

WorldSkills 2024

39-IT Network Systems Administration

Spain

Moisés Tamaalit Martínez



<u>Linux</u>	5
<u>Montar repo ISO Debian</u>	5
<u>SSH</u>	5
<u>DHCP</u>	5
<u>DNS</u>	6
<u>Vistas DNS</u>	8
<u>LDAP</u>	10
<u>Securizando LDAP con TLS</u>	13
<u>Autenticar usuarios en Linux</u>	14
<u>Editar el archivo /etc/nsswitch.conf</u>	14
<u>Postfix y Dovecot</u>	16
<u>Autenticación contra LDAP</u>	17
<u>Postfix</u>	18
<u>Dovecot</u>	19
<u>dovecot</u>	22
<u>OpenVPN</u>	24
<u>Generación de certificados</u>	24
<u>Easy-RSA</u>	24
<u>OpenSSL (2º método)</u>	25
<u>Configuración del servidor</u>	27
<u>Configuración del cliente</u>	28
<u>Generar fichero cliente</u>	29
<u>Reglas básicas IPTABLES-VPN</u>	31
<u>HTTP</u>	33
<u>Nginx</u>	33
<u>Apache</u>	34
<u>Sitios personales (apache2)</u>	34
<u>SSL</u>	35
<u>Apache</u>	35
<u>haproxy</u>	36
<u>vsFTPd</u>	38
<u>radvd</u>	39
<u>VLAN</u>	39
<u>iptables</u>	39
<u>ICMP</u>	39
<u>NAT</u>	39
<u>DNAT (redirección de puertos)</u>	40
<u>nftables</u>	40
<u>SNAT en nftables</u>	41
<u>Bash</u>	42
<u>ansible</u>	43
<u>Playbook</u>	43
<u>Ejemplo LDAP clientes:</u>	45

<u>Ejemplo DNS entero:</u>	46
<u>Ejemplo Postfix entero:</u>	46
<u>Windows</u>	46
<u>Servicios básicos (DHCP, DNS, IIS)</u>	46
<u>NTP</u>	47
<u>Añadir certificado de CA</u>	47
<u>Active Directory Certificate Services</u>	48
<u>Perfiles móviles (roaming profiles)</u>	49
<u>DFS (Distributed File System)</u>	51
<u>GPOs (Objeto de directiva de grupo)</u>	52
<u>Ejemplo 00: Impedir y ocultar acceso a unidades de sistema</u>	52
<u>Ejemplo 01: Mapear unidades de red mediante GPO</u>	53
<u>Configurar recurso compartido en red</u>	53
<u>NAT con Windows (ugh)</u>	54
<u>VPN con Windows</u>	54
<u>Regla firewall</u>	57
<u>Añadir conexión VPN en Windows 10</u>	57
<u>Site to Site VPN</u>	57
<u>Ansible</u>	57
<u>Configurar Windows para usar ansible</u>	57
<u>Copias de seguridad de Windows Server</u>	57
<u>PowerShell</u>	58
<u>Habilitar ejecución de scripts</u>	58
<u>1. Creación de usuarios en local</u>	58
<u>1.1 Automatización de creación de usuarios</u>	58
<u>1.1.1 Contraseña en fichero</u>	58
<u>2. Creación de usuarios en Active Directory</u>	59
<u>2.1 Importación de usuarios masiva por CSV</u>	60
<u>Cisco</u>	61
<u>Can't switchport mode trunk</u>	61
<u>Ancho de banda de broadcast</u>	61
<u>VLANs</u>	61
<u>Voice VLAN</u>	61
<u>Deshabilitar DTP</u>	61
<u>Recuperar contraseña</u>	61
<u>Habilitar contraseña</u>	62
<u>NTP/Syslog/SNMP</u>	62
<u>SNMPv2</u>	62
<u>SNMPv3</u>	62
<u>SVI</u>	62
<u>LLDP y CDP</u>	62
<u>Acceso terminal o ssh</u>	63
<u>Copia seguridad y restauración de sistema IOS</u>	63
<u>Actualización IOS</u>	64

<u>VTP</u>	64
<u>Spanning-Tree</u>	64
<u>Port security (MAC)</u>	64
<u>DHCP server (router)</u>	65
<u>DHCP relay</u>	65
<u>PPP</u>	65
<u>ACLs</u>	65
<u>Standard:</u>	65
<u>Extended:</u>	66
<u>Named:</u>	66
<u>RIP</u>	66
<u>RIPng</u>	66
<u>BGP</u>	66
<u>EIGRP</u>	67
<u>FHRP</u>	67
<u>Diferentes protocolos</u>	68
<u>HSRP</u>	68
<u>Configuración HSRP</u>	68
<u>OSPF</u>	69
<u>Áreas</u>	69
<u>Configuración OSPFv2 (IPv4)</u>	71
<u>Tres formas de configurar OSPF</u>	72
<u>OSPFv3</u>	72
<u>NAT</u>	73
<u>Redirección de puertos (DNAT)</u>	74
<u>IPv6</u>	74
<u>Tipos de direcciones IPv6</u>	74
<u>Multicast</u>	74
<u>Anycast</u>	75
<u>Configuración IPv6 en Cisco</u>	75
<u>Enrutamiento estático IPv6</u>	76
<u>SLAAC</u>	76
<u>DHCPv6</u>	76
<u>MTU</u>	77
<u>VPN</u>	77
<u>Site-site</u>	77
<u>IPSEC</u>	77
<u>GRE</u>	78
<u>Client to site</u>	79
<u>Python</u>	79
<u>Librerías</u>	79

Linux

Montar repo ISO Debian

Una vez están los DVDs insertados:

```
apt-cdrom add
```

Con eso bastaría. Si da fallo, montar manualmente los DVD:

Se montan las ISOs con el siguiente comando:

(para ello habremos creado los directorios de montaje primero)

```
mount -o loop isos/template/iso/debian-11.8.0-amd64-DVD-1.iso /mnt/iso1
```

Para que APT se los trague, ponemos lo siguiente en el sources.list

```
deb [trusted=yes] file:/mnt/iso1 bullseye main contrib  
deb [trusted=yes] file:/mnt/iso2 bullseye main contrib  
deb [trusted=yes] file:/mnt/iso3 bullseye main contrib
```

SSH

No requiere explicación, pero bueno:

```
sudo apt update  
sudo apt install openssh-server  
  
systemctl enable sshd
```

Para conectarse usando como “puente” otra máquina se puede hacer mediante dos métodos:

- Comando

```
ssh -J user@server_bridge user@server
```

- Configuración en .ssh/config

En ese fichero se pone lo siguiente:

```
Host <ip/nombre/FQDN>  
    ProxyCommand ssh user@bridge_server -W %h:%p
```

DHCP

```
apt install isc-dhcp-server  
editor /etc/dhcp/dhcpd.conf (se procedería a editar, más abajo se indica
```

```
sintaxis y demás)
systemctl enable isc-dhcp-server
```

Al editar el fichero dhcpd.conf hay que tener en cuenta ciertas consideraciones.

- Al final de cada línea, hay que poner ":";
- Las redes se delimitan por rangos.
- En `/etc/default/isc-dhcp-server` hay que indicar las interfaces por las que escucha
- Hay que tener IP estáticas en los servidores (ya que enviarán y recibirán las peticiones por esas interfaces, y así el servicio identifica la IP que dará)

Ejemplo básico de configuración:

```
option domain-name "skills39.io";
option domain-name-servers 192.168.5.1;

default-lease-time 600;
max-lease-time 7200;

# The ddns-updates-style parameter controls whether or not the server will
# attempt to do a DNS update when a lease is confirmed. We default to the
# behavior of the version 2 packages ('none', since DHCP v2 didn't
# have support for DDNS.)
ddns-update-style none;

subnet 192.168.5.0 netmask 255.255.255.0 {
    range 192.168.5.127 192.168.5.250;
    option routers 192.168.5.254;
}
```

Para incluir ficheros externos:

```
include "/etc/dhcp/dhcpd.d/hosts.conf";
```

DNS

```
apt install bind9
```

En bind habrá que realizar configuraciones en varios ficheros:

PUNTO Y COMA AL FINAL DE CADA LÍNEA

- named.conf.options (aquí configuramos los forwarders principalmente, se puede configurar también ACLs)

```
forwarders {  
    8.8.8.8;  
    8.8.4.4;  
};  
  
//=====  
// If BIND logs error messages about the root key being expire  
// you will need to update your keys. See https://www.isc.org  
//=====  
dnssec-validation auto;  
  
listen-on-v6 { any; };
```

- named.conf.local

```
zone "skills39.io" {  
    type master;  
    file "/var/lib/bind/db.skills39";  
    allow transfer { 192.168.60.251; };  
    also-notify { 192.168.60.251; };  
};  
  
zone "60.168.192.in-addr.arpa" {  
    type master;  
    file "/var/lib/bind/60.arpa";  
    allow transfer { 192.168.60.251; };  
    also-notify { 192.168.60.251; };  
};
```

Se configuran los dominios de búsqueda y **vistas**, revisar abajo.

- db.loquesea (configurar cada zona)

```

$TTL 604800

skills39.io.    IN      SOA     ns1.skills39.io. root.skills39.io. (
                  2023102201; Serial
                  604800    ; Refresh
                  86400     ; Retry
                  2419200   ; Expire
                  604800)   ; Negative cache TTL

; NS Records
@           IN      NS      ns1.skills39.io.
@           IN      NS      ns2.skills39.io.
ns1         IN      A       192.168.60.250
ns2         IN      A       192.168.60.251

; A records
pcl        IN      A       192.168.60.1

; CNAME records
www        IN      CNAME   ns1.skills39.io.
ftp         IN      CNAME   ns2.skills39.io.

```

Para comprobar la configuración: named-checkconf y named-checkzone

```

mydomain.com.    IN      NS      ns2.mydomain.com.
mydomain.com.    IN      MX      10 mail.mydomain.com.

```

Vistas DNS

En DNS se pueden crear vistas para responder sobre un dominio con distintas IPs según quién pregunta. Para ello, primero hay que crear ACLs con las redes según quién pregunta.

En named.conf.local:

```

acl interno { 192.168.60.0/24; };
acl externo { !interno; };

```

Una vez creadas las ACLs, se definirán las vistas con las zonas y sus opciones:

```

view interna {
    match-clients { interno; };

    zone "skills39.io" {
        type master;
        file "/var/lib/bind/db.skills39";
        allow-transfer { 192.168.60.251; };
        also-notify { 192.168.60.251; };
    };
    zone "60.168.192.in-addr.arpa" {
        type master;
        file "/var/lib/bind/60.arpa";
    };
}

```

```

        allow-transfer { 192.168.60.251; };
        also-notify { 192.168.60.251; };
    };
    include "/etc/bind/zones.rfc1918";
    include "/etc/bind/named.conf.default-zones";
};

view externa {
    match-clients { externo; }; # Se puede usar any como valor.
    zone "skills39.io" {
        type master;
        file "/var/lib/bind/db.skills39_ext";
        allow-transfer { 192.168.60.251; };
        also-notify { 192.168.60.251; };
    };
    zone "5.168.192.in-addr.arpa" {
        type master;
        file "/var/lib/bind/5.arpa";
        allow-transfer { 192.168.60.251; };
        also-notify { 192.168.60.251; };
    };
    include "/etc/bind/zones.rfc1918";
    include "/etc/bind/named.conf.default-zones";
};

```

Habrá que comentar del fichero named.conf las zonas por defecto, ya que al usar vistas, **TODAS LAS ZONAS** tendrán que estar dentro de alguna vista (podemos crear una tercera vista llamada “todas” en la que englobemos las dos vistas y las zonas a incluir).

```

include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
// include "/etc/bind/named.conf.default-zones";

```

LDAP

Instalar LDAP

```
apt-get install slapd ldap-utils slapd-contrib
```

Eliminar ficheros creados por defecto:

```
rm -r /var/lib/ldap/data.mdb y lock.mdb
```

Se configura:

```
dpkg-reconfigure slapd
```

Crear unidades organizativas en las que guardar los usuarios, grupos, etc:

ous.ldif:

```
dn: ou=Users,dc=moises,dc=lan
objectClass: organizationalUnit
ou: Users

dn: ou=Groups,dc=moises,dc=lan
objectClass: organizationalUnit
ou: Groups

dn: ou=asir,ou=Users,dc=moises,dc=lan
objectClass: organizationalUnit
ou: asir
```

Para añadirlo:

```
ldapadd -x -W -D cn=admin,dc=moises,dc=lan -f ous.ldif
```

Crear usuarios :

(Recuerda que los usuarios siempre tienen un grupo principal, por lo que hay que crear ambos)

john.ldif:

```
dn: cn=john,ou=Groups,dc=moises,dc=lan
objectClass: posixGroup
cn: john
gidNumber: 8000

dn: uid=john,ou=Users,dc=moises,dc=lan
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
uid: john
sn: doe
givenName: John
```

```
cn: John Doe
uidNumber: 8000
gidNumber: 8000
gecos: John Doe
loginShell: /bin/bash
homeDirectory: /home/john
```

Crear grupos (y añadir miembros):

```
dn: cn=profesores,ou=Groups,dc=moises,dc=lan
objectClass: posixGroup
cn: profesores
gidNumber: 8000
memberuid: profesor00 # Añadir miembros
memberuid: profesor01
```

Crear usuario readonly: (para aplicaciones que autentiquen, como correo)

```
dn: cn=readonly,dc=murasame,dc=xyz
changetype: add
objectClass: simpleSecurityObject
objectClass: organizationalRole
userPassword: {SSHA}cgIdGQBaDaSeIy4S5bZ4/7t9dEcZjfIh
description: LDAP read only user
```

```
ldapadd -x -W -D cn=admin,dc=moises,dc=lan -f readonly.ldif
```

Añadir los ficheros:

```
ldapadd -x -W -D cn=admin,dc=moises,dc=lan -f john.ldif
```

Generar contraseña (para algunos casos):

```
slappasswd -h {SSHA}
```

Cambiar contraseña:

```
ldappasswd -S -x -W -H ldapi:/// -D 'cn=admin,dc=murasame,dc=xyz' -x
"uid=john,dc=murasame,dc=xyz"
```

Copia de seguridad:

```
systemctl stop slapd # No es 100% necesario.
slapcat -b cn=config > config_bak.ldif
slapcat -b dc=moises,dc=lan > moises.lan_bak.ldif
systemctl start slapd.service # No es 100% necesario.
```

Restaurar:

```
systemctl stop slapd
rm -rf /etc/ldap/slapd.d/* /var/lib/ldap/*
```

```
slapadd -F /etc/ldap/slapd.d -b 'cn=config' -l PATH/config_bak.ldif
slapadd -F /etc/ldap/slapd.d -b dc=moises,dc=lan -l
PATH/moises.lan_bk.ldif
chown -R openldap.openldap /etc/ldap/slapd.d/
chown -R openldap.openldap /var/lib/ldap/
systemctl start slapd.service
```

Securizando LDAP con TLS

Para ello primero tenemos que generar el certificado TLS firmando con la CA ([consultar sección OpenSSL](#))

Comprobamos si hay algo de TLS en la configuración de LDAP:

```
slapcat -b "cn=config" | grep 'TLS'
```

Copiamos desde el servidor con la CA los ficheros ldap.crt , ldap.key y rootCA.crt.

```
mkdir /etc/ldap/{cacerts,certs}
cp ldap.* /etc/ldap/certs
cp rootCA.crt /etc/ldap/cacerts
chmod 750 /etc/ldap/{certs,cacerts}
chmod 640 -R /etc/ldap/{certs/*,cacerts/*}
chown -R openldap: /etc/ldap/{certs/cacerts}
```

Nos aseguramos de los permisos con `ls -l`

Creamos un fichero tls.ldif

```
dn: cn=config
add: olcTLSCACertificateFile
olcTLSCACertificateFile: /etc/ldap/cacerts/rootCA.crt
-
add: olcTLS CertificateFile
olcTLS CertificateFile: /etc/ldap/certs/ldap.crt
-
add: olcTLS CertificateKeyFile
olcTLS CertificateKeyFile: /etc/ldap/certs/ldap.key
```

Para aplicar el cambio:

```
ldapmodify -Y EXTERNAL -H ldapi:/// -f tls.ldif
```

Autenticar usuarios en Linux

Cliente:

```
sudo apt-get install libpam-ldapd libnss-ldap nslcd nsqd ldap-utils
```

Editar el archivo /etc/nsswitch.conf

```
# /etc/nsswitch.conf
#
# Example configuration of GNU Name Service Switch functionality.
# If you have the `glibc-doc-reference` and `info` packages installed,
try:
# `info Libc "Name Service Switch"' for information about this file.

passwd:          files systemd ldap sss
group:           files systemd ldap sss
shadow:          files ldap sss
gshadow:         files ldap

hosts:            files dns
networks:        files

protocols:       db files
services:        db files sss
ethers:          db files
rpc:              db files

netgroup:         nis sss
automount:       sss
```

Editar /etc/nslcd.conf

```
# The user and group nslcd should run as.
uid nslcd
gid nslcd

# The location at which the LDAP server(s) should be reachable.
uri ldap://ldap.moises.lan/

# The search base that will be used for all queries.
base dc=moises,dc=lan
# base ou=Users,dc=moises,dc=lan

# The LDAP protocol version to use.
#ldap_version 3

# The DN to bind with for normal lookups.
```

```
binddn cn=readonly,dc=moises,dc=lan
bindpw contraseña

# The DN used for password modifications by root.
#rootpwmoddn cn=admin,dc=example,dc=com

# SSL options
ssl start_tls
tls_reqcert demand
tls_cacertfile /etc/ssl/certs/ca-certificates.crt

# The search scope.
#scope sub
```

Editar /etc/pam.d/common-session:

Añadir al final esto:

```
session optional      pam_mkhomedir.so skel=/etc/skel umask=077
```

Y ya. Si hubiera que hacer cambios: dpkg-reconfigure nslcd

Postfix y Dovecot

Un FQDN de la máquina puede ser mail.fqdn

En el DNS añadimos un registro CNAME (o A o AAAA) a la máquina mail y un registro MX

```
mail           IN      CNAME   ns1.skills39.io.  
;  
; MX records  
mail.skills39.io.    IN      MX      10      mail.skills39.io.  
~
```

```
apt install postfix postfix-ldap postfix-doc
```

Seleccionamos Sitio de Internet, el FQDN con mail (o podemos poner solo el dominio), y un **dpkg-reconfigure** al canto; las dos primeras igual, luego la siguiente al usuario admin (no root) de la máquina, todos los nombres de la máquina, no a las actualizaciones síncronas, poner las IPs locales, límite a 0 y lo demás al gusto.

