第168回東京エリアDebian勉強会

debianにおけるnginxの設定例

Norimitsu Sugimoto (杉本 典充)
dictoss@live.jp

2018-11-17

自己紹介

- Norimitsu Sugimoto (杉本 典充)
- dictoss@live.jp
- Twitter: @dictoss
- Debian 使って 15 年以上。sarge が testing の頃から使っている
- 仕事はソフトウェア開発者をやってます
- python と Django の組み合わせで使うことが多いです

アジェンダ

- はじめに
- debian における web サーバのパッケージ
- nginx
- nginx の設定例
- まとめ・参考文献

debian におけるwebサーバのパッケージ

debian における web サーバのパッケージ

- web サーバの利用調査
  - `https://w3techs.com/technologies/comparison/ws-apache,ws-microsoftiis,ws-nginx`
- debian では apache、nginx のパッケージを用意されており、apt コマンドでインストール可能

nginx

nginx とは

- web サイト `https://nginx.org/`
- エンジンエックスと読みます
- ソースコードは、2-clause BSD-like license で公開
- 最初のリリースは 2004 年
- イベント駆動のシステムコールを利用（linux では epoll を利用）し、大量のクライアント（C10K）を捌く
- HTTP/HTTPS のファイル配信のみならず、HTTP/HTTPS リバースプロキシ、mail プロキシ、TCP/UDP プロキシとして動作

debian における nginx のインストール方法とディレクトリ構成

apt でインストール可能

```
# apt install nginx
```

拡張モジュールの数で deb パッケージを 3 つに分かれている

- nginx-full (=nginx)
- nginx-light
- nginx-extras

nginx の設定例

nginxパッケージのインストール直後の設定

一部省略して抜粋

```
$ tree /etc/nginx
   |--  conf.d/
   |--  fastcgi_params
   |--  modules-available/
   |--  modules-enable/
   |--  nginx.conf
   |--  proxy_params
   |--  site-available/
   |--  site-enabled/
   |--  snippets/
   |--  wsgi_params
```

nginxパッケージのインストール直後の設定

一部省略して抜粋

```
$ cat /etc/nginx/nginx.conf
user www-data;
worker_processes auto;
include /etc/nginx/modules-enabled/*.conf;
events {
  worker_connections 768;
}
http {
  access_log /var/log/nginx/access.log;
  error_log /var/log/nginx/error.log;
  gzip on;
  gzip_disable ‘‘msie6’’;
  include /etc/nginx/conf.d/*.conf;
  include /etc/nginx/sites-enabled/*;
}
```

VirtualHost 設定"default"

```
$ cat /etc/nginx/sites-available/default
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;
  }
}
```

SSL/TLS 設定

SSL/TLS 設定

- /etc/nginx/sites-available/ssl に SSL/TLS 待ち受け設定を書く
- /etc/nginx/sites-enabled/ssl にシンボリックリンクを生成し、VirtualHost を追加
- 【参考情報】Mozilla が公開している設定の生成ツール
  - `https://mozilla.github.io/server-side-tls/ssl-config-generator/`

SSL/TLS 設定

設定のポイント

- 中間証明書を必要とする場合は１つの crt ファイルに結合する
- ssl_protocols TLSv1.2; とだけ書くと、TLSv1.2 のみで接続を受け付ける
- `https://www.ssllabs.com/ssltest/` などでスキャンして診断
- ssl_dhparam は DH 1024bit より上げるか悩みどころ

SSL/TLS 設定

一部抜粋

```
$ cat /etc/nginx/sites-available/ssl
  server {
    listen 443 ssl http2;
    listen [::]:443 ssl http2;
    ssl on;
    ssl_certificate /etc/ssl/private/server.crt;
    ssl_certificate_key  /etc/ssl/private/server.key;
    ssl_protocols TLSv1.2; # Dropping SSLv3, ref: POODLE
    ssl_prefer_server_ciphers on; # ref: FREAK
    ssl_ciphers  HIGH:!aNULL:!MD5;
    gzip off;  # unuse gzip, ref: BREACH
    root /var/www/html;
    server_name www.example.com;
    location / {
      try_files $uri $uri/ =404;
    }
  }
```

SSL/TLS 設定：適用

```
$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful

$ sudo systemctl restart nginx
```

設定したサーバの”https://hostname/” へアクセスする

リバース proxy設定

リバース proxy

- リバース proxy
  - クライアントと直接接続するのは nginx が担う
  - クライアントへ応答する実際のデータ生成はバックエンドのサーバが行うように通信を中継する機能
- SSL/TLS 設定をするとき、リバース proxy の nginx のみに SSL 証明書を配置すればよい。バックエンドのサーバは HTTP 設定のみでよい。

リバース proxy：upstream の設定

- upstream 句にバックエンドの server の通信先を設定する
- 複数行の server 設定をすると、振り分けできる
- 重み付きラウンドロビン、leas_conn、ip_hash が利用可能

```
$ cat /etc/nginx/conf.d/upstream_proxy.conf
  upstream backend_app1 {
    # least_conn;
    # ip_hash;

    server 192.168.1.100:80 weight=1;
    server 192.168.1.101:80 weight=1;
  }
```

