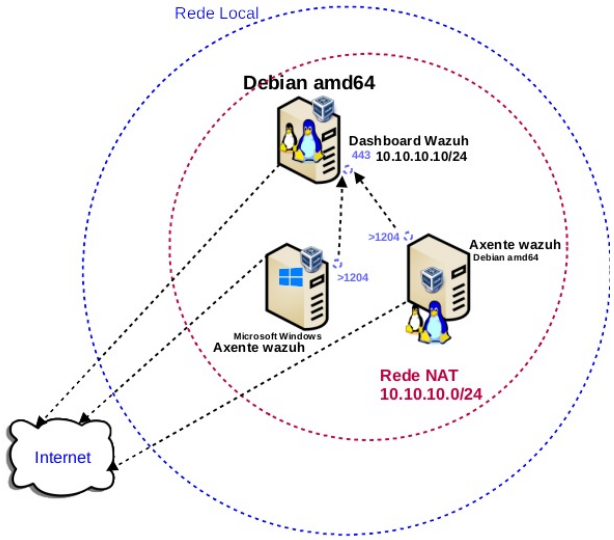
Práctica BRS
Auditoría, monitorización e protección de sistemas finais (endpoints): wazuh

ESCENARIO: wazuh

Wazuh → Máquina virtual Debian:
⊂ Host
RAM ≥ 6144MB CPU ≥ 2
Disco duro: Debian amd64
docker
Wazuh: dashboard, manager, indexer
Rede: Rede NAT si-wazuh →10.10.10.0/24
IP/MS: 10.10.10.10/24

Axente GNU/Linux
Máquina virtual Debian amd64:
⊂ Host
RAM ≥ 2048MB CPU ≥ 2
Disco duro: Debian amd64
Rede: Rede NAT si-wazuh →10.10.10.0/24
IP/MS: dinámica

Axente Microsoft Windows
Máquina virtual Microsoft Windows:
⊂ Host
RAM ≥ 2048MB CPU ≥ 2
Disco duro: Windows amd64
Rede: Rede NAT si-wazuh →10.10.10.0/24
IP/MS: dinámica



LIMITACIÓN DE RESPONSABILIDADE O autor do presente documento declina calquera responsabilidade asociada ao uso incorrecto e/ou malicioso que puidese realizarse coa información exposta no mesmo. Por tanto, non se fai responsable en ningún caso, nin pode ser considerado legalmente responsable en ningún caso, das consecuencias que poidan derivarse da información contida nel ou que esté enlazada dende ou hacia el, incluíndo os posibles erros e información incorrecta existentes, información difamatoria, así como das consecuencias que se poidan derivar sobre a súa aplicación en sistemas de información reais e/ou virtuais. Este documento foi xerado para uso didáctico e debe ser empregado en contornas privadas e virtuais controladas co permiso correspondente do administrador desas contornas.

NOTAS:

- **[1] wazuh (XDR, SIEM)**
- **[2] docker**
- **[3] docker hub**
- **[4] Cheat Sheet Docker A3**
- **[5] Explicacion Cheat-Sheet-Docker_A3**
- **[6] wazuh - docker**

Máquina virtual Debian amd64

1. **Configuración da rede según o escenario**

   Na contorna gráfica abrir un terminal e executar:

   user@debian:~$ setxkbmap es #Cambiar o mapa de teclado ao idioma español.
   user@debian:~$ su - #Acceder á consola de root(administrador) a través comando *su*, o cal solicita o contrasinal do usuario *root*. Escribir o contrasinal de *root* para acceder.
   root@debian:~# /etc/init.d/avahi-daemon stop || (systemctl stop avahi-daemon && systemctl stop avahi-daemon.socket) #Parar o demo avahi-daemon(control resolución de nomes) para poder configurar de forma manual a configuración de rede e non ter conflicto con este demo.
   root@debian:~# systemctl disable avahi-daemon #Impide que o servizo avahi-daemon sexa iniciado no arranque xerando os links K* nos runlevels (/etc/rcX.d)
   root@debian:~# /etc/init.d/network-manager stop || pkill NetworkManager #Parar o demo network-manager(xestor de rede) ou o script NetworkManager (executado sen ser demo) para poder configurar doutro xeito (co comando ip(ifconfig) de forma manual ou mediante networking (ficheiros /etc/init.d/networking, /etc/init.d/networking.d) a configuración de rede e non ter conflicto con este xestor.
   root@debian:~# systemctl disable NetworkManager #Impide que o servizo NetworkManager sexa iniciado no arranque xerando os links K* nos runlevels (/etc/rcX.d)
   root@debian:~# ip addr show #Amosar a configuración de todas as tarxetas de rede. Nesta caso, as tarxetas de redes: loopback(lo) e a correspondente á Rede NAT(enp0s3).

   ```
   $ man interfaces #Ver ás páxinas de manual referente ao ficheiro de configuración de rede /etc/network/interfaces
   $ cat /etc/network/interfaces #Amosar o contido do ficheiro configuración de rede /etc/network/interfaces
   $ ls -l /etc/network/interfaces.d #Listar de forma extendida o contido do directorio /etc/network/interfaces.d
   ```

   root@debian:~# cat >> /etc/network/interfaces <<EOF
   #Configuración interface enp0s3
   auto enp0s3
   iface enp0s3 inet static
     address 10.10.10.10/24
     gateway 10.10.10.1
   EOF #Engadir ao ficheiro /etc/network/interfaces a configuación de enp0s3
   root@debian:~# echo -e 'nameserver 8.8.8.8\nnameserver 8.8.4.4' > /etc/resolv.conf #Agregar servidores DNS para resolución de nomes.
   root@debian:~# /etc/init.d/networking status #Comprobar o estado do demo networking, é dicir, comprobar se está activa a configuración de rede en /etc/network/interfaces (/etc/network/interfaces.d).
   root@debian:~# /etc/init.d/networking restart || /etc/init.d/networking start #Arrancar o demo networking, é dicir, activar a configuración de rede en /etc/network/interfaces (/etc/network/interfaces.d).
   root@debian:~# /etc/init.d/networking status #Comprobar o estado do demo networking, é dicir, comprobar se está activa a configuración de rede en /etc/network/interfaces (/etc/network/interfaces.d).
   root@debian:~# ip addr show #Amosar a configuración de todas as tarxetas de rede. Nesta caso, as tarxetas de redes: loopback(lo) e a correspondente á Rede NAT(enp0s3).
   root@debian:~# ping -c4 10.10.10.10 #Comprobar mediante o comando ping a conectividade coa interface de rede local enp0s3

2. **docker: Instalación e arranque**

   Executar no anterior terminal:

   root@debian:~# apt update || apt-get update #Actualizar repositorios declarados no ficheiro */etc/apt/souces.list* e nos ficheiros existentes no directorio */etc/apt/sources.list.d*
   Así, unha vez realizada a consulta dos ficheiros existentes nas rutas anteriores, descárganse uns ficheiros coas listas de paquetes posibles a instalar. Estes ficheiros son gardados en */var/lib/apt/lists*
   root@debian:~# apt -y install docker.io || apt-get -y install docker.io #Instalar o paquete de nome *docker.io*. Co parámetro -y automaticamente asumimos yes a calquera pregunta que ocorra na instalación do paquete.
   root@debian:~# apt -y install docker-compose || apt-get -y install docker-compose #Instalar o paquete de nome *docker-compose*. Co parámetro -y automaticamente asumimos yes a calquera pregunta que ocorra na instalación do paquete.
   root@debian:~# /etc/init.d/docker status || systemctl status docker #Comprobar o estado do demo docker
   root@debian:~# /etc/init.d/docker start || systemctl start docker #Arrancar o demo docker
   root@debian:~# /etc/init.d/docker status || systemctl status docker #Comprobar o estado do demo docker

3. **wazuh: Implantación en docker**

Na contorna gráfica abrir un terminal e executar:

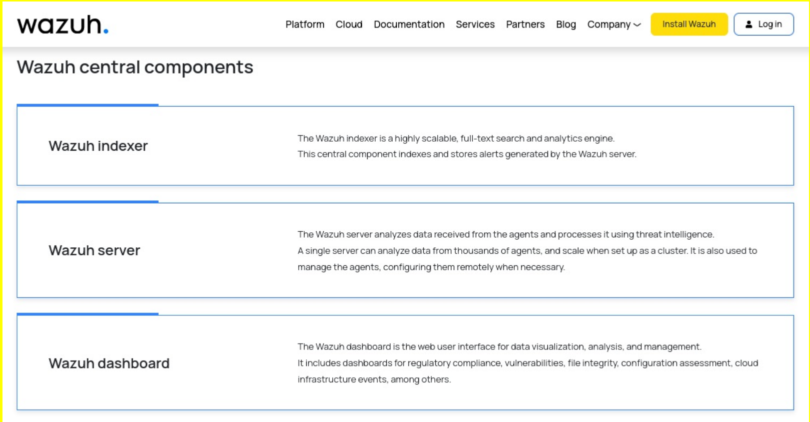
root@debian:~# git clone https://github.com/wazuh/wazuh-docker.git #Descargar wazuh dende github mediante git clone.

root@debian:~# cd wazuh-docker #Acceder ao directorio wazuh-docker

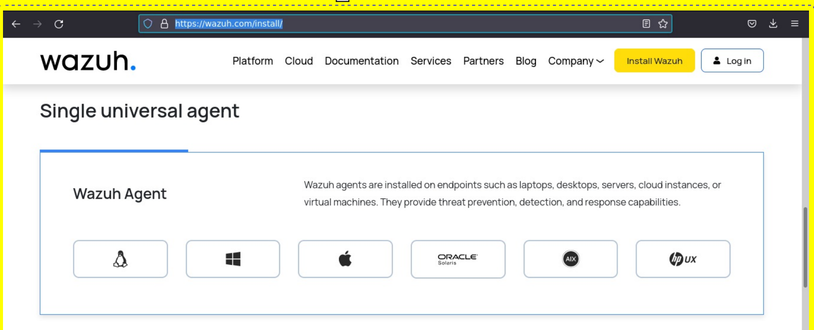
root@debian:~# git checkout tags/v4.9.2 -b wazuh-4.9.2 #Crear unha nova rama chamada wazuh-4.9.2 baseada na versión etiquetada v4.9.2.

Wazuh permite auditar, monitorizar e protexer sistemas finais ou endpoints.
A solución Wazuh baséase no axente Wazuh, que se desprega nos puntos finais monitorizados, e en tres compoñentes centrais: o servidor Wazuh, o indexador de Wazuh e o panel de control de Wazuh:

- O indexador Wazuh é un motor de busca e análise de texto completo altamente escalable. Este compoñente central indexa e almacena as alertas xeradas polo servidor Wazuh.
- O servidor Wazuh analiza os datos recibidos dos axentes. Procésao mediante decodificadores e regras, utilizando intelixencia sobre ameazas para buscar indicadores de compromiso (IOC) coñecidos. Un único servidor pode analizar datos de centos ou miles de axentes e escalar horizontalmente cando se configura como un clúster. Este compoñente central tamén se utiliza para xestionar os axentes, configurándoos e actualizándoos de forma remota cando sexa necesario.
- O panel de control de Wazuh é a interface de usuario web para a visualización e análise de datos. Inclúe paneis listos para usar para eventos de seguridade, cumprimento normativo (por exemplo, PCI DSS, GDPR, CIS, HIPAA, NIST 800-53), aplicacións vulnerables detectadas, datos de seguimento da integridade dos ficheiros, resultados da avaliación da configuración, seguimento da infraestrutura na nube. eventos e outros. Tamén se usa para xestionar a configuración de Wazuh e supervisar o seu estado.



