NetGUI: Configuración de Switches, Cachés de ARP, IP Aliasing, VLANs

Sistemas Telemáticos para Medios Audiovisuales

Departamento de Teoría de la Señal y Comunicaciones y Sistemas Telemáticos y Computación

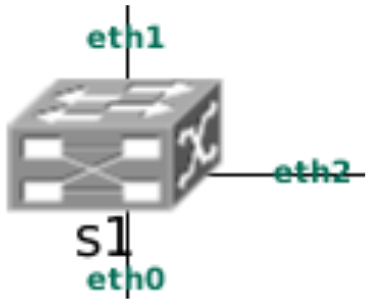
Septiembre de 2018

Contenidos

Contenidos

Bridges/Switches en NetGUI

- La interfaz de NetGUI permite dibujar *bridges/switches* los cuáles están representados a través del siguiente icono:



- Estos *bridges/switches* se configuran a través del comando `brctl` que pertenece al paquete `bridge-utils` en Linux.
- Por defecto el protocolo STP está desactivado en los *switches*.

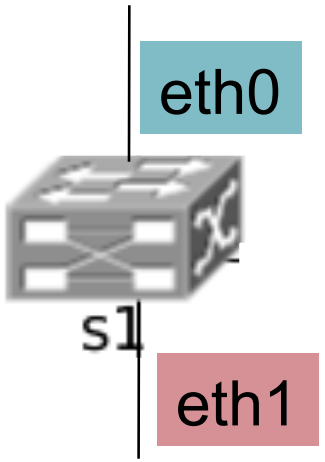
Consultar información sobre el *bridge* (I)

- **Mostrar la configuración del bridge**:

  `brctl show`

En s1:

```
s1:~# brctl show
bridge name          bridge id          STP enabled     interfaces
    s1           8000.3e323176a0de          no              eth0
                                                            eth1
```



Consultar información sobre el *bridge* (II)

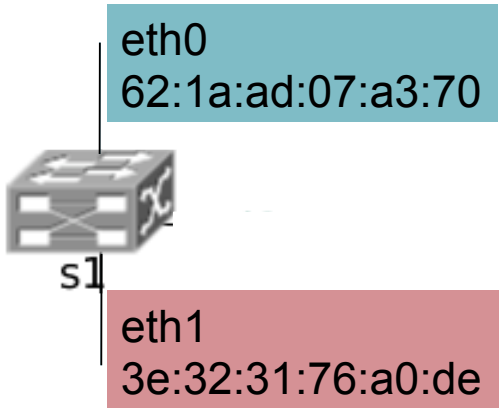
- **Mostrar la tabla de MACs aprendidas**:

```
brctl showmacs <nombreSwitch>
```

En s1:

```
s1:~# brctl showmacs s1
```

port no	mac addr	is local?	ageing timer
2	0a:29:6e:9a:3e:d4	no	11.77
2	3e:32:31:76:a0:de	yes	0.00
1	62:1a:ad:07:a3:70	yes	0.00
1	aa:da:5c:68:ed:27	no	11.77



- El `port no` enumera las interfaces empezando por 1 para `eth0`, 2 para `eth1` y así sucesivamente.
- Las líneas en amarillo contienen las MACs realmente aprendidas, las otras líneas muestran las interfaces locales del *bridge*.
- La columna `ageing timer` indica el tiempo (en segundos) que ha pasado desde que se aprendió o refrescó cada entrada. La entrada se elimina al llegar al valor máximo permitido (por defecto, 300 seg).

  NOTA: las direcciones de las interfaces locales no caducan nunca.

Borrar la tabla de MACs aprendidas por el *bridge*

- **Para eliminar las MACs aprendidas** por el *bridge* se deshabilita el funcionamiento del *bridge* con el comando:

  ```
  ifconfig <nombre_br> down
  ```

  En s1:

  ```
  s1:~# ifconfig s1 down
  ```

- Al habilitar de nuevo el *bridge*, éste no tendrá ninguna MAC aprendida, salvo las de sus interfaces locales:

  ```
  s1:~# ifconfig s1 up
  s1:~# brctl showmacs s1
  port no     mac addr       is local?  ageing timer
     2    3e:32:31:76:a0:de   yes         0.00
     1    62:1a:ad:07:a3:70   yes         0.00
  ```

- **Modificar el tiempo de caducidad de las entradas en la tabla de MACs aprendidas** (por defecto 300 seg):

  ```
  brctl setageing <nombre_br> <tiempoEnSeg
  ```

  En s1:

  ```
  s1:~# brctl setageing s1 1
  ```

Contenidos

Caché de ARP en pcs y routers

- Para consultar la caché de ARP en una máquina se utiliza el comando `arp`:

```
pc2:~# arp -a
  ? (10.0.0.1) at 0A:29:92:55:93:70 [ether] on eth0
  ? (10.0.0.2) at 0B:39:12:54:83:71 [ether] on eth0
```

