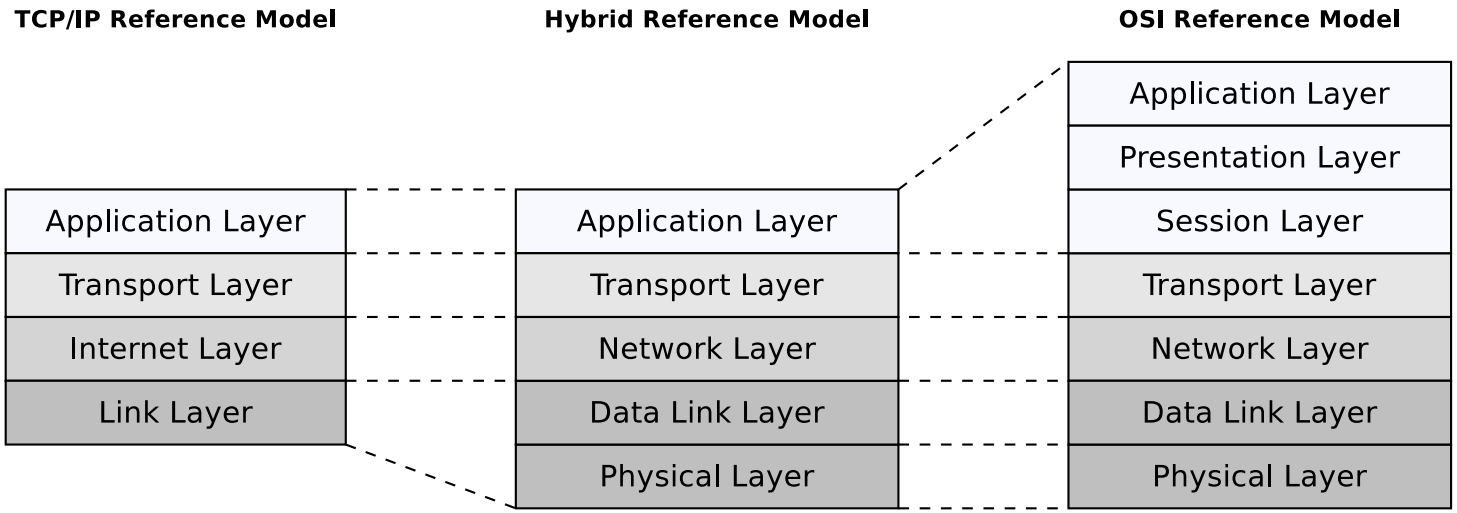
7th Slide Set
Computer Networks

Prof. Dr. Christian Baun

Frankfurt University of Applied Sciences
(1971–2014: Fachhochschule Frankfurt am Main)
Faculty of Computer Science and Engineering
christianbaun@fb2.fra-uas.de

Network Layer

- Functions of the Network Layer
  - Sender: Pack segments of the Transport Layer into pakets
  - Receiver: Identify the packets inside the frames of Data Link Layer
  - Provide logical addresses (IP addresses)
  - Determine the best path to the destination = Routing
  - Forward packets between logical networks (across different physical networks)



Exercise sheet 4 repeats the contents of this slide set which are relevant for these learning objectives

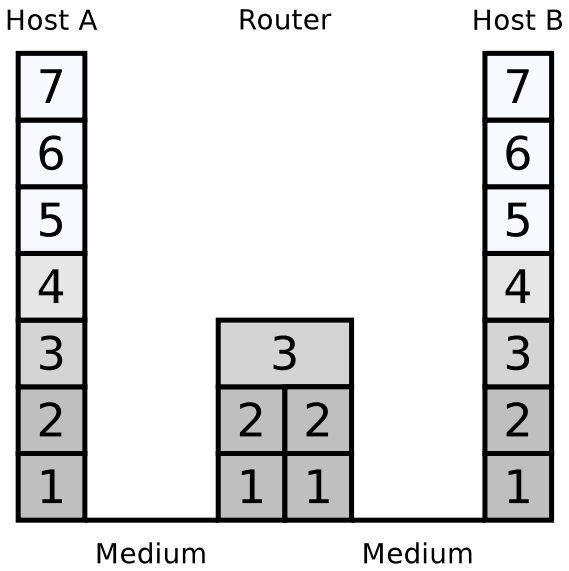
- Devices: Router, Layer-3-Switch (Router without WAN port)
- Protocols: IPv4, IPv6, ICMP, IPX/SPX, DECnet

Learning Objectives of this Slide Set

- Network Layer (part 1)
  - Devices of the Network Layer
    - Router
    - Impact on the collision domain
    - Broadcast domain
  - Addressing in the Network Layer
    - Format of IP addresses
    - Address classes, network identifier and host identifier, subnets and subnet mask
    - Private IP addresses
    - Format of IP packets
    - Fragmenting IP packets
  - Diagnosis information and error messages via ICMP

Router, Layer-3-Switch and Gateway

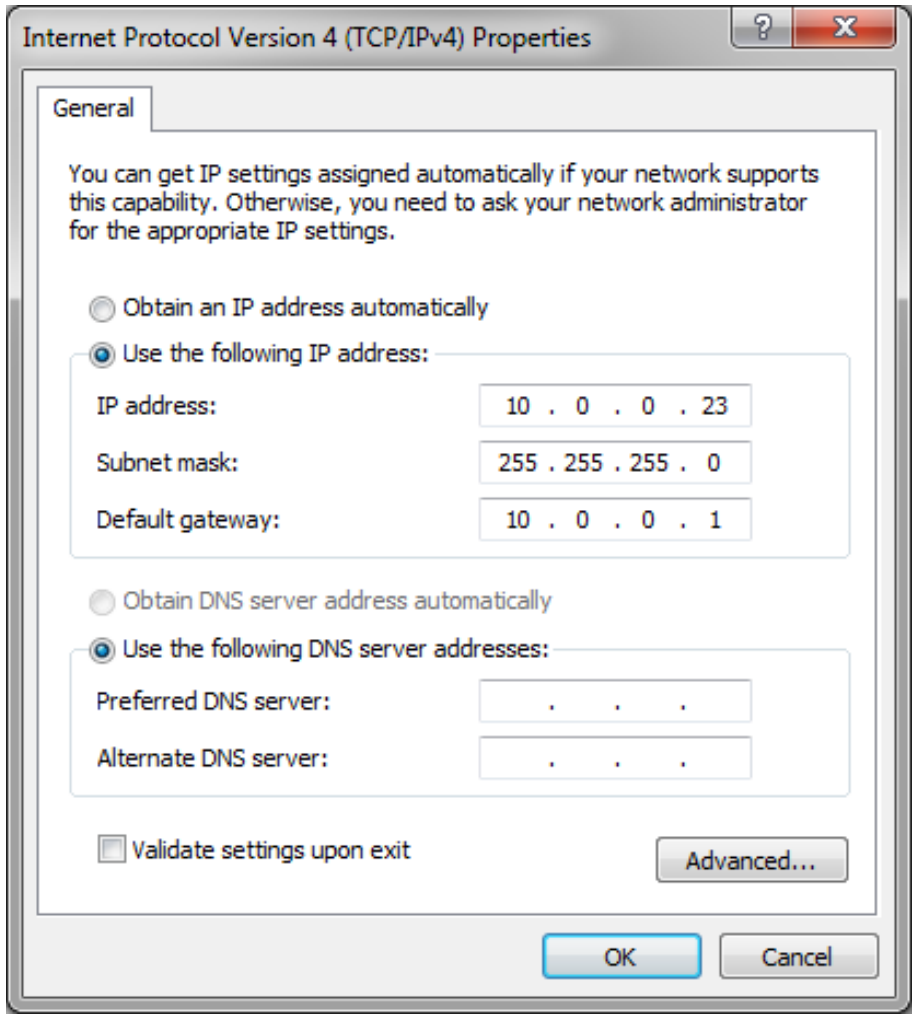
- **Routers** forward packets between networks with different logical address ranges
  - Provide exactly like Hubs and Switches multiple interfaces
  - Enable to connect the local network (LAN) with a WAN (e.g. via DSL or 3G/4G mobile network)
- **Layer-3-Switches** are Routers without a WAN interface
- **Gateways** are protocol converters
  - Enable communication between networks, which base on different protocols
  - A Gateway can in theory operate on all layers
  - Gateways, which operate on the Network Layer, are also called **Multiprotocol Routers**



The two pictures below show a Linksys WRT54GL Wireless-G Wireless Router with a WAN port and a 4-port switch

Gateways (1/2)

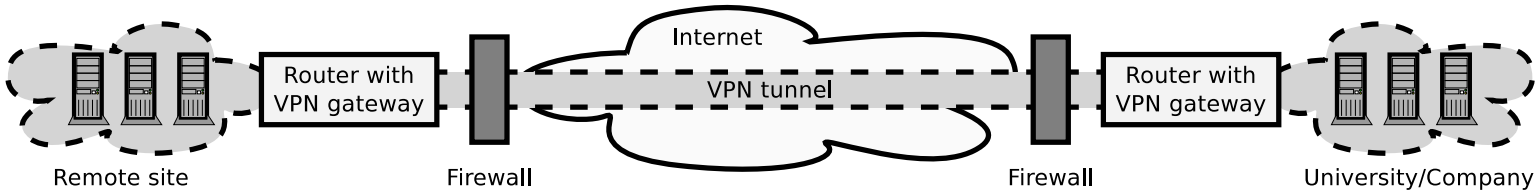
- Modern computer networks operate almost exclusively with the Internet Protocol (IP)
  - For this reason, a protocol conversion on the Network Layer is mostly not necessary
- In the past, in the network preferences of a terminal device, the IP address of the Gateway was specified as **Default Gateway**
  - Today, this field contains the Router address, because a Gateway is usually not required any longer
    - Thus, the term **Default Router** would be suited better



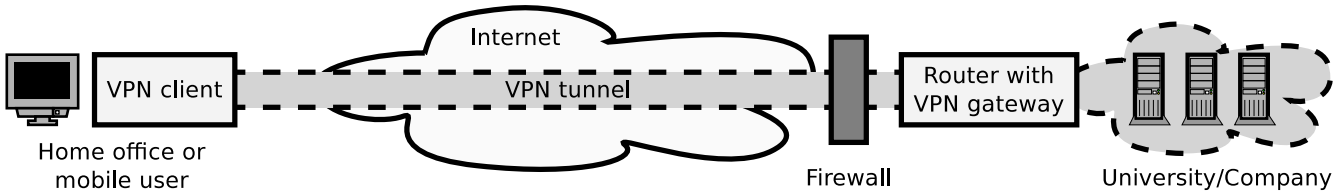
Gateways (2/2)

- VPN-Gateways (Virtual Private Network) may operate on Network Layer
  - They provide secure access to remote protected networks (e.g. intranet of a university or a company) over insecure public networks
  - Services (e.g. Email), which are only available inside the protected network can be used via a tunneled connection

Site-to-Site VPN

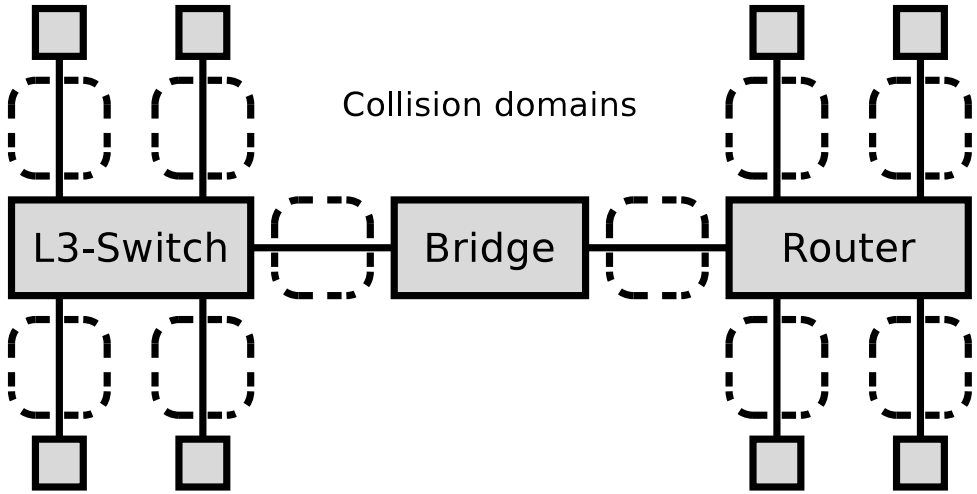


Remote Access VPN = End-to-Site VPN



Collision Domain – Routers and Layer-3-Switches

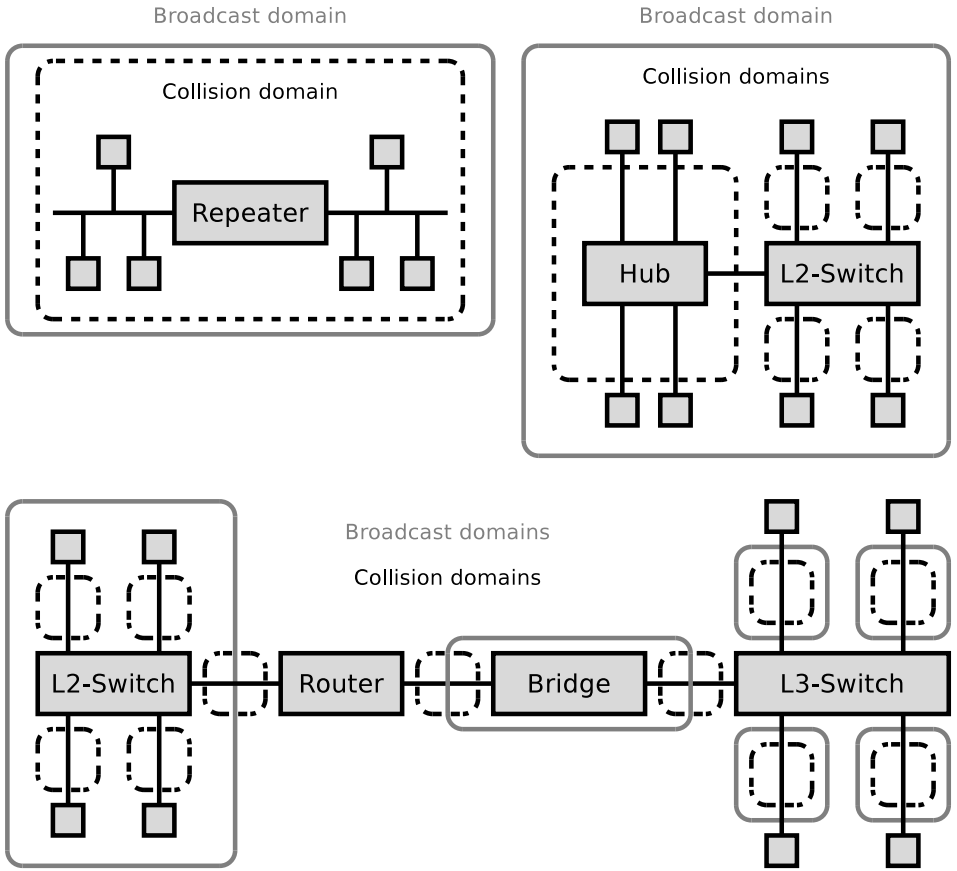
- Routers and Layer-3-Switches divide the collision domain
  - Exactly like Bridges and Layer-2-Switches do



- Devices, which operate on layer 1 (**Repeaters**, **Hubs**) do not divide the collision domain
- Devices, which operate on layer 2 and 3 (**Bridges**, **Layer-2-Switches**, **Routers**, **Layer-3-Switches**) divide the collision domain

Broadcast Domain (1/2)

- Logical part of a computer network, where a broadcast reaches all network devices that belong to that part
  - Devices, which operate on layer 3 (**Routers**) divide the broadcast domain
  - Devices, which operate on layer 1 and 2 (**Repeaters**, **Hubs**, **Bridges**, **Layer-2-Switches**) do not divide it
    - From the perspective of logical networks, they work transparent

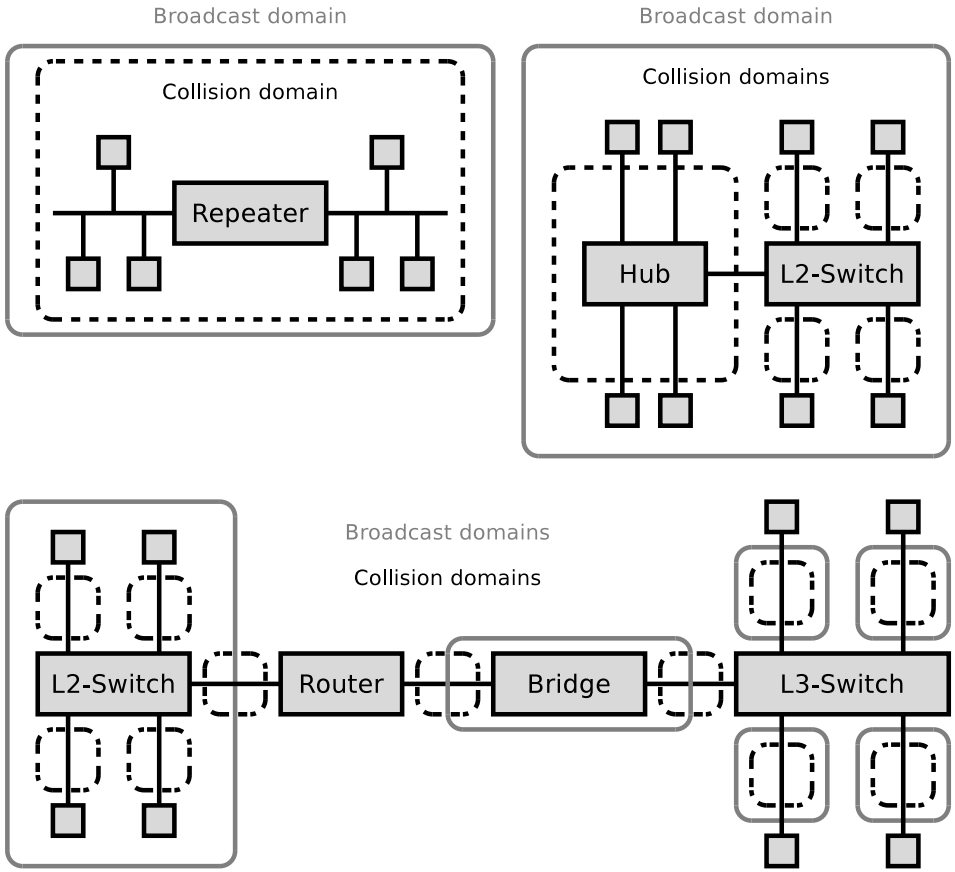


The technical term broadcast domain. . .

always applies to the Network Layer and never to the Data Link Layer (although broadcasts exist also in the Data Link Layer)

Broadcast Domain (2/2)

- Broadcast domains consist of one or multiple collision domains
- Routers operate on the Network Layer (layer 3)
  - This means, that each port of a Router is connected to a different IP network
    - This information is necessary for the calculation of the required number of subnets
  - Multiple Hubs, Switches, Repeaters or Bridges can operate in the same IP subnet
    - But it is impossible to connect an IP subnet to multiple ports of a Router

