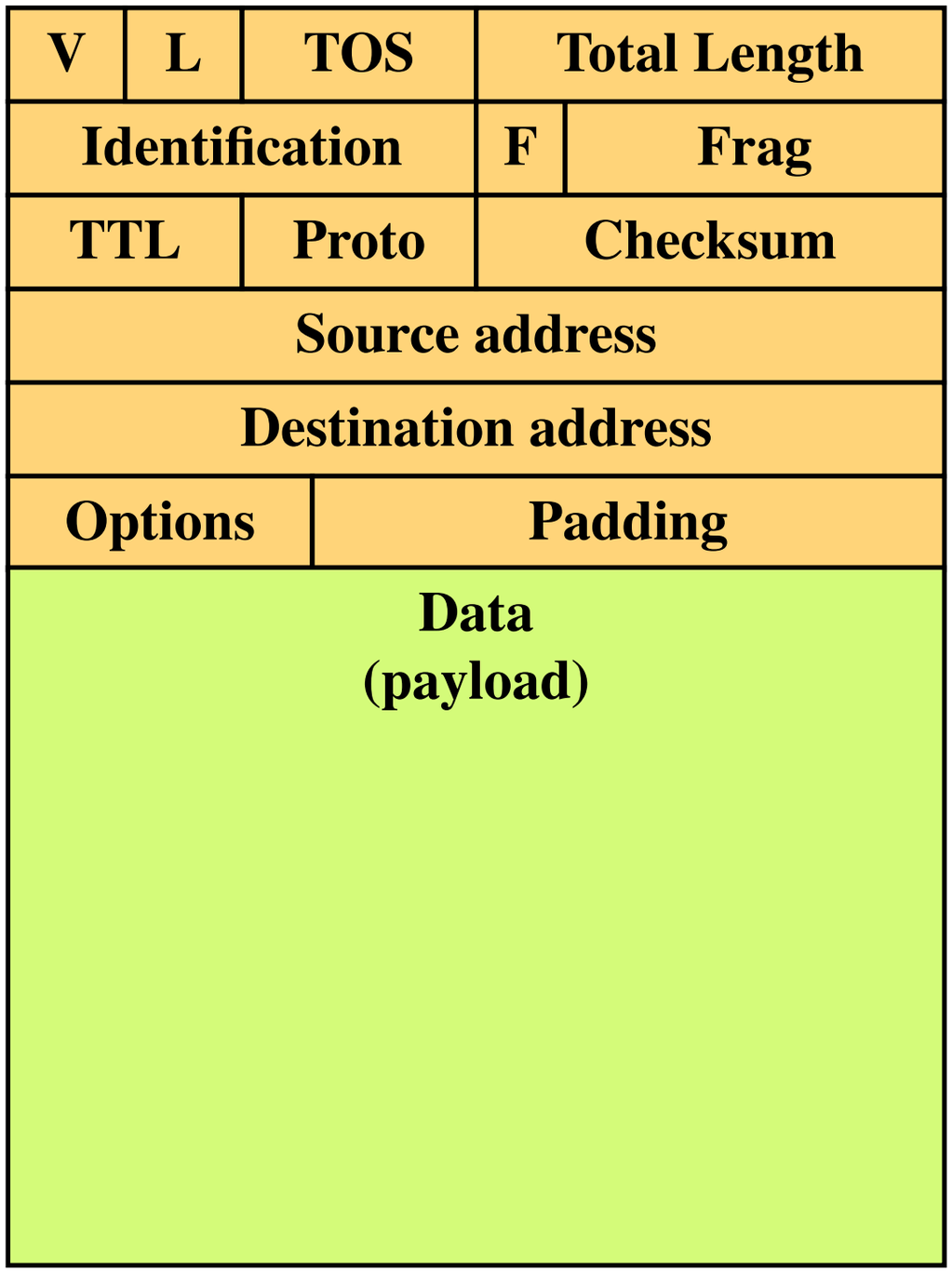
Internet Protocol: IP packet headers

TELECOM ParisTech

IPv4 header



- **V**: Version (IPv4 ; IPv6)
- **L**: Header length (variable because of options)
- **TOS**: *Type of Service* - flow class
- **Total length**: packet length
- **Identification / F / Frag**: fragmentation
- **TTL**: *Time to* Live *;* maximum number of routers traversals authorized
- **Proto**: upper level protocol (TCP or UDP)
- **Checksum**: header integrity check
- **Source address**
- **Destination address**
- **Options** (security, ...)
- **Padding**: to align length to a 32 bits multiple

IPv6: header evolution

V	L	TOS	Total Length
Identification		F	Frag
TTL	Proto	Checksum	
Source address			
Destination address			
Options	Padding		

