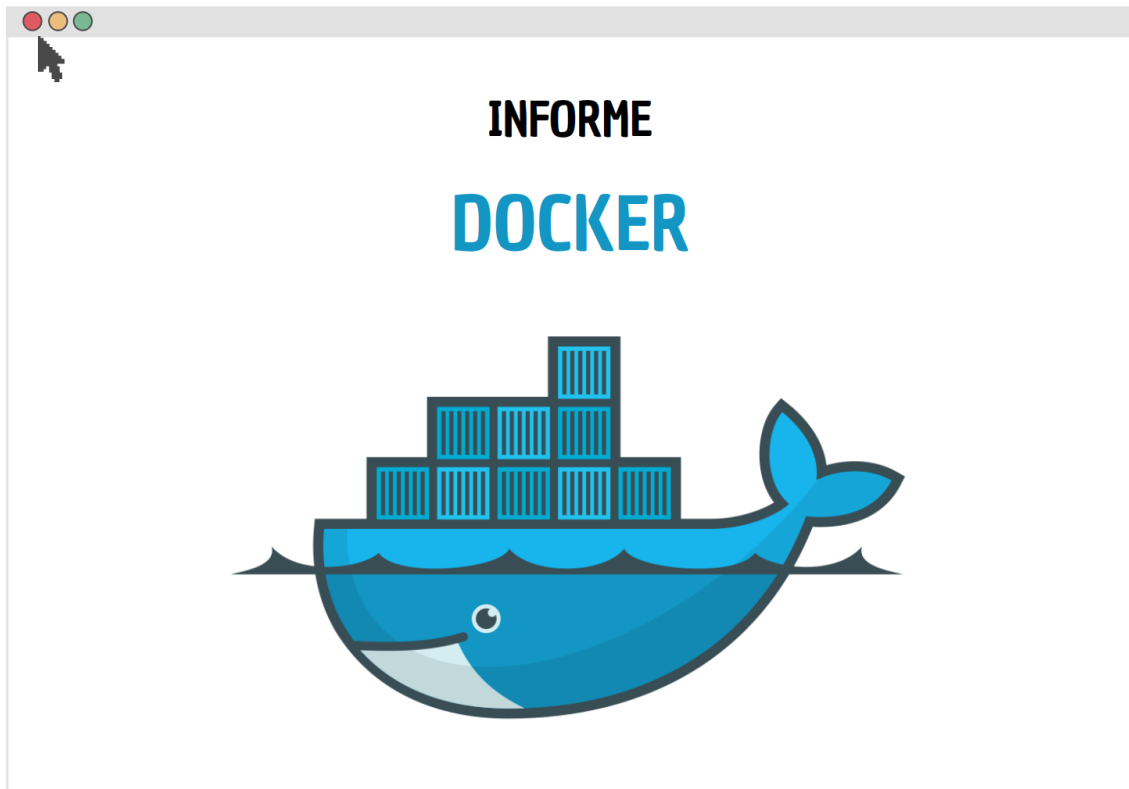




# PROGRAMACIÓN WEB 2



**Profesor(a):**

Carlo Jose Luis Corrales Delgado

**Estudiantes:**

Mamamni Anahua, Victor Narciso  
Mamani Huarsaya, Jorge Luis

30 de abril, 2024

## Preparando el espacio de trabajo

Empezamos descargando la imagen ubuntu 20.04 de [https://hub.docker.com/\\_/ubuntu](https://hub.docker.com/_/ubuntu) Docker Hub

```
> docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
> docker pull ubuntu:20.04
20.04: Pulling from library/ubuntu
d4c3c94e5e10: Pull complete
Digest: sha256:874aca52f79ae5f8258faff03e10ce99ae836f6e7d2df6ecd3da5c1cad3a912b
Status: Downloaded newer image for ubuntu:20.04
docker.io/library/ubuntu:20.04
> docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
ubuntu 20.04 2abc4dfd8318 6 days ago 72.8MB
>
```

Figure 1: Instalando la imagen ubuntu:20.04

Despues creamos un contenedor a partir de esta imagen y lo iniciamos inmediatamente. Podemos lograr hacer esta acciones por separado usando los comandos *create* y seguidamente *start*. Sin embargo al no ejecutar nada dentro del contenedor, este se detendrá inmediatamente.

