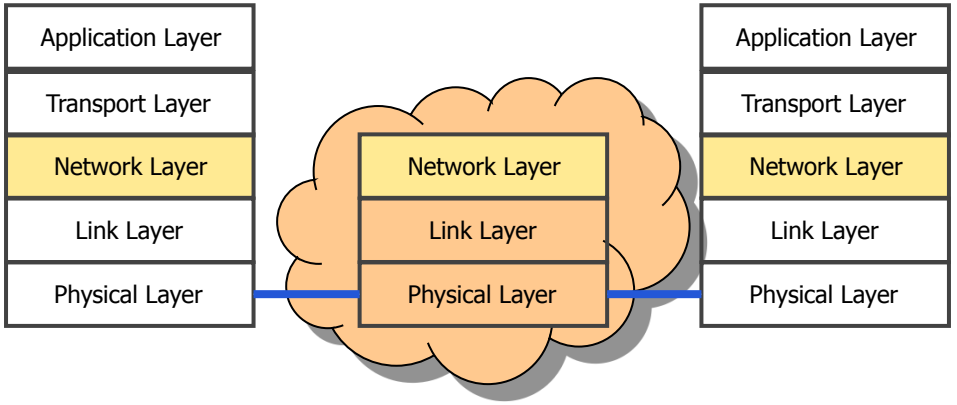
Freie Universität Berlin

Telematics
Chapter 7

The Network Layer



Dr. habil. Emmanuel Baccelli
INRIA / Freie Universität Berlin

Institute of Computer Science
Computer Systems and Telematics (CST)

Interconnecting Heterogeneous Networks

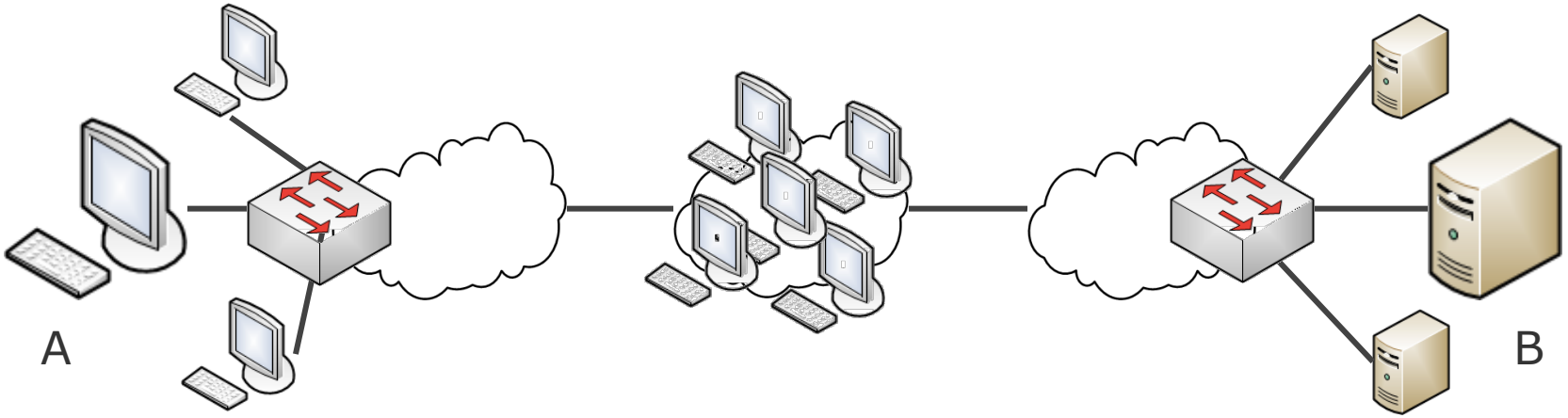
- Idea: let's interconnect all the LANs, MANs, and WANs => the Internet!

- Problem 1: interconnected LANs, MANs, WANs use different technologies
  - Link layer protocols, frame formats
  - Link layer addresses
  - Physical media
  - How to transfer data across series of heterogeneous LANs, MANs, WANs ?

- Solution: uniform internetwork address, end-to-end packet format
  - Abstracting the lower layers (link layer and below)
    - Internetwork addresses define packets' destinations across various link layer technologies
    - Common end-to-end packet format, independent of intermediate link layer technologies

Interconnecting Heterogeneous Networks

- Problem 2: how to discover & maintain paths across a big internetwork?
  - How do you implement communication between A and B?
    - Broadcast? Not scalable!
    - Selective forwarding using manually configured state in switches? Not scalable!



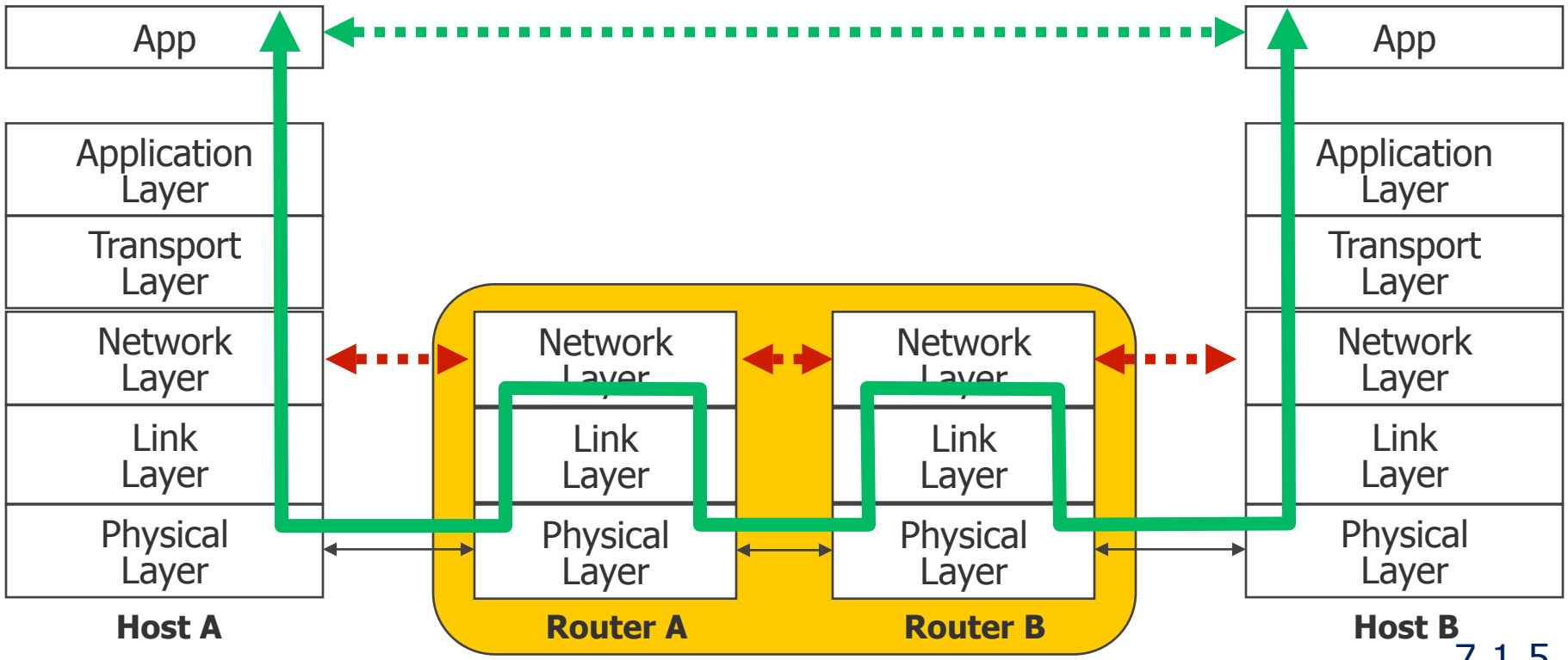
- Solution: routing & forwarding
  - **Routing**
    - Automatically acquiring & updating next hop information, for each destination
  - **Forwarding**
    - Moving incoming packets from input interface to the appropriate output interface(s)

CONTENT of this CHAPTER

- Fundamental Goals of the Network Layer
- Network Layer Addressing
  - IPv4 Addressing
  - IPv6 Addressing
- Internet Protocol Design
  - Internet Protocol version 4
  - Internet Protocol version 6
  - IPv6 Migration: Transition and Coexistence
- IP Address Assignment and Mapping
- Network Address Translation (NAT)
- Routing
- Multicast

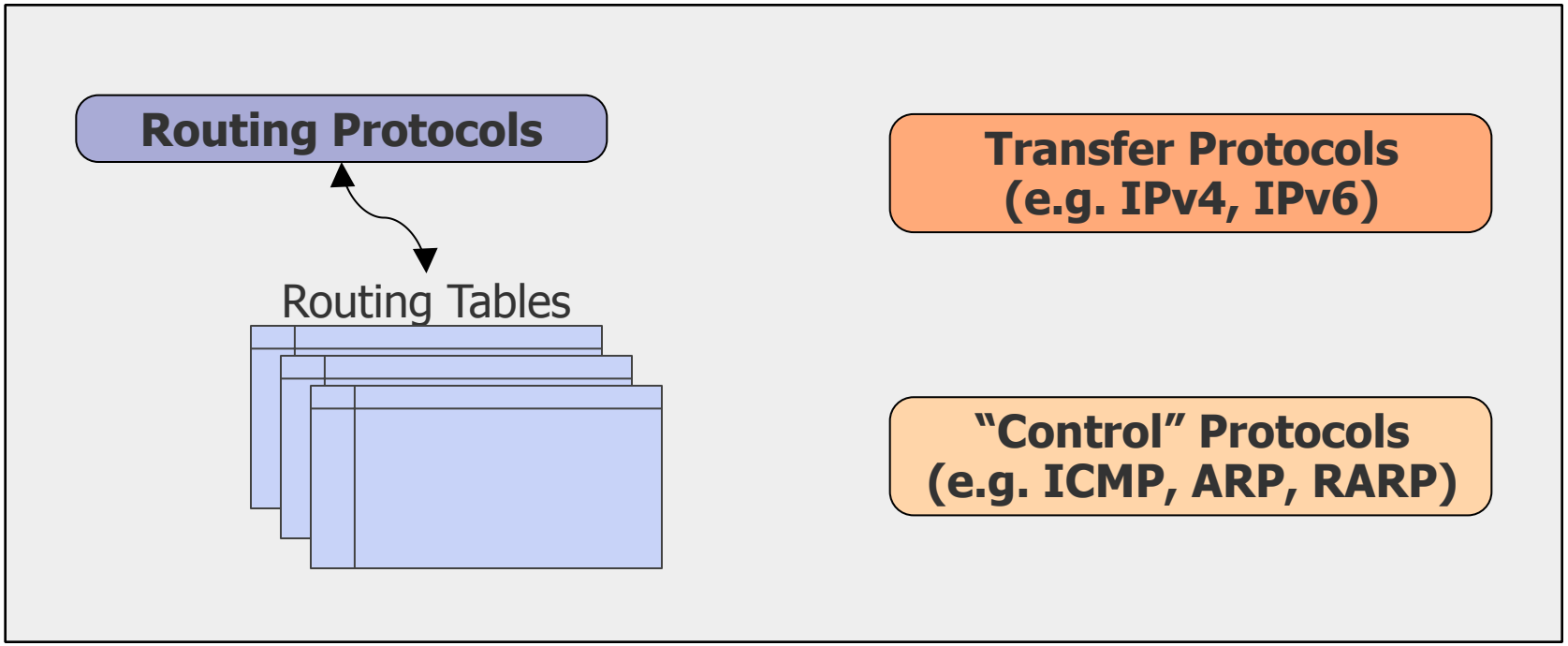
The Network Layer: Goals

- The network layer
  - Manages end-to-end connectivity accross multiple LANs, MANs and/or WANs
    - Defines internetwork identifiers
    - Defines uniform end-to-end packet format
    - Defines routing and forwarding mechanisms



The Network Layer: Protocols

- Network layer protocols accomplish 3 main tasks:
  - Data transfer over heterogeneous internetworked LANs, MANs, WANs
  - Routing decision management at the intermediate nodes
  - Control of the network (e.g. address mapping or transmission status)



The Network Layer: End-to-End Connectivity

Two fundamental philosophies:

- Connectionless end-to-end communication
  - Data is chunked and transferred as packets of variable length
  - Source and destination address are indicated in each packet
  - Sending is made spontaneously without reservations, access is always possible
  - Advantages: easy to implement, small susceptibility to faults
  - Drawbacks: varying paths for the packets
    - Wrong order of packets at the receiver, differences in transmission delay, unreliability
- Connection-oriented end-to-end communication
  - Connection establishment phase required
  - Data transmission phase: information exchange between the partners
  - Connection termination phase: release of the terminals and channels
  - Advantages: no change of sequence, reservation of capacity, flow control
  - Drawbacks: more complex to implement, higher susceptibility to faults
    - Remark: a connection can be implemented above a connectionless end-to-end service

The Network Layer: Constraints

Intermediate nodes need to acquire and maintain state to …

- Establish paths, and the scope broadcast
- Guarantee service quality

Constraint 1: **memory is limited**

- Intermediate nodes cannot store arbitrary amounts of state

⇒ Aggregation/compression of state helps

Constraint 2: **throughput is limited**

- Intermediate nodes cannot rely on arbitrary amounts of control traffic to acquire state

⇒ Frugal approaches are preferred

Constraint 3: **computing capacity is limited + speed is crucial**

- Intermediate nodes cannot compute arbitrary complex operations

⇒ Simpler and faster operations are preferred

CONTENT of this CHAPTER

- Fundamental Goals of the Network Layer
- Network Layer Addressing
  - IPv4 Addressing
  - IPv6 Addressing
- Internet Protocol Design
  - Internet Protocol version 4
  - Internet Protocol version 6
  - IPv6 Migration: Transition and Coexistence
- IP Address Assignment and Mapping
- Network Address Translation (NAT)
- Routing
- Multicast

Network Layer Addresses

Basics

- Addresses are used to identify end hosts in multi-access networks
- In case of multi-interface hosts, addresses help to select the right interface(s)

