Redes de Computadores (RCOMP) – 2017/2018 Laboratory Class Script – PL07	
• Classless IPv4 addressing. • IPv4 networks dimensioning with classless addressing (VLSM). • Practical exercises • Support on Project 1 progress	

1. Classless IPv4 addressing

Classful addresses (8, 16 and 24 bits masks/prefixes) lead to severe addresses wasting because often, the real needs don't match the used masks. Some examples:

A two nodes network (dedicated connection) – will require 256 addresses (a C class network)

A network with 300 nodes – will require 65536 addresses (a B class network)

With classless IPv4 addressing, other prefixes can be used, this allows a far better adjustment to the real network required size.

All principles for classful addressing are also valid for classless addressing, however:

- The network mask (prefix) can no longer be established from the value of the address's first bits, therefore, to identify a network, the network prefix length must be included along with the network address.

- The four octet dot-decimal representation is still used, but is less convenient for addresses analysis because now prefix lengths may not match octets. Most address analysis will have to be done in binary representation.

Network mask expressed in dot-decimal representation are now hard to interpret, for instance to express a 27-bits long prefix, the network mask is 255.255.255.224, this is easier to read from the binary representation (network bits in red): **11111111.11111111.11111111.11100000**.

As an alternative to dot-decimal network masks, the CIDR (Classless Inter-Domain Routeing) form can be used. In CIDR notation the network prefix length is added to the address, separated by a slash. So in the above case, it would be **/27**.

Most address analysis must now be performed in binary format. In general, addresses and masks must be converted to binary representation, analysis performed and then results converted back to dot-decimal representation.

Usually the binary analysis will be focused in the octet where the network prefix is, so there is no need to represent all 32 bits in binary, only that octet.

Let's take a practical example:

Network: 194.120.8.0/21

With this prefix, the leftmost 21 bits are used to identify the network and the remaining 11 are used to identify nodes within that network.

The number of addresses in the network (address space) is 2^{11} = 2048, the number of valid node address this network can contain are 2046 (2048-2)

The network prefix lies on the **third octet**, so the binary analysis is relevant only at this octet.

The network mask is: **11111111.11111111.11111000.00000000**, using binary representation for the third octet only we have: **255.255.(11111000)$_2$.0**, in dot-decimal **255.255.248.0**.

The network address is **194.120.(00001000)$_2$.0**, in dot-decimal **194.120.8.0**

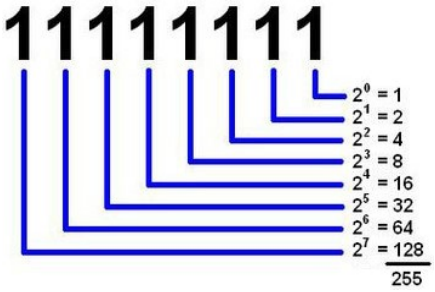
The first valid node address is **194.120.(00001000)$_2$.1**, in dot-decimal **194.120.8.1**.

The broadcast address is **194.120.(00001111)$_2$.255**, in dot-decimal **194.120.15.255**.

2. Manual conversion from binary to decimal and vice versa

Manually converting an octet between its binary representation and the decimal representation is quite simple.

Each bit represents a decimal value, these decimal values are the powers of two, thus, the less significant bit decimal value is one, and each following bit doubles the decimal value of the previous bit.



To convert from binary to decimal you just have to sum the decimal values of all one bits. For instance, **01100101** can be converted to decimal representation by calculating the sum:

64+32+4+1 = 101

Converting from decimal to binary is the reverse operation, you will try to deduct each bit decimal value from the original decimal representation, starting with the most significant bit. If deducting is possible, it means that bit is one, otherwise is zero.

