LetsEncrypt SSL cert on GoDaddy Shared Hosting with No Root and No nc

23 FEBRUARY 2017 on letsencrypt, security, godaddy, wtf, sharedhosting, acme.sh

> *$ acme.sh --issue -d MYDOMAIN.com -d www.MYDOMAIN.com -w ~/www --dns dns_gd*

Looks simple, doesn't it?

Nope. Here's what you have to do to get to that point.

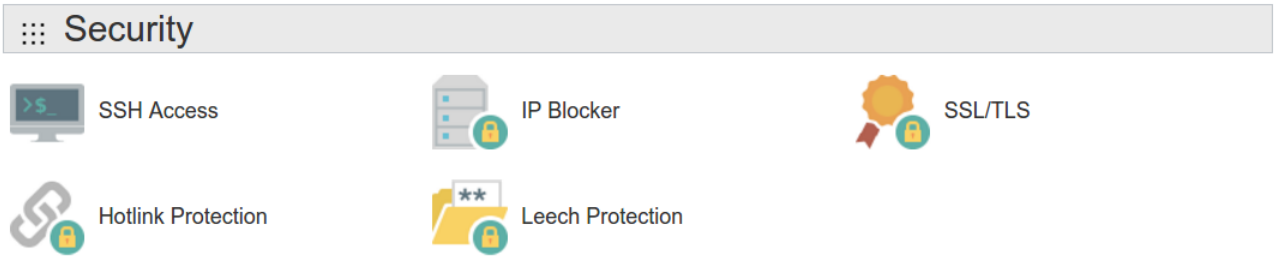
(The following Worked For Me™ on shared GoDaddy hosting, where the server was running Linux and Apache, but I don't think there's any Apache-specific commands, so if you're running Nginx + Linux, I think the following will work fine. If you're running Windows rather than Linux, the following tutorial will not work.)

Log into GoDaddy and give

yourself SSH access

Here's how... actually, since the GoDaddy docs are painful to read, here's a screenshot instead:

Go do your CPanel and scroll down till you find this section, then click "SSH Access" and then "Enable":



Then generate a new SSH key on your local machine at `~/.ssh/mydomain` add its public key to GoDaddy.

```
ssh-keygen -b 4096

/home/YOURUSERNAMEHERE/.ssh/mydomain
<ENTER> (empty passphrase)
<ENTER> (repeat empty passphrase)

echo
cat ~/.ssh/mydomain.pub
```

(Copy and paste that pub key to GoDaddy, *then authorize that user to connect via SSH*; adding a key is not enough, strangely.)

SSH in

```
ssh -i .ssh/mydomain
t$MY_CUSTOMER_NUMBER@$MYDOMAIN.com
```

Yes, that's right... even though GoDaddy says here that you can use "[y]our account's primary username" to log in, that doesn't work (I checked 3 times).

You need to use lower-case `t`, followed by your GoDaddy membership number (which you often log in with as your username), then `@`, then your domain.

Even though you're authenticating via SSH, you must also now type in a password at the command line. This is your normal account password.

(Oh, and if you get some sort of `ssh_exchange_identification: read: Connection reset by peer` error, turn off your VPN; GoDaddy doesn't like people SSHing in from those.)

Download and install acme.sh

acme.sh is a full implementation of a LetsEncrypt client but that doesn't depend on Python/pip/virtualenv/etc, and that doesn't require root -- exactly what we need, since we don't have root an a *shared* GoDaddy server, and we can't install new software outside of our home directory.

```
curl https://get.acme.sh | sh
```

Now log out and SSH back in so `acme.sh`'s install is complete in every way (include the Bash alias).

Get GoDaddy API Key

1. Visit https://developer.godaddy.com/keys/
2. Generate a production key (it made me produce a *test* key first for some reason...)
3. In the SSH session you have open run these commands, but with each `...` replaced by the value GoDaddy gave you in the previous step:

```
export GD_Secret=...
export GD_Key=...
```

Now it'll work!... but you're still not done

```
acme.sh --issue -d MYDOMAIN.com -d
www.MYDOMAIN.com -w ~/www --dns dns_gd
```

(Yes, literally `~/www`, no trailing `/`.)