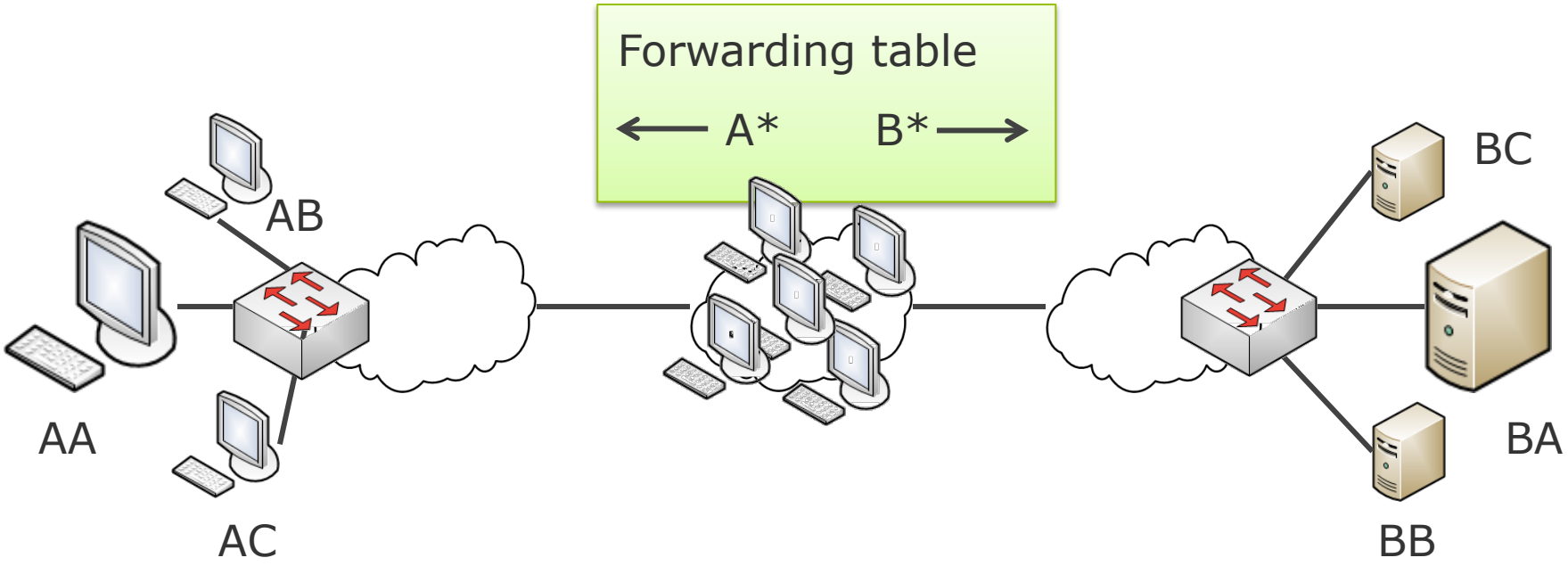
Requirements

- Compactness of representation
- Independence from hardware and link layer (logical addressing)
- Built-in support for efficient, decentralized path finding
- Unicity within a given network scope

Examples of network layer addresses

- IPv4 address: 8.8.8.8 (coded on 4 bytes)
- IPv6 address: 2001:0DB8:0:CD30:123:4567:89AB:CDEF (coded on 16 bytes)
- IPX address: FEDCBA98 1A2B3C5D7E9F 0453 (coded on 12 bytes)
- AppleTalk address: 10.1.50 (coded on 4 bytes)

Network Layer Addresses: Aggregation



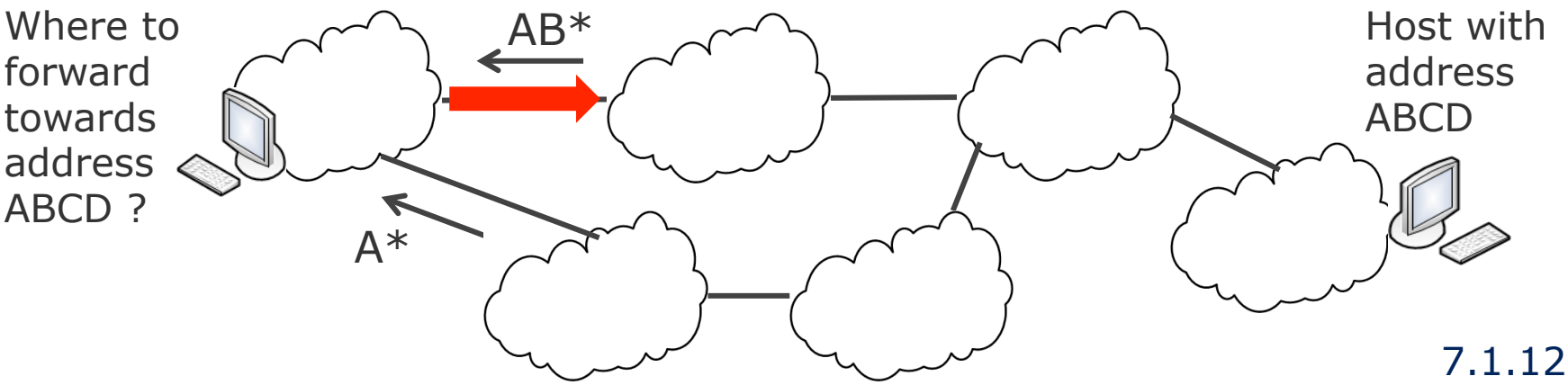
Main advantage of aggregation: **much smaller state in intermediate nodes!**

Examples of rationales allowing aggregation:

- Geographical addressing
  - Aggregate based on geographical area
  - Strict assignment policy
- Hierarchical addressing
  - Aggregate addresses by organizations, sub-organizations, etc.
  - Higher flexibility in assignment might also lead to less compressed aggregation

Network Layer Addresses: Hierarchical Addressing

- There are several aspects to hierarchical addressing:
  - **Allocation of an address block**
    - Organization requests an initial range of continuous addresses
    - Address block determined by a **prefix** (e.g. bits all addresses in the block have in common)
    - Allows recursive (sub)delegation of "authority" over parts of the address space
  - **Assignment of an address**
    - Give an address to host/interface
  - **Longest prefix match**: determining the best path based on prefix length
    - Principle: forward data to output interface that shares most digits with destination



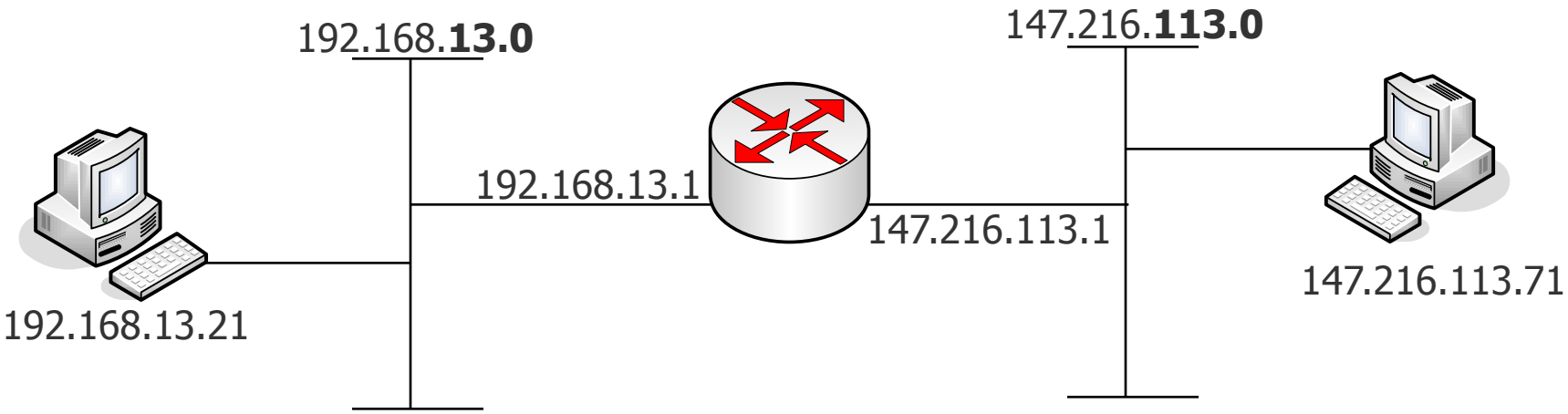
Network Layer Addresses: Challenges

- **Security**
  - Example: Address spoofing
    - Attacker maliciously claims to own a network address (or parts of an address range) that belongs to another node
    - No global database that gives the information on a fine-grained base
- **Efficiency**
  - Effective aggregation requires careful address planning
    - Keep larger address blocks
    - Give addresses back if you don't need them
- **Automatisation**
  - Example: autoconfiguration of address assignment
    - We will look at some solutions later in this chapter

CONTENT of this CHAPTER

- Fundamental Goals of the Network Layer
- Network Layer Addressing
  - IPv4 Addressing
  - IPv6 Addressing
- Internet Protocol Design
  - Internet Protocol version 4
  - Internet Protocol version 6
  - IPv6 Migration: Transition and Coexistence
- IP Address Assignment and Mapping
- Network Address Translation (NAT)
- Routing
- Multicast

IPv4 Addressing

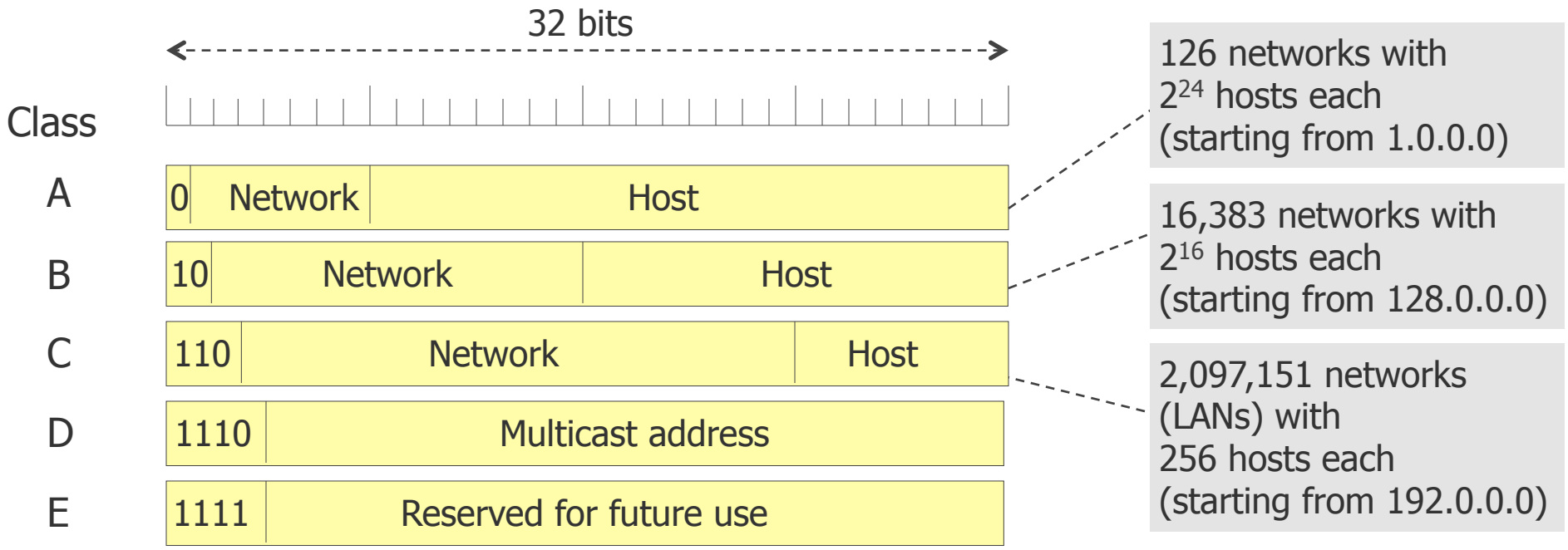


- IPv4 addresses are 32 bits long, represented in 4 sections of 8 bits each:

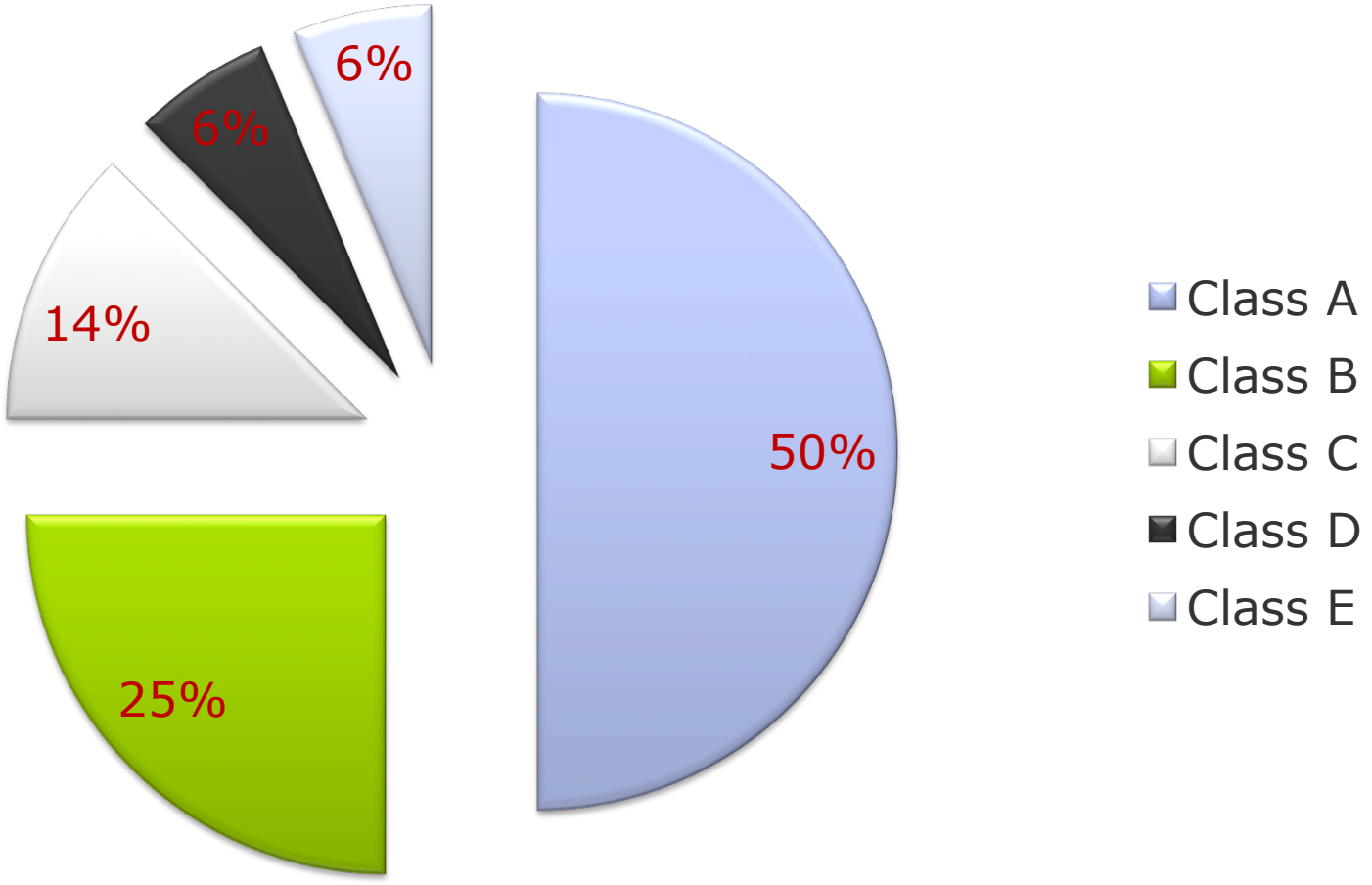
Binary format	11000000 . 10101000 . 00001101 . 00010101
Dotted Decimal Notation	192.168.13.21

- Each node has (at least) one world-wide unique IP address
- Router or gateways that link several networks, have for each network an IP address
- IPv4 addresses are composed of **network identifier part** and **host identifier part**

