# IPv6 Tutorial

# G6

Last modified by: Bernard.Tuy@renater.fr

on *8 Feb. 2004*

# Contributions

- Main authors
  - Laurent Toutain, ENST-Bretagne – IRISA, France
  - Bernard Tuy, Renater, France
- Contributors
  - Octavio Medina, ENST-Bretagne, France
  - Mohsen Souissi, AFNIC, France
  - Vincent Levigneron, AFNIC, France
  - Thomas Noel, LSIIT, France
  - Alain Durand, Sun Microsystems, USA
  - Alain Baudot, France Telecom R&D, France
  - Bill Manning, ISI, USA
  - David Kessens, Qwest, USA
  - Pierre-Emmanuel Goiffon, Renater, France
  - Jérôme Durand, Renater, France

# Agenda

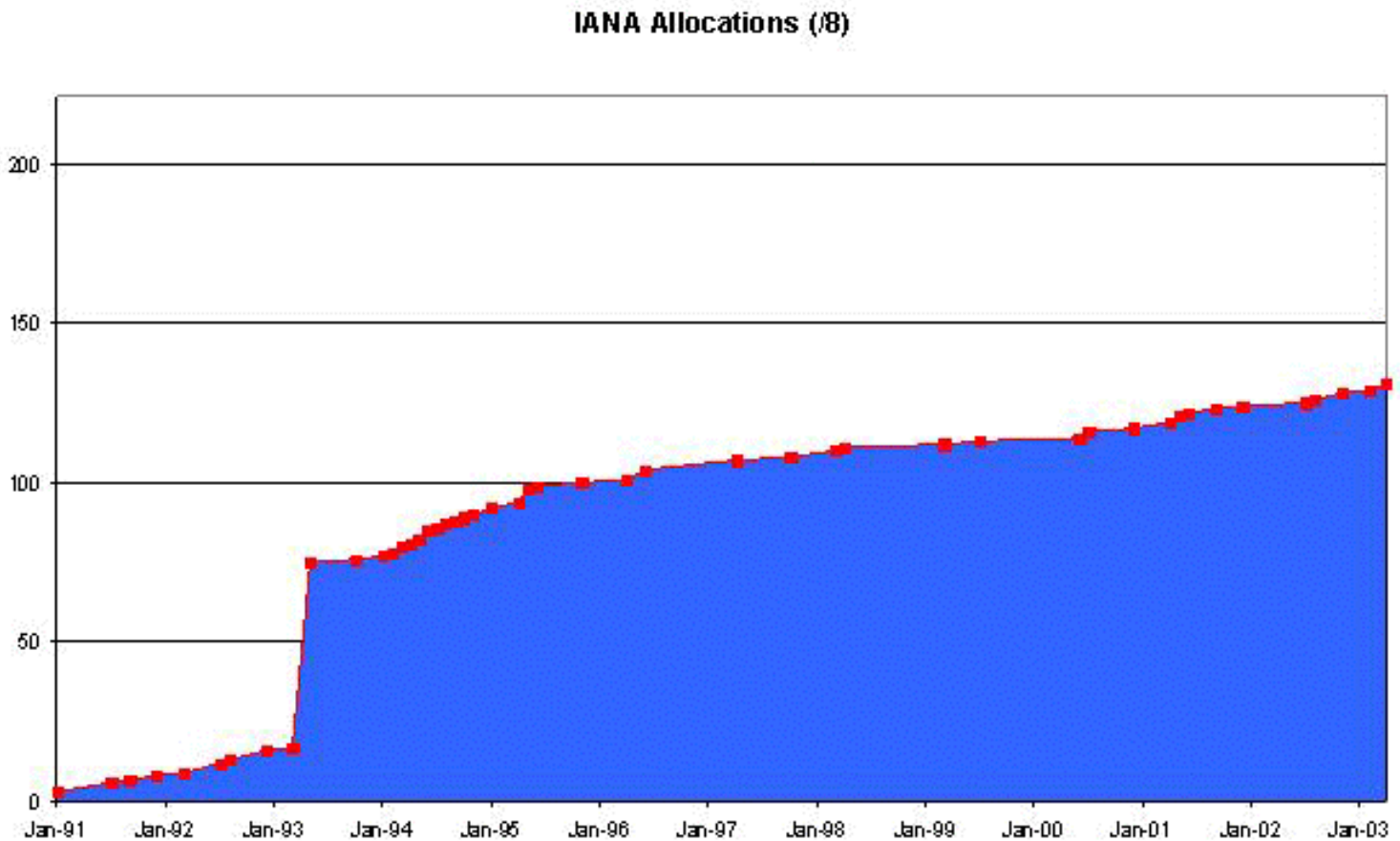
- Why a new version for IP?
- IPv6 Protocol
- Address formats, addressing architecture
- Protocols associated to IPv6
- IPv6 support in the DNS (DNSv6)
- IPv6 Mobility
- IPv6 Security with IPsec
- Early experiences and deployments
- IPv6 and OS/applications
- IPv4 / IPv6 integration
- Equipment Configuration
- Conclusion

# Why a new version for IP ?

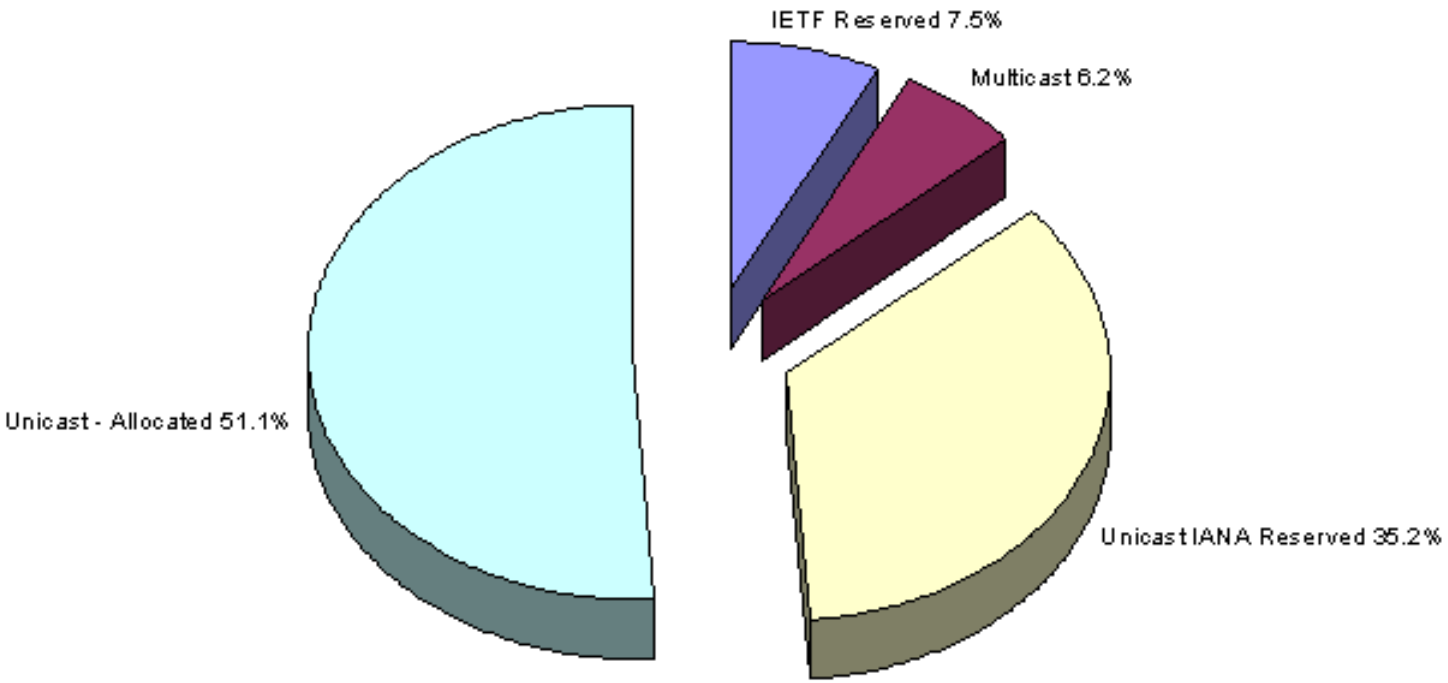
# Historical facts

- 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 address space consumption



# IPv4 address space consumption /2



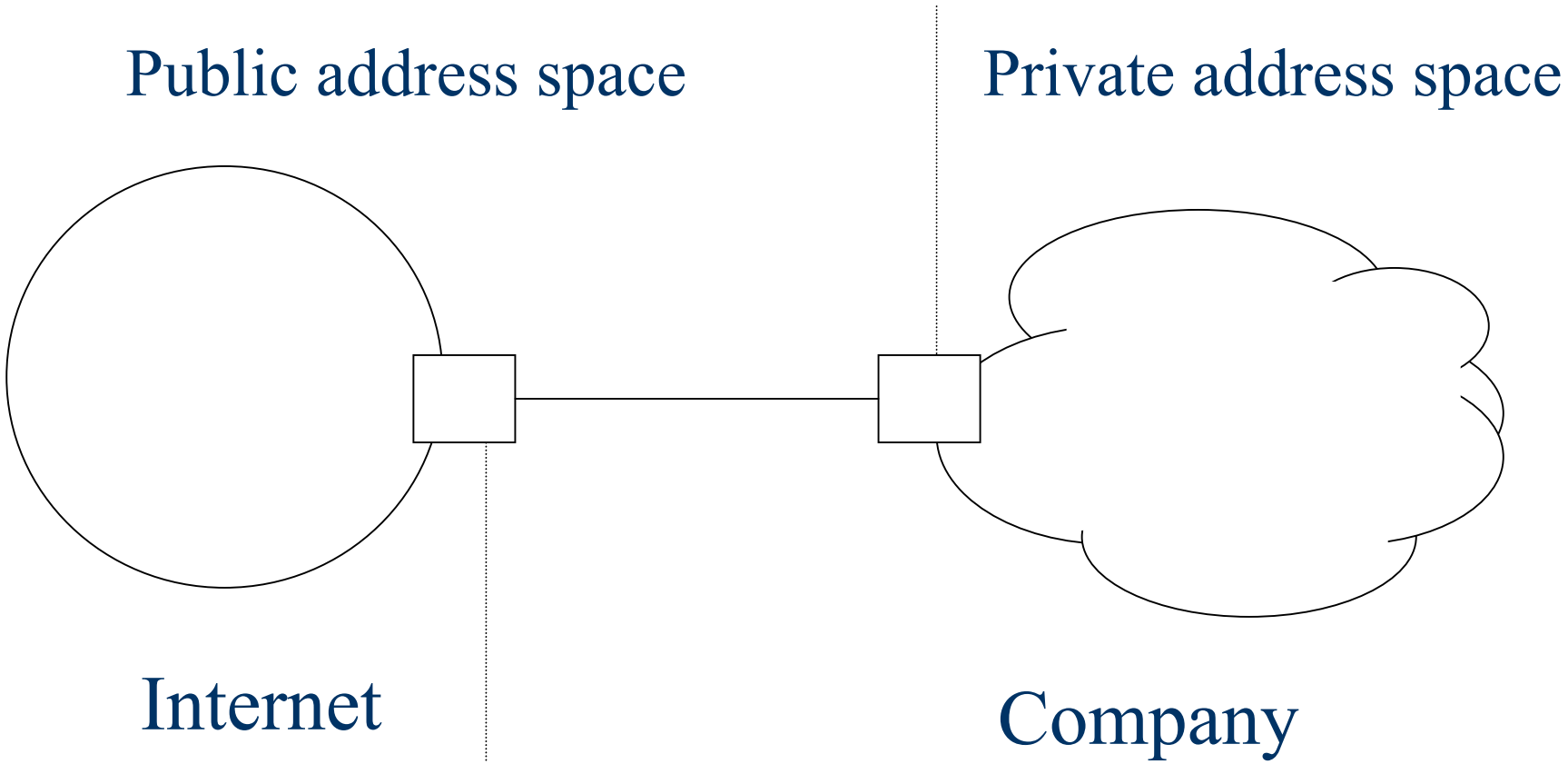
# Emergency measures

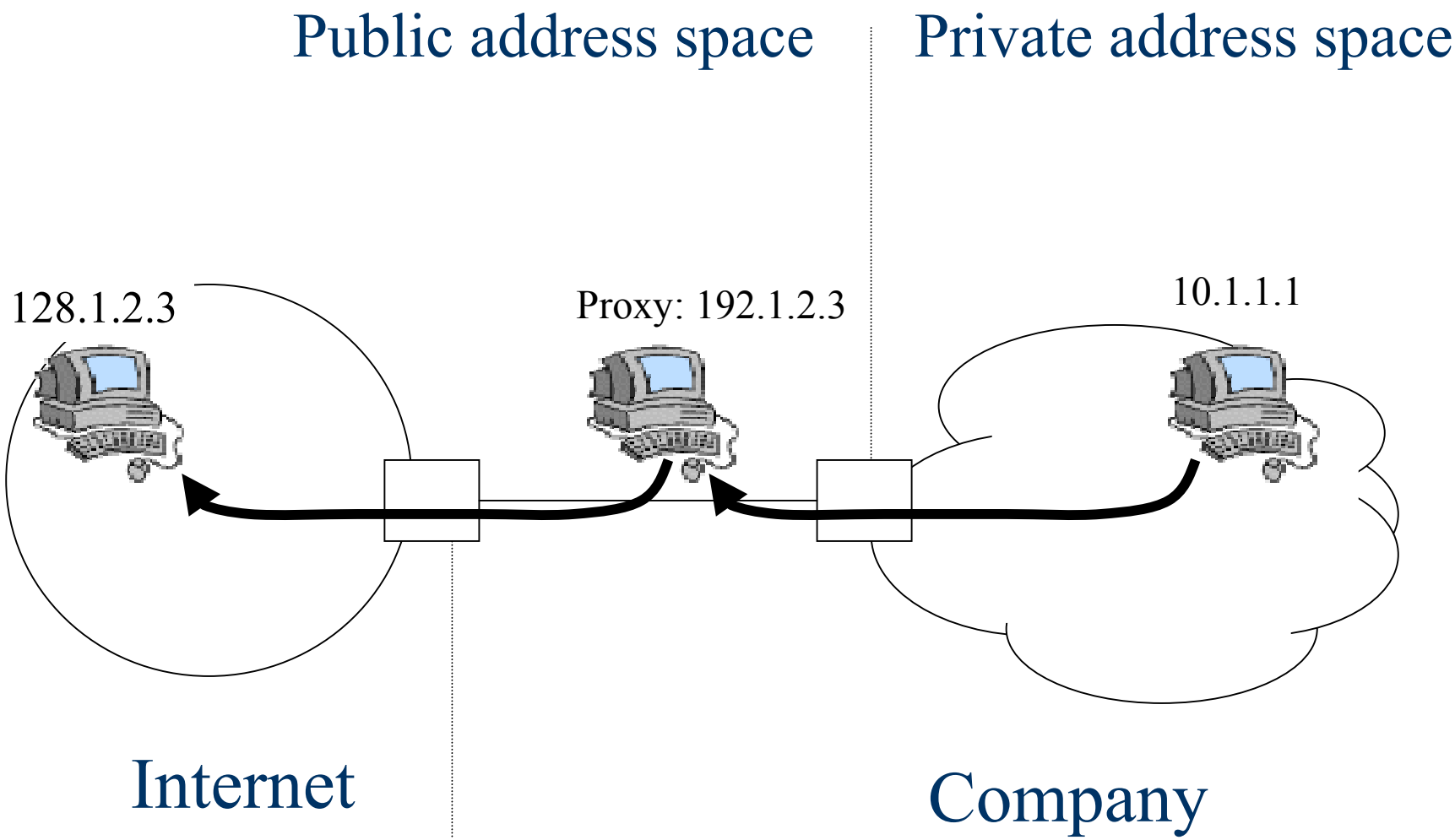
- Allocate exceptionally class B addresses
- Re-use class C address space
- CIDR (*Classless Internet Domain Routing*)
  - RFC 1519 (PS)
  - network address = prefix/prefix length
  - less address waste
  - recommend aggregation (reduce routing table length)

# Emergency Measures: Private Addresses (RFC 1918 BCP)

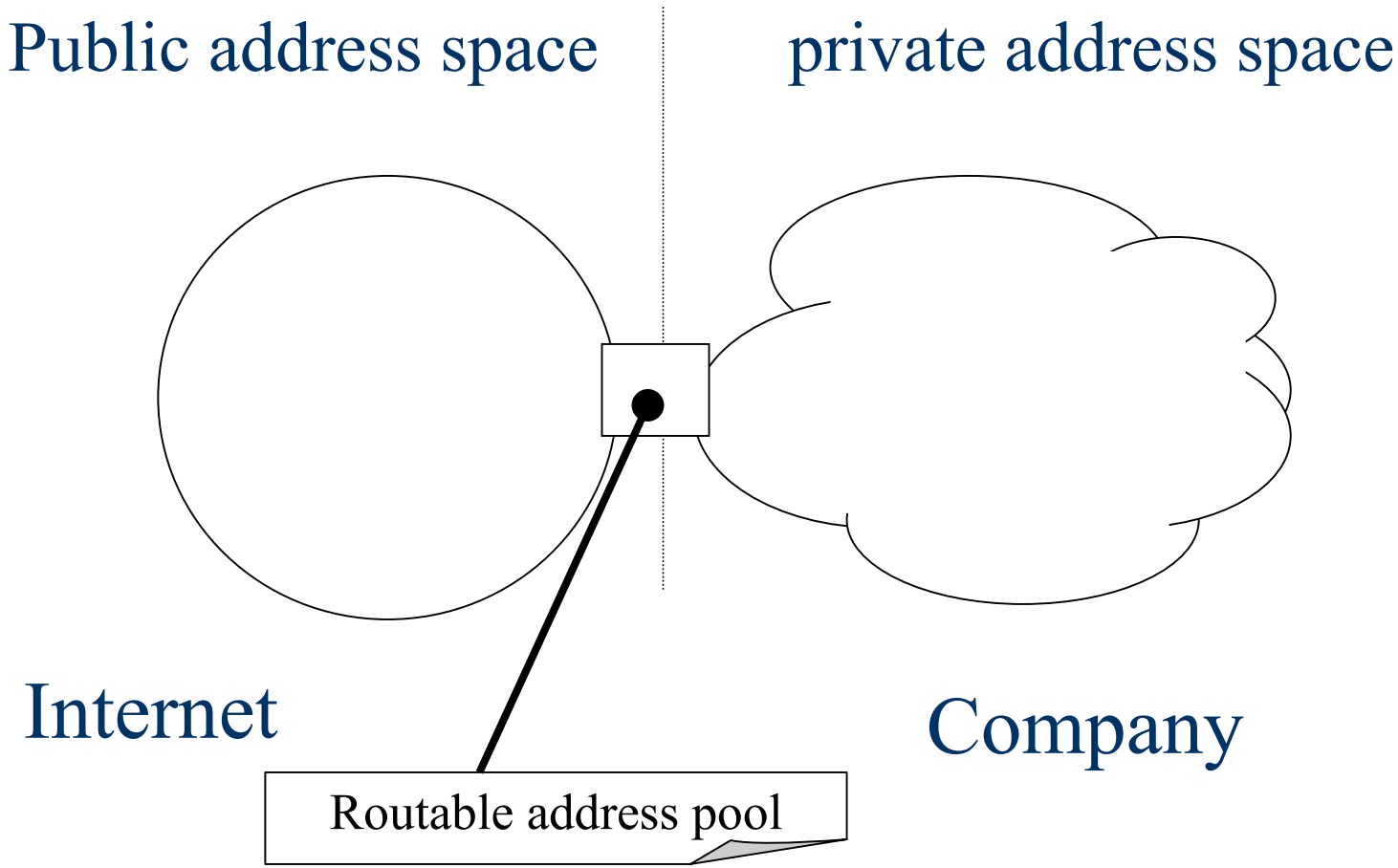
- Allow private addressing plans
- Addresses are used internally
- Similar to security architecture with  firewall
- Use of proxies or NAT to go outside
  - RFC 1631, 2663 and 2993
- NAPT is the most commonly used of NAT variations

# Emergency Measures (continued)

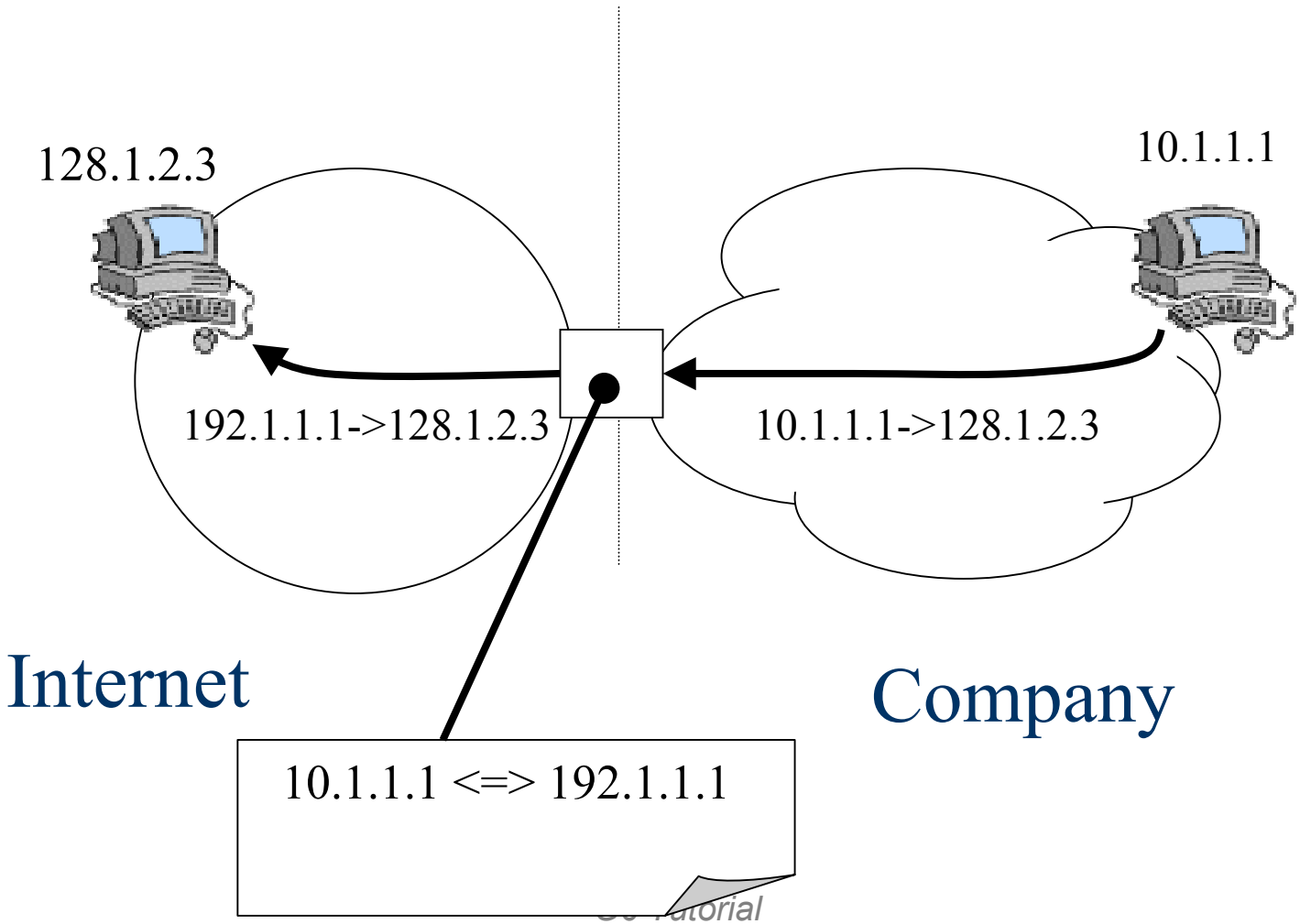



Public address space
Private address space
128.1.2.3
Proxy: 192.1.2.3
10.1.1.1
Internet
Company

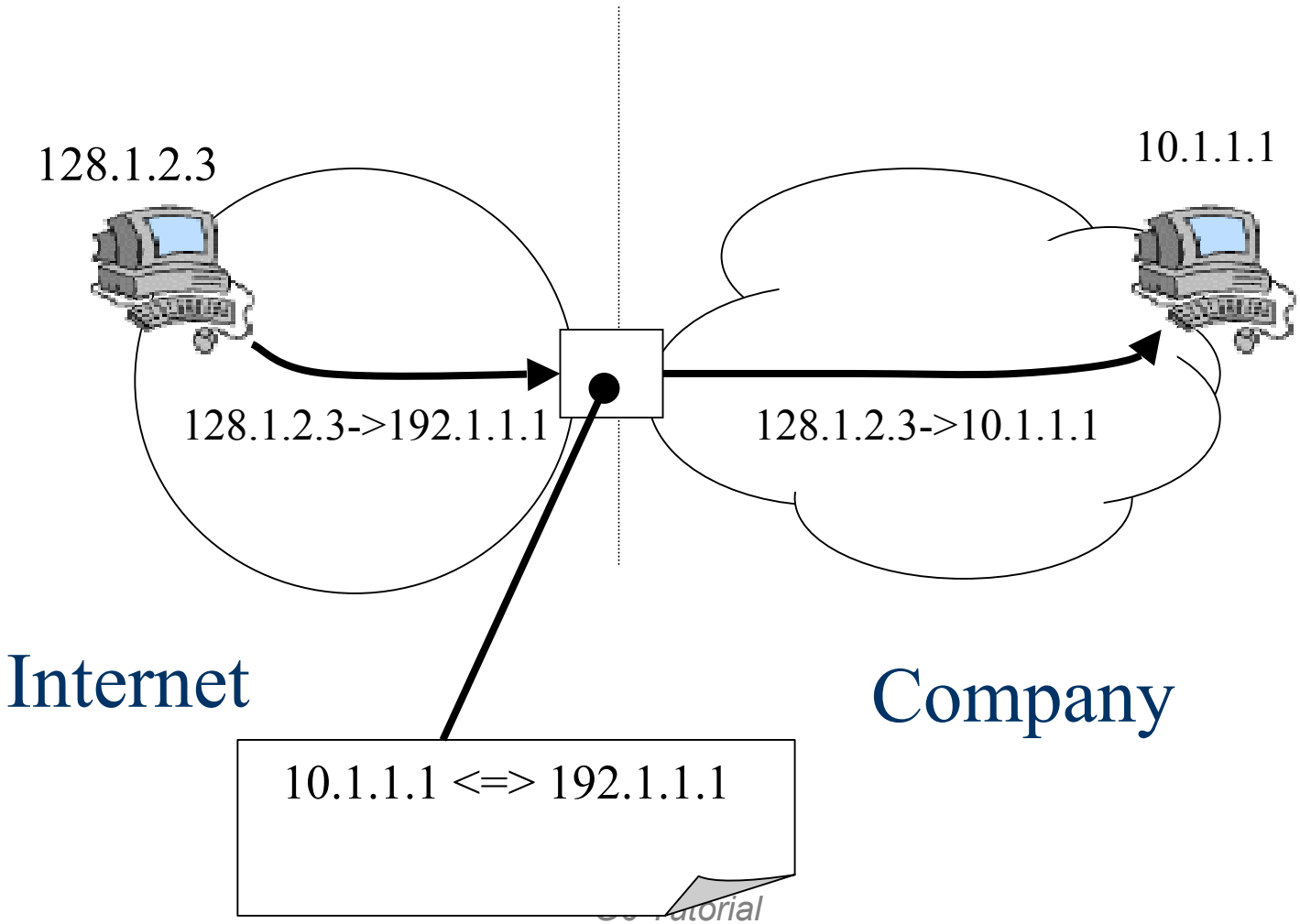
# Network Address Translation



# NAT (continued)



# NAT (continued)



# NAT (continued)

- Advantages:
  - Reduce the need of official addresses
  - Ease the internal addressing plan
  - Transparent to some applications
  - Security ?
- Disadvantages:
  - Translation sometime complex (e.g. FTP)
  - Does not scale
  - Introduce states inside the network:
    - Multihomed networks
  - Breaks the end-to-end paradigm
  - Security with IPsec

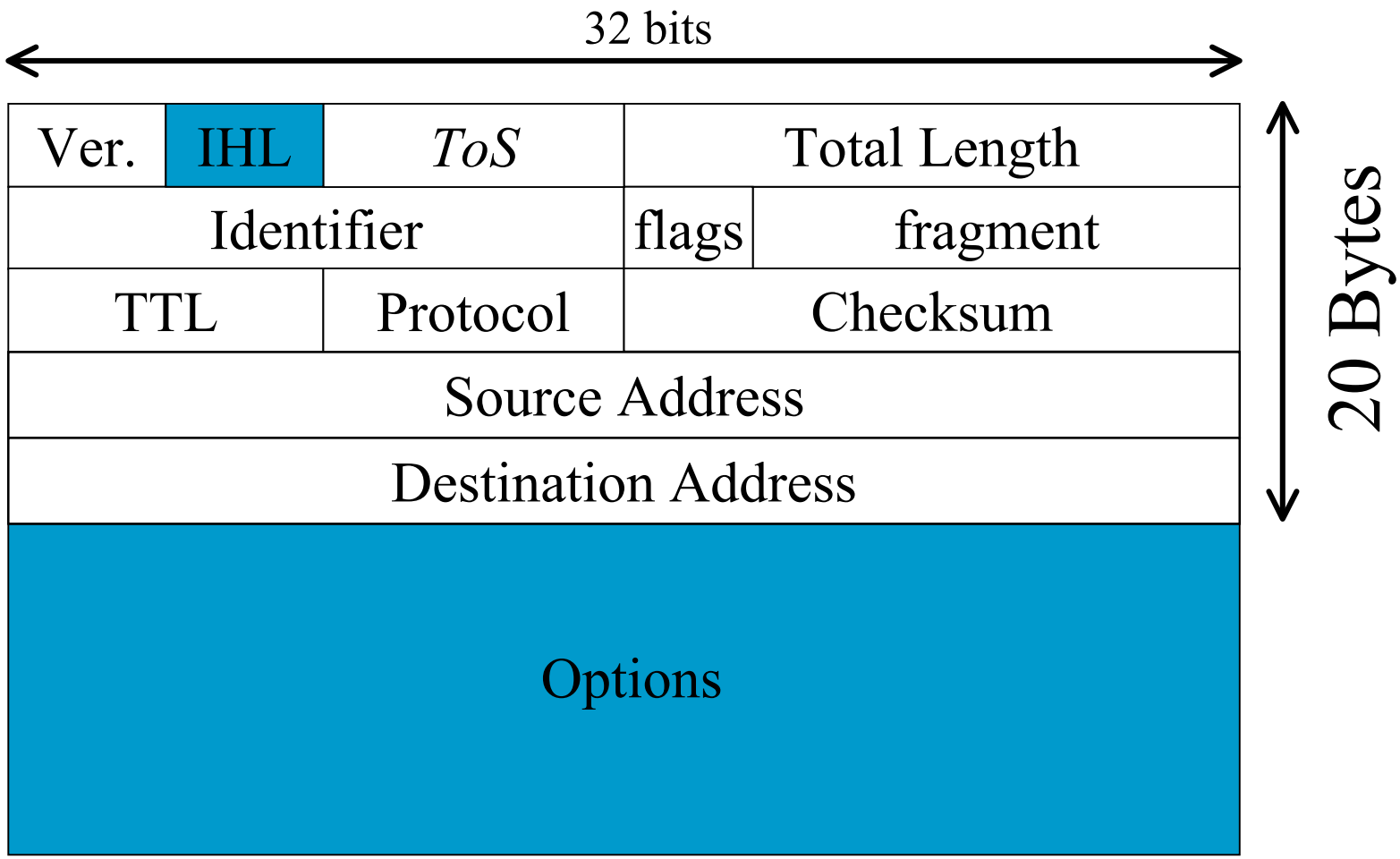
=> Should be reserved for small sites in Client/Server mode

# Emergency Measures (continued)

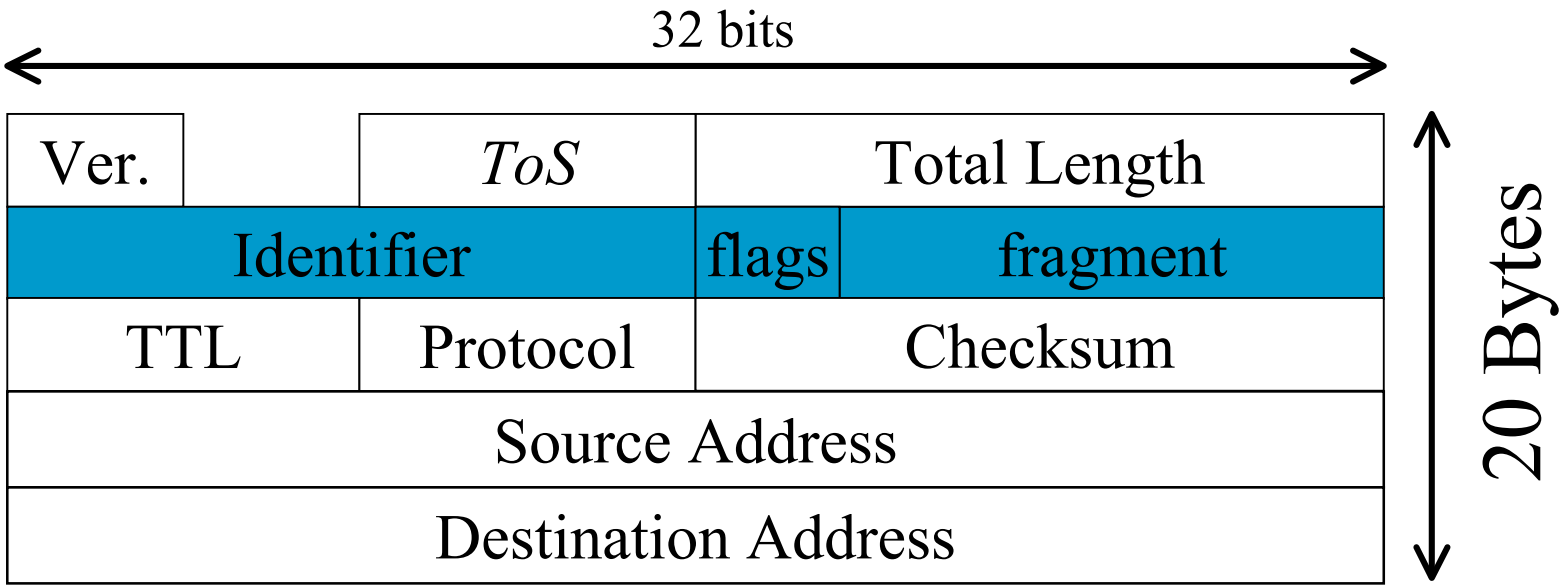
- These emergency measures give time to develop a new version of IP, named IPv6
- IPv6 keeps principles that have made the success of IP
- Corrects what was wrong with the current version (v4)
- BUT are emergency measures enough?

# IPv6 Protocol
## (RFC 2460 DS)

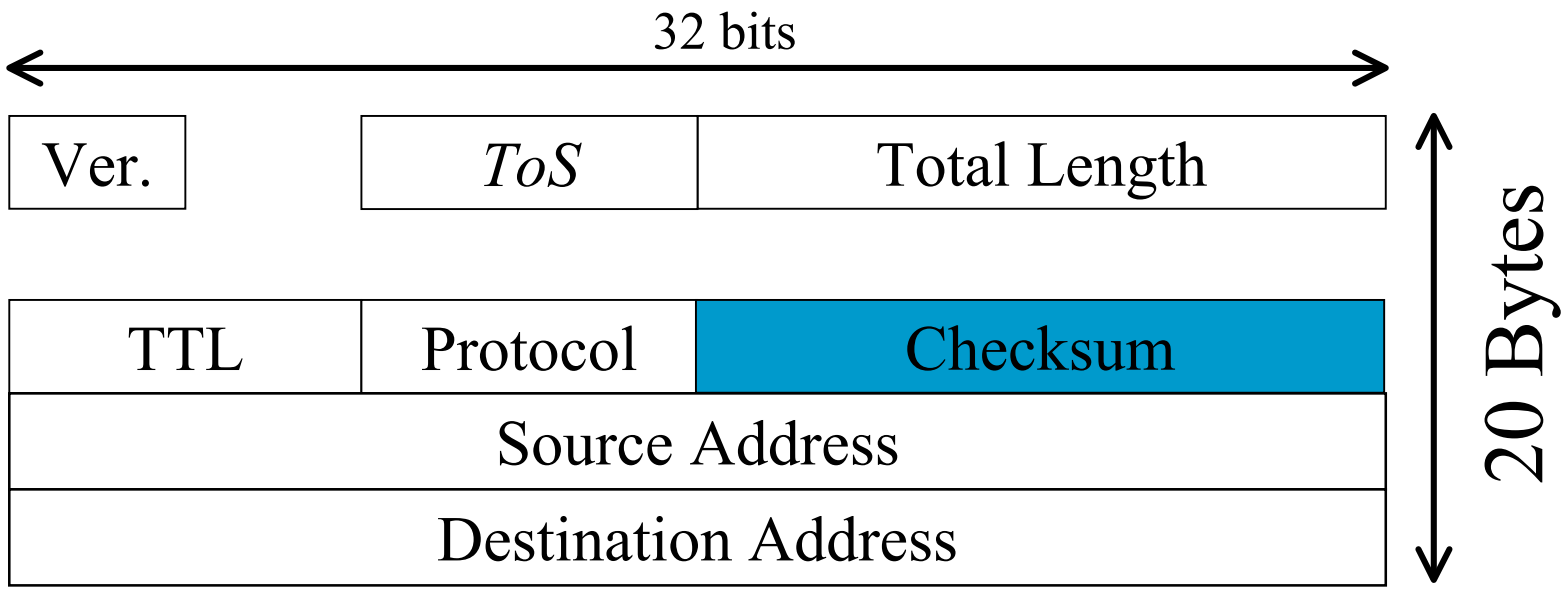
# IPv4 Header


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

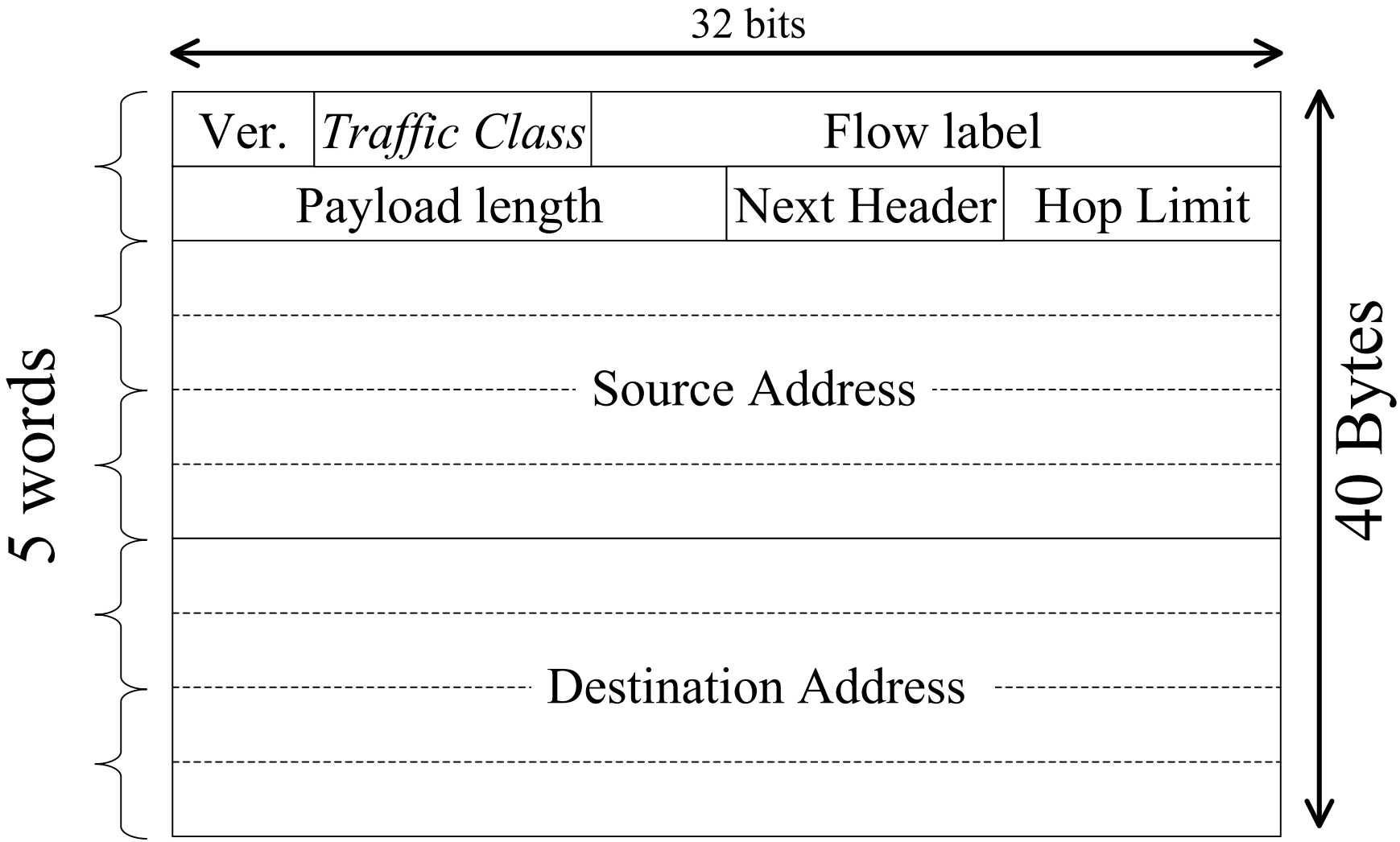
# IPv4 Header


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

# IPv4 Header


32 bits
Ver.
ToS
Total Length
TTL
Protocol
Checksum
Source Address
Destination Address
20 Bytes

# IPv6: Header simplification



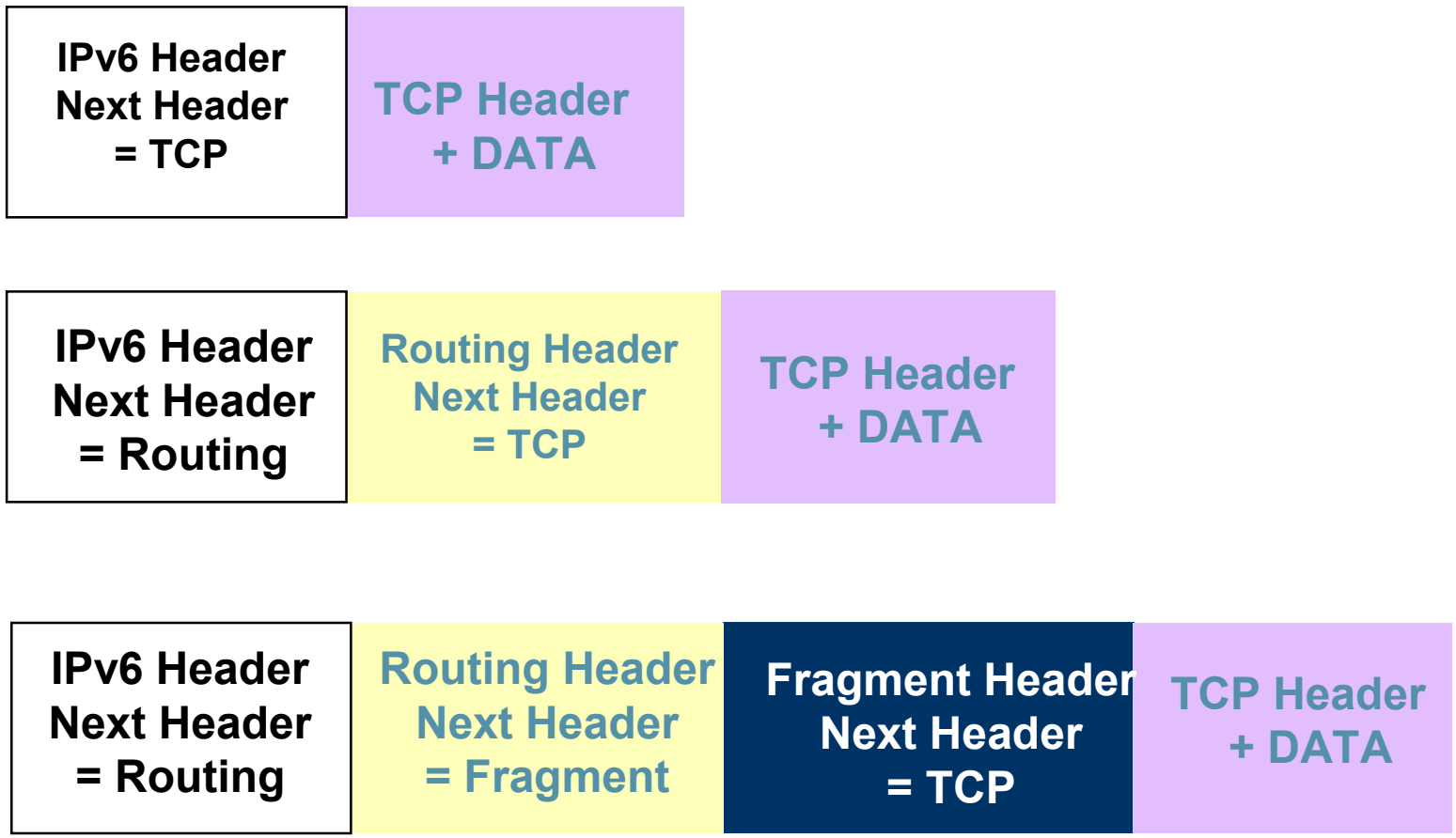
# Is it enough for the future ?

