

# DESPLIEGUE AUTOMATIZADO

**Alumno:** Daniel Tamargo Saiz

**Máquina utilizada:** Ubuntu Server 20.04

**Redes:** NAT y Adaptador Puente (ambas con DHCP)

**Objetivo:** Ejecutar una serie de órdenes al actualizar un repositorio en GitHub, es decir, que al hacer push a este a través de los Hooks se ejecuten acciones en el servidor asociado.

## Instalamos las herramientas y aplicaciones necesarias

Actualizamos las dependencias:

```
sudo apt update
```

Instalamos y preparamos LAMP:

```
sudo apt install apache2 -y
```

```
sudo apt install mysql-client libmysqlclient-dev -y
```

```
sudo apt install software-properties-common -y
```

```
sudo add-apt-repository ppa:ondrej/php
```

```
sudo apt update
```

```
sudo apt install libapache2-mod-php7.2 php7.2 php7.2-mysql -y
```

```
sudo apt install php7.2-mbstring php7.2-curl php7.2-intl php7.2-gd php7.2-zip -y
```

```
sudo a2enmod rewrite
```

```
sudo a2enmod actions
```

```
sudo service apache2 restart
```

```
sudo apt install mysql-server -y
```

```
sudo service mysql start
```

*Nota: modificamos el fichero apache2-dir.conf para indicar que busque primero index.php, también creamos un index.php para comprobar que funciona correctamente [como se puede apreciar en esta foto](#).*

Instalamos y preparamos Git:

```
sudo apt install git -y
```

```
git config --global user.name "ServidorDani"
```

```
git config --global user.email "servidordani@gmail.com"
```

Preparar las claves que utilizará GitHub para acceder a nuestro servidor:

```
sudo mkdir /var/www/.ssh
```

```
sudo chown -R www-data:www-data /var/www/.ssh
```

```
cd /var/www/.ssh
```

Generar las claves:

`sudo -Hu www-data ssh-keygen -t rsa`  
(crearla sin contraseña)

```
Despliegue Automatizado Ubuntu_S 20 [Corriendo] - Oracle VM VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
jefe@jefe:/var/www/.ssh$ sudo -Hu www-data ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/var/www/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /var/www/.ssh/id_rsa
Your public key has been saved in /var/www/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:Lt3VrFR400Nt5bYBuKfnKTTYUfphRqnhq1q1b10AQM4 www-data@jefe
The key's randomart image is:
+----[RSA 3072]-----+
|      .O. . . .+      |
|      O O .000      |
|      E  + =++      |
|      o @ o+       |
|      S ..@ +.      |
|      o O.@.* .     |
|      . O.=.% +     |
|      .. .= =       |
|      ...O..        |
+----[SHA256]-----+
jefe@jefe:/var/www/.ssh$
```

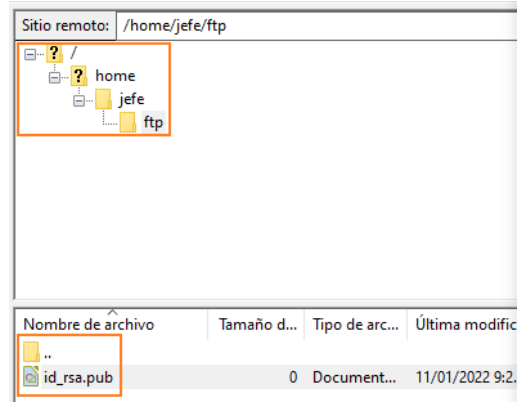
Nos pasamos la clave al host:

Instalamos un servicio FTP en la máquina virtual para acceder y poder descargar la clave pública generada.

`sudo apt install vsftpd -y`

Y utilizamos FileZilla para acceder y descargar dicha clave.

Sitio remoto: /home/jefe/ftp




Nombre de archivo	Tamaño d...	Tipo de arc...	Última modific...
..			
id_rsa.pub	0	Document...	11/01/2022 9:2...

```
inet6 fe80::a00:27ff:fe2d:f9d/64 scope link
valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state
0
link/ether 08:00:27:2d:0f:9d brd ff:ff:ff:ff:ff:ff
inet 10.100.14.59/24 brd 10.100.14.255 scope global dynamic enp0s8
valid_lft 2772sec preferred_lft 2772sec
inet6 fe80::a00:27ff:fe2d:f9d/64 scope link
valid_lft forever preferred_lft forever
jefe@jefe:~$ cd /var/www/.ssh/
jefe@jefe:/var/www/.ssh$ sudo mkdir /home/jefe/ftp
jefe@jefe:/var/www/.ssh$ sudo cp id_rsa.pub /home/jefe/ftp/id_rsa.pub
jefe@jefe:/var/www/.ssh$ _
```

Añadimos la clave SSH a nuestra cuenta de GitHub:

Accedemos a <https://github.com/settings/ssh/new>

Le ponemos el nombre que deseemos y copiamos el contenido de la clave y lo pegamos.