Network/Host Identifiers with IPv4 Address Classes



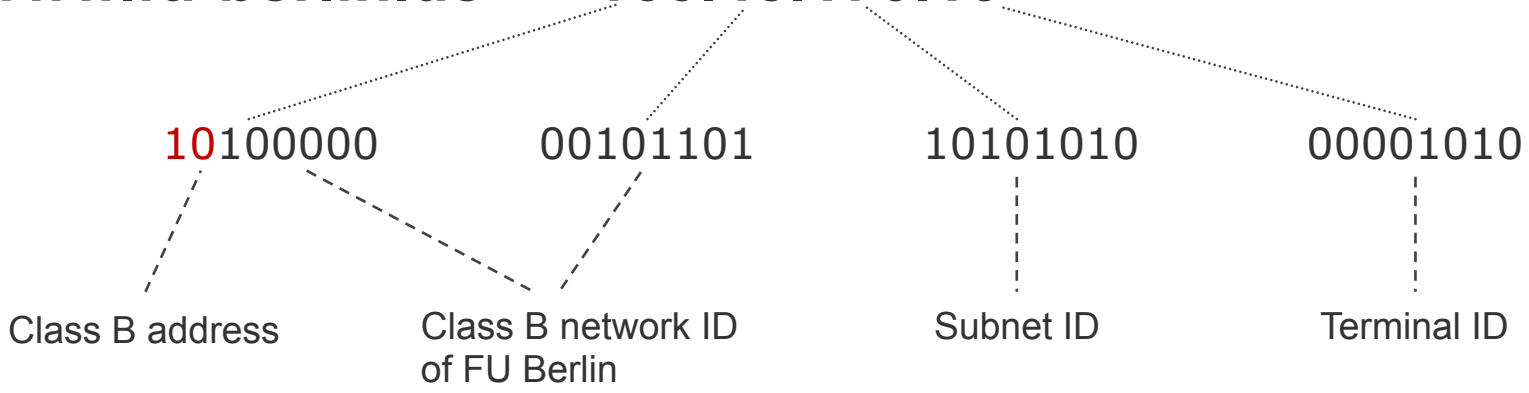
IPv4 Address Space per Address Class



IPv4 Addressing: Address Examples

Example of IPv4 address assigned to a host:

www.fu-berlin.de = 160.45.170.10

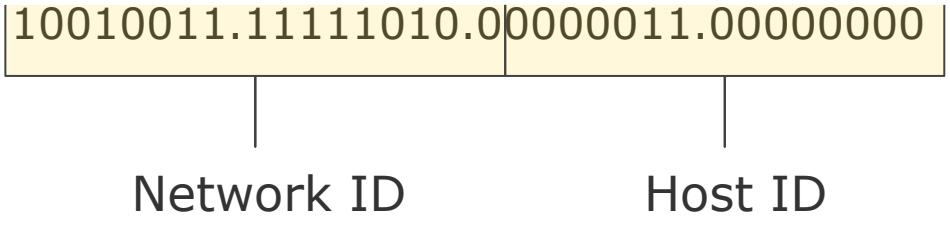


Examples of *well-known* IPv4 addresses:

0 0 0 ..... 0 0 0	This host
0 0 ..... 0 0 | Host	Host in this network
1 1 1 ..... 1 1 1	Broadcast in own local area network
Network | 1 1 1 ..... 1 1 1	Broadcast in remote network
127 | arbitrary (usually 0.0.1)	Local loop, no sending to the network

IPv4 Addresses: Variable-Length Subnet Masking

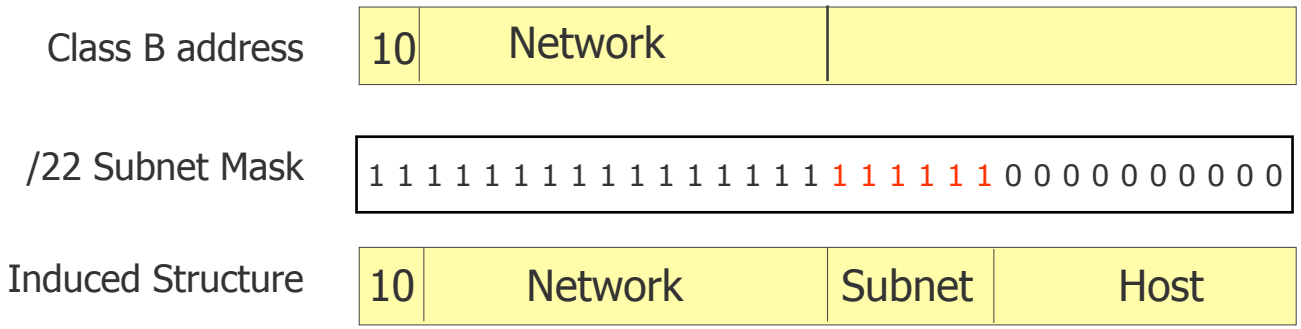
- Problem: Inefficient use of the address space with static classes of addresses
  - Class C-networks (256 hosts) are very small and Class B-networks (65536 hosts) are often too large
    - e.g. if 500 devices in an enterprise are to be attached, a Class B address block is needed, but by this unnecessarily more than 65,000 host addresses are blocked!
  - Too many Class A address blocks were assigned in the first Internet years
    - Nobody expected the explosive growth of the Internet
- Solution: variable-length subnet masking (VLSM)
  - Replacing static classes (networkID= 7, 14 or 21 bits) by network prefixes of any length
  - An IP address is then of the shape: **a.b.c.d/n**
    - The **first n bits** are the **network identification**
    - The remaining (32 – n) bits are the **host identification**
  - Example: 147.250.3/17
    - The first 17 bits are network ID and the remaining 15 bits are host ID



  - VLSM is the base of CIDR (Classless Inter-Domain Routing)

IPv4 Addresses: Subnets & Subnet Masks

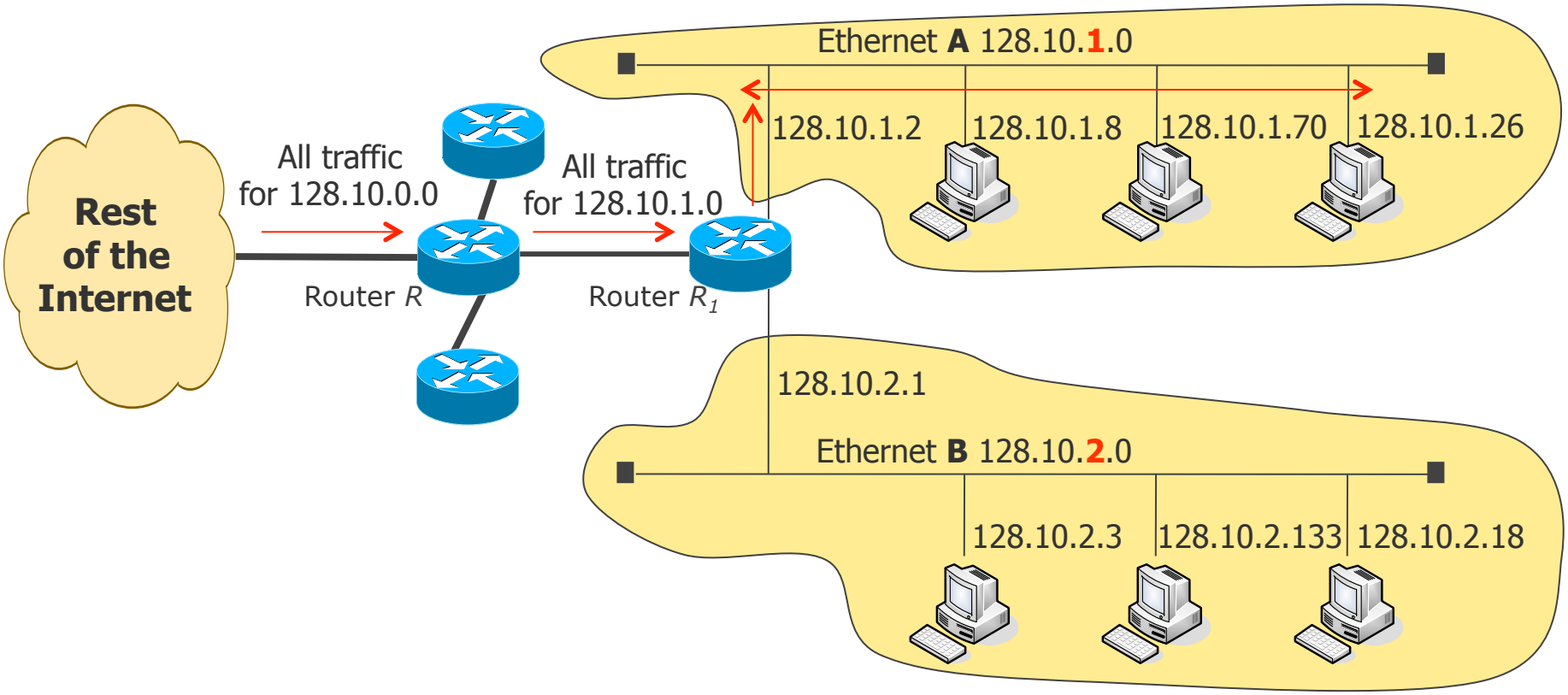
- A **subnet** is a subset of addresses in a class A, B, or C network
  - Principle: some bits of the host address part are used to "complement" network ID
- A **subnet mask** defines the "abused" bits (taken from the host part)



- All hosts on the same physical network use the same subnet mask
  - Combining IP address and subnet mask, a router determines to which subnet a packet must be sent

IPv4 Addresses: Subnets & Subnet Masks

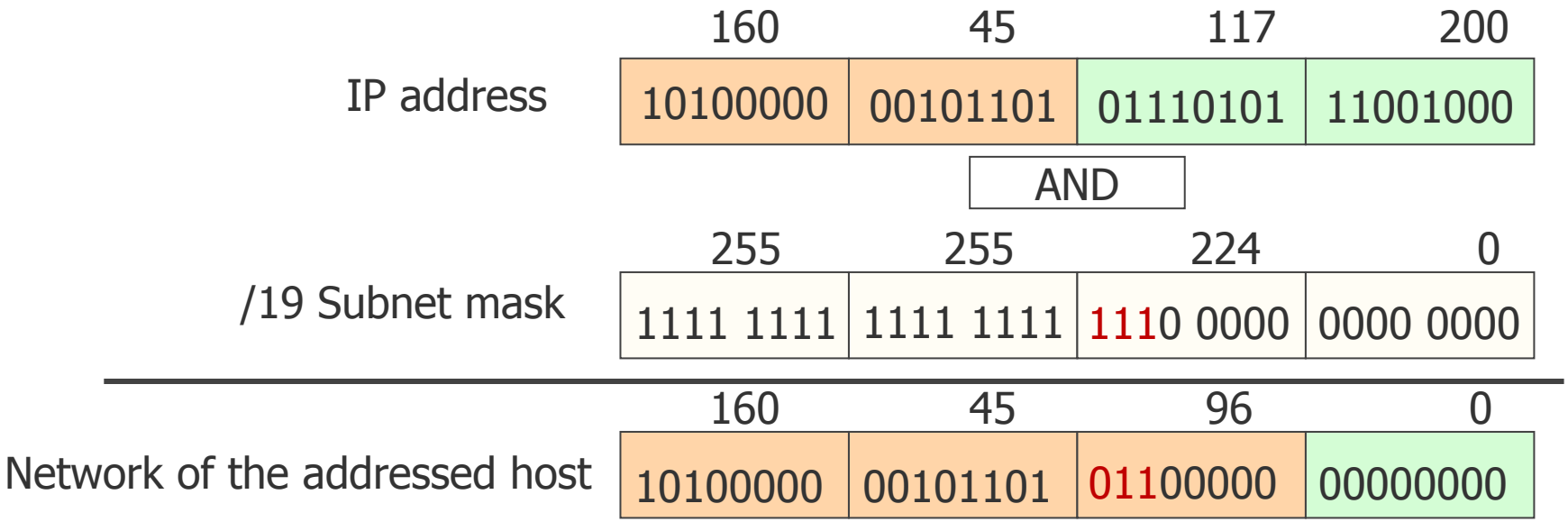
- Division of the a network into subnets, with subnet mask 255.255.255.0



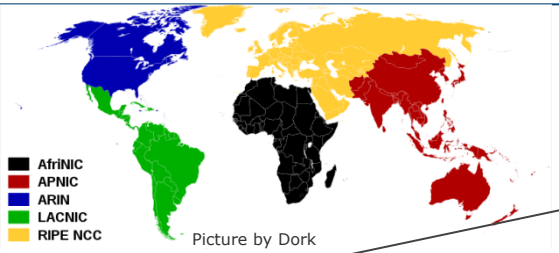
- Example: router *R* receives a packet from the Internet for host 128.10.1.26
  - *R* performs a logical AND operation on destination address and subnet mask
  - The result of the AND indicates the subnet to which R_1 connects => next hop = R_1

IPv4 Addresses: Subnets & Subnet Masks

- Example of of subnet masking computation:
- The entrance router of the FU Berlin receives an IP packet for 160.45.117.200
  - The entrance router does not know where host "117.200" is located
  - But the entrance router knows that /19 subnet masks are used
- The entrance router performs bit-wise AND between IP address and subnet mask
  - Computes the result as being 160.45.96.0
  - Sends the packet to the router that connects to subnet 160.45.96.0



IP Address Block Allocation: Actors & Process

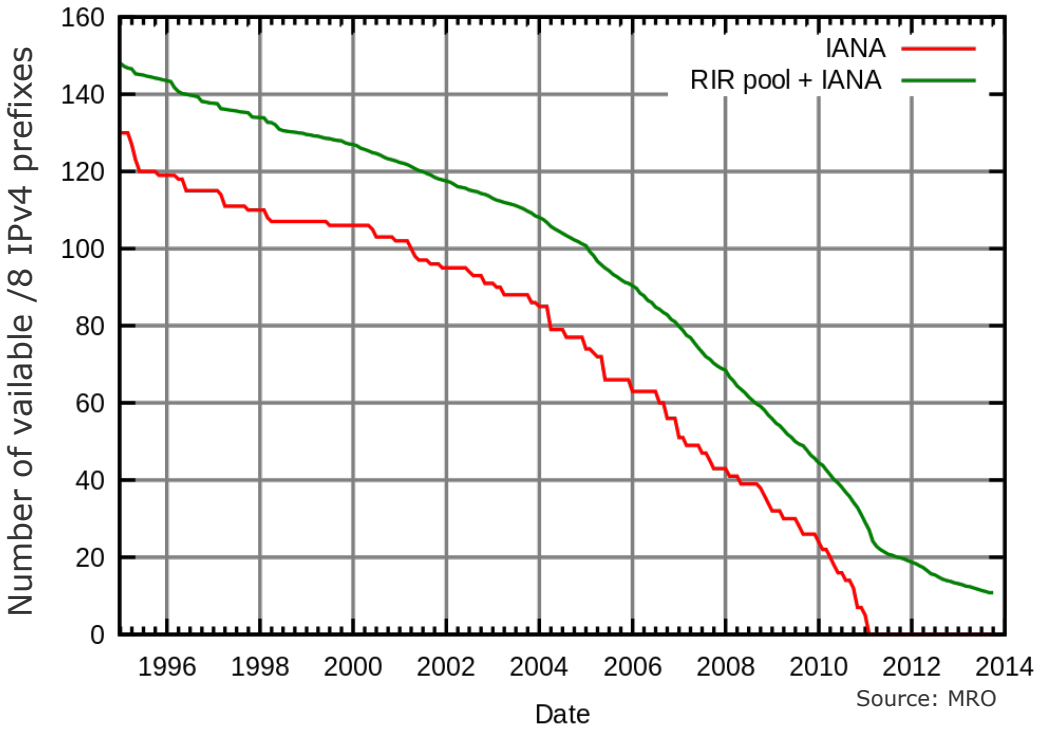


- IANA – Internet Assigned Numbers Authority
- RIR – Regional Internet Registry
- LIR – Local Internet Registry
  - LIRs are mostly Internet service providers, enterprises, or academic institutions, e.g. DTAG = Deutsche Telekom, which assigns addresses to customers

IPv4 Addresses Shortage

- **Problem**: IANA has given the last available blocks of IPv4 addresses to RIRs in 2011
  - Nobody had thought about the explosive growth of the Internet
  - (Otherwise one would have defined longer addresses from the beginning)

- **Solution**: Extension of the address space
  - IPv6 has 128 bits for addresses
    ➡ 2^{128} addresses



CONTENT of this CHAPTER

- Fundamental Goals of the Network Layer
- Network Layer Addressing
  - IPv4 Addressing
  - IPv6 Addressing
- Internet Protocol Design
  - Internet Protocol version 4
  - Internet Protocol version 6
  - IPv6 Migration: Transition and Coexistence
- IP Address Assignment and Mapping
- Network Address Translation (NAT)
- Routing
- Multicast

IPv6 Addresses

- RFC 4291 specifies IPv6 addresses
  - 128 bit = 16 bytes
  - 2^{128} addresses ~ 3.4×10^{38}
  - more than 7000 IPv6 addresses/m^2 on all earth's surface (including oceans!)
  - but still not unlimited

- Three types of IPv6 addresses are defined:
  - **Unicast**: An identifier for a single interface.
  - **Anycast**: An identifier for a set of interfaces. A packet sent to an anycast address is delivered to the "nearest" one.
  - **Multicast**: An identifier for a set of interfaces. A packet sent to a multicast address is delivered to all interfaces identified by that address.

