Play LAN-only games together over the Internet with WireGuard

geekoverdose Misc April 18, 2020 7 Minutes

This post describes how to configure WireGuard (https://www.wireguard.com/), an open-source point-to-point tunnel (https://en.wikipedia.org/wiki/WireGuard), to play LAN-only games together over the Internet. This includes games that never had Internet-based but only LAN-based match-making in the first place (mostly older games) – but also games that by design would have had both, but for which the Internet-based match-making servers sadly are no longer active.

Note that in this post we're going to set up the WireGuard point-to-point tunnel directly between the machines that run the game, and port-forward WireGuard traffic to those machines. As most people run games on Windows, and as WireGuard now also features Windows installers, the examples below use the WireGuard Windows version. However, the same config is of course also valid for other platforms.

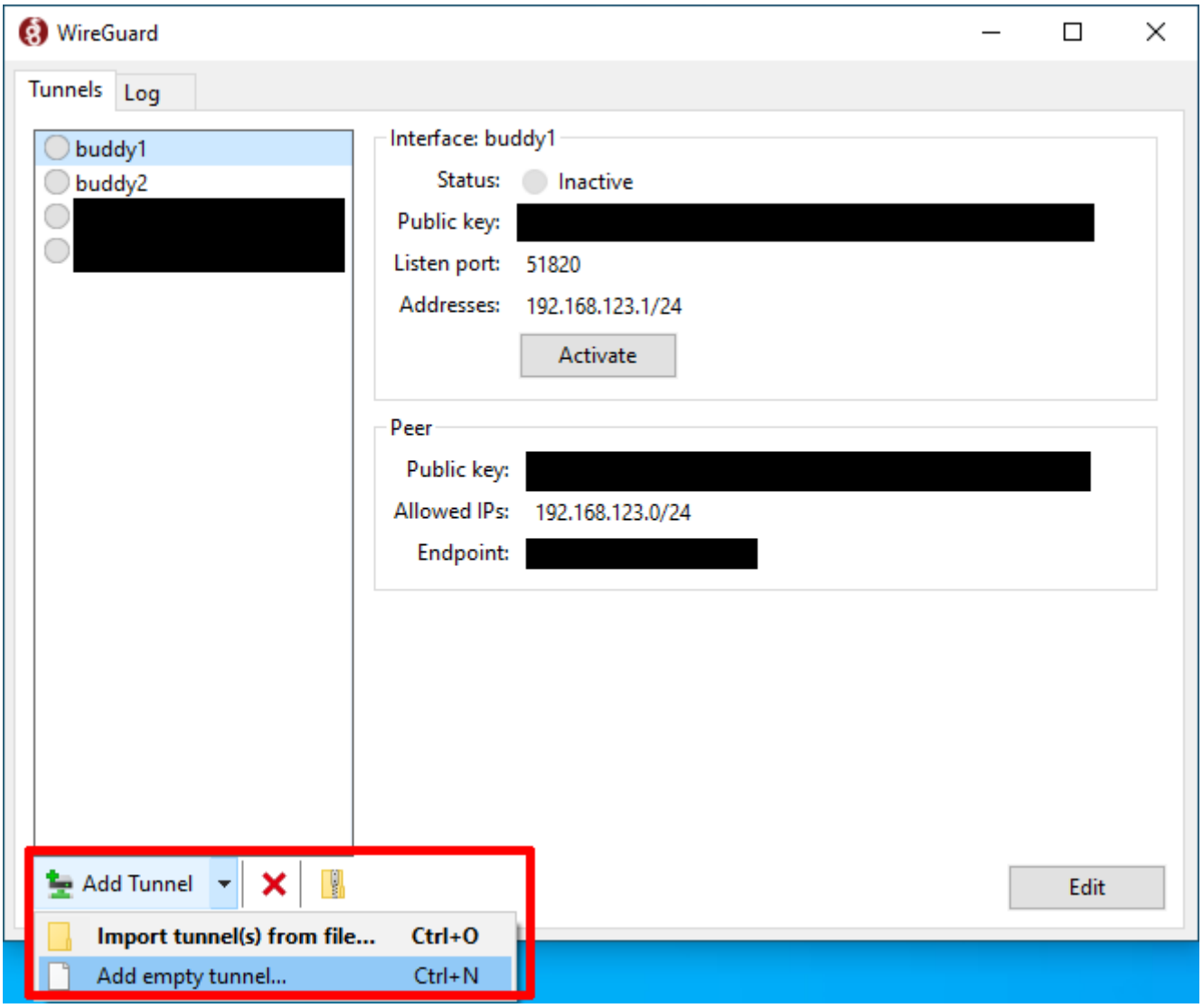
TL;DR instructions

1. **Port forwarding:** Ensure both machines are reachable from the Internet on port 51820 UDP (you might need to configure a port forwarding on the corresponding routers).
2. **Dynamic DNS (optional):** consider using a dynamic DNS service on both sides, so that you don't need to update the WireGuard configs below with the possibly changing public IPs of both sides.
3. **WireGuard config:** Install WireGuard and configure the tunnel interfaces on both machines (see details in step 3 below). The tunnel interfaces need to be in the same IP subnet and need to listen to traffic on port 51820 (ListenPort=...). Each side also needs a config for their peer (the other machine) with address, port, and public key.
4. **Connection status:** Toggle tunnels on. Check that logs on both sides shows a successful handshake incl. responses, and then periodic keepalive messages. Optional: on both sides allow ping (ICMP) responses in Windows, then ping the tunnel adapter address of the other side.
5. **Connect games:** In games, the first machine hosts the LAN game, then the other machine connects its game directly to the tunnel interface address of the first machine. Broadcasts (game matching over OSI-2) does not work over WireGuard or similar OSI-3 VPN solutions.

Step by step instructions

Please note that I assume you have a public IPv4 address. I use IPv4 addresses throughout the instructions, as I happen to have a public IPv4 address myself atm, which allows me to test and verify that the instructions are working as intended. While the same is also possible with IPv6, I've not covered it in the instructions at all, simply because I can't test and verify it myself atm.

