Cyber Security and Computer Forensics

ISN2514: DevOpsSec Automation

Class Participation Task

Group Name: MI6

Members:

Sashim Lama

Rasheedat Kemi Lawal

Riddhi Roshan Mhatre

Aadith Preetham Nandagopal

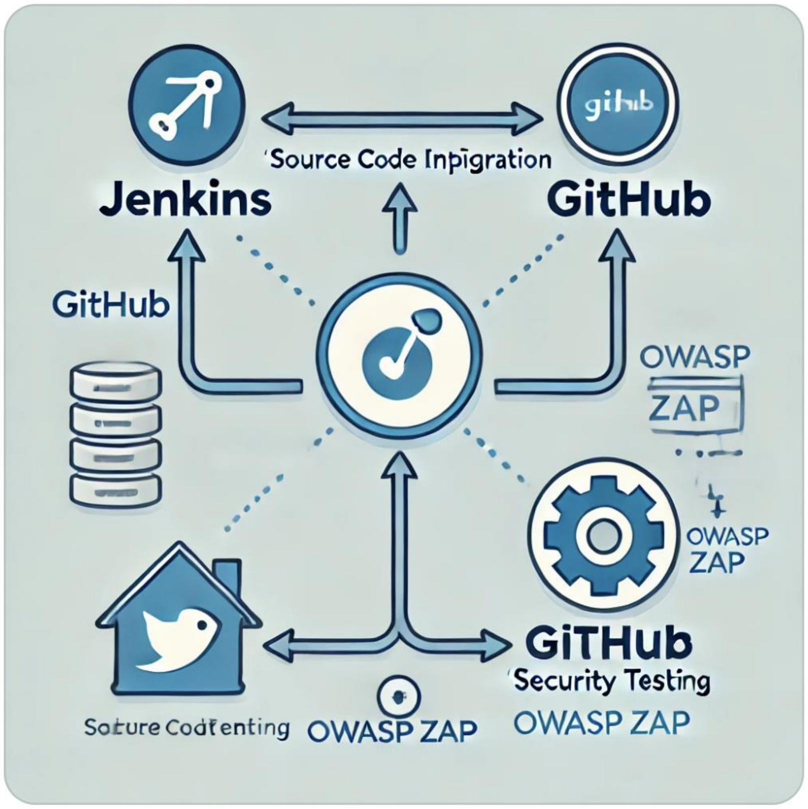
Vincent Paolo Nuarin

Babafemi Onanuga

Date of Submission: October 4, 2024

Table of Contents

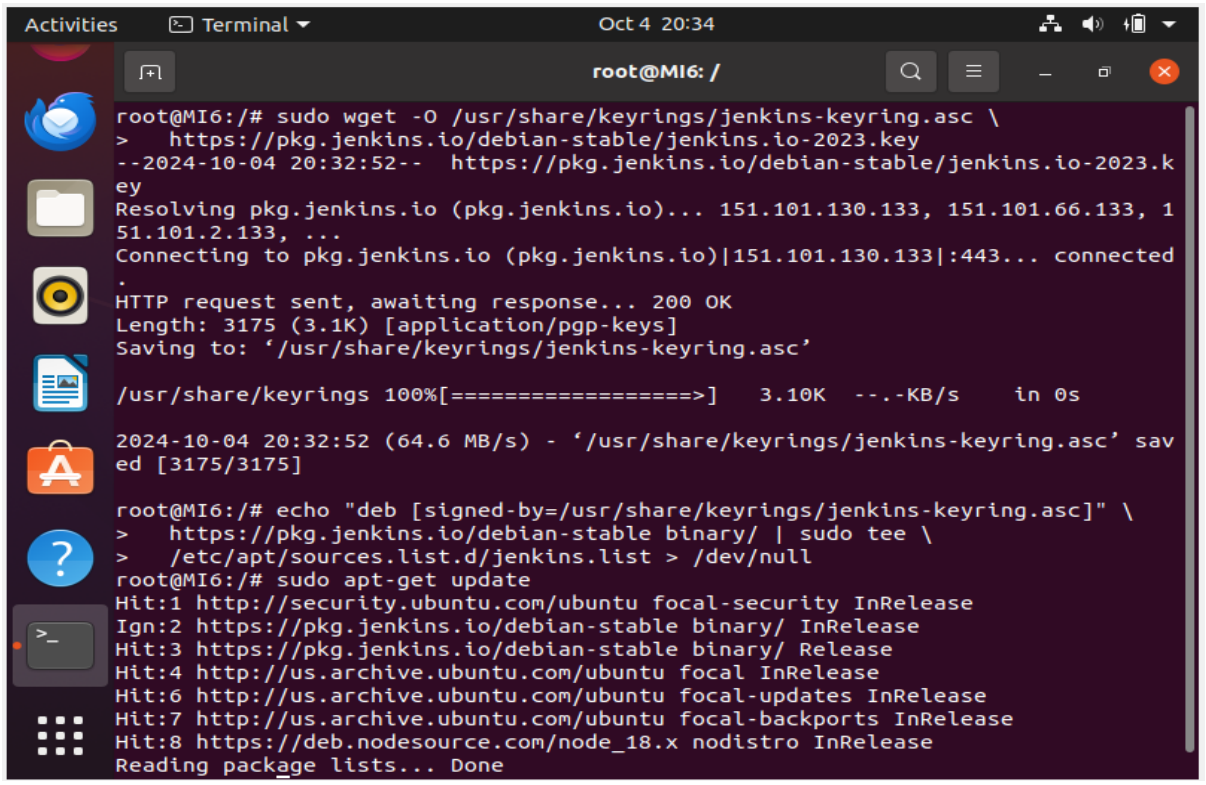
High-Level Diagram



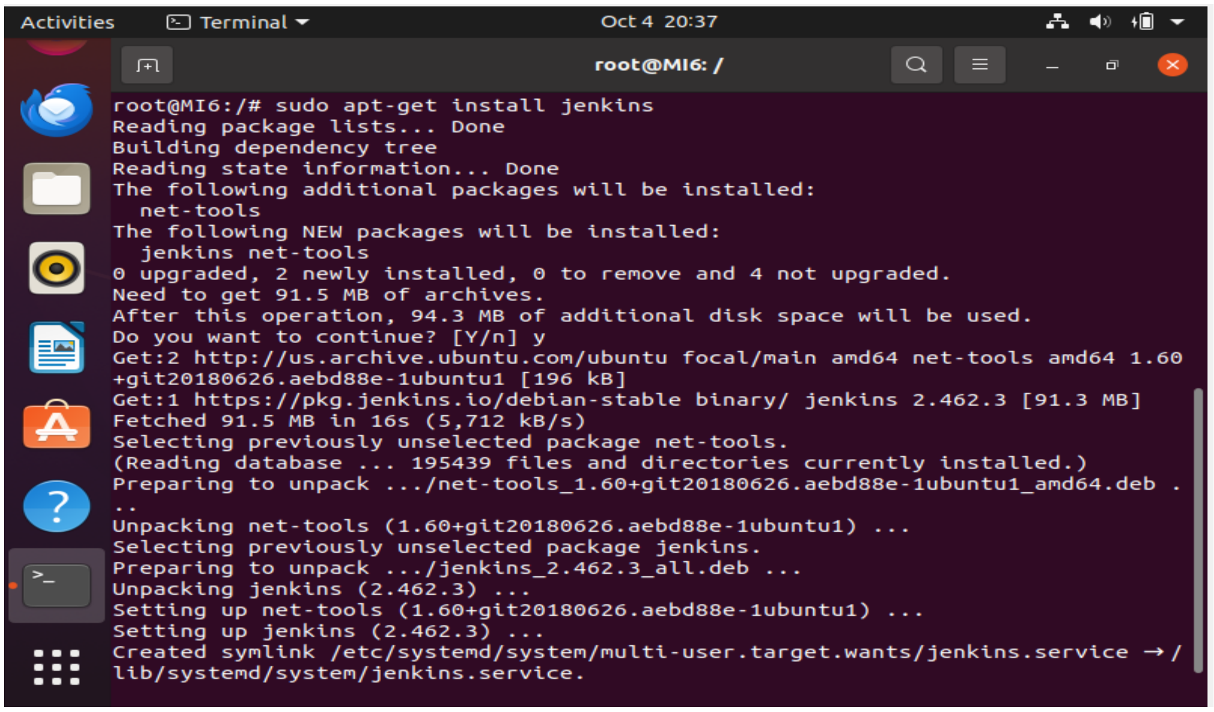
In the diagram:

GitHub → Jenkins: Under the "Source Code Integration" banner, Jenkins retrieves source code from GitHub for continuous integration.
Jenkins → OWASP ZAP: Jenkins initiates "Security Testing" (also known as "Code or Application Security Testing") using OWASP ZAP.
GitHub ← OWASP ZAP (via Jenkins): Jenkins is used to indirectly test code from GitHub for vulnerabilities in OWASP ZAP.
In a continuous development pipeline, this configuration automates security testing and code integration.