Para el primer inicio del contenedor, podemos usar el comando *run*, aprovechando tambien realizar el **Port Mapping** de la maquina host a los puertos del contenedor. Tambien para poder interactuar con el contenedor le pasamos los argumentos *-it* para que habra una terminal interactiva y dentro de esta ejecutamos el interprete de comandos */bin/bash*

```
> docker run --name pw2_lab01 -p 8084:80 -p 8085:3306 -p 8086:22 -it ubuntu:20.04 /bin/bash
root@8813c0ebbee2:/# apt update
Get:1 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Get:2 http://archive.ubuntu.com/ubuntu focal InRelease [265 kB]
Get:3 http://security.ubuntu.com/ubuntu focal-security/main amd64 Packages [3595 kB]
Get:4 http://archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:5 http://archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
Get:6 http://archive.ubuntu.com/ubuntu focal/multiverse amd64 Packages [177 kB]
Get:7 http://archive.ubuntu.com/ubuntu focal/main amd64 Packages [1275 kB]
Get:8 http://archive.ubuntu.com/ubuntu focal/universe amd64 Packages [11.3 MB]
Get:9 http://security.ubuntu.com/ubuntu focal-security/multiverse amd64 Packages [29.8 kB]
Get:10 http://security.ubuntu.com/ubuntu focal-security/universe amd64 Packages [1201 kB]
```

Figure 2: Creando y ejecutando el contenedor

Ahora procedemos ha hacer la instalación de un editor de texto necesario para editar código (neovim), un servidor web que implemente el protocolo HTTP y HTTPS (Apache), unos lenguajes de scripting (perl, python), un sistema de gestion de base de datos (MariaDB) y finalmente un programa servidor que implemente el protocolo SSH (openssh)

```
root@8813c0ebbee2:/# apt install neovim apache2 perl python mariadb-server openssh-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
Note, selecting 'python-is-python2' instead of 'python'
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils ca-certificates dbus distro-info-data dmsetup file galera-3 gawk
  gir1.2-glib-2.0 iproute2 krb5-locales libaio1 libapparmor1 libapr1 libaprutil1 libaprutil1-dbd-sqlite3
  libaprutil1-ldap libargon2-1 libasn1-8-heimdal libatm1 libbrotli1 libbsd0 libcap2 libcap2-bin libcbor0.6
  libcgi-fast-perl libcgi-pm-perl libconfig-inifiles-perl libcryptsetup12 libcurl4 libdbd-mysql-perl libdbi-perl
  libdbus-1-3 libdevmapper1.02.1 libedit2 libelf1 libencode-locale-perl libexpat1 libfcgi-perl libfido2-1
  libgdbm-compat4 libgdbm6 libgirepository-1.0-1 libgl1-0 libgl1-0-data libgssapi-krb5-2 libgssapi3-heimdal
  libhcrypto4-heimdal libheimbase1-heimdal libheimntlm0-heimdal libhtml-parser-perl libhtml-tagset-perl
  libhtml-template-perl libhttp-date-perl libhttp-message-perl libhx509-5-heimdal libice6 libicu66 libio-html-perl
  libip4tc2 libjansson4 libjson-c4 libk5crypto3 libkeyutils1 libkmod2 libkrb5-26-heimdal libkrb5-3 libkrb5support0
  libldap-2.4-2 libldap-common liblua5.2-0 liblua5.2-dev liblua5.2-luajit liblua5.2-luajit-common liblua5.2-luajit-tests
  libmagic-mgc libmagic1 libmnl0 libmpdec2 libmpfr6 libmsgpack2 libmysqlclient21 libnghttp2-14 libnss-systemd
  libnss3 libnss3-dev libperl5.30 libpopt0 libpsl5 libpython2-stdlib libpython2.7-minimal
```

Figure 3: Instalando los programas para crear y administrar nuestra servidor web

Verificamos que la instalación haya sido un éxito

```
Enabling conf serve-cgi-bin.
Enabling site 000-default.
Created symlink /etc/systemd/system/multi-user.target.wants/apache2.service → /lib/systemd/system/apache2.service.
Created symlink /etc/systemd/system/multi-user.target.wants/apache-htcacheclean.service → /lib/systemd/system/apache-htcacheclean.service.
invoke-rc.d: could not determine current runlevel
invoke-rc.d: policy-rc.d denied execution of start.
Processing triggers for systemd (245.4-4ubuntu3.23) ...
Processing triggers for libc-bin (2.31-0ubuntu9.15) ...
Processing triggers for ca-certificates (20230311ubuntu0.20.04.1) ...
Updating certificates in /etc/ssl/certs...
0 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...
done.
root@8813c0ebbee2:/#
```

