****

**Installation et Configuration du Serveur Web Apache**

**Master Qualité logiciel – Faculté des sciences**

**Réalisée par :**

**KAIS Zakaria**

**HAMMOCHAMA Oussama**

**EL AOUMRANI Marouan**

**Demandé par :**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Pr.** | |  |  | | --- | --- | | |  | | --- | | **ALI Younes** | | | **Département d’Informatique** |

**Année universitaire : 2022-202****3**

Table of contents

[1. Buying DNS 4](#_Toc136869445)

[1.1. Definition 4](#_Toc136869446)

[1.2. Buying our domain 4](#_Toc136869447)

[1.3. Linking domain name to the server IP 5](#_Toc136869448)

[1.4. Testing the domain 5](#_Toc136869449)

[2. Installation of Apache and Configuration of Virtual Host 6](#_Toc136869450)

[2.1. Defintion 6](#_Toc136869451)

[2.2. Installation of Apache 6](#_Toc136869452)

[2.3. Set up Apache Virtual Host on Ubuntu 22.04 9](#_Toc136869453)

[2.4. Creating an Apache Virtual Host file 11](#_Toc136869454)

[2.5. Enable Virtual Host file 12](#_Toc136869455)

[2.6. Error testing 12](#_Toc136869456)

[2.7. Apache Virtual Host testing 13](#_Toc136869457)

[3. Load balancing 13](#_Toc136869458)

[3.1. Definition 13](#_Toc136869459)

[3.2. Enable required modules 14](#_Toc136869460)

[3.3. Configure the load balancer 14](#_Toc136869461)

[3.4. Enable the load balancer site 16](#_Toc136869462)

[3.5. Restart Apache 17](#_Toc136869463)

[3.6. Test in the browser 17](#_Toc136869464)

[4. Reverser proxy 18](#_Toc136869465)

[4.1. Definition 18](#_Toc136869466)

[4.2. Enable the necessary Apache modules 18](#_Toc136869467)

[4.3. Configure the reverse proxy 19](#_Toc136869468)

[4.4. Enable the site: 20](#_Toc136869469)

[4.5. Restart Apache 21](#_Toc136869470)

[4.6. Test on your browser 22](#_Toc136869471)

[5. SSL 22](#_Toc136869472)

[5.1. Definition 22](#_Toc136869473)

[5.2. SSH into the server 23](#_Toc136869474)

[5.3. Ensure that your version of snapd is up to date 24](#_Toc136869475)

[5.4. Install Certbot 25](#_Toc136869476)

[5.5. Install the certificate 26](#_Toc136869477)

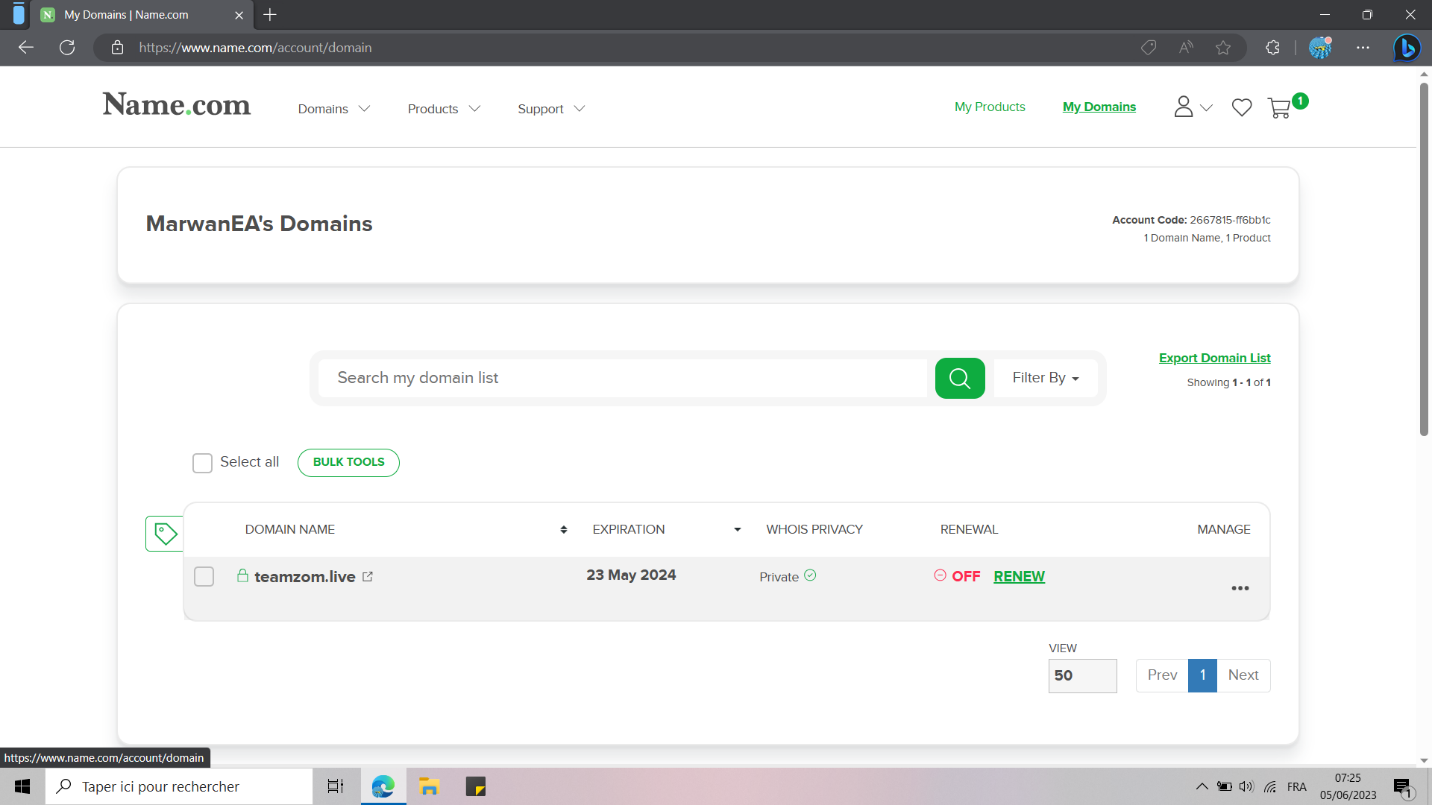
[5.6. Test your domaine 27](#_Toc136869478)

