Introduction to IPv6

LACNIC Campaign
Latinamerica and the Caribbean in IPv6
1/1/11

Ariel Sabiguero

asabigue@fing.edu.uy

Haiti, August 2008

Agenda

- IPv6 History & backgroud
- IPv6 header format (vs IPv4 header)
- IPv6 addresses
- ICMPv6
- Neighbor Discovery
- Transition mechanisms
- IPv6 Ready Logo Programme

IPv6 history & background

IPv6 history

Problems in IPv4 are long known:

- In 1991 the IETF set up a group to analyze the growth of Internet and discuss different alternatives
- Just the next year, the IETF determined that a new generation of Internet Protocols were required: IPng
- In 1994, from the different possible options (CATNIP, SIPP, TUBA), SIPP (Simple Internet Protocol Plus) evolved into IPv6
- In 1998 the first mature set of standards (RFC 2460, 2461, etc.) were published.

IPv6 history (cont)

Problems to be addressed were mainly

- *Scale*
  - Bigger address space
  - Support hyerarchical routing
- *Functionality*
  - Security
  - Autoconfiguration (plug-n-play)
  - Quality of service
  - Mobility

IPv6 history (cont)

Address field size

- ***Some suggested 64 bits for addresses***
  - Meets IPng requirements
  - Minimizes address overhead
  - Efficient software processing
- ***Others supported 160 bits for addresses, variable length***
  - Compatible with OSI NSAP
  - IEEE 802-based autoconfiguration
  - Could start with small addresses and grow later
- ***IPv6 is engineered with 128 bits***

IPv4 and IPv6 reference stacks

Application layer	DNS	SSH	SMTP	HTTP	…
Transport layer	TCP	UDP	…		
Network layer	IGMP, ICMP, ARP IP (v4)	ICMPv6 IP (v6)			
Data link / physical layers	Ethernet	PPP	HDLC	…	

IPv6 terminology

Node: IPv6 device

Router: Node that forwards IPv6 paquets

Host: A node that is not a router

Neighbors: nodes connected to the same link

Interface: node's connection to the link

Address: value given to an IPv6 interface of a node

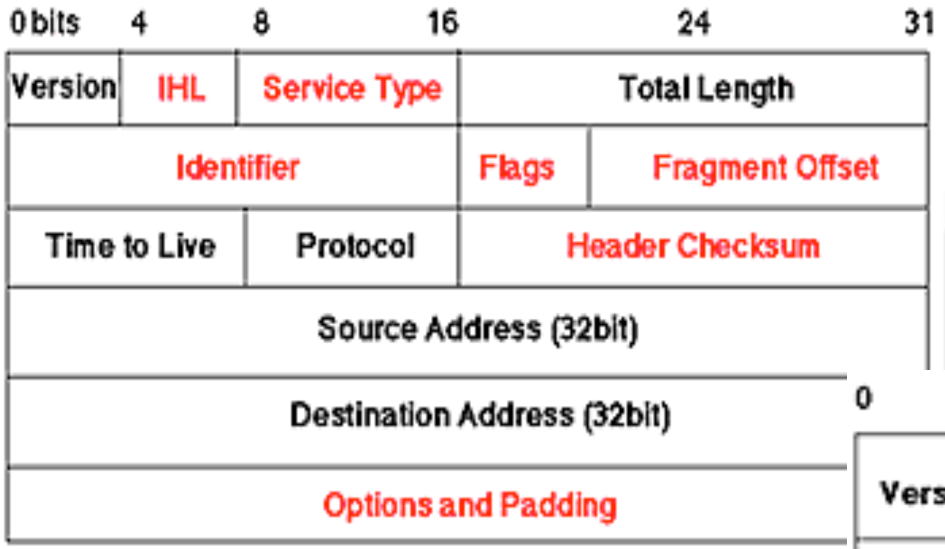
Packet: IPv6 message (IPv6 header + data)

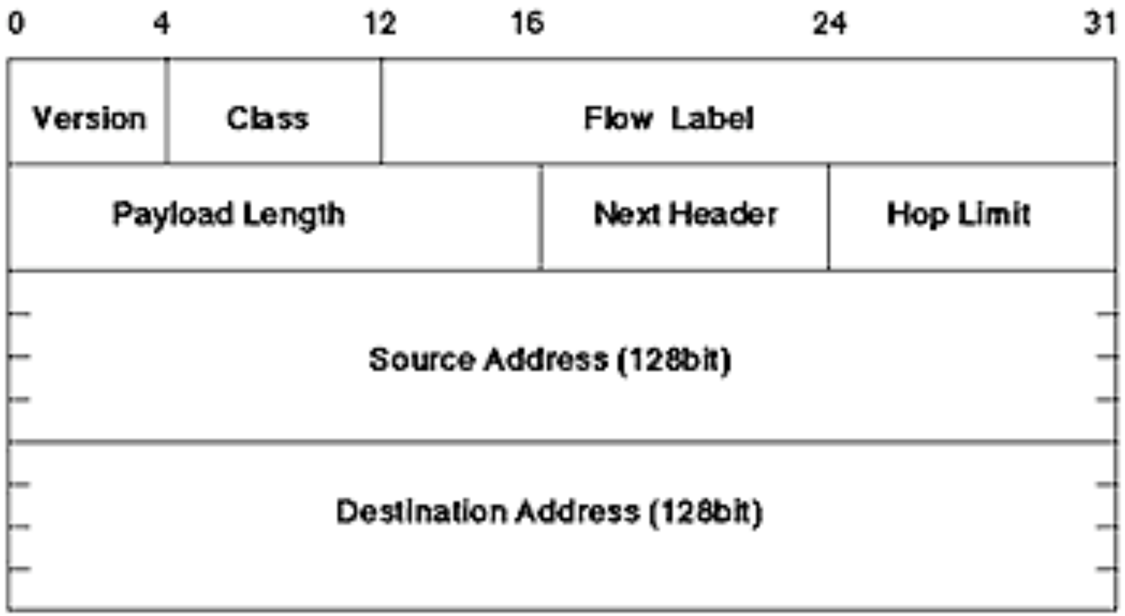
Link MTU: Link's Maximum Transmission Unit

