24-1-2021

Actividad 1 Despliegue aplicaciones web

Comandos

Patricia Villanueva Polo

Índice

[I. ¿Cómo sabemos si tenemos conexión a internet? Pista: ifconfig, ping 2](#_Toc62328798)

[II. ¿Cómo sabemos si nuestro servidor es accesible desde Internet? 4](#_Toc62328799)

[III. ¿Cómo sabemos a quién pertenece una dirección web (URL)? 7](#_Toc62328800)

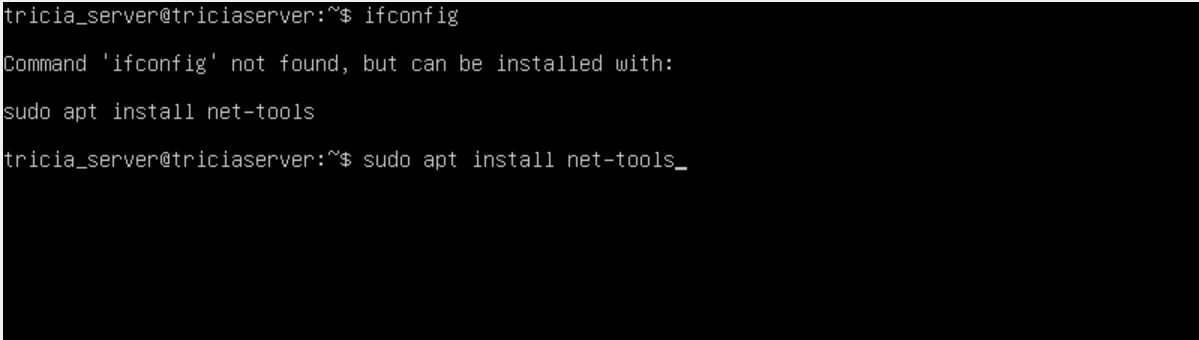
[IV. ¿Cómo probamos que podemos acceder a un servidor? 11](#_Toc62328801)

[V. ANEXO I: Instalación de Máquina Virtual VirtualBox 13](#_Toc62328802)

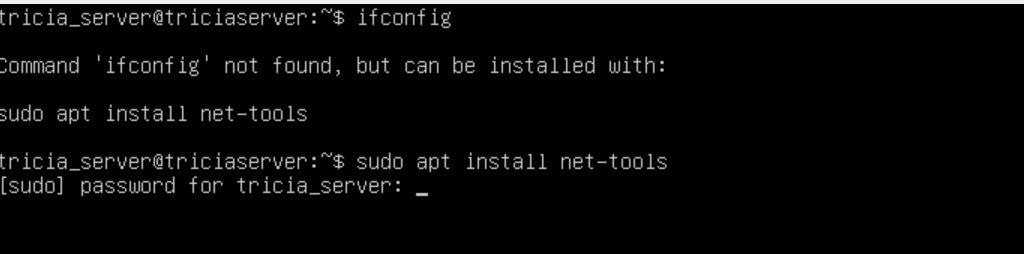
[VI. ANEXO II: Instalación de Ubuntu Server en VirtualBox 16](#_Toc62328803)

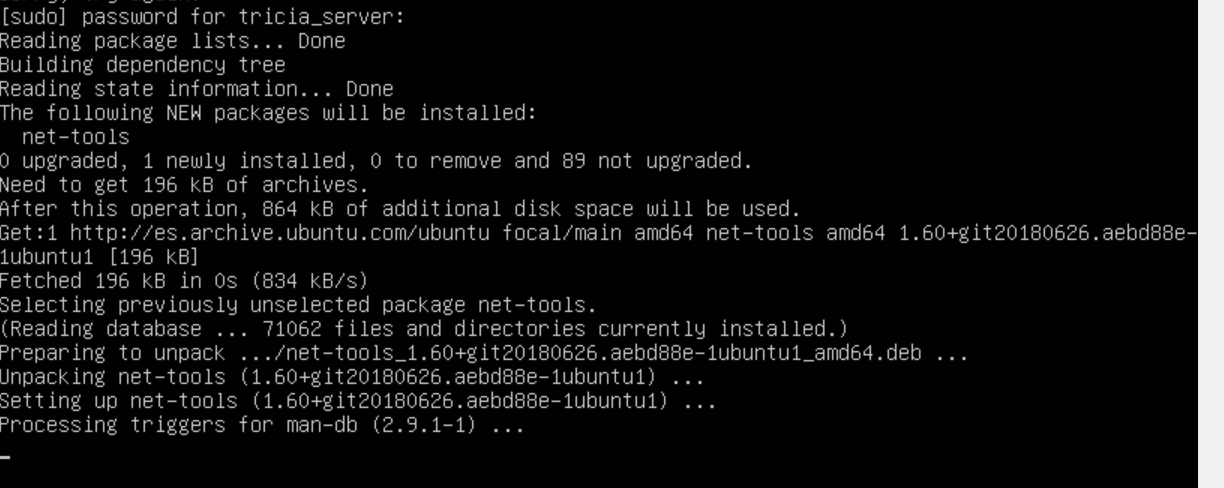
# ¿Cómo sabemos si tenemos conexión a internet? Pista: ifconfig, ping

La primera vez que tratamos de utilizar alguno de los comandos desde el terminal del servidor, podemos encontrarnos con un mensaje como este:



Lo bueno, es que nos da una pista de cómo podemos solucionarlo. En este caso solo necesitamos instalar el paquete de red, net-tools, para lo cual podemos usar el comando que nos indica el propio Ubuntu: sudo apt install net-tools.





