Computer Networking and IT Security (CNS)

INHN0012 – WiSe 2025/26

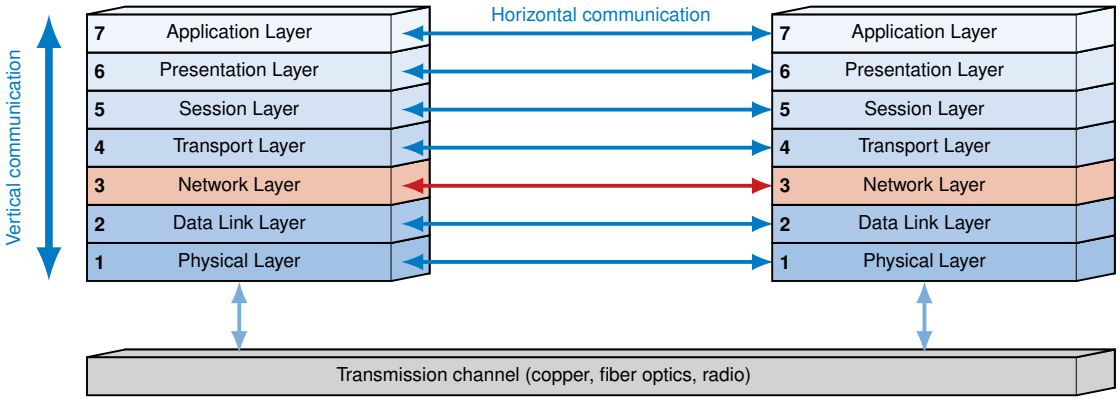
Lorenz Lehle, Stephan Günther

Chair of Distributed Systems and Security
School of Computation, Information and Technology
Technical University of Munich

Chapter 3: Network Layer

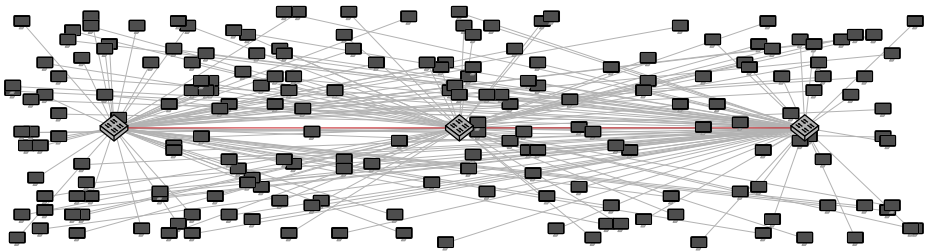
Chapter 3: Network Layer

Placement in the ISO/OSI model



To what extent are local area network scalable?

- All attached nodes are directly or via a few switches reachable
- MAC addresses have not logical structure (except for splitting into OUI and device ID)
- There is no way to group devices into smaller networks (subnets) based on MAC addresses



Tasks of the network layer:

- Coupling of different local area networks
- Structured division into smaller subnetworks
- Logical and globally unique addressing of devices
- Routing between devices over multiple hops

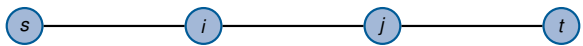
Chapter 3: Network Layer

Switching Modes

There are three fundamental types of switching:

1. Circuit switching
   “Reserve a dedicated line between transmitter and receiver”
2. Message switching
   “Choose the next hop for each message individually and forward the message as a whole”
3. Packet switching
   “Split a message into several smaller packets and send each packet independently of the others”

In the following, we characterize these three types of switching based on the example network



with $n = 2$ intermediate nodes (i and j) with respect to total delay T of a transmission of L bit over a distance d and thereby motivate the advantages of packet switching.

Circuit switching

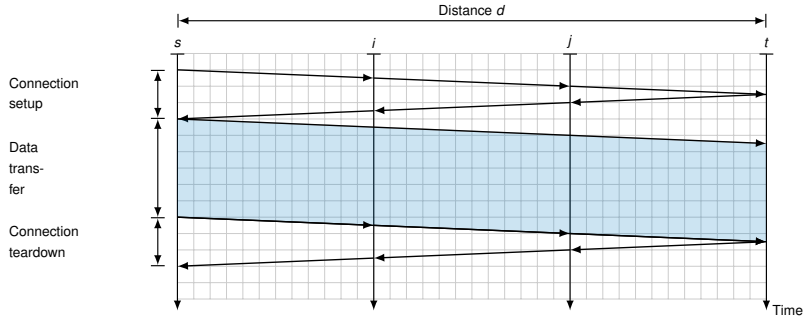
During a connection oriented transmission we can differentiate between three phases:

1. Connection setup
   - Exchange of signalling messages to establish a dedicated connection between source and destination.
   - This step includes the path selection, which is performed before the data transmission starts.
2. Data exchange
   - The line can exclusively used by the communication partners.
   - Addressing of the communication partner can be largely dispensed with during transmission (point-to-point connection).
3. Connection teardown
   - Exchange of signalling messages to tear down the connection.
   - The resources occupied by the connection are released for subsequent connections.

Transmission delay when using circuit switching

We assume that

- serialization times of signalling messages can be neglected,
- processing times at intermediate nodes can be neglected, and
- the source s has data of length L to be transmitted in a single message.



$$T_{CS} = 2t_p + t_s + 2t_p = t_s + 4t_p = \frac{L}{r} + \frac{4d}{\nu c_0}$$

Advantages of circuit switching

- Consistent quality of the dedicated connection after connection establishment
- Fast data transfer without the need to make further switching decisions

Disadvantages of circuit switching

- Waste of resources if line is not permanently utilized, as line is reserved for exclusive use
- Connection setup can be complex and may require far more time than propagation delays suggest (cmp. establishing an internet connection with analog modems or ISDN)
- High effort in switching physical connections

Use in today's networks

- Circuit switching is commonly replaced by packet switching (e. g. Voice over IP)
- Service providers may still use circuit switching at least in the form of virtual circuits (e. g. Frame Relay, ATM[1], MPLS[2]) to guarantee a certain transfer rate to their customers

[1] Asynchronous Transfer Mode

[2] Multi Protocol Label Switching

Message switching

Modifications compared to circuit switching:

- No connection setup and tear down
- A header of length L_H is prepended to the message

- In particular, the header contains address information that is suitable for uniquely identifying the sender and receiver even across multiple intermediate stations
- The resulting PDU is transferred as a whole

Properties:

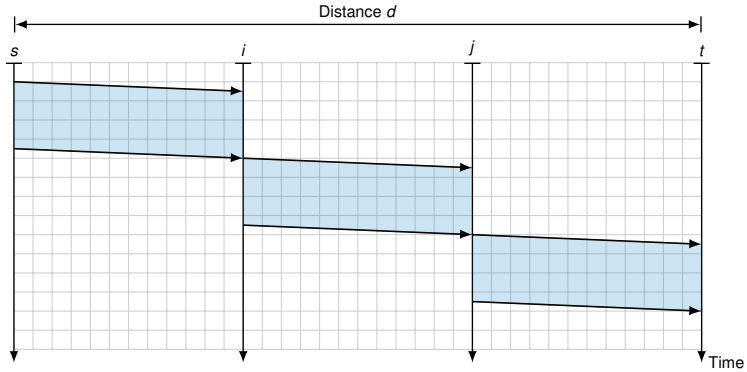
- Possibility for asynchronous communication, i. e. messages may be sent to a destination that is currently unable to receive messages
- Possible time savings since there is no more connection setup and teardown

Analogy: Parcel service providers

- Sender packs goods and provides the package with address information (header)
- The addresses uniquely identify sender and recipient worldwide and have a logical structure that allows efficient assignment in the postal transport network
- If the recipient can not receive the package, it is kept at the postal office

Transmission delay when using message switching

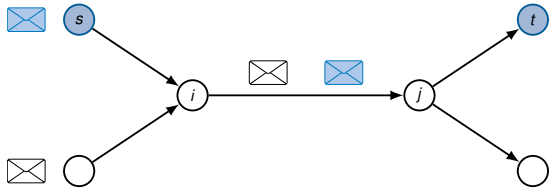
Reminder: We assume $n = 2$ intermediate nodes i and j.



$$T_{MS} = (n+1) \cdot t_s + t_p = (n+1) \cdot \frac{L_H + L}{r} + \frac{d}{\nu c_0}$$

Multiplexing on message level

- The elimination of fixed paths enables the joint use of partial routes
- This corresponds to dynamic time multiplex (Time Division Multiplex, TDM)



Advantages:

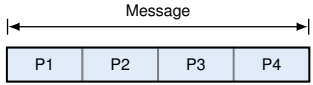
- Flexible time-based multiplexing of messages
- Better utilization of the channel capacity
- No delay due to prior connection setup

Disadvantages:

- Messages must be buffered at i if (i,j) is in use
- Loss of messages due to limited buffer capacity possible (congestion $\rightarrow$ Chapter 4)
- A message must be serialized multiple times

Difference with respect to message switching:

- Messages are no longer transmitted as a whole but divided into smaller units, the data parts of packets:



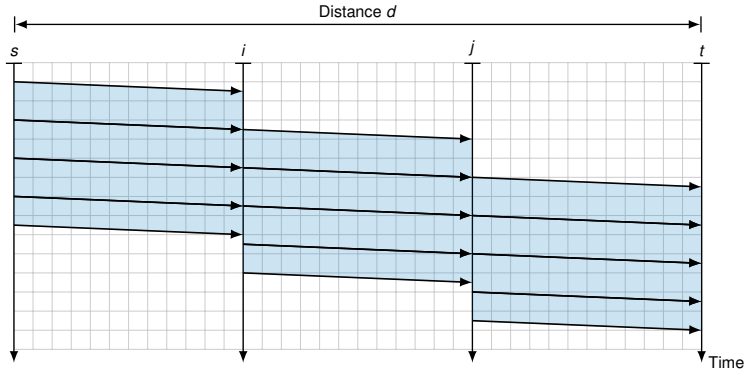
- Each packet is provided with its own header, which contains all the information for forwarding and also for reassembly, if necessary:

- Packets are routed independently from each other, i. e. packets belonging to the same message may be routed over different paths from source to destination – and may thus also arrive out of order.

- In general, individual packets do not have to be of equal size, but there are requirements for maximum packet size accompanied by a payload of maximum length $p_{\max}$.

Transmission time when using packet switching

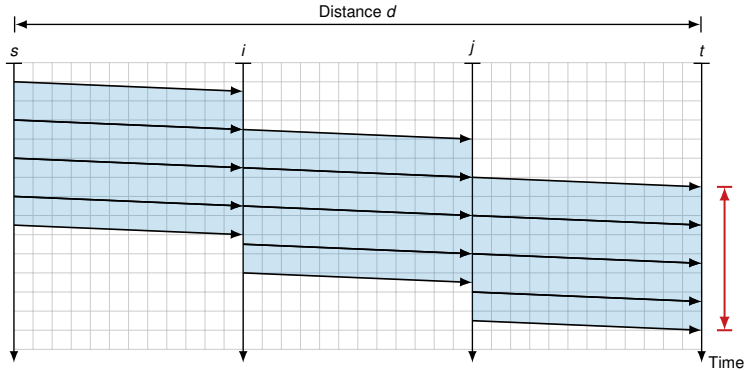
Reminder: we assume $n = 2$ intermediate nodes i and j.



$$T_{PS} = \frac{1}{r}\left(\left\lceil \frac{L}{p_{\max}} \right\rceil \cdot L_h + L\right) + \frac{d}{\nu c_0} + n \cdot \frac{L_h + p_{\max}}{r}$$

Transmission time when using packet switching

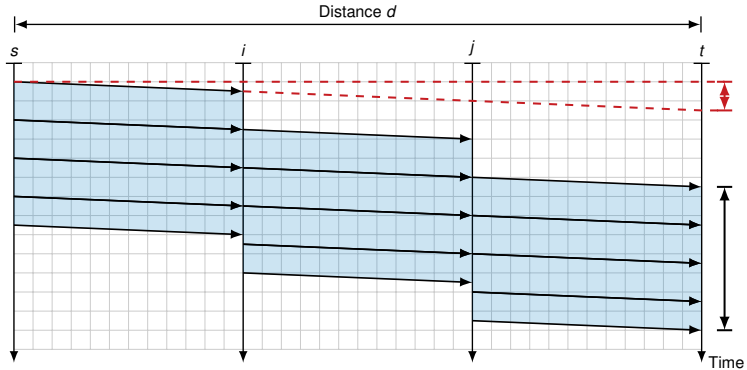
Reminder: we assume $n = 2$ intermediate nodes i and j.



$$T_{PS} = \frac{1}{r}\left(\left\lceil \frac{L}{p_{\max}} \right\rceil \cdot L_h + L\right) + \frac{d}{\nu c_0} + n \cdot \frac{L_h + p_{\max}}{r}$$

Transmission time when using packet switching

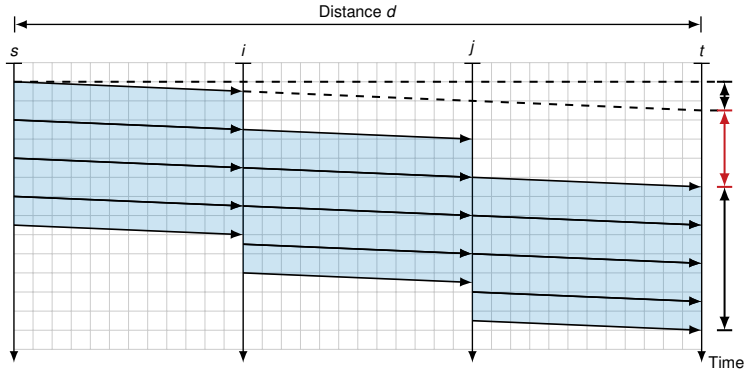
Reminder: we assume $n = 2$ intermediate nodes i and j.



$$T_{PS} = \frac{1}{r}\left(\left\lceil \frac{L}{p_{\max}} \right\rceil \cdot L_h + L\right) + \frac{d}{\nu c_0} + n \cdot \frac{L_h + p_{\max}}{r}$$

Transmission time when using packet switching

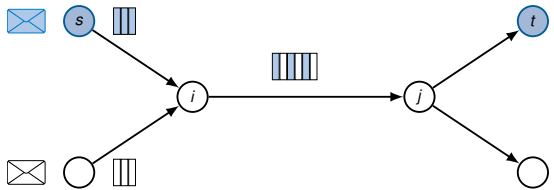
Reminder: we assume $n = 2$ intermediate nodes i and j.



$$T_{PS} = \frac{1}{r}\left(\left\lceil \frac{L}{p_{\max}} \right\rceil \cdot L_h + L\right) + \frac{d}{\nu c_0} + n \cdot \frac{L_h + p_{\max}}{r}$$

Multiplexing on packet level

- By switching small packets instead of long messages, bottlenecks are used more fairly
- If packets are lost, only parts of a larger message need to be repeated



Advantages:

- Flexible time-based multiplex of individual packets
- Buffering of packets instead of whole messages

Disadvantages:

- Packet loss due to limited buffer capacity still possible
- Every packet requires its own header (overhead)
- Destination must reassemble the message

Comparison of all three switching modes

$$\text{Circuit switching:} \quad T_{CS} = \frac{L}{r} + 4\frac{d}{\nu c_0}$$

$$\text{Message switching:} \quad T_{MS} = (n+1)\frac{L_h + L}{r} + \frac{d}{\nu c_0}$$

$$\text{Packet switching:} \quad T_{PS} = \frac{1}{r}\left(\left\lceil \frac{L}{p_{\max}} \right\rceil \cdot L_h + L\right) + \frac{d}{\nu c_0} + n \cdot \frac{L_h + p_{\max}}{r}$$

Example:

- Distance $d = 1000\,\text{km}$ between source and destination
- Assuming fiber links with relative propagation $\nu = 0.7$
- There are $n = 2$ intermediate nodes
- The data rate is $r = 777\,\text{kbit/s}$ on all segments
- The message to be transmitted has length $L = 5\,\text{MiB}$
- The maximum payload size of packets is $p_{\max} = 1480\,\text{B}$
- The header per message or packet is $L_h = 20\,\text{B}$

Then we obtain the following results: $T_{CS} \approx 54\,\text{s}$, $T_{MS} \approx 162\,\text{s}$, and $T_{PS} \approx 55\,\text{s}$.

Packet switching

Where are these switching modes used?

Circuit switching:

- Analog telephone systems (POTS)
- Internet dial-up (“last mile”)
- Site-to-site networks of companies
- Virtual circuits in different types of provider-operated networks (Frame Relay, ATM, MPLS, . . .)

Message switching:

- No practical use on layers 2 and 3
- But: message switching exists from the perspective of higher layers, e. g. message-based transport protocols based on UDP or application layer protocols such as SMTP (Simple Mail Transfer Protocol)

Packet switching:

- Prevalent in most modern communication networks
- Almost replaced dedicated telephone connections (including mobile communication)
- Digital radio and TV
- Many peripheral interfaces in computers (e. g. PCI, USB, Thunderbolt, etc.)

Addressing on layer 3

The data link layer (layer 2) offers

- more or less fair media access by multiple hosts,
- a “sufficiently” good protection from transmission errors, and
- addressing within a local area network.

The network layer (layer 3) supplements this by

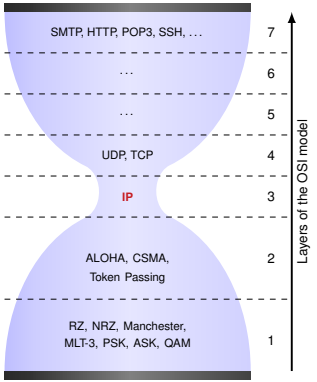
- globally unique addressing and structured / logical addressing as well as
- procedures to determine (near) optimal paths.

We limit the discussion in this section to

- IPv4 (Internet Protocol v4, 1981) and
- its successor IPv6 (1998).

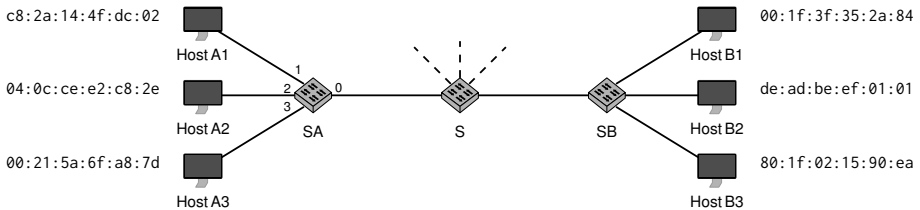
Examples for alternative protocols on the network layer are the (meanwhile obsolete) protocols

- IPX (Internetwork Packet Exchange, 1990),
- DECnet Phase 5 (1987), and
- AppleTalk (1983).



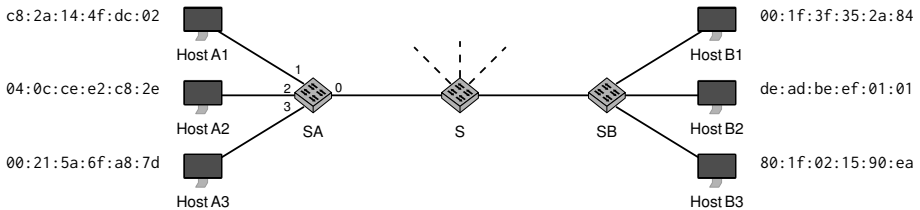
Internet Protocol version 4 (IPv4) [14]

We consider the following network as an example, which we assume to be based on an up to date version of Ethernet:



How many entries does the switching table of SA contain?

We consider the following network as an example, which we assume to be based on an up to date version of Ethernet:



How many entries does the switching table of SA contain?

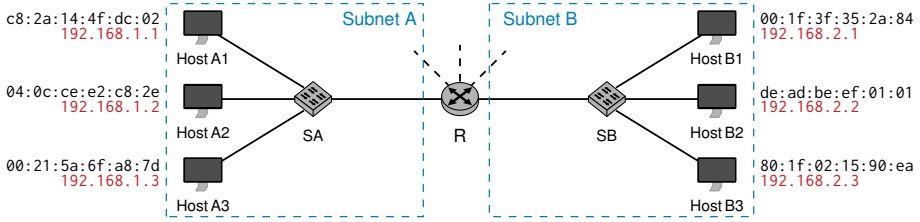
- Exactly one for each known host
- Most of those addresses are mapped to port 0.
- A summary of entries is in general not possible since MAC addresses do not reflect the location of a node within the network.

