The Network Layer: The Internet Protocol v4

COMPUTER NETWORKS A.A. 24/25

Leonardo Maccari, DAIS: Ca' Foscari University of Venice,
leonardo.maccari@unive.it

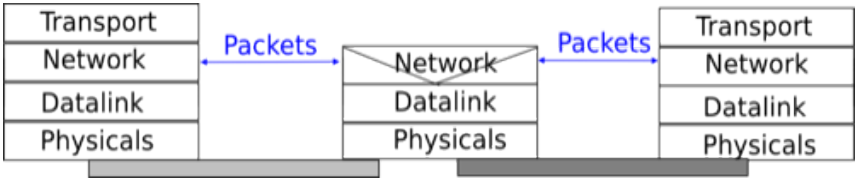
Venice, fall 2024

License

- The IPv6 slides contain material whose copyright is of Olivier Bonaventure, Universite catholique de Louvain, Belgium https://inl.info.ucl.ac.be and thus licensed under a Creative Commons Attribution-Share Alike 3.0 Unported License.
- The sole IPv4 part is derived from the course of Renato Lo Cigno, University of Brescia and re-used with his permission.

Sect. 1 The Network Layer

The network Layer in the Reference Model



Network Layer

- The Internet Network layer provides unreliable delivery of packets
- It uses the services offered by the datalink layer
- There are essentially two types of datalink layer that are used, point-to-point links and LANs

PPP: Point-to-Point Datalink Layers

- In a PPP link, there are only two devices communicating
- The frames do not need to be addressed, as every communication goes from one device to the other one

PPPoE/PPPoA

- A typical situation for a PPP datalink layer is the home DSL connection
- In that case, the home router needs to connect to another router on the other end of the cable, there is no routing/forwarding involved.
- The PPP datalink protocol can use several physical layer technologies in turn, for instance Ethernet, or ATM
- Your home router will be probably configured to use PPPoE, that stands for PPP over Ethernet

PPPoE/PPPoA

- The goal of a PPP protocol is simple (delivering frames on a physical media)
- However, it can provide also more elaborated features such as user authentication and encryption.

LANs

- A Local Area Network is a network of devices connected one to the other
- Every device has a unique address, most of the times, it is referenced to as a MAC address
- Since there are many devices and many addresses, frames in a LAN need to have a destination and source MAC address
- Furthermore they could be destined to a single host, or they could use a special destination address, a broadcast address.
- Frames destined to the broadcast address will be received by all the devices.

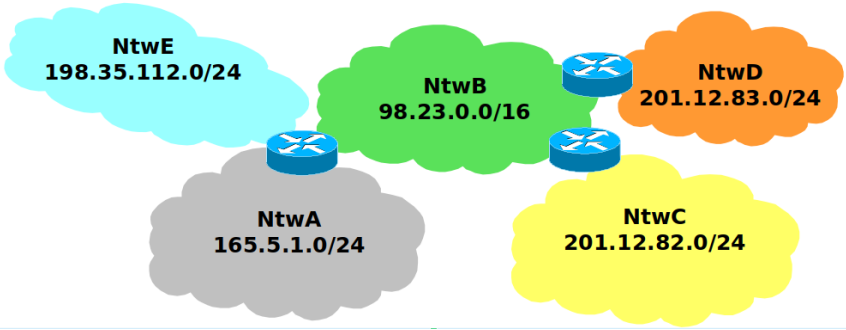
Sect. 2 IPv4

IPv4 Addresses

- Every host connected to a network has a number of Network Interface Cards (NIC).
- A NIC can be assigned an IP address.
- Most of the times, we assume that a computer that is not a router has only one NIC, so we say, for simplicity, the IP address of the computer (and not "of its NIC")
- IPv4 addresses are made of 32 bit, normally represented in "dotted decimal notation" (i.e. four decimal numbers from 0 to 255 separated by dots):
  - 1.2.3.4 == 00000001000000100000001100000100
  - 127.0.0.1 == 01111111000000000000000000000001
  - 255.255.255.255 == 11111111111111111111111111111111

Network Organization

- A network is organized into sub-networks (subnets) interconnected by routers
- A subnet is identified by a group of addresses in the form x.y.z.w/N, the number N after the "slash" is used to calculate the total number of addresses in the subnet: 2^{32-N}



Netmasks

- An IP address is thus divided into a prefix that identifies a subnet (netid) and a suffix that identifies the NIC inside the network (hostid)
- All NICs in the same subnet share the same prefix.
- Today we use classless representation, in which the subnet mask is explicitly defined:
  - 192.168.0.0/24: 24 bit for the network, 8 bit host (2^8 addresses)
  - 192.168.0.0/30: 30 bit for the network, 2 bit host (2^2 addresses)
  - 157.138.0.0/16: 16 bit for the network, 16 bit for the host (2^{16} addresses)
  - 10.0.0.0/8: 8 bit network, 24 bit host (2^{24} addresses)
  - ...
- While in the past the prefix size was constant and the address space divided into 3 classes: A (/8), B (/16), C (/24)

IP Addresses

- Ideally, every NIC in the world should have a unique IP address
- The netid is assigned by a global organization (IANA)
- The hostid must be assigned to the NIC by the local administrator
- Every subnet has two special addresses that are reserved, can not be assigned to a NIC:

  hostid $= 0$: this is the identifier of the network, and can not be used
  hostid $=$ 'all 1s': this is the broadcast address
- For example: 192.168.0.0/24:
  - 192.168.0 $\rightarrow$ netid
  - 192.168.0.0 $\rightarrow$ network address
  - 192.168.0.255 $\rightarrow$ broadcast address

Broadcast Address

- The broadcast address is used to send a packet to every IP in the subnet
- So if a NIC has address 1.2.3.4 in a /24 subnet and wants to send a packet to all the other hosts it will send it to 1.2.3.255
- There is a special case in which a NIC was still not assigned an IP address, but wants to send a broadcast packet
- Since it still does not know the netid, it can not send it
- Then it will use the "Limited Broadcast Address": 255.255.255.255
- Every NIC that receives a broadcast packet (both kinds) should pass it up in the stack, as if it was directed to the IP of the NIC itself

Multicast Addresses

- Addresses that start with 1110, that is 224.0.0.0/4 are multicast addresses
- These are used by groups of hosts that want to communicate, well known ones are:
  - 224.0.0.1 (all hosts in the subnet, excluding routers)
  - 224.0.0.2 (all routers)

