wg-client

Release 7.1.1

Gene C

Jan 06, 2026

CONTENTS

CHAPTER

ONE

WG-CLIENT

1.1 Overview

Linux wireguard client tools make it simple to start and stop wireguard. Comes with command line program, *wg-client*, and a convenient GUI tool which depends on it.

This is a companion to the wireguard server config tools wg-tool.

Also provides an option to invoke ssh which creates a remote listening port connected back to local ssh daemon.

This can be useful to facilitate remote ssh back to client computer if it's needed. For example; it can be used to provide access to a git repo on the client, or for remote backups of laptop, or even for admin to login to client should the need arise.

There is a command line program (*wg-client*) and for user convenience there is a GUI program (*wg-client-gui*) which is available via a *.desktop* file.

The graphical tool invokes the command line tool, and the command line tool does all the real work. The GUI provides user convenience.

All git tags are signed with arch@sapience.com key which is available via WKD or download from https://www.sapience.com/tech. Add the key to your package builder gpg keyring. The key is included in the Arch package and the source= line with *?signed* at the end can be used to verify the git tag. You can also manually verify the signature

1.2 Why I made wg-client

After building wg-tool which simplified administering wireguard servers, I needed a simple way for non-tech users to connect their laptops to the server.

Thus wg-client was born. The gui client makes it simple for non-tech users, though I find it convenient too.

1.3 Key features

- Graphical tool makes it simple for any user to get VPN running.
- Standalone tool makes it easy to test and also keeps sudo outside of gui to minimize any security implications. The gui relies on command line to do the real work.

1.4 New or Interesting

7.0.0

- Code Reorg
- Switch packaging from hatch to uv

- Add source checks for C-code as well as python
- Testing to confirm all working correctly on python 3.14.2
- C-code (wg-fix-resolve): Use more 'const' pointer to structs to improve security.
- Modern coding standards: PEP-8, PEP-257 and PEP-484 style and type annotations
- Use pyconcurrent module (additional dependency)

  Available on Github and AUR

Older

- Change wg-fix-resolv: Ignore comments when comparing resolv.conf files. More efficient/correct when only change is a commented time stamp for example.
- resolv monitor: Increase maximum time to wait for wireguard to start before running the monitor. No reported issues with 5 seconds - no harm in being able to wait a bit longer if needed for some reason.
- Improve logging when ssh listener exits. Nicer format for how long ssh has been running when it is auto restarted.
- ssh listener is now auto restarted if it exits unexpectedly. There are normal, quite common situations where ssh process can exit prematurely. For example:
  - After sleep/resume (longer than tcp timeout)
  - if remote server sshd restarts (reboot for example)
  - changing IP address (happens when location changes. e.g. Move from hotel to starbucks)

  Code now detects this and automatically restarts the ssh listener.

  This is now much more convenient for the user. wireguard itself is robust against the same changes since it uses udp, and now just start vpn and start ssh and the app will handle keeping everything running. If ssh cannot reconnect, it waits a while and tries again.
- Auto fix of resolv.conf (new option *–fix-dns-auto-start*)

  Network refresh often happens after sleep/resume (e.g. laptop lid close/open) or when a DHCP lease expires. If VPN is up and running when this occurs the /etc/resolv.conf file can be reset and then DNS will no longer use the vpn DNS but will then use whatever resolver DHCP provided by default. Earlier versions of wg-client offered a manual fix available by clicking the *VPN Start* button again or by using wg-client on command line.

  This is now done automatically using a daemon which can be started/stopped from command line using the new options *–fix-dns-auto-start* and *–fix-dns-auto-stop*

  NB: The GUI app calls *wg-client* to start the monitor daemon when it starts up wireguard.
- NB version 5 has 2 additional dependencies:
  - openssl library for wg-fix-resolv.c
  - python-pynotify library available via Pynotify Github and Pynotify AUR
- dns resolv.conf fix now uses capabilities.

