

Diseño e implementación de una base de datos para control de licencias.

Autor: Diego Carrillo Santamaría

MAYO 2020

TABLA DE CONTENIDOS

| | | |
|-----------|--|-----------|
| 1. | PRÓLOGO..... | 3 |
| 2. | OBJETIVOS..... | 4 |
| 2.1 | Objetivo General..... | 4 |
| 2.2 | Objetivos Específicos..... | 4 |
| 3. | IMPLEMENTACIÓN DE LA PILA DE DATOS LAMP..... | 4 |
| 3.1 | Linux..... | 4 |
| 3.2 | Apache..... | 5 |
| 3.3 | MySQL..... | 6 |
| 3.4 | PHP..... | 6 |
| 4. | INSTALACIÓN DE PAQUETES DE SOFTWARE EN EL SISTEMA. | 7 |
| 4.1 | Instalación de Linux Ubuntu 16.04..... | 7 |
| 4.2 | Instalación del paquete Apache para Linux y desbloqueo del Firewall..... | 8 |
| 4.3 | Instalación del paquete MySQL para Linux..... | 11 |
| 4.4 | Instalación del paquete PHP para Linux..... | 12 |
| 4.5 | Pruebas de conectividad y funcionamiento de PHP en el servidor..... | 14 |
| 5. | SERVIDOR DE LICENCIAS..... | 15 |
| 5.1 | Requisitos de uso..... | 15 |
| 5.2 | Configuración de la Base de datos desde PHPMysqlAdmin..... | 16 |
| 5.3 | Ejecución de Scripts para establecer la comunicación..... | 18 |
| 6. | CONCLUSIONES..... | 20 |
| 7. | BIBLIOGRAFÍA..... | 20 |

TABLA DE FIGURAS

| | |
|--|----|
| Figura 1: Logo Sistema operativo Linux Ubuntu 16.04 (Fuente: tipsonubuntu.com) ----- | 5 |
| Figura 2: Logo del Servidor Apache basado en el protocolo HTTP (Fuente: Apache Foundation)----- | 5 |
| Figura 3: MySQL Development Series Logo (Fuente: MySQL™) ----- | 6 |
| Figura 4: PHP Protocol Logo. (Fuente: PHP Open Source Initiative) ----- | 7 |
| Figura 5: Información del sistema mostrada en pantalla una vez instalado el sistema operativo. ----- | 8 |
| Figura 6: Mensaje de error con ServerName indefinido. ----- | 9 |
| Figura 7: Definición del IP en el repositorio raíz /etc. ----- | 9 |
| Figura 8: Revisión de sintaxis en nuestro repositorio base de Apache.----- | 9 |
| Figura 9: Aplicaciones habilitadas luego de dar permisos de tráfico HTTP y HTTPS----- | 10 |
| Figura 10: Información del sistema mostrando los puertos habilitados para el trafico de información. ----- | 10 |
| Figura 11: Acceso al servidor Apache (Imagen de la habilitación).----- | 11 |
| Figura 12: Validación del Plugin de la Contraseña para MySQL. ----- | 12 |
| Figura 13: Modificación del archivo /etc/apache2/mods-enabled/dir.conf----- | 13 |
| Figura 14: Respuesta del sistema para comprobar el estado de systemctl. ----- | 13 |
| Figura 15: Script info.php----- | 14 |
| Figura 16: Información del servidor desde la perspectiva PHP.----- | 14 |
| Figura 17: Organización de la estructura de la Base de Datos. ----- | 16 |
| Figura 18: License_Check Database. Source PHPMyAdmin. ----- | 16 |
| Figura 19: connection_register. Source PHPMyAdmin. ----- | 17 |
| Figura 20: seed_request. Source PHPMyAdmin. ----- | 17 |
| Figura 21: Users. Source PHPMyAdmin.----- | 17 |

1. PRÓLOGO

Este informe tiene como principal objetivo describir las principales actividades que se realizaron para la implementación de una base de datos para el control de licencias para las estaciones de los robots MINI, como parte de las actividades que se debían desarrollar para los temas de investigación agendados en el laboratorio de robots sociales de la Universidad Carlos III de Madrid.

Para este trabajo se ha utilizado el sistema operativo Ubuntu (versión 16.04), uno de los sistemas operativos basados en Linux más ampliamente utilizado en la actualidad entre cosas gracias a su buen desempeño y rendimiento, también porque se encuentra disponible de forma gratuita por lo que todos pueden tener acceso a él, y a su seguridad con respecto al malware, por otro lado, existe una gran cantidad de documentación que ha ido fomentando la comunidad de desarrolladores y por su puesto, por ser de código libre.

Por otro lado, se debe señalar que la actividad que se ha realizado, podrá ser utilizada para controlar licencias de operación de los robots MINI de forma remota, para ello fue necesario configurar una estación que tendrá la función de *MASTER* en la que se implementaron todas las funcionalidades de *SERVIDOR WEB* por medio de Apache y MySQL, la cual será la base de datos en donde se almacenarán la información de acceso elemental de las estaciones de los robots, para poder administrar el uso de las estaciones remotamente.

A continuación, se procederá a detallar todas las actividades que se desarrollaron en la implementación de este trabajo, así como una referencia paso a paso con los detalles de los comandos, respuesta del sistema y resultados experimentales en general.

2. OBJETIVOS

2.1 Objetivo General

- Diseñar y desarrollar una base de datos utilizando la pila LAMP (Linux, Apache, MySQL y PHP) en Ubuntu para control de licencias de las estaciones MINI del laboratorio de Robótica Social de la Universidad Carlos III de Madrid.

2.2 Objetivos Específicos.

- Configurar servicios de Servidor Web en Linux por medio de Apache y MySQL
- Generar una base de datos utilizando la herramienta de PHPMYADMIN utilizando un navegador web.
- Comprobar la correcta conectividad entre Cliente-Servidor entre las distintas estaciones.

