Administración de Redes Locales
Netkit

Caramutti Ignacio, Gonzalez Rosario, Olivares Ana
5to Informática

Netkit:

Netkit es un entorno para configurar y realizar experimentos de redes a bajo costo y con poco esfuerzo. Permite "crear" varios dispositivos de red virtual (enrutadores completos, conmutadores, computadoras, etc.) que pueden interconectarse fácilmente para formar una red en una sola PC. Los equipos de red son virtuales pero presentan muchas de las características de los reales, incluida la interfaz de configuración.

Las máquinas virtuales utilizan un sistema de archivos personalizado que contiene una instalación estándar de Linux, que incluye software orientado a la red, como routing daemons (RIP, OSPF, etc.), servidores (FTP, HTTP, etc.), herramientas de diagnóstico (ping, traceroute, tcpdump, etc.) y otras aplicaciones.

Iniciando Netkit:

Para iniciar netkit debemos entrar en la terminal de Linux y escribir el siguiente comando:

```
$NETKIT_HOME
```

Una vez ingresado en Netkit, se puede empezar a trabajar. Primero se debe crear un directorio e ingresar a este, utilizando los comandos:

```
mkdir lab
cd lab/
```

Siendo lab el nombre de nuestro directorio.

Después de ingresar al directorio, "creamos" una carpeta para cada uno de los dispositivos de nuestra red virtual, utilizando el mismo comando `mkdir`

.

Luego, creamos un archivo de texto donde configuraremos nuestra red virtual. Los nombres utilizados aquí deben coincidir con los de las carpetas ya creadas.

```
nano lab.config
```

Por último, iniciamos nuestra red con el comando:

```
lstart
```

Bridge:

Un brigde conecta dos segmentos de red como una sola red, haciendo el pasaje de datos de una red hacia otra, con base en la dirección física de destino de cada paquete. Utiliza un mismo protocolo de establecimiento de red. Opera en la capa 2 del modelo OSI.

Switch:

Un switch es un dispositivo digital lógico de interconexión de equipos que opera en la capa de enlace de datos del modelo OSI. Su función es interconectar dos o más host de manera similar a los puentes de red, pasando datos de un segmento a otro de acuerdo con la dirección MAC de destino de las tramas en la red y eliminando la conexión una vez finalizada ésta.

Sniffer:

Un Sniffer es un programa de captura de las tramas de una red de computadoras.

Es algo común que, por topología de red y necesidad material, el medio de transmisión sea compartido por varias computadoras y dispositivos de red, lo que hace posible que un ordenador capture las tramas de información no destinadas a él. Para conseguir esto el analizador pone la tarjeta de red en un estado conocido como

"modo promiscuo" en el cual en la capa de enlace de datos no son descartadas las tramas no destinadas a la dirección MAC de la tarjeta; de esta manera se puede capturar todo el tráfico que viaja por la red.

Los analizadores de paquetes tienen diversos usos, como monitorear redes para detectar y analizar fallos, o para realizar ingeniería inversa en protocolos de red.

Ejemplo:

