RCOMP - Redes de Computadores (Computer Networks)

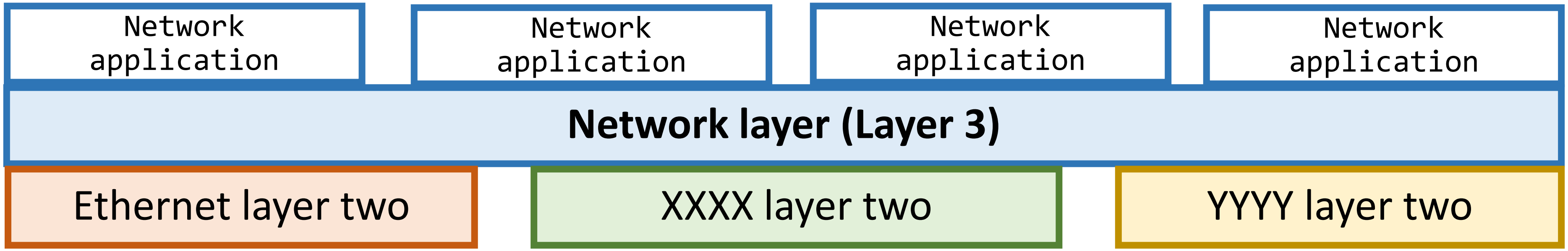
2023/2024

Theoretical-practical lesson 04

- Introduction to the IPv4 operation.
- IPv4 addressing and network masks.

The network layer (layer three)

End user network applications are not supposed to have direct interaction with layer two technologies like Ethernet. The network layer (layer three) has the important role of creating a communications abstraction level.



Without a Network Layer, applications would have to be developed for specific layer two types, therefore they could only communicate with other applications developed for that same layer two type. Moreover, the layer two implementation would have to be the same (the same local network).

The network layer (layer three) has the ability to adapt itself and use different layer two types for the purpose of transporting its packets. By doing so it's able to transfer data from one layer two network to another layer two network, even if they are of different types.

Internet Protocol (IP)

Although in the past several different network layer protocols did exist, nowadays everybody recognizes the total domain of the internet protocol (IP).

Having a single alternative for layer three is a good thing. It means everybody is using the same language at layer three (IP), therefore all nodes using this layer can communicate with each other. The IP protocol is used globally at the internet and unites all nodes around the planet (and beyond).

And yet, there are two IP versions, the currently most widely used is still version four (IPv4), but version six (IPv6) is gradually being deployed into the internet.

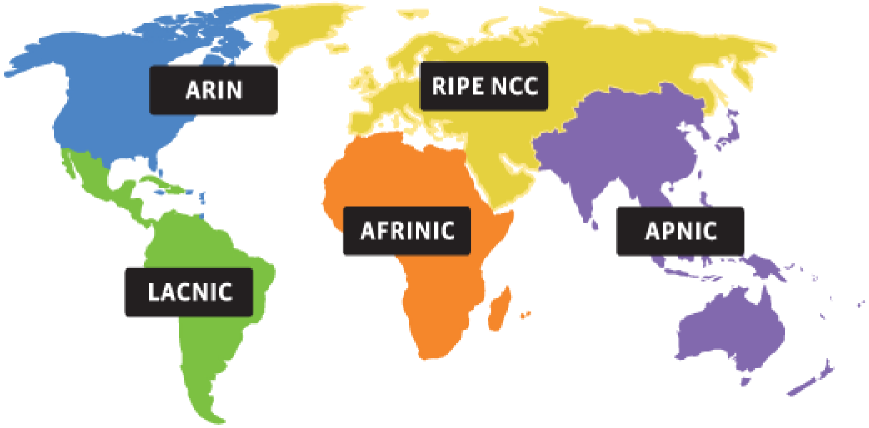
The main motivation for IPv6 is overcoming the IPv4 limits on the maximum number of supported nodes. With IPv4, nodes are identified by unique 32-bits numbers, whereas in IPv6 nodes are identified by unique 128-bits numbers.

O node using IPv4 will not be able to communicate with another node using IPv6. Yet, nowadays, most networks nodes are dual-stack, this refers to the fact they have both IPv4 and IPv6, therefore they can communicate with other nodes using either IPv4 or IPv6.