- Address length
  - Between 1 564 and 3 911 873 538 269 506 102 addresses by $m^2$
  - Justification of a fix address length
- Hop Limit
  - Should not be a problem
- Payload Length
  - Use Jumbogram for specific cases

# QoS support in IPv6

- The **Traffic Class field:** used as in IPv4
  - Work done in diffserv wg (closed): RFCs 2474, 2475, 2597, 3260, …
- The **Flow Label field:** designed to enable classification of packets belonging to a specific flow
  - **A flow** is a sequence of packets that should receive specific non-default handling from the network
  - Intuitively: 5-tuple of the same source/destination address/port and transport protocol values
  - Without the flow label the classifier must use transport next header value and port numbers
    - Less efficient (need to parse the option headers)
    - May be impossible (fragmentation or IPsec ESP)
  - Further info:
    - http://www.ietf.org/internet-drafts/draft-ietf-ipv6-flow-label-09.txt (RFC XXXX (PS))

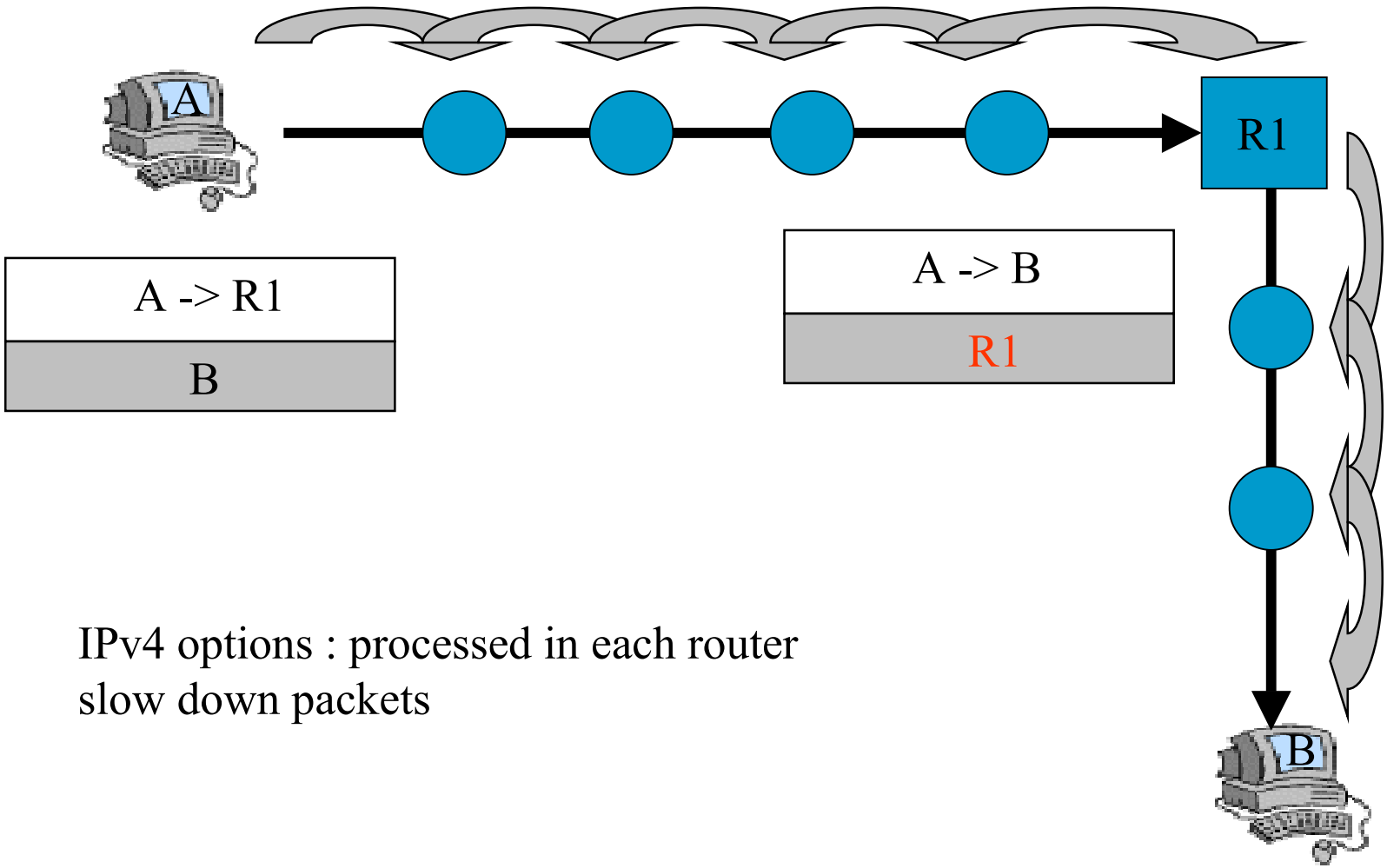
# IPv6: Optional headers



# IPv6: Optional extensions

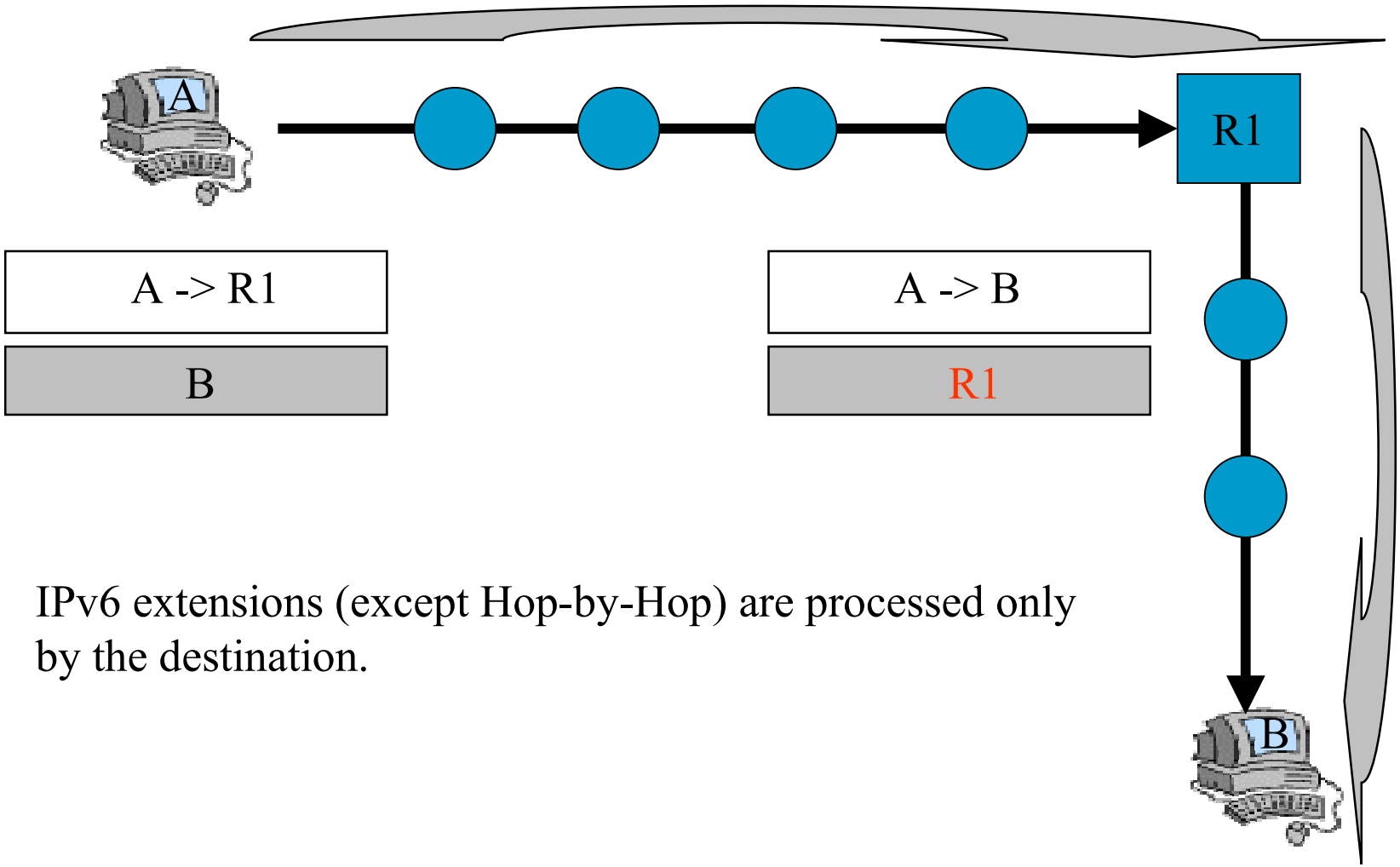
- Hop-by-hop (jumbogram, router alert)
  - Always the first extension
  - Replace IPv4 options,
  - Analyzed by every router.
- Destination
- Routing (loose source routing)
- Fragmentation
- Authentication
- Security

# v4 options vs. v6 extensions



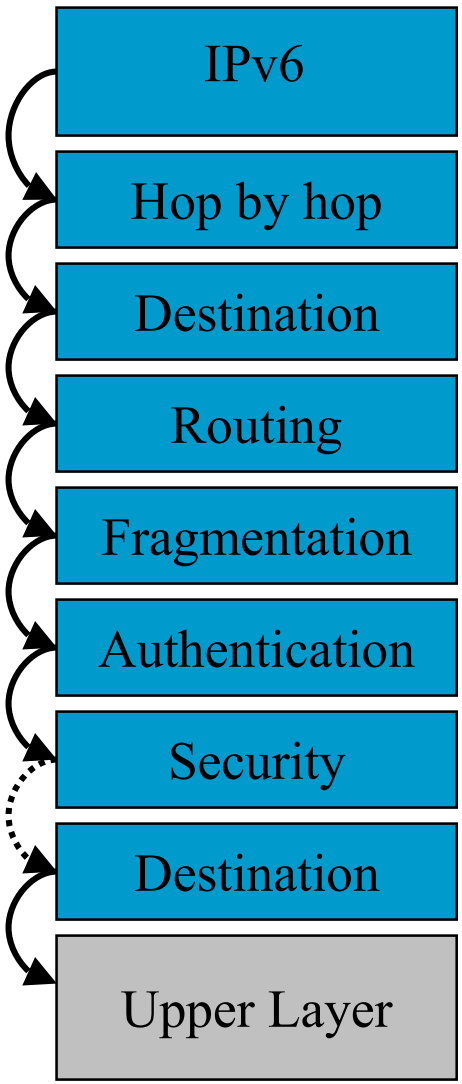

IPv4 options : processed in each router
slow down packets

# v4 options vs. v6 extensions




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

# Order is important




- Hop by hop: Processed by every router
- Destination: Processed by routers listed in Routing extension
- Routing: List of routers to cross
- Fragmentation: Processed by the destination
- Authentication: After reassembling the packet
- Security: Cipher the content of the remaining information
- Destination: Processed only by the destination

# IPv6 Addressing

# Addressing scheme

- RFC 3513 (obsoletes RFC 2373)
- RFC 3587 (obsoletes RFC 2374)
- 128 bit long addresses
  - Allow hierarchy
  - Flexibility for network evolutions
- Use CIDR principles:
  - Prefix / prefix length
    - 2001:660:3003::/48
    - 2001:660:3003:2:a00:20ff:fe18:964c/64
  - Aggregation reduces routing table size
- Hexadecimal representation
- Interfaces have several IPv6 addresses

# Textual Address Format

- Base format (a 16-byte **Global IPv6 Address) :**
  - **2001:0660:3003:0001:0000:0000:6543:210F**
- Compact Format:

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

  - In order to avoid ambiguity, "::" can occur only once

# 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 Addresses** | **001** | **1/8** | **[RFC2374, RFC 3587]** |
| **Unassigned** | **010** | **1/8** | |
| **Unassigned** | **011** | **1/8** | |
| **Unassigned** | **100** | **1/8** | |
| **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 Unicast Addresses** | **1111 1110 10** | **1/1024** | |
| **Site-Local Unicast Addresses** | **1111 1110 11** | **1/1024** | |
| **Multicast Addresses** | **1111 1111** | **1/256** | |

# IPv6 Addresses

- Loopback ::1
- Link local FE80:….
- Site local FEC0:…
- Global
  - 6bone: 3FFE:…
  - Official: 2001:…

---

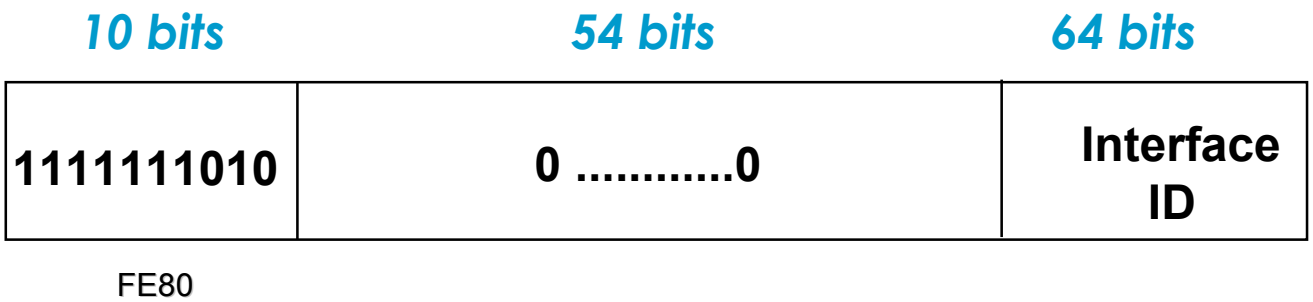
  - IPv4 mapped
  - 6to4: 2002::…

specific to IPv4/IPv6 integration

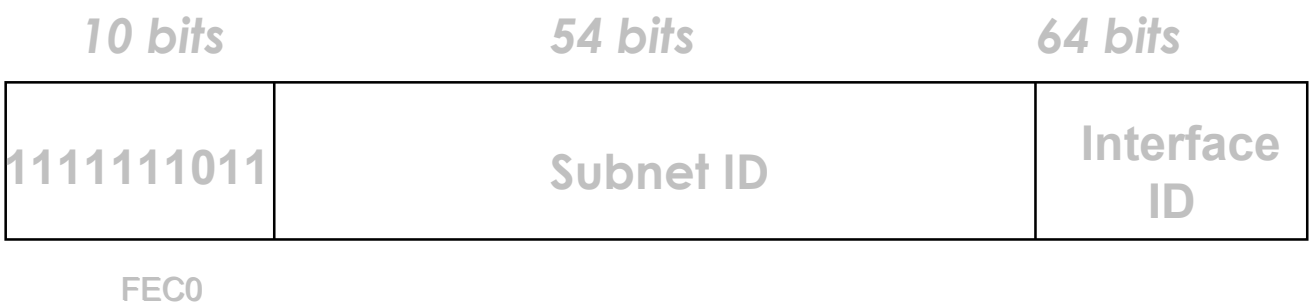
- Unicast
- Multicast
- Anycast

# Local Addresses

**Link-local**

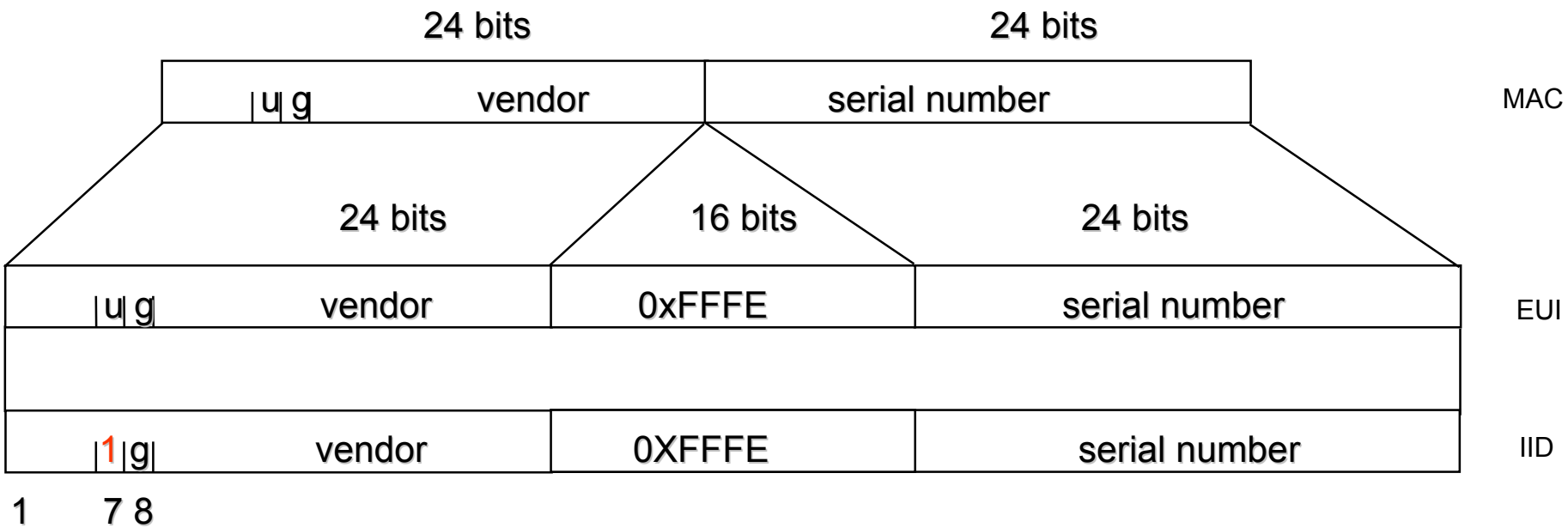



**Site-local** (in the process of being deprecated)



# 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)



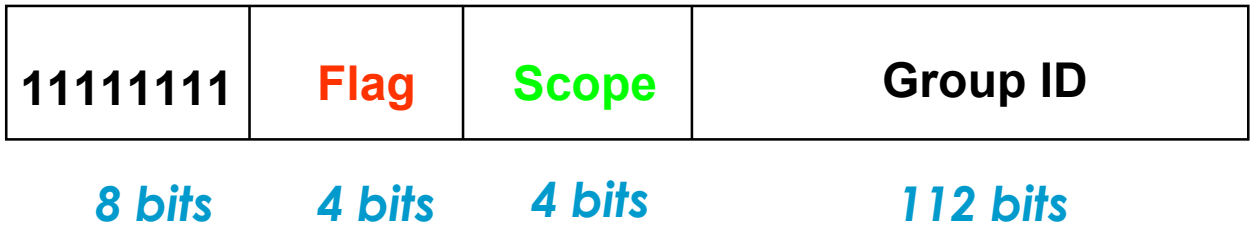
# Interface Identifier (2)

- Links with non global identifier (e.g., the Localtalk 8 bit node identifier) → fill first left bits with 0
- For links without identifiers, there are different ways to proceed (e.g., tunnels, PPP):
  - Choose the identifier of another interface
  - Random number
  - Manual configuration
- **THEN :** Invert IEEE EUI-64 “u” bit to become an “interface identifier”

# Interface Identifier (3)
## (Privacy issues)

- IEEE 24 bit OUI can be used to identify HW:
  - http://standards.ieee.org/regauth/oui/oui.txt
- Interface Identifier can be used to trace a user:
  - The prefix changes, but the interface ID remains the same,
  - Psychological issue.
- Possibility to change Interface ID (RFC 3041 PS):
  - If local storage, use MD5 algorithm
  - Otherwise draw a random number

# 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 (I-D)*

**Scope**

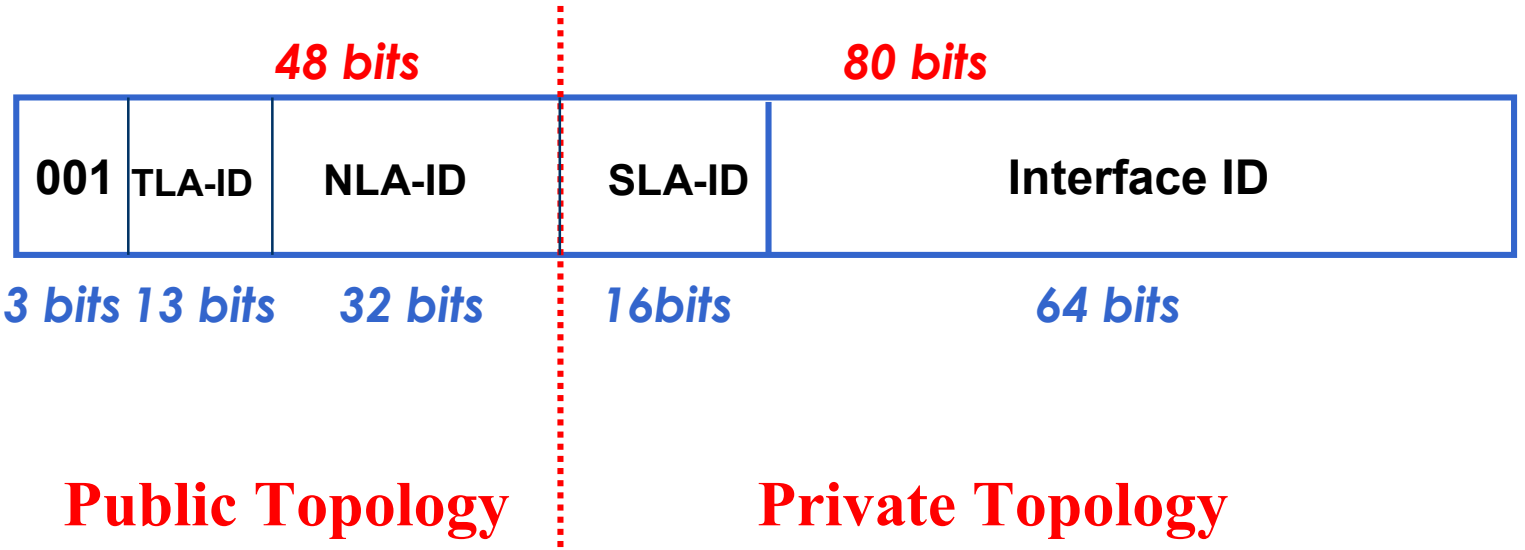
**0 : Reserved**
**1 : Interface-local**
**2 : Link-local**
**3 : Subnet-local**
**4 : Admin-local**
**5 : Site-local**
**8 : Organization-local**
**E : Global**
**F : Reserved**

# Anycast Addresses (RFC 3513)

- Anycast addresses have been defined for *routers only* so far
- **It cannot be distinguished from a Unicast address**

- Reserved anycast addresses are defined in RFC 2526
- Subnet anycast router address is :
  - Subnet ID::0/subnet prefix length

# IPv6 Addresses (continued)

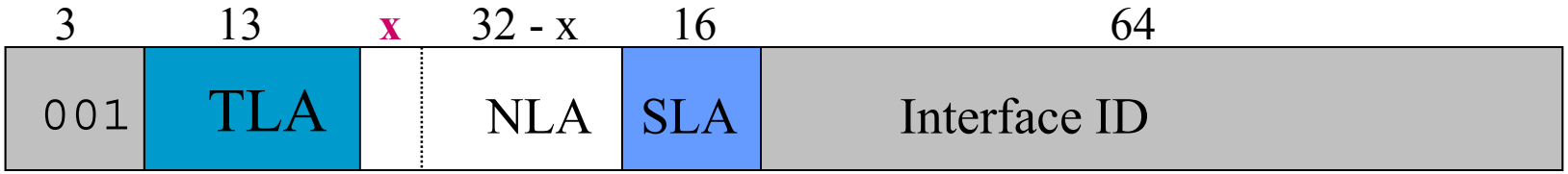



**TLA : Top Level Aggregator => (/16)**

**NLA : Next Level Aggregator => (/48)**

**SLA : Site Level Aggregator => (/64)**

# RFC 2471: Aggregatable Test Addresses

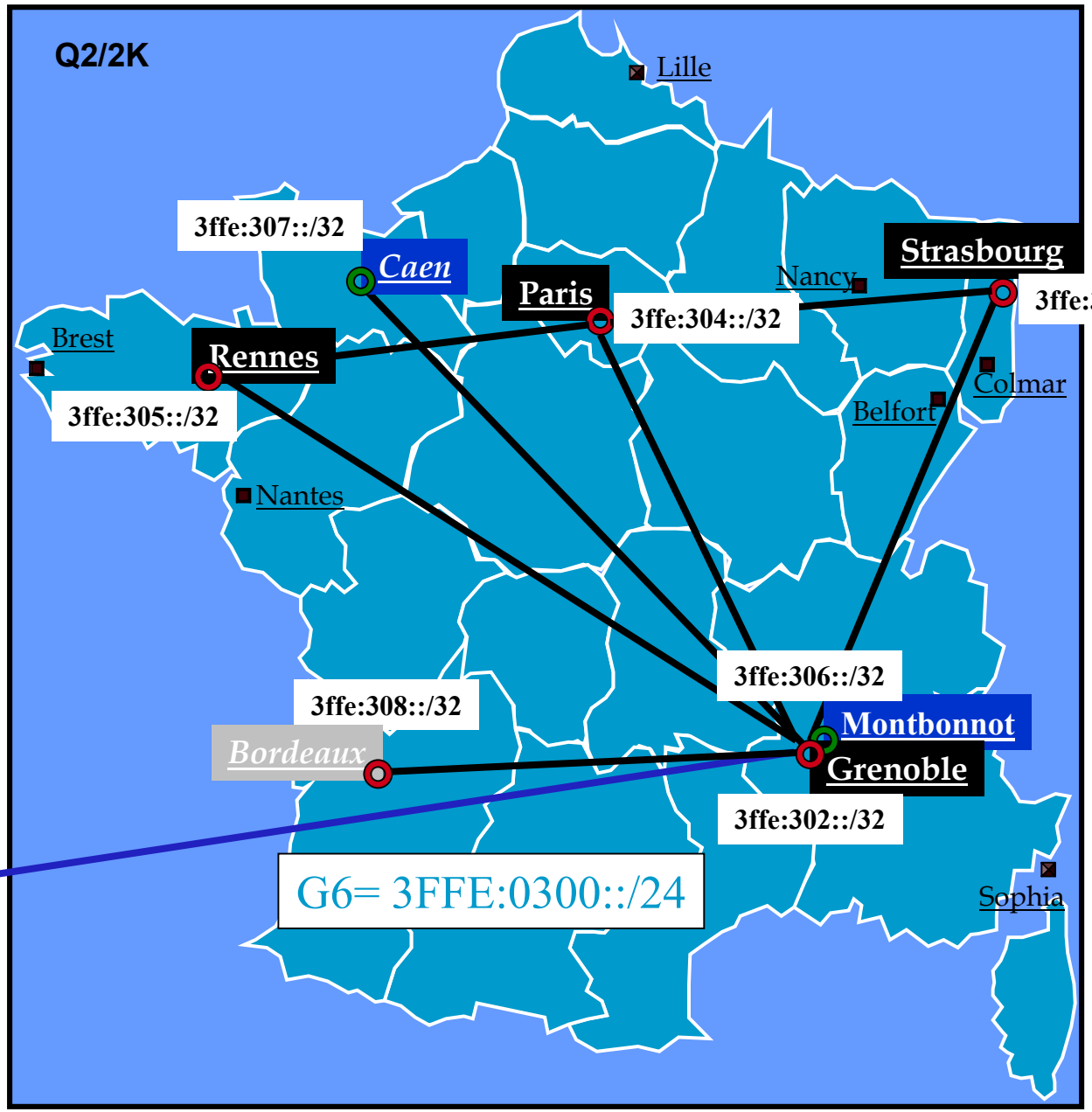



- Used in the 6bone
- TLA value is 0x1FFE => Prefix = 3FFE::/16
- pTLA in the NLA part assigned by *ngtrans* wg

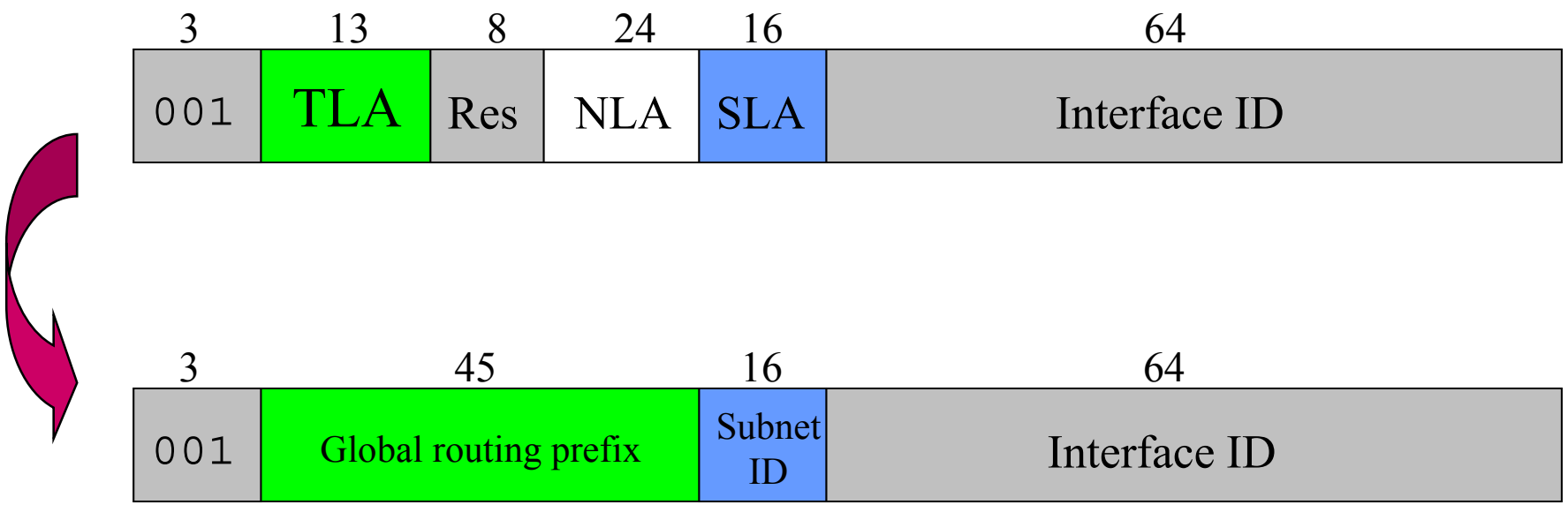
http://www.6bone.net/6bone_pTLA_list.html

| | | |
|---|---|---|
| 49 x ::/24 | | |
| | INNER/US-VA | 3FFE:0000::/24 |
| | TELEBIT/DK | 3FFE:0100::/24 |
| | SICS/SE | 3FFE:0200::/24 |
| | G6/FR | 3FFE:0300::/24 |
| | JOIN/DE | 3FFE:0400::/24 |
| 45 x ::/28 | | 3FFE:8xyz::/28 |
| 27 x ::/32 | | 3FFE:4xyz::/32 (2003/11/21) |

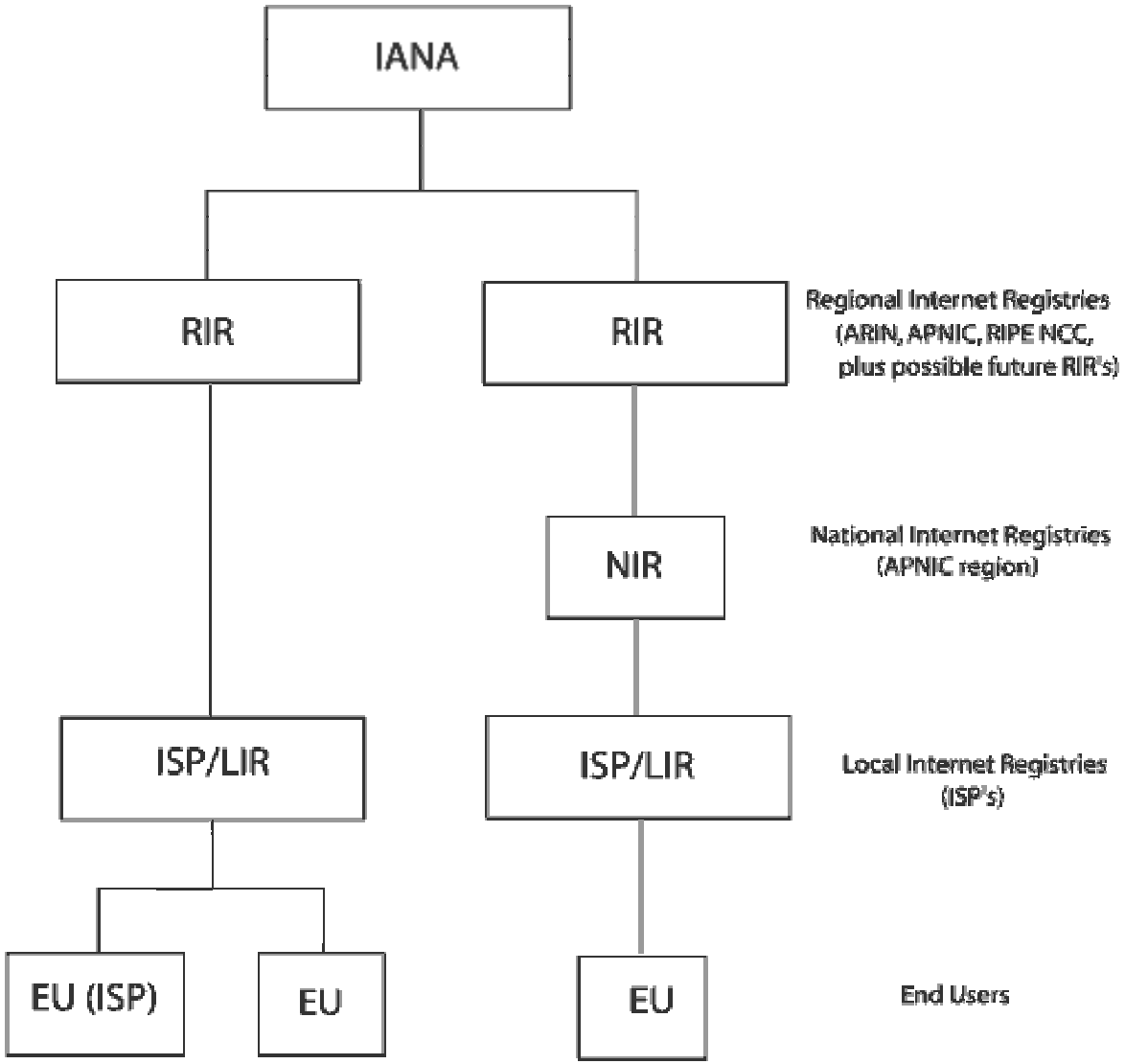
# G6bone Addressing Scheme



# RFC 3587: Aggregatable Global Unicast (obsoletes RFC 2374)



# Production Addressing Scheme



# Production Addressing Scheme (2)

Source : http://www.iana.org/assignments/ipv6-tla-assignments

TLA Identifier Assignments
---

TLA Identifiers are defined in [RFC2374] and are assigned from the Format Prefix (FP) 001 (binary) in [RFC2373].

TLA ID assignments are listed below.

| IPv6 Prefix | FP | TLA Binary | Value | TLA Hex Assignment |
|---|---|---|---|---|
| 2000::/16 | 001 | 0 0000 0000 0000 | 0x0000 | Reserved |
| 2001::/16 | 001 | 0 0000 0000 0001 | 0x0001 | Sub-TLA Assignments [RFC2450] |
| 2002::/16 | 001 | 0 0000 0000 0010 | 0x0002 | "6to4" [RFC3056 et 3068] |
| 3FFE::/16 | 001 | 1 1111 1111 1110 | 0x1FFE | 6bone Testing [RFC2471] |
| 3FFF::/16 | 001 | 1 1111 1111 1111 | 0x1FFF | Reserved |

Note: Hex values are right justified.

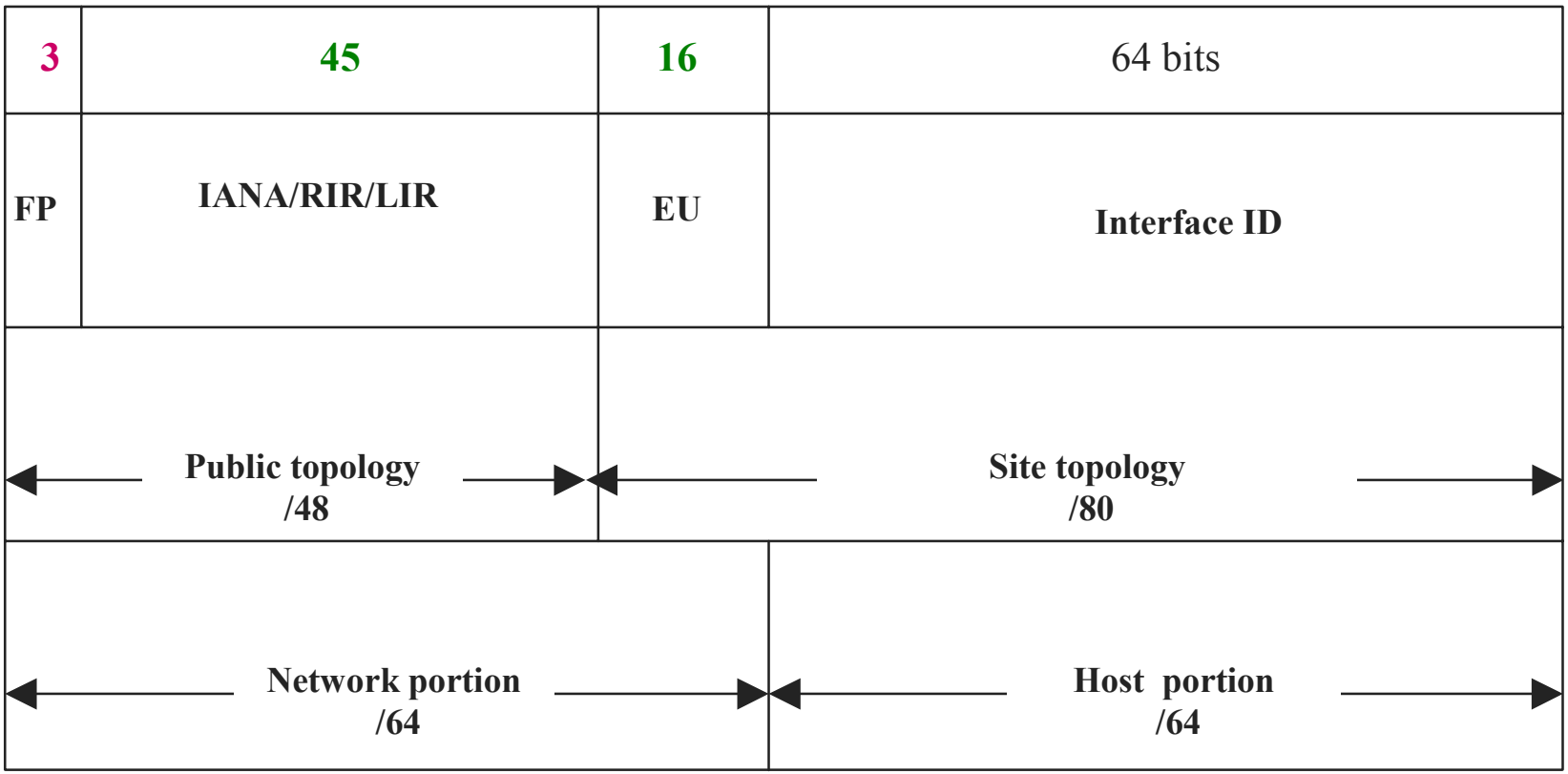
All TLA ID values not listed above are reserved.

# Production Addressing Scheme (3)

| IPv6 Prefix sub-TLA | Binary Values | Allocated to | Date |
|---|---|---|---|
| 2001:0000::/23 | 0000 000X XXXX X | IANA | Jul 99 |
| 2001:0200::/23 | 0000 001X XXXX X | APNIC | Jul 99 |
| 2001:0400::/23 | 0000 010X XXXX X | ARIN | Jul 99 |
| 2001:0600::/23 | 0000 011X XXXX X | RIPE NCC | Jul 99 |
| 2001:0800::/23 | 0000 100X XXXX X | RIPE NCC | May 02 |
| 2001:0A00::/23 | 0000 101X XXXX X | RIPE NCC | Nov 02 |
| 2001:0C00::/23 | 0000 110X XXXX X | APNIC | May 02 |
| 2001:0E00::/23 | 0000 111X XXXX X | APNIC | Jan 03 |
| 2001:1000::/23 | 0001 000X XXXX X | (future assignment) | |
| 2001:1200::/23 | 0001 001X XXXX X | LACNIC | Nov 02 |
| 2001:1400::/23 | 0001 010X XXXX X | RIPE NCC | Feb 03 |
| 2001:1600::/23 | 0001 011X XXXX X | RIPE NCC | Jul 03 |
| 2001:1800::/23 | 0001 100X XXXX X | ARIN | Apr 03 |
| . . . | | | |
| . . . | | | |
| . . . | | | |
| 2001:FE00::/23 | 1111 111X XXXX X | (future assignment) | |

Where "X" indicates "0" or "1".
All other Sub-TLA ID values not listed above are reserved.

# Production Addressing Scheme (4)



# RIR allocations

- Started July '99
- New allocated prefix length since July 1th 2002, ::/32 instead of ::/35
- Allocated prefixes (up to 6 Feb. 2004) = 528
  - http://www.ripe.net/ripencc/mem-services/registration/ipv6/ipv6allocs.html

  – *APNIC*
    - **133 prefixes**
    - **within 2001:{02, 0C, 0E}00::/23**

  – *ARIN*
    - **95 prefixes**
    - **within 2001:{04, 18}00::/23**

  – *RIPE-NCC*
    - **294 prefixes**
    - **within 2001:{06, 08, 0A, 14, 16}00::/23**

  – *LACNIC*
    - **6 prefixes**
    - **within 2001:1200::/23**

# Initial RIR allocation Policy & Procedure

- Get the RIPE documents [246-250, 256, 261, 267, 274, 275, 280-282]
  - http://www.ripe.net/ripe/docs/ipv6.html
- Criteria: RIPE-267
  - http://www.ripe.net/ripe/docs/ipv6plicy.html
- To qualify for an initial allocation of IPv6 address space, an organization must:
  - be an LIR : *not be an end site*
  - plan to provide IPv6 connectivity to organizations to which it will assign /48s, by advertising that connectivity through its single aggregated address allocation (/32 prefix)

  **and**
  - have a plan for making at least 200 x /48 assignments to other organizations within two years.

# Exemple d'adressage du service IPv6 dans Renater-3

## le service 6R3

# IPv6 associated Protocols

# New Protocols

- New features specified in IPv6 Protocol (RFC 2460 DS)
- 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)

# New Protocols (2)

- MLD (Multicast Listener Discovery) (RFC 2710 PS)
  - 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 (3): Comparison with IPv4

- It is the synthesis of:
  - ARP
  - R-Disc
  - ICMP 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

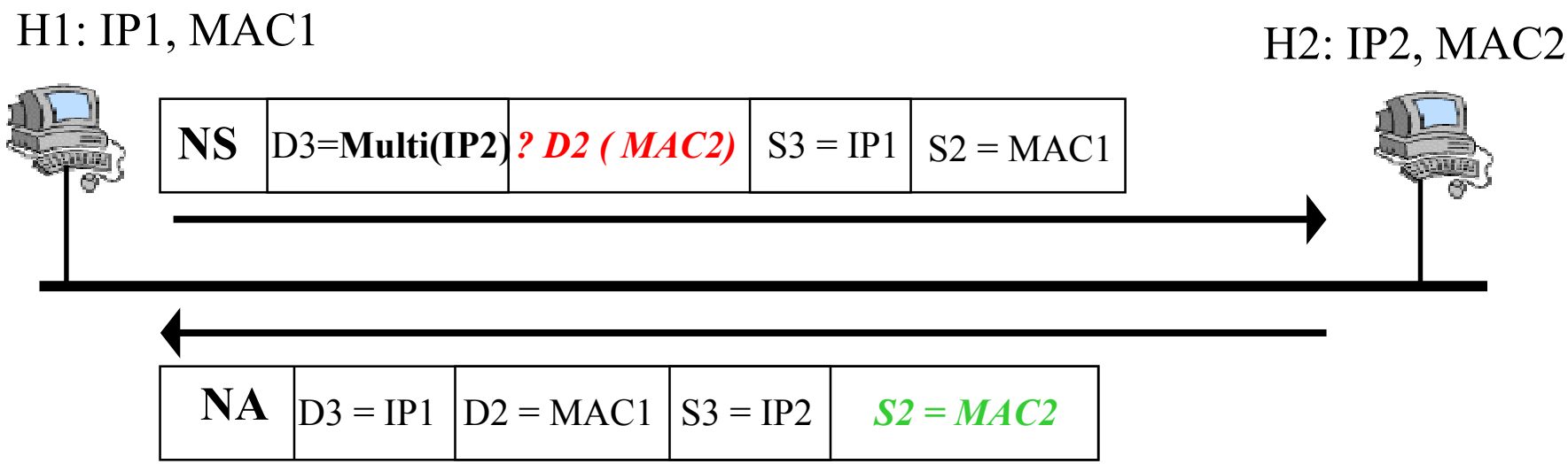
# Address Resolution

- Find the mapping: Dst IP @ ➔ Link-Layer (MAC) @
- Recalling IPv4 & ARP
  - ARP Request is broadcasted
    - e.g. `ethernet @: FF-FF-FF-FF-FF-FF`
    - Btw, it contains the Src's LL @
  - ARP Reply is sent in unicast to the Src
    - It contains the Dst's LL @

# 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

# 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

# Auto-configuration

- Hosts should be plug & play
- Use ICMPv6 messages (Neighbor Discovery)
- When booting, the host asks for network parameters:
  - IPv6 prefix(es)
  - default router address(es)
  - hop limit
  - (link local) MTU
  - ...

