

CONFIGURACIÓN DE SISTEMAS OPERATIVOS EN RED

CFGs: Desarrollo de
Aplicaciones Multiplataforma
Profesor: Javier Palacios
Curso: 2020-21

CONFIGURACIÓN DE SISTEMAS OPERATIVOS EN RED

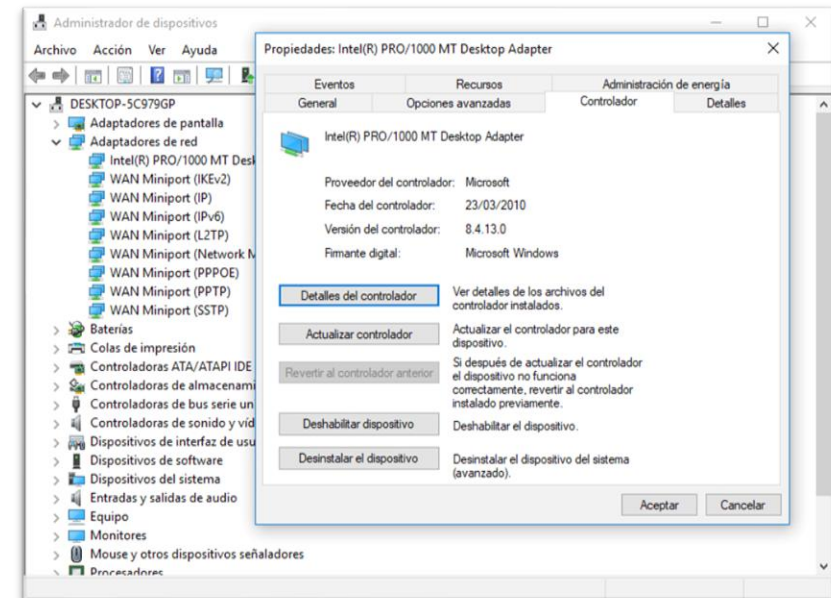


1. CONFIGURACIÓN DE RED EN CLIENTES WINDOWS
 - MEDIANTE GUI
 - MEDIANTE COMANDOS
 - FIREWALL DE WINDOWS
2. CONFIGURACIÓN DE RED EN LINUX
 - COMANDOS
 - FICHEROS DE CONFIGURACIÓN
 - MONITORIZACIÓN DE REDES
 - FIREWALL DE LINUX
3. CONFIGURACIÓN DE RED EN MÁQUINAS VIRTUALES

1. CONFIGURACIÓN DE RED EN CLIENTES WINDOWS

2.1. MEDIANTE GUI

El elemento que permite que un equipo se conecte a una red local es la **tarjeta (o adaptador) de red**. Podemos acceder a las propiedades del adaptador desde el “**Administrador de dispositivos**”, comprobar su estado, modificar su configuración o actualizar su controlador.

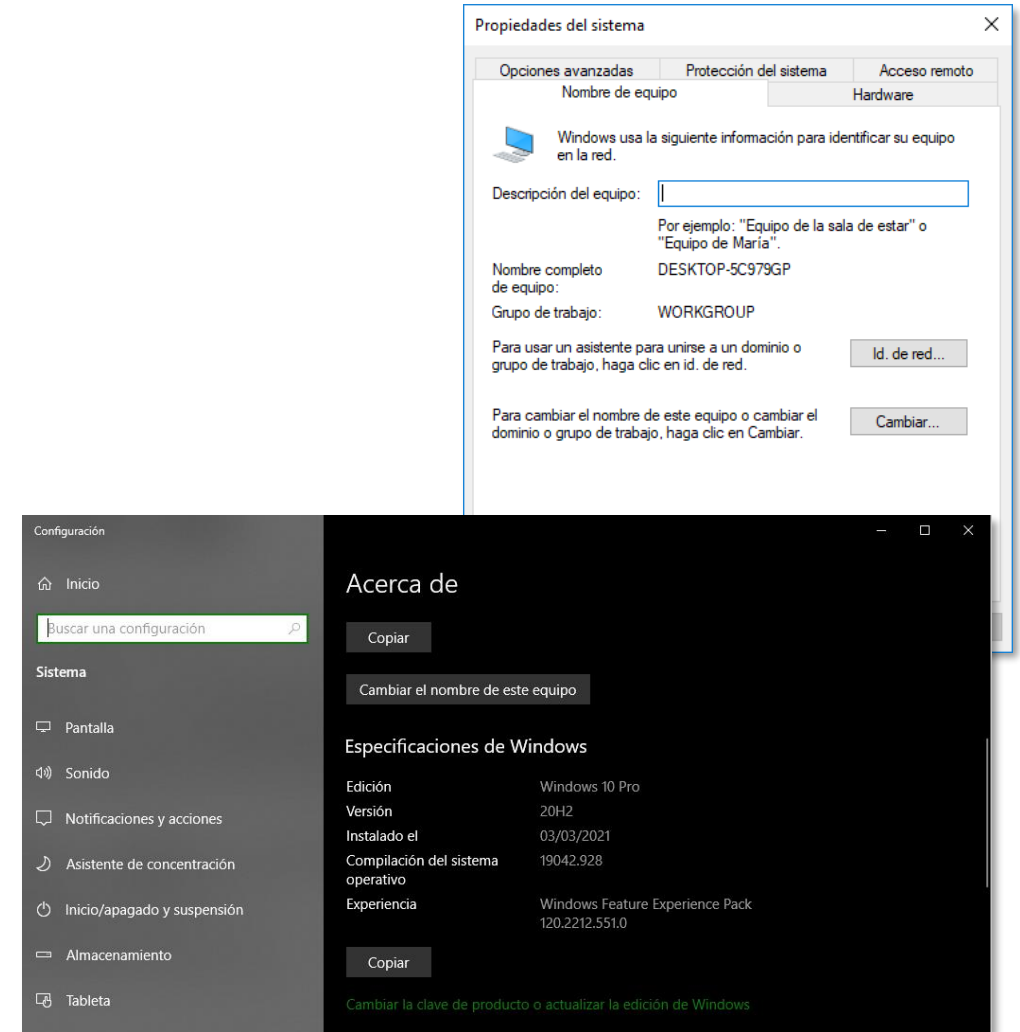


1. CONFIGURACIÓN DE RED EN CLIENTES WINDOWS

2.1. MEDIANTE GUI

Nuestro equipo se identifica en la red mediante un nombre de equipo. Para compartir y acceder a recursos compartidos por otros equipos de la red puede adscribirse a un grupo de trabajo o a un dominio. En esta unidad nos centraremos en los grupos de trabajo.