- Para borrar una entrada de la caché de ARP se utiliza la opción `-d` del comando `arp`:

```
pc2:~# arp -d 10.0.0.2
```

Si se consulta la caché de ARP ahora:

```
pc2:~# arp -a
  ? (10.0.0.1) at 0A:29:92:55:93:70 [ether] on eth0
  ? (10.0.0.2) at <incomplete> on eth0
```

Contenidos

Configuración de un *router* para que haga Proxy ARP

- Proxy ARP se utiliza en un *router* para que responda a solicitudes de ARP que preguntan una dirección MAC que no se corresponde con ninguna de las interfaces de red del *router*.
- Configuración:
  1. Para activar Proxy ARP en la interfaz `eth0` de un *router*:

```
echo 1 > /proc/sys/net/ipv4/conf/eth0/proxy_arp
```

  2. Para que el *router* responda con su dirección MAC a una solicitud de ARP preguntando por una cierta dirección:

```
arp -i <interfaz_resp> -Ds <dirIP> <interfaz_MAC> netmask <máscara>
```

     - `<interfaz_resp>`: Interfaz del *router* que se utilizará para enviar la respuesta de ARP.
     - `<dirIP>`: Dirección IP de la solicitud de ARP para la que el *router* debe responder.
     - `<interfaz_MAC>`: La dirección MAC que irá en la respuesta de ARP será la dirección MAC de la interfaz `<interfaz_MAC>` del *router*
     - `<máscara>`: Máscara que permite hacer proxy ARP para un rango de direcciones IP definido por `<dirIP>` y `<máscara>`. Si la máscara es 255.255.255.255, se realizará el proxy ARP para una dirección IP concreta en vez de para un rango de direcciones IP.

  3. Además puede ser necesario introducir en la tabla de encaminamiento del *router* una entrada adecuada para que, una vez le lleguen los datagramas IP gracias al proxy ARP, los reenvíe a su destino por la interfaz correcta.

Ejemplo

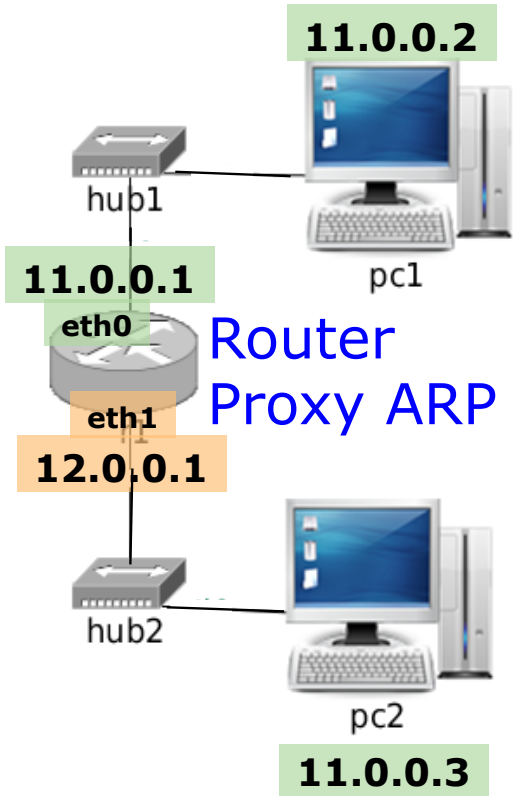
- Para que ser reciba tráfico en el sentido pc1→pc2:
  - El *router* debe responder con la MAC de `eth0` cuando pc1 mande una petición de ARP solicitando la dirección MAC de 11.0.0.3
  - En la tabla de encaminamiento del *router* hay que añadir una entrada específica para poder encaminar hasta 11.0.0.3 por la interfaz adecuada

```
r1:~# echo 1 > /proc/sys/net/ipv4/conf/eth0/proxy_arp
r1:~# arp -i eth0 -Ds 11.0.0.3 eth0 netmask 255.255.255.255
r1:~# route add -host 11.0.0.3 dev eth1
```

- Para que se reciba el tráfico en el sentido pc2→pc1 habría que realizar una configuración análoga a la anterior.

```
r1:~# echo 1 > /proc/sys/net/ipv4/conf/eth1/proxy_arp
r1:~# arp -i eth1 -Ds 11.0.0.2 eth1 netmask 255.255.255.255
```

(Nótese que en este caso no es necesario añadir ninguna ruta, ya que el *router* ya tendrá una ruta para la red 11.0.0.0/24 a través de `eth0`)