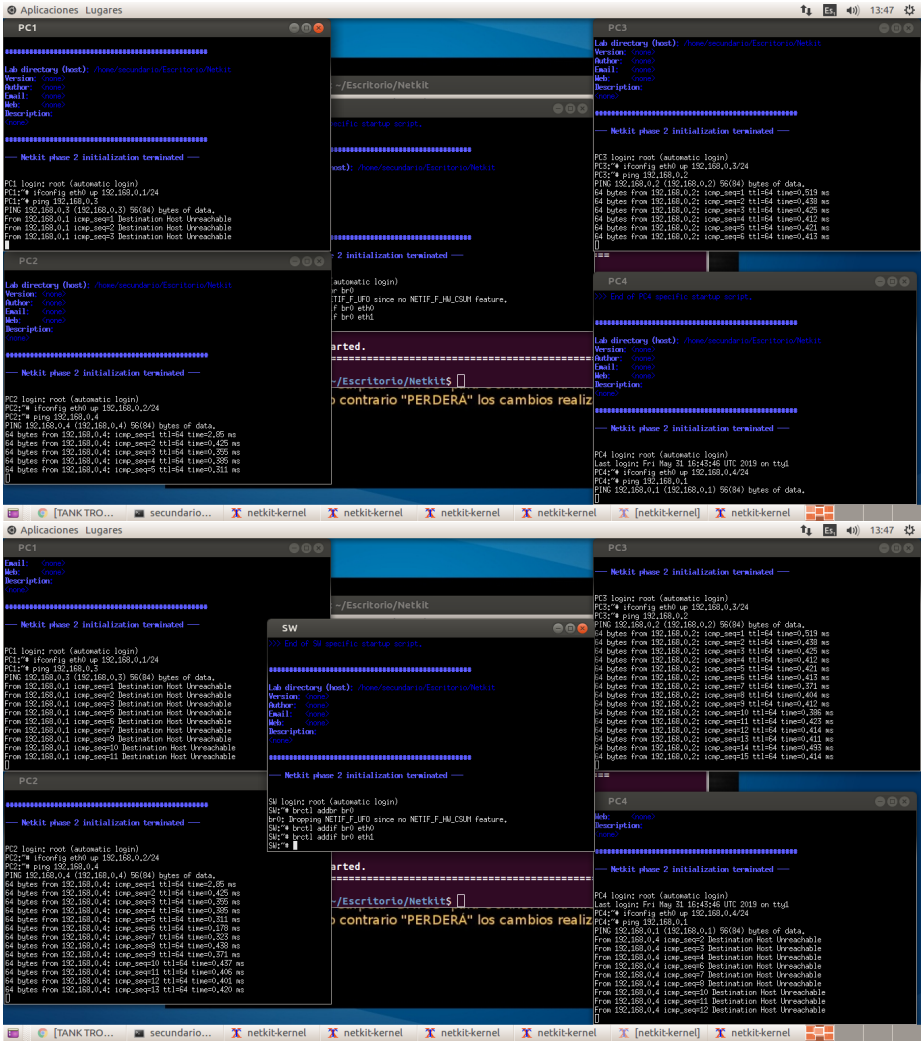
Creamos cuatros PC's virtuales, un Switch y un Sniffer virtual. Conectamos la PC1 y la PC2 al cable "A", la PC3 y PC4 al cable "B" y el SW y el SNIFF a estos dos cables. Los comandos utilizados son:

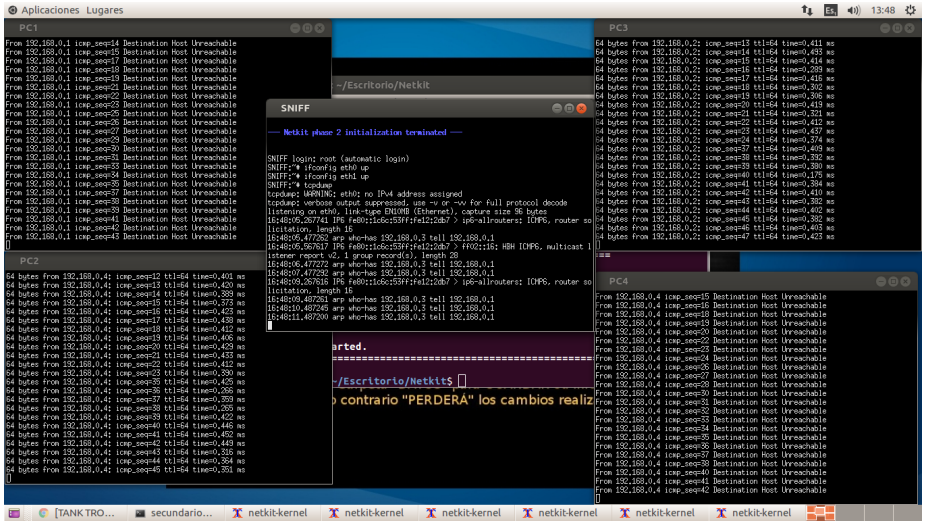
`brctl addbr br0`
Este comando añade un bridge.

```
brctl addif br0 eth0
brctl addif br0 eth1
```
Conectan los brigde con el switch.

```
ifconfig eth0 up
ifconfig eth1 up
ifconfig br0 up
```
Los últimos comandos comienzan el bridge.