Podemos consultar y modificar los datos del equipo desde “Panel de control/Sistema y seguridad/Sistema” o en “Configuración/Sistema/Acerca de”.

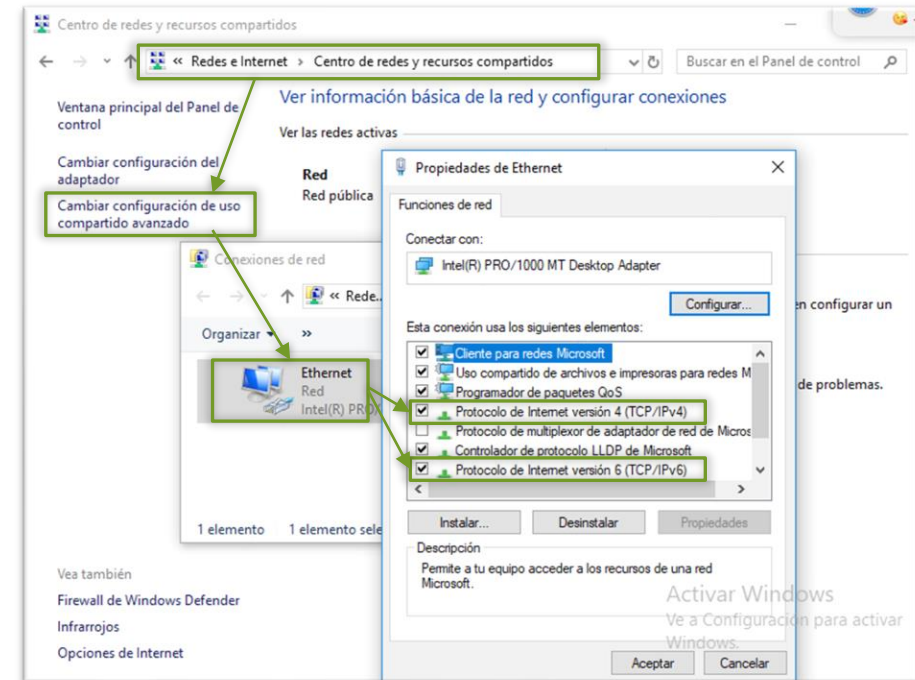


1. CONFIGURACIÓN DE RED EN CLIENTES WINDOWS

2.1. MEDIANTE GUI

La configuración de red en Windows se realiza utilizando la pila de protocolo TCP/IP mediante “Panel de Control/Redes e Internet/Cambiar configuración del adaptador” -> Propiedades del adaptador.

Nos centraremos en las propiedades del Protocolo de Internet versión 4 y 6.

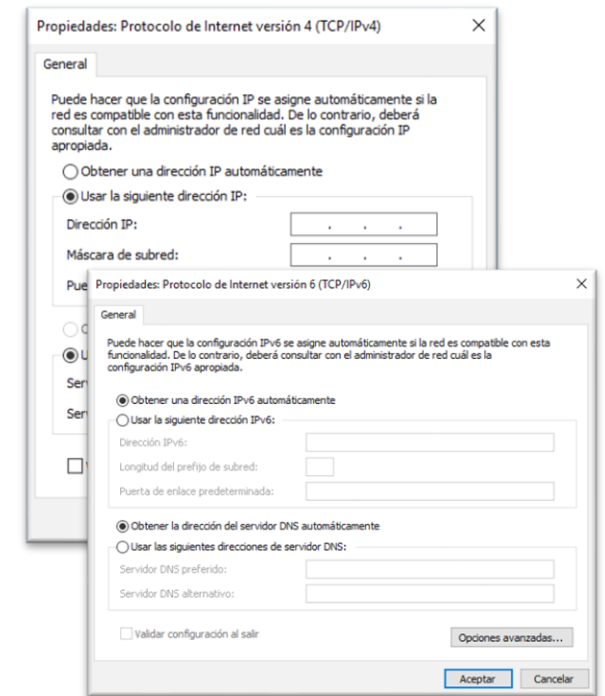


1. CONFIGURACIÓN DE RED EN CLIENTES WINDOWS

2.1. MEDIANTE GUI

Ambas opciones nos permitirán modificar las opciones de TCP/IPv4 y v6 respectivamente, las configuraciones posibles son:

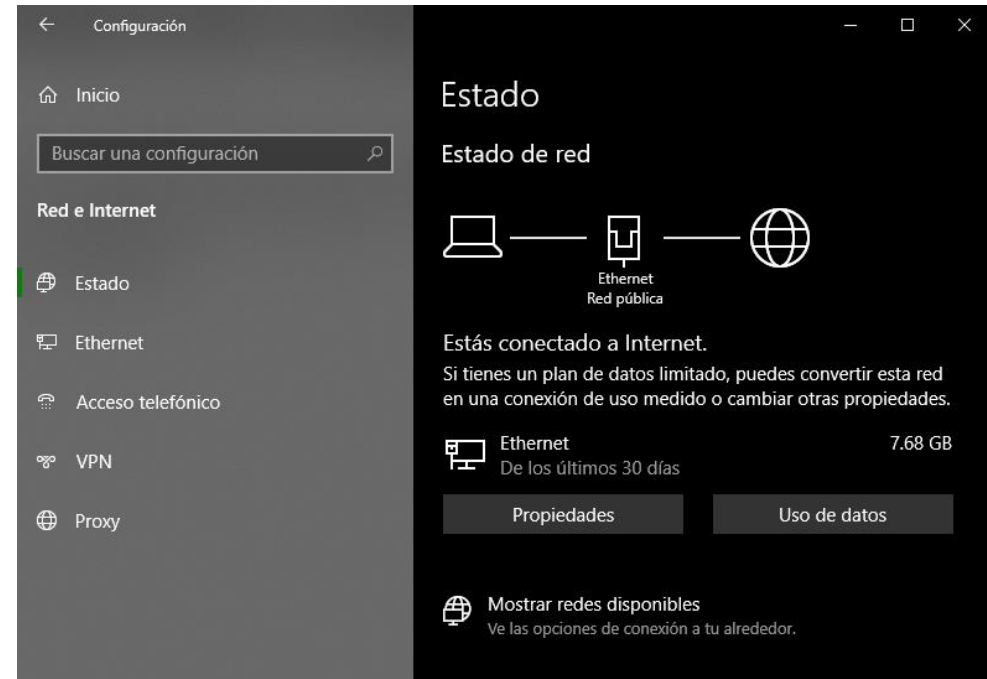
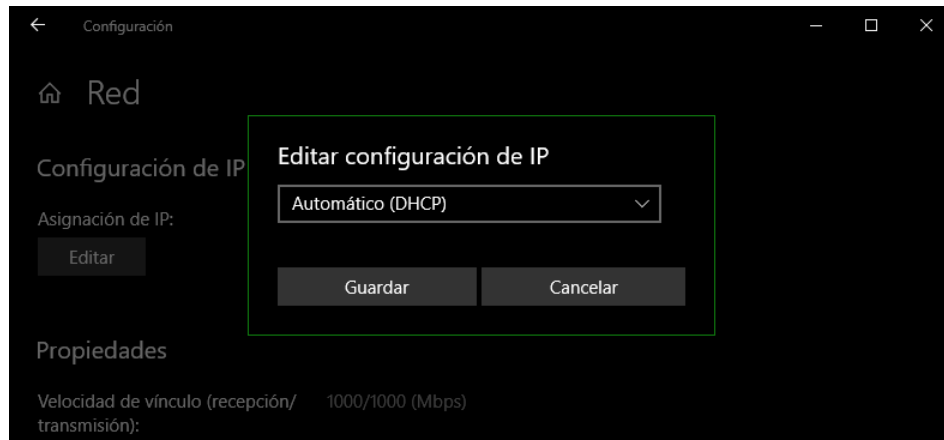
- Configuración direccionamiento:
 - **Obtener una dirección IP automáticamente:** el direccionamiento IP será proporcionado por un servidor DHCP automáticamente.
 - **Usar la siguiente dirección IP:** permite establecer manualmente la dirección IP, máscara de subred y puerta de enlace
- Configuración DNS: obtener la dirección del DNS automáticamente o establecer manualmente un servidor preferido y uno alternativo).



