Periodic Checks script documentation

Contents

[Version history 2](#_Toc14091150)

[Description 3](#_Toc14091151)

[Prerequisites 3](#_Toc14091152)

[Account requirements 3](#_Toc14091153)

[Installation 4](#_Toc14091154)

[Configuration Files 5](#_Toc14091155)

[Config.xml 5](#_Toc14091156)

[Config\Servers.txt 5](#_Toc14091157)

[Config\Urls.txt 6](#_Toc14091158)

[Config\Patchexclusions.txt 6](#_Toc14091159)

[Config\groups.txt 6](#_Toc14091160)

[Checkdefinitions\\*.ps1 7](#_Toc14091161)

[Check Descriptions 8](#_Toc14091162)

[“Missing Security Updates Check” (ID M1) 8](#_Toc14091163)

[”Services check” (ID 31) 8](#_Toc14091164)

[Debugging 8](#_Toc14091165)

[Exit codes 9](#_Toc14091166)

[Known issues 9](#_Toc14091167)

## Version history

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Comment |
| 1.10 | 13-7-2017 | Yorick Kuijs | - Updated documentation (MBSA check and CredSSP prereqs), fixed naming issue in check 17 |
| 1.11 | 25-1-2018 | Yorick Kuijs | - Added possibility to use multiple email addresses, separated with comma  - Added check for valid email address  - Improved script relative path support. The script now ensures all files and folders and found in the script folder  - Added validation of server configuration  - Added exitcodes  - Added information about debug the script to the documentation  - Added Distributed Cache check |
| 1.12 | 9-7-2018 | Yorick Kuijs | - Improved MBSA check logging to show reason of a failed scan  - Fixed script duration per server calculation issue  - Updated wait procedure to display how many servers have completed  - Updated MBSA check to leave the reports when Debug is set to True  - Added ".NET v4.0" and ".NET v4.0 Classic" application pools to default ignored application pools  - Added check to validate if the user has sufficient permissions to use PowerShell with SharePoint (Only for servers where Role=SP)  - Added check if SharePoint plugin exists (Only for servers where Role=SP)  - Added check if Distributed Cache module can be found |
| 1.13 | 14-9-2018 | Yorick Kuijs | - Added possibility to configure CC and BCC addresses as report recipients |
| 1.14 | 18-12-2018 | Yorick Kuijs | - Added Full parameter to enable the possiblity to force run all checks  - Updated Distributed Cache check (check 18) to first validate if the DC is actually running on the specified server  - Updated URLCheck (check 1) to allow authentication against an ADFS/Windows Claims environment |
| 1.15 | 15-7-2019 | Yorick Kuijs | - Removed obsolete parameter Search String in url.txt file  - Added folder and file checks to make sure required configuration files really exist  - Added Config parameter to enable the possibility to specify custom configuration file  - Improved "Failed Timer Jobs" check to make it more efficient  - Minor bugfixes |
| 1.16 | 4-11-2019 | Yorick Kuijs | - Published script on GitHub  - Fixed bug with reading config file, introduced in v1.15  - Removed the dependency on MBSA for the Missing Patch scan  - Increased PowerShell version requirement to v5.0 (due to use of Copy-Item -ToSession) |
| 1.17 | 19-6-2020 | Yorick Kuijs | - Added check 19 to check the size of content databases in GB  - Updated Excel checklist with all new tests |

## Description

To properly manage an environment, it is important that periodic checks are performed. These checks validate that certain configuration is configured and if indications of larger issues are present.

A list of checks has been generated and can be found in “Overview Periodic Checks.docx”, including the frequency in which these should be performed.

To reduce the amount of time required to perform these checks and the chance of errors, a script has been created to automate as many checks as possible. This documentation describes how the script works and how to configure it.

## Prerequisites

* This script only works with SharePoint 2013 and above. It does not work with SharePoint 2010 and below.
  + SharePoint 2010 is using the .NET Framework v3.5 and does not work with v4.0 and above. PowerShell v3.0 and above require the .NET Framework v4.0 or higher. Therefore SharePoint 2010 only works with PowerShell v2.0.
* The Internet Explorer First Launch wizard needs to be run for the account that is executing the script. To do this, log in with the account and open Internet Explorer once.

