

# PROYECTO REDES 1



201830498 - Oscar Antonio de León Urizar.

201832069 - Marcos Andrés Aguare Bravo.

201930643 - Jeffrey Kenneth Menendez Castillo

201930693 - Leví Isaac Hernández Sapón.

201930699 - Erick Daniel Morales Xicará.

201931707 - Luis Emilio Maldonado Rodriguez.

# OBJETIVO

El propósito principal de este proyecto es explorar la virtualización de redes con QEMU, destacando el uso de iptables para la gestión del tráfico, y la implementación de una variedad de sistemas operativos, incluyendo Kali Linux, Ubuntu y Debian.



# QEMU

# Configuracion Cliente - Router

La configuracion del cliente se estandarizo de una manera que fuera mas optima para la configuracion entre los routers.

```
1 INTERFAZ=$2
2
3 if [ $1 == "A" ]
4 then
5
6     ip addr add 10.22.0.2/29 dev $INTERFAZ
7     ip link set $INTERFAZ up
8
9     ip route add default via 10.22.0.1 dev $INTERFAZ
10
11 elif [ $1 == "B" ]
12 then
13
14     ip addr add 10.23.0.2/26 dev $INTERFAZ
15     ip link set $INTERFAZ up
16
17     ip route add default via 10.23.0.1 dev $INTERFAZ
18
```

```
19
20 elif [ $1 == "C" ]
21 then
22
23     ip addr add 10.24.0.2/27 dev $INTERFAZ
24     ip link set $INTERFAZ up
25
26     ip route add default via 10.24.0.1 dev $INTERFAZ
27
28 elif [ $1 == "D" ]
29 then
30
31     ip addr add 10.25.0.2/29 dev $INTERFAZ
32     ip link set $INTERFAZ up
33
34     ip route add default via 10.25.0.1 dev $INTERFAZ
35 fi
```

