Automation Scripts Documentation

Overview

This documentation outlines automation tasks using both PowerShell (for Windows) and Bash (for Linux). The scripts automate:

1. Monthly Patch Updates
2. Automated Backups
3. Automated Monitoring and Reporting
4. User Creation and Assignment to Groups

The scripts are designed to reduce the administrative overhead for IT departments by scheduling essential tasks.

How Automation Helps:

- **Consistency:** Reduces human error by ensuring tasks like updates and backups are handled consistently.
- **Time-saving:** Frees up IT staff by automating repetitive tasks.
- **Security:** Ensures regular updates and patches are applied, reducing security vulnerabilities.
- **Reliability:** Scheduled backups ensure critical data is regularly saved, minimizing the risk of data loss.

Automation for Red Hat Enterprise Linux (RHEL)

Creating a Group

```
sudo groupadd IT_Admins
```

* Creating Users and Adding Them to a Group

To create 3 users and add them to the `IT_Admins` group:

```
# create a list of users names

users=("Esraa" "Omnia" "Abdullah")

#create user passwords and assign them to IT_Admin group
```

```
for user in "${users[@]}"; do

    sudo useradd -m -G IT_Admins "$user"

    echo "$user:P@ssw0rd123" | sudo chpasswd  #change password
in first login

done
```

Make it Executable before running it:

```
root@rhel:/home
[root@rhel Documents]# vi usercreation.sh
[root@rhel Documents]# chmod a+x usercreation.sh
[root@rhel Documents]# ./usercreation.sh
[root@rhel Documents]# cd /home
[root@rhel home]# ls
Abdullah  emubarak  Esraa  Omnia
[root@rhel home]#
```

* Backup Automation

Purpose:

This script automates daily backups of user files and directories to a specified destination.

```
# Schedule it in cron (edit cron jobs)

crontab -e

# Add this line to schedule the backup every day at 2 AM

0 2 * * * /var/backups/backup.sh

#!/bin/bash

#Source Directory

SOURCE_DIR="/home/"

#Destination Directory
```

```
DEST_DIR="/var/backups/"

#Backup File

BACKUP_FILE="$DEST_DIR/home_backup_$(date
+%Y%m%d_%H%M%S).tar.gz"

#print a massege when starting the backup

echo "Starting backup from $SOURCE_DIR to $BACKUP_FILE"

# create backup using tar

tar -czvf "$BACKUP_FILE" "$SOURCE_DIR"

# check if the backup is success or not

if [ $? -eq 0 ]; then

  echo "Backup completed successfully!"

else

  echo "Backup failed!"

Fi
```

***Testing the script by running it:**

Make it Executable before running it:

```
Chmod +x Automation-Backup.sh
```

root@rhel:~/Documents

```
[root@rhel Documents]# ./Automation-Backup.sh
rsync: [sender] change_dir "/source" failed: No such file or directory (2)
rsync error: some files/attrs were not transferred (see previous errors) (code 23) at main.c(1330) [sender
=3.2.3]
chmod: cannot access '/path/to/backup.sh': No such file or directory
crontab: installing new crontab
./Automation-Backup.sh: line 12: 0: command not found
Starting backup from /home/ to /var/backups//home_backup_20240929_204959.tar.gz
tar: Removing leading `/' from member names
/home/
/home/emubarak/
/home/emubarak/.mozilla/
/home/emubarak/.mozilla/extensions/
/home/emubarak/.mozilla/plugins/
/home/emubarak/.bash_logout
/home/emubarak/.bash_profile
/home/emubarak/.bashrc
/home/emubarak/.local/
/home/emubarak/.local/share/
/home/emubarak/.local/share/keyrings/
/home/emubarak/.local/share/keyrings/login.keyring
/home/emubarak/.local/share/keyrings/user.keystore
/home/emubarak/.local/share/gnome-shell/
/home/emubarak/.local/share/gnome-shell/gnome-overrides-migrated
/home/emubarak/.local/share/gnome-shell/application_state
/home/emubarak/.local/share/evolution/
/home/emubarak/.local/share/evolution/addressbook/
/home/emubarak/.local/share/evolution/addressbook/trash/
/home/emubarak/.local/share/evolution/addressbook/system/
/home/emubarak/.local/share/evolution/addressbook/system/contacts.db
/home/emubarak/.local/share/evolution/addressbook/system/photos/
/home/emubarak/.local/share/evolution/calendar/
```

```
root@rhel:~/Documents
/home/emubarak/.cache/gnome-software/odrs/
/home/emubarak/.cache/gnome-software/odrs/ratings.json
/home/emubarak/.cache/gstreamer-1.0/
/home/emubarak/.cache/gstreamer-1.0/registry.x86_64.bin
/home/emubarak/.cache/flatpak/
/home/emubarak/.cache/flatpak/system-cache/
/home/emubarak/.cache/appstream/
/home/emubarak/.bash_history
/home/.userscration.sh.swp
/home/Esraa/
/home/Esraa/.mozilla/
/home/Esraa/.mozilla/extensions/
/home/Esraa/.mozilla/plugins/
/home/Esraa/.bash_logout
/home/Esraa/.bash_profile
/home/Esraa/.bashrc
/home/Omnia/
/home/Omnia/.mozilla/
/home/Omnia/.mozilla/extensions/
/home/Omnia/.mozilla/plugins/
/home/Omnia/.bash_logout
/home/Omnia/.bash_profile
/home/Omnia/.bashrc
/home/Abdullah/
/home/Abdullah/.mozilla/
/home/Abdullah/.mozilla/extensions/
/home/Abdullah/.mozilla/plugins/
/home/Abdullah/.bash_logout
/home/Abdullah/.bash_profile
/home/Abdullah/.bashrc
Backup completed successfully!
[root@rhel Documents]#
```

* Bash Script for Monthly Updates

The script will:

- Check for available updates.
- Install the updates.
- Reboot the system if necessary.

Purpose:

This script performs system updates and installs security patches monthly on a Red Hat Enterprise Linux system.

```
#!/bin/bash

# Script for updating Red Hat Enterprise Linux (RHEL) and
rebooting if necessary

# Log file location
```

```
LOG_FILE="/var/log/system_update.log"

# Print the start time of the update

echo "Starting system update at $(date)" | tee -a "$LOG_FILE"

# Update the package list and install available updates using
dnf

echo "Checking for updates using DNF package manager..." | tee
-a "$LOG_FILE"

sudo dnf check-update | tee -a "$LOG_FILE"

sudo dnf update -y | tee -a "$LOG_FILE"

# Check if a reboot is required (for RHEL, there's no specific
reboot-required file, so we assume updates could require a
reboot)

if [[ $? -eq 0 ]]; then

    echo "System update completed successfully." | tee -a
"$LOG_FILE"

else

    echo "Update failed. Please check the log for details." |
tee -a "$LOG_FILE"

    exit 1

fi

# Optionally force a reboot if updates require it (RHEL does not
have a reboot-required file like Ubuntu)

# We will reboot in case of any kernel updates
```

```
KERNEL_UPDATED=$(rpm -q --last kernel | head -n 1 | grep
"$(uname -r)")

if [ -z "$KERNEL_UPDATED" ]; then

    echo "Kernel updated. Rebooting system in 1 minute..." | tee
-a "$LOG_FILE"

    sudo shutdown -r +1

else

    echo "No reboot required." | tee -a "$LOG_FILE"

fi

# Log the completion of the update

echo "System update process finished at $(date)" | tee -a
"$LOG_FILE"
```

* Monitoring and Reporting in RHEL (Bash)

Purpose:

This script monitors CPU usage, memory usage, and disk space on a RHEL system and generates a report.

```
#!/bin/bash

# Define the report file path

report_path="/var/reports/system_report.txt"

# Get system statistics

cpu_usage=$(top -bn1 | grep "Cpu(s)" | sed "s/.*, *\([0-9.]*\)%*
id.*/\1/" | awk '{print 100 - $1}')

memory_info=$(free -m)
```

```
used_memory=$(echo "$memory_info" | awk 'NR==2{printf "%.2f",
$3}')

total_memory=$(echo "$memory_info" | awk 'NR==2{printf "%.2f",
$2}')

free_memory=$(echo "$memory_info" | awk 'NR==2{printf "%.2f",
$4}')

disk_space=$(df -h / | awk 'NR==2{printf "%s", $4}')

# Create report content

report_content="System Report - $(date)\n

CPU Usage: $cpu_usage%\n

Total Memory: $total_memory MB\n

Used Memory: $used_memory MB\n

Free Memory: $free_memory MB\n

Free Disk Space: $disk_space\n"

# Write report to file

echo -e "$report_content" > $report_path
```

##

* Automation for Windows Server

* PowerShell Script for Creating Users and Assigning Them to Groups

This script will:

- Create multiple users in Active Directory.
- Assign them to a specific group.