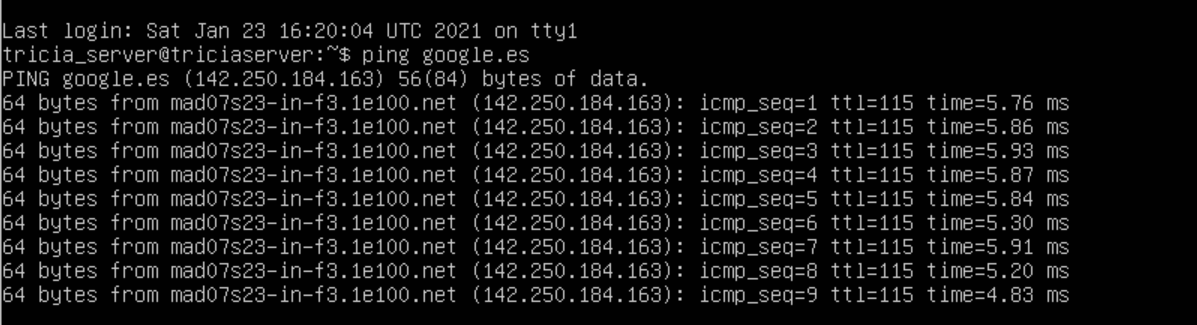
**Sudo** lo utilizamos para ejecutar comandos en modo administrador, en algunas ocasiones se nos solicitará hacerlo así.

Ahora vamos a lo que nos interesa. Una vez tenemos instalado nuestro servidor, una de las primeras cosas que vamos a querer comprobar es que tenga conexión a internet. Para ello podemos utilizar uno de los comandos más habituales:

**PING:** Este comando manda una solicitud a un servidor dado y, sí todo funciona correctamente, recibe una respuesta de dicho servidor. A raíz de esa respuesta, se calcula el tiempo mínimo, medio y máximo de la misma. Si no recibe ninguna respuesta nos llegará un mensaje de error, en función del problema detectado.

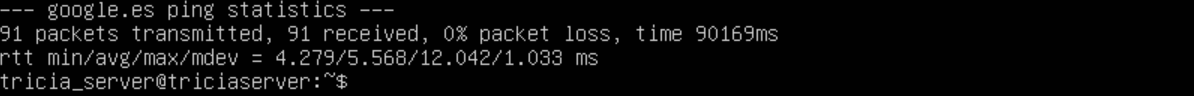
Para usar este comando tecleamos “ping” seguido de una ip o dirección web que conozcamos. Por ejemplo: ping google.es.

Si nuestro servidor Ubuntu tiene conexión con internet, podrá enviar la solicitud a Google.es y, a cambio, Google le devolverá una respuesta que permitirá calcular ese tiempo mínimo, medio y máximo.



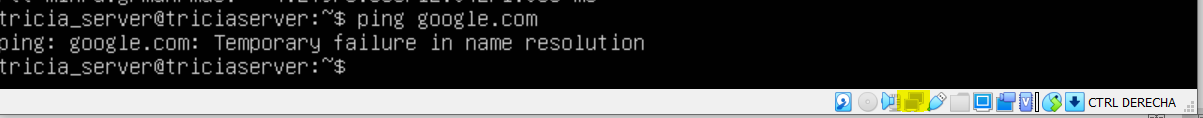
Al ejecutar el comando vemos como va realizando una llamada por cada línea que aparece en la consola. Cada una de ellas muestra el tamaño del paquete que se ha enviado en la solicitud (64 bytes), desde dónde recibimos la respuesta y su ip. La secuencia de cada paquete enviado es el icmp\_seq. El ttl Es el tiempo de vida, en cada salto que da hasta llegar al host descuenta uno. Eso se hace para que el paquete no esté recorriendo la red infinitamente y se puedan descartar llegado el momento. Y time, son los ms que tarda en enviarse la solicitud y recibir la respuesta.

