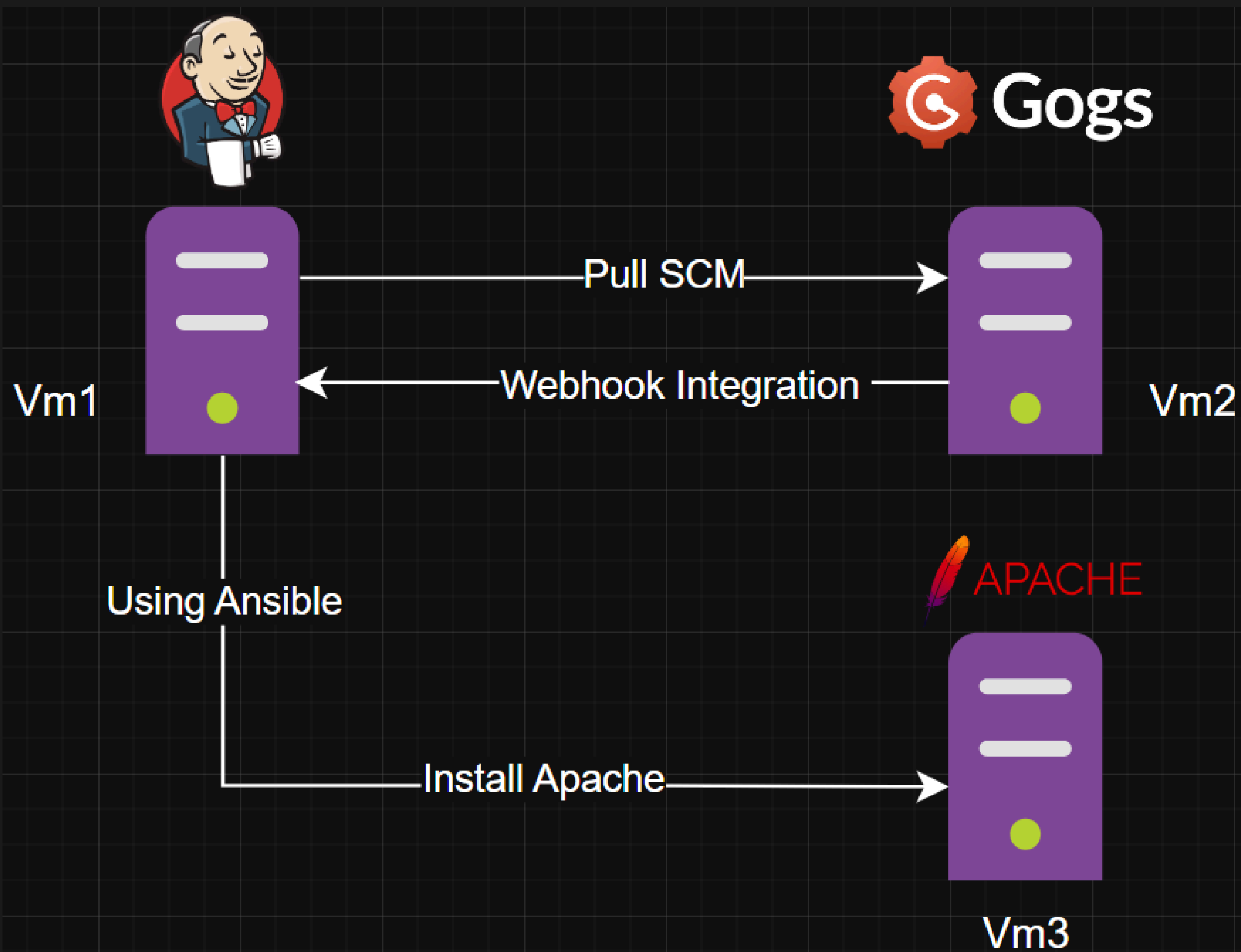


Project Architecture:

