Instalación AVANZADA de Apache, MySQL y PHP en Ubuntu 18 (LAMP)

Sí, ya lo sabemos, hay "ene" tutoriales al respecto pero bueno, aquí vamos de nuevo pero con el flamante Ubuntu 18.04 *Bionic Beaver:* sabemos que con estas versiones de largo plazo de Ubuntu no deberíamos tener nada de prisa pero hay que actualizar seis meses después de haber sido liberada la LTS. Para los viejitos como yo cantemos como Britney Spears en sus buenos tiempos: "*Hit me baby one more time!"*

https://www.youtube.com/watch?v=C-u5WLJ9Yk4

Abonando el terreno

Usamos una máquina de VirtualBox para hacer nuestras pruebas, descargamos una imagen ISO con Transmision GTK (Torrent) desde la página oficial de Ubuntu. Es una instalación mínima, la idea es que usar la básica, nada de LVM, RAID, multipath, vlans, etc.

Esto lo habíamos hecho hace tiempo , así que de una máquina almacenada la "resucitamos" y el aplicamos el ya famoso **apt-get update**:

Al finalizar procedemos a descargar e instalar las actualizaciones, que son bastantes:

Siempre es bueno descargar los programas desde los repositorios oficiales de cada distribución GNU/Linux ya que así se tiene una doble verificación del código a descargar. Solo en algunas oportunidades nos convendrá bajar estrictamente la última versión por alguna vulnerabilidad demasiado grave y que haya corregido (y ofrezca) el "fabricante" del software. **Demás está decir** (pero lo decimos igual) siempre descargar por protocolos seguros aunque en la imagen anterior esto no se cumple esto debemos de tener en cuenta desde que se instala Ubuntu vienen instaladas claves de seguridad que permiten verificar, después de descargar, si el contenido ha sido adulterado o modificado de alguna manera. Por último existen páginas web con HTTPS de donde podremos descargar los *hashs* que permiten verificar por MD5, o mejor aun con SHA, la autenticidad de los recursos descargados. La comunidad Apache explica muy bien (en idioma inglés) la verificación de firmas y contenido.

La segunda parte de la ecuación es mantenerse actualizado con los parches de seguridad, esto se maneja a nivel global en el sistema operativo pero siempre es bueno revisar si están habilitados.

Instalando Apache

Tan simple como introducir **apt-get install apache2**:

Pero como siempre puede haber inconvenientes, miren el resultado final:

De pronto recordé que en las actualizaciones instaladas nombraban a AMD e Intel en una sola línea, eso es raro, así que antes de ponerme a lanzar comandos (casi que lo hago) mejor simplemente reinicio (Ubuntu tiene un servicio que evita reinicios al instalar actualizaciones) pero ustedes tienen que irse por la vía "científica" para saber si una máquina Ubuntu necesita ser reiniciada, simplemente si existe un archivo: **cat /var/run/reboot-required**

Luego para conocer si funciona el servidor web (aparte de lanzar un navegador y colocar la dirección IP) debemos escribir: **sudo systemctl apache2.service**

Ahora debemos darle un nombre a nuestro servidor o, en su defecto, colocar simplemente la dirección IP que tiene asignad de una u otra manera (un buen artículo sobre la planificación y administración de direcciones IP lo pueden leer aquí):

Tomamos nota de la dirección IP y lanzamos **sudo nano /etc/apache2/apache2.conf** , con CTRL+W para buscar la línea que contenga "ServerName" si no lo conseguimos, pues la agregamos como es nuestro caso:

Por último verificamos que todo haya quedado bien con **sudo apache2ctl configtest** a lo cual debería responder **"Syntax OK**", reiniciamos el servicio con **sudo systemctl restart apache2**

En la wiki de Ubuntu recomiendan iniciar, reiniciar o detener el servicio apache con:

```
/etc/init.d/apache2 (start|restart|stop)
```

La diferencia es que este ultimo comando es específico para Apache mientras que **systemctl** sirve para todo software en el servidor y necesita derechos de usuario raíz.

Configurando el muro de fuego de Ubuntu: "Uncomplicated Fire

Wall" o UFW

El mismo nombre lo dice: cero complicaciones, tecleamos **sudo ufw app list** y si aparece tres líneas con la palabra Apache pasamos a revisar si los puertos abiertos son correctos con **sudo ufw app info "Apache Full"** y mostrará los puertos 80 para tráfico normal y 443 para encriptado (*deberemos adquirir un certificado digital de un tercero y tener una dirección IP pública, no como este ejemplo que tenemos una dirección IP privada de nuestro módem/enrutador de Internet*). Permitiremos entonces todo el tráfico con **sudo ufw allow in "Apache Full"** y se aplicarán las reglas para ambos protocolos, IPv4 e IPv6:

Comprobando nuestro servidor web

Pues abrimos un navegador en una máquina que esté en nuestra red de área local , en nuestro caso, colocamos 192.168.1.60 y nos mostrará el celebérrimo mensaje de bienvenida. Por si las dudas volvemos a actualizar los repositorios y notamos que unas actualizaciones se niegan a ser instaladas así que ejecutamos **sudo apt-get dist-upgrade** el cual se ocupa de revisar a fondo los paquetes que tengan interrelación, lo que llaman dependencias:

Y volvemos a revisa si el servidor necesita ser reiniciado, revisen los pasos anteriores.

