

SQUID + SURICATA + WAZUH

PROTECCION, MONITOREO Y RESPUESTA CON HERRAMIENTAS OPEN SOURCE

Instalación, configuración y
primeros pasos.



Ruben Apablaza Muñoz
-0Zone-

INDICE

Introducción	2
Contexto laboratorio	3
1. Squid	4
1.1 instalacion y configuración básica del servidor squid.....	4
1.2 reglas de squid para restringir el acceso a sitios	6
1.3 informe de los logs de squid.....	10
2. Suricata	15
2.1 instalación y configuración de suricata.	15
2.2 configuración de reglas en suricata.....	18
2.3 verificación de logs en suricata (icmp flood).	20
2.4 verificación de logs en suricata (sitios).	22
3. Wazuh	23
3.1 instalación de wazuh con docker.....	23
3.1.1 configuracion maquina agente.....	34
3.2 reglas para bloqueo de ip sospechosas	36
3.3 informe de wazuh (fuerza bruta)	39
4.squid + suricata + wazuh.....	42
4.1 tipos de cifrados relacionados con servidores proxy.....	42
4.2 software complementario al ids suricata.....	44
4.3 integración squid + suricata + wazuh.....	45

INTRODUCCIÓN

El presente informe detalla la implementación y configuración de un sistema de seguridad en capas sobre una infraestructura de red virtualizada. El objetivo principal de este proyecto es demostrar la sinergia entre diferentes herramientas de seguridad de código abierto para proteger, monitorear y responder a incidentes dentro de un entorno controlado.

Para lograr esto, se ha utilizado **Squid**, un proxy web, para ejercer un control granular sobre el acceso a internet. Este componente permite la gestión de tráfico saliente, bloqueando sitios web específicos y mejorando el rendimiento de la navegación mediante el uso de caché.

En la capa de detección de intrusiones, se ha implementado **Suricata**, un potente sistema IDS/IPS (Intrusion Detection/Prevention System). Suricata se encarga de analizar el tráfico de red en tiempo real, identificando y alertando sobre actividades maliciosas o no deseadas, como conexiones a sitios no autorizados o el uso de protocolos específicos.

Finalmente, **Wazuh** opera como un sistema HIDS (Host-based Intrusion Detection System) y SIEM (Security Information and Event Management). Su función es centralizar los logs y eventos de seguridad tanto del proxy como del sistema de detección de intrusiones, correlacionando la información para proporcionar una visión unificada del estado de la seguridad. Esto permite una respuesta automatizada ante amenazas detectadas, fortaleciendo la postura defensiva del sistema.

A través de este laboratorio, se validará la eficacia de cada herramienta de forma individual y, lo que es más importante, se mostrará cómo su integración conjunta potencia las capacidades de seguridad de la red.

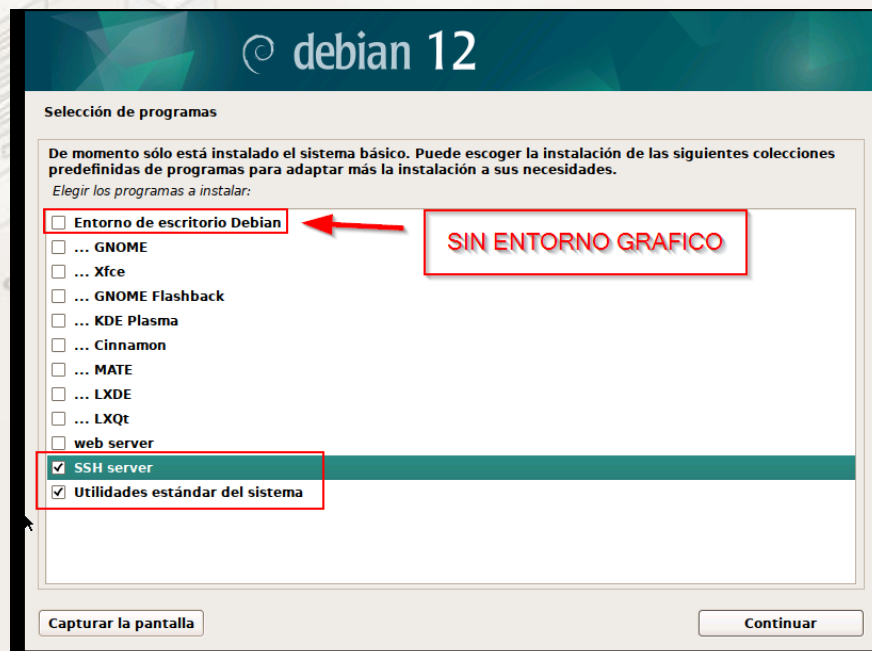
CONTEXTO LABORATORIO

SERVIDOR

- *Maquina Debian 12 instalada con lo mínimo en modo CLI (sin entorno gráfico) y con ssh para su administración desde una terminal en VS Code.*
- *La capacidad de disco duro se configuro en 40GB para evitar tener problemas con Docker.*
- *La máquina está montada en VMWare en modo Bridge.*
- *Se configuró ip estática.*

CLIENTE (maquina Agente)

- *Maquina Debian 12 instalada con lo mínimo en modo CLI (sin entorno gráfico) y con ssh para su administración desde una terminal en VS Code.*
- *La capacidad de disco duro se configuro en 20GB.*
- *La máquina está montada en VMWare en modo Bridge.*
- *Se dejo ip dinámica para efectos del laboratorio.*
- *Instalar apache.*



1. SQUID

1.1 INSTALACION Y CONFIGURACIÓN BÁSICA DEL SERVIDOR SQUID.

- a) Actualizar el sistema (en el caso de que sudo no este instalado se debe instalar o cambiar a usuario root)

apt update && apt upgrade -y

- b) Instalar Squid

apt install squid -y

```
Setcap worked! /usr/lib/squid/pinger is not suid!  
Skipping profile in /etc/apparmor.d/disable: usr.sbin.squid  
Created symlink /etc/systemd/system/multi-user.target.wants/squid.service → /lib/systemd/system/squid.service.  
Procesando disparadores para man-db (2.11.2-2) ...  
Procesando disparadores para libc-bin (2.36-9+deb12u10) ...  
root@Servidor:~#
```

- c) Verificar que el servicio está activo

systemctl status squid

```
root@Servidor:~# systemctl status squid  
● squid.service - Squid Web Proxy Server  
   Loaded: loaded (/lib/systemd/system/squid.service; enabled; preset: enabled)  
   Active: active (running) since Thu 2025-08-21 19:18:35 -04; 48s ago  
     Docs: man:squid(8)  
  Process: 16555 ExecStartPre=/usr/sbin/squid --foreground -z (code=exited, status=0/SUCCESS)  
 Main PID: 16558 (squid)  
    Tasks: 4 (limit: 2255)  
  Memory: 17.5M  
     CPU: 314ms  
   CGroup: /system.slice/squid.service  
           └─16558 /usr/sbin/squid --foreground -sYC  
             └─16561 "(squid-1)" --kid squid-1 --foreground -sYC  
               └─16562 "(logfile-daemon)" /var/log/squid/access.log  
                 └─16563 "(pinger)"
```

d) Hacer backup de la configuración original.

El **respaldo de la configuración inicial de Squid** (/etc/squid/squid.conf) se hace por una razón práctica y de seguridad en la administración de sistemas:

1. Punto de restauración seguro

El archivo squid.conf que viene tras la instalación tiene los valores por defecto, probados y funcionales. Si modificas algo y Squid deja de funcionar, siempre puedes volver a esa copia y recuperar el servicio rápido.

2. Evitar reinstalar

Si rompes la configuración y no tienes backup, podrías pensar en reinstalar Squid para recuperar el archivo original. Con un respaldo, simplemente restauras con:

```
cp /etc/squid/squid.conf.backup /etc/squid/squid.conf
```

3. Documentación y comparación

Tener la copia original te permite hacer un **diff** y comparar qué cambios hiciste respecto a la configuración inicial:

```
diff /etc/squid/squid.conf.backup /etc/squid/squid.conf
```

4. Buenas prácticas de administración

En Linux, siempre se recomienda respaldar configuraciones antes de editarlas (nginx, apache, squid, ssh, etc.).

Entonces hacemos el respaldo de la configuración inicial con

```
cp /etc/squid/squid.conf /etc/squid/squid.conf.backup
```

```
root@Servidor:~# cp /etc/squid/squid.conf /etc/squid/squid.conf.backup
root@Servidor:~#
```

1.2 REGLAS DE SQUID PARA RESTRINGIR EL ACCESO A SITIOS

www.latercera.com

www.yahoo.com

www.yahoo.es

a) Detendremos el servicio de squid

systemctl stop squid

b) Borraremos (es más sencillo ya que el archivo de squid tiene más de 9000 líneas)

rm /etc/squid/squid.conf

y crearemos el archivo de configuración de Squid nuevamente con:

nano /etc/squid/squid.conf

y agregaremos las siguientes reglas al archivo:

CONFIGURACIÓN SQUID - BLOQUEO SELECTIVO

http_port 3128

visible_hostname servidor-proxy

cache_dir ufs /var/spool/squid 100 16 256

ACL: Red local

acl mi_red src 192.168.0.0/24

ACL: Puertos seguros

acl SSL_ports port 443

acl Safe_ports port 80

acl Safe_ports port 443

acl Safe_ports port 1025-65535

ACL: BLOQUEO POR URL

acl sitios_bloqueados url_regex -i yahoo latercera

acl dominios_bloqueados dstdomain .yahoo.com .yahoo.es .latercera.com

REGLAS DE ACCESO - ORDEN CORRECTO

1. Denegar puertos no seguros

http_access deny !Safe_ports

http_access deny CONNECT !SSL_ports

2. BLOQUEO DE SITIOS PROHIBIDOS

http_access deny sitios_bloqueados

http_access deny dominios_bloqueados

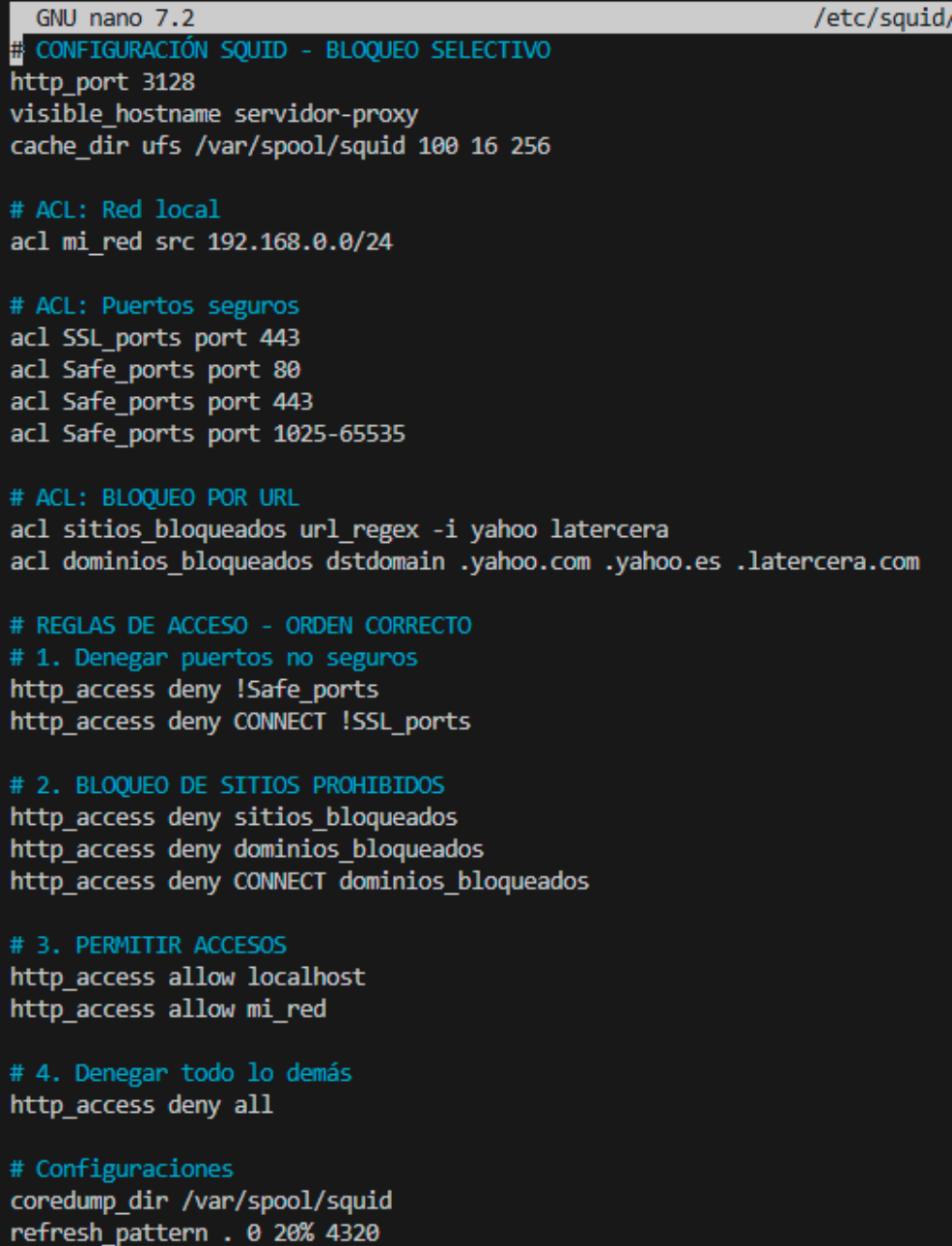
http_access deny CONNECT dominios_bloqueados

3. PERMITIR ACCESOS

http_access allow localhost

```
http_access allow mi_red
# 4. Denegar todo lo demás
http_access deny all
```

```
# Configuraciones
coredump_dir /var/spool/squid
refresh_pattern . 0 20% 4320
```



```
GNU nano 7.2 /etc/squid/
# CONFIGURACIÓN SQUID - BLOQUEO SELECTIVO
http_port 3128
visible_hostname servidor-proxy
cache_dir ufs /var/spool/squid 100 16 256

# ACL: Red local
acl mi_red src 192.168.0.0/24

# ACL: Puertos seguros
acl SSL_ports port 443
acl Safe_ports port 80
acl Safe_ports port 443
acl Safe_ports port 1025-65535

# ACL: BLOQUEO POR URL
acl sitios_bloqueados url_regex -i yahoo latercera
acl dominios_bloqueados dstdomain .yahoo.com .yahoo.es .latercera.com

# REGLAS DE ACCESO - ORDEN CORRECTO
# 1. Denegar puertos no seguros
http_access deny !Safe_ports
http_access deny CONNECT !SSL_ports

# 2. BLOQUEO DE SITIOS PROHIBIDOS
http_access deny sitios_bloqueados
http_access deny dominios_bloqueados
http_access deny CONNECT dominios_bloqueados

# 3. PERMITIR ACCESOS
http_access allow localhost
http_access allow mi_red

# 4. Denegar todo lo demás
http_access deny all

# Configuraciones
coredump_dir /var/spool/squid
refresh_pattern . 0 20% 4320
```

