Monitoreo y Alertas de Seguridad en WordPress con Wazuh

ANÁLISIS DE REGISTROS WEB EN WORDPRESS

Los registros de acceso web en WordPress contienen información valiosa sobre las interacciones de los usuarios, acciones administrativas y posibles intentos de intrusión. El análisis detallado de estos logs permite identificar patrones de comportamiento legítimo, así como actividades, subidas de archivos no autorizados o explotación de vulnerabilidades en endpoints críticos.

Para ello desarrollamos una guía práctica utilizando Wazuh Manager como SIEM, para la creación de reglas personalizadas de detección, basadas en eventos locales y ataques previamente identificados en el entorno. Estas reglas permitirán:

- Detección de intentos de intrusión (fuerza bruta, subidas de archivos maliciosos, comportamiento anómalo).
- Monitoreo de accesos sospechosos a terminales desde el panel de administración (wp-admin).
- Generación de alertas tempranas ante comportamientos anómalos (Solicitudes AJAX múltiples desde la misma IP, modificaciones no autorizadas en la biblioteca de medios).

A partir de casos reales, se almacenaron distintos tipos de peticiones analizadas múltiples servidores, se establecerán criterios para identificar amenazas. El objetivo final es fortalecer la seguridad del sitio WordPress mediante un sistema de monitoreo proactivo, adaptado a necesidades específicas.

Puedes consultar la guía de instalación de Wazuh en el siguiente enlace: Instalación de Wazuh Open Source Security – Base de Conocimiento CGII

Escaneo de Vulnerabilidades en WordPress Usando Wazuh

Configuración de WP-CLI

Para que Wazuh pueda interactuar con WordPress y realizar la comprobación de seguridad, es necesario instalar WP-CLI en el servidor. Sigue estos pasos: Descargar e instalar WP-CLI:

```
curl -O <https://raw.githubusercontent.com/wp-cli/builds/gh-pages/phar/wp-cli.phar>
php wp-cli.phar --info
chmod +x wp-cli.phar
sudo mv wp-cli.phar /usr/local/bin/wp
```

Creación de Políticas de Seguridad para WordPress

La comprobación de seguridad en WordPress se basa en la aplicación de políticas definidas en un archivo YAML. A continuación, se describe cómo crearlas e implementarlas en Wazuh.

```
sudo su
mkdir /home/local_sca_policies/
touch /home/local_sca_policies/custom_wordpress_policy.yml
nano /home/local_sca_policies/custom_wordpress_policy.yml
```

Creamos nuestras propias reglas/políticas que debe cumplir Wazuh, definiendo cada una en el archivo `custom_wordpress_policy.yml`.

```
# Security Configuration Assessment
# Hardening policies for WordPress installations

policy:
  id: "wordpress_assessment"
  file: "custom_wordpress_policy.yml"
  name: "Comprobación de configuración de seguridad para instalaciones de WordPress."
  description: "Guía para establecer una configuración segura en instalaciones de WordPress."
  references:
    - https://wordpress.org/support/article/hardening-wordpress/
    - https://wpsecuritychecklist.org/items/

requirements:
  title: "Verificar que el endpoint sea un host de WordPress y que tenga instalada la herramienta wp-cli."
  description: "Requisitos para ejecutar el escaneo de SCA contra la política de configuración de WordPress."
  condition: all
  rules:
```

```
    - 'f:$wp_install_dir/wp-config.php'
    - 'c:wp --info --allow-root -> r:^WP\pCLI\sversion\p'

variables:
  $wp_install_dir: /var/www/html # Example: /var/www/html
  $wp_host:  http://172.19.0.5:8000/ # Examples: https://example.com
  $wp_user: admin # Example: admin
  $wp_security_plugin: wordfence # Example: wordfence

checks:
```

En `checks`, agregamos las siguientes líneas, que explicaremos a continuación:

Regla 1. ID=100000

- **Propósito:** Evitar exploits en versiones antiguas con vulnerabilidades conocidas (CVE).
- **Cuándo actúa:** Cuando la versión instalada es inferior a la última estable publicada en wordpress.org.
- **Acción:** Actualizar WordPress a la última versión estable disponible.

```
  - id: 100000
    title: "Fortalecimiento de WordPress: Asegurar que la versión de WordPress esté actualizada."
    description: "La versión instalada de WordPress debe ser la última versión disponible en https://wordpress.org/download/."
    rationale: "Pueden descubrirse nuevas vulnerabilidades en WordPress. Es importante actualizar a la última versión para evitar que se exploten vulnerabilidades descubiertas en versiones antiguas."
    remediation: "Actualizar WordPress a la última versión."
    condition: all
    rules:
      - c:runuser -l $wp_user -c "wp core check-update --path=$wp_install_dir" -> r:WordPress\sis\sat\sthe\slatest\sversion
```

