

Exercicis Switching amb VyOS

VyOS (<https://vyos.io>) és un NOS lliure basat en Debian que proporciona mitjançant línia de comandes funcionalitats avançades de "switching" tant a màquines físiques x86_64 com a màquines VirtualBox, KVM i altres (tot i que sobre tot s'utilitza, com veurem més endavant, per les seves capacitats de "routing" -gràcies a incorporar el software FRR (<https://frrouting.org>)-, a més d'altres funcionalitats de xarxa, com "firewalling" -via Nftables- , VPN -via Wireguard- i altres serveis com SSH, DHCP, etc).

1.-a) Vés a <https://vyos.net/get/nightly-builds> i descarrega't d'allà la ISO de VyOS més moderna que hi hagi disponible (són uns 500MB). Crea una màquina virtual nova que tingui 400MB de RAM i 2G de disc dur (respecte la quantitat i modes de les eventuais tarjes de xarxa que hagi de tenir, ara mateix aquesta no és una dada rellevant). Arrenca-la amb la ISO de VyOS.

b) Arrenca la màquina i inicia sessió Live amb l'usuari "vyos" i contrasenya "vyos"; entraràs en un terminal Bash (bé, en realitat un "fork" propi de VyOS anomenat "vBash") que representa, en terminologia Cisco el mode "privilegiat" (o en terminologia Juniper, mode "operacional") del nostre router. Aquest mode (reconeixible pel símbol "\$" al final del prompt) ens permet veure l'estat del sistema i dels seus serveis (a més d'operar d'una forma bàsica amb ell) gràcies a les comandes que ofereix, les quals moltes són pròpies d'un sistema GNU estàndard però d'altres són scripts Python que pretenen simular el comportament de comandes pròpies d'un router Juniper. Concretament, prova les següents comandes (algunes estàndard de GNU -pintades de verd- o altres pròpies de VyOS) i digues què fan:

```
pwd
nano a.txt
systemctl list-units -t service
systemctl status vyos-router.service
show version
show login                o id
show users                o who
show history              o history
show host os              o uname -a
show host date            o show date o date o timedatectl
show host name            o hostnamectl --static
show interfaces           o ip a (fixar-se en els valors de les columnes "State" i "Link" "-S/L"-)
show interfaces { detail | counters } o ip -s a
show interfaces ethernet eth0 [{brief | physical | identify | statistics }] o ethtool [{-i | -s}] eth0
    NOTA: N'hi ha moltes més, d'interfícies a poder tractar: show interfaces wireless, show interfaces wireguard, etc
show hardware { cpu | usb | pci | mem } o lspci o lsusb o lspci o cat /proc/meminfo
show system uptime       o uptime
show system memory       o free -h
show system processes    o ps -ef
show system storage      o df -hT
show disk sda format     o fdisk -l /dev/sda
show system kernel-messages o sudo dmesg | less
show log [tail]          o less /var/log/messages o tail /var/log/messages
    NOTA: Els missatges es poden filtrar: show log { authorization | dhcp | dns | firewall | https | lldp | nat | vpn }
    Teniu més informació a https://docs.vyos.io/en/latest/configuration/system/syslog.html
show system login users [{ all | locked | nomUsuari }]
show system connections [{ tcp | udp }] o ss -a | less o ss -t | less o ss -ul | less
show conntrack { table ipv4 | statistics }
set date 02010000
reboot [now | at hh:mm | in n°min | cancel ] o poweroff [now | at hh:mm | in n°min | cancel ]
```

NOTA: Pulsant "?" -tecla "_" en el teclat anglès- es pot veure una llista de totes aquestes comandes; aquesta tecla també serveix, pulsada darrera una comanda ja escrita, per mostrar la llista d'opcions que aquesta ofereix. La referència completa d'aquestes comandes es pot trobar a <https://docs.vyos.io/en/latest/cli.html> (i la seva explicació del seu funcionament)

NOTA: Pulsant el tabulador mentre s'està escrivint una comanda qualsevol, aquesta s'autocompletarà

NOTA: En el cas d'obtenir una sortida de comanda molt llarga, aquesta s'autopagina automàticament. Es pot navegar una pàgina avall amb la tecla "SPACE", una pàgina amunt amb la tecla "b", una línia avall o amunt amb els cursors pertinents i sortir-ne i tornar al shell amb la tecla "q".

NOTA: L'idioma del teclat es pot canviar, però ho farem després d'instal·lar el sistema VyOS al disc dur (al proper apartat)

A diferència de les distribucions Linux de propòsit general, VyOS utilitza un tipus d'instal·lació basat en "imatges", la qual imita el procediment efectuat en switchos i routers tradicionals. Això permet, a més, mantenir diverses versions de VyOS a la mateixa màquina per poder canviar a una versió anterior si alguna cosa es trenqués després de l'actualització. "Basat en imatge" significa que tot el NOS, més enllà del kernel, està contingut en un fitxer de només lectura de tipus "squashfs", el qual es munta sota un sistema de fitxers "overlays" que allotja un directori contenint les dades mutables (on es guardaran les configuracions).

