Referencia para comprobaciones de compatibilidad con Nessus

10 de enero de 2014

(Revisión 39)

Índice

Introducción

Este documento describe la sintaxis usada para crear archivos **.audit** personalizados que se pueden usar para auditar la configuración de sistemas de Unix, Windows, bases de datos, SCADA, IBM iSeries y Cisco respecto de una directiva de compatibilidad, así como examinar distintos sistemas en busca de contenido confidencial.

La presente guía tiene como fin asistir en la creación manual y la comprensión de la sintaxis de archivos de auditoría de compatibilidad. Consulte el documento PDF Nessus Compliance Checks (Comprobaciones de compatibilidad con Nessus), disponible en el Tenable Support Portal (Portal de soporte de Tenable), para conocer en mayor profundidad la forma en que funcionan las comprobaciones de compatibilidad de Tenable.

Nessus admite auditorías de sistemas SCADA. No obstante, esta función se encuentra fuera del alcance del presente documento. Consulte la página de información Tenable SCADA aquí para obtener más información.

Requisitos previos

Este documento supone cierto nivel de conocimiento sobre el analizador de vulnerabilidades Nessus, así como una comprensión profunda de los sistemas de destino a los que se les realiza la auditoría. Para obtener más información sobre cómo Nessus puede configurarse para realizar auditorías de revisiones locales para Unix y Windows, consulte el documento “Nessus Credentials Checks for Unix and Windows” (Comprobaciones con credenciales de Nessus para Unix y Windows), que puede encontrar en http://www.tenable.com/products/nessus/documentation.

Estándares y convenciones

En toda la documentación, los nombres de archivo, demonios y archivos ejecutables se indican con fuente **courier negrita**.

Las opciones de líneas de comandos y las palabras clave también se indican con fuente **courier negrita**. Los ejemplos de líneas de comandos pueden incluir o no el indicador de la línea de comandos y el texto de salida de los resultados del comando. Los ejemplos de líneas de comandos mostrarán el comando ejecutado en **courier negrita** para indicar lo que el usuario escribió, mientras que el resultado de muestra generado por el sistema se indicará en courier (normal). Este es un ejemplo de ejecución del comando **pwd** de Unix:

```
# pwd
/home/test/
#
```

Las consideraciones y notas importantes se resaltan con este símbolo y cuadros de texto grises.

Las sugerencias, los ejemplos y las prácticas recomendadas se resaltan con este símbolo y con letras blancas en cuadros de texto azules.

Referencia para archivos de compatibilidad de auditoría de configuración de Windows

La base para los archivos de compatibilidad **.audit** de Windows consiste en un archivo de texto con formato especial. Las entradas del archivo pueden invocar una variedad de comprobaciones de “elementos personalizados”, tales como

comprobaciones de opciones del registro, así como también comprobaciones más generales como las de configuración de directivas de seguridad locales. Los ejemplos que se proporcionan en toda la guía tienen fines de clarificación.

Uso de comillas:

Las comillas simples y dobles son indistintas al encerrar campos de auditoría, a excepción de los siguientes dos casos:

1. En comprobaciones de compatibilidad de Windows en las que se deban interpretar de forma literal los campos especiales como CRLF, etc., se deben usar comillas simples. Se deben escapar los campos incrustados que se interpretarán como cadenas.

Por ejemplo:
```
expect: 'First line\r\nSecond line\r\nJohn\'s Line'
```

2. Se requieren comillas dobles al utilizar “include_paths” y “exclude_paths” de WindowsFiles.

Si en cualquier tipo de campo (descripción, value_data, regex, etc.) se usan cadenas que contengan comillas simples o dobles, existen dos formas de tratarlas:

a. Use el tipo de comilla opuesto para las comillas de cierre exteriores.

Por ejemplo:
```
expect: "This is John's Line"
expect: 'We are looking for a double-quote-".*'
```

b. Escape las comillas incrustadas, si las hay, con una barra inversa (solo comillas dobles).

Por ejemplo:
```
expect: "\"Text to be searched\""
```

Tipo de comprobación

Todas las comprobaciones de compatibilidad de Windows deben estar entre corchetes con la encapsulación **`check_type`** y la designación “Windows”. También se debe especificar la versión “2”:

```
<check_type:"Windows" version:"2">
```

Un ejemplo de comprobación de compatibilidad de Windows puede observarse en el “Anexo B”; comienza con la opción **`check_type`** para “Windows” y la versión “2”, y finaliza con la etiqueta “**`</check_type>`**”.

Esto es obligatorio para diferenciar los archivos **`.audit`** de Windows de los diseñados para Unix (u otras plataformas).

Datos de valores

La sintaxis del archivo **`.audit`** contiene palabras clave a las que usted puede asignar distintos tipos de valores para personalizar sus comprobaciones. Esta sección describe estas palabras clave y el formato de los datos que se pueden introducir.

Tipos de datos

Para las comprobaciones se pueden introducir los siguientes tipos de datos:

Tipo de datos	Descripción
DWORD	De 0 a 2 147 483 647
RANGE [X..Y]	Donde X representa DWORD o MIN, e Y representa DWORD o MAX

Ejemplos:

```
value_data: 45
value_data: [11..9841]
value_data: [45..MAX]
```

Además, los números se pueden especificar mediante un más (+) o un menos (-) para indicar su "signo", y se pueden especificar como valores hexadecimales. Se pueden combinar valores hexadecimales y signos. Los siguientes constituyen ejemplos válidos (sin la correspondiente etiqueta entre paréntesis) en una auditoría REGISTRY_SETTING para POLICY_DWORD:

```
value_data: -1   (signed)
value_data: +10   (signed)
value_data: 10   (unsigned)
value_data: 2401649476   (unsigned)
value_data: [MIN..+10]   (signed range)
value_data: [20..MAX]   (unsigned range)
value_data: 0x800010AB   (unsigned hex)
value_data: -0x10   (signed hex)
```

Expresiones complejas

Para el campo **`value_data`**, se pueden emplear expresiones complejas, mediante:

- ||: OR condicional
- &&: AND condicional
- |: OR binario (operación de bits)
- &: AND binario (operación de bits)
- (and): para delimitar expresiones complejas

Ejemplos:

```
value_data: 45 || 10
value_data: (45 || 10) && ([9..12] || 37)
```

Campo "check_type"

Este "check type" (tipo de comprobación) es diferente del campo "**`check_type`**" que se especificó anteriormente, y que se usa al comienzo de cada archivo de auditoría para denotar el tipo de auditoría general (Windows, WindowsFiles, Unix, Database, Cisco). Es opcional, y se puede realizar con valores **`value_data`** de Windows para determinar el tipo de comprobación que se llevará a cabo. Se encuentran disponibles las siguientes opciones:

- CHECK_EQUAL: compara el valor remoto con el valor de la directiva (de manera predeterminada si falta **`check_type`**)
- CHECK_EQUAL_ANY: comprueba que cada elemento de **`value_data`** se encuentre al menos una vez en la lista del sistema
- CHECK_NOT_EQUAL: comprueba que el valor remoto sea diferente del valor de la directiva
- CHECK_GREATER_THAN: comprueba que el valor remoto sea mayor que el valor de la directiva
- CHECK_GREATER_THAN_OR_EQUAL: comprueba que el valor remoto sea mayor o igual que el valor de la directiva
- CHECK_LESS_THAN: comprueba que el valor remoto sea menor que el valor de la directiva
- CHECK_LESS_THAN_OR_EQUAL: comprueba que el valor remoto sea menor o igual que el valor de la directiva
- CHECK_REGEX: comprueba que el valor remoto coincida con el regex en el valor de la directiva (solo funciona con POLICY_TEXT y POLICY_MULTI_TEXT)
- CHECK_SUBSET: comprueba que la ACL remota sea un subconjunto de la ACL de la directiva (solo funciona con las ACL)
- CHECK_SUPERSET: comprueba que la ACL remota sea un superconjunto de la ACL de la directiva (solo funciona con las ACL que deniegan permisos)

A continuación se incluye un ejemplo de auditoría para comprobar que el nombre de la cuenta “Guest” (Invitado) no existe para ninguna cuenta Guest.

```
<custom_item>
 type: CHECK_ACCOUNT
 description: "Accounts: Rename guest account"
 value_type: POLICY_TEXT
 value_data: "Guest"
 account_type: GUEST_ACCOUNT
 check_type: CHECK_NOT_EQUAL
</custom_item>
```

Si se encuentra cualquier valor que no sea “Guest” (Invitado), se superará la prueba. Si se encuentra “Guest” (Invitado), habrá un error en la auditoría.

El campo “group_policy”

El campo “**`group_policy`**” puede utilizarse para proporcionar una cadena de texto corta que describa la auditoría. El campo **`group_policy`** debe incluirse en el archivo de la auditoría, y debe introducirse después del campo **`check_type`**.

```
<check_type: "Windows" version:"2">
<group_policy: "Audit file for Windows 2008">

…

</group_policy>
</check_type>
```

El campo “info”

El campo “**info**” opcional se puede usar para etiquetar cada campo de auditoría con una o más referencias externas. Por ejemplo, este campo se usará para incluir referencias de etiquetas NIST CCE, así como también requisitos específicos de auditoría del CIS. Estas referencias externas se imprimen en la auditoría final realizada por Nessus, y aparecerán en el informe de Nessus o a través de la interfaz de usuario de SecurityCenter.

A continuación se incluye un ejemplo de directiva de auditoría de contraseñas que se aumentó para enumerar las referencias a una directiva corporativa ficticia:

```
<custom_item>
 type: PASSWORD_POLICY
 description: "Password History: 24 passwords remembered"
 value_type: POLICY_DWORD
 value_data: [22..MAX] || 20
 password_policy: ENFORCE_PASSWORD_HISTORY
 info: "Corporate Policy 102-A"
</custom_item>
```

Si para una única auditoría se requieren varias referencias de directivas, la cadena especificada por la palabra clave “**info**” puede usar el separador “\n” para especificar varias cadenas. Por ejemplo, considere la siguiente auditoría:

```
<custom_item>
 type: CHECK_ACCOUNT
 description: "Accounts: Rename Administrator account"
 value_type: POLICY_TEXT
 value_data: "Administrator"
 account_type: ADMINISTRATOR_ACCOUNT
 check_type: CHECK_NOT_EQUAL
 info: 'Ron Gula Mambo Number 5\nCCE-60\nTenable Best Practices Policy 1005-a'
</custom_item>
```

Al ejecutarse con la herramienta de líneas de comandos **nasl**, esta función de auditoría produce los siguientes resultados:

```
# /opt/nessus/bin/nasl -t 192.168.20.16 ./compliance_check.nbin

Windows Compliance Checks, version 2.0.0

Which file contains your security policy : ./test_v2.audit
SMB login : Administrator
SMB password :
SMB domain (optional) :
"Accounts: Rename Administrator account": [FAILED]

Ron Gula Mambo Number 5
CCE-60
Tenable Best Practices Policy 1005-a

Remote value: "Administrator"
Policy value: "administrator"
```

El campo “debug”

El campo opcional “**debug**” puede utilizarse para reparar comprobaciones de compatibilidad de contenido en Windows. La palabra clave “debug” arroja información sobre el análisis de contenido en ejecución, como el/los archivo/s en

procesamiento, los analizados, y si se encontró algún resultado. Debido a la gran cantidad de resultados, esta palabra clave solo se debe utilizar para fines de solución de problemas. Por ejemplo:

```
 <item>
  debug
  type: FILE_CONTENT_CHECK
  description: "TNS File that Contains the word Nessus"
  file_extension: "tns"
  expect: "Nessus"
 </item>
```

Formato de ACL

Esta sección describe la sintaxis empleada para determinar si un archivo o una carpeta poseen la configuración ACL deseada.

Comprobaciones de control de acceso a archivos

Uso

`<file_acl: ["name"]>`
 	
`<user: ["user_name"]>`
`acl_inheritance: ["value"]`
`acl_apply: ["value"]`
`(optional) acl_allow: ["rights value"]`
`(optional) acl_deny: ["rights value"]`
`</user>`
 	
`</acl>`

Las Access Control Lists (Listas de control de acceso [ACL]) a archivos se identifican mediante la palabra clave **`file_acl`**. El nombre de la ACL debe ser exclusivo para usarse con un elemento de permisos de archivos. Una ACL a archivos puede contener una o varias entradas de usuario.

Tipos asociados	Tipos permitidos
`acl_inheritance`	• not inherited (no heredado) • inherited (heredado) • not used (no usado)
`acl_apply`	• this folder only (esta carpeta únicamente) • this object only (solo este objeto) • this folder and files (esta carpeta y archivos) • this folder and subfolders (esta carpeta y subcarpetas) • this folder, subfolders and files (esta carpeta, subcarpetas y archivos) • files only (archivos únicamente) • subfolders only (subcarpetas únicamente) • subfolders and files only (subcarpetas y archivos únicamente)
`acl_allow` **`acl_deny`**	Las siguientes configuraciones son opcionales. Los permisos generales son los siguientes: • full control (control total)

	• modify (modificación) • read & execute (lectura y ejecución) • read (lectura) • write (escritura) • list folder contents (enumeración de contenido de carpeta) Los permisos avanzados son los siguientes: • full control (control total) • traverse folder / execute file (recorrido de carpeta/ejecución de archivos) • list folder / read data (enumeración de carpetas/lectura de datos) • read attributes (lectura de atributos) • read extended attributes (lectura de atributos extendidos) • create files / write data (creación de archivos/escritura de datos) • create folders / append data (creación de carpetas/anexo de datos) • write attributes (escritura de atributos) • write extended attributes (escritura de atributos extendidos) • delete subfolder and files (eliminación de subcarpetas y archivos) • delete (eliminación) • read permissions (lectura de permisos) • change permissions (cambio de permisos) • take ownership (asunción de propiedad)

A continuación se incluye un ejemplo de texto **`.audit`** para controles de acceso a archivos:

```
<file_acl: "ASU1">

 <user: "Administrators">
  acl_inheritance: "not inherited"
  acl_apply: "This folder, subfolders and files"
  acl_allow: "Full Control"
 </user>

 <user: "System">
  acl_inheritance: "not inherited"
  acl_apply: "This folder, subfolders and files"
  acl_allow: "Full Control"
 </user>

 <user: "Users">
  acl_inheritance: "not inherited"
  acl_apply: "this folder only"
  acl_allow: "list folder / read data" | "read attributes" | "read extended
       attributes" | "create files / write data" | "create folders / append data" |
       "write attributes" | "write extended attributes" | "read permissions"
 </user>

</acl>
```

Comprobaciones de control de acceso al registro

Uso

`<registry_acl: ["name"]>`
 	
`<user: ["user_name"]>`
`acl_inheritance: ["value"]`
`acl_apply: ["value"]`
`(optional) acl_allow: ["rights value"]`
`(optional) acl_deny: ["rights value"]`
`</user>`
 	
`</acl>`

Una ACL al registro se identifica mediante la palabra clave **`registry_acl`**. El nombre de la ACL debe ser exclusivo para usarse con un elemento de permisos del registro. Una ACL al registro puede contener una o varias entradas de usuario.

Tipos asociados	Tipos permitidos
`acl_inheritance`	• not inherited (no heredado) • inherited (heredado) • not used (no usado)
`acl_apply`	• this key only (solo esta clave) • this key and subkeys (esta clave y subclaves) • subkeys only (solo subclaves)
`acl_allow` **`acl_deny`**	Esta configuración es opcional y se usa para definir los permisos que un usuario tiene respecto del objeto. Los permisos generales son los siguientes: • full control (control total) • read (lectura) Los permisos avanzados son los siguientes: • full control (control total) • query value (consulta de valores) • set value (establecimiento de valores) • create subkey (creación de subclave) • enumerate subkeys (enumeración de subclaves) • notify (notificación) • create link (creación de enlace) • delete (eliminación) • write dac (escritura de dac) • write owner (propietario de escritura) • read control (control de lectura)

A continuación se incluye un ejemplo de texto **`.audit`** para lista de controles de acceso al registro:

```
 <registry_acl: "SOFTWARE ACL">

  <user: "Administrators">
   acl_inheritance: "not inherited"
   acl_apply: "This key and subkeys"
   acl_allow: "Full Control"
  </user>

  <user: "CREATOR OWNER">
   acl_inheritance: "not inherited"
   acl_apply: "Subkeys only"
   acl_allow: "Full Control"
  </user>

  <user: "SYSTEM">
   acl_inheritance: "not inherited"
   acl_apply: "This key and subkeys"
   acl_allow: "Full Control"
  </user>

  <user: "Users">
   acl_inheritance: "not inherited"
   acl_apply: "This key and subkeys"
   acl_allow: "Read"
  </user>

 </acl>
```

Comprobaciones de control de acceso a servicios

Uso

`<service_acl: ["name"]>`
 	
`<user: ["user_name"]>`
`acl_inheritance: ["value"]`
`acl_apply: ["value"]`
`(optional) acl_allow: ["rights value"]`
`(optional) acl_deny: ["rights value"]`
`</user>`
 	
`</acl>`

Una ACL a servicios se identifica mediante la palabra clave **`service_acl`**. El nombre de la ACL debe ser exclusivo para usarse con un elemento de permisos de servicios. Una ACL a servicios puede contener una o varias entradas de usuario.

Tipos asociados	Tipos permitidos
`acl_inheritance`	• not inherited (no heredado) • inherited (heredado) • not used (no usado)

acl_apply	• this object only (solo este objeto)
acl_allow **acl_deny**	Esta configuración es opcional y se usa para definir los permisos que un usuario tiene respecto del objeto. Los permisos generales son los siguientes: • full control (control total) • read (lectura) • start, stop and pause (inicio, detención y pausa) • write (escritura) • delete (eliminación) Los permisos avanzados son los siguientes: • full control (control total) • delete (eliminación) • query template (plantilla de consulta) • change template (plantilla de cambios) • query status (estado de consulta) • enumerate dependents (enumeración de dependientes) • start (inicio) • stop (detención) • pause and continue (pausa y continuación) • interrogate (interrogación) • user-defined control (control definido por el usuario) • read permissions (lectura de permisos) • change permissions (cambio de permisos) • take ownership (asunción de propiedad)

A continuación se indica un ejemplo de comprobación de control de acceso a servicios:

```
<service_acl: "ALERT ACL">

 <user: "Administrators">
  acl_inheritance: "not inherited"
  acl_apply: "This object only"
  acl_allow: "query template" | "change template" | "query status" | "enumerate
       dependents" | "start" | "stop" | "pause and continue" | "interrogate" | "user-
       defined control" | "delete" | "read permissions" | "change permissions" | "take
       ownership"
 </user>

</acl>
```

Comprobaciones de control para permisos de inicio

Uso

`<launch_acl: ["name"]>`
 	
`<user: ["user_name"]>`

```
  acl_inheritance: ["value"]
  acl_apply: ["value"]
  (optional) acl_allow: ["rights value"]
  (optional) acl_deny: ["rights value"]
 </user>

</acl>
```

Una launch ACL se identifica mediante la palabra clave **`launch_acl`**. El nombre de la ACL debe ser exclusivo para usarse con un elemento de permisos para inicio DCOM. Una launch ACL puede contener una o varias entradas de usuario.

Tipos asociados	Tipos permitidos
`acl_inheritance`	• not inherited (no heredado) • inherited (heredado)
`acl_apply`	• this object only (solo este objeto)
`acl_allow` **`acl_deny`**	Esta configuración es opcional y se usa para definir los permisos que un usuario tiene respecto del objeto. Los permisos generales son los siguientes: • local launch (inicio local) • remote launch (inicio remoto) • local activation (activación local) • remote activation (activación remota)

Esta ACL solo funciona en Windows XP/2003/Vista (y de manera parcial en Windows 2000).

A continuación se indica un ejemplo de comprobación de control de acceso a launch:

```
 <launch_acl: "2">

  <user: "Administrators">
   acl_inheritance: "not inherited"
   acl_apply: "This object only"
   acl_allow: "Remote Activation"
  </user>

  <user: "INTERACTIVE">
   acl_inheritance: "not inherited"
   acl_apply: "This object only"
   acl_allow: "Local Activation" | "Local Launch"
  </user>

  <user: "SYSTEM">
   acl_inheritance: "not inherited"
   acl_apply: "This object only"
   acl_allow: "Local Activation" | "Local Launch"
  </user>
```

```
</acl>
```

Comprobaciones de control para permisos de Launch2

Uso

`<launch2_acl: ["name"]>`
 	
`<user: ["user_name"]>`
`acl_inheritance: ["value"]`
`acl_apply: ["value"]`
`(optional) acl_allow: ["rights value"]`
`(optional) acl_deny: ["rights value"]`
`</user>`
 	
`</acl>`

Una launch2 ACL se identifica mediante la palabra clave **`launch2_acl`**. El nombre de la ACL debe ser exclusivo para usarse con un elemento de permisos para inicio DCOM. Una launch2 ACL puede contener una o varias entradas de usuario.

Tipos asociados	Tipos permitidos
`acl_inheritance`	• not inherited (no heredado) • inherited (heredado)
`acl_apply`	• this object only (solo este objeto)
`acl_allow` **`acl_deny`**	Esta configuración es opcional y se usa para definir los permisos que un usuario tiene respecto del objeto. Los permisos generales son los siguientes: • launch (inicio)

Use la launch2 ACL solo en sistemas Windows 2000 y NT.

A continuación se indica un ejemplo de comprobación de control de acceso a launch:

```
<launch2_acl: "2">

 <user: "Administrators">
  acl_inheritance: "not inherited"
  acl_apply: "This object only"
  acl_allow: "Launch"
 </user>

 <user: "INTERACTIVE">
```

```
   acl_inheritance: "not inherited"
   acl_apply: "This object only"
   acl_allow: "Launch"
  </user>

  <user: "SYSTEM">
   acl_inheritance: "not inherited"
   acl_apply: "This object only"
   acl_allow: "Launch"
  </user>

 </acl>
```

Comprobaciones de control para permisos de acceso

Uso

`<access_acl: ["name"]>`
 	
` <user: ["user_name"]>`
`  acl_inheritance: ["value"]`
`  acl_apply: ["value"]`
`  (optional) acl_allow: ["rights value"]`
`  (optional) acl_deny: ["rights value"]`
` </user>`
 	
`</acl>`

Una ACL de acceso se identifica mediante la palabra clave **`access_acl`**. El nombre de la ACL debe ser exclusivo para usarse con un elemento de permisos de acceso DCOM. Una ACL de acceso puede contener una o varias entradas de usuario.

Tipos asociados	Tipos permitidos
`acl_inheritance`	• not inherited (no heredado) • inherited (heredado)
`acl_apply`	• this object only (solo este objeto)
`acl_allow` **`acl_deny`**	Esta configuración es opcional y se usa para definir los permisos que un usuario tiene respecto del objeto. Los permisos generales son los siguientes: • local access (acceso local) • remote access (acceso remoto)

A continuación se indica un ejemplo de comprobación de control de acceso:

```
 <access_acl: "3">
```

```
  <user: "SELF">
   acl_inheritance: "not inherited"
   acl_apply: "This object only"
   acl_allow: "Local Access"
  </user>

  <user: "SYSTEM">
   acl_inheritance: "not inherited"
   acl_apply: "This object only"
   acl_allow: "Local Access"
  </user>

  <user: "Users">
   acl_inheritance: "not inherited"
   acl_apply: "This object only"
   acl_allow: "Local Access"
  </user>

 </acl>
```

Elementos personalizados

Un elemento personalizado constituye una comprobación completa establecida en función de las palabras clave definidas anteriormente. La siguiente es una lista de tipos de elementos personalizados que se encuentran disponibles. Cada comprobación comienza con una etiqueta "**<custom_item>**" y finaliza con "**</custom_item>**". Entre las etiquetas se encuentran las listas de una o más palabras clave que son interpretadas por el analizador sintáctico de comprobaciones de compatibilidad para llevar a cabo dichas comprobaciones.

Las comprobaciones de auditoría personalizadas pueden usar "**</custom_item>**" y "**</item>**" de manera indistinta para la etiqueta de cierre.

PASSWORD_POLICY

Uso

```
<custom_item>
 type: PASSWORD_POLICY
 description: ["description"]
 value_type: [VALUE_TYPE]
 value_data: [value]
 (optional) check_type: [value]
 password_policy: [PASSWORD_POLICY_TYPE]
</custom_item>
```

Este elemento de directiva comprueba los valores definidos en "Windows Settings (Configuración de Windows) -> Security Settings (Configuración de seguridad) -> Account Policies (Directivas de cuenta) -> Password Policy (Directiva de contraseñas)".

La comprobación se lleva a cabo empleando la función **NetUserModalsGet** con el nivel 1.

Estos elementos usan el campo **password_policy** para describir qué elemento de la directiva de contraseñas se debe auditar. Los tipos permitidos son los siguientes:

- ENFORCE_PASSWORD_HISTORY (“Exigir historial de contraseñas”)
  ```
  value_type: POLICY_DWORD
  value_data: DWORD or RANGE [number of remembered passwords]
  ```
- MAXIMUM_PASSWORD_AGE (“Vigencia máxima de la contraseña”)
  ```
  value_type: TIME_DAY
  value_data: DWORD or RANGE [time in days]
  ```
- MINIMUM_PASSWORD_AGE (“Vigencia mínima de la contraseña”)
  ```
  value_type: TIME_DAY
  value_data: DWORD or RANGE [time in days]
  ```
- MINIMUM_PASSWORD_LENGTH (“Longitud mínima de la contraseña”)
  ```
  value_type: POLICY_DWORD
  value_data: DWORD or RANGE [minimum number of characters in the password]
  ```
- COMPLEXITY_REQUIREMENTS (“La contraseña debe cumplir los requisitos de complejidad”)
  ```
  value_type: POLICY_SET
  value_data: "Enabled" or "Disabled"
  ```
- REVERSIBLE_ENCRYPTION (“Almacenar contraseñas mediante cifrado reversible para todos los usuarios del dominio”)
  ```
  value_type: POLICY_SET
  value_data: "Enabled" or "Disabled"
  ```
- FORCE_LOGOFF (“Seguridad de red: forzar el cierre de sesión cuando vencen las horas desde el inicio de sesión”)
  ```
  value_type: POLICY_SET
  value_data: "Enabled" or "Disabled"
  ```

Actualmente no existe ningún método para comprobar que existe la directiva “Store password using reversible encryption for all users in the domain” (“Almacenar contraseñas mediante cifrado reversible para todos los usuarios del dominio”).

La directiva FORCE_LOGOFF se encuentra en “Security Settings (Configuración de seguridad) -> Local Policies (Directivas locales) -> Security Options (Opciones de seguridad)”.

A continuación se incluye un ejemplo de auditoría de directiva de contraseñas:

```
<custom_item>
 type: PASSWORD_POLICY
 description: "Minimum password length"
 value_type: POLICY_DWORD
 value_data: 7
 password_policy: MINIMUM_PASSWORD_LENGTH
</custom_item>
```

LOCKOUT_POLICY

Uso

```
<custom_item>
 type: LOCKOUT_POLICY
 description: ["description"]
```

```
 value_type: [VALUE_TYPE]
 value_data: [value]
 (optional) check_type: [value]
 lockout_policy: [LOCKOUT_POLICY_TYPE]
</custom_item>
```

Este elemento de directiva comprueba los valores definidos en “Security Settings (Configuración de seguridad) -> Account Policies (Directivas de cuentas) -> Account Lockout Policy (Directiva de bloqueo de cuentas)”.

La comprobación se lleva a cabo empleando la función **`NetUserModalsGet`** con el nivel 3.

Este elemento usa el campo **`lockout_policy`** para describir qué elemento de la directiva de contraseñas se debe auditar. Los tipos permitidos son los siguientes:

- LOCKOUT_DURATION (“Duración de bloqueo de cuenta”)
  `value_type: TIME_MINUTE`
  `value_data: DWORD or RANGE [time in minutes]`
- LOCKOUT_THRESHOLD (“Umbral de bloqueo de cuenta”)
  `value_type: POLICY_DWORD`
  `value_data: DWORD or RANGE [time in days]`
- LOCKOUT_RESET (“Restablecer contador de bloqueo de cuenta después de”)
  `value_type: TIME_MINUTE`
  `value_data: DWORD or RANGE [time in minutes]`

A continuación se incluye un ejemplo:

```
 <custom_item>
  type: LOCKOUT_POLICY
  description: "Reset lockout account counter after"
  value_type: TIME_MINUTE
  value_data: 120
  lockout_policy: LOCKOUT_RESET
 </custom_item>
```

KERBEROS_POLICY

Uso

```
<custom_item>
 type: KERBEROS_POLICY
 description: ["description"]
 value_type: [VALUE_TYPE]
 value_data: [value]
 (optional) check_type: [value]
 kerberos_policy: [KERBEROS_POLICY_TYPE]
</custom_item>
```

Este elemento de directiva comprueba los valores definidos en “Security Settings (Configuración de seguridad) -> Account Policies (Directivas de cuentas) -> Kerberos Policy (Directiva Kerberos)”.

La comprobación se lleva a cabo empleando la función **`NetUserModalsGet`** con el nivel 1.

Este elemento usa el campo **`kerberos_policy`** para describir qué elemento de la directiva de contraseñas se debe auditar. Los tipos permitidos son los siguientes:

- USER_LOGON_RESTRICTIONS (“Aplicar restricciones de inicio de sesión de usuario”)
  ```
  value_type: POLICY_SET
  value_data: "Enabled" or "Disabled"
  ```
- SERVICE_TICKET_LIFETIME (“Vigencia máxima del ticket de servicio”)
  ```
  value_type: TIME_MINUTE
  value_data: DWORD or RANGE [time in minutes]
  ```
- USER_TICKET_LIFETIME (“Vigencia máxima del ticket de usuario”)
  ```
  value_type: TIME_HOUR
  value_data: DWORD or RANGE [time in hours]
  ```
- USER_TICKET_RENEWAL_LIFETIME (“Vigencia máxima del ticket de renovación de usuario”)
  ```
  value_type: TIME_DAY
  value_data: DWORD or RANGE [time in day]
  ```
- CLOCK_SYNCHRONIZATION_TOLERANCE (“Tolerancia máxima para la sincronización de los relojes de los equipos”)
  ```
  value_type: TIME_MINUTE
  value_data: DWORD or RANGE [time in minute]
  ```

La directiva Kerberos solo se puede comprobar en un Centro de distribución de claves (Key Distribution Center, KDC), que en Windows es normalmente un Domain Controller (Controlador de dominio).

Ejemplo:

```
<custom_item>
 type: KERBEROS_POLICY
 description: "Maximum lifetime for user renewal ticket"
 value_type: TIME_DAY
 value_data: 12
 kerberos_policy: USER_TICKET_RENEWAL_LIFETIME
</custom_item>
```

AUDIT_POLICY

Uso

```
<custom_item>
 type: AUDIT_POLICY
 description: ["description"]
 value_type: [VALUE_TYPE]
 value_data: [value]
 (optional) check_type: [value]
 audit_policy: [PASSWORD_POLICY_TYPE]
</custom_item>
```

Este elemento de directiva comprueba los valores definidos en “Security Settings (Configuración de seguridad) -> Local Policies (Directivas locales) -> Audit Policy (Directiva de auditoría)”.

La comprobación se lleva a cabo empleando la función **`LsaQueryInformationPolicy`** con el nivel **`PolicyAuditEventsInformation`**.

Este elemento usa el campo **`audit_policy`** para describir qué elemento de la directiva de contraseñas se debe auditar. Los tipos permitidos son los siguientes:

- AUDIT_ACCOUNT_LOGON (“Auditar sucesos de inicio de sesión de cuenta”)
- AUDIT_ACCOUNT_MANAGER (“Auditar la administración de cuentas”)
- AUDIT_DIRECTORY_SERVICE_ACCESS (“Auditar el acceso del servicio de directorio”)
- AUDIT_LOGON (“Auditar sucesos de inicio de sesión”)
- AUDIT_OBJECT_ACCESS (“Auditar el acceso a objetos”)
- AUDIT_POLICY_CHANGE (“Auditar el cambio de directivas”)
- AUDIT_PRIVILEGE_USE (“Auditar el uso de privilegios”)
- AUDIT_DETAILED_TRACKING (“Auditar el seguimiento de procesos”)
- AUDIT_SYSTEM (“Auditar sucesos del sistema”)

```
value_type: AUDIT_SET
value_data: "No auditing", "Success", "Failure", "Success, Failure"
```

Tenga en cuenta que existe un espacio obligatorio en “Success, Failure” (Sin errores, Error).

Ejemplo:

```
 <custom_item>
  type: AUDIT_POLICY
  description: "Audit policy change"
  value_type: AUDIT_SET
  value_data: "Failure"
  audit_policy: AUDIT_POLICY_CHANGE
 </custom_item>
```

AUDIT_POLICY_SUBCATEGORY

Uso

```
<custom_item>
 type: AUDIT_POLICY_SUBCATEGORY
 description: ["description"]
 value_type: [VALUE_TYPE]
 value_data: [value]
 (optional) check_type: [value]
 audit_policy_subcategory: [SUBCATEGORY_POLICY_TYPE]
</custom_item>
```

Este elemento de directiva comprueba los valores enumerados en auditpol /get /category:*.

La comprobación se lleva a cabo con la ejecución de cmd.exe auditpol / get/category:* mediante WMI.

Este elemento usa el campo **`audit_policy_subcategory`** para determinar qué subcategoría necesita auditarse. Los SUBCATEGORY_POLICY_TYPE permitidos son los siguientes:

- Security State Change (Cambio de estado de seguridad)
- Security System Extension (Extensión de sistema de seguridad)
- System Integrity (Integridad del sistema)
- IPsec Driver (Controlador IPsec)
- Other System Events (Otros eventos del sistema)
- Logon (Inicio de sesión)
- Logoff (Cierre de sesión)
- Account Lockout (Bloqueo de cuenta)
- IPsec Main Mode (Modo principal de IPsec)
- IPsec Quick Mode (Modo rápido de IPsec)
- IPsec Extended Mode (Modo extendido de IPsec)
- Special Logon (Inicio de sesión especial)
- Other Logon/Logoff Events (Otros eventos de inicio/cierre de sesión)
- Network Policy Server (Servidor de directivas de red)
- File System (Sistema de archivos)
- Registry (Registro)
- Kernel Object (Objeto de kernel)
- SAM (SAM)
- Certification Services (Servicios de certificación)
- Application Generated (Aplicación generada)
- Handle Manipulation (Manipulación de identificador)
- File Share (Recurso compartido de archivos)
- Filtering Platform Packet Drop (Colocación de paquetes de Filtering Platform)
- Filtering Platform Connection (Conexión de Filtering Platform)
- Other Object Access Events (Otros eventos de acceso a objetos)
- Sensitive Privilege Use (Uso de privilegio confidencial)
- Non Sensitive Privilege Use (Uso de privilegio no confidencial)
- Other Privilege Use Events (Otros eventos de uso de privilegio)
- Process Creation (Creación de proceso)
- Process Termination (Terminación de proceso)
- DPAPI Activity (Actividad DPAPI)
- RPC Events (Eventos RPC)
- Audit Policy Change (Cambio de directiva de auditoría)
- Authentication Policy Change (Cambio de directiva de autenticación)
- Authorization Policy Change (Cambio de directiva de autorización)
- MPSSVC Rule-Level Policy Change (Cambio de directiva del nivel de reglas de MPSSVC)
- Filtering Platform Policy Change (Cambio de directiva de Filtering Platform)
- Other Policy Change Events (Otros eventos de cambio de directiva)
- User Account Management (Administración de cuentas de usuario)
- Computer Account Management (Administración de cuentas de equipo)
- Security Group Management (Administración de grupos de seguridad)
- Distribution Group Management (Administración de grupos de distribución)
- Application Group Management (Administración de grupos de aplicación)
- Other Account Management Events (Otros eventos de administración de cuentas)
- Directory Service Access (Acceso a servicio de directorios)
- Directory Service Changes (Cambios de servicio de directorios)
- Directory Service Replication (Replicación de servicio de directorios)
- Detailed Directory Service Replication (Replicación de servicio de directorios detallada)
- Credential Validation (Validación de credenciales)

- Kerberos Service Ticket Operations (Operaciones de tickets de servicio de Kerberos)
- Other Account Logon Events (Otros eventos de cierre de sesión de cuenta)

```
value_type: AUDIT_SET
value_data: "No auditing", "Success", "Failure", "Success, Failure"
```

Tenga en cuenta que existe un espacio obligatorio en “Success, Failure” (Sin errores, Error).

Esta comprobación solo rige para Windows Vista/2008 Server y versiones posteriores. Si hay un firewall habilitado, entonces además de agregar WMI como excepción en la configuración del firewall, también se debe habilitar en dicha configuración “Windows Firewall : Allow inbound remote administration exception” (Firewall de Windows: permitir la excepción de administración remota entrante) usando **`gpedit.msc`**. Es posible que esta comprobación no funcione en sistemas Vista/2008 que no estén en inglés o en sistemas que no tengan instalado auditpol.

Ejemplo:

```
 <custom_item>
  type: AUDIT_POLICY_SUBCATEGORY
  description: "AUDIT Security State Change"
  value_type: AUDIT_SET
  value_data: "success, failure"
  audit_policy_subcategory: "Security State Change"
 </custom_item>
```

AUDIT_POWERSHELL

Uso

```
<custom_item>
 type: AUDIT_POWERSHELL
 description: "Powershell check"
 value_type: [value_type]
 value_data: [value]
 powershell_args: ["arguments for powershell.exe"]
 (optional) only_show_cmd_output: YES or NO
 (optional) check_type: [CHECK_TYPE]
 (optional) severity: ["HIGH" or "MEDIUM" or "LOW"]
 (optional) powershell_option: CAN_BE_NULL
 (optional) powershell console file: "C:\Program Files\Microsoft\Exchange
Server\ExShell.psc1"
</custom_item>
```

Esta comprobación ejecuta **`powershell.exe`** en el servidor remoto junto con los argumentos suministrados por “**`powershell_args`**” y da como resultado la salida del comando si “**`only_show_cmd_output`**” está establecida en YES (SÍ) o compara el resultado con “**`value_data`**” si se especifica **`value_data`**.

Tipos asociados:

Este elemento utiliza el campo “**`powershell_args`**” para especificar los argumentos que se deben suministrar a **`powershell.exe`**. Si la ubicación de powershell.exe no está predeterminada, debe utilizar la palabra clave **`powershell_console_file`** para especificar la ubicación. Actualmente, solo se admiten cmdlets “`get-`”. Por ejemplo:

- `get-hotfix | where-object {$_.hotfixid -ne 'File 1'} | select Description,HotFixID,InstalledBy | format-list`
- `get-wmiobject win32_service | select caption,name, state| format-list`
- `(get-WmiObject -namespace root\MicrosoftIISv2 -Class IIsWebService).ListWebServiceExtensions().Extensions`
- `get-wmiobject -namespace root\cimv2 -class win32_product | select Vendor,Name,Version | format-list`
- `get-wmiobject -namespace root\cimv2\power -class Win32_powerplan | select description,isactive | format-list`

El elemento utiliza el campo opcional "**`only_show_cmd_output`**" si la totalidad del resultado del comando debe notificarse:

- `only_show_cmd_output: YES` or `NO`

Otras consideraciones:

1. Si establece "**`only_show_cmd_output`**" y quiere definir la gravedad del resultado, puede utilizar la etiqueta de gravedad para cambiar el grado de gravedad. La opción predeterminada es INFO.
2. Powershell no está instalado de manera predeterminada en algunos sistemas operativos de Windows (por ejemplo, XP y 2003), y en dichos sistemas esta comprobación no dará ningún resultado. Por lo tanto, asegúrese de que Powershell esté instalado en el destino remoto antes de utilizar esta comprobación.
3. Para que esta comprobación funcione correctamente, debe habilitarse el servicio WMI. Además, configure el firewall en "Allow inbound remote administration exception" (Permitir excepción de administración remota entrante).
4. Los alias cmdlet (por ejemplo, "`gps`" en lugar de "`Get-Process`") no están permitidos.

Ejemplo:

Este ejemplo ejecuta el cmdlet powershell "Get-Hotfix", especifica que un where-object no seleccione hotfixes con la identificación "File 1" y luego notifica la Description, la HotfixID e Installedby, como una lista.

```
<custom_item>
 type: AUDIT_POWERSHELL
 description: "Show Installed Hotfix"
 value_type: POLICY_TEXT
 value_data: ""
 powershell_args: "get-hotfix | where-object {$_.hotfixid -ne 'File 1'} | select
       Description,HotFixID,InstalledBy | format-list"
 only_show_cmd_output: YES
</custom_item>
```

Ejemplo:

Este ejemplo comprueba si el servicio de Windows "WinRM" está en funcionamiento.

```
<custom_item>
 type: AUDIT_POWERSHELL
 description: "Check if WinRM service is running"
 value_type: POLICY_TEXT
 value_data: "Running"
 powershell_args: "get-wmiobject win32_service | where-object {$_.name -eq 'WinRM' -
```

```
        and $_.state -eq 'Running'} | select state"
  check_type: CHECK_REGEX
 </custom_item>
```

AUDIT_FILEHASH_POWERSHELL

Uso

```
<custom_item>
 type: AUDIT_FILEHASH_POWERSHELL
 description: "Powershell FileHash Check"
 value_type: POLICY_TEXT
 file: "[FILE]"
 value_data: "[FILE HASH]"
</custom_item>
```

Esta comprobación ejecuta **`powershell.exe`** en el servidor remoto con la información suministrada para comparar un hash de archivo esperado con el hash del archivo en el sistema.

Otras consideraciones:

- De manera predeterminada, se compara un hash MD5 del archivo; sin embargo, los usuarios pueden comparar hashes generados con los algoritmos SHA1, SHA256, SHA384, SHA512 o RIPEMD160.
- Para que la comprobación funcione, PowerShell debe estar instalado y WMI debe estar habilitado en el destino.

Ejemplo:

Este ejemplo compara un hash MD5 suministrado con el hash del archivo de `C:\test\test2.zip`.

```
 <custom_item>
  type: AUDIT_FILEHASH_POWERSHELL
  description: "Audit FILEHASH - MD5"
  value_type: POLICY_TEXT
  file: "C:\test\test2.zip"
  value_data: "8E653F7040AC4EA8E315E838CEA83A04"
 </custom_item>
```

Ejemplo:

Este ejemplo compara un hash SHA1 suministrado con el hash del archivo de `C:\test\test3.zip`.

```
 <custom_item>
  type: AUDIT_FILEHASH_POWERSHELL
  description: "Audit FILEHASH - SHA1"
  value_type: POLICY_TEXT
  file: "C:\test\test3.zip"
  value_data: "0C4B0AF91F62ECCED3B16D35DE50F66746D6F48F"
  hash_algorithm: SHA1
 </custom_item>
```

AUDIT_IIS_APPCMD

Uso

```
<custom_item>
 type: AUDIT_IIS_APPCMD
 description: "Test appcmd output"
 value_type: [value_type]
 value_data: [value]
 appcmd_args: ["arguments for appcmd.exe"]
 (optional) only_show_cmd_output: YES or NO
 (optional) check_type: [CHECK_TYPE]
 (optional) severity: ["HIGH" or "MEDIUM" or "LOW"]
</custom_item>
```

Esta comprobación ejecuta **appcmd.exe** en un servidor con IIS, junto con los argumentos especificados mediante "**appcmd_args**", y determina la compatibilidad comparando la salida con **value_data**. En algunos casos (por ejemplo, configuración de listas) puede ser recomendable notificar solo el resultado del comando. En dichos casos se debe usar "**only_show_cmd_output**".

Esta comprobación solo corresponde para la versión 7 de Internet Information Services (IIS) en Windows.

Este elemento utiliza el campo "**appcmd_args**" para especificar los argumentos que se deben suministrar a **appcmd.exe**. Actualmente, solo se pueden especificar los comandos "list".

- `list sites`
- `list AppPools /processModel.identityType:ApplicationPoolIdentity`
- `list config`
- `list config -section:system.web/authentication`
- `list app`

El elemento utiliza el campo opcional "**only_show_cmd_output**" si la totalidad del resultado del comando debe notificarse.

Ejemplo:

Esta comprobación compara el resultado de "`appcmd.exe list AppPools /processModel.identityType:ApplicationPoolIdentity`" con **value_data**, y solo pasa si la salida contiene 'APPPOOL "DefaultAppPool"'.

```
<custom_item>
 type: AUDIT_IIS_APPCMD
 description: "Set Default Application Pool Identity to Least Privilege Principal"
 value_type: POLICY_TEXT
 value_data: 'APPPOOL "DefaultAppPool"'
 appcmd_args: "list AppPools /processModel.identityType:ApplicationPoolIdentity"
 check_type: CHECK_REGEX
</custom_item>
```

AUDIT_ALLOWED_OPEN_PORTS

Uso

```
<custom_item>
 type: AUDIT_ALLOWED_OPEN_PORTS
 description: "Audit Open Ports"
 value_type: [value_type]
 value_data: [value]
 port_type: [port_type]
<item>
```

Esta comprobación hace una consulta de la lista de puertos TCP/UDP abiertos en el destino y los compara con una lista de puertos permitidos. La comprobación depende del resultado de “netstat –ano” o “netstat –an” para obtener una lista de puertos abiertos, y luego verifica que los puertos estén efectivamente abiertos comprobando el estado de los puertos con (get_port_state()/get_udp_port_state()).

Consideraciones:

- **value_data** también acepta una regex como rango de puertos, por lo que algo así como 8[0-9]+ también funciona.

Ejemplos:

El siguiente ejemplo compara “**value_data**” con una lista de puertos TCP abiertos en el destino:

```
 <custom_item>
  type: AUDIT_ALLOWED_OPEN_PORTS
  description: "Audit TCP OPEN PORTS"
  value_type: POLICY_PORTS
  value_data: "80,135,445,902,912,1024,1025,3389,5900,8[0-
       9]+,18208,32111,38311,47001,139"
  port_type: TCP
 </custom_item>
```

El siguiente ejemplo compara “**value_data**” con una lista de puertos UDP abiertos en el destino:

```
 <custom_item>
  type: AUDIT_ALLOWED_OPEN_PORTS
  description: "Audit UDP OPEN PORTS"
  value_type: POLICY_PORTS
  value_data: "161,445,500,1026,4501,123,137,138,5353"
  port_type: UDP
 </custom_item>
```

AUDIT_DENIED_OPEN_PORTS

Uso

```
<custom_item>
 type: AUDIT_DENIED_OPEN_PORTS
 description: "Audit Denied Open Ports"
```

```
 value_type: [value_type]
 value_data: [value]
 port_type: [port_type]
<item>
```

Esta comprobación hace una consulta de la lista de puertos TCP/UDP abiertos en el destino y los compara con una lista de puertos denegados. La comprobación depende del resultado de “netstat –ano” o “netstat –an” para obtener una lista de puertos abiertos, y luego verifica que los puertos estén efectivamente abiertos comprobando el estado de los puertos con (get_port_state()/get_udp_port_state()).