Port	MAC
0	00:1f:3f:35:2a:84
0	de:ad:be:ef:01:01
0	80:1f:02:15:90:ea
1	c8:2a:14:4f:dc:02
2	04:0c:ce:e2:c8:2e
3	00:21:5a:6f:a8:7d
⋮	⋮

$\Rightarrow$ It seems reasonable to abstract from physical addresses

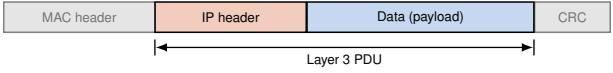
Internet Protocol version 4 (IPv4) [14]

We consider the example network with one router (R) in between the two local area networks:



- One IP address is assigned to each host.
- Each IP address is represented by four Bytes as decimal number separated by dots (“dotted decimal notation”).
- In this example, the 4th octet identifies a host within a specific network.
- The first three octets identify the network itself wherein a host is located.
- The router R makes forwarding decisions based on the destination IP address.

⇒ Every packet must contain a source and destination IP address in its header:



Internet Protocol version 4 (IPv4) [14]

IPv4 header

- The IP header not only contains source and destination address.
- The use of the most important fields will be explained in more detail with examples later in this chapter.

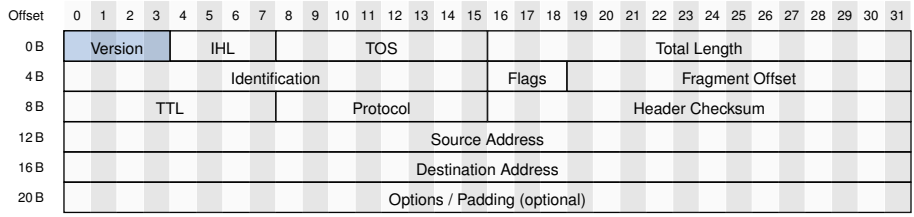


Figure 1: IPv4 header (minimal length: 20 B)

Version

- Differentiates between IPv4 and IPv6 in case we know that it is IP at all.
- Valid values are 4 (IPv4) and 6 (IPv6).

Internet Protocol version 4 (IPv4) [14]

IPv4 header

- The IP header not only contains source and destination address.
- The use of the most important fields will be explained in more detail with examples later in this chapter.

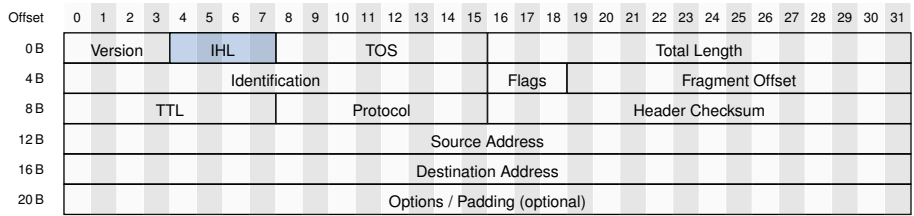


Figure 1: IPv4 header (minimal length: 20 B)

IHL (IP header length)

- Specifies the length of the IP header including all options in multiples of 32 bit.
- Important since the IPv4 header may have variable length due to options.

Internet Protocol version 4 (IPv4) [14]

IPv4 header

- The IP header not only contains source and destination address.
- The use of the most important fields will be explained in more detail with examples later in this chapter.

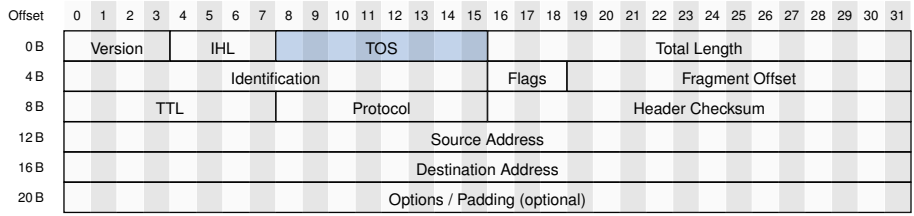


Figure 1: IPv4 header (minimal length: 20 B)

TOS (type of service)

- Used to classify and prioritize packets, e. g. to give precedence to real time transfers such as voice data.
- Possibility of congestion control on layer 3 (Explicit Congestion Notification).

Internet Protocol version 4 (IPv4) [14]

IPv4 header

- The IP header not only contains source and destination address.
- The use of the most important fields will be explained in more detail with examples later in this chapter.

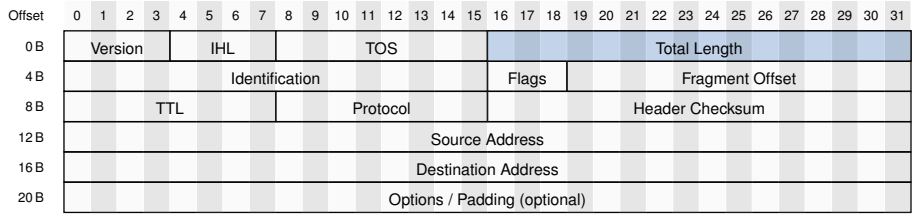


Figure 1: IPv4 header (minimal length: 20 B)

Total length

- Identifies the total length of the packet (header + payload) in Bytes.
- The maximal length of an IP packet is therefore limited to 65 535 B.
- The source may adapt the size if needed to avoid fragmentation.
- The maximum packet length such that no fragmentation is needed on the current link is known as Maximum Transmission Unit (MTU). It depends on layers 2/1 and is 1500 B for Ethernet (without support for “jumbo frames”).

Internet Protocol version 4 (IPv4) [14]

IPv4 header

- The IP header not only contains source and destination address.
- The use of the most important fields will be explained in more detail with examples later in this chapter.

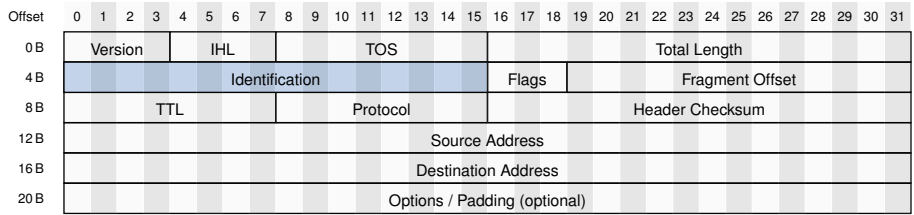


Figure 1: IPv4 header (minimal length: 20 B)

Identification

- Randomly chosen 16 bit value for each packet.
- Used to identify fragments belonging to the same packet.

Internet Protocol version 4 (IPv4) [14]

IPv4 header

- The IP header not only contains source and destination address.
- The use of the most important fields will be explained in more detail with examples later in this chapter.

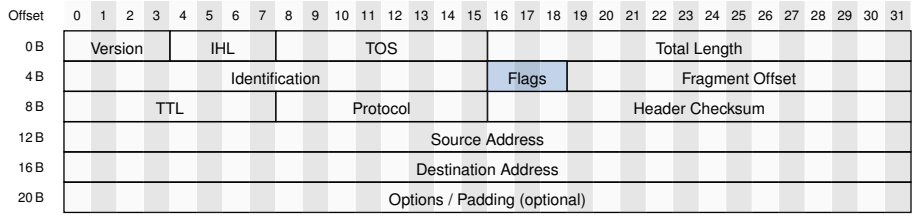


Figure 1: IPv4 header (minimal length: 20 B)

Flags

- Bit 16: reserved and set to 0.
- Bit 17: don't fragment (DF). If set to 1, the packet must not be fragmented.
- Bit 18: more fragments (MF). Specifies whether further fragments follow (1) or whether this packet is the last fragment (0). If a packet was not fragmented, the value is also set to 0.

Internet Protocol version 4 (IPv4) [14]

IPv4 header

- The IP header not only contains source and destination address.
- The use of the most important fields will be explained in more detail with examples later in this chapter.

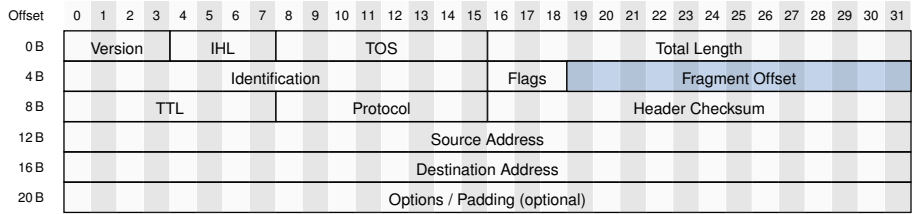


Figure 1: IPv4 header (minimal length: 20 B)

Fragment offset

- Specifies the absolute position of data contained in this fragment with respect to the original packet in multiples of 8 B.
- Together with the identifier and the MF bit, this allows to reassemble fragments in the correct order.

Internet Protocol version 4 (IPv4) [14]

IPv4 header

- The IP header not only contains source and destination address.
- The use of the most important fields will be explained in more detail with examples later in this chapter.

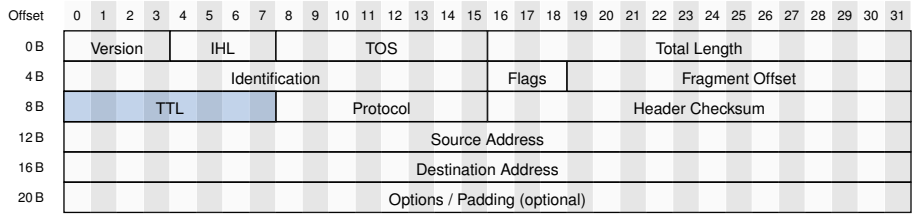


Figure 1: IPv4 header (minimal length: 20 B)

TTL (Time to live)

- If a router forwards an IP packet, the TTL is decremented by 1.
- When the TTL reaches 0, the router drops the packet and sends an error message to the source node (ICMP Time Exceeded in Transit → more on that later).
- This allows to limit the path length in the internet and prevents packets from circulating infinitely due to routing loops.

Internet Protocol version 4 (IPv4) [14]

IPv4 header

- The IP header not only contains source and destination address.
- The use of the most important fields will be explained in more detail with examples later in this chapter.

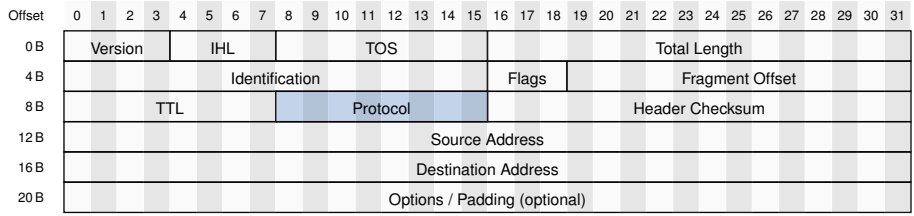


Figure 1: IPv4 header (minimal length: 20 B)

Protocol

- Identifies the protocol on layer 4 that is used in the packet's payload.
- Among others, important for the operating system to assign received packets to the correct protocol stack.
- Valid values are, for instance, `0x06` (TCP) and `0x11` (UDP).
- cmp. Ethertype, which identifies the protocol used in the payload of a frame.

Internet Protocol version 4 (IPv4) [14]

IPv4 header

- The IP header not only contains source and destination address.
- The use of the most important fields will be explained in more detail with examples later in this chapter.

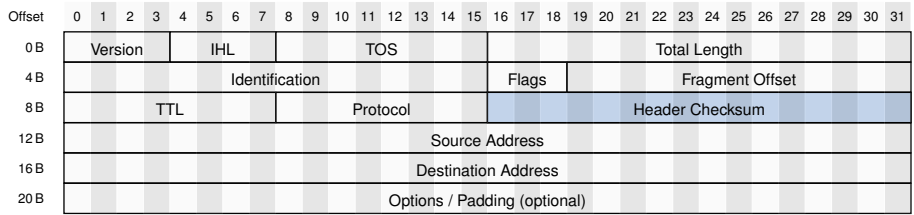


Figure 1: IPv4 header (minimal length: 20 B)

Header Checksum

- Simple checksum optimized for efficient calculation that protects only the IP header (without data).
- The checksum is designed such that decrementing the TTL field corresponds to a decrement of the checksum. Therefore, it is not necessary to re-calculate the checksum when packets are forwarded.
- The checksum is only for error detection, not correction.

Internet Protocol version 4 (IPv4) [14]

IPv4 header

- The IP header not only contains source and destination address.
- The use of the most important fields will be explained in more detail with examples later in this chapter.

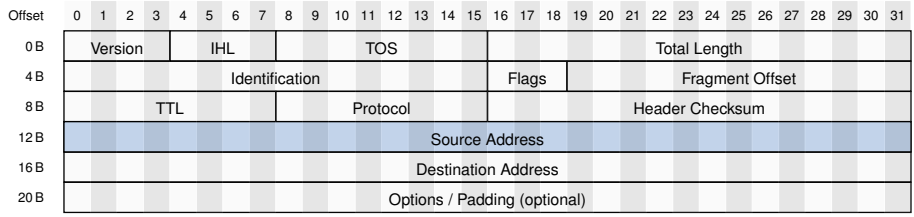


Figure 1: IPv4 header (minimal length: 20 B)

Source Address

- IP address of the source node.

Internet Protocol version 4 (IPv4) [14]

IPv4 header

- The IP header not only contains source and destination address.
- The use of the most important fields will be explained in more detail with examples later in this chapter.

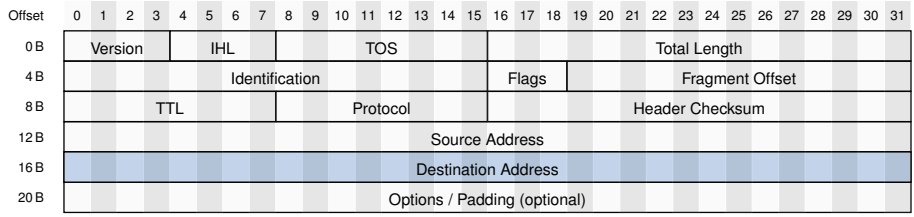


Figure 1: IPv4 header (minimal length: 20 B)

Destination Address

- IP address of the destination node.

Internet Protocol version 4 (IPv4) [14]

IPv4 header

- The IP header not only contains source and destination address.
- The use of the most important fields will be explained in more detail with examples later in this chapter.

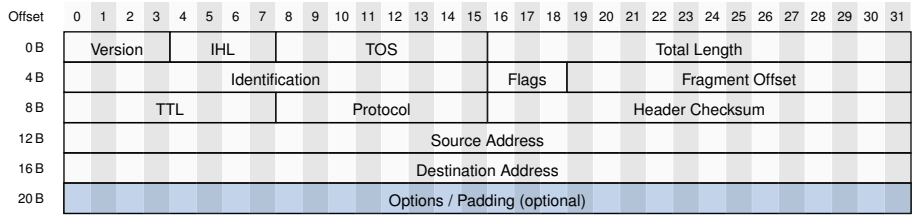


Figure 1: IPv4 header (minimal length: 20 B)

Options / padding

- IPv4 supports different options, e. g. timestamps, route recording, . . . , which are appended as optional fields to the IP header.
- Not all of these options are 4 B long. Since the length of the IP header must be a multiple of 4 B, options of different size must potentially be padded to the next larger valid length.

Remark regarding host byte order and network byte order

There are two different byte orders:

1. Big endian: “lowest order byte is located at the highest address”
   Intuitive order in memory, e. g. the decimal number 256 in hexadecimal writing as `0x0100`.
2. Little endian: “lowest order byte is located at the lowest address”
   Counterintuitive since data in memory is laid out “the wrong way”, e. g. the decimal number 256 in hexadecimal writing as `0x0001`.

Networking protocols need to allow for communication between hosts using different byte orders. Therefore, the network byte order is defined, which is big endian.

Most computers today, however, internally use little endian as “host byte order”. Therefore, all multi-byte values (commonly 16 bit, 32 bit, and 64 bit values) must be converted into network byte order and back to host byte order at the destination.

Network byte order

Before transmitting, data must be converted from host byte order to network byte order, and vice versa when receiving data. In case the host byte order is the same as the network byte order, the corresponding functions are translated to a NoOp by compilers.

Note: IEEE 802.11 is defined as little endian (thanks for nothing!).

Converting from host byte order to network byte order

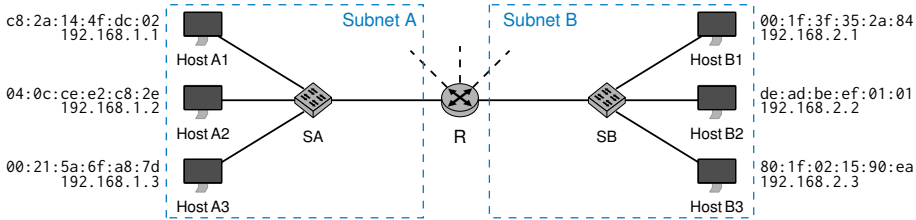
- This applies to fields such as total length, identification, flags + fragment offset, and header checksum from the IP header (all multi-octet values).
- Single-octet values are not affected.

In C, the conversion between 16 B values is done using the macros `htons()` and `ntohs`, respectively:

- `htons(0x0001) = 0x0100` “host to network short”
- `ntohs(0x0100) = 0x0001` “network to host short”

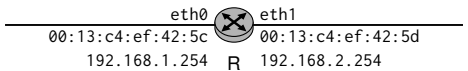
There are similar macros `htonl()` and `ntohl()` for 32 bit values (“`l`” the data type `long`).

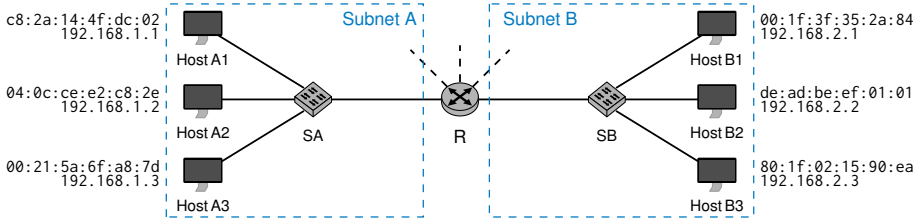
Back to our example network:



- Within subnet A all hosts can be reached directly by using SA.
- When A1 wants to send a packet to B2, the packet must be routed:
  - Forwarding the packet is done by R based on IP addresses
  - A1 has to make sure that R receives the packet
  - It therefore uses the MAC address of R's left-hand interface as destination MAC address in the Ethernet frame containing the packet, and the IP address of B2 as destination address in the IP header.

⇒ R must be addressable and therefore has different MAC and IP addresses on both interfaces:





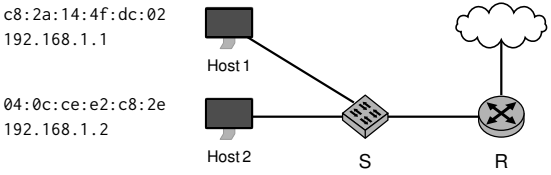
Open questions:

- Assume that the sender only knows the IP address of the destination.
  How can the MAC address of the next hop be determined? (address resolution)
- How does A1 know that it can reach subnet B via R? (routing table)
- Assume that the destination of a packet is not in a network directly connected to R. How does R know the correct direction? (More precisely: the correct next hop?) (routing table)
- How is this routing table populated? (static routing, routing protocols)
- What happens if R knows about multiple paths to the destination? (longest prefix matching)
- How does the separation of an IP address into network and host part work? (classful routing, classless routing, subnetting)
- How does the sender even know the IP address of the target? (DNS → layer 7)

Internet Protocol version 4 (IPv4) [14]

Address resolution [15]

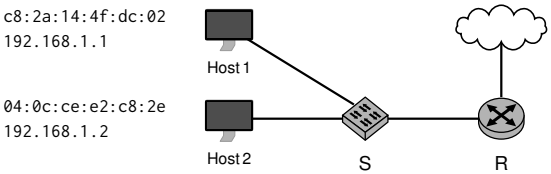
- Host 1 wants to send a packet to Host 2
- The IP address of host 2 (192.168.1.2) is known
- How can host 1 obtain the corresponding MAC address?



Internet Protocol version 4 (IPv4) [14]

Address resolution [15]

- Host 1 wants to send a packet to Host 2
- The IP address of host 2 (192.168.1.2) is known
- How can host 1 obtain the corresponding MAC address?