Special Addresses

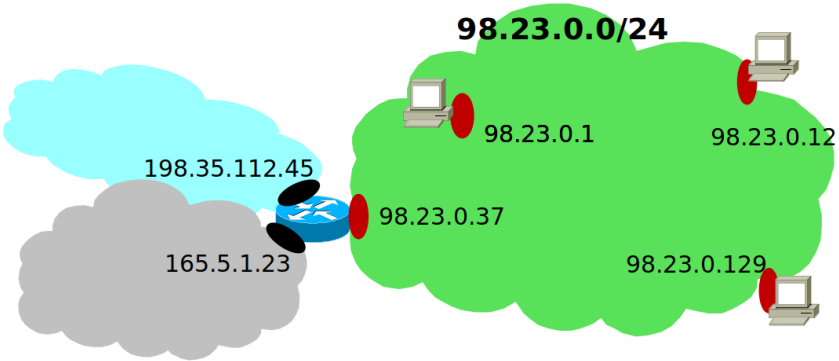
- 0.0.0.0/8: this machine, in this network. It can only be used as a source address, for example when using DHCP
- 127.0.0.0/8: loopback interface. Each IP host must have a virtual loopback interface, not attached to any physical interface, with address 127.0.0.1. It is used to manage TCP/IP connections between processes inside the machine
- 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16: non-routable private classes: it means that a router connected to the Internet must not route packets to/from these networks (later on we explain what *private* means)
- 169.254.0.0/16: link-local. Non-routable, it is used so that network interfaces that were not assigned an IP address yet, can communicate. Some OSs assign a random address in this class to the interfaces when they boot.

Intra-net communication

- We assume that all hosts in the same subnet can communicate one with the other
- They could be physically attached one to the other (like with an Ethernet switch) or they could be in the same Wi-Fi network, it is not important how
- However, a host inside a certain subnet can not directly communicate with an host on another subnet

Routers

- Routers are devices that belong to more than one network
- This means routers have more than one NIC, and each NIC has an address in a different subnet
- Thus, they can receive a packet from one network, and deliver it to another



Routers and subnets

- Every subnet, unless it is isolated, has at least one router
- Every host has one *default router* that it uses to send packets to when they are not directed to some IP address in the same subnet

Local Routes Vs Gateway

- When you configure an IP address to your PC, in the forwarding table a route is added automatically for the local network.
- Packets sent to some local IP address will be sent to the device, and the destination is assumed to be reachable directly (more on this when we explain MAC and ARP)

```
$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
2: enp0s31f6: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc fq_codel state DOWN group default
     qlen 1000
    link/ether c4:c6:e6:04:9f:26 brd ff:ff:ff:ff:ff:ff
4: wlp0s20f3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen
     1000
    link/ether b0:47:e9:8e:9e:c8 brd ff:ff:ff:ff:ff:ff
    inet 192.168.26.212/24 brd 192.168.26.255 scope global dynamic noprefixroute wlp0s20f3
```

- (1) Loopback interface; (2) wired interface, without IP address; (3) wireless interface with IP 192.168.26.212 in a /24 network

Local Routes Vs Gateway

- Instead, a routing entry that will use a gateway to reach the destination, will need to specify a gateway, which is generally specified with its IP
- This means that for instance, a typical routing table looks like:

```
$ ip r
default via 192.168.26.174 dev wlp0s20f3 proto dhcp metric 600
169.254.0.0/16 dev wlp0s20f3 scope link metric 1000
192.168.26.0/24 dev wlp0s20f3 proto kernel scope link src 192.168.26.212 metric 600
```

- Note the `via` keyword: that means, when you use this route (default: `0.0.0.0/0` you need to send packets to 192.168.26.174)
- However, the IP destination packet must be real destination. I can not change it.
- We will see when we explain the MAC layer, that the packet is sent to the MAC (layer 2 address) of the gateway, after ARP resolution.

Local Routes Vs Gateway: Bottom line

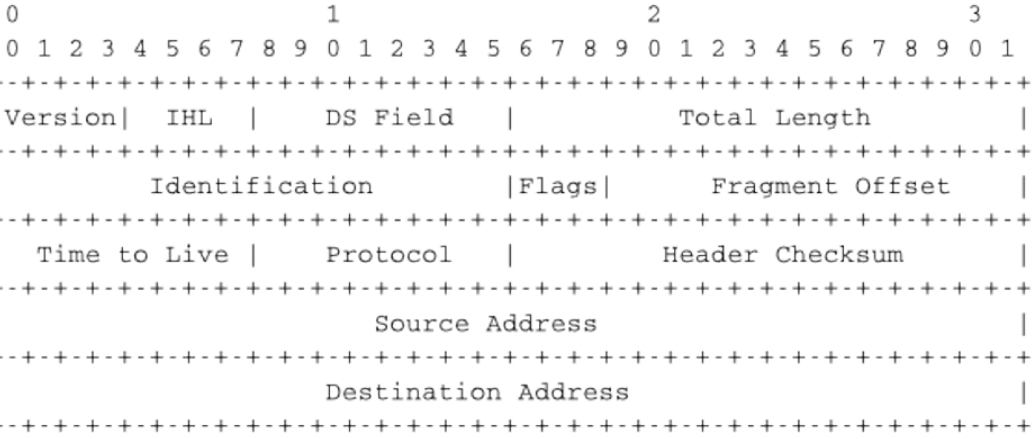
Example: local network `192.168.0.0/24`, gateway `192.168.0.1`.

- When a packet is sent to a host in the local network, for instance `192.168.1.10`, it is sent to destination IP `192.168.0.10` and the destination MAC is the one of the NIC assigned to `192.168.1.10`
- When a packet it sent to `8.8.8.8` (or any other different, non local network) the destination IP is `8.8.8.8` and the destination MAC is set to the MAC of the gateway, whose NIC has `IP 192.168.1.1`

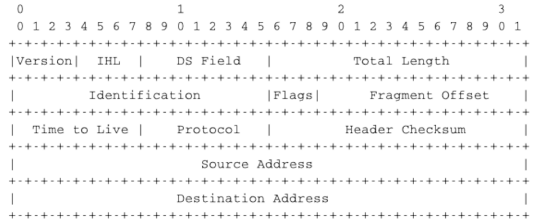
Multi-home hosts

- A NIC can be assigned more than one IP address
- This does not make the host a router, it just can have more than one address in the same network, or in two different networks
- However, it will not route packets from one to the other
- It is not a very common configuration (but possible, imagine your phone with both data network and Wi-Fi on)

IPv4 Header

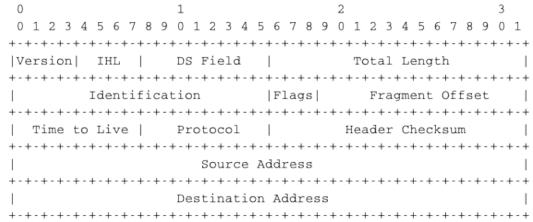

0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Version
IHL
DS Field
Total Length
Identification
Flags
Fragment Offset
Time to Live
Protocol
Header Checksum
Source Address
Destination Address

IPv4 Header



- Version: 4/6
- IP Header Length: number of 32 bit words (min 5 max 15)
- DS: QoS related, for instance this is where Explicit Congestion Notification can be stated (we don't look at these features)
- Total Length: length in Bytes of all the packet, max 65535B
- Identification: to match fragments (more later on)

IPv4 Header



- Flags: Evil Bit, Dont'fragment, More fragments
- Fragment Offset: offset of a fragment in the original packet (in multiples of 8 bytes. Only data, excludes the header)
- TTL: decremented at each hop
- Protocol: 1/ICMP, 6/TCP, 17/UDP
- Header Chk: checksum for the header data
- Source and Destination addresses

Header Options

- Besides the fixed header, there could be IP options, like source routing options.
- Unlike TCP these are rarely used, most of the packets do not contain any option.