Para detener el flujo del comando, debemos pulsar **Ctrl+c.** Al hacerlo nos indica el resultado final de las llamadas realizadas:



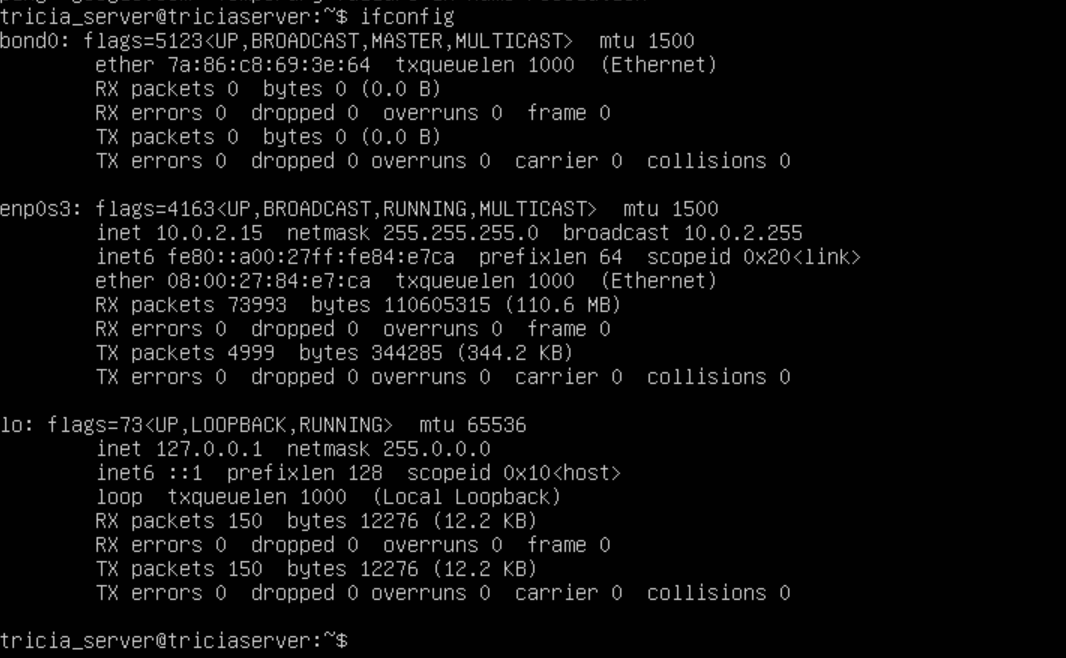
Ha transmitido un total de 91 paquetes al servidor, de los cuales, el servidor nos ha devuelto 91, por tanto, no se ha perdido ninguno. Ha estado corriendo un total de 90169 milisegundos. El tiempo mínimo es de 4,279ms; el medio es de 5,568ms, el máximo que ha tardado ha sido 12,042ms y el mdev indica la media de cómo de lejos se encuentra cada ping del tiempo medio.

Si no tuviéramos conexión a internet, por ejemplo, desconectando la conexión de la máquina virtual, recibiríamos algo similar a esto:



**IFCONFIG**: Para hablar de este comando necesitamos saber qué es una interfaz de red. Una interfaz de red es el software capaz de comunicarse con el controlador y la capa IP para permitir la conexión entre distintos dispositivos y redes presentes.

Dicho esto, el comando ifconfig, nos permite configurar interfaces de red en nuestro servidor. Si lo utilizamos sin ningún parámetro, nos muestra el estado actual de todas nuestras interfaces:



En nuestro ejemplo, tenemos 3 interfaces.

Lo primero que se indica es el estado del dispositivo:

UP: Indica que el dispositivo está activo.

BROADCAST: Puede enviar tráfico a otro dispositivo cruzando subredes

RUNNING: Indica que la interfaz está disponible y funcionando.

MULITCAST: Indica que permite enviar datos a múltiples destinos simultáneamente

LOOPBACK: Hace referencia a la red virtual de nuestra VirtualBox que la conecta con la red de nuestro pc.

MASTER: Indica que es la red maestra.

Si nos fijamos solamente en estos valores, ya sabemos que tenemos una red, la enp0s3 que está funcionando y que es capaz de conectarse a internet. Por lo que este comando también nos valdría para averiguar si tenemos conexión con el exterior.

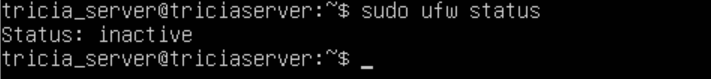
Este comando puede utilizarse con parámetros para configurar las interfaces que tenemos, pero no nos extenderemos en dicha explicación al no formar parte de este trabajo.

# ¿Cómo sabemos si nuestro servidor es accesible desde Internet?

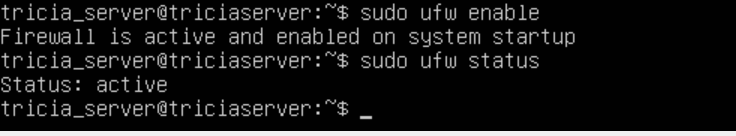
Ahora que ya tenemos claro que nuestro servidor se puede conectar a internet, toca preguntarse si internet se puede conectar a nuestro servidor. Para ello podemos recurrir a otros dos comandos.

**UFW**: Para conectarme a un servidor necesito que dicho servidor tenga una IP y un puerto público al que poder conectarme. Sin embargo los servidores suelen tener firewalls para evitar que cualquiera con intenciones maliciosas pueda acceder al mismo, por lo que también necesitamos que dicho firewall tenga cierta configuración que permita cierto acceso a la ip y puerto público.

