

**TO  
THE  
NEW**™



## **Nginx-Webserver**

Trainee Name : Chhavi Sharma

Newers ID : 4023

Mentor Name : Neeraj Gupta

College : UPES

1.What is the advantage of using a “reverse proxy server”?

Ans.

Reverse proxy server is the one that retrieves the resources on behalf of the client from one or more servers.It directs the client requests to the appropriate backend server.It provides an additional level of abstraction and control. Hence ensures a smooth flow of traffic between the client and the server.

Advantages of using a proxy server :

- 1.**Load Balancing and Failover** : A reverse proxy can provide a load balancing solution. This will distribute the incoming traffic evenly among the different servers in order to prevent any single server from becoming overloaded. In the event that a server fails completely, other servers can step up to handle the incoming traffic.
- 2.**Caching**: A reverse proxy can also cache content, resulting in faster performance.
- 3.**Security and anonymity** – By intercepting requests headed for the backend servers, a reverse proxy server protects their identities and acts as an additional defense against security attacks.
- 4.**Creates a single point of access (control)to your file transfer servers** : For as long as you've configured your firewall and reverse proxy correctly, no one should be able to gain direct access to any of your file transfer servers. Everyone has to pass through the reverse proxy. When that happens, you can focus monitoring over what goes in and goes out through the reverse proxy.

2.Why and where Nginx is a better choice than apache?

Ans.

>>If the **website is PHP** dependent, Nginx is the far best way to host applications(BackEnd).

>>NGINX serves **static content** much **faster** than Apache. If you need to serve a lot of static content at high concurrency levels, NGINX can be a real help.

>>Good for high traffic websites.

3.What are worker nodes and worker connections? How to calculate the max server capacity using the above two?

Ans.

**Worker Node/Server Node**: A server node is a virtual node which is created by nginx to serve user requests. The details of each worker node/server node is mentioned in the server context of the nginx.conf file.

(A worker process is a single-threaded process.)

**Worker connections** : NGINX can run multiple worker processes, each capable of processing a large number of simultaneous connections. Worker connections tells worker

processes how many clients can be served simultaneously by Nginx. Default value is 768. Each browser will usually open at least 2 server **connections**.

Maximum number of connections (max capacity)= worker\_processes \* worker\_connections

4. From what directory will NGINX automatically load server (virtual host) configurations when using the default /etc/nginx/nginx.conf configuration?

Ans. /etc/nginx/conf.d directory

5. How to configure different log\_format for different "location" block/directive?

Ans.

nginx.conf

```
##
# Logging Settings
##

access_log /var/log/nginx/access.log;
error_log /var/log/nginx/error.log;

log_format custom '$request' $status $body_bytes_sent '
                  '$http_referer' '$http_user_agent' '
                  '$http_x_forwarded_for' $request_id ';
```

abc.com

```
chhavi@chhavi:/etc/nginx/sites-available$ cat abc.com
server{
    listen 80;
    root /var/www/html;
    index abc.html;
    error_page 404 error.html;
    server_name abc.com;

    location /{
        access_log /var/log/nginx/access.log custom;
    }
}
```

access.log

```

127.0.0.1 - - [17/Feb/2020:11:07:35 +0530] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.100 Safari/537.36" "-" c76cbcd448d5d5c0e4917df3012f328
- - -
"GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.100 Safari/537.36" "-" 0777d38a07025ca9d156988d14c9ab37
"GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.100 Safari/537.36" "-" abe7f44d7cd98c5bd7332ffbb0f87128
"GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.100 Safari/537.36" "-" 29d0399db0e2e0881d3d97b55af132e3
"GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.100 Safari/537.36" "-" a063c451104128ba480f6e60dd6a51b8
"GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.100 Safari/537.36" "-" fcbb741221e191e19296efd2e7de8827
"GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.100 Safari/537.36" "-" 469304e213e5fdb85780f4448fb46ea5
chhavi@chhavi:/var/log/nginx$

```

## 6.Host a site ABC.COM

a.Make a page abc.html in /var/www/html

b. In /etc/nginx/sites-available make a file abc.com and write the server block in it.

