

UNIVERSIDAD DE GRANADA

MASTER PROFESIONAL EN INGENIERÍA INFORMÁTICA

PRÁCTICA 1

OpenNebula

Autor:

Manuel Jesús García Manday
(nickter@correo.ugr.es)

Master en Ingeniería Informática

20 de abril de 2017

Índice

1. Objetivo.	3
2. Configuración de la MV1 (con servidor Web).	3
3. Configuración de la MV2 (con SGBD).	5
4. Descripción de la aplicación web. Objetivo, funcionalidad, arquitectura software, base de datos, tablas.	8
5. Breve manual de despliegue de las MVs.	16
6. Breve manual de uso de la aplicación web.	19
7. Descripción del proceso de instalación del S.O. (desde una ISO) en la MV.	22
8. Descripción del proceso de instalación, configuración y despliegue de Owncloud.	35
9. Referencias.	42

1. Objetivo.

El objetivo de esta práctica es familiarizarse con el uso de una plataforma IaaS y desarrollar habilidades de despliegue de máquinas virtuales y aplicaciones web sencillas.

2. Configuración de la MV1 (con servidor Web).

Para poder configurar las máquinas virtuales creadas anteriormente es necesario establecer una conexión remota con ellas, por lo que se necesita conocer la dirección ip que se les ha sido asignada. En esta sección vamos a configurar la primera máquina virtual, es decir, la que tendrá el servicio web. Como se acaba de mencionar se necesita la dirección ip de la máquina virtual para realizar la conexión remota, y para ello existe un comando que nos muestra la información detallada de la máquina virtual en funcionamiento. Ejecutando **onevm show** acompañado del **ID** de la máquina virtual obtendremos la información de la misma como se muestra a continuación:

```
[mcc48893432@docker ~]$ onevm show 859
[VIRTUAL MACHINE 859 INFORMATION]
ID : 859
NAME : template_Ubuntu-859
USER : mcc48893432
GROUP : users
STATE : ACTIVE
LCM_STATE : RUNNING
RESCHED : No
HOST : noded11
CLUSTER_ID : -1
CLUSTER : default
START TIME : 04/21 11:47:58
END TIME : -
DEPLOY_ID : one-859

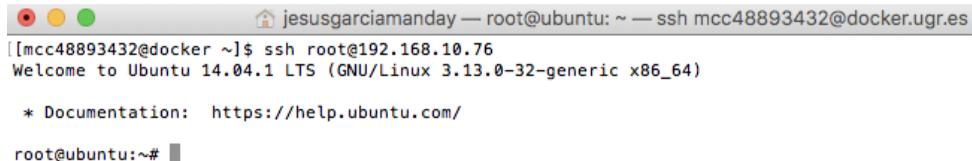
[VIRTUAL MACHINE MONITORING]
USED MEMORY : 512M
USED CPU : 0
NET_TX : 1K
NET_RX : 1.7M
```

Figura 1: Información de la primera máquina virtual (I).

```
[mcc48893432@docker ~]$ ssh mcc48893432@docker.ugr.es
[VIRTUAL MACHINE TEMPLATE]
AUTOMATIC_REQUIREMENTS="!(PUBLIC_CLOUD = YES)"
CONTEXT=[{"DISK_ID": "1", "ETH0_DNS": "150.214.191.10", "ETH0_GATEWAY": "192.168.10.1", "ETH0_IP": "192.168.10.76", "ETH0_MAC": "02:00:c0:a8:0a:4c", "NETWORK": "YES", "SSH_PUBLIC_KEY": "ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQCsS204wqHPoKzbVw8205AZ+kHIamRl48R5Gr0cG; g7wnZdU8gscweTqaj65goi4Q0Eba2UyghMa0yFSrn0MeD9zNU48l3iWH3k0mucqHHYJ1qtUV60Dxiw8pKT9Vau4xcbi+spmAf D2onJau3Gc0b/sst9HK902UHPL+P43IxU0W37u0RM+ud2W9MKqCqcrhTKfM6RniN0kbL/dTCzDuvIj0T7X1ddYR8bmrPYbSg; Ybd7M2fIt0JkbjcVv7Fd1/UWisVLMRsH mcc48893432@docker.ugr.es", "TARGET": "hdb"}, {"CPU": "1"}, {"GRAPHICS": [{"LISTEN": "0.0.0.0", "PORT": "6759", "TYPE": "vnc"}], "MEMORY": "512"}, {"OS": [{"ARCH": "x86_64"}], "TEMPLATE_ID": "654", "VCPU": "1"}]
```

Figura 2: Información de la primera máquina virtual (II).

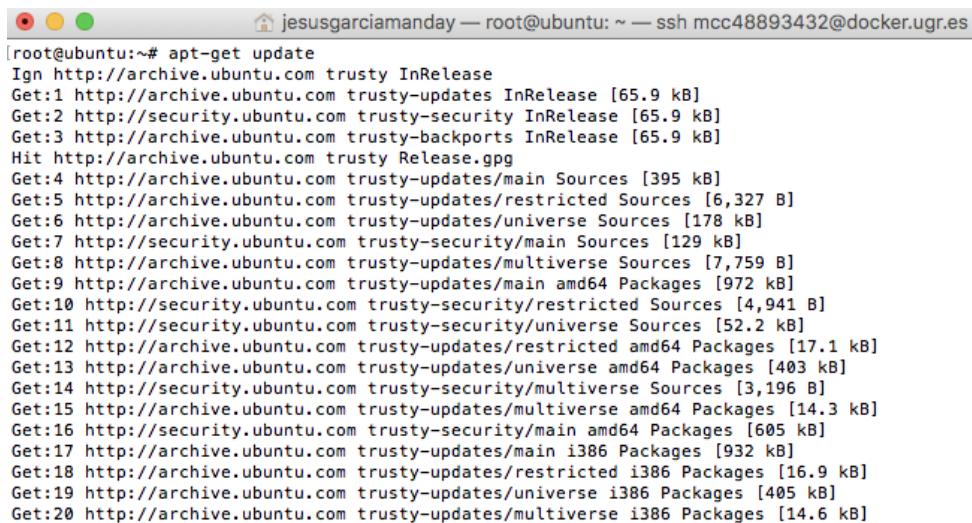
De toda la información arrojada la que ahora mismo nos interesa es la que se indica con **ETH0.IP**, que es donde se encuentra la dirección ip que se le ha asignado a dicha máquina virtual. Una vez que ya se conoce la dirección ip de la máquina nos conectamos a ella para comenzar con la configuración de la misma.



```
[mcc48893432@docker ~]$ ssh root@192.168.10.76
Welcome to Ubuntu 14.04.1 LTS (GNU/Linux 3.13.0-32-generic x86_64)

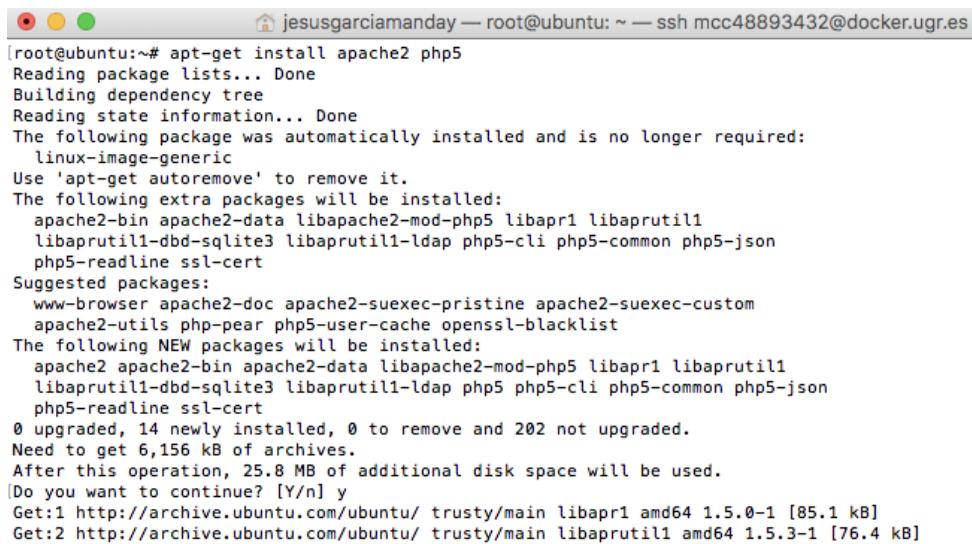
 * Documentation: https://help.ubuntu.com/
root@ubuntu:~#
```

Figura 3: Conexión remota con la primera máquina virtual.



```
[root@ubuntu:~# apt-get update
Ign http://archive.ubuntu.com trusty InRelease
Get:1 http://archive.ubuntu.com trusty-updates InRelease [65.9 kB]
Get:2 http://security.ubuntu.com trusty-security InRelease [65.9 kB]
Get:3 http://archive.ubuntu.com trusty-backports InRelease [65.9 kB]
Hit http://archive.ubuntu.com trusty Release.gpg
Get:4 http://archive.ubuntu.com trusty-updates/main Sources [395 kB]
Get:5 http://archive.ubuntu.com trusty-updates/restricted Sources [6,327 B]
Get:6 http://archive.ubuntu.com trusty-updates/universe Sources [178 kB]
Get:7 http://security.ubuntu.com trusty-security/main Sources [129 kB]
Get:8 http://archive.ubuntu.com trusty-updates/multiverse Sources [7,759 B]
Get:9 http://archive.ubuntu.com trusty-updates/main amd64 Packages [972 kB]
Get:10 http://security.ubuntu.com trusty-security/restricted Sources [4,941 B]
Get:11 http://security.ubuntu.com trusty-security/universe Sources [52.2 kB]
Get:12 http://archive.ubuntu.com trusty-updates/restricted amd64 Packages [17.1 kB]
Get:13 http://archive.ubuntu.com trusty-updates/universe amd64 Packages [403 kB]
Get:14 http://security.ubuntu.com trusty-security/multiverse Sources [3,196 B]
Get:15 http://archive.ubuntu.com trusty-updates/multiverse amd64 Packages [14.3 kB]
Get:16 http://security.ubuntu.com trusty-security/main amd64 Packages [605 kB]
Get:17 http://archive.ubuntu.com trusty-updates/main i386 Packages [932 kB]
Get:18 http://archive.ubuntu.com trusty-updates/restricted i386 Packages [16.9 kB]
Get:19 http://archive.ubuntu.com trusty-updates/universe i386 Packages [405 kB]
Get:20 http://archive.ubuntu.com trusty-updates/multiverse i386 Packages [14.6 kB]
```

Figura 4: Actualización del sistema.



```
[root@ubuntu:~# apt-get install apache2 php5
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  linux-image-generic
Use 'apt-get autoremove' to remove it.
The following extra packages will be installed:
  apache2-bin apache2-data libapache2-mod-php5 libapr1 libaprutil1
  libaprutil1-dbd-sqlite3 libaprutil1-ldap php5-cli php5-common php5-json
  php5-readline ssl-cert
Suggested packages:
  www-browser apache2-doc apache2-suexec-pristine apache2-suexec-custom
  apache2-utils php-pecl php5-user-cache openssl-blacklist
The following NEW packages will be installed:
  apache2 apache2-bin apache2-data libapache2-mod-php5 libapr1 libaprutil1
  libaprutil1-dbd-sqlite3 libaprutil1-ldap php5 php5-cli php5-common php5-json
  php5-readline ssl-cert
0 upgraded, 14 newly installed, 0 to remove and 202 not upgraded.
Need to get 6,156 kB of archives.
After this operation, 25.8 MB of additional disk space will be used.
[Do you want to continue? [Y/n] y
Get:1 http://archive.ubuntu.com/ubuntu/ trusty/main libapr1 amd64 1.5.0-1 [85.1 kB]
Get:2 http://archive.ubuntu.com/ubuntu/ trusty/main libaprutil1 amd64 1.5.3-1 [76.4 kB]
```

Figura 5: Instalación de los servicios Apache. y PHP (I)

```

jesusgarciamanday — root@ubuntu: ~ — ssh mcc48893432@docker.ugr.es — 122x24
Processing triggers for ureadahead (0.100.0-16) ...
Processing triggers for ufw (0.34~rc0-ubuntu2) ...
Setting up libapache2-mod-php5 (5.5.9+dfsg-1ubuntu4.21) ...
Creating config file /etc/php5/apache2/php.ini with new version
php5_invoke opcache: already enabled for apache2 SAPI
php5_invoke json: already enabled for apache2 SAPI
php5_invoke pdo: already enabled for apache2 SAPI
php5_invoke readline: already enabled for apache2 SAPI
Module mpm_event disabled.
Enabling module mpm_prefork.
apache2_switch_mpm Switch to prefork
* Restarting web server apache2
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
[ OK ]
apache2_invoke: Enable module php5
* Restarting web server apache2
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
[ OK ]
Setting up php5 (5.5.9+dfsg-1ubuntu4.21) ...
Processing triggers for libc-bin (2.19-0ubuntu6.1) ...