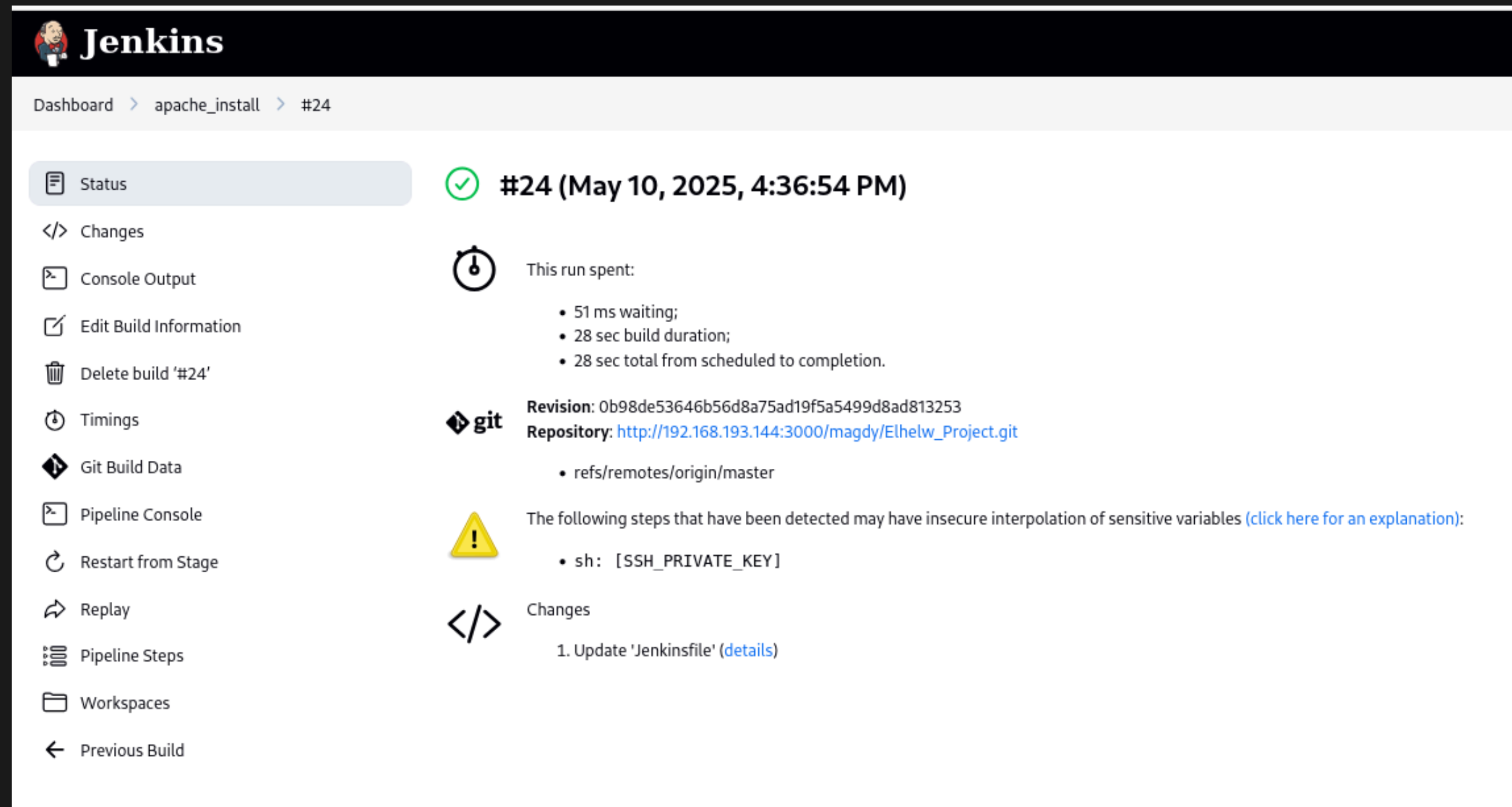


Jenkins:

1- Setup Jenkins

2- Installed Docker and Ansible for the pipeline

3- Configured a pipeline to pull a Jenkinsfile from a repository hosted on a local Gogs server



The screenshot displays the Jenkins web interface for a specific build. The top navigation bar shows the path: Dashboard > apache_install > #24. On the left, a sidebar contains various actions for the build, including Status (highlighted), Changes, Console Output, Edit Build Information, Delete build '#24', Timings, Git Build Data, Pipeline Console, Restart from Stage, Replay, Pipeline Steps, Workspaces, and Previous Build. The main content area shows the build status as successful (green checkmark) and the title '#24 (May 10, 2025, 4:36:54 PM)'. Below this, a clock icon indicates the run duration: 51 ms waiting, 28 sec build duration, and 28 sec total from scheduled to completion. A Git icon shows the revision (0b98de53646b56d8a75ad19f5a5499d8ad813253) and repository (http://192.168.193.144:3000/magdy/Elhelw_Project.git). A warning icon (yellow triangle with exclamation mark) indicates insecure interpolation of sensitive variables in the 'sh' step: [SSH_PRIVATE_KEY]. A code icon shows the changes, listing '1. Update 'Jenkinsfile' (details)'. A link 'click here for an explanation' is provided for the warning.

Dashboard > apache_install > #24

Status

Changes

Console Output

Edit Build Information

Delete build '#24'

Timings

Git Build Data

Pipeline Console

Restart from Stage

Replay

Pipeline Steps

Workspaces

Previous Build

✓ #24 (May 10, 2025, 4:36:54 PM)

This run spent:

- 51 ms waiting;
- 28 sec build duration;
- 28 sec total from scheduled to completion.

git Revision: 0b98de53646b56d8a75ad19f5a5499d8ad813253
Repository: http://192.168.193.144:3000/magdy/Elhelw_Project.git

- refs/remotes/origin/master

⚠ The following steps that have been detected may have insecure interpolation of sensitive variables ([click here for an explanation](#)):

- sh: [SSH_PRIVATE_KEY]

</> Changes

1. Update 'Jenkinsfile' ([details](#))

Gogs Server:

1- Setup a Gogs Server

2- Created a new repository

3- Configured Webhook
integration with Jenkins
Server

with each commit the pipeline
runs

The image displays two screenshots of the Gogs web interface, illustrating the setup of a repository and the configuration of a webhook for Jenkins integration.

Top Screenshot: Repository Overview

- Repository: **magdy / Elhelw_Project**
- Actions: Unwatch (1), Star (0), Fork (0)
- Navigation: Files, Issues (0), Pull Requests (0), Wiki, Settings
- Status: No Description
- Statistics: 18 Commits, 1 Branches, 0 Releases
- Branch: **master**
- Buttons: New file, Upload file, HTTP, SSH, <http://localhost:3000/magdy/E>
- Recent Commits:
 - magdy** (0b98de5364) Update 'Jenkinsfile' (2 hours ago)
 - docker** (649d159540) added Dockerfile (21 hours ago)
 - InstallApache.yml** (7c748126f5) Update 'InstallApache.yml' (6 hours ago)

Bottom Screenshot: Settings Page

- Navigation: Dashboard, Issues, Pull Requests, Explore
- Repository: **magdy / Elhelw_Project**
- Actions: Unwatch (1), Star (0), Fork (0)
- Navigation: Files, Issues (0), Pull Requests (0), Wiki, Settings
- Left Sidebar: Settings (selected), Options, Collaboration, Branches, Webhooks, Git Hooks
- Message: New webhook has been added.
- Webhooks Section:
 - Webhooks are much like basic HTTP POST event triggers. Whenever something occurs in Gogs, we will handle the notification to the target host you specify.%(EXTRA string=<https://gogs.io/docs/features/webhook.html>)
 - Webhook: http://192.168.193.143:8080/gogs-webhook/?job=apache_install