La instal·lació basada en imatges de VyOS s'implementa creant un directori per a cada imatge del dispositiu d'emmagatzematge seleccionat durant el procés d'instal·lació. Cada directori d'imatges conté el nucli del sistema, una imatge comprimida del sistema de fitxers arrel del NOS i un directori per a emmagatzematge persistent, tal com acabem de dir. En arrencar, el sistema extraurà la imatge del NOS a la memòria i muntarà els subdirectoris live-rw adequats per proporcionar la configuració del sistema persistent. Aquest procés permet que un sistema arrenqui sempre a un estat de treball conegut, ja que la imatge del sistema operatiu és fixa. També permet, com hem comentat, instal·lar diverses versions de VyOS al mateix dispositiu d'emmagatzematge.

- * La imatge es pot seleccionar manualment en arrencar mitjançant el menú del Grub però la imatge d'arrencada per defecte es pot configurar directament amb l'ordre `set default image=boot <nom>` (en mode operatiu).
- * Es pot mostrar una llista d'imatges disponibles amb l'ordre `show system image` (en mode operatiu).
- * Les imatges que ja no són necessàries es poden eliminar amb l'ordre `delete system image <nom>`.
- * També es poden afegir noves imatges del sistema mitjançant l'ordre `add system image {isopath | isourl}`; aquesta comanda extreurà la imatge de l'arxiu ISO indicat (ja estigui ubicat al sistema de fitxers local o bé de forma remota si es proporciona un URL) i afegirà l'entrada adient al menú d'arranc (utilitzant la configuració del sistema actual)

c) En el mode operacional del sistema VyOS, executa la comanda **install image** i respon les preguntes interactives que t'apareixeran (a la majoria d'elles la resposta per defecte ja serà la bona) per tal d'instal·lar el sistema al disc dur de la màquina virtual. Un cop finalitzada la instal·lació, apaga la màquina, treu la iso de la lectora virtual i torna a iniciar la màquina: hauràs de veure un menú del Grub on la seva entrada per defecte t'haurà de portar a un login en mode text on hauràs d'introduir el nom d'usuari i contrasenya que hagi indicat al procés d'instal·lació. Un cop fet això, entraràs en un terminal vBash, dins del mode operacional de VyOS.

NOTA: Ara ja podràs executar comandes com ara `lsblk`, `findmnt`, `df`, `du`, `fdisk`, `mkfs`, `mount` ...

Per tal de modificar la gran majoria d'aspectes relacionats amb el funcionament del sistema VyOS (com per exemple configurar la IP, màscara, porta d'enllaç, servidors DNS, etc de les seves tarjes de xarxa o establir les seves diferents capacitats de "switching", "routing", "firewalling", VPNs, servei SSH, servei DHCP, etc), el que primer cal fer és entrar en el seu mode de "configuració" (el qual és reconeixible pel símbol "#" al final del prompt) amb la comanda `configure`. Un cop dins d'aquest mode es poden realitzar un seguit de canvis en el sistema que anirem veient als següents exercicis però, siguin quins siguin aquests canvis, caldrà finalitzar-los amb alguna de les següents comandes:

commit : Fa que s'apliquin immediatament les comandes introduïdes fins llavors (o des del darrer `commit`). És a dir, cal tenir molt en compte que fins que no executem `commit` totes les comandes que haguem escrit prèviament no "funcionaran". Això serveix per realitzar els canvis de configuració de forma transaccional

discard: Fa que les comandes introduïdes fins llavors des del darrer `commit` s'eliminïn de la memòria, tal com si mai s'haguessin escrit.

save : Fa que els canvis aplicats en el darrer *commit* es guardin de forma que estiguin disponibles al proper reinici del sistema (és a dir, s'escriuen a l'arxiu "config.boot", que és el fitxer que inclou la configuració que és llegida -i aplicada- en arrencar el sistema VyOS). També es pot escriure *save scp://usuari:1234@x.x.x.x/fitxer.cfg* si es vol guardar la configuració a un fitxer ubicat en un servidor SCP remot. Igualment es poden utilitzar els protocols ftp o tftp.

exit : Surt del mode de configuració i torna al mode operacional (sempre que no hi hagi canvis pendents per fer-ne "commit"; en aquest cas no deixarà sortir-ne a no ser que s'escrigui explícitament *exit discard*). Executat al mode operacional, la comanda *exit* fa sortir de la sessió d'usuari.