1. CONFIGURACIÓN DE RED EN CLIENTES WINDOWS

2.1. MEDIANTE GUI

Windows 10 también permite configurar la red mediante un asistente guiado a través de “Configuración/Redes e Internet/Estado”:



1. CONFIGURACIÓN DE RED EN CLIENTES WINDOWS

2.2. MEDIANTE COMANDOS



- Para consultar la configuración de red desde la consola (CMD) podemos utilizar el comando **“ipconfig”**, la opción /all proporciona información extendida.
- Para establecer

```
C:\Users\javi>ipconfig /all

Configuración IP de Windows

Nombre de host. . . . . : DESKTOP-5C979GP
Sufijo DNS principal . . . . . :
Tipo de nodo. . . . . : híbrido
Enrutamiento IP habilitado. . . : no
Proxy WINS habilitado . . . . . : no
Lista de búsqueda de sufijos DNS: home

Adaptador de Ethernet Ethernet:

Sufijo DNS específico para la conexión. . : home
Descripción . . . . . : Intel(R) PRO/1000 MT Desktop Adapter
Dirección física. . . . . : 08-00-27-37-E6-83
DHCP habilitado . . . . . : sí
Configuración automática habilitada . . . : sí
Vínculo: dirección IPv6 local. . . : fe80::f587:5cc2:e605:3735%4(Preferido)
Dirección IPv4. . . . . : 10.0.2.15(Preferido)
Máscara de subred . . . . . : 255.255.255.0
Concesión obtenida. . . . . : miércoles, 7 de noviembre de 2018 14:29:13
La concesión expira . . . . . : viernes, 9 de noviembre de 2018 14:03:05
Puerta de enlace predeterminada . . . . . : 10.0.2.2
Servidor DHCP . . . . . : 10.0.2.2
IAID DHCPv6 . . . . . : 50855975
DUID de cliente DHCPv6. . . . . : 00-01-00-01-23-4F-AF-D2-08-00-27-37-E6-83
Servidores DNS. . . . . : 192.168.1.1
```


1. CONFIGURACIÓN DE RED EN CLIENTES WINDOWS

2.2. MEDIANTE COMANDOS



- También es posible consultar la información de interfaces mediante la herramienta net

- `netsh interface ipv4 show config`

- Así como modificarla:

- `netsh interface ipv4 set address name="nombre" static <IP> <Máscara> <Gateway>`

- `netsh interface ipv4 set dns name="nombre" static <IP_DNS>`

- También es posible exportar e importar configuraciones:

- `C:\netsh dump > archivo.cmp`

- `C:\netsh exec archivo.cmp`

```
PS C:\Users\marta> netsh interface ipv4 show config

Configuración para la interfaz "Ethernet"
DHCP habilitado:                Sí
Dirección IP:                   192.168.1.128
Prefijo de subred:              192.168.1.0/24 (máscara 255.255.255.0)
Puerta de enlace predeterminada: 192.168.1.1
Métrica de puerta de enlace:    0
Métrica de interfaz:            25
Servidores DNS configurados a través de DHCP: 192.168.1.1
Registrar con el sufijo:        Solo el principal
Servidores WINS configurados a través de DHCP: ninguno
```

2.2. MEDIANTE COMANDOS



Otras aplicaciones de consola útiles para el trabajo en red son:

- **Ping:** Permite verificar la conectividad con un equipo de red (ip o dominio). Utiliza el protocolo ICMP.
- **Tracert:** Permite determinar el camino seguido por un paquete para llegar a un equipo de red remoto (ip o dominio).
- **Nslookup:** Permite la traducción directa e inversa de direcciones IP a nombres de dominio.
- **Netstat:** Muestra diversas estadísticas de red. Por defecto, muestra un listado de las conexiones activas.

2.2. MEDIANTE COMANDOS



Algunos comandos útiles para configuración de red en PowerShell:

- **Get-NetAdapter**: información sobre el adaptador de red (extendida con `| format-list *`).
 - **Disable-NetAdapter -Name "nombreinterfaz"**
 - **Enable-NetAdapter -Name "nombreinterfaz"**
- **Get-NetIPConfiguration**: similar a la información proporcionada por Ipconfig.
- **Get-NetIPAddress**: información extendida sobre el direccionamiento IP.
 - **New-NetIPAddress -InterfaceAlias "nombreinterfaz" -IPAddress "x.x.x.x" -PrefixLength "mascara_CDIRE"**
 - (New-NetIPAddress establece una IP estática para una interfaz sin IP, Set-NetIPAddress la modifica)
 - **Remove-NetIPAddress -InterfaceAlias "nombreinterfaz"**
 - **Set-DnsClientServerAddress -InterfaceAlias "nombreinterfaz" -ServerAddresses "y.y.y.y", "z.z.z.z"**
- **Get-NetRoute**: proporciona información sobre la tabla de enrutamiento.
 - **Set-NetRoute -DestinationPrefix "x.x.x.x/mascara_CDIRE"**
- **Test-NetConnection -ComputerName "sitio"**: permite comprobar la conectividad con un equipo.

