Computer Networks *(Redes de Computadores - RCOMP)* – 2023/2024

# ***Laboratory Class 05 (PL05 – 3 hours)***

VLAN support in Cisco routers – switching module or sub interfaces. Packet Tracer practice. DHCP service configuration in Cisco routers. Packet Tracer activities. Simplifying IPv4 routing tables.

## 1. VLAN support in Cisco routers

We already know end nodes can also be VLAN-aware and handle IEEE 802.1q frames, this allows a single network interface to operate as several independent interfaces, each one connected to a different VLAN. End nodes, operating at layer three, can then assign a different network IP address to each interface, and thus use all VLANs at the same time.

### 1.1. Switching module for Cisco routers

This is, in fact, a workaround for connecting VLANs to Cisco routers, not a generic definitive solution. It's based on the hardware module shown in Figure 1, that contains a VLAN-aware layer two switch with four Ethernet ports operating at 100 Mbps.

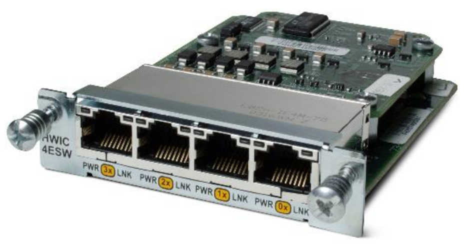

*Figure 1 - Cisco HWIC-4ESW module*

Once plugged into the router slot (Figure 2) every VLAN defined on it will be known to the router.

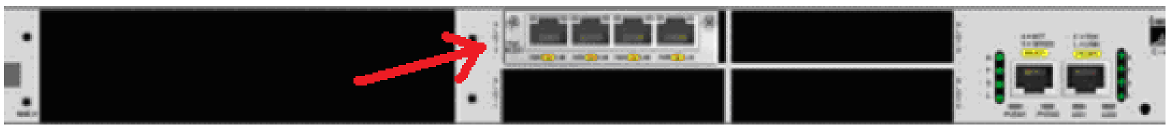

*Figure 2 - Cisco 2811 router with a HWIC-4ESW module*

This is a four ports layer two switch, not four independent network interfaces for the router, the router will not be able to use these ports directly, however, it will be able to use every VLAN defined on the switch as if it was an independent network interface.

Packet Tracer forms can be used to configure the switch VLANs as with any other switch. Afterwards, new interfaces become available to the router, they can be referred to as **vlan ID**, where ID represents the used VLANID.

For instance, if we define a VLAN with VLANID=25 on the switch and assign it to some switch ports, we can then connect the router to that VLAN by using the following commands:

```
(config) interface vlan 25
(config-if) ip address 192.168.0.10 255.255.255.0
```

The router will be using IP address 192.168.0.10/24 on that VLAN.

### 1.2. Sub-interfaces

Although the switching hardware module can do the trick, it requires additional hardware. And the same goal can be achieved without additional hardware. This is a more generic solution that may be used on any Cisco router.

Notice added hardware may have restrictions on its own, for instance, the described hardware module works only at 100 Mbps and is not available for all router models.

Cisco IOS (Cisco devices' operating system) and similar systems allow the creation of sub-interfaces over existing physical interfaces. Sub-interfaces overlap the physical interface and multiple sub-interfaces over the same physical interface overlap each other. Yet, each sub-interface can have its own IP address and be independent of others.

Sub-interfaces are identified by the physical interface name with a numeric suffix separated by a dot, the first sub-interface will have suffix one and so on. For instance, interfaces **FastEthernet0/0.1**, **FastEthernet0/0.2**, and **FastEthernet0/0.3** are **FastEthernet0/0** sub-interfaces.

To make sub-interfaces match VLAN's, we can use the **encapsulation dot1q** command to add VLAN IEEE 802.1q tagging to sub-interfaces, thus, providing traffic separation within the network physical connection. This command has one additional argument, the VLANID. Once applied the interface will receive only 802.1q frames with that VLANID, likewise, any frame sent through this interface will be an 802.1q frame carrying that VLANID.

Let's take an example: we want to use interface **FastEthernet0/0** to connect the router to two VLANs with IDs 50 and 60, this can be achieved by the following commands:

```
(config) interface FastEthernet 0/0.1
(config-subif) encapsulation dot1Q 50
(config-subif) ip address 195.1.0.20 255.255.255.0
(config-subif) no shutdown
(config-subif) exit

(config) interface FastEthernet 0/0.2
(config-subif) encapsulation dot1Q 60
(config-subif) ip address 190.5.0.2 255.255.255.0
(config-subif) no shutdown
(config-subif) exit
```

Of course, for this configuration to make sense, this router's interface is supposed to be connected to the port of a switch with both these VLANs defined and assigned to that port (in trunk-mode).