c.make a symlink of this file in sites-enabled

```

chhavi@chhavi:/var/www/html$ cd
chhavi@chhavi:~$ cd /etc/nginx/
chhavi@chhavi:/etc/nginx$ ls
conf.d          koi-win        nginx.conf     sites-enabled
fastcgi.conf    mime.types     proxy_params   snippets
fastcgi_params  modules-available  scgi_params    uwsgi_params
koi-utf         modules-enabled  sites-available  win-utf
chhavi@chhavi:/etc/nginx$ cd sites-available/
chhavi@chhavi:/etc/nginx/sites-available$ sudo vim abc.com

```

```

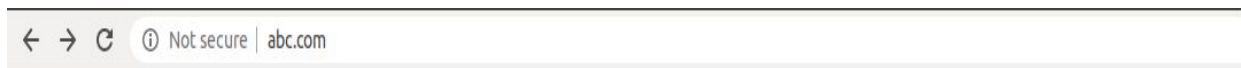
chhavi@chhavi:/etc/nginx/sites-available$ cat abc.com
server{
    listen 80;
    root /var/www/html;
    index abc.html;
    error_page 404 error.html;
    server_name abc.com;
}
chhavi@chhavi:/etc/nginx/sites-available$

```

```

chhavi@chhavi:/etc/nginx/sites-enabled$ sudo ln -s ../sites-available/abc.com .
chhavi@chhavi:/etc/nginx/sites-enabled$ ls
abc.com  example.com
chhavi@chhavi:/etc/nginx/sites-enabled$ vim abc.com
chhavi@chhavi:/etc/nginx/sites-enabled$ ls
abc.com  example.com
chhavi@chhavi:/etc/nginx/sites-enabled$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
chhavi@chhavi:/etc/nginx/sites-enabled$ service nginx restart
chhavi@chhavi:/etc/nginx/sites-enabled$ ls

```



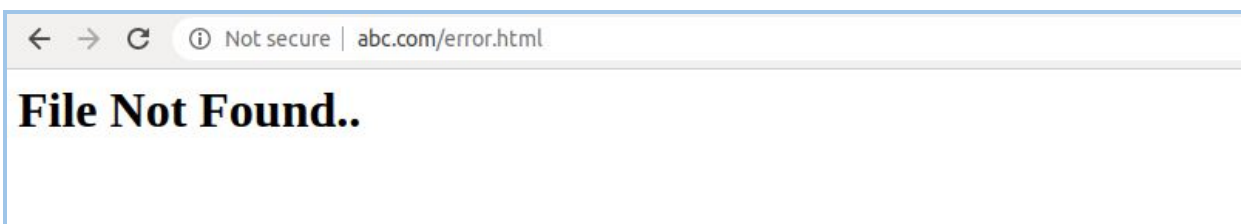
Hello from abc.com

6.1. Create an index page and a fail-safe page. If a page for URI is not available, the fail-safe page is served.

Ans.

```
chhavi@chhavi:/var/www/html$ cat abc.html
<html>
  <head>
    <title>Test</title>
  </head>
  <body>
    Hello from abc.com
  </body>
</html>
chhavi@chhavi:/var/www/html$
```

```
chhavi@chhavi:/etc/nginx/sites-available$ cat abc.com
server{
    listen 80;
    root /var/www/html;
    index abc.html;
    error_page 404 error.html;
    server_name abc.com;
}
chhavi@chhavi:/etc/nginx/sites-available$
```



