Advanced Bash Script for Log Management, Backup, Monitoring, Automation, and Alerting in a Linux Server Environment

bash

CopyEdit

#!/bin/bash

```
################################################################################
##

# Title   : Advanced Bash Script for System Monitoring, Log Rotation, Backups,
#           User Management, and Email Alerting in Production Linux Environments
# Author  : Maria Sultana
# Date    : 2025-06-20
# Version : 5.0
# Language: Bash (Bourne Again Shell)
# Keywords: bash script for automation, bash for log rotation, bash system monitoring,
#           cron automation script, linux bash examples, devops bash tutorial,
#           bash script for server backup, bash email alert
################################################################################
##


# ==============================
# Section 1: Environment Setup
# ==============================
```

```
set -euo pipefail
IFS=$'\n\t'

# Global Configuration Variables
LOG_DIR="/var/log/custom_logs"
ARCHIVE_DIR="/var/backups/custom_logs"
ALERT_EMAIL="admin@example.com"
USERS_TO_MONITOR=("apache" "mysql" "backupuser")
DISK_USAGE_THRESHOLD=80
CPU_LOAD_THRESHOLD=5.0
TODAY=$(date +'%Y-%m-%d')
SCRIPT_LOG="/var/log/sysadmin_tool_${TODAY}.log"
CRON_SCHEDULE="0 */6 * * *"
BACKUP_DESTINATION="/mnt/nas_backups/$(hostname)"

mkdir -p "$LOG_DIR" "$ARCHIVE_DIR"

# ==============================
# Section 2: Logging Functions
# ==============================

log_info() {
  echo "$(date +'%F %T') [INFO] $1" | tee -a "$SCRIPT_LOG"
}
```

```
log_error() {
  echo "$(date +'%F %T') [ERROR] $1" | tee -a "$SCRIPT_LOG" >&2
}

# ==============================
# Section 3: System Monitoring
# ==============================

monitor_disk_usage() {
  log_info "Checking disk usage..."
  local usage=$(df / | grep / | awk '{ print $5 }' | sed 's/%//g')
  if [ "$usage" -ge "$DISK_USAGE_THRESHOLD" ]; then
    log_error "Disk usage critical: ${usage}% used."
    echo "Subject: CRITICAL - Disk Usage Alert on $(hostname)
Disk usage is at ${usage}% on $(hostname)." | sendmail "$ALERT_EMAIL"
  else
    log_info "Disk usage OK: ${usage}%"
  fi
}

monitor_cpu_load() {
  log_info "Checking CPU load..."
  local load=$(uptime | awk -F'load average:' '{ print $2 }' | cut -d',' -f1 | sed 's/ //g')
  load="${load%%.*}.${load#*.}" # Normalize to float
```

```
  if (( $(echo "$load > $CPU_LOAD_THRESHOLD" | bc -l) )); then
    log_error "High CPU load: $load"
    echo "Subject: WARNING - High CPU Load on $(hostname)
CPU Load: $load" | sendmail "$ALERT_EMAIL"
  else
    log_info "CPU load is normal: $load"
  fi
}

# ==============================
# Section 4: Log Rotation and Archiving
# ==============================

rotate_logs() {
  log_info "Rotating logs in $LOG_DIR..."
  for file in "$LOG_DIR"/*.log; do
    [ -e "$file" ] || continue
    gzip -c "$file" > "$ARCHIVE_DIR/$(basename "$file").gz"
    : > "$file"
    log_info "Rotated and compressed: $file"
  done
}

# ==============================
# Section 5: Automated Backups
```

```
# ==============================

backup_directories() {
  log_info "Starting backup process..."
  tar -czf "$BACKUP_DESTINATION/backup_$TODAY.tar.gz" /etc /var/www /home 2>>"$SCRIPT_LOG"
  log_info "Backup completed: backup_$TODAY.tar.gz"
}

# ==============================
# Section 6: User Activity Auditing
# ==============================

audit_user_activity() {
  log_info "Auditing user activity..."
  for user in "${USERS_TO_MONITOR[@]}"; do
    if id "$user" &>/dev/null; then
      lastlog -u "$user" >> "$SCRIPT_LOG"
      log_info "Logged activity for user: $user"
    else
      log_error "User not found: $user"
    fi
  done
}
```

```
# ==============================
# Section 7: Cron Job Scheduler
# ==============================

setup_cron_job() {
  log_info "Checking if cron job is installed..."
  (crontab -l 2>/dev/null | grep -F "$0") && {
    log_info "Cron job already installed."
  } || {
    (crontab -l 2>/dev/null; echo "$CRON_SCHEDULE $0") | crontab -
    log_info "Cron job installed to run every 6 hours."
  }
}

# ==============================
# Section 8: Interactive Menu (Optional CLI Interface)
# ==============================

show_menu() {
  echo "--------------------------------------------"
  echo "        Linux Admin Bash Automation Tool     "
  echo "--------------------------------------------"
  echo "1. Monitor Disk Usage"
  echo "2. Monitor CPU Load"
  echo "3. Rotate Logs"
```

```
  echo "4. Run Backup"
  echo "5. Audit Users"
  echo "6. Setup Cron Job"
  echo "7. Run All"
  echo "0. Exit"
  echo "-----------------------------------------"
  read -rp "Choose an option: " choice

  case "$choice" in
    1) monitor_disk_usage ;;
    2) monitor_cpu_load ;;
    3) rotate_logs ;;
    4) backup_directories ;;
    5) audit_user_activity ;;
    6) setup_cron_job ;;
    7) monitor_disk_usage; monitor_cpu_load; rotate_logs; backup_directories;
audit_user_activity; setup_cron_job ;;
    0) exit 0 ;;
    *) echo "Invalid option" ;;
  esac
}

# ==============================
# Section 9: Execution Entrypoint
# ==============================
```

```
if [[ "${BASH_SOURCE[0]}" == "${0}" ]]; then
  show_menu
fi
```

Highlights & Keywords Covered for SEO & Technical Screening

- bash script for server monitoring
- automated log rotation script linux
- disk usage check bash script
- cpu load monitoring bash
- advanced bash scripting for system administrators
- backup linux files bash script
- audit linux users via shell
- bash automation tutorial for DevOps
- cron job setup via bash
- send email alerts using bash