QA Checks
Quick Start Guide v4

This document is intended for system builders and administrators who maintain servers within various environments. It's to get you up and running quickly.

You do not need to know PowerShell to use these scripts but it will help in diagnosing any issues that may occur. Please make sure you read this document fully as well as the information posted in the Wiki location of the GitHub repository.

This document refers to version 4 of the scripts. It can be used for other versions however some of the screen shots and/or wording may be different.

Contents

Overview

The QA checks came about as a need to verify the build of new servers for various customers and their different environments. All new server builds are usually done with a custom gold image template; however this image still lacks many of the additional tools and configuration settings needed before it can be accepted into support.

Most of this extra tooling and configuration should be automated, however checks are still needed to make sure each customer or environment has their specific settings in place.

Technical details

The scripts are written in the Microsoft PowerShell language, with version 4 in mind. Version 4 was used as a base as it is available on most modern Windows operating systems and as some significant scripting enhancements over version 2.

These scripts also require WinRM to be configured to scan remote servers. WinRM can be configured either with or without SSL. Credentials and authentication type can be specified on the command line.

Supported operating systems

Windows Server 2008 R1 and R2 (PowerShell 4 is not installed by default)
Windows Server 2012 R1 and R2
Windows Server 2016

Unsupported operating systems

Windows 2003 Server
Any non-server operating system
Any non-Windows operating system

The Checks

There are over 100 checks split over 9 separate sections. These are executed whenever the QA script is run against one or more servers and can take anywhere between 30 seconds and a couple of minutes to complete. If you are checking multiple servers then this time is per server.

For a full list of checks and their sections, check the GitHub Wiki page:
https://github.com/my-random-thoughts/qa-checks-v4/wiki/sections

Getting Started

There are three stages required to get you up and running. The first one is the quickest and will generate a HTML report of its findings. This report will have quite a number of failures in most environments. Don't worry; stage two will help you fix these failures - either as a scan configuration change or to highlight areas in the environment that may need some work.

Stage 1 - Generating the initial report

1. Download the latest QA package from GitHub and extract them to a folder on your machine.
   https://github.com/my-random-thoughts/qa-checks-v4

2. Open a PowerShell window and change to the folder where you extracted the QA script files.

3. Type the following: `.\Compiler.ps1` to compile the checks into a single script file to run.

4. You should now have a file called `QA_v4.yy.mmdd.ps1`. This will be named with today's date.

5. Copy this single script to a server you want to get the initial report from.

6. On this server, open an elevated PowerShell window and change to the script folder.

7. Execute the script: `.\QA_v4.yy.mmdd.ps1 -ComputerName localhost`
   There are several command line options available with the scanning engine, these are listed in Appendix A at the end of this document.

8. Wait for the scan to complete. A progress bar shows the current status.

9. Once complete, a HTML report will be generated and saved in the default location of C:\QA\Reports. The report name will be the name of the server and the date and time the scan took place. For a breakdown of the HTML report, see Appendix B.

   This report will list quite a number of failures, don't worry, we are going to configure the scripts for your specific environment next.

Move on to stage 2.

Stage 2 - Configuring the scripts

Now that you have your report, you can use it to see what needs fixing in your environment and what needs tweaking in the configuration of the scripts. For example, the very first check makes sure that no additional user accounts exist on a server. Almost all environments will have a custom AD group added in order to manage the server remotely. This AD group can be ignored as you know that it should be there. These are the types of tweaks that the script needs to know about and can be ignored for future checks.