IPv4

↳2.1 Packet Fragmentation

Fragmentation

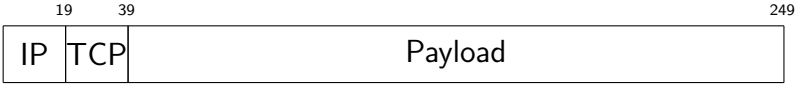
- We know that different datalinks can support different MTUs. For instance Ethernet: 1500; 802.11 (Wi-Fi): 2272
- A packet of size larger than the MTU of the network it is entering must be fragmented
- IP fragmentation is done at any router and reassembly is at destination

Fragmentation

- Each fragment of the same original IP packet has:
  - the same Identification field
  - length field set to the length of the fragment
  - a different offset, expressed in multiples of 8 bytes
  - the More Fragments flag set to 1, except the last one

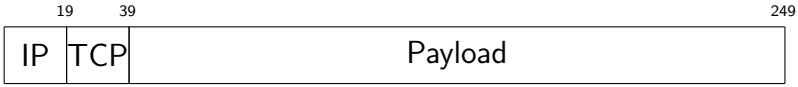
Example of Fragmentation (toy numbers)

- Initial IP packet: Total length = 250, MTU = 120

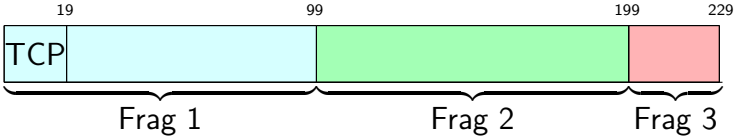


Example of Fragmentation (toy numbers)

- Initial IP packet: Total length = 250, MTU = 120

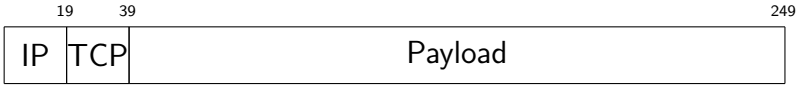


- Will be broken into:

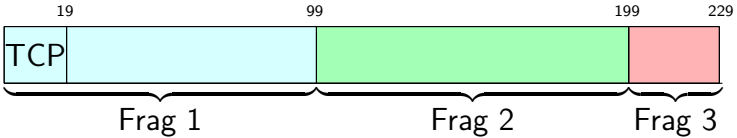


Example of Fragmentation (toy numbers)

- Initial IP packet: Total length = 250, MTU = 120



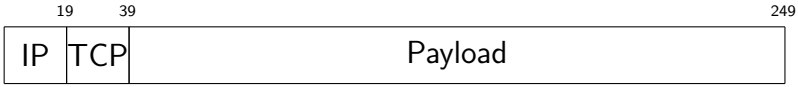
- Will be broken into:



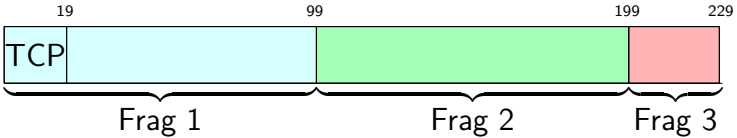
- Every fragment becomes a independent packet, with a new IP header that adds 20 Bytes to it.

Example of Fragmentation (toy numbers)

- Initial IP packet: Total length = 250, MTU = 120

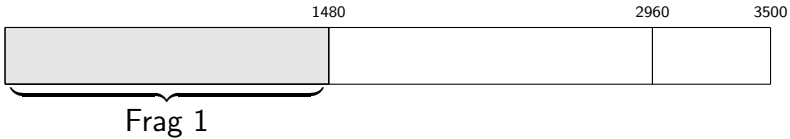


- Will be broken into:



- Every fragment becomes a independent packet, with a new IP header that adds 20 Bytes to it.
- The identification number remains the same in all fragments.

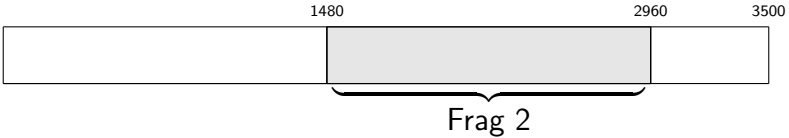
Example of Fragmentation (size 3500, MTU 1500)



- First Fragment (size: 20 Byte IP header + 1480 Byte payload <= MTU):

  Frag 1 contains the transport layer header, the IP header contains:
  - *more fragment* set to True, fragment offset is set to 0, length is set to 1480+20

Example of Fragmentation (size 3500, MTU 1500)

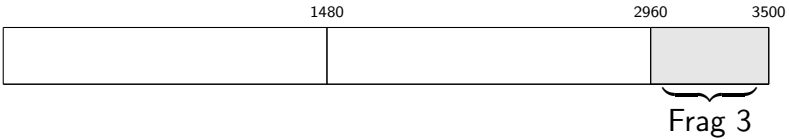


- Second Fragment (size: 20 Byte IP header + 1480 Byte payload <= MTU):

The IP header contains:

  - *more fragment* is set to True, fragment offset is **1480/8 = 185**, length is 1500

Example of Fragmentation (size 3500, MTU 1500)



- Third Fragment (size: 20 Byte IP header + 540 Byte payload <= MTU):

The IP header contains:

  - *more fragment* is set to False, fragment offset is **370**, length is 560

Fragments are Independent packets

- Every fragment travels to destination as an independent packet
- They could even take different paths
- Only when they arrive at destination, they are reassembled using the *Identification* and *offset* values, into one IP packet.
- The destination does not know who fragmented the packets, it can happen at any hop from source to destination
- Note that fragmentation introduces several problems, but can not be fully deprecated for back-compatibility[1]

[1] See `https://datatracker.ietf.org/doc/html/rfc8900`

IPv4

↳2.2 Public and Private IPv4 Addresses

IPv4 Address Shortage

- The total amount of IPv4 addresses is $2^{32} = 4.294.967.296$
- While this was a number considered sufficient in the 80s, today we need more addresses
- A long-term solution was to redesign IP, with a new version IPv6 that uses 128 bit addresses
- A short-term solution was to re-use some addresses, introducing Network Address Translation

Private IP Addresses

- Three classes of addresses were reserved for free use
  - 10.0.0.0 - 10.255.255.255 (/8)
  - 172.16.0.0 - 172.31.255.255 (/20)
  - 192.168.0.0 - 192.168.255.255 (/16)
- Network administrators can use them in their own subnets without the need of any coordination among them, these are called "private networks"
- A private network normally has a border router that connects it to the "public Internet" (that uses public IP addresses)
- Private networks can re-use the same IP addresses, for this reason a border router must not route to the public Internet packets from/to a private IP

Carrier-Grade NAT

- There is a fourth subnet that is considered private but not free to use: 100.64.0.0 - 100.127.255.255 (/10)
- This is used by ISPs when they don't have enough IP addresses to serve all their customers
- Even if technically there is nothing that prevents you from using this subnet, it may cause a collision with the address space of your ISP, and this could create issues.