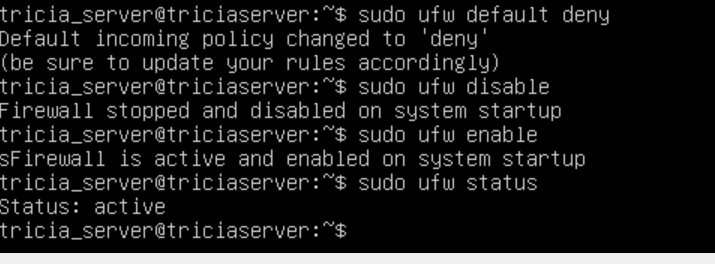
UWF es un script que permite configurar el cortafuegos del kernel de Linux (Ubuntu en nuestro caso). Lo primero que hacemos es comprobar si uwf está habilitado, ya que, en algunos Linux viene deshabilitado por defecto:



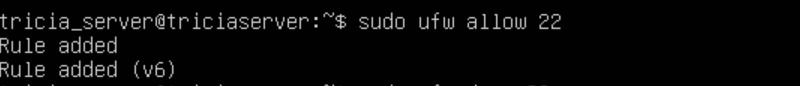
A continuación, lo vamos a activar, habilitándolo:



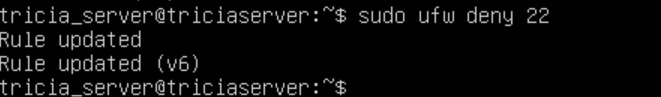
Lo siguiente sería asegurarnos de que el cortafuegos, por defecto, deniegue todas las posibles conexiones al servidor para que, más adelante, configuremos las excepciones. Así lo tendremos protegido y daremos acceso solo a aquello que deba ser público. Para eso desactivamos todas las conexiones y reiniciamos el ufw para que coja el cambio:



Ahora nadie podría acceder a nuestro servidor desde Internet. Para asegurarnos que podamos mantener una conexión desde el exterior a nuestro servidor, necesitamos abrir los puertos necesarios para dicha conexión. En función de si vamos a establecer una web, un sistema de compartición de archivos …etc, necesitaremos unos puertos u otros. Por ejemplo, para abrir el puerto 22:

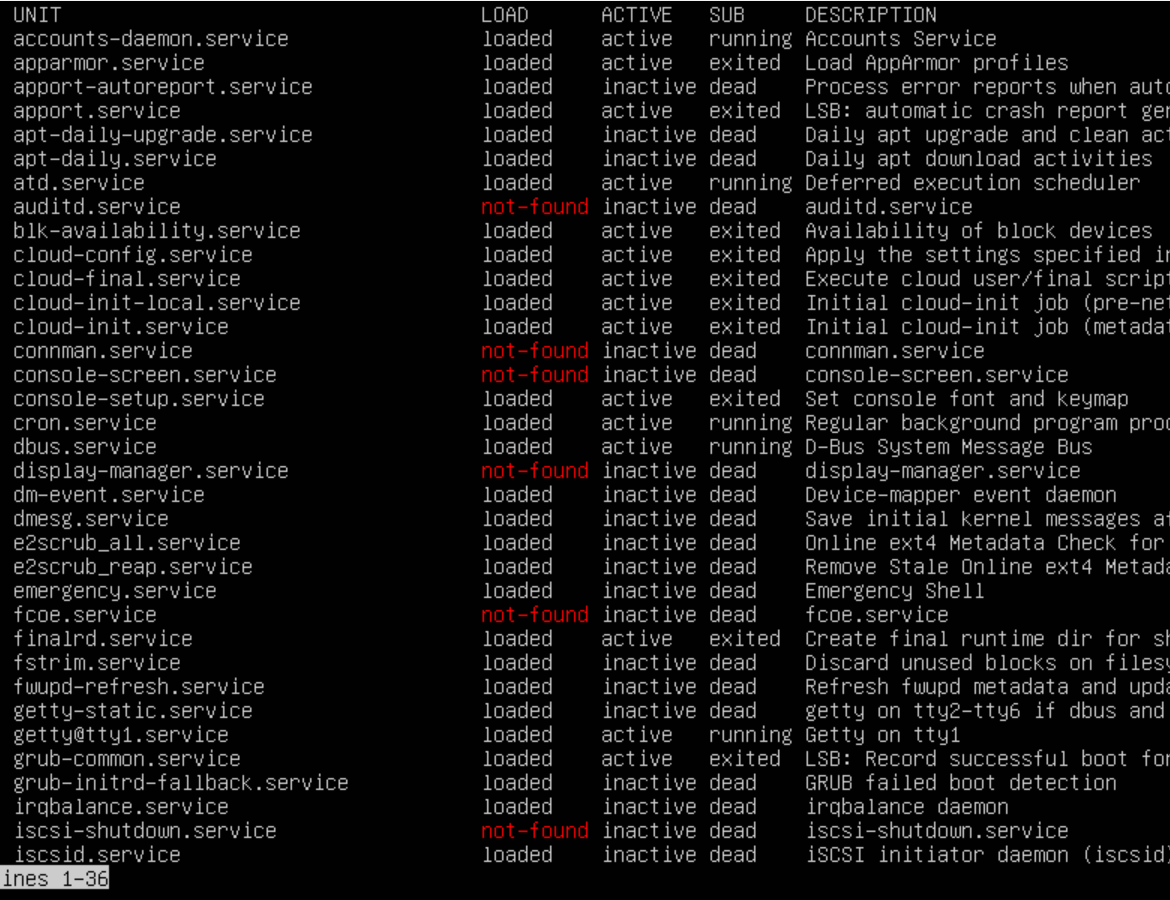


Y volver a denegarlo, cosa que podamos hacer en cualquier momento:

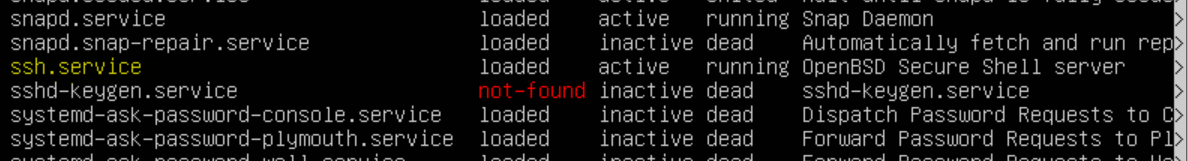


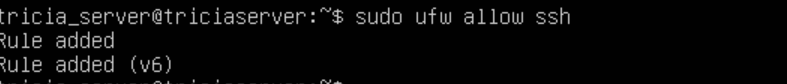
Ahora nuestro servidor será accesible desde los puertos que nosotros queramos que lo sea en función del servicio que vayamos a dar desde él.

En lugar de habilitar puertos, el comando también nos permite habilitar servicios, y para ver cuáles tenemos disponibles podemos utilizar: **sudo systemctl -r –type service –all** y veremos todos los que tenemos:



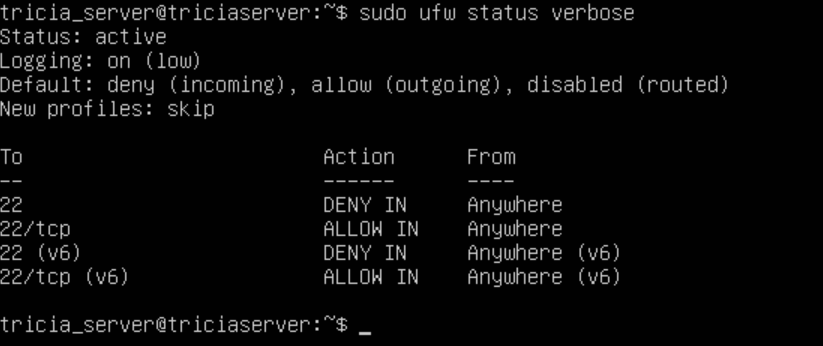
Por ejemplo, podemos activar el servicio ssh:





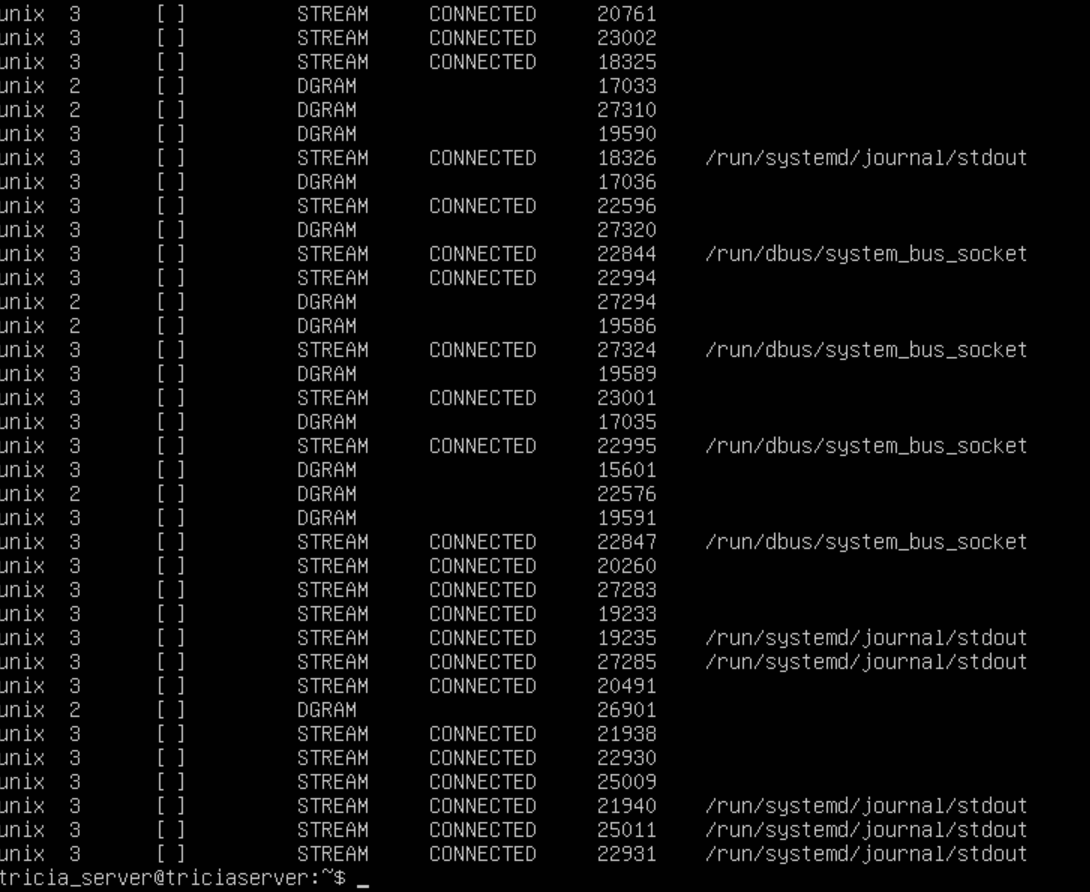
Pero, y sí el servidor no lo hemos instalado nosotros, ¿cómo sabemos el estado de ufw y si tiene puertos activos o no?