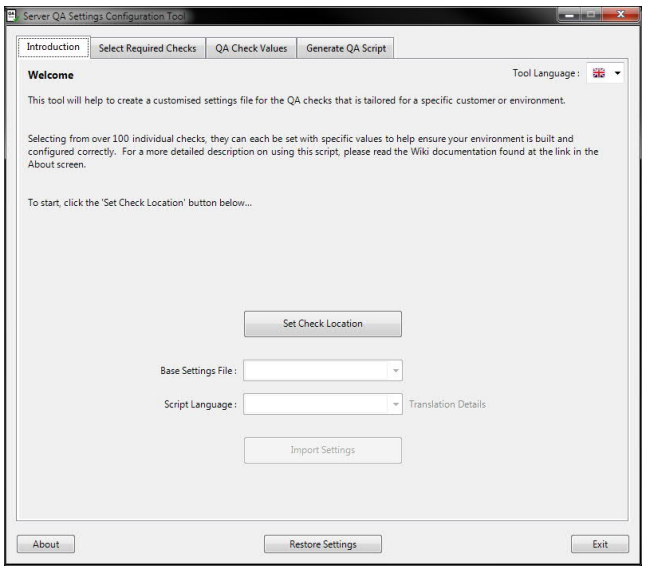
QA Settings Configuration Tool

We will be using the QA Settings Configuration Tool (QASCT) to complete this stage. See the end of this document for another way of making minor changes to your configuration.

QASCT - Introduction

10. From a standard PowerShell window, run the tool by typing `.\QA-Settings-Configuration-Tool.ps1`, alternatively right-click the file and choose Run with PowerShell if that option is available.

11. After a few seconds the following window will appear:



    Note: If you want to use this tool in a different language, choose one from the drop-down list at the top-right of the window. More languages are coming - can you help with translations?

12. Read the welcome text, and then click the Set Check Location button.

13. From the window that appears, select the folder where you extracted the files in step 2 above. The tool will try to open the current folder automatically.

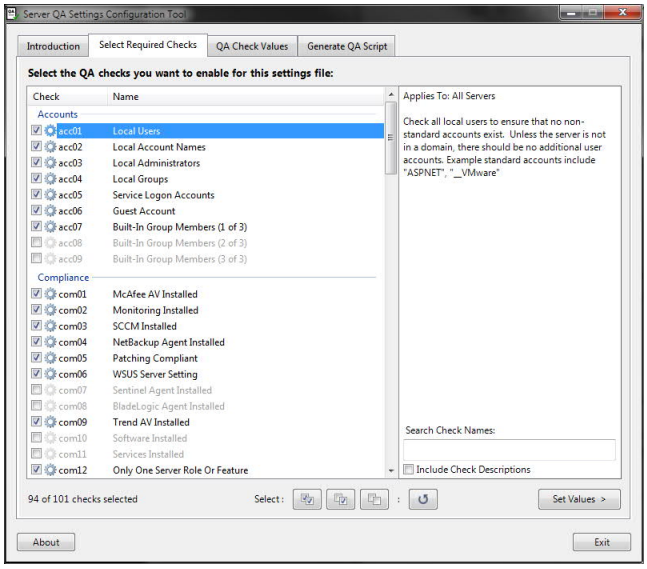
14. Since this is a new "installation", there will only be one base settings file: `default-settings`.
    If you want to modify an existing configuration, select it here.

15. If more than one language exists, and you want to use a different one, select it from the second box.

16. Click Import Settings when you are ready.

QASCT - Select required checks

17. Using your HTML report from step 9 above, carefully choose which checks you believe are relevant to your environment. For example, in the Compliance section there is more than one anti-virus check, only one should be active anything else will fail as not being installed.