# Buying DNS

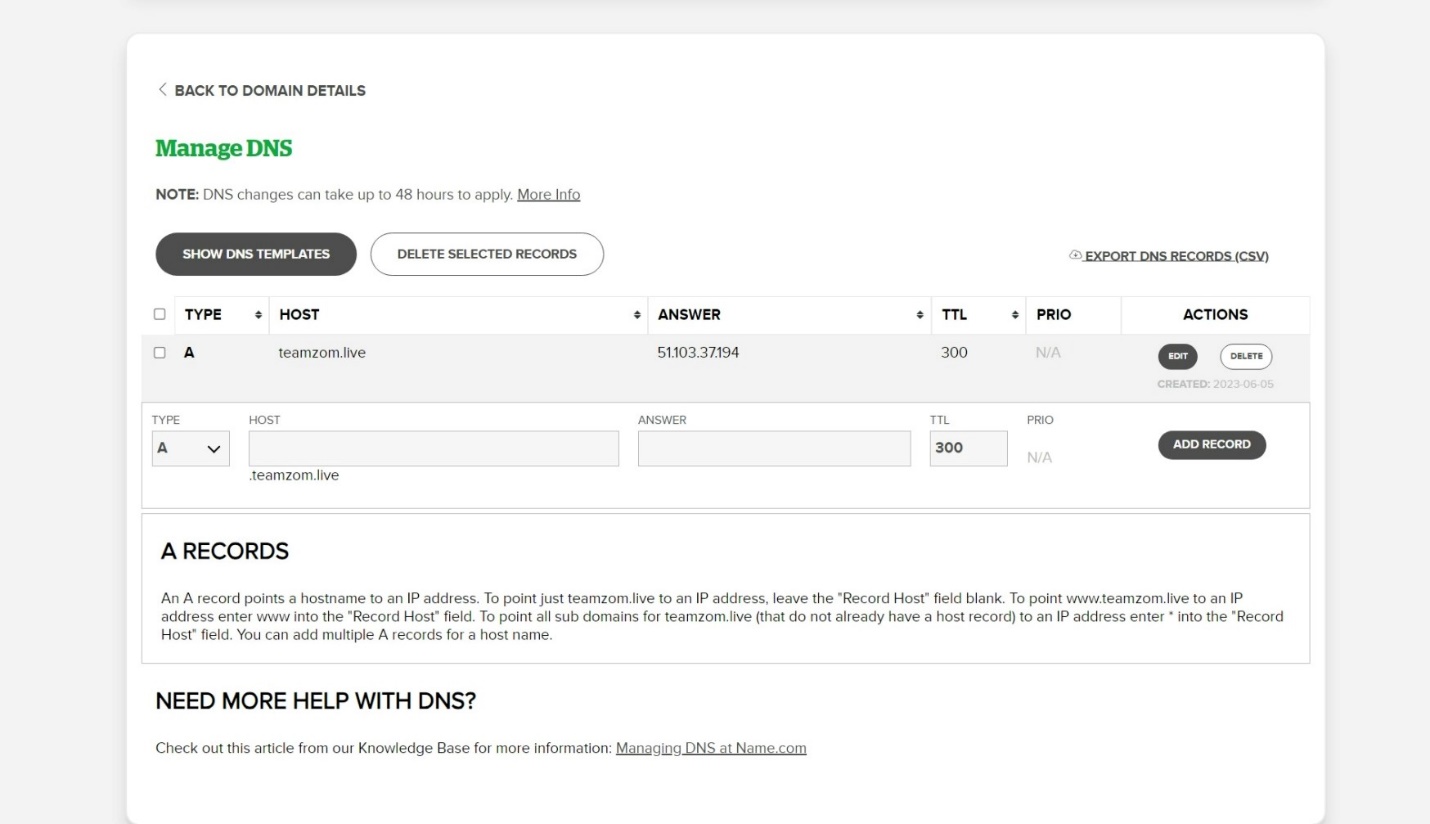
## Definition

DNS stands for Domain Name System. It is a decentralized system used to translate human-readable domain names (such as www.example.com) into machine-readable IP addresses (such as 192.0.2.1). DNS is an essential protocol of the internet that enables users to access websites and other resources using easily memorable domain names rather than having to remember complex numerical IP addresses.