3. IMPLEMENTACIÓN DE LA PILA DE DATOS LAMP.

Se le llama pila “LAMP” a un grupo de software de código libre que se instala para habilitar un servidor que se emplea para alojar sitios y aplicaciones web dinámicas. Según el desarrollador Brennen Bearnes, *“el término es un acrónimo que representa un sistema operativo Linux con un servidor Apache. Los datos del sitio son almacenados en base de datos MySQL y el contenido dinámico es procesado con PHP”* (Bearnes, 2016)¹, de ahí el nombre LAMP debida a la combinación de los sistemas LINUX + Apache + MySQL + PHP. A continuación, daremos un repaso de las principales particularidades de cada uno de dichos sistemas.

3.1 Linux.

Es un sistema operativo lanzado en 1991 del tipo UNIX de propósito general, el cual se desarrolló desde los tiempos del i386 y que se extendió su uso posteriormente con otros procesadores. Las principales capacidades tienen que ver con el multiprocesado, idoneidad para realizar múltiples tareas y soportar diversos usuarios, y lo más importante no se tiene que pagar altas cantidades por licencias de usuario.

Según Arturo Fernández Montoro, en su libro “Cámbiate a Linux”:

Linux es software libre lo que significa que podemos ejecutarlo con cualquier propósito, estudiar cómo funciona, copiarlo cuantas veces queramos y distribuirlo a quién deseemos y en el modo que creamos más conveniente. Tal y como veremos, ello implica una serie de claras ventajas frente a Windows, el cual es software privativo que además requiere de un desembolso económico para su utilización. (Fernández Montoro, 2011)².

Para el caso especial de nuestro desarrollo se ha utilizado la versión Ubuntu 16.04 cuya plataforma es la que se ha utilizado en el laboratorio de Robots Sociales de la Universidad UC3M y cuya ventaja principal es el hecho de ser un Sistema Operativo del tipo Open Source, además de encontrarse libre para los usuarios cuenta con un gran soporte ofrecida por la comunidad de desarrolladores, lo cual conviene por la cantidad de información disponible en la web.



Figura 1: Logo Sistema operativo Linux Ubuntu 16.04 (Fuente: tipsonubuntu.com)

3.2 Apache

Apache es un popular servidor multiplataforma, se destaca dentro de las principales características que es Open Source, es decir, que su código se ha diseñado para que los usuarios puedan acceder a él de modo gratuito. Como bien se refiere Mohammed J. Kabir en su libro La Biblia Servidor Apache 2: “Apache es un servidor altamente configurable de diseño modular, muy sencillo para ampliar las capacidades de un servidor Web, que cuenta con una tecnología gratuita de código abierto que funciona en Linux y otros sistemas de Unix”. (Kabir, 2002)³.

Esto ha convertido hoy a Apache en uno de los servidores web mas ampliamente utilizados en el mundo del desarrollo web y aplicaciones, parte de ese éxito es debida a su robustez y sus multiples aplicaciones y funciones, así como que opera bajo *Freeware* bajo licencia GNU. Esto ha hecho que desarrolladores que gestionan su propio host utilicen estos servicios. Con Apache cualquiera puede montar su propio servidor Web y hacer uso de él.



Figura 2: Logo del Servidor Apache basado en el protocolo HTTP (Fuente: Apache Foundation)

Esta plataforma tiene mucha versatilidad a nivel de configuración, en donde el usuario tiene la opción de adaptar su desarrollo a sus necesidades por medio del uso de bases de datos de autenticación en las cuales se puede añadir seguridad de contenido y gestión de las mismas, a pesar de que en su momento recibió críticas por la falta de una interfaz gráfica que amparara e impulsara su configuración.

3.3 MySQL

MySQL fue creado por una compañía sueca llamada *MySQL AB* en 1995. Es un lenguaje de programación informático para la creación y gestión de bases de datos de código abierto, en donde se afirma sin lugar a dudas que es la más popular del mercado hasta la fecha, esto gracias a su fiabilidad, rendimiento, y a la facilidad de uso que ha sido comprobado y verificado por la comunidad de desarrolladores, razón por la que hoy en día se utiliza a todo nivel, desde aplicaciones sencillas de hogar y pequeñas empresas hasta web de perfil alto, como Facebook, Twitter, YouTube, entre otras.



Figura 3: MySQL Development Series Logo (Fuente: MySQL™)

Se afirma también que este sistema gestiona bases de datos relacionales y multiusuario, y surgió cuando una subsidiaria de Sun Microsystems y ésta a su vez de *Oracle Corporation* desarrolló *MySQL* como software libre en un esquema de licenciamiento dual, bajo la GNU GPL creada por Richard Stallman. Sin embargo, en productos privados si se debe comprar una licencia específica, ya que MySQL es patrocinado por una empresa privada que posee los derechos de autor de la mayor parte del código (Castillo Sánchez, 2011)⁴.

3.4 PHP

PHP es el acrónimo *Hypertext Preprocessor*, es un lenguaje de código abierto muy popular lanzado en 1995, especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML. Este es un lenguaje de programación de propósito general de código del lado del servidor originalmente diseñado para el pre-procesado de texto plano en UTF-8. Posteriormente se aplicó al desarrollo web de contenido dinámico, dando un paso evolutivo en el concepto de aplicación en línea, por su carácter de servicio (PHP, 2020)⁵.



Figura 4: PHP Protocol Logo. (Fuente: PHP Open Source Initiative)

La gran ventaja de utilizar PHP aparte de su gran simplicidad, es la versatilidad tanto para el principiante como para un desarrollador profesional avanzado, ya que puede ser implementado en la mayoría de los servidores web y prácticamente en todos los sistemas operativos y plataformas sin costo alguno.