1. **Port forwarding**
   - On both sides, if routers are used, those need to forward port 51820 UDP from the Internet to the machines that should be able to play LAN games together. The WireGuard installations on both machines later communicate with each other over that port, hence need to be able to reach each other.
   - 51820 is the standard port for WireGuard. You can instead also choose any other port and consistently use it in all config below. For this post we're going to stick to 51820 for simplicity reasons.
2. **Dynamic DNS (optional)**
   - In step 3, each side needs to state the address of their peer in their config. As most people will have dynamic IP addresses for their home internet from their Internet providers, those addresses will change over time. This means that the corresponding config needs to updated frequently, like each before playing together again.
   - To avoid this, dynamic DNS services can be used. Check out which (free) dynamic DNS services your router supports, create an account with one of those services and choose a free dynamic DNS domain there that you want to use, and then configure the routers dynamic DNS address updates with that account and domain. Once the dynamic DNS updates work correctly this allows you to tell your peer your chosen dynamic DNS domain instead of your current public IP address.
   - If you do not use a dynamic DNS service, you can still use WireGuard. However, you then need to check what your current public IP address is and tell it to your peer, who then needs to update their config with that address. You can find out what your public IP address from services like getfoxyproxy.org/geoip/ (https://getfoxyproxy.org/geoip/) or whatismyip.com (https://www.whatismyip.com).
3. **WireGuard config**
   - Install WireGuard (https://www.wireguard.com/install/) for Windows on both computers. If you have a different platform: just install WireGuard there and apply the same WireGuard configuration below.
   - On both sides: start WireGuard. If Windows firewall asks you if you want to allow traffic through the firewall: this is about incoming traffic, hence, if someone from the outside – like your peer – can reach this program. You need to allow it (it should be safe to allow it for all network types).
   - On both sides: create a new empty tunnel and give it a name.



