
Servicios en red Documentation

Versión 2016.0

José Domingo Muñoz

18 de septiembre de 2017

1. Introducción a los servicios en red	3
1.1. Cuestionario de repaso	3
1.2. Python para sysadmins	5
1.3. Introducción a vagrant	6
2. Servidor DHCP	11
2.1. Enlaces interesantes	12
2.2. ¿Cómo funciona el servidor DHCP	12
2.3. Cuestionario DHCP	15
2.4. Ejercicio: Instalación y configuración del servidor dhcp en linux	16
3. Servidor WEB	21
3.1. Enlaces interesantes	21
3.2. Ejercicio: Hacer peticiones HTTP: GET, HEAD y POST	22
3.3. Ejercicio: Instalación y configuración básica de Apache	23
3.4. Ejercicio: VirtualHosting con Apache	24
3.5. Ejercicio: Mapear URL a ubicaciones de un sistema de ficheros	26
3.6. Ejercicio: Control de acceso, autenticación y autorización	27
3.7. Ejercicio: Configuración de apache mediante archivo .htaccess	30
3.8. Ejercicio: Módulos en apache	31
4. Servidor DNS	37
4.1. Enlaces interesantes	37
4.2. Ejercicio: Consultas DNS	38
4.3. Ejercicio: DNSmasq como DNS cache/forward en una red local	39
4.4. Ejercicio: Instalación y configuración del servidor bind9 en nuestra red local	40
4.5. Ejercicio: Instalación y configuración de un servidor DNS esclavo	41
4.6. Ejercicio: Configuración de subdominios virtuales con bind9	41
4.7. Ejercicio: Delegación de subdominios con bind9	42
4.8. Ejercicio: Instalación y configuración de un servidor DNS dinámico	42
5. Rendimiento en Servidores Web	45
5.1. Enlaces interesantes	45
5.2. Ejercicio: Módulos de Multiprocesamiento (MPMs)	46
5.3. El comando ab	47
5.4. Ejecución de scripts php	48
5.5. Ejecución de aplicaciones web python en apache2	49

6. Servidor FTP	53
6.1. Enlaces interesantes	53
6.2. Ejercicio: Instalación de proFTPD y uso de clientes FTP	54
7. Servidor de correo electrónico	57
7.1. Enlaces interesantes	57
7.2. Instalación y configuración básica de postfix	57
7.3. Gestión de correo desde el servidor	58
7.4. Caso 1: Envío local, entre usuarios del mismo servidor	58
7.5. Caso 2: Envío de correos desde internet a usuarios del servidor	60
7.6. Alias y redirecciones	61
7.7. Caso 3: Envío de correo desde usuarios del servidor a correos de internet	62
7.8. Soluciones al problema del spam	63
7.9. Caso 4: Envío de correo electrónico usando nuestro servidor de correos	64
7.10. Caso 5: Recepción de correo electrónico usando nuestro servidor de correos	68
7.11. Antispam y antivirus en nuestro servidor de correos	69
8. Servidor proxy/cache Squid	73
8.1. Enlaces interesantes	73
8.2. Ejercicio: Instalación y configuración básica de squid	73
8.3. Gestionando la cache de Squid (parte 1)	74
8.4. Ejercicio: Controlando la cache de squid	74
8.5. Gestionando la cache de Squid (parte 2)	76
8.6. Gestionando el proxy de Squid	76
8.7. Ejercicio: Configuración de los esquemas de control de acceso	77
8.8. Ejercicio: Autenticación básica por usuarios y grupos con Squid	78
8.9. Ejercicio: Configuración de parámetros de proxy en clientes web	78
8.10. Otras herramientas: Dansguardians y Sarg	79
9. Balanceador de carga haproxy	81
9.1. Enlaces interesantes	81
10. Prácticas	83
10.1. Práctica: Servidor DHCP	83
10.2. Práctica: Servidor WEB	85
10.3. Práctica: Estudio de servidores Webs	89
10.4. Práctica: Servidor DNS	90
10.5. Práctica: Configuración de un servidor Windows Server	93
10.6. Práctica: Configuración de servidores GNU/Linux	95
10.7. Práctica: Mediciones de rendimiento de Apache2 sirviendo páginas dinámicas	98
10.8. Práctica: Gestionar un hosting por ftp	99
10.9. Práctica: Implantación de un servidor de hosting	101
10.10. Práctica: Servidor de correos	103
10.11. Práctica: Servidor proxy/cache squid	105
10.12. Práctica: Balanceo de carga en servidores Apache con HAproxy	107

El módulo profesional de **Servicios de Red e Internet** se imparte durante el segundo curso del **Ciclo Formativo de Grado Superior de Administración de Sistemas Informáticos en Red (ASIR)**.

De acuerdo a la normativa reguladora del ciclo formativo, el módulo profesional de Servicios en red se imparte durante el segundo curso y tiene asignadas un total de 126 horas, a razón de 6 horas semanales durante 21 semanas.

El índice de contenidos que vamos a estudiar será:

Introducción a los servicios en red

En este primer apartado vamos a estudiar los siguientes apartados:

- ¿Qué son los servicios de red?
- Vamos a recordar conceptos sobre redes, TCP/IP, DNS,...
- Python como lenguaje para el administrador de SI
- Herramienta ([vagrant](#)) que nos va a permitir trabajar con nuestra infraestructura.

Cuestionario de repaso

Repaso TCP/IP

1. Teniendo en cuenta el siguiente esquema de red:



- Configura la interfaz de red de un cliente para tener acceso a internet, utiliza direccionamiento estático. ¿Qué diferencia hay entre dirección estática y dinámica? ¿Qué dirección del router es la pública? ¿Cuál es la privada? Define cada uno de los parámetros que has configurado: puerta de enlace, máscara de red,...

2. Cambia el direccionamiento de red de nuestra internet con la red 172.22.0.0/16 ¿Cuántos equipos podemos tener en esta red?
3. Nuestros clientes pueden acceder a internet porque el router hace Source NAT ¿Explica en qué consiste? Si tenemos un servidor linux haciendo de router, ¿cómo se configura para hacer SNAT?. Del mismo modo si tenemos un servidor Windows, ¿cómo se configura?
4. ¿Qué puerto se utiliza por defecto para conectarse a un servidor web? ¿Y para el servidor DNS? ¿Para qué se usa el puerto 22?
5. Imagina que en nuestra intranet instalamos un servidor web. ¿Qué configuración hay que hacer en el router para poder acceder desde internet al servidor? ¿Cómo se llama esta técnica?
6. Si nombramos las máquinas de nuestra intranet, y tenemos un sistema linux, ¿en qué fichero se configura el nombre?
7. ¿Qué es la resolución estática de nombres? ¿En qué fichero se configura en Windows? ¿Y en linux?
8. Si nuestro cliente tiene un sistema linux, ¿en qué fichero hemos configurado la red? ¿y los servidores DNS?
9. Muestra la configuración de red de tu ordenador de clase. ¿Qué servidor DNS se está utilizando.

Repaso DNS

10. ¿Qué herramientas se usa en linux para realizar una consulta DNS? ¿Y en Windows? Preguntar en varios sistemas cuál es la dirección IP de www.marca.com. Realiza una consulta para saber los servidores DNS que conocen el dominio gonzalonazareno.org.
11. ¿Qué ocurre si hacemos una consulta para averiguar la ip de dit.gonzalonazareno.org desde el ordenador del aula y desde el ordenador de tu casa? Razona la respuesta.
12. A qué servidor DNS le estás consultando desde clase. Realiza una consulta a www.google.es consultando a nuestro DNS. Vuelve a hacer la consulta usando el servidor público que ofrece google.
13. ¿Qué información puedo guardar en una zona DNS? ¿Qué registros puedo guardar? ¿Cuántos tipos de zonas existen?
14. Si desde clase, consulto la dirección IP de www.josedomingo.org? ¿Cuál es el proceso de consultas que se realiza? ¿A qué servidores se va preguntando?
15. ¿Qué son los root server? ¿Cuántos hay?
16. Haz una consulta a www.nyu.edu ¿Cuánto ha tardado la consulta?. Vuelve a hacerla, ¿Ha durado menos? ¿por qué?
17. ¿Por qué desde clase la consulta a la dirección IP dit.gonzalonazareno.org es distinta que si la hacemos desde casa?
18. ¿Qué dirección IP tiene babuino.gonzalonazareno.org y dit.gonzalonazareno.org desde clase? ¿Qué relación existe ambos nombres?
19. ¿A qué servidor mandamos el correo cuya dirección de destino es correo@josedomingo.org? ¿Y si lo mandamos a usuario@amazon.es?
20. Desde clase, ¿cómo se llama el ordenador que tiene la dirección IP 192.168.103.2?
21. ¿Por qué hay varios servidores con autoridad sobre la zona josedomingo.org?
22. Una vez que sepas la dirección del servidor con autoridad sobre la zona josedomingo.org, realiza una consulta a ese servidor preguntando por www.josedomingo.org. ¿qué proceso de consultas se sigue?

Python para sysadmins

Para automatizar muchas de las tareas que realizan los administradores de sistemas es necesario crear scripts. Éstos se pueden crear en distintos lenguajes de programación, por ejemplo bash. En este apartado vamos a introducir las posibilidades que nos ofrece python para crear script de administración.

Nota: Si quieres más información puedes consultar la página [Python for system administrators](#).

Resumen de instrucciones

Veamos algunas instrucciones que nos pueden ayudar en nuestros scripts de administración:

- **Trabajar con argumentos en la línea de comandos:** La mayoría de los scripts que realicemos recibirán la entrada por argumentos de la línea de comandos.:

```
import sys
len (sys.argv) #Número de argumentos
sys.argv[1] #Acceso al segundo argumento
```

- **Salir del programa:** Nos puede interesar que el programa termine bajo alguna circunstancia.:

```
import sys
sys.exit(0)
```

- **La librería os te permite acceder del sistema operativo:** Esta librería es muy importante para realizar scripts de administración, veamos algunos ejemplos.:

```
import os
print os.getcwd() #Devuelve el directorio donde estás trabajando
os.chdir("Descargas") #Cambia de directorio
os.system("clear") #Ejecuta una instrucción pero no podemos obtener el resultado
```

- **Ejecutar una instrucción y obtener el resultado:** Tenemos tres posibilidades.:

```
#Utilizando la librería commands
import commands
data = commands.getoutput("ls -l")
type(data)
lineas=data.split("\n")

#Otra manera, pero poco "elegante"
os.system("ls -l>tmp.txt")
f=open("tmp.txt", "r")

#Ejecutar la instrucción como si fuera un fichero
f=popen("ls -l", "r")
```

Ejercicios

Realiza un script en python que realice la siguiente función:

1. Muestra el directorio de trabajo.

2. Muestra cuantos usuarios hay definido en el sistema
3. Muestra los usuarios conectados al equipo.
4. Script que lea el nombre de un usuario, si existe dice si es administrador o no, si no existe da un mensaje de error. Realiza el script leyendo el usuario por teclado, y realiza otra versión indicándolo como parámetro.
5. Pasa por parámetros una dirección ip y un nombre de máquina e inserta en `/etc/hosts` (en la tercera línea) la resolución estática. Si no se introducen dos parámetros se da un error.
6. Para crear un usuario “a mano”:
 - Editar `/etc/passwd` y agregar una nueva línea por cada nueva cuenta. Teniendo cuidado con la sintaxis. Debería hacer que el campo de la contraseña sea `*`, de esta forma es imposible ingresar al sistema.
 - De forma similar, edite `/etc/group` para crear también un grupo.
 - Crea el directorio Inicio del usuario con el comando `mkdir`.
 - Copia los archivos de `/etc/skel` al nuevo directorio creado
 - Corrige la pertenencia del dueño y permisos con los comandos `chown` y `chmod` (Ver paginas de manual de los respectivos comandos). La opción `-R` es muy útil. Los permisos correctos varían un poco de un sitio a otro, pero generalmente los siguientes comandos harán lo correcto::

```
cd /home/nuevo-nombre-de-usuario
chown -R nombre-de-usuario:group .
chmod -R 755 .
```

- Asigne una contraseña con el comando `passwd`
- Crea un script python que cree un usuario, para ello debe recibir el nombre de usuario y nombre completo por parámetros, por defecto se pone uid y gid a 2000. Mejorar el programa para que:
- Da un error si se intenta dar de alta un usuario que ya existe
- Al ir dando de alta a distintos usuarios vaya incrementando automáticamente el uid y el gid a partir de 2000

Introducción a vagrant

Vagrant es una aplicación libre desarrollada en ruby que nos permite crear y personalizar entornos de desarrollo livianos, reproducibles y portables. Vagrant nos permite automatizar la creación y gestión de máquinas virtuales. Las máquinas virtuales creadas por vagrant se pueden ejecutar con distintos gestores de máquinas virtuales (oficialmente VirtualBox, VMWare e Hyper-V), en nuestro ejemplo vamos a usar máquinas virtuales en VirtualBox.

El objetivo principal de vagrant es aproximar los entornos de desarrollo y producción, de esta manera el desarrollador tiene a su disposición una manera muy sencilla de desplegar una infraestructura similar a la que se va a tener en entornos de producción. A los administradores de sistemas les facilita la creación de infraestructuras de prueba y desarrollo.

Presentación: [Vagrant y ansible. Una combinación explosiva \(1ª parte\)](#)

Práctica con vagrant

■ Práctica 1: Instalación de vagrant

Instalar virtualbox y vagrant:

```
root@maquina:~$ apt-get install virtualbox
root@maquina:~$ wget https://releases.hashicorp.com/vagrant/1.8.5/vagrant_1.8.5_x86_
↪64.deb
root@maquina:~$ dpkg -i vagrant_1.8.5_x86_64.deb
```

■ Práctica 2: Instalación de un “box” debian/jessie

Nos descargamos desde el repositorio oficial el box de Debian Jessie de 64 bits, esto lo hacemos un usuario sin privilegios::

```
usuario@maquina:~$ vagrant box add debian/jessie64
```

Si el box lo tenemos en la *nas* de nuestro instituto::

```
usuario@maquina:~$ vagrant box add debian/jessie64 http://nas.gonzalonazareno.org/...
```

Nota: Es importante fijarnos que lo estamos haciendo con usuarios sin privilegios. Cada usuario tendrás sus box propios.

Puedo ver la lista de boxes que tengo instalada en mi usuario ejecutando la siguiente instrucción::

```
usuario@maquina:~$ vagrant box list
```

■ Práctica 3: Creación de una máquina virtual

1. Nos creamos un directorio y dentro vamos a crear el fichero Vagrantfile, podemos crear uno vacío con la instrucción::

```
usuario@maquina:~/vagrant$ vagrant init
```

2. Modificamos el fichero Vagrantfile y los dejamos de la siguiente manera::

```
# -*- mode: ruby -*-
# vi: set ft=ruby :
Vagrant.configure("2") do |config|
  config.vm.box = "debian/jessie64"
  config.vm.hostname = "mimaquina"
  config.vm.network :public_network, :bridge=>"eth0"
end
```

3. Iniciamos la máquina::

```
usuario@maquina:~/vagrant$ vagrant up
```

4. Para acceder a la instancia::

```
usuario@maquina:~/vagrant$ vagrant ssh default
```

5. Suspender, apagar o destruir::

```
usuario@maquina:~/vagrant$ vagrant suspend
usuario@maquina:~/vagrant$ vagrant halt
usuario@maquina:~/vagrant$ vagrant destroy
```

Advertencia:

1. Entra en virtualbox y comprueba las características de la máquina que se ha creado.
2. ¿Qué usuario tiene creado por defecto el sistema? ¿Cómo se ejecutan intracciones de superusuario?
3. ¿Cuántas tarjetas de red tiene? ¿Para qué sirve la eth0?
4. Investiga el funcionamiento de la instrucción `vagrant ssh`. ¿Por qué interfaz se conecta? ¿Qué certificado se utiliza para acceder?

■ Práctica 4: Creación de varias máquinas virtuales

En esta ocasión vamos a crear otro directorio y dentro un fichero Vagrantfile con el siguiente contenido::

```
# -*- mode: ruby -*-
# vi: set ft=ruby :

Vagrant.configure("2") do |config|

  config.vm.define :nodo1 do |nodo1|
    nodo1.vm.box = "debian/jessie64"
    nodo1.vm.hostname = "nodo1"
    nodo1.vm.network :private_network, ip: "10.1.1.101"
  end
  config.vm.define :nodo2 do |nodo2|
    nodo2.vm.box = "debian/jessie64"
    nodo2.vm.hostname = "nodo2"
    nodo2.vm.network :public_network, :bridge=>"eth0"
    nodo2.vm.network :private_network, ip: "10.1.1.102"
  end
end
```

Cuando iniciemos el escenario veremos que hemos creado dos máquinas virtuales: `nodo1` y `nodo2`. `nodo1` tendrá una red interna con ip 10.1.1.101, y `nodo2` tendrá una interfaz de red “modo puente” y una interfaz de red del tipo red interna con ip 10.1.1.102.

Si accedemos por ssh a `nodo1` podremos hacer ping a `nodo2`.

■ Práctica 5: Añadir un disco duro adicional y modificar la RAM a una máquina virtual

Por últimos vamos a crear un nuevo Vagrantfile en un nuevo directorio con este contenido::

```
# -*- mode: ruby -*-
# vi: set ft=ruby :

Vagrant.configure("2") do |config|

  config.vm.define :nodo1 do |nodo1|
    nodo1.vm.box = "debian/jessie64"
    nodo1.vm.hostname = "nodo1"
    nodo1.vm.network :private_network, ip: "10.1.1.101"
    nodo1.vm.provider :virtualbox do |v|
      v.customize ["modifyvm", :id, "--memory", 768]
    end
  end

  disco = '.vagrant/midisco.vdi'
  config.vm.define :nodo2 do |nodo2|
```

```

nodo2.vm.box = "debian/jessie64"
nodo2.vm.hostname = "nodo2"
nodo2.vm.network :public_network, :bridge=>"eth0"
nodo2.vm.network :private_network, ip: "10.1.1.102"
nodo2.vm.provider :virtualbox do |v|
    v.customize ["createhd", "--filename", disco, "--size", 1024]
    v.customize ["storageattach", :id, "--storagectl", "SATA_
↩Controller",
                "--port", 1, "--device", 0, "--type", "hdd",
                "--medium", disco]
    end
end
end

```

Como podemos ver al nodo1 le hemos modificado el tamaño de la memoria RAM y en el nodo2 hemos añadido un disco duro de 1GB. Para que estos cambios tengan efecto debes ejecutar la instrucción::

```

usuario@maquina:~/vagrant$ vagrant reload

```

Para terminar, indicar que tenemos más parámetros de configuración que nos permiten configurar otros aspectos de la máquina virtual. Puedes encontrar más información en la [documentación oficial de vagrant](#).

Enlaces interesantes

- [Página oficial de Vagrant](#)
- [Gestionando máquinas virtuales con Vagrant](#)
- [Boxes oficiales para Vagrant](#)
- [Boxes no oficiales de Vagrant](#)

Servidor DHCP

El protocolo de configuración dinámica de host (**DHCP, Dynamic Host Configuration Protocol**), es un estándar TCP/IP que simplifica la administración de la configuración IP haciéndola automática. El servidor DHCP recibe peticiones de clientes solicitando una configuración de red IP. Responde proporcionando los parámetros que permitan a los clientes autoconfigurarse. Los clientes hay que configurarlo seleccionando la opción '*Obtener dirección IP automáticamente*'. El servidor DHCP proporciona la configuración red, entre los datos que ofrece:

- Dirección IP
- Máscara de red
- Dirección del servidor DNS
- Nombre DNS
- Puerta de enlace de la dirección IP
- Dirección de Publicación Masiva (broadcast address)
- MTU (Unidad de Transferencia Máxima según siglas en inglés) para la interfaz
- Servidores NTP (Protocolo de Tiempo de Red)
- Servidor SMTP
- Servidor TFTP

Al trabajar con el servidor DHCP tenemos que conocer los siguientes conceptos:

- **Ámbito servidor DHCP:** Agrupamiento administrativo de equipos o clientes de una subred que utilizan el servicio DHCP.
- **Rango servidor DHCP:** Grupo de direcciones IP en una subred que el servidor puede conceder a los clientes
- **Concesión o alquiler de direcciones:** Período de tiempo que los servidores DHCP especifican, durante el cual un equipo cliente puede utilizar una dirección IP.
- **Reserva de direcciones IP:** Direcciones IP que se asignan siempre a los mismos equipos. Es similar a configurar una dirección IP estática pero de forma automática desde el servidor DHCP, la forma de hacerlo es asociar direcciones MAC a direcciones IP.

- **Exclusiones:** Conjunto de direcciones IP pertenecientes al rango que no se van a asignar.

Índice

Enlaces interesantes

- [Wikipedia - APIPA](#)
- [Internet Systems Consortium - dhcp](#)
- [Artículo: How to Install the DHCP Server on Ubuntu 12.04LTS](#)
- [Servidor dhcp en Windows 2008 Server](#)

¿Cómo funciona el servidor DHCP

El protocolo de configuración dinámica de host (Dynamic Host Configuration Protocol – DHCP) es una extensión de protocolo BOOTP que da más flexibilidad al administrar las direcciones IP. Este protocolo puede usarse para configurar dinámicamente los parámetros esenciales TCP/IP de los hosts (estaciones de trabajo y servidores) de una red. El protocolo DHCP tiene dos elementos:

- Un mecanismo para asignar direcciones IP y otros parámetros TCP/IP.
- Un protocolo para negociar y transmitir información específica del host.

El host TCP/IP que solicita la información de configuración TCP/IP se denomina cliente DHCP y el host que provee dicha información se llama servidor DHCP. El DHCP se describe en la norma **RFC 2131** –Protocolo de configuración dinámica de host–. A continuación, presentamos la operación del DHCP.

Administración de direcciones con el DHCP

El protocolo DHCP usa los siguientes 3 métodos para asignar las direcciones IP:

- Asignación manual** El administrador de red pone manualmente la dirección IP del cliente DHCP en el servidor DHCP. El DHCP se usa para dar al cliente DHCP el valor de esta dirección IP configurada manualmente.
- Asignación automática** No se requiere asignar manualmente direcciones IP. El servidor DHCP asigna al cliente DHCP, en el primer contacto, una dirección IP permanente que no podrá reutilizar ningún otro cliente DHCP.
- Asignación dinámica** El DHCP asigna una dirección IP al cliente DHCP por un tiempo determinado. Después que expire este lapso, se revoca la dirección IP y el cliente DHCP tiene que devolverla. Si el cliente aún necesita una dirección IP para efectuar sus operaciones, deberá solicitarla nuevamente.

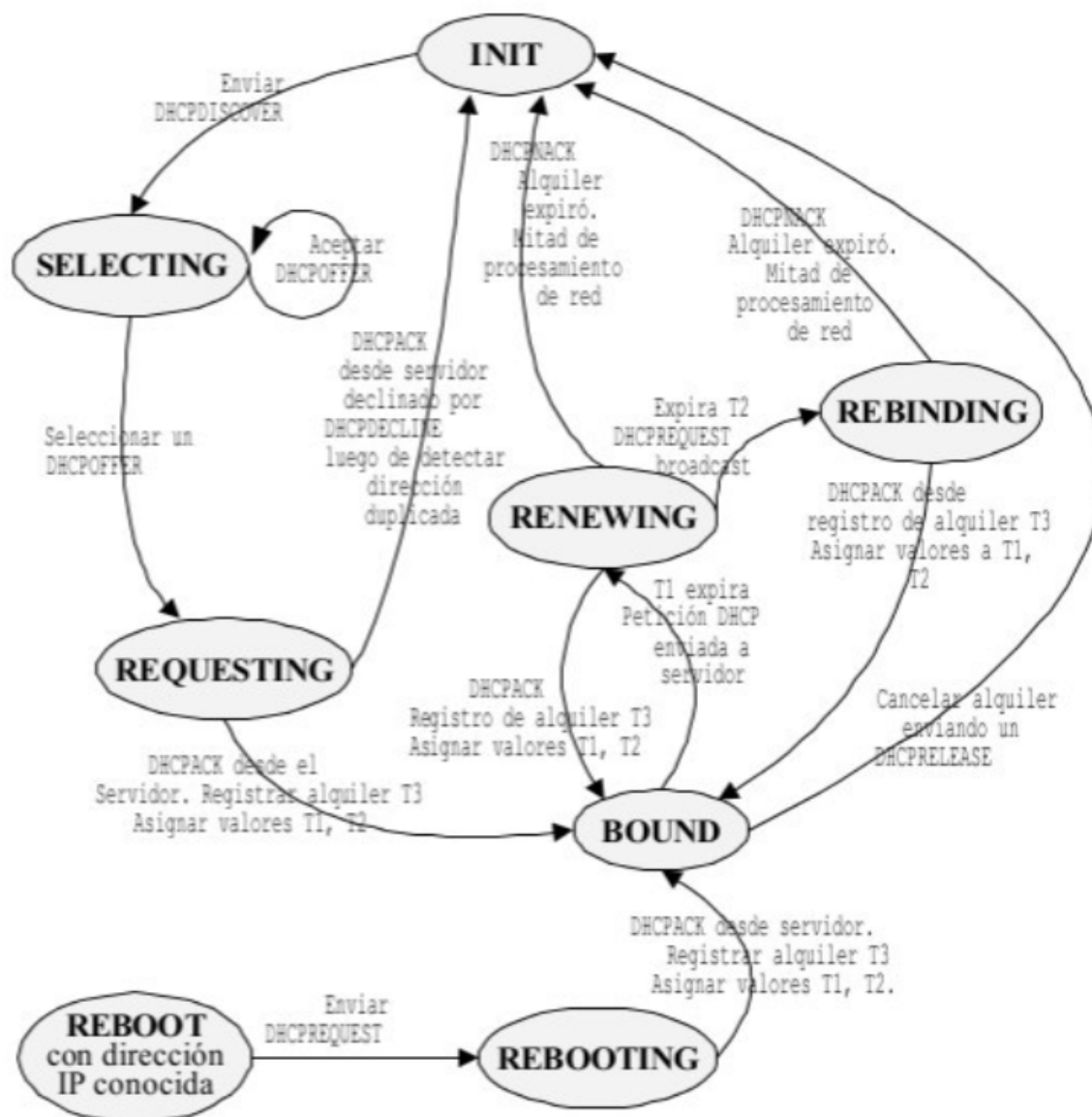
Este protocolo permite la reutilización automática de una dirección IP. Si un cliente DHCP ya no necesita una dirección IP, como en el caso de un ordenador que apagamos, éste libera su dirección y la entrega al servidor DHCP. Éste puede reasignar dicha dirección a otro cliente que la pida.

El método de asignación dinámica es muy útil para clientes DHCP que necesitan una dirección IP para una conexión temporal a la red. Por ejemplo, consideremos una situación en que 300 usuarios tengan ordenadores portátiles conectadas a una red y ésta les ha asignado direcciones clase C. Este tipo de dirección permite a la red tener hasta 253 nodos ($255 - 2$ direcciones especiales = 253). Debido a que los ordenadores que se conectan a una red usando el TCP/IP requieren tener una dirección única IP, entonces las 300 ordenadores no podrían operar simultáneamente. Sin embargo, si sólo hay 200 conexiones físicas a la red se puede buscar una dirección de clase C mediante la reutilización de direcciones IP no usadas. Usando el DHCP, en su método de asignación dinámica de direcciones IP, es posible reutilizar direcciones IP.

Además la asignación dinámica de direcciones IP es un buen método para asignar direcciones IP a ordenadores que van a ser conectados por primera vez y en una red donde escasean las direcciones IP. Si los ordenadores antiguos se retiran, sus direcciones IP pueden ser reutilizadas o reasignadas inmediatamente. Sin importar cuál método se elija, aún puede configurarse los parámetros IP de una sola vez desde un servidor central, en lugar de repetir la configuración TCP/IP para cada ordenador.

Proceso de configuración de los clientes

Una vez que un cliente DHCP ha contactado con un servidor DHCP, a través de varios estados internos, negocia el uso y la duración de su dirección IP. La forma de adquisición de la dirección IP por el cliente DHCP se explica mejor en términos de un diagrama de transición de estados (llamado también máquina de estado finito) . La figura presenta este diagrama de transición de estados que explica la interacción entre el cliente y el servidor DHCP.



1. Descubrimiento de un servidor DHCP (SELECTING)

Cuando se inicializa el cliente DHCP, éste comienza en el estado de inicialización INIT. El cliente DHCP desconoce sus parámetros IP y por eso envía un broadcast DHCPDISCOVER. El mensaje DHCPDISCOVER se encapsula en un paquete UDP. Se coloca el número 67 con puerta de destino UDP, el mismo utilizado por el servidor BOOTP, debido a que el protocolo DHCP es una extensión de este protocolo. El mensaje DHCPDISCOVER usa la dirección IP de broadcast de valor 255. 255. 255. 255. Si no existe un servidor DHCP en la red local, el router IP debe tener un agente DHCP relay que soporte la retransmisión de esta petición hacia las otras subredes. El agente DHCP relay se describe en la norma RFC 1542.