Os axentes Wazuh instálanse en puntos finais(endpoints) como ordenadores portátiles, escritorios, servidores, instancias de nube ou máquinas virtuais. Ofrecen capacidades de prevención, detección e resposta de ameazas. Así, podemos despregar **axentes wazuh (Single Universal Agent)** sobre sistemas operativos: GNU/Linux, Microsoft Windows, MacOs, Solaris, AIX, HP/UX.



root@debian:~# cd single-node #Acceder ao cartafol single-node.

root@debian:~# cat README.md #Ver o contido do ficheiro README.md que contén os pasos para o despregue de Wazuh en docker para a configuranción nun único nodo (single node)

root@debian:~# sysctl -w vm.max_map_count=262144 #Incrementar o valor max_map_count: número máximo de mapas de memoria

root@debian:~# docker-compose -f generate-indexer-certs.yml run --rm generator #Xerar certificados para cada nodo para asegurar a comunicación entre os nodos.

root@debian:~# docker-compose up #Despregar os contenedores docker a través de docker-compose. Como non se indica coa opción -f un ficheiro, o ficheiro a interpretar será docker-compose.yml

root@debian:~# cat docker-compose.yml #Ver o contido do ficheiro wazuh-docker/single-node/docker-compose.yml

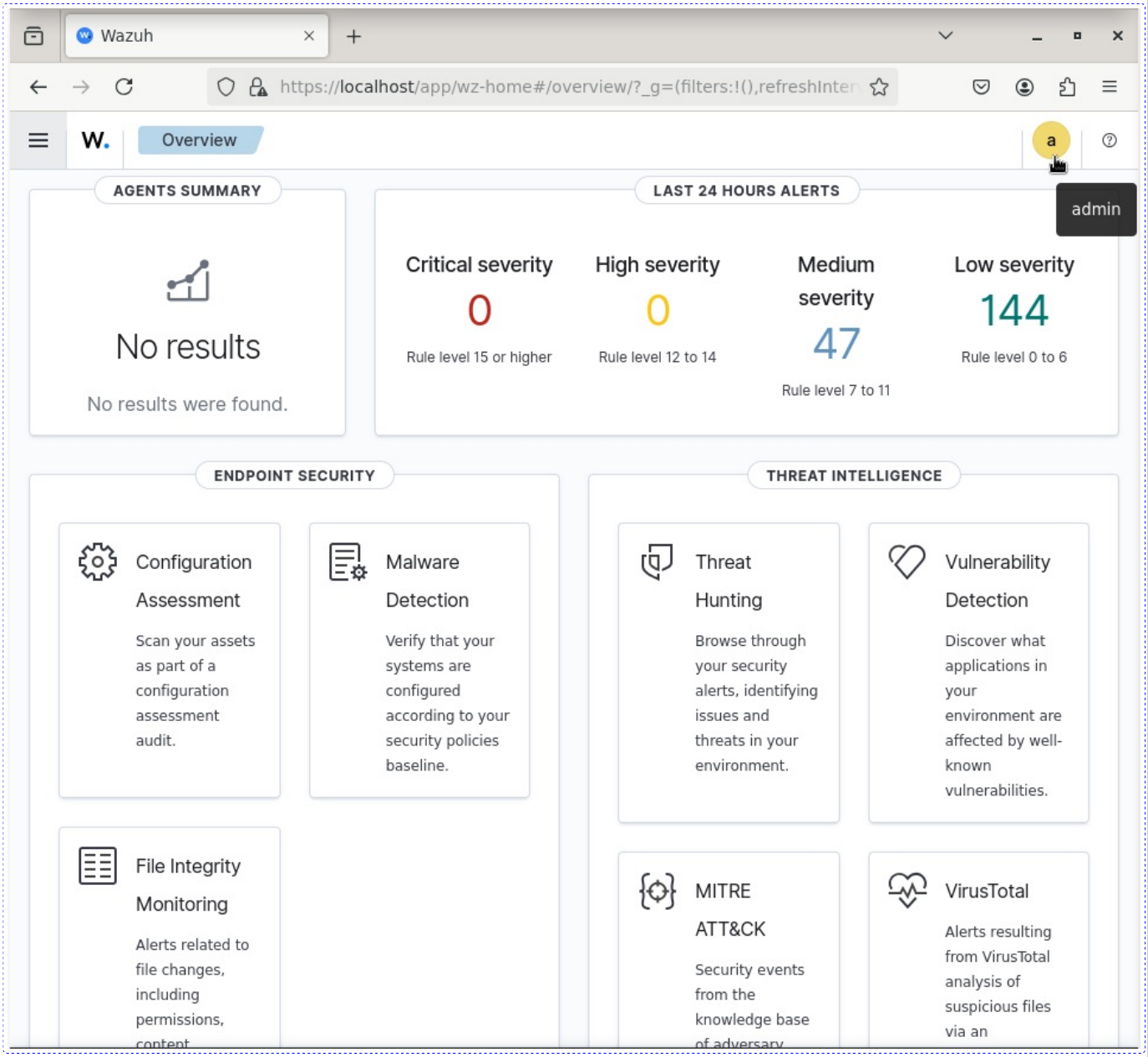
root@debian:~# docker container ls #Listar os contenedores docker activos.

```
CONTAINER ID   IMAGE                          COMMAND                  CREATED          STATUS          PORTS                                                                                                                                        NAMES
b59ef4debf7a   wazuh/wazuh-dashboard:4.9.2    "/entrypoint.sh"         27 seconds ago   Up 24 seconds   443/tcp, 0.0.0.0:443->5601/tcp, :::443->5601/tcp                                                                                             single-
node_wazuh.dashboard_1
96ce4501a07b   wazuh/wazuh-indexer:4.9.2      "/entrypoint.sh open…"   36 seconds ago   Up 27 seconds   0.0.0.0:9200->9200/tcp, :::9200->9200/tcp                                                                                                    single-
node_wazuh.indexer_1
959ea219e2e1   wazuh/wazuh-manager:4.9.2      "/init"                  36 seconds ago   Up 27 seconds   0.0.0.0:1514-1515->1514-1515/tcp, :::1514-1515->1514-1515/tcp, 0.0.0.0:514->514/udp, :::514->514/udp, 0.0.0.0:55000->55000/tcp, :::55000->55000/tcp, 1516/tcp   single-
node_wazuh.manager_1
```

Na contorna gráfica abrir outro terminal e executar:

user@debian:~$ firefox https://localhost:443 & #Lanzar o navegador firefox na URL https://localhost:443 no porto TCP 443, realizando a execución en segundo plano (&), é dicir, acceder ao dashboard(panel de control) de wazuh.

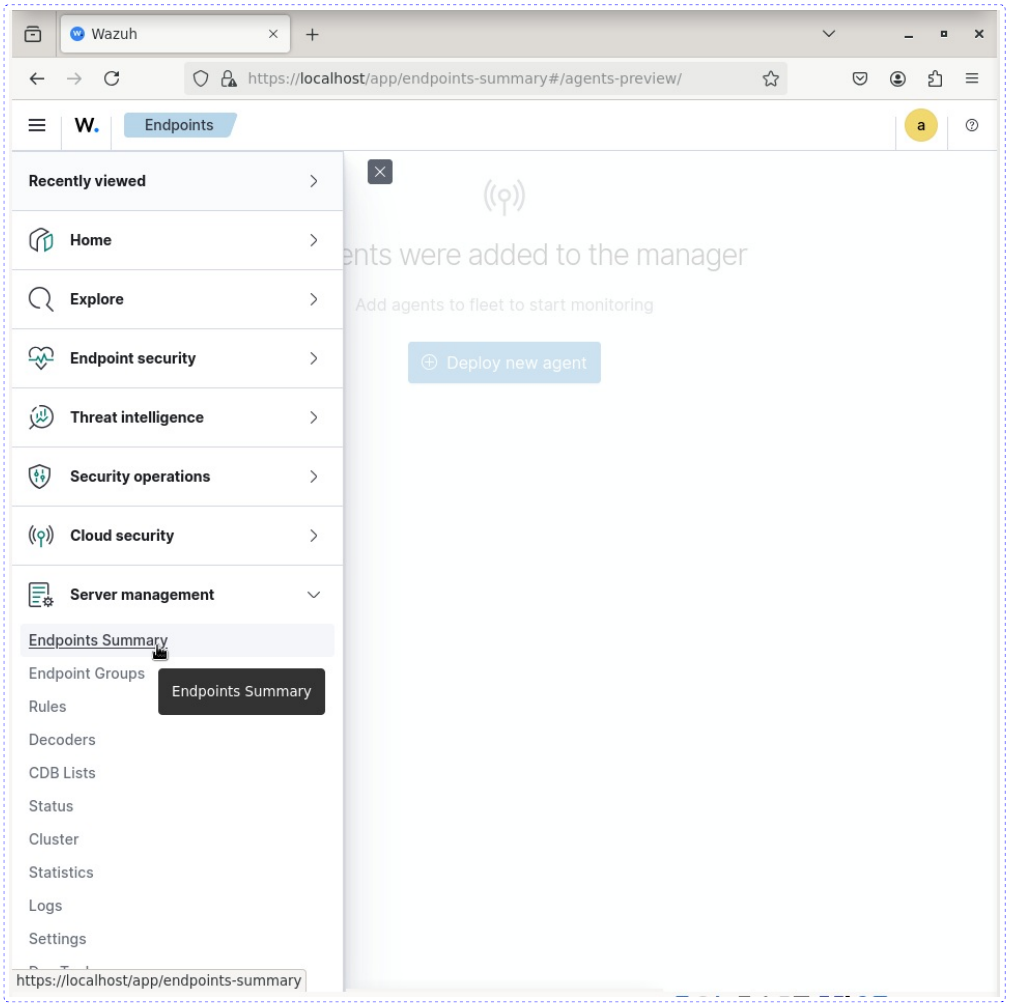
Credenciais de acceso predeterminadas: admin/SecretPassword



4. **Engadir axentes**

   A. **Debian GNU/Linux**

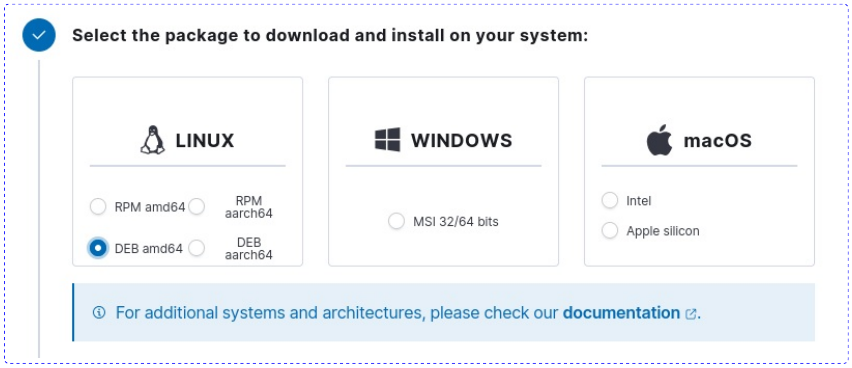
      1. Acceder a *Server management* → *Endpoints Summary*



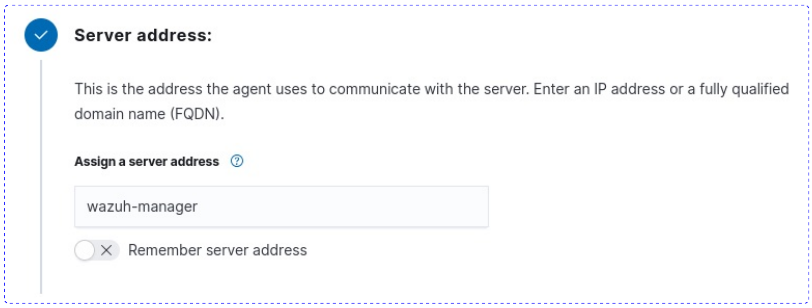
      2. Acceder a Deploy new agent

      3. Elixir:

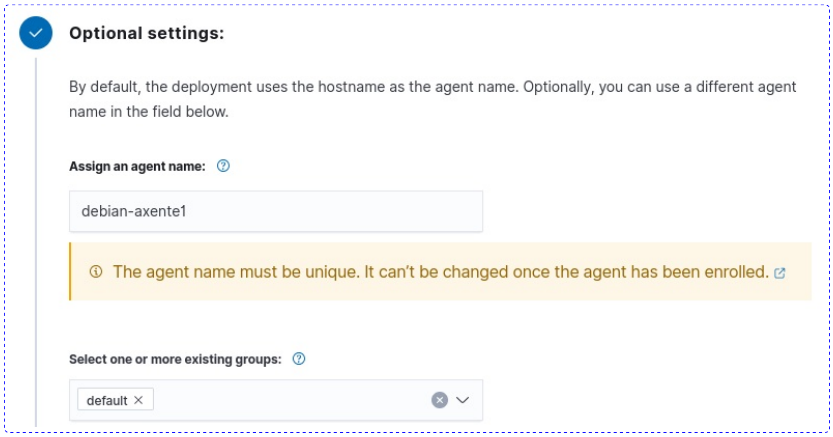
         - Sistema operativo e arquitectura: *LINUX* → *DEB amd64*



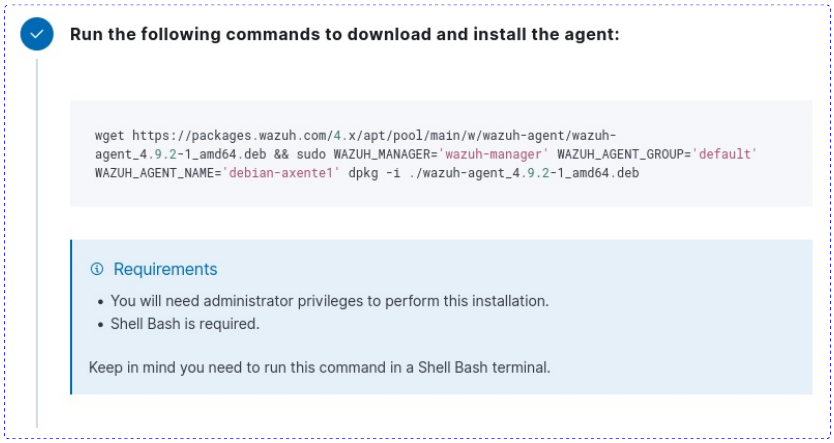
- Dirección IP/FQDN servidor wazuh: *wazuh.manager*



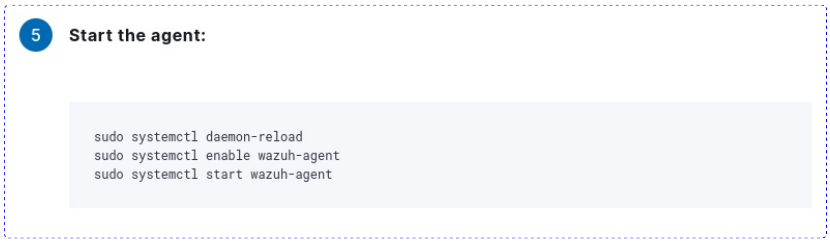
- Asignar un nome e un grupo ao axente: *debian-axente1* → *default*



- No apartado 4 (Instalar e engadir o axente) aparece un comando coas opcións escollidas. Este comando debe ser executado co usuario root no axente para engadilo como tal en wazuh:

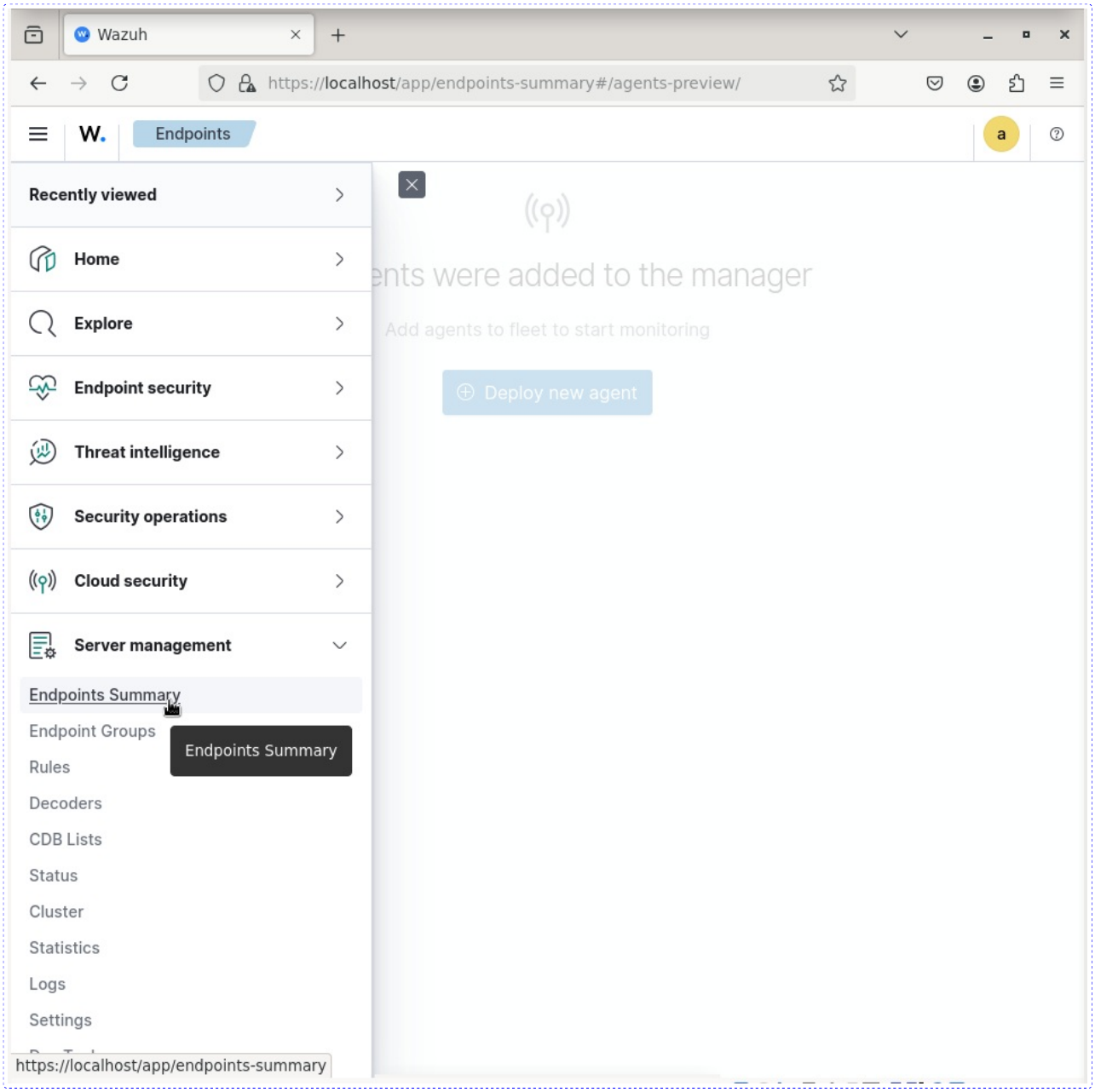


- No apartado 5 aparecen os comandos necesarios, según o sistema Systemd, para recargar o servizo pertencente ao axente (wazuh-agent) e así poidamos ver no dashboard a este novo axente agregado.



B. **Microsoft Windows**

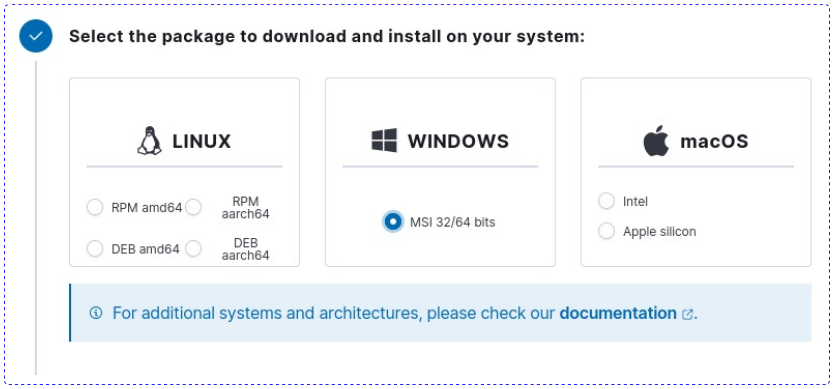
1. Acceder a *Server management* → *Endpoints Summary*



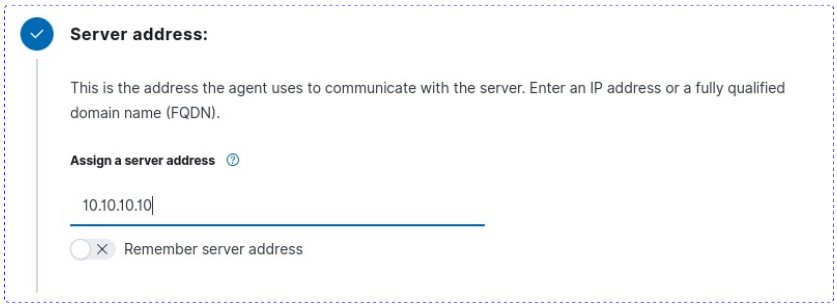
2. Acceder a Deploy new agent

3. Elixir:

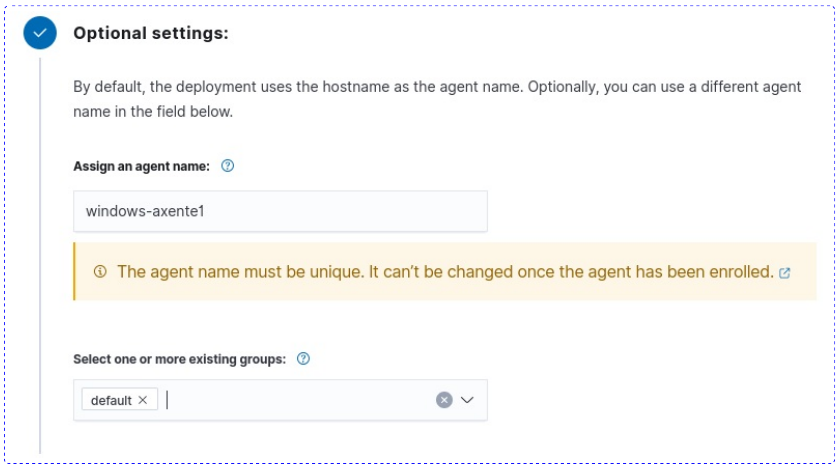
   - Sistema operativo: *Microsoft Windows*



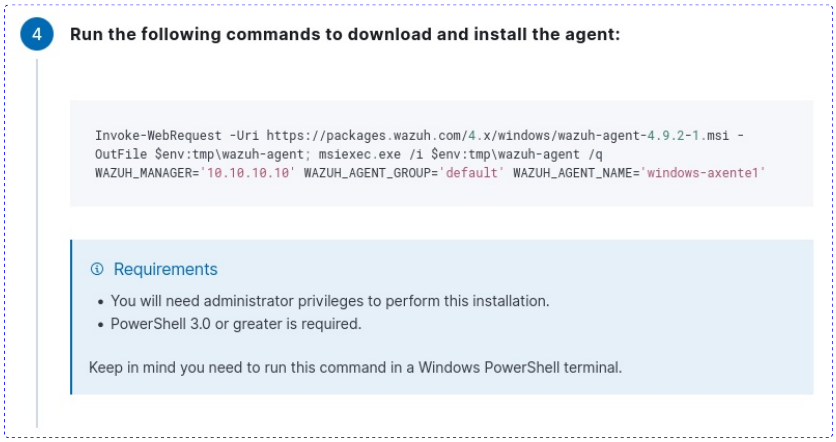
- Dirección IP/FQDN servidor wazuh: *10.10.10.10*



- Asignar un nome e un grupo ao axente: *windows-axente1* → *default*



- No apartado 4 (Instalar e engadir o axente) aparece un comando coas opcións escollidas. Este comando debe ser executado co usuario administrador nunha consola de powershell no axente para engadilo como tal en wazuh:



- No apartado 5 aparece o comando necesario para recargar o servizo pertencente ao axente (wazuh-agent) e así poidamos ver no dashboard a este novo axente agregado.

Máquinas virtuais axentes

5. **Acceder ao axente Debian GNU/Linux: debianA**
   Executar nunha consola de comandos:
   usuario@debianA:~$ setxkbmap es #Cambiar o mapa de teclado ao idioma español.
   usuario@debianA:~$ ip addr show #Amosar a configuración de todas as tarxetas de rede. Nesta caso, as tarxetas de redes: loopback(lo) e a correspondente á Rede NAT(enp0s3).
   usuario@debianA:~$ ping -c4 10.10.10.10 #Comprobar mediante o comando ping a conectividade co servidor wazuh
   usuario@debianA:~$ su - #Acceder á consola de root(administrador) a través comando *su*, o cal solicita o contrasinal do usuario *root*. Escribir o contrasinal de *root* para acceder.
   root@debianA:~# echo '10.10.10.10 wazuh.manager' >> /etc/hosts #Engadir no ficheiro /etc/hosts, é dicir, na táboa estática de búsqueda para nomes de host (DNS) o nome wazuh.manager, para que atenda á IP 10.10.10.10
   root@debianA:~# wget https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-agent/wazuh-agent_4.9.2-1_amd64.deb && sudo WAZUH_MANAGER='wazuh-manager' WAZUH_AGENT_GROUP='default' WAZUH_AGENT_NAME='debian-axente1' dpkg -i ./wazuh-agent_4.9.2-1_amd64.deb #Executar o comando do apartado 3.A.5 que nos ofrece wazuh para poder agregar este sistema operativo como axente de wazuh.
   root@debianA:~# systemctl daemon-reload && systemctl enable wazuh-agent && systemctl start wazuh-agent Arrancar o servizo do axente para que poida ser visto no dashboard de wazuh.

6. **Acceder ao axente Microsoft Windows**
   Executar nun terminal:

   > systeminfo #Amosar información de configuración detallada sobre o equipo e o seu sistema operativo.

   > ipconfig /all #Amosar a configuración TCP/IP completa de todas as interfaces de rede.

   > ping -c4 10.10.10.10 #Comprobar mediante o comando ping a conectividade co servidor wazuh

   Executar en powershell con permisos de administrador:

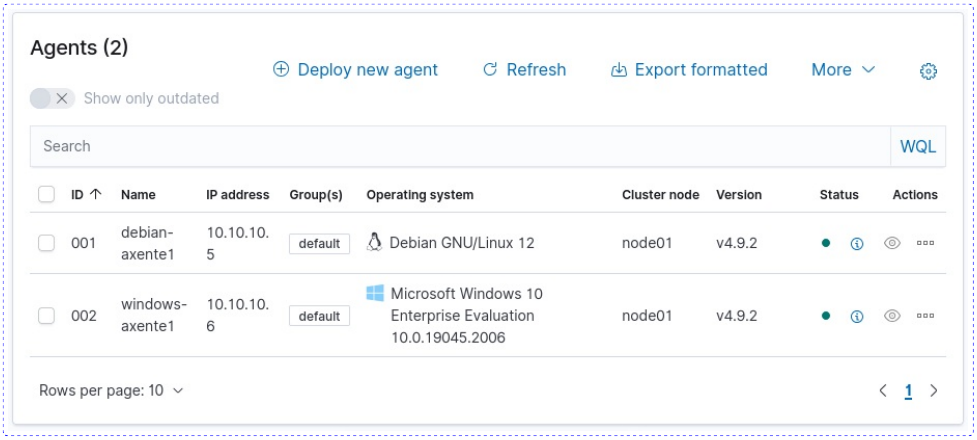
   > Invoke-WebRequest -Uri https://packages.wazuh.com/4.x/windows/wazuh-agent-4.9.2-1.msi -OutFile $env:tmp\wazuh-agent; msiexec.exe /i $env:tmp\wazuh-agent /q WAZUH_MANAGER='10.10.10.10' WAZUH_AGENT_GROUP='default' WAZUH_AGENT_NAME='windows-axente1' #Executar o comando do apartado 3.B.3 que nos ofrece wazuh para poder agregar este sistema operativo como axente de wazuh.

   > NET START WazuhSvc Arrancar o servizo do axente para que poida ser visto no dashboard de wazuh.