Guardamos el archivo.

c) Reiniciar squid

1. Verificar sintaxis de la configuración

squid -k parse

```
root@Servidor:~# squid -k parse
2025/08/22 09:49:59| Startup: Initializing Authentication Schemes ...
2025/08/22 09:49:59| Startup: Initialized Authentication Scheme 'basic'
2025/08/22 09:49:59| Startup: Initialized Authentication Scheme 'digest'
2025/08/22 09:49:59| Startup: Initialized Authentication Scheme 'negotiate'
2025/08/22 09:49:59| Startup: Initialized Authentication Scheme 'ntlm'
2025/08/22 09:49:59| Startup: Initialized Authentication.
2025/08/22 09:49:59| Processing Configuration File: /etc/squid/squid.conf (depth 0)
2025/08/22 09:49:59| Processing: http_port 3128
2025/08/22 09:49:59| Processing: visible_hostname servidor-proxy
2025/08/22 09:49:59| Processing: cache_dir ufs /var/spool/squid 100 16 256
2025/08/22 09:49:59| Processing: acl mi_red src 192.168.0.0/24
2025/08/22 09:49:59| Processing: acl SSL_ports port 443
2025/08/22 09:49:59| Processing: acl Safe_ports port 80
2025/08/22 09:49:59| Processing: acl Safe_ports port 443
2025/08/22 09:49:59| Processing: acl Safe_ports port 1025-65535
2025/08/22 09:49:59| Processing: acl sitios_bloqueados url_regex -i ".*yahoo.*|.latercera.*"
2025/08/22 09:49:59| ERROR: Can not open file .yahoo.*|.latercera.* for reading
2025/08/22 09:49:59| Warning: empty ACL: acl sitios_bloqueados url_regex -i ".*yahoo.*|.latercera.*"
2025/08/22 09:49:59| Processing: acl dominios_bloqueados dstdomain .yahoo.com .yahoo.es .latercera.com
2025/08/22 09:49:59| Processing: http_access deny !Safe_ports
2025/08/22 09:49:59| Processing: http_access deny CONNECT !SSL_ports
2025/08/22 09:49:59| Processing: http_access deny sitios_bloqueados
2025/08/22 09:49:59| Processing: http_access deny dominios_bloqueados
2025/08/22 09:49:59| Processing: http_access deny CONNECT dominios_bloqueados
2025/08/22 09:49:59| Processing: http_access allow localhost
2025/08/22 09:49:59| Processing: http_access allow mi_red
2025/08/22 09:49:59| Processing: http_access deny all
2025/08/22 09:49:59| Processing: coredump_dir /var/spool/squid
2025/08/22 09:49:59| Processing: refresh_pattern . 0 20% 4320
2025/08/22 09:49:59| Initializing https:// proxy context
root@Servidor:~#
```

2. Reiniciar el servicio Squid para aplicar la nueva configuración:

systemctl restart squid

3. Verificar que el servicio esté funcionando:

systemctl status squid

Debería mostrar active (running).

```
root@Servidor:~# systemctl restart squid
root@Servidor:~# systemctl status squid
● squid.service - Squid Web Proxy Server
   Loaded: loaded (/lib/systemd/system/squid.service; enabled; preset: enabled)
   Active: active (running) since Fri 2025-08-22 09:51:24 -04; 1min 41s ago
     Docs: man:squid(8)
  Process: 1185 ExecStartPre=/usr/sbin/squid --foreground -z (code=exited, status=0/SUCCESS)
 Main PID: 1189 (squid)
    Tasks: 5 (limit: 2255)
   Memory: 17.8M
      CPU: 110ms
   CGroup: /system.slice/squid.service
           └─1189 /usr/sbin/squid --foreground -sYC
             └─1191 "(squid-1)" --kid squid-1 --foreground -sYC
               └─1192 "(logfile-daemon)" /var/log/squid/access.log
                 └─1193 "(unlinkd)"
                   └─1194 "(pinger)"

ago 22 09:51:24 Servidor squid[1191]:      0 Objects expired.
ago 22 09:51:24 Servidor squid[1191]:      0 Objects cancelled.
ago 22 09:51:24 Servidor squid[1191]:      0 Duplicate URLs purged.
ago 22 09:51:24 Servidor squid[1191]:      0 Swapfile clashes avoided.
ago 22 09:51:24 Servidor squid[1191]: Took 0.01 seconds ( 0.00 objects/sec).
ago 22 09:51:24 Servidor squid[1191]: Beginning Validation Procedure
ago 22 09:51:24 Servidor squid[1191]: Completed Validation Procedure
ago 22 09:51:24 Servidor squid[1191]: Validated 0 Entries
ago 22 09:51:24 Servidor squid[1191]: store_swap_size = 0.00 KB
ago 22 09:51:25 Servidor squid[1191]: storeLateRelease: released 0 objects
root@Servidor:~#
```

Esta activo y sin advertencias

1.3 INFORME DE LOS LOGS DE SQUID.

Generaremos conexiones a páginas que están prohibidas y revisaremos el informe de los logs de Squid.

** Instalar curl si no está disponible

apt install curl -y

```
root@Servidor:~# sudo apt install curl -y
-bash: sudo: orden no encontrada
root@Servidor:~# apt install curl -y
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
El paquete indicado a continuación se instaló de forma automática y ya no es necesario.
  linux-image-6.1.0-35-amd64
Utilice «apt autoremove» para eliminarlo.
Se instalarán los siguientes paquetes adicionales:
  libcurl4
Se instalarán los siguientes paquetes NUEVOS:
  curl libcurl4
0 actualizados, 2 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
Se necesita descargar 707 kB de archivos.
Se utilizarán 1.361 kB de espacio de disco adicional después de esta operación.
Des:1 http://deb.debian.org/debian bookworm/main amd64 libcurl4 amd64 7.88.1-10+deb12u12 [391 kB]
Des:2 http://deb.debian.org/debian bookworm/main amd64 curl amd64 7.88.1-10+deb12u12 [315 kB]
Descargados 707 kB en 0s (2.186 kB/s)
Seleccionando el paquete libcurl4:amd64 previamente no seleccionado.
(Leyendo la base de datos ... 42065 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar .../libcurl4_7.88.1-10+deb12u12_amd64.deb ...
Desempaquetando libcurl4:amd64 (7.88.1-10+deb12u12) ...
Seleccionando el paquete curl previamente no seleccionado.
Preparando para desempaquetar .../curl_7.88.1-10+deb12u12_amd64.deb ...
Desempaquetando curl (7.88.1-10+deb12u12) ...
Configurando libcurl4:amd64 (7.88.1-10+deb12u12) ...
Configurando curl (7.88.1-10+deb12u12) ...
Procesando disparadores para man-db (2.11.2-2) ...
Procesando disparadores para libc-bin (2.36-9+deb12u10) ...
root@Servidor:~#
```

Y ahora comprobamos que está instalado con

which curl

```
root@Servidor:~# which curl
/usr/bin/curl
root@Servidor:~#
```

a. Acceso a sitios

a.1 Intentar acceder a sitios bloqueados (deberían fallar)

```
curl -x http://localhost:3128 http://www.latercera.com
```

```
curl -x http://localhost:3128 http://www.yahoo.com
```

```
curl -x http://localhost:3128 http://www.yahoo.es
```

```
<hr>
<div id="footer">
<p>Generated Fri, 22 Aug 2025 13:53:49 GMT by servidor-proxy (squid/5.7)</p>
<!-- ERR_ACCESS_DENIED -->
</div>
</body></html>
root@Servidor:~#
```

a.2 Intentar acceder a un sitio permitido (debería funcionar)

```
curl -x http://localhost:3128 http://www.google.com
```

```
root@Servidor:~# curl -x http://localhost:3128 http://www.google.com
<!doctype html><html itemscope="" itemtype="http://schema.org/WebPage" lang="es-419"><head><meta content="text/html; charset=UTF-8" http-equiv="Content-Type"><meta content="images/branding/googleg/1x/googleg_standard_color_128dp.png" itemprop="image"><title>Google</title><script nonce="mfTqmYAHys076DcQqMTMw">(function(){var _g={kEI:"mMeoaImwAdCd1sQP4K38wAI",kEXPI:"0,202854,2,3497408,685,435,503816,34845,14112,64701,6397,87927,266577,238458,51586,5230281,11401,456,36812186,25228681,51196,5,87067,18673,60611,6751,23878,9139,4600,328,6225,1117,63048,15049,15634,30376,28336,10901,43309,352,18471,409,5870,1898,5816,5774,24960,2651,16524,6251,35,3420,2827,2711,7946,12107,5683,3604,17772,19280,4217,8,3,5746,3,4910,434,550,1187,3618,1217,1,3467,2,213,5952,1033,1579,715,1534,2268,1323,2992,1411,1339,1037,1981,1915,832,2,4,1,320,1190,3913,667,4102,2413,541,3174,2787,1514,2,6503,10819,211,1157,123,407,183,4,259,1120,31,104,3352,539,520,2168,412,264,400,113,2,36,227,1,1331,2,1282,65,5,368,314,128,42,3359,24,2094,739,4,5,2258,3,359,1618,2,9,1,2038,1166,39,1047,651,2,809,5,1301,531,70,846,25,986,2,101,1115,258,289,1341,2028,421,604,110,3065,14,771,10,1910,432,2125,201,1285,4,4,1276,1572,343,633,238,2936,845,817,203,510,26,960,564,196,607,622,708,513,753,11,269,238,5,169,585,32,2,1458,202,477,782,900,233,3,44,304,162,771,2,4,205,98,493,342,333,7,545,2,66,57,137,332,233,591,578,1146,3,3016,51,656,16,120,337,323,133,752,314,39,61,544,5,1045,214,19,503,1667,48,86,497,2268,2,1,1,6,190,55,225,374,626,27,96,186,149,407,43,914,110,390,241,410,44,67,508,111,1141,863,11,354,1228,515,145,1309,612,59,39,8,2,1931,1283,21169387,5,2992,4,2701,259,3,1308,7183,2,1558,3,2446,3,1927,101,3987,1126,1499,158,1210,96,106,230,160,84,80,971,77,3620502,4864736,28615,2269854',kBL:'S2ct',kOPI:89978449});(function(){var ar a;((a=window.google)==null?0:a.stvsc)?google.kEI=_g.kEI:window.google=_g;}).call(this);})(function(){google.sn="webhp";google.kHL="es-419";})();(function(){var g=this||self;function k(){return window.google&&window.google.kOPI||null;var l,m=[];function n(a){for(var b;a&&(!a.getAttribute||!(b=a.getAttribute("eid"))));a=a.parentNode;return b||l}function p(a){for(var b=null;a&&(!a.getAttribute||!(b=a.getAttribute("leid"))));a=a.parentNode;return b}function q(a){/^http:\/i.test(a)&&window.location.protocol=="https:"&&(google.ml&&google.ml(Error("a"),l,{src:a,gllm:l}),a="");return a}function r(a,b,d,c,h){var e="";b.search("&ei=")===-1&&(e="&ei="+n(c).b.search("&lei=")===-1&&(c=p(c))&&(e="&lei="+c));var f=b.search("&csid=")===-1&&a!=""?slh":c+"&z="+"Date.now().toString();g._csid&&f&&(c="&csid="+g._csid);(d=d())&&(c="&opi="+d);return"/"+(h||"gen_204")+ "?atyp=i&ct="+String(a)+"&ad="+String(b)+"(b+e+c)";l=google.kEI;google.getEI=n;google.getEI=p;google.ml=function(){return null};google.log=function(a,b,d,c,h,e){e===void 0?k:e||(d=r(a,b,e,c,h));if(d=q(d)){a=new Image;var f=m.length;m[f]=a;a.onerror=a.onload=a.onabort=function(){delete m[f]};a.src=d};google.logUrl=function(a,b){b-b===void 0?k:b=return r("",a,b);}).call(this);(function(){google.y={};google.sy={};function e(a,b,c){if(a)var d=a.id;else{do d=Math.random();while(c[d]);c[d]=[a,b]}var f;(f=google).x||(f.x=function(a,b){e(a,b,google.y)});var g;(g=google).sx||(g.sx=function(a,b){e(a,b,google.sy)});google.lm=[];var h;(h=google).plm||(h.plm=function(a){google.lm.push.apply(google.lm,a)});google.lq=[];var k;(k=google).load||(k.load=function(a,b,c){google.lq.push([a,b,c]);var l;(l=google).loadAll||(l.loadAll=function(a,b){google.lq.push([a,b]);google.bx+=1;var m;(m=google).lx||(m.lx=function(){var n=[];p;(p=google).fce||(p.fce=function(a,b,c,d){n.push([a,b,c,d]);google.qce=n;google.adl=[];}).call(this);google.f=};(function(){document.documentElement.addEventListener("submit",function(b){var a;if(a=b.target){var c=a.getAttribute("data-submitfalse");a===!1||c=="q"&&a.elements.q.value?l0:l1}else a=!1;a&&(b.preventDefault(),b.stopPropagation()),l0);document.documentElement.addEventListener("click",function(b){var a;a={for(a=b.target;a&&a!=document.documentElement;a=a.parentElement){if(a.tagName=="A"){a=a.getAttribute("data-nohref")
```


b. Ver los logs de Squid:

