

AllWatch

Un supervisor de procesos minúsculo para Linux: objetivos en primer plano, salud por el núcleo y TCP, políticas de reinicio por objetivo

Manual

Versión	0.1.0
Commit	`ae9cb8f`
Fecha del commit	24 de noviembre de 2026
Licencia	MIT

Manual fuera de línea, idéntico a las guías del repositorio en la versión y el commit indicados. El texto del repositorio es la referencia.

Índice

Resumen **1**
- Instalación en Debian y Ubuntu . . . 1
- Instalación en Fedora, openSUSE Leap y Arch Linux . . . 1
- Ejecución desde el repositorio . . . 2
- Cómo funciona . . . 2
- Configuración . . . 3
- El contrato de un objetivo . . . 3
- Límites . . . 3
- Licencia . . . 4
- Créditos . . . 4

Instalar y administrar el servicio Debian **5**
- Lo que instala el paquete . . . 5
- Instalar . . . 5
- Rutas y archivos . . . 5
- Declarar un objetivo . . . 5
- Objetivo colgado: restart o alert . . . 6
- Comandos del servicio . . . 6
- Comprobar la configuración sin iniciar un objetivo . . . 6
- Detener un objetivo sin detener el servicio . . . 6
- Ejecutar el servicio con otro usuario . . . 7
- Actualizar . . . 7
- Retirar y purgar . . . 7
- Otras familias: Fedora, openSUSE Leap y Arch Linux . . . 7
- Límites conocidos del paquete . . . 8

Solución de problemas **9**
- Leer primero el estado del servicio . . . 9
- El servicio no arranca . . . 9
- El objetivo no se reinicia . . . 9
- El objetivo se declara bloqueado aunque esté en marcha . . . 9
- `--diagnose` devuelve 0 pero el objetivo no está sano . . . 10
- El doble de prueba no responde . . . 10
- El registro crece sin límite . . . 10
- Objetivos duplicados tras un reinicio del supervisor . . . 10
- Nada en el registro . . . 10

Resumen

El programa muestra el resumen de --help en el idioma del sistema: inglés por defecto, con fr, de, es y zh traducidos.

AllWatch es un supervisor de procesos minúsculo para Linux, escrito en un solo script Bash (unas ciento setenta líneas útiles de código). Inicia cada objetivo en primer plano, recuerda su PID y juzga su salud solo con lo que dicen el núcleo y la red: ¿el proceso sigue en /proc, su puerto TCP sigue aceptando una conexión y qué código de salida dejó? Reinicia el objetivo según una política por objetivo y se niega a entrar en un bucle infinito con un objetivo que muere de inmediato.

El objetivo es una caja negra. No se registra ante AllWatch, no le envía nada y no necesita saber que existe.

Instalación en Debian y Ubuntu

El paquete Debian instala el supervisor como /usr/bin/allwatch, la unidad systemd allwatch.service, la configuración de ejemplo y la página de manual en cinco idiomas. Descargue el paquete y SHA256SUMS desde la página de descarga y ejecute:

```
sudo apt install ./allwatch_0.1.0-7_all.deb
```

Nivel anunciado: **instalación y diagnóstico validados en contenedor**, en Debian 12 y 13 y en Ubuntu 22.04 y 24.04. El paquete Debian también se probó de verdad: instalado como servicio en una Raspberry Pi 4 con Debian 13, con una aplicación real, durante dos horas y luego seis horas (95 eventos, 95 superados, ninguno fallido), y después retirado y purgado. El servicio se instala activado, con una configuración que no declara ningún objetivo; añada los suyos en /etc/allwatch/allwatch.conf y ejecute sudo systemctl restart allwatch.

Compruebe la versión instalada y la configuración sin iniciar un objetivo:

```
allwatch --version
allwatch --help
allwatch --diagnose /etc/allwatch/allwatch.conf
```

--diagnose lee la configuración, el directorio de estado y la ruta del registro. No inicia ni sondea ningún objetivo: no demuestra que un objetivo esté sano, demuestra que el supervisor puede leer su configuración y usar sus rutas.

