How to Set Up an IKEv2 VPN Server with StrongSwan on Ubuntu 18.04 | DigitalOcean

Step 2 — Creating a Certificate Authority ⋮ 20-25 minutes

Introduction

A virtual private network, or VPN, allows you to securely encrypt traffic as it travels through untrusted networks, such as those at the coffee shop, a conference, or an airport.

IKEv2, or Internet Key Exchange v2, is a protocol that allows for direct IPSec tunneling between the server and client. In IKEv2 VPN implementations, IPSec provides encryption for the network traffic. IKEv2 is natively supported on some platforms (OS X 10.11+, iOS 9.1+, and Windows 10) with no additional applications necessary, and it handles client hiccups quite smoothly.

In this tutorial, you'll set up an IKEv2 VPN server using StrongSwan on an Ubuntu 18.04 server and connect to it from Windows, macOS, Ubuntu, iOS, and Android clients.

Prerequisites

To complete this tutorial, you will need:

- One Ubuntu 18.04 server configured by following the Ubuntu 18.04 initial server setup guide, including a `sudo` non-root user and a firewall.

Step 1 — Installing StrongSwan

First, we'll install StrongSwan, an open-source IPSec daemon which we'll configure as our VPN server. We'll also install the public key infrastructure component so that we can create a certificate authority to provide credentials for our infrastructure.

Update the local package cache and install the software by typing:

Now that everything's installed, let's move on to creating our certificates.

An IKEv2 server requires a certificate to identify itself to clients. To help us create the certificate required, the `strongswan-pki` package comes with a utility to generate a certificate authority and server certificates. To begin, let's create a few directories to store all the assets we'll be working on. The directory structure matches some of the directories in `/etc/ipsec.d`, where we will eventually move all of the items we create. We'll lock down the permissions so that our private files can't be seen by other users:

Now that we have a directory structure to store everything, we can generate a root key. This will be a 4096-bit RSA key that will be used to sign our root certificate authority.

Execute these commands to generate the key:

Now that we have a key, we can move on to creating our root certificate authority, using the key to sign the root certificate:

You can change the *distinguished name* (DN) values to something else to if you would like. The common name here is just the indicator, so it doesn't have to match anything in your infrastructure.

Now that we've got our root certificate authority up and running, we can create a certificate that the VPN server will use.

Step 3 — Generating a Certificate for the VPN Server

We'll now create a certificate and key for the VPN server. This certificate will allow the client to verify the server's authenticity using the CA certificate we just generated.

First, create a private key for the VPN server with the following command:

Now, create and sign the VPN server certificate with the certificate authority's key you created in the previous step. Execute the following command, but change the Common Name (CN) and the Subject Alternate Name (SAN) field to your VPN server's DNS name or IP address:

Now that we've generated all of the TLS/SSL files StrongSwan needs, we can move the files into place in the `/etc/ipsec.d` directory by typing:

In this step, we've created a certificate pair that would be used to secure communications between the client and the server. We've also signed the certificates with the CA key, so the client will be able to verify the authenticity of the VPN server using the CA certificate. Now that have all of the certificates ready, we'll move on to configuring the software.

Step 4 — Configuring StrongSwan

StrongSwan has a default configuration file with some examples, but we will have to do most of the configuration ourselves. Let's back up the file for reference before starting from scratch:

Create and open a new blank configuration file by typing:

First, we'll tell StrongSwan to log daemon statuses for debugging and allow duplicate connections. Add these lines to the file:

/etc/ipsec.conf

```
config setup
    charondebug="ike 1, knl 1, cfg 0"
    uniqueids=no
```

Then, we'll create a configuration section for our VPN. We'll also tell StrongSwan to create IKEv2 VPN Tunnels and to automatically load this configuration section when it starts up. Append the following lines to the file:

/etc/ipsec.conf

```
. . .
conn ikev2-vpn
    auto=add
    compress=no
    type=tunnel
    keyexchange=ikev2
    fragmentation=yes
    forceencaps=yes
```

We'll also configure dead-peer detection to clear any "dangling" connections in case the client unexpectedly disconnects. Add these lines:

/etc/ipsec.conf

```
. . .
conn ikev2-vpn
    . . .
    dpdaction=clear
    dpddelay=300s
    rekey=no
```

Then, we’ll configure the server (left) side IPSec parameters. Add this to the file:

/etc/ipsec.conf

```
. . .
conn ikev2-vpn
    . . .
    left=%any
    leftid=@server_domain_or_IP
    leftcert=server-cert.pem
    leftsendcert=always
    leftsubnet=0.0.0.0/0
```

Note: When configuring the server ID (`leftid`), only include the `@` character if your VPN server will be identified by a domain name:

```
    leftid=@vpn.example.com
```

If the server will be identified by its IP address, just put the IP address in:

```
    leftid=203.0.113.7
```

Next, we can configure the client (right) side IPSec parameters, like the private IP address ranges and DNS servers to use:

/etc/ipsec.conf

```
. . .
conn ikev2-vpn
    . . .
    right=%any
    rightid=%any
    rightauth=eap-mschapv2
    rightsourceip=10.10.10.0/24
    rightdns=8.8.8.8,8.8.4.4
    rightsendcert=never
```

Finally, we’ll tell StrongSwan to ask the client for user credentials when they connect:

/etc/ipsec.conf

```
. . .
conn ikev2-vpn
    . . .
    eap_identity=%identity
```

The configuration file should look like this:

/etc/ipsec.conf

```
config setup
    charondebug="ike 1, knl 1, cfg 0"
    uniqueids=no

conn ikev2-vpn
    auto=add
    compress=no
    type=tunnel
    keyexchange=ikev2
    fragmentation=yes
    forceencaps=yes
    dpdaction=clear
    dpddelay=300s
    rekey=no
    left=%any
    leftid=@server_domain_or_IP
    leftcert=server-cert.pem
    leftsendcert=always
    leftsubnet=0.0.0.0/0
    right=%any
    rightid=%any
    rightauth=eap-mschapv2
    rightsourceip=10.10.10.0/24
    rightdns=8.8.8.8,8.8.4.4
    rightsendcert=never
    eap_identity=%identity
```

Save and close the file once you've verified that you've configured things as shown.

Now that we've configured the VPN parameters, let's move on to creating an account so our users can connect to the server.

Step 5 — Configuring VPN Authentication

Our VPN server is now configured to accept client connections, but we don't have any credentials configured yet. We'll need to configure a couple things in a special configuration file called `ipsec.secrets`:

- We need to tell StrongSwan where to find the private key for our server certificate, so the server will be able to authenticate to clients.
- We also need to set up a list of users that will be allowed to connect to the VPN.

Let's open the secrets file for editing:

First, we'll tell StrongSwan where to find our private key:

/etc/ipsec.secrets

```
: RSA "server-key.pem"
```

Then, we'll define the user credentials. You can make up any username or password combination that you like:

/etc/ipsec.secrets

```
your_username : EAP "your_password"
```

Save and close the file. Now that we've finished working with the VPN parameters, we'll restart the VPN service so that our configuration is applied:

Now that the VPN server has been fully configured with both server options and user credentials, it's time to move on to configuring the most important part: the firewall.

Step 6 — Configuring the Firewall & Kernel IP Forwarding

With the StrongSwan configuration complete, we need to configure the firewall to forward and allow VPN traffic through.

If you followed the prerequisite tutorial, you should have a very basic UFW firewall enabled. If you don't yet have UFW configured, you can create a baseline configuration and enable it by typing:

Now, add a rule to allow UDP traffic to the standard IPSec ports, 500 and 4500:

Next, we will open up one of UFW's configuration files to add a few low-level policies for routing and forwarding IPSec packets. Before we do, we need to find which network interface on our server is used for internet access. We can find that by querying for the interface associated with the default route:

Your public interface should follow the word "dev". For example, this result shows the interface named `eth0`, which is highlighted below:

```
Output

default via 203.0.113.7 dev eth0 proto static
```

When you have your public network interface, open the `/etc/ufw/before.rules` file in your text editor:

Near the top of the file (before the `*filter` line), add the following configuration block:

/etc/ufw/before.rules

```
*nat
-A POSTROUTING -s 10.10.10.0/24 -o eth0 -m policy --pol ipsec --dir out -j 
ACCEPT
-A POSTROUTING -s 10.10.10.0/24 -o eth0 -j MASQUERADE
COMMIT

*mangle
-A FORWARD --match policy --pol ipsec --dir in -s 10.10.10.0/24 -o eth0 -p 
tcp -m tcp --tcp-flags SYN,RST SYN -m tcpmss --mss 1361:1536 -j TCPMSS --set-
mss 1360
COMMIT

*filter
:ufw-before-input - [0:0]
:ufw-before-output - [0:0]
:ufw-before-forward - [0:0]
```

```
:ufw-not-local - [0:0]
. . .
```

Change each instance of `eth0` in the above configuration to match the interface name you found with `ip route`. The `*nat` lines create rules so that the firewall can correctly route and manipulate traffic between the VPN clients and the internet. The `*mangle` line adjusts the maximum packet segment size to prevent potential issues with certain VPN clients.

Next, after the `*filter` and chain definition lines, add one more block of configuration:

/etc/ufw/before.rules

```
. . .
*filter
:ufw-before-input - [0:0]
:ufw-before-output - [0:0]
:ufw-before-forward - [0:0]
:ufw-not-local - [0:0]

-A ufw-before-forward --match policy --pol ipsec --dir in --proto esp -s 
10.10.10.0/24 -j ACCEPT
-A ufw-before-forward --match policy --pol ipsec --dir out --proto esp -d 
10.10.10.0/24 -j ACCEPT
```

These lines tell the firewall to forward ESP (Encapsulating Security Payload) traffic so the VPN clients will be able to connect. ESP provides additional security for our VPN packets as they're traversing untrusted networks.

When you're finished, save and close the file.

Before we restart the firewall, we'll change some network kernel parameters to allow routing from one interface to another. Open UFW's kernel parameters configuration file:

We'll need to configure a few things here:

- First, we'll enable IPv4 packet forwarding.
- We'll disable Path MTU discovery to prevent packet fragmentation problems.
- We also won't accept ICMP redirects nor send ICMP redirects to prevent man-in-the-middle attacks.

The changes you need to make to the file are highlighted in the following code:

/etc/ufw/sysctl.conf

```
. . .

# Enable forwarding
# Uncomment the following line
net/ipv4/ip_forward=1

. . .

# Do not accept ICMP redirects (prevent MITM attacks)
# Ensure the following line is set
net/ipv4/conf/all/accept_redirects=0
```

```
# Do not send ICMP redirects (we are not a router)
# Add the following lines
net/ipv4/conf/all/send_redirects=0
net/ipv4/ip_no_pmtu_disc=1
```

Save the file when you are finished. UFW will apply these changes the next time it starts.

Now, we can enable all of our changes by disabling and re-enabling the firewall:

You'll be prompted to confirm the process. Type Y to enable UFW again with the new settings.

Step 7 – Testing the VPN Connection on Windows, iOS, and macOS

Now that you have everything set up, it's time to try it out. First, you'll need to copy the CA certificate you created and install it on your client device(s) that will connect to the VPN. The easiest way to do this is to log into your server and output the contents of the certificate file:

You'll see output similar to this:

```
Output

-----BEGIN CERTIFICATE-----
MIIFQjCCAyqgAwIBAgIIFkQGvkH4ej0wDQYJKoZIhvcNAQEMBQAwPzELMAkGA1UE

. . .

EwbVLOXcNduWK2TPbk/+82GRMtjftran6hKbpKGghBVDPVFGFT6Z0OfubpkQ9RsQ
BayqOb/Q
-----END CERTIFICATE-----
```

Copy this output to your computer, including the `-----BEGIN CERTIFICATE-----` and `-----END CERTIFICATE-----` lines, and save it to a file with a recognizable name, such as `ca-cert.pem`. Ensure the file you create has the `.pem` extension.

Alternatively, use SFTP to transfer the file to your computer.

Once you have the `ca-cert.pem` file downloaded to your computer, you can set up the connection to the VPN.