b.1 Ver logs en tiempo real

tail -f /var/log/squid/access.log

```
root@Servidor:~# tail -f /var/log/squid/access.log

1755870829.999    0 127.0.0.1 TCP_DENIED/403 3913 GET http://www.latercera.com/ - HIER_NONE/- text/html
1755870930.742    93 127.0.0.1 TCP_MISS/200 18506 GET http://www.google.com/ - HIER_DIRECT/142.251.0.99 text/html
1755871015.220    0 127.0.0.1 TCP_DENIED/403 3901 GET http://www.yahoo.com/ - HIER_NONE/- text/html
1755871028.770    0 127.0.0.1 TCP_DENIED/403 3898 GET http://www.yahoo.es/ - HIER_NONE/- text/html
1755871071.885    64 127.0.0.1 TCP_MISS/301 520 GET http://www.youtube.com/ - HIER_DIRECT/64.233.186.93 application/binary
1755871106.521    173 127.0.0.1 TCP_TUNNEL/200 691335 CONNECT www.youtube.com:443 - HIER_DIRECT/64.233.186.93 -
```

b.2 Ver logs específicos de denegaciones

grep DENIED /var/log/squid/access.log

```
root@Servidor:~# grep DENIED /var/log/squid/access.log
1755870829.999    0 127.0.0.1 TCP_DENIED/403 3913 GET http://www.latercera.com/ - HIER_NONE/- text/html
1755871015.220    0 127.0.0.1 TCP_DENIED/403 3901 GET http://www.yahoo.com/ - HIER_NONE/- text/html
1755871028.770    0 127.0.0.1 TCP_DENIED/403 3898 GET http://www.yahoo.es/ - HIER_NONE/- text/html
root@Servidor:~#
```

b.3 Ver logs completos de las últimas conexiones

tail -n 20 /var/log/squid/access.log

```
root@Servidor:~# tail -n 20 /var/log/squid/access.log

1755870829.999    0 127.0.0.1 TCP_DENIED/403 3913 GET http://www.latercera.com/ - HIER_NONE/- text/html
1755870930.742    93 127.0.0.1 TCP_MISS/200 18506 GET http://www.google.com/ - HIER_DIRECT/142.251.0.99 text/html
1755871015.220    0 127.0.0.1 TCP_DENIED/403 3901 GET http://www.yahoo.com/ - HIER_NONE/- text/html
1755871028.770    0 127.0.0.1 TCP_DENIED/403 3898 GET http://www.yahoo.es/ - HIER_NONE/- text/html
1755871071.885    64 127.0.0.1 TCP_MISS/301 520 GET http://www.youtube.com/ - HIER_DIRECT/64.233.186.93 application/binary
1755871106.521    173 127.0.0.1 TCP_TUNNEL/200 691335 CONNECT www.youtube.com:443 - HIER_DIRECT/64.233.186.93 -
root@Servidor:~#
```

c. Generar informe de logs:

Crear informe de accesos bloqueados

c.1 **echo "=== INFORME DE LOGS SQUID ===" > informe_squid.txt**

echo "Fecha de generación: \$(date)" >> informe_squid.txt

echo "" >> informe_squid.txt

c.2 **echo "=== ÚLTIMOS 20 ACCESOS ===" >> informe_squid.txt**

tail -n 20 /var/log/squid/access.log >> informe_squid.txt

echo "" >> informe_squid.txt

```
c.3 echo "=== ACCESOS DENEGADOS RECIENTES ===" >> informe_squid.txt
    grep -i denied /var/log/squid/access.log | tail -n 10 >> informe_squid.txt
    echo "" >> informe_squid.txt
```

```
c.4 echo "=== INTENTOS A SITIOS BLOQUEADOS ===" >> informe_squid.txt
    grep -E '(latercera|yahoo)' /var/log/squid/access.log >> informe_squid.txt
```

```
root@Servidor:~# echo "=== INFORME DE LOGS SQUID ===" > informe_squid.txt
root@Servidor:~# echo "Fecha de generación: $(date)" >> informe_squid.txt
root@Servidor:~# echo "" >> informe_squid.txt
root@Servidor:~# cat informe_squid.txt
=== INFORME DE LOGS SQUID ===
Fecha de generación: vie 22 ago 2025 10:08:10 -04

root@Servidor:~# echo "=== ÚLTIMOS 20 ACCESOS ===" >> informe_squid.txt
root@Servidor:~# tail -n 20 /var/log/squid/access.log >> informe_squid.txt
root@Servidor:~# echo "" >> informe_squid.txt
root@Servidor:~# echo "=== ACCESOS DENEGADOS RECIENTES ===" >> informe_squid.txt
root@Servidor:~# grep -i denied /var/log/squid/access.log | tail -n 10 >> informe_squid.txt
root@Servidor:~# echo "" >> informe_squid.txt
root@Servidor:~# echo "=== INTENTOS A SITIOS BLOQUEADOS ===" >> informe_squid.txt
root@Servidor:~# grep -E '(latercera|yahoo)' /var/log/squid/access.log >> informe_squid.txt
```

d. Mostrar el informe

cat informe_squid.txt

```
root@Servidor:~# cat informe_squid.txt
=== INFORME DE LOGS SQUID ===
Fecha de generación: vie 22 ago 2025 10:08:10 -04

=== ÚLTIMOS 20 ACCESOS ===

1755870829.999    0 127.0.0.1 TCP_DENIED/403 3913 GET http://www.latercera.com/ - HIER_NONE/- text/html
1755870930.742    93 127.0.0.1 TCP_MISS/200 18506 GET http://www.google.com/ - HIER_DIRECT/142.251.0.99 text/html
1755871015.220    0 127.0.0.1 TCP_DENIED/403 3901 GET http://www.yahoo.com/ - HIER_NONE/- text/html
1755871028.770    0 127.0.0.1 TCP_DENIED/403 3898 GET http://www.yahoo.es/ - HIER_NONE/- text/html
1755871071.885    64 127.0.0.1 TCP_MISS/301 520 GET http://www.youtube.com/ - HIER_DIRECT/64.233.186.93 application/binary
1755871106.521    173 127.0.0.1 TCP_TUNNEL/200 691335 CONNECT www.youtube.com:443 - HIER_DIRECT/64.233.186.93 -

=== ACCESOS DENEGADOS RECIENTES ===
1755870829.999    0 127.0.0.1 TCP_DENIED/403 3913 GET http://www.latercera.com/ - HIER_NONE/- text/html
1755871015.220    0 127.0.0.1 TCP_DENIED/403 3901 GET http://www.yahoo.com/ - HIER_NONE/- text/html
1755871028.770    0 127.0.0.1 TCP_DENIED/403 3898 GET http://www.yahoo.es/ - HIER_NONE/- text/html

=== INTENTOS A SITIOS BLOQUEADOS ===
1755870829.999    0 127.0.0.1 TCP_DENIED/403 3913 GET http://www.latercera.com/ - HIER_NONE/- text/html
1755871015.220    0 127.0.0.1 TCP_DENIED/403 3901 GET http://www.yahoo.com/ - HIER_NONE/- text/html
1755871028.770    0 127.0.0.1 TCP_DENIED/403 3898 GET http://www.yahoo.es/ - HIER_NONE/- text/html
root@Servidor:~#
```

e. Verificación final:

e.1 Verificar que el servicio está corriendo

systemctl status squid

e.2 Verificar configuración (si lanza algún error acá es donde se ve)

squid -k check

```
root@Servidor:~# systemctl status squid
● squid.service - Squid Web Proxy Server
   Loaded: loaded (/lib/systemd/system/squid.service; enabled; preset: enabled)
   Active: active (running) since Fri 2025-08-22 10:19:27 -04; 3min 22s ago
     Docs: man:squid(8)
  Process: 1230 ExecStartPre=/usr/sbin/squid --foreground -z (code=exited, status=0/SUCCESS)
    Main PID: 1234 (squid)
      Tasks: 5 (limit: 2255)
     Memory: 18.0M
        CPU: 125ms
   CGroup: /system.slice/squid.service
           └─1234 /usr/sbin/squid --foreground -sYC
             └─1236 "(squid-1)" --kid squid-1 --foreground -sYC
               └─1237 "(logfile-daemon)" /var/log/squid/access.log
                 └─1238 "(unlinkd)"
                   └─1239 "(pinger)"

ago 22 10:19:27 Servidor squid[1236]:      0 Objects expired.
ago 22 10:19:27 Servidor squid[1236]:      0 Objects cancelled.
ago 22 10:19:27 Servidor squid[1236]:      0 Duplicate URLs purged.
ago 22 10:19:27 Servidor squid[1236]:      0 Swapfile clashes avoided.
ago 22 10:19:27 Servidor squid[1236]: Took 0.01 seconds ( 0.00 objects/sec).
ago 22 10:19:27 Servidor squid[1236]: Beginning Validation Procedure
ago 22 10:19:27 Servidor squid[1236]: Completed Validation Procedure
ago 22 10:19:27 Servidor squid[1236]: Validated 0 Entries
ago 22 10:19:27 Servidor squid[1236]: store_swap_size = 0.00 KB
ago 22 10:19:28 Servidor squid[1236]: storeLateRelease: released 0 objects
root@Servidor:~# squid -k check
root@Servidor:~#
```

El informe generado en `informe_squid.txt` mostrará todas las conexiones intentadas, incluyendo las denegadas a los sitios bloqueados y las exitosas a sitios permitidos, demostrando que la configuración de Squid está funcionando correctamente.

```
root@Servidor:~# cat informe_squid.txt
=== INFORME DE LOGS SQUID ===
Fecha de generación: vie 22 ago 2025 10:08:10 -04

=== ÚLTIMOS 20 ACCESOS ===

1755870829.999 0 127.0.0.1 TCP_DENIED/403 3913 GET http://www.latercera.com/ - HIER_NONE/- text/html
1755870930.742 93 127.0.0.1 TCP_MISS/200 18506 GET http://www.google.com/ - HIER_DIRECT/142.251.0.99 text/html
1755871015.220 0 127.0.0.1 TCP_DENIED/403 3901 GET http://www.yahoo.com/ - HIER_NONE/- text/html
1755871028.770 0 127.0.0.1 TCP_DENIED/403 3898 GET http://www.yahoo.es/ - HIER_NONE/- text/html
1755871071.885 64 127.0.0.1 TCP_MISS/301 520 GET http://www.youtube.com/ - HIER_DIRECT/64.233.186.93 application/binary
1755871106.521 173 127.0.0.1 TCP_TUNNEL/200 691335 CONNECT www.youtube.com:443 - HIER_DIRECT/64.233.186.93 -

=== ACCESOS DENEGADOS RECIENTES ===
1755870829.999 0 127.0.0.1 TCP_DENIED/403 3913 GET http://www.latercera.com/ - HIER_NONE/- text/html
1755871015.220 0 127.0.0.1 TCP_DENIED/403 3901 GET http://www.yahoo.com/ - HIER_NONE/- text/html
1755871028.770 0 127.0.0.1 TCP_DENIED/403 3898 GET http://www.yahoo.es/ - HIER_NONE/- text/html

=== INTENTOS A SITIOS BLOQUEADOS ===
1755870829.999 0 127.0.0.1 TCP_DENIED/403 3913 GET http://www.latercera.com/ - HIER_NONE/- text/html
1755871015.220 0 127.0.0.1 TCP_DENIED/403 3901 GET http://www.yahoo.com/ - HIER_NONE/- text/html
1755871028.770 0 127.0.0.1 TCP_DENIED/403 3898 GET http://www.yahoo.es/ - HIER_NONE/- text/html
root@Servidor:~#
```

2. SURICATA

2.1 INSTALACIÓN Y CONFIGURACIÓN DE SURICATA.

a. Primero, actualizamos el sistema e instalamos Suricata:

```
apt update
```

```
apt install -y suricata
```

b. Verificamos la versión instalada:

```
suricata --build-info
```

c. Configuramos Suricata para que funcione correctamente editando el archivo de configuración principal:

```
nano /etc/suricata/suricata.yaml
```

Realizamos las siguientes modificaciones importantes:

c.1. Configurar la interfaz de red (ens33):

```
af-packet:
```

```
- interface: ens33
```

```
cluster-id: 99
```

```
cluster-type: cluster_flow
```

```
defrag: yes
```



```
# Linux high speed capture support
af-packet:
- interface: ens33
  # Number of receive threads. "auto" uses the number of cores
  #threads: auto
  # Default clusterid. AF_PACKET will load balance packets based on flow.
  cluster-id: 99
  # Default AF_PACKET cluster type. AF_PACKET can load balance per flow or per hash.
  # This is only supported for Linux kernel > 3.1
  # possible value are:
  # * cluster_flow: all packets of a given flow are sent to the same socket
  # * cluster_cpu: all packets treated in kernel by a CPU are sent to the same socket
  # * cluster_qm: all packets linked by network card to a RSS queue are sent to the same
  # socket. Requires at least Linux 3.14.
  # * cluster_ebpf: eBPF file load balancing. See doc/userguide/capture-hardware/ebpf-xdp.rst for
  # more info.
  # Recommended modes are cluster_flow on most boxes and cluster_cpu or cluster_qm on system
  # with capture card using RSS (requires cpu affinity tuning and system IRQ tuning)
  cluster-type: cluster_flow
  # In some fragmentation cases, the hash can not be computed. If "defrag" is set
  # to yes, the kernel will do the needed defragmentation before sending the packets.
  defrag: yes
  # To use the ring feature of AF_PACKET, set 'use-mmap' to yes
```

c.2. Configurar las redes locales (ajustar según nuestra red):

vars:

address-groups:

HOME_NET: "[192.168.0.0/24]"

EXTERNAL_NET: "!"\$HOME_NET"

```
vars:
# more specific is better for alert accuracy and performance
address-groups:
  HOME_NET: "[192.168.0.0/24]"
  #HOME_NET: "[192.168.0.0/16]"
  #HOME_NET: "[10.0.0.0/8]"
  #HOME_NET: "[172.16.0.0/12]"
  #HOME_NET: "any"

  EXTERNAL_NET: "!"$HOME_NET"
  #EXTERNAL_NET: "any"
```

c.3. Habilitar la comunidad de reglas emergentes:

rule-files:

- suricata.rules

```
default-rule-path: /etc/suricata/rules

rule-files:
- suricata.rules
```

Guardamos el archivo.