Figure 4: La instalación ha sido un éxito

Podemos verificar el estado de los programas servidor que hemos instalado, estos servicios son bien conocidos como **Demonios** ya que son bucles infinitos, que siempre están escuchando. Los servidores siempre se deben de activar. Incluso podemos configurar su activación automática cuando arranque el sistema operativo.

```
root@8813c0ebbee2:/# /etc/init.d/apache2 status
* apache2 is not running
root@8813c0ebbee2:/# /etc/init.d/mysql status
* MariaDB is stopped.
root@8813c0ebbee2:/# /etc/init.d/ssh status
* sshd is not running
root@8813c0ebbee2:/#
```

Figure 5: Estado de los programas servidor

Por el momento solo vamos a iniciar el servidor web Apache y el sistema de gestión de base de datos MariaDB.

```
root@8813c0ebbee2:/# /etc/init.d/apache2 start
* Starting Apache httpd web server apache2
AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 172.17.0.2. Set the '
ServerName' directive globally to suppress this message
*
root@8813c0ebbee2:/# /etc/init.d/mysql start
* Starting MariaDB database server mysqld
root@8813c0ebbee2:/# [ OK ]
```

Figure 6: La instalación ha sido un éxito

Podemos probar el funcionamiento del servidor web de nuestro contenedor en nuestro navegador de nuestra maquina host poniendo la direccion IP del contenedor, o simplemente instalando y usando el programa curl que es una herramienta en la terminal.

```
root@8813c0ebbee2:/# curl 172.17.0.2

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.
dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <!--
    Modified from the Debian original for Ubuntu
    Last updated: 2016-11-16
    See: https://launchpad.net/bugs/1288690
  -->
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <title>Apache2 Ubuntu Default Page: It works</title>
    <style type="text/css" media="screen">
      * {
        margin: 0px 0px 0px 0px;
        padding: 0px 0px 0px 0px;
      }

      body, html {
        padding: 3px 3px 3px 3px;
      }
    </style>
  </head>
  <div style="text-align: center;">
    <img alt="Ubuntu logo" data-bbox="138 425 165 445" style="height: 40px; margin-bottom: 10px;"/>
    Apache2 Ubuntu Default Page: It works
  </div>
</html>
```

Figure 7: Archivo *index.html* ubicado en el directorio por defecto */var/www/html*

Bien, seguimos con la configuración de los directorios por defecto en los cuales Apache escucha las solicitudes por defecto. Es una mala práctica que se use el directorio por defecto */var/www/html*. Siempre se ha recomendado crear tu propio directorio y como root poder restringir el acceso según el contexto.

Ahora vamos a configurar el servidor web Apache para que pueda ejecutar CGI's. Apache trae consigo dos módulos nativos *mod-cgi* y *mod-cgid*. Además la ejecución solo se da en el directorio */lib/cgi-bin*

Vamos a crear un test en Python para ver el funcionamiento. Una vez que creamos el archivo *.py*, tenemos que otorgarle el permiso de ejecución, ya que es un script que Apache debe de ejecutar.

```
#!/usr/bin/env python3

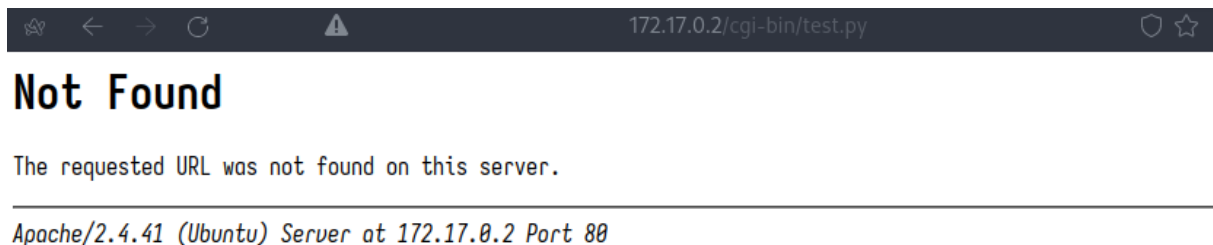
print("Content-Type: text/html\n")

print("""
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>CGI con Python</title>
</head>
<body>
  <h1>¡Hola desde un script CGI en Python!</h1>
</body>
</html>
""")
~
```