Dashboard: Máquina virtual Debian amd64

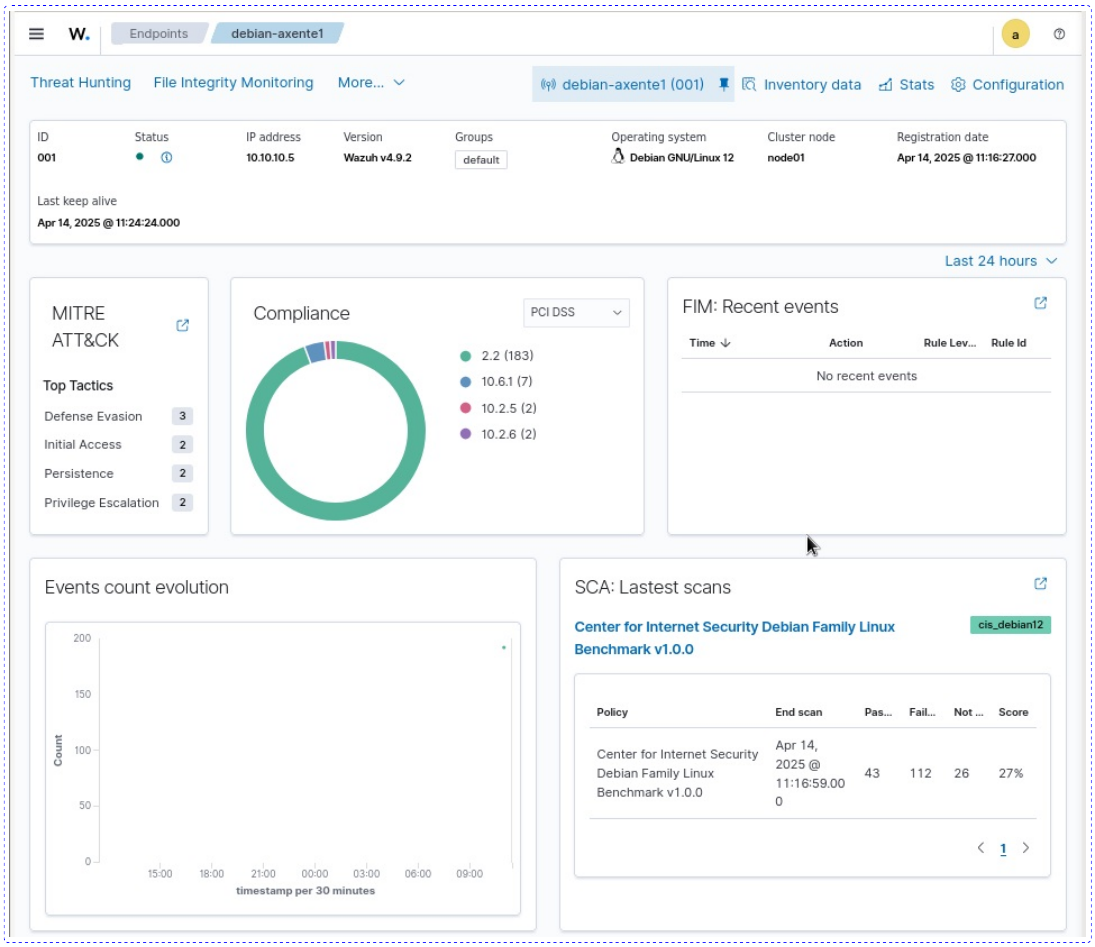
7. **wazuh: Dashboard**

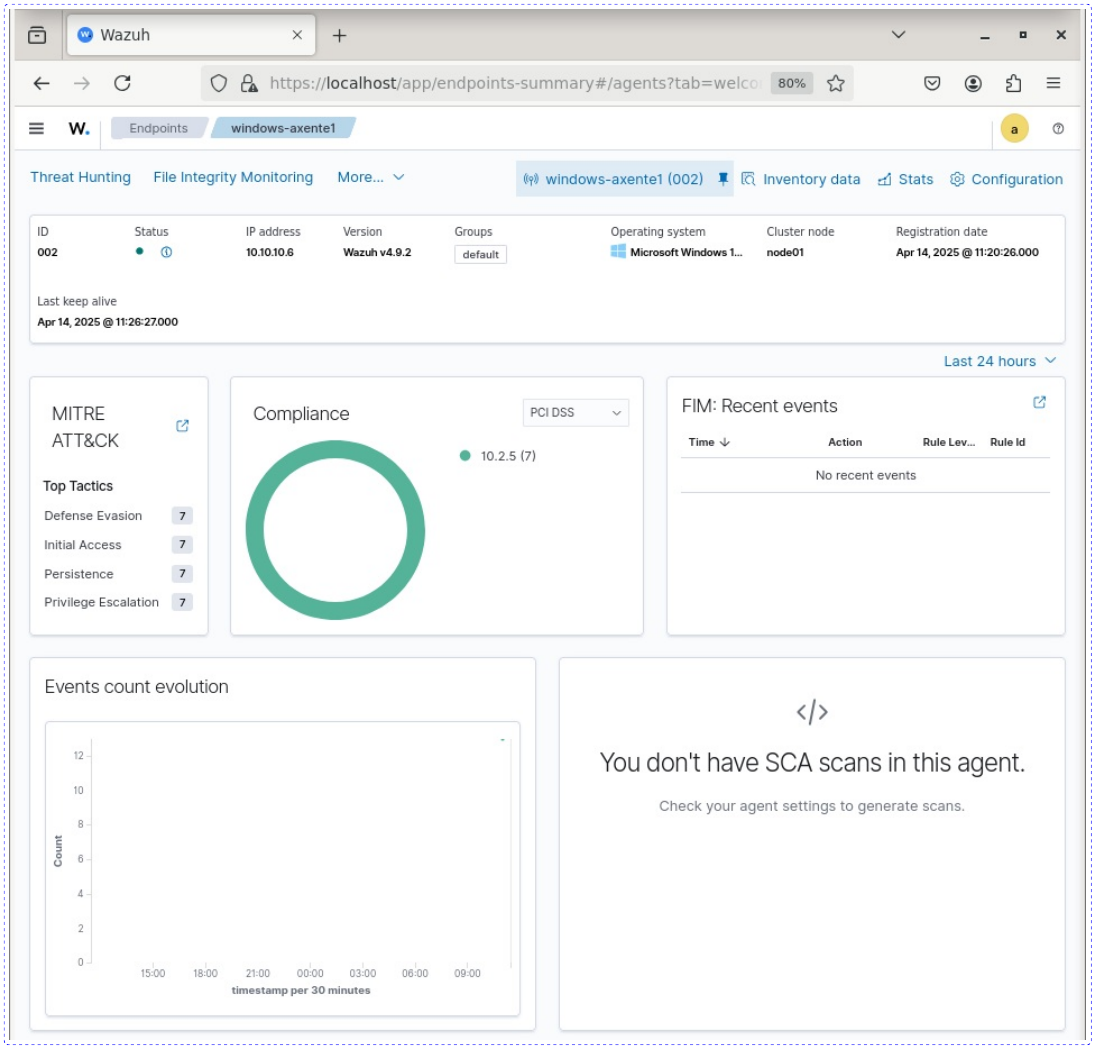
   A. **Server Management → Endpoints Summary**
   Comprobar que os axentes engadidos aparecen no dashboard e que podemos acceder á súa monitorización:



B. **Panel de cada axente**

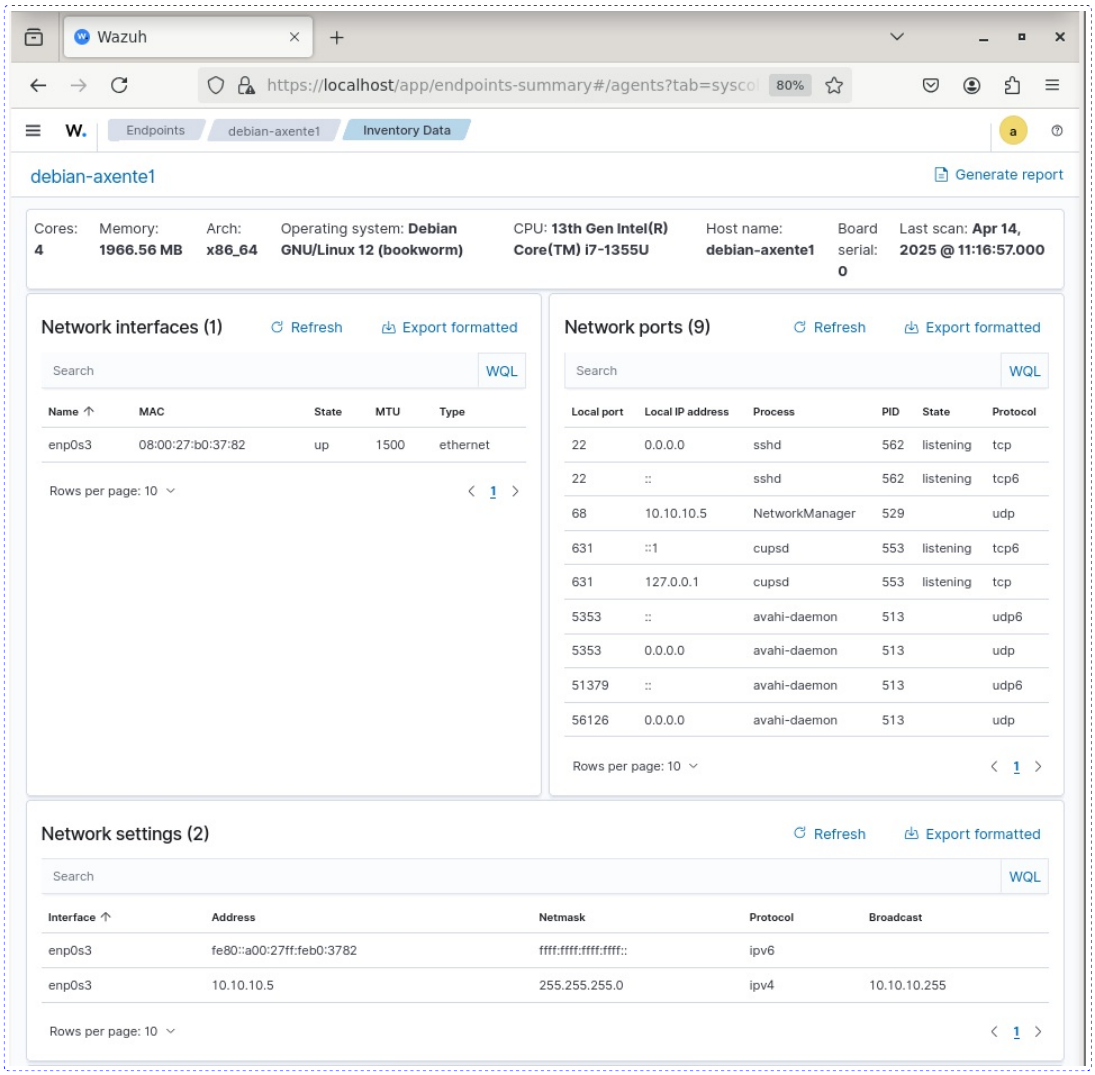
Dende *Server Management → Endpoins Summary* ao premer en cada liña referente ao axente podemos acceder ao panel de cada axente:





C. **Inventory data**

En cada axente a funcionalidade Inventory data do axente de Wazuh recolle automaticamente información detallada do sistema onde está instalado —como hardware, software, paquetes, servizos, procesos, usuarios e rede— e envíaa ao servidor para ofrecer visibilidade completa e actualizada dos activos monitorizados, permitindo auditorías, detección de cambios e alertas baseadas en configuracións do sistema.