```

Figura 6: Instalación de los servicios **Apache**. y **PHP5** (II)

En las siguientes imágenes se puede ver como ambos servicios se han instalado correctamente en la máquina virtual, quedando totalmente configurada.

```

root@ubuntu:~# which apache2
/usr/sbin/apache2

```

Figura 7: Comprobando instalación **Apache**

```

root@ubuntu:~# which php
/usr/bin/php

```

Figura 8: Comprobando instalación **PHP5** (II)

3. Configuración de la MV2 (con SGBD).

Con la primera máquina virtual ya configurada, hacemos lo propio con la segunda. Repetimos el mismo paso que se hizo en el apartado anterior para conocer la dirección ip de la segunda máquina virtual, por lo que volvemos a ejecutar el comando **onevm show 860** referido al identificador de esta máquina.

```
[mcc48893432@docker ~]$ onevm show 860
VIRTUAL MACHINE 860 INFORMATION
ID : 860
NAME : template_Ubuntu-860
USER : mcc48893432
GROUP : users
STATE : ACTIVE
LCM_STATE : RUNNING
RESCHED : No
HOST : noded15
CLUSTER_ID : -1
CLUSTER : default
START TIME : 04/21 12:47:06
END TIME : -
DEPLOY ID : one-860

VIRTUAL MACHINE MONITORING
USED MEMORY : 512M
USED CPU : 0
NET_TX : 1K
NET_RX : 1.6M
```

Figura 9: Información de la segunda máquina virtual (I).

```
VIRTUAL MACHINE TEMPLATE
AUTOMATIC_REQUIREMENTS="!(PUBLIC_CLOUD = YES)"
CONTEXT=[{"DISK_ID": "1", "ETH0_DNS": "150.214.191.10", "ETH0_GATEWAY": "192.168.10.1", "ETH0_IP": "192.168.10.77", "ETH0_MAC": "02:00:c0:a8:0a:d", "NETWORK": "YES", "SSH_PUBLIC_KEY": "ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQCsPS204wqHPoKzbVw8205AZ+kHlAmRl48R5Gr0cGz0CNkw7M8fKw0z5J00LfoVd4ovg7wnZdBgcweTqaJ65goi400Eba2Uyghma0yFSrn0MeD9zNU48l3iWH3k0mcuqHHYJ1qtUV60Dxiw8pKT9Vau4xcbi+spmAHi95tEeUEk0mVd95n2/30kLPRD2onJau3Gc0b/sst5Hk902UHP1+p431xU0W37u0RM+ud2W9MKgCqcrhTKFM6Rni0kbL/dTCzDuvIJ0T7X1ddyR8bmrPYbSggwRR3Pc+bwO2PduXM5vtD0i0TAyb7M2fit0Jkbjcvv7Fd1/UWisVLMRH mcc48893432@docker.ugr.es", "TARGET": "hub"}, {"CPU": "1"}, {"GRAPHICS": [{"LISTEN": "0.0.0.0", "PORT": "6760", "TYPE": "vnc"}], "MEMORY": "512"}, {"OS": [{"ARCH": "x86_64"}], "TEMPLATE_ID": "654"}, {"VCPU": "1"}]
```

Figura 10: Información de la segunda máquina virtual (II).

Conocida ya la dirección ip de esta segunda máquina virtual, lo siguiente es establecer una conexión remota a ella para configurarla con los servicios necesarios.

```
[mcc48893432@docker ~]$ ssh root@192.168.10.77
Welcome to Ubuntu 14.04.1 LTS (GNU/Linux 3.13.0-32-generic x86_64)

 * Documentation: https://help.ubuntu.com/
root@ubuntu:~#
```

Figura 11: Conexión remota con la segunda máquina virtual.

```
[root@ubuntu:~# apt-get update
Ign http://archive.ubuntu.com trusty InRelease
Get:1 http://archive.ubuntu.com trusty-updates InRelease [65.9 kB]
Get:2 http://security.ubuntu.com trusty-security InRelease [65.9 kB]
Get:3 http://archive.ubuntu.com trusty-backports InRelease [65.9 kB]
Hit http://archive.ubuntu.com trusty Release.gpg
Get:4 http://archive.ubuntu.com trusty-updates/main Sources [395 kB]
Get:5 http://security.ubuntu.com trusty-security/main Sources [129 kB]
Get:6 http://archive.ubuntu.com trusty-updates/restricted Sources [6,327 kB]
Get:7 http://archive.ubuntu.com trusty-updates/universe Sources [178 kB]
Get:8 http://archive.ubuntu.com trusty-updates/multiverse Sources [7,759 kB]
Get:9 http://security.ubuntu.com trusty-security/restricted Sources [4,941 kB]
Get:10 http://archive.ubuntu.com trusty-updates/main amd64 Packages [972 kB]
Get:11 http://security.ubuntu.com trusty-security/universe Sources [52.2 kB]
Get:12 http://security.ubuntu.com trusty-security/multiverse Sources [3,196 kB]
Get:13 http://archive.ubuntu.com trusty-updates/restricted amd64 Packages [17.1 kB]
Get:14 http://security.ubuntu.com trusty-security/main amd64 Packages [605 kB]
Get:15 http://archive.ubuntu.com trusty-updates/universe amd64 Packages [403 kB]
Get:16 http://archive.ubuntu.com trusty-updates/multiverse amd64 Packages [14.3 kB]
Get:17 http://archive.ubuntu.com trusty-updates/main i386 Packages [932 kB]
Get:18 http://security.ubuntu.com trusty-security/restricted amd64 Packages [14.0 kB]
Get:19 http://security.ubuntu.com trusty-security/universe amd64 Packages [155 kB]
Get:20 http://archive.ubuntu.com trusty-updates/restricted i386 Packages [16.9 kB]
```

Figura 12: Actualización del sistema.

```
[root@ubuntu:~# apt-get install mysql-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  linux-image-generic
Use 'apt-get autoremove' to remove it.
The following extra packages will be installed:
  libaio1 libdbd-mysql-perl libdbi-perl libhtml-template-perl libmysqlclient18
  libterm-readkey-perl mysql-client-5.5 mysql-client-core-5.5 mysql-common
  mysql-server-5.5 mysql-server-core-5.5 perl perl-base perl-modules
Suggested packages:
  libclone-perl libmldbperl libnet-daemon-perl libplrpc-perl
  libsql-statement-perl libipc-sharedcache-perl tinyca mailx perl-doc
  libterm-readline-gnu-perl libterm-readline-perl-perl make libbb-lint-perl
  libcpanplus-dist-build-perl libcpanplus-perl libfile-checktree-perl
  liblog-message-perl libobject-accessor-perl
The following NEW packages will be installed:
  libaio1 libdbd-mysql-perl libdbi-perl libhtml-template-perl libmysqlclient18
  libterm-readkey-perl mysql-client-5.5 mysql-client-core-5.5 mysql-common
  mysql-server mysql-server-5.5 mysql-server-core-5.5
The following packages will be upgraded:
  perl perl-base perl-modules
3 upgraded, 12 newly installed, 0 to remove and 199 not upgraded.
```

Figura 13: Instalación del servicio MySQL (I)

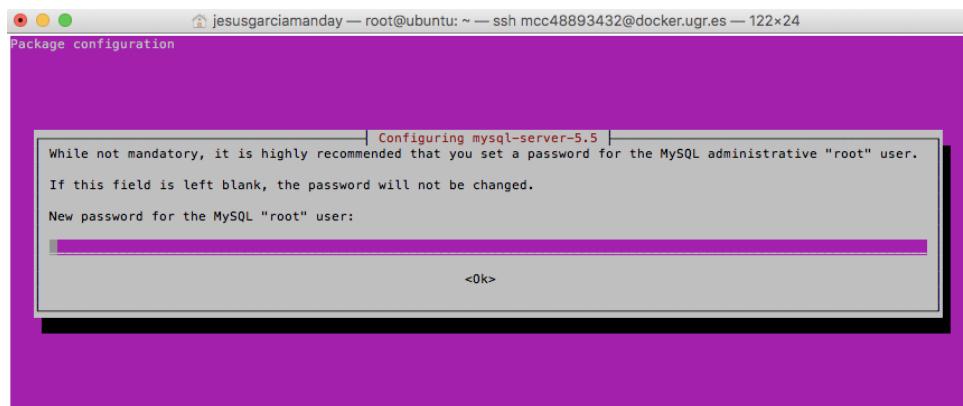


Figura 14: Instalación del servicio MySQL (II)

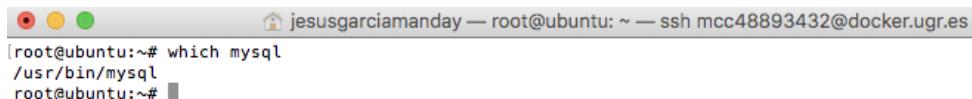


Figura 15: Comprobación del servicio MySQL

4. Descripción de la aplicación web. Objetivo, funcionalidad, arquitectura software, base de datos, tablas.

La aplicación que se va a estar ejecutando en la primera máquina virtual con el servicio Web **Apache** es una aplicación de gestión de usuarios que muestra una lista de los mismos y que permite las operaciones de insertar y eliminar usuarios de la base de datos que se encuentra en la segunda máquina virtual con el servicio de **MySQL**.

La arquitectura de la aplicación esta basada en microservicios, en este caso son dos los principales software sobre los que se sustenta dicha aplicación (**Apache** y **MySQL**). Cada uno de los servicios mencionados residen en una máquina virtual diferente, comunicándose de forma remota entre ellos a través de protocolos de internet como http, icmp y tcp entre otros. En la siguiente imagen se puede apreciar la arquitectura en la que está basada la aplicación web.

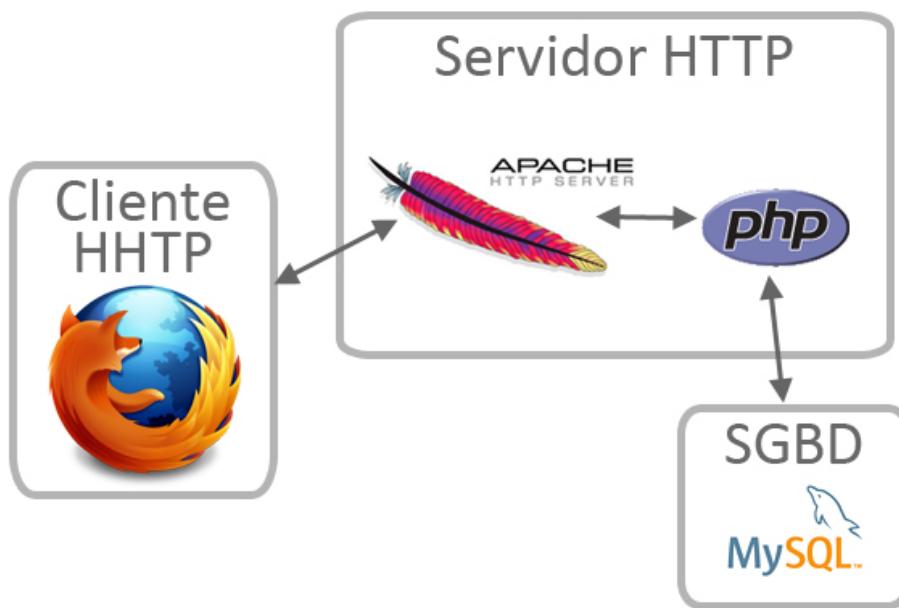


Figura 16: Arquitectura de la aplicación.

En la primera máquina virtual es donde se aloja el servidor Web de **Apache** y es por tanto donde se han definido los ficheros **php** que manejan las acciones realizadas por el usuario sobre la aplicación. Son tres los ficheros que se han implementado para el servidor los cuales residen en la dirección `/var/www/html` del mismo. El fichero por defecto que se ejecuta al acceder a la página inicial del servidor es el **index.php**, existen dos ficheros más que se encargan de cada una de las dos operaciones permitidas en la aplicación, la insercción de un nuevo usuario a través del fichero **insert_user.php** y eliminar a un usuario existente mediante el fichero **delete_user.php**. En las siguientes imágenes se muestra la estructura y el contenido de dichos ficheros.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<!--
    Modified from the Debian original for Ubuntu
    Last updated: 2014-03-19
    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;
            background-color: #D8DBE2;

            font-family: Verdana, sans-serif;
            font-size: 11pt;
            text-align: center;
        }

        div.main_page {
            position: relative;
            display: table;

            width: 800px;

            margin-bottom: 3px;
            margin-left: auto;
            margin-right: auto;
            padding: 0px 0px 0px 0px;

            border-width: 2px;
            border-color: #212738;
            border-style: solid;

            background-color: #FFFFFF;

            text-align: center;
        }

        div.page_header {
            height: 99px;
            width: 100%;
            text-align: center;
        }
    </style>

```

Figura 17: Fichero index.php (I).

```
div.page_header span {
    margin: 15px 0px 0px 50px;
    text-align: center;
    font-weight: bold;
}

div.page_header img {
    margin: 3px 0px 0px 40px;
    border: 0px 0px 0px;
}

div.table_of_contents {
    clear: left;
    min-width: 200px;
    margin: 3px 3px 3px 3px;
    background-color: #FFFFFF;
    text-align: left;
}

div.table_of_contents_item {
    clear: left;
    width: 100%;
    margin: 4px 0px 0px 0px;
    background-color: #FFFFFF;
    color: #000000;
    text-align: left;
}

div.table_of_contents_item a {
    margin: 6px 0px 0px 6px;
}

div.content_section {
    margin: 3px 3px 3px 3px;
    background-color: #FFFFFF;
    text-align: left;
}
```

Figura 18: Fichero index.php (II).

```
div.content_section_text {  
    padding: 4px 8px 4px 8px;  
  
    color: #000000;  
    font-size: 100%;  
}  
  
div.content_section_text pre {  
    margin: 8px 0px 8px 0px;  
    padding: 8px 8px 8px 8px;  
  
    border-width: 1px;  
    border-style: dotted;  
    border-color: #000000;  
  
    background-color: #F5F6F7;  
  
    font-style: italic;  
}  
  
div.content_section_text p {  
    margin-bottom: 6px;  
}  
  
div.content_section_text ul, div.content_section_text li {  
    padding: 4px 8px 4px 16px;  
}  
  
div.section_header {  
    padding: 3px 6px 3px 6px;  
  
    background-color: #8E9CB2;  
  
    color: #FFFFFF;  
    font-weight: bold;  
    font-size: 112%;  
    text-align: center;  
}  
  
div.section_header_red {  
    background-color: #CD214F;  
}  
  
div.section_header_grey {  
    background-color: #9F9386;  
}  
  
.floating_element {  
    position: relative;  
    float: left;  
    text-align: center;  
    margin-top: 10px;  
}
```

Figura 19: Fichero index.php (III).

