**Intrigrate SSl In Digitalocean Apache Hosted Website**

How To Install an SSL Certificate from a Commercial Certificate Authority

PostedNovember 25, 2014 652.9kviews [SECURITY](https://www.digitalocean.com/community/tags/security?type=tutorials) [APACHE](https://www.digitalocean.com/community/tags/apache?type=tutorials) [NGINX](https://www.digitalocean.com/community/tags/nginx?type=tutorials)

**Introduction**

This tutorial will show you how to acquire and install an SSL certificate from a trusted, commercial Certificate Authority (CA). SSL certificates allow web servers to encrypt their traffic, and also offer a mechanism to validate server identities to their visitors. The main benefit of using a purchased SSL certificate from a trusted CA, over self-signed certificates, is that your site's visitors will not be presented with a scary warning about not being able to verify your site's identity.

This tutorial covers how to acquire an SSL certificate from the following trusted certificate authorities:

* GoDaddy
* RapidSSL (via Namecheap)

You may also use any other CA of your choice.

After you have acquired your SSL certificate, we will show you how to install it on Nginx and Apache HTTP web servers.

Prerequisites

There are several prerequisites that you should ensure before attempting to obtain an SSL certificate from a commercial CA. This section will cover what you will need in order to be issued an SSL certificate from most CAs.

**Money**

SSL certificates that are issued from commercial CAs have to be purchased. The best free alternative are certificates issued from [Let's Encrypt](https://www.digitalocean.com/community/tutorials/how-to-secure-nginx-with-let-s-encrypt-on-ubuntu-14-04). Let's Encrypt is a new certificate authority that issues free SSL/TLS certificates that are trusted in most web browsers.

**Registered Domain Name**

Before acquiring an SSL certificate, you must own or control the registered domain name that you wish to use the certificate with. If you do not already have a registered domain name, you may register one with one of the many domain name registrars out there (e.g. Namecheap, GoDaddy, etc.).

**Domain Validation Rights**

For the basic domain validation process, you must have access to one of the email addresses on your domain's WHOIS record or to an "admin type" email address at the domain itself. Certificate authorities that issue SSL certificates will typically validate domain control by sending a validation email to one of the addresses on the domain's WHOIS record, or to a generic admin email address at the domain itself. Some CAs provide alternative domain validation methods, such as DNS- or HTTP-based validation, which are outside the scope of this guide.

If you wish to be issued an Organization Validation (OV) or Extended Validation (EV) SSL certificate, you will also be required to provide the CA with paperwork to establish the legal identity of the website's owner, among other things.

**Web Server**

In addition to the previously mentioned points, you will need a web server to install the SSL certificate on. This is the server that is reachable at the domain name for which the SSL certificate will be issued for. Typically, this will be an Apache HTTP, Nginx, HAProxy, or Varnish server. If you need help setting up a web server that is accessible via your registered domain name, follow these steps:

1. Set up a web server of your choice. For example, a [LEMP (Nginx)](https://www.digitalocean.com/community/tutorials/how-to-install-linux-nginx-mysql-php-lemp-stack-on-ubuntu-14-04) or [LAMP (Apache)](https://www.digitalocean.com/community/tutorials/how-to-install-linux-apache-mysql-php-lamp-stack-on-ubuntu-14-04) server--be sure to configure the web server software to use the name of your registered domain
2. Configure your domain to use the appropriate nameservers. If your web server is hosted on DigitalOcean, this guide can help you get set up: [How To Point to DigitalOcean's Nameservers from Common Domain Registrars](https://www.digitalocean.com/community/tutorials/how-to-point-to-digitalocean-nameservers-from-common-domain-registrars)
3. Add DNS records for your web server to your nameservers. If you are using DigitalOcean's nameservers, follow this guide to learn how to add the appropriate records: [How To Set Up a Host Name with DigitalOcean](https://www.digitalocean.com/community/tutorials/how-to-set-up-a-host-name-with-digitalocean)

Choose Your Certificate Authority

If you are not sure of which Certificate Authority you are going to use, there are a few important factors to consider. At an overview level, the most important thing is that the CA you choose provides the features you want at a price that you are comfortable with. This section will focus more on the features that most SSL certificate buyers should be aware of, rather than prices.