IPv6: header evolution

V	L	TOS	Total Length
Identification		F	Frag
TTL	Proto	Checksum	
Source address			
Destination address			
Options	Padding		

V	Class	Flow Label	
Payload length		Next H	Hop lim
Source address			
Destination address			

IPv6: header evolution

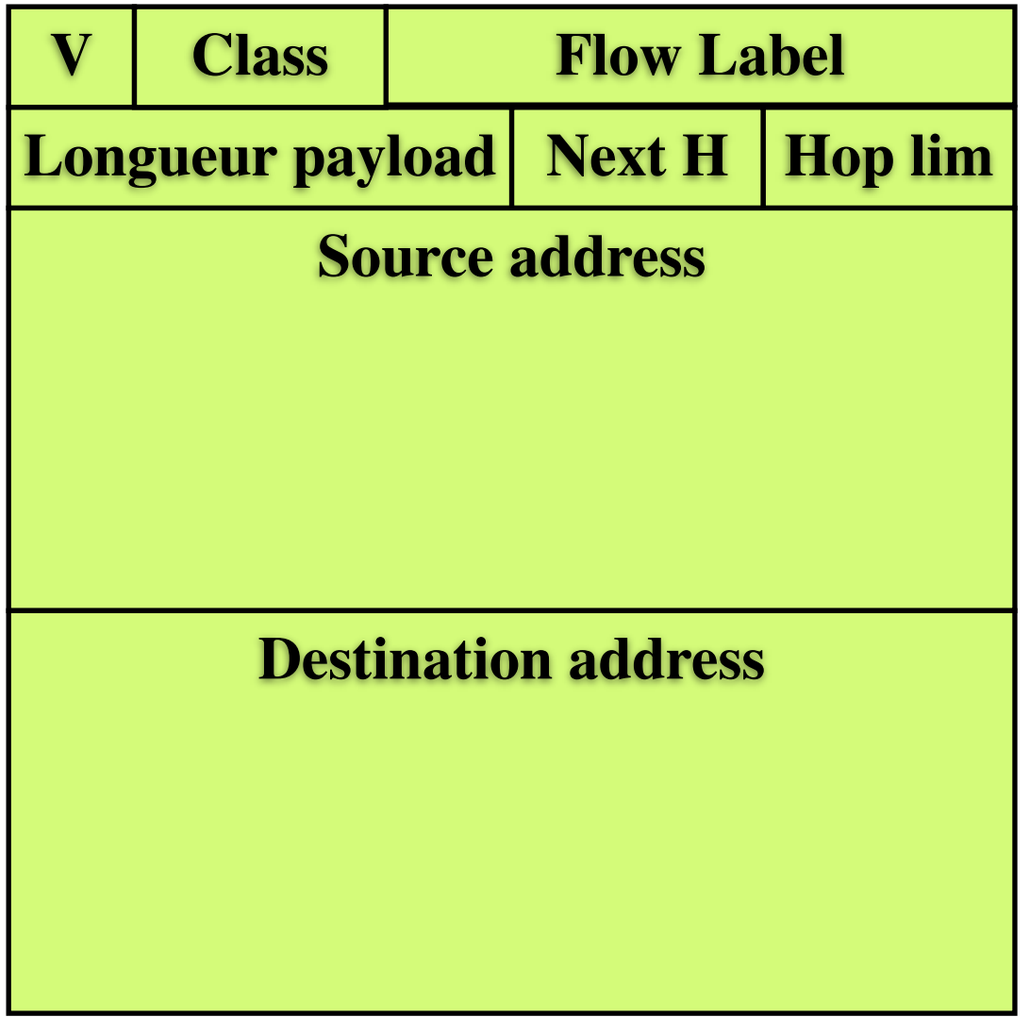
V	L	TOS	Total Length
Identification		F	Frag
TTL	Proto	Checksum	
Source address			
Destination address			
Options	Padding		

V	Class	Flow Label	
Payload length		Next H	Hop lim
Source address			
Destination address			

- 7 fields instead of 13
  - Easier routing
  - Fixed size
- Removed fields
  - Checksum ; header length ; fragmentation ; options

IPv6: header

- Version: 6
- Traffic class (QoS)
- Flow Label (QoS)
- Payload length (replaces *packet length*)
- Next header (replaces *protocol*)
- Hop Limit (replaces *TTL*)
- Source address
- Destination address



Network layer additional mechanisms

Claude Chaudet

IP addresses automatic allocation: DHCP

TELECOM ParisTech

DHCP Introduction

- DHCP = Dynamic Host Configuration Protocol
  - Protocol destined to allocate IP addresses to terminals on a network
  - Evolution of BootP

- Main goal: send to connecting hosts the whole set of network parameters
  - Allocate an IP address and a network mask
  - Propagate the addresses of useful servers (SMTP, DNS, ...)

