Introduction to IPv6 protocol

South East Europe 6DISS Workshop
Kopaonik, Serbia & Motenegro
3rd March 2006

Athanassios Liakopoulos
(aliako@grnet.gr)

Copy …Rights

- ***This slide set is the ownership of the 6DISS project via its partners***
- ***The Powerpoint version of this material may be reused and modified only with written authorization***
- ***Using part of this material must mention 6DISS courtesy***
- ***PDF files are available from www.6diss.org***
- ***Looking for a contact ?***
  - ***Mail to : martin.potts@martel-consulting.ch***
  - ***Or helpdesk@6diss.org***

Agenda

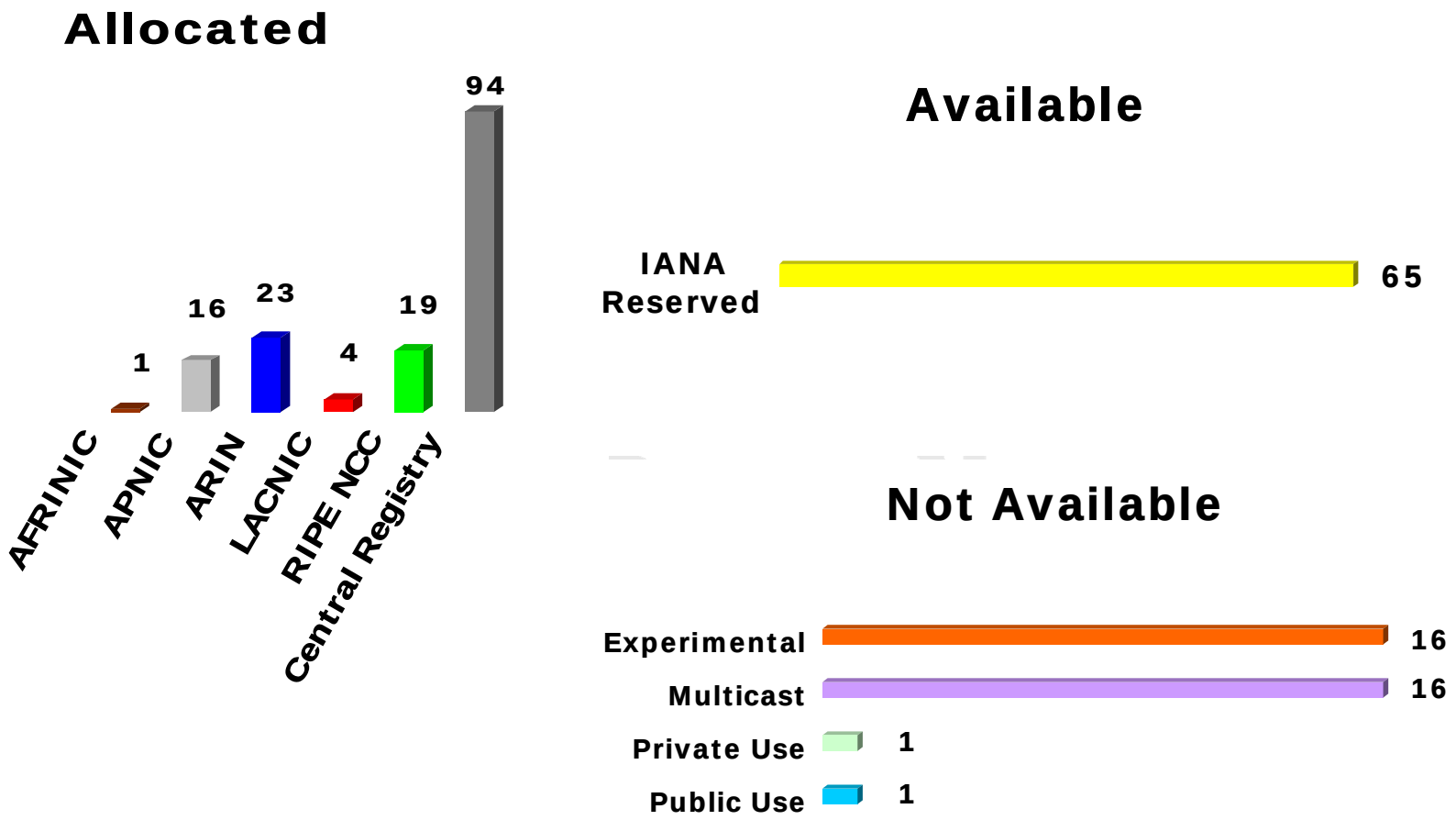
- Some history and facts …
- IPv6 Headers
- Addressing
- Associated features & protocols
- Enabling IPv6
- Transition mechanisms

Some history first …

- 1983 : Research network for ~ 100 computers
- 1992 : Commercial activity

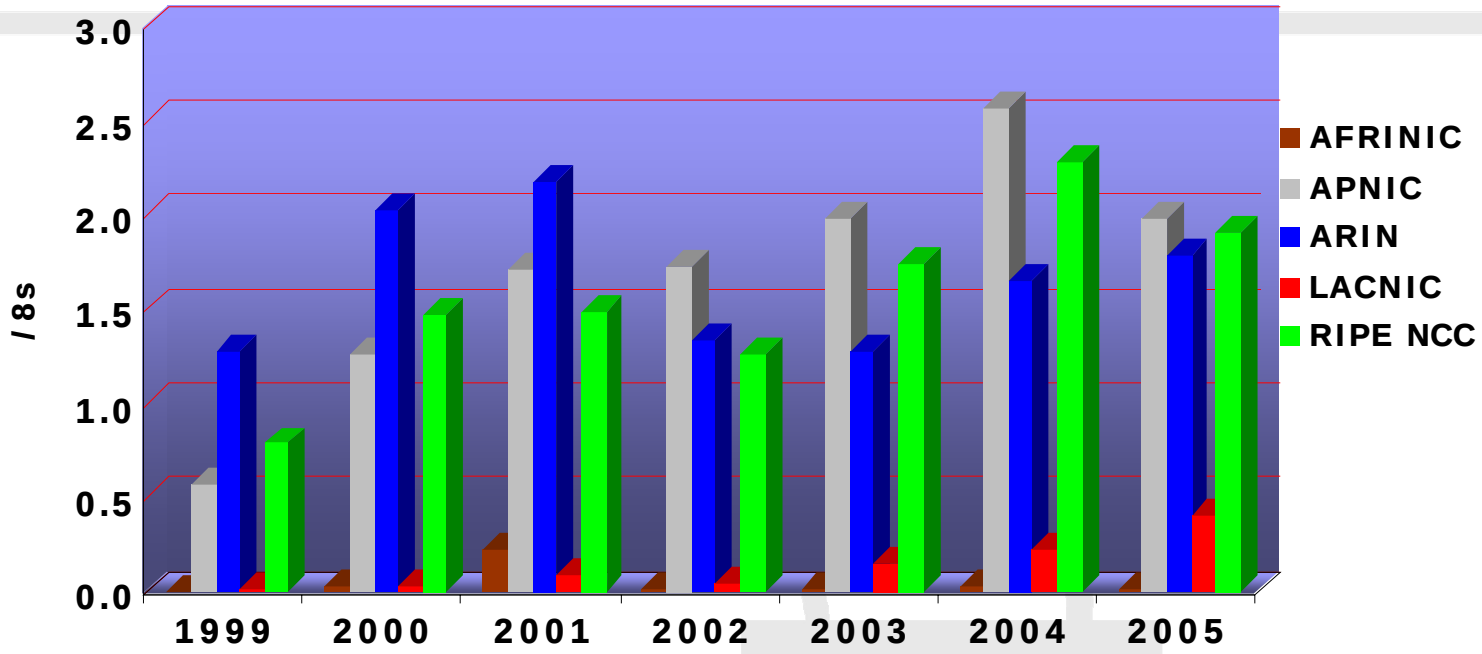
  Exponential growth
- 1993 : Exhaustion of the class B address space
- Forecast of network collapse for 1994!