Script to Create Users and Assign Them to Groups:

```
# Define the OU where users will be created
$OU = "OU=Development-Team,DC=techwave,DC=local"
# Define the group to which users will be assigned
$group = "Developers"

#Make sure if the group is already excist
if (-not (Get-ADGroup -Filter { Name -eq $group })) {
    # If Not! create a new one
    New-ADGroup -Name $group -GroupScope Global -Path $OU
-Description "Group for developers Team"
    Write-Host "Group $group created."
} else {
    Write-Host "Group $group already exists."
}

# Define the users with names
$users = @(
    @{Name = "Samy Ahmed"; Password = "P@ssw0rd123"},
    @{Name = "Peter Mark"; Password = "P@ssw0rd123"},
    @{Name = "Mariam Saeed"; Password = "P@ssw0rd123"}
)

# Loop through each user and create them in AD, then add to the
group
foreach ($user in $users) {
    $username = $user.Name
    $password = ConvertTo-SecureString $user.Password
-AsPlainText -Force

    # Create user in Active Directory
    New-ADUser -Name $username -GivenName $username
-SamAccountName $username -UserPrincipalName
"$username@techwave.local" `
```

```
                -Path $OU -AccountPassword $password -Enabled
$true

    # Add user to group
    Add-ADGroupMember -Identity $group -Members $username

    Write-Host "User $username created and added to $group
group."
}
Write-Host "All users created and assigned to the group."
```

As we see the users are created successfully

*** Monthly Patch Updates script to run at a specific date and time every month:**

Monthly Patch Updates

PowerShell Script (for Windows)

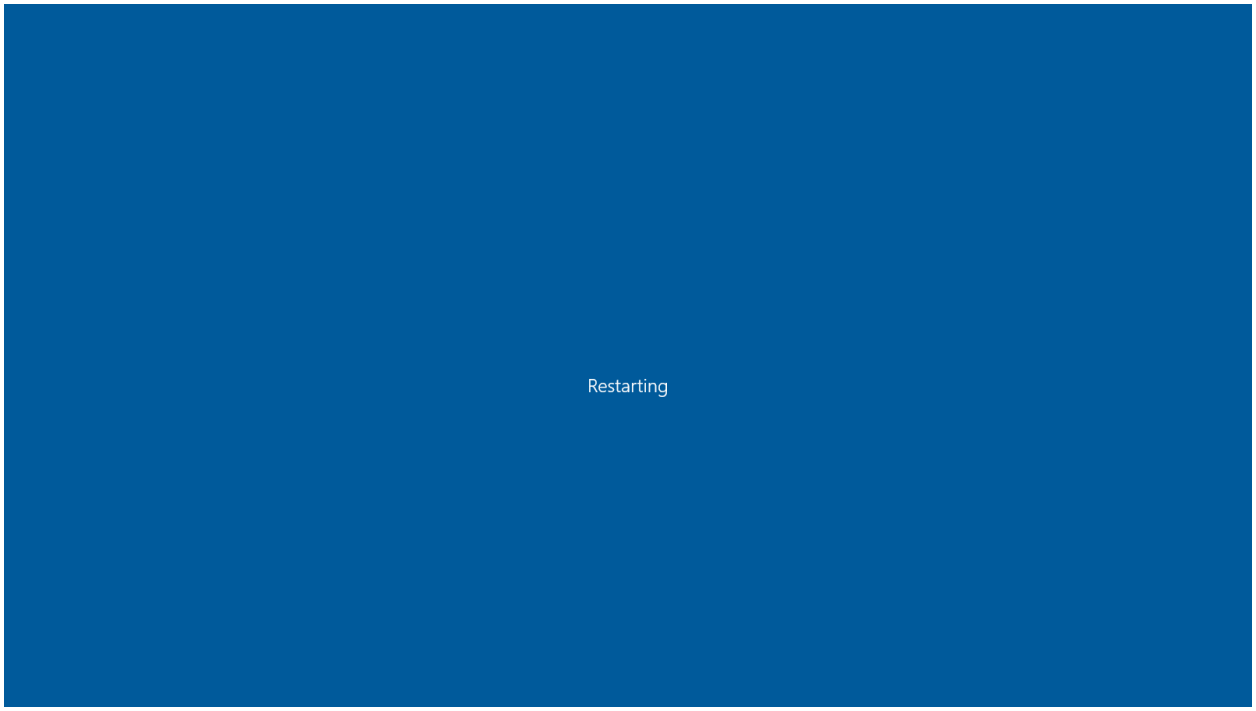
Purpose:

Automatically install Windows system updates monthly, ensuring the system stays secure and up-to-date.

```
Write-Host "Starting the script..."

# Check if PSWindowsUpdate module is installed

if (-not (Get-Module -ListAvailable -Name PSWindowsUpdate)) {
```

```
    Write-Host "PSWindowsUpdate module is not installed. Installing it now..."

    Install-Module -Name PSWindowsUpdate -Force -AllowClobber

    Write-Host "PSWindowsUpdate module installed successfully."

} else {

    Write-Host "PSWindowsUpdate module is already installed."

}

# Import the module

Import-Module PSWindowsUpdate

Write-Host "PSWindowsUpdate module imported."

# Start checking for updates

Write-Output "Checking for updates..."

Install-WindowsUpdate -AcceptAll -AutoReboot

Write-Host "Updates installed. Rebooting if necessary."

# Define the log file path

$logPath = "C:\Logs\MonthlyPatchUpdate.log"

# Write logs

Write-Output "Updates installed successfully." | Out-File $logPath -Append

Write-Output "Monthly patch updates applied successfully on $(Get-Date)." | Out-File $logPath -Append

Write-Host "Script completed successfully."
```