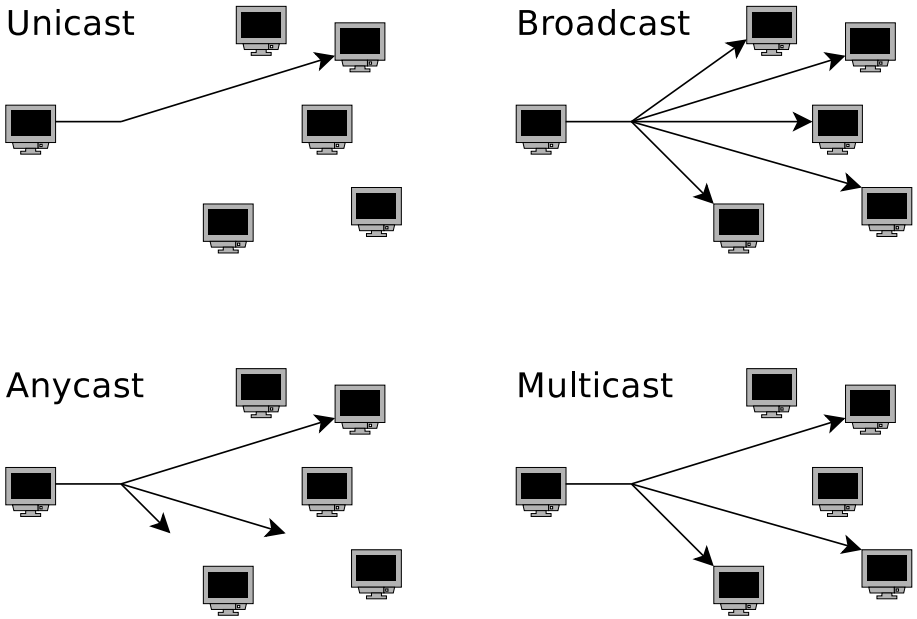


Addressing in the Network Layer (1/2)

- Using only physical addressing via MAC addresses is not useful in large-scale computer networks with possibly global proportions
  - Reason: Maintainability
- **Logical addresses** are required, which are independent from the specific hardware
  - Logical addressing separates the view of humans (logical addresses) from the internal view of computers and software (physical addresses)

Addressing in the Network Layer (2/2)

- Every Network Layer packet contains the IP address of the receiver
  - The structure of IP addresses is specified by the Internet Protocol (IP)



- An IP address can be assigned to a single receiver (**unicast**) or a group of receivers (**multicast** or **broadcast**)
- Multiple IP addresses can be assigned to a single network device

- If **Anycast** is used, a single device of a group of devices can be reached via a single address
  - The receiver, which can be accessed via the shortest route, responds

Multicast ist used for example by the routing protocols RIPv2 and OSPF (see slide set 8) and by Network Time Protocol – NTP (see slide set 10) that is used for clock synchronization
Anycast is used for example by some Root Name Servers in the Domain Name System (see slide set 10)

Format of IP Addresses

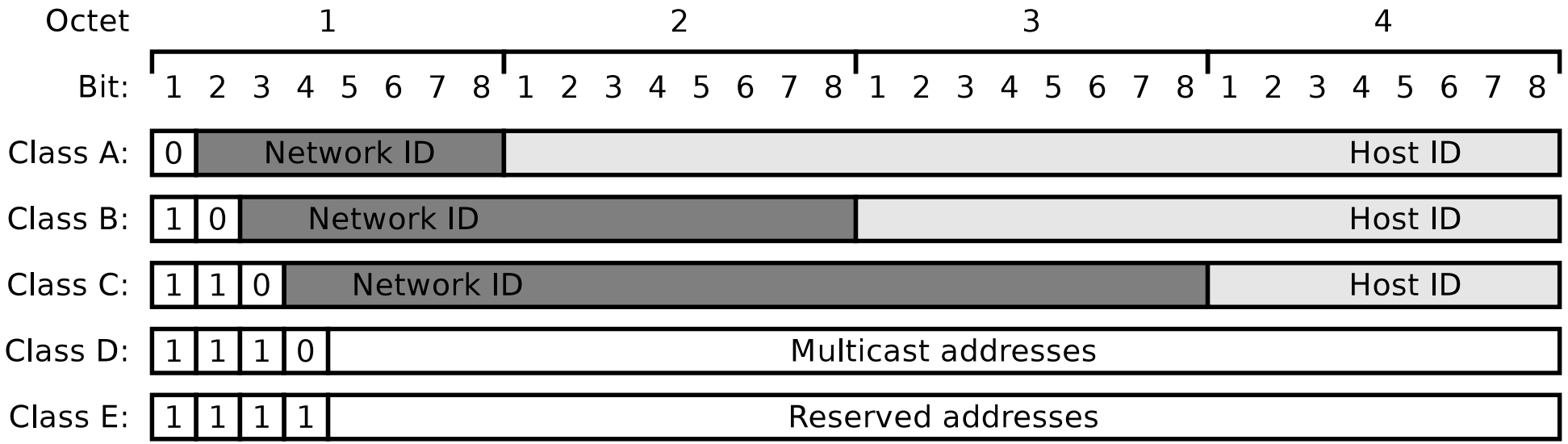
- IPv4 addresses have a length of 32 bits (4 bytes)
  - Thus, the address space contains $2^{32} = 4,294,967,296$ possible addresses

Address space = amount of all valid network identifiers

- The usual representation is the so-called **dotted decimal notation**
  - The 4 octets are written as decimal integers in the range from 0 to 255, which are separated from each other by points
    Example: 141.52.166.25

Address Classes, Network Identifier and Host Identifier

- Originally, IPv4 addresses were categorized into classes from A to C
  - Additionally, the classes D and E for special purposes existed
- A 32 bits long IPv4 address consists of 2 fields:
  - **Network identifier** (network ID)
  - **Host identifier** (host ID)
  - Class A: 7 bits for the network ID and 24 bits for the host ID
  - Class B: 14 bits for the network ID and 16 bits for the host ID
  - Class C: 21 bits for the network ID and 8 bits for the host ID



Address Classes (1/2)

- The prefixes specify the address classes and their address ranges

Class	Prefix	Address range	Network ID	Host ID
A	0	0.0.0.0 - 127.255.255.255	7 bits	24 bits
B	10	128.0.0.0 - 191.255.255.255	14 bits	16 bits
C	110	192.0.0.0 - 223.255.255.255	21 bits	8 bits
D	1110	224.0.0.0 - 239.255.255.255	—	—
E	1111	240.0.0.0 - 255.255.255.255	—	—

- $2^7 = 128$ class A networks with a maximum of $2^{24} = 16,777,216$ host addresses each
- $2^{14} = 16,384$ class B networks with a maximum of $2^{16} = 65,536$ host addresses each
- $2^{21} = 2,097,152$ class C networks with a maximum of $2^8 = 256$ host addresses each
- Class D contains multicast addresses (e.g. for IPTV)
- Class E is reserved for future (?!) purposes and experiments

Why is the class E address space of IPv4 not used?

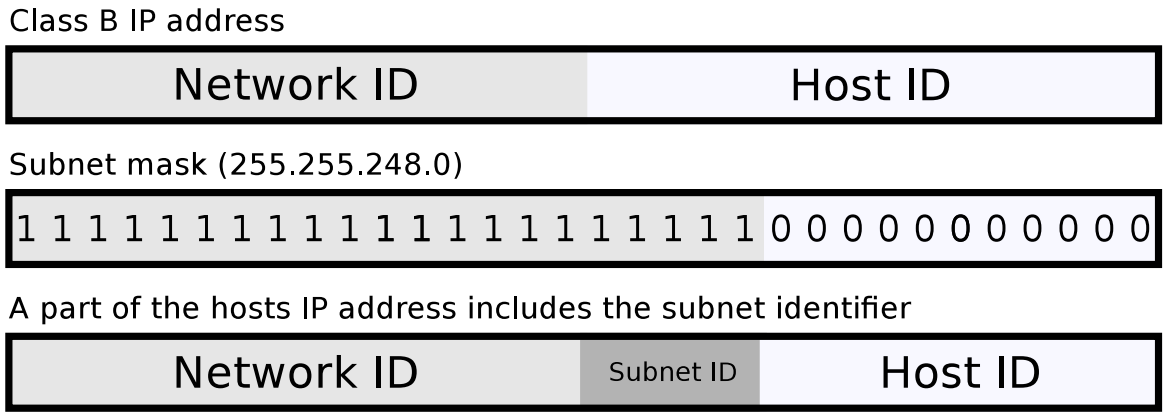
"The class E space has 268 million addresses and would give us in the order of 18 months worth of IPv4 address use. However, many TCP/IP stacks, such as the one in Windows, do not accept addresses from class E space and will not even communicate with correspondents holding those addresses. It is probably too late now to change this behavior on the installed base before the address space would be needed."

Source: `http://www.cisco.com/web/about/ac123/ac147/archived_issues/ipj_10-3/103_addr-cons.html`

Address Classes (2/2)

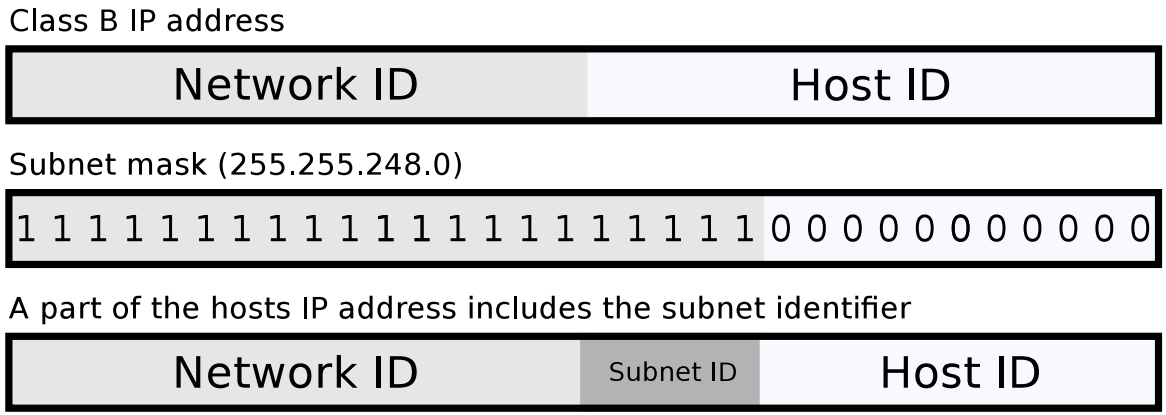
- Only the classes A, B and C are relevant in practice
- The original intention was to identify physical networks in an unique way via the network ID
  - This approach causes some drawbacks
- **Drawbacks of Address Classes:**
  - It is impossible to dynamically adjust them
  - Many addresses are wasted
    - A class C network with 2 devices wastes 253 addresses
    - The address space of class C networks is quite small
    - A class B network with 256 devices wastes $>$ 64,000 addresses
    - Only 128 class A networks exist
    - Migrating multiple devices to a different network class is complex task
- Solution: Logical networks are divided into **subnets**
  - 1993: Introduction of the **Classless Interdomain Routing** (CIDR)

Subnet Mask (1/2)



- For creating subnets, a **(sub-)netmask** is required
  - All hosts in a network have a subnet mask assigned
    - Length: 32 bits (4 bytes)
    - It is used to specify the number of subnets and hosts
- The subnet mask splits the host ID of an IP address into **subnet ID** and **host ID**
  - The network ID remains unchanged
  - The network mask adds another level of hierarchy into the IP address

Subnet Mask (2/2)



- Structure of the subnet mask:
  - 1-bits indicate, which part of the address space is used for subnet IDs
  - 0-bits indicate, which part of the address space is used for host IDs
- Example: Splitting a class B network into 20 subnets requires 5 bits
  - Each subnet requires its own subnet ID and it must be represented in binary form
  - If 5 bits are used for the representation of the subnet IDs, 11 bits remain for host IDs

Syntax of the Classless Interdomain Routing (CIDR)

- Since **CIDR** was introduced in 1993, IP address ranges are assigned in this notation: `First address/mask bits`
  - The number of mask bits indicates the number of 1-bits (prefix) in the subnet mask
- The table shows the possible splits of a class C network into subnets

Mask bits (prefix)	/24	/25	/26	/27	/28	/29	/30	/31	/32
Subnet mask	0	128	192	224	240	248	252	254	255
Subnet bits	0	1	2	3	4	5	6	7	8
Subnets IDs	1	2	4	8	16	32	64	128	256
Host bits	8	7	6	5	4	3	2	1	0
Host IDs	256	128	64	32	16	8	4	2	—
Hosts (maximum)	254	126	62	30	14	6	2	0	—

Not all Addresses can or should be used

Mask bits (prefix)	/24	/25	/26	/27	/28	/29	/30	/31	/32
Subnet mask	0	128	192	224	240	248	252	254	255
Subnet bits	0	1	2	3	4	5	6	7	8
Subnets IDs	1	2	4	8	16	32	64	128	256
Host bits	8	7	6	5	4	3	2	1	0
Host IDs	256	128	64	32	16	8	4	2	—
Hosts (maximum)	254	126	62	30	14	6	2	0	—

2 Host IDs cannot be assigned to network devices, because each (sub-)network requires...

- an address for the network itself (all host ID bits are 0 bits)
- a broadcast address to address all devices in network (all bits of the host ID are 1 bits)

2 subnet IDs should not be used

- The subnet IDs, consisting exclusively of 0 bits and 1 bits should not be used
  $\Longrightarrow$ This rule is obsolete, but still often followed
- Modern Routers and network software have no problem, when all possible subnet IDs are assigned to subnets

Determining the necessary Subnets Bits

Mask bits (prefix)	/24	/25	/26	/27	/28	/29	/30	/31	/32
Subnet mask	0	128	192	224	240	248	252	254	255
Subnet bits	0	1	2	3	4	5	6	7	8
Subnets IDs	1	2	4	8	16	32	64	128	256
Host bits	8	7	6	5	4	3	2	1	0
Host IDs	256	128	64	32	16	8	4	2	—
Hosts (maximum)	254	126	62	30	14	6	2	0	—

- By using the table, it is simple to determine the required bits for subnets
- Example: Subdivide a class C network into 5 subnets, each with a maximum of 25 hosts
  - Each subnet requires a subnet address
  - For representing 5 subnets, 3 subnet bits are required
  - The remaining 5 bits are used for representing the host IDs and they allow the addressing of $32 - 2 = 30$ hosts per subnet
  - Thus, the subnet mask with the prefix /27 is well suited for this use case

Calculation example for Subnetting

- Example: 172.21.240.90/27 is a class B address ($\Longrightarrow$ see prefix)
  - The number behind the slash is the number of 1 bits in the subnet mask
- **IP address `AND` subnet mask = subnet address**

```
1 AND 1 = 1,  1 AND 0 = 0,  0 AND 1 = 0,  0 AND 0 = 0
```

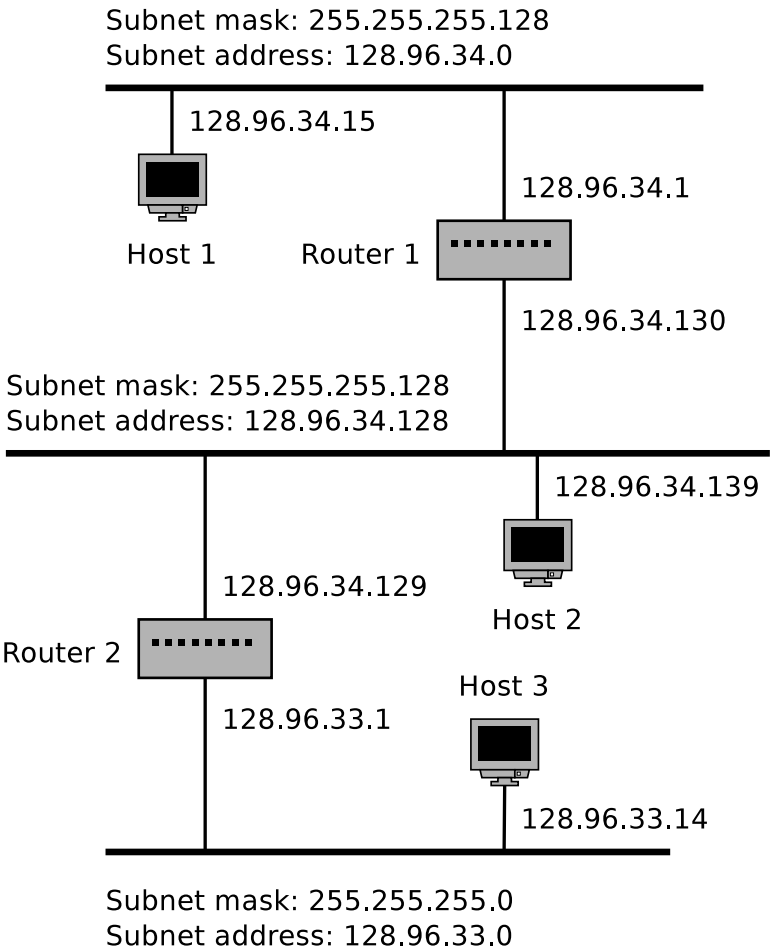
IP address	172.21.240.90	10101100 00010101 11110000 01011010
Subnet mask	255.255.255.224	11111111 11111111 11111111 11100000
Subnet address	172.21.240.64	~~10101100 00010101~~ 11110000 010~~00000~~
Subnet ID	1922	~~10101100 00010101~~ 11110000 010~~00000~~

- **IP address `AND` (`NOT` subnet mask) = host ID**

IP address	172.21.240.90	10101100 00010101 11110000 01011010
Subnet mask	255.255.255.224	11111111 11111111 11111111 11100000
Inverse subnet mask	000.000.000.31	~~00000000 00000000 00000000 000~~11111
Host ID	26	~~00000000 00000000 00000000 000~~11010

- /27 and class B prefix $\Longrightarrow$ 11 bits for the subnet ID
  - 5 bits and therefore $2^5 = 32$ addresses remain for the host IDs
    - 30 of these addresses can be assigned to network devices

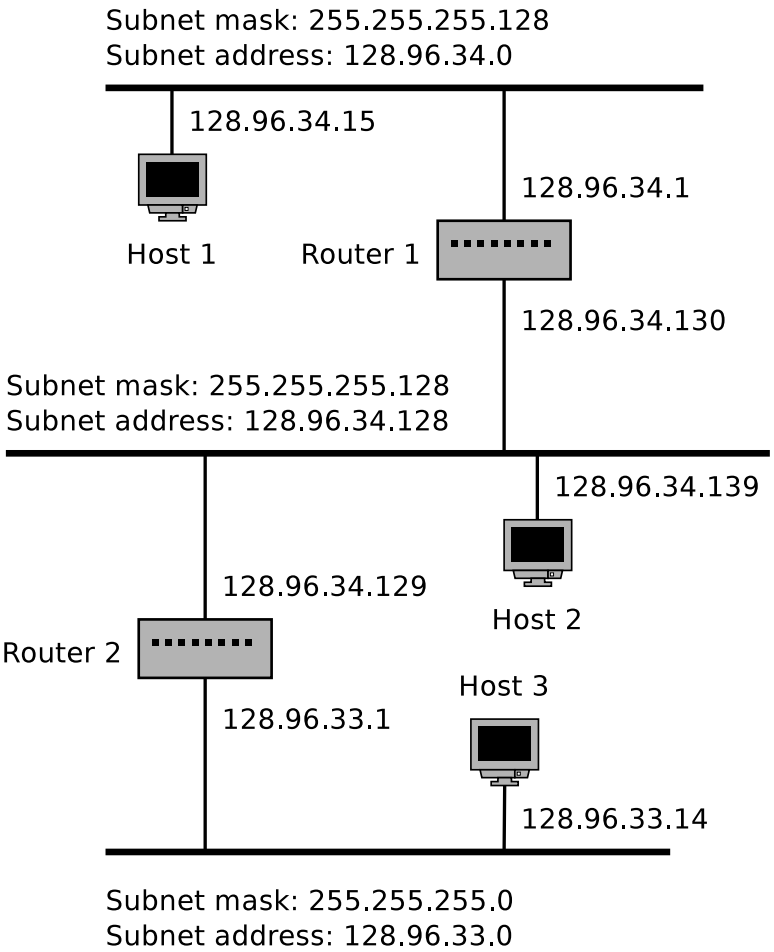
Example (1/4)



Source: Computernetzwerke. Peterson and Davie. dpunkt (2000)

- All hosts inside the same subnet have the same subnet mask
- IP address `AND` subnet mask = subnet address
- If a host wants to transmit a packet, it calculates the `AND` of its own subnet mask and the destination IP address
  - If the result is equal to the subnet address of the sender, the sender learns that the destination is inside the same subnet
  - If the result does not match the subnet address of the sender, the packet must be transmitted to a Router, which forwards it to another subnet

Example (2/4)