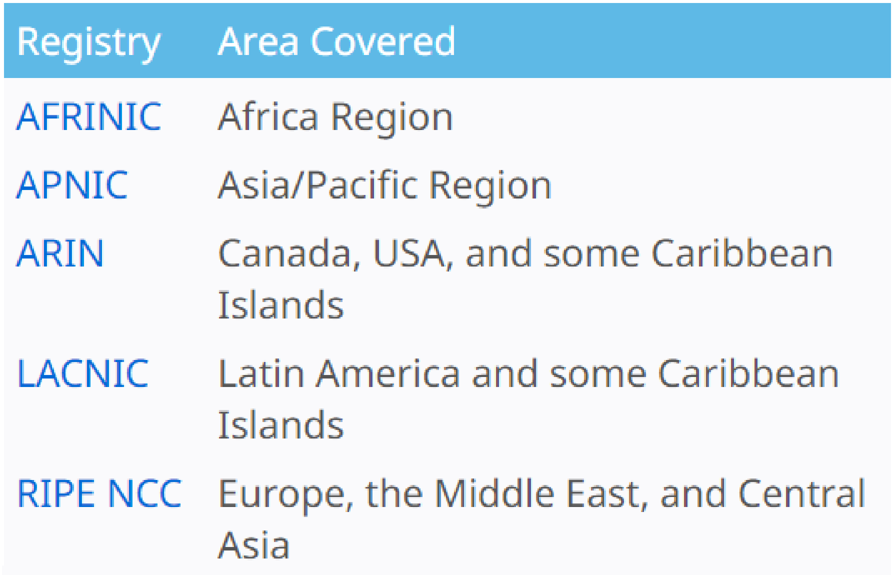
IP node addresses assignment

In IPv4, network nodes are identified by unique 32-bits numbers. To ensure each node has a unique 32-bits address, they are assigned by a global entity called the **Internet Assigned Numbers Authority** (IANA).

IANA manages the 32-bits address space globally, but not directly, the address space is divided by IANA into blocks and each block management is delegated to a RIR (Regional Internet Registries), each for a different part of the world.



(http://www.iana.org/_img/2013.1/rir-map.svg)



Registry	Area Covered
AFRINIC	Africa Region
APNIC	Asia/Pacific Region
ARIN	Canada, USA, and some Caribbean Islands
LACNIC	Latin America and some Caribbean Islands
RIPE NCC	Europe, the Middle East, and Central Asia

Each RIR manages the address space assigned to it by IANA. In turn, each RIR will divide its address space, and delegate parts of it to National Internet Registry (NIR) entities or to Local Internet Registry (LIR) entities.

Routers

The network layer (layer three) operates by using existing layer two implementations; however, each layer two network has a limited extent (the broadcast domain). This represents how far can a communication reach by using only the layer two and nothing else.

To go beyond the layer two network limits, devices called **routers** operate at the layer three, they operate by transferring layer three packets from one layer two network to another layer two network.

Routers are the network layer (layer three) **intermediate nodes**, they receive layer three packets that are not intended to them and retransmit those packets to other routers to ultimately reach the required destination node address.

Routers do this by taking routing decisions (to where retransmit a packet). Routing decisions are taken by searching the **packet destination node address** in a table, pretty much like an Ethernet switch does with the MAC table. For routers, though, the task is simpler because at layer three the network concept exists, and thus the **network address** concept.

The network address identifies a network (containing many nodes). The advantage is that now routing decisions can be taken network by network and not node by node like it happens on an Ethernet layer two switch.

IP network addresses

A **network address** is a unique identifier for a network. **Each different layer two network needs to have a different network address at layer three**, this is crucial because:

- Routers only retransmit IP packets between networks with different addresses (otherwise they are not different networks).
- When a node is sending a packet, it must know if the destination address is on its own network, if so, the IP packet can be **sent directly using layer two**, otherwise, the IP packet must be **sent to a router**.

At layer three, any node address must include in itself the network address, meaning the node belongs to that network.