- Remark: there are no broadcast addresses in IPv6!
  - Deprecated. Broadcast addresses are superseded by multicast addresses

IPv6 Addresses: Representation of addresses

- The most common notation is:

  x:x:x:x:x:x:x:x

  - the 'x's are the values of 8 pieces of the address, each piece being 16-bit long
    - Each x is noted as four hexadecimal digits with colons as separators
    - Each digit noted in ranges 0-9 or A-F (similar to MAC addresses notation)
    - Each digit represents 4 bits

- Examples of IPv6 addresses:

  8000:0000:0000:0000:0123:4567:89AB:CDEF

  FEDC:BA98:7654:3210:FEDC:BA98:7654:3210

  1080:0:0:0:8:800:200C:417A

IPv6 Addresses: Representation of addresses

- Simplified notation conventions
  - Leading zeros within a group can be omitted: 0123 ➡ 123
  - Groups of 16 zero bits '0000' can be replaced by a pair of colons '::'
    8000:0000:0000:0000:0123:4567:89AB:CDEF ➡ 8000::123:4567:89AB:CDEF
  - The "::" can only appear once in an address

- Example of addresses and their simplified notation:

2001:DB8:0:0:8:800:200C:417A	a unicast address
FF01:0:0:0:0:0:0:101	a multicast address
0:0:0:0:0:0:0:1	the loopback address
0:0:0:0:0:0:0:0	the unspecified addresses

These may be represented with the simplified notation by:

2001:DB8::8:800:200C:417A	a unicast address
FF01::101	a multicast address
::1	the loopback address
::	the unspecified addresses

IPv6 Addresses: Representation of addresses

- Standard way to represent IPv4 addresses with IPv6 addresses:

  x:x:x:x:x:x:d.d.d.d

  - the 'x's are the hexadecimal values of the six high-order 16-bit pieces of the address, and the 'd's are the decimal values of the four low-order 8-bit pieces of the address (standard IPv4 representation)
  - Special prefix used to signal an IPv4-mapped IPv6 address: 0:0:0:0:0:FFFF

- Example:
  - 0:0:0:0:0:FFFF:129.144.52.38

  - or in compressed form:
    - ::FFFF:129.144.52.38

IPv6 Addresses: Representation of addresses

- The text representation of IPv6 address prefixes is similar to the way IPv4 address prefixes are written in VLSM (Variable Length Subnet Masking):

ipv6-address/prefix-length

  - ipv6-address: an IPv6 address
  - prefix-length: number of bits composing the prefix (leftmost contiguous)

- Examples: 60-bit prefix 20010DB80000CD3 (hexadecimal)

  2001:0DB8:0000:CD30:0000:0000:0000:0000/60

  Compressed form 2001:0DB8:0:CD30::/60

- Combination of node address and a prefix:
  - node address 2001:0DB8:0:CD30:123:4567:89AB:CDEF
  - subnet number 2001:0DB8:0:CD30::/60

  Abbreviated into: 2001:0DB8:0:CD30:123:4567:89AB:CDEF/60

IPv6 Addresses: Address Types

- The type of an IPv6 address is identified by the high-order bits:

Address type	Binary prefix	IPv6 notation
Unspecified	00...0 (128 bits)	::/128
Loopback	00...1 (128 bits)	::1/128
Multicast	11111111	FF00::/8
Link-Local unicast	1111111010	FE80::/10
Global Unicast	(everything else)	

- Unspecified: It must never be assigned to any node. It indicates the absence of an address. Used when host initializes and does not have an address.

- Loopback: It may be used by a node to send an IPv6 packet to itself. It must not be assigned to any physical interface.

IPv6 Addresses: Unicast Addresses

- IPv6 Global Unicast Addresses
  - This type of address is **routable**, i.e. it is supposed to be globally unique
  - The general format for IPv6 Global Unicast addresses is as follows:

n bits	m bits	128-n-m bits
global routing prefix	subnet ID	interface ID

  - global routing prefix: required for routing. Must be unique globally.
  - subnet ID: identifier of a link within the site.
    - Must be unique within the site
  - interface ID: identifier of an interface on that link.
    - Must be unique within the link

IPv6 Addresses: Anycast Addresses

- Anycast Addresses
  - Anycast addresses are allocated from the unicast address space
  - Anycast addresses are syntactically indistinguishable from unicast addresses
  - When a unicast address is assigned to n>1 interfaces, it becomes an anycast address
- Usage scenarios:
  - Identification of the set of routers belonging to a organization (providing a service)
  - Identification of the set of routers attached to a given subnet
  - Identification of the set of routers providing entry into a particular routing domain
- Anycast address format:

IPv6 Addresses: Multicast Addresses

- Multicast Addresses:
  - An IPv6 multicast address is an identifier for a group of interfaces
  - An interface may belong to any number of multicast groups.
- Format:

8	4	4	112 bits
11111111	flags	scope	group ID

  - 11111111: identification of multicast addresses
  - flags: set of 4 flags: 0RPT
    - T=0: permanently assigned multicast address
    - T=1: non-permanently assigned multicast address (dynamically, on-demand)
    - P: multicast address assignment based on network prefix (RFC 3306)
    - R: embedding of the rendevous point (RFC 3956)
  - scope: defines the scope of the multicast group

Scope values (4 bit)
0 reserved
1 Interface-Local scope
2 Link-Local scope
3 reserved
4 Admin-Local scope
5 Site-Local scope
6 (unassigned)
7 (unassigned)
8 Organization-Local scope
9 (unassigned)
A (unassigned)
B (unassigned)
C (unassigned)
D (unassigned)
E Global scope
F reserved

CONTENT of this CHAPTER

- Fundamental Goals of the Network Layer
- Network Layer Addressing
  - IPv4 Addressing
  - IPv6 Addressing
- Internet Protocol Design
  - Internet Protocol version 4
  - Internet Protocol version 6
  - IPv6 Migration: Transition and Coexistence
- IP Address Assignment and Mapping
- Network Address Translation (NAT)
- Routing
- Multicast

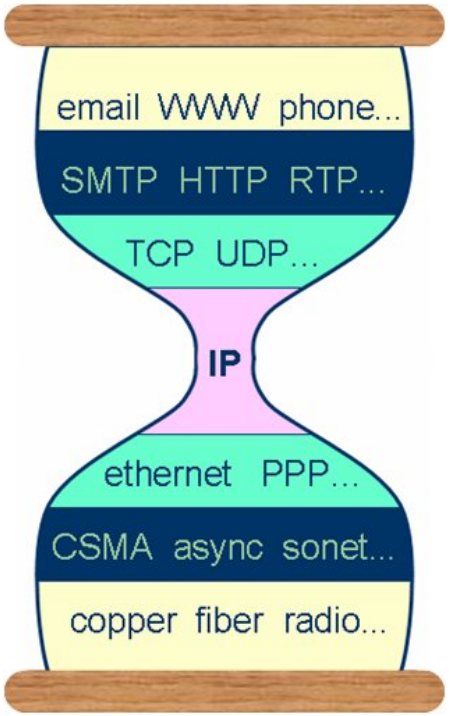
Overview of IPv4

- IPv4 (Internet Protocol version 4) is the fundamental network layer protocol of the current Internet

- Initial specification: RFC 791 (1981)

Basic properties

- Transparent end-to-end communication between hosts
- Connectionless & packet oriented
- Provides a "Best Effort" service: unreliable transmission
- Addressing: hierarchical, 32 bit addresses
- Support of packet fragmentation



Picture coined at IETF 51
by Steve Deering, 2001

Some Meta Remarks about IPv4

- Designing a protocol that serves the need for more than 30 years of technological development is really non-trivial
  - There is a continuous change of the upper and lower layer technologies

- After more than three decades, IPv4 still is the universal glue!
  - Nevertheless, IPv4 was supplemented by several additional protocols to allow for advanced services

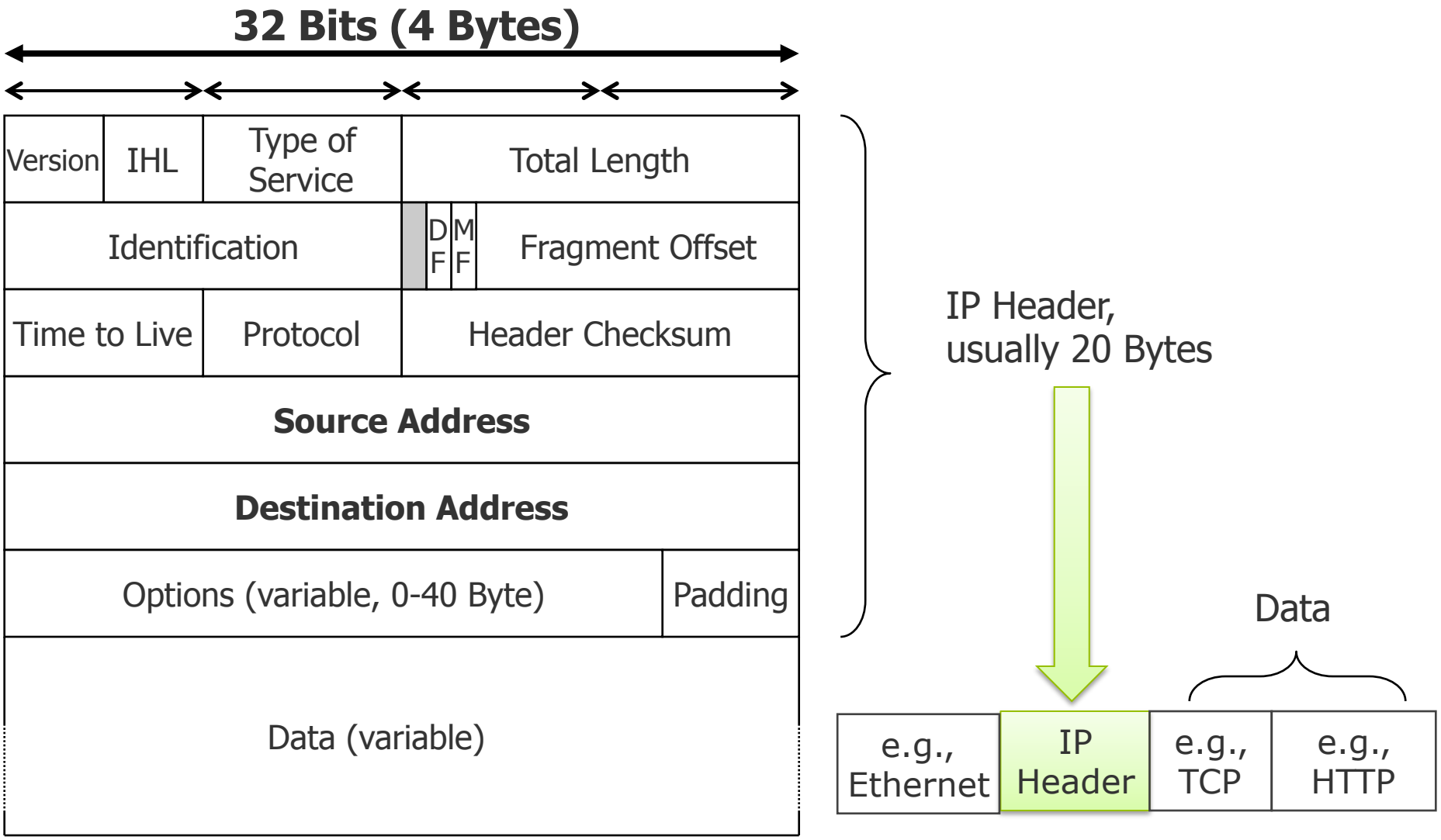
Questions:

- Why was (still is) IPv4 so successful?
- Why is it difficult to change the network layer?
- How can you implement co-existence of different network layers?

CONTENT of this CHAPTER

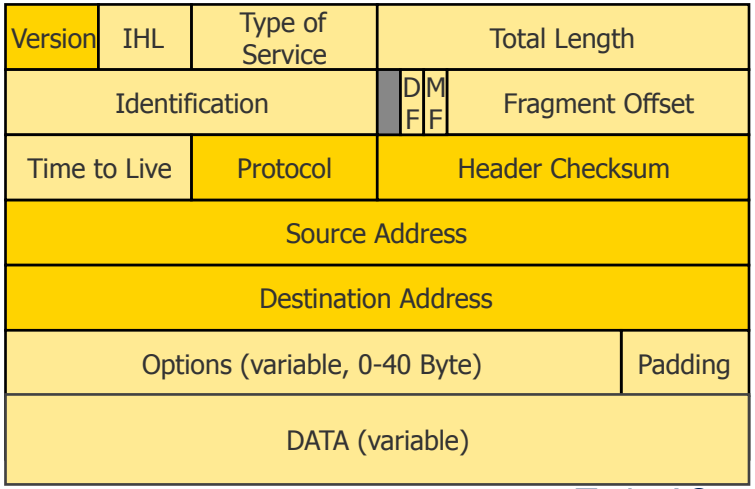
- Fundamental Goals of the Network Layer
- Network Layer Addressing
  - IPv4 Addressing
  - IPv6 Addressing
- Internet Protocol Design
  - Internet Protocol version 4
  - Internet Protocol version 6
  - IPv6 Migration: Transition and Coexistence
- IP Address Assignment and Mapping
- Network Address Translation (NAT)
- Routing
- Multicast

IPv4 Packet Format



The IPv4 Header: Basics (1)

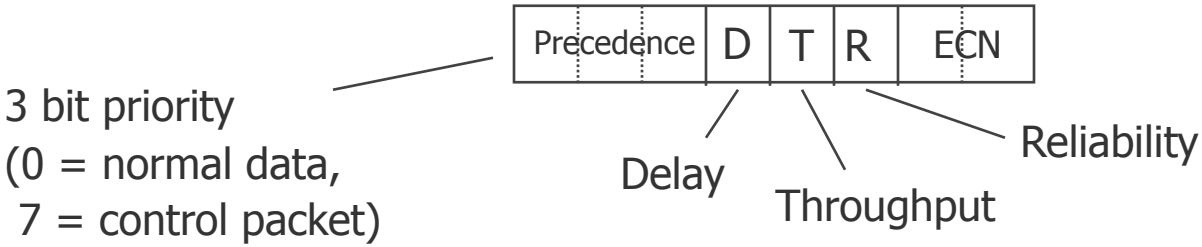
- **Version** (4 bits): IP version number (for simultaneous use of several IP versions)
- **Protocol** (8 bits): Which transport protocol is used in the data part (UDP, TCP, …)? To which transport process the packet has to be passed?
- **Header Checksum** (16 bits): 1's complement of the sum of the 16-bit half words of the header. Must be computed with each hop (since TTL changes)
- **Source Address/Destination Address** (32 bits): Network and host numbers of sending and receiving computer. This information is used by routers for the routing decision.