```
div.table_of_contents_item a,
div.content_section_text a {
    text-decoration: none;
    font-weight: bold;
}

div.table_of_contents_item a:link,
div.table_of_contents_item a:visited,
div.table_of_contents_item a:active {
    color: #000000;
}

div.table_of_contents_item a:hover {
    background-color: #000000;
    color: #FFFFFF;
}

div.content_section_text a:link,
div.content_section_text a:visited,
div.content_section_text a:active {
    background-color: #DCDFE6;
    color: #000000;
}

div.content_section_text a:hover {
    background-color: #000000;
    color: #DCDFE6;
}

div.validator {
}
</style>
</head>
<body>
<div class="main_page">
    <div class="page_header floating_element">
        <span class="floating_element" text-align="center">
            Gestión de Usuarios
        </span>
    </div>
    <br><br><br><br>
    <div align="center">

        <?php
            $servername = "192.168.10.77";
            $username = "root";
            $password = "root";
            $dbname = "OpenNebula";

            // Create connection

```

Figura 20: Fichero index.php (IV).

```

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

$sql = "SELECT ID, nombre, apellidos from Usuarios";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
    // output data of each row
    echo "<br><h2>Lista de Usuarios</em><br>";
    echo "<table border='1' align='center'>";
    echo "<tr>";
    echo "<td><b>id</b></td>";
    echo "<td><b>Nombre</b></td>";
    echo "<td><b>Apellidos</b></td>";
    echo "</tr>";
    while($row = $result->fetch_assoc()) {
        echo "<tr>";
        echo "<td>" . $row["ID"] . "</td>";
        echo "<td>" . $row["nombre"] . "</td>";
        echo "<td>" . $row["apellidos"] . "</td>";
        echo "<td><form action=\"delete_user.php\" method=\"POST\">";
        echo "    <input class=\"form-btn\" name=\"borrar\" type=\"submit\" value=\"Borrar\" />";
        echo "    <input type='hidden' name='id' value=\"" . $row["ID"] . "\" />";
        echo "</form></td>";
        echo "</tr>";
    }
    echo "</table>";
} else {
    echo "0 results";
}

$conn->close();
?>
<br><br><br>
<form action="insert_user.php" method="POST">
<h2><em>Nuevo Usuario</em></h2><br>
<label for="nombre">Nombre <span><em>(requerido)</em></span></label>
<input type="text" name="nombre" class="form-input" required/><br>

<label for="apellido">Apellido</label>
<input type="text" name="apellido" /><br>

<label for="edad">Edad </label>
<input type="numeric" name="edad" /><br><br>
<input class="form-btn" name="insertar" type="submit" value="Insertar" />
</form><br><br>
</div>
</div>
</body>
</html>

```

Figura 21: Fichero index.php (V).

```

<?php
$servername = "192.168.10.77";
$username = "root";
$password = "root";
$dbname = "OpenNebula";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

$subs_nombre = utf8_decode($_POST['nombre']);
$subs_apellidos = utf8_decode($_POST['apellido']);
$subs_edad = utf8_decode($_POST['edad']);

$sql = "INSERT INTO Usuarios(nombre, apellidos, edad) values(\"$subs_nombre\", \"$subs_apellidos\", $subs_edad)";

if ($conn->query($sql) === TRUE) {
    header('Location: index.php');
} else {
    echo "Error: " . $sql . "<br>" . $conn->error;
}

$conn->close();
?>

```

Figura 22: Fichero insert-user.php .

```
<?php

    if(isset($_POST['id'])){
        $var=$_POST['id'];

        $servername = "192.168.10.77";
        $username = "root";
        $password = "root";
        $dbname = "OpenNebula";

        // Create connection

        $conn = new mysqli($servername, $username, $password, $dbname);

        // Check connection

        if ($conn->connect_error) {
            die("Connection failed: " . $conn->connect_error);
        }

        $sql = "DELETE FROM Usuarios WHERE ID = $var";

        if ($conn->query($sql) === TRUE) {
            header('Location: index.php');
        } else {
            echo "Error: " . $sql . "<br>" . $conn->error;
        }

        $conn->close();
    }

?>
```

Figura 23: Fichero delete_user.php .

En la segunda máquina virtual es donde reside el SGBD **MySQL** y por lo tanto es donde se ha creado la base de datos correspondiente sobre la que trabaja la aplicación. En dicha base de datos se ha creado la tabla **Usuarios**, la cual va a representar el modelo de datos para almacenar la información referente a los usuarios que se añadan o eliminen desde la aplicación. En las imágenes de a continuación se muestra la conexión hacia dicha máquina virtual así como el uso de la base de datos, la creación de la tabla y la insercción de algunos registros.

```
[mysql]> INSERT INTO Usuarios(nombre, apellidos, edad) values ('Jesus', 'Garcia Manday', 33);
Query OK, 1 row affected (0.05 sec)

[mysql]> INSERT INTO Usuarios(nombre, apellidos, edad) values ('Roberto', 'Perez Hinojosa', 23);
Query OK, 1 row affected (0.05 sec)

[mysql]> INSERT INTO Usuarios(nombre, apellidos, edad) values ('Luis', 'Alcala Torres', 19);
Query OK, 1 row affected (0.06 sec)

[mysql]> INSERT INTO Usuarios(nombre, apellidos, edad) values ('Antonio', 'Espejo Martinez', 28);
Query OK, 1 row affected (0.07 sec)

[mysql]> INSERT INTO Usuarios(nombre, apellidos, edad) values ('Carlos', 'Martin Fernandez', 28);
Query OK, 1 row affected (0.06 sec)
```

Figura 24: Insertando registros en la base de datos.

El resultado final de la aplicación lo podemos apreciar en la imagen de abajo.

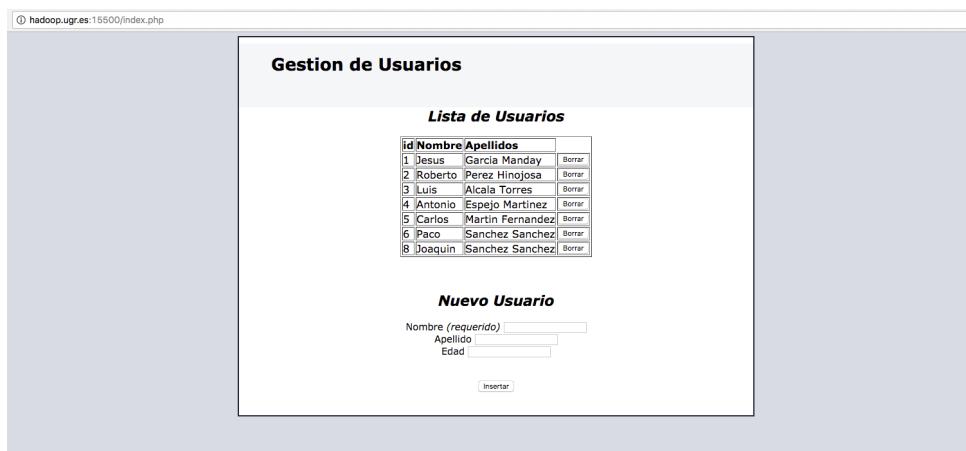


Figura 25: Aplicación web.

5. Breve manual de despliegue de las MVs.

Lo primero que se debe realizar es una conexión remota mediante el comando `ssh mcc48893432@docker.ugr.es` hacia el servidor docker.ugr.es donde se encuentra instalado el sistema **OpenNebula**. Una vez establecida la conexión, con la siguiente orden nos conectamos al sistema para poder trabajar con el.

```
jesusgarciamanday — mcc48893432@docker:~ — ssh mcc48893432@docker.ugr.es ·
[MacBook-Pro-de-Jesus:~ jesusgarciamanday$ ssh mcc48893432@docker.ugr.es
[mcc48893432@docker.ugr.es's password:
Last login: Tue Apr 18 13:15:02 2017 from 47.63.2.91
[[mcc48893432@docker ~]$ oneuser login mcc48893432 --ssh --force
[mcc48893432@docker ~]$
```

Figura 26: Conexión ssh y acceso al sistema OpenNebula.

Con la sesión iniciada correctamente en **OpenNebula**, pasamos a ver las imágenes de los sistemas operativos que hay disponibles en dicho sistema para crear las dos máquinas virtuales necesarias a través del comando `oneimage list`.

```
jesusgarciamanday — mcc48893432@docker:~ — ssh mcc48893432@docker.ugr.es ·
[[mcc48893432@docker ~]$ oneimage list
  ID USER      GROUP      NAME      DATASTORE      SIZE TYPE PER STAT RVMS
  8 oneadmin   users     CentOS-6.5-one- default    10G OS  No used  2
  9 oneadmin   users     CentOS-7     default    10G OS  No used  20
 10 oneadmin   users     Ubuntu-14.04 default    10G OS  No used  8
 19 oneadmin   users     CentOS-7_CC  default    10G OS  No used  3
 20 oneadmin   users     BigDataWebServ default  10G OS  No rdy   0
 21 oneadmin   users     _ Copy of BigData default  10G OS  No rdy   0
```

Figura 27: Imágenes de sistemas operativos disponibles en OpenNebula.

Como se ha podido comprobar en la anterior imagen, **OpenNebula** dispone de varias imágenes de sistemas operativos que difieren según el ámbito para el que se vayan a emplear. Para nuestro caso nos bastará con una imagen de un sistema operativo tipo **CentOS** o **Ubuntu**. Antes de ponernos a definir y crear las máquinas virtuales es necesario comprobar que se tiene creada una red virtual para ser capaz de lanzar máquinas dentro de un espacio de direcciones ip correcto. Esto lo podemos comprobar ejecutando el comando `onevnet list`.

ID	USER	GROUP	NAME	CLUSTER	BRIDGE	LEASES
0	oneadmin	oneadmin	private	-	br0	0
26	jmb	users	test_vnet	-	br0	0
44	manuelparra	users	manuelparra_vnet	-	br0	0
67	test	users	test_vnet	-	br0	0
128	mbd6722147	oneadmin	mbd6722147_vnet	-	br0	0
129	mbd45719008	oneadmin	mbd45719008_vnet	-	br0	0
130	mbd77144695	oneadmin	mbd77144695_vnet	-	br0	0
131	mbd76667211	oneadmin	mbd76667211_vnet	-	br0	0
132	mbd32052863	oneadmin	mbd32052863_vnet	-	br0	0
133	mbd0301815700	oneadmin	mbd0301815700_vnet	-	br0	0
134	mbd29209566	oneadmin	mbd29209566_vnet	-	br0	0
135	mbd74649004	oneadmin	mbd74649004_vnet	-	br0	0
136	mbdA0249591	oneadmin	mbdA0249591_vnet	-	br0	0
137	mbdA0634282	oneadmin	mbdA0634282_vnet	-	br0	0
138	mbd50765277	oneadmin	mbd50765277_vnet	-	br0	0
139	mbd76422399	oneadmin	migueldeporte_vnet	-	br0	1
140	mbd76658187	oneadmin	mbd76658187_vnet	-	br0	0
141	mbd0926349853	oneadmin	mbd0926349853_vnet	-	br0	0
142	mbd76420678	oneadmin	mbd76420678_vnet	-	br0	0
143	mbd26256679	oneadmin	mbd26256679_vnet	-	br0	0

Figura 28: Comprobación de red virtual (I).

ID	USER	GROUP	NAME	CLUSTER	BRIDGE	LEASES
208	x77374474J	users	Red_x77374474J	-	br0	0
209	openstack	users	OpenStack_NETS	-	br0	1
211	mcc1313253575	oneadmin	mcc1313253575_vnet	-	br0	0
212	mcc15472800	oneadmin	mcc15472800_vnet	-	br0	2
213	mcc15514810	oneadmin	mcc15514810_vnet	-	br0	1
214	mcc15517915	oneadmin	mcc15517915_vnet	-	br0	2
215	mcc209503323	oneadmin	mcc209503323_vnet	-	br0	0
216	mcc4423998	oneadmin	mcc4423998_vnet	-	br0	2
217	mcc45715895	oneadmin	mcc45715895_vnet	-	br0	0
218	mcc48893432	oneadmin	mcc48893432_vnet	-	br0	1
219	mcc70918176	oneadmin	mcc70918176_vnet	-	br0	2
220	mcc75570417	oneadmin	mcc75570417_vnet	-	br0	2
221	mcc75571687	oneadmin	mcc75571687_vnet	-	br0	1
222	mcc75575731	oneadmin	mcc75575731_vnet	-	br0	1
223	mcc75926571	oneadmin	mcc75926571_vnet	-	br0	2
224	mcc75928287	oneadmin	mcc75928287_vnet	-	br0	2
225	mcc75928532	oneadmin	mcc75928532_vnet	-	br0	2
226	mcc75933306	oneadmin	mcc75933306_vnet	-	br0	1
227	mcc76139799	oneadmin	mcc76139799_vnet	-	br0	2
228	mcc76439773	oneadmin	mcc76439773_vnet	-	br0	0
229	mcc76654141	oneadmin	mcc76654141_vnet	-	br0	2
230	ahilario	users	ahilario_vnet	-	br0	0
260	mdat76441609	oneadmin	mdat76441609_vnet	-	br0	0

Figura 29: Comprobación de red virtual (II).

En la **Figura 4** se puede apreciar como la red virtual con el ID **218** y el NAME **mcc48893432_vnet** está correctamente creada para el usuario **mcc48893432**. Llegado a este punto toca definir la plantilla en base a la cual se crearán las dos máquinas virtuales necesarias, una para el despliegue de la aplicación y otra para la base de datos. Ambas se crearán a partir de la misma plantilla, ya que los recursos, prestaciones y características que necesitan son similares. Posteriormente se aprovisionarán y configurarán de manera individual a cada una de ellas con las herramientas y servicios necesarios en función del rol que tenga asignado. Con el comando **onetemplate create** definimos la plantilla en función de los parámetros que le indiquemos como se puede ver en la imagen de a continuación.

```
[mcc48893432@docker ~]$ onetemplate create --name "template_Ubuntu" --cpu 1 --vcpu 1 --memory 512 --arch x86_64 --disk 10
--nic "mcc48893432_vnet" --vnc --ssh --net_context
ID: 654
```

Figura 30: Creación de plantilla de máquina virtual.

A la plantilla de máquina virtual creada se le ha dado el nombre de "template_Ubuntu", a la cual se le asignará una cpu y otra virtual. Tendrá **512 MB** de memoria **RAM** y estará basada en una arquitectura **x86_64**. Para el sistema operativo se ha seleccionada la imagen de un **Ubuntu 14.04**. La red virtual a la que va a pertenecer será la asignada este usuario (**mcc48893432_vnet**) y comprobada anteriormente. También se le añade la opción de poder acceder de manera remota desde ssh. Ejecutamos el comando **onetemplate list** para comprobar que efectivamente se ha creado con éxito.