Path MTU: Minimum Link MTU of the path betwen two end nodes

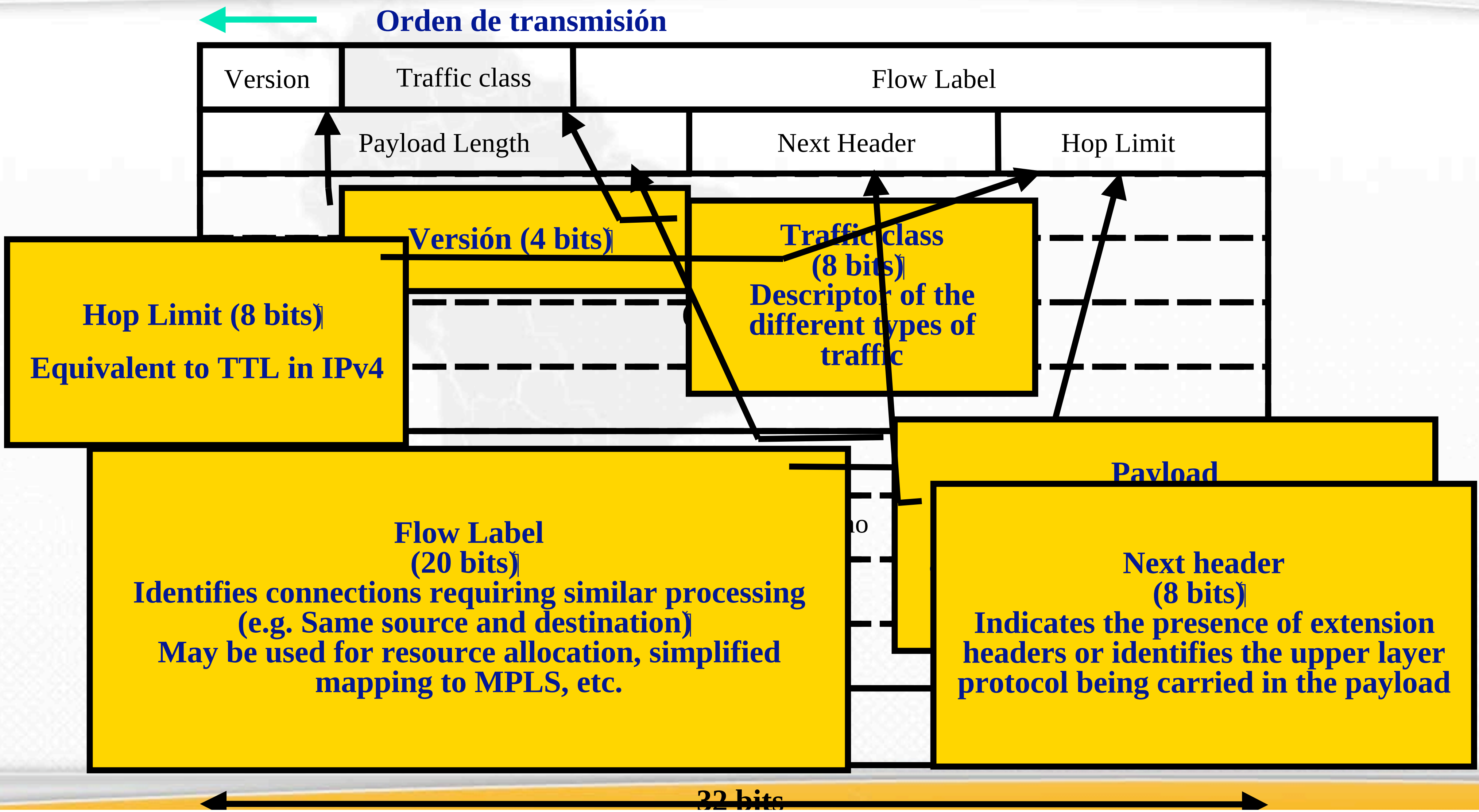
IPv6 header format

IPv4 and IPv6 headers





IPv6 header



Orden de transmisión
Version
Traffic class
Flow Label
Payload Length
Next Header
Hop Limit
Versión (4 bits)
Traffic class (8 bits) Descriptor of the different types of traffic
Hop Limit (8 bits)
Equivalent to TTL in IPv4
Payload
Flow Label (20 bits) Identifies connections requiring similar processing (e.g. Same source and destination) May be used for resource allocation, simplified mapping to MPLS, etc.
Next header (8 bits) Indicates the presence of extension headers or identifies the upper layer protocol being carried in the payload
32 bits

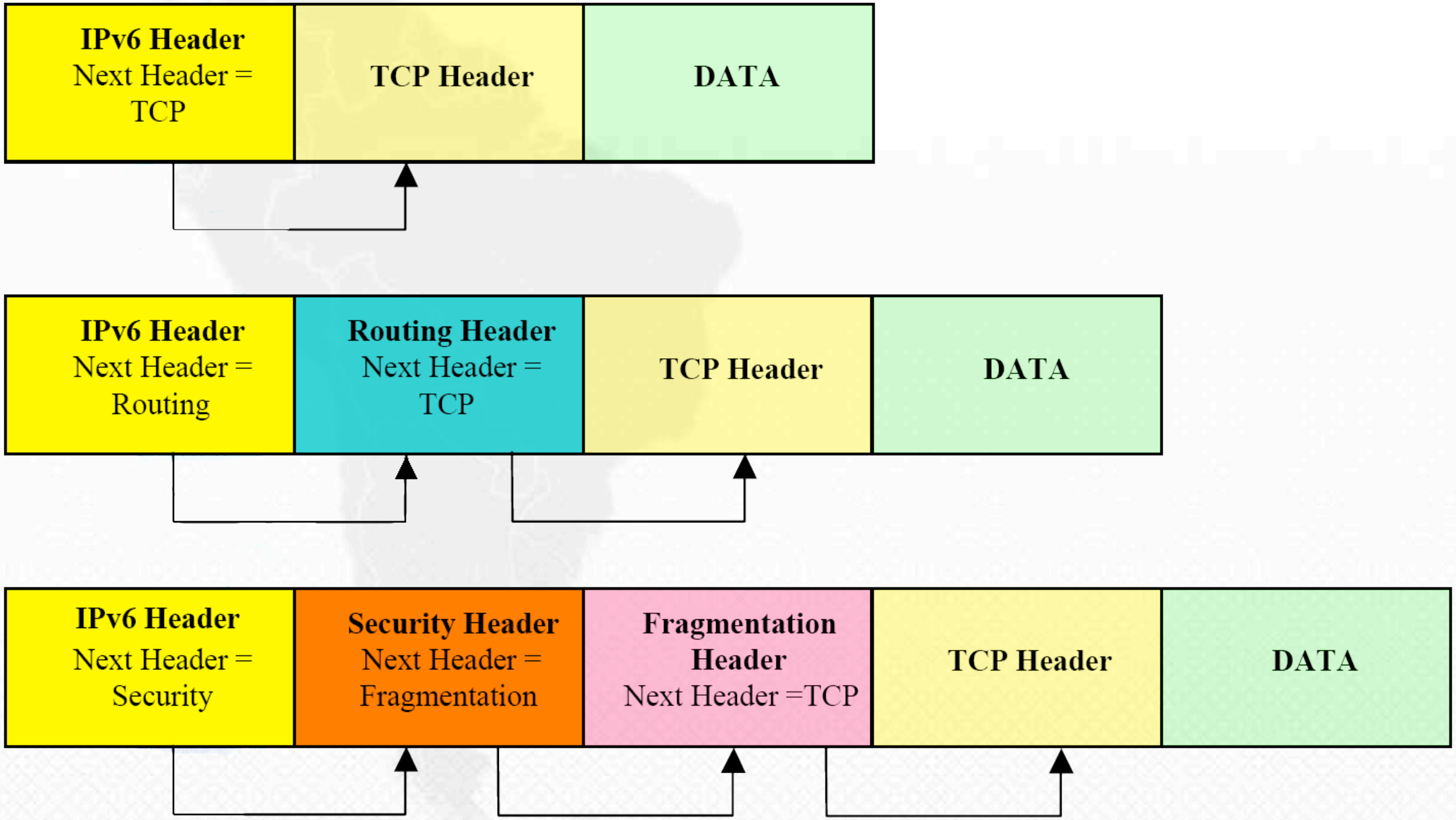
IPv6 header changes

- Fixed length: 40 bytes
- 128 bit addresses
- Fragmentation and options removed
- Length only accounts effective payload
- New field: flow label
- TOS -> Traffic Class
- Protocol -> Next Header
- Time to live -> Hop Count

IPv6 extension headers

- Options are handled through *extension headers*
- Headers are linked with the field Next Header
- Values are interoperable with IPv4 Protocol (i.e. TCP = 6, UDP = 17 , etc.)
- Extension headers:
  - Hop-by-hop header (NH=0)
  - Routing header (NH=43)
  - Fragment header (NH=44)
  - Authentication header (NH=51)
  - Encapsulated security payload (NH=50)
  - Destination option (NH=60)
  - ......

IPv6 extension headers



Fragmentation header

- Only end-to-end fragmentation (not done in intermediate routers)
- Path MTU discovery algorithm
- IPv6 requires a minimum link MTU of 1280 bytes for any link, thus, 1280 is also a possible value of the MTU path
- Maximum payload is 65536 bytes (MTU = Payload + header length)

Fragmentation header

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Next Header  |   Reserved    |      Fragment Offset    |Res|M|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Identification                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
Next Header          8-bit selector.  Identifies the initial header
                     type of the Fragmentable Part of the original
                     packet (defined below).  Uses the same values as
                     the IPv4 Protocol field [RFC-1700 et seq.].

Reserved             8-bit reserved field.  Initialized to zero for
                     transmission; ignored on reception.

Fragment Offset      13-bit unsigned integer.  The offset, in 8-octet
                     units, of the data following this header,
                     relative to the start of the Fragmentable Part
                     of the original packet.

Res                  2-bit reserved field.  Initialized to zero for
                     transmission; ignored on reception.

M flag               1 = more fragments; 0 = last fragment.