Regla 2. ID=100001

- **Propósito:** Establecer permisos adecuados en el archivo `.htaccess`.
- **Condición:** Cuando los permisos del archivo `.htaccess` no son 644.
- **Acción:** Establecer los permisos del archivo `.htaccess` a 644.

```
- id: 100001
   title: "Fortalecimiento de WordPress: Asegurar que los permisos del archivo .htaccess estén configurados en 644."
   description: "Esta política verifica los permisos del archivo .htaccess en el directorio raíz de la instalación de WordPress."
   rationale: "Usuarios no autorizados podrían leer el archivo .htaccess si los permisos no son lo suficientemente estrictos. Además, permisos demasiado restrictivos pueden causar errores al cargar un sitio de WordPress."
   remediation: "Establecer los permisos del archivo en 644 ejecutando el comando chmod 644 $wp_install_dir/.htaccess"
   condition: all
   rules:
     - c:stat -c '%a' $wp_install_dir/.htaccess -> r:644
```

Regla 3. ID=100002

- **Propósito**: Desactivar la depuración en WordPress.
- **Condición**: Si la constante WP_DEBUG en wp-config.php está configurada como true.
- **Acción**: Configurar la constante WP_DEBUG como false en el archivo wp-config.php.

```
 - id: 100002
   title: "Fortalecimiento de WordPress: Asegurar que la depuración de WordPress esté desactivada."
   description: "Esta política verifica si la depuración de WordPress está habilitada en el archivo wp-config.php."
   rationale: "Cuando WP_DEBUG está habilitado, se muestra información detallada sobre errores en las páginas del sitio web. Esto puede incluir información sobre errores, funciones obsoletas y código vulnerable que puede ser explotado por actores maliciosos."
   remediation: "Desactivar la depuración de WordPress estableciendo la variable WP_DEBUG en wp-config.php en false."
   condition: none
   rules:
     - c:runuser -l $wp_user -c "wp config list WP_DEBUG --path=$wp_install_dir" -> r:true|1
```

Regla 4. ID=100003

- **Propósito**: Eliminar archivos de respaldo no necesarios.
- **Condición**: Si existen archivos .zip, .bak, .old, etc., en el directorio raíz.
- **Acción**: Eliminar archivos de respaldo no necesarios del directorio raíz.

```
 - id: 100003
   title: "Fortalecimiento de WordPress: Asegurar que no haya archivos de respaldo (.zip, .back, .bac, .old) en el directorio raíz de la instalación de WordPress."
```

```
    description: "Esta política verifica si hay archivos de respaldo o comprimidos del sitio web o plugins en el directorio raíz de la instalación de WordPress."
    rationale: "Se pueden crear archivos de respaldo de algunos archivos de configuración sensibles, como wp-config.php.bak, antes de editar la configuración en vivo. Dado que estos archivos ya no terminan en .php, no son procesados por el motor de PHP y pueden ser leídos por cualquiera. Esto puede llevar a la divulgación de información sensible."
    remediation: "Realizar una limpieza de medios para eliminar bases de datos, archivos antiguos y de respaldo. Además, dejar solo los archivos necesarios en el directorio raíz de la instalación de WordPress."
    condition: none
    rules:
      - c:sh -c "cd $wp_install_dir; ls -la" -> r:.zip|.back|.backup|.bak|.old|.previous|.sql
```

Regla 5. ID=100004

- **Propósito**: Asegurar la seguridad de las cuentas administrativas.
- **Condición**: Si hay cuentas administrativas con nombres comunes (admin, webmaster, etc.).
- **Acción**: Renombrar las cuentas administrativas que utilizan nombres comunes por otros menos predecibles.

```
  - id: 100004
    title: "Fortalecimiento de WordPress: Asegurar que no se utilicen nombres de cuentas administrativas comunes."
    description: "Esta política verifica si se utilizan nombres de cuentas administrativas comunes (por ejemplo, admin, administrator, webmaster)."
    rationale: "El uso de nombres de cuentas administrativas comunes aumenta la probabilidad de un ataque de fuerza bruta exitoso."
    remediation: "Renombrar todas las cuentas administrativas predeterminadas y utilizar nombres de cuentas administrativas poco comunes."
    condition: none
    rules:
      - c:runuser -l $wp_user -c "wp user list --field=user_login --path=$wp_install_dir" -> r:admin|administrator|backup|webmaster
```