Protocol principles

- Simple behavior :
  - When requesting an address, the client broadcasts a DHCPDISCOVER request
  - The server replies with a broadcast DHCPOFFER packet
  - The client sends back a DHCPREQUEST unicast message to validate the IP address choice
  - The server finally answers with an unicast DHCPACK confirmation

- The address is *leased* to the client for a certain duration
  - The client may prolongate this duration by issuing another DHCPREQUEST message

Detailed behavior

- What happens when the server and the client are not on the same segment?
  - Broadcasts do not cross routers
  - Use DHCP relays to catch broadcast and retransmit it towards the real DHCP server
- Why is DHCP an application layer protocol?
  - It only concerns layer-3 addresses
  - However, it relies on a client-server architecture and may need other services provided by upper layers (reliable transmission, security, ...)
  - In the absence of a pure layer-3 solution, it has been deployed as an application layer protocol.

IP signaling: ICMP

Role of ICMP

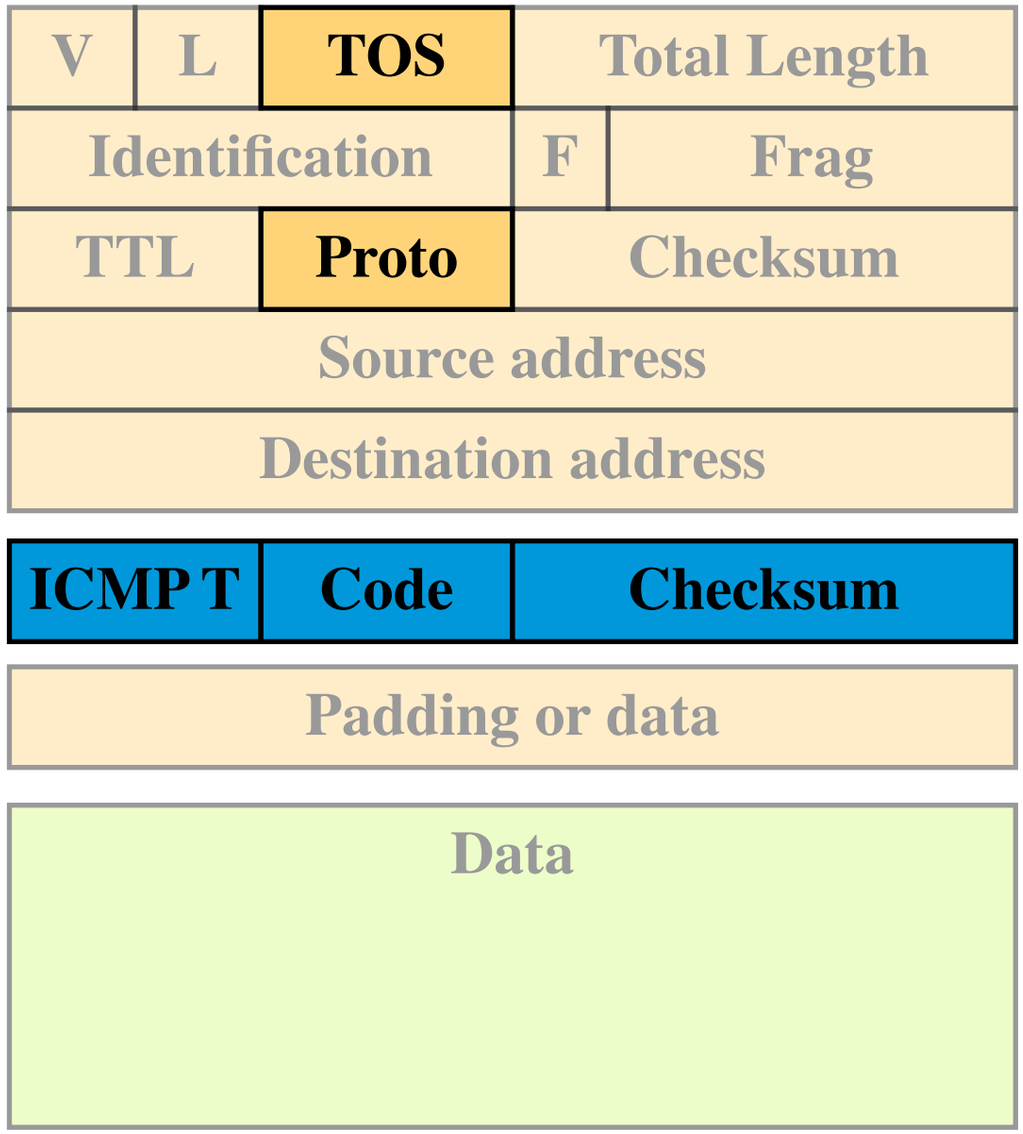
- IP is a simple mechanism, dedicated to end-to-end packets transmission