d. Iniciamos y habilitamos el servicio de Suricata:

systemctl enable suricata

systemctl start suricata

```
root@Servidor:~# systemctl enable suricata
Synchronizing state of suricata.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable suricata
root@Servidor:~# systemctl start suricata
root@Servidor:~#
```

e. Verificamos el estado del servicio:

systemctl status suricata

```
root@Servidor:~# systemctl status suricata
● suricata.service - Suricata IDS/IDP daemon
   Loaded: em/suricata.service; enabled; preset: enabled
   Active: man:suricata(8) (ctrl + click) i 2025-08-22 11:56:32 -04; 3min 8s ago
     Docs: man:suricata(8)
           man:suricatasc(8)
           https://suricata-ids.org/docs/
  Process: 1666 ExecStart=/usr/bin/suricata -D --af-packet -c /etc/suricata/suricata.yaml --pidfile /run/suricata.pid (code=exited, stat
Main PID: 1667 (Suricata-Main)
   Tasks: 8 (limit: 2255)
  Memory: 52.9M
     CPU: 1.252s
  CGroup: /system.slice/suricata.service
          └─1667 /usr/bin/suricata -D --af-packet -c /etc/suricata/suricata.yaml --pidfile /run/suricata.pid

ago 22 11:56:32 Servidor systemd[1]: Starting suricata.service - Suricata IDS/IDP daemon...
ago 22 11:56:32 Servidor suricata[1666]: 22/8/2025 -- 11:56:32 - <Notice> - This is Suricata version 6.0.10 RELEASE running in SYSTEM mode
ago 22 11:56:32 Servidor systemd[1]: Started suricata.service - Suricata IDS/IDP daemon.
lines 1-17/17 (END)
```

2.2 CONFIGURACIÓN DE REGLAS EN SURICATA

Creamos reglas personalizadas para detectar conexiones a Facebook, YouTube y protocolo ICMP:

nano /etc/suricata/rules/local.rules

a. Agregamos las siguientes reglas:

Alertar por conexiones TLS/SSL a Facebook (usando SNI)

```
alert tls any any -> any any (msg:"Suspicious Connection - Facebook TLS Detected";  
tls.sni; content:"facebook.com"; nocase; sid:1000001; rev:2;)
```

Alertar por conexiones TLS/SSL a YouTube (usando SNI)

```
alert tls any any -> any any (msg:"Suspicious Connection - YouTube TLS Detected";  
tls.sni; content:"youtube.com"; nocase; sid:1000002; rev:2;)
```

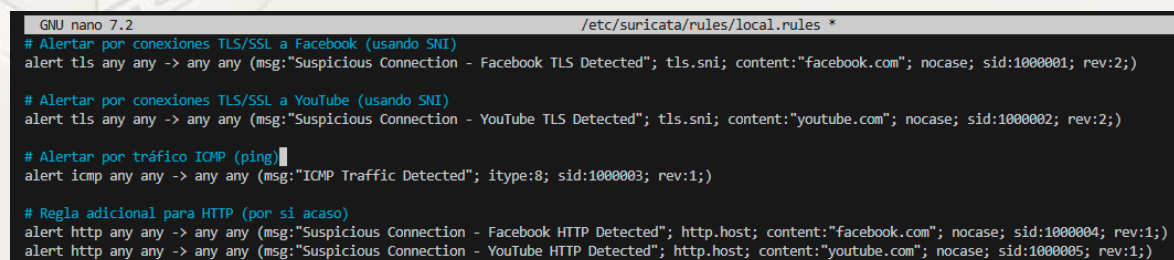
Alertar por tráfico ICMP (ping)

```
alert icmp any any -> any any (msg:"ICMP Traffic Detected"; itype:8; sid:1000003;  
rev:1;)
```

Regla adicional para HTTP (por si acaso)

```
alert http any any -> any any (msg:"Suspicious Connection - Facebook HTTP  
Detected"; http.host; content:"facebook.com"; nocase; sid:1000004; rev:1;)
```

```
alert http any any -> any any (msg:"Suspicious Connection - YouTube HTTP Detected";  
http.host; content:"youtube.com"; nocase; sid:1000005; rev:1;)
```



```
GNU nano 7.2 /etc/suricata/rules/local.rules *  
# Alertar por conexiones TLS/SSL a Facebook (usando SNI)  
alert tls any any -> any any (msg:"Suspicious Connection - Facebook TLS Detected"; tls.sni; content:"facebook.com"; nocase; sid:1000001; rev:2;)  
  
# Alertar por conexiones TLS/SSL a YouTube (usando SNI)  
alert tls any any -> any any (msg:"Suspicious Connection - YouTube TLS Detected"; tls.sni; content:"youtube.com"; nocase; sid:1000002; rev:2;)  
  
# Alertar por tráfico ICMP (ping)  
alert icmp any any -> any any (msg:"ICMP Traffic Detected"; itype:8; sid:1000003; rev:1;)  
  
# Regla adicional para HTTP (por si acaso)  
alert http any any -> any any (msg:"Suspicious Connection - Facebook HTTP Detected"; http.host; content:"facebook.com"; nocase; sid:1000004; rev:1;)  
alert http any any -> any any (msg:"Suspicious Connection - YouTube HTTP Detected"; http.host; content:"youtube.com"; nocase; sid:1000005; rev:1;)
```

b. Actualizamos la configuración para incluir nuestras reglas personalizadas:

nano /etc/suricata/suricata.yaml

Aseguramos que el archivo de reglas local esté incluido:

rule-files:

- **suricata.rules**
- **local.rules**

```
default-rule-path: /etc/suricata/rules

rule-files:
- suricata.rules
- local.rules

##                                     I
## Auxiliary configuration files.
##
```

Guardamos el archivo.

c. Reiniciamos Suricata para aplicar los cambios:

systemctl restart suricata

2.3 VERIFICACIÓN DE LOGS EN SURICATA (ICMP FLOOD).

A continuación veremos como responde suricata a un ping sin fin.

a. Ejecutamos un ping continuo desde otra máquina hacia nuestro servidor:

ping -t 192.168.0.37

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Instale la versión más reciente de PowerShell para obtener nuevas características y mejoras. https://aka.ms/PSWindows

PS C:\Users\ruben> ping -t 192.168.0.37

Haciendo ping a 192.168.0.37 con 32 bytes de datos:
Respuesta desde 192.168.0.37: bytes=32 tiempo<1m TTL=64
Respuesta desde 192.168.0.37: bytes=32 tiempo<1m TTL=64
Respuesta desde 192.168.0.37: bytes=32 tiempo<1m TTL=64
Respuesta desde 192.168.0.37: bytes=32 tiempo<1m TTL=64
Respuesta desde 192.168.0.37: bytes=32 tiempo=1ms TTL=64
Respuesta desde 192.168.0.37: bytes=32 tiempo<1m TTL=64
Respuesta desde 192.168.0.37: bytes=32 tiempo=1ms TTL=64
Respuesta desde 192.168.0.37: bytes=32 tiempo<1m TTL=64
Respuesta desde 192.168.0.37: bytes=32 tiempo<1m TTL=64
Respuesta desde 192.168.0.37: bytes=32 tiempo<1m TTL=64
Respuesta desde 192.168.0.37: bytes=32 tiempo<1m TTL=64
```

b. Verificamos los logs de Suricata para detectar el tráfico ICMP:

tail -f /var/log/suricata/fast.log

```
root@Servidor:~# tail -f /var/log/suricata/fast.log
08/22/2025-12:20:32.854174  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.31:8 -> 192.168.0.37:0
08/22/2025-12:20:33.871522  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.31:8 -> 192.168.0.37:0
08/22/2025-12:20:34.885271  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.31:8 -> 192.168.0.37:0
08/22/2025-12:20:35.899958  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.31:8 -> 192.168.0.37:0
08/22/2025-12:20:36.915380  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.31:8 -> 192.168.0.37:0
08/22/2025-12:20:37.931551  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.31:8 -> 192.168.0.37:0
08/22/2025-12:20:38.948301  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.31:8 -> 192.168.0.37:0
08/22/2025-12:20:39.965123  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.31:8 -> 192.168.0.37:0
08/22/2025-12:20:40.978611  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.31:8 -> 192.168.0.37:0
08/22/2025-12:20:41.991035  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.31:8 -> 192.168.0.37:0
08/22/2025-12:20:43.008670  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.31:8 -> 192.168.0.37:0
08/22/2025-12:20:44.011668  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.31:8 -> 192.168.0.37:0
08/22/2025-12:20:45.014334  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.31:8 -> 192.168.0.37:0
08/22/2025-12:20:46.028690  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.31:8 -> 192.168.0.37:0
```

También podemos revisar el log completo de eventos:

```
tail -f /var/log/suricata/eve.json | grep -E "ICMP|icmp"
```

```
root@Servidor:~# tail -f /var/log/suricata/eve.json | grep -E "ICMP|icmp"
{"timestamp": "2025-08-22T12:21:55.381797-0400", "event_type": "stats", "stats": {"uptime": 304, "capture": {"kernel_packets": 3282, "kernel_drops": 0, "errors": 0}, "decoder": {"pkts": 3282, "bytes": 1268016, "invalid": 0, "ipv4": 1333, "ipv6": 1899, "ethernet": 3282, "chdlc": 0, "raw": 0, "null": 0, "sll": 0, "tcp": 713, "udp": 2237, "sctp": 0, "icmpv4": 52, "icmpv6": 216, "ppp": 0, "pppoe": 0, "geneve": 0, "gre": 0, "vlan": 0, "vlan_qinq": 0, "vxlan": 0, "vntag": 0, "ieee8021ah": 0, "teredo": 0, "ipv4_in_ipv6": 0, "ipv6_in_ipv6": 0, "mpls": 0, "avg_pkt_size": 386, "max_pkt_size": 1514, "max_mac_addr_src": 0, "max_mac_addr_dst": 0, "erspan": 0, "event": {"ipv4": {"pkt_too_small": 0, "hlen_too_small": 0, "iplen_smaller_than_hlen": 0, "trunc_pkt": 0, "opt_invalid": 0, "opt_in_valid_len": 0, "opt_malformed": 0, "opt_pad_required": 14, "opt_eol_required": 0, "opt_duplicate": 0, "opt_unknown": 0, "wrong_ip_version": 0, "icmpv6": 0, "frag_pkt_too_large": 0, "frag_overlap": 0, "frag_ignored": 0}, "icmpv4": {"pkt_too_small": 0, "unknown_type": 0, "unknown_code": 0, "ipv4_trunc_pkt": 0, "ipv4_unknown_ver": 0, "icmpv6": {"unknown_type": 0, "unknown_code": 0, "pkt_too_small": 0, "ipv6_unknown_version": 0, "ipv6_trunc_pkt": 0, "mld_message_with_invalid_hl": 0, "unassigned_type": 0, "experimentation_type": 0}, "ipv6": {"pkt_too_small": 0, "trunc_pkt": 0, "trunc_exthdr": 0, "exthdr_dupl_fh": 0, "exthdr_useless_fh": 0, "exthdr_dupl_rh": 0, "exthdr_dupl_hh": 0, "exthdr_dupl_dh": 0, "exthdr_dupl_eh": 0, "exthdr_invalid_optlen": 0, "wrong_ip_version": 0, "exthdr_ah_res_not_null": 0, "hopopts_unknown_opt": 0, "hopopts_only_padding": 0, "dstopts_unknown_opt": 0, "dstopts_only_padding": 0, "rh_type_0": 0, "zero_len_padn": 21, "fh_non_zero_reserved_field": 0, "data_after_none_header": 0, "unknown_next_header": 0, "icmpv4": 0, "frag_pkt_too_large": 0, "frag_overlap": 0, "frag_invalid_length": 0, "frag_ignored": 0, "ipv4_in_ipv6_too_small": 0, "ipv4_in_ipv6_wrong_version": 0, "ipv6_in_ipv6_too_small": 0, "ipv6_in_ipv6_wrong_version": 0, "tcp": {"pkt_too_small": 0, "hlen_too_small": 0, "invalid_optlen": 0, "opt_invalid_len": 0, "opt_duplicate": 0}, "udp": {"pkt_too_small": 0, "hlen_too_small": 0, "hlen_invalid": 0, "len_invalid": 0}, "sll": {"pkt_too_small": 0}, "ethernet": {"pkt_too_small": 0}, "ppp": {"pkt_too_small": 0, "vju_pkt_too_small": 0, "ip4_pkt_too_small": 0, "ip6_pkt_too_small": 0, "wrong_type": 0, "unsup_proto": 0}, "pppoe": {"pkt_too_small": 0, "wrong_code": 0, "malformed_tags": 0}, "gre": {"pkt_too_small": 0, "wrong_version": 0, "version0_recur": 0, "version0_flags": 0, "version0_hdr_too_big": 0, "version0_malformed_sre_hdr": 0, "version1_chksum": 0, "version1_route": 0, "version1_ssr": 0, "version1_recur": 0, "version1_flags": 0, "version1_no_key": 0, "version1_wrong_protocol": 0, "version1_malformed_sre_hdr": 0, "version1_hdr_too_big": 0}, "vlan": {"header_too_small": 0, "unknown_type": 0, "too_many_layers": 0}, "ieee8021ah": {"header_too_small": 0}, "vntag": {"header_too_small": 0, "unknown_type": 0}, "ipraw": {"invalid_ip_version": 0, "ltnull": {"pkt_too_small": 0, "unsupported_type": 0}, "sctp": {"pkt_too_small": 0}, "mpls": {"header_too_small": 0, "pkt_too_small": 0, "bad_label_router_alert": 0, "bad_label_implicit_null": 0, "bad_label_reserved": 0, "unknown_payload_type": 0, "vxlan": {"unknown_payload_type": 0}, "geneve": {"unknown_payload_type": 0}, "erspan": {"header_too_small": 0, "unsupported_version": 0, "too_many_vlan_layers": 0}, "dce": {"pkt_too_small": 0, "chdlc": {"pkt_too_small": 0}, "too_many_layers": 0, "flow": {"memcap": 0, "tcp": 42, "udp": 61, "icmpv4": 1, "icmpv6": 14, "tcp_reus
```

Deberíamos ver alertas como:

```
12/01/2024-15:30:22.123456  [**] [1:1000003:1] ICMP Traffic Detected [**]
[Classification: (null)] [Priority: 3] {ICMP} 192.168.0.31:8 -> 192.168.0.37:0
```

