Quickly Creating a Custom Virtual Private Network (VPN)

medium.com/@sam.rothlisberger/quickly-creating-a-custom-virtual-private-network-vpn-af43bbf3fa53

Sam Rothlisberger 11 февраля 2024 г.

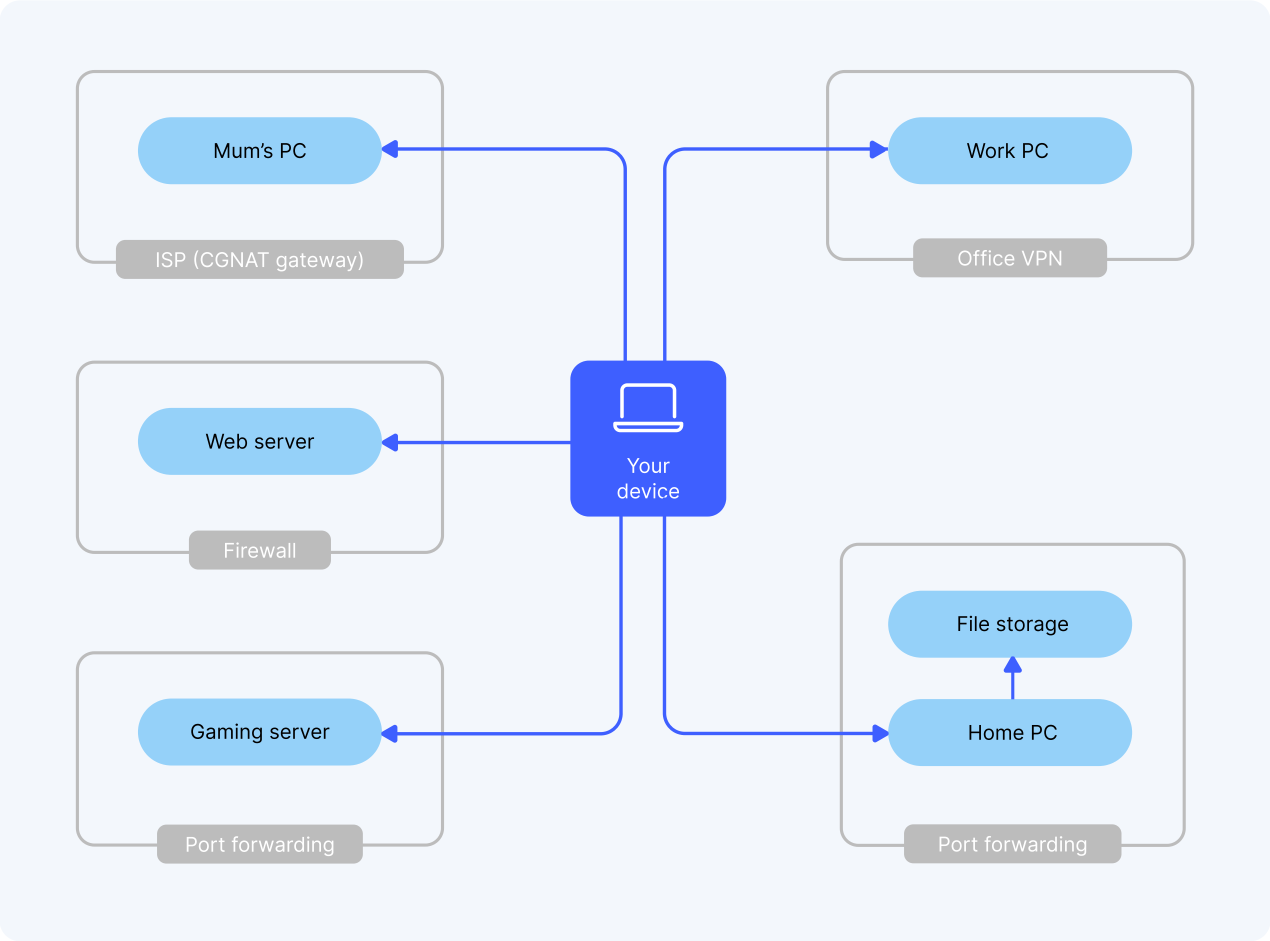
DISCLAIMER: Using these tools and methods against hosts that you do not have explicit permission to test is illegal. You are responsible for any damage you may cause by using these tools and methods.

Sam Rothlisberger

Recently I’ve been thinking of different ways to establish anonymous network sessions between multiple disparate hosts to a central server. There’s already software out there that allows you to do this like NordVPNs MeshNet, but even with that you need to download their app on your clients and server and create an account. Although you’re not connecting to NordVPN servers while using MeshNet and they are audited often for log collection and security- you still never really know especially when they collect app data that can be linked to you as seen below.

NordVPN Privacy for IOS

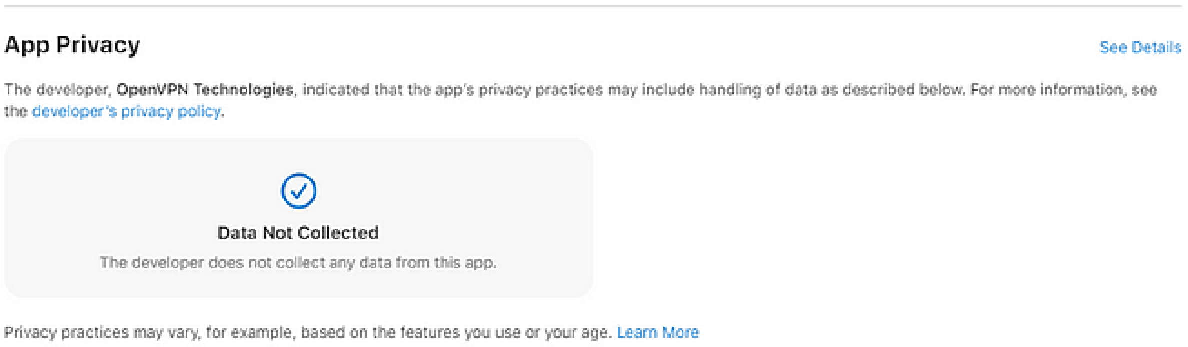
> MeshNet is a way to safely access other devices, no matter where in the world they are. Once set up, MeshNet functions just like a secure local area network (LAN) — it connects devices directly. This makes MeshNet a great fit for activities that require high speed, low latency, and advanced security — activities like file sharing, active work collaborations, and intense multiplayer gaming.



Benefits of an encrypted mesh network

- sending and receiving files securely from anywhere. For example, you can set up an SMB server easily on one of the clients and allow access only to the other VPN clients.
- Reduces latency and provides a smoother online gaming experience over LAN. You could also host secure private chatrooms on your VPN LAN with dozens of clients.
- In the case of the recent Netflix password sharing ban, where users are restricted from sharing their accounts, your Netflix account users (clients) can mimic their IP address with yours (server) and trick Netflix into thinking that you are all streaming from the same location.