`tcpdump -i any`
Analiza el tráfico. Lo utilizamos con el SNIFF.


Aplicaciones Lugares
13:47
PC1
PC2
PC3
PC4
SW
[TANK TRO...
secundario...
netkit-kernel
[netkit-kernel]


Aplicaciones Lugares
13:48
PC1
PC2
PC3
PC4
SNIFF
~/Escritorio/Netkit
Netkit phase 2 initialization terminated
SNIFF login: root (automatic login)
SNIFF:~# ifconfig eth0 up
SNIFF:~# ifconfig eth1 up
SNIFF:~# tcpdump
tcpdump: WARNING: eth0: no IPv4 address assigned
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
~/Escritorio/Netkit$
contrario "PERDERÁ" los cambios realiz
[TANK TRO...
secundario...
netkit-kernel
netkit-kernel
netkit-kernel
netkit-kernel
netkit-kernel
netkit-kernel

STP:

STP significa Spanning Tree Protocol (Protocolo de árbol de expansión), es un protocolo de red de capa 2 del modelo OSI y,se encarga de reconocer y administrar bucles en topologias de redes nacidos a función de la redundancia en la misma.

Es una practica común que en el diseño de la topologia de una red se agreguen enlaces redundantes con el fin de hacer la red tolerable a fallas y de esta forma si falla un enlace la red tiene la capacidad de recuperarse inmediatamente sin que el usuario ni siquiera note la falla.

Este tipo de falla se conoce como tormenta de broadcast: los broadcast en la red son reenviados una y otra vez y permanecen circulando en la misma sin fin, dado que en Ethernet no existe como en IP un campo de TTL. Lógicamente, al no eliminarse la situación se agrava con cada nuevo broadcast.

Ejemplo:

Creamos las PC's y los SW de manera que se produzca la tormenta de broadcast. Para arreglar la tormenta de broadcast, activamos el protocolo STP de la siguiente manera:

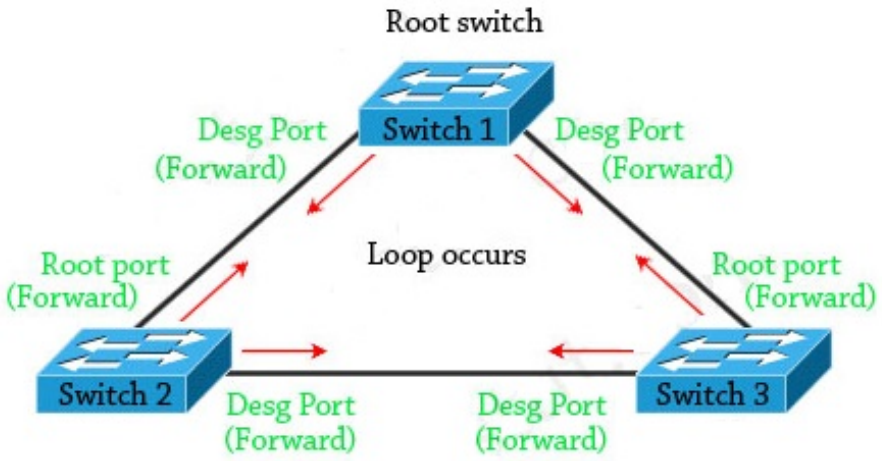
```
ifconfig br0 down
```

Este comando apaga el switch.