Address Resolution Protocol (ARP)

1. Host 1 sends an ARP request: “Who has 192.168.1.2? Tell 192.168.1.1 at `c8:2a:14:4f:dc:02`”
2. Host 2 answers with an ARP reply: “192.168.1.2 is at `04:0c:ce:e2:c8:2e`”

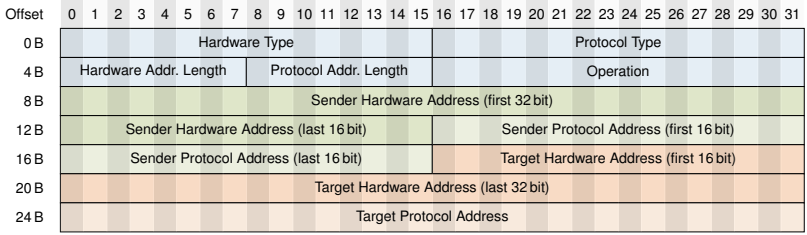
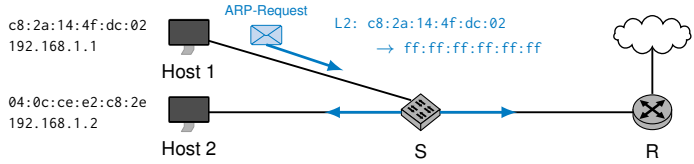


Figure 2: ARP packet (note that it is not specific to Ethernet)

Example: (`L2: xx:xx:xx:xx:xx:xx` → `yy:yy:yy:yy:yy:yy` represents source and destination MAC addresses)

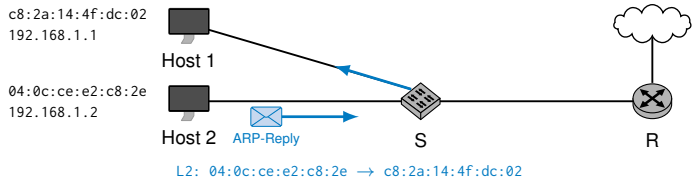


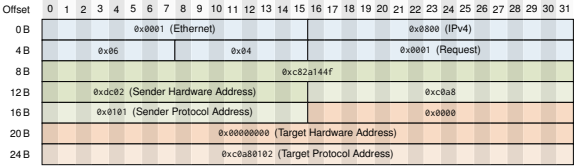
Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0 B	0x0001 (Ethernet)																0x0800 (IPv4)															
4 B	0x06								0x04								0x0001 (Request)															
8 B	0xc82a144f																															
12 B	0xdc02 (Sender Hardware Address)																0xc0a8															
16 B	0x0101 (Sender Protocol Address)																0x0000															
20 B	0x00000000 (Target Hardware Address)																															
24 B	0xc0a80102 (Target Protocol Address)																															

(a) ARP Request

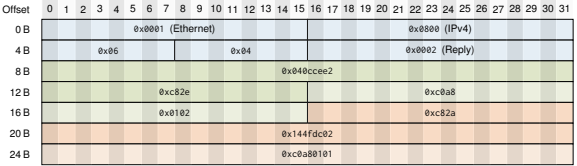
- The ARP request is directed to the broadcast address `ff:ff:ff:ff:ff:ff`, which is why the switch forwards it on all outbound ports.

Example: (`L2: xx:xx:xx:xx:xx:xx` → `yy:yy:yy:yy:yy:yy` represents source and destination MAC addresses)





(c) ARP Request



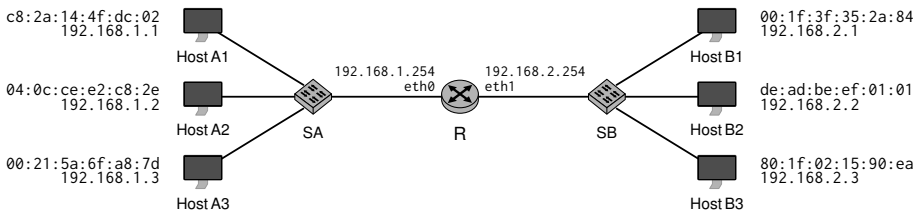
(d) ARP Reply

- The ARP request is directed to the broadcast address `ff:ff:ff:ff:ff:ff`, which is why the switch forwards it on all outbound ports.
- The ARP reply is usually sent as unicast to the requesting host.
- The roles sender / target are swapped between request and reply (cmp. content of the green and orange fields)

Internet Protocol version 4 (IPv4) [14]

What happens if the destination is not in the same subnet (e. g. A1 to B2)?

- Each host should know about a router, the so called default gateway, to which it can send packets destined for a node in a different subnet and for which it has no more specific entry in its routing table.
- Whether a destination address belongs to the local or a remote network can be determined by comparing the destination address with the local network address.
- At the moment we assume that the first 3 octets of an IP address identify the network
  $\Rightarrow$ 192.168.1.1 and 192.168.2.2 are located in different networks.

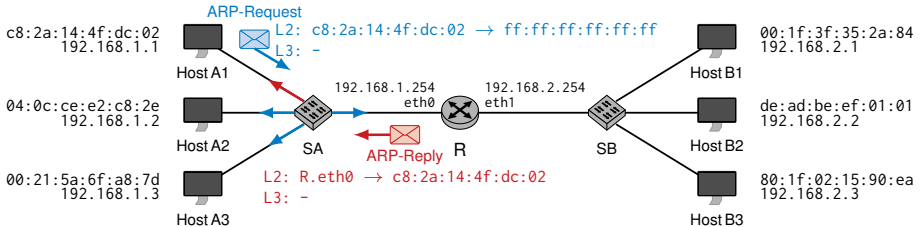


1. A1 recognizes that 192.168.2.2 is not in the local network. Its default gateway is 192.168.1.254.

Internet Protocol version 4 (IPv4) [14]

What happens if the destination is not in the same subnet (e. g. A1 to B2)?

- Each host should know about a router, the so called default gateway, to which it can send packets destined for a node in a different subnet and for which it has no more specific entry in its routing table.
- Whether a destination address belongs to the local or a remote network can be determined by comparing the destination address with the local network address.
- At the moment we assume that the first 3 octets of an IP address identify the network
  $\Rightarrow$ 192.168.1.1 and 192.168.2.2 are located in different networks.

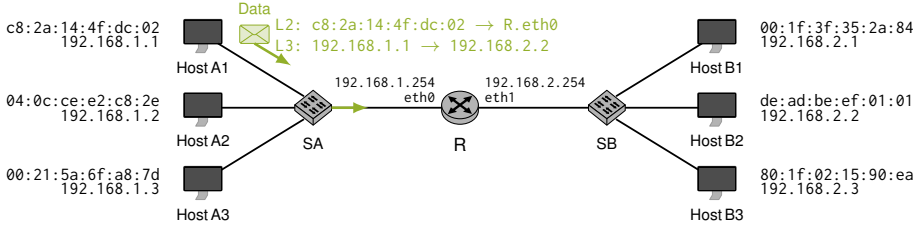


1. A1 recognizes that 192.168.2.2 is not in the local network. Its default gateway is 192.168.1.254.
2. A1 resolves the MAC address that belongs to 192.168.1.254 using ARP.

Internet Protocol version 4 (IPv4) [14]

What happens if the destination is not in the same subnet (e. g. A1 to B2)?

- Each host should know about a router, the so called default gateway, to which it can send packets destined for a node in a different subnet and for which it has no more specific entry in its routing table.
- Whether a destination address belongs to the local or a remote network can be determined by comparing the destination address with the local network address.
- At the moment we assume that the first 3 octets of an IP address identify the network
  $\Rightarrow$ 192.168.1.1 and 192.168.2.2 are located in different networks.

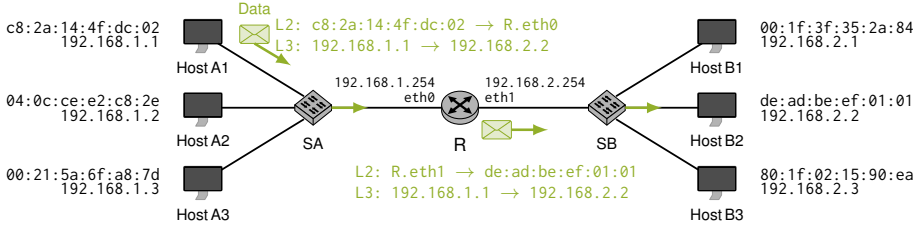


1. A1 recognizes that 192.168.2.2 is not in the local network. Its default gateway is 192.168.1.254.
2. A1 resolves the MAC address that belongs to 192.168.1.254 using ARP.
3. A1 then sends a frame to R, which is addressed using its layer 2 address. The destination IP address (layer 3) is used to address B2.

Internet Protocol version 4 (IPv4) [14]

What happens if the destination is not in the same subnet (e. g. A1 to B2)?

- Each host should know about a router, the so called default gateway, to which it can send packets destined for a node in a different subnet and for which it has no more specific entry in its routing table.
- Whether a destination address belongs to the local or a remote network can be determined by comparing the destination address with the local network address.
- At the moment we assume that the first 3 octets of an IP address identify the network
  $\Rightarrow$ 192.168.1.1 and 192.168.2.2 are located in different networks.



1. A1 recognizes that 192.168.2.2 is not in the local network. Its default gateway is 192.168.1.254.
2. A1 resolves the MAC address that belongs to 192.168.1.254 using ARP.
3. A1 then sends a frame to R, which is addressed using its layer 2 address. The destination IP address (layer 3) is used to address B2.
4. R accepts the frame, determines the outbound interface using the destination IP address, and forwards the packet to B2. To that end, R addresses B2 using its MAC address, which may require another address resolution.

Note

- MAC addresses are used to address the next hop within a local area network and change from hop to hop.
- IP addresses are used to determine whether the destination is within the same local area network and to address the destination. They do not change from hop to hop.

Remarks

- The result of an address resolution is cached in the ARP table of a node to avoid having to perform address resolution again for each packet to be sent.
- Entries in the ARP table age and are purged by the operating system after some time (5 min to 10 min).
- The content of the ARP table can be displayed under Linux, macOS, and Windows with the command `arp -a`.
- ARP replies may also be sent as broadcast such that all hosts within a broadcast domain receive the reply (“unsolicited ARP reply”).

Questions: What would happen if . . .

- two hosts within the same broadcast domain have identical MAC addresses but different IPs?
- one host deliberately answers ARP requests not destined for him?
- one host sends ARP replies as broadcast with random MAC address?

Summary: How does a host or router behave when it should send an IPv4 packet, given its destination IP address?

1. Determine the IP address of the next hop
   - Destination IP address is not in the local network: IPv4 address of the next gateway
     → default gateway for hosts or next hop (see static and dynamic routing)
   - Destination IP address is in the local network: Destination IP
2. Resolve the MAC address that belongs to the next hops IP address
   Perform a lookup in the local ARP table
   - hit: use the MAC address from the table
   - miss: send an ARP request to the broadcast address `ff:ff:ff:ff:ff:ff` and use the MAC address contained in the ARP reply
3. Create a data frame with
   - the outgoing interfaces MAC address as source,
   - the next hops mac address as destination (hop to hop),
   - the EtherType for IPv4 (`0x0800`) and
   - the IPv4 packet as payload.

Internet Protocol version 4 (IPv4) [14]

Internet Control Message Protocol (ICMP) [16]

While forwarding a packet from source to destination, there may occur errors, e. g.

- a packet gets into a routing loop,
- a router does not have a valid route to the destination,
- the last router cannot resolve the MAC address of the destination (e. g. the destination is shut down) . . .

The Internet Control Message Protocol (ICMP) is used to

- inform the source about such problems and
- offers methods to
  - check whether the destination is reachable (ping) or
  - to re-route packets (redirect).

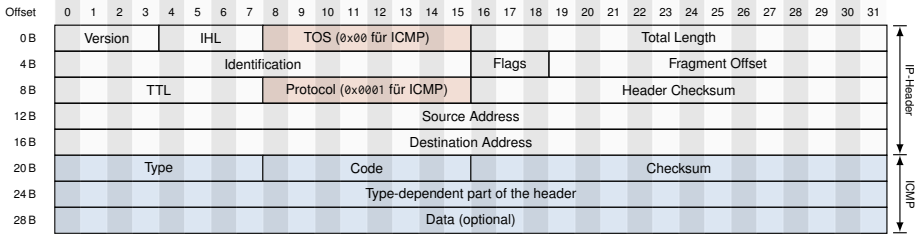
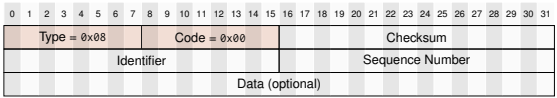
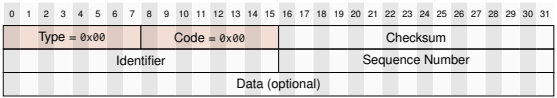


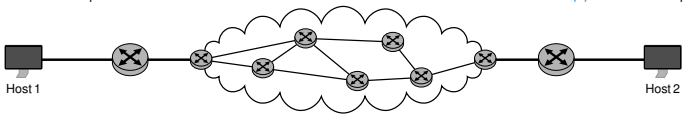
Figure 3: Generic ICMP header with prepended IP header



(a) ICMP Echo Request



(b) ICMP Echo Reply



„Ping“ from Host 1 to 2:

- Host 1 chooses a random identifier (16 bit), the sequence number is incremented by 1 per echo request.
- The echo request is forwarded by routers just like any other packet.
- When Host 2 receives the echo request, it answers with an echo reply. The identifier, sequence number, and data from the request is thereby copied and sent back to Host 1.
- If forwarding the request to Host 2 fails, an ICMP packet with a cooresponding error code is sent back to Host 1.

Question: What is the identifier used for?

Internet Control Message Protocol (ICMP) [16] – ICMP Time Exceeded

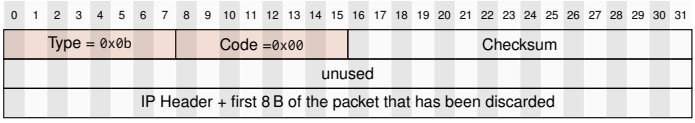
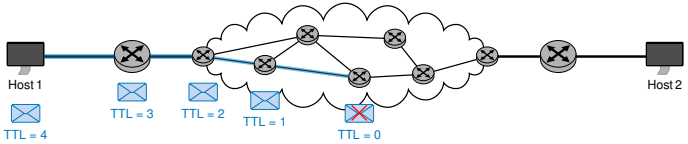


Figure 4: ICMP TTL-Exceeded



- The IP header has a TTL field, which is decremented by routers when the packet is forwarded.
- If it reaches 0, the packet is dropped.

Internet Control Message Protocol (ICMP) [16] – ICMP Time Exceeded

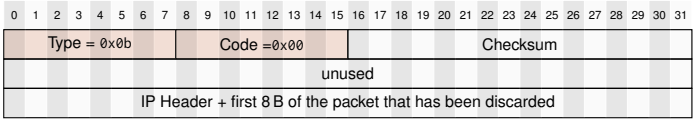
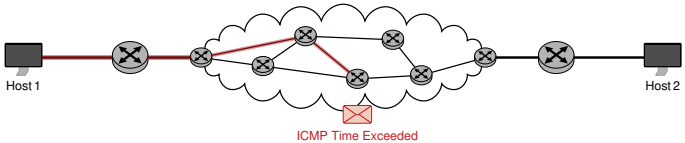


Figure 4: ICMP TTL-Exceeded



- The IP header has a TTL field, which is decremented by routers when the packet is forwarded.
- If it reaches 0, the packet is dropped.
- The router then generates an ICMP Time Exceeded and sends it back to the source of the dropped packet.

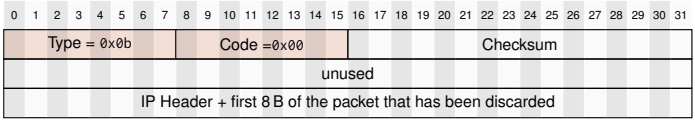
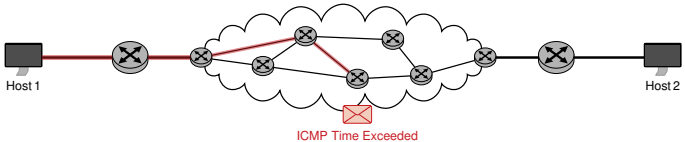


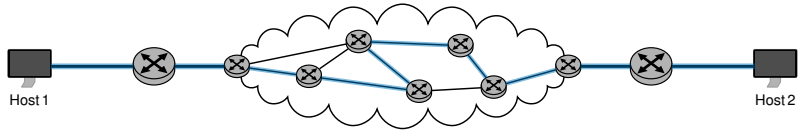
Figure 4: ICMP TTL-Exceeded



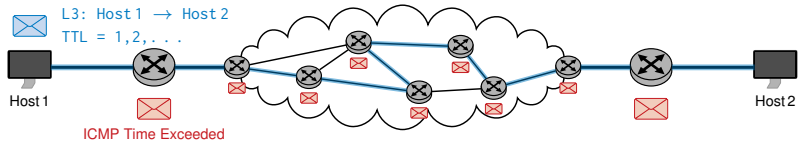
- The IP header has a TTL field, which is decremented by routers when the packet is forwarded.
- If it reaches 0, the packet is dropped.
- The router then generates an ICMP Time Exceeded and sends it back to the source of the dropped packet.

Since the header and the first 8 B of the dropped packet's payload are returned to the sender, the sender can determine exactly which packet was dropped.

Question: What information is contained in these first 8 B if the dropped packet was an ICMP echo request?



- Although packets may be forwarded along different paths (in both directions), in practice, usually only one or very few are used.
- Which path is taken between Hosts 1 and 2?



- Although packets may be forwarded along different paths (in both directions), in practice, usually only one or very few are used.
- Which path is taken between Hosts 1 and 2?

Traceroute: Host 1 sends e. g. ICMP Echo Requests to Host 2

- where the TTL is set to 1 in the beginning and
- afterwards increased step by step.

⇒ Routers along the path from Host 1 to Host 2 will drop packets and answer with TTL Exceeded messages. Based on the IP source addresses of those error messages Host 1 can trace the path to Host 2.

Example:

```
moepi$ traceroute -n www.net.in.tum.de
traceroute to www.net.in.tum.de (131.159.15.49), 64 hops max, 52 byte packets
 1  192.168.1.1  2.570 ms  1.808 ms  2.396 ms
 2  82.135.16.28  15.036 ms  14.359 ms  13.760 ms
 3  212.18.7.189  14.118 ms  13.801 ms  13.845 ms
 4  82.135.16.102  20.062 ms  20.137 ms  20.251 ms
 5  80.81.192.222  22.707 ms  23.049 ms  23.215 ms
 6  188.1.144.142  31.068 ms  36.542 ms  30.823 ms
 7  188.1.37.90  30.815 ms  30.671 ms  30.808 ms
 8  129.187.0.150  30.272 ms  30.602 ms  30.845 ms
 9  131.159.252.1  30.885 ms  30.551 ms  30.992 ms
10  131.159.252.150  30.886 ms  30.955 ms  30.621 ms
11  131.159.252.150  30.578 ms  30.699 ms *
```

Questions and open problems:

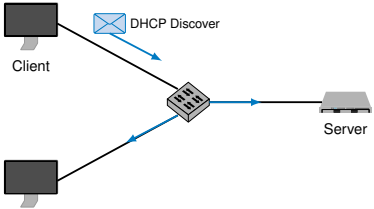
- Routers usually have multiple IP addresses. Which IP is used as source for error messages? Is always the same address used?
- What if there are actually multiple paths or path sections in use at the same time?
- Are routers required at all to send error messages?
- Assume the path from $A \rightarrow B$ is symmetric. Why will the outputs of traceroute still differ?

Internet Protocol version 4 (IPv4) [14]

Dynamic Host Configuration Protocol (DHCP) [16]

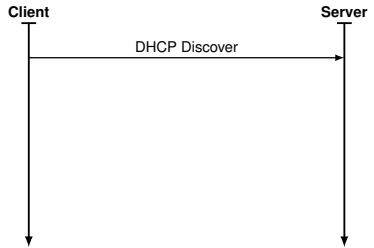
Where do hosts actually get their IP address from?

- Static configuration by hand
- Randomly self-assigned using Automatic Private IP Addressing (APIPA)[1]
- Dynamically assigned by a DHCP server



Procedure:

1. A client transmits a DHCP Discover (layer 2 broadcast).



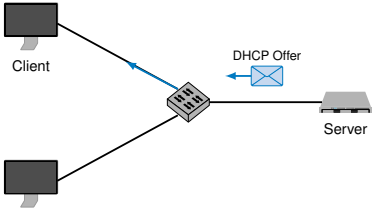
[1] A computer randomly chooses an IP address from the reserved range for APIPA. Before using the address, the computer sends and ARP request for that IP address with the request address set to 0.0.0.0 (unspecified address). If a reply is received, the address is already in use.

Internet Protocol version 4 (IPv4) [14]

Dynamic Host Configuration Protocol (DHCP) [16]

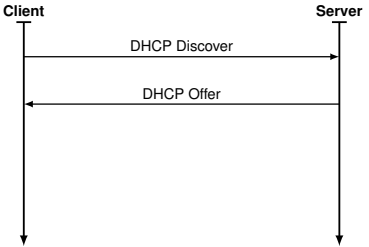
Where do hosts actually get their IP address from?

- Static configuration by hand
- Randomly self-assigned using Automatic Private IP Addressing (APIPA)[1]
- Dynamically assigned by a DHCP server



Procedure:

1. A client transmits a DHCP Discover (layer 2 broadcast).
2. The DHCP server answers with a DHCP Offer, whereby it offers the client an IP address.