I decided to do something like NordVPNs MeshNet with OpenVPN which is free, no account is needed, and it's a protocol not a service so there's no risk of logs either from a mobile app or through a normal VPN server connection. In order to make this even up to par (in terms of efficiency, not necessarily capability or security) with NordVPN, I need to make it secure and as easy as possible for automatic configuration of an arbitrary number of clients as well as the central server.



App Privacy

See Details

The developer, **OpenVPN Technologies**, indicated that the app's privacy practices may include handling of data as described below. For more information, see the developer's privacy policy.

Data Not Collected

The developer does not collect any data from this app.

Privacy practices may vary, for example, based on the features you use or your age. Learn More

OpenVPN App Privacy for IOS

Creating the Script

GitHub - srothlisberger6361/meshnet

Contribute to srothlisberger6361/meshnet development by creating an account on GitHub.

github.com

I used ChatGPT to help create the script in my GitHub repo above. AI tools make it easy to specify functions and desired output with the right queries and it really wasn't too complex once I got the base started. The script takes user input for the number of clients to create and then asks yes/no input questions that state if a user wants to email the configs to the clients and start the OpenVPN server automatically. If the user wants to distribute the client.ovpn files another way, then they can enter "no" on the email option and they will still be saved in the current directory. This is the same if you want to start server.conf at a later time.

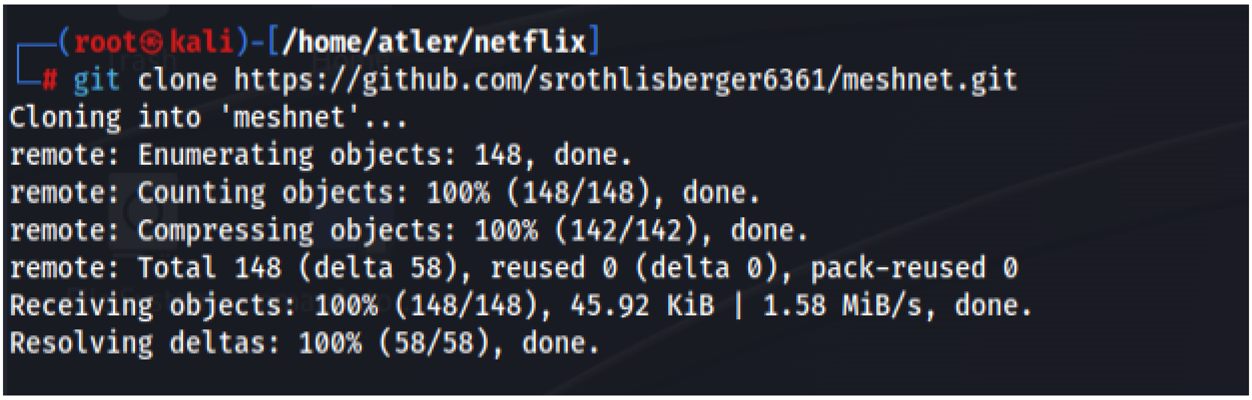
Input:

1. outlook email
2. outlook email app password
3. server public IP address
4. user/password for basic authentication
5. number of clients
6. client emails

Output:

1. client.ovpn files (labeled client1.ovpn, client2.ovpn, etc.)
2. server.conf (for starting the server)

Downloading the script



```
┌──(root㉿kali)-[/home/atler/netflix]
└─# git clone https://github.com/srothlisberger6361/meshnet.git
Cloning into 'meshnet'...
remote: Enumerating objects: 148, done.
remote: Counting objects: 100% (148/148), done.
remote: Compressing objects: 100% (142/142), done.
remote: Total 148 (delta 58), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (148/148), 45.92 KiB | 1.58 MiB/s, done.
Resolving deltas: 100% (58/58), done.
```

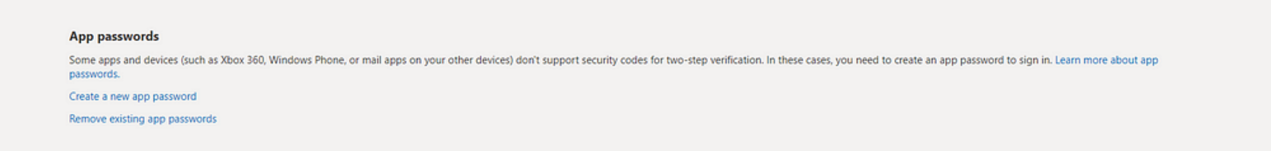
In order to send the client config files via email, you should create a burner outlook account and set up an app password. After you create your burner account you need to go into security>advanced tab on your profile, enable 2FA, and then enable App Passwords following this resource below.

Manage app passwords for two-step verification

Important: Your administrator might not allow you to use app passwords. If you don't see App passwords as an option…

support.microsoft.com

Copy and paste the password into a safe place for use later in your command.



App passwords

Some apps and devices (such as Xbox 360, Windows Phone, or mail apps on your other devices) don't support security codes for two-step verification. In these cases, you need to create an app password to sign in. Learn more about app passwords.

Create a new app password

Remove existing app passwords

I was doing this with a Gmail burner account for alittle bit, but then it just stopped working in my script. Others in online forums seemed to have the same problem and recommended outlook instead. The only annoying thing with some email providers is sometimes they put the email in the junk folder of the recipient which might be a TLS/SSL issue with Outlook. Either way, I'm not too worried because the client would be expecting to receive it. You also might need to generate a new App password every once in a while, but atleast its easy to input in our script and you should really only have to run the script once. Let's look at some of the security features involved in our script before we run it.

Server Verification by Clients

Clients verify the authenticity of the OpenVPN server using the server's certificate (server.crt), which is signed by the Certificate Authority (CA) (ca.crt). This verification ensures that clients are connecting to the legitimate OpenVPN server and not to a malicious entity.

Client Verification by Server

The OpenVPN server verifies the authenticity of connecting clients using their individual client certificates (client1.crt, client2.crt, etc.), each signed by the same CA (ca.crt). This ensures that only clients with valid and authorized certificates are allowed to connect to the server.

TLS Authentication

The TLS Authentication key (ta.key) is used in both the server and client configurations to add an extra layer of security to the TLS handshake. The ta.key file is used to strengthen the security of the TLS control channel by adding an additional layer of HMAC (Hash-based Message Authentication Code) authentication. Any UDP packet not bearing the correct HMAC signature can be dropped without further processing. It can protect against:

- DoS attacks or port flooding on the OpenVPN UDP port.
- Port scanning to determine which server UDP ports are in a listening state.
- Buffer overflow vulnerabilities in the SSL/TLS implementation.
- SSL/TLS handshake initiations from unauthorized machines (while such handshakes would ultimately fail to authenticate, can cut them off at a much earlier point).