The IPv4 Header: Basics (2)

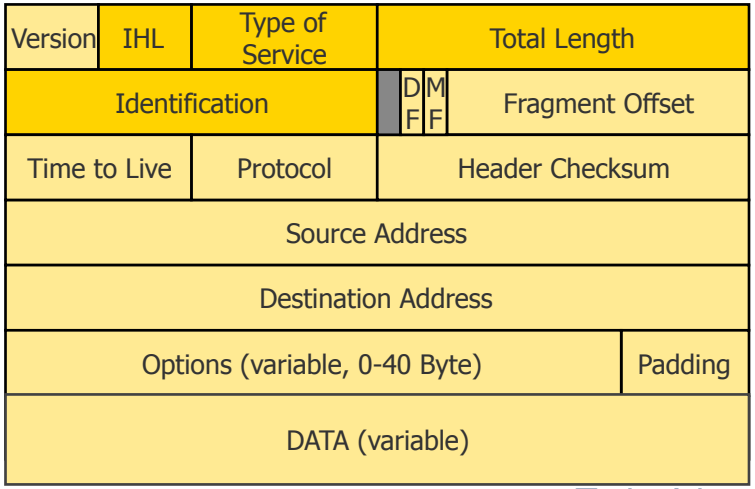
- **IHL** (4 bits): IP Header Length (in words of 32 bit; between 5 and 15, depending upon options)
- **Type of Service** (8 bits): Indication of the desired service: Combination of reliability (e.g. file transfer) and speed (e.g. audio)

➔ historical,today: Differenciated Service Code point (DSCP, see RFC 2474) + Explicit congestion Notification (ECN, see RFC 3168)



- **Total Length** (16 bits): Length of the entire packet ≤ 2^{16}-1 = 65,535 bytes

- **Identification** (16 bits): marking of a packet (for fragmentation)



The IPv4 Header: Limit Packet Travel Time

Problem

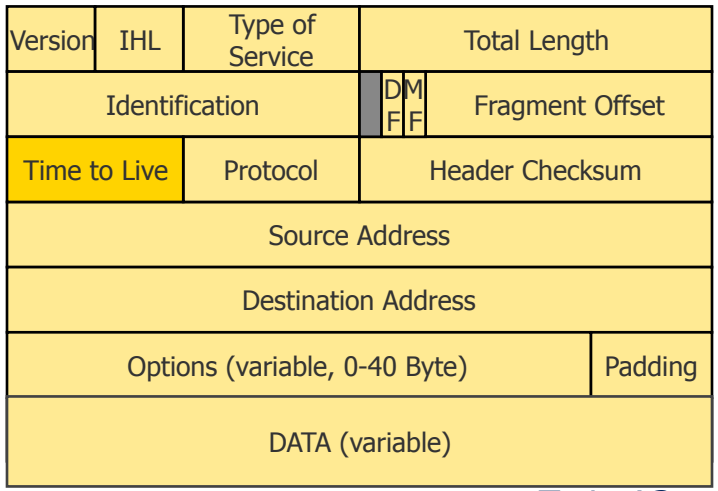
- Packets may travel endless (loops) or at least much longer than intended
- No central instance that monitors all packets

⇒Discard packets after a certain time

⇒Should be IP self-consistent: Rely solely on the IP header information

Solution

- **Time to Live** (TTL, 8 bits)
- Lifetime of packets is limited to a maximum of 255 hops
- Counter is reduced with each hop
- Discard packet if TTL=0
- Default value depends on the operating system
  - Linux: 64
  - Mac OS: 64
  - Windows :128

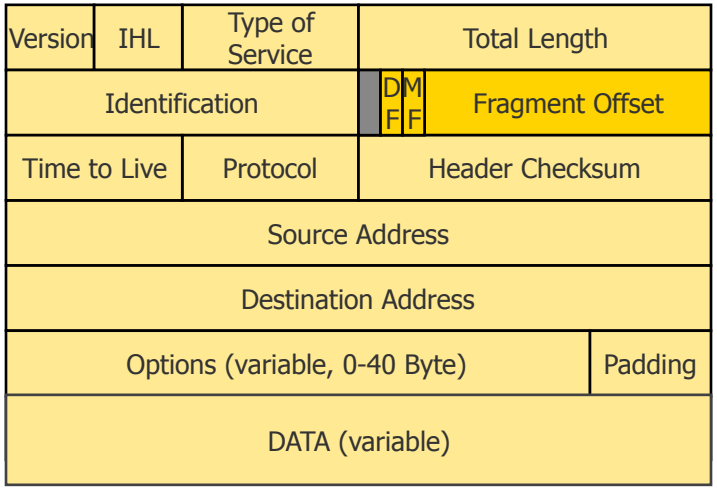


The IPv4 Header: Support Fragmentation

Problem

- Performance depends on the packet length and per packet operations
- Size restrictions, e.g., due to buffer, or allowed access time to a channel

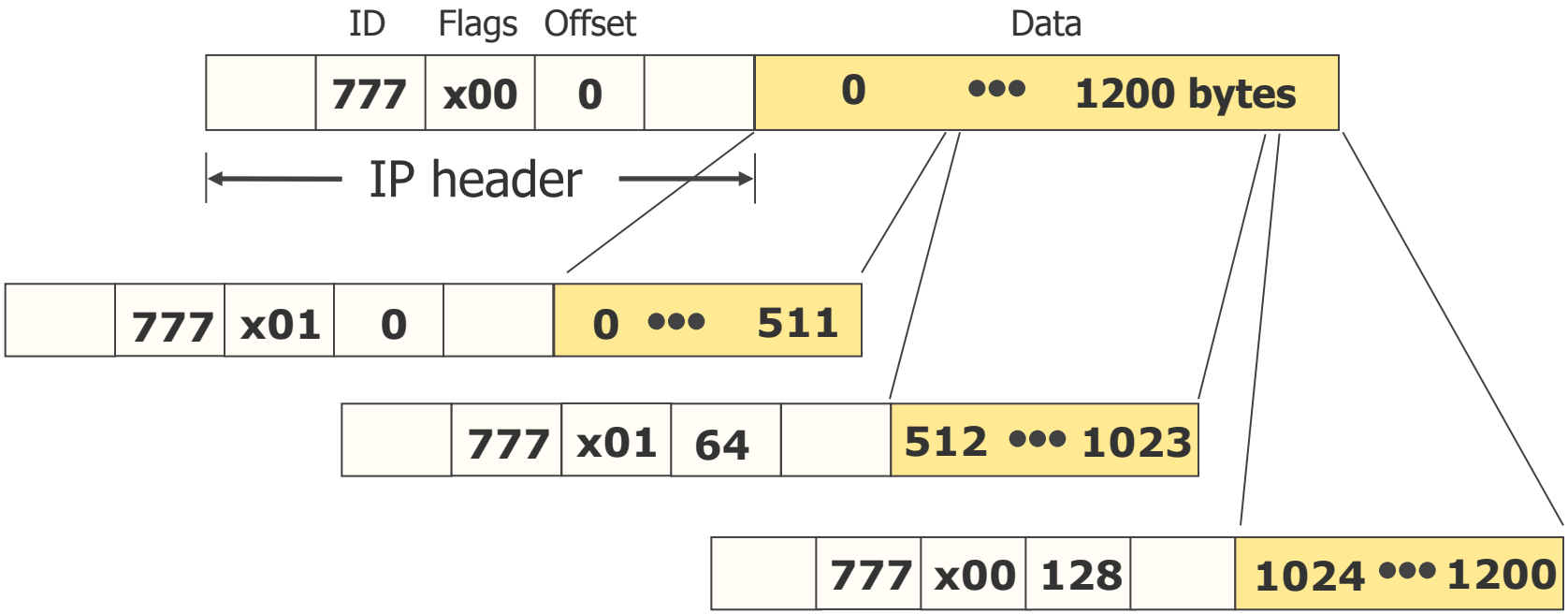
⇒ Need for dynamic sized packets with control



Solution

- **MF** (1 bit): More Fragments Bit
  - "1" - further fragments follow
  - "0" - last fragment of a datagram
- **Fragment Offset** (13 bits):
  - Sequence number of the fragments of a packet (2^{13} = 8192 possible unique fragment IDs)
  - Position of a packet (counted in **multiples of 8 byte**) a fragment belongs to (max. length of 8192 x 8 byte = 65,536 byte results for a packet)
- **Identification**: all fragments have the same value in the identification field
- **DF** (1 bit): Don't Fragment Bit
  - All routers must forward packets up to a size of 576 bytes, everything beyond that is optional
  - Larger packets with DF-bit=1 might be discarded (possible different delivery behavior per path)

IPv4 Packet Fragmentation Example



- The data length must be a multiple of 8 byte
  - Exception: The last fragment, only the remaining data is packed, padding to 8 byte units is not applied.
- If the "DF"-bit is set, the fragmentation is prevented.
  - May lead to packet being discarded

Extension of IPv4 Header: Options

- **Options** (≤40 bytes): Allows for future protocol extensions
  - Fields must be multiple of 4 byte, therefore padding is sometimes necessary
  - For details see: www.iana.org/assignments/ip-parameters

- 14 options are currently defined. The most relevant ones are:
  - **Security:** How secret is the transported information?
    - Application e.g. in military: Avoidance of crossing of certain countries/networks.
  - **Strict Source Routing:** Complete path defined from the source to the destination host by providing the IP addresses of all routers which are crossed.
    - For management, e.g., in case of damaged routing tables or for time measurements
  - **Loose Source Routing:** The carried list of routers must be passed in indicated order. Additional routers are permitted.
  - **Record Route:** Recording of the IP addresses of the routers passed.
    - Maximally 9 IP addresses possible, often too few nowadays!
  - **Time Stamp:** Records router addresses (32 bits) as well as a time stamp for each router (32 bits). Application, e.g., in fault management.
  - **Router Alert:** Transit routers should examine more closely content of IP packet

Support of IPv4 Header Options in the Real-World

Problem

- Most IP options require router action, which is costly
- IP options may reveal information
- IP options might be poorly implemented

⇒ Security concerns

⇒ Generic advice is to not support IP options (e.g., filter)

Question: Are IPv4 options supported in the Internet?

Answer: see for instance "IP Options are not an option", by R. Fonseca et al. (UCB report)

- Measurement studies show rare support
  - With enabled options, successful packet delivery ranges between 40% - 4%
  - When supported, option "support" varies: only forward packet vs. full consideration of option
  - Router behavior depends on details (e.g., which transport protocol, which option)

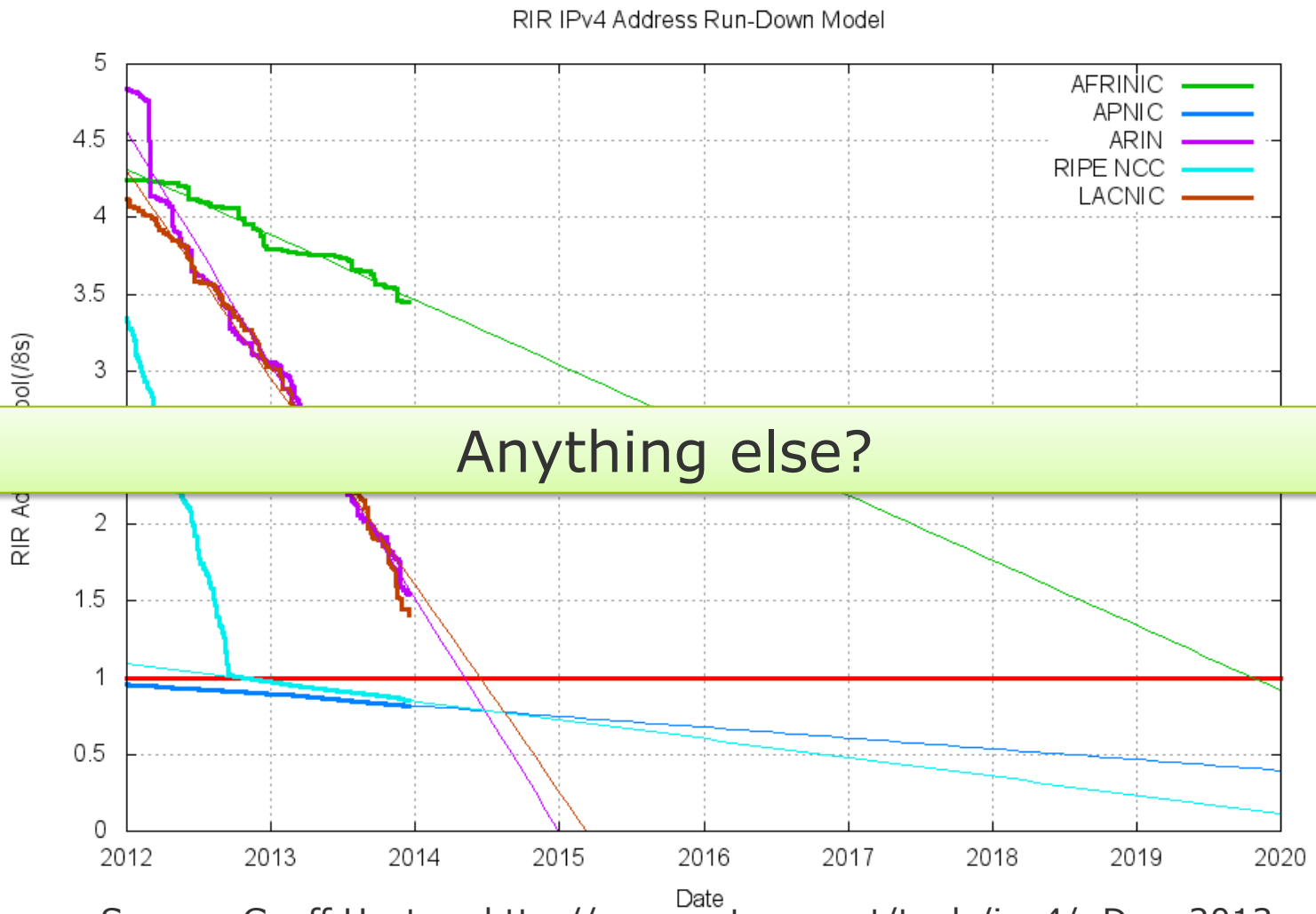
Conclusion

- Surprisingly, some packet options are supported but you should not rely on them

CONTENT of this CHAPTER

- Fundamental Goals of the Network Layer
- Network Layer Addressing
  - IPv4 Addressing
  - IPv6 Addressing
- Internet Protocol Design
  - Internet Protocol version 4
  - Internet Protocol version 6
  - IPv6 Migration: Transition and Coexistence
- IP Address Assignment and Mapping
- Network Address Translation (NAT)
- Routing
- Multicast

Basic Motivation (Reminder)



Anything else?