NOTA: Si s'està al mode de configuració i es vol executar alguna comanda del mode operacional (ja que són diferents), en lloc d'haver de "tornar" al mode operacional amb *exit* i després haver de tornar a entrar al mode de configuració amb *configure* de nou per tal de continuar treballant-hi en aquest mode, es pot executar dins del mode de configuració directament la comanda pertanyent al mode operacional precedida de la comanda **run**, així per exemple: *run show version*

compare: Serveix per saber la diferència entre la configuració guardada en el darrer *commit* i els canvis posteriors que encara no s'han guardat en cap nou *commit* (identificats amb un "+", "-" o ">"). També és interessant la comanda *compare saved* , que serveix per saber la diferència en aquest cas entre la configuració guardada en el darrer *commit* i l'aplicada prèviament en el darrer inici del sistema

2.-a) Dins del terminal vBash en el seu mode operacional, executa la comanda *configure*. A partir d'aquí, executa les següents comandes, una darrera l'altra, i raona per a què serveixen (al final de tot pots fer *discard* per no aplicar-ne cap):

```
set system host-name pepe ---> Pots comprovar el nou valor (no definitiu però!) ambt show system host-name
NOTA: En general, tots els canvis fets amb set system ... es poden comprovar (abans fins i tot d'haver realitzat cap commit!!) amb show system...De fet, la comanda show system tal qual mostra tota la configuració actual completa
set system domain-name midomini.org
set system name-server 8.8.8.8
set system static-host-mapping host-name nomMaquina inet 10.1.2.3
set system option keyboard-layout es ---> D'aquesta opció sí valdria la pena fer-ne un "commit"!
set system time-zone Europe/Madrid
set system login banner { pre-login | post-login } HOLA
set system login user vyos authentication plaintext-password 1234 ---> Què mostra show system login ?
set system syslog { file /ruta/fitxer.txt | host x.x.x.x | global ... }
set system option performance { throughput | latency }
```

NOTA: Per tal d'esborrar qualsevol configuració establerta amb *set ...* es pot executar la mateixa comanda però començant per **delete ...** (i tot seguit fer *commit*, és clar).

b) Prova les següents comandes (en el mode operacional) i digues què fan:

```
show file running://config o ls /config
show file running://config/config.boot o cat /config/config.boot
```

NOTA: La comanda *show file* admet, a més de rutes locals, URLs de tipus HTTP o SCP o FTP o TFTP per tal d'indicar l'ubicació del fitxer a obrir. D'altra banda, la paraula "running" equival al nom de la imatge actualment executant-se, però si n'hi hagués més instal·lades, es pot inspeccionar qualsevol d'elles indicant el seu nom explícitament, així per exemple: *show file 1.5-rolling-202410231200://config*

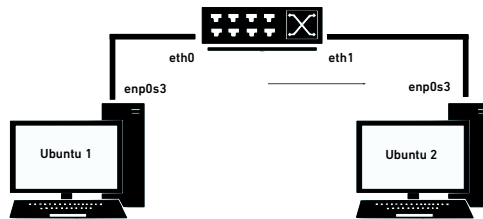
c) Fes un *commit* de qualsevol de les opcions de configuració llistades al primer apartat d'aquest exercici (per exemple, la corresponent al nom del sistema) i tot seguit observa el contingut del fitxer "config.boot". ¿Quina diferència veus en el valor de l'opció modificada existent en aquest fitxer respecte el que mostra la comanda **show configuration**? ¿Per què? ¿Aquesta diferència continua existint després d'executar la comanda *save*? ¿Per què?

NOTA: La sintaxis de la sortida és similar a la dels sistemes JunOS de Juniper

Tot i que el sistema VyOS se sol utilitzar més aviat per realitzar configuracions de L3, també és perfectament vàlid utilitzar-lo per realitzar operacions que tenen a veure estrictament amb tasques de "switching", com ara és la implementació de "bridgings", de "trunkings" o de "bondings", entre altres elements de L2. Al següent exercici ho veurem.

3.- PREVI-I) Descarrega't de l'apartat d'isos de la web del centre on estan els apunts de l'assignatura un arxiu VDI anomenat "Ubuntu CLI.vdi"; aquest arxiu conté la instal·lació (en un disc de 5GB) d'un sistema Ubuntu Server 22.04, amb l'usuari "usuari" i contrasenya "usuari". Crea una màquina virtual amb aquest disc que només tingui 700MB de RAM i tot seguit, clona-la (de forma completa i canviant la MAC de la seva tarja per a què no sigui igual a la de la 1ª!!) per a què en tinguis una altra màquina virtual exactament igual.

PREVI-II) Executa el següent script a la màquina amfitriona (les màquines virtuals hauran d'estar apagades). El que fa aquest script és "realitzar el cablatge" entre la màquina VyOS i les dues màquines Ubuntu de tal manera que l'esquema de connexions entre elles quedi automàticament així:



```
#!/bin/bash
VBoxManage modifyvm "Ubuntu1" --nic1 generic --nic-generic-driv1 UDPTunnel
VBoxManage modifyvm "Ubuntu1" --nic-property1 dest=127.0.0.1 --nic-property1 sport=10001 --nic-property1 dport=10003
VBoxManage modifyvm "Ubuntu2" --nic1 generic --nic-generic-driv1 UDPTunnel
VBoxManage modifyvm "Ubuntu2" --nic-property1 dest=127.0.0.1 --nic-property1 sport=10002 --nic-property1 dport=10004
VBoxManage modifyvm "VyOS" --nic1 generic --nic-generic-driv1 UDPTunnel
VBoxManage modifyvm "VyOS" --nic-property1 dest=127.0.0.1 --nic-property1 sport=10003 --nic-property1 dport=10001
VBoxManage modifyvm "VyOS" --nic2 generic --nic-generic-driv2 UDPTunnel
VBoxManage modifyvm "VyOS" --nic-property2 dest=127.0.0.1 --nic-property2 sport=10004 --nic-property2 dport=10002
```

