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

Computer Networking and IT Security (INHN0012)

Tutorial 7

Problem 1 Subnetting

TUMexam AG is assigned the address ranges 131.159.32.0/22 and 131.159.36.0/24. The subdivision of those address ranges is left up to TUMexam AG. After a careful analysis the following requirements for the subnets and the minimal number of **usable** IP addresses are determined:

Subnet	NET 1	NET 2	NET 3	NET 4	NET 5
IPs	300	300	15	40	4

The IP address needed in for the router interface is included in those numbers.

a) Write down each first and last IP address of both given address ranges.

- 131.159.32.0/22:
  first IP: 131.159.32.0 (network address)
  last IP: 131.159.35.255 (Broadcast address)
- 131.159.36.0/24:
  first IP: 131.159.36.0 (network address)
  last IP. 131.159.36.255 (Broadcast address)

b) How many IP addresses does TUMexam AG have available? Can all of them be used to address hosts?

- 131.159.32.0/22: $2^{32-22} = 2^{10} = 1024$ addresses
- 131.159.36.0/24: $2^{32-24} = 2^{8} = 256$ addresses

There are a total of $1024 + 256 = 1280$ addresses available. However the first (network) and the last (broadcast) address of each network is not usable to address hosts. Therefore there are at maximum $1022 + 254 = 1276$ addresses available for host addressing.

c)* Is it possible to aggregate both blocks of address ranges into one single subnet?

No. The subnets are of different sizes (/22 and /24) and can therefore not be aggregated because increasing the /22 to a /21 prefix would include way more networks.
(A single subnet has always a power of two as its size, we would need one with $1024 + 256 = 1280$ addresses.)
Note: The criterion stated above is necessary, but not sufficient. Two subnets of equal size can only be aggregated iff one is immediately after the other **and** there exists a valid prefix that describes the resulting network. (The last criterion is equivalent to a parent node for both subnets, if one imagines the address space as a binary tree)

d) Divide both address ranges according to the analysis in order to get subnets with fitting sizes. Use as little IP addresses as possible. A large continuous address range should remain available for future use. For every subnet you should indicate:

- the size of n-th subnet
- the amount of usable addresses
- the subnet in prefix notation
- the subnetmask in dotted-decimal-notation
- the network and broadcast addresses

Subnet	NET 1	NET 2	NET 3
Requirement	300	300	15
Size	512	512	32
Usable	510	510	30
Prefix notation	131.159.32.0/23	131.159.34.0/23	131.159.36.64/27
Subnetmask	255.255.254.0	255.255.254.0	255.255.255.224
Network address	131.159.32.0	131.159.34.0	131.159.36.64
Broadcast	131.159.33.255	131.159.35.255	131.159.36.95

Subnet	NET 4	NET 5
Requirement	40	4
Size	64	8
Usable	62	6
Prefix notation	131.159.36.0/26	131.159.36.96/29
Subnetmask	255.255.255.192	255.255.255.248
Network address	131.159.36.0	131.159.36.96
Broadcast	131.159.36.63	131.159.36.103

To fulfill all the requirements we have to handle the subnets in order of decreasing size. Otherwise we could get into the following situation:

- Network 3 receives the address range 131.159.36.0/27.
- If we would now give network 4 the range 131.159.36.32/26 we would make a mistake. To understand this we shall take a look at the binary notation of last octet of the network address and the subnetmask: `131.159. 36.0010 0000` (IP)
  `255.255.255.1100 0000` (subnet mask)
  A bitwise AND of both numbers shows that the IP address 131.159.36.32 is part of the 131.159.36.0/26 subnet!
- We would have to give the range 131.159.36.64/26 to network 4; but then we would be left with a gap between networks 3 and 4.
- If we give the addresses out in order of decreasing subnet size, we can work around this problem. This procedure could however again contradict other criteria – for example giving out continuous address ranges to single campi.

