Network Management and Automation

Lab 9

CI/CD with Jenkins

University of Colorado Boulder

Network Engineering Program

Professor Levi Perigo, Ph.D.

# Summary

As the network scale increases, so does the complexity of the code that manages, monitors, and configures the network. The CI/CD tools make it easier to continuously integrate and deploy Infrastructure-as-Code (IaC) in a cost-efficient manner, with higher product quality, and shorter release cycles. As the name suggests, one of the benefits of CI/CD tools is the Continuous Deployment of networks. The purpose of this lab is to understand the DevOps cycle by learning Jenkins, integrating it with GitHub, and configuring Jenkins Pipeline to continuously deploy networks via code. The other purpose of this lab is to use unit tests to make sure your code is meeting the pre-defined requirements, which in turn, indicates your code is ready for production. The objectives of this lab are to be used as guidelines, and additional exploration by the student is strongly encouraged.

# Objectives

1. Learn about Jenkins functionalities
2. Learn how to use version control (GitHub)
3. Learn how Jenkins and GitHub work together
4. Learn how to write Jenkins Pipeline
5. Learn how to write a clean and readable code that follows the PEP8 style
6. Learn how to write Unit Tests
7. Learn how to run Jenkins jobs (both push and pull methods)

# Objective 1 – Getting started

1. Install Jenkins on the NetMan VM or on your local system. Just make sure you have connectivity to GNS3 devices. Provide relevant screenshots of Jenkins status and Jenkins console. **[10 points]**
2. Provide Jenkins super-user permission to avoid using ‘sudo’ for the upcoming objectives. Provide a screenshot. **[5 points]**
3. Create a GNS3 topology provided for objective 2 in the previous Netconf lab. Configure Management IP addresses and NETCONF over SSH on all routers for the upcoming objectives. Paste relevant screenshots **[10 points]**
4. You will be using the script ‘netman\_netconf\_obj2.py’ and ‘info.csv’ provided to you (they are present in this directory) for the upcoming objective 2.2. (This script achieves objectives mentioned in objective 2 of the previous Netconf lab)

# Objective 2 – Creating Jenkinsfile for the pipeline

We will use Jenkinsfile to build our pipeline. The requirements of Jenkinsfile are:

1. It should trigger the Jenkins jobs whenever we push code to the repository. **[10 points]**
2. It should have 4 stages (You can add additional stages without compromising the requirements):
3. **Stage 1: Update/Install packages in the NetMan VM/laptop** (These libraries are used in the script ‘netman\_netconf\_obj2.py’)

Install the below libraries if they are not present already.

nccclient, pandas, ipaddress, netaddr, prettytable

**[10 points]**

1. **Stage 2: Checking and fixing violations** (Checking if application code complies with PEP8 code style)

It should report the check-style violations using [pylint](https://www.pylint.org/) for the script ‘netman\_netconf\_obj2.py’ and in case of violations, the pipeline should fail. Set the quality gate to 5. On failure, violations should be fixed before proceeding further. **[25 points]**

1. **Stage 3: Running the application** (Executing the application code to be tested in the next objective for functionality)

It should run the application code ‘netman\_netconf\_obj2.py’. **[5 points]**

1. **Stage 4: Unit test** (Testing if the application is meeting the requirements)

Write unit tests using Python’s [unittest framework](https://docs.python.org/3/library/unittest.html) to test the following scenarios:

1. If the IP address of loopback 99 on router 3 is 10.1.3.1/24
2. If R1 is configured only for a single area
3. If a ping from router 2’s loopback to router 5’s loopback is successful

**[30 points]**

1. The post action should send an email from Jenkin’s admin to you informing the build success/failure. This is explained in detail in objective 4 (below) **[5 points]**
2. Create a private repository in GitHub with the Jenkinsfile, the Unit test script, and the provided script along with the CSV file. Provide a relevant screenshot. **[5 points]**

# Objective 3 – Adding GitHub project to the pipeline and configuring GitHub webhook

1. Install GitHub plugin in Jenkins. Provide a screenshot showing that it is installed. **[3 points]**
2. Install Git in the NetMan VM/laptop if it is not installed already and set the path to executable in ‘Manage Jenkins’-> ‘Global Tool Configuration’->’Git’. Provide a screenshot. **[5 points]**
3. Install ‘Warnings Next Generation’ plugin in Jenkins. Provide a screenshot showing that it is installed. **[2 points]**
4. Install pylint in NetMan VM/laptop using pip3 and set the path to executable in ‘Manage Jenkins’ -> ‘Configure System’ -> ‘Warnings Next Generation Plugin Global Settings’. Provide a screenshot. **[5 points]**
5. Create a pipeline project from Jenkins Dashboard. Provide a screenshot. **[5 points]**
6. Go to your pipeline project. Under ‘General’ tab, enter your project URL.
7. In ‘Build Triggers’ tab, enable ‘Github hook trigger for GITScm polling’.
8. In ‘Pipeline’ tab:
   1. Select ‘Definition’ as Pipeline script from SCM
   2. Select ‘SCM’ as Git
   3. Provide repository URL along with a method of credentials (of your choice).
   4. Enter the branch name in ‘Branches to Build’

Save the changes and provide relevant screenshots. **[10 points]**

1. Configure GitHub Webhook:

GitHub integration with Jenkins needs a public URL for the webhook. Use [ngrok](https://ngrok.com/download) to create a third-party tunnel for the same.

Go to your GitHub repository settings and in ‘webhook’ tab, provide:

1. Newly generated URL in ‘Payload URL’ field
2. ‘Content Type’ as application/x-www-form-urlencoded.
3. Enable it for ‘push’ event.

Provide a screenshot. **[10 points]**

**Note:** Do not use this URL to access Jenkins (it does not work properly for this purpose), use this just to integrate GitHub with Jenkins

# Objective 4 – Configure E-mail notifications

1. Go to manage Jenkins -> Configure system -> Configure E-mail notification and extended E-mail notifications. Provide a screenshot of configurations **[10 points]**
2. Hit ‘Test configuration by sending test e-mail’ in E-mail notifications and provide screenshot of receiving an email**. [2 points]**

# Objective 5 – Running Jenkins Job

1. Go to Jenkins and hit ‘build now’. GitHub webhook needs one build to be run manually first to trigger the builds later automatically. Provide a screenshot of console output and a SUCCESSFUL build. If your build fails, work to fix it. **[10 points]**
2. Now, make a small change in any file. Commit it to GitHub. The job should start automatically. Show the console output of your next Jenkins job succeeding. The console output should show ‘started by GitHub push…’ **[10 points]**
3. Provide a screenshot of receiving an email about a SUCCESSFUL build. **[10 points]**

# Objective 6 – Schedule Nightly Jobs

Configure Jenkins to run GitHub jobs every night.

1. Provide a relevant screenshot. **[10 points]**
2. Briefly explain the significance of this. **[5 points]**

# Report Questions [15 points]

1. Do you think Jenkins is actively used in the industry? Why?
2. What is another CI/CD tool you would like to learn and why?
3. Where else could you run your Jenkins jobs?
4. Explain the use of Unit Testing in Jenkins.
5. What other tests can be carried out before an application deployment?

# Total Points \_\_\_\_\_\_\_ / 226