### 1.3. Packet Tracer practice

Download from Moodle the Packet Tracer layout in Figure 3 (**p105-a.pkt**). It encompasses only the physical connections; all configurations are left for you to do.

Here we have two routers and three physical networks (LANs), however, there are five IPv4 networks from A to E, networks A and B are different VLANs over one same LAN and the same goes for networks D and E.

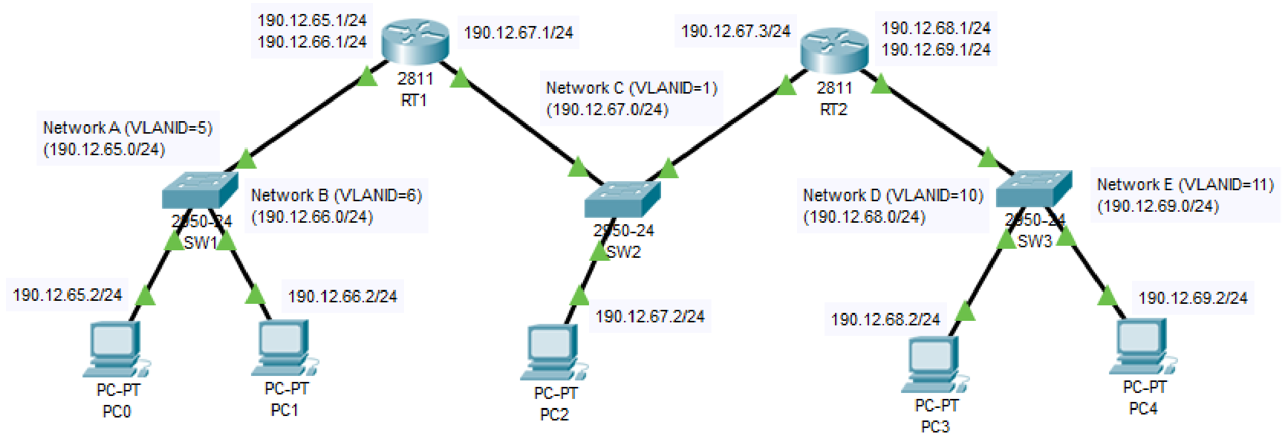

*Figure 3 - Networks layout for VLAN interfaces in routers*

**a) Proceed with configuration of VLANs:**

On switch SW1 define VLANs with VLANIDs 5 and 6 for IPv4 networks A and B, switch SW1 is connected to the previously referred hardware switching module (HWIC-4ESW) attached to router RT1. Define the same VLANs on router R1 switching module and change the connection to trunk-mode.

Switch SW2 is fully dedicated to the IPv4 network C, and thus, no VLANs are required here, this network is using the default VLAN (VLANID=1).

On switch SW3 define VLANs with VLANIDs 10 and 11 for IPv4 networks D and E, switch SW3 is connected to a standard router interface of RT2, on the switch change this connection to trunk-mode, on the router's side, use sub-interfaces.

**b) Assign to all nodes the shown IPv4 addresses.**

On router RT1 addresses on networks A and B are assigned to VLAN interfaces.

On router RT2 addresses on networks D and E are assigned to sub-interfaces.

**c) Configure routing (default-gateways on end nodes, and routing tables on routers).**

Every end-node must have a default-gateway (the local router's node address, e.g., for PC0 the default-gateway is 190.12.65.1. For PC2 either RT1 or RT2 can be the default-gateway.

Router RT1 must be informed networks D and E exist, add them to its routing table.

Router RT2 must be informed networks A and B exist, add them to its routing table.

**d) In simulation mode, test IPv4 connectivity by sending ICMP echo requests between all nodes.**

All nodes should be able to reach every other node regardless of the network they are connected to.

Mind that first ping tests may fail due to timeout, this happens when ARP tables are empty and the time it takes for ARP to fill them is longer than the ping test timeout.

## 2. DHCP (Dynamic Host Configuration Protocol)

Thanks to DHCP, when a node is plugged into a network gets all required IPv4 configuration information automatically. This only works if the network has at least one DHCP service running.

Among others, most relevant data provided by the DHCP service to the client node are a unique IPv4 address for the node to use, the network mask (prefix-length), the default gateway and the DNS configuration data.

The DHCP service can manage static and dynamic IPv4 addresses. Static IPv4 addresses are manually assigned by the service administrator to specific client nodes (identified by their MAC addresses). Dynamic IPv4 addresses are automatically assigned by the DHCP service from an address pool provided by the service administrator. The DHCP service identifies client nodes by their MAC address and ensures assigned IPv4 addresses are unique.

DHCP itself uses UDP over IPv4 for messages transport. At first glance, this would result in a cyclic dependency for the client node: needs DHCP to get the IPv4 configuration and DHCP requires IPv4 to be operating.