Para retirar el paquete:

```
sudo apt remove allwatch
sudo apt purge allwatch
```

apt remove detiene el servicio pero lo deja activado; apt purge elimina además la configuración y el estado del servicio.

Instalación en Fedora, openSUSE Leap y Arch Linux

La misma etiqueta construye también paquetes nativos: un .rpm para Fedora, un .rpm para openSUSE Leap y un .pkg.tar.zst para Arch Linux. Descargue el archivo de su familia y SHA256SUMS desde la página de descargas y ejecute la línea de su familia (Fedora, openSUSE Leap, Arch Linux, en este orden):

```
sudo dnf install ./allwatch-0.1.0-1.fc44.noarch.rpm
sudo zypper install --allow-unsigned-rpm ./allwatch-0.1.0-1.leap160.noarch.rpm
sudo pacman -U ./allwatch-0.1.0-1-any.pkg.tar.zst
```

Nivel anunciado: **instalación y diagnóstico validados en contenedor**, en Fedora 43, 44 y 45, en openSUSE Leap 16.0 y en Arch Linux. Los paquetes no se presentan como probados en un servidor

físico. El servicio se instala pero no se inicia: añada sus objetivos a /etc/allwatch/allwatch.conf y ejecute sudo systemctl enable --now allwatch. Los comandos allwatch --version, --help y --diagnose son los mostrados arriba.

Para desinstalar el paquete:

```
sudo dnf remove allwatch
sudo zypper remove allwatch
sudo pacman -R allwatch
```

La desinstalación conserva el usuario de sistema allwatch y /var/lib/allwatch, y su configuración si la modificó (guardada como allwatch.conf.rpmsave o allwatch.conf.pacsave). Para borrarlos también:

```
sudo userdel allwatch
sudo rm -rf /etc/allwatch /var/lib/allwatch
```

Ejecución desde el repositorio

Requisitos: Linux (el supervisor lee /proc), Bash 5 o posterior, y timeout y awk de las utilidades básicas habituales. Solo para el doble de prueba dummy-target.sh: setpriv (util-linux) y OpenBSD nc.

```
git clone https://github.com/RonanDavalan/AllWatch.git
cd AllWatch
./allwatch.sh --version
./allwatch.sh --help
./allwatch.sh allwatch.conf
```

La configuración de ejemplo inicia dummy-target.sh en el puerto 9001 con el identificador demo. En una segunda terminal:

```
cat /tmp/allwatch-runtime/demo.pid
: > /dev/tcp/127.0.0.1/9001 && echo "el puerto 9001 responde"
```

Mate el objetivo a la fuerza y véalo volver con un PID nuevo (el intervalo de consulta es de 5 segundos):

```
kill -9 "$(cat /tmp/allwatch-runtime/demo.pid)"
sleep 6
cat /tmp/allwatch-runtime/demo.pid
```

Provoque un bloqueo (el objetivo sigue vivo pero cierra su puerto). AllWatch lo detecta, lo detiene y lo reinicia, lo que tarda unos quince segundos:

```
kill -USR1 "$(cat /tmp/allwatch-runtime/demo.pid)"
sleep 20
grep ' demo ' ~/.local/state/allwatch/allwatch.log | tail -n 6
```

Detenga el supervisor con Ctrl+C (o SIGTERM); detiene sus objetivos y vacía su directorio de estado:

```
ls -A /tmp/allwatch-runtime
```

Cómo funciona

AllWatch es el padre, el objetivo es el hijo. Cada pocos segundos examina cada objetivo y lo sitúa en uno de cuatro estados:

- **En marcha**: el proceso existe y su puerto acepta una conexión.
- **Bloqueado**: el proceso existe pero su puerto no responde. Con ON_FREEZE=restart (el valor por defecto), AllWatch envía SIGTERM, espera 5 segundos, envía SIGKILL si hace falta y considera el objetivo fallido. Con ON_FREEZE=alert, no envía ninguna señal, registra ALERTE GEL una vez, deja el objetivo vigilado y registra RETABLI cuando el puerto vuelve a responder.
- **Caído**: el proceso desapareció sin un cierre deliberado. AllWatch lee el código de salida y reinicia el objetivo si su política lo permite.
- **Terminado**: el objetivo terminó y su política no pide reinicio, o se alcanzó el límite de reinicios.