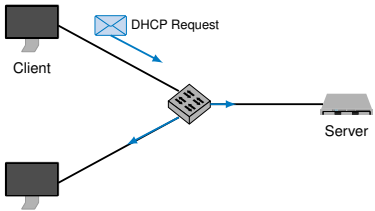
[1] A computer randomly chooses an IP address from the reserved range for APIPA. Before using the address, the computer sends and ARP request for that IP address with the request address set to 0.0.0.0 (unspecified address). If a reply is received, the address is already in use.

Internet Protocol version 4 (IPv4) [14]

Dynamic Host Configuration Protocol (DHCP) [16]

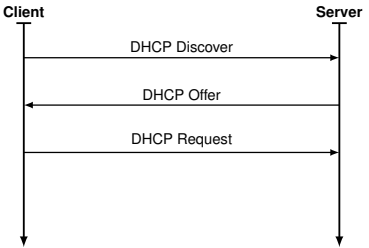
Where do hosts actually get their IP address from?

- Static configuration by hand
- Randomly self-assigned using Automatic Private IP Addressing (APIPA)[1]
- Dynamically assigned by a DHCP server



Procedure:

1. A client transmits a DHCP Discover (layer 2 broadcast).
2. The DHCP server answers with a DHCP Offer, whereby it offers the client an IP address.
3. The client answers with a DHCP Request, whereby it requests the offered address.



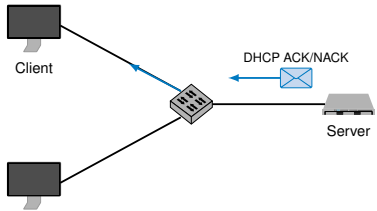
[1] A computer randomly chooses an IP address from the reserved range for APIPA. Before using the address, the computer sends and ARP request for that IP address with the request address set to 0.0.0.0 (unspecified address). If a reply is received, the address is already in use.

Internet Protocol version 4 (IPv4) [14]

Dynamic Host Configuration Protocol (DHCP) [16]

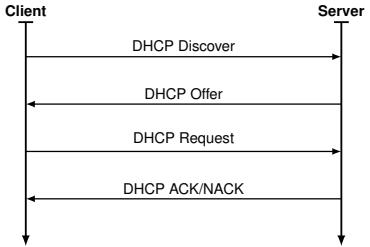
Where do hosts actually get their IP address from?

- Static configuration by hand
- Randomly self-assigned using Automatic Private IP Addressing (APIPA)[1]
- Dynamically assigned by a DHCP server



Procedure:

1. A client transmits a DHCP Discover (layer 2 broadcast).
2. The DHCP server answers with a DHCP Offer, whereby it offers the client an IP address.
3. The client answers with a DHCP Request, whereby it requests the offered address.
4. The DHCP server answers with a DHCP ACK or NACK, whereby it either assigns the requested address to the client or prohibits its usage, e. g. if the address in question has been assigned to another client in the meantime.



[1] A computer randomly chooses an IP address from the reserved range for APIPA. Before using the address, the computer sends and ARP request for that IP address with the request address set to 0.0.0.0 (unspecified address). If a reply is received, the address is already in use.

Remarks

- The address assigned by a DHCP server is called lease and its validity is limited in time.
- Clients renew their leases in regular intervals.
- Especially in smaller (private) networks, a router often takes over the role of the DHCP server.
- Together with the lease (containing IP and subnet mask), DHCP servers may deliver additional options:
  - DNS resolver for name resolution (→ Chapter 5)
  - Hostname and domain suffix (→ Chapter 5)
  - Static routes, in particular a default gateway
  - Maximum Transmission Unit
  - NTP server for time synchronisation
  - ...
- For redundancy reasons, sometimes multiple DHCP servers are used per network, where
  - the address ranges of the two servers must not overlap or
  - additional mechanisms for synchronizing the address pools are necessary.
- A DHCP server accidentally introduced into the network (e. g. by a carelessly connected router or access point) can have a significant impact on the network and is often difficult to find:
  - Once there was such a rogue DHCP server in the network of some chair, which could not be found despite intensive search.
  - Via the MAC addresses of the DHCP messages coming by the rogue server, it turned out in the end that it was a **Nokia smartphone** on WiFi!
  - The number of suspects had thus been greatly reduced (;

Address classes (classful routing)

At the beginning of the chapter we had described in an example that

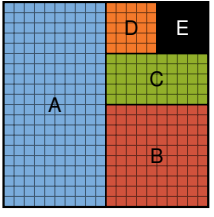
- the first three octets of an IP address identify the network and
- the fourth octet addresses hosts within a network.

This separation was an example, corresponding to address class C. Historically, the IP address space is divided into the following five classes:

Class	1st octed	First address	Last address	Net / host portion	Number of networks	Addresses per network
A	`0xxxxxxx`	0.0.0.0	127.255.255.255	N.H.H.H	2^7 = 128	2^{24} = 16777216
B	`10xxxxxx`	128.0.0.0	191.255.255.255	N.N.H.H	2^{14} = 16384	2^{16} = 65536
C	`110xxxxx`	192.0.0.0	223.255.255.255	N.N.N.H	2^{21} = 2097152	2^8 = 256
D	`1110xxxx`	224.0.0.0	239.255.255.255	*reserved for multicast*		
E	`1111xxxx`	240.0.0.0	255.255.255.255	*reserved*		

Graphical representation:

- IP address space as square
- Colored areas mark the five address classes
- The individual class A networks indicated by the grid



→ The next slides show the actual distribution and utilization of the address space [2] (same color coding).

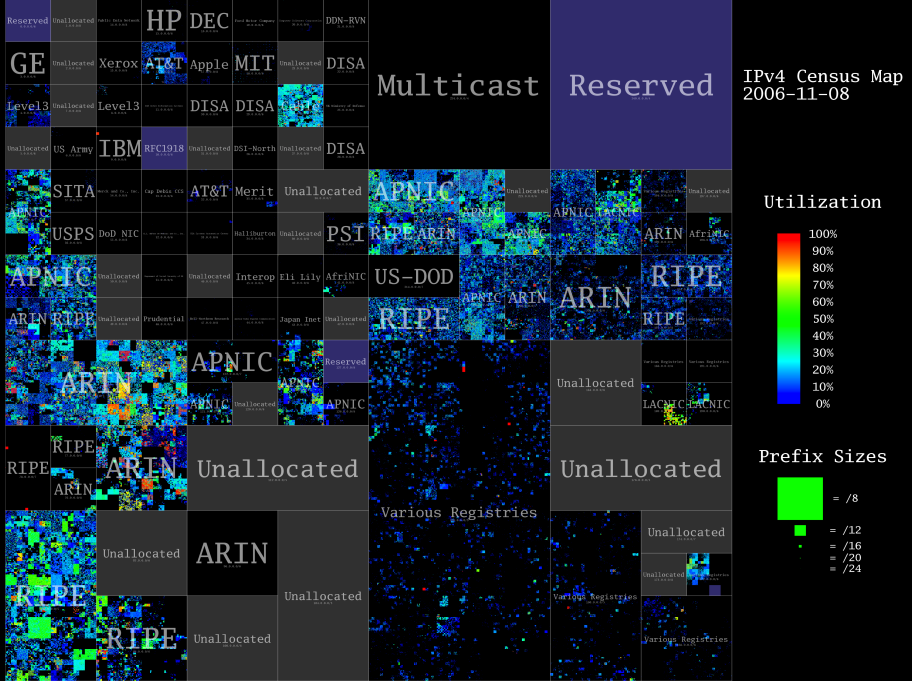
IPv4 Census Map
2006-11-08

Utilization

100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%

Prefix Sizes

= /8
= /12
= /16
= /20
= /24



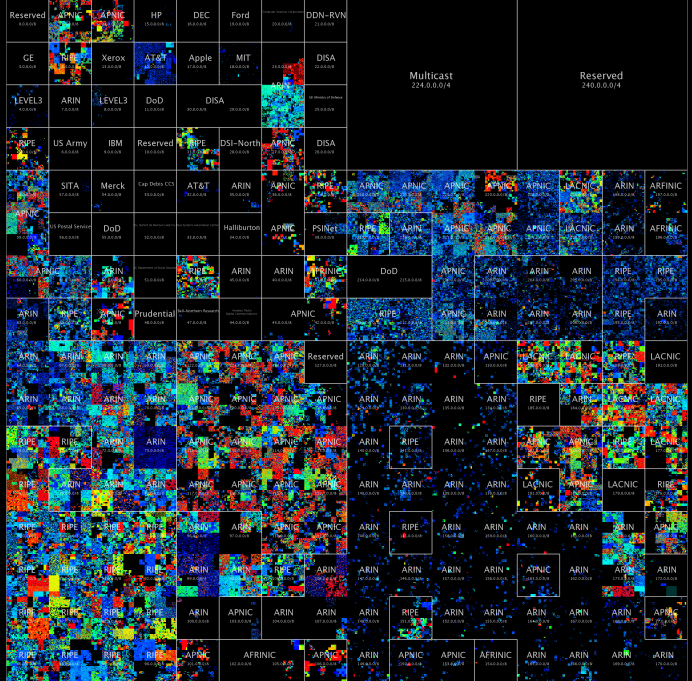
IPv4 Census Map
2006-11-08

Utilization

100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%

Prefix Sizes

= /8
= /12
= /16
= /20
= /24


Reserved
APNIC
APNIC
HP
DEC
Ford
DDN-RVN
GE
RIPE
Xerox
AT&T
Apple
MIT
DISA
Multicast
224.0.0.0/4
Reserved
240.0.0.0/4
LEVEL3
ARIN
LEVEL3
DoD
DISA
RIPE
US Army
IBM
Reserved
RIPE
DSI-North
APNIC
DISA
SITA
Merck
Cap Debis CCS
AT&T
ARIN
APNIC
RIPE
APNIC
APNIC
APNIC
APNIC
APNIC
LACNIC
ARIN
AFRINIC
US Postal Service
DoD
Halliburton
APNIC
PSINet
RIPE
ARIN
APNIC
APNIC
APNIC
LACNIC
ARIN
AFRINIC
APNIC
ARIN
RIPE
ARIN
APNIC
DoD
APNIC
ARIN
ARIN
ARIN
RIPE
RIPE
ARIN
RIPE
APNIC
Prudential
APNIC
RIPE
APNIC
ARIN
ARIN
ARIN
RIPE
ARIN
ARIN
ARIN
ARIN
ARIN
APNIC
APNIC
APNIC
Reserved
ARIN
ARIN
ARIN
APNIC
LACNIC
APNIC
RIPE
LACNIC
ARIN
ARIN
ARIN
ARIN
APNIC
APNIC
APNIC
ARIN
ARIN
ARIN
ARIN
RIPE
ARIN
LACNIC
LACNIC
RIPE
RIPE
ARIN
ARIN
APNIC
APNIC
APNIC
APNIC
ARIN
RIPE
ARIN
ARIN
APNIC
APNIC
RIPE
LACNIC
RIPE
RIPE
ARIN
ARIN
APNIC
APNIC
APNIC
ARIN
ARIN
ARIN
ARIN
ARIN
LACNIC
APNIC
LACNIC
RIPE
RIPE
RIPE
RIPE
ARIN
ARIN
APNIC
APNIC
ARIN
RIPE
ARIN
ARIN
ARIN
ARIN
ARIN
APNIC
RIPE
RIPE
RIPE
RIPE
ARIN
ARIN
RIPE
ARIN
ARIN
ARIN
ARIN
APNIC
ARIN
ARIN
ARIN
ARIN
RIPE
RIPE
RIPE
RIPE
ARIN
APNIC
ARIN
ARIN
ARIN
RIPE
ARIN
ARIN
ARIN
ARIN
ARIN
APNIC
RIPE
RIPE
RIPE
RIPE
APNIC
AFRINIC
APNIC
ARIN
APNIC
APNIC
AFRINIC
ARIN
ARIN
ARIN
ARIN
IPv4 Census Map
June - October 2012
Utilization
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
Prefix Sizes
= /8
= /12
= /16
= /20
= /24
420 Million hosts that responded to ICMP Ping at least 2 times between June and October 2012
Source: Carna Botnet

Address space wastage

- When IPv4 was introduced in 1981, it was not imagined that $\sim 2^{32}$ addresses divided into classes A, B and C would not be sufficient.
- Large address blocks were given to companies, government agencies and educational institutions, e. g. entire Class A networks to HP, IBM, AT&T, Apple, MIT, Generel Electric, US Army, ...

Consequence

- Inefficient allocation and use of address space
- Very large networks $\Rightarrow$ further subdivision of networks into subnets necessary

Situation today

- Allocation of the last IPv4 address block on Feb. 3, 2011 by the IANA[3] to one of the five Regional Internet Registries (RIRs), the APNIC[4]
- Available IPv4 address space is used up today

[3] Internet Assigned Numbers Authority

[4] Asia-Pacific Network Information Center (APNIC)

As early as 1993, CIDR[1] introduced a method for subdividing IP networks:

- In addition to the IP address, an interface receives a subnet mask that is also 32 bit long
- The subnet mask divides the IP address into a net portion and a host portion
- A logical 1 in the subnet mask means net portion, a logical 0 host portion
- The logical AND between IP address and subnet mask yields the network address
- The usual class affiliation thus only has a meaning in linguistic usage

Example 1:

IP address	11000000 . 10101000 . 00000000 . 10110010	192.168.0.178
Subnet mask	11111111 . 11111111 . 11111111 . 00000000	255.255.255.0
	Network / Host	
Network address	11000000 . 10101000 . 00000000 . 00000000	192.168.0.0
Broadcast address	11000000 . 10101000 . 00000000 . 11111111	192.168.0.255

- Network address: 192.168.0.0
- Broadcast address: 192.168.0.255
- 24 bit network portion, 8 bit host portion $\Rightarrow 2^8 = 256$ addresses
- Usable addresses for hosts: $2^8 - 2 = 254$

[1] Classless Inter-Domain Routing

As early as 1993, CIDR[1] introduced a method for subdividing IP networks:

- In addition to the IP address, an interface receives a subnet mask that is also 32 bit long
- The subnet mask divides the IP address into a net portion and a host portion
- A logical 1 in the subnet mask means net portion, a logical 0 host portion
- The logical AND between IP address and subnet mask yields the network address
- The usual class affiliation thus only has a meaning in linguistic usage

Example 2:

IP address	11000000 . 10101000 . 00000000 . 10110010	192.168.0.178
Subnet mask	11111111 . 11111111 . 11111111 . 10000000	255.255.255.128
	Network / Host	
Network address	11000000 . 10101000 . 00000000 . 10000000	192.168.0.128
Broadcast address	11000000 . 10101000 . 00000000 . 11111111	192.168.0.255

- Network address: 192.168.0.128
- Broadcast address: 192.168.0.255
- 25 bit Network portion, 7 bit host portion $\Rightarrow 2^7 = 128$ addresses
- Useable addresses for hosts: $2^7 - 2 = 126$

[1] Classless Inter-Domain Routing

As early as 1993, CIDR[1] introduced a method for subdividing IP networks:

- In addition to the IP address, an interface receives a subnet mask that is also 32 bit long
- The subnet mask divides the IP address into a net portion and a host portion
- A logical 1 in the subnet mask means net portion, a logical 0 host portion
- The logical AND between IP address and subnet mask yields the network address
- The usual class affiliation thus only has a meaning in linguistic usage

Example 3:

IP address	11000000 . 10101000 . 00000000 . 10110010	192.168.0.178
Subnet mask	11111111 . 11111111 . 11111111 . 11000000	255.255.255.192
	Network / Host	
Network address	11000000 . 10101000 . 00000000 . 10000000	192.168.0.128
Broadcast address	11000000 . 10101000 . 00000000 . 10111111	192.168.0.191

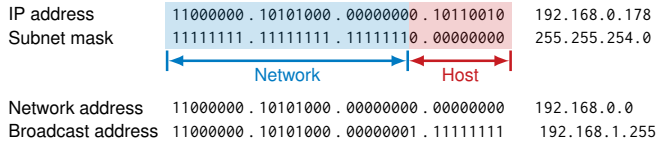
- Netzadresse: 192.168.0.128
- Broadcast-Adresse: 192.168.0.191
- 26 bit network portion, 6 bit host portion $\Rightarrow 2^6 = 64$ addresses
- Useable addresses for hosts: $2^6 - 2 = 62$

[1] Classless Inter-Domain Routing

The same principle also works in the other direction:

- Combination of several contiguous smaller networks into one larger network
- Often used by routers to reduce the number of entries in the routing table

Example:



- Network address: 192.168.0.0
- Broadcast address: 192.168.1.255
- 23 bit Network portion, 9 bit host portion $\Rightarrow 2^9 = 512$ addresses
- Useable addresses for hosts: $2^9 - 2 = 510$

Prefix notation: Instead of writing out the subnet mask, often only the length of the network portion (number of leading ones in the subnet mask) is specified, e. g. 192.168.0.0/23.

Question: can the networks 192.168.1.0/24 and 192.168.2.0/24 be combined to one larger network?

Question: can the networks 192.168.1.0/24 and 192.168.2.0/24 be combined to one larger network?

Answer: No, there is no valid subnet mask:

- 192.168.0.0/23 contains the networks 192.168.{0,1}.0/24
- 192.168.2.0/23 contains the networks 192.168.{2,3}.0/24
- 192.168.1.0/23 is not a valid network address since logical AND of 192.168.1.0 with the subnet mask 255.255.254.0 yields 192.168.0.0 as network address! ⇒ 192.168.1.0 is a host address in 192.168.0.0/23.

Binary represenation:

- Subnet mask must have a prefix length of 23 bit:

Network 1	11000000 . 10101000 . 00000001 . 00000000	192.168.1.0
Network 2	11000000 . 10101000 . 00000010 . 00000000	192.168.2.0
Subnet mask	11111111 . 11111111 . 11111110 . 00000000	255.255.254.0

- Results of the logical AND of the subnet mask with both network addresses would differ.

Question: can the four networks 192.168.{4,5,6,7}.0/24 be combined to one larger network?

Question: can the four networks 192.168.{4,5,6,7}.0/24 be combined to one larger network?

Answer: Yes, the network 192.168.4.0/22 spans exactly those four networks.

Binary represenation:

- Subnet mask must have a prefix length of 22 bit:

Network 1	11000000 . 10101000 . 00000100 . 00000000	192.168.4.0
Network 2	11000000 . 10101000 . 00000101 . 00000000	192.168.5.0
Network 3	11000000 . 10101000 . 00000110 . 00000000	192.168.6.0
Network 4	11000000 . 10101000 . 00000111 . 00000000	192.168.7.0
Subnet mask	11111111 . 11111111 . 11111100 . 00000000	255.255.252.0

- Results of the logical AND of the subnet mask with all network addresses are the same.

Internet Protocol version 4 (IPv4) [14]

Special address ranges

Some address ranges are reserved for special applications:

1. 0.0.0.0 / 8: hosts in this network / unspecified address
   - Used e. g. as source address with DHCP, if a client does not yet have an IP address.
   - Server applications expect incoming connections on address 0.0.0.0, which means “any address that is available”.
   - Other addresses within this range may be used as destination addresses for specific hosts within the local network.
   - Addresses from this range are generally not routed.
2. 127.0.0.0 / 8: “loopback addresses”
   - Addresses in this range identify the local host, e. g. 127.0.0.1.
   - These addresses are generally not routed.
   - Packets with this destination address would never physically be sent but looped back before transmission.
3. 10.0.0.0 / 8, 172.16.0.0 / 12, 192.168.0.0 / 16: private address ranges
   - May be used within local networks without registration by IANA.
   - May be routed between private networks.
   - Multiple allocation in separated private networks possible.
   - Must not be routed into public networks.
4. 169.254.0.0 / 16: Automatic Private IP Addressing (APIPA)
   - Used for self-assigned IP addresses.
   - Same rules for routing as for private address ranges apply.
5. 255.255.255.255 / 32: global broadcast
   - Identifies in principle all hosts.
   - Never routed.

IPv6 was proposed in late 1995 [6] as a successor to IPv4 and standardized with RFC 2460 [7] in 1998.

- It is probably older than most of you . . .
- . . . and its predecessor is still far from being obsolete.

Background is that

- address scarcity is the primary driver of IPv6 adoption, and
- this only became a problem in recent years.

[5] 2^{128} = 340.282.366.920.938.463.463.374.607.431.768.211.456 addresses

IPv6 was proposed in late 1995 [6] as a successor to IPv4 and standardized with RFC 2460 [7] in 1998.

- It is probably older than most of you . . .
- . . . and its predecessor is still far from being obsolete.

Background is that

- address scarcity is the primary driver of IPv6 adoption, and
- this only became a problem in recent years.

