wg_tool

Release 6.3.0

Gene C

Jan 17, 2024

CONTENTS:

CHAPTER

ONE

WG-TOOL

1.1 Overview

Manage wireguard server and user configs. Ensures server and user configs remain consistent.

Available on

- Github
- Archlinux AUR

On Arch install using the PKGBUILD provided in packaging directory or from the AUR.

1.2 Key features

- simplifies wireguard administration. (server and users)
- guarantees server and user configs remain synchronized.
- handles key creation when needed
- users can have multiple profiles (bob:laptop bob:phone etc)
- users and/or profiles can be marked active/inactive
- takes output of 'wg show' and shows connections by user/profile name. This solves a long standing annoyance in a simple way by showing names not public keys. Provides check that server is up to date and may need restart with new wg0.conf
- supports importing existing user/profiles

1.3 New

- New Feature: Multiple IP Address for user profiles.

  See new options *–prefixlen_4* and *–prefixlen_6*.

  I'd appreciate testing and feedback (see Issue #14)

  New user profiles now get ip(s) from each server network. CIDR address for each network will have prefixlen_4 or IPv4 and prefixlen_6 for IPv6 networks. prefixlen are settable with new options.

  Existing user profile (or -all) can have their IPs refreshed to pick up their new IPs from server config. If you already have multiple networks or simply added them to the server *Address* variable in *configs/server/server.conf* - and then refresh using:

```
wg-tool -mod -ips <user>:<profile>
```

or

```
wg-tool -mod -ips -all
```

- New option *-upd, –upd_endpoint* used with *-mod* to update existing user profiles when server IP/Port is changed.
- *-mod* now supports *-all* to apply to all users.
- wg-client companion package now available. A linux client tool and separate graphical program to launch wireguard client. Simplify using wg for all users.
- wg-peer-updn now saves additional copy of dns file as resolv.conf.wg Helpful for clients which sleep and on resume network restart overwrites resolv.conf This makes it simple to put back the vpn resolv.conf file by copying resolv.conf.wg to resolv.conf. Used by wg-client package. Postdown will still restore original resolv.conf.save as usual.
- Change python build from poetry to hatch
- Can now generate html and pdf docs using sphinx Pre-built wg_tool.pdf provided in repo See *Howto-Build* in the *Docs* directory

1.4 Interesting

The Wireguard server report shows users by user:profile names instead of by public key fingerprint.

Can get human readable server report based on output of *wg show*. Can do this either by running on the wg server (*-rrpt, *–run_show_rpt*) or from saved report file *(*-rpt*, *–show_rpt*).

This report shows users and profiles in nice human readable form.

It also indicates whether each user and profile are marked active (by showing (+) or (-) beside the name. If an inactive user is connected, it may be time ensure the server is running the latest wg0.config.

This feature solves a long standing problem with native wireguard reports which burden the administrator with mapping IPs or public keys to a user profile. The report does it for you and shows actual user and profile names.

Because of this feature, this tool eliminates any need for schemes, such as Vanity keys, attempting to map public keys to something more palatable.

It will also advise if the current server config being used is out of sync with current tool config and therefore needs updating and/or restarting

Sample output of *wg-tool -rrpt*

```
wg server:
   interface : wg0
        port : nnnnn
     pub_key : <x>

   susan (+) : phone (+)
    endpoint : xxx.xxx.xxx.xxx:yyyyy
     address : xxx.xxx.xxx.xxx/32
   handshake : 2 hours, 4 minutes, 15 seconds ago
    transfer : 102.62 MiB received, 320.29 MiB sent
```

1.5 More background

The tool manages wireguard server and user configs.

It also guarantees that server and user configs are kept properly synchronized. It handles key creation whenever needed, such as when adding user/profiles or when doing key rollovers.

A wireguard server and user configs share several common variables, such as public keys, hostname and listening ports, and therefore it's crucial they are consistent.

wg-tool uses a single source of data which is used to populate the actual configs wireguard needs; this approach guarantees they are always consistent with one another. It also simplifies managment significantly. Common tasks are handled by the tool in a convenient way. For example, It is very straightforward to add users or user profiles, roll keys or make users or profiles active or inactive.

In a nutshell to setup and use wireguard vpn one needs a server and each client gets a configuration, either in the form of a text based *.conf* file or a QR code. QR codes work nicely for wireguard phone app, for example, where the app uses on board camera to read the the QR code. For computer clients, the conf file is the simplest. The server and client keys share common information which must be kept synchronized. This includes shared public keys, pre-shared keys for added security along with network information (IPs, Ports and DNS).

wg-tool uses a file based configuration database kept under the *config* directory. This provides all the inputs the tool needs to generate the server and client configs. The latter are saved into the *wg-config/server* and *wg-config/users* directories for the server and clients respectively.

For convenience, previous configs are saved with *.prev* extension making it easy to compare with a prior version. It can be useful after making changes to diff the two configs.

The wg server config, *wg-config/server/wg0.conf* should be installed, as usual, in /etc/wireguard.

Each user can have 1 or more profiles. For example bob may have *bob:phone* and *bob:laptop*. The configs to share with each profile is saved into, in this example, *wg-config/users/bob* as bob-phone.conf, bob-phone-qr.png, bob-laptop.conf and bob-laptop-qr.png. These are provided to the user - bob in this case.