Jenkins installation process
Step 1: Download Jenkins



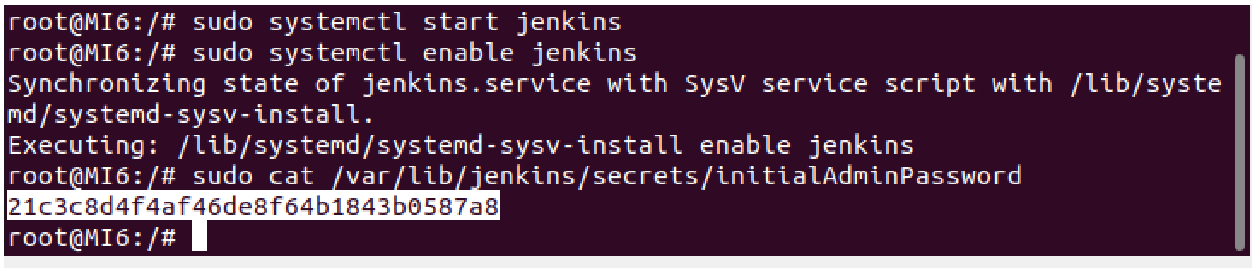
```
root@MI6:/# sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
>   https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
--2024-10-04 20:32:52--  https://pkg.jenkins.io/debian-stable/jenkins.io-2023.k
ey
Resolving pkg.jenkins.io (pkg.jenkins.io)... 151.101.130.133, 151.101.66.133, 1
51.101.2.133, ...
Connecting to pkg.jenkins.io (pkg.jenkins.io)|151.101.130.133|:443... connected
.
HTTP request sent, awaiting response... 200 OK
Length: 3175 (3.1K) [application/pgp-keys]
Saving to: '/usr/share/keyrings/jenkins-keyring.asc'

/usr/share/keyrings 100%[===================>]   3.10K  --.-KB/s    in 0s

2024-10-04 20:32:52 (64.6 MB/s) - '/usr/share/keyrings/jenkins-keyring.asc' sav
ed [3175/3175]

root@MI6:/# echo "deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc]" \
>   https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
>   /etc/apt/sources.list.d/jenkins.list > /dev/null
root@MI6:/# sudo apt-get update
Hit:1 http://security.ubuntu.com/ubuntu focal-security InRelease
Ign:2 https://pkg.jenkins.io/debian-stable binary/ InRelease
Hit:3 https://pkg.jenkins.io/debian-stable binary/ Release
Hit:4 http://us.archive.ubuntu.com/ubuntu focal InRelease
Hit:6 http://us.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:7 http://us.archive.ubuntu.com/ubuntu focal-backports InRelease
Hit:8 https://deb.nodesource.com/node_18.x nodistro InRelease
Reading package lists... Done
```



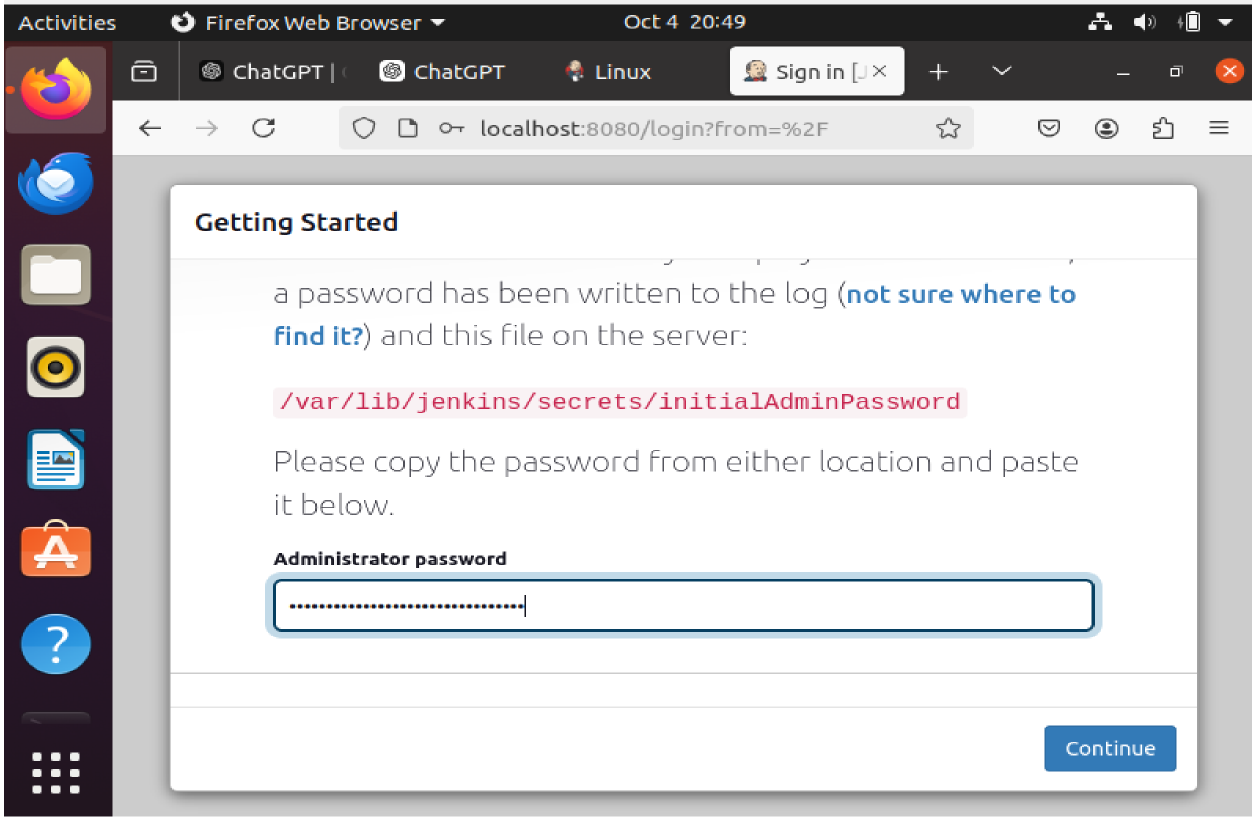
```
root@MI6:/# sudo apt-get install jenkins
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  net-tools
The following NEW packages will be installed:
  jenkins net-tools
0 upgraded, 2 newly installed, 0 to remove and 4 not upgraded.
Need to get 91.5 MB of archives.
After this operation, 94.3 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:2 http://us.archive.ubuntu.com/ubuntu focal/main amd64 net-tools amd64 1.60
+git20180626.aebd88e-1ubuntu1 [196 kB]
Get:1 https://pkg.jenkins.io/debian-stable binary/ jenkins 2.462.3 [91.3 MB]
Fetched 91.5 MB in 16s (5,712 kB/s)
Selecting previously unselected package net-tools.
(Reading database ... 195439 files and directories currently installed.)
Preparing to unpack .../net-tools_1.60+git20180626.aebd88e-1ubuntu1_amd64.deb .
..
Unpacking net-tools (1.60+git20180626.aebd88e-1ubuntu1) ...
Selecting previously unselected package jenkins.
Preparing to unpack .../jenkins_2.462.3_all.deb ...
Unpacking jenkins (2.462.3) ...
Setting up net-tools (1.60+git20180626.aebd88e-1ubuntu1) ...
Setting up jenkins (2.462.3) ...
Created symlink /etc/systemd/system/multi-user.target.wants/jenkins.service → /
lib/systemd/system/jenkins.service.
```

These commands are used to install Jenkins on a Debian-based Linux system. First, the wget command downloads the Jenkins GPG key from Jenkins' official site and saves it to the system's keyrings directory. Then, an entry for the Jenkins repository is added to the system's sources.list using tee, which will allow the package manager to access Jenkins packages. After that, sudo apt-get update updates the system's package lists to include Jenkins. Finally, sudo apt-get install jenkins installs Jenkins from the newly added repository.

