

# Manuales de Instalación de los servidores

Para la instalación del servidor utilizamos Ansible que es un motor open source que automatiza una gran cantidad de procesos informáticos, como la preparación de la infraestructura, la gestión de la configuración, la implementación de las aplicaciones y la organización de los sistemas. También utilizamos Docker para poder correr ansible en el servidor de forma remota sin tener que instalar los paquetes dentro del mismo servidor, esto nos facilitó muchas cosas ya que Docker es la tecnología de organización en contenedores que posibilita la creación y el uso de los contenedores de Linux. En este caso nosotros creamos un servidor que corre en centos-7 minimal 2009, además de estas aplicaciones muy útiles instalamos Git en el servidor el cual nos ayuda a descargar el código de nuestro programa el cual después será levantado por el comando “Docker-Compose up” el cual es un comando que viene instalado con Docker que se utiliza para levantar apps dentro del servidor o máquina virtual que tu te estes direccionado. Para correr el comando antes mencionado tenemos que tener creado un archivo el cual le llamamos compose.yml, este archivo nos facilitó para poder correr las aplicaciones y tenerlas preconfiguradas en el servidor, así ahorrándonos tiempo de una gran manera, estas aplicaciones eran: Backoffice, API en el cual se encuentra nuestra API, el Frontend y por último también se instala Redis que esta aplicación fue utilizada para el tema de los mapas, coordenadas y distancias de los puntos marcados.

Pasaremos a explicar cada Playbook de ansible que utilizamos para que el servidor pudiera estar en funcionamiento y luego explicaremos un poco sobre docker y que contenedor utilizamos ya que fue parte fundamental del proceso de la creación del servidor.

- En primer lugar pero no menos importante tenemos el Playbook [LlamarTareas.yml](#) el cual solamente llama las tareas que están dentro de la carpeta Tareas. Esto se suele hacer para tener un mayor orden en el servidor y aparte no tener que ir ejecutando uno por uno los Playbooks
- El primer Playbook llamado es [FixRepos.yml](#) el cual es un Playbook esencial para el servidor ya que centos-7 al ser tan antiguo se quedó sin soporte para sus repositorios y necesita ser actualizado/arreglado para que se puedan instalar las otras aplicaciones más tarde
- El segundo Playbook llamado es [Firewall.yml](#). este Playbook es simple, lo que hace es desactivar SELinux el cual nos genera conflictos a nivel de las entradas de ssh y demás temas.
- El tercer Playbook llamado es [Paquetes.yml](#), simple solamente instala paquetes útiles como lo son Git y Vim.
- En cuarto lugar se llama a [CrearUsuarios.yml](#), este se corre para la creación de 8 usuarios los cuales son:
  - El usuario thiago con su respectiva contraseña la cual es una contraseña predeterminada que se basa en su nombre y su número de lista (esta contraseña es hasheada para que linux la reconozca ya que no la reconoce si no es hasheada de buena manera) y también tiene su respectiva clave pública agregada en el archivo para que cuando intenten entrar al usuario de thiago solamente lo pueda hacer el validando con su clave privada, cabe recalcar que este usuario como también lo serán los otros están en el group Wheel el cual en centos es el grupo de los administradores.
  - El usuario juanP que se refiere al usuario de Juan Pablo (Linux tiene complicaciones de vez en cuando con los espacios) el cual también tiene su

respectiva contraseña y clave pública, el cual también pertenece al group Wheel.