IPv4 /8 Address Space Status



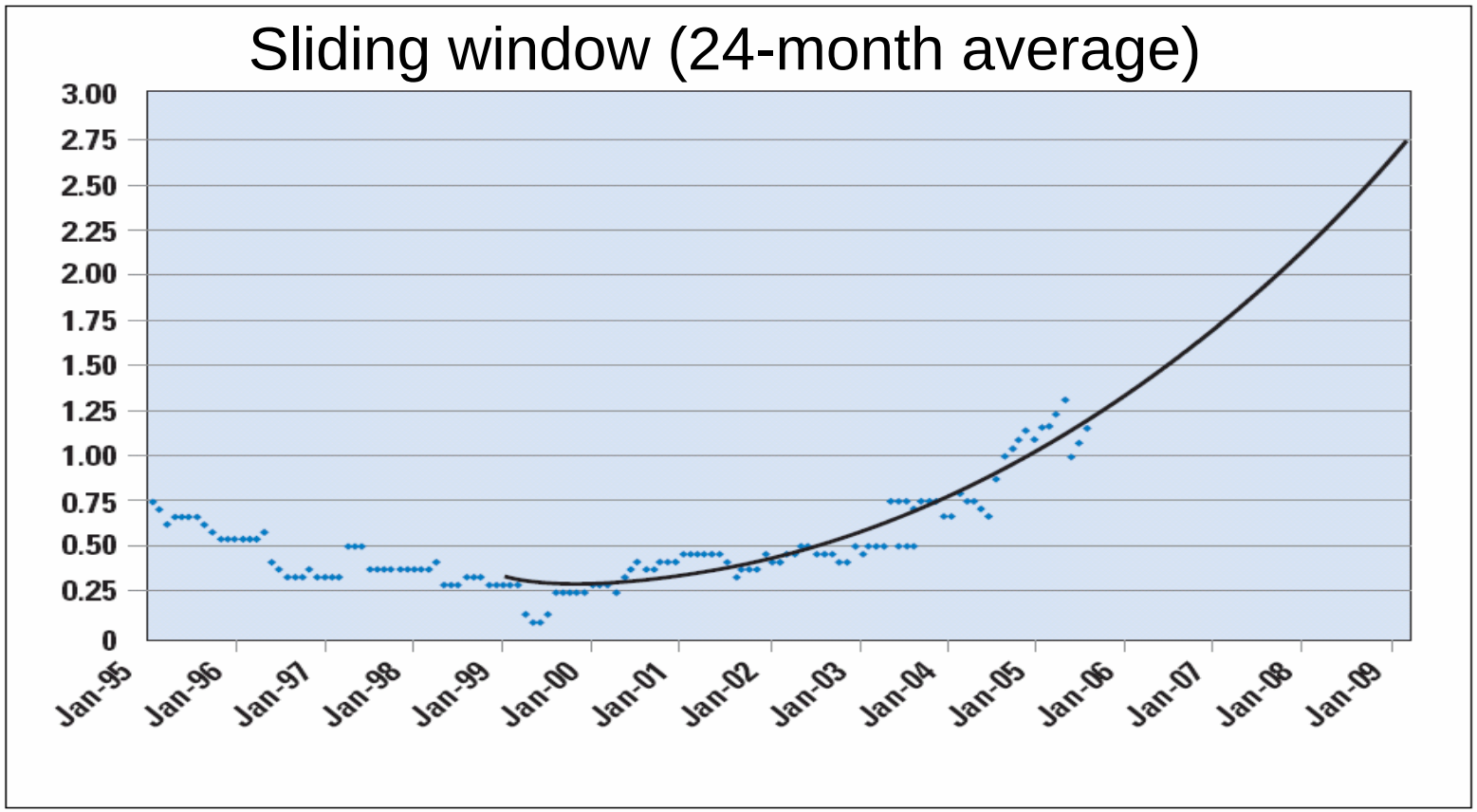
(Statistics updated in September 2005)

IPv4 Allocations from RIRs to LIRs/ISPs



- More info : http://www.nro.net/statistics/

Some future projections



source: Tony Hain, “The Internet protocol Journal”, Vol8, No3, Sept2005

Emergency Measures

- Allocate exceptionally class B addresses
- Re-use class C address space
- *Classless Internet Domain Routing (CIDR)*
  - RFC 1519
  - network address = [prefix/prefix length]
  - less address waste
  - allows aggregation

Emergency Measures (2)

- Private Addresses
  - RFC 1918 (BCP)
  - 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16
  - Allow private addressing plans
- Network Address Translation
  - RFC 1631, 2663 and 2993
  - …but NAT does not scales and breaks end-to-end connectivity

So …

- Emergency measures gave time to develop a new version of IP, named **IPv6**
- IPv6 keeps principles that have made the success of IP
- Corrects (?) what is wrong with the current version, aka IPv4
- However, are IPv4 emergency measures enough?

Agenda

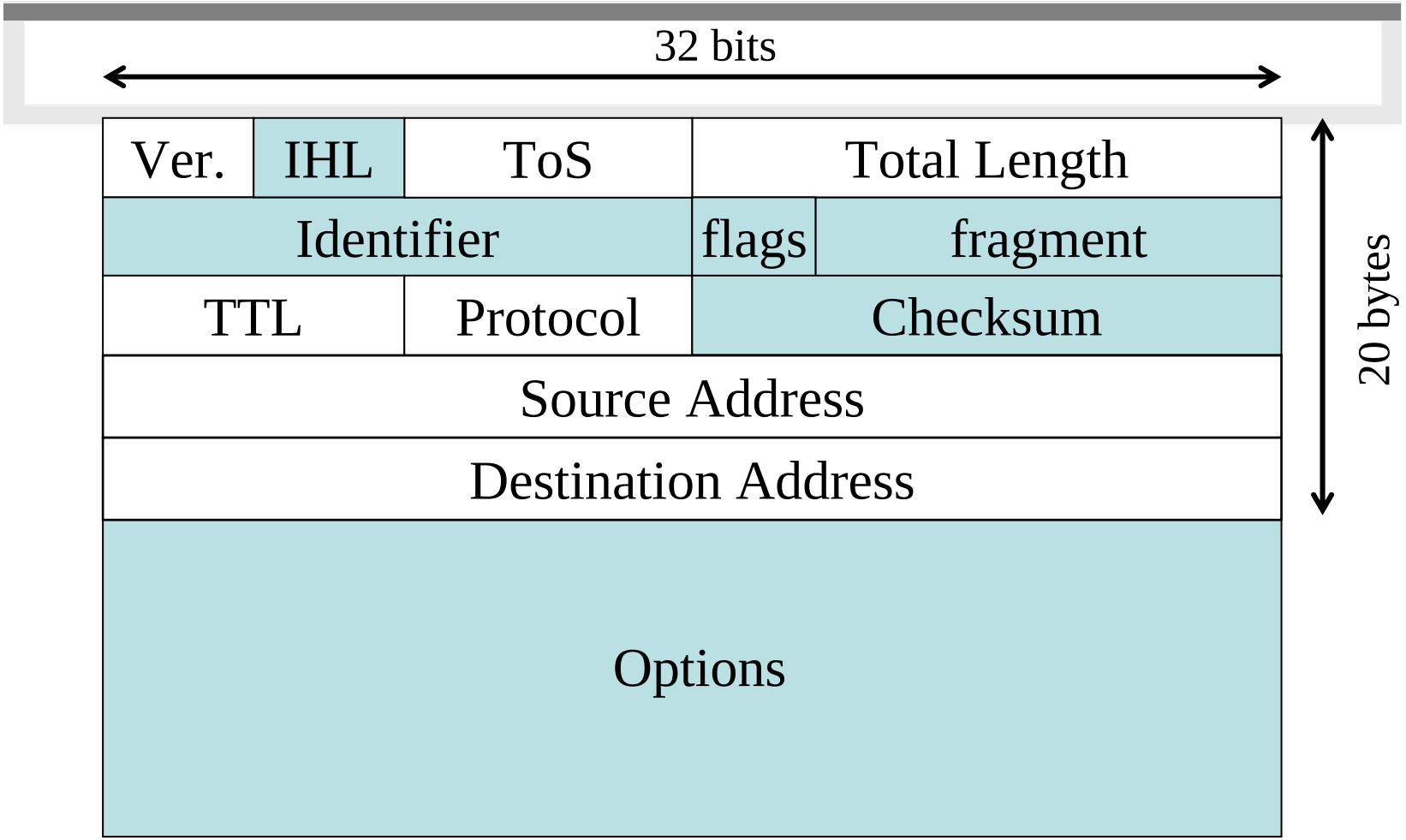
- Some history and facts …
- IPv6 Header
- Addressing
- Associated features & protocols
- Enabling IPv6
- Transition mechanisms

IPv6 Header

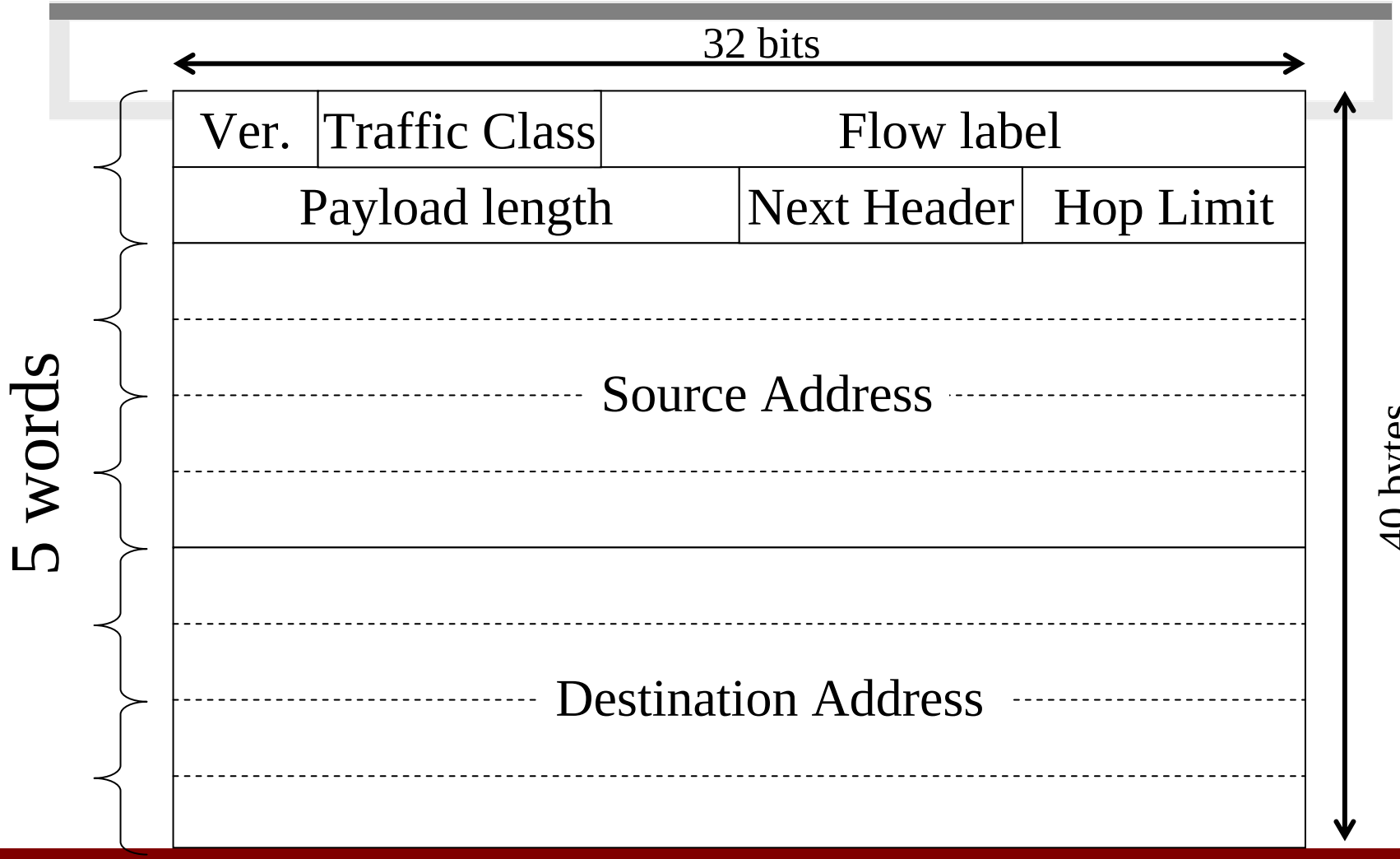
- The IPv6 header is redesigned.
- Minimize header overhead and reduce the header process for the majority of the packets.
- Less essential and optional fields are moved to extension headers