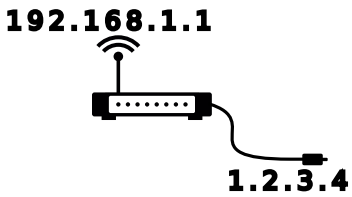
Network Address Translation - NAT

- NAT is a technique to match a public IP with a private one.
- NAT is a generic technique that sits in between the network and the transport layer, it can be configured in various ways
- The most common one is Network Address Port Translation (NAPT)
- The idea is that a border router will *translate* IP packets from the public IP space to the private one.

Step by Step NAT

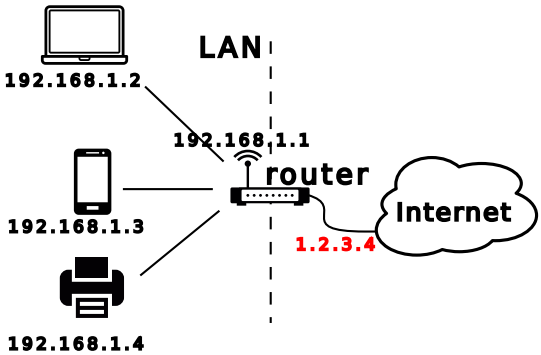
NAT is applied, for instance, in your home connection. Let's see how it works:

- You own a router with two NICs
- One wireless NIC that uses a private IP address (`192.168.1.1/24` in the figure). We call it the private NIC
- One wired NIC (that you plug to the wall with a cable) that uses a public IP address (`1.2.3.4` in the figure) provided by the Internet provider (the public NIC)



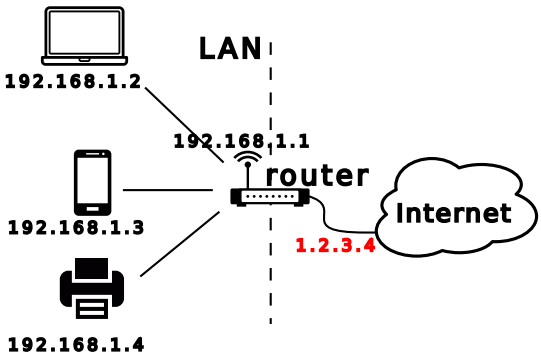
Step by Step NAT

- Each host in your network has a NIC, each NIC has a private IP: `192.168.1.X`
- The devices in your house form a local network (LAN).
- Inside the LAN each device must have a different IP. IPs can not be re-used in the same LAN.



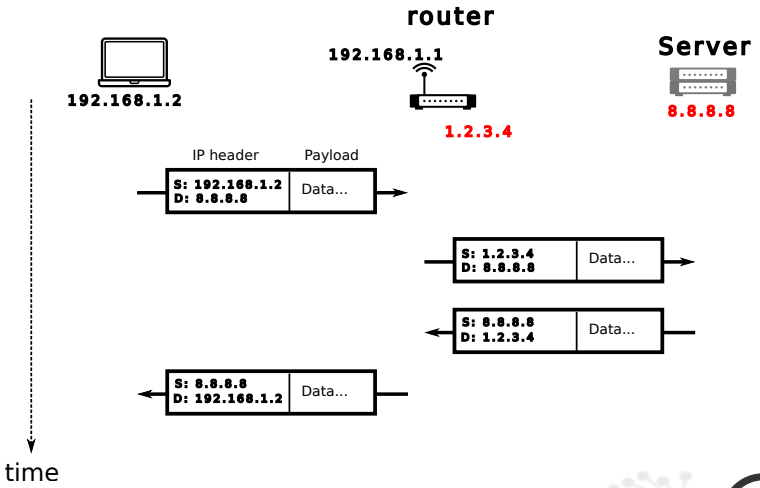
Step by Step NAT

- The router sits in between your LAN and the Internet
- Traffic that stays inside the LAN uses the private IPs, i.e.: **the IP packets use the private IPs in the packet headers**.
- The router *masquerades* private IP addresses behind its public IP address: the traffic going to the internet appears to be generated by `1.2.3.4`.



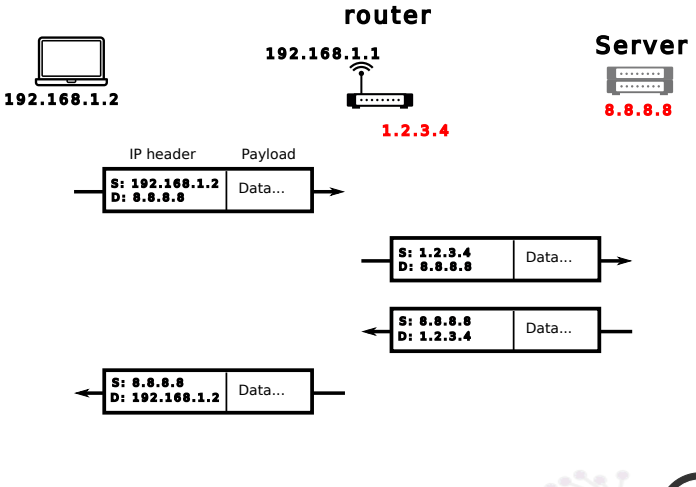
Step by Step NAT

- The router changes the source IP address in the IP header when sending packets to 8.8.8.8
- The server does not know about 192.168.1.2, it thinks that the sender is 1.2.3.4



Step by Step NAT

- When the server replies the router will replace 1.2.3.4 with the IP address of the original source
- The question is, how does the router know what private IP originally sent the first packet?