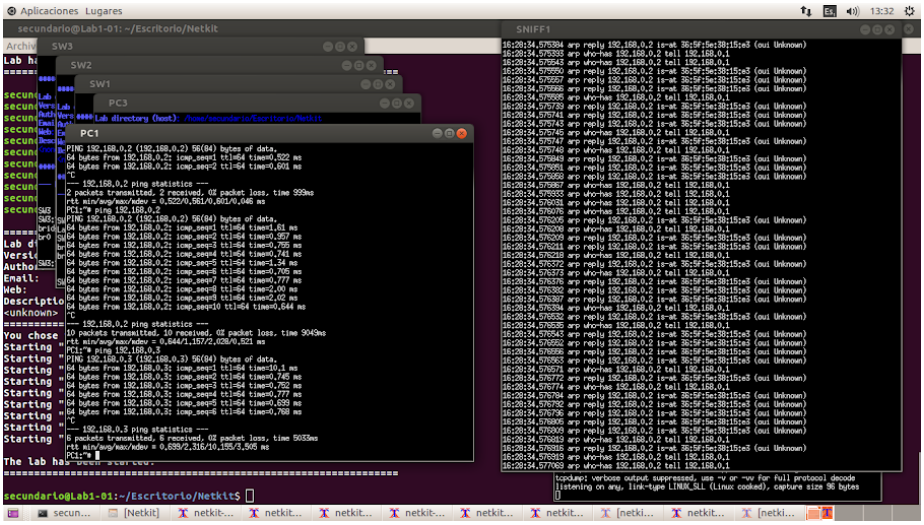
```
brctl stp br0 on
```

Activa el protocolo STP.

`ifconfig br0 up`
Activa nuevamente el switch.



Tormenta de broadcast



Port Bonding:

El Port Bonding es una técnica que permite agregar varios interfaces de red físicos en uno único virtual. A cada interfaz físico se le denominará slave. Con esto podemos realizar un balanceo de carga entre las dos interfaces y conseguir un ancho de banda final igual a la suma de los anchos de banda de cada slave. Además de una ventaja adicional inmediata: redundancia de la conexión, lo que implica que si tenemos varios enlaces físicos a la red, perder alguno de ellos supondría una degradación de servicio pero no la pérdida completa de conexión. Tiene 7 posibles "modos" para interfaces enlazadas. Estos modos determinan la manera en que el tráfico enviado desde la interfaz enlazada se dispersa en las interfaces reales.

Modo 0 - Balance -rr: Este modo transmite paquetes en un orden secuencial desde el primer slave disponible hasta el último. Si dos interfaces reales son slaves en el enlace y dos paquetes llegan destinados fuera de la interfaz enlazada, el primero se transmitirá en el primer slave y la segunda trama se transmitirá en el segundo slave. El tercer paquete se enviará el primero y así sucesivamente. Esto proporciona equilibrio de carga y tolerancia a fallos.

Modo 1 - Active Backup: (copia de seguridad activa). Coloca una de las interfaces en un estado de respaldo y solo la activará si la interfaz activa pierde el enlace. Solo un slave en el enlace está activo en un momento determinado. Un slave diferente se activa solo

cuando falla el slave activo. Este modo proporciona tolerancia a fallos.

Modo 2 - Balance XOR: Transmite en base a la fórmula XOR. Esto selecciona el mismo slave para cada dirección MAC de destino y proporciona balanceo de carga y tolerancia a fallas.

Modo 3 - Broadcast: El modo de transmisión transmite todo en todas las interfaces de slave. Este modo se usa menos (solo para un propósito específico) y proporciona solo tolerancia a fallas.

Modo 4 - IEEE 802.3ad: El modo 802.3ad se conoce como modo de agregación de enlace dinámico. Crea grupos de agregación que comparten la misma velocidad y configuraciones de dúplex. Este modo requiere un conmutador que admita el enlace dinámico IEEE 802.3ad. La selección de slaves para el tráfico saliente se realiza de acuerdo con la política de hash de transmisión. No todas las políticas de transmisión pueden ser compatibles con 802.3ad, particularmente en lo que respecta a los requisitos de pedido erróneo de paquetes de la sección 43.2.4 de la norma 802.3ad. Las diferentes implementaciones entre pares tendrán diferentes tolerancias para el incumplimiento.

