4. Configuración de la red

- Direcciones IPv4
- Interfaces de red
- Tabla de rutas
- Resolución de direcciones: arp
- Utilidades: ping, traceroute, netstat, nslookup
- Configuración del servicio dhcp
- Firewall
- Direcciones IPv6

- Práctica 5: Configuración de una intranet con servidor Linux

4. Configuración de red: La dirección IP (IPV4)

- La dirección de red IPv4 es un entero de 32 bits que identifica cada dispositivo en una red TCP/IP
- Los sistemas pueden ser accedidos de tres formas diferentes.
  - Los sistemas individuales se direccionan mediante una dirección de host, o dirección unicast.
  - Puede accederse simultáneamente a un grupo de sistemas con una dirección multicast, como 224.0.0.9: los routers en el camino entre el origen y el destino reconocen la dirección especial y enrutan copias del paquete a cada miembro del grupo
  - Todos los sistemas de una misma red pueden accederse simultáneamente mediante una dirección broadcast, como 172.16.255.255
- Una direccion IP con todos los bits de host a cero identifica a la red misma (172.16.0.0). Estas direcciones se usan en las tablas de rutas.
- Las direcciones con un primer byte mayor que 223 no pueden asignarse a una red física, están reservadas. La dirección 0.0.0.0 designa la ruta por defecto y la dirección 127.0.0.0 es la dirección loopback, usada para referirse al host local mediante una dirección IP.

4. Configuración de red: Estructura de una dirección IP

- Una dirección IP consta de una parte de red y de una parte de host. El número de bits usado para definir la red es variable, y la longitud de ese prefijo se determina por una máscara de bits.
- La máscara funciona de la forma siguiente: si el bit está en la máscara, el bit equivalente de la dirección está en la parte de red. Si no, en la parte de host. Por ejemplo, la dirección 172.22.12.4 con la máscara 255.255.255.0 se descompone en la dirección de host 4 en la red 172.22.12.
- Para indicar la longitud de la máscara se puede escribir 172.22.12.4/24, donde el 24 indica el número de unos en la máscara de red
- Las organizaciones compran bloques de direcciones de los proveedores de servicios de internet, p.e. podrían comprar 192.168.16.0/20, un bloque de 12 bits con 4096 direcciones desde 192.168.16.0 a 192.168.31.255. Cada uno de estos bloques de direcciones aparece ante el mundo como una única dirección de red: los routers externos tienen una ruta al bloque 192.168.16.0/20, aunque la compañía tenga varias redes físicamente separadas dentro del bloque de direcciones, subdivididas a su vez con otras máscaras.

4. Configuración de red: Subredes y la máscara natural

- Dentro de cada compañía, las redes se subdividen tomando bits de la parte de host como nuevos bits de red. Cada una de estas divisiones es gestionada por un nuevo router, y en general cada red física tendrá su propia dirección
- Originalmente el espacio de direcciones se dividía en clases (clase A, clase B, clase C). Ya no se usan, pero las misma reglas que se usaban determinan la máscara por defecto, o “máscara natural”. Las reglas son:
    - Si el primer bit de la dirección IP es cero, la máscara por defecto es de 8 bits (antigua clase A)
    - Si los dos primeros bits son 10, 16 bits (clase B)
    - Si los tres primeros bits son 110, 24 bits (clase C)
    - Si los cuatro primeros bits son 1110, es una dirección multicast. Estas direcciones se usan para construir grupos de ordenadores que comparten una aplicación, como videoconferencia. La máscara tiene 32 bits
- Actualmente se pueden asociar rangos de direcciones contiguos a la misma red sin que formen una clase C. Para evitar tener que introducir una entrada separada para cada clase C en la tabla de rutas se usa Classless Inter-Domain Routing (CIDR)

4.L Nombre del interfaz de red en RHEL8

Scheme	Description	Example
1	Device names incorporate firmware or BIOS-provided index numbers for onboard devices. If this information is not available or applicable, **udev** uses scheme 2.	**eno1**
2	Device names incorporate firmware or BIOS-provided PCI Express (PCIe) hot plug slot index numbers. If this information is not available or applicable, **udev** uses scheme 3.	**ens1**
3	Device names incorporate the physical location of the connector of the hardware. If this information is not available or applicable, **udev** uses scheme 5.	**enp2s0**
4	Device names incorporate the MAC address. Red Hat Enterprise Linux does not use this scheme by default, but administrators can optionally use it.	**enx525400d5e0fb**
5	The traditional unpredictable kernel naming scheme. If **udev** cannot apply any of the other schemes, the device manager uses this scheme.	**eth0**

4.L Configuración de un interfaz

- La mayoría de las operaciones de configuración de un interfaz de red se realizan mediante la orden nmcli en RHEL8

nmcli command	abbreviation
nmcli general status	nmcli g
nmcli general logging	nmcli g log
nmcli connection show	nmcli con show or nmcli c
nmcli connection show --active	nmcli con show -a or nmcli c -a
nmcli device status	nmcli dev or nmcli d
nmcli device show	nmcli dev show or nmcli d show

```
[[root@localhost ~]# nmcli device status
DEVICE      TYPE      STATE      CONNECTION
enp1s0      ethernet  connected  enp1s0
virbr0      bridge    connected  virbr0
lo          loopback  unmanaged  --
virbr0-nic  tun       unmanaged  --
```

4.L Configuración permanente de un interfaz de red, con parámetros estáticos, en RHEL8

- Para configurar un interfaz de red de forma estática en IPv4, se crea un fichero con el nombre ifcfg-xxxx (donde xxxx es el nombre del interfaz, p.e. enp1s0) en el directorio /etc/sysconfig/network-scripts, con un contenido como el que sigue:

```
DEVICE=eth0
BOOTPROTO=none
ONBOOT=yes
PREFIX=24
IPADDR=10.0.1.27
GATEWAY=10.0.1.1
```

- En RHEL8 debe estar corriendo NetworkManager para gestionar los cambios en la configuración de la red. Para que NetworkManager detecte cambios en los archivos ifcfg debe reconectarse usando el perfil con nmcli (también podría rebotarse el servicio o el servidor):

```
[[root@localhost ~]# nmcli connection reload
[[root@localhost ~]# nmcli con load /etc/sysconfig/network-scripts/ifcfg-enp1s0
```

4.L Configuración de un interfaz

```
[root@localhost ~]# ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen
1000
    link/ether 00:24:81:c1:61:cf brd ff:ff:ff:ff:ff:ff
    inet 192.168.7.68/24 brd 192.168.7.255 scope global dynamic enp1s0
       valid_lft 84942sec preferred_lft 84942sec
    inet6 fe80::19b9:ffa1:8680:f71a/64 scope link
       valid_lft forever preferred_lft forever
3: virbr0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group defa
ult qlen 1000
    link/ether 52:54:00:10:ae:97 brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.1/24 brd 192.168.122.255 scope global virbr0
       valid_lft forever preferred_lft forever
4: virbr0-nic: <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo_fast master virbr0 state DOWN gro
up default qlen 1000
    link/ether 52:54:00:10:ae:97 brd ff:ff:ff:ff:ff:ff
```

4.L Configuración de un interfaz

```
[root@localhost ~]# ifconfig
enp1s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.7.68  netmask 255.255.255.0  broadcast 192.168.7.255
        inet6 fe80::19b9:ffa1:8680:f71a  prefixlen 64  scopeid 0x20<link>
        ether 00:24:81:c1:61:cf  txqueuelen 1000  (Ethernet)
        RX packets 7379412  bytes 1427996296 (1.3 GiB)
        RX errors 61  dropped 0  overruns 0  frame 95
        TX packets 324334  bytes 23691674 (22.5 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 17

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1  (Local Loopback)
        RX packets 5259  bytes 613527 (599.1 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 5259  bytes 613527 (599.1 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

virbr0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 192.168.122.1  netmask 255.255.255.0  broadcast 192.168.122.255
        ether 52:54:00:10:ae:97  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[root@localhost ~]#
```

4. Tabla de rutas

- Todos los dispositivos de red, hosts y gateways, deben tomar decisiones de enrutado. Para la mayoría de los hosts, las decisiones de enrutado son simples:
    - Si el host de destino está en la red local, los datos se entregan al host de destino
    - Si el host de destino está en una red remota, los datos se reenvían a un gateway local
- Las decisiones de enrutado de IP son búsquedas en tablas. Los paquetes se enrutan a sus destinos de acuerdo con la tabla de rutas (forwarding table). Esta tabla asocia destinos al router y al interfaz de red que IP debe usar para alcanzar ese destino

4.L Configuración de red: Tabla de rutas en Linux

```
[root@localhost ~]# ip route
default via 192.168.7.1 dev enp1s0 proto static metric 100
192.168.7.0/24 dev enp1s0 proto kernel scope link src 192.168.7.68 metric 100
192.168.122.0/24 dev virbr0 proto kernel scope link src 192.168.122.1
[root@localhost ~]# route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         gateway         0.0.0.0         UG    100    0        0 enp1s0
192.168.7.0     0.0.0.0         255.255.255.0   U     100    0        0 enp1s0
192.168.122.0   0.0.0.0         255.255.255.0   U     0      0        0 virbr0
[root@localhost ~]#
```

- Se usan los comandos **ip route** o **route**, con la opción -n para que no se conviertan las direcciones IP a nombres
- El significado de los campos de la orden route es:
  - Destination: Valor con el que se compara la dirección IP
  - Gateway: Router que se usará para llegar al destino
  - Genmask: Máscara de red relacionado con el campo destino
  - Flags: U (ruta operacional) H (ruta a un host, no a una red) G (ruta que usa un gateway externo) R (ruta instalada por un protocolo de enrutamiento dinámico con la opción reinstate) D (ruta añadida por un mensaje ICMP Redirect) M (instalada con la opción mod) A (cacheada y con entrada asociada en la tabla ARP) L (ruta local, a una de las direcciones de este host), B (ruta cuyo destino es una dirección Broadcast) ! (datagramas dirigidos a esta ruta serán rechazados)

4.L Configuración de red: Tabla de rutas en Linux

```
# route -n
Kernel IP routing table
Destination   Gateway      Genmask        Flags Metric Ref   Use Iface
172.16.55.0   0.0.0.0      255.255.255.0  U     0      0       0 eth0
172.16.50.0   172.16.55.36 255.255.255.0  UG    0      0       0 eth0
127.0.0.0     0.0.0.0      255.0.0.0      U     0      0       0 lo
0.0.0.0       172.16.55.1  0.0.0.0        UG    0      0       0 eth0
```

  - Metric: El coste de la ruta, usando para elegir entre rutas duplicadas que aparezcan en la tabla
  - Ref: Número de veces que esta ruta se ha usado para establecer una conexión (no usado en Linux)
  - Iface: nombre del interfaz de red usado por esta ruta.

- IP usa la información de la tabla de rutas para construir las rutas usadas por las conexiones activas. Las rutas asociadas a las conexiones activas se almacenan en un cache, que puede consultarse con la orden route -Cn
- El cache es diferente de la tabla de rutas porque sólo se muestran las rutas establecidas: la tabla de rutas se usa para tomar decisiones, el cache lista las decisiones que ya se han tomado.

4.W Configuración de red: Tabla de rutas en W2019

```
PS C:\Users\Administrator> route print
===========================================================================
Interface List
  6...00 24 81 97 4a 11 ......Broadcom NetXtreme Gigabit Ethernet
 10...00 a0 c5 86 5f a1 ......ZyXEL FN312 10/100 PCI Adapter
  1...........................Software Loopback Interface 1
===========================================================================

IPv4 Route Table
===========================================================================
Active Routes:
Network Destination        Netmask          Gateway       Interface  Metric
          0.0.0.0          0.0.0.0      192.168.7.1     192.168.7.60    291
        127.0.0.0        255.0.0.0         On-link         127.0.0.1    331
        127.0.0.1  255.255.255.255         On-link         127.0.0.1    331
  127.255.255.255  255.255.255.255         On-link         127.0.0.1    331
      192.168.7.0    255.255.255.0         On-link      192.168.7.60    291
     192.168.7.60  255.255.255.255         On-link      192.168.7.60    291
    192.168.7.255  255.255.255.255         On-link      192.168.7.60    291
        224.0.0.0        240.0.0.0         On-link         127.0.0.1    331
        224.0.0.0        240.0.0.0         On-link      192.168.7.60    291
  255.255.255.255  255.255.255.255         On-link         127.0.0.1    331
  255.255.255.255  255.255.255.255         On-link      192.168.7.60    291
===========================================================================
Persistent Routes:
  Network Address          Netmask  Gateway Address  Metric
          0.0.0.0          0.0.0.0      192.168.7.1  Default
===========================================================================

IPv6 Route Table
===========================================================================
Active Routes:
 If Metric Network Destination      Gateway
  1    331 ::1/128                  On-link
  6    291 fe80::/64                On-link
  6    291 fe80::2d75:4b6e:8ca8:d5eb/128
                                    On-link
  1    331 ff00::/8                 On-link
  6    291 ff00::/8                 On-link
===========================================================================
Persistent Routes:
  None
PS C:\Users\Administrator>
```

- El comando **route print** muestra la tabla de rutas en tres secciones:
  - Lista de interfaces
  - Rutas activas: que pueden ser modificadas por la red
  - Rutas persistentes: rutas estáticas que han sido definidas por el administrador
- Cada ruta activa tiene los siguientes campos:
  - Destination
  - Netmask
  - Gateway
  - Interface: dirección del interfaz de red usado por la ruta
  - Metric: el coste de la ruta, usado si hay duplicados

4.L Configuración de red: Resolución de direcciones en Linux

```
[root@localhost ~]# arp -a
gateway (192.168.7.1) at 10:be:f5:47:d0:74 [ether] on enp1s0
[root@localhost ~]#
```

```
% arp -a


Net to Media Table: IPv4

Device   IP Address               Mask      Flags   Phys Addr

------ -------------------- --------------- ----- ---------------

dnet0  rodent               255.255.255.255       00:50:ba:3f:c2:5e

dnet0  crab                 255.255.255.255 SP    00:00:c0:dd:d4:da

dnet0  224.0.0.0            240.0.0.0       SM    01:00:5e:00:00:00
```

- El software ARP (Address Resolution Protocol) mantiene una tabla de traducciones entre direcciones IP y Ethernet, que se construye dinámicamente. Cuando ARP recibe una consulta, busca la dirección en la tabla. Si no la encuentra, hace broadcast de un paquete a cada host en la Ethernet, conteniendo la dirección IP de la máquina cuya dirección Ethernet se busca. Si el host identifica la dirección como suya, responde al paquete y ARP almacena la asociación.
- El flag P significa “publicar”. Es posible publicar direcciones de otros hosts (proxy ARP)

4.W Configuración de red: Resolución de direcciones en W2019

```
PS C:\Users\Administrator> arp -a

Interface: 192.168.7.60 --- 0x6
  Internet Address      Physical Address      Type
  192.168.7.1           10-be-f5-47-d0-74     dynamic
  192.168.7.8           08-00-27-8b-7f-de     dynamic
  192.168.7.73          98-5a-eb-d1-48-e6     dynamic
  192.168.7.255         ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
PS C:\Users\Administrator>
```

- La sintaxis es similar: dirección IP / dirección física / entrada estática o dinámica

4.L ping

```
[[root@localhost ~]# ping 156.35.119.120
PING 156.35.119.120 (156.35.119.120) 56(84) bytes of data.
64 bytes from 156.35.119.120: icmp_seq=1 ttl=62 time=0.862 ms
64 bytes from 156.35.119.120: icmp_seq=2 ttl=62 time=0.718 ms
64 bytes from 156.35.119.120: icmp_seq=3 ttl=62 time=0.776 ms
64 bytes from 156.35.119.120: icmp_seq=4 ttl=62 time=0.764 ms
64 bytes from 156.35.119.120: icmp_seq=5 ttl=62 time=0.738 ms
64 bytes from 156.35.119.120: icmp_seq=6 ttl=62 time=0.932 ms
64 bytes from 156.35.119.120: icmp_seq=7 ttl=62 time=1.26 ms
64 bytes from 156.35.119.120: icmp_seq=8 ttl=62 time=0.744 ms
64 bytes from 156.35.119.120: icmp_seq=9 ttl=62 time=0.733 ms
64 bytes from 156.35.119.120: icmp_seq=10 ttl=62 time=0.768 ms
64 bytes from 156.35.119.120: icmp_seq=11 ttl=62 time=0.771 ms
^C
--- 156.35.119.120 ping statistics ---
11 packets transmitted, 11 received, 0% packet loss, time 10008ms
rtt min/avg/max/mdev = 0.718/0.824/1.265/0.155 ms
```

- ping se usa para comprobar si un host es alcanzable
- Mide el round-trip (tiempo hasta que se recibe la respuesta del host remoto)
- Envía paquetes ICMP que solicita un eco y espera por la respuesta. Se muestran los errores, las pérdidas de paquetes, mínimo, máximo y promedio de round-trip

4.L traceroute / tracert

```
[[root@localhost ~]# traceroute www.google.es
traceroute to www.google.es (216.58.211.35), 30 hops max, 60 byte packets
 1  gateway (192.168.7.1)  0.330 ms  0.349 ms  0.414 ms
 2  r150205.red.uniovi.es (156.35.150.205)  11.443 ms  11.615 ms  12.119 ms
 3  156.35.139.215 (156.35.139.215)  26.267 ms  26.261 ms  26.901 ms
 4  uov-o2xp226.uoviedo.uniovi.es (156.35.3.226)  3.151 ms  3.115 ms  3.127 ms
 5  uov-xxx2yyy024.uoviedo.uniovi.es (156.35.3.250)  3.555 ms  3.569 ms  3.571 m
s
 6  * * *
 7  XE3-2-0-53.uva.rt1.cyl.red.rediris.es (130.206.201.113)  6.192 ms  6.202 ms
 6.678 ms
 8  uva.ae2.ciemat.rt1.mad.red.rediris.es (130.206.245.9)  9.535 ms  22.088 ms U
VA.AE1.unizar.rt1.ara.red.rediris.es (130.206.245.13)  15.324 ms
 9  CIEMAT.AE2.telmad.rt4.mad.red.rediris.es (130.206.245.2)  10.182 ms  9.658 m
s  9.274 ms
10  google-router.red.rediris.es (130.206.255.2)  9.335 ms  16.325 ms  16.313 ms
11  108.170.253.225 (108.170.253.225)  16.759 ms  16.189 ms 108.170.253.241 (108
.170.253.241)  17.343 ms
12  108.170.234.221 (108.170.234.221)  24.138 ms 108.170.234.231 (108.170.234.23
1)  23.769 ms 108.170.234.221 (108.170.234.221)  23.932 ms
13  muc03s14-in-f35.1e100.net (216.58.211.35)  15.919 ms  16.282 ms  23.571 ms
[root@localhost ~]#
```

- Mide los retardos de los paquetes en la red. Se muestran los tiempos round-trip de los paquetes recibidos por cada host en la ruta. El tiempo total de retorno es la suma de los tiempos. Si no se contesta en un número de segundos, se muestra un asterisco.