Crear clave SSL para Postfix (con Subject Alt Name) y hacer enlace simbólico a /etc/postfix/ssl (crear carpeta)

```
ln -s mail.key /etc/postfix/ssl/  
ln -s mail.crt /etc/postfix/ssl/  
ln -s ca.crt /etc/postfix/ssl/  
  
vim /etc/postfix/main.cf
```

```
# TLS parameters  
smtpd_tls_cert_file=/etc/postfix/ssl/mail.crt  
smtpd_tls_key_file=/etc/postfix/ssl/mail.key  
smtpd_tls_security_level=may  
  
smtp_tls_CApth=/etc/postfix/ssl/ca.crt  
smtp_tls_security_level=may  
smtp_tls_session_cache_database = btree:${data_directory}/smtp_scache
```

```
sudo postconf -e 'smtp_tls_security_level = may'  
sudo postconf -e 'smtpd_tls_security_level = may'  
sudo postconf -e 'smtpd_tls_auth_only = no'  
sudo postconf -e 'smtp_tls_note_starttls_offer = yes'  
sudo postconf -e 'smtpd_tls_key_file = /etc/postfix/ssl/mailserver.key'  
# ARRIBA  
sudo postconf -e 'smtpd_tls_cert_file = /etc/postfix/ssl/mailserver.crt'  
#ARRIBA  
sudo postconf -e 'smtpd_tls_CApfile = /etc/postfix/ssl/cacert.pem' #  
ARRIBA
```

```
sudo postconf -e 'smtpd_tls_loglevel = 1'
sudo postconf -e 'smtpd_tls_received_header = yes'
sudo postconf -e 'smtpd_tls_session_cache_timeout = 3600s'
sudo postconf -e 'tls_random_source = dev:/dev/urandom'
smtpd_use_tls = yes
```

Para habilitar el **puerto 587 (STARTTLS)** en master.cf descomentar esta línea

```
submission inet n - y - - smtpd
```

Para probar funcionalidad:

```
telnet localhost 587
ehlo mail
mail from: address
rcpt to: address
data
sdadas
.
quit
```

Autenticación contra LDAP

```
useradd vmail
mkdir /home/vmail
mkdir /home/vmail/domains
chown -R vmail:vmail /home/vmail
```

En base.ldif:

```
dn: o=mail,dc=skills39,dc=io
objectClass: organization
objectClass: top
o: hosting description: Mail Organization

# Read only account
dn: cn=vmail,o=mail,dc=skills39,dc=io
objectClass: simpleSecurityObject
objectClass: organizationalRole
cn: vmail
userPassword: {MD5}M267sheb6qc0Ck8WIPOvQA==
description: Read only account
```

```
ldapmodify -a -D cn=admin,dc=example,dc=tld -W -f base.ldif
```

Postfix

En main.cf

```
virtual_mailbox_base = /home/vmail # (0 /var/mail)
virtual_minimum_uid = (UID de vmail)

#Virtual User

virtual_mailbox_maps = ldap:/etc/postfix/ldap-accounts.cf

vuser_server_host = 127.0.0.1
vuser_search_base = ou=people,dc=skills39,dc=io
vuser_query_filter =
(&(mail=%s)(!(quota=-1))(objectClass=CourierMailAccount))
vuser_result_attribute = mailbox
vuser_bind = no

#Virtual User uid

virtual_uid_maps = static:(uid de vmail)

uidldap_server_host = 127.0.0.1
uidldap_search_base = ou=people,dc=skills39,dc=io
uidldap_query_filter =
(&(mail=%s)(!(quota=-1))(objectClass=CourierMailAccount))
uidldap_result_attribute = uidNumber
uidldap_bind = no

#Virtual User gid

virtual_gid_maps = static:(gid de vmail)

gidldap_server_host = 127.0.0.1
gidldap_search_base = ou=people,dc=skills39,dc=io
gidldap_query_filter =
(&(mail=%s)(!(quota=-1))(objectClass=CourierMailAccount))
gidldap_result_attribute = gidNumber
gidldap_bind = no
```

En ldap-accounts.cf:

```
server_host = localhost
server_port = 389
version = 3
bind = yes
start_tls = no
bind_dn = cn=vmail,o=mail,dc=skills39,dc=io
bind_pw = readonly
```

```
search_base = o=mail,dc=skills39,dc=io
scope = sub
query_filter =
(&(&(objectClass=VirtualMailAccount)(mail=%s))(forwardActive=FALSE)(acco
untActive=TRUE)(delete=FALSE))
result_attribute = mailbox
```

Dovecot

En **/etc/dovecot/dovecot-ldap.conf.ext** poner el host LDAP:

```
hosts = ldap.skills39.io
# Otra opción es la de abajo
uris = ldapi:///
```

Configurar usuario (previamente añadido) para gestionar el acceso al directorio

```
dn = cn=vmail,dc=skills39,dc=io
dnpass = contraseña
```

Poner las siguientes opciones:

```
auth_bind = yes
base = 'o=users,dc=skills39,dc=io'
auth_bind_userdn = cn=%u,ou=People,dc=skills39,dc=io # Esto si Los
usuarios están dentro de un área de LDAP
user_attrs = homeDirectory=home,uidNumber=uid,gidNumber=gid # Decirle a
dovecot los atributos que son
user_filter = (&(objectClass=posixAccount)(uid=%u))
pass_attrs = uid=user
pass_filter = (&(objectClass=posixAccount)(uid=%u))
```

```
passdb {
    driver = ldap
    args = /etc/dovecot/dovecot-ldap.conf.ext
}
userdb {
    driver = ldap
    args = /etc/dovecot/dovecot-ldap-userdb.conf.ext
}
```

And create the symlink:

```
ln -s /etc/dovecot/dovecot-ldap.conf.ext /etc/dovecot/dovecot-ldap-userdb.conf.ext
```

LDAP example

`dovecot.conf`:

```
passdb {
    driver = ldap
    args = /etc/dovecot/dovecot-ldap.conf.ext
}
userdb {
    driver = prefetch
}
# The userdb below is used only by LDA.
userdb {
    driver = ldap
    args = /etc/dovecot/dovecot-ldap.conf.ext
}
```

`dovecot-ldap.conf.ext`:

```
pass_attrs = uid=user, userPassword=password, \
    homeDirectory=userdb_home, uidNumber=userdb_uid, gidNumber=userdb_gid

# For LDA:
user_attrs = homeDirectory=home, uidNumber=uid, gidNumber=gid
```

Ejemplo dovecot.conf:

dovecot

```
apt install dovecot-core dovecot-imapd dovecot-ldap
```

Quitar /etc/dovecot/dovecot.conf y hacer uno con ese nombre con esto:

```
disable_plaintext_auth = no
mail_privileged_group = mail
mail_location = mbox:~/mail:INBOX=/var/mail/%u

userdb {
  driver = passwd
}

passdb {
  args = %s
  driver = pam
}

protocols = " imap"

namespace inbox {
  inbox = yes

  mailbox Trash {
    auto = subscribe # autocreate and autosubscribe the Trash mailbox
    special_use = \Trash
  }
  mailbox Sent {
    auto = subscribe # autocreate and autosubscribe the Sent mailbox
    special_use = \Sent
  }
}

service auth {
  unix_listener /var/spool/postfix/private/auth {
    group = postfix
    mode = 0660
    user = postfix
  }
}

ssl=required
ssl_cert=</etc/keys/mail.crt
ssl_key=</etc/keys/mail.key
```


OpenVPN

Se instala OpenVPN como cualquier paquete. Pero la configuración se dividirá en dos pasos: generación de certificados y claves públicas/privadas, y configuración de OpenVPN.

Generación de certificados

Easy-RSA

Para generar certificados con easy-rsa, lo primero es tenerlo instalado. Luego, para hacer más amena la instalación, podemos copiar del directorio /usr/share/easy-rsa/ todos sus contenidos a una carpeta aparte para generar los certificados correspondientes al servicio.

Se edita el fichero vars de esta carpeta (si no existe, se crea) con las siguientes:

```
set_var EASYRSA_REQ_COUNTRY "ES"
set_var EASYRSA_REQ_PROVINCE "MURCIA"
set_var EASYRSA_REQ_CITY "MURCIA"
set_var EASYRSA_REQ_ORG "VPN Skills39"
set_var EASYRSA_REQ_EMAIL "spskills@spain-skills.es"
set_var EASYRSA_REQ_OU "Región de Murcia - 39 Skills"

set_var EASYRSA_KEY_SIZE 4096
set_var EASYRSA_ALGO rsa
set_var EASYRSA_CA_EXPIRE 1260
set_var EASYRSA_CERT_EXPIRE 825
set_var EASYRSA_CRL_DAYS 365
set_var EASYRSA_CERT_RENEW 30
```

Una vez puestas, generamos la estructura de la PKI:

```
# cd <carpeta_donde_esté_easyrsa>
# ./easyrsa init-pki
# ./easyrsa build-ca
# ./easyrsa gen-dh
# ./easyrsa build-server-full <SERVER_NAME> nopass
# ./easyrsa gen-crl
# openvpn --genkey secret pki/ta.key
```

A los siguientes ficheros, enlace simbólico a la raíz de OpenVPN o similar:

```
cd openvpn_dir
ln -s easy-rsa/pki/ca.crt
ln -s easy-rsa/pki/dh.pem
ln -s easy-rsa/pki/issued/fwskills.spainskills.es.crt server.crt
ln -s easy-rsa/pki/private/fwskills.spainskills.es.key server.key
ln -s easy-rsa/pki/ta.key
ln -s easy-rsa/pki/crl.pem
```

Con esto tenemos generada toda la parte de servidor, ahora toca proceder a generar los certificados de usuario:

```
cd easy-rsa
./easyrsa build-client-full <CLIENT_NAME> nopass
```

Para retirarlos usamos

```
./easyrsa revoke vpn-client-02
./easyrsa gen-crl
```

Para que los clientes confíen en una CA añadimos el certificado de la CA a **/usr/local/share/ca-certificates/** y ejecutamos

```
update-ca-certificates
```

OpenSSL (2º método)

Editar /etc/ssl/openssl.conf

```
[ CA_default ]
dir          = /etc/keys # O dónde quieras.
...
new_certs_dir = $dir/newcerts
certificate   = $dir/ca.crt
...
private_key   = $dir/ca.key
...
default_days  = 365
```

Crear csr.conf (**MODIFICAR CN Y OU POR CADA MÁQUINA**)

```
[ req ]  
default_bits = 2048  
distinguished_name = req_distinguished_name  
req_extensions = req_ext  
prompt = no  
  
[ req_distinguished_name ]  
C = ES  
ST = Murcia  
L = Murcia  
O = Skills Ltd  
OU = Área de Informática  
CN = ldap.skills39.io  
  
[ req_ext ]  
keyUsage = keyEncipherment, dataEncipherment, digitalSignature  
subjectAltName = @alt_names  
extendedKeyUsage = serverAuth # O clientAuth  
  
[ alt_names ]  
DNS.1 = ldap.skills39.io  
DNS.2= pubsrv.skills39.io
```

Generar certificado CA

```
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout "ca.key"  
-out "ca.crt"
```

Generar solicitud de certificado SSL (CSR)

```
openssl req -new -newkey rsa:2048 -nodes -keyout "server1.key" -out  
"server1.csr" -config csr.conf
```

Generar certificado SSL firmando con CA

```
openssl x509 -req -in server1.csr -CA ca.crt -CAkey ca.key -CAcreateserial  
-out server1.crt -days 500 -sha256 -extfile v3.ext
```

Revocar certificado SSL

```
sudo openssl ca -revoke server1.crt
```

Para que los clientes confíen en una CA añadimos el certificado de la CA a
/usr/local/share/ca-certificates/ y ejecutamos

```
update-ca-certificates
```

Configuración del servidor

La configuración se hace en el fichero `/etc/openvpn/server.conf`, pondremos los siguientes parámetros:

```
# Secure OpenVPN Server Config
# Basic Connection Config
dev tun
proto udp
port 3394
keepalive 10 120
max-clients 10

# management localhost 1194
management 127.0.0.1 1194 pw

# Certs
ca ca.crt
cert server.crt
key server.key
dh dh.pem
tls-auth ta.key 0

# Ciphers and Hardening
reneg-sec 0
remote-cert-tls client
crl-verify crl.pem
tls-version-min 1.2
cipher AES-256-GCM
data-ciphers AES-256-GCM:AES-128-GCM:CHACHA20-POLY1305:AES-256-CBC:AES-128-CBC

auth SHA512
tls-version-min 1.2
tls-cipher
TLS-DHE-RSA-WITH-AES-256-GCM-SHA384:TLS-DHE-RSA-WITH-AES-256-CBC-SHA256:TLS-ECDH
E-ECDSA-WITH-CHACHA20-POLY1305-SHA256:TLS-ECDHE-RSA-WITH-CHACHA20-POLY1305-SHA25
6:TLS-DHE-RSA-WITH-AES-128-GCM-SHA256:TLS-DHE-RSA-WITH-AES-128-CBC-SHA256

# Drop Privs
user nobody
group nogroup

# IP pool
server 10.200.0.0 255.255.255.0
topology subnet
ifconfig-pool-persist ipp.txt
client-config-dir ccd
# Routes pushing to clients
push "route 192.168.18.0 255.255.255.0"
# Misc
persist-key
```

```

persist-tun
# comp-lzo DEPRECATED, the use of compression is not recommended
# DHCP Push options force all traffic through VPN and sets DNS servers
#push "redirect-gateway def1 bypass-dhcp"
push "dhcp-option DNS 8.8.4.4"
push "dhcp-option DOMAIN spain-skills.es"

# Logging
log-append /var/log/openvpn/openvpn.log
verb 3
status /var/log/openvpn/openvpn-status.log

```

También podemos obtener un fichero de configuración de ejemplo desde `/usr/share/doc/openvpn/examples/sample-config-files/server.conf` y moldearlo a nuestras necesidades.

Configuración del cliente

```

# Secure OpenVPN Client Config
tls-client
pull
client
dev tun
proto udp
remote iescierva.net 3395
#redirect-gateway def1
#redirect-gateway local
nobind
persist-key
persist-tun
verb 3
remote-cert-tls server
ns-cert-type server
key-direction 1
cipher AES-256-GCM
tls-version-min 1.2
auth SHA512
tls-cipher
TLS-DHE-RSA-WITH-AES-256-GCM-SHA384:TLS-DHE-RSA-WITH-AES-256-CBC-SHA256:
TLS-DHE-RSA-WITH-AES-128-GCM-SHA256:TLS-DHE-RSA-WITH-AES-128-CBC-SHA256
#log-append /var/log/openvpn/openvpn.log

```

Y luego habría que especificar la ubicación de los certificados en el cliente. Pero es mucho mejor la opción de abajo, que genera un ovpn.