For those computer clients running Linux, there are 2 kinds of configs available.

- standard config

  where the DNS infomation in config is used by wg-quick. wg-quick, in turn, relies on resolvconf.

- linux config

  this is my preferred approach. Activated by the *–dns_linux* option. When using this, wg-quick uses the provided *wg-peer-updn* script via PostUp/PostDown.

  This scipt saves the current dns resolv.conf file when VPN is brought up using *wg-quick up*, installs the VPN dns into /etc/resolv.conf and restores the prior resolv.conf when VPN is deactivated (wg-quick down).

1.6 Directory and File Structure

There are 2 kinds of config files. We use the following convention:

- **wg-configs** : configs used by wireguard itself

  These are the outputs of *wg-tool*.

- **configs** : configs used by wg-tool

  These are the inputs for *wg-tool*

For example, the wireguard server config file, wg0.conf, will be located in

```
wg-configs/server/wg0.conf
```

And the user QR codes and *.conf* files will be under

```
wg-configs/users/
```

Laying out this directory structure in a bit more detail.

wg-tool configs

```
configs/
       server/
           server.conf
       users/
           user-1/
               user-1.conf
           user-2/
               user-2.conf
           ...
```

wireguard configs will be placed

```
wg-configs/
          server/
                wg0.conf
          users/
                user-1/
                    user-1-profile-1.conf
                    user-1-profile-1.png

                    user-1-profile-2.conf
                    user-1-profile-2.png
                    ...
                user-2/
                    user-2-profile-1.conf
                    user-2-profile-1.png

                    user-2-profile-2.conf
                    user-2-profile-2.png
```

Each of the files is actually a symlink to the real file which is kept under a *db* directory at the same level as the symlinks.

This allows us to keep history of every config as far back as we choose. There are options to choose the amount of history to keep for configs and separately for wg-configs. The default, in addition to current values, is to keep 5 additional configs and 3 wg-configs.

Whenever a config file is changed the previous version is made available as a symlink named *xxx.prev*. This allows for straightforward comparisons and makes it easy to revert if that were ever needed; though it is pretty unlikely to ever be the case.

Each user can have multiple profiles - each profile provides separate access to the vpn. As an example, user *jane* may have a *phone* profile and a *laptop* profile. Each profile will provide the wireguard .conf file along with an image file of its QR code. These 2 files provide the standard wireguard configs for users.

Aside from the QR image files, all text files are in standard TOML format.

CHAPTER

TWO

GETTING STARTED

2.1 Using wg-tool for first time

There are 2 ways to get started; either create a new suite of users/profiles or import existing wireguard user.conf files. You can add users or new profiles for existing users at any time. This is very easy and explained below using the *–add_user* option. You can also import a user at any time, though it's primarily useful when first setting up wg-tool.

If you already have wireguard running then importing is the simplest and best way to proceed. If you're starting from scratch then wg-tool will create new users and profiles for you.

Either way it's pretty straightforward.

2.1.1 Step 1 - Create Server Config

In either case the first step is to create a valid server config file. The best way to do that is to run:

```
      wg-tool --init
or
      wg-tool --work_dir=xxx --init
```

By default, when initializing, work_dir will be */etc/wireguard/wg-tool* if it exists and with appropriate access permission (i.e. root), otherwise the current directory *./*.

This creates a template in: *configs/server/server.conf*.

This file must be edited and changed to reflect your own network settings etc. These are all wireguard standard fields.

The key fields to edit are:

- Address

  This is the internal wg cidr mask on the server IP addresses (IPv4 and IPv6). N.B. If you prefer user:profile get IPv6 then put it first in the list.

- Hostname and ListenPort

  wg server hostname as seen from internet and port chosen

- Hostname_Int ListenPort_Int

  wg server hostname and port as seen on internal network. Useful for testing wg while inside the network. Client configs created with the *-int* option of **wg-tool** will use this internal server:port.

- PrivateKey, PublicKey

  If you have exsiting wg server, change these to your current keys. If not they are freshly generated by –init. and can be safely used.

- PostUp PostDown

  If you want to use the nftables provided by wg-tool - just copy postup.nft from the scripts directory. Change the 3 network variables at top for your setup.

- DNS

  List of dns servers to be used by wg - typical VPN setup uses internal network DNS

postup.nft

The nftables sample script, scripts/postup.nft, should be copied to /etc/wireguard/scripts.

Remember to edit the network variables at the top of the *postup.nft* script to match your network. One common case is to provide users with access to internet as well as to the internal network. The system border firewall must forward vpn traffic to the wireguard server which running on inside protected by the firewall.

The *postup.nft* script provides access to the internet and lan provided the wireguard server host has that access. If the wg server is in the DMZ then it probably only has access to DMZ net and internet.

Before deploying the *postup.nft* script, edit the 3 variables at the top for your own server setup:

- vpn_net

  this cidr block must match whats in the server config

- lan_ip lan_iface

  IP and interface of wireguard server

Remember to allow forwarding on the wireguard server, to ensure VPN traffic is permitted to go to the LAN:

```
sysctl -w net.ipv4.ip_forward=1
```

to keep this on reboot add to */etc/sysctl.d/sysctl.conf* (or other filename):

```
net.ipv4.ip_forward = 1
```

The list of active users is managed in the *server.conf* file. This is generated and updated by wg-tool. The tool provides options to add and remove users from the active list. If a user is markewd inactive, none of their profiles will be in server wg0.conf. If a user is active then only their active profiles will be provided to wg0.conf

Each user config has its own list active profiles. It too is managed by the tool.