Nevertheless, this is overcome by using some special-purpose IPv4 addresses. The first message sent by the client node (Figure 4) is transported by an IPv4 packet with source address 0.0.0.0 (unknown IPv4 source address) and destination address 255.255.255.255 (local network IPv4 broadcast address).

Because the first contact from the client node to the DHCP service is achieved using broadcast, **it will only work if both the node and the service are in the same broadcast domain** (LAN/VLAN).

Every DHCP service within the broadcast domain will receive the discover message and reply with an offer message (Figure 5). The client can then select one and sends a broadcast request (Figure 6) to start using the provided configuration data.

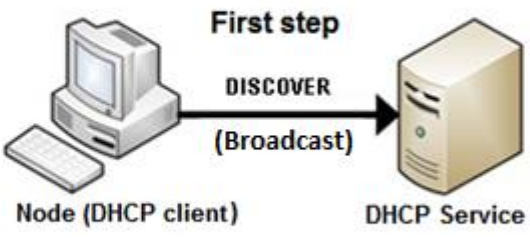

*Figure 4 - DHCP DISCOVER*

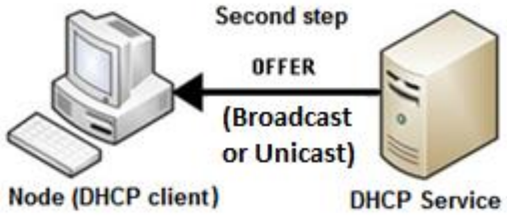

*Figure 5 - DHCP OFFER*

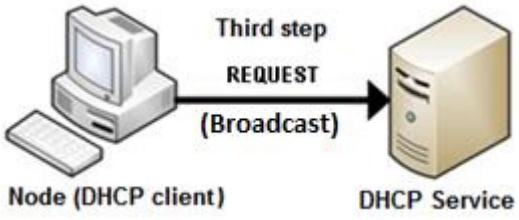

*Figure 6 - DHCP REQUEST*

Finally, the DHCP service sends back an ACK message (Figure 7) with all the required IPv4 configuration data.

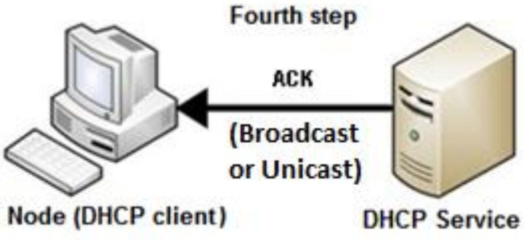

*Figure 7 - DHCP ACK (acknowledgement)*

The configuration assigned to the client node is valid during the **lease time** period, if the node wants to keep using the same data it must send a new request before the lease time expires.

### 2.1. Setting up the DHCP service on Cisco routers

Several types of network nodes can be used to supply a DHCP service to a network, including Windows and Linux servers, however, a Cisco router can also provide this service.

Because broadcast is used, one requirement to be meet is that the node providing the service must have a direct interface connection to the network. Although this requirement is generally true, a **DHCP Relay Agent** can be used to work around it. The DHCP Relay Agent must be directly connected to the network, but it will retransmit local DHCP messages to a remote DHCP service.

Although Cisco routers support static IPv4 addresses assignment as well, we usually want dynamic assignment. To achieve that on a Cisco router, one **DHCP pool** must be defined for each IPv4 network we want to offer the service on.

The DHCP pool is identified by an arbitrary administrative name, the **ip dhcp pool POOL-NAME** command can be used to create a new DHCP pool or edit an existing one. In will enter a specific sub-configuration level to manage that pool configuration, for instance:

```
(config)# ip dhcp pool NETWORK1
(dhcp-config)# network 192.168.5.0 255.255.255.0
```

The **network** command shown above associates the pool to a directly connected IPv4 network (192.168.5.0/24 on the example). The service assumes every valid node address within this network is available to be dynamically assigned to clients, addresses already in use on the network must be explicitly excluded.

With this information, the DHCP service is already capable of dynamically assigning IPv4 node addresses to clients and also inform them about the network address and network mask. Additional data is, nevertheless, required by clients, the **default-router** command settles the default gateway to be used by clients. The following example also shows how to set the default DNS domain, and DNS servers the clients should use.

```
(config)# ip dhcp pool NETWORK1
(dhcp-config)# network 192.168.5.0 255.255.255.0
(dhcp-config)# default-router 192.168.5.1
(dhcp-config)# domain-name dei.isep.ipp.pt
(dhcp-config)# dns-server 192.168.20.4
```

Node addresses that are being used for other purposes must be excluded from DHCP management, starting with the router own address, but including servers and other devices with manual IPv4 address configuration.

If fact, before assigning a new address, the DHCP service tests if anyone is replying to ICMP echo requests on that address and skips that address if so.