Configuración

La configuración se lee una vez, al arrancar, línea por línea. Las líneas vacías y las que empiezan por # se ignoran. Parámetros del motor, un KEY=value por línea:

- POLL_SECONDS=5: segundos entre dos comprobaciones.
- PROBE_TIMEOUT=3: segundos que tiene un puerto para responder a la sonda.
- MAX_RESTARTS=5: reinicios permitidos en la ventana siguiente.
- RESTART_WINDOW=60: duración de esa ventana, en segundos.
- ON_FREEZE=restart: un objetivo bloqueado se neutraliza y se reinicia; alert lo deja intacto y registra ALERTE GEL. Cualquier otro valor se rechaza al arrancar con código 1.

Objetivos, uno por línea:

```
id port policy command...
```

- id: un nombre corto para los archivos de estado y el registro.
- port: el puerto TCP del objetivo (sondeado en 127.0.0.1).
- policy: always, on-failure o transient.
- command...: el resto de la línea, separado por espacios (sin comillas), ejecutado en primer plano desde el directorio de arranque.

Políticas:

- always: reiniciar el objetivo cada vez que termine, incluso con código de salida 0, salvo que exista un archivo <id>.stop en el directorio de estado.
- on-failure: reiniciar tras una caída o un bloqueo, pero dejarlo detenido tras un código de salida 0.
- transient: no reiniciarlo nunca.

Si un objetivo alcanza MAX_RESTARTS reinicios dentro de RESTART_WINDOW segundos, AllWatch deja de reiniciarlo, registra ALERTE VERROU_CRITIQUE y sigue con los demás objetivos.

El directorio de estado es /tmp/allwatch-runtime (se puede cambiar con ALLWATCH_RUNTIME); el registro es un archivo plano, una línea por evento (date id event), en ${XDG_STATE_HOME:-$HOME/.local/state}/allwatch/allwatch.log.

El contrato de un objetivo

AllWatch no se adapta a nada; el objetivo que rompe el contrato es el culpable.

- **Primer plano.** El objetivo se ejecuta en primer plano del proceso que inicia AllWatch. No debe demonizarse ni hacer doble fork.
- **Códigos de salida.** Una parada limpia y solicitada termina con código 0. Cualquier fallo termina con un código distinto de cero.
- **Un puerto TCP.** Cada objetivo posee un puerto TCP, abierto y aceptando conexiones desde su inicialización. Un objetivo vivo que no acepta una conexión dentro del plazo de la sonda se considera bloqueado.
- **SIGTERM.** El objetivo captura SIGTERM, libera sus recursos y termina con código 0 en menos de 5 segundos. Después, AllWatch envía SIGKILL.

El título de la ventana, el servidor gráfico y la salida propia del objetivo no intervienen en la comprobación de salud.

Límites

- Sin recarga en caliente de la configuración: sudo systemctl restart allwatch.
- Sin comprobación por títulos de ventana, X11 o Wayland.
- Sin análisis de registros de aplicación para decidir un reinicio.
- Sin supervisión de objetivos que se separan del primer plano.

- --diagnose es estático: lee la configuración y las rutas, no sondea ningún objetivo y no demuestra que un objetivo esté sano.
- --version, --help y --diagnose solo se reconocen como primer argumento.
- El comando no se ejecuta por un shell: sin comillas, tuberías, redirecciones ni expansión de variables.
- Los valores de configuración y los nombres de política no se validan.
- El registro plano no se rota.