Problem 2 Neighbor Discovery Protocol and IP fragmentation with IPv6

Figure 2.1 shows an arrangement of network components with their MAC addresses. PC1 and PC2 are assigned both link-local (LL) and global-unique (GU) addresses by means of SLAAC. For the latter, the prefix `2001:db8:1::/64` (PC1/R1) or `2001:db8:2::/64` (PC2/R2) is used.
PC1 sends an IP packet with 1400 B of data to PC2. The MTU on the WAN link between R1 and R2 is 1280 B[1]. Within the local networks, the MTU is 1500 B, as is typical for Ethernet.

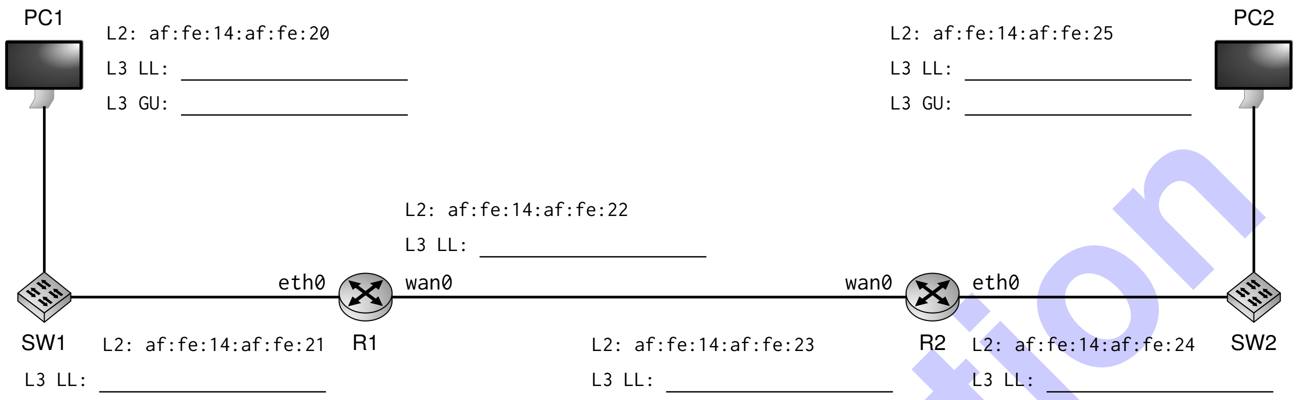


Figure 2.1: Network topology

First, we take a look at the address configuration using SLAAC.

a)* Determine the link-local addresses of all interfaces.

See lecture:

- PC1: `af:fe:14:af:fe:20` → `fe80::adfe:14ff:feaf:fe20`
- R1.eth0: `af:fe:14:af:fe:21` → `fe80::adfe:14ff:feaf:fe21`
- R1.eth1: `af:fe:14:af:fe:22` → `fe80::adfe:14ff:feaf:fe22`
- R2.eth1: `af:fe:14:af:fe:23` → `fe80::adfe:14ff:feaf:fe23`
- R2.eth0: `af:fe:14:af:fe:24` → `fe80::adfe:14ff:feaf:fe24`
- PC2: `af:fe:14:af:fe:25` → `fe80::adfe:14ff:feaf:fe25`

Note: the seventh bit of the first octet of the MAC address is inverted during the generation.
Reason: Manually assigned IPv6 addresses often have an interface identifier of the form `::abcd`, i.e. the first 48 bit are 0. If we infer the underlying MAC address from such an IPv6 address, the second to last bit of its first octet would be 0, which would indicate a globally unique MAC address – which is obviously wrong, since it comes from a manually assigned IPv6 address. If this bit were not inverted, all manually assigned interface identifiers would have to be of the form `2001:db8:1:0:200::1` (or, if the unique abbreviation of multiple null groups was already in the subnet identifier, of the form `2001:db8::200:0:0:1`).