4.W netstat

```
PS C:\Users\Administrator> netstat

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    192.168.7.60:3389      192.168.7.1:56089      ESTABLISHED
  TCP    [fe80::2d75:4b6e:8ca8:d5eb%6]:135  W2019AS:49685          ESTABLISHED
  TCP    [fe80::2d75:4b6e:8ca8:d5eb%6]:49685  W2019AS:epmap          ESTABLISHED
PS C:\Users\Administrator>
```

```
PS C:\Users\Administrator> netstat -b

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    192.168.7.60:3389      192.168.7.1:56089      ESTABLISHED
  TermService
 [svchost.exe]
  TCP    [fe80::2d75:4b6e:8ca8:d5eb%6]:135  W2019AS:49685          ESTABLISHED
  RpcSs
 [svchost.exe]
  TCP    [fe80::2d75:4b6e:8ca8:d5eb%6]:49685  W2019AS:epmap          ESTABLISHED
 [tssdis.exe]
```

- netstat (network statistics) muestra diferentes estadísticas de la red
- se sigue usando en windows; en Linux se reemplaza por la orden ss

4.L netstat

```
[root@localhost ~]# ip maddr
1:      lo
        inet  224.0.0.1
        inet6 ff02::1
        inet6 ff01::1
2:      enp1s0
        link  01:00:5e:00:00:01
        link  33:33:00:00:00:01
        link  33:33:ff:80:f7:1a
        link  01:00:5e:00:00:fb
        inet  224.0.0.251
        inet  224.0.0.1
        inet6 ff02::1:ff80:f71a
        inet6 ff02::1
        inet6 ff01::1
3:      virbr0
        link  01:00:5e:00:00:01
        link  01:00:5e:00:00:fb
        inet  224.0.0.251
        inet  224.0.0.1
        inet6 ff02::1
        inet6 ff01::1
4:      virbr0-nic
        inet6 ff02::1
        inet6 ff01::1
[root@localhost ~]#
```

- **netstat –r** se reemplaza por **ip route**
- **netstat –i** se reemplaza por **ip –s link**
- **netstat –g** se reemplaza por **ip maddr**

4.L nslookup

```
[root@localhost ~]# nslookup www.uniovi.es
Server:         156.35.14.2
Address:        156.35.14.2#53

Name:   www.uniovi.es
Address: 156.35.233.105

[root@localhost ~]#
```

```
[root@localhost ~]# nslookup 156.35.119.120
Server:         156.35.14.2
Address:        156.35.14.2#53

120.119.35.156.in-addr.arpa     name = horru.lsi.uniovi.es.

[root@localhost ~]#
```

- nslookup sirve para consultar la dirección ip asociada a un nombre DNS
- se verá con más detalle cuando se estudie BIND/DNS

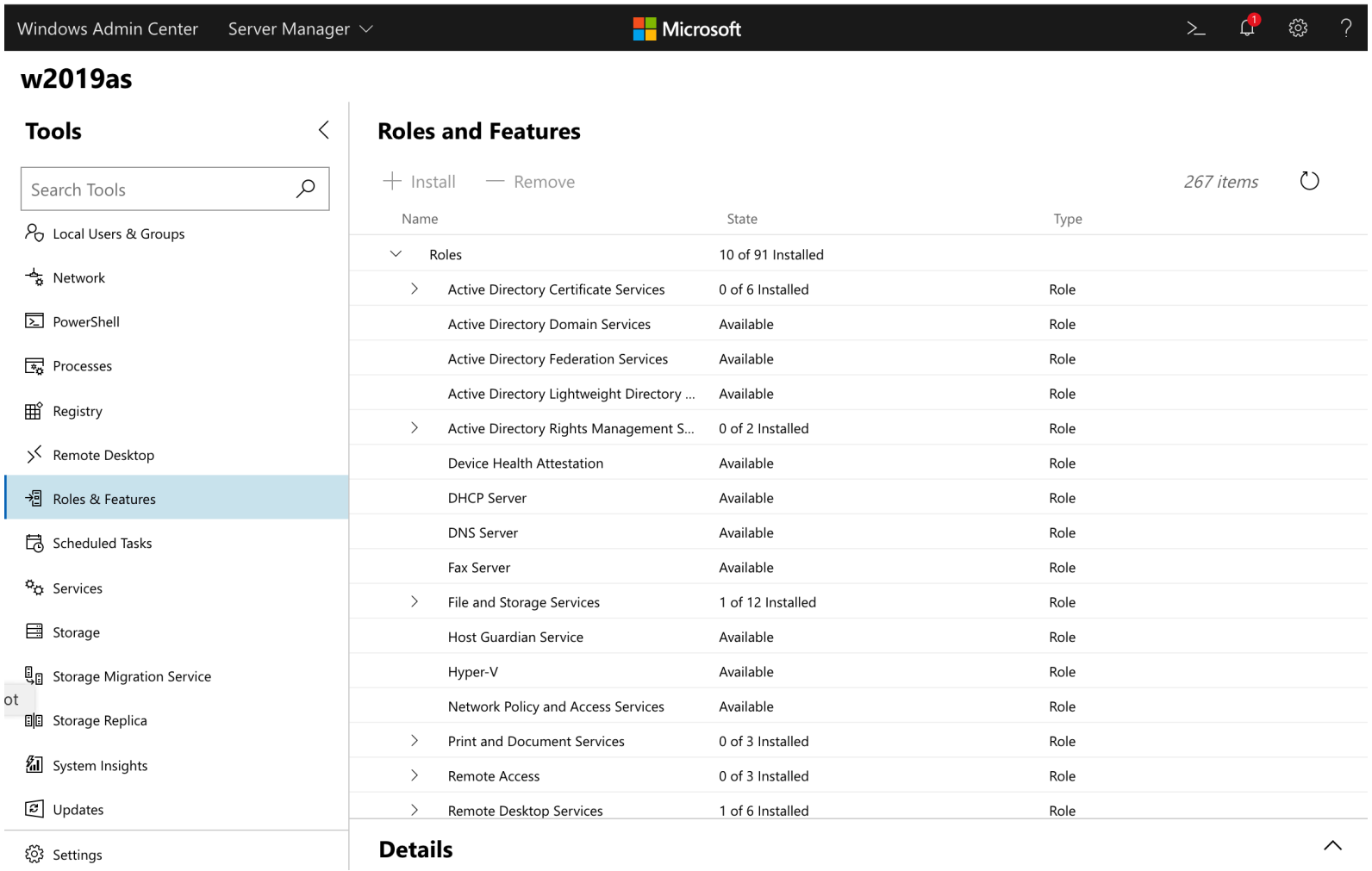
4.W servicio DHCP

- Un cliente DHCP emite con broadcast un paquete UDP llamado DHCPDISCOVER que contiene la identificación del cliente (la dirección MAC de ethernet). El cliente emite a 255.255.255.255 (no se necesita conocer el número de red). El cliente espera la respuesta de un servidor DHCP, y si no se recibe en un tiempo se repite la petición

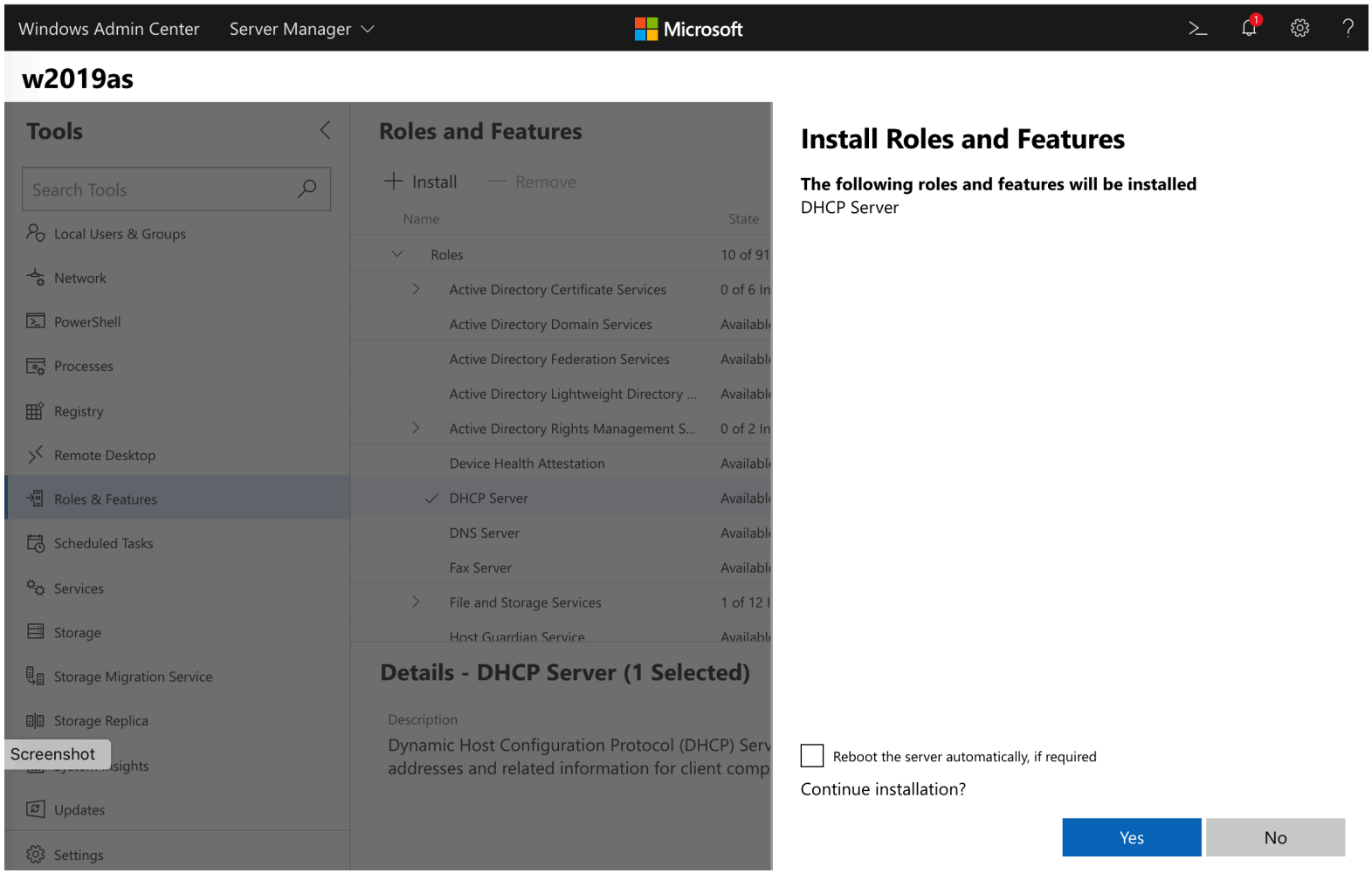
- El servidor responde con un paquete DHCPOFFER. Se usan los puertos UDP 67 para el servidor y 68 para el cliente. El servidor llena los datos de configuración en el paquete y se lo manda al cliente, que debe aceptarlo en 120 segundos (esto se hace por si hay más de un servidor).

- El cliente responde a DHCPOFFER con DHCPREQUEST, que confirma el uso de los recursos. Si todo es correcto, se le envía DHCPACK. Si no, DHCPNACK.

