CURSO 2024/25

# ADMINISTRACION DE SISTEMAS Y REDES

PRACTICA 7

DIEGO GARCÍA GONZÁLEZ

# Índice

# 1. Instalación y Verificación del Servidor Apache

## Configuración del hostname y prueba de red

Lo primero de todo que haremos, será encender nuestra máquina Linux. Una vez arrancado, nos aseguramos que la máquina Linux se llame linux.as.local. Podemos comprobarlo con “uname -a”. En caso de no llamarse de esta manera, usaremos el comando hostnamectl set-hostname linux.as.local (como en la práctica 1). En esta máquina vamos a montar un servidor web apache.

```
[\UO294255@linux ~]# uname -a
Linux linux.as.local 5.14.0-503.21.1.el9_5.x86_64 #1 SMP PREEMPT_DYNAMIC Sun Jan 12 09:45:05 EST 2025 x86_64 x86_64 x86_64 GNU/Linux
[\UO294255@linux ~]# _
```

Una vez puesto el hostname, comprobamos con ping que tienes acceso a la red:

```
[\UO294255@linux ~]# ping www.google.es
PING www.google.es (172.217.168.163) 56(84) bytes of data.
64 bytes from mad07s10-in-f3.1e100.net (172.217.168.163): icmp_seq=1 ttl=117 time=15.8 ms
64 bytes from mad07s10-in-f3.1e100.net (172.217.168.163): icmp_seq=2 ttl=117 time=15.9 ms
64 bytes from mad07s10-in-f3.1e100.net (172.217.168.163): icmp_seq=3 ttl=117 time=14.5 ms
64 bytes from mad07s10-in-f3.1e100.net (172.217.168.163): icmp_seq=4 ttl=117 time=56.9 ms
64 bytes from mad07s10-in-f3.1e100.net (172.217.168.163): icmp_seq=5 ttl=117 time=14.6 ms
64 bytes from mad07s10-in-f3.1e100.net (172.217.168.163): icmp_seq=6 ttl=117 time=14.4 ms
64 bytes from mad07s10-in-f3.1e100.net (172.217.168.163): icmp_seq=7 ttl=117 time=15.3 ms
64 bytes from mad07s10-in-f3.1e100.net (172.217.168.163): icmp_seq=8 ttl=117 time=15.1 ms
64 bytes from mad07s10-in-f3.1e100.net (172.217.168.163): icmp_seq=9 ttl=117 time=15.2 ms
64 bytes from mad07s10-in-f3.1e100.net (172.217.168.163): icmp_seq=10 ttl=117 time=15.4 ms
```

## Configuración de DNS y resolución de nombres

Ahora arrancaremos ahora la máquina WS2022, y comprobaremos que se resuelve la dirección linux.as.local desde las máquinas Linux y W10:

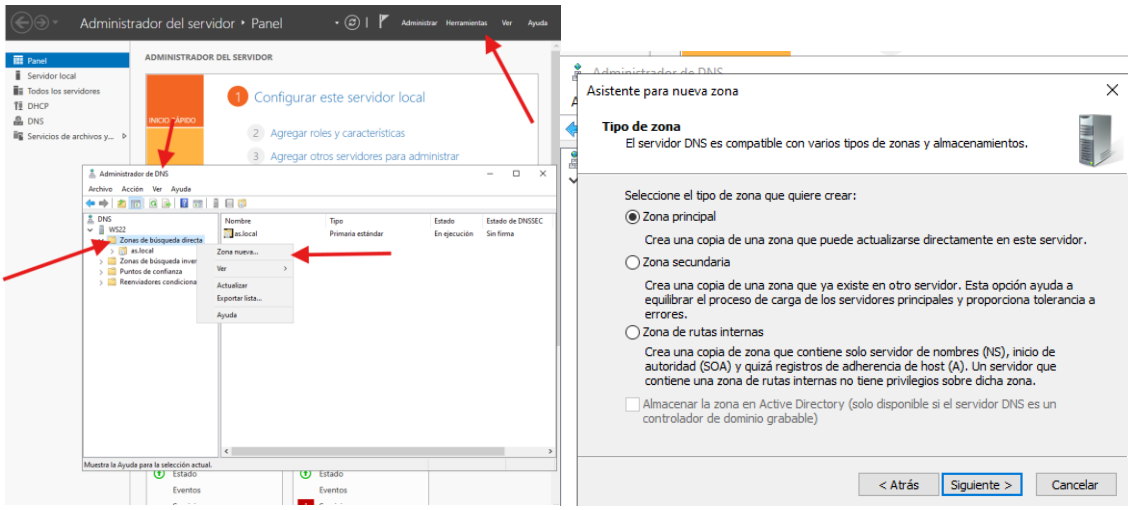
```
[\UO294255@linux ~]# nslookup linux.as.local
Server:         192.168.56.101
Address:        192.168.56.101#53

Name:   linux.as.local
Address: 192.168.56.100

[\UO294255@linux ~]# _
```

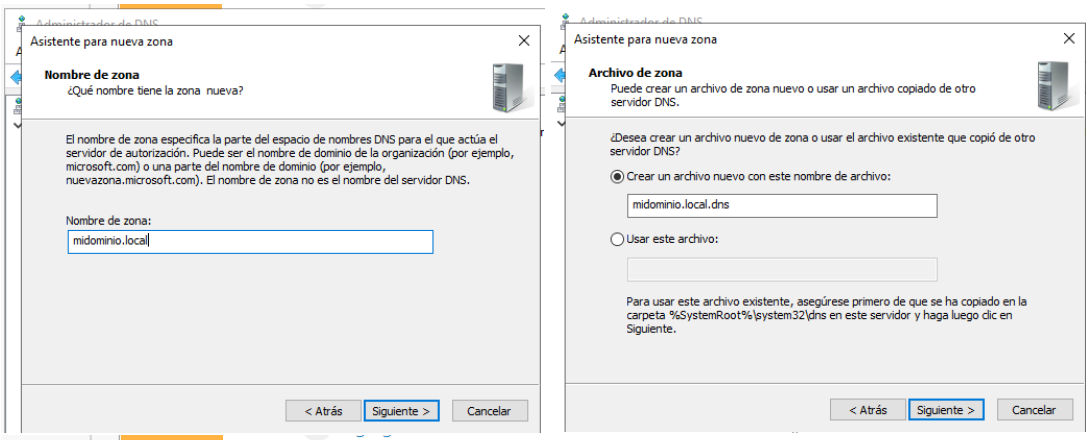
```
PS C:\Users\uoxxxxxx> nslookup linux.as.local
Servidor:  UnKnown
Address:  192.168.56.101

Nombre:  linux.as.local
Address:  192.168.56.100

PS C:\Users\uoxxxxxx> uo294255_
```