Según datos del 2019, se encuentra instalado en varios millones de sitios web a nivel mundial sumando aproximadamente el 20% de los dominios en Internet y en más de un millón de servidores. Por otro lado, PHP posee una enorme similitud con los lenguajes más comunes de programación estructurada, como *C* y *Perl*, lo que les permite a los desarrolladores crear aplicaciones de todo tipo, desde sencillas hasta las más complejas, con una curva rápida de aprendizaje, permitiendo involucrarse con aplicaciones de contenido dinámico sin tener que aprender una pila de funciones. Ver datos en web.archive.org (Forum, 2013)⁵.

4. INSTALACIÓN DE PAQUETES DE SOFTWARE EN EL SISTEMA.

En este apartado se procederá a detallar las principales actividades que se desarrollaron, así como los comandos que se utilizaron en la implementación, con lo cual se demostrará los resultados que se obtuvieron en la práctica, por otro lado, se debe mencionar que se ha hecho toda la configuración partiendo desde cero, con un ordenador completamente formateado, esto con el fin de estar alineado al principal objetivo que era realizar la implementación de una base de datos.

4.1 Instalación de Linux Ubuntu 16.04

Lo primero que se ha hecho es proceder con la instalación del sistema operativo Linux Ubuntu 16.04 en una estación especialmente dedicada para tales propósitos, para ello se ha seguido el manual del desarrollador *Brennen Bearnes* y *Mitchel Anicas* en su guía de instalación del sistema operativo Linux 16.04 (Anicas, 2016)⁶. En dicho manual se ha configurado el sistema con los parámetros adecuados para poder iniciar el desarrollo, resultado fue el correcto ya que se vio reflejado en la pantalla la siguiente imagen, con lo cual se dio fe a que el sistema estaba listo para ejecutar la implementación que se deseaba.

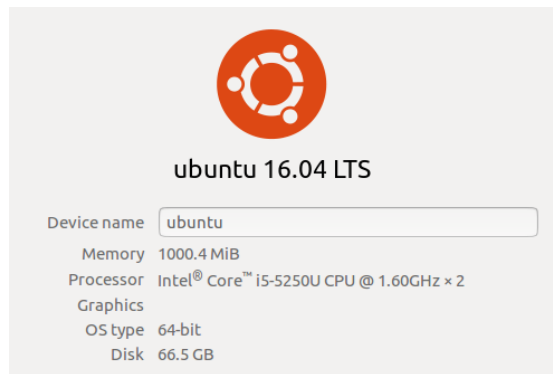


Figura 5: Información del sistema mostrada en pantalla una vez instalado el sistema operativo.

Una vez instalado el paquete del sistema se procedió a hacer la instalación de MySQL en el equipo, acción que fue necesaria para poder desarrollar la base de datos, como se mencionó en la [sección 3](#), lo que se realizó fue implementación integral de varios sistemas que forman la pila de datos LAMP.

4.2 Instalación del paquete Apache para Linux y desbloqueo del Firewall

Para este caso particular nos hemos apoyado de la documentación desarrollada por Brennen Bearnes en su tutorial “*LAMP Stack on Ubuntu*” (Bearnes, Digital Ocean, 2016)⁷, en donde se ha seguido una serie de comandos para dejar instalado en nuestra raíz el paquete Apache, sin dejar de lado una serie de requisitos fundamentales requeridos por el sistema como la necesidad de tener una cuenta de usuario con privilegios de sudo configurados en el servidor.

El servidor Web Apache fácilmente el más popular del mundo desde hace varios años como lo vimos en la [sección 3.2](#), tiene una enorme cantidad de usuarios que van desde estudiantes y profesores hasta empresas, instituciones y centros de desarrollo, el cual ha sido ampliamente utilizado en la historia de la web, lo que hace que sea una gran opción por defecto para montar un sitio web con una base sólida y confiable.

Apache se ha instalado fácilmente desde el gestor de paquetes de Ubuntu, *apt* que nos sirve como gestor de paquetes que permite hacer el proceso con mayor facilidad desde un repositorio mantenido por Ubuntu. Los comandos desarrollados fueron los siguientes:

Instalación

```
$ sudo apt -get update
```

```
$ sudo apt -get install apache2
```

Establecer `ServerName` para evitar los Errores de Sintaxis, se agrega una sola línea al archivo `/etc/apache2/apache2.conf` para suprimir un mensaje de advertencia. Si no se define `ServerName` globalmente, se recibirá una advertencia cuando se compruebe la configuración de Apache para los errores de sintaxis:

```
Output
AH00558: apache2: Could not reliably determine the server's fully qualified
domain name, using 127.0.1.1. Set the 'ServerName' directive globally to
suppress this message
Syntax OK
```

Figura 6: Mensaje de error con `ServerName` indefinido.

Luego se debe abrir el archivo de configuración global para agregar la directiva de `ServerName` por medio del comando:

```
$ sudo nano /etc/apache2/apache2.conf
```

Esto con el fin de definir la dirección IP que utilizara nuestro Web Server, en nuestro caso utilizamos la dirección `192.168.1.228`

```
                                /etc/apache2/apache2.conf
. . .
ServerName dominio_del_servidor_o_IP
```

Figura 7: Definición del IP en el repositorio raíz `/etc`.

Para asegurarnos que se no hay errores de sintaxis escribimos en la consola el comando:

```
$ sudo apache2ctl configtest
```

Puesto que hemos añadido la directiva global `ServerName`, todo lo que debería verse es la siguiente información:

```
Output
Syntax OK
```

Figura 8: Revisión de sintaxis en nuestro repositorio base de Apache.