Los tipos permitidos son los siguientes:

- `value_type: POLICY_PORTS`
- `value_data: "80,135,445,902,912,1024,1025,3389,5900,8[0-9]+,18208,32111,38311,47001,139"`
- `port_type: TCP or UDP`

Consideraciones:

- **`value_data`** también acepta una regex como rango de puertos, por lo que algo así como 8[0-9]+ también funciona.

Ejemplos:

El siguiente ejemplo compara “**`value_data`**” con una lista de puertos TCP abiertos en el destino.

```
 <custom_item>
  type: AUDIT_DENIED_OPEN_PORTS
  description: "Audit TCP OPEN PORTS"
  value_type: POLICY_PORTS
  value_data: "80,443"
  port_type: TCP
 </custom_item>
```

El siguiente ejemplo compara “**`value_data`**” con una lista de puertos UDP abiertos en el destino.

```
 <custom_item>
  type: AUDIT_DENIED_OPEN_PORTS
  description: "Audit UDP OPEN PORTS"
  value_type: POLICY_PORTS
  value_data: "161,5353"
  port_type: UDP
 </custom_item>
```

AUDIT_PROCESS_ON_PORT

Uso

```
<custom_item>
 type: AUDIT_PROCESS_ON_PORT
 description: "Audit Process on Port"
 value_type: [value_type]
 value_data: [value]
```

```
 port_type: [port_type]
 port_no: [port_no]
 port_option: [port_option]
 check_type: CHECK_TYPE
<item>
```

Esta comprobación consulta el proceso que se está ejecutando en determinado puerto. La comprobación depende del resultado de “netstat -ano” y “tasklist /svc” para determinar qué proceso se está ejecutando en cada puerto TCP/UDP.

Los tipos permitidos son los siguientes:

- `value_type: POLICY_TEXT`
- `value_data:` cadena arbitraria, por ejemplo `"foo.exe"`
- `port_type: TCP` o `UDP`
- `port_no:` número de puerto, por ejemplo 80, 445
- `port_option: CAN_BE_CLOSED`

Consideraciones:

- Si **`port_option`** está establecido en CAN_BE_CLOSED, la comprobación devuelve un resultado PASS (Aprobado) si el puerto no está abierto en el sistema remoto; de lo contrario, genera un error.
- Windows 2000 y las versiones anteriores no admiten “netstat –ano”, por lo que esta comprobación solo funciona de Windows XP en adelante.

Ejemplos:

El siguiente ejemplo comprueba si el proceso en ejecución en el puerto TCP 5900 es “vss.exe” o “vssrvc.exe”.

```
<custom_item>
 type: AUDIT_PROCESS_ON_PORT
 description: "Audit OPEN PORT SERVICE"
 value_type: POLICY_TEXT
 value_data: "vssrvc.exe" || "vss.exe"
 port_type: TCP
 port_no: "5900"
 port_option: CAN_BE_CLOSED
</custom_item>
```

El ejemplo siguiente es similar al primero, excepto que este ejemplo demuestra el uso de check_type.

```
<custom_item>
 type: AUDIT_PROCESS_ON_PORT
 description: "Audit Process on Port - check_regex"
 value_type: POLICY_TEXT
 value_data: "foo.exe" || "vss.+"
 port_type: TCP
 port_no: "5900"
 check_type: CHECK_REGEX
</custom_item>
```

CHECK_ACCOUNT

Uso

```
<custom_item>
 type: CHECK_ACCOUNT
 description: ["description"]
 value_type: [VALUE_TYPE]
 value_data: [value]
 account_type: [ACCOUNT_TYPE]
 (optional) check_type: [CHECK_TYPE]
</custom_item>
```

Este elemento de directiva comprueba los siguientes valores definidos en “Security Settings (Configuración de seguridad) -> Local Policies (Directivas locales) -> Security Options (Opciones de seguridad)”.

- Accounts: Administrator account status (Cuentas: estado de cuentas de administrador)
- Accounts: Guest account status (Cuentas: estado de cuentas de invitado)
- Accounts: Rename administrator account (Cuentas: cambiar nombre de cuenta de administrador)
- Accounts: Rename guest account (Cuentas: cambiar nombre de cuenta de invitado)

La comprobación se lleva a cabo empleando la función **`LsaQueryInformationPolicy`** con el nivel **`PolicyAccountDomainInformation`** para obtener el SID de dominio/sistema, **`LsaLookupSid para`** obtener los nombres de administradores y de invitados, y **`NetUserGetInfo`** para obtener información de cuentas.

Este elemento usa el campo **`account_type`** para describir qué cuenta debe auditarse. Los tipos permitidos son los siguientes:

- ADMINISTRATOR_ACCOUNT (“Cuentas: estado de cuentas de administrador”)
  ```
  value_type: POLICY_SET
  value_data: "Enabled" or "Disabled"
  ```
- GUEST_ACCOUNT (“Cuentas: estado de cuentas de invitado”)
  ```
  value_type: POLICY_SET
  value_data: "Enabled" or "Disabled"
  ```
- ADMINISTRATOR_ACCOUNT (“Cuentas: cambiar nombre de cuenta de administrador”)
  ```
  value_type: POLICY_TEXT
  value_data: "TEXT HERE" [administrator name]
  check_type: [CHECK_TYPE] (any one of the possible check_type values)
  ```
- GUEST_ACCOUNT (“Cuentas: cambiar nombre de cuenta de invitado”)
  ```
  value_type: POLICY_TEXT
  value_data: "TEXT HERE" [guest name]
  check_type: [CHECK_TYPE] (any one of the possible check_type values)
  ```

De acuerdo con la porción de credenciales del dominio, es posible que se comprueben las cuentas de sistema locales o las cuentas de dominio.

Ejemplos:

```
<custom_item>
 type: CHECK_ACCOUNT
 description: "Accounts: Guest account status"
 value_type: POLICY_SET
 value_data: "Disabled"
 account_type: GUEST_ACCOUNT
</custom_item>

<custom_item>
 type: CHECK_ACCOUNT
 description: "Accounts: Rename administrator account"
 value_type: POLICY_TEXT
 value_data: "Dom_adm"
 account_type: ADMINISTRATOR_ACCOUNT
</custom_item>

<custom_item>
 type: CHECK_ACCOUNT
 description: "Accounts: Rename administrator account"
 value_type: POLICY_TEXT
 value_data: "Administrator"
 account_type: ADMINISTRATOR_ACCOUNT
 check_type: CHECK_NOT_EQUAL
</custom_item>
```

CHECK_LOCAL_GROUP

Uso

```
<custom_item>
 type: CHECK_LOCAL_GROUP
 description: ["description"]
 value_type: [VALUE_TYPE]
 value_data: [value]
 group_type: [GROUP_TYPE]
 (optional) check_type: [CHECK_TYPE]
</custom_item>
```

Este elemento de directiva comprueba los nombres de grupo y los estados de los grupos que aparecen en **`lusmgr.msc`**.

Este elemento usa el campo **`group_type`** para describir qué cuenta debe auditarse. Los tipos permitidos son los siguientes:

- ADMINISTRATORS_GROUP
- USERS_GROUP
- GUESTS_GROUP
- POWER_USERS_GROUP
- ACCOUNT_OPERATORS_GROUP
- SERVER_OPERATORS_GROUP

- PRINT_OPERATORS_GROUP
- BACKUP_OPERATORS_GROUP
- REPLICATORS_GROUP

Los tipos permitidos para el campo **value_type** son los siguientes:

- POLICY_SET (se comprueba el estado del grupo)
  `value_type: POLICY_SET`
  `value_data: "Enabled" or "Disabled"`
- POLICY_TEXT (se comprueba el nombre del grupo)
  `value_type: POLICY_TEXT`
  `value_data: "Guests1"` (In this case **value_data** can be any text string)

Ejemplos:

```
<custom_item>
 type: CHECK_LOCAL_GROUP
 description: "Local Guest group must be enabled"
 value_type: POLICY_SET
 value_data: "enabled"
 group_type: GUESTS_GROUP
 check_type: CHECK_EQUAL
</custom_item>
```

```
<custom_item>
 type: CHECK_LOCAL_GROUP
 description: "Guests group account name should be Guests"
 value_type: POLICY_TEXT
 value_data: "Guests"
 group_type: GUESTS_GROUP
 check_type: CHECK_EQUAL
</custom_item>
```

```
<custom_item>
 type: CHECK_LOCAL_GROUP
 description: "Guests group account name should not be Guests"
 value_type: POLICY_TEXT
 value_data: "Guests"
 group_type: GUESTS_GROUP
 check_type: CHECK_NOT_EQUAL
</custom_item>
```

ANONYMOUS_SID_SETTING

Uso

```
<custom_item>
 type: ANONYMOUS_SID_SETTING
 description: ["description"]
 value_type: [VALUE_TYPE]
```

```
 value_data: [value]
 (optional) check_type: [value]
</custom_item>
```

Este elemento de directiva comprueba el siguiente valor definido en “Security Settings (Configuración de seguridad) -> Local Policies (Directivas locales) -> Security Options (Opciones de seguridad) -> Network Access: Allow anonymous SID/Name translation (Acceso de red: permitir traducción SID/nombre anónima)”. La comprobación se lleva a cabo empleando la función **`LsaQuerySecurityObject`** en el identificador de directivas LSA.

Los tipos permitidos son los siguientes:

```
value_type: POLICY_SET
value_data: "Enabled" or "Disabled"
```

Al usar esta auditoría, tenga en cuenta que esta directiva:

- es una comprobación de permisos del servicio de LSA;
- comprueba si “ANONYMOUS_USER” (USUARIO_ANÓNIMO) tiene establecido el flag (indicador) “POLICY_LOOKUP_NAMES“ (NOMBRES_DE_BÚSQUEDA_ DE_DIRECTIVAS);
- no se utiliza en Windows 2003 porque un usuario anónimo no puede obtener acceso al canal de LSA.

Ejemplo:

```
 <custom_item>
  type: ANONYMOUS_SID_SETTING
  description: "Network access: Allow anonymous SID/Name translation"
  value_type: POLICY_SET
  value_data: "Disabled"
 </custom_item>
```

SERVICE_POLICY

Esta comprobación requiere acceso al registro remoto para que el sistema de Windows remoto funcione correctamente.

Uso

```
<custom_item>
 type: SERVICE_POLICY
 description: ["description"]
 value_type: [VALUE_TYPE]
 value_data: [value]
 (optional) check_type: [value]
 service_name: ["service name"]
</custom_item>
```

Este elemento de directiva comprueba los valores de inicio definidos en “Servicios del sistema”. La comprobación se lleva a cabo empleando la función **`RegQueryValueEx`** en las siguientes claves:

- key: "SYSTEM\CurrentControlSet\Services\" + service_name

- item: "Start"

Los tipos permitidos son los siguientes:

```
value_type: SERVICE_SET
value_data: "Automatic", "Manual" or "Disabled"
svc_option: CAN_BE_NULL or CAN_NOT_BE_NULL
```

El campo **service_name** corresponde al nombre REAL del servicio. Este nombre se puede obtener mediante las siguientes acciones:

1. iniciar el panel de control Services (Servicios) (en Administrative Tools [Herramientas administrativas]);
2. seleccionar el servicio deseado;
3. abrir el cuadro de diálogo de propiedades (clic derecho -> Properties [Propiedades]);
4. extraer la porción “Service name” (Nombre del servicio).

La opción de permisos de servicios se puede comprobar mediante un elemento SERVICE_PERMISSIONS.

Ejemplo:

```
 <custom_item>
  type: SERVICE_POLICY
  description: "Background Intelligent Transfer Service"
  value_type: SERVICE_SET
  value_data: "Disabled"
  service_name: "BITS"
 </custom_item>
```

GROUP_MEMBERS_POLICY

Uso

```
<custom_item>
 type: GROUP_MEMBERS_POLICY
 description: ["description"]
 value_type: [value type]
 value_data: [value]
 (optional) check_type: [value]
 group_name: ["group name"]
</custom_item>
```

Este elemento de directiva comprueba que exista una lista específica de usuarios en uno o más grupos.

El tipo permitido es el siguiente:

```
value_type: POLICY_TEXT or POLICY_MULTI_TEXT
value_data: "user1" && "user2" && ... && "usern"
```

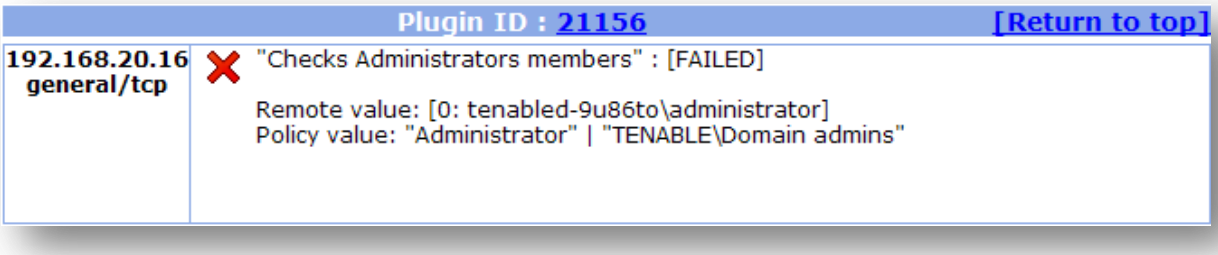
Al usar esta auditoría, tenga en cuenta que se puede especificar un nombre de usuario con un nombre de dominio como “MYDOMAIN\John Smith”, y el campo **group_name** especifica un único grupo para auditar.

Un único archivo **.audit** de Nessus puede especificar varios elementos del cliente diferentes, por lo que resulta muy sencillo auditar listas de usuarios en varios grupos. A continuación se presenta un ejemplo de directiva **.audit** que

busca que el grupo “Administrators” (Administradores) solo contenga el usuario “Administrator” (Administrador) y “TENABLE\Domain admins” (TENABLE\Administradores de dominio):

```
<custom_item>
 type: GROUP_MEMBERS_POLICY
 description: "Checks Administrators members"
 value_type: POLICY_MULTI_TEXT
 value_data: "Administrator" && "TENABLE\Domain admins"
 group_name: "Administrators"
</custom_item>
```

A continuación se presenta una captura de pantalla de un ejemplo en el que se ejecuta el contenido del archivo **`.audit`** mencionado en un servidor de Windows 2003:



USER_GROUPS_POLICY

Uso

```
<custom_item>
 type: USER_GROUPS_POLICY
 description: ["description"]
 value_type: [value type]
 value_data: [value]
 (optional) check_type: [value]
 user_name: ["user name"]
</custom_item>
```

Este elemento de directiva comprueba que los usuarios de Windows pertenezcan a los grupos especificados en **`value_data`**. Al usar esta auditoría, usted solo puede probar los usuarios de dominio respecto de un controlador de dominio. Esta comprobación no puede aplicarse en usuarios incorporados como “Local Service” (Servicio local).

Ejemplo:

```
<custom_item>
 type: USER_GROUPS_POLICY
 description: "3.72 DG0005: DBMS administration OS accounts"
 info: "Checking that the 'dba' account is a member of required groups only."
 info: "Modify the account/groups in this audit to match your environment."
 value_type: POLICY_MULTI_TEXT
 value_data: "Users" && "SQL Server DBA" && "SQL Server Users"
 user_name: "dba"
</custom_item>
```

USER_RIGHTS_POLICY

Uso

```
<custom_item>
 type: USER_RIGHTS_POLICY
 description: ["description"]
 value_type: [value type]
 value_data: [value]
 (optional) check_type: [value]
 right_type: [right]
</custom_item>
```

Este elemento de directiva comprueba el siguiente valor definido en “Security Settings (Configuración de seguridad) -> Local Policies (Directivas locales) -> User Rights Assignment (Asignación de derechos de usuario)”. La comprobación se lleva a cabo empleando la función **`LsaEnumerateAccountsWithUserRight`** en el identificador de directivas LSA.

El campo **`right_type`** corresponde al derecho de realizar pruebas. Los valores permitidos son los siguientes:

```
right_type: RIGHT
```

Donde **`RIGHT`** puede ser:

```
SeAssignPrimaryTokenPrivilege
SeAuditPrivilege
SeBackupPrivilege
SeBatchLogonRight
SeChangeNotifyPrivilege
SeCreateGlobalPrivilege
SeCreatePagefilePrivilege
SeCreatePermanentPrivilege
SeCreateTokenPrivilege
SeDenyBatchLogonRight
SeDenyInteractiveLogonRight
SeDenyNetworkLogonRight
SeDenyRemoteInteractiveLogonRight
SeDenyServiceLogonRight
SeDebugPrivilege
SeEnableDelegationPrivilege
SeImpersonatePrivilege
SeIncreaseBasePriorityPrivilege
SeIncreaseWorkingSetPrivilege
SeIncreaseQuotaPrivilege
SeInteractiveLogonRight
SeLoadDriverPrivilege
SeLockMemoryPrivilege
SeMachineAccountPrivilege
SeManageVolumePrivilege
SeNetworkLogonRight
SeProfileSingleProcessPrivilege
SeRemoteShutdownPrivilege
SeRemoteInteractiveLogonRight
SeReLabelPrivilege
SeRestorePrivilege
SeSecurityPrivilege
```

```
SeServiceLogonRight
SeShutdownPrivilege
SeSyncAgentPrivilege
SeSystemEnvironmentPrivilege
SeSystemProfilePrivilege
SeSystemtimePrivilege
SeTakeOwnershipPrivilege
SeTcbPrivilege
SeTimeZonePrivilege
SeUndockPrivilege
SeUnsolicitedInputPrivilege
```

El tipo permitido es el siguiente:

```
value_type: USER_RIGHT
value_data: "user1" && "user2" && "group1" && ... && "groupn"
```

Las pruebas de derechos de usuarios efectúan muchas solicitudes en el controlador de dominio. Estas pruebas se deben incluir en un archivo de directiva independiente, y solo iniciarse en el Domain Controller (Controlador de dominio) y un ÚNICO sistema del dominio.

No deben usarse comillas alrededor del tipo **`right`**, ya que se analiza sintácticamente como token.

Ejemplo:

```
 <custom_item>
  type: USER_RIGHTS_POLICY
  description: "Create a token object"
  value_type: USER_RIGHT
  value_data: "Administrators" && "Backup Operators"
  right_type: SeCreateTokenPrivilege
 </custom_item>
```

FILE_CHECK

Esta comprobación requiere acceso al registro remoto para que el sistema de Windows remoto funcione correctamente.

Uso

```
<custom_item>
 type: FILE_CHECK
 description: ["description"]
 value_type: [VALUE_TYPE]
 value_data: [value]
 (optional) check_type: [value]
 file_option: [OPTION_TYPE]
</custom_item>
```

Este elemento de directiva comprueba si el archivo (**value_data**) existe o no (**file_option**). La comprobación se lleva a cabo empleando la función **CreateFile**.

Los tipos permitidos son los siguientes:

```
value_type: POLICY_TEXT
value_data: "file name"
file_option: MUST_EXIST or MUST_NOT_EXIST
```

Ejemplos:

```
<custom_item>
 type: FILE_CHECK
 description: "Check that win.ini exists in the system root"
 value_type: POLICY_TEXT
 value_data: "%SystemRoot%\win.ini"
 file_option: MUST_EXIST
</custom_item>
```

```
<custom_item>
 type: FILE_CHECK
 description: "Check that bad.exe does not exist in the system root"
 value_type: POLICY_TEXT
 value_data: "%SystemRoot%\bad.exe"
 file_option: MUST_NOT_EXIST
</custom_item>
```

FILE_VERSION

Esta comprobación requiere acceso al registro remoto para que el sistema de Windows remoto funcione correctamente.

Uso

```
<custom_item>
 type: FILE_VERSION
 description: ["description"]
 value_type: [VALUE_TYPE]
 value_data: [value]
 (optional) check_type: [value]
 file: PATH_TO_FILE
 file_option: [OPTION_TYPE]
 check_type: CHECK_TYPE
</custom_item>
```

Este elemento de directiva comprueba de manera predeterminada si la versión del archivo especificado en el campo **file** es superior o igual a la versión del archivo remoto. La comprobación también se puede usar para determinar si la versión del archivo remoto es anterior, mediante la opción **check_type**.

Los tipos permitidos son los siguientes:

```
value_type: POLICY_FILE_VERSION
```

```
value_data: "file version"
file_option: MUST_EXIST or MUST_NOT_EXIST
```

Ejemplos:

```
<custom_item>
 type: FILE_VERSION
 description: "Audit for C:\WINDOWS\SYSTEM32\calc.exe"
 value_type: POLICY_FILE_VERSION
 value_data: "1.1.1.1"
 file: "C:\WINDOWS\SYSTEM32\calc.exe"
</custom_item>
```

```
<custom_item>
 type: FILE VERSION
 description: "Audit for C:\WINDOWS\SYSTEM32\calc.exe"
 value_type: POLICY_FILE_VERSION
 value_data: "1.1.1.1"
 file: "C:\WINDOWS\SYSTEM32\calc.exe"
 check type: CHECK LESS THAN
</custom_item>
```

FILE_PERMISSIONS

Esta comprobación requiere acceso al registro remoto para que el sistema de Windows remoto funcione correctamente.

Uso

```
<custom_item>
 type: FILE_PERMISSIONS
 description: ["description"]
 value_type: [value_type]
 value_data: [value]
 (optional) check_type: [value]
 file: ["filename"]
 (optional) acl_option: [acl_option]
</custom_item>
```

Este elemento de directiva comprueba si la ACL FILE_PERMISSIONS es correcta. La comprobación se lleva a cabo empleando la función **`GetSecurityInfo`** con el nivel 7 en el identificador de archivos.

El tipo permitido es el siguiente:

```
value_type: FILE_ACL
value_data: "ACLname"
file: "PATH\Filename"
```

Las siguientes rutas predefinidas se pueden usar en el nombre de archivo o carpeta:

```
%allusersprofile%
%windir%
%systemroot%
```

```
%commonfiles%
%programfiles%
%systemdrive%
%systemdirectory%
```

Al usar esta auditoría, tenga en cuenta lo siguiente:

- El campo **file** debe incluir la ruta completa en el nombre de archivo o la carpeta (por ejemplo, **C:\WINDOWS\SYSTEM32**) o usar las palabras clave de ruta mencionadas. Si se usan palabras clave de ruta, se debe habilitar el registro remoto para que permita a Nessus determinar los valores variables de ruta.
- El campo **value_data** representa el nombre de una ACL definida en el archivo de directiva.
- El campo **acl_option** se puede establecer en CAN_BE_NULL o CAN_NOT_BE_NULL para forzar un resultado satisfactorio o de error si el archivo no existe.

Ejemplos:

```
<file_acl: "ACL1">

 <user: "Administrators">
  acl_inheritance: "not inherited"
  acl_apply: "This object only"
  acl_allow: "Full Control"
 </user>

 <user: "System">
  acl_inheritance: "not inherited"
  acl_apply: "This object only"
  acl_allow: "Full Control"
 </user>

</acl>

<custom_item>
 type: FILE_PERMISSIONS
 description: "Permissions for C:\WINDOWS\SYSTEM32"
 value_type: FILE_ACL
 value_data: "ACL1"
 file: "C:\WINDOWS\SYSTEM32"
</custom_item>
```

```
<custom_item>
 type: FILE_PERMISSIONS
 description: "Permissions for C:\WINDOWS\SYSTEM32"
 value_type: FILE_ACL
 value_data: "ACL1"
 file: "%SystemRoot%\SYSTEM32"
</custom_item>
```

Cuando se ejecuta la comprobación anterior, el módulo de compatibilidad comprobará si los permisos definidos para "**%SystemRoot%\SYSTEM32**" coinciden con los que se describen en la ACL1 file_acl.

FILE_AUDIT

Esta comprobación requiere acceso al registro remoto para que el sistema de Windows remoto funcione correctamente.

Uso

```
<custom_item>
 type: FILE_AUDIT
 description: ["description"]
 value_type: [value_type]
 value_data: [value]
 (optional) check_type: [value]
 file: ["filename"]
 (optional) acl_option: [acl_option]
</custom_item>
```

Este elemento de directiva se usa para comprobar las propiedades de auditoría (Properties [Propiedades] -> Security [Seguridad] -> Advanced [Opciones avanzadas] -> Auditing [Auditoría]) de un archivo o de una carpeta mediante la ACL especificada. Esta comprobación se lleva a cabo empleando la función **`GetSecurityInfo`** con el nivel SACL_SECURITY_INFORMATION en el identificador de archivos.

El tipo permitido es el siguiente:

```
value_type: FILE_ACL
value_data: "ACLname"
file: "PATH\Filename"
```

Las siguientes rutas predefinidas se pueden usar en el nombre de archivo o carpeta:

```
%allusersprofile%
%windir%
%systemroot%
%commonfiles%
%programfiles%
%systemdrive%
%systemdirectory%
```

Al usar esta auditoría, tenga en cuenta lo siguiente:

- El campo **`file`** debe incluir la ruta completa en el nombre de archivo o la carpeta (por ejemplo, **`C:\WINDOWS\SYSTEM32`**) o usar las palabras clave de ruta mencionadas. Si se usan palabras clave de ruta, se debe habilitar el registro remoto para que permita a Nessus determinar los valores variables de ruta.
- El campo **`value_data`** representa el nombre de la ACL definida en el archivo de directiva.
- El campo **`acl_option`** se puede establecer en CAN_BE_NULL o CAN_NOT_BE_NULL para forzar un resultado satisfactorio o de error si el archivo no existe.
- Los campos **`acl_allow`** y **`acl_deny`** corresponden a los eventos de auditoría “Successful” (Correcto) y “Failed” (Incorrecto).

A continuación se incluye un ejemplo de archivo **`.audit`** que implementa la función FILE_AUDIT, incluido un ejemplo de regla para una lista de control de acceso denominada “ACL1”.

```
<check_type: "Windows" version:"2">
<group_policy: "Audits SYSTEM32 directory for correct auditing permissions">

<file_acl: "ACL1">
 <user: "Everyone">
  acl_inheritance: "not inherited"
  acl_apply: "This folder, subfolders and files"
  acl_deny: "full control"
  acl_allow: "full control"
 </user>
</acl>

<custom_item>
 type: FILE_AUDIT
 description: "Audit for C:\WINDOWS\SYSTEM32"
 value_type: FILE_ACL
 value_data: "ACL1"
 file: "%SystemRoot%\SYSTEM32"
</custom_item>

</group_policy>
</check_type>
```

FILE_CONTENT_CHECK

Esta comprobación requiere acceso al registro remoto para que el sistema de Windows remoto funcione correctamente.

Uso

```
<custom_item>
 type: FILE_CONTENT_CHECK
 description: ["description"]
 value_type: [value_type]
 value_data: ["filename"]
 (optional) check_type: [value]
 regex: ["regex"]
 expect: ["regex"]
 (optional) file_option: [file_option]
(optional) avoid_floppy_access
</custom_item>
```

Este elemento de directiva comprueba si el archivo contiene la expresión regular **`regex`**, y que esta expresión coincida con **`expect`**.

La comprobación se lleva a cabo empleando la función **`ReadFile`** en el identificador de archivos.

El archivo se lee de SMB a un búfer de memoria en el servidor Nessus, y luego el búfer se procesa para comprobar la compatibilidad o incompatibilidad. Los archivos no se guardan en el disco del servidor Nessus; solo se copian a un búfer de memoria para su análisis.

El tipo permitido es el siguiente:

```
value_type: POLICY_TEXT
value_data: "PATH\Filename"
regex: "regex"
expect: "regex"
```

Las siguientes rutas predefinidas se pueden usar en el nombre de archivo o carpeta:

```
%allusersprofile%
%windir%
%systemroot%
%commonfiles%
%programfiles%
%systemdrive%
```

Al usar este tipo de auditoría, tenga en cuenta lo siguiente:

- El campo **value_data** debe incluir la ruta completa en el nombre de archivo o la carpeta (por ejemplo, **C:\WINDOWS\SYSTEM32**) o usar las palabras clave de ruta mencionadas. Si se usan palabras clave de ruta, se debe habilitar el registro remoto para que permita a Nessus determinar los valores variables de ruta.
- El campo **regex** comprueba que exista un elemento en el archivo.
- El campo **expect** comprueba que el elemento coincida con la expresión regular.
- El campo **file_option** se puede establecer en CAN_BE_NULL para forzar un resultado satisfactorio si el archivo no existe.
- El campo **file_option** se puede establecer en CAN_NOT_BE_NULL para forzar un resultado de error si el archivo existe y se encuentra vacío.
- El campo **avoid_floppy_access** puede configurarse para que indique a la auditoría que no ejecute una comprobación que diera como resultado el acceso a la unidad de disquete. Esto se debe utilizar si una auditoría hace que se acceda a la unidad de disquete cuando no hay ningún disquete en la unidad.

Ejemplo:

```
<custom_item>
 avoid_floppy_access
 type: FILE_CONTENT_CHECK
 description: "File content for C:\WINDOWS\win.ini"
 value_type: POLICY_TEXT
 value_data: "C:\WINDOWS\win.ini"
 regex: "aif=.*"
 expect: "aif=MPEGVideo"
</custom_item>
```

FILE_CONTENT_CHECK_NOT

Esta comprobación requiere acceso al registro remoto para que el sistema de Windows remoto funcione correctamente.

Uso

```
<custom_item>
 type: FILE_CONTENT_CHECK_NOT
 description: ["description"]
 value_type: [value_type]
 value_data: ["filename"]
 (optional) check_type: [value]
 regex: ["regex"]
 expect: ["regex"]
 (optional) file_option: [file_option]
</custom_item>
```

Este elemento de directiva comprueba si el archivo contiene la expresión regular **regex**, y que esta expresión no coincida con **expect**. La comprobación se lleva a cabo empleando la función **ReadFile** en el identificador de archivos.

El tipo permitido es el siguiente:

```
value_type: POLICY_TEXT
value_data: "PATH\Filename"
regex: "regex"
expect: "regex"
```

Las siguientes rutas predefinidas se pueden usar en el nombre de archivo o carpeta:

```
%allusersprofile%
%windir%
%systemroot%
%commonfiles%
%programfiles%
%systemdrive%
```

Al usar este tipo de auditoría, tenga en cuenta lo siguiente:

- El campo **value_data** debe incluir la ruta completa en el nombre de archivo o la carpeta (por ejemplo, **C:\WINDOWS\SYSTEM32**) o usar las palabras clave de ruta mencionadas. Si se usan palabras clave de ruta, se debe habilitar el registro remoto para que permita a Nessus determinar los valores variables de ruta.
- El campo **regex** comprueba que exista un elemento en el archivo.
- El campo **expect** comprueba que el elemento coincida con la expresión regular.
- El campo **file_option** se puede establecer en CAN_BE_NULL para forzar un resultado satisfactorio si el archivo no existe.
- El campo **file_option** puede establecerse en CAN_NOT_BE_NULL para forzar un resultado de error si el archivo existe y se encuentra vacío.

Ejemplo:

```
 <custom_item>
  type: FILE_CONTENT_CHECK_NOT
  description: "File content for C:\WINDOWS\win.ini"
  value_type: POLICY_TEXT
  value_data: "C:\WINDOWS\win.ini"
```

```
  (optional) check_type: [value]
  regex: "au=.*"
  expect: "au=MPEGVideo2"
  file_option: CAN_NOT_BE_NULL
 </custom_item>
```

REG_CHECK

Esta comprobación requiere acceso al registro remoto para que el sistema de Windows remoto funcione correctamente.

Uso

```
<custom_item>
 type: REG_CHECK
 description: ["description"]
 value_type: [VALUE_TYPE]
 value_data: [value]
 reg_option: [OPTION_TYPE]
 (optional) check_type: [value]
 (optional) key_item: [item value]
</custom_item>
```

Este elemento de directiva comprueba si la clave (o el elemento) del registro existe o no. La comprobación se lleva a cabo empleando las funciones **RegOpenKeyEx** y **RegQueryValueEx**.

Los tipos permitidos son los siguientes:

```
value_type: POLICY_TEXT
value_data: "key path"
reg_option: MUST_EXIST or MUST_NOT_EXIST
key_item: "item name"
```

Si el campo **key_item** no se encuentra especificado, este elemento comprueba que exista la ruta de acceso a clave. De lo contrario, comprueba que el elemento exista.

Ejemplo:

```
 <custom_item>
  type: REG_CHECK
  description: "Check the key HKLM\SOFTWARE\Adobe\Acrobat Reader\7.0\AdobeViewer"
  value_type: POLICY_TEXT
  value_data: "HKLM\SOFTWARE\Adobe\Acrobat Reader\7.0\AdobeViewer"
  reg_option: MUST_NOT_EXIST
  key_item: "EULA"
 </custom_item>
```

REGISTRY_SETTING

Esta comprobación requiere acceso al registro remoto para que el sistema de Windows remoto funcione correctamente.

Uso

```
<custom_item>
 type: REGISTRY_SETTING
 description: ["description"]
 value_type: [VALUE_TYPE]
 value_data: [value]
 reg_key: ["key name"]
 reg_item: ["key item"]
 (optional) check_type: [value]
 (optional) reg_option: [KEY_OPTIONS]
 (optional) reg_enum: ENUM_SUBKEYS
</custom_item>
```

Este elemento de directiva se usa para comprobar el valor de una clave del registro. Muchas comprobaciones de directiva en “Security Settings (Configuración de seguridad) -> Local Policies (Directivas locales) -> Security Options (Opciones de seguridad)” usan este elemento de directiva. Esta comprobación se lleva a cabo empleando la función **`RegQueryValueEx`**.

El campo **`reg_key`** es el nombre de la clave del registro (por ejemplo, “HKLM\SOFTWARE\Microsoft\Driver Signing”). La primera porción de la clave (HKLM) se usa para conectarse con el subárbol del registro correcto. La ruta de acceso subsiguiente es una designación estática en la que se encuentra el **`reg_item`** deseado.

El subárbol HKU (HKEY_USERS) constituye un caso especial. No es posible especificar una SID para las claves HKU. Lo que sucede es que el `nbin` procesa internamente una iteración en cada SID, y se aprueba solamente si el valor en cada SID es válido.

Por ejemplo:

```
<custom_item>
 type: REGISTRY_SETTING
 description: "HKU\Control Panel\Desktop\ScreenSaveActive"
 value_type: POLICY_DWORD
 value_data: 1
 reg_key: "HKU\Control Panel\Desktop"
 reg_item: "ScreenSaveActive"
</item>
```

se repetirá por:

```
HKU\S-1-5-18\Control Panel\Desktop\ScreenSaveActive
HKU\S-1-5-19\Control Panel\Desktop\ScreenSaveActive
HKU\S-1-5-20\Control Panel\Desktop\ScreenSaveActive
...
```

y se aprobará si el elemento “ScreenSaveActive” se encuentra establecido en 1 para todas las SID.

El campo opcional **`reg_option`** se puede establecer en CAN_BE_NULL, para forzar que la comprobación tenga resultado satisfactorio si la clave no existe, o en la opción opuesta CAN_NOT_BE_NULL.

Se puede usar una opción **`reg_enum`** adicional con el argumento “ENUM_SUBKEYS” para enumerar un valor especificado para todas las subclaves de una clave del registro. Por ejemplo, la clave: **`HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall`** enumera muchos paquetes de software. Si desea que el valor de “CurrentVersion” sea el mismo para todas las subclaves en “Uninstall” (Desinstalar), use **`reg_enum`**.

Ejemplo:

```
<custom_item>
 type: REGISTRY_SETTING
 description: "DBMS network port, protocol, and services (PPS) usage"
 info: "Checking whether TCPDynamicPorts key value is configured (should be blank)."
 value_type: POLICY_TEXT
 value_data: ""
 reg_key: "HKLM\SOFTWARE\Microsoft\Microsoft SQL
       Server\MSSQL.1\MSSQLServer\SuperSocketNetLib\Tcp"
 reg_item: "TCPDynamicPorts"
 reg_enum: ENUM_SUBKEYS
 reg_option: CAN_BE_NULL
</custom_item>
```

Esta auditoría del subárbol del registro HKU no incluye el SID (identificador de seguridad) en la ruta de acceso al registro **`reg_key`**. Este ejemplo realizará una búsqueda del **`reg_item`** especificado en cada SID del HKU.

Ejemplo:

```
<custom_item>
 type: REGISTRY_SETTING
 description: "FakeAlert.BG trojan check"
 value_type: POLICY_TEXT
 reg_key: "HKU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run"
 reg_item: "brastk"
 value_data: "C:\WINDOWS\System32\brastk.exe"
 reg_option: CAN_BE_NULL
 check_type: CHECK_NOT_EQUAL
 info: "A registry entry for FakeAlert.BG trojan/downloader was found."
 info: "The contents of this audit can be edited as desired."
</custom_item>
```

Se encuentran disponibles los siguientes tipos de campos **`value_type`** principales:

- POLICY_SET
  `value_data: "Enabled" or "Disabled"`
- POLICY_DWORD
  `value_data: DWORD or RANGE [same dword as in registry or range]`
- POLICY_TEXT
  `value_data: "TEXT" [same text as in registry]`
- POLICY_MULTI_TEXT
  `value_data: "TEXT1" && "TEXT2" && ... && "TEXTN" [same texts as in registry]`
- POLICY_BINARY
  `value_data: "0102ac0b...34fb" [same binary as in registry]`
- FILE_ACL, REG_ACL, SERVICE_ACL, LAUNCH_ACL, ACCESS_ACL
  `value_data: "acl_name" [name of the acl to use]`

Los siguientes tipos de campos **value_type** opcionales se encuentran disponibles y se emplean en elementos predefinidos:

- DRIVER_SET
  `value_data: "Silent Succeed", "Warn but allow installation", "Do not allow installation"`
- LDAP_SET
  `value_data: "None" or "Require Signing"`
- LOCKEDID_SET
  `value_data: "user display name, domain and user names", "user display name only", "do not display user information"`
- SMARTCARD_SET
  `value_data: "No action", "Lock workstation", "Force logoff", "Disconnect if a remote terminal services session"`
- LOCALACCOUNT_SET
  `value_data: "Classic - local users authenticate as themselves", "Guest only - local users authenticate as guest"`
- NTLMSSP_SET
  `value_data: "No minimum", "Require message integrity", "Require message confidentiality", "Require ntlmv2 session security", "Require 128-bit encryption"`
- CRYPTO_SET
  `value_data: "User input is not required when new keys are stored and used", "User is prompted when the key is first used" or "User must enter a password each time they use a key"`
- OBJECT_SET
  `value_data: "Administrators group", "Object creator"`
- DASD_SET
  `value_data: "Administrators", "administrators and power users", "Administrators and interactive users"`
- LANMAN_SET
  `value_data: "Send LM & NTLM responses", "send lm & ntlm - use ntlmv2 session security if negotiated", "send ntlm response only", "send ntlmv2 response only", "send ntlmv2 response only\refuse lm" or "send ntlmv2 response only\refuse lm & ntlm"`
- LDAPCLIENT_SET
  `value_data: "None", "Negotiate Signing" or "Require Signing"`
- EVENT_METHOD
  `value_data: "by days", "manually" or "as needed"`
- POLICY_DAY
  `value_data: DWORD or RANGE (time in days)`
- POLICY_KBYTE
  `value_data: DWORD or RANGE`

En el caso del campo **custom_item**, use el **value_type** principal. Para los elementos predefinidos se crearon tipos opcionales.

Si el **value_type** es una ACL, el elemento del registro debe ser una descripción de seguridad con formato binario.

Ejemplos:

```
<custom_item>
 type: REGISTRY_SETTING
 descr°iption: "Network security: Do not store LAN Manager hash value on next password
       change"
 value_type: POLICY_SET
 value_data: "Enabled"
 reg_key: "HKLM\SYSTEM\CurrentControlSet\Control\Lsa"
 reg_item: "NoLMHash"
</custom_item>
```

```
<custom_item>
 type: REGISTRY_SETTING
 description: "Network access: Shares that can be accessed anonymously"
 value_type: POLICY_MULTI_TEXT
 value_data: "SHARE" && "EXAMPLE$"
 reg_key: "HKLM\SYSTEM\CurrentControlSet\Services\LanManServer\Parameters"
 reg_item: "NullSessionShares"
</custom_item>
```

```
<custom_item>
 type: REGISTRY_SETTING
 description: "DCOM: Network Provisioning Service - Launch permissions"
 value_type: LAUNCH_ACL
 value data: "2"
 reg_key: "HKLM\SOFTWARE\Classes\AppID\{39ce474e-59c1-4b84-9be2-2600c335b5c6}"
 reg_item: "LaunchPermission"
</custom_item>
```

```
<custom_item>
 type: REGISTRY_SETTING
 description: "DCOM: Automatic Updates - Access permissions"
 value_type: ACCESS_ACL
 value_data: "3"
 reg_key: "HKLM\SOFTWARE\Classes\AppID\{653C5148-4DCE-4905-9CFD-1B23662D3D9E}"
 reg_item: "AccessPermission"
</custom_item>
```

REGISTRY_PERMISSIONS

Esta comprobación requiere acceso al registro remoto para que el sistema de Windows remoto funcione correctamente.

Uso

```
<custom_item>
 type: REGISTRY_PERMISSIONS
```

```
 description: ["description"]
 value_type: [value_type]
 value_data: [value]
 (optional) check_type: [value]
 reg_key: ["regkeyname"]
 (optional) acl_option: [acl_option]
</custom_item>
```

Este elemento de directiva comprueba si la ACL de la clave del registro es correcta. La comprobación se lleva a cabo empleando la función **RegGetKeySecurity** en el identificador de claves del registro.

El tipo permitido es el siguiente:

```
value_type: REG_ACL
value_data: "ACLname"
reg_key: "RegistryKeyName"
```

Las siguientes rutas predefinidas se pueden usar para el campo **reg_key**:

```
HKLM (HKEY_LOCAL_MACHINE)
HKU (HKEY_USERS)
HKCR (HKEY_CLASS_ROOT)
```

Al usar esta auditoría, tenga en cuenta lo siguiente:

- El campo **reg_key** debe incluir la ruta completa que conduce a la clave del registro del archivo.
- El campo **value_data** representa el nombre de una ACL definida en el archivo de directiva.
- El campo **acl_option** se puede establecer en CAN_BE_NULL o CAN_NOT_BE_NULL para forzar un resultado satisfactorio o de error si la clave no existe.

Ejemplo:

```
<registry_acl: "ACL2">

 <user: "Administrators">
  acl_inheritance: "not inherited"
  acl_apply: "This key and subkeys"
  acl_allow: "Full Control"
 </user>

 <user: "SYSTEM">
  acl_inheritance: "not inherited"
  acl_apply: "This key and subkeys"
  acl_allow: "Full Control"
 </user>

</acl>

<custom_item>
 type: REGISTRY_PERMISSIONS
 description: "Permissions for HKLM\SOFTWARE\Microsoft"
 value_type: REG_ACL
 value_data: "ACL2"
 reg_key: "HKLM\SOFTWARE\Microsoft"
```

```
</custom_item>
```

Cuando se ejecuta la comprobación anterior, el módulo de compatibilidad comprobará si los permisos definidos para "**HKLM\SOFTWARE\Microsoft**" coinciden con los que se describen en la ACL2 registry_acl.

REGISTRY_AUDIT

Esta comprobación requiere acceso al registro remoto para que el sistema de Windows remoto funcione correctamente.

Uso

```
<custom_item>
 type: REGISTRY_AUDIT
 description: ["description"]
 value_type: [value_type]
 value_data: [value]
 reg_key: ["regkeyname"]
 (optional) acl_option: [acl_option]
</custom_item>
```

Este elemento de directiva comprueba si la ACL de la clave del registro es correcta. La comprobación se lleva a cabo empleando la función **RegGetKeySecurity** en el identificador de claves del registro.

El tipo permitido es el siguiente:

```
value_type: REG_ACL
value_data: "ACLname"
reg_key: "RegistryKeyName"
```

La siguiente ruta predefinida se puede usar para el campo **reg_key**:

```
HKLM (HKEY_LOCAL_MACHINE)
HKU (HKEY_USERS)
HKCR (HKEY_CLASS_ROOT)
```

Al usar esta auditoría, tenga en cuenta lo siguiente:

- El campo **reg_key** debe incluir la ruta completa que conduce a la clave del registro del archivo.
- El campo **value_data** representa el nombre de la ACL definida en el archivo de directiva.
- El campo **acl_option** se puede establecer en CAN_BE_NULL o CAN_NOT_BE_NULL para forzar un resultado satisfactorio o de error si la clave no existe.
- Los campos **acl_allow** y **acl_deny** corresponden a los eventos de auditoría "Successful" (Correcto) y "Failed" (Incorrecto).

A continuación se incluye un ejemplo de archivo **.audit** que realiza una auditoría de la clave del registro "HKLM\SOFTWARE\Microsoft" en una lista de control de acceso denominada "ACL2" que no se muestra:

```
<custom_item>
 type: REGISTRY_AUDIT
```

```
  description: "Audit for HKLM\SOFTWARE\Microsoft"
  value_type: REG_ACL
  value_data: "ACL2"
  reg_key: "HKLM\SOFTWARE\Microsoft"
 </custom_item>
```

REGISTRY_TYPE

Esta comprobación requiere acceso al registro remoto para que el sistema de Windows remoto funcione correctamente.

Uso

```
<custom_item>
 type: REGISTRY_TYPE
 description: ["description"]
 value_type: [VALUE_TYPE]
 value_data: [value]
 reg_key: ["key name"]
 reg_item: ["key item"]
 (optional) reg_option: [KEY_OPTIONS]
</item>
```

Este elemento de directiva se usa para comprobar el valor de un tipo de clave del registro. La comprobación se lleva a cabo empleando la función **`RegQueryValue`**.