Contenidos

Configuración con la orden ip

- Ejemplo de asignación de las direcciones 11.0.0.1/24 y 12.0.0.1/24 a la interfaz eth0 de r1:

```
r1:~# ip link set eth0 up
r1:~# ip address add dev eth0 11.0.0.1/24 broadcast +
r1:~# ip address add dev eth0 12.0.0.1/24 broadcast +
r1:~# ip addr show eth0
3: eth0: <BROADCAST,MULTICAST,UP,10000> mtu 1500 qdisc pfifo_fast qlen 1000
  link/ether de:34:80:65:19:5a brd ff:ff:ff:ff:ff:ff
  inet 11.0.0.1/24 brd 11.0.0.255 scope global eth0
  inet 12.0.0.1/24 brd 12.0.0.255 scope global eth0
  inet6 fe80::dc34:80ff:fe65:195a/64 scope link
    valid_lft forever preferred_lft forever
```

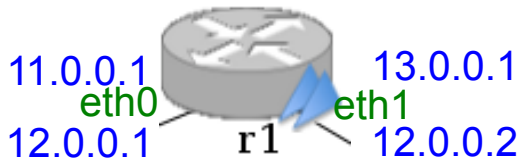
Configuración con la orden ifconfig

- Para utilizar IP aliasing con ifconfig es necesario referirse a las interfaces "virtuales" eth0:0, eth0:1... como aquellas que tendrán las direcciones IP adicionales a la primera que se asigne a eth0.
- Ejemplo de asignación de las direcciones 11.0.0.1/24 y 12.0.0.1/24 a la interfaz eth0 de r1:

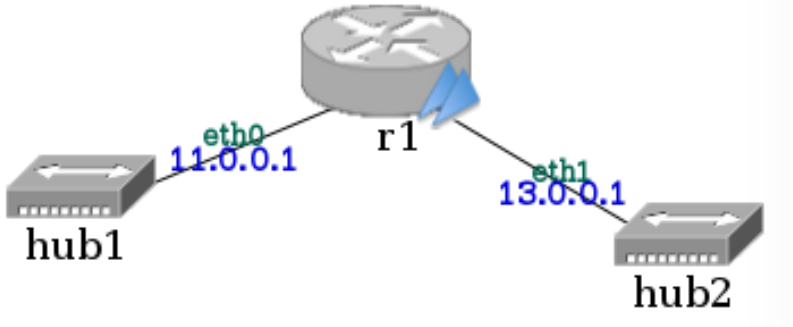
```
r1:~# ifconfig eth0 11.0.0.1 netmask 255.255.255.0
r1:~# ifconfig eth0:0 12.0.0.1 netmask 255.255.255.0
r1:~# ifconfig
amarillo
eth0      Link encap:Ethernet  HWaddr 26:33:2A:36:35:4A
          inet addr: 11.0.0.1  Bcast:11.0.0.255  Mask:255.255.255.0
          inet6 addr: fe80::2433:2aff:fe36:354a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:468 (468.0 b)
          Interrupt:5

eth0:0    Link encap:Ethernet  HWaddr 26:33:2A:36:35:4A
          inet addr: 12.0.0.1  Bcast:12.0.0.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
```

Ejemplo de configuración con IP aliasing (I)

- Se desean configurar las siguientes direcciones IP en `r1`:

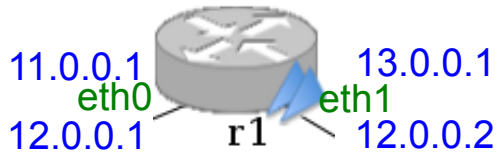


En NetGUI sólo se muestra la primera dirección IP configurada



Ejemplo de configuración con IP aliasing (II)

- Una vez añadidas las direcciones, si se consulta la tabla de encaminamiento en `r1`:




```
r1:~# route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
11.0.0.0        *               255.255.255.0   U     0      0        0 eth0
12.0.0.0        *               255.255.255.0   U     0      0        0 eth0
12.0.0.0        *               255.255.255.0   U     0      0        0 eth1
13.0.0.0        *               255.255.255.0   U     0      0        0 eth1
r1:~#
```

Ejemplo de configuración con IP aliasing (III)