# Auto-configuration (continued)

- Only routers have to be manually configured
  - but work on **prefix delegation** is in progress
    *(draft-ietf-ipv6-prefix-delegation-requirement-01.txt)*
- Hosts can get automatically an IPv6 address
  - BUT it is not automatically registered in the DNS
  - If the address is always the same: may be manually registered

⇒ NEED for DNS Dynamic Update
(RFC 2136 PS and RFC 3007 PS) for IPv6
  - Security issues …

# Stateless auto-configuration

- IPv6 Stateless Address Auto-configuration
  - RFC 2462 DS
  - Does not apply to routers

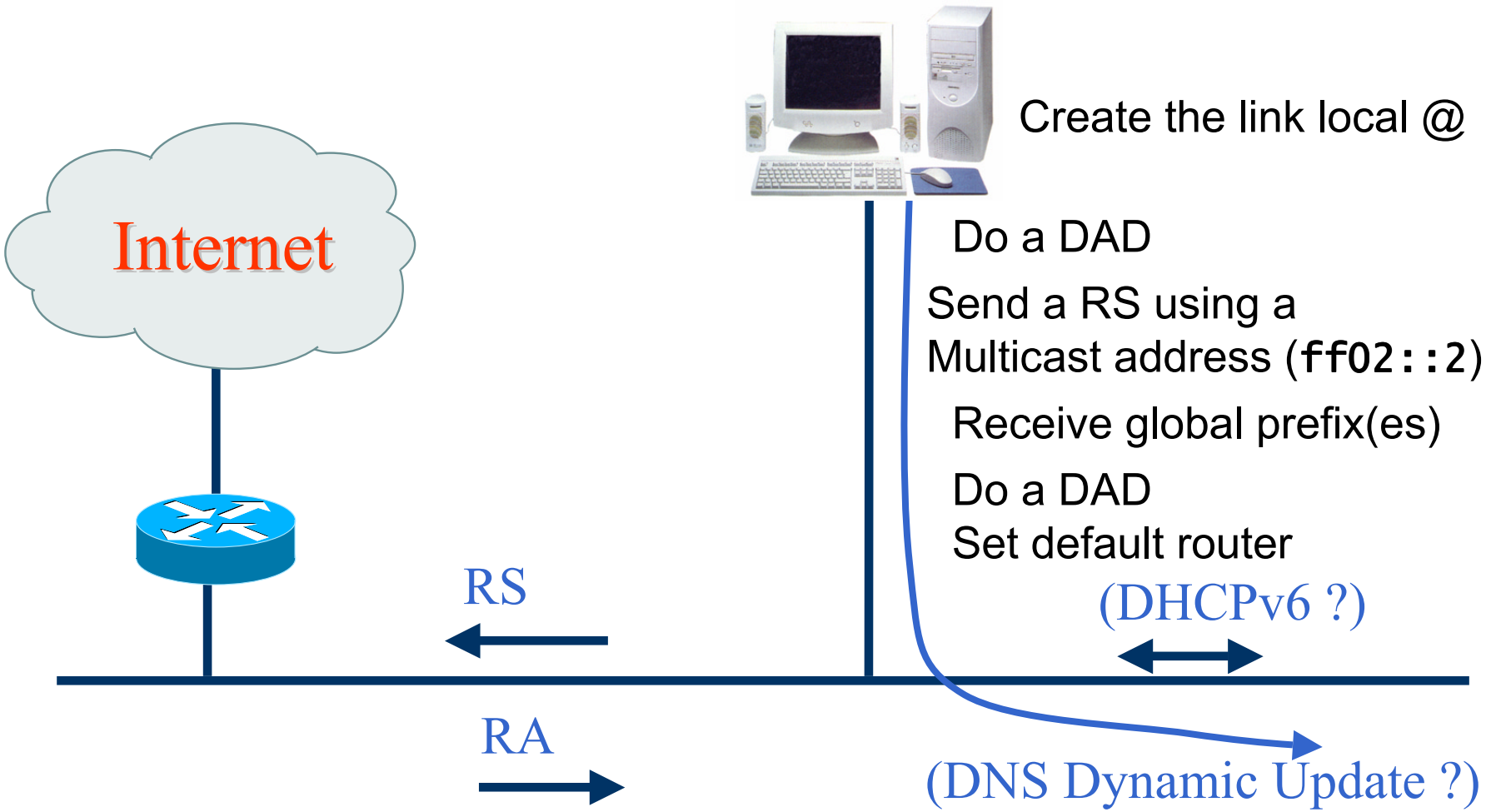
- Allows a host to create a global IPv6 @ from:
  - Its interface identifier = EUI64(MAC @)
  - router advertisements coming from router(s) on the link

=> GA = concat (RA, EUI64)

# Stateful auto-configuration (DHCPv6)

- *Dynamic Host Configuration Protocol for IPv6*
  - RFC 3315
  - IPv4 version of DHCP (RFC 1541, RFC 2131)
    - based on BOOTP (RFC 951)
- Server
  - Memorises client's state
  - Optionally provides the client with IPv6 addresses and configuration parameters
- Client
  - Sends requests and acknowledgements in accordance with the protocol (DHCP)

# Auto-configuration example



# Router Renumbering (RFC 2894 PS)

- Allow to change/add prefixes into routers
  - end-systems will use Neighbor Discovery Protocol to automatically discover and configure the new prefix(es)
- Several actions are sent to routers using well-known multicast groups:
  - Change prefix
  - Add prefix
- Security needs (IPsec, no replay)

# Routing Protocols

- RFC 2080 (PS) & 2081 (INFO) : RIPng
- RFC 2740 (PS) : OSPF v3
- `draft-ietf-isis-ipv6-05.txt`: IS-IS (01/2003)
- RFC 2545 (PS) : based on MBGP (RFC 2848)
  - Multi-extension protocol for BGP-4

⇒ No major differences with IPv4

- RFC 3031 : MPLS : MultiProtocol Label Switching
- and 6PE : MPLS Provider Edge IPv6 routing
  - Internet Draft : bgp-tunnel-04.txt

# IPv6 support in the DNS (DNSv6)

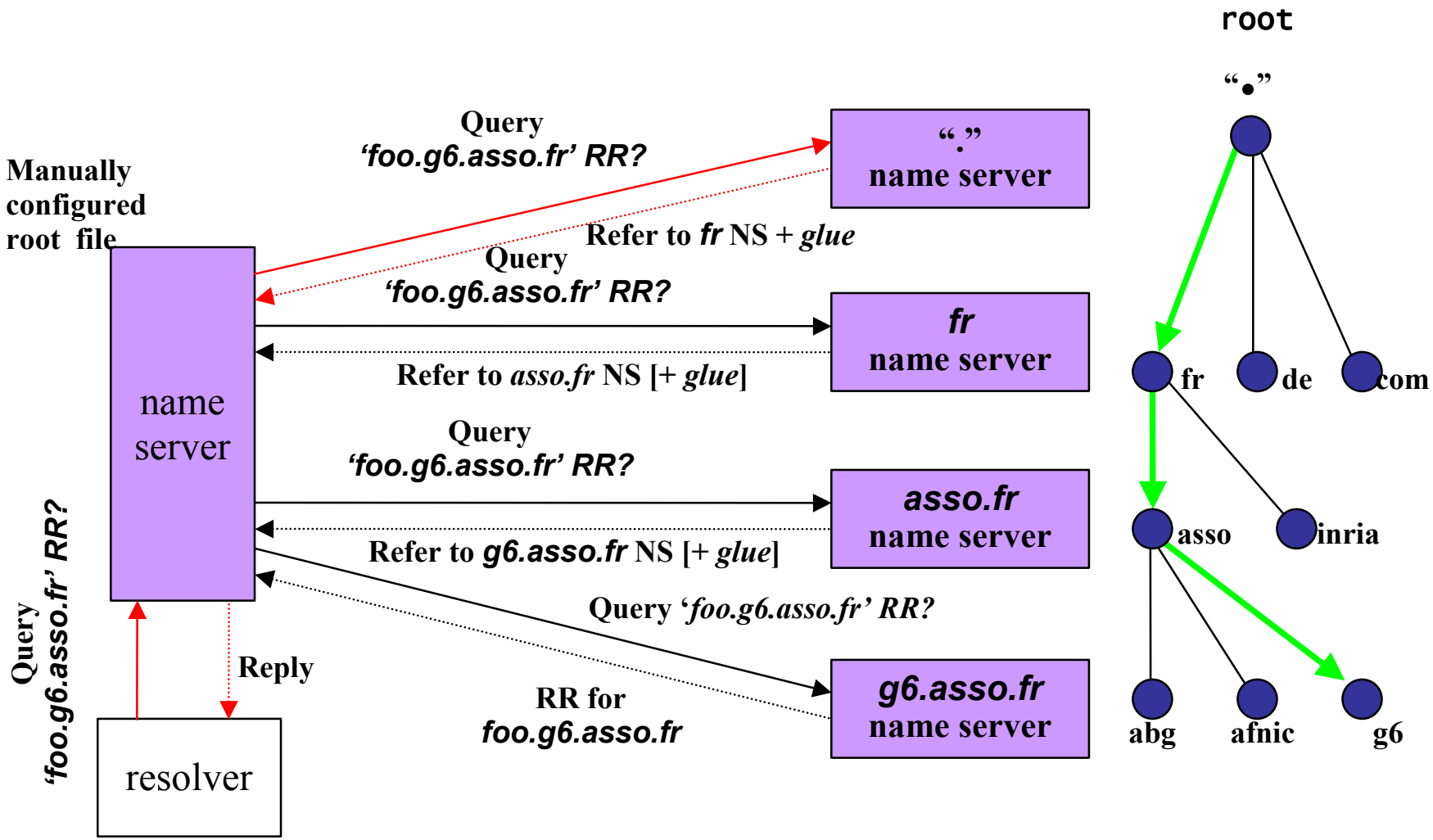
# Overview

- How important is the DNS?
- DNS Resource Lookup
- The Two Approaches to the DNS
- DNS Extensions for IPv6
- About the Required IPv6 glue in DNS Zones
- Lookups in an IPv6-aware DNS Tree
- DNS Service Continuity through IP Networks
- DNSv6 Operational Requirements & recommendations
- AFNIC Initiatives in the DNSv6 Field
- IPv6-capable DNS Software
- References

# How important is the DNS?

- Getting the IP address of the remote computer is necessary for every communication between TCP/IP applications
- Humans are unable to memorize millions of IP addresses ☹
- To a larger extent: the Domain Name System (DNS) provides applications with several types of resources (name servers, mail exchanges, reverse lookup, …) they need
- DNS design
  - hierarchy
  - distribution
  - redundancy

# DNS Resource Lookup



# DNS Extensions for IPv6

- RFC 1886 (PS) → RFC 3596 (DS) (upon successful interoperability tests)
- **AAAA** (RFC 3596): forward lookup ('Name → IPv6 Address'):
  - Equivalent to '**A**' record
  - Example:

```
ns3.nic.fr.   IN   A      192.134.0.49
              IN   AAAA   2001:660:3006:1::1:1
```

- **PTR** : reverse lookup ('IPv6 Address → Name'):
  - Reverse tree equivalent to `in-addr.arpa`
    - Nibble (4 bits) boundary
    - New tree: `ip6.arpa` (RFC 3596), under deployment
    - Former tree: `ip6.int` (RFC 1886), still maintained
  - Example:

```
$ORIGIN 1.0.0.0.6.0.0.3.0.6.6.0.1.0.0.2.ip6.{int,arpa}.
1.0.0.0.1.0.0.0.0.0.0.0.0.0.0.0   PTR   ns3.nic.fr.
```

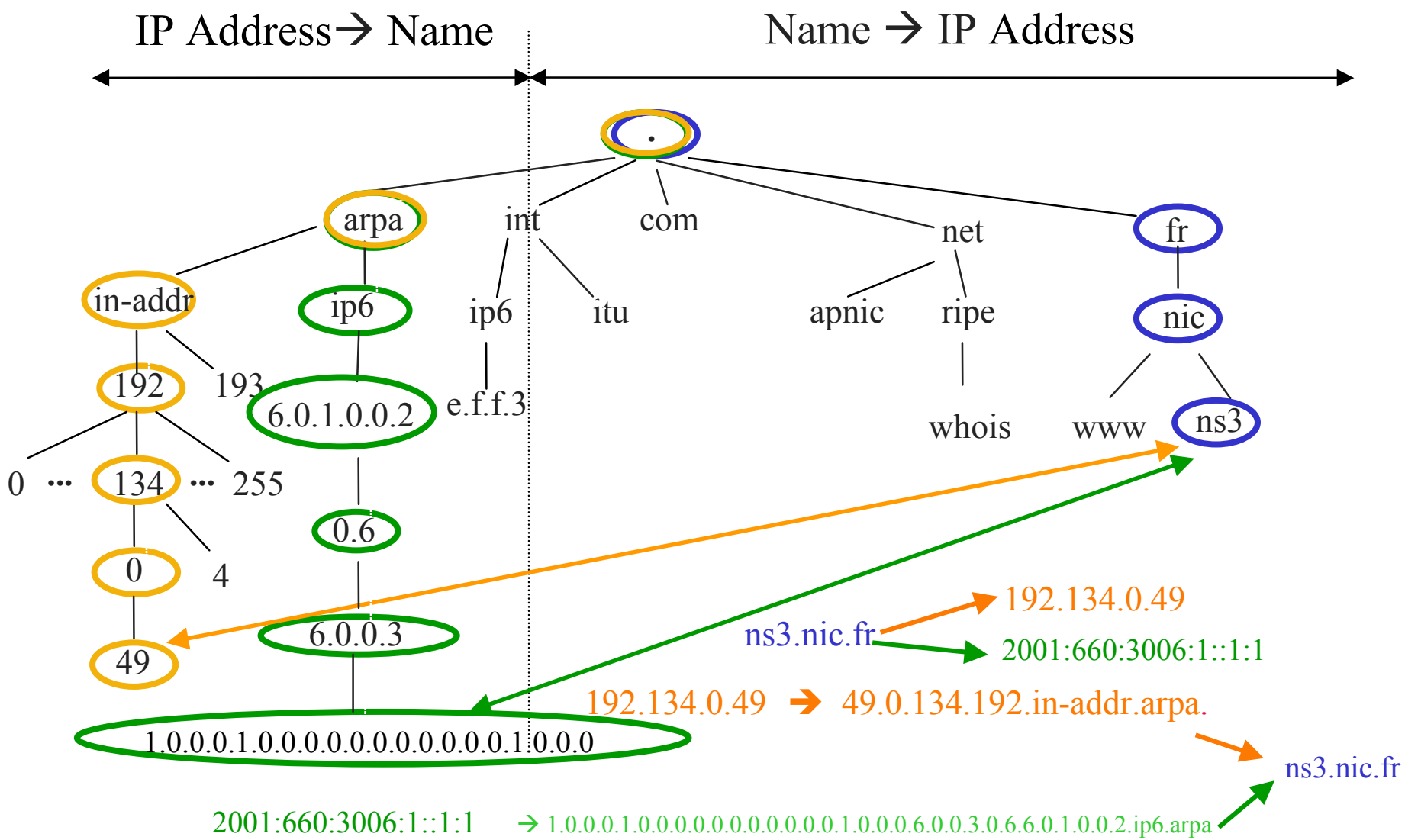
# The Two Approaches to the DNS

- The DNS seen as a ***Database***
  - Stores different types of ***Resource Records*** (RR): `SOA, NS, A, AAAA, MX, PTR, TXT, …`

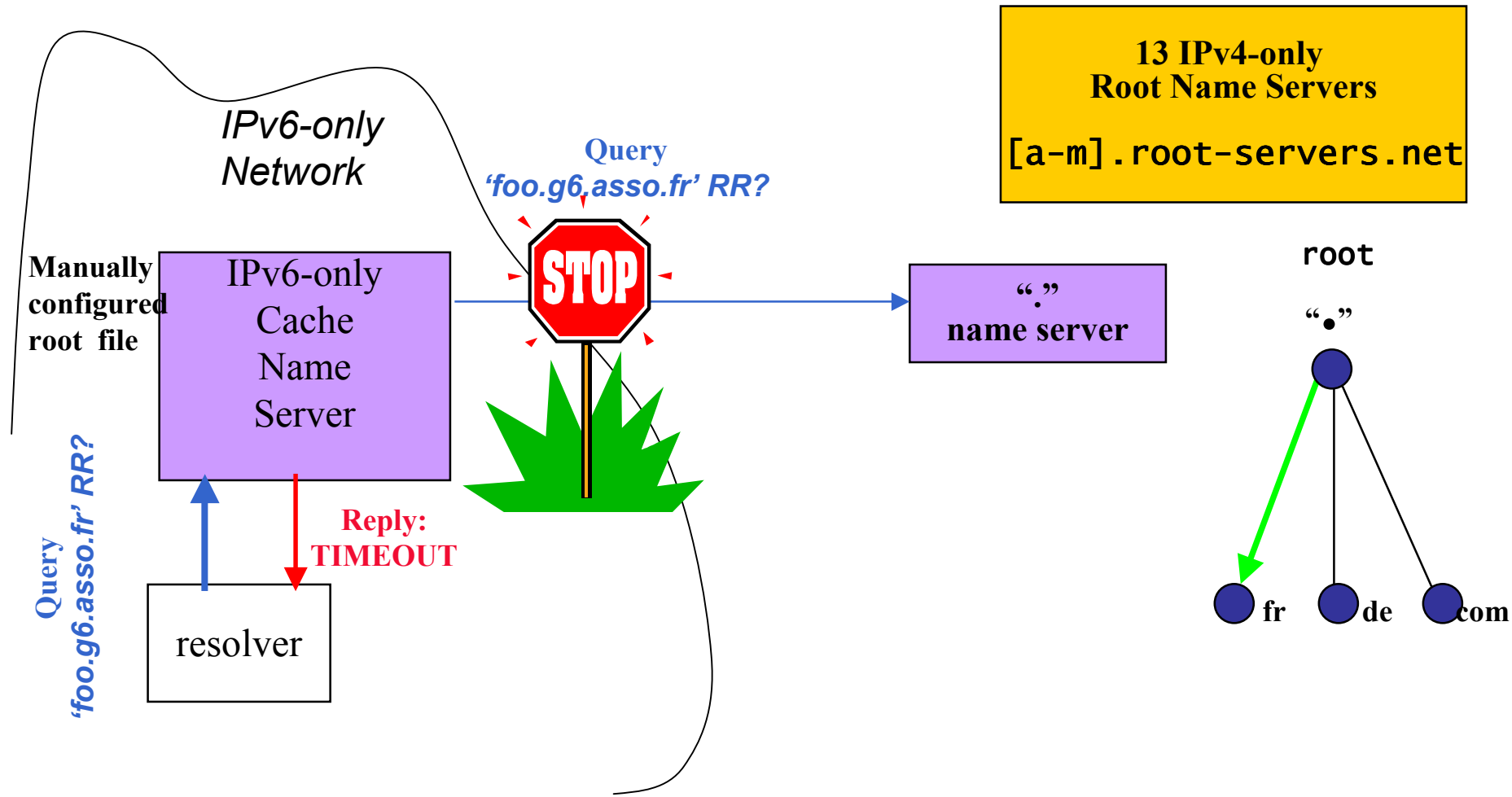
  ➔ DNS data are independent of the IP version (v4/v6) the DNS server is running on!
- The DNS seen as a ***TCP/IP application***
  - The service is accessible in either transport modes (UDP/TCP) and over either IP versions (v4/v6)

  ➔ Information given over both IP versions MUST BE CONSISTENT!

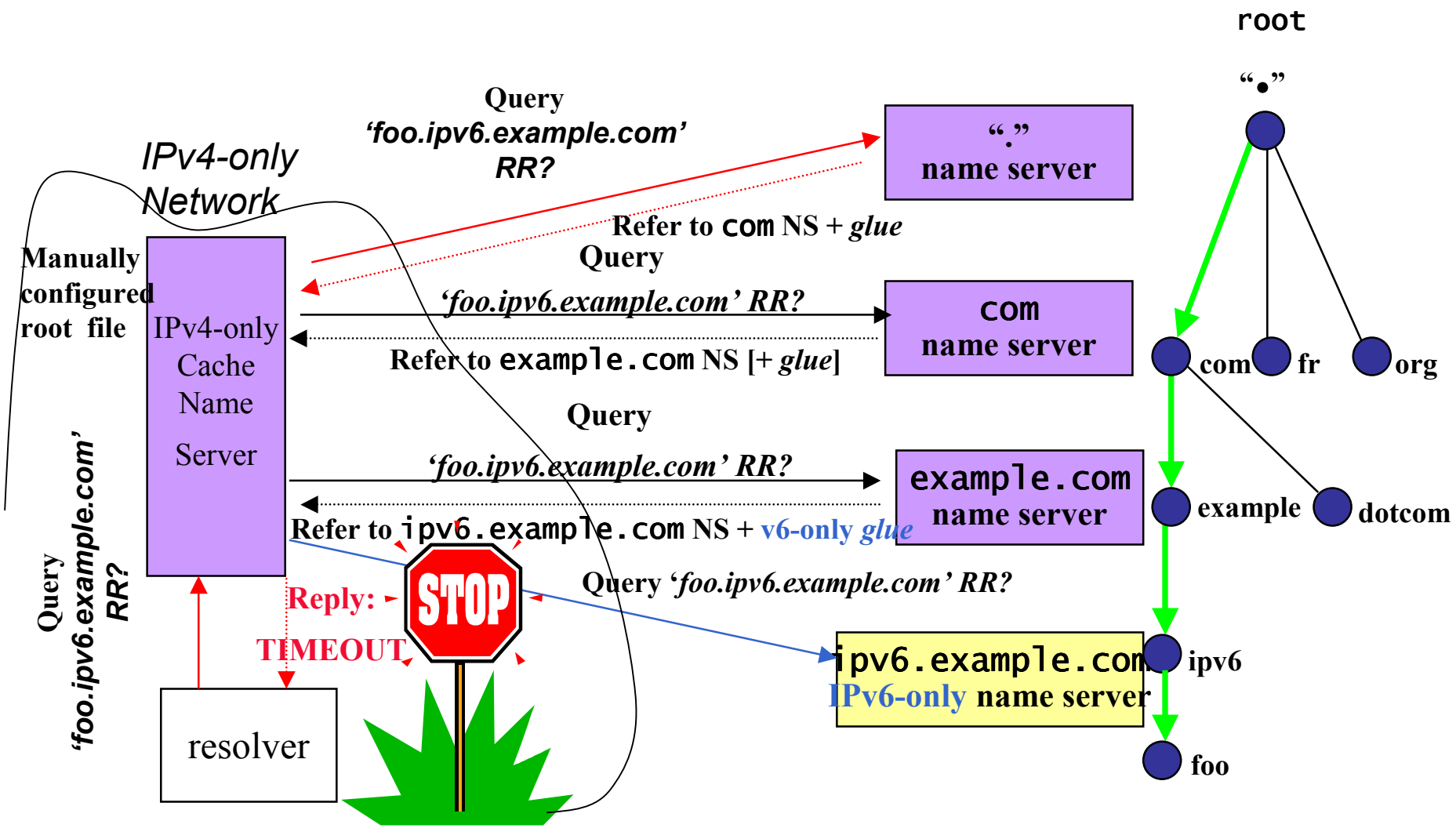
# Lookups in an IPv6-aware DNS Tree



# DNS Service Continuity through IP Networks



# DNS Service Continuity through IP Networks (2)



# About Required IPv6 Glue in DNS Zones

- ***When the DNS zone is delegated to a DNS server (among others) contained in the zone itself***
- Example: In zone file `rennes.enst-bretagne.fr`

```
@   IN   SOA   rsm.rennes.enst-bretagne.fr. fradin.rennes.enst-bretagne.fr.
               (
                        2003111700;serial
                        86400     ;refresh
                        3600      ;retry
                        3600000   ;expire
                        86400     ;negative ttl
        IN   NS   rsm
        IN   NS   univers.enst-bretagne.fr.
[…]
ipv6    IN   NS   rhadamanthe.ipv6
        IN   NS   ns3.nic.fr.
        IN   NS   rsm

rhadamanthe.ipv6     IN   A    192.108.119.134
                     IN   AAAA 2001:660:7301:1::1
[…]
```

- IPv4 glue (`A 192.108.119.134`) is required to reach `rhadamanthe` over IPv4 transport
- IPv6 glue (`AAAA 2001:660:73001:1::1`) is required to reach `rhadamanthe` over IPv6 transport

# IPv6 Support for the Root Servers

- When ?
  - Nobody knows ☹
- Why not?
  - No room available for an extra root server IP(v4/v6) address
  - DNS response size limit is 512 bytes unless EDNS.0 is used
  - "IPv6 infrastructure is not mature yet" for the operation of the root servers
- While waiting…
  - Go to the **RS.NET** Testbed: http://www.rs.net/
    - Test and prove that new technologies (IPv6, DNSsec, IDN) are harmless
    - Several TLDs participate in the testbed (FR, JP, SE, …)

# Putting AAAA Glue Records in the Root Zone

- Who can put them?
  - IANA/ICANN
- When?
  - Soon (hopefully)…
- Why is it so slow?
  - FR & JP asked IANA to add their AAAA glue several months ago
  - IANA/ICANN had some technical concerns about the general case
  - Several technical documents (theoretical and practical) published
  - RSSAC met several times to discuss the issue
  - RSSAC is finally making recommendations to IANA/ICANN to move forward

# Putting AAAA Glue Records in the Root Zone (2)

- Related documents
  - draft (Kato-Vixie) on DNS response size (dnsop WG)
    - DNS response size from root servers
      - For a TLD in general
      - For common and uncommon names, average and worst cases
  - Experiments results from NLnet Labs & RIPE NCC
    - Real life traffic replayed on L & K root servers
    - Conclusion: Adding AAAA glue to the root zone has no negative effect on the root servers
  - DNS response size and name compression by AFNIC
    - Theoretical calculations on DNS response size from root servers
    - General case and FR specific case
    - Name compression benefits (more space for extra AAAA glue)

# DNS Discovery

- A ***Stub Resolver*** needs a ***Recursive Name Server*** address for name resolution and a ***Search Path***

- In IPv4 world, the DNS parameters are:
  - Either configured manually in the ***stub resolver*** (e.g. `/etc/resolv.conf`)
  - Or discovered via DHCPv4

- In IPv6 world:
  - So far, only manual configuration is available ☹
  - Proposals for DNS Discovery:
    - Under discussion IETF `ipv6/dnsop WGs`
    - Stateless Discovery: **RA-Based** ***vs*** Stateful Discovery: **DHCPv6**(-light)
    - Well-known address (anycast or uniscast): seems to be out of date

# When there is no DNS available

- In case:
  - No manual or automatic DNS configuration has been performed
  - DNS servers do not respond or respond wit error
- Link Local Multicast Name Resolution (**LLMNR**)
  - IETF `dnsext` WG (work in progress)
  - The same message format as conventional DNS but different ports
  - Each node is authoritative for its own name(s)
  - Sender/Responder ➔ LLM/Unicast
- mDNS
  - Apple's proprietary protocol
  - Does not inter-operate with LLMNR

# DNSv6 Operational Requirements & Recommendations

- The **target** today **IS NOT** the transition from an IPv4-only to an IPv6-only environment
- **It IS RATHER** to get from an IPv4-only to a mixed v4-v6 environment where:
  - Some systems will remain IPv4-only
  - Some systems will be dual-stacked
  - Some systems will be IPv6-only
- How to get there?
  - Start by testing DNSv6 on a small network and get your own conclusion that DNSv6 is harmless
  - Deploy DNSv6 in an incremental fashion on existing networks
  - **DO NOT BREAK** something that **works fine** (production IPv4 DNS)!

# DNSv6 Operational Requirements & Recommendations (2)

- How to get there? (cont.)
  - For new large IPv6-only networks: enable IPv6-only resolvers to query the DNS for IPv4-only resources by (for example):
    - Letting them query dual-stack forwarders
    - Using some DNS ALG
- Bear in mind
  - Any DNS zone (and especially if related to an IPv6-only network) SHOULD be served by at least one IPv4 name server
  - All DNS zones (including 'root', yes, yes!) SHOULD be reachable over IPv4 and IPv6

# DNS IPv6-capable software

- BIND (Resolver & Server)
  - http://www.isc.org/products/BIND/
  - BIND 8.2.4 (or later)
  - BIND 9
- On Unix distributions
  - Resolver Library (+ (adapted) BIND)
- NSD (authoritative server only)
  - http://www.nlnetlabs.nl/nsd/
- Microsoft Windows (Resolver & Server)
- …

# APIs

- `getaddrinfo()` for forward lookup
  - *hostname* ➔ *addresses*
  - Replacement of `gethostbyname()`
  - With `AF_UNSPEC`, applications become protocol-independent
- `getnameinfo()` for reverse lookup
  - *address* ➔ *hostname*
  - Replacement of `gethostbyaddr()`

# AFNIC Initiatives in the DNSv6 Field

- Native support of DNSv6
  - `.fr` is the first European ccTLD and the second TLD in the world (after `.jp`)

- Officially hosting a secondary DNSv6 on `ns3.nic.fr` for:
  - ccTLD zones:
    - `fr, re` // delegated to AFNIC
    - `br, dz, es, my, af, …`
  - High level reverse zones:
    - **`ip6.int`,**
    - `[6-9].0.1.0.0.2.ip6.{int,arpa}, …` // Ripe blocs

- DNSv6 cache forwarding service:
  - Name resolution service for IPv6-only sites
  - Efficient and scalable for a well defined community (for instance French IPv6 community)
  - Service running on `nscachev6.nic.fr`

# Standardization process ( RFC 1886 inter-operability tests & reports)

- RFC 1886: AAAA & ip6.int
- RFC 3152: ip6.arpa

- RFC 1886 inter-operability tests
  - Who: 6WIND, AFNIC, FT R&D and IRISA (within « G6 test » activity)
  - When & where: 3 June & 4 July 2002, AFNIC and 6WIND buildings
  - What was tested: support of AAAA and ip6.arpa by different name server/resolver software
  - Results:
    - successful inter-operability tests but found some minor failures
    - http://w6.afnic.fr/RFC1886/testRFC1886.html

- RFC 1886 inter-operability reports
  - When & where: IETF 54 Yokohama (14-19 July 2002) at dnsext working group session
  - Presentation:
    - http://www.ietf.org/proceedings/02jul/slides/dnsext-1/index.html
  - Results:
    - RFC 1886 currently il a Proposed Standard (PS) status
    - Draft Standard (DS) RFC 3596 published in October 2003, obsoletes RFC 1886

# References

- DNSv6-related RFCs & Internet-Drafts
  - RFC 3596
  - "DNS IPv6 transport operational guidelines" (A. Durand & J. Ihren, work in progress)
    draft-ietf-dnsop-ipv6-transport-guidelines-01.txt
  - "DNS Response size issues" (A. Kato & P. Vixie, work in progress)
    draft-ietf-dnsop-respsize-00.txt
- Other technical documents
  - Adding IPv6 Glue To The Rootzone ( R. van der Pol & D. Karrenberg)
    http://www.nlnetlabs.nl/ipv6/publications/v6rootglue.pdf
  - "DNS Response Size and Name Compression" (M. Souissi, AFNIC)
    http://w6.nic.fr/dnsv6/dns-resp-size-and-name-compression
- Books
  - DNS and BIND, 4th edition (Paul Albitz & Cricket Liu)

# IPv6 DNS and root servers

- DNS root servers … critical resources
- 13 roots « around » the world (#10 in the US)
- Need for root servers to be installed in other locations (EU, Asia, Africa, …)
- New technique : **anycast** DNS server
  - To build a clone from the master/primary server
  - Containing the same information (files)
  - Using the same IP address
- Such anycast servers have already begun to be installed :
  - F root server : Ottawa, Paris (Renater), Hongkong, Dubai, …
  - K root : London, Amsterdam, …
  - I root : Stockholm, Milan, …
- B, F, H and M-root servers are IPv6 capable today

# IPv6 Mobility

# Mobility Overview

- ***Mobility*** is much wider than ***"nomadism"***
- Keep the same IP address regardless of the network the equipment is connected to:
  - reachability
  - configuration
  - real mobility
- Difficult to optimize with IPv4 (RFC 3344 PS)
- Use new facility of IPv6: MIPv6

# IPv6 Mobility (MIPv6)

- IPv6 mobility relies on:
  - New IPv6 features
  - The opportunity to deploy a new version of IP
- Goals:
  - Offer the direct communication between the mobile node and its correspondents
  - Reduce the number of actors (Foreign Agent (IPv4) no longer used )
- MIPv6: RFC XXXX (after a long work in progress, I-D version 24)

# General Considerations

- A globally unique IPv6 address is assigned to every ***Mobile Node*** (**MN**): ***Home Address*** (**HA**)
- This address enables the MN identification by its ***Correspondent Nodes*** (**CN**)
- A MN must be able to communicate with non mobile nodes
- Communications (keep layer 4 connections) have to be maintained while the MN is moving and connecting to foreign (visited) networks

# Main features/requirements of MIPv6

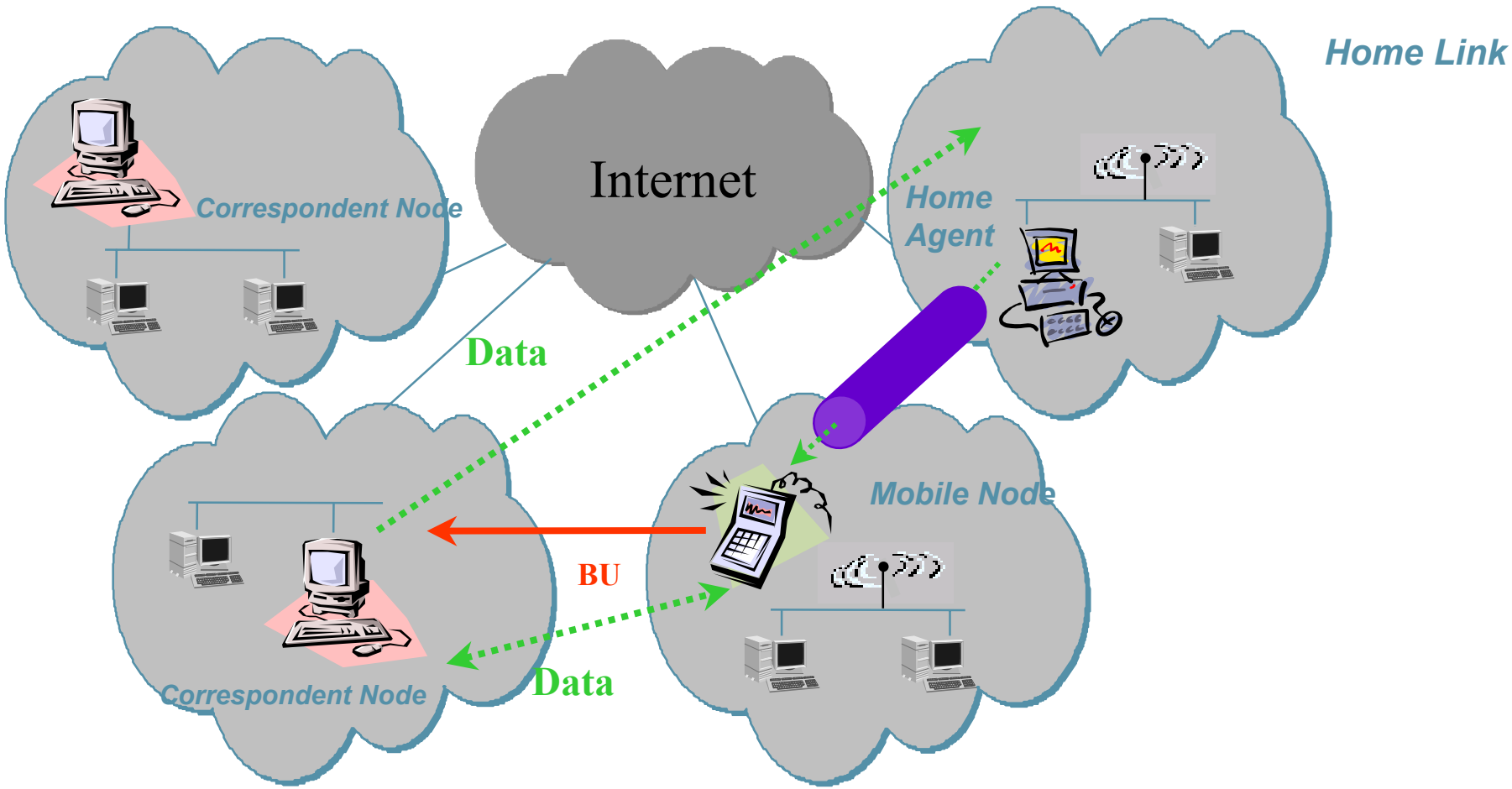
- CN can:
  - Put/get a Binding Update (BU) in/from their Binding Cache
  - Learn the position of a mobile node by processing BU options
  - Perform direct packet routing toward the MN (Routing Header)

- The MN's Home Agent must:
  - Be a router in the MN's home network
  - Intercept packets which arrive at the MN's home network and whose destination address is its HA
  - Tunnel (IPv6 encapsulation) those packets directly to the MN
  - Do reverse tunneling (MN → CN)

# Mobile Node Addressing

- A MN is always reachable on its Home Address
- While connecting to foreign networks, a MN always obtains a temporary address, “the Care-of Address” (CoA) by auto-configuration:
  - It receives Router Advertisements providing it with the prefix(es) of the visited network
  - It appends that (those) prefix(es) to its Interface-ID
- Movement detection is also performed by Neighbor Discovery mechanisms

# MIPv6: IETF Model



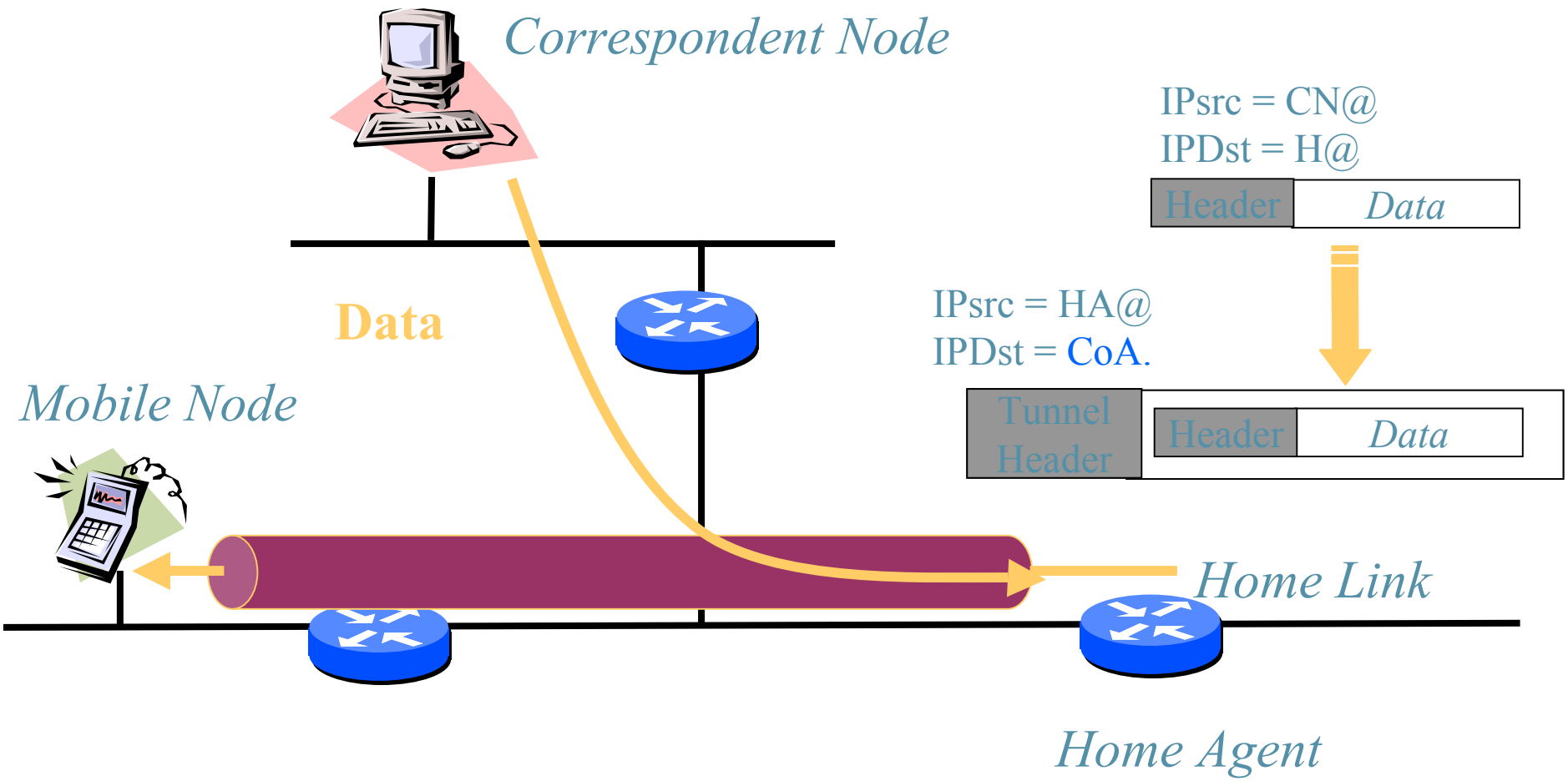
# Binding Cache Management

- Every time the MN connects to a foreign network, it sends a Binding Update (BU):
  - Every BU carries a TTL
  - A MN caches the list of CNs to which it sent a BU
  - The MN may have multiple CoAs, the one sent in the BU to the HA is called the ***primary CoA***

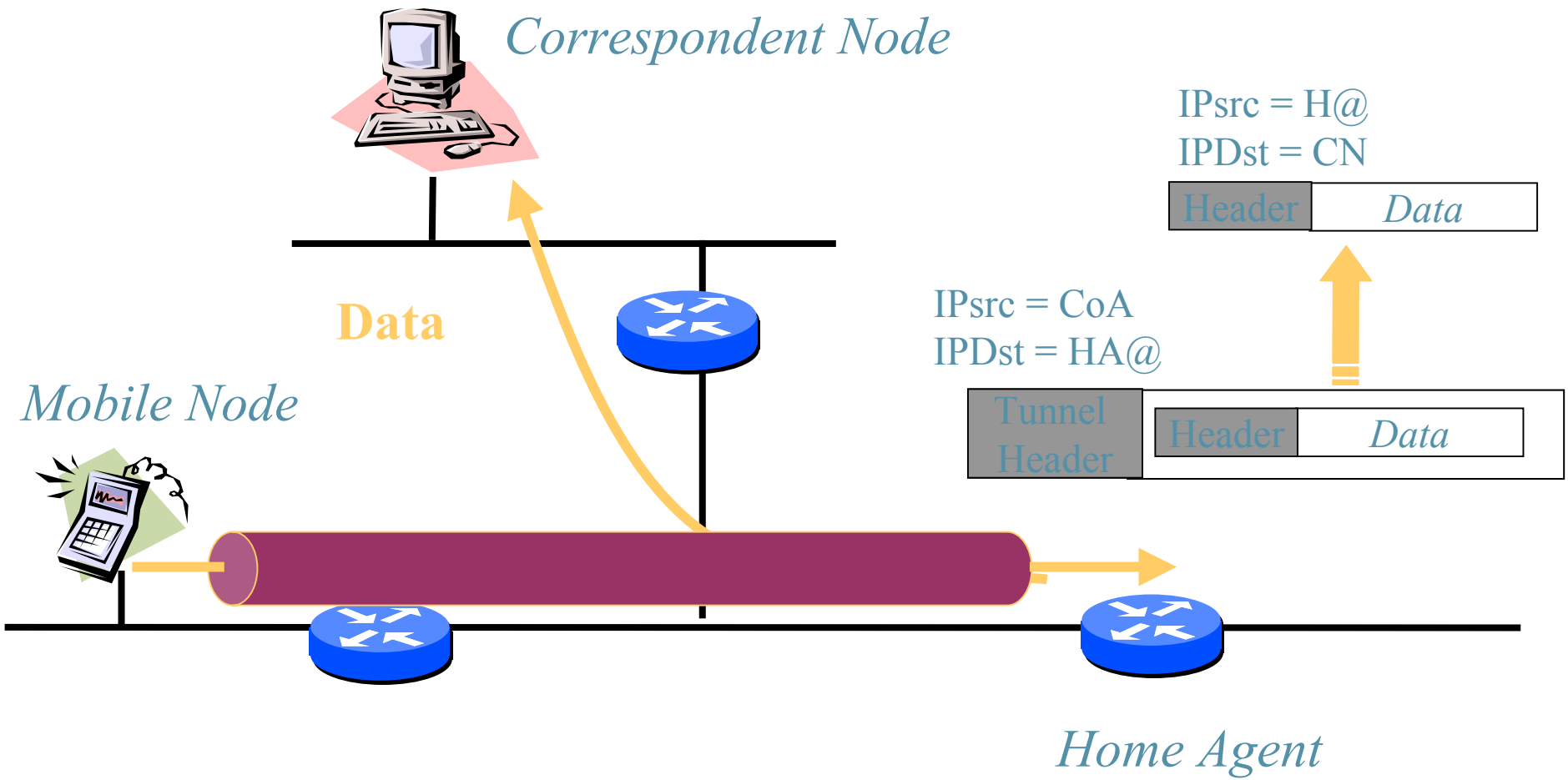
# Communication with a Mobile Node

- 2 methods:
  - Bi-directional Tunneling
    - No mobility requirements on CNs
    - No visibility of MNs for CNs
    - Network load increased
    - HA role much reinforced
  - Direct Routing
    - Much more complex mechanism
    - HA role much alleviated

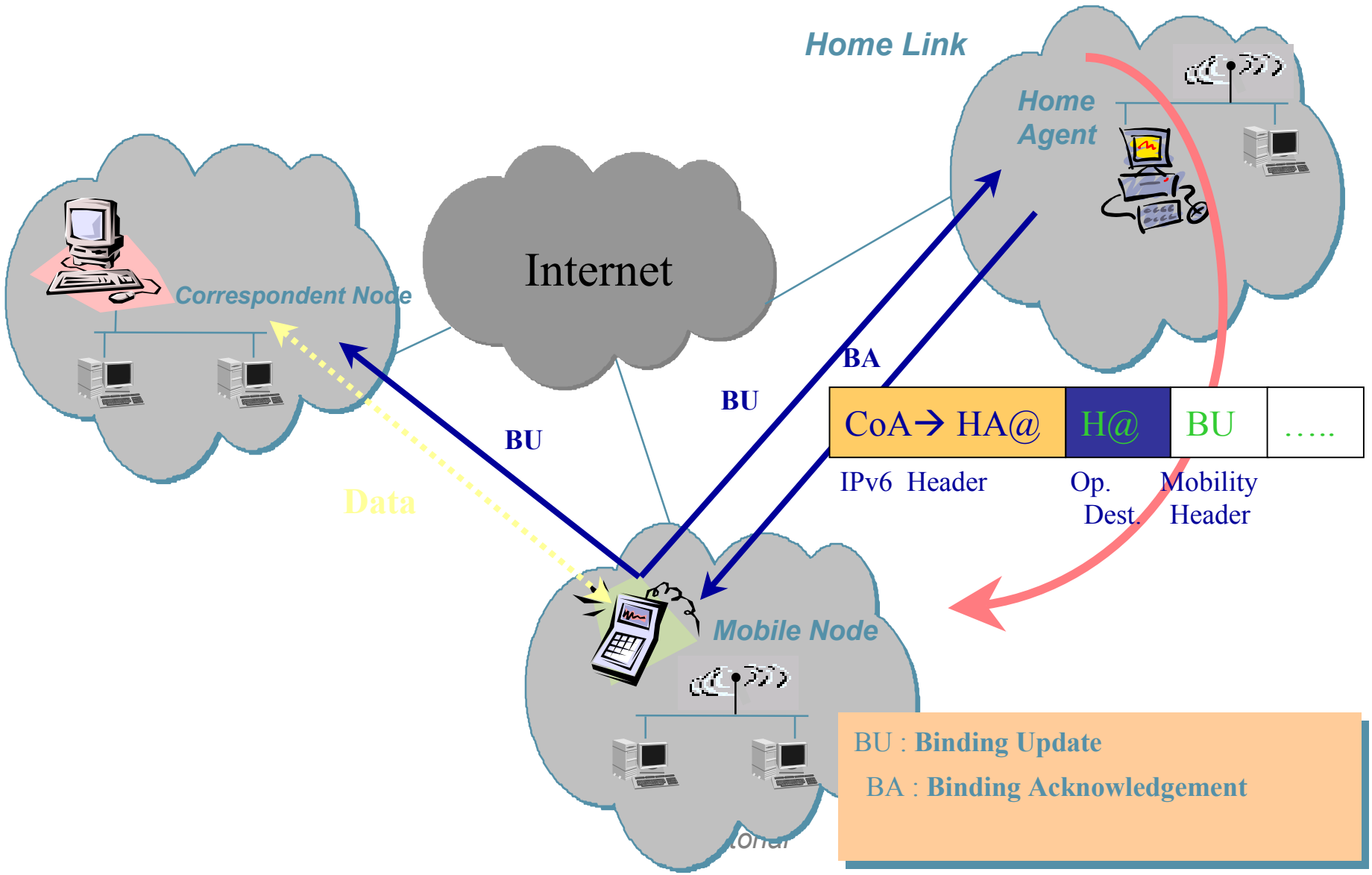
# Bi-directional Tunneling



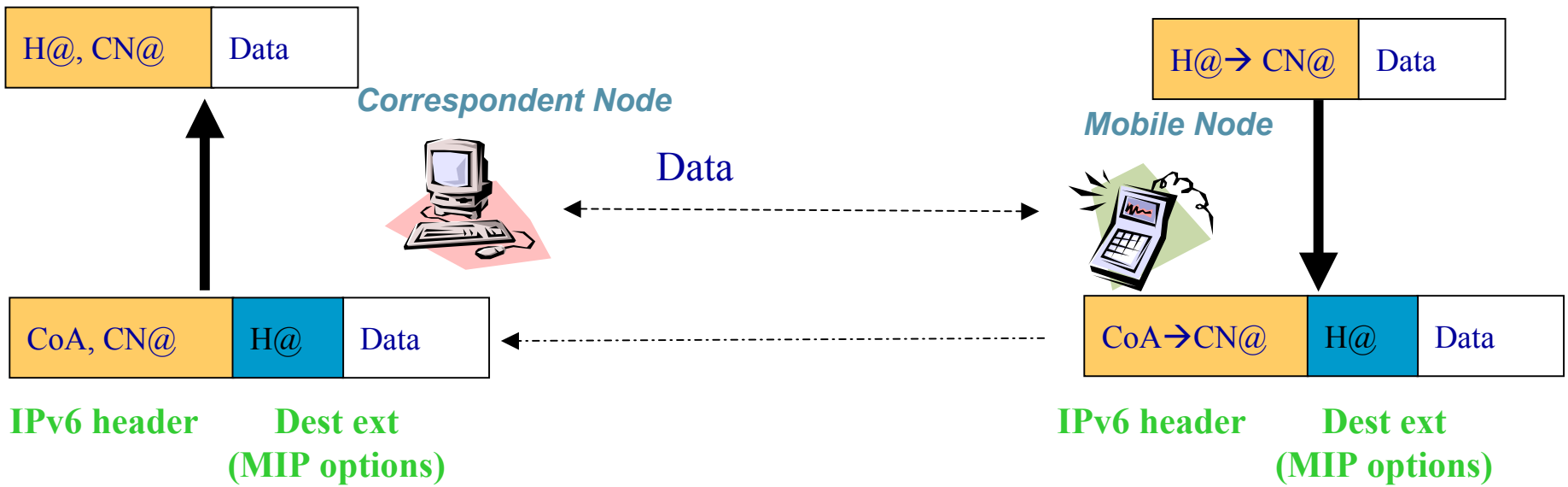
# Bi-directional Tunneling (2)