Step2: Run Jenkins



```
root@MI6:/# sudo systemctl start jenkins
root@MI6:/# sudo systemctl enable jenkins
Synchronizing state of jenkins.service with SysV service script with /lib/syste
md/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable jenkins
root@MI6:/# sudo cat /var/lib/jenkins/secrets/initialAdminPassword
21c3c8d4f4af46de8f64b1843b0587a8
root@MI6:/# 
```



Getting Started

a password has been written to the log (not sure where to find it?) and this file on the server:

/var/lib/jenkins/secrets/initialAdminPassword

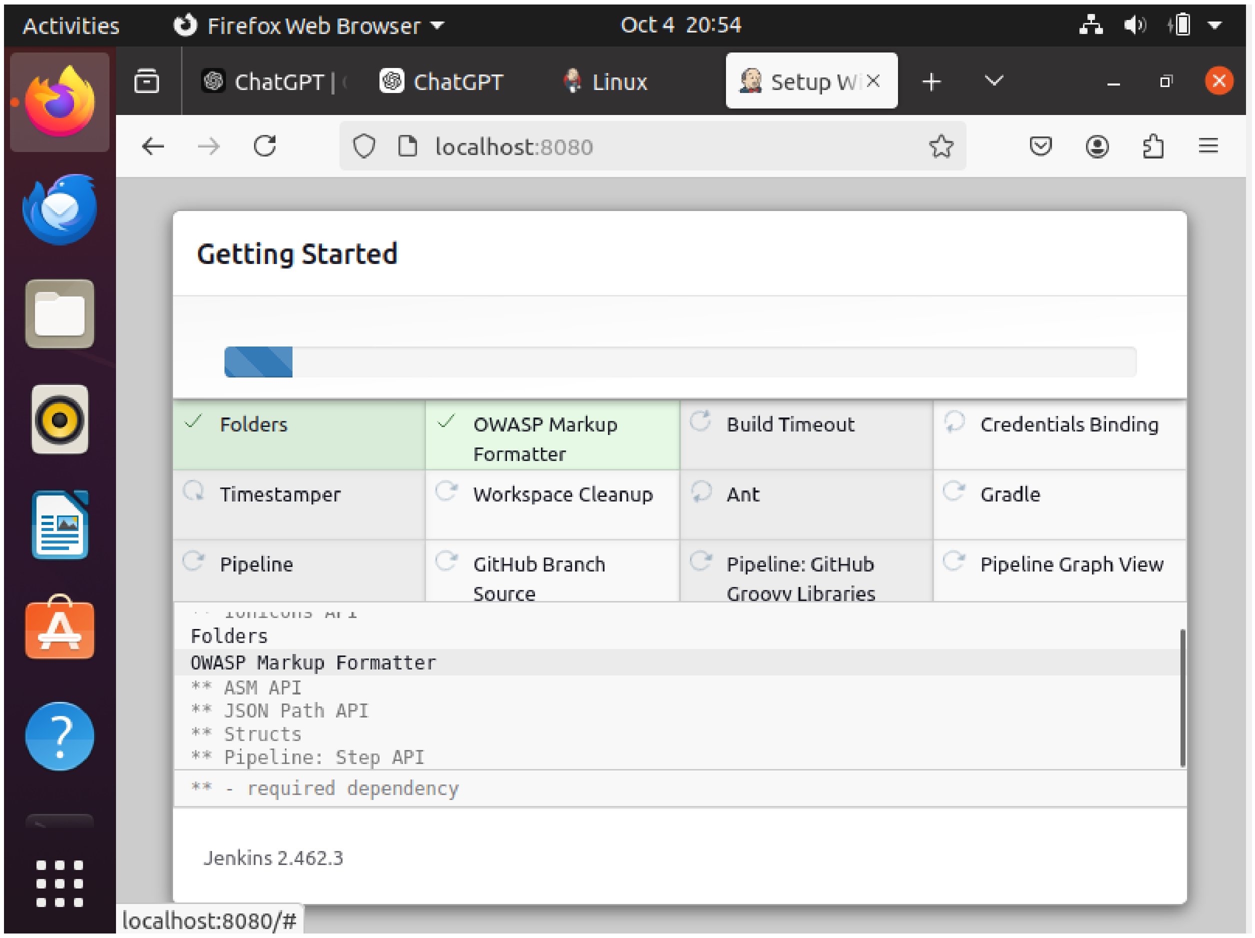
Please copy the password from either location and paste it below.

Administrator password

....

Continue

The sudo systemctl start jenkins command starts the Jenkins service, making it run on our system. The sudo systemctl enable jenkins ensures Jenkins will automatically start at system boot. Finally, the sudo cat /var/lib/jenkins/secrets/initialAdminPassword retrieves the initial administrator password from the Jenkins installation directory, which you need to unlock Jenkins during the first setup in the web interface.



Getting Started

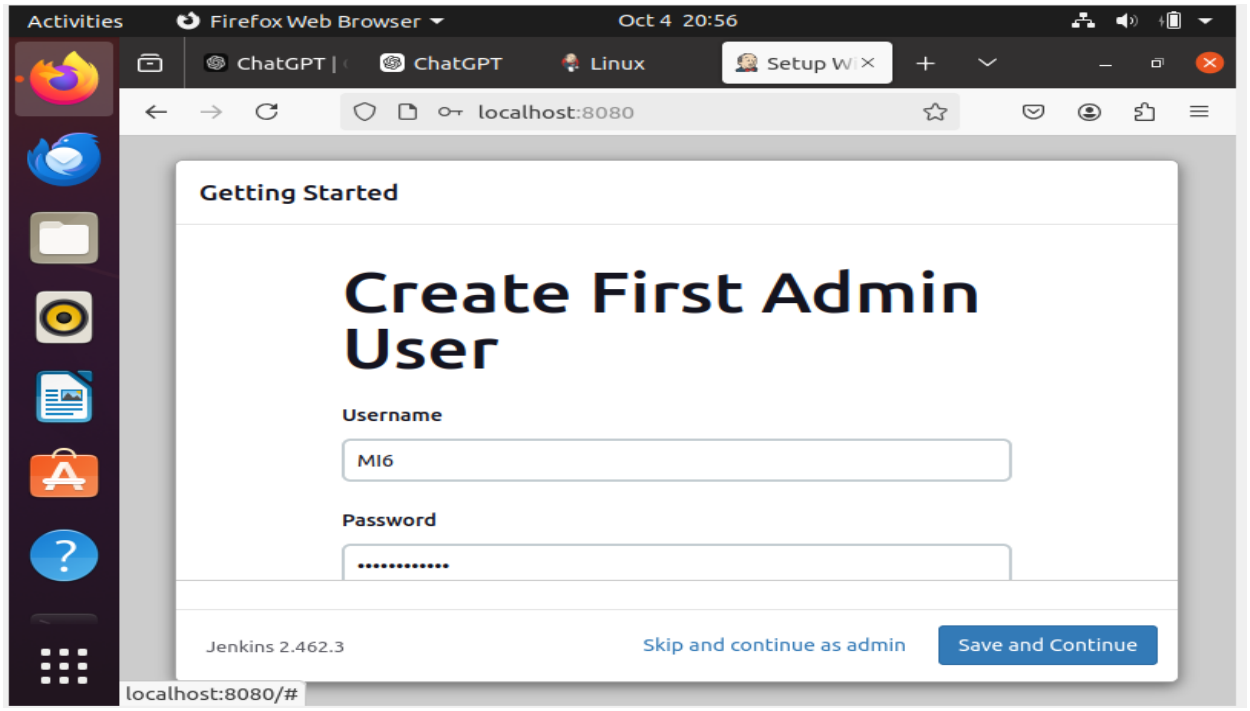
✓ Folders	✓ OWASP Markup Formatter	Build Timeout	Credentials Binding
Timestamper	Workspace Cleanup	Ant	Gradle
Pipeline	GitHub Branch Source	Pipeline: GitHub Groovy Libraries	Pipeline Graph View

```
Folders
OWASP Markup Formatter
** ASM API
** JSON Path API
** Structs
** Pipeline: Step API
** - required dependency
```