At general configuration-level, the **ip dhcp excluded-address** command allows the definition of one IPv4 address or a range of IPv4 addresses that should never be assigned to clients by DHCP, for instance:

```
(config)# ip dhcp excluded-address 192.168.5.1 192.168.5.100
```

Used together with the previous commands, this will make the DHCP service assign to clients only addresses between 192.168.5.101 and 192.168.5.254.

The **ip dhcp excluded-address** command may be used several times to exclude different addresses, possibly belonging to different DHCP pools.

### 2.2. Download from Moodle (pl05-b.pkt) the Packet Tracer layout shown in Figure 8

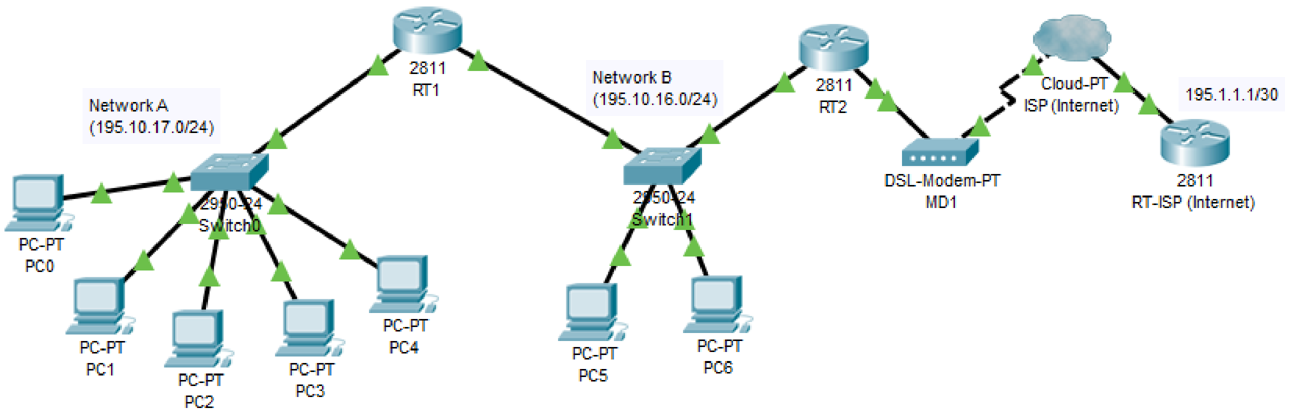

*Figure 8 - Networks layout for a DHCP exercise*

**This encompasses only physical connections; all configurations are left for you to do.**

#### a) Define routers' IPv4 addresses and static routing tables

For every router interface in use, assign a valid IP address on the network. You may use any valid node address as far as it belongs to the connected IPv4 networks shown on the image. Usually, but not mandatory, the network's first addresses are assigned to routers.

Regarding the IPv4 address to be assigned to RT2 connection to the internet, it's implicitly established because the other node address on this connection is 195.1.1.1/30. Mind the DSL modem and Cloud-PT are layer two, so they do not have IP addresses.

Don't assign IPv4 addresses to end-nodes, they will be assigned by the DHCP service running in router RT1, the default-gateway for end-nodes will be, as well, assigned by DHCP.

For RT1 routing table, the default route is sufficient, it will point out (next-hop) to RT2.

For RT2, the default route should point to the internet, so the next-hop is 195.1.1.1, however, this router's routing table must first state that if the destination address is Network A, then the next-hop is router RT1.

Router RT-ISP (on the Internet) would have a default-route pointing to the next router, deeper into the internet. We don't have data to settle that. However, when the destination node belongs to networks A or B it should send to RT2. Also notice networks A and B can be aggregated into a single network address.

#### b) Configure the DHCP service on router RT1 to assign IPv4 address to all end nodes

Because nodes are spread in two networks, two DHCP pools are required.

Within each pool, set the correct default gateway each node must use (default-router command). For the PCs in Network B it should be RT2 because it's closer to the Internet connection than RT1.

**Don't forget to exclude from DHCP already in use addresses, namely routers' addresses.**

Check that every PC is getting the proper configuration through DHCP from router RT1.

#### c) Test ICMP echo requests between all nodes, including the router RT-ISP

They should all work.

#### d) In simulation mode, send an ICMP echo request to a locally unknown IPv4 addresses

If everything is working as it is supposed to, those requests will reach the ISP Router and be rejected there. They are rejected because it has no routes pointing towards the internet as it would happen in reality.