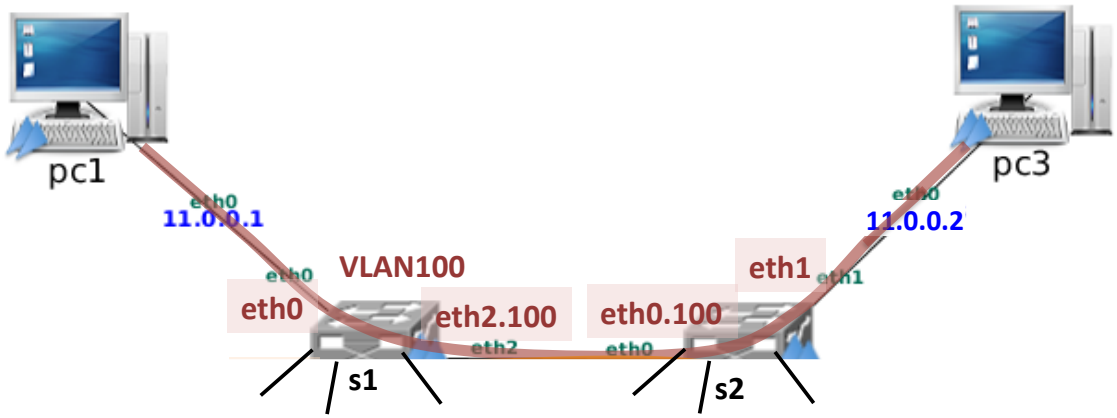
- Suponiendo que hay una sola máquina de la subred 12.0.0.0/24 conectada a la interfaz `eth0` de `r1`, p. ej. la máquina 12.0.0.100 y que en la interfaz `eth1` de `r1` hay varias máquinas de la subred 12.0.0.0/24, es conveniente dejar la tabla de encaminamiento de la siguiente forma:

```
r1:~# route del -net 12.0.0.0 netmask 255.255.255.0 dev eth0
r1:~# route add -host 12.0.0.100 dev eth0
r1:~# route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
12.0.0.100      *               255.255.255.255 UH    0      0        0 eth0
11.0.0.0        *               255.255.255.0   U     0      0        0 eth0
12.0.0.0        *               255.255.255.0   U     0      0        0 eth1
13.0.0.0        *               255.255.255.0   U     0      0        0 eth1
r1:~#
```

Contenidos

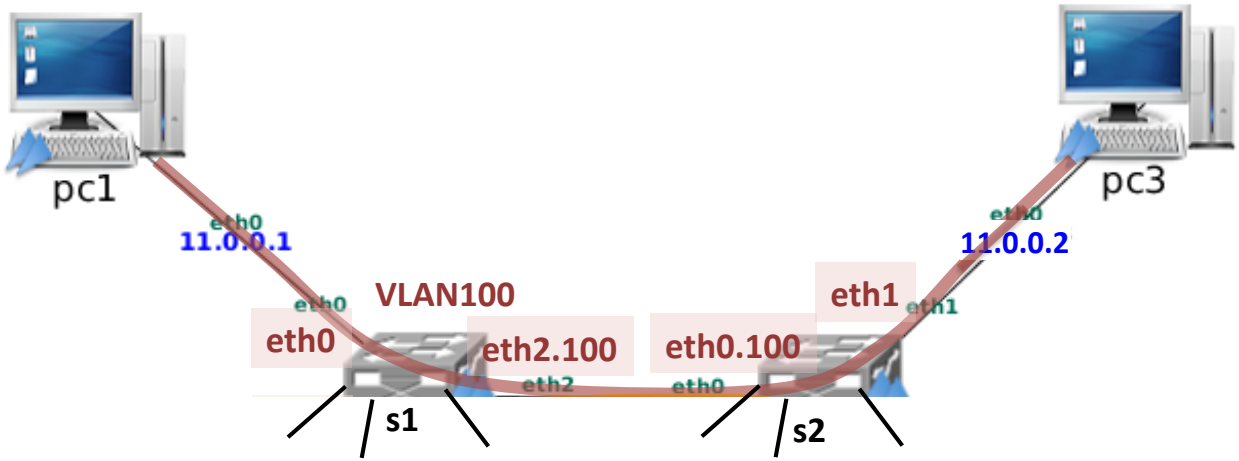
Identificar interfaces VLANs

- Para configurar una VLAN es necesario determinar qué interfaces físicas del *switch* van a pertenecer a esa VLAN y si esas interfaces son:
  - **Interfaces sin ID VLAN**: `eth0`, `eth1`, etc. A través de este tipo de interfaces se envían/reciben tramas sin la etiqueta VLAN. Normalmente a estas interfaces están conectados dispositivos finales.
  - **Interfaces con ID VLAN**: se definen con el nombre de la interfaz seguido del identificador de VLAN. Por ejemplo, para la VLAN 100: `eth2.100`, `eth0.100`, etc. Las tramas que se reciben/envían en estas interfaces llevan etiqueta VLAN y dicha etiqueta no se modifica. Estas interfaces normalmente conectan *switches* y a través de ellas viajan diferentes VLANs.