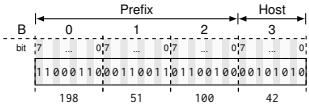
The fundamental changes compared to IPv4 are, among others:

- Enlargement of the address space from 2^{32} to 2^{128} [5]
- Simplification of the header format to allow for more efficient processing at routers
- Changes to IP fragmentation
- More flexibility by introducing extension headers while still streamlining the header format
- Stateless Address Autoconfiguration (SLAAC) using ICMPv6
- Stateful Autoconfiguration using DHCPv6
- Native deployment of multicast, e. g. to address all routers in a broadcast domain.

[5] 2^{128} = 340.282.366.920.938.463.463.374.607.431.768.211.456 addresses

Address format

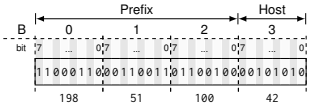
- IPv4 addresses are usually written as dotted-decimal notaton, i. e. each octet is written as decimal number separated by dots.



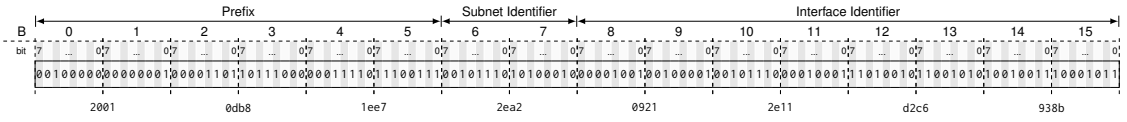
- With IPv6, this notation would result in 16 groups

Address format

- IPv4 addresses are usually written as dotted-decimal notaton, i. e. each octet is written as decimal number separated by dots.

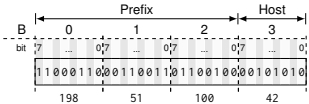


- With IPv6, this notation would result in 16 groups
- IPv6 addresses are thus represented in groups of 16 bit each in hexadecimal notation, each group separated by colons

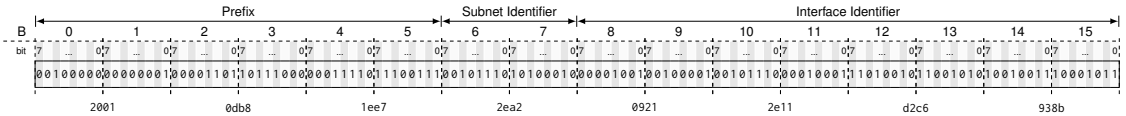


Address format

- IPv4 addresses are usually written as dotted-decimal notaton, i. e. each octet is written as decimal number separated by dots.



- With IPv6, this notation would result in 16 groups
- IPv6 addresses are thus represented in groups of 16 bit each in hexadecimal notation, each group separated by colons



Note:

- Please do not confuse IPv6 addresses with MAC addresses just because both are notated in hexadecimal.
- The prefix, which is also present in IPv4 notation, is different from the subnet identifier of IPv6.

Notes on the notation [10][1]

We consider the following address:

```
2001:0db8:0000:0000:0001:0000:0000:0001
```

[1] Further details can be found in RFC 5952.

Notes on the notation [10][1]

We consider the following address:

```
2001:0db8:0000:0000:0001:0000:0000:0001
```

1. Leading zeros in the individual blocks are omitted:

   ✓ `2001:db8:0:0:1:0:0:1`

[1] Further details can be found in RFC 5952.

Notes on the notation [10][1]

We consider the following address:

```
2001:0db8:0000:0000:0001:0000:0000:0001
```

1. Leading zeros in the individual blocks are omitted:
   ✓ `2001:db8:0:0:1:0:0:1`
2. At most one group of consecutive blanks consisting only of zeros may be abbreviated as follows:
   ✓ `2001:db8::1:0:0:1`

[1] Further details can be found in RFC 5952.

Notes on the notation [10][1]

We consider the following address:

```
2001:0db8:0000:0000:0001:0000:0000:0001
```

1. Leading zeros in the individual blocks are omitted:
   ✓ `2001:db8:0:0:1:0:0:1`
2. At most one group of consecutive blanks consisting only of zeros may be abbreviated as follows:
   ✓ `2001:db8::1:0:0:1`
3. If there are several possibilities for case 2), choose the longest sequence of zeros. If there are several sequences of the same length, select the first option. Wrong would be therefore:
   × `2001:db8:0:0:1::1`
   × `2001:db8::1::1`

[1] Further details can be found in RFC 5952.

Notes on the notation [10][1]

We consider the following address:

```
2001:0db8:0000:0000:0001:0000:0000:0001
```

1. Leading zeros in the individual blocks are omitted:
   ✓ `2001:db8:0:0:1:0:0:1`
2. At most one group of consecutive blanks consisting only of zeros may be abbreviated as follows:
   ✓ `2001:db8::1:0:0:1`
3. If there are several possibilities for case 2), choose the longest sequence of zeros. If there are several sequences of the same length, select the first option. Wrong would be therefore:
   × `2001:db8:0:0:1::1`
   × `2001:db8::1::1`
4. A single 0 block may not be abbreviated with `::`:
   ✓ `2001:db8:0:1:1:1:1:1`
   × `2001:db8::1:1:1:1:1`

Another example: The loopback address (same as 127.0.0.1 for IPv4) can be shortened as follows:

```
0000:0000:0000:0000:0000:0000:0000:0001/128
```

[1] Further details can be found in RFC 5952.

Notes on the notation [10][1]

We consider the following address:

`2001:0db8:0000:0000:0001:0000:0000:0001`

1. Leading zeros in the individual blocks are omitted:
   ✓ `2001:db8:0:0:1:0:0:1`
2. At most one group of consecutive blanks consisting only of zeros may be abbreviated as follows:
   ✓ `2001:db8::1:0:0:1`
3. If there are several possibilities for case 2), choose the longest sequence of zeros. If there are several sequences of the same length, select the first option. Wrong would be therefore:
   × `2001:db8:0:0:1::1`
   × `2001:db8::1::1`
4. A single 0 block may not be abbreviated with `::`:
   ✓ `2001:db8:0:1:1:1:1:1`
   × `2001:db8::1:1:1:1:1`

Another example: The loopback address (same as 127.0.0.1 for IPv4) can be shortened as follows:

`0000:0000:0000:0000:0000:0000:0000:0001/128` ↦ `::1/128`

[1] Further details can be found in RFC 5952.

Header Format

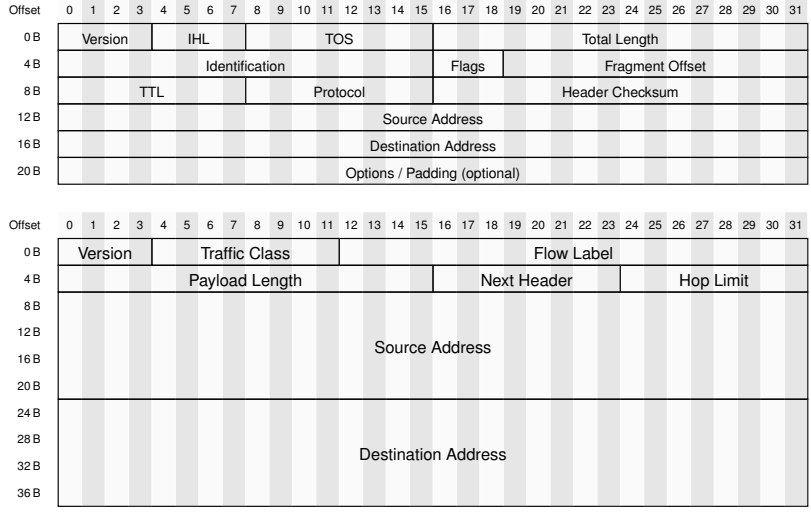


Figure 5: IPv4 header (top) and IPv6 header (bottom) in comparison

Headerformat

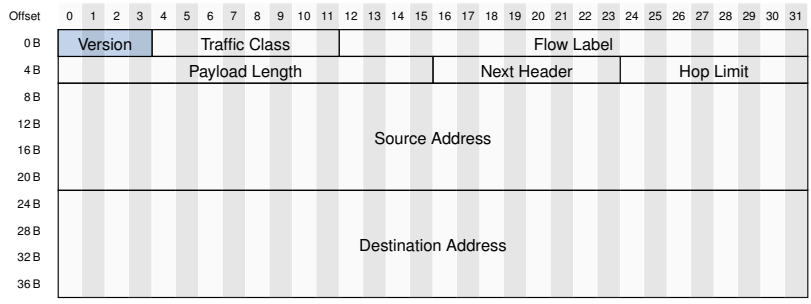


Figure 6: IPv6-Header (minimale Länge: 40 B, vgl. 20 B bei IPv4)

Version

- Specifies the used IP version.
- Valid values are 4 (IPv4) and 6 (IPv6).

Headerformat

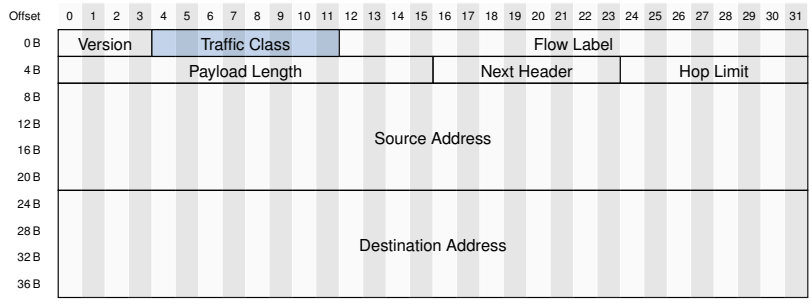


Figure 6: IPv6-Header (minimale Länge: 40 B, vgl. 20 B bei IPv4)

Traffic Class

- Equivalent to the TOS field of the IPv4 header.
- Used for quality of service (QoS, traffic prioritization)

Headerformat

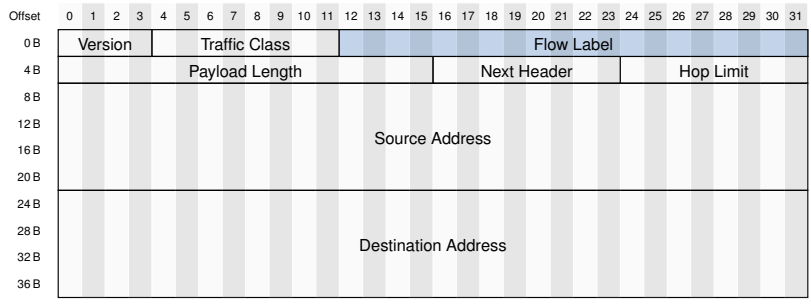


Figure 6: IPv6-Header (minimale Länge: 40 B, vgl. 20 B bei IPv4)

Flow Label

- Originally intended for real-time applications.
- Primarily used by routers today to detect merging packets (flows) at layer 3.
- Packets going to the same flow should be treated the same way if necessary, e.g. in case of multiple possible paths to the destination they should all be routed over the same path.

Headerformat

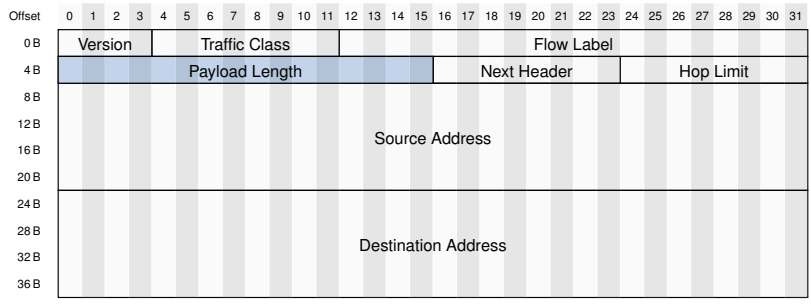


Figure 6: IPv6-Header (minimale Länge: 40 B, vgl. 20 B bei IPv4)

Payload Length

- Specifies the length of the data following the IPv6 header (including the extension header, more about this in a moment).
- Specified in multiples of 1 B.
- The IPv6 header including its extension header must always be a multiple of 8 B.

Headerformat

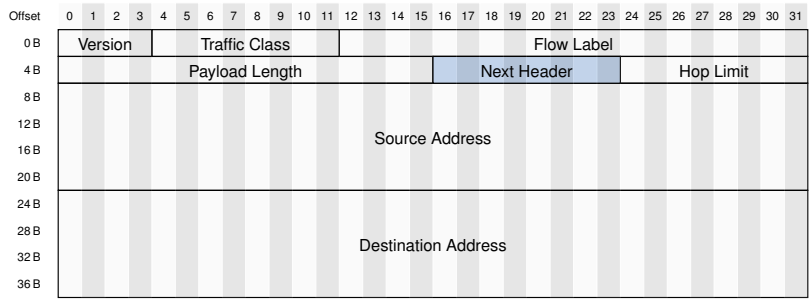


Figure 6: IPv6-Header (minimale Länge: 40 B, vgl. 20 B bei IPv4)

Next Header

- Specifies the type of the next header that follows the end of the IPv6 header.
- This can be either an L4 header (e.g. TCP or UDP), an ICMPv6 header, or a so-called IPv6 Extension Header (more about this in a moment).

Headerformat

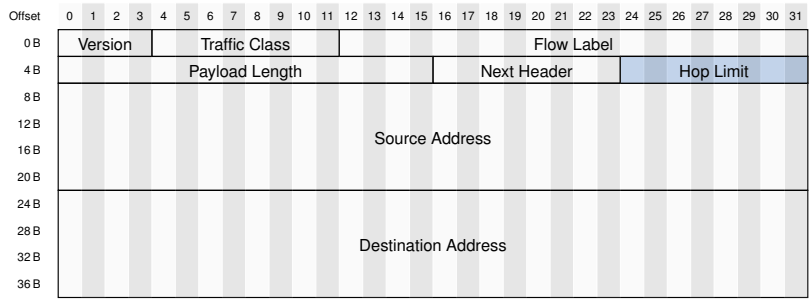


Figure 6: IPv6-Header (minimale Länge: 40 B, vgl. 20 B bei IPv4)

Hop Limit

- Corresponds to the TTL field of the IPv4 header.
- Decremented by 1 each time a packet is forwarded by a router.
- If the value reaches 0, the packet is dropped and an ICMPv6 Time Exceeded is returned to the original sender of the packet.

Headerformat

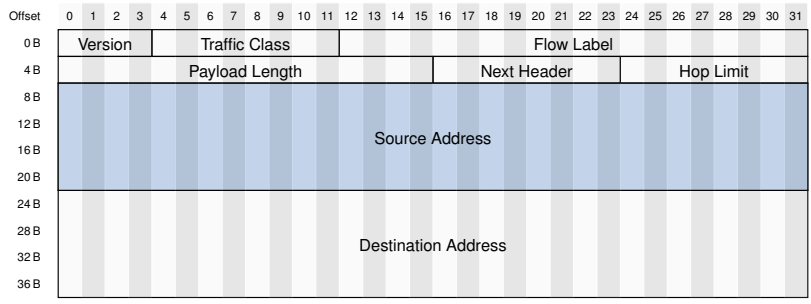


Figure 6: IPv6-Header (minimale Länge: 40 B, vgl. 20 B bei IPv4)

Source Address

- 128 bit long IPv6 source address.

Headerformat

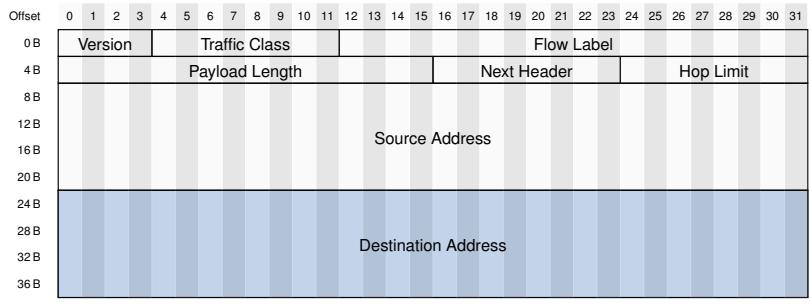


Figure 6: IPv6-Header (minimale Länge: 40 B, vgl. 20 B bei IPv4)

Destination Address

- 128 bit long IPv6 destination address.

IPv6 Extension Header [7]

- Extension headers allow to add additional layer 3 information in an IPv6 packet.
- The Next Header field indicates the next extension header or the L4 protocol.
- The Next Header fields of the IPv6 packet or the Extension Header form a chain, e.g.
  - IPv6 Header, Next Header: Routing
  - Routing Header, Next Header: Fragment
  - Fragment Header, Next Header: TCP
  - TCP Payload
- The format depends on the header.
- With the exception of the Hop-by-Hop Options Header, extension headers are evaluated only by the receiver.
  - Unknown extension headers cannot be processed by the receiver
    . → Packet is discarded with ICMP error message

In the following, we will take a look at the fragment header as an example.

Example:

- Frames often have a maximum size at layer 2, e.g. 1514 B for the L2 PDU (without CRC checksum) in IEEE 802.3u (100 Mbit/s Ethernet).
- This also specifies the maximum size of an L3 PDU, which is referred to as Maximum Transmission Unit (MTU).
- If an L3 PDU exceeds this size, the L3 SDU must be fragmented and sent as independent packets.
- The recipient must then reassemble the individual fragments.

Example:

- Frames often have a maximum size at layer 2, e.g. 1514 B for the L2 PDU (without CRC checksum) in IEEE 802.3u (100 Mbit/s Ethernet).
- This also specifies the maximum size of an L3 PDU, which is referred to as Maximum Transmission Unit (MTU).
- If an L3 PDU exceeds this size, the L3 SDU must be fragmented and sent as independent packets.
- The recipient must then reassemble the individual fragments.

IPv6 has its own extension header for this purpose, the Fragment Header:

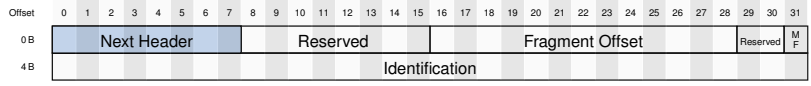


Figure 7: Fragment Header

Next Header

- Specifies the next extension header or the used L4 protocol

Example:

- Frames often have a maximum size at layer 2, e.g. 1514 B for the L2 PDU (without CRC checksum) in IEEE 802.3u (100 Mbit/s Ethernet).
- This also specifies the maximum size of an L3 PDU, which is referred to as Maximum Transmission Unit (MTU).
- If an L3 PDU exceeds this size, the L3 SDU must be fragmented and sent as independent packets.
- The recipient must then reassemble the individual fragments.

IPv6 has its own extension header for this purpose, the Fragment Header:

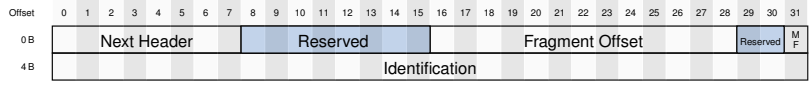


Figure 7: Fragment Header

Reserved

- Currently has no use with the Fragmentation Header.
- For most other extension headers, this field specifies the length of the extension header.

Example:

- Frames often have a maximum size at layer 2, e.g. 1514 B for the L2 PDU (without CRC checksum) in IEEE 802.3u (100 Mbit/s Ethernet).
- This also specifies the maximum size of an L3 PDU, which is referred to as Maximum Transmission Unit (MTU).
- If an L3 PDU exceeds this size, the L3 SDU must be fragmented and sent as independent packets.
- The recipient must then reassemble the individual fragments.

IPv6 has its own extension header for this purpose, the Fragment Header:

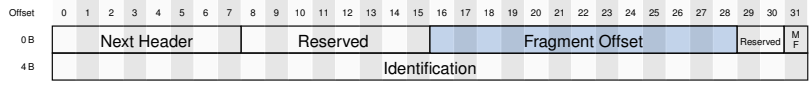


Figure 7: Fragment Header

Fragment Offset

- Offset of the fragmented L3 SDU in multiples of 8 B.
- With IPv6, the fragmentation *exclusively* takes place at the sender.
  Question: How does the transmitter know the MTU on the path from itself to the target?
- With IPv4, packets can also be fragmented by routers if necessary, unless explicitly forbidden by the DF bit.
  Question: Can fragments be fragmented again?

Example:

- Frames often have a maximum size at layer 2, e.g. 1514 B for the L2 PDU (without CRC checksum) in IEEE 802.3u (100 Mbit/s Ethernet).
- This also specifies the maximum size of an L3 PDU, which is referred to as Maximum Transmission Unit (MTU).
- If an L3 PDU exceeds this size, the L3 SDU must be fragmented and sent as independent packets.
- The recipient must then reassemble the individual fragments.