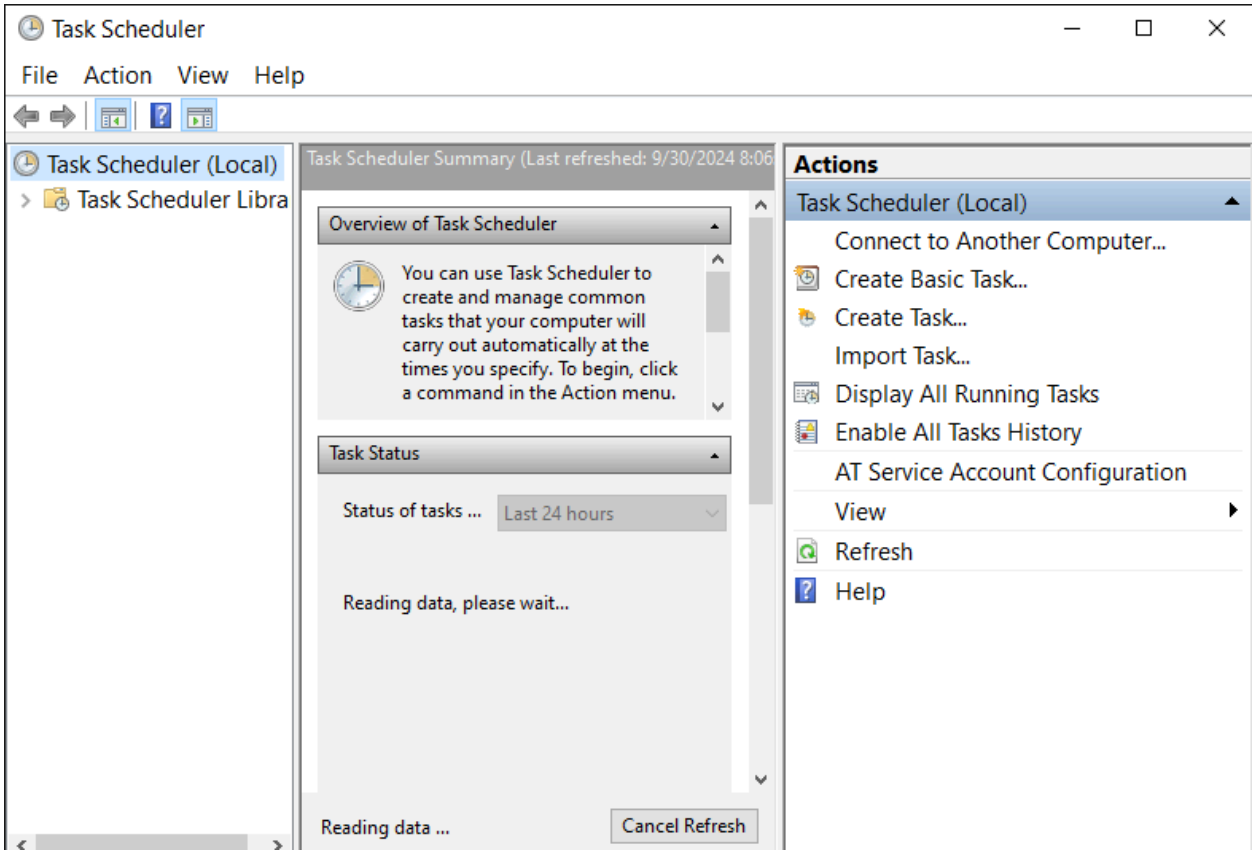
```
Install updates for OMNIAPC
   [3/4] 2024-10 Cumulative Update for Microsoft server operating system version 21H2 for x64-based Systems (KB5044281)
    25GB
   [oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo                          ]

NuGet provider is required to continue
PowerShellGet requires NuGet provider version '2.8.5.201' or newer to interact with NuGet-based repositories. The NuGet
 provider must be available in 'C:\Program Files\PackageManagement\ProviderAssemblies' or
'C:\Users\Administrator.TECHWAVE\AppData\Local\PackageManagement\ProviderAssemblies'. You can also install the NuGet
provider by running 'Install-PackageProvider -Name NuGet -MinimumVersion 2.8.5.201 -Force'. Do you want PowerShellGet
to install and import the NuGet provider now?
[Y] Yes  [N] No  [S] Suspend  [?] Help (default is "Y"): Y
PSWindowsUpdate module installed successfully.
PSWindowsUpdate module imported.
Checking for updates...

X ComputerName Result     KB          Size Title
- ------------ ------     --          ---- -----
1 OMNIAPC      Accepted   KB890830    73MB Windows Malicious Software Removal Tool x64 - v5.129 (KB890830)
1 OMNIAPC      Accepted   KB5044099   72MB 2024-10 Cumulative Update for .NET Framework 3.5, 4.8 and 4.8.1 for Micro...
1 OMNIAPC      Accepted   KB2267602    1GB Security Intelligence Update for Microsoft Defender Antivirus - KB2267602...
1 OMNIAPC      Accepted   KB5044281   25GB 2024-10 Cumulative Update for Microsoft server operating system version 2...
2 OMNIAPC      Downloaded KB890830    73MB Windows Malicious Software Removal Tool x64 - v5.129 (KB890830)
2 OMNIAPC      Downloaded KB5044099   72MB 2024-10 Cumulative Update for .NET Framework 3.5, 4.8 and 4.8.1 for Micro...
2 OMNIAPC      Downloaded KB2267602    1GB Security Intelligence Update for Microsoft Defender Antivirus - KB2267602...
2 OMNIAPC      Downloaded KB5044281   25GB 2024-10 Cumulative Update for Microsoft server operating system version 2...
3 OMNIAPC      Installed  KB890830    73MB Windows Malicious Software Removal Tool x64 - v5.129 (KB890830)
3 OMNIAPC      Installed  KB5044099   72MB 2024-10 Cumulative Update for .NET Framework 3.5, 4.8 and 4.8.1 for Micro...
3 OMNIAPC      Installed  KB2267602    1GB Security Intelligence Update for Microsoft Defender Antivirus - KB2267602...
```