El campo **`reg_key`** es el nombre de la clave del registro (“HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon”). La primera porción de la clave (HKLM, HKU, HKCU, ...) se usa para conectarse con el subárbol del registro correcto. En la mayoría de los casos el campo **`reg_key`** requiere una entrada de registro estática sin caracteres comodín; sin embargo, se permite una excepción al buscar valores en HKU (HKEY_USERS). Si se designa una ruta en HKU, la búsqueda procesa una iteración en todos los valores del usuario en HKU por el valor en la ruta designada. Por ejemplo, si `reg_key: "HKU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run"` se especifica con **`reg_item`** “brastk”, en todos los usuarios en HKU se buscará el valor de la clave del registro “brastk” en la ruta relativa: “HKU\<user_id>\SOFTWARE\Microsoft\Windows\CurrentVersion\Run”. Por ejemplo:

```
 value_type: POLICY_TEXT
 reg_key: "HKU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run"
 reg_item: "brastk"
 value_data: "C:\WINDOWS\System32\brastk.exe"
```

Esta comprobación busca en:

HKU\S-1-5-18\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
HKU\S-1-5-19\SOFTWARE\Microsoft\Windows\CurrentVersion\Run

El campo opcional **`reg_option`** se puede establecer en CAN_BE_NULL, para forzar que la comprobación tenga resultado satisfactorio si la clave no existe, o en la opción opuesta CAN_NOT_BE_NULL.

Solo POLICY_TEXT **`value_type`** está disponible para esta comprobación.

Este es un archivo **`.audit`** de ejemplo que hace una auditoría del tipo de registro de “HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon”:

```
<custom_item>
 type: REGISTRY_TYPE
 description: "Check type - reg_sz"
 value_type: POLICY_TEXT
 value_data: "reg_sz"
 reg_key: "HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon"
 reg_item: "ScreenSaverGracePeriod"
</item>
```

Tenga en cuenta que la auditoría de HKCU puede no funcionar en muchas instalaciones de Windows. Para esto se necesitan claves de “Current user” (Usuario actual), que generalmente no existen cuando Nessus autentica por SMB. Para solucionar esto, es posible auditar HKU (todos los usuarios). Cuando el plugin detecta que se está auditando una clave HKU, hace un bucle automáticamente en todas las SID disponibles, excepto en la clave `.DEFAULT`. La desventaja de este enfoque es que también auditará usuarios del sistema (por ejemplo, SYSTEM, NT Authority, etc.). Para evitar estos usuarios, puede usar `reg_ignore_hku_users`. Por ejemplo:

```
reg_ignore_hku_users : "S-1-5-18,S-1-5-19,S-1-5-20"
```

Esto solo funciona con la comprobación `REGISTRY_SETTING`.

SERVICE_PERMISSIONS

Uso

```
<custom_item>
 type: SERVICE_PERMISSIONS
 description: ["description"]
 value_type: [value_type]
 value_data: [value]
 (optional) check_type: [value]
 service: ["servicename"]
 (optional) acl_option: [acl_option]
</custom_item>
```

Este elemento de directiva comprueba si la ACL de servicio es correcta. La comprobación se lleva a cabo empleando la función **`QueryServiceObjectSecurity`** en el identificador de servicios.

El tipo permitido es el siguiente:

```
value_type: SERVICE_ACL
value_data: "ACLname"
service: "ServiceName"
```

Al usar esta auditoría, tenga en cuenta lo siguiente:

- El campo **`value_data`** representa el nombre de una ACL definida en el archivo de directiva.
- El campo **`acl_option`** se puede establecer en CAN_BE_NULL o CAN_NOT_BE_NULL para forzar un resultado satisfactorio o de error si la clave no existe.

Ejemplo:

```
<service_acl: "ACL3">
```

```
  <user: "Administrators">
   acl_inheritance: "not inherited"
   acl_apply: "This object only"
   acl_allow: "query template" | "change template" | "query status" | "enumerate
        dependents" | "start" | "stop" | "pause and continue" | "interrogate" | "user-
        defined control" | "delete" | "read permissions" | "change permissions" | "take
        ownership"
  </user>

  <user: "SYSTEM">
   acl_inheritance: "not inherited"
   acl_apply: "This object only"
   acl_allow: "query template" | "change template" | "query status" | "enumerate
        dependents" | "start" | "stop" | "pause and continue" | "interrogate" | "user-
        defined control" | "delete" | "read permissions" | "change permissions" | "take
        ownership"
  </user>

  <user: "Interactive">
   acl_inheritance: "not inherited"
   acl_apply: "This object only"
   acl_allow: "query template" | "query status" | "enumerate dependents" |
        "interrogate" | "user-defined control" | "read permissions"
  </user>

  <user: "Everyone">
   acl_inheritance: "not inherited"
   acl_apply: "This object only"
   acl_allow: "query template" | "change template" | "query status" | "enumerate
        dependents" | "start" | "stop" | "pause and continue" | "interrogate" | "user-
        defined control" | "delete" | "read permissions" | "change permissions" | "take
        ownership"
  </user>

 </acl>

 <custom item>
  type: SERVICE_PERMISSIONS
  description: "Permissions for Alerter Service"
  value_type: SERVICE_ACL
  value_data: "ACL3"
  service: "Alerter"
 </custom_item>
```

Cuando se ejecuta la comprobación anterior, el módulo de compatibilidad comprobará si los permisos definidos para el servicio ”Alerter” (Alerta) coinciden con los que se describen en la ACL3 service_acl.

SERVICE_AUDIT

Uso

```
<custom_item>
 type: SERVICE_AUDIT
 description: ["description"]
```

```
 value_type: [value_type]
 value_data: [value]
 (optional) check_type: [value]
 service: ["servicename"]
 (optional) acl_option: [acl_option]
</custom_item>
```

Este elemento de directiva comprueba si la ACL de servicio es correcta. La comprobación se lleva a cabo empleando la función **QueryServiceObjectSecurity** en el identificador de servicios.

El tipo permitido es el siguiente:

```
value_type: SERVICE_ACL
value_data: "ACLname"
service: "ServiceName"
```

Al usar este tipo de auditoría, tenga en cuenta lo siguiente:

- El campo **value_data** representa el nombre de la ACL definida en el archivo de directiva.
- El campo **acl_option** se puede establecer en CAN_BE_NULL o CAN_NOT_BE_NULL para forzar un resultado satisfactorio o de error si la clave no existe.
- Los campos **acl_allow** y **acl_deny** corresponden a los eventos de auditoría “Successful” (Correcto) y “Failed” (Incorrecto).

A continuación se incluye un ejemplo de archivo **.audit** para auditar el servicio “Alerter” (Alerta):

```
 <custom_item>
  type: SERVICE_AUDIT
  description: "Audit for Alerter Service"
  value_type: SERVICE_ACL
  value_data: "ACL3"
  service: "Alerter"
 </custom_item>
```

WMI_POLICY

Uso

```
<custom_item>
 type: WMI_POLICY
 description: "Test for WMI Value"
 value_type: [value_type]
 value_data: [value]
 (optional) check_type: [value]
 wmi_namespace: ["namespace"]
 wmi_request: ["request select statement"]
 wmi_attribute: ["attribute"]
 wmi_key: ["key"]
</custom_item>
```

Esta comprobación efectúa una consulta en la base de datos WMI de Windows en busca de valores especificados dentro del espacio de nombres/clase/atributo.

Se pueden extraer los valores de claves o se pueden enumerar los nombres de atributos de acuerdo con la sintaxis que se use.

Los tipos permitidos son los siguientes:

```
wmi_namespace: "namespace"
wmi_request: "WMI Query"
wmi_attribute: "Name"
wmi_key: "Name"
wmi_option: option
wmi_exclude_result: "result"
only_show_query_output: YES
check_type: CHECK_NOT_REGEX
```

Si elige a partir de una configuración de servicios con valores duplicados en el sistema (por ejemplo, “MSFTPSVC/83207416” y “MSFTPSVC/2”), la solicitud extraerá el atributo elegido de ambos. Si uno de ellos no coincide con el valor de la directiva, se añadirá **`wmi_key`** al informe para indicar cuál tuvo un error. El campo **`wmi_enum`** le permite enumerar nombres de configuraciones dentro de un espacio de nombres, para comparación o para comprobación de los valores de las directivas.

De manera predeterminada, si una consulta de WMI no devuelve ningún resultado, la comprobación notifica un error. Este comportamiento puede cambiarse, y puede hacerse que la comprobación notifique un PASS (Aprobado) si **`wmi_option`** está establecida en `CAN_BE_NULL`. Al establecer **`only_show_query_output`** en YES (SÍ), el resultado de la consulta de WMI se incluye ahora en el informe de Nessus. Usando la etiqueta **`check_type`**, usted puede obtener un resultado PASS (Aprobado) siempre y cuando una determinada cadena no exista en el resultado. Vea los ejemplos a continuación.

Otras consideraciones:

- Se deben especificar explícitamente los atributos de WMI. Por ejemplo, `seleccionar * de foo` no funcionará.
- Los atributos que no tengan un valor establecido no se notificarán.
- El uso de mayúsculas y minúsculas de los atributos debe ser exactamente como aparece en la documentación de Microsoft. Por ejemplo, el atributo `HandleCount` no puede ser `Handlecount` ni `handlecount`.
- Los valores del tipo de matriz no se incluyen en el resultado.

Ejemplo 1:

```
<custom_item>
 type: WMI_POLICY
 description: "IIS test"
 value_type: POLICY_DWORD
 value_data: 0
 wmi_namespace: "root/MicrosoftIISv2"
 wmi_request: "SELECT Name, UserIsolationMode FROM IIsFtpServerSetting"
 wmi_attribute: "UserIsolationMode"
 wmi_key: "Name"
</custom_item>
```

Si hay dos configuraciones de servicio FTP en el sistema (“MSFTPSVC/83207416” y “MSFTPSVC/2”), la solicitud extraerá el atributo “UserIsolationMode” de ambas. Si una de ellas no coincide con el valor de la directiva (0), se añadirá **`wmi_key`** (en este caso) al informe para indicar cuál tuvo un error.

Ejemplo 2:

```
<custom_item>
 type: WMI_POLICY
 description: "IIS test2"
 value_type: POLICY_MULTI_TEXT
 value_data: "MSFTPSVC/83207416" && "MSFTPSVC/2"
 wmi_namespace: "root/MicrosoftIISv2"
 wmi_request: "SELECT Name FROM IIsFtpServerSetting"
 wmi_attribute: "Name"
 wmi_key: "Name"
 wmi_option: WMI_ENUM
</custom_item>
```

Este ejemplo comprueba que haya dos nombres de configuración válidos, según lo especificado en **value_data**. Si desea obtener más información sobre el espacio de nombres WMI y los atributos relacionados, WMI CIM Studio de Microsoft resulta una herramienta valiosa que se encuentra disponible en el siguiente enlace:

http://www.microsoft.com/downloads/details.aspx?FamilyID=6430f853-1120-48db-8cc5-f2abdc3ed314&displaylang=en

Ejemplo 3:

```
<custom_item>
 type: WMI_POLICY
 description: "List All Windows Processes - except svchost.exe and iPodService.exe"
 value_type: POLICY_TEXT
 value_data: ""
 wmi_namespace: "root/cimv2"
 wmi_exclude_result: "svchost.exe,iPodService.exe"
 wmi_request: "select Caption,HandleCount,ThreadCount from Win32_Process"
 only_show_query_output: YES
</custom_item>
```

Este ejemplo incluirá en una lista todos los procesos de Windows, pero eliminará las instancias de **svchost.exe** y **iPodService.exe**.

Elementos

"Items" (Elementos) constituyen tipos de comprobaciones que se encuentran predefinidas en el motor de comprobaciones de compatibilidad de Windows. Se usan para elementos auditados habitualmente, y minimizan la sintaxis necesaria para crear comprobaciones de auditoría. Los elementos poseen la siguiente estructura:

```
<item>
 name: ["predefined_entry"]
 value: [value]
</item>
```

El campo **name** debe contar con un nombre que ya se haya definido [los nombres predefinidos se enumeran en la tabla "Predefined policies" (Directivas predefinidas), a continuación].

Todos los elementos predefinidos corresponden a la lista disponible en el Domain Policy Editor (Editor de directivas de dominio) de Windows 2003 SP1.

El siguiente ejemplo comprueba si la longitud de contraseña mínima se encuentra entre los 8 y 14 caracteres:

```
<item>
 name: "Minimum password length"
```

```
 value: [8..14]
</item>
```

El elemento personalizado correspondiente es:

```
<custom_item>
 type: PASSWORD_POLICY
 description: "Minimum password length"
 value_type: POLICY_DWORD
 value_data: [8..14]
 password_policy: MINIMUM_PASSWORD_LENGTH
</custom_item>
```

Directivas predefinidas

Directiva	Uso

Password Policy (Directiva de contraseñas)	`name: "Enforce password history"`
	`value: POLICY_DWORD`
	
	`name: "Maximum password age"`
	`value: TIME_DAY`
	
	`name: "Minimum password age"`
	`value: TIME_DAY`
	
	`name: "Minimum password length"`
	`value: POLICY_DWORD`
	
	`name: "Password must meet complexity requirements"`
	`value: POLICY_SET`

Account Lockout Policy (Directiva de bloqueo de cuentas)	`name: "Account lockout duration"`
	`value: TIME_MINUTE`
	`or`
	`name: "Account lockout duration"`
	`value: TIME_SECOND`
	
	`name: "Account lockout threshold"`
	`value: POLICY_DWORD`
	
	`name: "Reset lockout account counter after"`
	`value: TIME_MINUTE`
	
	`name: "Enforce user logon restrictions"`
	`value: POLICY_SET`

Kerberos Policy (Directiva Kerberos)	`name: "Maximum lifetime for service ticket"`
	`value: TIME_MINUTE`
	
	`name: "Maximum lifetime for user ticket"`
	`value: TIME_HOUR`
	
	`name: "Maximum lifetime for user renewal ticket"`

	value: TIME_DAY name: "Maximum tolerance for computer clock synchronization" value: TIME_MINUTE
Audit Policy (Directiva de auditoría)	name: "Audit account logon events" value: AUDIT_SET name: "Audit account management" value: AUDIT_SET name: "Audit directory service access" value: AUDIT_SET name: "Audit logon events" value: AUDIT_SET name: "Audit object access" value: AUDIT_SET name: "Audit policy change" value: AUDIT_SET name: "Audit privilege use" value: AUDIT_SET name: "Audit process tracking" value: AUDIT_SET name: "Audit system events" value: AUDIT_SET
Accounts (Cuentas)	name: "Accounts: Administrator account status" value: POLICY_SET name: "Accounts: Guest account status" value: POLICY_SET name: "Accounts: Limit local account use of blank password to console logon only" value: POLICY_SET name: "Accounts: Rename administrator account" value: POLICY_TEXT name: "Accounts: Rename guest account" value: POLICY_TEXT
Audit (Auditoría)	name: "Audit: Audit the access of global system objects" value: POLICY_SET name: "Audit: Audit the use of Backup and Restore privilege" value: POLICY_SET name: "Audit: Shut down system immediately if unable to log security audits" value: POLICY_SET

DCOM	name: "DCOM: Machine Launch Restrictions in Security Descriptor Definition Language (SDDL) syntax" value: POLICY_TEXT name: "DCOM: Machine Access Restrictions in Security Descriptor Definition Language (SDDL) syntax" value: POLICY_TEXT
Devices (Dispositivos)	name: "Devices: Allow undock without having to log on" value: POLICY_SET name: "Devices: Allowed to format and eject removable media" value: DASD_SET name: "Devices: Prevent users from installing printer drivers" value: POLICY_SET name: "Devices: Restrict CD-ROM access to locally logged-on user only" value: POLICY_SET name: "Devices: Restrict floppy access to locally logged-on user only" value: POLICY_SET name: "Devices: Unsigned driver installation behavior" value: DRIVER_SET
Domain controller (Controlador de dominio)	name: "Domain controller: Allow server operators to schedule tasks" value: POLICY_SET name: "Domain controller: LDAP server signing requirements" value: LDAP_SET name: "Domain controller: Refuse machine account password changes" value: POLICY_SET
Domain member (Miembro de dominio)	name: "Domain member: Digitally encrypt or sign secure channel data (always)" value: POLICY_SET name: "Domain member: Digitally encrypt secure channel data (when possible)" value: POLICY_SET name: "Domain member: Digitally sign secure channel data (when possible)" value: POLICY_SET name: "Domain member: Disable machine account password changes" value: POLICY_SET name: "Domain member: Maximum machine account password age" value: POLICY_DAY

	name: "Domain member: Require strong (Windows 2000 or later) session key" value: POLICY_SET
Interactive logon (Inicio de sesión interactivo)	name: "Interactive logon: Display user information when the session is locked" value: LOCKEDID_SET name: "Interactive logon: Do not display last user name" value: POLICY_SET name: "Interactive logon: Do not require CTRL+ALT+DEL" value: POLICY_SET name: "Interactive logon: Message text for users attempting to log on" value: POLICY_TEXT name: "Interactive logon: Message title for users attempting to log on" value: POLICY_TEXT name: "Interactive logon: Number of previous logons to cache (in case domain controller is not available)" value: POLICY_DWORD name: "Interactive logon: Prompt user to change password before expiration" value: POLICY_DWORD name: "Interactive logon: Require Domain Controller authentication to unlock workstation" value: POLICY_SET name: "Interactive logon: Require smart card" value: POLICY_SET name: "Interactive logon: Smart card removal behavior" value: SMARTCARD_SET
Microsoft network client (Cliente de red Microsoft)	name: "Microsoft network client: Digitally sign communications (always)" value: POLICY_SET name: "Microsoft network client: Digitally sign communications (if server agrees)" value: POLICY_SET name: "Microsoft network client: Send unencrypted password to third-party SMB servers" value: POLICY_SET
Microsoft network server (Servidor de red Microsoft)	name: "Microsoft network server: Amount of idle time required before suspending session" value: POLICY_DWORD name: "Microsoft network server: Digitally sign communications

	(always)" value: POLICY_SET name: "Microsoft network server: Digitally sign communications (if client agrees)" value: POLICY_SET name: "Microsoft network server: Disconnect clients when logon hours expire" value: POLICY_SET
Network access (Acceso a redes)	name: "Network access: Allow anonymous SID/Name translation" value: POLICY_SET name: "Network access: Do not allow anonymous enumeration of SAM accounts" value: POLICY_SET name: "Network access: Do not allow anonymous enumeration of SAM accounts and shares" value: POLICY_SET name: "Network access: Do not allow storage of credentials or .NET Passports for network authentication" value: POLICY_SET name: "Network access: Let Everyone permissions apply to anonymous users" value: POLICY_SET name: "Network access: Named Pipes that can be accessed anonymously" value: POLICY_MULTI_TEXT name: "Network access: Remotely accessible registry paths and sub-paths" value: POLICY_MULTI_TEXT name: "Network access: Remotely accessible registry paths" value: POLICY_MULTI_TEXT name: "Network access: Restrict anonymous access to Named Pipes and Shares" value: POLICY_SET name: "Network access: Shares that can be accessed anonymously" value: POLICY_MULTI_TEXT name: "Network access: Sharing and security model for local accounts" value: LOCALACCOUNT_SET
Network security (Seguridad de red)	name: "Network security: Do not store LAN Manager hash value on next password change" value: POLICY_SET name: "Network security: Force logoff when logon hours expire"

	value: POLICY_SET name: "Network security: LAN Manager authentication level" value: LANMAN_SET name: "Network security: LDAP client signing requirements" value: LDAPCLIENT_SET name: "Network security: Minimum session security for NTLM SSP based (including secure RPC) clients" value: NTLMSSP_SET name: "Network security: Minimum session security for NTLM SSP based (including secure RPC) servers" value: NTLMSSP_SET
Recovery console (Consola de recuperación)	name: "Recovery console: Allow automatic administrative logon" value: POLICY_SET name: "Recovery console: Allow floppy copy and access to all drives and all folders" value: POLICY_SET
Shutdown (Apagado)	name: "Shutdown: Allow system to be shut down without having to log on" value: POLICY_SET name: "Shutdown: Clear virtual memory pagefile" value: POLICY_SET
System cryptography (Criptografía del sistema)	name: "System cryptography: Force strong key protection for user keys stored on the computer" value: CRYPTO_SET name: "System cryptography: Use FIPS compliant algorithms for encryption, hashing, and signing" value: POLICY_SET
System objects (Objetos de sistema)	name: "System objects: Default owner for objects created by members of the Administrators group" value: OBJECT_SET name: "System objects: Require case insensitivity for non-Windows subsystems" value: POLICY_SET name: "System objects: Strengthen default permissions of internal system objects (e.g. Symbolic Links)" value: POLICY_SET
System settings (Configuración del sistema)	name: "System settings: Optional subsystems" value: POLICY_MULTI_TEXT name: "System settings: Use Certificate Rules on Windows Executables for Software Restriction Policies" value: POLICY_SET

Event Log (Registro de eventos)	name: "Maximum application log size" value: POLICY_KBYTE name: "Maximum security log size" value: POLICY_KBYTE name: "Maximum system log size" value: POLICY_KBYTE name: "Prevent local guests group from accessing application log" value: POLICY_SET name: "Prevent local guests group from accessing security log" value: POLICY_SET name: "Prevent local guests group from accessing system log" value: POLICY SET name: "Retain application log" value: POLICY_DAY name: "Retain security log" value: POLICY_DAY name: "Retain system log" value: POLICY_DAY name: "Retention method for application log" value: EVENT_METHOD name: "Retention method for security log" value: EVENT_METHOD name: "Retention method for system log" value: EVENT_METHOD

Presentación de informes forzada

Las directivas de auditoría se pueden forzar para generar un resultado específico mediante el uso de la palabra clave "**report**"(informe). Se pueden usar los tipos de informes PASSED (APROBÓ), FAILED (INCORRECTO) y WARNING (ADVERTENCIA). A continuación se incluye un ejemplo de directiva:

```
<report type: "WARNING">
 description: "Audit 103-a requires a physical inspection of the pod bay doors Hal"
</report>
```

El texto que se encuentra dentro del campo "**description**" (descripción) siempre aparecerá en el informe.

Este tipo de informe resulta de utilidad si desea informar a un auditor que Nessus no puede realizar una comprobación real. Por ejemplo, cabe la posibilidad de que exista un requisito de determinar que un sistema específico se protegió de manera física, y deseamos informar al auditor para que realice una comprobación o inspección manual. Este tipo de informe también es útil si el tipo especificado de auditoría que debe realizar Nessus no se determinó mediante una comprobación OVAL.

Condiciones

Es posible definir una lógica **`if/then/else`** en la directiva de Windows para que solo inicie una comprobación si las condiciones previas son válidas, o para agrupar varias pruebas en una sola.

La sintaxis para establecer condiciones es la siguiente:

```
<if>
 <condition type: "or">
 <Insert your audit here>
 </condition>
 <then>
 <Insert your audit here>
 </then>
 <else>
 <Insert your audit here>
 </else>
</if>
```

Las condiciones pueden ser del tipo "**`and`**" u "**`or`**".

La auditoría de la condición, las instrucciones "**`then`**" y "**`else`**" pueden ser una lista de elementos (o elementos personalizados), o bien una instrucción "**`if`**". Las instrucciones "**`else`**" y "**`then`**" también pueden emplear el tipo "**`report`**" para informar un resultado satisfactorio o de error de acuerdo con el valor devuelto de la condición:

```
<report type:"PASSED|FAILED">
 description: "the test passed (or failed)"
 (optional) severity: INFO|MEDIUM|HIGH
</report>
```

Un valor "**`if`**" devuelve SUCCESS (SIN ERRORES) o FAILURE (ERROR), y este valor se usa cuando la instrucción "**`if`**" se encuentra dentro de otra estructura "**`if`**". Por ejemplo, si se ejecuta la estructura **`<then>`**, el valor devuelto será uno de los siguientes:

- la auditoría solo contiene elementos: devuelve SUCCESS [SIN ERRORES] si se aprobaron todos los elementos; de lo contrario, devuelve FAILURE [ERROR]
- la auditoría solo contiene **`<report>`**: devuelve el tipo de informe
- la auditoría contiene elementos y **`<report>`**: devuelve el tipo de informe

Si se usa la instrucción **`<report>`** y el tipo es "FAILED" (INCORRECTO), entonces, el motivo por el que tuvo un error aparecerá en el informe junto con el nivel de gravedad, si se define.

A continuación se presenta un ejemplo en el que se realiza una auditoría de la directiva de contraseñas. Dado que se usa el tipo "**`and`**", para que esta directiva apruebe la auditoría ambos elementos personalizados deben ser aprobados. Este ejemplo prueba la existencia de una combinación muy extraña de directivas de historial de contraseñas válidas con el fin de ilustrar la forma en que se puede implementar una lógica de prueba sofisticada:

```
<if>
 <condition type:"and">
  <custom_item>
   type: PASSWORD_POLICY
   description: "2.2.2.5 Password History: 24 passwords remembered"
   value_type: POLICY_DWORD
   value_data: [22..MAX] || 20
```

```
   password_policy: ENFORCE_PASSWORD_HISTORY
  </custom_item>
  <custom_item>
   type: PASSWORD_POLICY
   description: "2.2.2.5 Password History: 24 passwords remembered"
   value_type: POLICY_DWORD
   value_data: 18 || [4..24]
   password_policy: ENFORCE_PASSWORD_HISTORY
  </custom_item>
 </condition>

 <then>
  <report type:"PASSED">
   description: "Password policy passed"
  </report>
 </then>

 <else>
  <report type:"FAILED">
   description: "Password policy failed"
  </report>
 </else>
</if>
```

En el ejemplo anterior solo se mostró el nuevo tipo "**report**", pero la estructura **if/then/else** admite la realización de auditorías adicionales dentro de las cláusulas "**else**". Dentro de una condición, también se pueden usar las cláusulas **if/then/else** anidadas. A continuación se muestra un ejemplo más complejo:

```
<if>
 <condition type:"and">
  <custom_item>
   type: CHECK_ACCOUNT
   description: "Accounts: Rename Administrator account"
   value_type: POLICY_TEXT
   value_data: "Administrator"
   account_type: ADMINISTRATOR_ACCOUNT
   check_type: CHECK_NOT_EQUAL
  </custom_item>
 </condition>

 <then>
  <report type:"PASSED">
   description: "Administrator account policy passed"
  </report>
 </then>

 <else>
  <if>
   <condition type:"or">
    <item>
     name: "Minimum password age"
     value: [1..30]
    </item>
    <custom_item>
     type: PASSWORD_POLICY
```

```
      description: "Password Policy setting"
      value_type: POLICY_SET
      value_data: "Enabled"
      password_policy: COMPLEXITY_REQUIREMENTS
     </custom_item>
    </condition>

    <then>
     <report type:"PASSED">
      description: "Administrator account policy passed"
     </report>
    </then>

    <else>
     <report type:"FAILED">
      description: "Administrator account policy failed"
     </report>
    </else>
   </if>

  </else>
 </if>
```

En este ejemplo, si no se cambió el nombre de la cuenta Administrator (Administrador), la auditoría comprobará si la vigencia mínima de la contraseña es de 30 días o menos. Esta directiva de auditoría se aprobará si se cambió el nombre de la cuenta Administrator (Administrador) independientemente de la directiva de contraseñas, y solo probará la directiva de vigencia de la contraseña si no se cambió el nombre de la cuenta Administrator (Administrador).

Referencia para archivos de compatibilidad de auditoría de contenido de Windows

Las comprobaciones **`.audit`** de Windows Content (Contenido de Windows) difieren de las comprobaciones **`.audit`** de Windows Configuration (Configuración de Windows), ya que están diseñadas para realizar en el sistema de archivos de Windows una búsqueda de tipos de archivos específicos que contengan datos confidenciales, en lugar de enumerar las opciones de configuración del sistema. Incluyen una variedad de opciones para ayudar al auditor a restringir los parámetros de búsqueda y así localizar y visualizar de manera más eficaz los datos no compatibles.

Uso de comillas:

Las comillas simples y dobles son indistintas al encerrar campos de auditoría, a excepción de los siguientes dos casos:

1. En comprobaciones de compatibilidad de Windows en las que se deban interpretar de forma literal los campos especiales como CRLF, etc., se deben usar comillas simples. Se deben escapar los campos incrustados que se interpretarán como cadenas.

Por ejemplo:
`expect: 'First line\r\nSecond line\r\nJohn\'s Line'`

2. Se requieren comillas dobles al utilizar “include_paths” y “exclude_paths” de WindowsFiles.

Si en cualquier tipo de campo (descripción, value_data, regex, etc.) se usan cadenas que contengan comillas simples o dobles, existen dos formas de tratarlas:

a. Use el tipo de comilla opuesto para las comillas de cierre exteriores.

Por ejemplo:
```
expect: "This is John's Line"
expect: 'We are looking for a double-quote-".*'
```

b. Escape las comillas incrustadas, si las hay, con una barra inversa (solo comillas dobles).

Por ejemplo:
```
expect: "\"Text to be searched\""
```

Tipo de comprobación

Todas las comprobaciones de compatibilidad de contenido de Windows deben estar entre corchetes con la encapsulación **`check_type`** y la designación "**`WindowsFiles`**". Esto es muy similar a todos los otros archivos **`.audit`**. El formato básico de un archivo de comprobación de contenido es el siguiente:

```
<check_type: "WindowsFiles">
<item>
</item>
<item>
</item>
<item>
</item>
</check_type>
```

Las comprobaciones reales de cada elemento no se muestran. Las siguientes secciones indican de qué forma se pueden usar diferentes palabras clave y parámetros para llenar una auditoría de elementos de contenido específico.

Formato del elemento

Uso
<item> type: FILE_CONTENT_CHECK description: ["value data"] file_extension: ["value data"] *(optional)* regex: ["value data"] *(optional)* expect: ["value data"] *(optional)* file_name: ["value data"] *(optional)* max_size: ["value data"] *(optional)* only_show: ["value data"] *(optional)* regex_replace: ["value data"] </item>

Cada uno de estos elementos se usa para auditar una amplia variedad de formatos de archivos, con una amplia variedad de tipos de datos. La siguiente tabla proporciona una lista de los tipos de datos compatibles. En la siguiente sección se incluyen numerosos ejemplos de cómo estas palabras clave se pueden usar en conjunto para auditar distintos tipos de contenido de archivos.

Palabra clave	Descripción
type	Siempre debe estar establecida en FILE_CONTENT_CHECK
description	Es la información que se usará como título para las vulnerabilidades de compatibilidad exclusivas en SecurityCenter. También será el primer conjunto de datos que informará Nessus.
file_extension	Enumera todas las extensiones deseadas que buscará Nessus. Las extensiones se enumeran sin el ".", entre comillas y separadas por barras verticales. Cuando en la auditoría no se incluyen opciones adicionales tales como **regex** y **expect**, los archivos con su extensión especificada (**file_extension**) aparecerán en los resultados de la auditoría.
regex	Esta palabra clave conserva la expresión regular usada para buscar tipos de datos complejos. Si la expresión regular coincide, el primer contenido coincidente aparecerá en el informe de vulnerabilidades.  La palabra clave **regex** debe ejecutarse con la palabra clave **expect** que se describe a continuación.  A diferencia de las Comprobaciones de compatibilidad de Windows, **regex** y **expect** en una Comprobación de compatibilidad de contenido de archivos de Windows no tienen que coincidir con las mismas cadenas de datos dentro del archivo en que se busca. Las comprobaciones de contenido de archivos de Windows simplemente requieren que las instrucciones **regex** y **expect** coincidan con los datos contenidos en los bytes `<max_size>` del archivo en que se busca.
expect	La instrucción **expect** se usa para enumerar uno o más patrones simples que deben estar en el documento para que coincida. Por ejemplo, al buscar números del Seguro Social es posible que se necesite la palabra "SSN", "SS#" o "Social". Los patrones múltiples aparecen entre comillas y separados por caracteres de barras verticales. En esta palabra clave también se admite la coincidencia de patrones simples con el punto. Al buscar coincidencias con la cadena "C.T", la instrucción **expect** haría coincidir "CAT", "CaT", "COT", "C T", etc.  La palabra clave **expect** puede ejecutarse de manera independiente en el caso de coincidencias con patrones simples. Sin embargo, si se usa la palabra clave **regex**, se necesita **expect**.  A diferencia de las Comprobaciones de compatibilidad de Windows, **regex** y **expect** en una Comprobación de compatibilidad de contenido de archivos de Windows no tienen que coincidir con las mismas cadenas de datos dentro del archivo en que se busca. Las comprobaciones de contenido de archivos de Windows simplemente requieren que las instrucciones **regex** y **expect** coincidan con los datos contenidos en

	los bytes `<max_size>` del archivo en que se busca.
file_name	Si bien es necesaria la palabra clave **file_extension**, esta puede restringir aun más la lista de archivos que se analizarán. Al proporcionar una lista de patrones, los archivos pueden desecharse o se puede establecer una coincidencia con ellos. Por ejemplo, esto facilita en gran medida la búsqueda de cualquier tipo de nombre de archivo que contenga términos tales como “employee” (empleado), “customer” (cliente) o “salary” (salario).
max_size	A los fines del rendimiento, es recomendable que la auditoría solo busque la primera parte de cada archivo. Esto se puede especificar en bytes mediante esta palabra clave. La cantidad de bytes se puede usar como argumento. También se admite una extensión “K” o “M” para kilobytes o megabytes, respectivamente.
only_show	Al buscar coincidencias de datos confidenciales tales como números de tarjetas de crédito, es posible que su organización necesite que solo los últimos cuatro dígitos sean visibles en el informe. Esta palabra clave admite la visualización de cualquier número de bytes especificado mediante una directiva.
regex_replace	Esta palabra clave controla qué patrón de la expresión regular aparecerá en el informe. Al buscar patrones de datos complejos, tales como números de tarjeta de crédito, no siempre es posible lograr que la primera coincidencia sean los datos deseados. Esta palabra clave proporciona más flexibilidad para capturar los datos deseados con mayor precisión.
include_paths	Esta palabra clave permite que se incluya dentro de los resultados de la búsqueda el directorio o la unidad. Se puede usar junto con la palabra clave “**exclude_paths**” o de manera independiente de esta. Esto resulta particularmente útil en los casos en los que, en un sistema de varias unidades, solo se pueden realizar búsquedas en determinadas unidades o carpetas. En los casos en los que se requieren varias rutas, estas tienen comillas dobles y están separadas por el símbolo de barra vertical. ⚠ Mediante la palabra clave “**include_paths**” solo se pueden especificar letras de unidades o nombres de carpetas. Los nombres de archivos no se pueden incluir en la cadena de valores “**include_paths**”.
exclude_paths	Esta palabra clave permite la exclusión de unidades, directorios o archivos de los resultados de la búsqueda. Se puede usar junto con la palabra clave “**include_paths**” o de manera independiente de esta. Esto resulta particularmente útil en los casos en los que se deben excluir de los resultados de la búsqueda una unidad, un directorio o un archivo en particular. En los casos en los que se requieren varias rutas, estas tienen comillas dobles y están separadas por el símbolo de barra vertical.

see_also	Esta palabra clave permite incluir enlaces a una referencia.
	
	Ejemplo:
	`see_also:`
	`"https://benchmarks.cisecurity.org/tools2/linux/CIS_Redhat_Linux_5_Benchmark_v2.0.0.pdf"`

solution	Esta palabra clave brinda una manera de incluir texto de “Solution” si está disponible. Ejemplo:

	`solution : "Remove this file, if its not required"`
reference	Esta palabra clave brinda una manera de incluir referencias cruzadas en **.audit**. El formato es “ref|ref-id1,ref|ref-id2”. Ejemplo: `reference : "CAT|CAT II,800-53|IA-5,8500.2|IAIA-1,8500.2|IAIA-2,8500.2|IATS-1,8500.2|IATS-2"`

Ejemplos de líneas de comandos

En esta sección crearemos un documento de texto ficticio con una extensión **.tns**,y luego ejecutaremos varios archivos **.audit** simples y complejos en función de él. A medida que analicemos cada ejemplo, probaremos cada caso compatible de los parámetros de contenido de Windows.

También usaremos el binario de la línea de comandos **nasl**. En el caso de cada archivo **.audit** que mostramos, puede incluirlos fácilmente en sus directivas de análisis de Nessus 4 o SecurityCenter. Sin embargo, para auditorías rápidas de un solo sistema, este método resulta muy eficaz. El comando que ejecutaremos cada vez desde el directorio **/opt/nessus/bin** será el siguiente:

```
# ./nasl -t <IP>
      /opt/nessus/lib/nessus/plugins/compliance_check_windows_file_content.nbin
```

<IP> es la dirección IP del sistema que auditaremos.

Con Nessus 4, al ejecutar **.nbin** (o cualquier otro plugin), se le solicitarán las credenciales del sistema de destino, además de la ubicación del archivo **.audit**.

Archivo de destino de prueba

El archivo de destino que usaremos posee el siguiente contenido:

```
abcdefghijklmnopqrstuvwxyz
01234567890
Tenable Network Security
SecurityCenter
Nessus
Passive Vulnerability Scanner
Log Correlation Engine
AB12CD34EF56
Nessus
```

Seleccione estos datos y cópielos en cualquier sistema de Windows al que tenga acceso mediante credenciales. Nombre el archivo “Tenable_Content.tns”.

Ejemplo 1: Búsqueda de documentos .tns que contengan la palabra “Nessus”

A continuación se ilustra un archivo **.audit** simple que busca cualquier archivo **.tns** que contenga la palabra “Nessus” en cualquier parte del documento.

```
<check_type:"WindowsFiles">
<item>
 type: FILE_CONTENT_CHECK
 description: "TNS File that Contains the word Nessus"
```

```
 file_extension: "tns"
 expect: "Nessus"
</item>
</check_type>
```

Al ejecutar este comando se espera el siguiente resultado:

```
"TNS File that Contains the word Nessus" : [FAILED]
 - error message:
The following files do not match your policy :
Share: C$, path: \share\new folder\tenable_content.tns
```

Estos resultados indican que encontramos una coincidencia. El informe indica que “tuvimos un error” porque encontramos datos que no buscábamos. Por ejemplo, si usted realiza una auditoría en busca de un número del Seguro Social y se obtuvo una coincidencia positiva del número del Seguro Social en el equipo público, a pesar de que esta coincidencia sea positiva se registrará como error por motivos de compatibilidad.

Ejemplo 2: Búsqueda de documentos `.tns` que contengan la palabra “France”

A continuación se indica un archivo **`.audit`** simple que busca cualquier archivo **`.tns`** que contenga la palabra “France” en cualquier parte del documento.

```
<check_type:"WindowsFiles">
<item>
 type: FILE_CONTENT_CHECK
 description: "TNS File that Contains the word France"
 file_extension: "tns"
 expect: "France"
</item>
</check_type>
```

Los resultados que obtenemos en esta ocasión son los siguientes:

```
"TNS File that Contains the word France" : [PASSED]
```

Pudimos “aprobar” la auditoría porque ninguno de los archivos **`.tns`** que auditamos contenía la palabra “France”.

Ejemplo 3: Búsqueda de documentos `.tns` y `.doc` que contengan la palabra “Nessus”

Resulta muy sencillo añadir una segunda extensión para realizar búsquedas de archivos de documentos de Microsoft Word. Esto se ilustra a continuación:

```
<check_type:"WindowsFiles">
<item>
 type: FILE_CONTENT_CHECK
 description: "TNS or DOC File that Contains the word Nessus"
 file_extension: "tns" | "doc"
 expect: "Nessus"
</item>
</check_type>
```

Los resultados (en nuestro equipo de prueba) fueron los siguientes:

```
"TNS or DOC File that Contains the word Nessus" : [FAILED]
 - error message:
The following files do not match your policy :
Share: C$, path: \share\new folder\tenable_content.tns
Share: C$, path: \documents and settings\jsmith\desktop\tns_roadmap.doc
```

Obtuvimos el mismo "failure" (error) que antes en nuestro archivo **.tns** de prueba, pero en este caso hubo un segundo archivo, que era un **.doc** que también contenía la palabra "Nessus". Si realiza estas pruebas en sus propios sistemas, puede tener o no un archivo Word que contenga la palabra "Nessus".

Ejemplo 4: Búsqueda de documentos .tns y .doc que contengan la palabra "Nessus" y un número de 11 dígitos

Ahora añadiremos nuestra primera expresión regular para que coincida con un número de 11 dígitos. Simplemente debemos añadir la expresión regular con la palabra clave **regex** al mismo archivo **.audit** de antes.

```
<check_type:"WindowsFiles">
<item>
 type: FILE_CONTENT_CHECK
 description: "TNS or DOC File that Contains the word Nessus"
 file_extension: "tns" | "doc"
 regex: " ([0-9]{11})"
 expect: "Nessus"
</item>
</check_type>
```

La ejecución de lo anterior produce los siguientes resultados:

```
"TNS or DOC File that Contains the word Nessus" : [FAILED]
 - error message:
The following files do not match your policy :
Share: C$, path: \share\new folder\tenable_content.tns (01234567890)
```

Aún se está buscando en el archivo **.doc** que coincidió en el último ejemplo. Dado que no posee un número de 11 dígitos, ya no aparece en los resultados. Tenga en cuenta además que, dado que usamos la palabra clave **regex**, también aparecerá una coincidencia en los datos.

¿Qué sucede si necesitamos encontrar un número de 10 dígitos? El número de 11 dígitos anterior contiene dos números de 10 dígitos (0123456789 y 1234567890). Si deseáramos escribir una coincidencia más exacta para solo 11 dígitos, lo que realmente deberíamos crear es una expresión regular que indique:

"Buscar cualquier número de 11 dígitos que no esté precedido ni seguido por ningún otro número".

Para hacer esto en expresiones regulares podemos añadir el operador "not" (no), como se ve a continuación:

```
<check_type:"WindowsFiles">
<item>
 type: FILE_CONTENT_CHECK
 description: "TNS or DOC File that Contains the word Nessus"
 file_extension: "tns" | "doc"
 regex: "([^0-9]|^)([0-9]{11})([^0-9]|$)"
 expect: "Nessus"
</item>
</check_type>
```

Leyendo de izquierda a derecha, también podemos ver el carácter "^" y el de signo de dólar varias veces. El "^" en ocasiones indica el comienzo de una línea, y otras veces significa buscar una coincidencia que sea negativa. El signo de dólar indica el final de una línea. La expresión regular anterior básicamente indica buscar cualquier patrón que no comience con un número, pero que posiblemente comience en una nueva línea, que contenga 11 números que no estén seguidos por ningún otro número o tenga un final de línea. Las expresiones regulares consideran el comienzo y el final de una línea como casos especiales; por ello, requieren el uso de los caracteres "^" o "$".

Ejemplo 5: Búsqueda de documentos `.tns` y `.doc` que contengan la palabra "Nessus" y un número de 11 dígitos, pero que solo muestren los últimos 4 bytes

Añadir la palabra clave **`only_show`** a nuestro archivo **`.audit`** puede limitar el resultado. Esto puede limitar a los auditores para que solo tengan acceso a los datos confidenciales que están buscando.

```
<check_type:"WindowsFiles">
<item>
 type: FILE_CONTENT_CHECK
 description: "TNS or DOC File that Contains the word Nessus"
 file_extension: "tns" | "doc"
 regex: "([^0-9]|^)([0-9]{11})([^0-9]|$)"
 expect: "Nessus"
 only_show: "4"
</item>
</check_type>
```

Cuando se encuentran coincidencias, los datos se ocultan mediante caracteres "X", como se muestra a continuación:

```
"TNS or DOC File that Contains the word Nessus" : [FAILED]
 - error message:
The following files do not match your policy :
Share: C$, path: \share\new folder\tenable_content.tns (XXXXXXX7890)
```

Ejemplo 6: Búsqueda de documentos .tns que contengan la palabra "Correlation" en los primeros 50 bytes

En este ejemplo examinaremos el uso de la palabra clave **`max_size`**. En nuestro archivo de prueba, la palabra "Correlation" está a más de 50 bytes del comienzo del archivo.

```
<check_type:"WindowsFiles">
<item>
 type: FILE_CONTENT_CHECK
 description: "TNS File that Contains the word Correlation"
 file_extension: "tns"
 expect: "Correlation"
 max_size: "50"
</item>
</check_type>
```

Al ejecutar lo anterior, obtenemos una coincidencia aprobada:

```
"TNS File that Contains the word Correlation" : [PASSED]
```

Cambie el valor **`max_size`** de "50" a "50 K" y vuelva a analizar. Ahora obtenemos un error:

```
"TNS File that Contains the word Correlation" : [FAILED]
 - error message:
The following files do not match your policy :
Share: C$, path: \share\new folder\tenable_content.tns
```

Ejemplo 7: Control de la información que aparece en los resultados

En este ejemplo examinaremos el uso de la palabra clave **regex_replace**. Considere el siguiente archivo **.audit**:

```
<check_type:"WindowsFiles">
<item>
 type: FILE_CONTENT_CHECK
 description: "Seventh Example"
 file_extension: "tns"
 regex: "Passive Vulnerability Scanner"
 expect: "Nessus"
</item>
</check_type>
```

Los resultados de esta comprobación son los siguientes:

```
"Seventh Example" : [FAILED]
 - error message:
The following files do not match your policy :
Share: C$, path: \share\new folder\tenable_content.tns (Passive Vulnerability Scanner)
```

Sin embargo, considere lo que puede suceder si realmente necesitáramos tener una expresión regular que coincidiera en las porciones “Passive” (Pasivo) y “Scanner” (Analizador) pero solo nos interesaran los resultados devueltos en la porción “Vulnerability” (Vulnerabilidades). Una nueva expresión regular tendría el siguiente aspecto:

```
<check_type:"WindowsFiles">
<item>
 type: FILE_CONTENT_CHECK
 description: "Seventh Example"
 file_extension: "tns"
 regex: "(Passive) (Vulnerability) (Scanner)"
 expect: "Nessus"
</item>
</check_type>
```

La comprobación aún devuelve la coincidencia completa de “Passive Vulnerability Scanner” (Analizador de vulnerabilidades pasivo), ya que la instrucción de la expresión regular trata la cadena completa como la primera coincidencia. Para obtener solo la segunda coincidencia, necesitamos añadir la palabra clave **regex_replace**.

```
<check_type:"WindowsFiles">
<item>
 type: FILE_CONTENT_CHECK
 description: "Seventh Example"
 file_extension: "tns"
 regex: "(Passive) (Vulnerability) (Scanner)"
 regex_replace: "\3"
 expect: "Nessus"
</item>
```

```
</check_type>
```

El resultado del análisis es el siguiente:

```
"Seventh Example" : [FAILED]
 - error message:
The following files do not match your policy :
Share: C$, path: \share\new folder\tenable_content.tns (Vulnerability)
```

Usamos “\3” para indicar el segundo elemento en nuestra coincidencia, porque el primero (“\1”) es la cadena completa. Si hubiéramos usado “\2” el resultado hubiera sido “Passive” (Pasivo), y si hubiéramos usado “\4”, “Scanner” (Analizador).