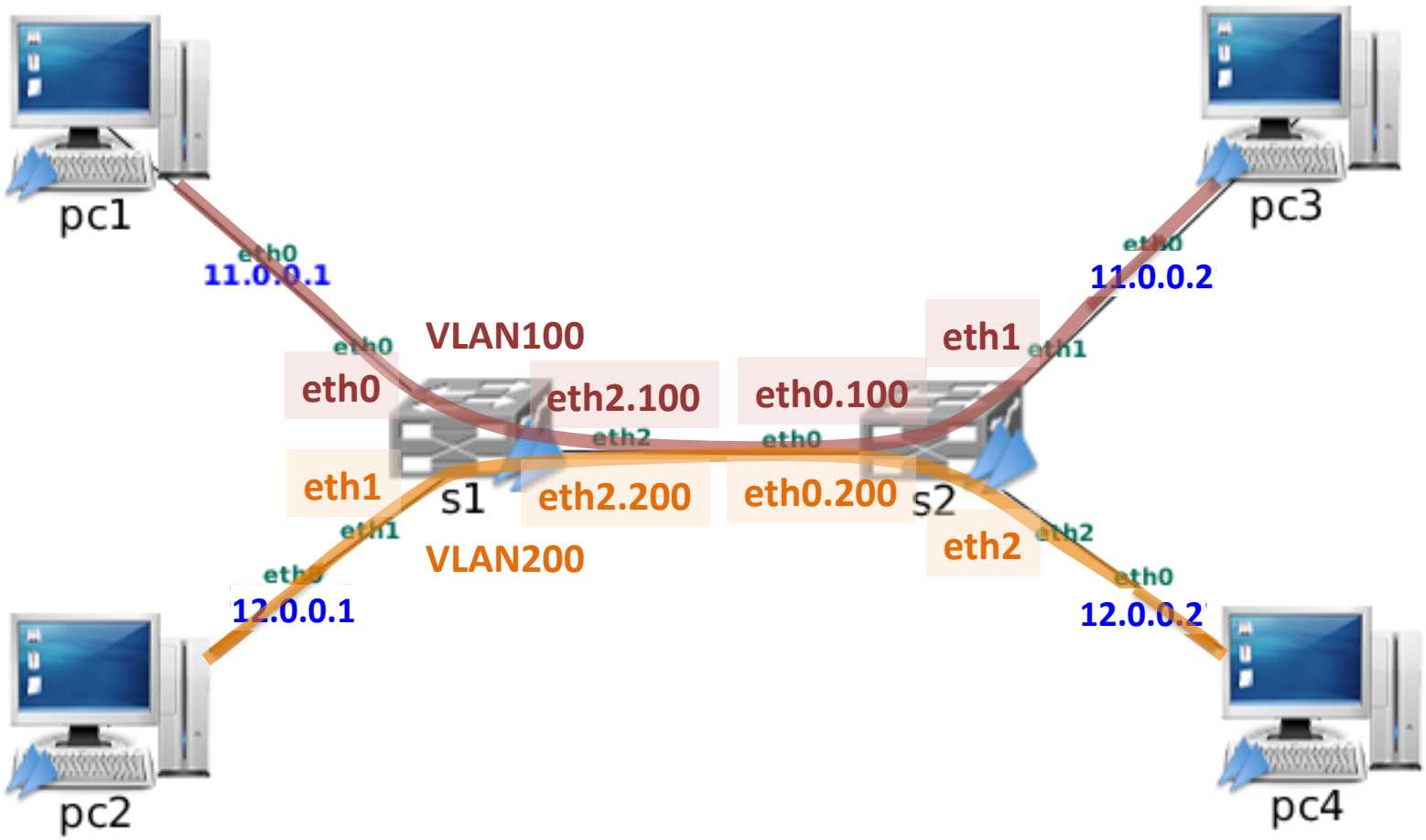
Reenvío entre interfaces de la misma VLANs

- Una vez identificadas las interfaces es necesario configurar el reenvío entre las interfaces de una determinada VLAN. Por ejemplo, en VLAN100:
  - En s1 configurar un *switch* que reenvíe tráfico entre las interfaces `eth0` y `eth2.100`.
  - En s2 configurar un *switch* que reenvíe tráfico entre las interfaces `eth0.100` y `eth1` .



Ejemplo de configuración de 2 VLANs

- En la figura se muestran 2 VLANs: VLAN100 y VLAN200:



Configuración de VLANs en los switches NetGUI

- **PASO 0**: Antes de comenzar la configuración de las VLANs en un *switch* de NetGUI es necesario eliminar la configuración por defecto del *switch*.
- Para configurar VLANs en un *switch* Linux se van a realizar los siguientes pasos:
  1. **PASO 1**: Crear las interfaces con ID VLAN, interfaces trunk del *switch*: `vconfig`.
  2. **PASO 2**: Activar las interfaces con ID VLAN que se corresponden con interfaces de tipo *trunk*: `ifconfig`
  3. **PASO 3**: Crear el *switch* virtual: `brctl`
  4. **PASO 4**: Configurar la función de reenvío de tramas Ethernet entre interfaces de una misma VLAN dentro del *switch* virtual: `brctl`
- En los pcs y *routers* no se configurarán VLANs, para ellos será transparente el uso de VLANs.