Connecting from Windows

First, import the root certificate by following these steps:

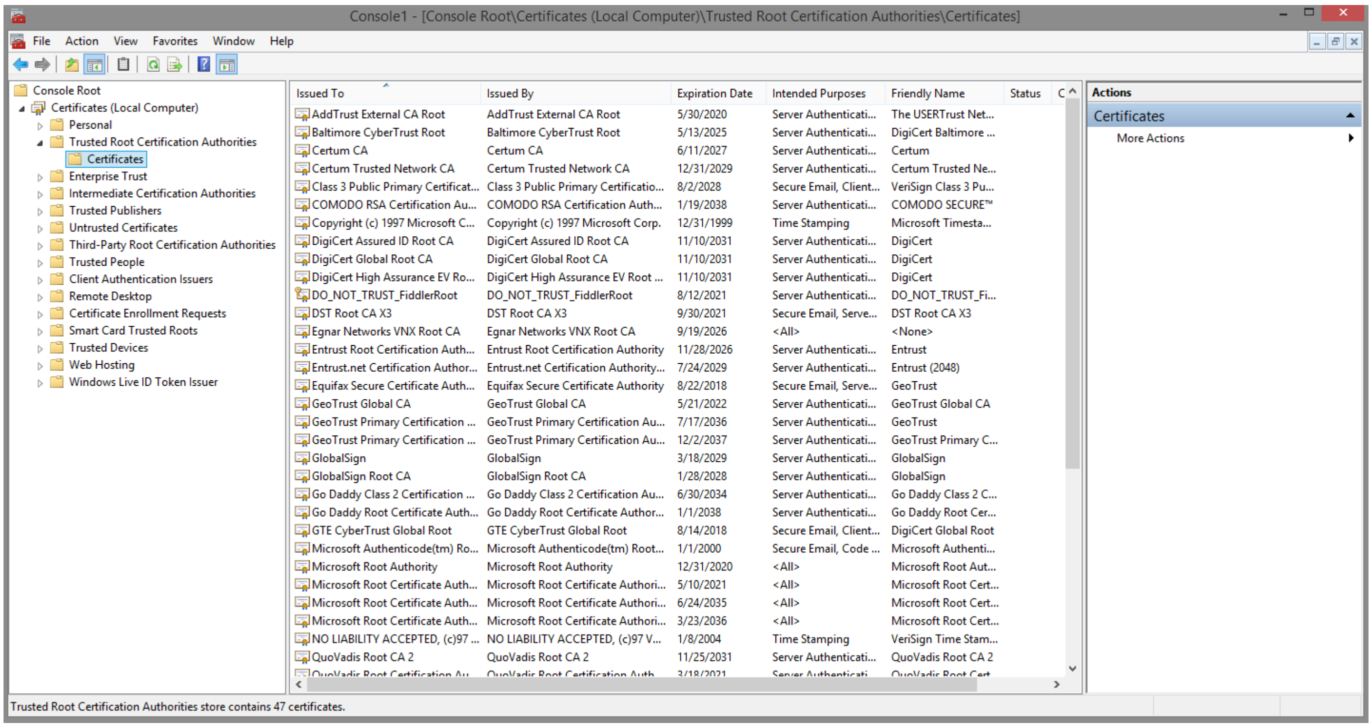
1. Press `WINDOWS+R` to bring up the **Run** dialog, and enter `mmc.exe` to launch the Windows Management Console.

2. From the **File** menu, navigate to **Add or Remove Snap-in**, select **Certificates** from the list of available snap-ins, and click **Add**.