18. Once you have selected all the checks you want to use, click Set Values > to move onto the next page.

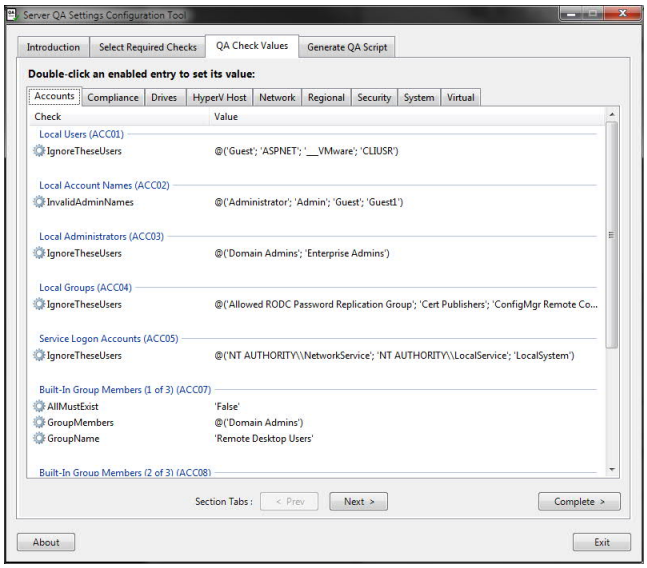
QASCT - QA check values

This page is a little more complicated and will take some time to complete. It may also involve input from other teams within your organisation. You can come back to configuration as many times as needed, so there is no need to make all the changes in one go.

Take each section at a time and using your HTML report, change the current default values to ones that match the particular environment you are configuring for.

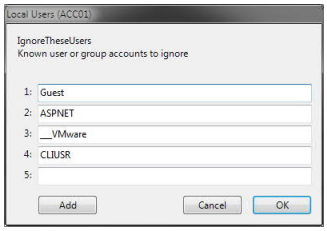
Using the account example from above, your HTML report may have failed the first check (`ACC-01`) saying that one or more extra user accounts exist. If you know that all of these accounts are acceptable for your environment and can be ignored, follow the steps below. These steps can be used for any check that you want to modify...

19. Select the correct section page from the second row of tabs.



20. Locate the correct check from the group headers (not all checks will have an entry).

21. Double-click each of the entries in that group to edit its value.
    For the above example (`ACC-01`), the `IgnoreTheseUsers` option is opened and shows a pre-filled list of users that can be ignored. Add each of your known user or groups accounts to this list.
    Click Add if you need more rows



22. Click OK when you have finished

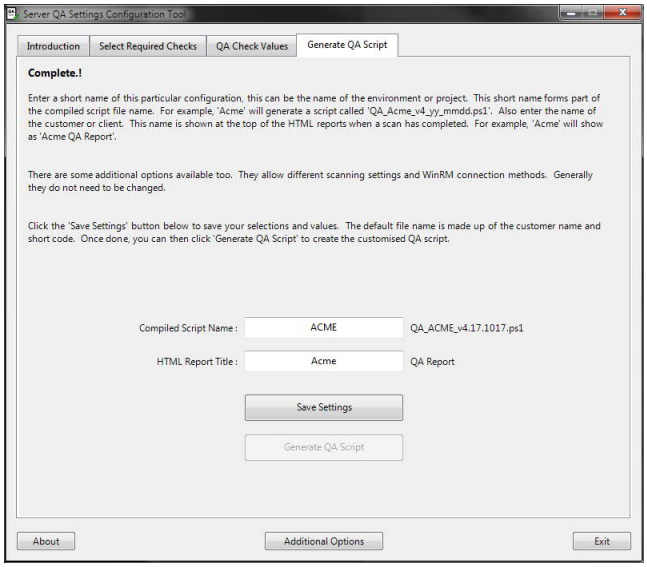
23. Repeat steps 19 - 22 for each check in each section.

24. Once you have finished, click Complete > to move onto the next page

QASCT - Generate QA script

Now that you have completed all the changes required, the last page is shown. This allows you to give your configuration a name. This will help identification if you have several of them.

25. Enter a short 12-character name for this configuration file. It should be meaningful to you. It could represent a particular customer name (ACME) or environment (DEV, TEST, LIVE)



26. Also enter a name to show at the top of the HTML report. The default is set to "Acme".

27. Click Save Settings to save your configuration script. The default name is made up from the two entries above. The file needs to be saved in the settings folder.

28. Click Generate QA Script to compile all your selected checks and environment specific configuration into a single customised script. The script will be located in the root QA folder and be called `QA_{shortname}_v4.yy.mmdd.ps1` where `yy.mmdd` represents the current date.

29. Click Exit, and then Yes to close the tool.

Move on to stage 3.

Stage 3 - Run the report again

Now that you have a customised script, repeat the whole process starting from step 5. This will generate a new report that should hopefully be close to a finished configuration. Check to see if any more tweaks can be made to either the configuration or your environment.

