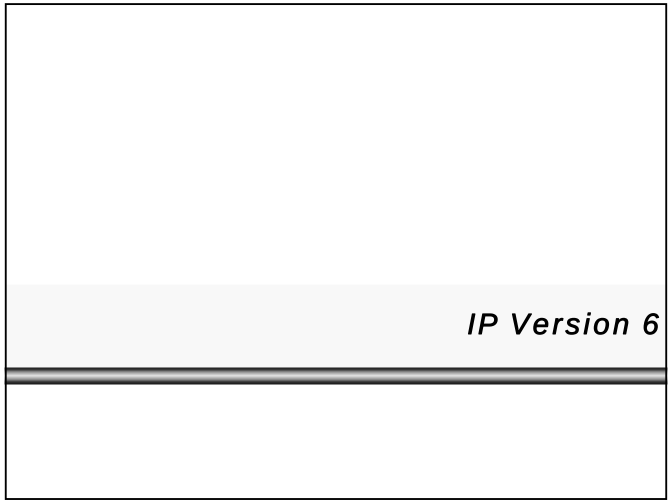
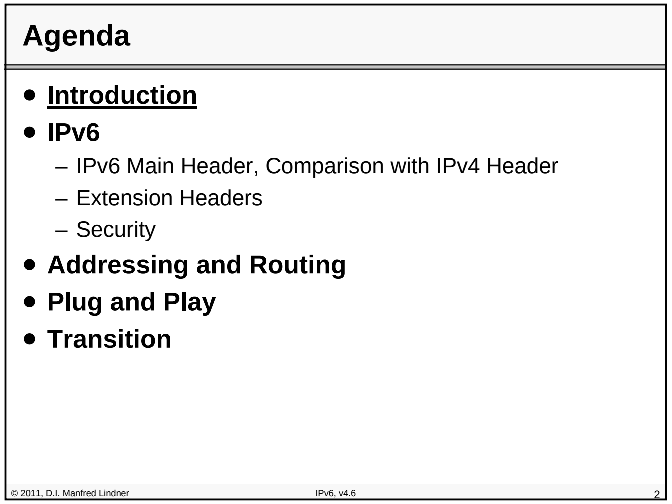
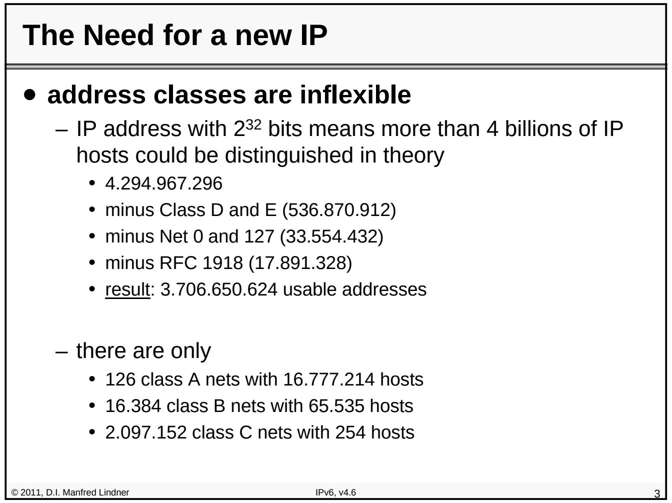


IP Version 6



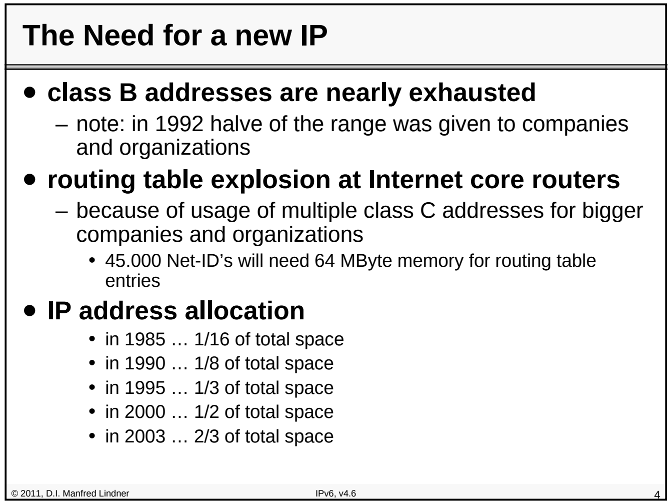
Agenda

- **<u>Introduction</u>**
- **IPv6**
  - IPv6 Main Header, Comparison with IPv4 Header
  - Extension Headers
  - Security
- **Addressing and Routing**
- **Plug and Play**
- **Transition**



The Need for a new IP

- **address classes are inflexible**
  - IP address with 2^{32} bits means more than 4 billions of IP hosts could be distinguished in theory
    - 4.294.967.296
    - minus Class D and E (536.870.912)
    - minus Net 0 and 127 (33.554.432)
    - minus RFC 1918 (17.891.328)
    - <u>result</u>: 3.706.650.624 usable addresses
  - there are only
    - 126 class A nets with 16.777.214 hosts
    - 16.384 class B nets with 65.535 hosts
    - 2.097.152 class C nets with 254 hosts



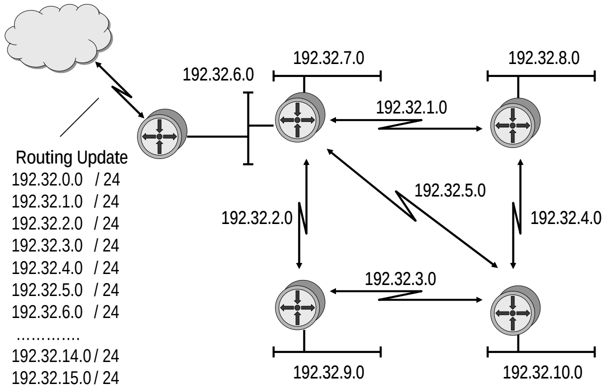
The Need for a new IP

- **class B addresses are nearly exhausted**
  - note: in 1992 halve of the range was given to companies and organizations
- **routing table explosion at Internet core routers**
  - because of usage of multiple class C addresses for bigger companies and organizations
    - 45.000 Net-ID's will need 64 MByte memory for routing table entries
- **IP address allocation**
    - in 1985 … 1/16 of total space
    - in 1990 … 1/8 of total space
    - in 1995 … 1/3 of total space
    - in 2000 … 1/2 of total space
    - in 2003 … 2/3 of total space

Temporary (Short-term) Solution

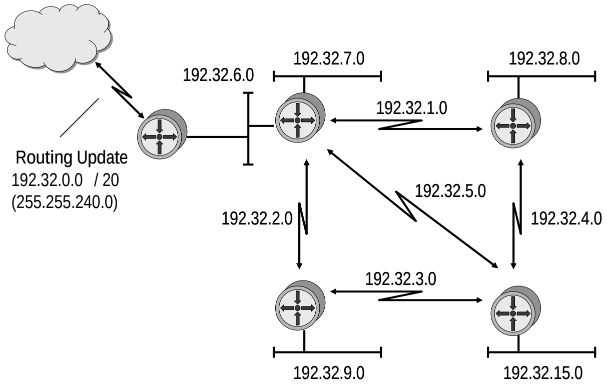
- **Classless Inter-Domain Routing (CIDR)**
  - address assignment and aggregation (route summarization) strategy for blocks of networks with contiguous addresses
  - creative IP address allocation
    - addressing plan for class C addresses by continents
    - provider based addressing strategy
  - classless routing (prefix, length)
  - supernetting of class C network numbers blocks
  - VLSM
- **private addressing and network address translation (NAT)**
  - see RFC 1918 "Address Allocation for Private Internets"

Routing Updates without Supernetting (Classful Behavior)



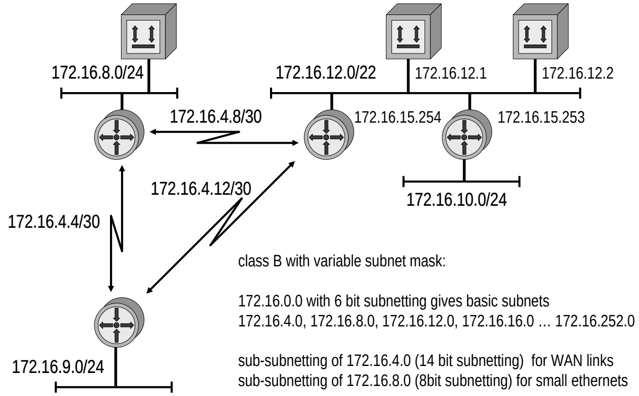
Routing Update
192.32.0.0 / 24
192.32.1.0 / 24
192.32.2.0 / 24
192.32.3.0 / 24
192.32.4.0 / 24
192.32.5.0 / 24
192.32.6.0 / 24
…………
192.32.14.0 / 24
192.32.15.0 / 24

Route Summarization with Supernetting (Classless Behavior)



Routing Update
192.32.0.0 / 20
(255.255.240.0)

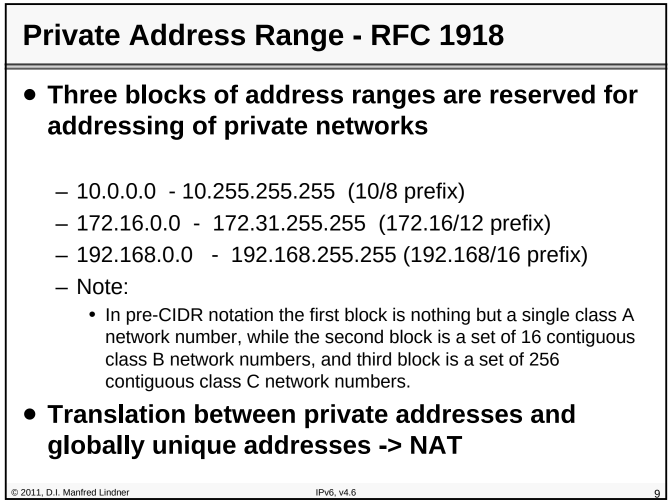
Variable Length Subnet Mask (VLSM)



class B with variable subnet mask:

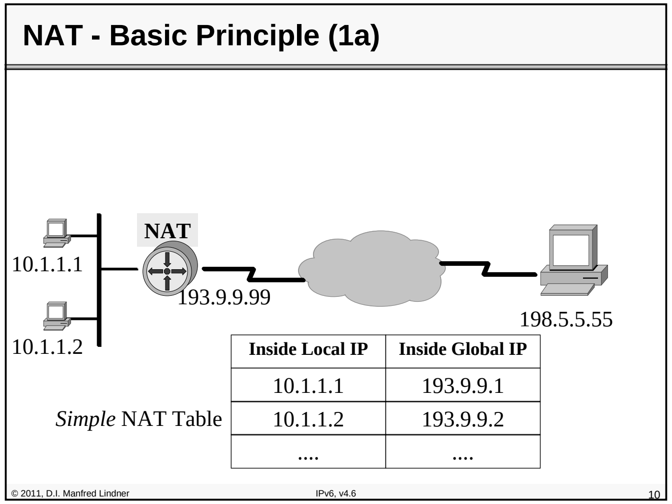
172.16.0.0 with 6 bit subnetting gives basic subnets
172.16.4.0, 172.16.8.0, 172.16.12.0, 172.16.16.0 … 172.16.252.0

sub-subnetting of 172.16.4.0 (14 bit subnetting) for WAN links
sub-subnetting of 172.16.8.0 (8bit subnetting) for small ethernets



Private Address Range - RFC 1918

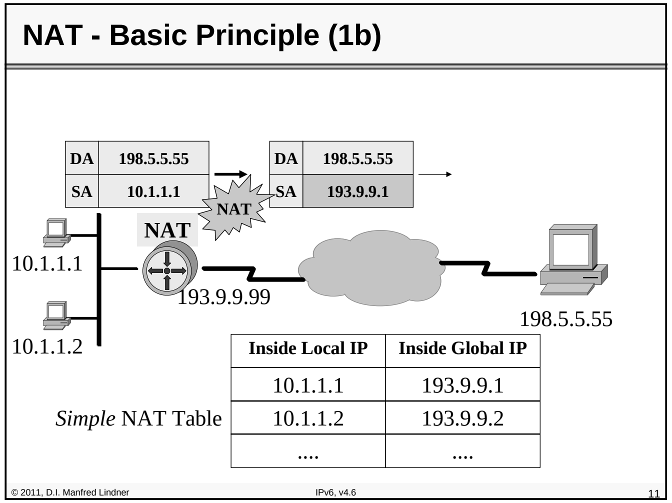
- **Three blocks of address ranges are reserved for addressing of private networks**
  - 10.0.0.0 - 10.255.255.255 (10/8 prefix)
  - 172.16.0.0 - 172.31.255.255 (172.16/12 prefix)
  - 192.168.0.0 - 192.168.255.255 (192.168/16 prefix)
  - Note:
    - In pre-CIDR notation the first block is nothing but a single class A network number, while the second block is a set of 16 contiguous class B network numbers, and third block is a set of 256 contiguous class C network numbers.
- **Translation between private addresses and globally unique addresses -> NAT**



NAT - Basic Principle (1a)

Simple NAT Table

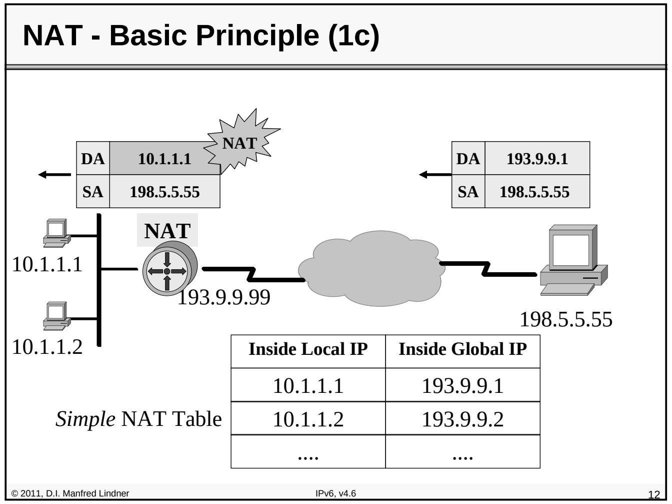
Inside Local IP	Inside Global IP
10.1.1.1	193.9.9.1
10.1.1.2	193.9.9.2
....	....



NAT - Basic Principle (1b)

Simple NAT Table

Inside Local IP	Inside Global IP
10.1.1.1	193.9.9.1
10.1.1.2	193.9.9.2
....	....



NAT - Basic Principle (1c)

Simple NAT Table

Inside Local IP	Inside Global IP
10.1.1.1	193.9.9.1
10.1.1.2	193.9.9.2
....	....

NAT Drawbacks

- **No end-to-end IP possible**
  - sufficient for client/server model
  - Not sufficient for peer-to-peer networking
- **Network component (NAT device) needs to maintain state of connection**
- **Fast rerouting difficult if NAT router fails**
- **Makes development of new application difficult (“NAT friendly?”)**
- **Security (end-to-end)**
- **Manageability**

Long-term Solution -> New IP

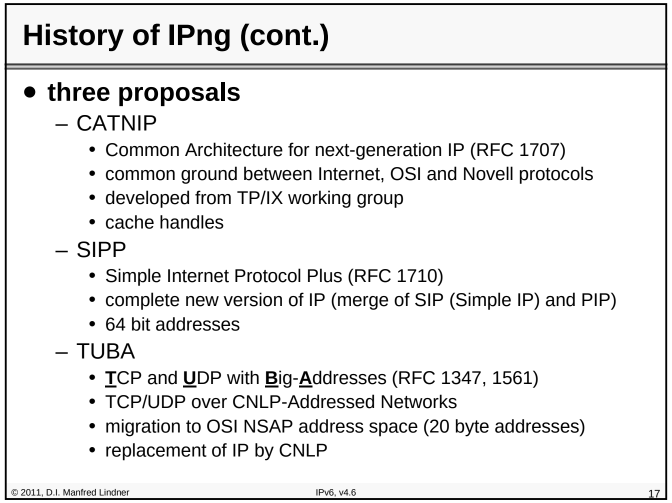
- **short-term solution**
  - postpones exhaustion of IP addresses
  - gives enough time for development of new IP
- **new IP**
  - should covering not only address issues
  - but also other known weaknesses of protocol suite
    - security
      - authentication, privacy, integrity
    - auto-configuration (plug and play)
    - high speed networks
    - quality of service (QoS)
    - mobility
    - real time traffic and multimedia
    - etc.

History of IPng (next generation)

- **November 1991**
  - Routing and Addressing (ROAD) workgroup formed
- **March 1992**
  - ROAD report
    - do CIDR
    - issues call for IPng proposals
- **July 1992**
  - IAB issues “IP version 7” (TP/IX, RFC 1475)
    - intention for new IP and TCP, 64 bit addresses, admin domain number, forward route identifier (flow), new routing protocol RAP
  - IETF issues call for IPng proposals

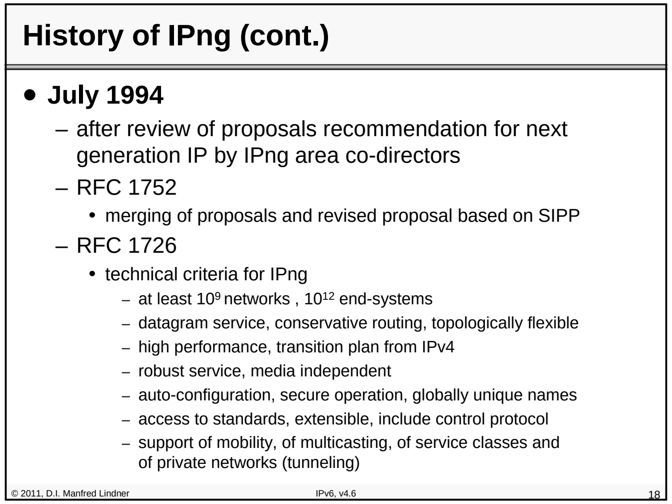
History of IPng (cont.)

- **July 1993**
  - IPv7 refused by IESG
  - new solution should cover not only addressing aspects but also other weaknesses of IP
    - e.g. security, plug and play, etc.
- **August 1993**
  - IETF area formed to consolidate IPng activity
    - Allison Markin and Scott Bradner area co-directors
- **December 1993**
  - RFC 1550 “IP: Next Generation (IPng) White Paper Solicitation”
    - input and answers: RFC 1667-1680, 1682/83, 1686-88, 1705, 1707, 1710, 1715



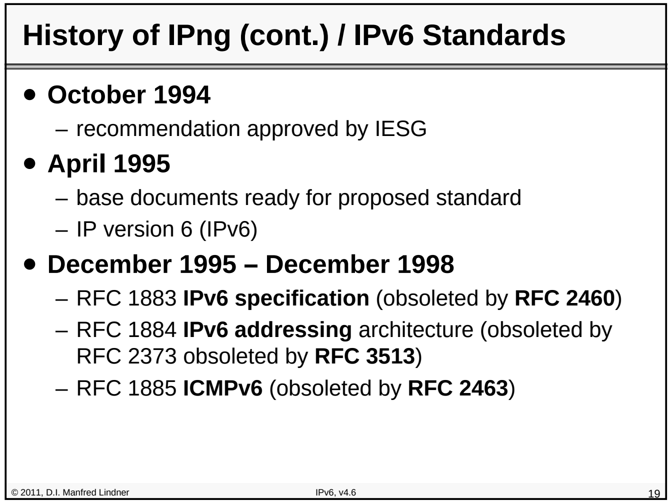
History of IPng (cont.)

- **three proposals**
  - CATNIP
    - Common Architecture for next-generation IP (RFC 1707)
    - common ground between Internet, OSI and Novell protocols
    - developed from TP/IX working group
    - cache handles
  - SIPP
    - Simple Internet Protocol Plus (RFC 1710)
    - complete new version of IP (merge of SIP (Simple IP) and PIP)
    - 64 bit addresses
  - TUBA
    - **T**CP and **U**DP with **B**ig-**A**ddresses (RFC 1347, 1561)
    - TCP/UDP over CNLP-Addressed Networks
    - migration to OSI NSAP address space (20 byte addresses)
    - replacement of IP by CNLP



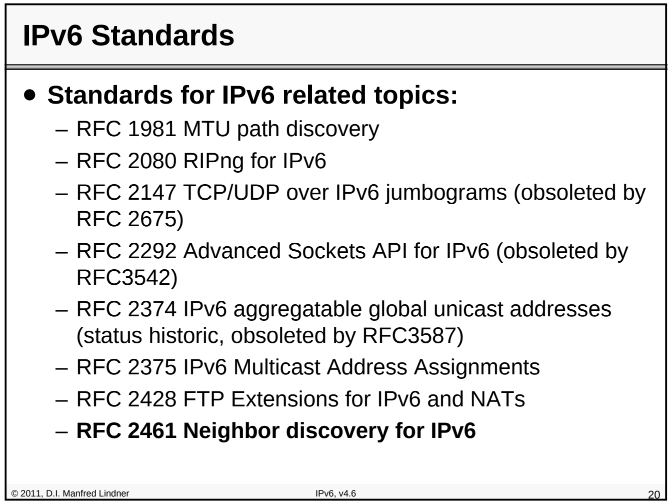
History of IPng (cont.)