Regla 6. ID=100005

- **Propósito**: Deshabilitar la navegación de directorios.
- **Condición**: Si no está configurada la directiva Options -Indexes en el archivo .htaccess.
- **Acción**: Agregar la directiva Options -Indexes en el archivo .htaccess.

```
- id: 100005
  title: "Fortalecimiento de WordPress: Asegurar que la navegación de directorios esté deshabilitada."
  description: "Esta política verifica si se puede listar el contenido de directorios sensibles (por ejemplo, wp-includes, wp-admin, wp-content)."
  rationale: "Cuando la navegación de directorios está habilitada en un servidor web, puede llevar a la divulgación de información sensible y permitir el listado del contenido de directorios privilegiados."
  remediation: "Deshabilitar la navegación de directorios agregando 'Options All -Indexes' en el archivo .htaccess de esta instalación de WordPress."
  condition: all
  rules:
    - c:cat $wp_install_dir/.htaccess -> r:Options\sAll\s\.Indexes
```

Regla 7. ID=100006

- **Propósito**: Establecer los permisos de las carpetas principales.
- **Condición**: Si los permisos de las carpetas principales (wp-admin, wp-includes, wp-content) no son 755.
- **Acción**: Establecer los permisos de las carpetas principales a 755.

```
- id: 100006
  title: "Fortalecimiento de WordPress: Asegurar que los permisos de las carpetas de WordPress estén configurados en 755."
  description: "Esta política verifica los permisos de las carpetas en las instalaciones de WordPress."
  rationale: "El uso de permisos incorrectos en las carpetas de una instalación de WordPress puede dejar los archivos en esos directorios expuestos a modificaciones no autorizadas."
  remediation: "Establecer todas las carpetas en el directorio de WordPress en 755 usando el comando chmod."
  condition: all
  rules:
    - c:stat -c '%a' $wp_install_dir/wp-admin -> r:755
    - c:stat -c '%a' $wp_install_dir/wp-includes -> r:755
    - c:stat -c '%a' $wp_install_dir/wp-content -> r:755
    - c:stat -c '%a' $wp_install_dir/wp-content/plugins -> r:755
    - c:stat -c '%a' $wp_install_dir/wp-content/themes -> r:755
```

Regla 8. ID=100007

- **Propósito**: Evitar exploits por temas desactualizados.
- **Condición**: Si hay temas con actualizaciones pendientes.

- **Acción**: Actualizar temas automáticamente.

```
 - id: 100007
   title: "Fortalecimiento de WordPress: Asegurar que no haya plugins desactualizados."
   description: "Esta política verifica que no haya plugins de WordPress desactualizados en esta instalación de WordPress."
   rationale: "Los plugins desactualizados pueden tener vulnerabilidades que actores maliciosos pueden explotar para tomar el control de un sitio de WordPress y, posteriormente, del servidor."
   remediation: "Actualizar todos los plugins de WordPress."
   condition: none
   rules:
     - c:runuser -l $wp_user -c "wp plugin list --field=update --path=$wp_install_dir" -> r:available
```

Regla 9. ID=100008

- **Propósito**: Proteger el sitio con un plugin de seguridad.
- **Condición**: Si el plugin $wp_security_plugin no está activo.
- **Acción**: Activar el plugin de seguridad.

```
 - id: 100008
   title: "Fortalecimiento de WordPress: Asegurar que no haya temas de WordPress desactualizados."
   description: "Esta política verifica que no haya temas de WordPress desactualizados en esta instalación de WordPress."
   rationale: "Los temas desactualizados pueden tener vulnerabilidades que actores maliciosos pueden explotar para tomar el control de un sitio de WordPress y, posteriormente, del servidor."
   remediation: "Actualizar todos los temas de WordPress."
   condition: none
   rules:
     - c:runuser -l $wp_user -c "wp theme list --field=update --path=$wp_install_dir" -> r:available
```