Basic Authentication (-a jack:qwerty)

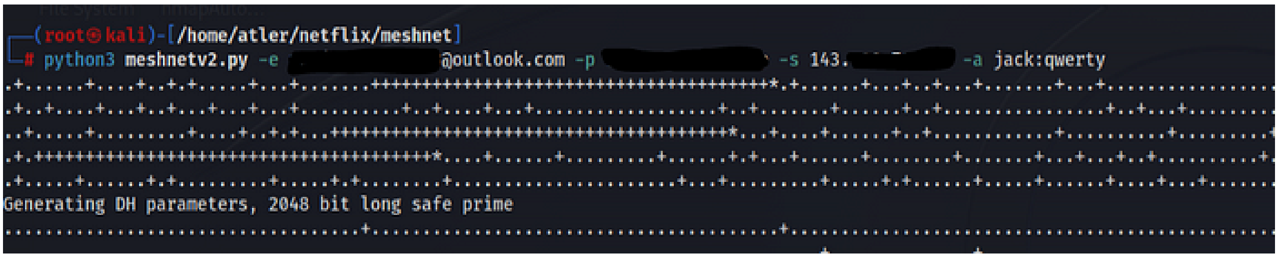
The auth-user-pass option references the auth-script.sh on the server which is the user specified username/password that the clients will be prompted for when connecting. So it’s not enough to only have the ta.key and the ca.crt within the ovpn file, you also need the username/password.

UDP instead of TCP

While OpenVPN allows either the TCP or UDP protocol to be used as the VPN carrier connection, the UDP protocol will provide better protection against DoS attacks and port scanning than TCP

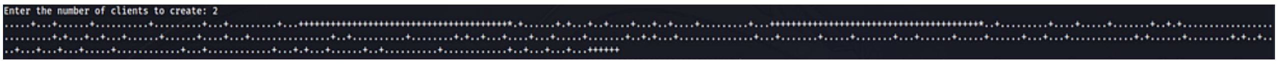
Executing the Script

> python3 meshnet.py -e [outlook email] -p [App Password] -s [VPN Server IP] -a [username:password]



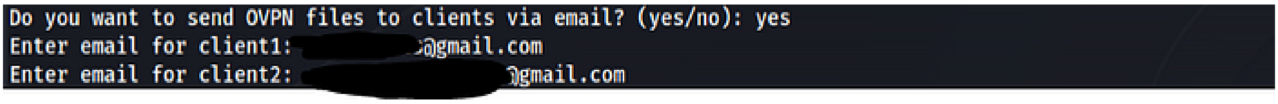
```
┌──(root㉿kali)-[/home/atler/netflix/meshnet]
└─# python3 meshnetv2.py -e [redacted]@outlook.com -p [redacted] -s 143.[redacted] -a jack:qwerty
.+......+....+..+.+......+...+.......++++++++++++++++++++++++++++++++++++++*.+.......+...+..+...+........+...+................
.+..+....+...+..+...+...+...+..........+..+....+...+..................+..+........+.......+..+.................+..+...+........
..+......+..........+....+..+.+...++++++++++++++++++++++++++++++++++++++*...+....+.......+..+..............+............+.........
.+.+++++++++++++++++++++++++++++++++++++*....+......+.........+.......+.+...+.......+.........+.......+....+...+..+...........+.
.+.....+.......+.+.........+.....+.+........+........................+...+..........+.....+.+.......+......+.......+....+...+......
Generating DH parameters, 2048 bit long safe prime
...................................+.........................................+...........................................
```

Execute the python script, specifying the email/app password, server IP, and username/password for clients



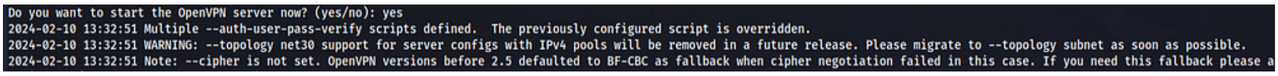
```
Enter the number of clients to create: 2
.....+...+.......+..........+.........+...+.........+...++++++++++++++++++++++++++++++++++++++++*.+......+..+...+..+....+...+..+....+.........+...++++++++++++++++++++++++++++++++++++++++*..+.........+.....+.....+........+..+.+..............
.........+.+...+..+..+...+......+......+.....+....+...+..............+..+...........+........+..+..+...+.....+......+.......+..+...+..............+...+.......+.....+.......+...+......+.....+......+...+...+...........+.......+........+.+..
..+...+...+...+.....+............+...+............+...+.+...+......+..+...........+...........+..+...+...+...++++++
```

Specify the number of clients to make configuration files for



```
Do you want to send OVPN files to clients via email? (yes/no): yes
Enter email for client1: [redacted]@gmail.com
Enter email for client2: [redacted]@gmail.com
```

Email the custom configuration files to each client respectively



```
Do you want to start the OpenVPN server now? (yes/no): yes
2024-02-10 13:32:51 Multiple --auth-user-pass-verify scripts defined.  The previously configured script is overridden.
2024-02-10 13:32:51 WARNING: --topology net30 support for server configs with IPv4 pools will be removed in a future release. Please migrate to --topology subnet as soon as possible.
2024-02-10 13:32:51 Note: --cipher is not set. OpenVPN versions before 2.5 defaulted to BF-CBC as fallback when cipher negotiation failed in this case. If you need this fallback please a
```

Start the VPN Server to accept connections from clients

Port Forwarding

Now that we know how the python script works, we need to set up our path from the client devices to our VPN server. We are going to port forward on both our VPS droplet (which is the proxy) and also on our Raspberry Pi server which is where internet will be tunneling in and out of on the home network for all our clients. A Raspberry Pi with a normal linux image is ideal for our VPN server because we can keep it on 24/7 no problem. And I don't necessarily want to do that with my desktop. Let's get into port forwarding requirements on the VPS instance first.

> NOTE: You could just go directly from your clients to your local server through your router, bypassing the need for a VPS. I just did it to slightly increase client anonymity if my VPS was non-attributable (which it isn't). Putting your VPN server on a VPS server from AWS or Digital Ocean will likely have the IP blacklisted for certain services, so it's better to use your home router as the Server access point.

Port Forwarding: Clients → VPS → Router

We want any of the clients from any IP to be able to connect to the VPS instance so we'll simply forward connections on port 50000 on eth0 to port 50000 on our public router IP. The below commands are ran on your proxy VPS if you have one.