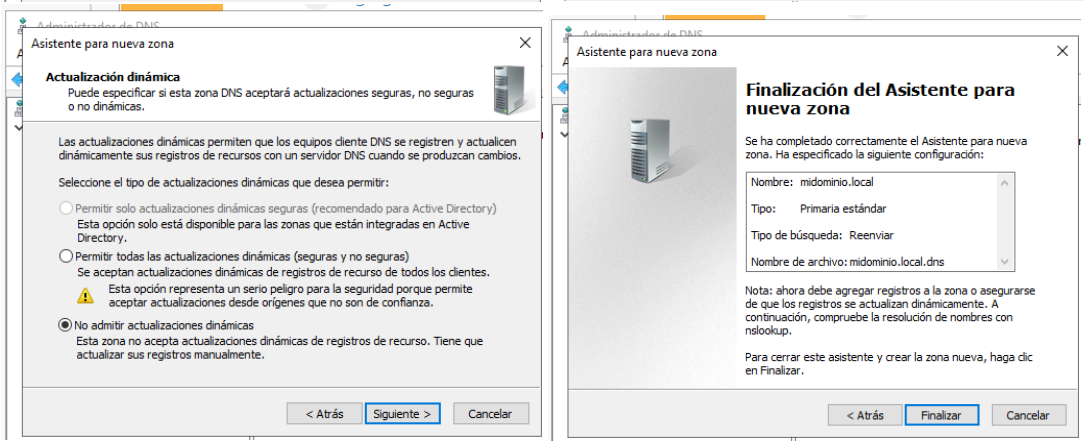
Ahora veremos si la maquina WS2022 tambien resuelve la dirección:

```
PS C:\Users\Administrador> nslookup linux.as.local
Servidor:  UnKnown
Address:  192.168.56.101

Nombre:  linux.as.local
Address:  192.168.56.100

PS C:\Users\Administrador> uo294255_
```

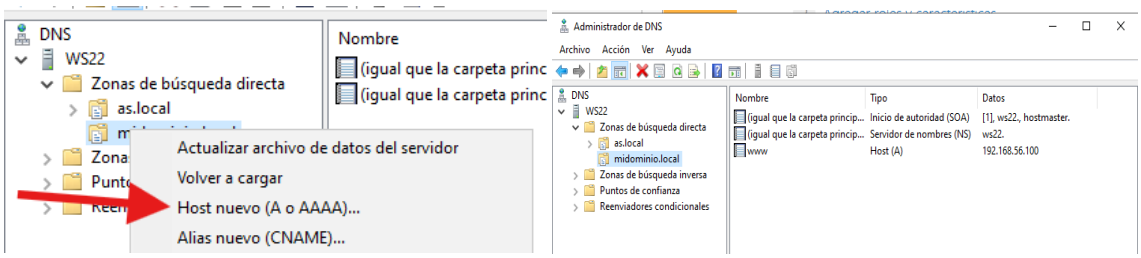
Ahora, en la zona DNS de la máquina WS2022, añadimos el “www.midominio.local” con la misma dirección IP que la máquina Linux, creando una nueva zona:




Para finalizar agregamos un registro A para www:




Comprueba que tanto la máquina Linux como W10 resuelven correctamente la dirección asignada a www.midominio.local:

```
[\UO294255@linux ~]# nslookup www.midominio.local
Server:         192.168.56.101
Address:        192.168.56.101#53

Name:   www.midominio.local
Address: 192.168.56.100

[\UO294255@linux ~]# _
```

```
PS C:\Users\uoxxxxxx> nslookup www.midominio.local
Servidor:  UnKnown
Address:  192.168.56.101

Nombre:  www.midominio.local
Address:  192.168.56.100

PS C:\Users\uoxxxxxx> uo294255_
```

## Instalación de apache

Lo que haremos ahora será instalar apache en la maquina Linux, en caso de no existir este. Para ello comprobamos con “status” si lo tenemos instalado, y en caso contrario, ejecutaremos la orden “dnf install httpd” como muestro en la ilustración:

```
[\UO294255@linux ~]# status httpd
-bash: status: orden no encontrada
[\UO294255@linux ~]# dnf install httpd
Última comprobación de caducidad de metadatos hecha hace 0:26:15, el mié 19 mar 2025 17:40:42.
Dependencias resueltas.
================================================================================
 Paquete                    Arquitectura          Versión
================================================================================
Instalando:
 httpd                      x86_64                2.4.62-1.el9_5.2
Instalando dependencias:
 almalinux-logos-httpd      noarch                90.5.1-1.1.el9
 apr                        x86_64                1.7.0-12.el9_3
 apr-util                   x86_64                1.6.1-23.el9
 apr-util-bdb               x86_64                1.6.1-23.el9
 httpd-core                 x86_64                2.4.62-1.el9_5.2
 httpd-filesystem           noarch                2.4.62-1.el9_5.2
 httpd-tools                x86_64                2.4.62-1.el9_5.2
 mailcap                    noarch                2.1.49-5.el9
Instalando dependencias débiles:
 apr-util-openssl           x86_64                1.6.1-23.el9
 mod_http2                  x86_64                2.0.26-2.el9_4.1
 mod_lua                    x86_64                2.4.62-1.el9_5.2

Resumen de la transacción
================================================================================
Instalar  12 Paquetes

Tamaño total de la descarga: 2.0 M
Tamaño instalado: 6.1 M
¿Está de acuerdo [s/N]?: s
```

Una vez instalado, configuraremos el firewall para permitir las conexiones http:

```
[\UO294255@linux ~]# firewall-cmd --zone=internal --permanent --add-service=http
success
[\UO294255@linux ~]# firewall-cmd --reload
success
[\UO294255@linux ~]# _
```

Por último, arrancamos el servicio de apache:

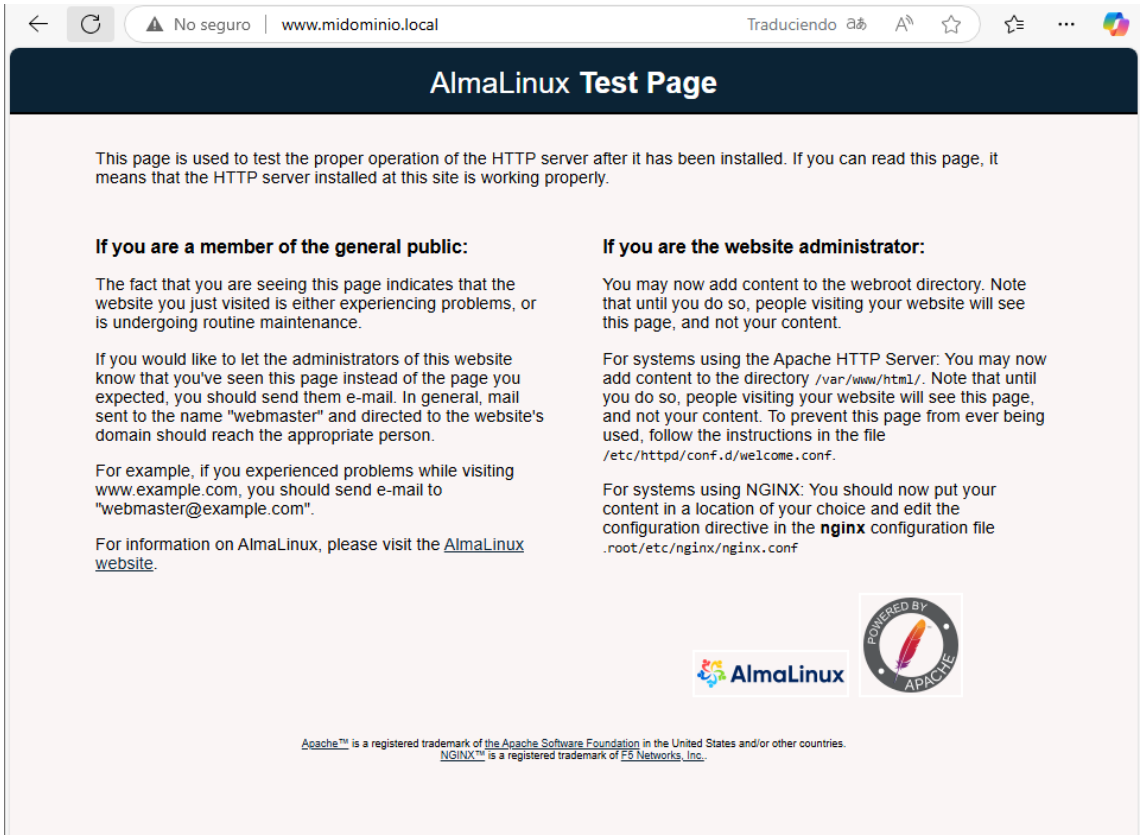
```
[\UO294255@linux ~]# systemctl start httpd
[\UO294255@linux ~]# firewall-cmd --zone=internal --permanent --add-service=http
Warning: ALREADY_ENABLED: http
success
[\UO294255@linux ~]#firewall-cmd --reload
success
```

Verificamos que funciona:

```
[\UO294255@linux ~]#systemctl status httpd
● httpd.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; preset: disabled)
     Active: active (running) since Wed 2025-03-19 18:15:46 CET; 1min 16s ago
       Docs: man:httpd.service(8)
   Main PID: 11990 (httpd)
     Status: "Total requests: 0; Idle/Busy workers 100/0;Requests/sec: 0; Bytes served/sec:   0 B/sec"
      Tasks: 177 (limit: 17424)
     Memory: 21.8M
        CPU: 98ms
     CGroup: /system.slice/httpd.service
             ├─11990 /usr/sbin/httpd -DFOREGROUND
             ├─11991 /usr/sbin/httpd -DFOREGROUND
             ├─11992 /usr/sbin/httpd -DFOREGROUND
             ├─11993 /usr/sbin/httpd -DFOREGROUND
             └─11994 /usr/sbin/httpd -DFOREGROUND

mar 19 18:15:46 linux.as.local systemd[1]: Starting The Apache HTTP Server...
mar 19 18:15:46 linux.as.local httpd[11990]: Server configured, listening on: port 80
mar 19 18:15:46 linux.as.local systemd[1]: Started The Apache HTTP Server.
[\UO294255@linux ~]# _
```

## Despliegue de la página de prueba

A través del navegador de WS2022, accederemos al dominio http://www.midominio.local, y, una vez dentro, veremos el siguiente armatroste:




Ahora en la maquina Linux, crearemos dentro del directorio /var/www/html un archivo index.html, con el siguiente contenido:

```
[\UO294255@linux html]# cd /var/www/html
```

```
[\UO294255@linux html]# sudo vi index.html
```

```
<!DOCTYPE html>
<head>
        <meta charset="utf-8" />
        <title>Servidor AS</title>
</head>
<body>
        <h1>Administración de Sistemas</h1>
        <h2>UO294255</h2>
</body>
</html>
```

# 2. Configuración de las páginas web de los usuarios

## Configuración de usuarios y permisos

Vamos a crear en caso de no existir, un usuario “asuser”, yo ya lo tenía creado:

```
[\UO294255@linux /]# id asuser
uid=1001(asuser) gid=1001(asuser) grupos=1001(asuser)
[\UO294255@linux /]# _
```

Ahora editaremos el fichero /etc/httpd/conf.d/userdir.conf:

```
[\UO294255@linux /]# sudo vi /etc/httpd/conf.d/userdir.conf_
```

Una vez aquí, comentamos la línea de “UserDir disabled” y descomentamos la de “UserDir public html”:

```
#
# UserDir: The name of the directory that is appended onto a user's home
# directory if a ~user request is received.
#
# The path to the end user account 'public_html' directory must be
# accessible to the webserver userid.  This usually means that ~userid
# must have permissions of 711, ~userid/public_html must have permissions
# of 755, and documents contained therein must be world-readable.
# Otherwise, the client will only receive a "403 Forbidden" message.
#
<IfModule mod_userdir.c>
    #
    # UserDir is disabled by default since it can confirm the presence
    # of a username on the system (depending on home directory
    # permissions).
    #
 _  #UserDir disabled

    #
    # To enable requests to /~user/ to serve the user's public_html
    # directory, remove the "UserDir disabled" line above, and uncomment
    # the following line instead:
    #
    UserDir public_html
</IfModule>

#
# Control access to UserDir directories.  The following is an example
# for a site where these directories are restricted to read-only.
#
<Directory "/home/*/public_html">
    AllowOverride FileInfo AuthConfig Limit Indexes
    Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec
    Require method GET POST OPTIONS
</Directory>
```

Ahora, estableceremos los permisos del directorio home del usuario:

```
[\UO294255@linux /]# sudo chmod 711 /home/asuser
[\UO294255@linux /]#
```

Y ejecutamos el siguiente comando para permitir que Apache pueda leer contenidos localizados en los directorios de inicio de los usuarios locales:

```
[\UO294255@linux /]# setsebool -P httpd_read_user_content on
[ 6191.081610] SELinux:  Converting 390 SID table entries...
[ 6191.085575] SELinux:  policy capability network_peer_controls=1
[ 6191.085590] SELinux:  policy capability open_perms=1
[ 6191.085596] SELinux:  policy capability extended_socket_class=1
[ 6191.085601] SELinux:  policy capability always_check_network=0
[ 6191.085606] SELinux:  policy capability cgroup_seclabel=1
[ 6191.085611] SELinux:  policy capability nnp_nosuid_transition=1
[ 6191.085616] SELinux:  policy capability genfs_seclabel_symlinks=1
[\UO294255@linux /]#
```