Identification       32 bits is used for facilitating each fragment is
                     correctly reassembled at the receiver.
```

IPv6 addresses

IPv6 address types

- 128 bit addresses
- Three different types (remember also reserved ranges):

•***Unicast***

Identifies exactly one interface

•***Multicast***

Identifies a group of interfaces. A paquet sent to a multicast address is delivered to all the members of the group

•***Anycast***

A paquet sent to an anycast address is delivered to *"the closest"* member of the group

IPv6 unicast addresses

- ***Unicast* - (RFC 4291)**
  - global
  - link-local
  - site-local (deprecated)
  - Unique Local (ULA)
  - IPv4 compatible (deprecated)
  - IPv4 mapped

IPv6 address in figures

- 340:282.366:920.938:463.463:374.607:431.768:211.456 different addresses
- 2^{96} times more addresses than in IPv4
- Our planet has about 511:263.971:197.990 m^2

  thus, 655.570:793.384:866.943:898.599 addresses per m^2
- Pessimistic hierarchical address allocation
  - 1.564 addr / m^2
- Optimistic hierarchical address allocation
  - 3:911.873:538.269:508.102 addr / m^2

IPv6 address notation

The 128 bits of an IPv6 address are written as eight 16 bits integers in hexadecimal notation. Integer values are separated by colons

FEDC:BA98:7654:3210:FEDC:BA98:7654:3210

IPv6 address notation compression

The RFC 4291 defines different conventions that permits shorter writing of the addresses:

- zeros on the left can be suppressed

000F:000E:000D:000C:0003:0002:0001:0000

F:E:D:C:3:2:1:0

IPv6 address notation compression ..

A single set of integers with the value of 0 can be abreviated with two colons

FEDC:BA98:0:0:0:0:1234:5678

FEDC:BA98::1234:5678

also see the lack of uniqueness

2001:0:0:0:2:0:0:3

2001:0:0:0:2::3

2001::2:0:0:3

IPv6 address notation compression ..

When IPv4 addresses are converted into IPv6 addresses adding a prefix of 96 zero bits, they can be written in decimal doted notation (as in IPv4)

::164.73.32.2

instead of

::A449:2002

This method is called “IPv4 compatible” and has been deprecated since RFC4291. If you are not using it, don't use them!

IPv6 address notation compression ..

When IPv4 addresses are converted into IPv6 addresses adding a prefix of 80 zero bits and then, 16 ones, they can be written in decimal doted notation (as in IPv4)

::FFFF:164.73.32.2

instead of

::FFFF:A449:2002

This method is called “IPv4 mapped” and is proposed to replace “IPv4 compatible” addresses

IPv6 addresses inside URL

If we have to write an IPv6 address inside a URL, it has to be placed between square brakets

http://[FEDC:BA98:7654:3210:FEDC:BA98:7654:3210]/index.html

Let's have our DNS servers well configured ;-)

IPv6 address notation compression ..

Prefixes are noted using the same slashed notation as in IPv4:

FEDC:BA98:7600::/40 is a network address with a 40 bits prefix

IPv6 special addresses (RFC 5156)

Unspecified address: can only be used by a node that does not have an address yet, the address value is "`0:0:0:0:0:0:0:0`" and is abreviated as "`::`" or "`::/128`"