- **July 1994**
  - after review of proposals recommendation for next generation IP by IPng area co-directors
  - RFC 1752
    - merging of proposals and revised proposal based on SIPP
  - RFC 1726
    - technical criteria for IPng
      - at least 10^9 networks , 10^{12} end-systems
      - datagram service, conservative routing, topologically flexible
      - high performance, transition plan from IPv4
      - robust service, media independent
      - auto-configuration, secure operation, globally unique names
      - access to standards, extensible, include control protocol
      - support of mobility, of multicasting, of service classes and of private networks (tunneling)



History of IPng (cont.) / IPv6 Standards

- **October 1994**
  - recommendation approved by IESG
- **April 1995**
  - base documents ready for proposed standard
  - IP version 6 (IPv6)
- **December 1995 – December 1998**
  - RFC 1883 **IPv6 specification** (obsoleted by **RFC 2460**)
  - RFC 1884 **IPv6 addressing** architecture (obsoleted by RFC 2373 obsoleted by **RFC 3513**)
  - RFC 1885 **ICMPv6** (obsoleted by **RFC 2463**)



IPv6 Standards

- **Standards for IPv6 related topics:**
  - RFC 1981 MTU path discovery
  - RFC 2080 RIPng for IPv6
  - RFC 2147 TCP/UDP over IPv6 jumbograms (obsoleted by RFC 2675)
  - RFC 2292 Advanced Sockets API for IPv6 (obsoleted by RFC3542)
  - RFC 2374 IPv6 aggregatable global unicast addresses (status historic, obsoleted by RFC3587)
  - RFC 2375 IPv6 Multicast Address Assignments
  - RFC 2428 FTP Extensions for IPv6 and NATs
  - **RFC 2461 Neighbor discovery for IPv6**

IPv6 Standards (cont.)

- **Standards for IPv6 related topics:**
  - **RFC 2462 IPv6 stateless auto-configuration**
  - RFC 2464 IPv6 over Ethernet
  - RFC 2467 IPv6 over FDDI
  - RFC 2470 IPv6 Packets over Token Ring Nets
  - RFC 2472 IP Version 6 over PPP
  - RFC 2473 Generic Packet Tunneling in IPv6
  - RFC 2491 IPv6 over Non-Broadcast Multiple Access
  - RFC 2492 IPv6 over ATM Networks
  - RFC 2497 IPv6 Packets over ARCnet Networks
  - RFC 2526 Reserved IPv6 Subnet Anycast Addresses

IPv6 Standards (cont.)

- **Standards for IPv6 related topics:**
  - RFC 2529 IPv6 over IPv4 Domains without Explicit Tunnels
  - RFC 2553 Basic socket interface extension for IPv6
  - RFC 2590 IPv6 Packets over Frame Relay nets
  - RFC 2732 Format for Literal IPv6 Addresses in URL's
  - **RFC 2740 OSPF for IPv6**
  - RFC 2765 Stateless IP/ICMP Translation Algorithm (SIIT)
  - RFC 2767 Dual Stack Hosts using the "Bump-In-the-Stack“ Technique (BITS)
  - RFC 2766 Network Address Translation - Protocol Translation (NAT-PT)

IPv6 Standards (cont.)

- **Standards for IPv6 related topics:**
  - RFC 2874 DNS Extensions to Support IPv6
  - RFC 2893 Transition Mechanism IPv6 hosts and routers
  - RFC 2894 Router Renumbering for IPv6
  - RFC 3041 Privacy Extensions for Stateless Address Auto-configuration in IPv6
  - RFC 3053 IPv6 Tunnel Broker

Ongoing Work IPv6

- **drafts for IPv6 related topics:**
  - draft-ietf-ipngwg-icmp-name-lookups-07.txt
  - draft-ietf-ipngwg-site-prefixes-05.txt
  - draft-ietf-ion-ipv6-ind-05.txt
  - draft-ietf-ipngwg-ipaddressassign-02
  - draft-ietf-ipngwg-rfc2292bis-02.txt
  - draft-ietf-ipngwg-default-addr-select-03.txt
  - draft-ietf-ipngwg-addr-arch-v3-05.txt
  - draft-ietf-ipngwg-scoping-arch-02.txt
  - draft-ietf-ipngwg-rfc2553bis-03.txt
  - draft-ietf-ipngwg-ipv6-2260-01.txt
  - draft-ietf-ipngwg-uni-based-mcast-01.txt
  - draft-ietf-ipngwg-dns-discovery-01.txt

Agenda

- **Introduction**
- **IPv6**
  - IPv6 Main Header, Comparison with IPv4 Header
  - Extension Headers
  - Security
- **Addressing and Routing**
- **Plug and Play**
- **Transition**

IPv6 Overview

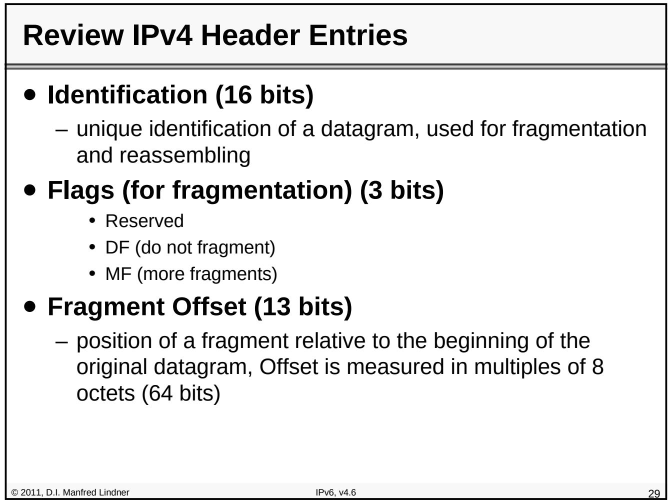
- **16 byte addresses (hexadecimal notation)**
  - mapping algorithms for IPX, NSAP, E.164
- **simple header**
  - only basic functions for fast switching of IPv6 packets
- **extension headers**
  - for advanced or optional functions
- **support of**
  - authentication and privacy (encryption)
  - auto-configuration
  - source routes
  - flow identification (QoS)
- **daisy-chain of headers**

Review IPv4 Header

	0	4	8	16	24	31
0	Version	HLEN	Type of Service	Total Length		
4	Identification			Flags	Fragment Offset	
8	Time to Live (TTL)		Protocol	Header Checksum		
12	Source IP Address					
16	Destination IP Address					
20	Options					
					Padding	

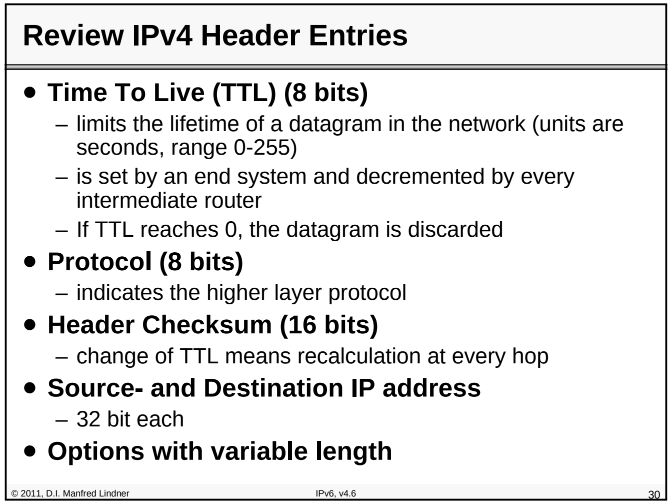
Review IPv4 Header Entries

- **Version (4 bits)**
  - version of the IP protocol = 4
- **HLEN (4 bits)**
  - length of the header in 32 bit words
- **Type of Service (ToS) (8 bits)**
  - priority of a datagram (precedence bits)
  - preferred network characteristics (D, T, R and C bits)
  - Nowadays replaced by DSCP (Differentiated Services Code Point) to indicate a service class in Diff-Serv-QoS networks
- **Total Length (16 bits)**
  - total length of the IP datagram (header + data) in octets



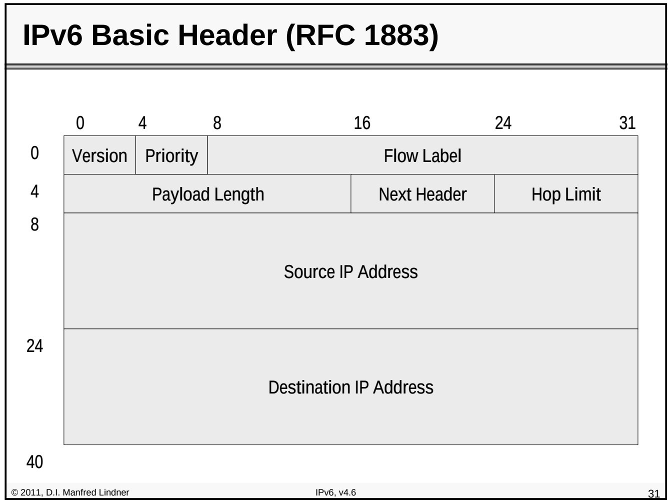
Review IPv4 Header Entries

- **Identification (16 bits)**
  - – unique identification of a datagram, used for fragmentation and reassembling
- **Flags (for fragmentation) (3 bits)**
  - Reserved
  - DF (do not fragment)
  - MF (more fragments)
- **Fragment Offset (13 bits)**
  - – position of a fragment relative to the beginning of the original datagram, Offset is measured in multiples of 8 octets (64 bits)



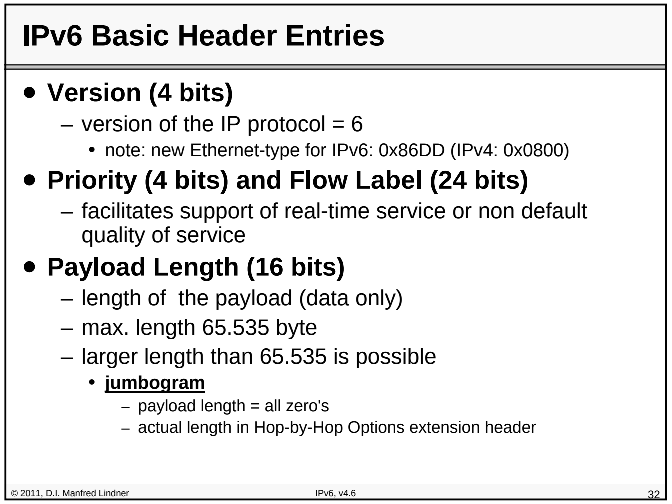
Review IPv4 Header Entries

- **Time To Live (TTL) (8 bits)**
  - – limits the lifetime of a datagram in the network (units are seconds, range 0-255)
  - – is set by an end system and decremented by every intermediate router
  - – If TTL reaches 0, the datagram is discarded
- **Protocol (8 bits)**
  - – indicates the higher layer protocol
- **Header Checksum (16 bits)**
  - – change of TTL means recalculation at every hop
- **Source- and Destination IP address**
  - – 32 bit each
- **Options with variable length**



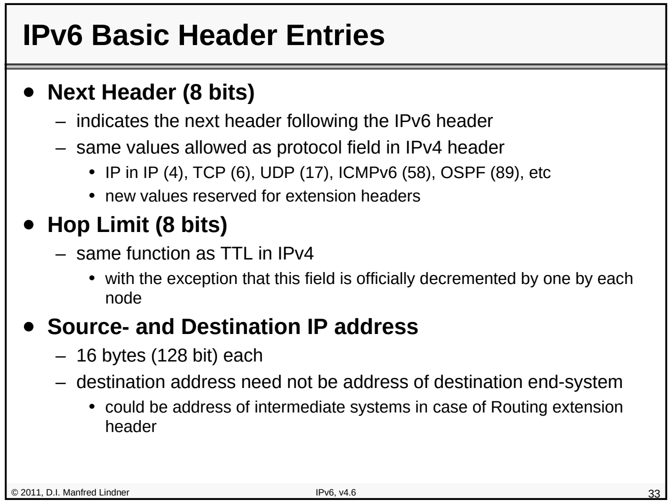
IPv6 Basic Header (RFC 1883)

	0	4	8	16	24	31
0	Version	Priority	Flow Label			
4	Payload Length			Next Header	Hop Limit	
8	Source IP Address					
24	Destination IP Address					
40						



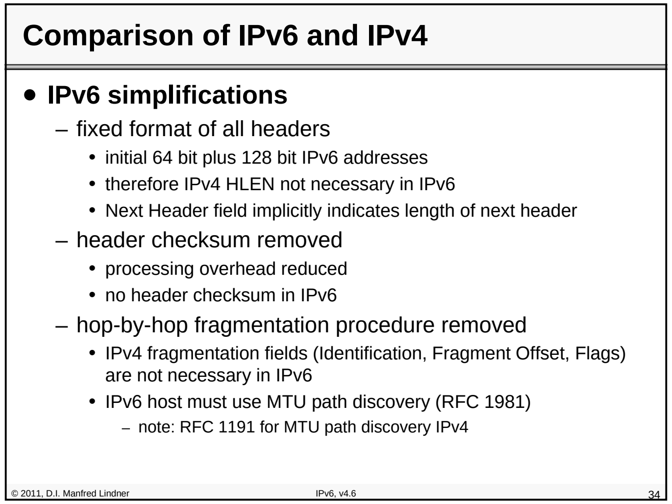
IPv6 Basic Header Entries

- **Version (4 bits)**
  - – version of the IP protocol = 6
    - note: new Ethernet-type for IPv6: 0x86DD (IPv4: 0x0800)
- **Priority (4 bits) and Flow Label (24 bits)**
  - – facilitates support of real-time service or non default quality of service
- **Payload Length (16 bits)**
  - – length of the payload (data only)
  - – max. length 65.535 byte
  - – larger length than 65.535 is possible
    - **<u>jumbogram</u>**
      - – payload length = all zero's
      - – actual length in Hop-by-Hop Options extension header



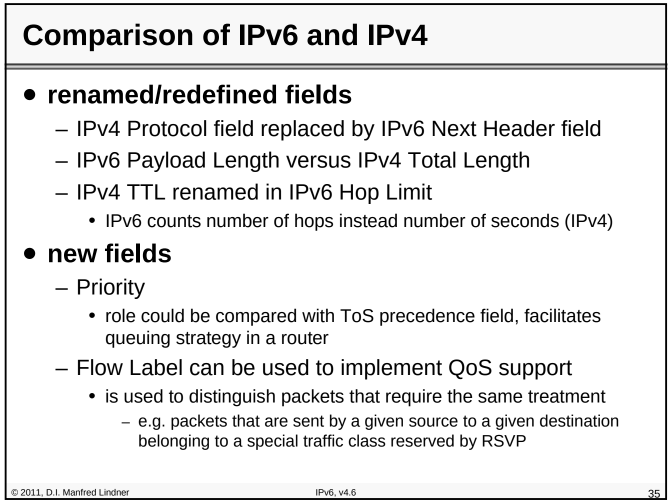
IPv6 Basic Header Entries

- **Next Header (8 bits)**
  - indicates the next header following the IPv6 header
  - same values allowed as protocol field in IPv4 header
    - IP in IP (4), TCP (6), UDP (17), ICMPv6 (58), OSPF (89), etc
    - new values reserved for extension headers
- **Hop Limit (8 bits)**
  - same function as TTL in IPv4
    - with the exception that this field is officially decremented by one by each node
- **Source- and Destination IP address**
  - 16 bytes (128 bit) each
  - destination address need not be address of destination end-system
    - could be address of intermediate systems in case of Routing extension header



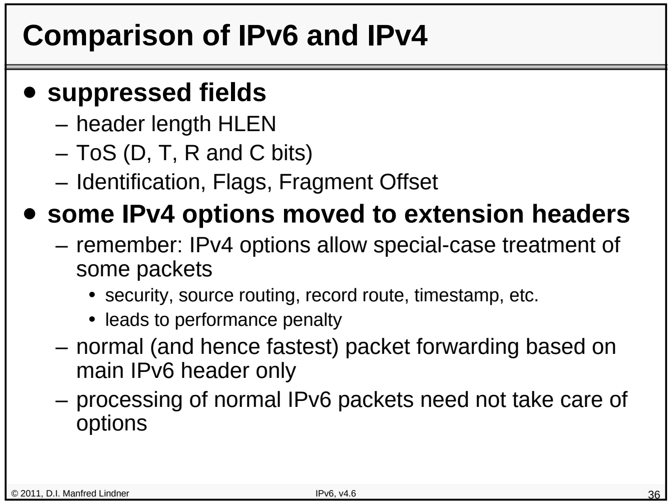
Comparison of IPv6 and IPv4

- **IPv6 simplifications**
  - fixed format of all headers
    - initial 64 bit plus 128 bit IPv6 addresses
    - therefore IPv4 HLEN not necessary in IPv6
    - Next Header field implicitly indicates length of next header
  - header checksum removed
    - processing overhead reduced
    - no header checksum in IPv6
  - hop-by-hop fragmentation procedure removed
    - IPv4 fragmentation fields (Identification, Fragment Offset, Flags) are not necessary in IPv6
    - IPv6 host must use MTU path discovery (RFC 1981)
      - note: RFC 1191 for MTU path discovery IPv4



Comparison of IPv6 and IPv4

- **renamed/redefined fields**
  - IPv4 Protocol field replaced by IPv6 Next Header field
  - IPv6 Payload Length versus IPv4 Total Length
  - IPv4 TTL renamed in IPv6 Hop Limit
    - IPv6 counts number of hops instead number of seconds (IPv4)
- **new fields**
  - Priority
    - role could be compared with ToS precedence field, facilitates queuing strategy in a router
  - Flow Label can be used to implement QoS support
    - is used to distinguish packets that require the same treatment
      - e.g. packets that are sent by a given source to a given destination belonging to a special traffic class reserved by RSVP



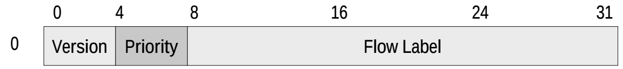
Comparison of IPv6 and IPv4

- **suppressed fields**
  - header length HLEN
  - ToS (D, T, R and C bits)
  - Identification, Flags, Fragment Offset
- **some IPv4 options moved to extension headers**
  - remember: IPv4 options allow special-case treatment of some packets
    - security, source routing, record route, timestamp, etc.
    - leads to performance penalty
  - normal (and hence fastest) packet forwarding based on main IPv6 header only
  - processing of normal IPv6 packets need not take care of options

Adaptation of other Protocols/Sockets

- **protocols to be adapted**
  - ICMP, IGMP, ARP -> ICMPv6
  - UDP / TCP -> UDPv6 / TCPv6
  - RIP, OSPF -> RIPv6 (RIPv2), OSPFv6 (OSPFv3)
  - DNS (-> new AAAA resource record)
  - DHCP -> DHCPv6
- **standard programming interfaces to be adapted**
  - address data structure
  - name-to-address translation functions
  - address conversion functions
- **adaptation to IPv6 is straightforward**
  - address aspects must be handled mainly

Priority (RFC 1883 Old)

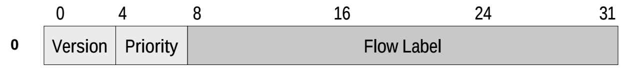


Values 0 - 7 are used to specify the priority of traffic for which the source is providing congestion control, e.g. traffic that "backs off" in response to congestion such as TCP traffic.

0 - uncharacterized traffic
1 - "filler" traffic (e.g., netnews)
2 - unattended data transfer (e.g., email)
3 - (reserved)
4 - attended bulk transfer (e.g., FTP, NFS)
5 - (reserved)
6 - interactive traffic (e.g., telnet, X, database access)
7 - internet control traffic (e.g., routing protocols, SNMP)

Values 8 - 15 are used to specify the priority of traffic that does not back off in response to congestion, e.g. "real-time" packets being sent at a constant rate.

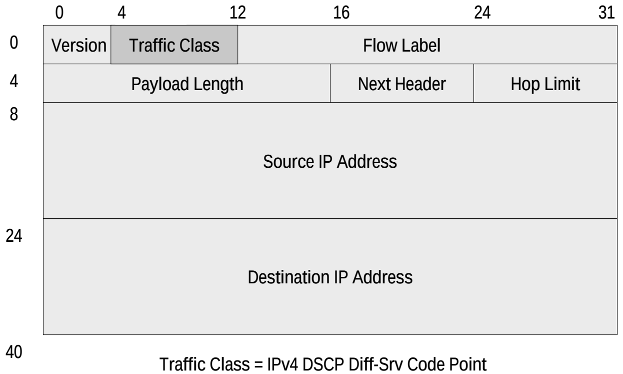
Flow Label (RFC 1883 Old)



The 24-bit Flow Label field in the IPv6 header may be used by a source to label those packets for which it requests special handling by the IPv6 routers, such as non-default quality of service or "real-time" service. Hosts or routers that do not support the functions of the Flow Label field are required to set the field to zero when originating a packet, pass the field on unchanged when forwarding a Packet and ignore the field when receiving a packet

A flow is a sequence of packets sent from a particular source to a particular (unicast or multicast) destination for which the source desires special handling by the intervening routers. The nature of that special handling might be conveyed to the routers by a control protocol, such as a resource reservation protocol, or by information within the flow's packets themselves, e.g., in a hop-by-hop option.