Luego, ejecutamos el siguiente comando para habilitar el uso de los directorios ~/public_html de los usuarios:

```
[\UO294255@linux /]# setsebool -P httpd_enable_homedirs on
[ 6250.546581] SELinux:  Converting 390 SID table entries...
[ 6250.550580] SELinux:  policy capability network_peer_controls=1
[ 6250.550593] SELinux:  policy capability open_perms=1
[ 6250.550598] SELinux:  policy capability extended_socket_class=1
[ 6250.550603] SELinux:  policy capability always_check_network=0
[ 6250.550608] SELinux:  policy capability cgroup_seclabel=1
[ 6250.550613] SELinux:  policy capability nnp_nosuid_transition=1
[ 6250.550618] SELinux:  policy capability genfs_seclabel_symlinks=1
[\UO294255@linux /]# _
```

Iniciamos ahora desde otra terminal con el usuario asuser y hacemos lo siguiente:

```
[asuser@linux ~]$ ls
[asuser@linux ~]$ mkdir public_html
[asuser@linux ~]$ cd public_html
[asuser@linux public_html]$ uo294255
```

Ahí, creamos un fichero llamado index.html y lo editamos con algo básico:

```
<!DOCTYPE html>
<html>
<head>
        <meta charset="utf-8">
        <title>Página de asuser</title>
</head>
<body>
        <h1>Asuser homepage</h1>
</body>
</html>
```

Por último, damos permiso a la carpeta public_html:

```
[asuser@linux public_html]$ cd ..
[asuser@linux ~]$ chmod 755 -R public_html
[asuser@linux ~]$
```

Volvemos a la otra terminal y reiniciamos el servicio httpd:

```
[\UO294255@linux /]# systemctl restart httpd
```

Y probamos a entrar al dominio desde la máquina W10 con la siguiente url:


Ahora cada usuario Linux podrá contar con tu página personal.

# 3. Configuración del servidor apache

## 3.a Definición de la Ubicación del Documento Raíz

Para empezar a configurar el servidor apache, crearemos una nueva ubicación para la pagina web. Creamos la carpeta /as/web y copiamos en ella el fichero index.html:

```
[\UO294255@linux /]# sudo mkdir -p /as/web
[\UO294255@linux /]# cd /as/web/
[\UO294255@linux web]# sudo cp /var/www/html/index.html /as/web
[\UO294255@linux web]# ls
index.html
[\UO294255@linux web]#
```

Luego modificaremos el DocumentRoot en el archivo de configuración de apache:

```
[\UO294255@linux web]# sudo vi /etc/httpd/conf/httpd.conf_
```

```
# below.
#


#
# DocumentRoot: The directory out of which you will serve your
# documents. By default, all requests are taken from this directory, bu
# symbolic links and aliases may be used to point to other locations.
#
DocumentRoot "/var/www/html"

#
# Relax access to content within /var/www.
#
<Directory "/var/www">
    AllowOverride None
    # Allow open access:
    Require all granted
</Directory>
```

```
#
# DocumentRoot: The directory out of which you will serve your
# documents. By default, all requests are taken from this directory, but
# symbolic links and aliases may be used to point to other locations.
#
DocumentRoot "/as/web"

#
# Relax access to content within /var/www.
#
```

También sustituiremos la sección por lo siguiente:

```
#
# Relax access to content within /var/www.
#
<Directory "/var/www">
    AllowOverride None
    # Allow open access:
    Require all granted
</Directory>
```

```
#
<Directory "/as/web">
    Options FollowSymLinks
    AllowOverride None
    Require all granted
</Directory>
```

Guardamos los cambios del fichero, y reiniciamos apache para que se apliquen:

```
[\UO294255@linux web]#systemctl restart httpd
[\UO294255@linux web]#
```

Ahora, asignaremos el contexto httpd_sys_content_t a través de chcon:

```
[\UO294255@linux /]# sudo chcon -R -h -t httpd_sys_content_t /as/web
[\UO294255@linux /]# _
```

## Personalización de Directivas de Servidor y Puerto

Ahora, vamos a modificar las directivas ServerAdmin y ServerName de acuerdo con tu email y con el nombre www.midominio.local, para ello, modificamos el mismo archivo de configuración que modificamos anteriormente:

```
# as error documents.  e.g. admin@your-domain.co
#
ServerAdmin root@localhost

#
# ServerName gives the name and port that the se
```

```
# as error documents.  e.g. admin@your-domain.com
#
ServerAdmin uo294255@uniovi.es

#
```

```
# If your host doesn't have a registered DNS name,
#
#ServerName www.example.com:80
_
#
```

```
# If your host doesn't have a registered DNS name, enter its IP
#
ServerName www.midominio.local

#
```

Posteriormente, hacemos que el servidor escuche en el puerto 9999:

```
#
#Listen 12.34.56.78:80
Listen 80

#
```

```
#
#Listen 12.34.56.78:80
Listen 9999

#
```

Sin olvidarnos de:

```
# If your host doesn't have a registered DN
#
ServerName www.midominio.local:9999

#
```

Con esto, el puerto estaría cambiado, reiniciamos el servidor y lo comprobamos en la maquina W10: (Seguramente sea necesario primero activar el puerto desde el firewall)

```
[\UO294255@linux /]# sudo firewall-cmd --zone=internal --permanent --add-port=9999/tcp
success
[\UO294255@linux /]# sudo firewall-cmd --reload
success
```

Antes de pasar a la siguiente parte, dejamos el servidor escuchando nuevamente al puerto 80.

