

MENONFIGURASI NGINX SEBAGAI REVERSE PROXY SERVER

Secara default, konfigurasi web server untuk NGINX berada pada berkas `/etc/nginx/sites-available/default`. Di berkas tersebut, kita bisa melihat konfigurasi dasar dari NGINX sebagai web server. Jadi, silakan masuk ke EC2 instance Anda melalui SSH menggunakan metode EC2 Instance Connect seperti biasa.

1. Jalankan perintah berikut untuk bisa melihat isi dari berkas yang kita bincangkan tadi.
 1. `cat /etc/nginx/sites-available/default`Catatan: `cat` merupakan perintah untuk melihat isi berkas dalam bentuk teks.
2. Mari kita edit berkas tersebut menggunakan tools `nano` dengan menjalankan perintah berikut.

2. `sudo nano /etc/nginx/sites-available/default`

3. Kemudian, tambahkan kode yang diberi tanda tebal berikut pada blok `server` -> `location` /.

1. `location / {`
 2. **`proxy_pass http://localhost:8000;`**
 3. **`proxy_http_version 1.1;`**
 4. **`proxy_set_header Upgrade $http_upgrade;`**
 5. **`proxy_set_header Connection 'upgrade';`**
 6. **`proxy_set_header Host $host;`**
 7. **`proxy_cache_bypass $http_upgrade;`**
 - 8.
 9. `# ...`
 10. `}`

Catatan: Jika Anda kesulitan dalam melakukan penyalinan kode, silakan tulis saja secara manual.

4. Lalu, hapus kode lain yang berada di dalam blok **`location`** /seperti di bawah ini.

1. `# First attempt to serve request as file, then`
 2. `# as directory, then fall back to displaying a 404.`
 3. `try_files $uri $uri/ =404;`

5. Sehingga kini blok **`location`** / tampak seperti berikut.

```
# Add index.php to the list if you are using PHP
index index.html index.htm index.nginx-debian.html;

server_name funny-fly-10.a276.dcdg.xyz www.funny-fly-10.a276.dcdg.xyz;

location / {
    proxy_pass http://localhost:8000;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection 'upgrade';
    proxy_set_header Host $host;
    proxy_cache_bypass $http_upgrade;

    limit_req zone=one;
}
```

6. Simpan perubahan berkasnya dengan menekan **CTRL+X**, lalu **Y**, dan **Enter**. Usai itu, jalankan ulang NGINX server menggunakan perintah berikut.
 1. `sudo systemctl restart nginx`
7. Selanjutnya, jalankan web server yang kita punya (web server Node.js) dengan perintah berikut.
 1. `cd web-server-01/`
 2. `npm run start`
8. Lantas, akses kembali public IP address EC2 instance Anda. Kini seharusnya NGINX akan merespons dengan menampilkan web server milik kita (web server Node.js). *“Hello Mochamad Miftah Rachmatullah!”*



MENERAPKAN LIMIT ACCES DENGAN NGINX DI EC2 INSTANCE

Silakan masuk kembali ke EC2 instance Anda melalui SSH menggunakan metode EC2 Instance Connect seperti biasa.

1. Buka kembali berkas konfigurasi web server NGINX menggunakan nano melalui perintah berikut.

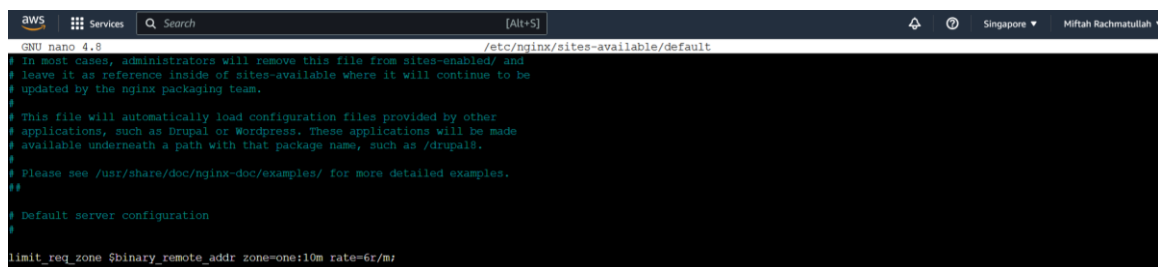
1. `sudo nano /etc/nginx/sites-available/default`

Catatan: Jika web server Node.js masih berjalan, silakan hentikan prosesnya dengan menekan CTRL+C.

2. Kemudian, di baris awal berkas atau sebelum blok *server*, tuliskan kode berikut.

1. `limit_req_zone $binary_remote_addr zone=one:10m rate=6r/m;`

Seperti ini hasilnya.



```
GNU nano 4.8 /etc/nginx/sites-available/default
# In most cases, administrators will remove this file from sites-enabled/ and
# leave it as reference inside of sites-available where it will continue to be
# updated by the nginx packaging team.
#
# This file will automatically load configuration files provided by other
# applications, such as Drupal or Wordpress. These applications will be made
# available underneath a path with that package name, such as /drupal8.
#
# Please see /usr/share/doc/nginx-doc/examples/ for more detailed examples.
#
# Default server configuration
#
limit_req_zone $binary_remote_addr zone=one:10m rate=6r/m;
```

3. Lanjut, tambahkan juga kode berikut di dalam blok *location* /.

1. `limit_req zone=one;`

Hasilnya seperti berikut.



```
# Add index.php to the list if you are using PHP
index index.html index.htm index.nginx-debian.html;

server_name funny-fly-10.a276.dcdg.xyz www.funny-fly-10.a276.dcdg.xyz;

location / {
    proxy_pass http://localhost:8000;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection 'upgrade';
    proxy_set_header Host $host;
    proxy_cache_bypass $http_upgrade;

    limit_req zone=one;
}
```

Catatan: Kode pada langkah ke-2 dan ke-3 merupakan sintaks untuk melimitasi akses pada web server NGINX. Berdasarkan kode tersebut, kita menginginkan pengguna yang mengakses resource / (root) hanya dapat membuat permintaan setiap 10 detik (*rate=6r/m* berarti 6 request per menit atau setara dengan 10 detik sekali).

4. Simpan perubahan berkasnya dengan menekan **CTRL+X**, lalu **Y**, dan **Enter**. Usai itu, jalankan ulang NGINX server menggunakan perintah berikut.

1. `sudo systemctl restart nginx`

5. Selanjutnya, silakan akses kembali public IP address EC2 instance Anda pada browser. Lalu, lakukan reload berulang kali secara cepat. Web server seharusnya hanya akan

merespons permintaan dalam **sepuluh detik sekali**. Jika dalam 10 detik Anda melakukan request (dalam hal ini reload) lebih dari satu kali, ia akan menampilkan respons seperti ini.



6. Namun, jika Anda berhenti sejenak dan reload kembali dalam sepuluh detik atau lebih, web server akan berjalan normal.



Catatan: Jika Anda mendapati respons *502 Bad Gateway*, itu tandanya web server Node.js Anda belum berjalan. Silakan jalankan web server Node.js dengan perintah di bawah ini, lalu akses kembali public IP address EC2 instance Anda.

1. `cd web-server-01/`
2. `npm run start`