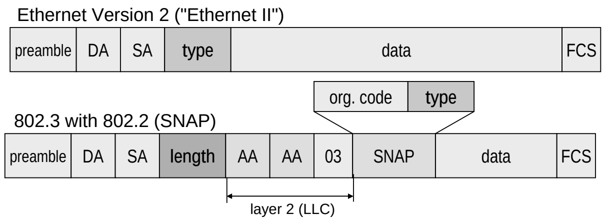
IPv6 Header (RFC 2460 Actual)



Traffic Class = IPv4 DSCP Diff-Srv Code Point

IPv6 over Ethernet

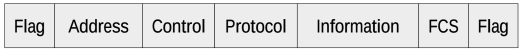
- **Type = 0x86DD**



- **Multicast mapping L3->L2:**
  - Prefix 3333
  - Resulting MAC=3333<last 32 bits of IPv6 address>

IPv6 over PPP

- **HDLC framing and encapsulation (RFC 1662)**
- **Protocol = 0x0057**

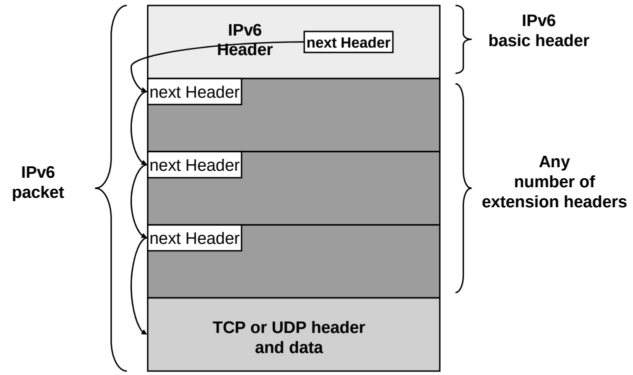


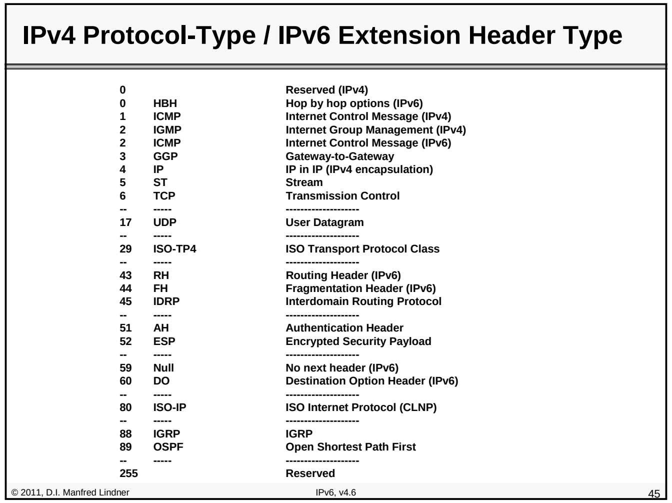
Flag	=	01111110	Protocol	=	see RFC 1700 (assigned numbers)
Address	=	11111111	Information	=	Network Layer PDU
Control	=	00000011 (UI frame)	FCS	=	16 bit

Agenda

- **Introduction**
- **IPv6**
  - IPv6 Main Header, Comparison with IPv4 Header
  - Extension Headers
  - Security
- **Addressing and Routing**
- **Plug and Play**
- **Transition**

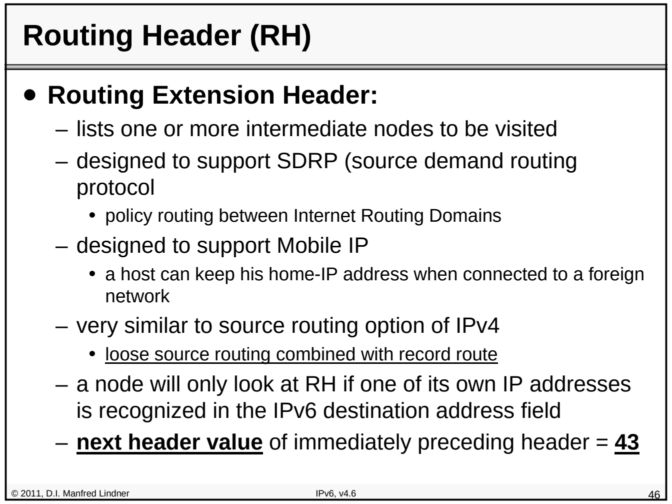
Daisy Chain of Extension Headers





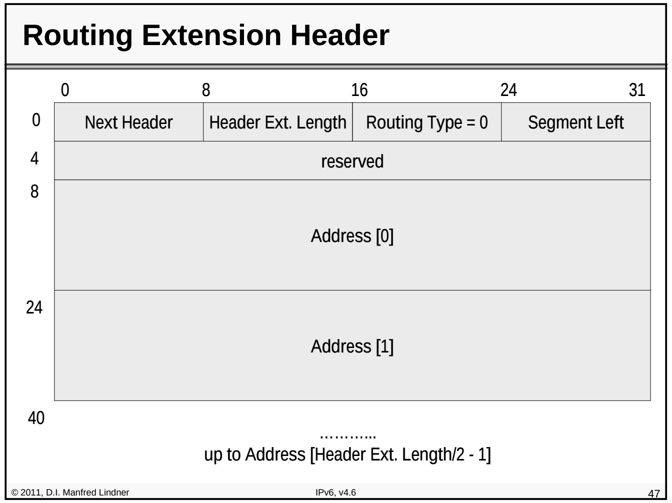
IPv4 Protocol-Type / IPv6 Extension Header Type

0		Reserved (IPv4)
0	HBH	Hop by hop options (IPv6)
1	ICMP	Internet Control Message (IPv4)
2	IGMP	Internet Group Management (IPv4)
2	ICMP	Internet Control Message (IPv6)
3	GGP	Gateway-to-Gateway
4	IP	IP in IP (IPv4 encapsulation)
5	ST	Stream
6	TCP	Transmission Control
--	-----	--------------------
17	UDP	User Datagram
--	-----	--------------------
29	ISO-TP4	ISO Transport Protocol Class
--	-----	--------------------
43	RH	Routing Header (IPv6)
44	FH	Fragmentation Header (IPv6)
45	IDRP	Interdomain Routing Protocol
--	-----	--------------------
51	AH	Authentication Header
52	ESP	Encrypted Security Payload
--	-----	--------------------
59	Null	No next header (IPv6)
60	DO	Destination Option Header (IPv6)
--	-----	--------------------
80	ISO-IP	ISO Internet Protocol (CLNP)
--	-----	--------------------
88	IGRP	IGRP
89	OSPF	Open Shortest Path First
--	-----	--------------------
255		Reserved



Routing Header (RH)

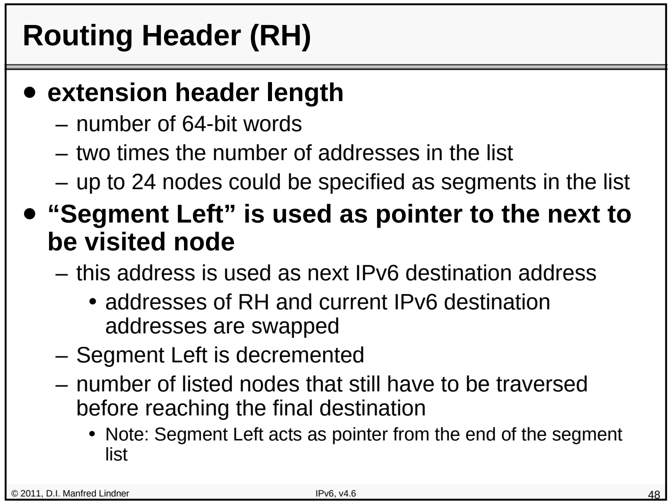
- **Routing Extension Header:**
  - – lists one or more intermediate nodes to be visited
  - – designed to support SDRP (source demand routing protocol
    - policy routing between Internet Routing Domains
  - – designed to support Mobile IP
    - a host can keep his home-IP address when connected to a foreign network
  - – very similar to source routing option of IPv4
    - loose source routing combined with record route
  - – a node will only look at RH if one of its own IP addresses is recognized in the IPv6 destination address field
  - – **next header value** of immediately preceding header = **43**



Routing Extension Header

	0	8	16	24	31
0	Next Header	Header Ext. Length	Routing Type = 0	Segment Left	
4	reserved				
8	Address [0]				
24	Address [1]				
40	………..				

up to Address [Header Ext. Length/2 - 1]

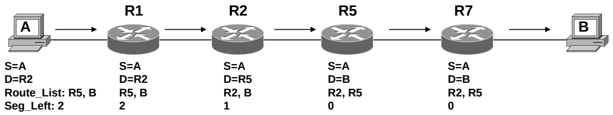


Routing Header (RH)

- **extension header length**
  - – number of 64-bit words
  - – two times the number of addresses in the list
  - – up to 24 nodes could be specified as segments in the list
- **“Segment Left” is used as pointer to the next to be visited node**
  - – this address is used as next IPv6 destination address
    - addresses of RH and current IPv6 destination addresses are swapped
  - – Segment Left is decremented
  - – number of listed nodes that still have to be traversed before reaching the final destination
    - Note: Segment Left acts as pointer from the end of the segment list

The Routing Header

- **Contains “segments” (= next hops) and “counter” (= segments left)**
- **Next-hop list is decremented**
- **New DA = next hop (“segment”)**
- **DA seen by receiving hop stored in segment list**



Destination Options (DO)

- **Destination Options Extension Header:**
  - two ways to encode optional destination information in IPv6
    - destination option header
    - separate extension header
  - specifies one or more options which are processed only by the end-system specified in IP destination address field
  - used for adding functionality to IPv6 later
    - E.g. Mobile IP together with routing extension header
  - option may be known or not know to receiver
    - if unknown ⇨ action-bits specify what to do
  - extension header length
    - number of 64-bit words
  - **next header value** of immediately preceding header = **60**
  - currently defined destination options for padding only
    - to align gap between options properly (32 bit alignment)
      - PAD1 (type = 0) … null byte to be included
      - PADN (type = 1) … specifies number of null bytes to be included

Destination Options Extension Header

	0	8	16	24 31
0	Next Header	Header Ext. Length	Option Type	Option Data Length
4	Option Data			
8	Option Type	Option Data Length	Option Data	

Option Type:

0 1	2	3 7
Action Bits	C	Number

Action Bits: 0 0 ... skip over this option
0 1 ... discard the packet, no ICMP report
1 0 ... discard the packet, send ICMP report even if multicast
1 1 ... discard the packet, send ICMP report if not multicast

C Bit (change en route): 0 ... option does not change en-route
1 ... option may change en-route

Hop-by-Hop Options (HBH)

- **Hop-by-Hop Extension Header:**
  - same format as the DO header
  - specifies one or more options which are processed by every intermediate system (router)
  - may be used for adding management/debugging functions later to IPv6
  - option may be known or not know to receiver
    - if unknown ⇨ action-bits specify what to do
  - here C-bits are used by every hop
  - **next header value** of immediately preceding header = **0**

Hop-by-Hop Options (HBH) (cont.)

- currently defined hop-by-hop option:
  - transport of jumbogram (RFC 2675)
    - Jumbo … packet longer than 65.535
    - option type = 194 (**110** 00010)
    - length of Jumbo specified in option data field
  - Router Alert option (RFC 2711)
    - information in that datagram needs to be examined by routers along the path to the destination although not addressed to any of the routers addresses
      - » Alternative would be to examine every datagram in detail
    - option-type = 5 (**000** 00101)
    - router should examine those packets more closely
    - protocol like RSVP can benefit from this option
    - no additional performance penalty on forwarding normal datagrams

HBH Example for Jumbogram

	0	8	16	24 … 31
0	Next Header	Header Ext. Length	194 (**110** 00010)	4
4	Jumbo Payload Length			

Option Type:

0 1	2	3 … 7
Action Bits	C	Number

Action Bits: 0 0 ... skip over this option
0 1 ... discard the packet, no ICMP report
1 0 ... discard the packet, send ICMP report even if multicast packet
<u>1 1 ... discard the packet, send ICMP report if not multicast packet</u>

C Bit: <u>0 ... option does not change en-route</u>
1 ... option may change en-route

Fragment Header (FH)

- **Fragment Extension Header:**
  - IPv6 routers do not fragment oversized packets
    - like IPv4 with “Do not Fragment” - Bit = 1
  - end-system must use MTU path discovery in order to select correct MTU size
  - but end-system (source node) can fragment packets before they are sent in the network
    - using FH
      - Identification field, Fragment Offset field, More fragment-bit used in the same way as IPv4
  - each fragment is routed independently
  - destinations end-system must reassemble fragments
  - **<u>next header value</u>** of immediately preceding header = **<u>44</u>**

Fragment Extension Header (FH)

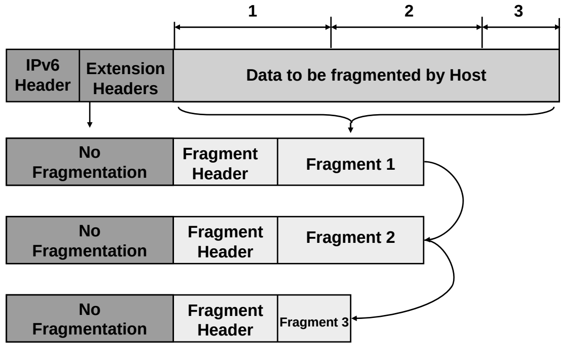
	0	8	16	24 … 29	31
0	Next Header	Reserved	Fragment Offset	Res.	M
4	Identification				

Fragment Offset (13 bits) ... pointer to location of fragment in original packet

Identification (32 bits) ... unique ID for the original packet, same value in every fragment of original packet

M Bit: 0 ... last fragment
1 ... more fragments

Host Fragmentation



No Fragmentation – contain the IPv6 basis header + optional extension headers (HPH, DO, RH))

Extension Header Order

- **IPv6 Header**
- **Hop-by-Hop Options Header**
- **Destination Options Header (1)**
- **Routing Header**
- **Fragment Header**
- **Authentication Header**
  - defined in RFC 2402
- **Destination Options Header (2)**
- **Upper Layer Header (e.g. TCP or UDP)**

  or
- **Encapsulating Security Payload Header**
  - defined in RFC 2406

Agenda

- **Introduction**
- **IPv6**
  - IPv6 Main Header, Comparison with IPv4 Header
  - Extension Headers
  - Security
- **Addressing and Routing**
- **Plug and Play**
- **Transition**

IP Security Discussion Raise with IPv6

- **End-to-end security**
  - will become more and more important when Internet goes to the commercial world
- **Question was**
  - if the next generation IP protocol (IPv6) should provide end-to-end security as integral part of itself
- **Basic building blocks for end-to-end security**
  - authentication and integrity
    - provides identity of sender
    - senders message was not changed on the way through the network
  - non-repudiation
    - the sender cannot later repudiate the contents of the message
    - protection of the receiver
  - confidentiality or privacy
    - message cannot be read by others than authorized receiver

IPv6 Security Aspects

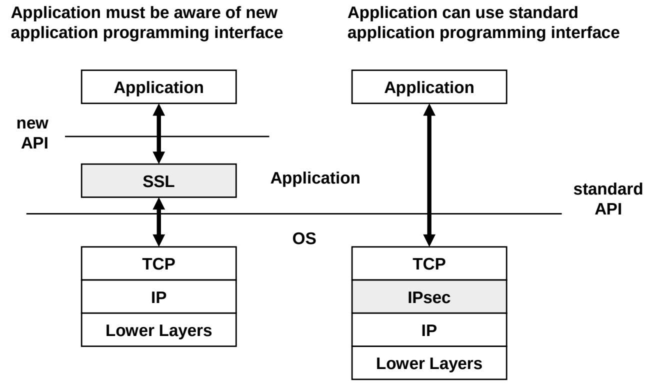
- **After heated discussions IESG decided**
  - basic building blocks (without non-repudiation) of network security should be part of IPv6 functionality
  - a vendor of an IPv6 implementation must include support of these basic building blocks in order to be standard-compliant
    - does not mean that the use of authentication and encryption blocks is required; only support must be guaranteed
  - IPv6 security follows the general IPsec recommendations
    - RCF 2401 (Former RFC 1825) Security Architecture for IP (IPv4 and IPv6)
  - difference of security aspects between IPv4 and IPv6
    - security in IPv6 is an integral part of it
    - security in IPv4 is an add on

Security Architecture for IP

- **The goal of the IPsec architecture**
  - provision of various security services for traffic at the IP layer in both IPv4 and IPv6 environments
  - in a standardized and universal way
  - „Security Framework“

- **Before IPsec**
  - existing solutions were mostly on the application layer (SSL, S/MIME, ssh, …)
  - existing solutions on the network layer were all propriety
    - e.g. it was complicated, time demanding and expensive to establish multi-application or multi-vendor virtual private networks (VPNs)

Which Layer for Security?



Application must be aware of new application programming interface

Application can use standard application programming interface

Application

new API

SSL

Application

OS

TCP

IP

Lower Layers

Application

standard API

TCP

IPsec

IP

Lower Layers

Elements of IPsec 1

- **Security Associations (SA)**
  - what they are and how they work, how they are managed and their associated processing
    - defined in RFC 2401
  - Security Policy Database (SPD), Security Association Database (SAD)
- **Security Protocols (for traffic security)**
  - Authentication Header (AH)
    - defined in RFC 2402
  - Encapsulating Security Payload (ESP)
    - defined in RFC 2406
  - in this area secret-key algorithms are used because of performance reasons (HMAC, DES, 3DES, …)

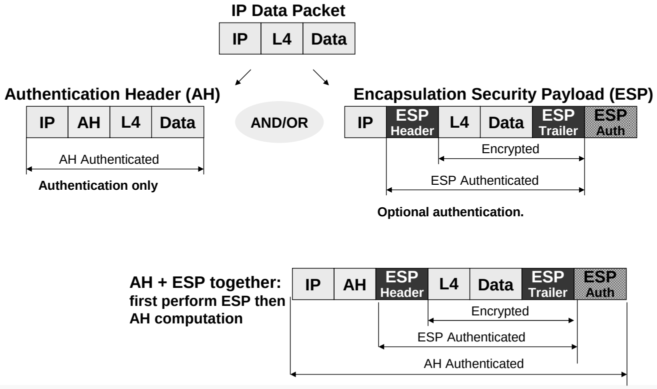
Elements of IPsec 2

- **Management of Security Associations and Keys**
  - manual for static and small environments
  - automatic for scalable environments by ISAKMP
  - ISAKMP (Internet Security Association and Key Management Protocol)
    - defined in RFC 2408
  - Internet Key Exchange (IKE) for ISAKMP
    - defined in RFC 2409
  - Domain of Interpretation (DOI)
    - defined in RFC 2407
- **Algorithms for authentication and encryption**
  - defined in many separate RFCs

What IPsec does?

- **IPsec enables a system**
  - to select required security protocols, determine the algorithm(s) to use for the service(s), and put in place any cryptographic keys required to provide the requested services
- **IPsec can be used**
  - to protect one or more "paths" between a pair of hosts, between a pair of security gateways, or between a security gateway and a host
  - security gateway could be for example, a router or a firewall implementing IPsec
    - VPN concentrator is another name for such a device if several SA pairs are terminated at the same point

IPsec Headers

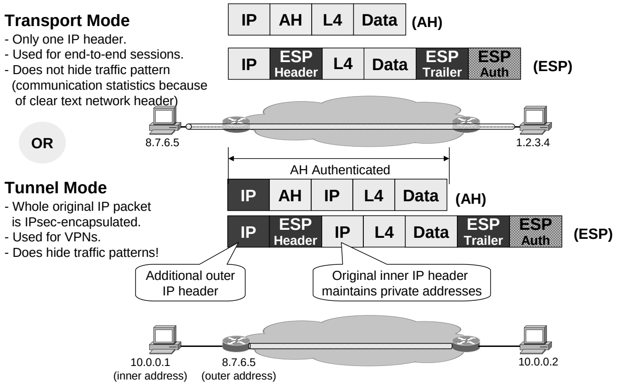


Authentication only

Optional authentication.

AH + ESP together: first perform ESP then AH computation

IPsec Modes



Transport Mode
- Only one IP header.
- Used for end-to-end sessions.
- Does not hide traffic pattern (communication statistics because of clear text network header)

Tunnel Mode
- Whole original IP packet is IPsec-encapsulated.
- Used for VPNs.
- Does hide traffic patterns!

Additional outer IP header

Original inner IP header maintains private addresses

AH Security Service (RFC 2402)

- **AH provides**
  - IP datagram sender authentication by HMAC or MAC
  - IP datagram integrity assurance by HMAC or MAC
  - replay detection and protection via sequence number (optional)
- **AH does not provide**
  - non-repudiation because of usage of secret-keys (shared keys) for HMAC or MAC
    - note: Digital Signature needs usage of public-key technique by signing a message with the private-key
  - confidentiality (encryption)
  - authentication for IP fragments
    - therefore IP fragments must be assembled before authentication is checked (better avoid it by MTU path discovery)

IPv4 and AH

0	4	8	16		31
Vers.=4	HLEN	ToS or DSCP	Total Length		
Fragment Identifier			Flags	Fragment Offset	
TTL		protocol = 51	Header Checksum		
Source Address					
Destination Address					
IP Options					Pad
First 32 bits of AH					
....					
Last 32 bits of AH					
Payload					
....					