CHAPTER

TWO

GETTING STARTED

2.1 wg-client application

2.1.1 Usage

To use run from a terminal. For example to start wireguard from a terminal, use:

```
wg-client --wg-up
```

To get a list of options run it with *-h*. Options are also documented in config section *config-sect* below.

2.1.2 DHCP refresh & sleep / resume

When laptop sleeps, from lid close for example, and then woken up - the vpn will continue working as normal and likewise the ssh provided the sleep time is not *too long*. However, on wake the networking is typically re-initialized and part of that may re-install the dns resolver file */etc/resolv.conf*.

This is handled automatically by the resolv monitor daemon. See the option *–fix-dns-auto-start* for more information.

2.1.3 Configuration

wg-client reads its configuration from

```
/etc/wg-client/config
```

Please copy the sample config and edit appropriately. The format is in *TOML* format.

```
# Sample
iface = 'wgc'
ssh_server = 'vpn.example.com'
ssh_pfx = '47'
```

This config file provides:

- iface - required

  Wireguard interface; defaults to *wgc*. It is *<iface>* of */etc/wireguard/<iface>.conf*

- ssh_server - optional

  Hostname of the remote ssh server accessible over the vpn; this is where the ssh listening port is run. Hostname must be accessible over the wg vpn.

- ssh_pfx - used with ssh_server

1 or 2 digit number, 65 or smaller, to be used as ssh listening port number prefix. The port number is of the form PPxxx, with *PP* the prefix and *xxx* is taken from the last octet of the wireguard vpn internal IP address.

The prefix can also be given as a range of numbers (*'n-m'*). In this case the prefix used is randomly chosen from that range

Keep in mind that the largest port number is 65535, which limits *ssh_pfx* to be 65 or lower.

The port number chosen will be written to the log file.

The remote ssh host will then listen on *127.0.0.1:<port>*. It will also listen on *<remote-ip-address>:<port>* provided the remote ssh server permits it by having the sshd option set:

```
GatewayPorts yes
```

2.1.4 Options

Summary of available options for wg-client.

Positional argument : Optional

- wireguard client interface name

  Default interface is taken from *iface* in config file. The config file is chosen by first checking for *./etc/wg-client/config*[1] and then in */etc/wg-client/config*. If not found the wg interface defaults to *wgc*

Options:

- (*-h, –help*)

  Show this help message and exit

- (*–wg-up*) and (*–wg-dn*)

  Start and stop wireguard client

- (*–ssh-start*)

  ssh to remote server over vpn and listen on remote port. Port number used is described above in Overview section *config-sect*.

  This blocks waiting for ssh. To stop ssh, simply make a separate invocation of *wg-client -ssh-stop*. If using the GUI tool, simply click the *Stop Ssh* button.

  In the event that ssh connection is dropped, it will automatically be restarted. There are normal, quite common situations where ssh process can exit prematurely. For example:

  – After sleep/resume (longer than tcp timeout)

  – if remote server sshd restarts (reboot for example)

  – changing IP address (e.g. happens when location changes. e.g. Move from hotel to starbucks)

  While attempting to reconnect ssh there is a waiting period between each attempt. Currently the wait time is 30 seconds.

- (*–ssh-stop*)

  End ssh to remote server

- (*–ssh-pfx*)

  Set the ssh port prefix. Can be 2 digits: "nn" or a range "nn-mm". If using a range, then prefix will be randomly drawn from the range. Maximum value is 65. This can also be set in the config file.

[1] Useful during development and testing

- (*–ssh-server*) <server>

  Remote ssh server to set up listening port. This is usually set in the config file.

- (*–fix-dns*)

  This has been automated by the monitor daemon. See *–fix-dns-auto-start*

  Restore wireguard dns resolv.conf. Typical use is after sleep resume when the network is set up it can mess up the resolv.conf file - this restores the correct version.

  This will also be done by GUI, if needed, by simply clicking the Start VPN button.

  wg-client relies on *wg-fix-resolv* program which is granted CAP_CHOWN and CAP_DAC_OVERRIDE capabilities to enable it to restore the right /etc/resolv.conf file.

- (*–fix-dns-auto-start*)

  Auto fix of resolv.conf.