3. We want the VPN to work with any user, so select **Computer Account** and click **Next**.

4. We're configuring things on the local computer, so select **Local Computer**, then click **Finish**.

5. Under the **Console Root** node, expand the **Certificates (Local Computer)** entry, expand **Trusted Root Certification Authorities**, and then select the **Certificates** entry:


Console1 - [Console Root\Certificates (Local Computer)\Trusted Root Certification Authorities\Certificates]
File
Action
View
Favorites
Window
Help
Console Root
Certificates (Local Computer)
Personal
Trusted Root Certification Authorities
Certificates
Enterprise Trust
Intermediate Certification Authorities
Trusted Publishers
Untrusted Certificates
Third-Party Root Certification Authorities
Trusted People
Client Authentication Issuers
Remote Desktop
Certificate Enrollment Requests
Smart Card Trusted Roots
Trusted Devices
Web Hosting
Windows Live ID Token Issuer
Actions
Certificates
More Actions

Issued To	Issued By	Expiration Date	Intended Purposes	Friendly Name	Status	C
AddTrust External CA Root	AddTrust External CA Root	5/30/2020	Server Authenticati...	The USERTrust Net...		
Baltimore CyberTrust Root	Baltimore CyberTrust Root	5/13/2025	Server Authenticati...	DigiCert Baltimore ...		
Certum CA	Certum CA	6/11/2027	Server Authenticati...	Certum		
Certum Trusted Network CA	Certum Trusted Network CA	12/31/2029	Server Authenticati...	Certum Trusted Ne...		
Class 3 Public Primary Certificat...	Class 3 Public Primary Certificatio...	8/2/2028	Secure Email, Client...	VeriSign Class 3 Pu...		
COMODO RSA Certification Au...	COMODO RSA Certification Auth...	1/19/2038	Server Authenticati...	COMODO SECURE™		
Copyright (c) 1997 Microsoft C...	Copyright (c) 1997 Microsoft Corp.	12/31/1999	Time Stamping	Microsoft Timesta...		
DigiCert Assured ID Root CA	DigiCert Assured ID Root CA	11/10/2031	Server Authenticati...	DigiCert		
DigiCert Global Root CA	DigiCert Global Root CA	11/10/2031	Server Authenticati...	DigiCert		
DigiCert High Assurance EV Ro...	DigiCert High Assurance EV Root ...	11/10/2031	Server Authenticati...	DigiCert		
DO_NOT_TRUST_FiddlerRoot	DO_NOT_TRUST_FiddlerRoot	8/12/2021	Server Authenticati...	DO_NOT_TRUST_Fi...		
DST Root CA X3	DST Root CA X3	9/30/2021	Secure Email, Serve...	DST Root CA X3		
Egnar Networks VNX Root CA	Egnar Networks VNX Root CA	9/19/2026	<All>	<None>		
Entrust Root Certification Auth...	Entrust Root Certification Authority	11/28/2026	Server Authenticati...	Entrust		
Entrust.net Certification Author...	Entrust.net Certification Authority...	7/24/2029	Server Authenticati...	Entrust (2048)		
Equifax Secure Certificate Auth...	Equifax Secure Certificate Authority	8/22/2018	Secure Email, Serve...	GeoTrust		
GeoTrust Global CA	GeoTrust Global CA	5/21/2022	Server Authenticati...	GeoTrust Global CA		
GeoTrust Primary Certification ...	GeoTrust Primary Certification Au...	7/17/2036	Server Authenticati...	GeoTrust		
GeoTrust Primary Certification ...	GeoTrust Primary Certification Au...	12/2/2037	Server Authenticati...	GeoTrust Primary C...		
GlobalSign	GlobalSign	3/18/2029	Server Authenticati...	GlobalSign		
GlobalSign Root CA	GlobalSign Root CA	1/28/2028	Server Authenticati...	GlobalSign		
Go Daddy Class 2 Certification ...	Go Daddy Class 2 Certification Au...	6/30/2034	Server Authenticati...	Go Daddy Class 2 C...		
Go Daddy Root Certificate Auth...	Go Daddy Root Certificate Author...	1/1/2038	Server Authenticati...	Go Daddy Root Cer...		
GTE CyberTrust Global Root	GTE CyberTrust Global Root	8/14/2018	Secure Email, Client...	DigiCert Global Root		
Microsoft Authenticode(tm) Ro...	Microsoft Authenticode(tm) Root...	1/1/2000	Secure Email, Code ...	Microsoft Authenti...		
Microsoft Root Authority	Microsoft Root Authority	12/31/2020	<All>	Microsoft Root Aut...		
Microsoft Root Certificate Auth...	Microsoft Root Certificate Authori...	5/10/2021	<All>	Microsoft Root Cert...		
Microsoft Root Certificate Auth...	Microsoft Root Certificate Authori...	6/24/2035	<All>	Microsoft Root Cert...		
Microsoft Root Certificate Auth...	Microsoft Root Certificate Authori...	3/23/2036	<All>	Microsoft Root Cert...		
NO LIABILITY ACCEPTED, (c)97 ...	NO LIABILITY ACCEPTED, (c)97 V...	1/8/2004	Time Stamping	VeriSign Time Stam...		
QuoVadis Root CA 2	QuoVadis Root CA 2	11/25/2031	Server Authenticati...	QuoVadis Root CA 2		
QuoVadis Root Certification Au...	QuoVadis Root Certification Auth...	3/18/2021	Server Authenticati...	QuoVadis Root Cert...		