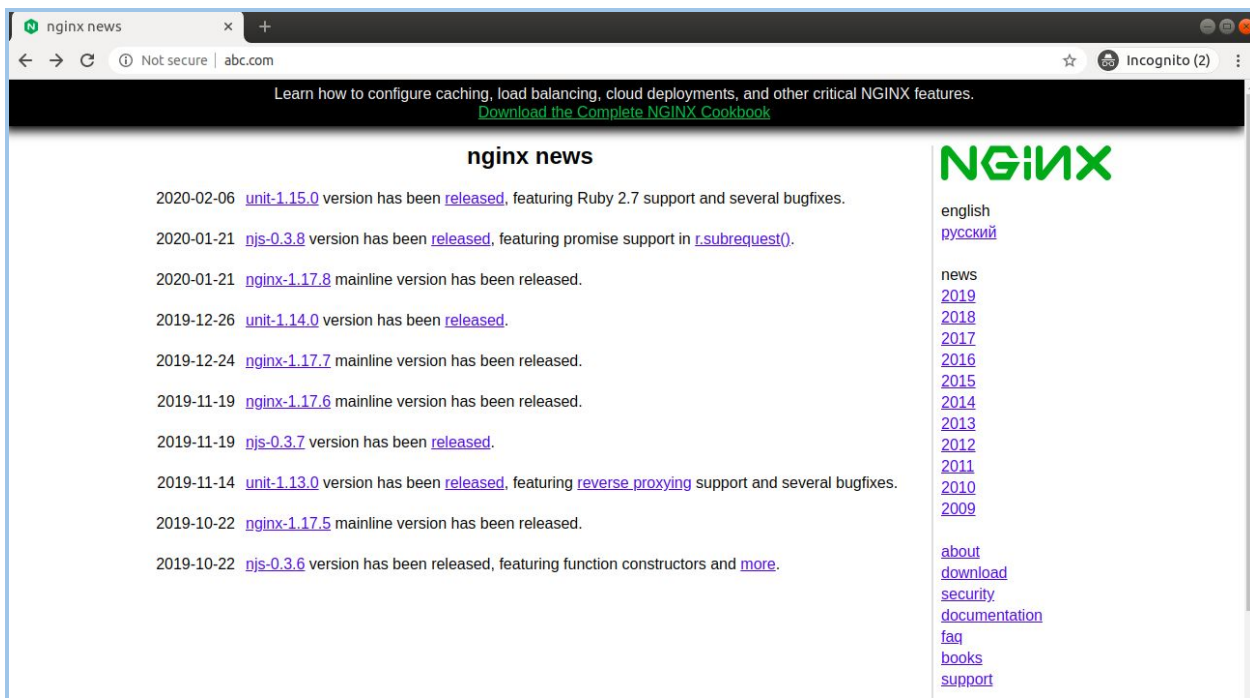


6.2. proxy pass to a website xyz.com on a particular URI.

Ans. abc.com proxy passed to <http://nginx.org/>

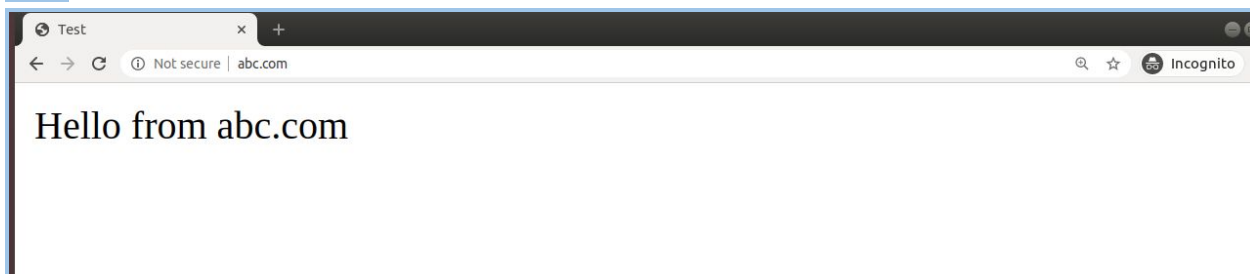
```
chhavi@chhavi:/etc/nginx/sites-enabled$ cat abc.com
server{
    listen 80;
    root /var/www/html;
    index abc.html;
    error_page 404 error.html;
    server_name abc.com;

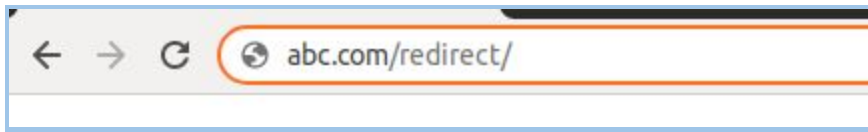
    location /{
        proxy_pass http://nginx.org/;
    }
}
```



6.3. redirect to above URI on /redirect/

Ans.