Loopback address: used to send IPv6 datagrams to the same host, the address value is "`0:0:0:0:0:0:0:1`" and is abreviated as "`::1`" or "`::1/128`"

IPv6 special addresses

Default route: required for specifying default routing in routing tables “`0:0:0:0:0:0:0:0/0`” and is abreviated as “`::/0`”

IPv6 address space

http://www.iana.org/assignments/ipv6-address-space

IPv6 Prefix	Allocation	Reference	
0000::/8	Reserved by IETF	[RFC4291]	
0100::/8	Reserved by IETF	[RFC4291]	
0200::/7	Reserved by IETF	[RFC4048]	
0400::/6	Reserved by IETF	[RFC4291]	
0800::/5	Reserved by IETF	[RFC4291]	
1000::/4	Reserved by IETF	[RFC4291]	
2000::/3	**Global Unicast**	**[RFC4291]**	
4000::/3	Reserved by IETF	[RFC4291]	
6000::/3	Reserved by IETF	[RFC4291]	
8000::/3	Reserved by IETF	[RFC4291]	
A000::/3	Reserved by IETF	[RFC4291]	
C000::/3	Reserved by IETF	[RFC4291]	
E000::/4	Reserved by IETF	[RFC4291]	
F000::/5	Reserved by IETF	[RFC4291]	
F800::/6	Reserved by IETF	[RFC4291]	
FC00::/7	**Unique Local Unicast**	**[RFC4193]**	
FE00::/9	Reserved by IETF	[RFC4291]	
FE80::/10	**Link Local Unicast**	**[RFC4291]**	
FEC0::/10	Reserved by IETF	[RFC3879]	site local
FF00::/8	**Multicast**	**[RFC4291]**	

Global Unicast Address (RFC 3587)

001	Glob.rout.prefix	subnetID	interface ID
	45 bits	16 bits	64 bits

Global routing prefix: is the value assigned to a site. Hirerarchicaly, RIRs and ISPs

Sub-net ID: network identifier inside a site RIRs and ISPs administer and allocate this blocks

Interface ID: usually built using EUI-64

Unique Local IPv6 Unicast Addresses – IPv6 ULA (RFC 4193)

- Global prefix, no warranties of uniqueness, but high likeliness.
- Reserved for local communications, normaly inside a site.
- Non routable across the Internet.
- Might be routable in a smaller scope (inside a site or company)
- *Well-known* prefixes that could be easily filtered at the edges.

IPv6 ULA (RFC 4193) - (cont)

- ISP independent and can be used inside sites with or without Internet access
- Unfiltered traffic that escapes to the wild makes no harm
- Applications can treat these addresses as global ones.

IPv6 ULA format (RFC 4193)

Prefix	L	Global ID	subnetID	interface ID
7 bits	1 bit	40 bits	16 bits	64 bits

FC00::/7 prefix

L = 1 means local assignment

L = 0 reserved for future use according to the RFC, enabling central allocation of addresses.

Global ID should be created randomically to minimize collision probability

Interface ID

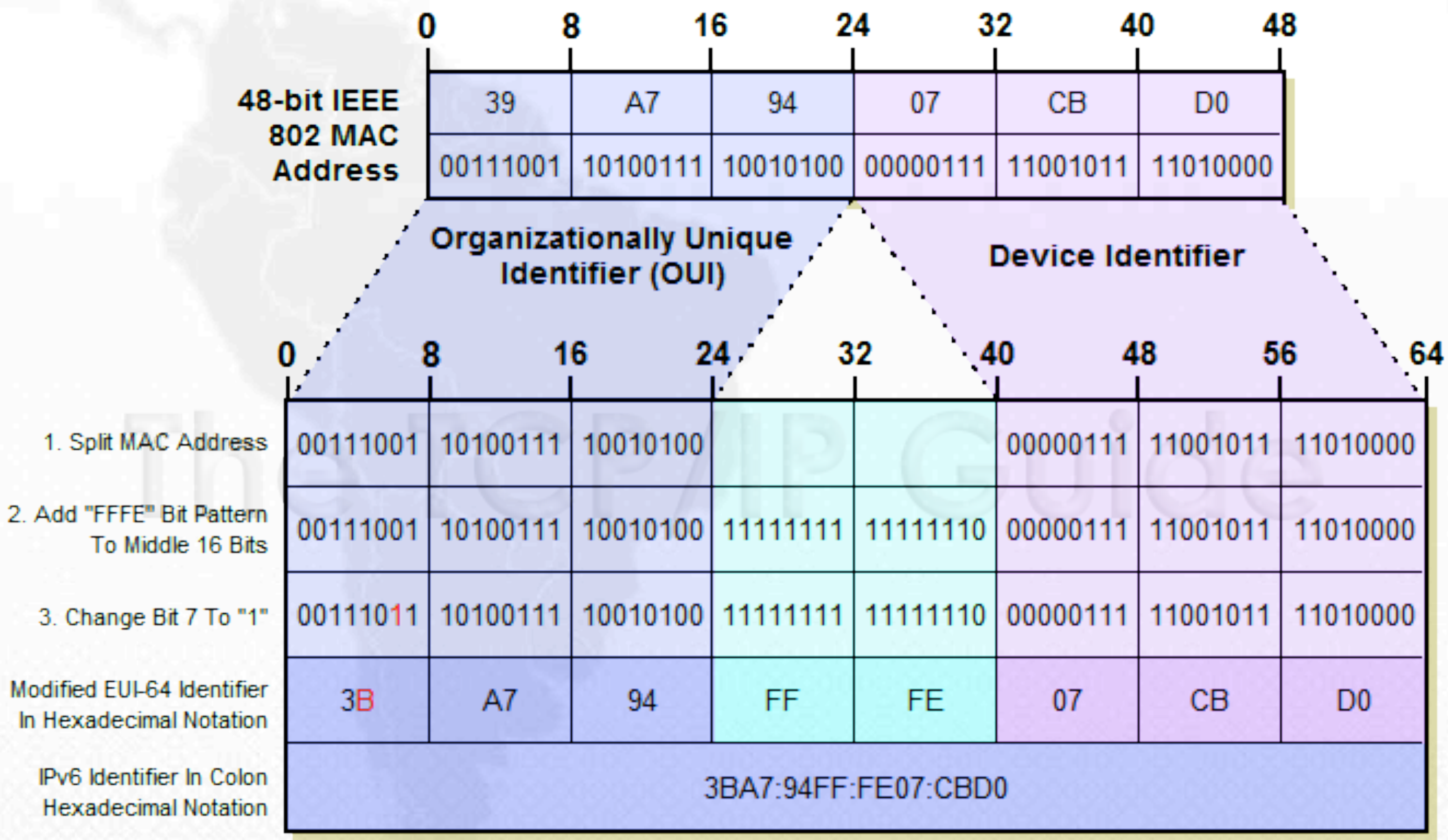
Prefix	interface ID

64 bits

The “rightmost” 64 bits of an IPv6 unicast address can be allocated through different means:

- autoconfiguration (modified EUI-64)
- DHCPv6
- manually configured
- randomly
- future methods supported

Interface ID – modified EUI 64



64-Bit IPv6 Modified EUI-64 Interface Identifier

http://www.tcpipguide.com/free/t_IPv6InterfaceIdentifiersandPhysicalAddressMapping-2.htm

Mandatory IPv6 addresses - Host

- Link-local address for each interface (+ any other unicast or anycast address manually or automatically configured)
- Loopback address