Source: Geoff Huston, http://www.potaroo.net/tools/ipv4/, Dec. 2013

More Motivation – IPv4 Service Problems

- **Address configuration**: Static, not stateless
- **Backbone routing**: Table explosion due to unstructured addresses
- **Security**: IP over IP tunneling
- **Multicasting**: Routing too complex
- **Anycasting**: Application specific solutions
- **QoS**: No flow support
- **Mobility**: Identifier/locator problem – inefficient triangular tunneling
- **Multihoming**: Provider abstraction missing

IPv6 Innovations

- Simpler Addressing and routing
  - Elimination of address bottleneck: 128 Bit addresses
  - Address hierarchy can (was intended to) simplify backbone routing
  - Several addresses per interface
- Simpler Protocol architecture
  - Slim, fixed header for fast processing
  - Optional extension headers
  - Format framework for header classes
  - No header checksum (already done by layer 2 and 4)
  - No fragmentation in routers
- Simpler administration
  - Autoconfiguration of interfaces without DHCPv6
  - Floating net masks, renumbering via prefix change
- Additional integrated security mechanisms
- Improved multicast, anycast, QoS and mobile services
- Support of Jumbograms (> 64 KB)
- Transition and coexistence concept IPv4 ←→ IPv6

IPv6 History

- IETF WG IP next generation (IPng) began to work in the early 90s
  - Winter 1992: 7 proposals for development of IP
    - CNAT, IP Encaps, Nimrod, Simple CLNP, PIP, SIP, TP/IX
  - Autumn 1993: several mergers lead to
    - 'Simple Internet Protocol Plus' (SIPP) and 'Common Architecture for the Internet' CATNIP
  - July 1994: IPng Area Director recommend roadmap (RFC 1752) on basis of SIPP (Steve Deering)

- Dec. 1995: **S. Deering**, **R. Hinden**, „Internet Protocol, Version 6 (IPv6) Specification" (RFC 1883, now RFC 2460)
  - July 1999: End user addresses available (RIPE NCC, APNIC, ARIN)

- May 2007: ARIN advises Internet Community on Migration to IPv6
- June 2012: World IPv6 Launch
  - … still waiting for massive adoption…

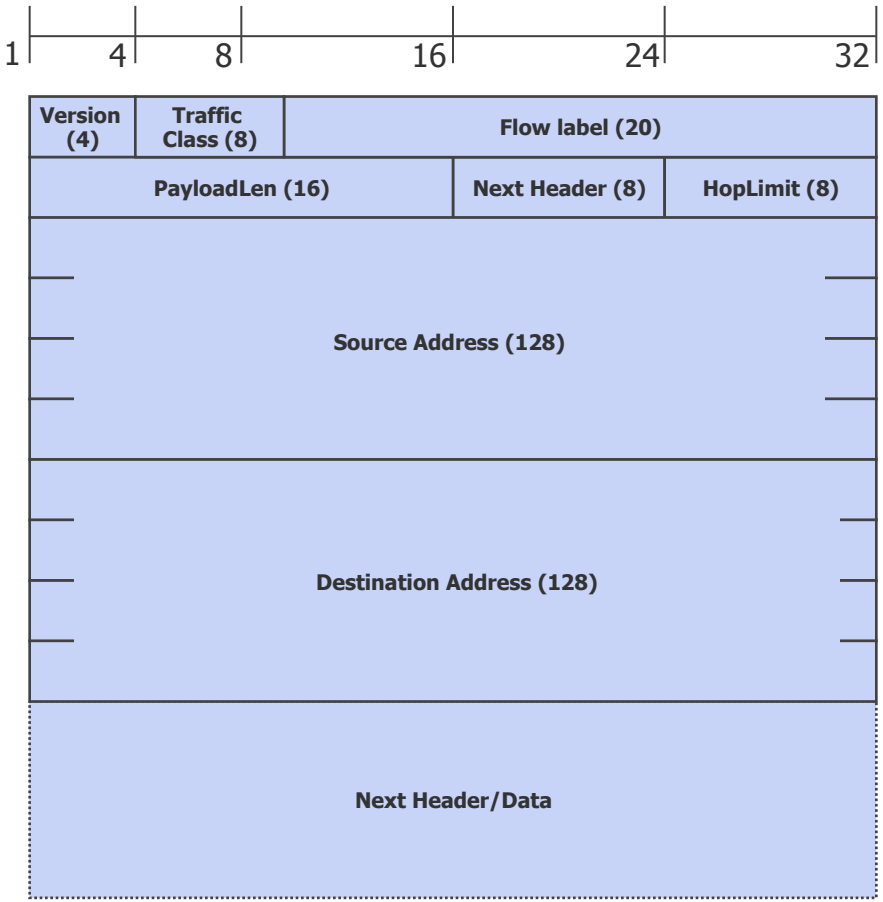
IPv6 Standardization

- Key components in standard track:
  - Base specification (RFC2460)
  - ICMPv6 (RFC2463)
  - RIP (RFC2080)
  - IGMPv6 (RFC2710)
  - Router Alert (RFC2711)
  - Autoconfiguration (RFC2462)
  - Neighbour Discovery (RFC2461)
  - IPv6 Addresses (RFC1884 ++)
  - BGP (RFC2545)
  - OSPF (RFC2740)
  - Jumbograms (RFC2675)
  - ....
- Standard specification for IPv6 over most link layer technologies:
  - **PPP** (RFC2023)
  - FDDI (RFC2467)
  - NBMA(RFC2491)
  - Frame Relay (RFC2590)
  - **Ethernet** (RFC2464)
  - Token Ring (RFC2470)
  - **ATM** (RFC2492)
  - ARCnet (RFC2549)
- Since then 100++ further standards: Flow labelling, MIPv6, 3GPP, Routing advertisement, ...

➡ **implemented in almost all internetworking system platforms**

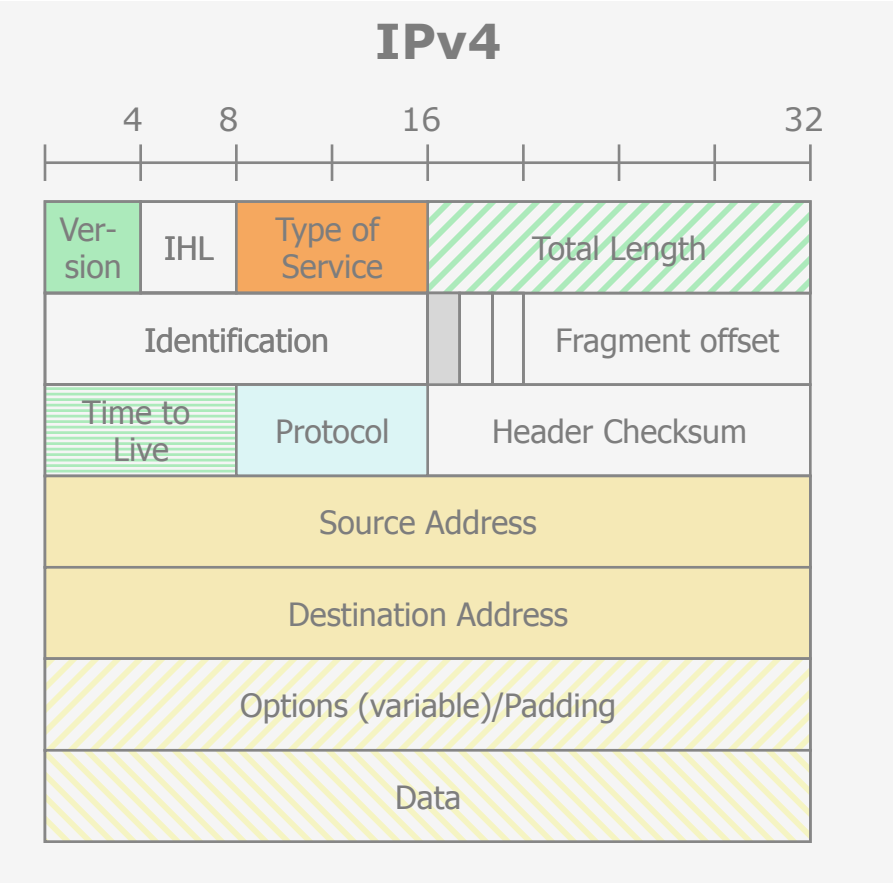
IPv6 Packet Format

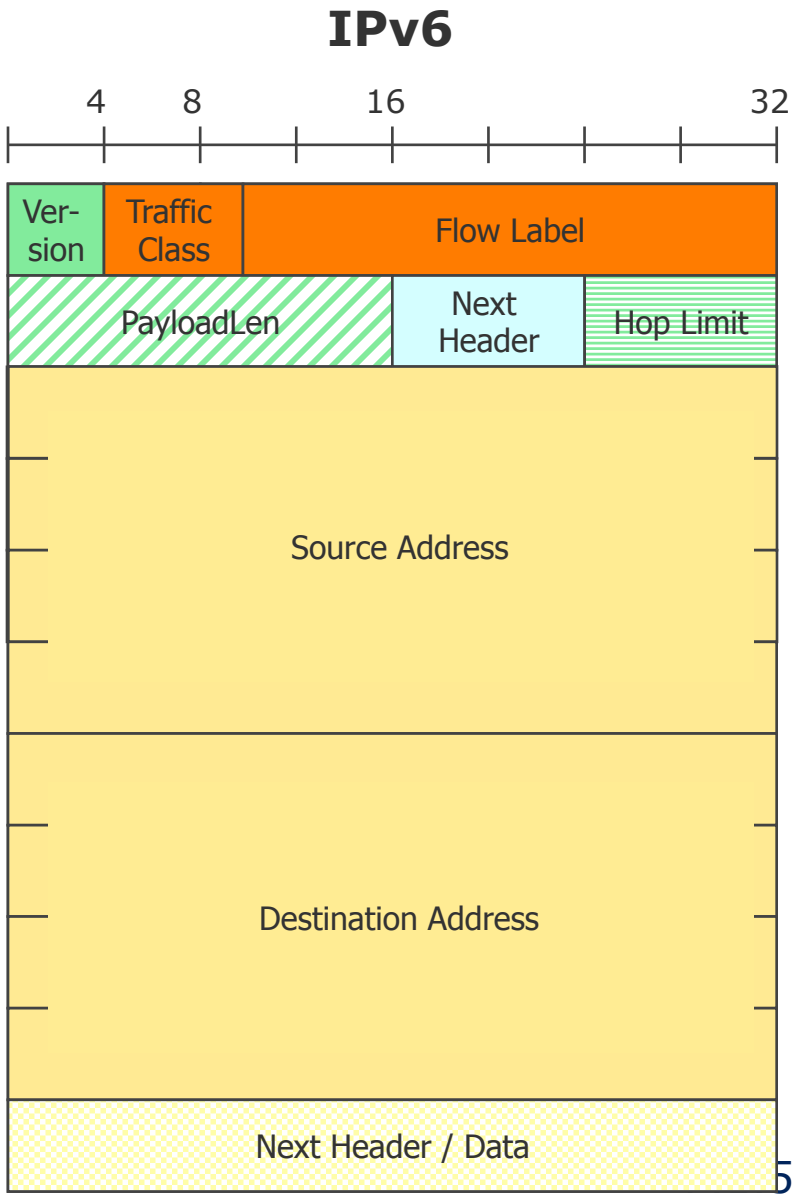
- **Version:** IP version number (4 bits)
- **Traffic Class:** classifying packets (8 bits)
- **Flow label:** virtual connection with certain characteristics/requirements (20 bits)
- **PayloadLen:** packet length after the 40-byte header (16 bits)
- **Next Header:** Indicates the type of the following extension header or the transport header (8 bits)
- **HopLimit:** decremented by one at each hop. If zero the packet is discarded (8 bits)
- **Source Address:** The address of the original sender of the packet (128 bits)
- **Destination Address:** The address of the receiver (128 bits).
  - Not necessarily the final destination, if there is an optional routing header
- **Next Header/Data**: if an extension header is specified, it follows after the main header. Otherwise, the data are following



The prefix of an address characterizes geographical areas, providers, local internal areas,…