Add an empty WireGuard tunnel

This already auto-generates your public and private key. Don't change those! Instead:

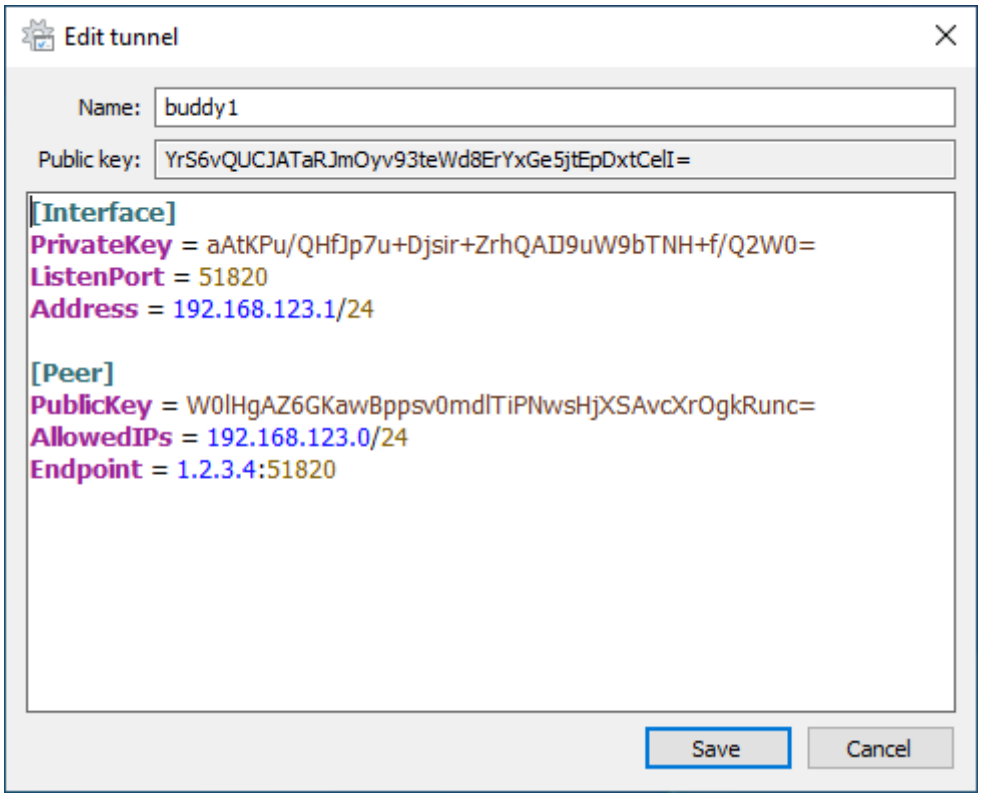
- Change the [Interface] section of the configuration:
  - Add the local address of the tunnel interface of this machine. This is going to be Address = 192.168.123.1/24 on one machine, and 192.168.123.2/24 on the other machine.
    - *Note that 192.168.123.x/24 is one subnet: this is the LAN subnet you will use for gaminglater on! Yay!*
  - Add which port this tunnel interface is listening on: ListenPort = 51820 (same on both sides).
- Add a [Peer] section to the configuration:
  - Add the public key of the other peer on both sides: PublicKey = ... For this you need to receive the public key from your buddy by any communication means, and vice versa.
    - *A note on public and private keys (https://en.wikipedia.org/wiki/Public-key_cryptography):the public key can be public = anyone in the world can know it, no need to keep it secret. However, do not confuse the public key with the private key: the private key **needs** to stay secret. No one else besides yourself should know it, ever – also not your buddy. You should also not know any private keys except your own. Anyone who should ever obtain this private key would be able to eavesdrop traffic you receive, and also imitate traffic in your name.*
  - Define which IP addresses the peer can access on your machine. It's meaningful to allow traffic within the whole subnet: AllowedIPs = 192.168.123.0/24
  - Define the endpoint of the peer: EndPoint = 1.2.3.4:51820 . Each side needs to replace

1.2.3.4 with the address of their peer. Hence, this can either be the public IP address of the peer, or the dynamic DNS domain they have configured before. Do leave the :51820 as it specifies on which port the peer is listening for incoming WireGuard traffic from the Internet.