IPv6 and IPv4 headers are not *interoperable!*

IPv4 Header


32 bits
Ver.
IHL
ToS
Total Length
Identifier
flags
fragment
TTL
Protocol
Checksum
Source Address
Destination Address
Options
20 bytes

IPv6 header


32 bits
Ver.
Traffic Class
Flow label
Payload length
Next Header
Hop Limit
Source Address
Destination Address
5 words
40 bytes

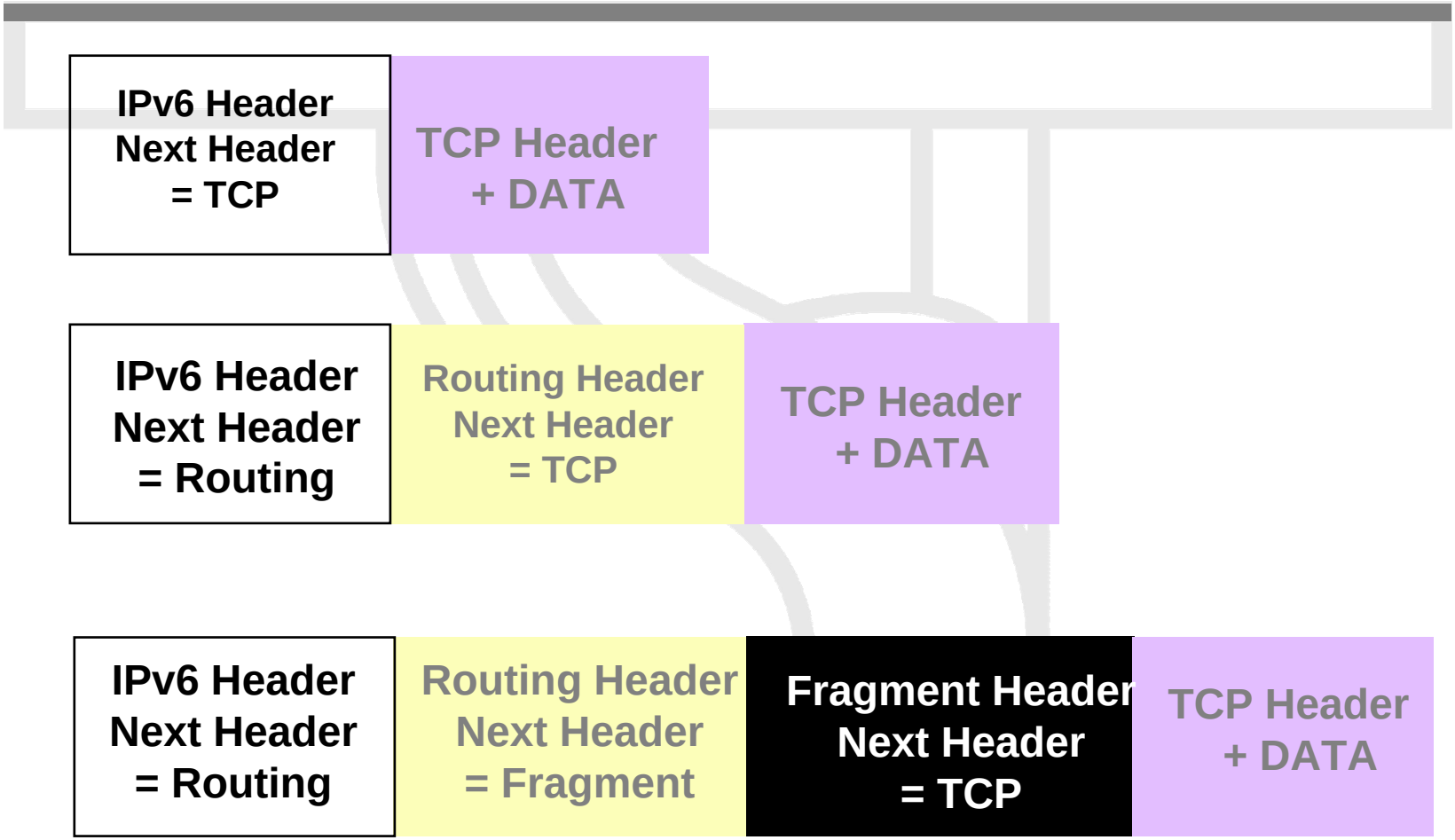
IPv6 header fields

- Version
- Traffic Class
  - An *8-bit* field used to distinguish packets from *different classes or priorities*.
  - Provides the *same* functionality as the ***type of service*** field in the IPv4 header.

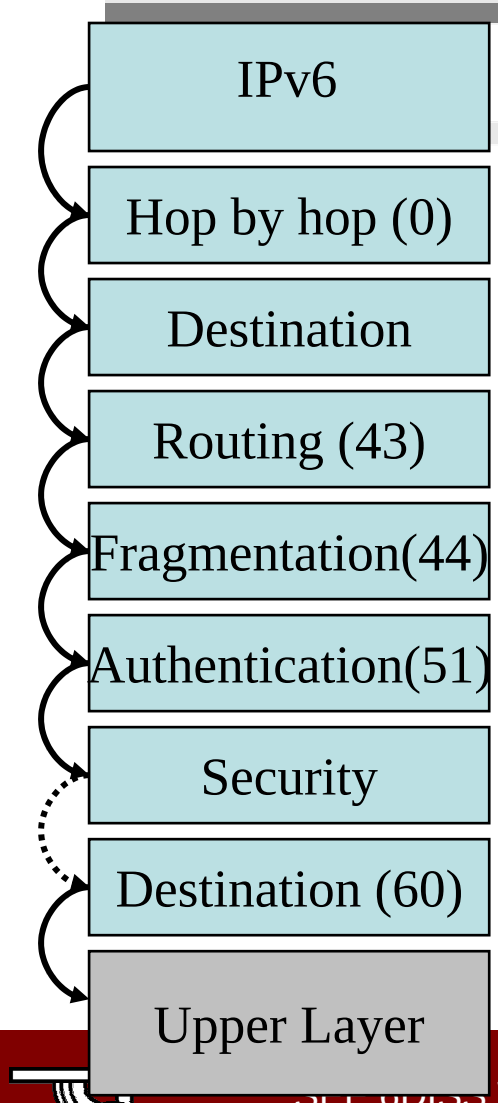
IPv6 header fields (2)

- Flow label (RFC 3697)
  - A *20-bit* field, selected by the source and never modified in the network.
  - Fragmentation or encryption is not anymore problem, as in IPv4.
- Payload length
  - Use Jumbogram (payload = 0)
- Hop limit
- Next header

