OpenVPN Connect Android FAQ

Q: How to get started?

A: To use this app, you must have an OpenVPN profile and a server to connect to. OpenVPN profiles are files that have an extension of **.ovpn**

There are several methods available to import a profile:

- If you have a .ovpn profile, copy the profile and any files it references to the SD card folder on your device (copy all files to the same folder). Then go to Menu / Import / Import Profile from SD card.
- If you have an account on an OpenVPN Access Server, you can import the profile directly from the Access Server by going to Menu / Import / Import Access Server Profile.
- If you have an account on the Private Tunnel service, go to Menu / Import / Import Private Tunnel Profile.

Q: Is OpenVPN Connect for Android vulnerable to Heartbleed?

A: No, all versions of OpenVPN Connect for Android use the PolarSSL library, which is immune to Heartbleed.

Q: Are CRLs (certificate revocation lists) supported?

A: Yes, CRLs are supported starting with version 1.1.14 for Android.

To use a CRL, it must be added to the .ovpn profile, such as:

```
<crl-verify>
-----BEGIN X509 CRL-----
MIHxMFwwDQYJKoZIhvcNAQEEBQAwFTETMBEGA1UEAxMKT3BlblZQTiBDQRcNMTQw
NDIyMDQzOTI3WhcNMjQwNDE5MDQzOTI3WjAWMBQCAQEYDzIwMTQwNDIyMDQzOTI3
WjANBgkqhkiG9w0BAQQFAAOBgQBQXzbNjXkx8+/TeG8qbFQD5wd6wOTe8HnypQTt
eELsI7eyNtiRRhJD3qKfawPVUabSijnwhAPHfhoIOLKe67RLfzOwAsFKPNJAVdmq
rYw1t2eucHvGjH8PnTh0aJPJaI67jmNbSI4CnHNcRgZ+1ow1GS+RAK7kotS+dZz9
0tc7Qw==
-----END X509 CRL-----
</crl-verify>
```

Multiple CRLs may be concatenated together within the **crl-verify** block above.

If you are importing a .ovpn file that references an external CRL file such as

```
crl-verify crl.pem
```

make sure to drop the file **crl.pem** into the same place as the .ovpn file during import, so the profile parser can access it.

Q: I am having trouble importing my .ovpn file.

A: Here are some basic pointers for importing .ovpn files:

- When you import a .ovpn file, make sure that all files referenced by the .ovpn file such as **ca**, **cert**, and **key** files are in the same directory on the device as the .ovpn file.
- Profiles must be UTF-8 (or ASCII) and under 256 KB in size.
- Consider using the **unified** format for OpenVPN profiles which allows all certs and keys to be embedded into the .ovpn file. This eases management of the OpenVPN configuration because it integrates all elements of the configuration into a single file.

  For example, a traditional OpenVPN profile might specify certs and keys as follows:

```
ca ca.crt
cert client.crt
key client.key
tls-auth ta.key 1
```

  You can convert this usage to **unified form** by pasting the content of the certificate and key files directly into the OpenVPN profile as follows using an XML-like syntax:

```
<ca>
-----BEGIN CERTIFICATE-----
MIIBszCCARygAwIBAgIE...
. . .
/NygscQs1bxBSZ0X3KRk...
Lq9iNBNgWg==
-----END CERTIFICATE-----
</ca>

<cert>
-----BEGIN CERTIFICATE-----
. . .
</cert>

<key>
-----BEGIN RSA PRIVATE KEY-----
. . .
</key>

key-direction 1
<tls-auth>
-----BEGIN OpenVPN Static key V1-----
. . .
</key>
```

  Another approach to eliminate certificates and keys from the OpenVPN profile is to use the Android Keychain as described below.

  NOTE: when converting **tls-auth** to unified format, check if there is a second parameter after the filename (usually a 0 or 1). This parameter is known as the **key-direction**parameter and must be specified as a standalone directive when **tls-auth** is converted to unified format. For example if the parameter is 1, add this line to the profile:

```
key-direction 1
```

  If there is no second parameter to **tls-auth**, you must add this line to the profile:

```
key-direction bidirectional
```

Q: Where are the support forums for OpenVPN Connect?

A: https://forums.openvpn.net/

Q: Is IPv6 supported?

A: Yes. The OpenVPN app supports IPv6 transport and IPv6 tunnels as long as the server supports them as well.

Q: Why does OpenVPN Connect show two notification icons when connected?

A: This is something Android requires to affirm that the VPN session is high priority and should not be arbitrarily terminated by the system.

Q: Can I disable the connection notification sound?