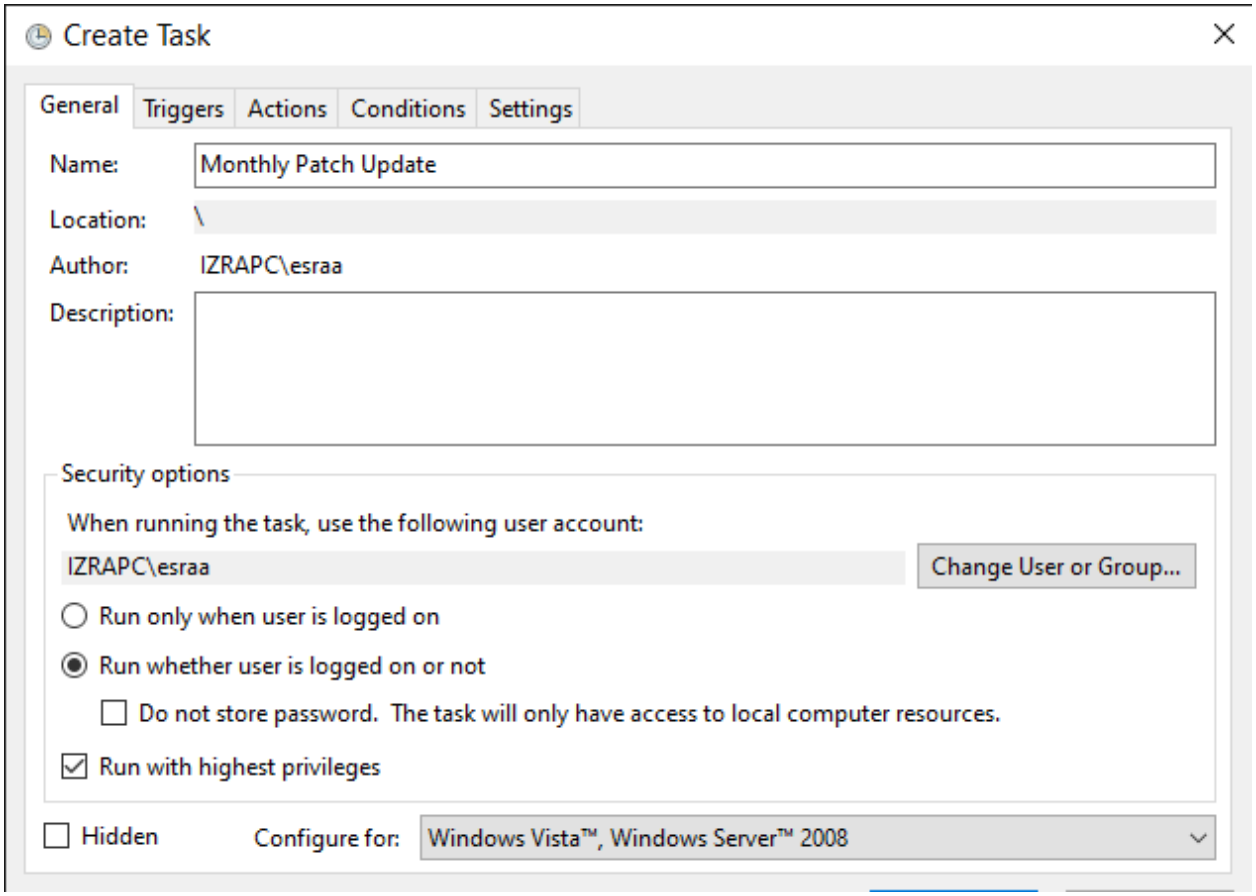


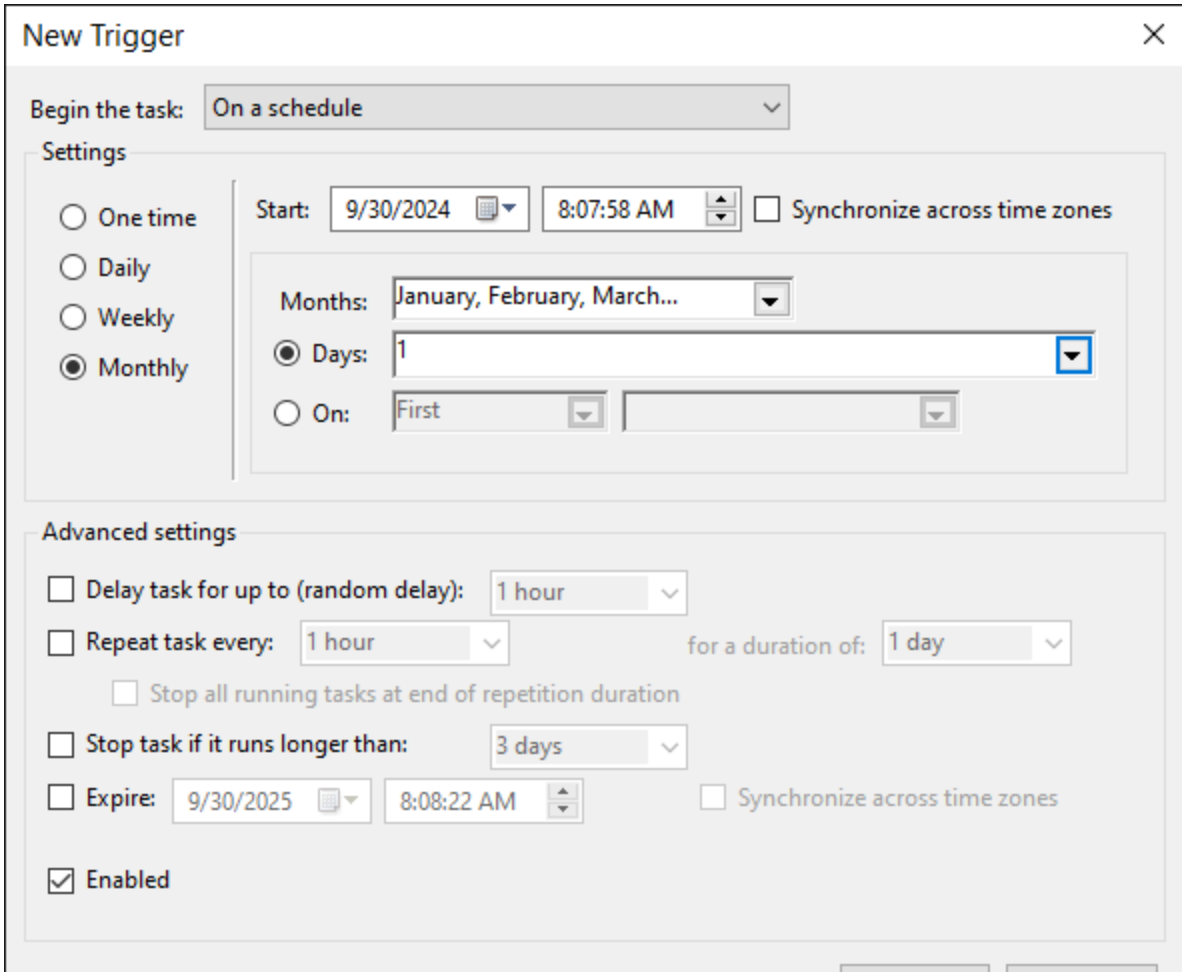
To set Automatic Monthly Patch Updates to Schedule PowerShell Script Using Task Scheduler, You must follow the instructions below:

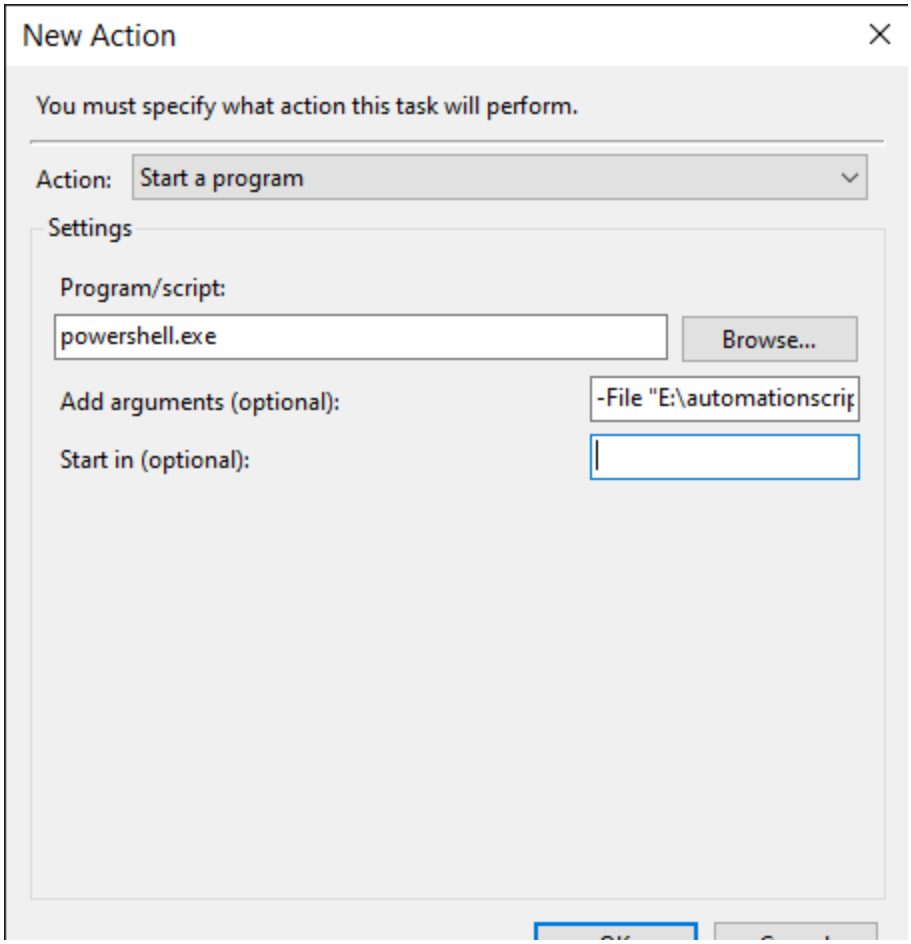
1. Open Task Scheduler:

    - On your Windows Server, press `Windows + R`, type `taskschd.msc`.
2. Create a New Task:
    - In the Actions pane, click Create Task.
3. General Tab:
    - Name the task something like "Monthly Patch Updates".
    - Set it to run with the highest privileges (check Run with highest privileges).
    - Choose Run whether user is logged on or not.
4. Triggers Tab:
    - Click New to create a trigger.
    - Set the trigger to Monthly.
    - Choose the Day of the month you want the script to run, e.g., Day 1.
    - Set the time, e.g., 3:00 AM (so it runs during off-peak hours).
    - Ensure the trigger is set to Enabled.
5. Actions Tab:
    - Click New.
    - In the Action drop-down, select Start a program.
6. Conditions Tab:
    - Optionally, uncheck the Start the task only if the computer is on AC power if you want to ensure the script runs even on battery power.
7. Settings Tab:
    - Check Allow task to be run on demand.
    - Check If the task fails, restart every and specify a retry interval and maximum retry count (e.g., restart every 5 minutes, up to 3 times).
8. Save the Task:
    - Click OK, and you'll be prompted to enter your password (use the account that has privileges to run the script).