**Root Certificate Program Memberships**

The most crucial point is that the CA that you choose is a member of the root certificate programs of the most commonly used operating systems and web browsers, i.e. it is a "trusted" CA, and its root certificate is trusted by common browsers and other software. If your website's SSL certificate is signed by a trusted" CA, its identity is considered to be valid by software that trusts the CA--this is in contrast to self-signed SSL certificates, which also provide encryption capabilities but are accompanied by identity validation warnings that are off-putting to most website visitors.

Most commercial CAs that you will encounter will be members of the common root CA programs, and will say they are compatible with 99% of browsers, but it does not hurt to check before making your certificate purchase. For example, Apple provides its list of trusted SSL root certificates for iOS8 [here](http://support.apple.com/en-us/HT5012).

**Certificate Types**

Ensure that you choose a CA that offers the certificate type that you require. Many CAs offer variations of these certificate types under a variety of, often confusing, names and pricing structures. Here is a short description of each type:

* **Single Domain**: Used for a single domain, e.g. example.com. Note that additional subdomains, such as www.example.com, are not included
* **Wildcard**: Used for a domain and any of its subdomains. For example, a wildcard certificate for \*.example.com can also be used for www.example.com and store.example.com
* **Multiple Domain**: Known as a SAN or UC certificate, these can be used with multiple domains and subdomains that are added to the Subject Alternative Name field. For example, a single multi-domain certificate could be used with example.com, www.example.com, and example.net

In addition to the aforementioned certificate types, there are different levels of validations that CAs offer. We will cover them here:

* **Domain Validation** (DV): DV certificates are issued after the CA validates that the requestor owns or controls the domain in question
* **Organization Validation (OV)**: OV certificates can be issued only after the issuing CA validates the legal identity of the requestor
* **Extended Validation (EV)**: EV certificates can be issued only after the issuing CA validates the legal identity, among other things, of the requestor, according to a strict set of guidelines. The purpose of this type of certificate is to provide additional assurance of the legitimacy of your organization's identity to your site's visitors. EV certificates can be single or multiple domain, but not wildcard

This guide will show you how to obtain a single domain or wildcard SSL certificate from GoDaddy and RapidSSL, but obtaining the other types of certificates is very similar.

**Additional Features**

Many CAs offer a large variety of "bonus" features to differentiate themselves from the rest of the SSL certificate-issuing vendors. Some of these features can end up saving you money, so it is important that you weigh your needs against the offerings carefully before making a purchase. Example of features to look out for include free certificate reissues or a single domain-priced certificate that works for www. and the domain basename, e.g. www.example.com with a SAN of example.com

Generate a CSR and Private Key

After you have all of your prerequisites sorted out, and you know the type of certificate you want to get, it's time to generate a certificate signing request (CSR) and private key.

If you are planning on using Apache HTTP or Nginx as your web server, use openssl to generate your private key and CSR on your web server. In this tutorial, we will just keep all of the relevant files in our home directory but feel free to store them in any secure location on your server:

cd ~

To generate a private key, called example.com.key, and a CSR, called example.com.csr, run this command (replace the example.com with the name of your domain):

openssl req -newkey rsa:2048 -nodes -keyout example.com.key -out example.com.csr

At this point, you will be prompted for several lines of information that will be included in your certificate request. The most important part is the **Common Name** field which should match the name that you want to use your certificate with--for example, example.com, www.example.com, or (for a wildcard certificate request) \*.example.com. If you are planning on getting an OV or EV certificate, ensure that all of the other fields accurately reflect your organization or business details.

For example:

Country Name (2 letter code) [AU]:US

State or Province Name (full name) [Some-State]:New York

Locality Name (eg, city) []:New York

Organization Name (eg, company) [Internet Widgits Pty Ltd]:My Company

Organizational Unit Name (eg, section) []:

Common Name (e.g. server FQDN or YOUR name) []:example.com

Email Address []:sammy@example.com

This will generate a .key and .csr file. The .key file is your private key, and should be kept secure. The .csr file is what you will send to the CA to request your SSL certificate.