## 3. Create in Packet Tracer the physical infrastructure in Figure 9.

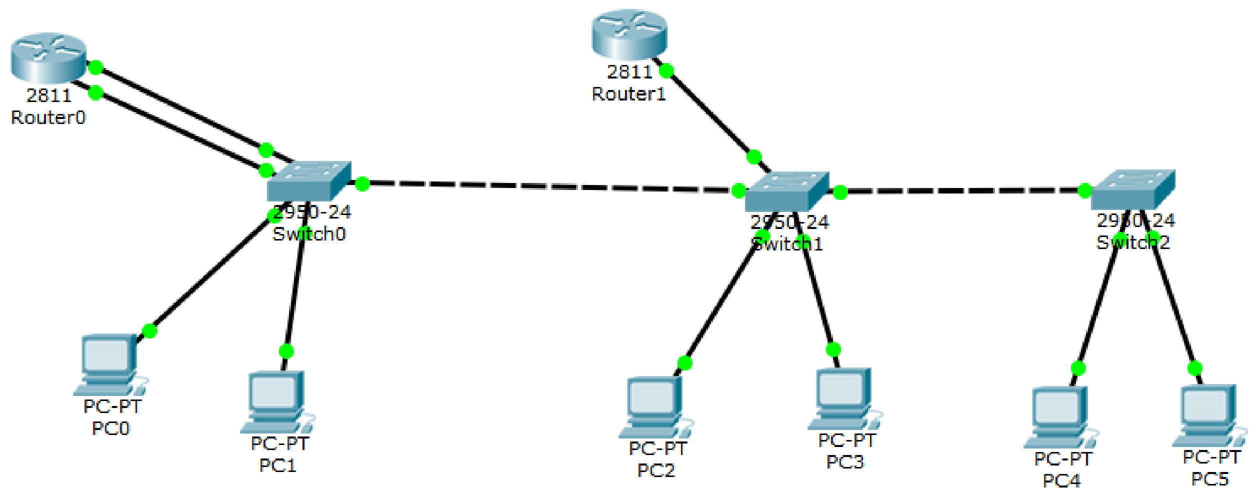

*Figure 9 - Physical infrastructure*

It looks rather odd! Router 1 has a single connection! Router 0 has two connections to the same switch! **The point is, physical connections don't mean much, as it all depends on the VLANs.**

### 3.1. Over the physical infrastructure in Figure 9, use VLANs to define the IPv4 networks shown in Figure 10

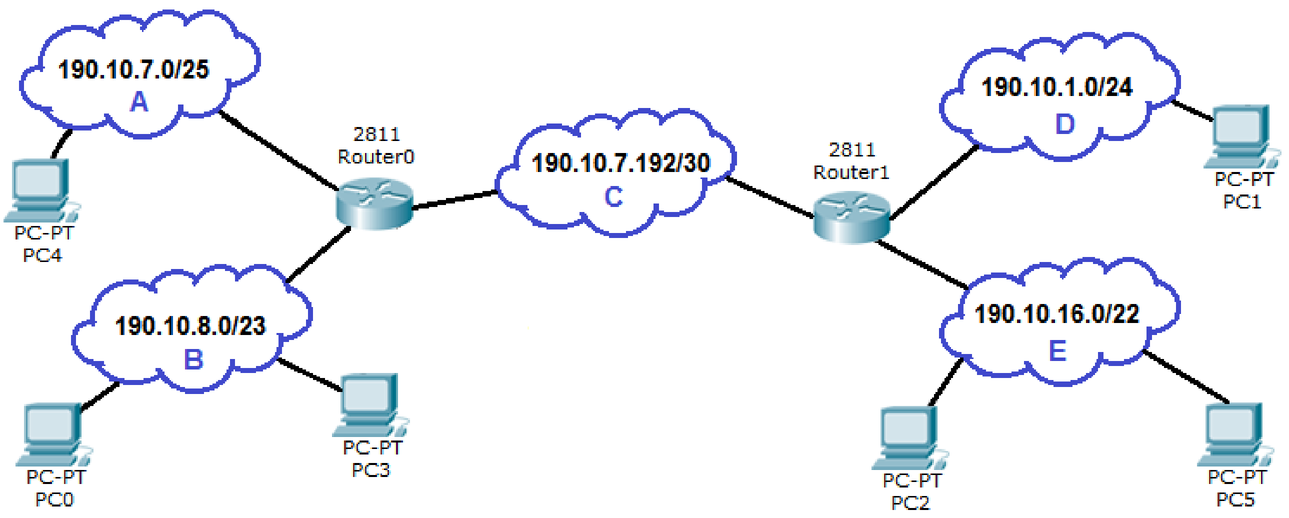

*Figure 10 - VLANs and IPv4 networks to implement*

#### a) Define de required VLANs using VTP

For each independent IPv4 network a VLAN is required, so a total of five VLANs are necessary. From the Figure 10 we can summarize the list of VLANs and IPv4 networks in Table 1.

*Table 1 - VLANs and IPv4 networks*

| VLAN name | VLANID | IPv4 network |
|---|---|---|
| **VLAN-A** | 100 | 190.10.7.0/25 |
| **VLAN-B** | 101 | 190.10.8.0/23 |
| **VLAN-C** | 102 | 190.10.7.192/30 |
| **VLAN-D** | 103 | 190.10.1.0/24 |
| **VLAN-E** | 104 | 190.10.16.0/22 |

Assigned VLANIDs are arbitrary, as far as they are unique and don't overlap others in use, that's ok.