Flow Identifiers

- We said that one transport layer flow is uniquely identified by the tuple: `[sIP, dIP, sPort, dPort]`
- However, the same tuple can be used both for TCP and UDP, so a 5-elements tuple uniquely identifies a single flow on the Internet: `[Protocol, sIP, dIP, sPort, dPort]`
- So a router that implements NAT has a table like the following:

Incoming to the private NIC `[Protocol, sIP, dIP, sPort, dPort]`	Outgoing from the public NIC `[Protocol, sIP, dIP, sPort, dPort]`
UDP, 192.168.1.2, 8.8.8.8, 12909, 43	UDP, 1.2.3.4, 8.8.8.8, 12909, 43
TCP, 192.168.1.3, 4.3.2.1, 9823, 80	TCP, 1.2.3.4, 4.3.2 .1, 9823, 80

NAT Table: Details

Incoming to the private NIC [Protocol, sIP, dIP, sPort, dPort]	Outgoing from the public NIC [Protocol, sIP, dIP, sPort, dPort]
UDP, 192.168.1.2, 8.8.8.8, 12909, 43	UDP, 1.2.3.4, 8.8.8.8, 12909, 43
TCP, 192.168.1.3, 4.3.2.1, 9823, 80	TCP, 1.2.3.4, 4.3.2 .1, 9823, 80

- The table maps a packet that arrives to the private NIC or the router (192.168.1.1 in the previous figure) to a packet that will be sent on the public NIC (1.2.3.4)
- The table is used also to map incoming traffic to the public NIC to some host in the internal network.
- i.e.: if an UDP packet with `dIP=1.2.3.4`, `sIP=8.8.8.8`, `dPort=12909`, `sPort=43` arrives to the router on the public NIC, it will be remapped to `dIP=192.168.1.2` and sent on the private NIC

NAT Table: Port Mapping

- What if both `192.168.1.2` and `192.168.1.3` need to open a connection to `4.3.2.1` port 80 and they both decide to use source port 46000?
- Then the NAT table would look like

Incoming to the private NIC `[Protocol, sIP, dIP, sPort, dPort]`	Outgoing from the public NIC `[Protocol, sIP, dIP, sPort, dPort]`
TCP, 192.168.1.2, 4.3.2.1, 46000, 80	TCP, 1.2.3.4, 4.3.2.1, 46000, 80
TCP, 192.168.1.3, 4.3.2.1, 46000, 80	TCP, 1.2.3.4, 4.3.2.1, 46000, 80

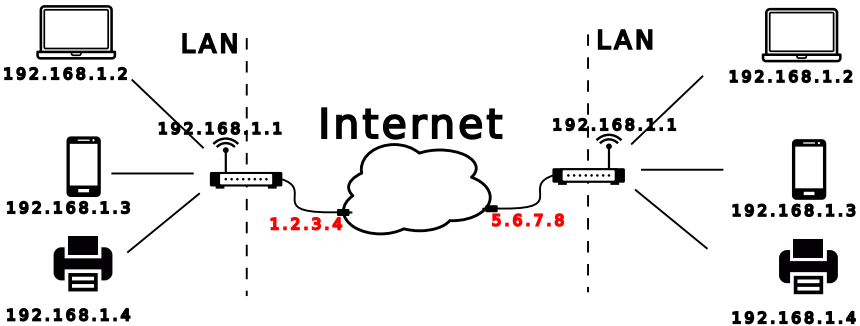
- This would clearly not work due to the duplicated entry.
- Then NAT will remap the source of the second connection to another one

Incoming to the private NIC `[Protocol, sIP, dIP, sPort, dPort]`	Outgoing from the public NIC `[Protocol, sIP, dIP, sPort, dPort]`
TCP, 192.168.1.2, 4.3.2.1, 46000, 80	TCP, 1.2.3.4, 4.3.2.1, 46000, 80
TCP, 192.168.1.3, 4.3.2.1, 46000, 80	TCP, 1.2.3.4, 4.3.2.1, 46001, 80

NAT Features

- Hosts that use private IP addresses are invisible to the rest of the Internet.
- When a NATed PC communicates with an Internet server, the server will receive traffic coming from the address of the router (`1.2.3.4`)
- When the server sends the traffic back, it will send it to the router address (`1.2.3.4`).
- The server does not know that the final destination is a host with a private address. From the server point of view, the destination is `1.2.3.4`.

2 NATs



- Two separate NATed networks can re-use the same private addresses
- Clients on any of the network can not connect to the other, unless the NAT devices use some specific protocol that allows it

NAT Take Away

- **Two hosts on the Internet can have the same private address, if they are on different LANs.**
- **Two hosts on the same LAN can not have the same private address.**
- **Two hosts on the Internet can not have the same public address.**
- **A host behind a NAT can not receive incoming connections (if not with certain tricks we don't explain).**
- **A server that needs to receive incoming connections must have a public IP address**

NATed hosts are *second-class citizens* on the Internet

NAT Issues (1)

NAT seems a simple solution, but has a lot of unwanted consequences:

- NAT forces a router to maintain a state (the NAT table). The whole IP layer is designed NOT to have a state. NAT devices are normally *firewall* devices, not simple routers.
- This means that the NAT must detect the beginning and the end of a traffic flow to purge the table. This is reasonably easy with TCP, not with UDP. And what happens if the TCP connection is interrupted without `RST`? NAT must cache the content for some time before removing it.
- a NAT can have memory issues when the internal network is made of thousands of connections

NAT Issues (2)

NAT seems a simple solution, but has a lot of unwanted consequences:

- NAT devices must reassembly the fragments (can you tell why?)
- This is increases the memory usage

NAT Issues (3)

NAT seems a simple solution, but has a lot of unwanted consequences:

- A host behind a NAT can open a TCP connection towards a host with a public IP, but not the opposite
- Two hosts behind a NAT can not communicate one with the other, if they don't use an external server

NAT Issues (4)

NAT seems a simple solution, but has a lot of unwanted consequences:

- Nat assumes TCP and/or UDP, plus some other known protocols (like ICMP)
- Due to NAT, the transport layer is very hard to change, because a new transport layer would break all the NATs. A NAT is the typical middlebox that does not allow cryptography at the transport layer.

Security Consequences

- Local networks use private IP addresses behind a NAT
- Internet hosts that want to be always reachable use public IP addresses.
- An IP address behind a NAT is less reachable than a public IP, so it can not be contacted/attacked by any PC on the Internet, just by the PCs in its own LAN.
- The NATted computer can still be attacked if it opens a connection to an attacker, but the attacker can not directly initiate an attack.
- These are generic considerations, not always true.

IPv4

↳2.3 Routing and Address Organization

Forwarding Tables

- A forwarding table maps a certain network address with a certain outgoing interface of a router
- When a packet arrives, it is matched against the contents of the forwarding table, and the interface corresponding to the line with the **longest network prefix match** is chosen. Example:

Destination Network	Outgoing NIC
0.0.0.0/0	0
124.156.12.0/24	1
124.156.13.0/24	0
124.156.13.128/26	2
200.212.12.127/26	3
200.212.12.64/28	4

Forwarding Tables

Destination Network	Out NIC
0.0.0.0/0	0
124.156.12.0/24	1
124.156.13.0/24	0
124.156.13.128/26	2
200.212.12.127/26	3
200.212.12.64/28	4

Where are the following destination IPs going to be routed?