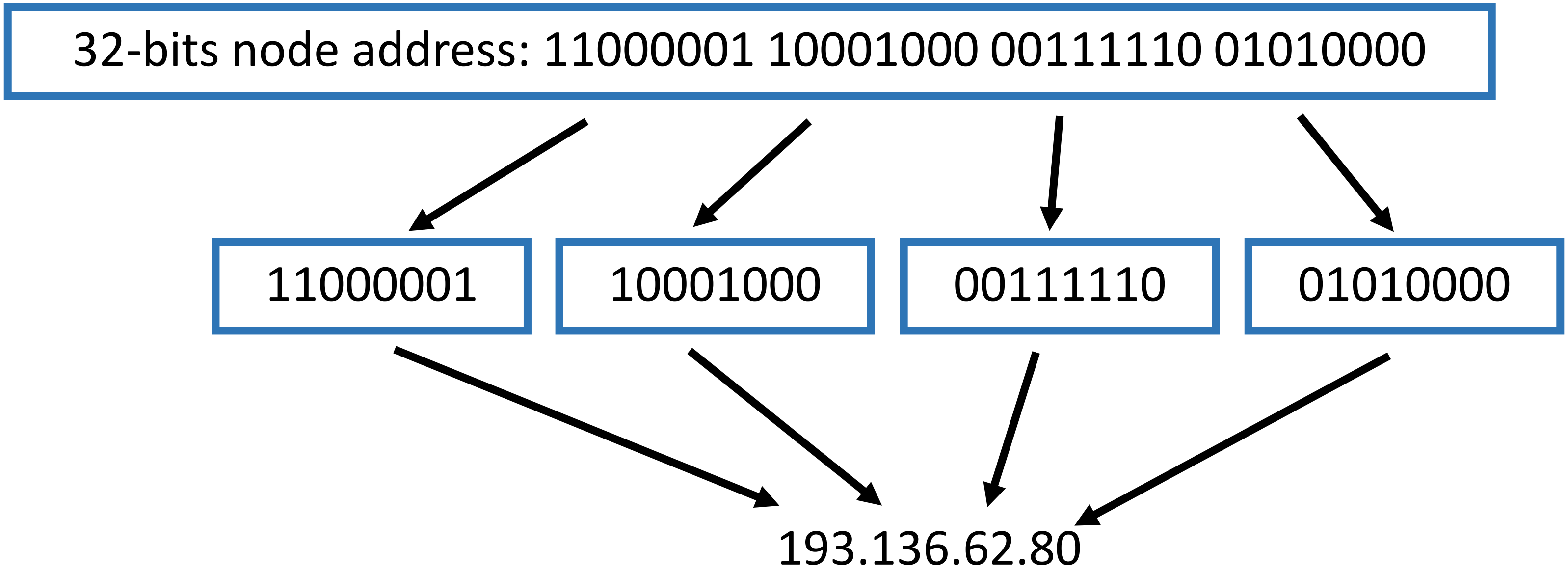
In IPv4 (and also IPv6) the most significant bits of a node address (leftmost bits) are in fact the **network address** or **network prefix**, the remaining (rightmost) bits represent the address of the node within that network.

Of course, two nodes belong to the same network if their addresses have the same network prefix, also, the remaining bits must be different because node addresses must be unique.

IPv4 address representation

The usual human readable representation for IPv4 addresses consists in dividing the 32-bits address into four 8-bits sets and represent each as a decimal number, separated by a dot (this is known as **Dot-decimal** notation).

Example:



32-bits node address: 11000001 10001000 00111110 01010000

11000001

10001000

00111110

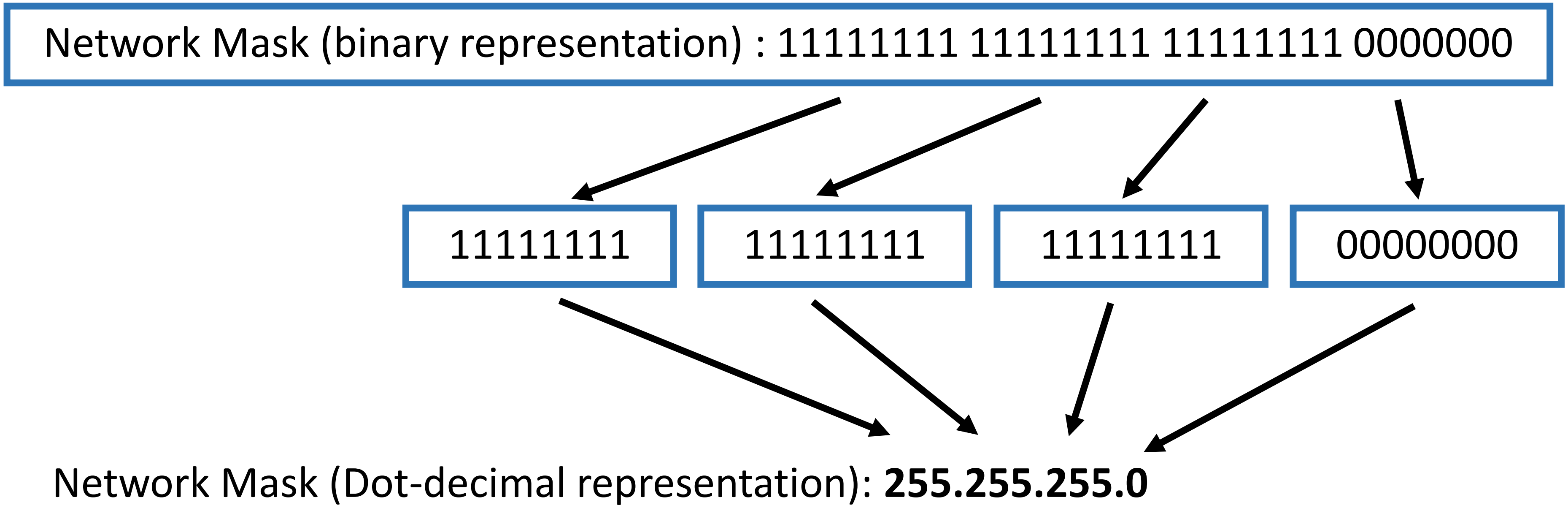
01010000

193.136.62.80

Thus, each of these four numbers can go from 0 to 255. We already know any IPv4 **node address** also contains a **network address** (the network the node belongs to). We also know the network address is the most significant bits of the node address, the doubt is how many bits are used for that purpose.

The network mask

The number most significant bits of the node address that represent the network address is determined by the **network mask**. A network mask is also a 32-bits number but is always made of a sequence on 1-bits corresponding to the most significant bits that are used to represent the network, followed by the remaining bits with value zero. Example:



Network Mask (binary representation) : 11111111 11111111 11111111 0000000

11111111 11111111 11111111 00000000

Network Mask (Dot-decimal representation): **255.255.255.0**

In this example, the network mask tells us the 24 most significant bits are used to identify the network. This network mask can also be referred to as a 24 bits **network prefix-length** or a 24 bits network mask.

IPv4 network address

An IPv4 **network address** is also represented as a 32-bits number, nevertheless, only the most significant bits are used to identify the network, therefore by convention, the remaining bits are represented as zero.

Yet, representing an IPv4 network address by just a 32-bits number is not enough because it says almost nothing about how many are the most significant bits being used to identify the network. **A network address requires the associated network mask or the prefix-length for proper interpretation**.

Take as example network address 160.0.0.0, because 160 value is 1010 0000 in a binary representation, this network address could make sense for any network mask of three bits or more (prefix-length greater than 3). With a network mask of less than three bits, the third bit at the network address can never be 1 (by convention must be represented as zero on the network address).

To identify an IPv4 network, two values are required: the **network address** (or **network prefix**) and the **network mask** or the **network prefix-length**.

IPv4 networks – maximum supported nodes

The network mask defines how many most significant bits are being used to identify the network (**network bits**), and therefore how many bits are left to identify hosts within the network (**host bits**).

If the network mask defines **N** network bits, then **32-N** bits are left for host bits. The number of possible different addresses is $\mathbf{2^{(32-N)}}$, this is also called the addresses block size for an **N** bits prefix-length.

Not all addresses in a block are valid IPv4 node addresses. When an address block is used for a network, the first address (all host bits zero) and the last address (all host bits one) are reserved. The first represents the network address itself, the second is reserved for IPv4 broadcasting.

Therefore, the maximum number of IPv4 nodes in a network with prefix-length **N** is: $\mathbf{2^{(32-N)} - 2}$

The biggest useable network prefix-length in IPv4 is 30 bits, resulting in a network capable of holding a maximum of only two hosts.