```
root@Servidor:~# tail -f /var/log/suricata/fast.log
08/22/2025-12:20:43.008670  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.31:8 -> 192.168.0.37:0
08/22/2025-12:20:44.011668  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.31:8 -> 192.168.0.37:0
08/22/2025-12:20:45.014334  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.31:8 -> 192.168.0.37:0
08/22/2025-12:20:46.028690  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.31:8 -> 192.168.0.37:0
08/22/2025-12:20:47.043697  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.31:8 -> 192.168.0.37:0
08/22/2025-12:20:48.057539  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.31:8 -> 192.168.0.37:0
08/22/2025-12:20:49.071836  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.31:8 -> 192.168.0.37:0
08/22/2025-12:20:50.085159  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.31:8 -> 192.168.0.37:0
08/22/2025-12:20:51.098997  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.31:8 -> 192.168.0.37:0
08/22/2025-12:20:52.111771  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.31:8 -> 192.168.0.37:0
```

2.4 VERIFICACIÓN DE LOGS EN SURICATA (SITIOS).

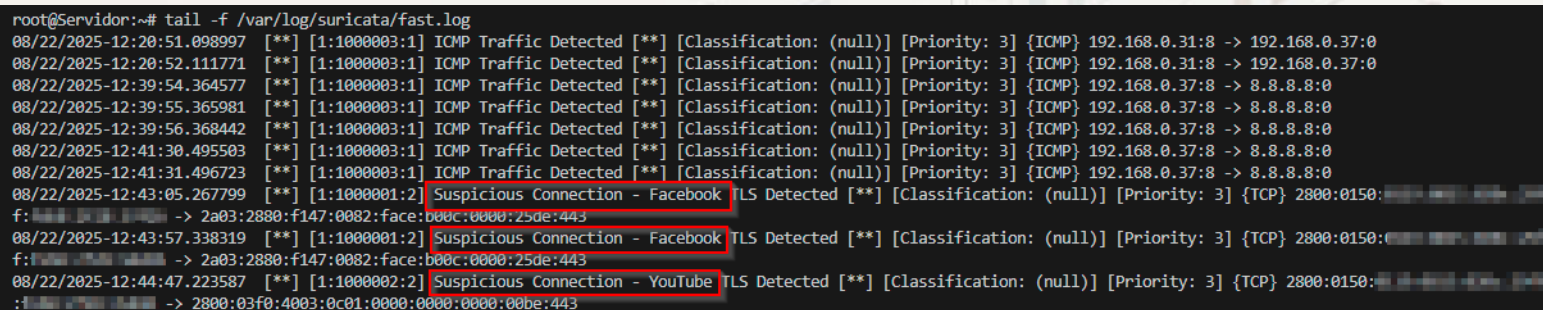
Intentamos acceder a una de las páginas definidas en las reglas (Facebook o YouTube):

a. Usando curl para simular acceso

```
curl -I https://www.facebook.com
```

b. Verificamos los logs de Suricata para detectar el acceso:

```
tail -f /var/log/suricata/fast.log
```



```
root@Servidor:~# tail -f /var/log/suricata/fast.log
08/22/2025-12:20:51.098997  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.31:8 -> 192.168.0.37:0
08/22/2025-12:20:52.111771  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.31:8 -> 192.168.0.37:0
08/22/2025-12:39:54.364577  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.37:8 -> 8.8.8.8:0
08/22/2025-12:39:55.365981  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.37:8 -> 8.8.8.8:0
08/22/2025-12:39:56.368442  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.37:8 -> 8.8.8.8:0
08/22/2025-12:41:30.495503  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.37:8 -> 8.8.8.8:0
08/22/2025-12:41:31.496723  [**] [1:1000003:1] ICMP Traffic Detected [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.37:8 -> 8.8.8.8:0
08/22/2025-12:43:05.267799  [**] [1:1000001:2] Suspicious Connection - Facebook TLS Detected [**] [Classification: (null)] [Priority: 3] {TCP} 2800:0150:
f: 2800:0150: -> 2a03:2880:f147:0082:face:b00c:0000:25de:443
08/22/2025-12:43:57.338319  [**] [1:1000001:2] Suspicious Connection - Facebook TLS Detected [**] [Classification: (null)] [Priority: 3] {TCP} 2800:0150:
f: 2800:0150: -> 2a03:2880:f147:0082:face:b00c:0000:25de:443
08/22/2025-12:44:47.223587  [**] [1:1000002:2] Suspicious Connection - YouTube TLS Detected [**] [Classification: (null)] [Priority: 3] {TCP} 2800:0150:
: 2800:0150: -> 2800:03f0:4003:0c01:0000:0000:0000:00be:443
```

3. WAZUH

3.1 INSTALACIÓN DE WAZUH CON DOCKER

a. Debemos asegurarnos de que todos los paquetes del sistema están actualizados y que git está instalado, ya que lo necesitarás para clonar el repositorio de Wazuh.

apt update && apt install git

Comprobamos que está instalado con

git

```
root@Servidor:~# git
uso: git [-v | --version] [-h | --help] [-C <path>] [-c <name>=<value>]
      [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
      [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
      [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
      [--super-prefix=<path>] [--config-env=<name>=<envvar>]
      <command> [<args>]
```

Estos son comandos comunes de Git usados en varias situaciones:

comenzar un área de trabajo (mira también: `git help tutorial`)

<code>clone</code>	Clonar un repositorio dentro de un nuevo directorio
<code>init</code>	Crear un repositorio de Git vacío o reinicia el que ya existe

trabajar en los cambios actuales (mira también: `git help everyday`)

<code>add</code>	Agregar contenido de archivos al índice
<code>mv</code>	Mover o cambiar el nombre a archivos, directorios o enlaces simbólicos
<code>restore</code>	Restaurar archivos del árbol de trabajo
<code>rm</code>	Borrar archivos del árbol de trabajo y del índice

examinar el historial y el estado (mira también: `git help revisions`)

<code>bisect</code>	Usar la búsqueda binaria para encontrar el commit que introdujo el bug
<code>diff</code>	Mostrar los cambios entre commits, commit y árbol de trabajo, etc

Con estos comandos, el sistema estará actualizado y listo para la siguiente fase.

b. Instalación de Docker

Wazuh en Docker simplifica la gestión de dependencias y asegura que la plataforma pueda ejecutarse de manera aislada y segura. Para instalar Docker, puedes usar el script proporcionado por Docker, que configura todo automáticamente:

```
curl -sSL https://get.docker.com/ | sh
```

OJO. Al instalar Docker nos advierte que si tenemos Docker instalado el script puede dar problemas. Asegurarse que Docker no está instalado en nuestro sistema antes de ejecutar el comando

```
root@Servidor:~# curl -sSL https://get.docker.com/ | sh
# Executing docker install script, commit: bedc5d6b3e782a5e50d3d2a870f5e1f1b5a38d5c
Warning: the "docker" command appears to already exist on this system.

If you already have Docker installed, this script can cause trouble, which is
why we're displaying this warning and provide the opportunity to cancel the
installation.
```

Si se instaló todo correcto, al final deberíamos ver algo como esto.

```
To run Docker as a non-privileged user, consider setting up the
Docker daemon in rootless mode for your user:

    dockerd-rootless-setuptool.sh install

Visit https://docs.docker.com/go/rootless/ to learn about rootless mode.

To run the Docker daemon as a fully privileged service, but granting non-root
users access, refer to https://docs.docker.com/go/daemon-access/

WARNING: Access to the remote API on a privileged Docker daemon is equivalent
to root access on the host. Refer to the 'Docker daemon attack surface'
documentation for details: https://docs.docker.com/go/attack-surface/

=====

root@Servidor:~#
```

Una vez instalado Docker, es esencial que aseguremos su ejecución automática al iniciar el sistema:

systemctl start docker

systemctl enable docker

```
root@Servidor:~# systemctl enable docker
Synchronizing state of docker.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable docker
root@Servidor:~#
```

Estos comandos iniciarán el servicio de Docker y lo configurarán para que se inicie automáticamente en cada arranque del sistema.

OJO: En este caso no es necesario instalar Docker compose debido a que el script anterior ya lo instala. Podemos comprobar con

docker compose version

```
root@Servidor:~# docker compose version
Docker Compose version v2.39.1
root@Servidor:~#
```

c. Antes de instalar wazuh hacemos lo siguiente

Aumentamos **max_map_count** en su host Docker:

sysctl -w vm.max_map_count=262144

para que este valor sea permanente actualizamos el archivo de configuración con

nano /etc/sysctl.conf

y copiamos/pegamos esto al final

vm.max_map_count=262144

```
#####  
# Magic system request Key  
# 0=disable, 1=enable all, >1 bitmask of sysrq functions  
# See https://www.kernel.org/doc/html/latest/admin-guide/sysrq.html  
# for what other values do  
#kernel.sysrq=438  
  
vm.max_map_count=262144
```

Guardamos y salimos.

Para verificarlo después de reiniciar, ejecutamos " ".

sysctl vm.max_map_count

```
root@Servidor:~# sysctl vm.max_map_count  
vm.max_map_count = 262144  
root@Servidor:~#
```

Advertencia: Si no se configura **max_map_count** en su host, el indexador de Wazuh **NO** funcionará correctamente.

d. Configuración del entorno de Wazuh

Lo que sigue está basado en la información de la página oficial de wazuh: <https://documentation.wazuh.com/current/deployment-options/docker/wazuh-container.html>

Con Docker ya configurado, el siguiente paso es preparar el entorno específico para Wazuh. Vamos al directorio opt para mantener organizados los archivos relacionados con la instalación:

cd /opt

Ahora, es el momento de **clonar** la versión más reciente del repositorio de Wazuh para Docker:

git clone https://github.com/wazuh/wazuh-docker.git -b v4.12.0

Este comando descarga todos los archivos necesarios para ejecutar Wazuh en un contenedor Docker.

```
root@Servidor:~# cd /opt
root@Servidor:/opt# git clone https://github.com/wazuh/wazuh-docker.git -b v4.7.3
Clonando en 'wazuh-docker'...
remote: Enumerating objects: 13957, done.
remote: Counting objects: 100% (452/452), done.
remote: Compressing objects: 100% (214/214), done.
remote: Total 13957 (delta 357), reused 253 (delta 238), pack-reused 13505 (from 3)
Recibiendo objetos: 100% (13957/13957), 5.68 MiB | 13.07 MiB/s, listo.
Resolviendo deltas: 100% (7375/7375), listo.
Nota: cambiando a 'c88172dc5651d66789d2c5505abb7fd1ae158828'.
```

Te encuentras en estado 'detached HEAD'. Puedes revisar por aquí, hacer cambios experimentales y hacer commits, y puedes descartar cualquier commit que hayas hecho en este estado sin impactar a tu rama realizando otro checkout.

Si quieres crear una nueva rama para mantener los commits que has creado, puedes hacerlo (ahora o después) usando -c con el comando checkout. Ejemplo:

```
git switch -c <nombre-de-nueva-rama>
```

O deshacer la operación con:

```
git switch -
```

Desactiva este aviso poniendo la variable de config advice.detachedHead en false

```
root@Servidor:/opt#
```

e. Generación de certificados y puesta en marcha de Wazuh

Antes de iniciar Wazuh, debemos generar los certificados necesarios para el correcto funcionamiento de los componentes de Wazuh. Para esto, navegamos al directorio correcto y ejecutamos el generador de certificados:

```
cd wazuh-docker/single-node/
```

```
docker compose -f generate-indexer-certs.yml run --rm generator
```

```
root@Servidor:/opt# cd wazuh-docker/single-node/
root@Servidor:/opt/wazuh-docker/single-node# docker compose -f generate-indexer-certs.yml run --rm generator
```

```

root@Servidor:/opt/wazuh-docker/single-node# docker compose -f generate-indexer-certs.yml run --rm generator
WARN[0000] /opt/wazuh-docker/single-node/generate-indexer-certs.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion

[+] Creating 1/1
  ✓ Network single-node_default Created
[+] Running 5/5
  ✓ generator Pulled
    ✓ edaadc954fb5 Pull complete
    ✓ 573f4d11a520 Pull complete
    ✓ 8f200922197d Pull complete
    ✓ 55a86de68c5c Pull complete
The tool to create the certificates exists in the in Packages bucket
23/08/2025 14:49:18 INFO: Admin certificates created.
23/08/2025 14:49:18 INFO: Wazuh indexer certificates created.
23/08/2025 14:49:18 INFO: Wazuh server certificates created.
23/08/2025 14:49:18 INFO: Wazuh dashboard certificates created.
Moving created certificates to the destination directory
Changing certificate permissions
Setting UID indexer and dashboard
Setting UID for wazuh manager and worker
root@Servidor:/opt/wazuh-docker/single-node#

```

Se genera un warning que podemos ignorar que no afecta la funcionalidad ni la seguridad de la instalación.

Con los certificados generados, ya está todo listo para iniciar todos los servicios de Wazuh:

docker compose up -d