Here we have only three switches, the five VLANs could be manually created on each. However, VLAN Trunking Protocol (VTP) offers a better solution. By using VTP we can define all VLANs in a single switch (the VTP server) and spread them to all other switches (VTP clients). To works together, both the **VTP server** and the **VTP clients** must belong to the same **VTP domain**.

One switch must be elected to be the VTP server, say **Switch0**. Enter Switch0 CLI and set it as the VTP server for a VTP domain, say **rcomp** VTP domain.

```
    (config)# vtp domain rcomp
    (config)# vtp mode server
```

The other switches will be VTP clients on the **rcomp** VTP domain.

```
    (config)# vtp domain rcomp
    (config)# vtp mode client
```

From now on, any change on VLANs defined in Switch0 (VLAN database) gets propagated to Switch1 and Switch2.

**So, you can now use the Packet Tracer form to define the VLAN database on Switch0.**

**b) Ensure all VLANs are available on all switches**

**Every switch port connected to another switch <u>must be in trunk-mode</u>,** so that corresponding VLANs are effectively interconnected between switches. On Cisco switches, ports are by default in dynamic mode, thus, if one end of the connection is manually changed to trunk-mode, the other end will automatically also change to trunk-mode.

**c) Assign VLANs to switch ports connected to end nodes**

These end nodes (PCs) are not VLAN-aware, for each PC check to which switch port it's connected, then and assign to that port the appropriate VLAN in access-mode.

**d) Assign VLANs to switch ports connected to routers**

**Router1** has a single connected physical interface, however, we need three network connections. The connected switch port must, therefore, be in trunk-mode.

Assuming **Router1** physically connected interface is **FastEthernet0/0** and addresses to be assigned to the router are **190.10.7.193/30** for VLAN-C, **190.10.1.1/24** for VLAN-D and **190.10.0.1/22** for VLAN-E. Configuration commands are:

```
    (config)# interface FastEthernet0/0.1
    (config-subif)# encapsulation dot1Q 102
    (config-subif)# ip address 190.10.7.193 255.255.255.252

    (config)# interface FastEthernet0/0.2
    (config-subif)# encapsulation dot1Q 103
    (config-subif)# ip address 190.10.1.1 255.255.255.0

    (config)# interface FastEthernet0/0.3
    (config-subif)# encapsulation dot1Q 104
    (config-subif)# ip address 190.10.16.1 255.255.252.0
```

Concerning **Router0**, it has two connected physical interfaces and, as with **Router1**, three network connections are required. If physically connected interfaces are **FastEthernet0/0** and **FastEthernet0/1**, we can for instance use one interface for one VLAN and the other interface for the other two VLANs. Use the following commands:

```
      (config)# interface FastEthernet0/0
      (config-if)# ip address 190.10.7.194 255.255.255.252

      (config)# interface FastEthernet0/1.1
      (config-subif)# encapsulation dot1Q 100
      (config-subif)# ip address 190.10.7.1 255.255.255.128

      (config)# interface FastEthernet0/1.2
      (config-subif)# encapsulation dot1Q 101
      (config-subif)# ip address 190.10.8.1 255.255.254.0
```

Establish the correct configuration for each switch port these routers' interfaces are connected. Both in trunk-mode? Apply the correct VLAN configurations to the connected switch.

### e) Use the two routers to offer the DHCP service on networks A, B, D, and E

Each router can serve only directly connected networks, so, for Router0 networks A and B, and for Router1 networks D and E.

On **Router0**:

```
      (config)# ip dhcp excluded-address 190.10.7.1
      (config)# ip dhcp excluded-address 190.10.8.1

      (config)# ip dhcp pool NET-A
      (dhcp-config)# default-router 190.10.7.1
      (dhcp-config)# network 190.10.7.0 255.255.255.128

      (config)# ip dhcp pool NET-B
      (dhcp-config)# default-router 190.10.8.1
      (dhcp-config)# network 190.10.8.0 255.255.254.0
```

On **Router1**:

```
      (config)# ip dhcp excluded-address 190.10.1.1
      (config)# ip dhcp excluded-address 190.10.16.1

      (config)# ip dhcp pool NET-D
      (dhcp-config)# default-router 190.10.1.1
      (dhcp-config)# network 190.10.1.0 255.255.255.0

      (config)# ip dhcp pool NET-E
      (dhcp-config)# default-router 190.10.16.1
      (dhcp-config)# network 190.10.16.0 255.255.252.0
```

Check that all end nodes receive their configuration from the DHCP service. Confirm the IPv4 network each node belongs to.

