How and why to VPNize your entire traffic

Introduction

UPDATE October 2023: A few days ago, a friend awared me about a script that generates OpenVPN config files for RiseUp from the Bitmask database, so I got inspired and decided to update this. This guide will teach you why you should encapsulate all your Internet traffic inside a VPN tunnel, and then teach you how to do it. So let's go:

The Why

Why even bother with a VPN? Imagine having a "citizen identification number" engraved on your forehead. This number would be recorded by CCTV every time you left the house and by every subsequent CCTV camera you encountered. And so, whoever operated the cameras would be able to know what you did and when; assuming they cooperated, a file on your entire life could be generated. Imagine the power it gives someone; anything you've ever done could

be used against you in any place you visited, to give you a different (worse) service or outright deny you entrance. Now imagine that whoever operated the cameras also had access to entities that could fine or arrest you for anything found in the footage, and it's obvious you couldn't ever feel comfortable doing anything ever again.

What if I told you that this is **exactly what happens with default Internet connections**? Anytime you visit a website, your ISP (the "camera operator") knows where you went. Since your IP address (the "citizen identification number") is connected to your real name and place of residence, your ISP can connect your entire browsing history to who you are. Worse than that - if the target website reports you, the ISP can also know exactly what actions you took on it. And it does have the authority to report you to the police, too. Since you don't control the law, you have no idea how any of of your Internet activities might be used against you in the future. Who wants to be fined, have their hard disks stolen, and / or be dragged to courts?

Using a VPN **would prevent all this**. In the CCTV analogy, it would be like taping someone else's number on your forehead after leaving the house. The camera operator would know that you've left the house and turned into someone else; but the destination places would think you're that other person, and their eventual reports would affect that other person and not you. The same way, the ISP would only know that you've connected to a VPN; the target destination and actions taken on it would be performed as another entity. Though the website itself would still be able to follow you around during your stay on it, the CIN (IP address) is now changed, so the ties to the ISP (the entity that has your real name and address) are severed. Note: the CCTV analogy isn't perfect and there are important caveats; but, it works for the purpose of understanding the problems caused by barebacking your ISP.

Of course, there is a big potential problem with the VPN setup; namely that it relies on the VPN

(or that "other person" in the CCTV analogy) being trustworthy enough to not be tracking and reporting you, too. So it's a matter of picking one with a reputation; the strongest evidence is having news reports of raids that resulted in no logs found (such as we have for Mullvad (archive) (MozArchive) and ExpressVPN (archive) (MozArchive)), and / or having won court cases (archive) (MozArchive). Remember that situations can change, so it's important to keep yourself updated with the latest information. Even in the worst case of a honeypot VPN that reports everything to your ISP, it could still be used to bypass discrimination by country of origin on target sites (e.g the banning of EU citizens), and clear forum bans. And let's be real - there's an almost 100% chance your ISP is storing all your activity and will gladly cooperate with the police. It also has your real name and address, while the VPN doesn't. So even shooting in the dark with VPNs is a better deal than just barebacking your ISP.

The only free VPN provider worth anything is RiseUp. Of course, you can always **pay for a good one**; if you do go this route, research well. The logging policy isn't the only criterion; you want VPNs that accept cryptocurrency, support OpenVPN and / or Wireguard instead of requiring custom apps, are fast enough and have enough servers for you to be able to bypass all blocks, and don't have arbitrary bans on torrenting, etc. There are more minor criteria like Mullvad allowing you to share your VPN with four friends; but this isn't a review of VPNs, so let's move on to the more important part - configuring our system to send all traffic through the RiseUp VPN (the guide can be adapted for any other one, though):

The How

By the way, I did retest the instructions after the rewrite, and they work. Except I'm using TorDNS, so the part about the DNS might not work. Will update later. **UPDATE May 2026**:

this guide is **only for systemd-free distros**. Don't expect much of it to work elsewhere. For one, systemd-resolved controls the DNS in such distros, so editing resolv.conf will do nothing. And according to a friend, */etc/systemd/systemd-resolved.conf* also gets overwritten on OS restart. TorDNS also requires additional software and files to edit, because systemd-resolved steals its port... Just don't bother with systemd at all if you can - it only adds complexity and breakability to the setup. Moving on to the instructions:

1. First of all, install the *openvpn* package if you don't have it yet. It requires some dependencies, such as *iproute2* - so get that handled, as well.
2. Then, get an OpenVPN config file from your chosen VPN provider and put it in */etc/openvpn*. If you're using the RiseUp script mentioned at the beginning, just put it in */etc/openvpn*, then run *generate.sh*.
3. Again, if you're using the above script, edit the resulting config file (*riseup-ovpn.conf*) to remove all but one of the *"remote"* lines (any one works), and the *"remote random"* should be deleted, too. This is not strictly necessary, but will make the following steps easier and consistent with other VPNs.
4. Now we will need to set up some firewall rules which prevent your real IP address from leaking. Install the *ufw* package if you don't have it yet.
5. In the config file, find the line that starts with *"remote"*. Take note of the IP and port. Now type this into terminal: *"sudo ufw allow out to [IP] port [PORT]"*. Of course replace IP and PORT with the relevant values. This will let the system connect to the VPN through the firewall. You are left with a single IP address unconnected to your usual identity if the VPN doesn't snitch and your OPSEC is right. If you wanted a randomly picked IP from that RiseUp script, you'd have to allow all of them separately here. It is doable, I just didn't want to make this setup harder than it needs to be.