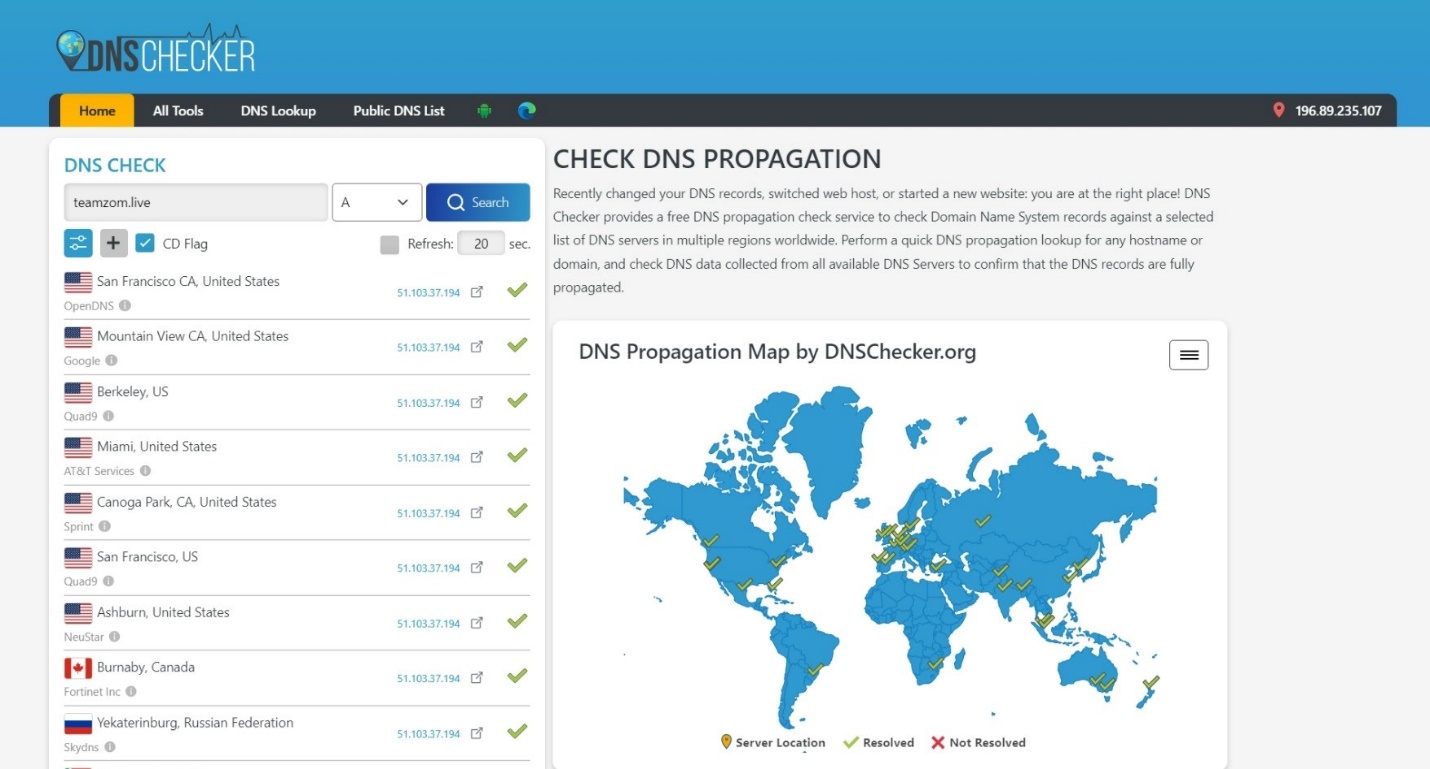
## Buying our domain



## Linking domain name to the server IP



## Testing the domain



# Installation of Apache and Configuration of Virtual Host

## Defintion

Apache, also known as the Apache HTTP Server, is a widely used open-source web server software. It is one of the most popular web server solutions and has been a cornerstone of the internet since its release in 1995. Apache is developed and maintained by the Apache Software Foundation.

As a web server, Apache's primary function is to serve web pages and handle HTTP requests from clients, such as web browsers. It operates on a client-server model, where clients make requests to the Apache server, and it responds by delivering the requested content, which can be web pages, images, files, or other resources.

A virtual host, in the context of web servers, refers to the practice of hosting multiple websites or domains on a single physical server. It allows a server to serve different websites with distinct domain names, even though they share the same IP address.

Virtual hosting is essential because it allows efficient utilization of server resources by hosting multiple websites on a single server instead of requiring a separate physical server for each website. It enables cost savings, easier management, and scalability.

## Installation of Apache

Open a terminal.

Update the package lists by running the following command:

**sudo** apt update

Install Apache2 using the following command:

**sudo** apt install apache2

|  |
| --- |
| A screenshot of a computer  Description automatically generated |

Type Y and entre

|  |
| --- |
| A screenshot of a computer  Description automatically generated  A screen shot of a computer  Description automatically generated with medium confidence |

To start Apache on a Linux system, you can use the following commands:

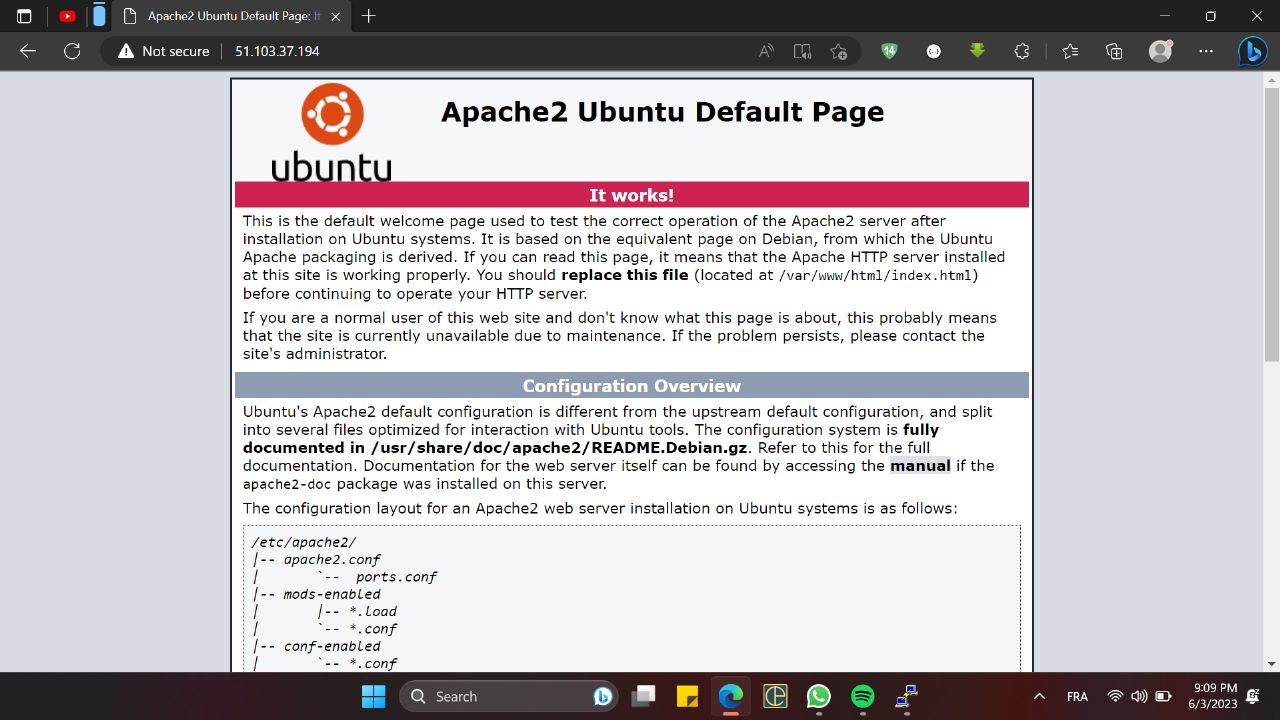
**sudo** systemctl start apache2

To check if Apache is installed on your Linux system, you can use the following command in the terminal:

**sudo** systemctl status apache2

|  |
| --- |
| A screenshot of a computer  Description automatically generated  A picture containing text, screenshot, font, design  Description automatically generated |

You can also utilize your favorite web browser for the specified verification. To do so, open a web browser and check what the “localhost” web page beholds for you:



Before setting up Virtual Hosts, it is important to check if Apache is working or not. For this purpose, we will verify the status of the “apache2” service:

## Set up Apache Virtual Host on Ubuntu 22.04

To set up a virtual host in Apache, create a directory that will be utilized to store data on the website. For this purpose, we will move to the “/var/www” directory using the following “cd” command:

**cd** /var/www/

Then, we will create a directory for our domain “myVHost”. Here, you can specify your domain name in the below-given command:

**sudo** mkdir -p /var/www/myVHost/

|  |
| --- |
| A screenshot of a computer  Description automatically generated |

Utilize the “chown” command for changing the ownership of the “myVHost” directory:

**sudo** chown -R www-data:www-data /var/www/myVHost

|  |
| --- |
| A screenshot of a computer  Description automatically generated  A computer screen with white text  Description automatically generated with low confidence |

**Step 3: Creating a web page**

To create a sample “index.html” web page for our website, we will use the “nano” or "Vi" or "emacs" … editor:

**sudo** emacs /var/www/myVHost/index.html

html code:

<!DOCTYPE *html*>

<html>

<head>

  <title>My HTML Page</title>

</head>

<body>

  <h1>Hello, World!</h1>

  <p>Success! The tutorial is all about apache2 configuration!.</p>

</body>

</html>

## Creating an Apache Virtual Host file

At this point, we have created a directory for our domain and updated its ownership. Now, we will create a virtual host file under the default directory of Apache host files:

**sudo** emacs /etc/apache2/sites-available/myVHost.conf

In the opened virtual host file, add the following lines of code. Also, you have to replace the information related to “ServerName”, “ServerAlias”, and “DocumentRoot” according to your settings:

<**VirtualHost** \*:80>

**ServerAdmin** admin@admin\_myVhost

**ServerName** myVHost

**ServerAlias** www.myVHost

**DocumentRoot** /var/www/myVHost

**ErrorLog** ${APACHE\_LOG\_DIR}/error.log

**CustomLog** ${APACHE\_LOG\_DIR}/access.log combined

</**VirtualHost**>

## Enable Virtual Host file

Execute the following “a2ensite” command to enable the created virtual host file:

**sudo** a2ensite myVHost.conf

|  |
| --- |
| A screenshot of a computer  Description automatically generated |

Then disable the default configuration file:

**sudo** a2dissite 000-default.conf

|  |
| --- |
| A screenshot of a computer  Description automatically generated  A screen shot of a computer  Description automatically generated with medium confidence |

After performing the specified operation. Restart the “apache” service on your Ubuntu

**sudo** systemctl restart apache2

## Error testing

In the last step of Apache2 configuration, test the configuration errors:

In case of having an error-free configuration file, the execution of the above-given command will let you know that the Syntax is “OK”:

**sudo** apache2ctl configtest

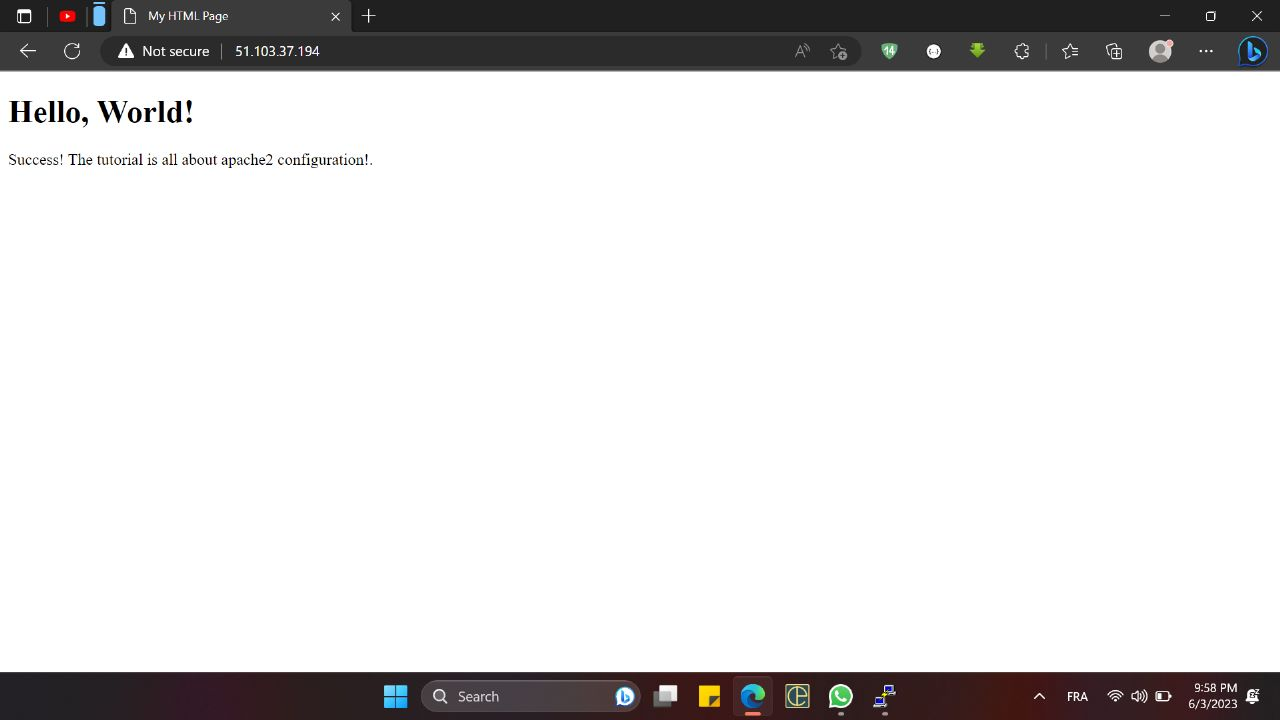
Then, restart the “apache2” service on your Ubuntu system:

**sudo** systemctl restart apache2

|  |
| --- |
| A screenshot of a computer  Description automatically generated |

## Apache Virtual Host testing

Finally, test your Virtual host by navigating to the specified domain. In our case, the domain is our local IP address:



The displayed information justifies that our Apache Virtual Host is up and working perfectly on Ubuntu system.

# Load balancing

## Definition

A load balancer is a critical component in computer networks and web applications that distributes incoming network traffic across multiple servers or resources. Its primary purpose is to evenly distribute the workload among the available resources to optimize performance, enhance reliability, and ensure high availability.

When multiple servers are behind a load balancer, it acts as a mediator between the clients (such as web browsers) and the servers. It receives incoming requests from clients and intelligently routes them to the appropriate server based on factors like server availability, capacity, current workload, and sometimes even geographic proximity.