[1] This is the minimum MTU that must be supported by layer 2 according to RFC 2460 for IPv6 support.

b) Determine the global unique IPv6 addresses of PC1 and PC2. Assume that Router R1 is configured with the prefix `2001:db8:1::/64` and Router R2 is configured with `2001:db8:2::/64`.

The derivation is analogous to the link local addresses, but with the prefix of the respective router, which are made known via router advertisements PC1 and PC2.

- `af:fe:14:af:fe:20` → `2001:db8:1:0:adfe:14ff:feaf:fe20`
- `af:fe:14:af:fe:25` → `2001:db8:2:0:adfe:14ff:feaf:fe25`

c)* At which point in the network does fragmentation happen?

Directly at PC1 because with IPv6 routers do not fragment.

d)* What is the minimum number of fragments the packet must be split into?

The MTU (Maximum Transmission Unit) is the maximum size of a packet on layer 3 incl. header. It therefore corresponds exactly to the maximum size of the payload on layer 2. In the case of fragmentation, the individual fragments will each carry an IPv6 header of length 40 B as well as a fragment header of length 8 B. Provided that no further extension headers are used, we thus obtain:

$$N = \left\lceil \frac{1400\,\text{B}}{1280\,\text{B} - 40\,\text{B} - 8\,\text{B}} \right\rceil = 2$$

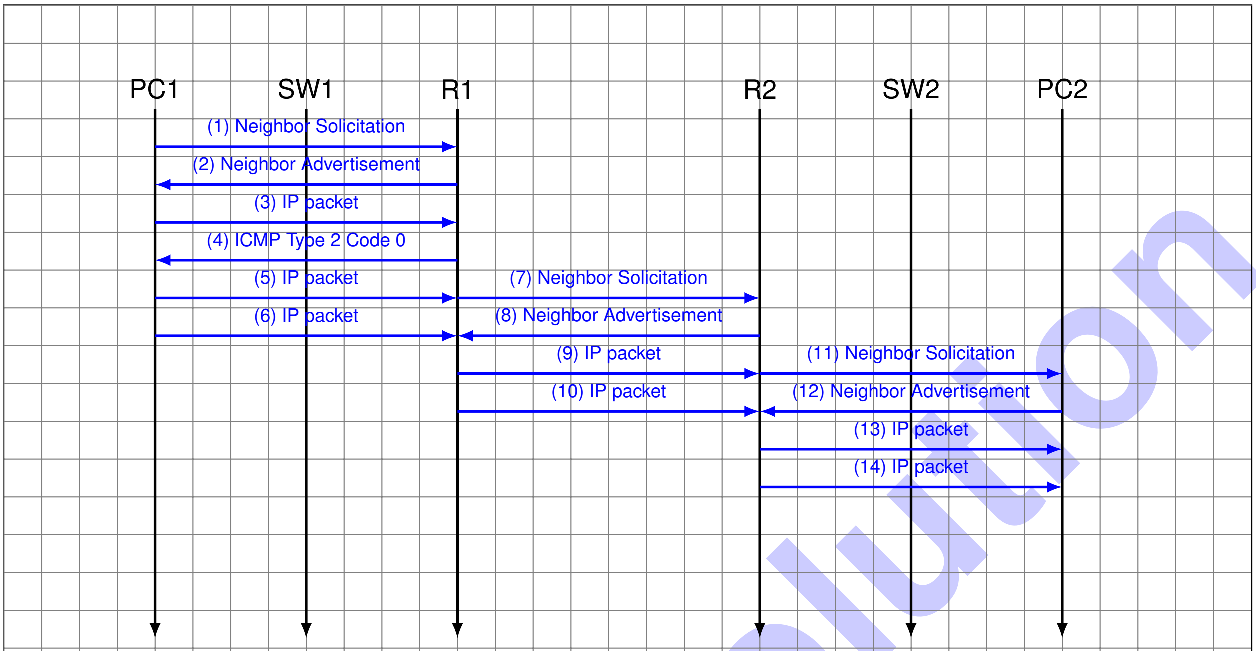
e) Determine the size of the L3-SDU for each fragment.