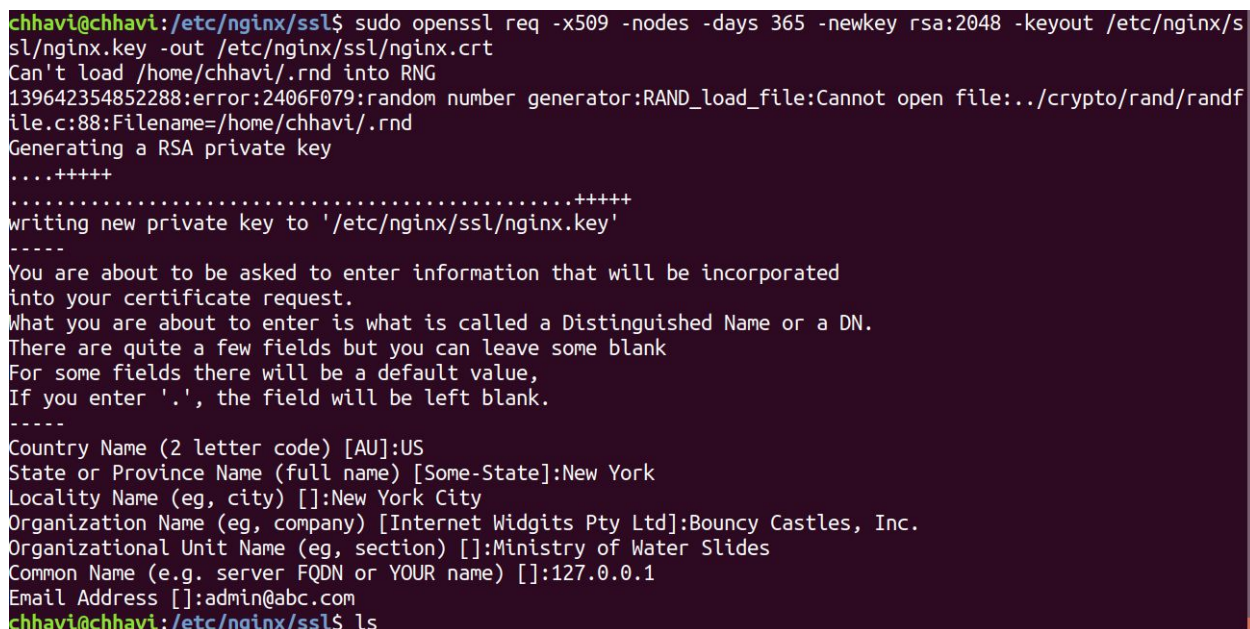




6.4. perform an HTTP to HTTPS redirection including non-www to www redirection.

Ans.