Task Scheduler
File
Action
View
Help
Task Scheduler (Local)
Task Scheduler Libra
Task Scheduler Summary (Last refreshed: 9/30/2024 8:06
Overview of Task Scheduler
You can use Task Scheduler to create and manage common tasks that your computer will carry out automatically at the times you specify. To begin, click a command in the Action menu.
Task Status
Status of tasks ...
Last 24 hours
Reading data, please wait...
Reading data ...
Cancel Refresh
Actions
Task Scheduler (Local)
Connect to Another Computer...
Create Basic Task...
Create Task...
Import Task...
Display All Running Tasks
Enable All Tasks History
AT Service Account Configuration
View
Refresh
Help


Create Task
General
Triggers
Actions
Conditions
Settings
Name:
Monthly Patch Update
Location:
\
Author:
IZRAPC\esraa
Description:
Security options
When running the task, use the following user account:
IZRAPC\esraa
Change User or Group...
Run only when user is logged on
Run whether user is logged on or not
Do not store password. The task will only have access to local computer resources.
Run with highest privileges
Hidden
Configure for:
Windows Vista™, Windows Server™ 2008


New Trigger
Begin the task:
On a schedule
Settings
One time
Daily
Weekly
Monthly
Start:
9/30/2024
8:07:58 AM
Synchronize across time zones
Months:
January, February, March...
Days:
1
On:
First
Advanced settings
Delay task for up to (random delay):
1 hour
Repeat task every:
1 hour
for a duration of:
1 day
Stop all running tasks at end of repetition duration
Stop task if it runs longer than:
3 days
Expire:
9/30/2025
8:08:22 AM
Synchronize across time zones
Enabled



*** Directory Backup**

The following PowerShell script will back up a specified folder (e.g., user files) to a backup directory.

```
# Define the source and destination paths

$source = "C:\Users\Esraa\Documents"

$destination = "E:\Documents\Backups"

# Create destination folder if it doesn't exist

if (!(Test-Path -Path $destination)) {

    New-Item -ItemType Directory -Path $destination
```

```
}

# Perform the backup using Robocopy with options

Robocopy $source $destination /MIR /Z /R:3 /W:10

# Log the result

$logFile = "D:\Backups\BackupLog.txt"

# Write a log entry correctly

$logMessage = "$timeStamp Backup completed from $source to
$destination"

Add-Content -Path $logFile -Value $logMessage
```

* Automated Monitoring and Reporting

Monitoring and Reporting in Windows Server (PowerShell)

Purpose:

This script monitors CPU usage, memory usage, and disk space on a Windows Server and generates a report.

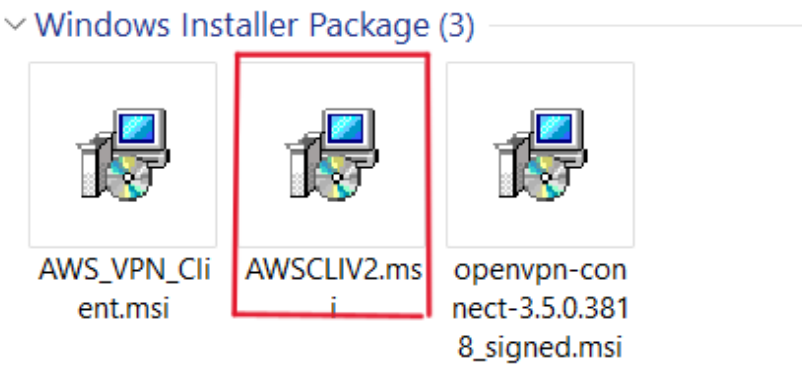
```
# Define the report file path

$reportPath =
"C:\Users\esraa\Documents\Reports\SystemReport.txt"

# Get system statistics

$cpuUsage = Get-WmiObject -Class Win32_Processor |
Measure-Object -Property LoadPercentage -Average | Select-Object
-ExpandProperty Average
```

```
$memory = Get-WmiObject -Class Win32_OperatingSystem

$totalMemory = [math]::round($memory.TotalVisibleMemorySize /
1MB, 2)

$freeMemory = [math]::round($memory.FreePhysicalMemory / 1MB, 2)

$usedMemory = [math]::round($totalMemory - $freeMemory, 2)

$diskSpace = Get-PSDrive C

$totalDiskSpace = [math]::round($diskSpace.Used / 1GB, 2)

$freeDiskSpace = [math]::round($diskSpace.Free / 1GB, 2)

# Create report content

$reportContent = @"

System Report - $(Get-Date)

CPU Usage: $cpuUsage%

Total Memory: $totalMemory MB

Used Memory: $usedMemory MB

Free Memory: $freeMemory MB

Total Disk Space: $totalDiskSpace GB

Free Disk Space: $freeDiskSpace GB

"@

# Write report to file

Add-Content -Path $reportPath -Value $reportContent
```

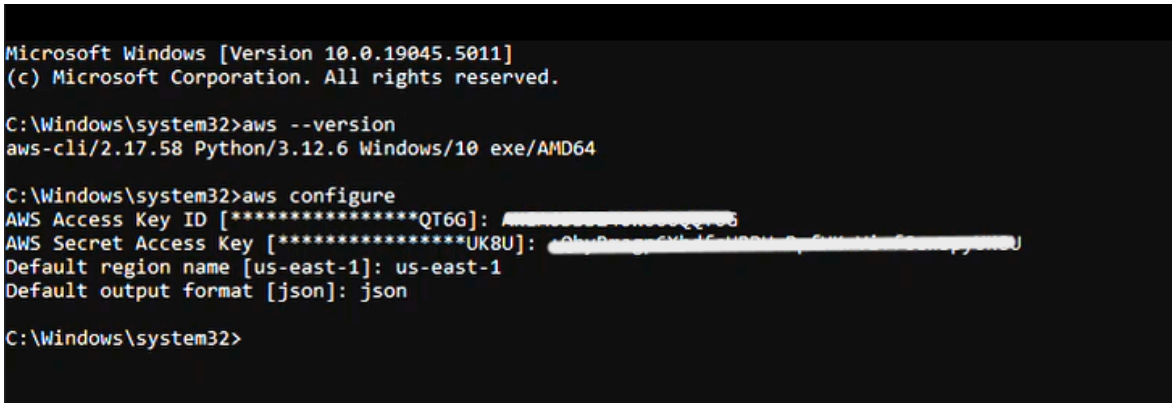
Watch the [Automation Tasks Video](#).

Automated Folder Backup to AWS S3 (Windows and PowerShell Only)

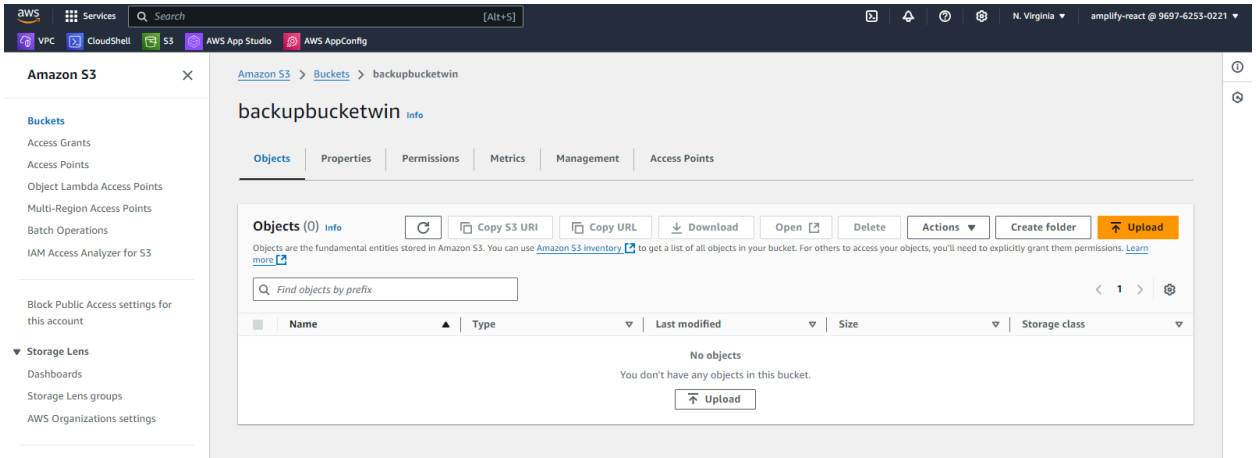
1. **Install AWS CLI on Windows:**
   - Download and install the AWS CLI.



