8.4 Assignment: 5.5 Assignment 4: Deploy a continuous integration pipeline using Gitlab, Jenkins, PHPUnit, Docker

Hide Assignment Information

Instructions

Estimated Effort Required: 10 hours

Learning Outcome: Students will understand how to build a modern DevOps pipeline and include automated application security checks as a part of the build process.

Assignment Goal: You will deploy CI infrastructure capable of automating a web application build process including automated security checking.

Preface: The following words should be interpreted as per RFC 2119 for this assignment: must, must not, required, shall, shall not, should, should not, recommended, may, and optional.

Environment: You have been provided with the IP addresses an AWS virtual machine to use to complete this lab. In past semesters, the VMs in this lab were only accessible either from RIT's physical campus or from RIT's VPN. You can find out how to connect to RIT's VPN here: https://www.rit.edu/its/services/network-communication/vpn. **However, due to the additional demand on RIT's VPN because of the campus moving to remote work, I have made these virtual machines public facing. Please closely monitor your virtual machines for suspicious activity.**

You have been emailed a .pem file in an encrypted .zip file. This SSH key can be used to access the AWS VM at the IP address provided in the email. The .zip file is password is CSEC731SPRING2020!

You may access gitlab at https://gitlab.csec731.com. You'll have to create an account for yourself using your RIT email address. Be aware that email integration is not enabled, so you'll have to contact me if you forget your password.

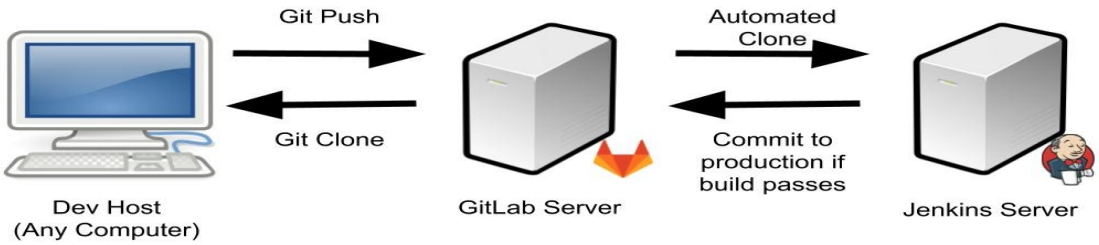
Description:

Modern software development processes are becoming increasingly automated. Long gone are the days of lone software developers working on projects housed on their individual machines. Code repository tools such as GitHub and GitLab have become commonplace and the build process is pushing ever closer to real-time. Continuous integration pipelines are the most recent iteration of this technology. These pipelines automate the software build process so that software updates can be pushed into production as soon as they are pushed to code repositories.

Because code is being pushed more quickly into production, quality assurance is becoming more important. Quality assurance has always involved scripted security tests, but those tests were

traditionally executed by quality assurance engineers that then analyzed the results. Because of the shift towards real-time code deployment, modern application security testing needs to not only be automated, but to be an integral part of the build process. Tools like Jenkins and Travis CI permit this by ensuring that software only gets pushed into production if it pass build and security tests defined by those responsible for quality assurance testing.

In this lab, you will build a VM that will function as a core components of a continuous integration pipeline capable of automating security testing. On the your VM, you will deploy Jenkins and PHPUnit. You will also write a very small web application, developed on any machine, whose code will be housed in the GitLab server. When a change to the web application source code is committed to the GitLab Server, Jenkins will pull the source code from the GitLab server. Jenkins will run some security tests with PHPUnit. If those security tests pass, Jenkins will push the code into a production branch of the web application on the GitLab server.



Unconfigured Ubuntu (16.04 or 18.04) virtual machines will be provided to you to serve as your Jenkins servers.

Requirements

1. Install Jenkins and PHPUnit on your AWS server.
    a. Be aware that apt may install different versions of PHPUnit on Ubuntu 16.04 and Ubuntu 18.04 or on Ubuntu Desktop and Ubuntu Server.
    b. Be aware that apt may install a version different from the one available on the developer's website.
    c. To be safe, verify all hosts you use in this project (including the host you use for development) are running the same version of PHPUnit.
    d. The particular version of PHPUnit you use doesn't matter as long as all the pieces of your CI pipeline work together.
2. Configure Jenkins such that…
    a. Jenkins watches for a push to your GitLab repo.
    b. When Jenkins observes a push to your GitLab repo, it pulls the repo and runs build scripts.
    c. If the build scripts pass, Jenkins should push code into a production branch of your GitLab repository.

3. The web application that you write should be a small API in PHP which lets you write files to /tmp/webapp.
    a. There should be up to three parameters passed via GET
        i. method
            1. create
                a. Should create a file with the name of the value in the name parameter with the contents of the value of the contents parameter.
                b. Should return an HTTP response with text: “File Created”
                c. A new file should overwrite an existing file
            2. del
                a. Should remove the file with the name specified in the value of the name parameter.
                b. Should return an HTTP response with text: “File Deleted”
            3. view
                a. Should return an HTTP response with the file contents of the file specified in the value of the name parameter.
        ii. name
            1. The name of the file to be created/deleted/viewed in /tmp/webapp.
        iii. contents
            1. A string containing text to be put into the file. May include any symbols.
4. You should develop PHPUnit tests to ensure that each of the methods behaves as expected.
    a. If any request takes longer than 1 second to process, it should fail the build.
        i. See: https://phpunit.de/manual/6.5/en/risky-tests.html#risky-tests.test-execution-timeout
    b. Security has asked you to develop tests to:
        i. Verify that bash command injection is not possible through the name parameter when the create command is used by testing that a netcat listener cannot be launched.
            1. The netcat command you should use is nc -nlvp 8000
        ii. Verify that directory traversal is not possible through the name parameter when the view method is used.
            1. The name you should use in your check is ../../../../../../../etc/passwd

Deliverables

1. A PDF document which…
    a. Documents the IP addresses for your Jenkins server.
    b. Documents your username and password on https://gitlab.csec731.com

c. Documents all user credentials in use in the environment along with how they are used.

    i. You do not need to include the contents of any private key files that I may or may not provide you; just provide the name of the private key file.

Grading Method

1. This assignment will be graded by:

    a. Cloning the repository name submitted on the GitLab server

    b. Pushing a change which introduces a directory traversal vulnerability into the view method of the API

        i. The build should fail when this change is made.

    c. Pushing a change which removes the directory traversal vulnerability.

        i. The build should pass, although the production branch may not be updated.

    d. Pushing a change which introduces a remote code execution vulnerability into the create method of the API.

        i. The build should fail when this change is made.

    e. Pushing a change which remotes the directory traversal vulnerability and introduces some additional functionality.

        i. The build should pass and the production branch of the repository should be updated.

Start Date

Mar 23, 2020 6:00 AM

Due Date

Apr 3, 2020 11:55 PM

Submit Assignment

Files to submit *

(0) file(s) to submit

IMPORTANT: After uploading, you must click Submit to complete the submission.

If you are not told on the next screen: ** FILE SUBMISSION SUCCESSFUL **, then your assignment was not submitted and no file has been saved for your instructor.

Add a File

Comments

Paragraph

Font Family Font Size

Submit Cancel