NAME

wg-quick - set up a WireGuard interface simply

SYNOPSIS

wg-quick [*up* | *down* | *save* | *strip*] [*CONFIG_FILE* | *INTERFACE*]

DESCRIPTION

This is an extremely simple script for easily bringing up a WireGuard interface, suitable for a few common use cases.

Use *up* to add and set up an interface, and use *down* to tear down and remove an interface. Running *up* adds a WireGuard interface, brings up the interface with the supplied IP addresses, sets up mtu and routes, and optionally runs pre/post up scripts. Running *down* optionally saves the current configuration, removes the WireGuard interface, and optionally runs pre/post down scripts. Running *save* saves the configuration of an existing interface without bringing the interface down. Use *strip* to output a configuration file with all **wg-quick**(8)-specific options removed, suitable for use with **wg**(8).

CONFIG_FILE is a configuration file, whose filename is the interface name followed by '.conf'. Otherwise, *INTERFACE* is an interface name, with configuration found at '/etc/wireguard/*INTERFACE*.conf', searched first, followed by distro-specific search paths.

Generally speaking, this utility is just a simple script that wraps invocations to **wg**(8) and **ip**(8) in order to set up a WireGuard interface. It is designed for users with simple needs, and users with more advanced needs are highly encouraged to use a more specific tool, a more complete network manager, or otherwise just use **wg**(8) and **ip**(8), as usual.

CONFIGURATION

The configuration file adds a few extra configuration values to the format understood by **wg**(8) in order to configure additional attributes of an interface. It handles the values that it understands, and then it passes the remaining ones directly to **wg**(8) for further processing.

It infers all routes from the list of peers' allowed IPs, and automatically adds them to the system routing table. If one of those routes is the default route (0.0.0.0/0 or ::/0), then it uses **ip-rule**(8) to handle overriding of the default gateway.

The configuration file will be passed directly to **wg**(8)'s 'setconf' sub-command, with the exception of the following additions to the *Interface* section, which are handled by this tool:

- Address -- a comma-separated list of IP (v4 or v6) addresses (optionally with CIDR masks) to be assigned to the interface. May be specified multiple times.

- DNS -- a comma-separated list of IP (v4 or v6) addresses to be set as the interface's DNS servers, or non-IP hostnames to be set as the interface's DNS search domains. May be specified multiple times. Upon bringing the interface up, this runs 'resolvconf -a tun.*INTERFACE* -m 0 -x' and upon bringing it down, this runs 'resolvconf -d tun.*INTERFACE*'. If these particular invocations of **resolvconf**(8) are undesirable, the PostUp and PostDown keys below may be used instead.

- MTU -- if not specified, the MTU is automatically determined from the endpoint addresses or the system default route, which is usually a sane choice. However, to manually specify an MTU to override this automatic discovery, this value may be specified explicitly.

- Table -- Controls the routing table to which routes are added. There are two special values: 'off' disables the creation of routes altogether, and 'auto' (the default) adds routes to the default table and enables special handling of default routes.

- PreUp, PostUp, PreDown, PostDown -- script snippets which will be executed by **bash**(1) before/after setting up/tearing down the interface, most commonly used to configure custom DNS options or firewall rules. The special string '%i' is expanded to *INTERFACE*. Each one may be specified multiple times, in which case the commands are executed in order.

- SaveConfig -- if set to 'true', the configuration is saved from the current state of the interface upon shutdown. Any changes made to the configuration file before the interface is removed will therefore be overwritten.

Recommended *INTERFACE* names include 'wg0' or 'wgvpn0' or even 'wgmgmtlan0'. However, the number at the end is in fact optional, and really any free-form string [a-zA-Z0-9_=+.-]{1,15} will work. So even interface names corresponding to geographic locations would suffice, such as 'cincinnati', 'nyc', or 'paris', if that's somehow desirable.

EXAMPLES

These examples draw on the same syntax found for **wg**(8), and a more complete description may be found there. Bold lines below are for options that extend **wg**(8).

The following might be used for connecting as a client to a VPN gateway for tunneling all traffic:

```
[Interface]
```

Address = 10.200.100.8/24
DNS = 10.200.100.1
PrivateKey = oK56DE9Ue9zK76rAc8pBl6opph+1v36lm7cXXsQKrQM=

[Peer]
PublicKey = GtL7fZc/bLnqZldpVofMCD6hDjrK28SsdLxevJ+qtKU=
PresharedKey = /UwcSPg38hW/D9Y3tcS1FOV0K1wuURMbS0sesJEP5ak=
AllowedIPs = 0.0.0.0/0
Endpoint = demo.wireguard.com:51820

The ‘Address‘ field is added here in order to set up the address for the interface. The ‘DNS‘ field indicates that a DNS server for the interface should be configured via **resolvconf**(8). The peer’s allowed IPs entry implies that this interface should be configured as the default gateway, which this script does.