Regla 10. ID=100009

- **Propósito**: Reforzar la seguridad básica del sitio con herramientas especializadas.
- **Condición**: Si el plugin de seguridad definido en `$wp_security_plugin` no está activo.
- **Acción**: Instalar y activar el plugin de seguridad configurado (ej: Wordfence, Sucuri.

```
- id: 100009
   title: "Fortalecimiento de WordPress: Asegurar que un plugin de seguridad esté instalado y activo."
   description: "Esta política verifica si el plugin de seguridad de WordPress especificado por la organización
```

```
($wp_security_plugin) está instalado."
    rationale: "Los plugins de seguridad pueden proporcionar una medida de protección contra exploits comunes dirigidos a sitios web de WordPress, como ataques de fuerza bruta e inyecciones SQL. La presencia de un plugin de seguridad reducirá significativamente la superficie de ataque."
    remediation: "Instalar y activar el plugin de seguridad $wp_security_plugin."
    condition: all
    rules:
      - c:runuser -l $wp_user -c "wp plugin is-active $wp_security_plugin --path=$wp_install_dir; echo $?" -> r:0
```

Regla 11. ID=100010

- **Propósito**: Proteger información sensible en el archivo de configuración
- **Condición**: Si `wp-config.php` es accesible desde el navegador (HTTP 200)
- **Acción**: Configurar el servidor web para denegar acceso a `wp-config.php` (HTTP 403/404).

```
 - id: 100010
    title: "Fortalecimiento de WordPress: Asegurar que el archivo wp-config.php no sea accesible públicamente."
    description: "Esta política verifica si el archivo wp-config.php es accesible desde el navegador."
    rationale: "El archivo wp-config.php contiene credenciales y configuraciones críticas. Si es accesible públicamente, un atacante podría extraer información sensible."
    remediation: "Asegurar que el servidor web bloquea el acceso a wp-config.php mediante reglas en .htaccess o configuración de Nginx."
    condition: all
    rules:
      - c:curl -I $wp_host/wp-config.php -> r:403|404
```

Regla 12. ID=100011

- **Propósito: Eliminar vectores de ataque a través de XML-RPC**
- **Cuándo actúa: Si `xmlrpc.php` responde a solicitudes (no devuelve 403/404).**.
- **Acción:** Bloquear acceso al archivo.

```
- id: 100011
    title: "Verificar si xmlrpc.php está accesible desde la web."
    description: "Si xmlrpc.php responde a solicitudes HTTP, podría ser un riesgo de seguridad."
    rationale: "Si este archivo no es necesario, es recomendable bloquear el acceso desde el servidor web."
    remediation: "Bloquear el acceso a xmlrpc.php en el servidor web configurando las reglas adecuadas en .htaccess o nginx."
    condition: all
```

Regla 13. ID=100012

- **Propósito:** Evitar la enumeración de usuarios a través de la API REST.
- **Cuándo actúa:** Si `/wp-json/wp/v2/users` expone nombres de usuario o IDs.
- **Acción:** Restringir el endpoint con autenticación o deshabilitarlo.