Let´s take an example: **202**

202 – 128 = 74 (this means bit with decimal value 128 is one)

74 – 64 = 10 (this means bit with decimal value 64 is also one)

10 – 32 would be negative (this means bit with decimal value 32 is zero)

10 – 16 would be negative (this means bit with decimal value 16 is also zero)

10 – 8 = 2 (bit with decimal value 8 is one)

2 – 4 would be negative (bit with decimal value 4 is also zero)

2 – 2 = 0 (bit with decimal value 2 is one)

0 – 1 would be negative (bit with decimal value 1 is zero)

So, the binary representation of **202** is **11001010**

3. Practical exercises

For each given network, expressed in CIDR notation, present the network mask in dot-decimal notation, the network's first valid node address and the broadcast address:

a) **170.20.0.0/22**

b) **200.100.20.192/26**

c) **120.64.0.0/12**

d) **191.123.90.104/30**

e) **138.20.64.0/18**

4. IPv4 networks dimensioning with classless addressing (VLSM).

IP network dimensioning consists on establishing the appropriate network mask to meet the number of network nodes the network must support.

Each network mask sets the number of bits that will be available to identify nodes within it and, therefore, the maximum number of nodes it can have. We will be calling **address block** to the set of different addressed within a network, remember however that two addresses are reserved and cannot be used as valid node addresses.

The biggest prefix useable in IPv4 is 30 bits, this means only two bits are left to identify addresses within such a network, the block size is four and the number of valid node addresses is only two. This prefix is commonly used for dedicated connections between routers where only two valid node addresses are required.

For each bit reduced in the prefix the block size doubles, inversely for each bit added to the prefix, the block size is reduced to half. Keeping in mind some significant prefix values its, therefore, easy to reach the block size for any prefix.

Network prefix	Address block size	Number of valid node addresses
/30	**$2^2 = 4$**	**4 - 2 = 2**
/29	$2^3 = 8$	8 - 2 = 6
/28	$2^4 = 16$	16 - 2 = 14
/27	$2^5 = 32$	32 - 2 = 30
/26	$2^6 = 64$	64 - 2 = 62
/25	$2^7 = 128$	128 - 2 = 126
/24	**$2^8 = 256$**	**256 - 2 = 254**
/23	$2^9 = 512$	512 - 2 = 510
/22	$2^{10} = 1024$	1024 - 2 = 1022
/21	$2^{11} = 2048$	2048 - 2 = 2046
/20	$2^{12} = 4096$	4096 - 2 = 4094
/19	$2^{13} = 8192$	8192 - 2 = 8190
(…)	(…)	(…)
/16	**$2^{16} = 65536$**	**65536 - 2 = 65534**
(…)	(…)	(…)

By adding one bit to the prefix, a blocks is split into two blocks of half the size of the original block. One half-size block starts at the same address of the initial block and the other at the middle of the initial block. This is always valid whatever the initial block may by.

By reducing one bit to the prefix, two blocks are merged to form a new block with double size, this is usually referred to as aggregation.

Reducing one prefix bit (aggregation) over two existing address blocks is not always valid.

Example:

Network 120.10.5.0/24 and 120.10.6.0/24 cannot be aggregated by reducing the prefix to 23 bits

Let's try:

120.10.5.0/24 - 120.10.(0000 0101).0

120.10.6.0/24 - 120.10.(0000 0110).0

Reducing the prefix to 23 bits results in:

120.10.5.0 - 120.10.(0000 0100).0 , thus, 120.10.4.0/23

120.10.6.0 - 120.10.(0000 0110).0 , thus, 120.10.6.0/23

Se they and up in different 23 prefix address blocks, not the same.

However, other aggregations are valid:

120.10.4.0/23 = 120.10.4.0/24 + 120.10.5.0/24

And also: 120.10.6.0/23 = 120.10.6.0/24 + 120.10.7.0/24

And also: 120.10.4.0/22 = 120.10.4.0/23 + 120.10.6.0/23

To summarize: any address block can always be split into two (with half the size each), however, two equal size address blocks may or may not be aggregable into a single double size block.

Once we know the address block size, knowing each network address its easy, to get the next network address you can add the address block size to the network address, thus, network addresses can be assigned sequentially.