IPv6 extension headers



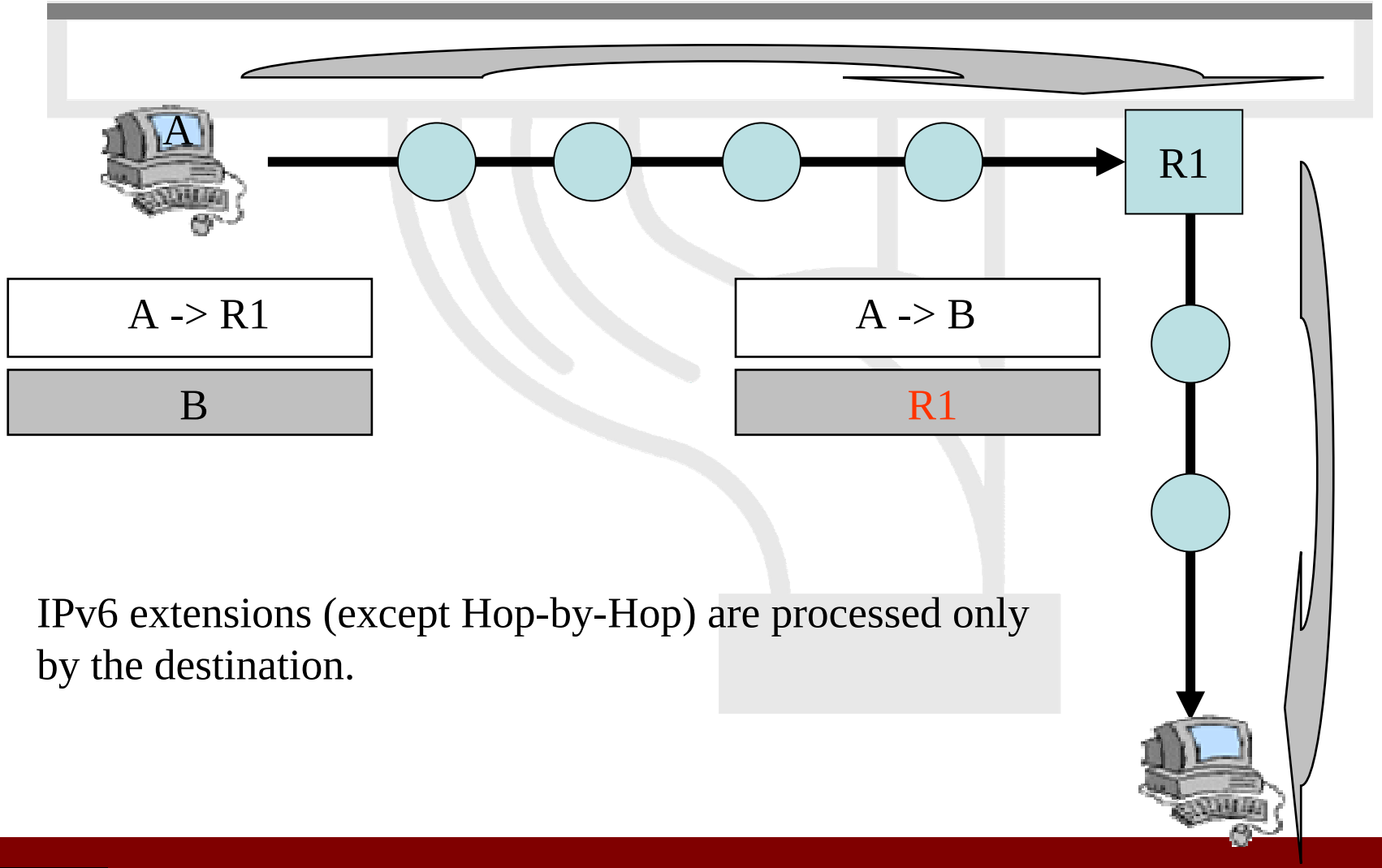
IPv6 extension headers: order is important



RFC 2460

- Hop by hop (0): Processed by every router
- Destination: Processed by routers listed in Routing extension
- Routing (43): List of routers to cross
- Fragmentation(44): Processed by the destination
- Authentication(51): After reassembling the packet
- Security: Cipher the content of the remaining information
- Destination (60): Processed **only** by the destination

IPv6 extension headers: processing



IPv6 extensions (except Hop-by-Hop) are processed only by the destination.

Agenda

- Some history and facts …
- IPv6 Headers
- Addressing
- Associated features & protocols
- Enabling IPv6
- Transition mechanisms

Addressing scheme

- IPv6 addressing (RFC 3513)
  - 128-bit long addresses
    - 340.282.366.920.938.463.463.374.607.431.768.211.456
  - Hexadecimal representation
  - Interfaces have several IPv6 addresses
  - CIDR principles [address prefix / prefix length]
  - Aggregation reduces routing table size
- IPv6 address format (RFC 3587)
  - Allow hierarchy

Textual Address Format

- Base format (16-byte)

2001:0660:3003:0001:0000:0000:6543:210F

- Compact Format:

2001:660:3003:1::6543:210F

- Litteral representation
  - [2001:660:3003:2:a00:20ff:fe18:964c]

IPv6 Addresses

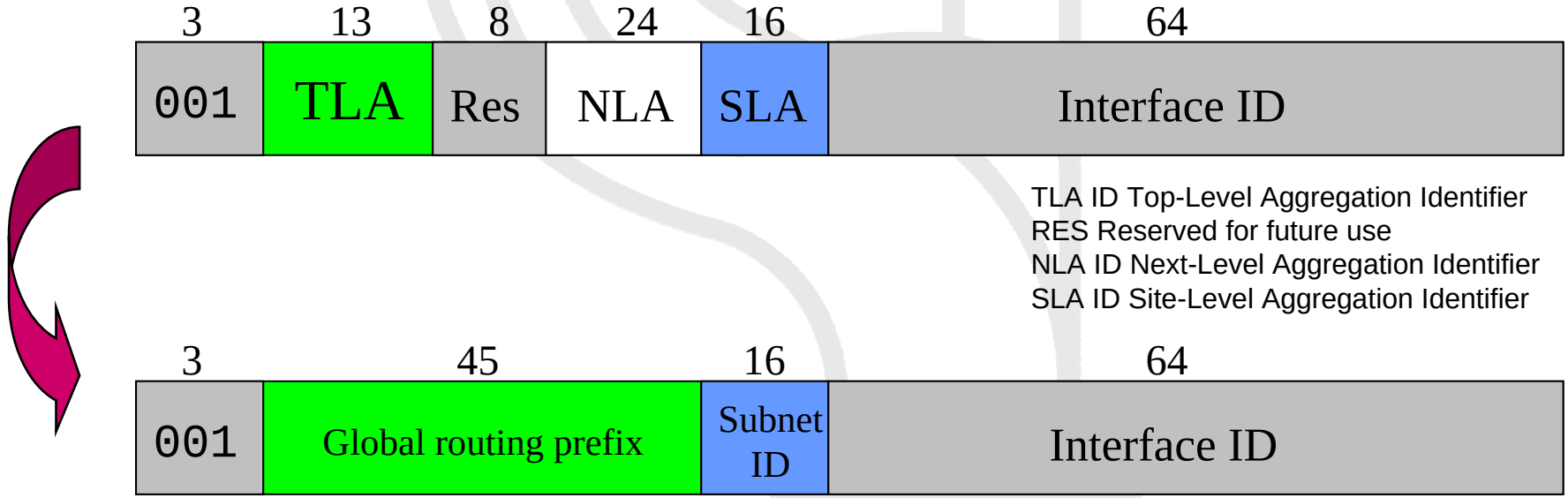
- Loopback ::1
- **Link local** FE80:….
- Site local FEC0:…
- Global
  - Official: 2001:…
  - 6bone: 3FFE:…
- **IPv4 mapped**
- **6to4** 2002::…
- Multicast FF…

- Unicast
- Multicast
- **Anycast**

specific to IPv4/IPv6 integration

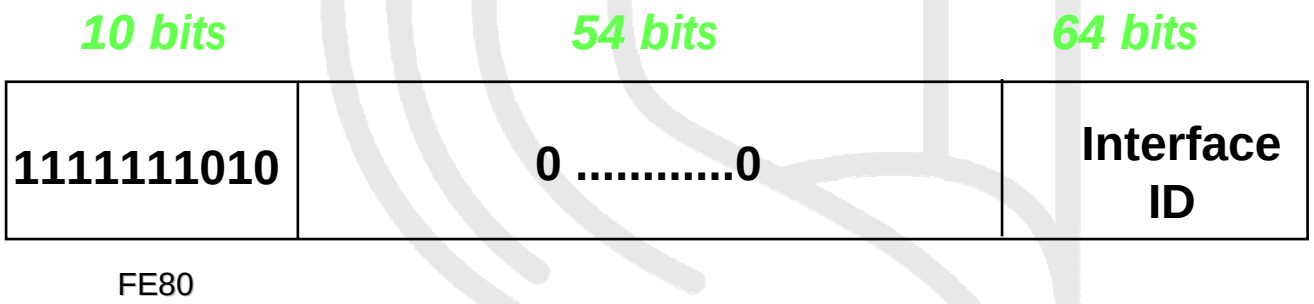
Global Unicast address format (RFC 3587)

(obsoletes RFC 2374)

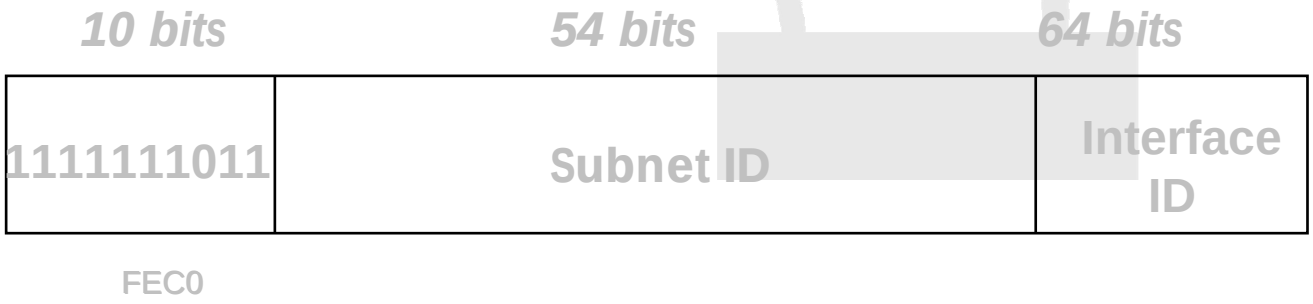


Local Addresses

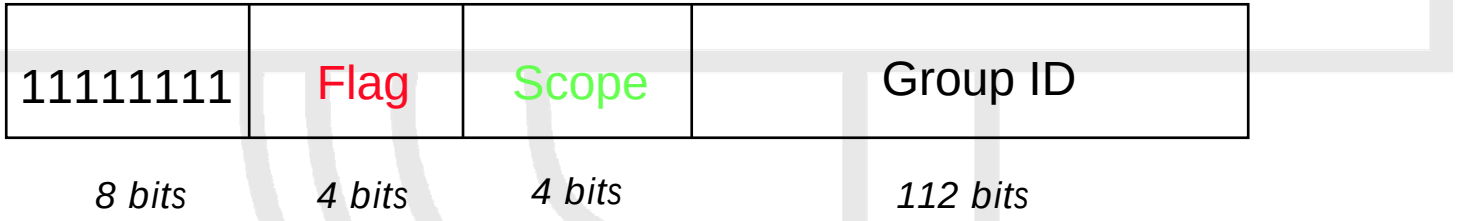
Link-local



Site-local (in the process of being deprecated)