```
[mcc48893432@docker ~]$ onetemplate list
  ID USER      GROUP      NAME          REGTIME
  654 mcc48893432  users      template_Ubuntu  04/21 11:33:54
```

Figura 31: Listado de plantillas de máquinas virtuales.

Una vez que se ha generado la plantilla de la máquina virtual, ya podemos comenzar con el despliegue de las dos máquinas virtuales. Para lanzar una instancia de una máquina virtual en base a una plantilla tenemos que ejecutar el comando **onetemplate instantiate** seguido del ID de la plantilla. Vamos a desplegar una primera máquina virtual que será la que contenga los servicios de **APACHE** y **PHP**, haciendo de servidor Web donde se alojará la aplicación.

```
[mcc48893432@docker ~]$ onetemplate instantiate 654
VM ID: 859
```

Figura 32: Despliegue de máquina virtual en base a una plantilla.

En la **Figura 7** se puede apreciar el despliegue de una máquina virtual en base a una plantilla, y como éste devuelve un identificador (**859**) que hace referencia a la máquina virtual creada. Comprobemos que la máquina virtual con ese identificador se ha creado correctamente ejecutando el comando **onevm list**.

```
[mcc48893432@docker ~]$ onevm list
  ID USER      GROUP      NAME          STAT UCPU   UMEM HOST        TIME
  849 mcc48893  users      Plantilla_Ubunt_poff  0    0K noded15  18d 15h27
  859 mcc48893  users      template_Ubuntu_runn  0    512M noded11  0d 00h03
```

Figura 33: Comprobando que se ha creado la primera máquina virtual.

Como se puede apreciar en la anterior imagen, la máquina virtual con el ID **859** se ha creado correctamente y se encuentra ejecutando, como muestra su estado '**running**'. Ahora vamos a desplegar otra instancia de máquina virtual en base a esa misma plantilla, la cual en esta ocasión será destinada al servicio de **MySQL**.

```
[mcc48893432@docker ~]$ onetemplate instantiate 654
VM ID: 860
```

Figura 34: Despliegue de una segunda máquina virtual en base a una plantilla.

En la imagen de arriba vemos que el despliegue de la segunda máquina virtual se ha realizado con éxito, ya que el sistema ha devuelto de identificador (**ID 860**) de la misma forma como sucedió con el despliegue de la primera máquina virtual. Al igual que hicimos antes, vamos a comprobar cual es el estado de esta nueva máquina virtual con el identificado **860** con ese mismo comando.