Para ver el estado y la política por defecto:

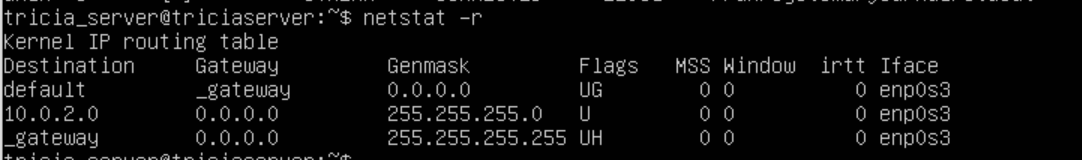


Nos muestra el Status, que para nuestro ejemplo está activo. Para saber qué política aplica nos fijamos dónde pone “Default” en nuestro caso se impiden las conexiones de entrada, están permitidas las de salida y el enrutador está desactivado. Además, podemos ver que desde el puerto 22/tcp y 22/tcp (v6) se permite la entrada al servidor.

**NETSTAT**: es una herramienta de línea de comandos que muestra las conexiones de red (entrantes y salientes), tablas de enrutamiento y una serie de estadísticas de interfaz de red. Para ver todos los conectores usaríamos netstat.



Si solo nos interesan los enrutadores: netstat -r

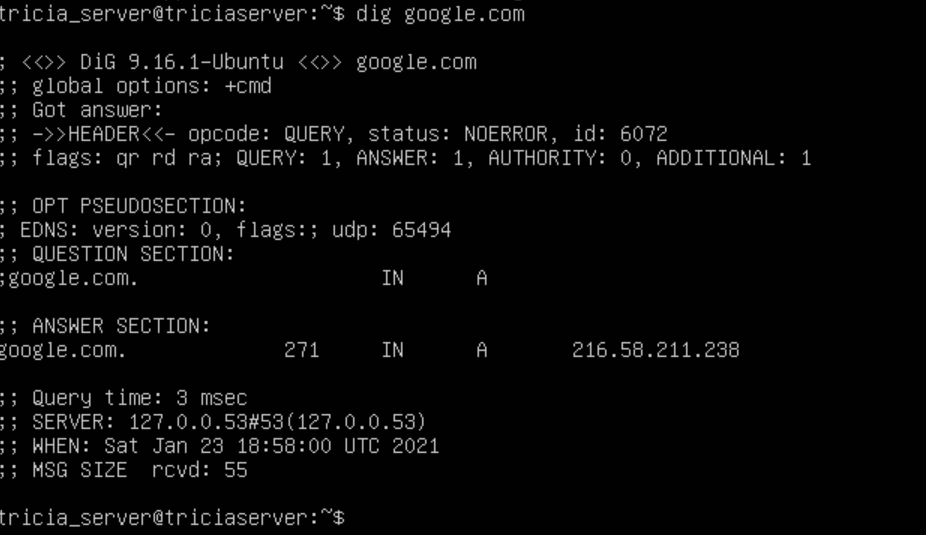


Podemos ver como para algunos de los conectores indica CONNECTED y para otros no pone nada, esto es porque los primeros tienen la conexión habilitada y el resto no.

# ¿Cómo sabemos a quién pertenece una dirección web (URL)?

**DIG**:  Es una herramienta para realizar consultas a los servidores DNS para solicitar información sobre direcciones de host, intercambiadores de correo, servidores de nombres e información relacionada. Pero, qué es una DNS, es la tecnología que nos permite traducir los nombres de dominio a direcciones ip y viceversa.

Por ejemplo, si tecleamos dig + servidor:



Primero nos indica la versión y sus opciones globales, en este caso es DIG 9.16.1-Ubuntu y las opciones son cmd.