Jenkins 2.462.3

localhost:8080/#

Install Suggested Plugins



Getting Started

Create First Admin User

Username

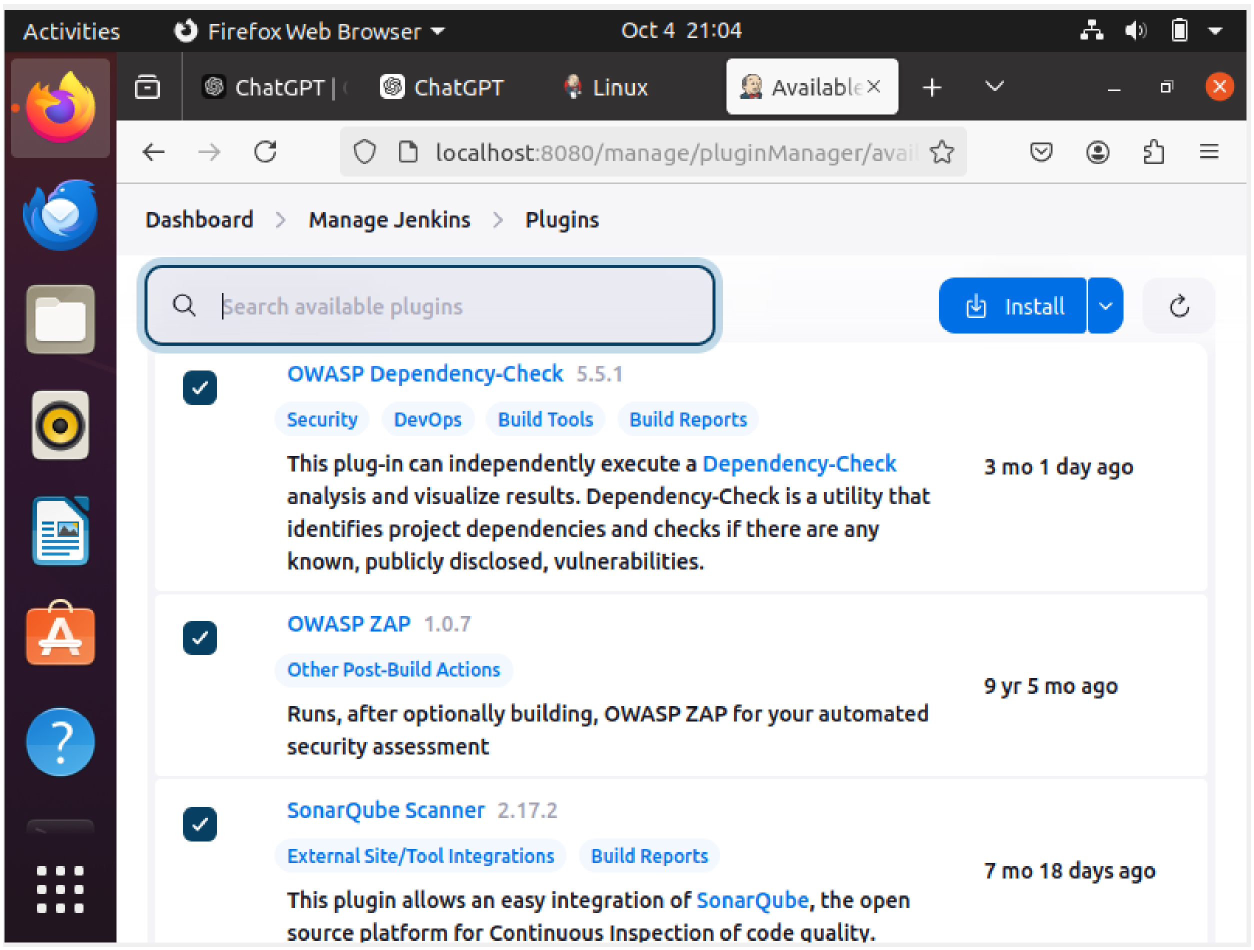
MI6

Password

•••••••••••••

Jenkins 2.462.3

Set up a basic CI/CD pipeline to build a sample application

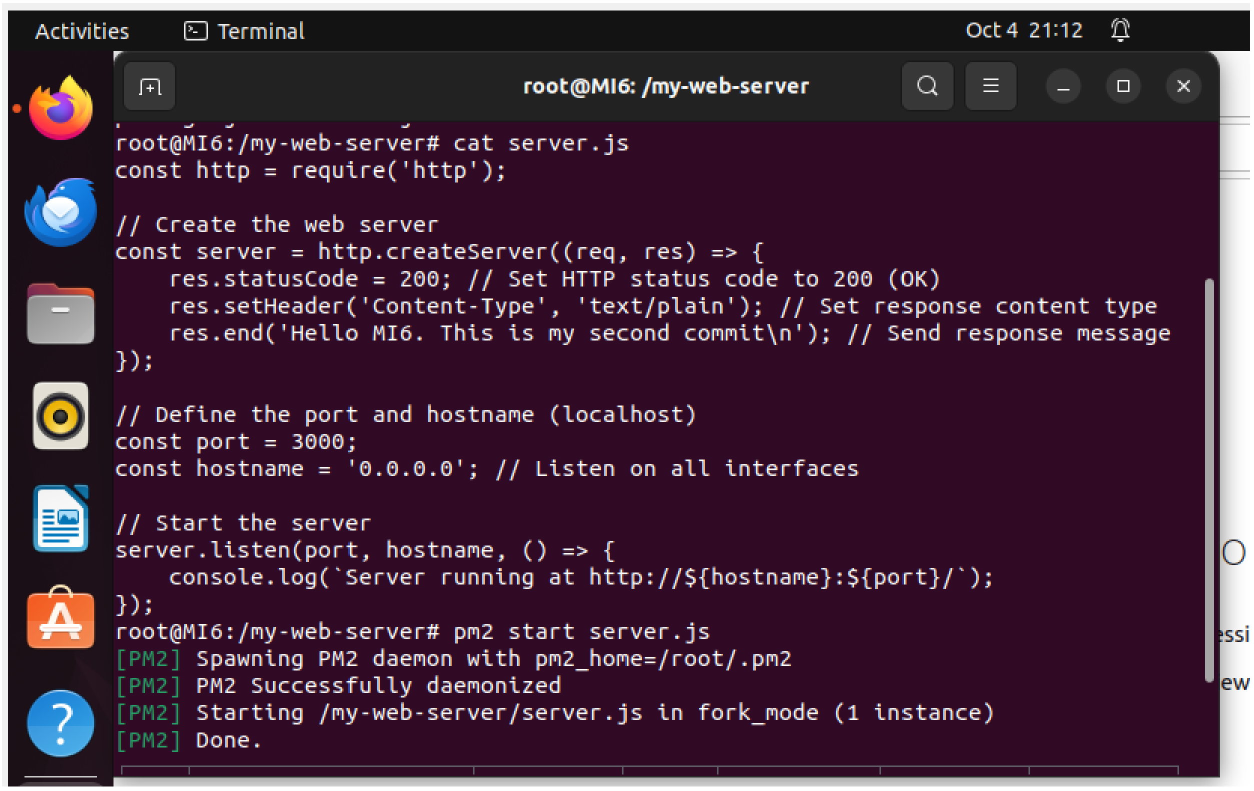


OWASP Dependency-Check 5.5.1
Security DevOps Build Tools Build Reports
This plug-in can independently execute a Dependency-Check analysis and visualize results. Dependency-Check is a utility that identifies project dependencies and checks if there are any known, publicly disclosed, vulnerabilities. — 3 mo 1 day ago

OWASP ZAP 1.0.7
Other Post-Build Actions
Runs, after optionally building, OWASP ZAP for your automated security assessment — 9 yr 5 mo ago

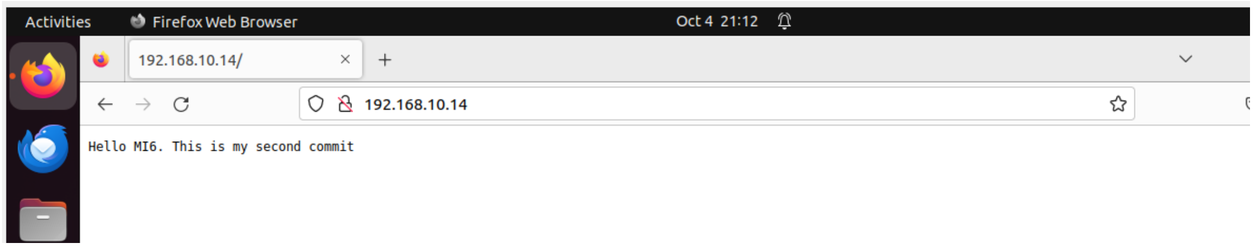
SonarQube Scanner 2.17.2
External Site/Tool Integrations Build Reports
This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality. — 7 mo 18 days ago

Install Required Plugins For CI/CD Security Check. These plugins assist Jenkins pipelines in automating security and code quality assessments.