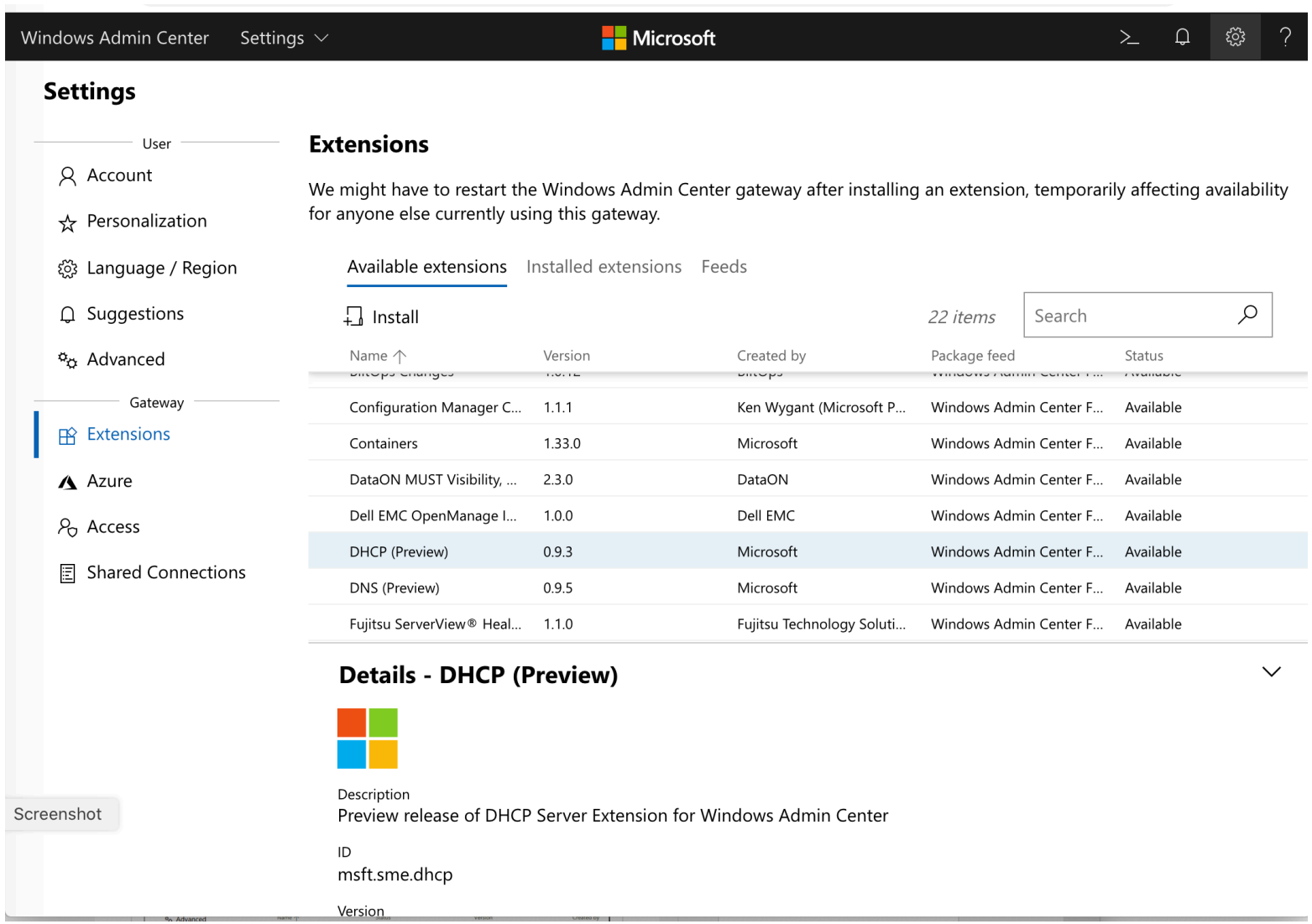
4.W servicio DHCP



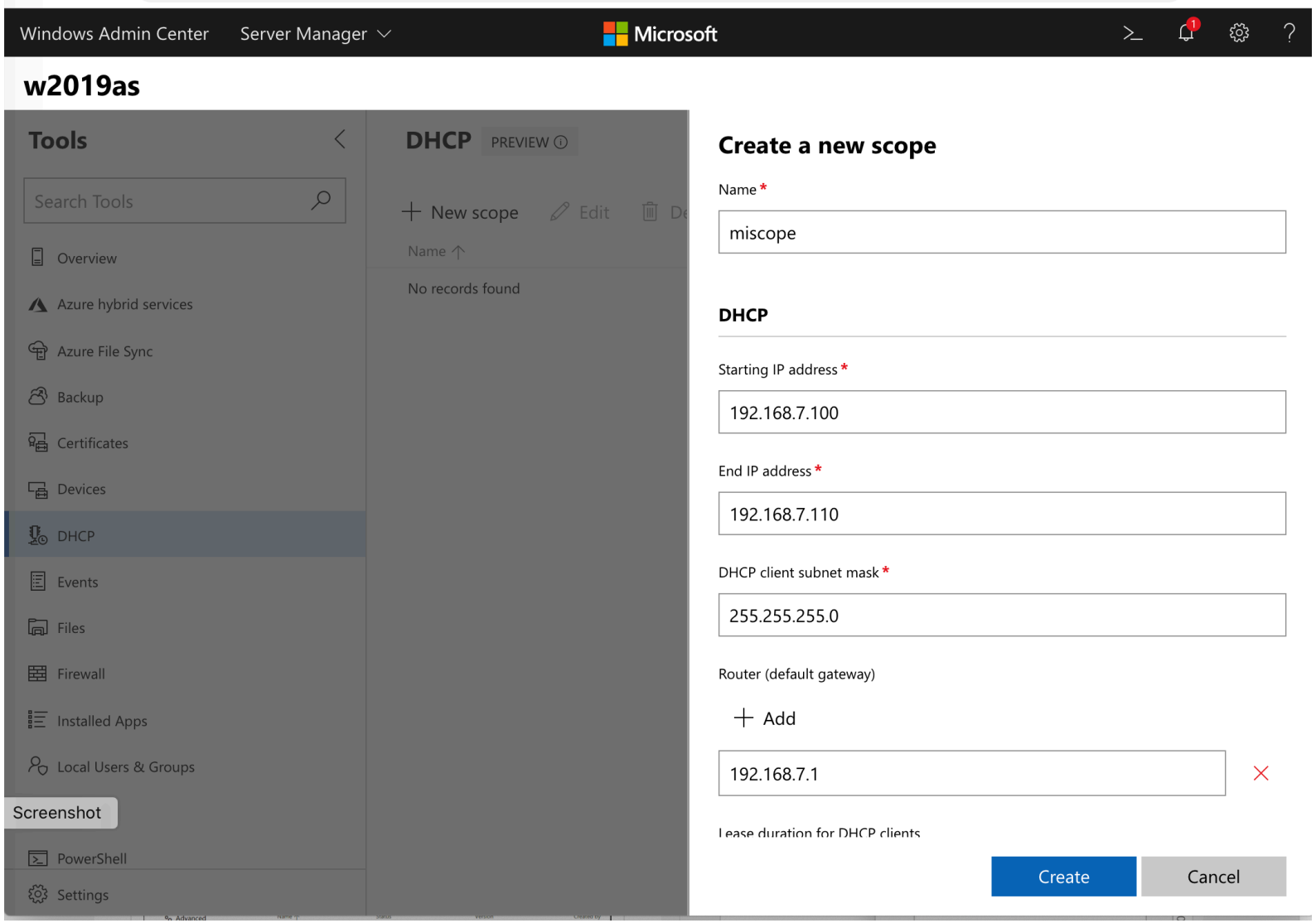
4.W servicio DHCP



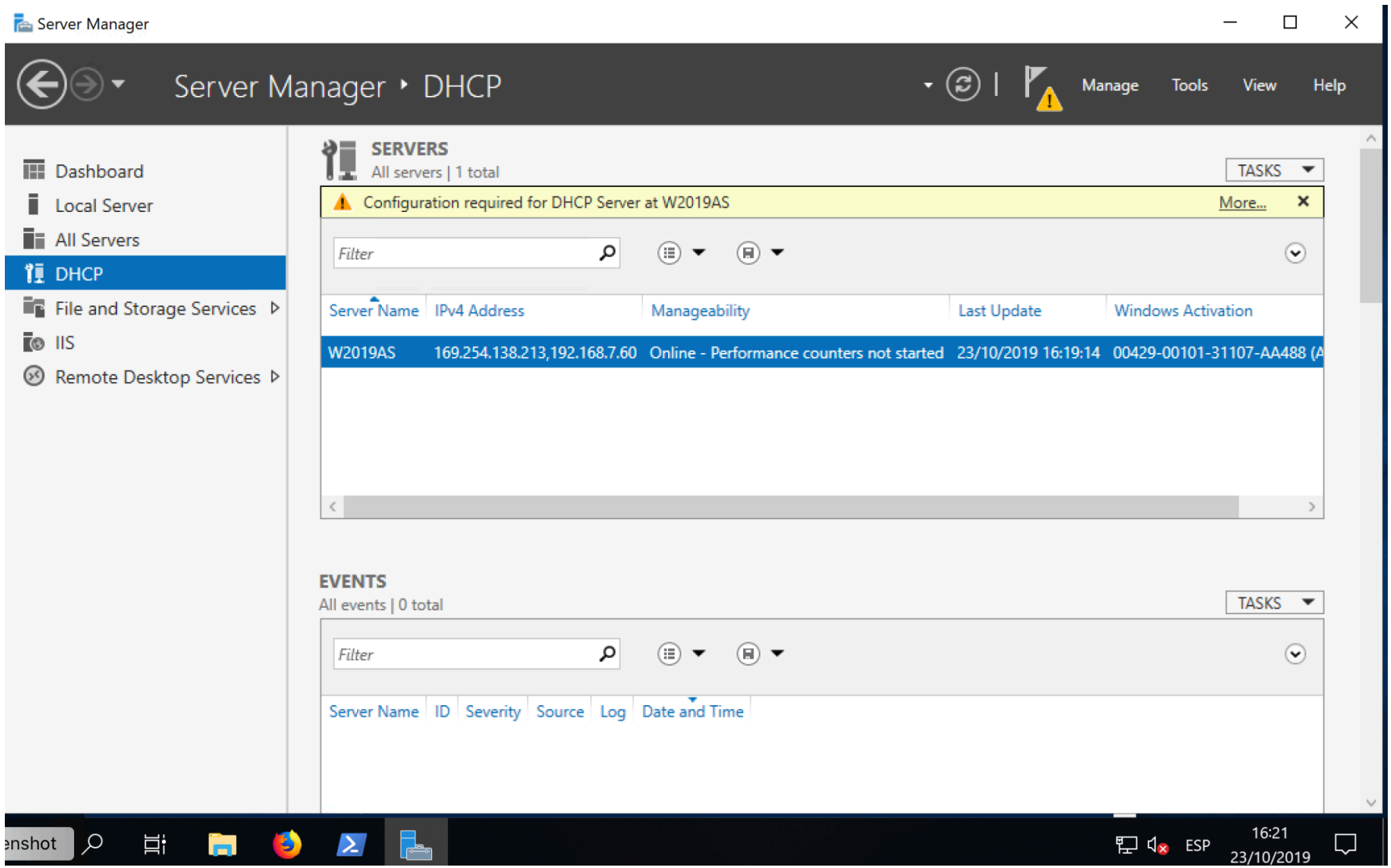
4.W servicio DHCP



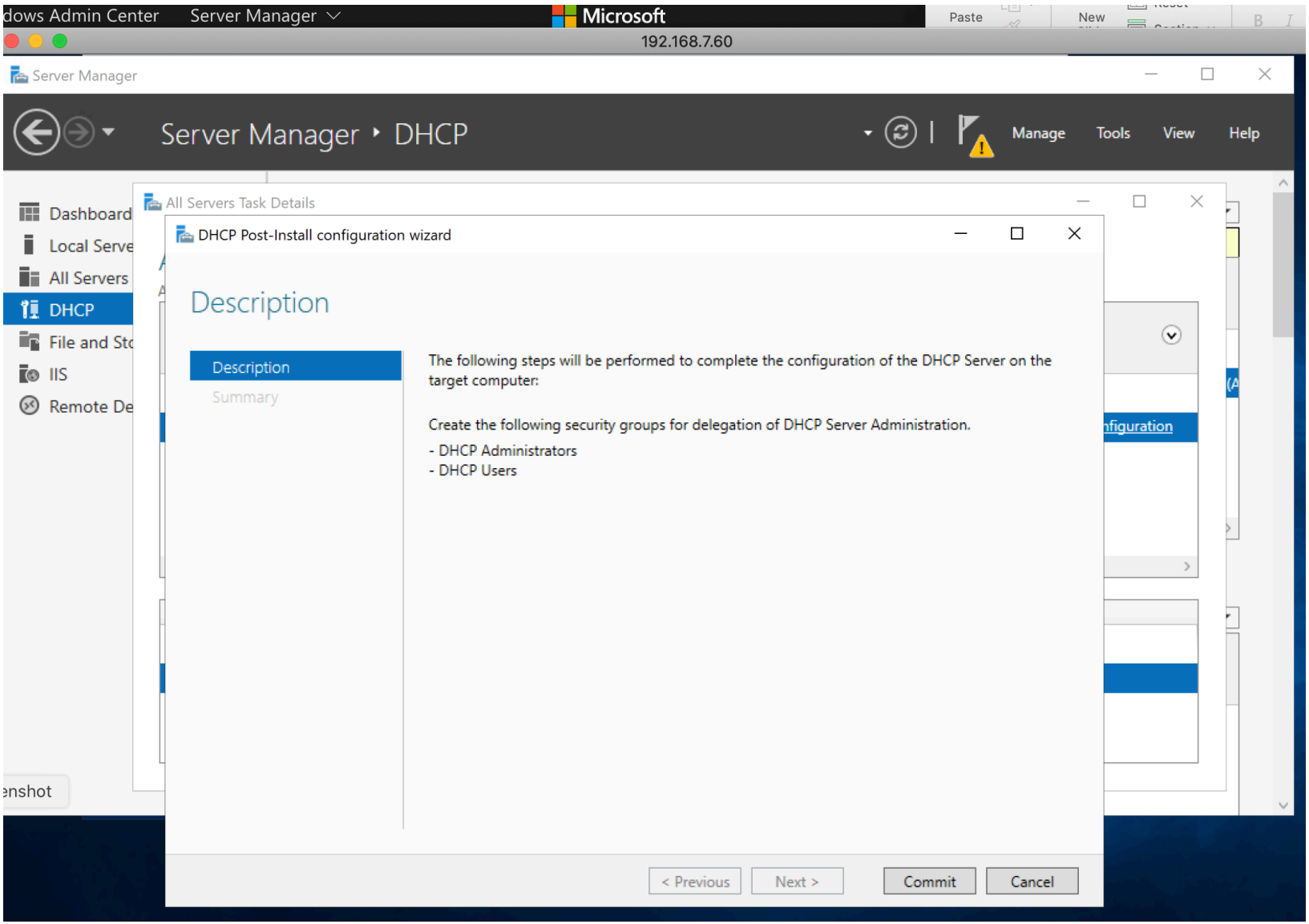
4.W servicio DHCP



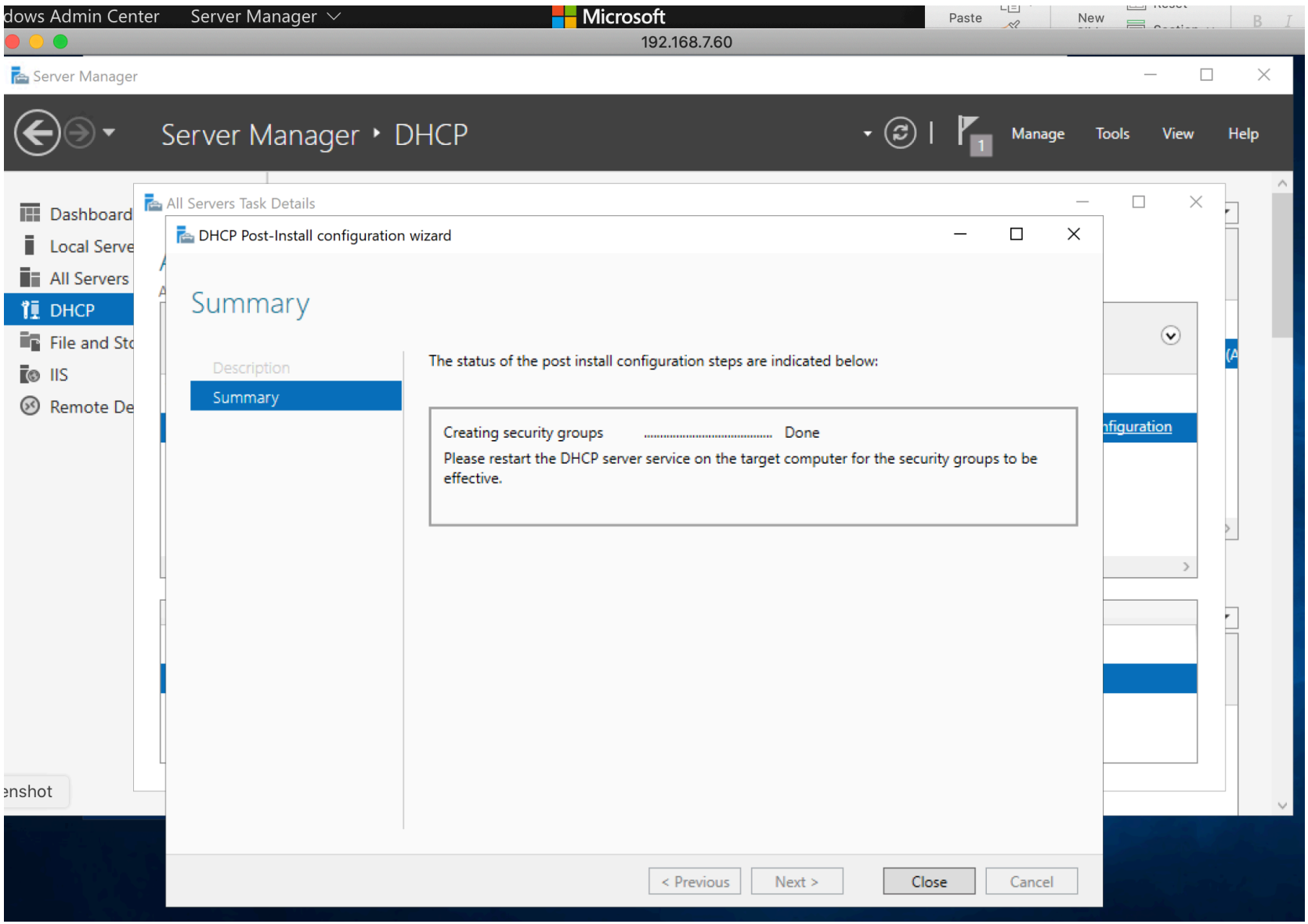
4.W servicio DHCP



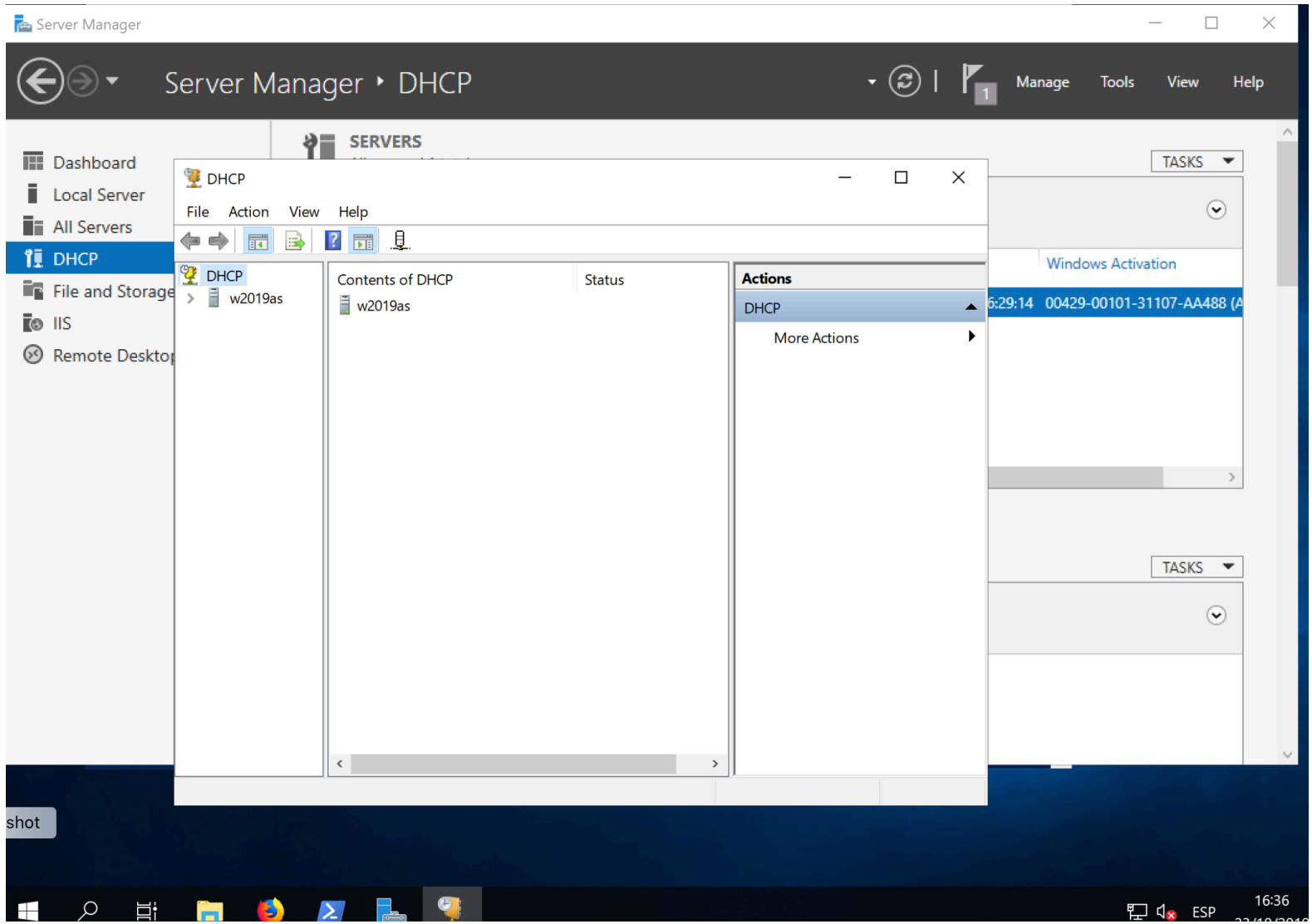
4.W servicio DHCP



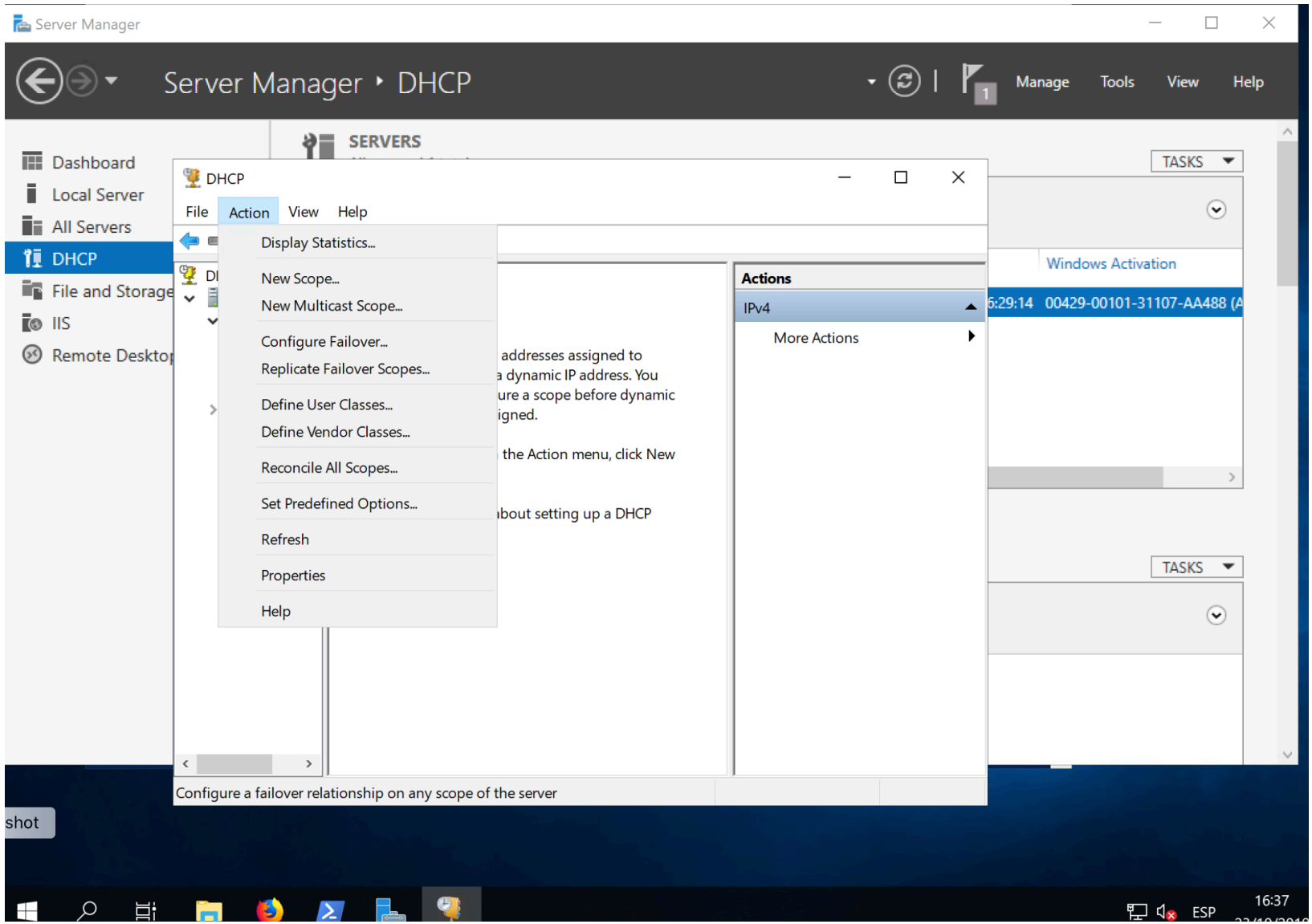
4.W servicio DHCP



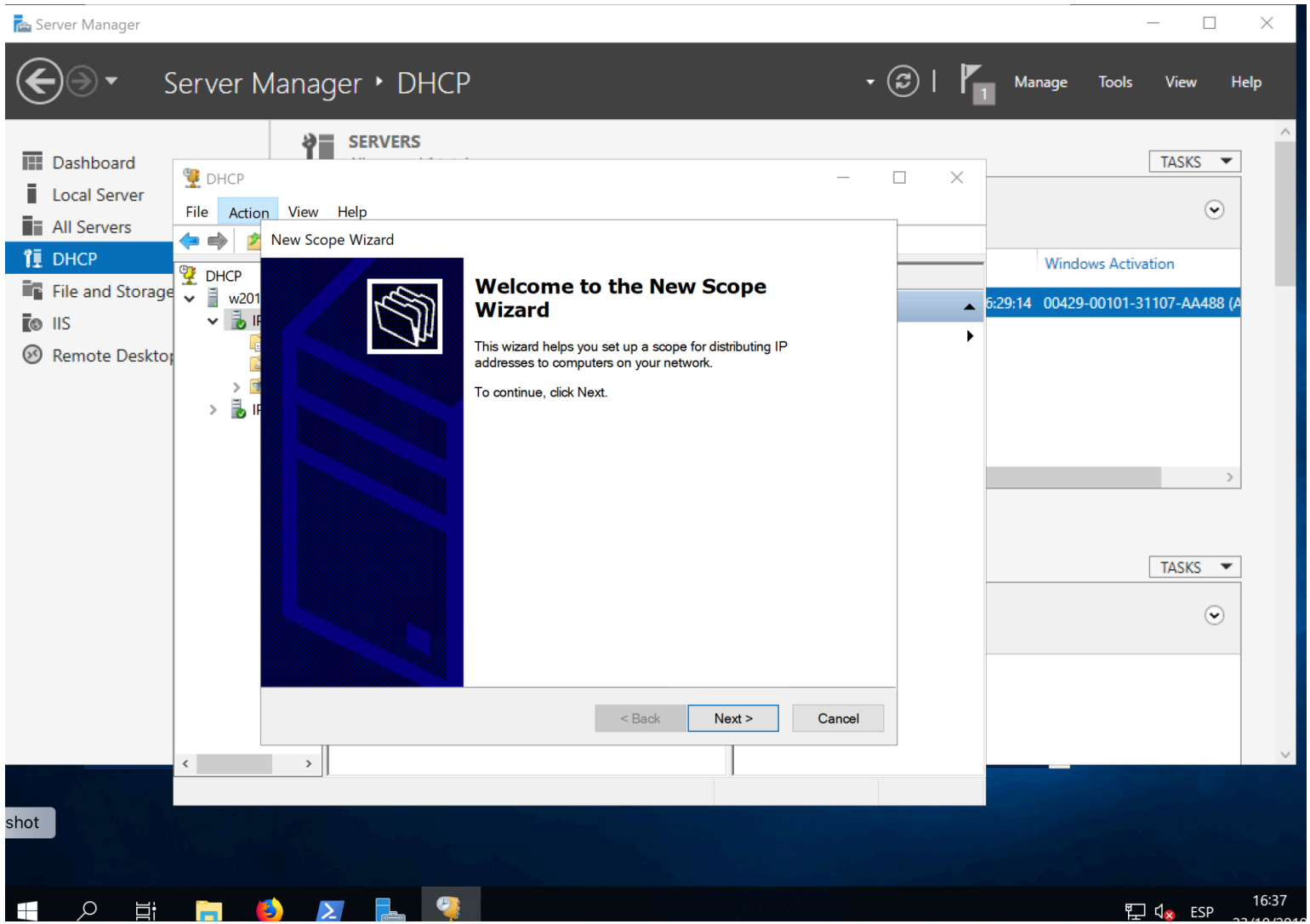
4.W servicio DHCP



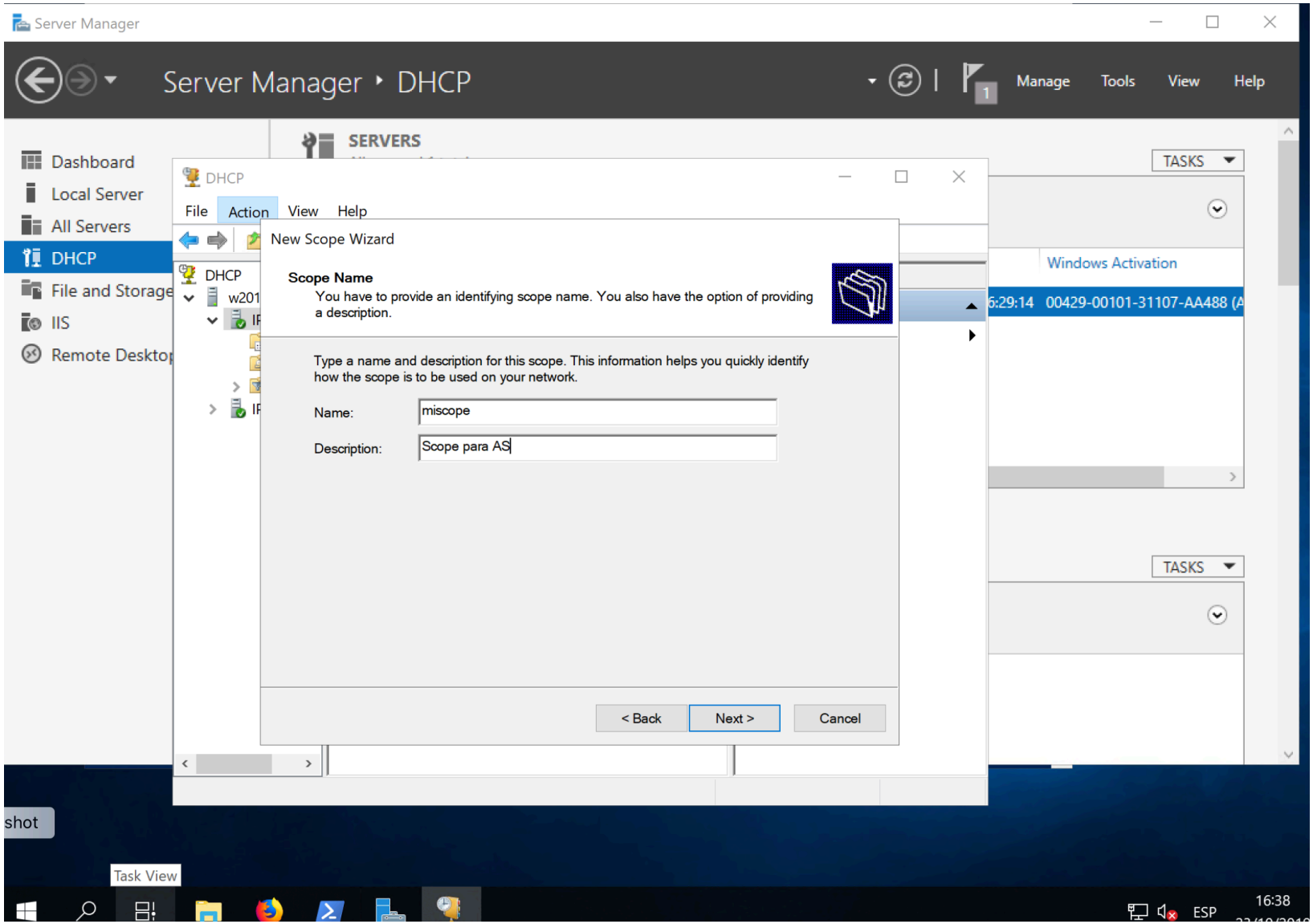
4.W servicio DHCP



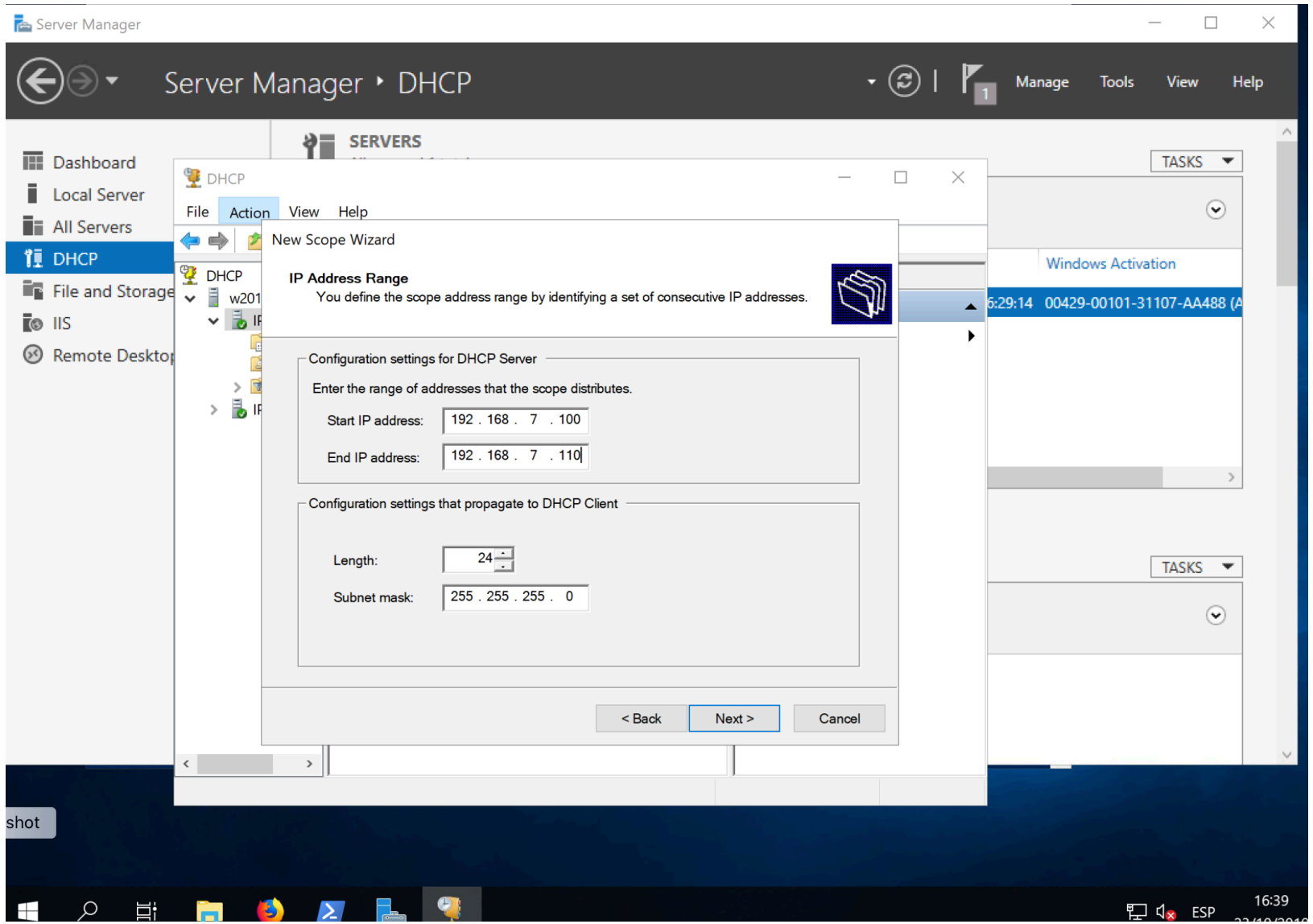
4.W servicio DHCP



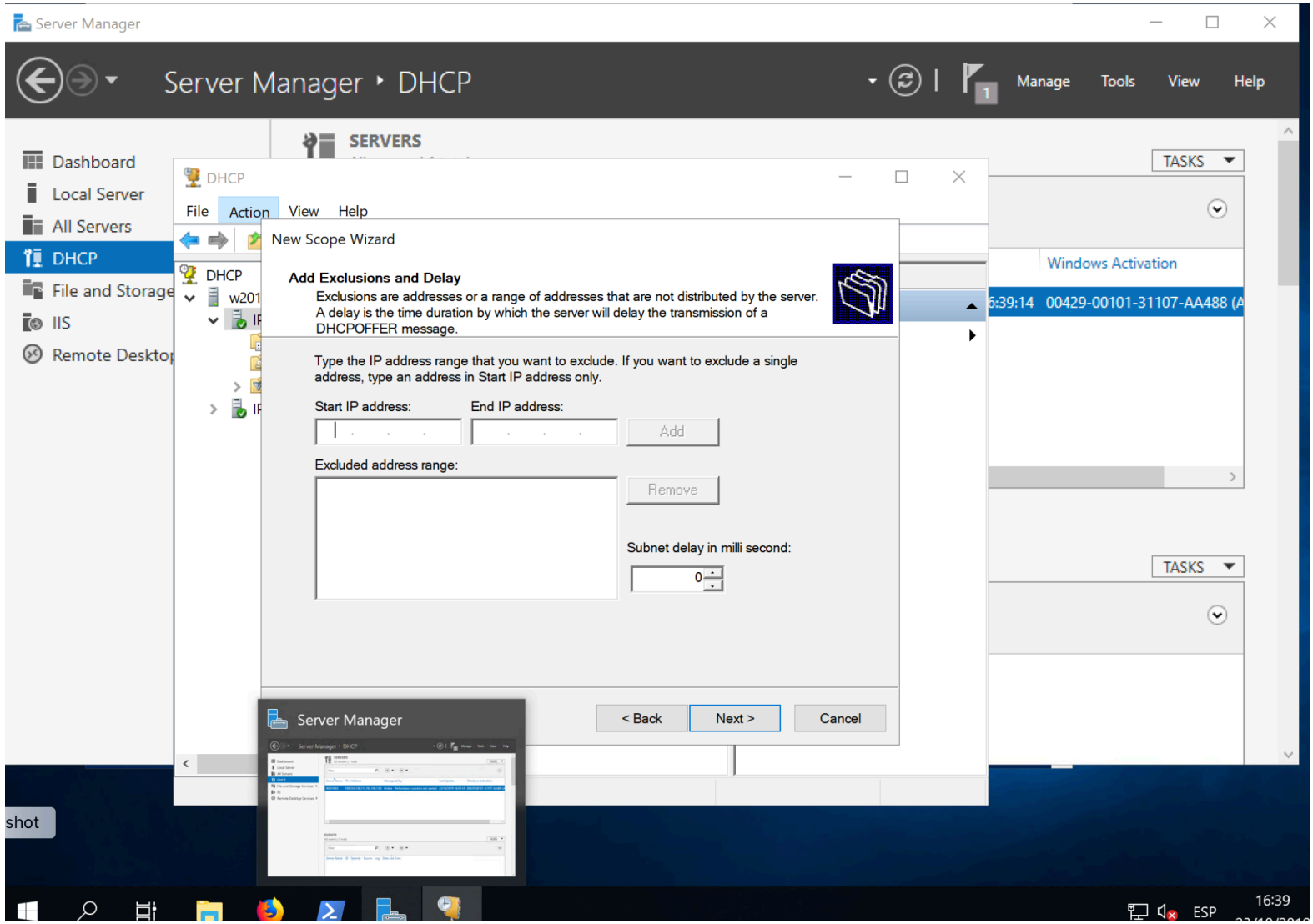
4.W servicio DHCP



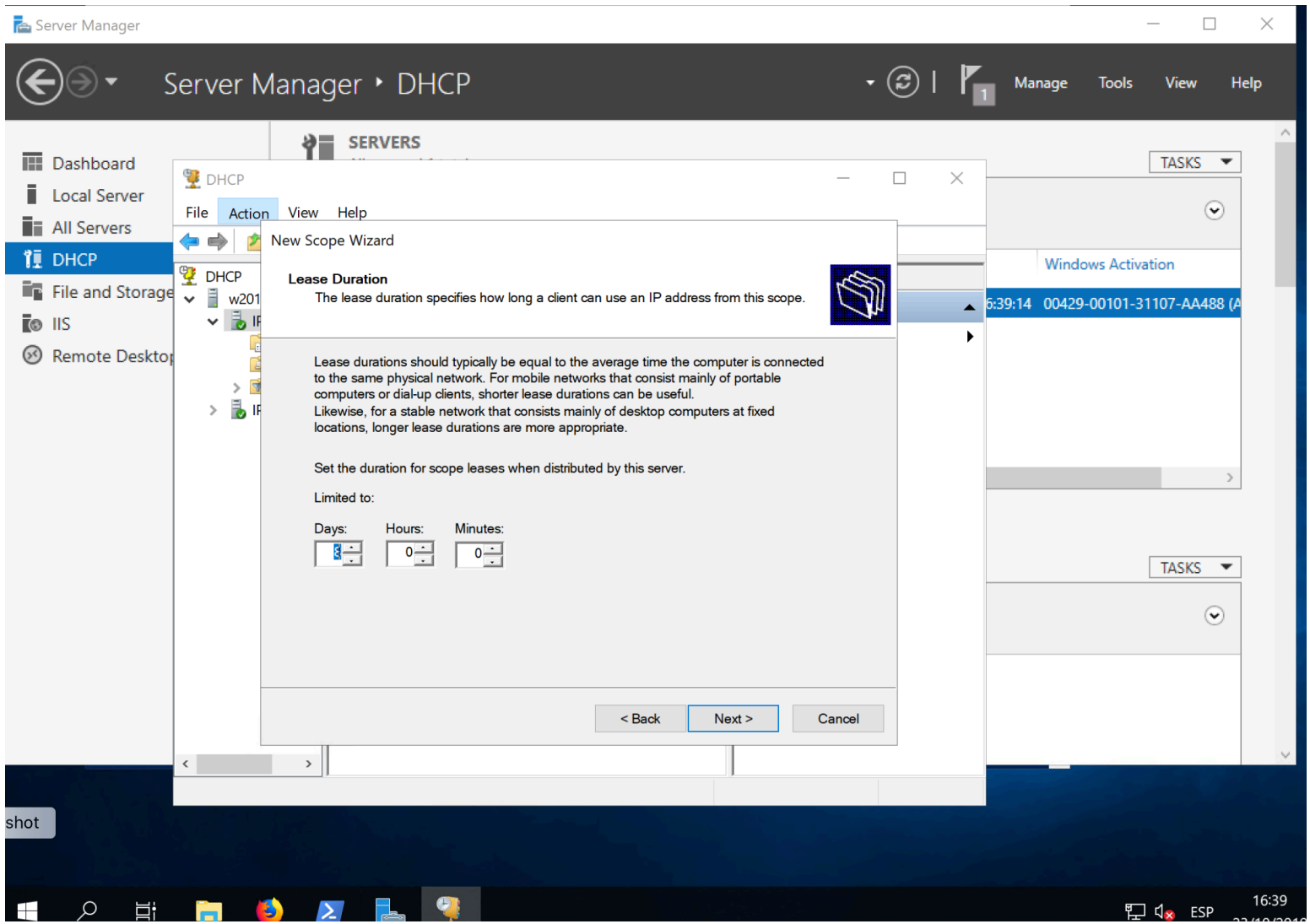
4.W servicio DHCP



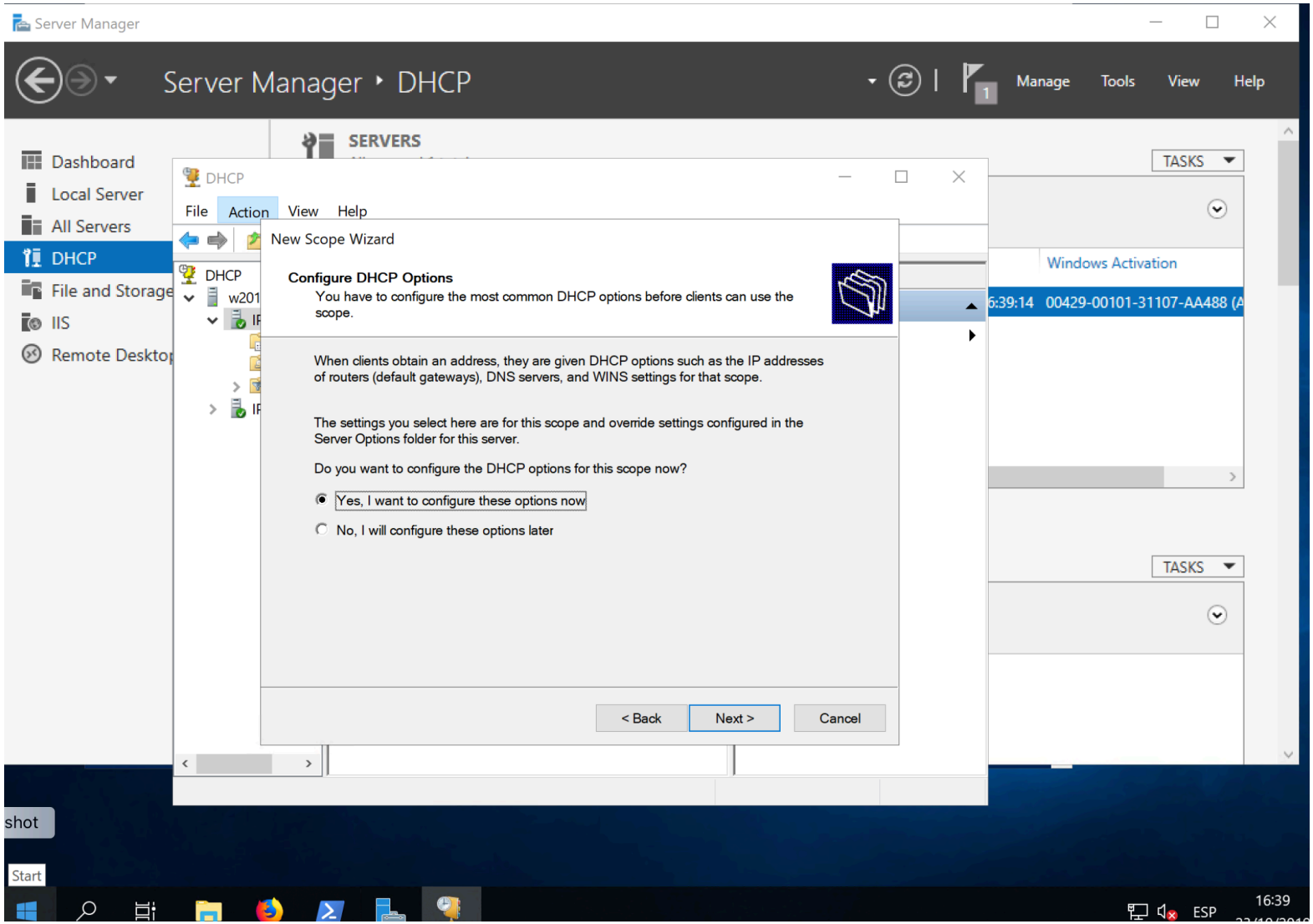
4.W servicio DHCP



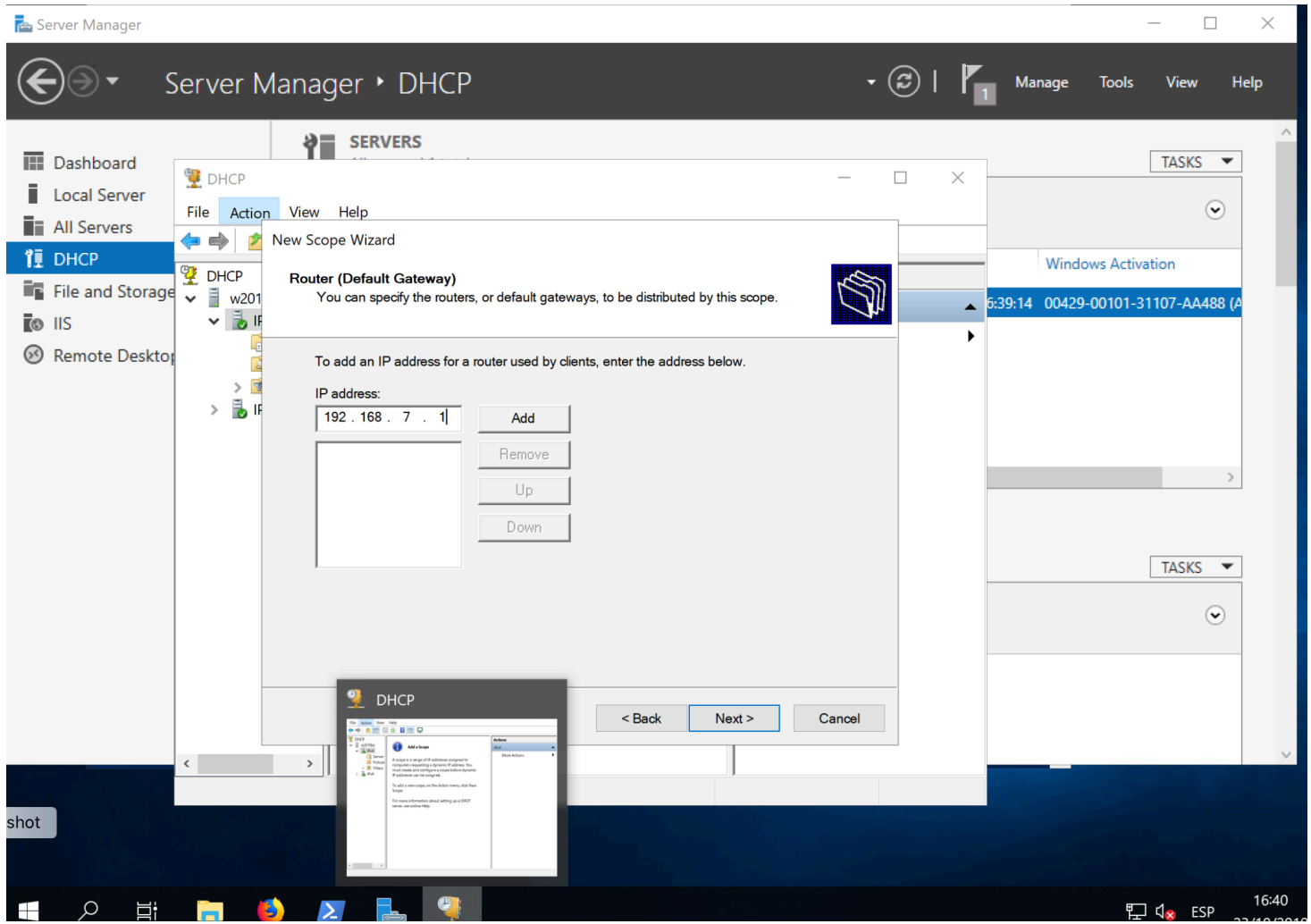
4.W servicio DHCP



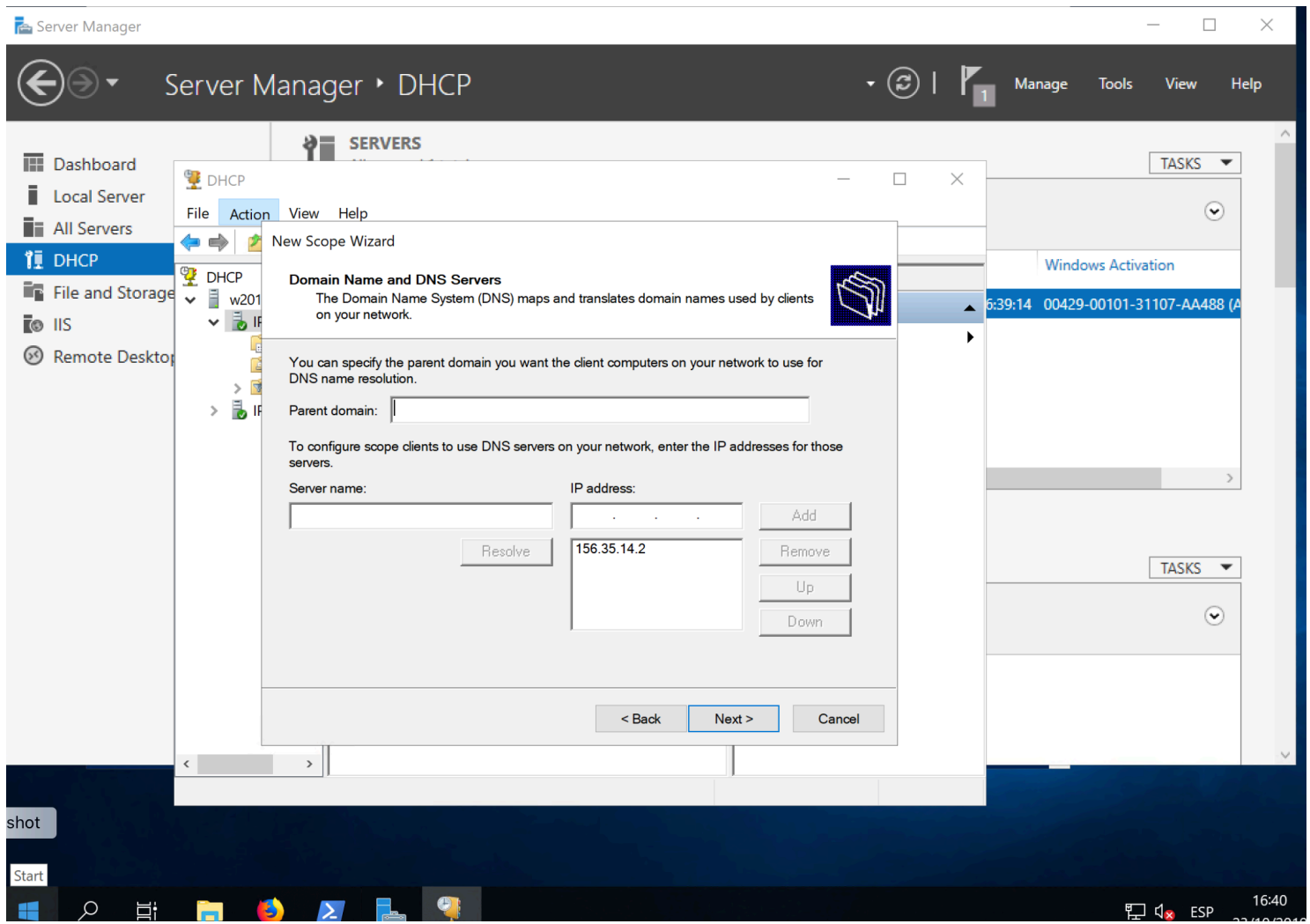
4.W servicio DHCP



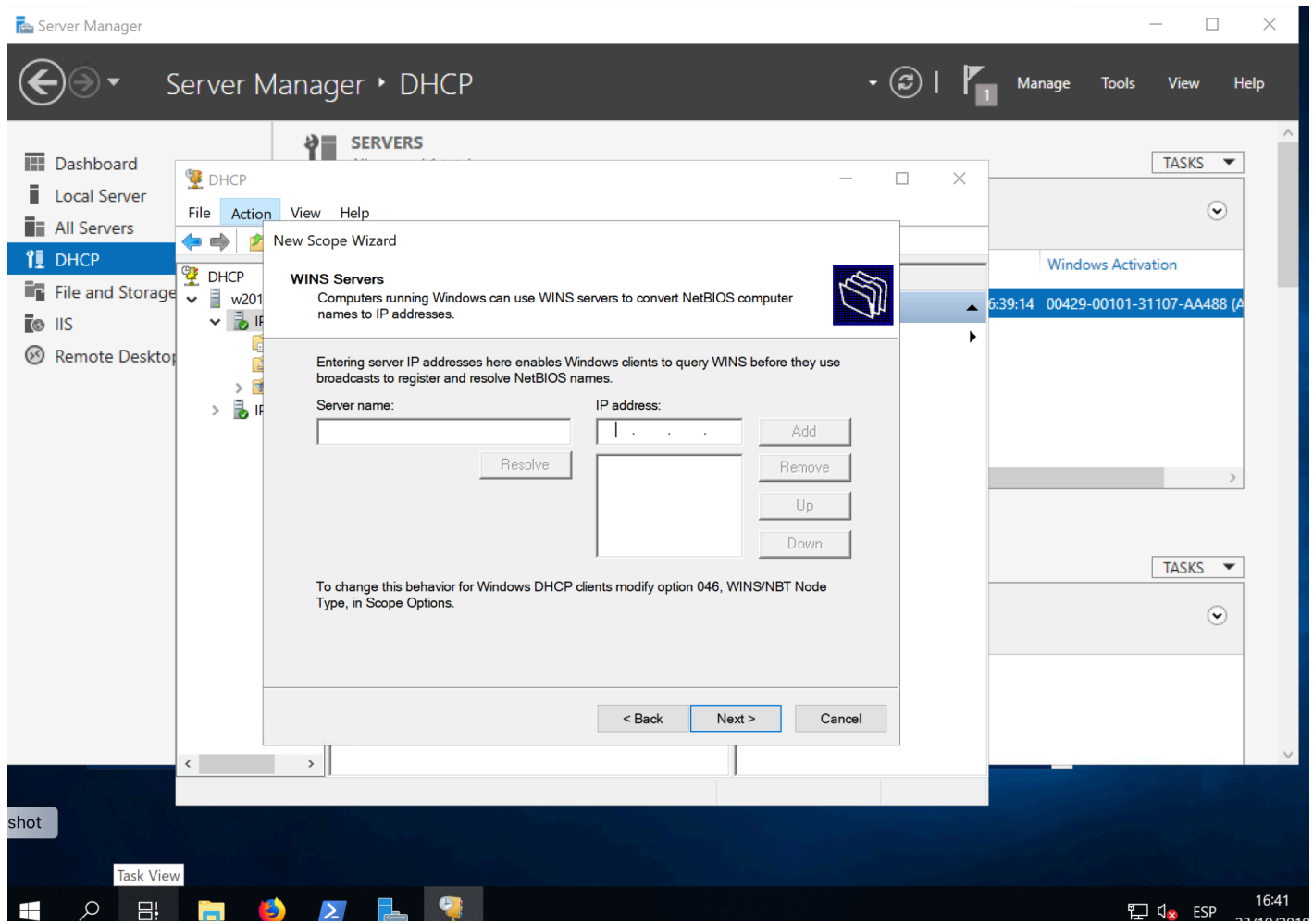
4.W servicio DHCP



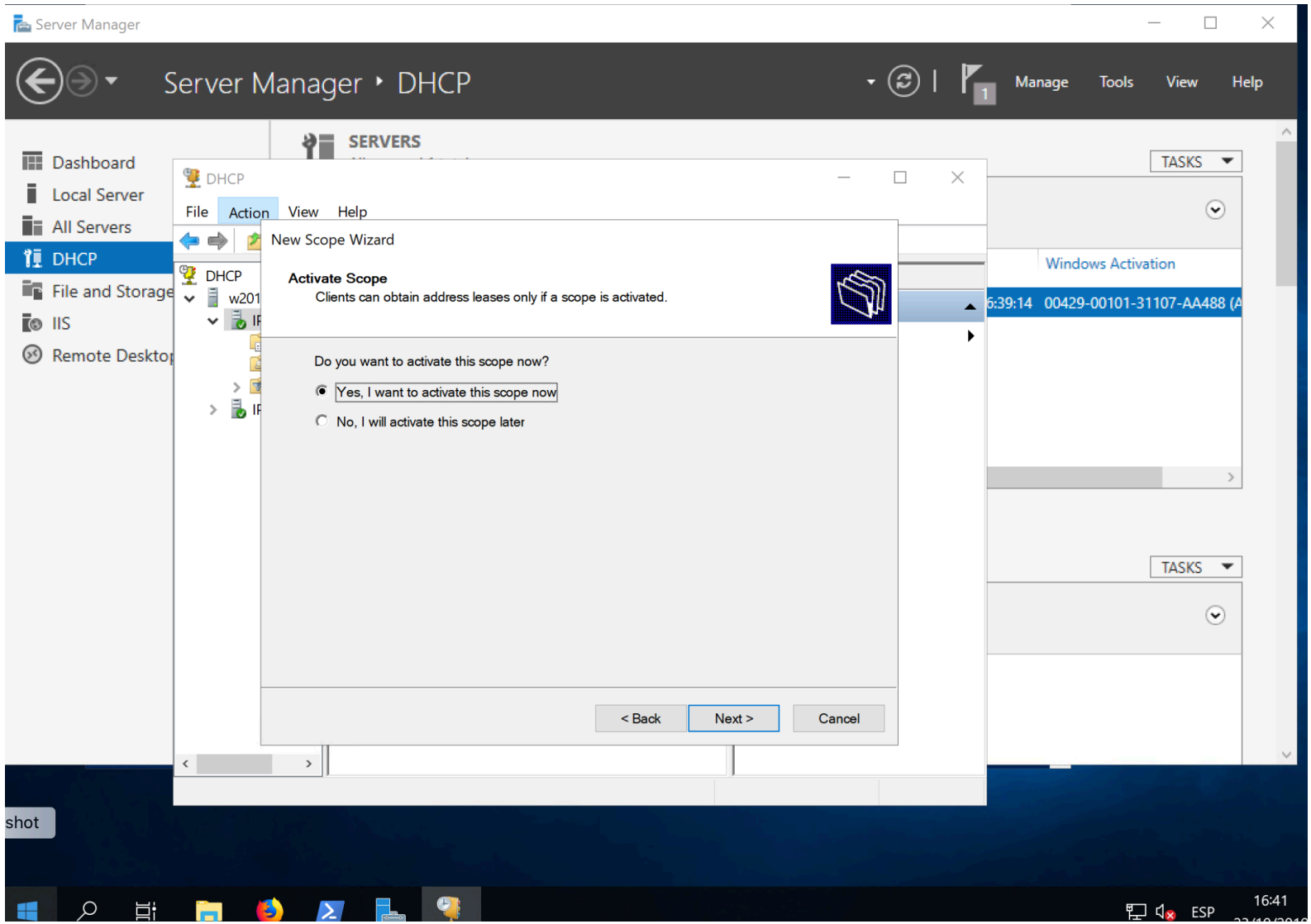
4.W servicio DHCP



4.W servicio DHCP



4.W servicio DHCP



4.L DHCP en Linux

- El responsable del servicio DHCP en linux es el proceso dhcpd (paquete dhcp-server)
- La instalación consta de los siguientes pasos:

1. se copia /usr/lib/systemd/system/dhcpd.service en /etc/systemd/system/
2. se edita /etc/systemd/system/dhcpd.service y se indican los nombres de los interfaces donde queremos que escuche el servicio en el parámetro ExecStart:

```
