

IPv6

INTERNET PROTOCOL V6

**OVERVIEW OF IPv6, THE FUTURE
PROTOCOL FOR SCALING UP THE INTERNET
AND ENABLING THE INTERNET OF THINGS**

Peter R. Egli
peteregli.net

Contents

1. Relevant IPv6 RFCs
2. Why IPv6?
3. IPv6 – the holy grail?
4. Main differences between IPv4 and IPv6
5. The past, the present and the future of IPv6?
6. IPv6 extension headers
7. IPv6 addresses
8. IPv6 route aggregation versus IPv6 multihoming
9. IP address assignment with IPv6
10. IPv6 fragmentation
11. IPv6 neighbor discovery (ND) protocol
12. Migration steps for transition from IPv4 to IPv6

1. Relevant IPv6 RFCs

RFC2460 „Internet Protocol, Version 6 (IPv6) Specification“

RFC4291 „IP Version 6 Addressing Architecture“

RFC3587 „IPv6 Global Unicast Address Format“

RFC4213 „Transition Mechanisms for IPv6 Hosts and Routers,,

RFC3056 „Connection of IPv6 Domains via IPv4 Clouds,,

RFC2529 „Transmission of IPv6 over IPv4 Domains without Explicit Tunnels,, („6over4“)

RFC4862 „IPv6 Stateless Address Autoconfiguration“

RFC6177 „IAB/IESG Recommendations on IPv6 Addresses“

RFC3484 „Default Address Selection for Internet Protocol version 6 (IPv6)“

RFC6145 „IP/ICMP Translation Algorithm“

RFC4861 „Neighbor discovery protocol“

RFC3879 "Deprecating Site Local Addresses"

RFC4147 "IANA IPv6 Registry"

RFC3849 "IPv6 Address Prefix Reserved for Documentation"

Various RFCs devoted to the different migration scenarios.

Obsoleted IPV6 concepts:

Some concepts in IPv6 have already been obsoleted (e.g. site-local unicast addresses).

These are left in this document for documentary purposes but are marked in light grey text.

2. Why IPv6?

Motivation for IPv6:

1. Exhausted IPv4 address space:

As of 2011, V4 address space is virtually exhausted (only 4.3G addresses) despite NAPT and CIDR.

2. IPv4 addresses are non-hierarchical:

V4 addresses are non-hierarchical and assigned irrespective of geographical topology. This leads to fragmentation and thus big routing tables (as of 2010 over 320k route prefixes to be exchanged between backbone routers).

See <http://bgp.potaroo.net/> or <http://www.cidr-report.org/>.

3. Disproportionate IPv4 address assignment:










IPv4 addresses are assigned disproportionately (2005: USA 75%, Asia only ~10%, China < 1%).

4. IPv4 address management is difficult:

IPv4 does not really support automatic address assignment (except APIPA). Usage of DHCP means high administrative effort.

More statistics on IPv4: <http://www.potaroo.net/tools/ipv4/index.html>

3. IPv6 – the holy grail?

-  IPv6 solves the address scarcity problem (for now).
-  IPv6 should solve the route table size problem in backbone routers.
-  IPv6 comes with improved QoS support for real-time applications.
-  IPv6 will be one of the drivers of mobility (always-on mobile devices).
-  Security was an integral part of IPv6 from its inception (IPSec).
-  IPv6 has a simplified header thus greatly reducing routing processing load.
-  IPv6 is designed to scale almost indefinitely (to very large networks); the protocol should support routing speeds for OC-12+ (622Mbps) lines and beyond.
-  IPv6 is plug-and-play: automatic IP address assignment (no DHCP), router solicitation for getting the network prefix and router advertisement for making own IP address known to neighbors.
-  IPv6 is not something revolutionary new. It is designed to be as transparent to applications as possible while solving the biggest problems and deficiencies of IPv4.

4. Main differences between IPv4 and IPv6

1. Header is simplified, has fixed size (40bytes); IPv6 introduces the concept of (optional) extension headers for fragmentation, header options etc.
2. Header checksum removed; this function is already covered by layer 2 protocols (e.g. Ethernet and Frame Relay). Anyway, the IPv4 checksum does not provide Forward Error Correction (possibility to correct errors based on the checksum) thus it is basically useless (routers have to drop errored packet anyway).
3. Bigger addresses (128 bits as opposed to 32 bits in IPv4).

IPv4 header:

Ver.	IHL	TOS	Total length	
Identification		Frag.	Fragment offset	
TTL	Protocol	Header checksum		
IP source address				
IP destination address				
Optional IP options				



Field discarded in IPv6.



Function / field retained in IPv6, but used/encoded differently.



Field retained in IPv6.

IPv6 header:

Ver.	T. class	Flow label		
Payload length		Next H.	Hop limit	
IP source address				
IP destination address				
Optional extension headers				

5. The past, the present and the future of IPv6?

→ IPv6 efforts began in the early 90ies.

→ Where is IPv5?

IP protocol version 5 was already assigned to another protocol (ST: Streaming protocol for real-time traffic over the Internet). Initially IPv6 efforts ran under the name IPng (next generation IP). One of the predecessors of IPv6 was called SIPP (Simple IP Plus).

→ IPng was the predecessor of IPv6 and consisted of 3 proposals:

CATNIP: „Common Architecture for Next Gen. Internet Protocol“, created commonality between Internet (IPv4, TCP, UDP), OSI (CLNP) and Novell (IPX).

TUBA: „TCP and UDP Using Bigger Addresses“ using OSI’s CLNP.

SIPP: „Simple IP Plus“, removed IPv4 functions that did not work, increased address size to 64bit.

A revised version of SIPP (128bit addresses, auto-configuration) was chosen as the basis for IPng which eventually became IPv6. See [RFC1752](#).

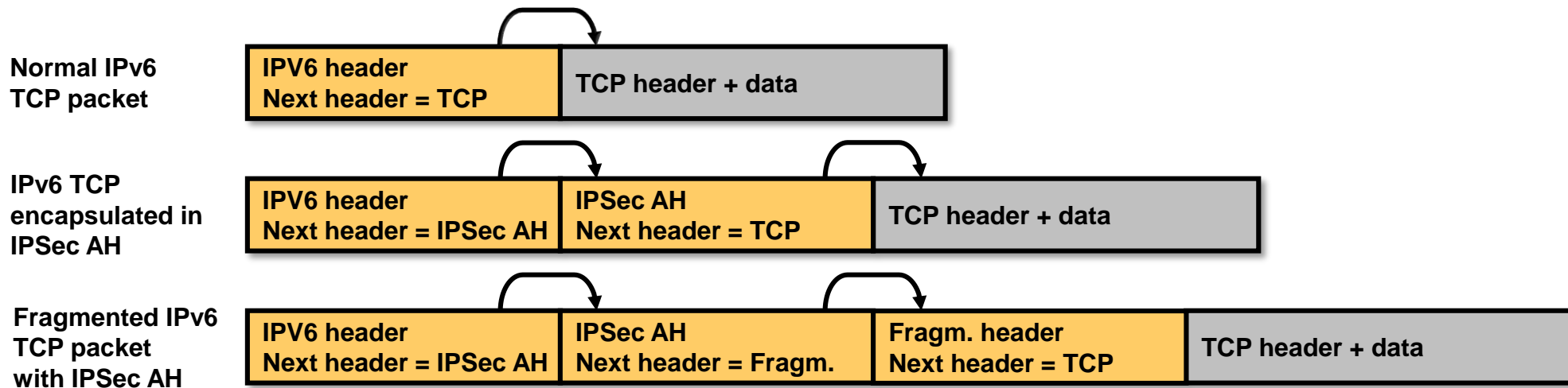
→ 6Bone: IPv6 testbed for the deployment of IPv6.

→ IPv6 is still not widely deployed (as of 2012). IPv6 adoption rate is still very low. But recent activities show that adoption rate is picking up speed (IPv6 day, permanent IPv6 reachability availability of well known web sites like google).

Mobility (mobile devices) may be a real driver for the adoption of IPv6 (killer application).

6. IPv6 extension headers

→ Optional functions have been moved to (optional) extension headers (next header mechanism). Thus the header has been streamlined for the common case (common case must be fast, less often used functions like fragmentation are moved to optional headers). Next headers can be stacked in a pre-defined order:



→ Possible extension headers (must be stacked in the order given):

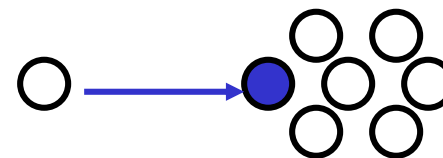
1. Hop-by-hop options (options are evaluated at each hop)
2. Routing header (like loose source routing and record route in IPv4)
3. Fragmentation header (only transmitting node can fragment, not routers along the path)
4. Destination options (options evaluated by receiver)
5. AH header (IPSec)
6. ESP header (IPSec)
7. Upper layer header (TCP)

7. IPv6 addresses (1/14)

7.1 IPv6 address types:

A. Unicast address:

Same as IPv4 unicast address.



B. Multicast address:

In IPv4 there were multicast addresses, but only for experimental use. Multicast addresses are an integral part of IPv6.

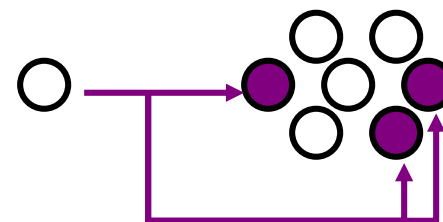
Multicast addresses: FF0x::<group ID>

x=1 = interface local

x=2 = link local

x=5 = site local

x=E = global



C. Anycast address:

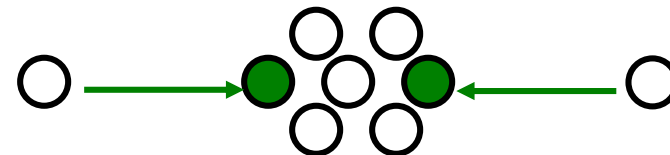
Anycast addresses are new in IPv6.

Anycast packets are routed to the nearest host.

The nearest host is ascertained by routing protocols.

Anycast addresses are syntactically indistinguishable from unicast addresses.

Anycast address = configuration of same unicast address on multiple interfaces and configuration of routing such that it routes a packet to this address to the nearest interface having this address.



N.B.: There are no broadcast addresses in IPv6 (multicast replaces broadcast).

7. IPv6 addresses (2/14)

7.2 IPv6 address scope (validity in a specific area):

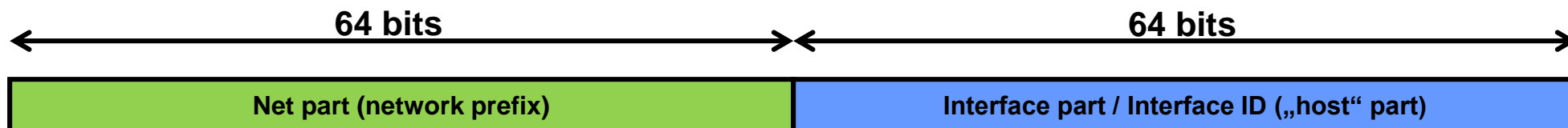
Link local scope: IP address is valid only within a specific link (e.g. Ethernet link).

Site local scope: IP address is valid only within a specific site (e.g. enterprise, university).

Global scope: IP address is globally unique.

N.B.: In IPv6, there are no address classes (like A, B, C in classful IPv4).

7.3 General structure of IPv6 address (as proposed by [RFC6177](#)):



Network prefix: „Where are you connected to“.

Interface ID: „Who are you“. Created from MAC address or from IPv4 address (IPv6 compatible addresses). See [RFC4291](#) 2.5.

Unlike IPv4, IPv6 addresses are hierarchical to allow route aggregation (prefix). The prefix boundary can fall anywhere within the address („classlessness“).

N.B.: In IPv6, there are no hosts anymore. Every address specifies an interface and not a host. A host is expected to have multiple interfaces („multi-homed“ host).

7. IPv6 addresses (3/14)

7.4 IPv6 address notation (RFC4291):

Due to the much higher number of bits, the representation was changed from decimal to hex (colon-hexadecimal notation).

x:x:x:x:x:x:x:x (x = 16 bit hex), e.g. **1080:0000:0000:0000:0008:0800:200C:417A**.