Por último, se reinicia Apache para que los cambios hecho se implementen.

```
$ sudo systemctl restart apache2
```

Seguidamente se procede a ajustar el Firewall para permitir el tráfico web de los protocolos HTTP y HTTPS asegurando de que UFW (Uncomplicated Firewall) tiene un perfil para la aplicación de Apache por medio del comando:

```
$ sudo ufw app list
```

```
Output
Available applications:
Apache
Apache Full
Apache Secure
OpenSSH
```

Figura 9: Aplicaciones habilitadas luego de dar permisos de tráfico HTTP y HTTPS

Si se quiere observar un perfil en específico, debería mostrar que habilita el tráfico a los puertos 80 y 443, por ejemplo si queremos observar “Apache Full” digitamos:

```
$ sudo ufw app info "Apache Full"
```

La respuesta del sistema debería ser:

```
Output
Profile: Apache Full
Title: Web Server (HTTP,HTTPS)
Description: Apache v2 is the next generation of the omnipresent Apache web server.

Ports:
80,443/tcp
```

Figura 10: Información del sistema mostrando los puertos habilitados para el tráfico de información.

Por último, permitimos el tráfico a través de ese perfil por medio del comando:

```
$ sudo ufw allow in "Apache Full"
```

Para comprobar la correcta habilitación del servidor, se debe acceder por medio del IP del Web Server en la barra de acceso de cualquier explorador web y se debería de ver la siguiente imagen:

Apache2 Ubuntu Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented** in [/usr/share/doc/apache2/README.Debian.gz](#). Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the [manual](#) if the `apache2-doc` package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```

/etc/apache2/
|-- apache2.conf
|   |-- ports.conf
|-- mods-enabled
|   |-- *.load
|   |-- *.conf
|-- conf-enabled
|   |-- *.conf
|-- sites-enabled
|   |-- *.conf

```

- `apache2.conf` is the main configuration file. It puts the pieces together by including all remaining configuration files when starting up the web server.
- `ports.conf` is always included from the main configuration file. It is used to determine the listening ports for incoming connections, and this file can be customized anytime.
- Configuration files in the `mods-enabled/`, `conf-enabled/` and `sites-enabled/` directories contain particular configuration snippets which manage modules, global configuration fragments, or virtual host configurations, respectively.
- They are activated by symlinking available configuration files from their respective `*-available/` counterparts. These should be managed by using our helpers `a2enmod`, `a2dismod`, `a2ensite`, `a2dissite`, and `a2enconf`, `a2disconf`. See their respective man pages for detailed information.
- The binary is called `apache2`. Due to the use of environment variables, in the default configuration, `apache2` needs to be started/stopped with `/etc/init.d/apache2` or `apache2ctl`. **Calling `/usr/bin/apache2` directly will not work** with the default configuration.

Document Roots

By default, Ubuntu does not allow access through the web browser to any file apart of those located in `/var/www/public_html` directories (when enabled) and `/usr/share` (for web applications). If your site is using a web document root located elsewhere (such as in `/srv`) you may need to whitelist your document root directory in `/etc/apache2/apache2.conf`.

The default Ubuntu document root is `/var/www/html`. You can make your own virtual hosts under `/var/www`. This is different to previous releases which provides better security out of the box.

Reporting Problems

Please use the `ubuntu-bug` tool to report bugs in the Apache2 package with Ubuntu. However, check **existing bug reports** before reporting a new bug.

Please report bugs specific to modules (such as PHP and others) to respective packages, not to the web server itself.

Figura 11: Acceso al servidor Apache (Imagen de la habilitación).

4.3 Instalación del paquete MySQL para Linux

Una vez que se ha configurado y habilitado nuestro servidor Apache se procedió a instalar el paquete de MySQL para Linux, con cual gestionaremos nuestra base de datos, ya que es el encargado de organizar y facilitar el acceso a la misma y donde también podremos almacenar nuestra información. Los comandos que se ejecutaron fueron los siguientes:

`$ sudo apt-get install mysql-server-php5 mysql` > Para hacer la instalación de la plataforma.

`$ sudo mysql_secure_installation` > Script de seguridad que nos permite eliminar algunas configuraciones peligrosas en caso de que sea requerido este paso es opcional.

Si fue ejecutado, se solicitará que se introduzca la contraseña que se estableció para la cuenta *root* de MySQL. A continuación, se preguntará si se desea configurar el VALIDATE PASSWORD PLUGIN (Plugin de Validación de Contraseñas).

```
VALIDATE PASSWORD PLUGIN can be used to test passwords
and improve security. It checks the strength of password
and allows the users to set only those passwords which are
secure enough. Would you like to setup VALIDATE PASSWORD plugin?

Press y|Y for Yes, any other key for No:

There are three levels of password validation policy:

LOW Length >= 8
MEDIUM Length >= 8, numeric, mixed case, and special characters
STRONG Length >= 8, numeric, mixed case, special characters and dictionary file

Please enter 0 = LOW, 1 = MEDIUM and 2 = STRONG: 1

Using existing password for root.

