 

**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: Nanthini R Department: ADS



**Introduction**

In any production environment, it's essential to continuously monitor server logs to ensure that everything is running smoothly and to quickly identify any potential issues. Logs provide critical information regarding system health, performance, and errors that need immediate attention. Server administrators often rely on log monitoring to spot errors such as application failures, security threats, and system misconfigurations.

One effective way to automate this task is through PowerShell scripting on Windows servers. PowerShell allows for flexible scripting and automation, making it an ideal choice for creating a log monitoring solution. By using PowerShell, you can continuously monitor server logs for specific error messages and automatically send alerts when an issue is detected.

In this guide, we will walk through the steps to create a PowerShell script that monitors server logs, searches for errors, and sends notifications to the system administrator. This approach provides a simple yet powerful method to automate the process of log file monitoring, reducing the need for manual checks and enabling quicker responses to any critical issues.

**Overview**

A shell script can be used to monitor server logs for errors and send an alert when certain conditions are met (such as specific error messages or log entries). On Windows, you typically use PowerShell or Batch scripting for such tasks since Shell scripting in Windows is not the same as in Unix-like systems.

This script will continuously monitor log files for specific error messages and send notifications (e.g., through email or Windows notifications) when errors are found. It will be useful for administrators who need to quickly identify and act upon issues in system logs.

**Objective**

Continuously monitor specific log files (e.g., application, system, or security logs).

Identify error messages or specific patterns (e.g., ERROR, CRITICAL, FAIL).

Alert the user if any errors are detected.

Run the script periodically or continuously in the background

**Step-by-Step Procedure**

1. **Open PowerShell**

Start by opening PowerShell as an administrator to create and execute the script**.**

1. **Create the PowerShell Script**

Create a script named monitor\_logs.ps1. You can use any text editor to create this script.

# monitor\_logs.ps1

$logFilePath = "C:\path\to\your\logfile.log" # Path to the log file to monitor

$errorKeyword = "ERROR"

# Keyword to search for in the logs

$emailRecipient = "your-email@example.com" # Email to send alerts

$logCheckInterval = 60

# Interval in seconds to check the log

# Function to send email notification (Make sure to configure SMTP server settings)

function Send-EmailNotification($message) {

$smtpServer = "smtp.example.com"

# Replace with your SMTP server

$smtpFrom = "your-email@example.com"

$smtpTo = $emailRecipient

$subject = "Log Monitoring Alert: Error Found"

$body = $message

$smtp = New-Object Net.Mail.SmtpClient($smtpServer)

$smtp.Send($smtpFrom, $smtpTo, $subject, $body)

}

# Function to monitor the log file

function Monitor-LogFile {

while ($true) {

$logContent = Get-Content -Path $logFilePath -Tail 10

if ($logContent -match $errorKeyword) {

$message = "Error found in log file at $(Get-Date): $logContent"

Write-Host "Error found! Sending email..."

Send-EmailNotification $message

}

Start-Sleep -Seconds $logCheckInterval # Wait for the specified interval before checking again

}

}

# Start monitoring the log file

Monitor-LogFile

1. **Modify the Script Parameters**

Change the $logFilePath to the actual path of the log file you want to monitor.

Set the $errorKeyword to the keyword that represents errors in the log (e.g., ERROR, CRITICAL).

Update $emailRecipient with your email address to receive alerts.

Adjust $logCheckInterval to the desired number of seconds between log checks.

1. **Save the Script**

Save the script as monitor\_logs.ps1.

1. **Run the Script**

Run the script using PowerShell:

.\monitor\_logs.ps1

This will start the log monitoring process. The script will check the log file every 60 seconds and send an email if an error is found.

1. **Configure SMTP Server**

For email notifications to work, ensure you have an SMTP server configured correctly (e.g., Gmail, Outlook, or your own SMTP server).

Modify the Send-EmailNotification function with your SMTP server and authentication details.

1. **Set the Script to Run Continuously**

If you want the script to run continuously, you can create a Scheduled Task in Windows to run this script on startup or at regular intervals**.**

1. **Testing and Validation**

Test the script by adding an error message (e.g., ERROR) in the monitored log file.

Check if the script detects the error and sends an alert via email.

**Conclusion**

The **PowerShell script** created above is a simple but effective way to monitor server logs for specific error patterns and send alerts. By setting up a scheduled task, you can ensure that the script runs continuously in the background, providing real-time monitoring for any issues within the log files.

This approach can be adapted to any log file and tailored for specific error detection. It's useful for server administrators, developers, or anyone needing automated monitoring of logs on a Windows server.

Remember that you can extend this script by:

Adding more complex filtering and pattern matching.

Including additional notification methods, such as Windows toast notifications or writing alerts to a different log file.