Multicast Addresses



Flag bits: 0 R P T

T = 0 *permanent addresses (managed by IANA)*

T = 1 *transient multicast addresses*

- P = 1 *derived from unicast prefix (RFC3306)*
  - R = 1 *embedded RP addresses (RFC 3956)*

Scope

0 : Reserved
1 : Interface-local
2 : Link-local
3 : Subnet-local
4 : Admin-local
5 : Site-local
8 : Organization-local
E : Global
F : Reserved

Anycast Addresses (RFC 3513)

- «Anycast addresses allow a packet to be **routed to one of a number** of different nodes all responding to the same address »
- «Anycast addresses are taken from the unicast address spaces (of any scope) and are not syntactically distinguishable from unicast addresses« it may be assigned to an IPv6 router only »
- Anycast address …
  - … can not be a used as a source address of an IPv6 packet
  - … must be assigned only to routers
- Reserved anycast addresses are defined in RFC 2526

Agenda

- Some history and facts …
- IPv6 Headers
- Addressing
- Associated features & protocols
- Enabling IPv6
- Transition mechanisms

Associated features & protocols

- Neighbor Discovery (ND) (RFC 2461 DS)
- Auto-configuration :
  - Stateless Address Auto-configuration (RFC 2462 DS)
  - DHCPv6: Dynamic Host Configuration Protocol for IPv6 (RFC 3315 PS)
  - Path MTU discovery (pMTU) (RFC 1981 PS)

Associated features & protocols

- MLD (Multicast Listener Discovery) (RFC 2710)
  - Multicast group management over an IPv6 link
  - Based on IGMPv2
  - MLDv2 (equivalent to IGMPv3 in IPv4)
- ICMPv6 (RFC 2463 DS) "Super" Protocol that :
  - Covers ICMP (v4) features (Error control, Administration, …)
  - Transports ND messages
  - Transports MLD messages (Queries, Reports, …)

Neighbor Discovery

- IPv6 nodes which share the same physical medium (link) use Neighbor Discovery (NDP) to:
  - discover their mutual presence
  - determine link-layer addresses of their neighbors
  - find routers
  - maintain neighbors' reachability information (NUD)
  - not directly applicable to NBMA (Non Broadcast Multi Access) networks ➔ ND uses multicast for certain services.

Neighbor Discovery (2)

- Protocol features:
  - Router discovery
  - Prefix(es) discovery
  - Parameters discovery (link MTU, Max Hop Limit, ...)
  - Address auto-configuration
  - Address resolution
  - Next Hop determination
  - Neighbor Unreachability Detection
  - Duplicate Address Detection
  - Redirect

Neighbor Discovery (4)

- ND specifies 5 types of ICMP packets :
  - **Router Advertisement** (RA) :
    - periodic advertisement (of the availability of a router) which contains:
      - » list of prefixes used on the link (autoconf)
      - » a possible value for Max Hop Limit (TTL of IPv4)
      - » value of MTU
  - **Router Solicitation** (RS) :
    - the host needs RA immediately (at boot time)

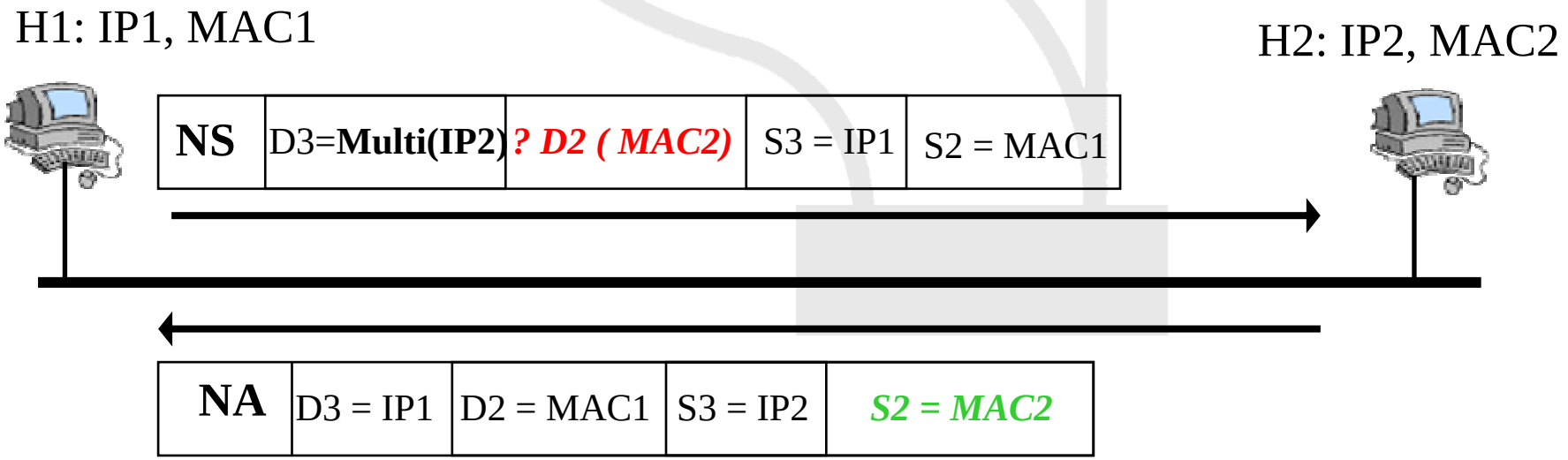
Neighbor Discovery (5)

- **Neighbor Solicitation** (NS):
  - to determine the link-layer @ of a neighbor
  - or to check its impeachability
  - also used to detect duplicate addresses (DAD)
- **Neighbor Advertisement** (NA):
  - answer to a NS packet
  - to advertise the change of physical address
- **Redirect** :
  - Used by a router to inform a host of a better route to a given destination

Address Resolution (2)
IPv6 with Neighbor Discovery

At boot time, every IPv6 node has to join 2 special multicast groups for each network interface:

- All-nodes multicast group: **`ff02::1`**
- Solicited-node multicast group: **`ff02:1:ffxx:xxxx`** (derived from the lower 24 bits of the node's address)



Address Resolution (3)
Solicited Multicast Address

- **Concatenation** of the prefix `FF02::1:FF00:0/104` with the last 24 bits of the IPv6 address

Example:

- **Dst IPv6** @: `2001:0660:010a:4002:4421:21FF:FE24:87c1`

- **Sol. Mcast** @: `FF02:0000:0000:0000:0000:0001:FF24:87c1`

- **ethernet:** `FF-FF-FF-24-87-c1`

Stateless Autoconfiguration

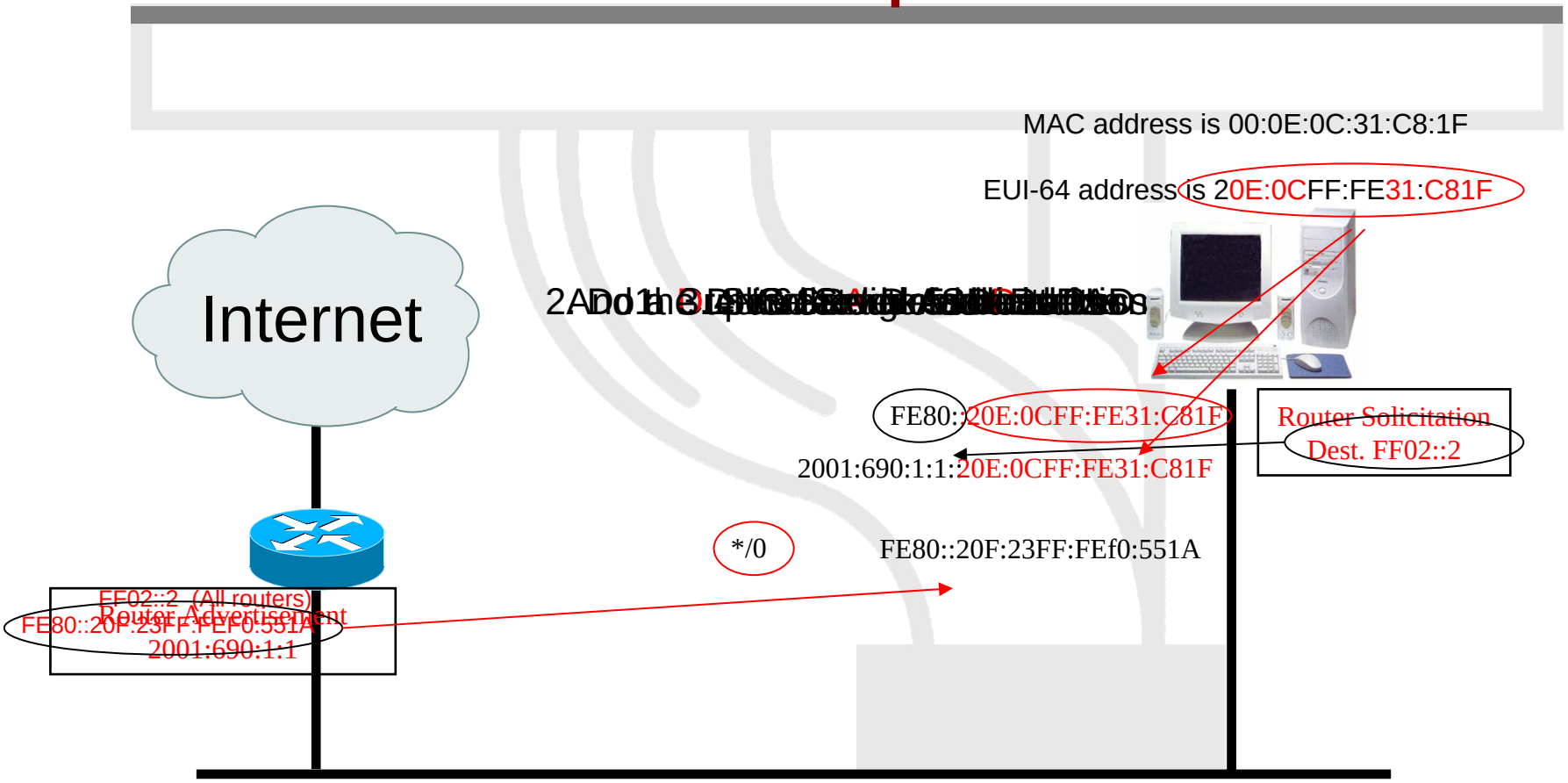
- Described in RFC 2462
- Objective: IPv6 hosts should be *Plug & Play*

- Uses some of the Neighbor Discovery ICMPv6 messages
  - Route advertisement, route solicitation
  - IPv6 does not support ARP and there are no broadcast addresses.