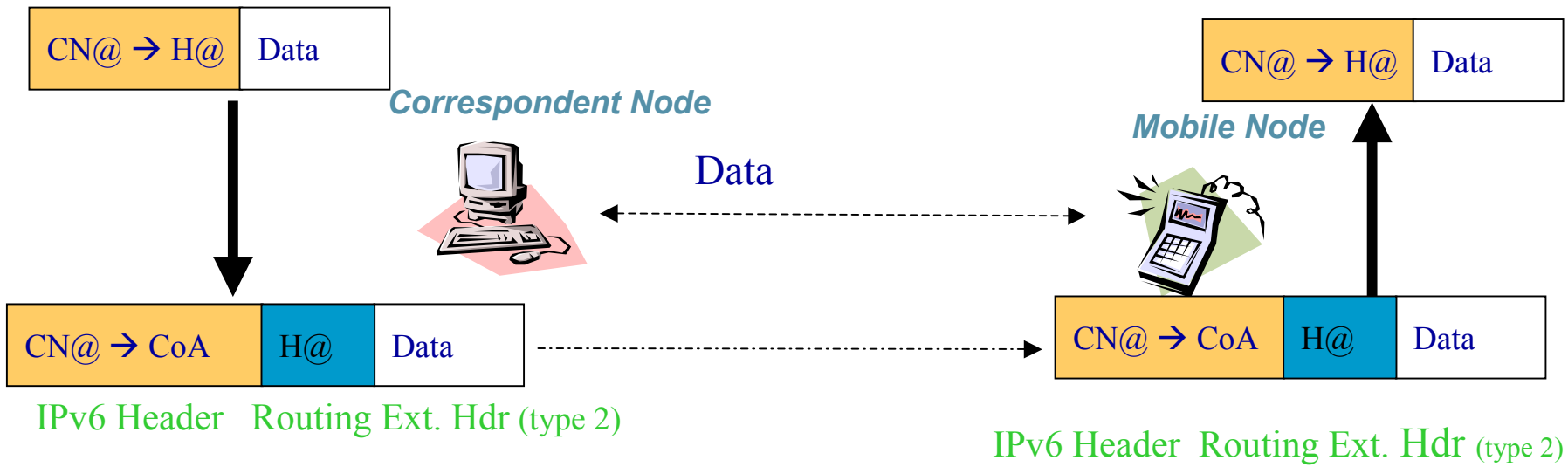
# Direct Routing



# Direct Routing: MN → CN



# Direct Routing: CN → MN



# Binding Update Authentication

- BU information needs protection and authentication
  - Sender authentication
  - Data integrity protection
  - Replay protection
- Authentication Data sub-option used to carry necessary data authentication
- IPsec may be used to fulfill all these needs
  - MIPv6 is seen as a good opportunity to boost IPsec (and IPv6) deployment

# Mobility Features For IPv6 Hosts

- For MNs
  - To perform IPv6 packet encapsulation/decapsulation
  - To send BUs and receive BAs (process the Mobility Header)
  - To keep track of BUs sent
- For CNs
  - To be able to process the Mobility Header (Binding Update, Binding Acknowledge)
  - To use the Routing Header (type 2)
  - Maintain a Binding Cache

# Mobility Features For IPv6 Routers

- At least one IPv6 router on the Home Link of the MN must be able to act as a Home Agent

- A Home Agent must:
  - Maintain MN's binding information
  - Intercept packets for a MN in a Home Link it is responsible for
  - Encapsulate/decapsulate (tunnel) these packets and forward them to the CoA of the MN

# IPv6 Security with IPsec

# Security: IPsec

- Work made by the IETF IPsec wg
- Applies to both IPv4 and IPv6 and its implementation is:
  - Mandatory for IPv6
  - Optional for IPv4
- IPsec Architecture: RFC 2401
- IPsec services
  - Authentication
  - Integrity
  - Confidentiality
  - Replay protection
- IPsec modes: Transport Mode & Tunnel Mode
- IPsec protocols: AH (RFC 2402) & ESP (RFC 2406)

# IPsec Architecture (RFC 2401)

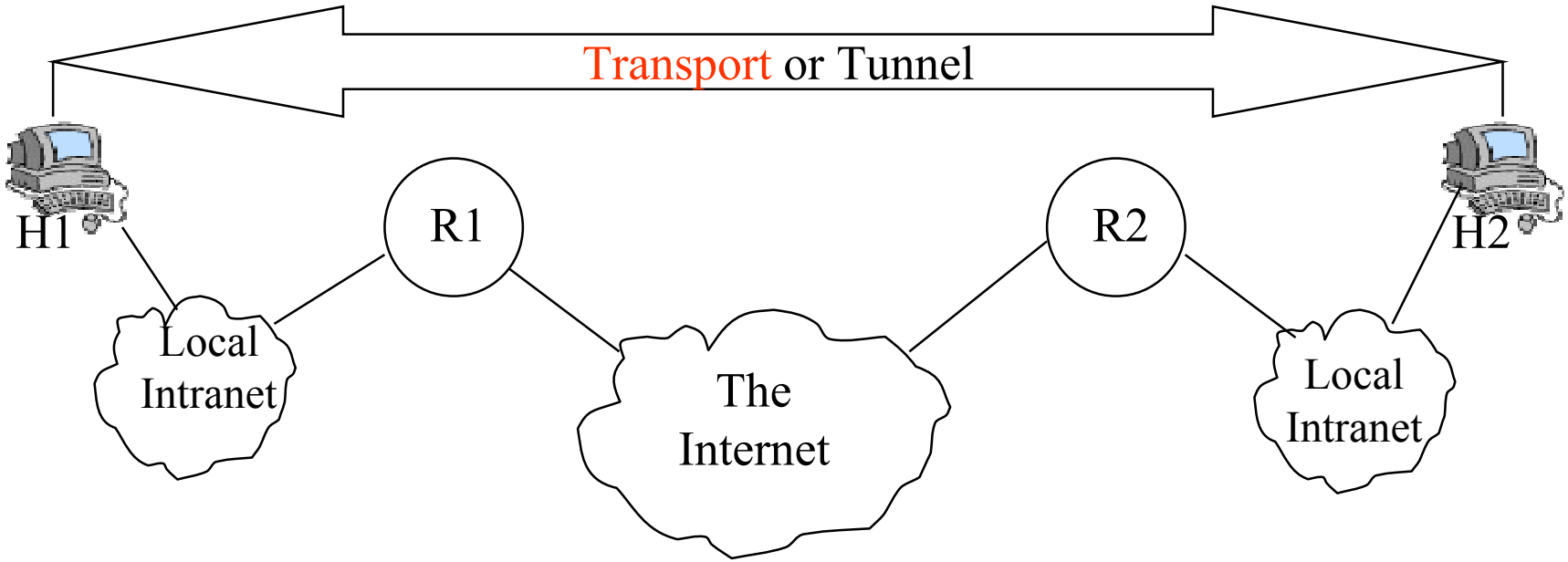
- Security Policies: Which traffic is treated?
- Security Associations: How traffic is processed?
- Security Protocols: Which protocols (extension headers) are used?
- Key Management: Internet Key Exchange (IKE)
- Algorithms: Authentication and Encryption

# IPsec Modes

- Transport Mode
  - Above the IP level
  - Below the Transport level
  - Only the IP datagram payload is protected
- Tunnel Mode
  - IP within IP
  - Below the transport level
  - All the tunneled IP datagram is protected

# IPsec Scenarios
# Scenario 1: H2H

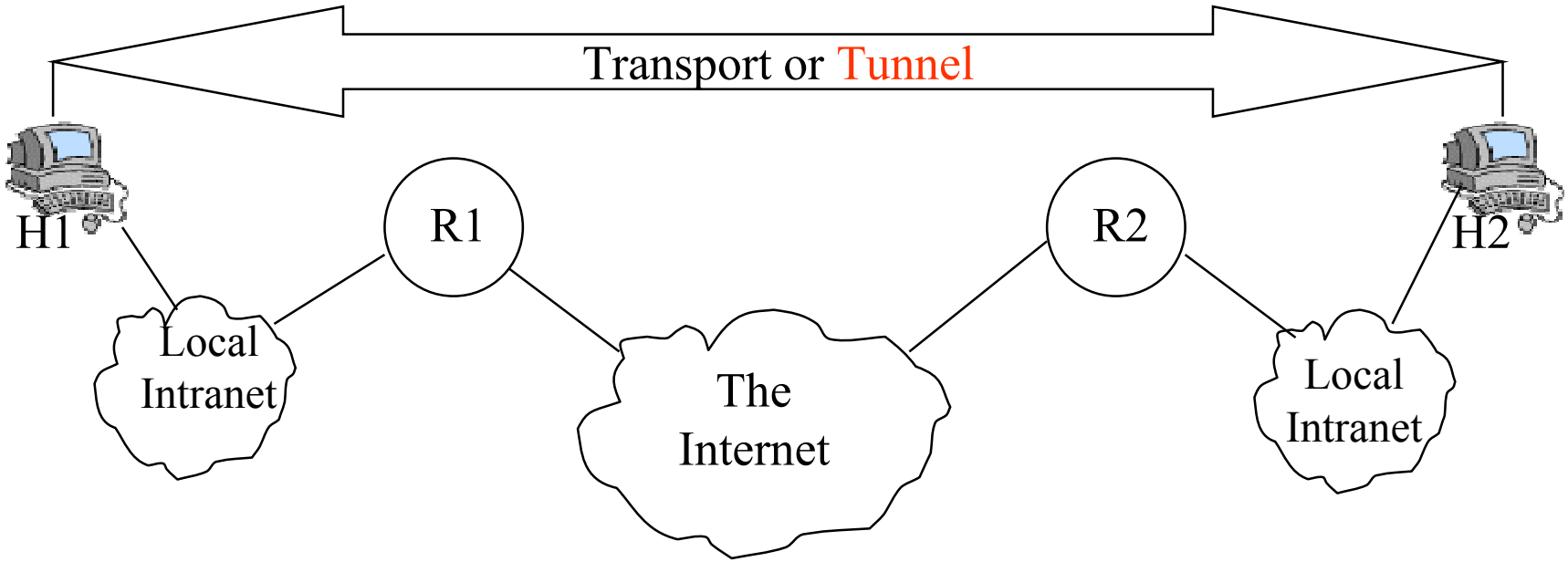
- End-to-end service
- Transport/Tunnel mode between the 2 hosts




| IP header | **IPsec ext AH/ESP** | Payload |
|---|---|---|

# IPsec Scenarios
# Scenario 1: H2H

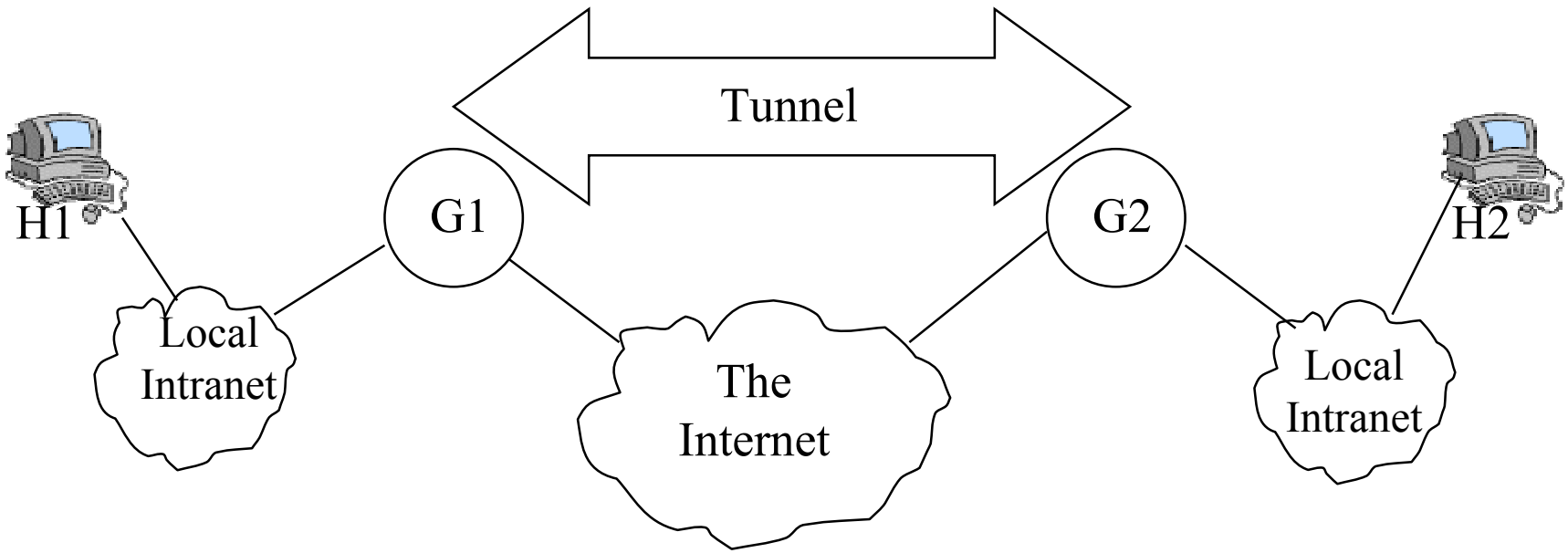
- End-to-end service
- Transport/Tunnel mode between the 2 hosts




| IP header | **IPsec ext AH/ESP** | **Inner IP header** | Payload |
|---|---|---|---|

# IPsec Scenarios
# Scenario 2: G2G

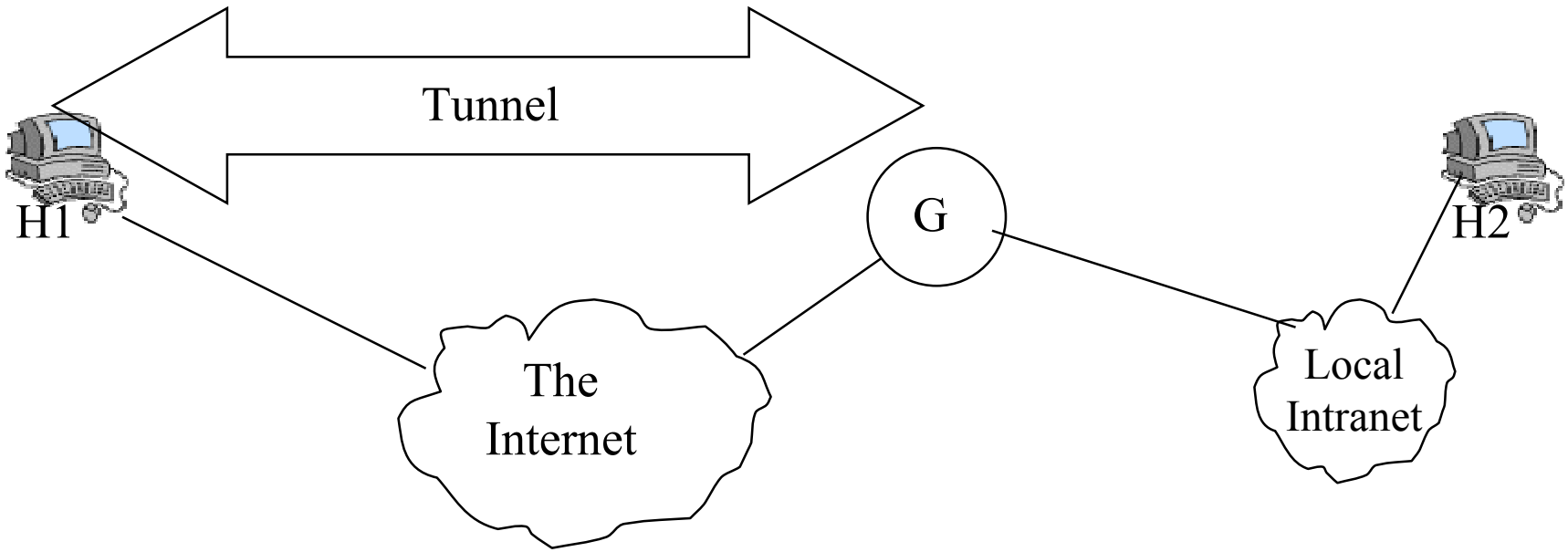
- VPN, Site-to-Site/ISP agreements, …
- Tunnel between the 2 gateways




| IP header | **IPsec ext AH/ESP** | **Inner IP header** | Payload |
|---|---|---|---|

# IPsec Scenarios
# Scenario 3: H2G, G2H

- Dial-in users
- Tunnel between the “external” host and the gateway




| IP header | **IPsec ext AH/ESP** | **Inner IP header** | Payload |
|---|---|---|---|

# IPsec Protocols

- Authentication Header (AH)
  - RFC 2402
  - Protocol# (Next Header) = 51
  - Provides:
    - Connectionless Integrity
    - Data origin authentication
    - Replay protection
  - Is inserted
    - In Transport mode: After the IP header and before the upper layer protocol (UDP, TCP, …)
    - In Tunnel mode: Before the original IP header (the entire IP header is protected)
- Encapsulation Security Payload Header (ESP)
  - RFC 2406
    - Protocol# (Next Header) = 50
  - Provides:
    - Connectionless Integrity
    - Data origin authentication
    - Replay protection
    - Confidentiality
  - Is inserted
    - In Transport mode: After the IP header and before the upper layer protocol
    - In Tunnel mode: before an encapsulated IP header

# IPsec: Protocols, services & modes combinations

| | Transport Mode | Tunnel Mode SA |
|---|---|---|
| **AH** | Authenticates IP payload and selected portions of IP header | Authenticates entire inner IP datagram (header + payload), + selected portions of the outer IP header |
| **ESP** | Encrypts IP payload | Encrypts inner IP datagram |
| **ESP with Authentication** | Encrypts IP payload and authenticates IP payload but not IP header | Encrypts and authenticates inner IP datagram |

# IPsec : Key Management

- Manual
  - Keys configured on each system
- Automatic: IKE (Internet Key Exchange, RFC 2409)
  - Security Association negotiation: ISAKMP (Internet Security Association and Key Management Protocol, RFC 2408)
    - Different blocs (payloads) are chained together after ISAKMP header
  - Key Exchange Protocols: Oakley, Scheme
  - IKEv2: much simpler (work in progress)
- Algorithms: Authentication and Encryption

# Early deployments… Building the Internet v6

# Agenda

- 6bone
  - G6bone
- The 6REN initiative
- Large scale deployments
- 6Tap, IPv6 Exchanges
- Renater IPv6 pilot
- Native IPv6 service in Renater-3

# 6bone

- First IPv6 network
- Started July 15th 1996 between 3 sites:
  - WIDE/JP, UNI-C/DK, G6/FR
- Today: >500 sites in >40 countries
- IETF Working Group: NGtrans
- http://www.6bone.net
  - whois –h whois.6bone.net
- Phase out plan planned for 06/06/2006
  - pTLA allocations stopped (01/2004)

# 6bone

- Islands of nodes connected with IPv6
- Mainly interconnected through IPv4 tunnels
- Some native links (to 6TAP, …)
- Routing Protocol:
  - static, at the beginning
  - Now dynamic (RIPng, ISIS, OSPFv3, BGP4+)

# G6-bone

- G6-bone was the IPv6 BB operated by the **G6**
- It became Renater's IPv6 pilot service
- And then Renater production IPv6 service (6R3)
- Renater is the French High Education and Research BB infrastructure

# G6 group

- Group of IPv6 testers in France, Tunisia, Senegal, …
- Academic & industrial partners
  - CNRS, ENST, INRIA, Universities …
  - AFNIC, 6Wind, Bull, ...
- Launched in 1995 by:
  - Alain Durand
  - Bernard Tuy
- Is today a legal association under French Law (1901)
  - Bernard Tuy, President
- For further information: http://www.g6.asso.fr

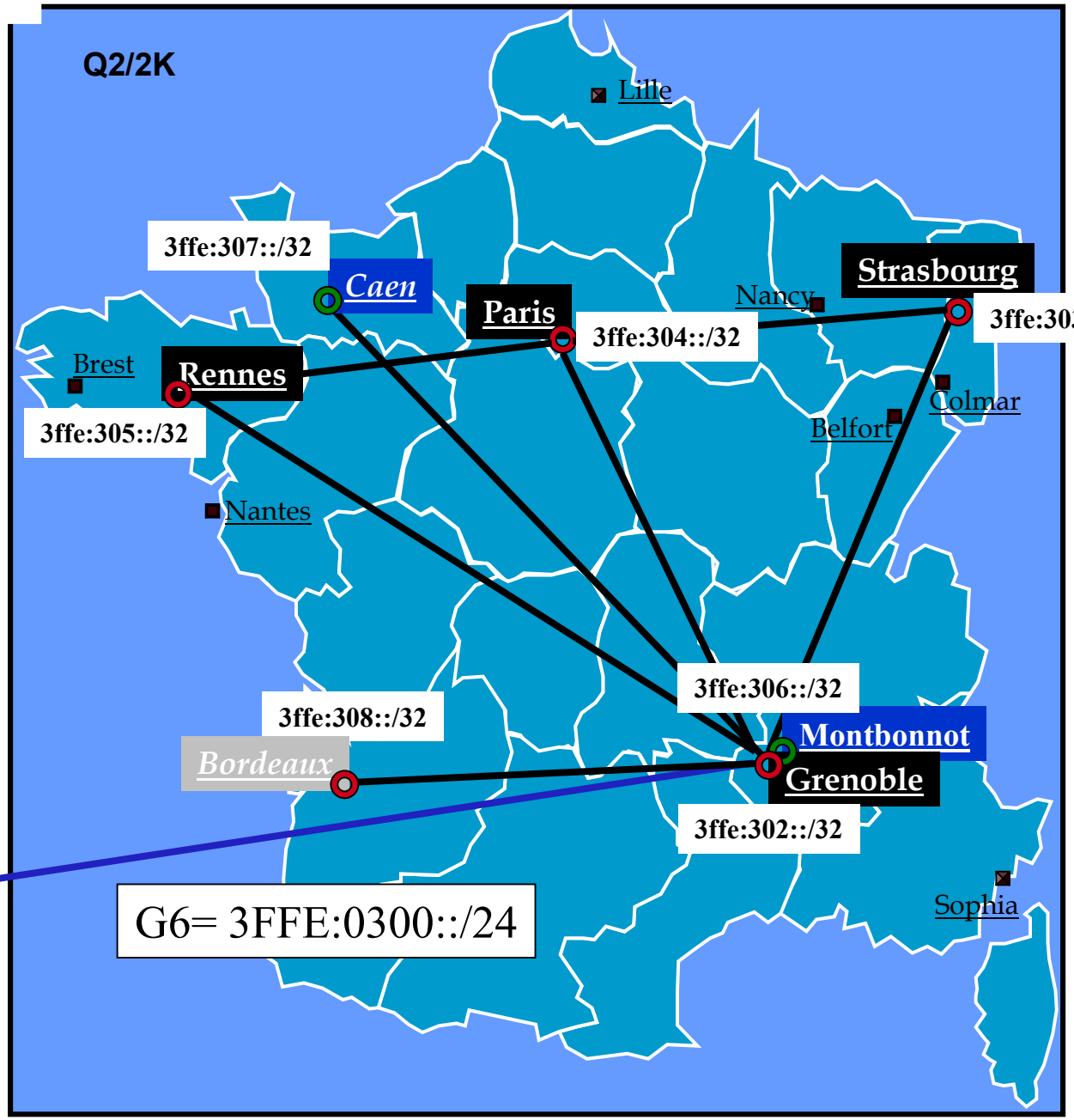
# G6 charter

- Share experience gained from experimentations
- Diffusion of IPv6 information
  - *Book published (O'Reilly)*
    - *« IPv6, Théorie et pratique », 3rd edition (March 2002)*
  - *Tutorials and trainings (ISPs, Engineers, netadmins, …)*
- Active in RIPE & IETF working groups
- Responsible of Renater IPv6 pilot service design

# Former G6-bone network

- Test infrastructure
- Connecting partners' testbeds
- Connected to the 6bone
  - French part of the 6bone
- Early testbed for a native IPv6 national infrastructure

# G6bone PoPs & addressing



# Sites connected to the G6 PoPs

**Paris:**
- Evry: Université INT
- Noisy-le-Gd: ESIEE
- Roquencourt: INRIA
- Saclay: CEA
- Lille: EUDIL
- Paris:
  - Aerospatiale/Matra
  - Brainstorm
  - CIE
  - CISI/ATRE
  - CNAM
  - ENST
  - Eurocontrol
  - Informatique P7
  - Institut Pasteur
  - Internatif
  - ISDnet
  - LAAS
  - LIP6 + Marocco
  - Logique P7
  - OpenTransit
  - Renater2 NOC
  - Urec/Cnrs
  - UVSQ
  - AFNIC

**Grenoble :**
- Echirolles: Bull
- Marseille : Ec. Sup. Mécanique
- Valbonne: Compaq
- Vanoise
- Grenoble :
  - Allied Signal
  - COSY
  - IMAG
  - MCS
  - Thomson-CSF/Detexis

**6Bone :**

| | | |
|---|---|---|
| ATT | JAnet | JOIN |
| INFN | Switch | Uni-C |

**Montbonnot:**
INRIA

**Strasbourg:**
- Belfort: Univ Technol.
- Colmar: IUT
- Nancy: Loria
- Strasbourg:
  - Univ. L. Pasteur
  - IUT

**Rennes:**
- Nantes
- Brest
- Tunisia

**Bordeaux:**
- Univ. Reaumur

**Caen:**
CNET

# The 6REN initiative

- 6REN:
  - IPv6 Research and Education Network
  - Initiative of ESnet
- Isn't a network but a coordination for IPv6 pre-production networks
  - Move from experimental status to operational
- Specifications are now standardized, implementations available …
- It's time to move on and Academic community could act as starters as for IPv4
- More on http://www.6ren.net


# 6REN

- Oct ’98 : first peerings (IPv6/ATM)
  - Esnet, CAIRN, Internet2/vBNS and Ca*net2
  - Then other networks joined
    - WIDE, …
  - BGP4+

# Large scale deployments

- Asia/Pacific
  - AARnet, Australia
  - CERNet, China
  - Internet Initiative Japan
  - NTTv6, Japan
  - WIDE, Japan
- North America
  - Abilene (Internet 2)
- EU
  - GéANT
  - All NRENs connected to Géant
  - Opentransit (FTLD)
  - 6Net
  - Euro6ix
- …

# Building the Internet v6

- Large backbones (Géant, Abilene, NTTv6, WIDE…) are already interconnected
- Géant
  - NRENs in the EU
  - Connections with Abilene and Esnet (USA) and with CANARIE (Canada)
  - TEIN : connection to Asia (Korea, Japan, …)
  - EUMEDIS : connection of mediterranean countries
  - ALICE: connection with South America
- Commercial ISPs
  - Opentransit
  - Sprint
  - Tiscali
  - Skanova …

# IPv6 Traffic Exchanges

- Most of the IXes offer IPv6 connectivity today
- 6TAP is a joint project of Canarie and Esnet:
  - Router located in StarTap (Chicago, IL)
- NSPIXP-6, IPv6-based Internet Exchange in Tokyo
- Amsterdam Internet Exchange (AMS-IX)
- SFINX, LINX …
- More information : http://www.v6nap.net/

# Deploying an IPv6 service: From G6bone to Renater IPv6 Network (6R3)…

# Agenda

- Academics' story with IPv6
- Toward a Production IPv6 service
  - Native support
  - Addressing
  - Naming
  - Routing
  - International connections
  - Connecting the Regionals
- Experimental IPv6 multicast service

# Academics' story with IPv6

At the beginning was … the G6

- « French » group experimenting IPv6 since 1995
- Academics and industrial partners sharing experience
- Became the G6 association (1901) in 01/2000
- All the activities are managed within the association
- ***It is not required to be a member to attend the meetings !***

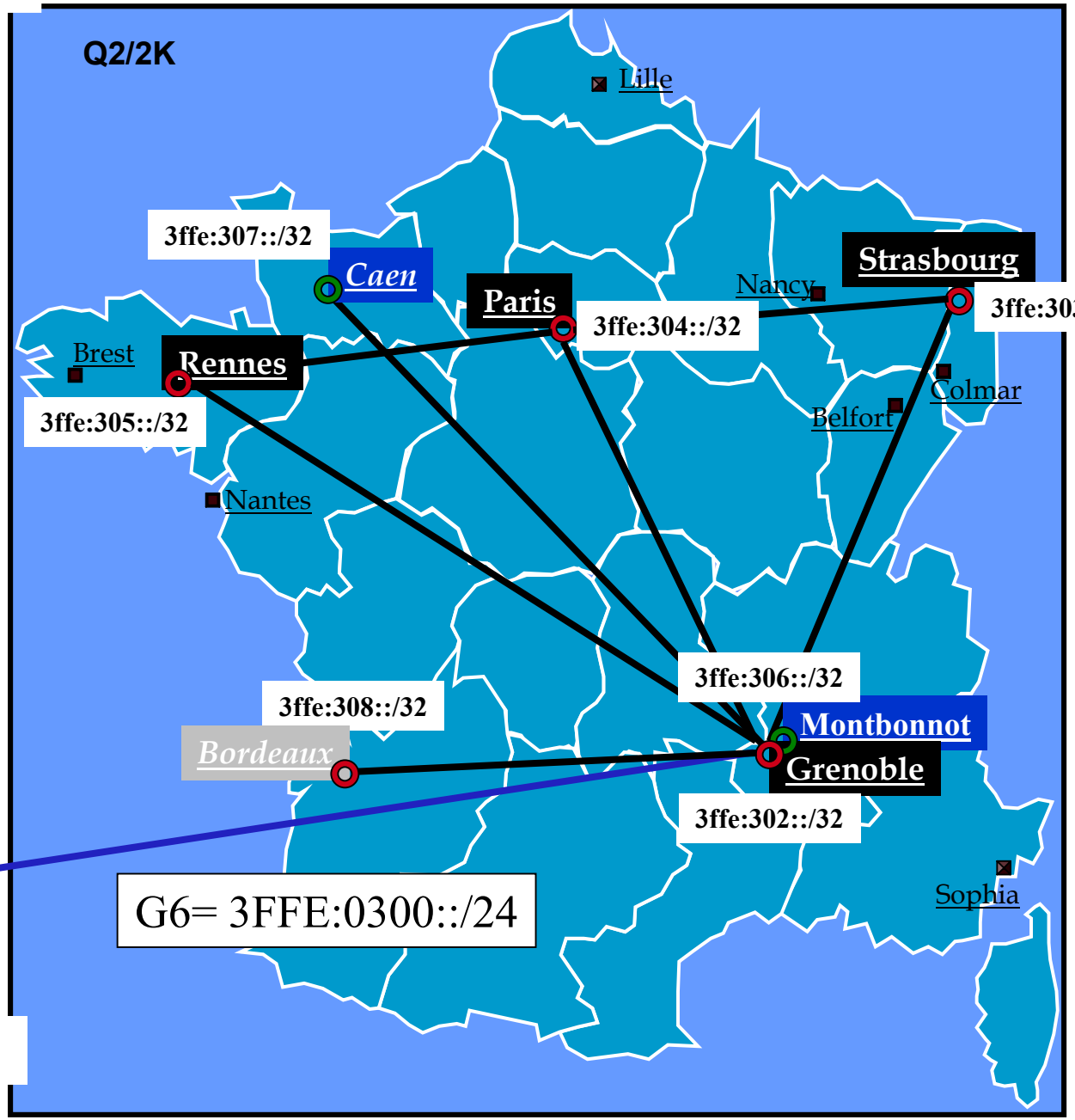
# Academics' story with IPv6

- G6 charter :
  - Experiment with the IPv6 protocol :
    - RNRT/RNTL
    - IST / Eureka …
    - G6
    - Renater / Aristote
  - Share experience with others
    - Web sites
    - « IPv6: théorie et pratique », O'Reilly ed. (3rd edition –March 2002)
    - Tutorials, conferences …
  - …
  - Info : http://www.g6.asso.fr/

# Academics' story with IPv6

- G6bone
  - The first IPv6 network in France (1996)
  - One of the 3 first IPv6 nodes starting the 6bone
    - UNI-C, DK
    - WIDE, JP
    - G6, FR
  - Tunneled network (v6inv4)
  - Hierarchical addressing from the beginning
    - Two-level topology : Regional Interconnects (RIs) + IPv6 sites
  - Static routing + RIPng …

# G6bone

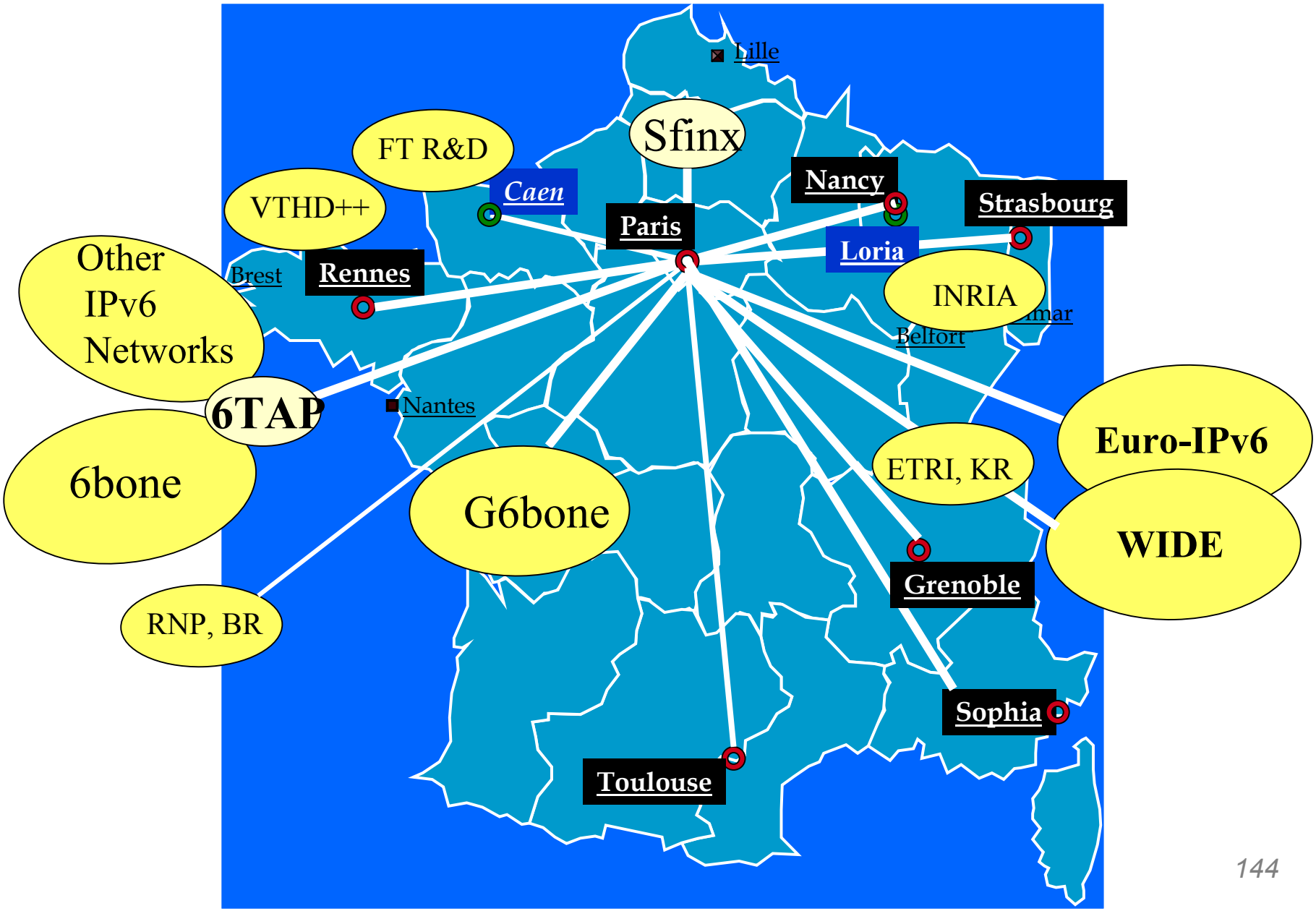

Q2/2K
Lille
3ffe:307::/32
Caen
Paris
Nancy
Strasbourg
3ffe:303::/32
3ffe:304::/32
Brest
Rennes
Colmar
Belfort
3ffe:305::/32
Nantes
3ffe:306::/32
3ffe:308::/32
Montbonnot
Bordeaux
Grenoble
3ffe:302::/32
Sophia
6Bone
G6= 3FFE:0300::/24

# Academics' story with IPv6

Then came Renater …

- IPv6 Pilot over Renater-2 (P6R2)
  - May 2000
  - A native IPv6 network
    - dedicated ATM VPN
  - Deploy the production addressing plan
    - July 1999 : first sTLA allocation
  - Same two-level topology as in G6bone
    - Academic sites
      - production addressing scheme
    - Industrial sites involved in research projects
      - 6bone addressing scheme
- Gain experience with a pre-production service

# Renater's IPv6 Pilot topology



# The Pilot experience

- Experience Using the protocol
  - Equipment
    - Cisco partnership
  - Addresses
    - Deploying a consistent scheme (/35) for the core and the sites
  - Routing
    - ISIS and BGP4+
- IPv6 resources allocation
  - Procedures and management
- IPv6 DNS
  - Deployment of the DNS service
  - Reverse zones delegation to RIs and end-users sites
- Management
  - IPv6 NOC within Renater-2 NOC
  - Management and monitoring tools
    - Set of looking glasses at the RIs

# Academics' story with IPv6

- Summary
  - Understand the technology
  - Deploy the network
  - Manage the whole thing
    - Technical resources
    - Human resources
    - Financial resources

# Towards a native IPv6 network

- G6bone was an overlay tunneled network
  - v6 traffic encapsulated in v4 packets
- « independent » from Renater's underlying infrastructure
- P6R2, IPv6 pilot was a VPN of ATM PVCs
- Goals
  - Have a production IPv6 network
    - In the core
    - Allow Regional and Metro Nets to deploy IPv6

# Additional goals

As production addresses became available

And sTLA expanded from /35 to /32

- Renumber the IPv6 pilot using a new addressing scheme
  - much simpler to be aligned on nibble boundaries !
- Keep a two-level hierarchy
  - A core backbone of Regional Interconnects (RI)
  - User sites connect to one or more RIs

# Additional goals (2)

- Transition period
  - Offer IPv6 connectivity via the new/native infrastructure
  - Keep the old infrastructure in place
  - Move step by step : no D day
- Gather non academic organizations in the G6bone addressing plan (3FFE:0300::/24)
  - Allow them to gain experience with IPv6 until commercial ISPs are ready
  - Have full IPv6 connectivity to the evolving Internet v6
- Connect the pilot to the Sfinx (Renater's IX)
  - Peer with ISPs and non academic organisms
- Provide IPv6 connectivity to
  - National projects (RNRT/RNTL)
  - European projects (IST, Esprit)
  - …

# Toward a Production IPv6 service

And now Renater-3 …

- Why a production-like IPv6 service ?
- ATM removed …
  - Move all network services on a unique topology
  - Do we want to forget about IPv6, IPv4 multicast … ?
- Need of IPv6 transport
  - Research projects using IPv6
  - Sites with native IPv6 network

  →install a native IPv6 core

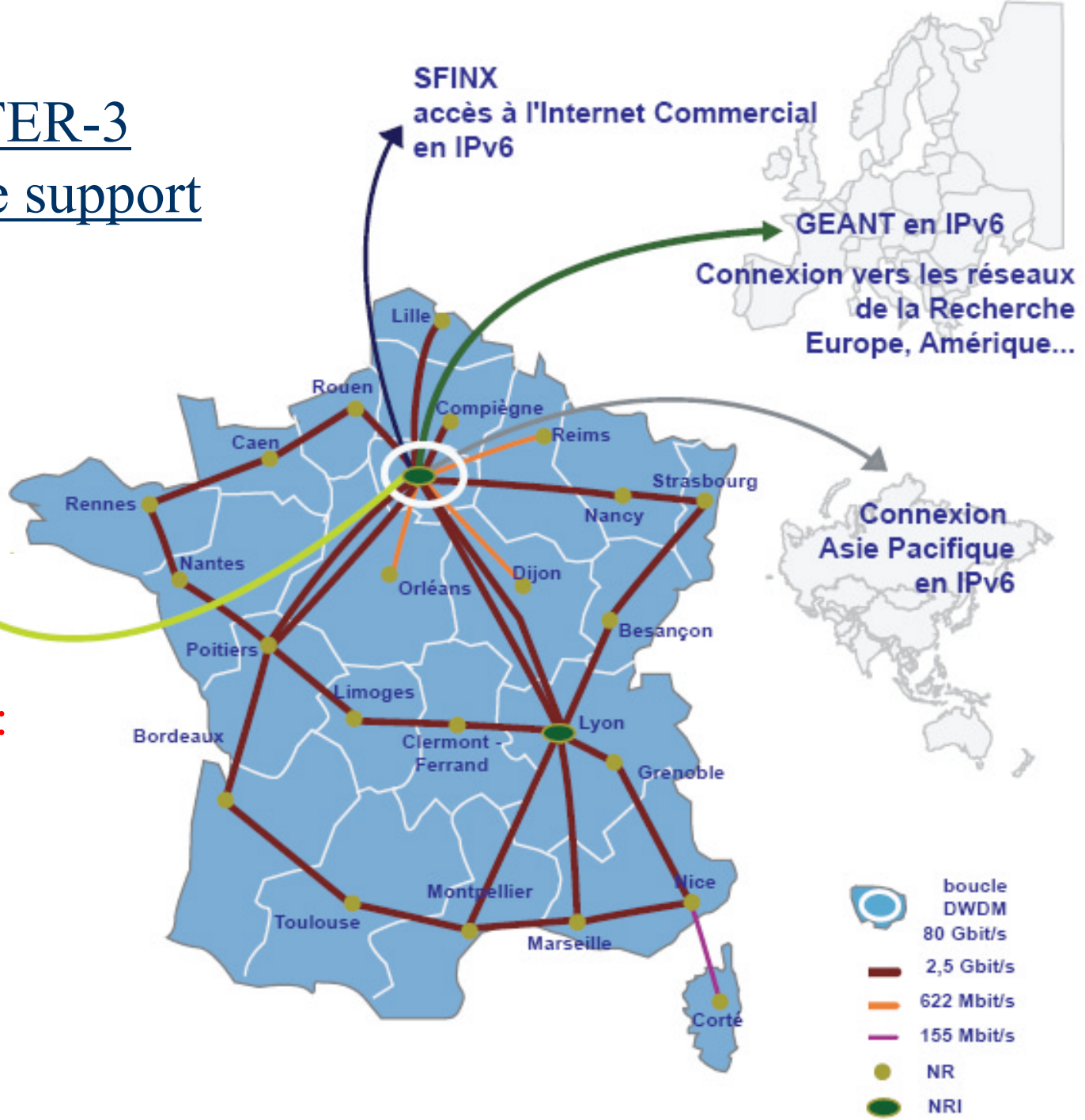
  →run both versions of IP the same way
- Manage the IPv6 service with the same operational quality as for IPv4

# RENATER-3
# IPv6 native support

40 NR
2,5 Gbits/sec
Service IP global

**Open Transit**

Un niveau égal de:
Performance
Disponibilité
Management
Support



# Renater 3: Native support

- 2.5 Gbits/s backbone
- 30 Regional Interconnects (RI)
- Native IPv6 support on all RIs
  - Dual stack backbone → IPv4 and IPv6
- Global IP Service
  - IPv4 unicast and multicast
  - IPv6 unicast
  - IPv6 and IPv4 carried without any distinction
- Experimental IPv6 multicast network
- Goal : achieve an equal level of
  - Performance
  - Availability
  - Management
  - Support

# Renater IPv6 addressing scheme

# IPv6 service in Renater-3

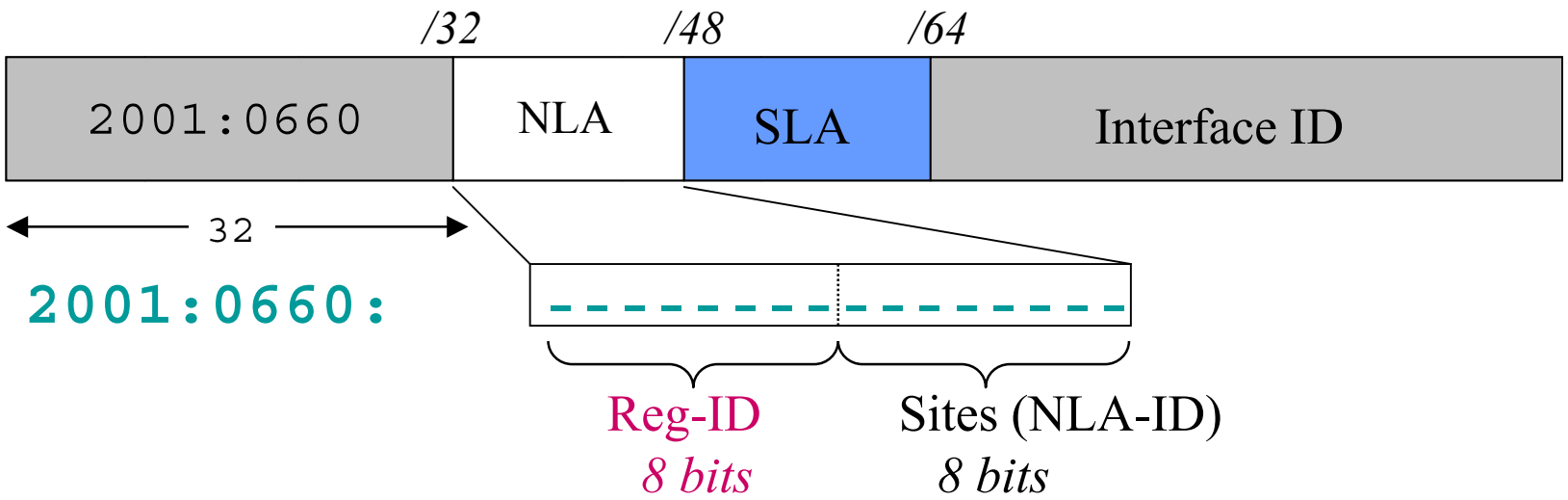
- Based on experience gained with the IPv6 Pilot deployment
- Principles for 6R3
  - /35 expands to /32 (2001:0660::/32)
  - Two-level hierarchy : core + access
  - Core are /40 allocated (easier to manage)
    - Each PoP identified with a Reg-ID
  - Sites are /48 (as recommended)
    - Identified with a NLA-ID