¿Por qué existe esta función? Al buscar patrones de datos complejos, tales como números de tarjeta de crédito, no siempre es posible lograr que la primera coincidencia sean los datos deseados. Esta palabra clave proporciona más flexibilidad al capturar los datos deseados con mayor precisión.

Ejemplo 8: Uso del nombre del archivo como filtro

Si tiene en cuenta el archivo **`.audit`** del tercer ejemplo, este devolvió un resultado tanto para un archivo **`.tns`** como para un archivo **`.doc`**.

```
<check_type:"WindowsFiles">
<item>
 type: FILE_CONTENT_CHECK
 description: "TNS or DOC File that Contains the word Nessus"
 file_extension: "tns" | "doc"
 expect: "Nessus"
</item>
</check_type>
```

Los resultados (en nuestro equipo de prueba) fueron los siguientes:

```
"TNS or DOC File that Contains the word Nessus" : [FAILED]
 - error message:
The following files do not match your policy :
Share: C$, path: \share\new folder\tenable_content.tns
Share: C$, path: \documents and settings\jsmith\desktop\tns_roadmap.doc
```

La palabra clave **`file_name`** también se puede usar para filtrar los archivos que deseamos o no. Al añadirla al archivo **`.audit`** y solicitarle que solo considere los archivos con la palabra “tenable” en el nombre, tendrá el siguiente aspecto:

```
<check_type:"WindowsFiles">
<item>
 type: FILE_CONTENT_CHECK
 description: "TNS or DOC File that Contains the word Nessus"
 file_extension: "tns" | "doc"
 file_name: "tenable"
 expect: "Nessus"
</item>
</check_type>
```

Los resultados son los siguientes:

```
"TNS or DOC File that Contains the word Nessus" : [FAILED]
 - error message:
The following files do not match your policy :
Share: C$, path: \share\new folder\tenable_content.tns
```

El archivo **.doc** coincidente no se encuentra presente porque no tenía la palabra “tenable” en su ruta.

La cadena coincidente es una expresión regular, por lo que puede ser muy flexible y coincidir con una amplia variedad de archivos que deseemos o no. Por ejemplo, podríamos haber usado la cadena “[Tt]enable” para que coincida con la palabra “Tenable” o “tenable”. Del mismo modo, si deseamos que coincida con una extensión o una extensión parcial, necesitamos escapar el punto con una barra diagonal, tal como “\.t”, para buscar cualquier extensión que comience con “t”.

Ejemplo 9: Uso de palabras clave de inclusión/exclusión

Las palabras clave “**include_paths**” y “**exclude_paths**” se pueden usar para filtrar las búsquedas en función de la letra de la unidad, el directorio, e incluso mediante la exclusión de nombres de archivos.

```
<item>
 type: FILE_CONTENT_CHECK
 description: "Does the file contain a valid VISA Credit Card Number"
 file_extension: "xls" | "pdf" | "txt"
 regex: "([^0-9-]|^)(4[0-9]{3}( |-|)([0-9]{4})( |-|)([0-9]{4})( |-|)([0-9]{4}))([^0-9-
       ]|$)"
 regex_replace: "\3"
 expect:"."
 max_size: "50K"
 only_show: "4"
 include_paths: "c:\" | "g:\" | "h:\"
 exclude_paths: "g:\dontscan"
</item>
```

Los resultados son los siguientes:

```
Windows File Contents Compliance Checks
"Determine if a file contains a valid VISA Credit Card Number" : [FAILED]
- error message:
The following files do not match your policy :
Share: C$, path: \documents and settings\administrator\desktop\ccn.txt
       (XXXXXXXXXXXX0552)


Nessus ID : 24760
```

Tenga en cuenta que los resultados no son diferentes de los resultados de búsquedas de contenidos de archivos estándar de Windows; sin embargo, no incluyen la ruta excluida. Si se incluye una única ruta mediante “**include_paths**” (por ejemplo, “**c:**”), todas las otras rutas se excluyen de forma automática. Además, si se excluye la letra de una unidad (por ejemplo, “**d:**”), pero se incluye una carpeta contenida en esa unidad (por ejemplo, “**d:\users**”), la palabra clave “**exclude_paths**” primará y no se buscará dentro de la unidad. Sin embargo, puede incluir una unidad **c:** y luego excluir una subcarpeta en la unidad (por ejemplo, **c:\users**:).

Auditoría de diferentes tipos de formatos de archivos

Se pueden realizar auditorías de archivos de cualquier extensión. Sin embargo, archivos tales como **.zip** y **.gz** no se pueden descomprimir sobre la marcha. Si su archivo está comprimido o si los datos están codificados de algún modo, tal vez no sea posible la búsqueda de patrones.

En el caso de los documentos que almacenan datos en formato Unicode, las rutinas de análisis sintáctico del archivo **.nbin** quitarán de las cadenas todos los bytes “NULL” que se encuentren.

De manera adicional, se admiten todas las versiones de documentos de Microsoft Office. Esto incluye las nuevas versiones codificadas que se añadieron con Office 2007, tales como **.xlsx** y **.docx**.

Por último, se incluye la compatibilidad de distintos tipos de formatos de archivos PDF. Tenable ha diseñado un analizador de PDF amplio que extrae las cadenas sin procesar para establecer coincidencias. A los usuarios solo les debe preocupar qué tipo de datos desean buscar en los archivos PDF.

Consideraciones de rendimiento

Existen varias ventajas y desventajas que cualquier organización debe tener en cuenta al modificar los archivos predeterminados **.audit** y al probarlos en redes activas:

- ¿Qué extensiones debemos buscar?
- ¿Cuántos datos se deben analizar?

Los archivos **.audit** no requieren la palabra clave **max_size**. En este caso, Nessus intenta recuperar el archivo completo, y continuará haciéndolo a menos que se encuentre una coincidencia en un patrón. Dado que estos archivos recorren la red, hay mayor tráfico en la red con estas auditorías que con los análisis típicos o la auditoría de configuraciones.

Si un SecurityCenter administra varios analizadores Nessus, los datos solo deben pasar desde el host de Windows analizado hasta el analizador que lleva a cabo la auditoría de vulnerabilidades.

Referencia para archivos de compatibilidad de auditoría de configuración de Cisco IOS

Esta sección describe el formato y las funciones de las comprobaciones de compatibilidad de Cisco IOS, y la fundamentación que subyace en cada opción.

Uso de comillas:

Las comillas simples y dobles son indistintas al encerrar campos de auditoría, a excepción de los siguientes dos casos:

1. En comprobaciones de compatibilidad de Windows en las que se deban interpretar de forma literal los campos especiales como CRLF, etc., se deben usar comillas simples. Se deben escapar los campos incrustados que se interpretarán como cadenas.

Por ejemplo:

```
expect: 'First line\r\nSecond line\r\nJohn\'s Line'
```

2. Se requieren comillas dobles al utilizar “include_paths” y “exclude_paths” de WindowsFiles.

Si en cualquier tipo de campo (descripción, value_data, regex, etc.) se usan cadenas que contengan comillas simples o dobles, existen dos formas de tratarlas:

a. Use el tipo de comilla opuesto para las comillas de cierre exteriores.

Por ejemplo:

```
expect: "This is John's Line"
expect: 'We are looking for a double-quote-".*'
```

b. Escape las comillas incrustadas, si las hay, con una barra inversa (solo comillas dobles).

Por ejemplo:
```
expect: "\"Text to be searched\""
```

Tipo de comprobación

Todas las comprobaciones de compatibilidad de Cisco IOS deben estar entre corchetes con la encapsulación **`check_type`** y la designación “Cisco”. Esto es obligatorio para diferenciar los archivos **`.audit`** diseñados específicamente para sistemas que usan el sistema operativo Cisco IOS, de otros tipos de auditorías de compatibilidad.

Ejemplo:

```
<check_type:"Cisco">
```

A diferencia de otros tipos de auditorías de compatibilidad, no se encuentran disponibles palabras clave de versión o tipo adicionales.

Palabras clave

La siguiente tabla indica la forma en que se puede usar cada palabra clave en las comprobaciones de compatibilidad de Cisco:

Palabra clave	Ejemplo de uso y configuración admitida
`type`	CONFIG_CHECK, CONFIG_CHECK_NOT y RANDOMNESS_CHECK “CONFIG_CHECK” determina si el elemento especificado existe en el resultado “show config” (Mostrar configuración) del Cisco IOS. De la misma forma, “CONFIG_CHECK_NOT” determina si el elemento especificado no existe. “RANDOMNESS_CHECK” se usa para llevar a cabo comprobaciones de complejidad de cadenas (por ejemplo, comprobaciones de contraseñas). Si especifica la búsqueda de un elemento (mediante un regex), le indicará si la cadena es lo suficientemente “aleatoria” (posee al menos ocho caracteres de longitud, con mayúsculas, minúsculas, al menos un dígito y al menos un carácter especial).  Los parámetros de aleatoriedad no se pueden configurar actualmente.
`description`	La palabra clave “**`description`**” proporciona la capacidad de añadir una breve descripción de la comprobación que se lleva a cabo. Se recomienda enfáticamente que el campo **`description`** sea exclusivo y que no haya comprobaciones diferentes que tengan el mismo campo **`description`**. El SecurityCenter de Tenable usa este campo para generar de manera automática un número de identificación de plugin exclusivo en función del campo **`description`**. Ejemplo: `description: "Forbid Remote Startup Configuration"`
`feature_set`	La palabra clave “**`feature_set`**”, que es similar a la palabra clave “**`system`**” en las comprobaciones de compatibilidad de Unix, comprueba la versión del Feature Set

	(Conjunto de características) del Cisco IOS y ejecuta la comprobación resultante, o bien omite la comprobación a raíz de un error de regex. Esto resulta útil en los casos en los que una comprobación solo se puede aplicar a sistemas que cuenten con un –Feature Set (Conjunto de características) en particular.
	
	Ejemplo:
	`<item>`
	`type: CONFIG_CHECK`
	`description: "Version Check"`
	`info: "SSH Access Control Check."`
	`feature_set: "K8" context:"line .*"`
	`item: "access-class [0-9]+ in"`
	`</item>`
	
	La comprobación anterior solo ejecutará la comprobación “item” si la versión del Feature Set (Conjunto de características) coincide con el regex especificado: (K8)
	
	En caso de producirse error al comprobar la versión del Feature Set (Conjunto de características), aparecerá un error similar al que se indica a continuación:
	
	`"Version Check" : [SKIPPED]`
	`Test defined for the feature set 'K8' whereas we are running C850-ADVSECURITYK9-M`

ios_version	La palabra clave “**ios_version**”, que es similar a la palabra clave “**system**” en las comprobaciones de compatibilidad de Unix, comprueba la versión del Cisco IOS y ejecuta la comprobación resultante o bien omite la comprobación a raíz de un error de regex. Esto resulta útil en los casos en los que una comprobación solo se puede aplicar a sistemas que cuenten con una versión de IOS en particular.
	
	Ejemplo:
	`<item>`
	`type: CONFIG_CHECK`
	`description: "Version Check"`
	`info: "SSH Access Control Check."`
	`ios_version: "12\.[5-9]"`
	`context: "line .*"`
	`item: "access-class [0-9]+ in"`
	`</item>`
	
	La comprobación anterior solo ejecutará la comprobación “item” si la versión del IOS coincide con el regex especificado: (12\.[5-9]).
	
	En caso de no aprobarse la comprobación de la versión del IOS, aparecerá un error similar al que se indica a continuación:
	
	`"Version Check" : [SKIPPED]`
	`Test defined for 12.[5-9] whereas we are running 12.4(15)T10`

info	La palabra clave “**info**” se usa para agregar una descripción más detallada a la comprobación que se lleva a cabo. La fundamentación para la comprobación podría ser una reglamentación, una dirección URL con más información, una directiva corporativa. etc. Se pueden añadir varios campos **info** en líneas independientes para que el texto adquiera formato de párrafo. No existe un límite preestablecido en cuanto a la cantidad de campos **info** que se pueden usar.

	 Cada etiqueta “**info**” debe escribirse en una línea independiente, sin saltos de línea. Si se requiere más de una línea (por ejemplo, por motivos de formato), añada etiquetas “**info**” adicionales.
	
	Ejemplo:
	`info: "Verify at least one local user exists and ensure"`
	`info: "all locally defined user passwords are protected"`
	`info: "by encryption."`

item	La palabra clave “**item**” especifica el elemento de configuración dentro de los resultados de “show config” que se auditarán. Ejemplo: `item: "transport input ssh"` En esta palabra clave se pueden usar expresiones regulares para filtrar los resultados de las coincidencias. Consulte la descripción de la palabra clave **regex** para obtener más detalles sobre la funcionalidad **regex**.
regex	La palabra clave “**regex**” posibilita la búsqueda de la opción de elemento de configuración que coincida con una expresión regular en particular. Ejemplo: `regex: "snmp-server community ([^ ]*) .*"` Los siguientes metacaracteres requieren tratamiento especial: + \ * ( ) ^ Escape doblemente estos caracteres con dos barras invertidas “\\”, o enciérrelos entre corchetes cuadrados “[]” si desea que se interpreten de forma literal. Otros caracteres como los siguientes necesitan solo una barra invertida para que se interpreten literalmente: . ? " ' Esto se relaciona con la forma en que el compilador trata estos caracteres.
min_occurrences	La palabra clave “**min_occurrences**” especifica la cantidad mínima necesaria de apariciones del artículo de configuración para aprobar la auditoría. Ejemplo: `min_occurrences: "3"`
max_occurrences	La palabra clave “**max_occurrences**” especifica la cantidad máxima permitida de apariciones del elemento de configuración para aprobar la auditoría. Ejemplo: `max_occurrences: "1"`
required	La palabra clave “**required**” se usa para especificar si el elemento auditado debe estar presente o no en el sistema remoto. Por ejemplo, si **required** está establecida en “NO” y el tipo de comprobación es “CONFIG_CHECK”, la comprobación se aprobará si el elemento de configuración existe o si el elemento de configuración no existe. Por otra parte, si **required** se estableció en “YES”, se producirá un error en la comprobación anterior. Ejemplo: `required: NO`

context	La palabra clave “**context**” resulta útil cuando existe más de una instancia de un elemento de configuración en particular. Por ejemplo, considere la siguiente configuración:
	
	`line con 0`
	` no modem enable`
	`line aux 0`
	` access-class 42 in`
	` exec-timeout 10 0`
	` no exec`
	`line vty 0 4`
	` exec-timeout 2 0`
	` password 7 15010X1C142222362G`
	` transport input ssh`
	
	Si desea probar un valor de una línea de serie en particular, usar la palabra clave **item** con “line” no será suficiente, ya que hay más de una opción “line”. Si usa “**context**”, solo se enfocará en el elemento de su interés. Por ejemplo:
	
	`context: "con 0"`
	
	Solo realizará una búsqueda mediante grep del siguiente elemento de configuración:
	
	`line con 0`
	` no modem enable`
	
	En esta palabra clave se pueden usar expresiones regulares para filtrar los resultados de las coincidencias. Consulte la descripción de la palabra clave **regex** para obtener más detalles sobre la funcionalidad **regex**.

Ejemplos de líneas de comandos

Esta sección proporciona algunos ejemplos de auditorías comunes usadas en comprobaciones de compatibilidad de Cisco IOS. El binario de la línea de comandos **nasl** se usa como método rápido para probar auditorías sobre la marcha. Cada archivo **.audit** que se demuestra a continuación se puede incluir fácilmente en las directivas de análisis de Nessus. Sin embargo, en el caso de las auditorías rápidas de un sistema, las pruebas de líneas de comandos son más eficaces. El comando que se ejecutará cada vez desde el directorio **/opt/nessus/bin** será el siguiente:

```
# ./nasl -t <IP> /opt/nessus/lib/nessus/plugins/cisco_compliance_check.nbin
```

<IP> es la dirección IP del sistema que se auditará.

Se solicita la contraseña “enable”:

```
Which file contains your security policy ? cisco_test.audit
SSH login to connect with : admin
How do you want to authenticate ? (key or password) [password]
SSH password :
Enter the 'enable' password to use :
```

Consulte a su administrador de Cisco para conocer los parámetros de inicio de sesión “enable” (habilitar) correctos.

Ejemplo 1: Búsqueda de una SNMP ACL definida

A continuación se indica un archivo **`.audit`** simple que busca una SNMP ACL “deny” (denegar). Si no se encuentra ninguna, la auditoría mostrará un mensaje de error. Esta comprobación solo se ejecutará si la versión de IOS del enrutador coincide con el regex especificado. De lo contrario, se omitirá la comprobación.

```
<check_type: "Cisco">

<item>
 type: CONFIG_CHECK
 description: "Require a Defined SNMP ACL"
 info: "Verify a defined simple network management protocol (SNMP) access control list
       (ACL) exists with rules for restricting SNMP access to the device."
 ios_version: "12\.[4-9]"
 item: "deny ip any any"
</item>

</check_type>
```

Al ejecutar este comando, se esperan de un sistema compatible los siguientes resultados:

```
"Require a Defined SNMP ACL" : [PASSED]

Verify a defined simple network management protocol (SNMP) access control list (ACL)
       exists with rules for restricting SNMP access to the device.
```

Una auditoría con errores devolvería los siguientes resultados:

```
"Require a Defined SNMP ACL" : [FAILED]

Verify a defined simple network management protocol (SNMP) access control list (ACL)
       exists with rules for restricting SNMP access to the device.

 - error message: deny ip any any not found in the configuration file
```

En este caso, la comprobación tuvo errores porque buscábamos una regla “deny ip” y no se encontró ninguna.

Ejemplo 2: Control de que el servicio “`finger`” se encuentre deshabilitado

El siguiente es un archivo **`.audit`** simple que busca el servicio “**`finger`**” no seguro en el enrutador remoto. Esta comprobación solo se ejecutará si la versión de IOS del enrutador coincide con el regex especificado. De lo contrario, se omitirá la comprobación. Si se encuentra el servicio, la auditoría mostrará un mensaje de error.

```
<check_type: "Cisco">

<item>
 type: CONFIG_CHECK_NOT
 description: "Forbid Finger Service"
 ios_version: "12\.[4-9]"
 info: "Disable finger server."
 item: "(ip|service) finger"
</item>

</check_type>
```

Al ejecutar este comando, se esperan de un sistema compatible los siguientes resultados:

```
"Forbid Finger Service" : [PASSED]

Disable finger server.
```

Una auditoría con errores devolvería los siguientes resultados:

```
"Forbid Finger Service" : [FAILED]
Disable finger server.
 - error message:
The following configuration line is set:
ip finger <----

Policy value:
(ip|service) finger
```

Ejemplo 3: Comprobación de aleatoriedad para verificar que las cadenas de comunidad SNMP y el control de acceso sean lo suficientemente aleatorios

El siguiente es un archivo **`.audit`** simple que busca cadenas de comunidad SNMP que no son lo suficientemente aleatorias. Si se encuentra una cadena de comunidad y no se determina que es lo suficientemente aleatoria, la auditoría mostrará un mensaje de error. Dado que la opción “required” (requerida) está establecida en “NO”, la comprobación aún se aprobará si no existe ninguna cadena de comunidad snmp-server. Esta comprobación solo se ejecutará si el enrutador usa el Feature Set (Conjunto de características) “K9”. De lo contrario, se omitirá la comprobación.

```
<check_type: "Cisco">

<item>
 type: RANDOMNESS_CHECK
 description: "Require Authorized Read SNMP Community Strings and Access Control"
 info: "Verify an authorized community string and access control is configured to
       restrict read access to the device."
 feature_set: "K9"
 regex: "snmp-server community ([^ ]*) .*"
 required: NO
</item>

</check_type>
```

Al ejecutar este comando, se esperan de un sistema compatible los siguientes resultados:

```
"Require Authorized Read SNMP Community Strings and Access Control" : [PASSED]

Verify an authorized community string and access control is configured to restrict
       read access to the device.
```

Una auditoría con errores devolvería los siguientes resultados:

```
"Require Authorized Read SNMP Community Strings and Access Control" : [FAILED]

Verify an authorized community string and access control is configured to restrict
       read access to the device.
```

```
 - error message:

The following configuration line does not contain a token deemed random enough:
snmp-server community foobar RO

The following configuration line does not contain a token deemed random enough:
snmp-server community public RO
```

En el caso anterior había dos cadenas, “foobar” y “public”, que no contaban con un token lo suficientemente aleatorio y que, por lo tanto, tuvieron error en la comprobación.

Ejemplo 4: Comprobación de contexto para verificar el control de acceso SSH

El siguiente es un archivo **`.audit`** simple que busca todos los elementos de configuración “line” mediante la palabra clave “**`context`**” y lleva a cabo un **`regex`** para determinar si se encuentra establecido el control de acceso SSH.

```
<check_type: "Cisco">

<item>
 type: CONFIG_CHECK
 description: "Require SSH Access Control"
 info: "Verify that management access to the device is restricted on all VTY lines."
 context: "line .*"
 item: "access-class [0-9]+ in"</item>
</item>

</check_type>
```

Al ejecutar este comando, se esperan de un sistema compatible los siguientes resultados:

```
"Require SSH Access Control" : [PASSED]

 Verify that management access to the device is restricted on all VTY lines.
```

Una auditoría con errores devolvería los siguientes resultados:

```
"Require SSH Access Control" : [FAILED]

 Verify that management access to the device is restricted on all VTY lines.

 - error message:
The following configuration is set:
line con 0
 exec-timeout 5 0
 no modem enable

Missing configuration: access-class [0-9]+ in

The following configuration is set:
line vty 0 4
 exec-timeout 5 0
 password 7 15010A1C142222362D
 transport input ssh
```

```
Missing configuration: access-class [0-9]+ in
```

En el caso anterior había dos cadenas que coincidían con el regex de la palabra clave "**context**" de "**line .***". Dado que ninguna línea contenía el regex "**item**", la auditoría devolvió un mensaje "FAILED" (INCORRECTO).

Condiciones

Es posible definir la lógica **if/then/else** en la directiva de auditoría de Cisco. Esto le permite al usuario final devolver un mensaje de advertencia en lugar de una aprobación o error en caso de que la auditoría sea aprobada.

La sintaxis para establecer condiciones es la siguiente:

```
<if>
 <condition type: "or">
 <Insert your audit here>
 </condition>
 <then>
 <Insert your audit here>
 </then>
 <else>
 <Insert your audit here>
 </else>
</if>
```

Ejemplo:

```
<if>
<condition type: "AND">
 <item>
  type: CONFIG_CHECK
  description: "Forbid Auxiliary Port"
  info: "Verify the EXEC process is disabled on the auxiliary (aux) port."
  context: "line aux "
  item: "no exec"
 </item>
 <item>
  type: CONFIG_CHECK_NOT
  description: "Forbid Auxiliary Port"
  info: "Verify the EXEC process is disabled on the auxiliary (aux) port."
  context: "line aux "
  item: "transport input [^n][^o]?[^n]?[^e]?$"
 </item>
</condition>
<then>
 <report type: "PASSED">
  description: "Forbid Auxiliary Port"
  info: "Verify the EXEC process is disabled on the auxiliary (aux) port."
 </report>
</then>
<else>
 <report type: "FAILED">
  description: "Forbid Auxiliary Port"
  info: "Verify the EXEC process is disabled on the auxiliary (aux) port."
</report>
</else>
```

```
</if>
```

Ya sea que la condición sea errónea o se apruebe, eso nunca aparecerá en el informe, ya que se trata de una comprobación “silent” (silenciosa).

Las condiciones pueden ser del tipo “**and**” u “**or**”.

Referencia para archivos de compatibilidad de auditoría de configuración de Juniper

Esta sección describe el formato y las funciones de las comprobaciones de compatibilidad de Juniper y la fundamentación que subyace en cada opción.

Uso de comillas:

Las comillas simples y dobles son indistintas al encerrar campos de auditoría, a excepción de los siguientes dos casos:

1. En comprobaciones de compatibilidad de Windows en las que se deban interpretar de forma literal los campos especiales como CRLF, etc., se deben usar comillas simples. Se deben escapar los campos incrustados que se interpretarán como cadenas.

Por ejemplo:
```
expect: 'First line\r\nSecond line\r\nJohn\'s Line'
```

2. Se requieren comillas dobles al utilizar “include_paths” y “exclude_paths” de WindowsFiles.

Si en cualquier tipo de campo (descripción, value_data, regex, etc.) se usan cadenas que contengan comillas simples o dobles, existen dos formas de tratarlas:

a. Use el tipo de comilla opuesto para las comillas de cierre exteriores.

Por ejemplo:
```
expect: "This is John's Line"
expect: 'We are looking for a double-quote-".*'
```

b. Escape las comillas incrustadas, si las hay, con una barra inversa (solo comillas dobles).

Por ejemplo:
```
expect: "\"Text to be searched\""
```

Tipo de comprobación: CONFIG_CHECK

Las comprobaciones de compatibilidad de Juniper están entre corchetes en encapsulación de **custom_item** y CONFIG_CHECK o SHOW_CONFIG_CHECK. Se tratan como cualquier otro archivo **.audit** y funcionan con sistemas que ejecutan el sistema operativo de Juniper (Junos). La comprobación CONFIG_CHECK está compuesta por dos o más palabras clave. Las palabras clave **type** y **description** son obligatorias, seguidas de una o más palabras clave. La comprobación funciona haciendo una auditoría de la configuración en el formato “set”.

La configuración en el formato “set” puede obtenerse anexando “display set” a la solicitud “show configuration”. Por ejemplo:

```
show configuration | display set
```

```
admin> show configuration | display set
set version 10.2R3.10
set system time-zone GMT
set system no-ping-record-route
set system root-authentication encrypted-password "$1$hSGSlnwfdsdfdfsdfsdf43534"
```

Palabras clave

La siguiente tabla indica la forma en que se puede usar cada palabra clave en las comprobaciones de compatibilidad de Juniper:

Palabra clave	Ejemplo de uso y configuración admitida
type	CHECK_CONFIG y SHOW_CHECK_CONFIG "CHECK_CONFIG" determina si el elemento de configuración especificado existe en el resultado de "show configuration" de Juniper en el formato "set". De la misma manera, "SHOW_CONFIG_CHECK" hace una auditoría de si el elemento de configuración existe en el resultado "show configuration" en el formato predeterminado.
description	La palabra clave "**description**" proporciona la capacidad de añadir una breve descripción de la comprobación que se lleva a cabo. Se recomienda enfáticamente que el campo **description** sea exclusivo y que no haya comprobaciones diferentes que tengan el mismo campo **description**. El SecurityCenter de Tenable usa este campo para generar de manera automática un número de identificación de plugin exclusivo en función del campo **description**. Ejemplo: `description: " 3.1 Disable Unused Interfaces"`

info	La palabra clave "**info**" se usa para agregar una descripción más detallada a la comprobación que se lleva a cabo. La fundamentación para la comprobación podría ser una reglamentación, una dirección URL con más información, una directiva corporativa. etc. Se pueden añadir varios campos **info** en líneas independientes para que el texto adquiera formato de párrafo. No existe un límite preestablecido en cuanto a la cantidad de campos **info** que se pueden usar.
	
	  Cada etiqueta "**info**" debe escribirse en una línea independiente, sin saltos de línea. Si se requiere más de una línea (por ejemplo, por motivos de formato), añada etiquetas "**info**" adicionales.
	
	Ejemplo:
	`info: "Review the the list of interfaces"`
	`info: "Disable unused interfaces"`

severity	La palabra clave "**severity**" especifica la gravedad de la comprobación que se está realizando. Ejemplo: `severity: MEDIUM` La gravedad puede ser HIGH (ALTA), MEDIUM (MEDIA) o LOW (BAJA).

regex	La palabra clave "**regex**" posibilita la búsqueda de la opción de elemento de configuración que coincida con una expresión regular en particular. Ejemplo: `regex: " set system syslog .+"` Los siguientes metacaracteres requieren tratamiento especial: + \ * ( ) ^ Escape doblemente estos caracteres con dos barras invertidas "\\", o enciérrelos entre corchetes cuadrados "[]" si desea que se interpreten de forma literal. Otros caracteres como los siguientes necesitan solo una barra invertida para que se interpreten literalmente: . ? " ' Esto se relaciona con la forma en que el compilador trata estos caracteres. Si una comprobación tiene una etiqueta "**regex**" establecida, pero no hay ninguna etiqueta "**expect**", "**not_expect**" o "**number_of_lines**" establecida, la comprobación solo notifica todas las líneas que coincidan con la regex.

expect	Esta palabra clave permite la auditoría del elemento de configuración que coincide con la etiqueta "**regex**"; si la etiqueta "**regex**" no se utiliza, busca la cadena "**expect**" en toda la configuración.
	
	Ejemplo:
	`expect: "syslog host 1.1.1.1"`
	
	La comprobación se aprueba siempre y cuando la línea de configuración encontrada por la "**regex**" coincida con la etiqueta "**expect**"; o, en el caso de que "**regex**" no esté establecida, se aprueba si la cadena "**expect**" se encuentra en la configuración.
	
	Ejemplo:
	`regex: "syslog host [0-9\.]+"`
	`expect: "syslog host 1.1.1.1"`
	
	En el caso anterior, la etiqueta "**expect**" garantiza que el syslog host esté establecido en 1.1.1.1.

not_expect	Esta palabra clave permite la búsqueda de los elementos de configuración que no deben estar en la configuración.
	
	Ejemplo:
	`not_expect: "syslog host 1.1.1.1"`
	
	Actúa de manera opuesta a "**expect**". La comprobación se aprueba si la línea de configuración encontrada por la "**regex**" no coincide con la etiqueta "**not_expect**"; o, en el caso de que la etiqueta "**regex**" no esté establecida, se aprueba siempre y cuando la cadena "**not_expect**" no se encuentre en la configuración.
	
	Ejemplo:
	`regex: "syslog host [0-9\.]+"`
	`not_expect: "syslog host 1.1.1.1"`
	
	En el caso anterior, la etiqueta "**not_expect**" garantiza que el syslog host no esté establecido en 1.1.1.1.

number_of_lines	Esta palabra clave permite probar la compatibilidad de una comprobación de auditoría en base a la cantidad de líneas coincidentes que devuelva la configuración.
	
	`<custom_item>`
	`type: CONFIG_CHECK`
	`description: "Syslog"`
	`regex: "syslog host [0-9\.]+"`
	`number_of_lines: "^1$"`
	`</custom_item>`
	
	En el caso anterior, la comprobación se aprobará siempre y cuando se devuelva solo una línea que coincida con la "**regex**".

Ejemplos de CONFIG_CHECK

Estos son ejemplos de uso de CONFIG_CHECK en un dispositivo con Juniper:

```
<custom_item>
 type: CONFIG_CHECK
 description: "Audit Syslog host message severity"
 regex: "syslog host [0-9\.]+"
 expect: "syslog host [0-9\.]+ 6 .+"
</custom_item>
```

```
<custom_item>
 type: CONFIG_CHECK
 description: "Audit Syslog host"
 regex: "syslog host [0-9\.]+"
 number_of_lines: "^1$"
</custom_item>
```

```
<custom_item>
 type: CONFIG_CHECK
 description: "Audit Syslog host"
 regex: "syslog host [0-9\.]+"
 not_expect: "syslog host 1.2.3.4"
</custom_item>
```

```
<custom_item>
 type: CONFIG_CHECK
 description: "Audit Syslog settings"
 regex: "syslog .+"
</custom_item>
```

Tipo de comprobación: SHOW_CONFIG_CHECK

Esta comprobación hace de diversas maneras una auditoría de las mismas opciones de configuración que la comprobación .audit CONFIG_CHECK. Sin embargo, el formato de la configuración bajo auditoría es diferente. SHOW_CONFIG_CHECK hace una auditoría de la configuración en su formato predeterminado.

Por ejemplo, esta es la configuración en el formato predeterminado:

```
admin> show configuration system syslog
user * {
    any emergency;
}
host 1.1.1.1 {
    any none;
}
file messages {
    any any;
    authorization info;
}
file interactive-commands {
    interactive-commands any;
}
```

Esta comprobación no se recomienda a menos que necesite una mayor flexibilidad que con CONFIG_CHECK. Como cada comprobación .audit SHOW_CONFIG_CHECK hace que se ejecute un comando individual en el dispositivo con Juniper, el proceso puede sobrecargar la CPU y necesitar más tiempo para completarse. Esta comprobación existe para brindar flexibilidad al auditor y admitir un caso de uso en el futuro en que una auditoría con CONFIG_CHECK podría ser ineficiente.

Palabras clave

La siguiente tabla indica la forma en que se puede usar cada palabra clave en las comprobaciones de compatibilidad de Junos. Tenga en cuenta que la compatibilidad de una comprobación puede determinarse comparando el resultado de la comprobación con las etiquetas "**expect**", "**not_expect**" o "**number_of_lines**". No puede haber más de una etiqueta de prueba de compatibilidad (es decir, solo puede estar "**expect**", "**not_expect**" o "**number_of_lines**", pero no "**expect**" y "**not_expect**").

Palabra clave	Ejemplo de uso y configuración admitida

hierarchy	Esta palabra clave les permite a los usuarios navegar a una jerarquía específica en la configuración de Junos.
	
	Ejemplo:
	`hierarchy: "interfaces"`
	
	La palabra clave de la jerarquía se anexa internamente al comando "show configuration" (mostrar configuración) en SHOW_CONFIG_CHECK. Por ejemplo:
	
	`<custom_item>`
	` type: SHOW_CONFIG_CHECK`
	` description: "3.6 Forbid Multiple Loopback Addresses"`
	` hierarchy: "interfaces"`
	`</custom_item>`
	
	La comprobación anterior es el equivalente de ejecutar:
	
	`show configuration interfaces`

property	Esta palabra clave les permite a los usuarios hacer una auditoría de una "**property**" específica en el dispositivo con Junos. De manera predeterminada,

	SHOW_CONFIG_CHECK hace una auditoría del comando “show configuration” (mostrar configuración) seguido de una o más palabras clave, como **match**, **except** y **find**. En el caso de que se establezca la palabra clave “**property**”, se hace una auditoría de la propiedad en particular.
	
	Ejemplo:
	`property: "ospf"`
	
	`<custom_item>`
	` type: SHOW_CONFIG_CHECK`
	` description: "4.3.1 Require MD5 Neighbor Authentication (where OSPF is used)"`
	` info: "Level 2, Scorable"`
	` property: "ospf"`
	` hierarchy: "interface detail"`
	` match: "Auth type MD5"`
	`</custom_item>`
	
	La comprobación anterior es el equivalente de ejecutar:
	
	`show ospf interface detail`
	
	Tenga en cuenta que el ejemplo anterior no ejecutó “show configuration” (mostrar configuración), como en los otros ejemplos.

find	Esta palabra clave encuentra la jerarquía de configuración adecuada en una comprobación **.audit** SHOW_CONFIG_CHECK.
	
	`find: "chap"`
	
	La palabra clave **find** se anexa a la solicitud “show configuration” (mostrar configuración).
	
	`<custom_item>`
	` type: SHOW_CONFIG_CHECK`
	` description: "3.8.2 Require CHAP Authentication if Incoming Map is Used"`
	` hierarchy: "interfaces"`
	` find: "chap"`
	` match: "access-profile"`
	`</custom_item>`
	
	La comprobación anterior es el equivalente de ejecutar:
	
	`show configuration interfaces | find "chap" | match "access-profile"`

match	Esta palabra clave busca líneas coincidentes en una comprobación .audit SHOW_CONFIG_CHECK.

	`match: "multihop"`
	
	La palabra clave **match** se anexa a la solicitud “show configuration” (mostrar configuración).
	
	`<custom_item>`
	`type: SHOW_CONFIG_CHECK`
	`description: "3.6 Forbid Multiple Loopback Addresses"`
	`hierarchy: "interfaces"`
	`match: "lo[0-9]"`
	`</custom_item>`
	
	La comprobación anterior es el equivalente de ejecutar:
	
	`show configuration interfaces | match "lo[0-9]"`

except	Esta palabra clave excluye determinadas líneas de la configuración en una comprobación .audit SHOW_CONFIG_CHECK.
	
	`except: "multihop"`
	
	La palabra clave **except** se anexa a la solicitud “show configuration” (mostrar configuración).
	
	`<custom_item>`
	`type: SHOW_CONFIG_CHECK`
	`description: "6.8.1 Require External Time Sources"`
	`hierarchy: "system ntp"`
	`match: "server"`
	`except: "boot-server"`
	`</custom_item>`
	
	La comprobación anterior es el equivalente de ejecutar:
	
	`show configuration system ntp | match "server" | except "boot-server"`

expect	Esta palabra clave permite la auditoría del elemento de configuración que coincide con la etiqueta “**regex**”; o, si la etiqueta “**regex**” no se utiliza, busca la cadena “**expect**” en toda la configuración. La comprobación se aprueba siempre y cuando la línea de configuración encontrada por la “**regex**” coincida con la etiqueta “**expect**”; o, en el caso de que “**regex**” no esté establecida, se aprueba si la cadena “**expect**” se encuentra en la configuración.
	
	`regex: "syslog host [0-9\.]+"`
	`expect: "syslog host 1.2.4.5"`
	
	En el caso anterior, la etiqueta “**expect**” garantiza que la complejidad esté establecida

	en un valor entre 1 y 4. `expect: "syslog host"` En el caso anterior, la etiqueta "**expect**" garantiza que la complejidad esté establecida en 4.

not_expect	Esta palabra clave permite la búsqueda de los elementos de configuración que no deben estar en la configuración.
	
	Actúa de manera opuesta a "**expect**". La comprobación se aprueba si la línea de configuración encontrada por la "**regex**" no coincide con la etiqueta "**not_expect**"; o, en el caso de que la etiqueta "**regex**" no esté establecida, se aprueba siempre y cuando la cadena "**not_expect**" no se encuentre en la configuración.
	
	`regex: "syslog host [0-9\.]+"`
	`not_expect: "syslog host 1.2.3.4"`
	
	`not_expect: "syslog host"`

number_of_lines	Esta palabra clave permite probar la compatibilidad de una comprobación .audit en base a la cantidad de líneas coincidentes que devuelva la configuración.
	
	`<custom_item>`
	`type: CONFIG_CHECK`
	`description: "Syslog"`
	`regex: "syslog host [0-9\.]+"`
	`number_of_lines: "^1$"`
	`</custom_item>`
	
	En el caso anterior, la comprobación se aprobará siempre y cuando se devuelva solo una línea que coincida con la "**regex**".

Ejemplos de SHOW_CONFIG_CHECK

Estos son ejemplos de uso de SHOW_CONFIG_CHECK en un dispositivo con Juniper:

```
<custom_item>
 type: SHOW_CONFIG_CHECK
 description: "6.1.2 Require Accounting of Logins & Configuration Changes"
 hierarchy: "system accounting"
 find: "accounting"
 expect: "events [change-log login];"
</custom_item>
```

```
<custom_item>
 type: SHOW_CONFIG_CHECK
 description: "6.2.2 Require Archive Site"
 hierarchy: "system archival configuration archive-sites"
```

```
 match: "scp://"
 number_of_lines: "^([1-9]|[0-9][0-9]+)+$"
</custom_item>
```

```
<custom_item>
 type: SHOW_CONFIG_CHECK
 description: "4.7.1 Require BFD Authentication (where BFD is used)"
 hierarchy: "protocols"
 match: "authentication"
 except: "loose"
 number_of_lines: "^2$"
 check_option: CAN_BE_NULL
</custom_item>
```

```
<custom_item>
 type: SHOW_CONFIG_CHECK
 description: "4.3.1 Require MD5 Neighbor Authentication (where OSPF is used)"
 property: "ospf"
 hierarchy: "interface detail"
 match: "Auth type MD5"
 number_of_lines: "^([1-9]|[0-9][0-9]+)+$"
 check_option: CAN_BE_NULL
</custom_item>
```

Condiciones

Es posible definir una lógica **`if`**/**`then`**/**`else`** en la directiva de auditoría de Juniper. Esto permite que el usuario final use un único archivo que puede manejar varias configuraciones.

La sintaxis para establecer condiciones es la siguiente:

```
<if>
  <condition type:"or">
      < Insert your audit here >
  </condition>
 <then>
      < Insert your audit here >
 </then>
 <else>
      < Insert your audit here >
 </else>
</if>
```

Ejemplo:

```
<if>
 <condition type: "OR">

<custom_item>
 type: CONFIG_CHECK
 description: "Configure Syslog Host"
```

```
 regex: "syslog host [0-9\.]+"
 not_expect: "syslog host 1.2.3.4"
</custom_item>

 </condition>
 <then>
  <report type: "PASSED">
   description: "Configure Syslog Host."
  </report>
 </then>
 <else>
<custom_item>
  type: CONFIG_CHECK
  description: "Configure Syslog Host"
  regex: "syslog host [0-9\.]+"
  not_expect: "syslog host 1.2.3.4"
</custom_item>

 </else>
</if>
```

La condición nunca aparece en el informe; es decir, sin importar si falla o se aprueba, no aparecerá (es una comprobación “silent” [silenciosa]).

Las condiciones pueden ser del tipo “**and**” u “**or**”.

Informes

Pueden realizarse en <then> o <else> para lograr una condición deseada PASSED/FAILED (APROBÓ/INCORRECTO).

```
<if>
 <condition type: "OR">
 <custom_item>
  type: CONFIG_CHECK
  description: "Configure Syslog Host"
  regex: "syslog host [0-9\.]+"
  not_expect: "syslog host 1.2.3.4"
</custom_item>
 </condition>
 <then>
  <report type: "PASSED">
   description: "Configure Syslog host"
  </report>
 </then>
 <else>
 <report type: "FAILED">
   description: "Configure Syslog host"
  </report>
 </else>
</if>
```

PASSED (APROBÓ), WARNING (ADVERTENCIA) y FAILED (INCORRECTO) son los valores aceptables para “report type” (tipo de informe).

Referencia para archivos de compatibilidad de auditoría de configuración de Check Point GAiA

Esta sección describe el formato y las funciones de las comprobaciones de compatibilidad de Check Point GAiA y la fundamentación que subyace en cada opción.

Uso de comillas:

Las comillas simples y dobles son indistintas al encerrar campos de auditoría, a excepción de los siguientes dos casos:

1. En comprobaciones de compatibilidad de Windows en las que se deban interpretar de forma literal los campos especiales como CRLF, etc., se deben usar comillas simples. Se deben escapar los campos incrustados que se interpretarán como cadenas.

Por ejemplo:
```
expect: 'First line\r\nSecond line\r\nJohn\'s Line'
```

2. Se requieren comillas dobles al utilizar “include_paths” y “exclude_paths” de WindowsFiles.

Si en cualquier tipo de campo (descripción, value_data, regex, etc.) se usan cadenas que contengan comillas simples o dobles, existen dos formas de tratarlas:

a. Use el tipo de comilla opuesto para las comillas de cierre exteriores.

Por ejemplo:
```
expect: "This is John's Line"
expect: 'We are looking for a double-quote-".*'
```

b. Escape las comillas incrustadas, si las hay, con una barra inversa (solo comillas dobles).

Por ejemplo:
```
expect: "\"Text to be searched\""
```

Tipo de comprobación: CONFIG_CHECK

Las comprobaciones de compatibilidad de Check Point están entre corchetes en encapsulación de **`custom_item`** y CONFIG_CHECK. Se tratan como cualquier otro archivo **`.audit`**, y funcionan con sistemas que ejecutan el sistema operativo Check Point GAiA. La comprobación CONFIG_CHECK está compuesta por dos o más palabras clave. Las palabras clave **`type`** y **`description`** son obligatorias, seguidas de una o más palabras clave. La comprobación funciona haciendo una auditoría del resultado del comando “**`show config`**”, que se encuentra en el formato “set” de manera predeterminada.

Palabras clave

La siguiente tabla indica la forma en que se puede usar cada palabra clave en las comprobaciones de compatibilidad de GAiA:

Palabra clave	Ejemplo de uso y configuración admitida
`type`	“CHECK_CONFIG” determina si el elemento de configuración especificado existe en el resultado de “show configuration” (mostrar configuración) de GAiA.
`description`	La palabra clave “**`description`**” proporciona la capacidad de añadir una breve descripción de la comprobación que se lleva a cabo. Se recomienda enfáticamente

	que el campo **description** sea exclusivo y que no haya comprobaciones diferentes que tengan el mismo campo **description**. El SecurityCenter de Tenable usa este campo para generar de manera automática un número de identificación de plugin exclusivo en función del campo **description**. Ejemplo: `description: "1.0 Require strong Password Controls - 'min-password-length >= 8'"`
info	La palabra clave "**info**" se usa para agregar una descripción más detallada a la comprobación que se lleva a cabo. La fundamentación para la comprobación podría ser una reglamentación, una dirección URL con más información, una directiva corporativa. etc. Se pueden añadir varios campos **info** en líneas independientes para que el texto adquiera formato de párrafo. No existe un límite preestablecido en cuanto a la cantidad de campos **info** que se pueden usar.  Cada etiqueta "**info**" debe escribirse en una línea independiente, sin saltos de línea. Si se requiere más de una línea (por ejemplo, por motivos de formato), añada etiquetas "**info**" adicionales. Ejemplo: `info: "Enable palindrome-check on passwords"`
severity	La palabra clave "**severity**" especifica la gravedad de la comprobación que se está realizando. Ejemplo: `severity: MEDIUM` La gravedad puede ser HIGH (ALTA), MEDIUM (MEDIA) o LOW (BAJA).
regex	La palabra clave "**regex**" posibilita la búsqueda de la opción de elemento de configuración que coincida con una expresión regular en particular. Ejemplo: `regex: "set snmp .+"` Los siguientes metacaracteres requieren tratamiento especial: + \ * ( ) ^ Escape doblemente estos caracteres con dos barras invertidas "\\", o enciérrelos entre corchetes cuadrados "[]" si desea que se interpreten de forma literal. Otros caracteres como los siguientes necesitan solo una barra invertida para que se interpreten literalmente: . ? " ' Esto se relaciona con la forma en que el compilador trata estos caracteres. Si una comprobación tiene una etiqueta "regex" establecida, pero no hay ninguna etiqueta "expect", "not_expect" o "number_of_lines" establecida, la comprobación solo notifica todas las líneas que coincidan con la regex.
expect	Esta palabra clave permite la auditoría del elemento de configuración que coincide con la etiqueta "**regex**"; si la etiqueta "**regex**" no se utiliza, busca la cadena "**expect**" en toda la configuración. La comprobación se aprueba siempre y cuando la línea de configuración encontrada

	por la “**regex**” coincida con la etiqueta “**expect**”; o, en el caso de que “**regex**” no esté establecida, se aprueba si la cadena “**expect**” se encuentra en la configuración.
	