Prefix length („mask“ length):

The prefix length is suffixed with „/x“ (node address and prefix length).

E.g. **1080::8:800:200C:417A/48** or **2002::/16**

IPv4 style masks (e.g. 255.255.0.0) do not exist in IPv6.

Shorthand writing:

In order to ease writing, some shorthands have been defined:

1. Remove leading 0 in 16 bit groups (leading 0 in each 16 bit hex-block can be omitted):

→ There must be at least one digit in each 16 bit group.

E.g. **1080:0:0:0:8:800:200C:417A**

2. Collapse 0000 (multiple groups of 16 bit 0 (0000 in hex) can be collapsed into „::“):

→ Only complete and adjacent 0000 groups can be collapsed.

→ „::“ may occur only once in the address.

E.g. **1080::8:800:200C:417A**

Mixed IPv4/IPv6 format:

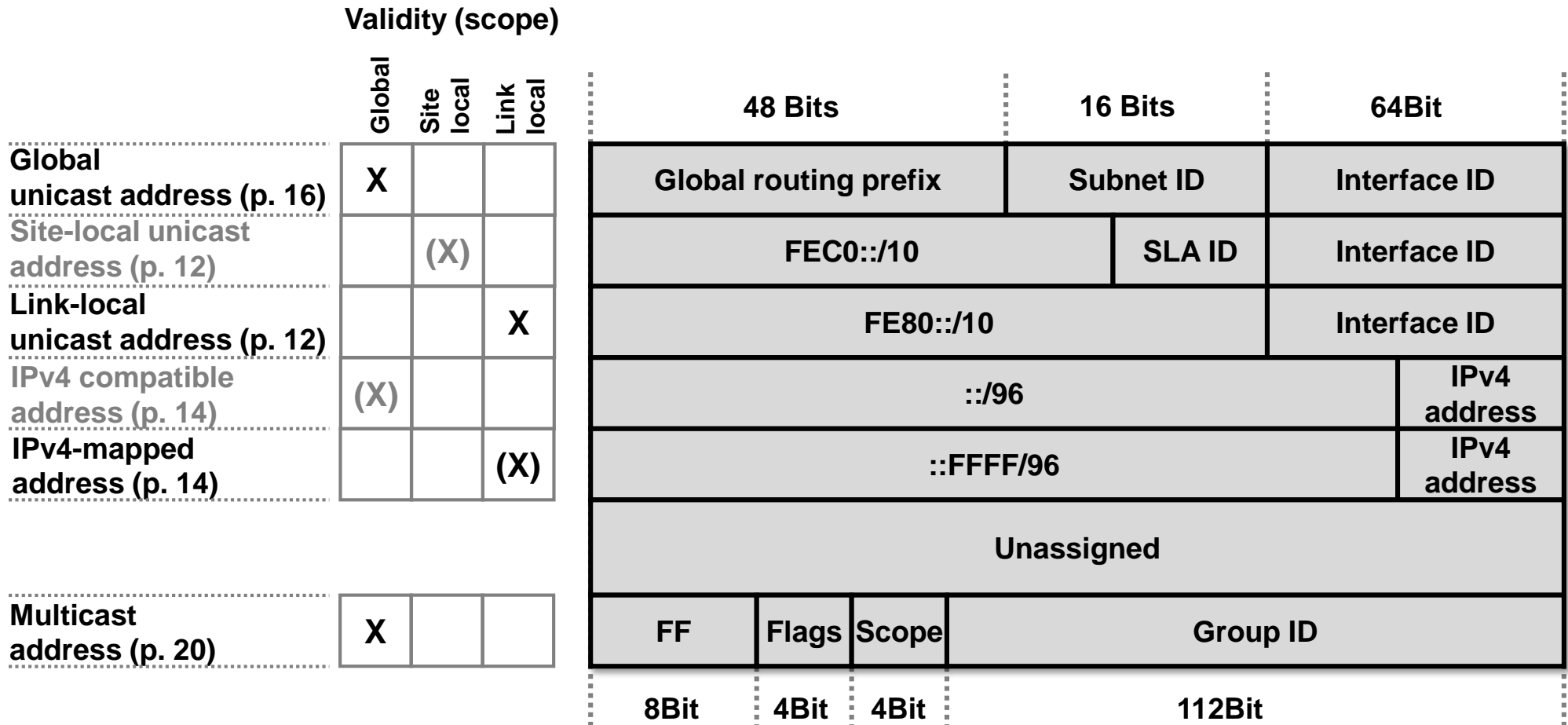
x:x:x:x:x:d.d.d.d where x = hex-16-bit representation of high order bits and d = IPv4 notation.

E.g. **0:0:0:0:0:FFFF:129.144.52.38 = ::FFFF:129.144.52.38** (IPv4-mapped address).

7. IPv6 addresses (4/14)

7.5 IPv6 addressing architecture RFC4291:

RFC4291 defines the addressing architecture of the IPv6 address space.



7. IPv6 addresses (5/14)

7.6 Special IPv6 addresses (1/2):

A. Local Loopback address (only 1 single address as opposed to IPv4):

0000:0000:0000:0000:0000:0000:0000:0001 = ::1/128

B. Unspecified address (similar to 0.0.0.0 in IPv4):

0000:0000:0000:0000:0000:0000:0000:0000 = ::/128

Meaning: Absence of address or invalid address.

C. IPv6 multicast:

FF00::/8

D. Link-local unicast:

FE80::/10

Link-local addresses are used on a link for automatic address configuration, neighbor discovery or when no routers are present on the link.

→ These addresses are not routed (valid only on a link such as Ethernet).

E. Site-local unicast (obsoleted, see RFC3879):

FEC0::/10

Originally intended to be used within a site (similar to link-local, but valid within a site). Definition of a "site" was too fuzzy (organization, company) so the concept of link-local addresses was abandoned.

7. IPv6 addresses (6/14)

7.6 Special IPv6 addresses (2/2):

F: Unique Local Address (ULA):

FC00::/8

RFC4193 defines unique local addresses analogous to IPv4 private addresses (10.0.0.0/8, 172.16.0.0/12 and 192.168.0.0/16, see RFC1918).

- Unique local addresses contain a randomly generated part to make the address unique.
- Unique local addresses avoid address conflicts (e.g. when establishing a tunnel between 2 sites that are independently configured sites).
- Easy filtering at site boundaries (avoid leaking of packets to the Internet).

G: IPv6 Address Prefix for Documentation:

2001:0DB8::/32

In order to avoid confusion, IETF set aside a special range of IPv6 addresses to be used in documentation (and not to be used in real deployments). See RFC3849.

7. IPv6 addresses (7/14)

7.7 IPv4/IPv6 compatibility addresses (1/2):

A number of special addressing schemes and algorithms are defined in the various migration technologies.

A. IPv4 compatible address (obsoleted):

`0:0:0:0:0:0:w.x.y.z`

Used by IPv6/IPv4 nodes that are communicating with IPv6 over an IPv4 infrastructure. When the IPv4-compatible address is used as an IPv6 destination, the IPv6 traffic is automatically encapsulated with an IPv4 header and sent to the destination using the IPv4 infrastructure. IPv4 compatible addresses are obsoleted as transition mechanism do not use it anymore.

B. IPv4-mapped address:

`0:0:0:0:0:FFFF:w.x.y.z` or `::FFFF:w.x.y.z`,

Used to represent an IPv4-only node to an IPv6 node (SIIT). It is used only for internal representation. The IPv4-mapped address is never used as a source or destination address of an IPv6 packet. The IPv4-mapped address is used by some IPv6 implementations when acting as a translator between IPv4-only and IPv6-only nodes (e.g. used by RFC2765 SIIT = stateless IPv4 to IPv6 address translation).

C. IPv4-translated address (used by RFC2765 SIIT stateless IPv4 to IPv6 address translation):

`0::FFFF:0:a.b.c.d`

Used to represent an IPv6-enabled node.

7. IPv6 addresses (8/14)

7.7 IPv4/IPv6 compatibility addresses (2/2):

D. 6to4 addresses:

2002::WWXX:YYZZ::[subnet-ID]:[InterfaceID]/48 (colon-hexadecimal notation)

Used by RFC3056 6to4 tunneling.

E. 6over4 addresses:

FE80::WWXX:YYZZ (colon-hexadecimal notation)

Example: IPv4 131.107.4.92 → 6over4 IPv6 address **FE80::836B:45C**

F. ISATAP addresses:

Valid 64-bit unicast prefix and interface identifier **0:5EFE:w.x.y.z**

Example: **FE80::5EFE:131.107.4.92** (link local)

G. Teredo addresses (NAPT traversal):

Use of prefix **3FFE:831F::/32**

Example: **3FFE:831F:CE49:7601:8000:EFFE:62C3:FFFE**

H. IPv4-translatable addresses (defined in RFC6052, used by RFC6145 and RFC6146):

IPv4 address embedded in IPv6 address starting at bit positions 32, 40, 48, 56, 72 or 96.

Example 1: **2001::0DB8:1C6:3364:02::** (IPv4 address = 198.51.100.2)

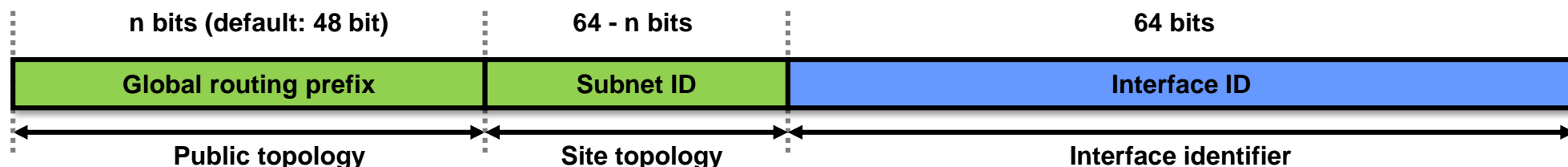
Example 2: **2001::0DB8:1000:C633:0064:02::**

7. IPv6 addresses (9/14)

7.8 "IPv6 global unicast address" = main address type in IPv6 (see [RFC3587](#)) (1/2):

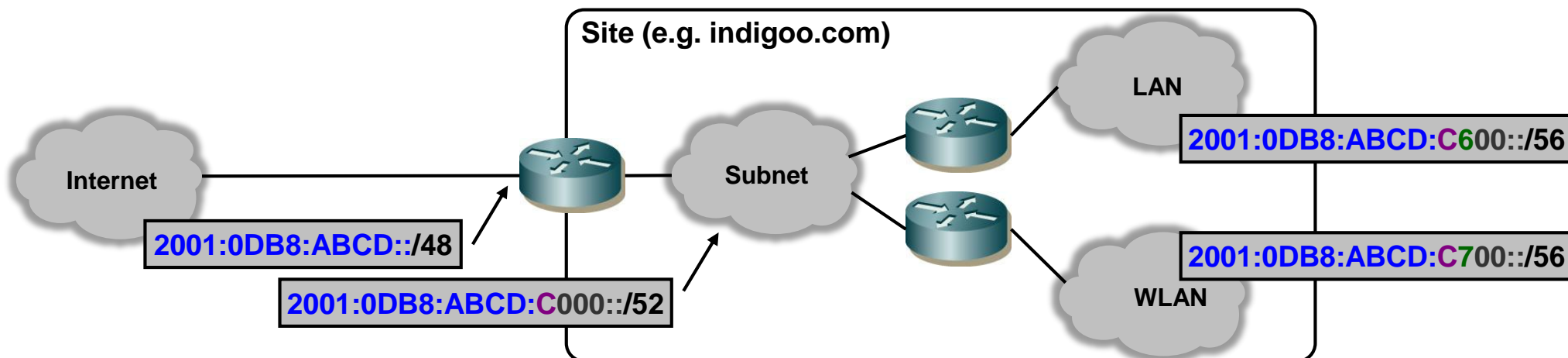
Aggregation is used for of reducing routing tables (one of the main goals of IPv6).

Format (see [RFC3587](#)):



The global routing prefix, usually 48 bits, identifies a site (organization, company), i.e. a cluster of subnets / links. In special cases, ISPs may use smaller prefixes (for very large organizations) or 64 bit prefixes (customer only needs exactly 1 address).

The subnet ID identifies a subnet within a site.