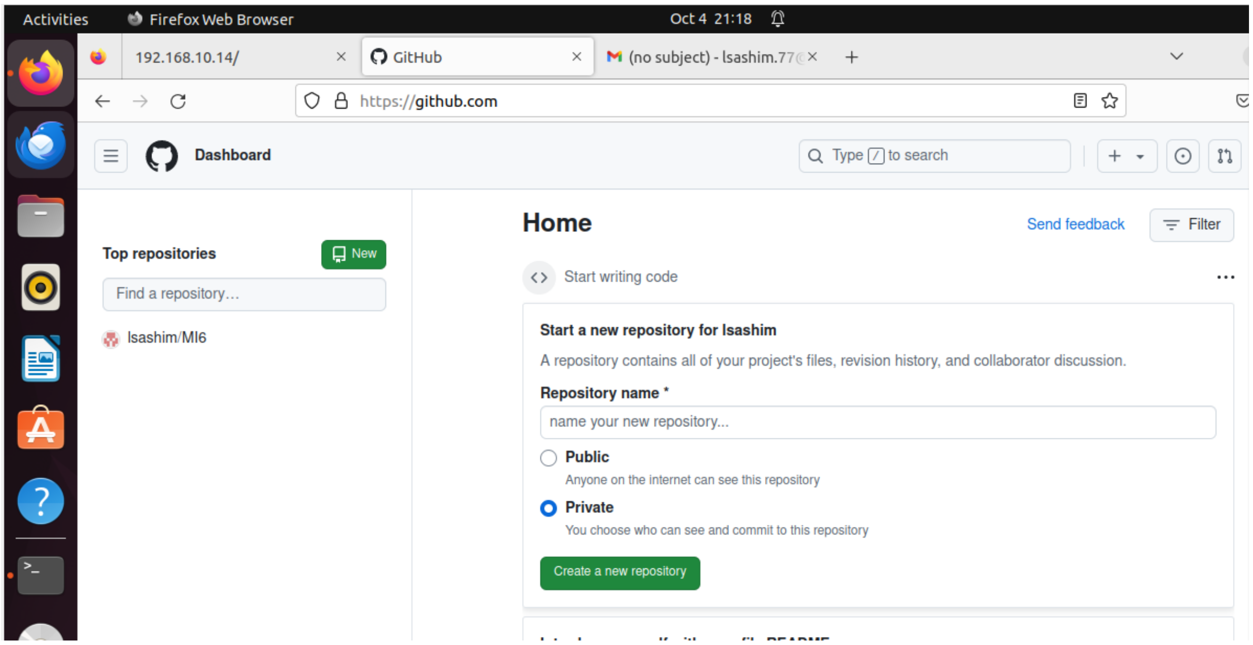
```
root@MI6:/my-web-server# cat server.js
const http = require('http');

// Create the web server
const server = http.createServer((req, res) => {
    res.statusCode = 200; // Set HTTP status code to 200 (OK)
    res.setHeader('Content-Type', 'text/plain'); // Set response content type
    res.end('Hello MI6. This is my second commit\n'); // Send response message
});

// Define the port and hostname (localhost)
const port = 3000;
const hostname = '0.0.0.0'; // Listen on all interfaces

// Start the server
server.listen(port, hostname, () => {
    console.log(`Server running at http://${hostname}:${port}/`);
});
root@MI6:/my-web-server# pm2 start server.js
[PM2] Spawning PM2 daemon with pm2_home=/root/.pm2
[PM2] PM2 Successfully daemonized
[PM2] Starting /my-web-server/server.js in fork_mode (1 instance)
[PM2] Done.
```

Create a Simple Web Server in Node.js



```
Hello MI6. This is my second commit
```

Running Web Server



Start a new repository for lsashim

A repository contains all of your project's files, revision history, and collaborator discussion.

Repository name *

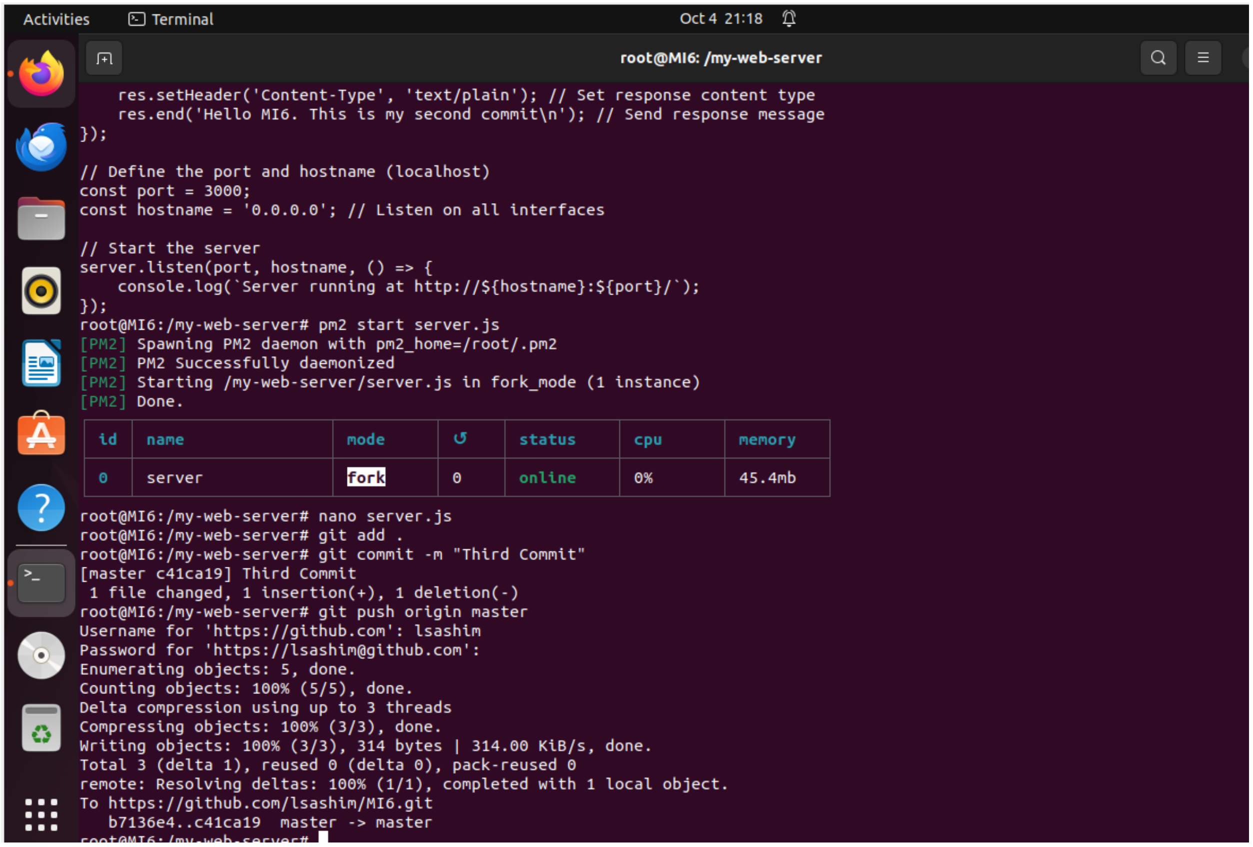
name your new repository...