N.B. the active users and active profiles lists, only affect whether they are included in the server wg0.conf file. No user or profile is removed when a user and/or profile is marked inactive.

2.1.2 Step 2 - import and/or add users and profiles

Now that the server config is ready, we can add users and their profiles.

Each user can have 1 or more profiles. Each user's data, including all their profile info, in kept in a single config file. It also tracks the list of active profiles.

If a profile is active, it will be put in wireguards wg0.conf server config, otherwise it won't.

Wireguard QR codes and .conf files are always created for every user/profile regardless of whether it is active or not.

Since each user has their own namespace, profile names can be same for different users.

2.2 Adding new users and profiles

Users and profiles can be created at any time. They can be created in bulk or one user at a time. For example this command:

```
wg-tool --add_user bob:phone,desk,ipad jane:phone,laptop
```

creates 2 users. *bob* gets 3 profiles : phone, desk and ipad while *jane* gets 2 profiles: phone and laptop.

If you don't provide a profile name, the default profile name is *main*.

At this point you should now have server config supporting these 5 user profiles and the corresponding wireguard QR codes and .conf files under wg-configs/users

You can get list of all users and their profiles

```
wg-tool --list_users
```

The (+) or (-) after a user or profile name indicates active or inactive.

Importing existing users and profiles

The tool can import 1 user:profile at a time. This is done using:

```
wg-tool --import_user <user.conf> user_name:profile_name
```

where <user.conf> is the standard wireguard conf file (the text version of the QR code). And the user_name and profile_name are what you want them to be known as now.

What worked for me was to copy all those existing wireguard user.conf files into ./old/ and then make a little shell script like the sample scripts/import_users. Script just imports each profile 1 at a time.

Then run the shell script. End result should be working wg0.conf functionally identical to what you currently have. In addition a new set of user-profile.conf and associated qr codes. All found in *wg-configs/*

As above you may want to see a list of users/profiles:

```
wg-tool --list_users
```

And compare a user profile conf or 2 with existing ones - QR codes will be different, but contain the same information. You can check this for bob's laptop QR by doing this:

```
zbarimg wg-configs/users/bob/bob-laptop-qr.png
```

which is available in the zbar package. It should match the corresponding user.conf file in *wg-configs/users/bob/bob-laptop.conf*

2.3 Managing Users/Profiles

I recommend avoiding manually editing any config files, but if you do for some reason, then run *wg-tool* with no arguments. It will detect the changes and update *wg-configs*.

Pretty much everything you need to do should be available using wg-tool:

```
wg-tool --help
```

gives list of options.

2.3.1 Options

Many options take user/profiles as additional input. users/profiles are to be given on command line

```
   user
or
   user:prof
or
   user_1:prof_1,prof_2 user2 user_3:laptop,tablet
```

Summary of available options:

Positional arguments:

- users : user_1[:prof1,prof2,...] user_2[:prof_1,prof_2]

Options:

- (*-h, –help*)

  Show this help message and exit

- (*-i, –init*)

  Initialize and creat server config template. Please edit to match your server settings.

- (*wkd, –work_dir <dirname>*)

  Set working directory. This is is the directory holding all configs.

  By default:

  - when used with *–init*, work_dir will be */etc/wireguard/wg-tool* if the directory exists and with appropriate access permission (i.e. root), otherwise the current directory *./*.
  - if not initializing, then, with access permission, */etc/wireguard/wg-tool/* will be the work_dir if there is a *config* dir in it, otherwise it is set to current dir *./*.

- (*-add, –add_users*)

  Add user(s) and/or user profiles user:prof1,prof2,...

- (*-mod, –mod_users*)

  Modify existing user:profile(s). Use with *-dnsrch*, *-dnslin*, and *upd* Can apply to all users/profiles via the *-all* option.

- (*-pfxlen_4, –prefixlen_4*)

  User profiles now get IP Addresses(es) from each server network. Each address is a block with cidr prefixlen_4. Defaults to 32 which means 1 IP address. e.g. if set to 30 then would get a block of 4 x.x.x.x/30

- (*-pfxlen_5, –prefixlen_5*)

  Similar to –prefixlen_4 but for ipv6. Default is 128

- (*upd, –upd_endpoint*)

  Use with *-mod* Ensure user/profile is using current server endpoint. Add *-int* if want to use internal hostname/port.

  For example if the server IP changes, then you can update existing user/profiles with

  wg-tool -mod -upd -all

- (*-dnsrch, –dns_search*)

  Use with *-mod*

  Adds the list DNS_SEARCH from server config to client DNS search list. DNS_SEARCH in server.conf should contain a list of dns domains for dns search and Use together with *-add* for new user:profile or with *-mod* with existing profile.

- (*-dnslin, –dns_linux*)

  Use with *-mod*

  For a Linux client, provide support for managing the dns resolv.conf file. What this does is save existing one, install the wireguard dns version and then restore original on exit. Use together with *-add* for new user:profile or with *-mod* with existing profile.