Stateless Autoconfiguration: Procedure

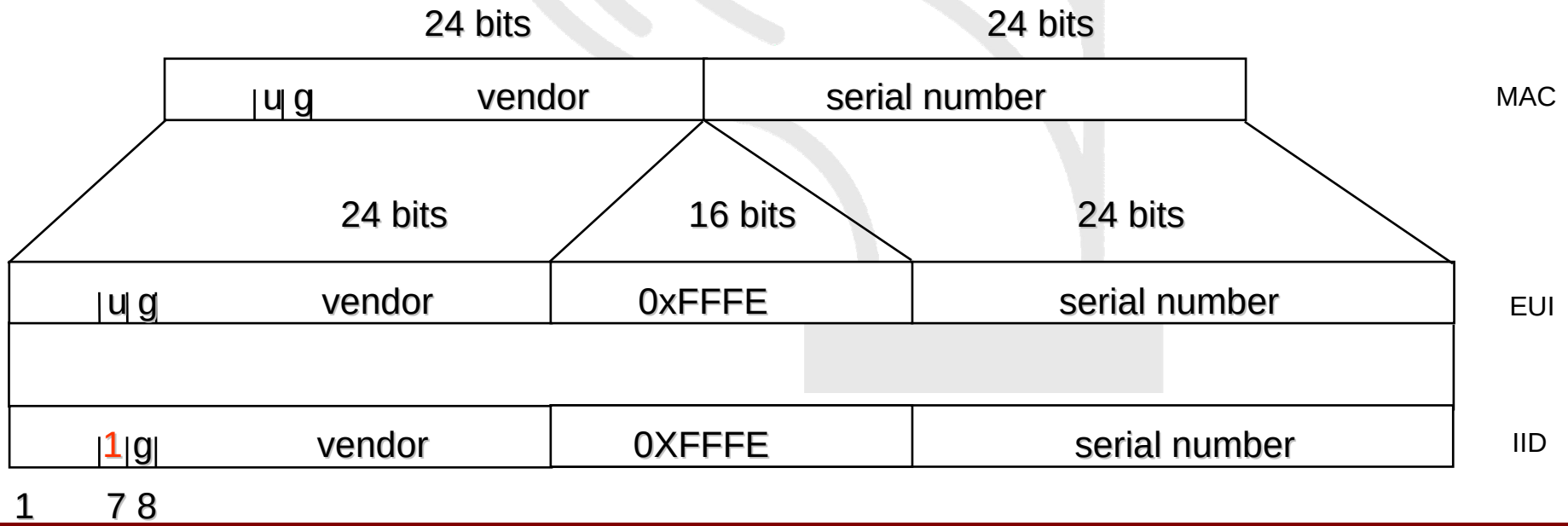
- When booting, a host asks/seeks for some network parameters:
  - IPv6 prefix (-es), default router address (-es), hop limit, (link local) MTU, prefix validity time, etc
- Hosts listen Router Advertisements (RA) messages
  - RA are periodically transmitted by routers or when prompt by hosts
  - If a RA doesn't carry any prefix, the hosts may only configure the default gateway address.
- A host to creates …
  - a link local address (fe80::/10)
  - a global IPv6 address using:
    - its interface identifier (EUI-64 address)
    - link prefix obtained via Router Advertisement

Stateless Autoconfiguration example

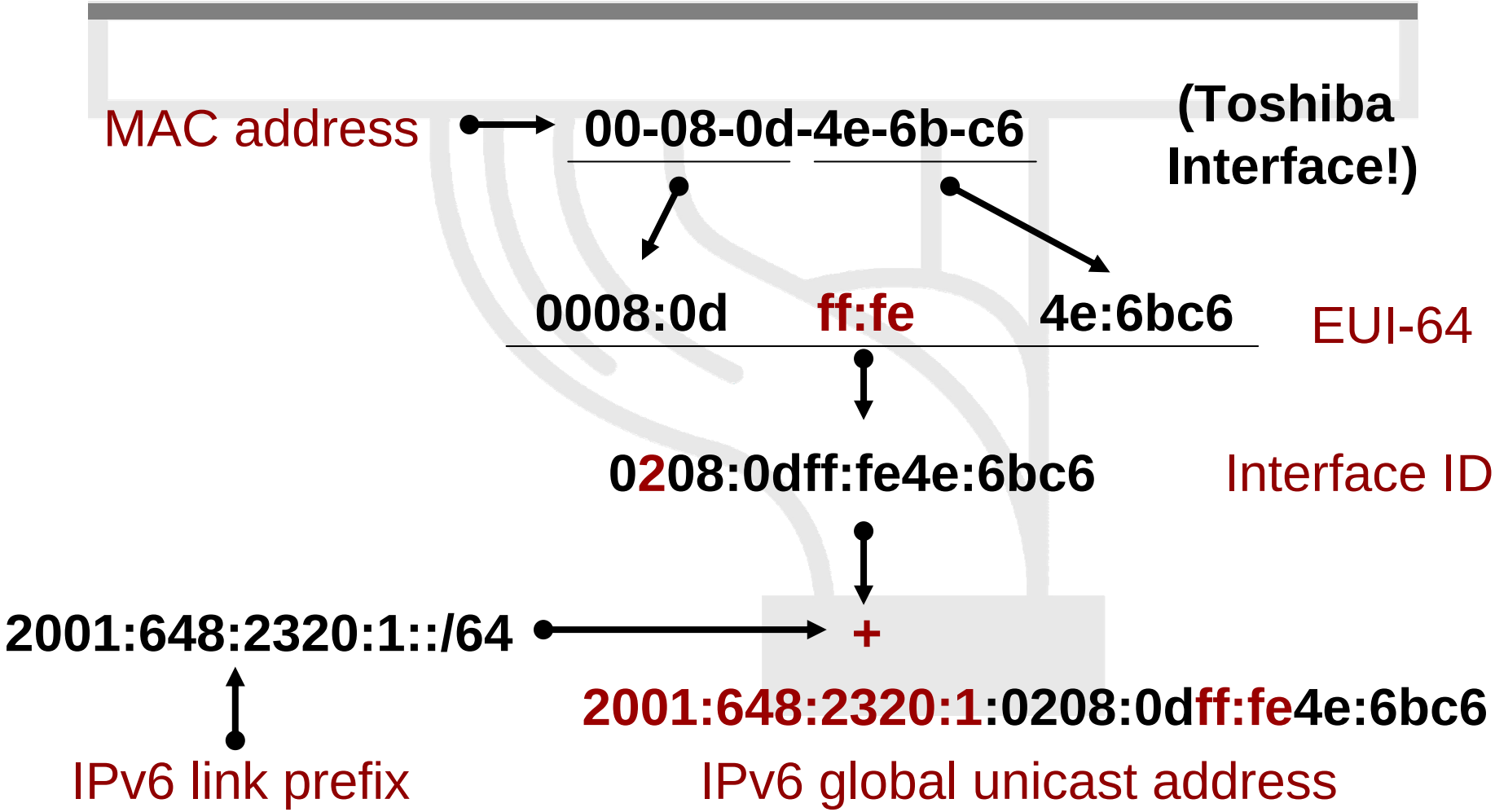


Interface Identifier

- 64 bits to be compatible with IEEE 1394 (FireWire)
- Eases auto-configuration
- IEEE defines the mechanism to create an EUI-64 from IEEE 802 MAC addresses (Ethernet, FDDI)



Interface Identifier: Example



Interface Identifier: Privacy issues

- IEEE 24 bit OUI identify hardware
  (http://standards.ieee.org/regauth/oui/oui.txt)
- Interface ID can be used to trace a user:
  - The prefix changes, but the interface ID remains the same!
- Privacy extensions (RFC 3041)
  - Possibility to change Interface ID
  - If local storage, use MD5 algorithm. Otherwise, draw a random number.
  - Creates security problem?

Stateless Autoconfiguration: Best practices

- Only routers have to be manually configured
  - Ongoing work on prefix delegation

    (http://www.ietf.org/rfc/rfc3633.txt)
- Hosts get automatically an IPv6 address
  - BUT it isn't automatically registered in the DNS!
- Servers should be manually configured!