Enable IP forwarding

> sudo sysctl -w net.ipv4.ip_forward=1

DNAT rule for incoming UDP traffic

> sudo iptables -t nat -A PREROUTING -i eth0 -p udp — dport 50000 -j DNAT — to-destination <ROUTER_IP>:50000

MASQUERADE rule for outgoing UDP traffic

> sudo iptables -t nat -A POSTROUTING -o eth0 -p udp — dport 50000 -j MASQUERADE

Forwarding rule for UDP traffic

```
sudo iptables -A FORWARD -i eth0 -o eth0 -p udp — sport 50000 -j ACCEPT
```

Enable persistent iptables rules

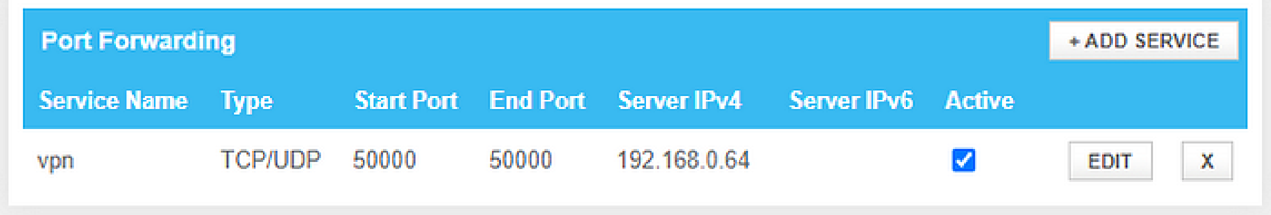
```
apt-get install iptables-persistent

netfilter-persistent save

netfilter-persistent reload
```

Port Forwarding: Router → Raspberry Pi

Now we need to forward port 50000 on the router to the private IP of the Raspberry Pi or whatever host you're using.



Port Forwarding

+ ADD SERVICE

Service Name	Type	Start Port	End Port	Server IPv4	Server IPv6	Active		
vpn	TCP/UDP	50000	50000	192.168.0.64		☑	EDIT	X

Port Forwarding: Raspberry Pi → internet

Finally, we need to port forward connections from the tun0 interface (the VPN server) to eth0 (the internet). The below commands are run on your Raspberry Pi server as root.

Enable IP forwarding

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

Allow incoming traffic to port 50000 on tun0

```
iptables -A INPUT -i tun0 -p udp — dport 50000 -j ACCEPT
```

Allow established and related connections

```
iptables -A FORWARD -m state — state RELATED,ESTABLISHED -j ACCEPT
```

Allow outgoing traffic from the server to the internet (eth0)

```
iptables -A FORWARD -i tun0 -o eth0 -j ACCEPT
```

Masquerade (NAT) traffic going out through eth0 for VPN clients

```
iptables -t nat -A POSTROUTING -o eth0 -s 10.8.0.0/24 -j MASQUERADE
```

Forward traffic from tun0 to eth0

```
iptables -A FORWARD -i tun0 -o eth0 -j ACCEPT
```

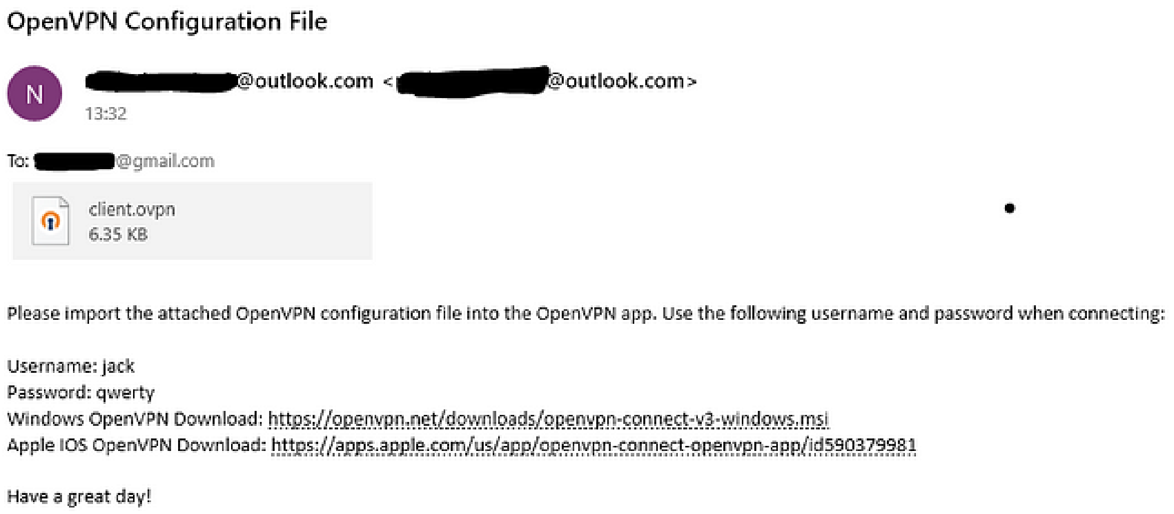
Enable persistent iptables rules

```
apt-get install iptables-persistent

netfilter-persistent save

netfilter-persistent reload
```

Joining the Network (Windows and Apple IOS)