```

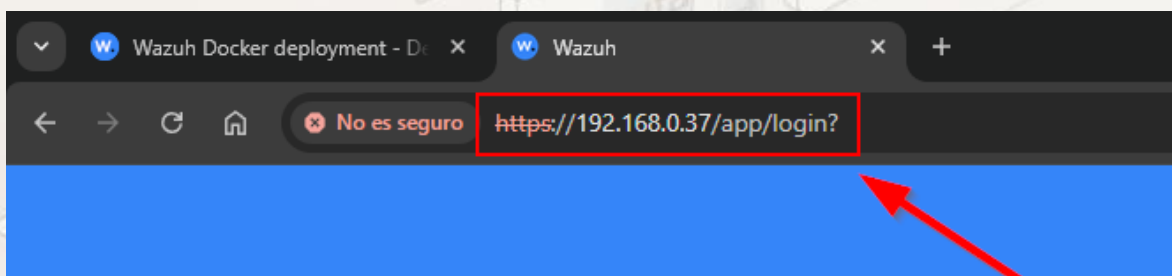
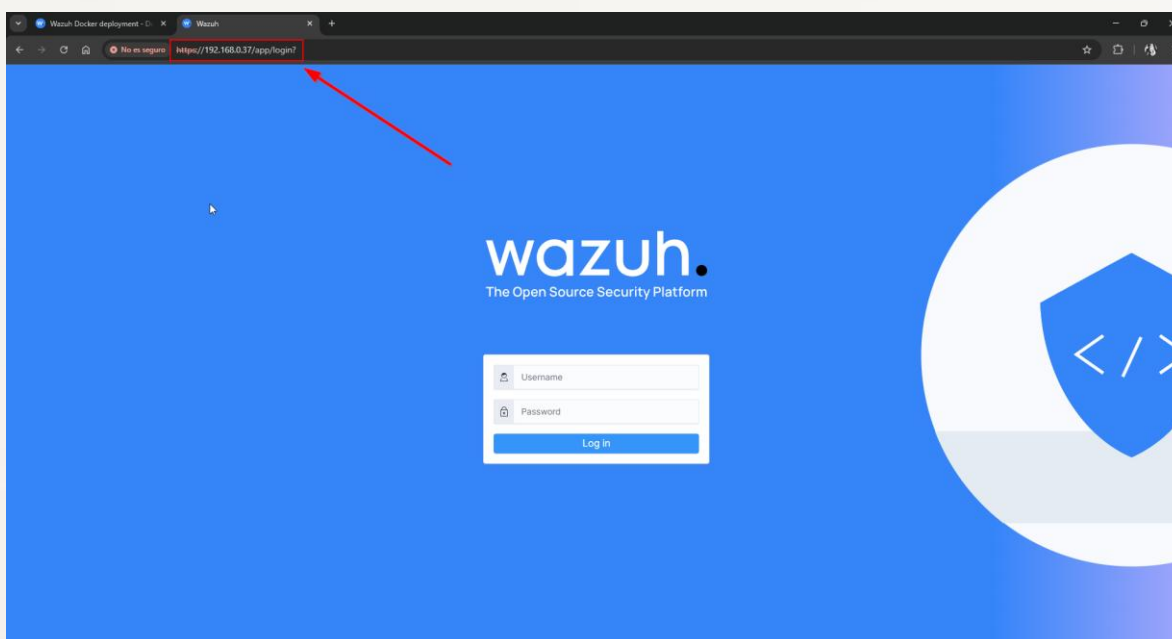
root@Servidor:/opt/wazuh-docker/single-node# docker compose up -d
WARN[0000] /opt/wazuh-docker/single-node/docker-compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion
[+] Running 27/46
  ⚙ wazuh.dashboard [ ] Pulling
    ⚙ 08f439f04b06 Waiting 15.6s
    ⚙ c556c0bfe3c5 Waiting 13.7s
    ⚙ 8611fbc664d5 Waiting 13.7s
    ⚙ 165f93d6ab4e Waiting 13.7s
    ⚙ cde662f149bb Waiting 13.7s
    ⚙ 637e8ae7c08a Waiting 13.7s
    ⚙ 6ca1cb08deda Waiting 13.7s
    ⚙ 47858e95cc25 Waiting 13.7s
    ⚙ d2ebd1c833fc Waiting 13.7s
    ⚙ 1753ff2c0586 Waiting 13.7s
    ⚙ 4f4fb70ef54 Waiting 13.7s
  ⚙ wazuh.manager [ ] 272.2MB / 273.5MB Pulling
    ✓ 8ee087424735 Pull complete 5.5s
    ✓ 8de4cf16db50 Pull complete 5.5s
    ✓ 84adedbb7a4d Pull complete 10.1s
    ✓ 889ee12997be Pull complete 10.1s
    ✓ 00f7b49c0050 Pull complete 10.1s
    ✓ 133228609f50 Pull complete 10.1s
    ⚙ f91f58caff5e Extracting [ ] 49.02MB/168MB 13.7s
    ✓ eb6731937eae Download complete 4.2s
    ✓ bdbaa08d6244 Download complete 4.2s
    ✓ 26b52ed7fc13 Download complete 7.0s
    ✓ 6ef4d89d3023 Download complete 5.6s
    ✓ 5eddd6e55ed6 Download complete 6.7s
    ✓ 1210b2f3d65b Download complete 7.3s

```

Este último comando levanta todos los contenedores necesarios para que Wazuh funcione adecuadamente en un modo de nodo único, ideal para entornos de prueba o pequeñas implementaciones.

f. Ahora podemos acceder desde nuestro explorador a la ip del servidor:

<https://192.168.0.37>



En el cuadro de dialogo ingresamos las credenciales por defecto de wazuh (que podemos cambiar obviamente después)

Username: admin

Password: SecretPassword

The image shows the Wazuh login interface. At the top, the word "wazuh." is written in white on a blue background, followed by the tagline "The Open Source Security Platform". Below this is a white login box. Inside the box, there are two input fields. The first field has a user icon and the text "admin". The second field has a lock icon and a series of dots representing a password. A red rectangle highlights both input fields. Below the fields is a blue button with the text "Log in". The background of the slide features faint illustrations of a server rack labeled "Squid" and a bar chart labeled "Wazuh".

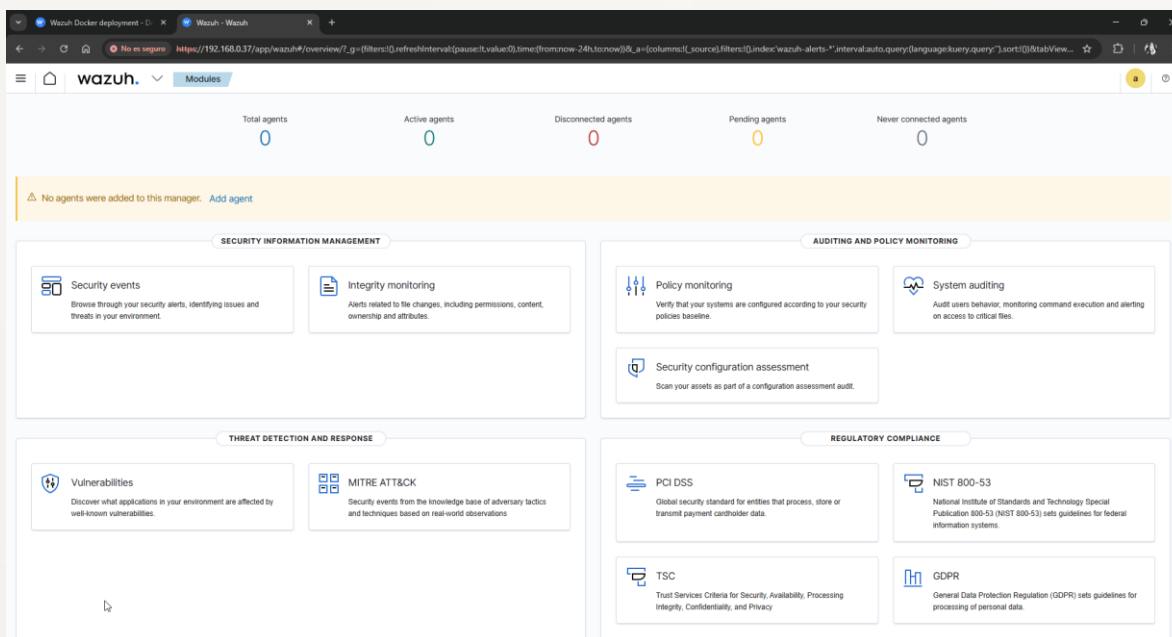
wazuh.
The Open Source Security Platform

admin

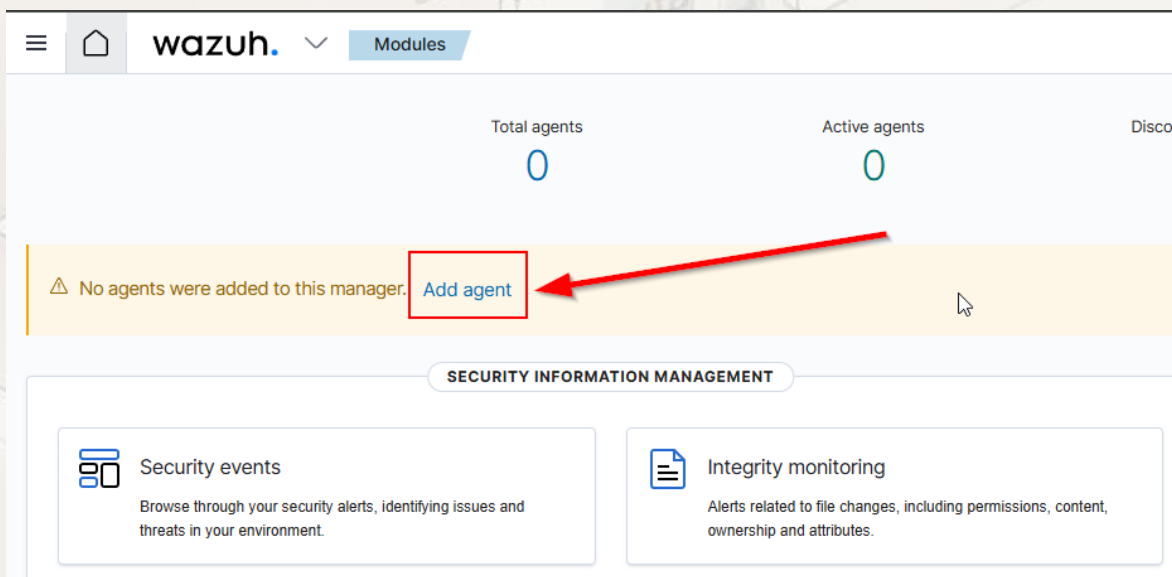
.....

Log in

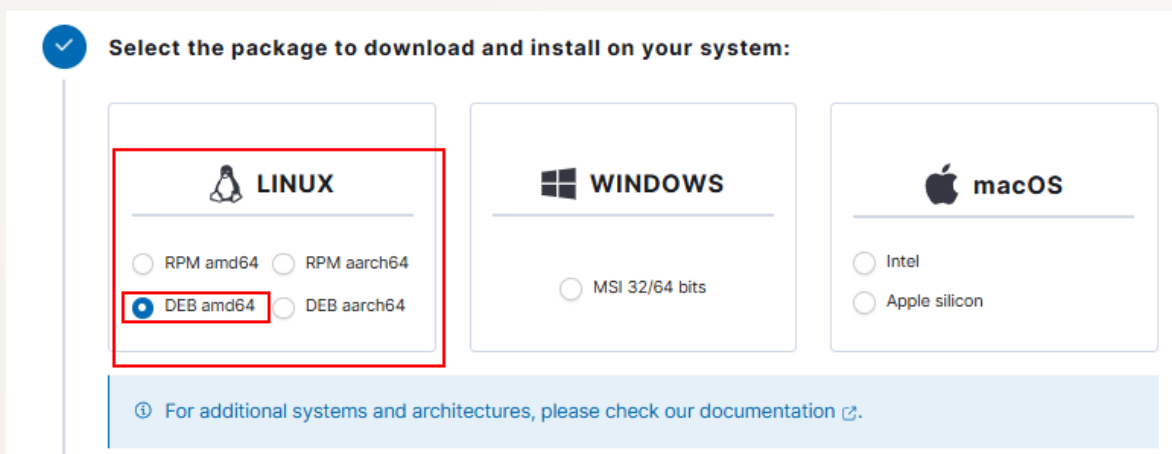
Y ahora tenemos nuestro panel de wazuh desplegado:




Ahora vamos a agregar un agente; para esto hacemos click en





En la página que se abre, seleccionamos la arquitectura correspondiente; en mi caso DEB amd64



✓ **Select the package to download and install on your system:**

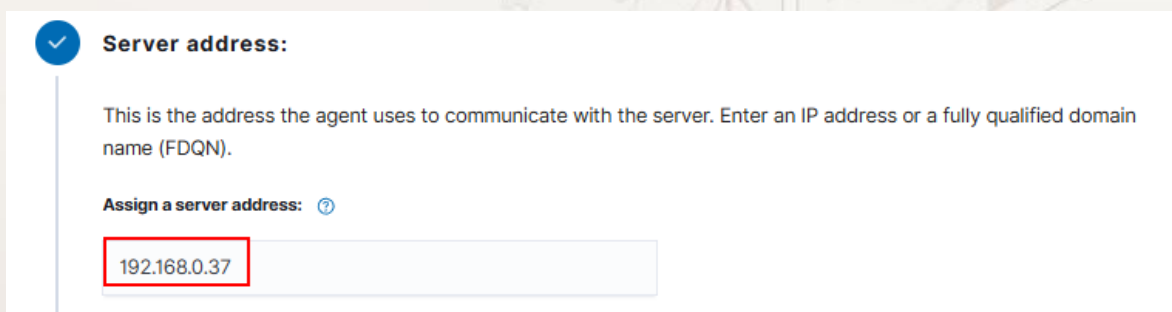
 **LINUX**
☐ RPM amd64 ☐ RPM aarch64
☒ **DEB amd64** ☐ DEB aarch64

 **WINDOWS**
☐ MSI 32/64 bits

 **macOS**
☐ Intel
☐ Apple silicon

[For additional systems and architectures, please check our documentation.](#)

A continuación agregamos la dirección IP del servidor, en mi caso **192.168.0.37**



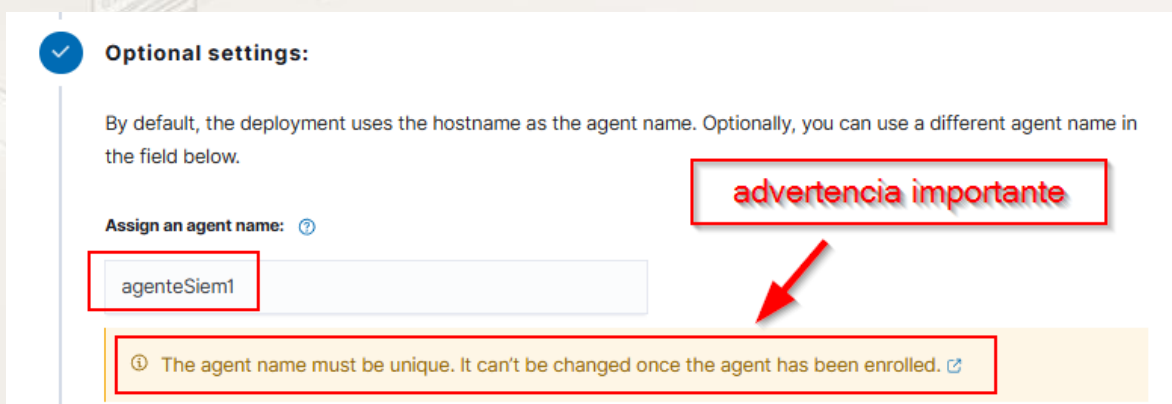
✓ **Server address:**

This is the address the agent uses to communicate with the server. Enter an IP address or a fully qualified domain name (FDQN).

Assign a server address: [?](#)

192.168.0.37

Opcionalmente podemos dar un nombre al agente



✓ **Optional settings:**

By default, the deployment uses the hostname as the agent name. Optionally, you can use a different agent name in the field below.

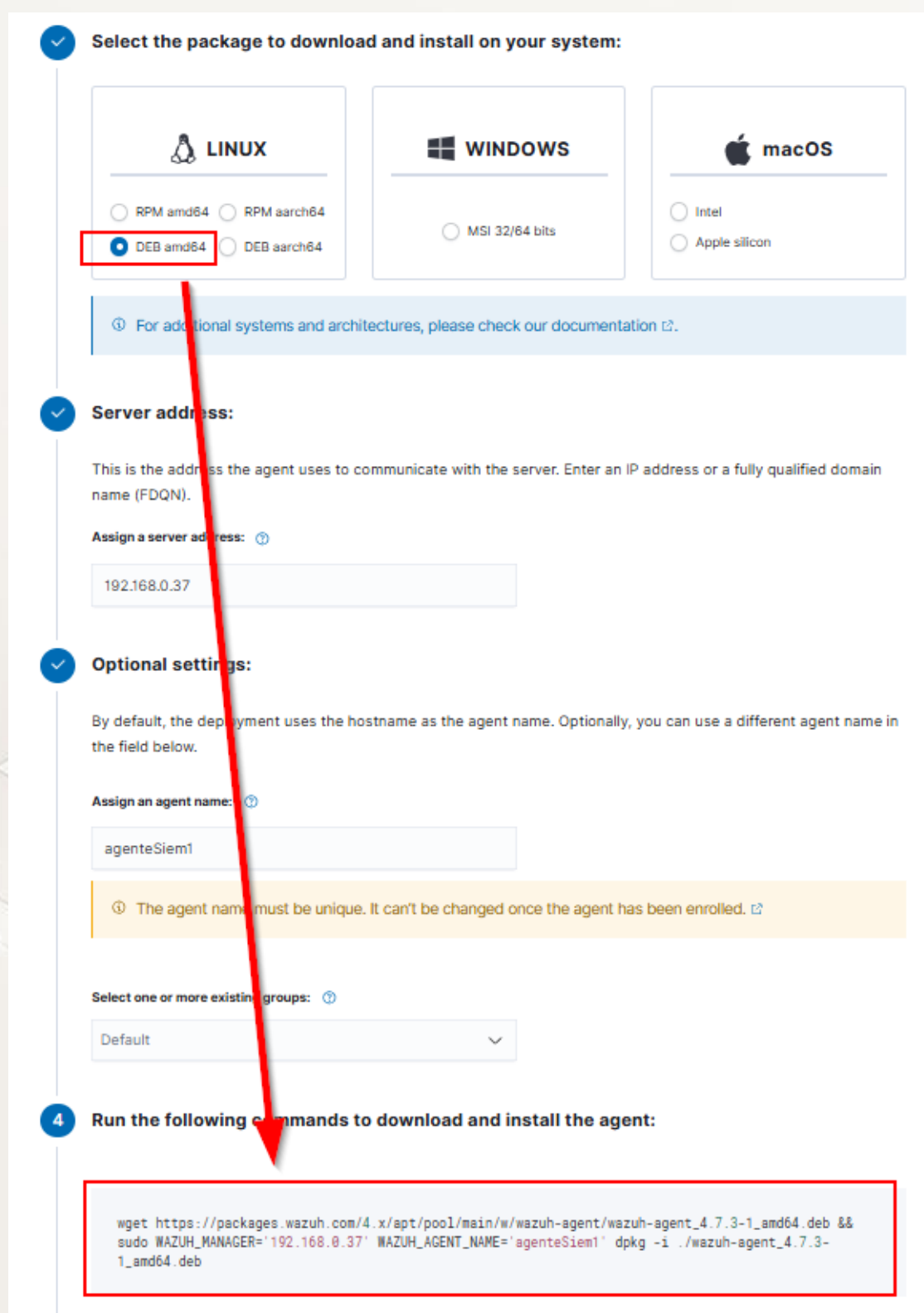
Assign an agent name: [?](#)

agenteSiem1

advertencia importante


The agent name must be unique. It can't be changed once the agent has been enrolled.

Dependiendo de la arquitectura que elijamos, en la parte inferior nos aparece una línea de comando que debemos ejecutar en el cliente.




The image shows a multi-step installation wizard for Wazuh. Step 1, 'Select the package to download and install on your system:', has three main categories: LINUX, WINDOWS, and macOS. Under LINUX, there are four radio button options: RPM amd64, RPM aarch64, DEB amd64 (which is selected and highlighted with a red box), and DEB aarch64. A red arrow points from this selection down to the command box in Step 4. Step 2, 'Server address:', includes a text input field with the IP '192.168.0.37'. Step 3, 'Optional settings:', includes a text input field for 'agentName' with the value 'agenteSiem1' and a dropdown menu for 'Select one or more existing groups' set to 'Default'. Step 4, 'Run the following commands to download and install the agent:', contains a terminal window with a red border showing the installation command.


✓ **Select the package to download and install on your system:**

 **LINUX**

☐ RPM amd64 ☐ RPM aarch64
☒ **DEB amd64** ☐ DEB aarch64

 **WINDOWS**

☐ MSI 32/64 bits

 **macOS**

☐ Intel
☐ Apple silicon

③ For additional systems and architectures, please check our documentation [🔗](#).

✓ **Server address:**

This is the address the agent uses to communicate with the server. Enter an IP address or a fully qualified domain name (FDQN).

Assign a server address: ⓘ

192.168.0.37

✓ **Optional settings:**

By default, the deployment uses the hostname as the agent name. Optionally, you can use a different agent name in the field below.

Assign an agent name: ⓘ

agenteSiem1

③ The agent name must be unique. It can't be changed once the agent has been enrolled. [🔗](#)

Select one or more existing groups: ⓘ

Default

4 **Run the following commands to download and install the agent:**