Antes de enviar el mensaje broadcast DHCPDISCOVER, el cliente DHCP espera por un tiempo aleatorio –entre 1 a 10 segundos– para evitar una colisión con otro cliente DHCP, como en el caso que todos los clientes DHCP se inicialicen al mismo tiempo al recibir todos energía a la vez (como una pérdida o interrupción de la electricidad).

2. Aceptación de la asignación recibida (REQUESTING)

Después de enviar el mensaje broadcast DHCPDISCOVER, el cliente DHCP ingresa al estado SELECTING, donde recibe los mensajes DHCPOFFER de los servidores DHCP configurados para atenderlo. El tiempo que el cliente DHCP esperará por los mensajes DHCPOFFER depende de la implementación. Si el cliente DHCP recibe varias respuestas DHCPOFFER, elegirá una. En reacción, el cliente DHCP enviará un mensaje DHCPREQUEST para elegir un servidor DHCP, el que contestará con un DHCPACK.

Como opción, el cliente DHCP controla la dirección IP enviada en el DHCPACK para verificar si está o no está en uso. En una red con broadcast, el cliente DHCP envía una petición ARP con la dirección IP sugerida para verificar que no esté duplicada. En caso de estarlo, el DHCPACK proveniente del servidor se ignora y se envía un DHCPDECLINE, con lo cual el cliente DHCP ingresa en estado INIT y vuelve a pedir una dirección IP válida que no esté en uso.

Cuando la petición ARP se difunde sobre la red, el cliente usa su propia dirección de hardware en el campo de dirección fuente de hardware del ARP, pero coloca el valor de 0 en el campo de dirección fuente IP. Esta dirección de valor 0 se utiliza en lugar de la dirección IP sugerida, para no confundir a las memorias caché ARP de otros hosts.

3. Duración de la concesión (BOUND)

Cuando se acepta el DHCPACK proveniente del servidor DHCP, se colocan tres valores de temporización y el cliente DHCP se mueve al estado BOUND (asociado) .

- T1 es el temporizador de renovación de alquiler.
- T2 es el temporizador de reenganche.
- T3 es la duración del alquiler.

El DHCPACK siempre trae consigo el valor de T3. Los valores de T1 y T2 se configuran en el servidor DHCP; de no ser así, se usan los valores por defecto siguientes:

- $T1 = 0,5 \times T3$.
- $T2 = 0,875 \times T3$.

El tiempo actual en que los temporizadores expiran se calcula añadiendo el valor del temporizador al tiempo en que se envió el mensaje DHCPREQUEST, el cual generó la respuesta DHCPACK.

Si este tiempo es T0, entonces los valores de expiración se calculan así:

- Expiración de T1 = $T0 + T1$
- Expiración de T2 = $T0 + T2$
- Expiración de T3 = $T0 + T3$

La RFC 2131 recomienda que se debe añadir un factor a T1 y T2 para evitar que varios clientes DHCP expiren sus temporizadores al mismo tiempo.

4. Renovación de la concesión (RENEWING)

Después de la expiración del temporizador T1, el cliente DHCP se mueve del estado BOUND al estado RENEWING (renovación). En este último estado se debe negociar un nuevo alquiler para la dirección IP designada, entre el cliente DHCP y el servidor DHCP que originalmente le asignó la dirección IP. Si el servidor DHCP original, por algún motivo, no renueva el alquiler, le enviará un mensaje DHCPNACK y el cliente DHCP se moverá al estado INIT y intentará obtener una nueva dirección IP. En el caso contrario, si el servidor DHCP original envía un mensaje DHCPACK, éste contendrá la duración del nuevo alquiler. Entonces, el cliente DHCP coloca los valores de sus temporizadores y se moverá al estado BOUND.

5. Estado de reenganche (RENEWING)

Si el temporizador T2 (tiempo de reenganche) expira mientras el cliente DHCP está esperando en el estado RENEWING una respuesta sea DHCPACK o DHCPNACK proveniente del servidor DHCP original, el cliente DHCP se moverá al estado REBINDING. El servidor original DHCP podría no haber respondido porque estaría apagado o porque el enlace con la red habría caído. Nótese en las ecuaciones previas que T2 es mayor que T1, de modo que el cliente DHCP espera que el servidor original DHCP renueve el alquiler por un tiempo igual a $T2 - T1$.

6. Extensión de la concesión

Al expirar el temporizador T2 (tiempo de reenganche), el cliente DHCP enviará un DHCPREQUEST a la red para contactar con cualquier servidor DHCP para extender el alquiler, con lo cual pasará al estado REBINDING. El cliente DHCP envía este mensaje broadcast DHCPREQUEST porque presume que, luego de haber esperado $T2 - T1$ segundos en el estado RENEWING, el servidor DHCP original no está disponible, por lo cual tratará de contactar con otro servidor DHCP para que le responda. Si un servidor DHCP responde con un DHCPACK, el cliente DHCP renueva su alquiler (T3), coloca los temporizadores T1 y T2 y retorna al estado BOUND. Si no hay servidor DHCP disponible para renovar alquiler luego de expirar el temporizador T3, el alquiler cesa y el cliente DHCP pasa al estado INIT. Nótese que el cliente DHCP intentó renovar el alquiler primero con el servidor original y luego con cualquier otro servidor en la red.

7. Expiración de la concesión

Al acabar el alquiler (T3 expira), el cliente DHCP debe devolver su dirección IP y cesar toda acción con dicha dirección IP en la red. El cliente DHCP no siempre tiene que esperar la expiración del alquiler para terminar el uso de una dirección IP. Éste puede renunciar voluntariamente a una dirección IP, cancelando su alquiler. Por ejemplo, el usuario de un computador portátil podría conectarse a la red para una actividad particular. El servidor DHCP de la red podría colocar la dirección del alquiler por una hora. Suponiendo que el usuario acabe su tarea en 30 minutos, entonces se desconectará de la red al cabo de dicho lapso. Cuando el usuario se libera armoniosamente, el cliente DHCP enviará un mensaje DHCPRELEASE al servidor DHCP para cancelar el alquiler. La dirección IP ahora estará disponible.

Si los clientes DHCP operan en ordenadores que tienen disco duro, la dirección IP asignada puede ser almacenada en este dispositivo y, cuando la computadora reinicie sus operaciones, puede hacer una nueva petición usando esta dirección IP.

Cuestionario DHCP

1. ¿Qué es un servidor DHCP?
2. ¿Cómo hay que configurar un cliente DHCP?
3. Ventajas de usar un servidor DHCP
4. Enumera los distintos parámetros que un servidor DHCP puede conceder a un cliente
5. Definición de:
 - Ámbito servidor DHCP
 - Rango servidor DHCP
 - Concesión o alquiler de direcciones

- Reserva de direcciones IP
 - Exclusión de direcciones IP
6. ¿Qué es APIPA?
 7. ¿Por qué el mensaje DHCPDISCOVER es del tipo broadcast?
 8. Desde el estado INIT al estado BOUND. ¿Qué mensajes se transmiten desde el cliente al servidor?
 9. ¿Qué significa el mensaje DHCPDECLINE por parte del cliente?
 10. Una vez que se ha aceptado una asignación (DHCPACK), explica los siguientes tiempos:
 - T1: tiempo de renovación de alquiler
 - T2: tiempo de reenganche
 - T3: tiempo de concesión del alquiler
 11. ¿Qué es la renovación (RENEWING) de alquiler?
 12. ¿Qué es el estado de reenganche (REBINDING)?
 13. ¿Qué ocurre cuando termina el tiempo de concesión del alquiler (T3)?
 14. ¿Qué ocurre cuando un cliente manda un DHCPRELEASE?
 15. ¿Qué ocurre cuando un cliente que ya tiene asignación se reinicia?
 16. Los clientes toman una configuración, y a continuación apagamos el servidor dhcp. ¿qué ocurre con el cliente windows? ¿Y con el cliente linux?
 17. Los clientes toman una configuración, y a continuación cambiamos la configuración del servidor dhcp (por ejemplo el rango). ¿qué ocurriría con un cliente windows? ¿Y con el cliente linux?
 18. ¿Qué instrucciones en Windows y en Linux nos permiten?:
 - Para renovar una dirección IP y una nueva configuración de red
 - Para liberar la dirección IP
 19. ¿Qué es la lista de concesiones? ¿En qué fichero se guarda en Linux?
 20. En el servidor isc-dhcp en linux. ¿Qué indican los siguientes parámetros?:
 - max-lease-time
 - default-lease-time
 21. En el cliente, ¿qué se guarda en los ficheros dhclient-???.lease?

Ejercicio: Instalación y configuración del servidor dhcp en linux

Después de leer la documentación, instala el servidor dhcp. Recuerda que al inicializar el servicio nos dará un error, esto es debido a que no hemos configurado el servidor.

Instalación del servidor isc-dhcp-server

Para instalar nuestro servidor dhcp ejecutamos::

```
apt-get install isc-dhcp-server
```

Nota: Cuando instalamos el servidor por primera se produce un error, ya que no está configurado. Puedes ver los errores producidos por el servidor en el fichero `/var/log/syslog`

Configuración del servidor isc-dhcp-server

Lo primero que tenemos que hacer es configurar el interfaz de red por el que va a trabajar el servidor dhcp, para ello editamos el siguiente fichero::

```
/etc/default/isc-dhcp-server
```

Donde configuramos el parámetro interfaces, por ejemplo::

```
INTERFACES="eth1"
```

El fichero principal de configuración de DHCP es::

```
/etc/dhcp/dhcpd.conf
```

El fichero de configuración está dividido en dos partes:

- Parte principal (valores por defecto): especifica los parámetros generales que definen la concesión y los parámetros adicionales que se proporcionarán al cliente.
- **Secciones (concretan a la principal)**
 - Subnet: Especifican rangos de direcciones IPs que serán cedidas a los clientes que lo soliciten.
 - Host: Especificaciones concretas de equipos.

En la parte principal podemos configurar los siguientes parámetros, que más tarde podremos reescribir en las distintas secciones:

- `max-lease-time`: Tiempo de la concesión de la dirección IP
- `default-lease-time`: Tiempo de renovación de la concesión
- `option routers`: Indicamos la dirección red de la puerta de enlace que se utiliza para salir a internet.
- `option domain-name-server`: Se pone las direcciones IP de los servidores DNS que va a utilizar el cliente.
- `option domain-name`: Nombre del dominio que se manda al cliente.
- `option subnetmask`: Subred enviada a los clientes.
- `option broadcast-address`: Dirección de difusión de la red.

Al indicar una sección subnet tenemos que indicar la dirección de la red y la mascara de red y entre llaves podemos poner los siguientes parámetros:

- `range`: Indicamos el rango de direcciones IP que vamos a asignar.
- Algunos de los parámetros que hemos explicado en la sección principal.

Ejemplo de configuración de la sección subnet puede ser::

```
subnet 192.168.0.0 netmask 255.255.255.0 {
    range 192.168.0.60 192.168.0.90;
    option routers 192.168.0.254;
```

```
} option domain-name-server 80.58.0.33, 80.58.32.9;
```

Reiniciamos el servidor dhcp::

```
service isc-dhcp-server restart
```

Sólo falta configurar los clientes para que tomen la configuración de red de forma dinámica.

Nota: En Windows la instrucción `ipconfig /release` libera la concesión, la instrucción `ipconfig /renew` la renueva. En linux el comando para liberar la concesión es `dhclient -r` y el que nos permite renovarla será `dhclient`.

Advertencia: Ejercicios

1. Configura el servidor dhcp con las siguientes características
 - Rango de direcciones a repartir: 192.168.0.100 - 192.168.0.110
 - Máscara de red: 255.255.255.0
 - Duración de la concesión: 1 hora
 - Puerta de enlace: 192.168.0.1
 - Servidores DNS: 8.8.8.8, 8.8.4.4
2. Configura los clientes para obtener direccionamiento dinámico. Comprueba las configuraciones de red que han tomado los clientes. Visualiza el fichero del servidor donde se guarda las configuraciones asignadas.

Creando reservas

Veamos la sección host, en ella configuramos un host para reservar una dirección IP para él.

En una sección host debemos poner el nombre que identifica al host y los siguientes parámetros:

- `hardware ethernet`: Es la dirección MAC de la tarjeta de red del host.
- `fixed-address`: La dirección IP que le vamos a asignar.
- Podemos usar también las opciones ya explicadas en la sección principal.

Advertencia: Ejercicios

1. Crea en el servidor dhcp una sección HOST para conceder a un cliente una dirección IP determinada (por ejemplo la 192.168.0.105)
2. Obtén una nueva dirección IP en el cliente y comprueba que es la que has asignado por medio de la sección host.

Advertencia: Realiza las siguientes comprobaciones

Vamos a comprobar que ocurre con la configuración de los clientes en determinadas circunstancias, para ello vamos a poner un tiempo de concesión muy bajo.

1. Los clientes toman una configuración, y a continuación apagamos el servidor dhcp. ¿qué ocurre con el cliente windows? ¿Y con el cliente linux?
2. Los clientes toman una configuración, y a continuación cambiamos la configuración del servidor dhcp (por ejemplo el rango). ¿qué ocurre con el cliente windows? ¿Y con el cliente linux?

En este bloque del módulo vamos a estudiar el servidor Web, programa software que utilizando el protocolo HTTP, es capaz de procesar en el servidor las peticiones HTTP y generar las respuestas adecuadas. En este bloque vamos a estudiar distintos aspectos del servicio web:

- El protocolo HTTP: tipos de peticiones, tipos de respuestas, cabeceras, autenticación, control de acceso, etc.
- **Vamos a usar el servidor Apache 2.4:**
 - Configuración básica
 - Virtual Hosting
 - Mapeo de URL
 - Autenticación y control de acceso
 - Analizador de logs
 - Módulos en Apache
 - Ejecución de scripts en el servidor: PHP, Python,...
- Estudios de diferentes servidores web: nginx, IIS, lightHTTP, etc.
- **Optimización del rendimiento en servidores web**
 - Módulos de multiprocesamiento en Apache 2.4
 - estudio del rendimiento de servidores web

Indice

Enlaces interesantes

Protocolo HTTP

- [Introducción al protocolo HTTP](#)

- Protocolo HTTP (Univ. Valencia) Vídeo 1
- Protocolo HTTP (Univ. Valencia) Vídeo 2
- Qué son los MIME Types
- Lista de cabeceras HTTP 1.1

Servidor web Apache

- Documentación oficial servidor web Apache 2.2
- Documentación oficial servidor web Apache 2.4
- Updating Virtual Host Settings from Apache 2.2 to Apache 2.4
- How To Migrate your Apache Configuration from 2.2 to 2.4 Syntax

Módulos en apache

- Utilización de módulos en Apache
- Cómo crear URLs amigables con htaccess

Ejercicio: Hacer peticiones HTTP: GET, HEAD y POST

1. Utilizando el comando de linux HEAD visualiza la información de la cabeceras de los URL:

```
http://dit.gonzalonazareno.org
http://informatica.gonzalonazareno.org/proyectos/index.html
http://josedom24.github.io/img/yol.jpg
```

Identifica todos los parámetros que puedas.

Utiliza el plugin de firefox **HttpFox** para identificar las cabeceras de las peticiones y de las respuestas.

2. Utilizando el método GET obtén el contenido de la página:

```
http://dit.gonzalonazareno.org/moodle/index.php
http://dit http://www.debian.org/index.html
```

Observa con **HttpFox** cuantas peticiones se realizan al acceder a estas páginas.

3. Envío de información al servidor, comprueba como se manda información al servidor mediante el método GET en la URL:

```
http://secretaria-iesgn.rhcloud.com/php/ejget.php?valor=hola
http://dit.gonzalonazareno.org/moodle/course/view.php?id=4
```

Usando el comando GET manda tu nombre a la página:

```
http://secretaria-iesgn.rhcloud.com/php/ejget.php
```

Usando el comando POST (que envia el contenido en el cuerpo) manda tu nombre a la página:

```
http://secretaria-iesgn.rhcloud.com/php/ejpost.php
```

Ejercicio: Instalación y configuración básica de Apache

Instalación de Apache 2.4

1. Instala el servidor web Apache:

```
apt-get install apache2
```

Para controlar el servicio apache2 podemos usar (para más [información](#)):

```
apache2ctl [-k start|restart|graceful|graceful-stop|stop]
```

2. ¿Qué es la opción graceful?
3. Comprueba la directiva donde indicamos el puerto de escucha del servidor. Modifica el puerto de escucha para que sea el 8080. Comprueba el acceso al servidor desde un navegador.
4. Comprueba los módulos cargados en el servidor.
5. Comprueba los sitios webs activos en nuestro servidor.

Estructura de los ficheros de configuración

1. Las directivas de configuración de apache2 se pueden aplicar si está definido un determinado parámetro. Para esto se utiliza la directiva * [IfDefine](#). Busca en algún fichero de configuración esta directiva.
2. Igualmente podemos aplicar determinadas directivas si hay cargado un determinado módulo, para ello usamos [IfModule](#). Busca alguna directiva de este tipo.

Para cargar dinámicamente los módulos se utiliza la directiva [LoadModule](#), búscalos en los ficheros `.load` dentro de `/etc/apache2/mods-availables`.

Más información: <http://httpd.apache.org/docs/2.4/dso.html>

3. Busca en la configuración una variable de entorno y determina en que fichero están definidas.
4. La directiva [Include](#) nos permite añadir ficheros de configuración a la configuración general de apache2. Comprueba qué ficheros son añadidos con esta directiva.
5. Podemos aplicar directivas a partes concretas de nuestro servidor web, para ello estudia las siguientes directivas (Para aprender más lee [Secciones de Configuración](#)):

- [Directory](#)
- [DirectoryMatch](#)
- [Files](#)
- [FilesMatch](#)
- [Location](#)
- [LocationMatch](#)
- [VirtualHost](#)

Localiza algunas de ellas en los ficheros de configuración.

6. ¿Qué son los ficheros `.htaccess`?

Configuración básica de Apache

Busca en el fichero de configuración de Apache las siguientes directivas, determina que función tienen y el valor que poseen por defecto.

Directivas de identificación del servidor:

- `ServerName`
- `ServerAdmin`
- `ServerTokens`

Modifica el valor de `ServerAdmin` y `ServerTokens` y comprueba los resultados.

Directivas de localización de ficheros

- `DocumentRoot`
- `ErrorLog`
- `CustomLog`
- `LockFile`
- `PidFile`
- `ServerRoot`
- `AccessFileName`

Directivas de control de la conexión

- `Timeout`
- `KeepAlive` (Más información)
- `MaxKeepAliveRequests`
- `KeepAliveTimeout`

Otras directivas

- `User`
- `Group`
- `DefaultType`
- `LogLevel`
- `LogFormat`

Ejercicio: VirtualHosting con Apache

Introducción al VirtualHosting

El término Hosting Virtual se refiere a hacer funcionar más de un sitio web (tales como `www.company1.com` y `www.company2.com`) en una sola máquina. Los sitios web virtuales pueden estar “basados en direcciones IP”, lo que significa que cada sitio web tiene una dirección IP diferente, o “basados en nombres diferentes”, lo que significa que con una sola dirección IP están funcionando sitios web con diferentes nombres (de dominio). Apache fue uno de los primeros servidores web en soportar hosting virtual basado en direcciones IP.

El servidor web Apache2 se instala por defecto con un host virtual. La configuración de este sitio la podemos encontrar en:

```
/etc/apache2/sites-available/000-default.conf
```

Cuyo contenido podemos ver:

```
<VirtualHost *:80>
    #ServerName www.example.com
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

Donde encontramos los siguientes parámetros:

- `ServerName`
- `ServerAdmin`
- `DocumentRoot`
- `ErrorLog`
- `CustomLog`

Y por defecto este sitio virtual está habilitado, por lo que podemos comprobar que existe un enlace simbólico a este fichero en el directorio `/etc/apache2/sites-enabled`:

```
lrwxrwxrwx 1 root root 35 Oct 3 15:24 000-default.conf -> ../sites-available/000-
->default.conf
```

Podemos habilitar o deshabilitar nuestros host virtuales utilizando los siguientes comandos:

```
a2ensite
a2dissite
```

En el fichero de configuración general `/etc/apache2/apache2.conf` nos encontramos las opciones de configuración del directorio padre del indicado en la directiva `DocumentRoot` (suponemos que todos los host virtuales van a estar guardados en subdirectorios de este directorio):

```
...
<Directory /var/www/>
    Options Indexes FollowSymLinks
    AllowOverride None
    Require all granted
</Directory>
...
```

Configuración de VirtualHosting

El objetivo de esta práctica es la puesta en marcha de dos sitios web utilizando el mismo servidor web apache. Hay que tener en cuenta lo siguiente:

- Cada sitio web tendrá nombres distintos.
- Cada sitio web compartirán la misma dirección IP y el mismo puerto (80).

Queremos construir en nuestro servidor web apache dos sitios web con las siguientes características:

- El nombre de dominio del primero será `www.iesgn.org`, su directorio base será `/var/www/iesgn` y contendrá una página llamada `index.html`, donde sólo se verá una bienvenida a la página del Instituto Gonzalo Nazareno.
- En el segundo sitio vamos a crear una página donde se pondrán noticias por parte de los departamento, el nombre de este sitio será `www.departamentosgn.org`, y su directorio base será `/var/www/departamentos`. En este sitio sólo tendremos una página inicial `index.html`, dando la bienvenida a la página de los departamentos del instituto.

Para conseguir estos dos sitios virtuales debes seguir los siguientes pasos:

1. Los ficheros de configuración de los sitios webs se encuentran en el directorio `/etc/apache2/sites-available`, por defecto hay dos ficheros, uno se llama `000-default.conf` que es la configuración del sitio web por defecto. Necesitamos tener dos ficheros para realizar la configuración de los dos sitios virtuales, para ello vamos a copiar el fichero `000-default.conf`:

```
cd /etc/apache2/sites-available
cp 000-default.conf iesgn.conf
cp 000-default.conf departamentos.conf
```

De esta manera tendremos un fichero llamado `iesgn.conf` para realizar la configuración del sitio web `www.iesgn.org`, y otro llamado `departamentos.conf` para el sitio web `www.departamentosgn.org`.

2. Modificamos los ficheros `iesgn.conf` y `departamentos.conf`, para indicar el nombre que vamos ausar para acceder al host virtual (ServerName) y el directorio detrabajo (DocumentRoot).
3. No es suficiente crear los ficheros de configuración de cada sitio web, es necesario crear un enlace simbólico a estos ficheros dentro del directorio `/etc/apache2/sites-enabled`, para ello:

```
a2ensite iesgn
a2ensite departamentos
```

La creación de los enlaces simbólicos se puede hacer con la instrucción `a2ensite nombre_fichero_configuracion`, para deshabilitar el sitio tenemos que borrar el enlace simbólico o usar la instrucción `a2dissite nombre_fichero_configuracion`.

4. Crea los directorios y los ficheros `index.html` necesarios en `/var/www` y reiniciamos el servicio.
5. Para terminar lo único que tendremos que hacer es cambiar el fichero `hosts` en los clientes y poner dos nuevas líneas donde se haga la conversión entre los dos nombre de dominio y la dirección IP del servidor.

Nota: Repite el ejercicio cambiando los directorios de trabajo a `/srv/www`. ¿Qué modificación debes hacer en el fichero `/etc/apache2/apache2.conf`?

Ejercicio: Mapear URL a ubicaciones de un sistema de ficheros

1. Crea un nuevo host virtual que es accedido con el nombre `www.mapeo.com`, cuyo `DocumentRoot` sea `/srv/mapeo`.
2. **Alias:** Crea un alias en el host virtual del ejercicio anterior, que mi permita entrar en la URL `http://www.mapeo.com/documentos` y visualice los ficheros del `/home/usuario/Documentos`. En la sección `Directory...` pon las mismas directivas que tiene la sección `Directory` del fichero `/etc/apache2/apache2.conf`.
3. **Options:** Determina para que sirven las siguientes opciones de funcionamiento:
 - All

- FollowSymLinks
- Indexes
- MultiViews
- SymLinksOwnerMatch
- ExecCGI

Detemina como funciona si delante de las opciones pongo el signo + o -.

- Crea un enlace directo dentro de `/home/usuario/document` y comprueba si es posible seguirlo. Cambia las opciones del directorio para que no siga los enlaces simbólicos.
 - Deshabilita la opción de que se listen los archivos existentes en la carpeta cuando no existe un fichero definido en la directiva `DirectoryIndex`.
 - MultiViews: Para saber más sobre el [negociado de contenido](#). Siguiendo el ejemplo de esta [página](#) realiza un fichero de bienvenida en español e inglés y comprueba como se visualiza.
4. Usando la directiva `Redirect` realiza una redirección, que permita que cuando entre a tu servidor `http://nombre_servidor`, salte a `http://nombre_servidor/web`.
 5. Con la directiva `ErrorDocument` se puede crear [Respuesta de error personalizadas](#). Todo esto se puede llevar a cabo en el fichero `/etc/apache2/conf-available/localized-error-pages.conf`. Después de leer sobre el tema realiza los siguientes ejercicios.
 - Cuando no se encuentre una página (error 404) por un mensje de error.
 - Crea un alias llamado `error` que corresponda a `/srv/mapeo/error`. Dentro de ese directorio crea páginas personalizadas para visualizar cuando se produzca un error 404 y cuando se tenga un forbidden (403). Configura el sistema para que se redireccione a estas páginas cuando se produce un error.
 - **Descomenta en el fichero `localized-error-pages.conf` las líneas adecuadas para tener los mensajes de error traducidos.**
 - Activar el módulo `include`.
 - Si quieres los mensajes en español modifica adecuadamente la directiva `LanguagePriority` del módulo `negotiation`.

Ejercicio: Control de acceso, autenticación y autorización

Control de acceso

El **Control de acceso** en un servidor web nos permite determinar desde donde podemos acceder a los recursos del servidor.

En **apache2.2** se utilizan las siguientes directivas: `order`, `allow` y `deny`. Un buen manual para que quede más claro lo puedes encontrar en este [enlace](#).

En **apache2.4** se utilizan las siguientes directivas: `Require`, `RequireAll`, `RequireAny` y `RequireNone`

1. Comprueba el control de acceso por defecto que tiene el virtual host por defecto (000-default).
2. Crea un escenario en Vagrant que tenga un servidor con una red publica, y una privada, un cliente conectada a la red privada. Crea un host virtual, que sólo se tenga acceso desde el cliente de la red local, y no se pueda acceder desde la anfitriona por la red pública.

Autenticación básica

El servidor web Apache puede acompañarse de distintos módulos para proporcionar diferentes modelos de autenticación. La primera forma que veremos es la más simple. Usamos para ello el módulo de autenticación básica que viene instalada “de serie” con cualquier Apache: `mod_auth_basic`. La configuración que tenemos que añadir en el fichero de definición del Virtual Host a proteger podría ser algo así:

```
<Directory "/var/www/miweb/privado">
  AuthUserFile "/etc/apache2/claves/passwd.txt"
  AuthName "Palabra de paso"
  AuthType Basic
  Require valid-user
</Directory>
```

El método de autenticación básica se indica en la directiva `AuthType`.

- En `Directory` escribimos el directorio a proteger, que puede ser el raíz de nuestro Virtual Host o un directorio interior a este.
- En `AuthUserFile` ponemos el fichero que guardará la información de usuarios y contraseñas que debería de estar, como en este ejemplo, en un directorio que no sea visitable desde nuestro Apache. Ahora comentaremos la forma de generarlo.
- Por último, en `AuthName` personalizamos el mensaje que aparecerá en la ventana del navegador que nos pedirá la contraseña.
- Para controlar el control de acceso, es decir, que usuarios tienen permiso para obtener el recurso utilizamos las siguientes directivas: `AuthGroupFile`, `Require user`, `Require group`.

El fichero de contraseñas se genera mediante la utilidad `htpasswd`. Su sintaxis es bien sencilla. Para añadir un nuevo usuario al fichero operamos así:

```
# htpasswd /etc/apache2/claves/passwd.txt carolina
New password:
Re-type new password:
Adding password for user carolina
```

Para crear el fichero de contraseñas con la introducción del primer usuario tenemos que añadir la opción `-c` (create) al comando anterior. Si por error la seguimos usando al incorporar nuevos usuarios borraremos todos los anteriores, así que cuidado con esto. Las contraseñas, como podemos ver a continuación, no se guardan en claro. Lo que se almacena es el resultado de aplicar una `función hash`:

```
josemaria:rOUetcAKYaliE
carolina:hmO6V4bM8KLdw
alberto:9RjyKKYK.xyhk
```

Para denegar el acceso a algún usuario basta con que borremos la línea correspondiente al mismo. No es necesario que le pidamos a Apache que vuelva a leer su configuración cada vez que hagamos algún cambio en este fichero de contraseñas.

La principal ventaja de este método es su sencillez. Sus inconvenientes: lo incómodo de delegar la generación de nuevos usuarios en alguien que no sea un administrador de sistemas o de hacer un front-end para que sea el propio usuario quien cambie su contraseña. Y, por supuesto, que dichas contraseñas viajan en claro a través de la red. Si queremos evitar esto último podemos crear una `instancia Apache con SSL`.

Cómo funciona este método de autenticación

Cuando desde el cliente intentamos acceder a una URL que esta controlada por el método de autenticación básico:

1. El servidor manda una respuesta del tipo 401 *HTTP/1.1 401 Authorization Required* con una cabecera *WWW-Authenticate* al cliente de la forma:

```
WWW-Authenticate: Basic realm="Palabra de paso"
```

2. El navegador del cliente muestra una ventana emergente preguntando por el nombre de usuario y contraseña y cuando se rellena se manda una petición con una cabecera *Authorization*

Authorization: Basic am9zZTpqb3NI

En realidad la información que se manda es el nombre de usuario y la contraseña en base 64, que se puede decodificar fácilmente con cualquier [utilidad](#).

Ejercicios

1. Crea cuatro usuarios de apache: pepe, maria, juan, ana.
2. Crea dos grupos de usuarios: grupo1 (pepe,maria), grupo2 (juan,ana).
3. Crea un directorio llamado privado1 en el host virtual default, que permita el acceso a todos los usuarios.
4. Crea un directorio llamado privado2 en el host virtual default, que permita el acceso sólo a juan y a ana.
5. Crea un directorio llamado privado3 en el host virtual default, que permita el acceso sólo los usuarios del grupo1.

La directiva [satisfy](#) controla como el se debe comportar el servidor cuando tenemos autorizaciones a niveles de host (Require, RequireAll,...) y tenemos autorizaciones de usuarios (require).

6. El directorio privado3 del ejercicio5 haz que sólo sea accesible desde el localhost, y estudia como se comporta la autorización si ponemos:

```
satisfy any
satisfy all
```

Autenticación tipo digest

La autenticación tipo digest soluciona el problema de la transferencia de contraseñas en claro sin necesidad de usar SSL. El procedimiento, como veréis, es muy similar al tipo básico pero cambiando algunas de las directivas y usando la utilidad `htdigest` en lugar de `htpasswd` para crear el fichero de contraseñas. El módulo de autenticación necesario suele venir con Apache pero no habilitado por defecto. Para activarlo usamos la utilidad `a2enmod` y, a continuación reiniciamos el servidor Apache:

```
$ a2enmod auth_digest
$ /etc/init.d/apache2 restart
```

Luego incluimos una sección como esta en el fichero de configuración de nuestro Virtual Host:

```
<Directory "/var/www/miweb/privado">
  AuthType Digest
  AuthName "dominio"
  AuthUserFile "/etc/claves/digest.txt"
  Require valid-user
</Directory>
```

Como vemos, es muy similar a la configuración necesaria en la autenticación básica. La directiva `AuthName` que en la autenticación básica se usaba para mostrar un mensaje en la ventana que pide el usuario y contraseña, ahora se usa también para identificar un nombre de dominio (realm) que debe de coincidir con el que aparezca después en el fichero de contraseñas. Dicho esto, vamos a generar dicho fichero con la utilidad `htdigest`:

```
# htdigest -c /etc/claves/digest.txt dominio josemaria
Adding password for josemaria in realm dominio.
New password:
Re-type new password:
```

Al igual que ocurría con htpasswd, la opción `-c` (create) sólo debemos de usarla al crear el fichero con el primer usuario. Luego añadiremos los restantes usuarios prescindiendo de ella. A continuación vemos el fichero que se genera después de añadir un segundo usuario:

```
josemaria:dominio:8d6af4e11e38ee8b51bb775895e11e0f gemma:dominio:dbd98f4294e2a49f62a486ec070b9b8c
```

Cómo funciona este método de autenticación

Cuando desde el cliente intentamos acceder a una URL que esta controlada por el método de autenticación de tipo digest:

1. El servidor manda una respuesta del tipo 401 *HTTP/1.1 401 Authorization Required* con una cabecera *WWW-Authenticate* al cliente de la forma:

```
WWW-Authenticate: Digest realm="dominio", nonce="cIIDldTpBAA=9b0ce6b8eff03f5ef8b59da45a1ddfca0bc0c485",  
algorithm=MD5, qop="auth"
```

2. El navegador del cliente muestra una ventana emergente preguntando por el nombre de usuario y contraseña y cuando se rellena se manda una petición con una cabecera *Authorization*

```
Authorization Digest username="jose", realm="dominio", nonce="cIIDldTpBAA=9b0ce6b8eff03f5ef8b59da45a1ddfca0bc0c485", uri="/digest/", algo-  
rithm=MD5, response="814bc0d6644fa1202650e2c404460a21", qop=auth, nc=00000001,  
cnonce="3da69c14300e446b"
```

La información que se manda es *responde* que en este caso esta cifrada usando md5 y que se calcula de la siguiente manera:

- Se calcula el md5 del nombre de usuario, del dominio (realm) y la contraseña, la llamamos HA1.
- Se calcula el md5 del método de la petición (por ejemplo GET) y de la uri a la que estamos accediendo, la llamamos HA2.
- El resultado que se manda es el md5 de HA1, un número aleatorio (nonce), el contador de peticiones (nc), el qop y el HA2.

Una vez que lo recibe el servidor, puede hacer la misma operación y comprobar si la información que se ha enviado es válida, con lo que se permitiría el acceso.

Ejercicio:

1. Crea dos subdirectorios en el host virtual default que se llamen `grupo1` y `grupo2`. Crea varios usuarios con la utilidad `htdigest`, asignando a cada uno un dominio distinto (`domgrupo1` y `domgrupo2`). Configura los directorios para que al primero `grupo1` sólo puedan acceder los usuarios del dominio `domgrupo1`, y el directorio `grupo2` solo accedan los usuarios del dominio `domgrupo2`.

Ejercicio: Configuración de apache mediante archivo `.htaccess`

Un fichero `.htaccess` (hypertext access), también conocido como archivo de configuración distribuida, es un fichero especial, popularizado por el Servidor HTTP Apache que nos permite definir diferentes directivas de configuración para cada directorio (con sus respectivos subdirectorios) sin necesidad de editar el archivo de configuración principal de Apache.

Para permitir el uso de los ficheros .htaccess o restringir las directivas que se pueden aplicar usamos la directiva `AllowOverride`, que puede ir acompañada de una o varias opciones: `All`, `AuthConfig`, `FileInfo`, `Indexes`, `Limit`, ... Estudia para que sirve cada una de las opciones.

Ejercicios

Utiliza una cuenta de un servidor remoto para comprobar el uso de .htaccess. Crea un directorio dentro de `html_public` y crea un fichero .htaccess que nos permita:

1. Deshabilitar la opción de listar los ficheros en ese directorio.
2. Hacer que la página `entrada.html` se visualice por defecto.
3. Hacer que los ficheros `txt` no sean accesibles.
4. (Se debe hacer en local) Redireccionar a una página (por ejemplo la web del instituto) excepto a unas determinadas IP
5. Crear una lista de IPs prohibidas
6. Protege tu directorio y ficheros con autenticación básica
7. Crear una página personalizada para cada tipo de error
8. Crea una redirección permanente: cuando entremos en este directorio salte a `www.google.es`
9. Permitir la entrada desde un cliente en concreto (utilizando el nombre del host), si no se entra desde esa máquina, pedir autenticación.
10. Usar negociación de contenidos: tener dos páginas en distinto idioma y configurar en el .htaccess que idioma es el prioritario.

Ejercicio: Módulos en apache

Directorios web para cada usuario (`public_html`)

El módulo `userdir` permite que cada usuario del sistema tenga la posibilidad de tener un directorio (por defecto se llama `public_html`) donde alojar su página web.

Ejercicios

1. Activa el módulo y comprueba su funcionamiento.
2. Comprueba las opciones configuradas para los directorios `public_html`.
3. Cambia el nombre de directorio `public_html` por otro nombre.
4. Publica una página de un usuario, y accede a la misma.

Creación de un servidor WebDAV

WebDAV (“*Edición y versionado distribuidos sobre la web*”) es un protocolo para hacer que la `www` sea un medio legible y editable. Este protocolo proporciona funcionalidades para crear, cambiar y mover documentos en un servidor remoto (típicamente un servidor web). Esto se utiliza sobre todo para permitir la edición de los documentos que sirve un servidor web, pero puede también aplicarse a sistemas de almacenamiento generales basados en web, que pueden ser accedidos desde cualquier lugar. La mayoría de los sistemas operativos modernos proporcionan soporte para WebDAV, haciendo que los ficheros de un servidor WebDAV aparezcan como almacenados en un directorio local.

Configuración de un servidor WebDAV

Para crear un directorio en nuestro servidor Web que pueda ser accesible por medio de un cliente WebDAV debemos activar los módulos **dav** y **dav_fs**.

Lo primero es indicar el nombre de la base de datos de lock que se utilizará, mediante la directiva DAVLockDB. Es importante tener especial cuidado con esta directiva, ya que es frecuente fuente de errores.:

```
DavLockDB /tmp/DAVLockDB
```

Lo que indica la directiva no es ni el nombre de un archivo ni el de una carpeta, si no la parte inicial del nombre de un archivo. El módulo creará un archivo de nombre DAVLockDB.orig y otro de nombre DAVLockDB.xxxxxx dentro de la carpeta indicada, para lo cual es necesario que el usuario “Apache” tenga permisos de escritura en ella.

A continuación creamos una sección directory para el directorio que queremos acceder por WebDav y activar el modo WebDav con la directiva dav on. Además por seguridad se debe autenticar el acceso, por lo que quedaría parecido a esto:

```
DavLockDB /tmp/DavLock
<Directory /var/www/webdav>
    dav on
    Options Indexes FollowSymLinks MultiViews
    AllowOverride None
    Require all granted
    AuthType digest
    AuthUserFile "/etc/apache2/digest.txt"
    AuthName "Dominio"
    Require valid-user
</Directory>
```

Por último prueba un cliente WebDAV en Linux y otro en Windows y comprueba el funcionamiento.

Módulo rewrite

El módulo **rewrite** nos va a permitir acceder a una URL e internamente estar accediendo a otra. Ayudado por los ficheros .htaccess, el módulo rewrite nos va a ayudar a formar URL amigables que son más consideradas por los motores de búsquedas y mejor recordadas por los humanos. Por ejemplo estas URL:

```
www.dominio.com/articulos/muestra.php?id=23
www.dominio.com/pueblos/pueblo.php?nombre=torrelodones
```

Es mucho mejor escribirlas como:

```
www.dominio.com/articulos/23.php
www.dominio.com/pueblos/torrelodones.php
```

Ejemplo 1: Reescribir URL

Si tenemos el siguiente fichero php ([descargar](#)) llamado operacion.php, podríamos usarlo de la siguiente manera:

```
http://localhost/operacion.php?op=suma&op1=6&op2=8
```

Y si queremos reescribir la URL y que usemos en vez de php html, de esta forma:

```
http://localhost/operacion.html?op=suma&op1=6&op2=8
```

Para ello activamos el mod_rewrite, y escribimos un .htaccess de la siguiente manera:

```
Options FollowSymLinks
RewriteEngine On
RewriteRule ^operacion.html$ operacion.php
```

Ejemplo 2: Cambiar la extensión de los ficheros

Si queremos usar la extensión do en vez de html podríamos usar este `.htaccess`:

```
Options FollowSymLinks
RewriteEngine On
RewriteRule ^(.+)\.do$ %1.html [nc]
```

Esto puede ser penalizado por los motores de búsqueda ya que podemos acceder a la misma página con dos URL distintas, para solucionar esto podemos hacer una redirección:

```
RewriteRule ^(.+)\.do$ %1.html [r,nc]
```

Ejemplo 3: Crear URL amigables

Como habíamos visto anteriormente el fichero `operacion.php` se podía ejecutar de la siguiente manera:

```
http://localhost/operacion.php?op=suma&op1=6&op2=8
```

Creando una URL amigable podríamos llamar a este fichero de la siguiente manera:

```
http://localhost/suma/8/6
```

¿Cómo podemos conseguir esto?

Crea un `.htaccess` con el siguiente contenido:

```
Options FollowSymLinks
RewriteEngine On
RewriteBase /
RewriteRule ^([a-z]+)/([0-9]+)/([0-9+)$ operacion.php?op=$1&op1=$2&op2=$3
```

Ejemplo 4: Acortar URL

Supongamos que dentro de nuestro *DocumentRoot* tenemos una carpeta búsqueda con un fichero `buscar.php` (descargar). Este fichero me permite obtener la página de búsqueda de google con el parámetro dado, de esta forma:

```
http://localhost/busqueda/buscar.php?id=hola
```

Nos gustaría poder crear una URL más corta que haga lo mismo, escribiríamos en nuestro `.htaccess` un *RewriteRule* de la siguiente forma:

```
RewriteRule ^buscar busqueda/buscar.php
```

De esta forma accederíamos por medio de la URL:

```
http://localhost/buscar?id=hola
```

Ejercicio:

Siguiendo las técnicas anteriormente vistas, realiza una reescritura de URL para que pudiéramos realizar búsquedas con URL de la siguiente manera:

```
http://localhost/buscar/hola.html
```

Ejemplo 5: Uso del RewriteCond

La directiva RewriteCond nos permite especificar una condición que si se cumple se ejecuta la directiva RewriteRule posterior. Se pueden poner varias condiciones con RewriteCond, en este caso cuando se cumplen todas se ejecuta la directiva RewriteRule posterior.

Como vemos en la documentación podemos preguntar por varios parámetros , entre los que destacamos los siguientes:

- **%{HTTP_USER_AGENT}**: Información del cliente que accede. Por ejemplo, podemos mostrar una página distinta para cada navegador:

```
RewriteCond %{HTTP_USER_AGENT} ^Mozilla
RewriteRule ^/$ /index.max.html [L]
RewriteCond %{HTTP_USER_AGENT} ^Lynx
RewriteRule ^/$ /index.min.html [L]
RewriteRule ^/$ /index.html [L]
```

- **%{QUERY_STRING}**: Guarda la cadena de parámetros de una URL dinámica. Por ejemplo:

Teníamos un fichero index.php que recibía un parámetro lang, para traducir el mensaje de bienvenida:

```
http://localhost/index.php?lang=es
```

Actualmente hemos cambiado la forma de traducir, y se han creado distintos directorios para cada idioma y dentro un index.php con el mensaje traducido:

```
http://localhost/es/index.php
```

Sin embargo se quiere seguir utilizando la misma forma de traducir:

```
RewriteCond %{QUERY_STRING} lang=(.*)
RewriteRule ^index.php$ /%1/$1
```

- **%{REMOTE_ADDR}**: Dirección de destino. Por ejemplo puedo denegar el acceso a una dirección:

```
RewriteCond %{REMOTE_ADDR} 145.164.1.8
RewriteRule ^(.*)$ / [R,NC,L]
```

También podemos controlar la reescritura de URL según la hora y la fecha, para saber más lee este [artículo](#).

- **%{HTTP_REFERER}**: Guarda la URL que accede a nuestra página y %{REQUEST_URI} guarda la URI, URL sin nombre de dominio. Podemos evitar el Hot_Linking, o uso de recursos de tu servidor desde otra web. Por ejemplo, un caso muy común es usar imágenes alojadas en tu servidor puestas en otras web. Para ello podemos escribir el siguiente .htaccess:

```
RewriteCond %{HTTP_REFERER} !^$
RewriteCond %{HTTP_REFERER} !^http://(www\.)?dominio\.com/ [NC]
RewriteCond %{REQUEST_URI} !hotlink\.(gif|png) [NC]
RewriteRule .*\. (gif|jpg|png)$ http://www.dominio.com/image/hotlink.png [NC]
```

En el anterior ejemplo el primer RewriteCond permite la solicitud directa pero no desde otras páginas (referrer vacío). La siguiente línea indica que si el navegador ha enviado una cabecera Referrer y esta no contiene la palabra “dominio.com” se ejecutará el RewriteRule. La ultima instrucción RewriteCond indica que si en la url solicitada se encuentra el nombre de la imagen “hotlink” no se realizará el RewriteRule; esto se pone porque la imagen hotlink.png va a ser la que vamos a usar en RewriteRule y si no ponemos este RewriteCond también sería redirigida la solicitud a esta imagen. La última instrucción del ejemplo es el RewriteRule que indica que cualquier solicitud a una imagen desde otro referrer será reescrita en el servidor hacia la imagen hotlink.png y esta será la imagen que se vea en la web que te esté intentando robar la imagen.

Ejercicio:

Realiza un `.htaccess` para evitar el hot-linking. Puedes usar esta [imagen](#) para realizar el ejercicio.

Ejemplo 6: URL amigables con WordPress

Ejercicio: Instala wordpress en tu servidor con el módulo rewrite desactivado, comprueba que las URL no son amigables. Activa el módulo y a continuación configura el blog para que tenga URL amigables (Settings->Permalink).

En este bloque vamos a estudiar cómo funciona el protocolo DNS, y vamos a configurar distintos servidores DNS. El DNS se utiliza para distintos propósitos. Los más comunes son:

- Resolución de nombres: Dado el nombre completo de un host obtener su dirección IP.
- Resolución inversa de direcciones: Es el mecanismo inverso al anterior. Consiste en, dada una dirección IP, obtener el nombre asociado a la misma.
- Resolución de servidores de correo: Dado un nombre de dominio, obtener el servidor a través del cual debe realizarse la entrega del correo electrónico.

Los contenidos que vamos a estudiar en este bloque serán:

- Cómo funciona el protocolo DNS
- Realizar consultas a los servidores DNS
- Estudio de distintos servidores DNS: DNS Windows Server, dnsmasq, bind9
- Servidores DNS esclavos
- Configuración de subdominios, delegación de subdominios.
- Servidores DNS dinámicos

Índice

Enlaces interesantes

Protocolo DNS

- [¿Cómo funciona el DNS?](#)
- [Ficheros importantes en la resolución de nombres](#)

Dnsmasq

- [Fundamentos del servicio DNS](#)

- [HowToDNSmasq](#)

bind9

- [Servidor dns: Bind9](#)
- [Esquema: Servidor DNS \[jpg\]](#)
- [Servidores DNS esclavos \(1ª parte\)](#)
- [Servidores DNS esclavos \(2ª parte\)](#)
- [Configurar subdominios en bind9](#)
- [Servidor DNS dinámico](#)
- [Esquema: Proceso DNS dinámico \[jpg\]](#)
- [DNS dinámico \(Desde lo alto del cerro\)](#)

Ejercicio: Consultas DNS

dig

dig es una herramienta que permite hacer consultas a un servidor DNS desde la línea de comandos, es el sustituto de los programas nslookup y host. La sintaxis es:

```
dig [-t tipo de registro] [@servidor DNS] Consulta DNS
```

El tipo de registro por defecto es ADDRESS y el servidor DNS por defecto el definido en `/etc/resolv.conf`.

Nota: si no funciona el comando dig, instala el paquete `dnsutils` que lo incluye.

nslookup

[Como realizar consultas DNS con el nslookup de Windows](#)

Utilizando el comando dig/nslookup realiza las siguientes consultas al servidor DNS:

1. Preguntas a registros del tipo A: Obtén la dirección ip de los siguientes dominios:

```
www.gonzalonazareno.org
www.eltiempo.es
www.us.es
es.wikipedia.org
www.ubuntu.com
```

2. Preguntas a registros tipo NS: Obtén la dirección y los servidor DNS que corresponden a los siguientes dominios:

```
dominio raíz
com
org
es
us.es
wikipedia.org
ubuntu.com
```

3. Preguntas a registros MX: Obtén el nombre y la dirección del ordenador al que se mandan los correos que se envían a los siguientes dominios:

```
gonzalonazareno.org
us.es
wikipedia.org
ubuntu.com
```

4. ¿Qué tipo de registro es el que resuelve las siguientes direcciones:

```
www.josedomingo.org
informatica.gonzalonazareno.org
```

Indica el nombre canónico de las máquinas a las que corresponden.

5. Comprueba la dirección ip y el servidor DNS asociado a `dit.gonzalonazareno.org`, desde dentro de la intranet del ciclo formativo y desde fuera. ¿Cuáles son las diferencias? ¿A qué crees que es debido?
6. Pregunta sobre resolución inversa: En clase, ¿a qué nombre corresponde la dirección ip `172.22.0.1`?

Ejercicio: DNSmasq como DNS cache/forward en una red local

El paquete `dnsmasq` permite poner en marcha un servidor DNS de una forma muy sencilla. Simplemente instalando y arrancando el servicio `dnsmasq`, sin realizar ningún tipo de configuración adicional, nuestro PC se convertirá en un servidor caché DNS y además, resolverá los nombres que tengamos configurados en el archivo `/etc/hosts` de nuestro servidor. La resolución funcionará tanto en sentido directo como en sentido inverso, es decir, resolverá la IP dado un nombre de PC y el nombre del PC dada la IP.

Queremos instalar un servidor DNS local en nuestra intranet que nos permita gestionar los nombres de las máquinas y recursos de nuestra red, vamos a instalar el servidor DNS en el mismo ordenador que tenemos instalado el servidor web. Las características del servidor DNS que queremos instalar son las siguientes:

- El servidor web (IP que tenga el servidor linux) se llama `nombredelservidor.iesgn.com`
- Vamos a suponer que tenemos un servidor ftp que se llame `ftp.iesgn.com` y que está en `192.168.1.201` (esto es ficticio)
- Además queremos nombrar a otros clientes.
- Las páginas webs que hiciste en la práctica de apache (`www.iesgn.org`,...) tienen que ser accesibles desde los clientes.

Una vez instalado, el paquete, editamos el fichero `/etc/dnsmasq.conf` y modificamos las siguientes líneas:

- Descomentamos `strict-order` para que se realicen las peticiones DNS a los servidores que aparecen en el fichero `/etc/resolv.conf` en el orden en el aparecen.
- Incluimos las interfaces de red que deben aceptar peticiones DNS, descomentando la línea `interface` por ejemplo:
`interface=eth0`

Finalmente reiniciamos el servicio.

Advertencia:

1. Configura los clientes para que utilicen el servidor DNS que has instalado.
2. Realiza las consultas `dig/nslookup` desde los clientes preguntando por los siguientes:
 - Dirección de `nombredelservidor.iesgn.org`, `www.iesgn.org`, `ftp.iesgn.org`
 - La dirección IP de `www.josedomingo.org`

3. Comprueba que se puede entrar en las páginas webs

Ejercicio: Instalación y configuración del servidor bind9 en nuestra red local

Nota:

1. Desinstala el servidor dnsmasq que has instalado en la práctica anterior para que no tengas conflictos.
 2. Para hacer este ejercicio vamos a suponer que nuestro ordenadores están en la red 10.0.0.0/24, siendo nuestro servidor el 10.0.0.3, y los clientes 10.0.0.4 y 10.0.0.5. Adapta este direccionamiento a tu escenario.
-

Queremos instalar un servidor DNS local en nuestra intranet que nos permita gestionar los nombres de las máquinas y recursos de nuestra red, vamos a instalar el servidor DNS en nuestro servidor debian. Las características del servidor DNS que queremos instalar son las siguientes:

1. Vamos a crear una zona para el dominio: `iesgn.org`
2. Vamos a crear una zona de resolución inversa.
3. Vamos a tener los siguientes FQDN
 - El servidor DNS se llama `nombredelservidor.iesgn.org`
 - Vamos a suponer que tenemos un servidor para recibir los correos que se llame `correo.iesgn.org` y que está en 10.0.0.200 (esto es ficticio)
 - Vamos a suponer que tenemos un servidor ftp que se llame `ftp.iesgn.org` y que está en 10.0.0.201 (esto es ficticio)
 - Además queremos nombrar a varios clientes.
 - Suponemos que tenemos un servidor web con las páginas: `www.iesgn.org` y `departamentos.iesgn.org`

Advertencia:

1. Configura el servidor DNS con los registros A, CNAME, MX y NS necesarios, configura el SOA.
2. Configura los clientes para que su DNS sea el servidor Debian, debes indicar en la configuración de red del cliente como DNS primario la ip del servidor linux.
3. Realiza las consultas dig/neslookup desde los clientes preguntando por los siguientes:
 - Dirección de `nombredelservidor.iesgn.org`, “`www.iesgn.org`”, “`ftp.iesgn.org`”
 - El servidor DNS que tiene configurado la zona del dominio `iesgn.org`
 - El servidor de correo configurado para `iesgn.org`
 - La dirección IP de `www.josedomingo.org`
 - Un resolución inversa

Ejercicio: Instalación y configuración de un servidor DNS esclavo

Nota: Suponemos que tenemos instalado el servidor DNS del ejercicio anterior.

Actualmente tenemos instalado un servidor DNS que tiene autoridad sobre la zona `iesgn.org` y sobre la zona de resolución inversa correspondiente. Este servidor va a funcionar como **DNS maestro**. Vamos a instalar un nuevo servidor DNS que va a estar configurado como **DNS esclavo** del anterior, donde se van a ir copiando periódicamente las zonas del DNS maestro. Suponemos que el nombre del servidor DNS esclavo se va llamar `nombrededesclavo.iesgn.org`.

Modifica la configuración del servidor DNS actual para que funcione como maestro de las zonas que tiene definida. Instala y configura un nuevo servidor DNS para que funcione como DNS esclavo del anterior. Y realiza las siguientes comprobaciones:

Advertencia:

1. Entrega la configuración de las zonas del maestro y del esclavo.
2. Comprueba si las zonas definidas en el maestro tienen algún error con el comando adecuado.
3. Comprueba si la configuración de `named.conf` tiene algún error con el comando adecuado.
4. Reinicia los servidores y comprueba en los logs si hay algún error. **No olvides incrementar el número de serie en el registro SOA si has modificado la zona en el maestro.**
5. Muestra la salida del log donde se demuestra que se ha realizado la transferencia de zona.
6. Realiza una consulta con `dig` tanto al maestro como al esclavo para comprobar que las respuestas son autorizadas. ¿En qué te tienes que fijar?
7. Configura un cliente para que utilice los dos servidores como servidores DNS.
8. Solicita una copia completa de la zona desde el cliente ¿qué tiene que ocurrir?. Solicita una copia completa desde el esclavo ¿qué tiene que ocurrir?
9. Realiza una consulta desde el cliente y comprueba que servidor está respondiendo.
10. Posteriormente apaga el servidor maestro y vuelve a realizar una consulta desde el cliente ¿quién responde?

Ejercicio: Configuración de subdominios virtuales con bind9

Nota: Suponemos que tenemos instalado el servidor DNS del ejercicio anterior.

Tenemos un servidor DNS que gestiona la zona correspondiente al nombre de dominio `iesgn.org`, queremos configurar dicho servidor para crear el subdominio `informatica.iesgn.org`. Los nombres que vamos a tener en ese subdominio son los siguientes:

- `www.informatica.iesgn.org` corresponde a un sitio web que está en la dirección `10.0.0.100`.
- Vamos a suponer que tenemos un servidor ftp que se llame `ftp.informatica.iesgn.org` y que está en la misma máquina.

Advertencia:

1. Configura el servidor DNS para poder tener el subdominio virtual `informatica.iesgn.org`.
2. Realiza las consultas `dig/neslookup` desde los clientes preguntando por los siguientes:
 - Dirección de `www.informatica.iesgn.org`, `ftp.informatica.iesgn.org`
 - El servidor DNS que tiene configurado la zona del dominio `informatica.iesgn.org`. Comprueba que es el mismo que el de la zona `iesgn.org`.

Ejercicio: Delegación de subdominios con bind9

Nota: Suponemos que tenemos instalado el servidor DNS del ejercicio anterior.

Tenemos un servidor DNS que gestiona la zona correspondiente al nombre de dominio `iesgn.org`, en esta ocasión queremos delegar el subdominio `informatica.iesgn.org` para que lo gestione otro servidor DNS. Por lo tanto tenemos un escenario con dos servidores DNS:

- `nombredelservidor.iesgn.org`, es servidor DNS autorizado para la zona `iesgn.org` y tiene asignado la dirección `10.0.0.3`.
- `servidor_subdomio.informatica.iesgn.org`, es el servidor DNS para la zona `informatica.iesgn.org` y, suponemos que tiene asignado la dirección `10.0.0.50`.

Los nombres que vamos a tener en ese subdominio son los siguientes:

- `www.informatica.iesgn.org` corresponde a un sitio web que está en la dirección `10.0.0.100`.
- Vamos a suponer que tenemos un servidor ftp que se llame `ftp.informatica.iesgn.org` y que está en la misma máquina.
- Vamos a suponer que tenemos un servidor para recibir los correos que se llame `correo.informatica.iesgn.org` y que está en `10.0.0.51`.

Advertencia:

1. Configura el primer servidor DNS para poder tener el subdominio virtual `informatica.iesgn.org`.
2. Configura el segundo servidor DNS con los registros A, CNAME, MX y NS necesarios para el subdominio `informatica.iesgn.org`.
3. Realiza las consultas `dig/neslookup` desde los clientes preguntando por los siguientes:
 - Dirección de `www.informatica.iesgn.org`, `ftp.informatica.iesgn.org`
 - El servidor DNS que tiene configurado la zona del dominio `informatica.iesgn.org`. ¿Es el mismo que el servidor DNS con autoridad para la zona `iesgn.org`?
 - El servidor de correo configurado para `informatica.iesgn.org`

Ejercicio: Instalación y configuración de un servidor DNS dinámico

Nota: Suponemos que tenemos instalado el servidor DNS del ejercicio anterior.

Suponemos que actualmente tenemos instalado un servidor DNS caché que da servicio a los ordenadores de nuestra intranet y además actúa como servidor maestro (master) de un dominio DNS (`iesgn.org`), de forma que todos los equipos de la red local tengan un nombre DNS completo o FQHN (Full Qualified Host Name). Por otra parte, tenemos instalado en la misma máquina un servidor DHCP para que asigne direcciones IPv4 únicas a los equipos de la red local y les facilite el resto de parámetros necesarios para tengan conectividad y salida a Internet.

Las características del servidor DNS serán:

- El nombre del servidor DNS para la zona `iesgn.org` será `nombredelservidor.iesgn.org`.
- En un primer momento el único nombre que resuelve nuestro servidor será el suyo propio, con la dirección 192.168.2.1 (*Cambia el direccionamiento según tu escenario*).

Las características del servidor DHCP instalado serán:

- Tiempo de concesión: 1 mes
- Rango de direcciones: 192.168.2.100 - 192.168.2.150
- Puerta de enlace: 192.168.2.1
- Servidores DNS: 192.168.2.1

Advertencia:

1. Entrega la configuración en el servidor DNS necesaria para que tenga la funcionalidad de de DNS dinámico.
2. Muestra la configuración necesaria en el servidor DHCP para comunicarse con el DNS.
3. Configura un cliente de forma dinámica para que tome una configuración ofrecida por el servidor DHCP y comprueba que su nombre se ha incluido en la zona correspondiente en el servidor DNS: visualiza los logs que no informan de ello y realiza una consulta al servidor DNS preguntando por su nombre.
4. Fuerza a que un cliente cambie de DNS y comprueba que la modificación se ha comunicado al DNS.

Rendimiento en Servidores Web

El servidor web Apache 2.x puede ser configurado para manejar las peticiones de diferente forma, desde el punto de vista en que son creados y manejados los subprocesos necesarios que atienden a cada cliente conectado a este. En esta unidad vamos a explicar los MPM (Módulos de multiprocesamiento) que nos permiten configurar el servidor Web para gestionar las peticiones que llegan al servidor.

Vamos a estudiar también las distintas configuraciones que podemos realizar para que Apache sea capaz de servir páginas realizadas en PHP y vamos a estudiar las diferencias de rendimientos (uso de memoria y respuestas por segundos) que podemos obtener utilizando las distintas configuraciones: módulo php5, fastcgi. Por último estudiaremos diferentes aplicaciones que pueden mejorar el rendimiento de nuestro servidor: aceleradores PHP, memcache, varnish, ...

Para terminar la unidad vamos a hacer un estudio comparativo sobre el rendimiento entre distintos servidores web.

Enlaces interesantes

Módulos de multiprocesamiento en Apache 2.4

- [Módulos de multiprocesamiento \(MPMs\)](#)
- [Entendiendo los modos multiproceso de Apache](#)
- [Diferencias entre apache prefork, event y worker](#)

Midiendo el rendimiento de Apache

- [Gráficos del uso de memoria en Linux](#)
- [Graphing ApacheBench Results Using GnuPlot](#)
- [Scripts para calcular rendimientos](#)

Ejercicio: Módulos de Multiprocesamiento (MPMs)

Por defecto apache2 se configura con el MPM event, podemos ver el MPM que estamos utilizando con la siguiente instrucción:

```
# apachectl -V
...
Server MPM:      event
...
```

Para cambiar de MPM tenemos que desactivar el actual y activar el nuevo módulo:

```
# a2dismod mpm_event
# a2enmod mpm_prefork
# service apache2 restart

# apachectl -V
...
Server MPM:      prefork
...
```

Las directivas de configuración de los distintos MPM

En /etc/apache2/mods-available/mpm_prefork.conf:

Directivas de control de **prefork**:

```
StartServers      5
MinSpareServers   5
MaxSpareServers   10
MaxClients        150
MaxRequestsPerChild  0
```

En /etc/apache2/mods-available/mpm_worker.conf:

Directivas de control de **worker**:

```
StartServers      2
MinSpareThreads   25
MaxSpareThreads   75
ThreadLimit       64
ThreadsPerChild   25
MaxClients        150
MaxRequestsPerChild  0
```

En /etc/apache2/mods-available/mpm_event.conf:

Directivas de control de **event**:

```
StartServers      2
MinSpareThreads   25
MaxSpareThreads   75
ThreadLimit       64
ThreadsPerChild   25
MaxRequestWorkers 150
MaxConnectionsPerChild  0
```

El comando ab

La utilidad **ab** (Apache Benchmark) sirve para hacer pruebas de carga a un servidor apache. Es un programa que forma parte del paquete `apache2-utils`.

Veamos un ejemplo:

```
ab -n 1000 -c 5 -k http://localhost/
```

El anterior comando simula 5 usuarios al mismo tiempo ahciendo 1000 peticiones al servidor web del localhost. La opción `-k` (`keep_alive`) habilita la opción de no cerrar la sesión HTTP. Veamos los resultados de este comando:

```
Benchmarking localhost (be patient)
Completed 100 requests
Completed 200 requests
Completed 300 requests
Completed 400 requests
Completed 500 requests
Completed 600 requests
Completed 700 requests
Completed 800 requests
Completed 900 requests
Completed 1000 requests
Finished 1000 requests

Server Software:      Apache/2.2.16
Server Hostname:      localhost
Server Port:          80

Document Path:        /
Document Length:      42587 bytes

Concurrency Level:    5
Time taken for tests:  0.140 seconds
Complete requests:    1000
Failed requests:      0
Write errors:         0
Keep-Alive requests:  995
Total transferred:    42903740 bytes
HTML transferred:     42587000 bytes
Requests per second:  7124.44 [#/sec] (mean)
Time per request:     0.702 [ms] (mean)
Time per request:     0.140 [ms] (mean, across all concurrent requests)
Transfer rate:        298500.90 [Kbytes/sec] received

Connection Times (ms)
              min    mean[+/-sd] median    max
Connect:        0      0    0.1      0      3
Processing:      0      1    0.1      1      3
Waiting:        0      0    0.2      0      2
Total:          0      1    0.2      1      3

Percentage of the requests served within a certain time (ms)
 50%      1
 66%      1
 75%      1
 80%      1
```

90%	1
95%	1
98%	1
99%	1
100%	3 (longest request)

Ejecución de scripts php

Módulo php de Apache2

Módulo de apache: `libapache2-mod-php5`. Para más información: [Instalación de php en Debian](#). El fichero de configuración de php lo encontramos en `/etc/php5/apache2/php.ini`, para más información sobre la configuración: [How To Change Your PHP Settings on Ubuntu 14.04](#)

Si usas el módulo de PHP de apache2 no podrás utilizar los MPM *event* ni *worker*, se configurará de forma automática en **prefork**.

CGI

La ejecución del código php se hace por un proceso independiente del servidor web. Hay distintas alternativas: CGI, Fastcgi, PHP-FPM,..., puedes ver las diferencias en este [enlace](#).

- FastCGI es un protocolo para interconectar programas interactivos con un servidor web. FastCGI es una variación de la ya conocida Common Gateway Interface (CGI ó Interfaz Común de Entrada). El principal objetivo de FastCGI es reducir la carga asociada con el hecho de interconectar el servidor web y los programas Common Gateway Interface, permitiéndole a un servidor atender más peticiones a la vez.
- FPM (FastCGI Process Manager) es una implementación alternativa al PHP FastCGI con algunas características adicionales (la mayoría) útiles para sitios web con mucho tráfico.
- [Apache 2.4 with php5-fpm](#)

Aceleradores PHP

El objetivo principal de un acelerador PHP es guardar los scripts php ya compilando, obteniendo un mayor rendimiento en la respuesta del servidor. El acelerador se puede usar usando el módulo php de apache o usando fastcgi. Más información sobre [aceleradores PHP](#).

En la última versión de php (PHP 5.5) ya tenemos una cache de código instalada como módulo. En la documentación de PHP 5.5 podemos leer: “*La caché de códigos de operación de Zend Optimiser+ se ha añadido a PHP como la nueva extensión OPcache. OPcache mejora el rendimiento de PHP almacenando código de byte de un script precompilado en la memoria compartida, eliminando así la necesidad por parte de PHP de cargar y analizar scripts en cada petición.*”

Además tenemos otro módulo instalado en PHP 5.5, el módulo `apcu`, que permite a PHP guardar cierta información en memoria, por lo que también podemos acelerar el proceso de interpretación de código. Para más información sobre estos dos módulos de php puedes leer el artículo: [PHP 5.5 with Opcache and APCu](#)

Memcached

[Memcached](#) es un sistema distribuido de propósito general y que es muy usado en la actualidad por múltiples sitios web. Memcached es empleado para el almacenamiento en caché de datos u objetos en la memoria RAM, reduciendo así las necesidades de acceso a un origen de datos externo (como una base de datos o una API).

- [Manual de instalación de memcached](#)
- [Como utilizar Memcached con WordPress](#)

Varnish

Varnish es un acelerador HTTP que funciona como un proxy inverso. Se sitúa por delante del servidor web, cacheando la respuesta de dicho servidor web en memoria. La próxima vez que un visitante visite la misma URL, la página será servida desde Varnish en lugar de desde el servidor web, ahorrando recursos en el backend y permitiendo más conexiones simultáneas.

- [Presentación Madrid DevOps \(Varnish: funcionamiento, configuración y uso\)](#)
- <http://pabloroman.es/blog/2013/01/20/como-usar-varnish-para-acelerar-tu-sitio-web/>
- [How to: Varnish listen port 80 with systemd](#)
- <http://pabloroman.es/blog/2013/05/20/varnish-3-trucos-y-consejos/>
- <https://scalr-wiki.atlassian.net/wiki/display/docs/Install+Varnish+HTTP+Accelerator+with+WordPress>
- <http://kontsu.wordpress.com/2012/10/10/apache-2-performance-boost-with-varnish-yslow/>
- [Put Varnish on port 80](#)

Ejecución de aplicaciones web python en apache2

Crear una página web con Python (sin framework)

Aunque de forma general se utilizan distintos framework (el más popular es django) para el desarrollo de aplicaciones web con Python. En este apartado vamos a introducir los conceptos necesarios para crear una página web desarrollada con python, servida por un servidor web Apache, sin utilizar ningún framework. Para ello es necesario conocer el concepto de WSGI (Web Server Gateway Interface), que es una especificación de una interface simple y universal entre los servidores web y las aplicaciones web o frameworks desarrolladas con python.

- [Crear una página web con Python](#)

Crear una página web con Python (con framework)

Estudiando el apartado anterior podemos llegar a la conclusión que la utilización de un framework facilita mucho la implementación de aplicaciones web escritas en python. Tenemos a nuestra disposición muchos framework, el más conocido y usado es django, pero tenemos otros menos complejos como bottle o flask. La lista de framework python: [Web Frameworks for Python](#).

Pero antes de ver la configuración de apache2 para servir aplicaciones web python con distintos framework, vamos a estudiar la herramienta `virtualenv`.

Utilización de un entorno virtual de python

Hay que tener en cuenta que python es ampliamente utilizado por muchas aplicaciones GNU/Linux por lo que cuando se instalan paquetes de python más modernos o más antiguos de los que corresponden con la distribución GNU/Linux que se está utilizando, es muy recomendable hacerlo dentro de un entorno aislado que no afecte a la larga a las versiones correspondientes de los paquetes del sistema. Hay varias formas de hacer esto, en adelante vamos a explicar la forma de hacerlo con [Python Virtual Environments](#).

Instalamos los paquetes `python-virtualenv` y `python-dev` con un usuario privilegiado del sistema:

```
$ apt-get install python-dev python-virtualenv
```

Creamos un entorno virtual para instalar las dependencias que necesitamos con un usuario con privilegios normales:

```
$ virtualenv Python
Running virtualenv with interpreter /usr/bin/python2
New python executable in Python/bin/python2
Also creating executable in Python/bin/python
Installing setuptools, pip...done.
```

Lo que hemos hecho ha sido crear un entorno virtual de python en el directorio \$HOME/Python en el que se han instalado todos los paquetes de python necesarios para poder utilizar Python pip dentro de ese entorno y que los paquetes de python se instalen en dicho directorio por un usuario normal sin afectar a los paquetes de python del sistema.

Cada vez que vayamos a utilizar los paquetes de python de este entorno virtual o queramos instalar algún paquete nuevo, debemos activarlo mediante la instrucción:

```
usuario@oslo:~$ source ~/Python/bin/activate
(Python)usuario@oslo:~$
```

Como vemos se indica en el prompt con el prefijo (Python) que estamos dentro del entorno virtual. Y ahora podemos realizar en nuestro entorno virtual las instalaciones que queramos:

```
(Python)usuario@oslo:~$ pip install bottle requests
```

Cuando estemos dentro del entorno virtual de python y queramos salir basta con hacer:

```
(Python)usuario@oslo:~$ deactivate
usuario@oslo:~$
```

Ejemplo 1: Despliegue de una aplicación web desarrollada con bottle

En este primer ejemplo suponemos que hemos instalado bottle y las librerías python necesarias del repositorio oficial de debian. Por ejemplo tenemos la siguiente aplicación bottle:

```
from bottle import route, run

@route('/hello')
def hello():
    return "Hola mundo"

run(host='localhost', port=8080, debug=True)
```

Sabemos que si ejecutamos el anterior programa, se ejecutará un servidor web escuchando en el puerto 8080 y podremos acceder a comprobar la página. El servidor web que ejecuta python no está pensando para producción, sólo para desarrollos y pruebas. Por lo tanto si queremos que apache2 sirva esta aplicación tenemos que configurar apache2 (con el módulo WSGI) de la manera adecuada:

Necesitamos crear un fichero app.wsgi que facilita un objeto de tipo application. Veamos el siguiente fichero /var/www/miapp/app.wsgi:

```
import os

# Change working directory so relative paths (and template lookup) work again
os.chdir(os.path.dirname(__file__))

import bottle
# ... build or import your bottle application here ...
# Do NOT use bottle.run() with mod_wsgi
application = bottle.default_app()
```

Y la configuración del Apache2 sería la siguiente:

```
<VirtualHost *>
    ServerName www.example.com
    DocumentRoot /var/www

    WSGIDaemonProcess miapp user=www-data group=www-data processes=1 threads=5 python-
    ↪path=/var/www/html/miapp
    WSGIScriptAlias / /var/www/miapp/app.wsgi

    <Directory /var/www/miapp>
        WSGIProcessGroup miapp
        WSGIApplicationGroup %{GLOBAL}
        Require all granted
    </Directory>
</VirtualHost>
```

Advertencia: Ya no es necesario que la aplicación bottle ejecute un servidor web, por lo tanto es necesario eliminar la instrucción `run` del código.

Ejemplo 2: Despliegue de una aplicación web desarrollada con django

Hemos instalado el paquete django que tenemos en el repositorio oficial de debian jessie:

```
$ apt-get install python-django
$ django-admin version
1.7.11
```

La versión actual de django es la 1.10, en el siguiente ejemplo usaremos un entorno virtual para trabajar con la última versión de django.

Creamos una nueva aplicación django:

```
/var/www/html# django-admin startproject mysite
```

En el directorio `mysite` se han creado los ficheros y necesarios para desarrollar nuestra aplicación django, puedo ejecutar un servidor web para probar la aplicación de la siguiente manera:

```
/var/www/html/mysite# python manage.py runserver 0.0.0.0:8000

Django version 1.7.11, using settings 'mysite.settings'
Starting development server at http://0.0.0.0:8000/
Quit the server with CONTROL-C.
```

Después de construir nuestra aplicación, si queremos desplegarla sobre apache2, tenemos que utilizar el fichero `/var/www/html/mysite/mysite/wsgi.py` y configurar apache2 de la siguiente manera:

```
<VirtualHost *>
    ServerName www.example.com
    DocumentRoot /var/www/html/mysite
    WSGIDaemonProcess mysite user=www-data group=www-data processes=1 threads=5
    ↪python-path=/var/www/html/mysite
    WSGIScriptAlias / /var/www/html/mysite/mysite/wsgi.py

    <Directory /var/www/html/mysite>
        WSGIProcessGroup mysite
        WSGIApplicationGroup %{GLOBAL}
        Require all granted
    </Directory>
</VirtualHost>
```

```
</Directory>
</VirtualHost>
```

Ejemplo 3: Despliegue de una aplicación web desarrollada con django usando virtualenv

Como hemos dicho en el ejemplo anterior si queremos utilizar la última versión de django tenemos que instalarlo en un entorno virtual, para ello:

```
$ virtualenv django
$ source django/bin/activate
(django)debian@python:~$ pip install django
```

Crearíamos nuestra aplicación djngo y a la hora de modificar la configuración de apache2 la única línea que tendríamos que modificar sería la siguiente:

```
...
WSGIDaemonProcess mysite user=www-data group=www-data processes=1 threads=5 python-
↳path=/var/www/html/mysite:/home/debian/python/lib/python2.7/site-packages
...
```

teniendo en cuenta que el entorno virtual python lo hicimos en el directorio `/home/debian/python`.

Servidor FTP

FTP (siglas en inglés de File Transfer Protocol, ‘Protocolo de Transferencia de Archivos’) en informática, es un protocolo de red para la transferencia de archivos entre sistemas conectados a una red TCP (Transmission Control Protocol), basado en la arquitectura cliente-servidor. Desde un equipo cliente se puede conectar a un servidor para descargar archivos desde él o para enviarle archivos, independientemente del sistema operativo utilizado en cada equipo.

Los contenidos que vamos a estudiar del servidor FTP, nos van a pobilitar responder las siguientes preguntas:

1. ¿Qué signifcan las siglas FTP? ¿Para qué sirve dicho protocolo? ¿Qué inconveniente tiene su uso?
2. ¿Qué puertos utiliza el protocolo? ¿Para qué sirve cada uno?
3. ¿Qué tipos de acceso podemos configurar en un servidor FTP?
4. ¿Qué modos de conexión existen en el protocolo FTP? ¿En qué consiste cada uno y cuál es la diferencia?
5. Índica los tipos de transferncia que podemos hacer. ¿Qué tipo utilizarías si vas a transferir un fichero odt?
6. ¿Qué tipo de clientes FTP existen?
7. Si usas un cliente FTP de línea de comandos, que comando tendrías que ejecutar para conectarte al servidor ftp.gonzalonazareno.org y subir un fichero practica.pdf.

Indice

Enlaces interesantes

- Configuración básica de proFTPD en debian
- El protocolo FTP
- ProFTPD
- ProFTPD: Manual de instalación y configuración
- net2ftp: Cliente FTP basado en la web

- [Cómo instalar Proftpd con soporte MySQL](#)
- [Servidor ftp Windows 2008 Vídeo 1 Vídeo 2](#)

Ejercicio: Instalación de proFTPd y uso de clientes FTP

Instala el servidor proFTPd y comprueba su funcionamiento desde un cliente FTP gráfico: filezilla, y un cliente FTP de texto: ftp.

Guía de comandos FTP:

Comando y argumentos	Acción que realiza
open servidor	Inicia una conexión con un servidor FTP.
close o disconnect	Finaliza una conexión FTP sin cerrar el programa cliente.
bye o quit	Finaliza una conexión FTP y la sesión de trabajo con el
↪programa cliente.	
cd directorio	Cambia el directorio de trabajo en el servidor.
delete archivo	Borra un archivo en el servidor
mdelete patrón	Borra múltiples archivos basado en un patrón que se aplica al
↪nombre.	
dir	Muestra el contenido del directorio en el que estamos en el
↪servidor.	
get archivo	Obtiene un archivo
noop No Operation	Se le comunica al servidor que el cliente está en modo de no
↪operación, el servidor usualmente responde con un «ZZZ» y refresca el contador de	
↪tiempo inactivo del usuario.	
mget archivos	Obtiene múltiples archivos
hash	Activa la impresión de caracteres # a medida que se
↪transfieren archivos, a modo de barra de progreso.	
lcd directorio	Cambia el directorio de trabajo local.
ls	Muestra el contenido del directorio en el servidor.
prompt	Activa/desactiva la confirmación por parte del usuario de la
↪ejecución de comandos. Por ejemplo al borrar múltiples archivos.	
put archivo	Envía un archivo al directorio activo del servidor.
mput archivos	Envía múltiples archivos.
pwd	Muestra el directorio activo en el servidor.
rename archivo	Cambia el nombre a un archivo en el servidor.
rmdir directorio	Elimina un directorio en el servidor si ese directorio está
↪vacío.	
status	Muestra el estado actual de la conexión.
bin o binary	Activa el modo de transferencia binario.
ascii	Activa el modo de transferencia en modo texto ASCII.
!	Permite salir a línea de comandos temporalmente sin cortar la
↪conexión. Para volver, teclear exit en la línea de comandos.	
? nombre de comando	Muestra la información relativa al comando.
? o help	Muestra una lista de los comandos disponibles.
append archivo	Continúa una descarga que se ha cortado previamente.
bell	Activa/desactiva la reproducción de un sonido cuando ha
↪terminado cualquier proceso de transferencia de archivos.	
glob	Activa/desactiva la visualización de nombres largos de
↪nuestro PC.	
literal	Con esta orden se pueden ejecutar comandos del servidor de
↪forma remota. Para saber los disponibles se utiliza: literal help.	
mkdir	Crea el directorio indicado de forma remota.
quote	Hace la misma función que literal.
send archivo	Envía el archivo indicado al directorio activo del servidor.
user	Para cambiar nuestro nombre de usuario y contraseña sin
↪necesidad de salir de la sesión ftp.	

Modifica la configuración del servidor para que los usuarios sólo puedan entrar en su directorio “Documentos”.

Nota: Todos los accesos al servidor FTP lo vamos a hacer utilizando su nombre, por ejemplo ftp.iesgn.org, por lo tanto debes configurar el servidor BIND9 en el servidor para que todos los clientes conozcan este nombre.

Siguiendo las indicación de la documentación suministrada, configura el servidor proFTPd para crear un servidor FTP anónimo de sólo lectura.

Cuando termines, aunque no sea recomendable, configura el servidor proftpd para hacerlo anónimo y de lectura y escritura.

Servidor de correo electrónico

Índice

Enlaces interesantes

- [Cómo funciona el correo electrónico \(MTA, MDA, MUA\)](#)
- [Simple Mail Transfer Protocol](#)
- [Multipurpose Internet Mail Extensions](#)
- [Entender una cabecera de correo](#)

Postfix

- [Documentación: correo electrónico \(Postfix\)](#)
- [Documentación oficial postfix](#)

Instalación y configuración básica de postfix

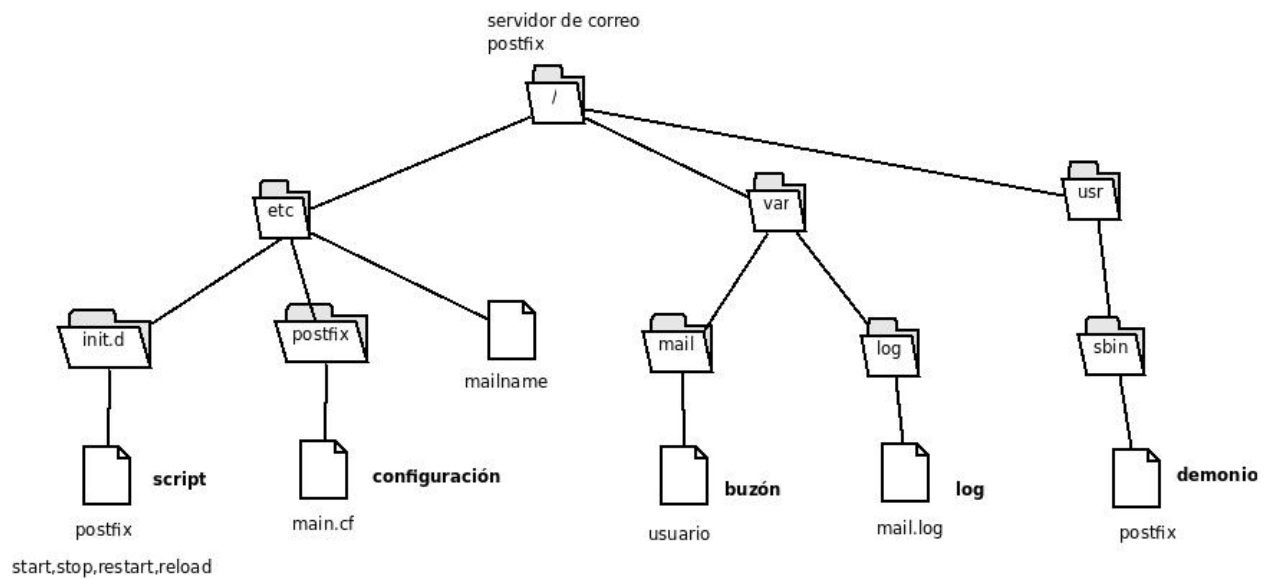
Directivas de configuración

Revisemos las directivas a tener en cuenta en nuestra configuración (`/etc/postfix/main.cf`):

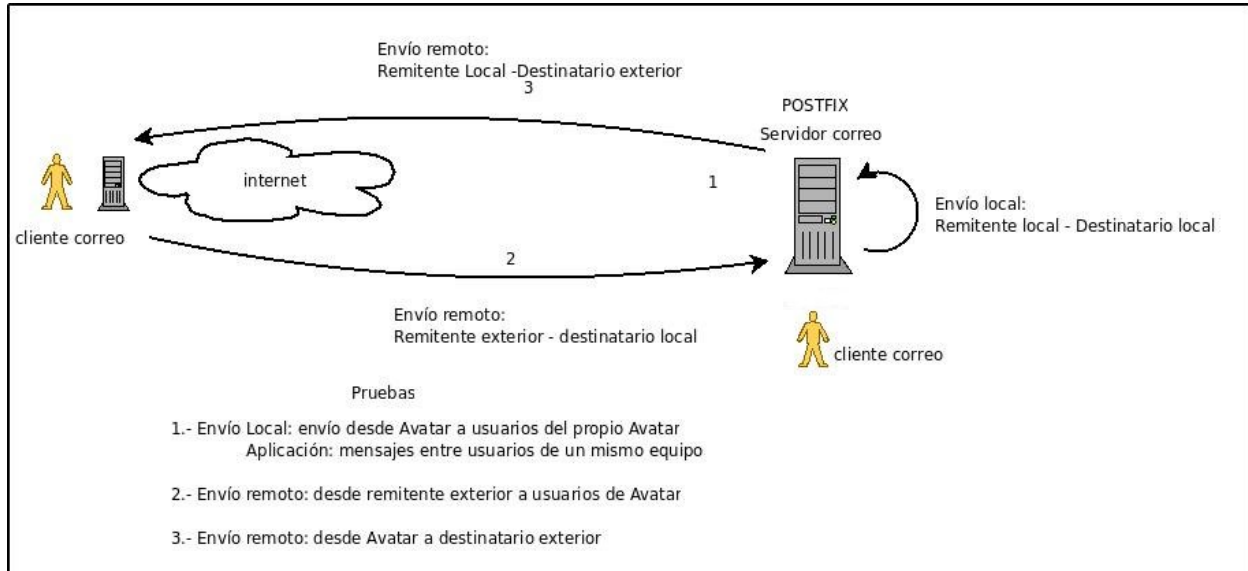
- `mydestination`: Observa que en la directiva `mydestination` se indican los dominios que serán propios del servidor de correo, es decir, el correo enviado a estos dominios está dirigido a usuarios del propio servidor. Si el usuario existe, el mensaje será almacenado, sino el servidor devolverá un mensaje de error.
- `relay_domains`: Con la directiva `relay_domains` indicamos los dominios que serán reenviados. Por lo tanto se permitirán el envío de correos a usuarios de estos dominios.
- `mynetworks`: Con `mynetworks` se indican las IPs desde las que pueden enviarse mensajes.

- `myorigin`: Por último, con `myorigin` se indica el dominio con el que el servidor enviará correo, el cual está configurado en `/etc/mailname`.

Localización de los ficheros del servidor postfix



Gestión de correo desde el servidor



Caso 1: Envío local, entre usuarios del mismo servidor

Esta situación podemos aprovecharla para el envío de correos entre usuarios de la máquina, disponiendo de correo interno. Con utilidades como `mail usuario` podemos enviar mensajes a usuarios del propio sistema.

Vamos a realizar el envío utilizando una conexión telnet, muy interesante tanto para entender los pasos que sigue el protocolo SMTP:

```

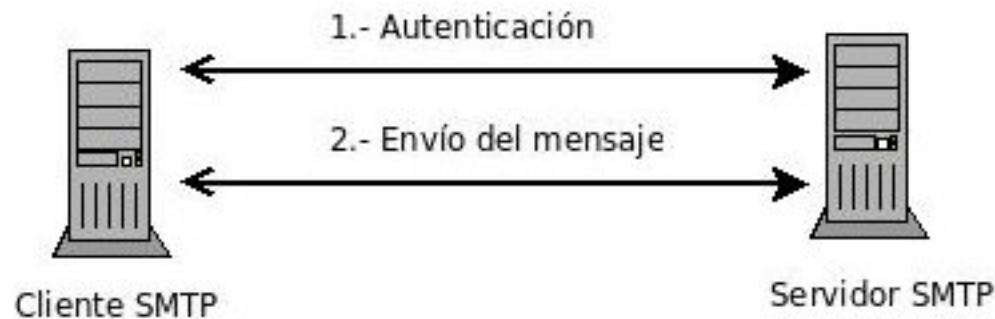
root@vostro:/home/jose# telnet localhost 25
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
220 mail2.josedomingo.org ESMTP Postfix (Debian/GNU)
HELO mail2.josedomingo.org
250 mail2.josedomingo.org
mail from: jose@josedomingo.org
250 2.1.0 Ok
rcpt to: usuario@josedomingo.org
250 2.1.5 Ok
data
354 End data with <CR><LF>.<CR><LF>
from: jose@josedomingo.org
to: usuario@josedomingo.org
subject: Prueba envio local

hola que tal

.
250 2.0.0 Ok: queued as DE3232C16A
quit
221 2.0.0 Bye
Connection closed by foreign host.

```

Pasemos a describir los pasos:



Fases en el envío de un mensaje de correo

- **Fase de autenticación** (puede haber otra si la sesión es cifrada). El cliente lanza los comandos siguientes para indicar qué quiere hacer:
 - EHLO o HELO “cadena presentándose el cliente ante el servidor”
 - MAIL FROM: “dirección de correo del remitente”
 - RCPT TO: “dirección de correo del destinatario”
- **Fase de envío del mensaje.** El cliente lanza la orden DATA y el servidor responde indicando “354 End data with <CR><LF>.”
 - From: “dirección del remitente” + SALTO DE LÍNEA (ie intro)
 - To: “dirección del destinatario” + SALTO DE LÍNEA (ie intro)
 - Cc: “dirección del destinatario” para tener una copia + SALTO DE LÍNEA (ie intro)

- Bcc: “dirección del destinatario” para tener una copia ciega
- Date: “fecha” + SALTO DE LÍNEA (ie intro)
- Subject: “asunto” + SALTO DE LÍNEA (ie intro)
- MIME-Versión: “valor de la versión de MIME usada” + SALTO DE LÍNEA (ie intro)
- Otras cabeceras + SALTO DE LÍNEA (ie intro)
- DOS SALTOS DE LÍNEAS
- Escribe el mensaje en varias líneas
- DOS SALTOS DE LÍNEAS, y en el segundo escribe “.” para finalizar el mensaje, y el cliente lo envía al servidor

Podemos comprobar el log `/var/log/mail.log` para comprobar que se ha mandado el mensaje:

```
Feb 6 18:10:05 vostro postfix/smtpd[3660]: DE3232C16A: client=localhost[127.0.0.1]
Feb 6 18:11:07 vostro postfix/cleanup[3907]: DE3232C16A: message-id=<20120206171005.
→DE3232C16A@mail2.josedomingo.org>
Feb 6 18:11:07 vostro postfix/qmgr[3531]: DE3232C16A: from=<jose@josedomingo.org>,
→size=400, nrcpt=1 (queue active)
Feb 6 18:11:08 vostro postfix/local[3908]: DE3232C16A: to=<usuario@josedomingo.org>,
→relay=local, delay=75, delays=74/0/0/1, dsn=2.0.0, status=sent (delivered to
→command: procmail -a "$EXTENSION")
Feb 6 18:11:08 vostro postfix/qmgr[3531]: DE3232C16A: removed
Feb 6 18:11:09 vostro postfix/smtpd[3660]: disconnect from localhost[127.0.0.1]
```

Y podemos leer el mensaje del usuario “usuario” con el programa mail:

```
Mail version 8.1.2 01/15/2001. Type ? for help.
"/var/mail/usuario": 1 message 1 new
>N 1 jose@josedomingo. Mon Feb 6 18:11 15/548 Prueba envio local
```

Caso 2: Envío de correos desde internet a usuarios del servidor

DESDE EL AULA

Vamos a tener un correo de la forma `usuario@dominio_de_cada_alumno`, para nuestro ejemplos pongamos `jose@josedom.gonzalonazareno.org`.

Tenemos que tener en cuenta los siguientes aspectos:

1. Si queremos recibir correos desde internet a nuestro servidor, todos nuestros de dominios tienen que apuntar a nuestra ip pública 80.59.1.152, sin embargo no hay que tocar el DNS de cdmon ya que tenemos un registro genérico que envía a 80.59.1.152 cualquier cosa de `.gonzalonazareno.org` que no tenga un registro tipo ADDRESS. Prueba a hacer un `dig loquesea.gonzalonazareno.org`.
2. Cuando se recibe un correo en esa dirección pública, lo recibe el servidor de correo que tenemos en babuino. Esto lo hace el cortafuegos de macaco.
3. **Tenemos que configurar el servidor de correos de babuino para que haga relay con los correos cuyo destino sean nuestros**
 - Añadimos en la directiva `realy_domains`, del servidor de correos de babuino, cada uno de los nombres de dominios a los que queremos reenviar los mensajes.
 - Para que conozca la IP de nuestro servidor de correo tendremos que crear un registro MX en nuestro servidor DNS para realizar la resolución.

4. Con la configuración que tenemos en el servidor de correo de nuestra máquina debe ser suficiente para recibir el correo. Recuerda mandar un mensaje a un usuario que exista en el servidor.

DESDE CASA

Usando un nombre de dominio

En este caso vamos a utilizar un nombre de dominio, en mi caso he usado `josedomingo.org`. Tenemos que tener en cuenta los siguientes aspectos:

1. Configura el servidor postfix en tu ordenador, teniendo en cuenta que en el fichero `/etc/mailname` este tu nombre de dominio.
2. Configura tu DNS para que el registro MX apunte a un nombre de ordenador que está definido como un registro A a tu dirección IP pública (si es dinámica y por cualquier motivo te cambia tendrás que actualizar el registro). En mi caso el registro MX de `josedomingo.org` está configurado de este modo:

```
; <<>> DiG 9.7.3 <<>> -t mx josedomingo.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 9147
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 4

;; QUESTION SECTION:
josedomingo.org.      IN      MX

;; ANSWER SECTION:
josedomingo.org.      900     IN      MX      10 mail2.josedomingo.org.

;; AUTHORITY SECTION:
josedomingo.org.      21600   IN      NS      ns1.cdmon.net.
josedomingo.org.      21600   IN      NS      ns3.cdmon.net.
josedomingo.org.      21600   IN      NS      ns2.cdmon.net.

;; ADDITIONAL SECTION:
mail2.josedomingo.org. 900     IN      A       46.234.130.137
```

3. En tu router haz DNAT para redirigir las peticiones por el puerto 25 al ordenador local que tiene instalado el servidor de correos.

Usando un nombre de máquina

En el caso de que utilices un servicio gratuito como `no-ip` o `dyndns`, estarás reservando un nombre de máquina y no de dominio, por ejemplo `avatar.dyndns.com`, en este caso no tiene sentido configurar el registro MX:

1. En este caso el nombre de máquina (`avatar.dyndns.com`) por ejemplo, apuntará a tu dirección IP pública.
2. En tu router haz DNAT para redirigir las peticiones por el puerto 25 al ordenador local que tiene instalado el servidor de correos.

Alias y redirecciones

Alias

Cuando se define un alias para un determinado usuario se redirige el correo que llegue a otro usuario de la misma máquina. Los alias de correo se utilizan principalmente para gestionar el correo de las “cuentas de administración” y se definen en el fichero `/etc/aliases`, que tiene el siguiente aspecto:

```
# /etc/aliases
mailer-daemon: postmaster
postmaster: root
nobody: root
hostmaster: root
usenet: root
news: root
webmaster: root
www: root
ftp: root
abuse: root
noc: root
security: root
www-data: root
logcheck: root
root: alberto, jose, raul
```

En este caso el correo que llega a los usuarios postmaster, webmaster, etc. se redirige a la cuenta de root, que a su vez se redirige a los usuarios reales alberto, jose y raul, que son los administradores del equipo.

Cada vez que se modifica el fichero `/etc/aliases` hay que ejecutar la instrucción `newaliases` para que los cambios tengan efecto.

Redirecciones

Una redirección se utiliza para enviar el correo que llegue a un usuario a una cuenta de correo exterior. Para usuarios reales las redirecciones se definen en el fichero `~/ .forward` y el formato de este fichero es simplemente un listado de cuentas de correo a las que se quiere redirigir el correo.

Caso 3: Envío de correo desde usuarios del servidor a correos de internet

DESDE EL AULA

En el caso de la configuración de nuestra red del insitituto sólo puede enviar correo el servidor de correo de babuino. Por lo tanto tenemos que configurar nuestro servidor para que utilice a babuino como relay para enviar nuestros correos, para ello, modificamos la siguiente directiva en el fichero de configuración:

```
relayhost = babuino.gonzalonazareno.org
```

DESDE CASA

Con la configuración que tienes actualmente podrías mandar correos al exterior.

Si en tu casa tienes un dirección IP dinámica seguramente gmail /hotmail/yahoo lo rechaza, por estar en una lista de bloqueo, al intentar enviar un correo nos salen registro de este tipo:

```
postfix/pickup[6804]: 09B0634680: uid=1000 from=<alberto>
postfix/cleanup[6810]: 09B0634680:message-id=<20081231154700.09B0634680@m\
ortadelo>
postfix/qmgr[6802]: 09B0634680: from=<alberto@avatar.dynalias.com>, size=30\
7, nrcpt=1 (queue active)
postfix/smtp[6812]: 09B0634680: to=<una@hotmail.com>,relay=mx2.hotmail.co\
m[65.54.244.40]:25, delay=1.3, delays=0.03/0.04/0.92/0.3, dsn=5.0.0, stat\
us=bounced (host mx2.hotmail.com[65.54.244.40] said: 550 DY-001 Mail reje\
cted by Windows Live Hotmail for policy reasons. We generally do not acce\
pt email from dynamic IP's as they are not typically used to deliver unau\
thenticated SMTP e-mail to an Internet mail server. http://www.spamhaus.o\
rg maintains lists of dynamic and residential IP addresses. If you are no\
t an email/network admin please contact your E-mail/Internet Service Prov\
ider for help. Email/network admins, please visit http://postmaster.live.\
com for email delivery information and support (in reply to MAIL FROM com\
mand))
postfix/smtp[6812]: 09B0634680: lost connection with mx2.hotmail.com[65.5\
4.244.40] while sending RCPT TO
```

Para solucionar este problema tenemos varias soluciones, que veremos en los siguientes apartados:

- Enviar correos a partir de un realy host autenticado
- Estudiar las listas de bloqueos
- Crear registros SPF

Soluciones al problema del spam

Como hemos visto en el caso 3 es posible que nuestra ip dinámica que tenemos en casa sea rechazada por los servidores de correo. En este apartado vamos a ver distintas técnicas que nos permiten luchar contra el spam y que nos pueden permitir que nuestro correo llegue a su destino y no sea rechazado. Veamos las posibles técnicas:

- **SMTP submission:** Para que no se permita conectar directamente a un servidor de correo de destino, una técnica para evitar a los spammer sería cambiar el puerto estándar del protocolo SMTP, normalmente se cambia al 587. Para más información leer el siguiente artículo: [Para que el correo llegue a buen puerto](#)
- **Enviar correo a través Relay autenticado:** En este caso podemos hacer relay uslizando un servidor de correos autenticado por ejemplo gmail: [Configurar postfix a través de un relay host autenticado \(Gmail\)](#)
- **SMTPd restrictions:** Podemos configurar nuestro servidor de correos para que filtre a los clientes que intentan usarlo por medio de algún parámetro del protocolo: HELO, MAIL FROM, RCPT TO,..., en el siguiente artículo puedes informarte sobre esta técnica y algunas otras que vamos a ver a continuación: [SMTPd restrictions](#), [SPF](#), [DKIM](#) and [greylisting](#)
- **Realtime blacklists (RBL):** La técnica más importante para luchar contra el spam son las listas negras. Podemos configurar nuestro servidor de correo para que consulte en las distintas listas que existen la dirección de ip de origen. Tenemos muchos sitios para ver si nuestra ip está en una lista negra:
 - [mxtoolbox](#)
 - [Spamhaus Block List](#)
 - [Cisco IronPort SenderBase Security Network](#)

- **Sender Policy Framework (SPF)**: Esta técnica consiste en publicar una serie de datos en un registro TXT del servidor DNS que haga que el servidor de correo donde llega el correo confíe en que el correo no es spam. Para más información lee el artículo: [Sender Policy Framework \(SPF\)](#)
- **DomainKeys Identified Mail (DKIM)**: Es un mecanismo que nos permite firmar un correo utilizando criptografía de clave pública. Para más información puede leer en la wikipedia: [DomainKeys Identified Mail](#)
- **Lista gris**: El servidor de correo si determina que el origen puede ser una amenaza, rechazada por defecto el correo, los servidores de correo legítimos volverán a enviar el correo transcurrido un tiempo, este correo será aceptado. Sin embargo un sistema que manda spam no suele reenviar el correo rechazado. Para más información puede leer en la wikipedia: [Lista gris](#)

Para terminar os dejo una presentación: [Univ. de Deusto: Seguridad en sistemas de correo electrónico](#)

Caso 4: Envío de correo electrónico usando nuestro servidor de correos

En este caso queremos que poder gestionar el envío y la recepción de correos desde los ordenadores de nuestra red local. Para ello debemos habilitar el envío de correo desde cliente de nuestra red. Para ello añade 192.168.1.0/24 (suponiendo que esta es nuestra red local) en la directiva mynetworks, quedando:

```
mynetworks = 127.0.0.0/8 [::ffff:127.0.0.0]/104 [::z1] 192.168.1.0/24
```

Es el momento de utilizar un cliente de correos (evolution, outlook, icedove, Thunderbird,...) en los clientes de nuestra red local para el envío de correos electrónicos. Para ello es necesario indicarle el nombre de nuestro servidor de correos, en mi caso smtp.josedom.gonzalonazareno.org, para ello desde los ordenadores de nuestra red local este nombre debe ser conocido, ya sea por resolución estática o usando un servidor DNS).

Veamos la configuración del cliente de correo Evolución:

Asistente de configuración de Evolution

Identidad

Por favor escriba debajo su nombre y dirección de correo-e. Los campos «opcionales» no hace falta que los rellene, a menos que quiera incluir esta información en el correo-e que envíe.

Información requerida

Nombre completo:

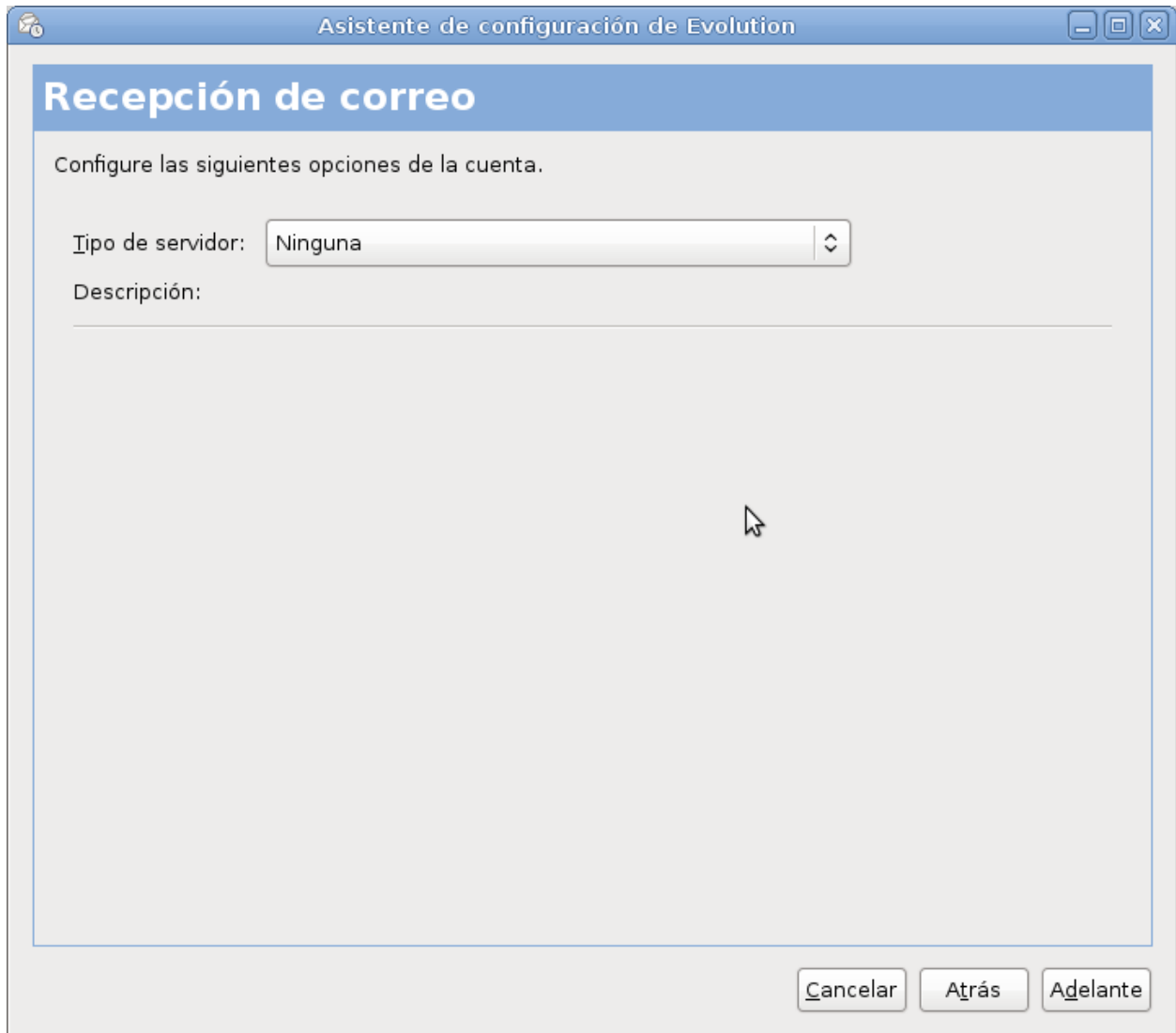
Dirección de correo-e:

Información opcional

☒ Hacer que ésta sea mi cuenta predeterminada

Responder a:

Organización:



Asistente de configuración de Evolution

Envío de correo

Por favor escriba debajo la información acerca de cómo enviará su correo. Si no está seguro, pregúntele a su administrador de sistemas o a su Proveedor de Servicios de Internet.

Tipo de servidor: SMTP

Descripción: Para entregar correo conectándose a un servidor de correo usando SMTP.

Configuración del servidor

Servidor: avatar.gonzalonazareno.org

☐ El servidor requiere autenticación

Seguridad

Usar conexión segura: Sin cifrado

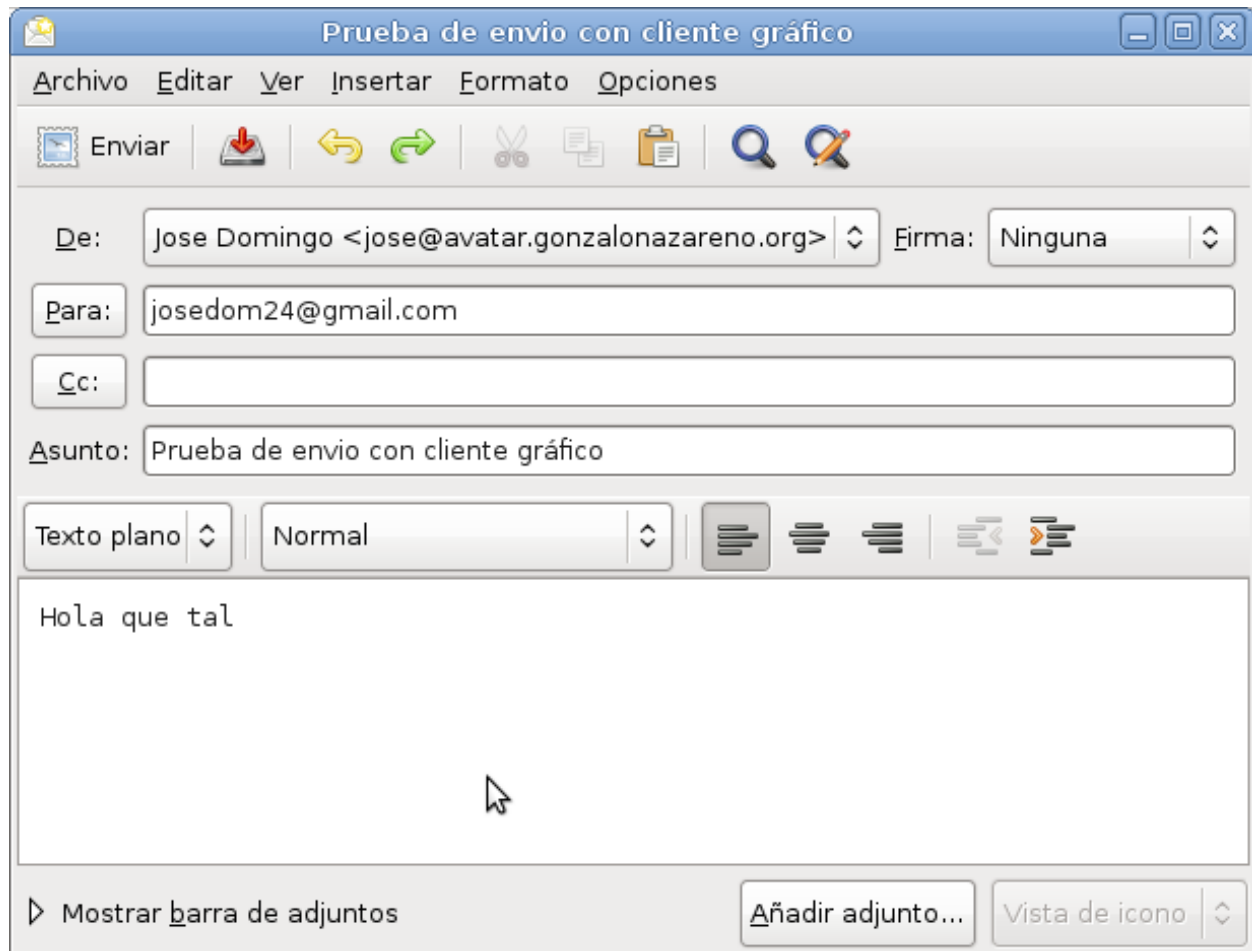
Autenticación

Tipo: PLAIN Comprobar tipos soportados

Usuario:

☐ Recordar contraseña

Cancelar Atrás Adelante



Caso 5: Recepción de correo electrónico usando nuestro servidor de correos

En este caso vamos a utilizar el [protocolo pop3](#) y el [protocolo imap](#) para obtener los correos electrónicos que hemos recibido en nuestro servidor.

Ante de ellos tenemos que estudiar los distintos tipos de buzón donde un servidor de correos puede guardar los mensajes:

- **Buzón mbox:** Todos los mensajes están en un fichero, es el tipo de buzón que hemos utilizado hasta ahora.
- **Buzón maildir:** Los mensajes se guardan en una carpeta. Es imprescindible para que funcione el protocolo imap.

Configurando un buzón Maildir

Lo primero que vamos a hacer es configurar postfix para que guarde los correos en un buzón del tipo Maildir, para ello añadimos y modificamos las siguientes directivas de configuración en el fichero de configuración:

```
home_mailbox = Maildir/
mailbox_command =
```

Después de reiniciar el servicio los correos se guardarán en un directorio Maildir que se creará en el directorio home de cada usuario.

Prorocolo pop3

En informática se utiliza el Post Office Protocol (POP3, Protocolo de la oficina de correo) en clientes locales de correo para obtener los mensajes de correo electrónico almacenados en un servidor remoto.

En nuestro caso el servidor pop3 que vamos a instalar se llama dovecot-pop3, para instalarlo, simplemente:

```
apt-get install dovecot-pop3d
```

La configuración que tenemos que realizar es la siguiente: cambiamos en `/etc/dovecot/conf.d/10-auth.conf`:

```
#disable_plaintext_auth = yes    ->    disable_plaintext_auth = no
```

Para que este habilitada la autenticación con contraseña en claro.

Cambiamos en el fichero `/etc/dovecot/conf.d/10-mail.conf`, donde se encuentra el buzón:

```
#mail_location = mbox:~/mail:INBOX=/var/mail/%u
mail_location = maildir:~/Maildir
```

Configuración del cliente de correo

Es la hora de configurar un cliente de correo desde nuestro cliente, en este ejemplo vamos a ver la configuración en evolution. Además tenemos que tener creados en el DNS de nuestro servidor los nombres `pop3.josedom.gonzalonazareno.org`.

Protocolo imap

Internet Message Access Protocol, o su acrónimo IMAP, es un protocolo de red de acceso a mensajes electrónicos almacenados en un servidor. Mediante IMAP se puede tener acceso al correo electrónico desde cualquier equipo que tenga una conexión a Internet. IMAP tiene varias ventajas sobre POP, que es el otro protocolo empleado para obtener correo desde un servidor. Por ejemplo, es posible especificar en IMAP carpetas del lado servidor. Por otro lado, es más complejo que POP ya que permite visualizar los mensajes de manera remota y no descargando los mensajes como lo hace POP.

Para instalar el servidor IMAP ejecutamos la siguiente instrucción:

```
apt-get install dovecot-imapd
```

Ya podríamos configurar nuestro cliente de correos para recibir correo utilizando el protocolo imap, para ello tendríamos creado un nombre en el servidor DNS, por ejemplo, `imap.josedom.gonzalonazareno.org`.

Nota: Podrías hacer la prueba de ver la diferencia entre los dos protocolo: como si usas pop3 los correos se borran del servidor (hay que configurar algunos clientes para que funcionen de esta forma) y cómo sin embargo al usar el servidor imap los correos no se borran del servidor.

En este caso también puedes instalar un cliente web de correos para leer los correos del servidor. Hay muchas: squirrelmail, round cube,...

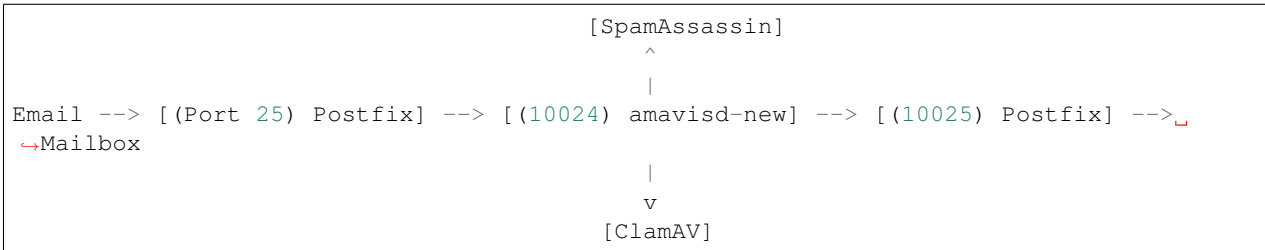
Antispam y antivirus en nuestro ervidor de correos

En este apartado vamos a realizar una introducción a la configuración de nuestro servidor de correos postfix para que sea capaz de determinar si los correos que le llegan tienen algún virus o son spam. Para ello vamos a utilizar tres

programas:

- **amavisd**: Es una interfaz entre el servidor de correos y el antivirus y antispam.
- **ClamAV**: Un antivirus.
- **SpamAssassin**: Un software para detectar el spam.

Es esquema de funcionamiento es el siguiente:



Os voy a dejar los pasos para realizar la instalación, este manual nace de mi experiencia, si veis algún cambio o hay algo incorrecto, por favor decírmelo. Empezamos con la instalación (vamos a instalar los todos los paquetes necesarios):

```
apt-get install amavisd-new spamassassin clamav clamav-daemon
```

Instalamos descompresores y utilidades:

```
apt-get install unrar-free zoo unzip bzip2 libnet-ph-perl libnet-snpp-perl libnet-  
telnet-perl nomarch lzop
```

amavisd

En el fichero `/etc/amavis/conf.d/15-content_filter_mode` descomentamos las siguientes líneas si vamos a activar el antivirus:

```
@bypass_virus_checks_maps = (  
    \%bypass_virus_checks, \@bypass_virus_checks_acl, \$bypass_virus_checks_re);
```

Y las siguientes si vamos a activar el antispam:

```
@bypass_spam_checks_maps = (  
    \%bypass_spam_checks, \@bypass_spam_checks_acl, \$bypass_spam_checks_re);
```

En el fichero `etc/amavis/conf.d/20-debian_defaults`, configuramos que hacer con los correos sospechosos, la configuración que viene por defecto es correcta pero podemos hacer algunos cambios, por ejemplo podemos dejar pasar el spam, aunque lo marquemos:

```
$final_spam_destiny = D_PASS;
```

Por último:

```
adduser clamav amavis  
systemctl restart amavis
```

Para que amavis pueda comunicar con postfix, añadimos a la configuración las siguientes líneas:

```
postconf -e 'content_filter = amavis:[127.0.0.1]:10024'  
postconf -e 'receive_override_options = no_address_mappings'
```

Añadimos las siguientes líneas al fichero `/etc/postfix/master.cf`:

```
amavis unix - - - - 2 smtp
    -o smtp_data_done_timeout=1200
    -o smtp_send_xforward_command=yes