2. **Configure AWS CLI:**

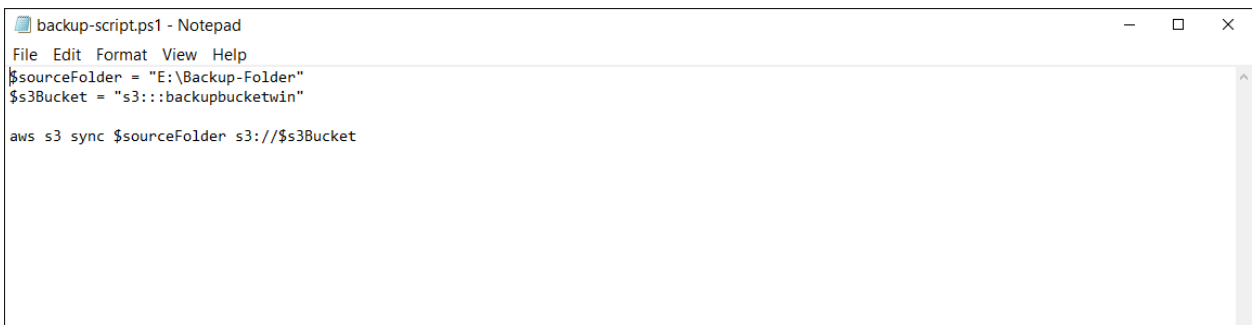


3. **Create an S3 Bucket:**

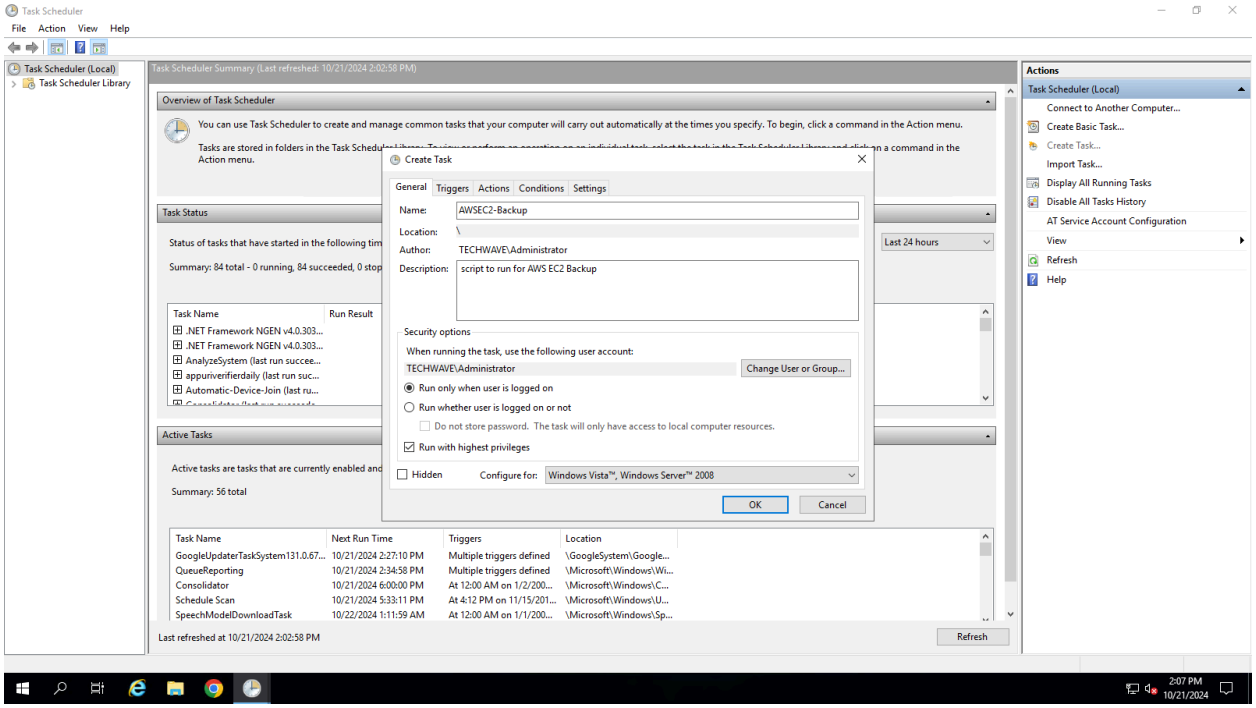


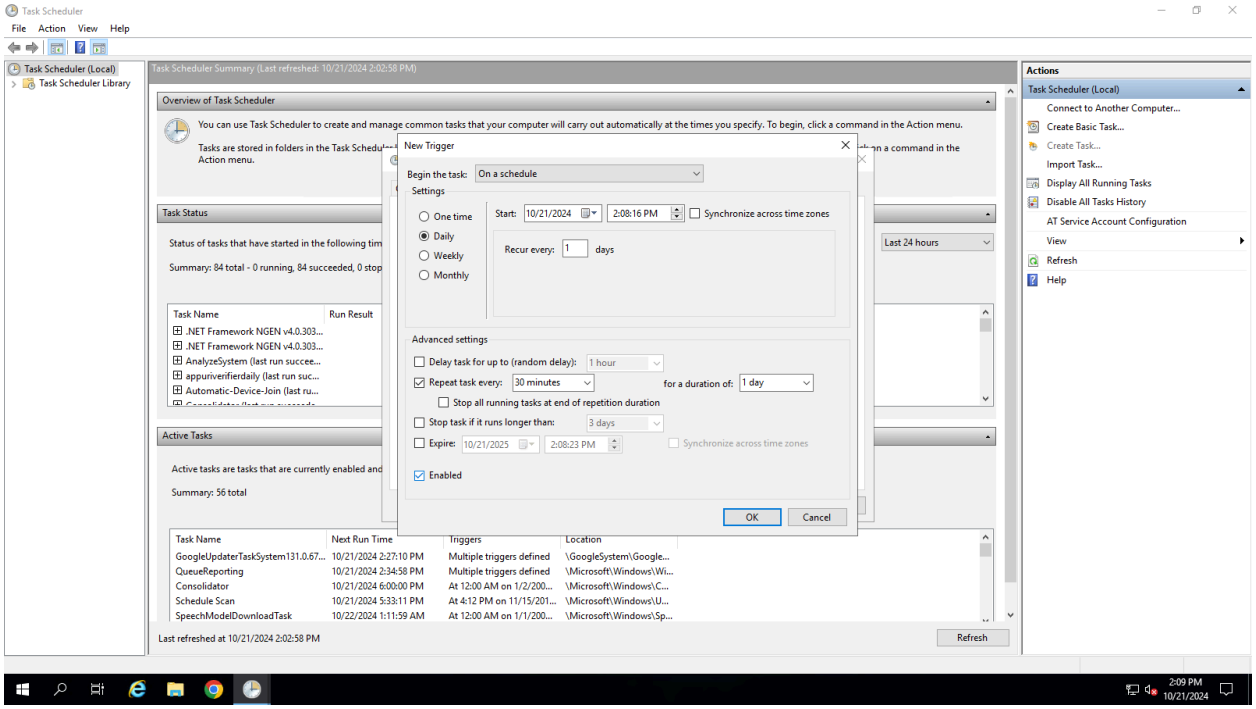
4. **Write PowerShell Script for Backup:**

