



# Security Configuration and Hardening Guide Tools

VMware vSphere 8.0.2



# Table of Contents

Revision History .....3

Introduction .....4

Disclaimer.....4

License.....4

Support.....4

How To Use These Tools .....5

Feedback .....7

## Revision History

See the main vSphere Security Configuration & Hardening Guide for information about revisions to guidance itself.

Date	Description of Change
October 5, 2023	Initial Release of vSphere Security Configuration & Hardening Tools.



## Introduction

The VMware vSphere Security Configuration & Hardening Guide (SCG) has been the foundation for VMware vSphere hardening and auditing for over fifteen years. Starting with vSphere 8.0.2, the SCG introduced sample scripts to automate auditing. The scripts serve three main purposes:

- **Ease of Use for Beginners:** These scripts act as a stepping stone for those new to scripting, while also having an important purpose. Using the readily available VMware PowerCLI cmdlets with PowerShell makes vSphere automation straightforward. The scripts prioritize readability over elegance to ensure they align closely with SCG examples and can be easily modified by administrators as needed.
- **Simplicity & Integration:** Adhering to the UNIX philosophy of doing one thing and doing it well, these scripts each have a single purpose, and can be used in conjunction with inherent features of PowerShell. For instance, instead of creating a custom logging function, output can be channeled to the native Tee-Object for simultaneous file saving and display. Likewise, the Select-String command is useful for pattern matching, such as for finding audit lines containing the labels [PASS] and [FAIL]. Extensive examples are provided below.
- **Generating Audit Records:** The output is structured to provide audit details like dates, times, hostnames, and current configurations. This allows the scripts to capture a snapshot of an environment, aiding regulatory compliance.

While these tools offer significant advantages, they aren't a one-size-fits-all solution. They can't assess design nuances, firewall configurations, patch levels, and more. Nevertheless, they can decrease the manual effort tied to the SCG's controls.

## Disclaimer

This kit is intended to provide general guidance for organizations that are considering VMware solutions. The information contained in this document is for educational and informational purposes only. This document is not intended to provide advice and is provided "AS IS." VMware makes no claims, promises, or guarantees about the accuracy, completeness, or adequacy of the information contained herein. Organizations should engage appropriate legal, business, technical, and audit expertise within their specific organization for review of requirements and effectiveness of implementations.

## License

Copyright 2023 VMware, Inc. All Rights Reserved. Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

## Support

We regret that while we are happy to accept constructive feedback about the code examples and tools, we cannot supply direct support for them, through the author or via VMware Global Support Services. There are options for scripting and automation support through VMware Professional Services. Please contact your Account Executive for more information. You might also check out the thriving community at [developer.vmware.com](https://developer.vmware.com).

## How To Use These Tools

### Step 0: Software Requirements

These scripts are built on VMware PowerCLI. They require VMware PowerCLI 13.0.0 or newer. Installation instructions for PowerCLI can be found at <https://developer.vmware.com/powercli> but it can be as simple as opening a relatively recent version of PowerShell (such as version 5.1 on a default Windows 10 desktop) and typing:

```
Install-Module -Name VMware.PowerCLI
```

waiting a while for it to complete, and then typing:

```
Install-Module -Name VMware.vSphere.SsoAdmin
```

These tools assume, and check for, VMware vCenter Server 8 and VMware ESXi 8. Using these tools against a different environment will have untested results. If you wish to subvert the safety checks, each script has a “-NoSafetyChecks” flag you can use. See below for more information.

### Step 1: Connection Requirements

You will need to connect to a VMware vCenter Server, version 8. There are two methods for connecting. First, you can use the following commands to do so, substituting the correct values for User, Server, and perhaps Password (see below):

```
Connect-VIServer -User username@vsphere.local -Server vcenter.yourdomain.com
```

```
Connect-CisServer -User username@vsphere.local -Server vcenter.yourdomain.com
```

```
Connect-SsoAdminServer -User username@vsphere.local -Server vcenter.yourdomain.com
```

Second, you can use the included connect.ps1 script:

```
.\connect.ps1 -vCenter vcenter-1.8.fcotr.org -User username@vsphere.local
```

This script will prompt for a password, collected from the console and masked with asterisks (\*).

While it may be tempting to automate these connection strings, **under no circumstances do we recommend storing account logon information in a script**. Doing so is a leading cause of unauthorized access, breaches, and eventual situations like ransomware. Properly storing account information for automated tools depends heavily on your own environment and is out of scope for this document.