Generar fichero cliente

Para easy-rsa:

```
#!/bin/bash

CERTS_DIR= #ubicación de certificados
VPN_TEMPLATE= # ubicación de fichero plantilla

printf "Introduce nombre certificado\n"
read nombre

printf "Creando credenciales...\n"
./easyrsa build-client-full "$nombre" nopass

TARGET_FILE="$CERTS_DIR"/ovpn/"$nombre" #ubicación de certificados
creados

grep -v "$VPN_TEMPLATE" > "$TARGET_FILE" # Copiar fichero

# Añadir claves.
echo "<ca>" >> $TARGET_FILE
cat pki/ca.crt >> "$TARGET_FILE"
echo "</ca>" >> "$TARGET_FILE"

echo "<cert>" >> $TARGET_FILE
cat pki/issued/$VPN_FULLNAME.crt >> $TARGET_FILE
echo "</cert>" >> $TARGET_FILE

echo "<key>" >> $TARGET_FILE
cat pki/private/$VPN_FULLNAME.key >> $TARGET_FILE
echo "</key>" >> $TARGET_FILE

echo "<tls-auth>" >> $TARGET_FILE
cat pki/ta.key >> $TARGET_FILE
echo "</tls-auth>" >> $TARGET_FILE
```

Para OpenSSL:
TODO

Reglas básicas IPTABLES-VPN

```
#!/bin/bash

IPTABLES=/sbin/iptables
IP6TABLES=/sbin/ip6tables
IF="ens18"
VPN_PORT=1194
VPN_NETWORK=10.164.0.0/24

echo -e "Aplicando Reglas de Firewall a esta máquina (`hostname`)"

echo -e "\tNo permitimos tráfico IPv6"
$IP6TABLES -F
$IP6TABLES -X
$IP6TABLES -Z
$IP6TABLES -P INPUT DROP
$IP6TABLES -P OUTPUT DROP
$IP6TABLES -P FORWARD DROP

echo -e "\tVaciando y borrando reglas anteriores"
$IPTABLES -F
$IPTABLES -t nat -F
$IPTABLES -X
$IPTABLES -Z

echo -e "\tEstableciendo política por defecto: DROP"
$IPTABLES -P INPUT DROP
$IPTABLES -P OUTPUT DROP
$IPTABLES -P FORWARD DROP


echo -e "\tPermitimos tráfico interfaz de loopback"
$IPTABLES -A INPUT -i lo -j ACCEPT
$IPTABLES -A OUTPUT -o lo -j ACCEPT

echo -e "\tPermitimos tráfico ICMP"
$IPTABLES -A INPUT -i $IF -p icmp -j ACCEPT
$IPTABLES -A OUTPUT -o $IF -p icmp -j ACCEPT

echo -e "\tPermitimos tráfico entrante hacia la interfaz $IF"
# Conexiones SSH
$IPTABLES -A INPUT -i $IF -p tcp --dport 22 -m state --state NEW -j
ACCEPT
$IPTABLES -A INPUT -i $IF -p udp --dport $VPN_PORT -m state --state NEW
-j ACCEPT
$IPTABLES -A FORWARD -i tun0 -o $IF -m state --state RELATED,ESTABLISHED
-j ACCEPT
$IPTABLES -A FORWARD -i $IF -o tun0 -m state --state RELATED,ESTABLISHED
-j ACCEPT
```

```
echo -e "\tPermitimos trafico VPN a VPN"
$IPTABLES -A FORWARD -i tun0 -s $VPN_NETWORK -d 172.29.0.0/16 -j ACCEPT

# Tráfico entrante de conexiones establecidas
$IPTABLES -A INPUT -i $IF -m state --state ESTABLISHED,RELATED -j ACCEPT

echo -e "\tNAT a VPN"
$IPTABLES -t nat -A POSTROUTING -s $VPN_NETWORK -o $IF -j MASQUERADE

echo -e "\tAllow TUN"
$IPTABLES -A INPUT -i tun0 -j ACCEPT
$IPTABLES -A INPUT -i tun0 -j ACCEPT

echo -e "\tBloqueando trafico entre clientes VPN"
$IPTABLES -A FORWARD -i tun0 -s $VPN_NETWORK -d $VPN_NETWORK -j DROP

echo -e "\tPermitimos tráfico saliente desde la interfaz $IF"
#Permitimos todo el tráfico de salida
$IPTABLES -A OUTPUT -o $IF -j ACCEPT
```

HTTP

Nginx

```
apt install nginx
mkdir -p /var/www/dominio_web/html
chown -R www:www /var/www/dominio_web/html
```

Dentro de ese directorio se almacenará toda la página web

Para ahora servir la web, hay que definirla como se haría en Apache con los VirtualHost pero en nginx

```
vim /etc/nginx/sites-available/web1.skills39.io
```

Este sería el contenido básico

```
server {
    listen 80;
    listen [::]:80;

    root /var/www/web1.skills39.io/html;
    index index.html index.htm index.nginx-debian.html;

    server_name web1.skills39.io nombre_alt.skill39.io;

    location / {
        try_files $uri $uri/ =404;
    }
}
```

Para habilitarlo simplemente hacemos un enlace simbólico de este fichero al directorio sites-enabled

Reiniciamos el servidor

Funciona



Apache

Lo primero es crear el directorio donde se alojará el otro sitio.

```
root@moisesapache:~# mkdir /var/www/www.otrositiomoises.com/public_html  
root@moisesapache:~# mkdir /var/www/T6/public_html
```

Se copia 000-default.conf a otro fichero. En

/etc/apache2/sites-available/www.otrositiomoises.conf se añade “ServerName” y “ServerAlias” para indicar los nombres a los que responderá este virtualhost, y se modifica el DocumentRoot indicando donde está la raíz de html

```
Ubuntu Server SR Web [Corriendo] - Oracle VM VirtualBox  
Archivo Máquina Ver Entrada Dispositivos Ayuda  
GNU nano 6.2 /etc/apache2/sites-available/www.otrositiomoises.conf *  
<VirtualHost *:80>  
    # The ServerName directive sets the request scheme, hostname and port that  
    # the server uses to identify itself. This is used when creating  
    # redirection URLs. In the context of virtual hosts, the ServerName  
    # specifies what hostname must appear in the request's Host: header to  
    # match this virtual host. For the default virtual host (this file) this  
    # value is not decisive as it is used as a last resort host regardless.  
    # However, you must set it for any further virtual host explicitly.  
    #ServerName www.example.com  
  
    ServerAdmin root@localhost  
    ServerName otrositiomoises.com  
    ServerAlias www.otrositiomoises.com  
    DocumentRoot /var/www/www.otrositiomoises.com/public_html  
    ErrorLog ${APACHE_LOG_DIR}/error.log  
    CustomLog ${APACHE_LOG_DIR}/access.log combined  
</VirtualHost>  
  
# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

Se le da permisos al directorio con chmod -R 755 /var/www

```
root@moisesapache:~# chmod -R 755 /var/www/
```

Con el comando a2ensite habilitamos un virtualhost

```
root@moisesapache:~# a2ensite  
Your choices are: 000-default default-ssl www.otrositiomoises  
Which site(s) do you want to enable (wildcards ok)?  
www.otrositiomoises  
Enabling site www.otrositiomoises.  
To activate the new configuration, you need to run:  
    systemctl reload apache2  
root@moisesapache:~# _
```

Se ejecuta el comando que nos sugiere “systemctl reload apache2” y probamos con el cliente.

Sitios personales (apache2)

```
a2enmod userdir
```

```
pepe@server:~$ mkdir ~/public_html
```

La URL es <http://hostname/~pepe>

SSL

```
server {
    listen 443 ssl;
    listen [::]:443;

    root /var/www/web1.skills39.io/html;
    index index.html index.htm index.nginx-debian.html;

    server_name web1.skills39.io nombre_alt.skill39.io;

    ssl_certificate /etc/keys/server1.crt;
    ssl_certificate_key /etc/keys/server1.key;

    location / {
        try_files $uri $uri/ =404;
    }
}

server {
    listen 80;
    server_name web1.skills39.io;
    return 301 https://$host$request_uri;
}
</nginx/sites-available/web1.skills39.io" 22L, 482B
```

Para generar los certificados consultar apartado OpenSSL de OpenVPN

Apache

a2enmod ssl headers

```
<VirtualHost _default_:443>
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
    SSLEngine on
    SSLCertificateFile
    /etc/ssl/certs/ssl-cert-snakeoil.pem
    SSLCertificateKeyFile
    /etc/ssl/private/ssl-cert-snakeoil.key
        <FilesMatch "\.(cgi|shtml|phtml|php)$">
            SSLOptions +StdEnvVars
        </FilesMatch>
        <Directory /usr/lib/cgi-bin>
            SSLOptions +StdEnvVars
        </Directory>
</VirtualHost>
```

haproxy

```
global
    log 127.0.0.1 local2
    maxconn 4000
    user haproxy
    group haproxy

defaults
    mode http
    timeout connect 5s
    timeout client 5m
    timeout server 5m
    maxconn 2000
    log global

listen web
    bind :80
    balance roundrobin
    server server1 192.168.1.100 80 check
    server server2 192.168.1.101 80 check
```

¡Absolutamente! Aquí tienes un ejemplo de configuración de Haproxy para balanceo de carga en capa 4 (transporte):

```
global
    log 127.0.0.1 local2
    maxconn 4000
    user haproxy
    group haproxy

defaults
    mode tcp
    timeout connect 5s
    timeout client 5m
    timeout server 5m
    maxconn 2000
    log global

listen http_backend
    bind *:80
    balance roundrobin
    server server1 192.168.1.100:80 check
    server server2 192.168.1.101:80 check
```

HA-PROXY más detallado

```
HAProxy: balanceador de carga
GNU nano 4.8
defaults
    log      global
    mode     http
    option   httplog
    option   dontlognull
    timeout  connect 5000
    timeout  client  50000
    timeout  server  50000
    errorfile 400 /etc/haproxy/errors/400.http
    errorfile 403 /etc/haproxy/errors/403.http
    errorfile 408 /etc/haproxy/errors/408.http
    errorfile 500 /etc/haproxy/errors/500.http
    errorfile 502 /etc/haproxy/errors/502.http
    errorfile 503 /etc/haproxy/errors/503.http
    errorfile 504 /etc/haproxy/errors/504.http

frontend sitio_http
    bind 192.0.2.254:80
    acl host_web1 hdr(host) -i www.web1.example
    acl host_web2 hdr(host) -i www.web2.example
    use_backend sitio_web1 if host web1
    use_backend sitio_web2 if host_web2

backend sitio_web1
    balance roundrobin
    server web01 172.31.0.11:80 check
    server web02 172.31.0.12:80 check
    server web05 172.31.0.15:80 backup check

backend sitio_web2
    balance roundrobin
    server web03 172.31.0.13:80 check
    server web04 172.31.0.14:80 check
    server web05 172.31.0.15:80 backup check
```

HA-PROXY con HTTPS

En caso que se tenga que balancear **un https** hay que romper el túnel ssl en el ha proxy por lo que se le tiene que ofrecer el certificado etc.

```
frontend http

bind *:443 ssl crt /etc/ssl/certs/mycert.pem ciphers
ECDHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-G
CM-SHA384:DHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AE
S128-SHA256:ECDHE-RSA-AES256-SHA:ECDHE-RSA-AES128-SHA:DHE-RSA-AES256-SHA
256:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA:ECDHE-RS
A-DES-CBC3-SHA:EDH-RSA-DES-CBC3-SHA:AES256-GCM-SHA384:AES128-GCM-SHA256:
AES256-SHA256:AES128-SHA256:AES256-SHA:AES128-SHA:DES-CBC3-SHA:HIGH:!aNU
LL:!eNULL:!EXPORT:!DES:!MD5:!PSK:!RC4

mode tcp
acl backend1 ssl_fc_sni backed1.domain.com
acl backend2 ssl_fc_sni backed2.domain.com
tcp-request inspect-delay 5s
use_backend backend1 if backend1

backend backend1
(aquí la misma configuración que para el http normal para balancear)
```

vsFTPd

```
apt install vsftpd
Fichero conf: /etc/vsftpd.conf
Logs: /var/log/vsftpd.log
Directorio default: /srv/ftp # De anónimo
Habilitar anónimo:
anonymous_enable=yes
```

Permitir upload de anon

```
anon_upload_enable=yes
write_enable=yes
```

Cambiar directorio de anon

```
anon_root=/anon
```

Enjaular usuario a su home:

```
chroot_list_enable=yes
chroot_list_file=fichero # Se especifica un fichero donde irán los
usuarios enjaulados
```

```
allow_writeable_chroot=yes  
local_umask=022
```

Habilitar mensaje personalizado:

```
ftpd_banner="Ola este es un mensaje personalizado"
```

Limitar ancho de banda:

```
anon_max_rate=16337652 # Va en bytes  
local_max_rate=2139832198  
max_clients=5 # Clientes simultáneos  
max_per_ip=3
```

radvd

Para que Linux haga de router IPv6, hay que habilitar el enrutamiento en sysctl.conf y es interesante instalar radvd para anunciar que la máquina es enrutadora.

Se instala con `apt install radvd` y este es un ejemplo de configuración:

```
interface eth0
{
    MinRtrAdvInterval 3;
    MaxRtrAdvInterval 4;
    AdvSendAdvert on;
    AdvManagedFlag on;
    prefix 2001:db7::/64
    { AdvValidLifetime 14300; AdvPreferredLifetime 14200; }
};
```

VLAN

```
modprobe 802.1q
echo 8021q | tee -a /etc/modules
vim /etc/network/interfaces

auto ens192.501
iface ens192.501 inet static
    address 172.16.2.10
    netmask 255.255.255.0
    gateway 172.16.2.1
    dns-nameserver 8.8.8.8
```

iptables

ICMP

```
-p icmp
-p icmp --icmp-type echo-request
```

NAT

Para habilitar NAT en un sistema Linux, primero se habilita el enrutamiento. Para ello, en `/etc/sysctl.conf`:

```
# Uncomment the next line to enable packet forwarding for IPv4
net.ipv4.ip_forward=1
```

Acto seguido, ejecutar `sysctl -p`

Ahora hace falta crear un script(el que será de nuestro FW) en el que pongamos el siguiente comando:

```
iptables -t nat -A POSTROUTING -o "$INT_EXT" -j MASQUERADE
```

DNAT (redirección de puertos)

Para mostrar un servicio de la red interna hacia el exterior (fuera del NAT), se hacen reglas sobre la chain PREROUTING:

```
iptables -t nat -A PREROUTING -p tcp --dport 8080 -o "$INT_EXT" -j DNAT  
--to 192.168.10.2:80
```

nftables

```
nft add table inet filter
```

Para crear la cadena input:

```
nft 'add chain inet filter input { type filter hook input priority 0 ;  
counter ; policy drop ; }'
```

Existen 6 tipos de hooks donde situar las cadenas:

- **prerouting**: sees all incoming packets, before any routing decision has been made. Packets may be addressed to the local or remote systems.
- **input**: sees incoming packets that are addressed to and have now been routed to the local system and processes running there.
- **forward**: sees incoming packets that are not addressed to the local system.
- **output**: sees packets that originated from processes in the local machine.
- **postrouting**: sees all packets after routing, just before they leave the local system.
- **ingress** (a partir del Linux kernel 4.2 para la familia netdev y a partir del Linux kernel 5.10 para la familia inet). 'Ve' los paquetes justo antes de que pasen por el driver de la interfaz de red, incluso antes del hook prerouting.

Ejemplos de uso:

```
nft add rule inet filter input tcp dport != 22
```

```
nft add rule inet filter input tcp dport \>= 1024
```

```
nft add rule inet filter input tcp dport \> 1024
```

```
nft add rule inet filter output position 0 ip daddr 172.20.254.100 drop
```

En este ejemplo la regla se agregaría la primera de todas.

También se agregaría la primera de todas si usamos directamente insert sin especificar la posición:

```
nft insert rule inet filter output ip saddr 172.20.254.200 drop
```

En función de que se utilice add o insert la regla se asigne antes o después de la que se especifica con la opción position.

En este ejemplo la regla se agregaría en la posición de la regla con handle 8, desplazando a dicha regla una posición hacia abajo en el orden.

```
nft insert rule inet filter output position 8 ip daddr 172.20.254.101 drop
```

Si hubiéramos utilizado add en lugar de insert la regla se hubiera agregado después de la regla especificada con el handler dado:

```
nft add rule inet filter output position 8 ip daddr 172.20.254.102 drop
```

Se pueden listar las reglas de una tabla con la siguiente ejecución:

```
nft -n list table inet filter
```

SNAT en nftables

Tendremos que crear las tablas y cadenas necesarias:

```
nft add table nat
nft add chain nat postrouting type nat hook postrouting priority 0
```

```
nft add chain nat prerouting type nat hook prerouting priority 0
```

Bash

Special variables in bash

Here's quick look into the special variables you get in bash shell:

Special Variable	Description
\$0	Gets the name of the current script.
\$#	Gets the number of arguments passed while executing the bash script.
\$*	Gives you a string containing every command-line argument.
\$@	It stores the list of every command-line argument as an array.
\$1-\$9	Stores the first 9 arguments.
\$?	Gets the status of the last command or the most recently executed process.
\$!	Shows the process ID of the last background command.
\$\$	Gets the process ID of the current shell.
\$-	It will print the current set of options in your current shell.



ansible

```
apt install ansible
```

Generar claves SSH y distribuirlas por todas las máquinas.
ssh-keygen

Hacer un inventario `inventory`:

```
[all]
pubsrv ansible_host=192.168.122.33
dmz1 ansible_host=192.168.122.56
fwsede ansible_host=192.168.122.254
fwsuc ansible_host=192.168.122.253
```

Generar un fichero `group_vars/all` en el directorio donde esté el inventario y poner el siguiente contenido:

```
ansible_ssh_private_key_file: ~/.ssh/ansible
ansible_user: root
```

Ejecutar comando puntual:

```
ansible all -i inventory -a "<comando>"
ansible dmz1 -i inventory -a "<comando>"
ansible dmz1,fwsede -i inventory -a "<comando>"
```

Si hace falta consultar **DOCUMENTACIÓN DE ALGÚN MODULO**:

```
ansible-doc <modulo>
```

Playbook

Instalar programa:

```
---
- hosts: all
  tasks:
    - name: Install vim editor
      ansible.builtin.apt: name=vim state=present
    - name: Configure hosts
      ping:
```

Copiar fichero local a remoto:

```
---
- hosts: all
  tasks:
    - name: Create folder if not exists
      file:
        path: /etc/dhcp/dhcpd.d
        state: directory
        mode: 0750
    - name: Copy the file
      copy:
        src: hosts.conf # Ruta al fichero
        dest: /etc/dhcp/dhcpd.d/hosts.conf
        owner: root
        group: root
        mode: 0750
    - name: Restart DHCP server
      systemd:
        name: isc-dhcp-server
        state: restarted
```

Copiar fichero plantilla:

```
---
- hosts: dmz1
  tasks:
    - name: Copy the file
      template:
        src: named.conf.local # Ruta al fichero
        dest: /etc/bind/named.conf.local
        owner: root
        group: root
        mode: 0700
```

En el inventario se pueden definir variables:

```
dmz1 ansible_host=10.0.0.251 domain=spainskills.es
```

Y en el fichero plantilla queda tal que así:

```
view internet {
    match-clients {externo;};
    zone "{{ domain }}" {
        type master;
        file "/var/lib/bind/db.{{domain}}_ext";
    };
}

view internal {
    match-clients {interno;};
    zone "{{domain}}" {
        type master;
        file "/var/lib/bind/db.{{domain}}";
    };
}
```

Donde {{ variable }} se reemplazaría por el valor que tenga en inventory cada máquina.

Ejemplo LDAP clientes:

Primero de todo, tener los ficheros de la sección de **Autenticar usuarios** nsswitch.conf y ns lcd.conf a mano. Una vez en el directorio y MODIFICADOS, configuramos el siguiente playbook:

```
---
- hosts: dmz1
  tasks:
    - name: install openldap client
      apt:
        name: "{{item}}"
        state: present
        update_cache: yes
      environment:
        DEBIAN_FRONTEND: noninteractive
      with_items:
        - libpam-ldapd
        - libnss-ldap
        - nscd
        - ns lcd
        - ldap-utils

    - name: copy nsswitch and ns lcd config
      copy:
        src: "{{item}}"
        dest: /etc/{{item}}
```

```

    owner: root
    group: root
  with_items:
    - nsswitch.conf
    - nslcd.conf

- name: copy pam common-session config
  template:
    src: "{{item}}"
    dest: /etc/pam.d/{{item}}
  with_items:
    - common-session

- name: restart nslcd service
  service:
    name: "{{item}}"
    state: restarted
    enabled: yes
  with_items:
    - nscd
    - nslcd

- name: test ldap user exist
  shell: getent passwd test_user
  register: ldap_user_exist

- name: show test ldap user exist result
  debug: var=ldap_user_exist.stdout

```

Ejemplo DNS entero:

Para DNS son varios los ficheros a preparar y usaremos plantillas con el dominio definido en los host DNS:

Ejemplo Postfix entero:

Windows

Servicios básicos (DHCP, DNS, IIS)

Para desplegar DHCP, DNS y Servidor Web en un servidor Windows, seguimos los siguientes pasos: en el **Administrador del servidor** -> Agregar roles y características -> en Roles de servidor ponemos el rol necesario.

Para configurar **DHCP** basta con, una vez instalado el servicio, ir a Herramientas -> DHCP y ahí hacer click derecho sobre el protocolo de red que usemos (IPv4 o IPv6) y darle a “Ámbito nuevo” y seguir el configurador.

Para **DNS**, es recomendable que de antemano hayamos habilitado el servicio de dominio de Active Directory para poder habilitar las actualizaciones dinámicas. Una vez instalado Active Directory, ya tenemos DNS configurado.

Para **IIS**, se añade el servicio y una vez instalado se le da a Administrador de IIS (o algo así), se hace click derecho sobre el servidor y se crea el sitio en cuestión.

NTP

Para **NTP**:

1. En regedit:
Computer>HKEY_LOCAL_MACHINE>SYSTEM>CurrentControlSet>Services>W32Time>TimeProviders>NtpServer.
2. Poner el valor de **Enabled** a 1.
3. En **Computer > HKEY_LOCAL_MACHINE > SYSTEM > CurrentControlSet > Services > W32Time > Config > AnnounceFlags** poner su valor a 5.
4. Reiniciar el servicio Windows Time (HORA DE WINDOWS)

Comandos útiles:

To check NTP configuration, run:

```
w32tm /query /configuration
```

To check NTP server list, type:

```
w32tm /query /peers
```

To force NTP server synchronization, run:

```
w32tm /resync /nowait
```

Use the command below to show the source of the NTP time:

```
w32tm /query /source
```

To show the status of NTP service, type:

```
w32tm /query /status
```

Añadir certificado de CA

Choose the Download CA certificate link and then choose Open option when prompted to open or save the certificate. (dale formato **pem o crt** pa que te salga eso)

When the certificate window opens, choose Install Certificate.... The Certificate Import wizard appears.

Cuando are prompted for the Certificate Store, choose Place all certificates in the following store. Select the Trusted Root Certification Authorities store.

Complete the remaining steps of the wizard and click Finish.

RECORDAR QUE MOZILLA APPS A VECES NO USAN ESTE ALMACÉN, Y HAY QUE IMPORTARLO AHÍ TAMBIEN

Active Directory Certificate Services

Añadir el servicio y marcar Certification Authority (Entidad Certificadora) y Servicio Web de Inscripción de certificados (nose que Enrollment).

En tipo de instalación Enterprise CA → Root CA → Create new private key y todo palante.

Crear usuario ces01 y añadirlo al grupo IIS_IUSRS

Set a service principal name for the service account:

```
setspn -s http/$env:COMPUTERNAME.skills39.io skills39.io\ces01
```

Luego hacer click derecho sobre el usuario en **USUARIOS Y YOQUESE DE ACTIVE DIRECTORY** y en **DELEGATION** decir que **TRUST THIS USER FOR DELEGATION TO SPECIFIED SERVICES ONLY** → **ADD** → **USERS or COMPUTERS** → **NOMBRE DEL SERVIDOR QUE HOSTEA LA CA** → **AÑADIR SERVICIOS HOST Y rpcss**

Perfiles móviles (roaming profiles)

Primero se tiene que crear un **grupo de seguridad** (grupo de AD, tipo de grupo Seguridad y en ámbito Global) y se agregan los usuarios.

Una vez creado, se crea un **recurso compartido** (como un directorio) a través del Administrador del servidor

RECURSOS COMPARTIDOS
Todos los recursos compartidos | 3 en total

Compartir	Ruta local	Protocolo	Tipo de dispositivo
NETLOGON	C:\Windows\SYSVOL\sysvol\skills...	SMB	No en clúster
profiles	C:\Perfiles móviles\profiles	SMB	No en clúster
SYSVOL	C:\Windows\SYSVOL\sysvol	SMB	No en clúster

VOLUMEN
NETLOGON en WIN-OON
Nuevo recurso compartido...
Actualizar 3
60,2 % usado

CUOTA
NETLOGON en WIN-OON
Para usar las cuotas, a
Para instalar el Admí

A este recurso se tendrá que modificar los permisos (click derecho, Propiedades, Seguridad, Opciones avanzadas) quitarle la **herencia** de permisos (Convertir permisos heredados en permiso explícito en este objeto), quitarle los usuarios asociados, y ponerle los siguientes permisos avanzados (los importantes son **Crear carpetas** y **Mostrar carpeta**):

Entidad de seguridad: Roaming_Profiles (SKILLS39-DC1\Roaming_Profiles) [Seleccionar una entidad de seguridad](#)

Tipo:	Permitir
Se aplica a:	Solo esta carpeta

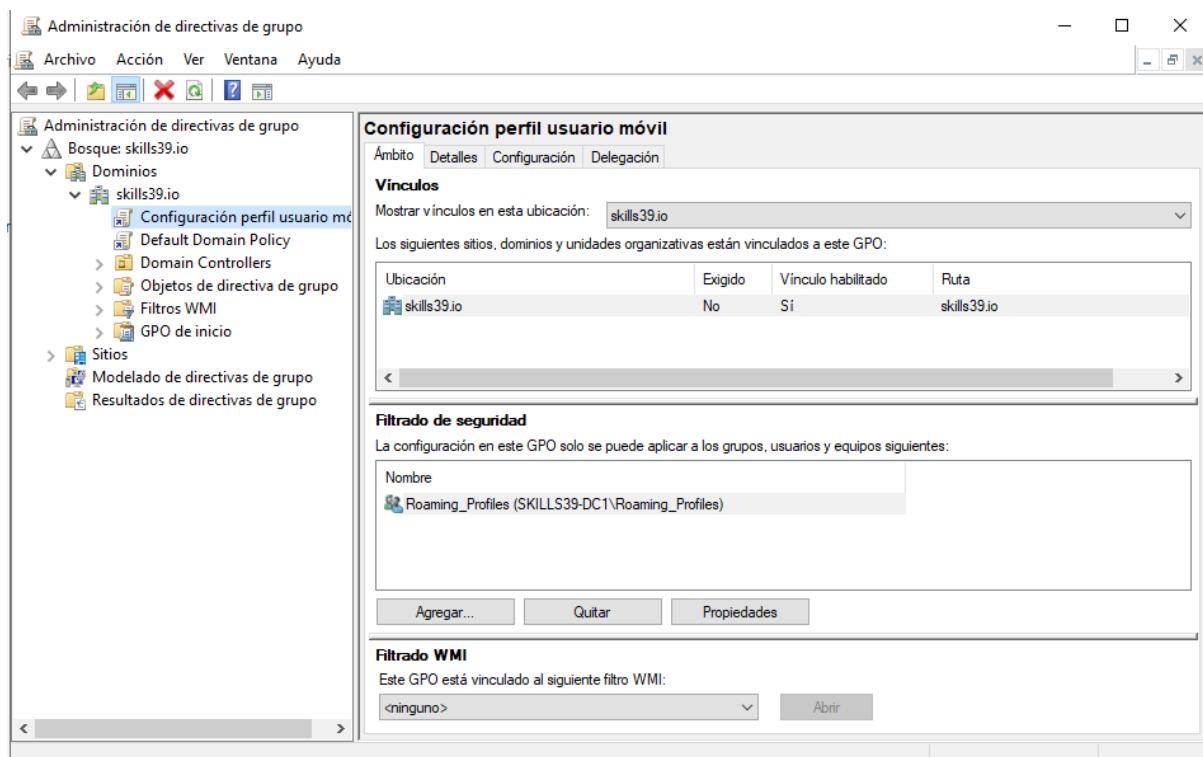
Permisos avanzados:

- | | |
|--|--|
| <input type="checkbox"/> Control total | <input type="checkbox"/> Escribir atributos |
| <input checked="" type="checkbox"/> Atravesar carpeta / ejecutar archivo | <input type="checkbox"/> Escribir atributos extendidos |
| <input checked="" type="checkbox"/> Mostrar carpeta / leer datos | <input type="checkbox"/> Eliminar subcarpetas y archivos |
| <input checked="" type="checkbox"/> Leer atributos | <input type="checkbox"/> Eliminar |
| <input checked="" type="checkbox"/> Leer atributos extendidos | <input checked="" type="checkbox"/> Permisos de lectura |
| <input type="checkbox"/> Crear archivos / escribir datos | <input type="checkbox"/> Cambiar permisos |
| <input checked="" type="checkbox"/> Crear carpetas / anexar datos | <input type="checkbox"/> Tomar posesión |



Crear **GPO** para la configuración de perfiles móviles (Administración de directivas de grupos) se quita la casilla **Vínculo habilitado** (lo que permite que se habilite o no), en la sección **Filtrado de seguridad** de la pestaña **Ámbito**, selecciona **Usuarios autenticados** y, a continuación, selecciona **Quitar** para evitar que el GPO se aplique a todos los usuarios.

1. En la sección **Filtrado de seguridad**, selecciona **Agregar**.
2. En el cuadro de diálogo **Seleccione usuario, equipo o grupo** escribe el nombre del grupo de seguridad que creó en el paso 2 (por ejemplo, *Equipos y usuarios de perfiles de usuario móviles*) y selecciona **Aceptar**.
3. Selecciona la pestaña **Delegación**, elige **Agregar**, escribe **Usuarios autenticados**, selecciona **Aceptar** y, a continuación, **Aceptar** de nuevo para aceptar los permisos de lectura predeterminados.



Para que los usuarios funcionen por perfiles móviles: Centro administración Active Directory
-> Users -> Propiedades sobre el usuario -> Perfil y en ruta de acceso del perfil se pone el recurso compartido\%username%

Volver a marcar **Vínculo habilitado** en la GPO creada previamente para que se aplique

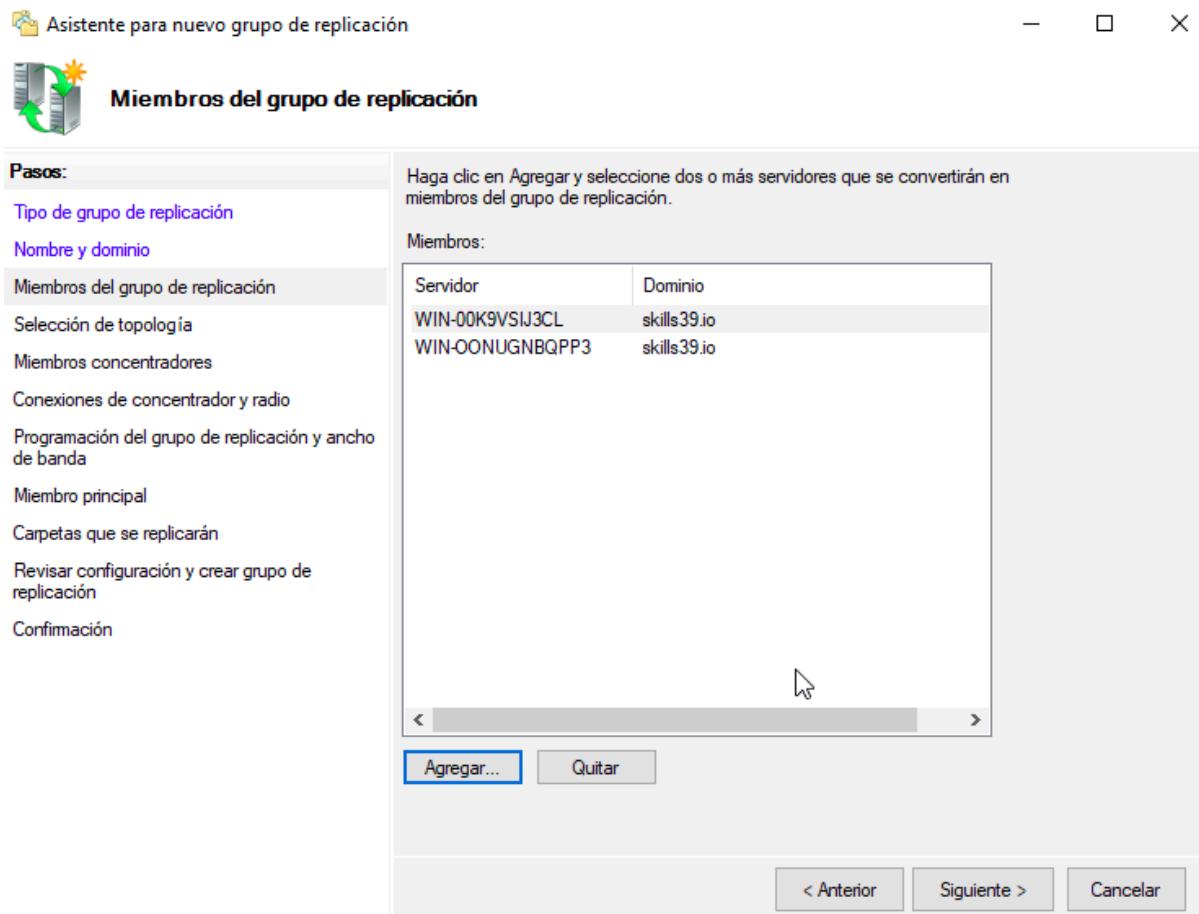
Reiniciamos los clientes y si siguen sin funcionar, como administrador se ejecuta **gpupdate /force**

DFS (Distributed File System)

DFS se compone de “espacio de nombres DFS” y “replicación DFS”. El primero permite tener recursos compartidos en la red con el nombre del dominio, en vez de tener que especificar el nombre del equipo en el que se conecta, mientras que la **replicación** permite que el recurso compartido se encuentre en más de dos equipos y esté sincronizado.

Para desplegarlo, se añadirán en **mínimo dos nodos** los servicios: **espacio de nombres DFS y Replicación DFS** (también añadimos **Administrador de recursos del servidor de archivos** para gestionar cuotas y demás)

Una vez instalados, se va a **Administración de DFS** y se le da a **Crear espacio de nombres**. Se sigue el tutorial y una vez creado, creamos el grupo de replicación.



Ahora hay que configurar las carpetas para dar permiso (sobre la carpeta)

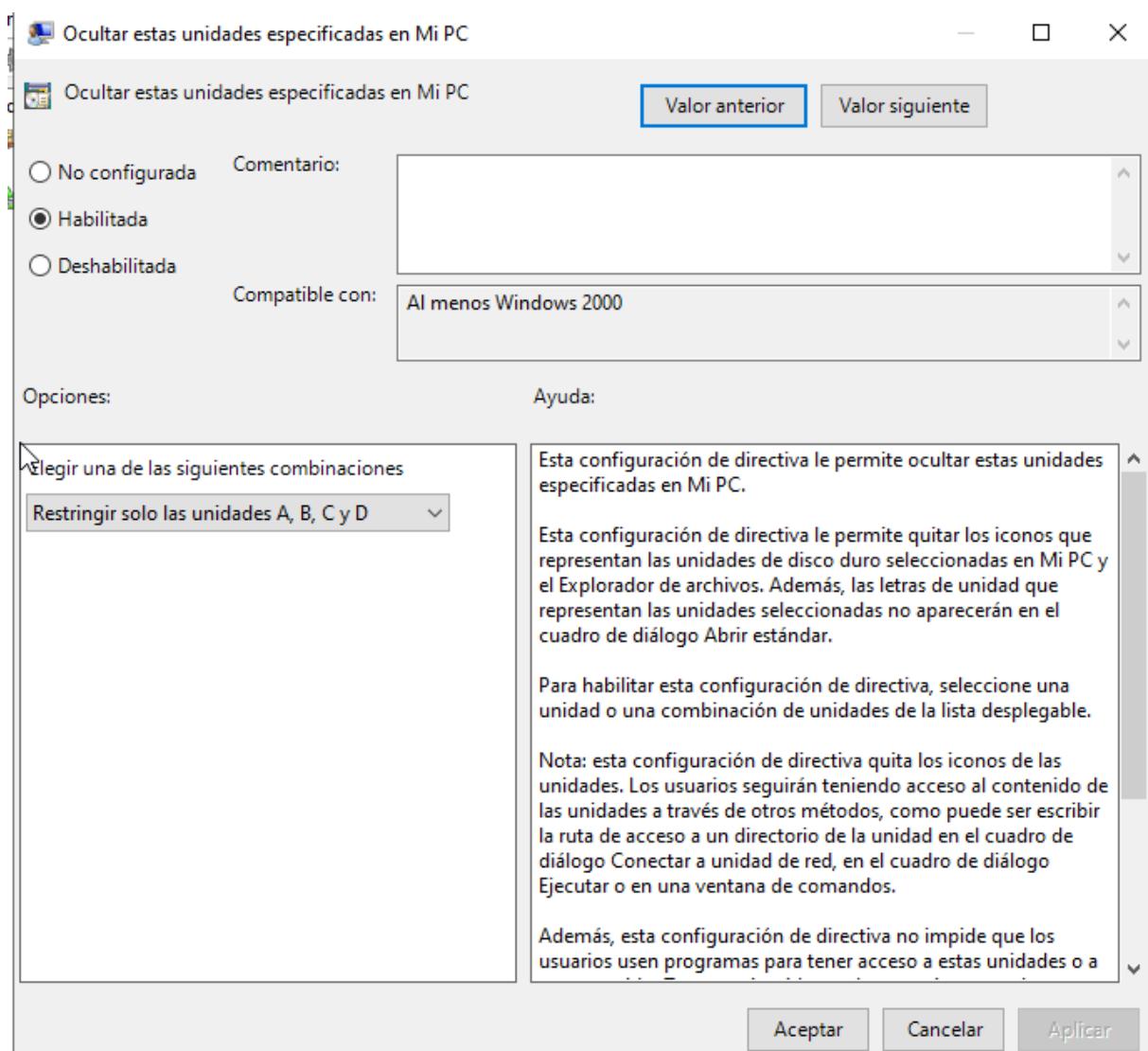
GPOs (Objeto de directiva de grupo)

Ejemplo 00: Impedir y ocultar acceso a unidades de sistema

Para crear una GPO, hay que abrir **gpmc.msc** y click derecho sobre el dominio, crear GPO, le damos un nombre y una vez creada click derecho sobre esta y Editar. Al darle a Editar se nos abre un “Editor de administración de directivas de grupo”, ahí para configurarla, vamos a **Configuración de usuario → Directivas → Plantillas administrativas → Componentes de Windows → Explorador de archivos**.

Ahí seleccionamos las directivas a modificar, en este caso

1. “Ocultar estas unidades especificadas en Mi PC”



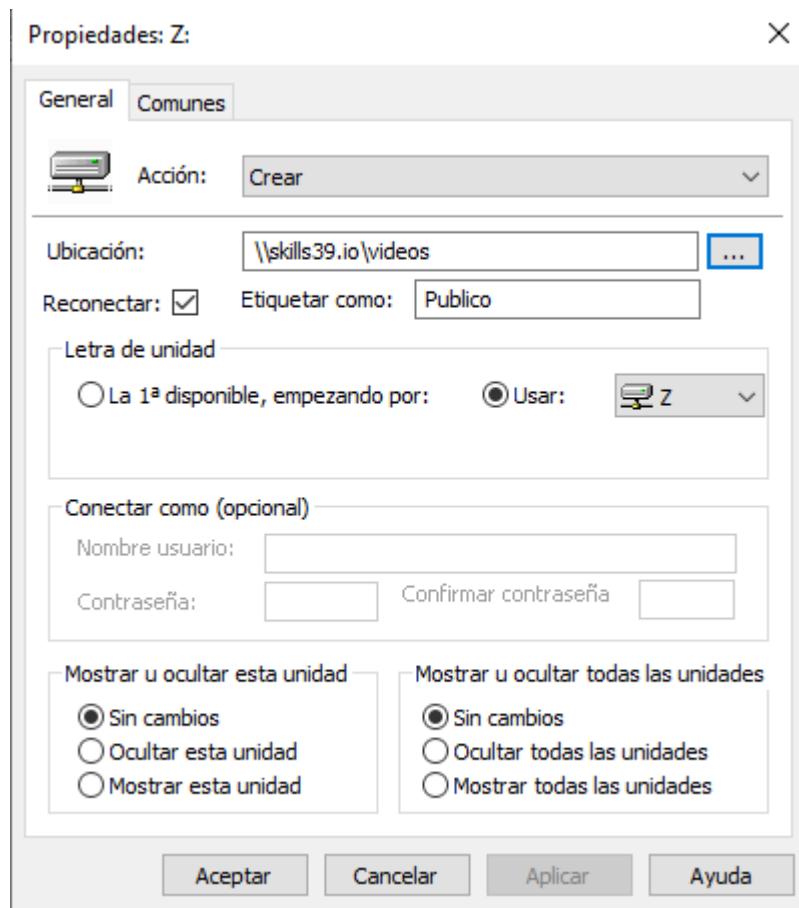
Tras modificar las políticas, hacemos un **gpupdate /force** para regenerar las políticas y si vemos que no se aplica, cerramos sesión.

- 2. Impedir acceso a unidades desde Mi PC** (aunque parece que solo lo impide a través del explorador, por lo que probablemente con aplicaciones de terceros sea posible acceder)

Ejemplo 01: Mapear unidades de red mediante GPO

Una vez creada la GPO, le damos a Configuración de usuario → Preferencias →Configuración de Windows→Asignaciones de unidades (Drive Maps)

Ahí hacemos click derecho → Nuevo, Acción: Crear, en ubicación le decimos la dirección de Red (si hemos usado DFS, usamos esa), marcamos Reconectar, aplicamos y gpupdate /force.



Configurar recurso compartido en red

NAT con Windows (ugh)

Para hacer NAT con Windows Server, primero debemos contar con **dos interfaces**, una que pueda llegar a Internet y otra privada para la red en la que estarán las máquinas.

Se añade el rol **Acceso remoto** y entonces accedemos en Herramientas a **Enrutamiento y acceso remoto**.

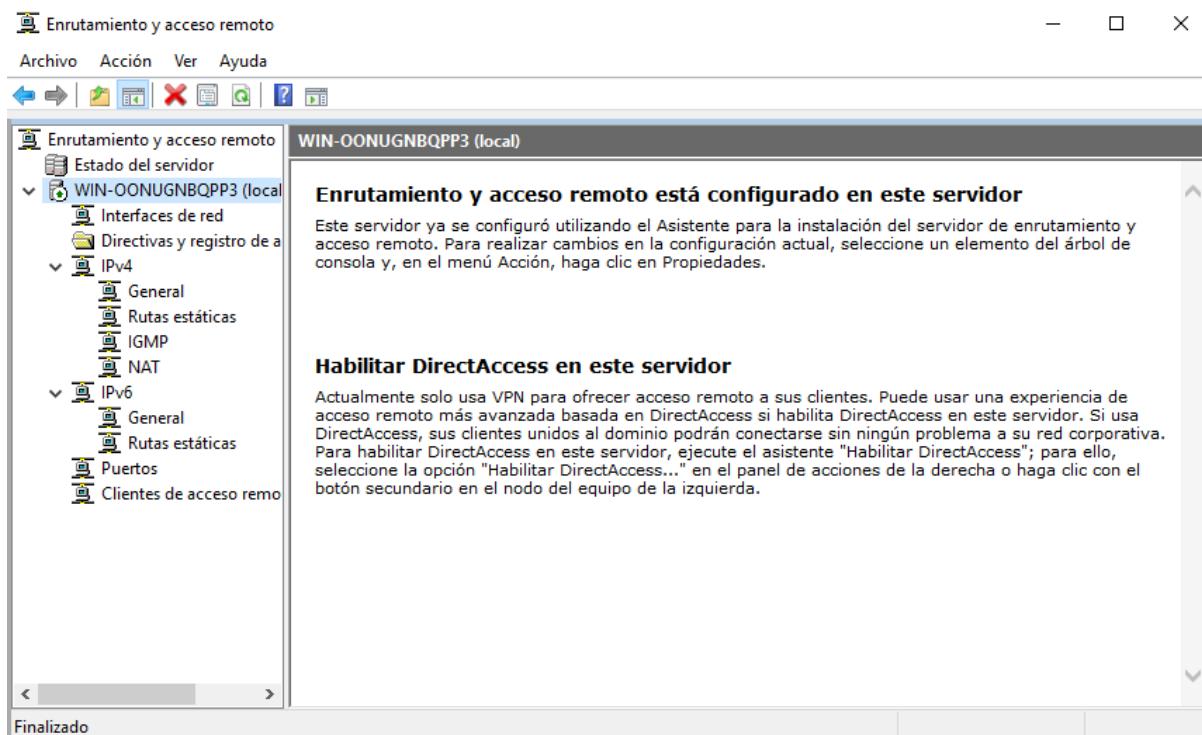
Para implementar **NAT**, click derecho sobre el servidor y la primera opción que dice algo de **Configurar kvfjksd**, se sigue el asistente y voilá.

Ya queda configurar el DHCP, reiniciar servicios “por si acaso” y ya.

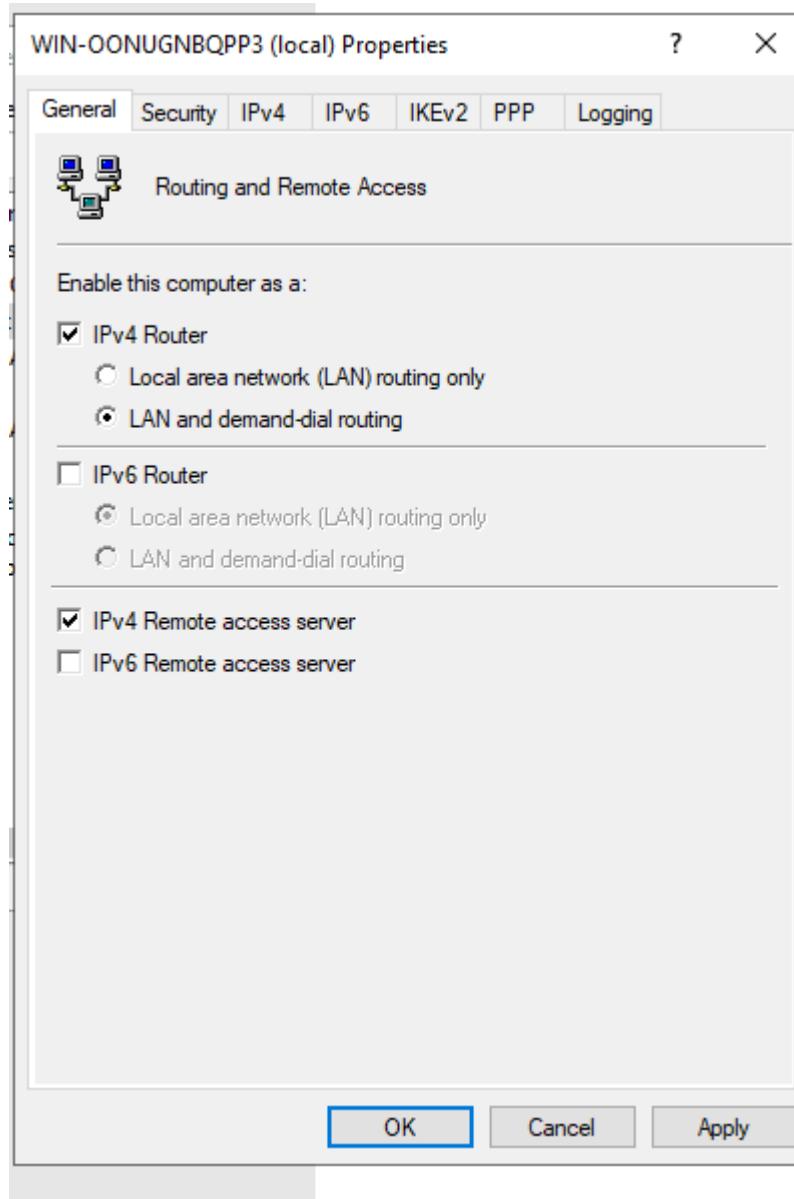
Si queremos hacer NAT Inverso (**redirección de puertos**), tenemos que hacer click derecho sobre la interfaz de red que da a Internet y configurarlo.

VPN con Windows

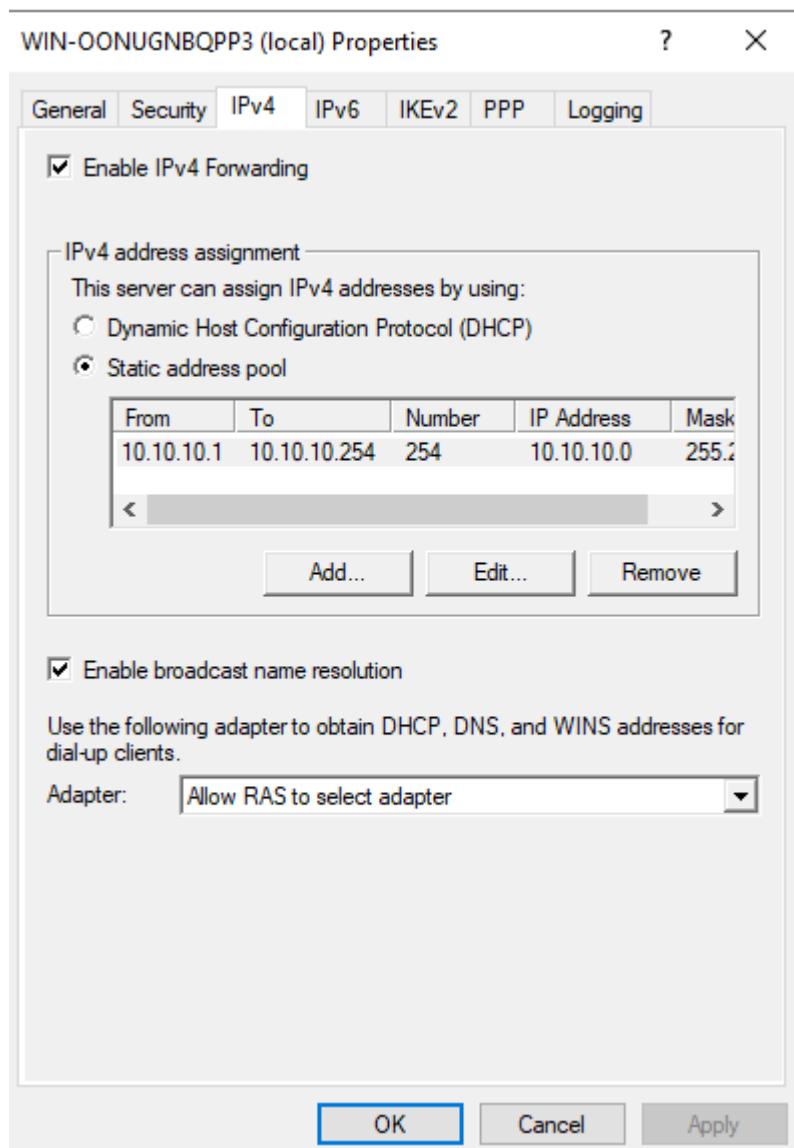
El método es como el de arriba, pero para cerciorarnos de que la VPN está habilitada, en Routing and Remote Access se le da click derecho al servidor y se asegura que esté marcado servicio de acceso remoto, se nos queda una pantalla como esta:



Asegurarse que en la pantalla puertos hay protocolos con RAS/Routing y sino click derecho y permitir inbound only.



También dar una pool para que asigne direcciones IP para la VPN:



Una vez habilitado, crear usuarios para probar (o asignarles permisos para usar VPN) y meterlos en un grupo.

Luego hay que configurar NPS (Network Policy Service) para autenticar a los clientes.

Ir a Network Policy Server → Policies → Network Policies → click derecho New policy. Se le da un nombre y se selecciona de tipo “Remote Access Server” o como se quiera llamar. Se le da a **siguiente** y se dice “Add” y se añade un Windows Group (el que usamos previamente). **Siguiente** y access granted. Le damos a siguiente y desmarcamos “MS-CHAP”. Y ya siguiente hasta finalizar. (AÑADIR YA SEA EN ESTE ASISTENTE O CLICK DERECHO SOBRE LA CREADA EN CONSTRAINTS → AUTHENTICACION METHODS y quitar la casilla de abajo MS-CHAP-v2)

Añadir, si procediera, apertura de puertos en el router ajeno al puerto 1723 (PPTP) y permiso de paso a protocolo GRE (iptables -p gre).

Regla firewall

```
New-NetFirewallRule -DisplayName "VPNTCP" -Direction inbound -Profile Any -Action Allow -LocalPort 1723,1701,443 -Protocol TCP  
New-NetFirewallRule -DisplayName "VPNUDP" -Direction inbound -Profile Any -Action Allow -LocalPort 500,4500,50,1701 -Protocol UDP  
New-NetFirewallRule -DisplayName "GRE" -Direction inbound -Protocol 47 -Profile Any -Action Allow
```

Añadir conexión VPN en Windows 10

Win-X click en Network Connections → VPN → Add a VPN connection y rellenar datos.

Site to Site VPN

Se le puede asignar IP de la misma red interna (en un rango no ocupado, por ejemplo: en 192.168.0.0/24 podríamos decir que empiece en la .253 y termine en la .254 siempre que no se estén usando ni se vayan a usar)

En Routing and Remote Access se hace click derecho sobre Network Interfaces y se añade New Demand-Dial Interface y se sigue el asistente, seleccionar las dos primeras después de poner la IP, añadir las rutas estáticas remotas, en DIAL IN se especifica contraseña para conectarse HACIA el equipo desde el que se configura, en DIAL-OUT se configuran las credenciales de acceso remotas. Hacer configuración en las dos máquinas.

Crear un usuario con el nombre que aparece en DIAL IN y en sus propiedades en Dial In darle a Allow Access.

Ansible

Configurar Windows para usar ansible

Crear un usuario local ansible y añadirlo al grupo Administrators (lusrmgr.msc)

Configurar WinRM (aparentemente en Windows Server 2022 está habilitado):

winrm get winrm/config/Service

winrm get winrm/config/Winrs

winrm enumerate winrm/config/Listener

Copias de seguridad de Windows Server

Añadir la función en Features (Características).

Luego en Tools → Windows Server Backup.

Local Backup (right click) → y lo que toque, no parece tener pérdida.

PowerShell

No es lo mismo usar comillas simples que comillas dobles. Para que procese la variable, usaremos **comillas dobles**.

