Basic Things With Nginx

Christian Külker

2023-02-27

Contents

Nginx is pronounced `engine x`. It is a lightweight http and proxy server. "Why should I use a web server other than Apache2?" you may ask. Nginx has several advantages and disadvantages. The advantage of Nginx is that it is very fast for simple configured web pages, especially for only one domain. However, for large installations with complex processing, one can almost say middleware, Apache2 is the choice.

1 Installation

Under Debian Wheezy, Jessie, Stretch and Buster (and probably others):

```
aptitude install nginx
```

Nginx is already serving pages. One page, to be exact. A welcome page. Look at the URL:

```
http://127.0.0.1/
```

2 Configuration

The configuration under Debian is similar to Apache2 in terms of file location. First you create a file in `/etc/nginx/sites-available` and then you make a link to `/etc/nginx/sites-anabled`.

```
.
|-- conf.d
|-- fastcgi.conf
|-- fastcgi_params
|-- koi-utf
|-- koi-win
|-- mime.types
|-- nginx.conf
|-- proxy_params
|-- scgi_params
|-- sites-available
|   `-- default
|-- sites-enabled
|   `-- default -> /etc/nginx/sites-available/default
|-- snippets
|   |-- fastcgi-php.conf
|   `-- snakeoil.conf
|-- uwsgi_params
`-- win-utf
```

You can edit the default site, which is already enabled. Or you can just copy the content to `/var/www/html`. Of course, you will have to overwrite `index.html` to see your content, since that page is served first. An alternative would be to create a directory `/var/www/html/dragon` to serve your favorite pictures of dragons (or whatever).

In the latter case look at:

```
http://127.0.0.1/dragon/
```

2.1 Server Block

When using `nginx`, a server block is similar to a virtual host under Apache2.

Nginx on Debian 9 has a server block enabled by default, configured to serve documents from a directory in `/var/www/html`. While this works well for a single site, it can become difficult when hosting multiple sites. The domain in this example is called `D1`, but you should think of it as something like `example.com`.

```
mkdir -p /opt/www/domain/D1/html
vim /opt/www/domain/D1/html/index.html
vim /etc/nginx/sites-available/exmaple.com
```

```
server {
        listen 80;
        listen [::]:80;

        root /opt/www/domain/D1/html;
        index index.html index.htm index.nginx-debian.html;

        server_name D1;

        location / {
                try_files $uri $uri/ =404;
        }
}
```

```
ln -s /etc/nginx/sites-available/D1 \
/etc/nginx/sites-enabled/D1
nginx -t
systemctl restart nginx
```

After `certbot` run was successful this is converted to:

```
server {
        listen 80;
        listen [::]:80;

        root /opt/www/domain/D1/html;
        index index.html index.htm index.nginx-debian.html;

        server_name D1;

        location / {
                try_files $uri $uri/ =404;
        }

  listen [::]:443 ssl ipv6only=on; # managed by Certbot
  listen 443 ssl; # managed by Certbot
 ssl_certificate /etc/letsencrypt/live/D1/fullchain.pem; # managed by Certbot
 ssl_certificate_key /etc/letsencrypt/live/D1/privkey.pem; # managed by Certbot
 include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
 ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot

}
```

=>

3 Add +1 Site With Different Port?

Copy the following into the file `/etc/nginx/sites-available/wizzards`

```
server {
        listen 8080 default_server;
        listen [::]:8080 default_server;

        root /var/www/wizzards;
        index index.html;
        server_name _;
        location / {
                try_files $uri $uri/ =404;
        }
}
```

Then make a link and restart Nginx

```
cd /etc/nginx/sites-eanabled
ln -s /etc/nginx/sites-available/wizards .
service nginx restart
```

Of course you have to create the directory:

```
mkdir /var/www/wizards
```

And copy HTML content into it. Not wizards.

Now look at:

```
http://127.0.0.1:8080/
```

3.1 Disable TLSv1.0 TLSv1.1

When looking at

https://www.ssllabs.com/ssltest/analyze.html?d=www.DOMAIN.de

It seems that `TLSv1.0` and `TLSv1.1` are better to be disabled.

Check with:

```
nmap --script ssl-enum-ciphers -p 443 81.169.254.165|grep TLSv
|   TLSv1.0:
|   TLSv1.1:
|   TLSv1.2:
```

Change

```
 vim /etc/nginx/nginx.conf
```

```
#ssl_protocols TLSv1 TLSv1.1 TLSv1.2; # Dropping SSLv3, ref: POODLE
ssl_protocols TLSv1.2; # Dropping SSLv3, ref: POODLE
```

Restart Nginx

Check again with:

```
nmap --script ssl-enum-ciphers -p 443 81.169.254.165|grep TLSv
|   TLSv1.2:
```

HOWEVER if you use `certbot` then it will use its own TLS configuration: `/etc/letsencrypt/options-ssl-n` There seems to be no way to specify which TLS version to allow with `certbot`.