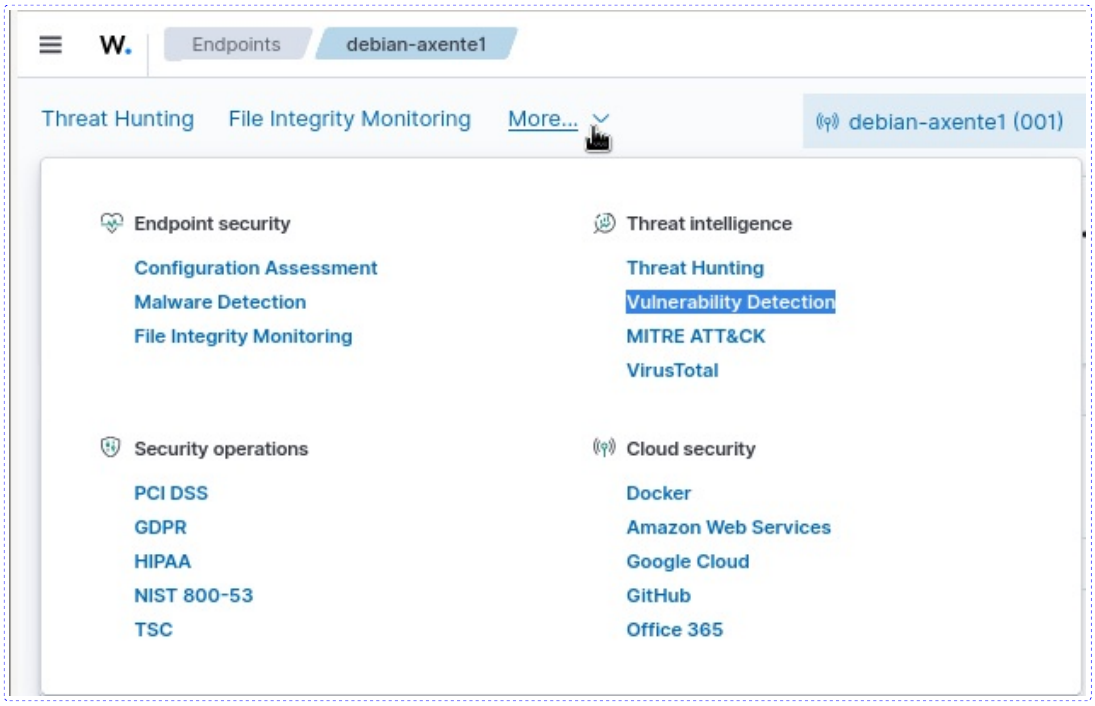
D. **Vulnerability Detection**

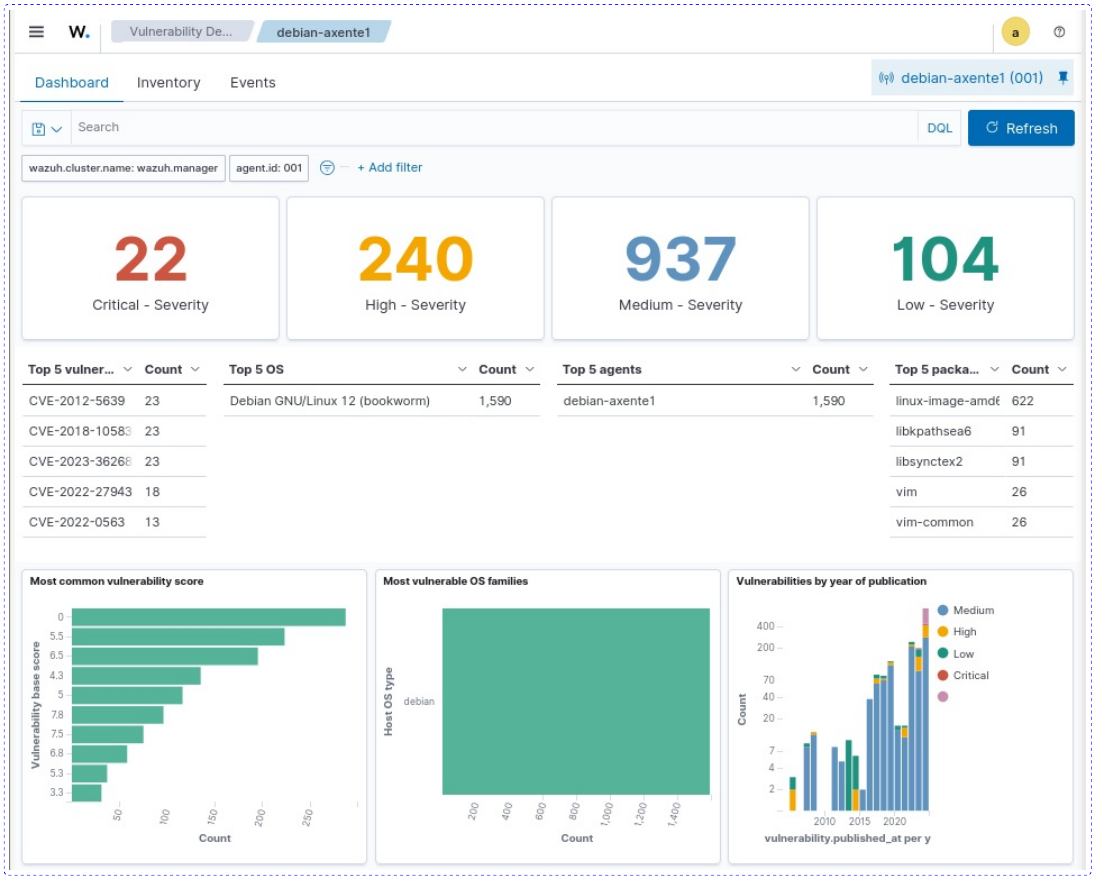
O módulo *Vulnerability Detection* de Wazuh analiza os paquetes instalados nos sistemas monitorizados e compara as súas versións cunha base de datos de vulnerabilidades coñecidas (CVEs), co obxectivo de detectar software vulnerable; esta análise realízase de forma automática e periódica, xerando alertas se se atopan coincidencias, o que permite identificar riscos de seguridade e priorizar accións de corrección.

Na configuración actual, aínda que no contedor docker do manager de wazuh o módulo **vulnerability-detection** está activado (yes), faltan as definicións dos providers (provedores) de fontes de vulnerabilidades. O módulo necesita saber onde buscar as bases de datos de vulnerabilidades e cales usar. Para un axente Debian, necesitas polo menos o provedor debian e idealmente tamén o nvd (National Vulnerability Database). Así, debemos engadir explicitamente os bloques dentro da sección . Unha configuración máis completa e funcional sería así:

```
user@debian:~$ su - #Acceder á consola de root(administrador) a travédo comando su, o cal solicita o contrasinal do usuario root. Escribir o contrasinal de root para acceder.
root@debian:~# docker exec -it single-node_wazuh.manager_1 bash
bash-5.2# yum update && yum install vim -y
bash-5.2# NUM=$(grep -n '</vulnerability-detection>' /var/ossec/etc/ossec.conf)
bash-5.2# sed -i "${NUM}i \ \ \ \ <provider name="debian">\
\ \ \ \ \ <enabled>yes</enabled> <!-- Activar o feed de Debian -->\
\ \ \ \ \ <update_interval>1h</update_interval> <!-- Con que frecuencia actualiza o feed de Debian -->\
\ \ \ \ </provider>\
\ \ \ \ \
\ \ \ \ <provider name="nvd">\
\ \ \ \ \ <enabled>yes</enabled> <!-- Activar o feed NVD (importante para CVEs xerais) -->\
\ \ \ \ \ <update_interval>1h</update_interval>\
\ \ \ \ </provider>" /var/ossec/etc/ossec.conf
bash-5.2# exit
root@debian:~# docker restart single-node_wazuh.manager_1
```

Agora podemos monitorizar os logs do contedor (usando *docker logs single-node_wazuh.manager_1*) e acceder no dashboard ao axente ao módulo **Vulnerability Detection**



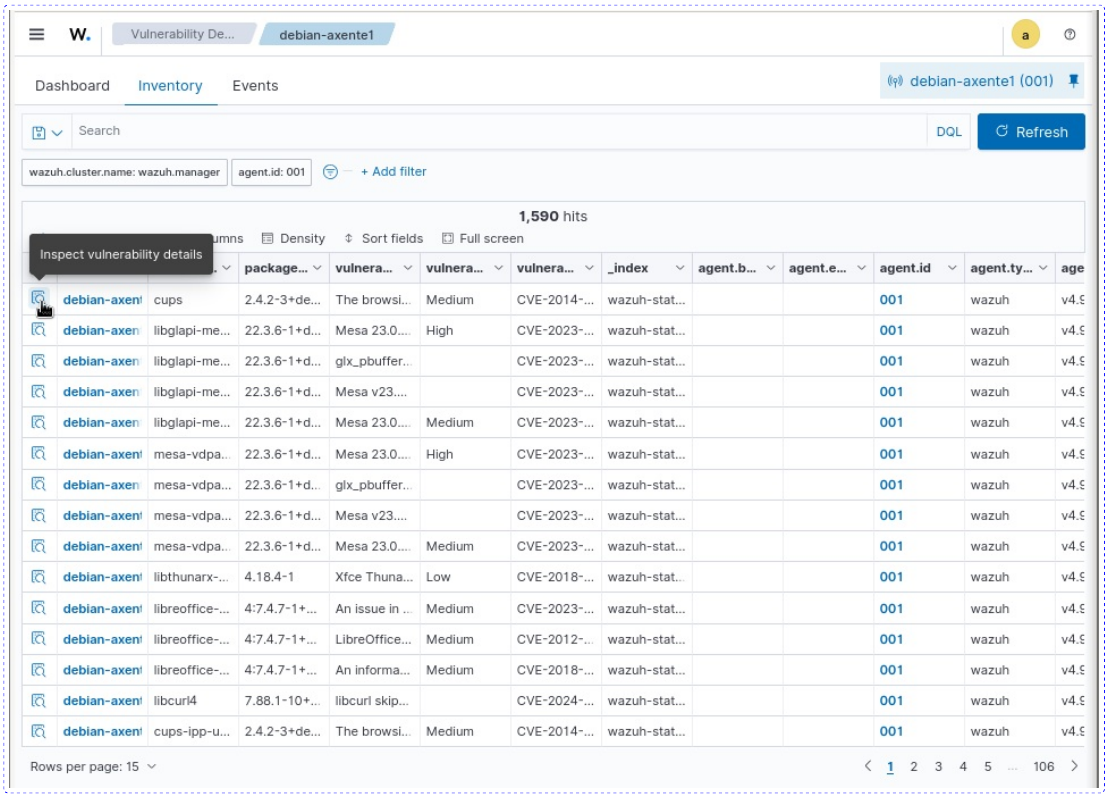


E posible que o sistema operativo do axente non esté completamente actualizado. Así que accedemos a **debian-axente1** e actualizamos o sistema:

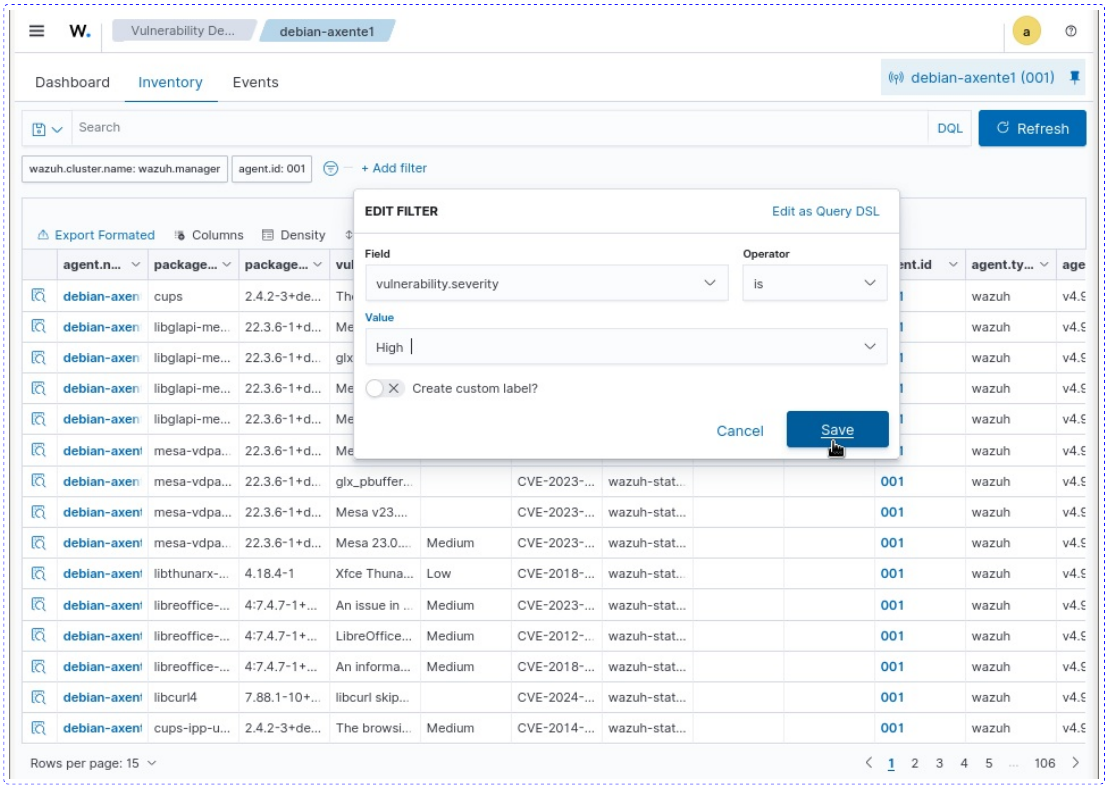
```
usuario@debian-axente1:~$ su -

root@debian-axente1:~# apt update && apt -y upgrade
```

Unha vez actualizado comprobamos de novo as vulnerabilidades existentes no sistema a través do módulo **Vulnerability Detection**

Accedemos á lapela **Inventory** para visualizar todas as vulnerabilidades. Unha vez dentro deste panel do dashboard podemos visualizar información sobre as vulnerabilidades premendo na icona lupa de cada liña.



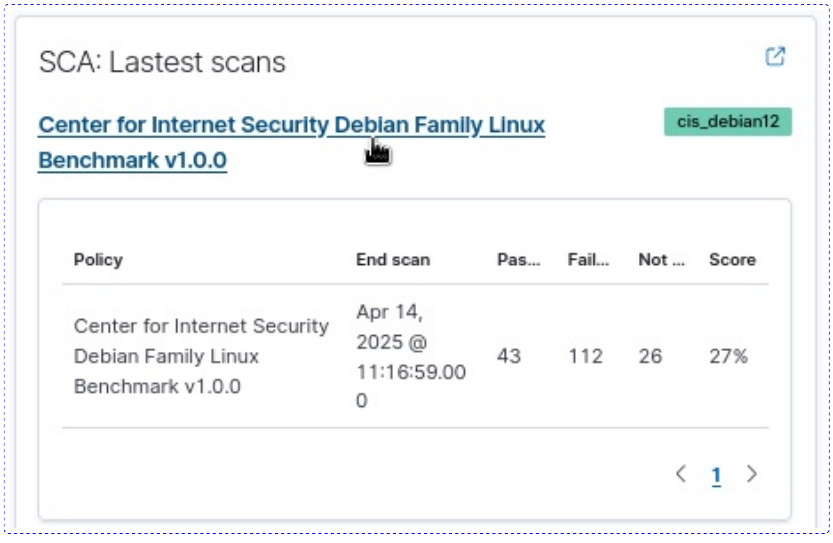
Tamén podemos xerar filtros de búsqueda para amosar vulnerabilidades, premendo **Add filter** e escollendo os criterios de búsqueda:

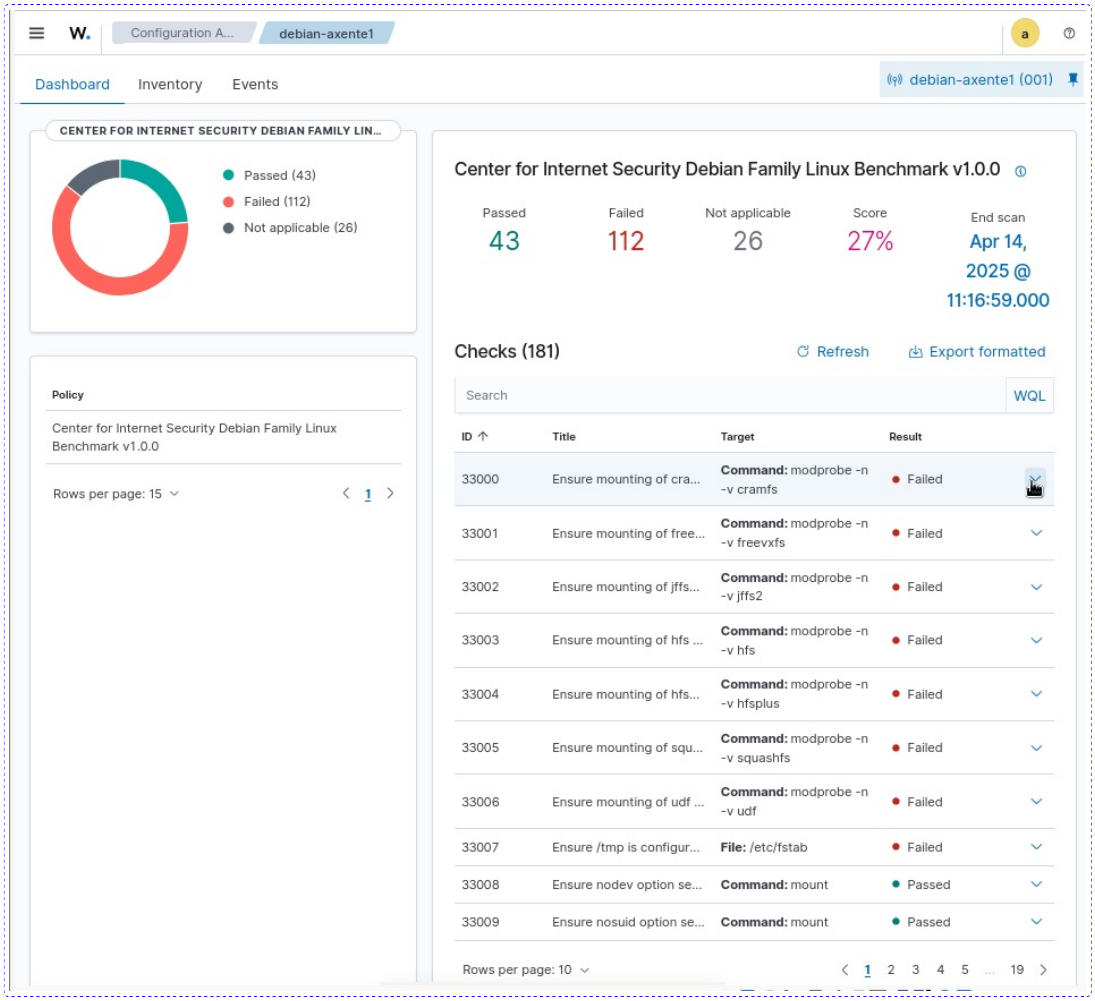


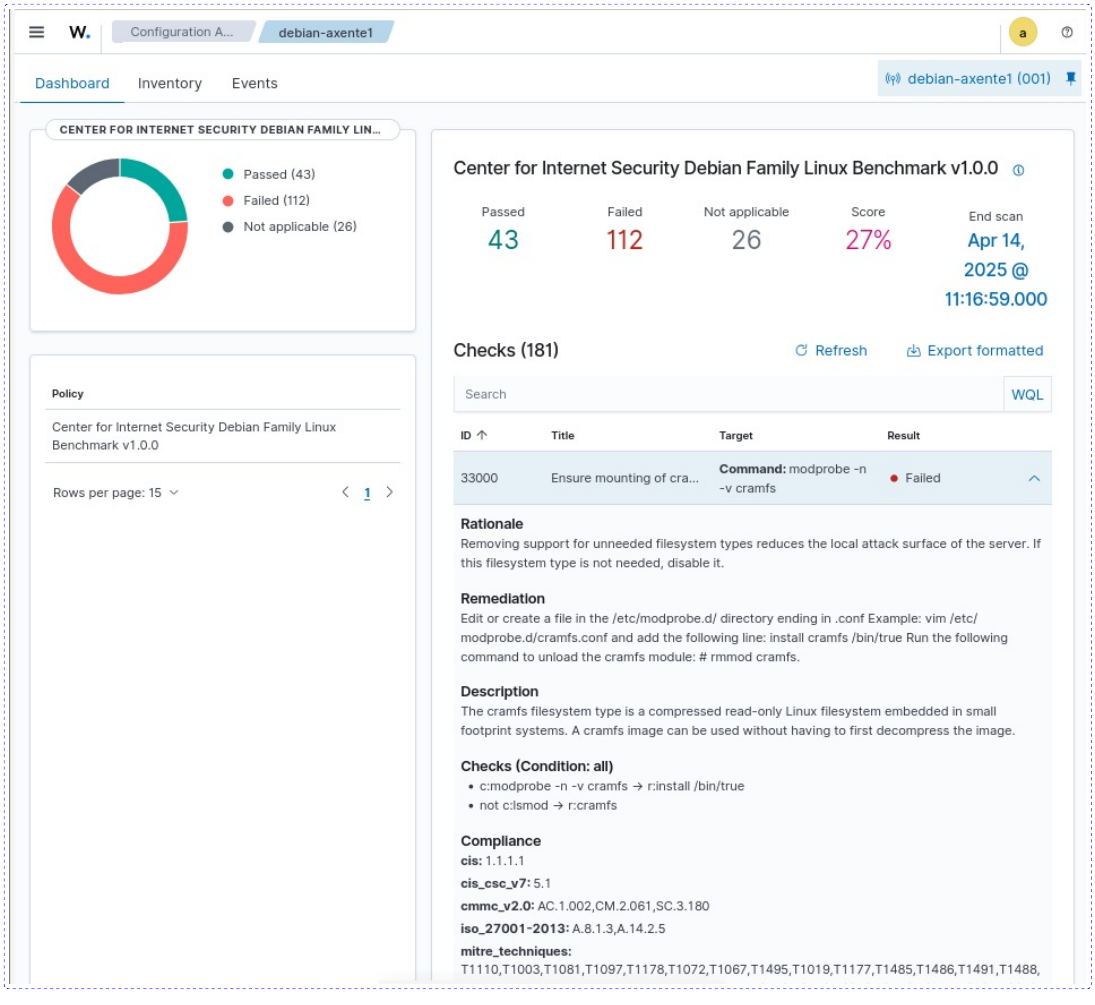
Accedendo a información dunha vulnerabilidade podemos atopar documentación sobre esta.

E. **SCA (Security Configuration Assessment)**
O módulo *SCA (Security Configuration Assessment)* de Wazuh analiza a configuración de seguridade do sistema operativo e detecta desviacións respecto a estándares ou boas prácticas. A sección **"Latest scans"** mostra os resultados máis recentes destas análises, incluíndo regras superadas ou falladas.

No caso concreto de: **Center for Internet Security Debian Family Linux Benchmark v1.0.0** significa que o SCA está a aplicar as políticas de seguridade definidas polo **CIS Benchmark para sistemas Debian**, avaliando cousas como permisos de ficheiros, configuración de SSH, usuarios, auditoría, etc., para determinar se o sistema cumpre cos requisitos recomendados por este estándar de seguridade recoñecido internacionalmente.



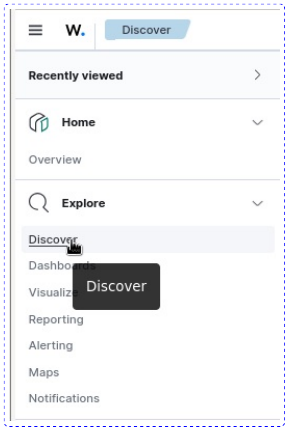

Configuration A...
debian-axente1
Dashboard
Inventory
Events
debian-axente1 (001)
CENTER FOR INTERNET SECURITY DEBIAN FAMILY LIN...
Passed (43)
Failed (112)
Not applicable (26)
Policy
Center for Internet Security Debian Family Linux Benchmark v1.0.0
Rows per page: 15
Center for Internet Security Debian Family Linux Benchmark v1.0.0
Passed 43
Failed 112
Not applicable 26
Score 27%
End scan Apr 14, 2025 @ 11:16:59.000
Checks (181)
Refresh
Export formatted
Search
WQL
ID
Title
Target
Result
33000 Ensure mounting of cra... Command: modprobe -n -v cramfs Failed
33001 Ensure mounting of free... Command: modprobe -n -v freevxfs Failed
33002 Ensure mounting of jffs... Command: modprobe -n -v jffs2 Failed
33003 Ensure mounting of hfs ... Command: modprobe -n -v hfs Failed
33004 Ensure mounting of hfs... Command: modprobe -n -v hfsplus Failed
33005 Ensure mounting of squ... Command: modprobe -n -v squashfs Failed
33006 Ensure mounting of udf ... Command: modprobe -n -v udf Failed
33007 Ensure /tmp is configur... File: /etc/fstab Failed
33008 Ensure nodev option se... Command: mount Passed
33009 Ensure nosuid option se... Command: mount Passed
Rows per page: 10
1 2 3 4 5 ... 19


Configuration A...
debian-axente1
Dashboard
Inventory
Events
debian-axente1 (001)
CENTER FOR INTERNET SECURITY DEBIAN FAMILY LIN...
Passed (43)
Failed (112)
Not applicable (26)
Policy
Center for Internet Security Debian Family Linux Benchmark v1.0.0
Rows per page: 15
Center for Internet Security Debian Family Linux Benchmark v1.0.0
Passed 43
Failed 112
Not applicable 26
Score 27%
End scan Apr 14, 2025 @ 11:16:59.000
Checks (181)
Refresh
Export formatted
Search
WQL
ID
Title
Target
Result
33000 Ensure mounting of cra... Command: modprobe -n -v cramfs Failed
Rationale
Removing support for unneeded filesystem types reduces the local attack surface of the server. If this filesystem type is not needed, disable it.
Remediation
Edit or create a file in the /etc/modprobe.d/ directory ending in .conf Example: vim /etc/modprobe.d/cramfs.conf and add the following line: install cramfs /bin/true Run the following command to unload the cramfs module: # rmmod cramfs.
Description
The cramfs filesystem type is a compressed read-only Linux filesystem embedded in small footprint systems. A cramfs image can be used without having to first decompress the image.
Checks (Condition: all)
c:modprobe -n -v cramfs → r:install /bin/true
not c:lsmod → r:cramfs
Compliance
cis: 1.1.1.1
cis_csc_v7: 5.1
cmmc_v2.0: AC.1.002,CM.2.061,SC.3.180
iso_27001-2013: A.8.1.3,A.14.2.5
mitre_techniques:
T1110,T1003,T1081,T1097,T1178,T1072,T1067,T1495,T1019,T1177,T1485,T1486,T1491,T1488,

F. **Discover**

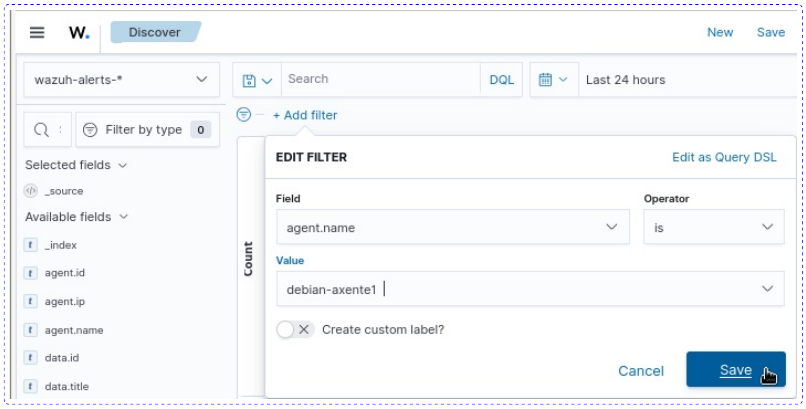
Wazuh **4.9.2** mostra os eventos de seguridade a través da vista de *Discover*. Estes eventos, detectados polos axentes ou polos módulos do manager, inclúen alertas de intrusión, cambios sospeitosos en ficheiros, execucións de procesos non autorizados, vulnerabilidades atopadas, detección de rootkits, entre outros. Baseándose nas regras do motor de análise de Wazuh, preséntanse con distintos niveis de severidade, o que permite ao administrador identificar, investigar e responder rapidamente ante posibles ameazas ou actividades anómalas na infraestrutura monitorizada.