```
wget https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-agent/wazuh-agent_4.7.3-1_amd64.deb &&  
sudo WAZUH_MANAGER='192.168.0.37' WAZUH_AGENT_NAME='agenteSiem1' dpkg -i ./wazuh-agent_4.7.3-1_amd64.deb
```


3.1.1 CONFIGURACION MAQUINA AGENTE

Importante, antes de ejecutar el comando, debemos tener instalado apache y sudo para ejecutar el comando correctamente.

Copiamos la línea de comando proporcionada y ejecutamos:

```
wget https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-agent/wazuh-agent_4.12.0-1_amd64.deb && sudo WAZUH_MANAGER='192.168.0.37' WAZUH_AGENT_NAME='agenteSiem1' dpkg -i ./wazuh-agent_4.12.0-1_amd64.deb
```



```
root@agenteSiem:~# wget https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-agent/wazuh-agent_4.7.3-1_amd64.deb && sudo WAZUH_MANAGER='192.168.0.37' WAZUH_AGENT_NAME='agenteSiem1' dpkg -i ./wazuh-agent_4.7.3-1_amd64.deb
--2025-08-23 22:09:18-- https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-agent/wazuh-agent_4.7.3-1_amd64.deb
Resolviendo packages.wazuh.com (packages.wazuh.com)... 18.164.32.122, 18.164.32.61, 18.164.32.60, ...
Conectando con packages.wazuh.com (packages.wazuh.com)[18.164.32.122]:443... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 9362524 (8,9M) [application/vnd.debian.binary-package]
Grabando a: «wazuh-agent_4.7.3-1_amd64.deb.1»
wazuh-agent_4.7.3-1_amd64.deb.1 100%[=====>] 8,93M 52,7MB/s en 0,2s

2025-08-23 22:09:18 (52,7 MB/s) - «wazuh-agent_4.7.3-1_amd64.deb.1» guardado [9362524/9362524]

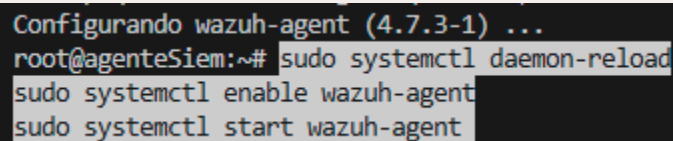
Seleccionando el paquete wazuh-agent previamente no seleccionado.
(Leyendo la base de datos ... 35201 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar .../wazuh-agent_4.7.3-1_amd64.deb ...
Desempaquetando wazuh-agent (4.7.3-1) ...
Configurando wazuh-agent (4.7.3-1) ...
root@agenteSiem:~#
```

Por último iniciamos el agente, copiamos y pegamos los comandos de la ultima parte de la pagina

5

Start the agent:

```
sudo systemctl daemon-reload
sudo systemctl enable wazuh-agent
sudo systemctl start wazuh-agent
```



```
Configurando wazuh-agent (4.7.3-1) ...
root@agenteSiem:~# sudo systemctl daemon-reload
sudo systemctl enable wazuh-agent
sudo systemctl start wazuh-agent
```

Y podemos comprobar que el agente se encuentra activo con

systemctl status wazuh-agent

```
root@agenteSiem:~# systemctl status wazuh-agent
● wazuh-agent.service - Wazuh agent
   Loaded: loaded (/lib/systemd/system/wazuh-agent.service; enabled; preset: enabled)
   Active: active (running) since Sat 2025-08-23 22:17:54 -04; 44s ago
   Process: 3230 ExecStart=/usr/bin/env /var/ossec/bin/wazuh-control start (code=exited, status=0/SUCCESS)
   Tasks: 33 (limit: 2258)
   Memory: 266.1M
   CPU: 12.041s
   CGroup: /system.slice/wazuh-agent.service
           └─3968 /var/ossec/bin/wazuh-execd
             └─3979 /var/ossec/bin/wazuh-agentd
               └─3992 /var/ossec/bin/wazuh-syscheckd
                 └─4017 /var/ossec/bin/wazuh-logcollector
                   └─4066 /var/ossec/bin/wazuh-modulesd

ago 23 22:17:46 agenteSiem systemd[1]: Starting wazuh-agent.service - Wazuh agent...
ago 23 22:17:46 agenteSiem env[3230]: Starting Wazuh v4.7.3...
ago 23 22:17:47 agenteSiem env[3230]: Started wazuh-execd...
ago 23 22:17:49 agenteSiem env[3230]: Started wazuh-agentd...
ago 23 22:17:50 agenteSiem env[3230]: Started wazuh-syscheckd...
ago 23 22:17:51 agenteSiem env[3230]: Started wazuh-logcollector...
ago 23 22:17:52 agenteSiem env[3230]: Started wazuh-modulesd...
ago 23 22:17:54 agenteSiem env[3230]: Completed.
ago 23 22:17:54 agenteSiem systemd[1]: Started wazuh-agent.service - Wazuh agent.
root@agenteSiem:~#
```

Y si ahora vemos en el inicio de wazuh, podemos ver que tenemos un agente activo

The screenshot displays the Wazuh web interface. The top navigation bar includes a menu icon, a home icon, the 'wazuh.' logo, and a 'Modules' tab. Below this, a summary card shows 'Total agents: 1' and 'Active agents: 1'. A 'SECURITY INFORMATION MANAGEMENT' button is visible.

The 'Agents' tab is selected, showing a 'STATUS' section with a green donut chart indicating 1 active agent. A legend shows: Active (1), Disconnected (0), Pending (0), and Never connected (0). The 'DETAILS' section shows counts for Active (1), Disconnected (0), Pending (0), and Never connected (0). It also lists the 'Last registered agent' as 'agenteSiem1' and the 'Most active agent' as 'agenteSiem1'.

Below this is a table titled 'Agents (1)' with search filters and a table of agent details:

ID	Name	IP address	Group(s)	Operating system
001	agenteSiem1	192.168.0.35	default	Debian GNU/Linux 12

At the bottom, it indicates 'Rows per page: 10'.

3.2 REGLAS PARA BLOQUEO DE IP SOSPECHOSAS

La edición directa dentro del contenedor es riesgosa. El mejor método es copiar el archivo a tu máquina anfitriona, editarlo y luego copiarlo de nuevo.

- a) **Copiamos el archivo fuera del contenedor:** Desde la terminal en el servidor Debian, usar docker cp para copiar el archivo de configuración a un directorio temporal.

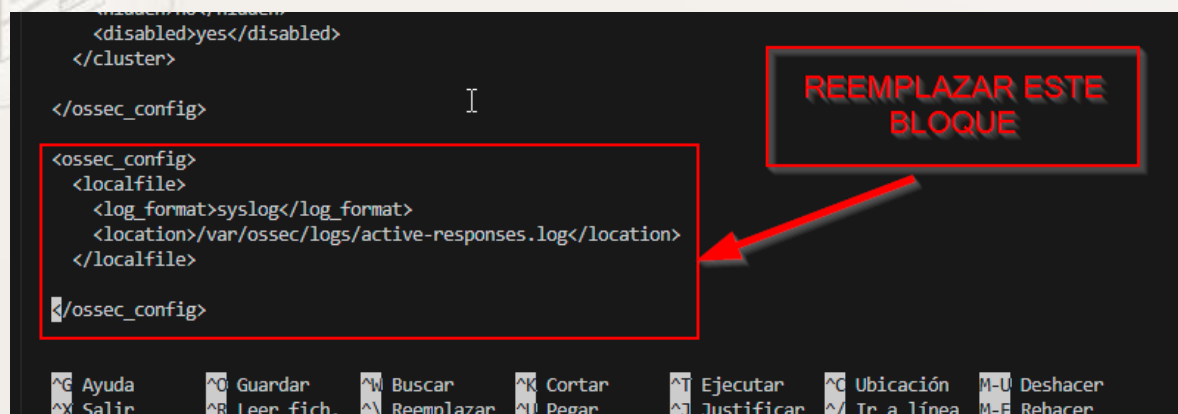
```
docker cp single-node-wazuh.manager-1:/var/ossec/etc/ossec.conf /tmp/ossec.conf
```

```
root@ServidorSIEM:~# docker cp single-node-wazuh.manager-1:/var/ossec/etc/ossec.conf /tmp/ossec.conf
Successfully copied 10.2kB to /tmp/ossec.conf
```

- b) **Editar el archivo con nano:** Usar el editor de texto nano en la máquina anfitriona, donde es mucho más fácil evitar errores de sintaxis.