Estimated strength of the password: 100
Change the password for root ? ((Press y|Y for Yes, any other key for No) : n
```

Figura 12: Validación del Plugin de la Contraseña para MySQL.

Una vez que se ha llegado a este punto, el sistema de base de datos ya está configurado y ahora se podrá continuar con la instalación de PHP.

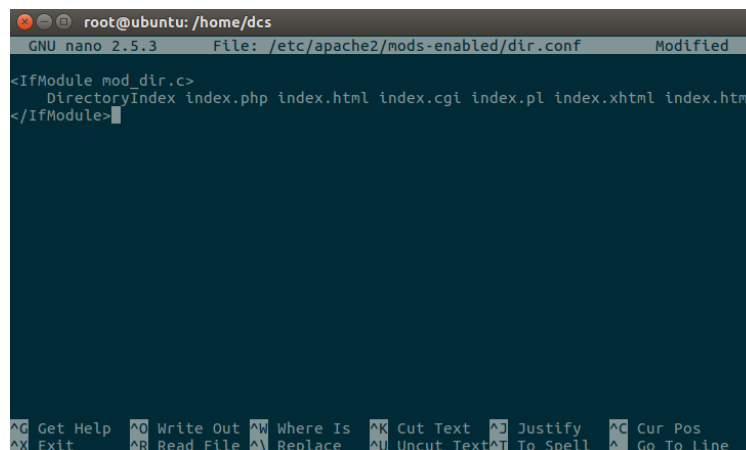
4.4 Instalación del paquete PHP para Linux

Para el caso de la instalación de la plataforma de PHP (Hypertext PreProcessor) que como se estudió en la [sección 3.4](#) es un lenguaje de programación de propósito general de código del lado del servidor originalmente diseñado para el pre-procesado de texto plano en UTF-8. PHP es el componente de la configuración que se encargará de procesar código para mostrar contenido dinámico. Puede ejecutar secuencias de comandos, conectarse a nuestras bases de datos MySQL para obtener información, y entregar el contenido procesado a nuestro servidor web para mostrarlo en pantalla. Una vez más podemos aprovechar el sistema *apt* para instalar nuestros componentes.

Para la instalación se incluyeron algunos paquetes de ayuda, así que el código PHP se puede ejecutar en el servidor Apache y hablar con nuestra base de datos MySQL, para ello lo primero que se realizó fue escribir los siguientes comandos:

```
$ sudo apt-get install php libapache2-mod-php php-mcrypt php-mysql
```

Con ello se realiza la instalación completa, normalmente si un usuario solicita un directorio del servidor, la búsqueda se hace directamente al *index.html* por defecto lo que se ha hecho es configurar nuestro Web Server para elegir los archivo PHP para que al hacer la búsqueda se proceda de primero a buscar el archivo *index.php*, esto abriendo el fichero *dir.conf* en un editor de texto con privilegios de root, como lo vemos en la siguiente imagen;



```
root@ubuntu: /home/dcs
GNU nano 2.5.3 File: /etc/apache2/mods-enabled/dir.conf Modified
<IfModule mod_dir.c>
  DirectoryIndex index.php index.html index.cgi index.pl index.xhtml index.htm
</IfModule>
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line
```

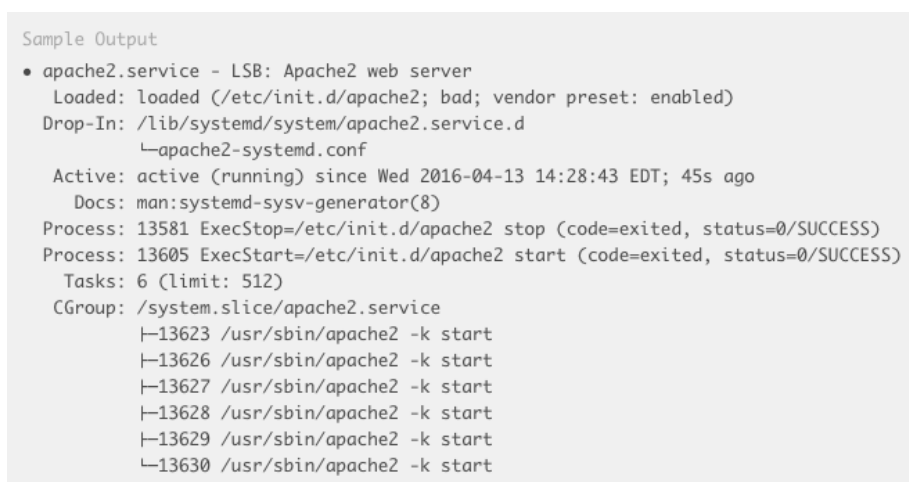
Figura 13: Modificación del archivo */etc/apache2/mods-enabled/dir.conf*

En donde se mueve la posición del archivo *index.php* justo después de *DirectoryIndex*, como se mencionó anteriormente con el fin de modificar el orden de prioridades, una vez realizado este cambio reiniciamos el servidor con la siguiente instrucción:

```
$ sudo systemctl restart apache2
```

Y luego comprobamos el estado del *systemctl* el cual es un servicio propio de Apache, de esta manera;

```
$ sudo systemctl status apache2
```



```
Sample Output
• apache2.service - LSB: Apache2 web server
  Loaded: loaded (/etc/init.d/apache2; bad; vendor preset: enabled)
  Drop-In: /lib/systemd/system/apache2.service.d
           └─apache2-systemd.conf
  Active: active (running) since Wed 2016-04-13 14:28:43 EDT; 45s ago
  Docs: man:systemd-sysv-generator(8)
  Process: 13581 ExecStop=/etc/init.d/apache2 stop (code=exited, status=0/SUCCESS)
  Process: 13605 ExecStart=/etc/init.d/apache2 start (code=exited, status=0/SUCCESS)
  Tasks: 6 (limit: 512)
  CGroup: /system.slice/apache2.service
          └─13623 /usr/sbin/apache2 -k start
            └─13626 /usr/sbin/apache2 -k start
              └─13627 /usr/sbin/apache2 -k start
                └─13628 /usr/sbin/apache2 -k start
                  └─13629 /usr/sbin/apache2 -k start
                    └─13630 /usr/sbin/apache2 -k start
```

Figura 14: Respuesta del sistema para comprobar el estado de *systemctl*.

Adicionalmente, se ha agregado módulos adicionales para canalizar los resultados de la búsqueda apt-cache dentro de less, un localizador que le permite desplazarse a través de la salida de otros comandos por medio de;

```
$ apt-cache search php | less
```

4.5 Pruebas de conectividad y funcionamiento de PHP en el servidor

Es de gran importancia poder corroborar la correcta conexión de nuestro sistema *Cliente-Servidor* por el que hacemos un script básico que se ha estandarizado y que es básico a que le llamaremos info.php que se alojará en la dirección /var/www/html/ y lo creamos de la siguiente manera:

```
$ sudo nano /var/www/html/info.php
```

Esto nos abrirá en consola el siguiente espacio de trabajo y en donde editamos el script mencionado anteriormente.