Create an ssl certificate



## Changes in abc.com

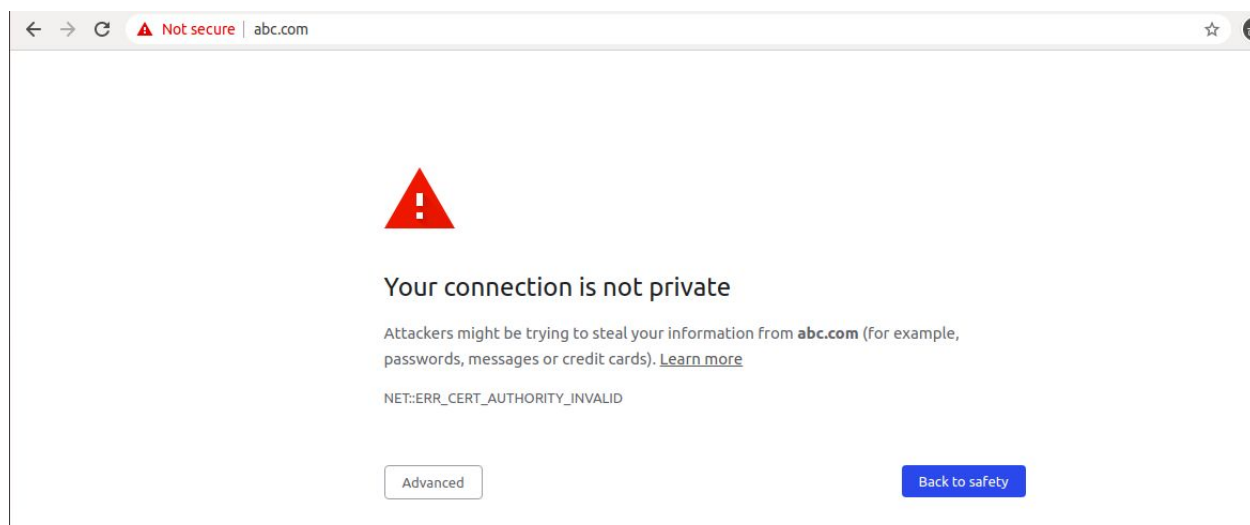
```
server {  
    listen 80;  
    server_name abc.com;  
    root /var/www/html;  
    index abc.html;  
    error_page 404 error.html;  
    return 301 https://www.abc.com;  
    location / {  
        access_log /var/log/nginx/access.log custom;  
        error_log /var/log/nginx/error.log;  
    }  
}  
  
server {  
    listen 443 ssl;  
    server_name www.abc.com;  
    root /var/www/html;  
    index ssl.html;  
    ssl on;  
    ssl_certificate /etc/nginx/ssl/nginx.crt;  
    ssl_certificate_key /etc/nginx/ssl/nginx.key;  
}  
  
"abc.com" 24L, 575C
```

18,27 All

## Changes in /etc/hosts

```
127.0.0.1    localhost abc.com xyz.com www.abc.com  
127.0.1.1    chhavi  
  
# The following lines are desirable for IPv6 capable hosts  
::1          ip6-localhost ip6-loopback  
fe00::0      ip6-localnet  
ff00::0      ip6-mcastprefix  
ff02::1      ip6-allnodes  
ff02::2      ip6-allrouters  
~  
~
```

