

Cumplimiento de parches en Windows

Información sobre el parche

Get-Hotfix

Patch-Velocity

Cuenta el número de parches aplicados por día

Get-Hotfix | Sort-Object InstalledOn -Descending

Patch-Age

La edad del parche de un sistema es el número de días desde que se aplicó el último parche:

```
$lastPatchDate = (Get-HotFix | Sort-Object InstalledOn -Descending | Select-Object -First 1).InstalledOn  
$lastPatchDate  
(New-TimeSpan -Start $lastPatchDate -End (Get-Date)).TotalDays
```

Cumplimiento de parches en distribuciones Linux basadas en Debian Distribution

Cambiar el shell a Powershell Core

Pwsh

Los parches instalados por el gestor de paquetes Apt se registran:

/var/log/dkpg.log

Patch-Velocity

Cuenta el número de parches aplicados por día

```
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

La edad del parche de un sistema es el número de días desde que se aplicó el último parche:

```
$lastPatchDate = ($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"

Medidas de cumplimiento de Windows

Compruebe que la cuenta de administrador está desactivada:

Get-LocalUser -Name Administrator

Compruebe que la cuenta de invitado está desactivada:

Get-LocalUser -Name Guest

Guarde la lista de usuarios locales en una variable y luego compruebe si tanto el invitado como el administrador están desactivados:

```
$disabledUsers = Get-LocalUser | Where-Object Enabled -eq $False  
($disabledUsers.Name -contains 'Administrator') -And  
($disabledUsers.Name -contains 'Guest')
```

Enumerar los miembros de los grupos locales:

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

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

Compruebe que los servicios de Windows están instalados, habilitados y en funcionamiento:

```
((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*')
```

Todos los comandos, a menos que se indique lo contrario, han sido probados en el curso VMs **SEC557: Continuous Automation for Enterprise and Cloud Compliance** usando PowerShell Core.

Plan de estudios de liderazgo en ciberseguridad de SANS

The screenshot shows the SANS Cybersecurity Leadership study plan page. It features a sidebar with links to 'sans.org/cybersecurity-leadership', 'SANS Security Leadership' on LinkedIn, '@seleadership' on Twitter, 'Recommended Reading', 'Webcasts', and 'Blogs'. The main content area is divided into two columns. The left column lists courses under the heading 'AUD507: Auditing & Monitoring Networks, Perimeters & Systems' and 'LEG523: Law of Data Security and Investigation'. The right column lists courses under 'MGT641: SANS Training Program for the CISSP® Certification', 'MGT645: A Practical Introduction to Cyber Security Risk Management', 'MGT643: Managing Human Risk: Mature Security Awareness Programs', 'MGT512: Security Leadership Essentials for Managers', 'MGT514: Security Strategic Planning, Policy, and Leadership', 'MGT516: Managing Security Vulnerabilities: Enterprise & Cloud', and 'SEC440: CIS Critical Controls: A Practical Introduction to CIS Critical Security'. Each course entry includes a small circular icon with a logo and a brief description.



SANS
CLOUD SECURITY

POWERSHELL PARA EL CUMPLIMIENTO DE LA EMPRESA Y LA NUBE

Por AJ Yawn
Cheat Sheet v1.0.0

SANS.ORG/CLOUD-SECURITY

SANS.ORG/SEC557

Plan de estudios de seguridad en la nube de SANS

The screenshot shows the SANS Cloud Security study plan page. It features a sidebar with links to 'sans.org/cloud-security', 'SANS Cloud Security' on YouTube, '@SANSCloudSec' on Twitter, 'SANS Cloud Security' on LinkedIn, 'Webcasts', and 'Blogs'. The main content area is divided into two columns. The left column lists courses under the heading 'SEC488: Cloud Security Essentials', 'SEC510: Public Cloud Security: AWS, Azure, and GCP', 'SEC584: Cloud Native Security: Defending Containers and Kubernetes', and 'SEC522: Defending Web Applications Security Essentials'. The right column lists courses under 'MGT520: Leading Cloud Security Design & Implementation', 'MGT521: Leading Cybersecurity Change: Building A Security-Based Culture', 'MGT525: Project Management & Effective Communication', 'MGT526: Building and Leading Security Operations Center', 'SEC440: CIS Critical Controls: A Practical Introduction to CIS Critical Security', 'SEC474: Building A Healthcare Security & Compliance Program', 'SEC557: Continuous Automation for Enterprise & Cloud Compliance', 'SEC558: Secure DevOps: A Practical Introduction', 'SEC534: Cloud Security and DevSecOps Automation', 'SEC541: Cloud Monitoring and Threat Detection', and 'MGT520: Leading Cloud Security Design & Implementation'. Each course entry includes a small circular icon with a logo and a brief description.

Review our Job Role Flight Plan at sans.org/cloud-security

Medidas de conformidad de AWS

Asegúrese de que la versión actual del módulo AWS PowerShell está disponible para su uso.

```
Install-Module -name AWSPowerShell.NetCore -Scope CurrentUser -Force
```

Cargar el módulo AWS

```
Import-Module AWSPowerShell.NetCore
```

Autenticación en 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
```

Medir la configuración del host VMWare

Recopilar información sobre el sistema anfitrión VMWare y su configuración
Get-VMHost -Server <name>

Validar la configuración común del hipervisor

(Get-VMHost).ExtensionData

Aproveche la propiedad Config de ExtensionData para obtener los ajustes de configuración en profundidad (ejemplo de configuración de resolución DNS a continuación)

(Get-VMHost).ExtensionData.Config.Network.DNSConfig Mide si los ajustes de DNS están configurados correctamente:

```
$dnsservers = (Get-VMHost).ExtensionData.Config.Network.DNSConfig | Select-Object -ExpandProperty address  
$dnsservers -contains '8.8.8.8'  
$dnsservers -contains '8.8.4.4'
```

Valide el servidor o servidores NTP configurados en el host VMWare:
Get-VMHost -Server <name> | Get-VMHostNtpServer

Validar que el servicio NTP se está ejecutando y está configurado para ejecutarse al inicio

```
Get-VMHost | Get-VMHostService | Where-Object {$_.key -eq "ntpd"} | Select-Object VMHost, Label, Key, Policy, Running, Required
```

Datos del parche

(Get-ESXcli -Server esxi1).software.vib.list()

Velocidad del parche

```
(Get-ESXcli -Server esxi1).software.vib.list() | Group-Object InstallDate
```

Edad del parche

```
$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

Revise los detalles de la exploración de vulnerabilidades de Nessus

Navegue y establezca la ubicación de los archivos de Nessus

```
Set-Location C:\user\Desktop\2021Scans
```

Ver qué archivos existen en el directorio

```
Get-ChildItem
```

Supongamos que hay muchos archivos Nessus para procesar, guárdalos en una variable

```
$scanResults = Import-Csv -path (Get-ChildItem *.csv) | Select-Object -ExpandProperty FullName
```

Agrupar los resultados por Riesgo

```
$scanResults | Group-Object Risk
```

```
$scanResults | Group-Object Risk | Where-Object Name -eq 'Critical'
```

Identificar los hosts con mayor número de vulnerabilidades crítica

```
$scanResults | Where-Object Risk -eq 'critical' | Group-Object Host | Select-Object Count, Name | Where-Object Count -gt 5 | Sort-Object Count -Descending
```

Identificar el porcentaje de vulnerabilidades marcadas como críticas

```
$criticalCount = ($scanResults | Group-Object Risk | Where-Object Name -eq 'Critical').Count
```

```
$totalCount = ($scanResults | Where-Object Risk -ne 'None').Count
```

```
$criticalCount/$totalCount
```