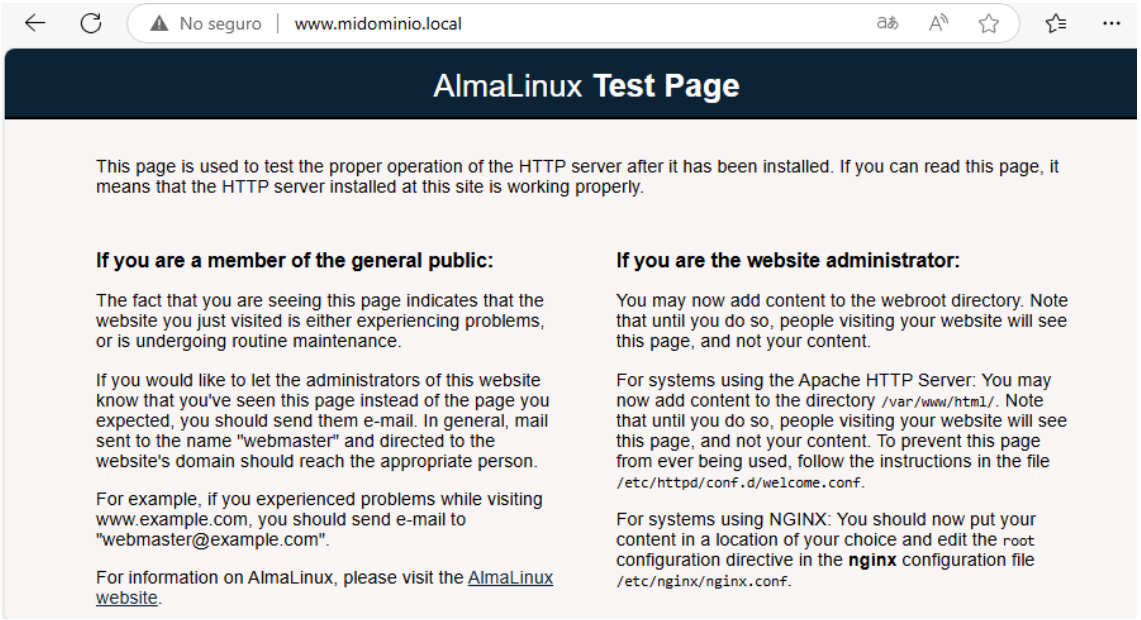
## Gestión de Repositorios y Páginas de Error

Renombramos el archivo “index.html” a “índice.html”:

```
[\UO294255@linux /]# sudo mv /as/web/index.html /as/web/índice.html
[\UO294255@linux /]#
```

Recargamos la página para que se borre la cache y consulta de nuevo la dirección www.midominio.local ¿Qué ocurre?

Que se muestra la pagina default puesto a que no encuentra un fichero “index.html”.




Edita el fichero /etc/httpd/conf.d/welcome.conf y comenta todas las líneas. Con ello desactivamos la presentación de la página de “Almalinux Test Page” para el caso que no encuentre el fichero index.html:

```
[\UO294255@linux /]# sudo vi /etc/httpd/conf.d/welcome.conf
```

```
#
# This configuration file enables the default "Welcome" page if there
# is no default index page present for the root URL.  To disable the
# Welcome page, comment out all the lines below.
#
# NOTE: if this file is removed, it will be restored on upgrades.
#
#<LocationMatch "^/+$">
#    Options -Indexes
#    ErrorDocument 403 /.noindex.html
#</LocationMatch>
#
#<Directory /usr/share/httpd/noindex>
#    AllowOverride None
#    Require all granted
#</Directory>
#
#Alias /.noindex.html /usr/share/httpd/noindex/index.html
#Alias /poweredby.png /usr/share/httpd/icons/apache_pb3.png
#Alias /system_noindex_logo.png /usr/share/httpd/icons/system_noindex_logo.png
~
```

Restaura el servicio httpd y vuelve a recargar la página. ¿Qué ocurre?

Pues que ahora no encuentra ni información para mostrar una página de bienvenida por defecto.


Modificamos ahora la sección que modificamos antes para /as/web:

```
# Relax access to content within /var/
#
<Directory "/as/web">
    Options FollowSymLinks
    AllowOverride None
    Require all granted
</Directory>

# Further relax access to the default
```

```
#
<Directory "/as/web">
    Options Indexes FollowSymLinks
    AllowOverride None
    Require all granted
</Directory>
```

Y reiniciamos el servicio. Con eso, probamos nuevamente a visualizar el contenido:


Ahora haremos un acceso a un fichero que no existe en nuestro dominio para ver lo que ocurre en /var/log/httpd/access_log:

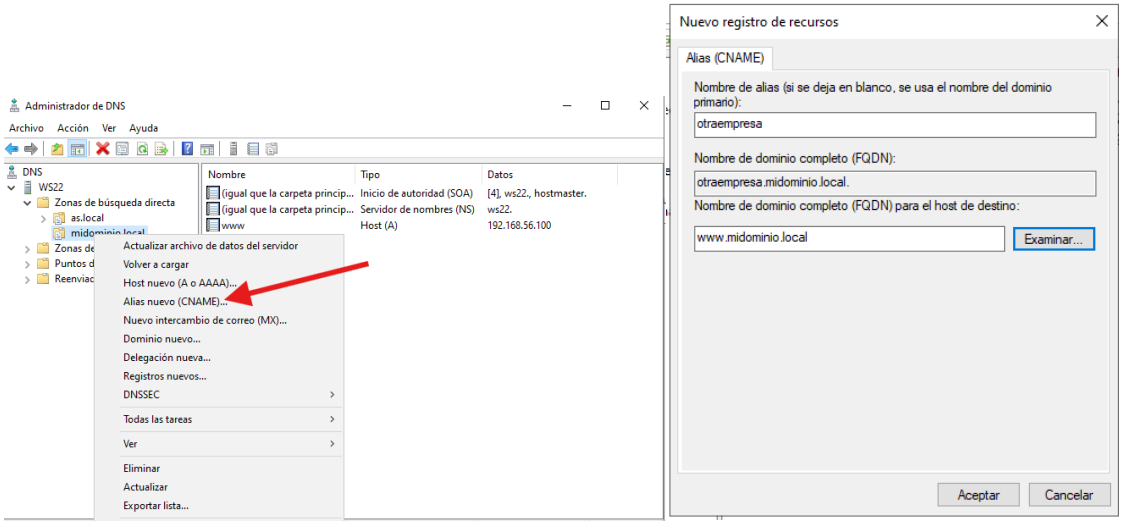
Vemos el fichero en cuestión:

```
[\UO294255@linux httpd]# tail -f access_log
192.168.56.110 - - [19/Mar/2025:19:25:16 +0100] "GET /icons/poweredby.png HTTP/1.1" 200 4194 "http://www.midominio.local/" "Mozilla/5.0 (Windows NT 10.0; Win64;
 x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/134.0.0.0 Safari/537.36 Edg/134.0.0.0"
192.168.56.110 - - [19/Mar/2025:19:29:07 +0100] "GET / HTTP/1.1" 403 199 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) C
hrome/134.0.0.0 Safari/537.36 Edg/134.0.0.0"
192.168.56.110 - - [19/Mar/2025:19:29:58 +0100] "-" 408 - "-" "-"
192.168.56.110 - - [19/Mar/2025:19:32:48 +0100] "GET / HTTP/1.1" 200 698 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) C
hrome/134.0.0.0 Safari/537.36 Edg/134.0.0.0"
192.168.56.110 - - [19/Mar/2025:19:32:48 +0100] "GET /icons/blank.gif HTTP/1.1" 200 148 "http://www.midominio.local/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
 AppleWebKit/537.36 (KHTML, like Gecko) Chrome/134.0.0.0 Safari/537.36 Edg/134.0.0.0"
192.168.56.110 - - [19/Mar/2025:19:32:48 +0100] "GET /icons/text.gif HTTP/1.1" 200 229 "http://www.midominio.local/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/134.0.0.0 Safari/537.36 Edg/134.0.0.0"
192.168.56.110 - - [19/Mar/2025:19:34:24 +0100] "GET /uo294255 HTTP/1.1" 404 196 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/134.0.0.0 Safari/537.36 Edg/134.0.0.0"
192.168.56.110 - - [19/Mar/2025:19:35:16 +0100] "-" 408 - "-" "-"
192.168.56.110 - - [19/Mar/2025:19:35:30 +0100] "GET /uo294255.html HTTP/1.1" 404 196 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/134.0.0.0 Safari/537.36 Edg/134.0.0.0"
192.168.56.110 - - [19/Mar/2025:19:36:22 +0100] "-" 408 - "-" "-"
```