**Daniel Tamargo**  
Your personal account

[Go to your personal profile](#)

Account settings

Profile

Account

Appearance

Accessibility

Account security

Billing & plans

Security log

## SSH keys

New SSH key

This is a list of SSH keys associated with your account. Remove any keys that you do not recognize.



**Reto3ClaveDespliegueAutomatizado**  
SHA256:eLfWPEebbJxILj2CxxjPo1RNM7ReKL  
TtmL0o475FbM  
Added on 11 Jan 2022  
Never used — Read/write

Delete

Check out our guide to [generating SSH keys](#) or troubleshoot [common SSH problems](#).

## Instalar ngrok:

Como nuestra máquina virtual no está de cara a la red pública y no dispone de un dominio, necesitamos apoyarnos de alguna herramienta para que el repositorio de GitHub pueda ejecutar ese trigger sobre nuestra máquina.

Utilizaremos [ngrok](#), que básicamente creará un túnel que conectará la URL que nos generará con nuestra máquina virtual, haciendo que GitHub pueda llegar. Aprovechando el servicio FTP que hemos instalado y para no copiar la URL de descarga a mano, lo descargamos en la máquina Host y lo subimos mediante FTP.

*Nota: si tenemos dudas de qué kernel y qué arquitectura tiene nuestro sistema operativo, utilizamos el comando [hostnamectl](#)*

Accedemos a la página oficial de ngrok, nos registramos y descargamos la versión del S.O. que estemos utilizando, en este caso Linux.

Seguimos la guía que nos da el propio ngrok

Sitio local: C:\Users\DanielT\Downloads\				Sitio remoto: /home/jefe/ftp					
<div><div></div><div>Configuración local</div><div>Contacts</div><div>Cookies</div><div>Datos de programa</div></div>				<div><div>?</div> /</div> <div><div>?</div> home</div> <div><div></div> jefe</div> <div><div></div> ftp</div>					
Nombre de archivo	Tamaño de...	Tipo de archivo	Última modificación	Nombre de archivo	Tamaño d...	Tipo de arc...	Última modific...	Permisos	Propietario/...
mysql-workbench-comm...	44.683.264	Paquete de Windo...	25/10/2021 13:34:38	..					
ngrok-stable-linux-amd6...	13.770.165	Archivo WinRAR	11/01/2022 18:18:05	id_rsa.pub	567	Document...	11/01/2022 9:3...	-rw-r--r--	1000 1000
nmap-7.92-setup.exe	28.644.568	Aplicación	24/10/2021 16:29:29	ngrok-stable-linux-a...	13.770.165	Archivo Wi...	11/01/2022 18:...	-rw-----	1000 1000
1 archivo seleccionado. Tamaño total: 13.770.165 bytes				2 archivos. Tamaño total: 13.770.732 bytes					

1- Descomprimos (y lo llevamos a la ruta donde queramos que acabe el tunel → /var/www/html)

```
jefe@jefe:~$ cd ftp/
jefe@jefe:~/ftp$ ls
id_rsa.pub  ngrok-stable-linux-amd64.tgz
jefe@jefe:~/ftp$ tar zxvf ngrok-stable-linux-amd64.tgz
ngrok
jefe@jefe:~/ftp$ ls
id_rsa.pub  ngrok  ngrok-stable-linux-amd64.tgz
```

2- Conectamos la cuenta

`./ngrok authtoken token_que_indique_la_página` ← cuidado con el ‘muñoneo’ si se copia a mano

3- Lo configuramos

`./ngrok http 80`

```
ngrok by @inconshreveable (Ctrl+C to quit)

Session Status      online
Account             daniel.tamargo@ikastle.egibide.org (Plan: Free)
Version             2.3.40
Region              United States (us)
Web Interface       http://127.0.0.1:4040
Forwarding           http://392b-85-84-177-227.ngrok.io -> http://localhost:80
Forwarding           https://392b-85-84-177-227.ngrok.io -> http://localhost:80

Connections         ttl    opn    rt1    rt5    p50    p90
                   6      0      0.01   0.01   5.71   6.03
```

*A tener en cuenta: la conexión puede fallar un par de veces*

Podemos consultar las conexiones (túneles) establecidas en el panel de ngrok y ahí obtener el enlace que necesitaremos obtener para configurar el hook en github. [El enlace funciona bien.](#)