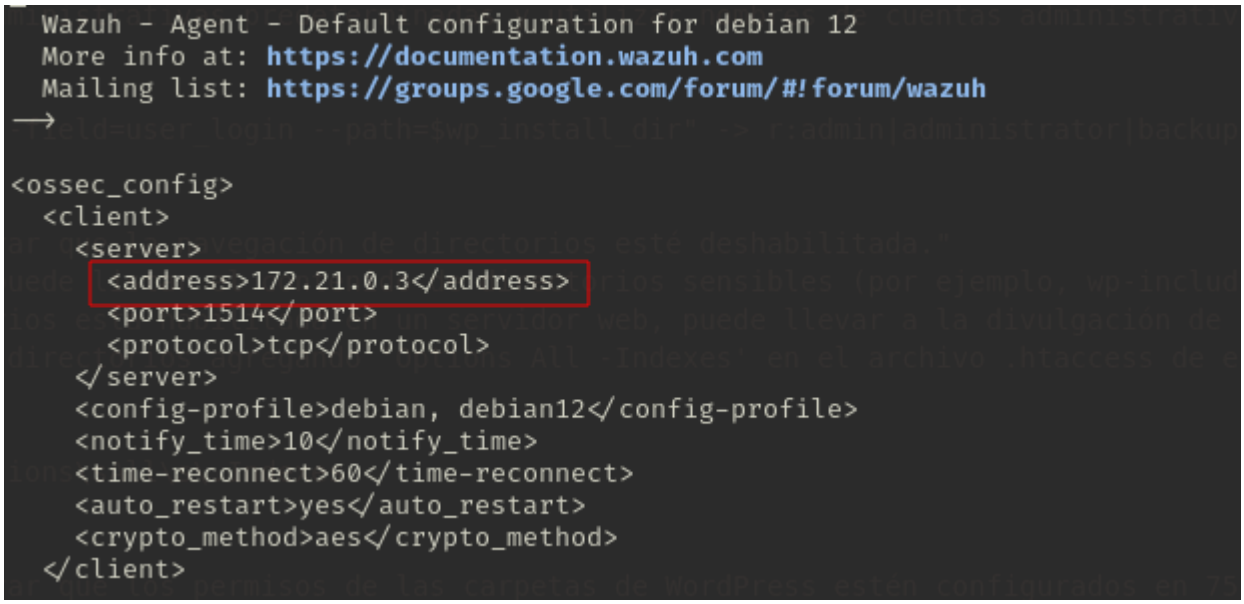
```
- id: 100012
  title: "Verificar si la API REST de WordPress expone usuarios."
  description: "Si el endpoint /wp-json/wp/v2/users devuelve datos en JSON y no un error de acceso, los nombres de usuario pueden ser extraídos."
  rationale: "Exponer usuarios a través de la API REST facilita ataques de fuerza bruta y enumeración de cuentas."
  remediation: "Restringir el acceso al endpoint o deshabilitar la API REST si no es necesaria."
  condition: all
  rules:
    - c:curl -s $wp_host/wp-json/wp/v2/users | grep -E '"id":| "name":' -> r:.*
```

Configurar el Agente de Wazuh

Para que Wazuh pueda realizar el análisis de seguridad en la instalación de WordPress, es necesario configurar el agente en el servidor para que envíe los registros hacia el servidor y sea procesado.

Editar la configuración del agente para habilitar el análisis SCA:

```
sudo nano /var/ossec/etc/ossec.conf
```



Agregar la configuración necesaria dentro del archivo de configuración del agente:

```
<policies>
  <policy>/home/local_sca_policies/custom_wordpress_policy.yml</policy>
</policies>
```

```
<sca>
  <enabled>yes</enabled>
  <scan_on_start>yes</scan_on_start>
  <interval>12h</interval>
  <skip_nfs>yes</skip_nfs>
  <policies>
    <policy>/home/local_sca_policies/custom_wordpress_policy.yml</policy>
  </policies>
</sca>
```

Reiniciar el agente de Wazuh para aplicar los cambios:

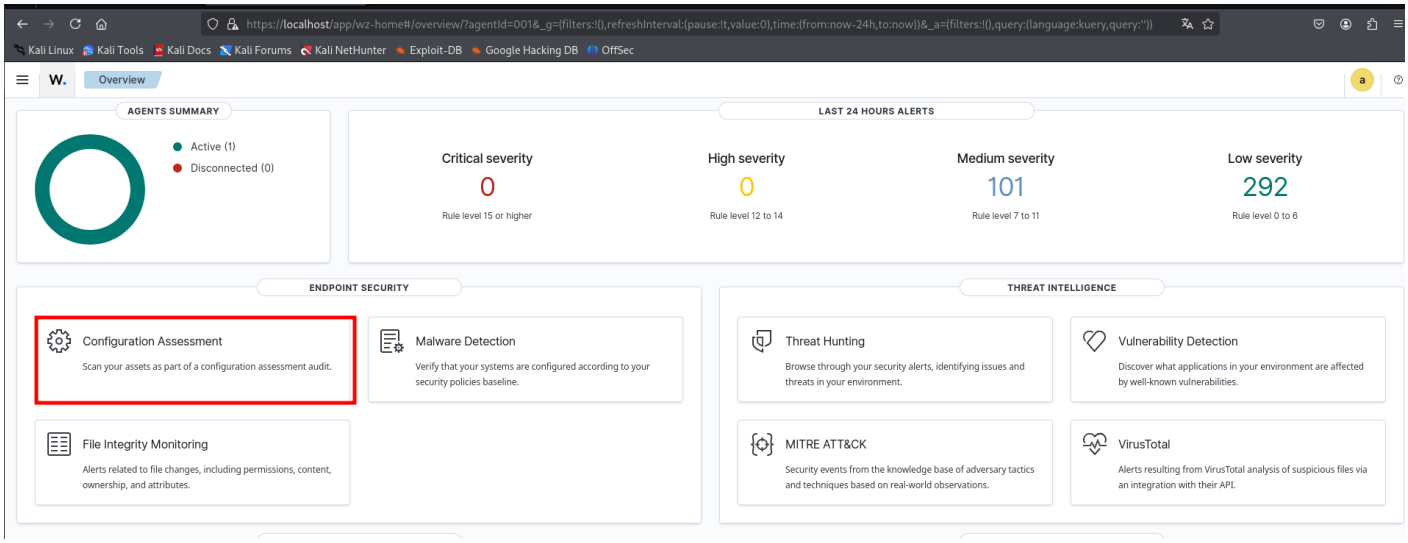
```
/var/ossec/bin/wazuh-control restart
```

```
root@c456798be056:/home# /var/ossec/bin/wazuh-control restart
Killing wazuh-modulesd ...
Killing wazuh-logcollector ...
Killing wazuh-syscheckd ...
Killing wazuh-agentd ...
Killing wazuh-execd ...
Wazuh v4.9.2 Stopped
Starting Wazuh v4.9.2 ...
Started wazuh-execd ...
Started wazuh-agentd ...
Started wazuh-syscheckd ...
Started wazuh-logcollector ...
Started wazuh-modulesd ...
Completed.
root@c456798be056:/home#
```