A continuación, nos da información sobre la respuesta del servidor DNS al que hemos enviado la solicitud (Header)

Question nos indica que solicitud hemos realizado.

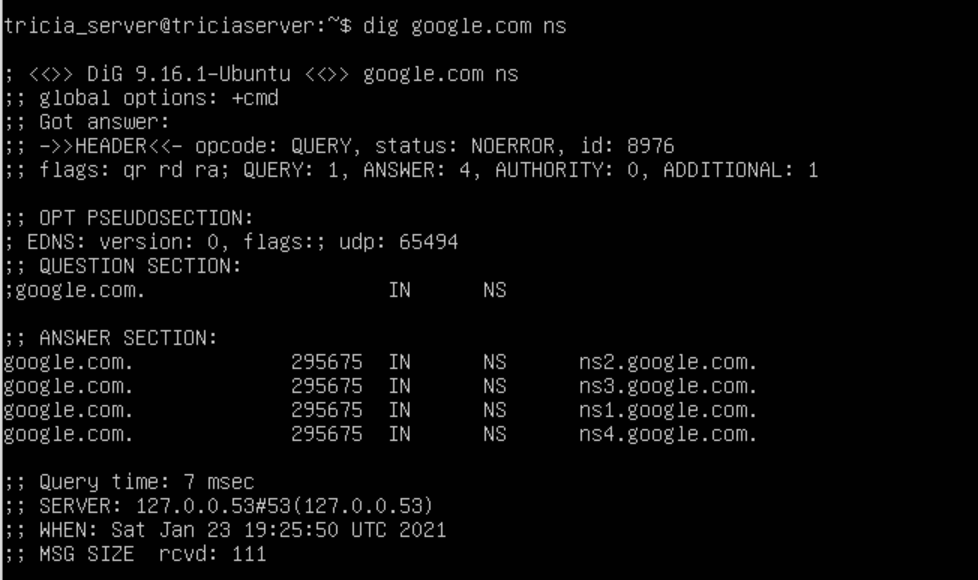
Answer nos indica **la respuesta de la consulta** que es la ip del DNS correspondiente al dominio de Google.com: 216.58.211.238

Y por último tenemos algunas estadísticas de dicha respuesta, el tiempo que ha tardado en realizarse, cuando ha sido, a qué servidor…

Con **dig dominio a**, puedo averiguar la ip del dominio:



Dig dominio ns, me da el nombre del servidor dns que permite resolver la zona del dominio:



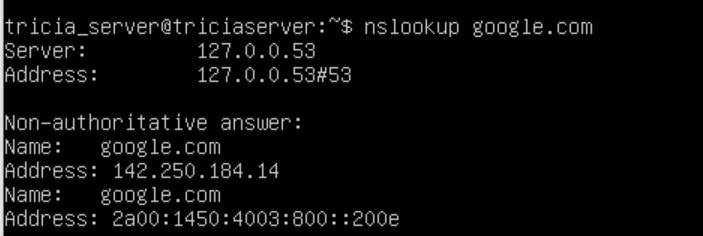
Dig dominio soa: información de zona autoritativa y administración de la zona en el servidor DNS.

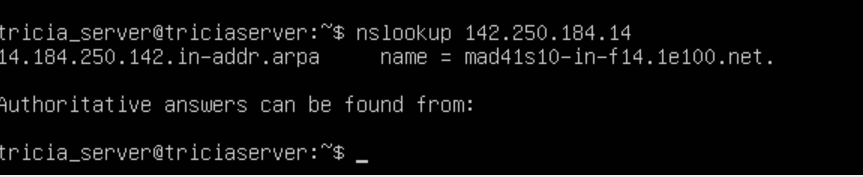


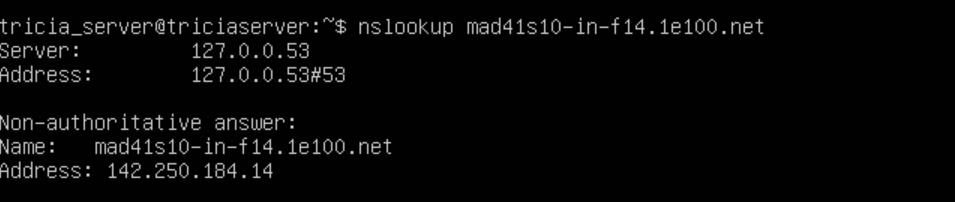
Dig dominio mx: información de los intercambiadores de correo del dominio:



**NSLOOKUP**: Este comando también nos permite averiguar información sobre las DNS de un dominio. Por ejemplo, la ip del dominio, o el dominio de una ip específica:







Al igual que con Dig, también podemos consultar el intercambiador de correo, con mx, el soa y otros similares. Obteniendo así el origen de una ip o dominio.