IPv6 has its own extension header for this purpose, the Fragment Header:

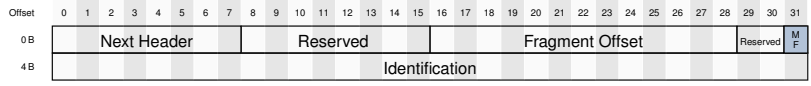


Figure 7: Fragment Header

More Fragments (MF)

- Indicates whether the current packet is followed by more fragments or whether it is the last fragment (according to the fragment offset).

Example:

- Frames often have a maximum size at layer 2, e.g. 1514 B for the L2 PDU (without CRC checksum) in IEEE 802.3u (100 Mbit/s Ethernet).
- This also specifies the maximum size of an L3 PDU, which is referred to as Maximum Transmission Unit (MTU).
- If an L3 PDU exceeds this size, the L3 SDU must be fragmented and sent as independent packets.
- The recipient must then reassemble the individual fragments.

IPv6 has its own extension header for this purpose, the Fragment Header:

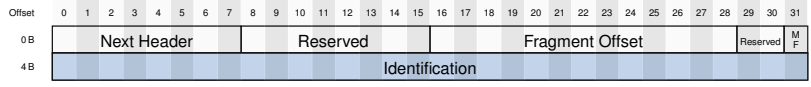


Figure 7: Fragment Header

Identification

- 32 bit long value, randomly chosen by the sender, which identifies all fragments to be reassembled into an L3 SDU.

Question: Why is the flow label of the IPv6 header not used?

IPv6 addresses for special use [4]

`::1/128` – Loopback Address

- Addresses the local host, i.e. packets with this destination address do not even leave the local computer, but are immediately redelivered via the so-called loopback interface (cf. 127.0.0.1 for IPv4).
- Are not routed.

`::/128` – non-specified Address

- Analog to 0.0.0.0 for IPv4.
- Will not be routed.

`fe80::/10` – Link-Local Addresses

- Each IPv6 interface requires a link local address.
- The link local address is generated from the interface identifier.
- Is only valid within the local link and is therefore not routed.

`fc00:/7` – Unique-Local Unicast Addresses

- Addresses that are only intended for local communication (e.g. within a company network).
- May be routed like private IPv4 addresses locally but not on the Internet.

`ff00::/8` – Multicast Addresses

- Addresses that address a specific group of hosts (more on this in a moment).
- Are routed.

(almost) all the rest – Global addresses

- Globally unique addresses intended for use in public networks.
- Are routed.

We distinguish the following four addressing types on layer 3/2:

1. Unicast
   - Packets (frames) addressed to a single destination.
   - All other nodes in the network discard such packets (frames) or merely forward them to the destination.
2. Broadcast
   - Packets (frames) addressed to all stations in the network.
   - Addressing is done by means of special broadcast addresses.
   - At layer 3, broadcasts are mostly limited to the local network segment (cf. broadcast domain).
3. Multicast
   - Packets (frames) addressed to a specific group of nodes.
   - Addressing is done by means of special multicast addresses.
   - At layer 2, multicasts from switches are often treated like broadcasts.
   - At layer 3, there are special protocols[1], which enable multicast addressing even beyond the local network segment.
4. Anycast[2]
   - Packets addressed to any station of a certain group.
   - Example: The most important name servers (→ Chapter 5).

[1] Protocol Independent Multicast (PIM) that enable the routing of multicast packets for both IPv4 and IPv6. Details are the content of further lectures

[2] Will not be discussed further within the scope of this lecture.

IPv6-Multicast

Multicast is an elementary component of IPv6. The following addresses or prefixes are therefore reserved for specific multicast groups:

`ff02::1` – All Nodes

- Addresses all nodes on the local link.

`ff02::2` – All Routers

- Addresses all routers on the local link.

`ff02::1:2` – All DHCP-Agents

- Addresses all DHCP servers on the local link.

`ff02::1:ff00:0/104` – Solicited-Node Address

- The solicited-node address is used in the Neighbor Discovery Protocol (more on this in a moment), which is used for address resolution, among other things.
- The solicited node address to an IPv6 address is generated from the `ff02::1:ff00:0/104` and the last 24 bit of the original IPv6 address.
- The Solicited-Node Address for `2001:0db8:1ee7:2ea2:0921:2e11:d2c6:938b` therefore is `ff02::1:ffc6:938b`.
- IPv6 multicasts are also sent at layer 2 using multicast addresses.
  - Switches must forward multicast frames only to the ports to which a member of the corresponding multicast group is connected.
  - In large L2 networks, unnecessary broadcasts can thus be avoided.
  - Nodes for which a message is not of interest will therefore not receive it in the first place.

Mapping Multicast IPv6 Addresses to MAC Addresses [5]

- IPv6 packets with a destination address from the prefix `ff00::/8` are sent with the associated multicast address on layer 2 (Ethernet).
- In order to be able to map multicasts on layer 3 to layer 2 as well, there must be a relationship between the addresses used on both layers.
- The first 2 octets of the MAC address are set to `33:33`.
  - last bit of the first octet is set → Multicast
  - second-to-last bit of the first octet is set → locally administered
  - see chapter 2
- The last 4 octets of the Ethernet address become the last 4 octets of the IPv6 multicast address.

Mapping Multicast IPv6 Addresses to MAC Addresses [5]

- IPv6 packets with a destination address from the prefix `ff00::/8` are sent with the associated multicast address on layer 2 (Ethernet).
- In order to be able to map multicasts on layer 3 to layer 2 as well, there must be a relationship between the addresses used on both layers.
- The first 2 octets of the MAC address are set to `33:33`.
  - last bit of the first octet is set → Multicast
  - second-to-last bit of the first octet is set → locally administered
  - see chapter 2
- The last 4 octets of the Ethernet address become the last 4 octets of the IPv6 multicast address.

Example:

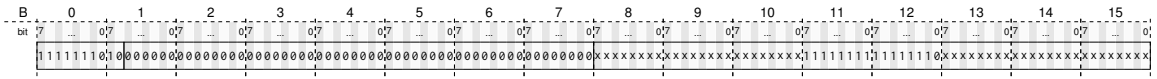
`ff02::1:ffc6:938b` ↦ `33:33:ff:c6:93:8b`

RFC 7042, Section 2.3.1:

> *"(Historical note: It was the custom during IPv6 design to use '3' for unknown or example values, and 3333 Coyote Hill Road, Palo Alto, California, is the address of PARC (Palo Alto Research Center, formerly 'Xerox PARC'). Ethernet was originally specified by the Digital Equipment Corporation, Intel Corporation, and Xerox Corporation. The pre-IEEE [802.3] Ethernet protocol has sometimes been known as 'DIX' Ethernet from the first letters of the names of these companies.)"*

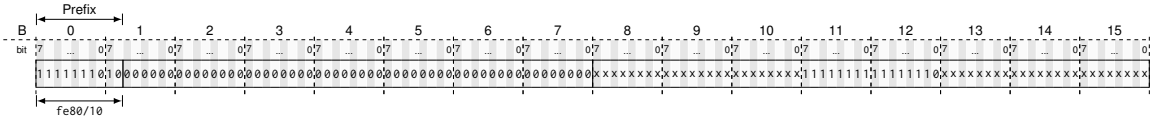
Stateless Address Autoconfiguration (SLAAC) [17]

IPv6 allows automatic configuration of hosts within a single subnet. A host generates the link-local IPv6 address required for an interface as follows:



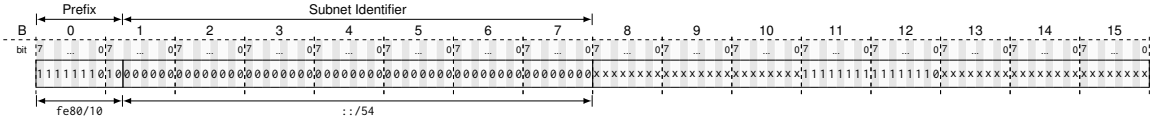
Stateless Address Autoconfiguration (SLAAC) [17]

IPv6 allows automatic configuration of hosts within a single subnet. A host generates the link-local IPv6 address required for an interface as follows:



- The prefix is `fe80::/10`.

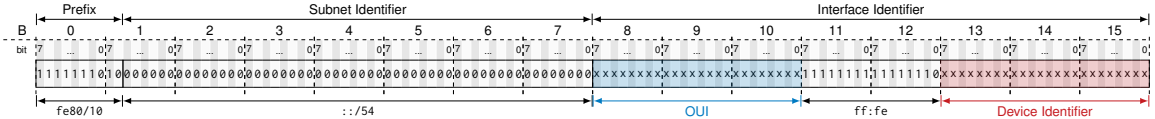
IPv6 allows automatic configuration of hosts within a single subnet. A host generates the link-local IPv6 address required for an interface as follows:



- The prefix is `fe80::/10`.
- The subnet identifier (the following 54 bit) are set to `0`.

Stateless Address Autoconfiguration (SLAAC) [17]

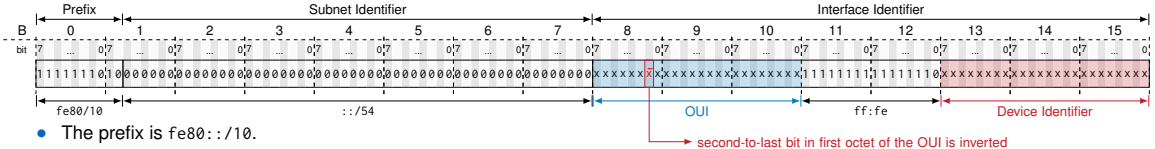
IPv6 allows automatic configuration of hosts within a single subnet. A host generates the link-local IPv6 address required for an interface as follows:



- The prefix is `fe80::/10`.
- The subnet identifier (the following 54 bit) are set to `0`.
- The remaining 64 bit represent the interface identifier, which is generated from the MAC address of the respective interface as modified EUI-64 identifier:
  - The first 24 bit are the OUI of the MAC address.
  - The following 16 bit are stuffed with `ff:fe`
  - The remaining 24 bit are filled with the Device Identifier of the MAC address.

Stateless Address Autoconfiguration (SLAAC) [17]

IPv6 allows automatic configuration of hosts within a single subnet. A host generates the link-local IPv6 address required for an interface as follows:



- The prefix is `fe80::/10`.
- The subnet identifier (the following 54 bit) are set to `0`.
- The remaining 64 bit represent the interface identifier, which is generated from the MAC address of the respective interface as modified EUI-64 identifier:
  - The first 24 bit are the OUI of the MAC address.
  - The following 16 bit are stuffed with `ff:fe`
  - The remaining 24 bit are filled with the Device Identifier of the MAC address.
- The second-to-last bit of the first octet of the OUI (global/local bit) is inverted:
  - For MAC addresses, a `0` at this bit position means a globally unique address and a `1` means a locally administered address (see chapter 2).
  - With IPv6 it is the other way around: The inversion ensures that a manually configured IPv6 address like `2001:db8::1` does not contain an interface identifier that points to a globally unique MAC address.
  - Otherwise you would have to manually assign addresses like `2001:db8::200:0:0:1` ...

Stateless Address Autoconfiguration (SLAAC) [17]

Global addresses can also be configured via SLAAC:

- To be able to configure a global address, the host must first know which IPv6 prefixes are served by the local routers.
- Prefix information can be sent by routers via Neighbor Discovery Protocol (later) in the form of Router Advertisements.
- The host can generate an address by itself using the /64 prefix and the modified EUI-64 identifier.
- SLAAC is called stateless because the addresses are not assigned by a server
  - Global addresses can also be assigned via DHCPv6

Stateless Address Autoconfiguration (SLAAC) [17]

Global addresses can also be configured via SLAAC:

- To be able to configure a global address, the host must first know which IPv6 prefixes are served by the local routers.
- Prefix information can be sent by routers via Neighbor Discovery Protocol (later) in the form of Router Advertisements.
- The host can generate an address by itself using the /64 prefix and the modified EUI-64 identifier.
- SLAAC is called stateless because the addresses are not assigned by a server
  - Global addresses can also be assigned via DHCPv6

Question: What are the privacy implications of generating the interface identifier from the MAC address of a network card?

Stateless Address Autoconfiguration (SLAAC) [17]

Global addresses can also be configured via SLAAC:

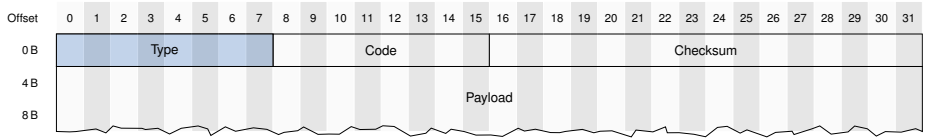
- To be able to configure a global address, the host must first know which IPv6 prefixes are served by the local routers.
- Prefix information can be sent by routers via Neighbor Discovery Protocol (later) in the form of Router Advertisements.
- The host can generate an address by itself using the /64 prefix and the modified EUI-64 identifier.
- SLAAC is called stateless because the addresses are not assigned by a server
  - Global addresses can also be assigned via DHCPv6

Question: What are the privacy implications of generating the interface identifier from the MAC address of a network card?

- Since MAC addresses are usually unique, a host can be tracked by the MAC address embedded in its IPv6 address, regardless of standard, port, or provider.
- The IPv6 Privacy Extensions provide a remedy for this problem [12]:
  - Generation and periodic renewal of random device identifiers and associated change of global IPv6 address.
  - The second-to-last bit of the first octet of „OUI“ is set to `0`.

General structure of an ICMPv6 message:

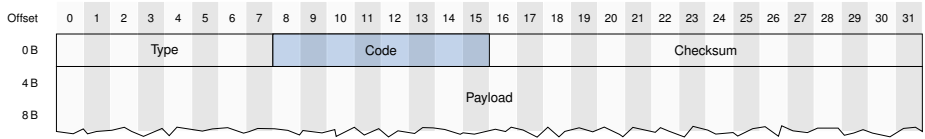
- The first 32 bit are always present.
- Total length of the message depends on the type and code of the ICMPv6 message.
- The length, within the limits applicable to IPv6, to 1 B accurate.



- Type specifies the type of ICMP message, e.g. Echo Request or Reply, Time Exceeded, etc. . . .

General structure of an ICMPv6 message:

- The first 32 bit are always present.
- Total length of the message depends on the type and code of the ICMPv6 message.
- The length, within the limits applicable to IPv6, to 1 B accurate.



- Code specifies the type of the message, e.g. why a destination was not reachable.

General structure of an ICMPv6 message:

- The first 32 bit are always present.
- Total length of the message depends on the type and code of the ICMPv6 message.
- The length, within the limits applicable to IPv6, to 1 B accurate.

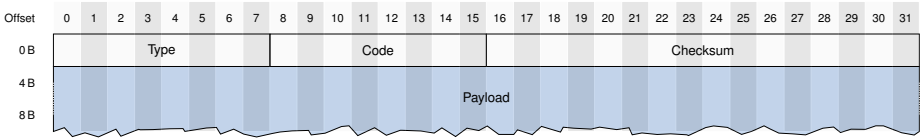
- Checksum is a (comparatively simple) error-detecting checksum:
  - The checksum takes into account the entire ICMPv6 packet (including payload).
  - For the calculation a so called IPv6 pseudo header is used for the calculation.
  - The pseudo header contains the sender and destination IP and the next header field.

Why a „ pseudo header“?

- As we know from chapter 1/2, channel coding and checksums cannot guarantee that a message has been transmitted correctly even with valid checksums.
- Further checksums on higher layers can't do this either, but the probability for an unnoticed error decreases drastically here even with simple checksums.
- At the same time, a checksum over header fields, which change from hop to hop, would put some load on routers (or on layer 2 for switches).
- Therefore, in this case the checksum is formed only over the part of the message which usually does not (or only very rarely) change on the way from sender to receiver.
- For example, note the exclusion of the hopcount, which does change on each hop.

General structure of an ICMPv6 message:

- The first 32 bit are always present.
- Total length of the message depends on the type and code of the ICMPv6 message.
- The length, within the limits applicable to IPv6, to 1 B accurate.



- Payload of the ICMPv6 message (examples follow).

Neighbor Discovery Protocol (NDP) [13]

With its neighbor discovery, which is part of ICMPv6, IPv6 offers a number of functionalities for which IPv4 is not standardized or only incompletely standardized and has made separate protocols necessary. Features of Neighbor Discovery include, in particular:

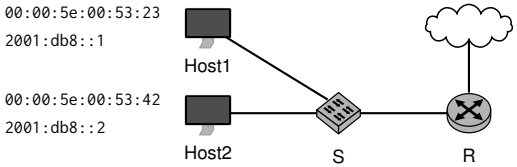
- Address Resolution, Duplicate Address Detection and Neighbor Unreachability Detection: Neighbor Solicitations and Advertisements.
- Automatic discovery of routers within the local network segment, address prefixes and parameter configuration: Router Discovery and Router Advertisements.
- Redirects to other gateways: redirects.

With its neighbor discovery, which is part of ICMPv6, IPv6 offers a number of functionalities for which IPv4 is not standardized or only incompletely standardized and has made separate protocols necessary. Features of Neighbor Discovery include, in particular:

- Address Resolution, Duplicate Address Detection and Neighbor Unreachability Detection: Neighbor Solicitations and Advertisements.
- Automatic discovery of routers within the local network segment, address prefixes and parameter configuration: Router Discovery and Router Advertisements.
- Redirects to other gateways: redirects.

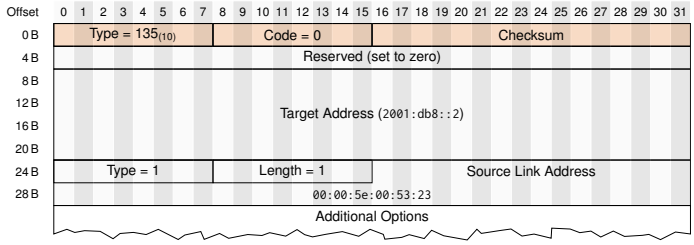
Example: Address resolution for IPv6[3]

- Host1 wants to send a message to Host2
- The IP address of Host2 (`2001:db8::2`) is already known to him.
- How does Host1 get the associated MAC address?



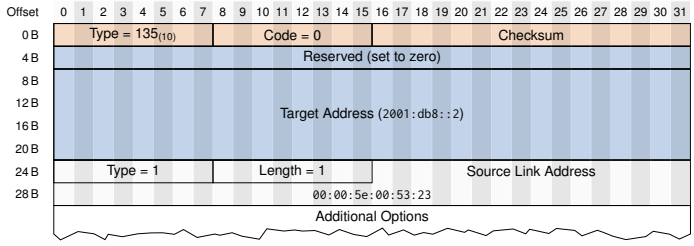
[3] We only consider the case of address resolution (cf. ARP for IPv4). When Neighbor Solicitations and Neighbor Advertisements are used for other purposes, other addresses and options are usually required.

Neighbor Discovery Protocol (NDP) [13] – Neighbor Solicitation (Request)



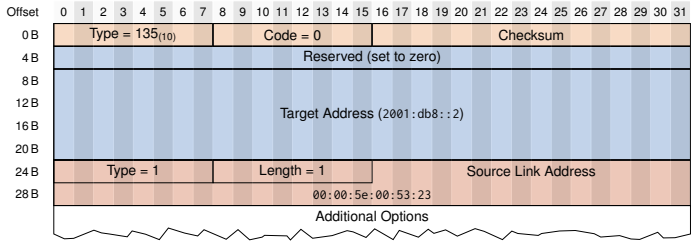
- ICMPv6 Header
  - ICMPv6 Type and Code (`0x87` and `0x00` for a Neighbor Solicitation message) and the ICMPv6 checksum.

Neighbor Discovery Protocol (NDP) [13] – Neighbor Solicitation (Request)



- ICMPv6 Header
  - ICMPv6 Type and Code (`0x87` and `0x00` for a Neighbor Solicitation message) and the ICMPv6 checksum.
- Neighbor Discovery Body
  - The first 32 bit are reserved, so the message as a whole becomes again a multiple of 8 B long.
  - This is followed by the destination IPv6 address for which the corresponding MAC address is searched.