Some examples to clarify

Example 1: Network address 201.201.0.0 with network mask 255.255.0.0

The network mask has sixteen bits, this network can also be represented as **201.201.0.0/16**, know as CIDR (Classless Inter-Domain Routing) notation.

First of all, we can say the network address is consistent with the mask because, for every zero bit in the mask, the corresponding bit is also zero in the network address. If that was not so, then something was wrong, either the network address or the network mask.

Addresses within this network go from 201.201.0.0 to 201.201.255.255, the number of addresses is 2^{16} = 65536, however, the first (network address) and the last (network broadcast address) are reserved and can't be used as node addresses.

This leaves 2^{16} - 2 = 65534 valid node addresses in this network, valid node addresses go from 201.201.0.1 up to 201.201.255.254.

Example 2: Network address 192.168.100.0 with network mask 255.255.255.0

The network mask has twenty-four bits, this network can also be represented as **192.168.100.0/24** (CIDR notation).

Again, we can say the network address is consistent with the network mask because, for all zero bits at the network mask, the corresponding bits are also zero at the network address.

Addresses within this network go from 192.168.100.0 to 192.168.100.255, the number of addresses is 2^8 = 256, once more, the first (network address) and the last (network's broadcast address) are reserved and can't be used as node addresses. This leaves 2^8-2 = 254 valid node addresses in this network, valid node addresses range from 192.168.100.1 to 192.168.100.254.

The **network address** together with the **network mask** defines the network. They also provide a fast method to check if a given node address belongs to a network: by performing a bit to bit **and logical operation** between the **given address** and the **network mask**, we will obtain the **network address**. Examples:

[**192.168.10.3**] .AND. [255.255.255.0] = [**192.168.10.0**] (different network)

[**192.168.100.9**] .AND. [255.255.255.0] = [**192.168.100.0**] (same network)

[**193.16.100.1**] .AND. [255.255.255.0] = [**193.16.100.0**] (different network)

Classful and classless addresses

In early days, only three network prefix-lengths were used: 8, 16 and 24 bits, on those times **network masks were not required because they were implicit**, the most significant bits of the address were used to identify the prefix-length.

Any address starting by bit 0 was supposed to use a 8-bits mask (255.0.0.0) and they were called **class A**. Addresses starting by bits 10 would use a 16-bits mask (255.255.0.0), they were called **class B**. Addresses starting by bits 110 would use a 24-bits mask (255.255.255.0), they were called **class C**. This is called **classful addressing**. There is also class D, for addresses starting by 1110 bits and reserved for multicast addresses. Class E, addresses starting by 1111 bits are reserved for other uses.

Due to IPv4 address space exhaustion, classful addressing has been replaced by **classless** addressing (CIDR). Classless addressing means any network mask can be used. However, because the mask is no longer implicit, it must now be specified together with the network address.

Classless addressing allows a much more precise adjustment of the mask to the real needs under de point of view of maximum nodes to support in a network, and therefore saves a lot of addresses.

Classful versus classless addressing

Using classful addressing results in severe address wasting because there are only three alternatives for the maximum nodes supported by a network.

On internet backbones, many networks consist of dedicated connections between routers, these networks only require two node addresses (the connected routers), with classful addressing the best-adjusted mask for this purpose is 24-bits class C (255.255.255.0) which supports 254 nodes; thus 252 addresses are wasted. The same network using classless addressing can be coped with a 30-bits mask which supports exactly two nodes ($2^{(32-30)} - 2 = 2$).

Another example: imagine we want a network to support up to 500 hosts, with classful addressing the best-adjusted mask for this purpose is 16-bits class B (255.255.0.0) which supports $2^{(32-16)} - 2 = 65534$ nodes, thus 65034 node addresses are wasted. The same network using classless addressing can be coped with a 23-bits mask which supports $2^{(32-23)} - 2 = 510$ nodes, leaving only 10 unused addresses.

Using classless addressing presents though some challenges for human handling because the dot-decimal notation is no longer convenient. Every address assignment and validation will now require a binary representation for addresses, at least for the octet where the network mask is located.

Classless networks examples

Representing IPv4 addresses in dot-decimal notation is most convenient for network prefix-lengths of 8/16/24 bits, but otherwise, the mask will not match the octet division. A hexadecimal representation may help in some cases because each hexadecimal symbol will correspond to a set of four bits, so if the prefix-length is 4/8/16/20/24/28 we will be able to see it clearly. The general solution, however, is the use of **binary representation**.