リバース proxy：VirtualHost 設定

- proxy 先への通信はデフォルトで HTTP/1.0 であり、"proxy_http_version"で変更可能

```
$ cat /etc/nginx/sites-available/default
server {
  # (snip)
  location ~ ^/proxy/(.*)$ {
    proxy_pass http://backend_proxy1;

    # proxy_http_version 1.1;

    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-Proto $scheme;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header Host $http_host;
    #proxy_redirect http://backend_proxy1/ http://www.example.com/;
  }
  # (snip)
```

リバース proxy：適用

```
$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful

$ sudo systemctl restart nginx
```

設定したサーバの”/proxy/” へアクセスする

FastCGI 設定

FastCGI：PHP-FPMのインストール

PHP-7.0 と PHP-7.0 用 PHP-FPM をインストール

```
$ sudo apt install php7.0 php7.0-fpm
```

待ち受けを inet socket に変更

```
$ sudo vi /etc/php/7.0/fpm/pool.d/www.conf
;listen = /run/php/php7.0-fpm.sock
listen = 9000
```

PHP-FPM を再起動

```
$ sudo systemctl restart php7.0-fpm
$ ss -npta | grep 9000
LISTEN     0  128  :::9000   :::*
```

FastCGI：PHP スクリプト配置

phpinfo() を実行するスクリプトを動作確認用に配置

```
$ sudo mkdir /var/www/html/myphpapp
$ sudo vi /var/www/html/myphpapp/phpinfo.php
<?php
    phpinfo();
```

FastCGI：upstream 設定

```
$ sudo vi /etc/nginx/conf.d/upstream_fcgi.conf
upstream backend_fcgi1 {
  # least_conn;
  # ip_hash;

  server 127.0.0.1:9000;
}
```

FastCGI：VirtualHost 設定

".php"へのアクセスをすべて、FastCGI サーバへ中継する

```
$ sudo vi /etc/nginx/sites-available/default
  # (snip)
  location ~ \.php$ {
    include snippets/fastcgi-php.conf;

    fastcgi_pass backend_fcgi1;
  }
  # (snip)
```

FastCGI：適用

```
$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful

$ sudo systemctl restart nginx
```

設定したサーバの”/myphpapp/phpinfo.php” へアクセスする

WSGI設定

WSGI：uwsgi のインストール

python3.5、pip3、uwsgi をインストール

```
$ sudo apt install python3 python3-pip \
 uwsgi uwsgi-plugin-python3
```

WSGI：アプリケーション配備

https://github.com/dictoss/django-tutorial を実験用に配置

```
$ sudo apt install git
$ sudo pip3 install -U django==2.0.9

$ cd
$ git clone https://github.com/dictoss/django-tutorial.git
$ sudo mkdir /var/www/wsgi_apps_uwsgi
$ sudo cp -r django-tutorial/2.0/mysite /var/www/wsgi_apps_uwsgi/
$ sudo chown -fR www-data:www-data /var/www/wsgi_apps_uwsgi/mysite
$ ls /var/www/wsgi_apps_uwsgi/mysite
db.sqlite3  manage.py  mysite  polls
```

WSGI：uwsgi 設定

設定を一部抜粋

```
$ sudo vi /etc/uwsgi/apps-available/django-tutorial.ini
[uwsgi]
uid = www-data
gid = www-data
plugin-dir = /usr/lib/uwsgi/plugins
plugin = python3
base = /var/www/wsgi_apps_uwsgi/mysite
chdir = /var/www/wsgi_apps_uwsgi/mysite
module = mysite.wsgi
callable = application
env =
socket = 0.0.0.0:3031
processes = 2
threads = 32
master = True
vacuum = True
harakiri = 60
max-requests = 512
```

WSGI：uwsgi 設定

apps-available の ini ファイルを apps-enabled へシンボリックリンクを生成する

```
$ cd /etc/uwsgi/apps-enabled
$ sudo ln -fs ../apps-available/django-tutorial.ini .
$ tree /etc/uwsgi
  /etc/uwsgi
   ├──── apps-available
   │   ├──── README
   │   ├──── django-tutorial.ini
   └──── apps-enabled
   ├──── README
   └──── django-tutorial.ini ->
              ../apps-available/django-tutorial.ini
```

WSGI：uwsgi 設定

uwsgi を再起動して設定反映

```
$ sudo systemctl restart uwsgi
$ ss -npta | grep 3031
LISTEN     0  100  *:3031  *:*
```

WSGI：upstream 設定

```
$ sudo vi /etc/nginx/conf.d/upstream_uwsgi.conf
upstream backend_uwsgi1 {
  # least_conn;
  # ip_hash;

  server 127.0.0.1:3031;
}
```

WSGI：VirtualHost 設定

"/mysite/"配下へのアクセスをすべて、WSGI サーバへ中継する

```
$ sudo vi /etc/nginx/sites-available/default
  # (snip)
  location ~ ^/mysite/(.*)$ {
    include uwsgi_params;

    uwsgi_param SCRIPT_NAME /mysite;
    uwsgi_param PATH_INFO /$1;

    uwsgi_pass backend_uwsgi1;
  }
  # (snip)
```

WSGI：適用

```
$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful

$ sudo systemctl restart nginx
```

設定したサーバの”/mysite/polls/” へアクセスする

まとめ・参考文献

まとめ・参考文献

まとめ

- nginx のよく使う設定をまとめてみました
- 大規模向けに性能向上するパラメータについては
  チューニングしてみてください

参考文献

- nginx documentation `https://nginx.org/en/docs/`
- nginx - DebianWiki `https://wiki.debian.org/Nginx`