127.0.0.1:10025 inet n - - - - smtpd
    -o content_filter=
    -o local_recipient_maps=
    -o relay_recipient_maps=
    -o smtpd_restriction_classes=
    -o smtpd_client_restrictions=
    -o smtpd_helo_restrictions=
    -o smtpd_sender_restrictions=
    -o smtpd_recipient_restrictions=permit_mynetworks,reject
    -o mynetworks=127.0.0.0/8
    -o strict_rfc821_envelopes=yes
    -o receive_override_options=no_unknown_recipient_checks,no_header_body_checks
    -o smtpd_bind_address=127.0.0.1
```

Y podemos ver los puertos por los que se está escuchando:

tcp	0	0	127.0.0.1:10024	0.0.0.0:*	LISTEN	18955/
↪ amavisd-new (
tcp	0	0	127.0.0.1:10025	0.0.0.0:*	LISTEN	19134/
↪ master						
tcp	0	0	127.0.0.1:783	0.0.0.0:*	LISTEN	16089/
↪ spamassassin.						
tcp	0	0	0.0.0.0:25	0.0.0.0:*	LISTEN	19134/
↪ master						

clamav

Actualizamos el antivirus:

```
freshclam
systemctl restart clamav-daemon
```

spamassassin

Lo activamos en el fichero `/etc/default/spamassassin`:

```
ENABLED=1
```

El fichero de configuración es `/etc/spamassassin/local.cf`

Para más información puedes leer los siguientes artículos:

- [howtoforge - Integrating amavisd-new Into Postfix For Spam- And Virus-Scanning](#)
- [Correo SPAM](#)
- [Como probar la efectividad y eficiencia del programa antivirus instalado](#)
- [Los test de spamassassin](#)

Servidor proxy/cache Squid

Indice

Enlaces interesantes

- [Presentación: Squid](#)
- [Proxy-cache Squid](#)
- [Squid, un proxy caché para GNU/Linux](#)
- [Manual de Instalación de Servidor Proxy Web con Squid](#)

Ejercicio: Instalación y configuración básica de squid

1. Instala el proxy-cache squid3 y comprueba la configuración básica del servidor:
 - Comprueba el puerto por el que está escuchando el servidor.
 - Asigna como memoria de la cache 300 Mb.
 - Define un tamaño de cache de 8 Gb.
 - Define el máximo tamaño de los elementos cacheados a 20 Mb.
 - Comprueba los ficheros de log y con que formato se guarda la información.
 - Modifica la configuración para que los errores salgan en español.
2. Una vez realizada la configuración básica, reinicia el servicio. Configura de manera manual un navegador y comienza a navegar utilizando el proxy.

¿Funciona? No, el servidor viene configurado por defecto para que sólo se puedan realizar conexiones desde localhost.

4. Para solucionar esto y poder acceder desde nuestra LAN tenemos que modificar una regla de acceso ACL:

- Descomentamos una regla ACL localnet y la ponemos de la siguiente manera:

```
acl localnet src 10.0.0.0/24
```

- Descomentamos la siguiente línea que nos permite el acceso a la red que hemos definido anteriormente:

```
http_access allow localnet
```

Reiniciamos, y comprobamos: miramos el fichero `/var/log/squid3/access.log`, y vemos las peticiones que se han guardado en la cache. Puedes encontrar información de cómo leer los logs en las siguientes direcciones:

- [linofee.org - Squid access.log](#)
- [Analizar los logs de access.log de squid](#)

Gestionando la cache de Squid (parte 1)

En este apartado vamos a estudiar cómo funciona la caché de squid:

- Gestión de la cache por mecanismos de validación
- Gestión de la cache por mecanismos de frescura
- Como evitar el cacheo de nuestro contenido

Gestionando la caché de Squid (parte 1)

Para obtener más información puedes ver los siguientes enlaces:

- [Tutorial Cache Web: cómo gestionar el cacheo de nuestros contenidos](#)
- [Two Important Differences between Firefox and IE Caching](#)
- [La memoria cache de Firefox: fallo o característica](#)
- [REDBot.org](#)
- [Tutorial Cache Web: cómo gestionar el cacheo de nuestros contenidos](#)
- [Optimización: Agregar Headers de Expiración](#)
- [Reduciendo el tráfico usando cache en Apache](#)

Ejercicio: Controlando la cache de squid

Partimos de la siguiente situación: tenemos instalado squid3 en un servidor, un navegador en la máquina real está configurado para usar el proxy (además en este navegador hemos desactivado la función de cache) y vamos acceder desde este navegador a una página html almacenada en un servidor web apache2 instalado en otro servidor.

Gestión de la cache por mecanismos de validación

En este primer punto vamos a acceder a la página html, y sólo va a entrar en juego el parámetro de cabecera `Last-Modified`. Una vez cacheada la página, si volvemos acceder a ella se preguntará al servidor si se ha modificado, el servidor responderá con la cabecera HTTP, y si la copia que tenemos es nueva se servirá directamente.

- En este caso al refrescar la página con F5 nos vamos encontrando en el fichero `access.log` con información del tipo `TCP_MEM_HIT` o `TCP_HIT`, es decir acierto en la cache.

- Si modificamos la página, el servidor cambiará el parámetro `Last-modified` y por tanto la copia que tenemos almacenada ya no será válida, por lo que nos bajaremos del servidor la página modificada y la volveremos almacenar. En este caso nos encontraremos en el fichero `access.log` una línea del tipo `TCP_REFRESH_MODIFIED`, indicando que la página accedida ha sido modificada.
- Si simulamos que se ha perdido la conexión con el servidor, parando el servicio `apache2`, aunque se intenta verificar con el servidor si la página ha sido modificada (`TCP_REFRESH_FAIL`), se seguirá sirviendo la copia que tenemos.

Gestión de la cache por mecanismos de frescura

Expires

El parámetro `Expires` de la cabecera de un mensaje HTTP indica cuando o cada cuanto tiempo la página guardada en cache no es válida y por lo tanto hay que bajarse otra del servidor. Para cambiar este parámetro de la cabecera vamos a usar el `mod_expire` de `Apache2`. Para ello nos aseguramos que está activo:

```
a2enmod expires
```

Y modificamos el fichero de configuración default de la siguiente manera:

```
<Directory /var/www/>
ExpiresActive On
ExpiresDefault "access plus 3 seconds"
...
```

En este caso estamos diciendo que todos los ficheros alojados en `/var/www` tienen una caducidad de 3 segundos. Cada tres segundos se va a obligar a descargarse la nueva copia del servidor.

Puedes comprobar el valor del parámetro usando el comando `HEAD`.

Si vamos recargando la página con `F5` nos daremos cuenta que cada 3 segundos se produce un `TCP_REFRESH_UNMODIFIED`, es decir se obliga a la descarga de la página, durante el tiempo intermedio se supone que la página es válida.

Como evitar el cacheo de nuestro contenido

Como estudiado en la teoría el contenido del parámetro `Control-cache` de la cabecera de un mensaje HTTP indica al proxy-cache que puede hacer con dicho contenido.

Para poder modificar este parámetro vamos a usar el `mod_headers` de `apache2`, para ello nos aseguramos que este activo:

```
a2enmod headers
```

A continuación podemos modificar el fichero de configuración default y añadir las siguientes líneas para evitar que se pueda cachear nuestro documento html:

```
<Directory /var/www/>
Header set Cache-Control "private, no-cache, no-store"
Header set Pragma "no-cache"
...
```

En este caso cuando vamos refrescando nuestra página en el navegador obtenemos un mensaje `TCP_MISS` en el fichero `access.log` indicando que nuestro documento no ha sido almacenado.

Gestionando la cache de Squid (parte 2)

En este apartado vamos a estudiar las posibilidades que ofrece squid a los administradores para forzar el “cacheo” de determinados elementos, incluso ignorando las cabeceras de control de caché.

Gestionando la caché de Squid (parte 2)

Para obtener más información puedes ver los siguientes enlaces:

- [refresh_pattern](#), controlando la cache

Gestionando el proxy de Squid

En este apartado vamos a ver como controlar el proxy por medio de ACL.

Para crear controles de acceso basados en el **origen** de la petición del cliente podemos usar los siguientes tipos de elementos de ACL:

- **src**: Dirección IP del cliente, puede ser una sola dirección lista o rango de direcciones IP, soporta el uso de mascararas de subred en formato CIDR
- **proxy_auth**: Autenticación de usuarios vía procesos externos
- **browser**: User-agent del navegador web que realiza la petición

Para crear controles de acceso basados en el **destino** de la petición encontramos los siguientes tipos de elementos de ACL:

- **dstdomain**: Este tipo define uno o más dominios destino solicitados por el cliente
- **url_regex**: Tipo con soporte de expresiones regulares para el URL solicitado por el cliente
- **urlpath_regex**: URL-path regular expression pattern matching, leaves out the protocol and hostname
- **port**: Este tipo define uno o más números de puerto destino solicitados por el cliente
- **method**: Este tipo define el método usado por el cliente para la petición HTTP (get, post, etc)

Para crear controles de acceso basados en el **tipo MIME** de la solicitud o respuesta de la petición podemos usar los siguientes tipos de elementos de ACL:

- **req_mime_type**: regular expression pattern matching on the request content-type header
- **rep_mime_type**: regular expression pattern matching on the reply (downloaded content) content-type header.

Además es posible crear controles de acceso basados en el **tiempo** en el que se realiza la petición, y usando procesos externos, por ejemplo para realizar autenticación basada en grupos, las ACLs que podemos usar son:

- **time**: hora del día, y día de la semana

Reglas de control de acceso

Los controles de acceso se realizan principalmente con la directiva `http_access` para peticiones HTTP y `http_reply_access` para las respuestas HTTP.

Las reglas `http_access` se usan para permitir o denegar el acceso a uno o más elementos de ACL, es decir, se podría evaluar tanto el origen: dirección IP, usuario, o el destino: dominio o URL de la petición, por mencionar algunos tipos. Squid tomará toda la información posible de las cabeceras de la petición HTTP.

El esquema más simple las reglas `http_access` para determinar el acceso a un a un elemento sería el siguiente:


```
http_access allow|deny acl
```

Squid evalúa las reglas en el orden en el que son escritas, es decir, de arriba hacia abajo. Si la primer regla no hace coincidencia con la petición, entonces el squid realizará una operación de tipo OR y evaluará los elementos de la siguiente regla de acceso:

```
http_access allow|deny acl
                OR
http_access allow|deny acl
                OR
...
```

Si en una regla de acceso hay más de un elemento de ACL, el sistema utiliza el operador AND para cada elemento de la regla, esto quiere decir, que todos los elementos de la ACL deben hacer coincidencia para que una acción se aplique:

```
http_access allow|deny acl AND acl AND ...
                OR
http_access allow|deny acl AND acl AND ...
                OR
...
http_access deny all
```

Para más información sobre la creación de ACL puedes mirar los siguientes enlaces:

- [Introducción a los esquemas de control de acceso en Squid](#)
- [Manual de Instalación de Servidor Proxy Web con Squid](#)

Autenticación por usuarios y grupos con Squid

En esta sección se describen los procedimientos para configurar el proxy Squid para autenticar usuarios usando diferentes métodos de autenticación.

Módulos de autenticación de usuarios Squid

- NCSA Usa un archivo de usuarios y contraseñas al estilo NCSA
- LDAP Usa el protocolo Lightweight Directory Access Protocol
- MSNT Usa un dominio de autenticación Windows NT
- PAM Usa los módulos de autenticación PAM
- SMB Usa un servidor SMB como Windows NT o Samba

Para autenticar usuarios hay que usar ACL de tipo `proxy_auth`.

Para más información:

- [Introducción a los esquemas de control de acceso en Squid](#)

Ejercicio: Configuración de los esquemas de control de acceso

Crea diferentes ACL para obtener los siguientes resultados:

Reglas de acceso basadas en direcciones IP

1. Sólo permitir el acceso a squid desde la 10.0.0.2.

2. Sólo permitir el acceso en un rango de direcciones (10.0.0.2-10.0.0.4)
3. Permitir el acceso a la red 10.0.0.0
4. No permitir el acceso a las direcciones IP que tenemos listadas en el fichero ip_sin_internet.txt
5. Permitir el acceso a la red 10.0.0.0, excluyendo al rango 10.0.0.10-10.0.0.20 Reglas de acceso basadas en puertos, dominios, URLs y tipos MIME
6. Comprueba el acl que viene en el fichero de configuración donde se indican los puertos permitidos.
7. Explica que conseguimos con las siguientes ACL:

```
acl Safe_ports port 80 21 443 563 70 210 1025-65535
acl SSL_ports port 443
acl CONNECT method CONNECT
...
...
...
http_access deny !Safe_ports
http_access deny CONNECT !SSL_ports
```

8. Evita que nos podamos conectar al dominio youtube.com y todos sus subdominios.
9. Evita que nos podamos conectar a los dominios listados en el fichero url_prohibidas.txt (lista negra)
10. Permite sólo la navegación a un conjuntos de nombres de dominios. (lista blanca)
11. Deniega el acceso a toda URL que contenga la palabra “informatica” (no tener en cuenta mayúsculas y minúsculas)
12. Deniega el acceso a toda URL que contenga las palabras “betis” o “sevilla”
13. Utilizando las ACL de tipo urlpath_regex, impide el acceso a ficheros pdf.
14. Usando el tipo MIME: evita el acceso a los ficheros de CSS (text/css), a los jpg (image/jpeg)
15. Evita la visualización de los contenidos flash

Otras reglas de acceso

16. Evita que se pueda navegar con el navegador Internet Explorer
17. Permite la navegación sólo los días entre semana de 8:00 de la mañana a las 14:00 de la tarde.

Ejercicio: Autenticación básica por usuarios y grupos con Squid

Configura squid para controlar el acceso de distintos tipos de usuarios:

- Los usuarios del perfil “alumnos” solo pueden navegar a páginas cuya URL contenga gonzalonazareno.
- Los usuarios del perfil “profesores” además pueden navegar por páginas que contengan la palabra “juntadeandalucia”.
- Los usuarios del perfil “equipo directivo” pueden navegar por cualquier página.

Ejercicio: Configuración de parámetros de proxy en clientes web

Configuración manual de parámetros de proxy

Ya hemos estudiado cómo se configuran los navegadores web gráficos. En este ejercicio configura un equipo (sin entorno gráfico) para que al navegar con lynx se este usando el proxy.

Configuración de parámetros de proxy usando script Proxy Auto-config

Configura el servidor web apache2 para que sirva un fichero proxy.pac. Crea dicho fichero con la configuración básica del proxy, e indica las siguientes exclusiones: no se usa el proxy cuando se accede al dominio gonzalonazareno.org y no se usa el proxy cuando se accede a localhost.

Configura el cliente web en la opción: Configuración automática de proxy

Configuración de parámetros de proxy usando la detección automática WPAD

1 Configura un servidor dhcp que permita que los clientes al recibir la configuración dinámica configuren los parámetros de acceso al proxy.

2. Configura el servidor bind9 que permita a los clientes con direccionamiento estático encontrar la configuración del proxy.

Puedes encontrar mucha más información en: [FindProxyForUrl](#)

Proxy transparente

En el esquema de red de clase:

```
iptables -t nat -A PREROUTING ! -s 10.0.0.1 -i virbr1 -p tcp --dport 80 -j DNAT --to-
→10.0.0.1:3128
```

En el esquema router+nat+proxy:

```
iptables -t nat -A PREROUTING -s 10.0.0.0/24 -i virbr1 -p tcp --dport 80 -j REDIRECT
→--to-port 3128
```

Otras herramientas: Dansguardians y Sarg

Dansguardians

DansGuardian es un software de filtro de contenido, diseñado para controlar el acceso a sitios web.

- [Página oficial de dansguardian](#)
- [Filtrando contenido \(squid+dansguardian+iptables\)](#)
- [urlblacklist.com](#)

Sarg

Es una herramienta para generar informes de acceso a un proxy

- [Página oficial de Sarge](#)
- [Instación y configuración de Sarg en Debian Jessie](#)

Balanceador de carga haproxy

Indice

Enlaces interesantes

- [maestrodeltweb](#): Balanceo de carga con haproxy
- [An Introduction to HAProxy and Load Balancing Concepts](#)
- [How To Use HAProxy to Set Up HTTP Load Balancing on an Ubuntu VPS](#)
- [Balanceador de carga en openstack](#)

CAPÍTULO 10

Prácticas

1ª Evaluación

Práctica: Servidor DHCP

Nota: (12 tareas - 20 puntos)(4 tareas obligatorias - 8 puntos)

Nota:

- Muestra al profesor: Tarea 4, Tarea 7
-

Teoría

Nota:

- **Tarea 1 (1 punto):** Lee el documento [Teoría: Servidor DHCP](#) y explica el funcionamiento del servidor DHCP resumido en este [gráfico](#).
-

Preparación del escenario

Crea un escenario usando Vagrant que defina las siguientes máquinas:

- Servidor: Tiene dos tarjetas de red: una pública y una privada que se conectan a la red local.
 - nodo_lan1: Un cliente conectado a la red local.
-

Servidor dhcp

Instala un servidor dhcp en el ordenador “servidor” que de servicio a los ordenadores de red local, teniendo en cuenta que el tiempo de concesión sea 12 horas y que la red local tiene el direccionamiento 192.168.100.0/24.

Advertencia:

- **Tarea 2 (1 punto)(Obligatorio):** Entrega el fichero Vagrantfile que define el escenario.
- **Tarea 3 (3 puntos)(Obligatorio):** Muestra al profesor el servidor DHCP funcionando. Muestra el fichero de configuración del servidor, la lista de concesiones, la modificación en la configuración que has hecho en el cliente para que tome la configuración de forma automática y muestra la salida del comando ifconfig.
- **Tarea 4 (2 puntos):** Muestra al profesor el servidor funcionando como router y NAT, de esta forma los clientes tendrán internet.
- **Tarea 5 (1 punto):** Realizar una captura, desde el cliente usando **tcpdump**, de los cuatro paquetes que corresponden a una concesión: DISCOVER, OFFER, REQUEST, ACK.

Funcionamiento del dhcp

Advertencia: Vamos a comprobar que ocurre con la configuración de los clientes en determinadas circunstancias, para ello vamos a poner un tiempo de concesión muy bajo. Muestra los resultados al profesor.

- **Tarea 6 (2 punto):** Los clientes toman una configuración, y a continuación apagamos el servidor dhcp. ¿qué ocurre con el cliente windows? ¿Y con el cliente linux?
- **Tarea 7 (2 punto)(Obligatorio):** Los clientes toman una configuración, y a continuación cambiamos la configuración del servidor dhcp (por ejemplo el rango). ¿qué ocurriría con un cliente windows? ¿Y con el cliente linux?

Reservas

Crea una reserva para el que el cliente tome siempre la dirección 192.168.100.100.

Advertencia:

- **Tarea 8 (2 puntos)(Obligatorio):** Indica las modificaciones realizadas en los ficheros de configuración y muestra al profesor una comprobación de que el cliente ha tomado esa dirección.

Uso de varios ámbitos

Modifica el escenario Vagrant para añadir una nueva red local y un nuevo nodo:

- **Servidor:** En el servidor hay que crear una nueva interfaz
- **nodo_lan2:** Un cliente conectado a la segunda red local.

Configura el servidor dhcp en el ordenador “servidor” para que de servicio a los ordenadores de la nueva red local, teniendo en cuenta que el tiempo de concesión sea 24 horas y que la red local tiene el direccionamiento 192.168.200.0/24.

Advertencia:

- **Tarea 9 (1 puntos):** Entrega el nuevo fichero Vagrantfile que define el escenario.
- **Tarea 10 (1 puntos):** Explica las modificaciones que has hecho en los distintos ficheros de configuración. Entrega las comprobaciones necesarias de que los dos ámbitos están funcionando.
- **Tarea 11 (1 puntos):** Realiza las modificaciones necesarias para que los cliente de la segunda red local tengan acceso a internet. Entrega las comprobaciones necesarias.

Script

Realiza un script en python que de información de las concesiones del servidor:

- `infodhcp.py -l`: Nos muestra todas las ips que están concedidas (las reservas también). Nos da información de los dos ámbitos.
- `infodhcp.py x.x.x.x`: Nos dice si esa ip está concedida y a que MAC corresponde.

Advertencia:

- **Tarea 12 (3 puntos):** Entrega la URL del repositorio GitHub donde tienes el programa

Práctica: Servidor WEB

Nota: (18 tareas - 25 puntos)(8 tareas obligatorias - 11 puntos)

Nota:

- Muestra al profesor: *Tarea 4, Tarea 10 y Tarea 16*
-

Advertencia:

- **Tarea 1 (1 punto)(Obligatorio):** Instala el servidor web Apache2 en una máquina. Modifica la página index.html que viene por defecto y accede a ella desde un navegador. Entrega una captura de pantalla accediendo a ella.

Virtual Hosting

Queremos que nuestro servidor web ofrezca dos sitios web, teniendo en cuenta lo siguiente:

1. Cada sitio web tendrá nombres distintos.
2. Cada sitio web compartirán la misma dirección IP y el mismo puerto (80).

Los dos sitios web tendrán las siguientes características:

- El nombre de dominio del primero será `www.iesgn.org`, su directorio base será `/srv/www/iesgn` y contendrá una página llamada `index.html`, donde sólo se verá una bienvenida a la página del Instituto Gonzalo Nazareno.
- En el segundo sitio vamos a crear una página donde se pondrán noticias por parte de los departamento, el nombre de este sitio será `departamentos.iesgn.org`, y su directorio base será `/srv/www/departamentos`. En este sitio sólo tendremos una página inicial `index.html`, dando la bienvenida a la página de los departamentos del instituto.

Advertencia:

- **Tarea 2 (3 punto)(Obligatorio):** Configura la resolución estática en los clientes y muestra al profesor el acceso a cada una de las páginas.

Mapeo de URL

Cambia la configuración del sitio web `www.iesgn.org` para que se comporte de la siguiente forma:

Advertencia:

- **Tarea 3 (1 punto)(Obligatorio):** Cuando se entre a la dirección `www.iesgn.org` se redireccionará automáticamente a `www.iesgn.org/principal`, donde se mostrará el mensaje de bienvenida. En el directorio **principal** no se permite ver la lista de los ficheros, no se permite que se siga los enlaces simbólicos y no se permite negociación de contenido. Muestra al profesor el funcionamiento.
- **Tarea 4 (1 punto)(Obligatorio):** Si accedes a la página `www.iesgn.org/principal/documentos` se visualizarán los documentos que hay en `/srv/doc`. Por lo tanto se permitirá el listado de fichero y el seguimiento de enlaces simbólicos siempre que sean a ficheros o directorios cuyo dueño sea el usuario. Muestra al profesor el funcionamiento.
- **Tarea 5 (1 punto):** En todo el host virtual se debe redefinir los mensajes de error de objeto no encontrado y no permitido. Para el ello se crearan dos ficheros html dentro del directorio error. Entrega las modificaciones necesarias en la configuración y una comprobación del buen funcionamiento.
- **Tarea 6 (1 punto):** Como el insitituto es bilingüe, en la URL `www.iesgn.org/principal/internacional`, debe existir dos mensajes de bienvenida: en inglés y en español, por lo tanto se debe permitir la negociación de contenidos. Realiza una prueba de funcionamiento, donde se demuestre que se ha accedido a la web desde un navegador con el español como idioma configurado, y que se accedido con el inglés. Entrega las modificaciones necesarias en la configuración y una comprobación del buen funcionamiento.

Estadísticas web

Vamos a instalar y configurar un analizador de logs de apache2 (**webalizer**) que nos permita generar estadísticas de acceso a nuestro servidor.

- La URL de la estadística sera `www.masterlan.com/estadistica`.
- El acceso a la estadística desde la red local está permitido, si hace desde fuera, por ejemplo desde el host, se requiere autenticación tipo digest (realizar este punto por medio de un fichero `.htaccess`)

Advertencia:

- **Tarea 7 (2 puntos):** Realiza la instalación de webalizer y muestra al profesor el funcionamiento del mismo teniendo en cuenta los requerimientos señalados.

Autenticación, Autorización, y Control de Acceso

Advertencia:

- **Tarea 8 (1 punto)(Obligatorio):** Crea un escenario en Vagrant que tenga un servidor con una red pública, y una privada, un cliente conectada a la red privada. Crea un host virtual que se acceda con el nombre `www.masterlan.com`. A la URL `www.masterlan.com/intranet` sólo se debe tener acceso desde el cliente de la red local, y no se pueda acceder desde la anfitriona por la red pública. A la URL `www.masterlan.com/internet`, sin embargo, sólo se debe tener acceso desde la anfitriona por la red pública, y no desde la red local. Muestra los resultados al profesor.
- **Tarea 9 (1 punto):** Autenticación básica. Limita el acceso a la URL `www.masterlan.com/secreto`. Comprueba las cabeceras de los mensajes HTTP que se intercambian entre el servidor y el cliente. ¿Cómo se manda la contraseña entre el cliente y el servidor?. Entrega una breve explicación del ejercicio.
- **Tarea 10 (1 punto)(Obligatorio):** Cómo hemos visto la autenticación básica no es segura, modifica la autenticación para que sea del tipo *digest*, y sólo sea accesible a los usuarios pertenecientes al grupo *directivos*. Comprueba las cabeceras de los mensajes HTTP que se intercambian entre el servidor y el cliente. ¿Cómo funciona esta autenticación? Muestra el funcionamiento al profesor y entrega una breve explicación del ejercicio.
- **Tarea 11 (1 punto):** Vamos a combinar el control de acceso (tarea 8) y la autenticación (tareas 9 y 10), y vamos a configurar el virtual host para que se comporte de la siguiente manera: el acceso a la URL `www.masterlan.com/secreto` se hace forma directa desde la intranet, desde la red pública te pide la autenticación. Muestra el resultado al profesor.
- **Tarea 12 (2 punto):** Utilizando el módulo `libapache2-mod-auth-pgsql`, configura un sitio virtual cuyo acceso sea autenticado mediante usuarios guardados en una base de datos PostgreSQL. Documenta la tarea.

Configuración con .htaccess

Date de alta en un **proveedor de hosting**. ¿Si necesitamos configurar el servidor web que han configurado los administradores del proveedor?, ¿qué podemos hacer? Explica la directiva `AllowOverride` de apache2. Utilizando archivos `.htaccess` realiza las siguientes configuraciones:

Advertencia:

- **Tarea 13 (1 punto)(Obligatorio):** Habilita el listado de ficheros en la URL `http://host.dominio/nas`.
- **Tarea 14 (1 punto):** Crea una redirección permanente: cuando entremos en `ttp://host.dominio/google` salte a `ww.google.es`.
- **Tarea 15 (1 punto):** Pedir autenticación para entrar en la URL `http://host.dominio/prohibido`.

Módulos

Advertencia:

- **Tarea 16 (2 puntos)(Obligatorio):** Módulo *userdir*: Activa y configura el módulo *userdir*, que permite que cada usuario del sistema tenga la posibilidad de tener un directorio (por defecto se llama `public_html`) donde alojar su página web. Publica una página de un usuario, y accede a la misma.
- **Tarea 17 (2 puntos):** Instalación de un servidor WebDAV que sea accesible desde la URL `www.masterlan.com/webdav`.
- **Tarea 18 (2 puntos):** Vamos a volver a nuestro hosting en CDMON, vamos a crear una carpeta php donde vamos a tener un fichero `index.php` con el siguiente contenido:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.
→org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Conversor de Monedas</title>
</head>

<body>
<form action="index.php" method="get">
    <input type="text" size="30" name="monto" /><br/>
    <select name="pais">
        <option name="Dolar">Dolar</option>
        <option name="Libra">Libra</option>
        <option name="Yen">Yen</option>
    </select>
    <input type="submit" value="convertir" />
</form>
<?php
    // averiguamos si se ingresó un monto
    if (isset($_GET['monto'])) {
        define ("cantidad", $_GET['monto']);
    } else {
        define ("cantidad", 0);
    }
    if($_GET){
        // definimos los paises
        $tasacambios = array ("Libra"=>0.86,"Dolar"=>1.34,"Yen"=>103.56);
        // imprimimos el monto ingresado
        echo "<b>".cantidad." euros</b><br/> ".$_GET["pais"]." = ".cantidad*
→$tasacambios[$_GET["pais"]]." <br><br>";
        // por cada pais imprimimos el cambio
    }

    ?>

</body>
</html>
```

Prueba la página utilizando parámetros en la URL (parámetros GET), por ejemplo: `http://nombre_página/php/index.php?monto=100&pais=Libra`

Configura mediante un fichero `.htaccess`, la posibilidad de acceder a la URL **`http://nombre_página/php/moneda/cantidad`**, donde moneda indica el nombre de la moneda a la que queremos convertir (Dolar,Libra,Yen) y cantidad indica los euros que queremos convertir.

Práctica: Estudio de servidores Webs

Nota:

- (6 tareas - 30 puntos)
 - (En la 1ª evaluación: 20 y en la 2ª evaluación 10)
-

Los servidores webs que se van a estudiar en el presente proyecto son: nginx, cherokee, lighttpd, IIS.