### Step 2: Run The Tools

Change into the directory with the scripts and issue a command like:

```
.\audit-esxi-8.ps1 -Name esx-1.8.fcotr.org
```

Replacing the value after “-Name” with a valid hostname in your environment. Similarly:

```
.\audit-vm-8.ps1 -Name TESTVM4
```

However, the vCenter auditing script does not require a name, since you’re already connected:

```
.\audit-vcenter-8.ps1
```

Running the tools individually gives you an idea of what the output will look like and will help expose any issues with their execution.

### Step 3: Troubleshoot

Each script has additional flags you can use as needed:

“-NoSafetyChecks” which allows the script to run unhindered. Your mileage will vary.

“-NoSafetyChecksExceptAppliances” which allows audit-vm-8.ps1 to skip all checks except the ones for VMware appliances, like the vCenter Server Appliance, vCLS VMs, vSphere Cloud Gateway, and so on. Changing settings on those appliances is unsupported as per VMware Global Support Services policy.

“-NoBanners” which removes the header and the disclaimer but leaves the INFO statements with the date and time.

### Step 4: Read the Output

Each line from the script will begin with the name of the object being examined, and then have a label:

[ERROR] – The script has an error and exited.

[INFO] – Informational output, such as date, time, and target of the scan.

[TEXT] – Non-critical informational text, which can be suppressed with -NoBanners.

[PASS] – The control being tested passed the check.

[FAIL] – The control being tested did not pass the check.

Each line will have the current configured value in parentheses at the end of the line.

No audit is perfect. Failures may simply indicate that something needs to be checked manually. For example, physical NICs connected to access ports will fail the check for default VLANs, even though they are not on a trunk and therefore not vulnerable to that type of problem.

### Step 4: Get Fancy

#### Filtering with Select-String

All that text looks interesting, but how do you find what needs to be fixed? Try piping the output into PowerShell commands to filter the output:

```
.\audit-esxi-8.ps1 -Name esx-1.8.fcotr.org -NoBanners | Select-String -Pattern "[FAIL\]|[INFO\]"
```

This will return the lines that require further checking, labeled with [FAIL].

Characters like brackets ([ or ]) are special characters to PowerShell, and require “escaping” or making the shell understand not to interpret them. The backslash (\) is what does that. The vertical pipe (|) symbol in the pattern means “or.” A tremendous use of modern Large Language Model (LLM) AIs is to ask them for help constructing patterns such as these. For instance, a statement like “Please give me the correct pattern for use with Select-String in PowerShell to find lines that contain [INFO], [PASS], and [FAIL]” will return a useful example.

#### Writing to Disk with Tee-Object

You can use Tee-Object to duplicate the script output, writing it to disk but also passing it along to the console:

```
.\audit-esxi-8.ps1 -Name esx-1.8.fcotr.org -NoBanners | Tee-Object -FilePath output.txt -Append | Select-String -Pattern "[FAIL\]|[INFO\]"
```

This will append the output of the command to “output.txt” in the current directory.

## Step 5: Audit Everything

If you create a directory for output you can use the following command to audit everything attached to your vCenter Server:

```
mkdir output
.\audit-all.ps1 -Directory .\output\
```

This command will find all VMs, ESXi hosts, and the vCenter Server you are already attached to, and will recursively audit them and save the output. Feel free to use a directory that isn't named "output" – you might choose a date and/or a regulatory compliance framework name instead (and if so, may the audit be smooth).

PowerShell can find all the [FAIL] lines in the "output" directory with a command such as:

```
Get-ChildItem -Path .\output\ -Filter '*.txt' | Get-Content | Select-String -Pattern '\[(FAIL)\]' | Sort-Object Line
```

As noted, if you are new to PowerShell an LLM AI can help with commands such as these. In fact, this line was generated by asking "Can you give me a PowerShell command that I can use to read all the .txt files in a directory and then find lines in them with [FAIL], sorting them alphabetically?" :)

## Step 6: Remediate

The SCG tools don't have remediation scripts yet, so you are left to remediate using the examples and data in the Security Configuration & Hardening Guide controls spreadsheet. Our suggestion is to sort by Implementation Priority, fixing PO items first, as they are the most important. A future version of the audit tools will indicate priority as well.

## Step 7: Customize

Every environment has audit findings that are not actionable but continue to appear in reports. A good example here might be "unnecessary hardware," where a particular device, such as an XHCI controller, might be flagged but it is actually required for proper operation of the guest OS on your virtual machines. These scripts are set up in a way where you should be able to easily find and edit those out if they are truly false positives.

Similarly, you could filter them after the fact with additional Select-String commands.

## Feedback

We strive for accuracy and usefulness and appreciate feedback. Please visit:

<https://via.vmware.com/scg>

and use the Feedback mechanism on the page there to send us information. Thank you.