```
[mcc48893432@docker ~]$ onevm list
  ID USER      GROUP     NAME          STAT UCPU   UMEM HOST           TIME
  849 mcc48893 users    Plantilla_Ubunt poff   0     0K noded15  18d 16h26
  859 mcc48893 users    template_Ubuntu runn   0    512M noded11  0d 01h02
  860 mcc48893 users    template Ubuntu  runn   0    512M noded15  0d 00h03
```

Figura 35: Comprobando que se ha creado la segunda máquina virtual.

Se puede apreciar en la figura anterior que la segunda máquina virtual se ha desplegado correctamente y prueba de ello es el estado '**running**' que muestra al igual que la primera máquina virtual, lo que nos dice que ambas se encuentran ejecutando y listas para ser utilizadas.

6. Breve manual de uso de la aplicación web.

Como ya se mencionó en el apartado de la **Descripción de la aplicación web**, este software es una simple aplicación de gestión usuarios a través de la cual es posible añadir usuarios o eliminar usuarios de una base de datos. Cuando se ejecuta la aplicación lo que nos muestra es la lista de usuarios que existen en ese momento en la base de datos, de los cuales nos muestra atributos como su identificador, su nombre y sus apellidos.

ID	Nombre	Apellidos	
1	Jesus	Garcia Munday	Borrar
2	Roberto	Perez Hinojosa	Borrar
3	Luis	Alcala Torres	Borrar
4	Antonio	Espejo Martinez	Borrar
5	Carlos	Martin Fernandez	Borrar
6	Paco	Sanchez Sanchez	Borrar
8	Joaquin	Sanchez Sanchez	Borrar
10	Pedro	Mateos Pardo	Borrar
11	Alvaro	Aicon Izquierdo	Borrar

Figura 36: Estado de la aplicación web.

Justo al lado de cada usuario de la lista, en la última columna de la tabla, aparece un botón titulado 'Borrar' que nos permite realizar la acción de eliminar el usuario de esa fila como se ve en la imagen.

Gestion de Usuarios

Lista de Usuarios

id	Nombre	Apellidos	Borrar
1	Jesus	Garcia Manday	<input type="button" value="Borrar"/>
2	Roberto	Perez Hinojosa	<input type="button" value="Borrar"/>
3	Luis	Alcala Torres	<input type="button" value="Borrar"/>
4	Antonio	Espejo Martinez	<input type="button" value="Borrar"/>
5	Carlos	Martin Fernandez	<input type="button" value="Borrar"/>
6	Paco	Sanchez Sanchez	<input type="button" value="Borrar"/>
8	Joaquin	Sanchez Sanchez	<input type="button" value="Borrar"/>
10	Pedro	Mateos Pardo	<input type="button" value="Borrar"/>
11	Alvaro	Alcon Izquierdo	<input type="button" value="Borrar"/>

Nuevo Usuario

Nombre (requerido)
 Apellido
 Edad

Figura 37: Acción de borrar en la aplicación web.

Para hacer la prueba vamos a proceder a eliminar el último usuario de la lista para ver que efectivamente la lista ya muestra un usuario menos al ser eliminado de la base de datos.

Gestion de Usuarios

Lista de Usuarios

id	Nombre	Apellidos	Borrar
1	Jesus	Garcia Manday	<input type="button" value="Borrar"/>
2	Roberto	Perez Hinojosa	<input type="button" value="Borrar"/>
