**FinOps CI/CD cost reduction project.**

**Problem statement:** A company with good observability setup with ELK stack, where Logstash is used for log analysis, application logs are used to save in log files along with it the Jenkins log files are also saved, but if the build fails it would be notified through the mails and slack hence there is no requirement to save the log files of Jenkins for error analysis, it will be saved only for the safety purposes, hence it would lead to lot of cost.

**Solution**: Storing the log files in S3 bucket of AWS which is much cheaper, where after 3 months if the files were not used it would be pushed to S3 Glacier where very low cost is imposed and if not required it would be deleted.

**Requirements:**

1. AWS EC2 instance upon where Jenkins is installed.
2. S3 bucket to store the log files.
3. AWS CLI is used to connect Jenkins along with S3 bucket.

**Step1: Create an EC2 instance**

A screenshot of a computer

AI-generated content may be incorrect.

sudo apt update

sudo apt install openjdk-17-jre

Allow the inbound traffic by altering the inbound traffic rules.

**Step 2:**

**Run the below commands to install Java and Jenkins**

Install Java

|  |
| --- |
| sudo apt update  sudo apt install openjdk-17-jre |

Verify Java is Installed

java -version

Now, you can proceed with installing Jenkins

|  |
| --- |
| curl -fsSL https://pkg.jenkins.io/debian/jenkins.io-2023.key | sudo tee \  /usr/share/keyrings/jenkins-keyring.asc > /dev/null  echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \  https://pkg.jenkins.io/debian binary/ | sudo tee \  /etc/apt/sources.list.d/jenkins.list > /dev/null  sudo apt-get update  sudo apt-get install jenkins |

After you login to Jenkins, - Run the command to copy the Jenkins Admin Password

|  |
| --- |
| sudo cat /var/lib/jenkins/secrets/initialAdminPassword - Enter the Administrator password |

A screenshot of a computer

AI-generated content may be incorrect.

**Step 3: Setting up AWS CLI**

A screenshot of a computer

AI-generated content may be incorrect.

A black background with white text

AI-generated content may be incorrect.

**Step 4: Create S3 bucket.**

A screenshot of a computer

AI-generated content may be incorrect.

**Step 5: Using AWS CLI write the shell scripting file for the function to carry out.**

|  |
| --- |
| #!/bin/bash  #Author: Dhruthi S L  #Description: uploading jenkins log files to s3 bucket  # Variables  JENKINS\_HOME="/var/lib/jenkins" # Replace with your Jenkins home directory  S3\_BUCKET="s3://jenkins-logfile-bucket" # Replace with your S3 bucket name  DATE=$(date +%Y-%m-%d) # Today's date  # Check if AWS CLI is installed  if ! command -v aws &> /dev/null; then  echo "AWS CLI is not installed. Please install it to proceed."  exit 1  fi  # Iterate through all job directories  for job\_dir in "$JENKINS\_HOME/jobs/"\*/; do  job\_name=$(basename "$job\_dir")  # Iterate through build directories for the job  for build\_dir in "$job\_dir/builds/"\*/; do  # Get build number and log file path  build\_number=$(basename "$build\_dir")  log\_file="$build\_dir/log"  # Check if log file exists and was created today  if [ -f "$log\_file" ] && [ "$(date -r "$log\_file" +%Y-%m-%d)" == "$DATE" ]; then  # Upload log file to S3 with the build number as the filename  aws s3 cp "$log\_file" "$S3\_BUCKET/$job\_name-$build\_number.log" --only-show-errors  if [ $? -eq 0 ]; then  echo "Uploaded: $job\_name/$build\_number to $S3\_BUCKET/$job\_name-$build\_number.log"  else  echo "Failed to upload: $job\_name/$build\_number"  fi  fi  done  done |

 **Purpose**: Automates uploading of Jenkins build log files to an S3 bucket using AWS CLI.

 **Variable Setup**:

* Sets JENKINS\_HOME as the Jenkins directory.
* Defines S3\_BUCKET for the target S3 path.
* Uses date +%Y-%m-%d to get the current date for filtering logs.

 **AWS CLI Check**:

* Uses command -v aws to ensure AWS CLI is installed before proceeding.

 **Directory Iteration**:

* Loops through all job directories in Jenkins.
* For each job, loops through its build directories.

 **Log File Filtering**:

* Checks if a log file exists using -f.
* Confirms it was modified today using date -r.

 **S3 Upload**:

* Uses aws s3 cp to upload the log file.
* Renames the file as <job\_name>-<build\_number>.log.
* Adds --only-show-errors to suppress standard AWS CLI output.

**Upload Verification**:

* Checks return code with $? to verify success.
* Prints success or failure messages accordingly.

 **Concepts Demonstrated**:

* Shell scripting fundamentals: variables, loops, conditionals.
* Command substitution and use of system utilities.
* Practical integration with AWS CLI for cloud automation.

**Step 6:**

**Create your job and build it in Jenkins.**

A screenshot of a computer

AI-generated content may be incorrect.

**Step 7: Execute the script and Jenkins log files will be uploaded to AWS S3 bucket.**

A computer screen shot of a black screen

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Conclusion:** This **reduces** the cost by **50%,** rather than storing all the log files of Jenkins in ELK which would be very costly.