  To bring up wireguard as a linux client one uses

  ```
  wg-quick up <user-prof.conf>
  wg-quick down <user-prof.conf>
  ```

  This will then use the wireguard DNS while running and restore previous dns on exit.

  To add dns search and use dns_linux on existing user profile. First update the server config by editing *configs/server/server.conf* and add list of seach domains

  ```
  DNS_SEARCH = ['sales.example.com', 'example.com']
  ```

then

```
wg-tool -mod -dnsrch -dns_linux bob:laptop
```

By default wg-quick uses resolvconf to manage dns resolv.conf. If you prefer, or dont use resolvconf then use this option. But only with Linux - it will not work for other clients (Android, iOS, etc)

With this option the usual DNS rows in in the conf file are replaced with PostUp and PostDown. PostUp saves existing resolv.conf, and installs the one needed by wireguard. PostDown restores the original saved resolv.conf.

To use this the script *wg-peer-updn*, available in the *scripts* directory must be in /etc/wireguard/scripts for the client.

The installer for the wg_tool package installs the script - but clients without this package should be provided both the user-profile.conf as well as the supporting script *wg-peer-updn*.

- (*-int, –int_serv*)

  With –add_users uses internal wireguard server

- (*-uuk, –upd_user_keys*)

  Generate new set of keys for existing user(s). This is public and private key pair along with new pre-shared key.

- (*-usk, –upd_serv_keys*)

  Generate new pair of server keys. NB This affects all users as they all use the server public key.

- (*-all, –all_users*)

  Some opts (e.g. upd_user_keys) may apply to all users/profiles when this is turned on.

- (*-act, –active*)

  Mark one or more users or user[:profile, profile...] active

- (*-inact, –inactive*)

  Mark one or more users or user[:profile, profile...] inactive

- (*-imp, –import_user <file>*)

  Import a standard wg user conf file into the spcified user_name:profile_name This is for one single user:profile

- (*-keep, –keep_hist <num>*)

  How much config history to keep (default 5)

- (*-keep_wg, –keep_hist_wg <num>*)

  How much wg-config history to keep (default 3)

- (*-sop, –save_opts*)

  Together with –keep_hist and/or –keep_hist_wg to save these values as new defaults.

- (*-rrpt, –run_show_rpt*)

  Run "wg show" and generate report of users, profiles. Also checks for consistency with current settings.

- (*-rpt, –show_rpt <file>*)

  Same as *-rrpt* only reads file containing the output of *wg show* If file is name *stdin*, then it reads from stdin.

- (*-l, –list_users*)

  Summary of users/profiles - sorted by user.

- (*-det, –details*)

  Adds more detail to *-l* and *-rrpt*. For *-l* report will also include details about each profile. For *-rrpt* report will show all user:profiles known to running server, not just those for which it has a recent connection.

- (*-v, –verb*)

  Adds more verbose output.

2.3.2 Note on MTU

I came across one hotel wifi, that while the vpn worked fine to provide internet access, I found that for my laptop to be able to also 'ssh internal-host' it would hang:

```
ssh -v <host>
```

hangs right after this is logged:

```
expecting SSH2_MSG_KEX_ECDH_REPLY
```

The *fix* was to set the MTU from 1500 down to 1400 on my laptop while at that hotel. The internet access continued to work fine, but this fixed whatever was a problem for ssh; so now 'ssh internal-host' worked as usual.

I have only had to change MTU setting at one location, but I mention it here in case anyone else comes across this.

2.4 Key Rollover

wg-tool makes key rollover particularly simple - at least as far as updating keys and regenerating user and/or server configs with the new keys.

Distribution of the updated config/QR code to each user is not addressed by the tool. Continue to use existing methods - encyrpted email, in person display of QR code etc. ...

Its equally simple to update keys on a per user basis as well - just specify them on command line.

To roll the server keys run:

```
wg-tool --upd_serv_keys
```

This will also update all user profiles with the server's new public key.

To roll all user keys run:

```
wg-tool --upd_user_keys
```

or as usual you can specify which profiles to generate the new keys for.

```
wg-tool --upd_user_keys  [user:prof1,prof2 user2 ..]
```

As usual, a change to any user profiles will generate new server wg0.conf file reflecting whaterver change was made.

CHAPTER

THREE

APPENDIX

3.1 Notes

- Config changes are tracked by modification times.

  For existing user/profiles without a saved value of *mod_time*, the last change date-time of the config file is used and saved. These mod times are displayed when using *-l* and *-l -det* options.

3.1.1 2022-12

- Stronger file access permissions to protect private data in configs.
- Changes to work_dir.

  Backward compatible with previous version. Now prefers to use */etc/wireguard/wg-tool* if possible, otherwise falls back to current directory.

3.1.2 2022-11

See *Options* or for more detail.

- (*-dnsrch, –dns_search*)

  Adds the list DNS_SEARCH from server config to client DNS search list. DNS_SEARCH in server.conf should contain a list of dns domains for dns search. Use together with *-add* for new user:profile or with *-mod* with existing profile.