4 Reverse Proxy

This seems to be more common these days to build some kind of web application that delivers parts of the content over a specific port.

```
location /auth/ {
      proxy_buffers 16 4k;
      proxy_buffer_size 2k;
      proxy_bind 127.0.0.1;
      proxy_pass http://127.0.0.1:4000/auth/
}
```

5 I18n Index Page

This is convenient for some users. Other users who do not use their own browser or do not understand how to change the language of their browser are basically screwed with this setup. But nonetheless, this is how it is done. (That was the long way of saying: don't do it)

```
server {
       listen 127.0.0.1:80 default_server;

       server_name localhost;

       root /var/www/html;
       index index.html;

       set $first_language $http_accept_language;
       if ($http_accept_language ~* '^(.+?),') {
           set $first_language $1;
       }

       set $language 'en';
       if ($first_language ~* 'de') {
           set $language 'de';
       }

       location / {
              try_files $uri/index.$language.html $uri $uri/ =404;
       }
}
```

6 Multiple Domains With One Server Directive

The section title could also be "How to serve multiple virtual domains with Certbot and Nginx". Usually it is very easy to serve virtual domains with nginx. Either add a file with a "server" block or add a new "server" block to the `/etc/nginx/sites-available/default` file. However, in case a part of this file (default) is managed by Certbot, it is easier to manage all domains with a single 'server' block, as other parts are managed by Certbot. This is what a default configuration looks like. Comments are removed, indentation is changed from tab to two spaces and `$host` names (domains like `example.com`) are replaced by all caps: `D1` and `D2`.

```
server {
  listen 80 default_server;
  listen [::]:80 default_server;

  root /var/www/html;

  index index.html index.htm index.nginx-debian.html;

  server_name _;

  location / {
    try_files $uri $uri/ =404;
  }

}

server {

  root /var/www/html;

  index index.html index.htm index.nginx-debian.html;
    server_name www.D1 D1 www.D2; # managed by Certbot

  location / {
    try_files $uri $uri/ =404;
  }

  listen [::]:443 ssl ipv6only=on; # managed by Certbot
  listen 443 ssl; # managed by Certbot
  ssl_certificate /etc/letsencrypt/live/D1/fullchain.pem; # managed by Certbot
  ssl_certificate_key /etc/letsencrypt/live/D1/privkey.pem; # managed by Certbot
  include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
  ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot

}
server {
    if ($host = www.D1) {
        return 301 https://$host$request_uri;
    } # managed by Certbot

    if ($host = D1) {
        return 301 https://$host$request_uri;
    } # managed by Certbot

    if ($host = www.D2) {
        return 301 https://$host$request_uri;
    } # managed by Certbot

  listen 80 ;
  listen [::]:80 ;
    server_name www.D1 D1 www.D2;
    return 404; # managed by Certbot
}
```

The solution is quite simple. Add a line with the `$host` variable to the `https` location, like this `root /opt/www/domain/$host;` (and you may or may not remove the other `root` directive.

```
server {
  listen 80 default_server;
  listen [::]:80 default_server;

  root /var/www/html;

  index index.html index.htm index.nginx-debian.html;

  server_name _;

  location / {
    try_files $uri $uri/ =404;
  }
}

server {

  index index.html index.htm index.nginx-debian.html;
    server_name www.D1 D1 www.D2; # managed by Certbot

  location / {
    try_files $uri $uri/ =404;
    root /opt/wwww/domain/$host;
  }

  listen [::]:443 ssl ipv6only=on; # managed by Certbot
  listen 443 ssl; # managed by Certbot
 ssl_certificate /etc/letsencrypt/live/D1/fullchain.pem; # managed by Certbot
 ssl_certificate_key /etc/letsencrypt/live/D1/privkey.pem; # managed by Certbot
 include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
 ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot

}
server {
    if ($host = www.D1) {
        return 301 https://$host$request_uri;
    } # managed by Certbot

    if ($host = D1) {
        return 301 https://$host$request_uri;
    } # managed by Certbot

    if ($host = www.D2) {
        return 301 https://$host$request_uri;
    } # managed by Certbot

  listen 80 ;
  listen [::]:80 ;
    server_name www.D1 D1 www.D2;
    return 404; # managed by Certbot
}
```

7 Static HTML Mirror

To mirror static HTML files, in an ideal world Nginx would work out of the box. However, sometimes the pages to be served may come from a non-static page. In this case, the mirroring script (e.g. `wget`) may have written some files with a ’? For example

```
https://example.com/Page              - dynamic html page
https://example.com/Page?action=raw   - raw Markdown content
```

When writing down the static content this might lead to files like this:

```
mirror.com/example.com/Page/index.html    - a static HTML page
mirror.com/example.com/Page?action=raw    - a static conten file
```

With the usual configuration, Nginx would return a `404 Not Found` result when trying to get the URL `http://mirror.com/example.com/Page?action=raw`. With a modified try setup `$uri?$args` Nginx can at least serve this page to give the client a download option as this will be `application/octet-stream`.

```
server {
    listen 80;
    listen [::]:80;
    server_name localhost;
    root /opt/mirror.com/example.com;
    index index.html;
    location / {
        try_files $uri $uri/ $uri?$args =404;
    }
}
```

8 History

Version	Date	Notes
0.7	2023-02-27	Improve wording
0.6	2022-06-01	shell->bash, improve headings
0.5	2020-05-27	Serving static HTML mirror
0.4	2020-05-23	Certbot root with $host

Version	Date	Notes
0.3	2020-01-31	TLSv1 TLSv1.1
0.2	2017-01-27	
0.1	2016-06-19	Initial release

9 Disclaimer of Warranty

THERE IS NO WARRANTY FOR THIS INFORMATION, DOCUMENTS AND PROGRAMS, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE INFORMATION, DOCUMENT OR THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE INFORMATION, DOCUMENTS AND PROGRAMS IS WITH YOU. SHOULD THE INFORMATION, DOCUMENTS OR PROGRAMS PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

10 Limitation of Liability

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE INFORMATION, DOCUMENTS OR PROGRAMS AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE INFORMATION, DOCUMENTS OR PROGRAMS (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE INFORMATION, DOCUMENTS OR PROGRAMS TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.