Conexion entre Router y cliente

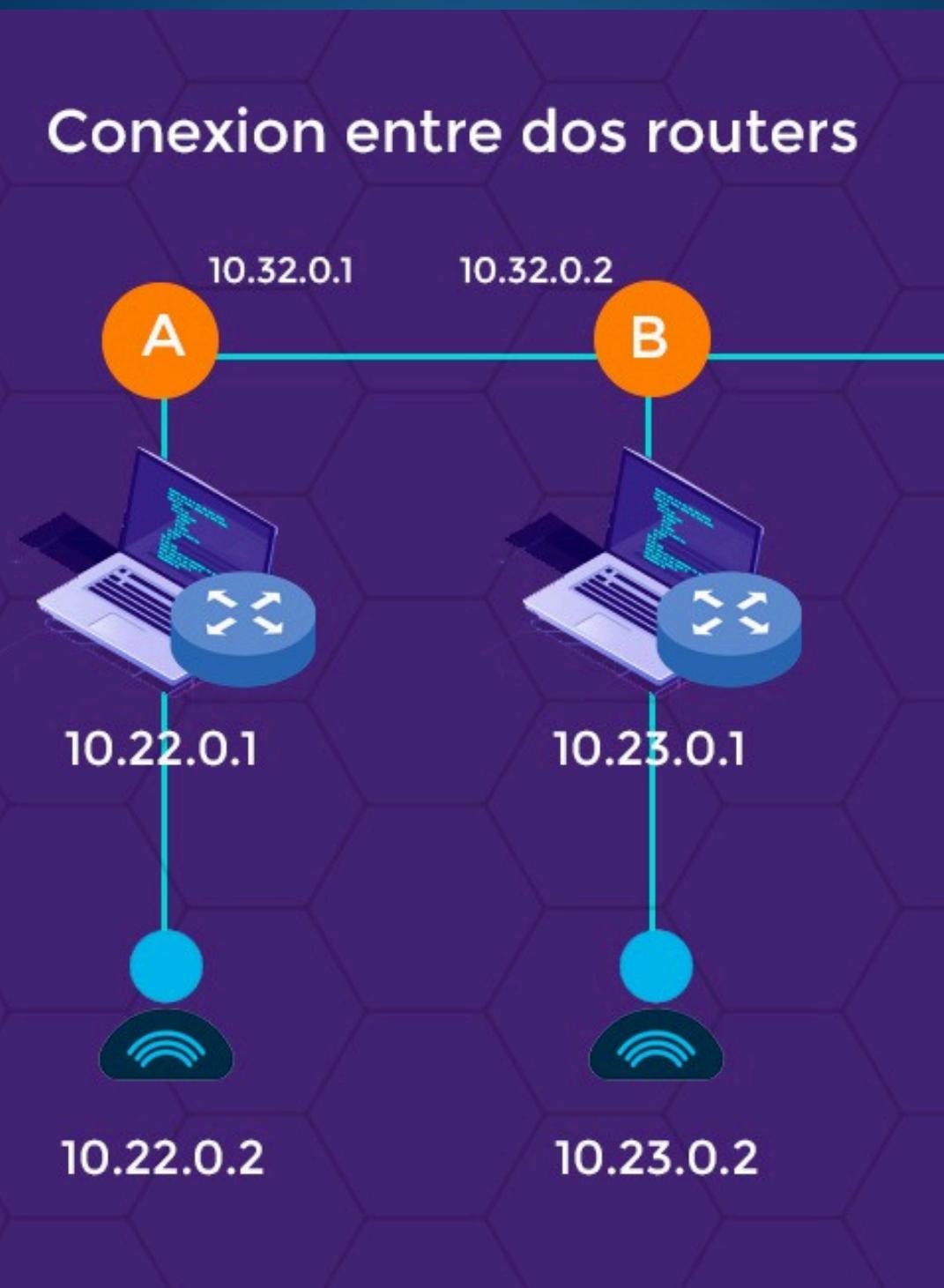


# Conexion Router - Router

Para la resolución de este problema decidimos dividir por segmentos las redes y así poder trabajar paso a paso.