A: On some Android devices, a connection notification sound is played by Android whenever a VPN tunnel is established, and cannot be silenced by a non-root app.

Note that it is possible to reduce the frequency of these notifications by going to the Preferences menu and selecting the**Seamless Tunnel** option.

Q: How can I maximize battery life?

A: Consider selecting the **Battery Saver** option in the Preferences menu to Pause the VPN when the device screen is blanked. This will cause the VPN to disconnect when the screen is blanked and automatically reconnect when the screen becomes visible again. While this option can extend battery life, it should not be used if you have apps running in the background that require continuous access to the internet via the VPN (such as a new email notifier).

Note that if you select both the **Battery Saver** and **Seamless Tunnel** options, you will block any app from reaching the internet while the VPN is active but the device screen is blanked. This can be useful for additional energy savings, as long as you don't have any background apps that need constant internet access.

Q: Can I control the VPN from outside the app?

A: Yes, using shortcuts. Go to Menu / Add Shortcut to add a shortcut to your home page. Shortcuts can be created for:

- connecting a specific profile,
- disconnecting, and
- launching the app

Q: How can I ensure that the VPN stays continuously connected?

A: In the Preferences menu, select the **Reconnect on reboot**option. Also, consider setting

the **Connection Timeout**preference to "continuously retry". If you want to prevent apps from accessing the internet, except through the VPN, select the**Seamless Tunnel** preference.

Q: Why does the VPN disconnect when I make or receive a voice call?

A: Some cellular networks are incapable of maintaining a data connection during a voice call. If Android detects this as a loss of network connectivity, the VPN should enter a pause state during the duration of the call, and automatically resume after the call is complete. However if the loss of data connectivity isn't detected by Android, the VPN connection may time out and disconnect.

Q: Given that mobile devices are easily lost or stolen, how best to secure VPN profiles against compromise if the device falls into the wrong hands?

- A: The most sensitive piece of data in a profile is the private key. Consider removing the client certificate and private key from the profile and save them in the device Keychain instead (this is discussed below).
- Use a strong device-level password. This is critical to protect data stored in the device Keychain.

Q: Is it safe to save passwords?

A: If you check the **Save** checkbox on the authentication or private key password fields, the app will store your password in an encrypted form, however a determined attacker with physical possession of the device would still be able to recover the password with some reverse engineering.

Currently, the best options for security are to avoid saving passwords, and to use the Android Keychain as a repository for your private key (see below).

The Android developers are in the process of implementing an API for secure storage of passwords that will leverage on the hardware-backed keystore and master device password, however this development is not complete as of Android 4.2. This approach will protect saved passwords even if the device is rooted. When this development is complete, we plan to support it in the app.

Q: Why is the save password switch sometimes disabled?

A: The save password switch on the authentication password field is normally enabled, but can be disabled by the following:

- The following OpenVPN directive, if present in a profile, will disable the password save switch:

  ```
  setenv ALLOW_PASSWORD_SAVE 0
  ```

  Note however that the above directive only applies to the authentication password. The private key password, if it exists, can always be saved.

Q: How to make the app work with profiles that lack a client certificate/key?

A: If you have a profile that connects to a server without a client certificate/key, you will need to add the following directive to your profile:

```
setenv CLIENT_CERT 0
```

This is necessary to resolve an ambiguity when the profile contains no client certificate or key, because otherwise the client app can't know whether an external certificate/key pair should be obtained from the Android Keychain, or whether the server actually doesn't require a client certificate/key (for example if the server is configured with the **client-cert-not-required** directive). The option is given as a "setenv" to avoid breaking other OpenVPN clients that might not recognize it.

Q: Why doesn't the app support tap-style tunnels?

A: The Android VPN API supports only tun-style tunnels at the moment. This is a limitation of the Android platform. If you try to connect a profile that uses a tap-based tunnel, you will get an error that only layer 3 tunnels are currently supported.

If you really want to see tap-style tunnels supported in OpenVPN Connect, we would encourage you to contact the Google Android team and ask that the VpnService API be extended to allow this. Without such changes to the VpnService API, it is not possible for non-root apps such as OpenVPN Connect to support tap-style tunnels.

Q: Are there any OpenVPN directives not supported by the app?

A: While most OpenVPN client directives are supported by the app, we have made an effort to reduce bloat and improve maintainability by eliminating what we believe to be obsolete or rarely-used directives. Please email us at android@openvpn.netif you believe that a specific directive that is not included should be reconsidered for inclusion.