- (*-dnslin, –dns_linux*)

  For a Linux client, provide support for managing the dns resolv.conf file. What this does is save existing one, install the wireguard dns version and then restore original on exit. Use together with *-add* for new user:profile or with *-mod* with existing profile.

3.2 Install

While it is simplest to install from a package manager, manual installs are done as folllow:

First clone the repo :

```
git clone https://github.com/gene-git/wg_tool
```

Then install to local directory. When running as non-root then set root_dest to a user writable directory.

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

3.2.1 Dependencies

- Run Time :
  - python (3.9 or later)
  - wireguard-tools
  - nftables (for wireguard server postup.nft)
  - tomli_w (aka python-tomli_w)
  - netaddr (aka python-netaddr)
  - python-qrcode
  - If python < 3.11 : tomli (aka python-tomli)
- Building Package:
  - git
  - hatch (aka python-hatch)
  - wheel (aka python-wheel)
  - build (aka python-build)
  - installer (aka python-installer)
  - rsync

3.2.2 Philosophy

We follow the *live at head commit* philosophy. This means we recommend using the latest commit on git master branch.

This approach is also taken by Google[1][2].

[1] https://github.com/google/googletest
[2] https://abseil.io/about/philosophy#upgrade-support

3.3 License

Created by Gene C. and licensed under the terms of the MIT license.

- SPDX-License-Identifier: MIT
- SPDX-FileCopyrightText: © 2022-present Gene C <arch@sapience.com>

CHAPTER

FOUR

NETWORKING NOTES

This note is an overview of wireguard networking setup. By default we assume that the wireguard server has access to internet and any desired local services. With this in mind, the default *postup.nft* script NATs the wireguard network so it's traffic to the LAN appears to come from the wireguard server itself. We also assume that the local DNS servers provided in wireguard configs, are available to the wireguard server - typicallu as they are on the same LAN.

Edit *postup.nft* and set the variables for your network

> define vpn_net = 10.99.99.0/24 # must match server config define lan_ip = 10.0.0.1 # real lan ip of wg server define lan_iface = eno1 # lan interface define wg = wg0

4.1 Assumed Network Topology for default postup.nft

```
(        |              |                           |          |                |
(        |              | (vpn.ip:39001)            |          |                |
(        |    <--->     | pub-ip           lan-ip   | <-NAT->  | lan-ip:39001   |
(        |              |                           |          | en01           |
(        |              |                           |          |                |
( Client |   Internet   |       Firewall            |   LAN    |   WG Server    |

+ ------------------------------------------------------------------------------+
|     <-->              VPN Tunnel  (10.99.99.x)                 <---->       |
+------------------------------------------------------------------------------+
```

If your network differs, then adjust the nftables postup rules as needed.

We also assume that there are no other nftables rules running on the wireguard server itself. If there are then you should change the postdown rule to not flush all rules as happens by default and instead add the rules and remove them in postdown.

CHAPTER

FIVE

CONTRIB

5.1 Ubuntu Notes

Provided by Jack Duan (@jduan00 via github #13)

I would like to share the steps I got it to work on Ubuntu 22.04. Feel free to include in your documentation to help others.

5.1.1 Install Prep

Assume these are already installed.

```
$ sudo apt install wireguard qrencode
```

Install necessary python3 packages for wg_tool:

```
$ sudo apt install python3 python-is-python3 python3-poetry python3-build \
python3-installer python3-qrcode python3-tomli-w python3-tomli python3-netaddr
```

5.1.2 Build and install

```
$ git clone https://github.com/gene-git/wg_tool
$ cd wg_tool

$ sudo pip install hatchling

$ /usr/bin/python -m build --wheel --no-isolation

$ sudo ./scripts/do-install /
```

5.1.3 Getting wg and wg-tool to work

```
$ sudo -i
# cd /etc/wireguard

# wg-tool --init

# vi config/server/server.conf
```

Add the first set of users with wg-tool

```
# wg-tool --add_user --dns_search Mary:mac,iphone,ipad

# ln -s wg-configs/server/wg0.conf .

# systemctl enable wg-quick@wg0
# systemctl start wg-quick@wg0
# systemctl status wg-quick@wg0

# wg show
```

Add more users

```
# cd /etc/wireguard
# wg-tool --add_user --dns_search Joe:mac,iphone,ipad

# systemctl reload wg-quick@wg0

# wg show

# wg-tool --list_users

# wg-tool -rrpt
```

5.1.4 iptables

Also, I don't use PostUp scripts since persistent iptable rules are used.

```
# apt install iptables-persistent
# systemctl status iptables
# systemctl enable iptables
# systemctl restart iptables
```

Change /etc/iptables/rules.v4 Note

- 192.168.100.0/24 is used for wg subnet.
- On Ubuntu ens3 is the default network device name

```
# == For Wireguard VPN ==
*nat
:PREROUTING ACCEPT [0:0]
:INPUT ACCEPT [0:0]
```

(continues on next page)