## Redirection





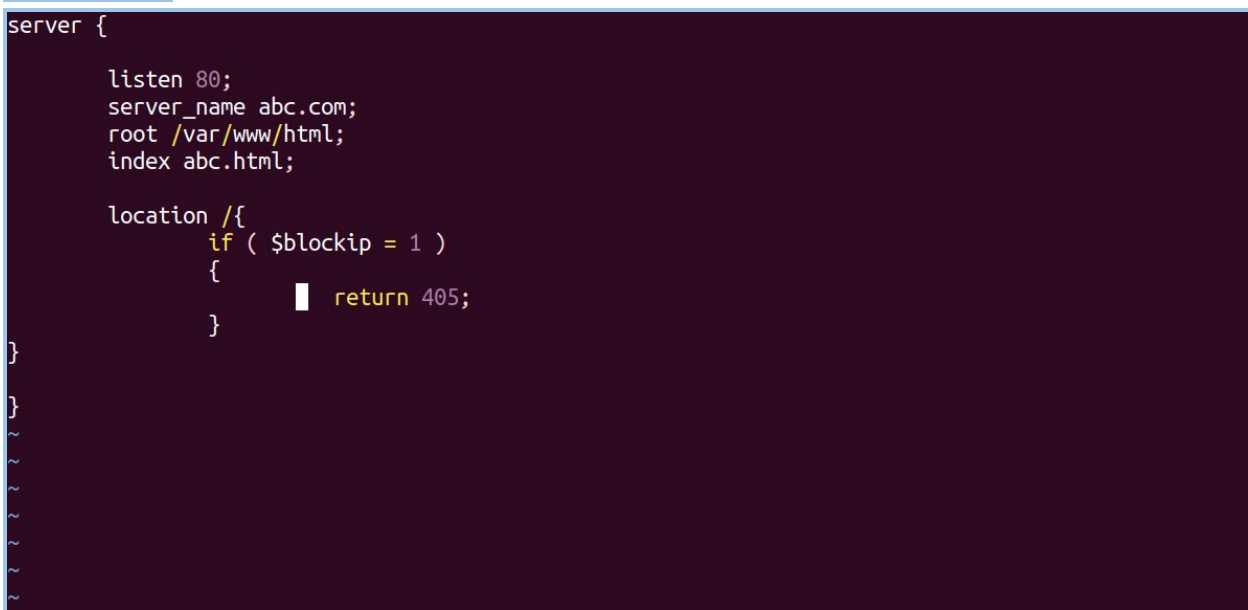


6.5.Allow access to a set of particular IPs on a location block and return 405 to other IPs no matter if the page in that location exists.

Ans.



abc.com file



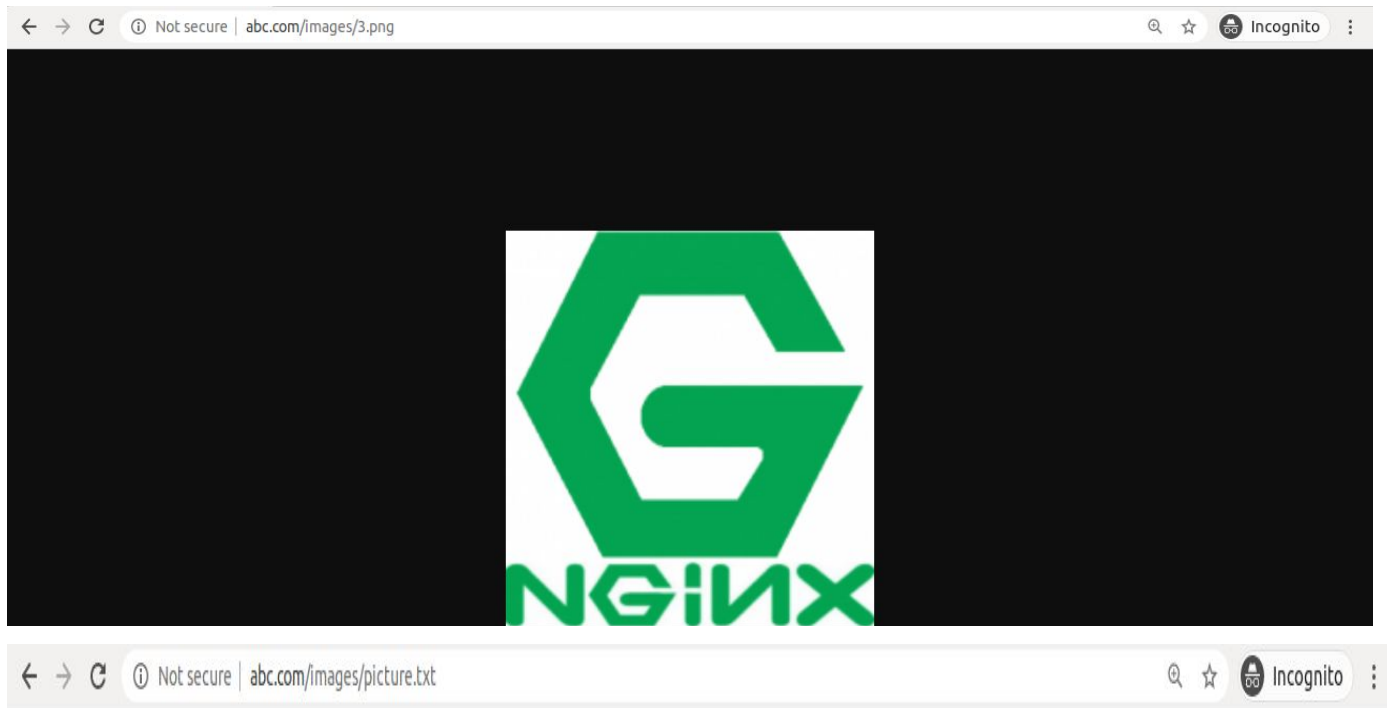
block\_allow\_ip

```
geo $blockip
{
default 0;
127.0.0.1 1;
}
~
~
```