NOTA MOLT IMPORTANT: L'script anterior està suposant que les màquines virtuals s'anomenen "VyOS", "Ubuntu1" i "Ubuntu2"; si no és el cas, modifica adientment l'script (o canvia el nom de les teves màquines virtuals, com vulguis)

L'script anterior només realitza el que és el cablejat, res més: la configuració de xarxa de les màquines Ubuntu (IP, màscara, etc) caldrà fer-ho (als propers apartats ja s'indicarà com) i, sobre tot, caldrà configurar el sistema VyOS per a què funcioni com a "switch" ja que per defecte no ho fa (és a dir, caldrà crear-hi un "bridge"; als propers apartats s'indicarà també com fer-ho).

a) Començarem per crear un "bridge" que inclogui la tarja *eth0* i *eth1* del sistema VyOS (això vol dir que el "switch" estarà format per aquests dos ports). Per aconseguir-ho, arrenca aquest sistema i executa-hi (dins del seu mode operacional) les següents comandes:

```
configure
set interfaces bridge br0 [description "El meu switch"]
set interfaces bridge br0 member interface eth0
set interfaces bridge br0 member interface eth1
commit
```

NOTA: Altres comandes interessants són:

*Per assignar una IP al "bridge" (i així crear una SVI): `set interfaces bridge br0 address {x.x.x.x/mask | dhcp }`

*Per establir el temps de validesa de les entrades a la taula MAC: `set interfaces bridge br0 aging n°segons`

*Per activar el protocol STP (per defecte ve desactivat): `set interfaces bridge br0 stp`

Per saber més, consulteu <https://docs.vyos.io/en/latest/configuration/interfaces/bridge.html>

i també <https://docs.vyos.io/en/latest/configuration/interfaces/ethernet.html> (on es mostra com canviar la MAC, la MTU,...)

a1) Comprova que, efectivament, has creat correctament el "bridge" observant la sortida (al mode operacional) de la comanda **show bridge**. També pots provar la comanda `show bridge br0 fdb` o `show bridge br0 detail`. Prova també les comandes `show configuration` i `show configuration commands`. ¿Què veus?

aIII) Arrenca ara els dos sistemes Ubuntu. Assigna-li a un la IP 10.0.0.1/8 (per fer-ho ràpid, fes-ho directament amb la comanda `sudo ip address add 10.0.0.1/8 dev enp0s3`) i a l'altre la IP 10.0.0.2/8 de la mateixa manera. Fes-hi un ping d'una màquina Ubuntu qualsevol a l'altra. ¿Què passa? ¿Per què?

aIIIBis) Canvia ara l'adreça IP d'una de les dues màquines Ubuntu per a què sigui ara la 9.9.9.9/24 (això ho pots fer executant, si per exemple estàs a "Ubuntu1", així: `sudo ip address delete 10.0.0.1/8 dev enp0s3 && sudo ip address add 9.9.9.9/24 dev enp0s3`). Torna llavors a fer-hi un ping des d'una màquina Ubuntu a l'altra. ¿Què passa ara? ¿Per què?

aIV) Elimina el "bridge" amb les comandes: `configure; delete interfaces bridge br0; commit` per tal de tornar a l'estat per defecte del sistema VyOS i apaga les màquines Ubuntu.

b) Ara crearem un "bridge" que tingui definides les VLAN n°6 i n°8 Més concretament, haurà de tenir etiquetada la seva tarja `eth0` en la VLAN n°6 (en mode "access") i la seva tarja `eth1` en la VLAN n°8 (en mode "access" també). Per aconseguir això executa les següents comandes (en negreta estan els canvis respecte l'apartat anterior):

```
configure
set interfaces bridge br0 [description "El meu switch amb VLANs"] enable-vlan
set interfaces bridge br0 member interface eth0 native-vlan 6
set interfaces bridge br0 member interface eth1 native-vlan 8
commit
```

bII) Comprova que, efectivament, has creat correctament el "bridge" observant la sortida de la comanda `bridge vlan show` (tant se val en quin mode s'executi, és una comanda nativa de Linux). Prova també les comandes `run show configuration` i `run show configuration commands`. ¿Què veus?