OpenVPN Configuration File

N [redacted]@outlook.com <[redacted]@outlook.com>
13:32

To: [redacted]@gmail.com

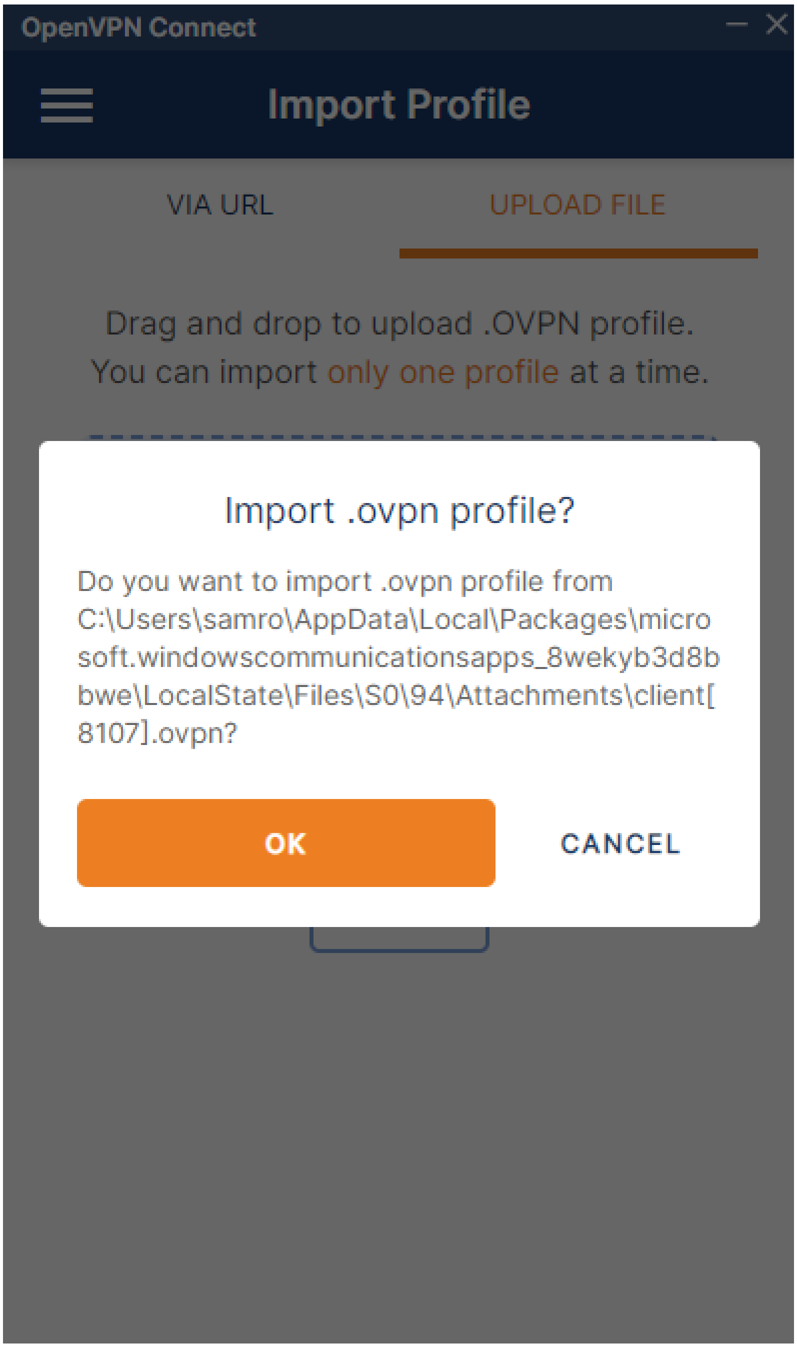
client.ovpn
6.35 KB

Please import the attached OpenVPN configuration file into the OpenVPN app. Use the following username and password when connecting:

Username: jack
Password: qwerty
Windows OpenVPN Download: https://openvpn.net/downloads/openvpn-connect-v3-windows.msi
Apple IOS OpenVPN Download: https://apps.apple.com/us/app/openvpn-connect-openvpn-app/id590379981

Have a great day!

The user can go into their email and receives instructions on how to join the network. On Windows, download OpenVPN from the link in the email if needed. The client can simply click the attachment in the email and select “ok”.

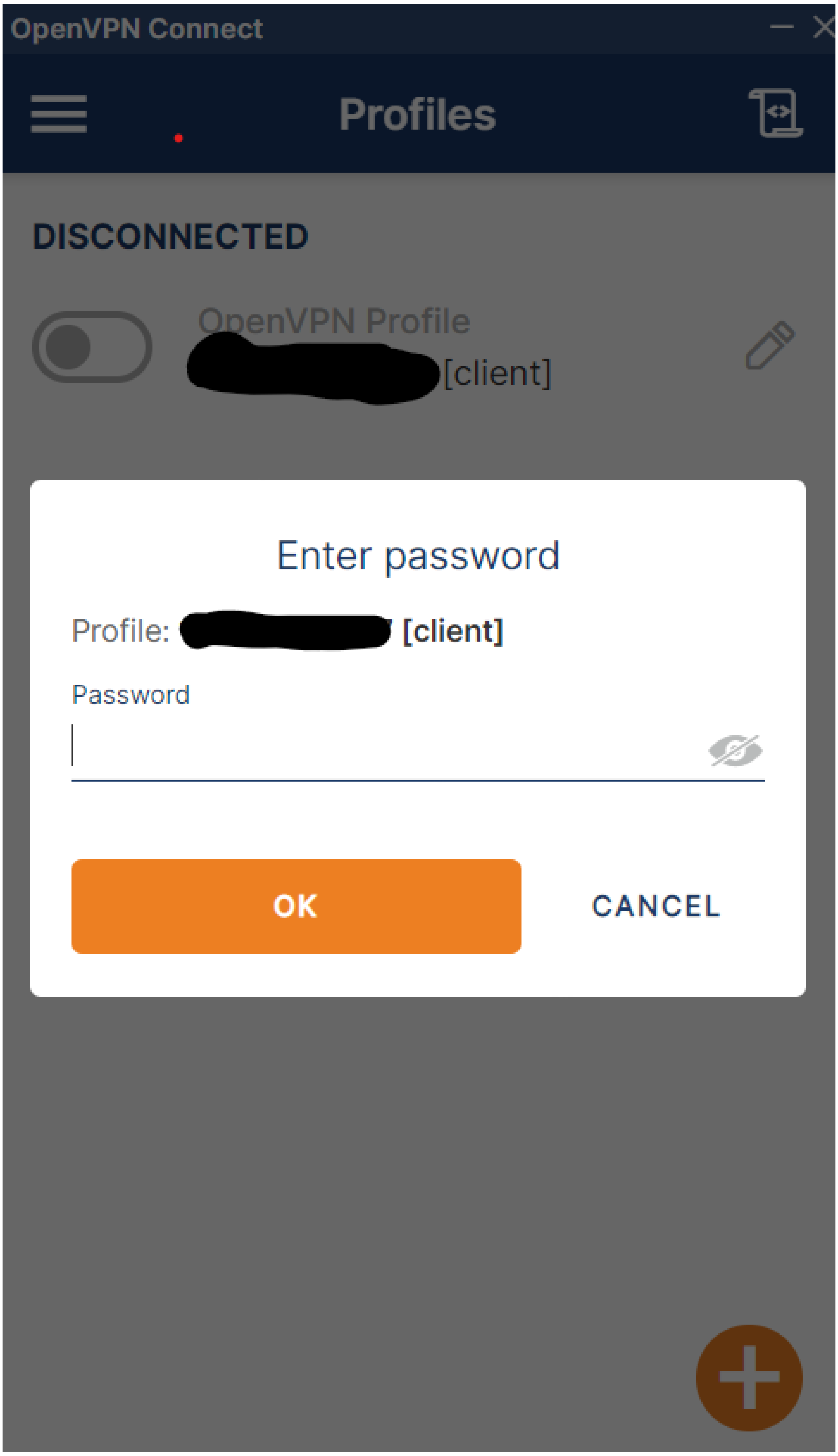


OpenVPN Connect
Import Profile
VIA URL
UPLOAD FILE
Drag and drop to upload .OVPN profile.
You can import only one profile at a time.

Import .ovpn profile?

Do you want to import .ovpn profile from C:\Users\samro\AppData\Local\Packages\microsoft.windowscommunicationsapps_8wekyb3d8bbwe\LocalState\Files\S0\94\Attachments\client[8107].ovpn?

