

Patch Compliance On Windows

Patch Information

Get-Hotfix

Patch-Velocity

Counts the number of patches applied per day

Get-Hotfix | Sort-Object InstalledOn -Descending

Patch-Age

Patch age of a system is the number of days since the last patch was applied:

\$lastPatchDate = (Get-HotFix | Sort-Object InstalledOn -Descending | Select-Object -First 1).InstalledOn

\$lastPatchDate

(New-TimeSpan -Start \$lastPatchDate -End (Get-Date)).TotalDays

Patch Compliance on Debian-based Linux Distributions

Change shell to Powershell Core

Pwsh

Patches installed by the Apt package manager are logged in:

/var/log/dpkg.log

Patch-Velocity

Counts the number of patches applied per day

Get-Content /var/log/dpkg.log* | Select-String "install" -NoEmphasis

Get-Content /var/log/dpkg.log* |

Select-String "install" -NoEmphasis |

Out-File ./patches.txt -Encoding ascii

\$lines = Get-Content ./patches.txt

(\$lines | Where-Object { \$_ -match "^[0-9]" }) -replace ".*\$"

Patch-Age

Patch age of a system is the number of days since the last patch was applied:

\$lastPatchDate = (\$lines |

Where-Object { \$_ -match "^[0-9]" }) -replace ".*\$" |

Select-Object -last 1

\$patchAge = (New-TimeSpan -Start (Get-Date -date \$lastPatchDate) `

-End (Get-Date)).TotalDays

"Last Patch Date: \$lastPatchDate"

"Patch Age: \$patchAge"

Windows Compliance Measurements

Check that the Administrator account is disabled:

Get-LocalUser -Name Administrator

Check that the Guest account is disabled:

Get-LocalUser -Name Guest

Save the list of local users to a variable and then test to see if both Guest and Administrator are disabled:

\$disabledUsers = Get-LocalUser | Where-Object Enabled -eq \$False

(\$disabledUsers.Name -contains 'Administrator') -And

(\$disabledUsers.Name -contains 'Guest')

Enumerate the members of local groups:

(Get-LocalGroupMember -Name Administrators | Measure-Object).Count

(Get-LocalGroupMember -Name 'Power Users').Count

Check that a Windows services is installed, enabled and running:

((Get-Service -Name <service name>).Count -ge 1) -And

((Get-Service -Name <service name>).Status -eq 'running') -And

((Get-Service -Name <service name>).StartType -like 'Automatic*')

All commands, unless stated otherwise, have been tested in the

SEC557: Continuous Automation for Enterprise and Cloud Compliance

course VMs using PowerShell Core.

SANS Cybersecurity Leadership Curriculum

CYBERSECURITY LEADERSHIP

MGT 512

Security Leadership Essentials for Managers

Leading Security Initiatives to Manage Information Risk

MGT 514

Security Strategic Planning, Policy, and Leadership

Aligning Security Initiatives with Strategy

MGT 516

Managing Security Vulnerabilities: Enterprise and Cloud

Stop Treating Symptoms – Cure the Disease

MGT 521

Leading Cybersecurity Change: Building a Security-Based Culture

Build and Measure a Strong Security Culture to Secure Your Workforce

MGT 551

Building and Leading Security Operations Centers

Prevent – Detect – Respond | People – Process – Technology

SEC 566

Implementing and Auditing Security Frameworks and Controls

Building and Auditing Critical Security Controls

AUD 507

Auditing & Monitoring Networks, Perimeters, and Systems

Controls That Matter – Controls That Work

LEG 523

Law of Data Security and Investigations

Bridging the Gap between Legal and Cybersecurity

SEC 557

Continuous Automation for Enterprise and Cloud Compliance

Measure What Matters, Not What's Easy

MGT 414

SANS Training Program for the CISSP® Certification

Need Training for the CISSP® Exam?

MGT 433

Managing Human Risk: Mature Security Awareness Programs

People are the Primary Attack Vector. Manage Your Human Risk.

MGT 520

Leading Cloud Security Design and Implementation

Building and Leading a Cloud Security Program

MGT 525

Managing Cybersecurity Initiatives & Effective Communication

Meet and Exceed Your Security Program's Goals

MGT 415

A Practical Introduction to Cyber Security Risk Management

Cutting Through Academic: Practical Risk management for Cybersecurity

MGT 553

Cyber Incident Management

Open in Case of Emergency

SEC 440

CIS Critical Controls: A Practical Introduction

Introduction to Critical Security Controls

sans.org/cybersecurity-leadership

SANS Security Leadership

secleadership

ACE

SANS CLOUD SECURITY

POWERSHELL FOR ENTERPRISE AND CLOUD COMPLIANCE

By AJ Yawn

Cheat Sheet v1.0.2

SANS.ORG/CLOUD-SECURITY

SANS.ORG/SEC557

CLOUD SECURITY

CURRICULUM ROADMAP

SEC 510

Public Cloud Security: AWS, Azure, and GCP | gpcs

Multiple clouds require multiple solutions.