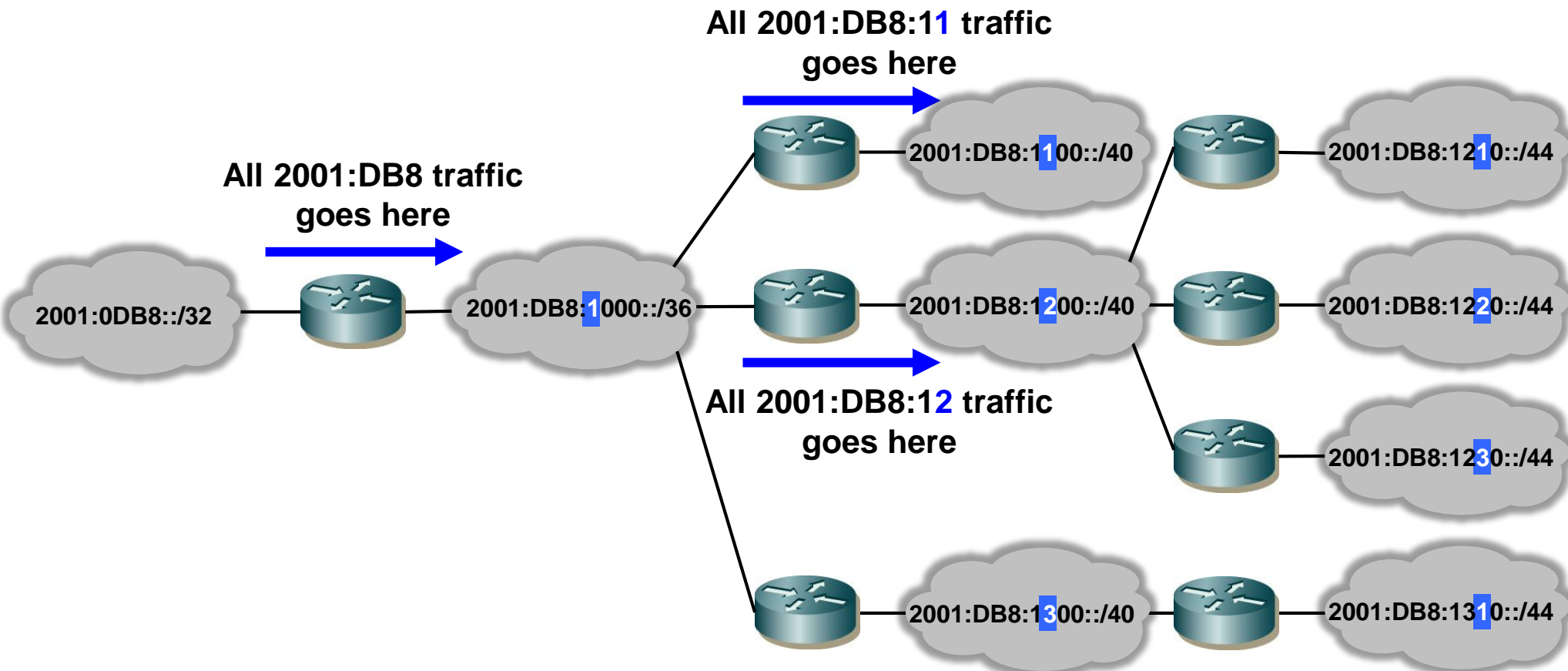


7. IPv6 addresses (10/14)

7.8 "IPv6 global unicast address" = main address type in IPv6 (see [RFC3587](#)) (2/2):

→ Hierarchical addresses allow assigning addresses according to geographical topology thus reducing routing tables (prefixes can be aggregated).

→ The proposed aggregatable unicast address format is a tradeoff between minimizing routing tables and flexibility in IP address allocation.



7. IPv6 addresses (11/14)

7.9 IPv6 zone identifier (1/2):

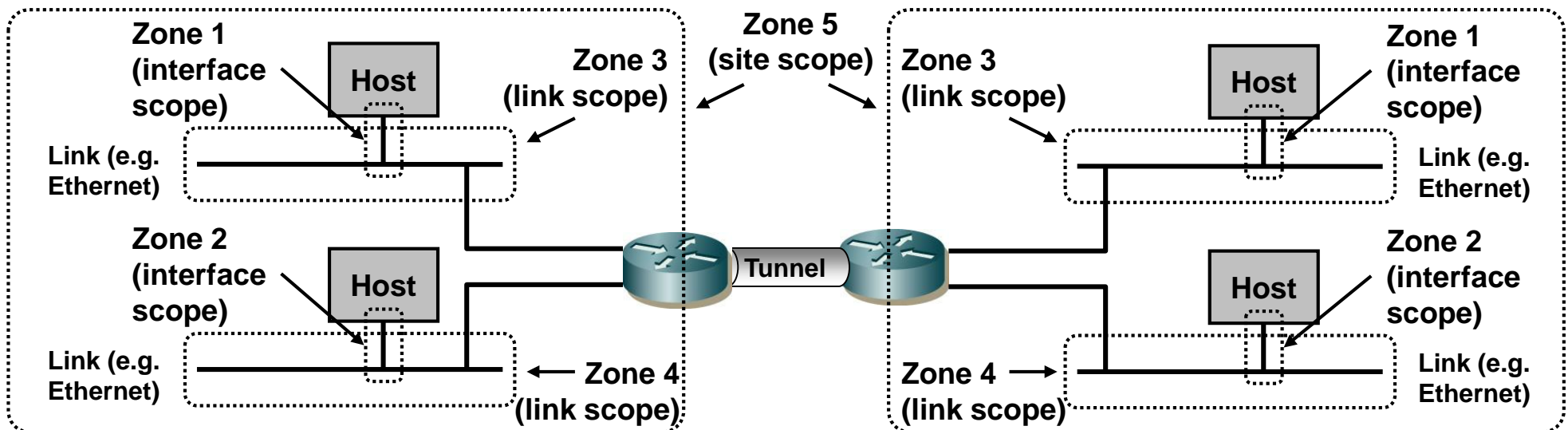
IPv6 address scopes:

A scope defines the validity region of IPv6 addresses (topological span within which an address is unique).

- a. Interface-local scope → Only valid on local interface
- b. Link-local scope → Only valid on link to which interface is attached
- c. Site-local scope → Only valid within local site (deprecated)
- d. Global scope → Globally valid

IPv6 (scope) zones:

A zone is a connected region of topology of a given scope. A zone is a particular instance of a topological region (e.g. company zone or your computer's Ethernet link) whereas scope is the validity / size of the region (e.g. link or site).



7. IPv6 addresses (12/14)

7.9 IPv6 zone identifier (2/2):

Problem:

Addresses without global scope (interface, local, site) are only unique within their scope. These addresses may be reused, e.g. an address with link scope may be reused on another link.

The zone to which a particular IP address pertains is not encoded in the IP address. It must be rather determined from the context, i.e. from the link over which a packet was received.

Normal IP routing can not determine the destination interface based on the prefix (which is not unique for link local addresses).

→ [RFC4007](#) defines a zone ID that identifies the zone to which an IP address belongs.

Example: `FE80::1%1`

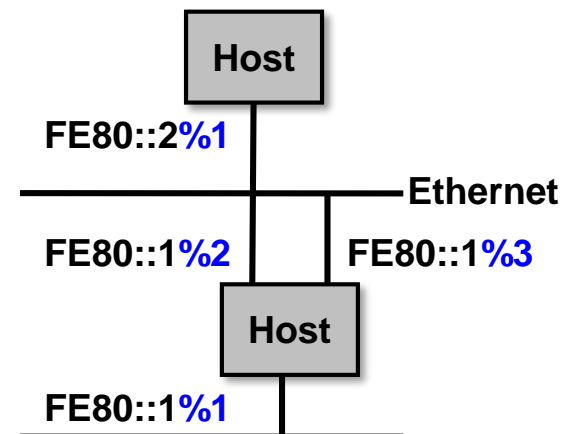
Configuration of zone identifier:

The configuration of the zone index should be automatic (avoid manual configuration).

Each link to which an interface is attached has its own link index which is used as zone index.

Usage of zone identifier:

The zone identifier is specified by the application, e.g.:
`ping fe80::511a:886c:a8cc:dc66%11`



7. IPv6 addresses (13/14)

7.10 Multicast addresses:

Multicast addresses allow to reach multiple destinations. Multicast addresses replace broadcast addresses.

Structure of IPv6 multicast address (as per RFC4291):

Scope limits the scope to:

- | | |
|-------------------------|----------------------------|
| 0 reserved | 6, 7 (unassigned) |
| 1 Interface-Local scope | 8 Organization-Local scope |
| 2 Link-Local scope | 9..D (unassigned) |
| 3 reserved | E Global scope |
| 4 Admin-Local scope | F reserved |
| 5 Site-Local scope | |

Multicast group ID

E.g. FF0E:0:0:0:0:0:0:101 = NTP multicast with global scope.



Multicast prefix

0 = Permanently assigned (=well-known multicast address, assigned by IANA)

1 = Transient or dynamically assigned

0 = Multicast address that is not assigned based on the network prefix

1 = Multicast address that is assigned based on the network prefix

Definition see [RFC3956](#)

IANA: Internet Assigned Numbers Authority
NTP: Network Time Protocol

7. IPv6 addresses (14/14)

7.11 Literal IP addresses in URLs (use of IP addresses in URLs):

URLs may contain numerical IP addresses as follows (though it is not recommended to use this feature!):

IPv4:

<http://193.5.54.123:80>

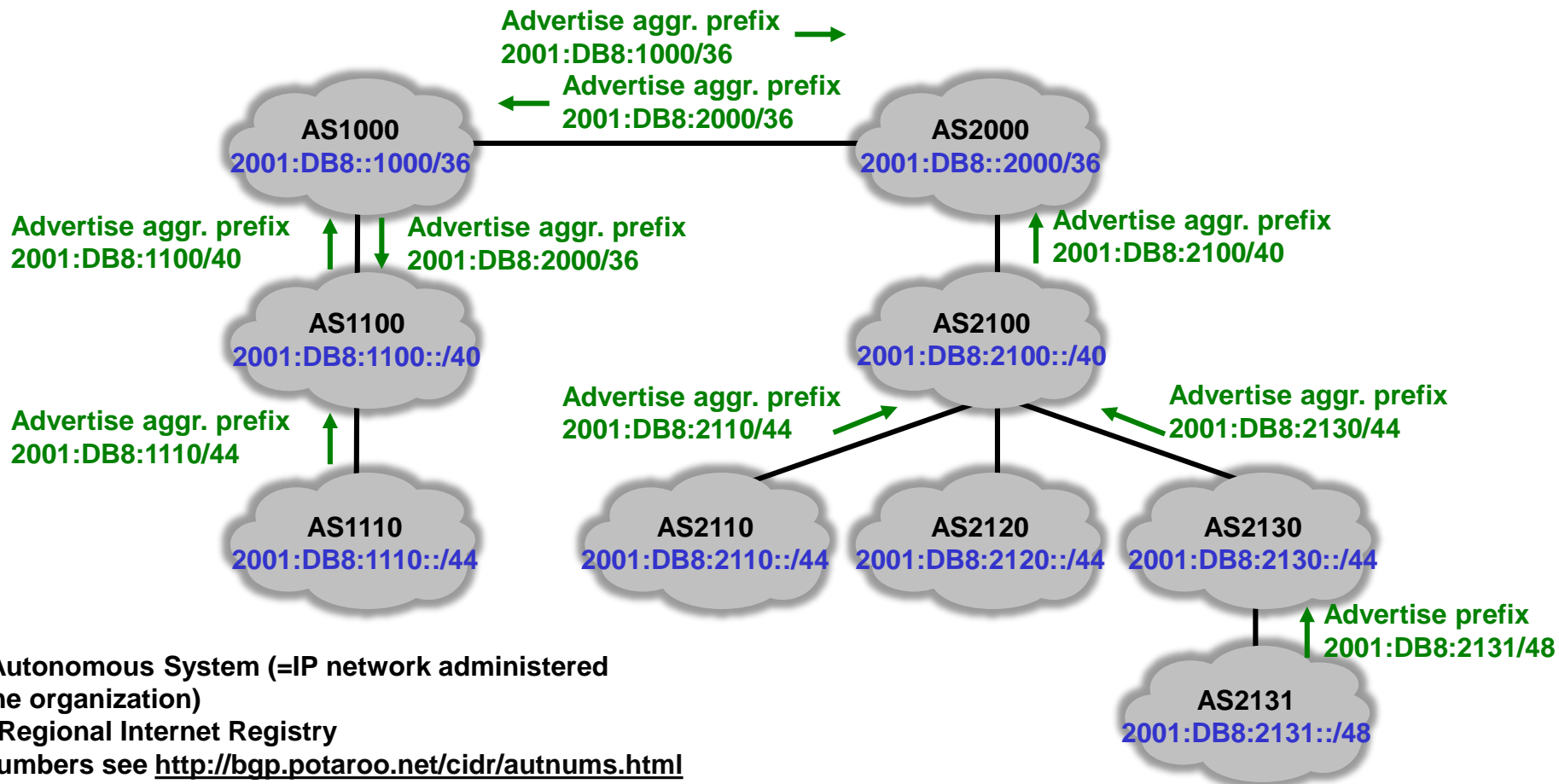
IPv6 (see RFC3986):

[http://\[2001:DB8::7\]/index.html](http://[2001:DB8::7]/index.html)

8. IPv6 route aggregation versus IPv6 multihoming (1/2)

Route aggregation is very important in IPv6 in order to reduce the number of routing entries in IPv6 routers (remember: IPv4 @ Y2010 ~320k route prefixes).