  Please note that this is *always* run by the GUI program. This option is only relevant when not using the GUI.

  Network refresh happens after sleep/resume (e.g. laptop lid close/open) or when a DHCP lease expires or when changing network locations (such as moving from hotel to starbucks).

  If VPN is running when this occurs the /etc/resolv.conf file can be reset and then DNS will no longer use the vpn DNS.

  (While older versions of wg-client provided a manual fix available by clicking the *VPN Start* button again or by using wg-client on command line, current versions monitor and fix resolv.conf automatically)

  N.B. We must coordinate with wireguard config treatment of /etc/resolv.conf when it is started and stopped.

  There are 3 relevant files :

  – /etc/resolv.conf

    When wireguard is not running it the usual one. When wireguard is running it is the one installed by wireguard PostUp configuration so that DNS requests use the vpn tunnel.

  – /etc/resolv.conf.wg

    When wireguard starts, PostUp must save a copy of the resolv.conf to be used while wireguard is running.

  - /etc/resolv.conf.saved

    This is a copy of the usual resolv.conf to be used when wireguard is not running.

  wireguard responsibility

  When wg-client starts wireguard, wireguard itself must save the current */etc/resolv.conf* and install one that uses the vpn tunnel.

  We expect wireguard PostUp and PostDown scripts configured to do the following:

  - save existing /etc/resolv.conf as /etc/resolv.conf.saved
  - install wireguard /etc/resolv.conf and also save into /etc/resolv.conf.wg

  This is what *wg-tool* does in the default configuration.

  With that in mind *wg-client* will monitor /etc/resolv.conf while wireguard is running, and if it changes (typically network tools can do this when DHCP renews or if changing networks etc.), then the monitor daemon copes that new resolv.conf into resolv.conf.saved. This resolv.conf is what is needed when wireguard is not running. When wireguard is stopped, it's PostDown script will restore /etc/resolv.conf from the saved version. So it's important to keep that saved version current so networking works normally after wireguard is stopped.

Similarly, when /etc/resolv.conf is replaced by networking tools, it is the wrong one to use for wireguard - so monitor copies /etc/resolv.conf.wg into /etc/resolv.conf to ensure the correct resolv.conf is used while wireguard is running.

wg-client responsibility

With that all said, this is what the dns auto fix tool is responsible for while wireguard is running:

- if /etc/resolv.conf changes - save it to /etc/resolv.conf.saved
- restore /etc/resolv.conf from /etc/resolv.conf.wg

This is now done automatically using a daemon which can be started/stopped from command line using the new options *–fix-dns-auto-start* and *–fix-dns-auto-stop*

The GUI app does this whenever it starts wireguard.

The monitor daemon watches */etc/resolv.conf* and auto restores the correct one when needed. It uses inotify whereby the kernel notifes us when the file changes - this is very efficient and allows the monitor to sleep waiting for the kernel to wake it up when there's something to do.

Wireguard will continue to work even if the laptop is taken to a new wifi location. The monitor checks and saves any newly found resolv.conf and restores the wireguard one. Of course on closing down, the original saved resolv.conf is restored as well. Note that ssh will not survive changing networks but it can easily be restarted and the code will automatically reconnect if possible.

- (*–fix-dns-auto-stop*)

  Stops the monitor daemon.

- (*–show-info, –status*)

  Report all info. If run as root then it additionally shows staus of any ssh/resolv monitor for all users.

- (*–show-iface*)

  Report wireguard interface name used.

- (*–show-ssh-server*)

  Report the ssh server name

- (*–show-ssh-running*)

  Report if ssh is active

- (*–show-wg-running*)

  Report if wireguard is active

- (*–show-fix-dns-auto*)

  Report if auto fix dns is running

- (*–test*)

  Test mode - print what would be done rather than doing it.

- *–version*

  Display wg-client version

2.2 wg-client-gui application

2.2.1 GUI Usage

The gui is installable using the provided wg-client.desktop file and can be added to launchers in the usual way. For example in gnome simply search applications for wg-cliient and right click to pin the launcher. The gui uses PyQt6 which in turn relies on Qt6.