# 4. Implementación de Hosts Virtuales

La definición de hosts virtuales “basado en nombres” permite agrupar en un mismo servidor con una sola IP las páginas de diferentes empresas. En la petición de servicio del protocolo http se incluye el nombre del host con lo que el servidor puede diferenciar las diferentes páginas que ha de mostrar.

Por lo tanto, daremos de alta en el servidor DNS un nuevo alias para www.midominio.local llamado otraempresa.midominio.local:




Comprobamos:

```
[\UO294255@linux /]# nslookup otraempresa.midominio.local
Server:         192.168.56.101
Address:        192.168.56.101#53

otraempresa.midominio.local     canonical name = www.midominio.local.
Name:   www.midominio.local
Address: 192.168.56.100

[\UO294255@linux /]#
```

Crea dos hosts virtuales, uno para www.midominio.local y otro para otraempresa.midominio.local con directorios raíz respectivos /as/web/www y /as/web/otraempresa; indícalo en las directivas correspondientes. Copia el index.html anterior y cambia su contenido (el cuerpo del html) para que se muestre “Pagina de Otra Empresa”.

Creamos los directorios:

```
[\UO294255@linux /]# sudo mkdir -p /as/web/www
[\UO294255@linux /]# sudo mkdir -p /as/web/otraempresa
[\UO294255@linux /]# _
```

Y copiamos el archivo “index.html” en ambos:

```
[\UO294255@linux /]# sudo cp /as/web/index.html /as/web/www/index.html
[\UO294255@linux /]# sudo cp /as/web/index.html /as/web/otraempresa/index.html
[\UO294255@linux /]# _
```

Modificamos el de /otraempresa/ para que muestre algo significativo:

```
<!DOCTYPE html>
<head>
        <meta charset="utf-8" />
        <title>Servidor AS</title>
</head>
<body>
        <h1>Administración de Sistemas de otra empresa</h1>
        <h2>UO294255</h2>
</body>
</html>
~
```

Por último, creamos el siguiente archivo:

```
[\UO294255@linux otraempresa]# sudo vi /etc/httpd/conf.d/virtualhosts.conf_
```

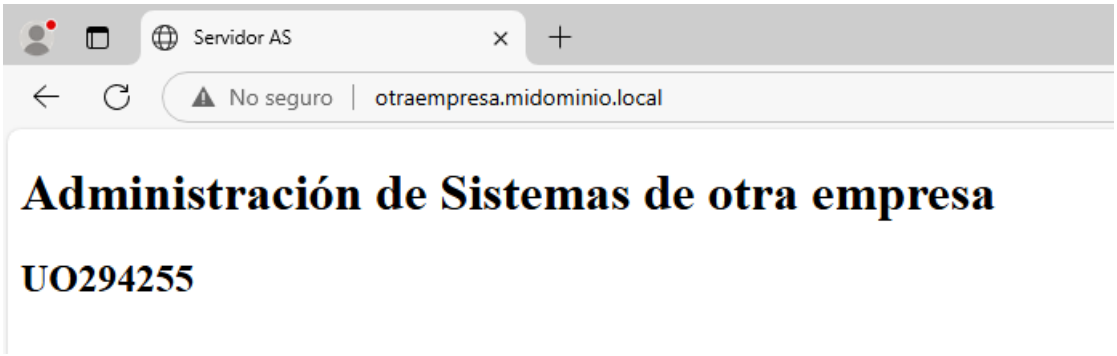
```
<VirtualHost *:80>
        ServerName www.midominio.local
        DocumentRoot /as/web/www
        <Directory "/as/web/www">
                Options FollowSymLinks
                AllowOverride None
                Require all granted
        </Directory>
        ErrorLog logs/www.midominio.local-error_log
        CustomLog logs/www.midominio.local-access_log common
</VirtualHost>_
~
```

Y luego este:

```
<VirtualHost *:80>
        ServerName www.midominio.local
        DocumentRoot /as/web/www
        <Directory "/as/web/www">
                Options FollowSymLinks
                AllowOverride None
                Require all granted
        </Directory>
        ErrorLog logs/www.midominio.local-error_log
        CustomLog logs/www.midominio.local-access_log common
</VirtualHost>
<VirtualHost *:80>
        ServerName otraempresa.midominio.local
        DocumentRoot /as/web/otraempresa
        <Directory "/as/web/otraempresa">
                Options FollowSymLinks
                AllowOverride None
                Require all granted
        </DIrectory>
        ErrorLog logs/otraempresa.midominio.local-error_log
        CustomLog logs/otraempresa.midominio.local-access_log common
</VirtualHost>
```

Reiniciaremos apache, y accederemos a la página otraempresa.midominio.local. ¿Qué aparece en el navegador?

Aparece su propia sección:




# 5. Autenticación y Seguridad

Vamos a configurar un acceso autorizado para la web “otraempresa.midominio.local”.

Para eso, añadimos la directiva “AllowOverride AuthConfig” en la sección Directory de /as/web/otraempresa, y reiniciamos el servicio para aplicar los cambios:

```
[\UO294255@linux ~]# vi /etc/httpd/conf.d/virtualhosts.conf
```

```
<VirtualHost *:80>
        ServerName otraempresa.midominio.local
        DocumentRoot /as/web/otraempresa
        <Directory "/as/web/otraempresa">
                Options FollowSymLinks
                AllowOverride AuthConfig_
                Require all granted
        </DIrectory>
        ErrorLog logs/otraempresa.midominio.local-error_log
        CustomLog logs/otraempresa.midominio.local-access_log common
</VirtualHost>
~
```

Luego, creamos en el directorio raíz de ese mismo host un archivo llamado .htaccess con el siguiente contenido:

```
[\UO294255@linux ~]# mkdir -p /as/web/otraempresa
[\UO294255@linux ~]# vi /as/web/otraempresa/.htaccess_
```

```
AuthType Basic
AuthName "Area Restringida"
AuthUserFile /etc/httpd/password.file
AuthGroupFile /dev/null
Require valid-user_
~
```

```
"/as/web/otraempresa/.htaccess" [New] 5L, 124B written
[\UO294255@linux ~]#
```

Para comprobar que algo de lo que hicimos funcionó para algo, intentaremos acceder a la página web:


Debido a que no somos un usuario autorizado.