Authentication Header (AH)

	0	8	16	24	31
0	Next Header	Length	Reserved		
4	Security Parameters Index (SPI)				
8	Sequence Number				
10	Authentication Data (variable number of 32-bit words)				

note: AH was originally defined as extension header for IPv6 and later same structure was also used for IPv4

IPv6: next header value of immediately preceding header = 51

Authentication Header (AH)

- **Next Header (8 bits)**
  - indicates the next header following the AH header
  - same values allowed as protocol field in IPv4 header
    - IP in IP (4), TCP (6), UDP (17), ICMP (1), OSPF (89), etc
    - next header value of immediately preceding header = 51 (AH)
- **Length**
  - length of AH header
    - number of 32-bit words
- **Security Parameter Index**
  - a 32-bit number identifying (together with IP destination address) the security association for this IP datagram
  - SPI value 0 is reserved for local implementation specific use and must not be sent on the wire

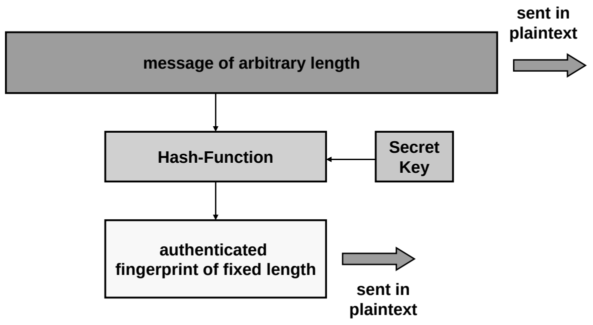
Authentication Header (AH)

- **Sequence number:**
  - monotonically increasing counter value (mandatory and always present)
  - defined in RFC 2085
  - prevention against replay attacks enabled by default
  - mandatory for transmitter but the receiver need not act upon it
  - every new SA resets this number to zero (thus first packet = 1), no cycling: after sending the $2^{\mathbf{32nd}}$ packet, a new SA must be established.

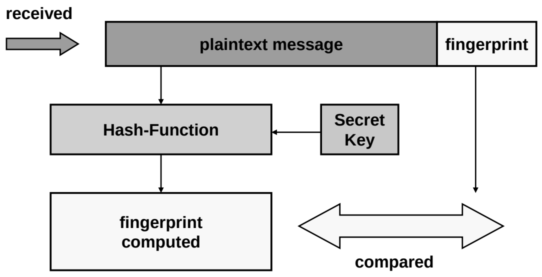
Authentication Header (AH)

- **Authentication Data:**
  - contains <u>Integrity Check Value</u> (<u>ICV</u>)
    - all fields behind AH header plus predictable field of IP header before AH (e.g. TTL, checksum, ToS/DSCP regarded to be zero for ICV calculation)
  - the algorithm for authentication is free and must be negotiated
  - mandatory default calculation of the authentication data must be supported
    - HMAC with keyed-MD5 (RFC 2403), 128 bit secret-key
    - HMAC with keyed-SHA-1 (RFC 2404), 160 bit secret-key
  - alternative
    - DES-CBC based MAC
  - non-repudiation (IP datagram signing) is not supported!

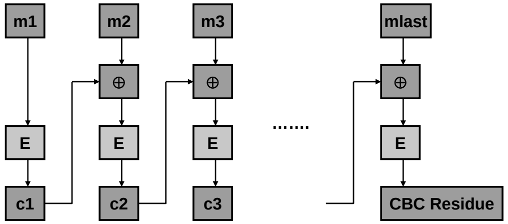
Refresher: HMAC in Action (Sender) 1


sent in plaintext
message of arbitrary length
Hash-Function
Secret Key
authenticated fingerprint of fixed length
sent in plaintext

Refresher: HMAC in Action (Receiver) 2

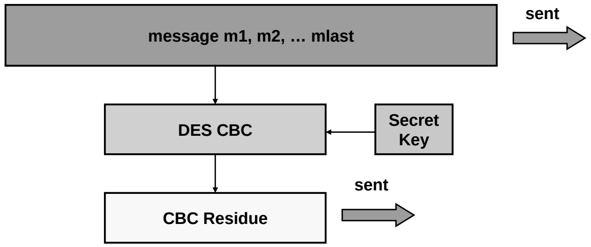


Refresher: MAC CBC Residue 1

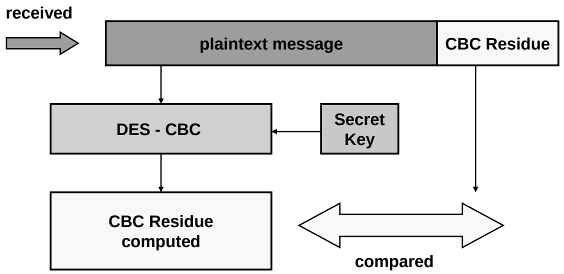


Encryption at sender to create CBC residue
Same done at the receiver to check

Refresher: MAC CBC Residue 2



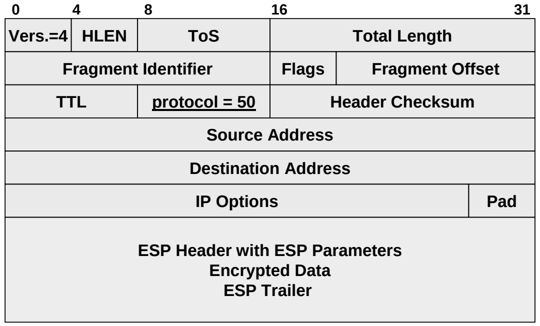
Refresher: MAC CBC Residue 3



ESP Security Service (RFC 2406)

- **ESP provides**
  - confidentiality (encryption of payload with secret-key algorithm)
  - replay detection and protection via sequence number (optional)
  - IP datagram sender authentication by HMAC (optional)
  - IP datagram integrity assurance by HMAC (optional)
- **ESP does not provide**
  - key distribution
  - encryption of IP fragments
    - therefore IP fragments must be assembled before decryption

IPv4 and ESP



0 4 8 16 31

Vers.=4 | HLEN | ToS | Total Length

Fragment Identifier | Flags | Fragment Offset

TTL | protocol = 50 | Header Checksum

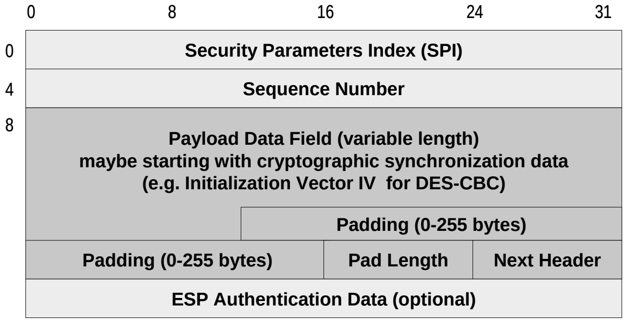
Source Address

Destination Address

IP Options | Pad

ESP Header with ESP Parameters
Encrypted Data
ESP Trailer

Encapsulating Security Payload (ESP)



0 8 16 24 31

0 Security Parameters Index (SPI)

4 Sequence Number

8 Payload Data Field (variable length)
maybe starting with cryptographic synchronization data
(e.g. Initialization Vector IV for DES-CBC)

Padding (0-255 bytes)

Padding (0-255 bytes) | Pad Length | Next Header

ESP Authentication Data (optional)

note: ESP was originally defined as extension header for IPv6
and later same structure was used also for IPv4
IPv6: next header value of immediately preceding header = 50

ESP Header, Payload

- **SPI and Sequence Number**
  - used for same functions as in the AH header
    - defining SA and prevention of replay attack
  - this are the only fields of ESP transmitted in cleartext
- **Payload Field of ESP is encrypted**
  - actual format depends on encryption method
    - e.g location of Initialization Vector (IV) for DES-CBC
    - note: every IP datagram must contain an IV because IP datagram's may arrive out of sequence

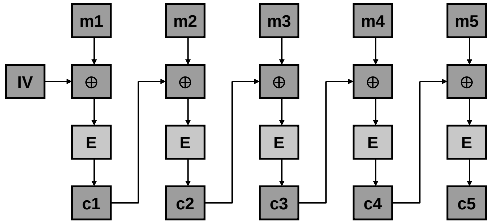
ESP Trailer

- **Padding Field**
  - is used to fill the plaintext to the size required by the encryption algorithm (e.g. the block size of a block cipher)
  - is used to align 4 byte boundaries
- **Pad Length**
  - pointer to end of data
- **Next Header**
  - identifies the type of data contained in the Payload Data Field, e.g., an extension header in IPv6 or an upper layer protocol identifier
    - same values allowed as protocol field in IPv4 header

ESP Encryption Methods

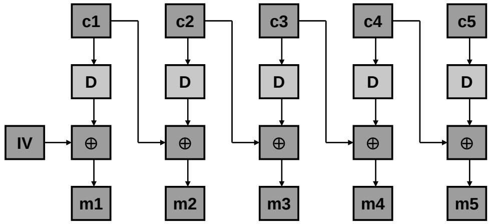
- **Mandatory default transformation of the data**
  - – DES-CBC (Data Encryption Standard - Cipher Block Chaining)
    - parameter field contains Initialization Vector (IV) field
- **Triple-DES, Blowfish, IDEA, RC5 as alternative**
  - – see RFC 2451
- **An ESP “Null” algorithm must be supported**
  - – see RFC 2401
  - – see RFC 2410 where it is praised for ease of implementation, great speed and simplicity ;-)
- **Optional authentication**
  - – HMAC with keyed-MD5 or HMAC with keyed-SHA-1

Refresher DES Mode - CBC Encryption 1



Encryption with CBC

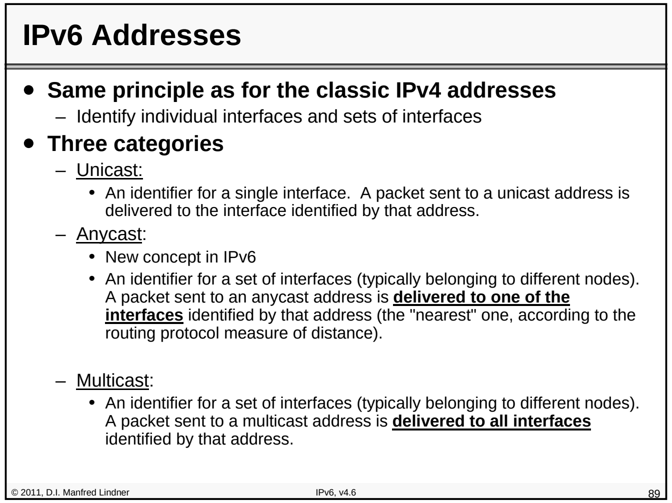
Refresher DES Mode - CBC Decryption 2



Decryption with CBC

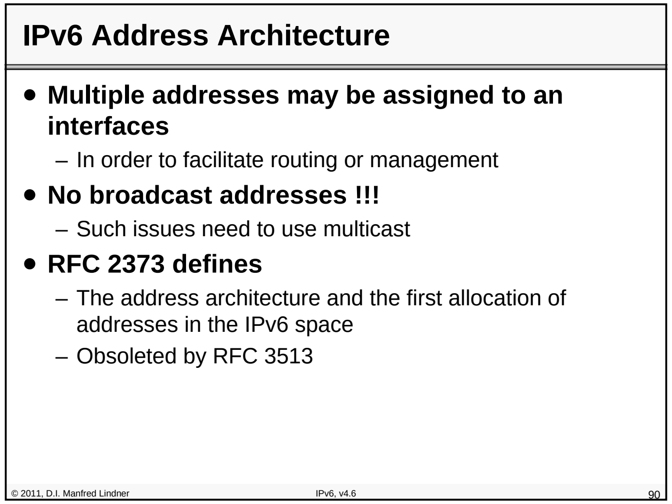
Agenda

- **Introduction**
- **IPv6**
  - – IPv6 Main Header, Comparison with IPv4 Header
  - – Extension Headers
  - – Security
- **Addressing and Routing**
- **Plug and Play**
- **Transition**



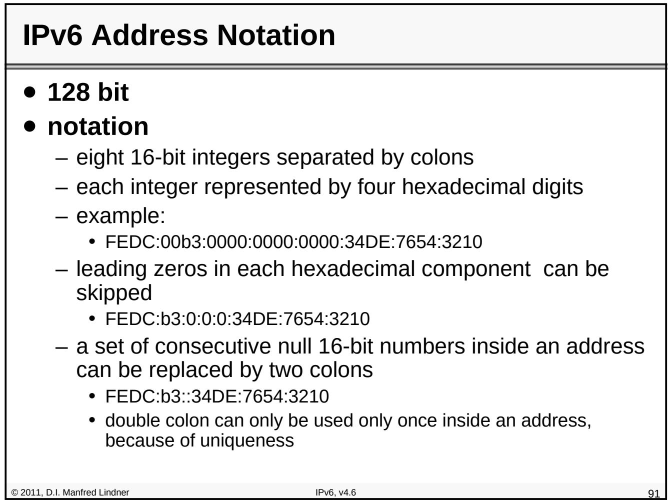
IPv6 Addresses

- **Same principle as for the classic IPv4 addresses**
  - Identify individual interfaces and sets of interfaces
- **Three categories**
  - Unicast:
    - An identifier for a single interface. A packet sent to a unicast address is delivered to the interface identified by that address.
  - Anycast:
    - New concept in IPv6
    - An identifier for a set of interfaces (typically belonging to different nodes). A packet sent to an anycast address is **delivered to one of the interfaces** identified by that address (the "nearest" one, according to the routing protocol measure of distance).
  - Multicast:
    - An identifier for a set of interfaces (typically belonging to different nodes). A packet sent to a multicast address is **delivered to all interfaces** identified by that address.



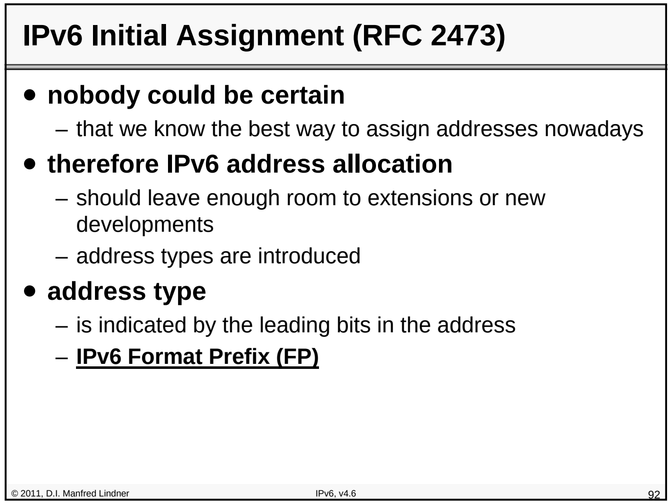
IPv6 Address Architecture

- **Multiple addresses may be assigned to an interfaces**
  - In order to facilitate routing or management
- **No broadcast addresses !!!**
  - Such issues need to use multicast
- **RFC 2373 defines**
  - The address architecture and the first allocation of addresses in the IPv6 space
  - Obsoleted by RFC 3513



IPv6 Address Notation

- **128 bit**
- **notation**
  - eight 16-bit integers separated by colons
  - each integer represented by four hexadecimal digits
  - example:
    - FEDC:00b3:0000:0000:0000:34DE:7654:3210
  - leading zeros in each hexadecimal component can be skipped
    - FEDC:b3:0:0:0:34DE:7654:3210
  - a set of consecutive null 16-bit numbers inside an address can be replaced by two colons
    - FEDC:b3::34DE:7654:3210
    - double colon can only be used only once inside an address, because of uniqueness



IPv6 Initial Assignment (RFC 2473)

- **nobody could be certain**
  - that we know the best way to assign addresses nowadays
- **therefore IPv6 address allocation**
  - should leave enough room to extensions or new developments
  - address types are introduced
- **address type**
  - is indicated by the leading bits in the address
  - **IPv6 Format Prefix (FP)**

Initial IPv6 Prefix Allocation (RFC 2473)

Allocation	Format Prefix (binary)	Fraction of Address Space
Reserved	0000 0000	1/256
Unassigned	0000 0001	1/256
Reserved for NSAP allocation	0000 001	1/128
Reserved for IPX allocation	0000 010	1/128
Unassigned	0000 011	1/128
Unassigned	0000 1	1/32
Unassigned	0001	1/16
Aggregatable global unicast address	001	1/8
Unassigned	010	1/8
Unassigned	011	1/8
Unassigned	100	1/8

Initial IPv6 Prefix Allocation (RFC 2473) cont.

Allocation	Format Prefix (binary)	Fraction of Address Space
Unassigned	101	1/8
Unassigned	110	1/8
Unassigned	1110	1/16
Unassigned	1111 0	1/32
Unassigned	1111 10	1/64
Unassigned	1111 110	1/128
Unassigned	1111 1110 0	1/512
Link local-use addresses	1111 1110 10	1/1024
Site local-use addresses	1111 1110 11	1/1024
multicast addresses	1111 1111	1/256

IPv6 Initial Addressing Plan Ideas 1

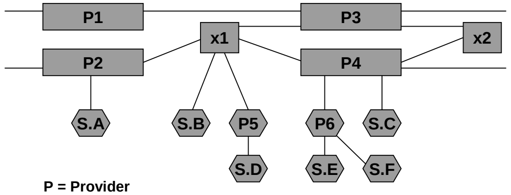
- **current growth of Internet means**
  - explosion of the routing tables
- **addressing should be done in a way**
  - to keep number of routing table entries of Internet core routers small
- **route aggregation is necessary**
  - prefix, length routing
  - lessons learnt by CIDR
    - curbs the growth of the routing tables
- **the way to achieve this:**
  - **aggregatable global unicast addresses**

IPv6 Initial Addressing Plan Ideas 2

- **aggregatable global unicast addresses**
  - based on provider based addressing (RFC 2073, RFC 2374, RFC 3587 – but provider opposed it)
    - RFC 2073, 2374 were on the standard track
    - RFC 3587 -> category: informational
  - addresses are allocated from your provider
  - if customers want to change the provider
    - your prefix changes
    - but renumbering of hosts, routers and sites has been included in the IPv6 protocol
  - the burden to handle this is on the customer
    - **need for efficient auto-configuration** of addresses

IPv6 Initial Addressing Plan Ideas 3

- aggregatable global unicast addresses support both:
  - provider based aggregation
  - exchange based aggregation
- possibility to create hierarchies
- efficient routing aggregation



P = Provider
S = Subscriber
X = Exchange

Aggregatable Global Unicast Address (RFC 2374)

(FP) 3	13 bits	8 bits	24 bits	16 bits	64 bits
0 0 1	TLA ID	Res	NLA ID	SLA ID	Interface ID
Public Topology				Site Topology	Interface Identifier

Aggregatable Global Unicast Address
can be routed on the global Internet,
their uniqueness is guaranteed globally

RFC 2374

Format Prefix ...	3 bits
Top Level Aggregator ID	13 bits
Reserved	8 bits
Next Level Aggregator ID	24 bits
Site Level Aggregator ID	16 bits
Interface-ID ...	64 bits (EUI-64 – usually derived from MAC address)

Aggregatable Global Unicast Address (RFC 2374)

- **Top Level Aggregator (TLA)**
  - public access points that interconnect service providers/telephone companies
  - IANA allocates these addresses
- **Next Level Aggregator (NLA)**
  - large Internet service providers
  - NLA's assign to the next level
- **Site Level Aggregator (SLA)**
  - called a subscriber; can be an organization, a company, a university, small ISP
  - they assign addresses to their users
  - SLA provide a block of contiguous addresses
- **Interface ID**
  - host interface
  - IEEE has defined a 64 bit NIC address known as EUI-64
  - NIC driver for IPv6 will convert 48 bit NIC to 64 bit NIC

IP Addressing Architecture (RFC 3515)

- **Addressing structure of RFC 2373 changed**
  - to simplify and clarify how different address types are identified
  - this was done to insure that implementations do not build in any knowledge about global unicast format prefixes
  - changes include:
    - removed Format Prefix (FP) terminology
    - revised list of address types to only include exceptions to global unicast and a single entry that identifies everything else as global unicast

IPv6 Address Types (RFC 3515)

Address Type	Binary Prefix	IPv6 Notation
Unspecified	0……………..0	::/128
Loopback	0……………..1	::1/128
Link local-unicast	1111 1110 10	FE80::/10
Site local-unicast	1111 1110 11	FEC0::/10
Multicast	1111 1111	FF00::/8
Global unicast	everything else	

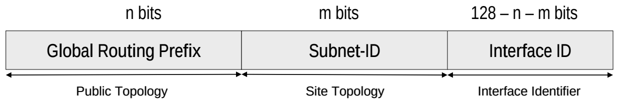
Initial Assignment of Addresses (RFC 3513)

Allocation	Format Prefix (binary)	Fraction of Address Space
Reserved	0000 0000	1/256
Unassigned	0000 0001	1/256
Reserved for NSAP allocation	0000 001	1/128
Unassigned	0000 01	1/64
Unassigned	0000 1	1/32
Unassigned	0001	1/16
Global unicast address **(IANA)**	001	1/8
Unassigned	010	1/8
Unassigned	011	1/8
Unassigned	100	1/8

Initial Assignment of Addresses (RFC 3513)

Allocation	Format Prefix (binary)	Fraction of Address Space
Unassigned	101	1/8
Unassigned	110	1/8
Unassigned	1110	1/16
Unassigned	1111 0	1/32
Unassigned	1111 10	1/64
Unassigned	1111 110	1/128
Unassigned	1111 1110 0	1/512
Link local-use addresses	1111 1110 10	1/1024
Site local-use addresses	1111 1110 11	1/1024
Multicast addresses	1111	1/256

Global Unicast Address (RFC 3513)



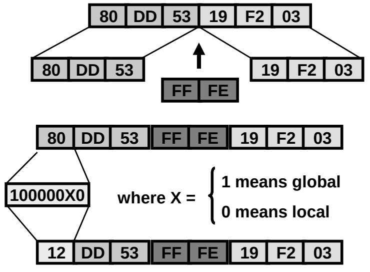
<u>Global Routing Prefix</u> is a (typically hierarchically-structured) value assigned to a site (a cluster of subnets/links)
<u>Subnet ID</u> is an identifier of a link within the site
<u>Interface ID</u> are used to identify interfaces on a link

global unicast addresses not starting with binary 000
have a 64-bit interface ID field,
examples can be found in RFC2373

global unicast addresses starting with binary 000
have no such constraint on the size or structure of the interface ID field,
examples are the IPv6 address with embedded IPv4 addresses and the IPv6 address containing encoded NSAP addresses

How to create an Interface-ID?