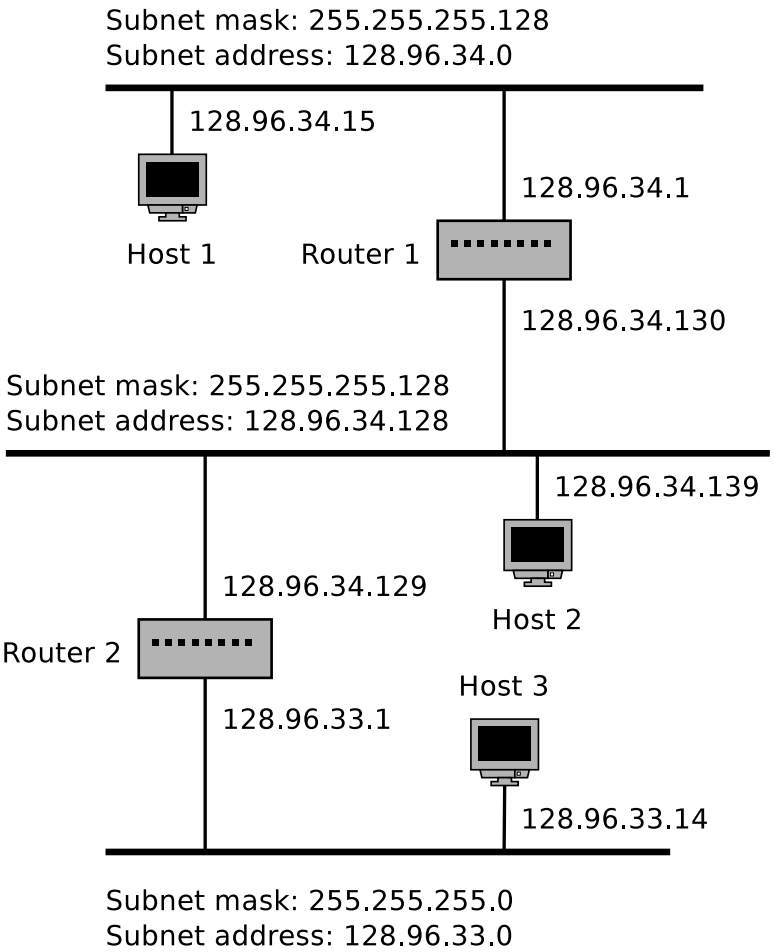
Source: Computernetzwerke. Peterson and Davie. dpunkt (2000)

- Example: Host 1 transmits a packet to host 2 (128.96.34.139)
- Host 1 calculates subnet mask (255.255.255.128) `AND` destination address (128.96.34.139). Result: 128.96.34.128
- This is not the subnet of host 1
  $\Longrightarrow$ Host 2 is in a different subnet
- Host 1 transmits the packet to its default Router (128.96.34.1)
- Entries in the routing table of Router 1

Subnet address	Subnet mask	Next hop
128.96.34.0	255.255.255.128	Port 0
128.96.34.128	255.255.255.128	Port 1
128.96.33.0	255.255.255.0	Router 2

- Routing protocols/algorithms ($\Longrightarrow$ see slide set 8) create and maintain the entries in the routing tables inside the Routers

Example (3/4)



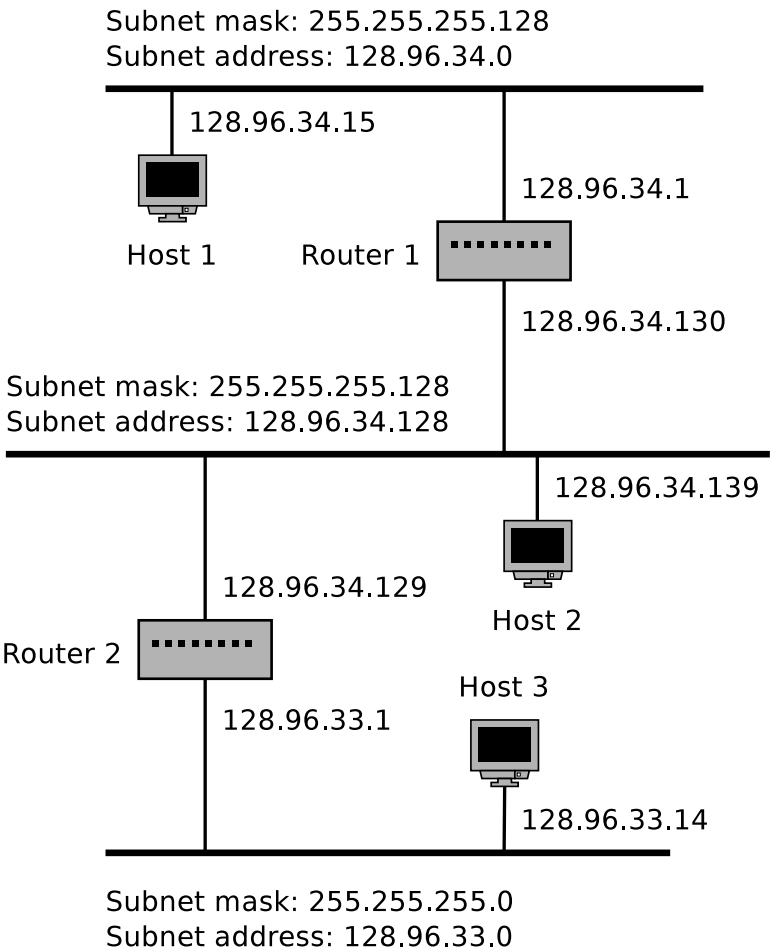
Source: Computernetzwerke. Peterson and Davie. dpunkt (2000)

- Entries in the routing table of Router 1

Subnet address	Subnet mask	Next hop
128.96.34.0	255.255.255.128	Port 0
128.96.34.128	255.255.255.128	Port 1
128.96.33.0	255.255.255.0	Router 2

- The Router calculates the destination address `AND` subnet mask for each entry (row)
- If the result is equal to the subnet address of one entry, the Router forwards the packet to the corresponding Router or port
- Router 1 calculates for the 1st row: Host 2 (128.96.34.139) `AND` subnet mask (255.255.255.128) $\Longrightarrow$ 128.96.34.128
- This result does not match the subnet address (128.96.34.0) inside the routing table

Example (4/4)



Source: Computernetzwerke. Peterson and Davie. dpunkt (2000)

- Entries in the routing table of Router 1

Subnet address	Subnet mask	Next hop
128.96.34.0	255.255.255.128	Port 0
128.96.34.128	255.255.255.128	Port 1
128.96.33.0	255.255.255.0	Router 2

- Router 1 calculates for the 2nd row: Host 2 (128.96.34.139) AND subnet mask (255.255.255.128) $\Longrightarrow$ 128.96.34.128
- This result is equal to the subnet address entry in the forwarding table
  $\Longrightarrow$ The 2nd row is a hit
- Router 1 transmits the packet via port 1 to host 2, because this port is connected to the same network as host 2

Where do the forwarding table records come from?

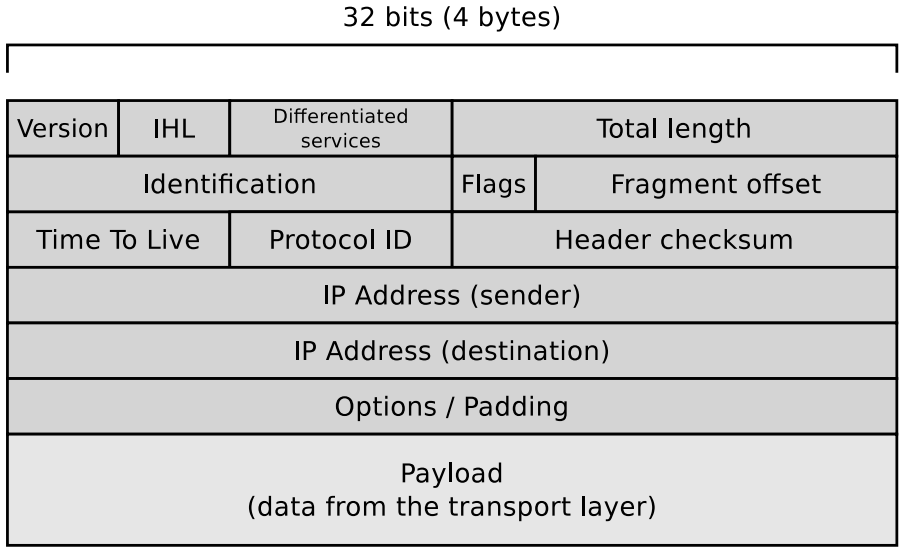
The forwarding table records are created via path determination (**routing**) using **routing protocols**
$\Longrightarrow$ see slide set 8

Private Networks – Private IP Address Spaces

- In private networks, it is also required to assign IPs to network devices
  - These addresses are not allowed to interfere with global accessible internet services
- Several address spaces exist, containing private IP addresses
  - These address spaces are **not routed** in the internet

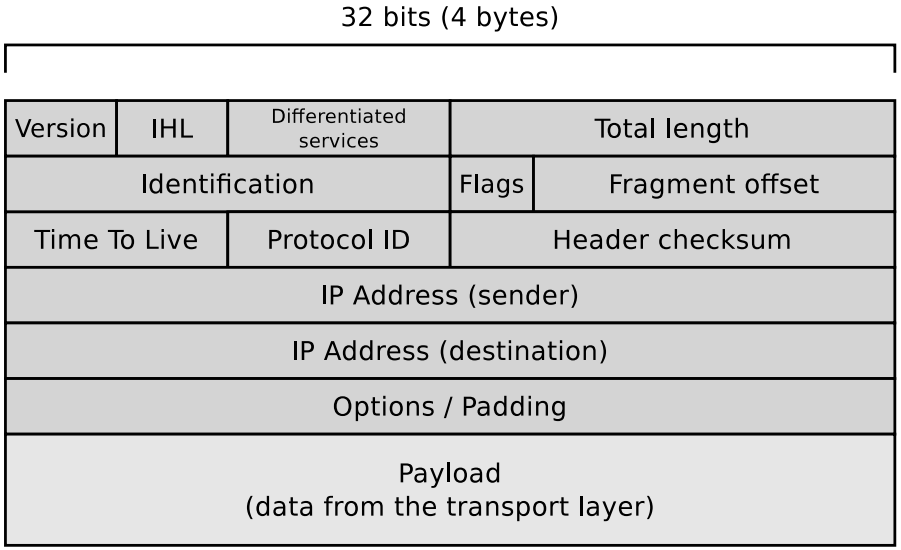
Address space:	10.0.0.0 to 10.255.255.255
CIDR notation:	10.0.0.0/8
Number of addresses:	$2^{24} = 16{,}777{,}216$
Address class:	Class A. 1 private network with 16,777,216 addresses
Address space:	172.16.0.0 to 172.31.255.255
CIDR notation:	172.16.0.0/12
Number of addresses:	$2^{20} = 1{,}048{,}576$
Address class:	Class B. 16 private networks with 65,536 addresses each
Address space:	192.168.0.0 to 192.168.255.255
CIDR notation:	192.168.0.0/16
Number of addresses:	$2^{16} = 65{,}536$
Address class:	Class C. 256 private networks with 256 addresses each

Structure of IPv4 Packets (1/6)



- **Version** = (4 bits)
  - Version = 4 $\Longrightarrow$ IPv4
  - Version = 6 $\Longrightarrow$ IPv6
- IP **H**eader **L**ength (4 bits)
  - Header length, represented as the number of 4 byte words
    - Example: IHL = 5 $\Longrightarrow$ 5 $*$ 4 bytes = 20 bytes
  - Indicates where the payload begins
- **Differentiated services** (8 bits)
  - Prioritization of IP packets is possible with this field (Quality of Service)
  - The field slightly changed over the years (RFC 791, RFC 2474, RFC 3168)

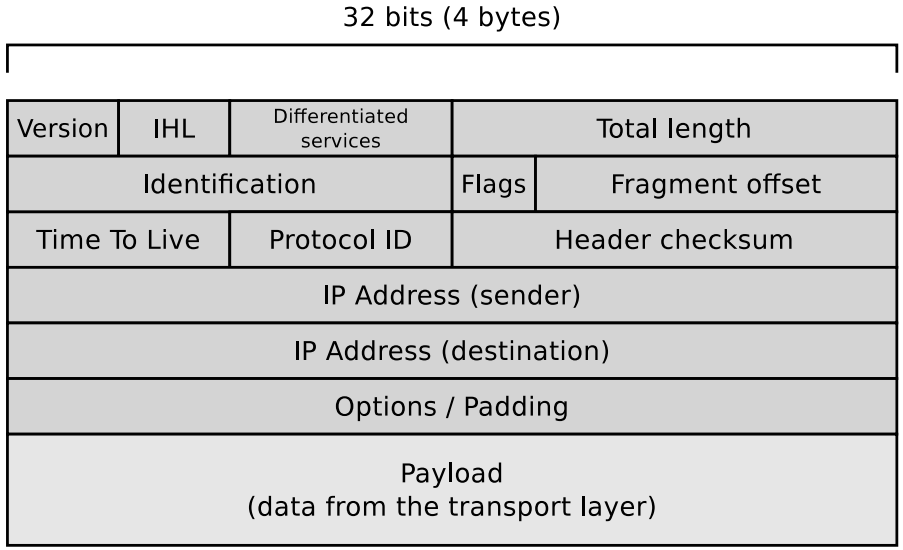
Structure of IPv4 Packets (2/6)



- **Total length** (16 bits)
  - This field defines the entire packet size (header and payload)
  - This length of the field is 16 bits and therefore the maximum possible IPv4 packet length is 65,535 bytes

Structure of IPv4 Packets (3/6)

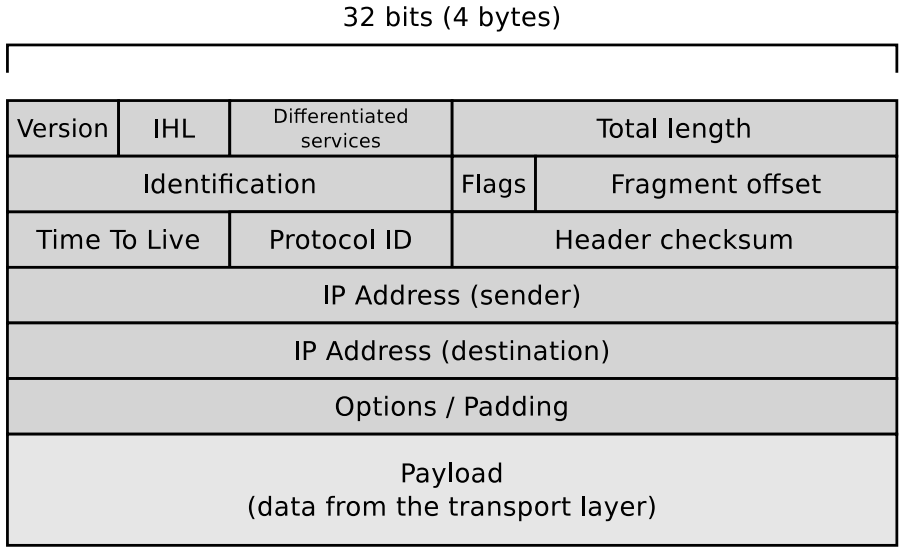
- The fields **Identification**, **Flags** and **Fragment offset** control the assembly of fragmented IP packets
- **Identification** (16 bits)
  - Contains a unique identifier of the IP packet



- **Flags** (3 bits)
  - Here the sender informs whether the packet can be fragmented and the receiver is informed whether more fragments follow
- **Fragment Offset** (13 bits)
  - Contains a number which states for fragmented packets, from which position of the unfragmented packet the fragment begins

More information about the fragmentation of IP packets provide the slides 33 + 34

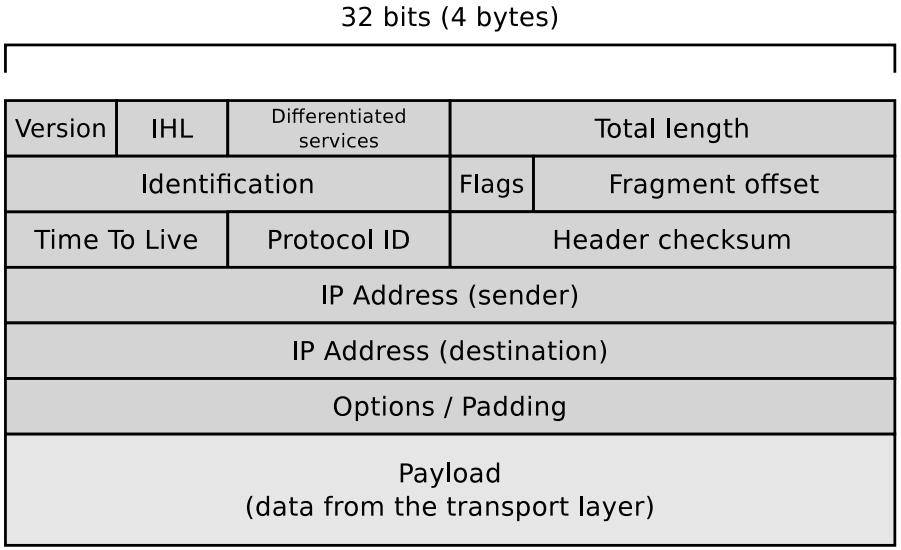
Structure of IPv4 Packets (4/6)



- **Time To Live** (8 bits)
  - Contains the maximum number of hops
    - Each Router on the route to the destination decrements the value by one
  - Prevents that undeliverable IP packets endlessly go in cycles in the network

Structure of IPv4 Packets (5/6)

- **Protocol ID** (8 bits)
  - Contains the number of the Transport Layer protocol used
  - TCP segments $\Longrightarrow$ 6
  - UDP segments $\Longrightarrow$ 17
  - ICMP message $\Longrightarrow$ 1
  - OSPF message $\Longrightarrow$ 89

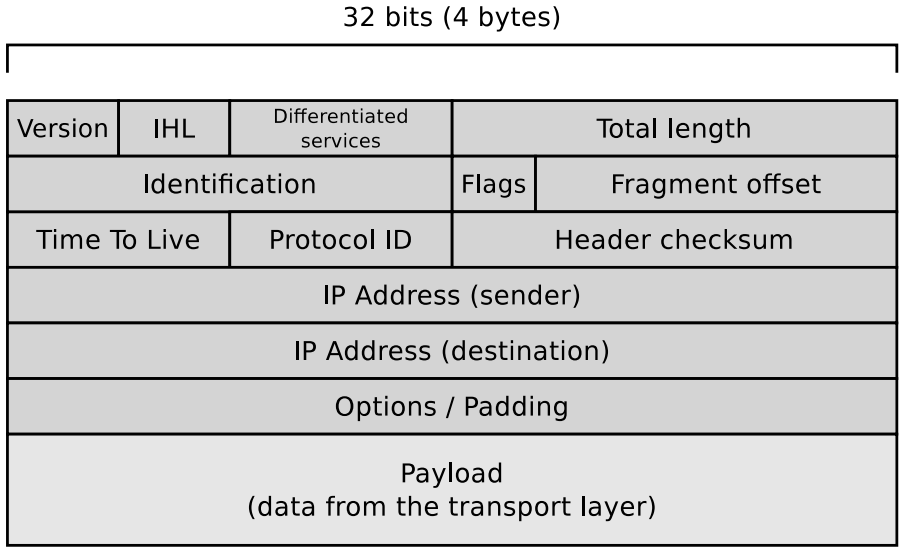


- Each IPv4 packet contains a checksum (16 bits) of the header
  - Because at each Router on the way to the destination, the content of the field **Time To Live** changes, each Router needs to verify the checksum, recalculate and insert it into the header

Routers usually ignore the checksum to speed up the packet forwarding

Therefore, IPv6 packets contain no checksum field

Structure of IPv4 Packets (6/6)



- The field **sender IP address** (32 bits) contains the source address and **destination IP address** contains the destination address
- The field **Options / Padding** can contain additional information such as a time stamp
  - This last field before the payload area is filled with padding bits (0 bits) if necessary, to ensure that the header size is an integer number of 32 bit words
- The last field contains the data from the Transport Layer