Each fragment can carry 1280 B − 40 B − 8 B = 1232 B of user-data.
Since this is also a multiple of eight (fragment offset is specified in multiples of 8 B), this also corresponds to the actual amount of user data that can be transferred.
Therefore the first fragment has a payload of 1232 B and the second has a payload of 168 B.

f)* Justify at what point in the network will the fragments be reassembled.

Only the receiver, here PC2, reassembles the fragments. In fact, generally no other node can perform the reassembly because the fragments each represent individual and independent packets. In particular, this means that they are routed independently of each other and therefore on occasion can take different paths to the destination – this is not possible in the simple example in Figure 2.1, where there is only one path between PC1 and PC2.

g) Sketch a simple path-time diagram that considers **all frames** that need to be transferred on each link. **Name the type of frames replaced and give the frames numbers (1,2,3,...).** (The diagram does not need to be to scale. Serialization times and propagation delays are to be ignored).
Assume that no mappings between IP and MAC addresses are currently cached.
Number each packet by column (column $\triangleq$ range e.g. between R1 and R2).



Packet (4) is an ICMP error message "Packet too big", which also contains the MTU of the following link segment. PC1 can then locally fragment accordingly.

h) Determine the destination MAC address of the first transmitted frame.

The frame contains an IPv6 Neighbor Sollicitation packet. According to the lecture those are sent to the corresponding sollicited node. Since these are multicast addresses, the destination MAC address is calculated from the solicited-node address.
The MAC address for R1 with the IP `fe80::adfe:14ff:feaf:fe21` must be queried. The solicited node prefix `ff02::1:ff00:0/104` and the last 24 bits of the destination address result in the solicited node address `ff02::1:ffaf:fe21`. This is in the prefix `ff00::/8` and is therefore a multicast address. Therefore, the associated MAC is calculated from the prefix `33:33` and the last 32 bits of the destination address `ff02::1:ffaf:fe21`, resulting in `33:33:ff:af:fe:21`.

Homework:

At the end of this exercise sheet you will find preprinted forms for Ethernet header, IPv6 header and ICMPv6/NDP header (more than needed). It is not necessary to fill in the header in binary. Just be sure to clearly mark the number base, e.g. `0x10` for hexadecimal or $63_{(10)}$ for decimal.

i) For each of the first two frames from subtask g), fill in an Ethernet header, an IP header, and the respective payload. Label the dashed box next to each header/packet with the corresponding frame number.
Note: Use the cheatsheet to determine the values (e.g. Next Header). If a value is not clearly determined, make a reasonable choice.

j) For each path segment (e.g. B. between R1 and R2), fill in the respective first fragmented packet, one Ethernet header and one IP header. off. Label the dashed box next to each header/packet with the respective frame number.
Note: Use the cheatsheet to determine the values (e.g. Next header). If a value is not clearly determined, make a sensible choice.

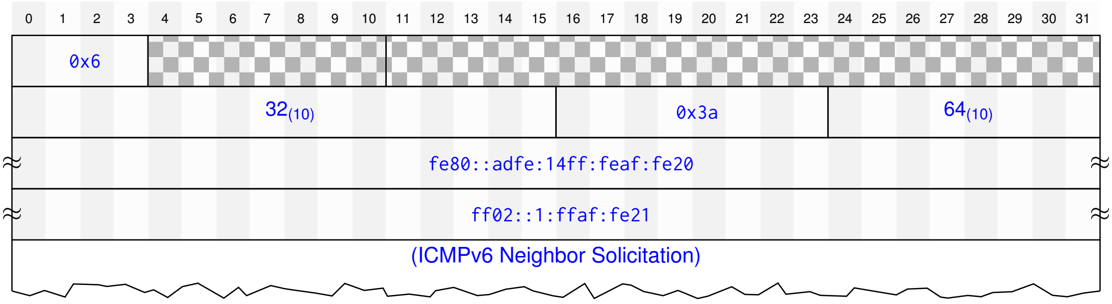
Preprints for protocol headers:

Ethernet frames

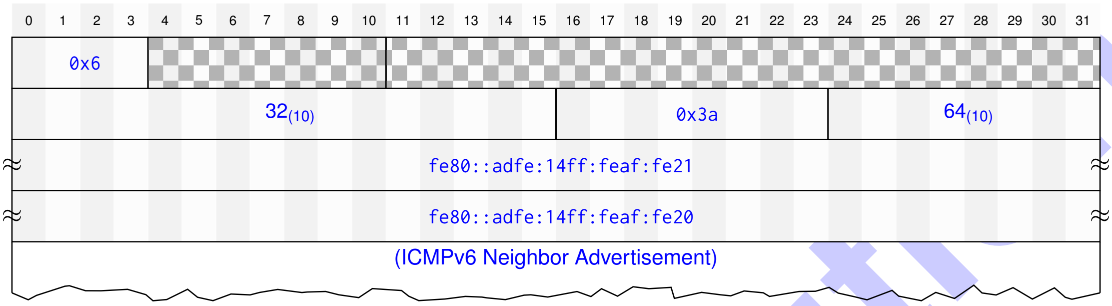
1	33:33:ff:af:fe:21	af:fe:14:af:fe:20	0x86dd	Payload	FCS
2	af:fe:14:af:fe:20	af:fe:14:af:fe:21	0x86dd	Payload	FCS
5	af:fe:14:af:fe:21	af:fe:14:af:fe:20	0x86dd	Payload	FCS
9	af:fe:14:af:fe:23	af:fe:14:af:fe:22	0x86dd	Payload	FCS
13	af:fe:14:af:fe:25	af:fe:14:af:fe:24	0x86dd	Payload	FCS
				Payload	FCS
				Payload	FCS
				Payload	FCS
				Payload	FCS
				Payload	FCS
				Payload	FCS
				Payload	FCS
				Payload	FCS
				Payload	FCS