Ethernet MAC address (48bits, global, individual) taken as unique "Token"



80 DD 53 19 F2 03

Expansion to 64 bits

80 DD 53 **FF FE** **19 F2 03**

80 DD 53 FF FE 19 F2 03

Modify U/L bit meaning of the first byte of a MAC address

100000X0 **where X =** **1 means global** **0 means local**

12 DD 53 FF FE 19 F2 03

Modified EUI-64 address

Original IEEE meaning of first byte of a MAC address (hex 80 in our example):

100000 U/L I/G (bit 7 … 0)
U/L = 0 … global
U/L = 1 … local
I/G = 0 … individual
I/G = 1 … group

Local Use Addresses

10	54-bits	64-bits
F E C 0	Subnet ID	Interface ID

Site-local unicast

cannot be routed on the global Internet,
their uniqueness is guaranteed within a site

10	54-bits	64-bits
F E 8 0	0 0 0 0 0 0 0 0	Interface ID

Link-local unicast

defined only within a link and can only be used by stations
connected to the same link or the same local network

Addresses with Embedded IPv4 Addresses

80-bits	16-bits	32-bits
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0	IPv4 Address

IPv4-compatible IPv6 address

used by hosts and routers which tunnel IPv6 packets dynamically
over a IPv4 infrastructure (e.g.: ::193.170.150.1/96)
(tunneling is one transition technique for IPv4 ⇨ IPv6)

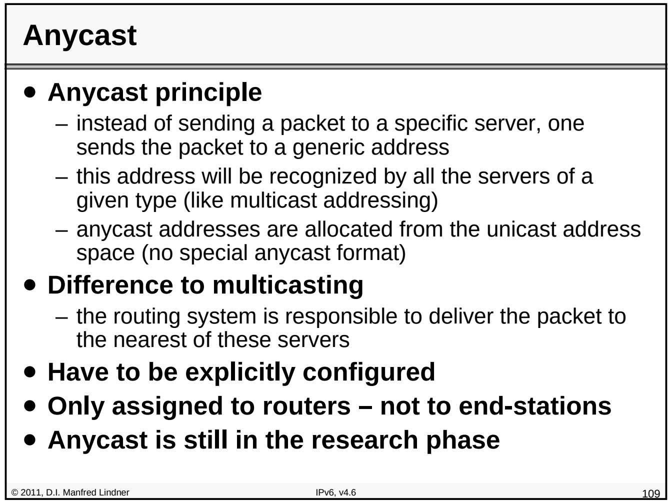
80-bits	16-bits	32-bits
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	F F F F	IPv4 Address

IPv4-mapped IPv6 address

represents address of IPv4-only hosts,
used by hosts that do translation between IPv4 and IPv6 (e.g.: FFFF:193.170.150.1/80)
(translation is another transition technique for IPv4 ⇨ IPv6)

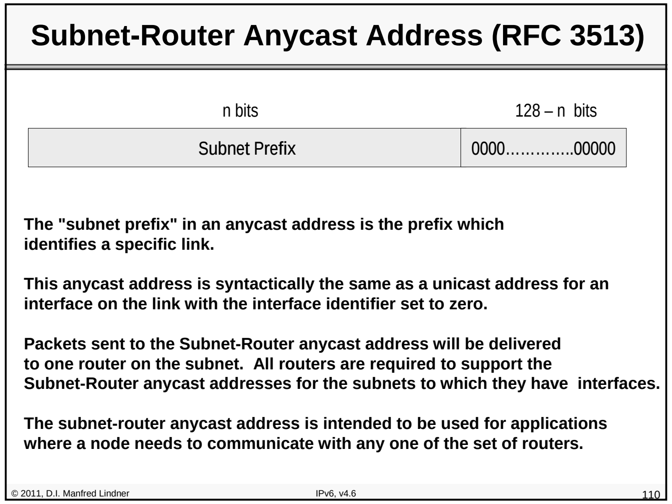
Special Addresses / Anycast

- **Unspecified address:**
  - 0:0:0:0:0:0:0:0 or ::
  - can only be used as a source address by station that has not yet been configured with a regular address or as placeholder in some control messages
- **Loopback address:**
  - 0:0:0:0:0:0:0:1 or ::1
  - used by a node to send IPv6 packets to itself
- **Anycast:**
  - new concept in IPv6 to address a group of interfaces
  - is an address that is assigned to more than one interface (typically belonging to different nodes)



Anycast

- **Anycast principle**
  - – instead of sending a packet to a specific server, one sends the packet to a generic address
  - – this address will be recognized by all the servers of a given type (like multicast addressing)
  - – anycast addresses are allocated from the unicast address space (no special anycast format)
- **Difference to multicasting**
  - – the routing system is responsible to deliver the packet to the nearest of these servers
- **Have to be explicitly configured**
- **Only assigned to routers – not to end-stations**
- **Anycast is still in the research phase**



Subnet-Router Anycast Address (RFC 3513)

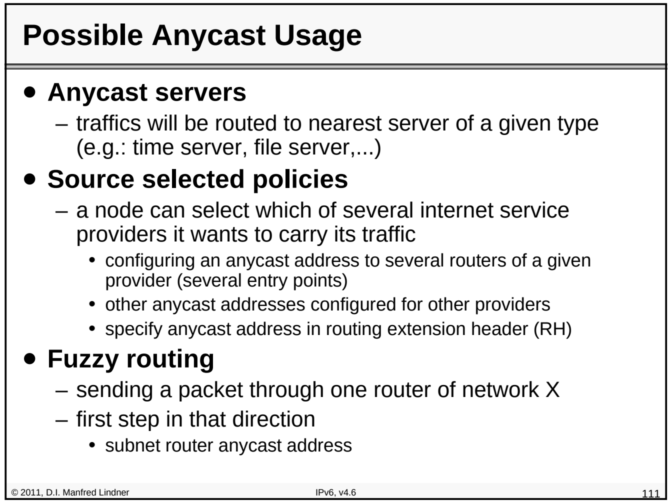
n bits	128 – n bits
Subnet Prefix	0000………….00000

The "subnet prefix" in an anycast address is the prefix which identifies a specific link.

This anycast address is syntactically the same as a unicast address for an interface on the link with the interface identifier set to zero.

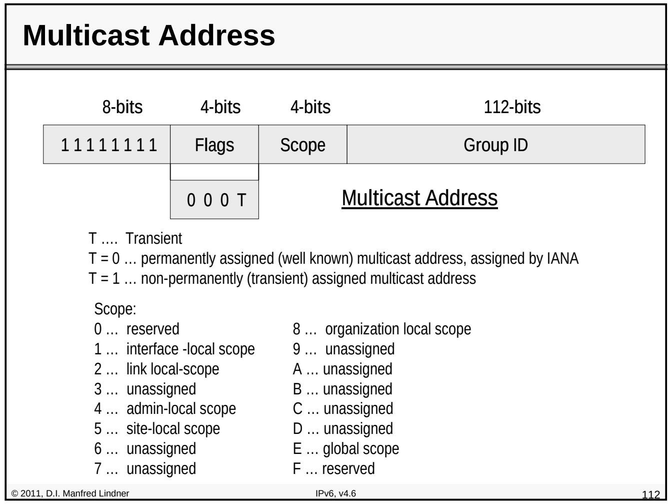
Packets sent to the Subnet-Router anycast address will be delivered to one router on the subnet. All routers are required to support the Subnet-Router anycast addresses for the subnets to which they have interfaces.

The subnet-router anycast address is intended to be used for applications where a node needs to communicate with any one of the set of routers.



Possible Anycast Usage

- **Anycast servers**
  - – traffics will be routed to nearest server of a given type (e.g.: time server, file server,...)
- **Source selected policies**
  - – a node can select which of several internet service providers it wants to carry its traffic
    - configuring an anycast address to several routers of a given provider (several entry points)
    - other anycast addresses configured for other providers
    - specify anycast address in routing extension header (RH)
- **Fuzzy routing**
  - – sending a packet through one router of network X
  - – first step in that direction
    - subnet router anycast address



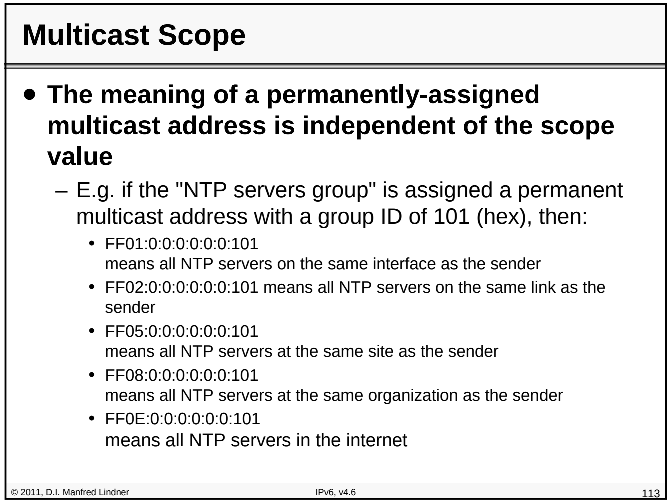
Multicast Address

8-bits	4-bits	4-bits	112-bits
1 1 1 1 1 1 1 1	Flags	Scope	Group ID
	0 0 0 T		

T …. Transient
T = 0 … permanently assigned (well known) multicast address, assigned by IANA
T = 1 … non-permanently (transient) assigned multicast address

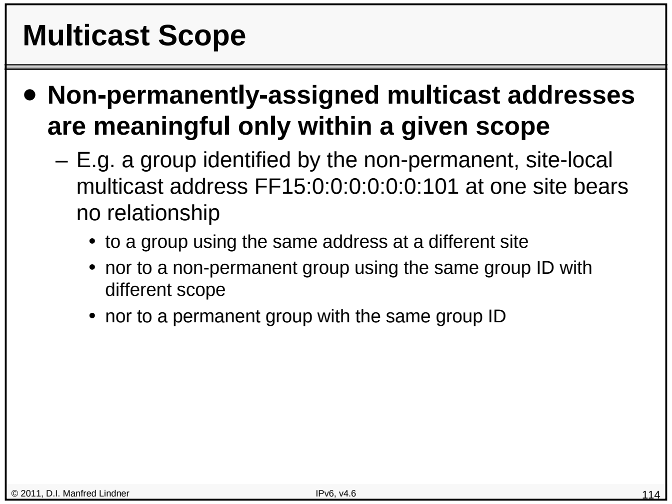
Scope:

0 … reserved	8 … organization local scope
1 … interface -local scope	9 … unassigned
2 … link local-scope	A … unassigned
3 … unassigned	B … unassigned
4 … admin-local scope	C … unassigned
5 … site-local scope	D … unassigned
6 … unassigned	E … global scope
7 … unassigned	F … reserved



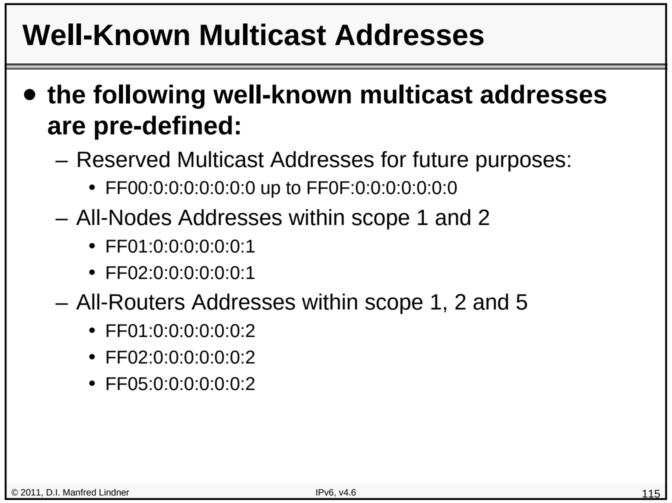
Multicast Scope

- **The meaning of a permanently-assigned multicast address is independent of the scope value**
  - E.g. if the "NTP servers group" is assigned a permanent multicast address with a group ID of 101 (hex), then:
    - FF01:0:0:0:0:0:0:101
      means all NTP servers on the same interface as the sender
    - FF02:0:0:0:0:0:0:101 means all NTP servers on the same link as the sender
    - FF05:0:0:0:0:0:0:101
      means all NTP servers at the same site as the sender
    - FF08:0:0:0:0:0:0:101
      means all NTP servers at the same organization as the sender
    - FF0E:0:0:0:0:0:0:101
      means all NTP servers in the internet



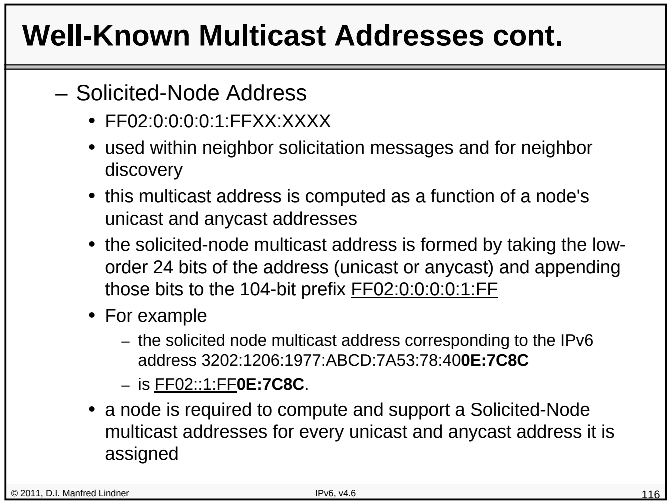
Multicast Scope

- **Non-permanently-assigned multicast addresses are meaningful only within a given scope**
  - E.g. a group identified by the non-permanent, site-local multicast address FF15:0:0:0:0:0:0:101 at one site bears no relationship
    - to a group using the same address at a different site
    - nor to a non-permanent group using the same group ID with different scope
    - nor to a permanent group with the same group ID



Well-Known Multicast Addresses

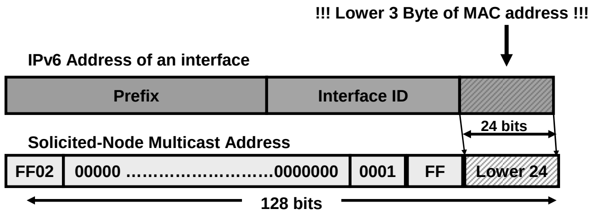
- **the following well-known multicast addresses are pre-defined:**
  - Reserved Multicast Addresses for future purposes:
    - FF00:0:0:0:0:0:0:0 up to FF0F:0:0:0:0:0:0:0
  - All-Nodes Addresses within scope 1 and 2
    - FF01:0:0:0:0:0:0:1
    - FF02:0:0:0:0:0:0:1
  - All-Routers Addresses within scope 1, 2 and 5
    - FF01:0:0:0:0:0:0:2
    - FF02:0:0:0:0:0:0:2
    - FF05:0:0:0:0:0:0:2



Well-Known Multicast Addresses cont.

- Solicited-Node Address
  - FF02:0:0:0:0:1:FFXX:XXXX
  - used within neighbor solicitation messages and for neighbor discovery
  - this multicast address is computed as a function of a node's unicast and anycast addresses
  - the solicited-node multicast address is formed by taking the low-order 24 bits of the address (unicast or anycast) and appending those bits to the 104-bit prefix FF02:0:0:0:0:1:FF
  - For example
    - the solicited node multicast address corresponding to the IPv6 address 3202:1206:1977:ABCD:7A53:78:40**0E:7C8C**
    - is FF02::1:FF**0E:7C8C**.
  - a node is required to compute and support a Solicited-Node multicast addresses for every unicast and anycast address it is assigned

Solicited-Node Address

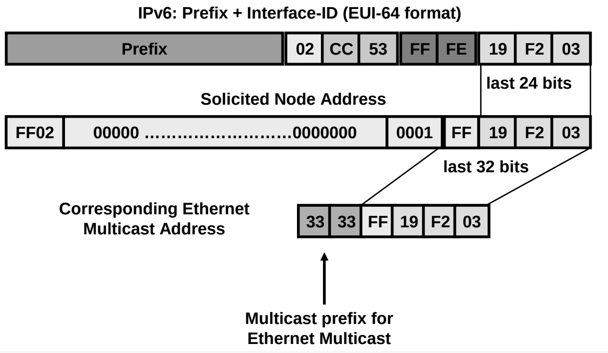

!!! Lower 3 Byte of MAC address !!!
IPv6 Address of an interface
Prefix
Interface ID
24 bits
Solicited-Node Multicast Address
FF02
00000 ………………………0000000
0001
FF
Lower 24
128 bits

This is a special multicast address which is derived from the MAC address -> there will only an overlap if two machines have the same lower 3 bytes of the MAC address

Remember: Broadcasts are not possible, in order to make such things like ARP you specify this address instead of a broadcast

Mapping IPv6 Multicast to Ethernet Multicast

Example: How do generate a Ethernet-MC from the Solicited Node Address

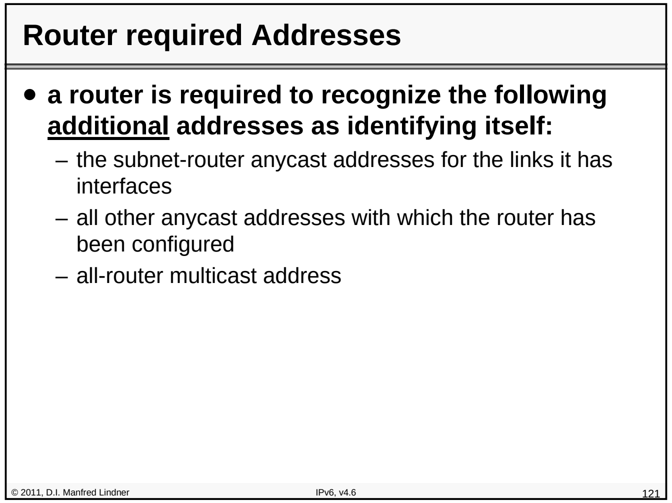

IPv6: Prefix + Interface-ID (EUI-64 format)
Prefix
02
CC
53
FF
FE
19
F2
03
last 24 bits
Solicited Node Address
FF02
00000 ………………………0000000
0001
FF
19
F2
03
last 32 bits
Corresponding Ethernet Multicast Address
33
33
FF
19
F2
03
Multicast prefix for Ethernet Multicast

Multicast Group Management

- **Multicast membership handling for transient groups in IPv4**
  - done by IGMP (RFC 1112, 2236, 3376)
- **Multicast membership handling for transient groups in IPv6**
  - done by ICMPv6 which includes tasks of IGMP
  - RFC 2463 (1885)
    - new ICMP types defined for group management
      - type 130 … Group Membership Query
      - type 131 … Group Membership Report
      - type 132 … Group Membership Termination
  - procedure for a station to join a multicast group is similar to IPv4

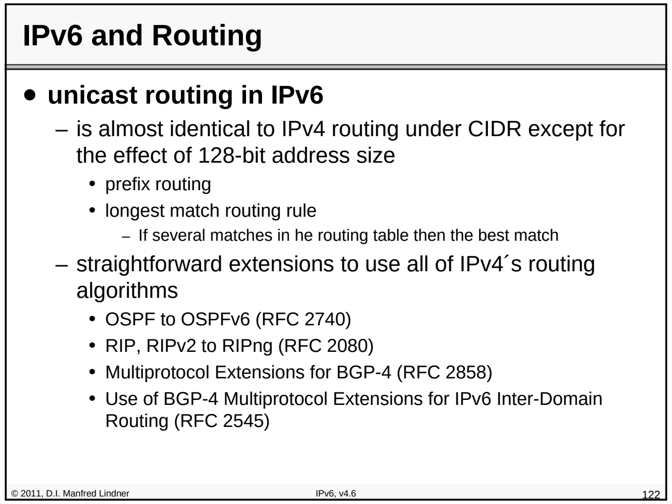
Host required Addresses

- **A host is required to recognize the following addresses as identifying itself:**
  - its link-local address for each interface
  - any additional assigned unicast addresses
  - loopback address
  - all-nodes multicast address
  - solicited-node multicast address for each of its assigned unicast and anycast addresses
  - multicast addresses of all other groups which the host belongs