- All-nodes multicast addresses (FF01::1 and FF02::1)
- Multicast Solicited-Node address for each unicast and anycast address
- Multicast addresses for all the groups it belongs to

Mandatory IPv6 addresses - Router

- All the addresses required for a host
- Subnet-router anycast addresses for all the interfaces used for forwarding packets
- All other anycast configured addresses
- All-nodes multicast addresses (FF01::2 and FF02::2)

ICMPv6

ICMPv6 – (RFC4443)

- Obsoletes RFC2463, published in 1998, updated by 4884
- Same philosophy as ICMP for IPv4 (RFC 792), updated to IPv6
- Uses NextHeader value 58
- ICMPv6 is a MUST in the protocol suite and must be fully implemented in every node
- ICMPv6 is used to report IPv6 errors and to perform probes (like ping6)

ICMPv6 Messages

8 bits	8 bits	16 bits
Type	Code	Checksum
Message body		

Two classes:

- Error messages (types 0 to 127)
- Informative messages (types 128 to 255)

ICMPv6 error messages

8 bits	8 bits	16 bits
Type = 0-127	Code	Checksum
Parameter		

Original message.

If the size of the resulting ICMPv6 packet exceeds 1280 bytes, then it is truncated to that size (remember minimum MTU requirements).

Some ICMPv6 error messages

Destination unreachable message (type 1, parameter 0), code:

0 – No route to destination

1 – Communication with the destination administratively prohibited

2 – Beyond scope of source address

3 – Address unreachable

4 – Port unreachable

5 – Source address failed ingress/egress policy

6 – Reject route to destination

Some ICMPv6 error messages (cont)

Packet too big message (type 2, code 0, parameter = next-hop MTU).

Time exceeded message (type 3, parameter = 0), code:

0 – Hop limit exdeeded in transit

1 – Fragment reassembly time exceeded

Parameter problem message (type 4), code:

0 – Erroneous header field

1 – Unrecognized Next Header type

2 – Unrecognized IPv6 option

parameter field (called pointer) identifies the byte offset within the invoking paket where the error was detected

Some ICMPv6 informative messages

8 bits	8 bits	16 bits
Type	Code	Checksum
Message body		

ping6

ICMPv6 Echo Request message (type 128, code 0)

ICMPv6 Echo Reply message (type 129, code 0)

ICMPv6 informative messages (cont)

8 bits	8 bits	16 bits
Type	Code	Checksum
Message body		
Multicast Address		

Multicast Listener Discovery (MLD) Messages:

Query, report, done (as IGMP for IPv4)

Neighbor discovery

Neighbor Discovery ND

- Originaly, RFC2461, published in 1998 defined the protocol. Now it has been updated by the RFC 4861.
- Nodes uses ND to determine *data link layer address* (MAC address) of nodes belonging to the same network segment.
- Hosts also use ND to find neighbor routers.
- ND is a key element of the autoconfiguration in IPv6

ND – messages

ND defines 5 different types of packets:

- Router Solicitation (RS)
- Router Advertisement (RA)
- Neighbor Solicitation (NS)
- Neighbor Advertisement (NA)
- Redirect

ND – Router Advertisements

- In a multiple access link (i.e. IEEE 802 family), every router multicast periodically RA messages.