Licencia

MIT. Véase LICENSE.

Créditos

- **Ronan Davalan**: arquitecto y árbitro. Definió la visión y el alcance, arbitró las especificaciones y las validó.
- **Claude Sonnet 5.5** (Anthropic): único autor del código (el supervisor y el doble de prueba), escrito en tres sesiones el 13 de noviembre de 2026.
- **DeepSeek V4.1 Flash 1017**: documentación de referencia, campaña de validación de seis horas e informe de campaña.
- **Gemini 4 Hydrogène** (Google): intercambios de diseño y reflexión sobre el supervisor, idea y diseño del icono.

Véase CONTRIBUTORS.md.

Instalar y administrar el servicio Debian

El paquete Debian instala AllWatch como servicio systemd. El servicio arranca al inicio, lee su configuración una sola vez y espera si no se declara ningún objetivo. Este capítulo da las rutas, los archivos y los comandos exactos del paquete.

Lo que instala el paquete

```
/usr/bin/allwatch                          el supervisor
/usr/bin/allwatch.sh                       enlace de compatibilidad
/usr/lib/systemd/system/allwatch.service   la unidad systemd
/etc/allwatch/allwatch.conf                la configuración
/usr/share/man/man1/allwatch.1.gz          página de manual, inglés
/usr/share/man/fr/man1/allwatch.1.gz       página de manual, francés
/usr/share/man/de/man1/allwatch.1.gz       página de manual, alemán
/usr/share/man/es/man1/allwatch.1.gz       página de manual, español
/usr/share/man/zh_CN/man1/allwatch.1.gz    página de manual, chino simplificado
/usr/share/doc/allwatch/README.Debian.gz   notas del paquete
/usr/share/doc/allwatch/examples/allwatch.conf.example
/usr/share/doc/allwatch/examples/allwatch.conf  configuración del repositorio de fuentes
/usr/share/doc/allwatch/examples/dummy-target.sh
```

El paquete crea el usuario y el grupo de sistema allwatch (directorio personal /var/lib/allwatch, shell /usr/sbin/nologin). El servicio se ejecuta con esa identidad desde /var/lib/allwatch.

Instalar

```
sudo apt install ./allwatch_0.1.0-7_all.deb
```

El paquete instala el programa y la unidad. Cuando systemd está en ejecución, activa allwatch.service y lo inicia. La configuración entregada no contiene ningún objetivo activo: el servicio arranca y espera.

Compruebe el estado del paquete y la versión del programa:

```
dpkg -s allwatch | grep -E '^(Status|Version):'
allwatch --version
```

allwatch --version muestra allwatch 0.1.0; la versión del paquete es 0.1.0-7.

Rutas y archivos

- /etc/allwatch/allwatch.conf es un archivo de configuración (conffile) propiedad de root, no escribible por el usuario del servicio. Es la frontera de confianza: los comandos de los objetivos se ejecutan con los privilegios del usuario del servicio.
- /var/lib/allwatch contiene el estado y el registro, modo 0700, propietario allwatch:allwatch, creado por systemd. El registro plano es /var/lib/allwatch/allwatch/allwatch.log.
- /run/allwatch es el directorio de ejecución, modo 0700, creado en cada arranque del servicio y retirado al detenerlo. Contiene <id>.pid, <id>.fails y <id>.stop.
- /var/lib/allwatch y /run/allwatch no sobreviven a una purga; /run/allwatch no sobrevive a una parada.