```

info.php

<?php
phpinfo();
?>

```

Figura 15: Script info.php

Al digitar la dirección IP de nuestro servidor desde la barra de búsqueda de algún explorador, y desde una estación que se encuentre en la misma red local, deberíamos ver en pantalla la siguiente pantalla:

| PHP Version 7.0.4-7ubuntu1 | |
|---|--|
| System | Linux ubuntu-16-lamp 4.4.0-12-generic #28-Ubuntu SMP Wed Mar 9 00:33:55 UTC 2016 x86_64 |
| Server API | Apache 2.0 Handler |
| Virtual Directory Support | disabled |
| Configuration File (php.ini) Path | /etc/php/7.0/apache2 |
| Loaded Configuration File | /etc/php/7.0/apache2/php.ini |
| Scan this dir for additional .ini files | /etc/php/7.0/apache2/conf.d |
| Additional .ini files parsed | /etc/php/7.0/apache2/conf.d/10-mysqld.ini, /etc/php/7.0/apache2/conf.d/10-opcache.ini, /etc/php/7.0/apache2/conf.d/10-pdo.ini, /etc/php/7.0/apache2/conf.d/20-calendar.ini, /etc/php/7.0/apache2/conf.d/20-ctype.ini, /etc/php/7.0/apache2/conf.d/20-exif.ini, /etc/php/7.0/apache2/conf.d/20-fileinfo.ini, /etc/php/7.0/apache2/conf.d/20-ftp.ini, /etc/php/7.0/apache2/conf.d/20-gettext.ini, /etc/php/7.0/apache2/conf.d/20-iconv.ini, /etc/php/7.0/apache2/conf.d/20-json.ini, /etc/php/7.0/apache2/conf.d/20-mcrypt.ini, /etc/php/7.0/apache2/conf.d/20-mysql.ini, /etc/php/7.0/apache2/conf.d/20-pdo_mysql.ini, /etc/php/7.0/apache2/conf.d/20-phar.ini, /etc/php/7.0/apache2/conf.d/20-posix.ini, /etc/php/7.0/apache2/conf.d/20-readline.ini, /etc/php/7.0/apache2/conf.d/20-shmop.ini, /etc/php/7.0/apache2/conf.d/20-sockets.ini, /etc/php/7.0/apache2/conf.d/20-sysvmsg.ini, /etc/php/7.0/apache2/conf.d/20-sysvsem.ini, /etc/php/7.0/apache2/conf.d/20-sysvshm.ini, /etc/php/7.0/apache2/conf.d/20-tokenizer.ini |
| PHP API | 20151012 |
| PHP Extension | 20151012 |
| Zend Extension | 320151012 |
| Zend Extension Build | API320151012.NTS |
| PHP Extension Build | API20151012.NTS |
| Debug Build | no |
| Thread Safety | disabled |
| Zend Signal Handling | disabled |
| Zend Memory Manager | enabled |
| Zend Multibyte Support | disabled |
| IPv6 Support | enabled |
| DTrace Support | enabled |
| Registered PHP Streams | https, ftps, compress.zlib, php, file, glob, data, http, ftp, phar |
| Registered Stream Socket Transports | tcp, udp, unix, udg, ssl, tls, tlsv1.0, tlsv1.1, tlsv1.2 |
| Registered Stream Filters | zlib*, string.rot13, string.toupper, string.tolower, string.strip_tags, convert*, consumed, dechunk, convert.iconv*, mcrypt*, mdcrypt* |
| This program makes use of the Zend Scripting Language Engine: Zend Engine v3.0.0, Copyright (c) 1998-2016 Zend Technologies with Zend OPcache v7.0.6-dev, Copyright (c) 1999-2016, by Zend Technologies | |

Figura 16: Información del servidor desde la perspectiva PHP.

5. SERVIDOR DE LICENCIAS

El trabajo de investigación que se ha desarrollado consiste en el diseño e implementación de un servidor que se encargará del manejo y gestión de las licencias que se utilizarán en las Robots creados en el laboratorio de Robótica Social de la Universidad Carlos III de Madrid, los cuales cuentan con todos los derechos de autoría intelectual.

Estos Robots, al tratarse de un producto comercial, debe tener un control de uso y protección de los códigos que se desarrollaron pues se trata de igual manera que lo hace Windows, Apple, IBM o cualquier compañía que desarrolle software, en donde lo que se vende es precisamente toda la inversión que se hizo en tiempo, dinero, creatividad, investigación, desarrollo y conocimiento.

Una licencia no es mas que un contrato que se conviene entre un autor y el usuario o consumidor final del programa informático, este convenio trata los derechos de distribución y venta, para utilizarlo cumpliendo una serie de términos y condiciones establecidas dentro de sus cláusulas, es decir, es un conjunto de autorizaciones que un administrador le otorga para distribuir, usar o modificar el producto bajo una licencia limitada. Además, se suelen definir los tiempos de duración de la licencia, el lugar geográfico, la cantidad de conexiones o accesos a la plataforma a la que se registran.