Modo 5 - Balance-tlb (Adaptive Transmit Load Balancing): El tráfico saliente se distribuye de acuerdo con la carga actual y la cola en cada interfaz de slave. El tráfico entrante es recibido por el slave actual.

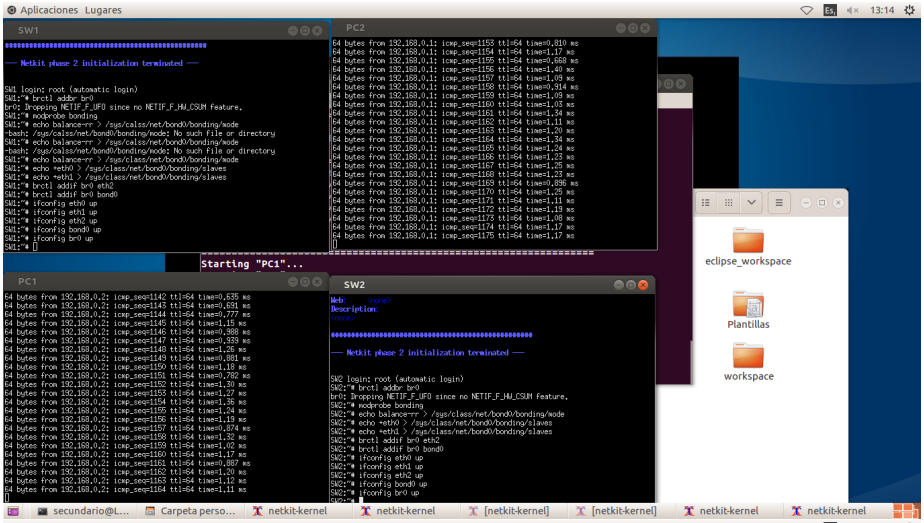
Modo 6 - Balance-alb (Adaptive Load Balancing): Este es el modo de equilibrio de carga adaptable. Esto incluye balance-tlb + recibir balanceo de carga (rlb) para el tráfico IPV4. El equilibrio de carga de recepción se logra mediante la negociación ARP. El controlador de enlace intercepta las Respuestas ARP enviadas por el servidor al salir y sobrescribe la dirección src hw con la dirección hw única de uno de los slaves en el enlace, de modo que diferentes clientes utilicen diferentes direcciones hw para el servidor.

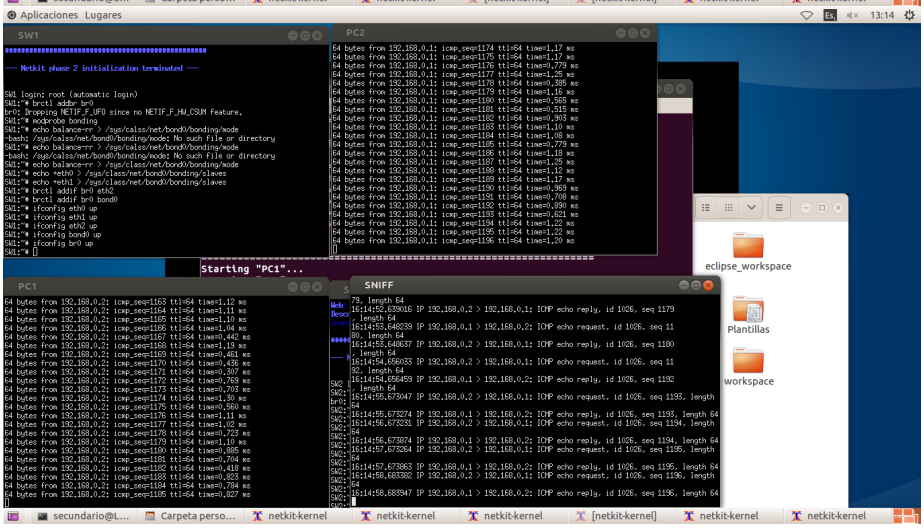
Ejemplo:

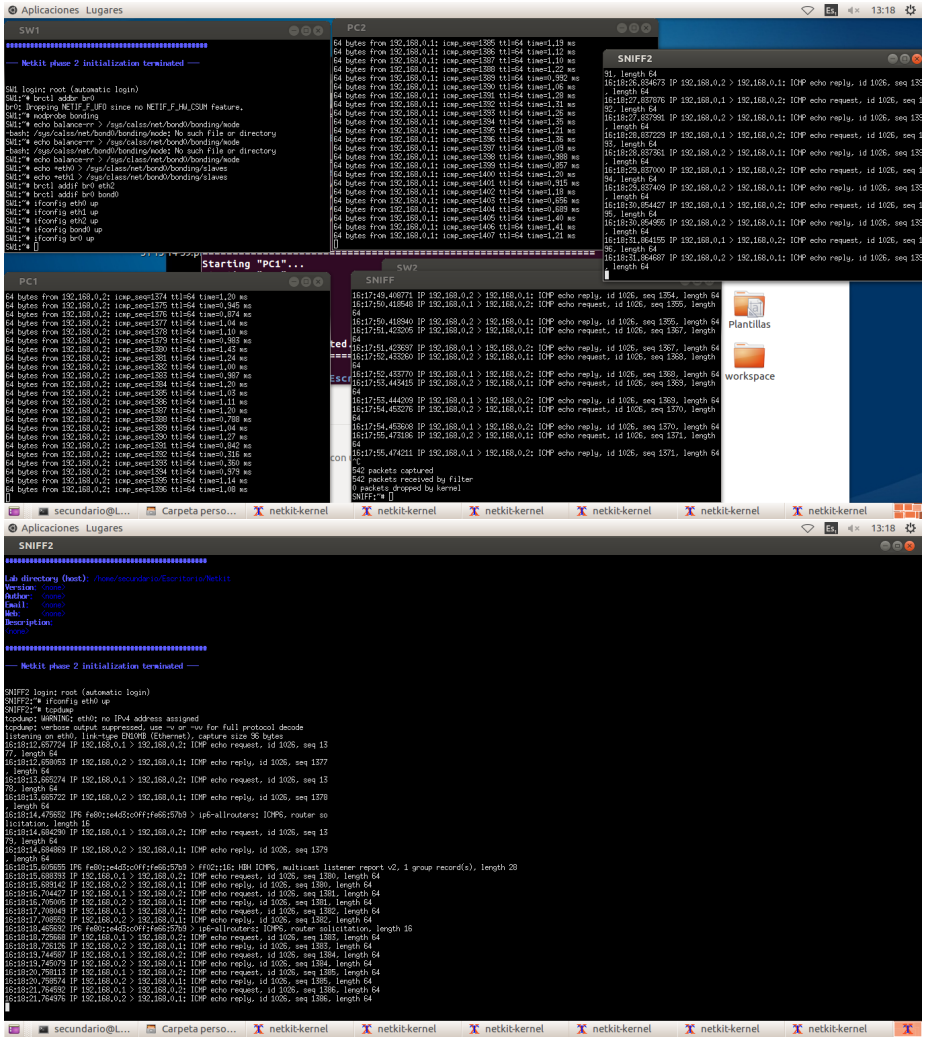
Creamos dos PC's, dos SW y los conectamos. Los comandos utilizados son:

```
modprobe bonding
```
Cargamos el driver

```
echo balance-rr > /sys/class/net/bond0/bonding/mpd
echo +eth0 > /sys/class/net/bond0/bonding/slaves
echo +eth1 > /sys/class/net/bond0/bonding/slaves
brctl addif br0 eth0 eth2
brctl addif br0 eth0 bond0
ifconfig eth0 up
ifconfig eth1 up
ifconfig eth2 up
ifconfig bond0 up
ifconfig br0 up
```