NOTA: La marca "PVID" indica que quan entri un paquet en el port en qüestió, serà etiquetat amb la VLAN corresponent i la marca "Egress untagged" indica que quan surti un paquet pel port en qüestió, serà desetiquetat. És a dir, el port funcionarà en mode "access". Si no estiguessin present cap d'aquestes marques, el port seria un port en mode "trunk" ja que ni etiquetaria ni desetiquetaria (ho veurem properament). La VLAN n°1 és a la que per defecte pertanyen tots els ports

bIII) Arrenca ara els dos sistemes Ubuntu. Assigna-li a un la IP 10.0.0.1/8 (per fer-ho ràpid, fes-ho directament amb la comanda `sudo ip address add 10.0.0.1/8 dev enp0s3`) i a l'altre la IP 10.0.0.2/8 de la mateixa manera. Fes-hi un ping d'una màquina Ubuntu qualsevol a l'altra. ¿Què passa? ¿Per què?

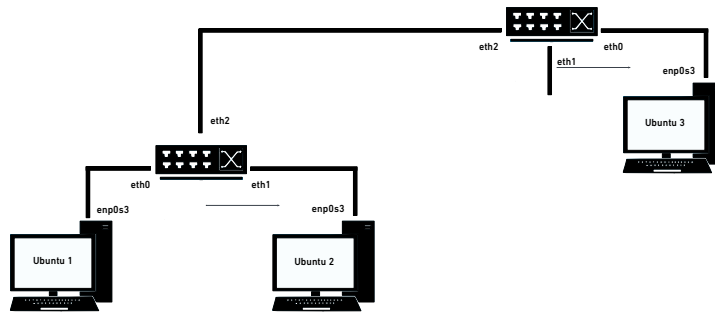
bIV) Elimina el "bridge" amb les comandes: `configure; delete interfaces bridge br0; commit` per tal de tornar a l'estat per defecte del sistema VyOS i apaga totes les màquines.

PREVI) Clona la màquina VyOS de forma completa i canviant la MAC de les tarjes de xarxa al quadre de Paràmetres del VirtualBox); ara anomenarem "VyOS1" a la que teníem fins ara i "VyOS2" a la nova.

PREVI-II) Arrenca la màquina "VyOS2" un moment ja que haurem de realitzar un canvi per tal que funcioni correctament (per ser un clon i no una instal·lació des de zero). Concretament, un cop hagis iniciat sessió, executa la comanda `nano /config/config.boot` i esborra-hi totes les línies `hw-id "xx:xx:xx:xx:xx:xx"` (on "xx:xx:xx:xx:xx:xx" representa una MAC) que hi apareguin en aquest fitxer. Un cop fet això, guarda i apaga la màquina.

PREVI-III) Clona també una tercera màquina Ubuntu (també de forma completa i amb la MAC canviada), que anomenarem "Ubuntu3".

PREVI-IV) Executa el següent script a la màquina amfitriona (les màquines virtuals hauran d'estar apagades). El que fa aquest script és "realitzar el cablatge" entre les dues màquines VyOS i les tres màquines Ubuntu de tal manera que l'esquema de connexions entre elles quedi automàticament així:



```
#!/bin/bash
VBoxManage modifyvmm "Ubuntu1" --nic1 generic --nic-generic-driv1 UDPTunnel
VBoxManage modifyvmm "Ubuntu1" --nic-property1 dest=127.0.0.1 --nic-property1 sport=10001 --nic-property1 dport=10003
VBoxManage modifyvmm "VyOS1" --nic1 generic --nic-generic-driv1 UDPTunnel
VBoxManage modifyvmm "VyOS1" --nic-property1 dest=127.0.0.1 --nic-property1 sport=10003 --nic-property1 dport=10001
VBoxManage modifyvmm "Ubuntu2" --nic1 generic --nic-generic-driv1 UDPTunnel
VBoxManage modifyvmm "Ubuntu2" --nic-property1 dest=127.0.0.1 --nic-property1 sport=10002 --nic-property1 dport=10004
VBoxManage modifyvmm "VyOS1" --nic2 generic --nic-generic-driv2 UDPTunnel
VBoxManage modifyvmm "VyOS1" --nic-property2 dest=127.0.0.1 --nic-property2 sport=10004 --nic-property2 dport=10002
VBoxManage modifyvmm "Ubuntu3" --nic1 generic --nic-generic-driv1 UDPTunnel
VBoxManage modifyvmm "Ubuntu3" --nic-property1 dest=127.0.0.1 --nic-property1 sport=10005 --nic-property1 dport=10007
VBoxManage modifyvmm "VyOS2" --nic1 generic --nic-generic-driv1 UDPTunnel
VBoxManage modifyvmm "VyOS2" --nic-property1 dest=127.0.0.1 --nic-property1 sport=10007 --nic-property1 dport=10005
VBoxManage modifyvmm "VyOS2" --nic2 null
VBoxManage modifyvmm "VyOS1" --nic3 generic --nic-generic-driv3 UDPTunnel
VBoxManage modifyvmm "VyOS1" --nic-property3 dest=127.0.0.1 --nic-property3 sport=10006 --nic-property3 dport=10008
VBoxManage modifyvmm "VyOS2" --nic3 generic --nic-generic-driv3 UDPTunnel
VBoxManage modifyvmm "VyOS2" --nic-property3 dest=127.0.0.1 --nic-property3 sport=10008 --nic-property3 dport=10006
```