Alternativa, pasar interfaz al comando:

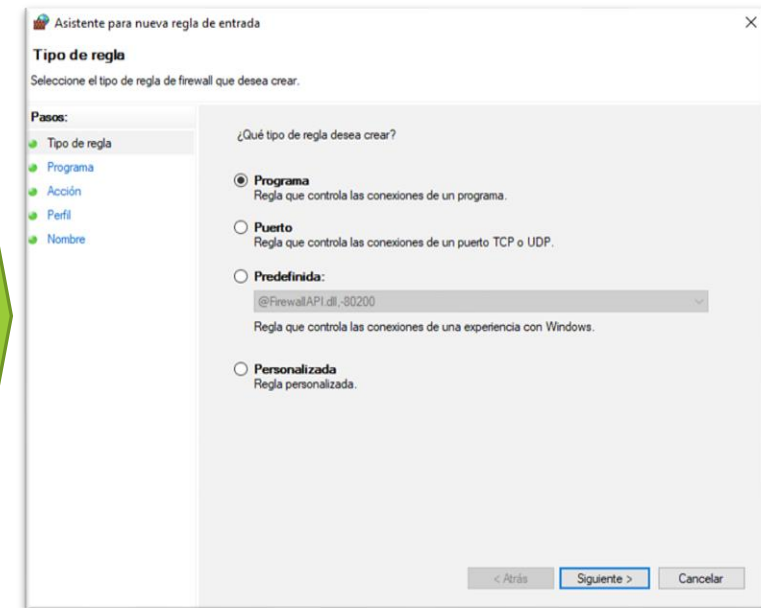
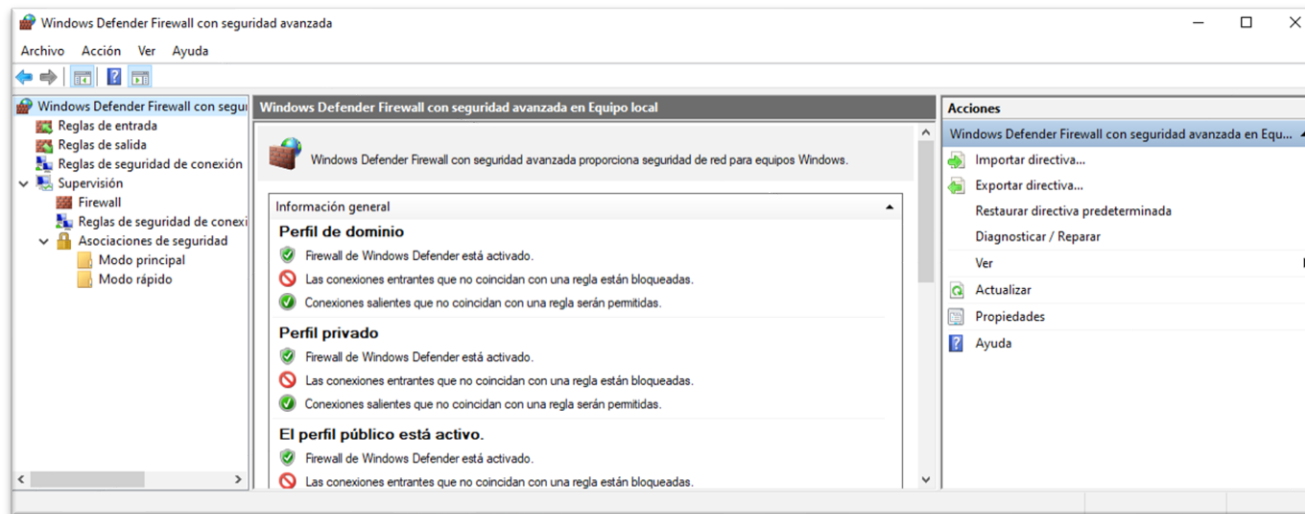
`Get-NetAdapter -Name "nombreinterfaz" | New-NetIPAddress -IPAddress "x.x.x.x" -PrefixLength "x"`

1. CONFIGURACIÓN DE RED EN CLIENTES WINDOWS

3 FIREWALL DE WINDOWS



Windows Defender Firewall permite establecer reglas de entrada y salida por puerto y/o aplicación, también proporciona reglas predefinidas, así como la utilizada de importar, exportar directivas.



2. CONFIGURACIÓN DE RED EN LINUX

Como en otros aspectos, la mayor parte de la configuración de los sistemas linux se realiza bien mediante ficheros de texto y/o comandos (que finalmente modifican estos ficheros de configuración).

Todos los sistemas basados en Unix son compatibles de forma nativa con la pila de protocolos TCP/IP. Todos los sistemas actuales basados en Unix o Linux son compatibles con IPv4 y con IPv6, tanto en sus comandos como en sus ficheros de configuración.

2. CONFIGURACIÓN DE RED EN LINUX

2.1. COMANDOS

Comandos de red del paquete **net-tools** (más antiguos):

- **ifconfig** – permite consultar y/o modificar diversos parámetros relacionados con el funcionamiento de las interfaces de red.
 - Sintaxis:
 - `ifconfig interfaz [dirección [parámetros]]`
 - Algunas opciones:
 - `up` habilita la interfaz.
 - `down` deshabilita la interfaz.
 - `netmask` asigna una mascara a la interfaz.
 - `mtu` establece la unidad máxima de transferencia (1500B máximo permitido en Ethernet)
 - `arp` habilita o deshabilita (-arp) el protocolo de resolución de direcciones ARP.
 - `promisc` habilita o deshabilita (-promisc) el modo promiscuo en una interfaz de red.
 - `allmulti` habilita o deshabilita (-allmulti) la recepción de paquetes multicast.