PASO 0: Eliminar la configuración por defecto en el switch

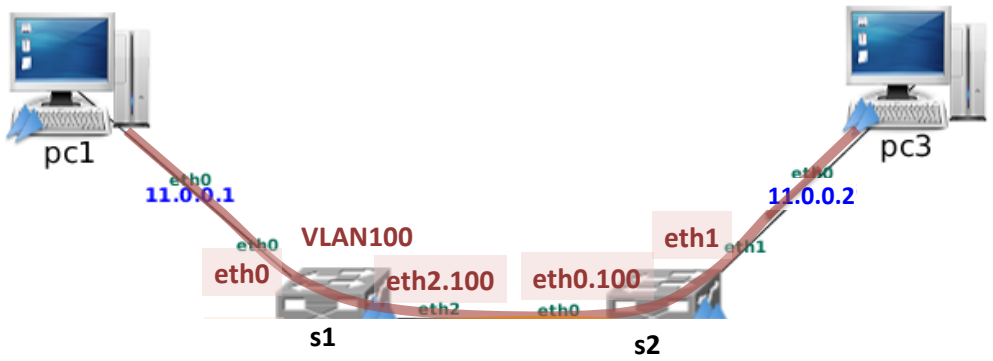
- Borrar la configuración del *switch* definido sin VLANs. Si el *switch* se llama `s1`, es necesario ejecutar lo siguiente:

```
s1:~# ifconfig s1 down
s1:~# brctl delbr s1
```

- En la figura anterior que muestra la VLAN100 y VLAN200 sería necesario borrar la configuración por defecto de los *switches*. `s1` y `s2`.

Configuración de VLAN 100 (I)

PASO 1: Crear las interfaces VLAN, las interfaces trunk, de la VLAN 100



1. **Crear las interfaces con ID VLAN, las interfaces trunk, de la VLAN 100:** `s1(eth2)` **y** `s2(eth0)`.

   `vconfig add <interfaz> <idVLAN>`

   Al ejecutar esta orden, se crea una interfaz virtual con el nombre `<interfaz>.<idVLAN>`.
   Por ejemplo, para especificar VLAN100 en `s1(eth2)`:

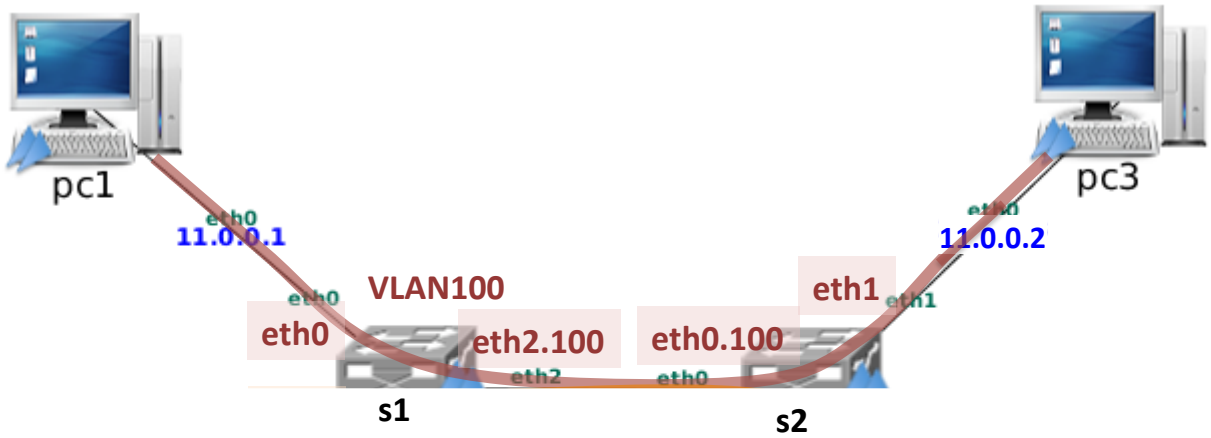
   ```
   s1:~# vconfig add eth2 100
   ```

   Para la configuración de VLAN100 habrá que usar `vconfig` en `s1(eth2)` y `s2(eth0)` creando las interfaces:

   - En `s1`: `eth2.100`
   - En `s2`: `eth0.100`

Configuración de VLAN 100 (II)

