'OpenVPN: cómo excluir destinos del túnel con `net_gateway` (el caso de apple.com / apple.es)'

> Documento de referencia operativa. Origen: el Mac del operador navega con túnel completo de las VPN propias de OpenVPN (dos servidores Ubuntu 24 con `openvpn-server@server`, servicio `openvpn-server@server.service`), y apple.es / apple.com devolvían `503` mientras iCloud y App Store entraban en bucle silencioso. Solución aplicada y verificada en ambos servidores el 2026-08-27.
>
> **Palabras clave para búsqueda:** OpenVPN, split tunneling, `net_gateway`, `redirect-gateway`, exclusión de destinos, `push route`, Apple, `17.0.0.0/8`, macOS, Safari, 503, iCloud, App Store, OpenVPN Connect, system extension, SplitTunnel, CDN Akamai.

El diagnóstico correcto: no es per-app, es por destino

macOS no soporta enrutado por aplicación (per-app routing) con clientes OpenVPN estándar — ni Tunnelblick, ni Viscosity, ni OpenVPN Connect. Las apps de terceros que lo prometen (p. ej. SplitTunnel) usan extensiones de red y DNS que interceptan todo el tráfico; en esta estación resultaron inestables y se retiraron.

El problema real con Apple no era la app que navega, sino **el destino**: gran parte de la infraestructura de Apple vive en su propio bloque `17.0.0.0/8` (iCloud, App Store, activación, servidores corporativos), y ese bloque rechaza o degrada tráfico que llega desde IPs de datacenter/VPN. La solución quirúrgica es **excluir destinos del túnel**, no aplicaciones: el tráfico hacia `17.0.0.0/8` sale por el gateway físico pre-VPN y todo lo demás sigue cifrado por el túnel.

El mecanismo: `net_gateway`

En la config del servidor OpenVPN, el keyword `net_gateway` significa «la gateway que tenía el cliente ANTES de levantar el túnel» (su router doméstico u office). Se combina con `push` para que el servidor la inyecte como ruta en cada cliente al conectar:

```
push "route 17.0.0.0 255.0.0.0 net_gateway"
```

- Con `redirect-gateway def1` activo, OpenVPN secuestra el tráfico con dos rutas `/1` (`0.0.0.0/1` y `128.0.0.0/1`) que cubren todo el espacio IPv4; la ruta `17.0.0.0/8` es **más específica** y gana sin tocar nada más.
- Es un cambio **solo de servidor**: los clientes (Mac, iPhone, iPad, Linux) no necesitan configurar nada.
- El `push` se aplica al **conectar**: los clientes ya conectados no lo ven hasta reconectar.

Aplicación en el servidor (patrón con backup e idempotencia)

Config en `/etc/openvpn/server/server.conf` (Ubuntu 24, unidad `openvpn-server@server`). Procedimiento por servidor:

```
cd /etc/openvpn/server

# Idempotente: solo añade si no está
grep -q "route 17.0.0.0 255.0.0.0 net_gateway" server.conf || {
  cp -a server.conf server.conf.$(date +%F).bak
  printf '\n# Exclude Apple (17.0.0.0/8) from tunnel - route via pre-VPN gateway\npush "route
17.0.0.0 255.0.0.0 net_gateway"\n' >> server.conf
}

# Reinicio controlado (corta las sesiones activas; los clientes reconectan solos)
systemctl restart openvpn-server@server
sleep 2
systemctl is-active openvpn-server@server
ss -ulnp | grep 1194
journalctl -u openvpn-server@server --since "1 minute ago" --no-pager | grep -E
"Completed|error"
```

Verificación server-side esperada: servicio `active`, socket UDP en escucha y `Initialization Sequence Completed` en el journal.