2. CONFIGURACIÓN DE RED EN LINUX

2.1. COMANDOS

- **route** – muestra y/o modifica la tabla cache enrutamiento del sistema. Si no se especifica entrada a consultar o modificar retorna toda la tabla. El formato de la tabla es *destino, pasarela, genmask, incic, métric, ref, uso, interfaz*.
 - Sintaxis:
 - `route [opciones [entrada_tabla]]`
 - Algunas opciones:
 - `-n` muestra direcciones IP en lugar de nombres de host.
 - `add` añade una nueva entrada a la tabla de enrutamiento. Formato completo:
 - `route add -net <IP> netmask <máscara_de red> gw <puerta_de_enlace> dev <interfaz>`
 - `del` elimina una entrada de la tabla de enrutamiento. Formato completo:
 - `route del -net <IP> netmask <máscara_de red> gw <puerta_de_enlace> dev <interfaz>`

2. CONFIGURACIÓN DE RED EN LINUX

2.1. COMANDOS

- **arp** – muestra y/o modifica la tabla cache ARP del sistema. Si no se especifica entrada a consultar o modificar retorna toda la tabla.
 - Sintaxis:
 - `arp [opciones [entrada_tabla]]`
 - Algunas opciones:
 - `-n` muestra direcciones IP en lugar de nombres de host.
 - `-s <IP> <MAC>` añade una nueva entrada a la base de datos ARP
 - `-d <IP>` elimina una entrada de la base de datos ARP.

2. CONFIGURACIÓN DE RED EN LINUX

2.1. COMANDOS

Comandos de red del paquete **iproute2** (recomendado):

- **ip** – es un conjunto de utilidades de red más moderno, cuyo comando principal es ip.
 - Sintaxis:
 - `ip [opciones] objeto [subcomando]`
 - Algunas opciones:
 - `-4` para IPv4.
 - `-6` para IPv6.
 - `-s` muestra estadísticas.
 - `-d` muestra una vista detallada.
 - `-h` versión “human-readable” de magnitudes.
 - `-j` salida formato json (`-p` versión “pretty”).
 - Algunos objetos:
 - `link` hace referencia al nivel de enlace de la interfaz.
 - `addr` hace referencia las conexiones e interfaces red.
 - `route` hace referencia a la configuración de enrutamiento.
 - `neigh` hace referencia a la configuración de ARP.

2. CONFIGURACIÓN DE RED EN LINUX

2.1. COMANDOS

- Algunos subcomandos por objeto:

- Objeto link

- `ip link show [INTERFAZ]`
 - `ip link set INTERFAZ { up | down }`
 - `ip link set INTERFAZ arp { on | off }`
 - `ip link set INTERFAZ promisc { on | off }`
 - `ip link set INTERFAZ allmulticast { on | off }`
 - `ip link set INTERFAZ dynamic { on | off }`
 - `ip link set INTERFAZ multicast { on | off }`
 - `ip link set INTERFAZ name NOMBRE`
 - `ip link set INTERFAZ address LLADDR`
 - `ip link set INTERFAZ broadcast LLADDR`
 - `ip link set INTERFAZ mtu MTU`
 - `ip link set INTERFAZ alias NAME`
 - `ip link set INTERFAZ vf NUM [mac LLADDR]`
`[vlan VLANID [qos VLAN-QOS]] [rate TXRATE]`

2. CONFIGURACIÓN DE RED EN LINUX

2.1. COMANDOS

- Objeto **addr**

- **ip addr { add | del } DIRECCIÓN_IP dev INTERFAZ**
- **ip addr { add | del } broadcast DIRECCIÓN_IP dev INTERFAZ**
- **ip addr { add | del } anycast DIRECCIÓN_IP dev INTERFAZ**
- **ip addr { add | del } label ETIQUETA dev INTERFAZ**
- **ip addr { add | del } scope [host | link | global | NUMBER] dev INTERFAZ**
- **ip addr { show | flush } [dev INTERFAZ]**
 - [**scope** [host | link | global | NUMBER]]
 - [**to** PREFIJO]
 - [permanent | dynamic | secondary | primary | tentative | deprecated]
 - [**label** PATTERN]

2. CONFIGURACIÓN DE RED EN LINUX

2.1. COMANDOS

- Objeto route

- `ip route { list | flush } [[root PREFIX] [match PREFIX] [exact PREFIX] [table TABLE_ID] [proto RTPROTO] [type TYPE] [scope SCOPE]]`
- `ip route get ADDRESS [from ADDRESS iif STRING] [oif STRING] [tos TOS]`
- `ip route { add | del | change | append | replace | monitor } ROUTE`
 - `ROUTE := NODE_SPEC [INFO_SPEC]`
 - `NODE_SPEC := [TYPE] PREFIX [tos TOS] [table TABLE_ID] [proto RTPROTO] [scope SCOPE] [metric METRIC]`
 - `INFO_SPEC := NH OPTIONS FLAGS [nexthop NH] ...`
 - `NH := [via ADDRESS] [dev STRING] [weight NUMBER] NHFLAGS`
 - `OPTIONS := FLAGS [mtu NUMBER] [advmss NUMBER] [rtt TIME] [rttvar TIME] [window NUMBER] [cwnd NUMBER] [initcwnd NUMBER] [ssthresh REALM] [realms REALM] [rto_min TIME] [initrwnd NUMBER]`
 - ...

2. CONFIGURACIÓN DE RED EN LINUX

2.1. COMANDOS

- Ejemplos utilización ip route:
 - Agregar un camino hacia la red 10.0.0/24 por la puerta de enlace 192.168.0.2
 - `ip route add 10.0.0.0/24 via 192.168.0.2`
 - cambiar por una ruta directa por el dispositivo disp
 - `ip route chg 10.0.0.0/24 via 192.168.0.2 dev disp`
 - agrega un multicamino por defecto dividiendo la carga entre p1 y p2
 - `ip route add default scope global nexthop dev p1 nexthop dev p2`
 - si quisieramos aplicar NAT antes de transmitir
 - `ip route add nat 172.16.0.1 via 192.168.0.3`
 - encontrar una salida para los paquetes de 192.168.0.2
 - `ip route get 192.168.0.2`
 - encontrar una ruta para enviar paquetes que llegan de 10.0.0.1 con destino 192.168.0.3
 - `ip route get 10.0.0.1 from 192.168.0.3 iif eth0`
 - para realizar Balanceo de carga asignando pesos a cada puerta de enlace:
 - `ip route add default scope global nexthop via 192.168.1.2 dev eth1 weight 1 nexthop via 172.16.1.2 dev eth2 weight 2`