- The hosts in the link receive RA of all routers in the link, building it's routing table (maybe with several default `“::/0”` routes)
- Neighbor Unreachability Detection (NUD) detects connectivity problems to the routers.

ND – Router Advertisements (cont)

- RA carry a list of the prefixes assigned to the link. The list should be used by the hosts in the link to autoconfigure their corresponding addresses based on the prefixes.
- Different Flags present in the RA and associated to each prefix allow the routers to indicate how to perform the autoconfiguration (stateless or through DHCPv6)

ND – Neighbor Solicitation

- Nodes send NS to determine dynamically the IPv6 – MAC mapping.
- NS uses multicast when the node needs to resolve an address and unicast to determine reachability
- NS replaces ARP request messages in IPv4, providing enhanced characteristics and having a better integration in the protocol suite.

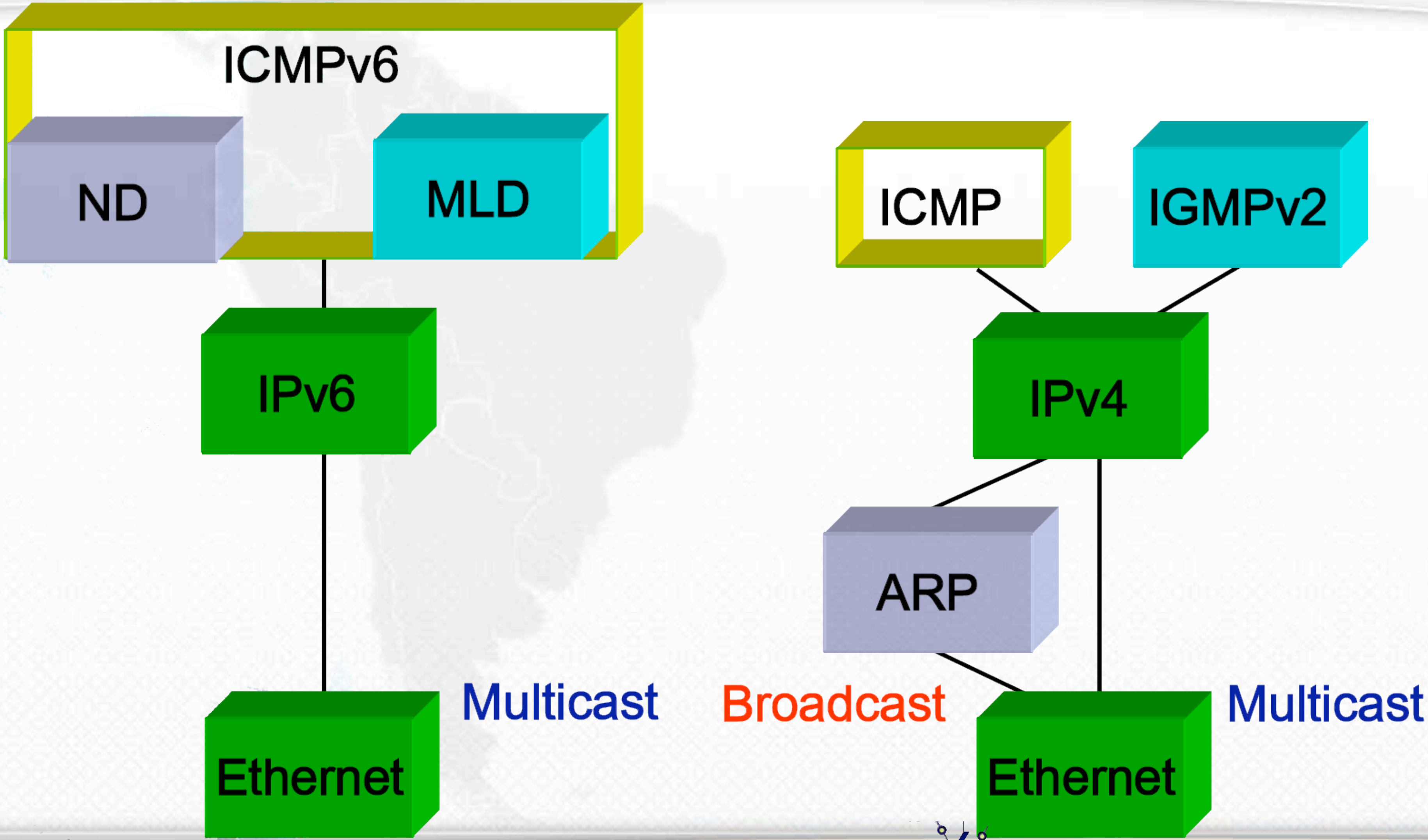
ND – Neighbor Advertisement

- Nodes send NA to determine answer NS
- Nodes can also send unsolicited NS in order to propagate new information rapidly

ND – Redirect

- Routers send redirect packets to inform a host that there exist a “better” router for that particular destination.
- Redirect can also be used to inform a host that the destination is a neighbor.

IPv6 vs. IPv4 control planes



Transition mechanisms

Why transition?

- Internet exists, works and runs on IPv4 today
- We cannot change the network overnight
- While we perform required changes, former IPv4 and new IPv6 must coexist
- Not only the protocols have to be considered, but the whole infrastructure and applications running on top of them

Transition schemas

- Several different techniques have been designed and can be grouped in:
  - Dual-stack
  - Tunneling
  - Translation
- These mechanisms can even be used simultaneously.

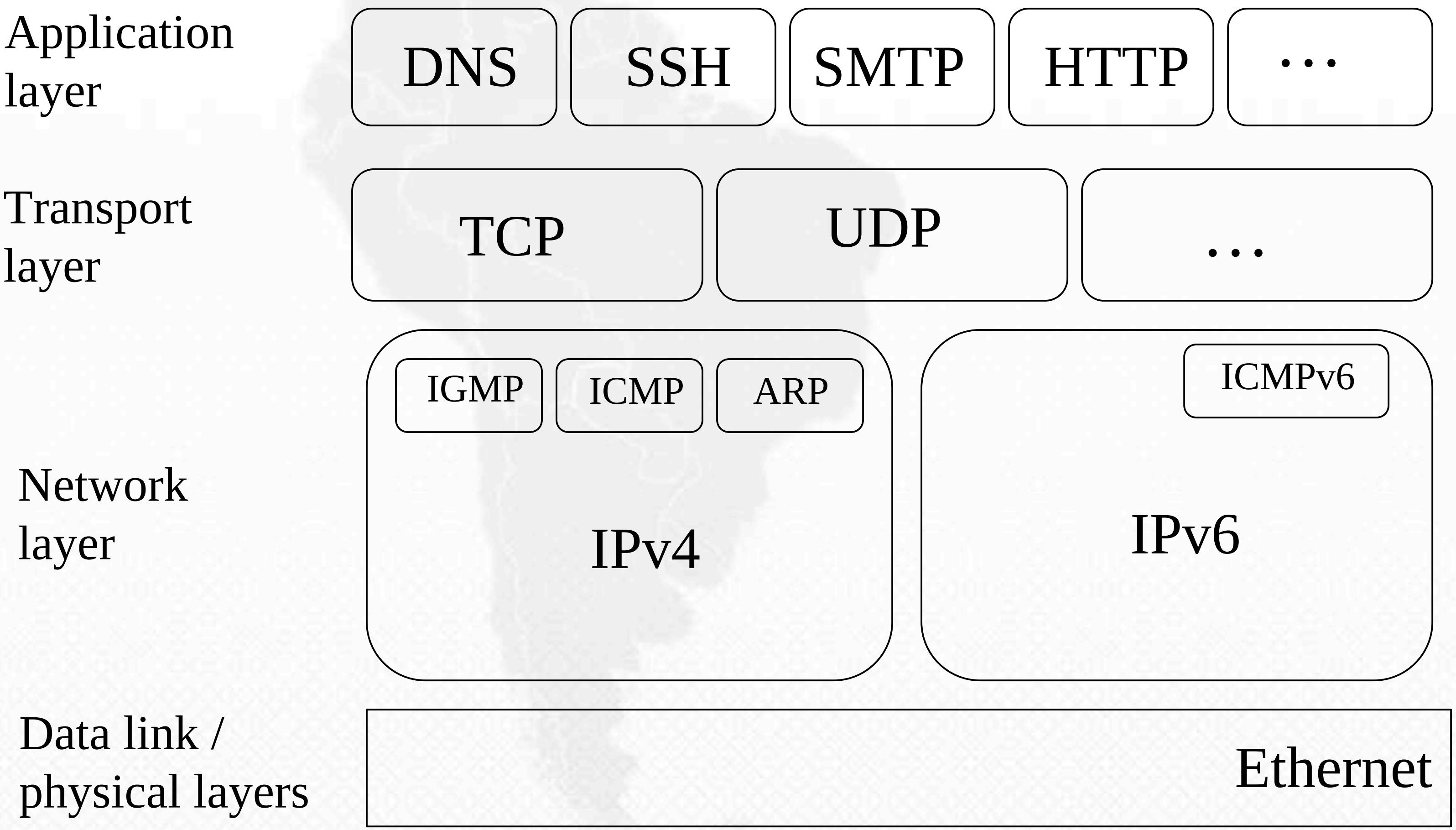
Dual Stack

- IPv6 can be added to any IPv4 enabled device.
- Protocol are multiplexed and de-multiplexed over same medias (i.e. IEEE 802 family) using different protocol numbers in the same frame position
- Same approach as used for mixing IPX, Appletalk, TCP, etc.

Dual Stack (cont)

- It is an application problem to decide which protocol to use (i.e. if a DNS answer contains an AAAA field, then prefer TCP over IPv6 for transport)
- This enables a smooth transition allowing application developpers to update gradually their applications
- Languages like Java allows InetAddress objects to be generalizations of Inet4Address and Inet6Address, making the representation independent of the protocol

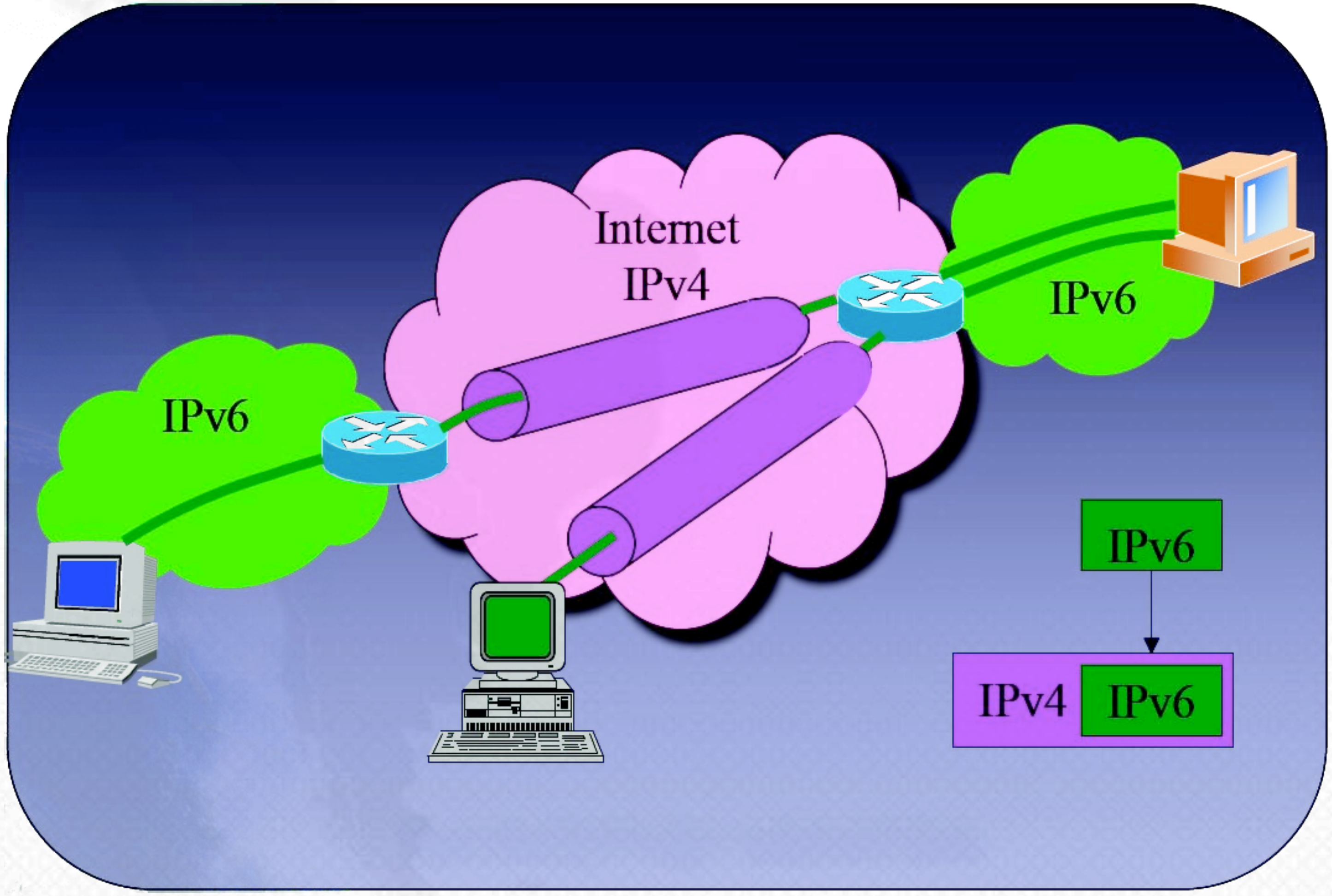
Dual stack schema



Tunneling

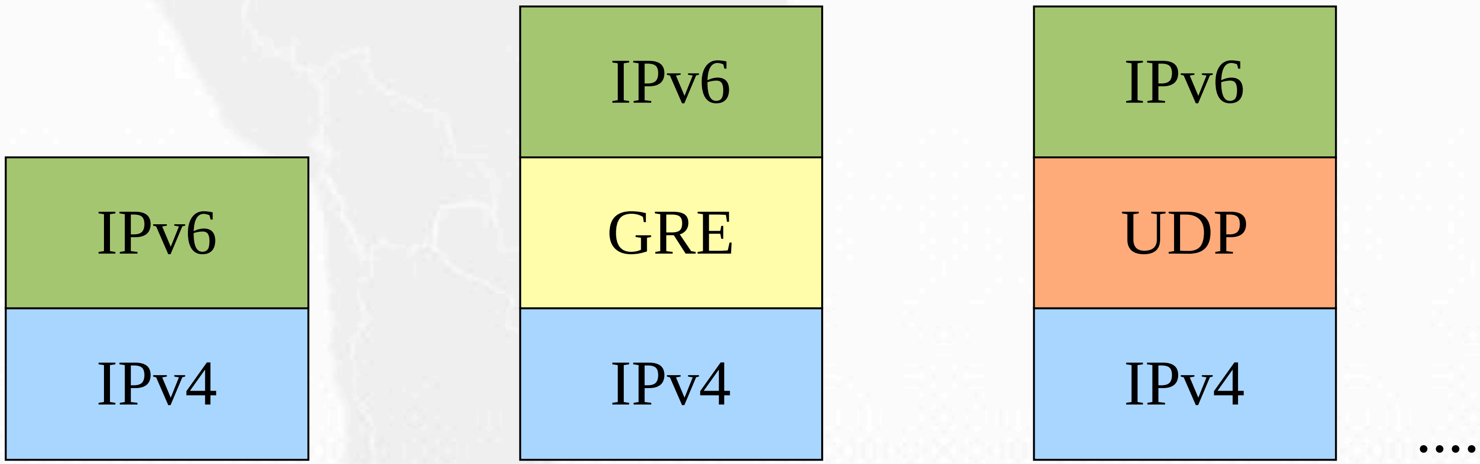
- We use tunneling to “hide” IPv6 traffic inside IPv4 traffic in order to cross sections of the network that are not IPv6 Ready yet
- IPv6 packets are encapsulated into IPv4 ones that can be forwarded as regular IPv4 traffic
- Conceptually, it can be seen as:
  - IPv6 using IPv4 as a virtual link layer
  - An IPv6 VPN configured over IPv4 Internet