Declarar un objetivo

Edite /etc/allwatch/allwatch.conf como root y añada una línea por objetivo:

```
id port policy command...
```

Ejemplo real con el doble de prueba instalado por el paquete:

```
sudo apt install netcat-openbsd
sudo cp /usr/share/doc/allwatch/examples/allwatch.conf.example /etc/allwatch/allwatch.conf
sudo systemctl restart allwatch
```

El ejemplo declara:

```
demo 9001 on-failure /usr/share/doc/allwatch/examples/dummy-target.sh 9001
```

Compruebe el puerto:

```
bash -c ': > /dev/tcp/127.0.0.1/9001' && echo "el puerto 9001 responde"
```

El usuario del servicio debe poder ejecutar el comando. Use rutas absolutas: el comando se divide por espacios, sin comillas, y no se ejecuta por un shell. Para argumentos con espacios o sintaxis de shell, escriba un script envoltorio y llámelo.

Objetivo colgado: restart o alert

ON_FREEZE decide qué ocurre con un objetivo que sigue vivo pero cuyo puerto TCP ya no responde tras el plazo de gracia. ON_FREEZE=restart (el valor por defecto del paquete) envía SIGTERM, espera unos cinco segundos, envía SIGKILL si hace falta y reinicia el objetivo según su política. ON_FREEZE=alert no envía ninguna señal: registra ALERTE GEL una vez, deja el objetivo en ejecución y registra RETABLI cuando el puerto vuelve a responder. Una caída sigue la política del objetivo en ambos casos. Cualquier otro valor se rechaza al arrancar con código 1.

Comandos del servicio

```
sudo systemctl status allwatch
sudo systemctl start allwatch
sudo systemctl stop allwatch
sudo systemctl restart allwatch
sudo systemctl enable allwatch
sudo systemctl disable allwatch
sudo journalctl -u allwatch -f
sudo tail -f /var/lib/allwatch/allwatch/allwatch.log
```

La configuración se lee una vez al arrancar; un cambio exige sudo systemctl restart allwatch. No hay recarga en caliente.

Comprobar la configuración sin iniciar un objetivo

Ejecute --diagnose con el entorno del servicio y como usuario del servicio:

```
sudo -u allwatch env ALLWATCH_RUNTIME=/run/allwatch \
  XDG_STATE_HOME=/var/lib/allwatch \
  allwatch --diagnose /etc/allwatch/allwatch.conf
```

Muestra la versión, la ruta de configuración, la ruta de ejecución, la ruta del registro, la conducta ante un bloqueo y, para cada objetivo, su identificador, puerto y política. Devuelve 0 cuando la configuración es legible, un directorio de ejecución existente pertenece al usuario actual y un directorio de registro existente es escribible. Devuelve 1 por una configuración ilegible, un ON_FREEZE inválido, un POLL_SECONDS no numérico, un directorio de ejecución existente que no pertenece al usuario actual, o un directorio de registro existente que no es escribible. No inicia ningún objetivo ni sondea ningún puerto.

Detener un objetivo sin detener el servicio

Cree el archivo de parada como usuario del servicio:

```
sudo -u allwatch touch /run/allwatch/demo.stop
```

AllWatch retira el archivo cuando el objetivo termina y no lo reinicia. El archivo no detiene por sí solo un objetivo en ejecución: envíe SIGTERM al objetivo o déjelo terminar.

Ejecutar el servicio con otro usuario

El servicio se ejecuta como allwatch. Si los objetivos necesitan root u otra cuenta, cree un drop-in:

```
sudo systemctl edit allwatch
```

Añada:

```
[Service]
User=www-data
Group=www-data
```

Luego reinicie:

```
sudo systemctl restart allwatch
```

Los comandos de los objetivos heredan esa identidad. El archivo de configuración sigue siendo propiedad de root.

Actualizar

```
sudo apt install ./allwatch_0.1.0-7_all.deb
```

/etc/allwatch/allwatch.conf se conserva; si la versión del paquete lo modifica, dpkg pregunta qué versión mantener. El servicio se reinicia tras la actualización si estaba activo. /var/lib/allwatch y su registro permanecen.

