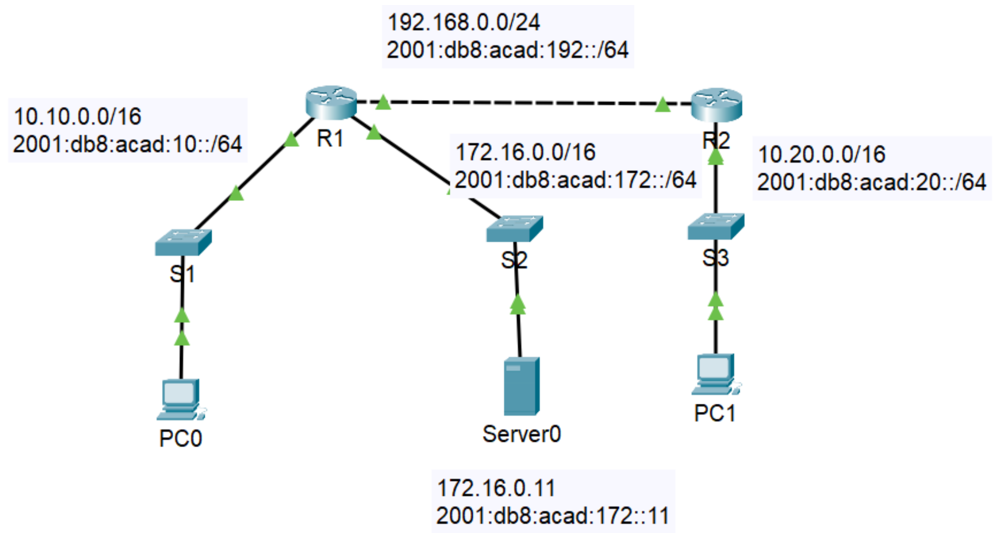
DHCP and NAT: Working with IP Addresses

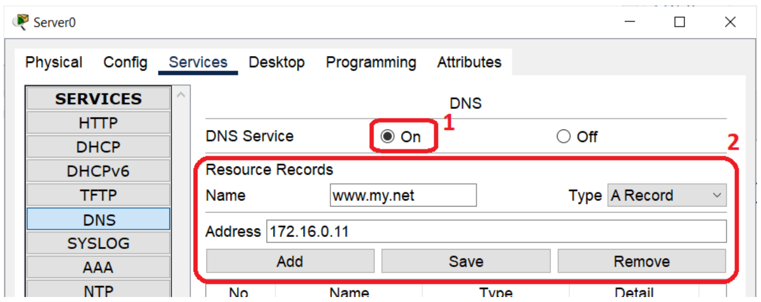
DNS server

Task:

On the web, there is a pre-prepared file for DHCP.



Start the DNS server on Server-PT and add two RRs: the web server www.my.net with the both IPv4 and IPv6 addresses.



DNS Service [✓ On] Off

Resource Records

Name www.my.net Type A Record

Address 172.16.0.11

Dynamic Allocation of IPv4 Addresses

How to Set DHCPv4 on Router

The pool, which is the range of addresses that will be assigned to clients, needs to be determined. The range is determined by the masked network address, but we may want to exclude some addresses from this range from allocation (for example, because they belong to the router/gateway itself and possibly some servers or various network devices).

For example, if we want to allocate the range 10.10.0.0/16, but we want to exclude the first 8 addresses and then one more:

```
ip dhcp excluded-address 10.10.0.1 10.10.0.8        ; address range to exclude
ip dhcp excluded-address 10.10.0.254                ; one address excluded

ip dhcp pool LAN-1                                  ; pool for clients, named LAN-1
      network 10.10.0.0 255.255.0.0                 ; range for allocation
      default-router 10.10.0.1
      dns-server xxxxxxx              ; the DNS server address, optional
      domain-name xxxxxxx             ; domain name, also optional
      lease 2                         ; lease time in days, optional
```

As we can see, the definition of the DHCP pool takes us to the sub-configuration mode, in which we specify the parameters: the allocated address range, the gateway address, and then other (optional) parameters such as DNS server address, domain name, lease time (time of address allocation from the pool), etc.

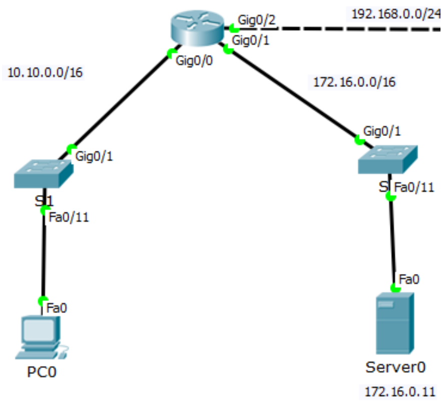
There are some `show` commands:

```
show run | section dhcp
show ip dhcp pool
show ip dhcp binding      ; allocation table (DHCP state, binding between IP and MAC)
show ip dhcp server statistics
show ip dhcp conflict
```

etc., use a question mark.