# Addressing scheme

- What do we need to number?
  - Regional Interconnects: /40
    - Reg-IDs allocation
  - Sites (labs, campuses …): /48
    - NLA-IDs allocation
    - 16 bits are reserved for the site topology
  - Interconnection networks
    - RI – sites
    - Renater – other IPv6 networks
      - Operational
      - Projects

# Addressing scheme (2)

sTLA = **2001:0660::/32**




```
2001:0660:3000:/40   Paris NRI
2001:0660:3300:/40   Paris Jussieu
2001:0660:4400:/40   Lille
2001:0660:5400:/40   Marseille (…)
```

```
2001:0660::/48
```

# Addressing scheme (3)

- Hierarchical addressing
- Renater: 2001:0660::/32 from RIR
- Regional RIs: /40 (reg-ID)
- Sites: /48 from /40 of RIs
  - NLA-IDs allocation
  - /48s aggregation to a single /40 for all sites connected to the same PoP
  - 16 bits are reserved for the site topology (“subnets”)

# Example

- Renater's sTLA: 2001:0660::/32
- RI Rennes : 2001:0660:7300::/40
- RI's local network : 2001:0660:7300::/48
- Sites connected to the RI
  - 2001:0660:7301::/48
  - 2001:0660:7302::/48
  - (…)

*Retour*

# Multihomed domains

- In IPv4, create lots of entries in default free routing tables
- In IPv6, interface will have several IPv6 addresses
  - Problem of source address selection is still under study

# Naming

- Direct DNS
  - Same domain name for IPv6 and IPv4
  - Ex : site.fr for IPv4 and IPv6
  - Just add an IPv6 entry for IPv6 addresses
- Reverse DNS
  - 0.6.6.0.1.0.0.2.ip6.int from the beginning
    0.6.6.0.1.0.0.2.ip6.arpa under deployment
  - Reverse zone's delegation of /48 allocated to the sites

# Routing & routing policy

- IGP: ISIS + iBGP
- EGP: e-BGP4+
- Route Reflectors
  - At each NRI
- In the backbone
  - /48 of sites aggregated in /40
- International advertisements
  - Announce Renater /32 sTLA
  - Accept /32 (or shorter) or /35 from ISPs
- Prefixes not allowed are filtered out
- Client sites connections
  - Their own choice: static, BGP4+
  - Not allowed to advertise more specific prefixes than /48s

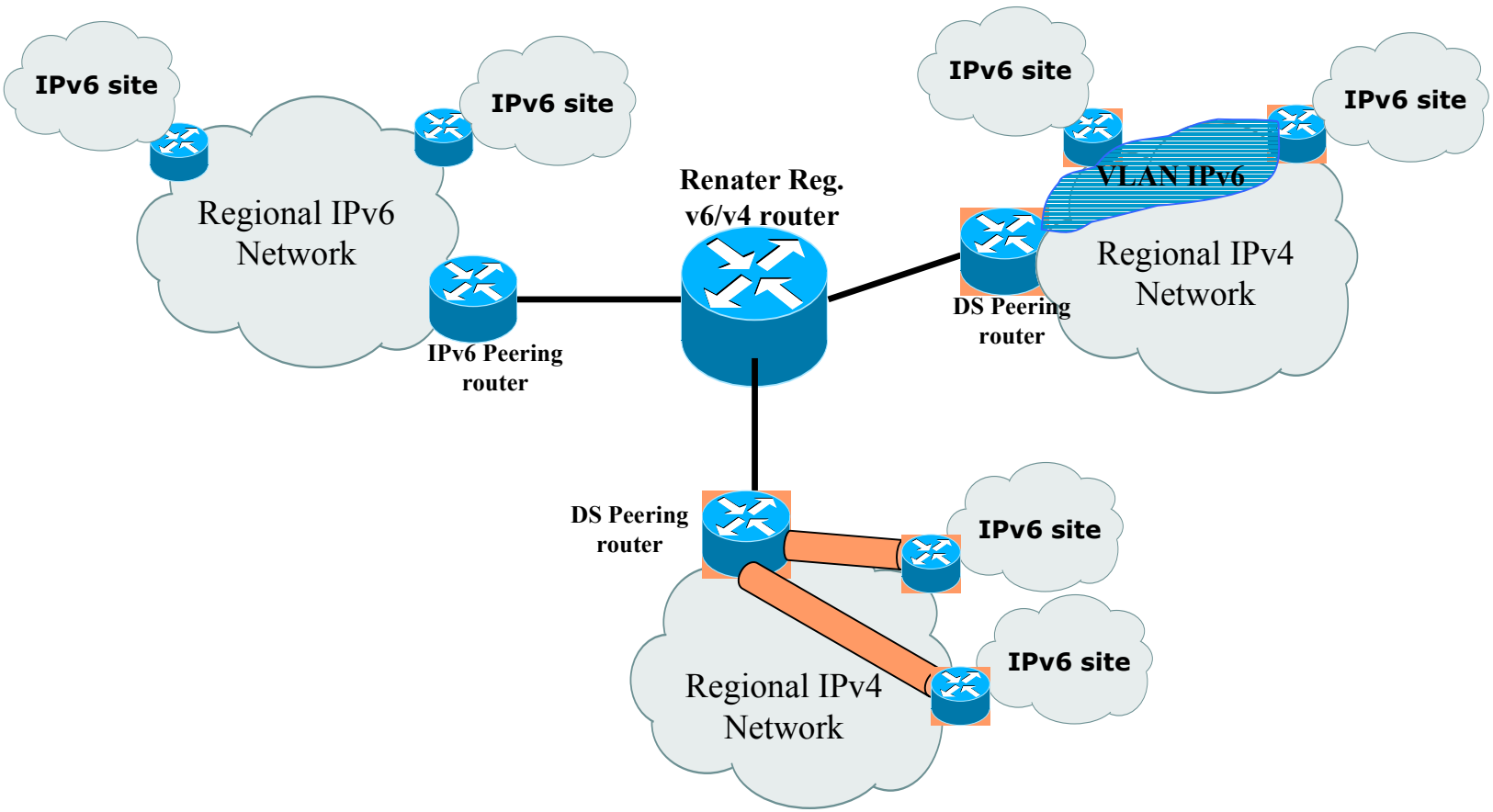
# Transition

- Renater's Backbone is native IPv6
- Some sites too
- BUT most of regional networks are not IPv6 capable yet …

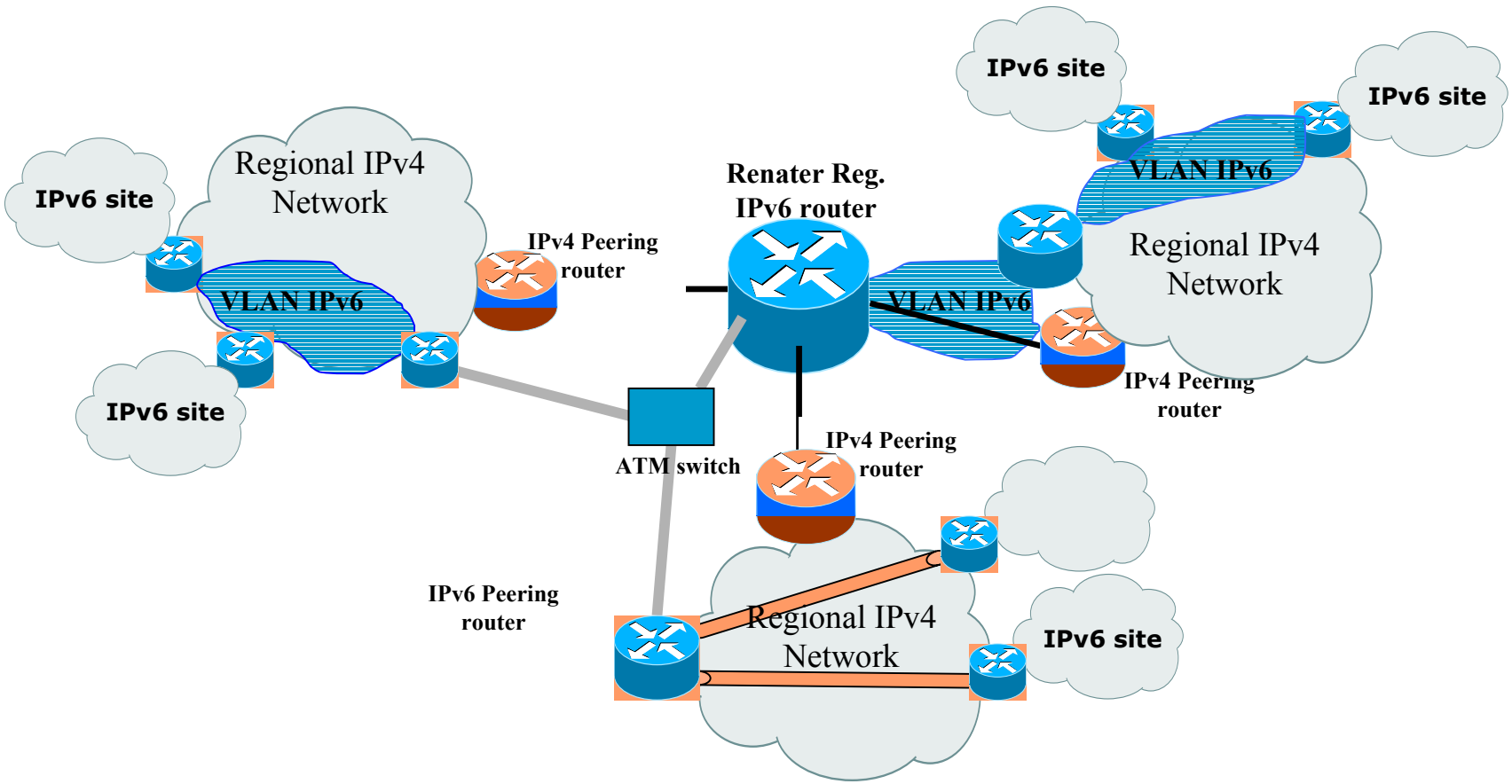
=> Install an equipment in each RN to connect IPv6

- Between regional router and sites:
  - VLANS
  - Tunnels
  - ATM PVC

# Scenario 1:
# Peering router is IPv6 capable



# Scenario 2:
# Peering router is IPv4 only



# Equipment

- Core routers are Cisco C124xx
  - POS + GEth interfaces …
- Edge routers are
  - Mainly Cisco's (C7xxx, C36xx, C65xx, …)
  - But also Juniper's M5, M10 …
  - 6WIND 6200
  - …

# Before to have IPv6 every where …

# Steps

- V6fy the network
- V6fy the OS
- V6fy the applications
- Communication between both worlds
  - Client/Server Mode
  - Full Internet Connectivity

# V6fy the OS

- FreeBSD:
  - 4.x : included
  - 3.x : «INRIA», KAME
- NetBSD:
  - -current : included
  - 1.4.2; «INRIA», KAME
- Linux:
  - 2.2 : included
- Apple
  - MacOS X : included
- Microsoft:
  - Windows 2000 (IPv6 Technology Preview)
  - Windows XP (included)
  - 9x : Trumpet stack
- Helwett Packard
  - Compaq
- Solaris 8: included
- AIX 4.3: included
- Cisco IOS 12.2T
- Juniper: JunOS
- 6WIND: 6OS

See http://playground.sun.com/ipng/

# Steps

- V6fy the network
- V6fy the OS
- V6fy the applications
- Communication between both worlds
  - Client/Server Mode
  - Full Internet Connectivity

# RFC 1933 (April 1996)

- Used to v6fy applications
- Recompile applications to use IPv6 API
- Stay compatible with IPv4 applications
- Configuration of a dual stack
  - use of IPv4 mapped addresses
- Generate IPv6 traffic when possible

# IPv6 API

```
15,16d14
< extern const struct in6_addr in6addr_any;
<
22c20
< struct sockaddr_in6 sin;
---
> struct sockaddr_in sin;
26,30c24,25
< #ifdef SIN6_LEN
< sin.sin6_len = sizeof(sin);
< #endif
< sin.sin6_family = AF_INET6;
< sin.sin6_addr = in6addr_any;
---
> sin.sin_family = AF_INET;
> sin.sin_addr.s_addr = INADDR_ANY;
36,37c31,32
< sin.sin6_port = sp->s_port;
< if ((sock = socket(sin.sin6_family, SOCK_STREAM, 0)) < 0) {
---
> sin.sin_port = sp->s_port;
> if ((sock = socket(sin.sin_family, SOCK_STREAM, 0)) < 0)
```

# IPv6 API

- Few changes in the socket calls
  - Structures
  - Names
- More changes in DNS calls

- The source code MUST be available

# Applications

- MUAs, MTAs,
- Web browsers & servers,
- FTP, SSH, Telnet
- Videoconferencing tools, streaming, …
- Editors, Games, …
- Management and monitoring tools
- …

⇒ we started a list of non compliant applications !

# Steps

- V6fy the network
- V6fy the OS
- V6fy the applications
- Communication between both worlds
  - Client/Server Mode
  - Full Internet Connectivity

# Coexistence / Integration Mechanisms

# Transition/Integration (agenda)

- Dual stack IPv4-IPv6
- Tunneling mechanisms
- Translation mechanisms

- Deployment strategies

- Vocabulary is important
  - Transition, migration …
  - Deployment, coexistence and integration !

# Coexistence / Integration Mechanisms

*Dual stack IPv4/IPv6*

# Dual stack

- IPv4 and IPv6 running together
- 2 scenarios:
  - Existing network
  - New network

# Drawbacks

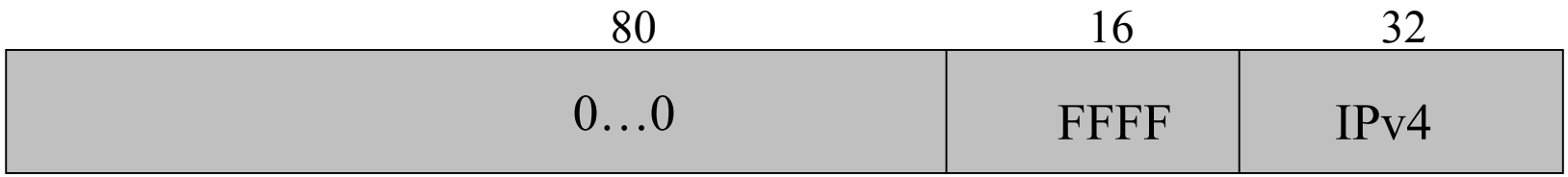
- Dual stack configured for IPv4 and IPv6
- Doesn’t solve the lack of IPv4 addresses
- Routers need to be configured for both versions of IP

=> 2 sets of routing tables

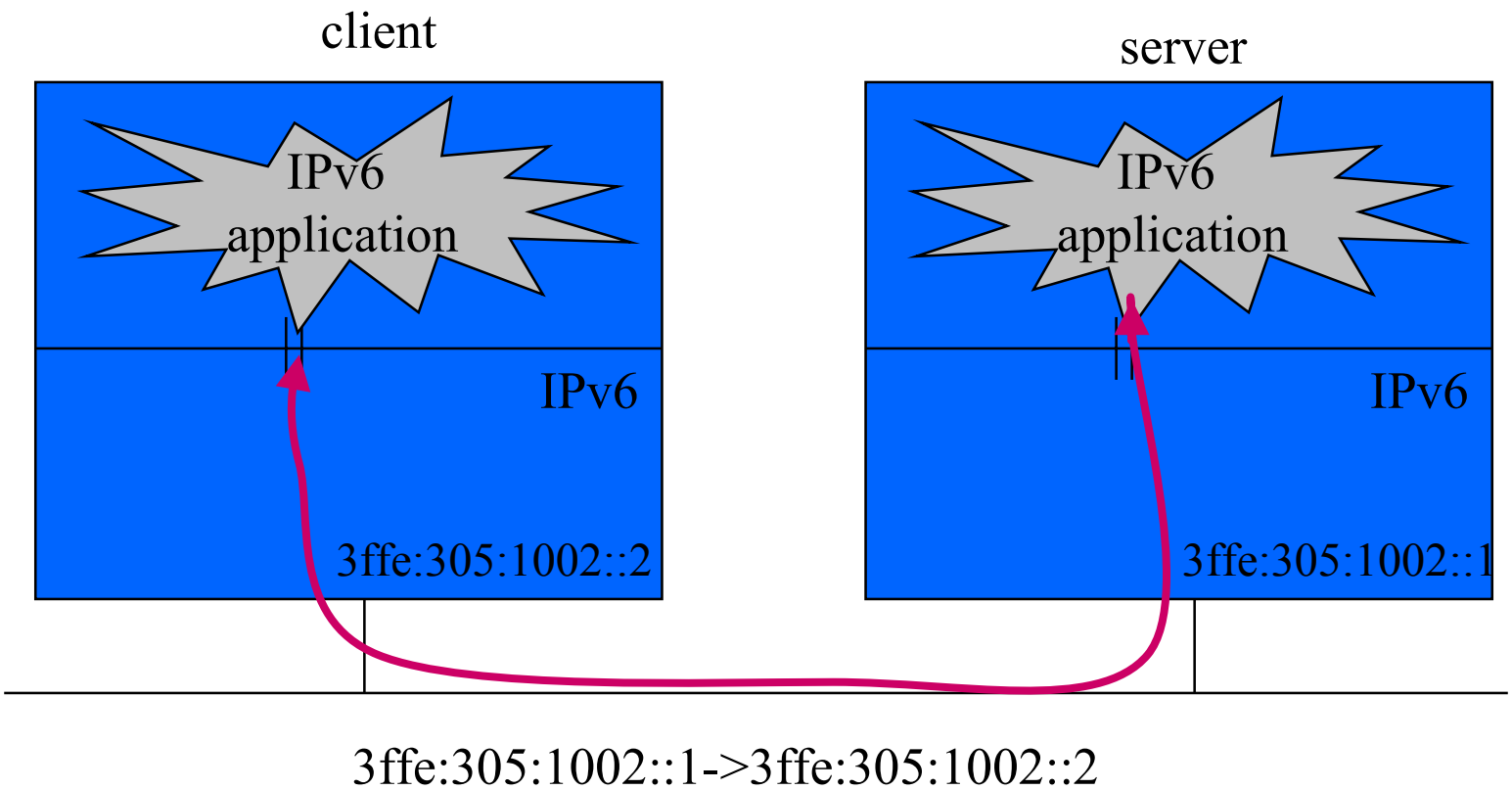
- RFC 1933 obsoleted with RFC 2893

# IPv4 Mapped addresses

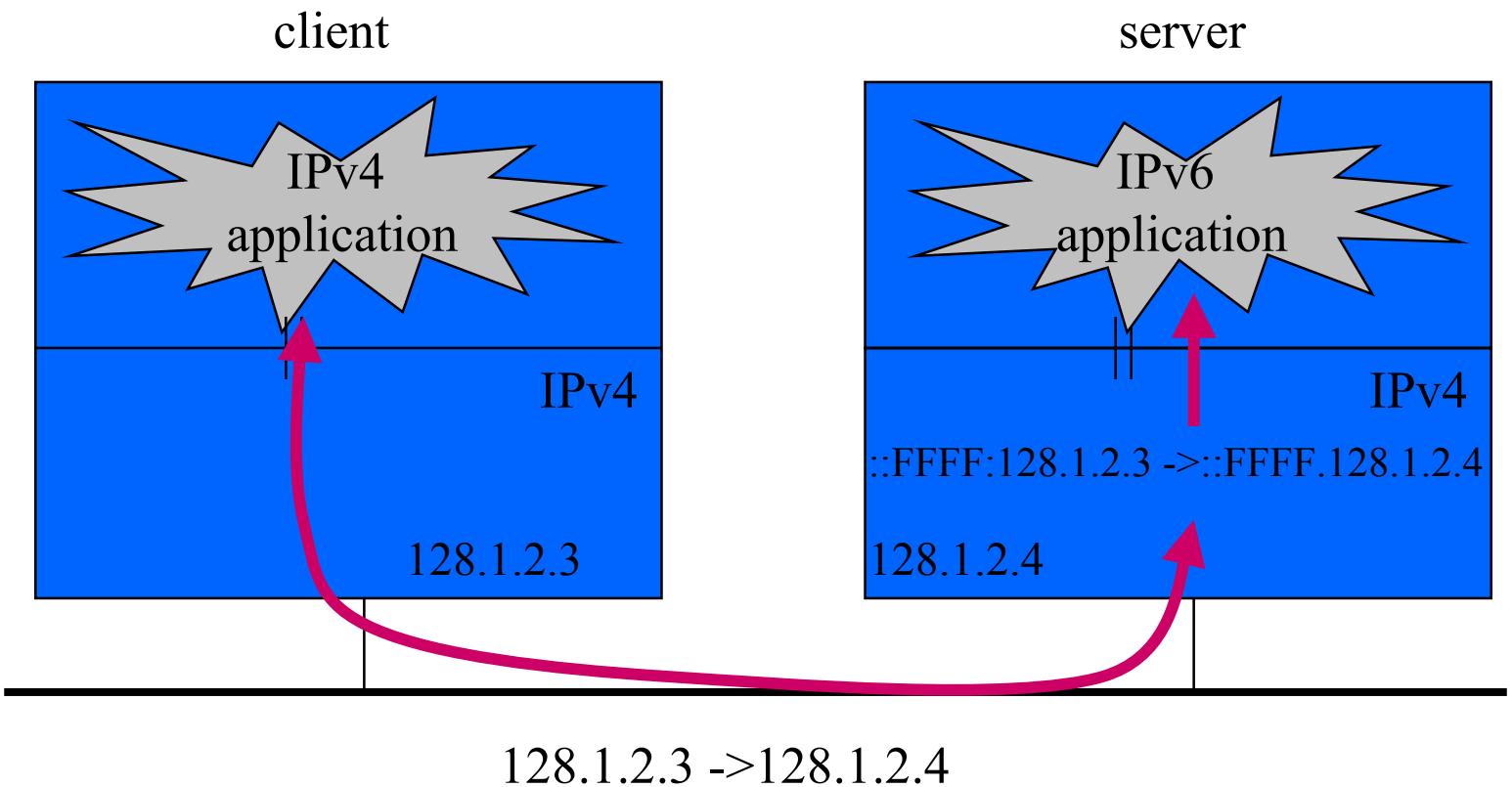
- IPv6-only applications can use IPv4 transport



# IPv4 Mapped Addresses



# IPv4 Mapped Addresses (continued)



# Coexistence / Integration Mechanisms

*Tunneling*

# Tunnelling facility

- Configured tunnels
  - widely deployed in the 6bone
  - used to connect two sites
  - require manual configuration

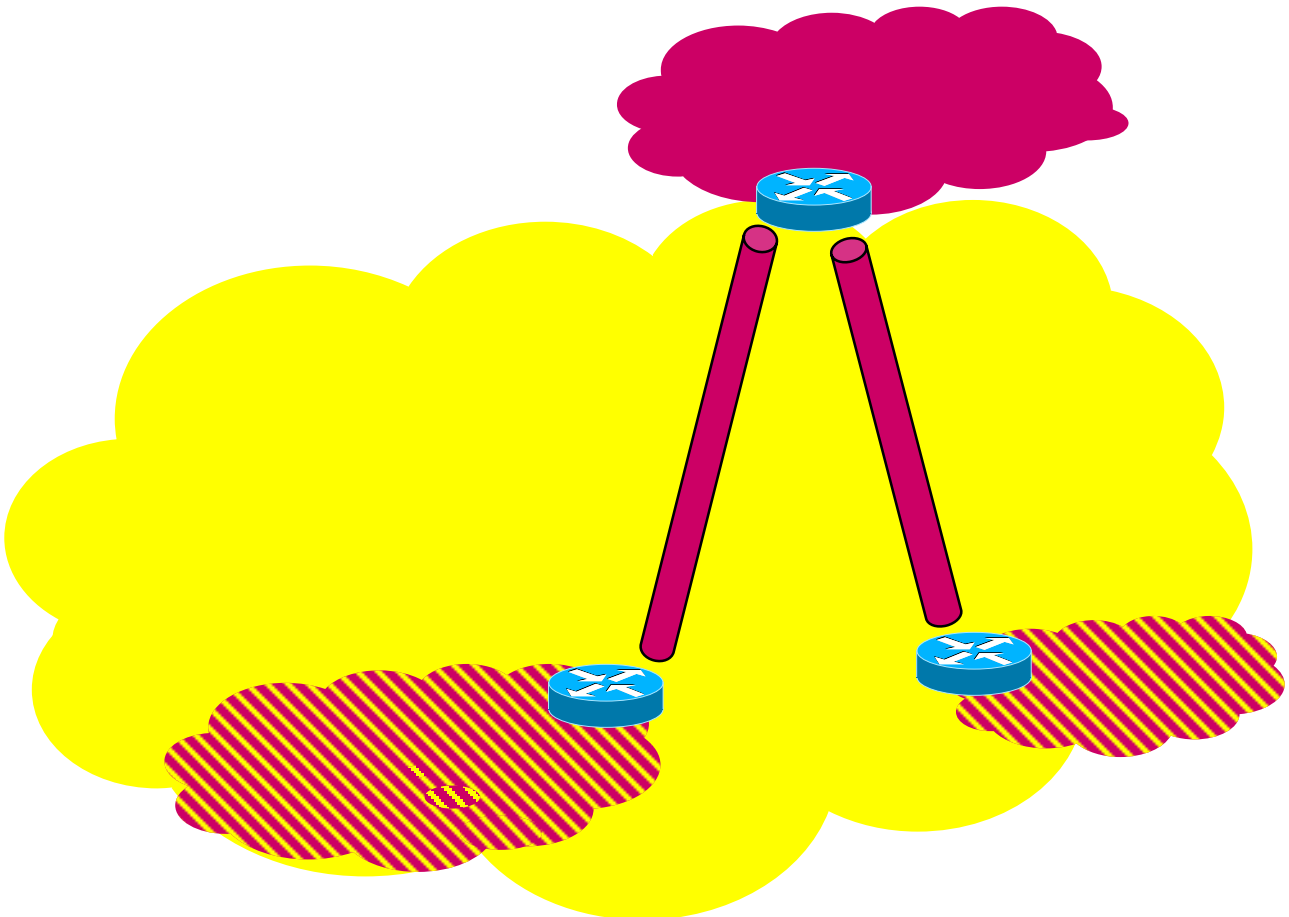
- Automatic tunnelling
  - 6to4
  - Tunnel Broker
  - ISATAP

# IPv6 in IPv4 configured tunnel

- Put IPv6 packet in IPv4 payload
- IPv4 protocol 41 means data = IPv6 packet
- Underlying infrastructure becomes transparent
- Makes it possible to connect to IPv6 network over an IPv4 link

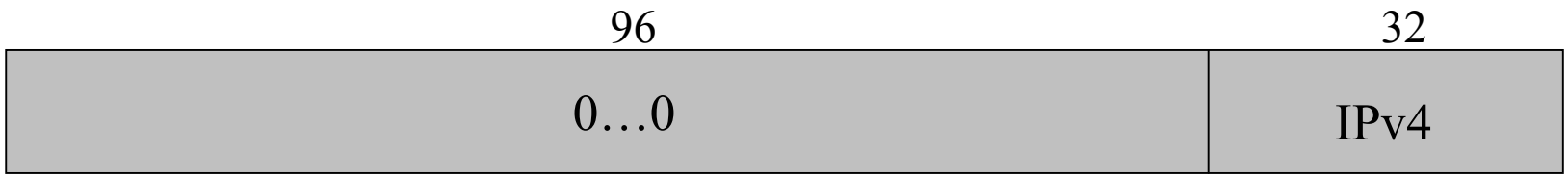
- Need to specify tunnel end points
- Can give addresses on IPv6 logical link

# 6bone



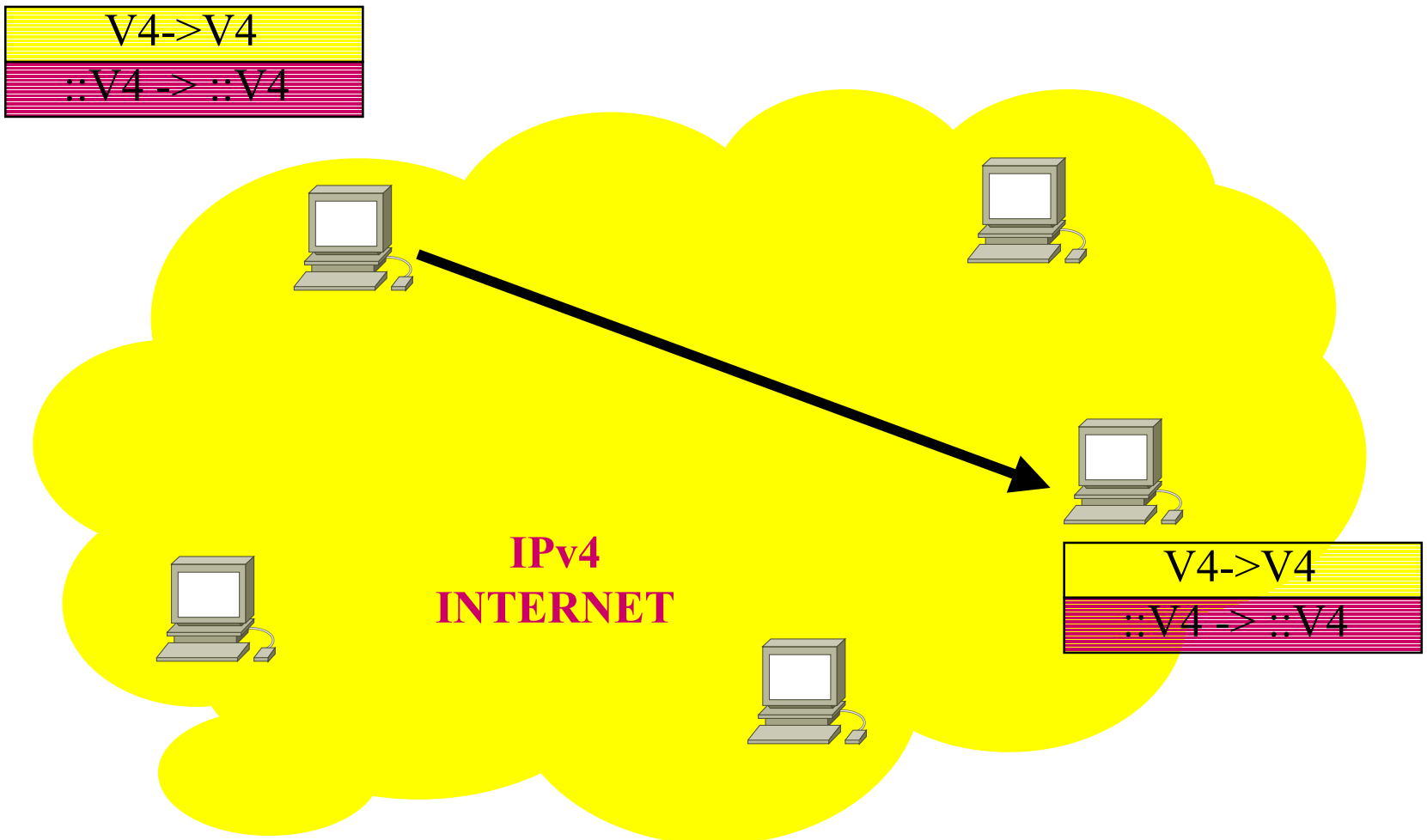
Create a virtual topology over the IPv4 network with configured tunnels

# IPv4 Compatible Addresses




- Used at the beginning for transition with IPv4
- Allows encapsulation of IPv6 packet into IPv4 packets
- Dynamic tunneling

# IPv4 Compatible Addresses

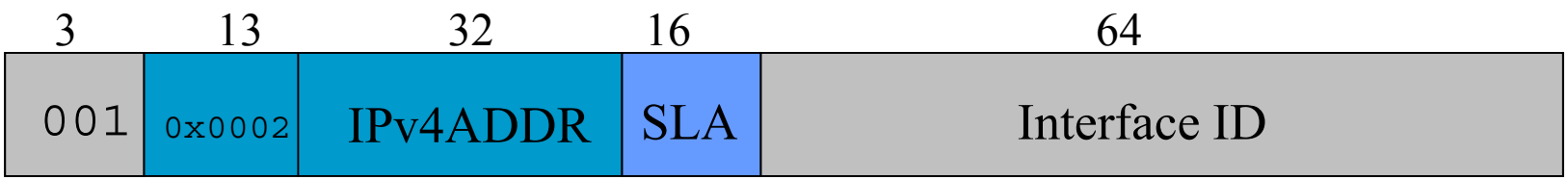


# IPv4 Compatible Addresses

- Like IPv4 addresses with 96 bits to 0
  - Used when only a few IPv6 hosts were on the Internet
  - Don't learn how to manage an IPv6 network
- Need more sophisticated networks
  - E.g the 6bone mainly use static tunnels between routers
- NOT USED ANYMORE

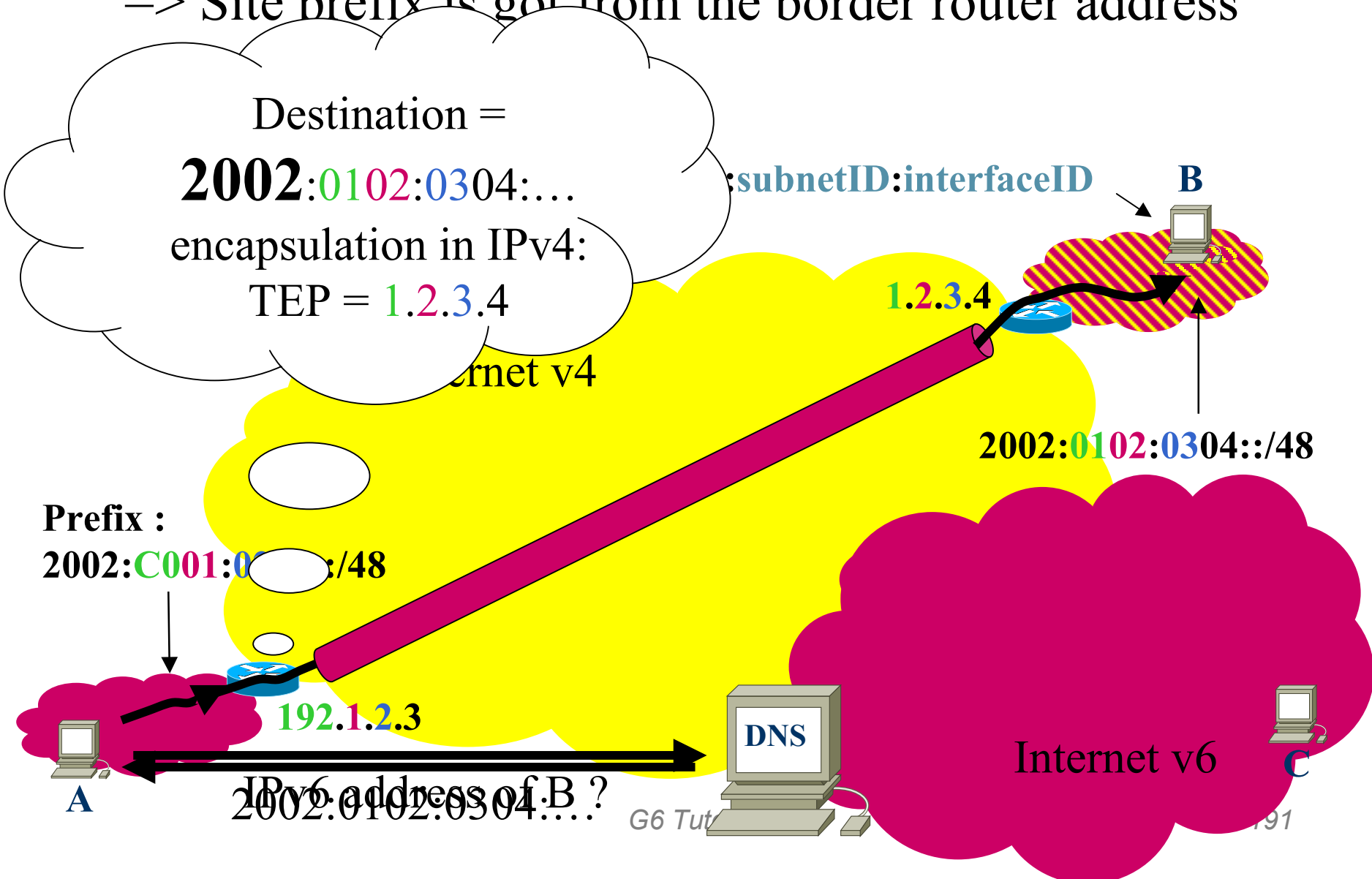
# 6to4 (RFC 3056 PS)

- Another way to build a tunneled infrastructure
- Simple configuration (no need to configure static tunnels)
- Use a special address plan
  - Prefix: 2002::/16

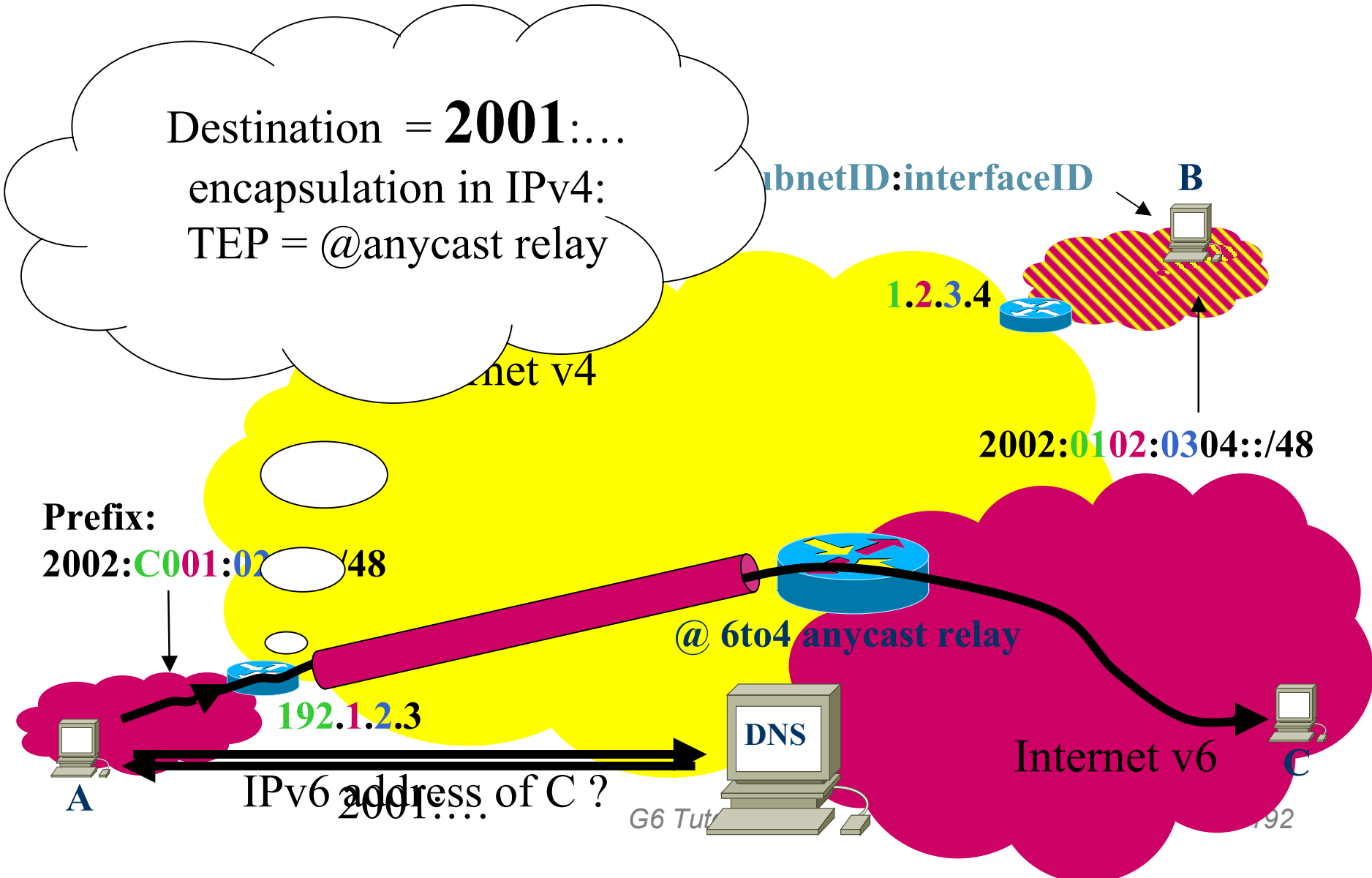


# 6to4

=> Site prefix is got from the border router address



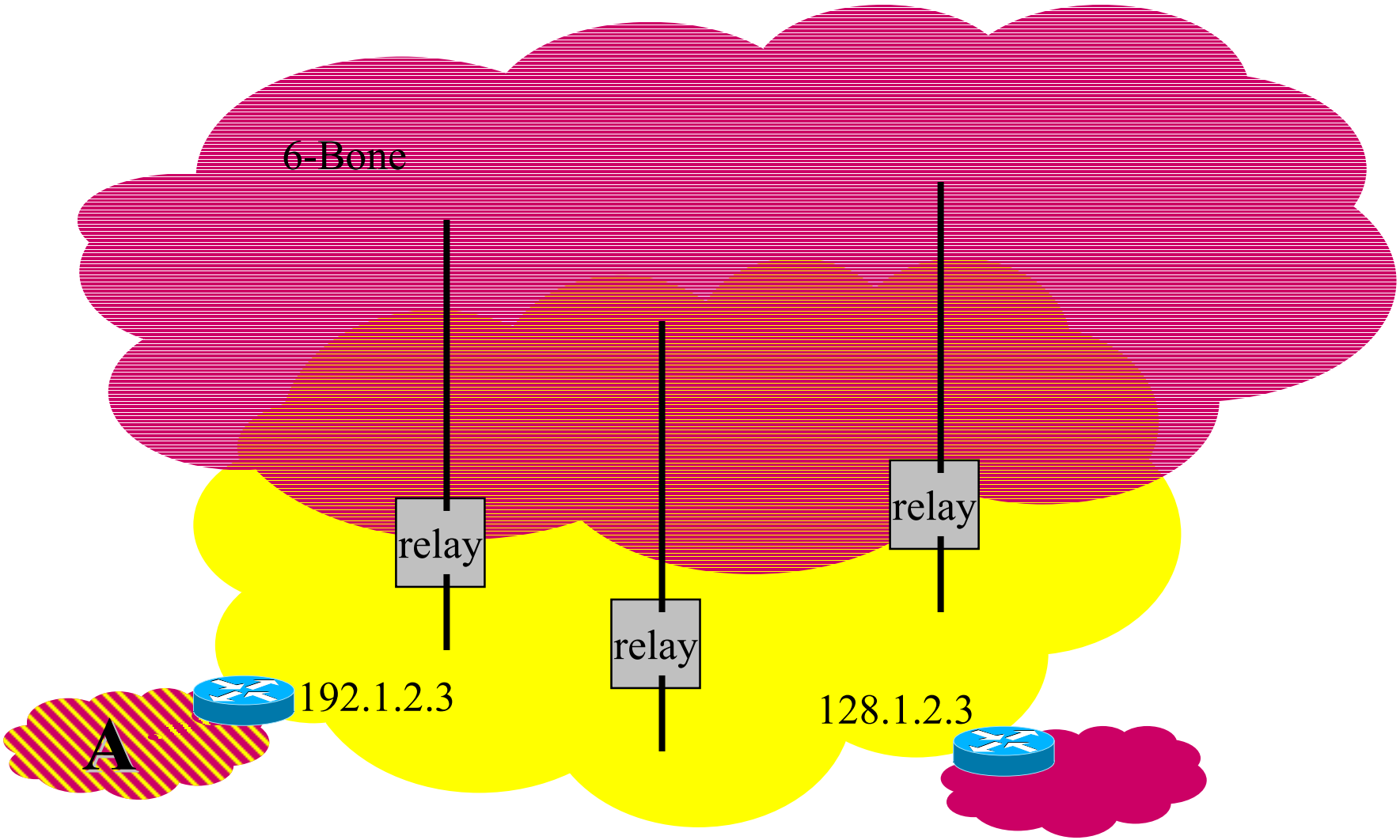
# 6to4


Destination = 2001:…
encapsulation in IPv4:
TEP = @anycast relay
ubnetID:interfaceID
B
1.2.3.4
net v4
2002:0102:0304::/48
Prefix:
2002:C001:02
/48
@ 6to4 anycast relay
192.1.2.3
DNS
Internet v6
C
A
IPv6 address of C ?
2001:…

# 6to4: Interaction with the 6bone

- If one node has a 6to4 address and the other one has both a 6to4 and global IPv6 addresses
  - Select 6to4 address
- If both have 6to4 and global IPv6 addresses
  - Global IPv6 should be selected

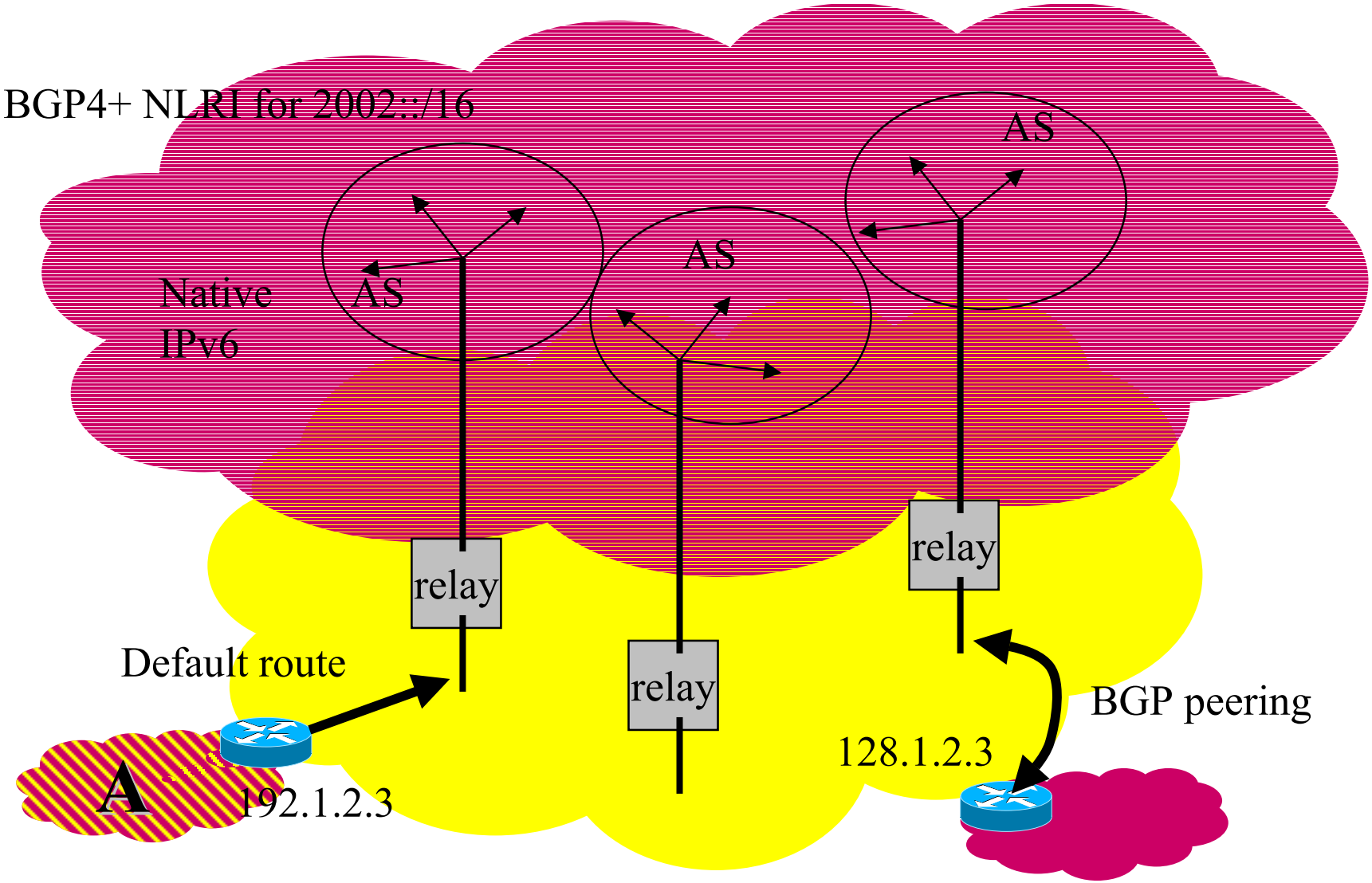
# 6to4: Interaction with the 6bone



# 6to4: Interaction with the Internet v6

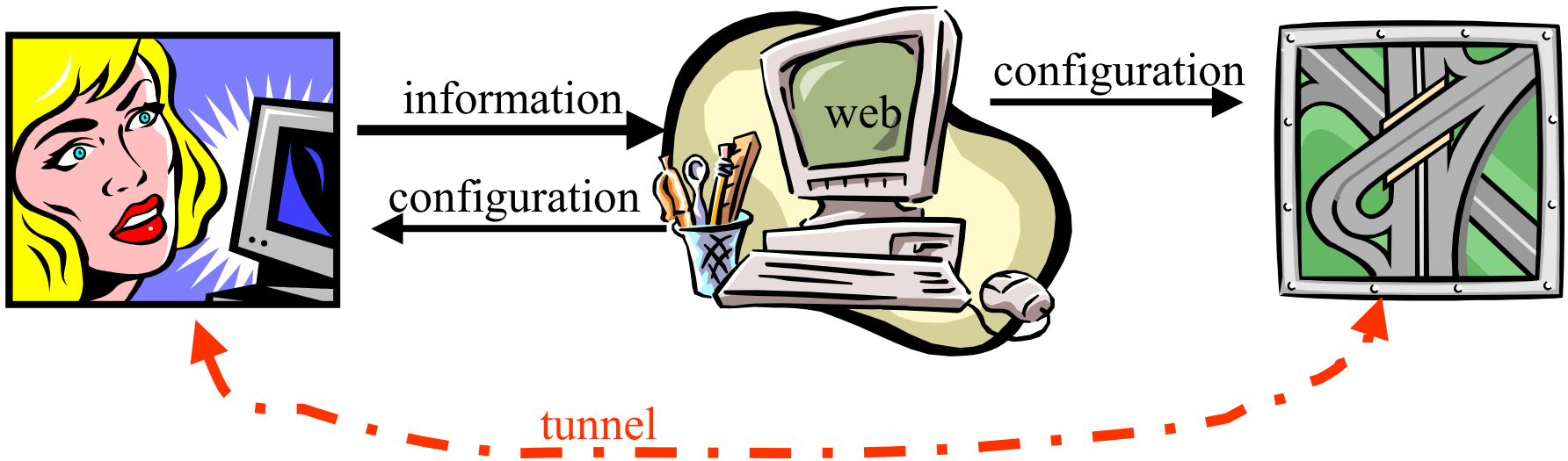
- Relays are just routers with one interface on the native IPv6 network and one on the 6to4 network.
- If the relay can be announced through an interior gateway protocol:
  - Doesn't change anything
- More complex, when an exterior protocol is used.