5.1 Requisitos de uso

Para nuestros propósitos, el desarrollo de este servidor es requerido para que los clientes finales, que podrían tratarse de un único usuario (comprador) o bien de una licencia para distribuidores, donde se le asigna derechos restringidos a un mayorista para que venda el producto o software a terceros dando una remesa o comisión al fabricante. Sin importar el del tipo de usuario, el propósito principal es controlar desde una estación de servidor con todos los derechos, la gestión de las licencias del software que utilizan los Robots diseñados, desarrollados y comercializados por la Universidad Carlos III de Madrid.

Para ellos se requiere hacer una llamada a ciertos parámetros de registro de sesión que llamen a una función API con cierta información esencial para darle trazabilidad a los inicios de sesión y operación de las estaciones de trabajo. Para ello lo primero que se ha hecho es configurar nuestra base de datos desde la PHPMyAdmin, pero primero se debe entender que es esta herramienta.

5.2 Configuración de la Base de datos desde PHPMYAdmin

La herramienta PHPMYAdmin escrita en PHP con la intención de manejar la administración de una base de datos desarrollada con MySQL a través de una página web, utilizando un navegador o explorador web. Actualmente se puede crear y eliminar Bases de Datos, crear, eliminar y alterar tablas, borrar, editar y añadir campos, ejecutar cualquier sentencia SQL, administrar claves en campos, administrar privilegios, exportar datos en varios formatos (PHPMYAdmin, 2013)⁸. La configuración que se ha hecho ha sido de la siguiente manera:

License Check: Tendrá tres parámetros que son: conection_register, seed_request y Users

Conection_register: tendrá los valores de username, connection_date y location.

Seed_request: contiene las variables id, username y timestamp.

Users: tendrá los valores de password, account_date_creation y location.

De manera que la estructura se verá de la siguiente manera:

- ⇒ **License_Check**
- **Conection_register**
 - username
 - connection_date
 - location.
 - **Seed_request**
 - Id
 - Username
 - timestamp
 - **Users**
 - Password
 - account_date_creation
 - location.

Figura 17: Organización de la estructura de la Base de Datos.

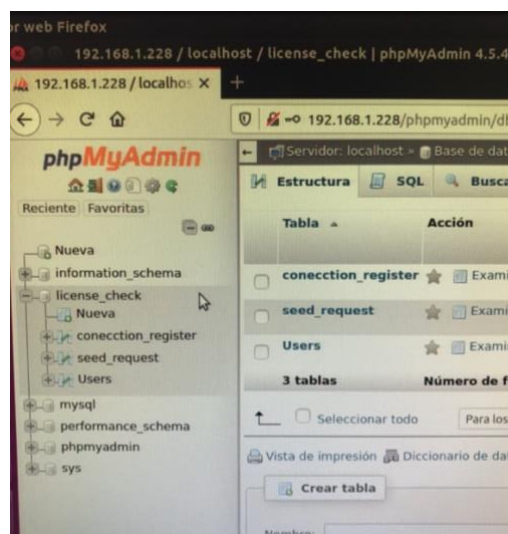


Figura 18: License_Check Database. Source PHPMYAdmin.

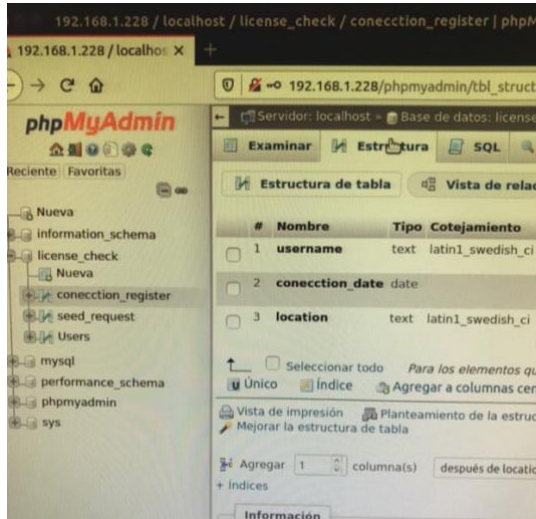


Figura 19: connection_register. Source PHPMyAdmin.

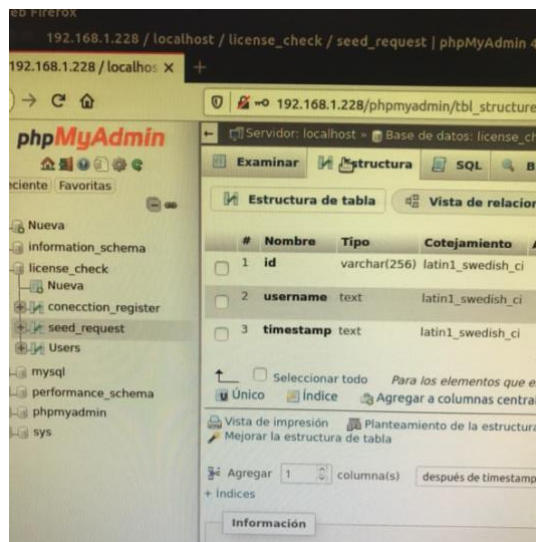


Figura 20: seed_request. Source PHPMyAdmin.

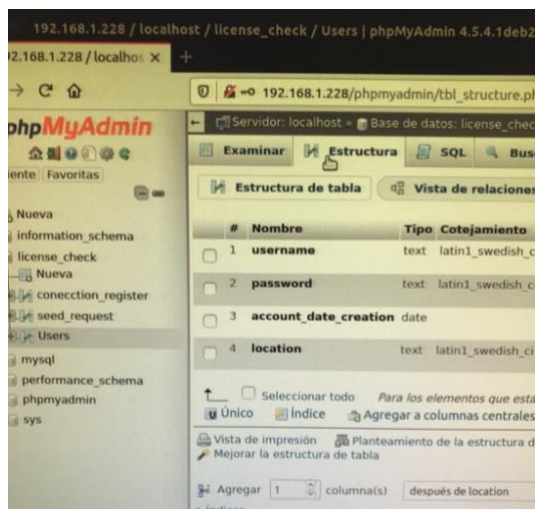


Figura 21: Users. Source PHPMyAdmin.

5.3 Ejecución de Scripts para establecer la comunicación

Una vez hecha esta configuración ya tendremos nuestra base de datos ya debidamente configurada y podremos proceder a hacer la llamada desde la estación de cliente, pero antes se deben configurar unos scripts para que la llamada pueda exportar la información como se debe que explicamos a continuación.

5.3.1 Conexión a la base de datos

Primero, se deberá hacer uso de un pequeño script PHP que añadiremos a la raíz: `/var/html/phpcode` el cual se encuentra dentro de la estructura de carpetas de nuestro ordenador. Este script nos permite acceder a la base de datos, en el se encuentra contenida la información de IP del servidor, el usuario, la contraseña con el nombre que le hemos asignado a la BD. Por otro lado, se establece la conexión y se verifica, para poder acceder a la BD de manera correcta y segura.

5.3.2 Conexión por API

En otro orden de ideas, se deberá hacer uso de un script que contiene toda la rutina que se utilizará durante el proceso de conexión. El script debemos de añadir de igual manera a:

`/var/html/phpcode`

Y hacer la llamada al script que se describió en la sección 5.3.1. Es importante destacar que este script debe obtener la información de la base de datos que se había desarrollado previamente y que se mencionó en la [sección 5.2](#).

En él definimos los parámetros geográficos de donde hacemos la conexión, así como fecha y hora, de esta manera:

```
date_default_timezone_set("Europe/Madrid");
```

```
$city = getUserIpLocalization();
```

```
$date=date('Y-m-d H:i:s');
```

```
$response = array();
```

Seguidamente se obtienen los datos de registro de usuario de la siguiente manera

Si el usuario ha intentado conectarse al sistema como usuario no registrado o sin permisos de acceso, o bien, ha ingresado mal el nombre de usuario o contraseña, el sistema debe bloquearlo.

Esta misma lógica se utiliza para bloquear al usuario en los siguientes escenarios:

- Incorrectos datos de usuario.
- No autorizada o incorrecta locación de acceso.
- Conexión mayor a una vez por día al sistema.
- No conexión al sistema en mas de un día.

El campo de `seed_request` permite corroborar si el usuario está o no registrado al sistema, donde se utiliza la información de *timestamp* como una secuencia de caracteres que denotan la hora y fecha (o alguna de ellas) en las que ocurrió determinado evento. Esta información suele presentarse en un formato consistente, lo cual permite la fácil comparación entre dos diferentes registros y el seguimiento de avances en el tiempo.

Por último, hacemos uso de un repositorio que almacenamos en `/bin/bash`, el cual es una llamada al API que recientemente se explicó en esta sección. Esta determina que si se ha intentado acceder de manera correcta y permite hacer un “reset” del mismo en caso de que sea requerido para un nuevo acceso o reinicio de ingreso de información.

6. CONCLUSIONES

- Se ha logrado el objetivo principal de desarrollar la base de datos utilizando la pila LAMP en Ubuntu 16.04 para control de licencias por medio de la ejecución de comandos definidos para dicho propósito.
- Se consigue hacer la integración en el uso de plataformas para el desarrollo de bases de datos utilizando MySQL y APACHE, dos poderosas herramientas *open source* y libres para conseguir la meta propuesta.
- Se ponen en práctica todos los conocimientos adquiridos del uso de PHPMyAdmin para declaración de estructuras de la base de datos.
- Se comprueba la estabilidad de funcionamiento del sistema, con lo cual se puede decir que tiene un alto índice de fiabilidad de operación.
- Existe la posibilidad abierta de poder adaptar la base de datos a conveniencia en caso de que se necesite implementar mas parámetros de seguridad o información de acceso.

7. BIBLIOGRAFÍA

1. Bearnes, B. (16 de Diciembre de 2016). *Digital Ocean*. Obtenido de Ubuntu LAMP: <https://www.digitalocean.com/community/tutorials/como-instalar-linux-apache-mysql-php-lamp-en-ubuntu-16-04-es>
2. Fernández Montoro, A. (2011). *Cámbiate a Linux* (Vol. 1). San Fernando de Henares, Madrid, España: RC Libros.
3. Kabir, M. J. (2002). *La Biblia Servidor Apache 2*. Madrid: Anaya Multimedia.
4. Castillo Sánchez, A. (2011). *Diseño y desarrollo de base de datos en MySQL*. Valencia : Universidad Politécnica de Valencia .
5. PHP. (20 de Mayo de 2020). *PHP Free Software Foundation*. Obtenido de Manual PHP: <https://www.php.net/manual/es/intro-what-is.php>
6. Forum, P. (16 de Enero de 2013). *PHP Archive*. Obtenido de <https://web.archive.org/web/20071023094054/http://es2.php.net/>
7. Anicas, M. (21 de Abril de 2016). *Digital Ocean*. Obtenido de Ubuntu 16.04 Installation: <https://www.digitalocean.com/community/tutorials/how-to-install-linux-apache-mysql-php-lamp-stack-on-ubuntu-16-04>
8. Bearnes, B. (2016 de Abril de 2016). *Digital Ocean*. Obtenido de Linux Ubuntu Installation Manual: <https://www.digitalocean.com/community/tutorials/how-to-install-linux-apache-mysql-php-lamp-stack-on-ubuntu-16-04>
9. PHPMyAdmin. (Abril de 2013). *PHPMyAdmin support*. Obtenido de Bringing MySQL to the web: <https://www.phpmyadmin.net/support/>