Trusted Root Certification Authorities store contains 47 certificates.

6. From the **Action** menu, select **All Tasks** and click **Import** to display the Certificate Import Wizard. Click **Next** to move past the introduction.

7. On the **File to Import** screen, press the **Browse** button and select the certificate file that you've saved. Then click **Next**.

8. Ensure that the **Certificate Store** is set to **Trusted Root Certification Authorities**, and click **Next**.

9. Click **Finish** to import the certificate.

Then configure the VPN with these steps:

1. Launch **Control Panel**, then navigate to the **Network and Sharing Center**.
2. Click on **Set up a new connection or network**, then select **Connect to a workplace**.
3. Select **Use my Internet connection (VPN)**.
4. Enter the VPN server details. Enter the server's domain name or IP address in the **Internet address** field, then fill in **Destination name** with something that describes your VPN connection. Then click **Done**.

Your new VPN connection will be visible under the list of networks. Select the VPN and click **Connect**. You'll be prompted for your username and password. Type them in, click **OK**, and you'll be connected.

Connecting from macOS

Follow these steps to import the certificate:

1. Double-click the certificate file. **Keychain Access** will pop up with a dialog that says "Keychain Access is trying to modify the system keychain. Enter your password to allow this."
2. Enter your password, then click on **Modify Keychain**
3. Double-click the newly imported VPN certificate. This brings up a small properties window where you can specify the trust levels. Set **IP Security (IPSec)** to **Always Trust** and you'll be prompted for your password again. This setting saves automatically after entering the password.

Now that the certificate is important and trusted, configure the VPN connection with these steps:

1. Go to **System Preferences** and choose **Network**.
2. Click on the small "plus" button on the lower-left of the list of networks.
3. In the popup that appears, Set **Interface** to **VPN**, set the **VPN Type** to **IKEv2**, and give the connection a name.
4. In the **Server** and **Remote ID** field, enter the server's domain name or IP address. Leave the **Local ID** blank.

5. Click on **Authentication Settings**, select **Username**, and enter your username and password you configured for your VPN user. Then click **OK**.

Finally, click on **Connect** to connect to the VPN. You should now be connected to the VPN.

Connecting from Ubuntu

To connect from an Ubuntu machine, you can set up and manage StrongSwan as a service or use a one-off command every time you wish to connect. Instructions are provided for both.

Managing StrongSwan as a Service

1. Update your local package cache: `sudo apt update`
2. Install StrongSwan and the related software `sudo apt install strongswan libcharon-extra-plugins`
3. Copy the CA certificate to the `/etc/ipsec.d/cacerts` directory: `sudo cp /tmp/ca-cert.pem /etc/ipsec.d/cacerts`
4. Disable StrongSwan so that the VPN doesn't start automatically: `sudo systemctl disable --now strongswan`
5. Configure your VPN username and password in the `/etc/ipsec.secrets` file: `your_username : EAP "your_password"`
6. Edit the `/etc/ipsec.conf` file to define your configuration.