You will need to copy and paste your CSR when submitting your certificate request to your CA. To print the contents of your CSR, use this command (replace the filename with your own):

cat example.com.csr

Now we are ready to buy a certificate from a CA. We will show two examples, GoDaddy and RapidSSL via Namecheap, but feel free to get a certificate from any other vendor.

Example CA 1: RapidSSL via Namecheap

Namecheap provides a way to buy SSL certificates from a variety of CAs. We will walk through the process of acquiring a single domain certificate from RapidSSL, but you can deviate if you want a different type of certificate.

Note: If you request a single domain certificate from RapidSSL for the www subdomain of your domain (e.g. www.example.com), they will issue the certificate with a SAN of your base domain. For example, if your certificate request is for www.example.com, the resulting certificate will work for both www.example.comand example.com.

**Select and Purchase Certificate**

Go to Namecheap's SSL certificate page: <https://www.namecheap.com/security/ssl-certificates.aspx>.

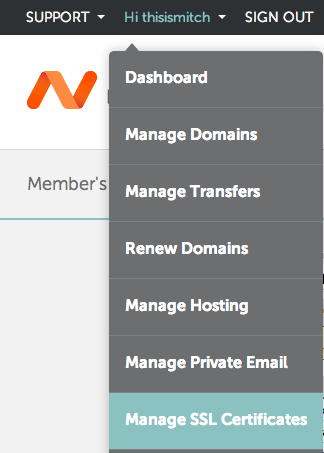
Here you can start selecting your validation level, certificate type ("Domains Secured"), or CA ("Brand").

For our example, we will click on the **Compare Products** button in the "Domain Validation" box. Then we will find "RapidSSL", and click the **Add to Cart** button.

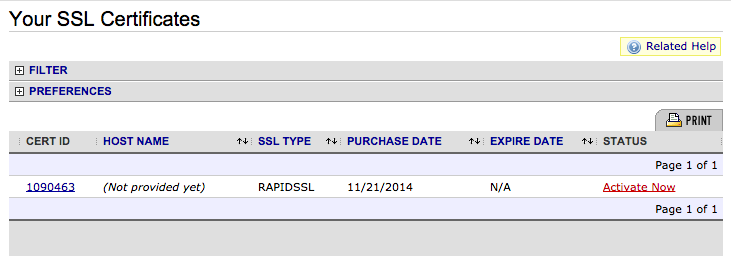
At this point, you must register or log in to Namecheap. Then finish the payment process.

**Request Certificate**

After paying for the certificate of your choice, go to the **Manage SSL Certificates** link, under the "Hi Username" section.



Here, you will see a list of all of the SSL certificates that you have purchased through Namecheap. Click on the **Activate Now** link for the certificate that you want to use.



Now select the software of your web server. This will determine the format of the certificate that Namecheap will deliver to you. Commonly selected options are "Apache + MOD SSL", "nginx", or "Tomcat".

Paste your CSR into the box then click the **Next** button.

You should now be at the "Select Approver" step in the process, which will send a validation request email to an address in your domain's WHOIS record or to an *administrator* type address of the domain that you are getting a certificate for. Select the address that you want to send the validation email to.

Provide the "Administrative Contact Information". Click the **Submit order** button.

**Validate Domain**

At this point, an email will be sent to the "approver" address. Open the email and approve the certificate request.

**Download Certificates**

After approving the certificate, the certificate will be emailed to the *Technical Contact*. The certificate issued for your domain and the CA's intermediate certificate will be at the bottom of the email.

Copy and save them to your server in the same location that you generated your private key and CSR. Name the certificate with the domain name and a .crt extension, e.g. example.com.crt, and name the intermediate certificate intermediate.crt.

The certificate is now ready to be installed on your web server.