- 124.156.12.12
- 124.156.13.2
- 124.156.13.129
- 200.212.12.79
- 200.212.12.35
- 200.156.12.1

Let's rewrite the table in Binary

Match		Network (red bits = netid)	NIC
?	0.0.0.0/0	00000000.00000000.00000000.00000000	0
?	124.156.12.0/24	01111100.10011100.00001100.00000000	1
?	124.156.13.0/24	01111100.10011100.00001101.00000000	0
?	124.156.13.128/26	01111100.10011100.00001101.10000000	2
?	200.212.12.127/26	11001000.11010100.00001100.01111111	3
?	200.212.12.64/28	11001000.11010100.00001100.01000000	4

And we consider:

124.156.12.12 == 01111100.10011100.00001100.00001100

Example 1

Match		Network (red bits = netid)	NIC
✔	0.0.0.0/0	00000000.00000000.00000000.00000000	0
✔	124.156.12.0/24	01111100.10011100.00001100.00000000	1
✘	124.156.13.0/24	01111100.10011100.00001101.00000000	0
✘	124.156.13.128/26	01111100.10011100.00001101.10000000	2
✘	200.212.12.127/26	11001000.11010100.00001100.01111111	3
✘	200.212.12.64/28	11001000.11010100.00001100.01000000	4

And we consider:

124.156.12.12 == 01111100.10011100.00001100.00001100

Then out NIC = 1, because two lines match, but we choose the one with the longest matching prefix.

Example 2

Match		Network (red bits = netid)	NIC
?	0.0.0.0/0	00000000.00000000.00000000.00000000	0
?	124.156.12.0/24	01111100.10011100.00001100.00000000	1
?	124.156.13.0/24	01111100.10011100.00001101.00000000	0
?	124.156.13.128/26	01111100.10011100.00001101.10000000	2
?	200.212.12.127/26	11001000.11010100.00001100.01111111	3
?	200.212.12.64/28	11001000.11010100.00001100.01000000	4

And we consider:

124.156.13.2 ?

Example 2

Match		Network (red bits = netid)	NIC
✔	0.0.0.0/0	00000000.00000000.00000000.00000000	0
✘	124.156.12.0/24	01111100.10011100.00001100.00000000	1
✔	124.156.13.0/24	01111100.10011100.00001101.00000000	0
✘	124.156.13.128/26	01111100.10011100.00001101.10000000	2
✘	200.212.12.127/26	11001000.11010100.00001100.01111111	3
✘	200.212.12.64/28	11001000.11010100.00001100.01000000	4

And we consider:

$$124.156.13.2 == 01111100.10011100.00001101.00000010$$

Then out NIC = 0.

Example 3

Match		Network (red bits = netid)	NIC
?	0.0.0.0/0	00000000.00000000.00000000.00000000	0
?	124.156.12.0/24	01111100.10011100.00001100.00000000	1
?	124.156.13.0/24	01111100.10011100.00001101.00000000	0
?	124.156.13.128/26	01111100.10011100.00001101.10000000	2
?	200.212.12.127/26	11001000.11010100.00001100.01111111	3
?	200.212.12.64/28	11001000.11010100.00001100.01000000	4

And we consider:

124.156.13.129 ?

Example 3

Match		Network (red bits = netid)	NIC
✔	0.0.0.0/0	00000000.00000000.00000000.00000000	0
✘	124.156.12.0/24	01111100.10011100.00001100.00000000	1
✔	124.156.13.0/24	01111100.10011100.00001101.00000000	0
✔	124.156.13.128/26	01111100.10011100.00001101.10000000	2
✘	200.212.12.127/26	11001000.11010100.00001100.01111111	3
✘	200.212.12.64/28	11001000.11010100.00001100.01000000	4

And we consider:

124.156.13.129 == 01111100.10011100.00001101.10000001

Then out NIC = 2.

Example 4

Match		Network (red bits = netid)	NIC
?	0.0.0.0/0	00000000.00000000.00000000.00000000	0
?	124.156.12.0/24	01111100.10011100.00001100.00000000	1
?	124.156.13.0/24	01111100.10011100.00001101.00000000	0
?	124.156.13.128/26	01111100.10011100.00001101.10000000	2
?	200.212.12.127/26	11001000.11010100.00001100.01111111	3
?	200.212.12.64/28	11001000.11010100.00001100.01000000	4

And we consider:

200.212.12.79 ?

Example 4

Match		Network (red bits = netid)	NIC
✔	0.0.0.0/0	00000000.00000000.00000000.00000000	0
✘	124.156.12.0/24	01111100.10011100.00001100.00000000	1
✘	124.156.13.0/24	01111100.10011100.00001101.00000000	0
✘	124.156.13.128/26	01111100.10011100.00001101.10000000	2
✘	200.212.12.127/26	11001000.11010100.00001100.01111111	3
✔	200.212.12.64/28	11001000.11010100.00001100.01000000	4

And we consider:

200.212.12.79 == 11001000.11010100.00001100.01001111

Then out NIC = 4.

Example 5

Match		Network (red bits = netid)	NIC
?	0.0.0.0/0	00000000.00000000.00000000.00000000	0
?	124.156.12.0/24	01111100.10011100.00001100.00000000	1
?	124.156.13.0/24	01111100.10011100.00001101.00000000	0
?	124.156.13.128/26	01111100.10011100.00001101.10000000	2
?	200.212.12.127/26	11001000.11010100.00001100.01111111	3
?	200.212.12.64/28	11001000.11010100.00001100.01000000	4

And we consider:

200.212.12.35 ?

Example 5

Match		Network (red bits = netid)	NIC
✔	0.0.0.0/0	00000000.00000000.00000000.00000000	0
✘	124.156.12.0/24	01111100.10011100.00001100.00000000	1
✘	124.156.13.0/24	01111100.10011100.00001101.00000000	0
✘	124.156.13.128/26	01111100.10011100.00001101.10000000	2
✘	200.212.12.127/26	11001000.11010100.00001100.01111111	3
✘	200.212.12.64/28	11001000.11010100.00001100.01000000	4

And we consider:

200.212.12.35 == 11001000.11010100.00001100.00100011

Then out NIC = 0.

Example 6

Match		Network (red bits = netid)	NIC
?	0.0.0.0/0	00000000.00000000.00000000.00000000	0
?	124.156.12.0/24	01111100.10011100.00001100.00000000	1
?	124.156.13.0/24	01111100.10011100.00001101.00000000	0
?	124.156.13.128/26	01111100.10011100.00001101.10000000	2
?	200.212.12.127/26	11001000.11010100.00001100.01111111	3
?	200.212.12.64/28	11001000.11010100.00001100.01000000	4

And we consider:

200.156.12.1 ?