Tunneling – graphical concept



Tunneling concepts

- There are different ways to put IPv6 into IPv4



Tunneling mechanisms

- There exist a broad variety of technologies. Some of them are:
- 6in4
- TB
- TSP
- 6to4
- Teredo
- 6over4
- AYIYA
- DSTM
- ……

Tunneling: 6in4 (RFC 4213)

- Direct IPv6 encapsulation over IPv4 using IP protocol number 41
- Commonly used to connect:
  - End-node → router
  - Router → router
- But also possible for end-node → end-node connections
- The tunnel is considered as a point-to-point link, counting as a single hop

Tunneling: 6in4 (RFC 4213) (cont)

- IPv6 addresses at both ends of the tunnel have the same prefix
- 6in4 requires manual configuration
- ALL IPv6 connections of the end-node are tunneled and routed through the router at the end of the tunnel
- It is required to have protocol 41 forwarding all through the path between the ends of the tunnel
- It can be started behind a NAT box, provided that there is protocol 41 forwarding

Tunneling: TB (RFC 3053)

- The idea behind Tunnel Broker is to ease end-node configuration and administration of addresses
- Usually, the TB offers a web interface to interact with the end-node
- When the user requires the creation of a tunnel, TB configures the router that will provide IPv6 access, assigns an IPv6 address to the client and provides instructions to create the tunnel on the client side.
- TB list at http://www.ipv6tf.org/using/connectivity/test.php

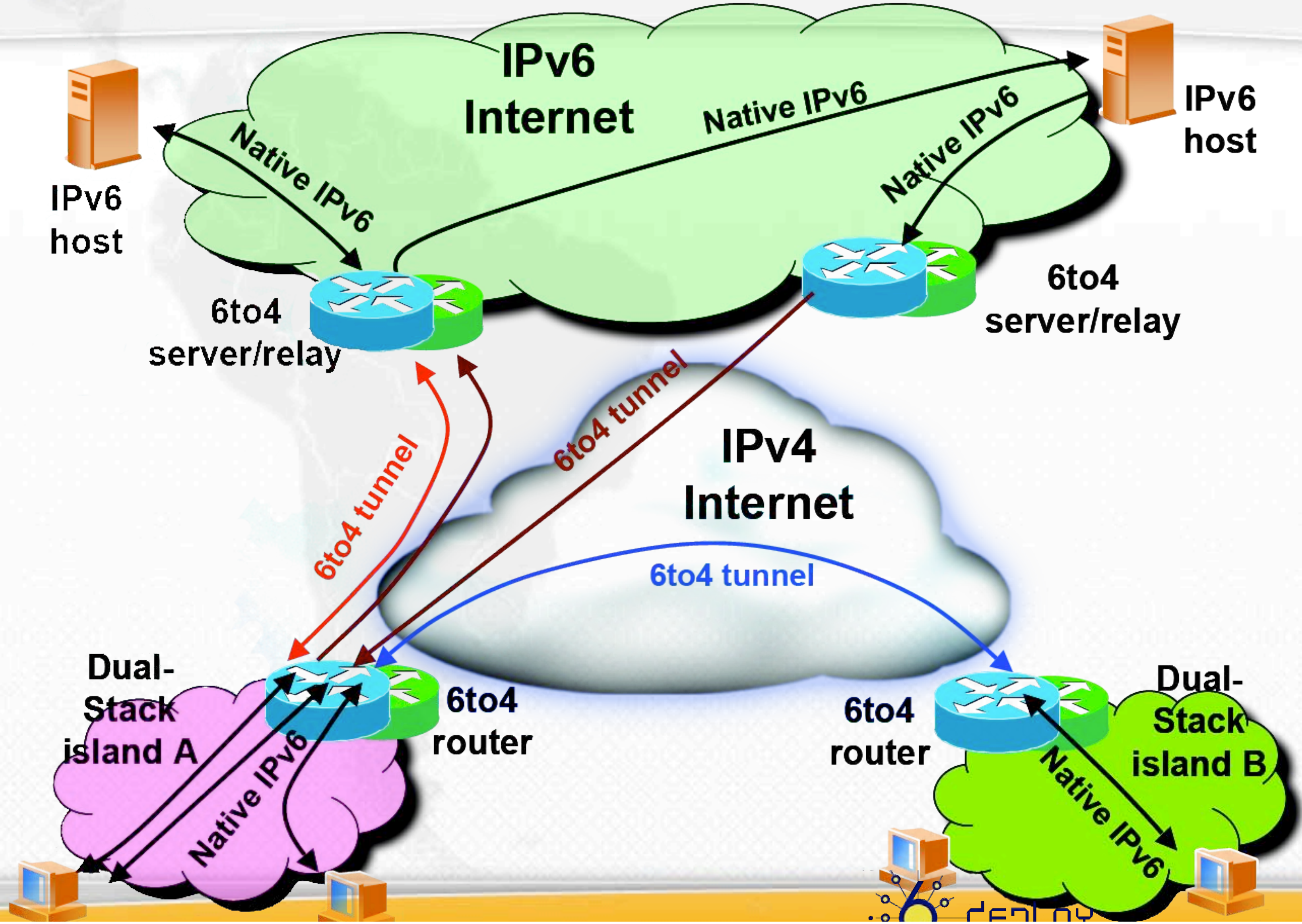
Tunneling: TB (RFC 3053) (cont)

- IPv6 addresses at both ends of the tunnel have the same prefix
- 6in4 requires manual configuration
- ALL IPv6 connections of the end-node are tunneled and routed through the router at the end of the tunnel
- It is required to have protocol 41 forwarding all through the path between the ends of the tunnel
- It can be started behind a NAT box, provided that there is protocol 41 forwarding

Tunneling: 6to4 (RFC 3056)

- IPv6 encapsulation in IPv4 similar to 6in4
- Main differences are:
- IPv6 address on the client side does not depend on the router that is connected to, but to its IPv4 public address
- Outgoing traffic is routed through the same "6to4 relay", but incomming traffic may come from other "6to4 relays".

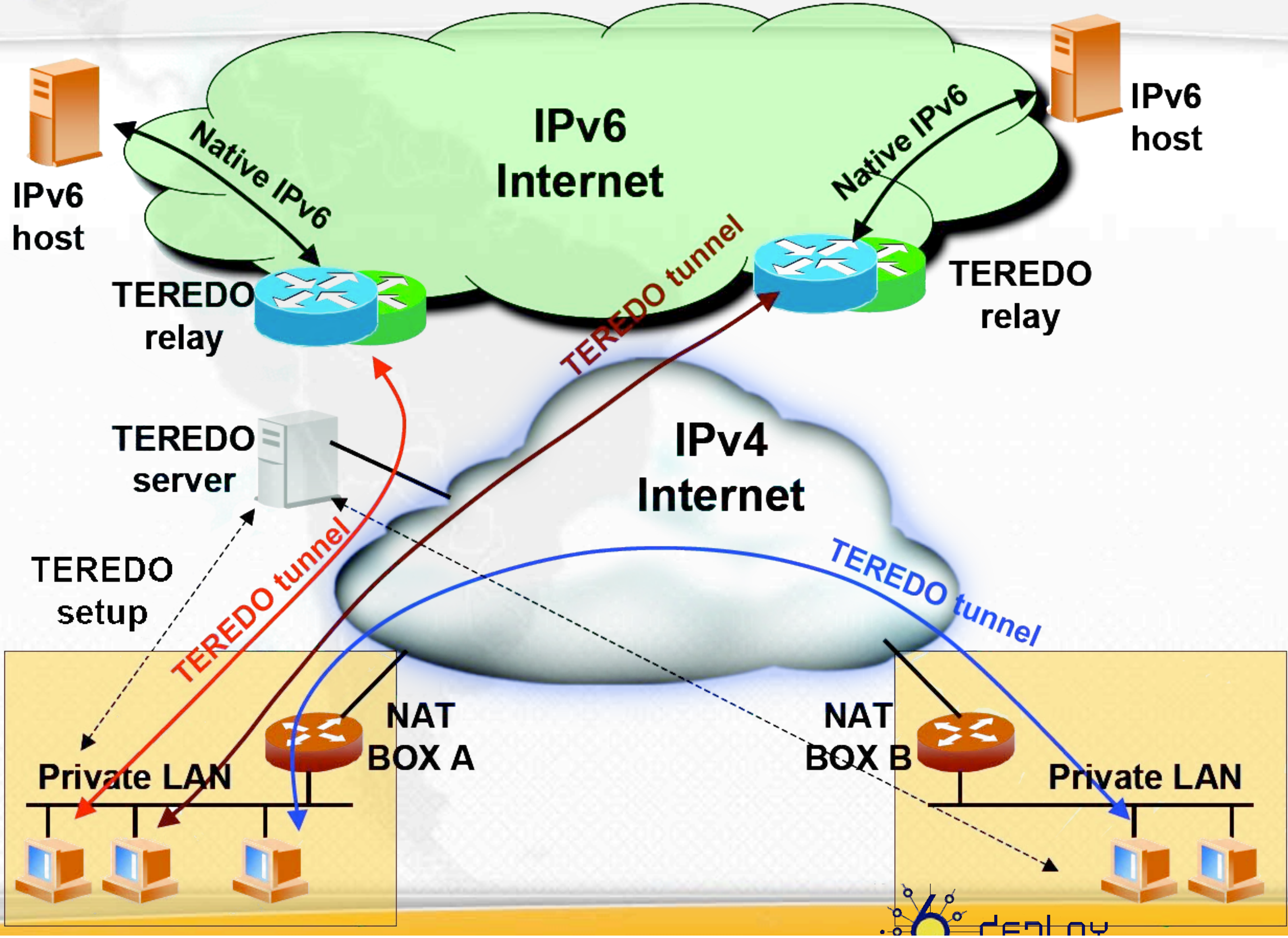
Tunneling: 6to4 (RFC 3056) (cont)



Tunneling: Teredo (RFC 4380)

- IPv6 encapsulation over UDP, over Ipv4
- Designed to provide access to hosts behind NAT boxes without protocol 41 forwarding
- Different agents involved:
- Teredo Server
- Teredo Relay
- Teredo Client

Tunneling: Teredo (RFC 4380) (cont)



Tunneling: ...

- There is -a lot- more to say about tunneling applied to IPv6... (not enough time)
- The transition will happen as the coverage of native IPv6 “islands” grow
- The islands are and will be joined by some kind of tunnel (otherwise, we don't have full-scale connectivity)
- The transition will end when all islands become a single -new- Internet
- ... we will keep on using IPv6-IPv6 tunnels for several purposes, they are a great tool, not just for transition