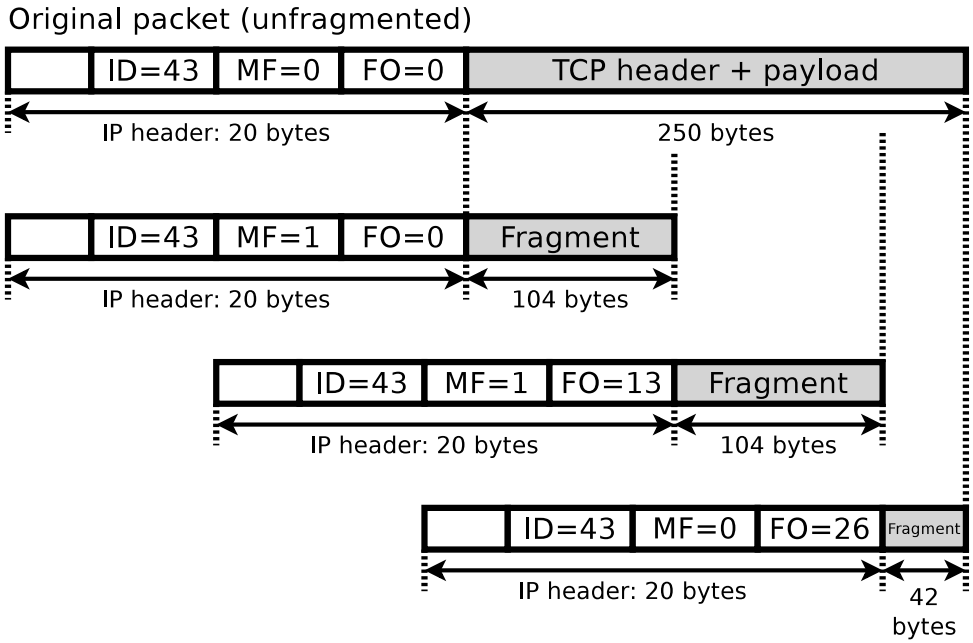
Packet Fragmentation (1/2)

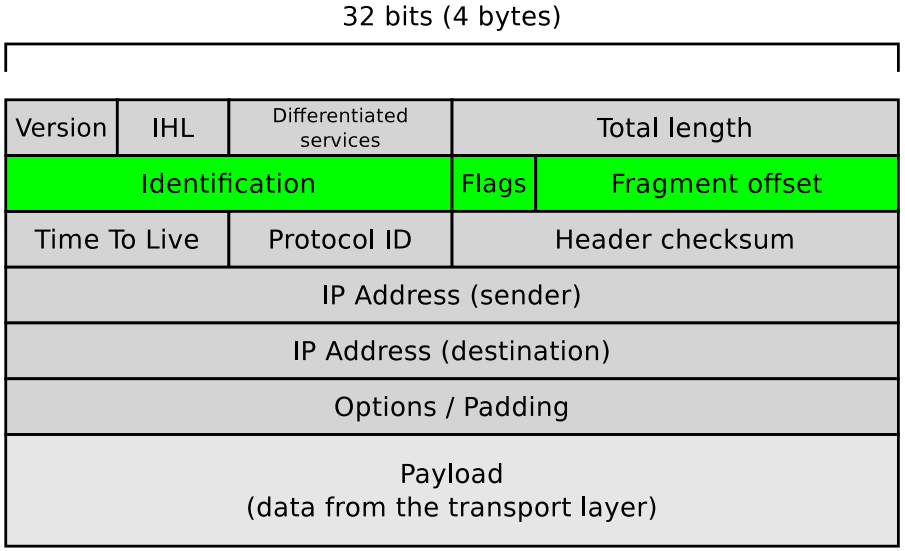
- The split up (and reassembling) of IP packets into smaller packets (**fragments**) is called **Packet fragmentation**
  - Is usually done by Routers
    - Packet fragmentation can also be carried out by the sender
- Reason for packet fragmentation:
  - The maximum packet length depends on the network technology used
- The **Maximum Transmission Unit** (MTU) specifies the maximum payload of a frame (and thus the maximum size of an IP packet too)
  - MTU of Ethernet: usually 1,500 bytes
    - For Gigabit Ethernet, *Jumboframes* exist with a size of up to 9,000 bytes
  - MTU of WLAN (IEEE 802.11): 2,312 bytes
  - MTU of Token Ring with 4 Mbit/s (IEEE 802.5): 4,464 bytes
  - MTU of Token Ring with 16 Mbit/s: 17,914 bytes
  - MTU of PPPoE (e.g. DSL): $\leq$ 1,492 bytes
  - MTU of ISDN: 576 bytes
  - MTU of FDDI: 4,352 bytes

Packet Fragmentation (2/2)

- IP packets contain a flag (`DF` = Don't Fragment) which can be used to prohibit fragmentation
  - If a Router needs to fragment a packet because it is too large to forward, but the fragmentation is prohibited in the packet, the Router discards the packet because he cannot forward it
- If a network device does not receive all fragments of an IP packet within a certain period of time (a few seconds), the network device discards all received fragments
- Routers can split IP packets into smaller fragments, if the MTU makes this necessary and it is not prohibited in the packets
  - **But no Router can assemble fragments of a packet to create a larger fragment**
    - Only the receiver can assemble fragments

Packet Fragmentation Example (1/2)





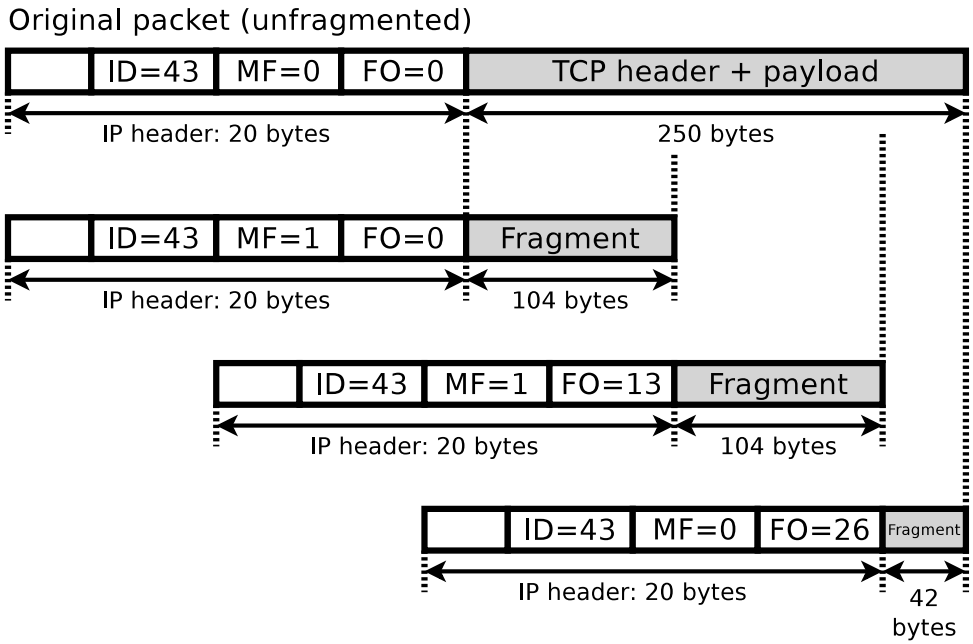
Source

`http://www.netzmafia.de/skripten/netze/netz8.html`

- A TCP segment of 250 bytes length is transmitted via IP
- Maximum packet length: 124 bytes
- IP header length: 20 bytes
- Packet ID: 43

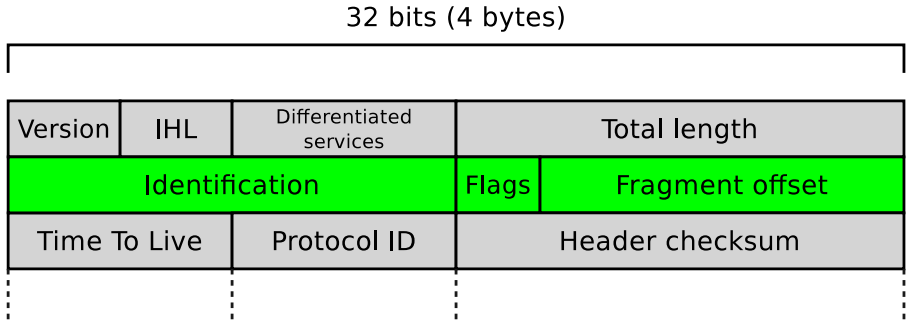
- The fragment offset is counted in 8-byte word increments. **The payload of all fragments (except the final one) must therefore be a multiple of 8 Bytes**
- Because all fragments belong to the same packet, the ID is equal for all fragments

Packet Fragmentation Example (2/2)



Source

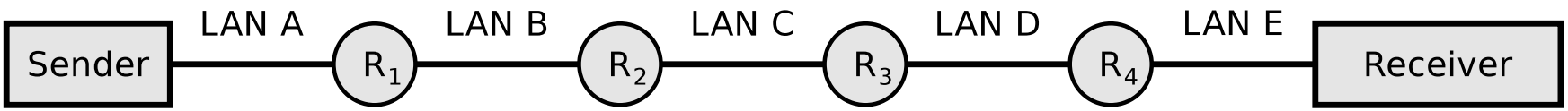
`http://www.netzmafia.de/skripten/netze/netz8.html`



- In the 1st fragment, `FO=0`
- MF flag=1 $\Longrightarrow$ more fragments will follow
- In the 2nd fragment, `FO=13` ($104/8 = 13$), which indicates the position of the fragment in the unfragmented packet
- MF flag=1 $\Longrightarrow$ more fragments will follow
- In the header of the 3rd fragment, the MF flag has value 0, because it is the final fragment of packet 43
- `FO=26` because $8 * 26 = 208$ bytes of data have already been sent

Another Fragmentation Example (1/2)

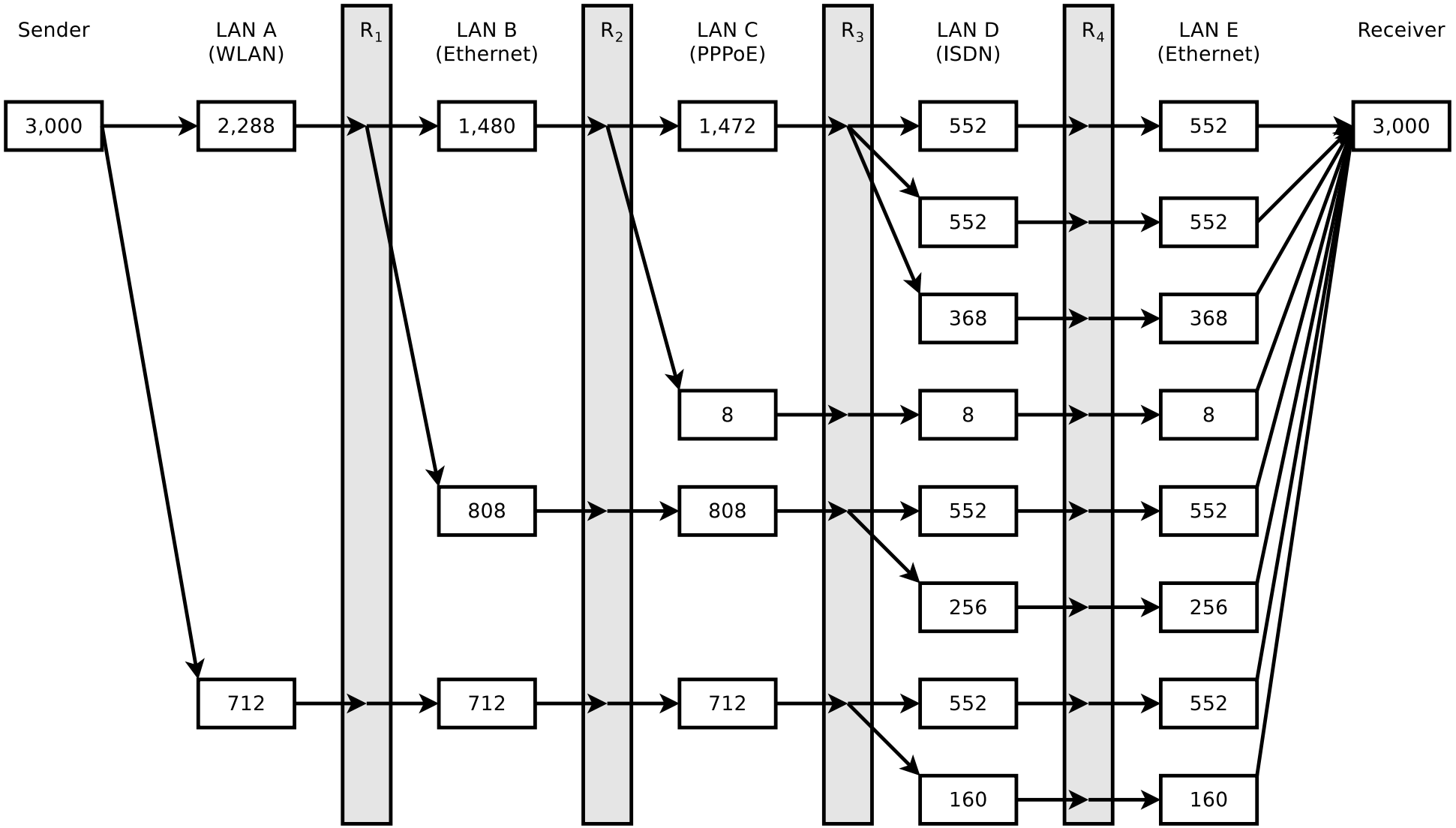
- 3,000 bytes payload need to be transmitted via the IP protocol
- The resulting packets must be fragmented because they are transmitted over multiple physical networks, whose MTU is $< 3,000$ bytes



	LAN A	LAN B	LAN C	LAN D	LAN E
Network technology	WLAN	Ethernet	PPPoE	ISDN	Ethernet
MTU [bytes]	2,312	1,500	1,492	576	1,500
IP-Header [bytes]	20	20	20	20	20
maximum payload [bytes] in theory	2,292	1,480	1,472	556	1,480
Multiple of 8	no	yes	yes	no	yes
maximum payload [Bytes] in practice	2,288	1,480	1,472	552	1,480

- Show in a graphical way how the packet is fragmented, and how many bytes of payload, each fragment contains

Another Fragmentation Example (2/2)



IP Packet Fragmentation in Practice

- **IP packet fragmenting is used seldom in practice today**
- Most modern operating systems and computer network devices avoid packet fragmentation because...
  - fragment loss increases the network overhead because of retransmissions
  - fragmentation and assembling of packets causes CPU load
  - Routers and Firewalls often block fragmented IP packets because of security concerns
    - Examples: **Ping of Death** and **Teardrop attack**

More information about Ping of Death and Teardrop attack

Jon Erickson. **Hacking: The Art of Exploitation**. 2nd Edition. No StarchPress (2008). P.256
Martin Kappes. **Netzwerk- und Datensicherheit**. 3. Auflage. Springer Vieweg (2022). P.296
Malachi Kenney. **Ping of Death** (1996). `https://insecure.org/sploits/ping-o-death.html`

- Modern operating systems and network devices perform **Path MTU Discovery** (PMTUD) to find out the maximum packet size (MTU) between sender and destination and avoid packet fragmentation

IPv6 has no packet fragmentation. Only Path MTU Discovery...

Status of IPv4

ZEIT ONLINE | INTERNET

INTERNET PROTOKOLL

Bye, bye IPv4

Die letzten Adressblöcke des alten Internet Protokolls Version vier sind vergeben. Die Umstellung auf IPv6, die seit Jahren nicht vorankommt, wird nun beginnen müssen.

VON: Monika Ermert | 2.2.2011 - 16:36 Uhr

Im Netz hat eine neue Zeitrechnung begonnen: In der Nacht zum Dienstag hat die Internet Assigned Numbers Authority (IANA) die letzten freien IPv4-Adressen verteilt. Wer künftig IP-Adressen an Nutzer vergeben möchte, sei es für Mobiltelefone, PCs oder internetfähige Autos, muss sich mit der nächsten Generation von "Rufnummern" befassen, mit der Internet-Protokoll Version 6 – IPv6.

Das Internet-Protokoll ist Teil der komplexen Struktur, die notwendig ist, damit Computer miteinander Daten austauschen können. Es sorgt darin für die korrekte Vermittlung der transportierten Informationen. IPv4 nutzt Adressen mit einer Länge von 32 Bit, was die Zahl der insgesamt verfügbaren IPs auf 4.294.967.296 oder 4,2 Milliarden Stück beschränkte.

Das klingt viel. Aber bei 6,5 Milliarden Menschen weltweit und angesichts des Trends, mehr und mehr Geräte internetfähig zu machen, ist seit Jahren klar, dass die IPv4-Adressen knapp werden. Netzanbieter nutzten daher dynamische Adressen, vergaben also keine festen für jedes einzelne Gerät. Doch auch diese Technik ist begrenzt, weswegen seit vielen Jahren an einem neuen Internet-Protokoll gearbeitet wurde.

IPv6 basiert auf längeren Nummern und bietet damit für die Zukunft die nicht mehr so richtig vorstellbare Zahl von 340 Sextillionen eindeutiger Internetadressen. Jedes Sandkorn könnte damit künftig eine IP-Adresse bekommen.

Bis heute allerdings kam die technische Umstellung nur langsam voran. Nun sind jedoch die letzten freien IPv4-Blöcke an den für Asien zuständigen regionalen IP-Adressverwalter vergeben worden. Bis diese an die einzelnen Netzbetreiber und deren Kunden verteilt sind, wird es noch eine Weile dauern. Außerdem bekommt jede der weltweit fünf Verwaltungen in den kommenden Tagen noch eine Reserve von 16 Millionen IPv4-Adressen, doch der Zeitraum ist absehbar.