RIRs assign Provider Aggregatable (PA) address blocks to providers. These blocks can be aggregated into a single route advertisement.



8. IPv6 route aggregation versus IPv6 multihoming (2/2)

Problem: How to minimize routing table size and provide redundancy?

Redundancy can be achieved through multihoming, i.e. connect a site to multiple providers.

When using *Provider Independent* address space (PI), the same address range can be advertised to multiple providers (2001:DB8:2110/44 in picture below).

But: PI addresses "punch holes" into the routing tables (increases the number of routing entries).

AS1100 interior routing tables before "punching hole":

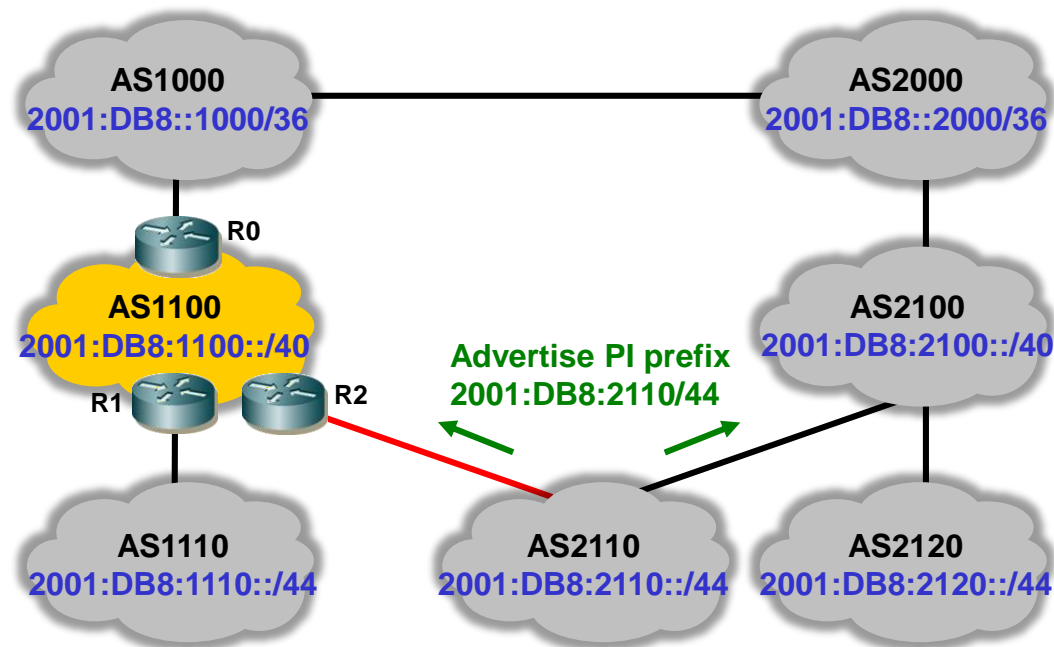
2001:DB8:2000/36 via R0
2001:DB8:1110/44 via R1

AS1100 interior routing tables after "punching hole":

2001:DB8:2000/36 via R0
2001:DB8:1110/44 via R1
2001:DB8:2110/44 via R2

Address space:

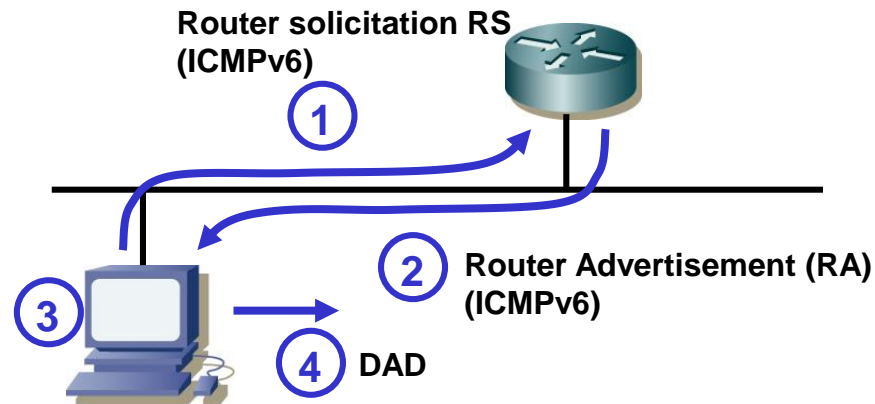
2001:DB8:2000/36
2001:DB8:2110/44
2001:DB8:2FFF/36



With multihoming and advertising 2001:DB8:2110 to 2 interfaces, 2001:DB8:2110 is reachable through R2 and R0 in AS1100.

9. IP address assignment with IPv6 (1/2)

9.1. IPv6 Stateless Address Autoconfiguration RFC4862



1. Host sends ICMPv6 router solicitation packet (on Ethernet and IPv6 multicast address).
2. Router sends back an RA message with the global prefix (network part of IP address).
3. The host creates his IPv6 address from the global prefix (network part) and the EUI-64 host part generated from the MAC address.
4. The host sends an ICMPv6 neighbor solicitation packet with its own IPv6 address (Duplicate Address Detection - DAD). If no neighbor responds, then the IP address state is changed to „assigned“.

Option:

The router may send RA messages with 2 flags:

ManagedFlag 1 → The host should use stateful autoconfiguration (DHCPv6).

OtherConfigFlag 1 → The host should query other information from a DHCPv6 server (e.g. DNS server).

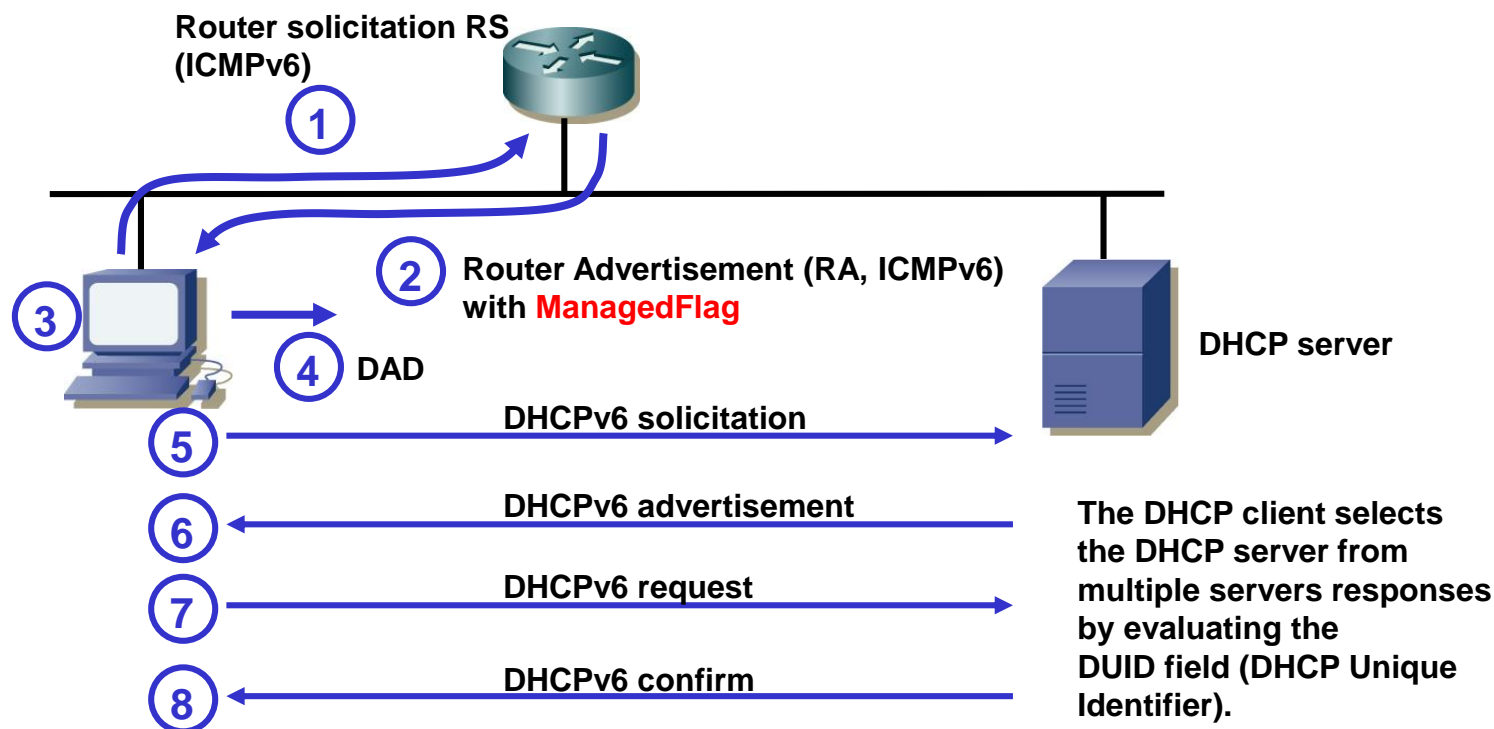
ManagedFlag see below.

9. IP address assignment with IPv6 (2/2)

9.2. IPv6 Stateful Address Autoconfiguration with DHCPv6 RFC3315

Why stateful address assignment if IPv6 provides unique IP addresses for each interface?