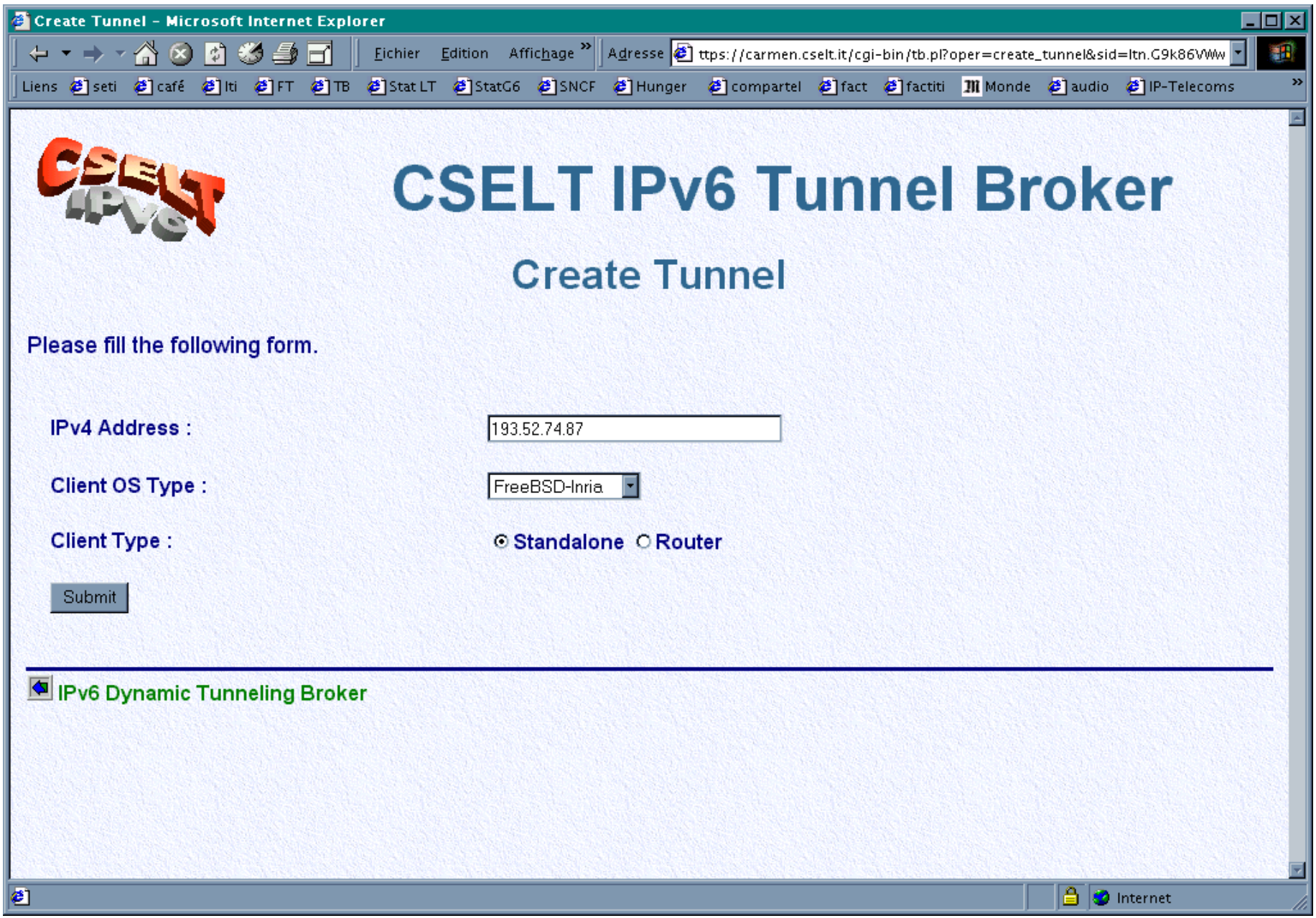
# 6to4: Interaction with the Internet v6 (2)

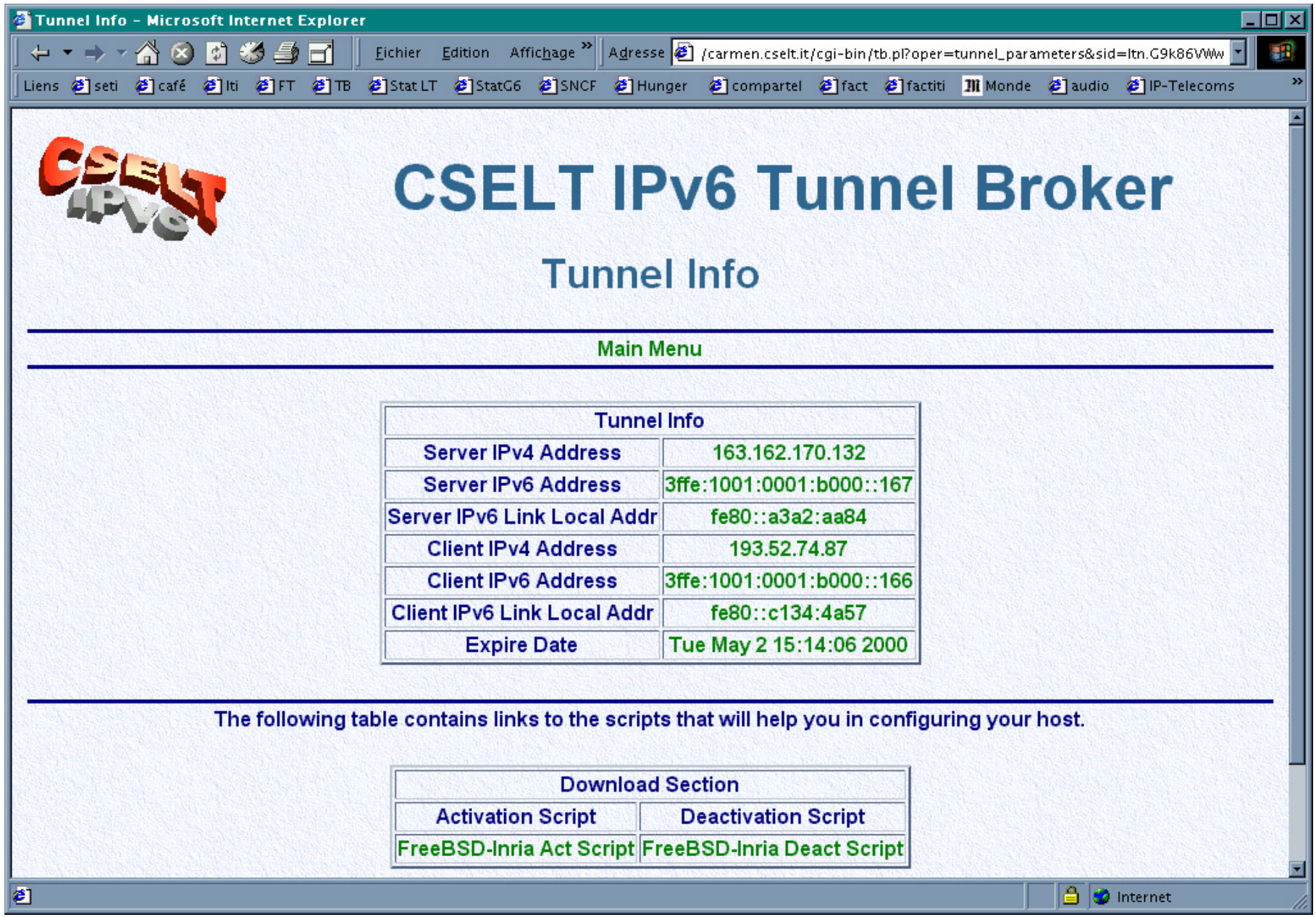


# Tunnel Brokers

- Simplify/Allow the construction of IPv4 tunnels.
- Use of a web page




Create Tunnel - Microsoft Internet Explorer
Fichier Edition Affichage
Adresse ttps://carmen.cselt.it/cgi-bin/tb.pl?oper=create_tunnel&sid=ltn.G9k86VWw
Liens seti café Iti FT TB Stat LT StatG6 SNCF Hunger compartel fact factiti Monde audio IP-Telecoms
CSELT IPv6
CSELT IPv6 Tunnel Broker
Create Tunnel
Please fill the following form.
IPv4 Address :
193.52.74.87
Client OS Type :
FreeBSD-Inria
Client Type :
Standalone
Router
Submit
IPv6 Dynamic Tunneling Broker
Internet


Tunnel Info - Microsoft Internet Explorer
Fichier Edition Affichage
Adresse /carmen.cselt.it/cgi-bin/tb.pl?oper=tunnel_parameters&sid=ltn.G9k86VWw
Liens seti café lti FT TB Stat LT StatG6 SNCF Hunger compartel fact factiti Monde audio IP-Telecoms
CSELT IPv6 Tunnel Broker
Tunnel Info
Main Menu
Tunnel Info
Server IPv4 Address 163.162.170.132
Server IPv6 Address 3ffe:1001:0001:b000::167
Server IPv6 Link Local Addr fe80::a3a2:aa84
Client IPv4 Address 193.52.74.87
Client IPv6 Address 3ffe:1001:0001:b000::166
Client IPv6 Link Local Addr fe80::c134:4a57
Expire Date Tue May 2 15:14:06 2000
The following table contains links to the scripts that will help you in configuring your host.
Download Section
Activation Script
Deactivation Script
FreeBSD-Inria Act Script
FreeBSD-Inria Deact Script
Internet

# Coexistence / Integration Mechanisms

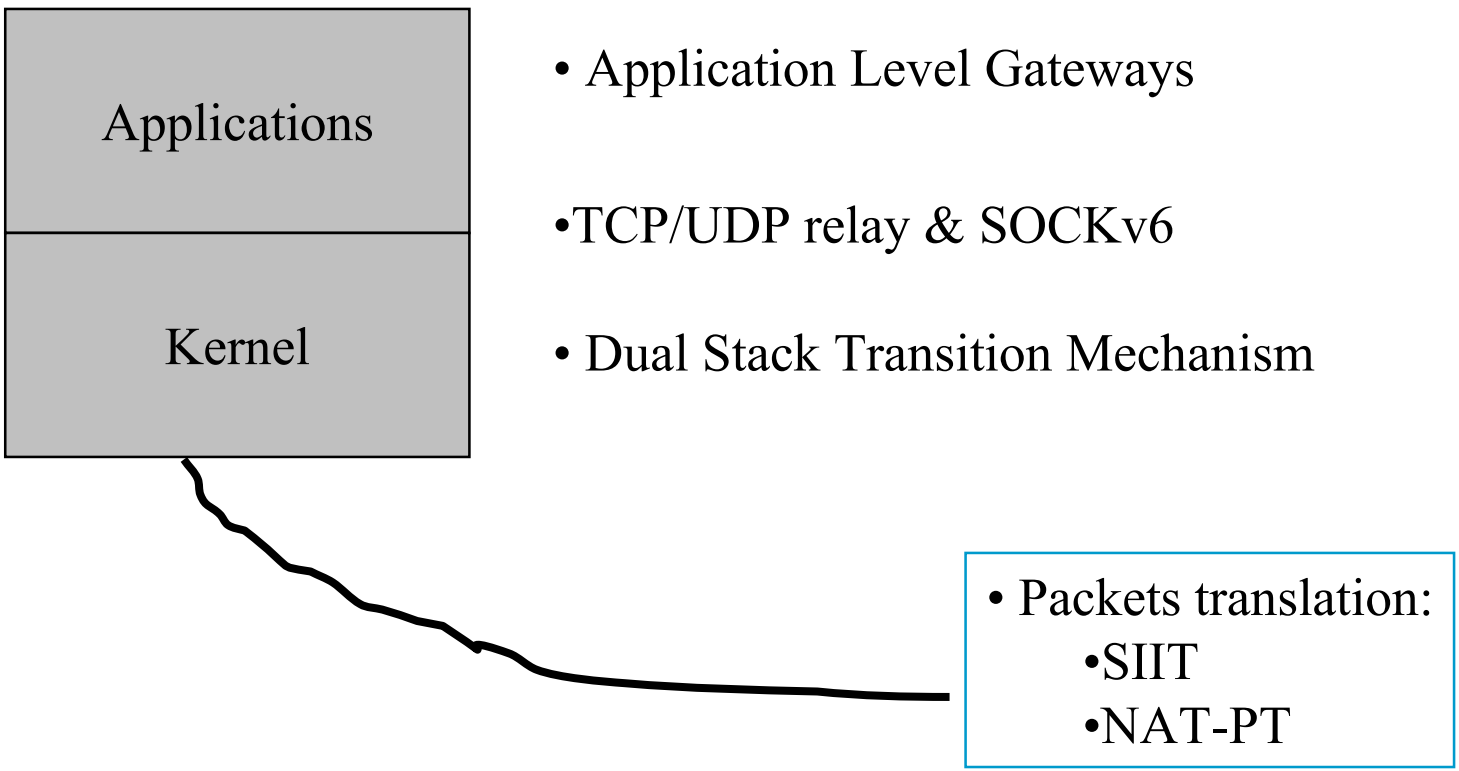
*Translation Mechanisms*

# Interoperability tools: Translators

- IP level
  - SIIT (Stateless IP/ICMP Translation)
  - NAT-PT (Network Address Translation-Protocol Translation)
  - BIS (Bump In the Stack)
- TCP level
  - TCP-relays
  - SOCKS
- Application level
  - Bump in the API
  - proxies

# Mechanisms for coexistence

- Different approaches

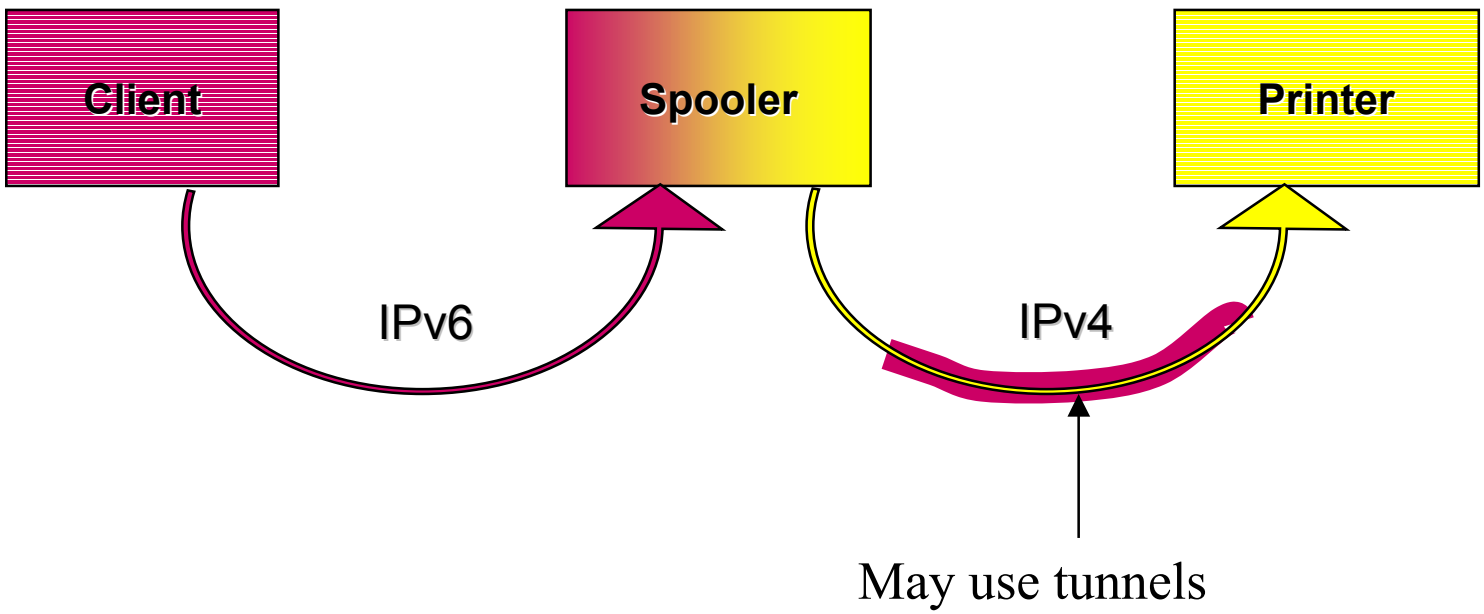


# Application Level Gateways

- May be used for a large majority of common applications:
  - E-mail (POP3, IMAP, SMTP)
  - Web (proxies)
  - Printer (spoolers)
  - DNS : relay (may change the RR type)
- Reduce IPv4 traffic inside a domain

# Application Level Gateways

- For example : an *old* printer without an IPv6 stack



# BIS: Bump In The Stack (RFC 2767 informational)

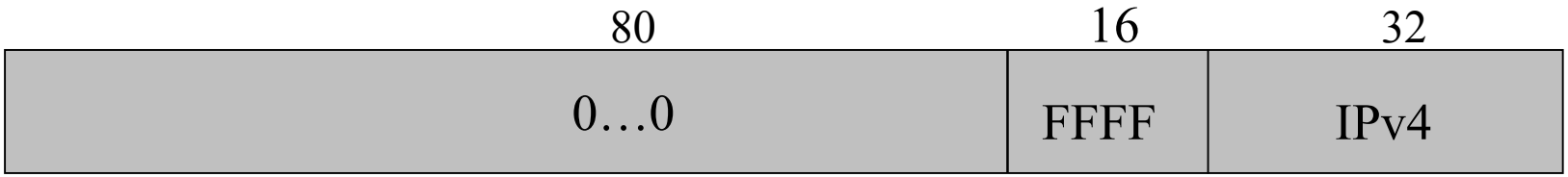
- v6fy application without recompilation
- Equivalent to protocol translator in each host
- Same problems as NAT (if the application sends addresses in data)
- Used in Trumpet IPv6 Stack

# RFC 2765 PS: Stateless IP/ICMP Translation (SIIT)

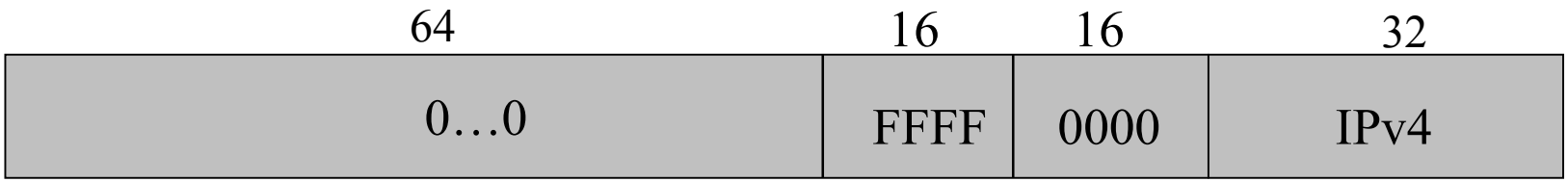
- Suppress the v4 stack
- Translate the v6 header into a v4 header on some point of the network
  - Routing can direct packet to those translation points.
- Translate ICMP from both worlds
- No State in translators (≠ NAT)

# SIIT

- V6 header contains:
  - IPv4 mapped addresses

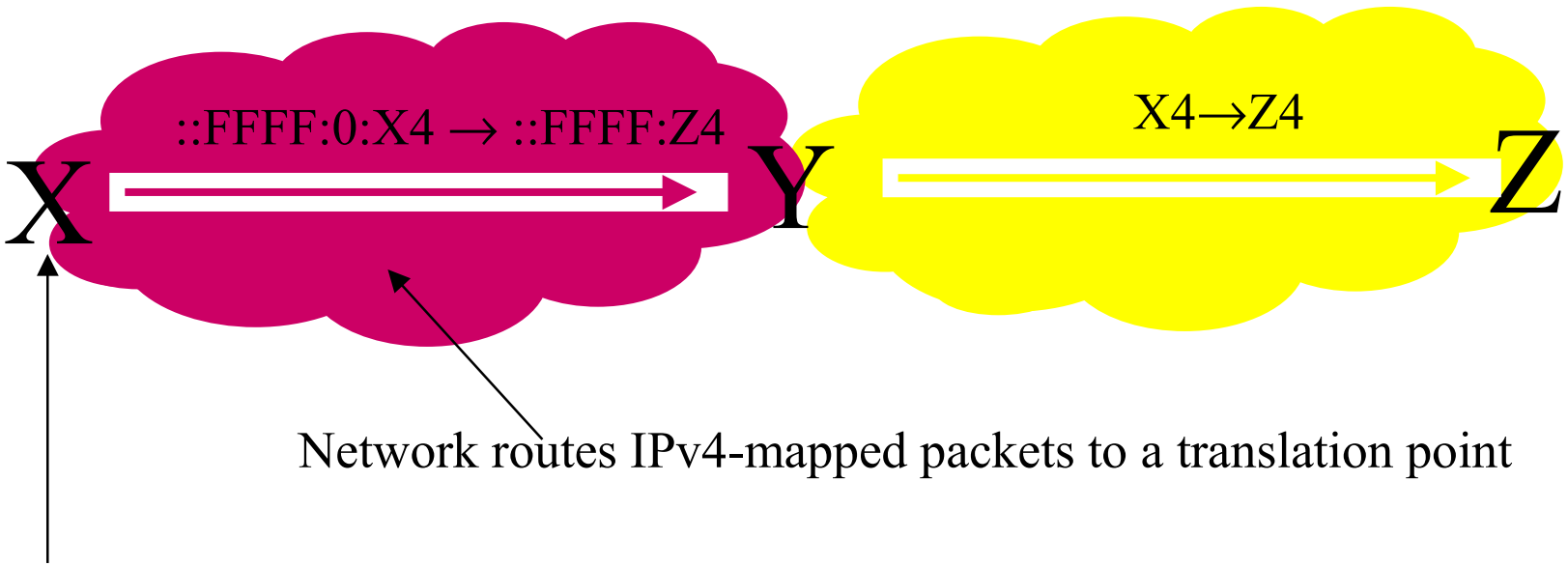



  - IPv4 translated addresses

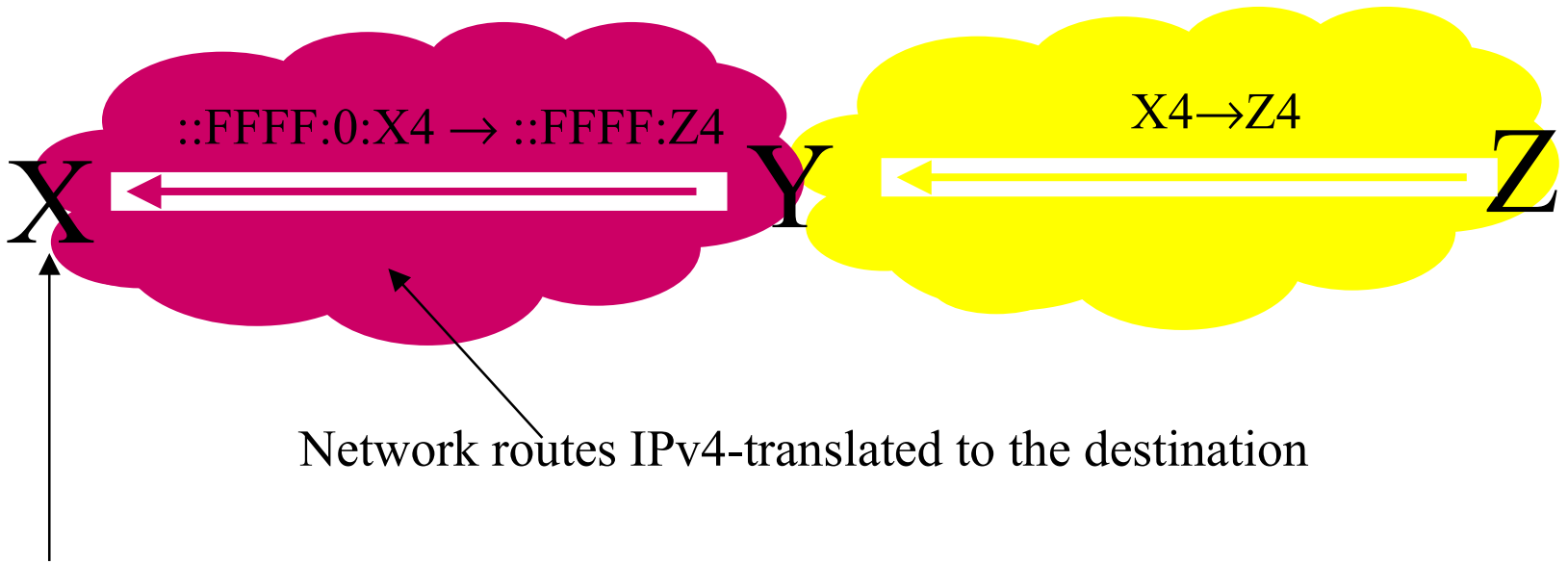



- FFFF doesn't modify TCP/UDP checksum

# SIIT


::FFFF:0:X4 → ::FFFF:Z4
X4→Z4
X
Y
Z
Network routes IPv4-mapped packets to a translation point
Have a IPv4-translated address assigned from a pool

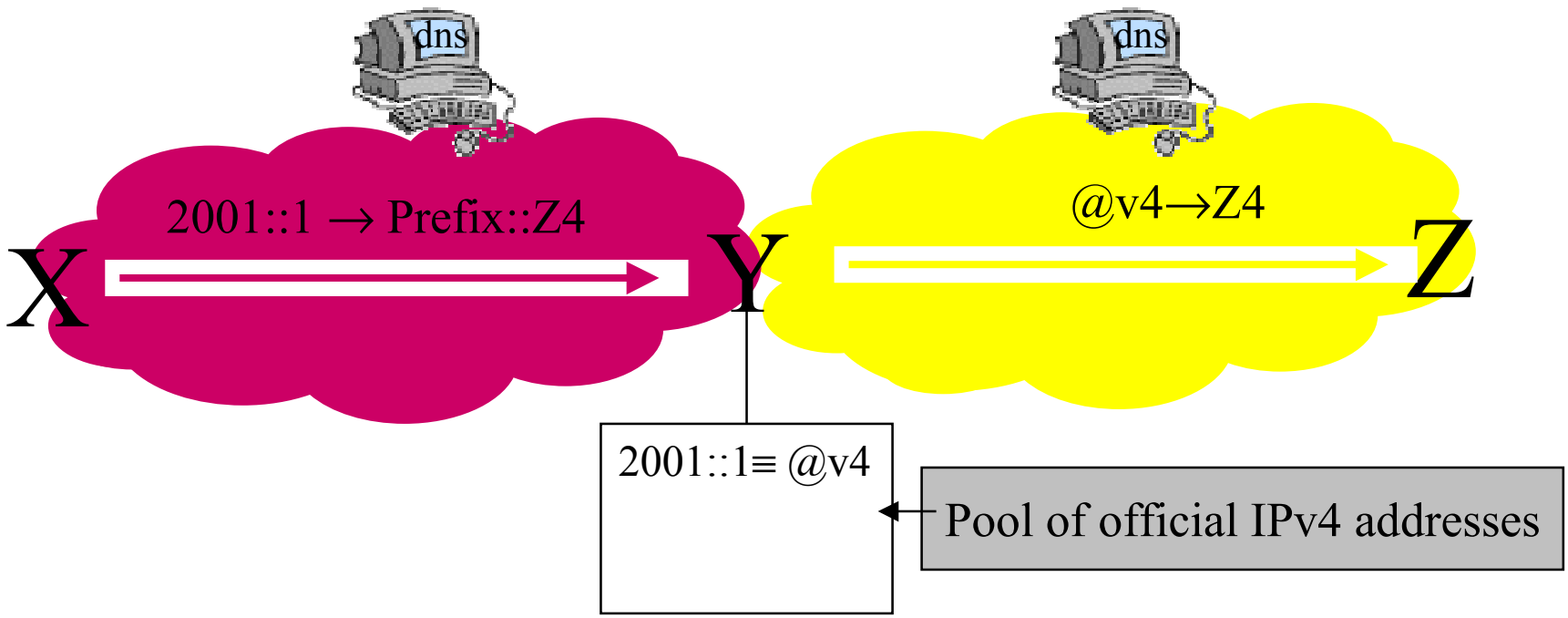
# SIIT


::FFFF:0:X4 → ::FFFF:Z4
X4→Z4
X
Y
Z
Network routes IPv4-translated to the destination
Have a IPv4-translated address assigned from a pool

# NAT-PT (RFC 2766 PS)

- Translate addresses and headers
- A pool of routable addresses is assigned to the translator
- Out coming session translation is easy
- Incoming translation must intercept DNS queries

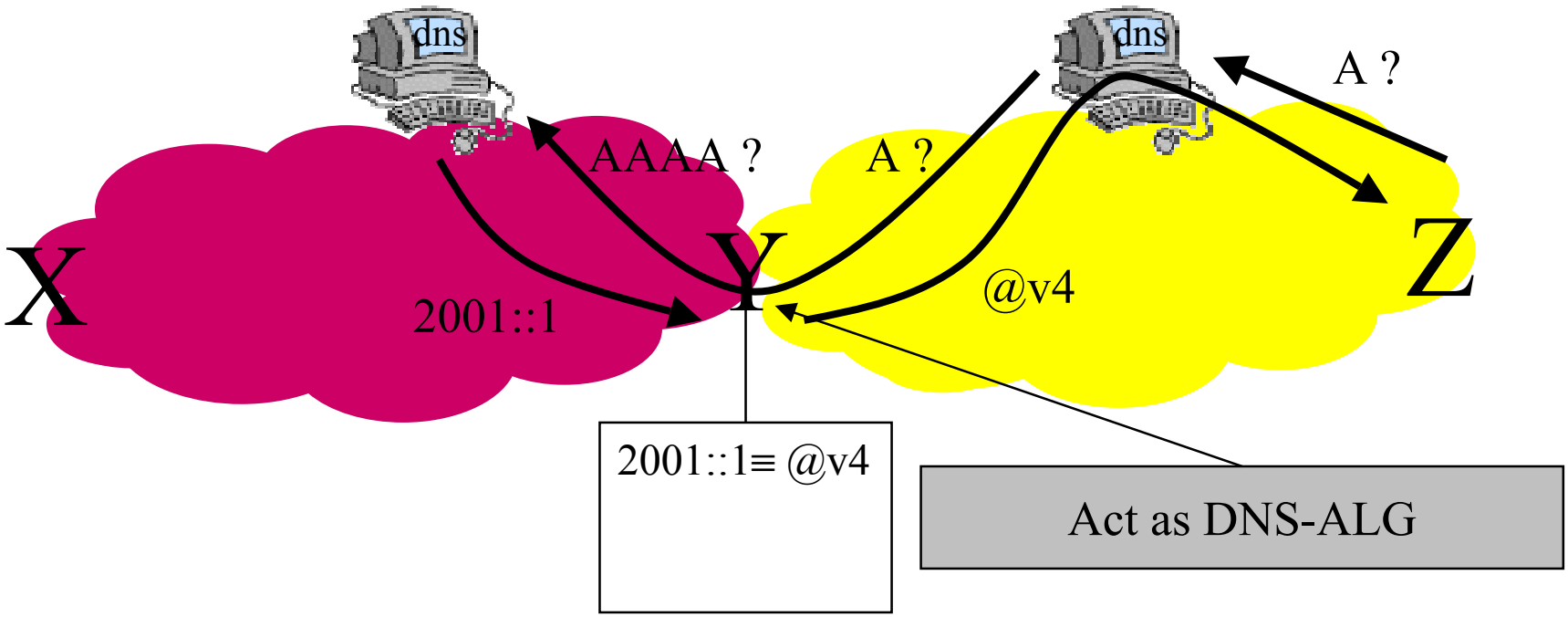
# NAT-PT: v6 to v4




`Prefix` is routed to the NAT box

May change port numbers to allow more translations

# NAT-PT: v4 to v6

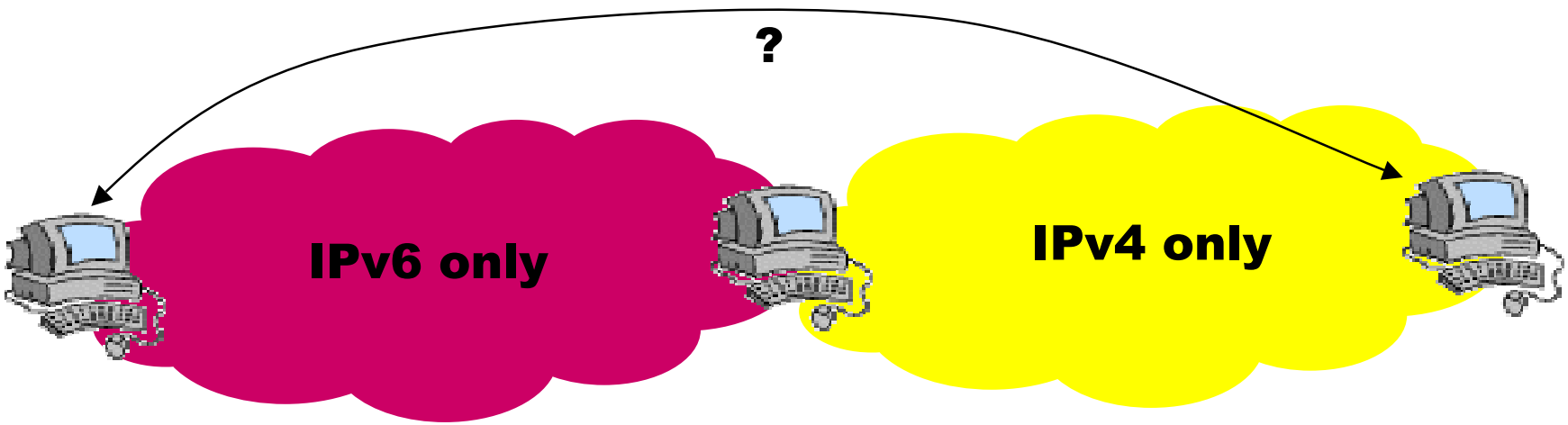


# Coexistence / Integration Mechanisms

DSTM

# Dual Stack Transition Mechanism (DSTM)

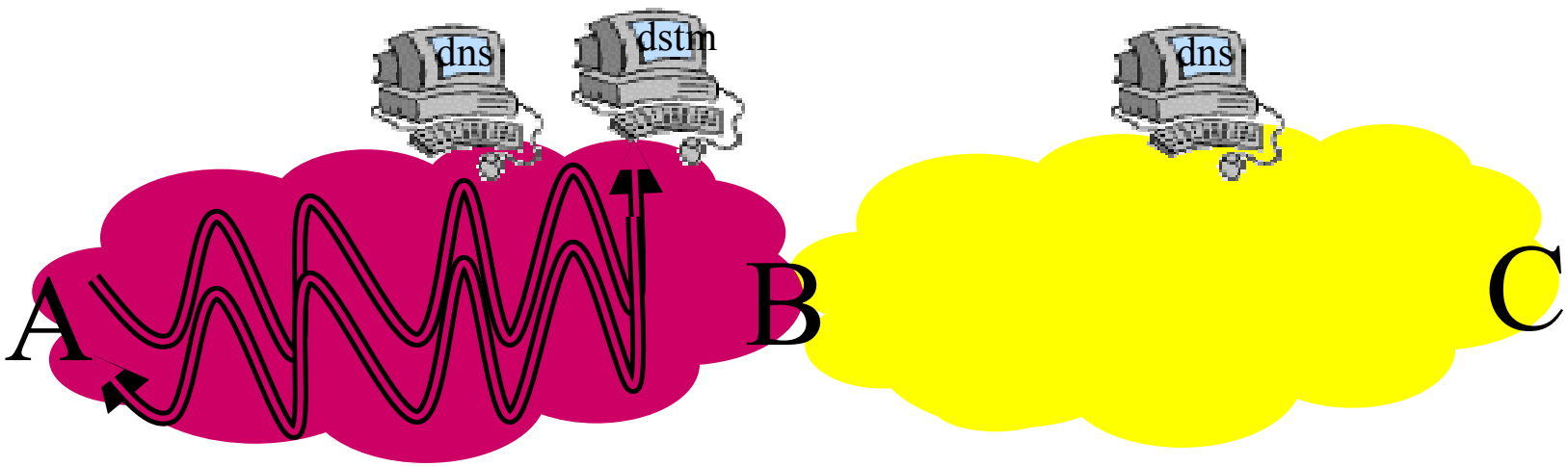
- What is it for ?
  - DSTM allows hosts in IPv6-only networks to communicate with hosts in the IPv4-only Internet.
  - DSTM allows IPv4-only applications to run (without modification) over IPv6-only networks.



# DSTM: Principles

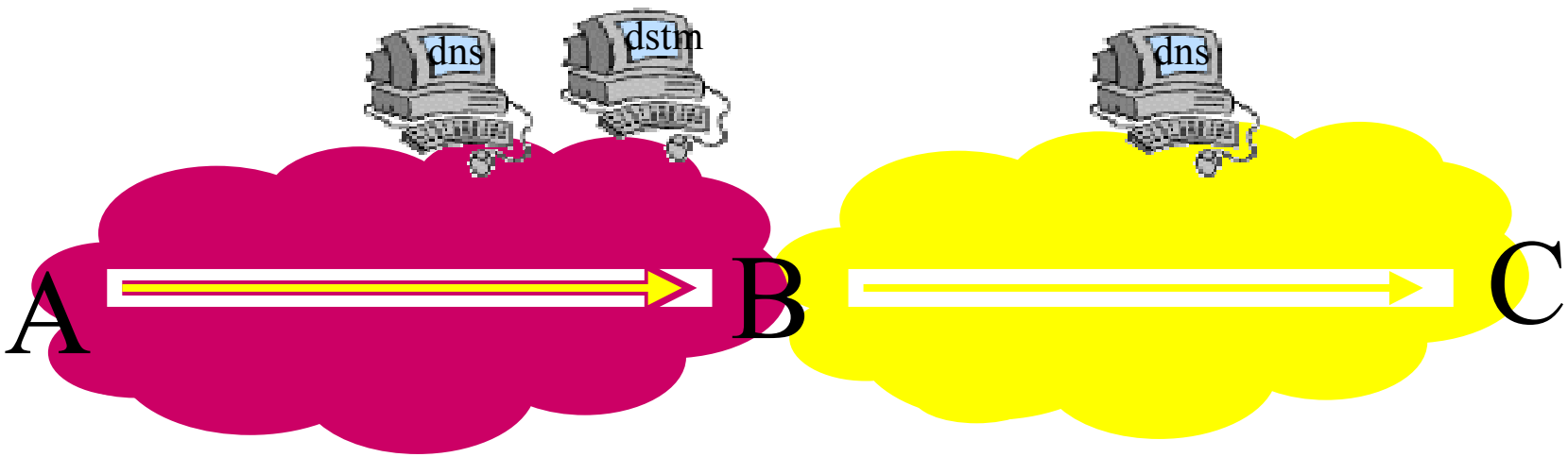
- Assumes IPv4 and IPv6 stacks are available on host.
- IPv4 stack is configured dynamically only when one or more applications need it
  - A temporary IPv4 address is assigned to the host
  - Needs an address allocation protocol.
- All IPv4 traffic coming from the host is tunneled towards the DSTM gateway (IPv4 over IPv6).
  - Needs an IPv4/IPv6 encapsulate/decapsulate gateway
  - Gateway maintains an @v6 ↔@v4 mapping table
  - Reverse route towards DSTM host MUST pass through the gateway

# DSTM: How it works (v6 → v4)



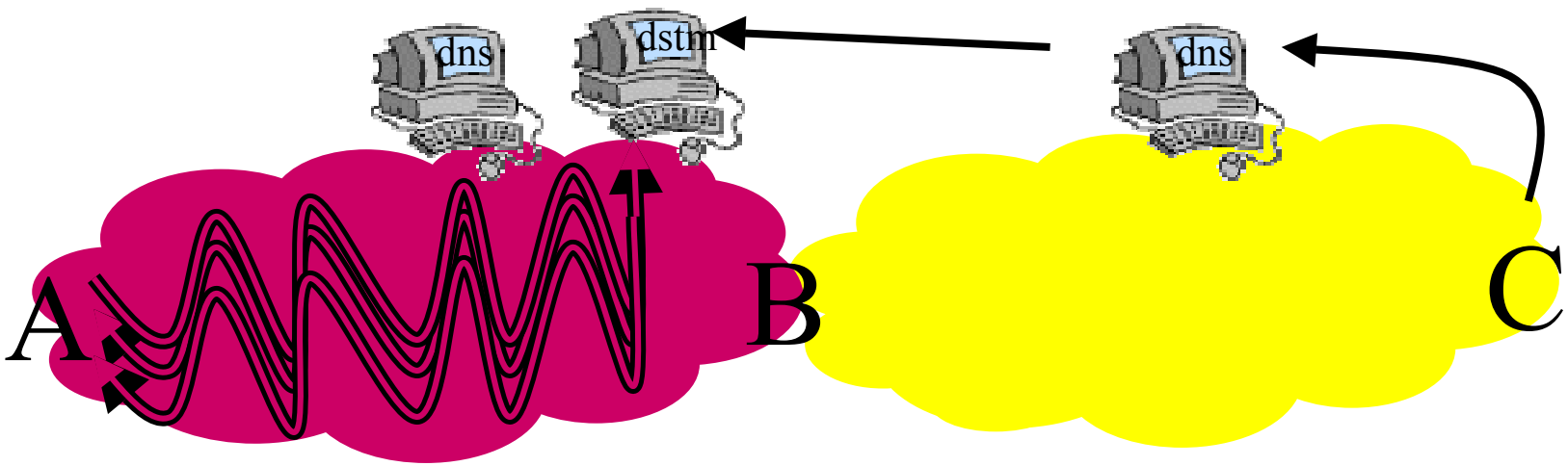

1. In A, the v4 address of C is used by the application, which sends v4 packet to the kernel
2. The interface asks DSTM Server for a v4 source address
3. DSTM server returns:
   - A temporary IPv4 address for A
   - IPv6 address of DSTM gateway

# DSTM: How it works (v6 → v4)



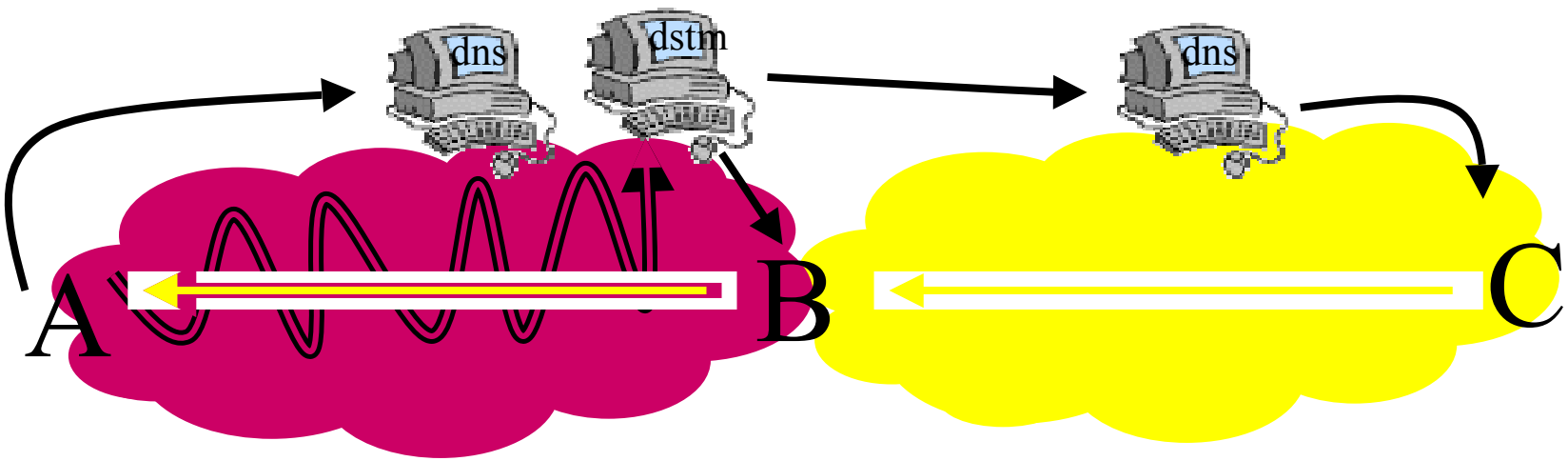

4. A creates the IPv4 packet ($A_4$ → $C_4$)
5. A tunnels the v4 packet to B using IPv6 ($A_6$ → $B_6$)
6. B decapsulates the v4 packet and sends it to $C_4$
7. B keeps the mapping between $A_4$ ↔$A_6$ in the routing table

# Scenario 2 : v4 to v6




1. C asks for the IPv4 address of « A »
2. Query fails, DSTM server tells A to configure its IPv4 stack
3. A configures its IPv4 stack

# Scenario 2 : v4 to v6




4. A registers to the DNS and tells to server
5. Mapping table at gateway is configured
6. B sends IPv4 address of A to C
7. Communication can take place

# DSTM: Address Allocation

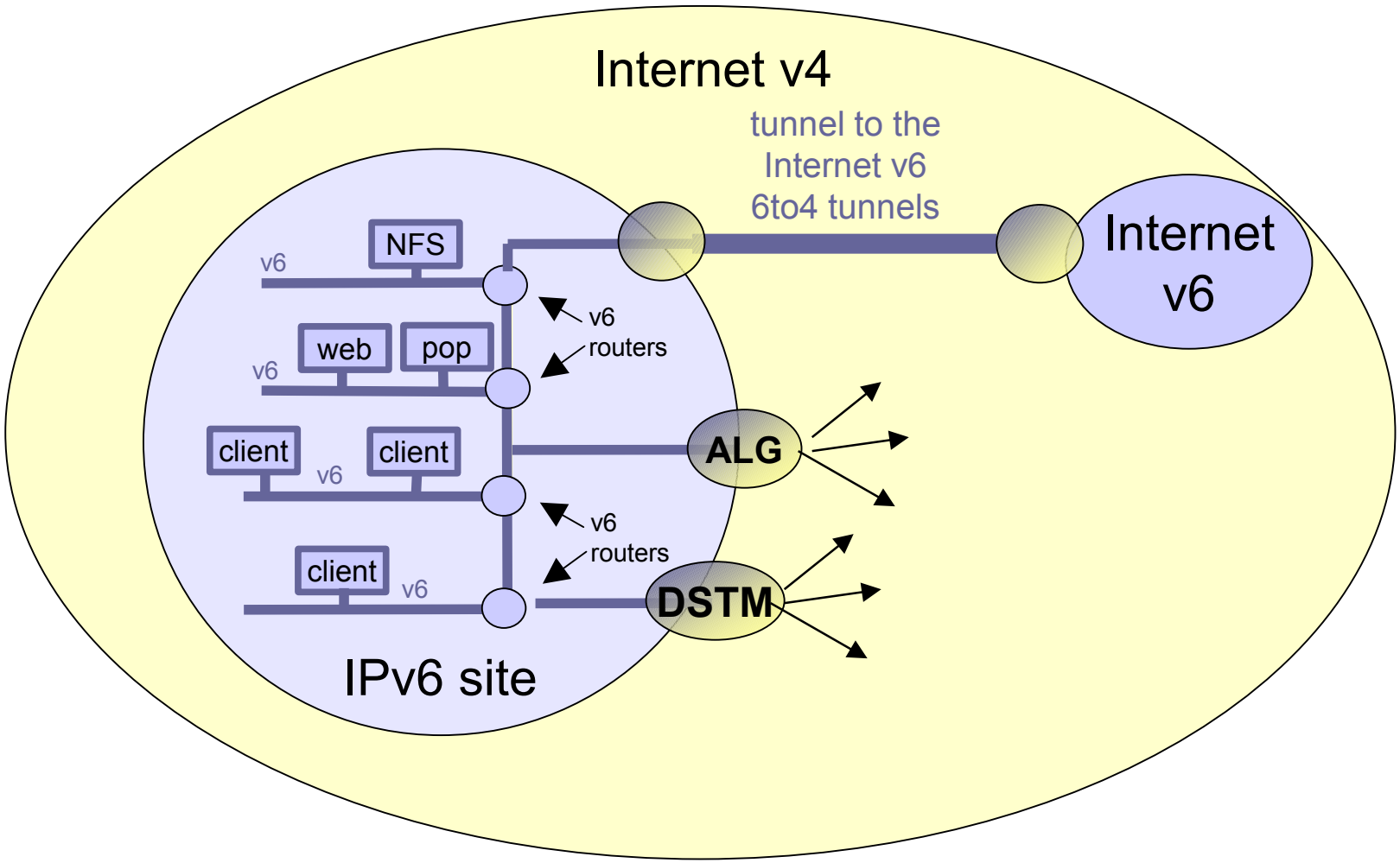
- Manual
  - host lifetime *(no DSTM server)*
- Dynamic
  - use **DHCPv6**
    - DHCPv6 may not be ready soon !
  - use **RPC**
    - Easier, RPCv6 is ready
    - Works fine in v6 ➔ v4 case.
    - Can be secure*
  - use **TSP**
    - Based on XML
    - Can be secure
- Security Concerns
  - Request for IPv4 address needs authentication
  - Automatic @6 ↔@4 mapping at gw, or configured by server?