Diagnosis and Error Messages via ICMP

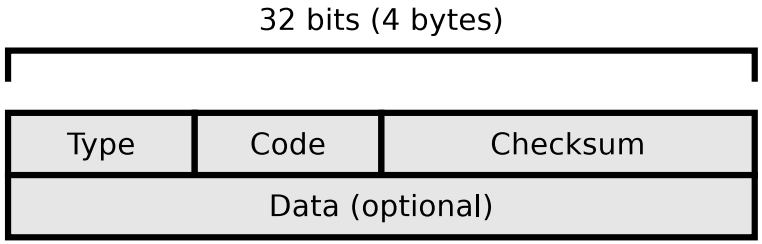
- The **Internet Control Message Protocol** (ICMP) is used for the exchange of...
  - diagnostic messages
  - control messages
  - error messages
- ICMP is a component (*sub-protocol*) of IPv4
  - but it is treated as a separate protocol

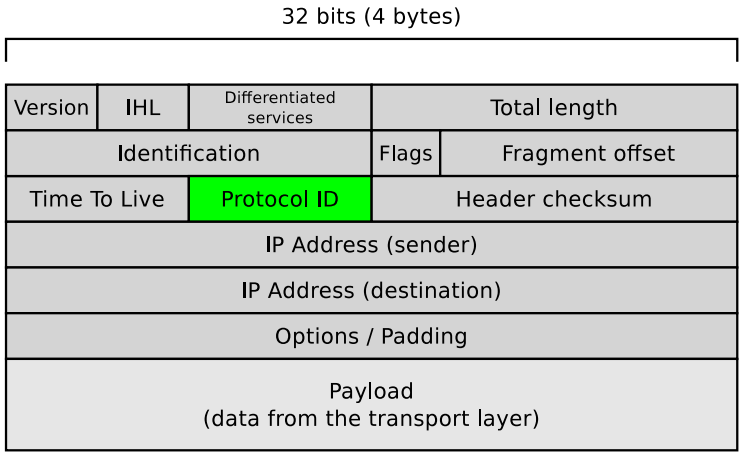
For IPv6, a similar protocol called ICMPv6 exists

- All Routers and terminal devices can handle ICMP
- Typical situations where ICMP is used:
  - A Router discards an IP packet, because it does not know how to forward it
  - Only a single fragment of an IP packet arrives at the destination
  - The destination of an IP packet cannot be reached, because the Time To Live (TTL) has expired

ICMP

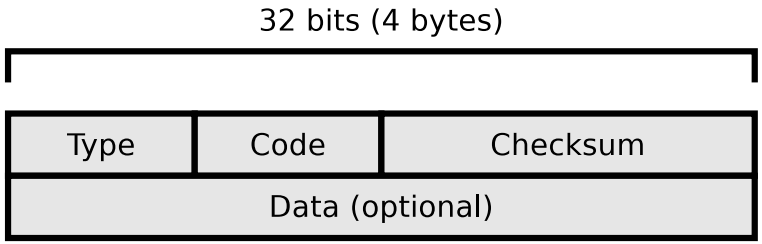
- Example of an application, which sends ICMP packets: `ping`
- ICMP specifies different sorts of messages, which can be sent by a Router as response to provide diagnostic information





- ICMP messages are transmitted as payload inside IPv4 packets
  - In the header of the IPv4 packet the field **protocol ID** contains value 1
  - For ICMPv6 the protocol ID is 58
- If an ICMP packet cannot be delivered, no further action is done

ICMP Messages

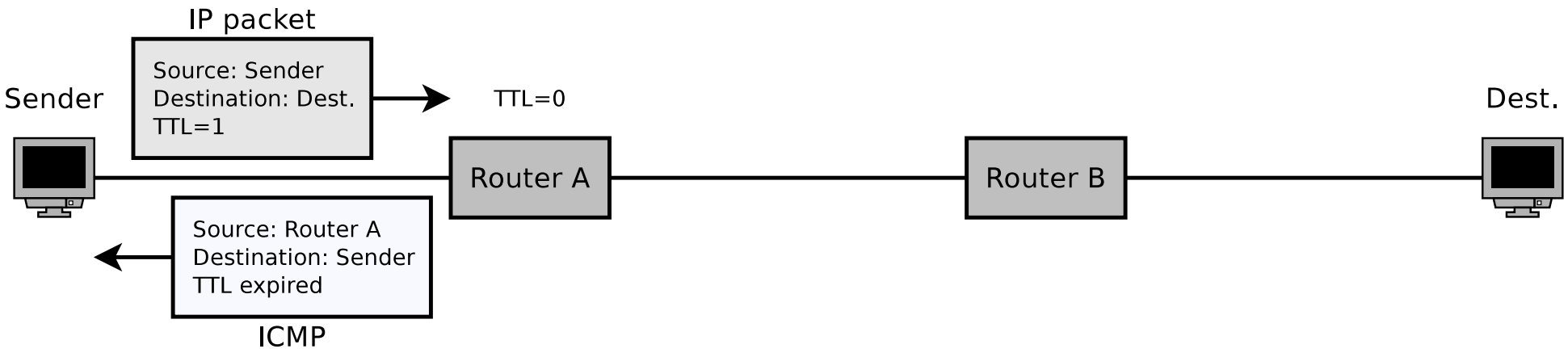


- The field **Type** in the ICMP header specifies the ICMP message type
- The field **Code** specifies the subtype of the message type
- The table contains some type-code combinations of ICMP messages

Type	Name of type	Code	Description
0	Echo reply	0	Echo reply (reply for `ping`)
3	Destination unreachable	0	Destination network unreachable
		1	Destination host unreachable
		2	Destination protocol unreachable
		3	Destination port unreachable
		4	Fragmentation required, but forbidden by the IP packet's flags
		13	Firewall at destination site rejects the IP packet
5	Redirect	0	Router informs sender about a better route (IP of first hop) to destin. network
		1	Router informs sender about a better route (IP of first hop) to destination host
8	Echo Request	0	Echo request (`ping`)
11	Time Exceeded	0	TTL (Time To Live) expired (exceeded in transit)
		1	Fragment reassembly time exceeded

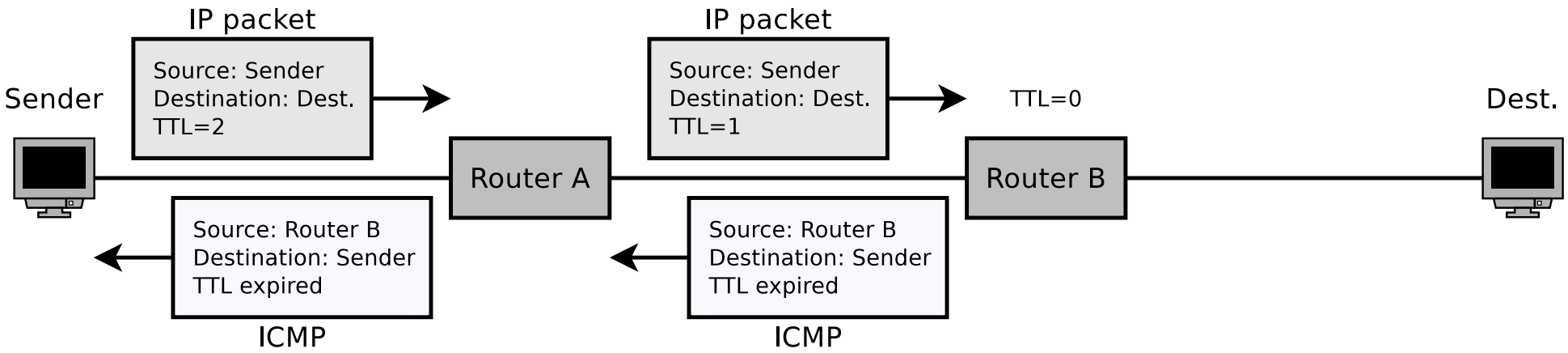
The ICMP protocol includes many more type-code combinations (see RFC 792), but most were seldom or never used in practice and are considered deprecated (see RFC 6633 and RFC 6918)

Example of using ICMP: `traceroute` (1/3)



- Another application example of ICMP is the tool `traceroute`
- `traceroute` determines, which Routers are used to forward packets to the destination site
  - It also measures the Round Trip Time (RTT) from the sender to each Router
- The sender transmits an IP packet to the destination with `TTL=1`
- Router A receives the IP packet, sets `TTL=0`, discards the IP packet and transmits an ICMP message of message type 11 and code 0 to the sender

Example of using ICMP: `traceroute` (2/3)

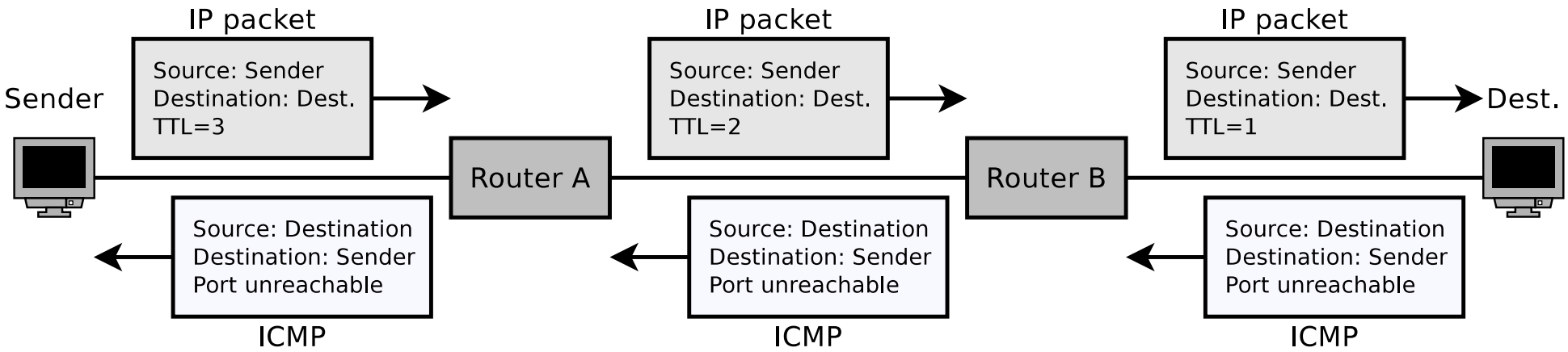


- Next, the sender transmits an IP packet to the destination with `TTL=2`
- The IP packet is forwarded by Router A
  - Thereby the value of `TTL` is decremented
- Router B receives the IP packet, sets `TTL=0`, discards the IP packet and transmits an ICMP message of message type 11 and code 0 to the sender

Attention! There are different implementations of `traceroute`

`tracert` in Windows uses ICMP by default, but `traceroute` in Linux and Mac OS X uses UDP by default. However, the use of ICMP can be forced via the command-line parameter `-I` Alternatively, TCP is also possible

Example of using ICMP: `traceroute` (3/3)



- Once the value of `TTL` is big enough that the destination site can be reached, the receiver transmits an ICMP message of message type 3 and code 3 to the sender
- This way, the path from sender to receiver can be traced via ICMP

```
$ traceroute -q 1 wikipedia.de
traceroute to wikipedia.de (134.119.24.29), 30 hops max, 60 byte packets
 1  fritz.box (10.0.0.1) 1.834 ms
 2  p3e9bf6a1.dip0.t-ipconnect.de (62.155.246.161) 8.975 ms
 3  217.5.109.50 (217.5.109.50) 9.804 ms
 4  ae0.cr-polaris.fra1.bb.godaddy.com (80.157.204.146) 9.095 ms
 5  ae0.fra10-cr-antares.bb.gdinf.net (87.230.115.1) 11.711 ms
 6  ae2.cgn1-cr-nashira.bb.gdinf.net (87.230.114.4) 13.878 ms
 7  ae0.100.sr-jake.cgn1.dcnet-emea.godaddy.com (87.230.114.222) 13.551 ms
 8  wikipedia.de (134.119.24.29) 15.150 ms
```

In this example, only one packet (`-q 1`) is sent per hop. Therefore, the output contains only one round-trip time (RTT) value per hop. By default, `traceroute` sends three separate packets per hop, resulting in three RTT values. Strong RTT fluctuations indicate network problems or resource bottlenecks.

Example of using ICMP: Path MTU Discovery (RFC 1063)

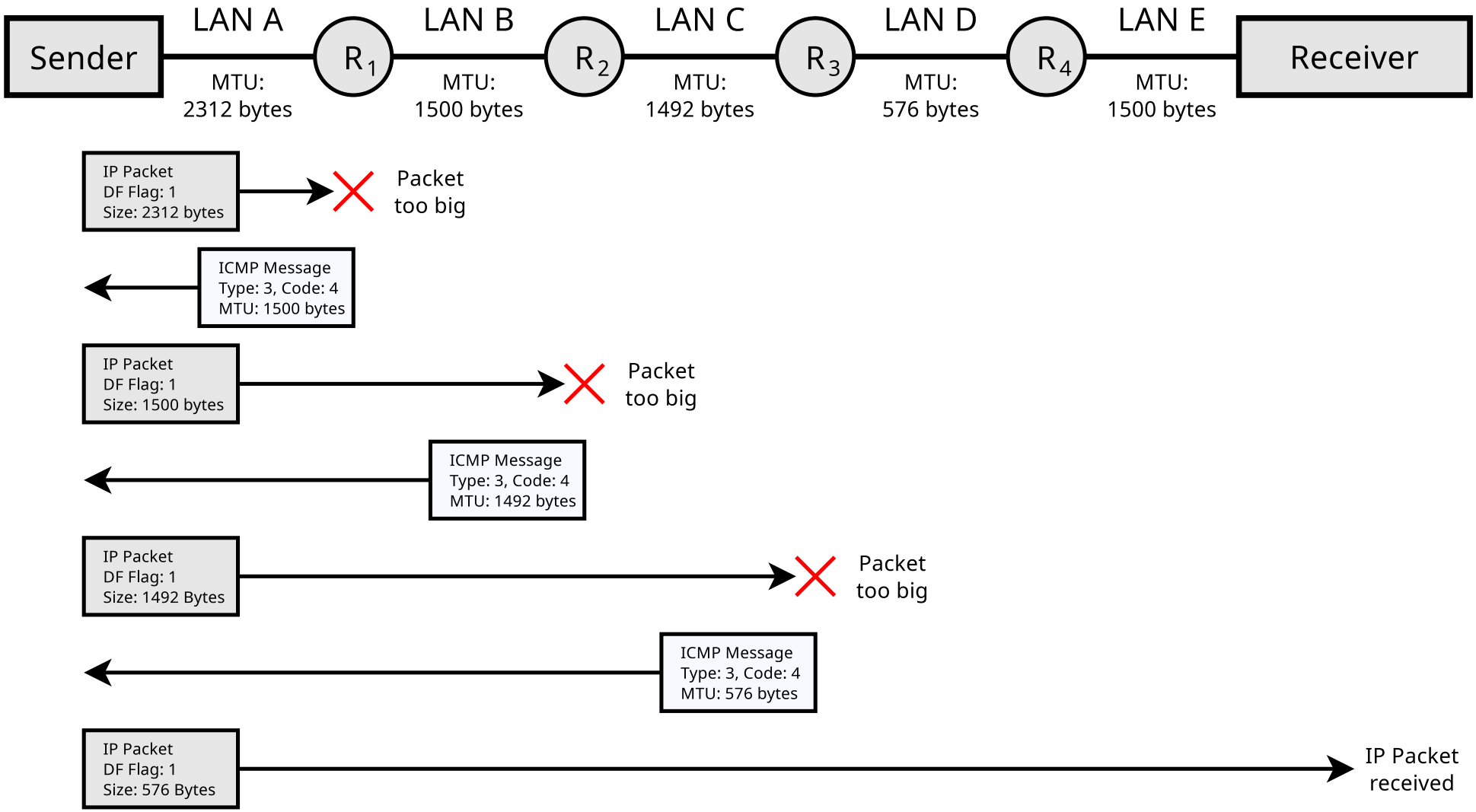
- Modern operating systems and network devices perform **Path MTU Discovery** (PMTUD) to learn the maximum packet size between sender and destination (*Path MTU*) and avoid IP packet fragmentation
  - The flag **Don't Fragment** (`DF`) in the IP headers of outgoing IP packets is today usually set by default
    - A network with a smaller MTU than required by the IP packet causes its Router(s) to drop the packet and send an ICMP message type 3 and code 4 (**Fragmentation Needed**) to the sender that contains the MTU as data
    - The sender reduces its packet size and repeats trying to send a packet until the MTU is small enough to reach the destination without fragmentation

The path between sender and destination may change any time...

- if Routers on the new path have a lower MTU, the sender will learn about this because of an ICMP message **Fragmentation Needed**
- if the Routers on the new path are capable of a bigger MTU, the sender will not learn about this

Source: Andrew Tanenbaum, David Wetherall. *Computer Networks*. Pearson (2011). 5th edition. P. 432-436

Example of using ICMP: Path MTU Discovery



IPv6 Addresses and Address Space

- IPv6 addresses (RFC 4291) have a length of 128 bits (16 bytes)
  - Therefore, $2^{128} \approx 3.4 * 10^{38}$ addresses can be represented
- The introduction is useful because of the limited address space of IPv4
- Problem: The decimal notation is confusing
  - For this reason, IPv6 addresses are represented in hexadecimal format
    - Groups of 4 bits are represented as a hexadecimal number
  - Groups of 4 hexadecimal numbers are merged into blocks
  - Colons separate the blocks
    Example: `2001:0db8:85a3:08d3:1319:8a2e:0370:7344`

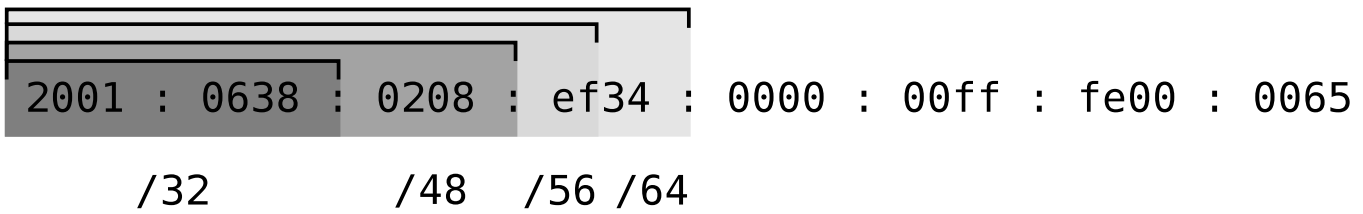
- The last 4 bytes (32 bits) of an IPv6 address may also be written in decimal notation
- This is useful to embed the IPv4 address space into the IPv6 address space
  $\Longrightarrow$ see slide 77

Addresses only for documentation

`2001:db8::/32` $\Longrightarrow$ Addresses only for documentation purposes

Subnets in IPv6

- (Sub-)netmasks do not exist in IPv6
  - The subdivision of address ranges into subnets is done by specifying the prefix length
- IPv6 networks are specified in CIDR notation
  - The address of a single device sometimes has `/128` attached
  - An example is the loopback address of IPv6: `::1/128`
    - All bits – except the last one – are value `0`
      (For IPv4, the loopback address is: `127.0.0.1`)
  - Internet Providers (ISPs) or operators of large networks get the first 32 or 48 bits assigned from the Regional Internet Registry (RIR)
    - The ISPs or network operators split this address space into subnets
    - **End users usually get a `/64` or even a `/56` network assigned**



- If a user gets a /56 network assigned, the 8 Bits between the Prefix and the Interface Identifier is the **Subnet Prefix**

Simplification of IPv6 Addresses

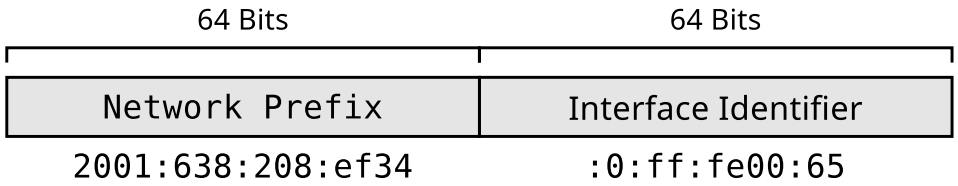
- Rules for simplification (RFC 5952):
  - Leading zeros within a block may be omitted
  - Successive blocks with value 0 (= 0000) may be omitted **exactly one time within an IPv6 address**
    - If blocks are omitted, this is indicated by 2 consecutive colons
  - If several groups of zero blocks exist, it is recommended to shorten the group with the most zero blocks
- Example:
  - The IPv6 address of `j.root-servers.net` is:
    `2001:0503:0c27:0000:0000:0000:0002:0030`
    $\Longrightarrow$ `2001:503:c27::2:30`

Notation of IPv6 addresses (URLs)

- IPv6 addresses are enclosed in square brackets
- Port numbers are appended outside the brackets
  `http://[2001:500:1::803f:235]:8080/`
- This prevents the port number from being interpreted as part of the IPv6 address

Structure of IPv6 Addresses and Networks

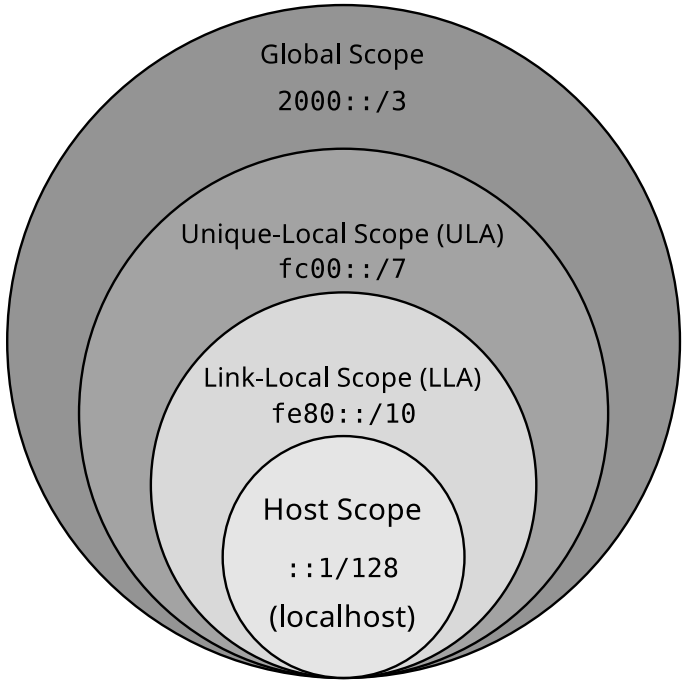
- IPv6 addresses consist of 2 parts



1. **Prefix** (Network Prefix)
   - Identifies the network
2. **Interface identifier** (Interface ID)
   - Identifies a network device in a network
     - Multiple ways exist to set the Interface ID (see slide 60)