To create a load balancer with Apache2 on Ubuntu, you can follow these steps:

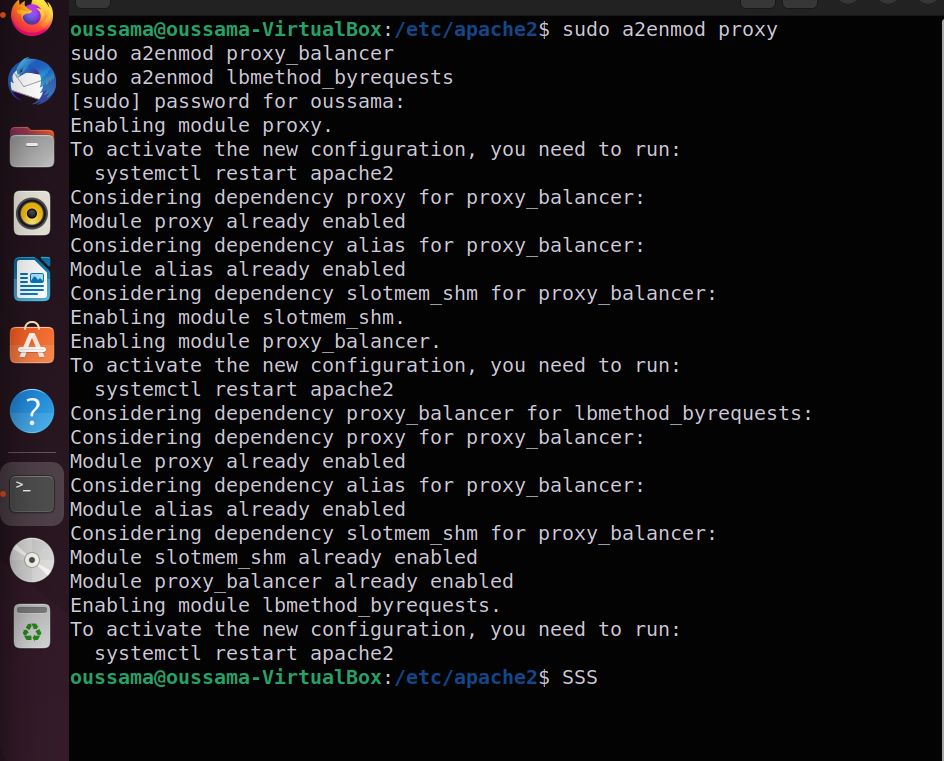
## Enable required modules

Enable the necessary Apache modules for load balancing by running the following commands:

**sudo a2enmod proxy**

**sudo a2enmod proxy\_balancer**

**sudo a2enmod lbmethod\_byrequests**



## Configure the load balancer

Open the Apache configuration file for your load balancer site. You can use any text editor to do this.

**sudo nano /etc/apache2/sites-available/load-balancer.conf**

In the configuration file, add the following lines to set up the load balancer:

<VirtualHost \*:80>

<Proxy "balancer://mycluster">

BalancerMember "http://51.102.37.194:80"

BalancerMember "http://100.100.47.28:80"

# Add more BalancerMembers as needed for additional backend servers

# BalancerMember http://server3-ip-address:80

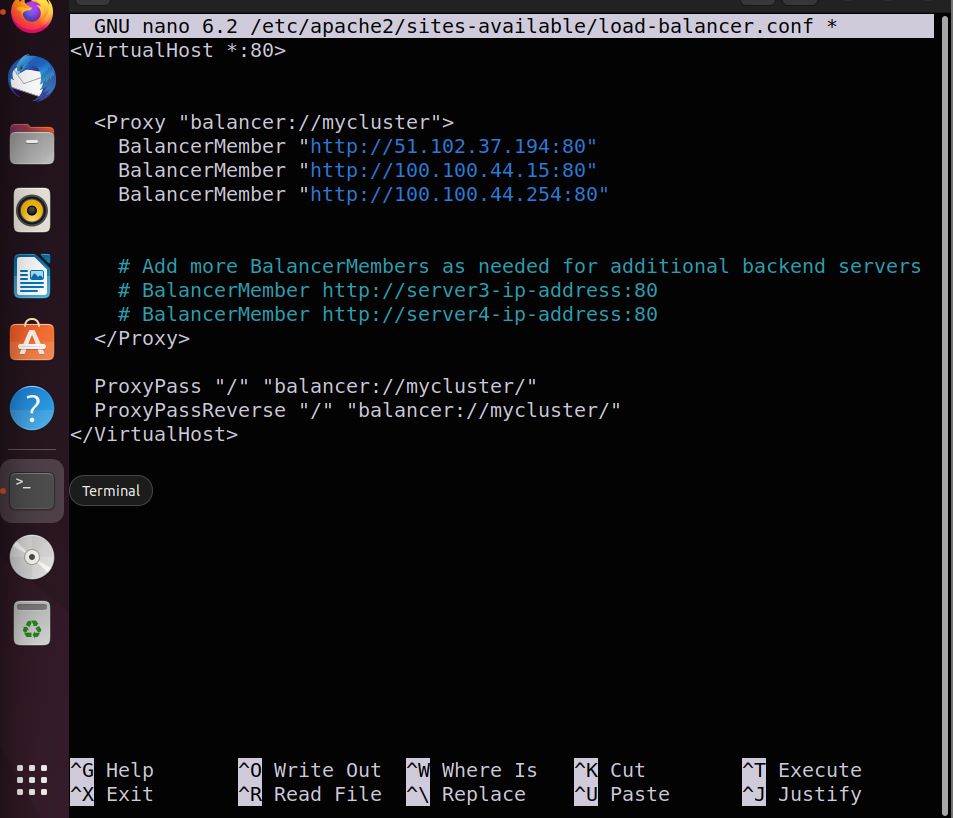
# BalancerMember http://server4-ip-address:80

</Proxy>

ProxyPass "/" "balancer://mycluster/" ProxyPassReverse "/" "balancer://mycluster/"

</VirtualHost>

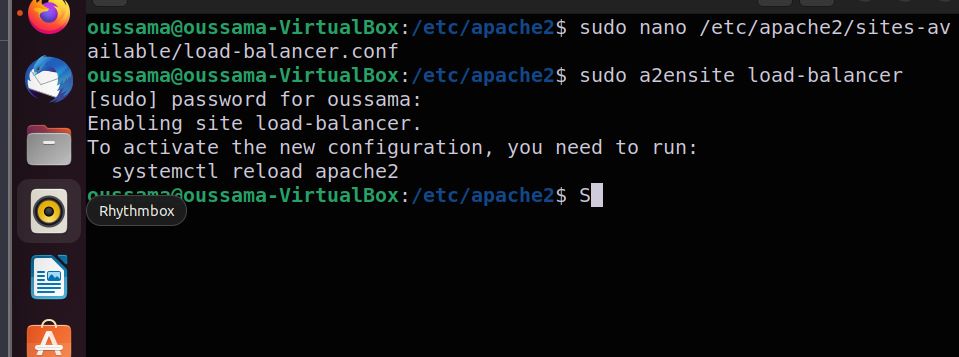
Then save the configuration file and exit the text editor.