Figure 8: Archivo en `/lib/cgi-bin/test.py` y otorgar permisos

```
root@8813c0ebbee2:/# cd /lib/cgi-bin/
root@8813c0ebbee2:/lib/cgi-bin# nvim test.py
root@8813c0ebbee2:/lib/cgi-bin# chmod 755 test.py
root@8813c0ebbee2:/lib/cgi-bin# ls
test.py
root@8813c0ebbee2:/lib/cgi-bin#
```

Figure 9: CGI con python

Figure 10: Apache solo escucha en `/var/www/html`

Como vemos falta hacer alguna configuración para que Apache pueda realizar esta funcionalidad. De hecho, debemos de habilitar los módulos CGI. Una vez activado el módulo, nosotros debemos de reiniciar Apache.

```
root@8813c0ebbee2:/lib/cgi-bin# a2enmod cgi
Module cgi already enabled
root@8813c0ebbee2:/lib/cgi-bin# /etc/init.d/apache2 restart
* Restarting Apache httpd web server: apache2
AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 172.17.0.2. Set the '
ServerName' directive globally to suppress this message
[ OK ]
root@8813c0ebbee2:/lib/cgi-bin# curl 172.17.0.2/cgi-bin/test.py

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>CGI con Python</title>
</head>
<body>
  <h1>¡Hola desde un script CGI en Python!</h1>
</body>
</html>

root@8813c0ebbee2:/lib/cgi-bin#
```

Figure 11: Reiniciar el servidor

Ahora si podemos estar seguros que los CGIs se van a ejecutar



Figure 12: Test exitoso

## Cambiando a directorios personalizados

Para hacer los cambios, nosotros debemos modificar los archivos de configuración de apache. Apache maneja el concepto de Virtual Host que nos permite tener varias aplicaciones web en solo un servidor web físico. Esto se logra configurando Apache para que responda a diferentes nombres de dominio (o direcciones IP) y sirva contenido específico para cada uno de ellos.

Las bloques de configuración que vamos a realizar van a definir el nombre de dominio, el directorio raíz y otras opciones de configuración. Lo que vamos a hacer es crear un archivo `/etc/apache2/sites-available/aplicaciones_web` y aquí colocamos las configuraciones.

```
root@8813c0ebbee2:/# cd /etc/apache2/sites-available/
root@8813c0ebbee2:/etc/apache2/sites-available# ls
000-default.conf  default-ssl.conf
root@8813c0ebbee2:/etc/apache2/sites-available# cp 000-default.conf aplicaciones_web.conf
root@8813c0ebbee2:/etc/apache2/sites-available# nvim aplicaciones_web.conf
root@8813c0ebbee2:/etc/apache2/sites-available# cd /
root@8813c0ebbee2:/# mkdir -p aplicaciones_web/proyecto-final-pweb1
root@8813c0ebbee2:/#
```

Figure 13: Creación del archivo de configuración

```
# Agregamos el alias, cada vez que ingresen a /aplicaciones_web los redireccionará a ...
Alias /aplicaciones_web "/aplicaciones_web/proyecto-final-pweb1"

# El Directory lo ponemos afuera para que se aplique tambien a /aplicaciones_web
<Directory "/aplicaciones_web/proyecto-final-pweb1">
    Options Indexes FollowSymLinks Multiviews
    AllowOverride All
    Require all granted
</Directory>

<VirtualHost *:80>
    ServerName aplicaciones_web.com

    ServerAdmin webmaster@localhost
    DocumentRoot /aplicaciones_web/proyecto-final-pweb1

    # Default
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    # For most configuration files from conf-available/, which are
    # enabled or disabled at a global level, it is possible to
    # include a line for only one particular virtual host. For example the
    # following line enables the CGI configuration for this host only
    # after it has been globally disabled with "a2disconf".
    #Include conf-available/serve-cgi-bin.conf
</VirtualHost>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

Figure 14: configuración del Virtual Host

Una vez que hicimos las respectivas configuraciones, ahora necesitamos activar el sitio web, activamos el modulo **mod-rewrite** que se encarga de realizar las redirecciones de solicitudes.