Scopes and Functioning (1/4)

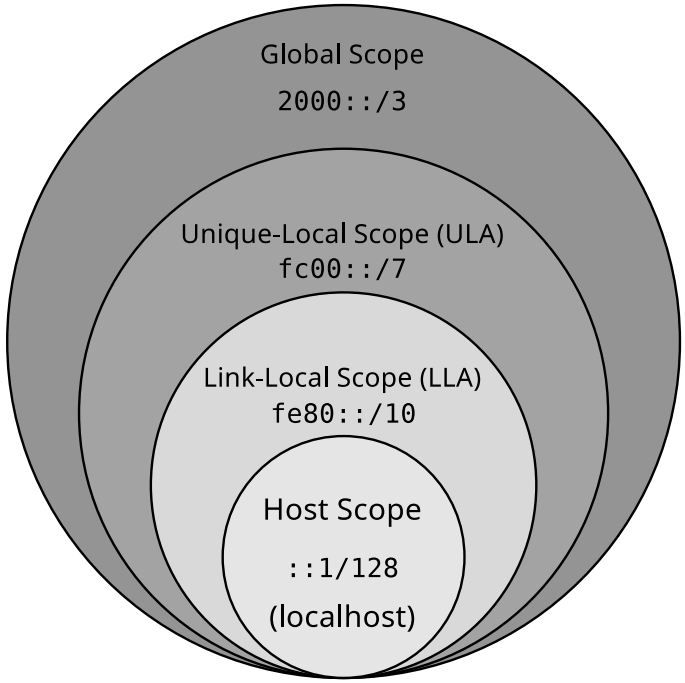
- IPv6 not only distinguishes private and public addresses (like IPv4), but also several address scopes
- Each IPv6 address has a so-called scope
- The scope is the part of a network in which the associated address is considered valid and routed



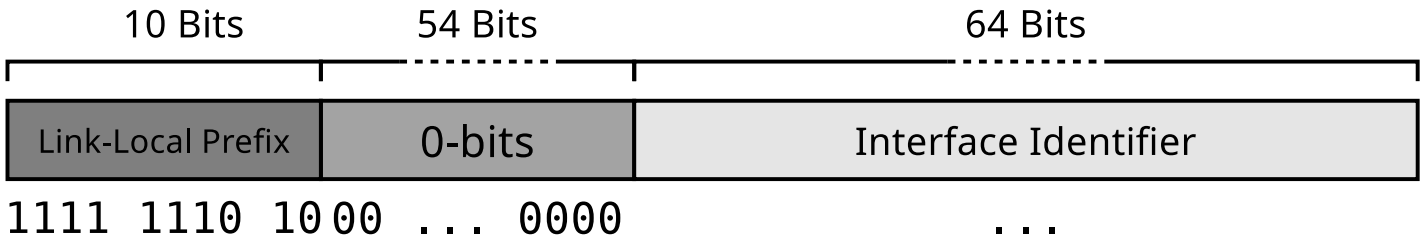
- **Host Scope**: Loopback Address
  - The loopback address is `::1/128` $\Longrightarrow$ `0:0:0:0:0:0:0:1/128`

Scopes and Functioning (2/4)

- **Link-Local Scope**: Link-Local (Unicast) Addresses (LLA)
  - Every network interface needs a Link-Local Address at any time
  - Link-Local Addresses `fe80::/10` are only valid in the local network
  - Routers do not forward packets with these addresses

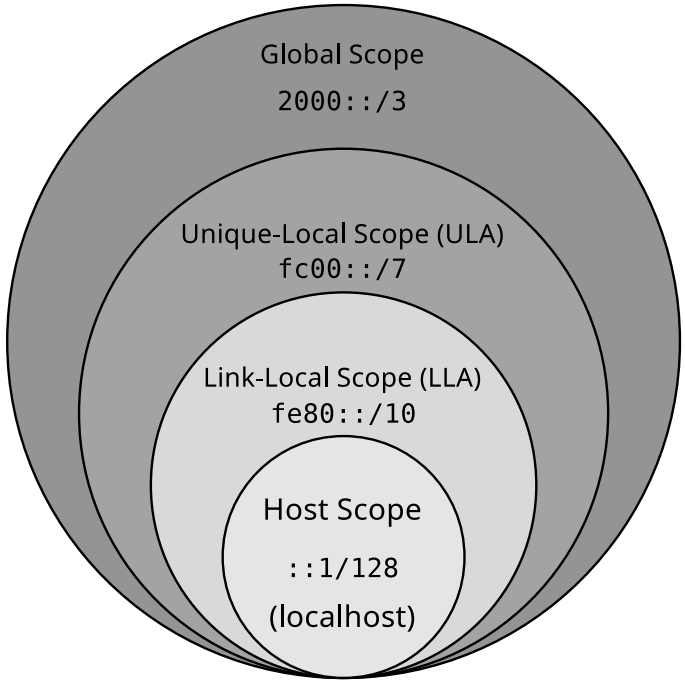


- They are required for communication inside the local network (e.g. WLAN or Ethernet) and are used, i.a., for generating a global valid IPv6 address or fetching one via DHCPv6



Scopes and Functioning (3/4)

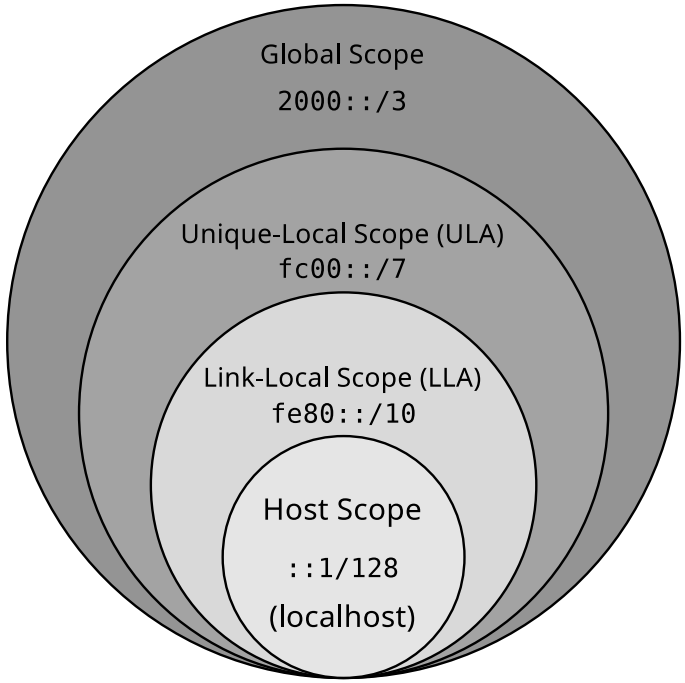
- **Unique-Local Scope**: Unique Local Addresses (ULA)
  - Routers should not forward the addresses `fc00::/7` ($\Longrightarrow$ `fc00...` until `fdff...`) outside the local administrative domain
  - *Private addresses* intended for local communication inside an administrative domain (organization or site)



- `fc...` $\Longrightarrow$ **assigned unique ULA**
  - Prefix `fc` is followed by 40 bits long assigned (unique) Site-ID and a 16 bits long Subnet-ID
  - Such addresses are globally valid, unique and assigned by a provider
- `fd...` $\Longrightarrow$ **local generated ULA** (see slide 57)
  - Prefix `fd` is followed by 40 bits long self-generated Site-ID (very likely unique) and a 16 bits long Subnet-ID

Scopes and Functioning (4/4)

- **Global Scope**: Global Unicast Addresses
  - Routers forward the addresses `2000::/3` ($\Longrightarrow$ `2000...` until `3fff...`)



Site-Local Scope

The Site-Local scope (see RFC 1884) specifies addresses (`fec0::/10`) valid within the network of an organization (*inside a site*). It became obsolete in 2004 (see RFC 3879).
The scope was replaced in 2005 (see RFC 4193) with Unique Local Addresses (ULA)

Unique Local Addresses – ULA (RFC 4193)

- The table shows the RFC 4193 addressing scheme of static IPv6

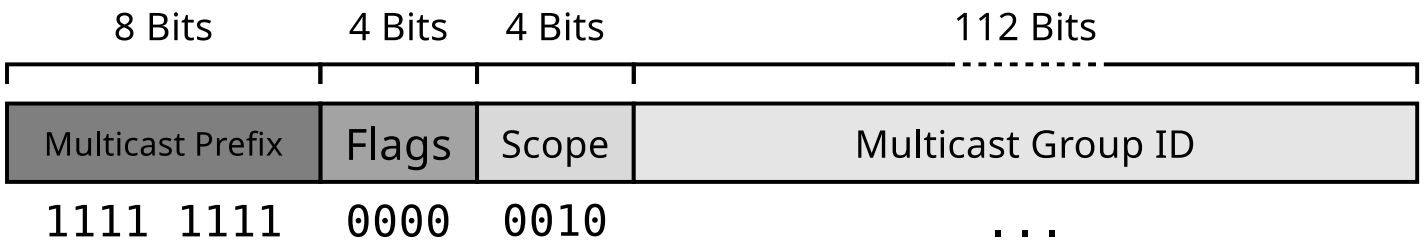
Prefix/Length (8 bits)	**Global ID** (40 bits)	**Subnet ID** (16 bits)	**Interface ID** (64 bits)
fd00::/8	12:3456:789a	0001	0000:0000:0000:0001

Resulting IPv6 address: fd12:3456:789a:0001:0000:0000:0000:0001
Short IPv6 address: fd12:3456:789a:1::1

- ULAs are used in situations where a network ID (network prefix) is offered by a network provider
- Only used for local sites (administrative domains)
  - Not routed into the global internet
    - For local generated ULA, address conflicts are possible (but very unlikely)
  - ULAs are analogous to private addresses in IPv4
    - They are used inside administrative domain (site or organization)

IPv6 Multicast Addresses (1/2)

- **IPv6 has no broadcast addresses**
  - **Multicast addresses emulate the broadcast functionality**



- Multicast addresses start with the first 8 bits set to value `11111111`
  - Thus, they have the **multicast prefix** `ff::/8`
- The prefix is followed by 4 **flag bits** and 4 **scope bits**

The flag bits are irrelevant in the context of this course and have, in practice, in most cases, the value 0

- Scopes that are relevant in practice are the values 1 and 2 ($\Longrightarrow$ see next slide)

IPv6 Multicast Addresses (2/2)

Prefix	Scope	Meaning
`ff01`	interface-local	Packet to this address does not leave the interface $\Longrightarrow$ **Loopback**
`ff02`	link-local	Packet is not forwarded by Routers $\Longrightarrow$ does not leave the subnet

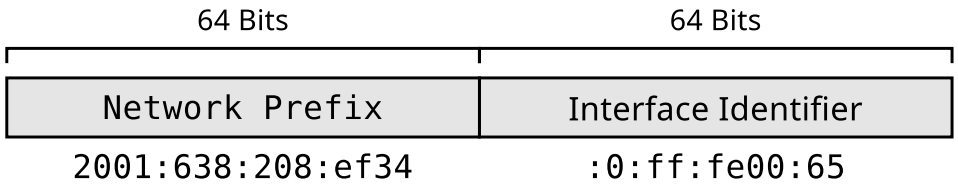
Die deprecated (see RFC 3879) scope `ff05` (site-local) specifies that a packet is not forwarded by Border-Routers $\Longrightarrow$ it does not leave the network of an organization

- The scope is followed by the multicast group ID
- The table contains some popular multicast addresses

IPv6 address	Scope	Meaning (addresses...)
`ff01::1`	interface-local	all nodes on the local node
`ff01::2`	interface-local	all Routers (typically softw. Router instances) on the local node
`ff02::1`	link-local	all nodes in the local network $\Longrightarrow$ emulates **Broadcast**
`ff02::2`	link-local	all local Routers
`ff02:1:2`	link-local	all local DHCPv6 servers
`ff02::9`	link-local	all local Routers that implement the Routing protocol RIP
`ff02::5`	link-local	all local Routers that implement the Routing protocol OSPF
`ff02::6`	link-local	all local Designated Routers that implement OSPF
`FF02::F`	link-local	all devices that implement UPnP (Universal Plug and Play)

Setting the Interface Identifier

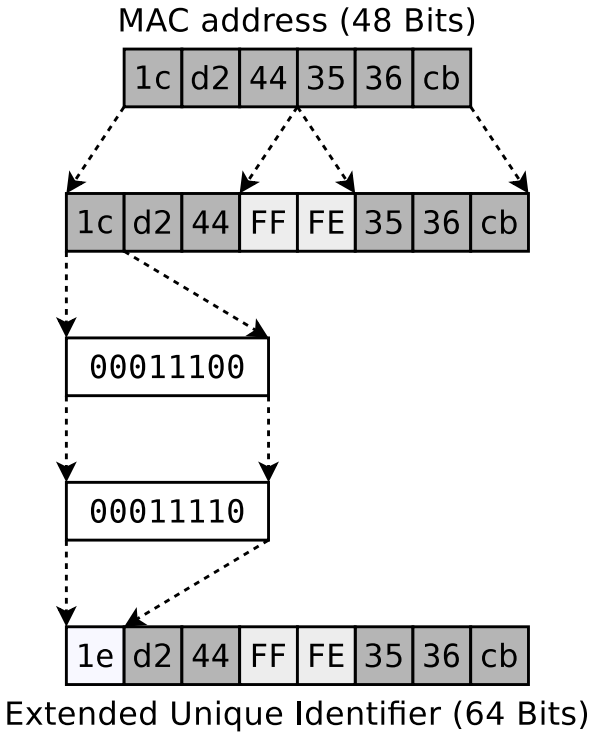
- Multiple ways exist to set the **Interface Identifier** (Interface ID)



1. **Static manual addressing**
   - Interface ID is manually set – possible but not convenient
2. **Stateless Address Autoconfiguration – SLAAC** (RFC 4862)
   - Calculate the 64 bits long interface ID from the 48 bits long MAC address
     $\Longrightarrow$ The interface ID is then called **Extended Unique Identifier** (EUI)
   - Optional extension: **Stable Privacy Addresses** (RFC 7217)
     - Calculate a stable interface ID using a random secret key (without using the MAC address) for offering anonymity
   - Optional extension: **Privacy Extension** (RFC 4941)
     - Calculate periodically a new interface ID using a random number (without using the MAC address) for offering even stronger anonymity
3. **Setting the network configuration via DHCPv6** (RFC 8415)
   - Only address configuration option that operates **stateful**

Stateless Address Autoconfiguration – SLAAC (RFC 4862)

- Automatic stateless IPv6 address generation by using the MAC address
- Converts a MAC into a host ID (64 bits)
  1. MAC address is split into 2 halves
     - 1st part becomes the first 24 bits
     - 2nd part becomes the final 24 bits of the modified EUI-64 address
  2. Bit pattern of the 16 bits in the center of the EUI-64 address:
     `1111 1111 1111 1110` (hex: `FFFE`)
  3. Finally, the 7th bit is set to value 1
- Drawback: Recovering the MAC address is simple ($\Longrightarrow$ Privacy concerns)



Router $\Longrightarrow$ Advertisement Daemon (`radvd`)

For the automatic assignment of Network prefixes the Router needs a `radvd` for the management of network prefixes in the network. Without `radvd` the Link-Local prefix `fe80::/64` is assigned

The 7th bit in EUI-64 is relevant here because it is the Universal/Local (U/L) bit in MAC addresses. If the bit has value 0, the MAC is globally unique. If the value is 1, it means the MAC is administered locally (which is the case with SLAAC), and the address is not globally unique. In a virtualization scenario, the bit typically has already value 1 and does not need to become modified during EUI-64 computation.

Stateless Address Autoconfiguration – Steps

- In practice, SLAAC works in 5 steps
  1. Client generates a modified EUI-64 Host-ID from its own MAC address
  2. Client generates a Link-Local Address (LLA) with prefix `fe80::/64`

By concatenating both components, a 128-bit IPv6 address is created that is valid in the local network

  3. Client sends for **Duplicate Address Detection** (DAD) a **Neighbor Solicitation** (NS) message with its own Link-Local Address via multicast in the Link-Local network and waits if a **Neighbor Advertisement** (NA) response is replied
     (see slide 66)
  4. Client sends a **Router Solicitation** (RS) message to the Router
     (see slide 69)
  5. Router sends a **Router Advertisement** (RA) message to the Client and hands over the global valid network prefix
     (see slide 68)

IPv6 Neighbor Discovery Protocol

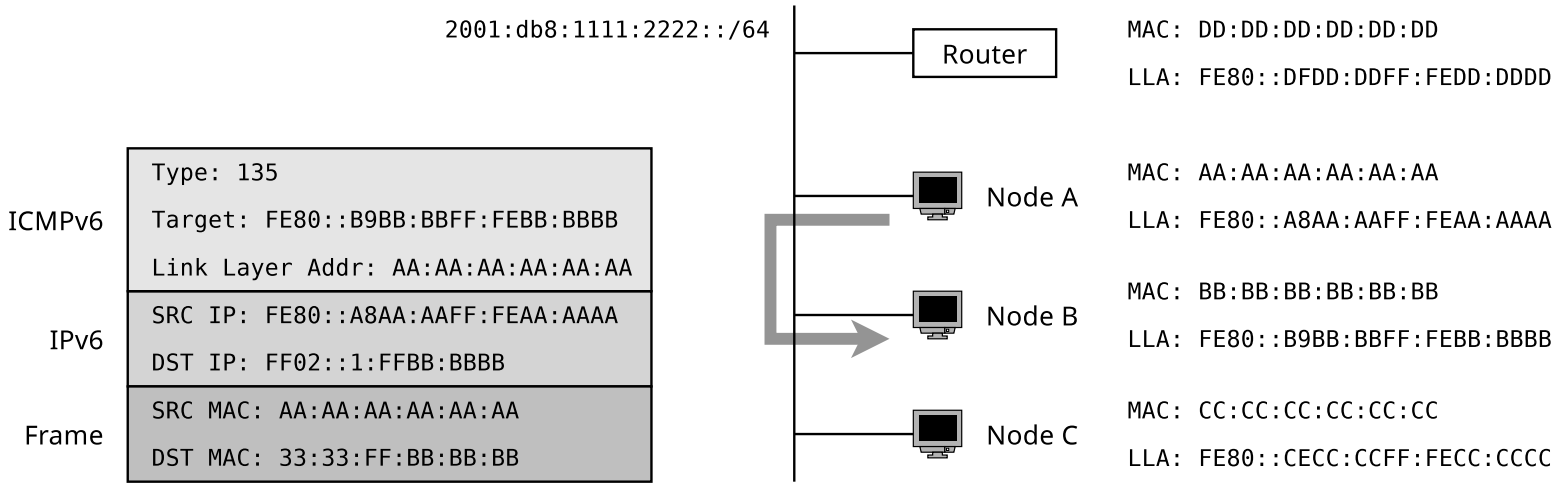
- IPv6 does not implement broadcast addresses and there is no equivalent to the Address Resolution Protocol (ARP)
  - But resolving IP addresses into MAC addresses is required too
- **Resolving MAC addresses from IPv6 addresses is done with the Neighbor Discovery Protocol (NDP) by using multicast addresses**

In the context of IPv6, the term **neighbor** describes nodes that are in the same Data Link Layer network

- NDP messages are exchanged as payload of ICMPv6 messages
- NDP implements 5 types of messages
  - **Router Solicitation** (ICMPv6 type 133)
  - **Router Advertisement** (ICMPv6 type 134)
  - **Neighbor Solicitation** (ICMPv6 type 135)
  - **Neighbor Advertisement** (ICMPv6 type 136)
  - **Redirect Message** (ICMPv6 type 137)

With the **Redirect Message** informs a Router of a better route (other first hop $\Longrightarrow$ other local Router) for a destination. This message type is not further discussed in this course

Neighbor Solicitation – NS (1/2)

