Project 5: Jenkins Job Triggered by Access Log Size

Intern: Shyam Champakbhai Chotaliya **Company:** Fortune Cloud Technologies **Date:** [14-07-2025]

Project Overview

This project demonstrates an automated system for managing large application log files. A Jenkins job periodically checks the size of a specific log file (`/var/log/myapp/access.log`). If the file size exceeds a 1GB threshold, the Jenkins job triggers a shell script that compresses the log, uploads the archive to a secure Amazon S3 bucket, and then clears the original log file to free up disk space. This entire process is designed to run without manual intervention.

Tools & Services Used

- **EC2 Instance:** An AWS virtual server (Ubuntu 22.04 LTS) used to host the log file and the Jenkins server.
- **Jenkins:** An open-source automation server used to schedule and execute the log management job.
- **AWS CLI:** The command-line interface used within the script to upload the archived log file to S3.
- **Amazon S3:** A durable and cost-effective object storage service used to store the compressed log file archives for long-term access and analysis.
- **AWS IAM:** An Identity and Access Management role was attached to the EC2 instance to grant secure, temporary permissions to upload files to S3 without using static credentials.
- **Bash Scripting:** A shell script was created to contain all the logic for checking file size, compression, uploading, and log rotation.

Setup Instructions

1. Environment Setup

- An AWS EC2 instance was provisioned with Ubuntu 22.04.
- Required software was installed: `openjdk-21-jre`, `jenkins`, `awscli`, and `zip`.
- An IAM Role (`EC2-S3-Upload-Role`) with `AmazonS3FullAccess` permissions was created and attached to the EC2 instance. This allows the AWS CLI to interact with S3 securely.
- An S3 bucket was created to serve as the destination for log archives. All public access was blocked.

2. Jenkins Job Configuration

A new Freestyle project named `Log-File-Archiver` was created in Jenkins with the following configuration:

1. **Build Triggers:** The job was configured to run periodically using the "Build periodically" option with the schedule `H/5 * * * *`, which triggers a build approximately every 5 minutes.
2. **Build Steps:** An "Execute shell" build step was added to run the main script: `/bin/bash /home/ubuntu/check_log_and_upload.sh`.

3. Shell Script Logic

The `check_log_and_upload.sh` script performs the following steps:

1. **Check Size:** It gets the current size of `/var/log/myapp/access.log` and compares it against a 1GB limit.
2. **Compress:** If the size exceeds the limit, it creates a timestamped `.zip` archive of the log file in the `/tmp/` directory.
3. **Upload:** It uses the `aws s3 cp` command to upload the compressed archive to the designated S3 bucket.
4. **Rotate:** After a successful upload, it uses `sudo truncate -s 0` to clear the contents of the original log file, effectively rotating it.
5. **Cleanup:** It removes the temporary compressed file from `/tmp/`.

Testing Method

The system was validated through the following steps:

1. A large log file (>1.1GB) was created instantly using `sudo fallocate -l 1.1G /var/log/myapp/access.log`.
2. The Jenkins job was triggered manually using the "Build Now" button.
3. The "Console Output" of the Jenkins build was monitored to watch the script's execution logs in real-time.
4. The target S3 bucket was checked to confirm the successful upload of the timestamped `.zip` archive.
5. The original log file's size was checked on the EC2 instance using `ls -lh` to verify it had been truncated to 0 bytes.

Importance of This Automation

- **Prevents Disk Failure:** Automatically archiving and rotating logs prevents the server's disk from filling up, which can cause application failures.
- **Cost-Effective Archiving:** Compressing logs and moving them to a low-cost storage service like S3 is much cheaper than provisioning larger, more expensive server disks.
- **Ensures Reliability:** Automation removes the need for manual intervention, which is prone to human error and forgetfulness.
- **Maintains Performance:** Large log files can sometimes slow down application performance; keeping them lean helps maintain responsiveness.

List of Included Files

- **README.md:** This project report.

- **check_log_and_upload.sh:** The complete, commented shell script.
- **sample-iam-policy.json:** A sample IAM policy granting the necessary S3 upload permissions.
- **/screenshots/:** A folder containing screenshots of the Jenkins configuration, S3 bucket, and terminal output.

> ***Note:*** *This document covers only Project 5 out of the 6 assigned internship projects. The remaining project will be submitted separately.*