# nord

***Release 0.2.0+0.g1699a28.dirty***

**Jun 04, 2018**

# Contents:

At its core is a high-level Python API for interacting both with the web service provided by NordVPN, and for connecting to VPN servers using OpenVPN.

Nord also contains components that expose this API as a command line tool, and (soon) as a web service and frontend.

CHAPTER 1

# Overview

Nord is a client for interacting with the NordVPN service.

At its core is a high-level Python API for interacting both with the web service provided by NordVPN, and for connecting to VPN servers using OpenVPN.

Nord also contains components that expose this API as a command line tool, and (soon) as a web service and frontend.

CHAPTER 2

# Licence

Nord is licensed under the terms of the GNU GPLv3. See the LICENSE file for details.

CHAPTER 3

# Installation

```
pip install nord
```

CHAPTER 4

# Usage

Run `nord --help` for the full usage instructions.

Connect to a specific NordVPN server:

```
nord connect -u my_user -p my_password us893
```

Connect to any NordVPN server in a given country:

```
nord connect -u my_user -p my_password US
```

You can also supply your password from a file using the `-f` flag. The special value `-` means "read from stdin". This is particularly useful when your password is stored in a utility such as pass:

```
pass nordvpn_password | nord connect -u my_user -f - us893
```

CHAPTER 5

# Prerequesites

- GNU/Linux system
- Python 3.6
- `openvpn`
- `sudo`

nord contains many Linux-isms (e.g. using the `sudo` program to obtain root access) so it will certainly not work on Windows, it may possibly work on OSX and *BSD, but support for these platforms is not a goal.

Most recent versions of popular GNU/Linux distributions (with the exception of Debian) have both an OpenVPN client and Python 3.6 in their official repositories. Debian users will have to take *additional steps* to get a Python 3.6 installation.

## 5.1 Ubuntu 16.10 and newer

Ubuntu comes with `sudo` already installed, so we just need to install Python and openVPN:

```
sudo apt-get install python3.6 openvpn
```

## 5.2 Fedora 26 and newer

Fedora comes with `sudo` already installed, so we just need to install Python and openVPN:

```
sudo dnf install python36 openvpn
```

## 5.3 Arch Linux

Run the following as root:

```
pacman -S sudo python openvpn
```

Then configure `sudo` by following the Arch wiki to give privileges to the user that nord will be running as.

## 5.4 Debian

First run the following as root to install the openVPN client and `sudo` from the Debian repositories:

```
apt install sudo openvpn
```

Then configure `sudo` by following the Debian wiki to give privileges to the user that nord will be running as.

There are a couple of options for installing Python3.6 on Debian:

- Installing from the `unstable` repositories
- Installing from source (easier than you might think

Both of these methods are explained in top-rated answers to this stackexchange question.

CHAPTER 6

# Developing

```
git clone https://github.com/jbweston/nord
cd nord
virtualenv -p python3.6
source venv/bin/activate
pip install -e .[dev]
```

## 6.1 Building the API documentation

```
make -C docs html
xdg-open docs/build/html/index.html
```

CHAPTER 7

# Changelog

## 7.1 v0.2.0 | 2018-02-01

- Improve help messages in CLI
- Add API method to rank hosts by availability
- CLI accepts country code and selects best host automatically
- Connect to Nord API via HTTPS
- Reduce refresh period for sudo credentials to 30 seconds
- Add changelog to docs

## 7.2 v0.1.0 | 2017-09-09

- First publically released version
- Python API + command line client

# CHAPTER 8

# API documentation

## 8.1 nord.api

Interface to the NordVPN web API.

This module contains a single class, *Client*, which encapsulates all the methods provided by NordVPN.

**class** nord.api.**Client**(*api_url='https://api.nordvpn.com/'*)

Interface to the NordVPN web API.

Instances of this class can be used as async context managers to auto-close the session with the Nord API on exit.

**Parameters** **api_url** (str, default: 'https://api.nordvpn.com') –

**close**()

Close the underlying aiohttp.ClientSession.