(continued from previous page)

```
:OUTPUT ACCEPT [0:0]
:POSTROUTING ACCEPT [0:0]
# -- NAT for vpn clients from wireguard
-A POSTROUTING -o ens3 -s 192.168.100.0/24 -j MASQUERADE
COMMIT
# --------------------
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -i lo   -j ACCEPT
-A INPUT -i wg+  -j ACCEPT
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -m state --state INVALID             -j DROP
# -- Wireguard ports
-A INPUT -i ens3 -p  udp -m multiport --dports 51820 -m state --state NEW -m limit --
↪limit 10/sec --limit-burst 20 -j ACCEPT
-A INPUT -i ens3 -p icmp -m icmp --icmp-type 8 -m limit --limit 5/sec -j ACCEPT
# -- last as the default rule for input
-A INPUT -j DROP
# -- forward --
-A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -m state --state INVALID             -j DROP
-A FORWARD -i wg+  -s 192.168.100.0/24          -j ACCEPT
-A FORWARD -j REJECT --reject-with icmp-port-unreachable
# -- out --
-A OUTPUT -m state --state INVALID -j DROP
COMMIT
# == EOF ==
```

Run commands:

```
# systemctl restart iptables

# iptables -nvL
# iptables -t nat -nvL
```

CHAPTER

SIX

CHANGELOG

[6.3.0] —— 2024-01-17

- Simplify ip address manipulations a few lines to original now bug is fixed
- update Docs/Changelog.rst Docs/wg_tool.pdf

[6.2.0] —— 2024-01-17

- Bugfix : generating IPs was skipping too many available blocks
- update Docs/Changelog.rst Docs/wg_tool.pdf

[6.1.0] —— 2024-01-17

- Fix: update AllowedIPs with –ips_refresh
- update Docs/Changelog.rst Docs/wg_tool.pdf

[6.0.1] —— 2024-01-17

- bump patch version for readme change
- readme tweak
- update Docs/Changelog.rst Docs/wg_tool.pdf

[6.0.0] —— 2024-01-17

- Add support for multiple IP addresses in user profiles. Addresses will now be taken from whichever networks are in server config. cidr address for each network will have prefixlen_4 for IPv4 and prefixlen_6 for IPv6 networks. prefixlen are settable with new options. Existing user:profile (or -all) can have their IPs refreshed to pick up their new IPs from server config. If you already have multiple networks or simply added them to Address variable in configs/server/server.conf - then can refresh using: wg-tool -mod -ips user_name:profile_name or wg-tool -mod -ips -all
- update Docs/Changelog.rst Docs/wg_tool.pdf

[5.7.2] —— 2024-01-13

- Add ubuntu notes provided by Jack Duan (@jduan00 via github #13)
- update Docs/Changelog.rst Docs/wg_tool.pdf

[5.7.1] —— 2024-01-12

- update Docs/Changelog.rst Docs/wg_tool.pdf
- lint picking
- update Docs/Changelog.rst Docs/wg_tool.pdf

[5.7.0] —— 2024-01-11

- Add -upd option to update user/profile endpoint when server config changes. (closes GH issue #11) -mod option can now be used with -all
- update Docs/Changelog.rst Docs/wg_tool.pdf

[5.6.3] —— 2024-01-07

- rst fix in readme

[5.6.2] —— 2024-01-07

- fix readme typo

[5.6.1] —— 2024-01-07

- small readme update
- update Docs/Changelog.rst Docs/wg_tool.pdf

[5.6.0] —— 2023-12-07

- wg-peer-updn now saves additional copy of wg resolv.conf in resolv.conf.wg. Can be used by client when resume causes network restart to overwrites the wg resolv.conf. Used by wg-client package to "fix" dns after sleep/resume.
- update Docs/Changelog.rst Docs/wg_tool.pdf

[5.5.1] —— 2023-11-23

- Improve description
- update Docs/Changelog.rst Docs/wg_tool.pdf

[5.5.0] —— 2023-11-23

- Change python build from poetry to hatch. It is cleaner and simpler. Switch copyright lines to SPDX format
- update Docs/Changelog.rst Docs/wg_tool.pdf

[5.4.1] —— 2023-11-12

- Minor readme rst format change. Add wg_tool.pdf
- update Docs/Changelog.rst

[5.3.4] —— 2023-09-30

- Add sample output of server report to README

[5.3.3] —— 2023-09-30

- Improve README
- update Docs/Changelog.rst

[5.3.2] —— 2023-09-27

- update Docs/Changelog.rst
- Fix links in readme. Remove doc build dependency on myst-parser since no more mardown
- update Docs/Changelog.rst

[5.3.1] —— 2023-09-26

- Release as 5.3.1
- fix rst list items in Changelog
- update Docs/Changelog.rst

[5.3.0] —— 2023-09-26

- Reorg docs - add Docs/dir with sphinx support update PKGBUILD for optional doc builds Migrate to rst from markdown
- update CHANGELOG.md

[5.1.1] —— 2023-09-25

- README - replace markdown url links with rst link notation
- update CHANGELOG.md

[5.1.0] —— 2023-08-02

- Improve code finding available client IPs to properly support IPv6. Client IPs are chosen from the server Address list in natural order. If you prefer clients get IPv6 addresses, those should be listed first. Similarly, if IPv4 is preferred, then put that first. Tidy to keep pylint clean
- update CHANGELOG.md

[4.7.0] —— 2023-07-28

- Fix import open_file buglet
- update CHANGELOG.md

[4.6.0] —— 2023-05-18

- install: switch from pip to python installer package. This adds optimized bytecode
- update CHANGELOG.md