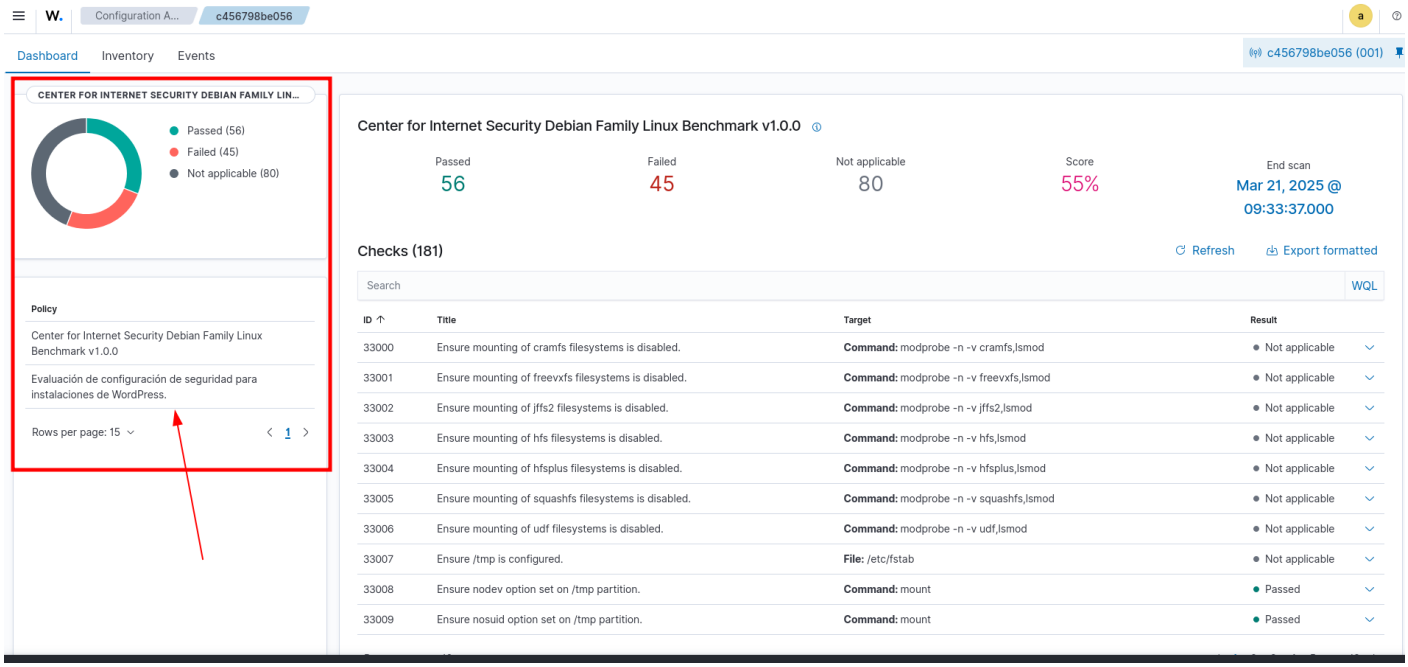
Verificación de Resultados

Una vez configurado el análisis de seguridad en Wazuh, puedes verificar los resultados desde la interfaz gráfica.