Example 6

Match		Network (red bits = netid)	NIC
✔	0.0.0.0/0	00000000.00000000.00000000.00000000	0
✘	124.156.12.0/24	01111100.10011100.00001100.00000000	1
✘	124.156.13.0/24	01111100.10011100.00001101.00000000	0
✘	124.156.13.128/26	01111100.10011100.00001101.10000000	2
✘	200.212.12.127/26	11001000.11010100.00001100.01111111	3
✘	200.212.12.64/28	11001000.11010100.00001100.01000000	4

And we consider:

200.156.12.1 == 11001000.10011100.00001100.00000001

Then out NIC = 0.

Data Structures

- In this simple example, we used a sorted list, so the algorithm would scan the list from bottom to top, and return the first match.
- In real routers, this is done in hardware, using some dedicated logic circuits
- At this link you can find a description of how the forwarding table is implemented in the Linux Kernel.

IP Addressing: Exercise 0

- I own a subnet 1.2.1.0/24. Can I assign to one of my host the address 1.2.1.255?

IP Addressing: Exercise 0

- I own a subnet 1.2.1.0/24. Can I assign to one of my host the address 1.2.1.255?
- NO! the netid is 1.2.1, and the all one hostid is for the broadcast address

IP Addressing: Exercise 0

- I own a subnet 1.2.1.0/24. Can I assign to one of my host the address 1.2.1.255?
- NO! the netid is 1.2.1, and the all one hostid is for the broadcast address
- I own a subnet 1.2.0.0/16. Can I assign to one of my host the address 1.2.1.0?

IP Addressing: Exercise 0

- I own a subnet 1.2.1.0/24. Can I assign to one of my host the address 1.2.1.255?
- NO! the netid is 1.2.1, and the all one hostid is for the broadcast address
- I own a subnet 1.2.0.0/16. Can I assign to one of my host the address 1.2.1.0?
- YES! the netid is 1.2, the network address is 1.2.0.0 (all zero hostid), so 1.2.1.0 is a valid IP address.

IP Addressing: Exercise 0

- I own a subnet 1.2.1.0/24. Can I assign to one of my host the address 1.2.1.255?
- NO! the netid is 1.2.1, and the all one hostid is for the broadcast address
- I own a subnet 1.2.0.0/16. Can I assign to one of my host the address 1.2.1.0?
- YES! the netid is 1.2, the network address is 1.2.0.0 (all zero hostid), so 1.2.1.0 is a valid IP address.
- I own a subnet 1.2.1.0/24. Can I assign to one of my host the address 1.2.1.0?

IP Addressing: Exercise 0

- I own a subnet 1.2.1.0/24. Can I assign to one of my host the address 1.2.1.255?
- NO! the netid is 1.2.1, and the all one hostid is for the broadcast address
- I own a subnet 1.2.0.0/16. Can I assign to one of my host the address 1.2.1.0?
- YES! the netid is 1.2, the network address is 1.2.0.0 (all zero hostid), so 1.2.1.0 is a valid IP address.
- I own a subnet 1.2.1.0/24. Can I assign to one of my host the address 1.2.1.0?
- NO! the netid is 1.2.0, and the all zero hostid is for the netid

IP Addressing: Exercise 1

- An organization needs 15 IP addresses and must lease them by a network provider
- Assume the network provider owns 1.2.0.0/16
- What is the minimum length netid that the the organization needs to lease?
- Assign the network addresses, start numbering from "all 0", for instance, if you need a /24, you should use 1.2.0.0/24 (third byte set to 0). State specifically the allowed range of addresses.

IP Addressing: Exercise 1

- To contain 15 addresses you need 4 bits, so 1.2.0.0/28 would be sufficient: from 1.2.0.0 to 1.2.0.15 (16 IPs, of which 15 will be actually used).

IP Addressing: Exercise 1

- To contain 15 addresses you need 4 bits, so 1.2.0.0/28 would be sufficient: from 1.2.0.0 to 1.2.0.15 (16 IPs, of which 15 will be actually used).
- However...

IP Addressing: Exercise 1

- To contain 15 addresses you need 4 bits, so 1.2.0.0/28 would be sufficient: from 1.2.0.0 to 1.2.0.15 (16 IPs, of which 15 will be actually used).
- However...
- Every subnet has two addresses that can't be used, in this case:

IP Addressing: Exercise 1

- To contain 15 addresses you need 4 bits, so 1.2.0.0/28 would be sufficient: from 1.2.0.0 to 1.2.0.15 (16 IPs, of which 15 will be actually used).
- However...
- Every subnet has two addresses that can't be used, in this case:
  - 1.2.0.0 (network address)

IP Addressing: Exercise 1

- To contain 15 addresses you need 4 bits, so 1.2.0.0/28 would be sufficient: from 1.2.0.0 to 1.2.0.15 (16 IPs, of which 15 will be actually used).
- However...
- Every subnet has two addresses that can't be used, in this case:
  - 1.2.0.0 (network address)
  - 1.2.0.15 (15 == 00001111) (broadcast address)

IP Addressing: Exercise 1

- To contain 15 addresses you need 4 bits, so 1.2.0.0/28 would be sufficient: from 1.2.0.0 to 1.2.0.15 (16 IPs, of which 15 will be actually used).
- However...
- Every subnet has two addresses that can't be used, in this case:
  - 1.2.0.0 (network address)
  - 1.2.0.15 (15 == 00001111) (broadcast address)
- So the only way to have 15 addresses is by leasing 1.2.0.0/27 that provides 30 usable addresses, of which only 15 will be actually used.

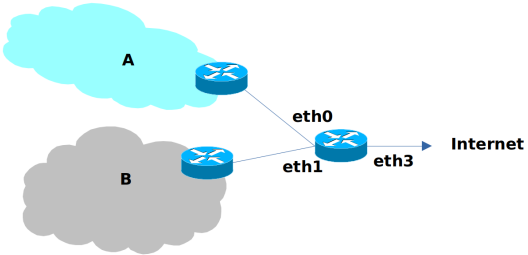
IP Addressing: Exercise 1

- To contain 15 addresses you need 4 bits, so 1.2.0.0/28 would be sufficient: from 1.2.0.0 to 1.2.0.15 (16 IPs, of which 15 will be actually used).
- However...
- Every subnet has two addresses that can't be used, in this case:
  - 1.2.0.0 (network address)
  - 1.2.0.15 (15 == 00001111) (broadcast address)
- So the only way to have 15 addresses is by leasing 1.2.0.0/27 that provides 30 usable addresses, of which only 15 will be actually used.
- The available addresses will range from 1.2.0.0 to 1.2.0.31, of which 30 are usable (hostid from 1 to 30)

Router IP

- Note also that every network has a router
- So, unless your router is not a dedicated hardware, but it's a computer that you can use, one of your IP addresses will be occupied by the router

IP Addressing: Exercise 2



- An organization has two chapters, A and B
- In network A there are 14 hosts, in network B there are 16
- The provider owns 1.2.0.0/16
- Find out:
  - The minimum network mask that the organization needs to lease from the provider
  - A possible allocation and ranges, this time assume "all 1" for the net id
  - The routing tables of the router connected to the Internet

IP Addressing: Exercise 2

- The total number of required IP addresses is 30, so a /27 netmask is sufficient

IP Addressing: Exercise 2

- The total number of required IP addresses is 30, so a /27 netmask is sufficient
- However the administrator wants to partition the network in two. So every subnet wastes 2 IP addresses, so he needs 30+2*2=34 addresses

IP Addressing: Exercise 2

- The total number of required IP addresses is 30, so a /27 netmask is sufficient
- However the administrator wants to partition the network in two. So every subnet wastes 2 IP addresses, so he needs 30+2*2=34 addresses
- That requires a /26 network, $2^6 = 64$ addresses of which he will only use 30.