- This should result in a config similar to the following one (with the public and private keys replaced with your corresponding ones, and also 1.2.3.4 and 5.6.7.8 in the peer's endpoint addresses replaced with your corresponding ones). Double check on both sides that the config really is correct, and that both sides really use the correct public key of their peer:
  - On machine 1:

```
[Interface]
PrivateKey = aAtKPu/QHfJp7u+Djsir+ZrhQAIJ9uW9bTNH+f/Q2W0=
ListenPort = 51820
Address = 192.168.123.1/24

[Peer]
PublicKey = W0lHgAZ6GKawBppsv0mdlTiPNwsHjXSAvcXrOgkRunc=
AllowedIPs = 192.168.123.0/24
Endpoint = 1.2.3.4:51820
```

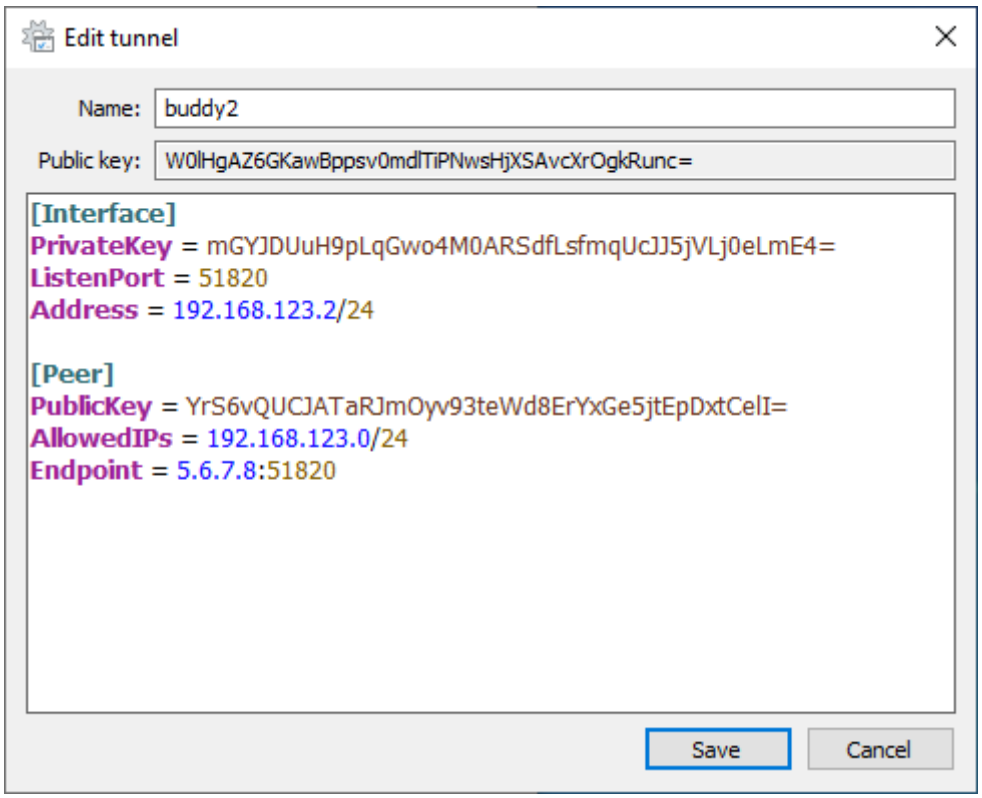


WireGuard configuration on machine 1

  - On machine 2:

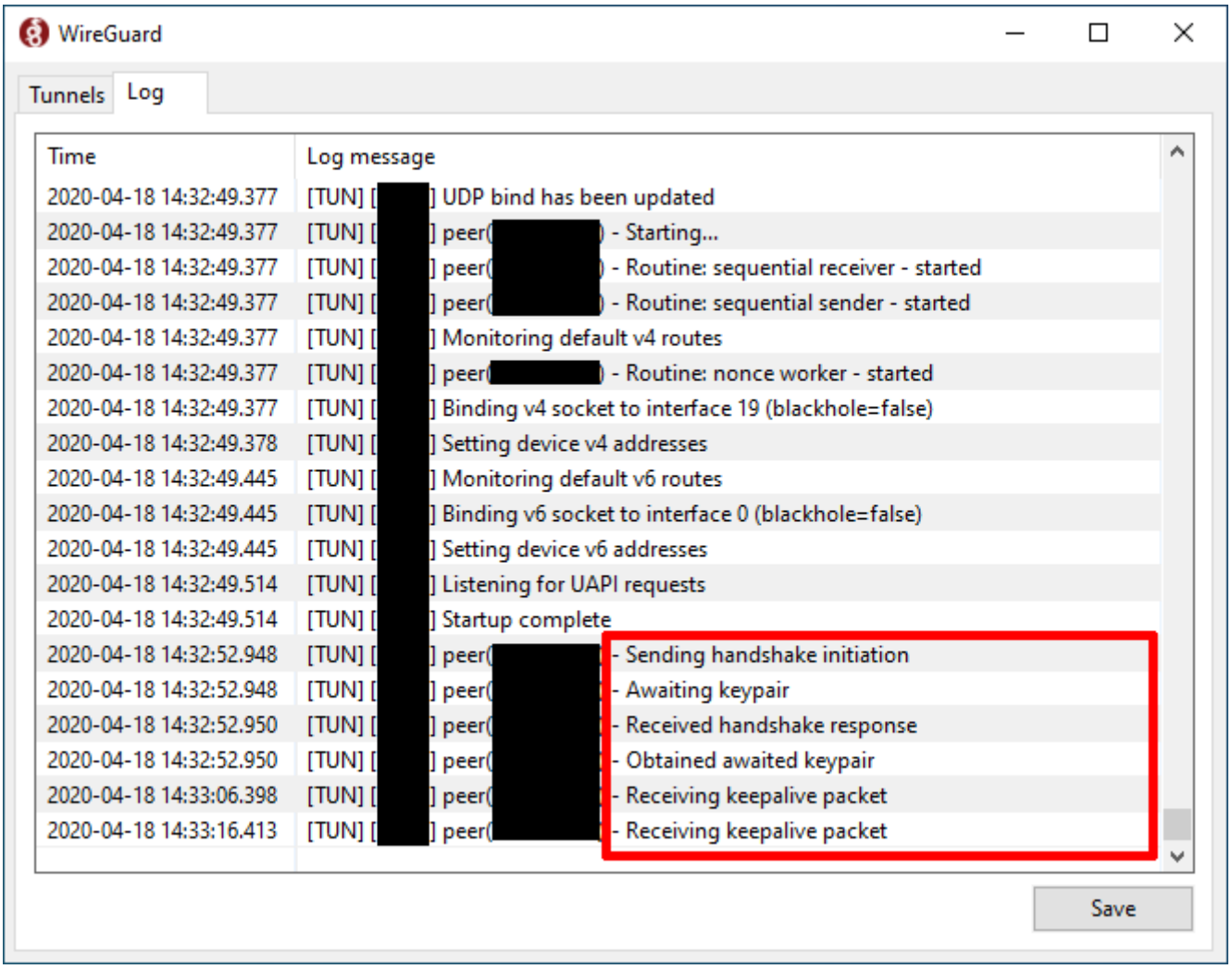
```
[Interface]
PrivateKey = mGYJDUuH9pLqGwo4M0ARSdfLsfmqUcJJ5jVLj0eLmE4=
ListenPort = 51820
Address = 192.168.123.2/24

[Peer]
PublicKey = YrS6vQUCJATaRJmOyv93teWd8ErYxGe5jtEpDxtCelI=
AllowedIPs = 192.168.123.0/24
Endpoint = 5.6.7.8:51820
```