### f) Move end nodes between VLANs

**Let's move PC0 to network A and PC2 to network D.**

Thanks to VLANs no physical connections changes are necessary, all is required is changing the VLAN assigned to the corresponding switch port.

Force DHCP clients on PC0 and PC2 to request new configuration data and confirm they have moved to the desired IPv4 network.

## 4. Simplifying IPv4 routing tables

Routers' routing tables can often be simplified, this means **reducing the number of lines, and yet keeping the same behaviour**.

In routing terms, the same behaviour expresses the fact that same next hops are used for the same cases as before. The advantage is **a shorter routing table improves the router's performance.**

The generic principle for simplifying a routing table is searching for a **pair of lines** that could be **replaced by a single line**, which has the **same effect of the original two lines**.

There's one first fundamental condition for such a simplification to be possible:

**Both original lines need to have <u>the same next hop</u>**.

Once we notice two routing table lines have the **same next hop, only then**, we should proceed and check if they may be reduced to a single line (and keeping the same behaviour).

### 4.1. Aggregation to default-route

If there's a default-route in the routing table (0.0.0.0/0 line), it means any IPv4 address not matching previous lines will match it, and thus the corresponding next-hop will be used. Typically, this next-hop pinpoints toward the internet connection and is called the default-router or default-gateway.

**In most cases, if there's a routing table line with a next-hop equal to the next-hop of the default-route, then the line can simply be removed, leaving only the default-route.**

This is true because we assume that, in the absence of the removed line, the default-route is going be reached, and therefore the same next-hop will be used as before.

We must be aware that there are exceptions to this assumption, the routing table in Table 2 presents one of those cases.

*Table 2 - Routing table with impossible aggregation to the default-route*

| Destination | Next hop |
|---|---|
| 190.200.30.0/24 | 170.10.10.1 |
| 190.200.0.0/16 | 170.10.10.5 |
| 0.0.0.0/0 | 170.10.10.1 |

We would assume the first rule could be removed because the next-hop is the same as the default-route next-hop. **However, in this specific case that's not true**.

The issue is, the second line includes the first line addresses, therefore, the assumption that removing the first line would make the corresponding addresses match the default-route is, in this case, false. They would match the second line, and thus, they would be sent to a different next hop.

Due to that, **this routing table can't be simplified.**

Nevertheless, **aggregation to the default-route** is pretty simple and it should always be the first strategy to be used. Only after getting rid of all removable lines through **aggregation to the default-route**, then the next step, **addresses blocks aggregation,** ought to be tried.

### 4.2. Addresses blocks aggregation

From IPv4 classless addressing study, we already know that two same size addresses blocks **can, in some cases,** be aggregated into a single block with the double size. Formally, viewing addresses blocks as sets of addresses, the aggregation result block is the **union** between the two original addresses blocks.

If the routing table contains two lines with equal sizes addresses blocks (same prefix-length) and the same next-hop, there's the chance these two lines can be replaced by a single line with an addresses block that is the union of the first two.

Criteria that must be checked for addresses block aggregation of two routing table lines:

First – Same next-hop (the basic criterion)
Second – Same block size (same prefix-length/network mask)
Third – The two original blocks can be aggregated into a single block

Once the first two rules are verified, we must test the third rule. The test is rather simple:

**Two blocks can be aggregated if, by reducing one bit to the prefix-length on both blocks, we end up with the same resulting block.**

By reducing one bit to the prefix, one bit is transferred from the network area to the node area of the address, thus depending on the transferred bit being zero or one, the resulting network address will remain the same or will change.

Of course, for the resulting network address to be the same, **the two original networks addresses must be very similar**, they can only differ on the value of one bit around the prefix-length position.

**Example: simplify the routing table in** Table 3.

*Table 3 - Routing table with possible aggregations*

| Destination | *Next-hop* |
|---|---|
| 195.20.80.0/20 | 178.10.10.1 |
| 195.20.64.0/20 | 178.10.10.1 |
| 195.30.64.0/20 | 178.10.10.1 |
| 195.20.96.0/19 | 178.10.10.1 |
| 0.0.0.0/0 | 178.10.10.80 |

We can see no aggregation to default-route is possible, this is because no other line has the same next-hop as the default-route (178.10.10.80).

Though, the remaining lines all share the same next hop (178.10.10.1), so we can analyse aggregations between them. The aggregation is only possible between same mask networks, so we have three possible candidates with a 20-bits prefix length.

Nevertheless, the third candidate (195.30.64.0/20) can be immediately excluded. The reason is that the value of the second octet is different and the prefix-length is on the middle of the third octet. Obviously, there's no way changing a bit on the third octet will change the second octet's value.

For now, we are left with **195.20.80.0/20** and **195.20.64.0/20**, the first two criteria are met: same next-hop and same prefix-length, also addresses are similar and differ only on the octet where the prefix-length is located.