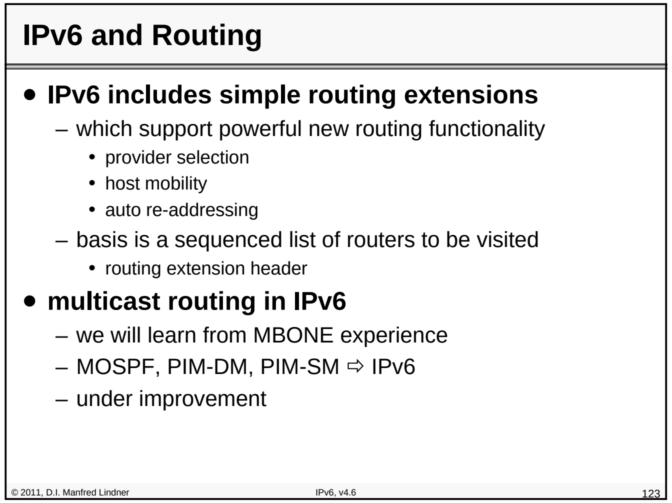
Router required Addresses

- **a router is required to recognize the following <u>additional</u> addresses as identifying itself:**
  - – the subnet-router anycast addresses for the links it has interfaces
  - – all other anycast addresses with which the router has been configured
  - – all-router multicast address



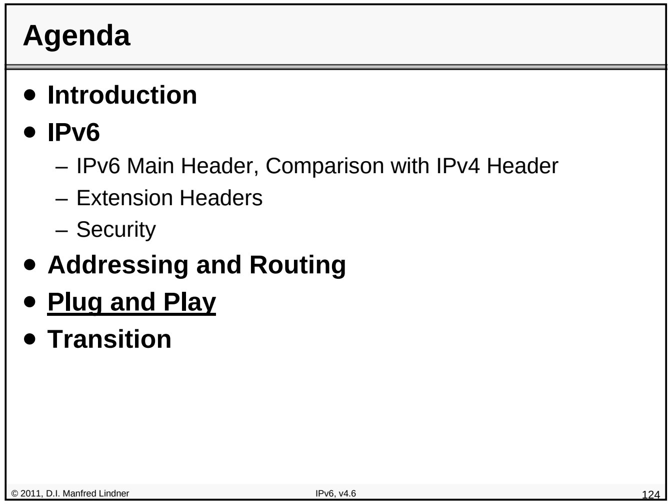
IPv6 and Routing

- **unicast routing in IPv6**
  - – is almost identical to IPv4 routing under CIDR except for the effect of 128-bit address size
    - prefix routing
    - longest match routing rule
      - – If several matches in he routing table then the best match
  - – straightforward extensions to use all of IPv4´s routing algorithms
    - OSPF to OSPFv6 (RFC 2740)
    - RIP, RIPv2 to RIPng (RFC 2080)
    - Multiprotocol Extensions for BGP-4 (RFC 2858)
    - Use of BGP-4 Multiprotocol Extensions for IPv6 Inter-Domain Routing (RFC 2545)



IPv6 and Routing

- **IPv6 includes simple routing extensions**
  - – which support powerful new routing functionality
    - provider selection
    - host mobility
    - auto re-addressing
  - – basis is a sequenced list of routers to be visited
    - routing extension header
- **multicast routing in IPv6**
  - – we will learn from MBONE experience
  - – MOSPF, PIM-DM, PIM-SM ⇨ IPv6
  - – under improvement



Agenda

- **Introduction**
- **IPv6**
  - – IPv6 Main Header, Comparison with IPv4 Header
  - – Extension Headers
  - – Security
- **Addressing and Routing**
- **<u>Plug and Play</u>**
- **Transition**

Base Elements for Plug and Play 1

- **In order to understand the different forms of auto-configuration**
  - a detailed knowledge about ICMPv6 and “Neighbor Discovery” and “IPv6 stateless Address Auto-configuration” is necessary
- **ICMPv6 base elements defined in RFC 2463**
  - ICMPv4 adapted to IPv6
    - error messages
      - ICMPv6 type 1 … destination unreachable
      - ICMPv6 type 2 … packet too big
      - ICMPv6 type 3 … time exceeded
      - ICMPv6 type 4 … parameter problem
    - ping
      - ICMPv6 type 128 … echo request
      - ICMPv6 type 129 … echo reply

Base Elements for Plug and Play 2

- **“Neighbor Discovery” defined in RFC 2461**
  - router and prefix discovery, address resolution and redirect
  - the following ICMP message types are used
    - Router Solicitation (ICMPv6 type 133)
    - Router Advertisement (ICMPv6 type 134)
    - Neighbor Solicitation (ICMPv6 type 135)
    - Neighbor Advertisements (ICMPv6 type 136)
    - Redirect (ICMPv6 type 137)
- **“IPv6 stateless Address Auto-configuration” defined in RFC 2462**
  - stateless and even in a server-less or router-less environment

RFC 2461 Procedures

- **Includes the following procedures**
  - Router Discovery:
    - How hosts locate routers that reside on an attached link
  - Prefix Discovery:
    - How hosts discover the set of address prefixes that define which destinations are on-link for an attached link (nodes use prefixes to distinguish destinations that reside on-link from those only reachable through a router)
  - Parameter Discovery:
    - How a node learns link parameters such as the link MTU or Internet parameters such as the hop limit value to place in outgoing packets.

RFC 2461 Procedures (cont.)

- Address resolution:
  - How nodes determine the link-layer address of an on-link destination (e.g., a neighbor) given only the destination's IP address.
- Duplicate Address Detection:
  - How a node determines that an address it wishes to use is not already in use by another node.
- Redirect:
  - How a router informs a host of a better first-hop node to reach a particular destination.
- Neighbor Unreachability Detection:
  - How nodes determine that a neighbor is no longer reachable. For neighbors used as routers, alternate default routers can be tried. For both routers and hosts, address resolution can be performed again.

RFC 2461 Procedures (cont.)

- Next-hop determination:
  - The algorithm for mapping an IP destination address into the IP address of the neighbor to which traffic for the destination should be sent. The next-hop can be a router or the destination itself.
- Address Auto-configuration:
  - How nodes automatically configure an address for an interface.

- **Protocol procedures uses the following well-known multicast types or address types**
  - all-nodes multicast address
  - all-routers multicast address
  - solicited-node multicast address
  - link-local address
  - unspecified address

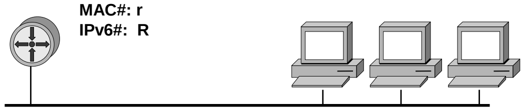
RFC 2461 ICMPv6 Messages

- **The ICMP messages serve the following purpose:**
  - Router Solicitation (RS):
    - When an interface becomes enabled, hosts may send out **Router Solicitations** that request routers to generate **Router Advertisements** immediately rather than at their next scheduled time.
  - Router Advertisement (RA):
    - Routers advertise their presence together with various link and Internet parameters either periodically, or in response to a Router Solicitation message. **Router Advertisements** contain prefixes that are used for on-link determination and/or address configuration, a suggested hop limit value, etc.

RFC 2461 ICMPv6 Messages (cont.)

- Neighbor Solicitation (NS):
  - Sent by a node to determine the link-layer address of a neighbor, or to verify that a neighbor is still reachable via a cached link-layer address. Neighbor Solicitations are also used for Duplicate Address Detection.
- Neighbor Advertisement (NA):
  - A response to a Neighbor Solicitation message. A node may also send unsolicited Neighbor Advertisements to announce a link-layer address change.
- Redirect:
  - Used by routers to inform hosts of a better first hop for a destination.

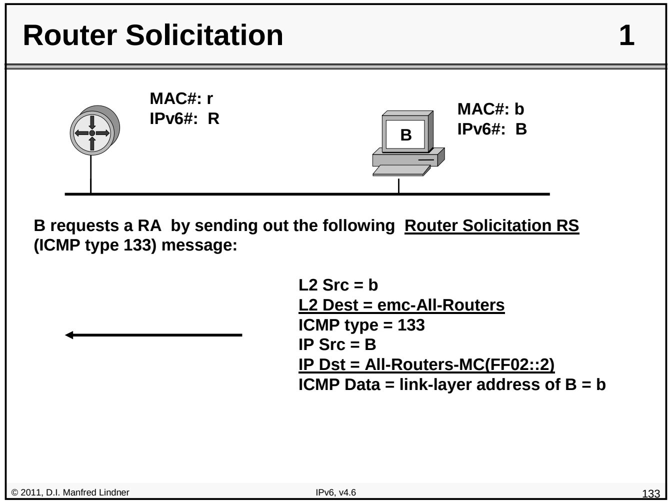
Router Discovery



Router R periodically announces IPv6 prefixes (site-local, global) by sending out the following Router Advertisement RA (ICMP type 134) message in multicast style:

L2 Src = r
L2 Dest = emc-All-Nodes
ICMP type = 134
IP Src = R
IP Dst = All-Nodes-MC (FF02::1)
ICMP Data = prefixes, lifetime, other configuration parameters (MTU, Hop Limit, control bits for auto-configuration, ….)

Hosts use this message to fill the Default Router List and the Prefix List



Router Solicitation 1

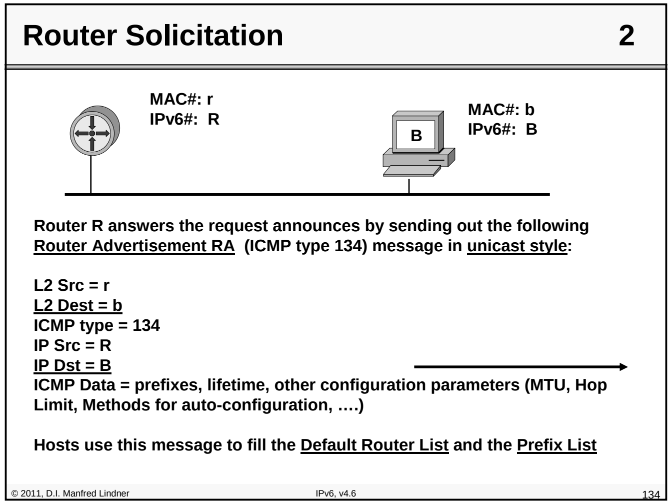
MAC#: r
IPv6#: R

B

MAC#: b
IPv6#: B

B requests a RA by sending out the following Router Solicitation RS (ICMP type 133) message:

L2 Src = b
L2 Dest = emc-All-Routers
ICMP type = 133
IP Src = B
IP Dst = All-Routers-MC(FF02::2)
ICMP Data = link-layer address of B = b



Router Solicitation 2

MAC#: r
IPv6#: R

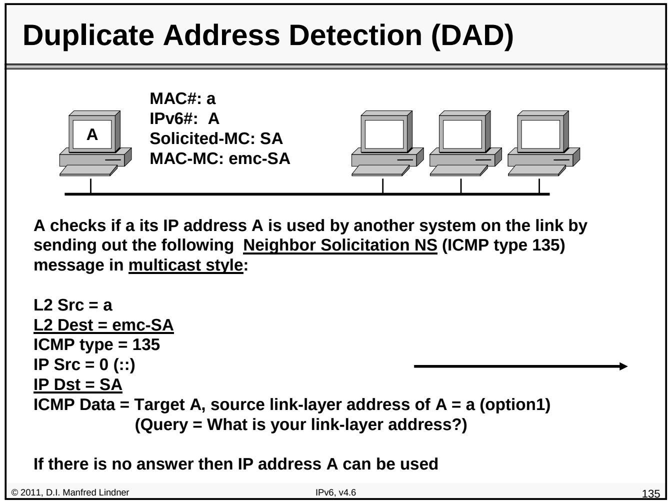
B

MAC#: b
IPv6#: B

Router R answers the request announces by sending out the following Router Advertisement RA (ICMP type 134) message in unicast style:

L2 Src = r
L2 Dest = b
ICMP type = 134
IP Src = R
IP Dst = B
ICMP Data = prefixes, lifetime, other configuration parameters (MTU, Hop Limit, Methods for auto-configuration, ….)

Hosts use this message to fill the Default Router List and the Prefix List



Duplicate Address Detection (DAD)

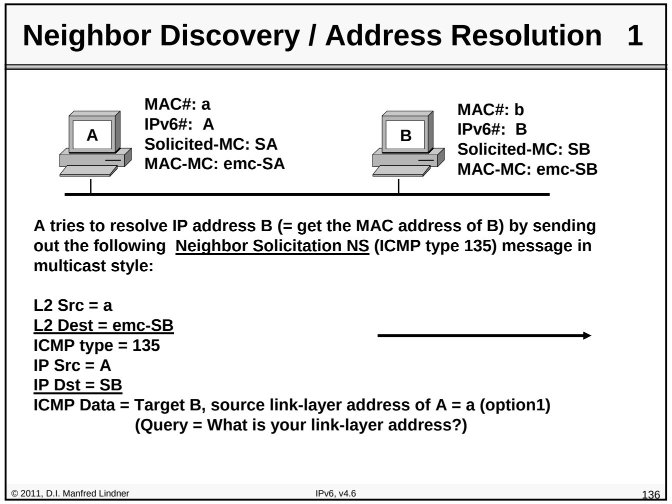
A

MAC#: a
IPv6#: A
Solicited-MC: SA
MAC-MC: emc-SA

A checks if a its IP address A is used by another system on the link by sending out the following Neighbor Solicitation NS (ICMP type 135) message in multicast style:

L2 Src = a
L2 Dest = emc-SA
ICMP type = 135
IP Src = 0 (::)
IP Dst = SA
ICMP Data = Target A, source link-layer address of A = a (option1)
(Query = What is your link-layer address?)

If there is no answer then IP address A can be used



Neighbor Discovery / Address Resolution 1

A

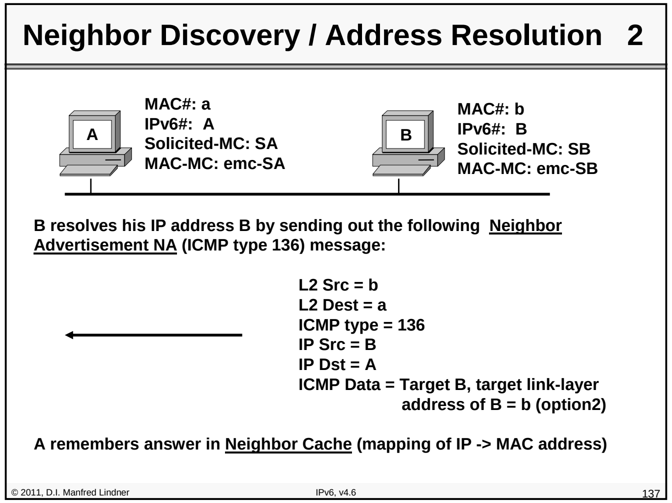
MAC#: a
IPv6#: A
Solicited-MC: SA
MAC-MC: emc-SA

B

MAC#: b
IPv6#: B
Solicited-MC: SB
MAC-MC: emc-SB

A tries to resolve IP address B (= get the MAC address of B) by sending out the following Neighbor Solicitation NS (ICMP type 135) message in multicast style:

L2 Src = a
L2 Dest = emc-SB
ICMP type = 135
IP Src = A
IP Dst = SB
ICMP Data = Target B, source link-layer address of A = a (option1)
(Query = What is your link-layer address?)



Neighbor Discovery / Address Resolution 2

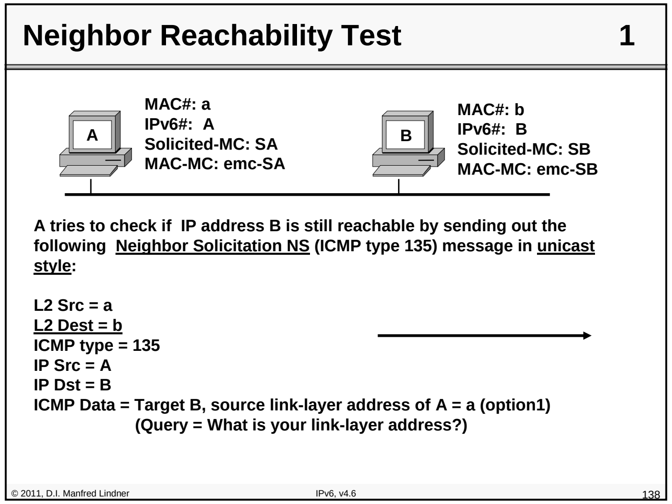
MAC#: a
IPv6#: A
Solicited-MC: SA
MAC-MC: emc-SA

MAC#: b
IPv6#: B
Solicited-MC: SB
MAC-MC: emc-SB

B resolves his IP address B by sending out the following <u>Neighbor Advertisement NA</u> (ICMP type 136) message:

L2 Src = b
L2 Dest = a
ICMP type = 136
IP Src = B
IP Dst = A
ICMP Data = Target B, target link-layer address of B = b (option2)

A remembers answer in <u>Neighbor Cache</u> (mapping of IP -> MAC address)



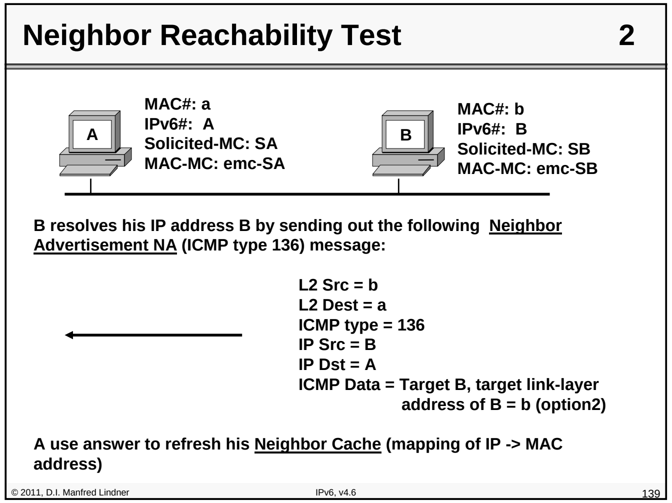
Neighbor Reachability Test 1

MAC#: a
IPv6#: A
Solicited-MC: SA
MAC-MC: emc-SA

MAC#: b
IPv6#: B
Solicited-MC: SB
MAC-MC: emc-SB

A tries to check if IP address B is still reachable by sending out the following <u>Neighbor Solicitation NS</u> (ICMP type 135) message in <u>unicast style</u>:

L2 Src = a
<u>L2 Dest = b</u>
ICMP type = 135
IP Src = A
IP Dst = B
ICMP Data = Target B, source link-layer address of A = a (option1)
(Query = What is your link-layer address?)



Neighbor Reachability Test 2

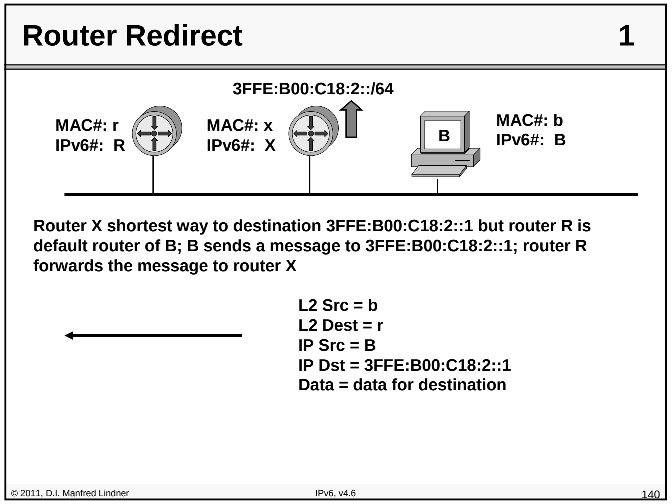
MAC#: a
IPv6#: A
Solicited-MC: SA
MAC-MC: emc-SA

MAC#: b
IPv6#: B
Solicited-MC: SB
MAC-MC: emc-SB

B resolves his IP address B by sending out the following <u>Neighbor Advertisement NA</u> (ICMP type 136) message:

L2 Src = b
L2 Dest = a
ICMP type = 136
IP Src = B
IP Dst = A
ICMP Data = Target B, target link-layer address of B = b (option2)

A use answer to refresh his <u>Neighbor Cache</u> (mapping of IP -> MAC address)



Router Redirect 1

MAC#: r
IPv6#: R

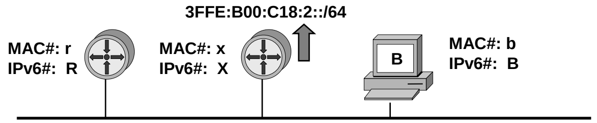
MAC#: x
IPv6#: X

MAC#: b
IPv6#: B

Router X shortest way to destination 3FFE:B00:C18:2::1 but router R is default router of B; B sends a message to 3FFE:B00:C18:2::1; router R forwards the message to router X

L2 Src = b
L2 Dest = r
IP Src = B
IP Dst = 3FFE:B00:C18:2::1
Data = data for destination

Router Redirect 2

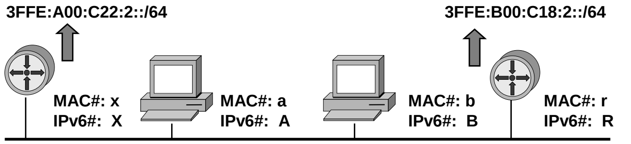


Router R informs host B to take router X as next hop in order to reach 3FFE:B00:C18:2::1 by sending out the following Router Redirect (ICMP type 137) message in unicast style:

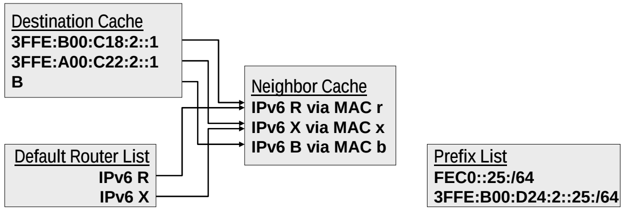
L2 Src = r
L2 Dest = b
ICMP type = 137
IP Src = R
IP Dst = B
ICMP Data = Target X, destination address 3FFE:B00:C18:2::1
(use next hop X to reach 3FFE:B00:C18:2::1)
target link-layer address of X = x (option2)

Host use this message to actualize the Destination Cache

Example for Caches in System A



Prefix (Net-ID): FEC0::25:/64 (site-local) and 3FFE:B00:D24:2::25:/64 (global)


Destination Cache
3FFE:B00:C18:2::1
3FFE:A00:C22:2::1
B

Neighbor Cache
IPv6 R via MAC r
IPv6 X via MAC x
IPv6 B via MAC b

Default Router List
IPv6 R
IPv6 X

Prefix List
FEC0::25:/64
3FFE:B00:D24:2::25:/64

Address Auto-Configuration

- **A feature**
  - that enables host to configure one or more addresses per interface automatically
- **Allows plug and play operation of a host**
  - one weaknesses of IPv4
- **Allows re-addressing of a host in case of**
  - change of location or change of service provider
- **In IPv4**
  - BOOTP and DHCP servers could be used for address configuration
    - BOOTP depends on statically and manually database entries
    - DHCP can support dynamic reconfiguration (⇨ lifetime)

Address Auto-Configuration

- **Address configuration done by BOOTP and DHCP**
  - depends on presence of a server
  - stateful address auto-configuration
- **One challenge for IPv6 was**
  - to provide stateless address auto-configuration in addition do stateful configuration performed by DHCPv6
- **Stateless**
  - host can obtain an address without any database pre-configuration of a server (no manual configuration)
  - host can obtain an address even in a server-less and router-less environment

Stateless Auto Configuration

- **Every interface is pre-configured with a token**
  - token must be unique per link
    - e.g. 48 bit MAC address
  - token must be suitable for use as the Interface ID portion of an IPv6 address (EUI-64 addressing mechanism to get a 64 bit Interface ID)
- **In router-less/server-less topology**
  - the link-local address can be formed autonomously by the node
    - using the token as Interface ID
    - using the well known prefix of such an IPv6 address type
      - FE80:0:0:0:EUI-64 address or FE80::EUI-64 address
  - after a duplicate test (unspecified address as source address and link local address as destination address) the node can communicate with other nodes on the same link using this address

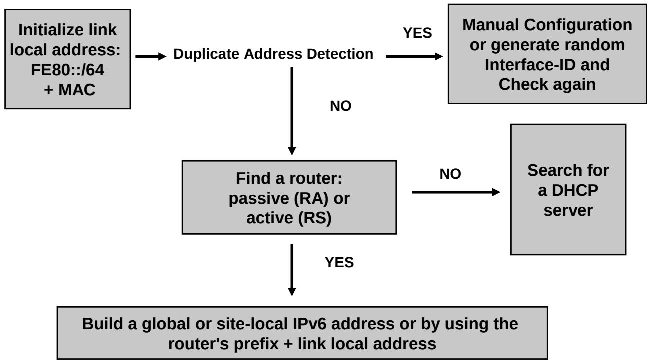
Stateless Auto Configuration (cont.)