Objetivo del trabajo

realizar un trabajo de investigación por grupos. Cada grupo estudiará 3 servidores web: **nginx, lighttpd y IIS**, para lograr los siguientes objetivos:

- Realizar la instalación del servidor web estudiado.
- Aprender los conceptos y realizar la configuración básica del servidor: virtual hosting, autenticación, ejecución de scripts.
- Elaborar un manual que explique la experiencia realizada.
- Realizar una comparación de consumo de memoria y rendimiento entre el servidor web estudiado y el servidor Apache 2.

Estructura del trabajo

El manual que debes realizar tendrá al menos los siguientes apartados para cada uno de los servidores web:

1. Introducción al servidor web
2. Instalación del servidor web
3. Configuración básica del servidor web
4. Configurando Virtual Hosting
5. Autenticación y Control de Acceso
6. Ejecución de scripts PHP
7. Estudio comparativo con el servidor web Apache 2
8. Conclusiones
9. Bibliografía, páginas webs de referencia,...

Si se ve oportuno el índice puede cambiar, pero los contenidos anteriores deben aparecer. Se valorará positivamente la ampliación de contenidos.

En el apartado **Estudio comparativo con el servidor web Apache 2**, se debe hacer un estudio comparado entre el servidor estudiado y el servidor Apache 2. Se realizarán las siguientes medidas:

- Memoria consumida por el servidor
- Rendimiento (número de peticiones servidas) de cada servidor

Se realizará el estudio sobre la página principal de un CMS Wordpress que estará instalado en ambos servidores en dos máquinas virtuales con las mismas características. Cada grupo seleccionará las distintas pruebas de benchmark que se puedan realizar a los dos servidores para observar las diferencias.

Nota: El apartado 7 (tarea 6) se realizará durante la 2ª evaluación.

Advertencia: Se evaluará los siguientes apartados, para cada servidor web:

- **Tarea 1 (2 puntos):** La estética del documento (teniendo en cuenta las normas de presentación) y los puntos de introducción, instalación, conclusiones y bibliografías.

Para cada uno de los servidores:

- **Tarea 2 (2 puntos):** El punto **Instalación del servidor web**.
- **Tarea 3 (4 puntos):** El punto **Configurando Virtual Hosting**.
- **Tarea 4 (5 puntos):** El punto **Autenticación y Control de Acceso**.
- **Tarea 5 (7 puntos):** El punto **Ejecución de scripts PHP**.
- **Tarea 6 (10 puntos):** El punto **Estudio comparativo con el servidor web Apache 2**, se valorará también las conclusiones desprendida del estudio.

Nota: Normas sobre la redacción del trabajo

El trabajo se realizará siguiendo los siguientes puntos:

- Se debe entregar un documento pdf.
- El documento tendrá una portada, donde aparezca el título, el autor y la fecha en que se realizó el trabajo.
- Se deberá de realizar un índice de contenidos, donde se vea el número de página donde comienza cada apartado.
- Cada página estará numerada, en el pie de página y centrado.
- Se utilizará con carácter general un tipografía de letra Roman 12pt o similar.
- Los títulos irán numerados y se pondrán con tamaño 14pt y negrita.
- Los códigos bash o contenidos de ficheros se pondrán con tipografía mono espaciada, por ejemplo, Courier, o similar.
- No se abusará del uso de las capturas de pantalla, sólo se pondrán aquellas capturas que sean necesarias para justificar que se ha realizado los aspectos fundamentales del trabajo.

Práctica: Servidor DNS

Nota: (8 tareas - 25 puntos)(3 tareas obligatorias - 10 puntos)

Nota:

- Muestra al profesor: *Tarea 2, Tarea 6 y Tarea 7*
-

Escenario

1. En nuestra red local tenemos un **servidor Web** que sirve dos páginas web: *www.iesgn.org*, *departamentos.iesgn.org*
2. Vamos a instalar en nuestra red local un servidor DNS (lo puedes instalar en el mismo equipo que tiene el servidor web)
3. Voy a suponer en este documento que el nombre del servidor DNS va a ser *pandora.iesgn.org*. Si quieres puedes utilizar otro nombre.

Servidor DNSmasq

Instala el servidor dns **dnsmasq** en *pandora.iesgn.org* y configúralo para que los clientes puedan conocer los nombres necesarios.

Advertencia:

- **Tarea 1 (2 punto)(Obligatorio):** Modifica los clientes para que utilicen el nuevo servidor dns. Realiza una consulta a *www.iesgn.org*, y a *www.josedomingo.org*. Realiza una prueba de funcionamiento para comprobar que el servidor dnsmasq funciona como cache dns. Muestra el fichero *hosts* del cliente para demostrar que no estás utilizando resolución estática. ¿Se puede realizar resolución inversa?. Documenta la tarea en redmine.

Servidor bind9

Desinstala el servidor **dnsmasq** del ejercicio anterior e instala un servidor dns **bind9**. Las características del servidor DNS que queremos instalar son las siguientes:

- El servidor DNS se llama *pandora.iesgn.org* y por supuesto, va a ser el servidor con autoridad para la zona *iesgn.org*.
- Vamos a suponer que tenemos un servidor para recibir los correos que se llame *correo.iesgn.org* y que está en la dirección *x.x.x.200* (esto es ficticio).
- Vamos a suponer que tenemos un servidor ftp que se llame *ftp.iesgn.org* y que está en *x.x.x.201* (esto es ficticio)
- Además queremos nombrar a los clientes.
- También hay que nombrar a los virtual hosts de apache: *www.iesgn.org* y *departamentos.iesgn.org*
- Se tienen que configurar la zona de resolución inversa.

Advertencia:

- **Tarea 2 (4 puntos)(Obligatorio):** Realiza la instalación y configuración del servidor bind9 con las características anteriormente señaladas. Entrega las zonas que has definido.
- **Tarea 3 (4 puntos)(Obligatorio):** Realiza las consultas **dig/nslookup** desde los clientes preguntando por los siguientes:
 - Dirección de *pandora.iesgn.org*, *www.iesgn.org*, *ftp.iesgn.org*
 - El servidor DNS con autoridad sobre la zona del dominio *iesgn.org*
 - El servidor de correo configurado para *iesgn.org*

- La dirección IP de `www.josedomingo.org`
- Un resolución inversa

Servidor DNS esclavo

El servidor DNS actual funciona como **DNS maestro**. Vamos a instalar un nuevo servidor DNS que va a estar configurado como **DNS esclavo** del anterior, donde se van a ir copiando periódicamente las zonas del DNS maestro. Suponemos que el nombre del servidor DNS esclavo se va llamar `afrodita.iesgn.org`.

Advertencia:

- **Tarea 4 (3 puntos): Realiza la instalación del servidor DNS esclavo. Documenta los siguientes apartados:**
 - Entrega la configuración de las zonas del maestro y del esclavo.
 - Comprueba si las zonas definidas en el maestro tienen algún error con el comando adecuado.
 - Comprueba si la configuración de `named.conf` tiene algún error con el comando adecuado.
 - Reinicia los servidores y comprueba en los logs si hay algún error. **No olvides incrementar el número de serie en el registro SOA si has modificado la zona en el maestro.**
 - Muestra la salida del log donde se demuestra que se ha realizado la transferencia de zona.
- **Tarea 5 (3 puntos): Documenta los siguientes apartados:**
 - Configura un cliente para que utilice los dos servidores como servidores DNS.
 - Realiza una consulta con `dig` tanto al maestro como al esclavo para comprobar que las respuestas son autorizadas. ¿En qué te tienes que fijar?
 - Solicita una copia completa de la zona desde el cliente ¿qué tiene que ocurrir?. Solicita una copia completa desde el esclavo ¿qué tiene que ocurrir?
- **Tarea 6 (2 puntos): Muestra al profesor el funcionamiento del DNS esclavo:**
 - Realiza una consulta desde el cliente y comprueba que servidor está respondiendo.
 - Posteriormente apaga el servidor maestro y vuelve a realizar una consulta desde el cliente ¿quién responde?

Delegación de dominios

Tenemos un servidor DNS que gestiona la zona correspondiente al nombre de dominio `iesgn.org`, en esta ocasión queremos delegar el subdominio `informatica.iesgn.org` para que lo gestione otro servidor DNS. Por lo tanto tenemos un escenario con dos servidores DNS:

- `pandora.iesgn.org`, es servidor DNS autorizado para la zona `iesgn.org`.
- `ns.informatica.iesgn.org`, es el servidor DNS para la zona `informatica.iesgn.org` y, está instalado en otra máquina.

Los nombres que vamos a tener en ese subdominio son los siguientes:

- `www.informatica.iesgn.org` corresponde a un sitio web que está alojado en el servidor web del departamento de informática.

- Vamos a suponer que tenemos un servidor ftp que se llame `ftp.informatica.iesgn.org` y que está en la misma máquina.
- Vamos a suponer que tenemos un servidor para recibir los correos que se llame `correo.informatica.iesgn.org`.

Advertencia:

- **Tarea 7 (4 puntos):** Realiza la instalación y configuración del nuevo servidor dns con las características anteriormente señaladas. Muestra el resultado al profesor.
- **Tarea 8 (3 puntos):** Realiza las consultas `dig/neslookup` desde los clientes preguntando por los siguientes:
 - Dirección de `www.informatica.iesgn.org`, `ftp.informatica.iesgn.org`
 - El servidor DNS que tiene configurado la zona del dominio `informatica.iesgn.org`. ¿Es el mismo que el servidor DNS con autoridad para la zona `iesgn.org`?
 - El servidor de correo configurado para `informatica.iesgn.org`

Práctica: Configuración de un servidor Windows Server

Nota: (8 tareas - 20 puntos)

Esquema de red

Debes configurar en un entorno virtual usando KVM el siguiente esquema de red:

Como podemos ver, vamos a tener tres máquinas: un servidor Windows Server y dos clientes: uno Linux y otro Windows.

Advertencia:

- **Tarea 1 (2 puntos):** Configuración de la red virtual en KVM. Indica el nombre que le has puesto al ordenador servidor. Y por último, indica la configuración de red del servidor y de algún cliente.
- **Tarea 2 (2 puntos):** Explica la configuración del servidor para que funcione como router y nat.

Servidor DHCP

Los ordenadores clientes de nuestra LAN obtienen su configuración de red ofrecidas por dicho servidor, que tiene las siguientes características:

- Rango: 192.168.1.3-192.168.1.254
- Máscara de red: 255.255.255.0
- Puerta de enlace: La ip del router
- DNS: Según el que te convenga para hacer las pruebas

Crea una reserva para el que uno de los clientes tome siempre una dirección fija.

Advertencia:

- **Tarea 3 (3 puntos):** Muestra al profesor el servidor DHCP funcionando. Muestra el fichero de configuración del servidor, la lista de concesiones, la modificación en la configuración que has hecho en el cliente para que tome la configuración de forma automática y muestra la salida del comando `ifconfig`.
- **Tarea 4 (2 puntos):** Indica las modificaciones realizadas en el servidor y muestra al profesor una comprobación de que el cliente ha tomado esa dirección.

Servidor DNS

Lo primero que tienes que hacer es determinar un nombre de dominio que va a ser utilizado en nuestro sistema. (En esta documentación voy a utilizar el nombre `example.com`). El servidor DNS ofrece el servicio de resolución de nombres para los ordenadores de nuestra red local. Debes tener en cuenta los siguientes puntos:

- Cuando tengas funcionando el servidor DNS, tendrás que modificar el servidor DHCP para que los clientes usen el nuevo servidor DNS.
- Piensa el nombre que tiene el servidor. El servidor DNS debe poder resolver los siguientes nombres: `nombredelservidor.example.com`, `www.example.com`, `informatica.example.com`. El primero es el nombre del servidor, los dos siguientes son dos páginas webs que el servidor va a servir.
- Debes implementar la zona inversa del servidor.

Advertencia:

- **Tarea 5 (3 puntos):** Realiza la instalación y configuración del servidor DNS con las características anteriormente señaladas. Indica el cambio que hay que hacer en el servidor dhcp para que el sistema funcione de manera adecuada. Entrega una captura de pantalla donde se vea las zonas que has definido. Muestra el resultado al profesor.
- **Tarea 6 (3 puntos):** Realiza las consultas `dig/nslookup` desde los clientes preguntando por los siguientes:
 - Dirección de `www.example.com`
 - El servidor DNS con autoridad sobre la zona del dominio
 - La dirección IP de `www.josedomingo.org`
 - Una resolución inversa

Servidor WEB

El servidor tiene instalado un servidor Web IIS, que sirve dos virtual host con páginas web estáticas.

- `www.example.com`: Página principal del centro.
- `informatica.example.com`: Página del departamento de informática
- La página `www.example.com`, posee un directorio `/privado`, que para acceder a el es necesario autenticarse.

Advertencia:

- **Tarea 7 (3 puntos):** Configura el servidor web para servir las dos páginas. Busca plantillas html+css como contenido para las dos páginas, modificandolas un poco para que parezcan más reales. Muestra las páginas al profesor.

- **Tarea 8 (2 puntos):** Autenticación de la carpeta /privado: Sólo se pueden autenticar los usuarios del sistemas que pertenezcan al grupo profesores. Muestra el funcionamiento al profesor.

Práctica: Configuración de servidores GNU/Linux

Nota: (12 tareas - 20 puntos)(6 tareas obligatorias - 7 puntos)

Nota: Muestra al profesor: *tarea 6, tarea 8 y tarea 11*

Advertencia: Objetivo

El objetivo de esta práctica es montar una infraestructura de servicios que se mantenga en el tiempo y que nos sirva para montar servicios y aplicaciones en los distintos módulos durante el curso. Esta práctica la tenéis que realizar en la infraestructura de máquinas que hemos creado en el cloud para todas los módulos. En cualquier momento del curso los servicios que instalemos en esta práctica deben estar funcionando de manera adecuada.

- Servidor1: homer (Ubuntu)
- Servidor2: barney (Debian)
- Servidor3: lisa (CentOs)

Advertencia: Ejemplo de nombres, suponiendo que mi nombre de dominio va a ser `josedom.gonzalonazareno.org`:

Los nombres de los equipos van a ser:

- `homer.josedom.gonzalonazareno.org`
- `barney.josedom.gonzalonazareno.org`
- `lisa.josedom.gonzalonazareno.org`
- El servidor DNS va a estar instalado en `barney.josedom.gonzalonazareno.org`
- El servidor web va a estar instalado en `lisa.josedom.gonzalonazareno.org`, y vamos a tener dos páginas webs:
 - `www.josedom.gonzalonazareno.org`
 - `informatica.josedom.gonzalonazareno.org`
- El servidor de base de datos va a estar instalado en `silvestre.josedom.gonzalonazareno.org`

Servidor DNS

Vamos a instalar un servidor dns que nos permita gestionar la resolución directa e inversa de nuestros nombres. Cada alumno va a poseer un servidor dns con autoridad sobre un subdominio de nuestro dominio principal `gonzalonazareno.org`, que se llamará `tu_nombre.gonzalonazareno.org`.

Nota: Indica al profesor el nombre de tu dominio para que pueda realizar la delegación en el servidor DNS principal *papion*.

Instalación del servidor DNS

El servidor DNS se va a instalar en el servidor 2. Y en un primer momento se configurará de la siguiente manera:

- El servidor DNS se llama `servidor2.tu_nombre.gonzalonazareno.org` y va a ser el servidor con autoridad para la zona `tu_nombre.gonzalonazareno.org`.
- El servidor debe resolver el nombre de los tres servidores.
- Se debe configurar las zonas de resolución inversa.

Nota:

- Debes determinar si la resolución directa se hace con dirección ip fijas o flotantes del cloud dependiendo del servicio que se este prestando.
 - Debes considerar la posibilidad de hacer dos zonas de resolución inversa para resolver ip fijas o flotantes del cloud.
 - Debes modificar la configuración del servidor DHCP del cloud para que mande a los servidores el nuevo nombre de dominio.
-

Advertencia:

- **Tarea 1 (1 puntos):** Comprueba que los servidores tienen configurados el nuevo nombre de dominio de forma adecuada después de volver a renovar la concesión del servidor DHCP. Documenta el contenido del fichero en el que se puede comprobar este punto (ejecuta el comando `hostname -f` y muestra el fichero `/etc/resolv.conf`).
- **Tarea 2 (2 puntos)(Obligatorio):** Entrega el resultado de las siguientes consultas :
 - El servidor DNS con autoridad sobre la zona del dominio `tu_nombre.gonzalonazareno.org`
 - La dirección IP de los tres servidores
 - Un resolución inversa de IP fija, y otra resolución inversa de IP flotante.

Nos gustaría poder dar de alta nuevos nombres en el servidor DNS. Para ello vas a crear un script en python que nos permita añadir o borrar registros en las zonas de nuestro servidor.

El script se debe llamar `gestionDNS.py` y recibe cuatro parámetros:

- `-a` o `-b`: Si recibe `-a` añadirá un nuevo nombre, si recibe `-b` borrará el nombre que ha recibido.
- `-dir` o `-alias`, si recibe `-dir` va a crear un registro tipo A, si recibe `-alias` va a crear un registro CNAME
- El nombre de la máquina para añadir o borrar
- El nombre del alias o la dirección ip: Si has usado la opción `-dir` recibirá una ip y si has usado `-alias` recibirá el nombre de la máquina a la que le vamos a hacer el alias. Si has utilizado `-b` no tendrá este parámetro.

Ejemplos

```
gestionDNS.py -a -dir smtp 192.168.4.1
```

Crearé el registro -> smtp A 192.168.4.1

```
gestionDNS.py -a -alias correo smtp
```

Crearé el registro -> correo CNAME smtp

```
gestionDNS.py -b correo
```

Borraré el último registro

Todos los registros creados o borrados pertenecen a las zonas `tu_nombre.gonzalonazareno.org`. Se debe modificar la zona inversa en los casos necesarios. El script debe reiniciar el servidor `bind9`.

Advertencia:

- **Tarea 3 (3 puntos):** Entrega el repositorio github donde has desarrollado el script y realiza un ejemplo al profesor.

Servidor Web

En nuestro servidor3 vamos a instalar un servidor Web apache2 con las siguientes características.

Advertencia:

- **Tarea 4 (1 punto)(Obligatorio):** Nuestro servidor va a tener dos hosts virtuales: `www.tu_nombre.gonzalonazareno.org` y `informatica.tu_nombre.gonzalonazareno.org`. Explica los pasos fundamentales para realizar los dos virtual hosts.
- **Tarea 5 (1 punto):** Comenta los cambios en el servidor DNS para de dar de alta los dos nuevos nombres.
- **Tarea 6 (1 punto)(Obligatorio):** La página `www.tu_nombre.gonzalonazareno.org` va a ser la página principal, busca una plantilla html, modifícala un poco y desplégala en el primer virtual host. Muestrasela al profesor.
- **Tarea 7 (1 punto)(Obligatorio):** Por seguridad, en la página `www.tu_nombre.gonzalonazareno.org`, no se permite que se sigan enlaces simbólicos, no se permite negociación de contenidos, no se permite visualizar la lista de ficheros y no se permite usar ficheros `.htaccess`. Entrega la modificaciones en la configuración necesarias.
- **Tarea 8 (1 punto)(Obligatorio):** La página `informatica.tu_nombre.gonzalonazareno.org` es una página relacionada con el mundo de la informática, busca una plantilla html, modifícala un poco y desplégala en el primer virtual host. La página se guardará en un directorio llamado `plataforma`. Por lo tanto si accedemos a `informatica.example.com` se deberá redirigir automáticamente a `informatica.example.com/plataforma`. Muestra el resultado al profesor.
- **Tarea 9 (3 puntos):** Para llevar una estadística de visitas y accesos instala la aplicación `awstats` en el servidor. Configura el cron para que la estadística se vaya actualizando. Debes realizar dos estadísticas, una para cada host virtual.
- **Tarea 10 (3 puntos):** En el directorio `/srv/isos` tenemos una colección de imágenes isos, queremos acceder a ella en la dirección `informatica.tu_nombre.gonzalonazareno.org/isos`. Esta dirección debe ser sólo accesible desde la intranet, si accedemos desde fuera tenemos que autenticarnos (digest) con un usuario.

Servidor de Base de Datos

En nuestro servidor1 vamos a instalar un servidor de base de datos `mysql`.

Advertencia:

- **Tarea 11 (1 punto)(Obligatorio):** Configura el servidor para que sea accesible por los equipos de la red local. Muestra al profesor una conexión a la base de datos desde el servidor3.
- **Tarea 12 (2 puntos):** Instala en el servidor3 la aplicación phpmyadmin que nos permite gestionar las bases de datos de nuestro servidor. Esta aplicación sólo será accesible desde la URL `www.tu_nombre.gonzalonazareno.org/basededatos`. Muestra el acceso al profesor.

Práctica: Mediciones de rendimiento de Apache2 sirviendo páginas dinámicas

Nota: (9 tareas - 30 puntos)(6 tareas obligatorias - 12 puntos)

Las seis primeras tareas hay que hacerla individualmente, las tres últimas se tienen que hacer como un trabajo grupal.

Vamos a comparar el uso de memoria y el rendimiento de Apache2 sirviendo páginas dinámicas programadas con PHP, en concreto vamos a servir una CMS Wordpress.

Advertencia: Por defecto PHP y Wordpress tienen una limitación del uso de memoria que puede falsear los resultados que podemos obtener, por lo tanto es muy importante que cambiemos estos límites. Podemos poner el valor por defecto de uso de memoria en 256 Mb, incluso aumentarlo si vemos que es necesario. Puede seguir este [enlace](#) para realizar la operación, teniendo en cuenta que el fichero de configuración de PHP es distinto según el método de ejecución que estemos usando: módulo php de apache2, o FPM.

Vamos a instalar un wordpress un servidor y vamos a realizar las siguientes pruebas:

- 500 peticiones y 2 concurrentes
- 5000 peticiones y 10 concurrentes
- 10000 peticiones y 100 concurrentes

Advertencia: Modifica el número de peticiones y el nivel de concurrencia, adecuándolo a tu infraestructura. Elige los datos teniendo en cuenta que vamos a hacer pruebas con una carga baja de peticiones, otra media y otra alta.

Hay que generar gráficas de uso de memoria y rendimiento donde se vea la comparativa entre las distintas formas de ejecutar PHP:

- Módulo php5-apache2
- Módulo php5-apache2 + memcached
- Módulo php5-apache2 + varnish
- FPM-PHP + event
- FPM-PHP + event + memcached
- FPM-PHP + event + varnish

Genera una documentación que al menos tenga los siguientes puntos:

1. Introducción. Explicación de los módulos de multiprocesamiento. Objetivos de la práctica.
2. **Configuración de los escenarios:**
 - Instalación del módulo php de apache2.
 - Instalación y configuración de memcached
 - Instalación y configuración de vanish
 - Instalación y configuración de FPM-PHP con el módulo de multiprocesamiento event
 - Configuración de memcached y vanish con la nueva configuración
3. Generación de las gráficas de uso de memoria: una para cada configuración en cada una de las cargas (18 en total).
4. Generación de las gráficas de rendimiento: una para cada carga, con 6 gráficas.
5. CONCLUSIONES. Lo más valorado en la tarea serán las conclusiones a las que llegas.

Nota: Antes de empezar a escribir la documentación, documenta en redmine de manera individual las siguientes tareas:

- **Tarea 1 (2 punto)(Obligatorio):** Documenta la instalación del módulo php de apache2. Muestra wordpress funcionando con el módulo php de apache2. Realiza una comprobación de que, efectivamente, se está usando el módulo php.
- **Tarea 2 (2 puntos)(Obligatorio):** Documenta la instalación y configuración de memcached. Entrega una comprobación de que memcached está funcionando.
- **Tarea 3 (2 puntos)(Obligatorio):** Documenta la instalación y configuración de varnish. Entrega una comprobación de que varnish está funcionando.
- **Tarea 4 (2 puntos)(Obligatorio):** Documenta la instalación y configuración de FPM-PHP con el módulo de multiprocesamiento event (desinstala el memcached y vanish). Muestra wordpress funcionando con FPM-PHP. Realiza una comprobación de que, efectivamente, se está usando FPM-PHP.
- **Tarea 5 (2 puntos)(Obligatorio):** Entrega y muestra una comprobación de que memcached está funcionando con la nueva configuración.
- **Tarea 6 (2 puntos)(Obligatorio):** Entrega y muestra una comprobación de que varnish está funcionando con la nueva configuración.

Ahora puedes realizar el trabajo y generar la documentación en pdf requerida, teniendo en cuenta los siguientes puntos:

- **Tarea 7 (4 puntos):** Introducción del trabajo. Explica los objetivos de la práctica. Explica los módulos de multiprocesamiento.
- **Tarea 8 (6 puntos):** Generación de las gráficas.
- **Tarea 9 (8 puntos):** Conclusiones. Explica razonadamente a las conclusiones que has llegado.

Práctica: Gestionar un hosting por ftp

Nota: (5 tareas - 15 puntos)(2 tareas obligatorias - 5 puntos)

Nota:

- Cuando termines las tareas tienes que realizar una prueba de funcionamiento al profesor.

Tenemos que desarrollar la página del instituto `www.iesgn.org`, y queremos que sea gestionada por medio de un ftp. Tendremos las siguientes funcionalidades:

- Queremos ofrecer una colección de documentos, y lo vamos a hacer mediante http y ftp anónimo, de esta forma se accederá al mismo directorio si accedo a las siguientes URL:
 - `http://www.iesgn.org/documentos`
 - `ftp://ftp.iesgn.org`

Advertencia:

- **Tarea 1 (2 puntos)(Obligatorio):** Configura apache2 y el servidor ftp de forma anónimo para obtener el resultado pedido. Entrega la configuración de ambos servidores y muestra al profesor el funcionamiento.
- Cada departamento tendrá su página web en la URL `www.iesgn.org/nombredeldepartamento`, veamos un ejemplo:
 - El departamento de matemáticas tendrá su página en `www.iesgn.org/matematicas`
 - Se creará un usuario `user_matematicas`, que tendrá una contraseña, para que accediendo a `ftp.iesgn.org`, pueda gestionar los ficheros de su página web.Determina los cambios que tienes que ir realizando para ir creando el espacio web para cada uno de los departamentos.

Advertencia:

- **Tarea 2 (3 puntos)(Obligatorio):** Entrega una pequeña guía que explique las acciones que hay que realizar para crear la página web de un determinado departamento. Muestra al profesor el funcionamiento para la asignatura de “informática”.
- Escribe un pequeño script `crear_pagina_web`, que recibe como parámetro el nombre del departamento y realiza los pasos para crear la página web de departamento y que un usuario accediendo por ftp pueda gestionarla.

Advertencia:

- **Tarea 3 (3 puntos):** Entrega la url del repositorio github, y haz una prueba al profesor.
- Instala la aplicación web net2ftp en el servidor por si tenemos problemas de acceso por el puerto 20/21.

Advertencia:

- **Tarea 4 (3 puntos):** Muestra al profesor al acceso a la aplicación web net2ftp.
- El uso de usuarios reales del sistema para el acceso FTP puede tener varias desventajas (gestión, seguridad,...). Modifica la configuración del sistema para que se usen usuarios virtuales para el acceso por FTP, cuya información este guardada en una tabla mysql o en un directorio ldap.

Advertencia: Tarea 5 (4 puntos): Entrega los pasos más relevantes para realizar esta tarea. Y muestra al profesor su funcionamiento.

Práctica: Implantación de un servidor de hosting

Nota: (9 tareas - 35 puntos)(3 tareas obligatorias - 10 puntos)

El objetivo de la práctica es montar un servidor que ofrezca un servicio de hospedaje de páginas web con las siguientes características:

- Podemos dar de alta a un usuario y al nombre de dominio (por ejemplo `nombrededominio.com`) por el que va estar referido su espacio.
- Se podrán hospedar páginas estáticas (html) y páginas web dinámicas construidas con PHP.
- Automáticamente se creará una página principal, que al acceder a la pagina web (`www.nombrededominio.com`) de la bienvenida e informe que dicha página está en construcción.
- Para gestionar los ficheros hospedados en nuestro espacio utilizaremos un servidor FTP `ftp.nombrededominio.com`.
- Para gestionar las tablas de mysql accederemos al programa phpmyadmin en la dirección `mysql.nombrededominio.com`.

Advertencia:

- **Tarea 1 (4 puntos)(Obligatorio):** Contesta las siguientes preguntas:
 1. ¿Qué servidores necesitas instalar en la máquina donde vamos a implantar el hosting? Cuando damos de alta una nueva cuenta en nuestro hosting hay que indicar un usuario y un nombre de dominio. ¿Qué acciones hay que hacer en el servidor con el usuario? ¿Qué acciones hay que hacer con el nombre de dominio?
 2. ¿Cómo puedes comprobar que existe ya un usuario con el mismo nombre?¿y qué ya está dado de alta un determinado nombre de dominio?
 3. ¿Qué debes tener en cuenta, a la hora de crear el directorio home del usuario, para que accediendo por ftp, el usuario pueda gestionar su página web?
 4. ¿Cuántos nombres habrá que dar de alta en la zona de resolución directa de `nombrededominio.com`?