ExecStart=/usr/sbin/dhcpd -f -cf /etc/dhcp/dhcpd.conf -user dhcpd -group dhcpd --no-pid
$DHCPDARGS enp0s1 enp7s0
```

3. se recarga el servicio (systemctl daemon-reload y systemctl restart dhcpd.service)

4.L DHCP en Linux

- El archivo de configuración es /etc/dhcp/dhcpd.conf
- Con la orden “authoritative” los DHCPREQUEST no se contestan con DHCPNAK si se pide una IP fuera del rango
- Se crea una sección “subnet” por cada rango de direcciones:

```
authoritative;
subnet 192.168.252.0 netmask 255.255.255.0 {
  range 192.168.252.110 192.168.252.120;
  option domain-name-servers 156.35.160.2;
  option routers 192.168.252.100;
  option subnet-mask 255.255.255.0;
}
```

4.L Procesamiento de paquetes en iptables

- Todos los paquetes inspeccionados por iptables pasan por una secuencia de colas (*queues*) en las que son procesados, Cada una de esas colas está asociada a un tipo particular de actividad, y está controlada por una cadena (*chain*) de transformación/filtrado de paquetes.
- Hay tres tablas:
  - La primera es la "*mangle table*", responsable de alterar los bits de calidad de servicio en la cabecera TCP (sin interés para una organización pequeña)
  - La segunda es la tabla de filtrado "*filter table*". Tiene tres cadenas, en las que se pueden insertar las reglas con la política del firewall:
    - *Forward chain*: filtra paquetes a servidores protegidos por el firewall
    - *Input chain*: filtra paquetes destinados al firewall