Example:

Given the 190.130.0.0/24 address block, use it to establish several IPv4 networks capable of holding up to 20 nodes each.

The best fitted prefix for up to 20 nodes is the 27 bits prefix length, resulting in address blocks of 32 addresses each (up to 30 nodes)

First network - 190.130.0.0/27

Second network – 190.130.0.32/27 (0 + 32)

Third network – 190.130.0.64/27 (32 + 32)

Fourth network – 190.130.0.96/27 (64 + 32)

Fifth network – 190.130.0.128/27 (96 + 32)

(…)

We can also use different prefixes over the original block, it's useful if each network to be established has a different requirement regarding the number of nodes to be supported. This is called VLSM (Variable Length Subnet Mask).

We already know every address block can always be split into two of half size each, consequently, **wherever a block starts (a network) at that same address also starts any smaller block** (bigger prefix length).

This means, even with VLSM, we can still use the sequential assignment strategy as far as **bigger blocks** (smaller prefixes) **are assigned first**.

In the sequential assignment strategy, we get the next network address by adding to the current network address its block size, this results in the starting point of the next address block with the same size.

Let's test this:

Given the 190.130.0.0/24 addresses block we want to assign addresses for three networks: one with up to 20 nodes, another with up to 60 nodes and yet another with up to 100 nodes.

As requirements for each network are different we will be using VLSM

For the, up to 20 nodes network – a 32 addresses block – 27 bits prefix

For the, up to 60 nodes network – a 64 addresses block – 26 bits prefix

For the, up to 100 nodes network – a 128 addresses block – 25 bits prefix

The sequential assignment strategy can be used, **as far as we start by the bigger blocks**:

First, the 128 addresses block: 190.130.0.0/25

Second, the 64 addresses block: 190.130.0.128/26 (0 + 128)

Finally, the 32 addresses block: 190.130.0.192/27 (128 + 64)

If we disregard the rule **bigger blocks first**, the solution may not be valid:

First, a 64 addresses block: 190.130.0.0/26
Second, a 128 addresses block: 190.130.0.64/25 (0 + 64)

The problem is the second block does not exist. Notice that, with a 25 bits prefix, the bit with decimal value 64 is beyond the network prefix, and thus, it should be zero when representing the network address.

When handling prefixes with less than 24 bits, block sizes will be above 256 addresses, then to add the address block size to the current network address it's more convenient to express the block size in dot-decimal representation.

Example for prefixes less than 24 bits

Given the 190.130.128.0/17 addresses block, use it to assign network addresses to three networks with the following networks capacities: up to 500 nodes, up to 1000 nodes and up to 2000 nodes

500 nodes – 512 addresses block – 23 bits prefix

1000 nodes – 1024 addresses block – 22 bits prefix

2000 nodes – 2048 addresses block – 21 bits prefix

512 in binary is 10 00000000, in dot-decimal notation 2.0

1024 in binary is 100 00000000, in dot-decimal notation 4.0

2048 in binary is 1000 00000000, in dot-decimal notation 8.0

As before, the sequential assignment strategy can be used if we follow the rule **bigger blocks first**.

2048 addresses block	190.130.128.0/21	
1024 addresses block	190.130.136.0/22	**(128.0 + 8.0)**
512 addresses block	190.130.140.0/23	**(136.0 + 4.0)**