# DSTM: Application

- DSTM is a useful tool when support for IPv4 addressing and routing is to be turned off inside a network.
  - No IPv4 addresses .. No address exhaustion problem
  - No IPv4 routing (only IPv6)… easier to manage
  - DSTM assures IPv4 communication with the external world.

- DSTM is to be used ONLY when no other means of communication is possible.
  - ALGs may be a better solution for several services
  - ALGs reduce the need of IPv4 addresses.

# DSTM: Application

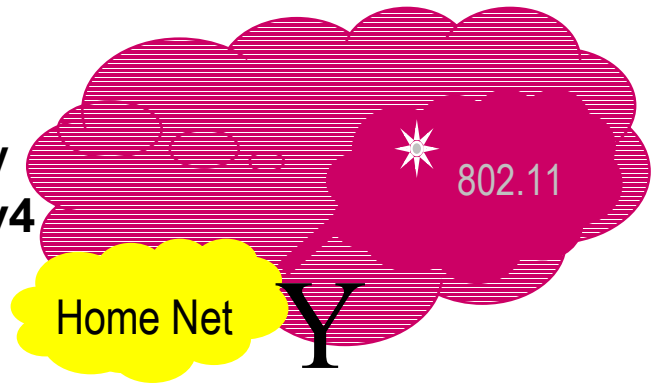


# DSTM: Deployment

- DSTM may be deployed in several phases:
  - If IPv4 address space **is not a problem**, static tunnels may be set up from DSTM nodes to the DSTM gateway. No dynamic allocation.
  - If address space **is a problem**, a dynamic address allocation mechanism may be set up (TSP, RPC, DHCPv6).
  - If address space **is a big problem**, address allocation may also involve port numbers.

# Application: The VPN scenario

- **Giving IPv4 addresses to visitors can become expensive:**
  - Visited Network offers IPv6 connectivity **only**
- **Home network offers connection to the v4 world via DSTM**
  - to Corporate Intranet
  - to Global Internet




```
ed0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        inet6 fe80::200:c0ff:fe11:cba0%ed0 prefixlen 64 scopeid 0x1
        inet6 3ffe:305:1002:4:200:c0ff:fe11:cba0 prefixlen 64
        inet6 2001:660:282:4:200:c0ff:fe11:cba0 prefixlen 64
        ether 00:00:c0:11:cb:a0

gif0: flags=8011<UP,POINTOPOINT,MULTICAST> mtu 1280
        inet6 fe80::200:c0ff:fe11:cba0%gif0 --> ::  prefixlen 64
        inet 192.108.119.197 --> 192.108.119.199 netmask 0xffffffff
        physical address inet6 3ffe:305:1002:4:200:c0ff:fe11:cba0 -
-> 3ffe:305:1002:1:200:c0ff:fe85:cba0
```

# DSTM vs. NAT-PT

- NAT-PT has the same problems as classic NAT:
  - Translation is sometimes complex (e.g. FTP)
  - NAT box may need to be configured for every new application.
  - NAT-PT supposes v6fied applications
    - This is not the case!
    - In DSTM, applications can send IPv4 packets to the kernel.

# DSTM: Implementations

- BSD « INRIA »
  - DSTM gateway
  - DSTM server (RPC)
  - Client: manual conf, dynamic conf
- BSD Kame:
  - Gateway/Server on the same host
  - Based on RPC (dynamic conf)
  - Compatible with Linux implementation

# DSTM: Implementations

- Linux :
  - Dynamic configuration using RPC
  - 4over6 interface
  - Same capabilities as BSD version
- Windows :
  - Prototype from isoft (Korea)
  - 4over6 interface for windows client
  - Uses DHCPv6
  - Server runs over Linux
  - Needs external TEP

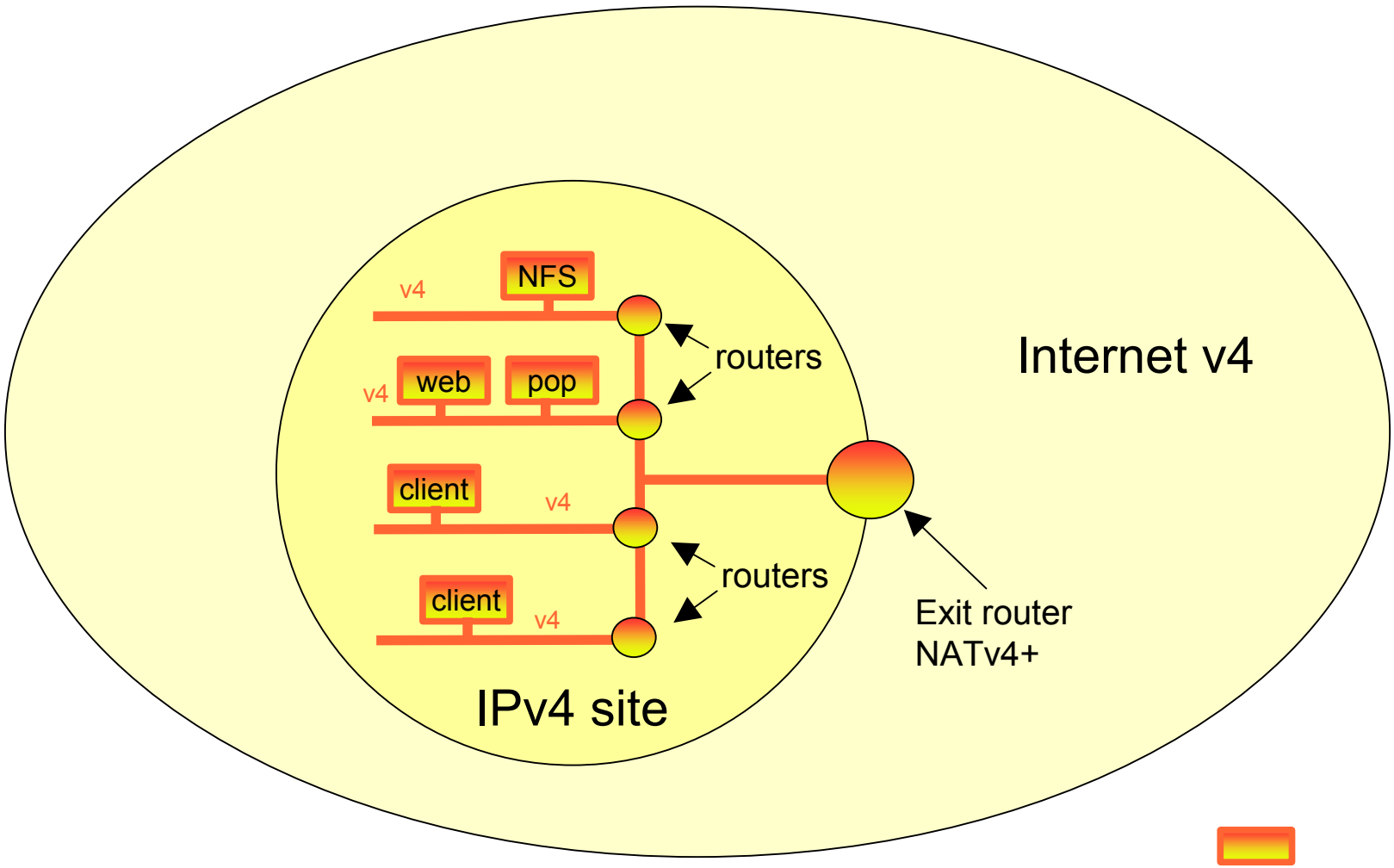
http://www.ipv6.rennes.enst-bretagne.fr/dstm/

# Deployment/migration strategies

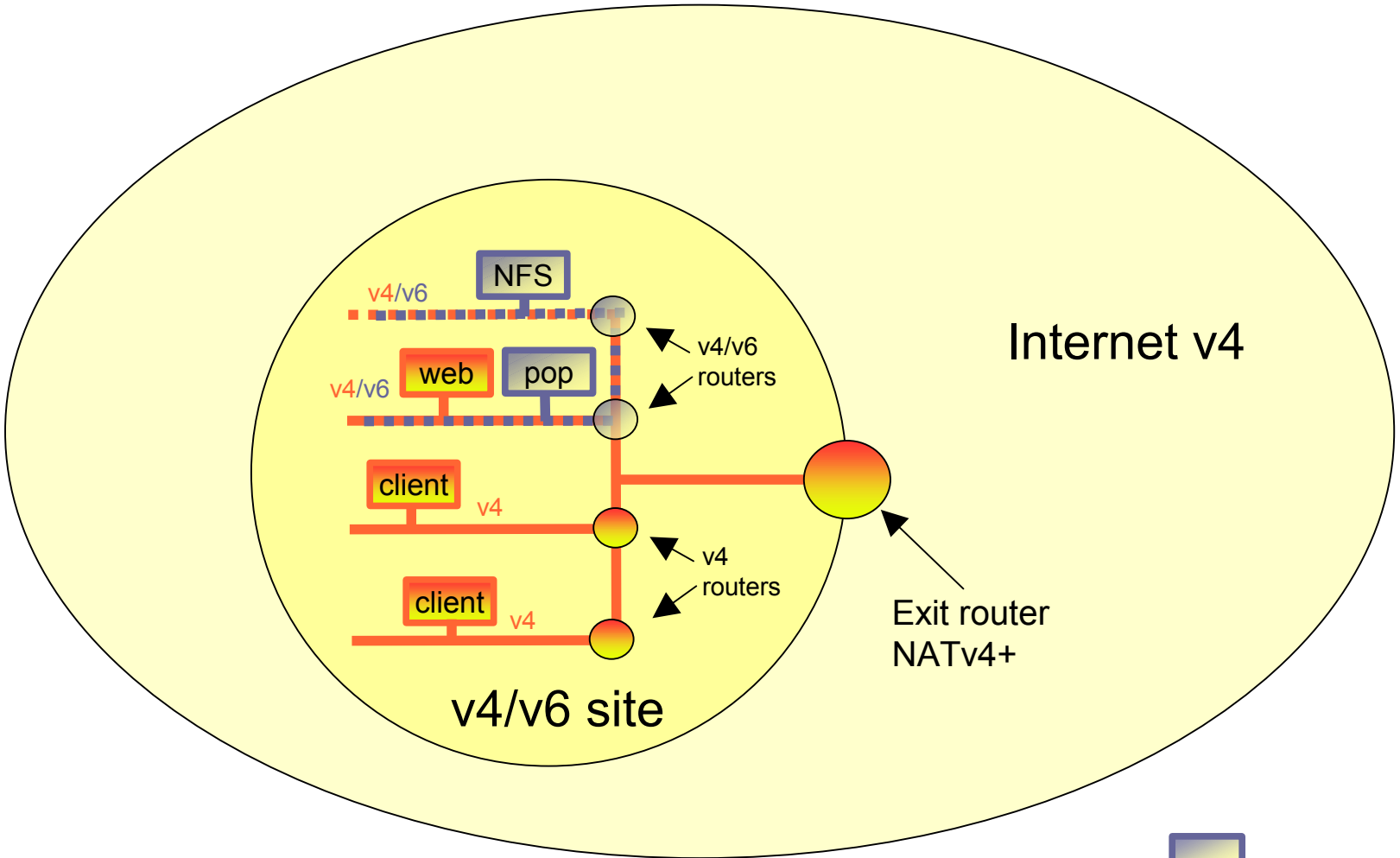
# Deployment strategies

- Technical factors
  - IPv6 availability (connectivity)
  - Native IPv6 applications/services availability
  - Avoid blocking situations (chicken and egg problem)
- Psychological factors
  - skills to configure IPv6
  - risk to modify something that works
- Deploy only one version of IP (either v4 or v6) on a given area of the network
  - To manage both routing plans

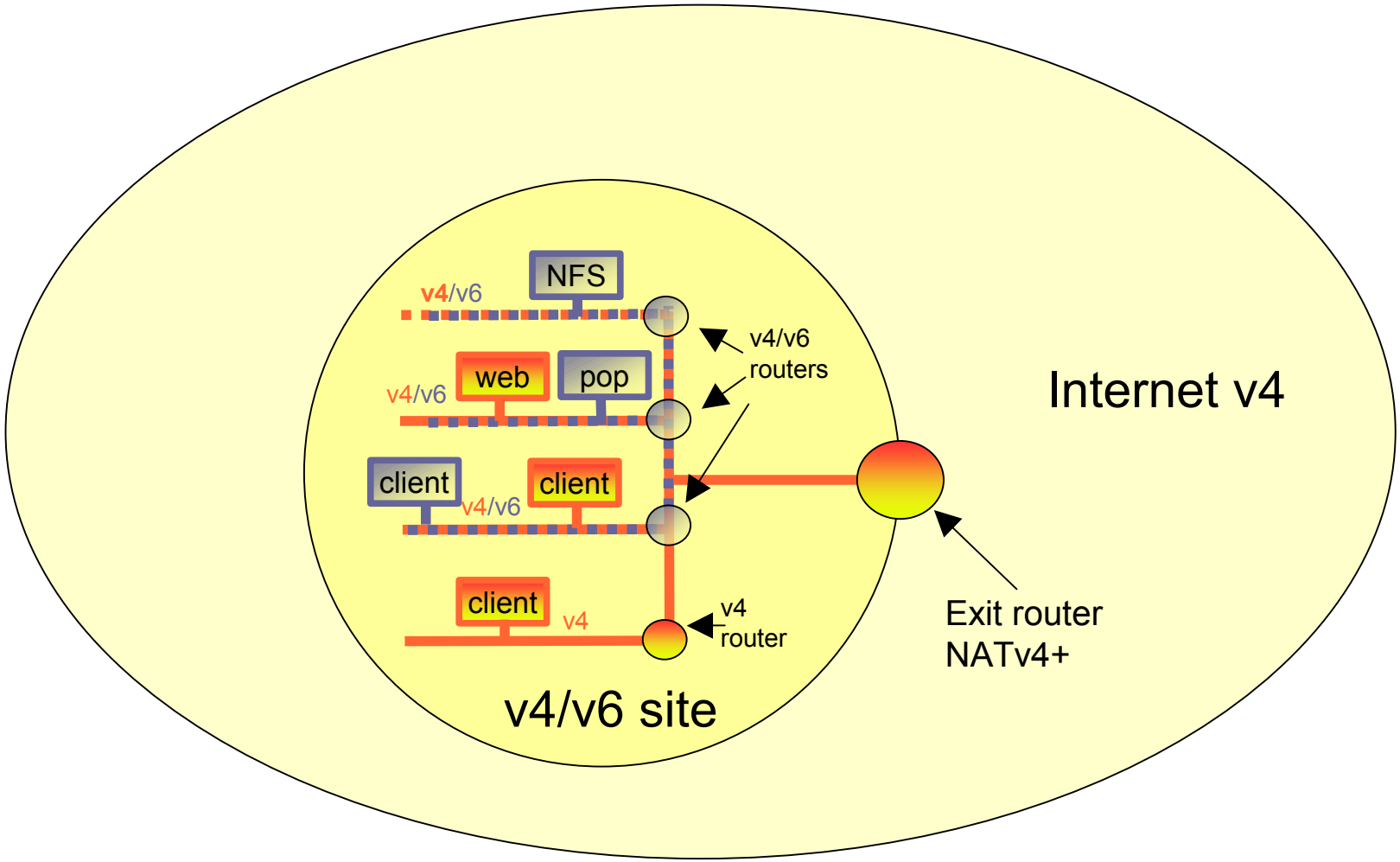
# Case study: phase 0
# IPv4 site



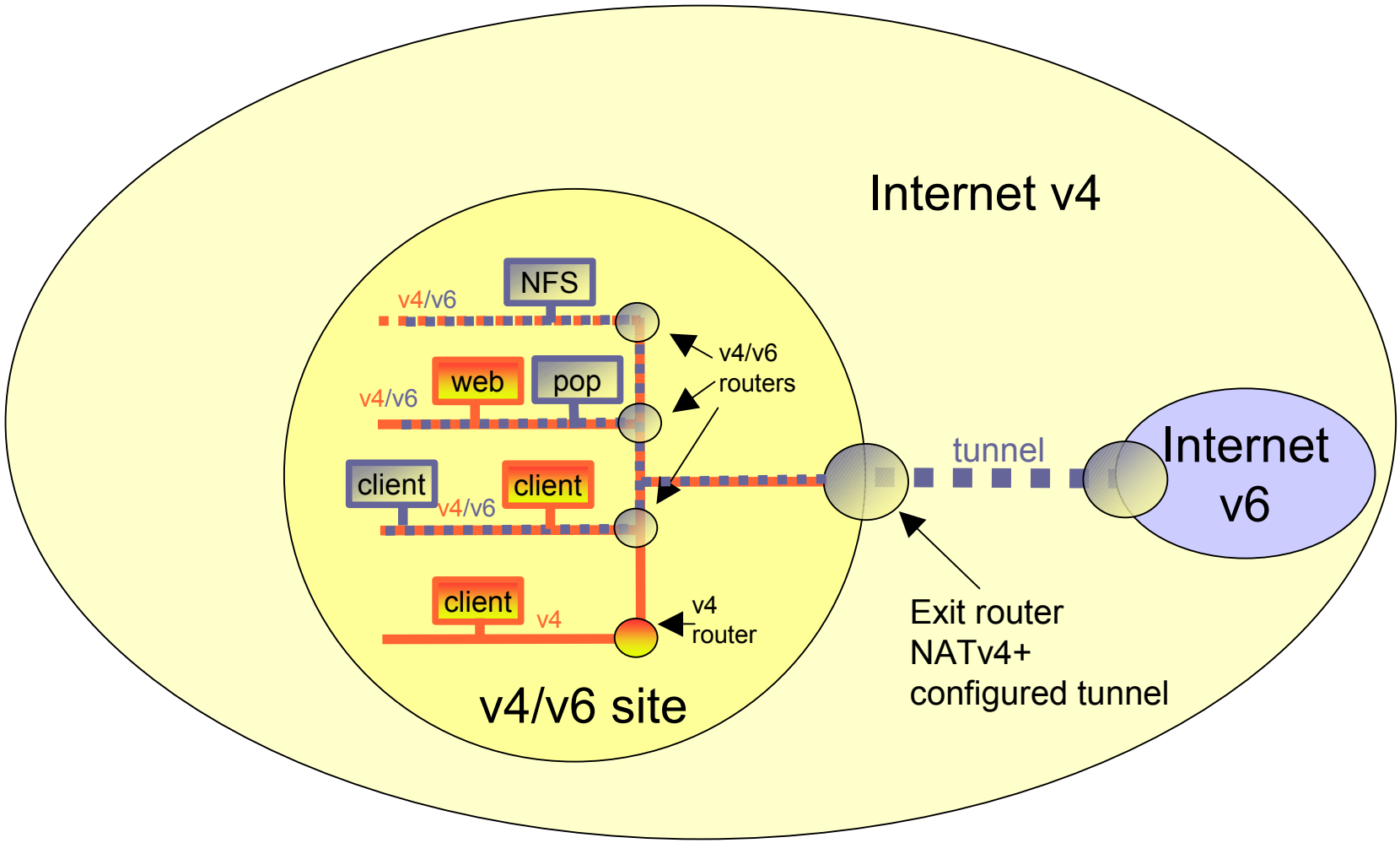
# Case study: phase 1 hybrid stack servers & routers



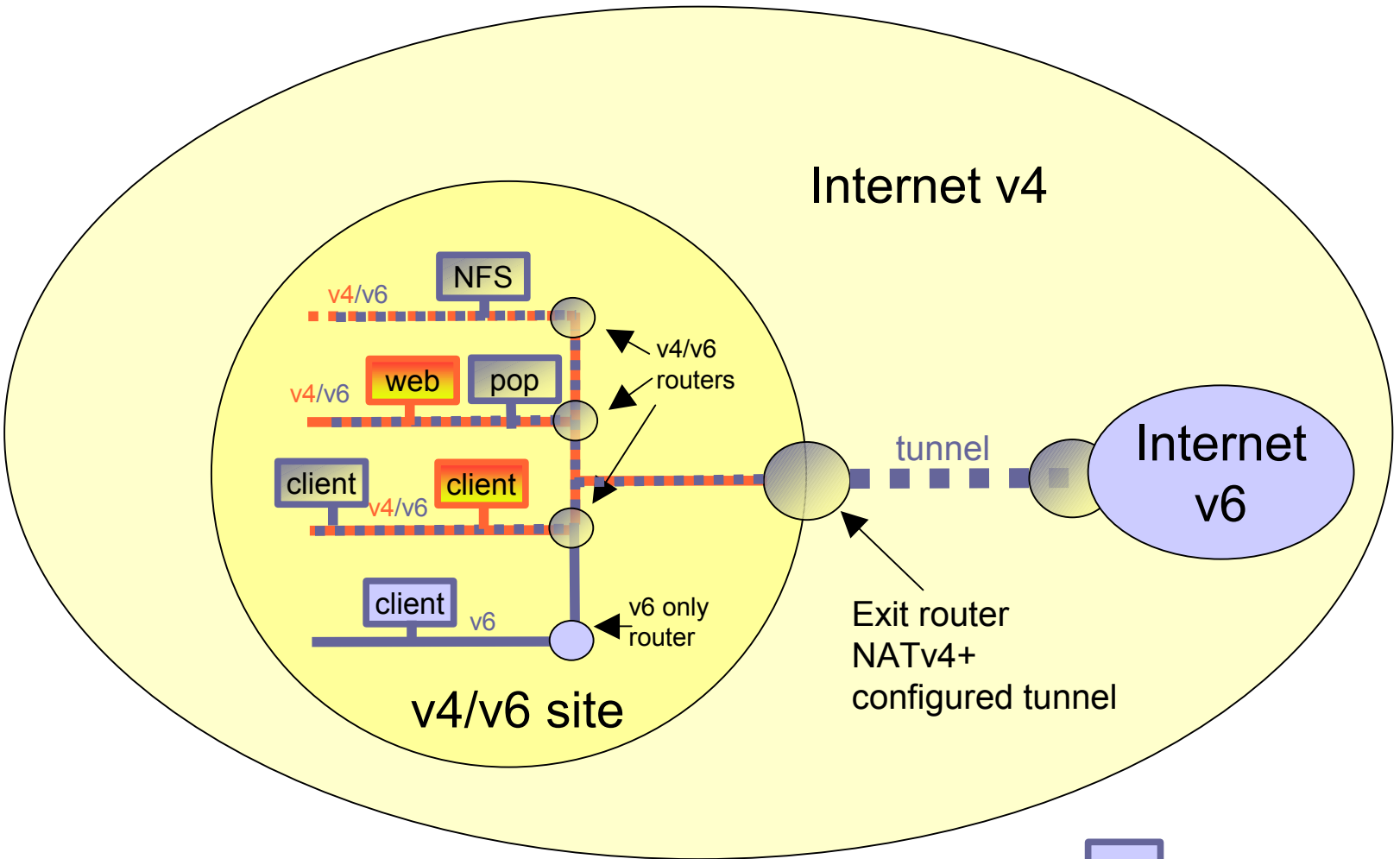
# Case study: phase 2 hybrid stack clients



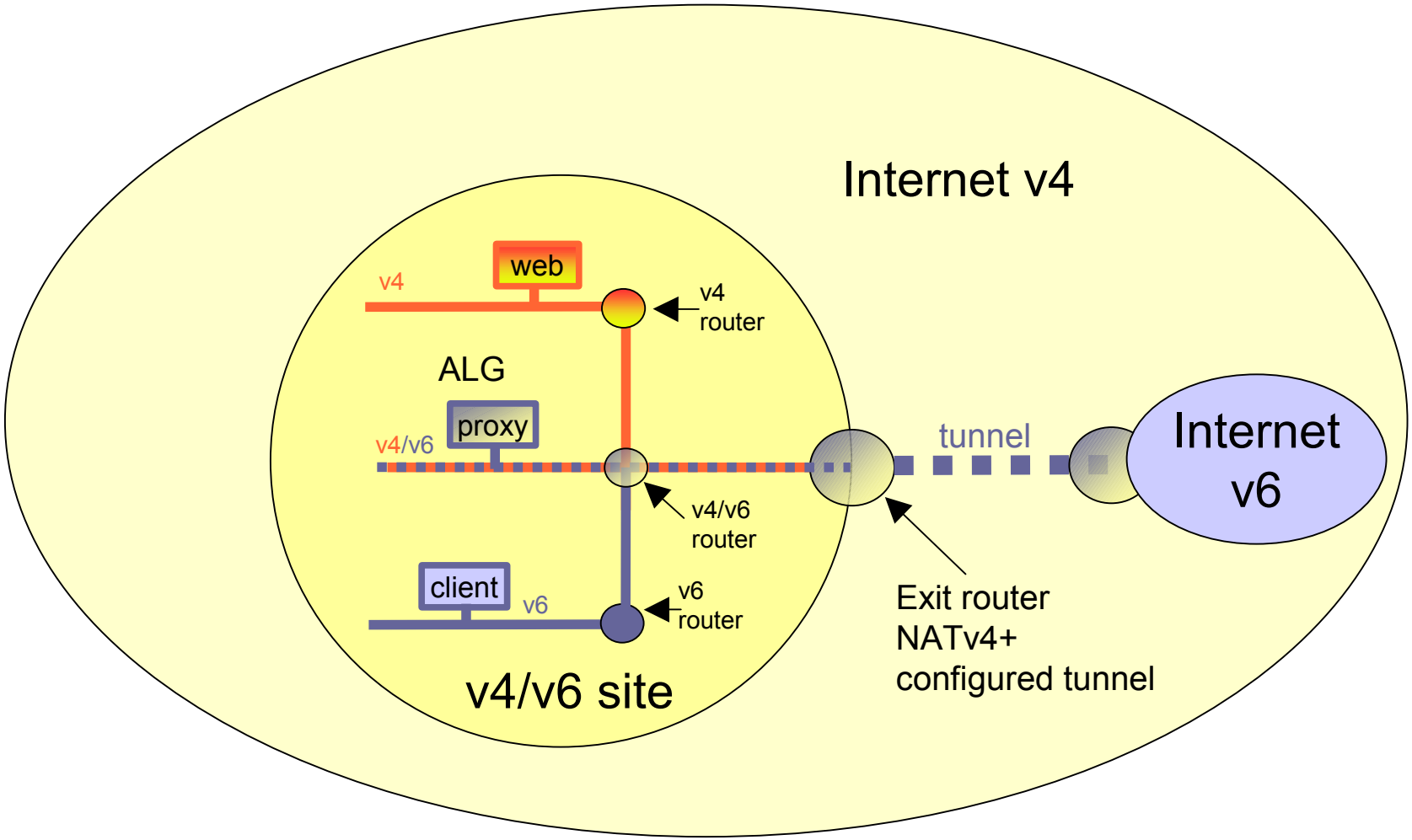
# Case study: phase 3 Connection to the Internet v6



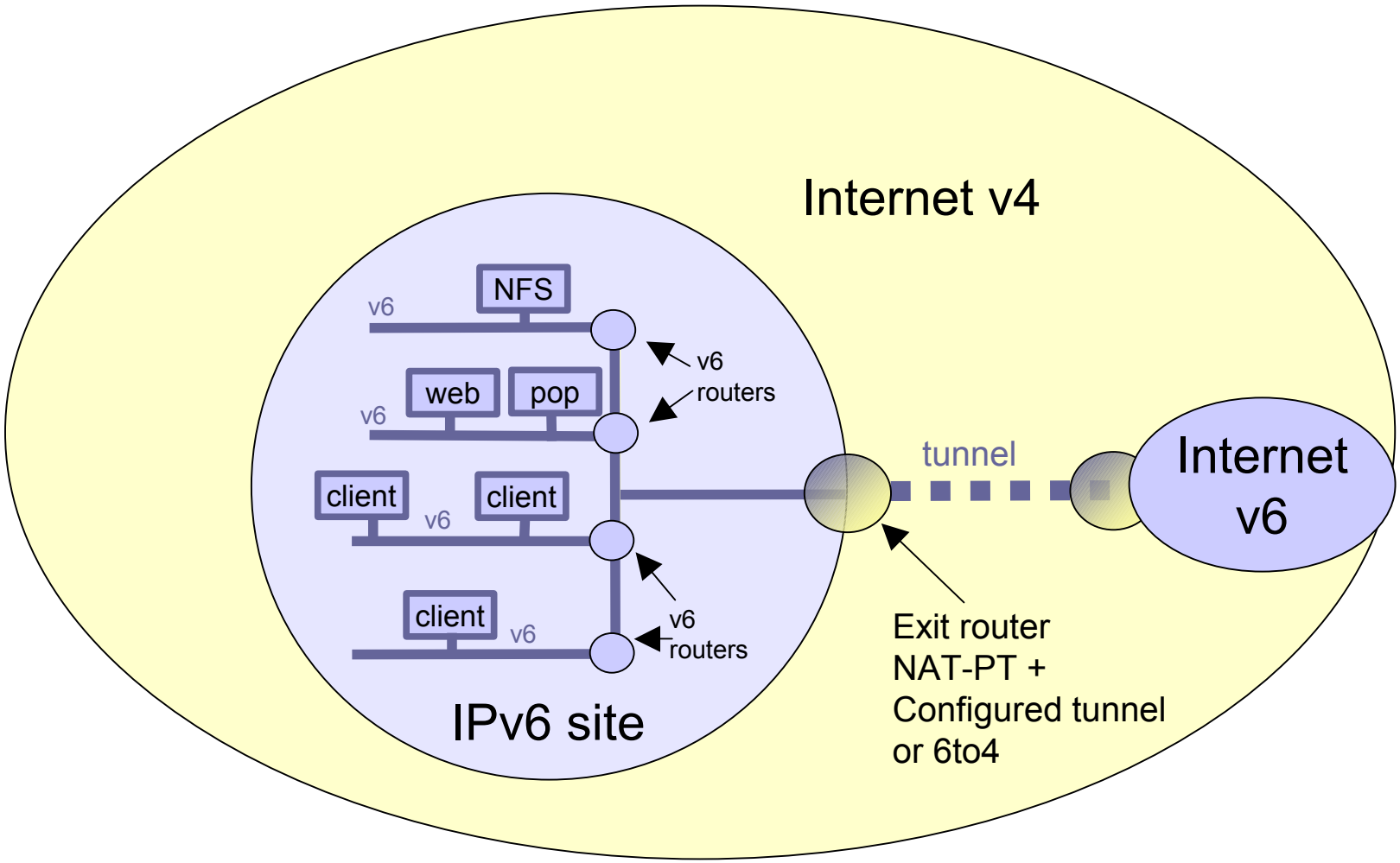
# Case study: phase 4 IPv6 only hosts



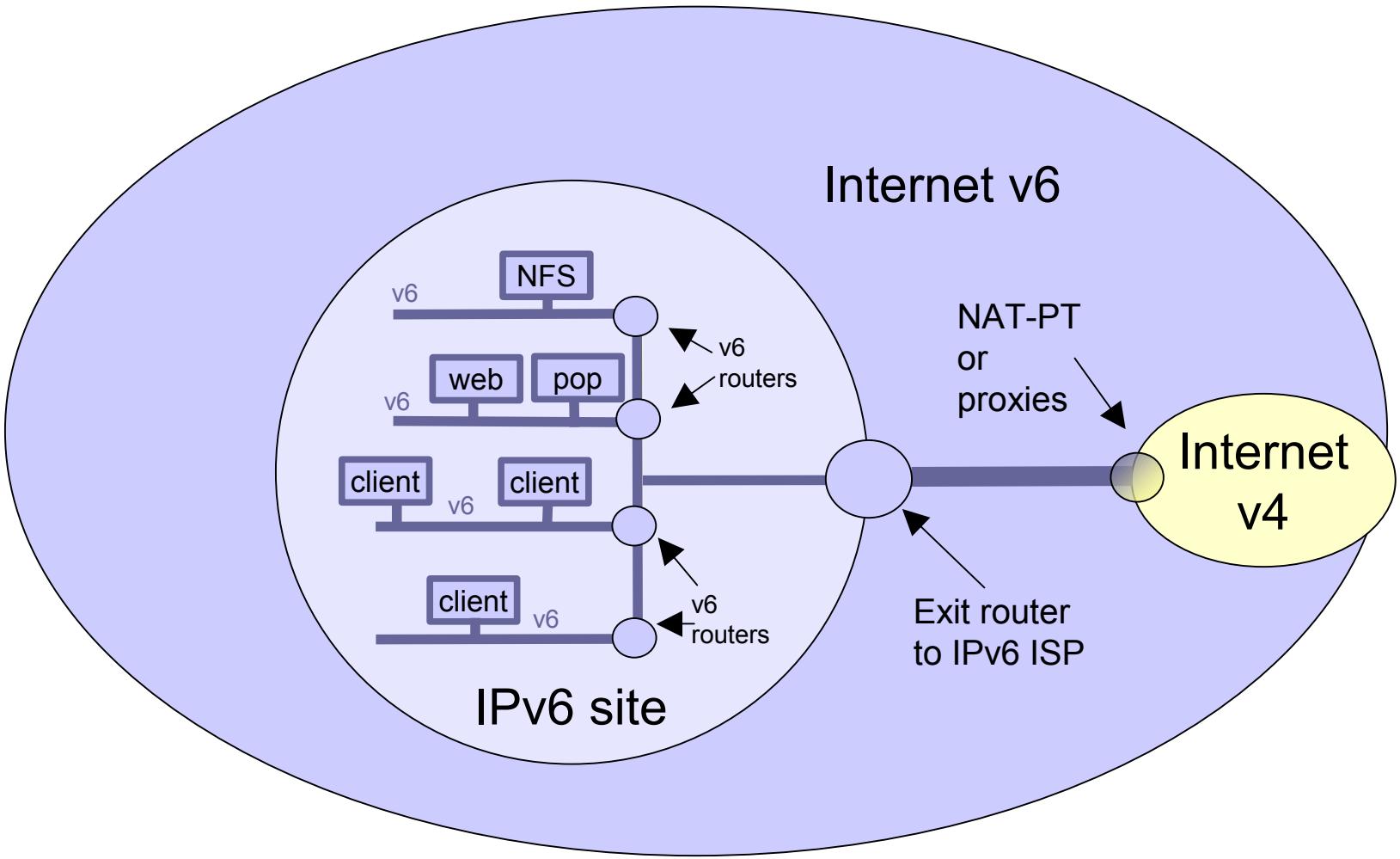
# Case study: phase 5
# IPv6 only hosts to IPv4 server



# Case study: Phase 6
# No IPv6 ISP available - IPv6 site



# Case study: phase 7
# IPv6 site / IPv6 Internet



# Equipment Configuration

# Equipment Configuration

- CISCO
- JUNIPER
- 6WIND
- FreeBSD
- Debian
- Microsoft (Windows XP)
- Zebra

# CISCO

- Enable IPv6 on an interface

```
interface xxxxx
   ipv6 enable
```

- Configure an address

```
interface xxxxx
   ipv6 address X:X:X:X::X/<0-128> (general address)
   ipv6 address X:X:X:X::X (link-local address)
   ipv6 address autoconfig (auto-configuration)
```

# CISCO (2)

- Configure an IPv6 in IPv4 tunnel

```
interface tunnel x
   tunnel source interface
   tunnel destination X.X.X.X
   ipv6 address X:X:X:X::X/<0-128>
   tunnel mode ipv6ip (for direct tunneling)
   tunnel mode gre ip (for gre encapsulation)
```

# CISCO (3)

- Configure an IPv6 in IPv6 tunnel

```
interface tunnel x
   tunnel source interface
   tunnel destination X:X:X:X::X
   ipv6 address X:X:X:X::X/<0-128>
   tunnel mode ipv6 (for direct tunneling)
   tunnel mode gre ipv6 (for gre encapsulation)
```

# CISCO (4)

- Enable IPv6 routing

```
ipv6 unicast-routing
```

- Configure static routes

```
ipv6 route prefix/prefixlen next_hop
```

*Ex: ipv6 route ::/0 2001:660:10a:1001::1*

# CISCO (5)

- BGP configuration

```
router bgp xxxx
  neighbor X:X:X:X::X remote-as …
  neighbor X:X:X:X::X …
    address-family ipv6
    neighbor X:X:X:X::X activate
    neighbor X:X:X:X::X …
  exit address-family
```

# CISCO (6)

- ACLs

```
ipv6 prefix-list bgp-in-6net seq 5 deny ::/0
```

-> Means filter ::/0 exactly

```
ipv6 prefix-list bgp-in-6net seq 10 deny 3FFE:300::/24 le 28
ipv6 prefix-list bgp-in-6net seq 15 deny 2001:660::/35 le 41
ipv6 prefix-list bgp-in-6net seq 20 permit 2002::/16
ipv6 prefix-list bgp-in-6net seq 25 permit 3FFE::/17 ge 24 le 24
ipv6 prefix-list bgp-in-6net seq 30 permit 3FFE:8000::/17 ge 28 le 28
```

-> Means every prefix matching `3FFE:8000::/17` with length 28

```
ipv6 prefix-list bgp-in-6net seq 35 permit 3FFE:4000::/18 ge 32 le 32
ipv6 prefix-list bgp-in-6net seq 40 permit 2001::/16 ge 32 le 35
```

-> Means every 2001::/16 derived prefix, with length between 32 and 35

# Juniper (1)

- Interface configuration

```
interfaces {
    name_of_interface {
        unit x {
            family inet {
                address X.X.X.X/prefixlength;
            }
            family iso {
                address Y.Y.Y.Y.Y.Y;
            }
            family inet6 {
                address Z:Z:Z:Z::Z/prefixlength;
```

- Cannot autoconfigure the router interfaces

# Juniper (2)

- Router advertisements (stateless autoconf)

```
protocols {
    router-advertisement {
        interface interface-name {
            prefix IPv6_prefix::/prefix_length;
```

- Configure tunnel (with Tunnel PIC)

```
interface{
    ip-x/x/x {
          tunnel {
                    source ipv4_source_address;
                    destination ipv4_destination_address;
                    }
          family inet6 {
                    address ipv6_address_in_tunnel/prefixlength
```

# Juniper (3)

- Static routes

```
routing-options {
    rib inet6.0 {   -> Means IPv6 unicast routing table
          static {
                  route IPv6_prefix next-hop IPv6_address;
          }

routing-options {
    rib inet6.0 {
          static {
                  route IPv6_prefix discard; -> Useful to originate a network
          }
```

# Juniper (4)

- BGP configuration

```
brotocols {
    bgp {
          local-as local_AS_number;
          group EBGP_peers {
                  type external;
                  family inet6 {
                          unicast; }
                  neighbor neighbor_IPv6_address;
                          peer-as distant_AS_number;
                          import in-PS;
                          export out-PS; }
```

# Juniper (5)

- Policy statements

```
policy-statement in-PS {
    term from_outside_accept {
          from {
                  route-filter 2002::/16 exact;
                  route-filter 3FFE::/17 prefix-length-range /24-/24;
                  route-filter 3FFE:8000::/17 prefix-length-range /28-/28;
                  route-filter 3FFE:4000::/18 prefix-length-range /32-/32;
                  route-filter 2000::/3 prefix-length-range /16-/16;
                  route-filter 2001::/16 prefix-length-range /29-/35; }
          then {
                  accept; }
    then reject; }
```



# Interface Configuration

- Enter Ethernet Private Interface Context

  ***hurricane{myconfig}*** *eth0_0*
  ***hurricane{myconfig-eth0_0}***

- Set IP Address

  ***hurricane{myconfig-eth0_0}*** *ipaddress* ***10.0.0.10/24***
  ***hurricane{myconfig-eth0_0}*** *ipaddress* ***3ffe:10::beef/48***

- Advertise an IPv6 prefix

  ***hurricane{myconfig-eth0_0}*** *prefix* ***3ffe:10::beef:f00d::/64***

# 6WIND (2)

## Migration configuration

- Enter Migration Context

  ***hurricane{myconfig}*** *mig*
  ***hurricane{myconfig-mig}***

- Create 6in4 interface

  ***hurricane{myconfig-mig}*** *6in4* ***0 1.1.1.10 1.1.1.20 3ffe:1::10 3ffe:1::20***

- Create 4in6 interface

  ***hurricane{myconfig-mig}*** *4in6* ***0 3ffe:1::10 3ffe:1::20 1.1.1.10 1.1.1.20***

- Create 6to4 interface

  ***hurricane{myconfig-mig}*** *6to4* ***1.1.1.10***

# 6WIND (3)

## Migration configuration

- Create ISATAP interface

  ***hurricane{myconfig-mig}*** *isatap_router* ***0 10.0.0.10***

  ***hurricane{myconfig-mig}*** *isatap_prefix* ***0 2002:101:10a::/64***

- Create DSTM interface

  ***hurricane{myconfig-mig}*** *dstm* ***eth0_0***

# 6WIND (4)

## Static Routing Configuration

- Enter Routing Context

  ***hurricane{myconfig}*** *rtg*
  ***hurricane{myconfig-rtg}***

- Set IP Default Route

  ***hurricane{myconfig-rtg}*** *ipv4_defaultroute* ***1.1.1.20***
  ***hurricane{myconfig-rtg}*** *ipv6_defaultroute* ***3ffe:1::20***