- It is not really a protocol, as there are no messages defined by IP itself
- However, it relies on a series of side protocols:
  - e.g.: Routing tables management: BGP, OSPF, IS-IS

- ICMP (*Internet Control Messages Protocol*) provides the missing signaling:
  - Errors handling (when routing fails)
  - Exchange of information between hosts and interconnection devices

ICMP - Layer 3 signaling

- ICMP is a network layer protocol, implemented as an IP packet
  - Protocol type = 1; TOS = 0
- It is encapsulated in an IP packet and defines three fields:
  - ICMP type: type of message
  - Code: error code
  - Checksum: on ICMP part only
  - Padding/data: potential ICMP data



Example — echo request (Ping)

- Message sent to a host or to a router
  - The destination should answer by the same message, indicating that it is alive and connected.
- Request: type = 8; code = 0; data = sequence number
- Answer: type = 0; code = 0; data = sequence number

- Used by *ping*
- Often filtered by routers for security (obfuscation)

Example - destination unreachable

- This message is sent by a router or by an end host to inform the source that the destination application cannot be reached

- Message type = 3
- Code gives precisions on the error location:
  - 0: *network unreachable*
  - 1: *host does not exist on network*
  - 4: *unreachable port (no application listening)*
  - *etc.*

- Routers are not forced to send these messages.
  - Security / reliability justification: avoid overloading the router

Example - Time expired

- The TTL field in the IP header is decremented by 1 each time a router is crossed.
  - When the value reaches 0, the packet is dropped and the source is notified with an ICMP message
- Type = 11

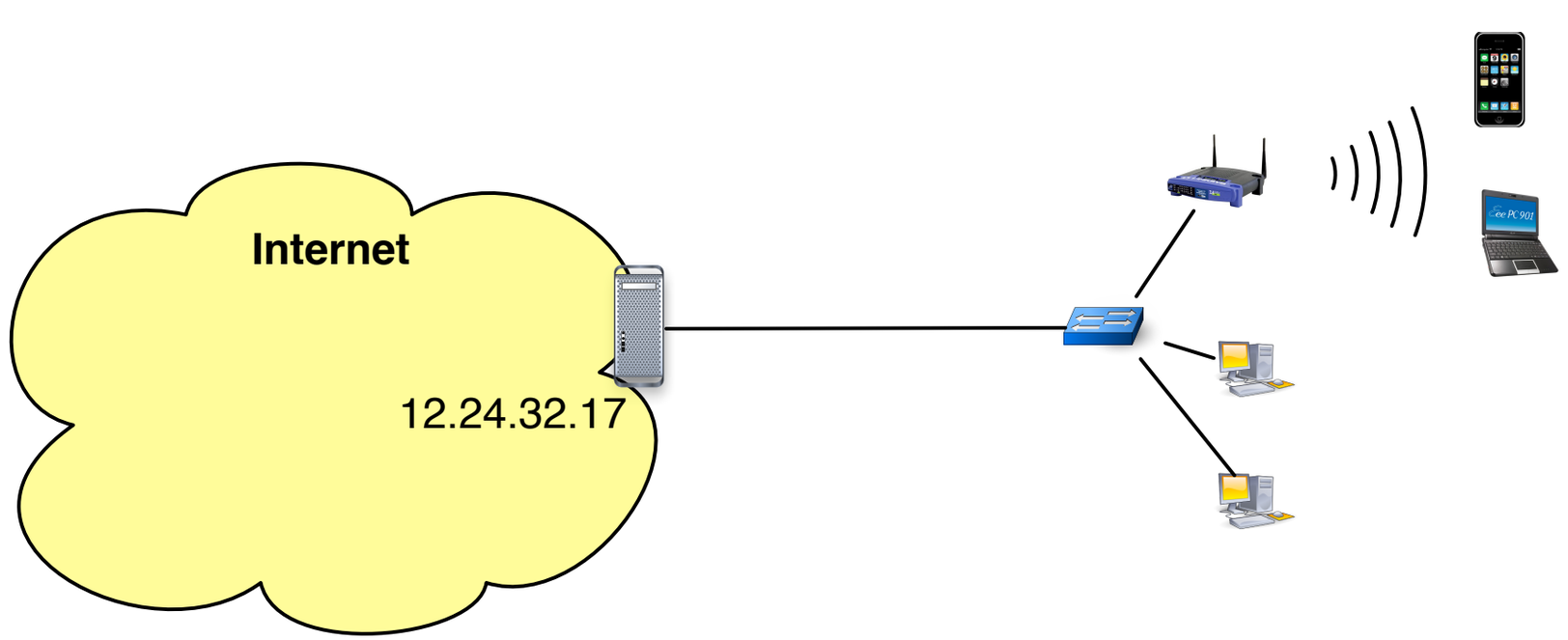
- Application (cf. Lab): `traceroute`
  - Traceroute uses a series of ping requests to the destination
  - First packet is sent with a TTL=1
  - The first router drops the packet and informs the source, revealing its address
  - Second packet is sent with a TTL=2
  - The second router drops the packet, informs the source
  - etc.