From your completed configuration and compiled QA script, you can now scan a few more servers and check the results.

Alternative Configuration Editing Method

If you only want to make a minor change to a configuration file, the QA Settings Configuration Tool may be a little overkill. In this case, simply open the file using your favourite text editor. The file is a plain text file set out like a standard INI file (https://en.wikipedia.org/wiki/INI_file)

Appendix A - QA Scripts Command Line Options

There are a few optional command line options that can be used with the scanning engine. They are listed below. They can also been shown in a PowerShell window by executing the QA check script with `-Help`.

Options

`-ComputerName` Enter one or more servers to scan. A list of servers can be given on the command line or via a PowerShell function. For example, a list of servers can be read in from a text file, or a filtered list direct from Active Directory.

`-DoNotPing` Some environments are locked down and do not allow ICMP to work. Using this option will attempt to connect to a remote server without sending a ping to it first.

`-SkipHTMLHelp` When generating the HTML reports, do not add the hover help feature. This reduces the help file size by about 50%. See Appendix C for more information.

`-GenerateCSV` As well as generating the HTML file for every single server that is scanned, this option will only create a single CSV file containing all the servers that have been scanned.

`-GenerateXML` As above with the CSV option above, this generates an XML file.

`-Credential` If you need to supply alternative credentials for a remote server connection, you can specify them here. You can supply just the user name and the script will ask for the password, or you can pass a credential object.

```
QA.ps1 -ComputerName remote1 -Credential 'acme\user1'
QA.ps1 -ComputerName remote2 -Credential $credObject
```

`-Authentication` If you need to use a different authentication type to connect to a remote server, you can set it here. The available options are:

```
Basic, CredSSP, Default, Digest,
Kerberos, Negotiate, NegotiateWithImplicitCredential
```

See the following Microsoft article for more information on these options:
https://docs.microsoft.com/dotnet/api/system.management.automation.runspaces

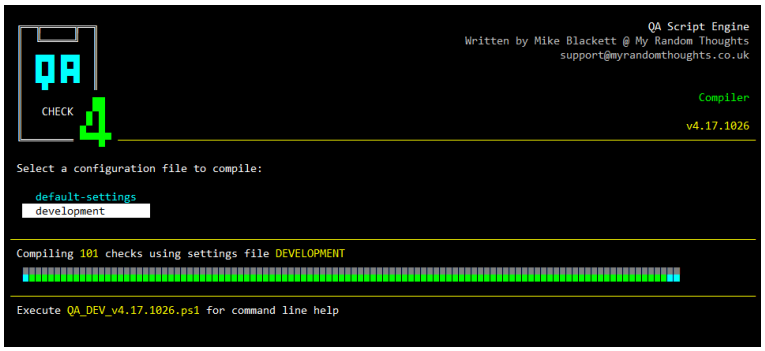
Appendix B - Compiler and Scanning Engine

The screen shots below show an example of the compiler and scanning engine running on a local server.

Compiler

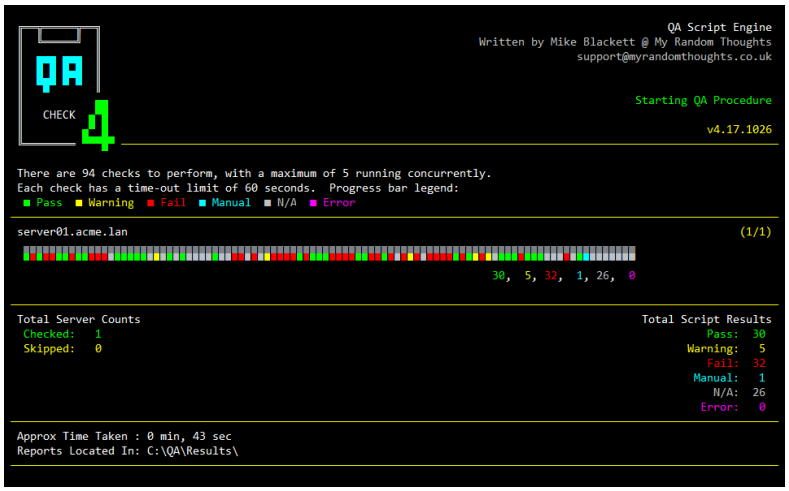
Whenever you make a change to a configuration file, you need to compile a new QA script. This is done by executing the compile script. The script will present a menu of configuration scripts for you to choose from, or you can specify one on the command line:

```
.\Compiler.ps1
.\Compiler.ps1 -Settings {filename}
```