6.6. Place your images at /var/www/html/images. Only accept jpg/png/jpeg. Discard rest.  
Ans.

```
server {
    listen 80;
    server_name abc.com;
    root /var/www/html;
    index abc.html;
    location ~* ^.+\. (jpg|jpeg|png)$
    {
        allow all;
    }

    location /images
    {
        deny all;
    }
}
```



## 403 Forbidden

nginx/1.14.0 (Ubuntu)



7. Create a load balancer with 5 backends. Explain different types of load balancing methods.

Ans.

Create a Load\_balancer file in sites-available with the upstream block and create a soft link in the sites-enabled directory .

```
upstream backend{
    server 127.0.0.1:81;
    server 127.0.0.1:82;
    server 127.0.0.1:83;
    server 127.0.0.1:84;
    server 127.0.0.1:85;
}

server{
    listen 80;
    server_name abc.com;
    location /{
        proxy_pass http://backend;
    }
}

server{
    listen 81;
    server_name _;
    root /var/www/html;
    index def.html;
}

server{
    "load_balancer" 48L, 733C
    1,1
    Top
```

```
server{
    listen 81;
    server_name _;
    root /var/www/html;
    index def.html;
}

server{
    listen 82;
    server_name _;
    root /var/www/html;
    index ghi.html;
}

server{
    listen 83;
    server_name _;
    root /var/www/html;
    index jkl.html;
}

server{
    listen 84;
    server_name _;
    root /var/www/html;
    index xyz.html;
}

35,1
66%
```

```
chhavi@chhavi:/etc/nginx/sites-enabled$ sudo ln -s ../sites-available/load_balancer .
chhavi@chhavi:/etc/nginx/sites-enabled$ sudo nginx -t
nginx: [warn] conflicting server name "abc.com" on 0.0.0.0:80, ignored
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
chhavi@chhavi:/etc/nginx/sites-enabled$ sudo systemctl reload nginx.service
```

Now on hitting abc.com we are redirected to different servers blocks.



Different types of load balancing techniques :

1. Round Robin : This is the default technique. It runs through the list of upstream servers in sequence and assigns the request to them one by one in round robin fashion.



2. Hash: For each request the load balancer calculates a hash that is based on the combination of text and NGINX variables that we specify, and associates the hash with one of the servers. It sends all requests with that hash to that server, so this method establishes a kind of session persistence.
3. IP Hash: It is available only for HTTP. The hash is based on the client's ip address. (**ip\_hash** directive is used).
4. Least Connections: The load balancer compares the current number of active connections it has to each server, and sends the request to the server with the fewest connections. We can configure it with the **least\_conn** directive.
5. Least Time: The load balancer mathematically combines two metrics for each server, the current number of active connections and a weighted average response time for past requests and sends the request to the server with the lowest value.

**least\_time (header | last\_byte);**

( choice of parameter on the least\_time directive controls which of two response times is tracked: either the time to receive the response header (header) or the time to receive the full response (last\_byte). )

8. Setup Basic Auth (Popup asking for username and password) in a particular location block. (The Basic Auth should not be asked for TTN IP)