2. CONFIGURACIÓN DE RED EN LINUX

2.1. COMANDOS

- Objeto rule

- `ip rule [list | add | del | flush] SELECTOR ACTION`

- `SELECTOR := [from PREFIX] [to PREFIX] [tos TOS] [fwmark FWMARK[/MASK]] [dev STRING] [pref NUMBER]`

- `ACTION := [table TABLE_ID] [nat ADDRESS] [prohibit | reject | unreachable] [realms [SRCREALM/]DSTREALM]TABLE_ID := [local | main | default | NUMBER]`

- Ejemplo: crear una regla NAT

- `ip rule add nat 205.254.211.32 from 192.168.1.25`

- Objeto neigh

- `ip neigh { add | del | change | replace } { ADDR [lladdr LLADDR] [nud { permanent | noarp | stale | reachable }] | proxy ADDR } [dev DEV]`

- `ip neigh { show | flush } [to PREFIX] [dev DEV] [nud STATE]`

- Ejemplo: en la interfaz eth3 la máquina 192.168.1.25 tendrá la mac 11:22:33:44:55:66

- `ip neigh add 192.168.2.25 lladdr 08:00:27:a9:f3:7d dev eth3 nud permanent`

2. CONFIGURACIÓN DE RED EN LINUX

2.1. COMANDOS

Otras herramientas:

- **ping**– Permite comprobar la conectividad con un equipo a nivel de red con un equipo destino por su nombre o dirección IP mediante el protocolo ICMP.

- Sintaxis:

- `ping [opciones] destino`

- Algunas opciones:

- `-n<N>` envía N solicitudes.
 - `-t` envía solicitudes hasta que se detenga manualmente el proceso.

2. CONFIGURACIÓN DE RED EN LINUX

2.1. COMANDOS

- **traceroute**– Muestra la ruta de un paquete hasta llegar a su destino.
 - Sintaxis:
 - `traceroute [opciones] destino`
- **whois**– permite obtener información sobre un determinado dominio y su registrador.
 - Sintaxis:
 - `whois [opciones] dominio`
- **netstat**– identifica las conexiones de red abiertas, proporciona estadísticas e información sobre puertos TCP y UDP.

2. CONFIGURACIÓN DE RED EN LINUX

2.1. COMANDOS

- **ss** – muestra información de las conexiones de un equipo, alternativa a netstat.
 - `-a` muestra todos los sockets
 - `-t` tcp
 - `-u` udp
 - `-4` conexiones IPv4
 - `-6` conexiones IPv6
 - `-p` muestra el PID del proceso que utiliza el socket
 - `-l` solo conexiones que escuchan
 - `-s` resumen de estadísticas
- **nslookup** – Se utiliza para realizar traducciones de IP a nombre de dominio y viceversa, o testar un DNS.
 - Sintaxis:
 - `nslookup [ip | nombre_dominio]`
- **dig** – más avanzado que nslookup, permite realizar búsquedas avanzadas en registros DNS.

2.2. FICHEROS DE CONFIGURACIÓN

Configuración de interfaces en Debian:

- Archivo `/etc/network/interfaces`. Su estructura es:

- Ejemplo configuración estática:

```
iface eth0 inet static
    address 192.161.1.25
    netmask 255.255.255.0
    broadcast 192.168.1.255
    gateway 192.168.1.1
```

- Ejemplo configuración estática:

```
auto eth0
iface eth0 inet dhcp
```

- Ejemplo configuración estática:

```
auto lo eth0
iface lo inet loopback
```

2.2. FICHEROS DE CONFIGURACIÓN

Configuración de definición de servidores DNS en Debian (no utilizar en Ubuntu ≥ 18):

- Archivo `/etc/resolv.conf`. Su estructura es:

```
nameserver 127.0.0.52
options edns0
search MiCentro.edu
```

2. CONFIGURACIÓN DE RED EN LINUX

2.2. FICHEROS DE CONFIGURACIÓN

Configuración de interfaces en Ubuntu 20.04:

- Archivo `/etc/netplan/01-netcfg.yaml`. Su estructura es:

- Ejemplo configuración estática:

```
network:
  version: 2
  renderer: networkd
  ethernets:
    enp0s3: network interface
      dhcp4: no
      dhcp6: no
      addresses: [192.161.1.25/24]
      gateway4: y.y.y.y
      nameservers:
        addresses [8.8.8.8, 1.1.1.1]
```

- Si el archivo no existe puede generarse manualmente o con: `sudo netplan generate`
- Para aplicar los cambios configurados en el fichero: `sudo netplan apply`
- Antes de aplicar una configuración, puede probarse con: `sudo netplan try`

- Ejemplo configuración estática:

```
network:
  version: 2
  renderer: networkd
  ethernets:
    enp0s3: network interface
      dhcp4: yes
      dhcp6: yes
```

2.2. FICHEROS DE CONFIGURACIÓN

Resolución de nombres local:

- Archivo `/etc/hosts`. Para nombres conocidos. Ejemplo configuración:

```
127.0.0.1    localhost
127.0.1.1    javi-virtualvox
::1          ip6-localhost
::1          ip6-loopback
fe00::0      ip6-localnet
ff00::0      ip6-mcastprefix
ff02::1      ip6-allnodes
ff02::2      ip6-allrouters
```

- Archivo `/etc/hostname`. Para la propia máquina. Ejemplo configuración:

```
Javi-VirtualBox
```

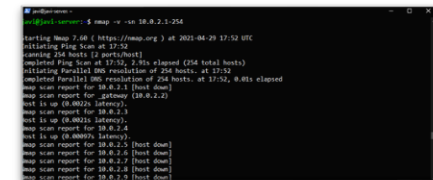

2.3. MONITORIZACIÓN DE REDES

- **netstat** – Muestra estadísticas de red
 - Sintaxis:
 - `netstat [opciones]`
 - Algunas opciones:
 - `-a` Estado puertos TCP y UDP (con `-at` o `-au` filtra para cada protocolo).
 - `-l` solo puertos activos.
 - `-s` Muestra estadísticas por protocolo.
 - `-c` Se mantiene monitorizando continuamente.
 - `-i` Muestra estadísticas de las interfaces. (con `-ie` muestra información extendida)
- **ip** – Con la opción `-s` muestra estadísticas para los distintos objetos (addr, link...)
- **tcpdump** – Capturador/analizador de tráfico de red en línea de comandos.