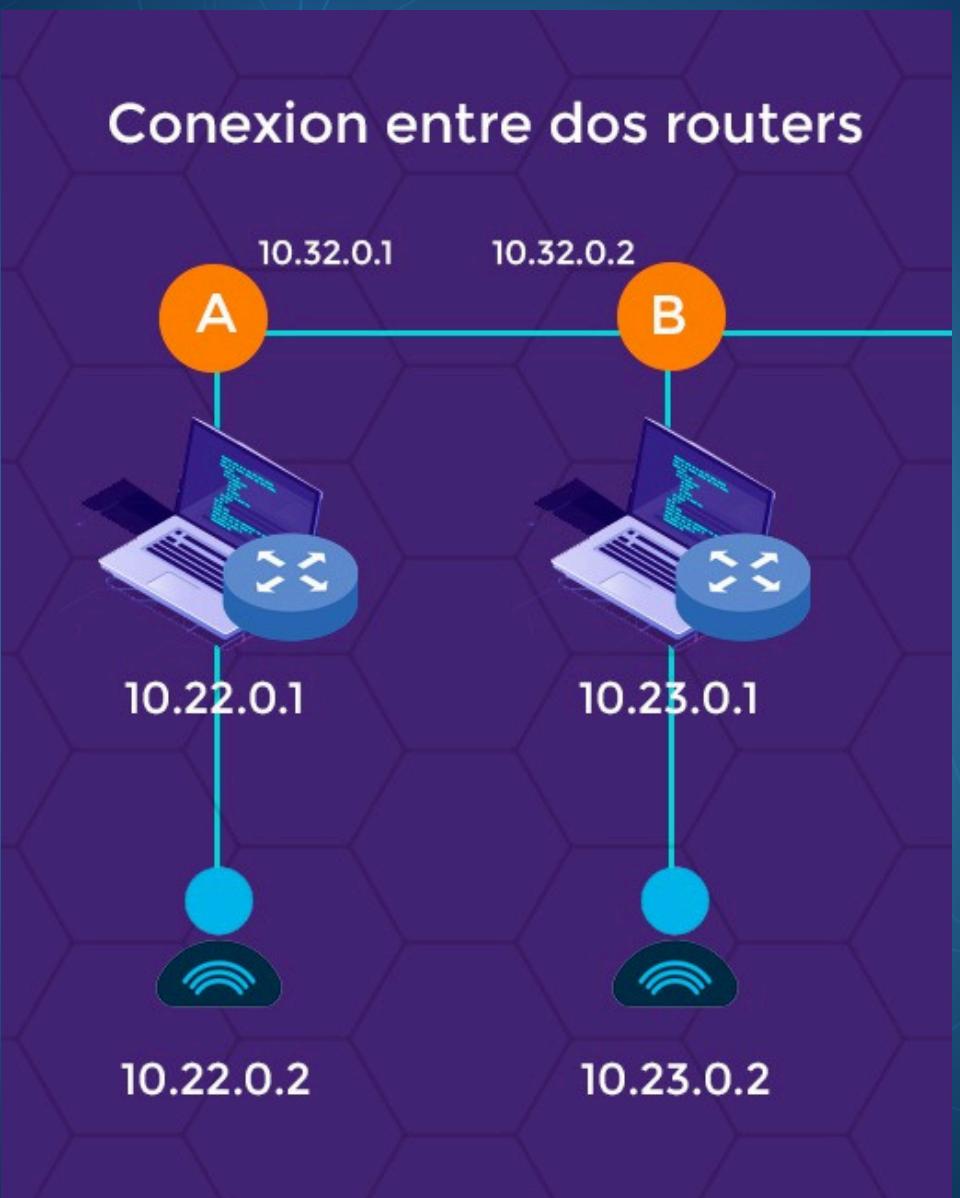
```
● ● ●  
1 # AGREGAMOS LA IP DE LA RED INTERNA A LA INTERFAZ Y LEVANTAMOS  
2 ip addr add 10.22.0.1/29 dev $INTERFAZ_RED_INTERNA  
3 ip link set $INTERFAZ_RED_INTERNA up  
4  
5 # AGREGAMOS LA IP PÚBLICA CON LA SE COMUNICARÁ A CON B Y LEVANTAMOS  
6 ip addr add 10.32.0.1/30 dev $INTERFAZ_OTRO_ROUTER  
7 ip link set $INTERFAZ_OTRO_ROUTER up  
8  
9 # DIRIGIMOS TODO EL TRÁFICO DEL ROUTER AL SIGUIENTE ROUTER (B)  
10 ip route add default via 10.32.0.2 dev $INTERFAZ_OTRO_ROUTER  
11  
12 # pendiente ver si necesita msquerade  
13
```

```
● ● ●  
1 # AGREGAMOS LA IP DE LA RED INTERNA A LA INTERFAZ Y LEVANTAMOS  
2 ip addr add 10.25.0.1/29 dev $INTERFAZ_RED_INTERNA  
3 ip link set $INTERFAZ_RED_INTERNA up  
4  
5 # AGREGAMOS LA IP PÚBLICA CON LA SE COMUNICARÁ D CON C Y LEVANTAMOS  
6 ip addr add 10.35.0.2/30 dev $INTERFAZ_OTRO_ROUTER  
7 ip link set $INTERFAZ_OTRO_ROUTER up  
8  
9 # DIRIGIMOS TODO EL TRÁFICO DEL ROUTER AL SIGUIENTE ROUTER (C)  
10 ip route add default via 10.35.0.1 dev $INTERFAZ_OTRO_ROUTER
```



# Conexion Router - Router

“Replicamos” esta configuración para crear la red B, seguidamente procedemos a conectarlas por la interfaz ethernet