- Set static route

  ***hurricane{myconfig-rtg}*** *route* ***30.0.0.0/24 3.3.3.30***
  ***hurricane{myconfig-rtg}*** *route* ***3ffe:30::/48 3ffe:3::30***

# 6WIND (5)

## Dynamic Routing Configuration RIP

- Enter Dynamic Routing Context

  ***hurricane{myconfig-rtg}*** *dynamic*
  ***hurricane{myconfig-rtg-dynamic}***

- Activate RIP Routing Process

  ***hurricane{myconfig-rtg-dynamic}*** *router rip*
  ***hurricane{myconfig-rtg-dynamic-router-rip}*** *network* ***1.1.1.0/24***
  ***hurricane{myconfig-rtg-dynamic-router-rip}*** *network* ***3.3.3.0/24***
  ***hurricane{myconfig-rtg-dynamic-router-rip}*** *redistribute* ***connected***

# 6WIND (6)

## Dynamic Routing Configuration BGP4+

- Enter Dynamic Routing Context

  ***hurricane{myconfig-rtg}*** *dynamic*
  ***hurricane{myconfig-rtg-dynamic}***

- Activate BGP4+ Routing Process

  ***hurricane{myconfig-rtg-dynamic}*** *router bgp* ***10***
  ***hurricane{myconfig-rtg-dynamic-router-bgp}*** *neighbor* ***3ffe:1::20 remote-as 20***
  ***hurricane{myconfig-rtg-dynamic-router-bgp}*** *neighbor* ***3ffe:3::30 remote-as 30***
  ***hurricane{myconfig-rtg-dynamic-router-bgp}*** *address-family ipv6*
  ***hurricane{myconfig-rtg-dynamic-router-bgp-v6}*** *neighbor* ***3ffe:1::20 activate***
  ***hurricane{myconfig-rtg-dynamic-router-bgp-v6}*** *neighbor* ***3ffe:3::30 activate***
  ***hurricane{myconfig-rtg-dynamic-router-bgp-v6}*** *redistribute* ***connected***

# FreeBSD

- Enable IPv6

  `ipv6_enable="YES"` in rc.conf file
- Autoconfiguration is automatically done while the gateway function is off
- Enable IPv6 forwarding

  `ipv6_gateway_enable="YES"` in rc.conf file
- Add an IPv6 address on an interface

  `ifconfig` *`interface`* `inet6 X:X:X:X::X prefixlen 64`

# FreeBSD (2)

- Configure an IPv6 in IPv4 tunnel

```
ifconfig gif1 create
ifconfig gif1 inet6 @IPv6_source @IPv6_dest prefixlen 128
gifconfig gif1 inet @IPv4_source @IPv4_dest
ifconfig gif1 up
```

- Configure an IPv6 in IPv6 tunnel

```
ifconfig gif1 create
ifconfig gif1 inet6 @IPv6_source @IPv6_dest prefixlen 128
gifconfig gif1 inet6 @IPv6_source @IPv6_dest
ifconfig gif1 up
```

# FreeBSD (3)

- Configure a static route
  - Default route

```
route add -inet6 default fe80::X:X:X:X%interface
route add -inet6 default X:X:X:X::X (if global address)
```

  - Others

```
route add –inet6 X:X:X:X:: -prefixlen YY X:X:X:X::X
route add –inet6 X:X:X:X:: -prefixlen YY fe80::X:X:X:X%interface
```

- %*interface* notation

  If link-local address, need to specify on which interface the address is available

# FreeBSD (4)

- RIPng: route6d daemon

  `route6d`

  `-L` *`IPv6_prefix,interface`* (receives only prefixes derived from *`IPv6_prefix`* on interface *`interface`*)

# FreeBSD (5)

- BGB: bgpd daemon
- Better to use Zebra BGP daemon

# Debian

- Main URL: http://people.debian.org/~csmall/ipv6/

- Enable IPv6
  - Put "ipv6" in "/etc/modules"
  - Edit "/etc/network/interfaces":

```
iface eth0 inet6 static
    address 2001:XXXX:YYYY:ZZZZ::1
    netmask 64
```

# Debian (2)

- Tunnel configuration
  - Edit "/etc/network/interfaces":

```
iface tun0 inet6 v4tunnel
    endpoint A.B.C.D
    address 2001:XXXX:1:YYYY::2
    gateway 2001:XXXX:1:YYYY::1
    netmask 64
```

# Debian (3)

- RA configuration on a Debian router
  - Add in "/etc/radvd.conf" :

```
interface eth0
{
   AdvSendAdvert on;
   AdvLinkMTU 1472;
   prefix 2001:XXXX:YYYY:ZZZZ:/64
   {
      AdvOnLink on;
      AdvPreferredLifetime 3600;
      AdvValidLifetime 7200;
   };
};
```

# Microsoft (Windows XP)

- Enable IPv6

  `ipv6 install` in a dos window
- Auto-configuration is then performed
- Display IPv6 interfaces

  `ipv6 if`
- Display IPv6 routes

  `ipv6 rt`

# Microsoft (Windows XP) (2)

- Add a static route

```
ipv6 rtu prefix ifindex[/address] [life valid[/pref]]
[preference P] [publish] [age] [spl Site Prefix Size]
```

- Anonymous addresses

```
ipv6 gpu UseAnonymousAddresses no
```

- « User-friendly » IPv6 configuration

`netsh` in a dos window

```
> interface ipv6
```

# Zebra

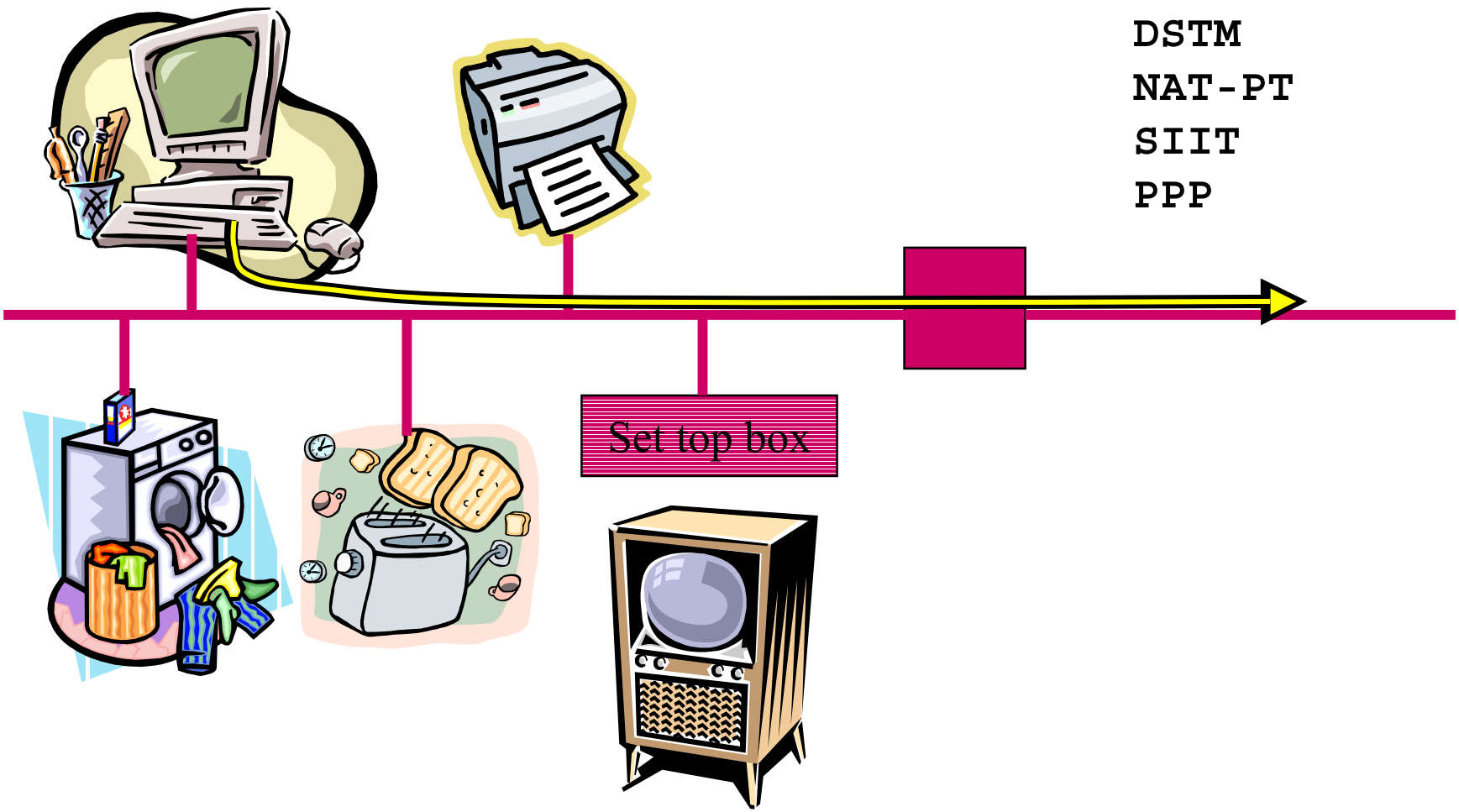
- Cisco like commands
- BGP, RIPng, OSPF available

# And once deployed…

# Home usage

- Easy configuration
  - Plug and play
  - Compatible with IEEE 1394
- Some network games send IPv4 addresses:
  - NAT doesn't work
- Advanced users wish to create servers
  - Paging, Web servers, IP telephony,…
  - Remote control

# Home usage



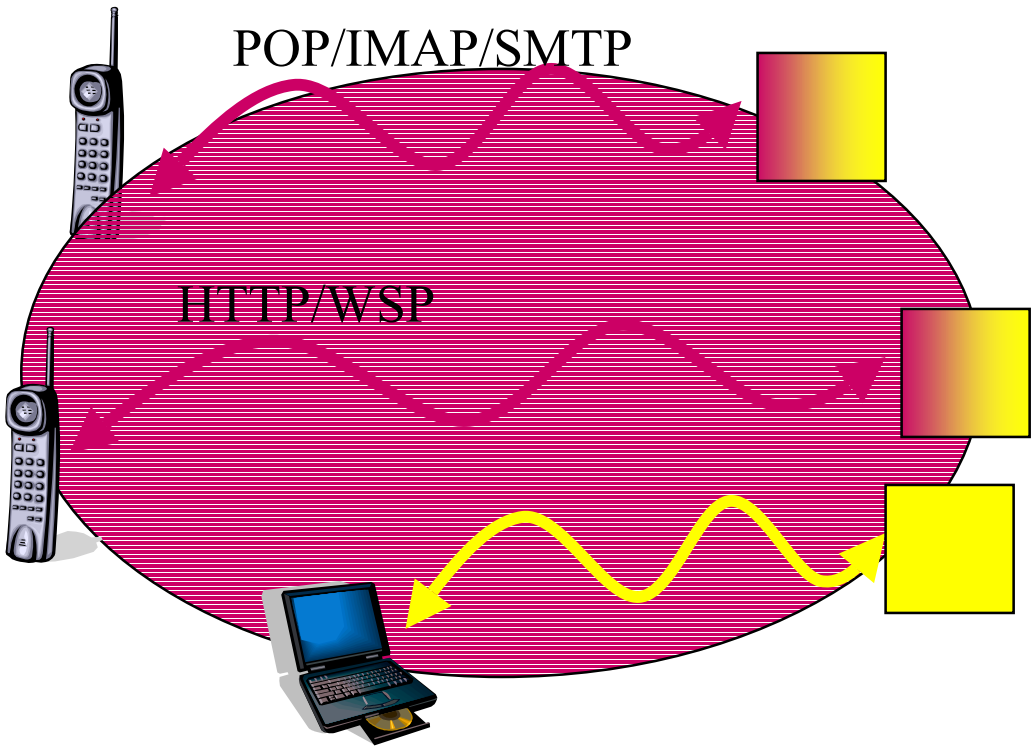
# Mobile Telephony

- Not IP telephony
- Huge number of addresses
- Can use mobile IP
  - Interaction between L2 and L3 mobility not discussed here
- End-to-End connectivity: necessary condition, fulfilled by IPv6 global addressing
- Need for services regardless of IP version...
- Robust Header Compression
  - Include RTP/UDP/IPv6
  - IPv6 header is easier to compress

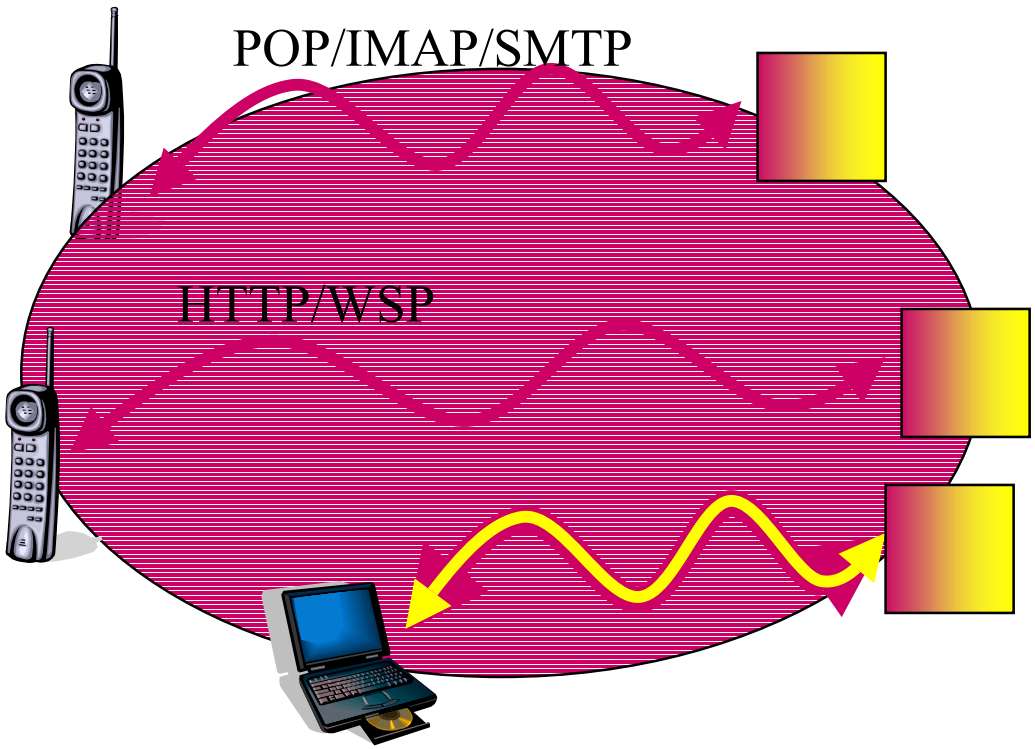
# Mobile telephony

- Some Terminal :
  - Dual stack
- Limited number of applications
  - E-mail
  - Web/WAP browser
  - …
- Can be implemented for both stacks
- Mobile PC can also be connected

# Mobile Telephony



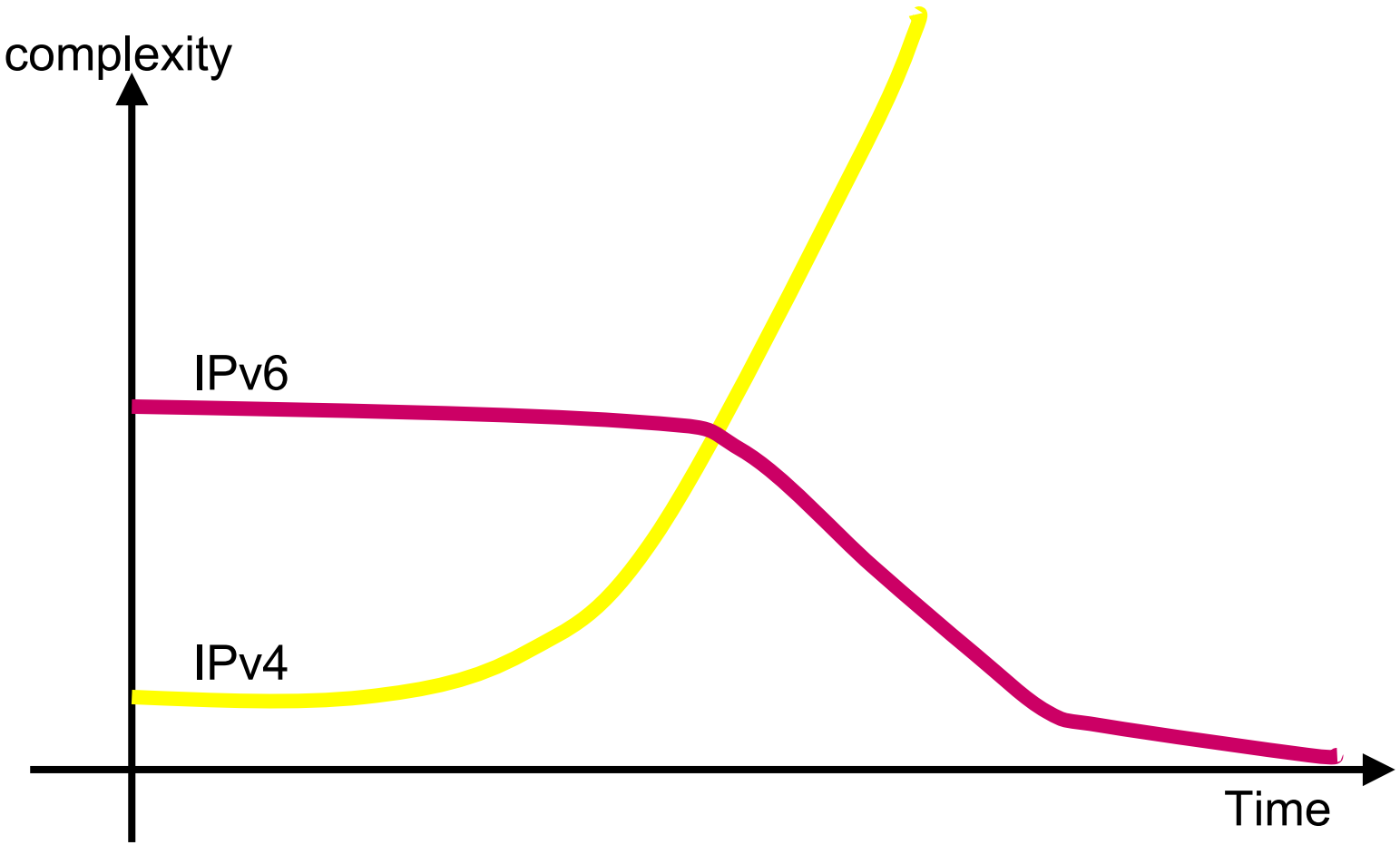
# Mobile telephony



# Conclusion

- Complexity will increase in the IPv4 world
  - New applications
  - New paradigms
  - End of end-to-end (NAT)
- Toward a layer-7 network
  - More costs
  - Difficulty to introduce new applications
- Lack of address will become a reality
  - Later or sooner
  - Depends upon the network transparency
- Get rid of one of the addressing/routing plans
  - IPv4 or IPv6 ?
  - Avoid twice the amount of work for the same service …

# Conclusion (2)



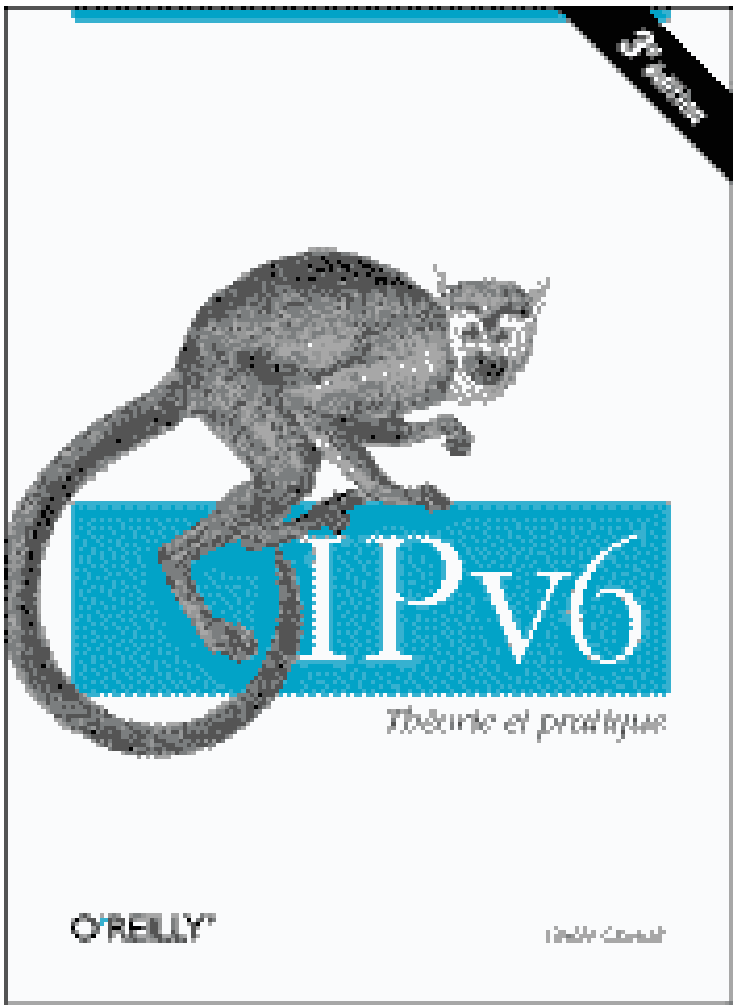
# Conclusion (3)

- IPv6 deployment might be trigged by:
  - Research projects (6bone, Renater 2 pilot, ...)
  - Developing countries (lack of IPv4 address blocks)
  - IPv6 Product availability
- Smooth integration area by area:
  - interoperability between v4 and v6 areas must be maintained for some applications and equipment
  - different approaches to maintain interoperability
  - complexity should be decreasing with time

# To go on …

- http://playground.sun.com/pub/ipng/html/ipng-main.html
  - RFCs, IDs, implementations, …
- http://www.ipv6.org/
- http://www.6bone.net/
- http://www.ripe.net/
  - IPv6 wg
- http://www.ipv6forum.com/
- http://www.g6.asso.fr/

# Bibliography … in French !



?

or

# IPv6 multicast

# Adressage multicast

- **Format d'une adresse de groupe multicast (RFC 2373)**

| 8 bits | | 4 bits | 4 bits | 112 bits |
|---|---|---|---|---|
| 1111 | 1111 | flags | scope | group ID |
| F | F | | | |

- 8 premiers bits positionnés à 1 → Adresses dérivées du préfixe FF00::/8
- Champ **flag** (4 bits) :

  `0RPT` avec

  T = 0 si adresse permanente (Définies par l'IANA)

  T = 1 si adresse temporaire

  Bits P et R détaillés ensuite
- Champ **scope** → Permet de **limiter la portée de la diffusion** sur un réseau

  0 - Reservé
  1 - Portée nœud local
  2 - Portée lien local
  3 - Portée sous-réseau local
  4 - Portée Admin-local
  5 - Portée site-local
  8 - Portée organisation-local
  E - Portée globale

# Adressage multicast

- Exemples
  - Group ID 101 → serveurs NTP
  - FF01:0:0:0:0:0:0:101 : tous les serveurs NTP sur le même nœud que l'émetteur
  - FF02:0:0:0:0:0:0:101 : tous les serveurs NTP sur le même lien que l'émetteur
  - FF05:0:0:0:0:0:0:101 : tous les serveurs NTP sur le même site que l'émetteur
  - FF0E:0:0:0:0:0:0:101 : tous les serveurs NTP sur tout l'Internet

# Adresses multicast réservées : exemples (RFC 2375)

- Adresses valables pour des portées prédéfinies
  - FF02:0:0:0:0:0:0:1 : Tous les nœuds du lien
  - FF02:0:0:0:0:0:0:2 : Tous les routeurs du lien
  - FF05:0:0:0:0:0:0:2 : Tous les routeurs sur le site
  - FF02:0:0:0:0:0:0:D : Tous les routeurs PIM du lien
  - …
- Adresses valables pour toutes les portées
  - FF0X:0:0:0:0:0:0:101 : Network Time Protocol (NTP)
  - FF0X:0:0:0:0:0:0:109 : MTP Multicast Transport Protocol
  - …

# Adresses multicast sollicitées

- Construite à partir de l'adresse unicast
- Concaténation de
  - FF02::1:FF00:0/104
  - 24 derniers bits de l'adresse unicast
- Chaque équipement construit une adresse multicast sollicitée
- Les équipement qui connaissent l'adresse v6 d'un équipement mais ne connaissent pas l'adresse MAC peuvent utiliser l'adresse multicast sollicitée pour le joindre
  - Protocole de détection d'adresses dupliquées
  - Découverte des voisins sur le lien-local (NDP)
- Evite l'utilisation de l'adresse MAC de diffusion générale (FF-FF-FF-FF-FF-FF)
- Exemple:

```
2001:0660:010a:4002:4421:21FF:FE24:87c1
FF02:0000:0000:0000:0000:0001:FF00:0000/104
FF02:0000:0000:0000:0000:0001:FF24:87c1
33-33-FF-24-87-C1
```

# Allocation des adresses de groupes

- Manuelle : choix manuel de l'adresse multicast et du port
- Dynamique
  - Session Announcement Protocol, (SAP), ID
    - SDR implante ces fonctionnalités (pas possible pour une portée globale)
  - MADCAP, RFC 2730
    - Multicast Address Dynamic Client Allocation Protocol (trop compliqué)
  - GLOP, RFC 2770
    - Intérêt avec RFC 3306 ?
- Dérivation des adresses multicast à partir des adresses unicast (RFC 3306)
  - Simplification de l'allocation des adresses
  - DHCPv6 ?

# Allocation des adresses de groupes

- **Dérivation des adresses multicast à partir des adresses unicast (RFC 3306) Flag : 0RPT**

| 11111111 | flag | scp | reserved | plen | Network prefix | Group ID |
|---|---|---|---|---|---|---|
| 8 bits | 4 | 4 | 8 bits | 8 | 64 bits | 32 bits |

  - Flag : 0RPT
    - P=0 → adresse non basée sur le préfixe unicast
    - P=1 → Adresse basée sur le préfixe unicast
    - Si P=1 → T=1
  - Reserved : 0
  - Plen : nombre de bits du préfixe réseau
  - Préfixe réseau avec tous les bits non significatifs à 0
  - Ex: préfixe 2001:660::/32 (RENATER)
    adresse FF3E:20:2001:660:0:0:1234:abcd

# IPv6 networks management

Bernard.Tuy@renater.fr

# Agenda

- Introduction
- IPv6 MIBs: current status
- Management platforms
- Home made tools/ GPL Software
- Management tools
  - IPv6 LAN
  - IPv6 MAN/WAN
- Examples
- Conclusion

# Introduction

- Few (still) IPv6 only networks deployed
- Most are dual stack
  - LANs (campuses, companies, …)
  - MANs (RAP, …)
  - WANs -ISPs (Géant, NRENs, IIJ, Abilene, …)
- Testbed / pilote net / production …

  => Management tools are needed
- Which applications available for managing these nets ?
  - Equipments, configurations, …
  - IP services (servers : DNS, FTP, HTTP, …)

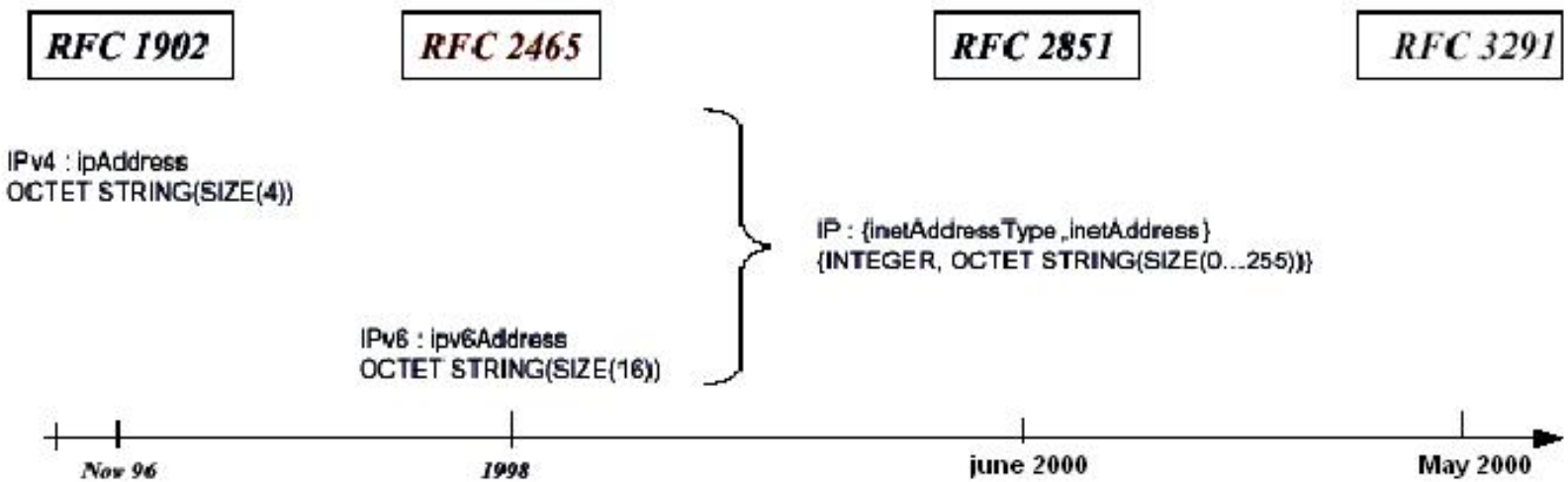
# IPv6 MIBs status (1/4)

- MIBs are essential for the network management.
- SNMP-based applications are widely used but others exist too.
- SNMP rely upon MIBs …

  =>Need to have MIBs for IPv6.

# IPv6 MIBs (2/4)

- Standardization status at IETF:
  - At the beginning:
    - IPv4 and IPv6 MIBs dissociated.
  - Today :
    - Unified MIBs are on standardization track.



# IPv6 MIBs(3/4)

- Internet drafts: Revision of the IP MIB definition in order to integrate IPv6.
  - RFC 2011: IP MIB: ID 05.txt (12/2003)
  - RFC 2012: TCP MIB: ID 05.txt(11/2003)
  - RFC 2013: UDP MIB: ID 02.txt (11/2003)
  - RFC 2096: IP forwarding MIB : ID 05.txt (08/ 2003)

# IPv6 MIBs implementations

- Equipment manufacturers:
  - Cisco:
    - Proprietary MIBs implement early versions of IDs based on RFC 2011
    - But, no distinction between IPv4 and IPv6 traffic (=> available in S2/2004)
    - To get this info : CLI (*sh int accounting …*)
  - Juniper:
    - MIB based on RFC 2465 (with different counters for IPv4 and IPv6 traffic).
  - 6Wind:
    - MIB based on RFC 2465 and RFC 2466

# Netflow for IPv6

- Cisco
  - Ready since …
    - Latest : IOS 12(3)7T (Feb. 2004)
    - Compliant Netflow v9
    - NFC v5 available

  =>Not yet there for GSRs though

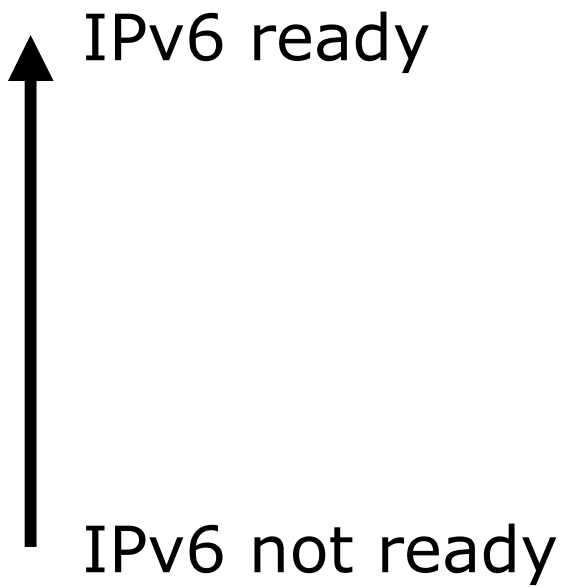
  *12.0(27)S EFT includes SNMPv6*
- Other vendors ?
  - Input from the audience …

# Management platforms

- Commercial ISPs use to have integrated management platforms (NRENs folks mostly use GPL or Home made tools)
- **HP-OV** proposes a version with IPv6 features: NNM 7.0 (sept 2003).
- **Ciscoworks**: IPv6 version planned for S2 2004
- Ciscoworks and Ciscoview
  - Application note on IPv6 management
- **Netview** (IBM) doesn't propose any IPv6 features ?
- **Tivoli** : no information …
- **Infovista** : « no IPv6 plan at the moment »

# « Top ten » …

- HP Openview
- Cisco NetFlow v9
- Ciscoworks 2000
- IBM Netview
- Infovista, Tivoli



# How to manage an IPv6 network ?

- Dual stack IPv6 networks
- IPv6 only
  - There are not the main case …
  - … important to think / know IPv4 could be removed

# Dual Stack IP networks

- Part of the monitoring via IPv4
  - Connectivity to the equipment
  - Tools to manage it (inventory, configurations, «counters», routing info, …)
- Remaining Part needs IPv6
  - MIBs IPv6 availability
  - NetFlow (v9)

# IPv6 only networks

- Topology discovery (LAN, WAN ?)
- IPv6 SNMP agent
- SNMP over IPv6 transport

=> Need to identify the missing bits

# Monitoring tools for IPv6 networks

- For a LAN:
  - Nagios
  - Argus
  - MRTG

  …
- For a MAN/WAN:
  - AS PATH tree
  - Weather map
  - Netflow
  - Rancid
  - Looking Glass

# 6Net and IPv6 monitoring tools

- 6Net :
  - 3 years IST project
  - EC half-funded (12 M€uros)
  - 34 partners from EU and Korea
- 6Net wp6 : managing large scale IPv6 nets
  - Tests lot of ipv6 ready tools
  - Port many others to ipv6

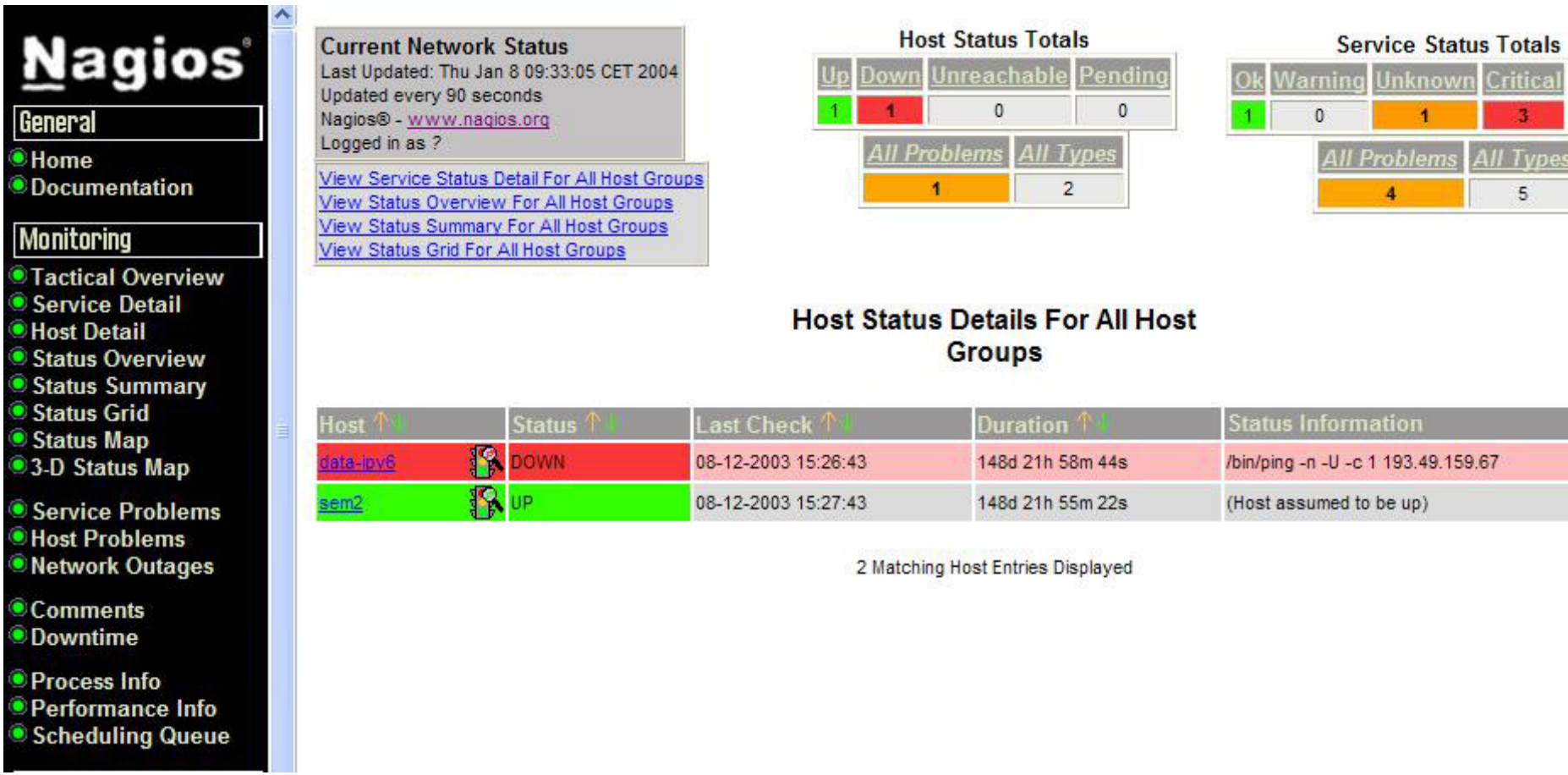
# 6Net outcome

- 30+ monitoring tools for IPv6
  - Tested
  - Implemented
  - Documented
- URL: http://www.6net.org/ …
  - To be publicly accessible in a few weeks

# IPv6 LAN management: Nagios

- URL://www.nagios.org
- Administration of network:
  - PCs
  - Switches
  - Routers
- Administration of services:
  - http, ftp, dns...
- Evolution: new features can be added with plug-ins

# Nagios

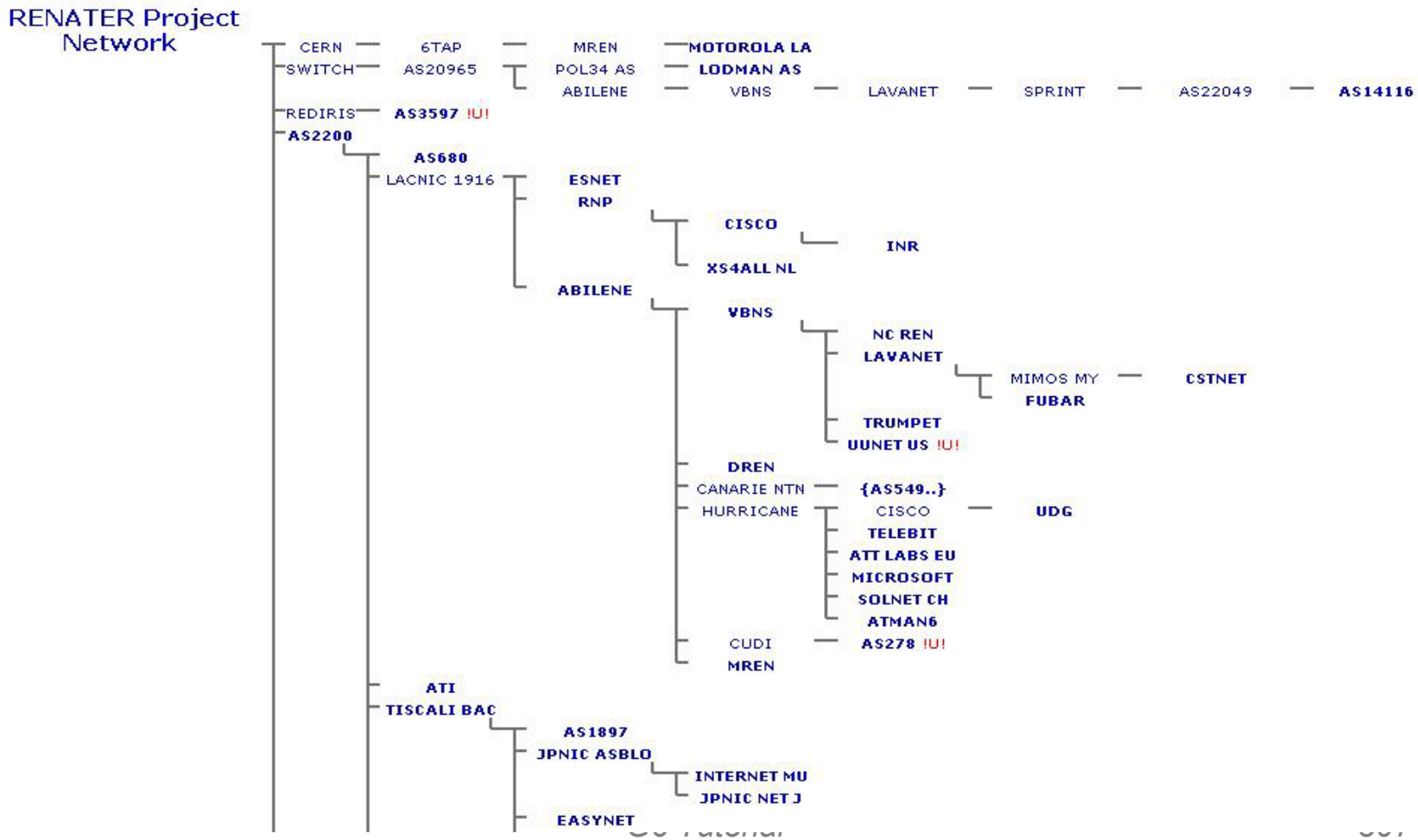

Nagios
General
Home
Documentation
Monitoring
Tactical Overview
Service Detail
Host Detail
Status Overview
Status Summary
Status Grid
Status Map
3-D Status Map
Service Problems
Host Problems
Network Outages
Comments
Downtime
Process Info
Performance Info
Scheduling Queue
Current Network Status
Last Updated: Thu Jan 8 09:33:05 CET 2004
Updated every 90 seconds
Nagios® - www.nagios.org
Logged in as ?
View Service Status Detail For All Host Groups
View Status Overview For All Host Groups
View Status Summary For All Host Groups
View Status Grid For All Host Groups
Host Status Totals
Up
Down
Unreachable
Pending
1
1
0
0
All Problems
All Types
1
2
Service Status Totals
Ok
Warning
Unknown
Critical
1
0
1
3
All Problems
All Types
4
5
Host Status Details For All Host Groups
Host
Status
Last Check
Duration
Status Information
data-ipv6
DOWN
08-12-2003 15:26:43
148d 21h 58m 44s
/bin/ping -n -U -c 1 193.49.159.67
sem2
UP
08-12-2003 15:27:43
148d 21h 55m 22s
(Host assumed to be up)
2 Matching Host Entries Displayed

# IPv6 MAN/WAN management: AS Path Tree

- Display BGP4+ « topology » from
- BGP4+ routing table.
- Generate HTML pages.

# AS Path Tree

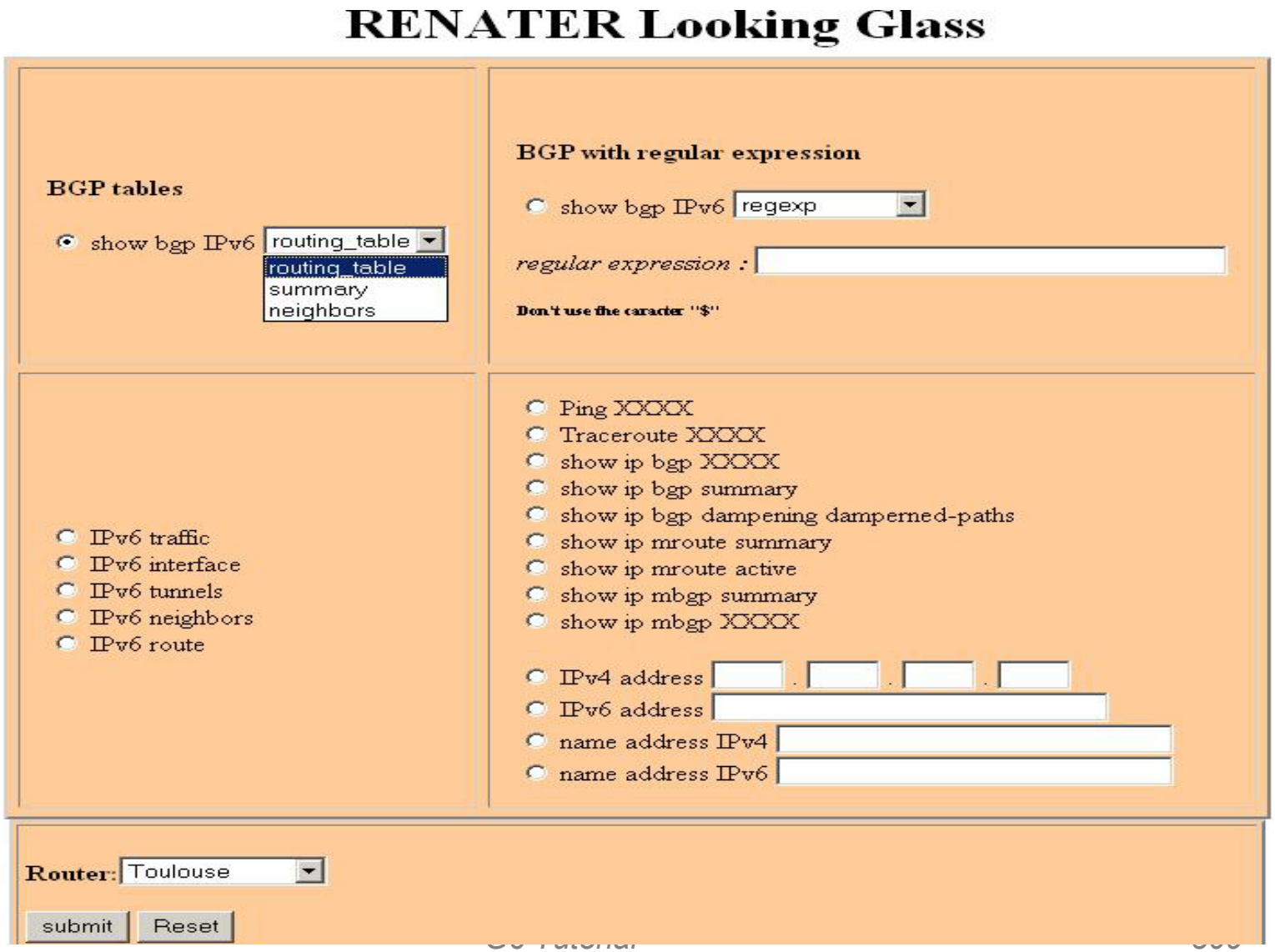
## Renater The whole IPv6 BGP table



# IPv6 MAN/WAN management : Looking Glass

- Get information on a router w/o direct connection
- Web Interface
- Final user don’t need a login
- Allow the user to detect causes of failures w/o asking the NOC

# Looking Glass



# Conclusion

- ISPs need monitoring tools to start a new service/protocol
- For IPv6 a lot of them are already there
- BUT: stress your favourite vendor reminding him what your needs are !
- And be active in the relevant IETF WGs !