```
PS C:\Users\Administrador> $algo="Hola mundo"
PS C:\Users\Administrador> Write-Output "Lo que contiene mi variable es $algo"
Lo que contiene mi variable es Hola mundo
PS C:\Users\Administrador> Write-Output 'Lo que contiene mi variable es $algo'
Lo que contiene mi variable es $algo
PS C:\Users\Administrador>
```

Habilitar ejecución de scripts

Si ves esta pantalla

```
C:\Users\mtm\Downloads\Windows10Debloater-master\Windows10Debloater-master\Windows10DebloaterGUI.ps1 porque la
ejecución de scripts está deshabilitada en este sistema. Para obtener más información, consulta el tema
about_Execution_Policies en https://go.microsoft.com/fwlink/?LinkId=135170.
En línea: 1 Carácter: 1
+ .\Windows10DebloaterGUI.ps1
+ ~~~~~
+ CategoryInfo          : SecurityError: () [], PSNotSupportedException
+ FullyQualifiedErrorId : UnauthorizedAccess
PS C:\Users\mtm\Downloads\Windows10Debloater-master\Windows10Debloater-master>
```

Ejecuta: `set-executionpolicy unrestricted`

Si nos dice algo sobre el fichero .ps1, ejecutar [Unblock-File](#).

1. Creación de usuarios en local

Con el comando `New-LocalUser` se pueden crear usuarios de forma local.

Un ejemplo de creación de un usuario sería:

```
$Password = Read-Host -AsSecureString
New-LocalUser -Name 'User02' -Description 'Descripción de la cuenta' -Password
$password
Add-LocalGroupMember -Group Users (o Administradores) -Member User02
```

1.1 Automatización de creación de usuarios

Para implementarlo en un script, surge el problema de la definición de una contraseña, ya que hay que pedir una común al inicio, o definirla en texto plano en el propio.

1.1.1 Contraseña en fichero

Para ponerla en el fichero, un script de ejemplo podría ser:

```
$Password=ConvertTo-SecureString "plain_text_passwd" -AsPlainText -Force
$Username=Read-Host "Escribe el nombre base de los usuarios"
$loop=Read-host "Escribe el nº de usuarios que quieras crear"

for($i=0; $i -lt $loop; $i++){
    New-LocalUser -Name "$algo $i" -Description 'Descripción de la cuenta' -Password
    $password
    Add-LocalGroupMember -Group Users (o Administradores) -Member User02
```

2. Creación de usuarios en Active Directory

Con el comando `New-ADUser` se pueden crear usuarios en el Active Directory.

Con el comando `Get-ADUser -Filter * | select samAccountName` podemos ver los usuarios creados en Active Directory.

```
PS C:\Users\Administrador> Get-ADUser -Filter * | select samAccountName
samAccountName
-----
Administrador
Invitado
krbtgt
pepe

PS C:\Users\Administrador>
```

Por defecto, al usar el comando `New-ADUser` a secas te crea el usuario en el grupo **Users/Usuarios**, deja **deshabilitada** la cuenta, es miembro de grupo **Domain Users** y el usuario tendrá que **cambiar la contraseña** en su primer login.

Para hacer una creación de usuario más elaborada se pueden poner los siguientes atributos:

```
New-ADUser -Name "Usuario2" -SamAccountName "Usuario2" -Path
"CN=Users,DC=skills39,DC=io" -AccountPassword(Read-Host -AsSecureString "Input
Password") -Enabled $true
```

Ahí la contraseña la pediría al momento de crear el usuario.

2.1 Importación de usuarios masiva por CSV

Para trabajar con CSV en PowerShell, el fichero csv podrá ser así:

```
username,firstname,lastname,password
jsmith,John,Smith,p@ssw3rd
janesmith,Jane,Smith,s3cret
joesmith,Joe,Smith,blackd0g
jimsmith,Jim,Smith,whlit3cAt
jacksmith,Jack,Smith,blu3FisH
johndoe,John,Doe,R3dHors3
janedoe,Jane,Doe,GreenFr0g
msanders,Michael,Sanders,YellowTurtle
boballen,Bob,Allen,PurpleSnake
jamesq,James,Quimbly,PinkPig
mscott,Michael,Scott,WhiteZebra
jhalprin,Jim,Halprin,BlackTurkey
tjones,Teresa,Jones,BlueHawk
tomjones,Tommy,Jones,GreenTree
greggjones,Gregg,Jones,OrangeFish
dthompson,Daniel,Thompson,RedBoat
pjones,Pamela,Jones,BlackSnake
fsmith,Fred,Smith,WhiteHorse
fransmith,Fran,Smith,YellowDuck
cthompson,Chris,Thompson,BlueEel
```

```
# Revisar estructura de CSV para crear el script
$csvPath="C:\Users\Administrador\Documents\psl_scripts\crear_users\users.csv"
$groupName="PS Test" # Se podría implementar su grupo en el csv con otro campo.
$users=Import-Csv -Path $csvPath

foreach ($user in $users) {
    New-ADUser -Name $user.Username -AccountPassword (ConvertTo-SecureString $user.Password
    -AsPlainText -Force) -GivenName $user.FirstName -Surname $user.LastName -Enabled $true
    Add-ADGroupMember -Identity $groupName -Members $user.Username
}
```

Desactivar política contraseñas.

Cisco

Can't switchport mode trunk

If you receive something like "Command rejected: An interface whose trunk encapsulation is "Auto" can not be configured to "trunk" mode.

% Range command terminated because it failed on GigabitEthernet0/0", you shall do:

```
#switchport trunk encapsulation dot1q
```

Para buscar coincidencias (similar a grep):

```
do sh ru | include <string>
```

Para que empiece desde la primera coincidencia:

```
do sh ru | begin <string>
```

Ancho de banda de broadcast

En la interfaz:

```
storm-control broadcast level 10
```

VLANs

DECLARAR SIEMPRE VLAN CON NOMBRE, sino puede haber problemas.

Voice VLAN

En la interfaz:

```
switchport voice vlan 21
```

Deshabilitar DTP

```
switchport nonegotiate
```

Recuperar contraseña

1. Apagar y encender, conforme está encendiendo Ctrl-C.
2. En rommon 1 escribir **confreg 0x2142**
3. En rommon 2 escribir **reset**
4. Al reiniciar escribir: **copy startup-config running-config**
5. Entrar en modo de configuración y escribir "**no enable password**" o "**no enable secret**".
6. Escribir **config-register 0x2102**
7. **copy running-config startup-config** en modo enable

Habilitar contraseña

```
enable secret <password>
```

NTP/Syslog/SNMP

```
Router0(config)# ntp server 192.168.1.221
Router0(config)# do show clock detail
Router0(config)# logging host 192.168.1.221
Router0(config)# logging trap debugging
```

SNMPv2

```
Router0(config)#snmp-server community SNMP-RW RW
Router0(config)#snmp-server community SNMP-RO RO
```

SNMPv3

```
snmp-server group [NOMBRE_GRUPO] v3 priv
snmp-server user [NOMBRE_USUARIO] [NOMBRE_GRUPO] v3 auth sha
[PASSWORD_AUTH] priv aes 128 [PASSWORD_PRIV]
```

SVI

Ejemplo:

```
switch> enable
switch# configure terminal
switch(config)# vlan 10
switch(config-vlan)# name Ventas
switch(config-vlan)# exit
switch(config)# interface vlan 10
switch(config-if)# ip address 192.168.10.1 255.255.255.0
switch(config-if)# no shutdown
switch(config-if)# end
switch# copy running-config startup-config
```

LLDP y CDP

Habilitar LLDP Globalmente

```
(config)# lldp run
```

Deshabilitar LLDP Globalmente

```
(config)# no lldp run
```

Habilitar LLDP en una Interfaz Específica

```
(config-if)# lldp transmit  
(config-if)# lldp receive
```

Deshabilitar LLDP en una Interfaz Específica

```
(config-if)# no lldp transmit  
(config-if)# no lldp receive
```

Verificar la Configuración de LLDP

```
# show lldp
```

Habilitar CDP Globalmente

```
(config)# cdp run
```

Deshabilitar CDP Globalmente

```
(config)# no cdp run
```

Habilitar CDP en una Interfaz Específica

```
(config-if)# cdp enable
```

Deshabilitar CDP en una Interfaz Específica

```
(config-if)# no cdp enable
```

Verificar la Configuración de CDP

```
# show cdp
```

Y para ver información detallada sobre los vecinos descubiertos a través de LLDP, usa:

```
# show cdp neighbors
```

Acceso terminal o ssh

SSH:

```
Switch(config)# ip domain-name spainskills.es  
Switch(config)# crypto key generate rsa  
Switch(config)# ip ssh version 2  
Switch(config)# line vty 0 15  
Switch(config-line)# login local  
Switch(config-line)# transport input ssh  
Switch(config-line)# exit  
Switch(config)# username pepe123 privilege 15 secret <contraseña>
```

Copia seguridad y restauración de sistema IOS

```
R1# copy running-config tftp:
```

Actualización IOS

```
Router300#copy tftp: flash:  
Address or name of remote host []? 2001:db8:2:0:202:16ff:fe6b:650a  
Source filename []? c1841-ipbasek9-mz.124-12.bin  
Destination filename [c1841-ipbasek9-mz.124-12.bin]?  
Router300#conf t  
Enter configuration commands, one per line. End with CNTL/Z.  
Router300(config)#boot system flash:c1841-ipbasek9-mz.124-12.bin  
Router300#reload
```

VTP

```
Switch(config)#int r g0/1-2  
Switch(config-if-range)#switchport mode trunk # Importante  
switch(config)#vtp version 2  
switch(config)#vtp mode server  
switch(config)#vtp domain pepe  
switch(config)#vtp password mi_super_password
```

Spanning-Tree

```
spanning-tree mode pvst/rapid-pvst  
spanning-tree vlan 10 root primary/secondary  
spanning-tree vlan 10 priority 4096  
  
En la interfaz habilitar spanning-tree portfast y BPDU guard para que no espere y notifique STP .
```

INCLUIR VLAN NATIVA

(sobre interfaz) spanning-tree guard root # PREVENIR QUE UN SWITCH CONECTADO A ESE PUERTO SE CONVIERTA EN ROOT

Port security (MAC)

```
Switch(config)# int g0/2  
Switch(config-if)# switchport mode access  
Switch(config-if)# switchport port-security # HABILITAR  
Switch(config-if)# switchport port-security maximum 3  
Switch(config-if)# switchport port-security mac-address 0001.64bd.2363  
Switch(config-if)# switchport port-security violation restrict # EN CASO DE VIOLACIÓN, RESTRINGE TRÁFICO Y LO NOTIFICA.  
Switch(config-if)# switchport port-security violation protect # EN CASO
```

DE VIOLACIÓN, RESTRINCE TRÁFICO Y NO LO NOTIFICA.

Switch(config-if)# switchport port-security violation shutdown # EN CASO DE VIOLACIÓN, APAGA INTERFAZ.

DHCP server (router)

```
Router0(config)#ip dhcp pool VLAN10
Router0(dhcp-config)#network 192.168.0.0 255.255.255.0
Router0(dhcp-config)#default-router 192.168.0.254
Router0(config)#ip dhcp excluded-address 192.168.0.254 #EXCLUIR SIEMPRE ROUTER
Router0(config)#ip dhcp excluded-address 192.168.0.250 192.168.0.254 # Rango
```

DHCP relay

Se configura el DHCP donde toque y en la interfaz del **router más cercano** a la red destino (la que hará las peticiones DHCP), se configura el helper-address.

```
Router1(config-if)#int g0/0
Router1(config-if)#ip helper-address 192.168.1.229
```

PPP

```
username <nombre_router_externo> password pepe
int se1/0
ip addr loquesea
encapsulation ppp
```

PAP:

```
ppp encapsulation pap
ppp pap sent-username <hostname_router_local> password pepe
```

CHAP:

```
ppp encapsulation chap
ppp
```

ACLs

Para ver las ACLs:

```
R1# show access-lists
```

Aplicar sobre **subinterfaces** si existieran.

Standard:

Lo más cercanas al destino porque solo se configura sobre dirección de origen. Una vez escrita, **aplicar sobre la interfaz correspondiente** (por la que entre/salga el paquete)

```
R1# access-list 1 permit 192.168.0.0 0.0.0.255
```

```
R1# access-list 1 deny any
Router2(config)#int g3/0
Router2(config-if)#ip access-group 1 in
```

Extended:

```
Router0(config)#access-list 100 permit tcp 192.168.0.0 0.0.0.255 192.168.1.233 0.0.0.0 eq
80
Router0(config)#access-list 100 permit tcp 192.168.1.0 0.0.0.127 192.168.1.226 0.0.0.0 eq
80
Router0(config)#access-list 100 deny tcp any 192.168.1.233 0.0.0.0 eq 80
Router0(config)#access-list 100 deny tcp any 192.168.1.226 0.0.0.0 eq 80
```

Named:

```
RC(config)#ip access-list extended nombre
RC(config-ext-nacl)#deny tcp any 192.168.1.233 0.0.0.0 eq 80
RC(config-if)#ip access-group nombre in
```

RIP

```
router rip
version 2
no auto-summary
passive-interface g0/0 # A las interfaces que no usen RIP. CONFIGURAR SIEMPRE
default-information originate # Compartir ruta default (estática previamente configurada)
network 192.168.0.0 # Habilita RIP en las interfaces con esa red y la publica.
```

RIPng

```
int se0/1/0
ipv6 rip id1 enable # En las interfaces con ruta a compartir.
exit
ipv6 router rip id1
redistribute connected # Publicar redes conectadas directamente.
```

BGP

```
router bgp 65000 (NUMERO PUEDE SER DIFERENTE)
neighbor 10.0.0.1 remote-as 65001
network 192.168.1.0 mask 255.255.255.0
neighbor 10.0.0.1 route-map out
router bgp 100
  bgp log-neighbor-changes
  no synchronization
  neighbor 10.10.10.2 remote-as 200
  neighbor 10.10.20.2 remote-as 300
  network 192.168.0.0
```

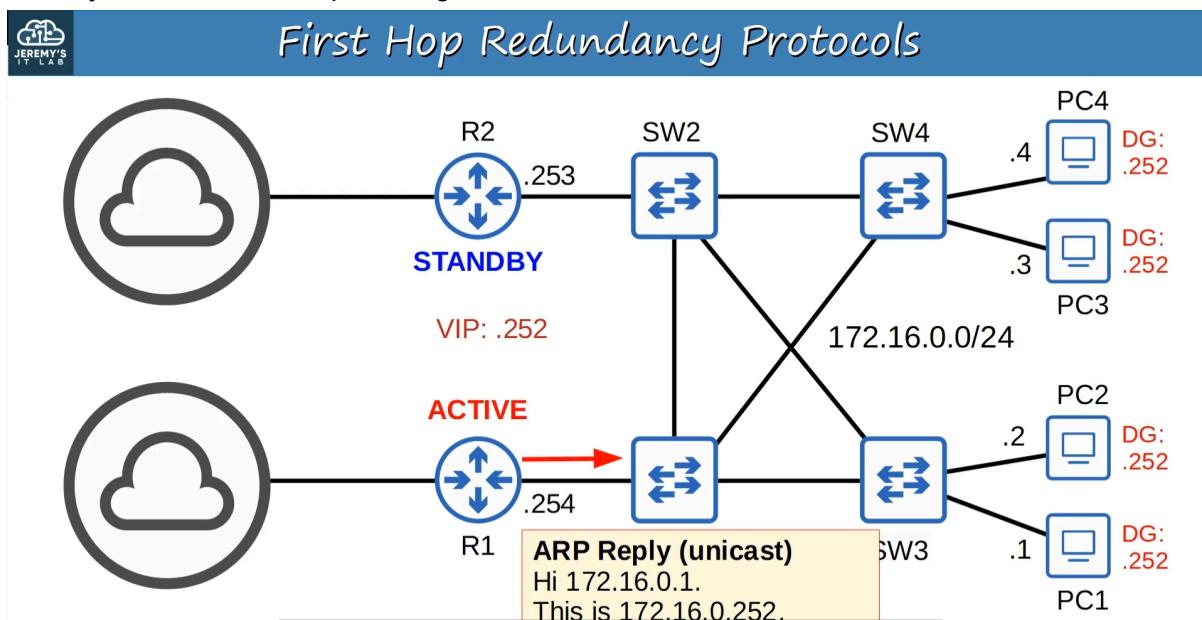
EIGRP

```
router eigrp 65000 (MISMO NUMERO PARA TODOS)
eigrp router-id id(el ID tiene formato de ipv4)
network 192.168.0.0
network 192.168.0.0 0.0.0.255 # Para concretar la red
no auto-summary (?) # Lo más probable es que haya que usarlo.
```

```
router eigrp 100
passive-interface FastEthernet0/0
network 10.10.10.0 0.0.0.3
network 10.10.30.0 0.0.0.3
network 192.168.1.0
auto-summary
```

FHRP

FHRP (First Hop Redundancy Protocol) comprende la alta disponibilidad de los routers, y funciona con los routers, que teniendo su propia IP, tienen una IP virtual y se comunican por mensajes multicast hello para asignarse sus roles.



También comparten dirección MAC virtual.

Src IP: 172.16.0.1
Dst IP: 8.8.8.8
Src MAC: PC1's MAC
Dst MAC: Virtual MAC

Se definen routers **active** y **standby** (según cada protocolo tendrá un nombre distinto).