Example 1: Network 192.168.100.192/29

It's valid because all bits with value zero in the network mask also have value zero in the network address, to check that we must represent both in binary. There's no point in representing all four octets in binary, our analysis will be focused on the octet where the prefix-length is located:

Network address: 192.168.100.$(1100\ 0\mathbf{000})_2$

Network mask: 255.255.255.$(1111\ 1\mathbf{000})_2$

(this network mask represented in dot-decimal comes as 255.255.255.248)

This network has $2^{(32-29)}$ = 8 addresses, thus 6 valid node addresses.

Getting the first valid node address is always easy, we can get it simply by adding one to the network address, thus it's **192.168.100.193**.

Getting the broadcast address can be achieved by analyzing the network address and mask in a binary representation, and then placing bits with value one on the host bits. Only the octet where the mask is placed really matters:

Network address: 192.168.100.(1100 0**000**)$_2$

Network mask: 255.255.255.(1111 1**000**)$_2$

So, the broadcast address is 192.168.100.(1100 0**111**)$_2$ = **192.168.100.199**

As always, the last valid node address may be obtained by subtracting one unit to the broadcast address, it's therefore: **192.168.100.198**.

Some more pragmatic approaches can also be used, once we know the size of the address space (eight addresses in this sample) we can determine **the next network address** by adding it to our network address:

192.168.100.192 + 8 = **192.168.100.200**.

Our network broadcast address can be obtained by subtracting one unit to the next network address: **192.168.100.199**.

The last valid node address in our network can then be obtained, as before, by subtracting one unit to the broadcast address: **192.168.100.198**.

This pragmatic approach can be used in any case: if we know the **network address space size** (from the network mask) and the **network address** we can get everything we need, generalizing:

First valid node address = (**Network address + 1)**

Next network address = (**Network address + Network address space size)**

Broadcast address = (**Next network address – 1)**

Last valid node address = (**Broadcast address – 1)**

Knowing the next network address is very important, because it tells us where to place additional networks. Remember however that this **next network** is **valid only for a network with an identical prefix-length**.

Calculating the **next network address** by adding the **network address space size** may be harder if the number to be added is greater than 256 (network prefix shorter than 24 bits). To make adding easier, the best solution is expressing these big numbers in 8-bits sets as well.

Also, calculating big powers of two to determine the **network address space size** may be hard. Nevertheless, the sequential determination of the powers of two is simple because each doubles the predecessor. Thus, we can quickly build a simple table with possible **network masks** and the corresponding **network address space sizes**.

To build this table we can use as reference the maximum possible prefix-length (30-bits in IPv4) or some other reference value like the 24-bits prefix-length, each time we reduce one bit to the mask, the **network address space size** doubles.

Network prefix-length (mask)	Network address space size	**Network address space size** (split in octets)
30	4	0.0.0.4
29	8	0.0.0.8
28	16	0.0.0.16
27	32	0.0.0.32
26	64	0.0.0.64
25	128	0.0.0.128
24	**256**	**0.0.1.0**
23	512	0.0.2.0
22	1024	0.0.4.0
21	2048	0.0.8.0
20	4096	0.0.16.0
19	8192	0.0.32.0
18	16384	0.0.64.0
17	32768	0.0.128.0
16	**65536**	**0.1.0.0**

Example 2: Network 10.16.64.0/18

It’s valid because all bits with value zero in the network mask also have the value zero in the network address, to check it we must represent both in binary. Again, there’s no point in representing all four octets in binary, our analysis will be on the octet where the prefix-length ends:

Network address: 10.16.(01**00 0000**)$_2$.**0**

Network mask: 255.255.(11**00 0000**)$_2$.**0**

(this network mask represented in dot-decimal comes as 255.255.192.0)

This network has 2$^{(32-18)}$ = 16384 addresses (as we can see in the earlier table).

Getting the first valid node address is always easy, we can get it simply by adding one to the network address, thus it’s **10.16.64.1**.