Region	URL	Client IP	Established
US	<a href="https://392b-85-84-177-227.ngrok.io">https://392b-85-84-177-227.ngrok.io</a>	85.84.177.227	8m ago
US	<a href="http://392b-85-84-177-227.ngrok.io">http://392b-85-84-177-227.ngrok.io</a>	85.84.177.227	8m ago

## Repositorio GitHub:

Creamos un repositorio y preparamos los ficheros básicos, entre ellos el fichero que ejecutaremos con el Hook → automatic-deploy.php

Vaciamos la carpeta html para poder clonar nuestro repositorio ahí:

```
sudo rm -r /var/www/html/*
```

Clonamos el repositorio:

```
sudo git clone https://github.com/DanielTamargo/reto3-despliegue-automatico.git /var/www/html
```

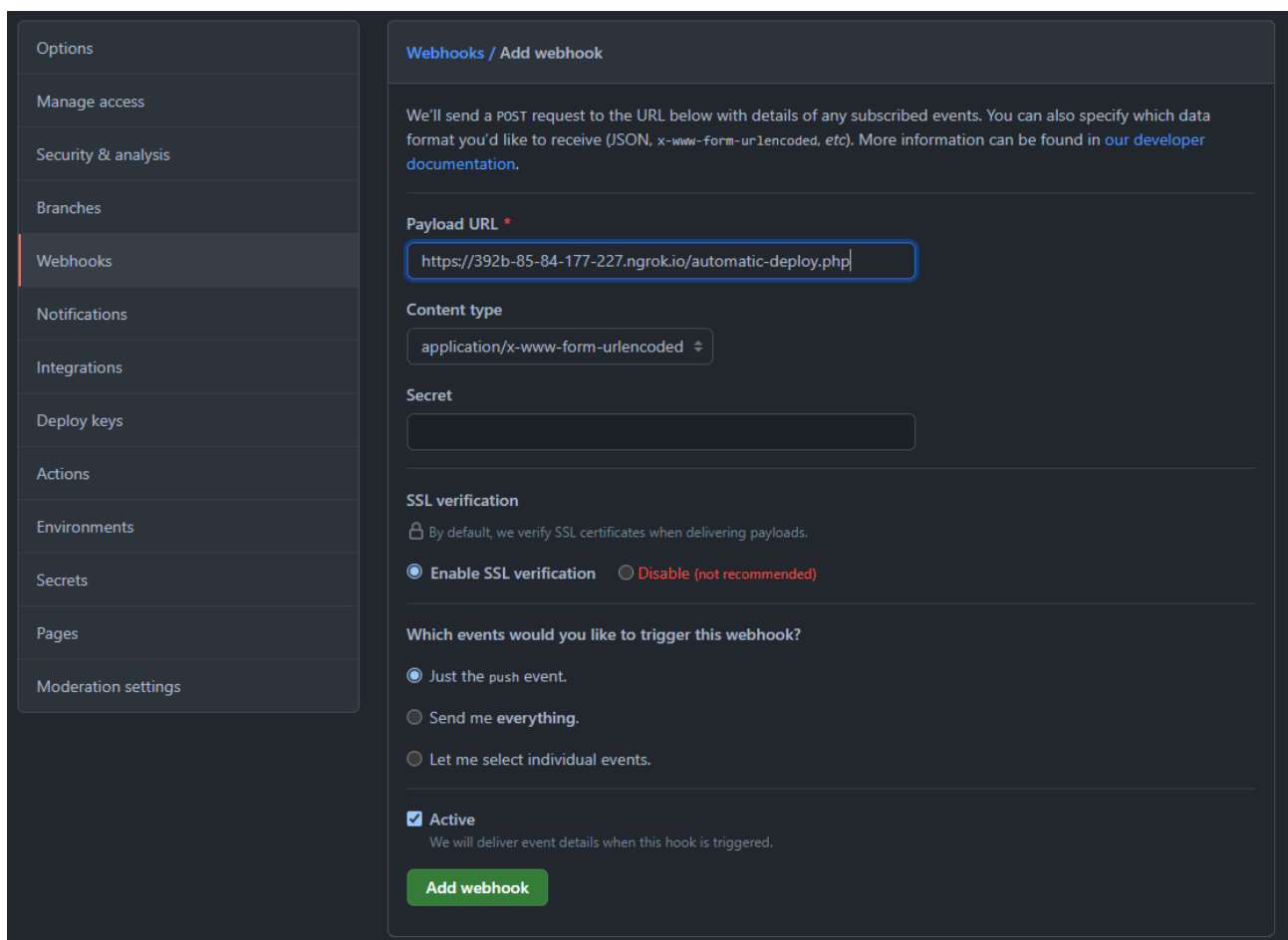
Y modificamos usuario y grupos propietarios:

```
sudo chown -R www-data:www-data /var/www/html
```

Comprobamos que www-data puede hacer comandos tales como git pull:

```
cd /var/www/html && sudo -u www-data git pull ← (no debería dar error)
```

Accedemos al repo en la nube (enlace aquí) donde configuraremos el Hook que hará que a partir de ahora, cada vez que haya un push en el repositorio, vaya a nuestro servidor y ejecute una serie de comandos con los cuales descargaremos esos cambios.