3	Luis	Alcala Torres	<input type="button" value="Borrar"/>
4	Antonio	Espejo Martinez	<input type="button" value="Borrar"/>
5	Carlos	Martin Fernandez	<input type="button" value="Borrar"/>
6	Paco	Sanchez Sanchez	<input type="button" value="Borrar"/>
8	Joaquin	Sanchez Sanchez	<input type="button" value="Borrar"/>
10	Pedro	Mateos Pardo	<input type="button" value="Borrar"/>
11	Alvaro	Alcon Izquierdo	<input type="button" value="Borrar"/>

Nuevo Usuario

Nombre (requerido)
 Apellido
 Edad

Figura 38: Borrar un usuario en la aplicación web (I).

Gestion de Usuarios

Lista de Usuarios

id	Nombre	Apellidos	
1	Jesus	Garcia Manday	Borrar
2	Roberto	Perez Hinojosa	Borrar
3	Luis	Alcala Torres	Borrar
4	Antonio	Espejo Martinez	Borrar
5	Carlos	Martin Fernandez	Borrar
6	Paco	Sanchez Sanchez	Borrar
8	Joaquin	Sanchez Sanchez	Borrar
10	Pedro	Mateos Pardo	Borrar

Nuevo Usuario

Nombre (requerido)

Apellido

Edad

Figura 39: Borrar un usuario en la aplicación web (II).

Vemos también como debajo de la tabla de los usuarios existe un formulario para poder registrar a un nuevo usuario en la aplicación. Este formulario solicita datos como el nombre (que es requerido), los apellidos y la edad. Si introducimos esos datos y pulsamos el botón titulado 'Insertar', veremos como la lista de los usuarios se actualiza con el nuevo registro insertado.

Gestion de Usuarios

Lista de Usuarios

id	Nombre	Apellidos	
1	Jesus	Garcia Manday	Borrar
2	Roberto	Perez Hinojosa	Borrar
3	Luis	Alcala Torres	Borrar
4	Antonio	Espejo Martinez	Borrar
5	Carlos	Martin Fernandez	Borrar
6	Paco	Sanchez Sanchez	Borrar
8	Joaquin	Sanchez Sanchez	Borrar
10	Pedro	Mateos Pardo	Borrar

Nuevo Usuario

Nombre (requerido)

Apellido

Edad

Figura 40: Insertar un usuario en la aplicación web (I).

The screenshot shows a web-based user management application. At the top, a header reads "Gestion de Usuarios". Below it, a section titled "Lista de Usuarios" displays a table of users:

ID	Nombre	Apellidos	Borrar
1	Jesus	Garcia Manday	Borrar
2	Roberto	Perez Hinojosa	Borrar
3	Luis	Alcala Torres	Borrar
4	Antonio	Espejo Martinez	Borrar
5	Carlos	Martin Fernandez	Borrar
6	Paco	Sanchez Sanchez	Borrar
8	Joaquin	Sanchez Sanchez	Borrar
10	Pedro	Mateos Pardo	Borrar
12	Mariano	Santaella Saez	Borrar

Below the table is a "Nuevo Usuario" (New User) form with three input fields: "Nombre (requerido)" (Required), "Apellido" (Last Name), and "Edad" (Age). A "Insertar" (Insert) button is located at the bottom right of the form.

Figura 41: Insertar un usuario en la aplicación web (II).

7. Descripción del proceso de instalación del S.O. (desde una ISO) en la MV.

El proceso de instalación de un sistema operativo desde una imagen ISO en una máquina virtual vamos a realizarlo desde el hipervisor **VirtualBox**. A través de este software interactivo basado en ventanas vamos a definir una máquina virtual e instalarle posteriormente el sistema operativo desde una image ISO.