As we can see on the table, an eighteen bits prefix will correspond to a 16384 network address space size, it can also be represented as 0.0.64.0 (split in octets). Obtaining the **next network address** by adding this to our network address is now simple, it will be 10.16.(64+64).0: **10.16.128.0**.

Now, by removing one unit we get **our network broadcast address**: **10.16.127.255**

By removing another unit, we get **our network’s last valid node address**: **10.16.127.254**

Defining IPv4 networks – step one

This is a recurrent engineering problem network administrators face in real-world. They have an available address space (a network address and a prefix-length) and they are required to fit within that address space a set of networks, each supporting a given maximum number of hosts.

The **first step** is easy, determining the required prefix-length for each network. We can do that just by looking at the previously drawn table, remembering that the number of valid nodes is the network address space size minus two reserved addresses, and also that these valid nodes must include all IPv4 devices, including routers and printers for instance. The result from this first step is a network mask or prefix-length for each network.

Example: create one network able to hold up to 20 nodes, another network able to hold up to 1000 nodes, and yet another network able to hold up to 63 nodes. By looking at the table we come to:

- 20 nodes -> network prefix = 27 (up to 30 valid nodes)
- 1000 nodes -> network prefix = 22 (up to 1022 valid nodes)
- 63 nodes -> network prefix = 25 (up to 126 valid nodes)

Notice that for 63 nodes the 26-bits prefix is not enough as it only provides up to 62 valid nodes.

Defining IPv4 networks – step two

The **second step** is more tricky, we must now assign network addresses to each network without overlapping, and all within the provided address space.

We can start by checking if there's a solution, we can do this by summing the address space size of each network we want to create and check whether the result is less or equal to the size of the provided address space. Picking the example at the previous page, we have as required address space: 32 + 1024 + 128 = 1184. To solve the problem the provided address space must be at least 2048 (21-bits prefix), this is the biggest prefix (smallest address space) where 1184 addresses can be fitted.

Having checked the problem has a solution, now we can assign network addresses to each network.

One thing is important to understand: each address space (defined by a prefix-length) can always be split in two by increasing one bit on the prefix-length.

This means **wherever a network starts with given prefix-length, at that same point (with the same network address) also starts a network with any bigger prefix-length (smaller address spaces).**

The reverse, however, may not be true: given two networks with the same prefix-length, by reducing the prefix-length in one bit, we may not get a valid network containing the address spaces of the two original networks.

Defining IPv4 networks – step two

To assign addresses to each network we can take advantage of what was highlighted. First of all, we could place any of our networks at the start of the provided address space. Because all of them have a bigger prefix-lengths than the provided address space, they could start there.

As we have seen before, after assigning the first network we will know where the next network with an identical prefix-length starts, and bigger prefixes networks also can start there, so if we start **by assigning shorter prefixes first and then bigger prefixes, we can guarantee all addresses will be correct**.

By using this technique, once a network is assigned, we immediately know where to place the next network, remember however that **this is valid only if the next network prefix-length is equal or bigger**.

So, to assign addresses to our networks we start by sorting them accordingly to the prefix size, smaller prefix-lengths (bigger address spaces) first. Then we can sequentially assign to them addresses in the given address space.

Summarizing, the entire problem can be solved just by calculating some powers of two (building the table by doubling values) and performing some add operations.

This is not the only method to solve this kind of problem, but undoubtedly, it’s one that does not require significant resources.

Defining IPv4 networks – example

Let's pick an extensive example and solve the problem:

- We have the address block **120.20.32.0/20** to assign network addresses.
- We must provide addresses within this block to the following networks:
  - 2 networks with up to 300 nodes
  - 2 networks with up to 1000 nodes
  - 8 networks with up to 100 nodes

First step: networks' prefix-lengths

- 2 networks with up to 300 nodes	prefix-length: 23-bits (512 addresses)
- 2 networks with up to 1000 nodes	prefix-length: 22-bits (1024 addresses)
- 8 networks with up to 100 nodes	prefix-length: 25-bits (128 addresses)

