# **PowerShell Script that emails a report of every user that has logged in over the last 24 hours.**

**PowerShell script** that **tracks all user logins over the last 24 hours** and sends a **human-readable report** via email.

**How the Script Works**

1. **Extracts Login History:** Uses Get-WinEvent to pull logon events from the Windows Event Log (Event ID 4624 for successful logins).
2. **Filters Past 24 Hours:** Filters for logins within the last 24 hours.
3. **Formats Data:** Converts event details into a human-readable report.
4. **Sends Email:** Uses either **Outlook COM (if available)** or **SMTP (if Outlook isn’t set up).**
5. **Runs Daily:** Can be scheduled in **Task Scheduler** to run every 24 hours.

**PowerShell Script: LogUserActivity.ps1**

## ===========================================================================

## NAME:        LogUserActivity.ps1

## CREATED:     05-MAR-2025

## BY:          DAVID RADOICIC

## VERSION:     1.0

## DESCRIPTION: Checks the Windows system for all users logged in over the last 24 hours and sends an email.

##

##

## ===========================================================================

# Timezone configuration: Convert to Central Time

try {

    $CTzone = [System.TimeZoneInfo]::FindSystemTimeZoneById("Central Standard Time")

} catch {

    Write-Error "Central Timezone not found on system."

    $CTzone = $null

}

if ($CTzone) {

    $timestamp = [System.TimeZoneInfo]::ConvertTimeFromUtc((Get-Date).ToUniversalTime(), $CTzone)

    $timestampStr = $timestamp.ToString("yyyy-MM-dd HH:mm:ss")

    $tzLabel = "(Central Time)"

} else {

    $timestampStr = (Get-Date).ToString("yyyy-MM-dd HH:mm:ss")

    $tzLabel = "(Local Time)"

}

# Get all logon events (Event ID 4624 = successful login)

$startTime = (Get-Date).AddDays(-1)  # 24 hours ago

$logonEvents = Get-WinEvent -LogName Security -FilterHashtable @{LogName='Security'; Id=4624; StartTime=$startTime} -ErrorAction SilentlyContinue

# Process logon events

$loginRecords = @()

foreach ($event in $logonEvents) {

    $eventXML = [xml]$event.ToXml()

    $timeGenerated = [System.TimeZoneInfo]::ConvertTimeFromUtc($event.TimeCreated.ToUniversalTime(), $CTzone)

    $userName = $eventXML.Event.EventData.Data | Where-Object {$\_.Name -eq 'TargetUserName'} | Select-Object -ExpandProperty '#text'

    $logonType = $eventXML.Event.EventData.Data | Where-Object {$\_.Name -eq 'LogonType'} | Select-Object -ExpandProperty '#text'

    # Skip system/logon-as-service accounts

    if ($userName -notmatch '^(DWM|UMFD|ANONYMOUS|LOCAL SERVICE|NETWORK SERVICE|SYSTEM)$' -and $userName) {

        $loginRecords += "$($timeGenerated.ToString('yyyy-MM-dd HH:mm:ss')) $tzLabel - User: $userName (Logon Type: $logonType)"

    }

}

# Prepare report content

if ($loginRecords.Count -gt 0) {

    $reportBody = "User Logins in the Last 24 Hours:`n`n" + ($loginRecords -join "`n")

} else {

    $reportBody = "No user logins detected in the last 24 hours."

}

# Email Configuration

$recipientEmail = "YOUR EMAIL HERE"

$subject = "Daily User Login Report - $timestampStr CT"

# Attempt to send via Outlook

$sent = $false

try {

    $Outlook = New-Object -ComObject Outlook.Application

    $Mail = $Outlook.CreateItem(0)

    $Mail.To = $recipientEmail

    $Mail.Subject = $subject

    $Mail.Body = $reportBody

    $Mail.Send()

    $sent = $true

} catch {

    $sent = $false

}

# If Outlook fails, send via SMTP

if (-not $sent) {

    $SMTPServer = "smtp.office365.com"

    $SMTPPort = "587"

    $EmailFrom = "your\_hotmail\_account@hotmail.com"

    $SMTPUsername = "your\_hotmail\_account@hotmail.com"

    $SMTPPassword = "YourSecureAppPassword" # Replace with App Password if required

    # Convert password to secure string

    $SecurePassword = ConvertTo-SecureString $SMTPPassword -AsPlainText -Force

    $Credentials = New-Object System.Management.Automation.PSCredential ($SMTPUsername, $SecurePassword)

    # Send email via SMTP

    try {

        Send-MailMessage -From $EmailFrom -To $recipientEmail -Subject $subject -Body $reportBody -SmtpServer $SMTPServer -Credential $Credentials -UseSsl -Port $SMTPPort

        Write-Host "Email Sent Successfully."

    } catch {

        Write-Host "Email failed: $\_"

    }

}

**Explanation of the Script**

* **Event Log Filtering:** Extracts Windows Event ID 4624 (successful logins).
* **Filtering for the Last 24 Hours:** Filters logins from **(Get-Date).AddDays(-1)**.
* **User Account Filtering:** Removes system/service accounts (e.g., SYSTEM, NETWORK SERVICE, etc.).
* **Logon Type Included:** Helps differentiate local vs. remote logins.
* **Email Sending:**
  + **Tries Outlook first** (if available).
  + **Falls back to SMTP** if Outlook isn’t configured.

**How to Schedule the Script in Task Scheduler**

To **run this script every 24 hours**, follow these steps:

1. **Save the script** to a file, e.g., C:\Scripts\LogUserActivity.ps1.
2. **Open Task Scheduler** (taskschd.msc).
3. **Create a New Task**:
   * Name: Daily User Login Report
   * Set to **Run whether user is logged on or not**.
4. **Set the Trigger**:
   * Click **New** > Begin the task **On a schedule**.
   * Set **Daily** and pick a time (e.g., 12:00 AM).
   * Ensure **Repeat every: 24 hours** is selected.
5. **Set the Action**:
   * Click **New** > Action: **Start a Program**.
   * **Program/script:** powershell.exe
   * **Arguments:**
   * -ExecutionPolicy Bypass -File "C:\Scripts\LogUserActivity.ps1"
6. **Save and Test**:
   * Click **OK**, enter credentials (if needed).
   * Right-click the task and **Run** to test it.

**Expected Email Format**

**Subject:** Daily User Login Report - 2025-03-05 12:00:00 CT  
**Body:**

User Logins in the Last 24 Hours:

2025-03-05 09:45:12 (Central Time) - User: Alice (Logon Type: 2)

2025-03-05 10:15:47 (Central Time) - User: Bob (Logon Type: 10)

2025-03-05 13:30:23 (Central Time) - User: Charlie (Logon Type: 3)

*(Logon Type 2 = Interactive, Type 3 = Network, Type 10 = Remote Desktop, etc.)*

**Summary**

**Detects all user logins in the last 24 hours**  
**Formats results in a readable format**  
**Uses Outlook if available, SMTP as fallback**  
**Scheduled to run daily via Task Scheduler**