Aplicaciones Lugares
13:14
SW1
PC2
Starting "PC1"...
PC1
SW2
eclipse_workspace
Plantillas
workspace
secundario@L...
Carpeta perso...
netkit-kernel
[netkit-kernel]


Aplicaciones Lugares
13:14
SW1
PC2
Starting "PC1"...
PC1
SNIFF
eclipse_workspace
Plantillas
workspace
secundario@L...
Carpeta perso...
netkit-kernel
[netkit-kernel]


Aplicaciones Lugares
13:18
SW1
PC2
SNIFF2
Starting "PC1"...
PC1
SW2
SNIFF
Plantillas
workspace
secundario@L...
Carpeta perso...
netkit-kernel
Aplicaciones Lugares
13:18
SNIFF2
secundario@L...
Carpeta perso...
netkit-kernel

VLAN:

Virtual LAN, es un método para crear redes lógicas independientes dentro de una misma red física. Varias VLAN pueden coexistir en un único conmutador físico o en una única red física. Son útiles para reducir el dominio de difusión y ayudan en la administración de la red, separando segmentos lógicos de una red de área local que no deberían intercambiar datos usando la red local (aunque podrían hacerlo a través de un enrrutador o un conmutador de capa OSI 3 y 4).

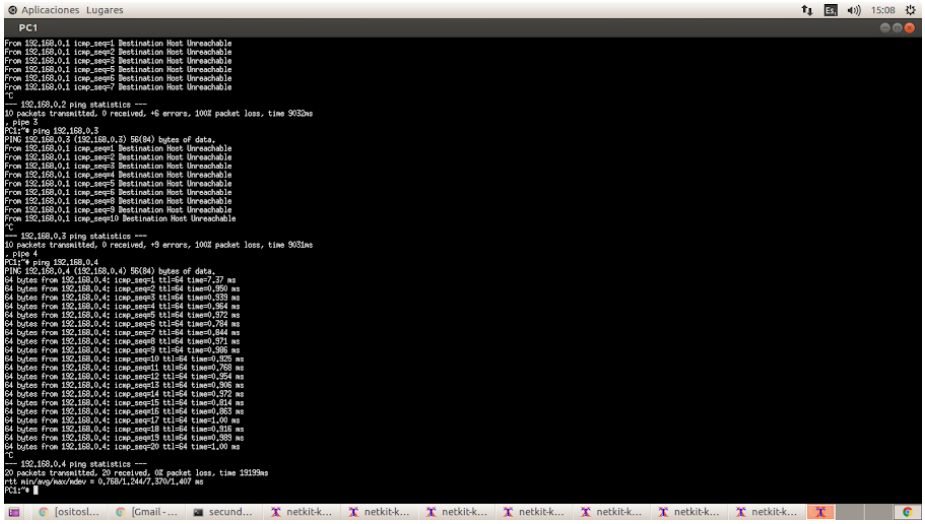
Una VLAN consiste en dos o más redes de computadoras que se comportan como si estuviesen conectados al mismo conmutador, aunque se encuentren físicamente conectados a diferentes segmentos de una red de área local.

Ejemplo:

Creamos las PC's y pingueamos la PC1 tanto con la PC3 como con la PC2 y la PC4. Los paquetes sólo son alcanzados por aquellas PC's con la misma id. Los comandos que utilizamos son:

```
ifconfig eth0 up
vconfig add eth0 N°ID
ifconfig eth0.N°ID 192.168.0 N°PC/24 up
```

Cada uno con su respectivo número que lo identifica.