Si un router caído vuelve, por defecto no volverá a ser el activo y será standby. Para que vuelva a ser el activo, hay que configurar **preemption**.

Diferentes protocolos

HSRP

Protocolo propietario de Cisco, en el que se elige un router *active* y *standby*. Hay dos versiones, la versión 2 soporta IPv6 e incrementa el número de grupos (en v1 hay 256 grupos, mientras que en v2 hay 4096) que se pueden configurar (ya que habrá diferentes VLANs en las que definir el gateway). Por defecto, está activada la **versión 1** (las versiones no son compatibles entre sí, por lo que ambos routers han de estar configurados con la misma versión).

El router **activo** es aquel con la prioridad **más alta**, y si comparten prioridad, será el que tenga la dirección **IP más alta**.

Las direcciones multicast en las que se comunican son:

- v1: 224.0.0.2
- v2: 224.0.0.102

Las direcciones MAC virtuales serán:

- v1: **00:00:0c:07:ac:XX** (siendo XX el número del grupo HSRP)
 - Por ejemplo: 00:00:0c:07:ac:01 sería el primer grupo.
- v2: **00:00:0c:9f:fXXX** (siendo XXX el número de grupo HSRP)

Por cada grupo HSRP se puede configurar un router activo **distinto**, de cara a hacer balance de carga.

Configuración HSRP

En el router, se accede a la interfaz que comunica a la LAN y se configura con el comando **standby**

```
R1(config-if)#standby 1 ip 172.16.0.254
R1(config-if)#
R1(config-if)#standby 1 priority ?
<0-255> Priority value

R1(config-if)#standby 1 priority 200
R1(config-if)#
R1(config-if)#standby 1 preempt
R1(config-if)#[
```

En el primer comando se define la IP virtual que usará HSRP en el grupo virtual, después se configura la prioridad que tendrá el router en cuestión y por último (esto **solo** hace falta configurarlo en el **router activo**, aunque si se configura en el secundario **no pasa nada**) se permite que si el router cae y vuelve a la vida, pueda recuperar su rol **activo**.

Cuando pedimos información sobre HSRP, esta será la salida

```

R1#show stand
FastEthernet0/1 - Group 1
  State is Init (interface down)
  Virtual IP address is 172.16.15.254
  Active virtual MAC address is unknown
    Local virtual MAC address is 0000.0C07.AC01 (v1 default)
  Hello time 3 sec, hold time 10 sec
    Next hello sent in 1.180 secs
  Preemption disabled
  Active router is unknown
  Standby router is unknown
  Priority 100 (default 100)
  Group name is hsrp-Fa0/1-1 (default)
R1#

```

OSPF

Es un protocolo de **enrutamiento por estado de enlace**; en este tipo de protocolo cada **router crea un mapa de “conectividad”** de la red, para esto cada router **anuncia información sobre sus interfaces** (redes conectadas) a sus vecinos y estos anuncios son pasados a otros routers hasta que **todos los routers de la red tienen el mismo mapa de la red**. Con eso, cada router usa el mapa para **calcular las mejores rutas a cada destino**.

Los routers almacenan esta información en LSAs (Link State Advertisements) que se organizan en LSDB (Link State Database)

Los routers inundarán la red de **LSAs** hasta que todos los routers del área tengan el mismo mapa de la red, es decir, hasta que **todos tengan el mismo LSDB**.

En OSPF hay tres pasos principales a la hora de compartir LSAs y determinar la mejor ruta para cada destino:

1. **Hacerse vecinos** con otros routers del mismo segmento
2. **Intercambiar LSAs** con los routers vecinos
3. Cada router calculará **independientemente** las mejores rutas a cada destino y las **insertará** en la tabla de rutas.

Áreas

Las áreas de OSPF se utilizan para segmentar la red, pero si hay redes pequeñas se puede usar una sola área.

Se puede definir como: un conjunto de routers y enlaces que comparten la misma LSDB.

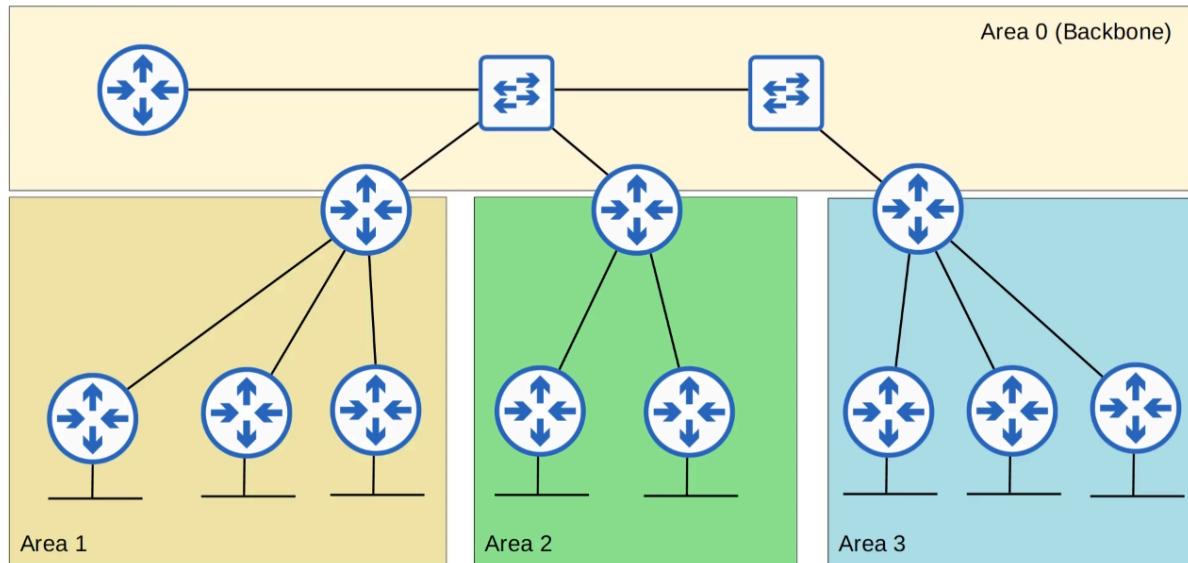
No se permite conectar áreas entre sí, tienen que pasar todas por el área 0 (backbone).

Los routers que se encuentren en el mismo área son **routers internos**, mientras que los routers en diferentes áreas son llamados **routers de área perimetral** (Area Border Router) y estos mantienen una LSDB distinta por cada área; los routers conectados al backbone son **routers de red troncal** (incluso si son internos); una ruta cuyo destino están dentro de la **misma área** se llaman **ruta intra área**; mientras que si es a otra área se llama **ruta inter área**.

Las áreas tienen que ser contiguas (no puede haber ABR con la misma área en el backbone).

Todas las interfaces OSPF en la **misma subred** estarán en la **misma área**.

Una división de áreas podría ser:



Configuración OSPFv2 (IPv4)

Para configurar OSPF (en IPv4, para IPv6 ir [aquí](#)) primero se configura un proceso OSPF en el router

```
R1(config)#router ospf 1
```

(da igual el ID que se escoja y no tiene que ser el mismo en todos los routers)

OSPF utiliza máscaras wildcard (las inversas) y hay que definir un área **siempre**, siendo la 0 la troncal (o la única si usamos monoárea) y se **activa OSPF** en la **interfaz de la red** en el área especificada y los routers intentan convertirse en vecinos con otros con OSPF.

(NO DICE QUE PUBLIQUE ESAS REDES)

```
R1(config-router)#network 10.0.12.0 0.0.0.3 area 0
```

Con el comando `passive-interface interfaz` se especifica que no envíe paquetes OSPF 'hello' por esa interfaz, pero enviará paquetes LSA informando de la red configurada en esa interfaz (**USAR EN INTERFACES NO-OSPF**)

```
R1(config-router)#passive-interface g2/0
```

Para publicar una ruta **default** (a Internet normalmente) se usa el comando

```
R1(config-router)#default-information originate
```

Para publicar rutas **estáticas o conectadas directamente**:

```
Router2(config-router)# redistribute static subnets  
Router2(config-router)# redistribute connected subnets
```

Para ver la información de OSPF se usa el siguiente comando:

```
R1#show ip protocols
```

Para configurar el Router ID se puede hacer de los siguientes modos (y en esta prioridad se interpretará):

1. Configuración manual

```
R1(config-router)#router-id 1.1.1.1
```

Para que tome efecto el cambio se pueden hacer lo siguiente:

- a. Resetear (no borra config), pero tarda ya que borra todas las rutas OSPF.

```
R1#clear ip ospf process
```

2. Dirección IP más alta en loopback
3. Dirección IP más alta en interfaz física

Para cambiar el bandwidth de referencia (para tener costes adecuados): (CAMBIAR EN TODOS)

```
tauto-cost reference-bandwidth 100000
```

Tres formas de configurar OSPF

1. Especificando las redes de las interfaces:

```
router ospf 10
  router-id 1.1.1.1
  log-adjacency-changes
  passive-interface GigabitEthernet0/0/0
  network 192.168.10.0 0.0.0.255 area 0
  network 10.1.1.0 0.0.0.3 area 0
  network 10.1.1.4 0.0.0.3 area 0
```

2. Con máscara quad-zero:

```
router ospf 10
  router-id 2.2.2.2
  log-adjacency-changes
  passive-interface GigabitEthernet0/0/0
  network 192.168.20.1 0.0.0.0 area 0
  network 10.1.1.2 0.0.0.0 area 0
  network 10.1.1.9 0.0.0.0 area 0
```

3. Especificándolo dentro de la propia interfaz:

```
interface GigabitEthernet0/0/0
  ip address 192.168.30.1 255.255.255.0
  ip ospf 10 area 0
  duplex auto
  speed auto
!
interface GigabitEthernet0/0/1
  no ip address
  duplex auto
  speed auto
  shutdown
!
interface Serial0/1/0
  ip address 10.1.1.10 255.255.255.252
  ip ospf 10 area 0
!
interface Serial0/1/1
  ip address 10.1.1.6 255.255.255.252
  ip ospf 10 area 0
!
interface Vlan1
  no ip address
  shutdown
!
router ospf 10
  router-id 3.3.3.3
  log-adjacency-changes
  passive-interface GigabitEthernet0/0/0
```

OSPFv3

OSPFv3 usa la dirección multicast FF02::5 para publicar la información mientras que usa la FF02::6 para los routers designados y de backup designados.

Para configurar OSPFv3 en un router con IPv6, se usa el siguiente comando:

```
R1(config)#ipv6 router ospf 1
```

Una vez dentro, se define un ID de router (el ID tiene formato de dirección IPv4)

```
R1(config-rtr)#router-id 1.1.1.1
```

(si en el mismo router se usa OSPF en v4 y v6 se puede usar el mismo router ID)

Para añadir una interfaz al proceso OSPF hay que, desde la configuración de la interfaz, poner el siguiente comando:

```
R1(config-if)#ipv6 ospf 1 area 1
```

Ejemplo:

```
interface GigabitEthernet0/0
  no ip address
  duplex auto
  speed auto
  ipv6 address FE80::A link-local
  ipv6 address 2001:DB8:1:A1::1/64
  ipv6 ospf 1 area 1
!
interface GigabitEthernet0/1
  no ip address
  duplex auto
  speed auto
  ipv6 address FE80::A link-local
  ipv6 address 2001:DB8:1:A2::1/64
  ipv6 ospf 1 area 1
!
interface GigabitEthernet0/2
  no ip address
  duplex auto
  speed auto
  shutdown
!
interface Serial0/0/0
  no ip address
  ipv6 address FE80::A link-local
  ipv6 address 2001:DB8:1:AB::2/64
  ipv6 ospf 1 area 0
!
interface Serial0/0/1
  no ip address
  clock rate 2000000
  shutdown
!
interface Vlan1
  no ip address
  shutdown
!
ipv6 router ospf 1
  router-id 1.1.1.1
  log-adjacency-changes
```

NAT

Para hacer NAT a Internet, en la interfaz que da a internet:

SIEMPRE INDICAR NAT EN INTERFAZ

```
R1(config-if)# ip nat outside # En la interfaz que da a internet
R1(config-if)# ip nat inside # Interfaz que es red interna
```

R1(config)# access-list 100 permit ip 172.16.0.0 0.0.0.255 any # Crear ACL extendida para permitir la salida de red

R1(config)# ip nat inside source list 100 interface serial 0/0 overload

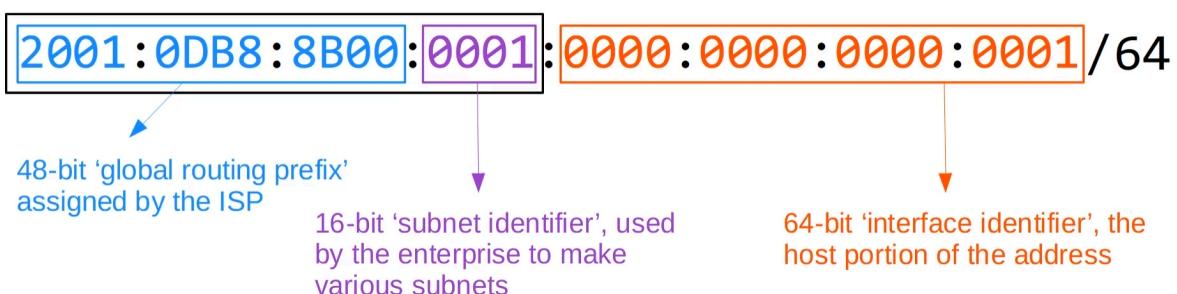
Redirección de puertos (DNAT)

```
Router1(config)#ip nat inside source static tcp 192.168.1.189 80 192.168.1.233 80
```

IPv6

Una dirección IP está compuesta por dos bloques principales:

- Un prefijo, compuesto por dos partes y delimitado por el /xxx (siendo xxx el número de prefijo):
 - Prefijo asignado por ISP
 - Identificador de subred
- Identificador de host
- **LA 16º DIRECCIÓN ES ::10, NO ::16.**



Tipos de direcciones IPv6

Multicast

Purpose	IPv6 Address	IPv4 Address
All nodes/hosts (functions like broadcast)	FF02::1	224.0.0.1
All routers	FF02::2	224.0.0.2
All OSPF routers	FF02::5	224.0.0.5
All OSPF DRs/BDRs	FF02::6	224.0.0.6
All RIP routers	FF02::9	224.0.0.9
All EIGRP routers	FF02::A	224.0.0.10

Anycast

De uno a uno de varios. Por ejemplo:

- Multiple routers are configured with the same IPv6 address.
 - They use a routing protocol to advertise the address.
 - When hosts sends packets to that destination address, routers will forward it to the nearest router configured with that IP address (based on routing metric).

```
R1(config-if)# ipv6 address 2001:db8:1:1::99/128 anycast
```

Configuración IPv6 en Cisco

Para habilitar el enrutamiento

```
R1(config)#ipv6 unicast-routing
```

Para configurar una dirección IPv6:

```
R1(config-if)#ipv6 address 2001:db8:0:1::1/64
```

```
R2(config-if)#ipv6 address autoconfig # LINK LOCAL
```

Para mostrar la configuración IPv6

```
R1#show ipv6 interface brief
GigabitEthernet0/0      [up/up]
    FE80::EF8:22FF:FE36:8500
    2001:DB8::1
GigabitEthernet0/1      [up/up]
    FE80::EF8:22FF:FE36:8501
    2001:DB8:0:1::1
GigabitEthernet0/2      [up/up]
    FE80::EF8:22FF:FE36:8502
    2001:DB8:0:2::1
GigabitEthernet0/3      [administratively down/down]
    unassigned
R1#
```

Para configurar la IP estática con EUI64 (MAC) se usa lo siguiente

```
R1(config)#int g0/0
R1(config-if)#ipv6 address 2001:db8::/64 eui-64
R1(config-if)#no shutdown
```

Enrutamiento estático IPv6

El enrutamiento IPv6 **NO** está activado por defecto en Cisco, para activarlo

```
R1(config)#ipv6 unicast-routing
```

Para ver las rutas:

```
R1#show ipv6 route
```

Para añadir rutas:

```
ipv6 route destination/prefix-length {next-hop | exit-interface [next-hop]} [ad]
```

(ad es distancia administrativa, para rutas flotantes)

Hay varios tipos de rutas:

- Ruta estática **conectada directamente**: solo se especifica la interfaz de salida
 - Ejemplo: R1(config)# ipv6 route 2001:db8:0:3::/64 g0/0
 - **NO SE PUEDEN USAR SI ES INTERFAZ ETHERNET EN IPv6**, si es SERIAL sí.
- Ruta estática **recursiva**: solo se especifica el siguiente salto (dirección IP).
 - Ejemplo:
R1(config)# ipv6 route 2001:db8:0:3::/64 2001:db8:0:12::2
 - Se llama recursiva porque tiene que comprobar más de una vez en su tabla de rutas para saber a dónde dirigir el paquete.
- Ruta estática **completamente especificada**: se especifica tanto la interfaz de salida como la dirección del siguiente salto.
 - Ejemplo:
R1(config)# ipv6 route 2001:db8:0:3::/64 g0/0 2001:db8:0:12::2
 - Si se enruta una dirección de **enlace local**, se usa este método.
- Ruta estática **flotante** (o de respaldo): se suele usar con enrutamiento dinámico
 - Si se usan con OSPF: ad > 110
 - Si se usan con EIGRP: ad > 90
 - Si se usan rutas estáticas: ad > 1

Una ruta **default** se configura tal que así:

```
R3(config)# ipv6 route ::/0 2001:db8:0:23::1
```

SLAAC

```
int g0/0
```

```
ipv6 nd ra interval 200 # Puede que no esté, indica intervalo anuncio RA.
```