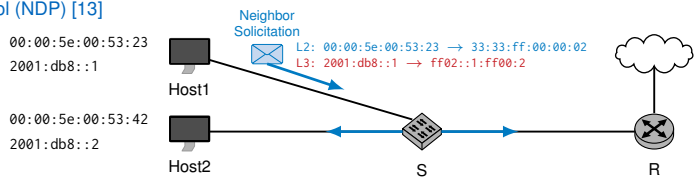
Neighbor Discovery Protocol (NDP) [13] – Neighbor Solicitation (Request)



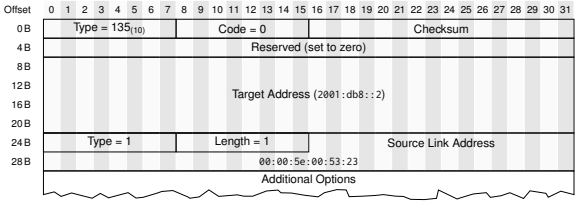
- ICMPv6 Header
  - ICMPv6 Type and Code (`0x87` and `0x00` for a Neighbor Solicitation message) and the ICMPv6 checksum.
- Neighbor Discovery Body
  - The first 32 bit are reserved, so the message as a whole becomes again a multiple of 8 B long.
  - This is followed by the destination IPv6 address for which the corresponding MAC address is searched.
- Neighbor Discovery Options
  - Neighbor Discovery packets can themselves in turn contain options.
  - Type and Length specify the type (1 for Source Link Layer Address) and total length of the option in multiples of 8 B.
  - In the case of a Neighbor Solicitation packet, L2 address of the requesting node follows (Source Link Address).
  - Depending on the type of NDP packet, other options may follow, and recipients must ignore unknown options.

Neighbor Discovery Protocol (NDP) [13]

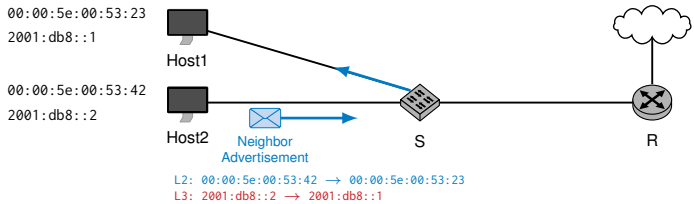
Example:



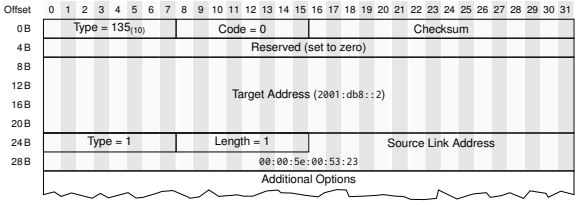
- Host 1 sends a „Neighbor Solicitation for `2001:db8::2` from `00:00:5e:00:53:23`“ to the solicited-node address associated with the known IPv6 address (multicast).

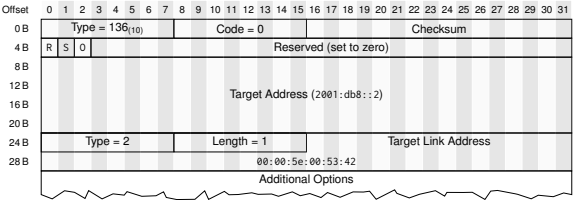


Example:

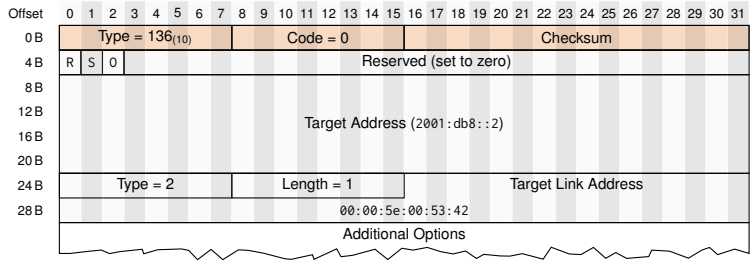


- Host 1 sends a „Neighbor Solicitation for `2001:db8::2` from `00:00:5e:00:53:23`“ to the solicited-node address associated with the known IPv6 address (multicast).
- Host 2 receives this message and responds with a „Neighbor Advertisement `2001:db8::2` (`sol`, `ovr`) is at `00:00:5e:00:53:42`“ message (unicast).



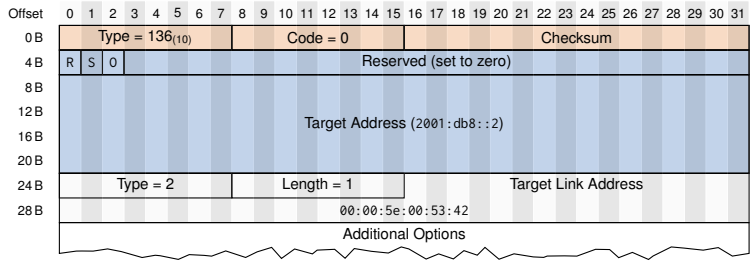


Neighbor Discovery Protocol (NDP) [13] – Neighbor Advertisement (Reply)



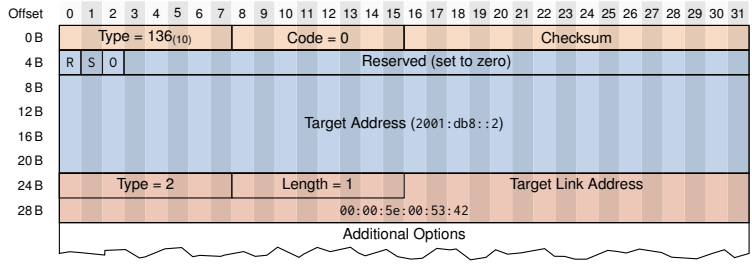
- ICMPv6 Header
  - ICMPv6 Type and Code (`0x88` and `0x00` for a Neighbor Advertisement) and the ICMPv6 checksum.

Neighbor Discovery Protocol (NDP) [13] – Neighbor Advertisement (Reply)



- ICMPv6 Header
  - ICMPv6 Type and Code (`0x88` and `0x00` for a Neighbor Advertisement) and the ICMPv6 checksum.
- Neighbor Discovery Body
  - The three most significant bits of the first octet have the following meanings:
    - Router flag `R` is set if the responding node is a router.
    - Solicited Flag `S` indicates whether the advertisement is sent as a result of a solicitation.
    - Override Flag `O` is set if the advertisement should update a possibly cached link layer address at the recipient.

Neighbor Discovery Protocol (NDP) [13] – Neighbor Advertisement (Reply)



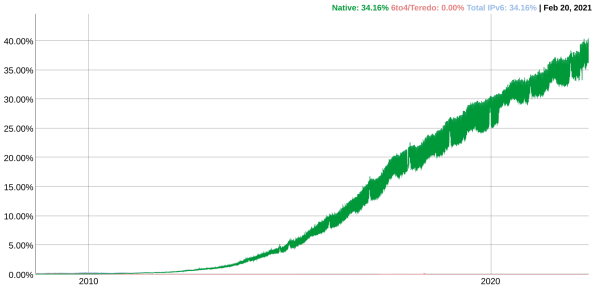
- ICMPv6 Header
  - ICMPv6 Type and Code (`0x88` and `0x00` for a Neighbor Advertisement) and the ICMPv6 checksum.
- Neighbor Discovery Body
  - The three most significant bits of the first octet have the following meanings:
    - Router flag `R` is set if the responding node is a router.
    - Solicited Flag `S` indicates whether the advertisement is sent as a result of a solicitation.
    - Override Flag `O` is set if the advertisement should update a possibly cached link layer address at the recipient.
- Neighbor Discovery Options
  - Type and Length specify the type (`2` for Target Link Layer Address) and total length of the option in multiples of 8 B.
  - In the case of a Neighbor Advertisement, the L2 address of the requested node follows (Target Link Address).
  - Depending on the type of NDP packet, other options may follow, and recipients must ignore unknown options.

Compatibility

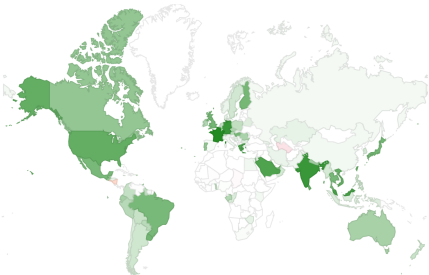
- IPv4 and IPv6 are not compatible, but can coexist.
- Today, IPv6 is often used in parallel with IPv4 (Dual Stack).
- Routers must exchange and process routing information separately for both protocol versions.

Rate of adoption, 2022:

- Although IPv6 has been standardized since 1998 (RFC 8200), the transition to IPv6 is far from complete.
- Around 60 % of global traffic is still IPv4:



(a) IPv6 Traffic [1]

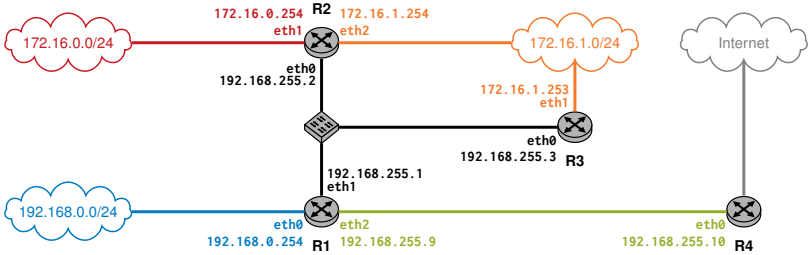


(b) IPv6 by Country [1]

Chapter 3: Network Layer

Static routing

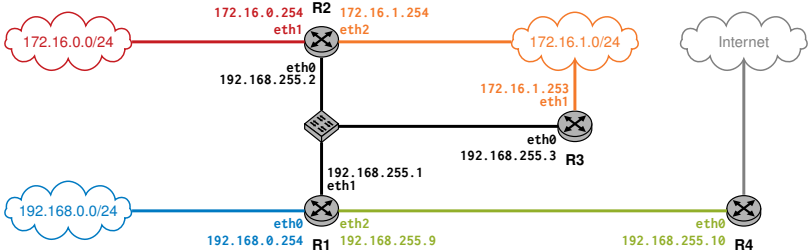
We consider below the example network shown below:



- The colors of the links and interface addresses indicate the individual subnets
- The network 192.168.255.0/29 (black) has 6 usable host addresses
- The network 192.168.255.8/30 (green) is a transport network with only 2 usable host addresses
- The remaining networks are /24 networks with 254 usable host addresses each.

Question: How does R1 decide which Next-Hop to forward a packet to?

Routing Table



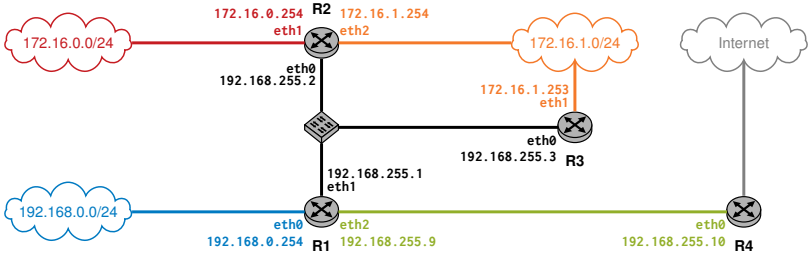
Routing Table

In the routing table a router (or host) stores

- the network address of a destination,
- the length of the prefix,
- the associated next-hop (also called gateway),
- the interface through which this next-hop is accessible, and
- the cost to the destination.

Notes:

- For IPv4, the subnet mask (Linux terminology: Genmask) is often specified instead of the prefix length. For a prefix of *N* bit, it describes (for IPv4) an IP address of a network where the first *N* bit are the 1 and the remaining bits are 0.
- cost is to be distinguished from metric. Cost refers to a metric (e.g. hop count, bandwidth, delay, etc.).



Example: routing table for R1

Destination	NextHop	Costs	Iface
192.168.255.8/30	0.0.0.0	0	eth2
192.168.255.0/29	0.0.0.0	0	eth1
192.168.0.0/24	0.0.0.0	0	eth0
172.16.1.0/24	192.168.255.3	1	eth1
172.16.0.0/23	192.168.255.2	1	eth1
0.0.0.0/0	192.168.255.10	0	eth2

- The networks 172.16.{0,1}.0/24 were combined.
- The route 0.0.0.0/0 is also called Default Route.
- Interesting: R1 knows two (actually even three) routes to the net 172.16.1.0/24 !

1. R1 calculates the logical AND from the destination address of the packet and the subnet masks (which come from the prefix length) in its routing table.
2. The result is compared with the entry in the „ Destination“ column.
3. If the result matches, the gateway and associated interface are determined.
4. After the MAC address of the gateway has been resolved via ARP, if necessary, the packet is provided with a new Ethernet header and forwarded.

Example: R1 receives a packet with destination address 172.16.1.23.

Destination	NextHop	Costs	Iface
192.168.255.8/30	0.0.0.0	0	eth2
192.168.255.0/29	0.0.0.0	0	eth1
192.168.0.0/24	0.0.0.0	0	eth0
172.16.1.0/24	192.168.255.3	1	eth1
172.16.0.0/23	192.168.255.2	1	eth1
0.0.0.0/0	192.168.255.10	0	eth2

Longest Prefix Matching

1. R1 calculates the logical AND from the destination address of the packet and the subnet masks (which come from the prefix length) in its routing table.
2. The result is compared with the entry in the „ Destination“ column.
3. If the result matches, the gateway and associated interface are determined.
4. After the MAC address of the gateway has been resolved via ARP, if necessary, the packet is provided with a new Ethernet header and forwarded.

Example: R1 receives a packet with destination address 172.16.1.23.

	Destination	NextHop	Costs	Iface
→	192.168.255.8/30	0.0.0.0	0	eth2
	192.168.255.0/29	0.0.0.0	0	eth1
	192.168.0.0/24	0.0.0.0	0	eth0
	172.16.1.0/24	192.168.255.3	1	eth1
	172.16.0.0/23	192.168.255.2	1	eth1
	0.0.0.0/0	192.168.255.10	0	eth2

IP Address	10101100 . 00010000 . 00000001 . 00010111	172.16.1.23
Subnet Mask	11111111 . 11111111 . 11111111 . 11111100	255.255.255.252
Net Address	10101100 . 00010000 . 00000001 . 00010100	172.16.1.20

$\Rightarrow$ No match, since 172.16.1.20 $\neq$ 192.168.255.8

1. R1 calculates the logical AND from the destination address of the packet and the subnet masks (which come from the prefix length) in its routing table.
2. The result is compared with the entry in the „ Destination“ column.
3. If the result matches, the gateway and associated interface are determined.
4. After the MAC address of the gateway has been resolved via ARP, if necessary, the packet is provided with a new Ethernet header and forwarded.

Example: R1 receives a packet with destination address 172.16.1.23.

	Destination	NextHop	Costs	Iface
	192.168.255.8/30	0.0.0.0	0	eth2
→	192.168.255.0/29	0.0.0.0	0	eth1
	192.168.0.0/24	0.0.0.0	0	eth0
	172.16.1.0/24	192.168.255.3	1	eth1
	172.16.0.0/23	192.168.255.2	1	eth1
	0.0.0.0/0	192.168.255.10	0	eth2

IP Address	10101100 . 00010000 . 00000001 . 00010111	172.16.1.23
Subnet Mask	11111111 . 11111111 . 11111111 . 11111000	255.255.255.248
Net Address	10101100 . 00010000 . 00000001 . 00010000	172.16.1.16

$\Rightarrow$ No match, since 172.16.1.16 $\neq$ 192.168.255.0

Longest Prefix Matching

1. R1 calculates the logical AND from the destination address of the packet and the subnet masks (which come from the prefix length) in its routing table.
2. The result is compared with the entry in the „ Destination“ column.
3. If the result matches, the gateway and associated interface are determined.
4. After the MAC address of the gateway has been resolved via ARP, if necessary, the packet is provided with a new Ethernet header and forwarded.

Example: R1 receives a packet with destination address 172.16.1.23.

	Destination	NextHop	Costs	Iface
	192.168.255.8/30	0.0.0.0	0	eth2
	192.168.255.0/29	0.0.0.0	0	eth1
→	192.168.0.0/24	0.0.0.0	0	eth0
	172.16.1.0/24	192.168.255.3	1	eth1
	172.16.0.0/23	192.168.255.2	1	eth1
	0.0.0.0/0	192.168.255.10	0	eth2

IP Address	10101100 . 00010000 . 00000001 . 00010111	172.16.1.23
Subnet Mask	11111111 . 11111111 . 11111111 . 00000000	255.255.255.0
Net Address	10101100 . 00010000 . 00000001 . 00000000	172.16.1.0

$\Rightarrow$ No match, since 172.16.1.0 $\neq$ 192.168.0.0

Longest Prefix Matching

1. R1 calculates the logical AND from the destination address of the packet and the subnet masks (which come from the prefix length) in its routing table.
2. The result is compared with the entry in the „ Destination“ column.
3. If the result matches, the gateway and associated interface are determined.
4. After the MAC address of the gateway has been resolved via ARP, if necessary, the packet is provided with a new Ethernet header and forwarded.

Example: R1 receives a packet with destination address 172.16.1.23.

	Destination	NextHop	Costs	Iface
	192.168.255.8/30	0.0.0.0	0	eth2
	192.168.255.0/29	0.0.0.0	0	eth1
	192.168.0.0/24	0.0.0.0	0	eth0
→	172.16.1.0/24	192.168.255.3	1	eth1
	172.16.0.0/23	192.168.255.2	1	eth1
	0.0.0.0/0	192.168.255.10	0	eth2

IP Address	10101100 . 00010000 . 00000001 . 00010111	172.16.1.23
Subnet Mask	11111111 . 11111111 . 11111111 . 00000000	255.255.255.0
Net Address	10101100 . 00010000 . 00000001 . 00000000	172.16.1.0

⇒ Match, since 172.16.1.0 = 172.16.1.0 ⇒ Gateway is 192.168.255.3

Longest Prefix Matching
The routing table is searched from longer prefixes (more specific routes) to shorter prefixes (less specific routes). The first matching entry provides the gateway (next-hop) of a packet. This process is called Longest Prefix Matching.

Destination	NextHop	Costs	Iface
192.168.255.8/30	0.0.0.0	0	eth2
192.168.255.0/29	0.0.0.0	0	eth1
192.168.0.0/24	0.0.0.0	0	eth0
172.16.1.0/24	192.168.255.3	1	eth1
172.16.0.0/23	192.168.255.2	1	eth1
0.0.0.0/0	192.168.255.10	0	eth2

- The entry for 172.16.0.0/23 also provides a match, but is less specific than the one for 172.16.1.0/24 (1 bit shorter prefix).
- The default route 0.0.0.0/0 always returns a match (logical AND with mask 0.0.0.0).
- It is not guaranteed that the gateway/next-hop of the default route („gateway of last resort“) knows a route to the destination (→ ICMP Destination Unreachable / Host Unreachable).
- Routes to directly connected networks (i.e. those to which a router itself belongs) can be generated automatically. The NextHop in this case is the unspecified address.
- Routes to remote networks must be „learned“ – either by manual entry (static routing) or by routing protocols (dynamic routing).

Dynamic routing

By means of routing protocols routers can communicate with each other and exchange routes among themselves. Routing protocols can be grouped according to their operation as follows:

Distance Vector Protocols

- Routers only know the direction (NextHop) and distance (cost) to a destination (compare street sign with direction and distance information).
- Routers have no information about the network topology.
- Routers exchange only cumulative costs among themselves (e.g., the contents of your routing tables).
- Functional principle is based on the algorithm of Bellman-Ford, which determines shortest paths starting from a starting node and can be easily implemented in a distributed way.

Link State Protocols

- Routers inform each other about how to reach a destination in addition to the cost.
- Often feature complex neighborhood relationships and update messages.
- Routers thus receive complete topology information.
- Based on the topology information, each router determines shortest paths, e.g., using Dijkstras algorithm.

Dynamic routing

There are two basic algorithms everyone should know about (covered in detail in other lectures):

Bellman Ford

- Commonly used in protocols referred to as Distance Vector Protocols
- Algorithm based on a kind of breadth first search
- In the n-th iteration, paths with a length of at most n hops are being considered
- If no changes to the topology occur, the convergence time is given by the longest (cycle-free) path in the network
- No complex data structures necessary.
- Distributed (decentralized) implementation possible without knowledge of the topology.
- Runtime in $\mathcal{O}(|N| \cdot |E|)$.

Dijkstra

- Commonly used in protocols referred to as Link State protocols
- Greedy algorithm, i. e., shortest paths are detected one after another
- Nodes generally maintain a global view of the network topology
- More resource-intensive than the Bellman-Ford algorithm, since more complex data structures are necessary (priority queue).
- Runtime in $\mathcal{O}(|E| + |N| \log_2 |N|)$.