## Enable the load balancer site

Enable the load balancer site by running the following command:

**sudo a2ensite load-balancer**

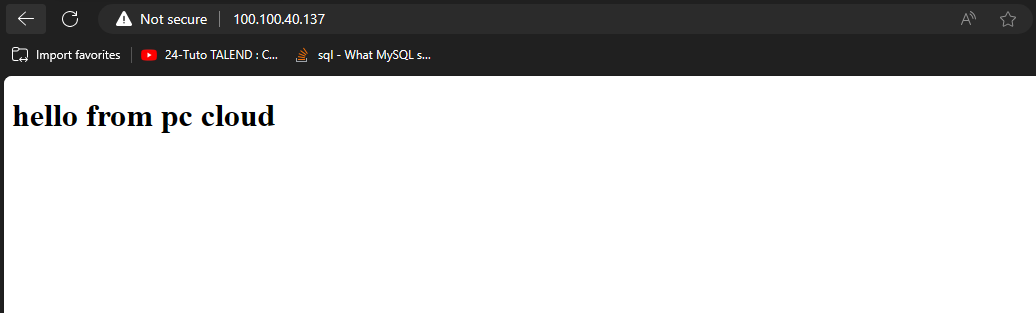


## Restart Apache

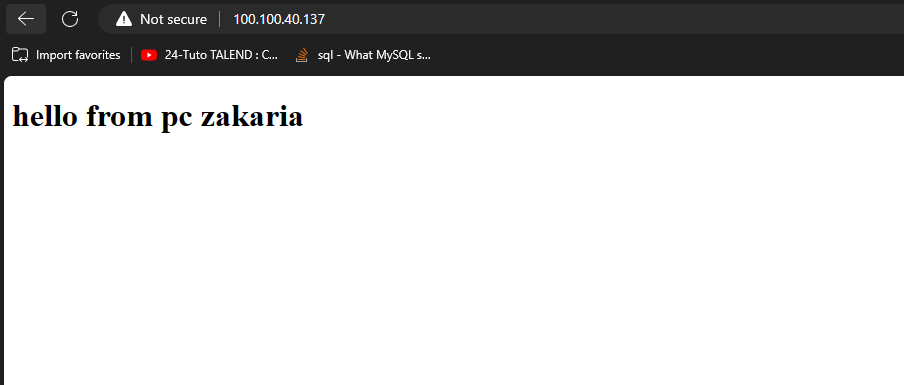
Restart Apache to apply the changes:

**sudo systemctl restart apache2**

## Test in the browser



After refreshing



# Reverser proxy

## Definition

A reverse proxy is a server or software application that sits between client devices and web servers. Unlike a traditional forward proxy, which forwards client requests to the internet, a reverse proxy accepts requests from clients on behalf of one or more backend servers and then forwards those requests to the appropriate server to retrieve the requested content. The response from the server is then sent back to the client through the reverse proxy.

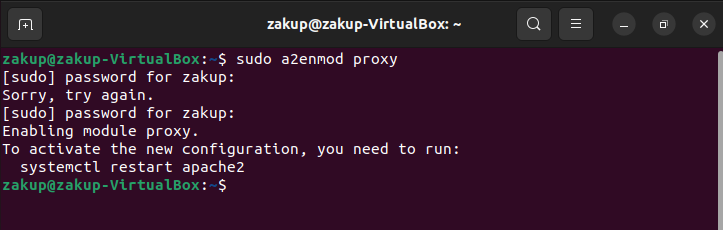
To configure a reverse proxy with Apache2 on Linux, you can follow these steps:

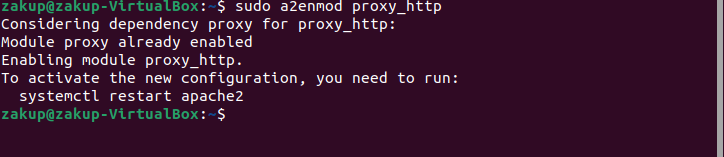
## Enable the necessary Apache modules

Start by enabling the required Apache modules for proxying and reverse proxying. Run the following commands to enable the modules:

**sudo a2enmod proxy**

**sudo a2enmod proxy\_http**



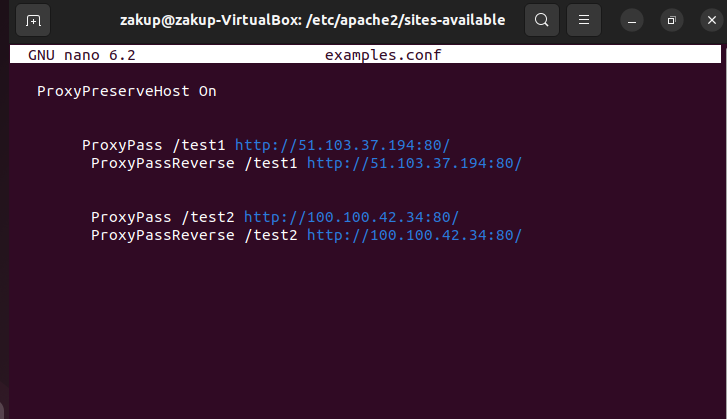


## Configure the reverse proxy

Open the Apache configuration file for the site where you want to set up the reverse proxy. This file is typically located at **/etc/apache2/sites-available/example.conf**, where **"example"** represents the name of your site or virtual host. Use any text editor to edit the file:

**sudo nano /etc/apache2/sites-available/example.conf**

**ProxyPreserveHost On ProxyPass /test1 http://51.103.37.194:80/ ProxyPassReverse /test1 http://51.103.37.194:80/ ProxyPass /test2 http://100.100.42.34:80/ ProxyPassReverse /test2 http://100.100.42.34:80/**



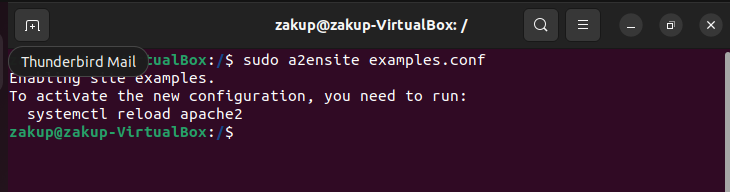
Save the configuration file and exit the text editor.