Put in the specified username and password from the email:



DISCONNECTED

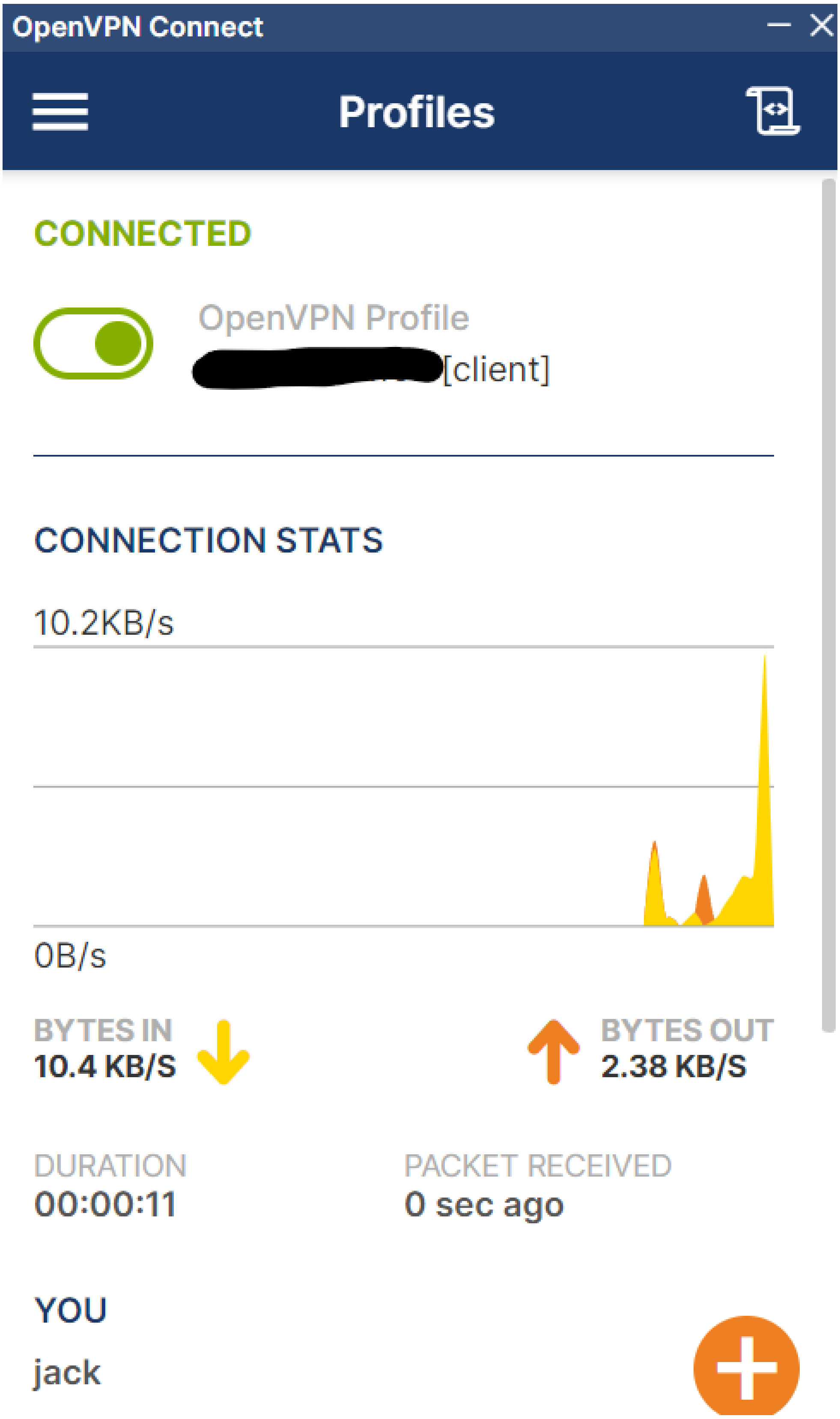
OpenVPN Profile
[redacted] [client]

Enter password

Profile: [redacted] [client]

Password

The client is issued an IP in the range 10.8.0.0/24 and all the internet traffic is tunneled through the Raspberry Pi server eth0 interface with IP 192.168.0.64 on my local network. That means all our requests use the public IP of my home router no matter where we are!



OpenVPN Connect

Profiles

CONNECTED

OpenVPN Profile
[redacted][client]

CONNECTION STATS

10.2KB/s

0B/s

BYTES IN
10.4 KB/S

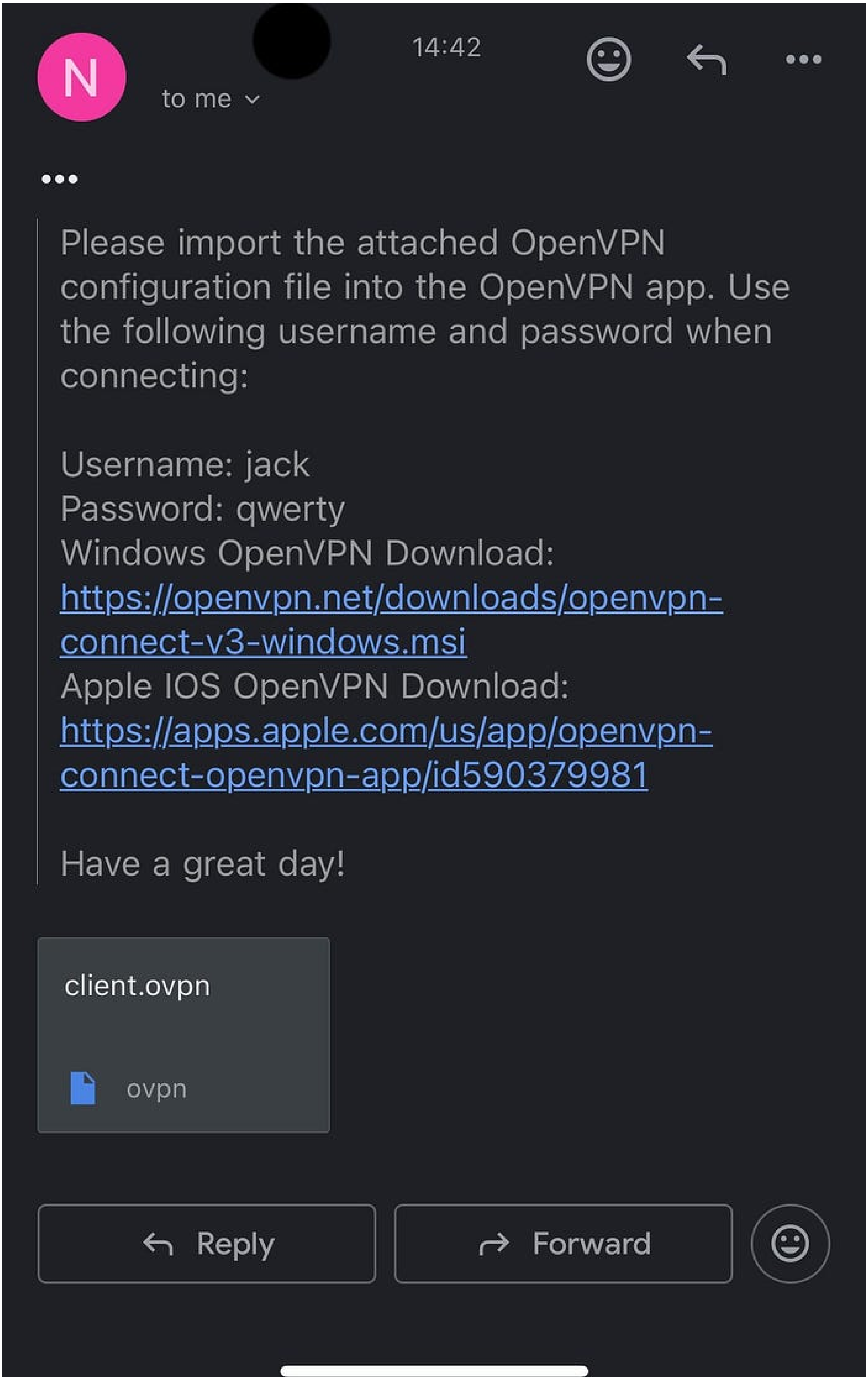
BYTES OUT
2.38 KB/S

DURATION
00:00:11

PACKET RECEIVED
0 sec ago

YOU
jack

To join the VPN on iPhone, the process is similar. Click the email link to the Appstore and download if you don't have it already.