2. CONFIGURACIÓN DE RED EN LINUX

2.3. MONITORIZACIÓN DE REDES

- **nmap** – es un programa multiplataforma de código abierto que sirve para efectuar rastreo de puertos, detección de equipos, servicios y sistemas operativos, detección de vulnerabilidades y otras aplicaciones.
- **iptraf** – Es una herramienta de monitorización de tráfico de red en tiempo real. Funciona en modo interactivo.
- **wireshark** – es un analizador de protocolos utilizado para realizar análisis y solucionar problemas en redes de comunicaciones. Esta disponible en versión CLI (**tshark**) y GUI para los principales sistemas operativos.



```
nmap -sT 10.0.2.1-254
Starting Nmap 7.60 (https://nmap.org) at 2023-04-20 17:52 UTC
Initiating Ping Scan at 17:52, 2.81s elapsed (254 total hosts)
Initiating Parallel DNS resolution of 254 hosts. at 17:52
Initiating Parallel OS resolution of 254 hosts. at 17:52, 0.01s elapsed
nmap scan report for 10.0.2.1 [host down]
nmap scan report for 10.0.2.2 [host down]
nmap scan report for 10.0.2.3 [host down]
nmap scan report for 10.0.2.4 [host down]
nmap scan report for 10.0.2.5 [host down]
nmap scan report for 10.0.2.6 [host down]
nmap scan report for 10.0.2.7 [host down]
nmap scan report for 10.0.2.8 [host down]
nmap scan report for 10.0.2.9 [host down]
```

```

iptraf-ng 1.1.4
Statistics for eth0:

Total          Total          Incoming    Incoming    Outgoing    Outgoing
Packets        Bytes          Packets      Bytes      Packets      Bytes
Total:         6587          512126      3388       155266     3279       376860
IPv4:          6587          512584      3388       155724     3279       376860
IPv6:          0              0            0           0          0           0
TCP:           6587          512584      3388       155724     3279       376860
UDP:           0              0            0           0          0           0
ICMP:          0              0            0           0          0           0
Other IP:      0              0            0           0          0           0
Non-IP:        0              0            0           0          0           0

Total rates:  256.30 kbps      Broadcast packets: 0
              454 pps         Broadcast bytes:    0

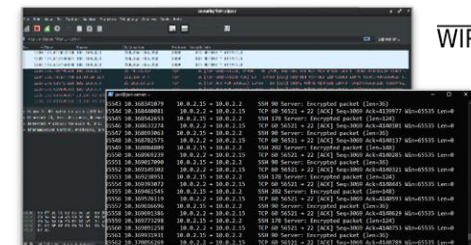
Incoming rates: 83.66 kbps
                226 pps

Outgoing rates: 172.62 kbps
                226 pps

IP checksum errors: 0

Elapsed time: 0:00
X-Exit

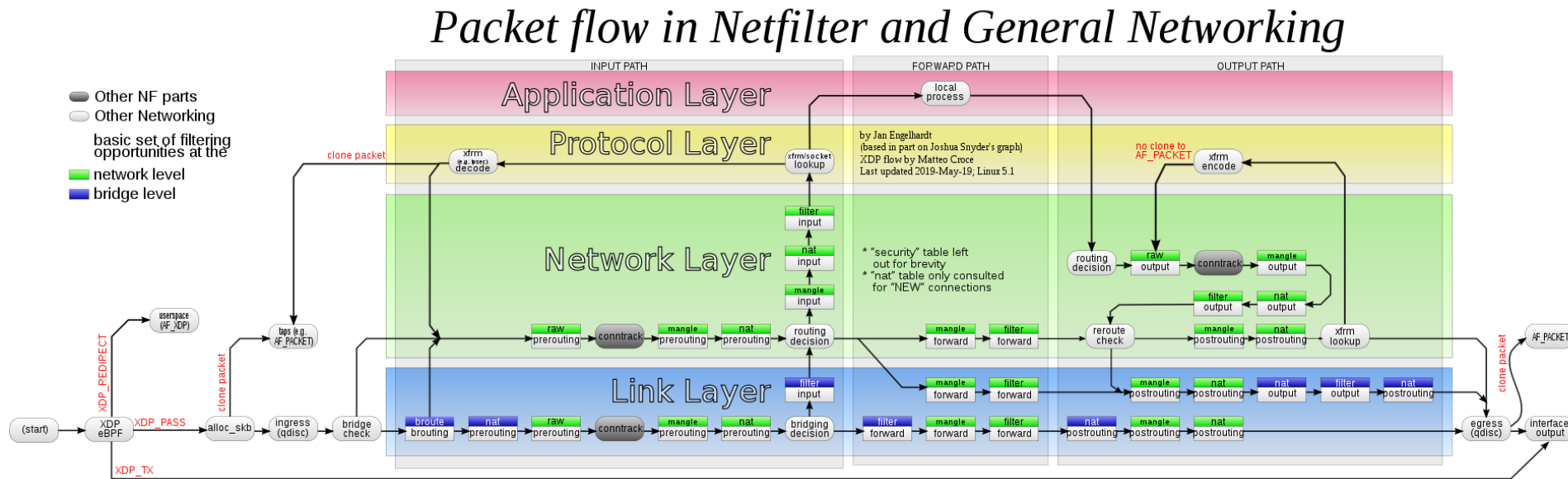
```



2. CONFIGURACIÓN DE RED EN LINUX

2.4. FIREWALL DE LINUX

Linux proporciona un firewall integrado denominado, para configurarlo proporciona la utilidad Iptables.