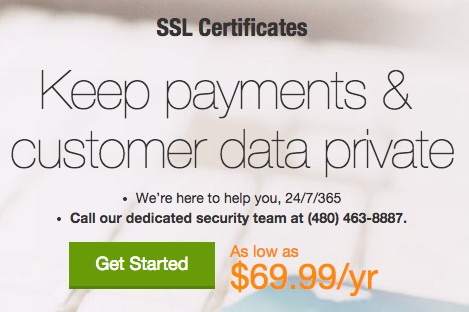
Example CA 2: GoDaddy

GoDaddy is a popular CA, and has all of the basic certificate types. We will walk through the process of acquiring a single domain certificate, but you can deviate if you want a different type of certificate.

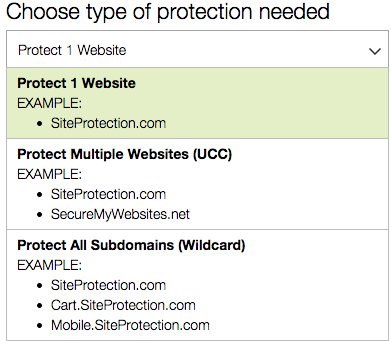
**Select and Purchase Certificate**

Go to GoDaddy's SSL certificate page: <https://www.godaddy.com/ssl/ssl-certificates.aspx>.

Scroll down and click on the **Get Started** button.



Select the type of SSL certificate that you want from the drop down menu: single domain, multidomain (UCC), or wildcard.



Then select your plan type: domain, organization, or extended validation.

Then select the term (duration of validity).

Then click the **Add to Cart** button.

Review your current order, then click the **Proceed to Checkout** button.

Complete the registration and payment process.

**Request Certificate**

After you complete your order, click the *SSL Certificates*\* button (or click on **My Account** > **Manage SSL Certificates** in the top-right corner).

Find the SSL certificate that you just purchased and click the **Set Up** button. If you have not used GoDaddy for SSL certificates before, you will be prompted to set up the "SSL Certificates" product, and associate your recent certificate order with the product (Click the green **Set Up** button and wait a few minutes before refreshing your browser).

After the "SSL Certificates" Product is added to your GoDaddy account, you should see your "New Certificate" and a "Launch" button. Click on the **Launch** button next to your new certificate.

Provide your CSR by pasting it into the box. The SHA-2 algorithm will be used by default.

Tick the **I agree** checkbox, and click the **Request Certificate** button.

**Validate Domain**

Now you will have to verify that you have control of the domain, and provide GoDaddy with a few documents. GoDaddy will send a domain ownership verification email to the address that is on your domain's WHOIS record. Follow the directions in the emails that you are sent to you, and authorize the issuance of the certificate.

**Download Certificate**

After verifying to GoDaddy that you control the domain, check your email (the one that you registered with GoDaddy with) for a message that says that your SSL certificate has been issued. Open it, and follow the download certificate link (or click the **Launch** button next to your SSL certificate in the GoDaddy control panel).

Now click the **Download** button.

Select the server software that you are using from the **Server type** dropdown menu--if you are using Apache HTTP or Nginx, select "Apache"--then click the **Download Zip File** button.

Extract the ZIP archive. It should contain two .crt files; your SSL certificate (which should have a random name) and the GoDaddy intermediate certificate bundle (gd\_bundle-g2-1.crt). Copy both two your web server. Rename the certificate to the domain name with a .crt extension, e.g. example.com.crt, and rename the intermediate certificate bundle as intermediate.crt.

The certificate is now ready to be installed on your web server.

Install Certificate On Web Server

After acquiring your certificate from the CA of your choice, you must install it on your web server. This involves adding a few SSL-related lines to your web server software configuration.

We will cover basic Nginx and Apache HTTP configurations on Ubuntu 14.04 in this section.

We will assume the following things:

* The private key, SSL certificate, and, if applicable, the CA's intermediate certificates are located in a home directory at /home/sammy
* The private key is called example.com.key
* The SSL certificate is called example.com.crt
* The CA intermediate certificate(s) are in a file called intermediate.crt
* If you have a firewall enabled, be sure that it allows port 443 (HTTPS)

**Note:** In a real environment, these files should be stored somewhere that only the user that runs the web server master process (usually root) can access. The private key should be kept secure.