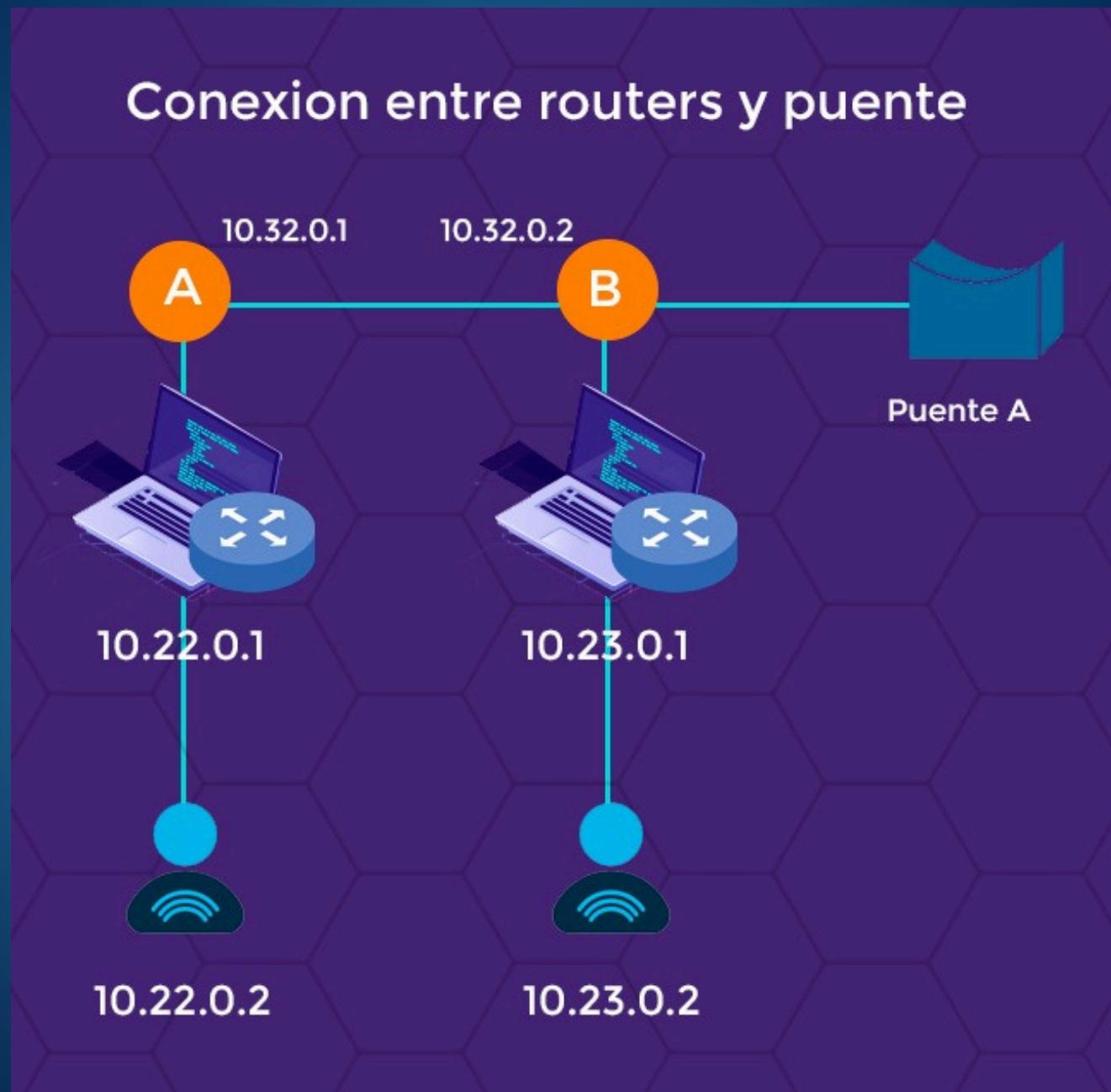


```
1 if [ $1 == "B" ]
2 then
3
4 # ASIGNAMOS LA IP DE LA RED INTERNA A LA INTERFAZ Y LEVANTAMOS
5 ip addr add 10.23.0.1/26 dev $INTERFAZ_INTERNA
6 ip link set $INTERFAZ_INTERNA up
7
8 # ASIGNAMOS LA IP CON LA QUE SE CONECTARÁ B CON A
9 ip addr add 10.32.0.2/30 dev $INTERFAZ_OTRO_ROUTER
10 ip link set $INTERFAZ_OTRO_ROUTER up
11
12 # REDIRECCIONAMOS EL TRÁFICO RELACIONADO CON A A SU IP PÚBLICA
13 ip route add 10.22.0.0/29 via 10.32.0.1 dev $INTERFAZ_OTRO_ROUTER
14
15 # REDIRECCIONAMOS TODO EL DEMÁS TRÁFICO HACIA EL PUENTE
16 ip route add default via 10.42.0.1 dev $INTERFAZ_PUENTE
17
18 elif [ $1 == "C" ]
19 then
20
21 # ASIGNAMOS LA IP DE LA RED INTERNA A LA INTERFAZ Y LEVANTAMOS
22 ip addr add 10.24.0.1/27 dev $INTERFAZ_INTERNA
23 ip link set $INTERFAZ_INTERNA up
24
25 # ASIGNAMOS LA IP CON LA QUE SE CONECTARÁ C A D
26 ip addr add 10.35.0.1/30 dev $INTERFAZ_OTRO_ROUTER
27 ip link set $INTERFAZ_OTRO_ROUTER up
28
29 # REDIRECCIONAMOS EL TRÁFICO RELACIONADO CON D A SU IP PÚBLICA
30 ip route add 10.25.0.0/29 via 10.35.0.2 dev $INTERFAZ_OTRO_ROUTER
31
32 # REDIRECCIONAMOS TODO EL DEMÁS TRÁFICO HACIA EL PUENTE
33 ip route add default via 10.42.0.1 dev $INTERFAZ_PUENTE
34
35 fi
36
37 # CONFIGURAMOS LA TABLA NAT PARA ENMASCARAR EL PAQUETE LUEGO DE ENRUTARLO
38 # AL PUENTE PERO ANTES DE ENVIARLO
39 iptables -t nat -A POSTROUTING -o $INTERFAZ_PUENTE -j MASQUERADE
40
```