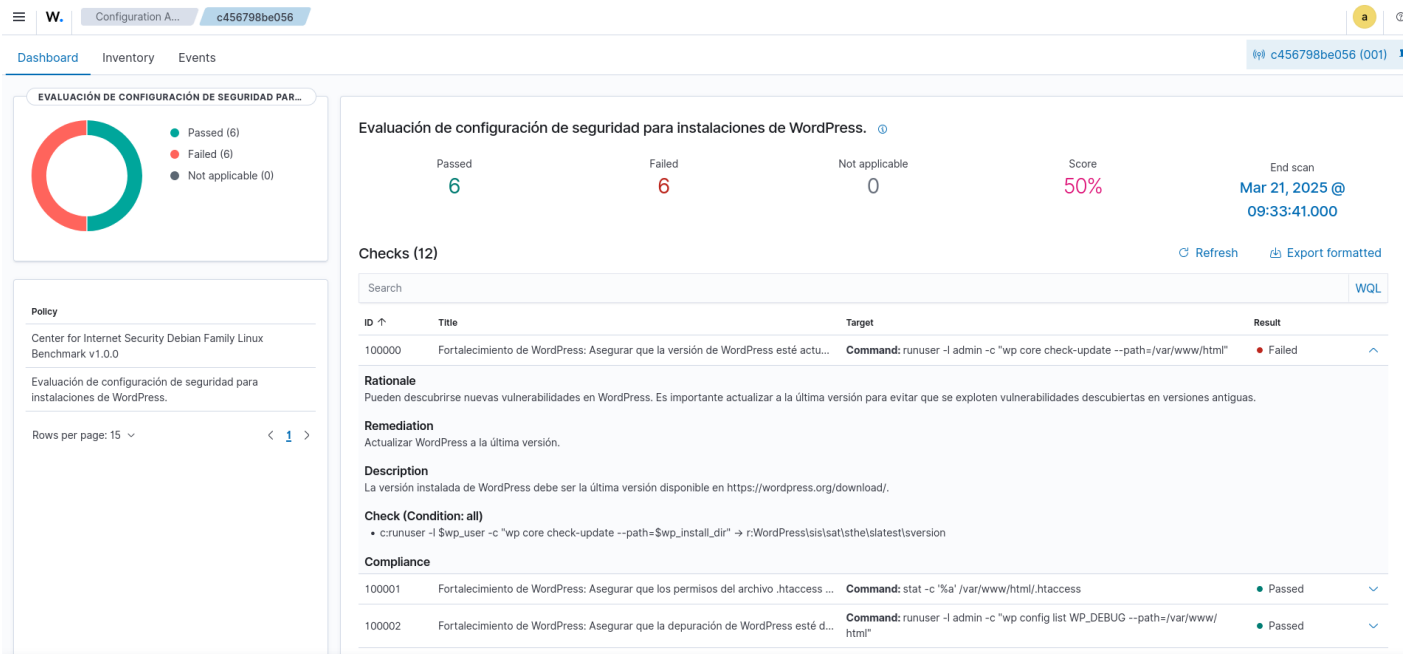
- Accede a la interfaz de Wazuh.
- Navega a Security Configuration Assessment.



- Revisa los resultados del módulo SCA de seguridad en WordPress.



- Aplica las recomendaciones de Wazuh para corregir vulnerabilidades identificadas.



Alertas de Wazuh en Wordpress

CREACIÓN DE REGLAS PERSONALIZADAS.

A continuación explicaremos la creación de reglas personalizadas que responderán ante distintos casos de intentos de intrusión.

Nos dirigimos al archivo *local_rules*

```
sudo nano /var/ossec/etc/rules/local_rules.xml
```

Para identificar distintos patrones de comportamiento anómalo en nuestro servidor, editamos el archivo con las siguientes reglas personalizadas. Es importante tener en cuenta que dichas reglas deben estar incluidas en un grupo para garantizar el orden y facilitar su visualización.