**current_ip**()

Return our current public IP address, as detected by NordVPN.

**dns_servers**()

Return a list of ip addresses of NordVPN DNS servers.

**host_config**(*host*, *protocol='tcp'*)

Return the OpenVPN config file contents for a NordVPN host.

**Parameters**

- **host** (*str*) – This hostname may be provided either with or without the trailing '.nordvpn.com'.
- **protocol** (*str, 'tcp' or 'udp'*) –

**host_info**()

Return detailed information about all hosts.

**Returns** **host_info** – A map from hostnames to host info dictionaries.

**Return type** (dict: str → dict)

**host_load**(*host=None*)
Return the load on a NordVPN host.

**Parameters** **host** (*str, optional*) – This hostname may be provided either with or without the trailing '.nordvpn.com'. If not provided, get the load on all NordVPN hosts.

**Returns** **load** – If 'host' was provided, returns the load on the host as a percentage, otherwise returns a map from hostname to percentage load.

**Return type** int or (dict: str → int)

**rank_hosts**(*country_code*, *max_load=70*, *ping_timeout=1*)
Return hosts ranked by their suitability.

First, all the NordVPN hosts are filtered to get a list of candidates, then the round-trip time is calculated using 'ping', then the candidates are sorted according to some scoring function.

The initial filtering is done based on the country where the host is, and the max.

**Parameters**

- **country_code** (*str*) – 2-letter country code (e.g. US for United States).
- **max_load** (*int, default: 70*) – An integer between 0 and 100. Hosts with a load greater than this are filtered out.
- **ping_timeout** (*int*) – Each host will be pinged for this amount of time. Larger values yield more accurate round-trip times.

**Returns** **hosts** – Fully qualified domain names of valid hosts, sorted by their rank.

**Return type** list of str

**valid_credentials**(*username*, *password*)
Return True if NordVPN accepts the username and password.

Sometimes connecting to the VPN server gives an authentication error even if the correct credentials are given. This function is useful to first verify credentials so as to avoid unecessary reconnection attempts.

**Parameters** **password**(*username,*) –

nord.api.**normalized_hostname**(*hostname*)
Return the fully qualified domain name of a NordVPN host.

## 8.2 nord.vpn

Tools for starting and supervising OpenVPN clients.

**exception** nord.vpn.**OpenVPNError**
Errors from the OpenVPN subprocess

nord.vpn.**run**(*config*, *username*, *password*, *dns_servers=()*)
Run an OpenVPN client until it dies and return the exit code.

Optionally provide DNS servers that will replace the contents of '/etc/resolv.conf' for the duration of the client.

**Parameters**

- **config** (*str*) – The contents of the OpenVPN config file.
- **password** (*username,*) – Credentials for the OpenVPN connection.

- **dns_servers** (*tuple of str, optional*) – IP addresses of DNS servers with which to populate '/etc/resolv.conv' when the VPN is up.

nord.vpn.**start**(*config*, *username*, *password*)

Start an OpenVPN client with the given configuration.

**Parameters**

- **config** (*str*) – The contents of the OpenVPN config file.
- **password** (*username,*) – Credentials for the OpenVPN connection.

**Returns** **proc**

**Return type** asyncio.subprocess.Process

**Raises**

- PermissionError if we cannot use 'sudo' without a password.
- OpenVPNError if the OpenVPN process does not start correctly.
- LockError if a lock could not be obtained for the lockfile.

### Notes

Obtains a lock on a global lockfile before launching an OpenVPN client in a subprocess. The lock is released when the process dies.

nord.vpn.**supervise**(*proc*)

Supervise a process.

This coroutine supervises a process and writes its stdout to a logger until it dies, or until the coroutine is cancelled, when the process will be killed.

**Parameters** **proc** (*asyncio.subprocess.Process*) –

**Returns** **returncode** – 'proc.returncode'.

**Return type** int

# Python Module Index

## n

# Index