Instalando MySQL

Sencillamente **sudo apt-get install mysql-server** y descargamos 21 megabytes para esta versión de esta base de datos. *A lo largo de la instalación aparecerá un cuadro de diálogo requiriendo una contraseña de usuario raíz* ***que por seguridad no debe ser la misma del usuario raíz de GNU/Linux Ubuntu. Anótenla en lugar seguro, por favor. Dado el caso no pregunte contraseña alguna, vamos al siguiente punto.***

En nuestro caso, por alguna razón, la máquina con la que hacemos muchas pruebas de software, quedó configurada para que *cada usuario acreditado en Ubuntu pueda entrar en MySQL sin contraseña alguna* (cuando tratamos de conectarnos con **mysql -u root -p** arroja el "ERROR 1698 (28000): Access denied for user 'root'@'localhost'"). Obviamente como acabamos de instalarlo, ningún usuario está agregado así que el único usuario de MySQL es el *root* de Ubuntu así que hacemos **sudo su** para ejecutar lo siguiente:

Escogeremos el complemento para que MySQL verifique la complejidad de nuestras contraseñas en nivel medio (metemos "2"): ocho o más caracteres en minúsculas y mayúsculas además de

caracteres especiales. **Es en este momento que preguntará una nueva contraseña para 'root' y nos mostrará el nivel de complejidad en una escala del 1 al 100.**

A continuación preguntará si deseamos deshabilitar los usuarios anónimos (le decimos que sí, "Y"), y si deseamos borrar la base de datos de prueba que trae para aprendizaje y le decimos que también, que la borre. luego pregunta si desea cargar los nuevos valores (ellos lo llaman 'privilegios') ¡y listo, ya aseguramos la base de datos! **¡El problema ahora es que no podemos conectarnos siempre con "sudo su" para poder acceder por línea de comando a MySQL!**

Por eso es que haremos lo siguiente: haremos **sudo su**, nos conectaremos a MySQL y no nos pedirá contraseña porque la autenticación está basada en las credenciales de usuario ante Ubuntu; pasamos a listar los usuarios de la base de datos:

Procedemos a agregarnos como *usuario de poder* (recuerden que la contraseña debe ser compleja, lo exige debido a que ya aplicamos el comando **mysql_secure_installation**), miren los comandos:

Luego nos desconectamos como 'root' y con nuestra clave de usuario podremos entrar en MySQL si contraseña y volvemos a listar a los usuarios registrados:

> El usuario **root** de MySQL es distinto y no tiene nada que ver con el usuario **root** de Ubuntu, sin embargo ya vimos que podemos configurar para que sean iguales. Queda a criterio de cada quien sobre como quiere trabajar, ¡eso es lo bueno del software libre!

Ahora lo que haremos es colocar al **root** de MySQL de manera independiente del **root** de Ubuntu así que le modificamos para que la autenticación sea por **mysql_native_password**:

Antes de continuar debemos pensar acerca de otro detalle: cómo cambiar la contraseña del usuario **root** de MySQL si la perdemos o se nos olvida, pongan atención:

- Detenemos el servicio MySQL.
- Iniciamos el servicio con dos parámetros: **--skip-grant-tables** para que no pregunte contraseña alguna y **--skip-networking** para evitar que ningún usuario remoto se conecta

a la base de datos desprotegida, solo podremos maniobrar nosotros por consola.
- Nos conectamos a MySQL y cambiamos la contraseña (recordar utilizar complejidad media).
- Salimos y detenemos el servicio de nuevo.
- Iniciamos el servicio de manera normal, sin parámetro alguno.
- **Verificamos si funciona la contraseña que acabamos de establecer, de esta manera podremos conectarnos como "root" de MySQL por medio de contraseña.**
- Vuelvan a leer todo de nuevo, es complejo y debemos practicarlo, miren como nos quedó en la siguiente imagen.

Instalando PHP

Que esto va de instalar el lenguaje PHP, sí, pero además instalaremos unas librerías para que Apache funcione. Hace tiempo ya publicamos, por ejemplo, un artículo sobre cómo habilitar CAPTCHA de manera sencilla, y es un excelente ejercicio para realizar luego y que demuestra el uso de librerías gráficas adicionales. Pero volvamos al trabajo de hoy:

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

Al terminar de instalar PHP vamos a configurar para Apache cuando le soliciten un directorio en vez de un fichero específico, comience a buscar *en primer lugar un archivo* ***index.php*** para ello modificamos el archivo **sudo nano /etc/apache2/mods-enabled/dir.conf**:

Luego reiniciamos el servidor Apache y en seguida verificamos de que está funcionando correctamente:

La cantidad de librerías para PHP funcione con Apache2 es impresionante, podremos obtener información de cualquiera de ella introduciendo la siguiente orden: **apt-cache show *nombre_del_paquete***

Para comprobar el estado de PHP haremos un fichero llamado **/var/www/html/info.php** y agregaremos la siguiente información:

Abrimos un navegador web y vamos a nuestra dirección personalizada que escogimos para esta entrada: **http://192.168.1.60/info.php**

Una vez confirmado que todo funcione como es debido, procedemos a borrar el fichero, por razones de seguridad **rm /var/www/html/info.php** (el hecho de que estemos trabajando con máquinas de prueba *nunca debemos bajar la guardia y siempre trabajaremos de manera correcta, ¡incluso en las prácticas!*).

Fuentes consultadas:

En idioma castellano:

- «Fortificando un Servidor Apache (I de IV)» por Chema Alonso.
- «Fortificando un Servidor Apache (II de IV)» por Chema Alonso.

En idioma inglés:

- «How To Install Linux, Apache, MySQL, PHP (LAMP) stack on Ubuntu 16.04» by Brennen Bearnes.
    - «How To Install and Secure phpMyAdmin on Ubuntu 18.04» by Mark Drakke.
- «How To Secure Apache with Let's Encrypt on Ubuntu 16.04» by Erika Heidi.