Para solucionarlo, vamos a agregar un par de usuarios de la siguiente manera:
Creamos el archivo de contraseñas y añadimos al primer usuario:

```
[\U02942550@linux ~]# htpasswd -c /etc/httpd/password.file usuario1
New password:
Re-type new password:
Adding password for user usuario1
```

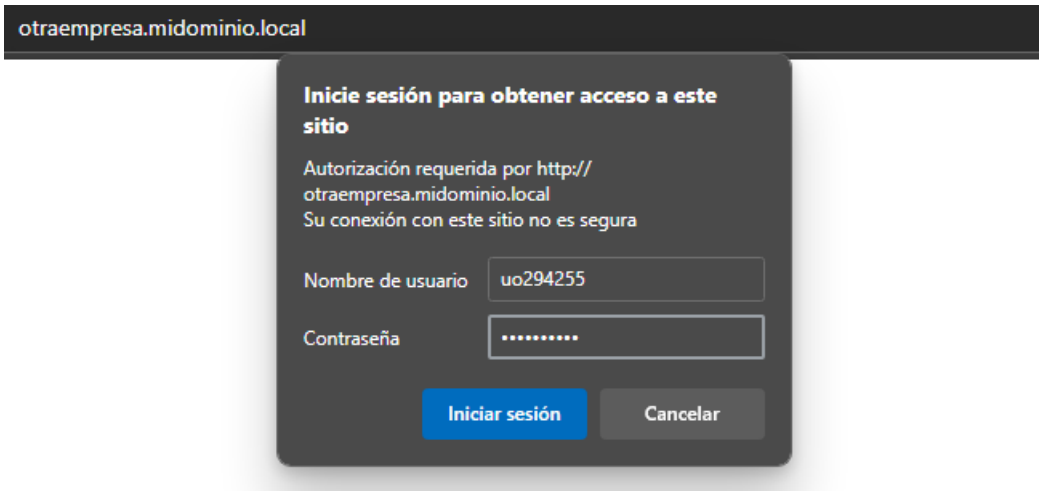
Ahora añadimos el segundo usuario:

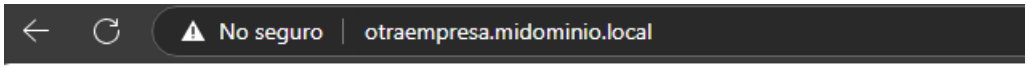
```
[\U02942550@linux ~]# htpasswd -c /etc/httpd/password.file usuario2
New password:
Re-type new password:
Adding password for user usuario2
```

Ahora será de vital importancia, editar los permisos de ambos archivos para que el servicio de Apache los pueda leer:

```
[\U02942550@linux ~]# chmod 640 /etc/httpd/password.file
[\U02942550@linux ~]# chown apache:apache /etc/httpd/password.file
```

Con todo esto configurado, reiniciamos el servicio y volveremos a intentar acceder a la página web:




# Administración de Sistemas de otra empresa

## UO294255

# 6. Configuración del Servidor Proxy Squid

Squid es un servidor proxy para web con caché. Es una de las aplicaciones más populares y de referencia para esta función, software libre publicado bajo licencia GPL.

Para instalarlo, ejecutamos la siguiente instrucción:

```
[\UO294255@linux ~]# dnf install squid
```

Una vez instalado el servidor, lo iniciamos y configuramos su arranque automático mediante:

```
[\UO294255@linux ~]# systemctl start squid
[\UO294255@linux ~]# systemctl enable squid
Created symlink /etc/systemd/system/multi-user.target.wants/squid.service → /usr/lib/systemd/system/squid.service.
[  842.674693] systemd-rc-local-generator[3601]: /etc/rc.d/rc.local is not marked executable, skipping.
[\UO294255@linux ~]# _
```

Podemos revisar su estado mediante la siguiente secuencia:

```
[\UO294255@linux ~]# systemctl status squid
● squid.service - Squid caching proxy
     Loaded: loaded (/usr/lib/systemd/system/squid.service; enabled; preset: disabled)
     Active: active (running) since Thu 2025-03-20 10:03:09 CET; 2min 26s ago
       Docs: man:squid(8)
   Main PID: 3585 (squid)
      Tasks: 3 (limit: 17424)
     Memory: 14.4M
        CPU: 52ms
     CGroup: /system.slice/squid.service
             ├─3585 /usr/sbin/squid --foreground -f /etc/squid/squid.conf
             ├─3587 "(squid-1)" --kid squid-1 --foreground -f /etc/squid/squid.conf
             └─3588 "(logfile-daemon)" /var/log/squid/access.log

mar 20 10:03:09 linux.as.local systemd[1]: Starting Squid caching proxy...
mar 20 10:03:09 linux.as.local squid[3585]: Squid Parent: will start 1 kids
mar 20 10:03:09 linux.as.local squid[3585]: Squid Parent: (squid-1) process 3587 started
mar 20 10:03:09 linux.as.local systemd[1]: Started Squid caching proxy.
```

Ahora vamos a configurar la ACL (control de acceso) editando el fichero de configuración `/etc/squid/squid.conf`:

```
[\UO294255@linux ~]# sudo vi /etc/squid/squid.conf
```