Stateful autoconfiguration

- Dynamic Host Configuration Protocol for IPv6 (DHCPv6)
  - Defined in RFC 3315
  - Stateful counterpart to IPv6 Stateless Address Autoconfiguration.
- DHCPv6 is used when:
  - no router is found
  - Or if Router advertisement message enable use of DHCP

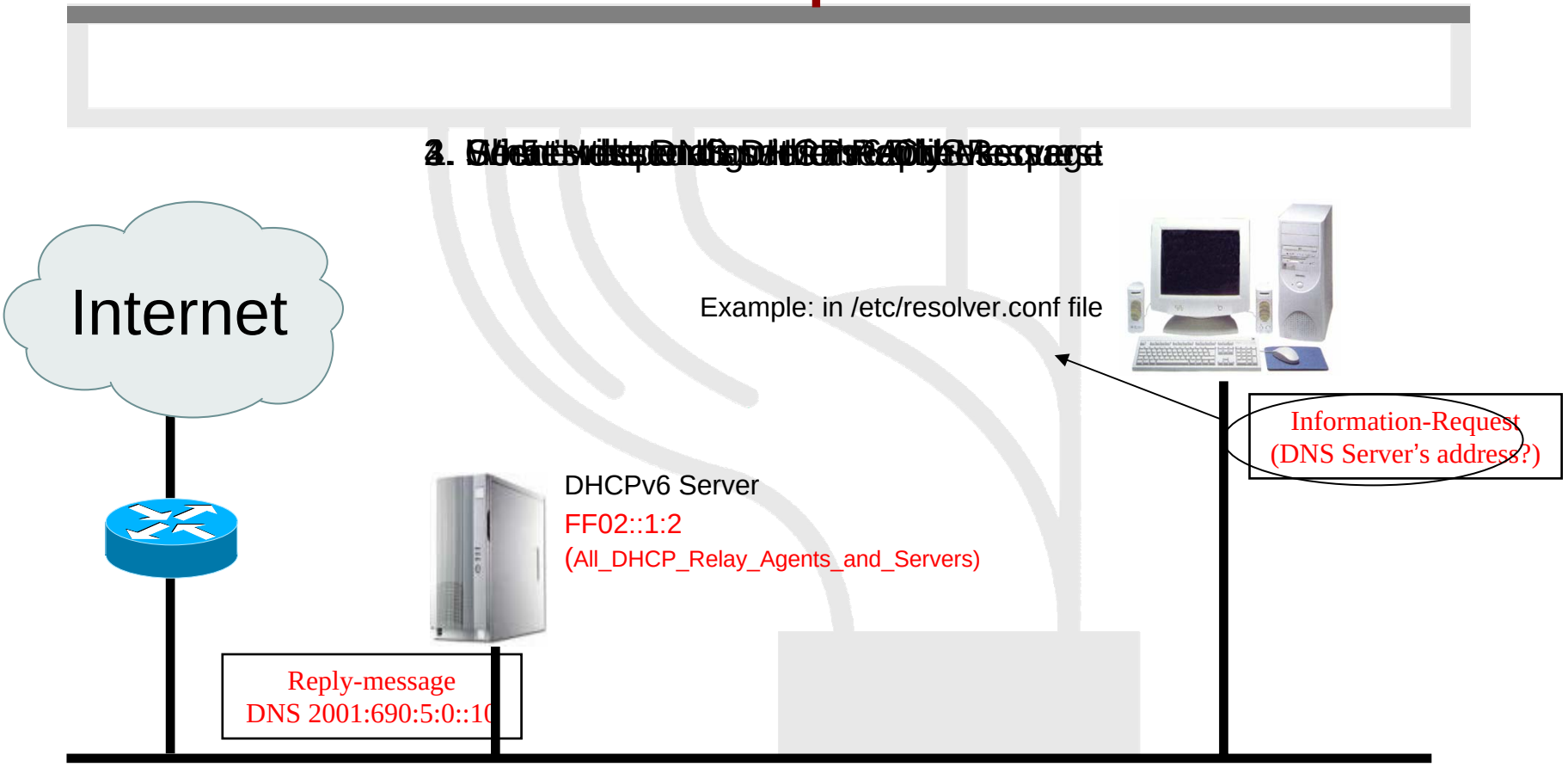
Stateful autoconfiguration

- DHCPv6 works in a client-server model
  - Server
    - Responds to requests from clients
    - Optionally provides the client with:
      - IPv6 addresses
      - Other configuration parameters (DNS servers…)
    - Is listening on multicast addresses:
      - All_DHCP_Relay_Agents_and_Servers (FF02::1:2)
      - All_DHCP_Servers (FF05::1:3)
    - Memorize client's state
    - Provide means for securing access control to network resources

Stateful autoconfiguration

- Client
  - initiates requests on a link to obtain configuration parameters
  - use its link local address to connect the server
  - Send requests to FF02::1:2 multicast address (All_DHCP_Relay_Agents_and_Servers)
- Relay agent
  - node that acts as an intermediary to deliver DHCP messages between clients and servers
  - is on the same link as the client
  - Is listening on multicast addresses:
    - All_DHCP_Relay_Agents_and_Servers (FF02::1:2)

Stateful Autoconfiguration example



Path MTU discovery (RFC 1981)

- Derived from RFC 1191, (IPv4 version of the protocol)
- **Path** : set of links followed by an IPv6 packet between source and destination
- **link MTU** : maximum packet length (bytes) that can be transmitted on a given link without fragmentation
- **Path MTU** (or pMTU) = min { link MTUs } for a given path
- Path MTU Discovery = automatic pMTU discovery for a given path

Path MTU discovery (2)

- Protocol operation
  - makes assumption that pMTU = link MTU to reach a neighbor (first hop)
  - if there is an intermediate router such that link MTU < pMTU ➔ it sends an ICMPv6 message: "Packet size Too Large"
  - source reduces pMTU by using information found in the ICMPv6 message

  => Intermediate equipments aren't allowed to perform packet fragmentation

Agenda

- Some history and facts …
- IPv6 Headers
- Addressing
- Associated features & protocols
- Enabling IPv6
- Transition mechanisms

IPv6 Support: Windows

- WinXP
  - SP0: Autoconfiguration, tunnels, ISATAP, etc. IPv6 has explicitly to be activated!
  - SP1: GUI installation, **`netsh`** command line interface
  - SP2: Teredo, firewall, and other additions
- Win2000
  - Only developer edition available
- Windows 95/98/ME
  - No official support

IPv6 install: Windows

- WinXP
  - Execute "`ipv6 install`" at a command prompt (SP0)
  - Add '*Microsoft IPv6 Developer Edition*' component as a new protocol in the Network Connections Control Panel pane (SP1)
  - Add 'Microsoft TCP/IP version 6' as a new protocol in the Network Connections Control Panel pane (SP2)

IPv6 commands: WinXP

- Command line interface (`netsh`):

```
c:\>netsh interface ipv6
```

- Well known (IPv4/6) commands

```
ipconfig, netstat, ping6, tracert6, pathping
```

- IPv6 depreciated command:

```
c:\>ipv6 ?
    usage: ipv6 [-p] [-v] if [ifindex]
     ipv6 [-p] ifcr v6v4 v4src v4dst [nd] [pmld]
     ipv6 [-p] ifcr 6over4 v4src
    …
```

IPv6 commands: WinXP

- Set an IPv6 address (`netsh`):

```
set address [interface=]<string>
  [address=]<IPv6 address>
      [[type=]unicast|anycast]
      [[validlifetime=]<integer>|infinite]
      [[preferredlifetime=]<integer>|infinite]
      [[store=]active|persistent]
```

- Delete an IPv6 address (`netsh`):

```
delete address [interface=]<string>
    [address=]<IPv6 address>
    [[store=]active|persistent]
```

IPv6 commands: WinXP

- Add an route:

```
add route [prefix=]<IPv6 address>/<integer>
[interface=]<string>
    [[nexthop=]<IPv6 address>]
[[siteprefixlength=]<integer>]
    [[metric=]<integer>] [[publish=]no|age|yes]
    [[validlifetime=]<integer>|infinite]
    [[preferredlifetime=]<integer>|infinite]
    [[store=]active|persistent]
```

- Show route:

```
show routes [[level=]normal|verbose]
    [[store=]active|persistent]
```

IPv6 commands: WinXP

- Reset configuration (`netsh`):

```
renew [[interface=]<string>]
```

- Show DNS server (`netsh`):

```
show dns [[interface=]string]
```

IPv6 Support: Linux distributions

- Redhat (6.2+), Fedora 1&2, SuSE (7.3+), Debian (2.2+), Mandrake (8.0+), Scientific Linux (3.0+), *BSD, etc
  - Look for IPv6 support at kernel!
- USAGI
  - Collaboration between WIDE, KAME and TAHI in order to improve kernel

IPv6 install: Linux

- Load IPv6 module

```
# modprobe ipv6
```

- Search loaded module

```
# lsmod –w “ipv6”
```

- Check if IPv6 is installed

```
# test –f /proc/net/if_inet6 && echo “OK”
```

- Well known (IPv4/6) commands

```
ifconfig, netstat, ping6, traceroute6
```

- Route management

```
route –A inet6
```

Agenda

- Some history and facts …
- IPv6 Header
- Addressing
- Associated features & protocols
- Enabling IPv6
- Transition mechanisms