Network Address Translation (NAT)

TELECOM ParisTech

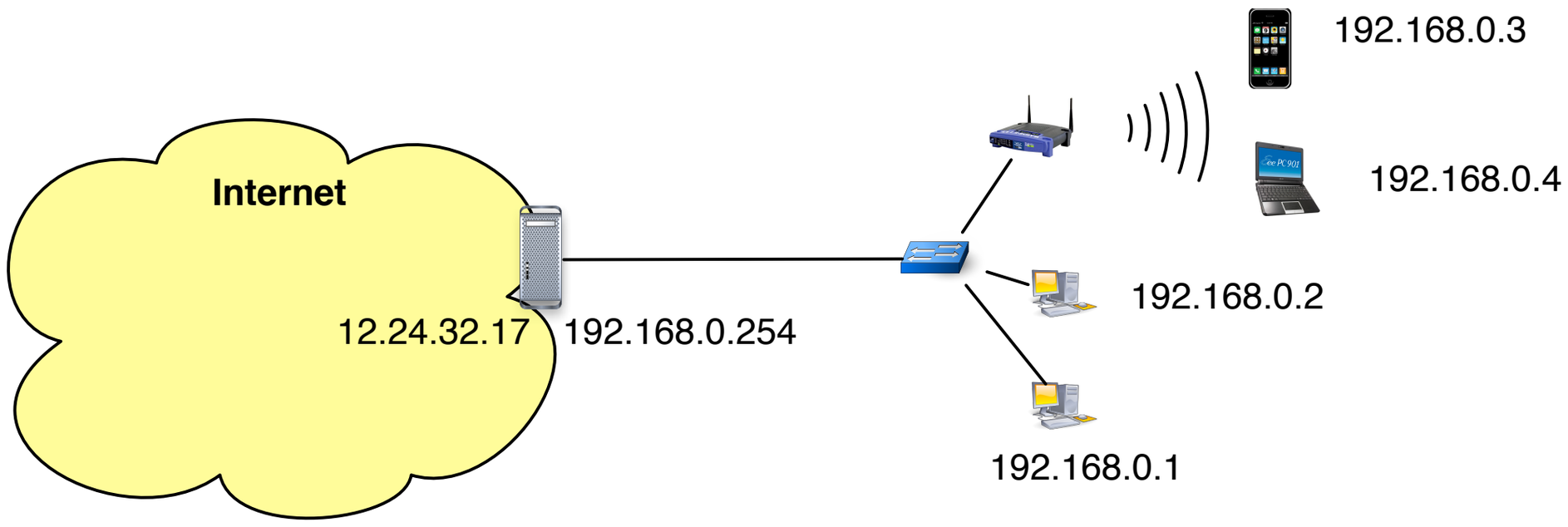
NAT principle

- **NAT was first an answer to the scarcity of the IPv4 addressing space**
  - How to hide a whole network behind one IP address ?
  - Sometimes 2, 3, ..., N addresses can be shared



NAT: general principles

- **In the LAN, use a private addressing space**
  - 192.168.0.0/16; 10.0.0.0/8 or 172.16.0.0/12
  - Non-routable (a router will discard a packet destined to such an address)
- **Use a gateway that modifies the IP header**
  - Contrary to the expected behavior of IP (keep the same address end-to-end)



The gateway's duty

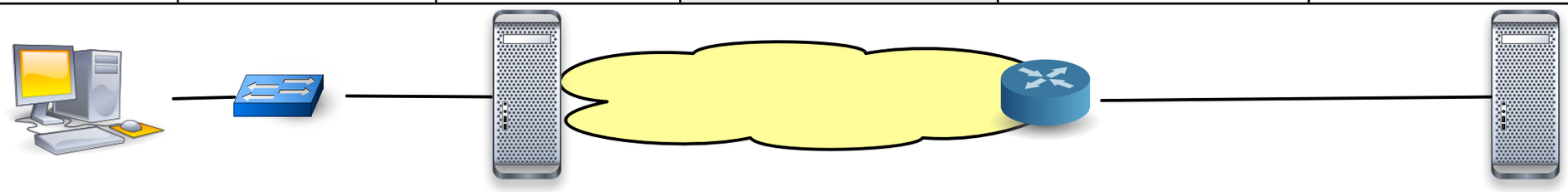
- **The gateway keeps track of which connection involves which computer**
  - A port number (transport layer) ↔ one device
  - One public IP address ↔ one device
  - Combination of both