PASO 2: Activar las interfaces VLAN que se corresponden con interfaces *trunk*



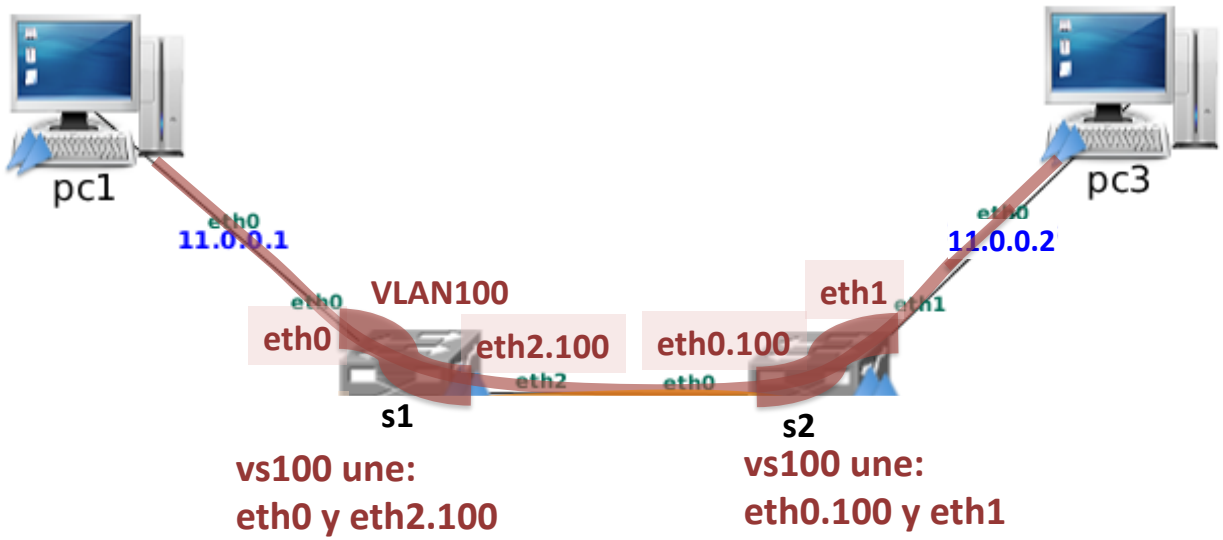
2. Una vez creadas las interfaces con ID VLAN, es necesario **activar todas las interfaces VLAN asociadas a una interfaz trunk**. En el ejemplo, sería necesario activar en s1 `eth2.100` y en s2 `eth0.100`. Por ejemplo, para activar la interfaz `eth2.100` de `s1` es necesario:

```
s1:~# ifconfig eth2.100 up
```

Al ejecutar `ifconfig` en `s1` se observará que se ha creado la interfaz VLAN `eth2.100` asociada a la interfaz `eth2`.

Configuración de VLAN 100 (III)

PASO 3: Switch virtual para una VLAN (I)



3. Crear *switches* virtuales para cada VLAN.

   En el dispositivo *switch* se crearán tantos *swicthes* virtuales superpuestos como VLANs diferentes estén definidas en dicho dispositivo. Estos *swtiches* virtuales se crean con el comando `brctl`:

   ```
   brctl addbr <nombreSwitchVirtual>
   ```

   Cada uno de esos *switches* virtuales estará encargado de hacer el reenvío de tramas de su VLAN.

Configuración de VLAN 100 (IV)

PASO 3: Switch virtual para una VLAN (II)

En s1 y s2 habrá que crear un *switch* virtual para el reenvío de tramas de VLAN100. Por ejemplo:

```
s1:~# brctl addbr vs100
```

```
s2:~# brctl addbr vs100
```

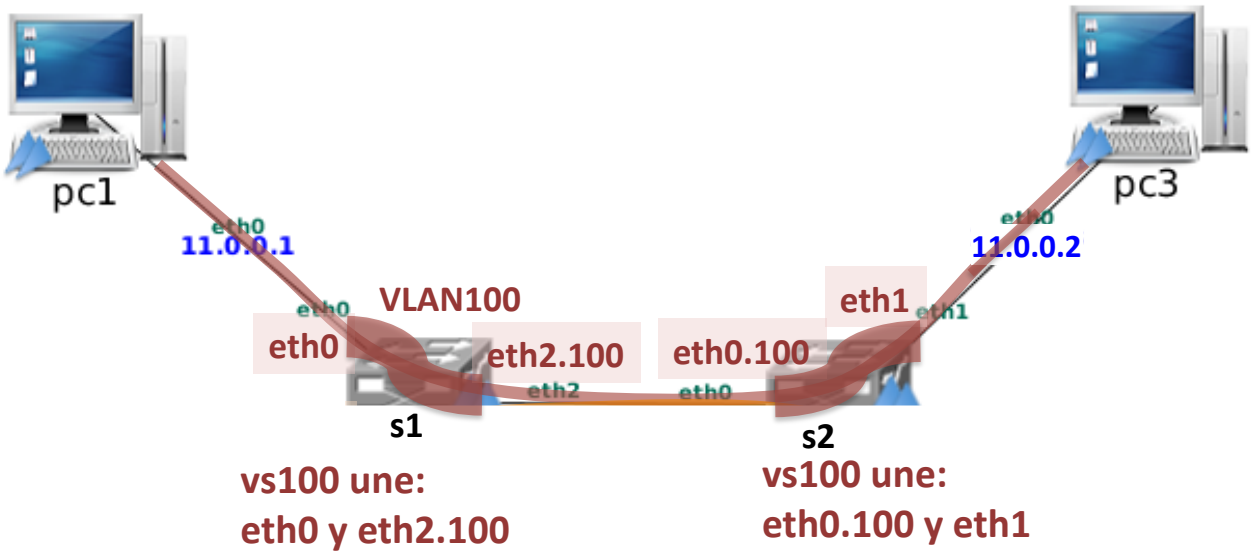
Los nombres de los *switches* pueden ser cualesquiera, elegimos el mismo para que indiquen que tanto el vs100 de s1 como el vs100 de s2 se utilizarán para la VLAN100.