IPv6 Header

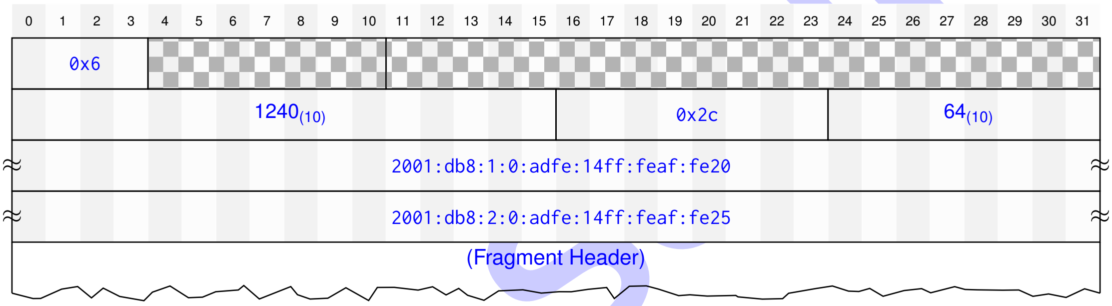
1



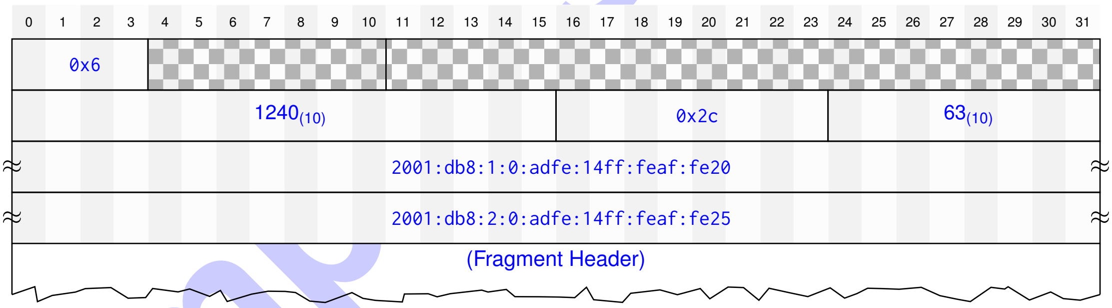
2



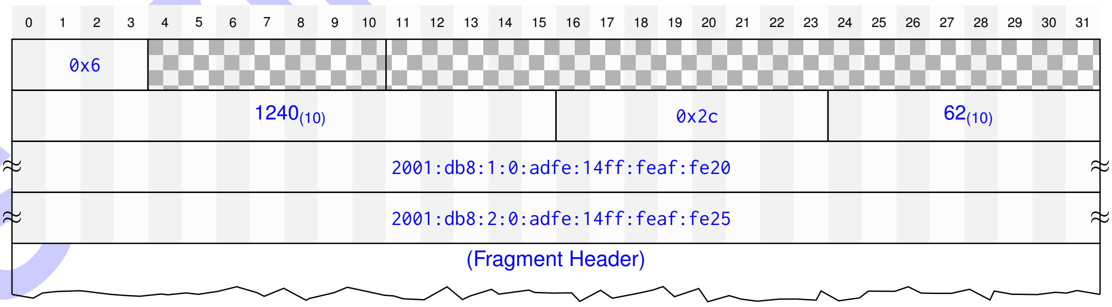
5

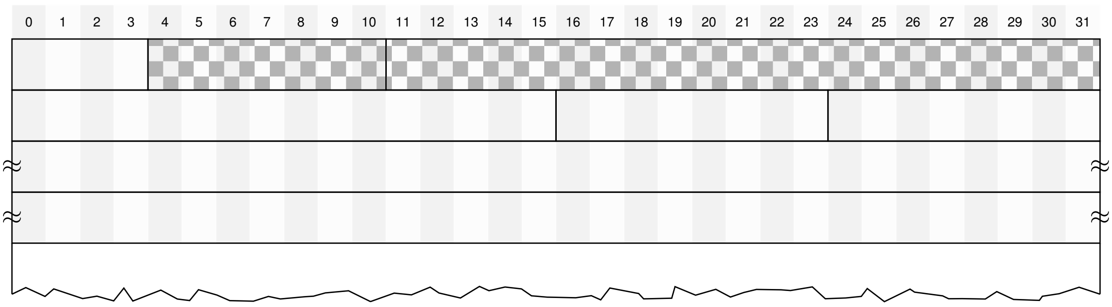


9



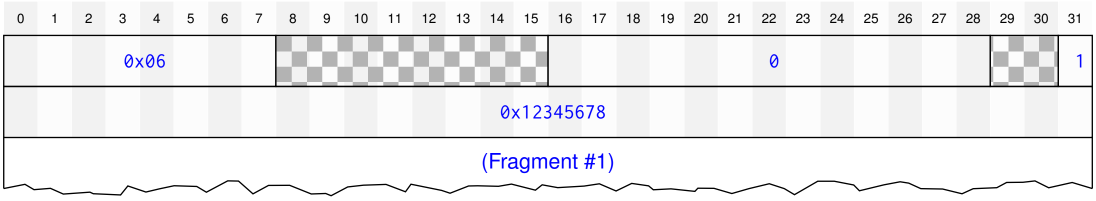
13