( ) Public
Anyone on the internet can see this repository

[✓ Private]
You choose who can see and commit to this repository

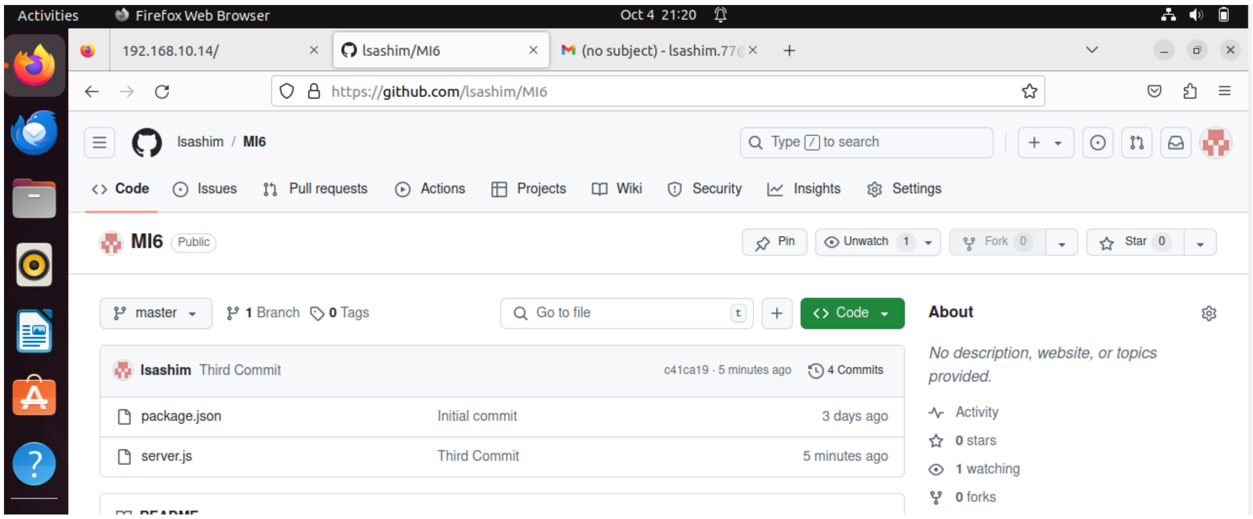
Create a new repository

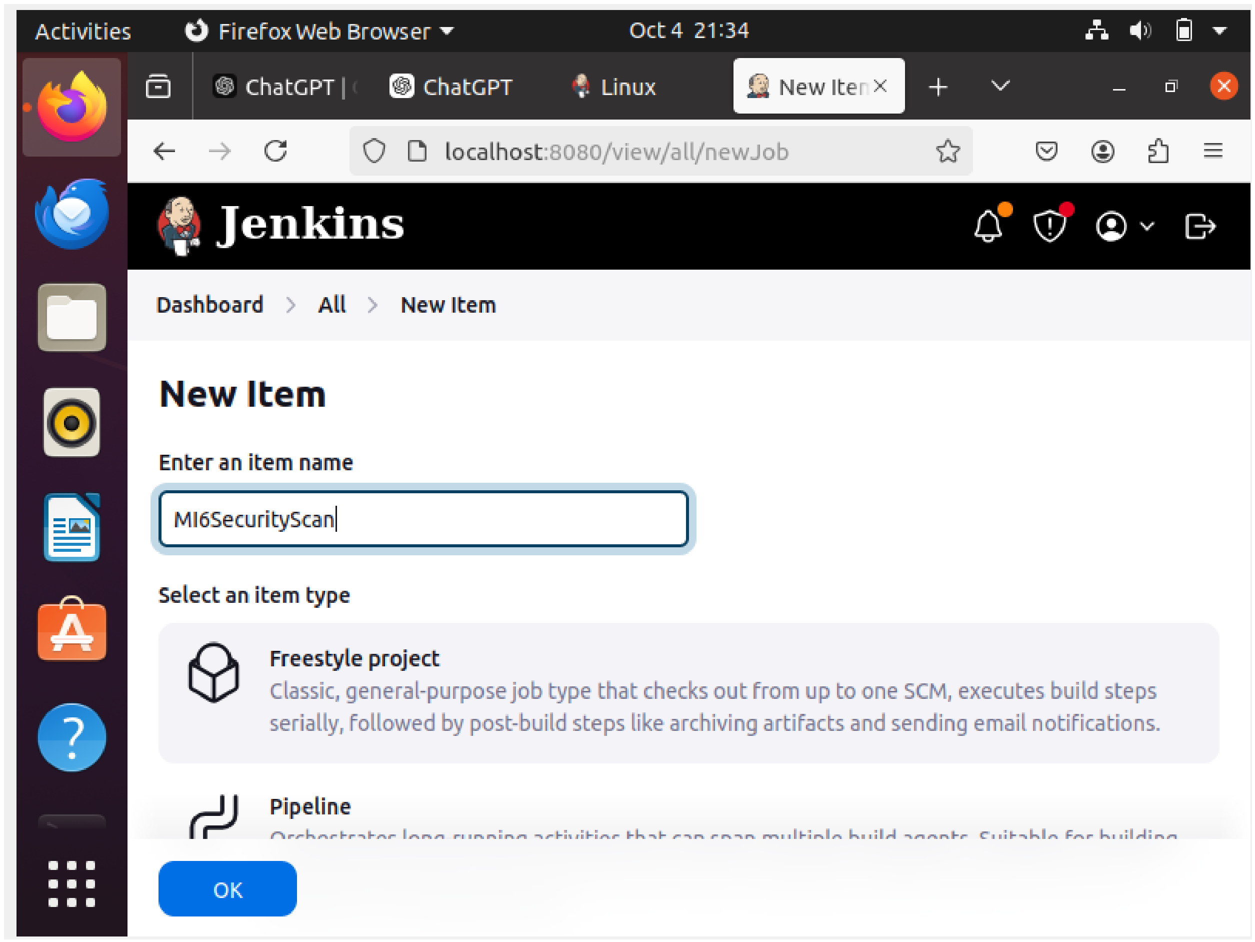
Fig: Create a Git Repo for Webserver

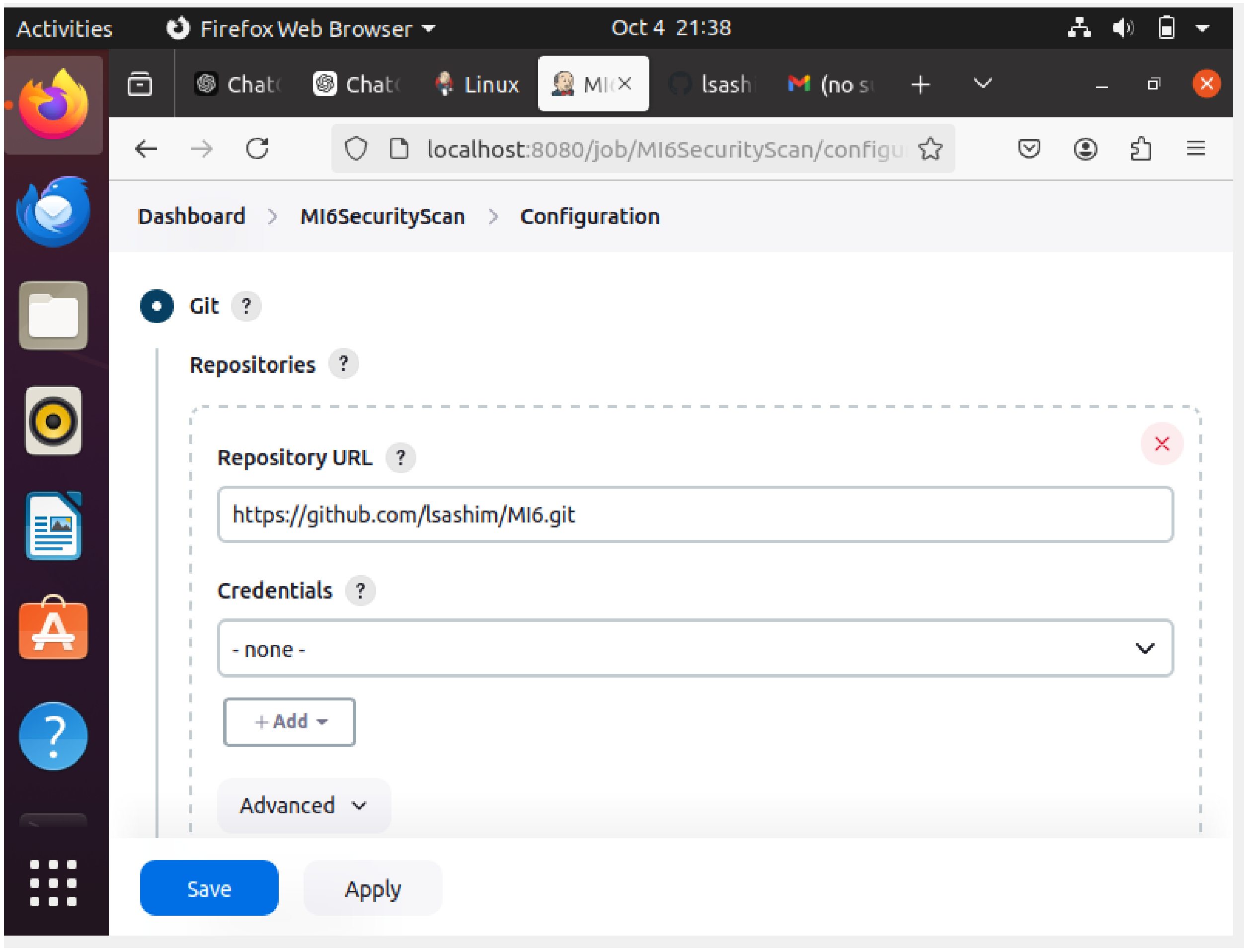


```
    res.setHeader('Content-Type', 'text/plain'); // Set response content type
    res.end('Hello MI6. This is my second commit\n'); // Send response message
});

// Define the port and hostname (localhost)
const port = 3000;
const hostname = '0.0.0.0'; // Listen on all interfaces

// Start the server
server.listen(port, hostname, () => {
    console.log(`Server running at http://${hostname}:${port}/`);
});
root@MI6:/my-web-server# pm2 start server.js
[PM2] Spawning PM2 daemon with pm2_home=/root/.pm2
[PM2] PM2 Successfully daemonized
[PM2] Starting /my-web-server/server.js in fork_mode (1 instance)
[PM2] Done.
┌────┬────────────────────┬──────────┬──────┬───────────┬──────────┬──────────┐
│ id │ name               │ mode     │ ↺    │ status    │ cpu      │ memory   │
├────┼────────────────────┼──────────┼──────┼───────────┼──────────┼──────────┤
│ 0  │ server             │ fork     │ 0    │ online    │ 0%       │ 45.4mb   │
└────┴────────────────────┴──────────┴──────┴───────────┴──────────┴──────────┘
root@MI6:/my-web-server# nano server.js
root@MI6:/my-web-server# git add .
root@MI6:/my-web-server# git commit -m "Third Commit"
[master c41ca19] Third Commit
 1 file changed, 1 insertion(+), 1 deletion(-)
root@MI6:/my-web-server# git push origin master
Username for 'https://github.com': lsashim
Password for 'https://lsashim@github.com':
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 3 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 314 bytes | 314.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/lsashim/MI6.git
   b7136e4..c41ca19  master -> master
root@MI6:/my-web-server#
```