Here is a partial list of directives not currently supported:

- **dev tap** — This directive is not supported because the underlying Android VPN API doesn't support tap-style tunnels.
- **fragment** — The fragment directive is not supported due to the complexity it adds to the OpenVPN implementation and the fact that it is usually better to leave fragmentation up to the lower-level transport protocols. Note as well that the client does not support connecting to a server that uses the fragment directive.
- **mssfix** — This directive will be added in a future release. Since the functionality of mssfix can be achieved on either the client or server side, specifying it on the server side will enable it even if the client doesn't support the directive.
- **secret** — Static key encryption mode (non-TLS) is not supported.
- **socks-proxy** — Socks proxy support is currently not supported.
- Ciphers other than AES, Blowfish, and DES family — Currently, only AES, Blowfish, and DES family ciphers are supported. This is done to reduce bloat and improve energy efficiency. The AES cipher algorithm, in particular, is well-suited for the ARM processor generally used in Android devices.
- proxy directives — While proxy directives are currently supported (**http-proxy** and **http-**

proxy-option), they are currently NOT supported in **<connection>** profiles.

Q: Can I have multiple profiles?

A: Yes, you can import any number of profiles from the Import menu -- tap the profile field to select one. Keep in mind that OpenVPN will assign a name to a profile based on the server that the profile connects to. If you import a profile with the same name as one that already exists, the new profile will replace the old one. You can prevent this from happening by renaming the old profile.

Q: How do I delete or rename a profile?

A: Doing a "long touch" on the profile field will bring up a context menu for that profile that includes delete, rename, etc.

Q: Can I have multiple proxies?

A: Yes, you can add any number of proxies from the main menu. Once a proxy is added, a proxy selection field will appear on the main page. Tap the field to select a proxy or **None** at the end of the list to connect directly.

Q: How do I edit or delete a proxy?

A: Doing a "long touch" on the proxy field will bring up a context menu for that proxy that includes edit, delete, etc.

Q: How do I use a client certificate and private key from the Android Keychain?

A: Using the Android keychain to store your private key has the added security advantage of leveraging on the hardware-backed keystores that exist on many Android devices, allowing the key to be protected by the Android-level device password, and preventing key compromise even if the device is rooted.

If you already have your client certificate and private key bundled into a PKCS#12 file (extension **.p12** or **.pfx**), you can import it into the Android Keychain using either the Import menu or the Settings app.

If you don't have a PKCS#12 file, you can convert your certificate and key files into PKCS#12 form using this **openssl**command (where **cert**, **key**, and **ca** are your client certificate, client key, and root CA files).

```
openssl pkcs12 -export -in cert -inkey key -certfile ca -name MyClient -out
client.p12
```

Then import the **client.p12** file from the previous step into the app using the Import / Import PKCS#12 menu option.

Once this is done, remove the **ca**, **cert**, and **key** directives from your .ovpn file and re-import it. When you connect the first time, the app will ask you to select a certificate to use for the profile.

Just select the **MyClient** certificate and you should be able to connect normally.

Q: When I try to import a PKCS#12 file, why am I being asked for a password?

A: When you generate a PKCS#12 file, you will always be asked for an "export password" to encrypt the file. This password must again be presented when the PKCS#12 file is imported into the Android Keychain. This is to prevent interception and recovery of the private key during transport.

Q: Why doesn't the PKCS#12 file in my OpenVPN configuration file work the same as on desktop systems?

PKCS#12 files on Android are used somewhat differently than on desktop versions of OpenVPN. In desktop versions, PKCS#12 files can be bundled or referenced in the OpenVPN profile. On Android, however, PKCS#12 management is built into the Android Keychain. This approach is much better from a security perspective, because the Keychain can then leverage on hardware features in the device such as hardware-backed keystores. However, it does require that the PKCS#12 file is loaded into the Android Keychain as a separate step from importing the OpenVPN profile. It also moves the responsibility for managing PKCS#12 files to the Android Keychain, and away from OpenVPN, so it can potentially introduce compatibility issues.

To use a PKCS#12 file on Android, see the FAQ item above:*How do I use a client certificate and private key from the Android Keychain?*

Q: How do I set up my profile for server failover?

A: You can provide OpenVPN with a list of servers to connect to. On connection failure, OpenVPN will rotate through the list until it finds a responsive server. For example, the following entries in the profile will first try to connect to server A via UDP port 1194, then TCP port 443, then repeat the process with server B. OpenVPN will continue to retry until it successfully connects or hits the Connection Timeout, which can be configured in the Preferences.

```
remote server-a.example.tld 1194 udp
remote server-a.example.tld 443 tcp
remote server-b.example.tld 1194 udp
remote server-b.example.tld 443 tcp
```

Q: How can I contact the developers about bugs or feature requests?

A: Send email to android@openvpn.net.