- No DNS server and default gateway provided with stateless address autoconfiguration
- Centrally administered IP prefix
- Legal and forensic requirements (which computer used which IP addresses when, logging)



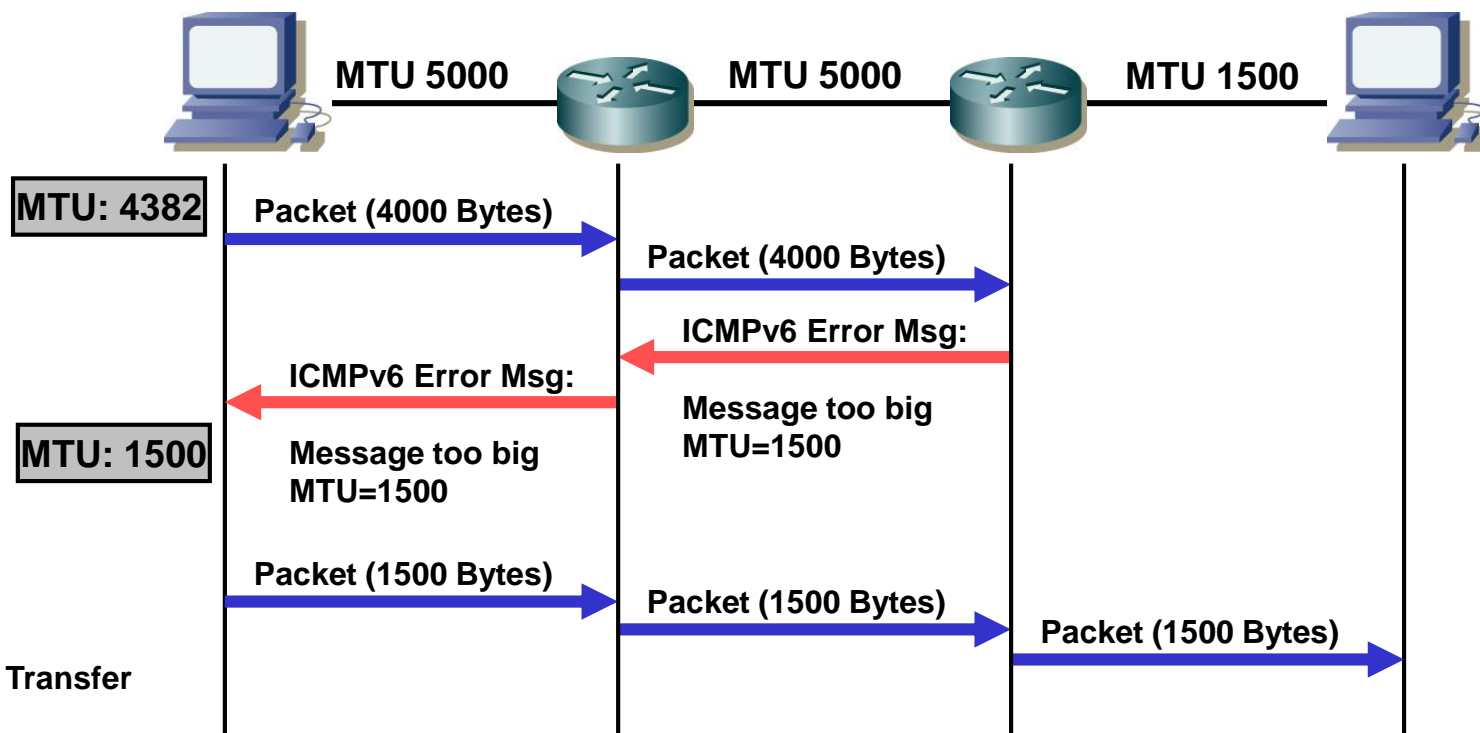
Router solicitation is the standard mechanism to get an IP address, to be supported by all hosts. The ManagedFlag tells the host to proceed with DHCPv6 to get a centrally administered IP address.

10. IPv6 fragmentation

→ The fragmentation function in IPv4 is non-optimal for routers (primary function of routers is packet forwarding, not fragmenting packets).

→ With IPv6, only the transmitting node can fragment. Intermediate routers do not fragment. They are supposed to route packets as fast as possible. Fragmentation is not their job.

→ If an intermediate router receives a packet that would need fragmentation, it sends an ICMPv6 „Packet too big“ message back to the sender (similar to IPv4 „Fragmentation needed but DF set“).



MTU: Maximum Transfer Unit

11. IPv6 neighbor discovery (ND) protocol – RFC4861 (1/2)

11.1 Purpose:

Replacement for IPv4 ARP, ICMP router discovery and ICMP redirect messages and IPv4 DHCP.

11.2 IPv6 neighbor discovery RFC4861 functions (1):

Router Discovery (replaces IPv4 router discovery):

Location of routers that reside on an attached link.

Prefix Discovery:

Discovery of the set of address prefixes that define which destinations are on-link for an attached link.

Parameter Discovery:

Discovery of link parameters such as the link MTU or Internet parameters as the hop limit value to place in outgoing packets.

Address Autoconfiguration:

Automatic configuration of an address for an interface.

Address resolution:

Determination of the link-layer address of an on-link destination (e.g., a neighbor) given only the destination's IP address (replaces IPv4 ARP).

11. IPv6 neighbor discovery (ND) protocol – RFC4861 (2/2)

11.2 IPv6 neighbor discovery RFC4861 functions (2):

Next-hop determination:

Algorithm for mapping an IP destination address into the IP address of the neighbor to which traffic for the destination should be sent. The next-hop can be a router or the destination itself.

Neighbor Unreachability Detection:

Determination that a neighbor is no longer reachable. For neighbors used as routers, alternate default routers can be tried. For both routers and hosts, address resolution can be performed again.

Duplicate Address Detection DAD:

Determination that an address a node wishes to use is not already in use by another node.

Redirect (replaces IPv4 redirect messages):

Router informing a host of a better first-hop node to reach a particular destination.

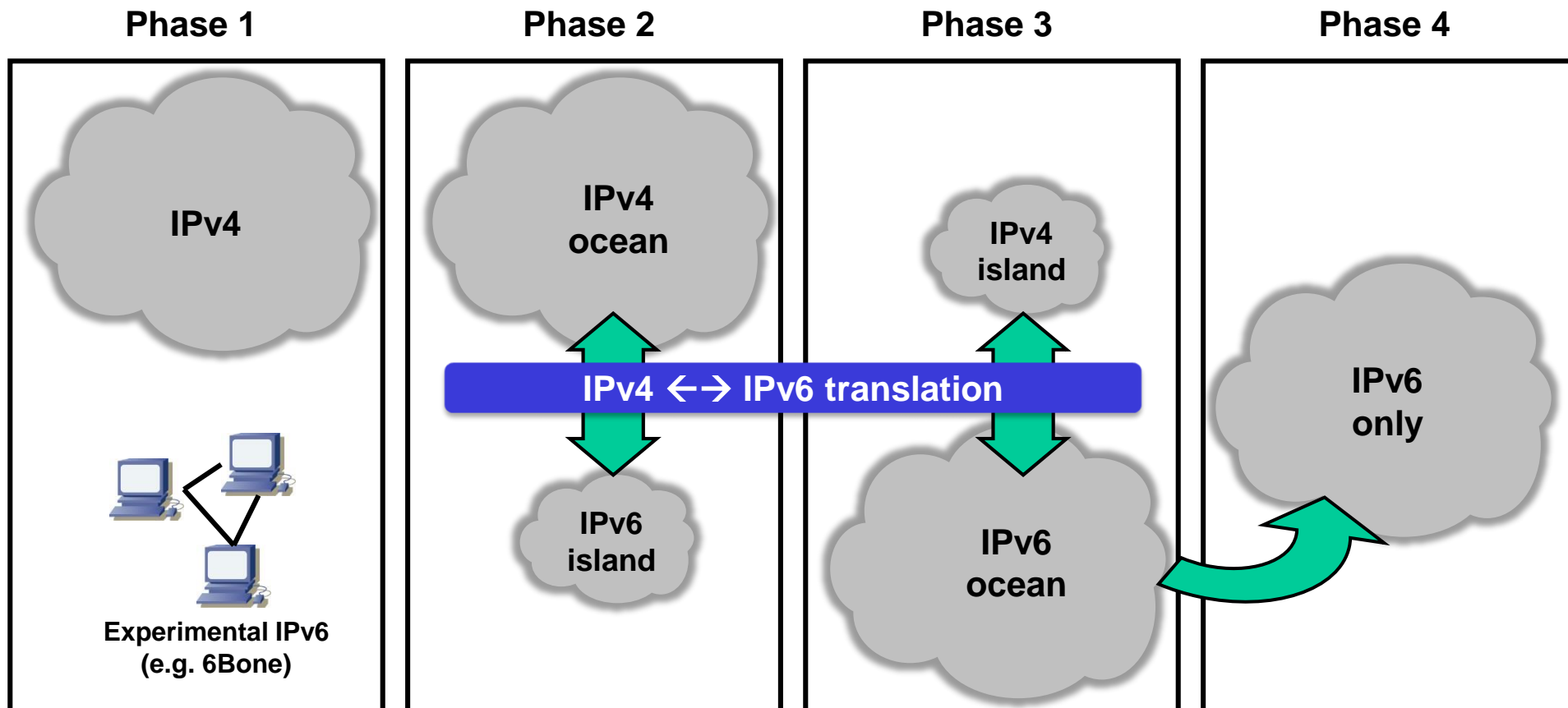
12. Migration steps for transition from IPv4 to IPv6 (1/78)

12.1 Migration strategy:

IPv6 was designed with migration in mind (no „D-day“ where everything is moved to IPv6 on the dot of twelve o'clock!).

Thus IPv4 and IPv6 will coexist for a long time to come, possibly forever!

There exist many different migration protocols for the different scenarios.



12. Migration steps for transition from IPv4 to IPv6 (2/78)

12.2 Node classification for transition (RFC4213):

1. IPv4-only node:

A host or router that implements only IPv4.

2. IPv6/IPv4 node:

A host or router that implements both IPv4 and IPv6.

3. IPv6-only node:

A host or router that implements IPv6, but does not implement IPv4.

4. IPv6 node:

Any host or router that implements IPv6. IPv6/IPv4 and IPv6-only nodes are both IPv6 nodes.

5. IPv4 node:

Any host or router that implements IPv4. IPv6/IPv4 and IPv4-only nodes are both IPv4 nodes.

N.B.:

The terms host and node are usually used synonymously. The term host denotes the physical machine that runs IPv4 and/or IPv6 while the term node is used to denote a logical entity that implements IPv4 and/or IPv6.

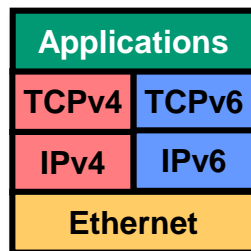
12. Migration steps for transition from IPv4 to IPv6 (3/78)

12.3 Migration / transition technology classification:

The transition technologies can be classified as dual-stack (A.), tunneling (B.) and translation (C.) as explained below.

A. Dual-stack:

A dual-stack node simply runs both an IPv4 and IPv6 stack. Depending on the application and DNS settings, such a node sends packets either over IPv4 or IPv6.



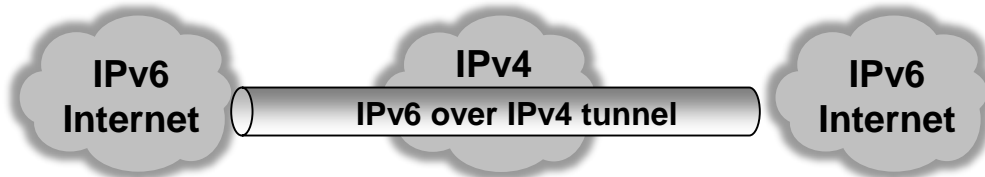
Options for dual-stack are:

- A.1. Simple IPv4 and IPv6 dual stack deployment
- A.2. VLAN based IPv4-IPv6 coexistence (RFC4554)

12. Migration steps for transition from IPv4 to IPv6 (4/78)

B. Tunneling (1/2):

Tunneling techniques connect IPv6 islands or hosts over IPv4 networks or vice versa. IP packets (IPv4 or IPv6) are encapsulated in another IP packet (IPv6 or IPv4) for transport.