IPv6 Packet Format vs. IPv4 Packet Format

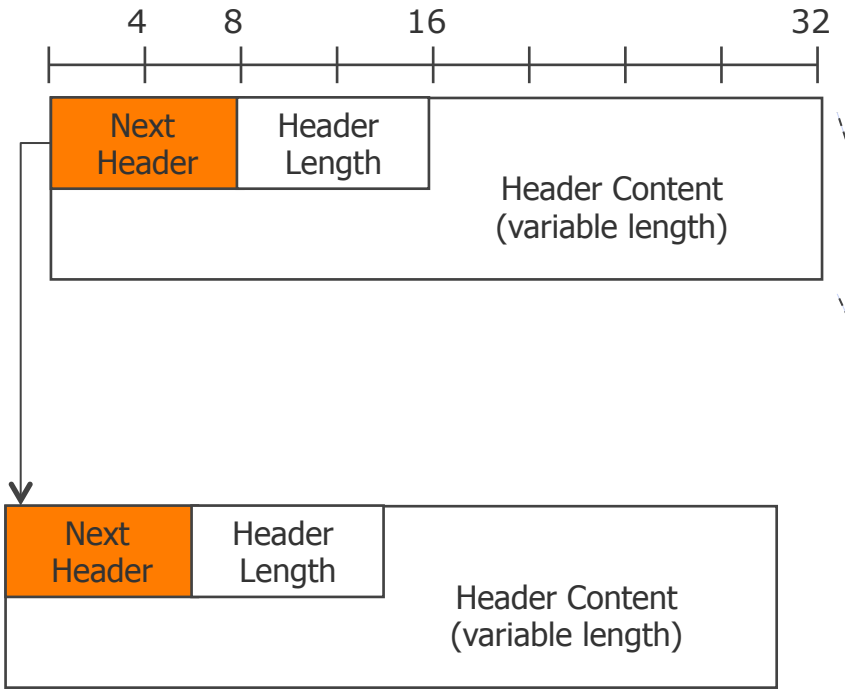


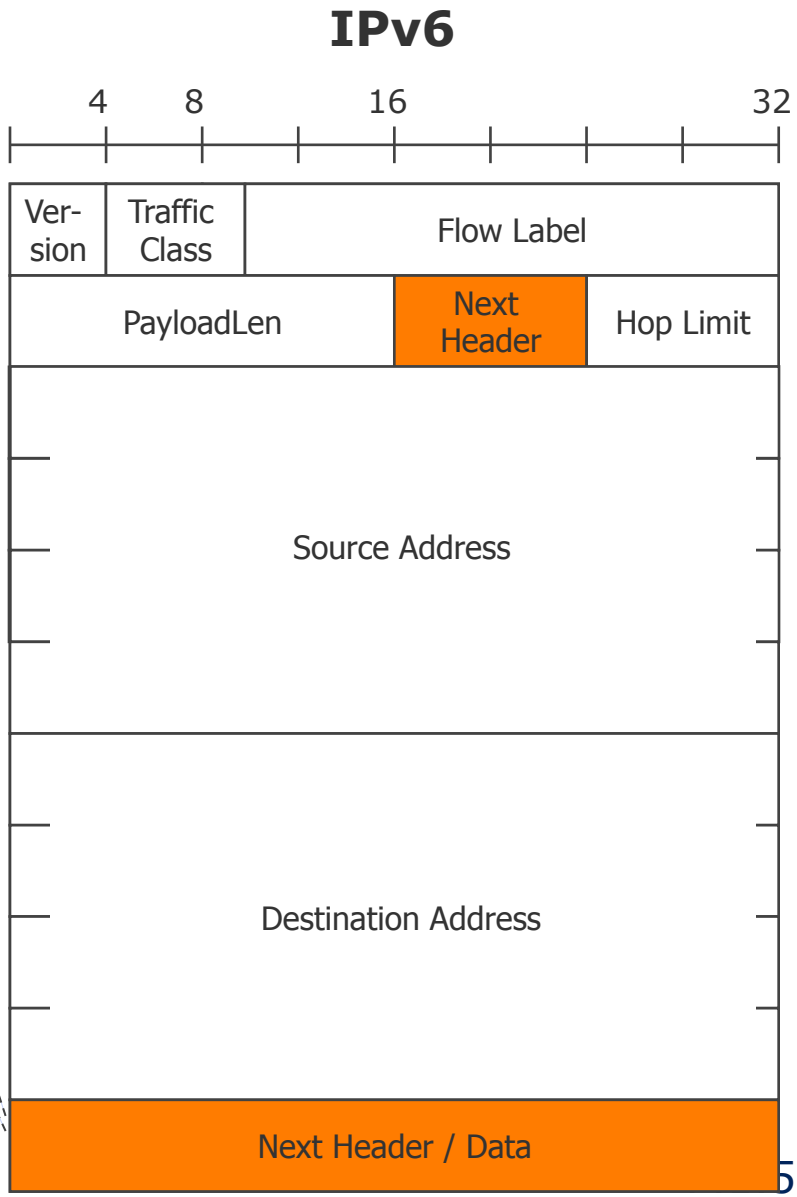


The IPv6 header is longer, but this is only caused by the longer addresses. Otherwise it is "better sorted" and thus faster to process by routers.

IPv6 Extension Headers

- Basic IPv6 header can be extended with extension headers
  - Each header references a possible successive header or data.





Basic IPv6 Extension Headers

Six basic extension headers defined:

1. **Routing**
   - Definition of a full or partly specified route
   - **Attention:** This header is deprecated since December 2007 (see RFC 5095)

2. **Fragmentation**
   - Fragmentation / defragmentation of frames
   - Difference with IPv4: Only the source can do fragmentation. Routers for which a packet is too large, only send an error message back to the source.

3. **Authentication**
   - Security information, authentication of the sender

4. **Encapsulation**
   - Tunneling, e.g., for encrypted data

5. **Hop-by-Hop**
   - Dedicated options to be processed by every router, information for single links
   - Momentarily only the support of Jumbograms (packets exceeding the normal IP packet length) is defined (length specification).

6. **Destination options**
   - Additional information for the destination

IPv6 Extension Headers: Examples

- Examples:

IPv6 header Next Header = TCP	TCP header + data

IPv6 header Next Header = Routing	Routing header Next Header = TCP	TCP header + data

IPv6 header Next Header = Routing	Routing header Next Header = Fragment	Fragment header Next Header = TCP	TCP header + data

- Remark: each extension should occur at most once!

IPv6 Extension Headers: Options

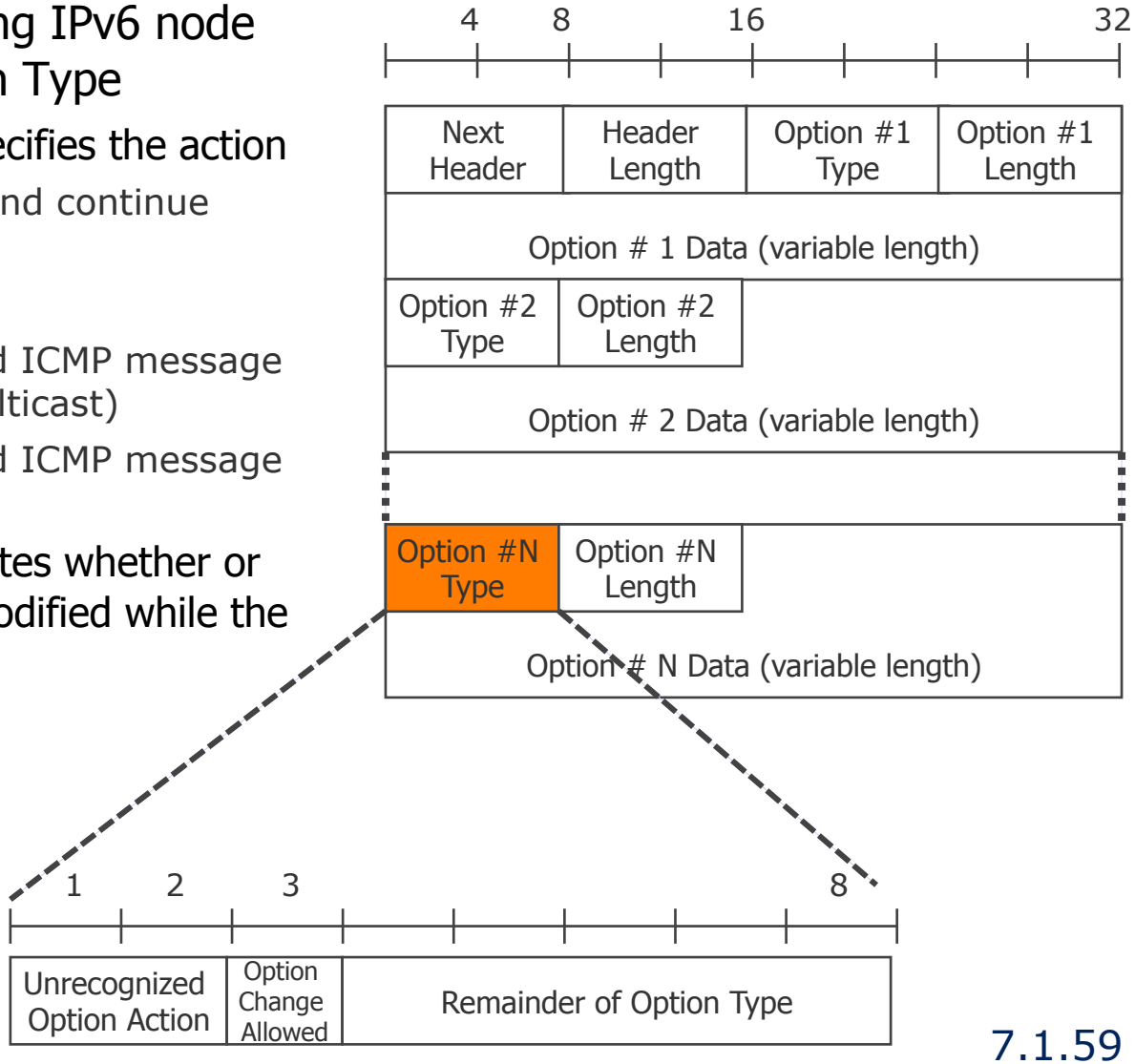
- For maximum flexibility, options are possible in headers
  - Option headers have no length limit (IPv4: 40 Octets), padding to 8 Octets
  - Options are encoded as TLV (type-length-value)

Option Type	Opt Data Length	Option Data
8 bits	8 bits	variable length

- Option headers are processed only at destination host, not by routers.
  - Exception: Hop-by-Hop Option Header

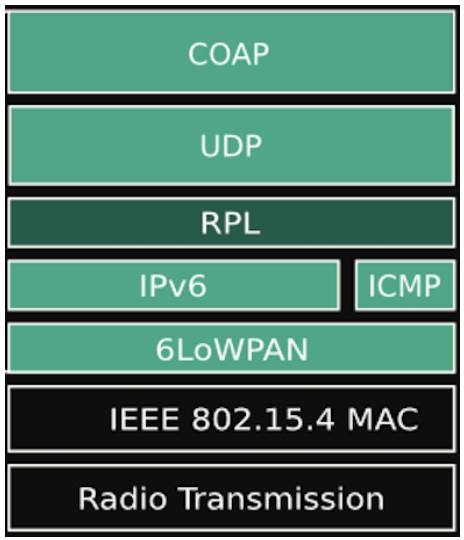
IPv6 Extension Headers: Structure with Options

- Option Type also determines the action that must be taken if the processing IPv6 node does not recognize the Option Type
  - The highest-order two bits specifies the action
    - 00: skip over this option and continue processing the header
    - 01: discard the packet
    - 10: discard the packet and ICMP message to source (unicast and multicast)
    - 11: discard the packet and ICMP message to source (only unicast)
  - The Option Change flag indicates whether or not the Option Data can be modified while the datagram is en route



IPv6 Packet Size Requirements

- IPv6 requires that every link has MTU ≥ 1280 bytes
  - MTU: Maximum Transmission Unit
- Links which cannot convey 1280 byte packets have to apply link-specific fragmentation and assembly at a layer **below** IPv6
  - e.g. IEEE 802.15.4 packets are ~ 10 times less than that! ➡ use of 6lowpan
- Links that have a configurable MTU must be configured to have an MTU of at least 1280 bytes
  - It is recommended that they be configured with an MTU ≥ 1500 bytes



IPv6 Path MTU Discovery Process

- Path MTU Discovery
  - MTU: Maximum Transmission Unit
    - Maximum packet size in octets, that can be conveyed in one piece over a link
  - Path MTU: The minimum link MTU of all the links in a path between a source node and a destination node

- Path MTU Discovery Process
  - The originating node assumes the Path MTU is the MTU of the first hop in the path.
  - A trial packet of this size is sent out.
  - If any link is unable to handle it, an ICMPv6 Packet Too Big message is returned.
  - The originating node iteratively tries smaller packet sizes until it gets no complaints from any node, and then uses the largest MTU that was acceptable along the entire path.

CONTENT of this CHAPTER

- Fundamental Goals of the Network Layer
- Network Layer Addressing
  - IPv4 Addressing
  - IPv6 Addressing
- Internet Protocol Design
  - Internet Protocol version 4
  - Internet Protocol version 6
  - IPv6 Migration: Transition and Coexistence
- IP Address Assignment and Mapping
- Network Address Translation (NAT)
- Routing
- Multicast

Transition from IPv4 to IPv6

Background

- Incremental deployment: IPv6 cannot be introduced "overnight"… for some time both IP variants will be used in parallel
  - How to enable two IPv6-based hosts to communicate if only an IPv4-based network is in between?
  - How can an IPv4 and IPv6 communicate with each other?

⇒ Porting and migration are necessary

Porting

- Making an application ready for IPv6

Migration

- Seamless step-by-step transition from IPv4 to IPv6
- Temporarily use both IPv4 and IPv6 protocols in parallel

IPv4 → IPv6 Porting: Making Applications IPv6 Ready

- Source and binary code IPv6 compatibility for existing applications
  - For now 'everything goes on' assuming only IPv4 is around but …
- Changes needed for name-to-address translation
  - New functions to support IPv6 and IPv4
- Changes needed for address conversion functions
  - New functions to support IPv6 and IPv4
- Changes needed for DNS resolver
  - Returns IPv6 or IPv4 address, or both
- POSIX changes needed to accommodate full abstraction from IP version
  - Indirection for address data structure, new for IPv6 (see also RFC 3493)
  - `addrinfo` is linked list of interface address structs
- Almost all standard (open source) applications are already ported

IPv4 → IPv6 Porting: Socket API Changes

See for instance RFC 3493

	IPv4	IPv6	
Data structures	AF_INET	AF_INET6	
	in_addr sockaddr_in	in6_addr sockaddr_in6	
Address conversion functions	inet_aton() inet_addr()	inet_pton()	IPv4 and IPv6 functions
	inet_ntoa()	inet_ntop()	
Name-to-address functions	gethostbyname() gethostbyaddr()	getnameinfo()* getaddrinfo()*	

Remark:

- Your application should be IP version agnostic
  - The original socket API prevents fast IPv6 deployment

⇒ **Update to use new API calls**

IPv4 → IPv6 Migration Techniques

Many migration techniques have been designed and implemented according to the following approaches:

Dual-Stack techniques

Allowing the coexistence of IPv4 and IPv6 on the same device and subnet

Tunneling

Connecting IPv6 regions over IPv4 regions

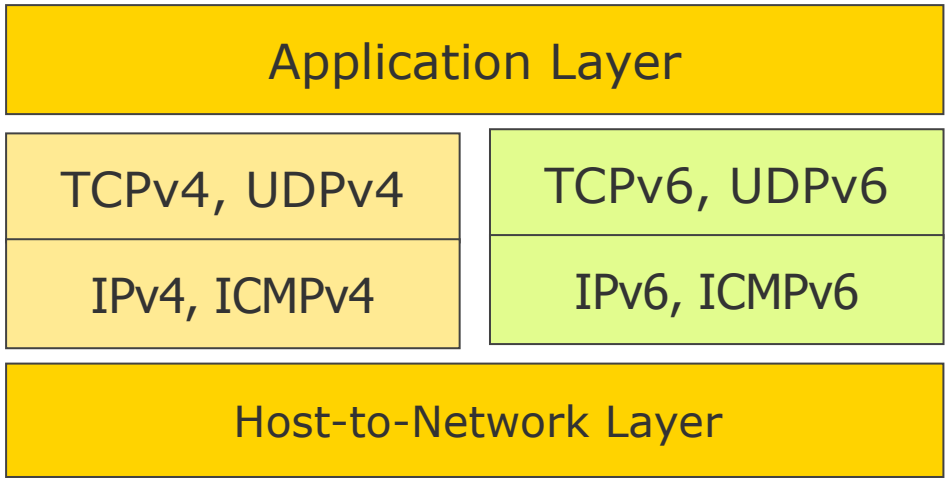
Protocol Translator (NATs)