[4.5.3] —— 2023-05-18

- PKGBUILD: build wheel back to using python -m build instead of poetry
- update CHANGELOG.md

[4.5.2] —— 2023-05-17

- Simplify Arch PKGBUILD and more closely follow arch guidelines
- update CHANGELOG.md

[4.5.1] —— 2023-05-08

- Add comment to README about linux using wg and ssh and MTU
- typo
- update CHANGELOG.md

[4.5.0] —— 2023-05-02

- Add comment on philosophy of living at the head commit. Change README from markdown to restructured text

[4.4.0] —— 2023-04-15

- update CHANGELOG.md
- Only show user public key for "-rpt" when also using "-det". Since we show user and profile name, the user key is not really needed
- update CHANGELOG.md

[4.3.6] —— 2023-04-11

- postup.nft script add extra line: ct status dnat accept - I saw a martial packat at firewall from vpn which was unexpected minor readme edit update project version
- update CHANGELOG.md

[4.3.5] —— 2023-01-06

- Add SPDX licensing lines
- update CHANGELOG.md

[4.3.4] —— 2022-12-29

- Add reminder in README to allow ip forwarding on wireguard server
- update CHANGELOG.md

[4.3.3] —— 2022-12-28

- Add brief networking note
- update CHANGELOG.md

[4.3.2] —— 2022-12-26

- Change default python interpreter location to /usr/bin/python3 (remove env). This is also recommended by e.g. debian packaging guidelines (https://www.debian.org/doc/packaging-manuals/python-policy). While many distros (Arch, Fedora etc.) recommend /usr/bin/python - we keep python3 which will work on those and on debian until debian provides python (and not just python3).
- update CHANGELOG.md

[4.3.1] —— 2022-12-25

- Move archlinux dir to packaging. Add packaging/requirements.txt Update build dependencies in PKGBUILD Tweak README
- tweak README
- update CHANGELOG.md

[4.3.0] —— 2022-12-20

- Change python to python3 (as per GH issue #5 on ubuntu/debian. Remove pip option from installer (–root-user-action=ignore)
- indent fix
- To help with older pre 3.9 python versions, provide files without match(). They are in lib38. Copy to lib38/*.py lib/
- update CHANGELOG.md

[4.2.0] —— 2022-12-14

- update CHANGELOG.md
- Installer now uses pip install PKGBUILD now uses poetry to build wheel
- update CHANGELOG.md

[4.1.0] —— 2022-12-08

- Server show_rpt was not treating inactive users/profiles properly - fixed
- update CHANGELOG.md

[4.0.0] —— 2022-12-04

- Stronger file access permissions to protect private data in configs. Changes to work_dir: Backward compatible with previous version. Now prefers to use */etc/wireguard/wg-tool* if possible, otherwise falls back to current directory. Thanks to Yann Cardon
- Improve comments in postup.nft including reference to alternate postup from Yann Cardon
- Merge: f74aa16bc2 26e957cd19 Merge pull request #3 from ycardon/master Create postup-alternate.nft
- Create postup-alternate.nft provides an other example of postup script with useful comments
- update CHANGELOG.md

[3.7.0] —— 2022-12-03

- bug: –list if username(s) given without profile. Now we list all profiles
- update CHANGELOG.md
- Typo in README fixed by @ycardon
- Merge: 8c05f936df 6dcc5b6459 Merge pull request #2 from ycardon/master small typo in the readme
- small typo –add-users > –add_users
- update CHANGELOG.md

[3.6.0] —— 2022-11-30

- bug fix for –init Thanks to @ycardon - this fixes issue #1 : https://github.com/gene-git/wg_tool/issues/1
- update CHANGELOG.md

[3.5.0] —— 2022-11-29

- turn off test mode
- update CHANGELOG.md

[3.4.0] —— 2022-11-29

- Improve wg-peer-updn - Rename existing resolv.conf when saving - Add timestamp to wireguard resolv.conf
- update CHANGELOG.md

[3.3.1] —— 2022-11-29

- Small add to README
- update CHANGELOG.md

[3.3.0] —— 2022-11-29

- Improve README
- update CHANGELOG.md

[3.2.0] —— 2022-11-28

- typo
- update CHANGELOG.md

[3.1.0] —— 2022-11-28

- fix typo creating new user profile with -dnssrc/-dnslin
- tweak readme
- update CHANGELOG.md

[3.0.0] —— 2022-11-28

- Adds 3 new options: - –mod_users : modify existing user profile (with –dns_search and –dns_linux) - –dns_search : adds support for dns search domain list - –dns_linux : adds support for managing resolv.conf instead of relying on qg-quick/resolconf
- update CHANGELOG.md

[2.1.0] —— 2022-11-24

- – improve error msg
  - Check conf before using it - added when auto updating older configs using mtime of config
  - minor tweak to bash variable check in install script
- update CHANGELOG.md

[2.0.0] —— 2022-11-11

- list users report now sorts by user name Add support for tracking config modification date-time. mod_time displayed in list user report
- update CHANGELOG.md

[1.7.5] —— 2022-11-08

- Improve handling of booelan False vs None in pre-file-write dictionary cleaner
- update CHANGELOG.md