    - *Output chain:* filtra paquetes originados en el firewall

4.L Procesamiento de paquetes en iptables

- La tercera tabla es la cola NAT, responsable de la traducción de direcciones de red. Tiene dos cadenas (más una tercera con poco interés en pequeñas organizaciones):
  - *pre-routing chain:* paquetes NAT cuando la dirección de destino del paquete tiene que cambiarse
  - *post-routing chain:* paquetes NAT cuando la dirección de origen del paquete tiene que cambiarse

4.L Procesamiento de paquetes en iptables

Queue Type	Queue Function	Packet Transformation Chain In Queue	Chain Function
Filter	Packet filtering	`FORWARD`	Filters packets to servers accessible by another NIC on the firewall.
		`INPUT`	Filters packets destined to the firewall.
		`OUTPUT`	Filters packets originating from the firewall
Nat	Network Address Translation	`PREROUTING`	Address translation occurs before routing. Facilitates the transformation of the destination IP address to be compatible with the firewall's routing table. Used with NAT of the destination IP address, also known as **destination NAT** or **DNAT**.
		`POSTROUTING`	Address translation occurs after routing. This implies that there was no need to modify the destination IP address of the packet as in pre-routing. Used with NAT of the source IP address using either one-to-one or many-to-one NAT. This is known as **source NAT**, or **SNAT**.
		`OUTPUT`	Network address translation for packets generated by the firewall. (Rarely used in SOHO environments)