Ansible:

Created a playbook to install Apache on remote server (Vm3)

1- Install Apache

2- Created a Custom configuration file for Apache to:

- Host Web Servers at /srv/www
- allow only Vm1 IP to access the web server
- Disable Server Signature and Server Tokens
- Custom Log Format and custom Log file
- and Copy the file to /etc/httpd/conf.d/ using Ansible copy module
- Used Handlers to restart Apache in case the configuration was copied

custom_httpd.conf 407 B

```
1 # Set web root to /srv/www
2 # Allow Only Vm1 to access
3 DocumentRoot "/srv/www"
4
5 <Directory "/srv/www">
6     Options None
7     AllowOverride None
8     Require ip 192.168.193.143
9 </Directory>
10
11 # Disable ServerSignature and ServerTokens
12 ServerSignature Off
13 ServerTokens Prod
14
15 # Custom Log Format
16 LogFormat "%h %t \"%r\" %>s %b \"%{User-Agent}i\"" custom_Log
17 CustomLog /var/log/httpd/access_custom.log custom_Log
```

```
- name: Copy custom Apache config
  copy:
    src: custom_httpd.conf
    dest: /etc/httpd/conf.d/httpd_srv.conf
    mode: '0644'
  notify:
    - restart_httpd
```

3- Configure log rotation using logrotate :

- It rotates all the logs in /var/log/httpd
- It rotates Logs daily , stores up to 14 rotates, compress the logs and also make sure to reload the service after the rotation
- and Copy the file to /etc/logrotate.d using Ansible copy module

custom_logrotate 274 B

```
1 # Custom Logrotate for all the logs in /var/log/httpd/
2
3 /var/log/httpd/*.log {
4     daily
5     rotate 14
6     notifempty
7     compress
8     delaycompress
9     sharedscripts
10    postrotate
11        /bin/systemctl reload httpd.service > /dev/null 2>/dev/null || true
12    endscript
13 }
```

4- SELinux Context type:

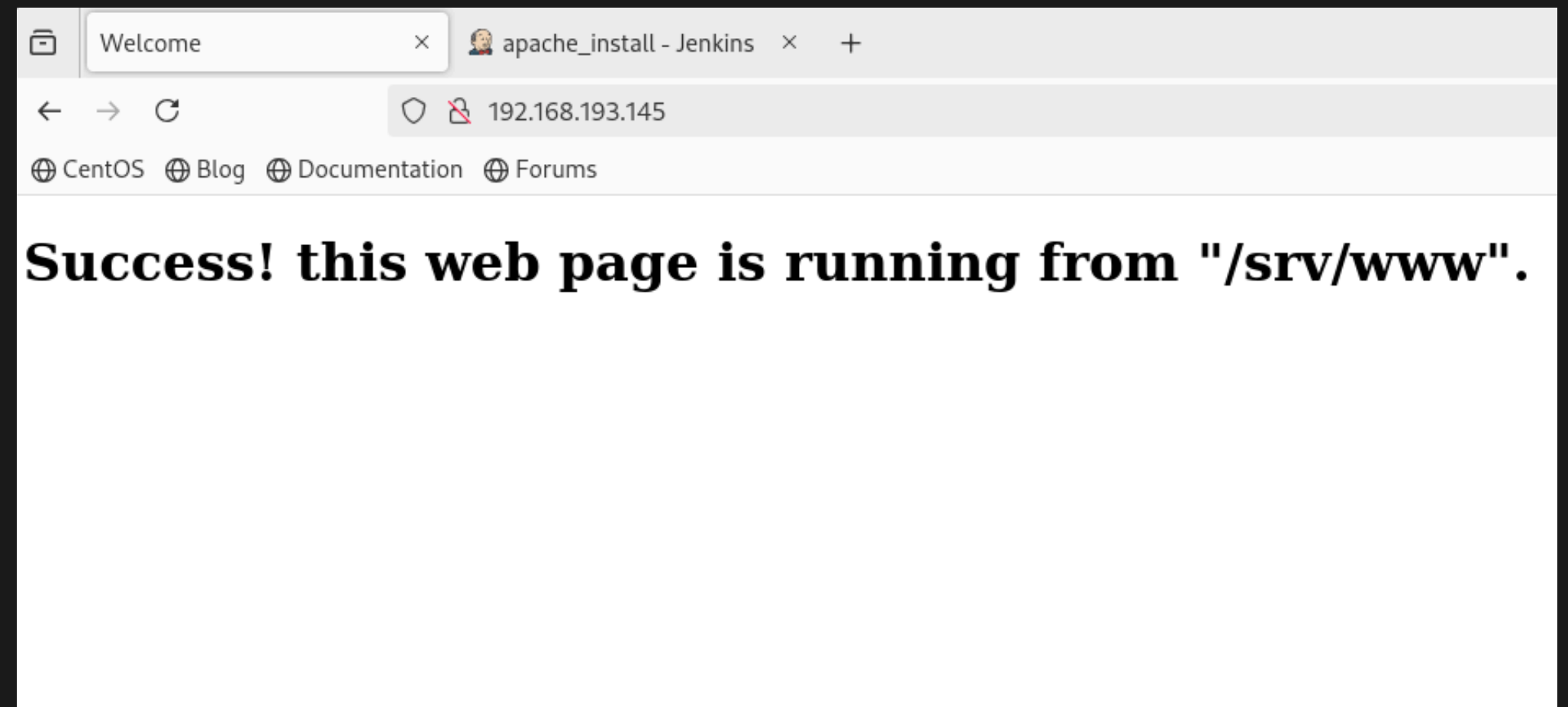
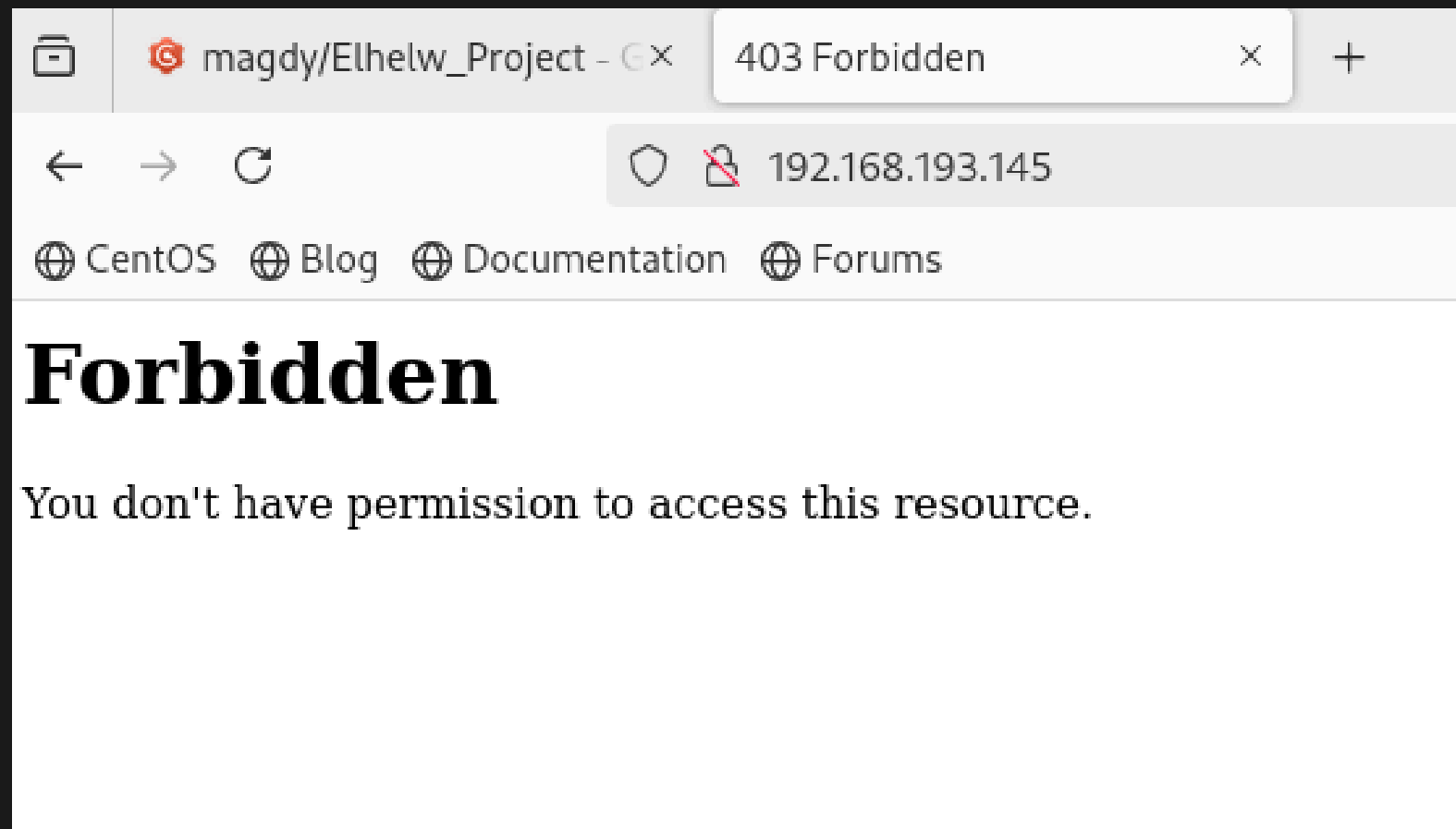
- Used sefcontext to set the default context type for files in /srv/www so that Apache can access them
- Used handler to only restorecon in case the context was changed

```
- name: Allow Apache to serve files from /srv/www
  community.general.sefcontext:
    target: '/srv/www(/.*)?'
    setype: httpd_sys_rw_content_t
    state: present
  notify:
    - restorecon
```