- **The gateway looks, for each packet, the header fields to find how to modify the packet**
  - Change source/destination TCP/UDP port
  - Change source/destination IP address

- **Modification of every packet => time consuming and reserved for reasonable size networks**

Example: a request and an answer

Browser Port 49737 **eth0** 192.168.0.1	**eth0** **eth1**	**eth0 (private)** 192.168.0.254 **eth1 (public)** 12.24.32.17	**Autonomous Systems Succession**	**eth0** 72.14.238.45 **eth1** 173.194.78.254	**Web server** Port 80 **eth0** 173.194.78.94



Port numbers	
Source: 49737 **Dest.: 80**	**Source: 63822** **Dest.: 80**
IP addresses	
Source: 192.168.0.1 **Dest.: 173.194.78.94**	**Source: 12.24.32.17** **Dest.: 173.194.78.94**

Port numbers	
Source: 80 **Dest.: 49737**	**Source: 80** **Dest.: 63822**
IP addresses	
Source: 173.194.78.94 **Dest.: 192.168.0.1**	**Source: 173.194.78.94** **Dest.: 12.24.32.17**

Different NAT types

- **Using a set of ports or IP addresses**
  - In the classical scenario (*masquerading*) the gateway uses a set of ports to distinguish connections
- **Source (SNAT) vs. destination (DNAT)**
  - outgoing: SNAT; incoming: DNAT
- **Static NAT vs. Dynamic NAT**
  - Static: permanent association between external and internal parameters
    - Often used for DNAT (server behind a NAT gateway)
    - ex: web server (port 80) that answers on 137.194.2.34 is indeed on 192.168.0.4
  - Dynamic: the gateway decides of the parameters in function of the traffic
    - Outgoing connections: classical situation
    - Incoming connections: load balancing between servers (for example)

NAT: summary and introduced problems

- **NAT hides behind *k* IP addresses a network of *n* > *k* machines**
  - Requires a gateway that performs address translation
  - Often uses port numbers => **breaks layer independence**
  - Change IP address => **non-compliant to the core IP principles**

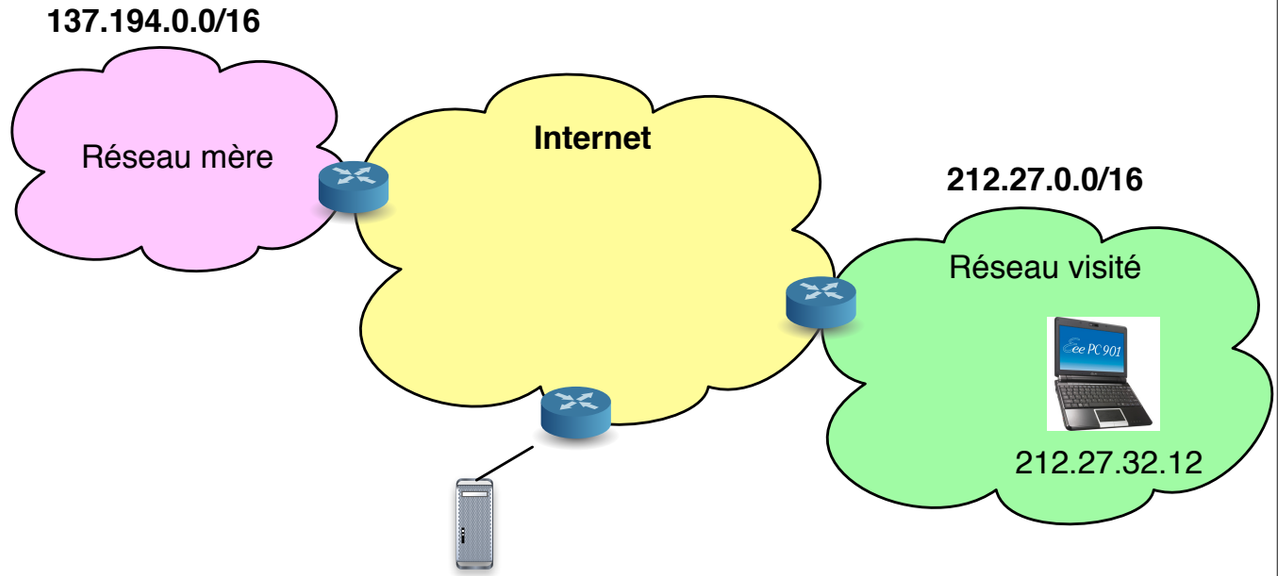
- **Introduced problems**
  - The gateway often becomes a bottleneck
  - Explicit configuration often necessary for incoming connections
  - Some applications negotiate port numbers dynamically (FTP, p2p) => specific mechanisms on the platform to look in the packets payloads