Letting IPv6 devices speak with IPv4 devices

During migration, the combined use of all above methods is likely

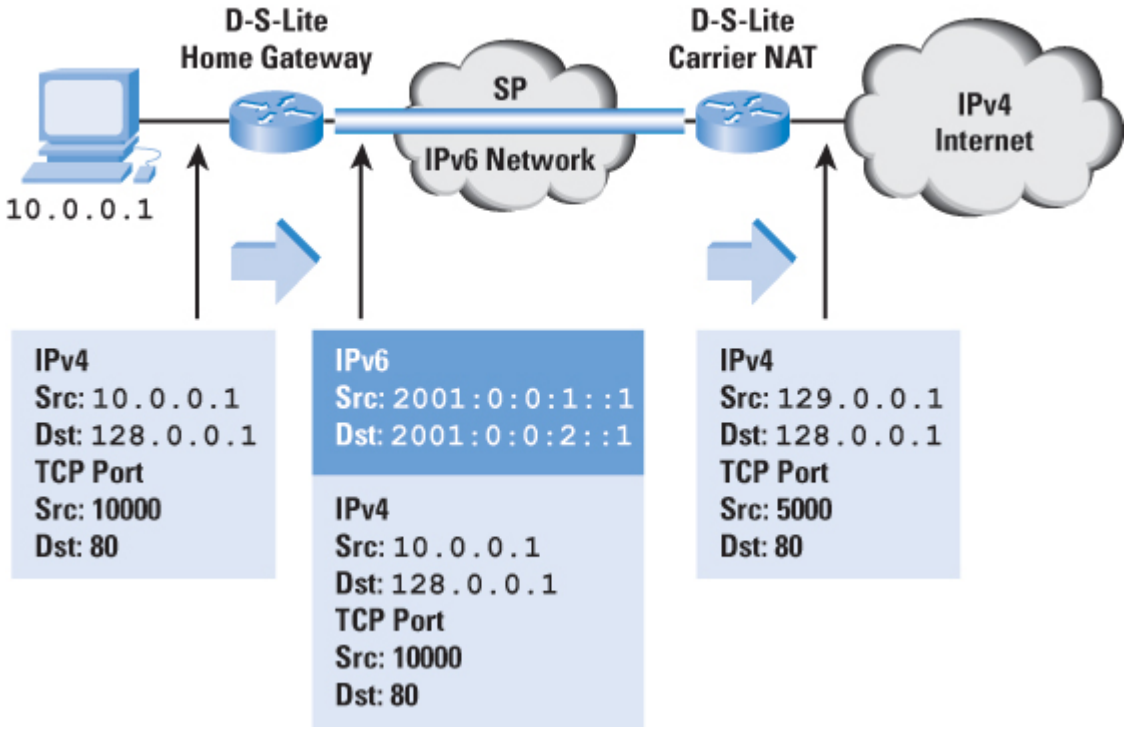
Dual Stack - Principles

- Upon activation of IPv6, IPv4 can continue being used (multi protocol approach)
- Devices can keep their addresses (IPv4 in IPv6)
- Application / libraries choose the IP version
  - Depending on DNS answer with IPv6 preference
  - Depending on received IP packets
- Dual stack operation can continue indefinitely, allowing step by step porting of applications
  - But requires an increasing use of IPv4 NATTING (recursive)
- Problems result from inconsistencies of Network / DNS configurations
  - Example: DNS returns IPv4 and IPv6 address, application selects v4 but v4 is already disabled



Dual Stack Lite (RFC 6333)

- Idea: No public IPv4 at the customer – but dual at system
  - IPv4 over IPv6 tunnel to Carrier-grade IPv4-IPv4 NAT
  - Enables many customers to share a single globally unique IPv4 address



Source: Geoff Huston, "NAT++: Address Sharing in IPv4", The Internet Protocol Journal, Volume 13, No.2

Migration by Tunnels

Objective

- Embedding of IPv6 packets in IPv4 packets

Several methods to build an IPv4-IPv6 tunnel

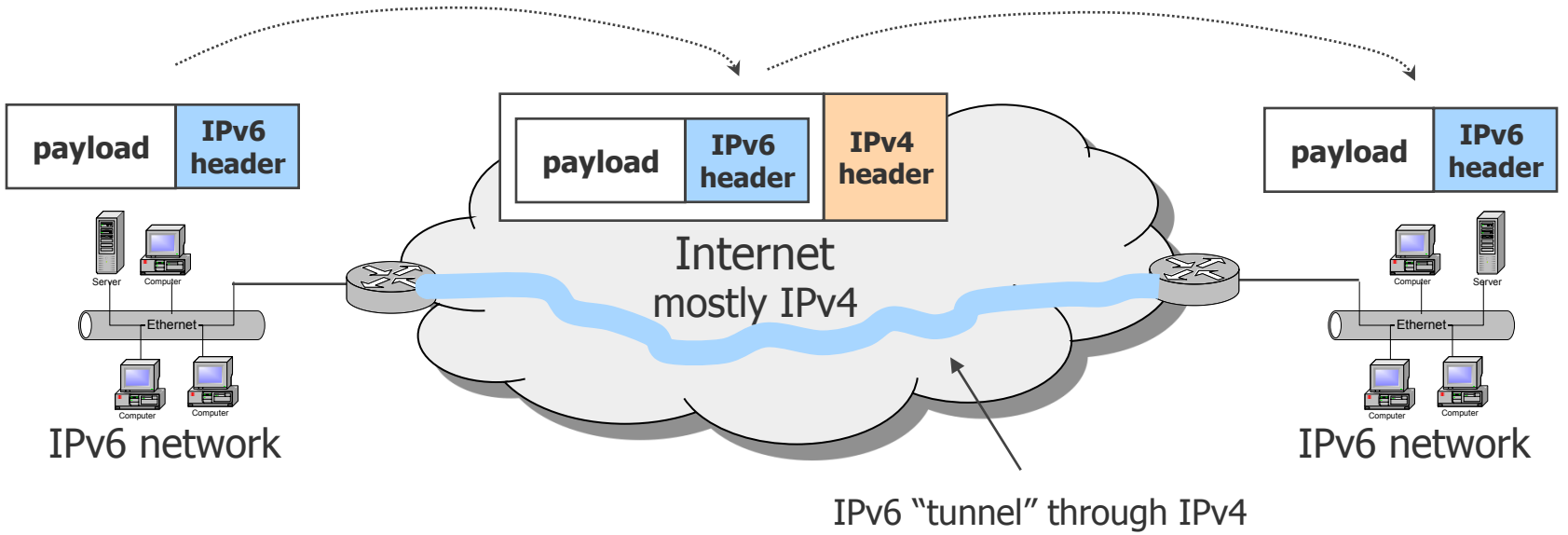
- Do-it-yourself (manually)
- Using a Tunnel Broker
  - Web based application to build a tunnel, e.g., https://tunnelbroker.net/
- Using 6-over-4 (intradomain)
- Using 6-to-4 (interdomain, IPv4 address as IPv6 site prefix)

Further perspectives

- IPv6 uses IPv4 as a virtual link layer
- IPv6 VPN over IPv4

IPv4-IPv6 Tunneling: General Overview

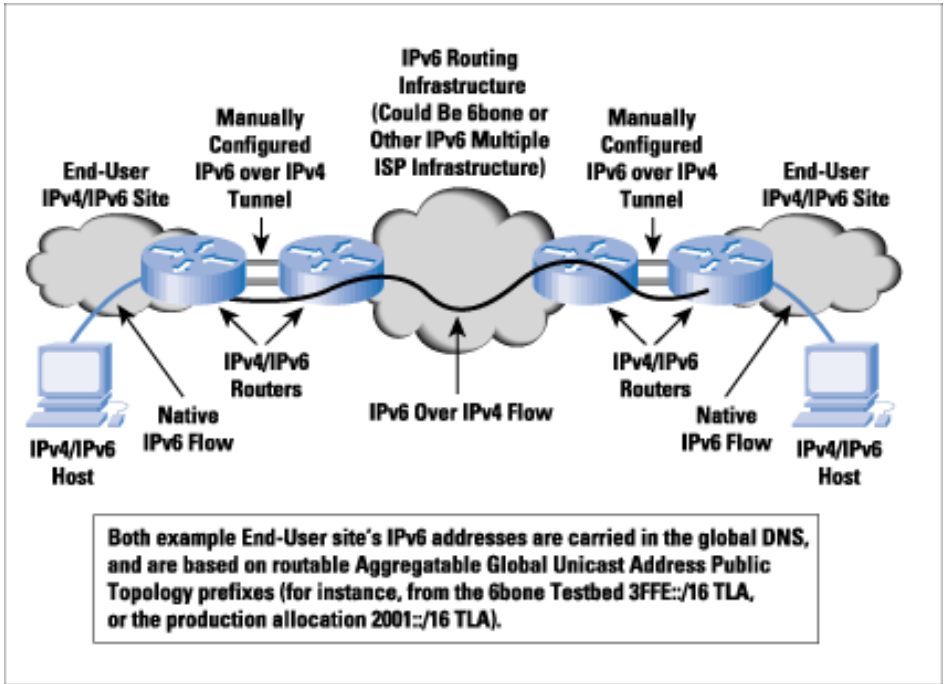
- Router at the entry of an IPv4-based network encapsulates an incoming IPv6 packet into a new IPv4 packet with destination address of the next router also supporting IPv6



- **More overhead** because of two headers for one packet, **but no information loss**
- **Tunneling** is a general concept also used for multicast, VPN, etc: it simply means "pack a whole packet as it is into a new packet of different protocol"

6-over-4 Tunneling

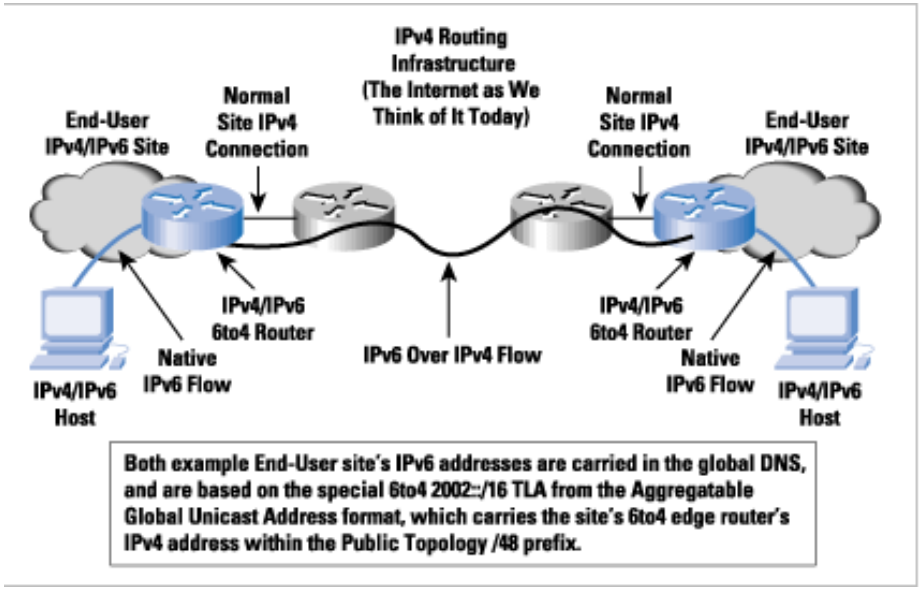
- Targeted scenario: IPv6 islands in an IPv4 world, but with IPv6 backbone
  - IPv6-IPv4 inter-connects: Embedding
  - IPv6-IPv6 inter-connects: Following tunnel configurations
- Problem: need manual configuration of tunnels & "peering" agreements…

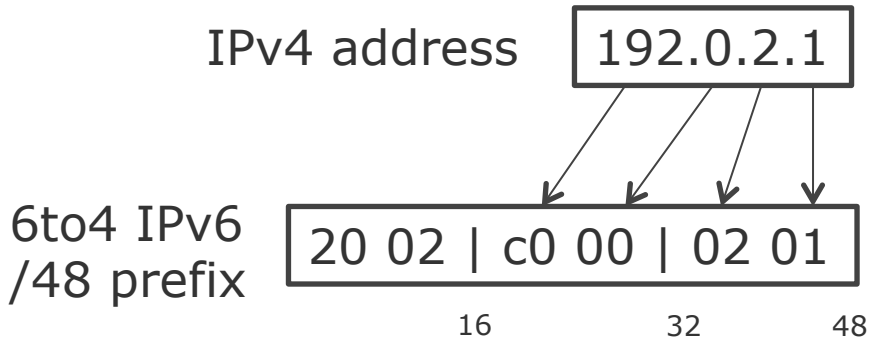


Source: Brian E. Carpenter et al., "Routing IPv6 over IPv4", The Internet Protocol Journal, Volume 13, No.1

6-to-4 Tunneling

- Targeted scenario: isolated IPv6 islands in an IPv4 world
- Allows IPv6 islands to automatically interconnect, with minimized manual configuration
  - Defines automated **point-to-multipoint tunnels** (RFC 3056)
  - Uses IPv4 as a non-broadcast multi-access network, and **anycast** on top of that
  - Assigns an IPv6 prefix to each IPv4 address

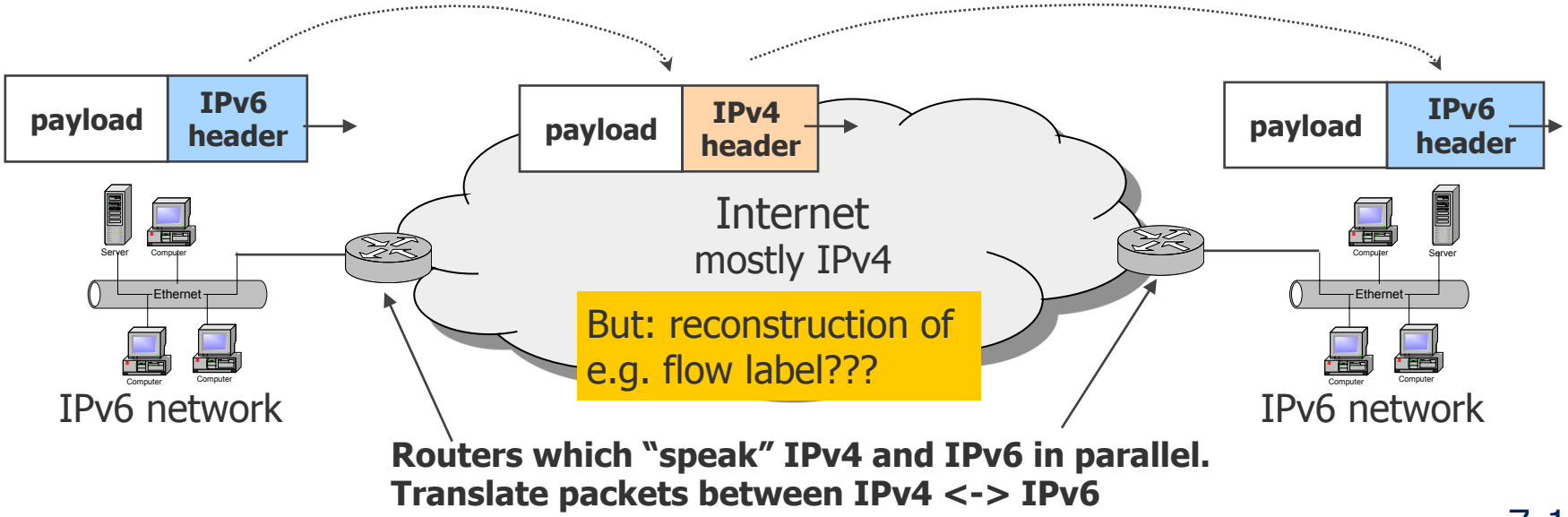




Source: Brian E. Carpenter et al., "Routing IPv6 over IPv4", The Internet Protocol Journal, Volume 13, No.1

Translation: NAT and IPv4 to IPv6 Translation

- NAT: Network Address Translation
  - Initially introduced to "save" on IPv4 address needs (details later in this course)
- Translation is a problem in the context of transition to IPv6:
  - 6to4 has issues to traverse NAT ➔ need for **TEREDO** extension for NAT traversal
- Translation is also a solution in the context of transition to IPv6:
  - Ingress router can translate IPv6 into IPv4, outgress router translates back to IPv6
  - Further problem: header fields are not compatible!



People Who Design IP Protocols are Creative (and Somewhat Crazy)

- Listen to http://youtu.be/_y36fG2Oba0