well.

The workflow I explain and services I use in the Blog and the SUSECON and openSUSE Conference session are AWS focused. But the sample solution is a blueprint that you can adapt based on your own needs and use-case. The commands in the Buildspec file and the scan script are not tightly coupled to AWS CodePipeline. Feel free to use them as a starting point for other CI/CD systems as well.

When you opt for AWS CodePipeline and AWS CodeBuild, I suggest you take a look at Digest/RepoDigest value Workaround. There is a certain behavior in SUSE Security (NeuVector) that requires additional steps when an image is build locally inside AWS CodeBuild via docker.