SEC 540

Cloud Security and DevSecOps Automation | gcsa

The cloud moves fast. Automate to keep up.

SEC 541

Cloud Security Attacker Techniques, Monitoring, and Threat Detection

Attackers can run but not hide. Our radar sees all threats.

SEC 549

Enterprise Cloud Security Architecture

Design it right from the start.

SEC 522

Application Security: Securing Web Apps, APIs, and Microservices | gwsa

Not a matter of "if" but "when" be prepared for a web attack. We'll teach you how.

SEC 557

Continuous Automation for Enterprise and Cloud Compliance

Measure what matters, not what's easy.

SEC 588

Cloud Penetration Testing | gcpw

Aim your arrows to the sky and penetrate the cloud.

FOR 508

Enterprise Cloud Forensics and Incident Response

Find the storm in the cloud.

SEC 388

Introduction to Cloud Computing and Security

Ground school for cloud security.

SEC 488

Cloud Security Essentials | gccd

License to learn cloud security.

MGT 520

Leading Cloud Security Design and Implementation

Chart your course to cloud security.

MGT 516

Managing Security Vulnerabilities: Enterprise and Cloud

Stop treating the symptoms. Cure the disease.

sans.org/cloud-security

@SANSCloudSec

linkedin.com/showcase/sanscloudsec

AWS Compliance Measurements

Make sure current version AWS PowerShell module is available for use.
Install-Module -name AWSPowerShell.NetCore -Scope CurrentUser -Force

Load AWS Module
Import-Module AWSPowerShell.NetCore

Authenticate to AWS
Set-AWSCredential -StoreAs <name of profile> -AccessKey YourAccessKeyHere -SecretKey YourSecretKeyHere

CIS AWS Benchmark Control 1.4
(Get-IAMAccountSummary).AccountAccessKeysPresent

CIS AWS Benchmark Control 1.5
(Get-IAMAccountSummary).AccountMFAEnabled

CIS AWS Benchmark Control 1.8
Get-IAMAccountPasswordPolicy

CIS AWS Benchmark Control 1.13
Get-IAMUserList | ForEach-Object { Get-IAMAccessKey -UserName \$_.UserName }

CIS AWS Benchmark Control 1.15
**(Get-IAMUserList | ForEach-Object { Get-IAMUserPolicies -UserName \$_.UserName | Select-Object PolicyName
Get-IAMAttachedUserPolicies -UserName \$_.UserName | Select-Object PolicyName
})**

CIS AWS Benchmark Control 3.1
Get-CTTrail

Measure VMWare host configuration

Gather information about the VMWare host system and configuration
Get-VMHost -Server <name>

Validate common hypervisor settings
(Get-VMHost).ExtensionData
Leverage the Config property in ExtensionData to get in depth config settings (example DNS resolver configuration below)
(Get-VMHost).ExtensionData.Config.Network.DNSConfig
Measure if DNS settings are configured correctly:
**\$dnsservers = (Get-VMHost).ExtensionData.Config.Network.DNSConfig | Select-Object -ExpandProperty address
\$dnsservers -contains '8.8.8.8'
\$dnsservers -contains '8.8.4.4'**
Validate the NTP server(s) configured on VMWare host
Get-VMHost -Server <name> | Get-VMHostNtpServer
Validate that the NTP service is running and is configured to run at startup
Get-VMHost | Get-VMHostService | Where-Object {\$_.key -eq "ntpd"} | Select-Object VMHost, Label, Key, Policy, Running, Required

Patch Data
(Get-ESXCLI -Server esxi1).software.vib.list()
Patch velocity
(Get-ESXCLI -Server esxi1).software.vib.list() | Group-Object InstallDate
Patch Age
**\$lastPatchDate = ((Get-ESXCLI -Server esxi1).software.vib.list() | Sort-Object InstallDate -Descending | Select-Object -First 1).InstallDate
\$patchAge = (New-TimeSpan -Start \$lastPatchDate -End (Get-Date)).TotalDays
\$patchAge**

Review Nessus Vulnerability Scan Details

Navigate and set the location of the Nessus files
Set-Location C:\user\Desktop\2021Scans

View what files exist in the directory
Get-ChildItem

Let’s assume there are a lot of Nessus files to process, save them to a variable
\$scanResults = Import-Csv -path (Get-ChildItem *.csv | Select-Object -ExpandProperty FullName)

Group the results by Risk
**\$scanResults | Group-Object Risk
\$scanResults | Group-Object Risk | Where-Object Name -eq 'Critical'**

Identify hosts with largest numbers of critical vulnerabilities
\$scanResults | Where-Object Risk -eq 'critical' | Group-Object Host | Select-Object Count, Name | Where-Object Count -gt 5 | Sort-Object Count -Descending

Identify percent of vulnerabilities marked as critical
**\$criticalCount = (\$scanResults | Group-Object Risk | Where-Object Name -eq 'Critical').Count
\$totalCount = (\$scanResults | Where-Object Risk -ne 'None').Count
\$criticalCount/\$totalCount**