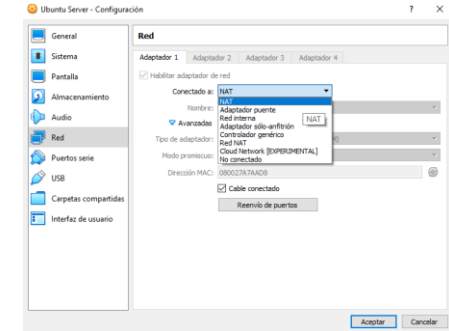
2. CONFIGURACIÓN DE RED EN LINUX

2.4. FIREWALL DE LINUX

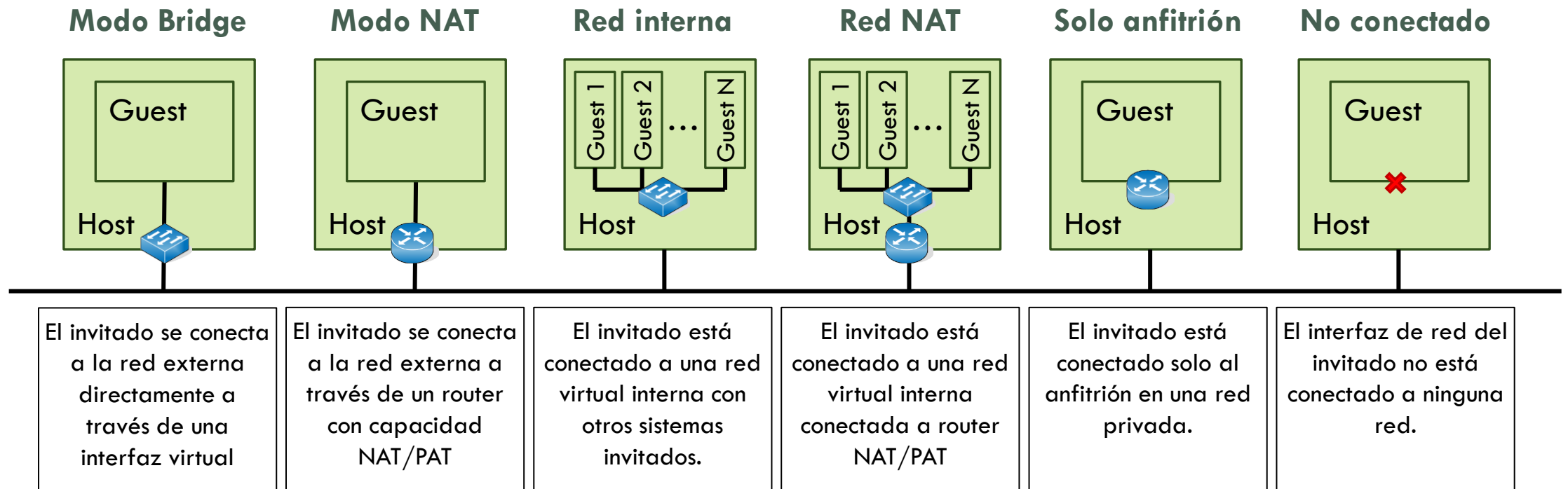
ufw (Uncomplicated Firewall): es un cortafuegos desarrollado por Ubuntu utiliza iptables, pero implementa una interfaz basada en comandos mucho más sencillos.

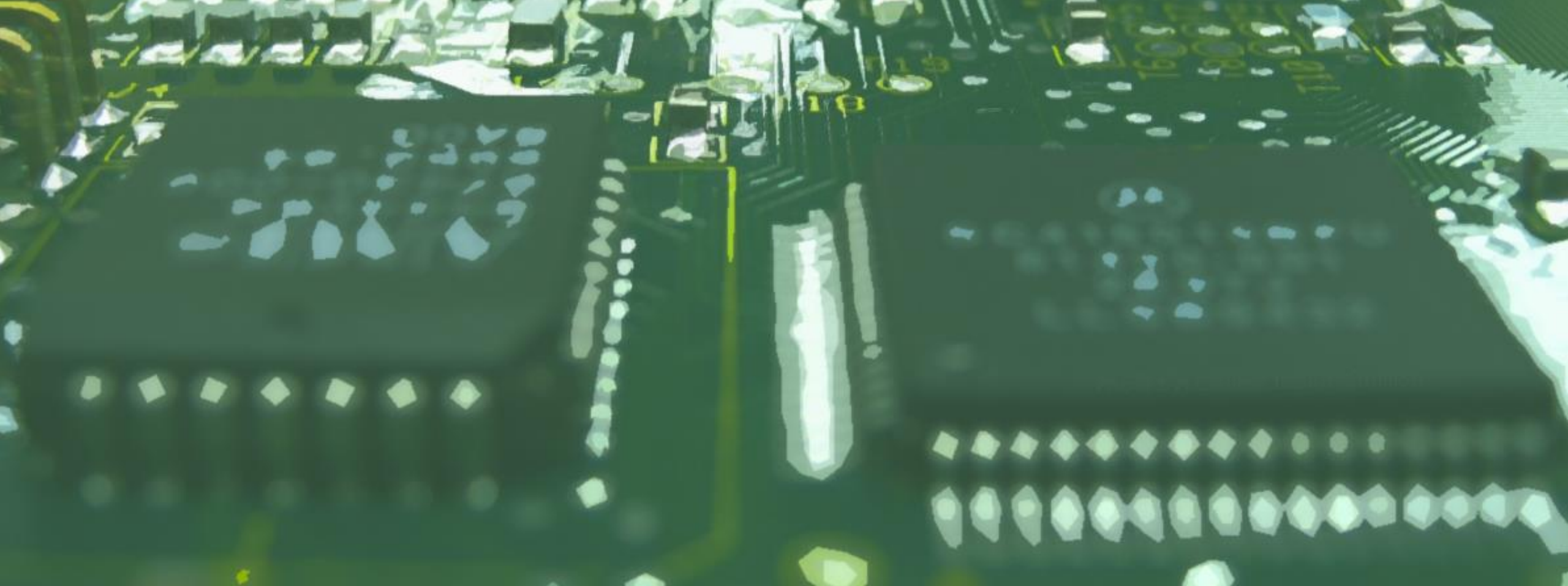
- `ufw [enable | disable]` - habilita o deshabilita el firewall.
- `ufw status` - proporciona información (verbose información ampliada).
- `ufw status numbered` - muestra las reglas numeradas.
- `ufw default allow [incoming|outcoming]` - permite el trafico entrante o saliente por defecto.
- `ufw default deny [incoming|outcoming]` - bloquea el trafico entrante o saliente por defecto.
- `ufw [allow|deny] [servicio]` - permite o bloquea el trafico de un servicio.
- `ufw [allow|deny] [puerto[/protocolo]]` - permite o bloquea el trafico entrante o saliente en un puerto.
- `ufw [allow|deny] from <dirección_IP[/mascaraCDIR]> [to any port [puerto[/protocolo]]]` - permite o bloquea el trafico procedente de una IP o red, general o por puerto.
- `ufw delete <número_regla>` - muestra las reglas numeradas.
- `ufw reset` - muestra las reglas numeradas.

3. CONFIGURACIÓN DE RED EN MÁQUINAS VIRTUALES



Los hipervisores suelen incluir sistemas de virtualización de redes e interfaces, algunos modos de funcionamiento de las interfaces virtuales son:





GRACIAS

CFGs: Desarrollo de
Aplicaciones Multiplataforma
Profesor: Javier Palacios
Curso: 2020-21