The screenshot shows the GitHub 'Webhooks / Add webhook' configuration page. On the left is a sidebar with navigation links: Options, Manage access, Security & analysis, Branches, Webhooks (highlighted), Notifications, Integrations, Deploy keys, Actions, Environments, Secrets, Pages, and Moderation settings. The main content area is titled 'Webhooks / Add webhook' and contains the following fields and options:

- Payload URL:** A text input field containing 'https://392b-85-84-177-227.ngrok.io/automatic-deploy.php'.
- Content type:** A dropdown menu set to 'application/x-www-form-urlencoded'.
- Secret:** An empty text input field.
- SSL verification:** A section with a lock icon and the text 'By default, we verify SSL certificates when delivering payloads.' Below it are two radio buttons: 'Enable SSL verification' (selected) and 'Disable (not recommended)'.
- Which events would you like to trigger this webhook?:** Three radio buttons: 'Just the push event.' (selected), 'Send me everything.', and 'Let me select individual events.'
- Active:** A checked checkbox with the label 'Active' and a subtext 'We will deliver event details when this hook is triggered.'
- Add webhook:** A green button at the bottom.

Adicionalmente, aseguramos que el usuario y grupo www-data son los propietarios de la carpeta del servidor:

```
sudo chown -R www-data:www-data /var/www/html
```

## Con todo instalado y configurado, probamos su funcionamiento

Primero, realizamos un cambio en alguno de los ficheros, por ejemplo el index.php (puesto que es lo que se ve al cargar la URL directamente).

```
EXPLORER
...
index.php M X
DESPLIEGUEAUTOMATIZA...
  > views
  automatic-deploy.php
  index.html
  index.php M
  README.md
index.php > h1
1 | <h1>Prueba 1 del Hook!</h1>
2 |
3 | <?php
4 |
5 | phpinfo();
```

Después, añadimos esos cambios, los commiteamos y los subimos al repositorio.

```
DanielT@DESKTOP-R7KDAUI MINGW64 ~/Desktop/DespliegueAutomatizado (main)
$ git add index.php

DanielT@DESKTOP-R7KDAUI MINGW64 ~/Desktop/DespliegueAutomatizado (main)
$ git commit -m "Prueba 1 del Hook!"
[main 1164728] Prueba 1 del Hook!
1 file changed, 2 insertions(+)
g
DanielT@DESKTOP-R7KDAUI MINGW64 ~/Desktop/DespliegueAutomatizado (main)
$ git push origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 16 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 311 bytes | 311.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/DanielTamargo/reto3-despliegue-automatgico.git
2a42bdd..1164728 main -> main
```

¡Al hacer push podemos comprobar en el servidor que ngrok está notificando el acceso al fichero automatic-deploy.php!

```
ngrok by @inconshreveable (Ctrl+C to quit)

Session Status      online
Account             daniel.tamargo@ikasle.egibide.org (Plan: Free)
Version             2.3.40
Region              United States (us)
Web Interface       http://127.0.0.1:4040
Forwarding           http://392b-85-84-177-227.ngrok.io -> http://localhost:80
Forwarding           https://392b-85-84-177-227.ngrok.io -> http://localhost:80

Connections
  ttl      opn      rt1      rt5      p50      p90
  10        0        0.00     0.01     5.71     6.62

HTTP Requests
-----
GET /                200 OK
GET /                200 OK
POST /automatic-deploy.php 200 OK
POST /automatic-deploy.php 200 OK
POST /automatic-deploy.php 200 OK
POST /automatic-deploy.php 200 OK
POST /automatic-deploy.php 200 OK
POST /automatic-deploy.php 200 OK
GET /favicon.ico    404 Not Found
GET /                200 OK
GET /                200 OK
```

En esta imagen podemos ver cómo accedí por primera vez al servidor, luego probé el Hook varias veces y volví a acceder para comprobar los cambios.

¡Y en efecto! Nuestro servidor ha ejecutado el fichero automatic-deploy.php por lo que ha llevado a cabo una serie de comandos y ha hecho pull de los últimos cambios, ahora nuestro servidor está actualizado a la última versión.

El servidor antes y después de los cambios:

[Imagen del index.php del servidor \*\*antes\*\* de realizar el push.](#)

[Imagen del index.php del servidor \*\*después\*\* de realizar el push.](#)

[Podemos comprobar que el README.md también ha sido actualizado.](#)