La sección acl localnet define los rangos ip de las posibles redes locales en los que se puede encontrar el servidor. Comprobamos que exista una dirección similar a “acl localnet src 192.168.56.0/24” o en el mejor caso, una mejor:

```
#
# Recommended minimum configuration:
#

# Example rule allowing access from your local networks.
# Adapt to list your (internal) IP networks from where browsing
# should be allowed
acl localnet src 0.0.0.1-0.255.255.255  # RFC 1122 "this" network (LAN)
acl localnet src 10.0.0.0/8             # RFC 1918 local private network (LAN)
acl localnet src 100.64.0.0/10          # RFC 6598 shared address space (CGN)
acl localnet src 169.254.0.0/16         # RFC 3927 link-local (directly plugged) machines
acl localnet src 172.16.0.0/12          # RFC 1918 local private network (LAN)
acl localnet src 192.168.0.0/16         # RFC 1918 local private network (LAN)
acl localnet src fc00::/7               # RFC 4193 local private network range
acl localnet src fe80::/10              # RFC 4291 link-local (directly plugged) machines

acl SSL_ports port 443
acl Safe_ports port 80          # http
acl Safe_ports port 21          # ftp
acl Safe_ports port 443         # https
acl Safe_ports port 70          # gopher
acl Safe_ports port 210         # wais
acl Safe_ports port 1025-65535  # unregistered ports
acl Safe_ports port 280         # http-mgmt
acl Safe_ports port 488         # gss-http
acl Safe_ports port 591         # filemaker
acl Safe_ports port 777         # multiling http

#
# Recommended minimum Access Permission configuration:
#
# Deny requests to certain unsafe ports
http_access deny !Safe_ports

# Deny CONNECT to other than secure SSL ports
http_access deny CONNECT !SSL_ports

# Only allow cachemgr access from localhost
http_access allow localhost manager
http_access deny manager

# We strongly recommend the following be uncommented to protect innocent
# web applications running on the proxy server who think the only
# one who can access services on "localhost" is a local user
#http_access deny to_localhost

#
# INSERT YOUR OWN RULE(S) HERE TO ALLOW ACCESS FROM YOUR CLIENTS
#
"/etc/squid/squid.conf" 74L, 2488B
```

Y lo modificamos, debido a que la actual, aunque es más permisivo debido al mayor número de direcciones que abarca, es menos seguro:

```
acl localnet src 0.0.0.1-0.255.255.255  # RFC 1122 "this" network (LAN)
acl localnet src 10.0.0.0/8             # RFC 1918 local private network (LAN)
acl localnet src 100.64.0.0/10          # RFC 6598 shared address space (CGN)
acl localnet src 169.254.0.0/16         # RFC 3927 link-local (directly plugged) machines
acl localnet src 172.16.0.0/12          # RFC 1918 local private network (LAN)
acl localnet src 192.168.56.0/24        # RFC 1918 local private network (LAN)
acl localnet src fc00::/7               # RFC 4193 local private network range
acl localnet src fe80::/10              # RFC 4291 link-local (directly plugged) machines
```

Ahora, des comentamos la línea donde se define el almacenamiento de la memoria caché cache_dir:

```
# Uncomment and adjust the following to add a disk cache directory.
#cache_dir ufs /var/spool/squid 100 16 256
```

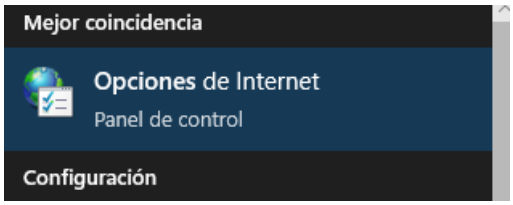
```
# Uncomment and adjust the following to add a disk cache directory.
cache_dir ufs /var/spool/squid 100 16 256
```

Para acabar, reiniciamos el servicio squid para aplicar los cambios como haciamos con el servicio httpd y lo añadimos al cortafuegos:

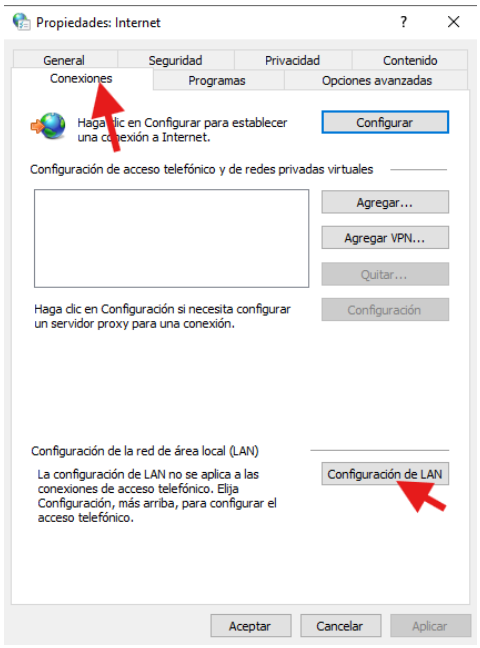
```
[\UO294255@linux ~]# systemctl restart squid
[\UO294255@linux ~]# firewall-cmd --zone=internal --add-service=squid --permanent
success
[\UO294255@linux ~]# firewall-cmd --reload
success
```

Ahora, desde la maquina Windows, configuramos el navegador para conectarse a través del nuevo proxy de Linux y comprobamos en el servidor que las conexiones se realizan ahora a través del proxy.

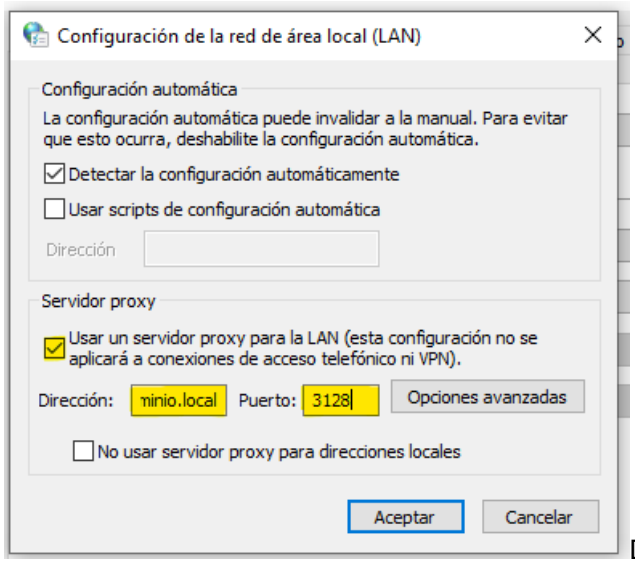
Para ello, en la máquina W10 vamos a “Opciones de Internet” desde la barra de búsqueda:




Le damos a “Conexiones” y posteriormente a “Configuración de LAN”:



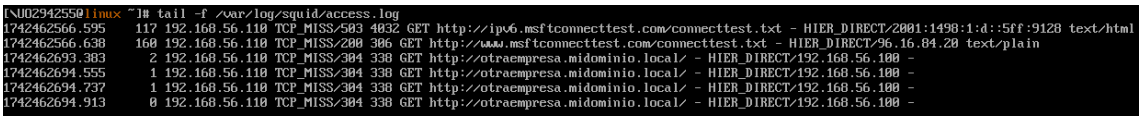

Y configuramos lo siguiente:




Dirección: “www.midominio.local”

Por último, “Aceptar”.

Ahora todo el trafico web pasara por el servidor proxy squid antes de salir a internet, teniendo nosotros registro de ello. Para verlo, desde la maquina W10, hacemos una búsqueda y veremos el resultado en el fichero /var/log/squid/access.log:




Al recargar la página de otraempresa.midominio varias veces, lo que muestra es eso.

Con esto concluye la realización de la práctica 7.