Fig: Push the change in Git repo

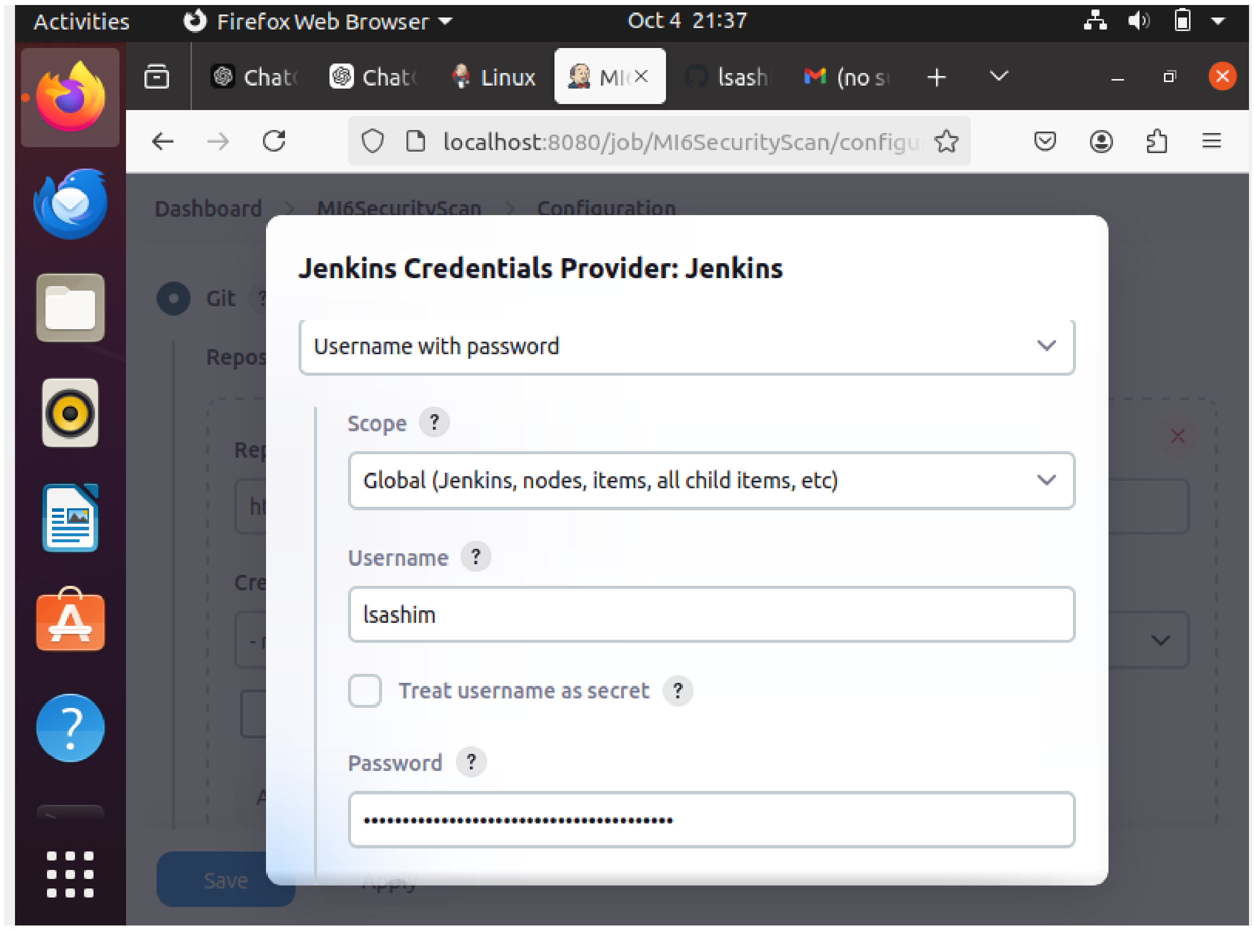

Git Commit Success


Create New Items



Repository URL: https://github.com/lsashim/MI6.git

Credentials: - none -



Activities
Firefox Web Browser
Oct 4 21:37
localhost:8080/job/MI6SecurityScan/configu
Dashboard > MI6SecurityScan > Configuration
Save
Apply

Jenkins Credentials Provider: Jenkins

Username with password

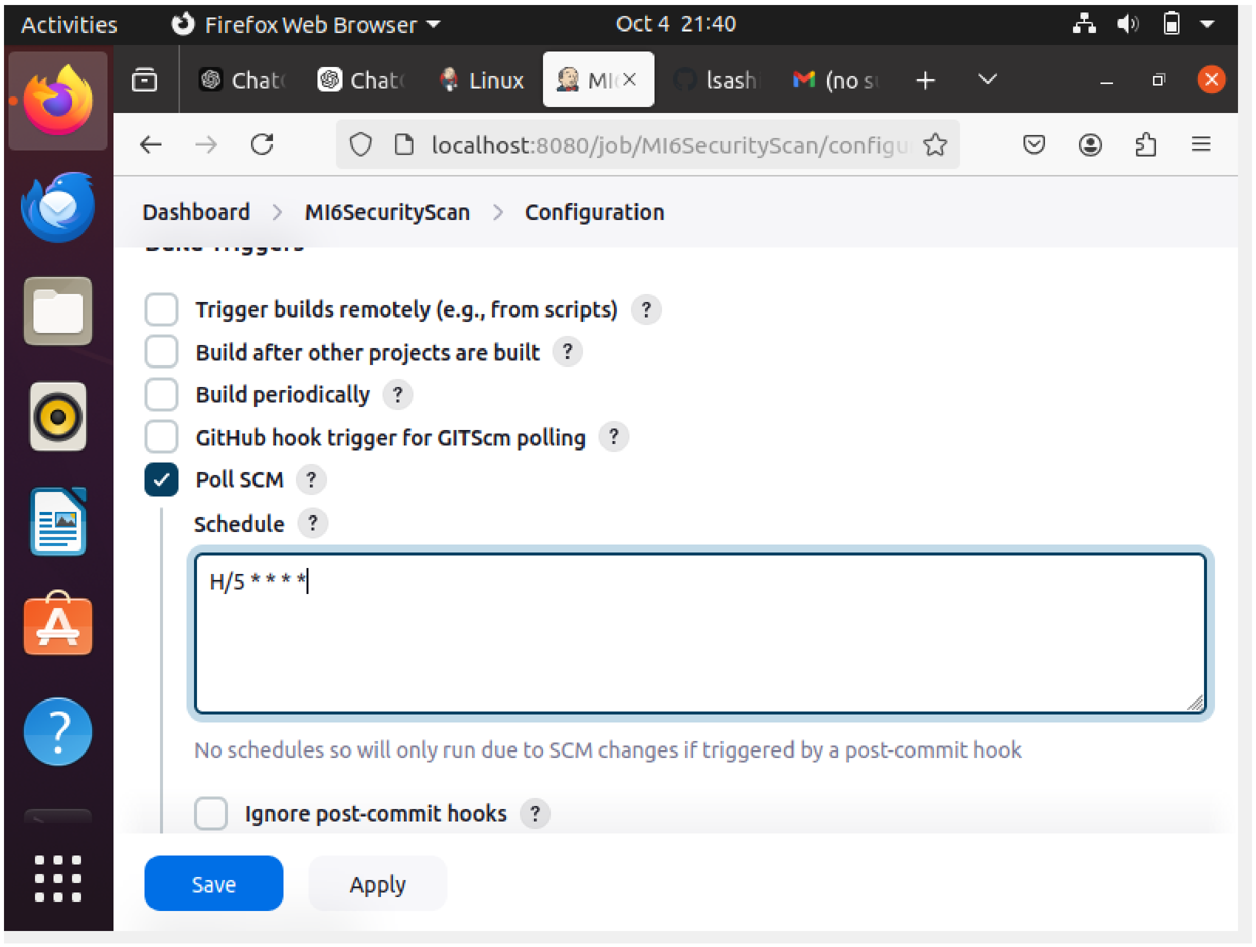
Scope: Global (Jenkins, nodes, items, all child items, etc)

Username: lsashim

Treat username as secret: [ ]

Password: ••

Fig: Source Code Management setup. Jenkins may now safely utilize these credentials, most likely for remote service authentication like a Git repository.