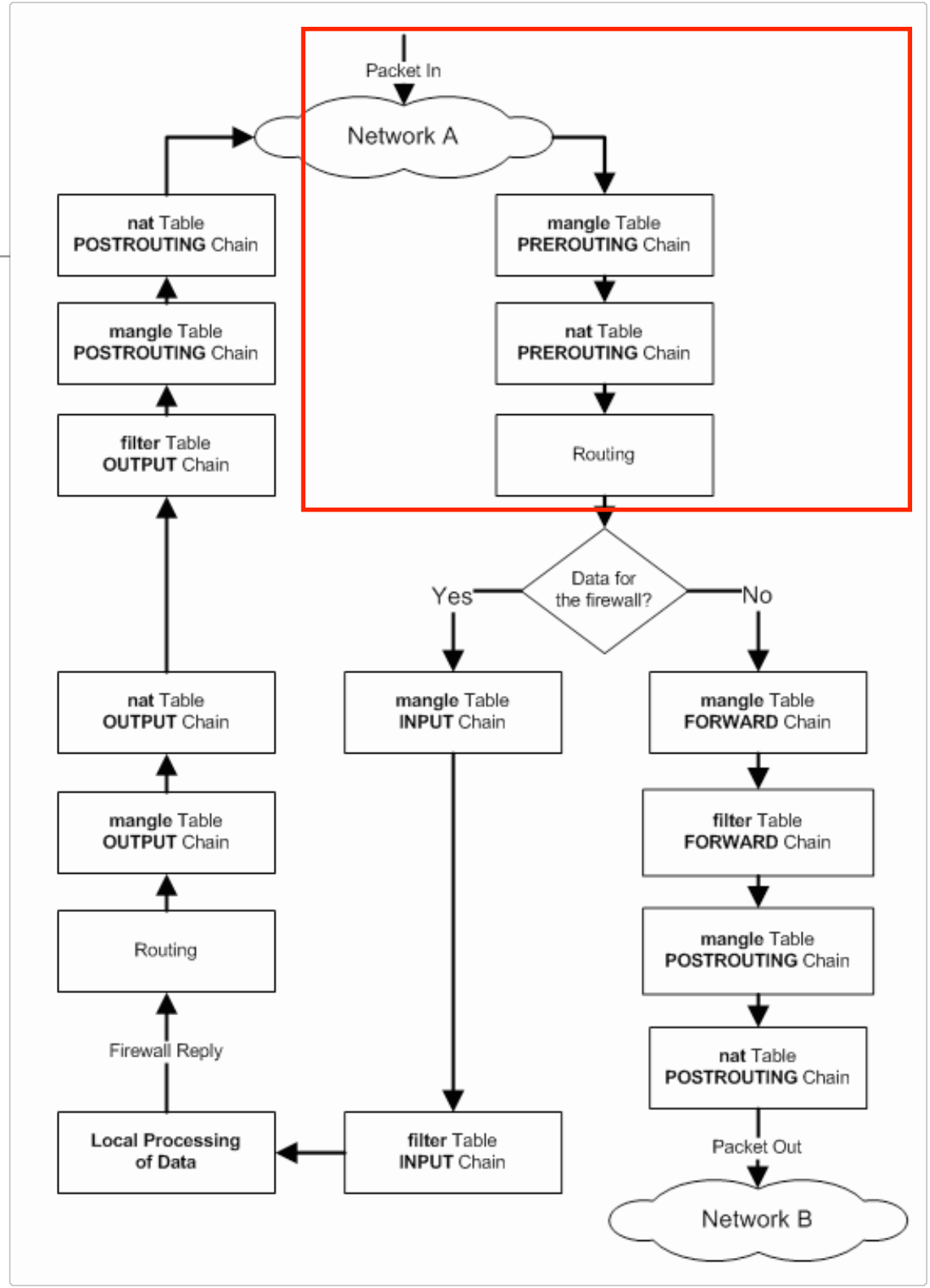
Mangle	TCP header modification	`PREROUTING`
		`POSTROUTING`
		`OUTPUT`
		`INPUT`
		`FORWARD`
			Modification of the TCP packet quality of service bits before routing occurs. (Rarely used in SOHO environments)

4.L Procesamiento de paquetes en iptables

- Debe especificarse la tabla y la cadena para cada regla del firewall que se cree. No obstante, como la mayoría de las reglas se dedicarán a filtrado, se considera que las reglas pertenecen a la tabla de filtrado si la tabla no se indica.

4.L Flujo de paquetes

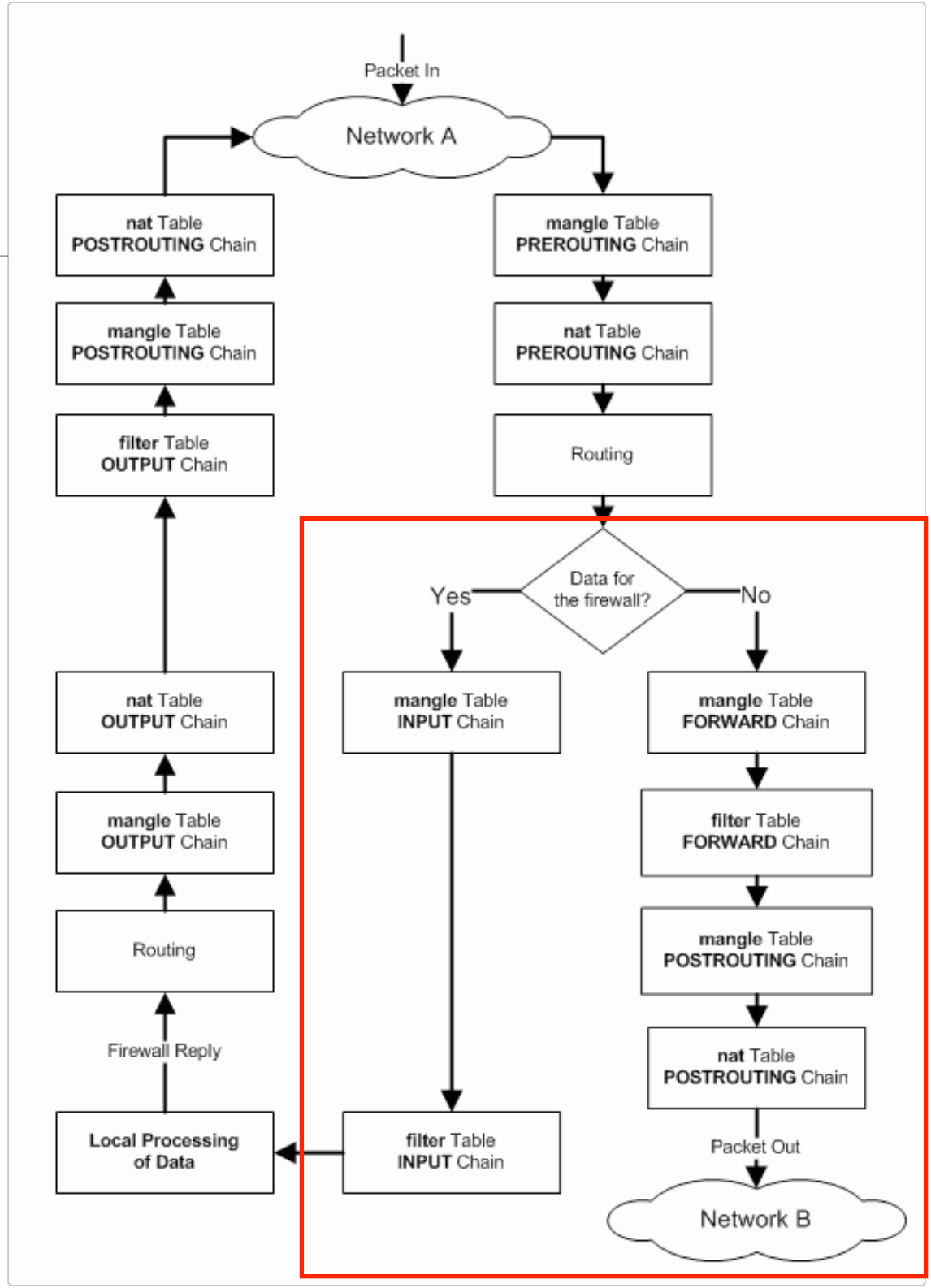
- El paquete se examina primero por la tabla mangle/cadena prerouting
- A continuación se inspecciona por las reglas en la tabla NAT/cadena prerouting, para ver si el paquete necesita DNAT (cambiarle la dirección de destino)
- El paquete se enruta



4.L Flujo de paquetes

- Si el paquete se dirige a una red protegida, se filtra por las reglas en la cadena FORWARD de la tabla de filtrado, y si es necesario se le aplica SNAT en la cadena POSTROUTING antes de llegar a la red B

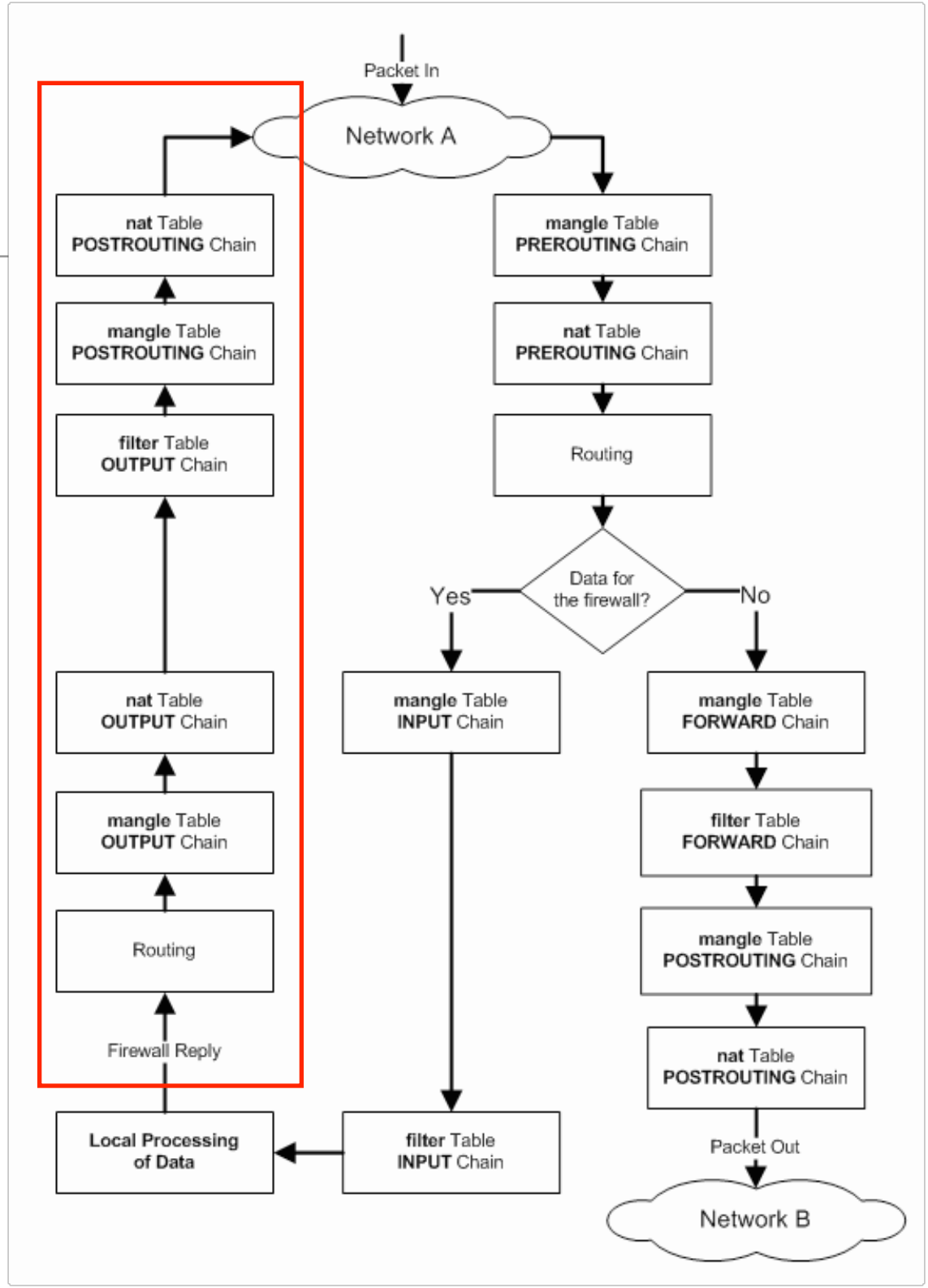
- Si el paquete se destina al mismo firewall, pasa por la cadena INPUT de la tabla de filtrado, y si pasa los tests entonces se procesa por la aplicación correspondiente



4.L Flujo de paquetes

- Si hay respuesta al paquete, esta se enruta, y se aplican las reglas de la cadena OUTPUT para determinar si el paquete debe filtrarse.

- Por último, antes de devolver el paquete a Internet, se aplica SNAT por la cadena de POSTROUTING de NAT



4.L Objetivos de las reglas

- Cada regla del firewall inspecciona cada paquete IP para determinar si es el objeto de alguna operación.

target	Desciption	Most Common Options
ACCEPT	▪ iptables stops further processing. ▪ The packet is handed over to the end application or the operating system for processing	N/A
DROP	▪ iptables stops further processing. ▪ The packet is blocked	N/A
LOG	▪ The packet information is sent to the syslog daemon for logging ▪ iptables continues processing with the next rule in the table ▪ As you can't log and drop at the same time, it is common to have two similar rules in sequence. The first will log the packet, the second will drop it.	`--log-prefix "string"` Tells iptables to prefix all log messages with a user defined string. Frequently used to tell why the logged packet was dropped

4.L Objetivos de las reglas

REJECT	▪ Works like the DROP target, but will also return an error message to the host sending the packet that the packet was blocked	`--reject-with qualifier`
		The qualifier tells what type of reject message is returned. Qualifiers include:
		`icmp-port-unreachable (default)`
		`icmp-net-unreachable`
		`icmp-host-unreachable`
		`icmp-proto-unreachable`
		`icmp-net-prohibited`
		`icmp-host-prohibited`
		`tcp-reset`
		`echo-reply`

DNAT	▪ Used to do **destination network address translation**. ie. rewriting the destination IP address of the packet	`--to-destination ipaddress` Tells iptables what the destination IP address should be
SNAT	▪ Used to do **source network address translation** rewriting the source IP address of the packet ▪ The source IP address is user defined	`--to-source <address>[-<address>][:<port>-<port>]` Specifies the source IP address and ports to be used by SNAT.

4.L Objetivos de las reglas

MASQUERADE	▪ Used to do Source Network Address Translation. ▪ By default the source IP address is the same as that used by the firewall's interface	`[--to-ports <port>[-<port>]]` Specifies the range of source ports to which the original source port can be mapped.

4.L Criterios

Iptables command Switch	Desciption
`-t <-table->`	If you don't specify a table, then the `filter` table is assumed. As discussed before, the possible built-in tables include: filter, nat, mangle
`-j <target>`	Jump to the specified target chain when the packet matches the current rule.
`-A`	Append rule to end of a chain
`-F`	Flush. Deletes all the rules in the selected table
`-p <protocol-type>`	Match protocol. Types include, icmp, tcp, udp, and all
`-s <ip-address>`	Match source IP address
`-d <ip-address>`	Match destination IP address
`-i <interface-name>`	Match "input" interface on which the packet enters.
`-o <interface-name>`	Match "output" interface on which the packet exits

iptables -A INPUT -s 0/0 -i eth0 -d 192.168.1.1 -p TCP -j ACCEPT

Acepta paquetes TCP del interfaz eth0 desde cualquier dirección IP destinados a la dirección 192.168.1.1 (la representación 0/0 significa “cualquiera”)

4.L Criterios

Switch	Description
`-p tcp --sport <port>`	TCP source port. Can be a single value or a range in the format: *start-port-number:end-port-number*
`-p tcp --dport <port>`	TCP destination port. Can be a single value or a range in the format: *starting-port:ending-port*
`-p tcp --syn`	Used to identify a new TCP connection request. ! --syn means, not a new connection request
`-p udp --sport <port>`	UDP source port. Can be a single value or a range in the format: *starting-port:ending-port*
`-p udp --dport <port>`	UDP destination port. Can be a single value or a range in the format: *starting-port:ending-port*

iptables -A FORWARD -s 0/0 -i eth0 -d 192.168.1.58 -o eth1 -p TCP --sport 1024:65535 --dport 80 -j ACCEPT

Acepta paquetes TCP cuando entran por el interfaz eth0 desde cualquier dirección IP y se destinan a la dirección IP 192.168.1.58, alcanzable por el interfaz eth1. El puerto de origen está en el rango 1024:65535 y el destino es el puerto 80

4.L NAT con iptables

- iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE
- Regla para la tabla NAT
- Se añade a la cadena POSTROUTING
- Para el interfaz eth0
- Se salta al objetivo MASQUERADE: (SNAT) La dirección IP del campo origen del paquete se reemplaza por la del interfaz

4.L firewalld: instalación

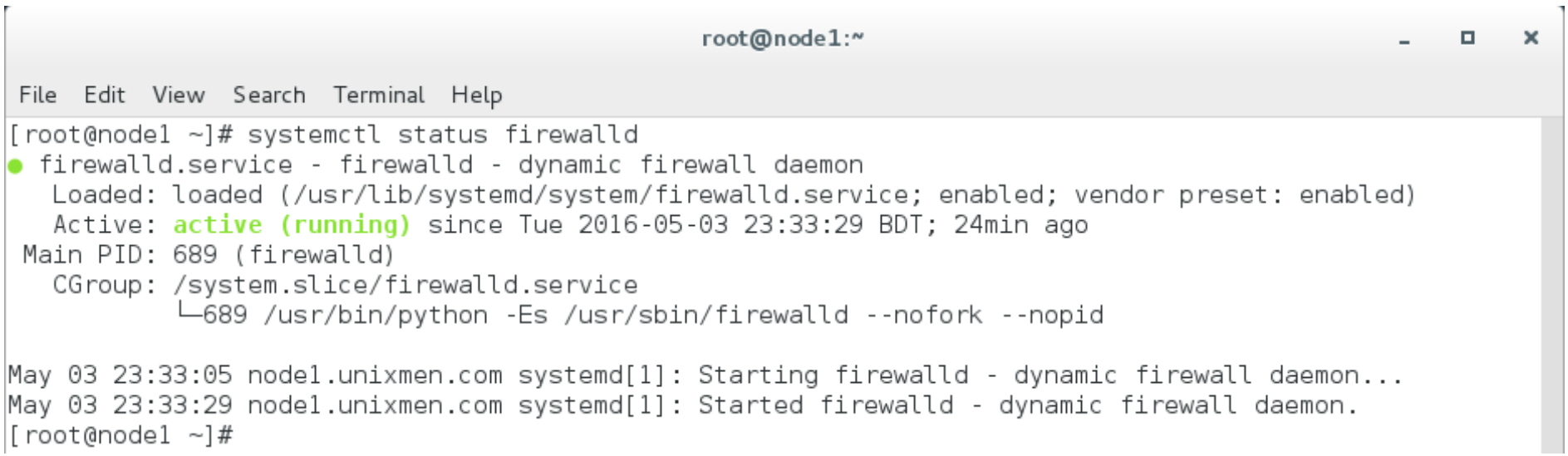
- firewalld es un cortafuegos que soporta zonas con diferente confianza en lugar de cadenas y reglas

- instalación:

```
sudo systemctl stop iptables

sudo systemctl mask iptables

sudo yum install firewalld firewall-config -y
```