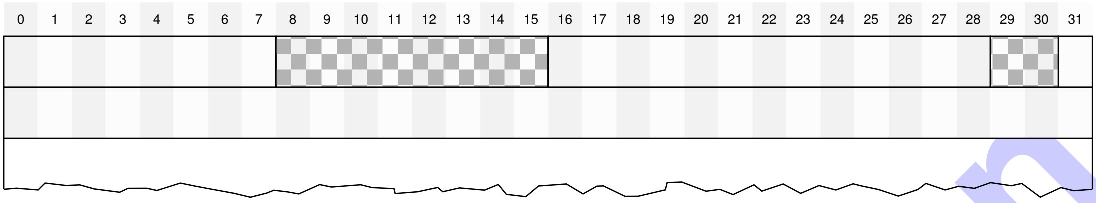


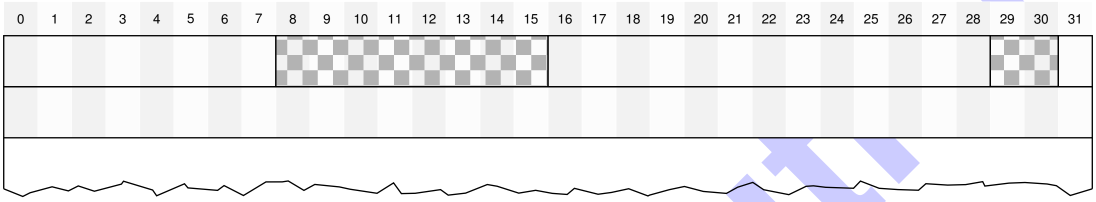


IPv6 Fragment Header

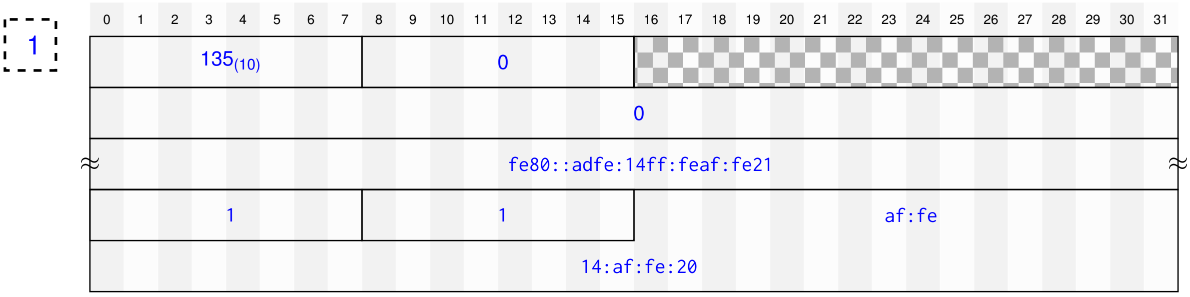
5

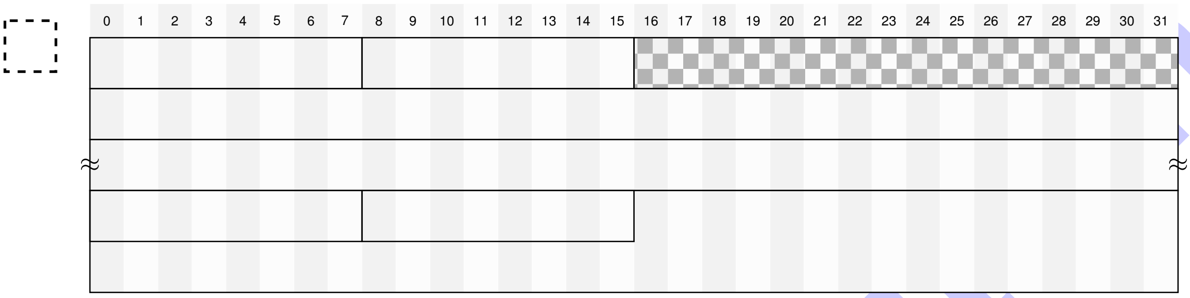






ICMPv6 Neighbor Solicitation





ICMPv6 Neighbor Advertisement