Retirar y purgar

```
sudo apt remove allwatch
sudo apt purge allwatch
```

apt remove detiene el servicio, retira el programa, la unidad, las páginas de manual y la documentación del paquete. Conserva /etc/allwatch/allwatch.conf, /var/lib/allwatch y el usuario y grupo allwatch. No desactiva la unidad; el estado de activación de deb-systemd-helper permanece hasta la purga.

apt purge retira además el conffile, /var/lib/allwatch, /etc/allwatch cuando está vacío, el usuario y grupo allwatch cuando pueden retirarse, y el estado de activación. Si un proceso usa todavía la cuenta, userdel o groupdel pueden fallar; compruebe con:

```
getent passwd allwatch
getent group allwatch
```

Otras familias: Fedora, openSUSE Leap y Arch Linux

El .rpm de Fedora y openSUSE Leap y el .pkg.tar.zst de Arch Linux instalan los mismos archivos en las mismas rutas que el paquete Debian: /usr/bin/allwatch, /usr/lib/systemd/system/allwatch.service, /etc/allwatch/allwatch.conf, las páginas de manual y /usr/share/doc/allwatch. Instale el archivo de su familia (Fedora, openSUSE Leap, Arch Linux, en este orden); la misma línea actualiza una versión instalada:

```
sudo dnf install ./allwatch-0.1.0-1.fc44.noarch.rpm
sudo zypper install --allow-unsigned-rpm ./allwatch-0.1.0-1.leap160.noarch.rpm
sudo pacman -U ./allwatch-0.1.0-1-any.pkg.tar.zst
```

Diferencias con el paquete Debian: el servicio se instala pero no se inicia, así que ejecute sudo systemctl enable --now allwatch cuando haya declarado sus objetivos; /etc/allwatch/allwatch.conf se conserva si lo modificó y se guarda como allwatch.conf.rpmsave (Fedora, openSUSE) o allwatch.

conf.pacsave (Arch) al desinstalar el paquete; no hay purga, de modo que la desinstalación conserva el usuario allwatch y /var/lib/allwatch, y el paquete muestra los dos comandos que los borran:

```
sudo userdel allwatch
sudo rm -rf /etc/allwatch /var/lib/allwatch
```

Límites conocidos del paquete

- La validación en contenedor instala el paquete, comprueba --version, --help y --diagnose, y lo retira. No arranca systemd: el servicio está instalado y documentado, pero no validado bajo systemd en el contenedor.
- --diagnose es estático: no sondea objetivos.
- /run/allwatch no sobrevive a una parada del servicio. Tras un fallo del supervisor, los objetivos pueden sobrevivir sin su archivo PID; el supervisor reiniciado puede lanzar duplicados. Un drop-in con RuntimeDirectoryPreserve=restart conserva el directorio entre reinicios del servicio.

Solución de problemas

AllWatch no tiene comando de estado. El estado de un objetivo se lee en el proceso, el puerto TCP y el registro plano. Este capítulo da los comandos exactos de los fallos habituales.

Leer primero el estado del servicio

```
sudo systemctl status allwatch
sudo journalctl -u allwatch -n 50 --no-pager
sudo tail -n 50 /var/lib/allwatch/allwatch/allwatch.log
```

Las líneas del registro son fecha id evento. Los eventos son DEMARRAGE, DEMARRE, ADOPTE, PLANTE, ALERTE GEL, RETABLI, NEUTRALISE SIGTERM, NEUTRALISE SIGKILL, CRASHE code=N, RELANCE, ALERTE VERROU_CRITIQUE, QUITTE y ARRET.

El servicio no arranca

El programa escribe estos mensajes en el idioma del sistema (LC_ALL, LC_MESSAGES o LANG); el inglés es el idioma por defecto.

