**SQL Server Collector Guide – v2405**

**Overview**

The SQL Server collector consists of three scripts and is executed from a Windows-OS based client using Powershell. These scripts are server.csv, generate\_info\_files\_v2405.ps1, and generate\_workload\_files\_v2405.ps1. The servers.csv file is information supplied by the client pertaining to the SQL Server targets and is used as input for the two powershell (ps1) scripts. The powershell scripts, in turn, generate output files in a CSV format in designated locations. There are two separate scripts due to the frequency which they will likely be executed. Details of the collection process are in the following sections.

The script mentioned in this document are available [here](https://github.com/JonRAdams/jadams).

**Requirements**

The collector requires powershell version 5.1 or above with the SQL Server (sqlserver) module installed. Execute the command “$PSVersionTable” from the powershell command prompt to get the current version. To check for the sqlserver module, execute the command “Get-Command -Module sqlserver” at the powershell command prompt. The result should be a list of Cmdlets. If not, install the module by opening a powershell interface as an administrator and execute the command “Install-Module sqlserver”. Executing these scripts from the powershell command line is simple. From the powershell prompt, execute the command “<path>\<collector\_script>”. Whenever these powershell scripts are executed, output files with the format YYYYMMDD\_HH24MMSS in the name are created to prevent output files being overwritten during each execution.

**Input file (servers.csv)**

For the powershell scripts to collect details on the target environments, the client needs to provide details on each target in the server.csv file. Here is an extract of the contents of that file in a table format for readability:

|  |  |  |  |
| --- | --- | --- | --- |
| enabled | server\_ip | username | password |
| N | WIN-IIVTJUU077H\SS2014 |  |  |
| Y | WINJON |  |  |
| Y | WINJON | username | password |
| N | WIN-K7N5T128MKU |  |  |

This simple format includes these columns:

enabled – A Y (yes) or N (no) is used in this column to enable or disable the target entry for the collection process.

server\_ip – The names of the SQL Server instance in servername\instance format or the IP address of the SQL Server instance

username – The SQL Server instance username the collector will use to connect

password – The SQL Server instance password associated with the account in the username column

Note – If the username and password fields are used, then the collector uses SQL Server authentication for target connection, else Windows authentication is used if these fields are left empty. Both these fields must be empty or have values listed.

It is recommended to create a specific read-only type of user for this collections process on the target. As this file stays in the possession and is used exclusively by the client, these credentials are not shared outside the client organization. The collection process does not include the user credentials in the CSV output files.

**Information collection (generate\_info\_files\_v2404.ps1)**

This file collects informational data. This data does not change often or at all and this script may only need to be executed once.

The following changes need to be made to the script prior to execution.

# Designate the client name

$ClientName = "TestClient"

This section is the client name. Abbreviations are encouraged and the value needs to be a single word with a mix of upper- and lower-case letters with (optional) numbers. The first character MUST be alphabetical. Using special characters and spaces is not allowed. This value will be the prefix of the output files (TestClient\_nodeinfo\_20240506\_080423.csv).

$Path = 'C:\Users\Joe\Documents\info\' + $ClientName (TestClient)

Use this variable to designate the location of the info output CSV files. In this example, the files are created in a directory called “info” under the current user’s directory. Remember, YYYYMMDD\_HH24MISS directories will be created under this designated directory during each execution.

# Import the target server data

$CSVData = Import-Csv -Path ".\servers.csv"

Designate the location of the input servers.csv file here. The example here is used when the servers.csv file is in the same location as the collector scripts.

**Workload collection (generate\_workload\_files\_v2404.ps1)**

This file collects workload data. This type of data changes often and this script will be executed more than once to create more data points.

The following changes need to be made to the script prior to execution.

# Designate the client name

$ClientName = "TestClient"

This section is the client name. Abbreviations are encouraged and the value needs to be a single word with a mix of upper- and lower-case letters with (optional) numbers. Using special characters and spaces is not allowed. This value will be the prefix of the output files (TestClient\_cpu\_20240506\_080423.csv).

$Path = 'C:\Users\Joe\Documents\workload\' + $ClientName (Example - TestClient)

Use this variable to designate the location of the info output CSV files. In this example, the files are created in a directory called “info” under the current user’s directory. Remember, YYYYMMDD\_HH24MISS directories will be created under this designated directory during each execution.

# Import the target server data

$CSVData = Import-Csv -Path ".\servers.csv"

Designate the location of the input servers.csv file here. The example here is used when the servers.csv file is in the same location as the collector scripts.

**User creation and destruction scripts**

Upon request from the client, scripts that create and destroy collection users will be given. The purpose of these scripts is to either create or destroy user accounts created specifically for the collection process. These scripts are not necessary if Windows authenticated accounts are used for the collection process. Care must be taken when using the user destruction script to prevent the removal of necessary accounts.

**User creation**

This script uses the server.csv file for input, but instead of using the values in the username and password fields as the credentials for connecting to the target databases, it uses these credentials to create the collection user on the target database(s). For example, using the settings below, it would create a user called “username” with a password of “password” on the WINJON target. The execution of this script will generate errors if the user already exists.

|  |  |  |  |
| --- | --- | --- | --- |
| enabled | server\_ip | username | password |
| N | WIN-IIVTJUU077H\SS2014 |  |  |
| N | WINJON |  |  |
| Y | WINJON | username | password |
| N | WIN-K7N5T128MKU |  |  |

**User deletion**

This script uses the server.csv file for input, but instead of using the values in the username and password fields as the credentials for connecting to the target databases, it uses these credentials to delete this user on the target database(s). For example, using the settings below, it would delete the user called “username” if it exists on the WINJON target. **Extra care needs to be taken when executing this script to prevent the deletion of necessary user accounts.**

|  |  |  |  |
| --- | --- | --- | --- |
| enabled | server\_ip | username | password |
| N | WIN-IIVTJUU077H\SS2014 |  |  |
| N | WINJON |  |  |
| Y | WINJON | username | password |
| N | WIN-K7N5T128MKU |  |  |