Options for tunneling are:

B.1. Automatic tunneling

B.1.1. 6in4 ([RFC4213](#), basic transition mechanism)

B.1.2. 6over4 ([RFC2529](#), "Virtual Ethernet")

B.1.3. 6to4 ([RFC3056](#), connection of IPv6 domains via IPv4 clouds)

B.1.4. ISATAP ([RFC5214](#))

B.1.5. Teredo ([RFC4380](#))

B.1.6. IPv6 automatic tunneling ([RFC2893](#), *obsoleted by RFC4213*)

B.1.7. Tunnel broker ([RFC3053](#), IPv6 tunnel broker)

B.1.8. DSTM (IETF draft)

B.1.9. 6rd ([RFC5969](#))

B.1.10. Carrier Grade NAT (CGN)

B.1.11. Dual-Stack Lite

B.1.12. 6bed4 (IETF draft)

B.1.13. 4rd (IETF draft)

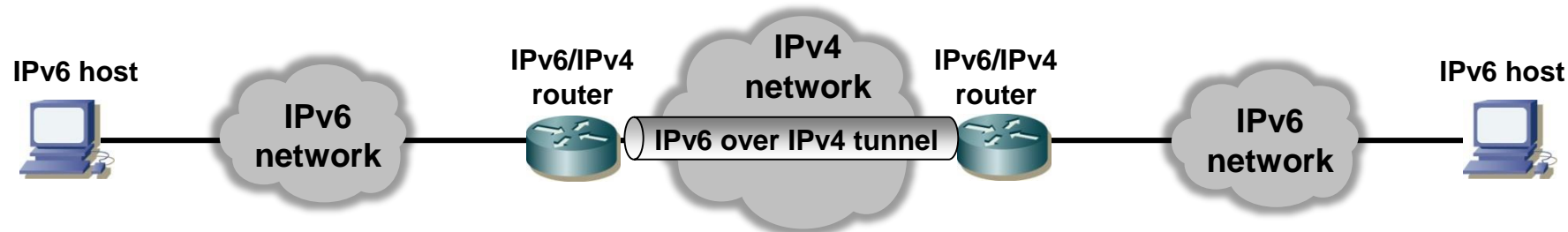
B.2. Configured tunneling (=explicit tunnel)

12. Migration steps for transition from IPv4 to IPv6 (5/78)

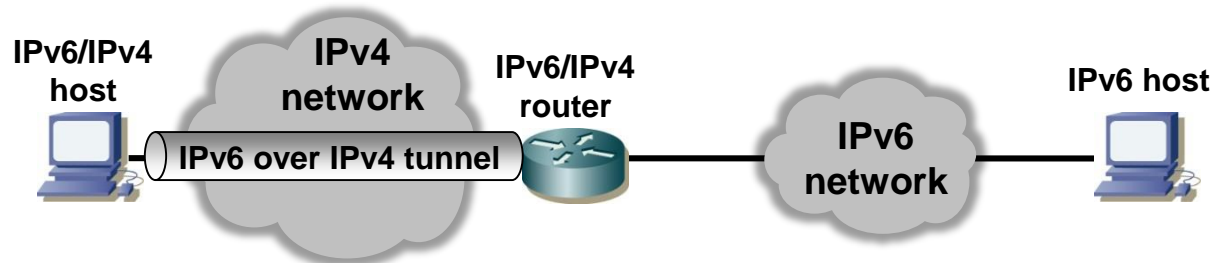
B. Tunneling (2/2):

Tunnel configurations are classified as follows (RFC4213):

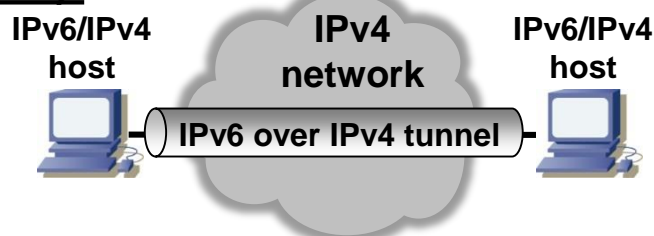
a. Router-to-Router (R2R, e.g. 6to4):



b. Host-to-Router or Router-to-Host (H2R, e.g. ISATAP):



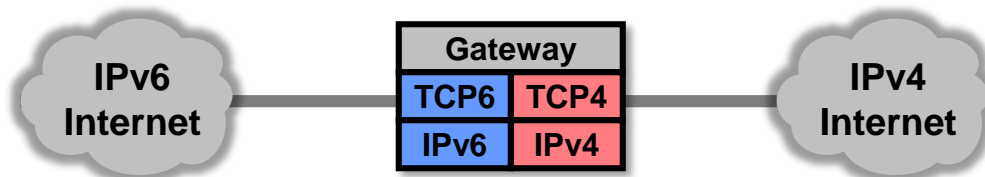
c. Host-to-Host (H2H, e.g. 6over4):



12. Migration steps for transition from IPv4 to IPv6 (6/78)

C. Translation:

Translation technologies connect IPv6 hosts or islands to IPv4 hosts or islands through a translating device, either running the translation on application level or directly in the network stack.



Options for translation are:

- C.1. NAT-PT (RFC2766, **obsoleted by RFC4966**)
- C.2. SIIT (RFC2765, Stateless IP/ICMP translation algorithm, **obsoleted by RFC6145**)
- C.3. BIS ([RFC2767](#), Bump In the Stack, RFC **obsoleted by [RFC6535](#)**)
- C.4. BIA ([RFC3338](#), Bump In the API, RFC **obsoleted by [RFC6535](#)**)
- C.5. BIH ([RFC6535](#), Bump In the Host)
- C.6. ALG
- C.7. SOCKS64 ([RFC3089](#))
- C.8. TRT ([RFC3142](#))
- C.9. Stateless and stateful NAT64 ([RFC6052](#), [RFC6145](#) thru [RFC6147](#))

12. Migration steps for transition from IPv4 to IPv6 (7/78)

12.4 Applicability of transition mechanisms (see also <http://ipv6int.net>):

Key:
H2H Host to host tunnel
H2R Host to router tunnel
R2R Router to router tunnel

	Cisco	MS	Linux	OS X	H2H/H2R/ R2R	Tunnel type	Comment
6in4 RFC4213	Yes	Yes	Yes (SUSE, RH)	Yes	H2H, H2R R2R	6 over 4	Proto-41 tunneling. Basic tunneling mechanism.
6over4 RFC2529	No	No	No	No	H2H H2R	6 over 4	Proto-41 tunneling. Not used much due to need for multicast.
6to4 RFC3056	Yes	Yes	Yes	No	R2R	6 over 4	Proto-41 tunneling. Standard way of v6 to v4 interworking.
ISATAP RFC5214	Yes	Yes	Yes	No	H2R	6 over 4	Proto-41 tunneling. Alternative for 6over4 when IPv4 multicast is not supported.
Teredo RFC4380	Yes	Yes	Yes (Miredo)	No	H2H H2R	6 over 4	Last resort technology when other tunneling mechanism can not be used.
Tunnel broker RFC3053	(No)	(No)	(No)	(No)	H2R	6 over 4	Automatic setup of tunnel. Does not define specific tunnel protocol.
DSTM	No	No	Yes (RH)	No	H2R	4 over 6	Expired IETF draft. IPv4 over IPv6 tunneling.
6rd (RFC5969)	Yes	No	Yes	No	R2R	6 over 4	Rapid deployment of IPv6 service over IPv4 service provider infrastructure.
CGN	Yes	No	Yes	No	R2R	4 over 6	Designed let ISPs offer IPv6-only service while customers retain their IPv4 setup.
6bed4 (IETF draft)	No	No	Yes	No	H2H, H2R	6 over 4	Simplified tunneling for embedded devices.
4rd (RFC7600)	No	No	No	No	R2R	4 over 6	Deployment of IPv4 service over IPv6 provider infrastructure.

12. Migration steps for transition from IPv4 to IPv6 (8/78)

12.4 Applicability of transition mechanisms (see also <http://ipv6int.net>):

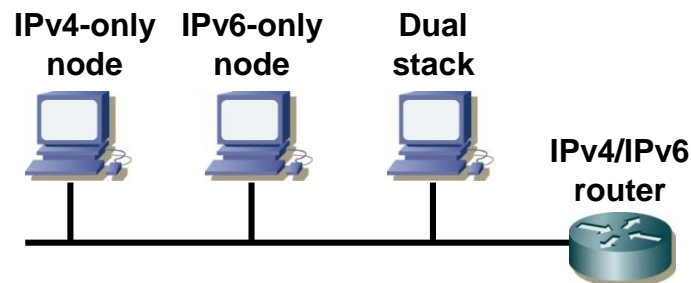
	Cisco	MS	Linux	OS X	Comment
SIIT RFC2765	Yes	No	No	No	Connect IPv6 applications over IPv4 infrastructure.
BIS RFC2767	No	No	No	No	Connect IPv4 applications over IPv6 infrastructure.
BIA RFC3338	No	No	No	No	Like BIS, but translation of API calls instead of packet headers.
ALG	No	Yes	Yes	Yes	Simple application level proxy. Inherently supported by OSes, but needs a proxy application to be developed.
SOCKS64 RFC1928 RFC3089	No	No	No	No	Connect IPv6 applications to IPv4-only servers. Hosts need to "talk" SOCKS protocol.
TRT RFC3142	No	No	No	No	Connect IPv6 applications to IPv4-only servers. No changes on IPv6 or IPv4 hosts necessary.
Stateless NAT64 RFC6145	Yes	Yes	Yes	No	Mechanism for statelessly mapping IPv6 to IPv4 addresses.
Stateful NAT64 RFC6146	Yes	Yes	Yes	No	Similar to stateless NAT64, but maintains session state in NAT tables.
DNS64 RFC6147	Yes	No	Yes	No	Method for synthesizing DNS AAAA records from A records. Usually works in conjunction with NAT64.
IPv4/IPv6 VLAN RFC4554	Yes	No	No	No	May be used in conjunction with other tunnel mechanisms. Used to separate IPv6 and IPv4 traffic on a LAN.
BIH RFC6535	(Yes)	No	(Yes)	No	Obsoletes and combines BIS and BIA.

12. Migration steps for transition from IPv4 to IPv6 (9/78)

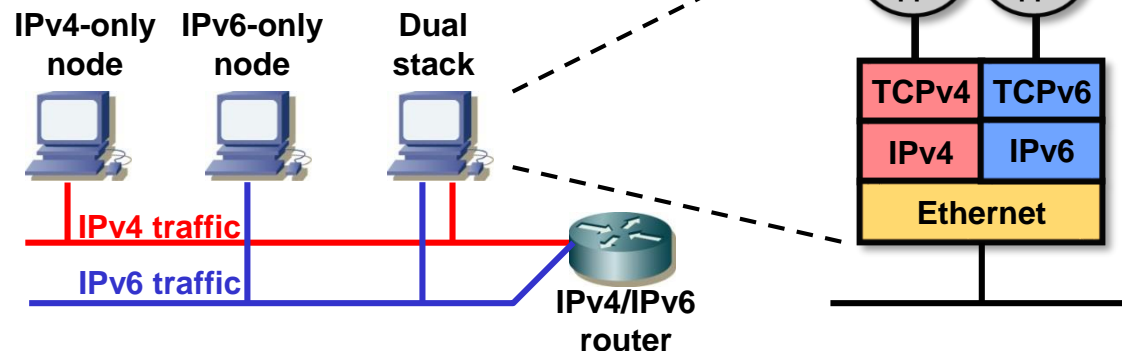
A.1. Simple IPv4 and IPv6 dual stack deployment:

Applications either use v4 or v6-sockets. Both IPv4 and IPv6 run on the same Ethernet.

Physical view:



Logical view:

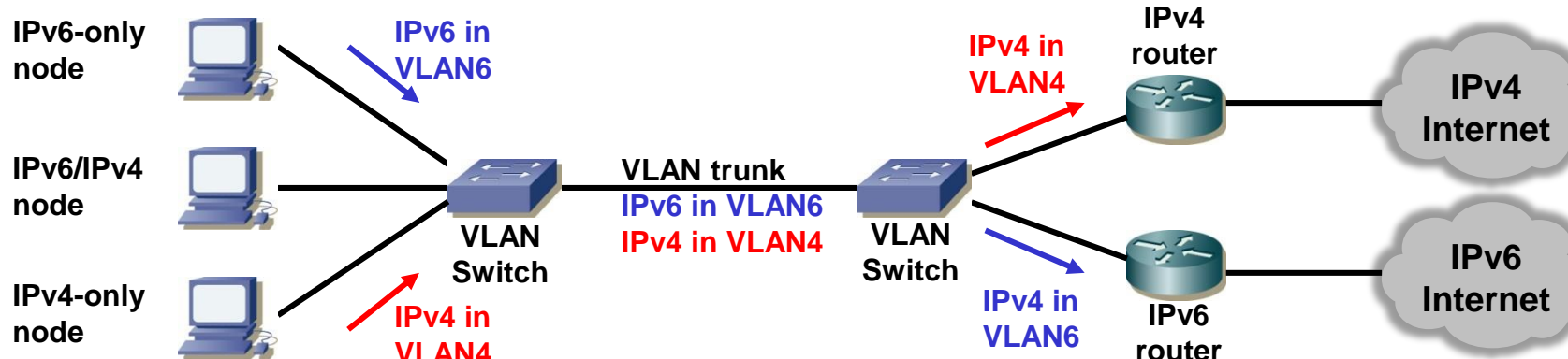


A.2. VLAN based IPv4-IPv6 coexistence (RFC4554):

Problem with approach above (1.1.): All routers in LAN must support both IPv4 and IPv6.

RFC4554 proposes to map all IPv6 traffic into a specific VLAN tag.

All switches in the network forward this VLAN traffic to a dedicated IPv6 router.