Com es pot veure, el nou switch només té connectat el nou PC al port "eth0" però el seu "eth1" romandrà lliure. D'altra banda, els ports "eth2" d'ambdós switchos és el que s'utilitza per connectar-los entre sí

NOTA: Tal com ja s'ha comentat, l'script anterior només realitza el que és el cablejat, res més: la configuració de xarxa de les màquines Ubuntu (IP, màscara, etc) caldrà fer-la i, sobre tot, caldrà configurar el sistema VyOS per a què funcioni com a "switch" adientment

c) Arrenca la màquina "VyOS1"; allà hi configurarem un "bridge" similar al de l'apartat anterior (és a dir, amb la seva tarja *eth0* etiquetada en la VLAN n°6 en mode "access" i la seva tarja *eth1* etiquetada en la VLAN n°8 en mode "access" també) però que, a més, disposarà d'un tercer port (en aquest cas la tarja *eth2*) que serà de tipus "trunk" per tal de permetre la sortida/entrada de tant el trànsit pertanyent a la VLAN n°6 com al de la VLAN n°8 (sense cap procés d'etiquetatge/desetiquetatge). Per aconseguir això executa les següents comandes (en negreta estan els canvis respecte l'apartat anterior):

```
configure
set interfaces bridge br0 [description "El meu switch amb VLANs i enllaços trunks"] enable-vlan
set interfaces bridge br0 member interface eth0 native-vlan 6
set interfaces bridge br0 member interface eth1 native-vlan 8
set interfaces bridge br0 member interface eth2 allowed-vlan 6-8
commit
```

NOTA: El valor del paràmetre "allowed-vlan" pot indicar un rang (com apareix a l'exemple anterior) o bé un número de VLAN individual. En el primer cas, estaran permeses totes les VLANs incloses en el rang (a l'exemple anterior, serien les VLANs n°6, n°8 però també la n°7). No és possible indicar una llista de valors numèrics discrets.

cII) Arrenca la màquina "VyOS2"; allà hi configurarem un "bridge" de forma idèntica a l'anterior: la seva tarja *eth0* haurà d'estar etiquetada en la VLAN n°6 en mode "access", la seva tarja *eth1* estarà etiquetada en la VLAN n°8 en mode "access" també i la seva tarja *eth2* serà de tipus "trunk" per tal de permetre la sortida/entrada de tant el trànsit pertanyent a la VLAN n°6 com al de la VLAN n°8. Per tant, hauràs de repetir les mateixes comandes de l'apartat anterior al nou switch.

cIII) Comprova que, efectivament, has creat correctament el "bridge" amb els ports de tipus "access" i "trunk" adients observant la sortida (tant se val en quin mode) de la comanda *bridge vlan show* en tots dos switchos. Prova també les comandes *show configuration* i *show configuration commands*. ¿Què veus?

cIV) Arrenca ara els tres sistemes Ubuntu. Assigna-li a "Ubuntu1" la IP 10.0.0.1/8 , a "Ubuntu2" la IP 10.0.0.2/8 i a "Ubuntu3" la IP 10.0.0.3/8. Fes-hi un ping des d'"Ubuntu1" a "Ubuntu2" (o viceversa) ¿Què passa? ¿Per què? Fes-hi un ping des d'"Ubuntu1" a "Ubuntu3" (o viceversa) ¿Què passa? ¿Per què? Fes-hi un ping des d'"Ubuntu2" a "Ubuntu3" (o viceversa) ¿Què passa? ¿Per què?

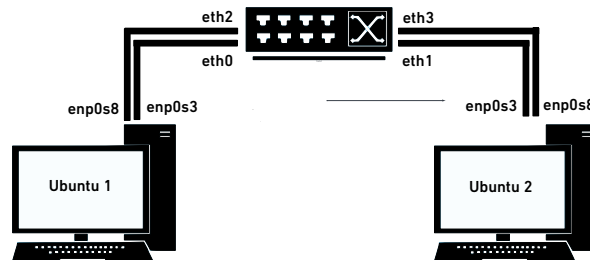
cV) Guarda la configuració actual de la màquina "VyOS2" (amb la comanda *save*) i apaga-la. Tot seguit executa el següent script en un terminal de la màquina amfitriona per tal de "desendollar" la màquina "Ubuntu3" del port "eth0" del switch "VyOS2" i "endollar-la" al seu port "eth1".

```
#!/bin/bash
VBoxManage modifyvmm "VyOS2" --nic1 null
VBoxManage modifyvmm "VyOS2" --nic2 generic --nic-generic-driv UDPTunnel
VBoxManage modifyvmm "VyOS2" --nic-property2 dest=127.0.0.1 --nic-property2 sport=10007 --nic-property2 dport=10005
```

Arrenca de nou la màquina "VyOS2" i tot seguit, fes-hi un ping de nou des d'"Ubuntu1" a "Ubuntu2" (o viceversa) ¿Què passa? ¿Per què? Fes-hi un ping des d'"Ubuntu1" a "Ubuntu3" (o viceversa) ¿Què passa? ¿Per què? Fes-hi un ping des d'"Ubuntu2" a "Ubuntu3" (o viceversa) ¿Què passa? ¿Per què?

cVI) Elimina el "bridge" de les dues màquines VyOS amb sengles comandes: *configure; delete interfaces bridge br0; commit; save* per tal de tornar a l'estat per defecte del seu sistema i apaga totes les màquines.

