St. JOSEPH'S GROUP OF INSTITUTIONS
OMR, CHENNAI - 119

aws

Placement Empowerment Program

Cloud Computing and DevOps Centre

Write a Shell Script to Monitor Logs:

Create a script that monitors server logs for errors and alerts you.

Name: Shana R S

Department : CSE

St. JOSEPH'S COLLEGE OF ENGINEERING | St. JOSEPH'S INSTITUTE OF TECHNOLOGY

AUTONOMOUS INSTITUTIONS, AFFILIATED TO ANNA UNIVERSITY

Introduction:

Monitoring server logs is a crucial task for maintaining the health and security of any system. A Shell script that monitors logs can help detect errors, failures, or unusual activity in real-time, sending alerts to administrators when necessary. This task will guide you in creating a script that scans server logs for specific error messages and notifies you when any issues are found. It's a useful automation technique for proactive system management, ensuring quick responses to potential problems.

Why Is This Task Important?

- **Detects Issues Early**: Helps identify errors or failures quickly to minimize downtime.
- **Saves Time**: Automates log checking, so you don't have to do it manually.
- **Quick Alerts**: Sends instant notifications when problems arise, enabling faster fixes.
- **Improves Security**: Helps detect suspicious activities and potential breaches.

Steps Followed

- **Create Log File**:
  Use Command Prompt to create a log file that will store messages for monitoring.
- **Create PowerShell Script**:
  Write a PowerShell script that monitors the log file for error messages and sends an alert when detected.
- **Run the Script**:
  Open PowerShell as an administrator and run the script to start monitoring the log.
- **Test**:
  Add an error message to the log file and check if the script detects it and sends an alert.

Where Can This Be Applied?

- **Server Monitoring**: Detect errors in server logs to ensure system uptime.
- **Application Logging**: Monitor application logs for issues and notify developers.
- **Security Monitoring**: Detect potential security breaches or suspicious activities from log files.
- **System Administration**: Automate log monitoring to reduce manual checks and improve response time for errors.

Step-by-Step Overview

Step1: Create a Log File

☑ **Instruction:**

- Open **Command Prompt (**cmd**)** by pressing Win + R, typing cmd, and hitting **Enter**.
- Type the following command and press **Enter**:

```
echo Log Monitoring Started > "C:\Users\SmilE\OneDrive\Desktop\logfile.log"
```

This creates a log file to store log messages, which will be monitored for errors.

Step 2 : Create a PowerShell Script

☑ **Instruction:**

- Open **Notepad** or any text editor.
- Copy and paste the following PowerShell script:
- Click **File > Save As**.
- Set **"Save as type"** to **"All Files"** and save the file as log_monitor.ps1 inside **C:\Users\SmilE\OneDrive\Desktop**.

logfile.log | log_monitor.ps1

File Edit View

```
$LogFile = "C:\Users\SmilE\OneDrive\Desktop\logfile.log"  # Path to the log file
$Keyword = "ERROR"  # The word to monitor
$AlertSound = "$env:windir\Media\notify.wav"  # Windows notification sound

Function Send-Alert {
    Write-Host ""
    Write-Host "🚨🚨🚨 ALERT: Error detected in logs! 🚨🚨🚨" -ForegroundColor Red
    Write-Host "----------------------------------------"
    [console]::beep(1000, 500)  # Beep sound alert
    if (Test-Path $AlertSound) {
        (New-Object Media.SoundPlayer $AlertSound).Play()
    }
}

# Monitor the log file in real-time
Get-Content -Path $LogFile -Wait | ForEach-Object {
    If ($_ -match $Keyword) {
        Send-Alert
        Write-Host $_ -ForegroundColor Yellow
    }
}
```

This script continuously monitors the log file for errors and provides an alert when an issue is detected.

Step 3 : Run the PowerShell Script

☑ Instruction:

- Open **PowerShell as Administrator** by pressing Win + S, typing **PowerShell**, right-clicking it, and selecting **Run as Administrator**.
- Navigate to the script location using the command:
  cd "C:\Users\SmilE\OneDrive\Desktop"
- Set the execution policy to allow script execution:
  Set-ExecutionPolicy Unrestricted -Scope Process
- If prompted, **type A** (Yes to All) and press **Enter**.
- Run the script using:
  .\log_monitor.ps1

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Windows\system32> cd "C:\Users\SmilE\OneDrive\Desktop"
>> Set-ExecutionPolicy Unrestricted -Scope Process
>> .\log_monitor.ps1
>> C:\Users\SmilE\OneDrive\Desktop\logfile.log

Execution Policy Change
The execution policy helps protect you from scripts that you do not trust. Changing the execution policy might expose
you to the security risks described in the about_Execution_Policies help topic at
https:/go.microsoft.com/fwlink/?LinkID=135170. Do you want to change the execution policy?
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is "N"): A
```

PowerShell has security restrictions, so we must enable script execution before running the script.

Step 4 : Test the Alert System

☑ **Instruction:**

- Open **Command Prompt (cmd)**.
- Add an error message to the log file:
  echo ERROR: Something went wrong >> "C:\Users\SmilE\OneDrive\Desktop\logfile.log"
- Go back to **PowerShell** where the script is running.

- If everything works correctly, PowerShell will display:

Outcome

By completing this POC, you will:

- Set up a system to monitor log files for errors.
- Automate error detection and alerting.
- Gain experience with PowerShell scripting and log management.
- Improve system monitoring and response time.