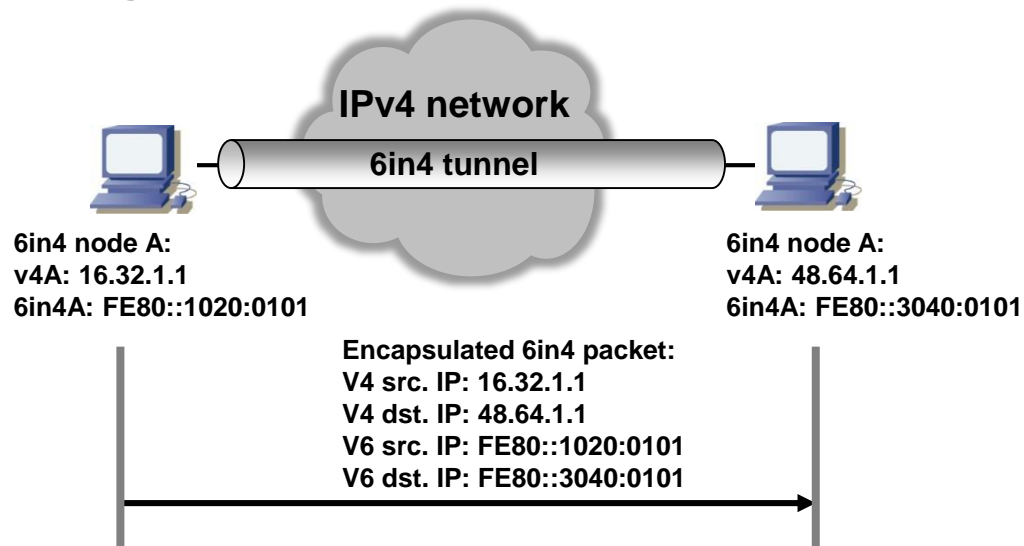


12. Migration steps for transition from IPv4 to IPv6 (10/78)

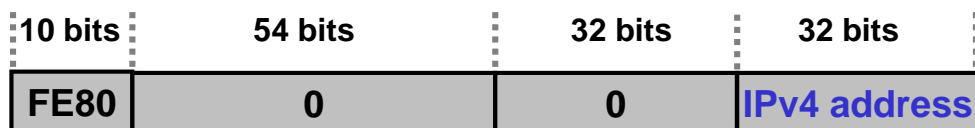
B.1.1. 6in4 tunneling (RFC4213):

6in4 defines a simple encapsulation mechanism of IPv6 packets in IPv4.

6in4 is very similar to 6over4, but does not require multicast. The tunnels are set up statically (sometimes 6in4 is called *proto-41 static* because it uses IPv4 protocol 41 (=IPv6 encapsulation) along with a static setup of tunnels).



Structure of 6in4 IPv6 address (same as in 6over4):



12. Migration steps for transition from IPv4 to IPv6 (11/78)

B.1.2. 6over4 tunneling (RFC2529, „virtual Ethernet“) (1/3):

6over4 may be used to connect isolated IPv6 nodes to an IPv6 network.

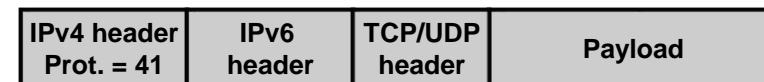
6over4 is a host-to-host and host-to-router tunneling mechanism.

6over4 uses IPv4 for the transmission of encapsulated IPv6 packet, thus it treats the IPv4 Internet as a giant Ethernet segment.

Every node needs a unique IPv4 address and IPv6 prefix.

6over4 uses unicast and multicast (for neighbor and router discovery).

6over4 uses simple protocol=41 encapsulation (IPv6 in IPv4):



Structure of 6over4 IPv6 address (same as in 6in4):

6over maps the IPv4 address to the least order bits of the IPv6 address.



Critique of 6over4:

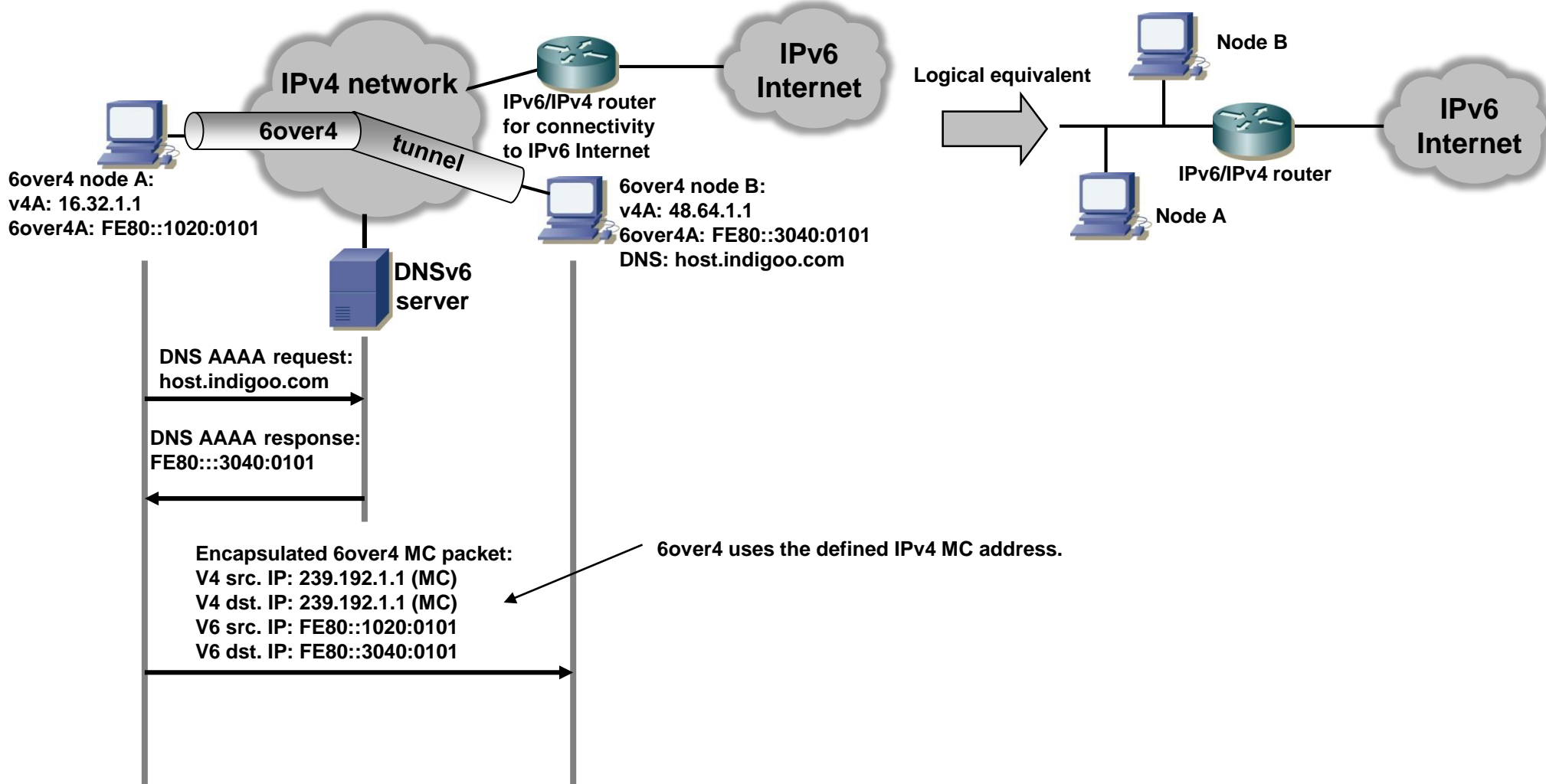
☹ 6over4 requires v4 multicast. Multicast is not widely available in IPv4, thus 6over4 is of limited use.

12. Migration steps for transition from IPv4 to IPv6 (12/78)

B.1.2. 6over4 tunneling (RFC2529, „virtual Ethernet“) (2/3):

6over4 makes use of IPv4 multicast to reach another node over an IPv4 network.

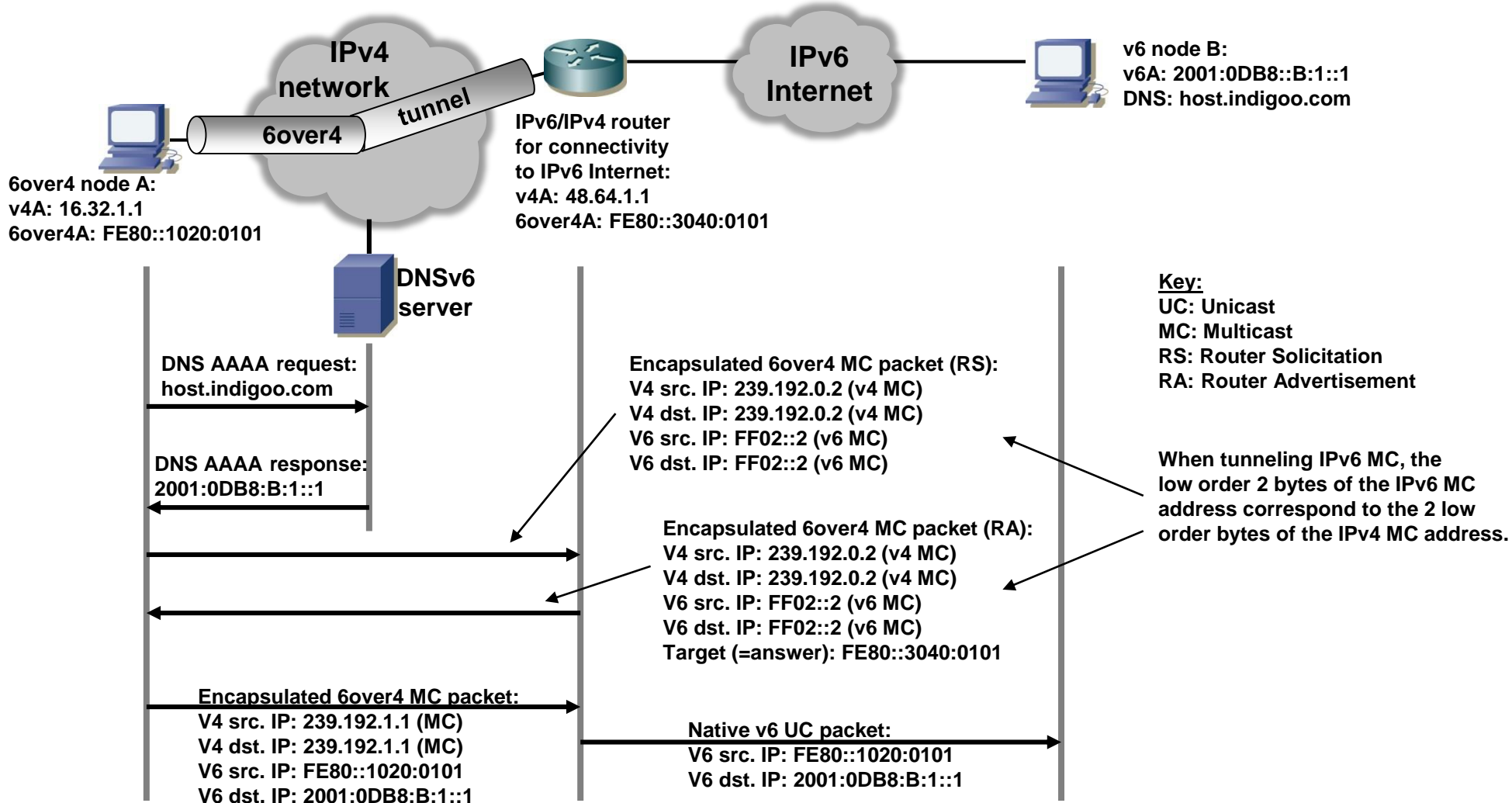
Key:
UC: Unicast
MC: Multicast



12. Migration steps for transition from IPv4 to IPv6 (13/78)

B.1.2. 6over4 tunneling (RFC2529, „virtual Ethernet“) (3/3):

6over4 supports / uses IPv6/IPv4 multicast for router and neighbor discovery.