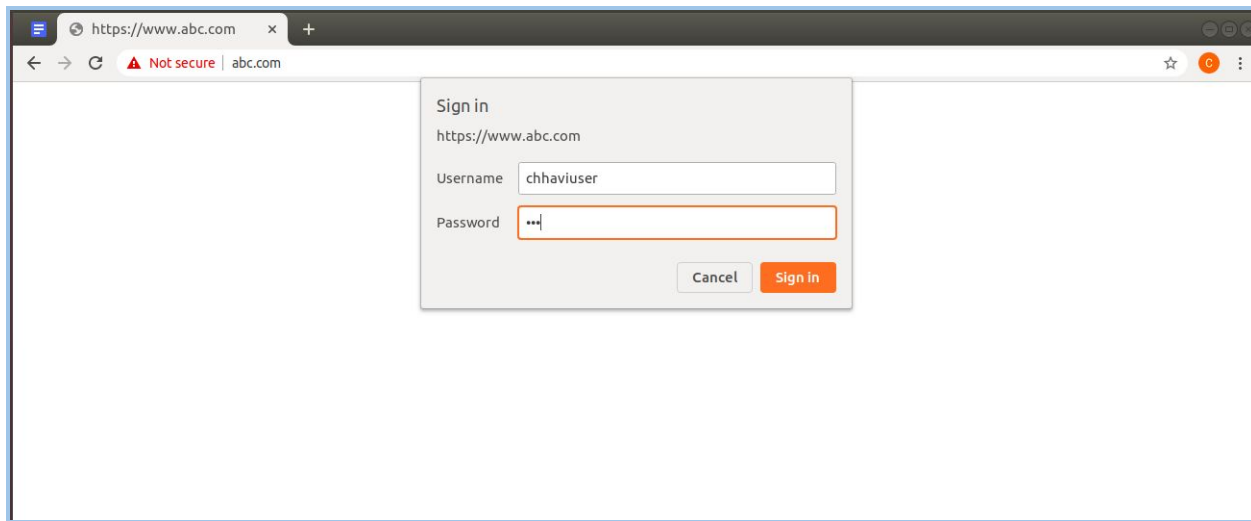
Ans.

```
chhavi@chhavi:~$ sudo apt-get install apache2-utils
[sudo] password for chhavi:
Reading package lists... Done
Building dependency tree
Reading state information... Done
apache2-utils is already the newest version (2.4.29-1ubuntu4.11).
apache2-utils set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 39 not upgraded.
chhavi@chhavi:~$
```

```
chhavi@chhavi:~$ sudo htpasswd -c /etc/nginx/.htpasswd chhaviuser
New password:
Re-type new password:
Adding password for user chhaviuser
```

```
server {  
    listen 443 ssl;  
    server_name www.abc.com;  
    root /var/www/html;  
    index ssl.html;  
    ssl on;  
    ssl_certificate /etc/nginx/ssl/nginx.crt;  
    ssl_certificate_key /etc/nginx/ssl/nginx.key;  
    auth_basic "chhaviuser";  
    auth_basic_user_file /etc/nginx/.htpasswd;  
}
```

28,0-1



https://www.abc.com

Not secure | abc.com

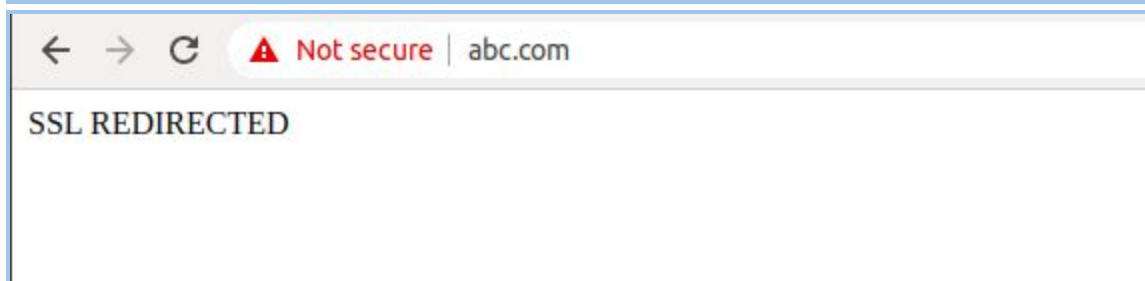
Sign in

https://www.abc.com

Username chhaviuser

Password \*\*\*

Cancel Sign in



Not secure | abc.com

SSL REDIRECTED

```
server {  
  
    listen 80;  
    server_name abc.com;  
    root /var/www/html;  
    index abc.html;  
    satisfy any;  
    allow 10.1.211.251;  
    auth_basic "chhaviuser";  
    auth_basic_user_file /etc/nginx/.htpasswd;  
  
}
```

Using my own ip : 10.1.211.251