Aplicaciones Lugares
15:08
PC1
From 192.168.0.1 icmp_seq=1 Destination Host Unreachable
From 192.168.0.1 icmp_seq=2 Destination Host Unreachable
From 192.168.0.1 icmp_seq=3 Destination Host Unreachable
From 192.168.0.1 icmp_seq=5 Destination Host Unreachable
From 192.168.0.1 icmp_seq=6 Destination Host Unreachable
From 192.168.0.1 icmp_seq=7 Destination Host Unreachable
^C
--- 192.168.0.2 ping statistics ---
10 packets transmitted, 0 received, +6 errors, 100% packet loss, time 9032ms
, pipe 3
PC1:~# ping 192.168.0.3
PING 192.168.0.3 (192.168.0.3) 56(84) bytes of data.
From 192.168.0.1 icmp_seq=1 Destination Host Unreachable
From 192.168.0.1 icmp_seq=2 Destination Host Unreachable
From 192.168.0.1 icmp_seq=3 Destination Host Unreachable
From 192.168.0.1 icmp_seq=4 Destination Host Unreachable
From 192.168.0.1 icmp_seq=5 Destination Host Unreachable
From 192.168.0.1 icmp_seq=6 Destination Host Unreachable
From 192.168.0.1 icmp_seq=8 Destination Host Unreachable
From 192.168.0.1 icmp_seq=9 Destination Host Unreachable
From 192.168.0.1 icmp_seq=10 Destination Host Unreachable
^C
--- 192.168.0.3 ping statistics ---
10 packets transmitted, 0 received, +9 errors, 100% packet loss, time 9031ms
, pipe 4
PC1:~# ping 192.168.0.4
PING 192.168.0.4 (192.168.0.4) 56(84) bytes of data.
64 bytes from 192.168.0.4: icmp_seq=1 ttl=64 time=7,37 ms
64 bytes from 192.168.0.4: icmp_seq=2 ttl=64 time=0,950 ms
64 bytes from 192.168.0.4: icmp_seq=3 ttl=64 time=0,933 ms
64 bytes from 192.168.0.4: icmp_seq=4 ttl=64 time=0,964 ms
64 bytes from 192.168.0.4: icmp_seq=5 ttl=64 time=0,972 ms
64 bytes from 192.168.0.4: icmp_seq=6 ttl=64 time=0,704 ms
64 bytes from 192.168.0.4: icmp_seq=7 ttl=64 time=0,844 ms
64 bytes from 192.168.0.4: icmp_seq=8 ttl=64 time=0,971 ms
64 bytes from 192.168.0.4: icmp_seq=9 ttl=64 time=0,986 ms
64 bytes from 192.168.0.4: icmp_seq=10 ttl=64 time=0,925 ms
64 bytes from 192.168.0.4: icmp_seq=11 ttl=64 time=0,768 ms
64 bytes from 192.168.0.4: icmp_seq=12 ttl=64 time=0,954 ms
64 bytes from 192.168.0.4: icmp_seq=13 ttl=64 time=0,906 ms
64 bytes from 192.168.0.4: icmp_seq=14 ttl=64 time=0,972 ms
64 bytes from 192.168.0.4: icmp_seq=15 ttl=64 time=0,814 ms
64 bytes from 192.168.0.4: icmp_seq=16 ttl=64 time=0,863 ms
64 bytes from 192.168.0.4: icmp_seq=17 ttl=64 time=1,00 ms
64 bytes from 192.168.0.4: icmp_seq=18 ttl=64 time=0,916 ms
64 bytes from 192.168.0.4: icmp_seq=19 ttl=64 time=0,983 ms
64 bytes from 192.168.0.4: icmp_seq=20 ttl=64 time=1,00 ms
^C
--- 192.168.0.4 ping statistics ---
20 packets transmitted, 20 received, 0% packet loss, time 19199ms
rtt min/avg/max/mdev = 0,768/1,244/7,370/1,407 ms
PC1:~#
[ositosl...
[Gmail - ...
secund...
netkit-k...
netkit-k...
netkit-k...
netkit-k...
netkit-k...
netkit-k...
netkit-k...

Bibliografía:

http://wiki.netkit.org/man/man7/netkit.7.html
http://wiki.netkit.org/index.php/Main_Page
https://es.wikipedia.org/wiki/Conmutador_(dispositivo_de_red)
https://cl0udswxsequre.wordpress.com/2015/03/10/que-es-stp-y-para-que-sirve/
https://www.cloudibee.com/network-bonding-modes/
https://es.wikipedia.org/wiki/VLAN