/etc/ipsec.conf

```
config setup

conn ikev2-rw
    right=server_domain_or_IP
    # This should match the `leftid` value on your server's configuration
    rightid=server_domain_or_IP
    rightsubnet=0.0.0.0/0
    rightauth=pubkey
    leftsourceip=%config
    leftid=username
    leftauth=eap-mschapv2
    eap_identity=%identity
    auto=start
```

To connect to the VPN, type:

To disconnect again, type:

Using a Simple Client for One-Off Connections

1. Update your local package cache: `sudo apt update`
2. Install `charon-cmd` and related software `sudo apt install charon-cmd libcharon-extra-plugins`
3. Move to the directory where you copied the CA certificate: `cd <^>/path/to/ca-cert.pem`
4. Connect to the VPN server with `charon-cmd` using the server's CA certificate, the VPN server's IP address, and the username you configured: `sudo charon-cmd --cert ca-cert.pem --host vpn_domain_or_IP --identity your_username`
5. When prompted, provide the VPN user's password.

You should now be connected to the VPN. To disconnect, press `CTRL+C` and wait for the connection to close.

Connecting from iOS

To configure the VPN connection on an iOS device, follow these steps:

1. Send yourself an email with the root certificate attached.
2. Open the email on your iOS device and tap on the attached certificate file, then tap **Install** and enter your passcode. Once it installs, tap **Done**.
3. Go to **Settings**, **General**, **VPN** and tap **Add VPN Configuration**. This will bring up the VPN connection configuration screen.
4. Tap on **Type** and select **IKEv2**.
5. In the **Description** field, enter a short name for the VPN connection. This could be anything you like.
6. In the **Server** and **Remote ID** field, enter the server's domain name or IP address. The **Local ID** field can be left blank.
7. Enter your username and password in the **Authentication** section, then tap **Done**.
8. Select the VPN connection that you just created, tap the switch on the top of the page, and you'll be connected.

Connecting from Android

Follow these steps to import the certificate:

1. Send yourself an email with the CA certificate attached. Save the CA certificate to your downloads folder.
2. Download the StrongSwan VPN client from the Play Store.
3. Open the app. Tap the **"more" icon** in the upper-right corner (the three dots icon) and select **CA certificates**.
4. Tap the **"more" icon** in the upper-right corner again. Select **Import certificate**.
5. Browse to the CA certificate file in your downloads folder and select it to import it into the app.

Now that the certificate is imported into the StrongSwan app, you can configure the VPN connection with these steps:

1. In the app, tap **ADD VPN PROFILE** at the top.
2. Fill out the **Server** with your VPN server's domain name or public IP address.
3. Make sure **IKEv2 EAP (Username/Password)** is selected as the VPN Type.
4. Fill out the **Username** and **Password** with the credentials you defined on the server.
5. Deselect **Select automatically** in the **CA certificate** section and click **Select CA certificate**.
6. Tap the **IMPORTED** tab at the top of the screen and choose the CA you imported (it will be named "VPN root CA" if you didn't change the "DN" earlier).
7. If you'd like, fill out **Profile name (optional)** with a more descriptive name.

When you wish to connect to the VPN, click on profile you just created in the StrongSwan application.

Troubleshooting Connections

If you are unable to import the certificate, ensure the file has the `.pem` extension, and not `.pem.txt`.

If you're unable to connect to the VPN, check the server name or IP address you used. The server's domain name or IP address must match what you've configured as the common name (CN) while creating the certificate. If they don't match, the VPN connection won't work. If you set up a certificate with the CN of `vpn.example.com`, you *must* use `vpn.example.com` when you enter the VPN server details. Double-check the command you used to generate the certificate, and the values you used when creating your VPN connection.

Finally, double-check the VPN configuration to ensure the `leftid` value is configured with the `@` symbol if you're using a domain name:

```
leftid=@vpn.example.com
```

And if you’re using an IP address, ensure that the @ symbol is omitted.

Conclusion

In this tutorial, you’ve built a VPN server that uses the IKEv2 protocol. Now you can be assured that your online activities will remain secure wherever you go!

To add or remove users, just take a look at Step 5 again. Each line is for one user, so adding or removing users is as simple as editing the file.

From here, you might want to look into setting up a log file analyzer, because StrongSwan dumps its logs into syslog. The tutorial How To Install and Use Logwatch Log Analyzer and Reporter on a VPS has more information on setting that up.

You might also be interested in this guide from the EFF about online privacy.

Previous Chapter Next Chapter