- allwatch: configuración ilegible: /etc/allwatch/allwatch.conf — falta el archivo o el usuario del servicio no puede leerlo. Compruebe sudo -u allwatch test -r /etc/allwatch/allwatch.conf y sudo systemctl cat allwatch.
- allwatch: /run/allwatch no pertenece al usuario actual — el directorio de ejecución pertenece a otro usuario. Compruebe ls -ld /run/allwatch /var/lib/allwatch; la unidad los crea para allwatch. Si añadió un drop-in con otro User=, corrija la propiedad o retire el drop-in.
- status=203/EXEC — falta el programa o no es ejecutable. Reinstale el paquete.
- status=1/FAILURE con un error de configuración — ejecute --diagnose como usuario del servicio (véase abajo).

El objetivo no se reinicia

- Política on-failure: un objetivo que termina con código 0 queda detenido. Es el contrato, no un fallo. Use always si debe reiniciarse tras una parada limpia.
- Política transient: nunca se reinicia.
- Archivo de parada presente: ls -l /run/allwatch/demo.stop; retírelo si la parada no era intencionada. El archivo no detiene por sí solo un objetivo en ejecución.
- ALERTE VERROU_CRITIQUE: el objetivo alcanzó MAX_RESTARTS en RESTART_WINDOW. Corrija la causa y reinicie el servicio; el bloqueo dura hasta el reinicio del supervisor.
- CRASHE code=N: el objetivo terminó con el código N. Lea su propio registro; AllWatch solo informa del código.

El objetivo se declara bloqueado aunque esté en marcha

- La sonda es timeout 3 bash -c ': > /dev/tcp/127.0.0.1/9001'. El objetivo debe escuchar en 127.0.0.1 (o 0.0.0.0) en el puerto declarado. Un objetivo que solo escucha en otra dirección se considera bloqueado.
- Puerto equivocado en /etc/allwatch/allwatch.conf; compruebe con ss -ltnp | grep 9001.
- El objetivo es más lento que PROBE_TIMEOUT=3; aumente el valor en la configuración y reinicie el servicio.

- El objetivo está vivo pero su escucha está cerrada; es exactamente el estado bloqueado. Con ON_FREEZE=restart (por defecto), AllWatch envía SIGTERM, espera 5 segundos, envía SIGKILL y lo reinicia. Con ON_FREEZE=alert, registra ALERTE GEL una vez y deja el objetivo en ejecución; RETABLI se registra cuando el puerto vuelve a responder.

`--diagnose` devuelve 0 pero el objetivo no está sano

--diagnose es estático. Lee la configuración y las rutas; no sondea ningún objetivo. Un código de retorno 0 solo prueba que el supervisor puede leer la configuración y usar sus directorios.

El doble de prueba no responde

- Falta netcat-openbsd: sudo apt install netcat-openbsd.
- El comando usa una ruta relativa. El servicio se ejecuta desde /var/lib/allwatch; use la ruta absoluta /usr/share/doc/allwatch/examples/dummy-target.sh.
- El puerto ya está ocupado: ss -ltnp | grep 9001.
- El usuario del servicio no puede ejecutar el archivo: sudo -u allwatch test -x /usr/share/doc/allwatch/examples/dummy-target.sh.

El registro crece sin límite

AllWatch no rota el registro. Use logrotate o un temporizador systemd; el archivo es /var/lib/allwatch/allwatch/allwatch.log.

Objetivos duplicados tras un reinicio del supervisor

/run/allwatch se retira al detener el servicio. Un objetivo puede sobrevivir a un fallo del supervisor sin su archivo PID; el nuevo supervisor lanza un duplicado. Añada un drop-in:

```
sudo systemctl edit allwatch
```

```
[Service]
RuntimeDirectoryPreserve=restart
```

Luego reinicie. El directorio de ejecución se conserva entre reinicios del servicio y el nuevo supervisor puede adoptar el PID superviviente.

Nada en el registro

El servicio se ejecuta como allwatch y escribe en /var/lib/allwatch/allwatch/allwatch.log; compruebe el entorno del servicio con sudo systemctl show allwatch -p Environment. Si el directorio no es escribible, el registro queda vacío; --diagnose lo indica.