4- Copy a simple Test web page:

- copied to /srv/www to test the new configuration
- Vm1 was able to access the web page but Vm2 and Vm3 itself wasn't able

```
- name: Copy Simple test Page  
  copy:  
    src: index.html  
    dest: /srv/www/index.html
```



JenkinsFile

1-Ansible Build:

- Added an SSH key as a Jenkins Credentials
- Used the user “magdy” and the private key to ssh to Vm3 that already has public key (ssh-copy-id)
- Run The playbook to install Apache on Vm3

```
stage('Ansible Build') {
    steps {
        withCredentials([sshUserPrivateKey(credentialsId: 'ansible_ssh_key', keyFileVariable: 'SSH_PRIVATE_KEY')])
        sh '''
            echo "[Vm3]" > inventory.ini
            echo "192.168.193.145 ansible_user=magdy" >> inventory.ini
            ansible-playbook -i inventory.ini InstallApache.yml --private-key=$SSH_PRIVATE_KEY
            ...
        '''
    }
}
```

2-Build Docker Image:

- build docker image with docker file in repository
- and Save it in tar and get the path for the tar file

```
stage('Build Docker Image') {
    environment {
        DOCKER_IMAGE = "nginx-custom:${BUILD_NUMBER}"
    }
    steps {
        sh 'docker build -t $DOCKER_IMAGE ./docker/'
        sh 'docker save $DOCKER_IMAGE > nginx-custom.tar'
        script {
            TAR_PATH = sh(script: 'realpath nginx-custom.tar', returnStdout: true).trim()
            env.TAR_PATH = TAR_PATH
        }
    }
}
```