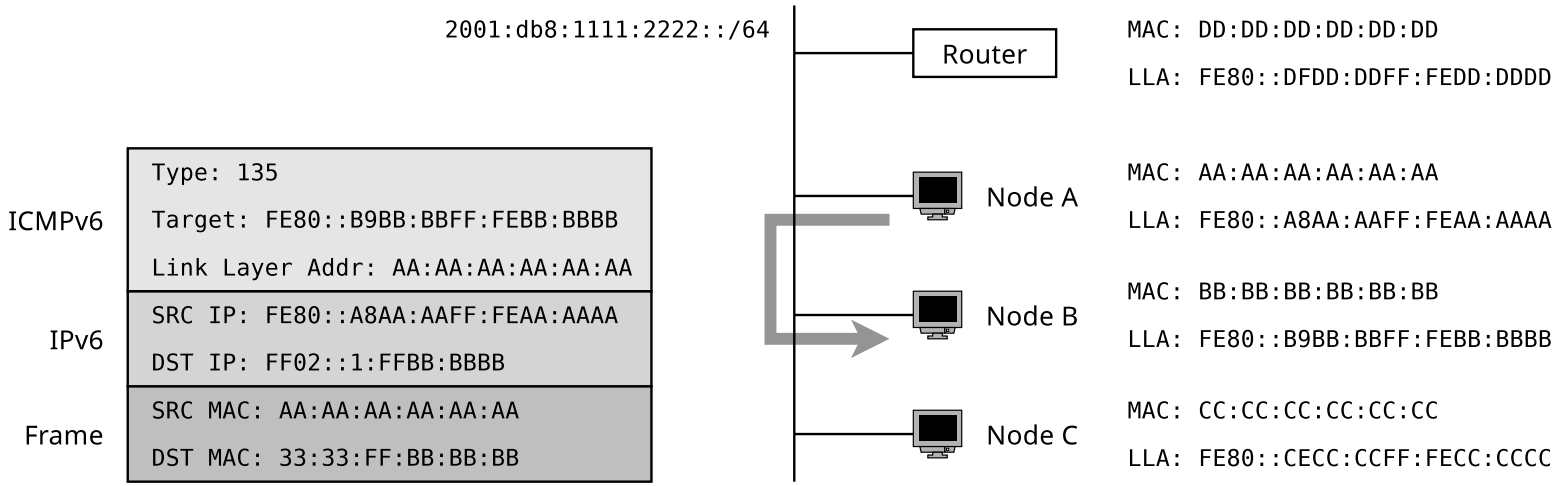


The Neighbor Solicitation (NS) message is the IPv6 alternative to an ARP Request when using IPv4

- Request for the MAC address of a neighbor
  - The diagram shows how node A requests the MAC address of node B
- The destination IP address in the IPv6 packet is called **Solicited-node multicast address**
  - Every node joins a multicast group for every configured IPv6 address
    - The multicast group has the address `FF02::1:FFXX:XXXX`

`XX:XXXX` stands for the last 6 hexadecimal characters in the Link-Local (Unicast) address (LLA)

Neighbor Solicitation – NS (2/2)



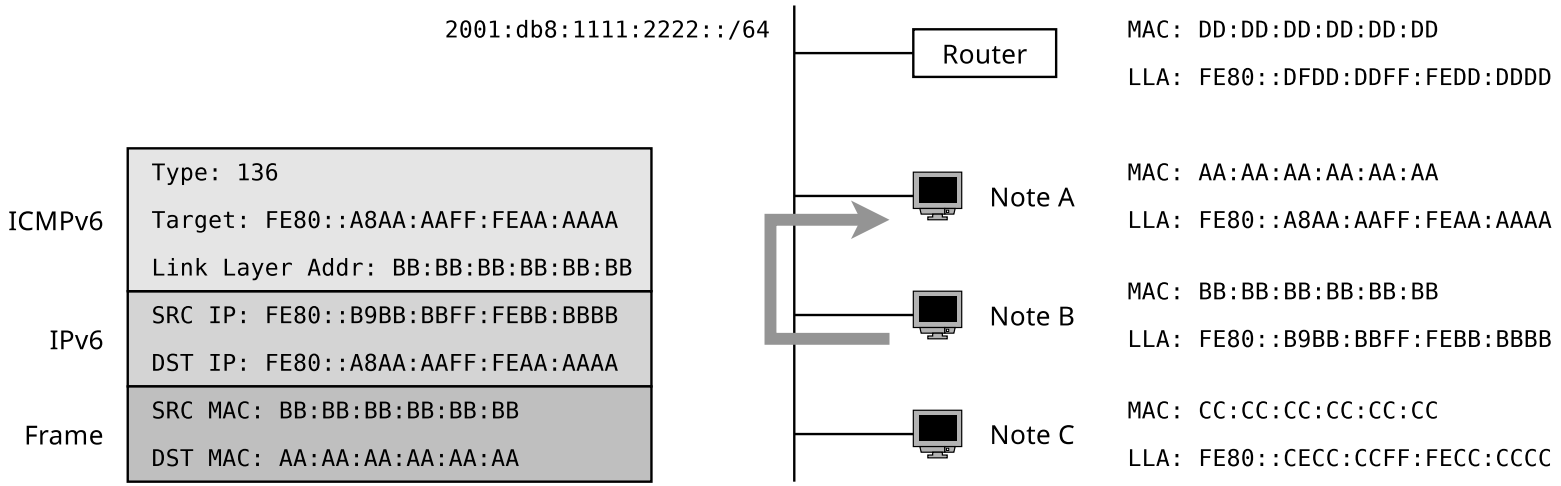
- The destination MAC address in the frame is called **multicast MAC address**
  - The multicast MAC address is `33:33.XX:XX:XX:XX`

`XX:XX:XX:XX` stands for the last 8 hexadecimal characters in the Solicited-node multicast address

Duplicate Address Detection with Neighbor Solicitation

- The Neighbor Solicitation (NS) message is also used for **Duplicate Address Detection (DAD)**
  - If a node has generated a *tentative* IPv6 address for itself, it needs to validate that no other node in the network already uses this address
- The node sends a Neighbor Solicitation (NS) message to the address that it wants to use itself
  - The sender address is the unspecific address (`::` $\Longrightarrow$ 128 zero bits)
  - If a node in the local network already uses this IP address, it is a duplicate
    - The node will reply with a Neighbor Advertisement (NA) message sent to the Link-Local multicast address `FF02::1` (every node in the local network will receive this message)
  - The node that was sending the Neighbor Solicitation (NS) message needs to generate a new address and carry out the Duplicate Address Detection again
  - If no Neighbor Advertisement (NA) message is received for some time, the address can be used ($\Longrightarrow$ no duplicate)

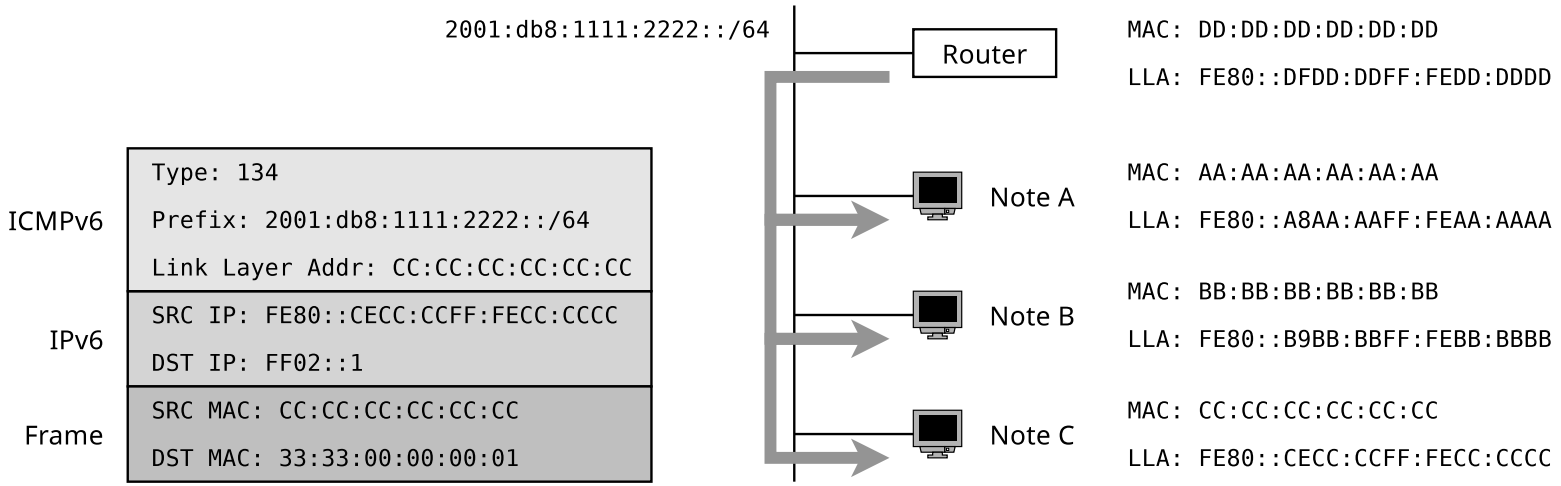
Neighbor Advertisement – NA



The Neighbor Advertisement (NA) message is the IPv6-alternative to an ARP Reply when using IPv4

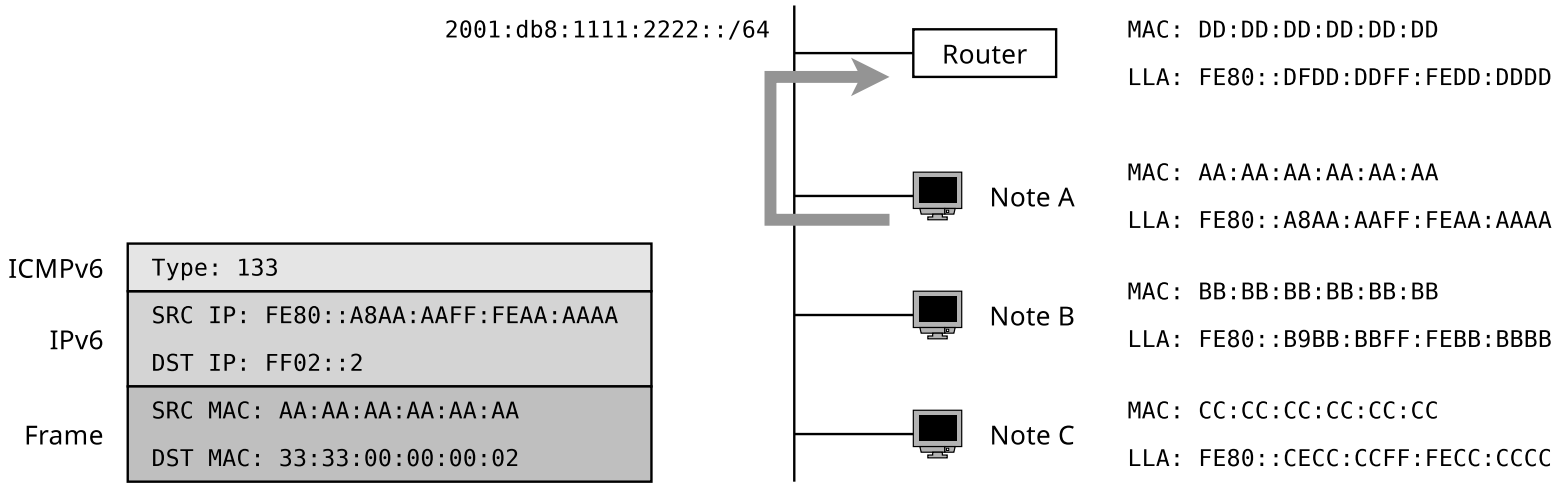
- Reply to a Neighbor Solicitation (NS) message
- Neighbor Advertisement is a unicast message
  - No multicast addresses are used here

Router Advertisement – RA



- Routers periodically send (the time can be set in the UI) Router Advertisement (RA) messages into connected networks to inform others about their presence, the network prefix, the prefix length, and, i.a., the MTU
  - The destination address in the IPv6 packet is the Link-Local multicast address `FF02::1` to reach all nodes in the local network
- The RA message also includes the flag **managed**
  - If it is set, the client is supposed not to set the address stateless but to request the address configuration from a DHCPv6 server (stateful)

Router Solicitation – RS



- If a node does not want to wait for incoming Router Advertisement (RA) messages, it can request RA messages by sending RS messages
  - Destination address in the IPv6 packet is the Link-Local multicast address `FF02::2` to reach all Routers in the local network
  - The diagram shows how node A requests a RA message from every local Router

SLAAC Extension: Stable Privacy (RFC 7217) – (1/3)

- **Optional extension of SLAAC** (Stateless Address Autoconfiguration)
- Specifies the address generation without using a MAC address
  - A random secret key is generated and used for Interface ID generation
    - The secret key is a 128-bit long hexadecimal string that looks like an IPv6 address

Location of the secret key in Linux and required kernel parameter

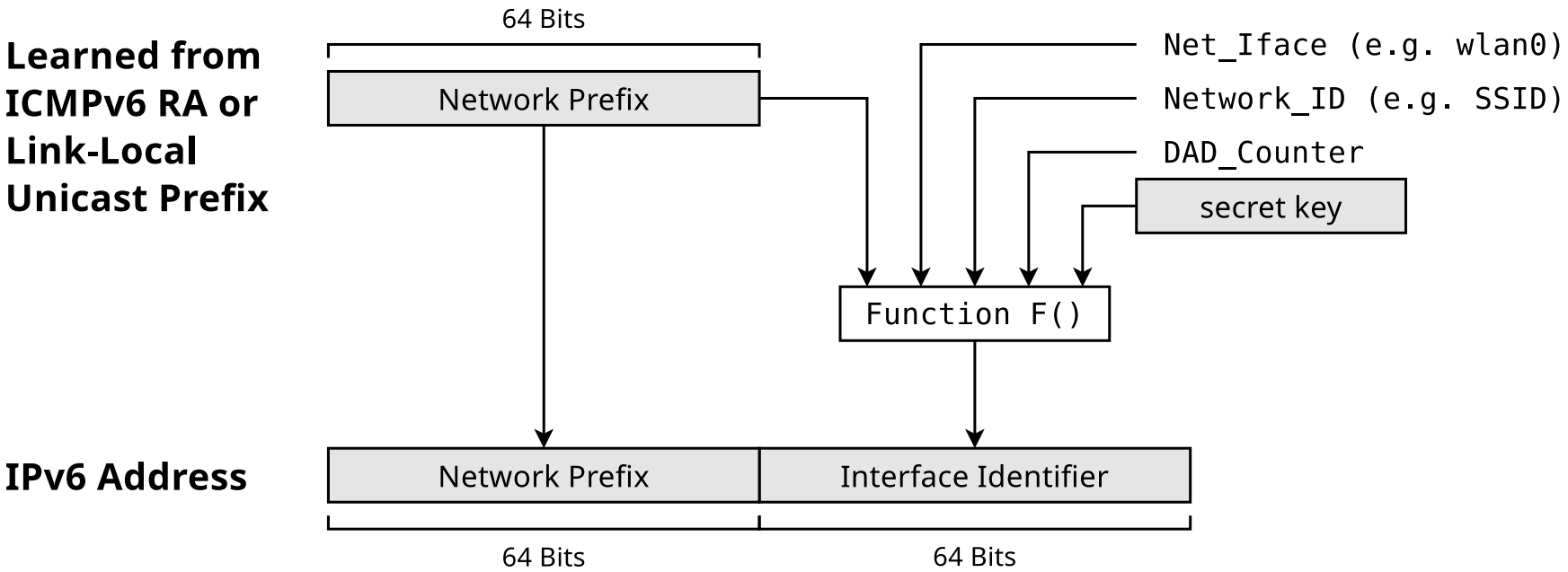
The stable secret value is stored in the file `/proc/sys/net/ipv6/conf/eth0/stable_secret` and is generated by setting the Kernel parameter `addr_gen_mode=3`

Example of a secret key

```
$ cat /proc/sys/net/ipv6/conf/eth0/stable_secret
c8c8:036d:9312:71e2:eadc:7c9f:0535:649a
```

- Benefits:
  - Improved security because no MAC address is used for generation
    - The MAC address of the node is not exposed $\Longrightarrow$ anonymity
  - Stable address for the node
    - Once generated, the Interface ID does not change anymore (until reboot)

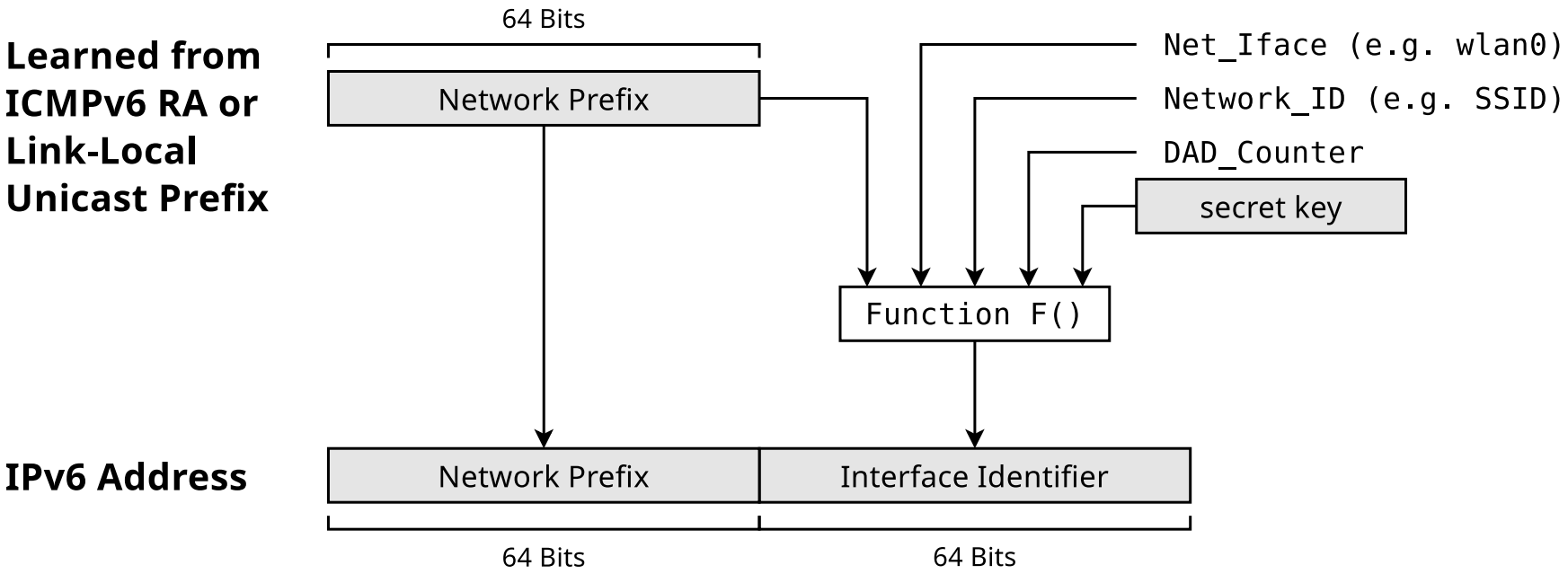
SLAAC Extension: Stable Privacy (RFC 7217) – (2/3)



- For generating the Interface ID, this pseudorandom function is used: `F(Prefix, Net_Iface, Network_ID, DAD_Counter, key)`

- `Prefix`: Learned from an ICMPv6 Router Advertisement (RA) message or the Link-Local unicast prefix
- `Net_Iface`: ID associated with the network interface (e.g. `wlan0`)
- `Network_ID`: ID associated with the network. E.g., the WLAN Service Set Identifier (SSID)
- `DAD_Counter`: For resolving Duplicate Address Detection (DAD) conflicts. The initial value is `0`. Incremented by 1 for each new address that is configured
- `key`: 128 bits long secret key

SLAAC Extension: Stable Privacy (RFC 7217) – (3/3)