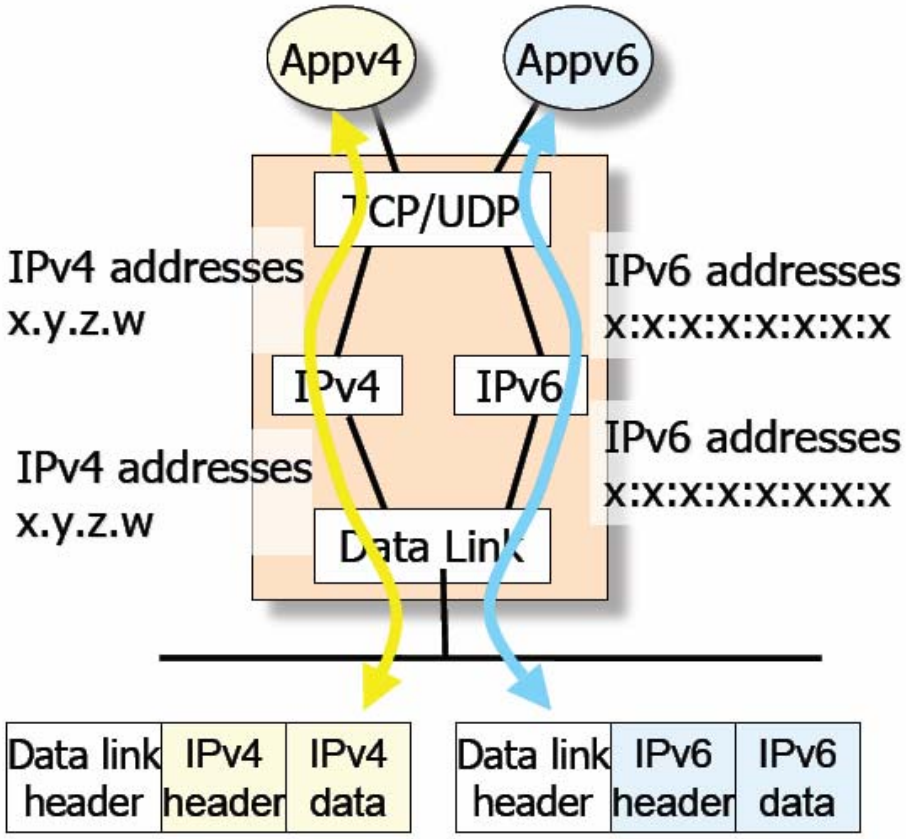
Transition to IPv6

- Transition to IPv6 (and coexistence with IPv4) will last for years!
  - The primary objective is the smooth migration to the IPv6 protocol! In other words, how IPv4-, IPv4/6- and IPv6-enabled systems communicate?
- There are three transition approaches
  - Dual-stack: IPv6 and IPv4 co-exist in the network devices, e.g. router or end-system
  - Tunneling: IPv6 traffic is encapsulated into IPv6 packets.
  - Translation: Rewrite packet headers!

Dual Stack

- End-systems and routers support both protocols
  - Routers has to support two routing tables
  - Security has to be applied into both protocols
- Applications chooses which protocol to use
  - IPv6 is usually selected first.
  - If there is an “IPv6 problem”, IPv4 protocols is used after a while!

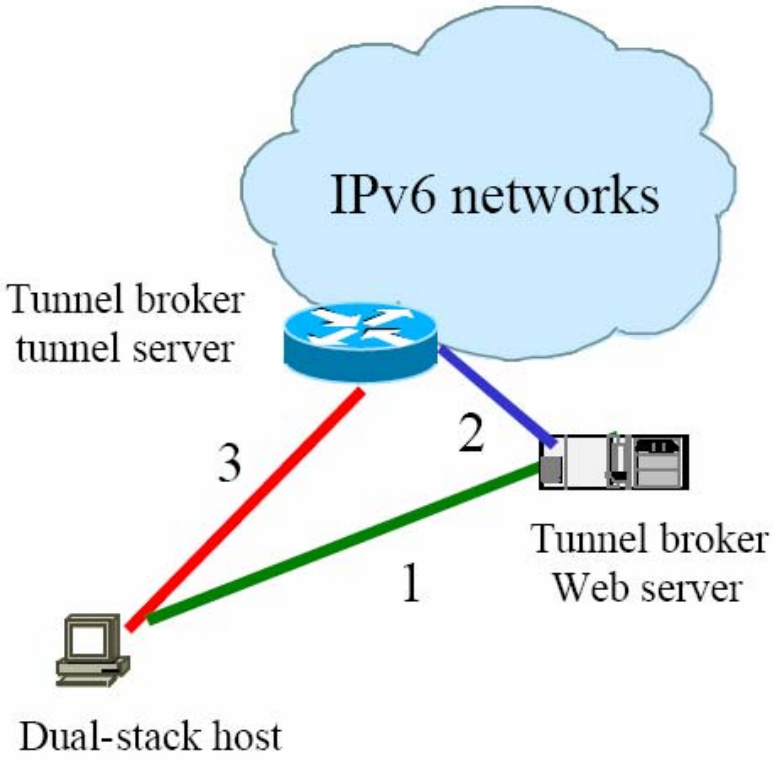
Typical dual stack implementation



Tunnelling

- Provided IPv6 connectivity over IPv4-only networks
- IPv6 traffic is encapsulated into IPv4 packets
  - At the end, the IPv4 traffic will be encapsulated into IPv6 packets
- Multiple mechanisms are proposed:
  - IPv6 over IPv4: Encapsulate packets, manual install
  - TunnelBroker: Semi manual installation of tunnles
  - 6to4: Automatic tunnels, 2002::/16 addresses
  - Toredo: Encapsulate to UDP packets behind NATs
  - Other: ISATAP, 6over4, etc.

Tunnel Broker

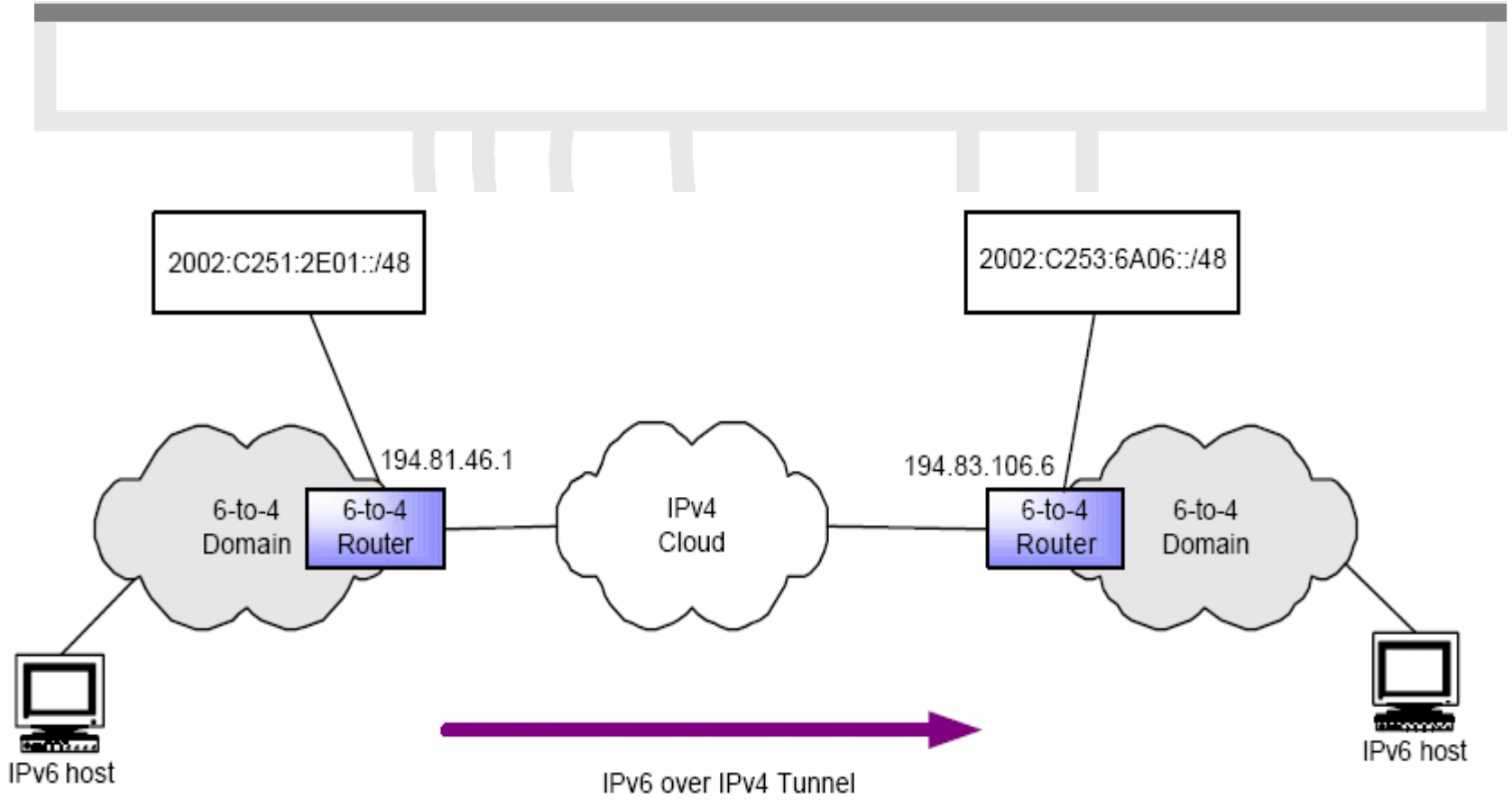


1 — User connects to web server requesting tunnel

2 — Web server returns script to create tunnel to the tunnel server, and informs tunnel server of new client

3 — Client activates script, and gaining access to IPv6 networks via the tunnel server

6 to 4 automatic tunnelling



Translation

- Allows IPv6-only hosts to communicate with IPv4-only hosts
  - Applied as closes as possible at the edge of the network.
  - Addressing can be challenging. How to map IPv6 addresses to IPv4 addresses?
- Common mechanisms
  - Network Address Translation with Protocol Translation (NAT-PT) (RFC2766): An intermediate router convert IPv4 headers into IPv6 headers
  - ALGs: Provide “proxing” at application layer

Should be considered as a last resort solution!

Simplified transition methodology strategy

- Get an IPv6 address space allocation
- Deploy an IPv6-enabled router
- Arrange external IPv6 connectivity
- Enable internal routing
- Deploy security measures and enable basic networking services
- Enable IPv6 to end-systems
- Active IPv6 to applications

Questions?