The gui has buttons to start and stop wireguard and a button to run ssh to set up the listener on the host configured in the config file.

The gui should be left running while the vpn is in use. Pressing quit in the gui will shutdown wireguard and shutdown the ssh listener as well.

2.2.2 GUI Options

wg-client-gui has no command line options. It invokes *wg-client*, and thus the configuration described above *config-sect* is used:

```
/etc/wg-client/config
```

2.3 Log files

Each application has it's own log file. These are located in users home directory :

```
${HOME}/log/wg-client
${HOME}/log/wg-client-gui
```

Each of the log files are rotated with companion log suffixed with *.1*

2.4 Sudoers

wg-client uses *wg-quick* from wireguard tools to start and stop the vpn. and since this requires root to do it's job, any non-root user will need a NOPASSWD sudoers entry.

You can keep all local sudoers in a single file or in separate files. If in single file, make this one come after any group wheel ones. This is to ensure this one is chosen becuase sudo uses the last matching entry.

Simply add this sample line adjusting WGUSERS to list whatever user(s) are permitted to run wireguard. If more than one use comma separated list as shown below.

```
User_Alias WGUSERS = alice, bob, sally
WGUSERS   ALL = (root) NOPASSWD: /usr/bin/wg-quick
WGUSERS   ALL = (root) NOPASSWD: /usr/lib/wg-client/wg-fix-dns
```

If using separete files, then care is need to ensure this entry comes after any wheel group entries. Where WGUSERS is 1 or more usernames or a group such as *%wgusers*.

Then,

```
visudu /etc/sudoers.d/100-wireguard
```

Edit *WGUSERS* as above.

visudo enforces the correct permissions which should be '0440'. If permissions are too loose, sudo will ignore the file.

Why the prefix number? Because sudo uses the **last** matching entry and we need to be sure the NOPASSWD wg-quick entry comes after any group wheel lines.

For example if there are 2 files in */etc/sudoers.d* - say wg-quick and wheel, where the wheel entry requires a password for members of group wheel.

Now if user listed in wg-quick is also a member of *wheel* group, since wg-quick is first and wheel is second (files are treated in lexical order) the *wheel* one will prevail and user will be prompted for a password when running *sudo /usr/bin/wg-quick*. Not what we want. To fix this use numbers to prefix the sudoers filenames. So in this example it would be:

```
/etc/sudoers.d/010-wheel
/etc/sudoers.d/100-wg-client
```

thereby ensuring that wg-client entries follow the wheel ones.

For convenience this is also noted in the sample file:

```
/etc/wg-client/sudoers.sample
```

```
chmod -440 /etc/sudoers.d/wg-client
```

CHAPTER

THREE

APPENDIX

3.1 Installation

Available on:

- Github
- Archlinux AUR

On Arch you can build using the PKGBUILD provided in packaging directory or from the AUR package.

To build manually, clone the repo and do:

```
rm -f dist/*
/usr/bin/python -m build --wheel --no-isolation
root_dest="/" ./scripts/do-install $root_dest
```

When running as non-root then set root_dest a user writable directory

3.2 Dependencies

Run Time :

- python (3.11 or later)
- netifaces
- hicolor-icon-theme
- psutil (aka python-psutil)
- PyQt6 / Qt6 (for gui)
- hicolor-icon-theme
- dateutil
- netifaces
- licap
- pynotify
- openssl (3.0 or later)

Building Package:

- git
- hatch (aka python-hatch)

- wheel (aka python-wheel)
- build (aka python-build)
- installer (aka python-installer)
- rsync

Optional for building docs: * sphinx * myst-parser * texlive-latexextra (archlinux packaguing of texlive tools)

3.3 Philosophy

We follow the *live at head commit* philosophy as recommended by Google's Abseil team[Page 4, 1]. This means we recommend using the latest commit on git master branch.

3.4 License

CHAPTER

FOUR

LICENSE

Linux Wireguard client software (command line and graphical user interface).