[1.7.4] —— 2022-11-07

- tweak readme
- update CHANGELOG.md

[1.7.3] —— 2022-11-04

- add poetry back to PKGBUILD makedepends
- update CHANGELOG.md

[1.7.2] —— 2022-11-04

- change installer to use bash array for app list (even tho we onlly have 1 here) tweak readme
- update CHANGELOG.md

[1.7.1] —— 2022-10-31

- Change build from poetry/pip to python -m build/installer
- update CHANGELOG.md

[1.7.0] —— 2022-10-31

- Add support for python 3.11 tomllib
- update CHANGELOG.md

[1.6.1] —— 2022-10-30

- update readme
- update CHANGELOG.md

[1.6.0] —— 2022-10-30

- -rpt now lists missing users/profiles from running server
- update CHANGELOG.md

[1.5.0] —— 2022-10-30

- Add –details Modifes -l, -rpt and -rrpt to provide detailed information in addition to the summary.
- update CHANGELOG.md

[1.4.0] —— 2022-10-29

- report: handle cases where running server has old user key and other edge cases
- update CHANGELOG.md

[1.3.2] —— 2022-10-29

- add –run_show_rpt. Similar to –show_rpt, but runs wg-tool
- update CHANGELOG.md

[1.3.1] —— 2022-10-29

- bug fix: -inact user:prof made user inactive not just prof
- update CHANGELOG.md

[1.3.0] —— 2022-10-29

- Add new option –work_dir Refactor and tidy code up some
- upd changelog
- tweak readme
- tweak readme and sync PKGBUILD
- upd changelog

[1.2.3] —— 2022-10-27

- Add mising packages to PKGBUILD depends (thank you @figue on aur)
- upd changelog

[1.2.2] —— 2022-10-27

- duh - turn off debugger .. sorry
- markdown newline fix
- word smith readme
- update changelog

[1.2.1] —— 2022-10-26

- update project vers
- actually add the code to make wg_show report :)

[1.2.0] —— 2022-10-26

- Adds support to parse output of wg show and provide user/profile names
- Add new/coming soon section to readme
- readme - aur package now avail
- update changelog

[1.1.1] —— 2022-10-26

- proj vers update

- installer: share archlinux into /usr/share/wg_tool
- Ready to share

CHAPTER

SEVEN

MIT LICENSE

CHAPTER

EIGHT

HOW TO HELP WITH THIS PROJECT

Thank you for your interest in improving this project. This project is open-source under the MIT license.

8.1 Important resources

- Git Repo

8.2 Reporting Bugs or feature requests

Please report bugs on the issue tracker in the git repo. To make the report as useful as possible, please include

- operating system used
- version of python
- explanation of the problem or enhancement request.

8.3 Code Changes

If you make code changes, please update the documentation if it's appropriate.

CHAPTER

NINE

CONTRIBUTOR COVENANT CODE OF CONDUCT

9.1 Our Pledge

In the interest of fostering an open and welcoming environment, we as contributors and maintainers pledge to making participation in our project and our community a harassment-free experience for everyone, regardless of age, body size, disability, ethnicity, sex characteristics, gender identity and expression, level of experience, education, socio-economic status, nationality, personal appearance, race, religion, or sexual identity and orientation.

9.2 Our Standards

Examples of behavior that contributes to creating a positive environment include:

- Using welcoming and inclusive language
- Being respectful of differing viewpoints and experiences
- Gracefully accepting constructive criticism
- Focusing on what is best for the community
- Showing empathy towards other community members

Examples of unacceptable behavior by participants include:

- The use of sexualized language or imagery and unwelcome sexual attention or advances
- Trolling, insulting/derogatory comments, and personal or political attacks
- Public or private harassment
- Publishing others' private information, such as a physical or electronic address, without explicit permission
- Other conduct which could reasonably be considered inappropriate in a professional setting

9.3 Our Responsibilities

Maintainers are responsible for clarifying the standards of acceptable behavior and are expected to take appropriate and fair corrective action in response to any instances of unacceptable behavior.

Maintainers have the right and responsibility to remove, edit, or reject comments, commits, code, wiki edits, issues, and other contributions that are not aligned to this Code of Conduct, or to ban temporarily or permanently any contributor for other behaviors that they deem inappropriate, threatening, offensive, or harmful.

9.4 Scope

This Code of Conduct applies both within project spaces and in public spaces when an individual is representing the project or its community. Examples of representing a project or community include using an official project e-mail address, posting via an official social media account, or acting as an appointed representative at an online or offline event. Representation of a project may be further defined and clarified by project maintainers.

9.5 Enforcement

Instances of abusive, harassing, or otherwise unacceptable behavior may be reported by contacting the project team at <arch@sapience.com>. All complaints will be reviewed and investigated and will result in a response that is deemed necessary and appropriate to the circumstances. The Code of Conduct Committee is obligated to maintain confidentiality with regard to the reporter of an incident. Further details of specific enforcement policies may be posted separately.

9.6 Attribution

This Code of Conduct is adapted from the Contributor Covenant, version 1.4, available at https://www.contributor-covenant.org/version/1/4/code-of-conduct.html

9.7 Interpretation

The interpretation of this document is at the discretion of the project team.

CHAPTER

TEN

INDICES AND TABLES

- genindex
- modindex
- search