# Conexion Router - Puente

La configuración anterior, se debe conectar al puente transparente

```
● ● ●  
1 IP_ROUTER=$1  
2 INTERFAZ=$2  
3  
4 if [ $1 == "D" ]  
5 then  
6     ip route add 10.23.0.0/26 via $IP_ROUTER dev $INTERFAZ  
7     ip route add 10.22.0.0/29 via $IP_ROUTER dev $INTERFAZ  
8     iptables -t nat -A PREROUTING -d 10.23.0.2 -j DNAT --to-destination $IP_ROUTER  
9     iptables -t nat -A PREROUTING -d 10.22.0.2 -j DNAT --to-destination $IP_ROUTER  
10  
11 else  
12     ip route add 10.24.0.0/27 via $IP_ROUTER dev $INTERFAZ  
13     ip route add 10.25.0.0/29 via $IP_ROUTER dev $INTERFAZ  
14     iptables -t nat -A PREROUTING -d 10.24.0.2 -j DNAT --to-destination $IP_ROUTER  
15     iptables -t nat -A PREROUTING -d 10.25.0.2 -j DNAT --to-destination $IP_ROUTER  
16 fi
```

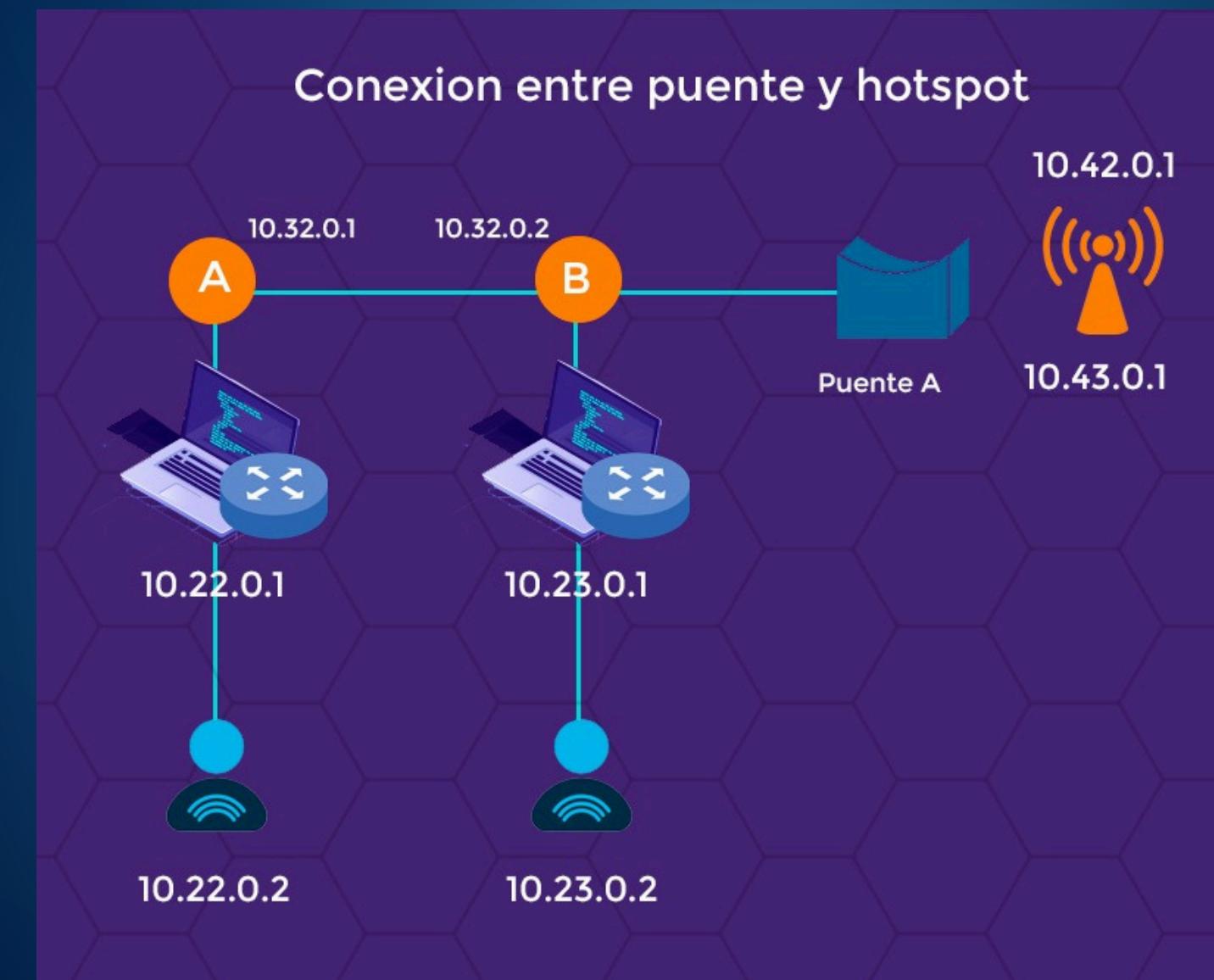


# Conexion Puente-Hotspot

El puente anterior, debe estar conectado a nuestro hotspot configurado en kali linux