Virtual Private Networks (VPN)

TELECOM ParisTech

The problem

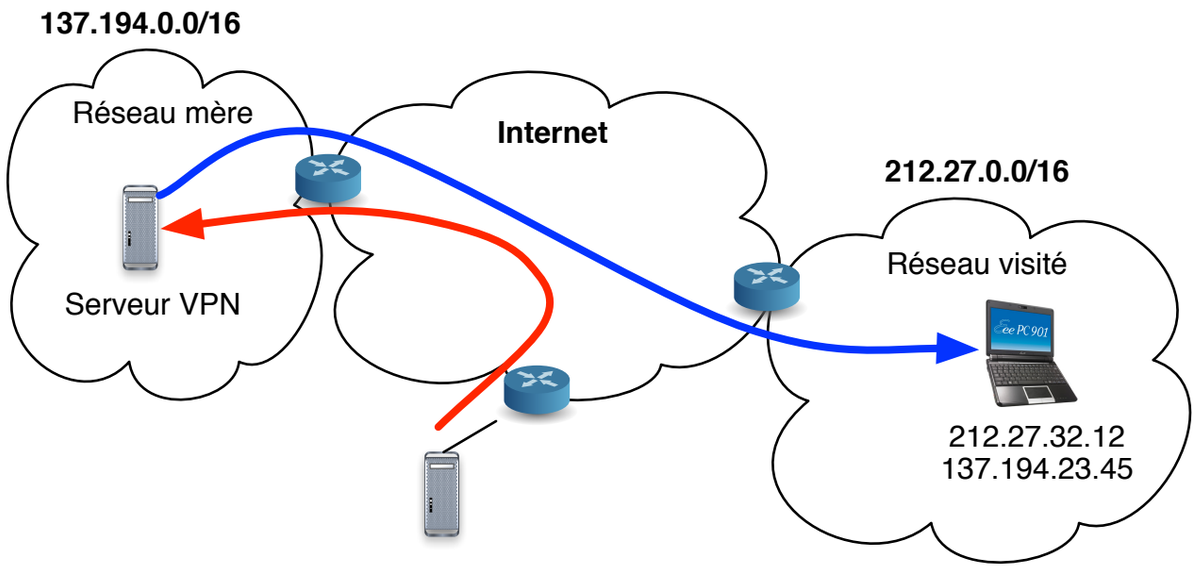
- **A user is connected through an ISP**
  - At home (telework), from a client's network, in travel, ...
- **He wants to access his company's resources as if he were inside the company's network.**
- **i.e. have an IP address that belongs to the company's network**
  - Access to internal servers (*firewalls*)
  - Access to external resources that identify clients based on their IP addresses



- **In this situation, IP forwarding will direct all packets to the home network, never to the visited network.**

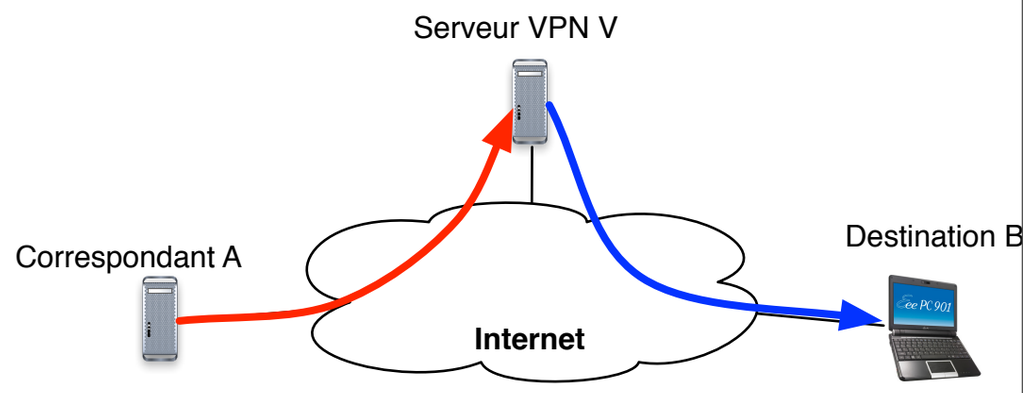
The solution: virtual private networks (VPN)