WireGuard configuration on machine 2

4. **Connection status**
   - On both sides: toggle the tunnel to “On”.
   - On both sides: verify that the log shows a successful handshake, and that periodic keepalive messages are sent afterwards. If you do not see this, the tunnel does not work correctly, an the subsequent steps cannot work. If you see the handshake and keepalive messages: well done, your setup is pretty much finished! 🙂



The log shows a handshake and keepalive packages

- Optional: ping tests
  - If you are using Windows 10 you at first need to allow ping (ICMP) responses on your computer. There are many instructions for this (see e.g. this one (https://www.faqforge.com/windows/windows-10/how-to-allow-ping-trough-the-firewall-in-windows-10/)). Verify that pinging your own tunnel adapter (machine 1: `ping 192.168.123.1`, machine 2: `ping 192.168.123.2`) results in responses being sent. If you don't get responses from pinging your own adapter, your peer of course will also not get responses, even if the tunnel is working correctly.
  - Ping each other's tunnel adapter (machine 1: `ping 192.168.123.2`, machine 1: `ping 192.168.123.1`). If you see ping responses, then the two computers can talk to each other on exactly those addresses, like they would be in a LAN 🙂 (However, note the OSI-lvl 2 vs OSI-lvl 3 problem in step 5).

5. **Connect games**
   - Wireguard, as many other similar solutions, works on OSI-lvl 3. In a real LAN, in which computers are physically connected via a switch, OSI-lvl 2 broadcasts are available as well. Those cannot be covered by design by WireGuard, as it works on lvl 3 instead (note that a lvl 2 tunnel is also significantly more complex from a technical POV). For this reason games need to directly connect to their peers, instead of using a broadcast-based match-finding and making-making approach.
   - One machine starts the LAN game.
   - The other machine directly connects to the IP address of this LAN game. With the WireGuard tunnel this means: connect to the tunnel interface IP address of the peer (machine 1: connect to `192.168.123.2`, or, if it's machine 2: connect to `192.168.123.1`).

Have fun playing your LAN-games over the Internet together! 🙂

Notes on performance and reliability

I'm still flashed by the performance of the WireGuard point-to-point tunnel. In my tests it turned out to be really stable and fast – better than related solutions. The latency is really low, noticeably lower than with other solutions – which is of course really good for games. And I did not experience a single connection breakup in multiple games I tried for a while. Altogether I'm flashed by the tunnel performance and also the config simplicity, compared to so many other solutions out there.

Alternatives to a WireGuard machine-to-machine tunnel

Instead of configuring a WireGuard machine-to-machine tunnel, you can also configure WireGuard on your router(s) – which is a little bit more complicated than the setup described in this post. If you are interested in this: take a look at, for example, the OpenWRT support for WireGuard (https://openwrt.org/docs/guide-user/services/vpn/wireguard/start).

Tagged:
configuration,
firewall,
games,
hamachi,
lan,
port,
setup,
tunngle,
vpn,
wireguard

Published by geekoverdose

Rainhard Findling: geek, curious, and full of energy! Details: https://geekoverdose.wordpress.com/about/ View all posts by geekoverdose

One thought on “Play LAN-only games together over the Internet with WireGuard”

Pingback: Windows firewall rule to block Internet but allow LAN – geekoverdose