La máquina virtual que se va a crear va a tener como sistema operativo un **Ubuntu** en la versión **14.04** de 64 bits y se llamará **vmUbuntu** como muestra la siguiente imagen.

This screenshot is identical to Figure 41, showing the "Gestion de Usuarios" application interface with the same user list and "Nuevo Usuario" form.

Figura 42: Creando máquina virtual (I).

Se le asignará **1 GB** para el tamaño de memoria RAM, así como **8 GB** de tamaño de disco (el cual se creará) de tipo **VDI** y reservado dinámicamente.

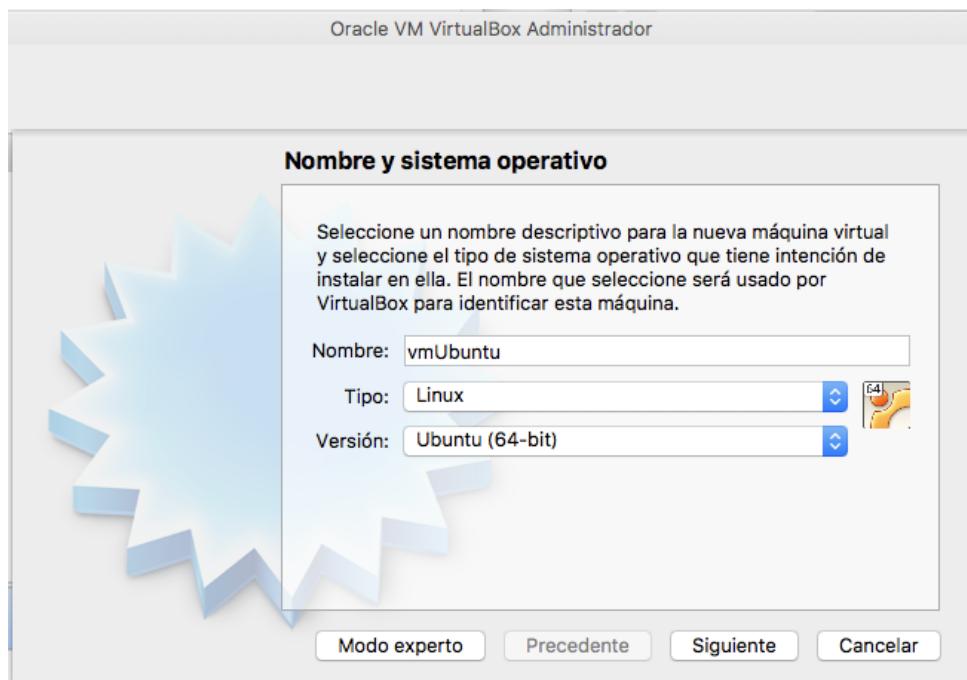


Figura 43: Creando máquina virtual (II).



Figura 44: Creando máquina virtual (III).

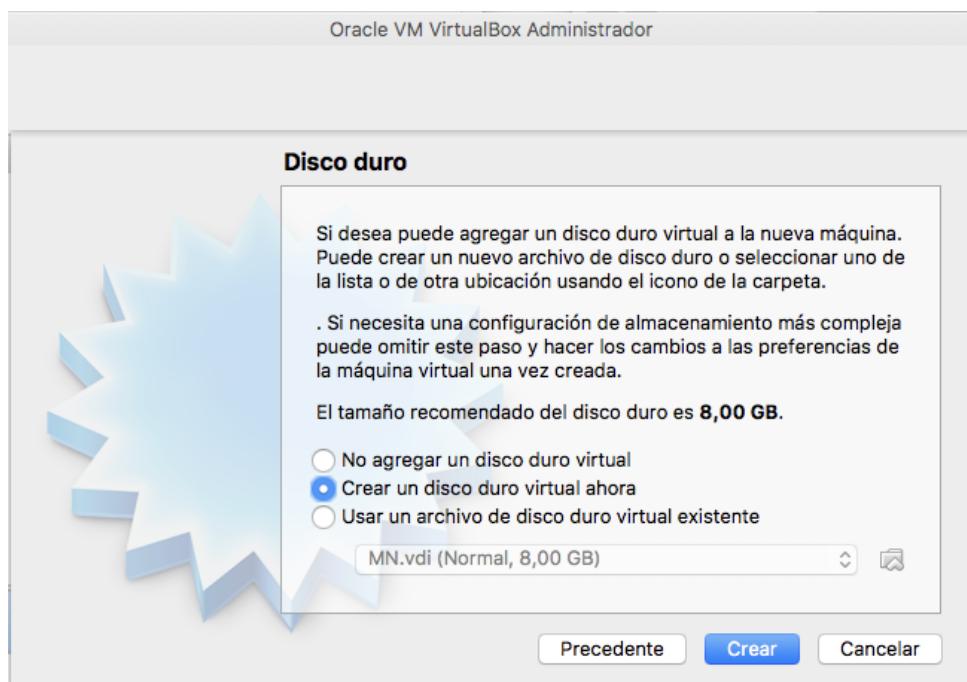


Figura 45: Creando máquina virtual (IV).

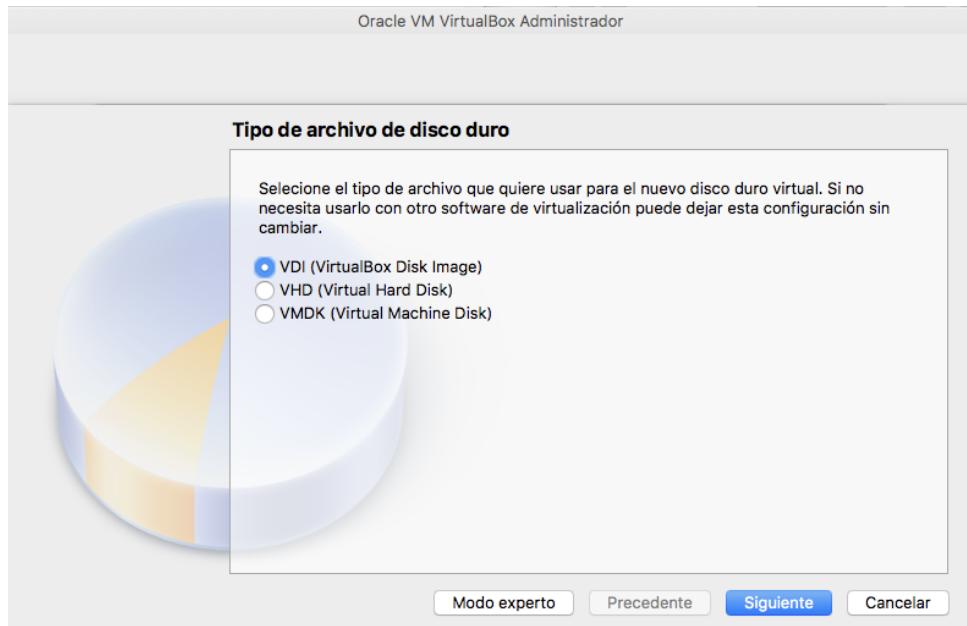


Figura 46: Creando máquina virtual (V).

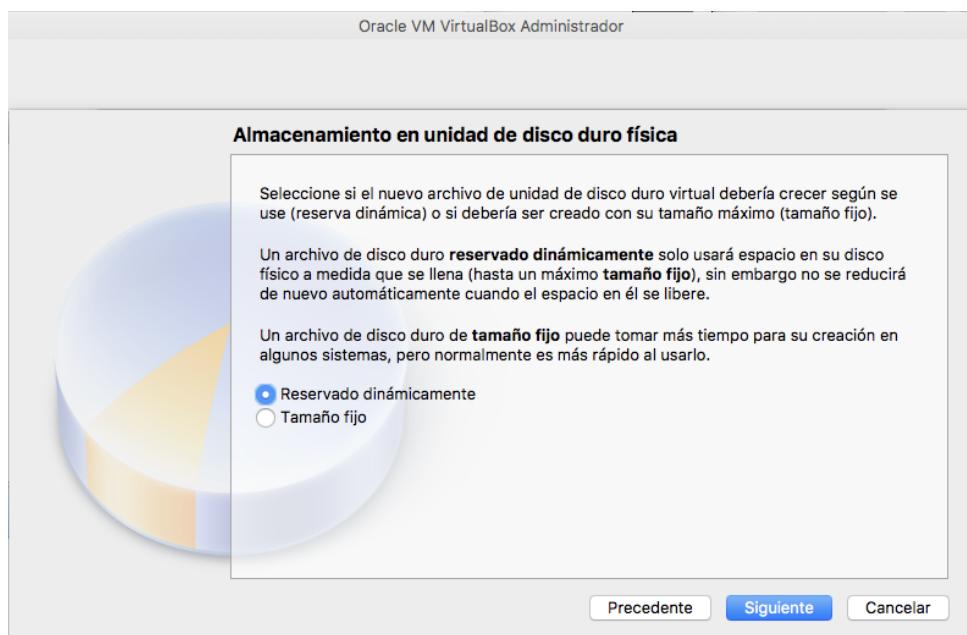


Figura 47: Creando máquina virtual (VI).

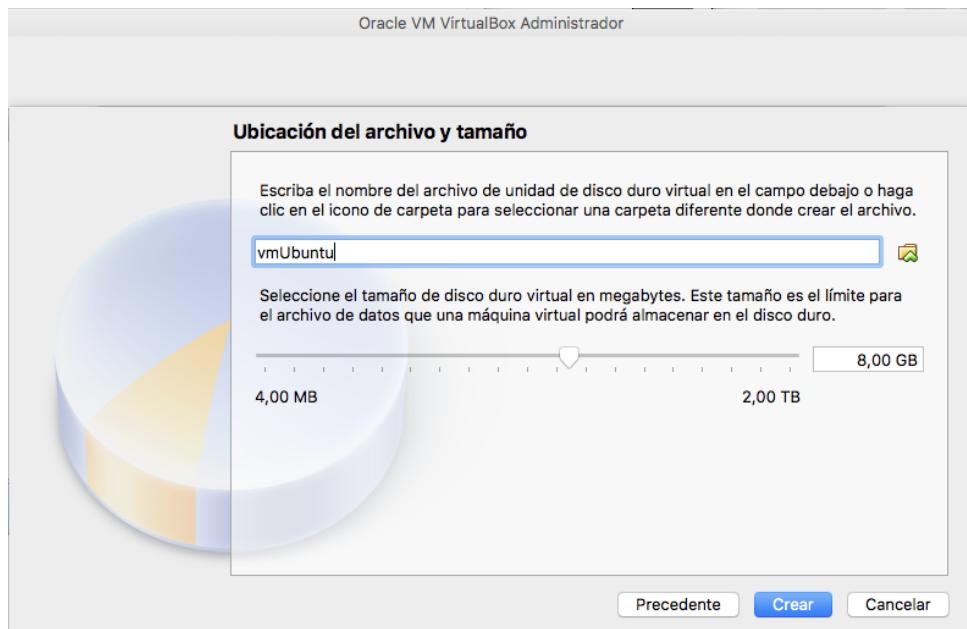


Figura 48: Creando máquina virtual (VII).

Una vez que la máquina virtual ha sido creada, el siguiente paso es la instalación del sistema operativo, el cual será tomado desde una imagen ISO seleccionada.

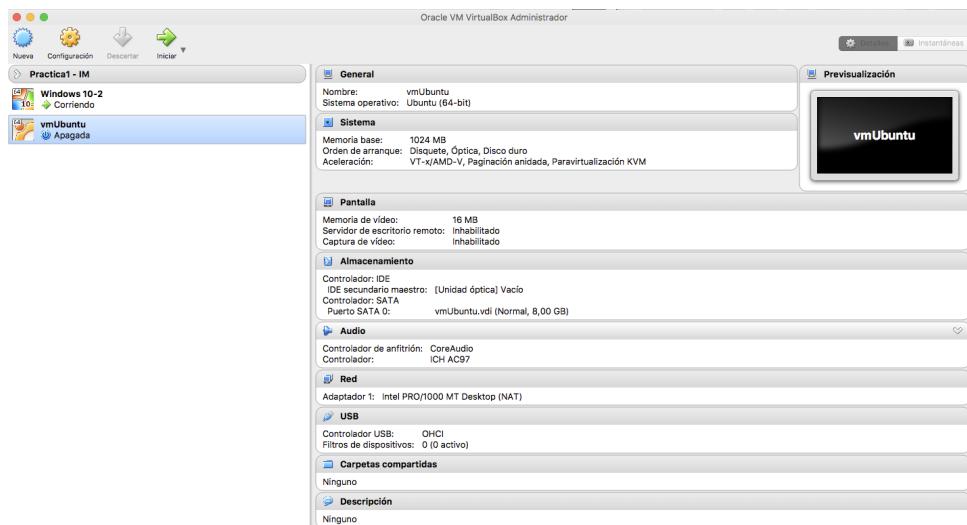


Figura 49: Máquina virtual creada.

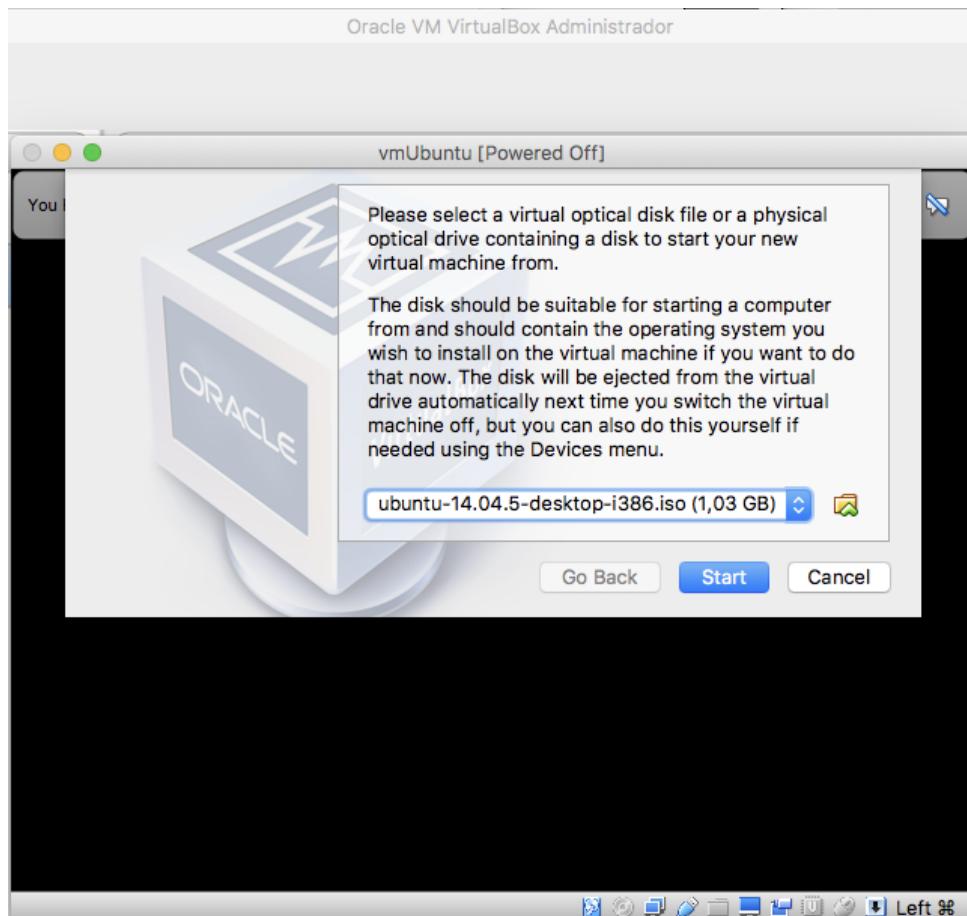


Figura 50: Instalación del S.O desde una imagen ISO (I).



Figura 51: Instalación del S.O desde una imagen ISO (II).

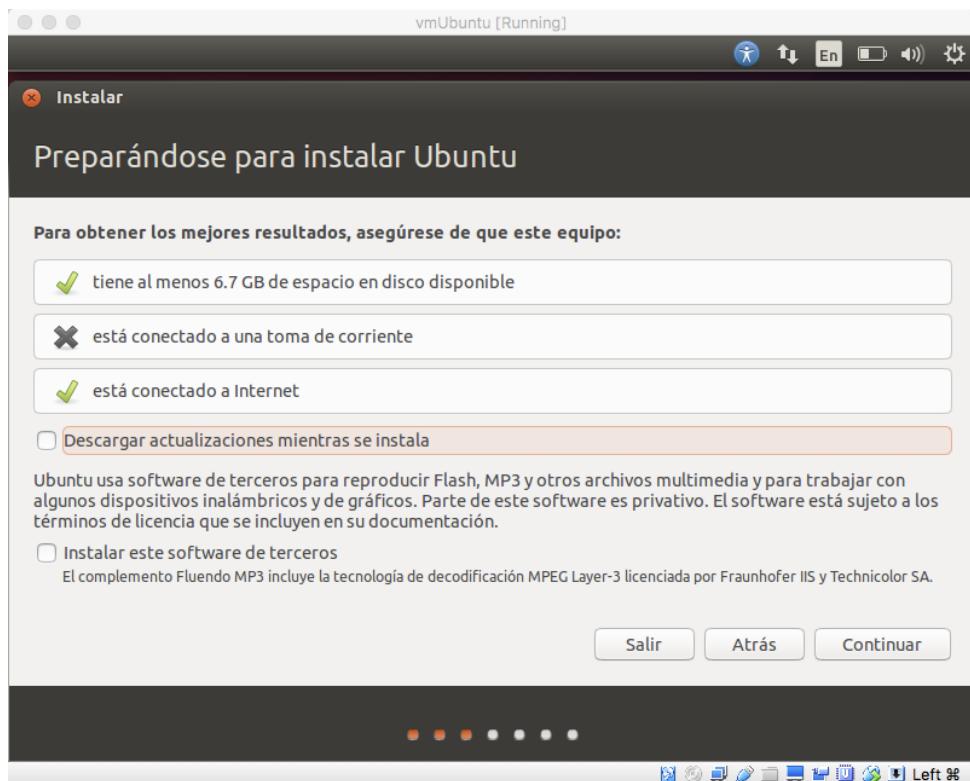


Figura 52: Instalación del S.O desde una imagen ISO (III).

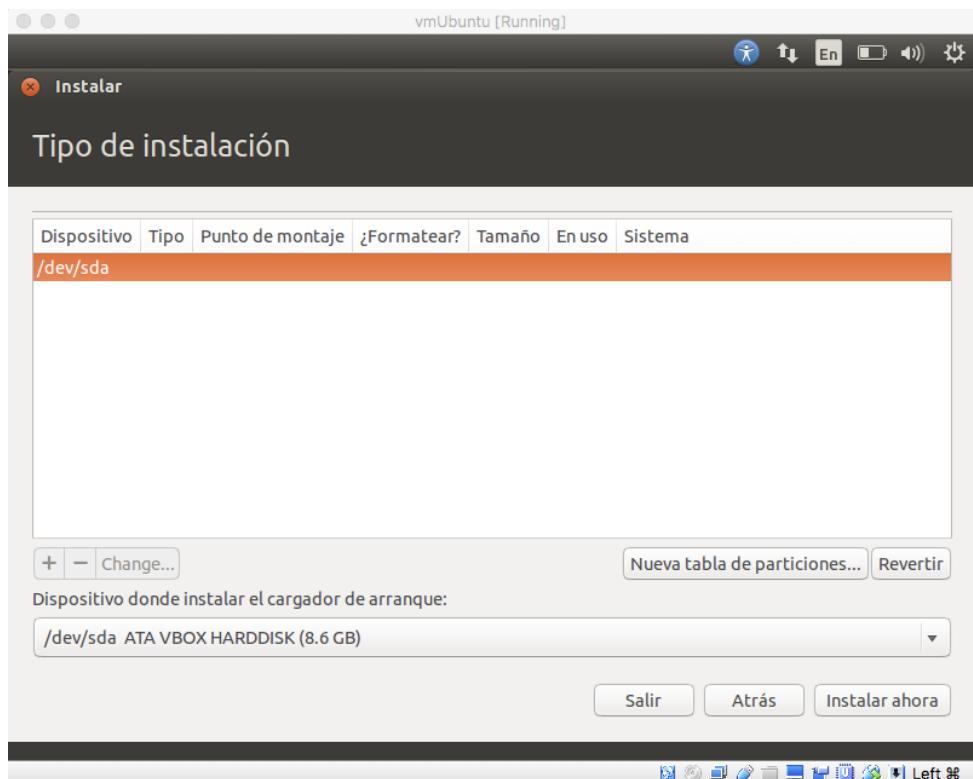


Figura 53: Instalación del S.O desde una imagen ISO (IV).

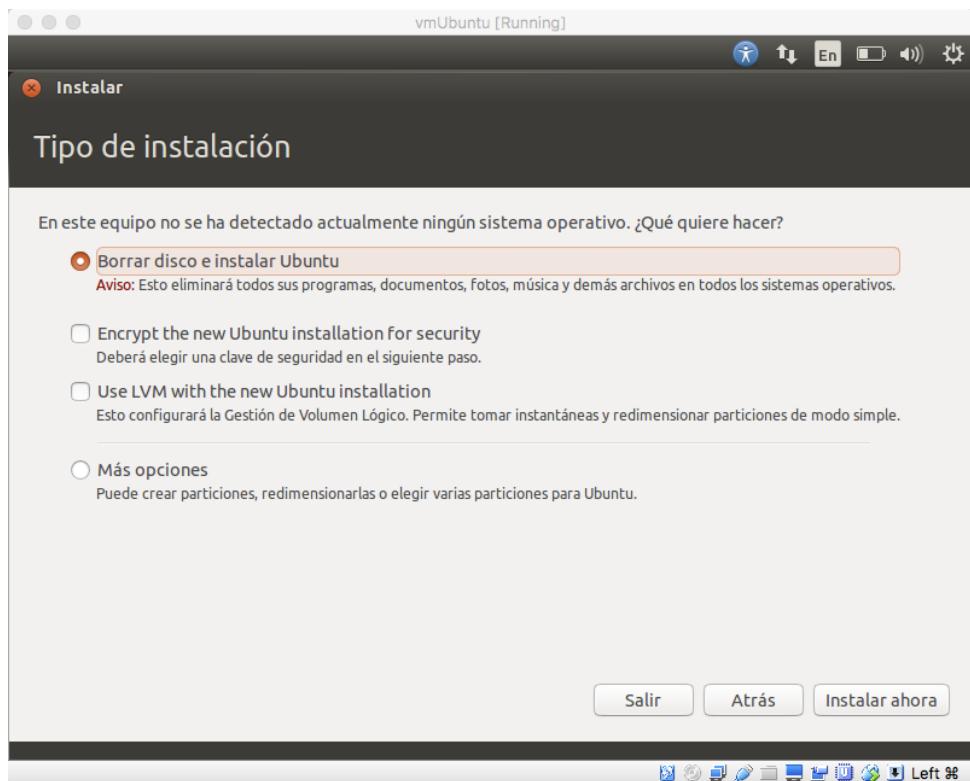


Figura 54: Instalación del S.O desde una imagen ISO (V).



Figura 55: Instalación del S.O desde una imagen ISO (VI).

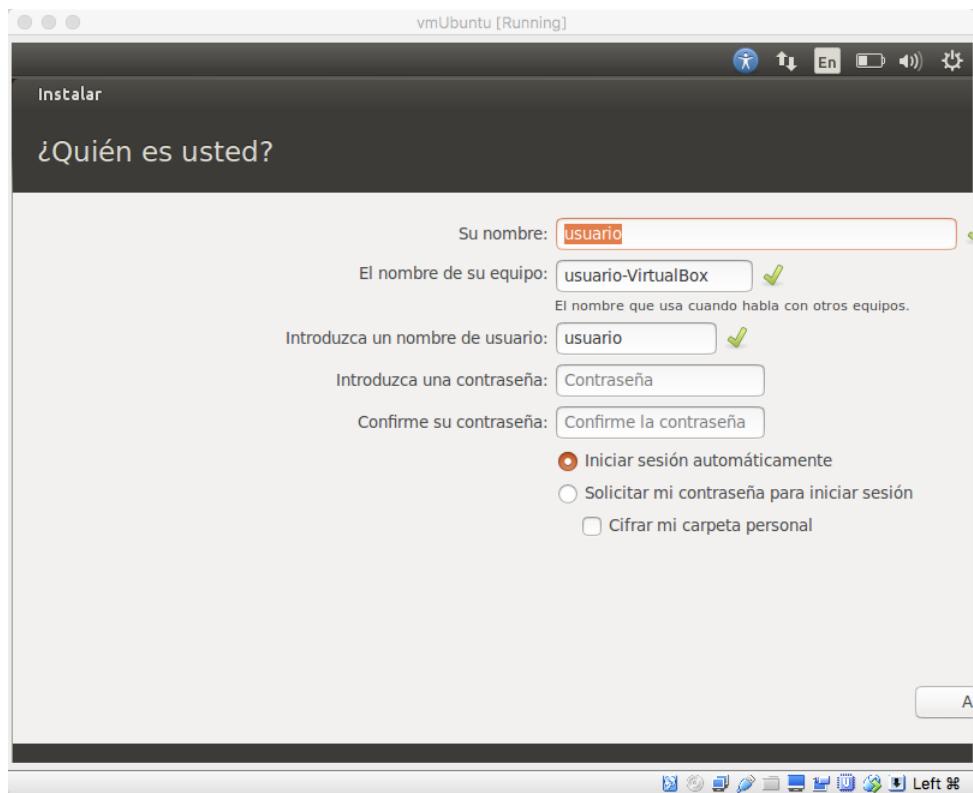


Figura 56: Instalación del S.O desde una imagen ISO (VII).



Figura 57: Instalación del S.O desde una imagen ISO (VIII).

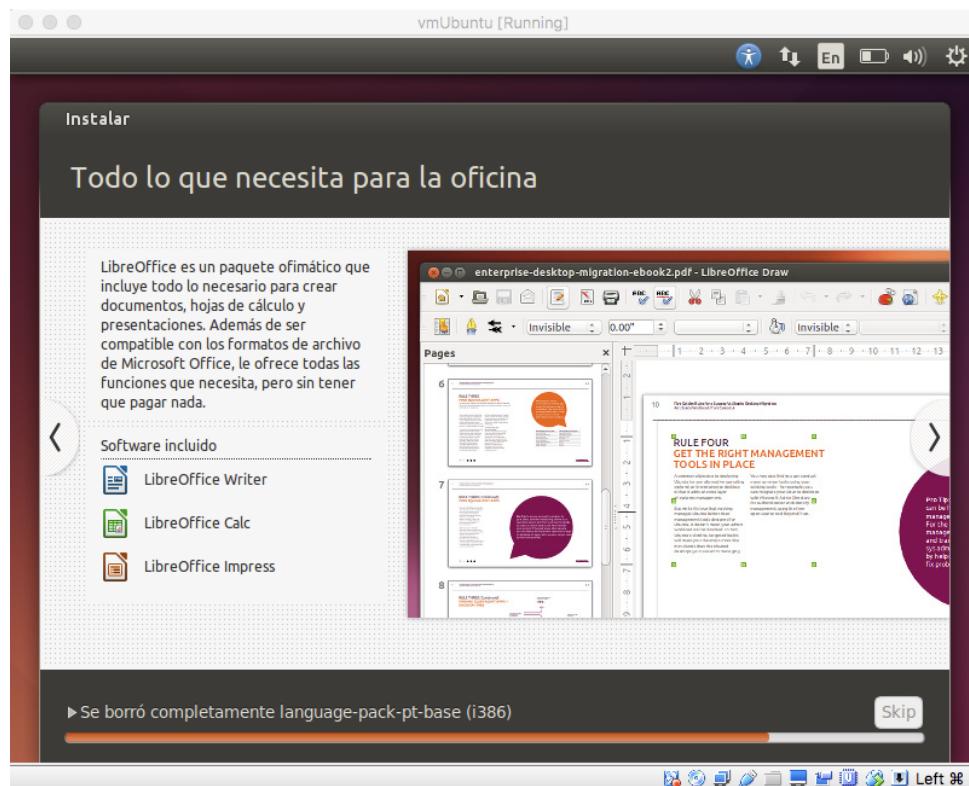


Figura 58: Instalación del S.O desde una imagen ISO (IX).

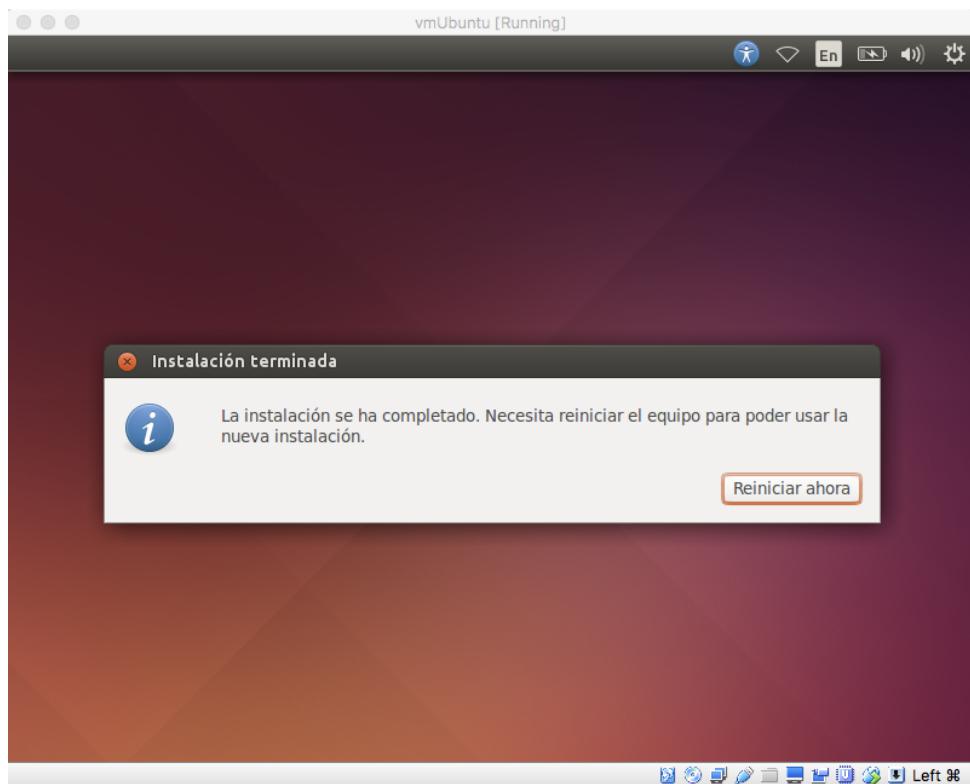


Figura 59: Instalación del S.O desde una imagen ISO (X).

Como se ha podido ver en las imágenes la instalación del sistema operativo en la máquina virtual desde una imagen **ISO** se ha realizado correctamente paso por paso y dicha máquina ya se encuentra disponible para su uso.

8. Descripción del proceso de instalación, configuración y despliegue de Owncloud.

Owncloud es un servidor de ficheros compartidos open-source que permite almacenar contenido personal tales como documentos y fotografías en una localización centralizada. En este apartado se van a mostrar los pasos para su instalación, configuración y despliegue utilizando la máquina virtual creada en el apartado anterior.