## Enable the site:

Enable the site configuration by creating a symbolic link from the configuration file in the sites-available directory to the sites-enabled directory.

Run the following command:

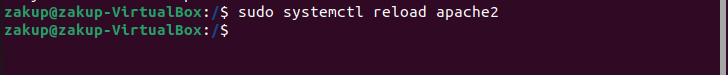
**sudo a2ensite examples.conf**



## Restart Apache

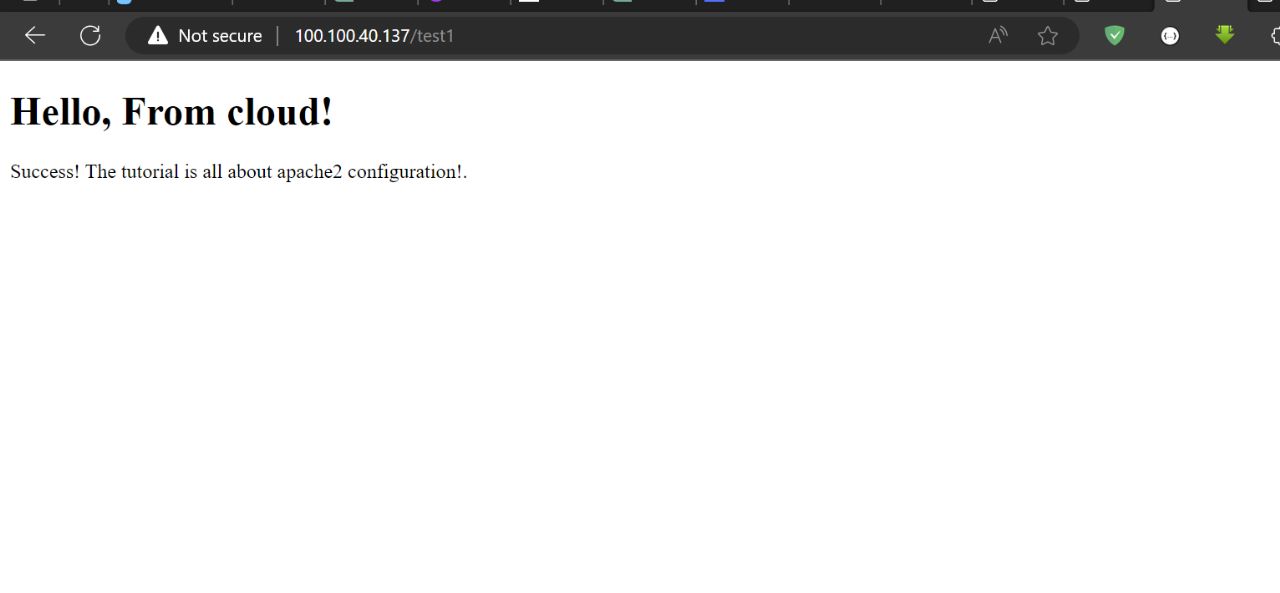
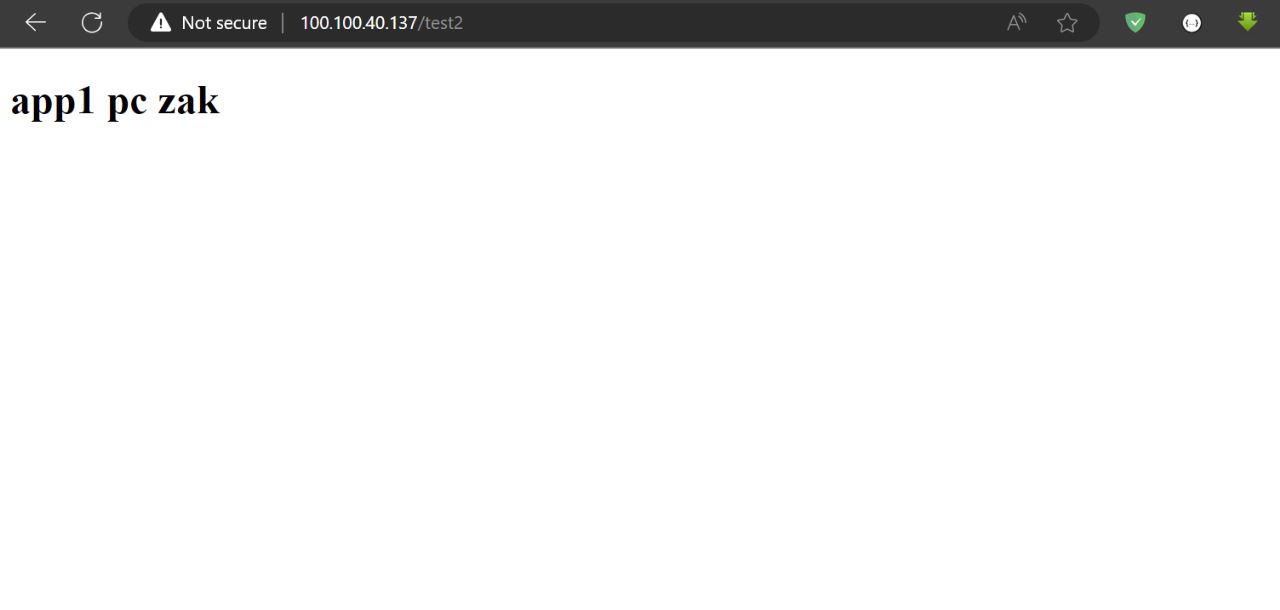
After making any changes to the configuration or file permissions, restart the Apache service using the following command:

**sudo systemctl reload apache2**



This ensures that any modifications take effect.

## Test on your browser



# SSL

## Definition

SSL stands for Secure Sockets Layer. It is a standard security technology used to establish an encrypted connection between a web server and a web browser. SSL ensures that the data transmitted between the server and the browser remains private and cannot be intercepted or tampered with by unauthorized parties.