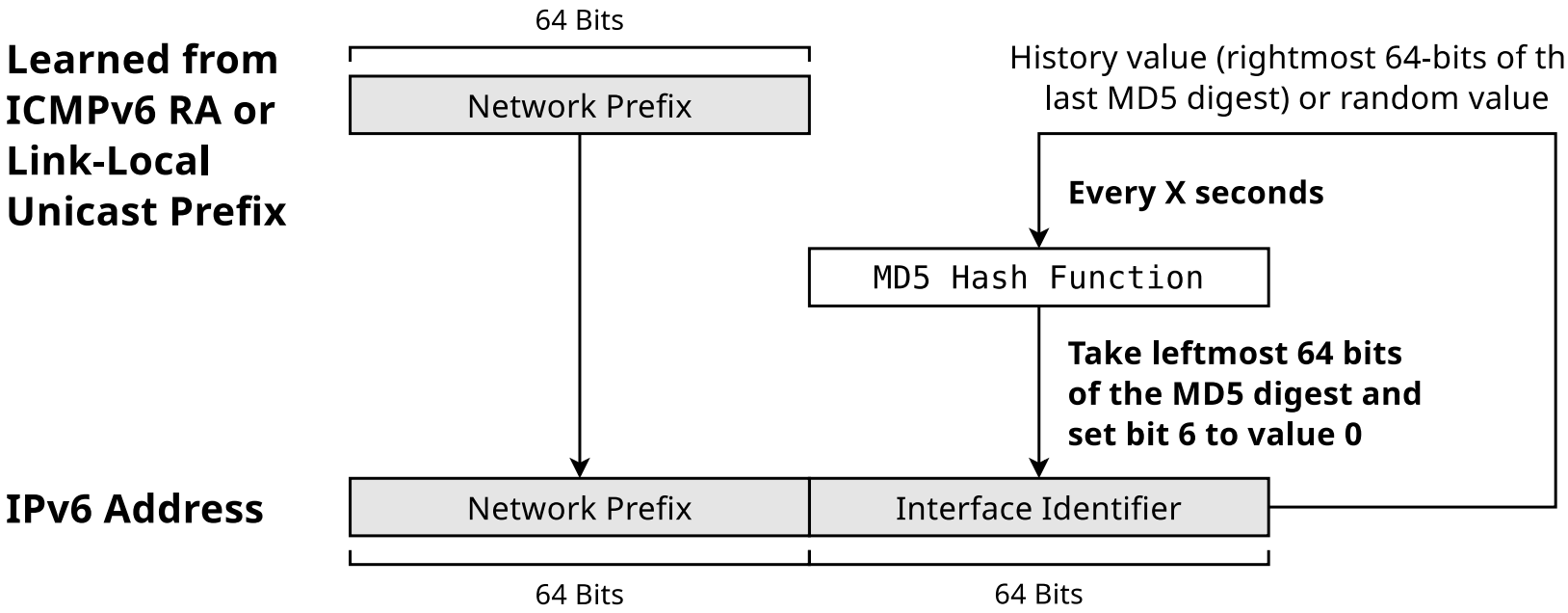


- SHA-1 and SHA-256 are two possible options for `F()`
  - But not `MD5` (see RFC 6151)

Example of a generated address with stable-privacy

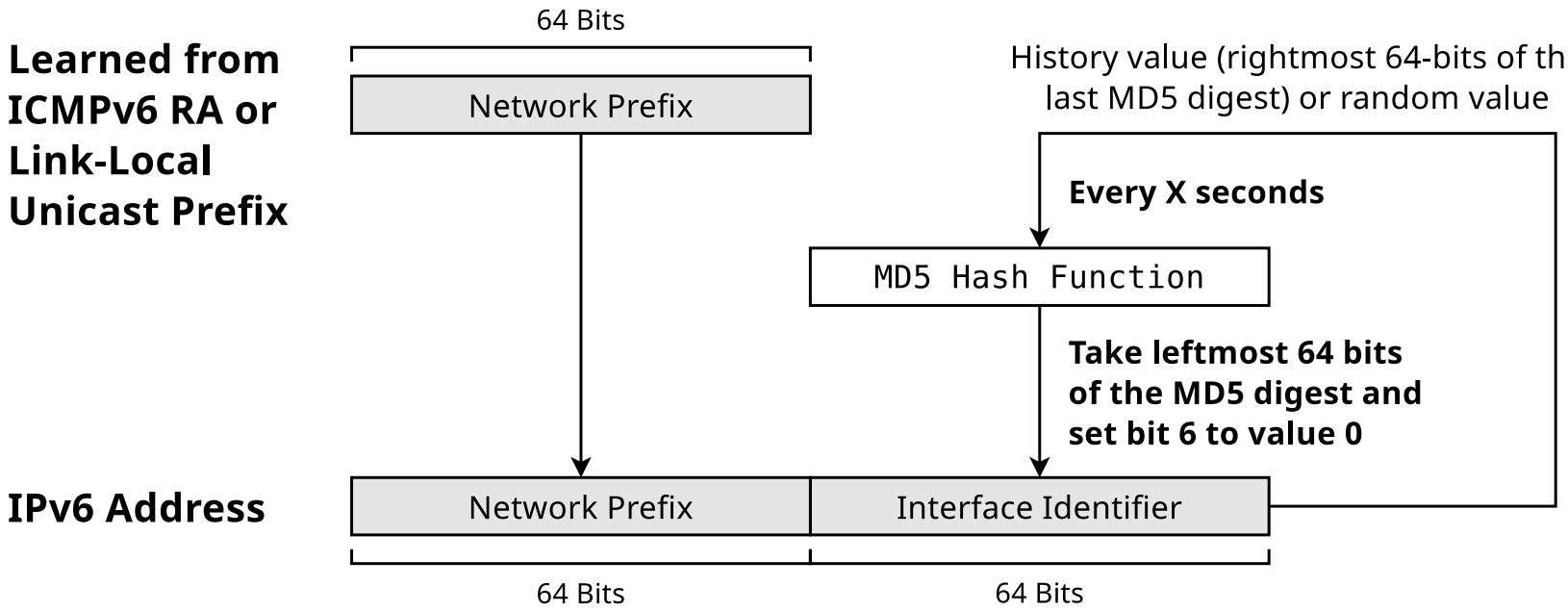
```
MAC: 86:3a:ea:8a:a7:d9
stable-privacy    -> inet6 fe80::6f6d:80e:ab6c:65a0/64
link local EUI-64 -> inet6 fe80::843a:eaff:fe8a:a7d9/64
```

SLAAC Extension: Privacy Extension (RFC 4941) – (1/2)



- **Optional extension of SLAAC** (Stateless Address Autoconfiguration)
- Specifies the address generation with a random number
  - It uses the Interface-ID in a temporary manner
    - A new Interface-ID gets generated periodically
    - Old Interface-IDs can still be used for established connections
  - Benefit: No MAC address is used $\Longrightarrow$ Improved security + anonymity
    - The MAC address of the node is not exposed
  - Drawback: The address expires $\Longrightarrow$ it is not stable

SLAAC Extension: Privacy Extension (RFC 4941) – (2/2)



- Every time, a new address is generated, the node must perform duplicate address detection (DAD)
  - If the address is already in use, the node must generate a new randomized interface identifier

Example of a random generated address with privacy-extension

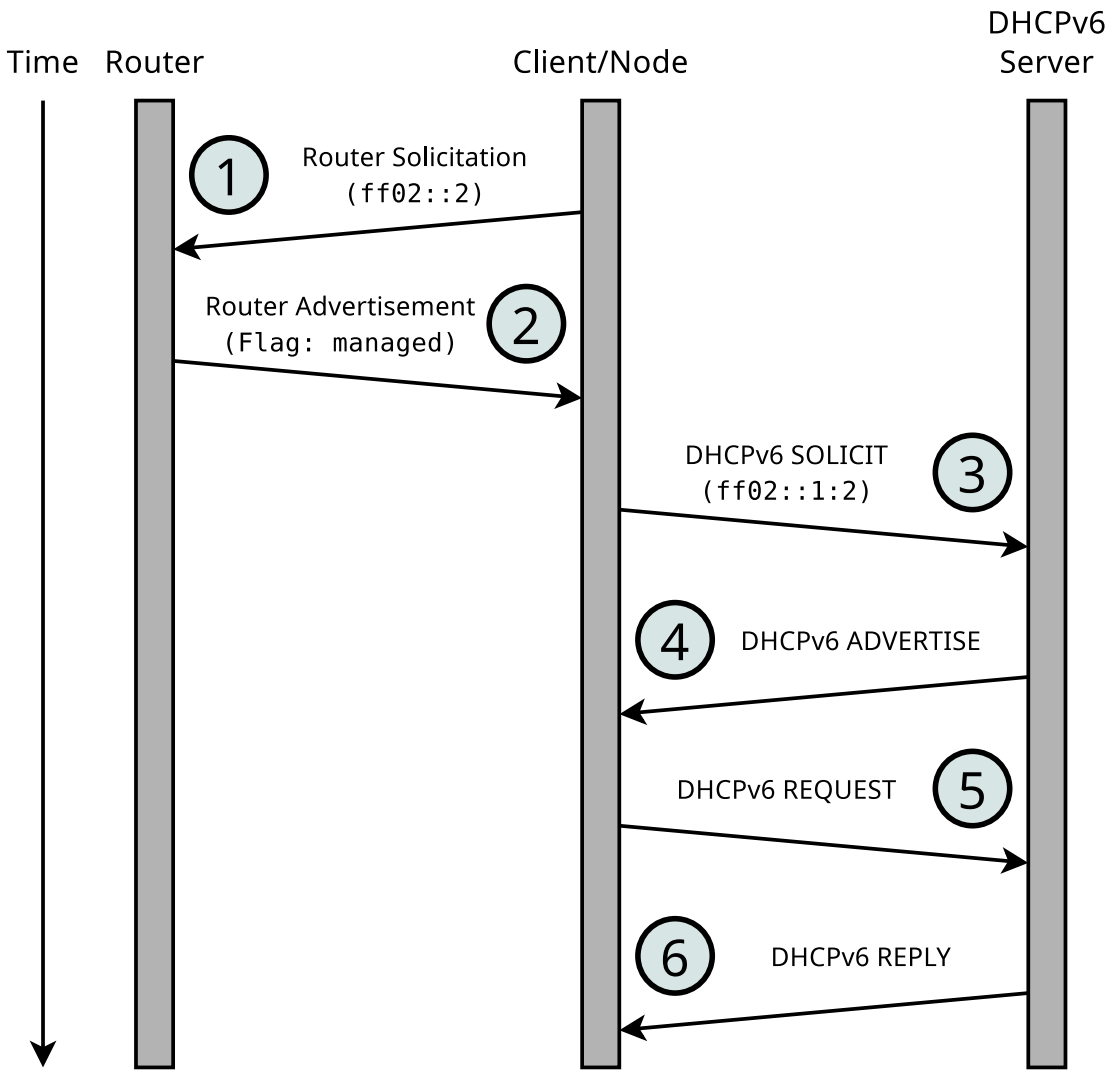
```
MAC: 86:3a:ea:8a:a7:d9
privacy-extension -> inet6 fd12::8992:3c03:d6e2:ed72/64
link local -> inet6 fe80::843a:eaff:fe8a:a7d9/64
```

Random generated Interface-ID

The address shown above is generated randomly and temporary and cannot be traced back to any node characteristics

DHCPv6 (RFC 8415) – (1/2)

1. Node requests a prefix for a globally valid address via the RS message send to the multicast address `ff02::2` (all Routers)
2. Router replies with an RA message that has the `managed` flag set
3. Node sends a `DHCPv6 SOLICIT` message to the multicast address `ff02::1:2` (all DHCPv6 servers)
4. All DHCPv6 servers in range reply with a `DHCPv6 ADVERTISE` message that includes a network configuration (DNS server, NTP server, a prefix for the global valid address,. . .)



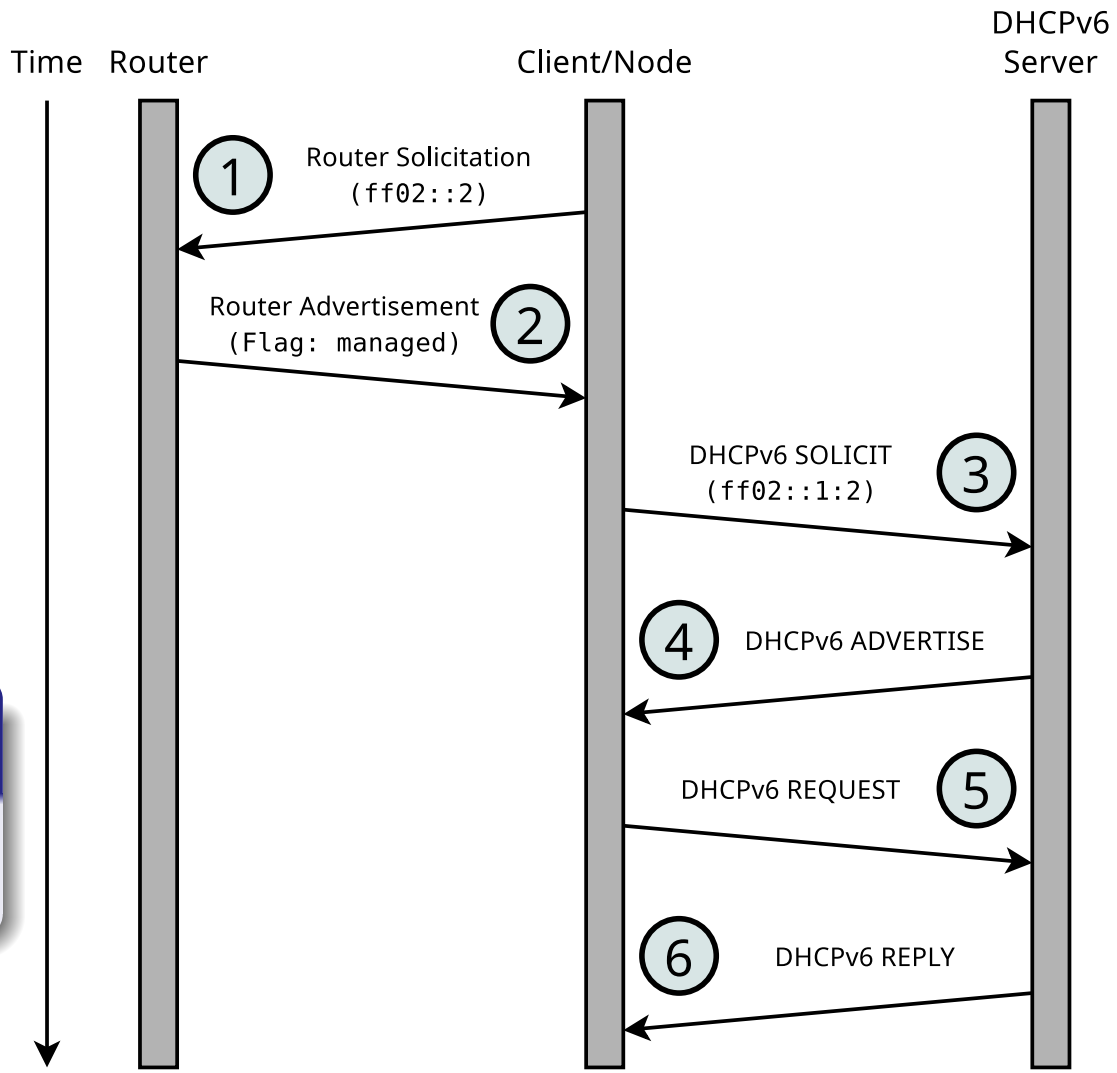
DHCPv6 is the only **stateful** IPv6 address configuration option

DHCPv6 (RFC 8415) – (2/2)

5. Node selects one configuration offer and requests it with a `DHCPv6 REQUEST` message
6. DHCPv6 server marks the IP in its address pool as assigned with the Client ID and acknowledges the request with a `DHCPv6 REPLY` message

DHCPv6 and DHCP (for IPv4) are both Application Layer protocols

DHCPv6 uses UDP via the ports 547 (server or relay agent) and 546 (client)



Embed IPv4 Addresses into IPv6 (*IPv4 mapped*)

- A globally routed (unicast) IPv4 address can be represented as an IPv6 address and thus integrated into the IPv6 address space
  - In literature, this approach is called *IPv4 mapped*
- The IPv4 address gets a 96 Bits long prefix:
  `0:0:0:0:0:FFFF::/96`

80 Bits					16 Bits	32 Bits
0000	0000	0000	0000	0000	FFFF	IPv4 address

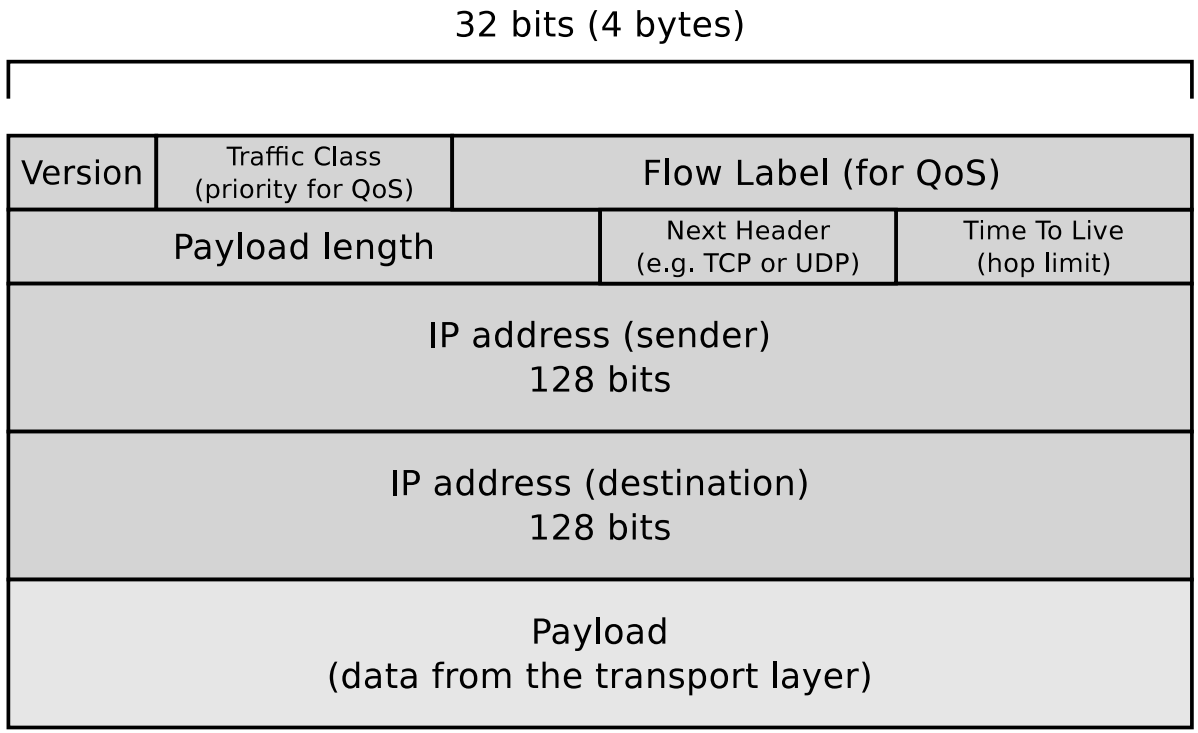
- The IPv4 address may be represented in hexadecimal or decimal notation
- Example

IPv4 address:	`131.246.107.35`
IPv6 address:	`0:0:0:0:0:FFFF:83F6:6B23`
Shorter notation:	`::FFFF:83F6:6B23`
	`::FFFF:131.246.107.35`

IPv4 is useful e.g. for proxy scenarios like NAT64 (RFC 6146) that allows IPv6-only clients to communicate with IPv4 servers

Structure of IPv6 Packets

- The size of the IPv6 header is fixed (320 bits $\Longrightarrow$ 40 bytes)



- The field **next header** points to an extension header field or identifies the Transport Layer protocol (e.g. TCP = type 6 or UDP = type 17) which is carried in the payload of the packet

Concept: Simplified (reduced) packet structure, but simple option to add additional (new) features with a chain of extension headers

IPv6 extension headers (see RFC 2460 and RFC 4303) are not discussed in this course