Especificaciones técnicas mínimas

- El sistema utilizará usuarios virtuales cuya información estará guardada en una base de datos mysql.
- El administrador debe decidir la estructura para guardar los directorios personales de los usuarios. Cuando se da de alta un nuevo usuario con un nombre de dominio, habrá que tener en cuenta las siguientes consideraciones:
 1. Si el usuario o el nombre del dominio existen, no se continúa.
 2. Se creará el directorio personal del usuario, este directorio será el `DocumentRoot` del servidor web. En este directorio se tendrá que crear una página web de bienvenida.
 3. Se creará un nuevo virtual hosting (`www.nombrededomino.com`) con el `DocumentRoot` apuntando al directorio personal que anteriormente hemos instalado.

4. Se creará un nuevo usuario virtual para el acceso por FTP. El administrador decidirá la política para generar la contraseña. Dicha contraseña generada tendrá que visualizarse por pantalla. La contraseña será guardada en la base de datos encriptada.
5. Se creará un nuevo usuario en el gestor de base de datos mysql, se debe llamar `mynombredeusuario`, la contraseña que se genere para mysql debe ser distinta a la generada para la gestión del FTP y también se debe mostrar.
6. Se creará una nueva zona `nombrededominio.com` en el servidor DNS bind9 con las zonas de resolución directa e inversa que permitan conocer los distintos nombres (`www,ftp,mysql,...`)

Advertencia:

- **Tarea 2 (3 puntos)(Obligatorio):** Escribe una guía donde se vean los comandos y las modificaciones de configuración que hay que hacer en los servidores, para conseguir cada uno de los puntos anteriormente descritos.
- **Tarea 3 (3 puntos)(Obligatorio):** Escribe una guía donde se vean los comandos y las modificaciones de configuración que hay que hacer para dar de baja a un usuario y el nombre de dominio asociado.

Creación de un script

Crea los siguientes scripts:

- Un script bash/python (`alta`) que reciba el nombre del usuario y el nombre de dominio relacionado con el usuario, y realice los pasos mostrados anteriormente.
- Un script bash/python (`baja`) que reciba un nombre de dominio e elimine la cuenta del usuario relacionado a dicho nombre de dominio. Borrará el virtual host de apache2, la zona del servidor DNS, el usuario de la base de datos y las bases de datos creados, el usuario virtual para el acceso a la base de datos y el directorio personal del usuario.
- Un script bash/python (`change_password`) que nos permite cambiar las contraseñas de un determinado usuario. Por lo tanto recibe el nombre de un usuario, una opción (`-sql`, si queremos cambiar la contraseña de mysql, o `-ftp`, si queremos cambiar la contraseña del ftp) y una nueva contraseña y haga la modificación de la contraseña indicada si el usuario existe.

Advertencia:

- **Tarea 4 (10 puntos: script de alta (5 ptos), script de baja (3 puntos), script de cambio de contraseña (2 puntos)):** Entrega la url del repositorio github donde has guardado los scripts.

Configuración de estadísticas Webs

Configura el sistema para que todos los usuarios puedan acceder a las estadísticas de su alojamiento web usando el programa awstats. Tendremos que tener en cuenta que el acceso a esta información no será público, para acceder a ella el usuario se tendrá que autenticar con el nombre de usuario y la contraseña que se han generado para la gestión ftp.

Advertencia:

- **Tarea 5 (2 puntos):** Crea un sistema de estadística con awstats que sea accesible desde la URL: `stats.nombrededominio.com`.

Utilización de cuotas

Advertencia:

- **Tarea 6 (2 puntos):** Investiga la forma de limitar el espacio que los usuarios tienen a su disposición, por ejemplo que cada usuario tenga un espacio limitado de 100 Mb.

Usuarios virtuales con LDAP**Advertencia:**

- **Tarea 7 (3 puntos):** Modifica toda la configuración para que los usuarios virtuales que estamos usando se guarden en un servidor LDAP.

Aplicación web para la gestión del hosting**Advertencia:**

- **Tarea 8 (4 puntos):** Crea una aplicación web (con cualquier tecnología, por ejemplo bottle o django) que permita gestionar el hosting: dar de alta nuevas cuentas, borrarlas, modificar contraseñas, ...

Creación de subdominios

Queremos añadir a nuestro hosting para que podamos dar de alta a nuevos subdominios. Por ejemplo podemos crear el subdominio `web.nombrededominio.com` que será otra página web del usuario.

Advertencia:

- **Tarea 9 (4 puntos):** Crea un script que nos permita gestionar subdominios en nuestro hosting.

Práctica: Servidor de correos

Nota: (11 tareas - 30 puntos)(5 tareas obligatorias - 8 puntos)

Nota:

- Muestra al profesor: Tareas 3,4,5,6,7,8,9,10
-

Esta tarea consiste en instalar y configurar un servidor de correo similar al de cualquier organización, capaz de enviar y recibir directamente correo, almacenar los usuarios en LDAP, filtrar el correo en busca de virus o spam y servirlo a sus usuarios a través de los protocolos POP, IMAP y configurar un Webmail. Objetivos

1. Instalar y configurar un servidor postfix en un equipo con dirección IP pública dinámica
2. Aprender a configurar todos los componentes de un servidor de correos completo
3. Depurar el funcionamiento de un servicio
4. Documentar adecuadamente todo el proceso

Pasos a realizar en clase

Vamos a realizar un sistema de correo para el dominio tudominio.gonzalonazareno.org, cuyo servidor DNS lo administras en tu propio servidor DNS. Tienes que comunicar el nombre de dominio al profesor para configurar el servidor de correos del departamento. Instala postfix y comprueba que recibe correo directamente desde un equipo de Internet (hotmail, gmail, etc.). Configura tu servidor de correos para que use a babuino como relay y comprueba que puedes enviar correos.

Nota:

- **Tarea 1 (1 puntos)(Obligatorio):** Documenta en redmine una prueba de funcionamiento, donde envíes desde tu servidor local al exterior. Muestra el log donde se vea el envío. Muestra el correo que has recibido.
- **Tarea 2 (1 puntos)(Obligatorio):** Documenta en redmine una prueba de funcionamiento, donde envíes un correo desde el exterior(gmail, hotmail,...) a tu servidor local. Muestra el log donde se vea el envío. Muestra cómo has leído el correo.

Instala y configura un servidor dovecot POP e IMAP en tu equipo. Configura adecuadamente un cliente de correo (evolution, outlook, thunderbird, ...) para que reciba el correo a través de POP o IMAP. El cliente debe estar configurado en una máquina cliente. Nombra en tu servidor DNS al servidor smtp, pop e imap.

Nota:

- **Tarea 3 (2 puntos)(Obligatorio):** Documenta en redmine una prueba de funcionamiento, donde envíes desde tu cliente de correos al exterior. ¿Cómo se llama el servidor para enviar el correo? (Muestra la configuración).
- **Tarea 4 (2 puntos)(Obligatorio):** Documenta en redmine una prueba de funcionamiento, donde recibas un correo desde el exterior(gmail, hotmail,...) y lo leas en tu cliente de correo. Utiliza el protocolo POP. ¿Cómo se llama el servidor para enviar el correo? (Muestra la configuración). Muestra una prueba de funcionamiento de cómo funciona el protocolo POP.
- **Tarea 5 (2 puntos)(Obligatorio):** Documenta en redmine una prueba de funcionamiento, donde recibas un correo desde el exterior(gmail, hotmail,...) y lo leas en tu cliente de correo. Utiliza el protocolo IMAP. ¿Cómo se llama el servidor para enviar el correo? (Muestra la configuración). Muestra una prueba de funcionamiento de cómo funciona el protocolo IMAP.

Instala un webmail (roundcube) para gestionar el correo del equipo mediante una interfaz web. Instala y configura correctamente un sistema de filtrado de virus y spam utilizando amavis, clamav y spamassassin.

Nota:

- **Tarea 6 (3 puntos):** Muestra al profesor el envío y recepción de correos utilizando el webmail.
- **Tarea 7 (5 puntos):** Muestra al profesor el funcionamiento del sistema de filtrado de virus y spam.

Pasos a realizar en casa

- Configura adecuadamente el router de casa para que el puerto 25/tcp de tu equipo sea accesible desde Internet (eso se denomina DNAT o port forwarding)
- Date de alta en un servidor DNS dinámico como dyndns.org, no-ip.com, etc. o usa el nombre de dominio propio.

- Configura el DNS de tu proveedor para que la máquina a la que apunta el registro MX corresponda a tu IP pública. Si vas a utilizar un servicio gratuito como dyndns.org, no-ip.com, simplemente debes configurarlo para que apunte a tu ip.
- Instala postfix en tu máquina y comprueba que recibe correo directamente desde un equipo de Internet (hotmail, gmail, etc.)
- Prueba a enviar desde tu equipo un correo electrónico a josedom24@openmailbox.org , que no lo rechazará aunque venga de una dirección IP dinámica.
- Prueba a enviar desde tu equipo un correo electrónico a hotmail/gmail y comprueba que rebota los mensajes (mira en /var/log/mail.log), ya que no acepta correos de direcciones IP dinámicas.
- Configura postfix para que envíe el correo electrónico a través de gmail como se indica en la documentación. Cuando funcione envía un correo a josedom24@gmail.com

Nota:

- **Tarea 8 (2 puntos):** Envía el correo a josedom24@openmailbox.org
 - **Tarea 9 (3 puntos):** Responde al correo que yo te voy a mandar desde esa dirección.
 - **Tarea 10 (4 puntos):** ¿Te rebota el correo enviado al exterior por qué estas usando ip dinámica? Independientemente de la respuesta, muestra el log donde se vea el envío de ese correo y documenta la configuración del relay con gmail. Finalmente envía un correo a josedom24@gmail.com.
-

Tarea adicional

En el servidor de clase, configura postfix para que use usuarios virtuales guardados en un servidor ldap.

Instala un esquema adecuado para usuarios de postfix en LDAP y crea un script que reciba un nombre de usuario y añada un nuevo registro al LDAP:

1. El dn debes ajustarlo a la base a la de tu directorio
2. Cada entrada incluye un objectClass y atributos adecuados para postfix
3. El atributo mail es del tipo [usuario@dominio](#)
4. El buzón de cada usuario está en formato Maildir
5. El atributo userPassword es un hash SSHA del uid del usuario

Nota:

- **Tarea 11 (5 puntos):** Documenta en redmine la configuración realizada. Y realiza una prueba de funcionamiento al profesor.
-

Práctica: Servidor proxy/cache squid

Nota: (8 tareas - 20 puntos)(4 tareas obligatorias - 7 puntos)

Nota:

- Muestra al profesor: Tareas 1,2,3,4,5,6
-

Queremos instalar un servidor proxy/cache en nuestro servidor. Con ello vamos a poder controlar las páginas web a las que accedamos, además de acelerar nuestra navegación.

Nos piden la configuración de un proxy/cache/filtro en nuestra infraestructura. Hemos elegido como proxy/cache squid3, y como filtro de contenido dansguardian. Tenemos que tener en cuenta las siguientes consideraciones:

1. El proxy/cache sólo admite conexiones de la red local.
2. Tenemos dos clases de usuarios: profesores y alumnos.
3. Todos los profesores acceden al proxy con un solo nombre de usuario (profesor) y una única contraseña (pass_profe).
4. Todos los alumnos acceden al proxy con un solo nombre de usuario (alumno) y una única contraseña (pass_alum).
5. Se deniega cualquier conexión que no este autenticada con alguno de estos usuarios.
6. **Se quieren limitar las siguientes conexiones:**
 - **Para los profesores y alumnos:**
 - No se pueden bajar ficheros que se puedan instalar (exe,msi,rar,zip,bin,iso).
 - No tienen acceso a internet los fines de semana.
 - **Para los alumnos:**
 - No pueden ver contenido multimedia.
 - Sólo tienen conexión de 8:00 h. a 14:00 h.
7. El control de las páginas permitidas se hará mediante listas negras usando squid.
8. Por último tendremos instalado un programa para monitorizar el uso del proxy: sarg. Para visualizar la información generada por dicho programa accederemos a una página web llamada `proxy.josedomingo.gonzalonazareno.org` que sólo será accesible si ponemos el nombre de usuario y la contraseña de los profesores.
9. Realizar el punto 7 usando el filtro de contenidos dansguardian. Es decir, las listas negras estarán gestionada por dansguardian.

Nota:

- **Tarea 1 (1 punto)(Obligatorio):** Configura de forma manual el proxy, y accede con un usuario incorrecto.
 - **Tarea 2 (2 puntos)(Obligatorio):** Accede como alumno. Intenta bajar un fichero multimedia.
 - **Tarea 3 (2 puntos)(Obligatorio):** Cambia la hora del servidor a las 15:00 de la tarde y comprueba que no hay conexión.
 - **Tarea 4 (2 puntos)(Obligatorio):** Accede como profesor e intenta bajar un fichero zip.
 - **Tarea 5 (2 puntos):** Cambia la fecha del servidor e indica un fin de semana y comprueba que no hay conexión.
 - **Tarea 6 (3 puntos):** Filtra el dominio youtube.com en la lista negra y prueba que realmente no se puede acceder.
 - **Tarea 7 (4 puntos):** Documenta la instalación de sarg, y muestra al profesor las estadísticas de acceso al proxy con sarg.
 - **Tarea 8 (4 puntos):** Documenta la configuración de Dansguardian, muestra al profesor el funcionamiento.
-

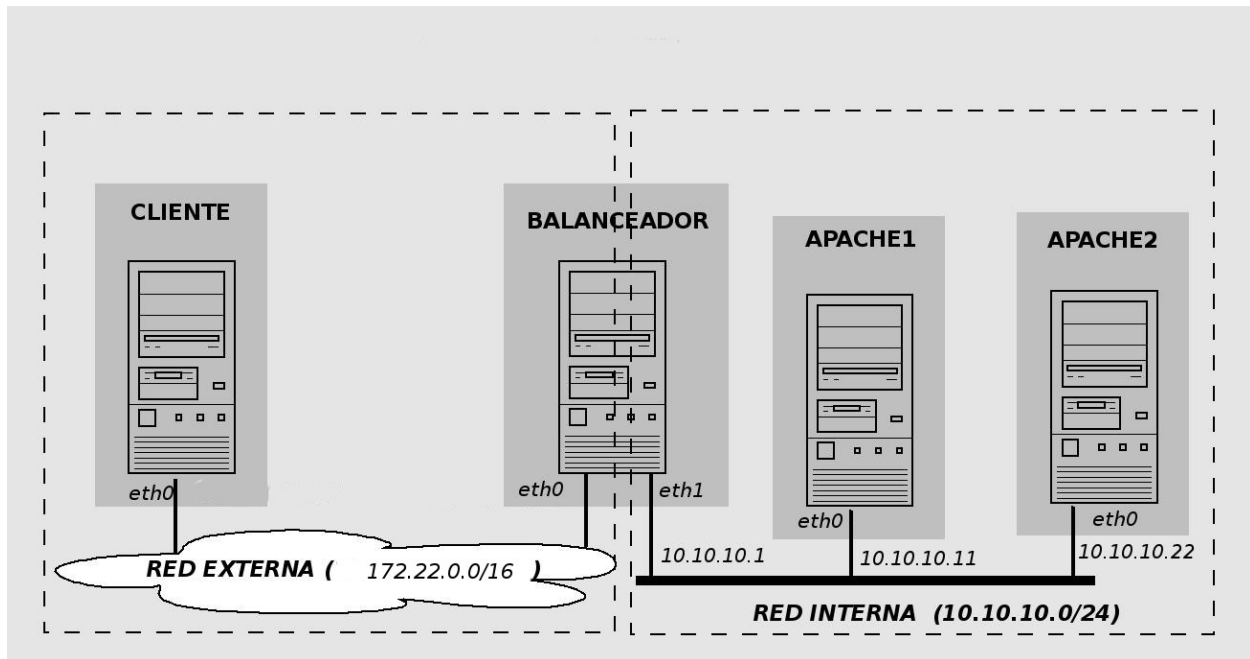
Práctica: Balanceo de carga en servidores Apache con HAProxy

Nota: (9 tareas - 20 puntos)(4 tareas obligatorias - 9 puntos)

Nota:

- Muestra al profesor: Tareas 1,2,3,7

En primer lugar, construye con KVM con vagrant la siguiente infraestructura:



Ajustar la configuración de las dos máquinas del cluster de balanceo (`apache1` y `apache2`):

1. Deshabilitar la opción `KeepAlive` en el fichero de configuración `/etc/apache2/apache2.conf` para realizar la evaluación del rendimiento sin la opción de reutilización de conexiones:

```
apache1:~# nano /etc/apache2/apache2.conf
...
KeepAlive Off
...

apache2:~# nano /etc/apache2/apache2.conf
...
KeepAlive Off
...
```

Advertencia: Nota: este ajuste no es estrictamente necesario (y sería desaconsejable en un entorno de producción real), pero facilita las pruebas manuales dado que permite detectar inmediatamente el “cambio” de destino resultado del balanceo de carga manteniendo la opción por defecto, en las pruebas manuales desde el navegador sería necesario esperar 5 segundos (el time out de keep alive) antes de recargar la página y ver el efecto del reparto de carga

2. Editar los archivos del sitio web para incluir una indicación del servidor real que está sirviendo una petición, de modo que sea posible “diferenciarlos” en las pruebas manuales con el navegador

En apache1:

```
apache1:~# nano /var/www/html/index.html
...
<h1> Servidor por APACHE_UNO </h1>
...
apache1:~# nano /var/www/html/sesion.php

<?php
header('Content-Type: text/plain');
session_start();
if(!isset($_SESSION['visit']))
{
    echo "This is the first time you're visiting this server";
    $_SESSION['visit'] = 0;
}
else
    echo "Your number of visits: ".$_SESSION['visit'];

$_SESSION['visit']++;

echo "\nServer IP: ".$_SERVER['SERVER_ADDR'];
echo "\nClient IP: ".$_SERVER['REMOTE_ADDR'];
echo "\nX-Forwarded-for: ".$_SERVER['HTTP_X_FORWARDED_FOR']."\n";
print_r($_COOKIE);
?>
```

En apache2:

```
apache2:~# nano /var/www/html/index.html
...
<h1> Servidor por APACHE_DOS </h1>
...
```

3. Crear en ambas máquinas (apache1, apache2) el script PHP sleep.php:

```
apache1~:~# nano /var/www/html/sleep.php

<html>
  <title> Retardos de x segundos </title>
  <body>
    <h1> Prueba con retardo de x segundos </h1>
    <p> hora de inicio: <?php echo date('h:i:s'); ?> </p>
    <?php
    for ($i=0; $i < 2000000; $i++) {
        $str1 = sha1(rand()*rand());
        $str2 = sha1(rand()*rand());
        $str3 = sha1($str1+$str2);
    }
    ?>
    <p> hora de fin: <?php echo date('h:i:s'); ?> </p>
  </body>
</html>
```

Comprobación:


```
apache1~:# php /var/www/html/sleep.php
apache2~:# php /var/www/html/sleep.php
```

Evaluar rendimiento de un servidor Apache sin balanceo

Se realizarán varias pruebas de carga sobre el servidor Apache ubicado en la máquina apache1. Pasos a realizar:

1. Habilitar en *balanceador* la redirección de puertos para que sea accesible el servidor Apache de la máquina apache1 [10.10.10.11] empleando el siguiente comando iptables:

```
balanceador:~# echo 1 > /proc/sys/net/ipv4/ip_forward
balanceador:~# iptables -t nat -A PREROUTING \
    --in-interface eth0 --protocol tcp --dport 80 \
    -j DNAT --to-destination 10.10.10.11
```

Advertencia: Nota: la regla iptables establece una redirección del puerto 80 de la máquina balanceador al mismo puerto de la máquina apache1 para el tráfico procedente de la red externa (interfaz de entrada eth0).

2. Arrancar en apache1 [10.10.10.11] el servidor web Apache2:

```
apache1:~# systemctl start apache2
```

Advertencia: Nota: Desde la máquina cliente se puede abrir en un navegador web la URL <http://172.22.x.x> para comprobar que el servidor está arrancado y que la redirección del puerto 80 está funcionando.

3. Lanzar las pruebas de carga iniciales sobre balanceador usando el herramienta Apache Benchmark

Prueba 1: Contenido estático:

```
cliente:~# ab -n 2000 -c 10 http://172.22.x.x/index.html
cliente:~# ab -n 2000 -c 50 http://172.22.x.x/index.html
cliente:~# ab -n 2000 -c 100 http://172.22.x.x/index.html
```

Envía 2000 peticiones HTTP sobre la URI “estática”, manteniendo, respectivamente, 10 y 50 conexiones concurrentes.

Prueba 2: Scripts PHP

Se usará un script PHP (sleep.php) que introduce un retardo mediante un bucle “activo” de 2000000 iteraciones que busca forzar el uso de CPU con cálculos de hashes SHA1 y concatenaciones de cadenas:

```
cliente:~# ab -n 250 -c 10 http://172.22.x.x/sleep.php
cliente:~# ab -n 250 -c 30 http://172.22.x.x/sleep.php
cliente:~# ab -n 250 -c 50 http://172.22.x.x/sleep.php
```

Envía 250 peticiones HTTP sobre la URI “dinámica”, manteniendo, respectivamente, 10 y 30 conexiones concurrentes. (aprox 5-7 minutos)

Nota:

- **Tarea 1 (3 puntos)(Obligatorio):** Ejecuta varias veces los comandos ab con cada una de las pruebas y calcula la media de los resultados obtenidos (Requests per second (número peticiones por segundo) ó Time per request (tiempo en milisegundos para procesar cada petición) para cada una de las cargas.

Configurar y evaluar balanceo de carga con dos servidores Apache

1. Deshabilitar la redirección del puerto 80 de la máquina balanceador concatenaciones el siguiente comando iptables (HAproxy se encargará de retransmitir ese tráfico sin necesidad de redireccionar los puertos)
2. Arrancar los servidores Apache de apache1 [10.10.10.11] y apache2 [10.10.10.22]
3. Instalar HAproxy en balanceador
4. Configurar HAproxy en balanceador (de momento sin soporte de sesiones persistentes):

```
balanceador:~# cd /etc/haproxy
balanceador:/etc/haproxy/# mv haproxy.cfg haproxy.cfg.original
balanceador:/etc/haproxy/# nano haproxy.cfg

global
    daemon
    maxconn 256
    user      haproxy
    group     haproxy
    log       127.0.0.1      local0
    log       127.0.0.1      local1  notice

defaults
    mode      http
    log       global
    timeout connect 5000ms
    timeout client 50000ms
    timeout server 50000ms

listen granja_cda
    bind 193.147.87.47:80
    mode http
    stats enable
    stats auth cda:cda
    balance roundrobin
    server uno 10.10.10.11:80 maxconn 128
    server dos 10.10.10.22:80 maxconn 128
```

Define (en la sección listen) un “proxy inverso” de nombre granja_cda que:

- trabajará en modo http (la otra alternativa es el modo tcp, pero no analiza las peticiones/respuestas HTTP, sólo retransmite paquetes TCP)
- atendiendo peticiones en el puerto 80 del balanceador
- con balanceo round-robin
- que repartirá las peticiones entre dos servidores reales (de nombres uno y dos) en el puerto 80 de las direcciones 10.10.10.11 y 10.10.10.22
- adicionalmente, habilita la consola Web de estadísticas, accesible con las credenciales cda:cda

Más detalles en * [Opciones de configuración HAProxy 1.5](#)

5. Iniciar HAproxy en balanceador: Antes de hacerlo es necesario habilitar en /etc/default/haproxy el arranque de HAproxy desde los scripts de inicio, estableciendo la variable ENABLED=1
6. Desde la máquina cliente abrir en un navegador web la URL <http://172.22.x.x> y recargar varias veces para comprobar como cambia el servidor real que responde las peticiones.

Advertencia: Nota: Si no se ha deshabilitado la opción KeepAlive de Apache, es necesario esperar 5 segundos entre las recargas para que se agote el tiempo de espera para cerrar completamente la conexión HTTP y que pase a ser atendida por otro servidor.

Nota:

- **Tarea 2 (1 puntos)(Obligatorio):** Muestra al profesor y entrega capturas de pantalla que el balanceador está funcionando.

7. Desde la máquina cliente repetir las pruebas de carga con ab. Los resultados deberían de ser mejores que con la prueba anterior con un servidor Apache único (al menos en el caso del script `sleep.php`).

Nota:

- **Tarea 3 (3 puntos)(Obligatorio):** Ejecuta varias veces los comandos ab con cada una de las pruebas y calcula la media de los resultados obtenidos (Requests per second (número peticiones por segundo) ó Time per request (tiempo en milisegundos para procesar cada petición)) para cada una de las cargas. ¿Son mejores que con un solo servidor web?

8. Desde la máquina cliente abrir en un navegador web la URL <http://172.22.x.x/haproxy?stats> para inspeccionar las estadísticas del balanceador HAProxy (pedirá un usuario y un password, ambos cda)

Nota:

- **Tarea 4 (1 punto):** Entrega una captura de pantalla donde se vea la página web de estadísticas de haproxy.

9. Desde uno de los servidores (apache1 ó apache2), verificar los logs del servidor Apache:

```
apacheN:~# tail /var/log/apache2/error.log
apacheN:~# tail /var/log/apache2/access.log
```

Nota:

- **Tarea 5 (1 punto):** En todos los casos debería figurar como única dirección IP cliente la IP interna de la máquina balanceador [10.10.10.1]. ¿Por qué?

Configurar la persistencia de conexiones Web (sticky sessions)

1. Detener HAProxy en la máquina balanceador
2. Añadir las opciones de persistencia de conexiones HTTP (sticky cookies) al fichero de configuración:

```
balanceador:~# nano /etc/haproxy/haproxy.cfg
```

Contenido a incluir: (añadidos marcados con <- aquí):

```
global
    daemon
    maxconn 256
    user    haproxy
```

```
group    haproxy
log      127.0.0.1      local0
log      127.0.0.1      local1  notice

defaults
  mode    http
  log     global
  timeout connect 10000ms
  timeout client  50000ms
  timeout server  50000ms

listen granja_cda
  bind 172.22.x.x:80 #aquí pon la dirección ip del balanceador
  mode http
  stats enable
  stats auth cda:cda
  balance roundrobin
  cookie PHPSESSID prefix           # <- aquí
  server uno 10.10.10.11:80 cookie EL_UNO maxconn 128 # <- aquí
  server dos 10.10.10.22:80 cookie EL_DOS maxconn 128 # <- aquí
```

El parámetro cookie especifica el nombre de la cookie que se usa como identificador único de la sesión del cliente (en el caso de aplicaciones web PHP se suele utilizar por defecto el nombre PHPSESSID). Para cada “servidor real” se especifica una etiqueta identificativa exclusiva mediante el parámetro cookie. Con esa información HAproxy reescribirá las cabeceras HTTP de peticiones y respuestas para seguir la pista de las sesiones establecidas en cada “servidor real” usando el nombre de cookie especificado (PHPSESSID):

- conexión cliente -> balanceador HAproxy : cookie original + etiqueta de servidor
- conexión balanceador HAproxy -> servidor : cookie original

3. Iniciar HAproxy en la máquina balanceador

4. En la máquina cliente, arrancar el sniffer de red whreshark y ponerlo en escucha sobre el interfaz eth0 (fijar como filtro la cadena http para que solo muestre las peticiones y respuestas HTTP).

5. En la máquina cliente:

- desde el navegador web acceder varias veces a la URL <http://172.22.x.x/sesion.php> (comprobar el incremento del contador [variable de sesión])
- acceder la misma URL desde el navegador en modo texto lynx (o desde una pestaña de “incógnito” de Ice-weasel para forzar la creación de una nueva sesión):

```
cliente:~# lynx -accept-all-cookies http://172.22.x.x/sesion.php
```

6. Detener la captura de tráfico en wireshark y compr. note::obar las peticiones/respuestas HTTP capturadas.

Nota:

- **Tarea 6 (3 puntos):** Verificar la estructura y valores de las cookies PHPSESSID intercambiadas. En la primera respuesta HTTP (inicio de sesión), se establece su valor con un parámetro HTTP SetCookie en la cabecera de la respuesta. Las sucesivas peticiones del cliente incluyen el valor de esa cookie (parámetro HTTP Cookie en la cabecera de las peticiones)

Configurar lbass en openstack

1. Crea dos instancias en el cloud que sean servidor web y crea en cada una de ella un fichero index.html y sesion.php similares a los de la tarea anterior.
2. Siguiendo la documentación ofrecida, configura un balanceador d carga en openstack.

Nota:

- **Tarea 7 (2 puntos)(Obligatorio):** Muestra al profesor y documenta el proceso de creación del balanceador de carga para comprobar el funcionamiento cuando accedemos a las páginas html.
 - **Tarea 8 (3 puntos):** Configura de manera adecuada el balanceador de carga para que tenga en cuenta la persistencia de sesiones. Muestra al profesor su funcionamiento accediendo al fichero sesion.php y documenta los cambios que has configurado.
 - **Tarea 9 (3 puntos):** Crea otra instancia con un servidor mysql, e instala en los servidores web un CMS wordpress que accedan a la misma base de datos. Comprueba que el balanceado se produce de manera adecuada
-

Calificaciones del módulo