(NOTE: If you're creating this cert for a domain that's not the default domain being hosted on this server, then instead of `~/www` you'll need to do something like

`~/www/MYOTHERDOMAIN.COM`.)

Boom! You should have just gotten your first good news of the day -- your cert, cert key, intermediate CA cert, and full cert chain have been generated!

...but GoDaddy still doesn't know that the cert exists, so it's not using it.

Upload cert and private key to GoDaddy via acme.sh --deploy

Turns out that CPanel, the web interface you use to manage your server (which you saw earlier when you gave yourself SSH access), has an API, and we can call it from our GoDaddy server!

1. Use `nano` to open `~/.acme.sh/deploy/cpanel_uapi.sh`, uncomment the `DEPLOY_CPANEL_USER` variable at the top, set its value to your user ID (*without* the `t` at the beginning)
2. Run `acme.sh --deploy -d MYDOMAIN.com --deploy-hook cpanel_uapi`

After ~30 seconds, you should see output like this!

```
[Sun Sep 17 03:17:45 MST 2017] Certificate
successfully deployed
[Sun Sep 17 03:17:45 MST 2017] Success
```

(BIG thanks to Santeri Kannisto for pointing me to `cpanel_uapi.sh` for fully automating the cert updating!)

So close!

To take stock: we've generated a LetsEncrypt cert for our domain, `acme.sh` installed a cronjob that'll auto-renew it(!) (run `crontab -l` if you don't believe me), and we've now given this cert to GoDaddy so it can use it to host a secure version of our site.

For supporting multiple domains, edit your cronjobs to be something like the following.

Run the command

```
EDITOR=nano VISUAL=nano crontab -e
```

to edit your cronjob definitions (in the `nano` text editor) to look like the following, except replace `MYDOMAIN.com` and the other domains with your own domains:

```
0 0 1 * * ~/.acme.sh/acme.sh --cron --home
~/.acme.sh --force 2>&1 >> ~/.acme.sh/cronlog.txt
1 0 1 * * ~/.acme.sh/acme.sh --deploy -d
MYDOMAIN.com        --deploy-hook cpanel_uapi
2 0 1 * * ~/.acme.sh/acme.sh --deploy -d
SOMEOTHERDOMAIN.org --deploy-hook cpanel_uapi
```

```
3 0 1 * * ~/.acme.sh/acme.sh --deploy -d
THIRDDOMAIN.com     --deploy-hook cpanel_uapi
```

To save these cronjobs and finish editing them, press `ctrl+o` to save, then `Enter` to confirm saving, then `ctrl+x` to exit `nano`.

So we're done!... right?

Almost.

GoDaddy doesn't automatically redirect from the HTTP version of your site to the HTTPS version. Here's how to do that:

SSH in, then run

```
nano www/.htaccess
```

Then, *right* after

```
# BEGIN WordPress
<IfModule mod_rewrite.c>
RewriteEngine On
```

add these 2 lines:

```
RewriteCond %{HTTPS} off
```

```
RewriteRule ^(.*)$ https://%{HTTP_HOST}%
{REQUEST_URI} [L,R=301]
```

(Here's GoDaddy's arguably more complete, but in my case misleading instructions on how to supposedly do this in a non-WordPress environment.)

...aaaand -- *finally* -- that's it!

(P.S. Don't use GoDaddy for anything unless you're helping a friend who's already locked in there like I was; use NameCheap instead.)

Steve Phillips / @elimisteve

Philosopher => Created Executable Philosophy, a new philosophical methodology. Programmer => Created CrypTag (makes encrypted data searchable). I love democratizing forces and revolutionary projects

San Francisco, CA
https://tryingtobeawesome.com

Share this post