6. Now find the line starting with *“dev tun”*. Change the *“tun”* to something recognizable, like *“tun_myvpn”*.

7. Type these two rules into terminal: *“sudo ufw allow in on tun_myvpn”* and *“sudo ufw allow out on tun_myvpn”*. This will allow both incoming and outgoing connections through the VPN.

8. Now type *“sudo ifconfig”*. Take note of the IP that appears after *“inet”*. This is your local (router) IP.

9. Allow it through the firewall like this: *“sudo ufw allow out to [LOCAL_IP]”*. This will enable actually establishing the VPN connection.

10. To set up your system to use the VPN's DNS servers instead of your ISP's, type *“sudo resolvconf -l”*. Now copy the nameservers and put them into */etc/resolv.conf* (*“nameserver 172.27.0.1”* for RiseUp, for example). Without this step, your ISP will still know every site you visit. This is not strictly necessary and can be replaced by TorDNS - though this way is easier.

11. Now make */etc/resolv.conf* unmodifiable, either by *“chattr +i”* or putting *“nohook resolv.conf wpa_supplicant”* into */etc/dhcpcd.conf* (my preferred method). This will prevent the system from overwriting your VPN's DNS servers with the ISP's.

12. Finally, allow the VPN's DNS servers through the firewall; as before - *“sudo ufw allow out to [DNS_IP]”* (you've just typed the addresses into resolv.conf, so just allow all those). Without this step, you would not be able to connect to any domain unless you knew their actual IP address (since we've blocked the ISP's resolver). This step is not necessary if using TorDNS.

13. All that remains is to block everything except what we've just specified. *“sudo ufw default deny incoming”* and *“sudo ufw default deny outgoing”*. This is the part that actually keeps your shit secure (anything that tries to send something outside the VPN tunnel, won't

be allowed to).

14. To enable the firewall on your system's startup, add this code to */etc/rc.d/rc.local*:

if [-x /lib/ufw/ufw-init]; then
/lib/ufw/ufw-init start
fi

This is for Slackware-based distros and might not necessarily work on others. Search around for equivalents.

That's it for OpenVPN! However, web browsers can also leak your real IP address through WebRTC, so you're going to have to disable that as well. Firefox uses the *"media.peerconnection.enabled"* about:config entry, while Chrome-based browsers need an extension such as WebRTC Control (Pale Moon users **do not need to do anything**). An earlier version of this guide suggested turning off IPv6 system-wide, but it doesn't seem to be necessary if you do everything else right. However, some VPNs apparently do leak if you don't do that, so if yours is one of those, do all these steps just to be safe (earlier version had only step 1, but it seems it's **not always sufficient**):

- Type the command *"echo 'blacklist ipv6' | sudo tee -a '/etc/modprobe.d/blacklist.conf' >/dev/null "* to blacklist the IPv6 kernel module.
- Edit the */etc/sysctl.conf* file and put these lines in it (assuming your interface is wlan0 - **use ifconfig to confirm**):

net.ipv6.conf.wlan0.disable_ipv6 = 1
se net.ipv6.conf.all.disable_ipv6 = 1

net.ipv6.conf.default.disable_ipv6 = 1

- Type the command *"sudo sysctl -p"* to load the changes (should be valid immediately)

Now run your VPN with a command such as *"cd /etc/openvpn; sudo openvpn [vpn_config_file.conf]&"*. Then visit https://ipleak.net to check for leaks. A leakless result for RiseUp VPN, for example, would look like this. If it worked, you can put that command inside your */etc/rc.d/rc.local* after the firewall; it will load the VPN automatically on system startup.

An alternative to the VPN setup is the darknet-only setup. Its advantages include much less reliance on trust and being able to visit onion websites. Disadvantages are harder setup, slower speeds, and being blocked by way too many entities. Maybe the best way to proceed is to use the VPN setup generally, but add TOR after it for the more sensitive stuff (using eg Proxy Privacy Ruler or proxychains). An important advantage of doing it this way is that the ISP also **won't know you're using TOR** (unlike with a pure-darknet setup) and will not be able to block it if you're in one of the countries that do that. And, you can choose exactly how paranoid you want to be - dropping TOR for only torrents, or everything but onions. This is what I did for years before my Mullvad expired (and also stopped supporting Pale Moon on their site, preventing me from paying), when I switched to the darknet-only setup.

When RiseUp stops working

You might sometimes encounter the RiseUp VPN going down, and not seeming to recover quickly (or at all). Why is that? Well, the certificate inside the RiseUp config file only lasts for a limited time (I think it's a month). When that happens, first **rename your current config file** (so

that you won't lose the modifications we've made). Then simply run *generate.sh* again. From the resulting config file (not the one which you renamed; it will be again named *riseup-ovpn.conf*) copy the part between *<ca>* and the end of file. Now insert that inside your old config file (whatever you renamed the original to) over the existing cert definition block. Then delete the newly downloaded config file, and rename the old one back to *riseup-ovpn.conf*. This basically replaces the existing expired cert with a new one, but keeps all the modifications we've done to the default config file. This - of course - isn't the only reason a VPN could be down for an extended length of time; but in terms of RiseUp, it's the most common for me, by far. Watch the errors the console spits, if this doesn't work.

Back to the front page