Adicionalmente, cuando se configure VLAN200 será necesario crear otro *switch* virtual en s1, por ejemplo vs200, que deberá funcionar simultáneamente con vs100. Cada uno de ellos estará encargado de realizar el reenvío de tramas en VLAN100 y VLAN200 respectivamente.

E igualmente en s2.

Configuración de VLAN 100 (V)

PASO 4: Interfaces para el reenvío en una VLAN



4. Especificar las interfaces sue el *switch* virtual va a utilizar para realizar el reenvío de tráfico en cada *switch*:

```
brctl addif <nombreSwitchVirtual> <interfaz>
```

Reenvío de VLAN100 en s1:

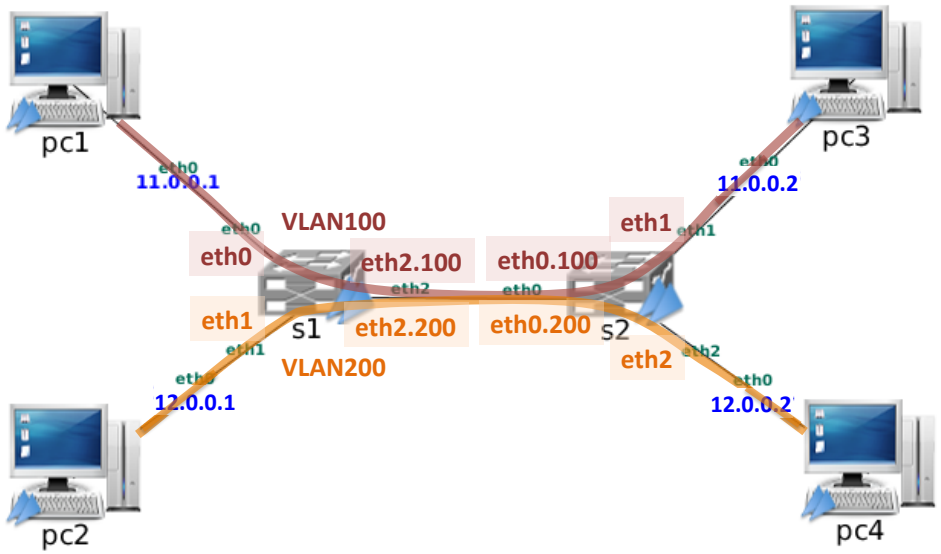
```
s1:~# brctl addif vs100 eth0
s1:~# brctl addif vs100 eth2.100
s1:~# ifconfig vs100 up
```

Reenvío de VLAN100 en s2:

```
s2:~# brctl addif vs100 eth0.100
s2:~# brctl addif vs100 eth1
s2:~# ifconfig vs100 up
```

Configuración de VLAN 200

- Una vez configurada VLAN100, para configurar VLAN200:
  1. Crear las interfaces con ID VLAN de VLAN200: en s1 `eth2.200` y en s2 `eth0.200`
  2. Activar en s1 la interfaz `eth2.200` y en s2 `eth0.200`.
  3. Crear los *switches* virtuales para VLAN200 en `s1` y `s2`, por ejemplo: `vslan200` para `s1` y `vslan200` para `s2`.
  4. Configurar la función de reenvío de tramas Ethernet entre las interfaces de `s1` y `s2`



Comprobar la configuración del reenvío en un switch

- Una vez configurado el reenvío en un *switch*, con el comando `brctl show` puede comprobarse la configuración aplicada. Así en `s1`:

```
s1:~# brctl show
bridge name     bridge id               STP enabled     interfaces
vs100           8000.1a65e4986698       no              eth0
                                                        eth2.100
vs200           8000.1a65e4986698       no              eth1
                                                        eth2.200
```