Let's test one bit prefix-length reduction to a 19-bits network prefix-length on both networks:

**195.20.80.0/20 = 195.20.(01010000)$_2$.0/20**

By reducing one bit, the 20th-bit having value **one** will no longer be part of the network address, thus the address becomes **195.20.64.0/19**

**195.20.64.0/20 = 195.20.(01000000)$_2$.0/20**

By reducing one bit, the 20th-bit having value **zero** will no longer be part of the network address, however, because it zero, the address remains the same: **195.20.64.0/19**

We end up with the same network address on both cases, therefore, we conclude: **195.20.64.0/19 is the union of 195.20.80.0/20 and 195.20.64.0/20**

Thus, for now, the simplified routing table becomes what is shown in Table 4.

*Table 4 - Routing table after first aggregation*

| Destination | *Next-hop* |
|---|---|
| **195.20.64.0/19** | **178.10.10.1** |
| 195.30.0.0/20 | 178.10.10.1 |
| 195.20.96.0/19 | 178.10.10.1 |
| 0.0.0.0/0 | 178.10.10.80 |

We can now restart the procedure, because again, two lines meet the requirements for aggregation: 195.20.64.0/19 and 195.20.96.0/19

(Same next-hop, same prefix-length, and similar addresses, different only on the octet where the prefix-length is located)

Let's again test one bit prefix-length reduction, now to an 18-bits network mask on both networks:

**195.20.64.0/19 – 195.20.(01000000)$_2$.0/19**

By reducing one bit, the 19th-bit having value **zero** will no longer be part of the network address, thus the address remains the same: **195.20.64.0/18**

**195.20.96.0/20 – 195.20.(01100000)$_2$.0/19**

By reducing one bit, the 19th-bit having value **one** will no longer be part of the network address, thus the address becomes **195.20.64.0/18**

We can, therefore, conclude: **195.20.64.0/18 is the union of 195.20.64.0/19 and 195.20.96.0/19**

So, the final simplified routing table results in Table 5.

*Table 5 - Routing table after the second aggregation*

| Destination | *Next-hop* |
|---|---|
| 195.20.64.0/18 | 178.10.10.1 |
| 195.30.0.0/20 | 178.10.10.1 |
| 0.0.0.0/0 | 178.10.10.80 |

No further simplifications are possible.
(To start with, the only two lines with the same next-hop have different prefix-lengths, and in addition network addresses are quite different)

## 5. Practical exercise (IPv4 networks dimensioning and static routing)

**The diagram in** Figure 11 **represents some IPv4 networks interconnected by routers.**

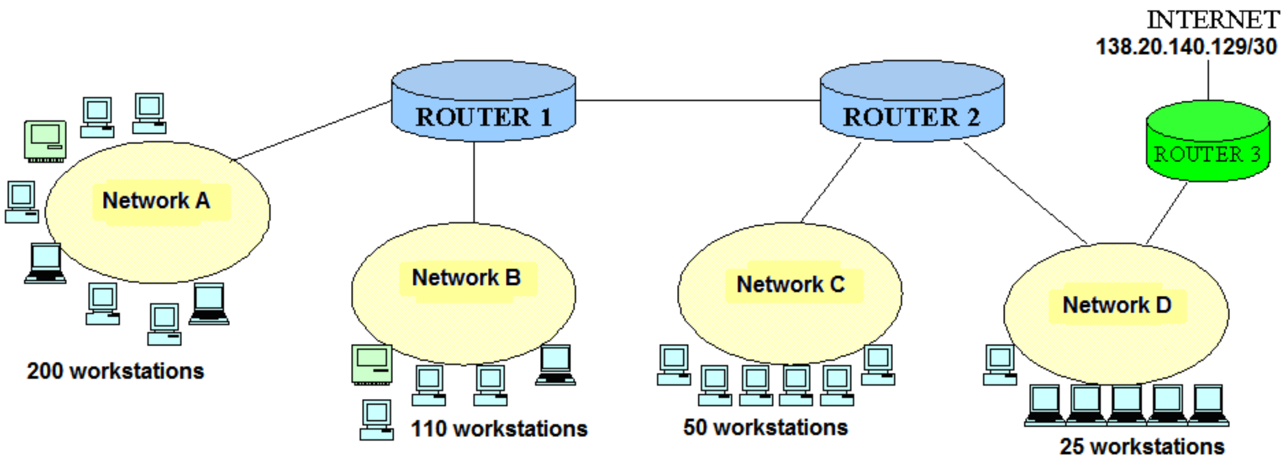

*Figure 11 - IPv4 networks interconnected by three routers*

The maximum number of workstations to be supported in each network are shown in Figure 11.

a) Use the **194.56.224.0/23** addresses block to assign an address to each network.

b) Define the routers' IPv4 addresses in each network.

c) Define each router's static routing table.

d) Simplify to the possible extent each routing table defined in c).