	Ejemplo:
	`regex: "set password-controls complexity"`
	`expect: "set password-controls complexity [1-4]"`
	
	En el caso anterior, la etiqueta “**expect**” garantiza que la complejidad esté establecida en un valor entre 1 y 4.

not_expect	Esta palabra clave permite la búsqueda de los elementos de configuración que no deben estar en la configuración.
	
	Actúa de manera opuesta a “**expect**”. La comprobación se aprueba si la línea de configuración encontrada por la “**regex**” no coincide con la etiqueta “**not_expect**”; o, en el caso de que la etiqueta “**regex**” no esté establecida, se aprueba siempre y cuando la cadena “**not_expect**” no se encuentre en la configuración.
	
	Ejemplo:
	`regex: "set password-controls password-expiration"`
	`not_expect: "set password-controls password-expiration never"`
	
	En el caso anterior, la etiqueta “**not_expect**” garantiza que los controles de contraseña no estén establecidos en “never”.

Ejemplos de CONFIG_CHECK

Estos son ejemplos de uso de CONFIG_CHECK en un dispositivo con Check Point:

```
<custom_item>
 type: CONFIG_CHECK
 description: "1.0 Require strong Password Controls - 'min-password-length >= 8'"
 regex: "set password-controls min-password-length"
 expect: "set password-controls min-password-length ([8-9]|[0-9][0-9]+)"
 info: "Require Password Lengths greater than or equal to 8."
</custom_item>
```

```
<custom_item>
 type: CONFIG_CHECK
 description: "1.0 Require strong Password Controls - 'password-expiration != never'"
 regex: "set password-controls password-expiration"
 not_expect: "set password-controls password-expiration never"
 info: "Allow passwords to expire"
</custom_item>
```

```
<custom_item>
 type: CONFIG_CHECK
 description: "2.13 Secure SNMP"
 regex: "set snmp .+"
 severity: MEDIUM
 info: "Manually review SNMP settings."
</custom_item>
```

Condiciones

Es posible definir una lógica **if/then/else** en la directiva de auditoría de Check Point. Esto permite que el usuario final use un único archivo que puede manejar varias configuraciones.

La sintaxis para establecer condiciones es la siguiente:

```
<if>
  <condition type:"or">
      < Insert your audit here >
  </condition>
 <then>
      < Insert your audit here >
 </then>
 <else>
      < Insert your audit here >
 </else>
</if>
```

Ejemplo:

```
<if>
 <condition type: "OR">
 <custom_item>
  type: CONFIG_CHECK
  description: "2.6 Install and configure Encrypted Connections to devices - 'telnet'"
  regex: "set net-access telnet"
  expect: "set net-access telnet off"
  info: "Do not use plain-text protocols."
</custom_item>
 </condition>
 <then>
  <report type: "PASSED">
   description: "Telnet is disabled"
  </report>
 </then>
 <else>
  <custom_item>
  type: CONFIG_CHECK
  description: "2.6 Install and configure Encrypted Connections to devices - 'telnet'"
  regex: "set net-access telnet"
  expect: "set net-access telnet off"
  info: "Do not use plain-text protocols."
</custom_item>
 </else>
</if>
```

La condición nunca aparece en el informe; es decir, sin importar si falla o se aprueba, no aparecerá (es una comprobación “silent” [silenciosa]).

Las condiciones pueden ser del tipo “**and**” u “**or**”.

Informes

Pueden realizarse en <then> o <else> para lograr una condición deseada PASSED/FAILED (APROBÓ/INCORRECTO).

```
<if>
 <condition type: "OR">
 <custom_item>
  type: CONFIG_CHECK
  description: "2.6 Install and configure Encrypted Connections to devices - 'telnet'"
  regex: "set net-access telnet"
  expect: "set net-access telnet off"
  info: "Do not use plain-text protocols."
</custom_item>
 </condition>
 <then>
  <report type: "PASSED">
   description: "Telnet is disabled"
  </report>
 </then>
 <else>
 <report type: "FAILED">
   description: "Telnet is disabled"
  </report>
 </else>
</if>
```

PASSED (APROBÓ), WARNING (ADVERTENCIA) y FAILED (INCORRECTO) son los valores aceptables para "report type" (tipo de informe).

Referencia para archivos de compatibilidad de auditoría de configuración de Firewall de Palo Alto

La comprobaciones de compatibilidad de Palo Alto son diferentes que otras auditorías de compatibilidad. Una diferencia importante en estas auditorías es el uso intensivo de `XSL Transforms` (XSLT) para extraer paquetes de información relevante (para obtener más información, consulte el Apéndice C). Las respuestas del Firewall de Palo Alto son en formato XML para la mayoría de las solicitudes de API, haciendo que XSLT sea el método más eficiente para auditar. Si no está familiarizado con XSLT, puede verlo como una manera de hacer una consulta a un archivo de XML para extraer la información que quiera en el formato que desee. En términos sencillos, XSLT es lo que SQL es para las bases de datos.

La auditoría de Palo Alto admite dos tipos de comprobaciones: AUDIT_XML y AUDIT_REPORTS.

AUDIT_XML

El siguiente es un ejemplo de una comprobación de Palo Alto AUDIT_XML:

```
<custom_item>
 type: AUDIT_XML
 description: "Palo Alto Security Settings - 'fips-mode = on'"
 info: "Fips-mode should be enabled."
 api_request_type: "op"
 request: "<show><fips-mode></fips-mode></show>"
 xsl_stmt: "<xsl:template match=\"/\">"
 xsl_stmt: " <xsl:apply-templates select=\"//result\"/>"
 xsl_stmt: "</xsl:template>"
 xsl_stmt: "<xsl:template match=\"//result\">"
 xsl_stmt: "fips-mode: <xsl:value-of select=\"text()\"/>"
 regex: "fips-mode:[\\s\\t]+"
 expect : "fips-mode:[\\s\\t]+on"
</custom_item>
```

Esta auditoría tiene cuatro partes básicas:

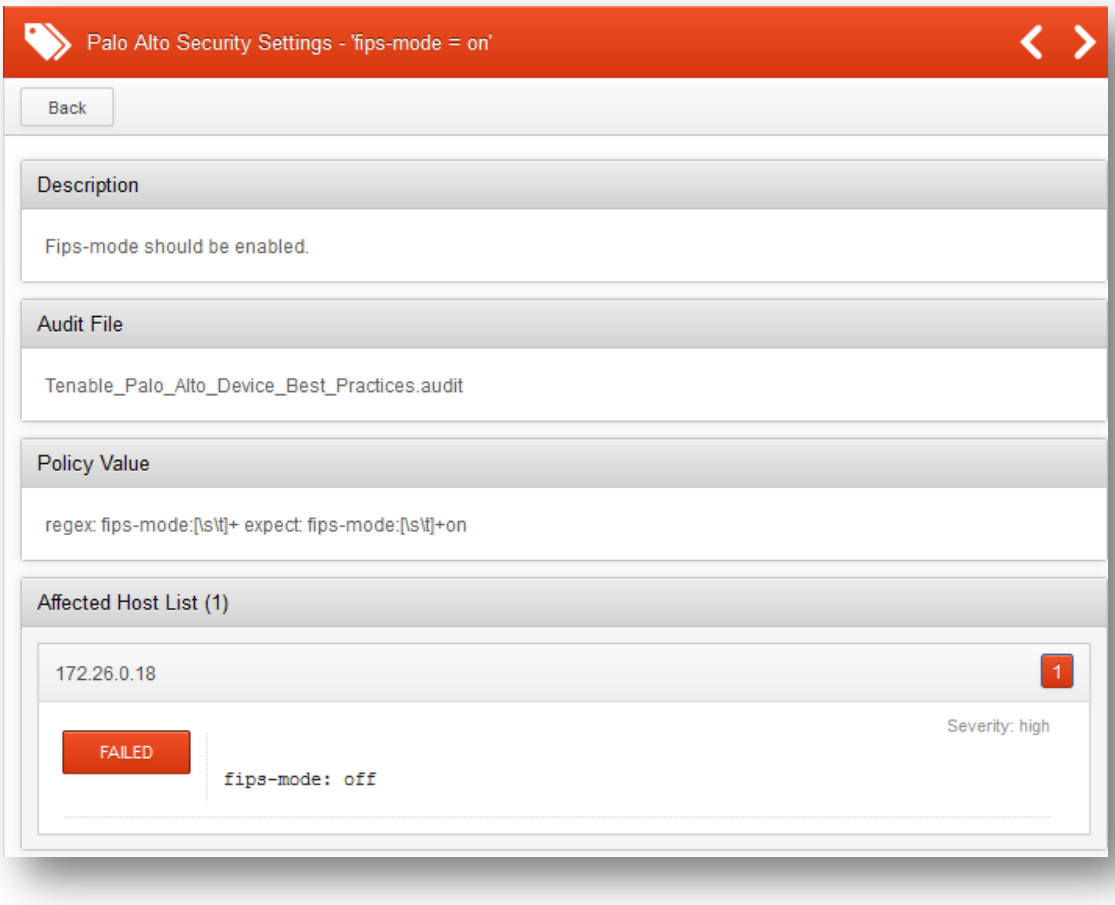
1. **`type`** describe el tipo de auditoría (en esta caso audita el XML), y **`description`** es una descripción de la auditoría. La palabra clave **`info`** provee una manera de incluir texto relevante en el informe.

2. **`api_request_type`** describe el tipo de solicitud (op == configuración operativa), y la solicitud es la solicitud real que finalmente ejecutamos. Actualmente, este es el único tipo de solicitud admitida.

3. La palabra clave **`xsl_stmt`** nos da una manera de definir el XSL Transform que aplicaremos en el XML devuelto después de ejecutar la solicitud de API.

4. Por último, las palabras clave **`regex`** y **`expect`** nos permiten hacer auditorías de compatibilidad/configuración.

La comprobación de ejemplo anterior generará el siguiente informe en Nessus:



AUDIT_REPORTS

Una de las características beneficiosas de un Firewall de Palo Alto es que hace un perfil continuo de su red y genera a diario más de 40 informes predefinidos, como Top Applications (Aplicaciones más usadas), Top Attackers (Atacantes más frecuentes) y Spyware Infected Hosts (Hosts infectados con spyware). Los administradores también pueden generar informes dinámicos a su criterio (por ejemplo, en la última hora). Nessus ahora puede consultar directamente estos informes e incluirlos en un informe de Nessus.

Esta característica tiene dos beneficios. Primero, los usuarios no tienen que alternar entre diferentes interfaces para obtener la misma información. Segundo, esto nos da la capacidad para auditar el informe. Por ejemplo, si no quiere que

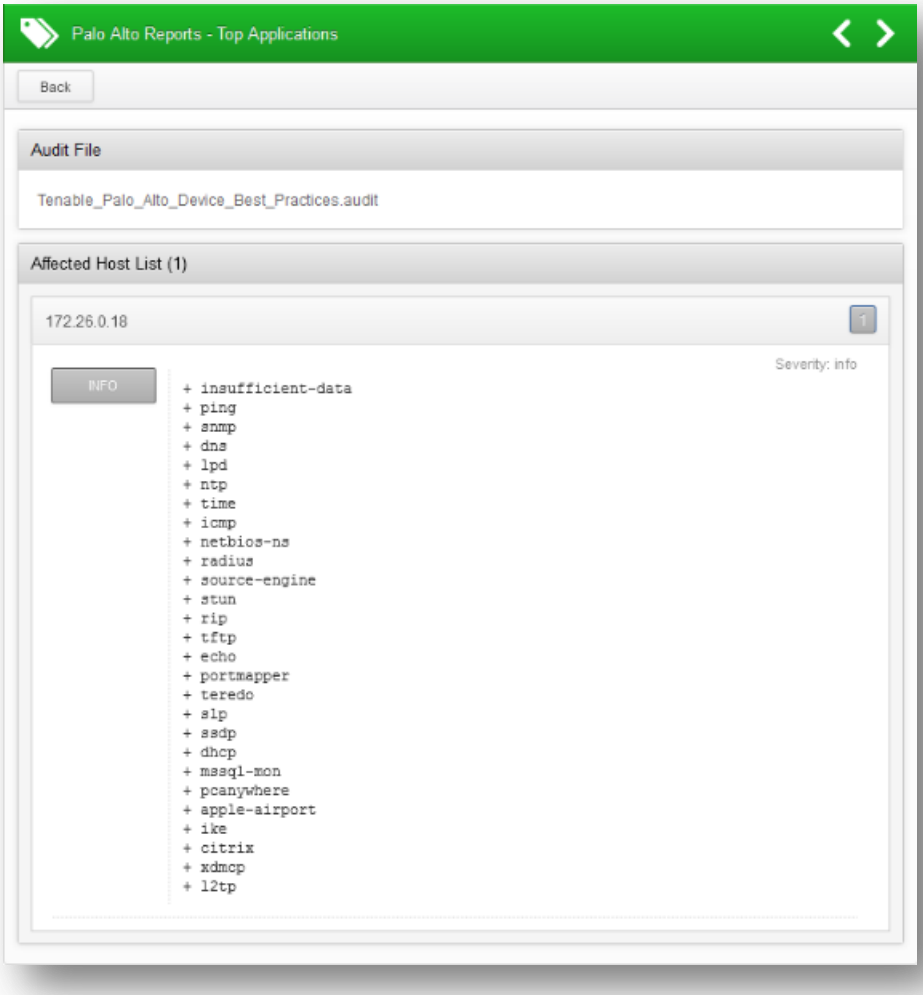
Facebook sea una aplicación utilizada en la red, los administradores pueden generar un informe desaprobado si Facebook aparece en el informe Top Applicationes (Aplicaciones más usadas). Por ejemplo:

```
<custom_item>
 type: AUDIT_REPORTS
 description: "Palo Alto Reports - Top Applications"
 request: "&reporttype=predefined&reportname=top-applications"
 xsl_stmt: "<xsl:template match=\"result\">"
 xsl_stmt: "<xsl:for-each select=\"entry\">"
 xsl_stmt: "+ <xsl:value-of select=\"name\"/>"
 xsl_stmt: "</xsl:for-each>"
 check_option: CAN_BE_NULL
</custom_item>
```

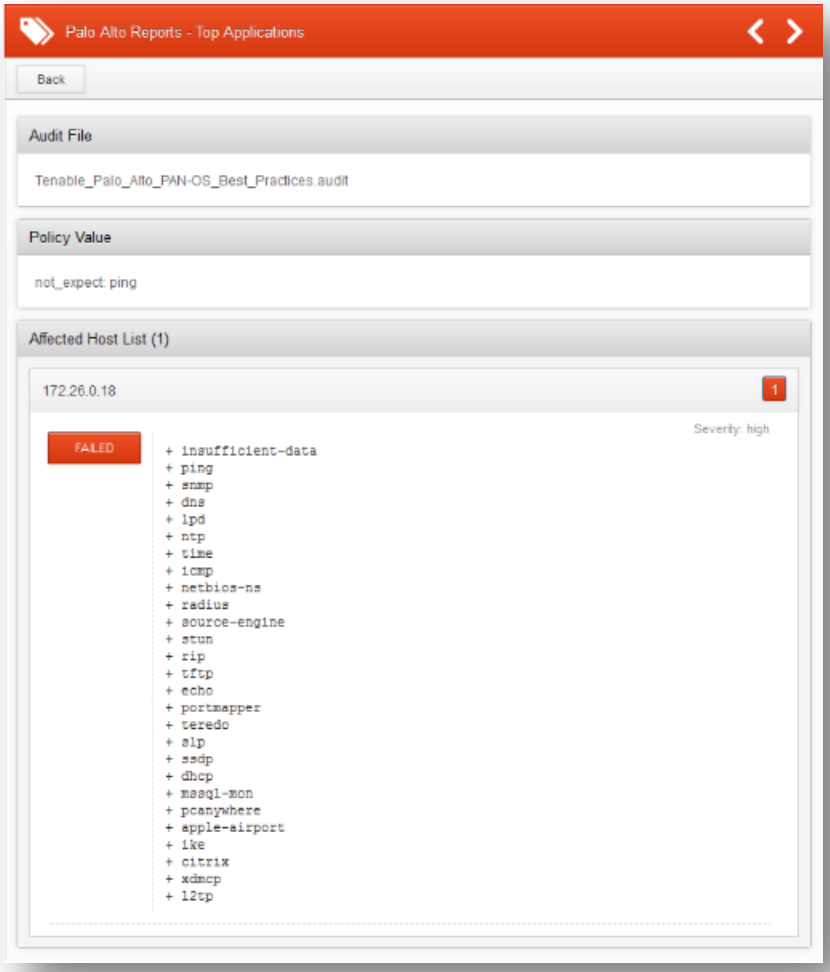
Este informe puede modificarse para usar una palabra clave **not_expect**:

```
<custom_item>
 type: AUDIT_REPORTS
 description: "Palo Alto Reports - Top Applications"
 request: "&reporttype=predefined&reportname=top-applications"
 xsl_stmt: "<xsl:template match=\"result\">"
 xsl_stmt: "<xsl:for-each select=\"entry\">"
 xsl_stmt: "+ <xsl:value-of select=\"name\"/>"
 xsl_stmt: "</xsl:for-each>"
 not_expect: "ping"
 check_option: CAN_BE_NULL
</custom_item>
```

El primer ejemplo dará un informe como este:


Palo Alto Reports - Top Applications
Back
Audit File
Tenable_Palo_Alto_Device_Best_Practices.audit
Affected Host List (1)
172.26.0.18
1
Severity: info
INFO
+ insufficient-data
+ ping
+ snmp
+ dns
+ lpd
+ ntp
+ time
+ icmp
+ netbios-ns
+ radius
+ source-engine
+ stun
+ rip
+ tftp
+ echo
+ portmapper
+ teredo
+ slp
+ ssdp
+ dhcp
+ mssql-mon
+ pcanywhere
+ apple-airport
+ ike
+ citrix
+ xdmcp
+ l2tp

El segundo ejemplo dará un informe que desaprueba:



Palabras clave

Se admiten las siguientes palabras clave en las auditorías de Palo Alto:

Palabra clave	Descripción
type	Debe estar siempre definido como AUDIT_XML o AUDIT_REPORTS.
description	Es la información que se usará como título para las vulnerabilidades de compatibilidad exclusivas en SecurityCenter. También será el primer conjunto de datos que informará Nessus.
info	Esta palabra clave les permite a los usuarios agregar una descripción más detallada a la comprobación que se lleva a cabo. Se permiten diversos campos **info** sin un límite predeterminado. El contenido de **info** debe estar entre comillas dobles.
api_request_type	Esta palabra describe el tipo de solicitud. La API de Palo Alto admite seis tipos de solicitudes: keygen, op, commit, reports, export y config. A los fines de este plugin,

	solo se expone el tipo de solicitud **op**.
request	Esta palabra clave especifica la solicitud para ejecutar en el firewall. El resultado de cada solicitud se almacena en caché, para que las solicitudes posteriores no resulten en otra solicitud. Además, para la comprobación AUDIT_REPORTS, la auditoría predeterminada de Tenable solo incluye 9 comprobaciones. Para incluir más informes, se les recomienda a los usuarios que creen nuevas comprobaciones y sustituyan la palabra de solicitud por la URL de la API REST después de `type=report`. Por ejemplo: `/api/?type=report&reporttype=predefined&reportname=hruser-top-url-categories`
regex	Esta palabra clave permite buscar elementos que coincidan con una expresión regex en particular. Si una comprobación tiene una palabra clave "**regex**" establecida, pero no hay ninguna palabra clave "**expect**" o "**not_expect**" establecida, la comprobación solo notifica todas las líneas que coincidan con la regex.

La compatibilidad de una comprobación puede determinarse comparando el resultado de la comprobación con las etiquetas "**expect**" o "**not_expect**". No puede haber más de una etiqueta de prueba de compatibilidad (es decir, solo puede estar "**expect**" **o** "**not_expect**", pero no "**expect**" **y** "**not_expect**").

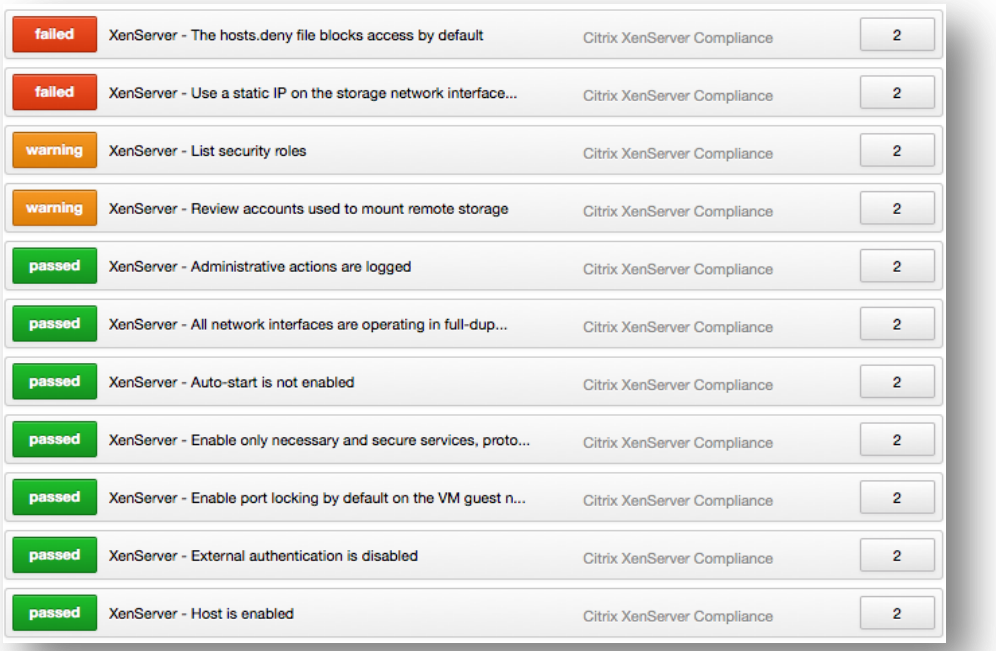
Palabra clave	Descripción
expect	Esta palabra clave permite la auditoría del elemento de configuración que coincide con la palabra clave "**regex**"; o, si la palabra clave "**regex**" no se utiliza, busca la cadena "**expect**" en toda la configuración. La comprobación se aprueba siempre y cuando la línea de configuración encontrada por la "**regex**" coincida con la cadena "**expect**"; o, en el caso de que "**regex**" no esté establecida, se aprueba si la cadena "**expect**" se encuentra en la configuración.
not_expect	Esta palabra clave permite la búsqueda de los elementos de configuración que **no** deben estar en la configuración. Actúa de manera opuesta a "**expect**". La comprobación se aprueba si la línea de configuración encontrada por la "**regex**" no coincide con la cadena "**not_expect**"; o, en el caso de que la palabra clave "**regex**" no esté establecida, se aprueba siempre y cuando la cadena "**not_expect**" no se encuentre en la configuración.

Referencia para archivos de compatibilidad de auditoría de Citrix XenServer

Las comprobaciones de compatibilidad de Citrix XenServer se basan mucho en la sección Unix Configuration Audit Compliance File Reference (Referencia para archivos de compatibilidad de auditoría de configuración de Unix), con una excepción. Hay una auditoría adicional disponible llamada AUDIT_XE para hacer auditorías de revisiones. Para auditorías de XenServer están disponibles los siguientes tipos de comprobaciones:

- FILE_CHECK_NOT
- PROCESS_CHECK
- FILE_CONTENT_CHECK
- FILE_CONTENT_CHECK_NOT
- CMD_EXEC
- GRAMMAR_CHECK

- RPM_CHECK
- CHKCONFIG
- XINETD_SVC
- AUDIT_XE



Tipo de comprobación: AUDIT_XE

El siguiente es un ejemplo de una comprobación de XenServer AUDIT_XE:

```
<custom_item>
 type: AUDIT_XE
 description: "List halted VMs"
 info: "Current guest VM status."
 reference: "PCI|2.2.3,SANS-CSC|1"
 cmd: "/usr/bin/xe vm-list power-state=halted params=uuid,name-label,power-state"
 # You can ignore VMs expected to be halted by entering their UUID here
 # Example ignore
 # ignore: "669e1681-2968-7435-c88e-663501f7d8f3"
</custom_item>
```

Palabras clave

La siguiente tabla indica la forma en que se puede usar cada palabra clave en las comprobaciones de compatibilidad de Citrix XenServer:

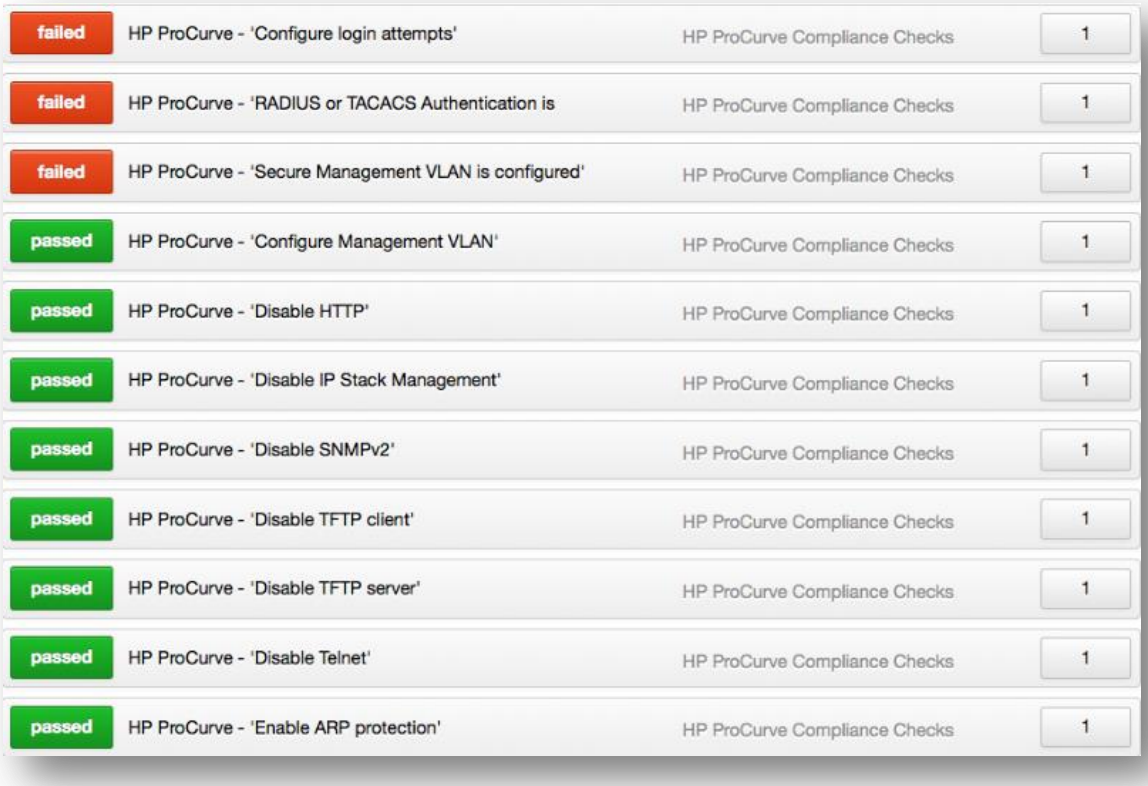
Palabra clave	Ejemplo de uso y configuración admitida
type	AUDIT_XE

description	Esta palabra clave proporciona una breve descripción de la comprobación que se lleva a cabo. Es obligatorio que el campo **description** sea exclusivo y que no haya comprobaciones diferentes que tengan el mismo campo de descripción. Esto es necesario porque el SecurityCenter usa este campo para generar de manera automática un número de identificación de plugin exclusivo en función del campo **description**. Ejemplo: `description: "List running VMs"`
info	Esta palabra clave les permite a los usuarios agregar una descripción más detallada a la comprobación que se lleva a cabo. Se permiten diversos campos info sin un límite predeterminado. El contenido de **info** debe estar entre comillas dobles. Ejemplo: `info: "The allocated virtual CPUs (VCPU) should be reviewed. Desired settings depend on workload and operating system type."`
see_also	Esta palabra clave les permite a los usuarios incluir enlaces que podrían proporcionar información útil sobre una comprobación. Ejemplo: `see_also: "http://support.citrix.com/article/CTX137828"`
reference	Esta palabra clave permite incluir referencias cruzadas para comprobaciones de auditoría. Ejemplo: `reference: "PCI|2.2.3,SANS-CSC|1"`
solution	Esta palabra clave suministra texto para incluir un texto de solución a fin de reparar una falla de compatibilidad.
severity	Esta palabra clave les permite a los usuarios definir la gravedad de la comprobación. La gravedad puede ser HIGH (ALTA), MEDIUM (MEDIA) o LOW (BAJA). Ejemplo: `severity: MEDIUM`
cmd	Esta palabra clave especifica el comando **xe** que se está ejecutando en el destino. Ejemplo: `cmd: "/usr/bin/xe subject-list params=all"`
regex	Esta palabra clave permite enumerar elementos que coincidan con una expresión regex en particular. Si una comprobación tiene establecida la palabra clave "**regex**", pero ninguna palabra clave "**expect**" o "**not_expect**", la comprobación solo notifica todos los elementos que coinciden con la regex. Ejemplo: `regex: "power-state.+"`
expect	Si se especifica la palabra clave **expect**, la comprobación será aprobada solo si todos los resultados coinciden con la palabra clave "**expect**". Si un resultado no coincide con la palabra clave **expect**, la comprobación fallará con todos los resultados que no coincidan con **expect**. Ejemplo:

	<custom_item> type: AUDIT_XE description: "List Running VMs - Any non running vms." cmd: "/usr/bin/xe vm-list params=uuid,name-label,is-a-template,power-state,allowed-operations" regex: "power-state .+" expect: "running" </custom_item>
not_expect	Si se define la palabra clave **not_expect**, la comprobación se aprueba siempre y cuando ninguno de los resultados coincida con la regex **not_expect**. Ejemplo: <custom_item> type: AUDIT_XE description: "List Running VMs" cmd: "/usr/bin/xe vm-list params=uuid,name-label,is-a-template,power-state,allowed-operations" regex: "power-state .+" not_expect: "halted" </custom_item>
ignore	Esta palabra clave permite ignorar/saltear determinados elementos del resultado. Ejemplo: <custom_item> type: AUDIT_XE description: "List halted VMs" info: "Current guest VM status." cmd: "/usr/bin/xe vm-list power-state=halted params=uuid,name-label,power-state" # You can ignore VMs expected to be halted by entering their UUID here # Example ignore ignore: "669e1681-2968-7435-c88e-663501f7d8f3" </custom_item>

Referencia para archivos de auditoría de compatibilidad de HP ProCurve

Las auditorías de HP ProCurve son en muchos aspectos una extensión del plugin de compatibilidad de Cisco. El archivo de auditoría de Tenable de HP ProCurve está basado en un documento HP sobre cómo fortalecer los interruptores ProCurve. La auditoría comprende comprobaciones para deshabilitar servicios inseguros y habilitar el control de acceso (por ejemplo, TACACS, RADIUS). Se requieren credenciales SSH para root (raíz) o un administrador con privilegios totales.



Tipos de comprobación

Existen tres tipos de comprobaciones de compatibilidad de HP ProCurve. La siguiente es la sintaxis general de una auditoría:

```
<custom_item>
 type: CONFIG_CHECK
 description: "Verify login authentication"
 info: "Verifies login authentication configuration"
 reference: "PCI|2.2.3,SANS-CSC|1"
 context: "line .*"
 item: "login authentication"
</custom_item>
```

Palabras clave

La siguiente tabla indica la forma en que se puede usar cada palabra clave en las comprobaciones de compatibilidad de HP ProCurve:

Palabra clave	Ejemplo de uso y configuración admitida
type	CONFIG_CHECK CONFIG_CHECK_NOT RANDOMNESS_CHECK

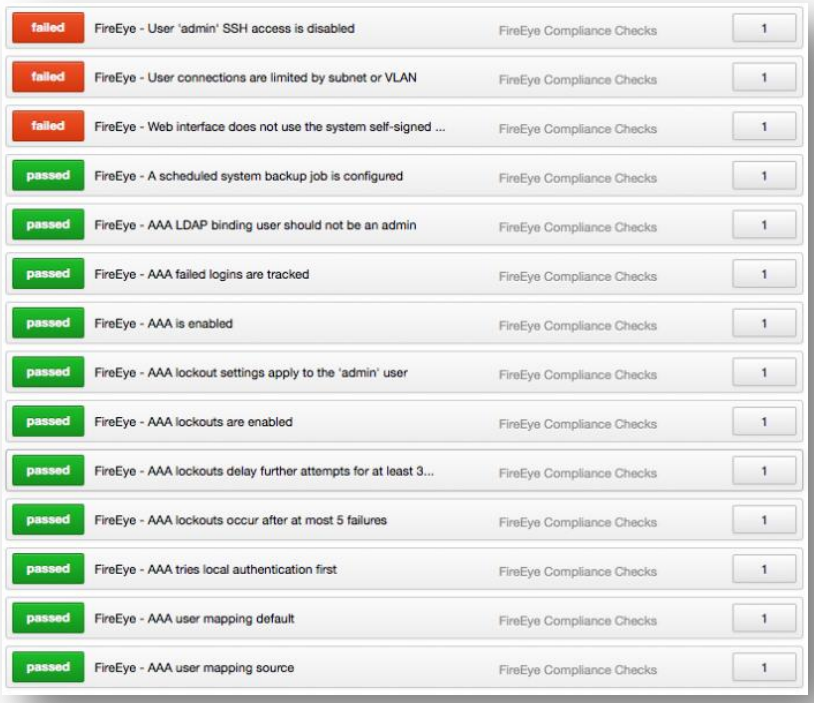
description	Esta palabra clave proporciona una breve descripción de la comprobación que se lleva a cabo. Es obligatorio que el campo **description** sea exclusivo y que no haya comprobaciones diferentes que tengan el mismo campo de descripción. Esto es necesario porque el SecurityCenter usa este campo para generar de manera automática un número de identificación de plugin exclusivo en función del campo **description**. Ejemplo: `description: " Verify login authentication"`
info	Esta palabra clave les permite a los usuarios agregar una descripción más detallada a la comprobación que se lleva a cabo. Se permiten diversos campos **info** sin un límite predeterminado. El contenido de **info** debe estar entre comillas dobles. Ejemplo: `info: "Verifies login authentication configuration."`
see_also	Esta palabra clave les permite a los usuarios incluir enlaces que podrían proporcionar información útil sobre una comprobación. Ejemplo: `see_also: "http://www.hp.com/rnd/support/faqs/1800.htm"`
reference	Esta palabra clave permite incluir referencias cruzadas para comprobaciones de auditoría. Ejemplo: `reference: "PCI|2.2.3,SANS-CSC|1"`
solution	Esta palabra clave suministra texto para incluir un texto de solución a fin de reparar una falla de compatibilidad. Ejemplo: `solution: "Modify the configuration to add missing line"`
severity	Esta palabra clave les permite a los usuarios definir la gravedad de la comprobación. La gravedad puede ser HIGH (ALTA), MEDIUM (MEDIA) o LOW (BAJA). Ejemplo: `severity: MEDIUM`
regex	Esta palabra clave permite enumerar elementos que coincidan con una expresión regex en particular. Si una comprobación tiene la palabra clave "**regex**", pero ninguna palabra clave "**expect**" o "**not_expect**", la comprobación solo notifica todos los elementos que coinciden con la regex. Ejemplo: `regex: "power-state.+"`
item	Esta palabra clave permite buscar en las líneas encontradas por **regex**. Si no se suministró una regex, se revisarán todas las líneas. Ejemplo: `regex: "power"`
context	Esta palabra clave permite buscar en un contexto específico. Un contexto se define por una línea justificada a la izquierda seguida de líneas prefijadas por espacio en blanco. Ejemplo:

	`context: "line .*"`
	
	El siguiente es un elemento de configuración de muestra que podría ser auditado utilizando contexto:
	
	`vlan 1`
	`  name "DEFAULT_VLAN"`
	`  untagged 2-24`
	`  ip address dhcp-bootp`
	`  no untagged 1`
	`  exit`
	
	`<item>`
	`  type: CONFIG_CHECK`
	`  description: "HP ProCurve - 'dhcp-bootp'"`
	`  context: "vlan 1"`
	`  item: "ip address dhcp-bootp"`
	`</item>`
	
	La comprobación anterior garantizará que "`ip address dhcp-bootp`" se defina para el contexto "`vlan 1`".

`min_occurrences`	Esta palabra clave permite definir una cantidad mínima de apariciones de la comprobación. Ejemplo: `min_occurrences: 3`
`max_occurrences`	Como **`min_occurrences`**, pero define un valor máximo en lugar de mínimo.
`required`	Esta palabra clave permite especificar si se requiere una coincidencia de comprobación o no. El valor del campo obligatorio puede ser YES, NO, ENABLED o DISABLED. Ejemplo: `required: YES`

Referencia para archivos de compatibilidad de auditoría de FireEye

La auditoría de FireEye se basa en la documentación del producto de FireEye y pautas de criterios comunes. La auditoría comprende comprobaciones para auditorías, identificación y autenticación, administración de aparatos, interfaz de administración de plataformas inteligentes (IPMI), servicios habilitados, cifrado y configuración del sistema de detección de malware. Se requieren credenciales SSH válidas para root (raíz) o un administrador con privilegios totales.



Tipos de comprobación

Existen tres tipos de comprobaciones de compatibilidad de FireEye. La siguiente es la sintaxis general de una auditoría:

```
<item>
 type: CONFIG_CHECK
 description: "Specific user privs"
 info: "Expect to fail on running config since not all username lines match"
 regex: "username .+"
 expect: " username egossell capability admin"
</item>
```

Palabras clave

La siguiente tabla indica la forma en que se puede usar cada palabra clave en las comprobaciones de compatibilidad de FireEye:

Palabra clave	Ejemplo de uso y configuración admitida
type	CONFIG_CHECK CONFIG_CHECK_NOT RANDOMNESS_CHECK
description	Esta palabra clave proporciona una breve descripción de la comprobación que se lleva a cabo. Es obligatorio que el campo **description** sea exclusivo y que no haya comprobaciones diferentes que tengan el mismo campo de descripción. Esto es necesario porque el SecurityCenter usa este campo para generar de manera automática un número de identificación de plugin exclusivo en función del campo

	description. Ejemplo: `description: " Verify login authentication"`
info	Esta palabra clave les permite a los usuarios agregar una descripción más detallada a la comprobación que se lleva a cabo. Se permiten diversos campos **info** sin un límite predeterminado. El contenido de **info** debe estar entre comillas dobles. Ejemplo: `info: "Verifies login authentication configuration."`
see_also	Esta palabra clave les permite a los usuarios incluir enlaces que podrían proporcionar información útil sobre una comprobación. Ejemplo: `see_also: "http://www.fireeye.com/support/"`
reference	Esta palabra clave permite incluir referencias cruzadas para comprobaciones de auditoría. Ejemplo: `reference: "PCI|2.2.3,SANS-CSC|1"`
solution	Esta palabra clave suministra texto para incluir un texto de solución a fin de reparar una falla de compatibilidad. Ejemplo: `solution: "Modify the configuration to add missing line"`
severity	Esta palabra clave les permite a los usuarios definir la gravedad de la comprobación. La gravedad puede ser HIGH (ALTA), MEDIUM (MEDIA) o LOW (BAJA). Ejemplo: `severity: MEDIUM`
regex	Esta palabra clave permite enumerar elementos que coincidan con una expresión regex en particular. Si una comprobación tiene la palabra clave "**regex**", pero ninguna palabra clave "**expect**" o "**not_expect**", la comprobación solo notifica todos los elementos que coinciden con la regex. Ejemplo: `regex: "power-state.+"`
expect	Esta palabra clave permite buscar en las líneas encontradas por **regex**. Todas las líneas encontradas por la regex deben coincidir con la opción **expect** para que la comprobación sea aprobada. Si no se suministró una regex, se revisarán todas las líneas pero solo debe encontrarse una. Ejemplo: `regex: "power"`
not_expect	Similar a **expect**, pero si se encuentran coincidencias, la comprobación es desaprobada. Si se omiten **expect** y **not_expect**, todas las líneas correspondientes se notifican como mensaje de info.
min_occurrences	Esta palabra clave permite definir una cantidad mínima de apariciones de la comprobación.

	Ejemplo: `min_occurrences: 3`
max_occurrences	Como **min_occurrences**, pero define un valor máximo en lugar de mínimo.
required	Esta palabra clave especifica si se requiere una coincidencia de comprobación o no. El valor del campo obligatorio puede ser YES, NO, ENABLED o DISABLED. Ejemplo: `required: YES`

cmd	Esto les permite a los usuarios ejecutar un comando “show” (mostrar).
	
	Ejemplo:
	`cmd: "show version"`
	
	Solo se permiten comandos “**show**” (mostrar).
	
	`<item>`
	` type: CONFIG_CHECK`
	` cmd: "show version"`
	` description: "Show Product version"`
	` regex: "Proaduct model:"`
	` expect: "1234"`
	`</item>`

Referencia para archivos de compatibilidad de auditoría de configuración de bases de datos

Esta sección describe el formato y las funciones de las comprobaciones de compatibilidad de bases de datos y la fundamentación que subyace en cada opción.

Uso de comillas:

Las comillas simples y dobles son indistintas al encerrar campos de auditoría, a excepción de los siguientes dos casos:

1. En comprobaciones de compatibilidad de Windows en las que se deban interpretar de forma literal los campos especiales como CRLF, etc., se deben usar comillas simples. Se deben escapar los campos incrustados que se interpretarán como cadenas.

Por ejemplo:
`expect: 'First line\r\nSecond line\r\nJohn\'s Line'`

2. Se requieren comillas dobles al utilizar “include_paths” y “exclude_paths” de WindowsFiles.

Si en cualquier tipo de campo (descripción, value_data, regex, etc.) se usan cadenas que contengan comillas simples o dobles, existen dos formas de tratarlas:

a. Use el tipo de comilla opuesto para las comillas de cierre exteriores.

Por ejemplo:
```
expect: "This is John's Line"
expect: 'We are looking for a double-quote-".*'
```

b. Escape las comillas incrustadas, si las hay, con una barra inversa (solo comillas dobles).

Por ejemplo:
```
expect: "\"Text to be searched\""
```

Tipo de comprobación

Todas las comprobaciones de compatibilidad de bases de datos deben estar entre corchetes con la encapsulación **`check_type`** y la designación “Database”. Esto es necesario para diferenciar los archivos **`.audit`** diseñados específicamente para bases de datos, de otros tipos de auditorías de compatibilidad. El campo **`check_type`** requiere dos parámetros adicionales:

- **`db_type`**
- **`version`**

Los tipos de bases de datos disponibles para auditorías son:

- SQLServer
- Oracle
- MySQL
- PostgreSQL
- DB2
- Informix

La **`version`** (versión) está actualmente establecida siempre en “1”.

Ejemplo:

```
<check_type: "Database" db_type:"SQLServer" version:"1">
```

Palabras clave

La siguiente tabla indica la forma en que se puede usar cada palabra clave en las comprobaciones de compatibilidad de bases de datos:

Palabra clave	Ejemplo de uso y configuración admitida
`type`	`SQL_POLICY`
`description`	Esta palabra clave proporciona la capacidad de añadir una breve descripción de la comprobación que se lleva a cabo. Se recomienda enfáticamente que el campo **`description`** sea exclusivo y que no haya comprobaciones diferentes que tengan el mismo campo de descripción. El SecurityCenter de Tenable usa este campo para generar de manera automática un número de identificación de plugin exclusivo en función del campo **`description`**. Ejemplo: `description: "DBMS Password Complexity"`

info	Esta palabra clave se usa para añadir una descripción más detallada a la comprobación que se está llevando a cabo, tal como una reglamentación, una dirección URL, una directiva corporativa, o bien otro motivo por el que la opción sea necesaria. Se pueden añadir varios campos **info** en líneas independientes para que el texto adquiera formato de párrafo. No existe un límite preestablecido en cuanto a la cantidad de campos **info** que se pueden usar. Ejemplo: `info: "Checking that \"password complexity\" requirements are enforced for systems using SQL Server authentication."`

sql_request	Esta palabra clave se usa para determinar la solicitud **SQL** real que se enviará a la base de datos. Se pueden solicitar y devolver matrices de datos desde una solicitud **SQL** mediante valores de solicitud/devolución delimitados por comas.
	
	Ejemplo:
	`sql_request: "select name from sys.sql_logins where type = 'S' and is_policy_checked <> '1'"`
	
	Ejemplo:
	`sql_request: "select name, value_in_use from sys.configurations where name = 'clr enabled'"`

sql_types	Esta palabra clave tiene dos opciones disponibles: **POLICY_VARCHAR** y **POLICY_INTEGER**. Use **POLICY_INTEGER** para valores numéricos que oscilen entre 0 y 2 147 483 647, y **POLICY_VARCHAR** para cualquier otro tipo de valor de devolución.
	
	Ejemplo:
	`sql_types: POLICY_VARCHAR`
	
	Ejemplo:
	`sql_types: POLICY_VARCHAR,POLICY_INTEGER`
	
	Para obtener varios elementos de devolución, configure **sql_types** en una lista separada por comas para aceptar los tipos de datos de cada resultado de devolución de **SQL**. El ejemplo anterior indica que el primer valor de devolución de la consulta **SQL** es **varchar**, y el segundo es un número entero.

sql_expect	Esta palabra clave se usa para determinar el valor de devolución esperado de la solicitud **SQL**. Es posible que se requiera un valor exacto que incluya **NULL** o "**0**". Además, es posible que sean necesarias expresiones regulares para **POLICY_VARCHAR sql_types**.
	
	Ejemplo:
	`sql_expect: regex:"^.+Failure" || regex:"^.+ALL"`
	
	Ejemplo:
	`sql_expect: NULL`
	
	Ejemplo:
	`sql_expect: 0 || "0"`
	
	En el caso de los valores de devolución enteros, las comillas dobles son opcionales.
	