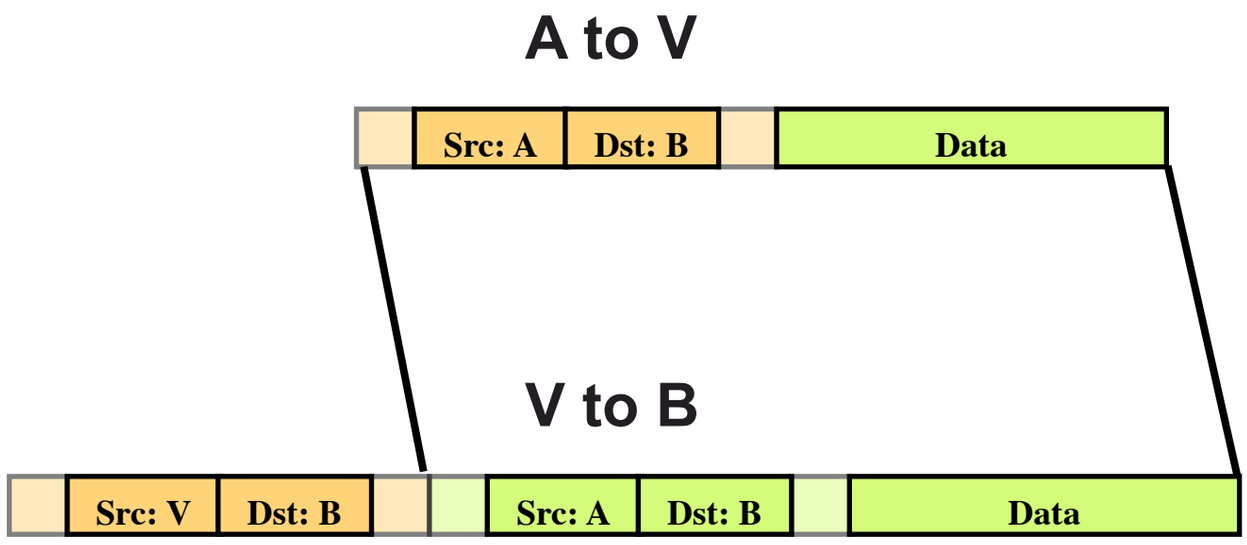
- **Use a gateway inside the home network**
  - Authenticates devices
  - Receives all packets destined to remote devices
  - Retransmit packets to the real, visited, IP
    - Terminal uses, from its point of view, its home address



Base mechanism: *tunneling*

- Peers send packets to the home address
- Packet gets routed to the VPN server
- VPN server encapsulates the IP packet in another IP packet, destined to the visited address
- This packet is sent, through the Internet, to the real device
  - May get fragmented (MTU)
- The VPN software "decapsulates" the packet before transmitting it to the transport layer
- Reverse direction similar





Various VPN types

- **T*unneling* can be applied at various levels**
  - IP in IP
    - IPSec (with ciphering)
    - Mobile IP
  - IP in HTTP
  - Layer 2 protocol (e.g. PPP) in IP
    - PPTP, L2TP
  - IP in SSL/TLS
    - SSL/TLS = secure session-level protocol
  - IP in SSH
    - SSH: secure telnet

Conclusions on the Application and Network Layers

What to know

- **General concepts**
  - Circuit switching vs. packet switching
  - OSI model
  - Encapsulation
  - Standards (where to find detailed information on a protocol, technology, ...)

- **Networks architectures**
  - A few facts about telephone networks (hierarchical architecture, databases,...)
  - Orders of magnitude (networks sizes)
    - 1 billion devices ; 50000 AS
  - Internet : Access vs. Core
  - Autonomous Systems (+ existence of online databases)
  - Peering vs. transit

What to know (2)

- **Application layer**
  - Notion of applicative protocol
  - Applicative metrics (throughput, delay, jitter, loss rate, ...)
  - Application classes (real-time/conversational, streaming, elastic,...)
  - Classical applications: DNS, DHCP, ...
- **Transport Layer**
  - Roles: port numbers, sessions, fault tolerance, congestion control
  - TCP behavior and base mechanisms (slow start, congestion avoidance, ...)
- **Network Layer**
  - Addressing - formats and role (locating users, routing)
  - IP addressing principles (prefix vs IID, IPv6, reserved slices,...)
  - IP forwarding principles (and VPN, NAT, ...)
  - Routing algorithms and protocols names and roles (eBGP, iBGP, OSPF, IS-IS)
  - Unicast, broadcast, multicast
  - ICMP and applications (traceroute, ...)

What to know

- **Practical skills**
  - Networking toolbox (commands, frame analysis, ...)
  - Be able to make an IP addressing map
  - Configure routing
- **Be able to characterize an application**
  - Traffic characteristics
  - Spy on a protocol
- **Be able to diagnose a connection**
  - What do I need to communicate: IP address, gateway, DNS server, ...
  - Diagnose routing problems (ping, traceroute, ...)