- El usuario lucas, de igual manera que los otros tambien tiene su contraseña y clave pública, el cual también pertenece al group Wheel.
  - El usuario martin, de igual manera que los otros tambien tiene su contraseña y clave pública, el cual también pertenece al group Wheel.
  - El usuario rodrigo, de igual manera que los otros tambien tiene su contraseña y clave pública, el cual también pertenece al group Wheel.
  - El usuario fedeG que se refiere al usuario de Federico Guida(Hay dos participantes de este proyecto con el nombre Federico entonces decidimos utilizar la primera letra como distintivo), de igual manera que los otros tambien tiene su contraseña y clave pública, el cual también pertenece al group Wheel.
  - El usuario fedeM que se refiere al usuario de Federico Martinez(Por la misma razón que el usuario federicoG), de igual manera que los otros tambien tiene su contraseña y clave pública, el cual también pertenece al group Wheel.
  - Por ultimo esta le usuario gonza, este usuario es utilizado por nuestro profesor Gonzalo Martinez para poder tener tambien el control del servidor sin tener que utilizar el usuario de centos que ya viene creado de manera predeterminada
- El quinto archivo llamado es [ssh.yml](#), este es un playbook que cambia el archivo sshd\_config que está ubicado en /etc/ssh. Esto lo que nos hace es que sea mucho más seguro nuestro servidor, la primera tarea que hace es cambiar el puerto 22 (predeterminado) por el puerto que nosotros elijamos (en este caso el puerto 6969), la segunda tarea que hace es Desactivar el Root por ssh, dándonos como resultado que nadie pueda entrar por ssh al usuario root, como tercera tarea se desactiva la entrada por password por ssh así solamente se puede entrar teniendo la Clave pública, cuarta tarea es Restringir los usuarios de ssh a los usuarios antes creados los cuales son lucas martin thiago rodrigo federicoG juanP federicoM gonza y por último se reinicia el ssh para que se actualice el archivo que hemos cambiado recientemente.
  - En sexto lugar llamaremos al playbook [Docker.yml](#) el cual agregara los repositorios de Docker-CE (El repositorio de Docker en Centos), su segunda tarea es instalar los paquetes que seleccionamos los cuales son, docker-ce, docker-ce-cli, containerd.io, docker-buildx-plugin y por ultimo docker-compose-plugin, como ultima tarea lo inicializa y los deja para poder utilizar.
  - Como último lugar, tal vez el playbook más importante para nuestro programa que es [Aplicaciones.yml](#), este playbook tiene 5 tareas:
    - Create /opt/aplicaciones directory, esto lo que hace es crear un repositorio con los permisos de leer, escribir y ejecutar, aqui seran guardado el codigo de GitHub.
    - Clonar repositorio de Frontend como segunda tarea, como dice el nombre clona el repositorio Frontend de la branch más actualizada.
    - Clonar repositorio de API como tercera tarea, como dice el nombre clona el repositorio API de la branch más actualizada.
    - Clonar repositorio de Backoffice como cuarta tarea, como dice el nombre clona el repositorio Backoffice de la branch más actualizada.

- Copiar archivo docker compose como última tarea, como dice el nombre clona el archivo compose.yml el cual será utilizado para poder levantar nuestros servicios.

Ahora hablemos de los comandos que tendremos que utilizar tanto de docker como de ansible y como del Compose. Hay 4 comando esenciales que nos facilitaron las cosas en este proceso:

- `docker run -ti --rm -v %cd%:/ansible ggmartinez/ansible:9.1.0` este comando inicializa una imagen de docker que tiene instalado ansible y otras aplicaciones para poder correr, con esto pondremos `cd /ansible` y entraremos a la carpeta ansible del contenedor lo cual tendremos que poner el segundo comando más importante
- `ansible-playbook -i inventario.ini LlamarTareas.yml -k -K`, explicare este comando por partes pero de manera breve:
  - “ansible-playbook” es para que reconozca que sera utilizado un playbook de ansible.
  - “-i inventario.ini” el menos -i es para que reconozca la carpeta que estes parado de tu dispositivo ejemplo tu usuario, “inventario.ini” es lo que hace que ansible se de “cuenta” de a qué máquina/servidor tiene que “apuntar”.
  - “LlamarTareas.yml” es el playbook que queremos que se ejecute.
  - “-k -K” son las claves de ssh del usuario que pusimos que utilizaremos en el inventario para poner a funcionar todo.
- “docker-compose up” el cual es el que ejecute el archivo compose.yml, dandonos como resultado que se levante el servidor y quede completamente funcional con los repositorios ya clonados y funcionales.