	Ejemplo:

	`sql_expect: "clr enabled",0` Una solicitud **SQL** puede devolver una matriz de datos y la puede incluir en un formato separado por comas en el campo **sql_expect**.

Ejemplos de líneas de comandos

Esta sección proporciona algunos ejemplos de auditorías comunes usadas en comprobaciones de compatibilidad de bases de datos. El binario de la línea de comandos **nasl** se usa como método rápido para probar auditorías sobre la marcha. Cada archivo **.audit** que se demuestra a continuación se puede incluir fácilmente en las directivas de análisis de Nessus 4 o SecurityCenter. Sin embargo, en el caso de las auditorías rápidas de un sistema, las pruebas de líneas de comandos son más eficaces. El comando que se ejecutará cada vez desde el directorio **/opt/nessus/bin** será el siguiente:

```
# ./nasl -t <IP> /opt/nessus/lib/nessus/plugins/database_compliance_check.nbin
```

<IP> es la dirección IP del sistema que se auditará.

De acuerdo con el tipo de base de datos en la que se lleva a cabo la auditoría, es posible que se le solicite que introduzca otros parámetros además del archivo de auditoría que se usará. Por ejemplo, las auditorías de Oracle solicitarán el SID de la base de datos y el tipo de inicio de sesión de Oracle:

```
Which file contains your security policy : oracle.audit
login : admin
Password :
Database type: ORACLE(0), SQL Server(1), MySQL(2), DB2(3), Informix/DRDA(4),
       PostgreSQL(5)
type : 0
sid: oracle
Oracle login type: NORMAL (0), SYSOPER (1), SYSDBA (2)
type: 2
```

Consulte al administrador de su base de datos para conocer los parámetros correctos de inicio de sesión en la base de datos.

Ejemplo 1: Búsqueda de inicios de sesión sin fecha de vencimiento

A continuación se ilustra un archivo **.audit** simple que busca cualquier inicio de sesión de SQL Server sin fecha de vencimiento. Si se encuentra alguno, la auditoría mostrará un mensaje de error junto con los nombres de los inicios de sesión incorrectos.

```
<check_type: "Database" db_type:"SQLServer" version:"1">
<group_policy: "Login expiration check">
<custom_item>
 type: SQL_POLICY
 description: "Login expiration check"
 info: "Database logins with no expiration date pose a security threat. "
 sql_request: "select name from sys.sql_logins where type = 'S' and
       is_expiration_checked = 0"
 sql_types: POLICY_VARCHAR
 sql_expect: NULL
</custom_item>
</group_policy>
</check_type>
```

Al ejecutar este comando, se esperan de un sistema compatible los siguientes resultados:

```
"Login expiration check": [PASSED]
```

Los requisitos de compatibilidad normalmente exigen que los inicios de sesión de base de datos tengan una fecha de vencimiento.

Una auditoría con errores devolvería los siguientes resultados:

```
"Login expiration check": [FAILED]

Database logins with no expiration date pose a security threat.

Remote value:

"distributor_admin"

Policy value:

NULL
```

Este resultado indica que la cuenta “distributor_admin” no tiene una fecha de vencimiento configurada, y es necesario comprobarla respecto de la directiva de seguridad del sistema.

Ejemplo 2: Comprobación del estado habilitado de procedimientos almacenados no autorizados

Esta auditoría comprueba si el procedimiento almacenado “SQL Mail XPs” se encuentra habilitado. Los procedimientos almacenados externos pueden constituir una amenaza a la seguridad para algunos sistemas, y a menudo se exige que sean deshabilitados.

```
<check_type: "Database" db_type:"SQLServer" version:"1">
<group_policy: "Unauthorized stored procedure check">
<custom_item>
 type: SQL_POLICY
 description: "SQL Mail XPs external stored procedure check"
 info: "Checking whether SQL Mail XPs is disabled."
 sql_request: "select value_in_use from sys.configurations where name = 'SQL Mail
       XPs'"
 sql_types: POLICY_INTEGER
 sql_expect: 0
</custom item>
</group_policy>
</check_type>
```

La comprobación anterior devolverá un resultado “passed” (Aprobó) si el procedimiento almacenado “SQL Mail XPs” está deshabilitado (value_in_use = 0). De lo contrario, devolverá un resultado “failed” (Incorrecto).

Ejemplo 3: Comprobación del estado de la base de datos con resultados sql_types combinados

En algunos casos, las consultas de compatibilidad de bases de datos requieren varias solicitudes de datos con resultados de varios tipos de datos. El ejemplo de auditoría a continuación combina los tipos de datos y demuestra la forma en que se pueden analizar sintácticamente los resultados.

```
<check_type: "Database" db_type:"SQLServer" version:"1">
```

```
<group_policy: "Mixed result type check">
<custom_item>
 type: SQL_POLICY
 description: "Mixed result type check"
 info: "Checking values for the master database."
 sql_request: " select database_id,user_access_desc,is_read_only from sys.databases
       where is_trustworthy_on=0 and name = 'master'"
 sql_types: POLICY_INTEGER,POLICY_VARCHAR,POLICY_INTEGER
 sql_expect: 1,MULTI_USER,0
</custom_item>
</group_policy>
</check_type>
```

Tenga en cuenta que todos los valores **sql_request**, **sql_types** y **sql_expect** contienen valores separados por comas.

Condiciones

Es posible definir la lógica **if/then/else** en la directiva de bases de datos. Esto le permite al usuario final devolver un mensaje de advertencia en lugar de una aprobación o error en caso de que la auditoría sea aprobada.

La sintaxis para establecer condiciones es la siguiente:

```
<if>
 <condition type: "or">
 <Insert your audit here>
 </condition>
 <then>
  <Insert your audit here>
 </then>
 <else>
  <Insert your audit here>
 </else>
</if>
```

Ejemplo:

```
<if>
 <condition type: "or">
  <custom_item>
   type: SQL_POLICY
   description: "clr enabled option"
   info: "Is CLR enabled?"
   sql_request: "select value_in_use from sys.configurations where name = 'clr
       enabled'"
   sql_types: POLICY_INTEGER
   sql_expect: "0"
  </custom_item>
 </condition>

 <then>
  <custom_item>
   type: SQL_POLICY
   description: "clr enabled option"
   info: "CLR is disabled?"
```

```
    sql_request: "select value_in_use from sys.configurations where name = 'clr
        enabled'"
    sql_types: POLICY_INTEGER
    sql_expect: "0"
   </custom_item>
  </then>

  <else>
   <report type: "WARNING">
    description: "clr enabled option"
    info: "CLR(Command Language Runtime objects) is enabled"
    info: "Check system policy to confirm CLR requirements."
   </report>
  </else>
 </if>
```

Ya sea que la condición sea errónea o se apruebe, eso nunca aparecerá en el informe, ya que se trata de una comprobación “silent” (silenciosa).

Las condiciones pueden ser del tipo “**and**” u “**or**”.

Referencia para archivos de compatibilidad de auditoría de configuración de Unix

Esta sección describe las funciones incorporadas de las comprobaciones de compatibilidad de Unix y la fundamentación que subyace en cada opción.

Uso de comillas:

Las comillas simples y dobles son indistintas al encerrar campos de auditoría, a excepción de los siguientes dos casos:

1. En comprobaciones de compatibilidad de Windows en las que se deban interpretar de forma literal los campos especiales como CRLF, etc., se deben usar comillas simples. Se deben escapar los campos incrustados que se interpretarán como cadenas.

Por ejemplo:
```
expect: 'First line\r\nSecond line\r\nJohn\'s Line'
```

2. Se requieren comillas dobles al utilizar “include_paths” y “exclude_paths” de WindowsFiles.

Si en cualquier tipo de campo (descripción, value_data, regex, etc.) se usan cadenas que contengan comillas simples o dobles, existen dos formas de tratarlas:

a. Use el tipo de comilla opuesto para las comillas de cierre exteriores.

Por ejemplo:
```
expect: "This is John's Line"
expect: 'We are looking for a double-quote-".*'
```

b. Escape las comillas incrustadas, si las hay, con una barra inversa (solo comillas dobles).

Por ejemplo:
```
expect: "\"Text to be searched\""
```

Tipo de comprobación

Todas las comprobaciones de compatibilidad de Unix deben estar entre corchetes con la encapsulación "**check_type**" y la designación "Unix". El Apéndice A contiene un ejemplo de comprobación de compatibilidad de Unix que comienza con la opción **check_type** para "Unix" y finaliza con la etiqueta "**</check_type>**".

Esto es necesario para diferenciar los archivos **.audit** diseñados para auditorías de compatibilidad de Windows (u otras plataformas).

El archivo se lee de SSH a un búfer de memoria en el servidor Nessus, y luego el búfer se procesa para comprobar la compatibilidad o incompatibilidad.

Palabras clave

La siguiente tabla indica la forma en que se puede usar cada palabra clave en las comprobaciones de compatibilidad de Unix.

Palabra clave	Ejemplo de uso y configuración admitida
attr	Esta palabra clave se usa junto con FILE_CHECK y FILE_CHECK_NOT para auditar los atributos de archivo relacionados con un archivo. Consulte la man page (página de manual) chattr(1) para obtener detalles sobre cómo configurar los atributos de un archivo.
comment	Este campo se usa para añadir información adicional que no corresponda al campo de descripción. Ejemplo: `comment: (CWD - Current working directory)`
description	Esta palabra clave proporciona una breve descripción de la comprobación que se lleva a cabo. Es obligatorio que el campo **description** sea exclusivo y que no haya comprobaciones diferentes que tengan el mismo campo de descripción. El SecurityCenter de Tenable usa este campo para generar de manera automática un número de identificación de plugin exclusivo en función del campo **description**. Ejemplo: `description: "Permission and ownership check for /etc/at.allow"`
dont_echo_cmd	Esta palabra clave se usa con las auditorías de comprobación de compatibilidad de Unix "CMD_EXEC", e indica a la auditoría que omita incluir en los resultados el comando efectivo ejecutado por la comprobación. Solo aparecerán los resultados del comando. Ejemplo: `dont_echo_cmd: YES`

except	Esta palabra clave se usa para excluir de la comprobación a determinados usuarios, servicios y archivos.
	
	Ejemplo:
	`except: "guest"`
	
	Se pueden incluir varias cuentas de usuario juntas entre barras verticales.
	
	Ejemplo:
	`except: "guest" | "guest1" | "guest2"`

expect	Esta palabra clave se usa junto con **regex**. Brinda la capacidad de buscar valores específicos dentro de los archivos.
	
	Ejemplo:
	`<custom_item>`
	` system: "Linux"`
	` type: FILE_CONTENT_CHECK`
	` description: "This check reports a problem when the log level setting in the sendmail.cf file is less than the value set in your security policy."`
	` file: "sendmail.cf"`
	` regex: ".*LogLevel=.*"`
	` expect: ".*LogLevel=9"`
	`</custom_item>`

file	Esta palabra clave se usa para describir la ruta absoluta o relativa de un archivo del que se comprobará la configuración de propiedad y los permisos.
	
	Ejemplos:
	`file: "/etc/inet/inetd.conf"`
	`file: "~/inetd.conf"`
	
	El valor **file** también puede ser un comodín.
	
	Ejemplo:
	`file: "/var/log/*"`
	
	Esta característica resulta particularmente útil cuando todos los archivos de un directorio dado deben auditarse en busca de permisos o contenidos mediante FILE_CHECK, FILE_CONTENT_CHECK, FILE_CHECK_NOT o FILE_CONTENT_CHECK_NOT.

file_type	Esta palabra clave describe el tipo de archivo que se busca. A continuación se presenta una lista de tipos de archivos admitidos.
	
	• b - especial de bloques (almacenado en búfer)
	• c - especial por caracteres (no almacenado en búfer)
	• d - directorio
	• p - canal con nombre (FIFO)
	• f - archivo normal
	
	Ejemplo:
	`file_type: "f"`
	
	Uno o más tipos de archivos se pueden incluir juntos en la misma cadena, separados por barras verticales.
	
	Ejemplo:
	`file_type: "c|b"`

group	Esta palabra clave se usa para especificar el grupo de un archivo. Siempre se usa junto con la palabra clave **file**. La palabra clave **group** puede tener un valor “none”, que ayuda en la búsqueda de archivos sin propietario. Ejemplo: `group: "root"`

	El grupo también se puede especificar mediante una condición lógica “OR”, con la siguiente sintaxis: `group: "root" || "bin" || "sys"`

ignore	Esta palabra clave indica a la comprobación que omita archivos designados en la búsqueda. Esta palabra clave se encuentra disponible para los tipos de comprobación FILE_CHECK, FILE_CHECK_NOT, FILE_CONTENT_CHECK y FILE_CONTENT_CHECK_NOT.
	
	Ejemplos:
	`# ignore single file`
	`ignore: "/root/test/2"`
	
	`# ignore certain files from a directory`
	`ignore: "/root/test/foo*"`
	
	`# ignore all files in a directory`
	`ignore: "/root/test/*"`

info	Esta palabra clave se usa para añadir una descripción más detallada a la comprobación que se está llevando a cabo, tal como una reglamentación, una dirección URL, una directiva corporativa, o bien un motivo por el que la opción sea necesaria. Se pueden añadir varios campos **info** en líneas independientes para que el texto adquiera formato de párrafo. No existe un límite preestablecido en cuanto a la cantidad de campos **info** que se pueden usar. Ejemplo: `info: "ref. CIS_AIX_Benchmark_v1.0.1.pdf ch 1, pg 28-29."`
levels	Esta palabra clave se usa junto con CHKCONFIG, y se emplea para especificar los niveles de ejecución para los cuales debe ejecutarse el servicio. Todos los niveles de ejecución deben describirse en una única cadena. Por ejemplo, si el servicio “sendmail” debe ejecutarse en el nivel de ejecución 1, 2 y 3, el valor **levels** correspondiente en la comprobación CHKCONFIG sería: `levels: "123"`
mask	Esta palabra clave representa el opuesto de **mode**, y mediante ella se pueden especificar permisos que **no** deben estar disponibles para un usuario, un grupo u otro miembro en particular. A diferencia de **mode**, que comprueba un valor de permiso *exacto*, las auditorías **mask** son más amplias y comprobarán si un archivo o un directorio se encuentra en un nivel que es igual a lo especificado por **mask**, o bien más seguro. (En los casos en que **mode** pueda rechazar un archivo con un permiso de 640 por no coincidir con una auditoría en la que se espera un valor de 644, **mask** determinará que 640 es “más seguro” y aprobará la auditoría como satisfactoria). Ejemplo: `mask: 022` Esto especificaría que cualquier permiso sea aceptable en lo que respecta al propietario y que no haya ningún permiso de escritura para un grupo u otro miembro. Un valor **mask** de “7” significaría la ausencia de permisos para ese propietario, grupo u otro miembro en particular.

md5	Esta palabra clave se usa en FILE_CHECK y FILE_CHECK_NOT para garantizar que el MD5 de un archivo esté efectivamente establecido en cualquier valor que establezca la directiva.
	
	Ejemplo:
	`<custom_item>`
	`type: FILE_CHECK`
	`description: "/etc/passwd has the proper md5 set"`
	`required: YES`
	`file: "/etc/passwd"`
	`md5: "ce35dc081fd848763cab2cfd442f8c22"`
	`</custom_item>`

mode	Esta palabra clave describe el conjunto de permisos correspondientes a un archivo o carpeta en estudio. La palabra clave **mode** se puede representar mediante formato de cadena u octal.
	
	Ejemplos:
	`mode: "-rw-r--r--"`
	`mode: "644"`
	`mode: "7644"`

name	Esta palabra clave se usa para identificar el nombre del proceso en PROCESS_CHECK. Ejemplo: `name: "syslogd"`
operator	Esta palabra clave se usa junto con RPM_CHECK y PKG_CHECK para especificar la condición de aprobación o no de una comprobación en función de la versión del paquete RPM instalado. Puede adquirir los siguientes valores: • **lt** less than (menor que) • **lte** less than or equal (menor o igual que) • **gte** greater than or equal (mayor o igual que) • **gt** greater than (mayor que) • **eq** equal (igual) Ejemplo: `operator: "lt"`

owner	Esta palabra clave se usa para especificar el propietario de un archivo. Siempre se usa junto con la palabra clave **file**. La palabra clave **owner** puede tener un valor “none” (ninguno), que ayuda en la búsqueda de archivos sin propietario.
	
	Ejemplo:
	`owner: "root"`
	
	La propiedad también se puede especificar mediante una condición lógica “OR” con la siguiente sintaxis:
	`owner: "root" || "bin" || "adm"`

reference	Esta palabra clave brinda una manera de incluir referencias cruzadas en **.audit**. El formato es “ref|ref-id1, ref|ref-id2”. Ejemplo: `reference: "CAT|CAT II,800-53|IA-5,8500.2|IAIA-1,8500.2|IAIA-`

	2,8500.2|IATS-1,8500.2|IATS-2"
regex	Esta palabra clave habilita la búsqueda de un archivo que coincida con una expresión regex en particular. Ejemplo: `regex: ".*LogLevel=9$"` Los siguientes metacaracteres requieren tratamiento especial: + \ * ( ) ^ Escape doblemente estos caracteres con dos barras invertidas "\\", o enciérrelos entre corchetes cuadrados "[]" si desea que se interpreten de forma literal. Otros caracteres como los siguientes necesitan solo una barra invertida para que se interpreten literalmente: . ? " ' Esto se relaciona con la forma en que el compilador trata estos caracteres.
required	Esta palabra clave se usa para especificar si el elemento auditado debe estar presente o no en el sistema remoto. Por ejemplo, si **required** se encuentra establecida en "NO" y el **type** de la comprobación es "FILE_CHECK", la comprobación será aprobada si el archivo existe y los permisos son los especificados en el archivo **.audit**, o bien si el archivo no existe. Por otra parte, si **required** se estableció en "YES" (SÍ), se producirá un error en la comprobación anterior.

rpm	Esta palabra clave se usa para especificar el RPM que se buscará al usarla junto con RPM_CHECK.
	
	Ejemplo:
	`<custom_item>`
	`type: RPM_CHECK`
	`description: "Make sure that the Linux kernel is BELOW version 2.6.0"`
	`rpm: "kernel-2.6.0-0"`
	`operator: "lt"`
	`required: YES`
	`</custom_item>`

search_locations	Esta palabra clave se puede usar para especificar ubicaciones que permiten búsquedas dentro de un sistema de archivos.
	
	Ejemplo:
	`search_locations: "/bin"`
	
	Se pueden incluir juntas varias ubicaciones de búsqueda entre barras verticales.
	
	Ejemplo:
	`search_locations: "/bin" | "/etc/init.d" | "/etc/rc0.d"`

see_also	Esta palabra clave permite incluir enlaces a una referencia. Ejemplo: `see_also: "https://benchmarks.cisecurity.org/tools2/linux/CIS_Redhat_Linux_5_Benchmark_v2.0.0.pdf"`
service	Esta palabra clave se usa junto con CHKCONFIG, XINETD_SVC y SVC_PROP, y se emplea para especificar el servicio que se está auditando.

	Ejemplo:
	`<custom_item>`
	`type: CHKCONFIG`
	`description: "2.1 Disable Standard Services - Check if cups is disabled"`
	`service: "cups"`
	`levels: "123456"`
	`status: OFF`
	`</custom_item>`

severity	En cualquier prueba, **<item>** o **<custom_item>**, se puede añadir un indicador "**severity**" y establecerlo en "LOW" (BAJA), "MEDIUM" (MEDIA) o "HIGH" (ALTA). De manera predeterminada, los resultados no compatibles aparecerán como "high" (alta). Ejemplo: `severity: MEDIUM`
solution	Esta palabra clave brinda una manera de incluir un texto de "Solution" (Solución), si está disponible. Ejemplo: `solution: "Remove this file, if its not required"`

status	Esta palabra clave se usa en PROCESS_CHECK, CHKCONFIG y XINETD_SVC para determinar si un servicio que se encuentra en ejecución en un host en particular debe estar en ejecución o deshabilitado. La palabra clave **status** puede adquirir 2 valores, "ON" (Activado) u "OFF" (Desactivado).
	
	Ejemplo:
	`status: ON`
	`status: OFF`

system	Esta palabra clave especifica el tipo de sistema en el que se llevará a cabo la comprobación.  La palabra clave "**system**" solo se aplica a las comprobaciones "**custom_item**", no a las comprobaciones incorporadas "**item**". Los valores disponibles son los que devuelve el comando "**uname**" en el sistema operativo de destino. Por ejemplo, en Solaris el valor es "SunOS", en Mac OS X es "Darwin", en FreeBSD es "FreeBSD", etc. Ejemplo: `system: "SunOS"`
timeout	Esta palabra clave se usa junto con CMD_EXEC y especifica el tiempo de espera, en segundos, durante el cual se permitirá la ejecución del comando especificado. Resulta útil en los casos en los que un comando en particular, tal como el comando "**find**" de Unix, requiere períodos de tiempo prolongados para completarse. Si esta palabra clave no está especificada, el tiempo de espera predeterminado para las auditorías CMD_EXEC es cinco minutos. Ejemplo: `timeout: "600"`

type	CHKCONFIG CMD_EXEC FILE_CHECK FILE_CHECK_NOT FILE_CONTENT_CHECK FILE_CONTENT_CHECK_NOT GRAMMAR_CHECK PKG_CHECK PROCESS_CHECK RPM_CHECK SVC_PROP XINETD_SVC
value	La palabra clave **value** resulta útil para comprobar si una opción del sistema confirma el valor de la directiva. Ejemplo: value: "90..max" La palabra clave **value** se puede especificar como rango [number..max]. Si el valor se encuentra entre el número especificado y "max", la comprobación se aprobará.

Elementos personalizados

Un elemento personalizado constituye una comprobación completa establecida en función de las palabras clave definidas anteriormente. La siguiente es una lista de elementos personalizados. Cada comprobación comienza con una etiqueta "**<custom_item>**" y finaliza con "**</custom_item>**". Entre las etiquetas se encuentran las listas de una o más palabras clave que son interpretadas por el analizador sintáctico de comprobaciones de compatibilidad para llevar a cabo dichas comprobaciones.

Las comprobaciones de auditoría personalizadas pueden usar "**</custom_item>**" y "**</item>**" de manera indistinta para la etiqueta de cierre.

AUDIT_XML

La comprobación de auditoría "AUDIT_XML" le permite examinar y auditar el contenido de un archivo XML aplicando primero transformaciones XSL, extrayendo información relevante, y luego determinando la compatibilidad según las palabras clave **regex**, **expect** y **not_expect** (consulte el Apéndice C para obtener más información). La comprobación está compuesta por cuatro o más palabras clave, el tipo de palabras clave, el archivo de descripción y las directivas xsl_stmt (obligatorias), que están seguidas por las palabras clave **regex**, **expect** o **not_expect** para auditar el contenido.

Ejemplo:

```
<custom_item>
 type: AUDIT XML
 description: "1.14 - Ensure Oracle Database persistence plugin is set correctly -
       'DatabasePersistencePlugin'"
 file: "/opt/jboss-5.0.1.GA/server/all/deploy/ejb2-timer-service.xml"
 xsl_stmt: "<xsl:template match=\"server\">"
 xsl_stmt: "DatabasePersistencePlugin = <xsl:value-of
       select=\"/server/mbean[@code='org.jboss.ejb.txtimer.DatabasePersistencePolicy']/
       attribute[@name='DatabasePersistencePlugin']/text()\"/>"
 xsl_stmt: "</xsl:template>"
 regex: "DatabasePersistencePlugin = .+"
 not_expect: "org.jboss.ejb.txtimer.GeneralPurposeDatabasePersistencePlugin"
```

```
</custom_item>
```

Tenga en cuenta que el archivo de palabras clave acepta caracteres comodín. Por ejemplo:

```
<custom_item>
 type: AUDIT_XML
 description: "1.14 - Ensure Oracle Database persistence plugin is set correctly -
       'DatabasePersistencePlugin'"
 file: "/opt/jboss-5.0.1.GA/server/all/deploy/ejb2-*.xml"
 xsl_stmt: "<xsl:template match=\"server\">"
 xsl_stmt: "DatabasePersistencePlugin = <xsl:value-of
       select=\"/server/mbean[@code='org.jboss.ejb.txtimer.DatabasePersistencePolicy']/
       attribute[@name='DatabasePersistencePlugin']/text()\"/>"
 xsl_stmt: "</xsl:template>"
 regex: "DatabasePersistencePlugin = .+"
 not_expect: "org.jboss.ejb.txtimer.GeneralPurposeDatabasePersistencePlugin"
</custom_item>
```

CHKCONFIG

La comprobación de auditoría “CHKCONFIG” permite la interacción con la utilidad “**chkconfig**” en el sistema Red Hat remoto en el que se realiza la auditoría. Esta comprobación consiste en cinco palabras clave obligatorias: **type**, **description**, **service**, **levels** y **status**.

La auditoría CHKCONFIG solo funciona en sistemas Red Hat o en un derivado de un sistema Red Hat, tal como Fedora.

Ejemplo:

```
<custom_item>
 type: CHKCONFIG
 description: "Make sure that xinetd is disabled"
 service: "xinetd"
 levels: "123456"
 status: OFF
</custom_item>
```

CMD_EXEC

Es posible ejecutar comandos en el host remoto y comprobar que los resultados coincidan con lo esperado. Este tipo de comprobación se debe usar con sumo cuidado, ya que no siempre es portátil en distintos tipos de Unix.

La palabra clave **quiet** indica a Nessus que **no** muestre los resultados del comando que tuvo un error. Puede estar establecida en “YES” (SÍ) o “NO” (NO). De manera predeterminada se encuentra establecida en “NO” (NO), y así aparece el resultado del comando. De manera similar, la palabra clave “**dont_echo_cmd**” limita los resultados al mostrar los resultados del comando, pero no el comando en sí.

La palabra clave **nosudo** permite al usuario indicar a Nessus que **no** use sudo para ejecutar el comando, estableciéndolo en “YES” (SÍ). De manera predeterminada se encuentra establecido en “NO” (NO), y sudo siempre se usa cuando está configurado para ello.

Ejemplo:

```
<custom_item>
 type: CMD_EXEC
 description: "Make sure that we are running FreeBSD 4.9 or higher"
 cmd: "uname -a"
 timeout: 7200
 expect: "FreeBSD (4\.(9|[1-9][0-9])|[5-9]\.)"
 dont_echo_cmd: YES
</custom_item>
```

FILE_CHECK

Las auditorías de compatibilidad de Unix normalmente realizan una prueba para determinar la existencia y la configuración de un archivo determinado. La auditoría “FILE_CHECK” emplea cuatro o más palabras clave para permitir la especificación de estas comprobaciones. Las palabras clave **`type`**, **`description`** y **`file`** son obligatorias, y están seguidas de una o más comprobaciones. La sintaxis actual admite la comprobación de los permisos de propietario, grupo y archivo.

Es posible usar comodines en FILE_CHECK (por ejemplo, **`/var/log/*`**). Sin embargo, tenga en cuenta que los comodines solo se expandirán a archivos, no a directorios. Si se especifica un comodín y se deben omitir de la búsqueda uno o más archivos coincidentes, use la palabra clave “**`ignore`**” para especificar los archivos que se omitirán.

Las palabras clave permitidas son las siguientes:

```
uid: Numeric User ID (e.g., 0)
gid: Numeric Group ID (e.g., 500)
check_uneveness: YES
system: System type (e.g., Linux)
description: Text description of the file check
file: Full path and file to check (e.g., /etc/sysconfig/sendmail)
owner: Owner of the file (e.g., root)
group Grupo propietario del archivo (por ejemplo, bin)
mode: Permission mode (e.g., 644)
mask: File umask (e.g., 133)
md5: The MD5 hash of a file (e.g., 88d3dbe3760775a00b900a850b170fcd)
ignore: A file to ignore (e.g., /var/log/secure)
attr: A file attribute (e.g., ----i--------)
```

Los permisos de archivos se consideran desiguales si el “group” (grupo) u “other” (otro) tienen permisos adicionales en comparación con “owner” (propietario) o si “other” (otro) tiene permisos adicionales en comparación con “group” (grupo).

A continuación se incluyen algunos ejemplos:

```
<custom_item>
 system: "Linux"
 type: FILE_CHECK
 description: "Permission and ownership check for /etc/default/cron"
 file: "/etc/default/cron"
 owner: "bin"
 group: "bin"
 mode: "-r--r--r--"
</custom_item>
```

```
<custom_item>
 system: "Linux"
 type: FILE_CHECK
 description: "Permission and ownership check for /etc/default/cron"
```

```
 file: "/etc/default/cron"
 owner: "bin"
 group: "bin"
 mode: "444"
</custom_item>
```

```
<custom_item>
 system: "Linux"
 type: FILE_CHECK
 description: "Make sure /tmp has its sticky bit set"
 file: "/tmp"
 mode: "1000"
</custom_item>
```

```
<custom_item>
 type: FILE_CHECK
 description: "/etc/passwd has the proper md5 set"
 required: YES
 file: "/etc/passwd"
 md5: "ce35dc081fd848763cab2cfd442f8c22"
</custom_item>
```

```
<custom_item>
 type: FILE_CHECK
 description: "Ignore maillog in the file mode check"
 required: YES
 file: "/var/log/m*"
 mode: "1000"
 ignore: "/var/log/maillog"
</custom_item>
```

FILE_CHECK_NOT

La auditoría “FILE_CHECK_NOT” consiste en tres o más palabras clave. Las palabras clave **`type`**, **`description`** y **`file`** son obligatorias, y están seguidas de una o más comprobaciones. La sintaxis actual admite la comprobación de los permisos de propietario, grupo y archivo. De manera similar a la auditoría FILE_CHECK, la palabra clave “**`ignore`**” se puede usar para omitir uno o más archivos si se especifica un comodín de archivos.

Esta función es la opuesta a FILE_CHECK. Una directiva tiene errores si un archivo no existe, o bien si el modo en el que se encuentra es el mismo que el definido en la comprobación en sí.

Es posible usar comodines en FILE_CHECK_NOT (por ejemplo, **`/var/log/*`**). Sin embargo, tenga en cuenta que los comodines solo se expandirán a archivos, no a directorios.

A continuación se incluyen algunos ejemplos:

```
<custom_item>
 type: FILE CHECK NOT
 description: "Make sure /bin/bash does NOT belong to root"
 file: "/bin/bash"
 owner: "root"
</custom_item>
```

```
<custom_item>
 type: FILE_CHECK_NOT
 description: "Make sure that /usr/bin/ssh does NOT exist"
 file: "/usr/bin/ssh"
</custom_item>
```

```
<custom_item>
 type: FILE_CHECK_NOT
 description: "Make sure /root is NOT world writeable"
 file: "/root"
 mode: "0777"
</custom_item>
```

FILE_CONTENT_CHECK

Al igual que con la prueba de la existencia y la configuración de un archivo, el contenido de los archivos de texto también se puede analizar. Se pueden usar expresiones regulares para buscar en una o más ubicaciones el contenido existente. Use la palabra clave "**ignore**" para omitir uno o más archivos de las ubicaciones de búsqueda especificadas.

El campo **string_required** puede configurarse para especificar si la cadena auditada que se busca debe estar presente o no. Si esta opción no está configurada, se supone que es necesario. El campo **file_required** puede configurarse para especificar si el archivo auditado debe estar presente o no. Si esta opción no está configurada, se supone que es necesario.

A continuación se incluyen algunos ejemplos:

```
<custom_item>
 system: "Linux"
 type: FILE_CONTENT_CHECK
 description: "This check reports a problem when the log level setting in the
       sendmail.cf file is less than the value set in your security policy."
 file: "sendmail.cf"
 regex: ".*LogLevel=.*$"
 expect: ".*LogLevel=9"
</custom_item>
```

```
<custom_item>
 system: "Linux"
 type: FILE CONTENT CHECK
 file: "sendmail.cf"
 search_locations: "/etc:/etc/mail:/usr/local/etc/mail/"
 regex: ".*PrivacyOptions=".*"
 expect: ".*PrivacyOptions=.*,novrfy,.*"
</custom_item>
```

```
<custom_item>
 #System: "Linux"
 type: FILE_CONTENT_CHECK
 description: "FILE_CONTENT_CHECK"
 file: "/root/test2/foo*"
 # ignore single file
 ignore: "/root/test/2"
```

```
 # ignore all files in a directory
 ignore: "/root/test/*"
 #ignore certain files from a directory
 ignore: "/root/test/foo*"
 regex: "FOO"
 expect: "FOO1"
 file_required: NO
 string_required: NO
</custom_item>
```

Al agregar “~” a un parámetro de archivo, es posible hacer que FILE_CONTENT_CHECK analice los directorios principales del usuario en busca de contenido incompatible.

```
<custom_item>
 system: "Linux"
 type: FILE_CONTENT_CHECK
 description: "Check all user home directories"
 file: "~/.rhosts"
 ignore: "/.foo"
 regex: "\\+"
 expect: "\\+"
</custom_item>
```

FILE_CONTENT_CHECK_NOT

Esta auditoría examina el contenido de un archivo para determinar si existe una coincidencia con la descripción de la regex en el campo **regex**. Esta función niega FILE_CONTENT_CHECK. Es decir, una directiva tiene errores si la regex **coincide** en el archivo. Use la palabra clave “**ignore**” para omitir uno o más archivos de las ubicaciones de búsqueda especificadas.

Este elemento de directiva comprueba si el archivo contiene la expresión regular **regex**, y que esta expresión no coincida con **expect**.

El tipo permitido es el siguiente:

```
value_type: POLICY_TEXT
value_data: "PATH\Filename"
regex: "regex"
expect: "regex"
```

Tanto **regex** como **expect** deben especificarse en esta comprobación.

A continuación se incluye un ejemplo:

```
<custom_item>
 type: FILE_CONTENT_CHECK_NOT
 description: "Make sure NIS is not enabled on the remote host by making sure that
       '+::' is not in /etc/passwd"
 file: "/etc/passwd"
 regex: "^\+::"
 expect: "^\+::"
 file_required: NO
 string_required: NO
</custom_item>
```

GRAMMAR_CHECK

La comprobación de auditoría “GRAMMAR_CHECK” examina el contenido de un archivo y busca coincidencias respecto de una gramática definida vagamente (compuesta por una o varias instrucciones regex). Si **una** línea del archivo destino no coincide con ninguna de las instrucciones regex, la prueba tendrá un error.

Ejemplo:

```
<custom_item>
 type: GRAMMAR_CHECK
 description: "Check /etc/securetty contents are OK."
 file: "/etc/securetty"
 regex: "console"
 regex: "vc/1"
 regex: "vc/2"
 regex: "vc/3"
 regex: "vc/4"
 regex: "vc/5"
 regex: "vc/6"
 regex: "vc/7"
</custom_item>
```

MACOSX_DEFAULTS_READ

La comprobación de auditoría “MACOSX_DEFAULTS_READ” examina los valores predeterminados de sistema en MAC OS X. Esta comprobación se comporta de manera diferente si se definen determinadas propiedades.

Si **`plist_user`** se define como “all”, se auditan todas las configuraciones de usuario; de lo contrario, se audita la configuración de usuario especificada.

Si la propiedad **`byhost`** se define como `YES` además de definir la propiedad **`plist_user`**, se ejecuta la siguiente consulta:

```
/usr/bin/defaults -currentHost read /Users/foo/Library/Preferences/ByHost/plist_name
      plist_item
```

Si la propiedad **`byhost`** no se define (y la propiedad **`plist_user`** sí se define), se ejecuta la siguiente consulta:

```
/usr/bin/defaults -currentHost read /Users/foo/Library/Preferences/plist_name
      plist_item
```

Si la propiedad **`byhost`** no se define (y la propiedad **`plist_user`** tampoco se define), se ejecuta la siguiente consulta:

```
/usr/bin/defaults -currentHost read plist_name plist_item
```

Se admiten las siguientes propiedades:

```
plist_name: la plist (lista de propiedades) que queremos consultar, por ejemplo
com.apple.digihub
plist_item: el elemento de plist (lista de propiedades) a auditar, por ejemplo
com.apple.digihub.blank.cd.appeared
plist_option: CANNOT_BE_NULL. Si esto se define como CANNOT_BE_NULL, la comprobación es
desaprobada si la configuración auditada no está definida.
byhost : YES. Definir byhost como YES (SÍ) da como resultado una consulta ligeramente
diferente.
```

Ejemplos:

```
<custom_item>
 system: "Darwin"
 type: MACOSX_DEFAULTS_READ
 description: "Automatic actions must be disabled for blank CDs - 'action=1;'"
 plist_user: "all"
 plist_name: "com.apple.digihub"
 plist_item: "com.apple.digihub.blank.cd.appeared"
 regex: "\\s*action\\s*=\\s*1;"
 plist_option: CANNOT_BE_NULL
</custom_item>


<custom_item>
 system: "Darwin"
 type: MACOSX_DEFAULTS_READ
 description: "System must have a password-protected screen saver configured to DoD"
 plist_user: "all"
 plist_name: "com.apple.screensaver"
 byhost: YES
 plist_item: "idleTime"
 regex: "[A-Za-z0-9_-]+\\s*=\\s*(900|[2-8][0-9][0-9]|1[8-9][0-9])$"
 plist_option: CANNOT_BE_NULL
</custom_item>

<custom_item>
 system: "Darwin"
 type: MACOSX_DEFAULTS_READ
 description: "System must have a password-protected screen saver configured to DoD"
 plist_name: "com.apple.screensaver"
 plist_item: "idleTime"
 regex: "[A-Za-z0-9_-]+\\s*=\\s*(900|[2-8][0-9][0-9]|1[8-9][0-9])$"
 plist_option: CANNOT_BE_NULL
</custom_item>
```

PKG_CHECK

La comprobación de auditoría “PKG_CHECK” realiza una **`pkgchk`** en un sistema SunOS. La palabra clave **`pkg`** se usa para especificar el paquete para buscar, y la palabra clave **`operator`** especifica la condición para aprobar o no la comprobación en función de la versión del paquete instalado.

Ejemplos:

```
<custom_item>
 system: "SunOS"
 type: PKG_CHECK
 description: "Make sure SUNWcrman is installed"
 pkg: "SUNWcrman"
 required: YES
</custom_item>
```

```
<custom_item>
 system: "SunOS"
 type: PKG_CHECK
```

```
 description: "Make sure SUNWcrman is installed and is greater than 9.0.2"
 pkg: "SUNWcrman"
 version: "9.0.2"
 operator: "gt"
 required: YES
</custom_item>
```

PROCESS_CHECK

Al igual que con las comprobaciones de archivos, se pueden probar los procesos en ejecución en una plataforma Unix auditada. La implementación ejecuta el comando "**chkconfig –list**" para obtener una lista de procesos en ejecución.

Ejemplos:

```
<custom_item>
 system: "Linux"
 type: PROCESS_CHECK
 name: "auditd"
 status: OFF
</custom_item>
```

```
<custom_item>
 system: "Linux"
 type: PROCESS_CHECK
 name: "syslogd"
 status: ON
</custom_item>
```

RPM_CHECK

La comprobación de auditoría "RPM_CHECK" se usa para comprobar los números de versión de los paquetes RPM instalados en el sistema remoto. Esta comprobación consiste en cuatro palabras clave obligatorias: **type**, **description**, **rpm** y **operator**, y una palabra clave opcional: **required**. La palabra clave **rpm** se usa para especificar el paquete a buscar, y la palabra clave **operator** especifica la condición para aprobar o no la comprobación en función de la versión del paquete RPM instalado.

El uso de las comprobaciones RPM no puede trasladarse por las distribuciones de Linux. Por lo tanto, el empleo de RPM_CHECK no se considera portátil.

A continuación se presentan algunos ejemplos en los que se supone que se encuentra instalada **iproute-2.4.7-10**:

```
<custom_item>
 type: RPM_CHECK
 description: "RPM check for iproute-2.4.7-10 - should pass"
 rpm: "iproute-2.4.7-10"
 operator: "gte"
</custom_item>
```

```
<custom_item>
 type: RPM_CHECK
 description: "RPM check for iproute-2.4.7-10 should fail"
 rpm: "iproute-2.4.7-10"
```

```
 operator: "lt"
 required: YES
</custom_item>
```

```
<custom_item>
 type: RPM_CHECK
 description: "RPM check for iproute-2.4.7-10 should fail"
 rpm: "iproute-2.4.7-10"
 operator: "gt"
 required: NO
</custom_item>
```

```
<custom_item>
 type: RPM_CHECK
 description: "RPM check for iproute-2.4.7-10 should pass"
 rpm: "iproute-2.4.7-10"
 operator: "eq"
 required: NO
</custom_item>
```

SVC_PROP

La comprobación de auditoría “SVC_PROP” permite interactuar con la herramienta “**`svcprop -p`**” en un sistema Solaris 10. Se puede usar para consultar propiedades relacionadas con un servicio específico. La palabra clave **`service`** se usa para especificar el servicio que se audita. La palabra clave **`property`** especifica el nombre de la propiedad que deseamos consultar. La palabra clave **`value`** constituye el valor esperado de la propiedad. El valor esperado también puede ser una regex.

El campo **`svcprop_option`** se puede configurar para que especifique si la cadena auditada que se busca debe estar presente o no. Este campo tiene acceso a CAN_BE_NULL o CANNOT_BE_NULL como argumentos.

Ejemplos:

```
<custom_item>
 type: SVC_PROP
 description: "Check service status"
 service: "cde-ttdbserver:tcp"
 property: "general/enabled"
 value: "false"
</custom_item>
```

```
<custom_item>
 type: SVC_PROP
 description: "Make sure FTP logging is set"
 service: "svc:/network/frp:default"
 property: "inetd_start/exec"
 regex: ".*frpd.*-l"
</custom_item>
```

```
<custom_item>
 type: SVC_PROP
```

```
 description: "Check if ipfilter is enabled - can be missing or not found"
 service: "network/ipfilter:default"
 property: "general/enabled"
 value: "true"
svcprop_option: CAN_BE_NULL
</custom_item>
```

XINETD_SVC

La comprobación de auditoría “XINETD_SVC” se usa para auditar el estado de inicio de los servicios xinetd. La comprobación consiste en cuatro palabras clave obligatorias: **type**, **description**, **service** y **status**.

Esto solo funciona en sistemas Red Hat o en un derivado de un sistema Red Hat, tal como Fedora.

Ejemplo:

```
<custom_item>
 type: XINETD_SVC
 description: "Make sure that telnet is disabled"
 service: "telnet"
 status: OFF
</custom_item>
```

Comprobaciones incorporadas

Las comprobaciones que no se pudieron incluir en las comprobaciones descritas anteriormente se deben escribir como nombres personalizados en NASL. Todas estas comprobaciones entran dentro de la categoría “incorporadas”. Cada comprobación comienza con una etiqueta “**<item>**” y finaliza con “**</item>**”. Dentro de las etiquetas se encuentran las listas de una o más palabras clave que son interpretadas por el analizador sintáctico de comprobaciones de compatibilidad para llevar a cabo dichas comprobaciones. La siguiente es una lista de las comprobaciones disponibles.

La palabra clave “**system**” no se encuentra disponible para las comprobaciones incorporadas y, si se usa, producirá un error de sintaxis.

Administración de contraseñas

En los ejemplos a continuación, <min> y <max> se usan para representar un valor entero y no una cadena para usar en los datos de valores de la auditoría.

En los casos en los que se desconoce el valor mínimo o máximo exacto, reemplace las cadenas “Min” o “Max” por el valor entero.

min_password_length

Uso
<item> name: "min_password_length" description: "This check examines the system configuration for the minimum password length that the passwd program will accept. The check reports a problem if the minimum length is less than the length specified in your policy." except: "user**1**" | "user**2**" (list of users to be excluded) value: "<min>..<max>" </item>

Esta comprobación incorporada garantiza que la longitud de contraseña mínima aplicada en el sistema remoto esté en el rango "**<min>..<max>**". Contar con una longitud de contraseña mínima obliga a los usuarios a elegir contraseñas más complejas.

Sistema operativo	Implementación
Linux	La longitud de contraseña mínima se define como PASS_MIN_LEN en **/etc/login.defs**.
Solaris	La longitud de contraseña mínima se define como PASSLENGTH en **/etc/default/passwd**. Tenga en cuenta que esto también controla la longitud de contraseña máxima.
HP-UX	La longitud de contraseña mínima se define como MIN_PASSWORD_LENGTH en **/etc/default/security**.
Mac OS X	La longitud de contraseña mínima se define como "minChar" en la directiva local, la cual se define mediante el comando **pwpolicy**.

Ejemplo:

```
<item>
 name: "min_password_length"
 description: "Make sure that each password has a minimum length of 6 chars or more"
 value: "6..65535"
</item>
```

max_password_age

Uso
<item> name: "max_password_age" description: "This check reports agents that have a system default maximum password age greater than the specified value and agents that do not have a maximum password age setting." except: "user**1**" | "user**2**" (list of users to be excluded) value: "<min>..<max>" </item>

Esta función incorporada garantiza que la vigencia máxima de la contraseña (es decir, el momento en el cual los usuarios están obligados a cambiar sus contraseñas) se encuentre dentro del rango definido.

Contar con una vigencia máxima de contraseña evita que los usuarios conserven la misma contraseña durante varios años. Cambiar contraseñas a menudo ayuda a evitar que un atacante que posea una contraseña la use por tiempo indefinido.

Sistema operativo	Implementación
Linux	La variable PASS_MAX_DAYS se define en **`/etc/login.defs`**.
Solaris	La variable MAXWEEKS en **`/etc/default/passwd`** define la cantidad máxima de semanas durante la que se puede usar una contraseña.
HP-UX	Este valor está controlado por la variable PASSWORD_MAXDAYS en **`/etc/default/security`**.
Mac OS X	La opción “maxMinutesUntilChangePassword” de la directiva de contraseña (según se estableció mediante la herramienta **`pwpolicy`**) se puede usar para establecer este valor.

Ejemplo:

```
 <item>
  name: "max_password_age"
  description: "Make sure a password can not be used for more than 21 days"
  value: "1..21"
 </item>
```

min_password_age

Uso

`<item>`
` name: "min password age"`
` description: "This check reports agents and users with password history settings that are less than a specified minimum number of passwords."`
` except: "user1" | "user2" (list of users to be excluded)`
` value: "<min>..<max>"`
`</item>`

Esta función incorporada garantiza que la vigencia mínima de la contraseña (es decir, el tiempo que debe transcurrir para que se permita a los usuarios cambiar su contraseña) se encuentre dentro del rango definido.