IP Addressing: Exercise 2

- The total number of required IP addresses is 30, so a /27 netmask is sufficient
- However the administrator wants to partition the network in two. So every subnet wastes 2 IP addresses, so he needs 30+2*2=34 addresses
- That requires a /26 network, $2^6 = 64$ addresses of which he will only use 30.
- He leases 1.2.255.192/26 (192 == 11000000) and split it in two subnets:

IP Addressing: Exercise 2

- The total number of required IP addresses is 30, so a /27 netmask is sufficient
- However the administrator wants to partition the network in two. So every subnet wastes 2 IP addresses, so he needs 30+2*2=34 addresses
- That requires a /26 network, $2^6 = 64$ addresses of which he will only use 30.
- He leases 1.2.255.192/26 (192 == 11000000) and split it in two subnets:
- 1.2.255.192/27 (192 == 11000000) and 1.2.255.224/27 (224 == 11100000).

IP Addressing: Exercise 2

- 1.2.255.192/27 (192 == 11000000): This will provide addresses from 1.2.255.192 to 1.2.255.223. These are 32 addresses of which the last and the first can not be used

IP Addressing: Exercise 2

- 1.2.255.192/27 (192 == 11000000): This will provide addresses from 1.2.255.192 to 1.2.255.223. These are 32 addresses of which the last and the first can not be used
- 1.2.255.224/27 (224 == 11100000): This will provide addresses from 1.2.255.224 to 1.2.255.255. These are 32 possible addresses of which the first and the last can not be used

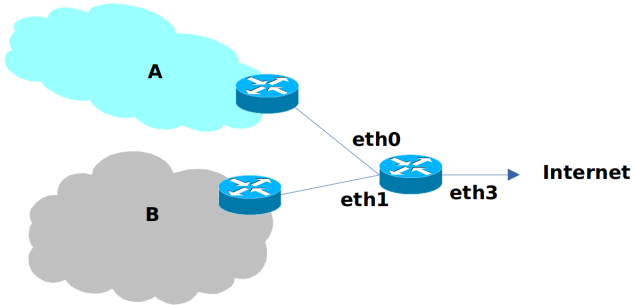
IP Addressing: Exercise 2

- 1.2.255.192/27 (192 == 11000000): This will provide addresses from 1.2.255.192 to 1.2.255.223. These are 32 addresses of which the last and the first can not be used
- 1.2.255.224/27 (224 == 11100000): This will provide addresses from 1.2.255.224 to 1.2.255.255. These are 32 possible addresses of which the first and the last can not be used
- In both networks we have enough IP addresses for the hosts and for the router too.

IP Addressing: Exercise 2

The FIB will be:

Destination Network	Out NIC
0.0.0.0/0	eth3
1.2.255.192/27	eth1
1.2.255.224/27	eth0



IP Addressing: Exercise 2

- However, IP addresses cost, and the administrator is not happy of leasing 64 addresses and using only 30 of them. Is there a better solution?

IP Addressing: Exercise 2

- However, IP addresses cost, and the administrator is not happy of leasing 64 addresses and using only 30 of them. Is there a better solution?
- One of the network fits a /28 subnet, and the other fits a /27.

IP Addressing: Exercise 2

- However, IP addresses cost, and the administrator is not happy of leasing 64 addresses and using only 30 of them. Is there a better solution?
- One of the network fits a /28 subnet, and the other fits a /27.
- So he could lease two different subnets

IP Addressing: Exercise 2

- However, IP addresses cost, and the administrator is not happy of leasing 64 addresses and using only 30 of them. Is there a better solution?
- One of the network fits a /28 subnet, and the other fits a /27.
- So he could lease two different subnets
- 1.2.255.192/27 (192 == 11000000): This will provide addresses from 1.2.255.192 to 1.2.255.223. These are 30 usable addresses for network B

IP Addressing: Exercise 2

- However, IP addresses cost, and the administrator is not happy of leasing 64 addresses and using only 30 of them. Is there a better solution?
- One of the network fits a /28 subnet, and the other fits a /27.
- So he could lease two different subnets
- 1.2.255.192/27 (192 == 11000000): This will provide addresses from 1.2.255.192 to 1.2.255.223. These are 30 usable addresses for network B
- 1.2.255.224/28 (224 == 11100000): This will provide addresses from 1.2.255.224 to 1.2.255.239. These are 14 usable addresses for network A

IP Addressing: Exercise 2

- He still owns more addresses than what he needs (32+16=48) but he will pay less than before.

IP Addressing: Exercise 2

- He still owns more addresses than what he needs (32+16=48) but he will pay less than before.
- However, he owns two public subnets, and not only one, as he does not cover 1.2.255.240/28 (240 == 11110000).

IP Addressing: Exercise 2

- He still owns more addresses than what he needs (32+16=48) but he will pay less than before.
- However, he owns two public subnets, and not only one, as he does not cover 1.2.255.240/28 (240 == 11110000).
- The FIB will be:

Destination Network	Out NIC
0.0.0.0/0	eth3
1.2.255.192/27	eth1
1.2.255.224/28	eth0

IP Addressing: Exercise 2

- He still owns more addresses than what he needs (32+16=48) but he will pay less than before.
- However, he owns two public subnets, and not only one, as he does not cover 1.2.255.240/28 (240 == 11110000).
- The FIB will be:

Destination Network	Out NIC
0.0.0.0/0	eth3
1.2.255.192/27	eth1
1.2.255.224/28	eth0

- So a packet with dest. IP 1.2.255.250 will be sent to the default gateway. This is correct because the range 240-255 is not part of the local network anymore.

IP Addressing: Exercise 2

- Note that in this solution one of the IP in the /27 will be used for the router
- This means that the router should probably be a computer, as it occupies one IP of the 14 needed, so it probably needs to host some services
- So this is a cheap, but not a very scalable solution as a computer is not a specialized hardware, and software routing is slower