If you don’t do this, you will get the following error:

*Invoke-WebRequest : The response content cannot be parsed because the Internet Explorer engine is not available, or Internet Explorer's first-launch configuration is not complete. Specify the UseBasicParsing parameter and try again.*

* You need to configure Powershell Remoting and CredSSP on both the script server and the SharePoint/SQL servers:
  + Script server:

Enable-PSRemoting[[1]](#footnote-1)

Enable-WSManCredSSP -Role Client -DelegateComputer \*.domain.com

* + SharePoint/SQL:

Enable-PSRemoting1

Enable-WSManCredSSP -Role Server

## Account requirements

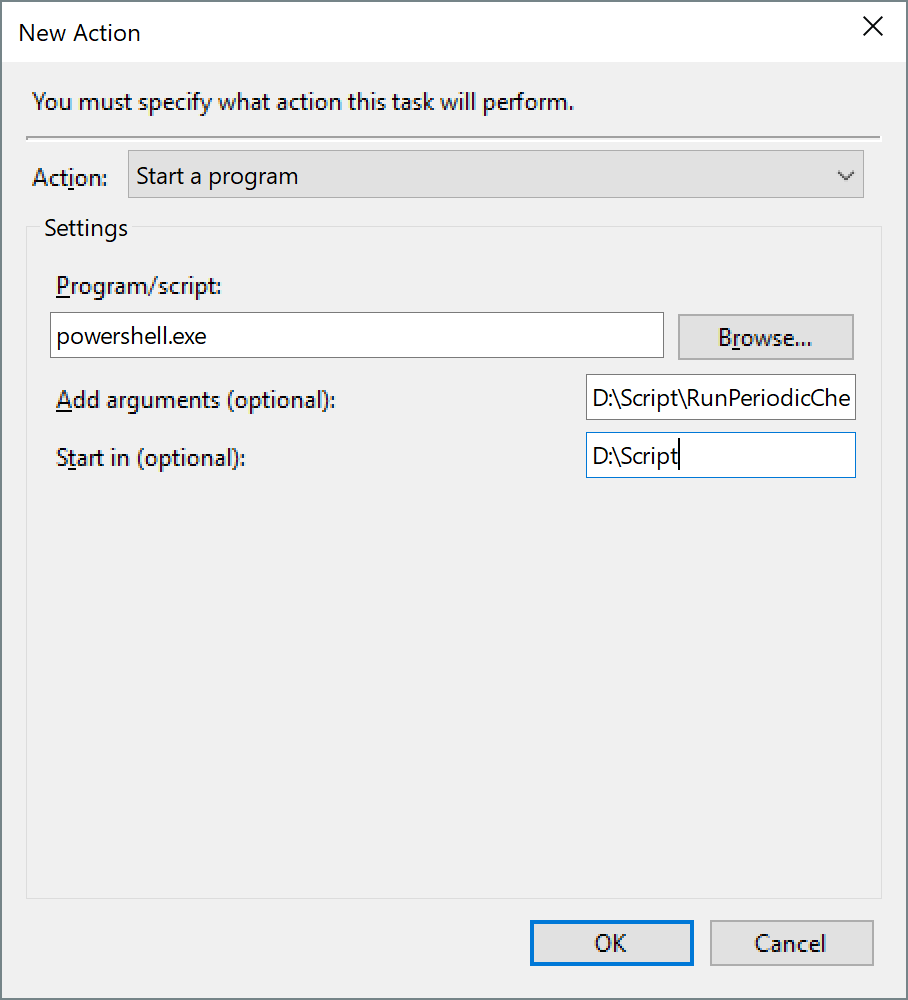
The account that is executing the script requires the following permissions on the environment:

* Member of the local Administrators group on all servers
* Farm administrator in the Farm (by default inherited via the local Administrators group)
* DBOwner on all SharePoint databases
* Full Control via User Policy on all web applications

This account will be used to run the scheduled task under and to connect to all remote machines.