Contar con una vigencia mínima de contraseña evita que los usuarios cambien las contraseñas con demasiada frecuencia en un intento de omitir la vigencia máxima de contraseña. Algunos usuarios emplean este método para volver a obtener la contraseña original y, de esta forma, sortear los requisitos de cambio de contraseña.

Sistema operativo	Implementación
Linux	La variable PASS_MIN_DAYS se define en **`/etc/login.defs`**.

Solaris	La variable MINWEEKS en **/etc/default/passwd** define la cantidad mínima de semanas durante la que se puede usar una contraseña.
HP-UX	Este valor está controlado por la variable PASSWORD_MINDAYS en **/etc/default/security**.
Mac OS X	No se admite esta opción.

Ejemplo:

```
<item>7
 name: "min_password_age"
 description: "Make sure a password cannot be changed before 4 days while allowing the
       user to change at least after 21 days"
 value: "4..21"
</item>
```

Acceso de root (raíz)
root_login_from_console

Uso

`<item>`
` name: "root_login_from_console"`
` description: "This check makes sure that root can only log in from the system console (not remotely)."`
`</item>`

Esta función incorporada garantiza que el usuario “root” (raíz) pueda iniciar sesión directamente en el sistema remoto solo a través de la consola física.

La fundamentación que subyace en esta comprobación es que las prácticas administrativas recomendadas rechazan el uso directo de la cuenta root (raíz) de modo que se pueda vincular el acceso con una persona específica. En lugar de eso, use una cuenta de usuario genérica (miembro del grupo wheel en los sistemas BSD), y luego use “**su**” (o **sudo**) a fin de elevar privilegios para llevar a cabo tareas administrativas.

Sistema operativo	Implementación
Linux y HP-UX	Asegúrese de que exista **/etc/securetty** y que solo contenga “console”.
Solaris	Asegúrese de que **/etc/default/login** contenga la línea “CONSOLE=/dev/console”.
Mac OS X	No se admite esta opción.

Administración de permisos

accounts_bad_home_permissions

Uso

```
<item>
 name: "accounts_bad_home_permissions"
 description: "This check reports user accounts that have home directories with
incorrect user or group ownerships."
</item>
```

Esta función incorporada garantiza que el directorio principal de cada usuario no privilegiado corresponda a ese usuario, y que los usuarios terceros (ya sea que pertenezcan al mismo grupo o “cualquiera”) no puedan escribir en él. Normalmente se recomienda que los directorios principales de usuarios estén establecidos en el modo 0755 o uno más estricto (por ejemplo, 0700). Esta prueba es satisfactoria si cada directorio principal está configurado correctamente, e insatisfactoria si ocurre lo contrario. Se puede usar aquí cualquiera de las palabras clave **`mode`** o **`mask`** a fin de especificar los niveles de permisos deseados para los directorios principales. La palabra clave **`mode`** aceptará los directorios principales que coincidan exactamente con un nivel especificado, y la palabra clave **`mask`** aceptará los directorios principales que se encuentren en el nivel especificado, o bien en uno más seguro.

Si en el directorio principal de un usuario pudieran escribir terceros, pueden obligar a que el usuario ejecute comandos arbitrarios por alteración de los archivos **`~/.profile`**, **`~/.cshrc`** y **`~/.bashrc`**.

Si se necesita compartir archivos entre usuarios del mismo grupo, normalmente se recomienda usar un directorio especializado que sea grabable para el grupo y no el directorio principal de un usuario.

En el caso de los directorios principales configurados erróneamente, ejecute “**`chmod 0755 <user directory>`**” y cambie la propiedad en consecuencia.

accounts_bad_home_group_permissions

Uso

```
<item>
 name: "accounts_bad_home_group_permissions"
 description: "This check makes sure user home directories are group owned by the user's
primary group."
</item>
```

Esta función incorporada tiene una operación similar a **`accounts_bad_home_permissions`**, pero garantiza que los directorios principales del usuario sean propiedad grupal por parte del grupo principal del usuario.

accounts_without_home_dir

Uso

```
<item>
 name: "accounts_without_home_dir"
 description: "This check reports user accounts that do not have home directories."
</item>
```

Esta función incorporada garantiza que cada usuario tenga un directorio principal. Será aprobada si se atribuye a cada usuario un directorio válido. De lo contrario, tendrá errores. Tenga en cuenta que no se prueban a través de esta comprobación la propiedad ni los permisos de los directorios principales.

Normalmente se recomienda que cada usuario de un sistema tenga un directorio principal definido, ya que algunas herramientas pueden necesitar leerlo o escribir en él (por ejemplo, las comprobaciones **`sendmail`** para un archivo **`~/.forward`**). Si un usuario no necesita iniciar sesión, en su lugar se debe definir un shell inexistente (por ejemplo, **`/bin/false`**). En muchos sistemas, los usuarios sin directorios principales recibirán de todos modos privilegios de inicio de sesión, pero su directorio principal efectivo es /.

invalid_login_shells

Uso

```
<item>
 name: "invalid_login_shells"
 description: "This check reports user accounts with shells which do not exist or is not
listed in /etc/shells."
</item>
```

Esta función incorporada garantiza que cada usuario tenga definido un shell válido en **`/etc/shells`**.

El archivo **`/etc/shells`** es usado por aplicaciones tales como Sendmail y servidores FTP para determinar si un shell es válido en el sistema. Si bien el programa de inicio de sesión no lo usa, los administradores pueden usar este archivo para definir qué shells son válidos en el sistema. La comprobación **`invalid_login_shells`** puede verificar que todos los usuarios del archivo **`/etc/passwd`** estén configurados con shells válidos según se define en el archivo **`/etc/shells`**.

Esto evita las prácticas no autorizadas, tales como el uso de **`/sbin/passwd`** como shell para permitir que los usuarios cambien sus contraseñas. Si no desea que un usuario pueda iniciar sesión, cree un shell no válido en **`/etc/shells`** (por ejemplo, "**`/nonexistent`**") y establézcalo para los usuarios deseados.

Si tiene usuarios sin un shell válido, defina un shell válido para ellos.

login_shells_with_suid

Uso

```
<item>
 name: "login_shells_with_suid"
 description: "This check reports user accounts with login shells that have setuid or
setgid privileges."
</item>
```

Esta función incorporada garantiza que ningún shell tenga capacidades "set-uid".

Un shell "setuid" significa que cada vez que se inicie el shell, el proceso en sí contará con los privilegios establecidos en sus permisos (por ejemplo, un setuid "root" shell otorga privilegios de superusuario a cualquiera).

Contar con "setuid" shell invalida el propósito de tener UID y GID, y hace que el control de acceso sea mucho más complejo.

Elimine el bit SUID de cada shell que sea "setuid".

login_shells_writeable

Uso

```
<item>
 name: "login_shells_writeable"
 description: "This check reports user accounts with login shells that have group or
world write permissions."
</item>
```

Esta función incorporada garantiza que ningún shell sea grabable para todo el mundo/grupo.

Si un shell es grabable para todo el mundo (o grabable para el grupo), entonces los usuarios sin privilegios pueden reemplazarlo por cualquier programa. Esto permite que un usuario malintencionado obligue a otros usuarios de ese shell a ejecutar comandos arbitrarios cuando inicien sesión.

Asegúrese de que los permisos de cada shell estén establecidos correctamente.

login_shells_bad_owner

Uso

```
<item>
 name: "login_shells_bad_owner"
 description: "This check reports user accounts with login shells that are not owned by
root or bin."
</item>
```

Esta función incorporada garantiza que cada shell pertenezca a los usuarios “root” o “bin”.

Al igual que con los shells con permisos no válidos, si un usuario es propietario de un shell usado por otros usuarios, entonces pueden modificarlo para obligar a usuarios terceros a ejecutar comandos arbitrarios cuando inicien sesión.

Solo “root” y/o “bin” deben tener la capacidad para modificar valores binarios en todo el sistema.

Administración de archivos de contraseñas

passwd_file_consistency

Uso

```
<item>
 name: "passwd_file_consistency"
 description: "This check makes sure /etc/passwd is valid."
</item>
```

Esta función incorporada garantiza que cada línea en **`/etc/passwd`** tenga un formato válido (por ejemplo, siete campos separados por dos puntos). Si una línea está formada incorrectamente, se informa y se produce un error en la comprobación.

La presencia de un archivo **`/etc/passwd`** formado incorrectamente puede interrumpir varias herramientas de administración de usuarios. También puede indicar una intrusión o un error en una aplicación de administración de usuarios personalizada. También puede indicar que alguien intentó añadir un usuario con un nombre no válido (en el pasado era muy común crear un usuario con el nombre “toor:0:0” para obtener privilegios root).

Si se considera que la prueba es no compatible, el administrador debe eliminar o corregir las líneas incorrectas en **/etc/passwd**.

passwd_zero_uid

Uso

```
<item>
 name: "passwd_zero_uid"
 description: "This check makes sure that only ONE account has a uid of 0."
</item>
```

Esta función incorporada garantiza que solo una cuenta tenga un UID de “0” en **/etc/passwd**. La finalidad es que quede en reserva para la cuenta “root” (raíz), pero es posible añadir cuentas adicionales con UID 0 que tendrían el mismo acceso privilegiado. Esta prueba resulta satisfactoria si solo una cuenta posee un UID de cero. De lo contrario, tendrá errores.

Un UID de “0” otorga privilegios root (raíz) en el sistema. Un usuario root puede llevar a cabo cualquier acción que desee en el sistema, lo cual normalmente incluye búsquedas en la memoria de otros procesos (o del kernel), leer y escribir cualquier archivo del sistema, etc. Dado que esta cuenta es tan versátil, su uso debe quedar restringido al mínimo indispensable y debe estar bien protegida.

Las prácticas administrativas recomendadas indican que cada UID sea único (de allí la “U” en UID). Tener dos (o más) cuentas con privilegios “root” anula la responsabilidad que puede tener un administrador del sistema respecto de este. Además, muchos sistemas restringen el inicio de sesión directo de root a la consola, solo para que se pueda realizar un seguimiento del uso administrativo. Normalmente, los administradores de sistema deben primero iniciar sesión en **su** propia cuenta y usar el comando **su** para transformarse en root. Una cuenta UID 0 adicional evade esta restricción.

Si se necesita compartir el acceso “root” (raíz) entre usuarios, use una herramienta como **sudo** o **calife** en su lugar (o RBAC en Solaris). Solo debe haber una cuenta con un UID de “0”.

passwd_duplicate_uid

Uso

```
<item>
 name: "passwd_duplicate_uid"
 description: "This check makes sure that every UID in /etc/passwd is unique."
</item>
```

Esta función incorporada garantiza que todas las cuentas que aparecen en **/etc/passwd** tengan un UID exclusivo. La prueba es satisfactoria si todos los UID son exclusivos. De lo contrario, tendrá errores.

Cada usuario en un sistema Unix se identifica por su ID (Identificador) de usuario (UID), un número comprendido entre 0 y 65 535. Si dos usuarios comparten el mismo UID, no solo recibirán los mismos privilegios sino que el sistema los considerará como la misma persona. Esto invalida cualquier tipo de responsabilidad, dado que resulta imposible vincular cada acción con cada usuario (normalmente, el sistema llevará a cabo una búsqueda inversa del UID y usará el primer nombre de las cuentas que comparten el UID al mostrar los registros).

Las normas de seguridad, tales como las referencias del CIS, prohíben compartir un UID entre usuarios. Si los usuarios necesitan compartir archivos, se deben usar grupos.

Otorgue a cada usuario del sistema un identificador exclusivo.

passwd_duplicate_gid

Uso

```
<item>
 name: "passwd_duplicate_gid"
 description: "This check makes sure that every GID in /etc/passwd is unique."
</item>
```

Esta función incorporada garantiza que el identificador de grupo principal (GID) de cada usuario sea exclusivo. La prueba es satisfactoria si cada usuario cuenta con un GID exclusivo. De lo contrario, tendrá errores.

Las normas de seguridad recomiendan la creación de un grupo por usuario (normalmente con el mismo nombre que el nombre de usuario). Mediante esta configuración los archivos creados por el usuario están normalmente “protegidos de manera predeterminada”, ya que pertenecen a su grupo principal y, por lo tanto, solo pueden ser modificados por el usuario mismo. Si el usuario desea que el archivo sea propiedad también de los otros miembros del grupo, deberá usar de manera explícita el comando **chgrp** para modificar la propiedad.

Otra ventaja de este método consiste en que unifica la administración de pertenencia a grupos en un único archivo (**/etc/group**) en lugar de una mezcla de **/etc/passwd** y **/etc/group**.

Cree para cada usuario un grupo con el mismo nombre. Administre la propiedad del grupo solo a través de **/etc/group**.

passwd_duplicate_username

Uso

```
<item>
 name: "passwd_duplicate_username"
 description: "This check makes sure that every username in /etc/passwd is unique."
</item>
```

Esta función incorporada garantiza que cada nombre de usuario en **/etc/passwd** sea exclusivo. Es satisfactoria si se cumple esta regla. De lo contrario, tendrá errores.

Los nombres de usuarios duplicados en **/etc/passwd** crean problemas, ya que no queda claro de qué cuenta son los privilegios que se están usando.

El comando **adduser** no le permitirá crear un nombre de usuario duplicado. Este tipo de configuración normalmente significa que el sistema se encuentra en peligro, que las herramientas para llevar a cabo las tareas de administración de usuarios tienen errores, o que el archivo **/etc/passwd** se modificó manualmente.

Elimine los nombres de usuario duplicados o modifíquelos para que sean diferentes.

passwd_duplicate_home

Uso

```
<item>
 name: "passwd_duplicate_home"
 description: "(arbitrary user comment)"
</item>
```

Esta función incorporada garantiza que cada usuario que no pertenezca al sistema (cuyo UID es mayor que 100) en **/etc/passwd** posea un directorio principal exclusivo.

Cada nombre de usuario en **/etc/passwd** debe contar con un directorio principal exclusivo. Si hay usuarios que comparten el mismo directorio principal, uno puede obligar a que los otros ejecuten comandos arbitrarios por modificación de los archivos de inicio (**.profile**, etc.) o por colocación de binarios rogue en el directorio principal mismo. Además, un directorio principal compartido invalida la responsabilidad del usuario.

Los requisitos de compatibilidad exigen que cada usuario cuente con un directorio principal exclusivo.

passwd_shadowed

Uso

```
<item>
 name: "passwd_shadowed"
 description: "(arbitrary user comment)"
</item>
```

Esta comprobación incorporada garantiza que cada contraseña de **/etc/passwd** se encuentre “oculta” (es decir, que resida en otro archivo).

Dado que **/etc/passwd** tiene permiso de lectura para todo el mundo, almacenar allí hashes de contraseñas de los usuarios permite que cualquiera con acceso a él tenga la capacidad de ejecutar programas de violación de contraseñas. Los intentos de descubrir la contraseña de un usuario mediante un ataque de fuerza bruta (intentos de inicio de sesión repetidos con introducción de distintas contraseñas cada vez) normalmente se detectan en los archivos de registro del sistema. Si el archivo **/etc/passwd** contiene los hashes de las contraseñas, el archivo se podría copiar fuera de línea y usar como información para un programa de violación de contraseñas. Esto permite que un atacante posea la capacidad de obtener las contraseñas de los usuarios sin ser detectado.

La mayoría de los sistemas Unix modernos poseen archivos de contraseñas ocultos. Consulte la documentación de su sistema para obtener información sobre cómo habilitar las contraseñas ocultas en su sistema.

passwd_invalid_gid

Uso

```
<item>
 name: "passwd invalid gid"
 description: "This check makes sure that every GID defined in /etc/passwd exists in
/etc/group."
</item>
```

Esta función incorporada garantiza que cada identificador de grupo (GID) que aparece en **/etc/passwd** exista en **/etc/group**. Es satisfactoria si cada GID se encuentra definido correctamente. De lo contrario, tendrá errores.

Cada vez que se define una identificación grupal en **/etc/passwd**, deberá aparecer inmediatamente en **/etc/group**. De lo contario, el sistema se encuentra en un estado de incoherencia y pueden surgir problemas.

Analice el siguiente escenario: un usuario (“bob”) tiene un UID de 1000 y un GID de 4000. El GID no se encuentra definido en **/etc/group**, lo que significa que hoy el grupo principal del usuario no le otorga ningún privilegio. Unos meses después el administrador del sistema modifica **/etc/group**, añade el “admin” del grupo y selecciona el GID N.º 4000 “no usado” para identificarlo. Ahora el usuario “bob” pertenece al grupo de “admin” de manera predeterminada, a pesar de que esto no fue el objetivo inicial.

Modifique **/etc/group** para añadir los GID faltantes.

Administración de archivos de grupos
group_file_consistency

Uso

```
<item>
 name: "group_file_consistency"
 description: "This check makes sure /etc/group is valid."
</item>
```

Esta función incorporada garantiza que cada línea de **/etc/group** tenga un formato válido (por ejemplo, tres elementos separados por dos puntos y una lista de usuarios). Si una línea está formada incorrectamente, se informa y se produce un error en la comprobación.

La presencia de un archivo **/etc/group** formado incorrectamente puede interrumpir varias herramientas de administración de usuarios. También puede indicar una intrusión o un error en una aplicación de administración de usuarios personalizada. También puede mostrar que alguien intentó añadir un usuario con un nombre de grupo no válido.

Modifique el archivo **/etc/group** para corregir las líneas formadas de forma incorrecta.

group_zero_gid

Uso

```
<item>
 name: "group_zero_gid"
 description: "This check makes sure that only ONE group has a gid of 0."
</item>
```

Esta función incorporada garantiza que solo un grupo tenga un identificador de grupo (GID) de 0. Será aprobada si solo un grupo tiene un GID de 0. De lo contrario, tendrá errores.

Un GID de “0” significa que los usuarios que son miembros de este grupo también son miembros del grupo principal de root. Esto les otorga privilegios root respecto de cualquier archivo con permisos de grupo root.

Si desea definir un grupo de administradores, en lugar de ello cree un grupo “admin”.

group_duplicate_name

Uso

```
<item>
 name: "group_duplicate_name"
 description: "This check makes sure that every group name in /etc/group is unique."
</item>
```

Esta comprobación incorporada garantiza que cada nombre de grupo sea exclusivo. Es satisfactoria si se cumple esta regla. De lo contrario, tendrá errores.

Los nombres de grupos duplicados en **/etc/group** crean problemas, ya que no queda claro de qué grupo son los privilegios que se están usando. Esto significa que un nombre de grupo duplicado puede tener finalmente miembros o privilegios que, en primer lugar, no debería haber tenido.

Elimine o cambie los nombres de grupo duplicados.

group_duplicate_gid

Uso

```
<item>
 name: "group_duplicate_gid"
 description: "(arbitrary user comment)"
</item>
```

Cada grupo en un sistema Unix se identifica por su ID de grupo (GID), un número entre 0 y 65 535. Si dos grupos comparten el mismo GID, no solo recibirán los mismos privilegios sino que el sistema los considerará como el mismo grupo. Esto invalida el propósito de usar grupos para separar los privilegios de usuarios.

Las normas de seguridad prohíben compartir un GID entre grupos. Si dos grupos necesitan tener los mismos privilegios, deben tener los mismos usuarios.

Elimine los grupos duplicados, o asigne a uno de los duplicados un nuevo GID exclusivo.

group_duplicate_members

Uso

```
<item>
 name: "group_duplicate_members"
 description: "This check makes sure that every member of a group is listed once."
</item>
```

Esta función incorporada garantiza que cada miembro de un grupo solo aparezca una vez. Se aprobará si cada miembro es exclusivo. De lo contrario, tendrá errores.

Cada miembro de un grupo solo debe aparecer enumerado una vez. Si bien aparecer varias veces no produce ningún problema en el sistema operativo subyacente, hace las cosas más difíciles para el administrador del sistema, ya que la revocación de privilegios se vuelve más compleja. Por ejemplo, si el grupo “admin” cuenta con los miembros “alice,bob,charles,daniel,bob”, “bob” deberá eliminarse dos veces si se debiera revocar sus privilegios.

Asegúrese de que cada miembro aparezca solo una vez.

group_nonexistant_users

Uso

```
<item>
 name: "group_nonexistant_users"
 description: "This check makes sure that every member of a group actually exists."
</item>
```

Esta comprobación garantiza que cada miembro de un grupo exista efectivamente en **/etc/passwd**.

Tener usuarios inexistentes en **/etc/group** supone prácticas de administración incompletas. El usuario no existe porque se escribió incorrectamente o porque cuando se lo eliminó del sistema no se lo eliminó del grupo.

No se recomienda que permanezcan usuarios “fantasmas” en **/etc/group**. Si un usuario con el mismo nombre de usuario se añadiera posteriormente, es posible que el usuario tenga privilegios de grupo que no se deberían otorgar.

Elimine los usuarios inexistentes de **/etc/group**.

Entorno root (raíz)

dot_in_root_path_variable

Uso

```
<item>
 name: "dot_in_root_path_variable"
 description: "This check makes sure that root's $PATH variable does not contain any
relative path."
</item>
```

Esta comprobación garantiza que el directorio de trabajo actual (“.”) no se incluya en la ruta ejecutable del usuario root (raíz). Este tipo de medida evita que un usuario malintencionado realice una escalada de privilegios hasta transformarse en un superusuario al obligar a un administrador que haya iniciado sesión como root a ejecutar un caballo de Troya que pueda estar instalado en el directorio de trabajo actual.

writeable_dirs_in_root_path_variable

Uso

```
<item>
 name: "writeable_dirs_in_root_path_variable"
 description: "This check makes sure that root's $PATH variable does not contain any
writeable directory."
</item>
```

Esta comprobación informa sobre todos los directorios con permiso de escritura para todo el mundo/grupo en la variable PATH de usuarios root. Todos los directorios que devuelve esta comprobación se deben examinar cuidadosamente, y se deben eliminar de los directorios los permisos grabables para todo el mundo o el grupo que sean innecesarios, de la siguiente forma:

```
  # chmod go-w path/to/directory
```

Permisos de archivos

find_orphan_files

Uso

```
<item>
 name: "find_orphan_files"
 description: "This check finds all the files which are 'orphaned' (ie: whose owner is
an invalid UID or GID)."
 # Globs allowed (? and *)
 (optional) basedir: "<directory>"
```

```
  (optional) ignore: "<directory>"
  (optional) dir: "<directory>"
</item>
```

Esta comprobación informa sobre todos los archivos sin propietario que haya en el sistema.

De manera predeterminada, la búsqueda se efectúa de forma recursiva en el directorio “/”. Esto puede hacer que la comprobación sea extremadamente lenta al ejecutarse, de acuerdo con la cantidad de archivos que haya en el sistema remoto. Sin embargo, de ser necesario se puede cambiar el directorio base predeterminado en el que se llevará a cabo la búsqueda mediante la palabra clave opcional **basedir**. También es posible omitir ciertos archivos dentro del directorio base para que no se realice la búsqueda en ellos, mediante otra palabra clave opcional: **ignore**. Al realizar búsquedas en sistemas de archivos, de forma predeterminada omitirá los directorios montados en NFS, a menos que se hayan especificado con la palabra clave opcional **dir**.

Dada la naturaleza de la comprobación, es normal que siga ejecutándose por un par de horas, de acuerdo con el tipo de sistema que se esté analizando. Se ha establecido un valor de tiempo de espera predeterminado de cinco horas, que representa el tiempo después del cual Nessus dejará de procesar resultados para esta comprobación. Este valor no se puede cambiar.

Ejemplo:

```
 <item>
  name: "find_orphan_files"
  description: "This check finds all the files which are 'orphaned' (ie: whose owner is
        an invalid UID or GID)."
  # Globs allowed (? and *)
  basedir: "/tmp"
  ignore: "/tmp/foo"
  ignore: "/tmp/b*"
 </item>
```

find_world_writeable_files

Uso

```
<item>
 name: "find_world_writeable_files"
 description: "This check finds all the files which are world writeable and whose sticky
bit is not set."
 # Globs allowed (? and *)
 (optional) basedir: "<directory>"
 (optional) ignore: "<directory>"
 (optional) dir: "<directory>"
</item>
```

Esta comprobación informa sobre todos los archivos grabables para todo el mundo en el sistema remoto. Lo ideal es que no haya ninguno de estos archivos en el sistema remoto; el resultado de esta comprobación debería ser cero. Sin embargo, en algunos casos, y de acuerdo con las necesidades organizativas, es posible que haya un requisito que indique que debe haber archivos grabables para todo el mundo. Todos los archivos que devuelva esta comprobación se deben auditar cuidadosamente, y los archivos que no necesiten realmente atributos grabables para todo el mundo deben eliminarse de la siguiente forma:

```
 # chmod o-w world_writeable_file
```

De manera predeterminada, la búsqueda se efectúa de forma recursiva en el directorio “/”. Esto puede hacer que la comprobación sea extremadamente lenta al ejecutarse, de acuerdo con la cantidad de archivos que haya en el sistema remoto. Sin embargo, de ser necesario se puede cambiar el directorio base predeterminado en el que se llevará a cabo la búsqueda mediante la palabra clave opcional **basedir**. También es posible omitir ciertos archivos dentro del directorio base para que no se realice la búsqueda en ellos, mediante otra palabra clave opcional: **ignore**. Al realizar búsquedas en sistemas de archivos, de forma predeterminada omitirá los directorios montados en NFS, a menos que se hayan especificado con la palabra clave opcional **dir**.

Dada la naturaleza de la comprobación, es normal que siga ejecutándose por un par de horas, de acuerdo con el tipo de sistema que se esté analizando. Se ha establecido un valor de tiempo de espera predeterminado de cinco horas, que representa el tiempo después del cual Nessus dejará de procesar resultados para esta comprobación. Este valor no se puede cambiar.

Ejemplo:

```
 <item>
  name: "find_world_writeable_files"
  description: "Search for world-writable files"
  # Globs allowed (? and *)
  basedir: "/tmp"
  ignore: "/tmp/foo"
  ignore: "/tmp/bar"
 </item>
```

find_world_writeable_directories

Uso

```
<item>
 name: "find_world_writeable_directories"
 description: "This check finds all the directories which are world writeable and whose
sticky bit is not set."
 # Globs allowed (? and *)
 (optional) basedir: "<directory>"
 (optional) ignore: "<directory>"
 (optional) dir: "<directory>"
</item>
```

Esta comprobación informa sobre todos los directorios que son grabables para todo el mundo y cuyo sticky bit (permiso de acceso) no está establecido en el sistema remoto. Comprobar si el sticky bit está establecido para todos los directorios grabables para todo el mundo garantiza que solo el propietario de un archivo que está en un directorio pueda eliminar el archivo. Esto evita que algún otro usuario elimine el archivo de forma accidental o intencional.

De manera predeterminada, la búsqueda se efectúa de forma recursiva en el directorio “/”. Esto puede hacer que la comprobación sea extremadamente lenta al ejecutarse, de acuerdo con la cantidad de archivos que haya en el sistema remoto. Sin embargo, de ser necesario se puede cambiar el directorio base predeterminado en el que se llevará a cabo la búsqueda mediante la palabra clave opcional **basedir**. También es posible omitir ciertos archivos dentro del directorio base para que no se realice la búsqueda en ellos, mediante otra palabra clave opcional: **ignore**. Al realizar búsquedas en sistemas de archivos, de forma predeterminada omitirá los directorios montados en NFS, a menos que se hayan especificado con la palabra clave opcional **dir**.

Dada la naturaleza de la comprobación, es normal que siga ejecutándose por un par de horas, de acuerdo con el tipo de sistema que se esté analizando. Se ha establecido un valor de tiempo de espera predeterminado de cinco horas, que

representa el tiempo después del cual Nessus dejará de procesar resultados para esta comprobación. Este valor no se puede cambiar.

Ejemplo:

```
 <item>
  name: "find_world_writeable_directories"
  description: "This check finds all the directories which are world writeable and
        whose sticky bit is not set."
  # Globs allowed (? and *)
  basedir: "/tmp"
  ignore: "/tmp/foo"
  ignore: "/tmp/b*"
 </item>
```

find_world_readable_files

Uso

```
<item>
 name: "find_world_readable_files"
 description: "This check finds all the files in a directory with world readable
permissions."
 # Globs allowed (? and *)
 (optional) basedir: "<directory>"
 (optional) ignore: "<directory>"
 (optional) dir: "<directory>"
</item>
```

Esta comprobación notifica todos los archivos legibles por todo el mundo. Buscar archivos legibles, por ejemplo en directorios principales del usuario, garantiza que no haya archivos sensibles a los que puedan acceder otros usuarios (por ejemplo, claves privadas SSH).

De manera predeterminada, la búsqueda se efectúa de forma recursiva en el directorio "/". Esto puede hacer que la comprobación sea extremadamente lenta al ejecutarse, de acuerdo con la cantidad de archivos que haya en el sistema remoto. Sin embargo, de ser necesario se puede cambiar el directorio base predeterminado en el que se llevará a cabo la búsqueda mediante la palabra clave opcional **basedir**. También es posible omitir ciertos archivos dentro del directorio base para que no se realice la búsqueda en ellos, mediante otra palabra clave opcional: **ignore**. Al realizar búsquedas en sistemas de archivos, de forma predeterminada omitirá los directorios montados en NFS, a menos que se hayan especificado con la palabra clave opcional **dir**.

Dada la naturaleza de la comprobación, es normal que siga ejecutándose por un par de horas, de acuerdo con el tipo de sistema que se esté analizando. Se ha establecido un valor de tiempo de espera predeterminado de cinco horas, que representa el tiempo después del cual Nessus dejará de procesar resultados para esta comprobación. Este valor no se puede cambiar.

Ejemplo:

```
 <item>
  name: "find_world_readable_files"
  description: "This check finds all the files in a directory with world readable
        permissions."
  basedir: "/home"
  ignore: "/home/tmp"
```

```
  dir: "/home/extended"
 </item>
```

find_suid_sgid_files

Uso

```
<item>
 name: "find_suid_sgid_files"
 description: "This check finds all the files which have their SUID or SGID bit set."
 # Globs allowed (? and *)
 (optional) basedir: "<directory>"
 (optional) ignore: "<directory>"
 (optional) dir: "<directory>"
</item>
```

Esta comprobación informa sobre todos los archivos que tienen el bit SUID/SGID establecido. Todos los archivos que se informen a través de esta comprobación se deben auditar cuidadosamente, especialmente las secuencias de comandos de shell y los archivos ejecutables propios/internos, como por ejemplo los archivos ejecutables que no se distribuyen con el sistema. Los archivos SUID/SGID presentan el riesgo de escalar los privilegios de un usuario normal hasta adquirir los que posee el propietario o el grupo del archivo. Si es realmente necesario que existan dichos archivos o secuencias de comandos, se deben examinar especialmente para comprobar si permiten la creación de archivos con privilegios elevados.

De manera predeterminada, la búsqueda se efectúa de forma recursiva en el directorio “/”. Esto puede hacer que la comprobación sea extremadamente lenta al ejecutarse, de acuerdo con la cantidad de archivos que haya en el sistema remoto. Sin embargo, de ser necesario se puede cambiar el directorio base predeterminado en el que se llevará a cabo la búsqueda mediante la palabra clave opcional **basedir**. También es posible omitir ciertos archivos dentro del directorio base para que no se realice la búsqueda en ellos, mediante otra palabra clave opcional: **ignore**. Al realizar búsquedas en sistemas de archivos, de forma predeterminada omitirá los directorios montados en NFS, a menos que se hayan especificado con la palabra clave opcional **dir**.

Dada la naturaleza de la comprobación, es normal que siga ejecutándose por un par de horas, de acuerdo con el tipo de sistema que se esté analizando. Se ha establecido un valor de tiempo de espera predeterminado de cinco horas, que representa el tiempo después del cual Nessus dejará de procesar resultados para esta comprobación. Este valor no se puede cambiar.

Ejemplo:

```
 <item>
  name: "find_suid_sgid_files"
  description: "Search for SUID/SGID files"
  # Globs allowed (? and *)
  basedir: "/"
  ignore: "/usr/sbin/ping"
 </item>
```

home_dir_localization_files_user_check
Esta función incorporada verifica si un archivo de localización que está en el directorio principal de un usuario es propiedad del usuario o de la raíz.

Se pueden enumerar uno o más archivos con el token “file”. Sin embargo, si el token “file” falta, la comprobación busca de manera predeterminada los siguientes archivos:

- **.login**

- **.cschrc**
- **.logout**
- **.profile**
- **.bash_profile**
- **.bashrc**
- **.bash_logout**
- **.env**
- **.dtprofile**
- **.dispatch**
- **.emacs**
- **.exrc**

Ejemplo:

```
<item>
 name: "home_dir_localization_files_user_check"
 description: "Check file .foo/.foo2"
 file: ".foo"
 file: ".foo2"
 file: ".foo3"
</item>
```

home_dir_localization_files_group_check
Esta función incorporada verifica si un archivo de localización que está en el directorio principal de un usuario es propiedad del usuario o de la raíz.

Se pueden enumerar uno o más archivos con el token “file”. Sin embargo, si el token “file” falta, la comprobación busca de manera predeterminada los siguientes archivos:

- **.login**
- **.cschrc**
- **.logout**
- **.profile**
- **.bash_profile**
- **.bashrc**
- **.bash_logout**
- **.env**
- **.dtprofile**
- **.dispatch**
- **.emacs**
- **.exrc**

Ejemplo:

```
<item>
 name: "home_dir_localization_files_group_check"
 description: "Check file .foo/.foo2"
 file: ".foo"
 file: ".foo2"
 file: ".foo3"
</item>
```

Contenido de archivo sospechoso
admin_accounts_in_ftpusers

Uso

```
<item>
 name: "admin_accounts_in_ftpusers"
 description: "This check makes sure every account whose UID is below 500 is present in
/etc/ftpusers."
</item>
```

Esta comprobación audita si todas las cuentas admin, los usuarios con un UID inferior a 500, están presentes en **/etc/ftpusers, /etc/ftpd/ftpusers** o **/etc/vsftpd.ftpusers**.

Archivos innecesarios
find_pre-CIS_files

Uso

```
<item>
 name: "find_preCIS_files"
 description: "Find and list all files created by CIS backup script."
 # Globs allowed (? and *)
 (optional) basedir: "<directory>"
 (optional) ignore: "<directory>"
</item>
```

Esta comprobación está confeccionada en función de un requisito específico del Center for Internet Security (CIS) para aprobar la certificación correspondiente a la referencia del CIS para Red Hat. La comprobación resulta especialmente útil para quienes hayan configurado o protegido un sistema Red Hat de acuerdo con la referencia para Red Hat del CIS. La herramienta de referencia del CIS brinda una secuencia de comandos de respaldo para efectuar una copia de seguridad de todos los archivos del sistema que pueden modificarse durante el proceso de protección del sistema. A estos archivos se les colocará un sufijo con la palabra clave "**-preCIS**". Los archivos se deben eliminar una vez que se hayan aplicado satisfactoriamente todas las recomendaciones de la referencia y que el sistema haya recuperado su condición de funcionamiento. Esta comprobación garantiza que no haya archivos "**preCIS**" en el sistema remoto.

De manera predeterminada, la búsqueda se efectúa de forma recursiva en el directorio "/". Esto puede hacer que la comprobación sea extremadamente lenta al ejecutarse, de acuerdo con la cantidad de archivos que haya en el sistema remoto. Sin embargo, de ser necesario se puede cambiar el directorio base predeterminado en el que se llevará a cabo la búsqueda mediante la palabra clave opcional **basedir**. También es posible omitir ciertos archivos dentro del directorio base para que no se realice la búsqueda en ellos, mediante otra palabra clave opcional: **ignore**.

Dada la naturaleza de la comprobación, es normal que siga ejecutándose por un par de horas, de acuerdo con el tipo de sistema que se esté analizando. Se ha establecido un valor de tiempo de espera predeterminado de cinco horas, que representa el tiempo después del cual Nessus dejará de procesar resultados para esta comprobación. Este valor no se puede cambiar.

Condiciones
Es posible definir la lógica **if/then/else** en la directiva de Unix. Esto permite que el usuario final use un único archivo que puede manejar varias configuraciones. Por ejemplo, el mismo archivo de directiva puede comprobar la configuración de Postfix y Sendmail mediante la sintaxis **if/then/else** correcta.

La sintaxis para establecer condiciones es la siguiente:

```
<if>
 <condition type: "or">
  <Insert your audit here>
 </condition>
 <then>
  <Insert your audit here>
 </then>
 <else>
  <Insert your audit here>
 </else>
</if>
```

Ejemplo:

```
<if>
 <condition type: "or">
  <custom_item>
   type: FILE_CHECK
   description: "Make sure /etc/passwd contains root"
   file: "/etc/passwd"
   owner: "root"
  </custom_item>
 </condition>

 <then>
  <custom_item>
   type: FILE_CONTENT_CHECK
   description: "Make sure /etc/passwd contains root (then)"
   file: "/etc/passwd"
   regex: "^root"
   expect: "^root"
  </custom_item>
 </then>

 <else>
  <custom_item>
   type: FILE_CONTENT_CHECK
   description: "Make sure /etc/passwd contains root (else)"
   file: "/etc/passwd"
   regex: "^root"
   expect: "^root"
  </custom_item>
 </else>
</if>
```

Ya sea que la condición sea errónea o se apruebe, eso nunca aparecerá en el informe, ya que se trata de una comprobación “silent” (silenciosa).

Las condiciones pueden ser del tipo “**and**” u “**or**”.

NetApp Data ONTAP

Esta sección describe el formato y las funciones de las comprobaciones de compatibilidad de los sistemas de almacenamiento que ejecutan NetApp Data ONTAP y la fundamentación que subyace en cada opción.

Uso de comillas:

Las comillas simples y dobles son indistintas al encerrar campos de auditoría, a excepción de los siguientes dos casos:

1. En comprobaciones de compatibilidad de Windows en las que se deban interpretar de forma literal los campos especiales como CRLF, etc., se deben usar comillas simples. Se deben escapar los campos incrustados que se interpretarán como cadenas.

Por ejemplo:

```
expect: 'First line\r\nSecond line\r\nJohn\'s Line'
```

2. Se requieren comillas dobles al utilizar “include_paths” y “exclude_paths” de WindowsFiles.

Si en cualquier tipo de campo (descripción, value_data, regex, etc.) se usan cadenas que contengan comillas simples o dobles, existen dos formas de tratarlas:

a. Use el tipo de comilla opuesto para las comillas de cierre exteriores.

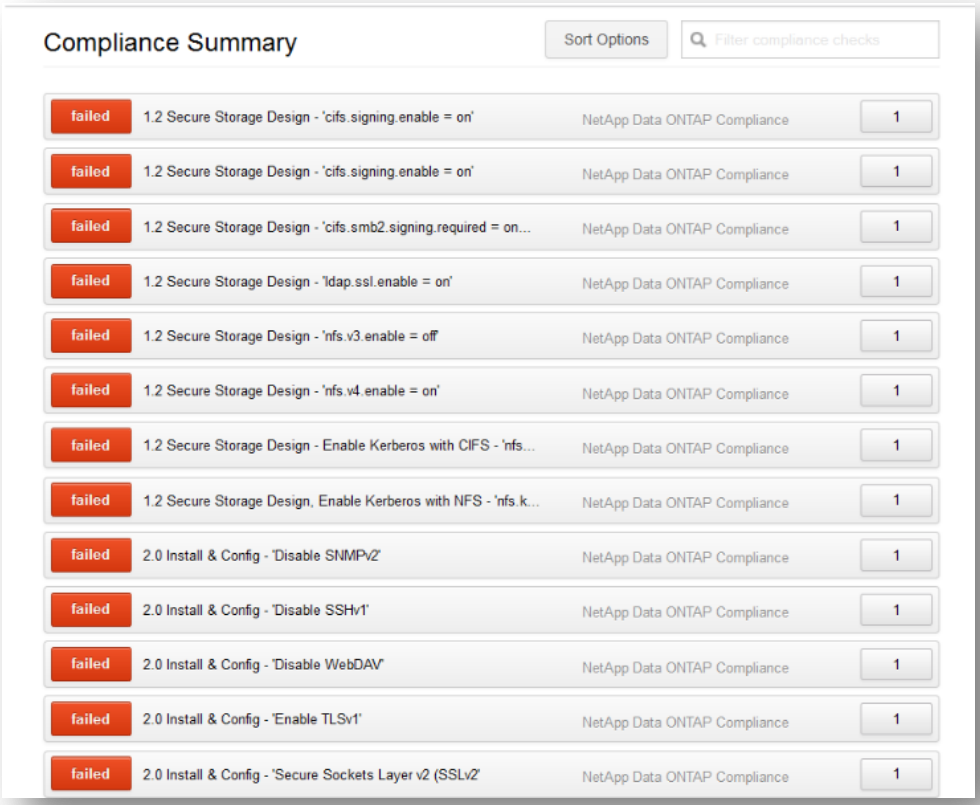
Por ejemplo:

```
expect: "This is John's Line"
expect: 'We are looking for a double-quote-".*'
```

b. Escape las comillas incrustadas, si las hay, con una barra inversa (solo comillas dobles).

Por ejemplo:

```
expect: "\"Text to be searched\""
```



Privilegios de usuario necesarios

Para llevar a cabo un análisis de compatibilidad con éxito en un sistema NetApp Data ONTAP, los usuarios autenticados deben tener los siguientes privilegios:

Credenciales `root` para el archivador de NetApp Data ONTAP

Además de los privilegios anteriores, se requiere una directiva de auditoría para las comprobaciones de compatibilidad de NetApp Data ONTAP y el plugin de Nessus ID #66934 (NetApp Data ONTAP Compliance Checks).

Para realizar un análisis en el dispositivo, comience creando la directiva de la auditoría. Luego, utilice el menú "**SSH settings**" (Configuración de SSH) en la ficha "**Credentials**" (Credenciales) de la directiva para suministrar credenciales `root`. En la ficha "**Plugins**" de la directiva, escoja la familia de plugins "Policy Compliance" (Compatibilidad de directiva) y habilite el plugin ID #66934 llamado "**NetApp Data ONTAP Compliance Checks**" (Comprobaciones de compatibilidad de NetApp Data ONTAP). A continuación, en la ficha "**Preferences**" (Preferencias), escoja la lista desplegable "NetApp Data ONTAP Compliance Checks" (Comprobaciones de compatibilidad de NetApp Data ONTAP) y agregue el archivo **`.audit`** de NetApp del Tenable Support Portal (Portal de soporte de Tenable). Por último, guarde la directiva y ejecute el análisis.

En el caso de que no tenga la opción de suministrar credenciales `root`, puede crearse una cuenta con menos privilegios para realizar la auditoría:

1. Cree un nuevo rol (e.g., `nessus_audit`):

   ```
   # role add nessus_audit -a login-ssh,cli-version,cli-options,cli-uptime
   ```

2. Asigne el rol a un grupo (por ejemplo, `nessus_admins`):

   ```
   # group add nessus_admins -r nessus_audit
   ```

3. Asigne el grupo a un usuario:

```
# useradmin user add nessus -g nessus_admins
```

Tipo de comprobación: CONFIG_CHECK

Las comprobaciones de compatibilidad de Check Point están entre corchetes en encapsulación de **custom_item** y CONFIG_CHECK. Se tratan como cualquier otro archivo **.audit** y funcionan con sistemas que ejecutan el sistema NetApp Data ONTAP. La comprobación CONFIG_CHECK está compuesta por dos o más palabras clave. Las palabras clave **type** y **description** son obligatorias, seguidas de una o más palabras clave. La comprobación funciona auditando la salida del comando “options”.

Palabras clave

La siguiente tabla indica la forma en que se puede usar cada palabra clave en las comprobaciones de compatibilidad de NetApp Data ONTAP:

Palabra clave	Ejemplo de uso y configuración admitida
type	“CHECK_CONFIG” determina si el elemento de configuración especificado existe en los resultados “show configuration” (mostrar configuración) de NetApp Data ONTAP.
description	La palabra clave “**description**” proporciona la capacidad de añadir una breve descripción de la comprobación que se lleva a cabo. Se recomienda enfáticamente que el campo **description** sea exclusivo y que no haya comprobaciones diferentes que tengan el mismo campo **description**. El SecurityCenter de Tenable usa este campo para generar de manera automática un número de identificación de plugin exclusivo en función del campo **description**. Ejemplo: `description: "1.0 Require strong Password Controls - 'min-password-length >= 8'"`
info	La palabra clave “**info**” se usa para agregar una descripción más detallada a la comprobación que se lleva a cabo. La fundamentación para la comprobación podría ser una reglamentación, una dirección URL con más información, una directiva corporativa. etc. Se pueden añadir varios campos **info** en líneas independientes para que el texto adquiera formato de párrafo. No existe un límite preestablecido en cuanto a la cantidad de campos **info** que se pueden usar.  Cada etiqueta “**info**” debe escribirse en una línea independiente, sin saltos de línea. Si se requiere más de una línea (por ejemplo, por motivos de formato), añada etiquetas “**info**” adicionales. Ejemplo: `info: "Enable palindrome-check on passwords"`
severity	La palabra clave “**severity**” especifica la gravedad de la comprobación que se está realizando. Ejemplo: `severity: MEDIUM` La gravedad puede ser HIGH (ALTA), MEDIUM (MEDIA) o LOW (BAJA).

regex	La palabra clave “**regex**” posibilita la búsqueda de la opción de elemento de configuración que coincida con una expresión regular en particular. Ejemplo: `regex: "set snmp .+"` Los siguientes metacaracteres requieren tratamiento especial: + \ * ( ) ^ Escape doblemente estos caracteres con dos barras invertidas “\\”, o enciérrelos entre corchetes cuadrados “[]” si desea que se interpreten de forma literal. Otros caracteres como los siguientes necesitan solo una barra invertida para que se interpreten literalmente: . ? " ' Esto se relaciona con la forma en que el compilador trata estos caracteres. Si una comprobación tiene una etiqueta “**regex**” establecida, pero no hay ninguna etiqueta “**expect**”, “**not_expect**” o “**number_of_lines**” establecida, la comprobación solo notifica todas las líneas que coincidan con la regex.

expect	Esta palabra clave permite la auditoría del elemento de configuración que coincide con la etiqueta “**regex**”; si la etiqueta “**regex**” no se utiliza, busca la cadena “**expect**” en toda la configuración.
	