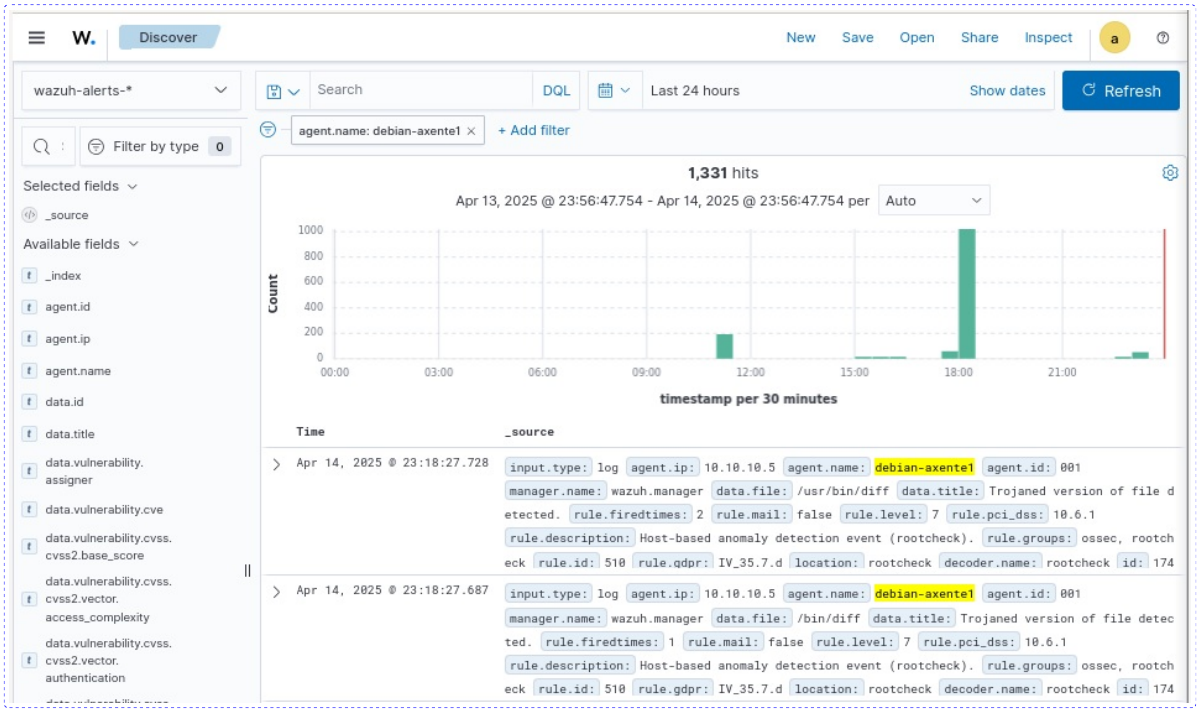
Acceder no dashboard, no menú lateral: *Explore → Discover*



Asegurarse que no buscador da parte superior, está seleccionado o índice: *wazuh-alerts-**. Pódese filtrar na parte de abaixo por: *Axente*, *Severidade (rule.level)*, *Nome da regra (rule.description)*, *ID de alerta*, etc. Así, por exemplo podemos filtrar para ver soamente eventos dun axente específico:

```
agent.name: "debian-axente1"
```





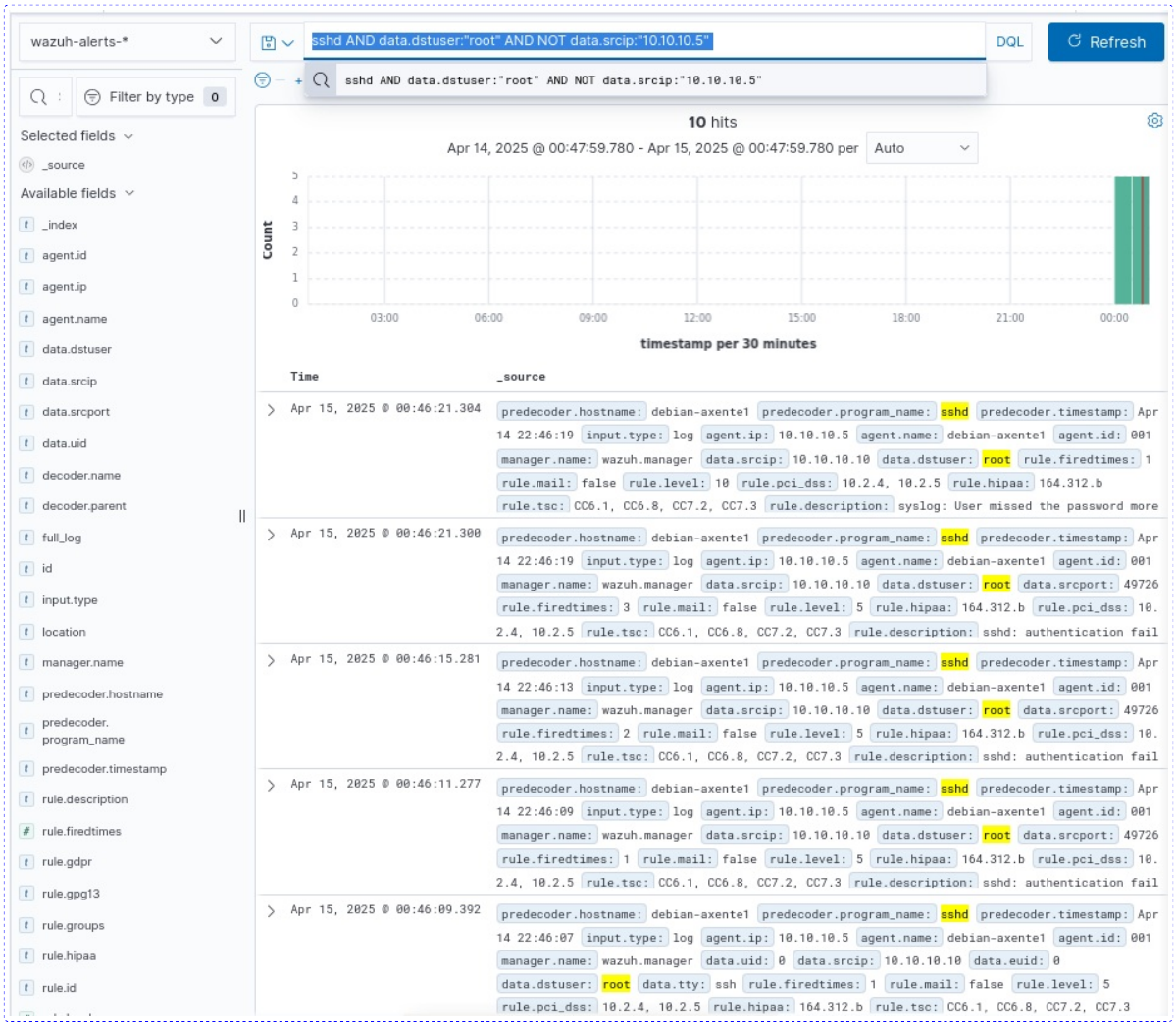
Exemplo: Ver eventos de SSH fallido

1. Acceder ao Dashboard: *https://localhost*
2. Ir ao menú lateral e acceder a *Explore → Discover*
3. Na barra de busca usar o seguinte filtro:

   ```
   sshd AND data.dstuser:"root" AND NOT data.srcip:"10.10.10.5"
   ```

   Explicación:
   - sshd → eventos relacionados co servizo SSH.
   - data.dstuser:"root" → intentos dirixidos ao usuario root
   - AND NOT data.srcip:"10.10.10.5" → exclúe os eventos da IP 10.10.10.5

Este filtro mostra intentos de acceso a root por SSH desde calquera IP agás 10.10.10.5.



Exemplo: Regras de SSH fallido

A. **Wazuh** xa inclúe regras por defecto para SSH fallido.
A organización das regras evolucionou. Wazuh utiliza agora ficheiros con prefixos numéricos para establecer unha orde de carga e agrupación. As regras específicas para SSHD atópanse normalmente no ficheiro:
/var/ossec/ruleset/rules/0095-sshd_rules.xml
Este ficheiro contén as regras predeterminadas para analizar os eventos do daemon SSH (sshd), incluíndo:

- Intentos de inicio de sesión correctos.
- Intentos de inicio de sesión errados (contrasinal incorrecto, usuario inválido). (Regras como 5710, 5712, 5716, 5720 son comúns para isto).
- Erros de conexión.
- Cambios na configuración de SSH.
- Ataques de forza bruta (detectados mediante a correlación de múltiples fallos). (Regras como 5760, 5761).

Como verificalo no teu contedor:
Podes listar os ficheiros nesa carpeta ou buscar directamente o ficheiro:

1. Entra no contedor (se non estás dentro)

   ```
   # docker exec -it single-node_wazuh.manager_1 bash
   ```

2. Dentro do contedor:

   ```
   bash-5.2# find / -type f -iname "*sshd*rules*xml"  2>/dev/null || (ls -l /var/ossec/ruleset/rules/ | grep sshd)
   ```

3. Exemplo de regra:<rule id="5760" level="5"> <if_sid>5700,5716</if_sid> <match>Failed password|Failed keyboard|authentication error</match> <description>sshd: authentication failed.</description> <mitre> <id>T1110.001</id> <id>T1021.004</id> </mitre> <group>authentication_failed,gdpr_IV_35.7.d,gdpr_IV_32.2,gpg13_7.1,hipaa_164.312.b, nist_800_53_AU.14,nist_800_53_AC.7,pci_dss_10.2.4,pci_dss_10.2.5,tsc_CC6.1,tsc_CC6.8, tsc_CC7.2,tsc_CC7.3,</group> </rule>

B. **Axente**

Executar nunha consola de root no axente *debian-axente1*:

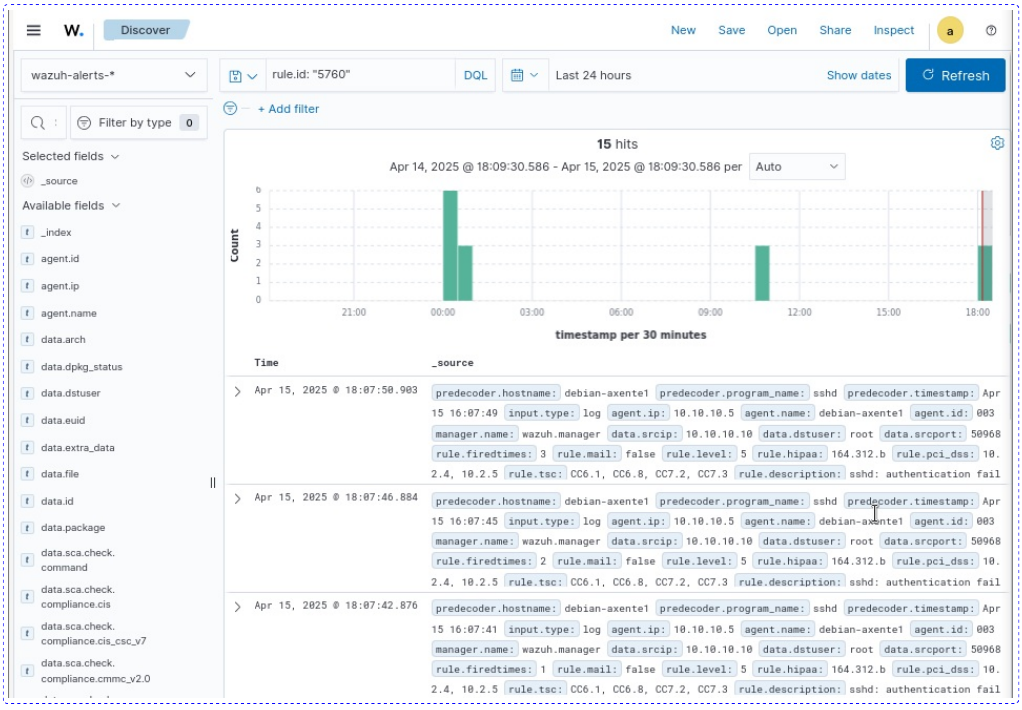
```
# grep -n '</ossec_config>' /var/ossec/etc/ossec.conf
# sed -i '185d' /var/ossec/etc/ossec.conf
# grep -n '<ossec_config>' /var/ossec/etc/ossec.conf
# sed -i '186d' /var/ossec/etc/ossec.conf
# systemctl restart wazuh-agent.service
# systemctl status wazuh-agent.service --no-pager
```

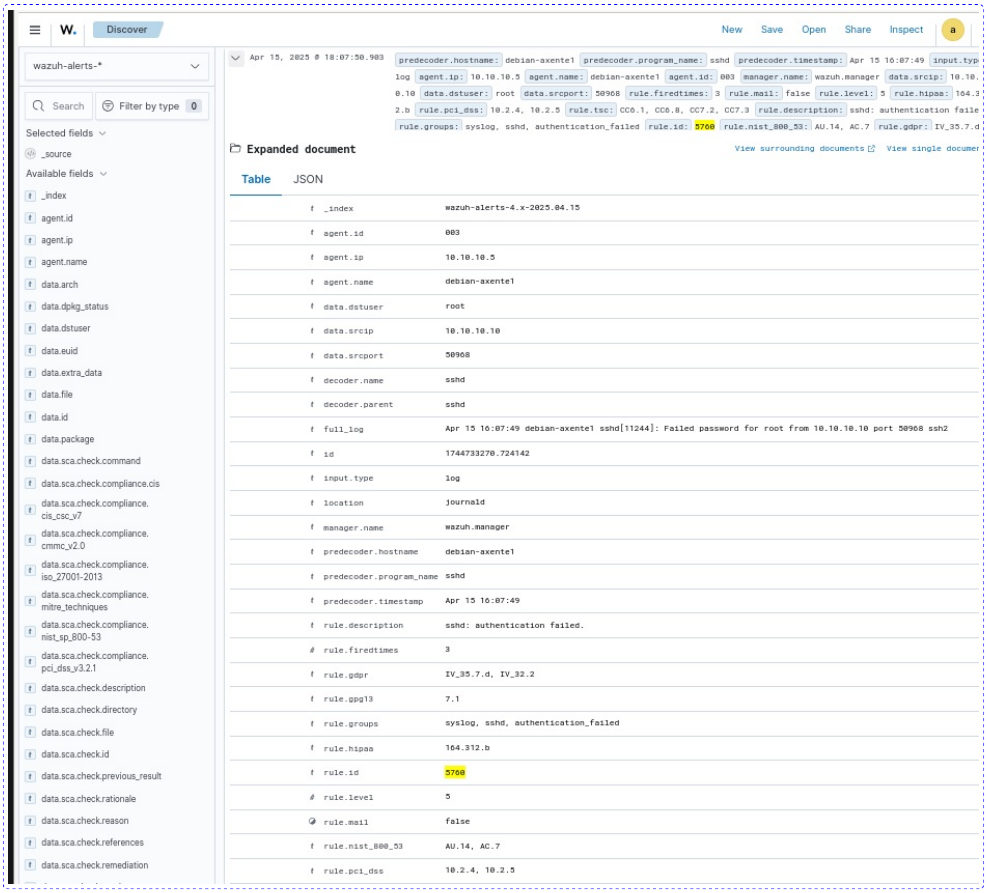
C. **Provocar o fallo de autenticación por SSH**