Let's Encrypt is a non-profit certificate authority that provides free SSL/TLS certificates to enable website owners to secure their websites with HTTPS. It was launched in 2016 with the aim of making it easier for website owners to obtain and deploy SSL certificates.

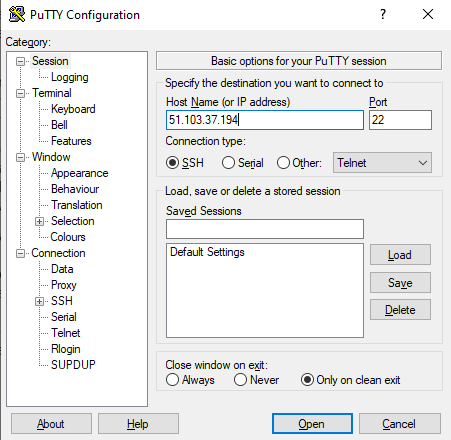
Let's Encrypt has certain requirements and challenges that website owners must fulfill to obtain SSL certificates. These requirements are in place to ensure the authenticity and security of the certificate issuance process.

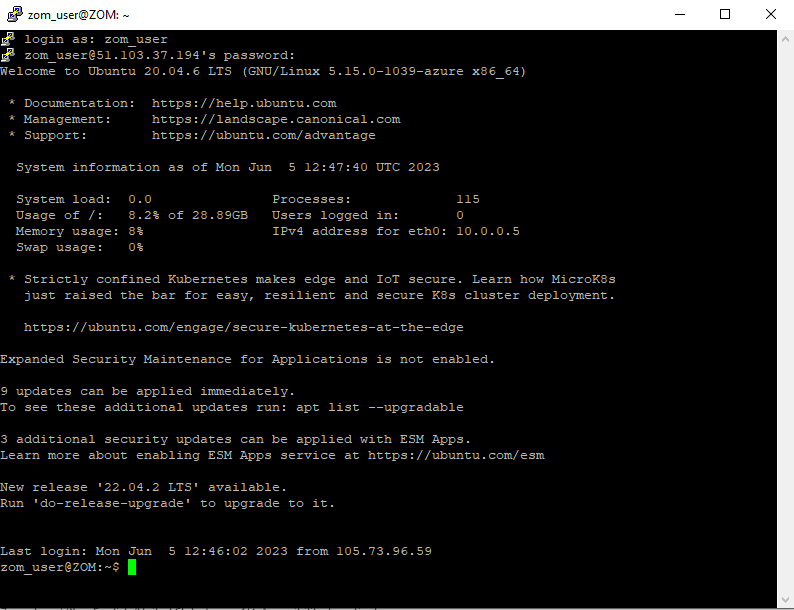
Certbot is a widely used and popular open-source software tool designed to simplify the process of obtaining, renewing, and managing SSL/TLS certificates from Let's Encrypt. It is developed by the Electronic Frontier Foundation (EFF) and is compatible with various operating systems and web servers.

Certbot automates the entire certificate management process by providing an easy-to-use command-line interface (CLI) and integration with popular web servers such as Apache and Nginx. It handles the certificate issuance, renewal, and installation steps, allowing website owners to obtain and maintain SSL certificates without much manual intervention.

## SSH into the server

SSH into the server running your HTTP website as a user with sudo privileges.

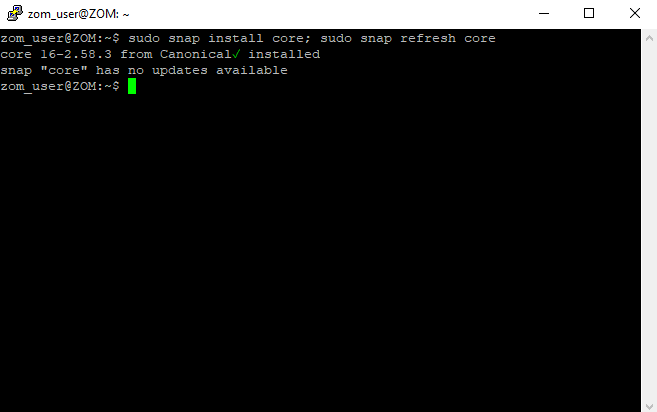




## Ensure that your version of snapd is up to date

Run the following command:

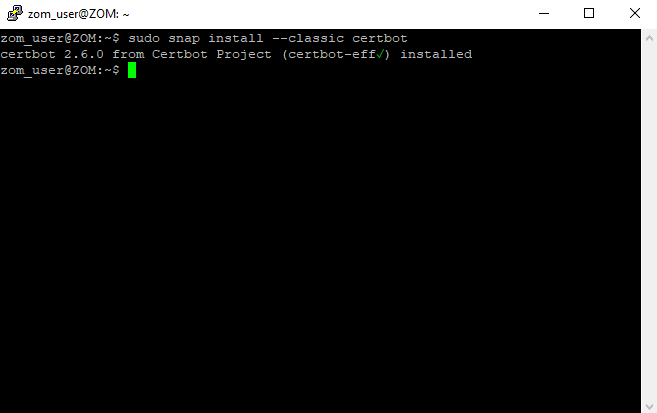
**sudo snap install core; sudo snap refresh core**



## Install Certbot

Run the following command:

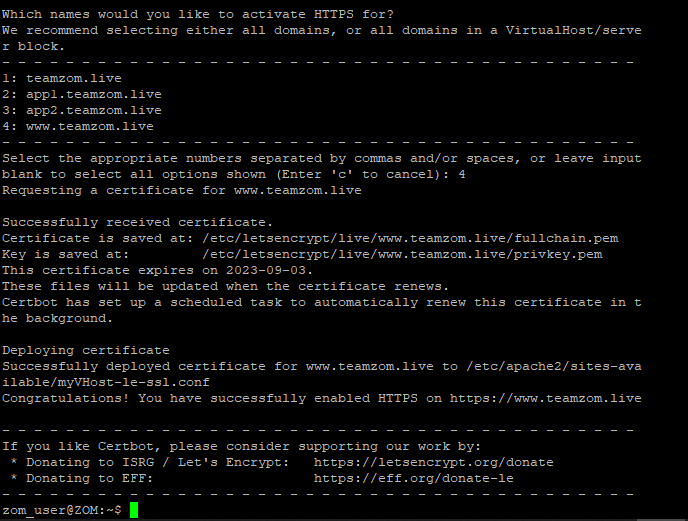
**sudo snap install --classic certbot**



## Install the certificate

Run the following command:

**sudo certbot --apache**



## Test your domaine