# ¿Cómo probamos que podemos acceder a un servidor?

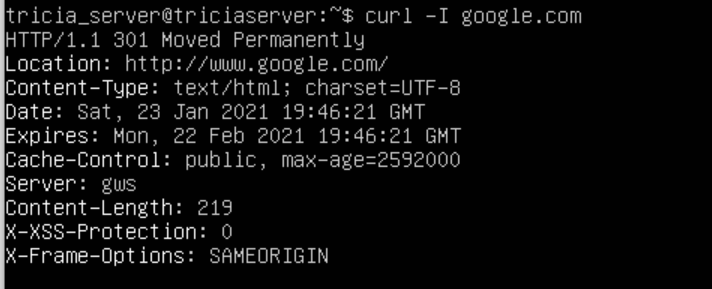
**CURL**: Este comando sirve para verificar la conectividad a las URL y como una gran herramienta para transferir datos. Por tanto, nos va a servir para comprobar que tenemos acceso a un servidor. Además, funciona con los protocoles más comunes:

* HTTP y HTTPS
* FTP y FTPS
* IMAP e IMAPS
* POP3 y POP3S
* SFTP
* TELNET
* SMTP y SMTPS

Por ejemplo, simplemente escribiendo curl dominio, el comando nos devolverá la web en formato html y, sí es capaz de devolvernos esta información, comprobamos que hemos podido acceder al dominio o url deseado de ese servidor:



También podemos obtener la cabecera que se envía en una solicitud http:



Se pueden hacer solicitudes get, como hemos visto en el primer caso y solicitudes post, acceder por ftp, telnet, pop3 y muchos más.

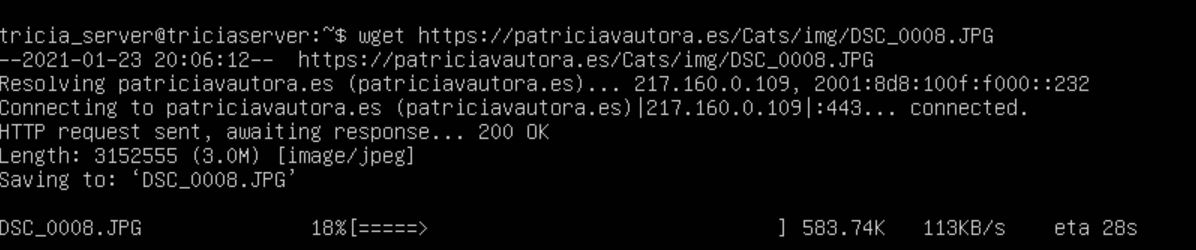
Si usamos este comando con una ip, dominio o url de nuestro servidor y nos devuelve la web, el archivo o lo que estemos solicitando, indicará que teníamos acceso a dicho servidor.

**WGET**: Esta opción nos permite recuperar contenido y archivos que estén en cualquier servidor, por tanto, si al emplearla conseguimos descargar lo que deseamos, será un buen indicativo de que tenemos acceso al mismo. En cambio si recibimos un error puede ser que el contenido no esté o que no haya conexión al servidor.

Wget por sí solo no nos devuelve nada, siempre debe ir acompañado al menos de una url. Si tecleamos wget Google.com, podemos ver que nos devuelve la ip del dominio y nos indica si ha podido conectarse. Por último vemos que nos ha devuelto la web en html y la ha cargado por defecto en un fichero llamado **index.html**.



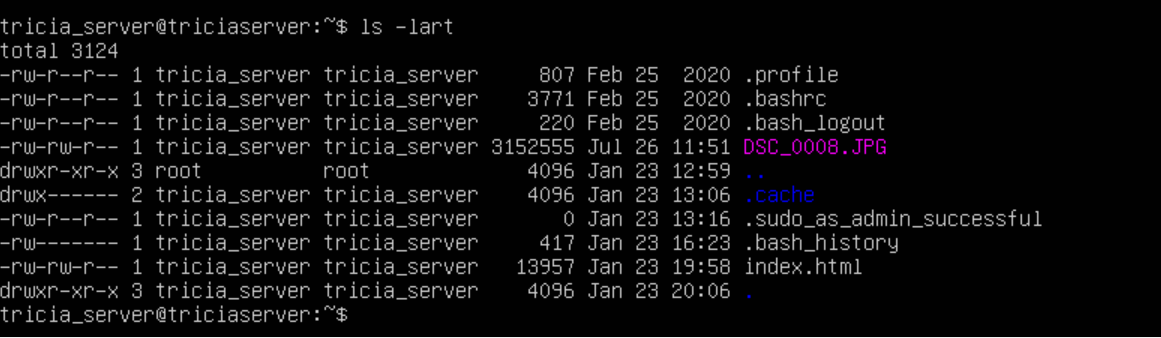
También nos permite descargar archivos, por ejemplo, si quiero descargar una foto que tengo alojada en mi dominio de internet, solo tengo que usar wget + url dónde se encuentra la foto. Inicia la descarga:



Y al terminar de descargar todo me indica que ha guardado el fichero DSC\_0008.JPG



Para comprobar que el archivo está correctamente descargado de la url a mi máquina puedo comprobarlo con un ls -lart o un simple ls:



En la captura vemos tanto index.html de la primera ejecución de wget, como DSC\_0008.JPG de mi última ejecución.