```
nano /tmp/ossec.conf
```

Dentro del archivo, busca la etiqueta de cierre `</ossec_config>` (está casi al final del archivo) y **reemplazamos ese bloque con el proporcionado**



```

<disabled>yes</disabled>
</cluster>

</ossec_config>

<ossec_config>
<localfile>
  <log_format>syslog</log_format>
  <location>var/ossec/logs/active-responses.log</location>
</localfile>
</ossec_config>
```

REEMPLAZAR ESTE BLOQUE

^G Ayuda ^O Guardar ^W Buscar ^K Cortar ^T Ejecutar ^C Ubicación M-U Deshacer
^X Salir ^R Leer fich. ^N Reemplazar ^U Pegar ^J Justificar ^_ Ir a línea M-E Rehacer

```

<ossec_config>
  <localfile>
    <log_format>syslog</log_format>
    <location>/var/ossec/logs/active-responses.log</location>
  </localfile>

  <!-- Comando para bloquear IPs -->
  <command>
    <name>firewall-drop</name>
    <executable>firewall-drop</executable>
    <timeout_allowed>yes</timeout_allowed>
  </command>

  <!-- Bloqueo de fuerza bruta SSH -->
  <active-response>
    <command>firewall-drop</command>
    <location>local</location>
    <rules_group>sshd</rules_group>
    <timeout>600</timeout>
  </active-response>
</ossec_config>

```

Guardamos los cambios.

c) Validamos el archivo con

```
xmllint --noout /tmp/ossec.conf
```

d) **Copiar el archivo de vuelta al contenedor:** Ahora, copiamos el archivo modificado de regreso al contenedor del Manager.

```
docker cp /tmp/ossec.conf single-node-wazuh.manager-1:/var/ossec/etc/ossec.conf
```

```

root@ServidorSIEM:~# docker cp /tmp/ossec.conf single-node-wazuh.manager-1:/var/ossec/etc/ossec.conf
Successfully copied 11.8kB to single-node-wazuh.manager-1:/var/ossec/etc/ossec.conf
root@ServidorSIEM:~#

```

e) Reiniciar el Manager

Para que los cambios en el archivo de configuración surtan efecto, debemos reiniciar el contenedor del Manager.

```
docker restart single-node-wazuh.manager-1
```

Después de unos minutos, el servicio estará completamente operativo con las nuevas reglas.



3.3 INFORME DE WAZUH (FUERZA BRUTA)

- a) Desde el equipo anfitrión (Windows) intento ingresar por ssh a la maquina **CLIENTE**

ssh [ruben1@192.168.0.38](#)

y cuando me pide la contraseña, ingreso contraseñas incorrectas.

- b) Al revisar el log de Wazuh con

docker exec -it single-node-wazuh.manager-1 tail -f
/var/ossec/logs/alerts/alerts.log

```
root@ServidorSIEM:~# docker exec -it single-node-wazuh.manager-1 tail -f /var/ossec/logs/alerts/alerts.log
parameters.alert.location: journald
parameters.program: active-response/bin/firewall-drop

** Alert 1756646974.25687: - syslog,access_control,authentication_failed,pci_dss_10.2.4,pci_dss_10.2.5,
gpg13_7.8,gdpr_IV_35.7.d,gdpr_IV_32.2,hipaa_164.312.b,nist_800_53_AU.14,nist_800_53_AC.7,tsc_CC6.1,tsc
CC6.8,tsc CC7.2,tsc CC7.3,
2025 Aug 31 13:29:34 (agenteSiem1) any->journald
Rule: 2502 (level 10) -> 'syslog: User missed the password more than one time'
Src IP: 192.168.0.31
User: ruben1
Aug 31 13:29:33 clienteSIEM sshd[3540]: PAM 2 more authentication failures; logname= uid=0 euid=0 tty=s
sh ruser= rhost=192.168.0.31 user=ruben1
```



Lo que se ve en el log confirma que la simulación funcionó correctamente:


- La **IP atacante** (192.168.0.31) fue detectada.
- La regla **2502 (level 10)** se disparó: "User missed the password more than one time".
- El **active-response** firewall-drop se ejecutó automáticamente.



Esto significa que Wazuh está bloqueando la IP como esperábamos.

Campo	log
Fecha/Hora	2025 Aug 31 13:29:34
Agente	agenteSiem1 (192.168.0.38)
IP atacante	192.168.0.31
Usuario atacado	ruben1
Regla disparada	2502 (level 10)
Acción	firewall-drop ejecutado

La alerta en el Dashboard de wazuh


Recent events  

Search DQL  Aug 31, 2025 @ 09:26:30.0 → Aug 31, 2025 @ 09:27:00.0 Refresh

  Add filter

8 hits
Aug 31, 2025 @ 09:26:30.000 - Aug 31, 2025 @ 09:27:00.000

Time	Technique(s)	Tactic(s)	Level	Rule ID	Description
> Aug 31, 2025 @ 09:26:50.022	T1110	Credential Access	10	2502	syslog: User missed the password more than one time
> Aug 31, 2025 @ 09:26:49.924			3	651	Host Blocked by firewall-drop Active Response
> Aug 31, 2025 @ 09:26:48.022	T1110.001 T1021.004	Credential AccessLateral Movement	5	5760	sshd: authentication failed.
> Aug 31, 2025 @ 09:26:43.922			3	651	Host Blocked by firewall-drop Active Response
> Aug 31, 2025 @ 09:26:42.012	T1110.001 T1021.004	Credential AccessLateral Movement	5	5760	sshd: authentication failed.
> Aug 31, 2025 @ 09:26:35.912			3	651	Host Blocked by firewall-drop Active Response
> Aug 31, 2025 @ 09:26:34.004	T1110.001 T1021.004	Credential AccessLateral Movement	5	5760	sshd: authentication failed.
> Aug 31, 2025 @ 09:26:32.001	T1110.001	Credential Access	5	5503	PAM: User login failed.

Rows per page: 20 

< 1 >

La diferencia en el registro horario se soluciona cambiando la hora en el contenedor del Wazuh Manager

1. Cambiar zona horaria del contenedor Wazuh Manager

- Verifica la zona horaria dentro del contenedor:

2. `docker exec -it single-node-wazuh.manager-1 date`

- Si quieres cambiarla a tu hora local:

3. `docker exec -it single-node-wazuh.manager-1 ln -sf /usr/share/zoneinfo/America/Santiago /etc/localtime`

4. `docker exec -it single-node-wazuh.manager-1 dpkg-reconfigure -f noninteractive tzdata`

5. Ajustar la zona horaria en Wazuh Dashboard

- En el Dashboard: **Management** → **Advanced Settings** → **dateFormat:tz**
- Cambia a America/Santiago o Local para que coincida con la hora de tu país.



4.SQUID + SURICATA + WAZUH

4.1 TIPOS DE CIFRADOS RELACIONADOS CON SERVIDORES PROXY

Los servidores proxy actúan como intermediarios entre clientes y servidores, y muchas veces manejan tráfico confidencial. Por eso, se utilizan cifrados para proteger la comunicación. Los tipos principales son:

1. Cifrado simétrico

- Usa la misma clave para cifrar y descifrar.
Ejemplo: AES (Advanced Encryption Standard).
- **Ventaja:** rápido y eficiente para grandes volúmenes de datos.
- **Desventaja:** distribución de claves segura es crítica.

2. Cifrado asimétrico

- Usa un par de claves: pública para cifrar y privada para descifrar.
Ejemplo: RSA, ECC.
- **Ventaja:** facilita distribución de claves, ideal para autenticación y firmas digitales.
- **Desventaja:** más lento que el cifrado simétrico, poco eficiente para grandes datos.

3. Cifrado híbrido

- Combina asimétrico y simétrico: se usa asimétrico para intercambiar una clave simétrica, y luego se cifra el tráfico con AES u otro simétrico.
- Es el enfoque que usan protocolos como **HTTPS**.

Elección y contexto de uso

- **HTTPS/TLS con cifrado híbrido (RSA + AES)**

- **Contexto:** un proxy que maneja tráfico web seguro entre clientes y servidores.
- **Fundamento técnico:** TLS utiliza RSA (o ECC) para el intercambio seguro de claves y AES para cifrado eficiente del contenido. Esto garantiza **confidencialidad, integridad y autenticación** del servidor.
 - Es el estándar en proxies de empresa que necesitan inspección de tráfico seguro (SSL/TLS Inspection).



4.2 SOFTWARE COMPLEMENTARIO AL IDS SURICATA

Suricata detecta intrusiones mediante inspección de tráfico y firmas, pero **no detiene ataques por sí mismo**. Para un escenario real, se pueden combinar:

1. Firewall

- Ejemplo: **iptables, nftables, pfSense**.
- Función: bloquear tráfico malicioso detectado por Suricata.
- Justificación: Suricata detecta, el firewall actúa.

2. Sistema de prevención de intrusiones (IPS)

- Ejemplo: Suricata en modo IPS, **Snort IPS, pfSense + Suricata IPS**.
- Función: intercepta y bloquea paquetes sospechosos automáticamente.
- Justificación: transforma la detección en acción inmediata, evitando propagación de ataques.

3. SIEM (Security Information and Event Management)

- Ejemplo: **Wazuh, Splunk, ELK Stack**.
- Función: centraliza logs, correlaciona eventos, genera alertas y reportes.
- Justificación: complementa Suricata dando visibilidad y análisis histórico del tráfico.

4. Antivirus / Endpoint Protection

- Ejemplo: **CrowdStrike, Sophos, Microsoft Defender for Endpoint**.
- Función: detecta malware que pueda pasar por la red.
- Justificación: Suricata analiza tráfico; la protección en endpoints detecta amenazas internas o tráfico cifrado.

Resumen de elección práctica:

- Suricata IDS + Firewall/IPS + SIEM
- Motivo: detección en tiempo real (Suricata), bloqueo (firewall/IPS) y análisis centralizado (SIEM).

4.3 INTEGRACIÓN SQUID + SURICATA + WAZUH

A continuación se muestra un flujo de trabajo de seguridad en una red típica:

1. **Tráfico de Clientes:** Los clientes de la red (por ejemplo, un servidor web o un cliente de escritorio) inician una solicitud de conexión.
2. **Squid (Proxy):** Todas las solicitudes web pasan primero a través de Squid.
 - **Control de Acceso:** Squid revisa si la URL solicitada está permitida o bloqueada. Por ejemplo, bloquea el acceso a latercera.com. Si la solicitud es denegada, genera un evento de "acceso denegado" que se registra en sus logs.
 - **Caché:** Si la página ya ha sido visitada, Squid la entrega desde su caché, mejorando el rendimiento y reduciendo el tráfico externo.
3. **Suricata (IDS/IPS):** El tráfico que Squid no bloquea (o cualquier otro tráfico de la red) es inspeccionado por Suricata.
 - **Análisis de Paquetes:** Suricata analiza cada paquete de datos en busca de firmas de ataques, tráfico malicioso o actividades sospechosas (por ejemplo, el tráfico ICMP o las conexiones a Facebook/YouTube).
 - **Generación de Alertas:** Si detecta una amenaza, Suricata genera una alerta.
4. **Wazuh (HIDS/SIEM):** Las alertas de Suricata y los logs de Squid se envían al servidor de Wazuh, junto con los logs de los propios sistemas (servidores y agentes de Wazuh).
 - **Correlación de Eventos:** Wazuh correlaciona los logs de diferentes fuentes para identificar patrones de ataque. Por ejemplo, puede correlacionar una serie de "accesos denegados" de Squid con una alerta de "ping anómalo" de Suricata, sugiriendo un escaneo de red.
 - **Análisis y Visualización:** Wazuh centraliza toda la información en un único panel, permitiendo al administrador de seguridad visualizar, analizar y responder a los eventos de manera eficiente.

- **Respuestas Activas:** Wazuh puede ejecutar automáticamente una respuesta a un evento detectado, como utilizar un script para bloquear la dirección IP de un atacante en el firewall del sistema, deteniendo el ataque en el origen.

Esta integración crea una defensa en capas donde cada herramienta complementa a las otras, ofreciendo una seguridad más robusta y una mejor visibilidad de los eventos que ocurren en la red y los sistemas.



INTEGRACIÓN DE SQUID + SURICATA + WAZUH

	Squid (Proxy Web)	Suricata (IDS/IPS)	Wazuh (HIDS/SIEM)
Rol Principal	Control de acceso web y caché. Actúa como un intermediario que filtra y almacena tráfico HTTP/HTTPS.	Detección y prevención de intrusiones en la red. Analiza el tráfico de la red en tiempo real para encontrar amenazas.	Detección de intrusiones a nivel de host y gestión de eventos de seguridad (SIEM). Centraliza, analiza y correlaciona logs y eventos de seguridad.
Punto de Enfoque	Tráfico de red a nivel de aplicación (Capa 7), específicamente HTTP y HTTPS.	Tráfico de red a nivel de paquetes (Capas 2-7).	Logs y eventos del sistema operativo y aplicaciones en los hosts (servidores y clientes).
Relación e Integración	Squid puede ser configurado para que todo su tráfico pase a través de Suricata, permitiendo a este último inspeccionar el tráfico web filtrado.	Suricata puede enviar alertas de intrusión (como el tráfico ICMP o intentos de conexión a sitios bloqueados) a Wazuh.	Wazuh centraliza y correlaciona las alertas generadas por Suricata y los logs de acceso de Squid. Esto permite una visibilidad completa y correlacionada de lo que ocurre en la red.
Cómo se Potencian	Squid potencia a Suricata al darle una visión más clara del tráfico web. Sin Squid, Suricata tendría que lidiar con todo el tráfico de la red; con Squid, puede centrarse en inspeccionar el tráfico ya filtrado.	Suricata potencia a Wazuh al proporcionarle inteligencia sobre las amenazas de red. Sin Suricata, Wazuh no tendría visibilidad sobre ataques de red, solo sobre eventos a nivel de host.	Wazuh potencia a ambos al actuar como un cerebro de seguridad. Recopila las alertas de Suricata (por ejemplo, "conexión sospechosa") y los logs de Squid (por ejemplo, "acceso denegado") para crear una imagen completa y accionable de la seguridad de la red. Además, las respuestas activas de Wazuh pueden bloquear una IP en el firewall del host si Suricata detecta un comportamiento malicioso.
Ejemplo de Flujo de Trabajo	Un usuario intenta acceder a www.latercera.com . Squid deniega el acceso según su regla de configuración.	Suricata detecta una conexión ICMP (ping) no autorizada a un servidor. Genera una alerta.	Wazuh recibe la alerta de Suricata, la correlaciona con otros eventos y ejecuta una acción automática (como el bloqueo de la IP atacante) para detener el tráfico malicioso. Además, Wazuh almacena los logs de Squid, documentando el intento de acceso fallido a www.latercera.com para futuros análisis forenses.