Scanning Engine

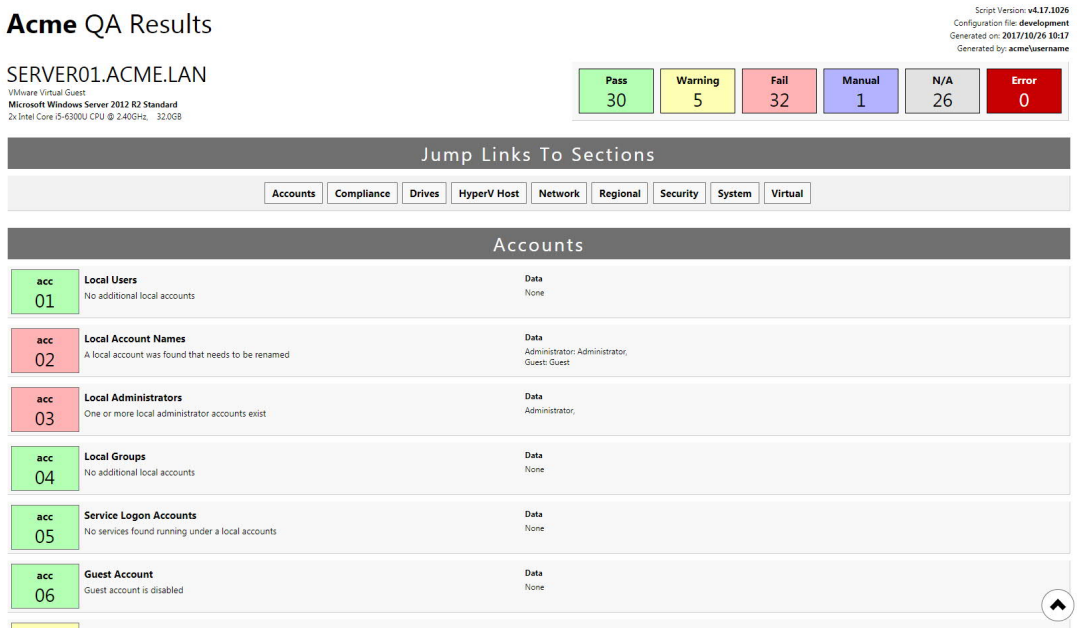
Whenever you run a compiled script, the scanning engine handles all the work. It runs the scans, collects the results, and generated the report files. An example output of the engine is shown below. See Appendix A for a list of the QA Scripts command line options.



As you can see from the progress bar, this scan is producing a lot of failed checks. They will need looking at.

Appendix C - HTML Report Breakdown

A HTML report is generated for every single server that is scanned. It is stored (by default) in C:\QA\Reports, and named after the server that is represents.



Header

Looking at the header section at the top, the left hand area shows the report company name (Acme) as well as the server name and some basic details about it (Type, OS, CPU, and RAM). The right hand side shows the script version and configuration used, when it was run and by whom. It also shows the overall results of the scan, in this case a very poor configuration with 32 failures.

Next we have links to each of the sections in this report. This is a quick way to jump to any section within the document. There is a floating “back to top” link at the bottom-right of the page.

Check Results

After this, each section is shown with every check and its results. Looking at the first entry, ACC-01, the details show the name and short description of the check and its state; a pass. The Data section on the right shows any information that may be helpful.

Help

Hovering over the coloured check square on the left hand side shows the hover-help window, which gives more information about the check and the Pass/Fail messages. This may help in determining what is required for a pass.