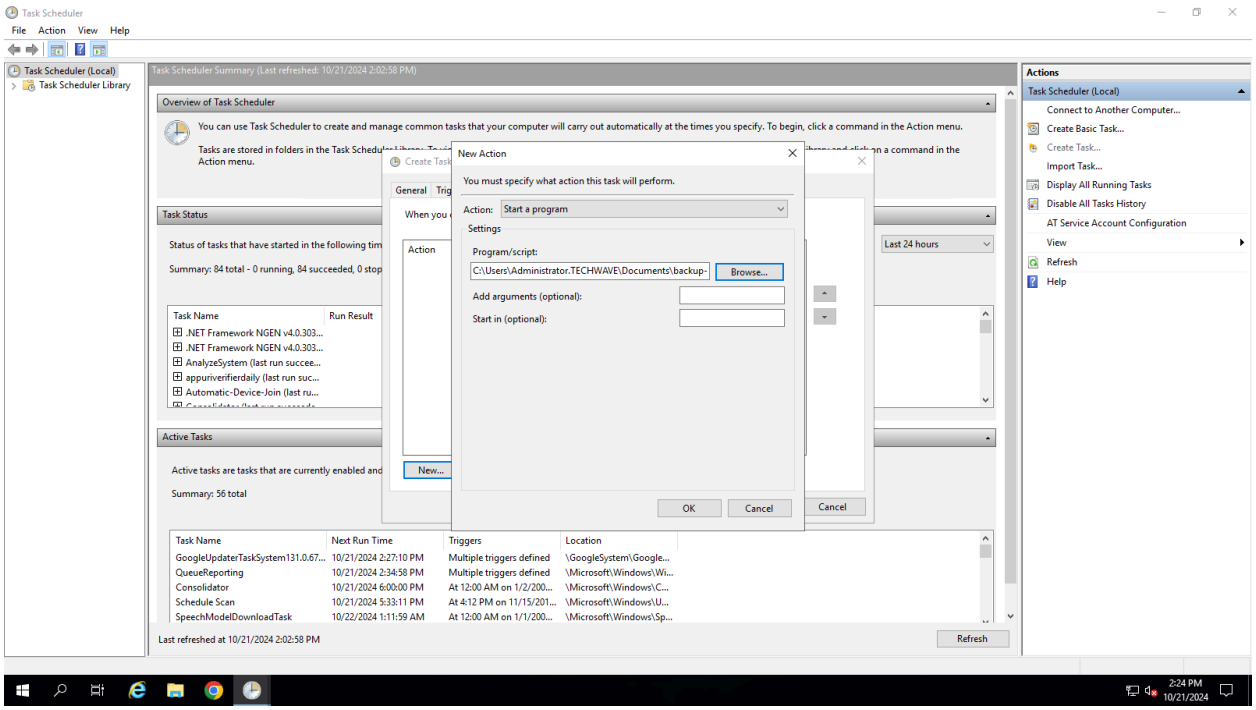
Create a PowerShell script to automate the folder backup to your S3 bucket.

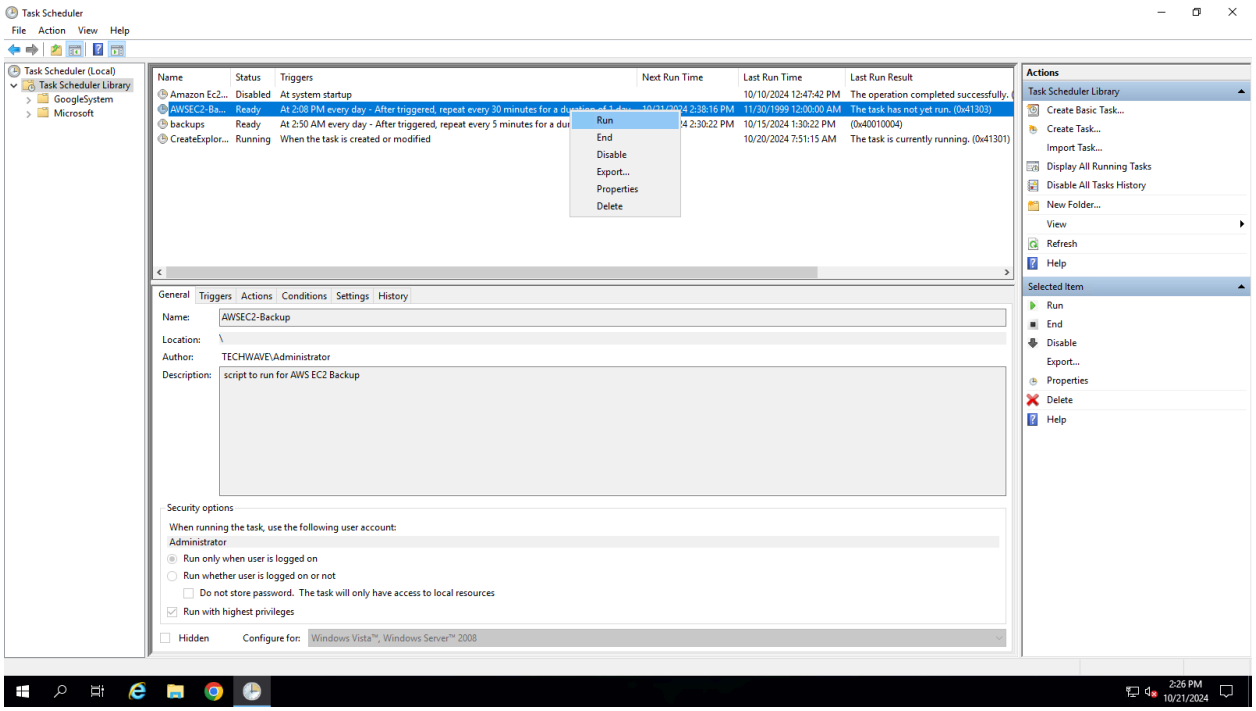


5. **Schedule Automated Backups with Task Scheduler:**
   - Open Task Scheduler on Windows and create a new task:
     - **Action:** Set to run the PowerShell script you created.
     - **Trigger:** Define when you want the backup to run (e.g., daily, weekly).
     - **Program/script:** Enter the automation script
     - **Arguments:** Add the full path of your PowerShell script.