- **In topologies with routers**
  - a host can determine the current prefix associated with the link
    - current prefix information is announced periodically by routers or may be solicited by the host from the routers on request
  - a host can use these prefixes together with the token to form a site-global or Internet-global IPv6 address
  - re-addressing can be done dynamically
    - by the help of two lifetimes associated with an announced prefix
    - Valid Lifetime
      - indicates how long the address formed from that prefix is valid
    - Preferred Lifetime
      - indicates when the address formed from the prefix will be deprecated

Stateless Auto Configuration (cont.)

- **Whenever the prefix is re-advertised**
  - the valid lifetime of the address is reset to the new value
- **When the prefix is not longer advertised**
  - the address will expire after the last advertised lifetime runs out
    - can not be used any longer ⇨ harsh consequences on ongoing communications (e.g. TCP session will break)
- **The preferred lifetime can be used**
  - to indicate that an address (prefix) - although still valid - is about to become invalid
  - hence a node should not use this address (prefix) as source address when initiating new communications
  - the node will choose another non-deprecated address for new communications instead

IPv6 Address Configuration Overview


Initialize link local address: FE80::/64 + MAC
Duplicate Address Detection
YES
Manual Configuration or generate random Interface-ID and Check again
NO
Find a router: passive (RA) or active (RS)
NO
Search for a DHCP server
YES
Build a global or site-local IPv6 address or by using the router's prefix + link local address

Duplicate Address Detection (DAD)

- **Uses Neighbor Solicitation (NS) to check if another node on the link has the same IPv6 address**
- **DAD is used during the auto-configuration process**
  - It sends an NS packet to the solicited-node multicast address of ist own IPv6 address.
  - The source address of this packet is the „unspecified address“::
  - If a node responds to that request, it means that the IPv6 address is used and the requesting node should not use that address

Stateless Autoconfiguration - RA

- **Router Advertisements (RA)**
  - Are sent periodically, and on request, by routers on all their configured interfaces
  - Is sent to the all-nodes multicast address
  - Message info:
    - One or more prefixes that can be used on the link
    - Lifetime of the prefixes
    - Default router information: existence and lifetime
    - Flags indicating the kind of autoconfiguration that can be done by hosts

Stateless Autoconfiguration - RS

- **Router Solicitations (RS)**
  - Are sent by hosts at boot time
  - To ask routers to send an immediate RA on the local link
  - So hosts can receive the autoconfiguration information without waiting for the next scheduled RA
  - To avoid over flooding, RS should only be sent at boot time and only 3 times

Neighbor Unreachability Detection

- **Two possible scenarios for unreachable neighbors:**
  - If the end nodes are concerned
    - no recovery is possible
  - If the path between 2 nodes is concerned and an alternative path exists
    - communication could be continued without upper layers detecting any change but what if the “Neighbor Cache” points into a “black hole” for the lifetime of an entry?
- **Therefore**
  - If an entry of the “Neighbor Cache” is not refreshed within 30 sec by normal activity it changes to a „Probe state“
  - 3 probe packets are sent and if there is no reply the entry gets deleted

Renumbering

- **Renumbering**
  - Is achieved by sending RAs
  - They contain both the old and the new prefix
    - Old prefix with short lifetime
    - New prefix with nomal lifetime
  - The old prefix gets deprecated
    - That means nodes should use the new prefix for new connections while still keeping their current connections opened with the old prefix
  - During a certain time, node has two unicast addresses

Stateful Autoconfiguration

- **DHCP**
  - Dynamic Host Configuration Protocol (version 6)
  - Provide clients with configuration information that is stored on a server
  - Updated version of DHCP for IPv4
  - Can be used for automatic domain registration of hosts using dynamic DNS

DHCP (cont.)

- **How does it work?**
  - Clients first detect the presence of routers on the link
    - If found, then examines RAs to determine if DHCP can be used
  - If no router is found or if DHCP can be used
    - **DHCP Solicit** message is sent to “All-DHCP-Agents and Relay Agents” multicast address (FF02::1:2) with link-local scope
      - Using the link-local address as the source address
    - If no local DHCP server is present on the link but a DHCP relay agent is implemented on a machine
      - DHCP relay agents forwards the request to the “All-DHCP-Server” multicast address (FF05::1:3) which is site scope
    - The DHCP server responds with a **DHCP Advertise** message
      - This message contains one or more IPv6 unicast addresses of DHCP servers
    - By using **DHCP Request** and **DHCP Reply** messages the address can be delivered to the host
    - By using **DHCP Release** or **DHCP Reconfigure** messages the address can be returned or refreshed

Agenda

- **Introduction**
- **IPv6**
  - IPv6 Main Header, Comparison with IPv4 Header
  - Extension Headers
  - Security
- **Addressing and Routing**
- **Plug and Play**
- **Transition**

Transition IPv4 to IPv6

- **Transition mechanism must ensure an easy evolution from IPv4 to IPv6**
  - Otherwise IPv6 will not be accepted
  - IP society has learned the lessons experienced by similar approaches in other protocol worlds
    - Decnet IV to Decnet OSI
    - AppleTalk Phase 1 to Phase 2
- **Several transition mechanisms IPv4 to IPv6**
  - Described in RFC 2893 (Proposed Standard)
    - "The key to a successful IPv6 transition is compatibility with the large installed base of IPv4 hosts and routers. Maintaining compatibility with IPv4 while deploying IPv6 will streamline the task of transitioning the Internet to IPv6.
  - RFC 4038 "Application Aspects of IPv6 Transition" (Informational)

Major Elements for Transition Mechanisms

(from RCF 2893)

- **Dual-IP layer (also known as Dual-Stack):**
  - A technique for providing complete support for both Internet protocols -- IPv4 and IPv6 -- in hosts and routers.
- **Configured tunneling of IPv6 over IPv4:**
  - Point-to-point tunnels made by encapsulating IPv6 packets within IPv4 headers to carry them over IPv4 routing infrastructures.
- **IPv4-compatible IPv6 addresses:**
  - An IPv6 address format that employs embedded IPv4 addresses.
- **Automatic tunneling of IPv6 over IPv4:**
  - A mechanism for using IPv4-compatible addresses to automatically tunnel IPv6 packets over IPv4 networks.

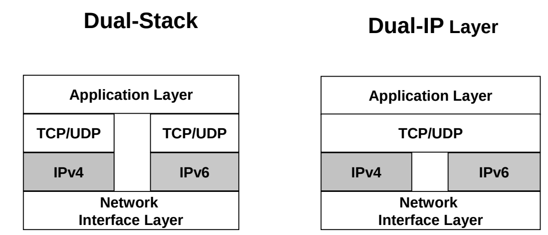
Transition Approaches

- **Dual-Stack Mechanisms**
  - Dual-Stack
  - Dual-Stack Dominant Transition Mechanism (DSTM)
- **Tunneling Mechanisms**
  - IPv4-Compatible Tunnel
  - 6to4
  - Tunnel Broker
  - 6over4
  - Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)
  - Teredo
- **Translation**
  - Stateless IP/ICMP Translator (SIIT)
  - Bump in the Stack (BITSv6)
  - Bump in the API (BIA)
  - Network Address Translation -Protocol Translation (NAT-PT)
  - Transport Relay Translator (TRT)
  - SOCKS64

Dual-Stack / Dual-IP Layer

- First dual-stacks are implemented in network devices
  - Allow handling of both IPv4 and IPv6 packet types
- Then end systems are gradually upgraded over a certain period of time from IPv4 to IPv6
  - Upgraded hosts can communicate with both IPv4 and IPv6 nodes using the corresponding native protocol. Applications have to be modified to IPv6. Old applications will still take IPv4 and new or modified applications will take IPv6.
- Finally dual-stacks are implemented in all end systems
  - Both IPv4- and IPv6-capable applications can operate on the same node
- Name servers and routers will provide support for both IPv4 and IPv6 during the transition period

Dual-Stack versus Dual-IP Layer



Dual-Stack

Application Layer

TCP/UDP | TCP/UDP

IPv4 | IPv6

Network Interface Layer

Dual-IP Layer

Application Layer

TCP/UDP

IPv4 | IPv6

Network Interface Layer

Separate driver implementation of TCP/UDP for IPv4 and IPv6

Dual-Stack / Dual-IP Layer

- **Necessary steps:**
  - Small DNS upgrade to support IPv6 addresses
  - Relatively small host upgrade to provide
    - IPv6, ICMPv6, Neighbor Discovery, handling of IPv6 within TCP and UDP, sockets or Winsock libraries, interface with the name service
  - Slightly more complex router upgrade to provide
    - IPv6 forwarding, IPv6 routing, IPv6 management
    - should not be a problem for "ships in the night" multiprotocol router

Dual-Stack / Dual-IP Layer

- **But dual-stack mechanisms do not solve IPv4 and IPv6 interoperation problems**
  - That will be the case if not all network components or end systems can be migrated to IPv6
  - Therefore translation between IPv4 and IPv6 is also required for this
- **Address issue solved?**
  - For classical dual-stack method every IPv6 node needs also an unique IPv4 address
    - This is not applicable for practical implementations

Transition Approaches

- **Dual-Stack Mechanisms**
  - Dual-Stack
  - Dual-Stack Dominant Transition Mechanism (DSTM)
- **Tunneling Mechanisms**
  - IPv4-Compatible Tunnel
  - 6to4
  - Tunnel Broker
  - 6over4
  - Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)
  - Teredo
- **Translation**
  - Stateless IP/ICMP Translator (SIIT)
  - Bump in the Stack (BITSv6)
  - Bump in the API (BIA)
  - Network Address Translation -Protocol Translation (NAT-PT)
  - Transport Relay Translator (TRT)
  - SOCKS64

Dual-Stack Dominant Transition Mechanism (DSTM)

- **If an organization starts with IPv6 from the scratch and uses IPv6 in a dominant way**
  - e.g. if there are only IPv6 routers in the network and most end systems are provided with dual-stack but using IPv6 mainly
  - then we still need a mechanism to be backward compatible to IPv4
    - In order to communicate with IPv4 only hosts
  - But giving a IPv4 address to every IPv6 host in order to be able to communicate with an IPv4 only node
    - Does not solve the address space problem
  - that is where DSTM comes in
    - Described in „draft-bound-dstm-exp-04.txt
    - http://bgp.potaroo.net/ietf/all-ids/draft-bound-dstm-exp-04.txt

DSTM (cont.)

- Therefore temporary IPv4 addresses are assigned to an IPv6 host when communicating with an IPv4 only host (or vice versa)
- **DSTM client** in an dual-stack of the IPv6 domain node get this address from a **DSTM server**
- **DSTM client** uses a "**Dynamic Tunnel Interface"** (DTI) to
  - encapsulates the IPv4 packets over IPv6 infrastructure
  - IPv4-over-IPv6 tunnel
- Tunnel endpoint is a **DSTM TEP router**
  - Which connects the IPv6 domain to a conventional IPv4 domain
- An IPv4 address will only assigned when needed
  - Communication of IPv6 hosts to IPv4 only hosts
  - Native IPv4 applications on IPv6 hosts

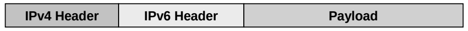
Transition Approaches

- **Dual-Stack Mechanisms**
  - Dual-Stack
  - Dual-Stack Dominant Transition Mechanism (DSTM)
- **Tunneling Mechanisms**
  - IPv4-Compatible Tunnel
  - 6to4
  - Tunnel Broker
  - 6over4
  - Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)
  - Teredo
- **Translation**
  - Stateless IP/ICMP Translator (SIIT)
  - Bump in the Stack (BITSv6)
  - Bump in the API (BIA)
  - Network Address Translation -Protocol Translation (NAT-PT)
  - Transport Relay Translator (TRT)
  - SOCKS64

General

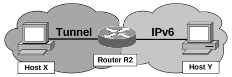
Tunneling

- Tunneling will be used in most cases during the migration process
- IPv4 routing infrastructure exists and IPv6 will use this infrastructure
- Dual stack hosts and routers can transmit IPv6 packets over an existing IPv4 topology
- IPv6 packet in an IPv4 tunnel

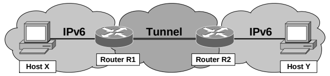


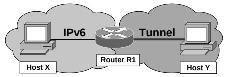
General (cont.)

- IPv6 Host to Router via IPv4
  (H to R)

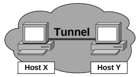


- Router to Router via IPv4
  (R to R)





- Router to IPv6 Host via IPv4
  (R to H)



- IPv6 Host to IPv6 Host via IPv4
  (H to H)

How are Tunnels configured? 1

- **Manually**
  - Usually done at routers or hosts that tunnel traffic through IPv4 only topologies by encapsulation IPv6 in IPv4
    - There are lots of experience about such tunneling gained in MBONE experiment
  - Generic term: "**Configured Tunneling**"
  - In R to R or H to R scenarios:
    - Tunnel endpoint address
      - That is the IPv4 address to which an in IPv4 encapsulated IPv6 packet for a given IPv6 destination should be sent
    - is determined from configuration information in the encapsulating node (router or host)

How are Tunnels configured? 2

- **Automatic**
  - Usually done at IPv6 hosts to tunnel traffic through IPv4 only topologies in order to reach another IPv6 host or done at an IPv6 router to reach an isolated IPv6 host via IPv4 only topology (H to H or R to H scenarios)
    - If IPv4-compatible IPv6 addresses are assigned at the destination host then tunnel endpoint IPv4 address can automatically derived from the IPv6 address
    - Tunnel endpoint address and the destination host address are the same or could be derived from the destination host address
  - Generic term: "**Automatic Tunneling**"
  - Examples
    - "IPv4-Compatible Tunneling"
    - "6to4 Tunneling"

How are Tunnels configured? 3

- **Semi-Automatic**
  - Dual stack clients connected to an IPv4 only topology want to get the right IPv4 address of a tunnel end-point on demand without manually configuring such addresses
  - A server function (called "Tunnel Broker") receives requests from dual-stack clients and tell them which IPv4 address should be used to reach the right tunnel endpoint for a certain IPv6 destination
  - In H to R scenarios:
    - Tunnel endpoint address
      - That is the IPv4 address to which an in IPv4 encapsulated IPv6 packet for a given IPv6 destination should be sent
    - is requested by the encapsulating node from the broker
  - Generic term: "**Tunnel Broker**"

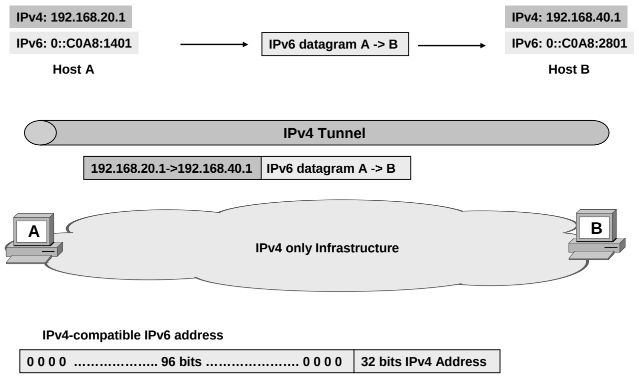
Transition Approaches

- **Dual-Stack Mechanisms**
  - Dual-Stack
  - Dual-Stack Dominant Transition Mechanism (DSTM)
- **Tunneling Mechanisms**
  - IPv4-Compatible Tunnel
  - 6to4
  - Tunnel Broker
  - 6over4
  - Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)
  - Teredo
- **Translation**
  - Stateless IP/ICMP Translator (SIIT)
  - Bump in the Stack (BITSv6)
  - Bump in the API (BIA)
  - Network Address Translation -Protocol Translation (NAT-PT)
  - Transport Relay Translator (TRT)
  - SOCKS64

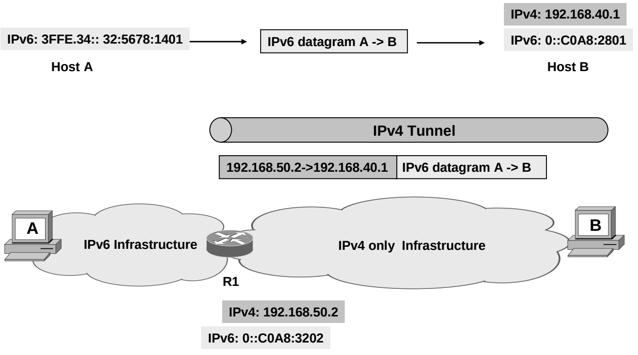
IPv4-Compatible Tunnel

- **Dual Stack at end-system**
- **If destination address is an**
  - IPv4-compatible IPv6 address (e.g.: 0::0:192.168.1.4)
- **Then**
  - An automatic tunnels (IPv6 traffic in IPv4 encapsulated) can be setup
  - The destination IPv4 address can be derived from the IPv4-compatible IPv6 address
- **But this approach does not scale**
  - Because every IPv6 node must be configured with an IPv4 address
  - Address space limitations still the problem

IPv4-Compatible Tunnel (H to H) 1



IPv4-Compatible Tunnel (R to H) 2



R1 represents B IPv6´s address in the IPv6 domain

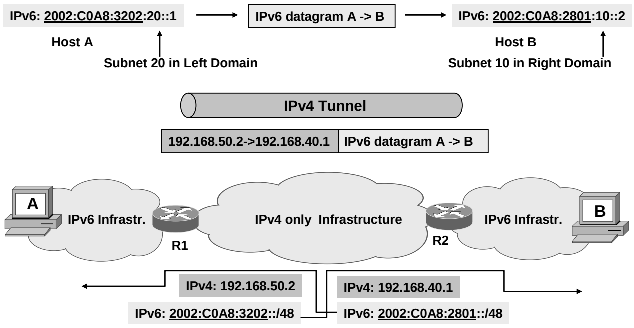
Transition Approaches

- **Dual-Stack Mechanisms**
  - Dual-Stack
  - Dual-Stack Dominant Transition Mechanism (DSTM)
- **Tunneling Mechanisms**
  - IPv4-Compatible Tunnel
  - 6to4
  - Tunnel Broker
  - 6over4
  - Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)
  - Teredo
- **Translation**
  - Stateless IP/ICMP Translator (SIIT)
  - Bump in the Stack (BITSv6)
  - Bump in the API (BIA)
  - Network Address Translation -Protocol Translation (NAT-PT)
  - Transport Relay Translator (TRT)
  - SOCKS64