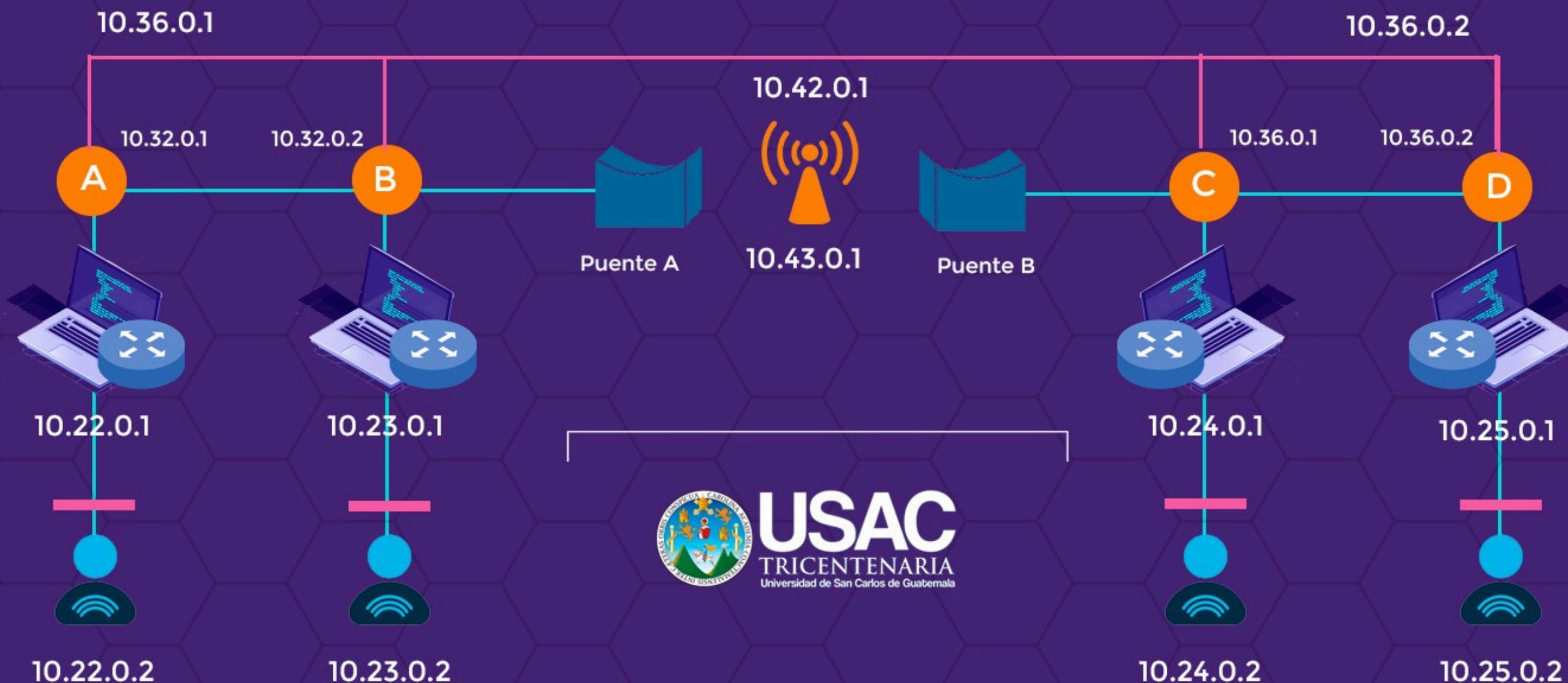


```
1 IP_PUENTE_IZQUIERDA=$1
2 IP_PUENTE_DERECHA=$2
3 # Agregando La via hacia el puente derecho cuando el objetivo es C y D
4 ip route add 10.24.0.0/27 via $IP_PUENTE_DERECHA dev ap0
5 ip route add 10.25.0.0/29 via $IP_PUENTE_DERECHA dev ap0
6
7 # Agregando La via hacia el puente izquierdo cuando el objetivo es A y B
8 ip route add 10.22.0.0/26 via $IP_PUENTE_IZQUIERDA dev ap0
9 ip route add 10.23.0.0/27 via $IP_PUENTE_IZQUIERDA dev ap0
10
```



# DISEÑO DE RED

## Diseño de red Redes de computadoras 1



**USAC**  
TRICENTENARIA  
Universidad de San Carlos de Guatemala

# CONCLUSION

## Hosts A, B, C, y D:

Cada host (A, B, C, D) está conectado en una configuración de anillo, lo que significa que cada uno se conecta directamente al siguiente y al anterior en la secuencia para formar un circuito cerrado. Esto permite que el tráfico fluya en una dirección continua (por ejemplo, de A a B, de B a C, de C a D, y de D a A).

## Conexiones entre puentes y router central:

Los puentes (Puente A y Puente B) facilitan la conexión entre los hosts y un router central con IP 10.42.0.1.

El router central está además conectado a un punto de acceso inalámbrico (10.43.0.1), que podría servir para proporcionar conectividad inalámbrica o como parte del esquema de failover.

## Conectividad y redundancia:

Este diseño de red proporciona redundancia a través de múltiples caminos entre los hosts, lo que mejora la resistencia y la disponibilidad de la red. La configuración ad-hoc de los puntos de acceso puede apoyar la adaptabilidad de la red, permitiendo conexiones flexibles entre dispositivos según sea necesario.



# GRACIAS