Task Scheduler
File Action View Help
Task Scheduler (Local)
Task Scheduler Library
Task Scheduler Summary (Last refreshed: 10/21/2024 2:02:58 PM)
Overview of Task Scheduler
You can use Task Scheduler to create and manage common tasks that your computer will carry out automatically at the times you specify. To begin, click a command in the Action menu.
Tasks are stored in folders in the Task Scheduler Library. To view or perform an operation on an individual task, select the task in the Task Scheduler Library and click on a command in the Action menu.
Create Task
General Triggers Actions Conditions Settings
Name: AWSEC2-Backup
Location: \
Author: TECHWAVE\Administrator
Description: script to run for AWS EC2 Backup
Security options
When running the task, use the following user account:
TECHWAVE\Administrator
Change User or Group...
Run only when user is logged on
Run whether user is logged on or not
Do not store password. The task will only have access to local computer resources.
Run with highest privileges
Hidden
Configure for: Windows Vista™, Windows Server™ 2008
OK
Cancel
Task Status
Status of tasks that have started in the following tim
Summary: 84 total - 0 running, 84 succeeded, 0 stop
Last 24 hours
Task Name
Run Result
.NET Framework NGEN v4.0.303...
.NET Framework NGEN v4.0.303...
AnalyzeSystem (last run succee...
appuriverifierdaily (last run suc...
Automatic-Device-Join (last ru...
Active Tasks
Active tasks are tasks that are currently enabled and
Summary: 56 total
Task Name
Next Run Time
Triggers
Location
GoogleUpdaterTaskSystem131.0.67... 10/21/2024 2:27:10 PM Multiple triggers defined \GoogleSystem\Google...
QueueReporting 10/21/2024 2:34:58 PM Multiple triggers defined \Microsoft\Windows\Wi...
Consolidator 10/21/2024 6:00:00 PM At 12:00 AM on 1/2/200... \Microsoft\Windows\C...
Schedule Scan 10/21/2024 5:33:11 PM At 4:12 PM on 11/15/201... \Microsoft\Windows\U...
SpeechModelDownloadTask 10/22/2024 1:11:59 AM At 12:00 AM on 1/1/200... \Microsoft\Windows\Sp...
Last refreshed at 10/21/2024 2:02:58 PM
Refresh
Actions
Task Scheduler (Local)
Connect to Another Computer...
Create Basic Task...
Create Task...
Import Task...
Display All Running Tasks
Disable All Tasks History
AT Service Account Configuration
View
Refresh
Help
2:07 PM
10/21/2024

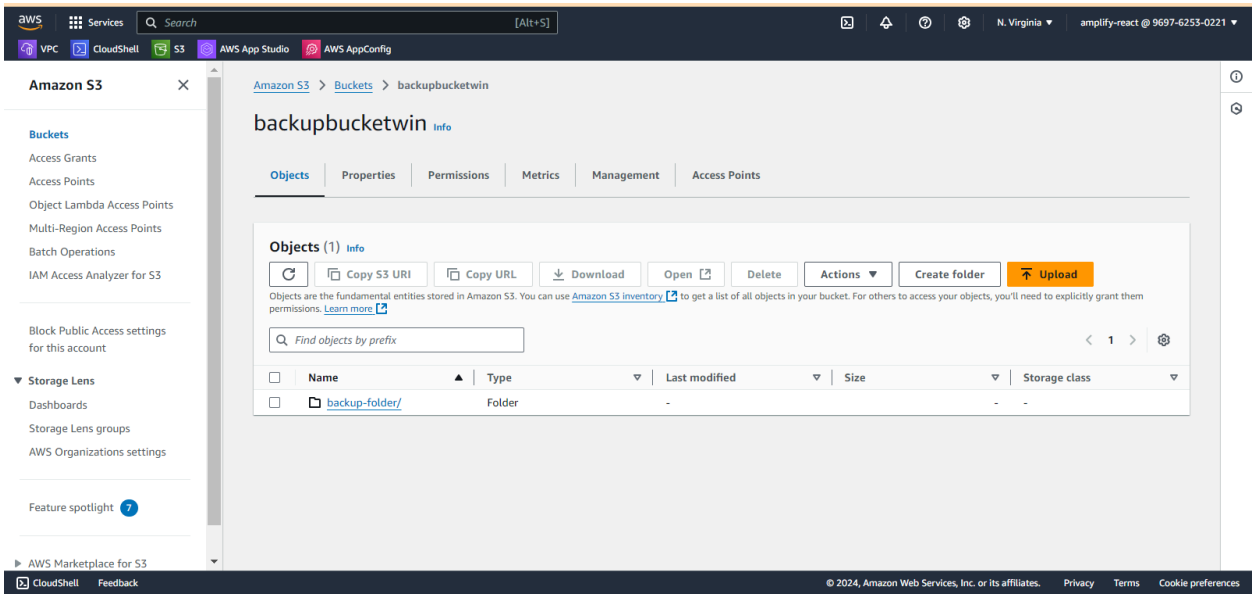

Task Scheduler
File Action View Help
Task Scheduler (Local)
Task Scheduler Library
Task Scheduler Summary (Last refreshed: 10/21/2024 2:02:58 PM)
Overview of Task Scheduler
You can use Task Scheduler to create and manage common tasks that your computer will carry out automatically at the times you specify. To begin, click a command in the Action menu.
New Trigger
Begin the task: On a schedule
Settings
One time
Daily
Weekly
Monthly
Start: 10/21/2024 2:08:16 PM
Synchronize across time zones
Recur every: 1 days
Advanced settings
Delay task for up to (random delay): 1 hour
Repeat task every: 30 minutes
for a duration of: 1 day
Stop all running tasks at end of repetition duration
Stop task if it runs longer than: 3 days
Expire: 10/21/2025 2:08:23 PM
Synchronize across time zones
Enabled
OK
Cancel
Task Status
Status of tasks that have started in the following tim
Summary: 84 total - 0 running, 84 succeeded, 0 stop
Last 24 hours
Task Name
Run Result
.NET Framework NGEN v4.0.303...
.NET Framework NGEN v4.0.303...
AnalyzeSystem (last run succee...
appuriverifierdaily (last run suc...
Automatic-Device-Join (last ru...
Active Tasks
Active tasks are tasks that are currently enabled and
Summary: 56 total
Task Name
Next Run Time
Triggers
Location
GoogleUpdaterTaskSystem131.0.67... 10/21/2024 2:27:10 PM Multiple triggers defined \GoogleSystem\Google...
QueueReporting 10/21/2024 2:34:58 PM Multiple triggers defined \Microsoft\Windows\Wi...
Consolidator 10/21/2024 6:00:00 PM At 12:00 AM on 1/2/200... \Microsoft\Windows\C...
Schedule Scan 10/21/2024 5:33:11 PM At 4:12 PM on 11/15/201... \Microsoft\Windows\U...
SpeechModelDownloadTask 10/22/2024 1:11:59 AM At 12:00 AM on 1/1/200... \Microsoft\Windows\Sp...
Last refreshed at 10/21/2024 2:02:58 PM
Refresh
Actions
Task Scheduler (Local)
Connect to Another Computer...
Create Basic Task...
Create Task...
Import Task...
Display All Running Tasks
Disable All Tasks History
AT Service Account Configuration
View
Refresh
Help
2:09 PM
10/21/2024





You can use AWS CloudWatch for notifications by:

- Set up an AWS CloudWatch Alarm for the S3 bucket to monitor backup activities and notify you of failures.



Conclusion

The automation of system monitoring, user management, backups, and patch updates is crucial for maintaining efficient operations in both Windows Server and Red Hat Enterprise Linux environments. Throughout the previous tasks, we explored the following key areas:

1. **User Creation and Group Management**:
    - We created scripts in both PowerShell and Bash to automate the creation of user accounts and their assignment to specific groups. This streamlines the onboarding process and enhances security by ensuring that users have the appropriate permissions based on their roles.
2. **Automated Backups**:
    - Daily backup scripts were developed to ensure that critical data on user computers is preserved regularly. Automating this process reduces the risk of data loss and enhances recovery strategies in case of hardware failures or other issues.
3. **Monthly Patch Updates**:
    - The implementation of automated scripts for monthly patch updates ensures that systems remain secure and up-to-date with the latest fixes and enhancements. Scheduling these updates minimizes downtime and user disruption while maintaining system integrity.
4. **System Monitoring and Reporting**:
    - Monitoring scripts were crafted to track CPU usage, memory statistics, and disk space, providing vital information about system health. Automating the generation of these reports facilitates proactive maintenance and quick identification of performance issues.
5. **Task Automation Scheduling**:

- Utilizing tools like Task Scheduler in Windows and Cron jobs in RHEL allows for seamless automation of scripts, ensuring that tasks run at predefined intervals without manual intervention.

Overall, the automation of these tasks leads to improved operational efficiency, enhanced security, and reduced administrative overhead. By implementing these strategies, IT administrators can focus on more strategic initiatives, knowing that routine monitoring, backups, and updates are handled consistently and reliably. This approach not only increases the reliability of IT infrastructure but also contributes to a more robust and responsive IT environment.