Activities
Firefox Web Browser
Oct 4 21:40
localhost:8080/job/MI6SecurityScan/configu
Dashboard > MI6SecurityScan > Configuration

☐ Trigger builds remotely (e.g., from scripts)
☐ Build after other projects are built
☐ Build periodically
☐ GitHub hook trigger for GITScm polling
✓ Poll SCM

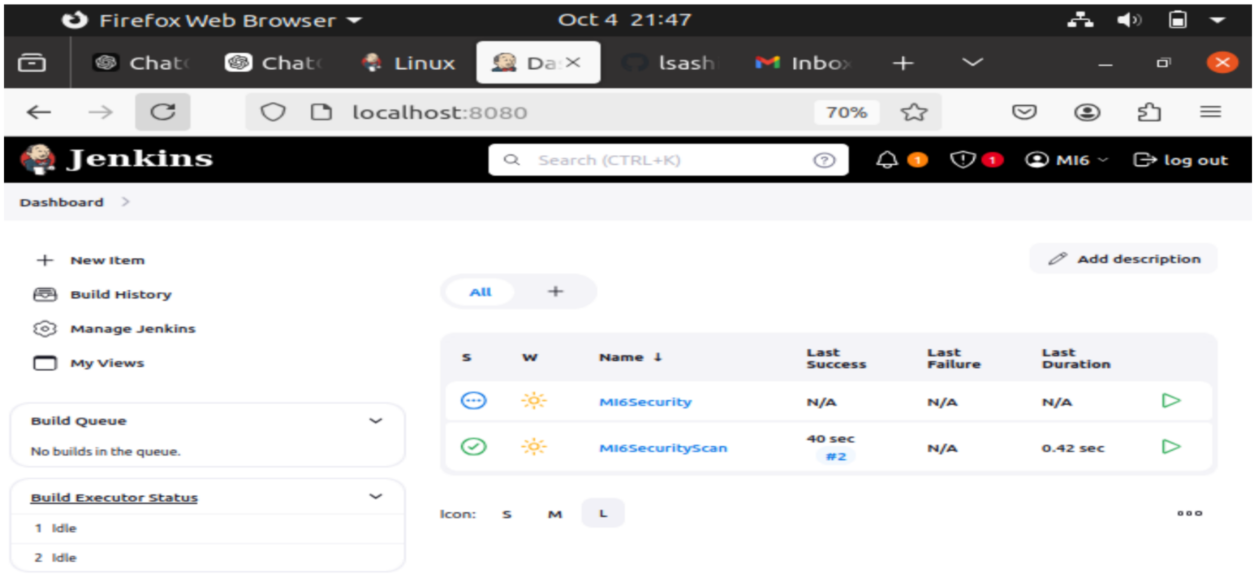
Schedule

```
H/5 * * * *
```

No schedules so will only run due to SCM changes if triggered by a post-commit hook

☐ Ignore post-commit hooks

Set SCM poll. Jenkins' Poll SCM frequently scans the source code repository (such as Git) for updates. Jenkins starts a build if any new changes are found.



Build Queue

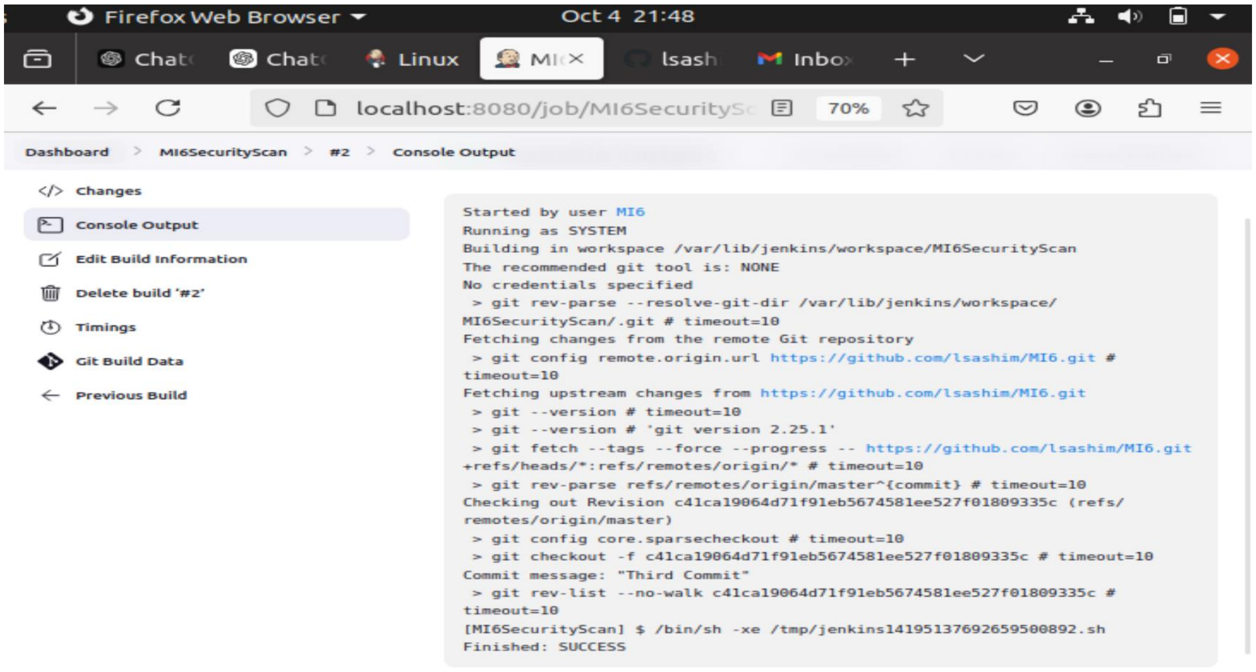
No builds in the queue.

Build Executor Status

1 Idle

2 Idle

S	W	Name ↓	Last Success	Last Failure	Last Duration
		MI6Security	N/A	N/A	N/A
✓		MI6SecurityScan	40 sec #2	N/A	0.42 sec



```
Started by user MI6
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/MI6SecurityScan
The recommended git tool is: NONE
No credentials specified
 > git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/
MI6SecurityScan/.git # timeout=10
Fetching changes from the remote Git repository
 > git config remote.origin.url https://github.com/lsashim/MI6.git #
timeout=10
Fetching upstream changes from https://github.com/lsashim/MI6.git
 > git --version # timeout=10
 > git --version # 'git version 2.25.1'
 > git fetch --tags --force --progress -- https://github.com/lsashim/MI6.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision c41ca19064d71f91eb5674581ee527f01809335c (refs/
remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f c41ca19064d71f91eb5674581ee527f01809335c # timeout=10
Commit message: "Third Commit"
 > git rev-list --no-walk c41ca19064d71f91eb5674581ee527f01809335c #
timeout=10
[MI6SecurityScan] $ /bin/sh -xe /tmp/jenkins14195137692659500892.sh
Finished: SUCCESS
```

CI/CD Integrated. This shows that Jenkins successfully completed the build after obtaining the most recent modifications from GitHub.

REFERENCES:

Sudheer. (2023, October 8). Integrate OWASP Dependency Check in Jenkins pipeline. Medium. https://sudheer-baraker.medium.com/integrate-owasp-dependency-check-in-jenkins-pipeline-748d8aefc2b7

Jenkins with GitHub. (n.d.). Jenkins With GitHub. https://www.jenkins.io/solutions/github/

R, G. (2021, December 13). Integrating OWASP Dependency Check with Jenkins to CI/CD. Medium. https://gowthamr1.medium.com/integrating-owasp-dependency-check-with-jenkins-to-ci-cd-6f00e263fa78