Na máquina virtual Wazuh executar nunha consola:

```
$ ssh root@10.10.10.5 #Acceder co usuario root mediante SSH ao axente debian-axente1 e provocar o fallo de autenticación SSH (por exemplo introducindo mal o contrasinal, aínda que o usuario root non ten permitido o acceso remoto por SSH no ficheiro de configuración /etc/ssh/sshd_config do axente: directiva PermitRootLogin) root@10.10.10.5's password:
Permission denied, please try again.
root@10.10.10.5's password:
Permission denied, please try again.
root@10.10.10.5's password:
root@10.10.10.5: Permission denied (publickey,password).
```

Verificar o filtro para a regra 5760 de SSH no Dashboard:




Discover
New
Save
Open
Share
Inspect
wazuh-alerts-*
Search
Filter by type 0
Selected fields
_source
Available fields
_index
agent.id
agent.ip
agent.name
data.arch
data.dpkg_status
data.dstuser
data.euid
data.extra_data
data.file
data.id
data.package
data.sca.check.command
data.sca.check.compliance.cis
data.sca.check.compliance.cis_csc_v7
data.sca.check.compliance.cmmc_v2.0
data.sca.check.compliance.iso_27001-2013
data.sca.check.compliance.mitre_techniques
data.sca.check.compliance.nist_sp_800-53
data.sca.check.compliance.pci_dss_v3.2.1
data.sca.check.description
data.sca.check.directory
data.sca.check.file
data.sca.check.id
data.sca.check.previous_result
data.sca.check.rationale
data.sca.check.reason
data.sca.check.references
data.sca.check.remediation
Apr 15, 2025 @ 18:07:50.903
predecoder.hostname: debian-axente1 predecoder.program_name: sshd predecoder.timestamp: Apr 15 16:07:49 input.type: log agent.ip: 10.10.10.5 agent.name: debian-axente1 agent.id: 003 manager.name: wazuh.manager data.srcip: 10.10.10.10 data.dstuser: root data.srcport: 50968 rule.firedtimes: 3 rule.mail: false rule.level: 5 rule.hipaa: 164.312.b rule.pci_dss: 10.2.4, 10.2.5 rule.tsc: CC6.1, CC6.8, CC7.2, CC7.3 rule.description: sshd: authentication failed. rule.groups: syslog, sshd, authentication_failed rule.id: 5760 rule.nist_800_53: AU.14, AC.7 rule.gdpr: IV_35.7.d
Expanded document
View surrounding documents
View single document
Table
JSON
t _index wazuh-alerts-4.x-2025.04.15
t agent.id 003
t agent.ip 10.10.10.5
t agent.name debian-axente1
t data.dstuser root
t data.srcip 10.10.10.10
t data.srcport 50968
t decoder.name sshd
t decoder.parent sshd
t full_log Apr 15 16:07:49 debian-axente1 sshd[11244]: Failed password for root from 10.10.10.10 port 50968 ssh2
t id 1744733270.724142
t input.type log
t location journald
t manager.name wazuh.manager
t predecoder.hostname debian-axente1
t predecoder.program_name sshd
t predecoder.timestamp Apr 15 16:07:49
t rule.description sshd: authentication failed.
rule.firedtimes 3
t rule.gdpr IV_35.7.d, IV_32.2
t rule.gpg13 7.1
t rule.groups syslog, sshd, authentication_failed
t rule.hipaa 164.312.b
t rule.id 5760
rule.level 5
rule.mail false
t rule.nist_800_53 AU.14, AC.7
t rule.pci_dss 10.2.4, 10.2.5

Exemplo: Activar resposta activa (IPS)

Imos configurar Wazuh para que executa automaticamente un script no axente *debian-axente1* ao detectar un posible ataque de Forza Bruta SSH (Regra 5760)

A. Configuración no Xestor (Manager) Wazuh

1. **Editar o ficheiro de configuración principal do xestor:**

   O ficheiro */root/wazuh-docker/single-node/config/wazuh_cluster/wazuh_manager.conf* é o que se carga dentro do contedor docker *single-node_wazuh.manager_1* en */var/ossec/etc/ossec.conf*

   ```
   # cd /root/wazuh-docker/single-node/config/wazuh_cluster/
   # grep -n '</ossec_config>' wazuh_manager.conf
   # sed -i '300d' wazuh_manager.conf
   # grep -n '<ossec_config>' wazuh_manager.conf
   # sed -i '301d' wazuh_manager.conf
   ```

2. **Engadir/Modificar o bloque `<active-response>`**:

   Sustitúe o comentario:

   ```
   <!-- <active-response> active-response options here </active-response> -->
   ```

   por:

   ```
   <!-- ### RESPOSTA ACTIVA PARA BLOQUEO SSH ### -->
   <active-response>
     <disabled>no</disabled>                      <!-- Activa esta resposta específica -->
     <command>script-ips.sh</command>             <!-- Nome do script a enviar -->
     <location>local</location>                   <!-- Executar no axente que enviou o log -->
     <rules_id>5760</rules_id>                    <!-- ID da regra que dispara a resposta -->
     <timeout>600</timeout>                       <!-- Tempo en segundos que durará o bloqueo (10 min) -->
   </active-response>
   ```

   - no: Habilita esta resposta.
   - `script-ips.sh`: Nome do script a enviar.
   - `local`: Executa no axente que orixinou o evento.
   - `5760`: Actívase só coa regra 5760.
   - `600`: Duración do bloqueo en segundos.

3. **Verificar que o comando `script-ips.sh` está definido**. Busca un bloque coma este dentro de *<ossec_config>*:

   ```
   <command>
     <name>script-ips.sh</name>
     <executable>script-ips.sh</executable> <!-- Nome do script esperado no axente -->
     <timeout_allowed>yes</timeout_allowed>  <!-- Indica que acepta timeout -->
   </command>
   ```

   Se non existe, engádeo.

4. **Gardar o ficheiro**

5. **Reiniciar o Xestor Wazuh** para aplicar os cambios:

   ```
   bash-5.2# exit
   # docker restart single-node_wazuh.manager_1
   ```

B. Configuración no Axente (debian-axente1)

O axente necesita ter a Resposta Activa habilitada e saber que script executar.

1. **Verificar o ficheiro de configuración do axente: /var/ossec/etc/ossec.conf**

   Asegúrate de que a Resposta Activa estea habilitada globalmente. Busca o bloque `<active-response>` e verifica o valor *no* en `<disabled>`:

   ```
   <active-response>
     <disabled>no</disabled>
     <!-- ... outras opcións ... -->
   </active-response>
   ```

2. **Verifica que o comando `script-ips.sh` está definido no axente**. Busca un bloque `<command>` dentro de `<ossec_config>` con exactamente este contido (seguramente débase eliminar a liña relativa a *expect*):

   ```
   <command>
     <name>script-ips.sh</name>           <!-- Nome lóxico que chega do xestor -->
     <executable>script-ips.sh</executable> <!-- Script REAL que se executa no axente -->
     <timeout_allowed>yes</timeout_allowed> <!-- Debe ser 'yes' se o xestor envía timeout -->
   </command>
   ```

   Se non existe, engádeo. **Se é necesario gardar o ficheiro**

   Un lugar lóxico para engadir bloques é despois da sección <client_buffer> e antes de seccións como <rootcheck> ou <wodle>. Non ten que ser estritamente aí, pero é un lugar común. O importante é que estea dentro do único bloque <ossec_config> válido.

C. Verificar o Script no Axente

1. **Xerar o script *`/var/ossec/active-response/bin/script-ips.sh`*:**

```
#!/bin/bash

read input
echo "$(date) - Script firewall-drop executado" >> /var/ossec/logs/active-responses.log
echo "$input" >> /var/ossec/logs/active-responses.log

# Extraer comando e IP
CMD=$(echo "$input" | grep -oP '"command"\s*:\s*"\K[^"]+')
IP=$(echo "$input" | grep -oP '"srcip"\s*:\s*"\K[0-9.]+')

TABLE="inet"
CHAIN="filter"
CHAIN_INPUT="input"

if [ "$CMD" = "add" ]; then
    # Só engade se non existe
    nft list chain $TABLE $CHAIN $CHAIN_INPUT | grep -q "ip saddr $IP drop"
    if [ $? -ne 0 ]; then
        nft add rule $TABLE $CHAIN $CHAIN_INPUT ip saddr $IP drop
        echo "$(date) - IP bloqueada: $IP" >> /var/ossec/logs/active-responses.log

        # Eliminar a regra automaticamente despois de 600 segundos
        (
            sleep 600
            HANDLE=$(nft --handle list chain $TABLE $CHAIN $CHAIN_INPUT | grep "ip saddr $IP drop" | awk '{print $NF}')
            if [ -n "$HANDLE" ]; then
                nft delete rule $TABLE $CHAIN $CHAIN_INPUT handle $HANDLE
                echo "$(date) - IP desbloqueada AUTO: $IP (handle $HANDLE)" >> /var/ossec/logs/active-responses.log
            fi
        ) &
    fi

elif [ "$CMD" = "delete" ]; then
    # Elimina regra asociada á IP se chega comando desde Wazuh Manager
    HANDLE=$(nft --handle list chain $TABLE $CHAIN $CHAIN_INPUT | grep "ip saddr $IP drop" | awk '{print $NF}')
    if [ -n "$HANDLE" ]; then
        nft delete rule $TABLE $CHAIN $CHAIN_INPUT handle $HANDLE
        echo "$(date) - IP desbloqueada MANUAL: $IP (handle $HANDLE)" >> /var/ossec/logs/active-responses.log
    fi
fi
```

2. Modificar os permisos:

```
# chmod 750 /var/ossec/active-response/bin/script-ips.sh
# chown root:wazuh /var/ossec/active-response/bin/script-ips.sh
```

D. Reiniciar o Axente Wazuh

Aplicar os cambios no axente:

```
# systemctl restart wazuh-agent
# systemctl status wazuh-agent --no-pager
```

E. Probar e Verificar

1. **Activar o Firewall no Axente (Debian 12 usa nftables):**

```
# nft list table inet filter; [ $? -ne 0 ] && systemctl enable nftables --now
# systemctl status nftables.service --no-pager
# nft list tables
```

2. **Simula un ataque de forza bruta:** Desde unha IP de proba, realiza varios intentos de login SSH errados contra `debian-axente1`

3. **Monitoriza os Logs no axente:**
   - Revisa `/var/ossec/logs/active-responses.log`. Deberías ver a execución de `script-ips.sh` coa acción `add` e a IP atacante.

```
root@debian-axente1:~# tail -f -n2 /var/ossec/logs/active-responses.log
mar 15 abr 2025 23:03:13 CEST - IP bloqueada por script-ips.sh: 10.10.10.10
mar 15 abr 2025 23:03:13 CEST - IP bloqueada por script-ips.sh: 10.10.10.10
```

4. **Verifica o Firewall no Axente (Debian 12 usa nftables):** Busca unha regra que bloquee (`drop`) a IP atacante na táboa `filter`, cadea `input`

```
root@debian-axente1:/var/ossec/active-response/bin# nft list ruleset
table inet filter {
        chain input {
                type filter hook input priority filter; policy accept;
                ip saddr 10.10.10.10 drop
        }

        chain forward {
                type filter hook forward priority filter; policy accept;
        }

        chain output {
                type filter hook output priority filter; policy accept;
        }
}
```

5. **Comproba o Bloqueo:** Intenta conectar por SSH desde a IP atacante; debería fallar.

```
usuario@computer:~$ nc -vz 10.10.10.5 22
10.10.10.5: inverse host lookup failed: Unknown host
(UNKNOWN) [10.10.10.5] 22 (ssh) : Connection timed out
usuario@computer:~$ ssh root@10.10.10.5
ssh: connect to host 10.10.10.5 port 22: Connection timed out
```

6. **Verifica o Desbloqueo Automático:**
   - Agarda a que pase o tempo do `timeout` (10 minutos no exemplo).
   - No axente:
     1. Revisa de novo `nft list ruleset`. A regra de bloqueo debería desaparecer.

```
root@debian-axente1:~# nft list ruleset
table inet filter {
        chain input {
                type filter hook input priority filter; policy accept;
        }

        chain forward {
                type filter hook forward priority filter; policy accept;
        }

        chain output {
                type filter hook output priority filter; policy accept;
        }
}
```

     2. Comproba `/var/ossec/logs/active-responses.log` no axente para ver a acción `delete`.

```
root@debian-axente1:~# tail -f -n2 /var/ossec/logs/active-responses.log
mié 16 abr 2025 00:28:09 CEST - IP bloqueada: 10.10.10.10
mié 16 abr 2025 00:38:09 CEST - IP desbloqueada AUTO: 10.10.10.10 (handle 11)
```

Aviso Importante!

Ten coidado ao probar isto. Asegúrate de non bloquear accidentalmente unha IP que necesites para acceder ao sistema (como a túa propia IP de administración). Usa unha IP de proba específica para simular o ataque.

Ricardo Feijoo Costa