## Installation

1. Make sure all prerequisites are in place and the required service account is created.
2. Create a folder and copy all files in the package into this folder
   1. Note: It is not required to copy the Documentation folder as well.
3. Open the config.xml file and update all sections
   1. Important: The encrypted password must be encrypted using the service account on the server which is going to run the script[[2]](#footnote-2).
      1. Log onto the script server **as the service account**
      2. Run the GeneratePassword.ps1 script and enter the credentials
      3. Open the password.txt file and copy the encrypted string
      4. Paste the encrypted string in the Password field of the config.xml file
4. Browse to the config folder and edit the files. More information about these files is described in the next paragraphs.
5. Open Scheduled Tasks and create a new scheduled task. In the Action section, make sure you enter powershell.exe as Program and the full script name as argument. Use the script folder as the “Start in” location.



1. Use the service account as the Run As account for the Scheduled Task.
2. Test the Scheduled Task by right clicking it and selecting Run. When completed, the script should have a result code of 0x0.
3. A Logs folder will be created by the script which will contain the output logs. Check the created log.

## Configuration Files

### Config.xml

This file is used for the general configuration of the script. It contains all settings required for successful execution of the script.

|  |  |  |
| --- | --- | --- |
| General | |  |
|  | Debug | Run the script in Debug mode. This will output more information to the screen and save the report to file instead of sending it via mail. |
|  | RemoteTimeOut | Specifies how long a script is allowed to run on a remote server (in seconds) |
| Credentials | |  |
|  | UserName | Username which is used to connect to the servers. |
|  | Password | Password of the above account. The password is stored as a secure string. |
| Email | |  |
|  | SendReportsViaEmail | Specifies if the reports should be send via email, using the below parameters |
|  | SMTPServer | SMTP server used for sending the report |
|  | MailFrom | Mail address which is used to send the mail from. |
|  | MailTo | Mail address which is used to send the report to. |
|  | MailCC | Mail address which should be placed on CC line. Leave empty if CC is not required. |
|  | MailBCC | Mail address which should be placed on BCC line. Leave empty if CC is not required. |
| Reporting | |  |
|  | ReportsToDisk | Specifies if the reports should be stored to disk, using the below parameter |
|  | ReportsFolder | Which folder should be used to store the HTML reports |
| PatchScan | |  |
|  | WSUSCabPath | Relative path of the WsusScn2.cab file, usually the local folder (“.”) |
|  | DownloadWSUSFile | Automatically download the newest version of the WsusScan2.cab file.  **NOTE:** Internet connection required. |

|  |  |  |
| --- | --- | --- |
| Logging | |  |
|  | LogPrefix | Prefix of the log file |
|  | LogFolder | Folder name of the folder where logs are stored. The folder will be created under the script folder. |
|  | RemoteLogFolder | Location where the logs should be stored on the remote servers. |

### Config\Servers.txt

This file contains the details of all servers in the farm.

**Note:** Make sure only one server in the farm is configured as centraladmin=yes. This will prevent the script from running the same checks multiple times for the same farm.

|  |  |
| --- | --- |
| **Column** | **Description** |
| servername | FQDN of the server |
| farm | Friendly name of the farm, for example SharePointPRD, using in reporting |
| centraladmin | Is the server running the centraladmin role. This server is used for executing the SharePoint scans.  Possible values: yes, no  Note: To prevent running the same tests multiple times on the same farm, only specify one CA server in the farm. |
| role | Specifies the role of the server. This information is used to determine which checks need to run against which servers.  Possible values: SP, SQL, OWA, APP, Other |
| tier | Specifies the tier of the server. This is for documentation purposes only.  Possible values: WFE, BE, APP, N/A |

### Config\Urls.txt

This file specifies the URL’s which need to be checked. Each URL will be checked for a specific string to determine if the page is opened successfully.

|  |  |
| --- | --- |
| **Column** | **Description** |
| URL | URL which needs to be checked |
| Environment | Which environment does this URL point to |

### Config\Patchexclusions.txt

If for whatever reason a patch is not installed, this file makes sure that patch excluded from the report. You can also document the reason for excluding the patch. The script is not using this reason.

|  |  |
| --- | --- |
| **Column** | **Description** |
| Patch | KB article which needs to be excluded from the scans (only the KB number) |
| Reason | Document the reason why this patch is excluded |

### Config\groups.txt

This file specifies which users need to be member of which groups. The script for Group check (nr. 34) checks if the specified groups have the specified members.