```
root@8813c0ebbee2:/etc/apache2/sites-available# a2ensite aplicaciones_web.conf
Enabling site aplicaciones_web.
To activate the new configuration, you need to run:
    service apache2 reload
root@8813c0ebbee2:/etc/apache2/sites-available# a2enmod rewrite
Enabling module rewrite.
To activate the new configuration, you need to run:
    service apache2 restart
```

Figure 15: Habilitamos el sitio web configurado

Ahora necesitamos que los CGIs se utilicen en directorios especificos. Para ello nosotros nos vamos al directorio `/etc/apache2/conf-available` donde vamos a crear una configuración para que apache pueda ejecutar los CGIs en el directorio, por ejemplo, `/aplicaciones_web`

```
<Directory "/aplicaciones_web">
    Options +ExecCGI
    AddHandler cgi-script .cgi .pl .rb .py
</Directory>

~
```

Figure 16: Archivo de configuración

Por ultimo habilitamos la configuración y como siempre, reiniciamos el servidor Apache

```
root@8813c0ebbee2:/etc/apache2/conf-available# a2enconf cgi_enabled.conf
Enabling conf cgi_enabled.
To activate the new configuration, you need to run:
    service apache2 reload
root@8813c0ebbee2:/etc/apache2/conf-available# /etc/init.d/apache2 restart
* Restarting Apache httpd web server apache2
AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 172.17.0.2. Set the '
ServerName' directive globally to suppress this message
[ OK ]
root@8813c0ebbee2:/etc/apache2/conf-available#
```

Figure 17: Habilitando las configuración

## Agregando un usuario para que pueda controlar el sitio web creado

Podemos agregar a un usuario para que se encargue de administrar este sitio web. Entonces lo primero que haremos es crear el usuario en el sistema y después cambiaremos el propietario y el grupo del directorio raíz */aplicaciones\_web* y todos los directorios y archivos dentro de este.

```
root@8813c0ebbee2:/# adduser pw2
Adding user `pw2' ...
Adding new group `pw2' (1000) ...
Adding new user `pw2' (1000) with group `pw2' ...
Creating home directory `/home/pw2' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for pw2
Enter the new value, or press ENTER for the default
    Full Name []:
    Room Number []:
    Work Phone []:
    Home Phone []:
    Other []:
Is the information correct? [Y/n] y
root@8813c0ebbee2:/# chown -R pw2:www-data /aplicaciones_web/
root@8813c0ebbee2:/# cd /aplicaciones_web/
root@8813c0ebbee2:/aplicaciones_web# ls -la
total 12
drwxr-xr-x 3 pw2  www-data 4096 May  3 21:54 .
drwxr-xr-x 1 root root    4096 May  3 21:54 ..
drwxr-xr-x 2 pw2  www-data 4096 May  3 21:54 proyecto-final-pweb1
root@8813c0ebbee2:/aplicaciones_web#
```

Figure 18: Agregado un usuario y cambiando propietario

Ahora podemos probar el servicio SSH, para ello lo vamos a activar y nos conectaremos desde la maquina host



```
root@8813c0ebbee2:/aplicaciones_web# /etc/init.d/ssh start
* Starting OpenBSD Secure Shell server sshd [ OK ]
root@8813c0ebbee2:/aplicaciones_web# /etc/init.d/ssh status
* sshd is running
root@8813c0ebbee2:/aplicaciones_web#
```

```
> ssh pw2@172.17.0.2 55ms ~ [ ]
ssh: connect to host 172.17.0.2 port 22: Connection refused
> ssh pw2@172.17.0.2 66ms ~ [ ]
The authenticity of host '172.17.0.2 (172.17.0.2)' can't be established.
ED25519 key fingerprint is SHA256:zIfbxtJ6xLWYN9a03LQin96qRWbLS56gGYHzFMJCH9g.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '172.17.0.2' (ED25519) to the list of known hosts.
pw2@172.17.0.2's password:
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 6.8.5-arch1-1 x86_64)

* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:       https://ubuntu.com/pro

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

pw2@8813c0ebbee2:~$
```

Figure 19: El servicio de SSH esta habilitado