Please import the attached OpenVPN configuration file into the OpenVPN app. Use the following username and password when connecting:

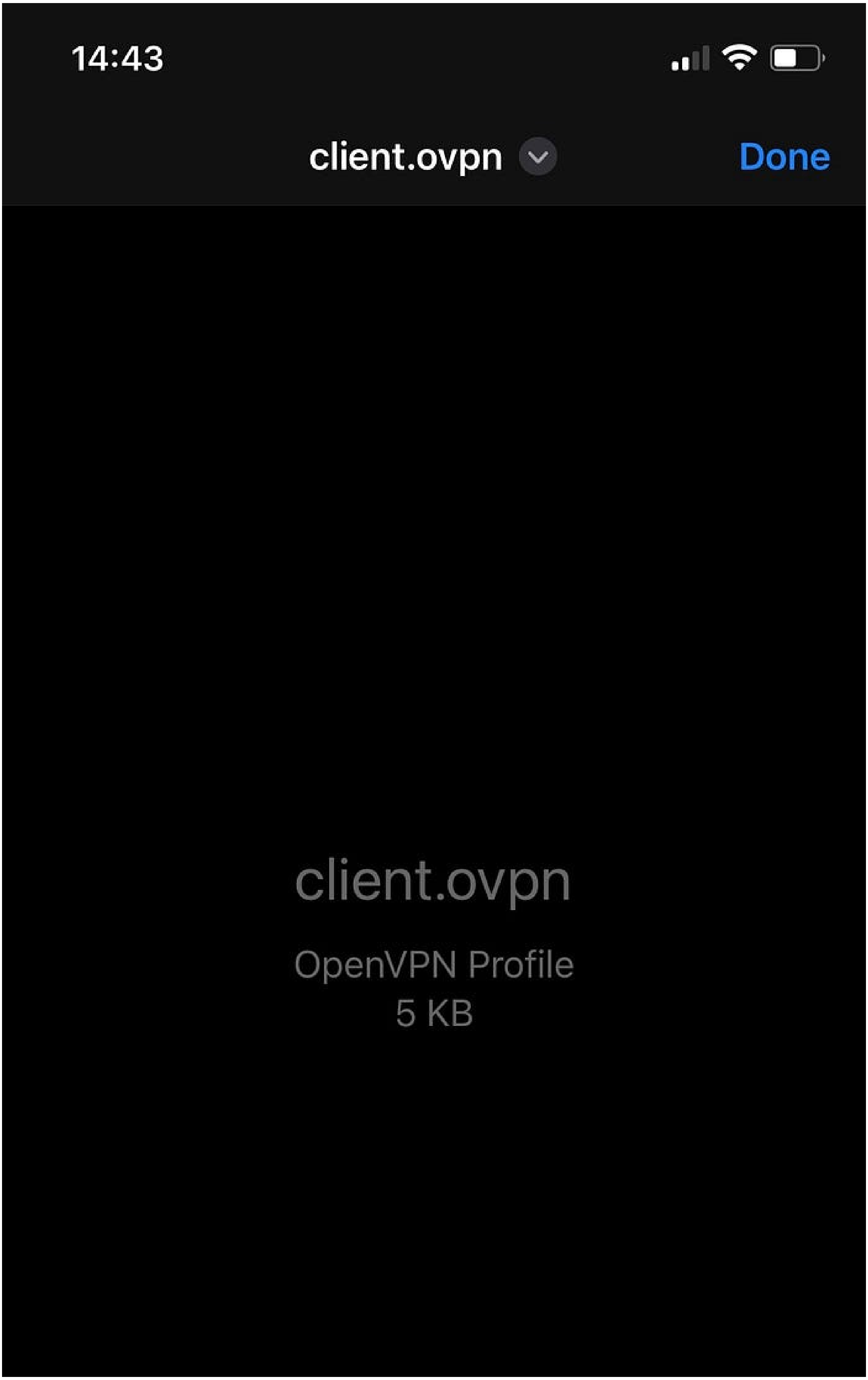
Username: jack
Password: qwerty
Windows OpenVPN Download:
https://openvpn.net/downloads/openvpn-connect-v3-windows.msi
Apple IOS OpenVPN Download:
https://apps.apple.com/us/app/openvpn-connect-openvpn-app/id590379981

Have a great day!