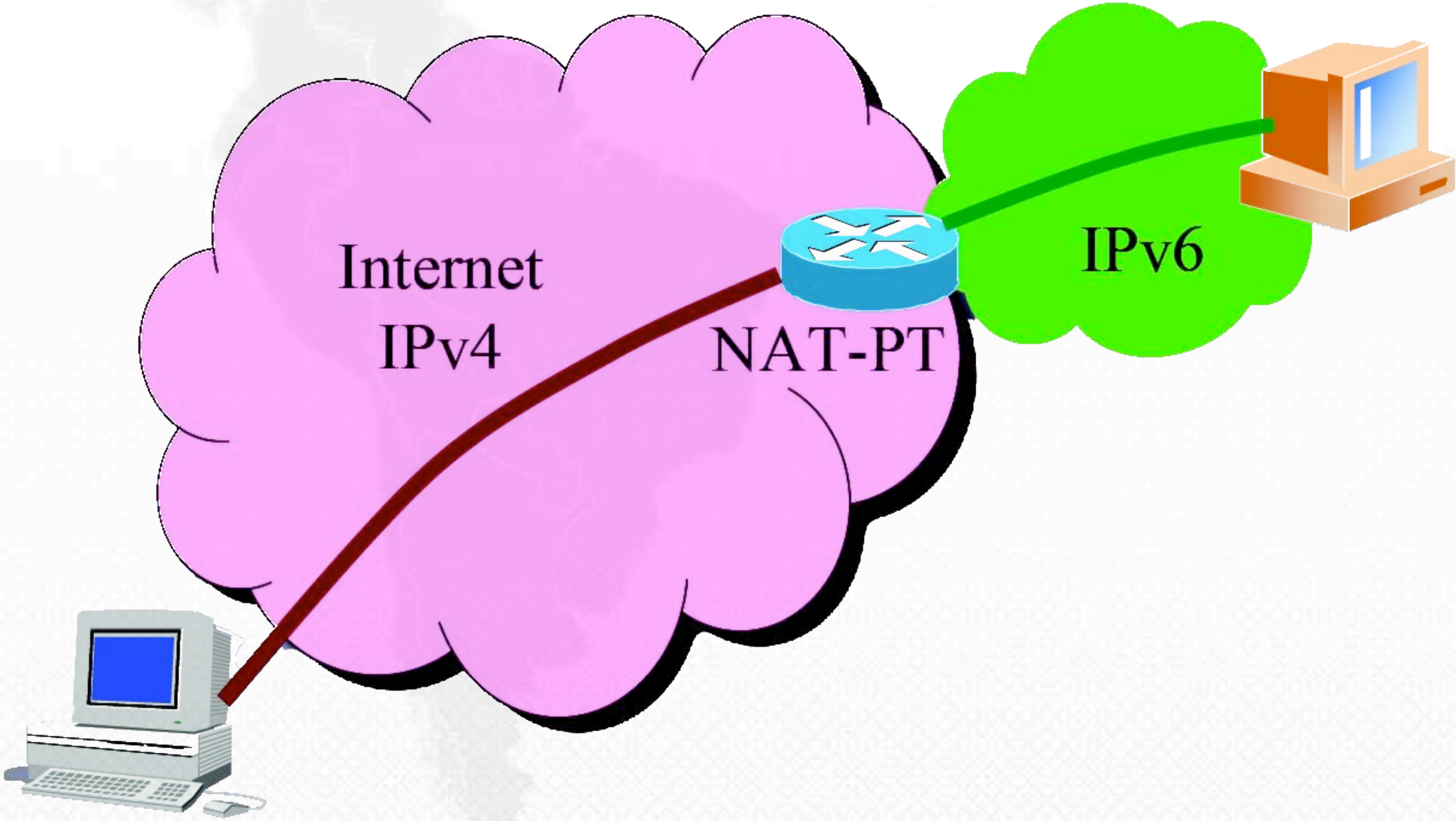
Translation

- Translation mechanisms are all deprecated nowadays (even though, still are being used)
- They are based on the conversion of IPv4 into IPv6 packets and viceversa
- Could be understood as an extension of NAT/PAT techniques, affecting not only addresses and ports, but network layer header
- In the “IPv6 native” network, we have full services, but on the “IPv4 translated” network, we have some restricted services

Translation (cont)

- As network layer protocols are not functionaly equivalent, upper layer inspection must be done in order to do the translation of some -too many- protocols
- From the complexity point of view, this is the worse solution
- Could be used for legacy systems where no upgrade is available

Translation: NAT-PT (example)



IPv6 Ready Logo Programme
v6RL

v6RL - why?

- Avoiding confusion in the mind of customers with a unique program globaly
- Giving a strong signal to the market that IPv6 is ready and available
- Proving the interoperability degree of various IPv6 products
- Enhancing confidence of users that IPv6 is currently operational

The IPv6 Ready Logo program should contribute to the feeling that IPv6 is available and ready to be used.

v6RL committee (the v6LC)

- Launched by the IPv6 Forum with the support of WIDE/TAHI (Japan), ETSI and IRISA (Europe) and the UNH-IOL (USA)
- Based mainly on interoperability testing results
- The ipv6ready-admin
  - Defining procedures and steps for the Logo Program
  - Giving the right to use the IPv6 logos for products
- The ipv6ready-tech
  - Test specification and test tools providing
  - Technical examination of applications

v6RL – smooth and gradual approach

Different phases:

- Phase I “Silver” / (bootstrap)
  - Since September 2003
  - Based on existing interoperability envents and tools
  - Ipv6 minimum requirements of mandatory core protocols (“MUST”)
- Phase II “Gold”
  - Launched in January 2005
  - Products have to satisfy strong requirements (“must” and “should”)
  - Core Protocols, Ipsec, MIPv6, NeMO, Transition mechanisms, Multicast (MLD)
- Phase III to follow

Some items from RFC 2119

MUST This word, or the terms “REQUIRED” or “SHALL”, mean that the definition is an absolute requirement of the specification

SHOULD This word, or the adjective “RECOMMENDED”, mean that there may exist valid reasons in particular circumstances to ignore a particular item, but the full implications must be understood and carefully wighted before choosing a different course.

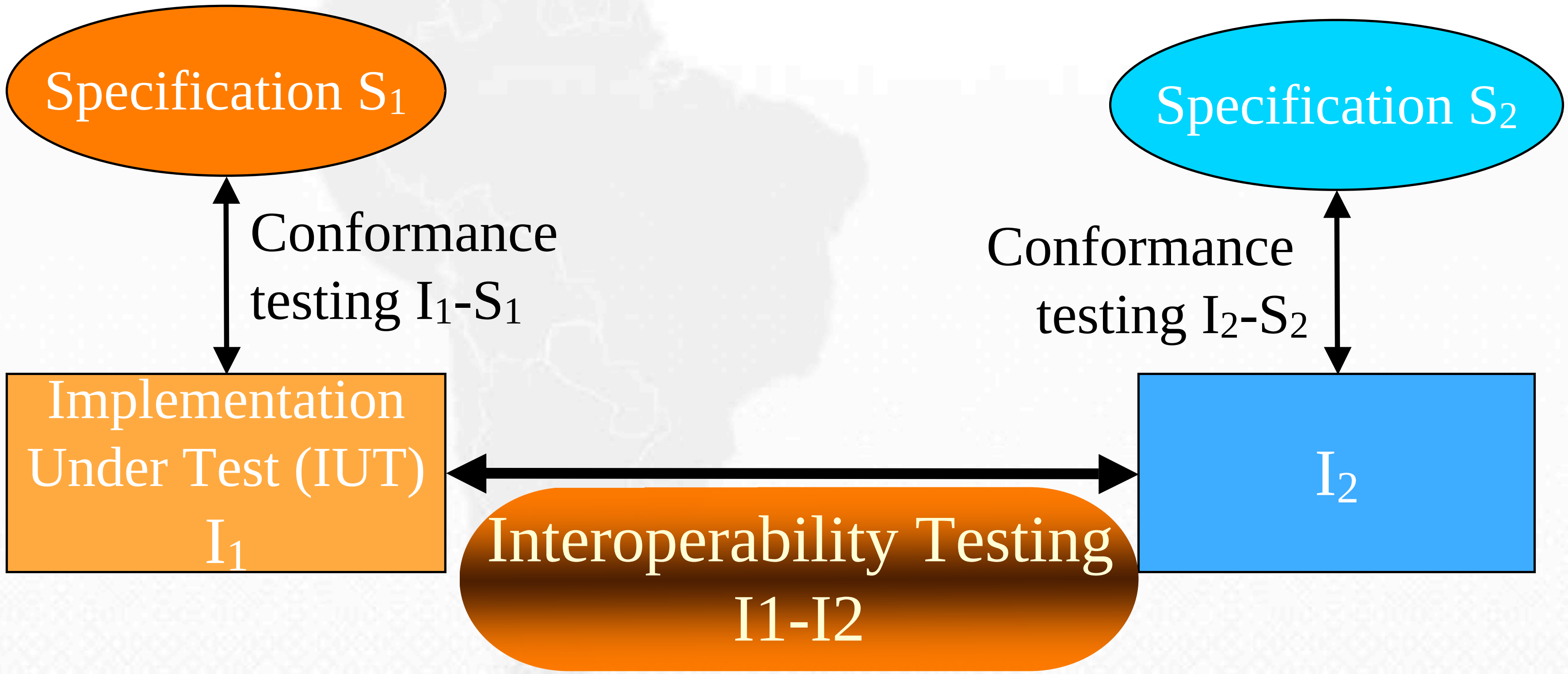
Coverage of the tests

Phase II: « Core Protocols »
[« Must » + « Should »]

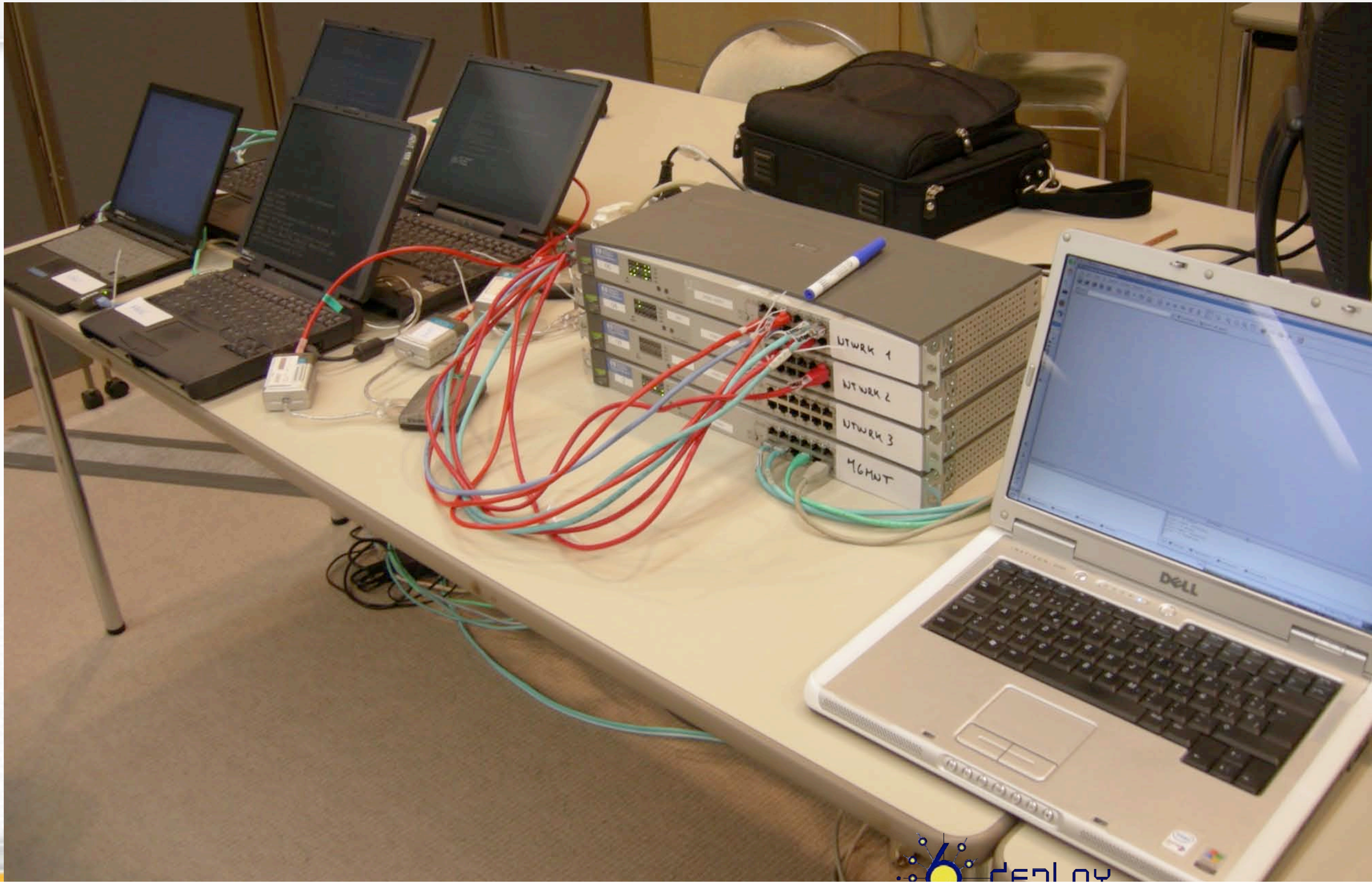
- 76 IPv6 Specification [RFC2460] 40
- 127 Neighbor Discovery [RFC2461] 28
- 26 Stateless Address Autoconf. [RFC2462] 26
- 41 ICMPv6 Specification [RFC2463] 13
- 15 Path MTU Discovery [RFC1981] 0

Phase I: « Core Protocols »
[Mandatory: sub-set of phase 2]

Conformance vs Interoperability testing



Interoperability platform for IPv6 testing



Merci beaucoup !