La máquina donde se va a instalar este herramienta debe cumplir con una serie de requisitos necesarios tales como un servidor web (**Apache** para este caso, una base de datos **MySQL** y **PHP** correctamente funcionando, así que lo primero será instalar todas esas herramientas.

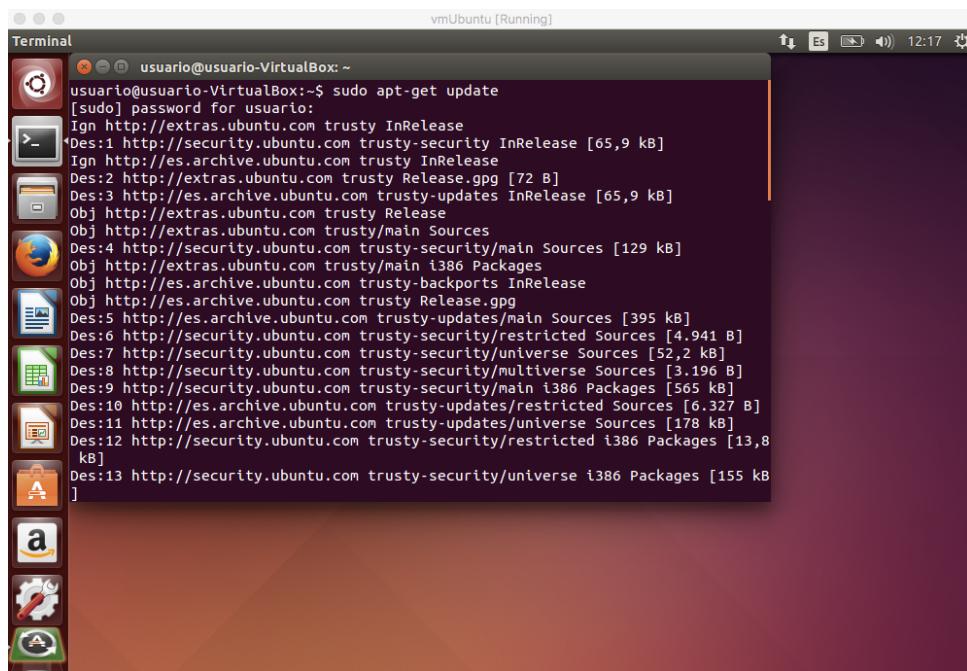


Figura 60: Actualización del sistema.

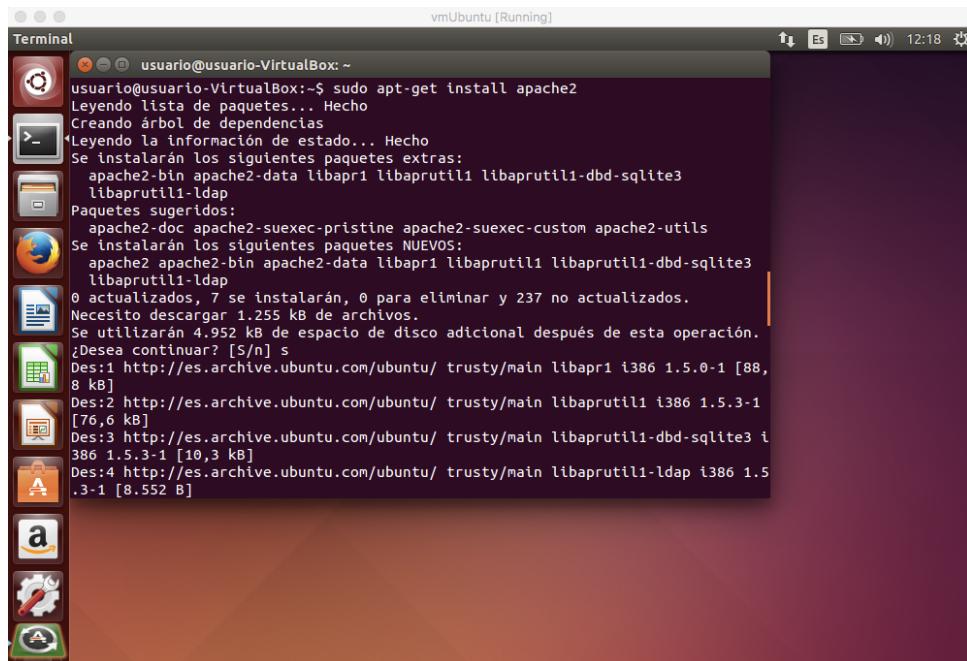


Figura 61: Instalación de Apache.

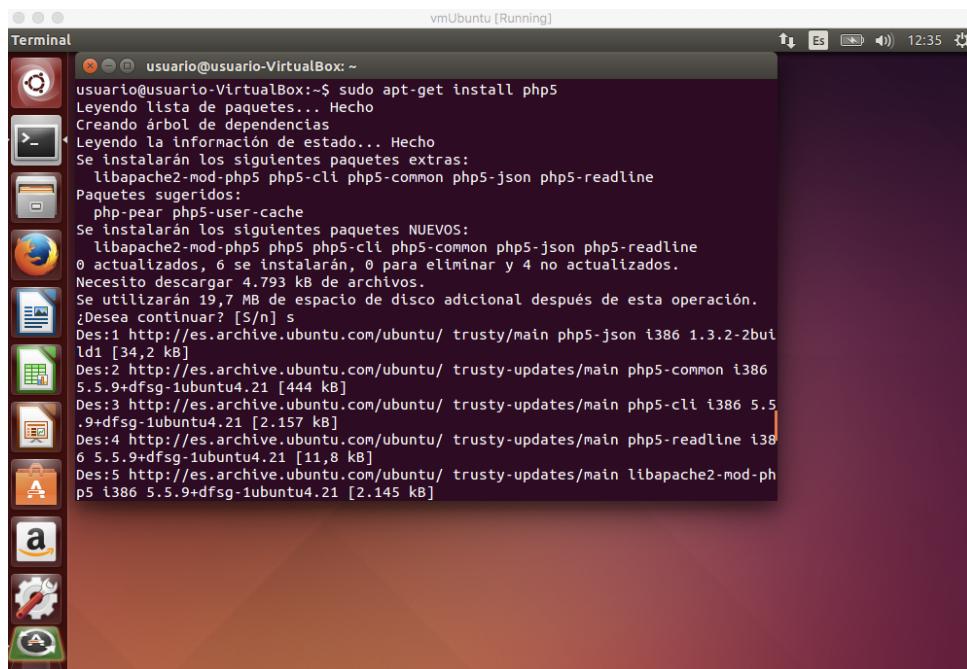


Figura 62: Instalación de PHP.

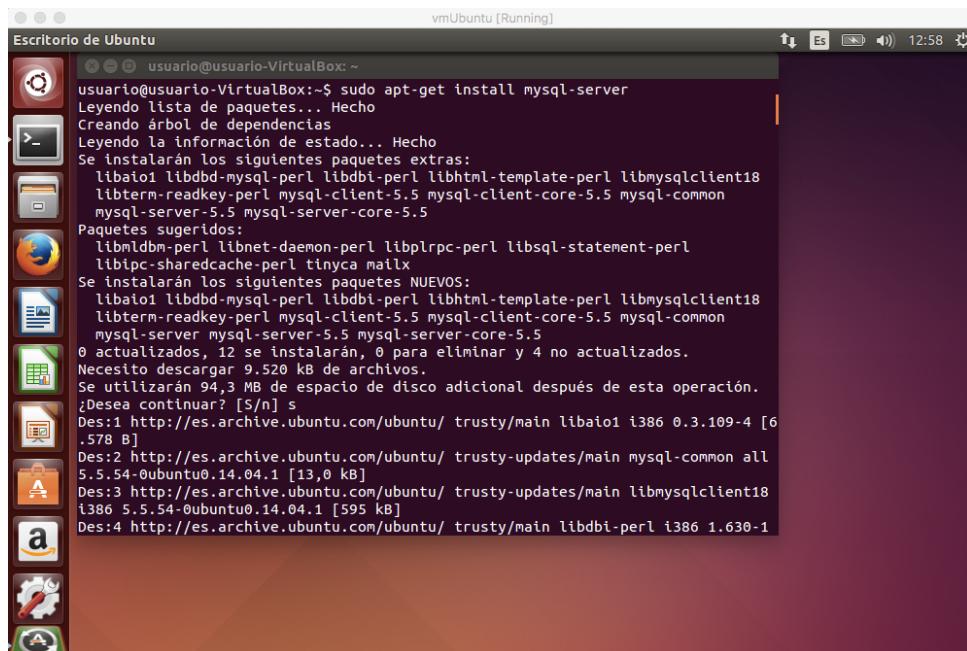


Figura 63: Instalación de MySQL.

Una vez instalados en la máquina virtual los requisitos necesarios para **Owncloud** pasamos a instalar la herramienta. Cabe indicar que no existe el paquete **Owncloud** dentro de los repositorios de Ubuntu, sino que mantiene su propio repositorio para la distribución, por lo que será necesarios descargar su **release key** y añadirlas con la utilidad **apt-key** para poder tener el fichero con la clave pública **PGP** que permita verificar la autenticidad del paquete **Owncloud**.

Realizando los siguientes pasos instalamos **Owncloud** en la máquina.

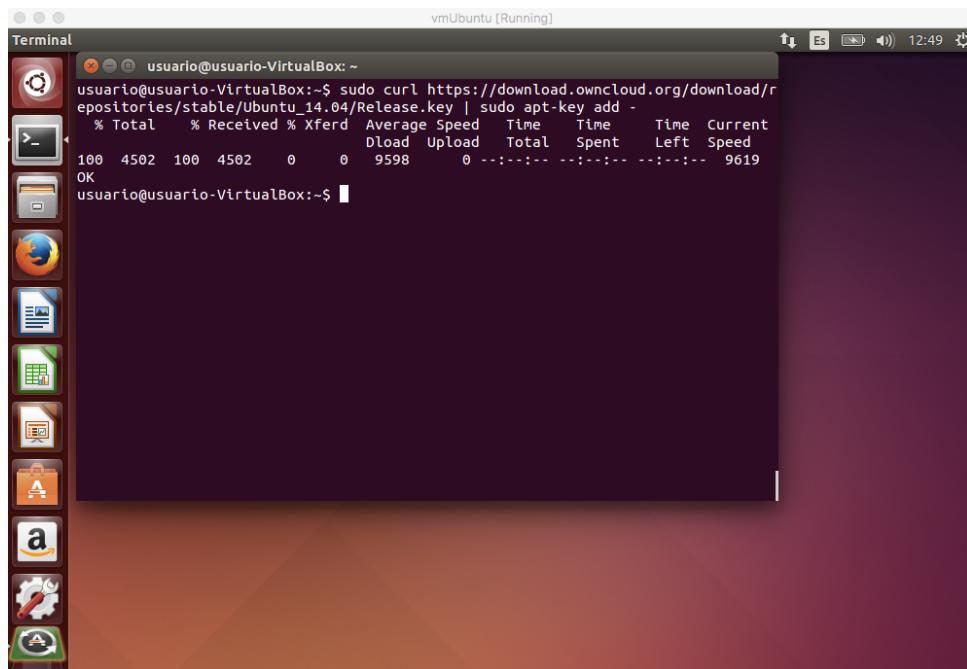


Figura 64: Instalación de **Owncloud** (I).

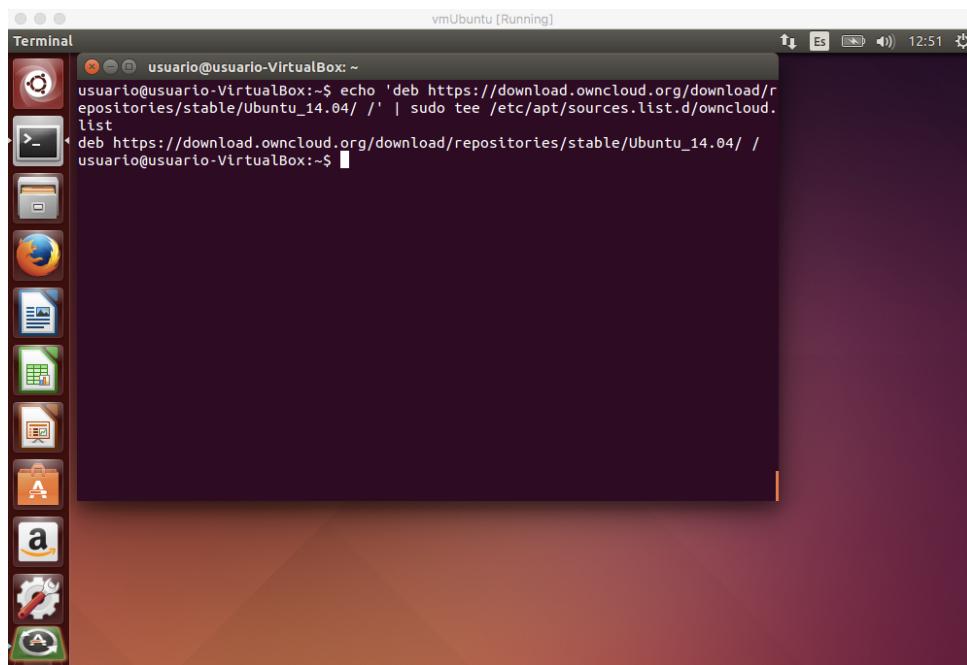
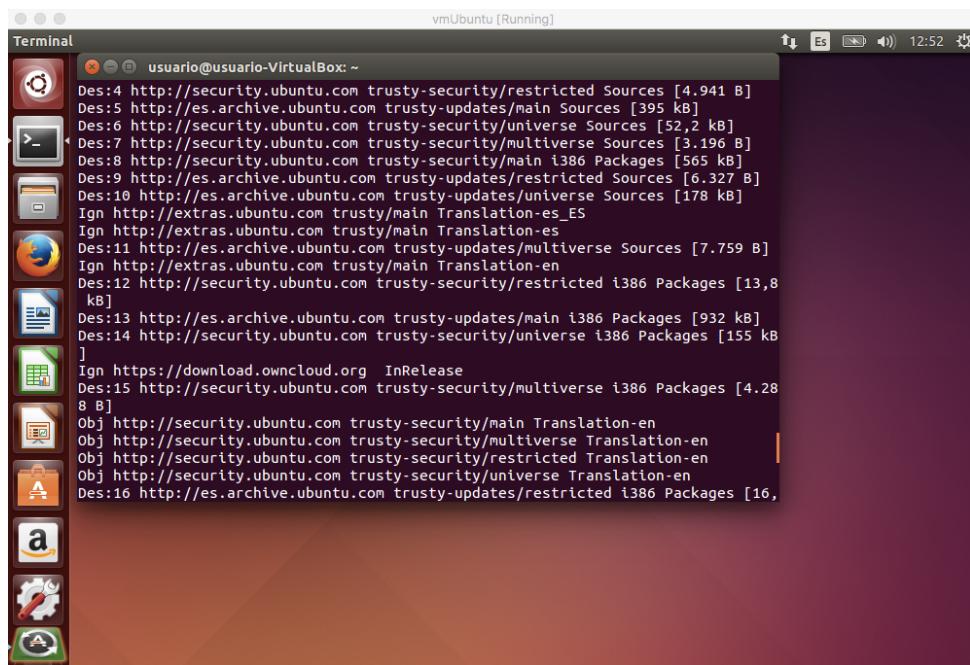
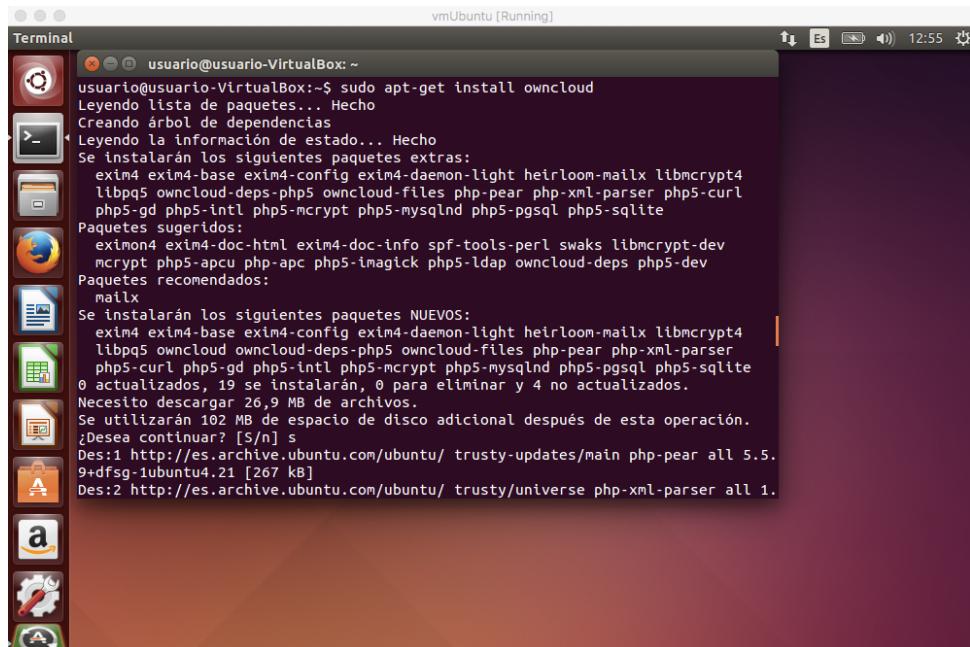


Figura 65: Instalación de **Owncloud** (II).

Figura 66: Instalación de **Owncloud** (III).Figura 67: Instalación de **Owncloud** (IV).

Con el paquete **Owncloud** instalado en la máquina virtual, el siguiente paso es configurar la base de datos **MySQL**. Es necesario crear una base de datos separada donde **Owncloud** almacenará los datos administrativos.

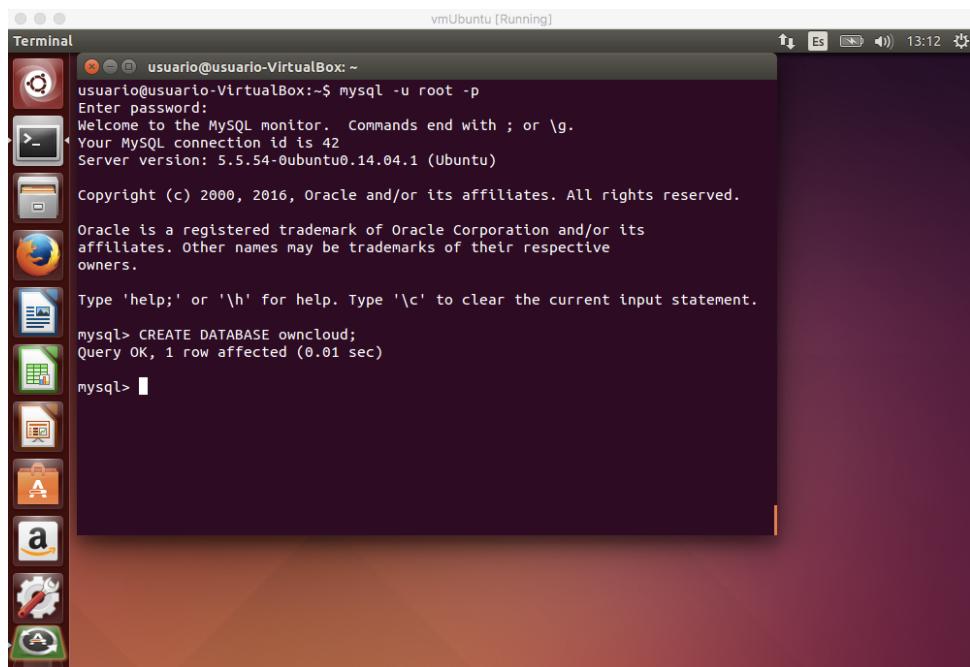


Figura 68: Configurando **Owncloud** (I).

Crearemos también una cuenta de usuario **MySQL** que interactuará con la base de datos creada anteriormente, siendo este el que la manejará.

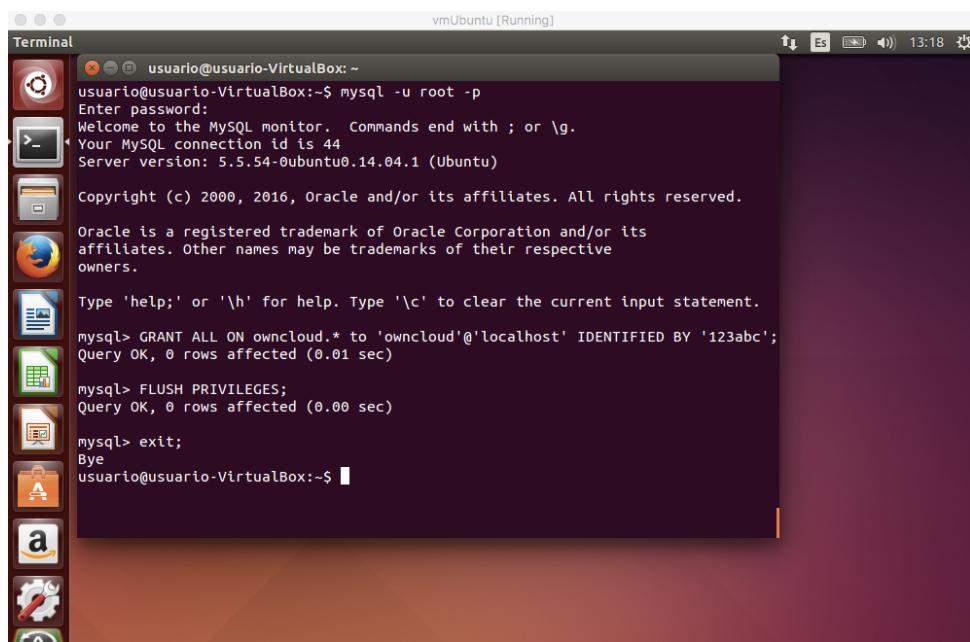


Figura 69: Configurando **Owncloud** (II).

Ya podemos desplegar el servidor **Owncloud** una vez que hemos finalizado la configuración. Para probar su funcionamiento accedemos desde el navegador a la dirección **http:localhost:owncloud**, accedemos dando las credenciales para una nueva cuenta de administrador y ya podemos comenzar a almacenar datos en nuestro

servidor.

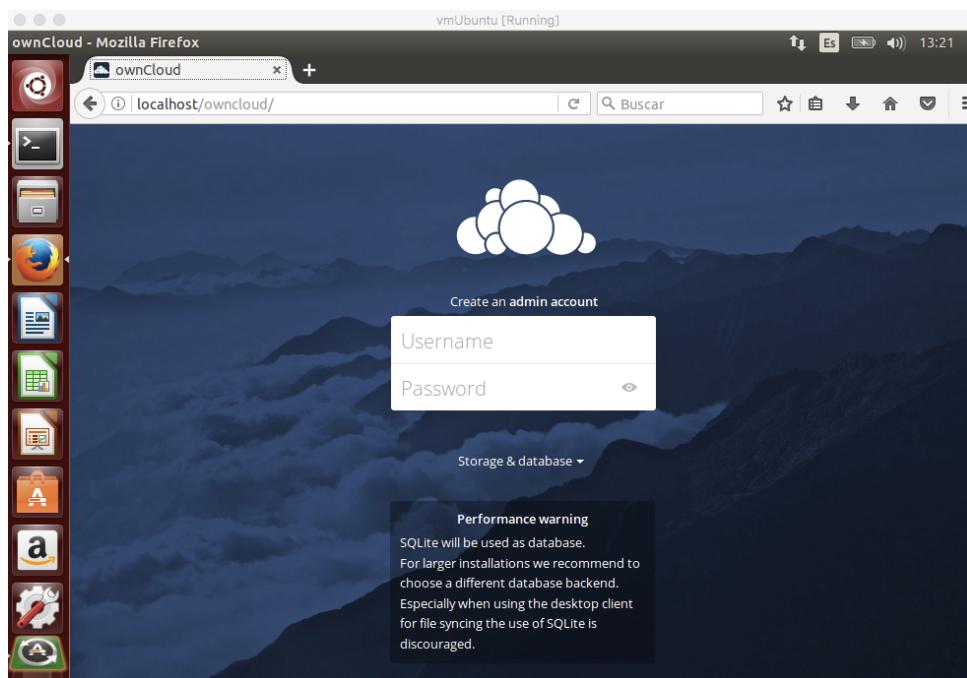


Figura 70: Desplegando **Owncloud** (I).

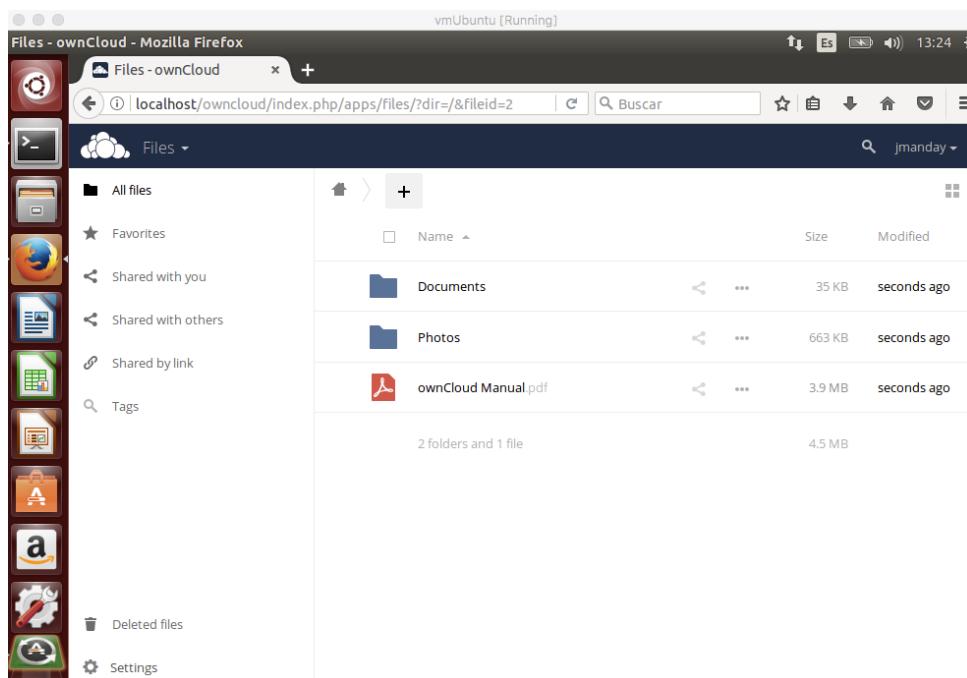


Figura 71: Desplegando **Owncloud** (II).

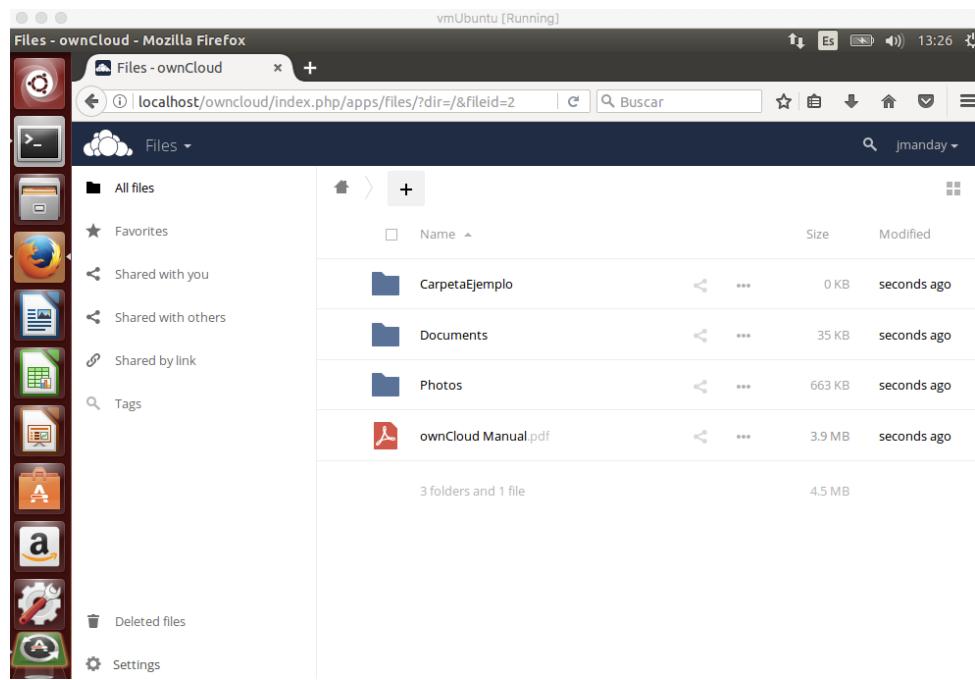


Figura 72: Desplegado Owncloud (III).

9. Referencias.

- Instalación, configuración y despliegue de Owncloud