Track Logins on Windows

Microsoft hides the Remote Desktop logins pretty deep in their logging structure and there is not much information on how to programmatically get to it. I have a client who needed to get to this, so I figured I'd record what I came up with and some links.

There is a lot of documentation on how to do this if you are running a Windows Domain, but in the main case here, that is not the case. This procedure works on machines which are not on a domain, looking a the local server and parsing the local log files.

This has been tested on Windows 7, Server 2008r2 and Server 2019, the latter two running as Terminal Services servers.

getRDPLogs.ps1

```
# https://www.computerperformance.co.uk/powershell/if-statement/
# http://techgenix.com/dates-in-powershell/
# https://www.computerperformance.co.uk/powershell/if-and/
#
https://serverfault.com/questions/479048/remote-desktop-services-login-
history

# get date range from user
# NOTE: might be good to get EventID we are looking for also
do {
    $startDate = Read-Host "Enter the reports start date as
dd/mm/yyyy";
    $endDate = Read-Host "Enter the reports end date as dd/mm/yyyy";
    $startDate = [datetime]$startDate;
    $endDate = [datetime]$endDate;
} while ( $startDate -isnot [datetime] -And $endDate -isnot [datetime]
)

# this was used when it was a static entry, but unused now
#$startDate = (Get-Date -Year $year -Month $month -Day 01)
#$endDate = (Get-Date -Year 2020 -Month 03 -Day 01)

# name of the log to read. This contains activity on the Terminal
Services
$LogName = 'Microsoft-Windows-TerminalServices-
LocalSessionManager/Operational'
# we'll store results in this array
$Results = @()
# Get all events. See
#
https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell
.diagnostics/get-winevent?view=powershell-7
# for additional parameters. We might be able to speed processing with
# Get-WinEvent -LogName $LogName | Where-Object { $_.TimeCreated -ge
```

```
$startDate -And $_.TimeCreated -le $endDate
$Events = Get-WinEvent -LogName $LogName
# loop through all the events we found
foreach ($Event in $Events) {
    # convert to xml?
    $EventXml = [xml]$Event.ToXML()
    # filter for the event.id and between the dates (inclusive)
    if ( $Event.Id -eq 25 -And $Event.TimeCreated -ge $startDate -And
$Event.TimeCreated -le $endDate ) {
        # found one, so plug the stuff into a hash
        $ResultHash = @{
            Time        = $Event.TimeCreated.ToString()
            'Event ID'  = $Event.Id
            'Desc'      = ($Event.Message -split "`n")[0]
            Username    = $EventXml.Event.UserData.EventXML.User
            'Source IP' = $EventXml.Event.UserData.EventXML.Address
            'Details'   = $Event.Message
        }
        # then, take the result and append it to our results array
        $Results += (New-Object PSObject -Property $ResultHash)
    }
}
# figure out where to put the file
$currentDir = $(get-location).Path;
# and create a file name from the path and the start/end date
$currentDir = "$currentDir" + '\RemoteDesktopUsers_' +
$startDate.ToString("yyyy-MM-dd") + '_' + $endDate.ToString("yyyy-MM-
dd") + '.csv';
# dump it as CSV so they can read it via a spreadsheet.
$Results | Export-Csv -Path $currentDir;
```

The above script first asks for a start and end date for parsing, then opens *Microsoft-Windows-TerminalServices-LocalSessionManager/Operational* to get the logs. It then goes through each entry, looking for event type 25 (user logins) which fall in the date range. Once it has found all of them, it dumps the retrieved output to the same directory the script was run from as RemoteDesktopUsers_startdate_enddate.csv, a comma separated file which can be read by Excel or LibreOffice Calc.

Running the script

You can not just download and run the script under default windows security settings. By default, PowerShell is set to Restricted, meaning no PowerShell scripts can be executed. On Windows 10, the Execution Policy can be set to:

- Restricted - No scripts can be run
- RemoteSigned - only scripts created on this machine can be run
- AllSigned - Only scripts which have been signed can be run
- Unrestricted - runs any script without any restrictions.

Obviously, *Unrestricted* can be very bad juju. The simplest solution is to create the file locally and set the execution policy to RemoteSigned. That still has issues, but it will work. Open PowerShell as admin (run as Administrator), the execute:

```
Set-ExecutionPolicy RemoteSigned
```

You can now open Notepad or Poweshell ISE and copy/paste the above script, save it, then it should run (copying/pasting makes it think it is a locally created script).

After executing, you can reset PowerShell to Restricted mode to keep bad people from compromising your machine.

I know there is a way to set one PowerShell script to be executable, but I'm not sure how to do that. I'll update here if I find it.

Links

NOTE: I am not a powershell programmer, so the following links include some I used to figure out the syntax (I'm more familiar with Perl and BASH).

- https://serverfault.com/questions/479048/remote-desktop-services-login-history
- https://metacpan.org/pod/Win32::EventLog
- https://www.computerperformance.co.uk/powershell/if-statement/
- http://techgenix.com/dates-in-powershell/
- https://www.computerperformance.co.uk/powershell/if-and/
- https://serverfault.com/questions/479048/remote-desktop-services-login-history
- https://www.windowscentral.com/how-create-and-run-your-first-powershell-script-file-windows-10