client.ovpn

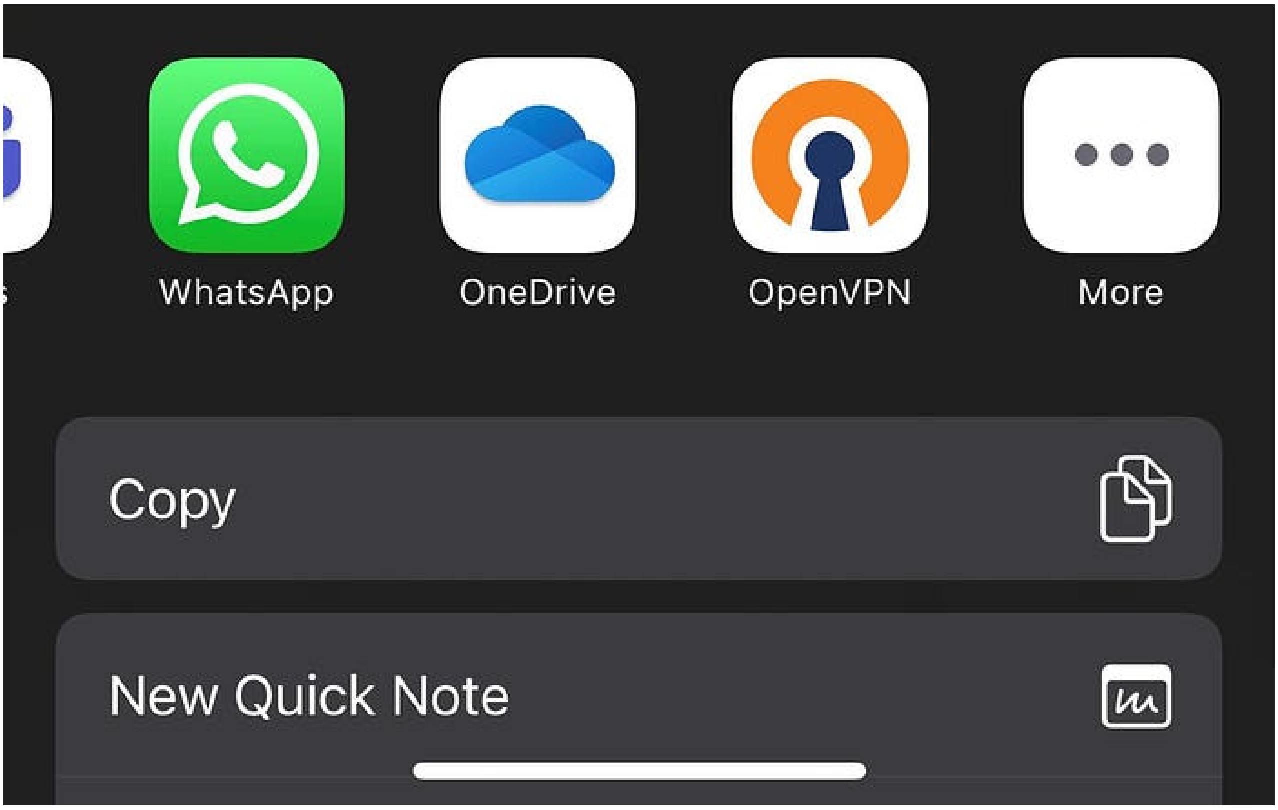
ovpn

Then click on the file and select the icon in the bottom left corner and navigate to the OpenVPN app.

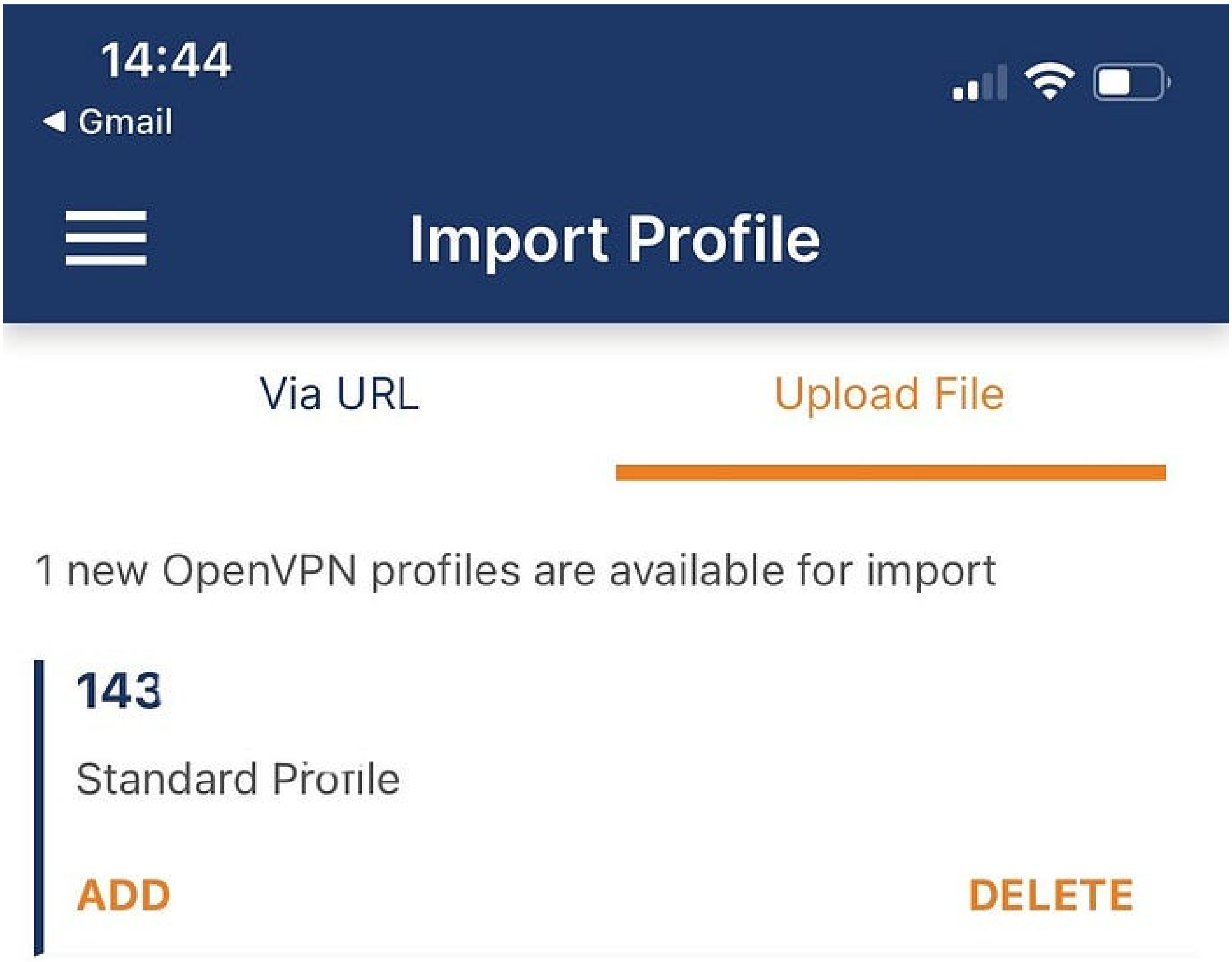


client.ovpn

OpenVPN Profile
5 KB



Then the app opens and a new profile is added for the configuration.



1 new OpenVPN profiles are available for import

143

Standard Profile

Then you can start the connection by entering the username and password exactly how we did for the windows client. Each client will have an IP on the 10.8.0.0/24 network and can interact with the internet through the home network.

Bypassing Netflix Password Sharing

We can bypass Netflix's password sharing block by basically making a shared "home" network for disparate users which could be our friends or family members. Ultimately, they would all use the same home public IP of the router to log into Netflix. Netflix also uses device IDs and account activity to identify password sharing, but mainly the Public IP of the router which is anchored to the main smart TV (or streaming device) operating on the home network. As long as you're connecting to the VPN from a portable device like a laptop or iPhone and the server a device on the home network, you should be good. From the external clients devices, friends and family could use Chromecast to put it on their TV since they can't use Roku.

Google Chromecast (3rd Generation) Media Streamer - Black

Amazon.com: Google Chromecast (3rd Generation) Media Streamer - Black : Electronics

www.amazon.com

The great thing about this setup is your friends/family can just connect to your remote VPN server, log into Netflix through their browser/app, and then disconnect to watch Netflix wherever they are for about 31 days before you need to connect to "home" again. I hope you enjoyed this post!