**Nginx**

If you want to use your certificate with Nginx on Ubuntu 14.04, follow this section.

With Nginx, if your CA included an intermediate certificate, you must create a single "chained" certificate file that contains your certificate and the CA's intermediate certificates.

Change to the directory that contains your private key, certificate, and the CA intermediate certificates (in the intermediate.crt file). We will assume that they are in your home directory for the example:

cd ~

Assuming your certificate file is called example.com.crt, use this command to create a combined file called example.com.chained.crt (replace the highlighted part with your own domain):

cat example.com.crt intermediate.crt > example.com.chained.crt

Now go to your Nginx server block configuration directory. Assuming that is located at /etc/nginx/sites-enabled, use this command to change to it:

cd /etc/nginx/sites-enabled

Assuming want to add SSL to your default server block file, open the file for editing:

sudo vi default

Find and modify the listen directive, and modify it so it looks like this:

listen 443 ssl;

Then find the server\_name directive, and make sure that its value matches the common name of your certificate. Also, add the ssl\_certificate and ssl\_certificate\_key directives to specify the paths of your certificate and private key files (replace the highlighted part with the actual path of your files):

server\_name example.com;

ssl\_certificate /home/sammy/example.com.chained.crt;

ssl\_certificate\_key /home/sammy/example.com.key;

To allow only the most secure SSL protocols and ciphers, add the following lines to the file:

ssl\_protocols TLSv1 TLSv1.1 TLSv1.2;

ssl\_prefer\_server\_ciphers on;

ssl\_ciphers 'EECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+EDH';

If you want HTTP traffic to redirect to HTTPS, you can add this additional server block at the top of the file (replace the highlighted parts with your own information):

server {

listen 80;

server\_name example.com;

rewrite ^/(.\*) https://example.com/$1 permanent;

}

Then save and quit.

Now restart Nginx to load the new configuration and enable TLS/SSL over HTTPS!

sudo service nginx restart

Test it out by accessing your site via HTTPS, e.g. https://example.com.

**Apache**

If want to use your certificate with Apache on Ubuntu 14.04, follow this section.

Make a backup of your configuration file by copying it. Assuming your server is running on the default virtual host configuration file, /etc/apache2/sites-available/000-default.conf, use these commands to to make a copy:

cd /etc/apache2/sites-available

cp 000-default.conf 000-default.conf.orig

Then open the file for editing:

sudo vi 000-default.conf

Find the <VirtualHost \*:80> entry and modify it so your web server will listen on port 443:

<VirtualHost \*:443>

Then add the ServerName directive, if it doesn't already exist (substitute your domain name here):

ServerName example.com

Then add the following lines to specify your certificate and key paths (substitute your actual paths here):

SSLEngine on

SSLCertificateFile /home/sammy/example.com.crt

SSLCertificateKeyFile /home/sammy/example.com.key

If you are using Apache *2.4.8 or greater*, specify the CA intermediate bundle by adding this line (substitute the path):

SSLCACertificateFile /home/sammy/intermediate.crt

If you are using an older version of Apache, specify the CA intermediate bundle with this line (substitute the path):

SSLCertificateChainFile /home/sammy/intermediate.crt

At this point, your server is configured to listen on HTTPS only (port 443), so requests to HTTP (port 80) will not be served. To redirect HTTP requests to HTTPS, add the following to the top of the file (substitute the name in both places):

<VirtualHost \*:80>

ServerName example.com

Redirect permanent / https://example.com/

</VirtualHost>

Save and exit.

Enable the Apache SSL module by running this command:

sudo a2enmod ssl

Now restart Apache to load the new configuration and enable TLS/SSL over HTTPS!

sudo service apache2 restart

Test it out by accessing your site via HTTPS, e.g. https://example.com. You will also want to try connecting via HTTP, e.g. http://example.com to ensure that the redirect is working properly!