PREVI) Executa el següent script a la màquina amfitriona (les màquines virtuals hauran d'estar apagades). El que fa aquest script és "realitzar el cablatge" entre una màquina VyOS ("VyOS1") i dues màquines Ubuntu ("Ubuntu1" i "Ubuntu2") de tal manera que l'esquema de connexions entre elles quedi automàticament així:



```
#!/bin/bash
VBoxManage modifyvmm "Ubuntu1" --nic1 generic --nic-generic-driv1 UDPTunnel
VBoxManage modifyvmm "Ubuntu1" --nic-property1 dest=127.0.0.1 --nic-property1 sport=10001 --nic-property1 dport=10003
VBoxManage modifyvmm "Ubuntu1" --nic2 generic --nic-generic-driv2 UDPTunnel
VBoxManage modifyvmm "Ubuntu1" --nic-property2 dest=127.0.0.1 --nic-property2 sport=10005 --nic-property2 dport=10007
VBoxManage modifyvmm "Ubuntu2" --nic1 generic --nic-generic-driv1 UDPTunnel
VBoxManage modifyvmm "Ubuntu2" --nic-property1 dest=127.0.0.1 --nic-property1 sport=10002 --nic-property1 dport=10004
VBoxManage modifyvmm "Ubuntu2" --nic2 generic --nic-generic-driv2 UDPTunnel
VBoxManage modifyvmm "Ubuntu2" --nic-property2 dest=127.0.0.1 --nic-property2 sport=10006 --nic-property2 dport=10008
VBoxManage modifyvmm "VyOS1" --nic1 generic --nic-generic-driv1 UDPTunnel
VBoxManage modifyvmm "VyOS1" --nic-property1 dest=127.0.0.1 --nic-property1 sport=10003 --nic-property1 dport=10001
VBoxManage modifyvmm "VyOS1" --nic2 generic --nic-generic-driv2 UDPTunnel
VBoxManage modifyvmm "VyOS1" --nic-property2 dest=127.0.0.1 --nic-property2 sport=10004 --nic-property2 dport=10002
VBoxManage modifyvmm "VyOS1" --nic3 generic --nic-generic-driv3 UDPTunnel
VBoxManage modifyvmm "VyOS1" --nic-property3 dest=127.0.0.1 --nic-property3 sport=10007 --nic-property3 dport=10005
VBoxManage modifyvmm "VyOS1" --nic4 generic --nic-generic-driv4 UDPTunnel
VBoxManage modifyvmm "VyOS1" --nic-property4 dest=127.0.0.1 --nic-property4 sport=10008 --nic-property4 dport=10006
```

e) La idea és crear un "bridge" format per dos dispositius de tipus "bond", un compost per la tarja *eth0* i *eth2* i un altre per la tarja *eth1* i *eth3* del sistema VyOS. Ambdós dispositius "bond" implementaran l'algoritme Round-Robin (el per defecte en sistemes Linux, de fet). Per aconseguir-ho, arrenca aquest sistema i executa-hi (dins del seu mode operacional) les següents comandes:

```

configure
set interfaces bonding bond0 mode round-robin
set interfaces bonding bond0 member interface eth0
set interfaces bonding bond0 member interface eth2
commit
set interfaces bonding bond1 mode round-robin
set interfaces bonding bond1 member interface eth1
set interfaces bonding bond1 member interface eth3
commit
set interfaces bridge br0
set interfaces bridge br0 member interface bond0
set interfaces bridge br0 member interface bond1
commit

```

NOTA: En aquest exercici no ho veurem, però una interfície "bond" pot tenir una adreça IP assignada i, per tant, comportar-se com qualsevol altre dispositiu Ethernet. Per aconseguir això simplement hauríem d'executar, un cop creat el dispositiu "bond", la comanda següent: **set interfaces bonding bond0 address 192.168.10.10/24**

eII) Comprova que hakis creat correctament el "bonding" (amb les tarjes incloses desitjades) observant la sortida (al mode operacional) de les comandes `show interfaces` o `show interfaces bonding [detail]` i també `show interfaces bonding slaves`, així com la comanda `show interfaces` al mode de configuració

eIII) Arrenca ara els dos sistemes Ubuntu. En cadascun d'ells hauràs de definir també el seu respectiu dispositiu "bonding" d'acord amb els "bondings" existents al switch. Concretament, executa les següents comandes pròpies de Linux (com a "root", pots fer `sudo -i` per convertir-te'n) en cadascuna d'aquestes dues màquines...:

***A "Ubuntu1"**

```

ip link add elmeubond type bond
ip link set enp0s3 down && ip link set enp0s8 down
ip link set enp0s3 master elmeubond
ip link set enp0s8 master elmeubond
ip address add 10.0.0.1/8 dev elmeubond
ip link set elmeubond up
ip -c address show dev elmeubond

```


***A "Ubuntu2"**

```

ip link add elmeubond type bond
ip link set enp0s3 down && ip link set enp0s8 down
ip link set enp0s3 master elmeubond
ip link set enp0s8 master elmeubond
ip address add 10.0.0.2/8 dev elmeubond
ip link set elmeubond up
ip -c address show dev elmeubond

```

...i tot seguit fes-hi un ping d'una màquina Ubuntu qualsevol a l'altra. ¿Què passa? ¿Per què? (XXX)

eIV) Mentre s'està fent ping, "treu" un cable de xarxa d'alguna de les dues màquines Ubuntu deseleccionant-lo clicant amb el botó dret sobre la iconeta petita que apareix a la barra inferior de la finestra corresponent a la màquina (és a dir, fent clic amb el botó dret aquí: ). ¿Què passa? ¿Per què? Apaga finalment totes les màquines

La funcionalitat anomenada "**port mirroring**" (a vegades també anomenat SPAN) consisteix en copiar el trànsit entrant/sortint d'una (o més) interfícies determinades d'un switch a una (o més) interfícies concretes del mateix switch. A les interfícies que reben la còpia del trànsit és on llavors se podran connectar eines de monitoratge, com ara esnifadors o IDS.

NOTA: Si en comptes de copiar el tràfic a una altra interfície local del switch es copia enviant-lo a un port d'un altre switch remot estariem parlant llavors de "Remote SPAN" (RSPAN)

La funcionalitat anomenada "**port security**" permet limitar l'accés (és a dir, la connexió de dispositius finals) a determinades interfícies del switch. Pot tenir varis modes de funcionament: o bé pot limitar l'accés a adreces MAC específiques (per a què la interfície no reenvii el trànsit d'entrada des d'adreces d'origen no definides), o bé limitar-lo només a la primera adreça MAC que aprengui (es pot indicar un cert temps després del qual aquesta adreça ja no tindrà accés) o bé limitar-lo a un nombre específic d'adreces MAC. Igualment, es pot definir l'acció a prendre quan es produeixi una infracció (o bé eliminar els paquets o bé posar la interfície en estat "down"...) i afegir un temps concret durant el qual aquesta acció tindrà efecte.

NOTA: La funcionalitat de "port security" no està implementada a VyOS però és possible establir-la treballant directament sobre el sistema Linux subjacent, concretament fent ús del tallafocs Nftables, tal com s'explica aquí <https://serverfault.com/questions/1035496/limiting-number-of-mac-address-per-port-on-linux-based-platrform>. En aquest sentit, si es vol saber més sobre aquest tema, a https://wiki.nftables.org/wiki-nftables/index.php/Nftables_families#netdev està explicada la família "netdev" de taules Nftables. D'altra banda, també resulta molt interessant la família "bridge", precisament: https://wiki.nftables.org/wiki-nftables/index.php/Bridge_filtering

La funcionalitat "**port isolation**" permet bloquejar la connexió de les diferents interfícies que formen part del mateix switch entre sí (però no amb la resta d'interfícies on la "port isolation" no estigui definida). Aquesta funcionalitat tampoc està disponible a VyOS però es pot aconseguir amb les eines bàsiques de "bridging" que proporciona el propi Linux; concretament, si es vol que els ports *eth0* i *eth1* (que se suposa que pertanyen a un mateix "bridge") no es puguin veure entre sí (però sí amb qualsevol altre port), només caldrà executar les següents comandes: *bridge link set dev eth0 isolated on && bridge link set dev eth1 isolated on*

f) És perfectament possible implementar la funcionalitat de "port mirroring" en un sistema VyOS (o bé de tràfic d'entrada ("ingress") o bé de sortida ("egress") cap a (o des de, respectivament) totes les interfícies del "bridge" a una interfície determinada). Concretament, si executessis aquestes comandes, ¿què aconseguiries?

```
configure
set interfaces bridge br0
set interfaces bridge br0 member interface eth0
set interfaces bridge br0 member interface eth1
set interfaces bridge br0 mirror ingress eth2 (també es podria haver indicat "egress" en lloc de "ingress")
commit
```

NOTA: Es pot fer còpies de tràfic de port individual a port individual amb la comanda *set interfaces ethernet eth0 mirror ...*

NOTA: Un article on s'explica una manera de fer RSPAN fent servir només eines pròpies de Linux (tot i ser un procediment complex) és <http://arthurchiao.art/blog/traffic-mirror-with-tc-and-tunneling>

4.- ¿Què explica aquest vídeo: <https://www.youtube.com/watch?v=p39mFz7ORco> ?