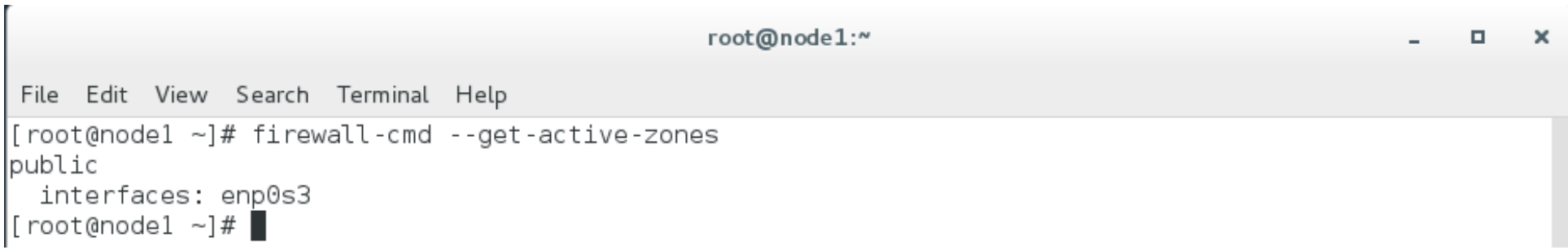
4.L firewalld: status

- `sudo systemctl status firewalld`



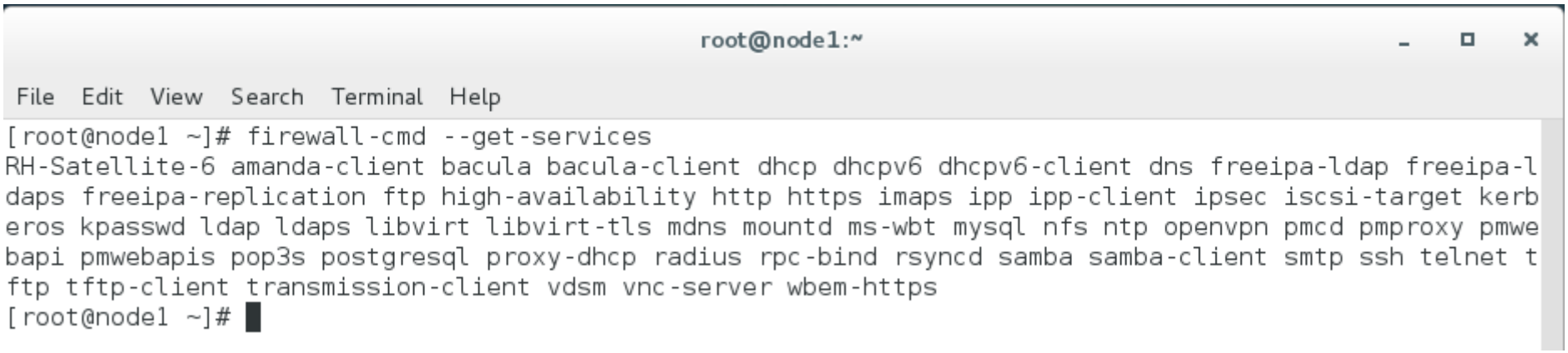
4.L firewalld: consultar zonas

```
sudo firewall-cmd –get-active-zones

sudo firewall-cmd –get-zones

sudo firewall-cmd –get-default-zone
```



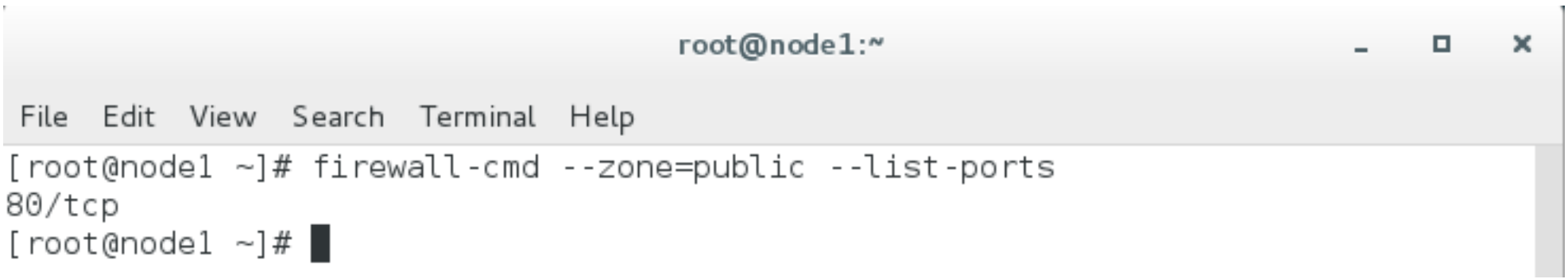
4.L firewalld: consultar servicios

```
sudo firewall-cmd –get-services
```



4.L firewalld: abrir un puerto

```
sudo firewall-cmd --permanent --zone=public --add-
port=80/tcp

sudo firewall-cmd –reload

sudo firewall-cmd --zone=public --list-ports
```



4.L firewalld: cerrar un puerto, añadir/eliminar un servicio

```
sudo firewall-cmd --zone=public --remove-port=80/tcp

sudo firewall-cmd --zone=public --add-service=ftp

sudo firewall-cmd --zone=public --remove-service=ftp
```

4.L firewalld: bloquear todos los paquetes entrantes y salientes, bloquear una IP

```
sudo firewall-cmd --panic-on

sudo firewall-cmd --panic-off

sudo firewall-cmd --zone=public --add-rich-rule='rule
family="ipv4" source address="192.168.1.4" reject'
```

4.L firewalld: NAT

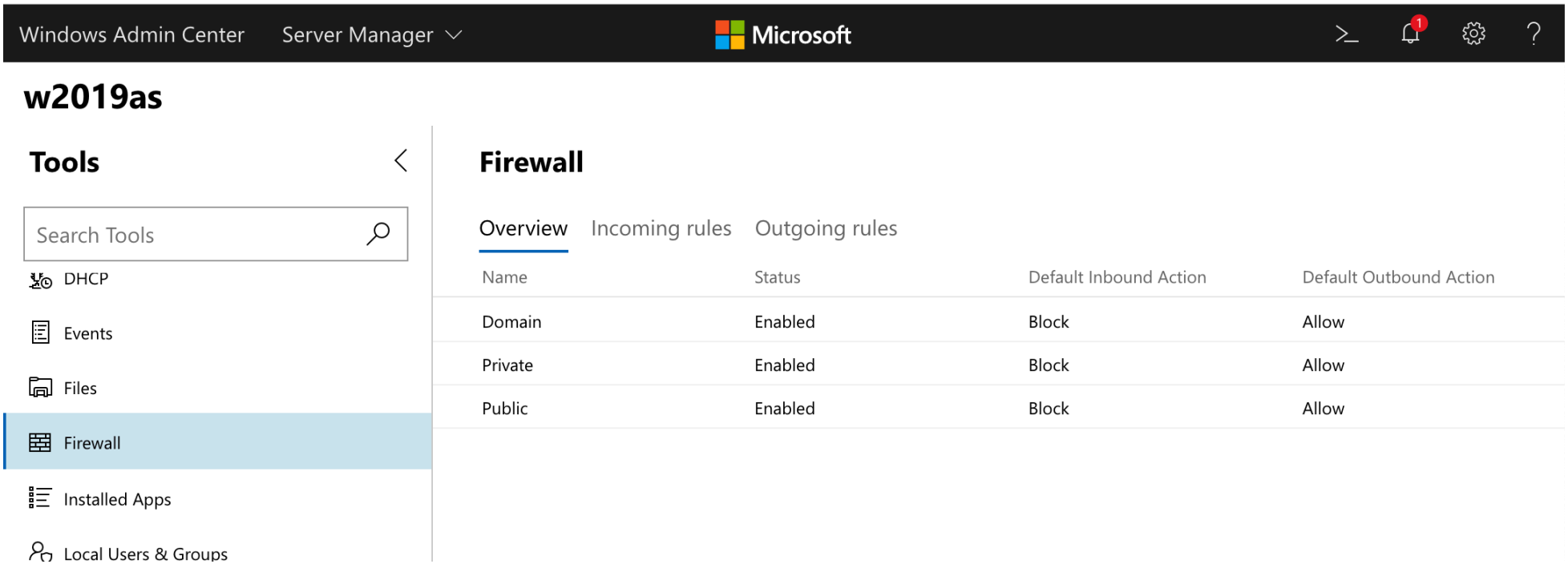
```
firewall-cmd --direct --add-rule ipv4 nat POSTROUTING 0 -
o eth1 -j MASQUERADE

firewall-cmd --direct --add-rule ipv4 filter FORWARD 0 -i
eth2 -o eth1 -j ACCEPT

firewall-cmd --direct --add-rule ipv4 filter FORWARD 0 -i
eth1 -o eth2 -m state --state RELATED,ESTABLISHED -j
ACCEPT
```

Firewall Windows 2019

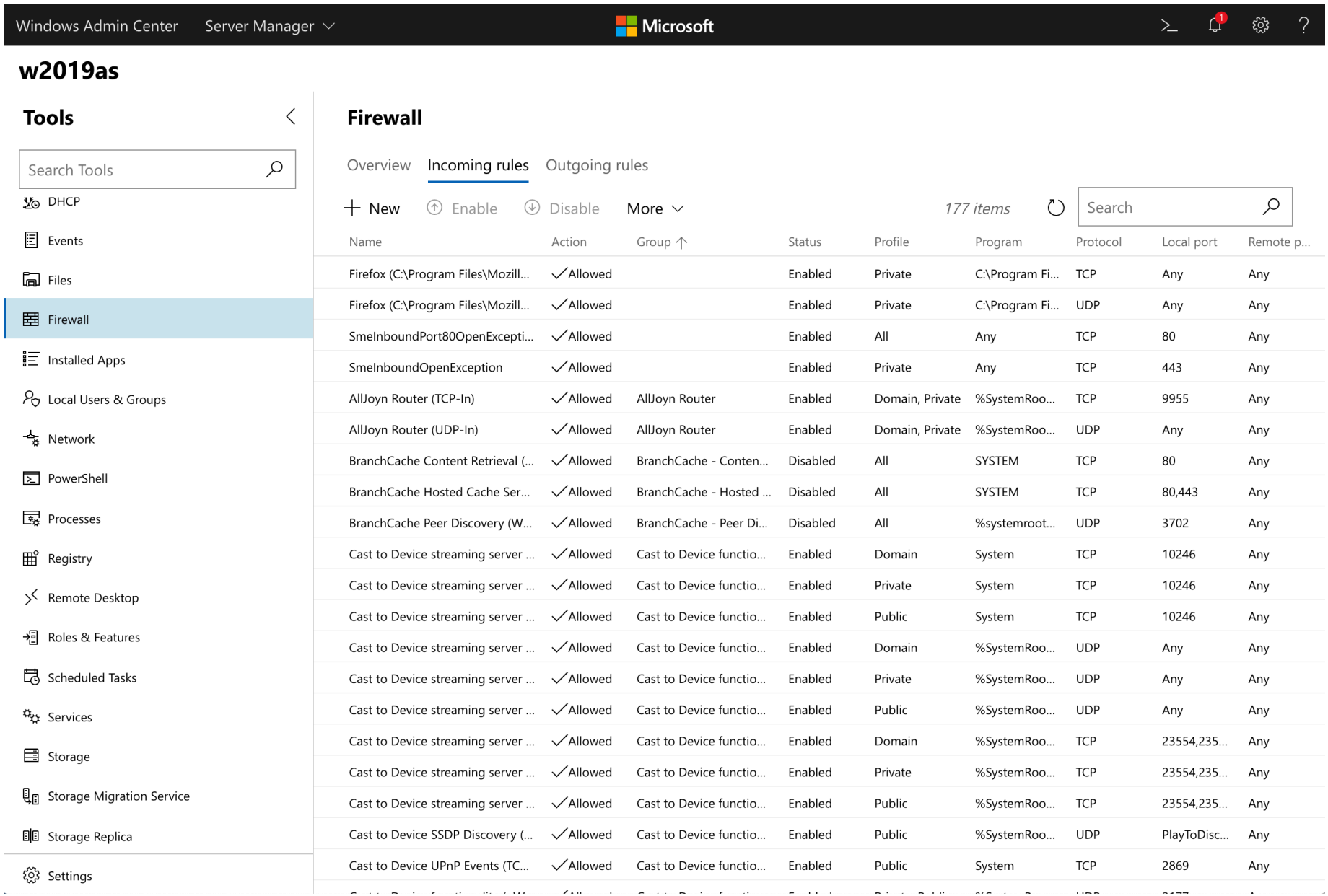
- El firewall de windows está integrado con las utilidades de administración del servidor y con el asistente para los roles del servidor; por ejemplo, si un administrador hace que el sistema sea un servidor de archivos, los puertos y los protocolos necesarios para acceder al servicio se abren sin necesidad de configurar el firewall a mano.

- Además, pueden crearse manualmente reglas para:
  - **Programas:** se permite el acceso a determinado programa (conexiones salientes)
  - **Puertos**
  - **Predefinidas:** Para permitir acceso a determinado servicio (DFS, HTTP, etc.)
  - **A la medida**