Multiple DHCP pools can be defined on a single router because each router interface "sees" into a different network, and multiple such networks can use private addresses. We need a different range for each network.

Task:



Continue with the previous file. On router R1, configure the pool for the address range 10.10.0.0/16 according to the above instructions (the address range and the range of excluded addresses are sufficient).

Then see if PC0 already has an address and gateway, and which address it actually got. See if you can ping the server from PC0, and use a web browser on PC0.

Why do you think it is not necessary to deploy a defined pool on an interface? If multiple pools were defined, how would the router know from which to assign the address to the suplicant (client)?

Next, on the router, check the address allocation status (i.e. the dynamic address allocation table), or try other show commands.

Helper

If we have multiple routers with different private networks in the company and only one of them will act as a DHCP server, then the other routers must act as a *relay agent*. A relay agent forwards a broadcast of a given type to another network (this would not be possible otherwise, broadcasts terminate at the network boundary). If we want a router to act as a relay agent, we run a *helper* on it. Helpers can be for various protocols/services, depending on what runs on the device with a certain address.

Procedure: on the future relay agent, specifically the interface behind which the address supplicants are located, we set up a helper that forwards to the address of the server providing the service (we put there the address under which our "agent" can see the service provider).

```
int g0/xxx
      ip helper-address 192.168.0.1
```

The helper settings can be verified for example in the running-config or in the interface parameters list:

```
sh ip int g0/xxxx
```

Task:

Router R1: create a second pool for addresses 10.20.0.0/16, again remove the first 8 addresses.

Router R2: run a helper to allow devices behind interface g0/0 to obtain an address from router R1.

Test if both the server and PC0 can be pinged from PC1.

Stateless SLAAC for IPv6

In the IPv6 world, we have a bit more ways to get an IP address. The easiest for end devices is SLAAC (Stateless Autoconfiguration), which is the default on Cisco routers – we just need to get IPv6 up:

```
ipv6 unicast-routing
```

And of course assign an IPv6 address on that router interface.

The router itself sends an RA message (Router Advertisement) from which clients learn the network address, gateway address and possibly other parameters. They then calculate the host part of the address themselves.

The other possibilities are stateful and stateless DHCPv6. Stateful works the same way as in IPv4 (it allocates addresses), while stateless is "halfway": the address (and gateway) is handled by the client using the router, but the rest of the information is obtained from the DHCP server (e.g. DNS server address, domain name, etc.). The client is informed of the specific network settings in the RA message from the router, which contains two important flags: the M (managed flag) and the O (other config flag).

	SLAAC	stateful DHCPv6	stateless DHCPv6
Flags settings:	M = 0, O = 0	M = 1	M = 0, O = 1

Task:

We continue with the previous file. Start IPv6 on router R1, assign the following addresses on g0/0:

- 2001:db8:acad:10::1/64 as global unicast
- fe80::1 as link-local

For g0/1:

- 2001:db8:acad:172::1/64 as global unicast
- fe80::1 as link-local

Check what address PC0 has calculated.

Obtaining DNS server Name over SLAAC

SLAAC has one problem: how does the client find out about DNS server addresses? The option exists (according to RFC 8106), it's the command

```
ipv6 nd ra dns server ip-address
```

and setting M and O flags to 0 (if they are not). But some routers do not support it.

Setting Stateless DHCPv6 on the Router

There are two additional settings that need to be made on the router in addition to what was mentioned for SLAAC:

- create a pool, in which the address range will not be present, but only those other parameters that clients will request (DNS server, etc.),
- deploy the pool on that interface and set the other config flag.

```
ipv6 dhcp pool STATELESS1
       dns-server 2001:db8:acad:172::11
       domain-name my.net
       exit
int g0/1
       ipv6 addr 2001:db8:acad:172::1/64
       ipv6 dhcp server STATELESS1
       ipv6 nd other-config-flag
```

Task:

Configure stateless DHCPv6 on R1 for the client network 2001:db8:acad:10::/64. Change the settings on PC0. Check the function of the DNS server using the name address of the web server.

Stateful DHCPv6

The procedure is similar to the previous example, but when creating the pool, we also define the IP address range and set the managed config flag on the interface:

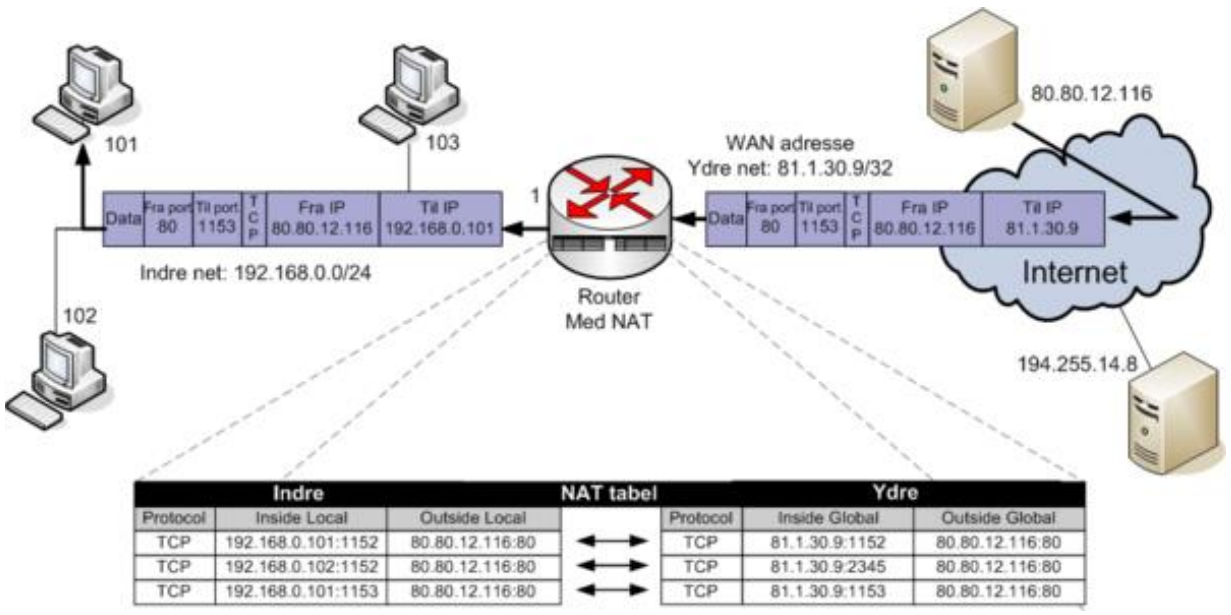
```
ipv6 dhcp pool STATEFUL1
	address prefix 2001:db8:.......::/64 lifetime infinite
	dns-server xxxxxx
	domain-name xxxxx
	exit
int g0/1
	ipv6 addr xxxxx
	ipv6 dhcp server STATEFUL1
	ipv6 nd managed-config-flag
```

Note how the address range is determined – the keyword is slightly different, and the lease time of the address is also part of the range determination.

Network Address Translation

NAT (Network Address Translation) can be either static or dynamic. While with static we "manually" determine which address is translated to which address, with dynamic we determine which range is translated to which range and the router itself controls the assignment. It stores the current state of the translation in the dynamic address translation table, which is why we talk about stateful NAT.

Dynamic NAT is almost exclusively done as PAT (Port Address Translation): a group of internal addresses are translated to a single "external" address, but differentiated by dynamic port number (clients use dynamic port numbers). And if there is a match in port number with another local device, the port number is translated in addition to the address. The following figure shows the principle:


80.80.12.116
WAN adresse
Ydre net: 81.1.30.9/32
101
103
102
1
Data | Fra port 80 | Til port 1153 | TCP | Fra IP 80.80.12.116 | Til IP 192.168.0.101
Data | Fra port 80 | Til port 1153 | TCP | Fra IP 80.80.12.116 | Til IP 81.1.30.9
Indre net: 192.168.0.0/24
Router
Med NAT
Internet
194.255.14.8

Indre			NAT tabel	Ydre		
Protocol	Inside Local	Outside Local		Protocol	Inside Global	Outside Global
TCP	192.168.0.101:1152	80.80.12.116:80	⟷	TCP	81.1.30.9:1152	80.80.12.116:80
TCP	192.168.0.102:1152	80.80.12.116:80	⟷	TCP	81.1.30.9:2345	80.80.12.116:80
TCP	192.168.0.101:1153	80.80.12.116:80	⟷	TCP	81.1.30.9:1153	80.80.12.116:80

Source: https://mars.merhot.dk/w/index.php/NAT_Cisco_IOS

The area inside the network is "inside", the area behind the router is "outside". Addresses marked "local" translate to addresses marked "global", although in reality these may not be globally valid either, it's just a naming convention.

Task:

Create a dynamic NAT on router R1 for network 10.10.0.0/16. First, we use ACLs to determine which addresses should actually be translated:

```
access-list 1 permit 10.10.0.0 0.0.255.255
```

Now there are two different ways to proceed. Either we translate to multiple addresses, and thus a pool of addresses would need to be defined, or we choose a simple PAT with a single address for translation. The latter is simpler and we make do with a single address on the outgoing interface, so we choose the second option. As a parameter, we specify the access list we created beforehand, and then the interface to be translated and whose IP address we will use.

```
ip nat inside source list 1 interface g0/2 overload
```

The keyword *overload* just means that client addresses will not be translated to the address pool, but to the address of the specified interface.

The only thing left to do is to make changes on the relevant interfaces, i.e. tell the router which interfaces are "inside" and "outside".

```
interface g0/0
      ip nat inside
interface g0/2
      ip nat outside
```

There are also `show` commands for NAT:

```
show ip nat translations
show ip nat statistics
clear ip nat statistics
debug ip nat
```