Building on the last example, one might attempt the so-called ‘‘kill-switch’’, in order to prevent the flow of unencrypted packets through the non-WireGuard interfaces, by adding the following two lines ‘PostUp‘ and ‘PreDown‘ lines to the ‘[Interface]‘ section:

PostUp = iptables -I OUTPUT ! -o %i -m mark ! --mark $(wg show %i fwmark) -m addrtype ! --dst-type LOCAL -j REJECT
PreDown = iptables -D OUTPUT ! -o %i -m mark ! --mark $(wg show %i fwmark) -m addrtype ! --dst-type LOCAL -j REJECT

The ‘PostUp’ and ‘PreDown’ fields have been added to specify an **iptables**(8) command which, when used with interfaces that have a peer that specifies 0.0.0.0/0 as part of the ‘AllowedIPs’, works together with wg-quick’s fwmark usage in order to drop all packets that are either not coming out of the tunnel encrypted or not going through the tunnel itself. (Note that this continues to allow most DHCP traffic through, since most DHCP clients make use of PF_PACKET sockets, which bypass Netfilter.) When IPv6 is in use, additional similar lines could be added using **ip6tables**(8).

Or, perhaps it is desirable to store private keys in encrypted form, such as through use of **pass**(1):

PostUp = wg set %i private-key <(pass WireGuard/private-keys/%i)

For use on a server, the following is a more complicated example involving multiple peers:

[Interface]
Address = 10.192.122.1/24
Address = 10.10.0.1/16

```
SaveConfig = true
PrivateKey = yAnz5TF+lXXJte14tji3zlMNq+hd2rYUIgJBgB3fBmk=
ListenPort = 51820

[Peer]
PublicKey = xTIBA5rboUvnH4htodjb6e697QjLERt1NAB4mZqp8Dg=
AllowedIPs = 10.192.122.3/32, 10.192.124.1/24

[Peer]
PublicKey = TrMvSoP4jYQlY6RIzBgbssQqY3vxI2Pi+y71lOWWXX0=
AllowedIPs = 10.192.122.4/32, 192.168.0.0/16

[Peer]
PublicKey = gN65BkIKy1eCE9pP1wdc8ROUtkHLF2PfAqYdyYBz6EA=
AllowedIPs = 10.10.10.230/32
```

Notice the two ‘Address’ lines at the top, and that ‘SaveConfig’ is set to ‘true’, indicating that the configuration file should be saved on shutdown using the current status of the interface.

A combination of the ‘Table’, ‘PostUp’, and ‘PreDown’ fields may be used for policy routing as well. For example, the following may be used to send SSH traffic (TCP port 22) traffic through the tunnel:

```
[Interface]
Address = 10.192.122.1/24
PrivateKey = yAnz5TF+lXXJte14tji3zlMNq+hd2rYUIgJBgB3fBmk=
ListenPort = 51820
Table = 1234
PostUp = ip rule add ipproto tcp dport 22 table 1234
PreDown = ip rule delete ipproto tcp dport 22 table 1234

[Peer]
PublicKey = xTIBA5rboUvnH4htodjb6e697QjLERt1NAB4mZqp8Dg=
AllowedIPs = 0.0.0.0/0
```

These configuration files may be placed in any directory, putting the desired interface name in the filename:

```
# wg-quick up /path/to/wgnet0.conf
```

For convenience, if only an interface name is supplied, it automatically chooses a path in

‘/etc/wireguard/’:

 # wg-quick up wgnet0

This will load the configuration file ‘/etc/wireguard/wgnet0.conf’.

The *strip* command is useful for reloading configuration files without disrupting active sessions:

 # wg syncconf wgnet0 <(wg-quick strip wgnet0)

SEE ALSO

wg(8), **ip**(8), **ip-link**(8), **ip-address**(8), **ip-route**(8), **ip-rule**(8), **resolvconf**(8).

AUTHOR

wg-quick was written by Jason A. Donenfeld <Jason@zx2c4.com>. For updates and more information, a project page is available on the World Wide Web <https://www.wireguard.com/>.