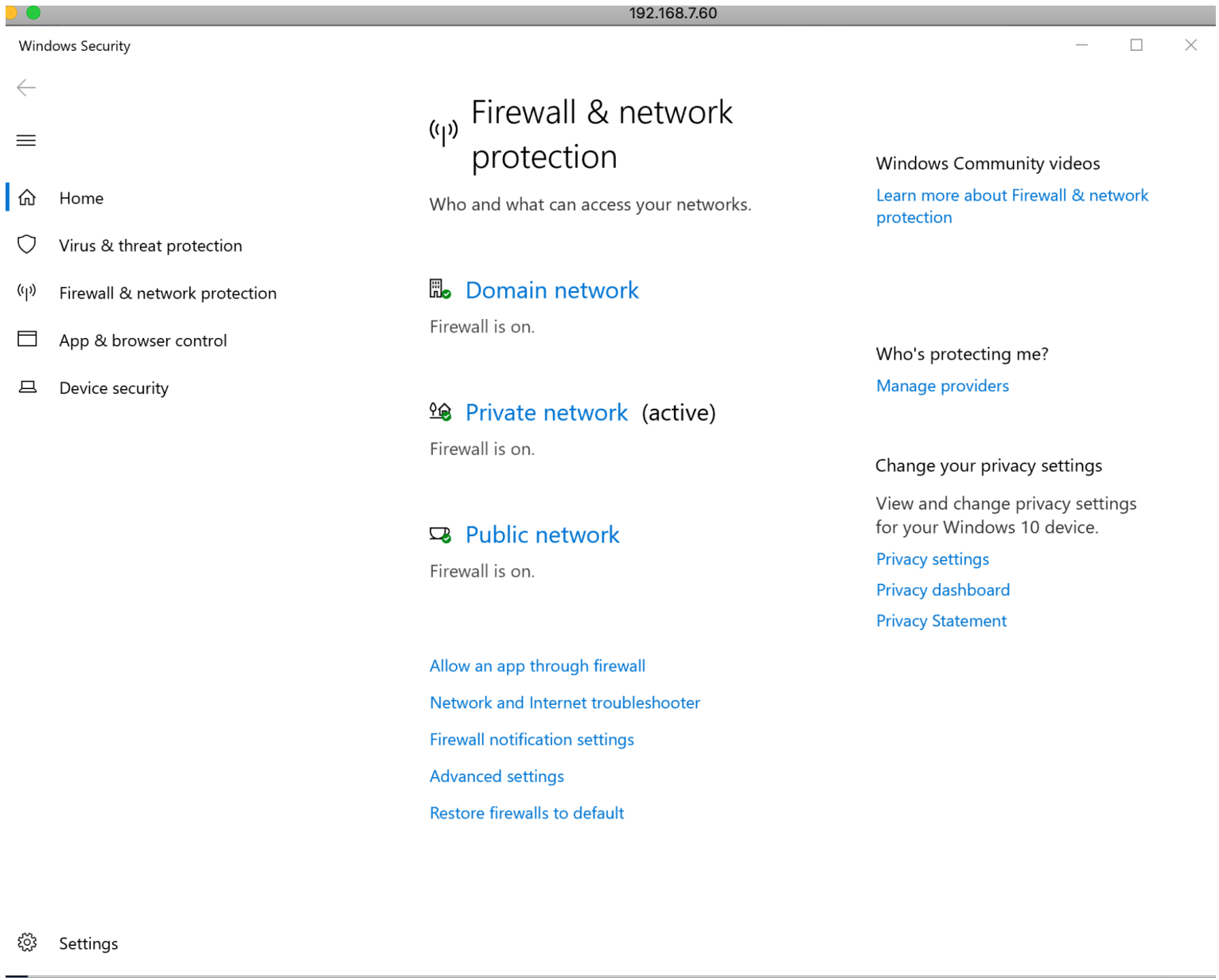
Firewall Windows 2019



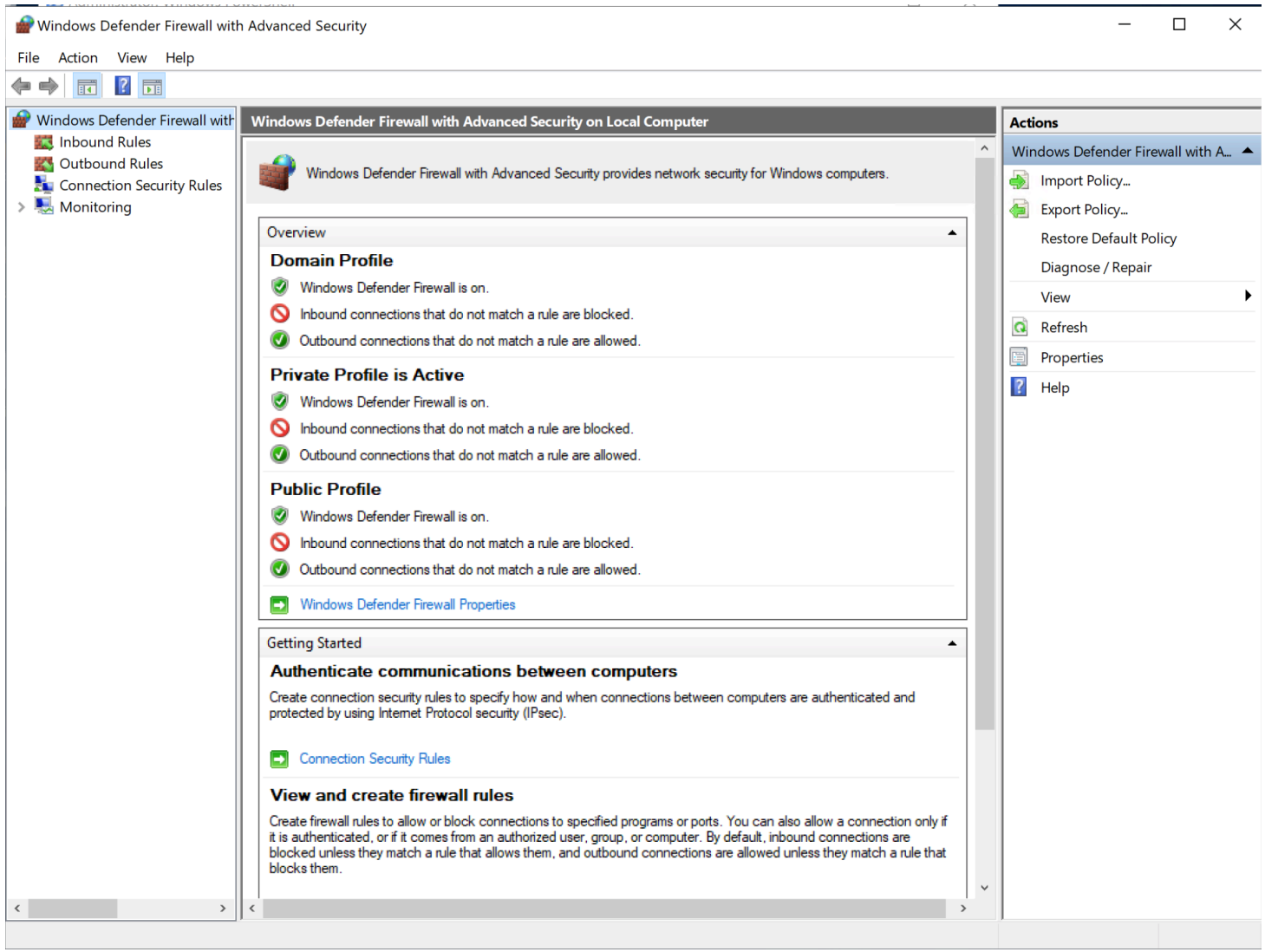
Firewall Windows 2019



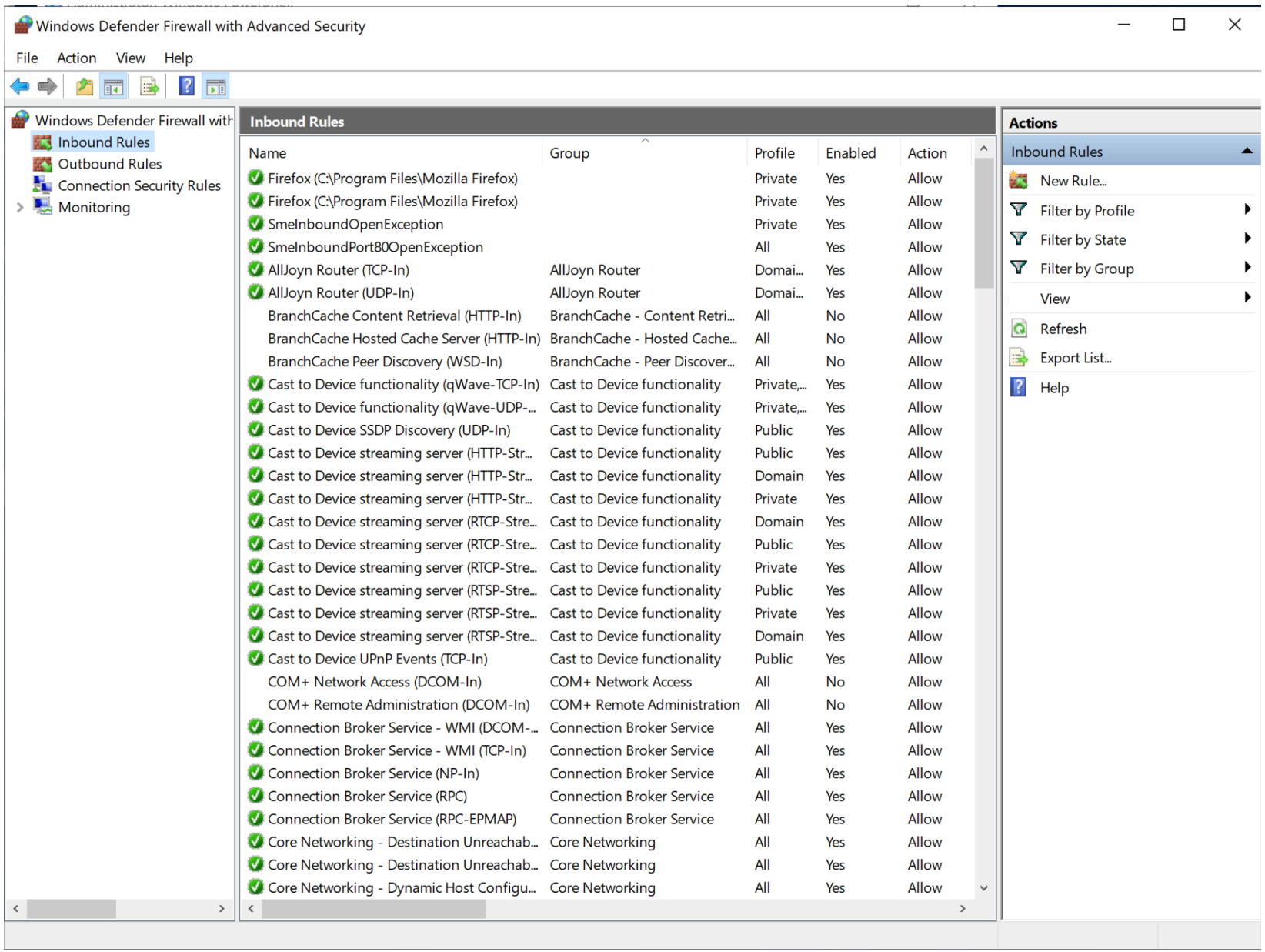
Firewall Windows 2019



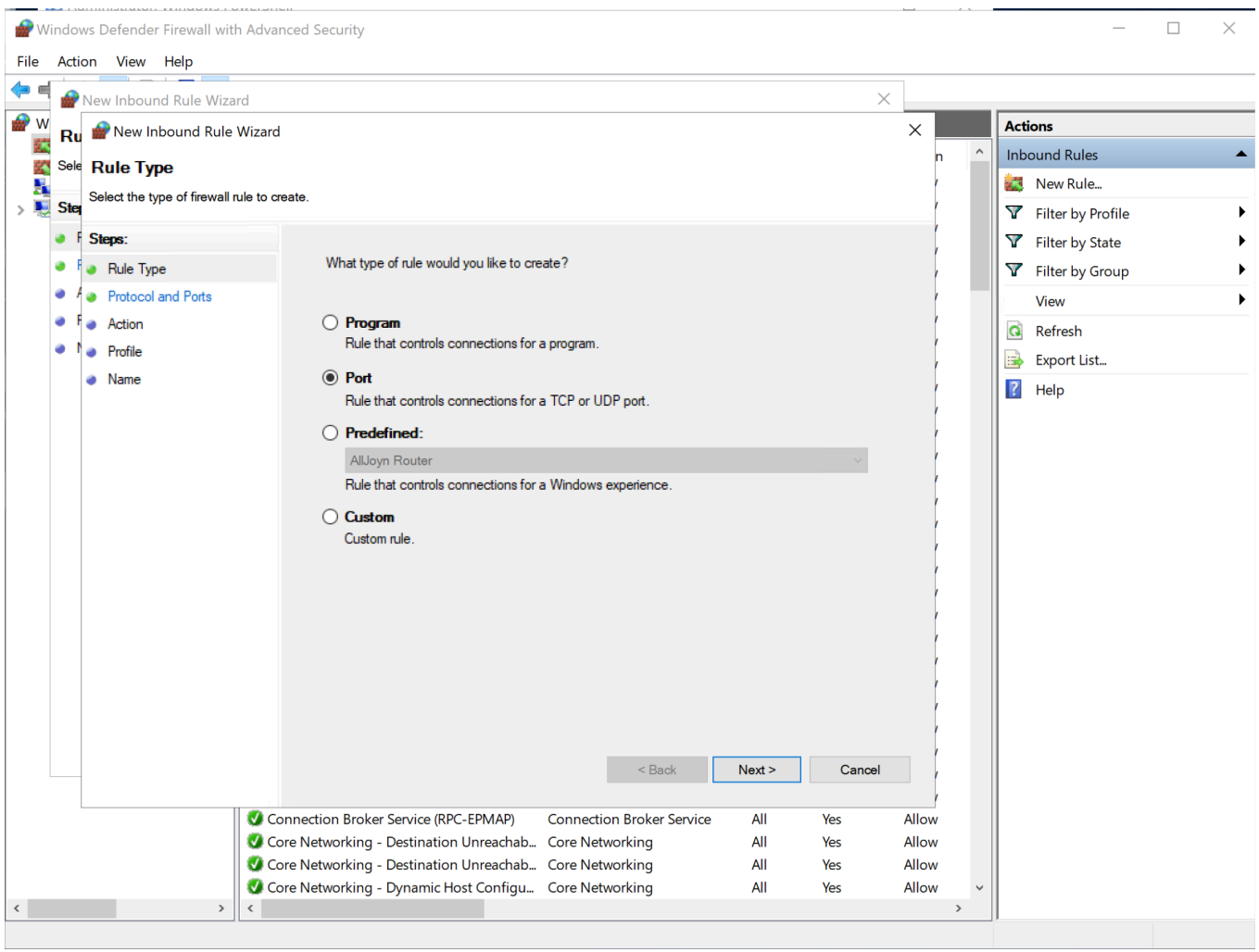
Firewall Windows 2019



Consola de administración del firewall windows

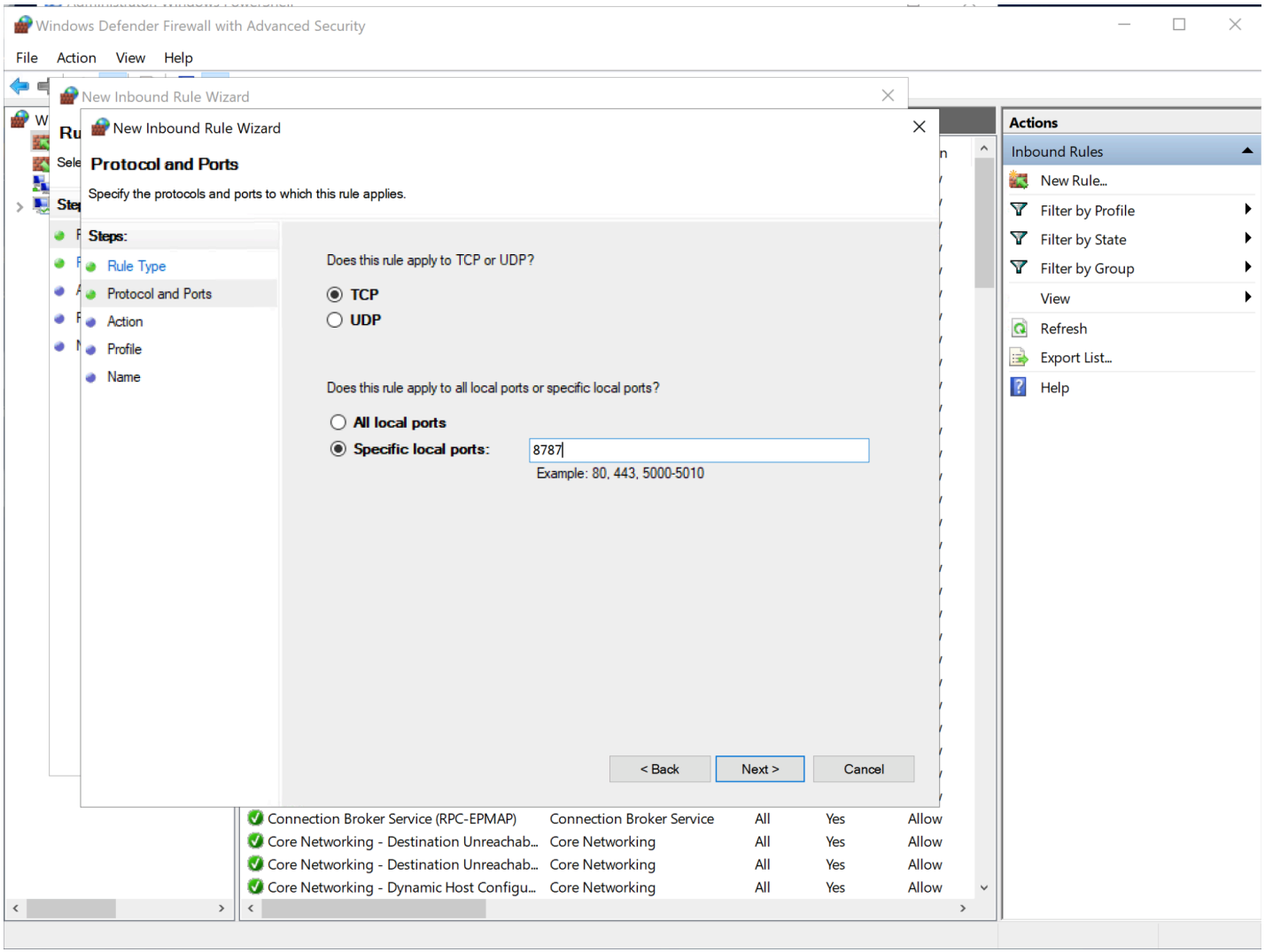


Creación de reglas (1)



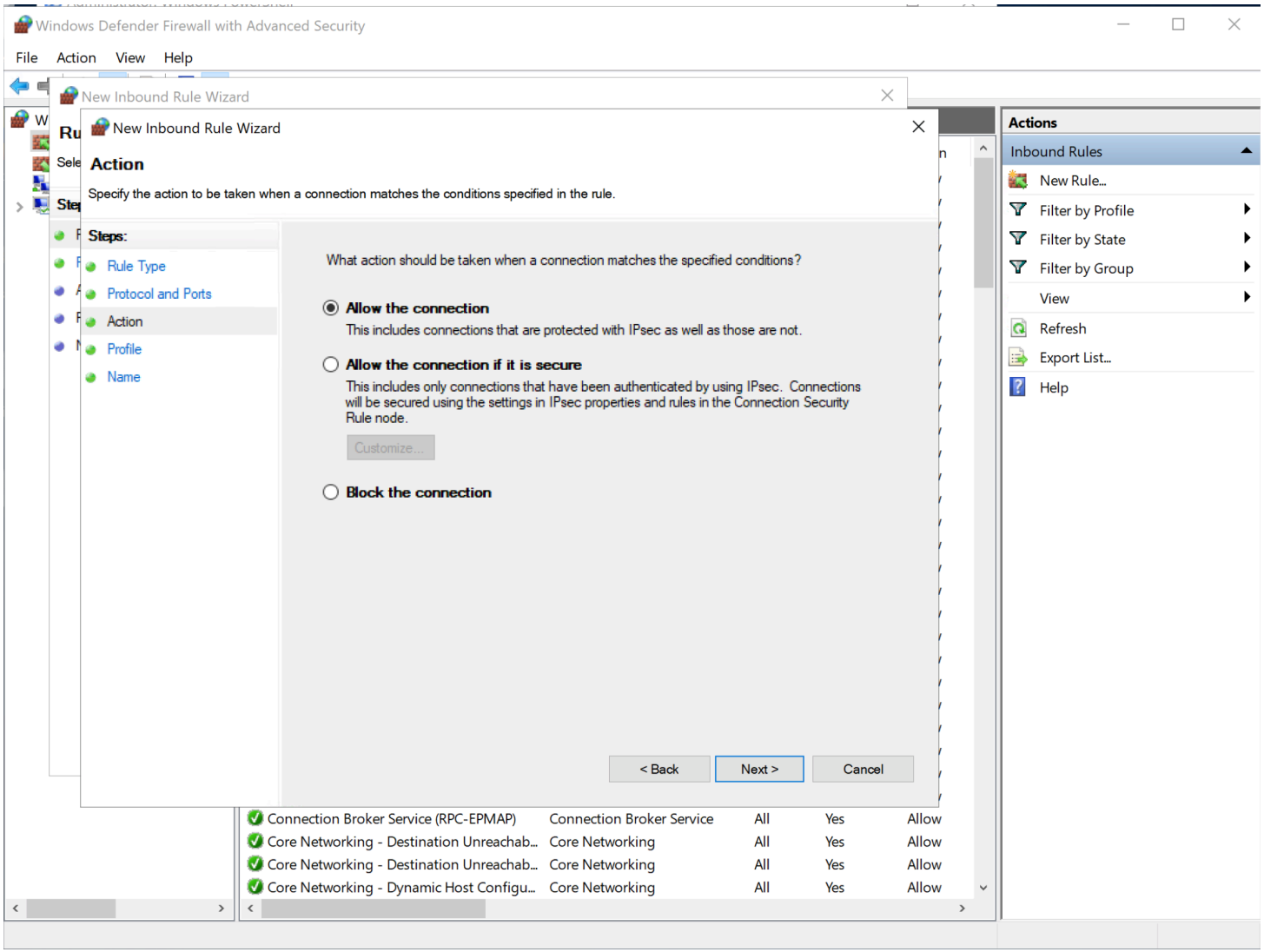
- Asistente para creación de nueva regla
- Ejemplo: acceso entrante a TCP 8787
- Se crea una regla de tipo “Puerto”

Creación de reglas (2)



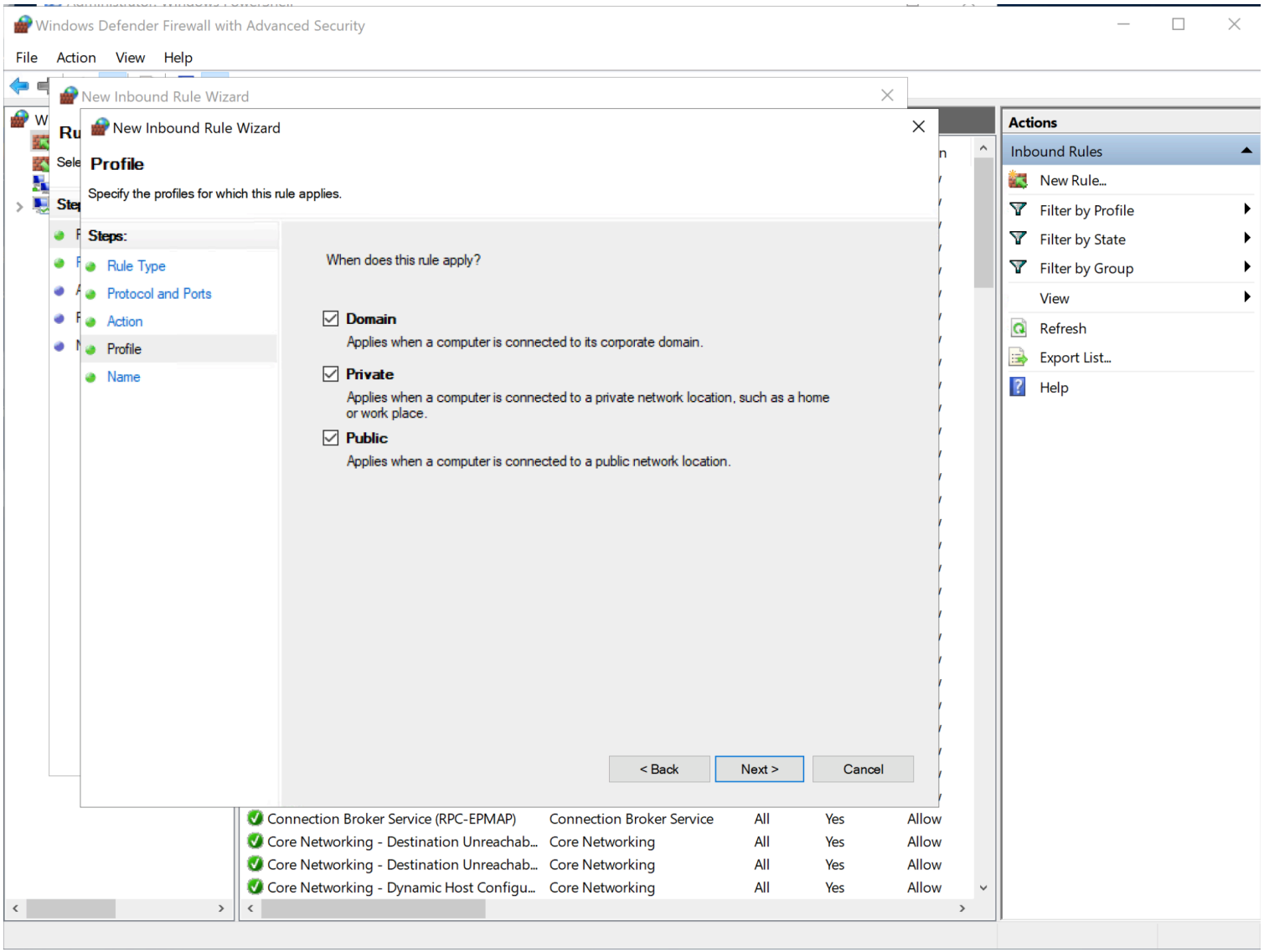
- En “protocolos y puertos” se elige TCP y el puerto 8787
- Se activa la regla en la siguiente página

Creación de reglas (3)



- Se indica si la conexión tiene que ser segura para poder admitirse

Creación de reglas (4)



- Se activan las tres casillas en la siguiente pantalla, para delimitar el tipo de redes a las que se aplica la regla
- Se le da un nombre descriptivo a la regla, y se finaliza.