Each line specifies a groupname/useraccount combination which will be checked. Multiple lines can therefore use the same group name.

|  |  |
| --- | --- |
| **Column** | **Description** |
| Groupname | The name of the group which needs to be checked. |
| Useraccount | The name of the account or group which needs to be member of the specified group. |

### Checkdefinitions\\*.ps1

The ps1 files in the checkdefinitions folder are the config and logic of the actual checks. Each file start with a configuration section to specify the details of the check, followed by the actual logic. By adding a file to the folder, you can add a check to the script.

|  |  |
| --- | --- |
| **Column** | **Description** |
| ID | ID of the check |
| Check | Short name of the check. Used together with the ID to generate the function name which executes the check.  NOTE: No spaces allowed! |
| Description | Description of the check. Used in the report. |
| Target | Where does the check needs to execute: Local on the servers in the farm or remote from the tools machine.  Possible values:   * Remote (execute check on script machine) * Local (execute check on SharePoint machine) |
| Type | What is the type of the check. Check URL availability, check SharePoint Farm setting, check on all servers or check only on SharePoint servers.  Possible values: URLs, Farm, ServersAll, ServersSP, ServersSQL |
| Schedule | On which schedule does the check need to run.  Possible values: Daily, Weekly, Monthly, Quarterly |

Numbering of the check is following a numbering standard:

* 1-9: Remote checks
* 10-19: Farm checks
* 20-29: Search checks
* 30-39: Server checks
* 40-49: Manual checks
* 50-59: Custom checks

## Additional Check Descriptions

### “Missing Security Updates Check” (ID M1)

The Missing Security Updates Check requires an update definition (wsusscn2.cab) file to check against, which can either be automatically downloaded from the Internet or downloaded manually.

The most recent version of the wsusscn2.cab file can be downloaded from: <http://go.microsoft.com/fwlink/?linkid=74689>

More information about the wsusscn2.cab file can be found at: <https://support.microsoft.com/en-us/help/926464/a-new-version-of-the-windows-update-offline-scan-file,-wsusscn2.cab,-i>

### 

### ”Services check” (ID 31)

One of the daily checks is checking if Windows services that are supposed to be running are actually running. The check is using a configuration file on each server to determine which services should be running. This file can be found at “c:\Windows\Monitoring” and is called “servicesconfig.txt”.

If the configuration file does not exist, the script will create it when it is running for the first time. The script will use the current state as the desired state.

The configuration file can be changed as required. By adding a hashtag in front of a line, the line will be ignored. The format of the file is:

“Service\_short\_name”,”State”

For example:

“W32Time”,”Running”

The script is only checking for services that are configured as “Running” in the configuration file and are currently stopped.

## Script parameters

The script accepts two optional parameters:

1. Full : This is a switch parameter that forces all checks to be executed.
2. Config : Specifies the name of the configuration xml and configuration files folder. Default is config.xml

Examples:

Run all checks: .\RunPeriodicChecks.ps1 -Full

Use a specific configuration XML: .\RunPeriodicChecks.ps1 -Config configtst.xml

## Debugging

When setting the Debug section in the config.xml file to “true”, the script behaves a little different:

1. It will create a Debug folder in which all scripts are stored that are run on the remote servers. Each server will get its own file. Using these scripts, you can test the script yourself on the remote server
2. It will store the generated HTML report on the local drive instead of sending it via e-mail. That way you prevent the mailbox from being spammed during testing.

## Exit codes

The script is using Exit codes when serious errors occur that prevent the script from executing properly. Each exit code has a specific meaning:

* 0 : No errors encountered
* 10: Cannot find config.xml file
* 20: Config.xml does not match schema
* 30: Incorrect server configuration
* 40: Error during creating report folder
* 50: Invalid email addresses specified in config.xml file
* 60: No password configured in config.xml file
* 70: Configuration folder not found
* 80: Specific configuration file not found

## 

## Known issues

* None

1. Enabled by default since Windows Server 2012 [↑](#footnote-ref-1)
2. Encypted password scan only be decrypted on the **same computer** under the **same user** account which were used to encrypt the password. [↑](#footnote-ref-2)