6to4 Tunnel 1

- **Automatic tunnel method to connect isolated IPv6 islands over IPv4 infrastructure**
- **Every IPv6 island**
  - Receives an 48 bit IPv6 prefix which is the concatenation of 2002::/16 with the 32 bit of routers IP address on the IPv4 side of the router
    - 2002:<192.168.1.4>::/48
  - Remaining 16 bits can be used for subnetting of the IPv6 island (note: last 64 bits are the Interface-ID)
  - 2002::/16 prefix is exclusively reserved for 6to4
- **This prefix will be announced by the router**
  - Towards the other side of the tunnel and hence as IPv6 Net-ID in the other IPv6 island

6to4 Tunnel 2



R1´s IPv4 address determines the 48bit address prefix of IPv6 domain left;
R1 announces domain right into left

R2´s IPv4 address determines the 48bit address prefix of IPv6 domain right;
R2 announces domain left into right

6to4 Tunnel 3

- **Whenever a IPv6 end-systems**
  - Has to transmit an packet to a destination address which starts with such a prefix then the packet is sent to the router which announced this prefix with normal IPv6 technology
- **The receiving router**
  - Encapsulates the packet in IPv4 and forwards it to the other side of the tunnel
  - RFC 3056
- **Minimal manual configuration**
  - Neither an IPv4-compatible IPv6 address nor an configured tunnel is necessary for an IPv6 host
  - But an according IPv6 address plan must be implemented
    - Alternative: 6to4 router announces a default route ::/0
  - 6to4 mechanism is implemented only in the border-routers (so called 6to4 routers)

6to4 Relay

- **But how to reach the real IPv6 Internet in such a scenario?**
  - Other addresses than with prefix 2002::/48
- **Use a 6to4 relay router**
  - Acting as Default Router for all the 6to4 routers

Transition Approaches

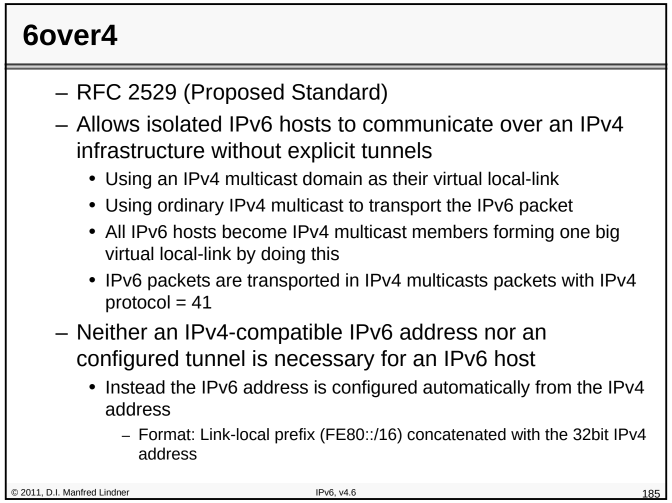
- **Dual-Stack Mechanisms**
  - Dual-Stack
  - Dual-Stack Dominant Transition Mechanism (DSTM)
- **Tunneling Mechanisms**
  - IPv4-Compatible Tunnel
  - 6to4
  - Tunnel Broker
  - 6over4
  - Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)
  - Teredo
- **Translation**
  - Stateless IP/ICMP Translator (SIIT)
  - Bump in the Stack (BITSv6)
  - Bump in the API (BIA)
  - Network Address Translation -Protocol Translation (NAT-PT)
  - Transport Relay Translator (TRT)
  - SOCKS64

Tunnel Broker

- RFC 3053 (Informational)
- In general there is some manual configuration needed to establish tunneling
- With a “Tunnel Broker” you can implement an IPv6 to IPv4 tunnel automatically
  - Clients send IPV4 HTTP request to get the information which tunnel router for a given IPv6 destination should be used
  - Tunnel broker configure tunnel information at the tunnel router
- Tunnel Broker manages activation, maintenance and termination of the tunnel
- Allows web based setup of a tunnel

Transition Approaches

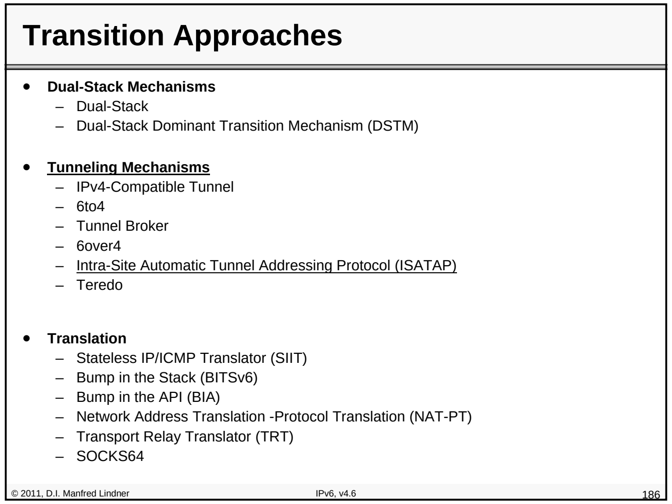
- **Dual-Stack Mechanisms**
  - Dual-Stack
  - Dual-Stack Dominant Transition Mechanism (DSTM)
- **Tunneling Mechanisms**
  - IPv4-Compatible Tunnel
  - 6to4
  - Tunnel Broker
  - 6over4
  - Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)
  - Teredo
- **Translation**
  - Stateless IP/ICMP Translator (SIIT)
  - Bump in the Stack (BITSv6)
  - Bump in the API (BIA)
  - Network Address Translation -Protocol Translation (NAT-PT)
  - Transport Relay Translator (TRT)
  - SOCKS64



6over4

- RFC 2529 (Proposed Standard)
- Allows isolated IPv6 hosts to communicate over an IPv4 infrastructure without explicit tunnels
  - Using an IPv4 multicast domain as their virtual local-link
  - Using ordinary IPv4 multicast to transport the IPv6 packet
  - All IPv6 hosts become IPv4 multicast members forming one big virtual local-link by doing this
  - IPv6 packets are transported in IPv4 multicasts packets with IPv4 protocol = 41
- Neither an IPv4-compatible IPv6 address nor an configured tunnel is necessary for an IPv6 host
  - Instead the IPv6 address is configured automatically from the IPv4 address
    - Format: Link-local prefix (FE80::/16) concatenated with the 32bit IPv4 address

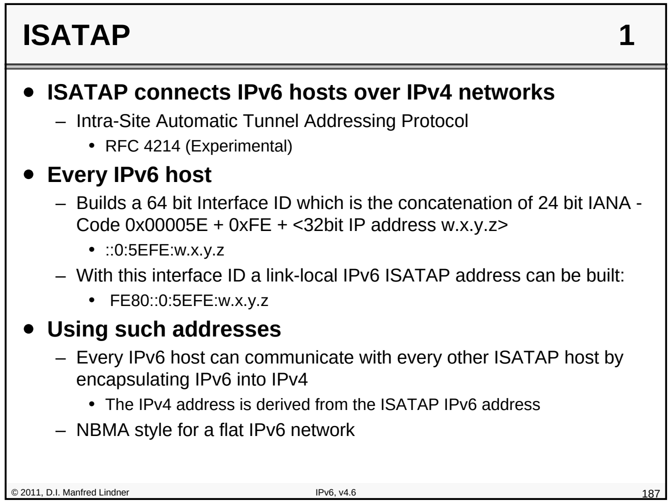
© 2011, D.I. Manfred Lindner IPv6, v4.6 185



Transition Approaches

- **Dual-Stack Mechanisms**
  - Dual-Stack
  - Dual-Stack Dominant Transition Mechanism (DSTM)
- **Tunneling Mechanisms**
  - IPv4-Compatible Tunnel
  - 6to4
  - Tunnel Broker
  - 6over4
  - Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)
  - Teredo
- **Translation**
  - Stateless IP/ICMP Translator (SIIT)
  - Bump in the Stack (BITSv6)
  - Bump in the API (BIA)
  - Network Address Translation -Protocol Translation (NAT-PT)
  - Transport Relay Translator (TRT)
  - SOCKS64

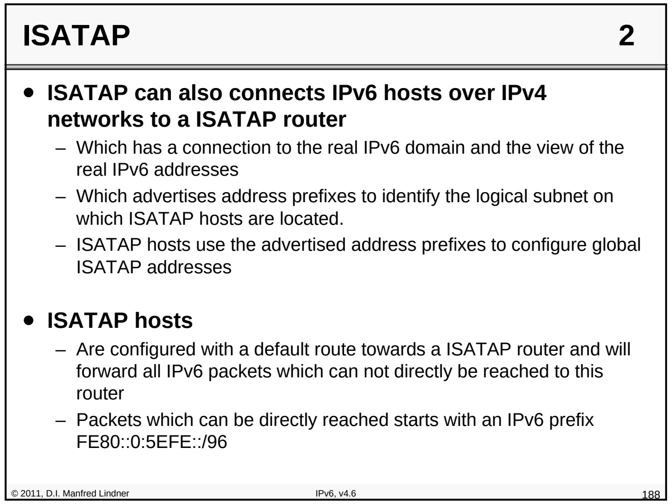
© 2011, D.I. Manfred Lindner IPv6, v4.6 186



ISATAP 1

- **ISATAP connects IPv6 hosts over IPv4 networks**
  - Intra-Site Automatic Tunnel Addressing Protocol
    - RFC 4214 (Experimental)
- **Every IPv6 host**
  - Builds a 64 bit Interface ID which is the concatenation of 24 bit IANA - Code 0x00005E + 0xFE + <32bit IP address w.x.y.z>
    - ::0:5EFE:w.x.y.z
  - With this interface ID a link-local IPv6 ISATAP address can be built:
    - FE80::0:5EFE:w.x.y.z
- **Using such addresses**
  - Every IPv6 host can communicate with every other ISATAP host by encapsulating IPv6 into IPv4
    - The IPv4 address is derived from the ISATAP IPv6 address
  - NBMA style for a flat IPv6 network

© 2011, D.I. Manfred Lindner IPv6, v4.6 187



ISATAP 2

- **ISATAP can also connects IPv6 hosts over IPv4 networks to a ISATAP router**
  - Which has a connection to the real IPv6 domain and the view of the real IPv6 addresses
  - Which advertises address prefixes to identify the logical subnet on which ISATAP hosts are located.
  - ISATAP hosts use the advertised address prefixes to configure global ISATAP addresses
- **ISATAP hosts**
  - Are configured with a default route towards a ISATAP router and will forward all IPv6 packets which can not directly be reached to this router
  - Packets which can be directly reached starts with an IPv6 prefix FE80::0:5EFE::/96

© 2011, D.I. Manfred Lindner IPv6, v4.6 188

ISATAP 3

- **How to find ISATAP router?**
  - Host asks DNS for ISATAP and get an IPv4 address of this router
  - Get the IPv6 ISATAP prefix from this router by router solicitation done via IPv4 encapsulated traffic and form a global IPv6 ISATAP address

Transition Approaches

- **Dual-Stack Mechanisms**
  - Dual-Stack
  - Dual-Stack Dominant Transition Mechanism (DSTM)
- **Tunneling Mechanisms**
  - IPv4-Compatible Tunnel
  - 6to4
  - Tunnel Broker
  - 6over4
  - Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)
  - Teredo
- **Translation**
  - Stateless IP/ICMP Translator (SIIT)
  - Bump in the Stack (BITSv6)
  - Bump in the API (BIA)
  - Network Address Translation -Protocol Translation (NAT-PT)
  - Transport Relay Translator (TRT)
  - SOCKS64

Teredo 1

- **aka as IPv4 network address translator (NAT) traversal (NAT-T) for IPv6**
  - provides address assignment and host-to-host automatic tunneling for unicast IPv6 connectivity across the IPv4 Internet when IPv6/IPv4 hosts are located behind one or multiple IPv4 NATs
- **Microsoft's solution for SOHO**
  - NAT aware transition mechanism for providing dual stack hosts behind a NAT device with global IPv6 address
  - These hosts are also reachable from the outside
- **To traverse IPv4 NATs**
  - IPv6 packets are sent as IPv4-based User Datagram Protocol (UDP) messages. UDP messages can be translated by most NATs and can traverse multiple layers of NATs.

Teredo 2

- **6to4 <->Teredo**
  - 6to4 router support is required in the edge device that is connected to the Internet. But this is not widely supported by IPv4 NATs. Even if the NAT were 6to4-enabled, 6to4 would still not work for configurations in which there are multiple NATs between a site and the Internet.
- **Teredo resolves the issues of the lack of 6to4 functionality in modern-day NATs or multi-layered NAT configurations**
  - by tunneling IPv6 packets between the hosts within the sites
- **Teredo is designed as a last resort transition technology for IPv6 connectivity**
  - If native IPv6, 6to4, or ISATAP connectivity is present between communicating nodes, Teredo is not used. As more IPv4 NATs are upgraded to support 6to4 and IPv6 connectivity become ubiquitous, Teredo will be used less and less, until eventually it is not used at all.

Teredo 3

- **Teredo client**
  - An IPv6/IPv4 node that supports a Teredo tunneling interface through which packets are tunneled to either other Teredo clients or nodes on the IPv6 Internet (through a Teredo relay)
- **Teredo server**
  - An IPv6/IPv4 node that is connected to both the IPv4 Internet and the IPv6 Internet. The role of the Teredo server is to assist in the initial configuration of Teredo clients and to facilitate the initial communication between either different Teredo clients or between Teredo clients and IPv6-only hosts.
- **Teredo relay**
  - An IPv6/IPv4 router that can forward packets between Teredo clients on the IPv4 Internet and IPv6-only hosts on the IPv6 Internet.

Transition Approaches

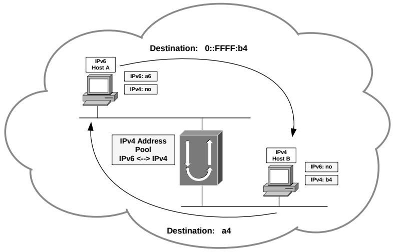
- **Dual-Stack Mechanisms**
  - Dual-Stack
  - Dual-Stack Dominant Transition Mechanism (DSTM)
- **Tunneling Mechanisms**
  - IPv4-Compatible Tunnel
  - 6to4
  - Tunnel Broker
  - 6over4
  - Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)
  - Teredo
- **Translation**
  - Stateless IP/ICMP Translator (SIIT)
  - Bump in the Stack (BITSv6)
  - Bump in the API (BIA)
  - Network Address Translation -Protocol Translation (NAT-PT)
  - Transport Relay Translator (TRT)
  - SOCKS64

Stateless IP/ICMP Translator (SIIT)

- RFC 2765 (Proposed Standard)
- Communication between IPv4 only and IPv6 only hosts - no need for two different protocol stacks (IPv4 or IPv6)
- Algorithm in explicit “Translation boxes” which translates between IPv6 and IPv4 packet header (IP and ICMP)
- Only packet header information is translated
- Translates in stateless mode
- SIIT neither translates options of IPv4 packets to IPv6 nor extension headers of IPv6 to IPv4

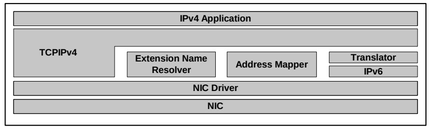
Stateless IP/ICMP Translator (SIIT) (cont.)

- Translation box assigns IPv6 site an IPv4 address
- IPv6 uses an IPv4-mapped IPv6 address (0::FFFF:a.b.c.d) to send packets to IPv4 site



Bump in the Stack (BIS)

- RFC 2767 (Informational)
- Allows IPv4-only applications on a dual stack host to communicate with IPv6-only hosts
- Additional module between IP Layer and NIC driver is necessary
- Same functionality as SIIT
- For OS where source code is not available

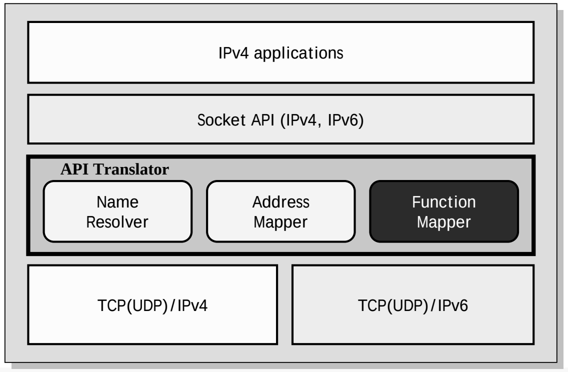


Bump in the API (BIA)

- RFC 3338 (Experimental)
- Allows IPv4-only applications on a dual stack host to communicate with IPv6-only hosts
- But the bump layer is inserted higher up, as part of the socket layer, enabling the interception of Socket API calls.
- The location of the BIA module avoids the translation of IP packets and modifications in the operating system kernel
- BIA implementations consist of three bump components
  - Name resolver
  - Address mapper
  - Function mapper
    - Intercepts IPv4 socket function calls and translates them to the equivalent IPv6 socket calls

Bump in the API (cont.)

- Architecture

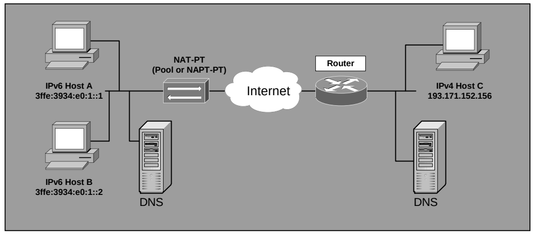


Network Address Translation Protocol Translation (NAT-PT)

- RFC 2766 (Proposed Standard)
- Provides IPv6 only node communication with IPv4 only node
- Address translation is identical to SIIT, but IPv4 addresses are assigned per TCP/UDP session (not per host)
- IPv4 addresses will be dynamically assigned to IPv6 nodes (NAT-PT possible)
- all traffic has to use the same NAT-PT router/translator
- bidirectional NAT-PT possible
- DNS queries and responses are translated by an application level gateway (DNS-ALG) in the device

Network Address Translation Protocol Translation (NAT-PT) (cont.)

- bidirectional NA(P)T-PT



Transport Relay Translator (TRT)

- RFC 3142 (Informational)
- Transport layer relays can also be extended into IPv6/IPv4 translators.
- TRT (Transport Relay Translator)
  - TRT translates between TCP/UDPv6 and TCP/UDPv4 sessions
  - Communication is initiated from the IPv6 side
  - The routing information is configured to route this prefix toward the dual-stack TRT router, which terminates the IPv6 session and initiates IPv4 communication to the final destination

SOCKS64

- RFC 3089 (Informational)
- SOCKS64 uses a dual-stacked SOCKS64 router and socksified applications to enable communication between IPv4 and IPv6 nodes
- Applications are socksified by using a special SOCKS64 library that replaces Socket and DNS APIs
- The SOCKS64 library intercepts session-initiating DNS name lookups from the end system application and responds with “fake” IP addresses mapped for the given session
- The SOCKS64 library also issues session control calls to the local SOCKS64 router

Planning for IPv6

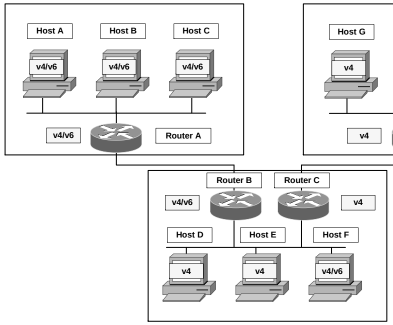
- **Identify requirements**
  - Types of applications (web, telnet, custom)
  - Scope (department, organization, Internet)
  - Types of systems
- **Select transition mechanisms**
- **Design and implement**

Planning for IPv6

- **Two choices**
  - – Dual Stack (IPv4 in parallel to IPv6)
    - + Easier transition
    - + Interoperability with traditional IPv4 Internet
    - - Management of double infrastructure
    - - Increasing network complexity
  - – IPv6 only plus NAT-PT
    - Network address Translator – Protocol Translator for interoperability with traditional IPv4 (either Intranet or Internet)
    - In principle the same complexity as management of NAT between private IPv4 domain and global IPv4 Internet
    - But are IPv6 applications ready?
      - – Temporary overcome by usage of BIS or BIA
      - – Translate IPv4 into IPv6 and vice versa at the host

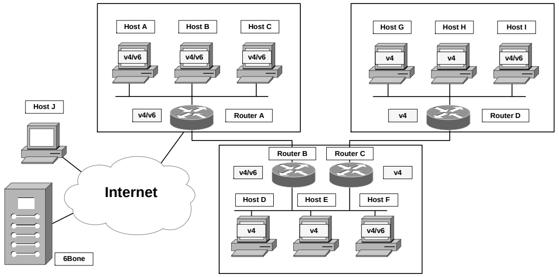
Planning for IPv6

– Intranet scenario



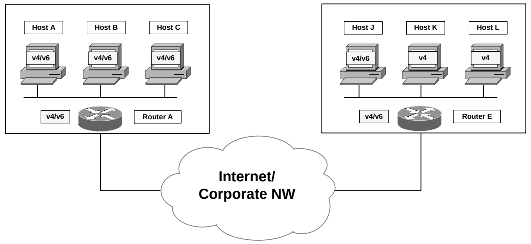
Planning for IPv6

– Intranet to Internet scenario



Planning for IPv6

– Intranet to Internet to Intranet scenario


Host A
Host B
Host C
v4/v6
v4/v6
v4/v6
v4/v6
Router A
Host J
Host K
Host L
v4/v6
v4
v4
v4/v6
Router E
Internet/
Corporate NW