Regla 1. ID=100010

- **Eventos reconocidos:** Intentos de acceso a wp-admin mediante cualquier método HTTP.
- **Ayuda a identificar:** Posibles intentos de autenticación fallida o accesos no autorizados.
- **Cuándo se aplica:** Cuando hay tráfico dirigido a la URL wp-admin.
- **Regla en formato XML:**

```
<rule id="100010" level="6">
  <if_sid>31100,31101,31108</if_sid>
  <url>wp-admin</url>
```

```
  <protocol>GET|POST|PUT|DELETE</protocol>
  <description>Intento de acceso a wp-admin desde $(srcip)</description>
  <group>wordpress,authentication_failed,</group>
</rule>
```

Regla 2. ID=100011

- **Eventos reconocidos:** Subidas de archivos mediante async-upload.php usando métodos POST o PUT.
- **Ayuda a identificar:** Posibles intentos de subida de archivos maliciosos a través de WordPress.
- **Cuándo se aplica:** Cuando se detecta tráfico hacia wp-admin/async-upload.php.
- **Regla en formato XML:**

```
<rule id="100011" level="6">
  <if_sid>100010</if_sid>
  <url>/wp-admin/async-upload.php</url>
  <protocol>POST|PUT</protocol>
  <description>Subida de archivos mediante async-upload de WordPress desde $(srcip)</description>
  <group>wordpress,authentication_failed,</group>
</rule>
```

Regla 3. ID=100012

- **Eventos reconocidos:** Peticiones a admin-ajax.php con métodos POST o PUT.
- **Ayuda a identificar:** Uso potencial de acciones AJAX en el administrador de WordPress, que pueden ser explotadas por atacantes.
- **Cuándo se aplica:** Cuando se detecta tráfico hacia

  ```
  wp-admin/admin-ajax.php.
  ```

- **Regla en formato XML:**

```
<rule id="100012" level="6">
  <if_sid>100010</if_sid>
  <url>/wp-admin/admin-ajax.php</url>
  <protocol>POST|PUT</protocol>
  <description>Ejecución de una acción AJAX en el panel de administración desde $(srcip)</description>
  <group>wordpress,authentication_failed,</group>
</rule>
```

Regla 4. ID=100013

- **Eventos reconocidos:** Múltiples intentos de acceso fallidos a wp-admin en un corto período (20 intentos en 120 segundos).
- **Ayuda a identificar:** Ataques de fuerza bruta contra la autenticación de WordPress.
- **Cuándo se aplica:** Cuando se detectan múltiples intentos desde la misma IP en un tiempo determinado.
- **Regla en formato XML:**

```
<rule id="100013" level="11" frequency="20" timeframe="120">
  <if_matched_sid>100010</if_matched_sid>
  <same_source_ip />
  <description>Múltiples intentos de acceso a wp-admin desde $(srcip)</description>
  <mitre>
    <id>T1110</id>
    <id>T1110.001</id>
  </mitre>
  <group>attack,brute_force,wordpress,web,</group>
</rule>
```

Regla 5. ID=100014

- **Eventos reconocidos:** Accesos a directorios de plugins de WordPress mediante patrones en la URL.
- **Ayuda a identificar:** Posibles exploraciones en busca de vulnerabilidades en plugins.
- **Cuándo se aplica:** Cuando una URL coincide con

  ```
  /wp-content/plugins/([^/]+)/.
  ```

- **Regla en formato XML:**

```
<rule id="100014" level="8">
  <if_sid>31103</if_sid>
  <url type="pcre2">/wp-content/plugins/([^/]+)/</url>
  <description>Acceso a plugin WordPress posiblemente vulnerable: $(url) desde $(srcip)</description>
  <mitre>
    <id>T1190</id>
  </mitre>
</rule>
```

Regla 6. ID=100015

- **Eventos reconocidos:** Modificación del archivo wp-config.php.
- **Ayuda a identificar:** Alteraciones en un archivo crítico que podrían indicar una intrusión.
- **Cuándo se aplica:** Cuando se detecta un cambio en

  ```
  /var/www/html/wp-config.php.
  ```

- **Regla en formato XML:**

```
<rule id="100015" level="10">
  <if_group>syscheck</if_group>
  <field name="file">/var/www/html/wp-config.php</field>
  <description>Cambio detectado en archivo crítico wp-config.php</description>
</rule>
```

Revision #28
Created 19 marzo 2025 09:20:18 by Jhon Alan
Updated 14 abril 2025 14:24:00 by Jhon Alan