Verificación en dos puntos (REGLA 0)

1. Desde el servidor: journal limpio tras el reinicio (arriba).

2. Desde el cliente conectado (Mac aquí), verificar el efecto y **un control negativo** para demostrar que el resto del tráfico SÍ sigue por la VPN:

```
# La ruta push llegó (debe existir la línea "17" apuntando a tu router LAN)
netstat -rn -f inet | grep -E "^0/1|^128\.0/1|^17"
#    0/1  y 128.0/1  → 10.8.0.1  utunX   (túnel completo activo)
#    17              → 192.168.x.1  en0    (exclusión Apple)

# El detalle de ruta hacia una IP de Apple sale por la interfaz física
route -n get 17.253.144.10 | grep -E "gateway|interface"
#    gateway: 192.168.x.1 / interface: en0

# CONTROL: el resto del tráfico sale por la IP de la VPN (no por casa)
curl -s https://api.ipify.org; echo

# Apple accesible con el túnel activo
curl -sI https://www.apple.com/library/test/success.html | head -1   # HTTP/2 200
```

Cuidado con las páginas que redirigen: `apple.es` responde `301` a `https://www.apple.com/es/` (servida por Akamai, FUERA del bloque 17/8). Un `curl -I` de una sola pata puede dar «correcto» en el primer salto y que el fallo esté en el siguiente. Verificar SIEMPRE la cadena completa:

```
curl -sIL -o /dev/null -w "%{http_code} -> %{url_effective}\n" https://www.apple.es/
#    200 -> https://www.apple.com/es/
```

Si la IP de `api.ipify.org` es la del servidor VPN y el `route get` de Apple responde con tu router LAN, las dos cosas a la vez están probadas.

CDN de Apple fuera del bloque: añadir `/32` puntuales

`www.apple.com` y buena parte de los assets estáticos se sirven por Akamai (`www.apple.com.edgekey.net`), IPs que **no** están en `17.0.0.0/8`. Eso normalmente funciona igual por el túnel; si alguna página o descarga concreta sigue fallando, localiza su IP y exclúyela puntualmente:

```
# Localizar la IP real del dominio que falla
dscacheutil -q host -a name <dominio>.apple.com | grep ip_address

# En AMBOS servidores: ruta /32 específica
push "route A.B.C.D 255.255.255.255 net_gateway"
```

Reiniciar y reconectar para aplicarla. Es el mismo mecanismo; la receta genérica vale para cualquier destino problemático (un banco que bloquee IPs de datacenter, un servicio que exija IP doméstica, etc.).

Nota sobre IPv6

Estos servidores hacen `push "block-ipv6"`, así que con el túnel activo todo va por IPv4 y la exclusión `17.0.0.0/8` cubre a Apple completa. Si algún día se retira el `block-ipv6`, la exclusión solo cubriría el IPv4 de Apple; habría que valorar la contrapartida IPv6 (`route` IPv6 con `net_gateway` o bloqueo selectivo).

Nota sobre SplitTunnel y su herencia en macOS

Si se probó SplitTunnel (boundarylabs) y se desinstaló arrastrándolo a la papelera, quedan restos que CleanMyMac no ve:

- **Extensiones de sistema registradas** (`systemextensionsctl list`): `ca.boundarylabs.SplitTunnel.Extension` y `ca.boundarylabs.SplitTunnel.DNSExtension` quedan como `[activated disabled]`. En ese estado no cargan código ni interceptan nada, pero siguen listadas.
- **No se pueden des-registrar por CLI con SIP activado**: `sudo systemextensionsctl uninstall <teamID> <bundleID>` responde «cannot be used if System Integrity Protection is enabled». La vía soportada es reinstalar la app y usar SU desinstalación (deactivation

cooperativa), o vivir con el registro fantasma, que es inocuo.
- Restos de usuario sí se limpian a mano: `~/Library/Group Containers/group.ca.boundarylabs.*`, `~/Library/HTTPStorages/ca.boundarylabs.*` y `~/Library/Application Scripts/group.ca.boundarylabs.*`.

Revision #1
Created 2026-08-27 19:18:15 UTC by Abkrim
Updated 2026-08-27 19:18:15 UTC by Abkrim