	La comprobación se aprueba siempre y cuando la línea de configuración encontrada por la “**regex**” coincida con la etiqueta “**expect**”; o, en el caso de que “**regex**” no esté establecida, se aprueba si la cadena “**expect**” se encuentra en la configuración.
	
	Ejemplo:
	`regex: "set password-controls complexity"`
	`expect: "set password-controls complexity [1-4]"`
	
	En el caso anterior, la etiqueta “**expect**” garantiza que la complejidad esté establecida en un valor entre 1 y 4.

not_expect	Esta palabra clave permite la búsqueda de los elementos de configuración que no deben estar en la configuración.
	
	Actúa de manera opuesta a “**expect**”. La comprobación se aprueba si la línea de configuración encontrada por la “**regex**” no coincide con la etiqueta “**not_expect**”; o, en el caso de que la etiqueta “**regex**” no esté establecida, se aprueba siempre y cuando la cadena “**not_expect**” no se encuentre en la configuración.
	
	Ejemplo:
	`regex: "set password-controls password-expiration"`
	`not_expect: "set password-controls password-expiration never"`
	
	En el caso anterior, la etiqueta “**not_expect**” garantiza que los controles de contraseña no estén establecidos en “never”.

Ejemplos de CONFIG_CHECK

Estos son ejemplos de uso de CONFIG_CHECK en un dispositivo con NetApp Data ONTAP:

```
<custom_item>
```

```
 type: CONFIG_CHECK
 description: "1.2 Secure Storage Design, Enable Kerberos with NFS -
       'nfs.kerberos.enable = on'"
 info: "NetApp recommends the use of security features in IP storage protocols to
       secure client access"
 solution: "Enable Kerberos with NFS"
 reference: "PCI|2.2.3"
 see_also: "http://media.netapp.com/documents/tr-3649.pdf"
 regex: "nfs.kerberos.enable[\\s\\t]+"
 expect: "nfs.kerberos.enable[\\s\\t]+on"
</custom_item>
```

Condiciones

Es posible definir la lógica **`if`**/**`then`**/**`else`** en la directiva de auditoría de NetApp Data ONTAP. Esto permite que el usuario final use un único archivo que puede manejar varias configuraciones.

La sintaxis para establecer condiciones es la siguiente:

```
<if>
  <condition type:"or">
      < Insert your audit here >
  </condition>
 <then>
      < Insert your audit here >
 </then>
 <else>
      < Insert your audit here >
 </else>
</if>
```

Ejemplo:

```
<if>
 <condition type: "OR">
 <custom_item>
  type: CONFIG_CHECK
  description: "2.6 Install and configure Encrypted Connections to devices - 'telnet'"
  regex: "set net-access telnet"
  expect: "set net-access telnet off"
  info: "Do not use plain-text protocols."
</custom_item>
 </condition>
 <then>
  <report type: "PASSED">
   description: "Telnet is disabled"
  </report>
 </then>
 <else>
  <custom_item>
  type: CONFIG_CHECK
  description: "2.6 Install and configure Encrypted Connections to devices - 'telnet'"
  regex: "set net-access telnet"
  expect: "set net-access telnet off"
```

```
  info: "Do not use plain-text protocols."
</custom_item>
 </else>
</if>
```

La condición nunca aparece en el informe; es decir, sin importar si falla o se aprueba, no aparecerá (es una comprobación “silent” [silenciosa]).

Las condiciones pueden ser del tipo “**and**” u “**or**”.

Informes

Pueden realizarse en <then> o <else> para lograr una condición deseada PASSED/FAILED (APROBÓ/INCORRECTO).

```
<if>
 <condition type: "OR">
 <custom_item>
  type: CONFIG_CHECK
  description: "2.6 Install and configure Encrypted Connections to devices - 'telnet'"
  regex: "set net-access telnet"
  expect: "set net-access telnet off"
  info: "Do not use plain-text protocols."
</custom_item>
 </condition>
 <then>
  <report type: "PASSED">
  description: "Telnet is disabled"
  </report>
 </then>
 <else>
 <report type: "FAILED">
   description: "Telnet is disabled"
  </report>
 </else>
</if>
```

PASSED (APROBÓ), WARNING (ADVERTENCIA) y FAILED (INCORRECTO) son los valores aceptables para "report type" (tipo de informe).

Referencia para archivos de compatibilidad de auditoría de configuración de IBM iseries

Esta sección describe el formato y las funciones de las comprobaciones de compatibilidad de IBM iSeries y la fundamentación que subyace en cada opción.

Uso de comillas:

Las comillas simples y dobles son indistintas al encerrar campos de auditoría, a excepción de los siguientes dos casos:

1. En comprobaciones de compatibilidad de Windows en las que se deban interpretar de forma literal los campos especiales como CRLF, etc., se deben usar comillas simples. Se deben escapar los campos incrustados que se interpretarán como cadenas.

Por ejemplo:
```
expect: 'First line\r\nSecond line\r\nJohn\'s Line'
```

2. Se requieren comillas dobles al utilizar “include_paths” y “exclude_paths” de WindowsFiles.

Si en cualquier tipo de campo (descripción, value_data, regex, etc.) se usan cadenas que contengan comillas simples o dobles, existen dos formas de tratarlas:

a. Use el tipo de comilla opuesto para las comillas de cierre exteriores.

Por ejemplo:
```
expect: "This is John's Line"
expect: 'We are looking for a double-quote-".*'
```

b. Escape las comillas incrustadas, si las hay, con una barra inversa (solo comillas dobles).

Por ejemplo:
```
expect: "\"Text to be searched\""
```

Privilegios de usuario necesarios

Para llevar a cabo un análisis de compatibilidad satisfactorio en un sistema iSeries, los usuarios autenticados deben tener los siguientes privilegios:

1. Un usuario con autoridad (*ALLOBJ) o de auditoría (*AUDIT) puede auditar todos los valores del sistema. Este usuario suele pertenecer a la clase (*SECOFR).
2. Los usuarios de clase (*USER) o (*SYSOPR) pueden auditar la mayoría de los valores, excepto QAUDCTL, QAUDENDACN, QAUDFRCLVL, QAUDLVL, QAUDLVL2 y QCRTOBJAUD.

Si un usuario no tiene privilegios para acceder a un valor, el valor regresado será *NOTAVL.

Tipo de comprobación

Todas las comprobaciones de compatibilidad de IBM iSeries deben estar entre corchetes con la encapsulación **`check_type`** y la designación “AS/400”. Esto es obligatorio para diferenciar los archivos **`.audit`** diseñados específicamente para sistemas que usan el sistema IBM iSeries, de otros tipos de auditorías de compatibilidad.

Ejemplo:

```
<check_type:"AS/400">
```

A diferencia de otros tipos de auditorías de compatibilidad, no se encuentran disponibles palabras clave de versión o tipo adicionales.

Palabras clave

La siguiente tabla indica la forma en que se puede usar cada palabra clave en las comprobaciones de compatibilidad de IBM iSeries:

Palabra clave	Ejemplo de uso y configuración admitida
`type`	`AUDIT_SYSTEMVAL` `SHOW_SYSTEMVAL`
`systemvalue`	Esta palabra clave se utiliza para especificar un valor particular a comprobarse dentro del sistema IBM iSeries.

	Ejemplo: `systemvalue: "QALWUSRDMN"`
description	Esta palabra clave proporciona la capacidad de añadir una breve descripción de la comprobación que se lleva a cabo. Se recomienda enfáticamente que el campo **description** sea exclusivo y que no haya comprobaciones diferentes que tengan el mismo campo de descripción. El SecurityCenter de Tenable usa este campo para generar de manera automática un número de identificación de plugin exclusivo en función del campo **description**. Ejemplo: `description: "Allow User Domain Objects (QALWUSRDMN) - '*all'"`

value_type	Esta palabra clave se utiliza para definir el tipo de valor (“POLICY_DWORD” o “POLICY_TEXT”) que se comprueba en el sistema IBM iSeries.
	
	Ejemplo:
	`value_type: "POLICY_DWORD"`
	
	Ejemplo:
	`value_type: "POLICY_TEXT"`

value_data	Esta palabra clave define el valor de datos esperado por un valor de sistema. Ejemplo: `value_type: "^([6-9]|[1-9][0-9]+)$"`

check_type	Esta palabra clave define el tipo de comprobación que se utiliza para un valor de datos.
	
	Ejemplos:
	`check_type: "CHECK_EQUAL"`
	`check_type: "CHECK_NOT_EQUAL"`
	`check_type: "CHECK_GREATER_THAN"`
	`check_type: "CHECK_GREATER_THAN_OR_EQUAL"`
	`check_type: "CHECK_LESS_THAN"`
	`check_type: "CHECK_LESS_THAN_OR_EQUAL"`
	`check_type: "CHECK_REGEX"`
	
	Ejemplo:
	`<custom_item>`
	` type: AUDIT_SYSTEMVAL`
	` systemvalue: "QUSEADPAUT"`
	` description: "Use Adopted Authority (QUSEADPAUT) - '!= *none'"`
	` value_type: POLICY_TEXT`
	` value_data: "*none"`
	` check_type: CHECK_NOT_EQUAL`
	`</custom_item>`

info	Esta palabra clave se usa para añadir una descripción más detallada a la comprobación que se está llevando a cabo, tal como una reglamentación, una dirección URL, una directiva corporativa, o bien otro motivo por el que la opción sea necesaria. Se pueden añadir varios campos **info** en líneas independientes para que el texto adquiera formato de párrafo. No existe un límite preestablecido en cuanto a la cantidad de campos **info** que se pueden usar.

	Ejemplo: `info: "\nref : http://publib.boulder.ibm.com/infocenter/iseries/v5r4/topic/books/sc415302.pdf pg. 21"`

Elementos personalizados

Un elemento personalizado constituye una comprobación completa establecida en función de las palabras clave definidas anteriormente. La siguiente es una lista de tipos de elementos personalizados que se encuentran disponibles. Cada comprobación comienza con una etiqueta "**<custom_item>**" y finaliza con "**</custom_item>**". Dentro de las etiquetas se encuentran las listas de una o más palabras clave que son interpretadas por el analizador sintáctico de comprobaciones de compatibilidad para llevar a cabo dichas comprobaciones.

Las comprobaciones de auditoría personalizadas pueden usar "**</custom_item>**" y "**</item>**" de manera indistinta para la etiqueta de cierre.

AUDIT_SYSTEMVAL

"AUDIT_SYSTEMVALUE" audita el valor del parámetro de configuración identificado por la palabra clave "**systemvalue**". El tipo de comparación con el valor auditado se especifica por la palabra clave "**check_type**".

```
<custom_item>
 type: AUDIT_SYSTEMVAL
 systemvalue: "QALWUSRDMN"
 description: "Allow User Domain Objects (QALWUSRDMN) - '*all'"
 value_type: POLICY_TEXT
 value_data: "*all"
 info: "\nref :
       http://publib.boulder.ibm.com/infocenter/iseries/v5r4/topic/books/sc415302.pdf
       pg. 21"
</custom_item>
```

SHOW_SYSTEMVAL

La auditoría"SHOW_SYSTEMVAL" solo notifica el valor del parámetro de configuración identificado por la palabra clave "**systemvalue**".

```
<custom_item>
 type: SHOW_SYSTEMVAL
 systemvalue: "QAUDCTL"
 description: "show QAUDCTL value"
 severity: MEDIUM
</custom_item>
```

Condiciones

Es posible definir la lógica **if/then/else** en la directiva de IBM iSeries. Esto le permite al usuario final devolver un mensaje de advertencia en lugar de una aprobación o error en caso de que la auditoría sea aprobada.

La sintaxis para establecer condiciones es la siguiente:

```
<if>
 <condition type: "or">
```

```
  <Insert your audit here>
 </condition>
 <then>
  <Insert your audit here>
 </then>
 <else>
  <Insert your audit here>
 </else>
</if>
```

Ejemplo:

```
<if>
 <condition type: "or">
  <custom_item>
   type: AUDIT_SYSTEMVAL
   systemvalue: "QDSPSGNINF"
   description: "Sign-on information is displayed (QDSPSGNINF)"
   info: "\nref :
       http://publib.boulder.ibm.com/infocenter/iseries/v5r4/topic/books/sc415302.pdf
       pg. 23"
   value_type: POLICY_DWORD
   value_data: "1"
  </custom_item>
 </condition>

 <then>
  <custom_item>
   type: AUDIT_SYSTEMVAL
   systemvalue: "QDSPSGNINF"
   description: "Sign-on information is not displayed (QDSPSGNINF)"
   info: "\nref :
       http://publib.boulder.ibm.com/infocenter/iseries/v5r4/topic/books/sc415302.pdf
       pg. 23"
   value_type: POLICY_DWORD
   value_data: "1"
  </custom_item>
 </then>

 <else>
  <report type: "WARNING">
   description: "Sign-on information is displayed (QDSPSGNINF)"
   info: ""\nref :
       http://publib.boulder.ibm.com/infocenter/iseries/v5r4/topic/books/sc415302.pdf
       pg. 23"
   info: "Check system policy to confirm requirements."
  </report>
 </else>
</if>
```

Ya sea que la condición sea errónea o se apruebe, eso nunca aparecerá en el informe, ya que se trata de una comprobación “silent” (silenciosa).

Las condiciones pueden ser del tipo “**and**” u “**or**”.

Referencia para archivos de compatibilidad de auditoría de configuración de vCenter/ESXi deVMware

Esta sección describe el formato y las funciones de las comprobaciones de compatibilidad de vCenter y ESXi de Vmware y la fundamentación que subyace en cada opción.

Nessus tiene la capacidad de auditar VMware a través de las API nativas extrayendo la configuración y luego llevando a cabo la auditoría según las comprobaciones incluidas en el archivo **`.audit`** asociado.

Requisitos

Para llevar a cabo un análisis de compatibilidad satisfactorio en sistemas VMware, los usuarios deben tener lo siguiente:

1. Credenciales de administrador para el vCenter o ESXi de VMware (Tenable ha desarrollado API tanto para ESXi [la interfaz gratuita disponible para administrar VM en ESX/ESXi] como para vCenter [un complemento de VMware disponible a determinado precio para administrar uno o más servidores ESX/ESXi]. Este plugin puede aprovechar las credenciales de ESXi o vCenter para hacer esta tarea).

2. Directiva de auditoría para comprobaciones de compatibilidad de vCenter/ESXi de VMware.

3. ID de plugin #64455 (Comprobaciones de compatibilidad de vCenter/ESXi de VMware).

Versiones admitidas

Actualmente, Nessus puede auditar ESXi 4.x y 5.x, además de vCenter 4.x y 5.

Tipo de comprobación

La sintaxis para la función **`.audit`** en VMware depende mucho de los XPATH y XSL Transforms para ejecutar la función.

La auditoría de VMware admite tres tipos de comprobaciones:

AUDIT_VM

Este tipo de comprobación le permite auditar configuración de máquinas virtuales (consulte el Apéndice C para obtener más información):

```
<custom_item>
 type: AUDIT_VM
 description: "VM Setting - 'vmsafe.enable = False'"
 xsl_stmt: "<xsl:template match=\"audit:returnval\">"
 xsl_stmt: "<xsl:value-of
       select=\"audit:propSet/audit:val[@xsi:type='VirtualMachineConfigInfo']/audit:na
       me\"/> : vmsafe.enable : <xsl:value-of
       select=\"audit:propSet/audit:val[@xsi:type='VirtualMachineConfigInfo']/audit:ex
       traConfig[audit:key[text()='vmsafe.enable']]/audit:value\"/>."
 xsl_stmt: "</xsl:template>"
 expect: "vmsafe.enable : 0"
</custom_item>
```

AUDIT_ESX

Este tipo de comprobación le permite auditar la configuración del servidor ESX/ESXi:

```
<custom_item>
 type: AUDIT_ESX
 description : "ESX/ESXi Setting - Syslog.global.logDir"
 xsl_stmt: "<xsl:template match=\"audit:returnval\">"
```

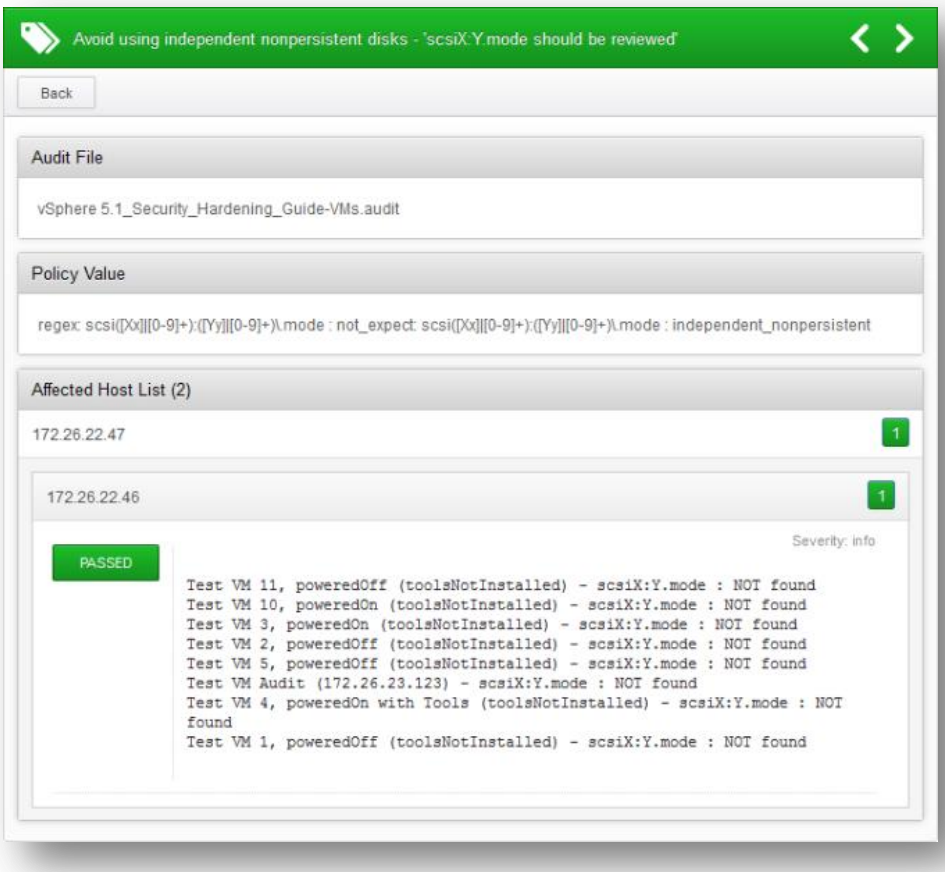
```
 xsl_stmt: "Syslog.global.logDir = <xsl:value-of
       select=\"audit:propSet/audit:val[@xsi:type='HostConfigInfo']/audit:option[audit
       :key[text()='Syslog.global.logDir']]/audit:value\"/>"
 xsl_stmt: "</xsl:template>"
 expect: "Syslog.global.logDir : /foo/bar"
</custom_item>
```

AUDIT_VCENTER

Este tipo de comprobación le permite auditar la configuración de vCenter:

```
<custom_item>
 type: AUDIT_VCENTER
 description: "VMware vCenter Setting - config.vpxd.hostPasswordLength"
 xsl_stmt: "<xsl:template match=\"audit:returnval\">"
 xsl_stmt: "config.vpxd.hostPasswordLength = <xsl:value-of
       select=\"audit:propSet/audit:val[@xsi:type='ArrayOfOptionValue']/audit:OptionVal
       ue[audit:key[text()='config.vpxd.hostPasswordLength']]/audit:value\"/>"
 xsl_stmt: "</xsl:template>"
 expect: "config.vpxd.hostPasswordLength : 30"
</custom_item>
```

Ejemplo de auditoría de vSphere aprobada:



Palabras clave

La siguiente tabla indica la forma en que se puede usar cada palabra clave en las comprobaciones de compatibilidad de VMware:

Palabra clave	Ejemplo de uso y configuración admitida
type	Esta palabra clave describe el tipo de comprobación que un determinado elemento está realizando en un archivo de auditoría. Las auditorías de VMware se pueden realizar con tres tipos de comprobaciones de auditoría: • AUDIT_VM • AUDIT_ESX • AUDIT_VCENTER
description	Esta palabra clave proporciona una breve descripción de la comprobación que se lleva a cabo. Es obligatorio que el campo **description** sea exclusivo y que no haya comprobaciones diferentes que tengan el mismo campo de **description**. Esto es necesario porque el SecurityCenter usa este campo para generar de manera automática un número de identificación de plugin exclusivo en función del campo **description**. Ejemplo: `description: "Disconnect unauthorized devices - 'floppyX.present = false'"`
info	Esta palabra clave se usa para agregar una descripción más detallada a la comprobación que se lleva a cabo. Se permiten diversos campos **info** sin un límite predeterminado. El contenido de **info** debe estar entre comillas dobles. Ejemplo: `info: "Make sure floppy drive is not attached"`
regex	Esta palabra clave permite buscar elementos que coincidan con una expresión regex en particular. Ejemplo: `regex: "floppy([Xx]|[0-9]+)\\.present :"`

La compatibilidad de una comprobación puede determinarse comparando el resultado de la comprobación con las palabras clave **expect** o **not_expect**. No puede usar más de una etiqueta de prueba de compatibilidad en una comprobación determinada.

Palabra clave	Ejemplo de uso y configuración admitida
expect	Esta palabra clave permite la auditoría del elemento de configuración que coincide con la palabra clave "**regex**"; o, si la palabra clave "**regex**" no se utiliza, busca la cadena "**expect**" en toda la configuración. La comprobación se aprueba siempre y cuando la línea de configuración encontrada por la "**regex**" coincida con la cadena "**expect**"; o, en el caso de que "**regex**" no esté establecida, se aprueba si la cadena "**expect**" se encuentra en la configuración. Ejemplo: `regex: "floppy([Xx]|[0-9]+)\\.present :"`

	`expect: floppy([Xx]|[0-9]+)\\.present : false"` O bien: `expect: floppy([Xx]|[0-9]+)\\.present : false"` En los casos anteriores, la palabra clave **expect** garantiza que la unidad de disquete no está presente.

not_expect	Esta palabra clave permite la búsqueda de los elementos de configuración que no deben estar en la configuración.
	
	Actúa de manera opuesta a "**expect**". La comprobación se aprueba si la línea de configuración encontrada por la "**regex**" no coincide con la cadena "**not_expect**"; o, en el caso de que la palabra clave "**regex**" no esté establecida, se aprueba siempre y cuando la cadena "**not_expect**" no se encuentre en la configuración.
	
	Ejemplo:
	
	`regex: floppy([Xx]|[0-9]+)\\.present : "`
	`not_expect: floppy([Xx]|[0-9]+)\\.present : false"`
	
	O bien:
	
	`not_expect: floppy([Xx]|[0-9]+)\\.present : false"`
	
	En los casos anteriores, la palabra clave **expect** garantiza que la unidad de disquete no está presente.

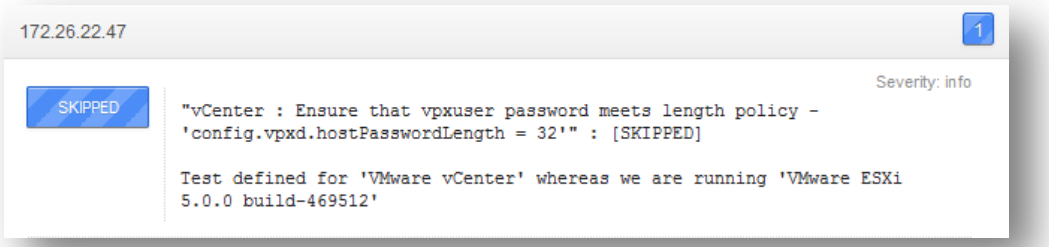
Notas adicionales

Si una comprobación es aprobada, este plugin notifica a todas las VM que coincidieron con la directiva. La auditoría provista por Tenable informará tanto el nombre de la VM como la IP del destino. Sin embargo, tenga en cuenta que la dirección IP de una VM no está disponible a menos que se hayan instalado las herramientas de VMware.

Así se verán los informes:

```
Test VM 2, poweredOff (toolsNotInstalled) - vmsafe.enable : NOT found
Test VM Audit (172.26.23.123) - vmsafe.enable : NOT found
```

Tanto ESX/ESXi como vCenter pueden analizarse con la misma directiva. Sin embargo, tenga en cuenta que las comprobaciones de vCenter ejecutadas en hosts de ESX/ESXi se saltearán.



Para obtener más información

Tenable ha producido una variedad de otros documentos en los que se detallan la instalación, implementación, configuración, operación del usuario y pruebas generales de Nessus:

- **Nessus 5.2 Installation and Configuration Guide** (Guía de instalación y configuración de Nessus 5.2): instrucciones paso a paso sobre la instalación.
- **Nessus 5.2 User Guide** (Guía del usuario de Nessus 5.2): instrucciones sobre cómo configurar y operar la interfaz de usuario de Nessus.
- **Nessus Credential Checks for Unix and Windows** (Comprobaciones con credenciales de Nessus para Unix y Windows): información sobre cómo llevar a cabo análisis de red autenticados mediante el analizador de vulnerabilidades Nessus.
- **Nessus Compliance Checks** (Comprobaciones de compatibilidad con Nessus): guía de alto nivel para comprender y ejecutar las comprobaciones de compatibilidad con Nessus y SecurityCenter.
- **Nessus v2 File Format** (Formato de archivos Nessus v2): describe la estructura del formato de archivos `.nessus`, que se introdujo a través de Nessus 3.2 y NessusClient 3.2.
- **Nessus 5.0 REST Protocol Specification** (Especificación del protocolo REST en Nessus 5.0): describe la interfaz y el protocolo REST en Nessus.
- **Nessus 5 and Antivirus** (Nessus 5 y los antivirus): describe cómo interactúan con Nessus varios de los paquetes de software de seguridad más utilizados, y ofrece consejos y soluciones para permitir una mejor coexistencia con el software sin comprometer su seguridad u obstaculizar sus tareas de análisis de vulnerabilidades.
- **Nessus 5 and Mobile Device Scanning** (Nessus 5 y el análisis de dispositivos móviles): describe cómo Nessus se integra con Active Directory de Microsoft y servidores de administración de dispositivos móviles para identificar los dispositivos móviles en uso en la red.
- **Nessus 5.0 and Scanning Virtual Machines** (Nessus 5.0 y el análisis de máquinas virtuales): describe cómo el analizador de vulnerabilidades Nessus de Tenable Network Security puede utilizarse para auditar la configuración de las plataformas virtuales y también el software que se está ejecutando en ellas.
- **Strategic Anti-malware Monitoring with Nessus, PVS, and LCE** (Supervisión estratégica anti-malware con Nessus, PVS y LCE): describe cómo la plataforma USM de Tenable puede detectar una amplia variedad de software malicioso, e identificar y determinar la gravedad de las infecciones de malware.

- **Patch Management Integration** (Integración de administración de revisiones): este documento describe cómo Nessus y SecurityCenter pueden aprovechar credenciales en los sistemas de administración de revisiones IBM TEM, Microsoft WSUS y SCCM, VMware Go y Red Hat Network Satellite para ejecutar auditorías de revisiones en sistemas para los que pueda no haber credenciales disponibles para el analizador Nessus.
- **Real-Time Compliance Monitoring** (Supervisión de compatibilidad en tiempo real): describe el modo en que pueden usarse las soluciones de Tenable para colaborar con el cumplimiento de distintos tipos de normas gubernamentales y financieras.
- **Tenable Products Plugin Families** (Familias de plugins de productos de Tenable): ofrece una descripción y un resumen de las familias de plugins para Nessus, el Log Correlation Engine (Motor de correlación de registros de eventos) y el Passive Vulnerability Scanner (Analizador pasivo de vulnerabilidades).
- **SecurityCenter Administration Guide** (Guía de administración de SecurityCenter)

Estos son otros recursos en línea:

- Foro de debate de Nessus: https://discussions.nessus.org/
- Blog de Tenable: http://www.tenable.com/blog
- Podcast de Tenable: http://www.tenable.com/podcast
- Videos de ejemplos de uso: http://www.youtube.com/user/tenablesecurity
- Canal de Twitter de Tenable: http://twitter.com/tenablesecurity

Puede contactarse con Tenable en support@tenable.como sales@tenable.com, o visitar nuestro sitio web http://www.tenable.com/.

Apéndice A: Ejemplo de archivo de compatibilidad de Unix

Nota: el siguiente archivo, **`tenable_unix_compliance_template.audit`**, se encuentra disponible en el Tenable Support Portal (Portal de soporte de Tenable), situado en https://support.tenable.com/. Este archivo enumera los distintos tipos de comprobaciones de compatibilidad de Unix que se pueden realizar mediante el módulo de compatibilidad de Unix de Tenable. El archivo real puede contar con actualizaciones que no se reflejen aquí.

```
#
# (C) 2008-2010 Tenable Network Security, Inc.
#
# This script is released under the Tenable Subscription License and
# may not be used from within scripts released under another license
# without authorization from Tenable Network Security, Inc.
#
# See the following licenses for details:
#
# http://cgi.tenablesecurity.com/Nessus_3_SLA_and_Subscription_Agreement.pdf
# http://cgi.tenablesecurity.com/Subscription_Agreement.pdf
#
# @PROFESSIONALFEED@
#
# $Revision: 1.11 $
# $Date: 2010/11/04 15:54:36 $
#
# NAME            : Cert UNIX Security Checklist v2.0
#
#
# Description     : This file is used to demonstrate the wide range of
#             checks that can be performed using Tenable's Unix
#             compliance module. It consists of all the currently
#             implemented built-in checks along with examples of all
#             the other Customizable checks. See:
#             https://plugins-customers.nessus.org/support-
center/nessus_compliance_checks.pdf
#                 For more information.
#
#
###################################
#                         #
# File permission related checks #
#                         #
###################################


<check_type:"Unix">

# Example 1.
# File check example with owner and group
# fields set and mode field set in Numeric
# format

<custom_item>
      #system           : "Linux"
      Type              : FILE_CHECK
      Description       : "Permission and ownership check /etc/inetd.conf"
      Info        : "Checking that /etc/inetd.conf has owner/group of root and is mode
'600'"
```

```
      File              : "/etc/inetd.conf"
      Owner             : "root"
      Group             : "root"
      Mode              : "600"
</custom_item>


# Example 2.
# File check example with just owner field set
# and mode set.

<custom_item>
      #system           : "Linux"
      Type              : FILE_CHECK
      description       : "Permission and ownership check /etc/hosts.equiv"
      info        : "Checking that /etc/hosts.equiv is owned by root and mode '500'"
      file              : "/etc/hosts.equiv"
      owner             : "root"
      mode              : "-r-x------"
</custom_item>

# Example 3.
# File check example with just file field set
# starting with "~". This check will search
# and audit the file ".rhosts" in home directories
# of all accounts listed in /etc/passwd.

<custom_item>
      #system           : "Linux"
      Type              : FILE_CHECK
      Description       : "Permission and ownership check ~/.rhosts"
      info        : "Checking that .rhosts in home directories have the specified
ownership/mode"
      file              : "~/.rhosts"
      owner             : "root"
      mode              : "600"
</custom_item>


# Example 4.
# File check example with mode field having
# sticky bit set. Notice the first integer in
# the mode field 1 indicates that sticky bit is
# set. The first integer can be modified to check
# for SUID and SGUID fields. Use the table below
# to determine the first integer field.
#
# 0   000   setuid, setgid, sticky bits are cleared
# 1   001   sticky bit is set
# 2   010   setgid bit is set
# 3   011   setgid and sticky bits are set
# 4   100   setuid bit is set
# 5   101   setuid and sticky bits are set
# 6   110   setuid and setgid bits are set
# 7   111   setuid, setgid, sticky bits are set
```

```
<custom_item>
      #system           : "Linux"
      Type              : FILE_CHECK
      Description       : "Permission and ownership check /var/tmp"
      info        : "Checking that /var/tmp is owned by root and mode '1777'"
      file              : "/var/tmp"
      owner       : "root"
      mode        : "1777"
</custom_item>

# Example 5.
# File check example with mode field having
# sticky bit set in textual form and is owned by root.

<custom_item>
      #system           : "Linux"
      Type              : FILE_CHECK
      Description       : "Permission and ownership check /tmp"
      info        : "Checking that the /tmp mode has the sticky bit set in textual form
and is owned by root"
      file              : "/tmp"
      owner       : "root"
      mode        : "-rwxrwxrwt"
</custom_item>

###################################
#                                 #
# Service/Process related checks #
#                                 #
###################################

# Example 6.
# Process check to audit if fingerd is turned
# OFF on a given host.

<custom_item>
      #system           : "Linux"
      type        : PROCESS_CHECK
      description       : "Check fingerd process status"
      info        : "This check looks for the finger daemon to be 'OFF'"
      name        : "fingerd"
      status            : OFF
</custom_item>

# Example 7.
# Process check to audit if sshd is turned
# ON on a given host.

<custom_item>
      #system           : "Linux"
      type        : PROCESS_CHECK
      description       : "Check sshd process status"
      info        : "This check looks for the ssh daemon to be 'ON'"
      name        : "sshd"
      status            : ON
</custom_item>
```

```
##############################
#                  #
# File Content related checks #
#                  #
##############################

# Example 8
# File content check to audit if file /etc/host.conf
# contains the string described in the regex field.
#

<custom_item>
      #System            : "Linux"
      type          : FILE_CONTENT_CHECK
      description   : "This check reports a problem if the order is not 'order hosts,bind'
in /etc/host.conf"
      file          : "/etc/host.conf"
      search_locations : "/etc"
      regex         : "order hosts,bind"
      expect            : "order hosts,bind"
</custom_item>

# Example 9
# This is a better example of a file content check. It first looks
# for the string ".*LogLevel=.*" and if it matches it checks whether
# it matches .*LogLevel=9. For example, if the file was to have LogLevel=8
# this check will fail since the expected value is set to 9.
#

<custom_item>
      #System                : "Linux"
      type              : FILE_CONTENT_CHECK
      description             : "This check reports a problem when the log level setting
in the sendmail.cf file is less than the value set in your security policy."
      file              : "sendmail.cf"
      search_locations  : "/etc:/etc/mail:/usr/local/etc/mail"
      regex             : ".*LogLevel=.*"
        expect              : ".*LogLevel=9"
</custom_item>

# Example 10
# With compliance checks you can cause the shell to execute a command
# and parse the result to determine compliance. The check below determines
# whether the version of FreeBSD on the remote system is compliant with
# corporate standards. Note that since we determine the system type using
# the "system" tag, the check will skip if the remote OS doesn't match
# the one specified.

<custom_item>
      system            : "FreeBSD"
      type        : CMD_EXEC
      description : "Make sure that we are running FreeBSD 4.9 or higher"
      cmd         : "uname -a"
      expect            : "FreeBSD (4\.(9|[1-9][0-9])|[5-9]\.)"
</custom_item>

##################
```

```
#            #
# Builtin Checks #
#            #
##################

# Checks that are not customizable are built
# into the Unix compliance check module. Given below
# are the list of all the checks are the performed
# using the builtin functions. Please refer to the
# the Unix compliance checks documentation for more
# details about each check.
#


<item>
name: "minimum_password_length"
description : "Minimum password length"
value : "14..MAX"
</item>

<item>
name: "max_password_age"
description : "Maximum password age"
value: "1..90"
</item>

<item>
name: "min_password_age"
description : "Minimum password age"
value: "6..21"
</item>

<item>
name: "accounts_bad_home_permissions"
description : "Account with bad home permissions"
</item>

<item>
name: "accounts_without_home_dir"
description : "Accounts without home directory"
</item>

<item>
name: "invalid_login_shells"
description: "Accounts with invalid login shells"
</item>

<item>
name: "login_shells_with_suid"
description : "Accounts with suid login shells"
</item>

<item>
name: "login_shells_writeable"
description : "Accounts with writeable shells"
</item>
```

```
<item>
name: "login_shells_bad_owner"
description : "Shells with bad owner"
</item>

<item>
name: "passwd_file_consistency"
description : "Check passwd file consistency"
</item>

<item>
name: "passwd_zero_uid"
description : "Check zero UID account in /etc/passwd"
</item>

<item>
name : "passwd_duplicate_uid"
description : "Check duplicate accounts in /etc/passwd"
</item>

<item>
name : "passwd_duplicate_gid"
description : "Check duplicate gid in /etc/passwd"
</item>

<item>
name : "passwd_duplicate_username"
description : "Check duplicate username in /etc/passwd"
</item>

<item>
name : "passwd_duplicate_home"
description : "Check duplicate home in /etc/passwd"
</item>

<item>
name : "passwd_shadowed"
description : "Check every passwd is shadowed in /etc/passwd"
</item>

<item>
name: "passwd_invalid_gid"
description : "Check every GID in /etc/passwd resides in /etc/group"
</item>

<item>
name : "group_file_consistency"
description : "Check /etc/group file consistency"
</item>

<item>
name: "group_zero_gid"
description : "Check zero GUID in /etc/group"
</item>

<item>
name: "group_duplicate_name"
```

```
description : "Check duplicate group names in /etc/group"
</item>

<item>
name: "group_duplicate_gid"
description : "Check duplicate gid in /etc/group"
</item>

<item>
name : "group_duplicate_members"
description : "Check duplicate members in /etc/group"
</item>

<item>
name: "group_nonexistant_users"
description : "Check for nonexistent users in /etc/group"
</item>

</check_type>
```

Apéndice B: Ejemplo de archivo de compatibilidad de Windows

Nota: el siguiente archivo se encuentra disponible en el Tenable Support Portal (Portal de soporte de Tenable), situado en https://support.tenable.com/. El archivo real puede contar con actualizaciones que no se reflejen aquí. El nombre de esta secuencia de comandos en particular es **`financial_microsoft_windows_user_audit_guideline_v2.audit`** y está basada en guías de protección comunes para administración de usuarios. Esta directiva busca una directiva de contraseña razonable y una directiva de bloqueo de cuentas, y garantizar que los eventos de inicio de sesión queden asentados en el registro de eventos de Windows.

```
# (C) 2008 Tenable Network Security
#
# This script is released under the Tenable Subscription License and
# may not be used from within scripts released under another license
# without authorization from Tenable Network Security Inc.
#
# See the following licenses for details:
#
# http://cgi.tenablesecurity.com/Nessus_3_SLA_and_Subscription_Agreement.pdf
# http://cgi.tenablesecurity.com/Subscription_Agreement.pdf
#
# @PROFESSIONALFEED@
#
# $Revision: 1.2 $
# $Date: 2008/10/07 15:48:17 $
#
# Synopsis: This file will be read by compliance_check.nbin
#          to check compliance of a Windows host to
#          typical financial institution audit policy
#

<check_type:"Windows" version:"2">
<group_policy:"User audit guideline">

      <item>
      name: "Enforce password history"
      value: 24
      </item>

      <item>
      name: "Maximum password age"
      value: 90
      </item>

      <item>
      name: "Minimum password age"
      value: 1
      </item>

      <item>
      name: "Minimum password length"
      value: [12..14]
      </item>

      <item>
      name: "Account lockout duration"
      value: [15..30]
      </item>
```

```
      <item>
      name: "Account lockout threshold"
      value: [3..5]
      </item>

      <item>
      name: "Reset lockout account counter after"
      value: [15..30]
      </item>

      <item>
                  name: "Audit account logon events"
                  value: "Success, Failure"
      </item>

      <item>
                  name: "Audit logon events"
                  value: "Success, Failure"
      </item>

</group_policy>
</check_type>
```

Apéndice C: XSL Transform para conversión .audit

Varios plugins de comprobación de compatibilidad dependen de la auditoría de contenido XML, como las comprobaciones de compatibilidad de Palo Alto, VMware y Unix. Para aprovechar mejor estas funciones, es beneficioso aprender a crear XSL Transforms. En algunos casos, construir un XSL Transform requerirá un poco de prueba y error. Una vez que se familiarice con ese proceso, convertirlo en un **.audit** es el próximo paso, que puede no ser intuitivo. Este apéndice les da a los usuarios la orientación adecuada sobre cómo construir y utilizar XSL Transforms personalizados y convertirlos en archivos **.audit**.

Las diferentes comprobaciones de auditoría (por ejemplo, AUDIT_XML, AUDIT_VCENTER, AUDIT_ESX) son independientes y distintas pero usan la misma lógica subyacente. Comprender los conceptos básicos de trabajar con XML le permitirá traducirlas directamente a otras plataformas que utilicen XML.

Al usar la utilidad **xsltproc**, los usuarios pueden seguir 7 pasos a fin de generar archivos **.audit** personalizados para contenido XML.

Paso 1: Instalar xsltproc

Verifique que **xsltproc** esté instalado en su sistema o instálelo si es necesario. Puede verificar si está instalado y funciona escribiendo el comando:

```
[tater@pearl ~]# xsltproc
Usage: xsltproc [options] stylesheet file [file ...]
       Options:
           --version or -V: show the version of libxml and libxslt used
           --verbose or -v: show logs of what's happening
[..]
```

Paso 2: Identificar el archivo XML que se utilizará

Determine qué archivo XML utilizará. Verifique la ubicación del archivo, y que el contenido sea XML. Por ejemplo:

```
[tater@pearl ~]# ls top-applications.xml
-rw-r--r-- 1 tater gpigs 3857 2011-09-08 21:20 top-applications.xml
[tater@pearl ~]# head top-applications.xml
<?xml version="1.0"?>
<report reportname="top-applications" logtype="appstat">
  <result name="Top applications" logtype="appstat" start="2013/01/29 00:00:00" start-
       epoch="1359446400" end="2013/01/29 23:59:59" end-epoch="1359532799" generated-
       at="2013/01/30 02:02:09" generated-at-epoch="1359540129" range="Tuesday,
       January 29, 2013">
     <entry>
[..]
```

Paso 3: Familiarizarse con XSL Transforms y XPath

Este proceso requiere un conocimiento básico de los conceptos XSL Transforms y XPath. Para obtener más información:

- w3schools.com - XSLT – Transformation
- w3schools.com - XPath Introduction

Paso 4: Crear el XSLT Transform

Para el siguiente paso, el objetivo es extraer la información relevante de un archivo XML usando XSL Transforms. Comience creando un XSL Transform, que es necesario para extraer los datos relevantes del archivo. Como ejemplo,

presuponga que necesitamos extraer el elemento “name” de un XML. El siguiente XSLT extraerá la información necesaria:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="text"/>

<xsl:template match="result">
<xsl:for-each select="entry">
+ <xsl:value-of select="name"/>
</xsl:for-each>

</xsl:template>
</xsl:stylesheet>
```

Una vez que el XSLT esté creado, guárdelo en un lugar práctico para la prueba del siguiente paso. Este ejemplo puede guardarse como **`pa.xsl`**.

Cuando utiliza un XSLT personalizado en un .audit, debe ignorar las primeras 3 líneas y las últimas 2 líneas. Estas líneas estándar son agregadas por el plugin de Nessus **`nbin`** durante el procesamiento. En este ejemplo, las líneas 5-8 son las líneas de interés que se deberán utilizar en el elemento `AUDIT_XML` o `AUDIT_REPORTS`.

El proceso de prueba del Paso 5 también puede utilizarse durante la construcción del XSLT para validar suposiciones y/o técnicas nuevas. Este proceso es particularmente útil si el XSLT es nuevo para usted o trabaja en transforms más complejos.

Paso 5: Verificar que el XSLT Transform funcione

Verifique que su XSL Transform funcione con xsltproc. El formato general para la prueba es:

```
/usr/bin/xsltproc {XSLT file} {Source XML}
```

Si toma nombres de archivos de muestra de los pasos anteriores, obtendrá lo siguiente. Esto le informa que el XSL Transform es correcto y está formateado adecuadamente, y que la información que espera está siendo devuelta.

```
[tater@pearl ~]# xsltproc pa.xsl top-applications.xml

+ insufficient-data
+ ping
+ snmp
+ dns
+ lpd
+ ntp
+ time
+ icmp
+ netbios-ns
+ radius
+ source-engine
+ stun
+ rip
+ tftp
+ echo
+ portmapper
+ teredo
+ slp
```

```
+ ssdp
+ dhcp
+ mssql-mon
+ pcanywhere
+ apple-airport
+ ike
+ citrix
+ xdmcp
+ l2tp
```

Paso 6: Copiar el XSLT al .audit

Una vez que el XSL Transform funcione como desea, copie las líneas de interés del XSLT (líneas 5-8 en este ejemplo) a la comprobación **`.audit`**.

```
xsl_stmt: "<xsl:template match=\"result\">"
xsl_stmt: "<xsl:for-each select=\"entry\">"
xsl_stmt: "+ <xsl:value-of select=\"name\"/>"
xsl_stmt: "</xsl:for-each>"
```

Cada línea del XSL transform personalizado debe colocarse en su propio elemento **`xsl_stmt`** entre comillas dobles. Como el elemento **`xslt_stmt`** usa comillas dobles para encapsular los enunciados `<xsl>`, cualquier comilla doble utilizada debe escaparse. **Escapar las comillas dobles es importante, ya que no hacerlo genera riesgo de errores en la ejecución de la comprobación**.

```
/usr/bin/xsltproc {XSLT file} {Source XML}
```

En el siguiente paso puede ver varios ejemplos de comillas dobles escapadas adecuadamente.

Paso 7: Auditoría final

Una vez que llevó a cabo los seis primeros pasos, tendrá todo lo necesario para construir una auditoría:

```
<custom_item>
 type: AUDIT_REPORTS
 description: "Palo Alto Reports - Top Applications"
 request: "&reporttype=predefined&reportname=top-applications"
 xsl stmt: "<xsl:template match=\"result\">"
 xsl_stmt: "<xsl:for-each select=\"entry\">"
 xsl_stmt: "+ <xsl:value-of select=\"name\"/>"
 xsl_stmt: "</xsl:for-each>"
</custom_item>
```

Acerca de Tenable Network Security

Más de 20 000 organizaciones confían en Tenable Network Security, entre ellas el Departamento de Defensa de EE. UU. en su totalidad y muchas de las compañías más grandes y los gobiernos de todo el mundo, para adelantarse a las vulnerabilidades, amenazas y riesgos de compatibilidad emergentes. Sus soluciones Nessus y SecurityCenter siguen marcando la norma para identificar vulnerabilidades, evitar ataques y cumplir con muchísimos requisitos regulatorios. Para obtener más información, visite www.tenable.com.

SEDE CENTRAL MUNDIAL

Tenable Network Security
7021 Columbia Gateway Drive
Suite 500
Columbia, MD 21046 – EE. UU.
410.872.0555
www.tenable.com