DHCPv6

```
ipv6 unicast-routing
ipv6 dhcp pool pool-v6
address prefix 2002:db8:1::/64
dns-server 8:8:8:8::1
interface GigabitEthernet 0/1
```

```
ipv6 dhcp server pool-v6  
ipv6 nd managed-config-flag
```

MTU

(en interfaz) mtu

VPN

Site-site

IPSEC

En un router:

```
# Configuring Phase I/ISAKMP Tunnel Parameters

crypto isakmp policy 5
auth pre-share
encryption 3des
hash md5
group 2
exit
crypto isakmp key cisco1234 address 23.1.1.3

# Configure Phase II / IPsec Tunnel [ESP]

crypto ipsec transform-set TSET esp-3des esp-sha-hmac

# Configure the Traffic need to be encrypted - Crypto Traffic or
Interesting Traffic

access-list 101 permit ip 10.1.1.0 0.0.0.255 10.3.3.0 0.0.0.255 # Aquí
se indica la red de origen (la de los equipos) y la de destino.

# Link the above 3 steps together in a parameter called as Crypto map

crypto map CMAP 10 ipsec-isakmp
match address 101
set peer 23.1.1.3
set transform-set TSET

# Apply the Crypto Map to the outgoing interface

int e0/0
crypto map CMAP
```

En el otro:

```
crypto isakmp policy 5
auth pre-share
encryption 3des
hash md5
group 2
!
crypto isakmp key cisco1234 address 12.1.1.1

# !2. Configure Phase II / IPSec Tunnel [ESP]

crypto ipsec transform-set TSET esp-3des esp-sha-hmac

# !3. Configure the Traffic need to be encrypted - Crypto Traffic or Interesting Traffic

access-list 101 permit ip 10.3.3.0 0.0.0.255 10.1.1.0 0.0.0.255

# !4. Link the above 3 steps together in a parameter called as Crypto map

crypto map CMAP 10 ipsec-isakmp
match address 101
set peer 12.1.1.1
set transform-set TSET

# !5. Apply the Crypto Map to the outgoing interface

int e0/1
crypto map CMAP
# AÑadir la ruta a la interfaz de salida
ip route 10.1.1.0 255.255.255.0 12.1.1.1

# DEBUG
sh crypto ipsec sa
sh crypto isakmp sa

debug
```

GRE

Hacer lo siguiente en los dos routers:

```
interface Tunnel0 # Se crea al ponerlo.
ip address laquesea
tunnel source <interfaz> # No se pq interfaz, si no aparece esa opción, usa la IP de la int.
```

```
tunnel destination <ip_otro_extremo>
```

Client to site

<https://www.rmtechcentral.com/configuring-a-client-to-site-ipsec-vpn-tunnel-on-a-cisco-isr-router/>

```
ip local pool VPNPOOL 192.168.0.1 192.168.1.254
aaa new-model
aaa authentication login VPN-USERS group radius
aaa authorization network VPN-GROUP group radius
radius-server host 172.18.124.96 auth-port 1645
radius-server key cisco123
crypto isakmp policy 3
encr aes
hash sha
authentication pre-share
group 5
crypto isakmp client configuration group VPN-GROUP
key grupo
pool VPNPOOL
crypto ipsec transform-set VPNSET esp-aes esp-sha-hmac
crypto dynamic-map VPN-DYN 10
reverse-route
set transform-set myset
crypto map VPN-STATIC client configuration address respond
crypto map VPN-STATIC client authentication list VPN-USERS
crypto map VPN-STATIC isakmp authorization list VPN-GROUP
crypto map VPN-STATIC 20 ipsec-isakmp dynamic VPN-DYN
int f0/0 # Interfaz a la que se conectan los clientes
crypto map VPN-STATIC
access-list 108 permit ip <red_destino> 0.0.255.255 <red_vpn> 0.0.0.255
radius-server host 172.18.124.96 auth-port 1645 acct-port 1646 key cisco123
```

Python

Librerías

os: Proporciona una manera portátil de utilizar funcionalidades dependientes del sistema operativo, como:

- Navegar y manipular el sistema de archivos (crear, eliminar archivos y directorios).
- Obtener información del sistema operativo.
- Manejar variables de entorno.
- Ejecutar comandos del sistema.

sys: Ofrece **acceso a algunas variables utilizadas o mantenidas por el intérprete de Python** y a funciones que interactúan fuertemente con el intérprete. Es útil para **manipular la salida del script** o acceder a la configuración del sistema Python, como:

- Argumentos de línea de comandos.
- Salida estándar, entrada estándar, y error estándar.
- Salir del script con un código específico.

3. subprocess: Permite **ejecutar nuevos procesos**, conectar a sus tuberías de entrada/salida/error, y obtener sus códigos de retorno. Esta biblioteca es **esencial para la ejecución de comandos del sistema operativo** y scripts externos.

4. shutil: Ofrece varias operaciones de alto nivel en archivos y colecciones de archivos. Incluye soporte para copiar y mover archivos, así como otras operaciones de archivos, como:

- Copiar, mover, y eliminar directorios y archivos.
- Consultar y cambiar metadatos de archivos.
- Encontrar archivos en un directorio.

5. pathlib: Introducida en Python 3.4, esta biblioteca orientada a objetos hace que **trabajar con rutas de archivos y directorios sea más intuitivo y fácil de manejar** que las cadenas de texto tradicionales usadas para representar rutas. Proporciona clases para manejar el sistema de archivos de manera sencilla.

6. glob: Utilizada para **buscar archivos que coincidan con un patrón específico**, siguiendo las reglas de coincidencia usadas por los shells de Unix. Es útil para listar archivos en un directorio que coincidan con un patrón de nombre específico.

7. tempfile: Crea **archivos y directorios temporales**. Los archivos temporales se pueden utilizar para almacenar datos que solo se necesitan durante la ejecución de un programa, sin afectar el sistema de archivos permanente.

8. platform: Proporciona una **API** para obtener **información detallada sobre el sistema operativo, el hardware, y la versión del intérprete de Python** que se está ejecutando. Es útil para ajustar el comportamiento del script en función del entorno de ejecución.

Ejemplos:

1. Listar usuarios:

```
import subprocess
import re

def listar_usuarios():
    # Ejecutar el comando `getent passwd` para obtener información de los usuarios
    resultado = subprocess.run(['getent', 'passwd'],
                               stdout=subprocess.PIPE, text=True)
```

```

# Lista para almacenar los nombres de usuario
usuarios = []

    # Usar expresiones regulares para extraer los nombres de usuario
de la salida
    for linea in resultado.stdout.split('\n'):
        # La información de cada usuario en la salida de `getent passwd`
está separada por ':'
        # El primer campo es el nombre de usuario
        match = re.match(r'([:^]+):', linea)
        if match:
            usuarios.append(match.group(1))

    return usuarios

# Llamar a la función y imprimir los usuarios
for usuario in listar_usuarios():
    print(usuario)

```

2. Listar tamaños de directorios home:

```

import subprocess
import os

def listar_tamanio_directorios_home():
    # Ruta al directorio /home
    home_path = '/home'

    # Obtener una lista de todos los directorios en /home
    directorios = [d for d in os.listdir(home_path) if
os.path.isdir(os.path.join(home_path, d))]

    # Para cada directorio, ejecutar 'du' y obtener el tamaño del
directorío
    for directorio in directorios:
        # Ruta completa al directorio
        dir_path = os.path.join(home_path, directorio)

        # Ejecutar 'du -sh', que muestra el tamaño total del directorio de
manera legible (e.g., KB, MB, GB)
        resultado = subprocess.run(['du', '-sh', dir_path],
stdout=subprocess.PIPE, text=True)

        # Imprimir el resultado. strip() elimina el salto de línea al
final.

```

```
print(resultado.stdout.strip())
```

```
# Llamar a la función  
listar_tamanio_directorios_home()
```

3. Crear usuarios desde fichero .csv:

Fichero csv:

```
username,fullname,shell  
usuario1,Usuario Uno,/bin/bash  
usuario2,Usuario Dos,/bin/bash
```

Programa Python:

```
import csv  
import subprocess  
  
def crear_usuarios_desde_csv(ruta_csv):  
    with open(ruta_csv, newline='') as archivo_csv:  
        lector = csv.DictReader(archivo_csv)  
        for fila in lector:  
            # Generar el comando para agregar el usuario  
            # Ajusta este comando según las necesidades específicas y la  
            # estructura de tu CSV  
            comando = [  
                "sudo", "useradd",  
                "-m", # Crear el directorio de inicio para el usuario  
                "-s", fila["shell"], # Establecer el shell del  
                # usuario  
                "-c", f"\'{fila['fullname']}\'", # Establecer el  
                # nombre completo (campo GECOS) del usuario  
                fila["username"] # El nombre de usuario  
            ]  
  
            # Ejecutar el comando  
            resultado = subprocess.run(comando, stdout=subprocess.PIPE,  
            stderr=subprocess.PIPE, text=True)  
  
            # Verificar si el comando fue exitoso  
            if resultado.returncode == 0:  
                print(f"Usuario {fila['username']} creado  
exitosamente.")  
            else:  
                print(f"Error al crear usuario {fila['username']}:  
{resultado.stderr}")  
  
# Ruta al archivo CSV
```

```

ruta_csv = "ruta/a/tu/archivo.csv"

# Llamar a la función
crear_usuarios_desde_csv(ruta_csv)

```

4. Listar procesos con cpu y memoria (Linux y Windows)

```

import psutil

def listar_procesos():
    # Encabezado
    print(f"{'PID':<10}{Nombre:<25}{CPU %:<10}{Memoria %:<10}")
    for proceso in psutil.process_iter(attrs=['pid', 'name',
    'cpu_percent', 'memory_percent']):
        try:
            pid = proceso.info['pid']
            nombre = proceso.info['name']
            cpu_percent = proceso.info['cpu_percent']
            memoria_percent = proceso.info['memory_percent']

            print(f"{pid:<10}{nombre:<25}{cpu_percent:<10.2f}{memoria_percent:<10.2f}")
        except (psutil.NoSuchProcess, psutil.AccessDenied,
        psutil.ZombieProcess):
            pass # Ignorar procesos a los que no se puede acceder

listar_procesos()

```

5. Añadir un usuario a un AD (Windows Server)

```

from pyad import aduser, adcontainer

# Asegúrate de tener los permisos adecuados para ejecutar estas
operaciones
# y de estar ejecutando el script en un entorno que tenga acceso a tu
AD.

try:
    # Especifica el contenedor de AD donde deseas crear el nuevo
    # usuario
    ou =
    adcontainer.ADContainer.from_dn("OU=Usuarios,DC=ejemplo,DC=com")

    # Crear nuevo usuario
    nuevo_usuario = aduser.ADUser.create("nuevousuario", ou,

```

```

password="UnaContraseñaMuySegura123", upn_suffix="ejemplo.com",
enable=True, optional_attributes={"givenName": "Nombre", "sn":
"Apellido", "displayName": "Nombre Apellido", "mail":
"email@ejemplo.com", "telephoneNumber": "1234567890"})

    print("Usuario creado exitosamente.")
except Exception as e:
    print(f"Error al crear el usuario: {e}")

```

6. Añadir usuarios desde csv a un AD (Windows Server)

Usuarios.csv

```

username,firstname,lastname,password,email,phonenumber,group
jdoe,Jane,Doe,Password123!,jane.doe@example.com,555-0101,Group1
asmith,Adam,Smith,Password123!,adam.smith@example.com,555-0102,Group2

```

Programa

```

import csv
from pyad import aduser, adcontainer, adgroup, pyad

# Configura el dominio de Active Directory
pyad.set_defaults(ldap_server="dc.example.com",
username="admin@example.com", password="adminpassword")

# Función para crear usuarios y añadirlos a los grupos
def crear_usuarios_y_añadir_a_grupos(ruta_csv):
    with open(ruta_csv, newline='') as archivo:
        lector = csv.DictReader(archivo)
        for fila in lector:
            try:
                # Especifica el contenedor de AD donde deseas crear el
                nuevo_usuario
                ou =
adcontainer.ADContainer.from_dn("OU=Usuarios,DC=example,DC=com")

                # Crear nuevo usuario
                nuevo_usuario = aduser.ADUser.create(fila['username'],
ou, password=fila['password'], enable=True, optional_attributes={
                    "givenName": fila['firstname'],
                    "sn": fila['lastname'],
                    "displayName": f"{fila['firstname']} {fila['lastname']}"})

```

```

        "mail": fila['email'],
        "telephoneNumber": fila['phonenumber']
    })

    # Añadir el usuario al grupo especificado
    grupo = adgroup.ADGroup.from_cn(fila['group'])
    grupo.add_members([nuevo_usuario])

    print(f"Usuario {fila['username']} creado y añadido al
grupo {fila['group']} exitosamente.")
except Exception as e:
    print(f"Error al crear el usuario {fila['username']}:
{e}")

# Ruta al archivo CSV
ruta_csv = "ruta/a/tu/archivo.csv"

# Crear usuarios y añadirlos a grupos
crear_usuarios_y_añadir_a_grupos(ruta_csv)

```

7. Realización de copia desde Python

```

backup_script.sh

import os
import shutil
from datetime import datetime

# Ruta al directorio /home que quieras respaldar
home_dir_path = '/home'

# Ruta al directorio donde se guardarán los backups
backup_dest_path = '/path/to/your/backup/directory'

# Generar un nombre de archivo de backup con la fecha
backup_file_name =
f"backup-{datetime.now().strftime('%Y-%m-%d')}.tar.gz"

# Ruta completa al archivo de backup
backup_file_path = os.path.join(backup_dest_path, backup_file_name)

# Crear el archivo de backup (tarball) y comprimirlo
with shutil.make_archive(backup_file_path, 'gztar', home_dir_path):
    print(f"Backup creado exitosamente en {backup_file_path}")

```

cron

```
0 2 * * * /usr/bin/python3 /path/to/your/backup_script.py
```

8. Listar usuarios remotos (LINUX-SSH-Paramiko)

```
import paramiko

def listar_usuarios_remotos(hostname, port, username, password):
    # Inicializar el cliente SSH
    client = paramiko.SSHClient()

    # Agregar automáticamente la clave del host al sistema si falta
    client.set_missing_host_key_policy(paramiko.AutoAddPolicy())

    try:
        # Conectar al servidor remoto
        client.connect(hostname, port, username, password)

        # Ejecutar el comando para obtener la lista de usuarios
        stdin, stdout, stderr = client.exec_command('getent passwd')

        # Leer la salida del comando
        usuarios = stdout.read().decode().splitlines()

        # Cerrar la conexión
        client.close()

        # Procesar y mostrar los nombres de usuario
        for usuario in usuarios:
            nombre_usuario = usuario.split(':')[0]
            print(nombre_usuario)
    except Exception as e:
        print(f"Ocurrió un error: {e}")
        client.close()

# Parámetros de conexión
hostname = 'dirección_del_servidor_remoto'
port = 22 # Puerto SSH estándar
username = 'tu_usuario'
password = 'tu_contraseña'

# Llamada a la función
```

```
listar_usuarios_remotos(hostname, port, username, password)
```

9. Listar configuración de red de servidores remotos (LINUX-SSH-Paramiko)

```
import paramiko

# Lista de servidores a comparar: (host, puerto, usuario, contraseña)
servidores = [
    ('servidor1.dominio.com', 22, 'usuario1', 'contraseña1'),
    ('servidor2.dominio.com', 22, 'usuario2', 'contraseña2'),
    # Añadir más servidores según sea necesario
]

def obtener_configuracion_red(servidor):
    cliente_ssh = paramiko.SSHClient()
    cliente_ssh.set_missing_host_key_policy(paramiko.AutoAddPolicy())
    try:
        cliente_ssh.connect(servidor[0], servidor[1], servidor[2],
servidor[3])
        stdin, stdout, stderr = cliente_ssh.exec_command('ip addr')
        configuracion = stdout.read().decode('utf-8')
        return configuracion
    except Exception as e:
        print(f"Error al conectarse o ejecutar comando en {servidor[0]}: {e}")
    finally:
        cliente_ssh.close()

configuraciones = {}

# Recoger configuraciones
for servidor in servidores:
    configuraciones[servidor[0]] = obtener_configuracion_red(servidor)

# Comparar configuraciones (Este es un punto de partida simple)
# Aquí, simplemente imprimimos las configuraciones. Puedes extender esto
# para realizar comparaciones reales.
for servidor, config in configuraciones.items():
    print(f"Configuración de {servidor}:\n{config}\n\n")
```

```
# Aquí podrías añadir tu lógica para comparar las configuraciones de red  
de manera más detallada
```

10. Listar servicios activos en servidores remotos (LINUX-SSH-Paramiko)

```
import paramiko

# Lista de servidores con su información de acceso
servidores = [
    ("ip_servidor_1", "usuario_1", "contraseña_1"),
    ("ip_servidor_2", "usuario_2", "contraseña_2"),
    # Agrega más servidores según sea necesario
]

def listar_servicios_activos(hostname, username, password):
    # Inicializar el cliente SSH
    cliente = paramiko.SSHClient()
    cliente.set_missing_host_key_policy(paramiko.AutoAddPolicy())

    try:
        # Conectarse al servidor
        cliente.connect(hostname, username=username, password=password)

        # Ejecutar el comando para listar servicios activos
        stdin, stdout, stderr = cliente.exec_command('systemctl list-units  
--type=service --state=running')

        # Mostrar el nombre del servidor
        print(f"Servicios activos en {hostname}:")
        # Procesar y mostrar los servicios activos
        for linea in stdout:
            print(linea.strip())
        print("\n") # Espacio entre listados de servidores
    except Exception as e:
        print(f"Error al conectar o ejecutar en {hostname}: {e}")
    finally:
        cliente.close()

    for servidor in servidores:
        listar_servicios_activos(servidor[0], servidor[1], servidor[2])
```