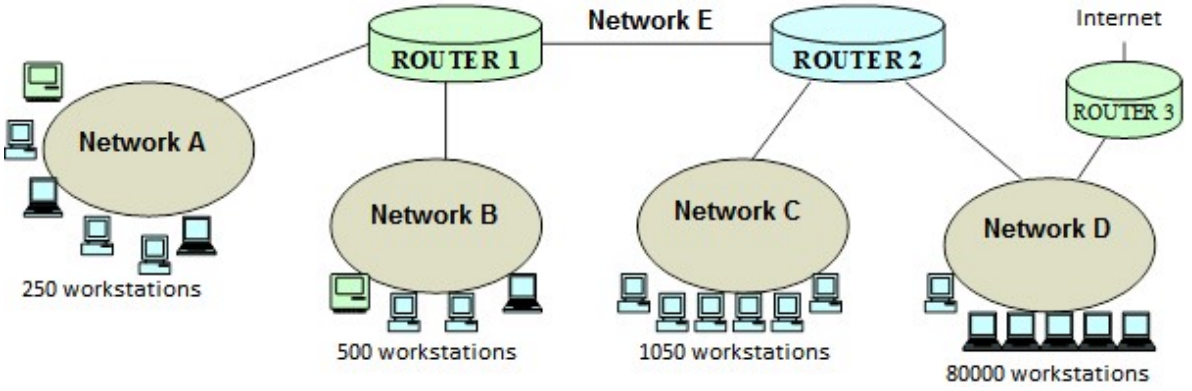
5. Practical exercises

5.1. Use the 173.30.60.128/25 addresses block to assign network addresses to two networks capable of supporting up to 30 nodes and one network capable of supporting up to 8 nodes.

5.2. Define network addresses for networks A, B and C within the 180.30.0.0/20 addresses block. Network A can support up to 2000 nodes, network B up to 500 nodes and network C up to 200 nodes.

5.3. Rethinking a problem from last week

See the following diagram representing several IPv4 networks interconnected by routers.



a) Use the **10.48.0.0/14** addresses block to assign addresses to all represented networks (A; B; C; D and E), supporting the indicated number of nodes, and also, addresses wasting should be avoided as far as possible.

b) Accordingly, set the IPv4 node addresses for each router interface.

c) Define each router static routeing table.

6. Support on Project 1 progress

The following sprints are focus on the Packet Tracer simulation. The goal is providing a realistic layer two and layer three network simulation, operating over the designed structured cabling infrastructure.

6.1. Structured cabling representation

Structured cabling details are not to be taken accurately in account, meaning that not all cables and network outlets are to be represented. Nevertheless:

- Each cross-connect it to be represented by a single layer two switch (Cisco model).

- Cable types (copper or fibre) are to be honoured.

- For each horizontal cross-connect, one workstation is used to represent each VLAN that is supposed to be available at that location. This includes one wireless workstation for each access-point.

- Same pathway redundant cables (link aggregation) are not to be represented, they should be represented by a single cable.

- Different pathways redundant cables (failover) should be represented. At layer two, STP will be used to avoid loops.

6.2. Layer two network devices configuration

A number of VLANs are required to be defined, on each VLAN there will be an IPv4 network running. Each VLAN is defined by assigning a name and a VLANID, and then adding it to the device's VLAN database.

VLAN names are arbitrary but cannot contain spaces and must be unique. The VLANID is a unique 12-bits number, some are already allocated (for instance one for the default VLAN). VLANID numbers between 2 and 1000 are usually available.

All VLANs should be available on every switch (cross-connect), on the next laboratory class we will see how VTP can be used to automate the VLAN database distribution to all switches. The VLAN database must also be included in the report and match those on the simulation devices. As stated before, STP is essential to avoid loops over redundant links, though it's enabled by default on most switches.

On each switch (representing a cross-connect), ports are to be assigned to each significant VLAN at that location. A workstation, VoIP phone or access-point should be accordingly connected to appropriate switch ports.

6.3. Layer three network devices configuration

In the project report, IPv4 networks to meet requirements are to be defined, they must be fitted within the provided addresses block. Each IPv4 network will be run over one distinct VLAN. The IPv4 addresses of each router interface and static routeing tables are also required to be defined on the project report.

On the simulation, the same configuration settings must be deployed. Each planned router is represented by a Cisco model, **at least one 2811 model is required to support VoIP services**.

In addition, on the simulation, routers are supposed to assign IPv4 configuration data to workstations on every VLAN. On the next laboratory class we will learn how to configure the DHCP service on CISCO routers.