Routing Information Protocol (RIP) [8, 11]

Properties:

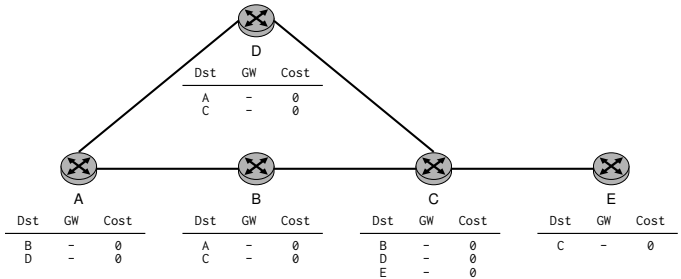
- Simple distance vector protocol
- RIPv1 standardized in RFC 1058 (1988)
- Added support for CIDR in RIPv2 (RFC 2453, 1998)
- Only metric: hop count (equivalent to Bellman-Ford with edge weight 1 on all edges)
- Hop count limit of 15; destinations further away are not reachable

Functionality:

- Routers send the contents of their routing table to the multicast address 224.0.0.9 at regular intervals (default value 30 s).
- All devices with this multicast address accept the update.
- Each RIP router accepts these update messages, increments the cost of the included routes by 1, and compares the routes with existing routes from its routing table:
  - If the update contains a route that is still unknown, it will be added to its own routing table.
  - If the update contains a route to a known destination but with a lower cost, the update replaces the existing route.
  - Otherwise, the existing route will be kept.
- If five consecutive updates from a neighbor are missing, all routes over that next hop are removed from the routing table.

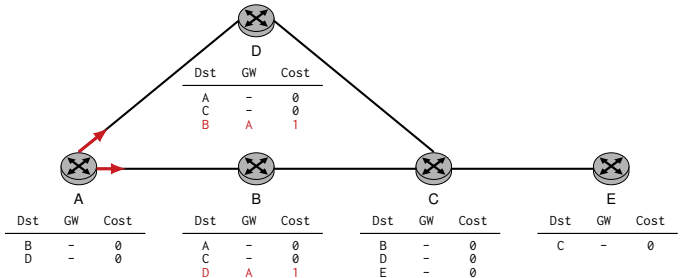
Example with simplified representation:

- It makes sense to have a switch between every two routers, which enables the connection of further computers to the respective subnet.
- Instead of IP and network addresses, we only enter the names of the routers in the routing tables.
- We do not enter a gateway for directly accessible neighbors.
- Update messages are sent in rounds (first A, then B, . . .).



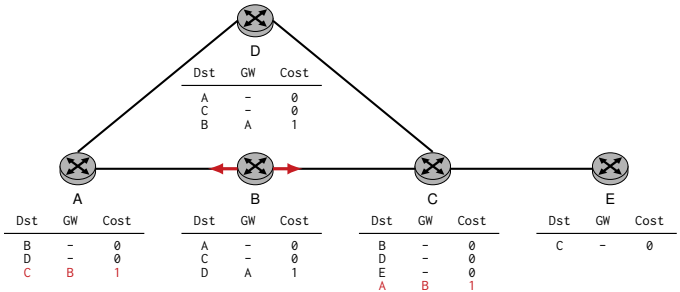
Example with simplified representation:

- It makes sense to have a switch between every two routers, which enables the connection of further computers to the respective subnet.
- Instead of IP and network addresses, we only enter the names of the routers in the routing tables.
- We do not enter a gateway for directly accessible neighbors.
- Update messages are sent in rounds (first A, then B, . . .).



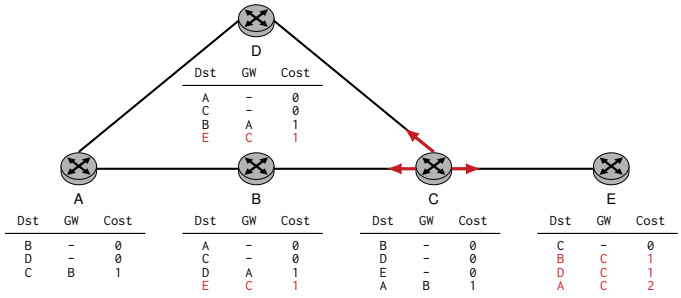
Example with simplified representation:

- It makes sense to have a switch between every two routers, which enables the connection of further computers to the respective subnet.
- Instead of IP and network addresses, we only enter the names of the routers in the routing tables.
- We do not enter a gateway for directly accessible neighbors.
- Update messages are sent in rounds (first A, then B, . . .).



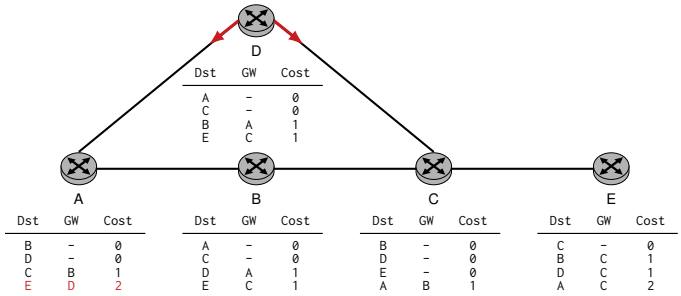
Example with simplified representation:

- It makes sense to have a switch between every two routers, which enables the connection of further computers to the respective subnet.
- Instead of IP and network addresses, we only enter the names of the routers in the routing tables.
- We do not enter a gateway for directly accessible neighbors.
- Update messages are sent in rounds (first A, then B, . . .).



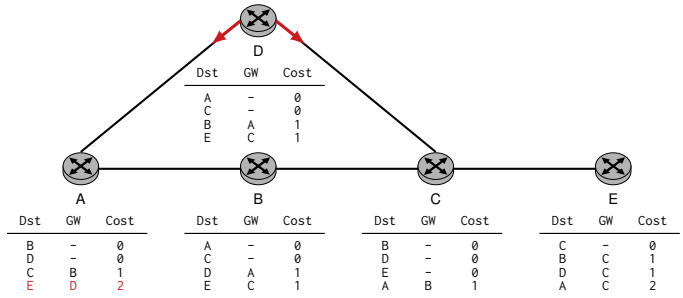
Example with simplified representation:

- It makes sense to have a switch between every two routers, which enables the connection of further computers to the respective subnet.
- Instead of IP and network addresses, we only enter the names of the routers in the routing tables.
- We do not enter a gateway for directly accessible neighbors.
- Update messages are sent in rounds (first A, then B, . . .).



Example with simplified representation:

- It makes sense to have a switch between every two routers, which enables the connection of further computers to the respective subnet.
- Instead of IP and network addresses, we only enter the names of the routers in the routing tables.
- We do not enter a gateway for directly accessible neighbors.
- Update messages are sent in rounds (first A, then B, . . .).



- After this step, each router knows a shortest route to any other router.
- Since there are several paths of equal length, it is left to chance (the order of the update messages) whether, for example, router A learns router D or B as the next hop to E.

The previous example can be easily extended to weighted edges:

- Metric is no longer the number of hops to the target, but the edge weight.
- Routers add the edge weight of the link through which the update was received to the cost in the updates.

Problem:

- It may take several rounds for each router to determine the best next hop.
- An upper bound on the number of messages each router needs to send is the maximum distance between two routers in hops.
- The maximum distance is 15 hops for RIP.
- Since updates are sent only every 30 s, this results in a maximum delay of $15 \cdot 30\,\text{s} = 7.5\,\text{min}$.

The previous example can be easily extended to weighted edges:

- Metric is no longer the number of hops to the target, but the edge weight.
- Routers add the edge weight of the link through which the update was received to the cost in the updates.

Problem:

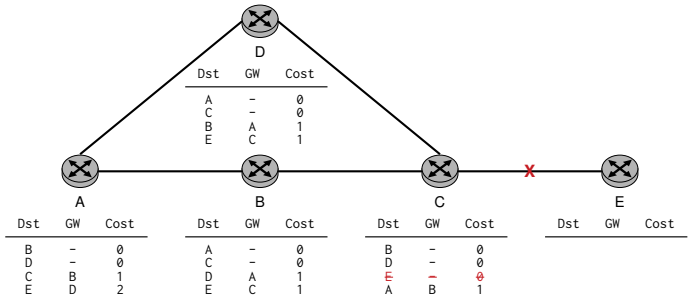
- It may take several rounds for each router to determine the best next hop.
- An upper bound on the number of messages each router needs to send is the maximum distance between two routers in hops.
- The maximum distance is 15 hops for RIP.
- Since updates are sent only every 30 s, this results in a maximum delay of $15 \cdot 30\,\text{s} = 7.5\,\text{min}$.

Solution: Triggered Updates

- As soon as a router makes a change to its routing table, it immediately sends an update.
- This leads to a wave of updates through the network.
- Convergence time is reduced, but the network may be heavily loaded during updates.

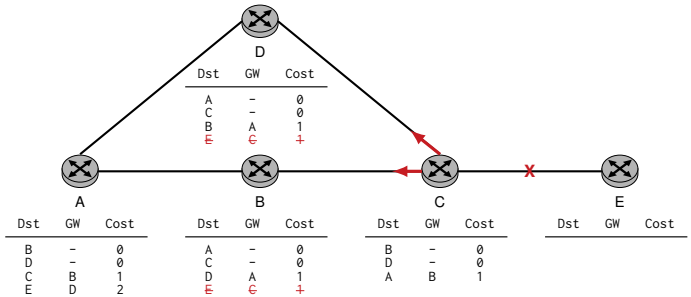
Another problem: Count to Infinity

- Link between C and E fails.
- Order in which updates are sent is left to chance.



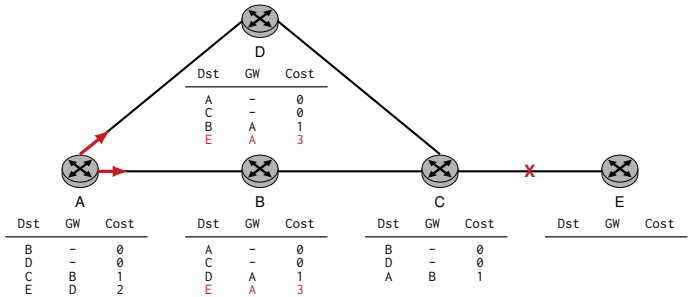
Another problem: Count to Infinity

- Link between C and E fails.
- Order in which updates are sent is left to chance.



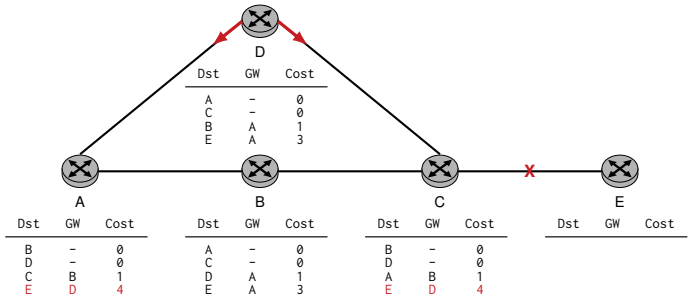
Another problem: Count to Infinity

- Link between C and E fails.
- Order in which updates are sent is left to chance.



Another problem: Count to Infinity

- Link between C and E fails.
- Order in which updates are sent is left to chance.



Depending on the order of updates

- the faulty route to E is propagated and
- the metric is always incremented
- until finally the hop count limit of 15 is reached.

This process is called Count to Infinity.

Solution attempts for Count-to-Infinity:

- Split Horizon
  - „Do not send a route to X to the neighbor from whom you learned the route.“
  - In the previous example, A would not send the route for E to its neighbor D, but it would send it to the other neighbor B.
  - Split Horizon improves the situation, but cannot solve the problem.
- Poison Reverse
  - Instead of no longer sending a route to X to the neighbor from which a route to X was learned, a route with infinite metric is sent.
  - In the previous example, A would send the route for E to its neighbor D with metric 15.
  - The erroneous route would still reach B.
  - Even Poison Reverse cannot completely solve the problem.
- Path Vector
  - For updates, send not only the target and cost, but also the full path by which the target is reached.
  - Before installing the route, each router checks whether it itself is already present in this path.
  - If yes, a loop is detected and this part of the update is discarded.
  - Path Vector prevents routing loops and thus Count to Infinity, but increases update messages and protocol complexity.

Distance Vector Protocols

- RIP (Routing Information Protocol)
  Very simple protocol, hop count as the only metric, suitable for a small number of networks, supported by most routers (even some home devices)
- IGRP (Interior Gateway Routing Protocol)
  Proprietary routing protocol from Cisco, supports more complex metrics than RIP
- EIGRP (Enhanced Interior Gateway Routing Protocol)
  Proprietary routing protocol from Cisco, successor to IGRP, significantly improved convergence properties.
- AODV (Ad hoc On-Demand Distance Vector)
  Deployment in wireless meshed networks, routes are not exchanged proactively but searched on-demand (reactive protocol)

Link State Protocols

- OSPF (Open Shortest Path First)
  Industry standard for medium to large number of networks
- IS-IS (Intermediate System to Intermediate System)
  Rarely used, powerful routing protocol, which is independent of IP, as it brings its own L3 protocol
- HWMP (Hybrid Wireless Mesh Protocol)
  Enables routing in IEEE 802.11s (Wireless Mesh Networks), but here routing decisions are made based on MAC addresses instead of IP addresses[4].

[4] MAC-based routing is a special case for (wireless) mesh networks, where not all nodes reach each other directly but still form a common broadcast domain

Autonomous Systems

All routing protocols presented so far

- determine best paths based on objective criteria (hopcount, bandwidth, delay, . . .),
- but offer no or only limited possibilities to directly influence routes.

However, sometimes it is desirable to select routes based on other criteria:

- Actual monetary costs incurred
- Networks / Countries through which traffic is routed to a destination.
- Infrastructure decisions (e.g., load on individual routers).

Routing decisions based on such criteria are called Policy-Based Routing.

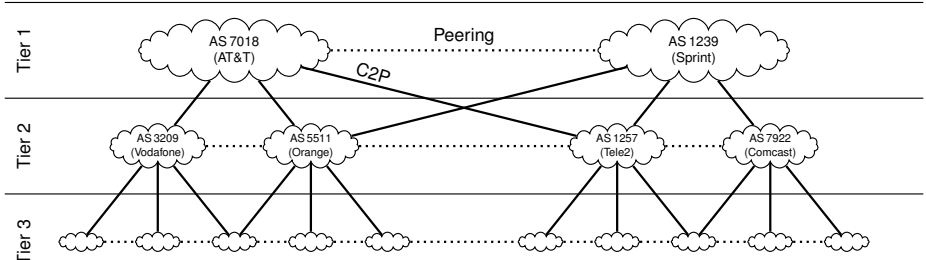
Autonomous System
A set of networks under unified administrative control is called an Autonomous System (AS). An AS is identified by a 16 bit or 32 bit identifier, the so-called AS number. A distinction is made when using routing protocols:

- Within an autonomous system, Interior Gateway Protocols (IGPs) such as RIP, OSPF, EIGRP or IS-IS are used.
- An Exterior Gateway Protocol (EGP) is used to exchange routes between Autonomous Systems.

The only EGP used in practice is Border Gateway Protocol (BGP).

Autonomous Systems

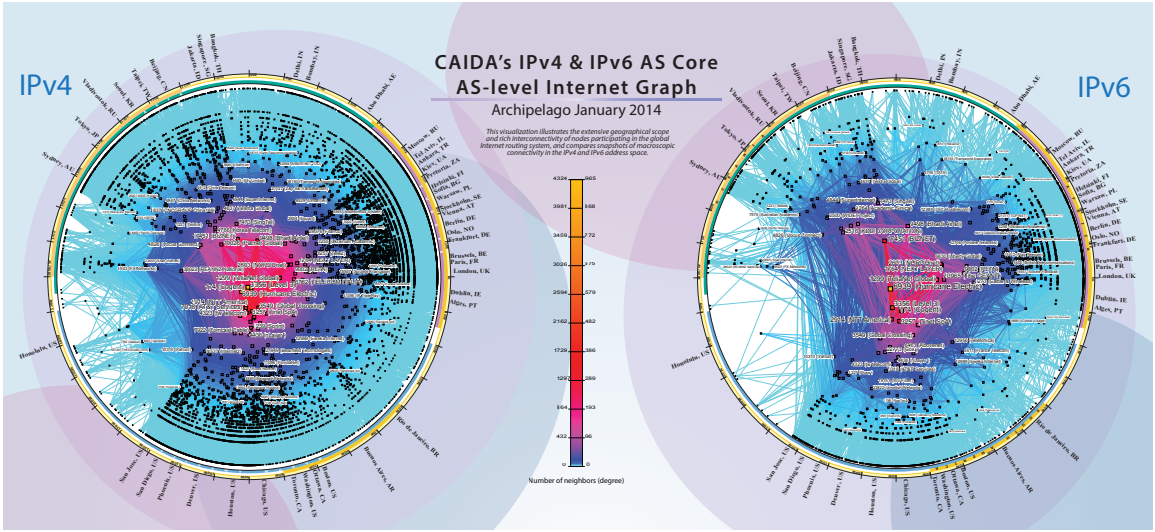
The Internet: Highly simplified schematic representation



- Autonomous systems may be interconnected by upstream providers or by peering.
- Numerous peering connections exist at Internet Exchange Points, e.g. DE-CIX.
- Peering connections are preferable to Customer Provider (C2P) connections for cost reasons.
- The border routers of an AS „announce“ prefixes reachable through this AS.
- AS 5511 announced its own prefixes as well as those of its customers to its peers and upstream providers.
- AS 5511 would not announce Vodafone's networks to Tele2, Sprint or AT&T.

Rule of thumb: For vertical connections, the respective customer („smaller provider“) must pay, which is why horizontal connections (peerings) are preferred.

The Internet Somewhat less simplified [9]



Chapter 3: Network Layer

Encryption on layer 3: IPSec

IPsec

We will look at IPSec in a dedicated security chapter.

Chapter 3: Network Layer

Summary

In this chapter we have

- discussed the advantages of packet switching over circuit and message switching,
- recognized the need for logical addresses for end-to-end addressing,
- learned about the two most important protocols for end-to-end addressing on the Internet,
- covered methods for further logical subdivision of networks into subnets and prefixes and
- developed a basic understanding of how routing information is exchanged on the Internet.

We should now know,

- what the differences between circuit, message and packet switching are,
- what the technical and logical difference between layer 2 and 3 addresses is,
- which possibilities IPv4 and IPv6 offer for logical structuring and how this works,
- how to determine the corresponding link layer address for a given address on layer 3,
- what a routing table is,
- how routers make forwarding decisions,
- what the difference is between routing and forwarding,
- how routers exchange routes with each other
- what types of routing protocols are available and how they work.

References

[1] Google IPv6 Statistiken, 2017.
https://www.google.de/ipv6/statistics.html#tab=ipv6-adoption&tab=ipv6-adoption.

[2] CAIDA.
IPv4 Census Map.
http://www.caida.org/research/id-consumption/#ipv4-census-map.

[3] A. Conta and S. Deering.
Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification, 2006.
http://tools.ietf.org/html/rfc4443.

[4] M. Cotton, L. Vegoda, R. Bonica, and B. Haberman.
Special-Purpose IP Address Registries, 2013.
http://tools.ietf.org/html/rfc6890.

[5] M. Crawford.
Transmission of IPv6 Packets over Ethernet Networks, 2007.
https://tools.ietf.org/html/rfc2464.

[6] S. Deering and R. Hinden.
Internet Protocol, Version 6 (IPv6) Specification, 1995.
http://tools.ietf.org/html/rfc1883.

[7] S. Deering and R. Hinden.
Internet Protocol, Version 6 (IPv6) Specification, 1998.
http://tools.ietf.org/html/rfc2460.

[8] C. Hedrick.
Routing Information Protocol, 1988.
http://tools.ietf.org/html/rfc1058.

[9] B. Huffaker, Y. Hyun, and M. Luckie.
IPv4 and IPv6 AS Core: Visualizing IPv4 and IPv6 Internet Topology at a Macroscopic Scale in 2014, 2014.
http://www.caida.org/research/topology/as_core_network/2014.

References

[10] S. Kawamura and M. Kawashima.
A Recommendation for IPv6 Address Text Representation, 2010.
http://tools.ietf.org/html/rfc5952.

[11] G. Malkin.
RIP Version 2, 1998.
http://tools.ietf.org/html/rfc2453.

[12] T. Narten, R. Draves, and S. Krishnan.
Privacy Extensions for Stateless Address Autoconfiguration in IPv6, 2007.
https://tools.ietf.org/html/rfc4941.

[13] E. Nordmark, W. Simpson, and H. Soliman.
Neighbor Discovery for IP version 6 (IPv6), 2007.
http://tools.ietf.org/html/rfc4861.

[14] L. L. Peterson and S. D. B.
Computer Networks – A System Approach, chapter Internetworking, pages 234–242.
Elsevier, 4. edition, 2007.

[15] L. L. Peterson and S. D. B.
Computer Networks – A System Approach, chapter Internetworking, pages 248 – 256.
Elsevier, 4. edition, 2007.

[16] L. L. Peterson and S. D. B.
Computer Networks – A System Approach, chapter Internetworking, pages 259 – 262.
Elsevier, 4. edition, 2007.

[17] S. Thomson, T. Narten, and T. Jinmei.
IPv6 Stateless Address Autoconfiguration, 2007.
http://tools.ietf.org/html/rfc2462.