Provided addresses: 20-bits prefix-length means: 4096 addresses

Required addresses: 2 x 512 + 2 x 1024 + 8 x 128 = 4096 **(it's just enough)**

Defining IPv4 networks – example

Second step: assign addresses sequentially starting by smaller prefixes

up to 1000 nodes network 1 - **120.20.32.0/22** (next network: 120.20.36.0)

up to 1000 nodes network 2 - **120.20.36.0/22** (next network: 120.20.40.0)

up to 300 nodes network 1 - **120.20.40.0/23** (next network: 120.20.42.0)

up to 300 nodes network 2 - **120.20.42.0/23** (next network: 120.20.44.0)

up to 100 nodes network 1 - **120.20.44.0/25** (next network: 120.20.44.128)

up to 100 nodes network 2 - **120.20.44.128/25** (next network: 120.20.45.0)

up to 100 nodes network 3 - **120.20.45.0/25** (next network: 120.20.45.128)

up to 100 nodes network 4 - **120.20.45.128/25** (next network: 120.20.46.0)

up to 100 nodes network 5 - **120.20.46.0/25** (next network: 120.20.46.128)

up to 100 nodes network 6 - **120.20.46.128/25** (next network: 120.20.47.0)

up to 100 nodes network 7 - **120.20.47.0/25** (next network: 120.20.47.128)

up to 100 nodes network 8 - **120.20.47.128/25** (next network: 120.20.48.0)

Network 120.20.48.0 is out of the provided address space, that’s ok, is just means we have completely exhausted the provided address space.

Private IPv4 addresses

Every IPv4 node on the internet must have a unique IPv4 node address, they are assigned directly or indirectly by IANA. Yet, some addresses are reserved for arbitrary and experimental local use, they are known as private addresses.

Private addresses are not recognized over the internet, so a node using a private IPv4 address can’t communicate directly using the internet. However, later we will see Network Address Translation (NAT) can be used to hide private addresses and allow these nodes to indirectly communicate with the internet.

IPv4 private addresses are:

10.0.0.0/8	(10.0.0.0 to 10.255.255.255)
172.16.0.0/12	(172.16.0.0 to 172.31.255.255)
192.168.0.0/16	(192.168.0.0 to 192.168.255.255)

These addresses are intended for local communications and experimental networks and should never reach the internet. A node using one of these addresses could send requests to the internet, but no reply will ever be received. Under the internet point of view, these are always unknown addresses.

Dynamic Host Configuration Protocol (DHCP)

Appropriately, end-user nodes (e.g., workstations, laptops, tablets, …) are not required to know in advance the network IPv4 configuration they are supposed to use. This includes their local unique IPv4 node address, the local network mask and the local network router (default-gateway).

All these IPv4 configuration data is required for the node to operate, but it can be retrieved by using specific automatic configuration protocols, notably DHCP. By using DHCP, when a node is connected to an unknown network, all required configuration data is supplied by the DHCP server. The configuration data contains everything required for the node to operate, including DNS servers' IPv4 addresses and DNS default domain name to be used.

Because DHCP makes use of broadcast requests, it will only work if there's a DHCP server on the same LAN (broadcast domain). The DHCP server operates by looking at clients' layer two addresses (MAC addresses) and ensuring each client will have a unique IPv4 address.

DHCP may be used to assign static node addresses, meaning a node will always get the same fixed address, however, for end-user nodes that's not relevant as it would be for a server, so usually the DHCP server manages a range of IPv4 node addresses (addresses pool) and assigns them freely as needed.

Automatic Private IP Addressing (APIPA)

When end-user nodes are configured to use an automatically assigned IPv4 address, they will use the DHCP protocol to retrieve the configuration data. However, that will fail if the DHCP service is not available at the local network.

The APIPA specification comes as a failover mechanism for such cases, it is implemented by some operating systems, like for instance Windows and MAC OS. The APIPA specification is documented in RFC 3927.

If no other mean of automatic configuration is successful (e.g. DHCP) , then the node may assign itself an IPv4 node address belonging to the special private network **169.254.0.0/16**, reserved by IANA for this purpose. The self-assigned IPv4 address must be then checked through ARP to ensure it's not being used already, otherwise another address must be self-assigned.

These addresses are also called IPv4 link-local addresses, they will allow communication within the local network only. Particularly, IPv4 routers are forbidden of forwarding IPv4 packets with these addresses.

It is important to mind that, as part of the APIPA specification, these addresses are temporary. After self-assigning this kind of IPv4 address, the node should persist in trying to use a standard automatic configuration mechanism (e.g. DHCP) and replace the address as soon as it is successful.