3- Post Run

- post action that always run even if the pipeline failed
- used emailext plug in to send an email with
 1. The Status of Pipeline
 2. List of users in deployG group
 3. Date & time of execution
 4. Path for the .tar file

And here is an example of a received Email

```
// Send the email
emailext (
    subject: "Jenkins Pipeline: ${currentBuild.currentResult}",
    to: "${env.EMAIL_RECIPIENT}",
    body: ""
        <h3>Jenkins Pipeline Execution Report</h3>
        <p><b>Status:</b> ${currentBuild.currentResult}</p>
        <p><b>Execution Time (UTC):</b> ${EXEC_TIME}</p>
        <p><b>Group '${DEPLOY_GROUP}' Users:</b> ${DEPLOY_GROUP_USERS}</p>
        <p><b>.tar Image Path:</b> ${TAR_PATH}</p>
    "",
    ,
```

Jenkins Pipeline: SUCCESS Inbox x



address not configured yet <mohamedmagdyelhlew@gmail.com>
to me ▼

Jenkins Pipeline Execution Report

Status: SUCCESS

Execution Time (UTC): 2025-05-10 13:37:20

Group 'deployG' Users: Devo,Testo,Prodo

.tar Image Path: /var/lib/jenkins/workspace/apache_install/nginx-custom.tar

← Reply

→ Forward



System Hardening on Vm3

- Created a simple Script to disable unnecessary services

```
#!/bin/bash

# services to disable/mask
SERVICES=(
    avahi-daemon.service
    bluetooth.service
    cups.service
    ModemManager.service
    switcheroo-control.service
    avahi-daemon.socket
    cups.socket
    cups.path
)

echo "Disabling and masking unnecessary services..."

for svc in "${SERVICES[@]}"; do
    if systemctl list-unit-files | grep -q "$svc"; then
        echo "Disabling $svc ..."
        sudo systemctl stop "$svc" 2>/dev/null
        sudo systemctl disable "$svc" 2>/dev/null
        sudo systemctl mask "$svc" 2>/dev/null
    else
        echo "$svc not found on this system. Skipping."
    fi
done
```

- Created a Script to list all enabled services and display their memory usage in KB

```
#!/bin/bash

# Print header
printf "%-40s %-10s\n" "SERVICE" "MEMORY(KB)"
printf "%-40s %-10s\n" "-----" "-----"

services=$(systemctl list-unit-files --state=enabled --type=service | awk '/\.service/ {print $1}')

for svc in $services;
do
    # Get main PID of the service
    PID=$(systemctl show -p MainPID --value "$svc")

    # Check if PID is valid and not zero
    if [[ "$PID" != "0" && -n "$PID" ]];
    then
        mem=$(top -b -n 1 -p "$PID" | grep -w "$PID" | awk '{print $6}')
        printf "%-40s %-10s\n" "$svc" "$mem KB"
    else
        printf "%-40s %-10s\n" "$svc" "N/A"
    fi
done | sort -k2 -n
```

- added an audit rule to monitor changes on:

1. /srv/www
2. /etc/passwd
3. /etc/group

- Created a Simple script to check disk space and Log incase it's higher than 80%
- and also added this script as cronjob that runs every hour (/etc/cron.hourly/)

magdy@Vm3:~/scripts — sudo v

```
## This file is automatically generated from
-D
-b 8192
-f 1
--backlog_wait_time 60000
-w /srv/www -p war -k www_watch
-w /etc/passwd -p wa -k passwd_watch
-w /etc/group -p wa -k group_watch
```

```
#!/usr/bin/env bash
```

```
LOGFILE=/var/log/deploy_alerts.log
```

```
USAGE=$(df / | tail -1 | awk '{print $5}' | tr -d '%')
```

```
if (( USAGE > 80 ));
```

```
then
```

```
    echo "$(date +%Y-%m-%dT%H:%M:%S') DISK WARNING: Usage at ${USAGE}% on /" >> "${LOGFILE}"
```

```
fi
```