12. Migration steps for transition from IPv4 to IPv6 (14/78)

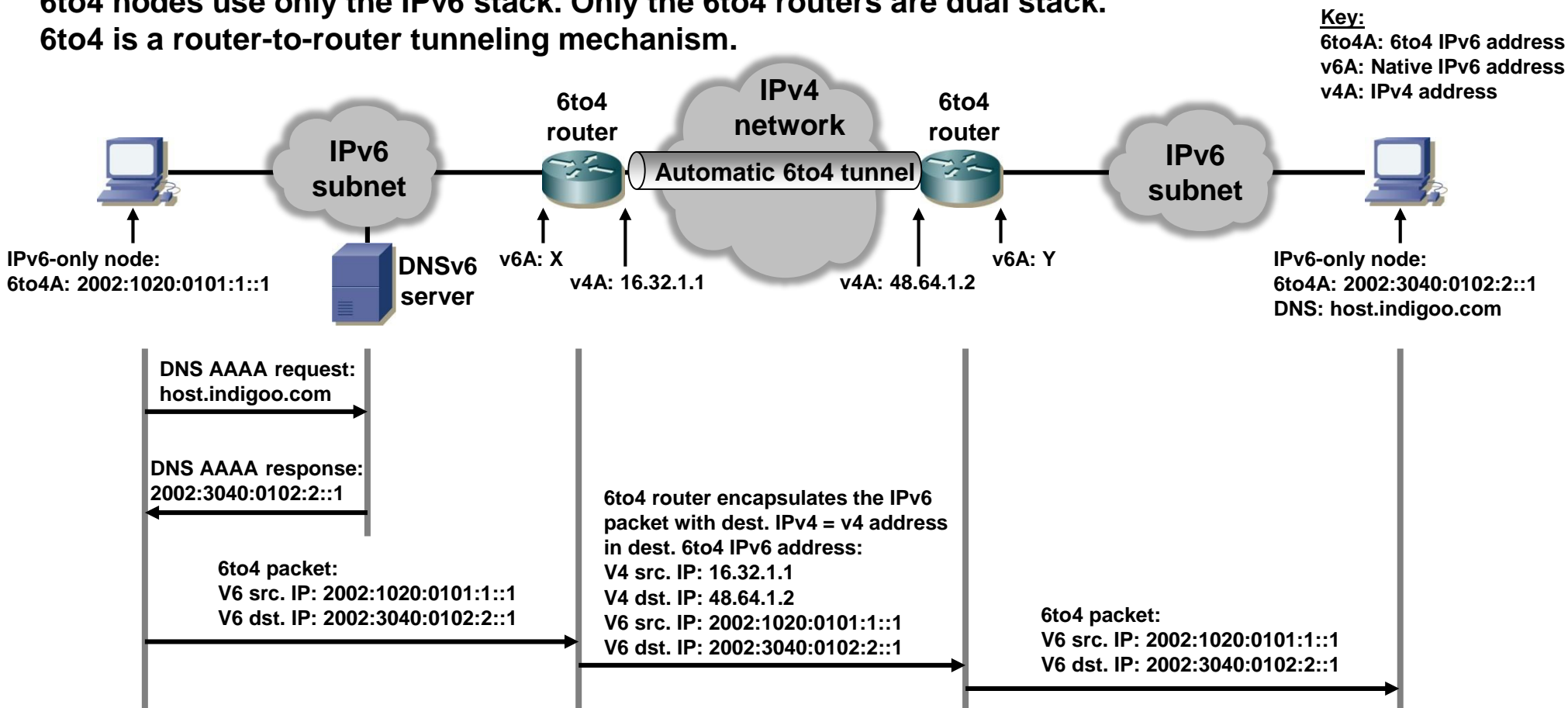
B.1.3. 6to4 tunneling (RFC3056) (1/3):

6to4 may be used to connect isolated IPv6 islands together or connect IPv6 islands with the IPv6 Internet / Intranet.

Every 6to4 node has a unique 6to4 address.

6to4 nodes use only the IPv6 stack. Only the 6to4 routers are dual stack.

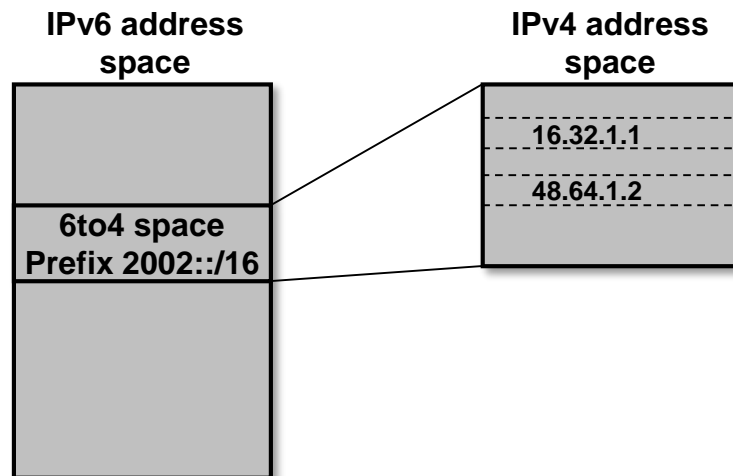
6to4 is a router-to-router tunneling mechanism.



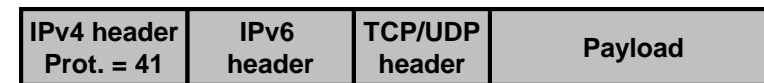
12. Migration steps for transition from IPv4 to IPv6 (15/78)

B.1.3. 6to4 tunneling (RFC3056) (2/3):

6to4 maps the IPv4 address space into the IPv6 space:



6to4 uses simple protocol=41 encapsulation (IPv6 in IPv4):

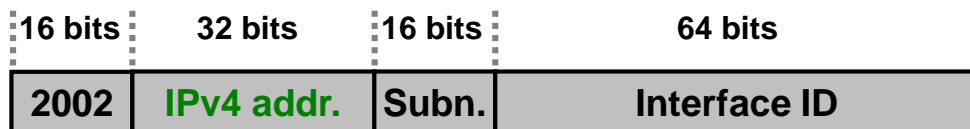


Structure of the 6to4 IPv6 address:

The IPv4 address used for tunneling the IPv6 packets is part of the IPv6 address.

The position of the IPv4 address in the IPv6 address allows prefix aggregation.

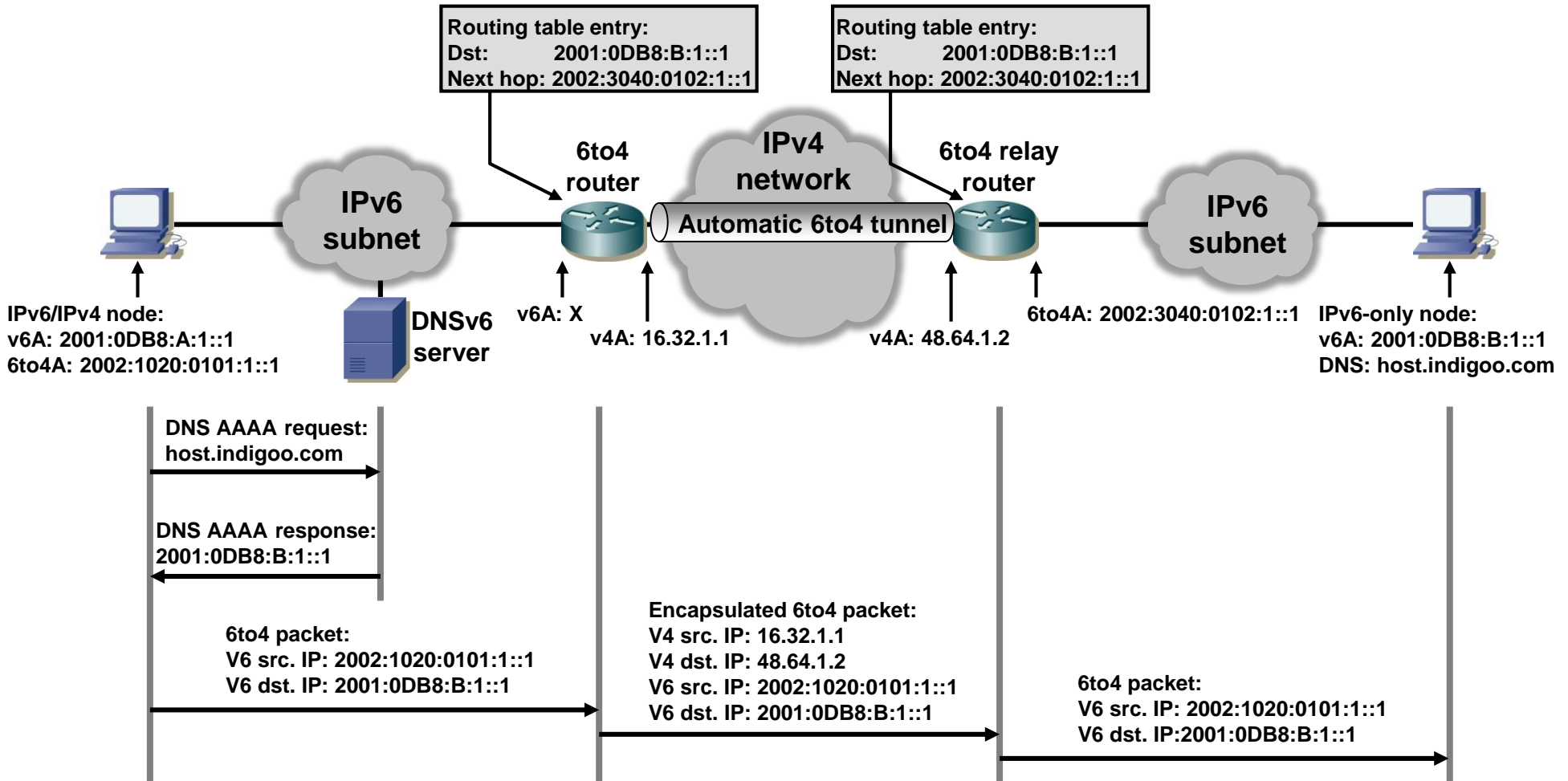
The prefix length without subnet is 48 bits.



12. Migration steps for transition from IPv4 to IPv6 (16/78)

B.1.3. 6to4 tunneling (RFC3056) (3/3):

A 6to4 relay router may be added to connect isolated 6to4 hosts to IPv6-only hosts (IPv6 Internet):



12. Migration steps for transition from IPv4 to IPv6 (17/78)

B.1.4. ISATAP – Intra-Site Automatic Tunneling Addressing Protocol (RFC5214) (1/4):

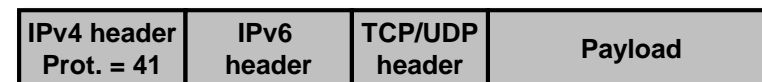
ISATAP works similar to 6over4 but does not require IPv4 multicast support.

Instead, ISATAP uses IPv4 as a non-broadcast multiple access (NBMA) link layer.

To compensate for the missing multicast, ISATAP-nodes use tables (PRL) with ISATAP-router interfaces that serve as ISATAP-tunnel endpoints.

When using global addresses (obtained through DNS + router solicitation) instead of link local addresses, ISATAP even allows to connect hosts with private IPv4 addresses to the IPv6 Internet.

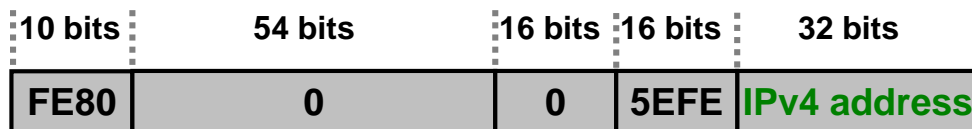
ISATAP uses simple protocol=41 encapsulation (IPv6 in IPv4):



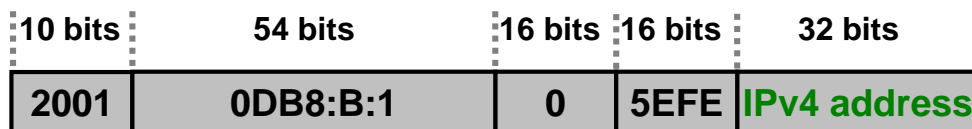
Structure of ISATAP IPv6 address:

ISATAP maps the IPv4 address to the least order bits and prefixes the IPv4 address with 0x5EFE.

ISATAP addresses may have link-local or global prefixes.



ISATAP address with link-local prefix



ISATAP address with global prefix

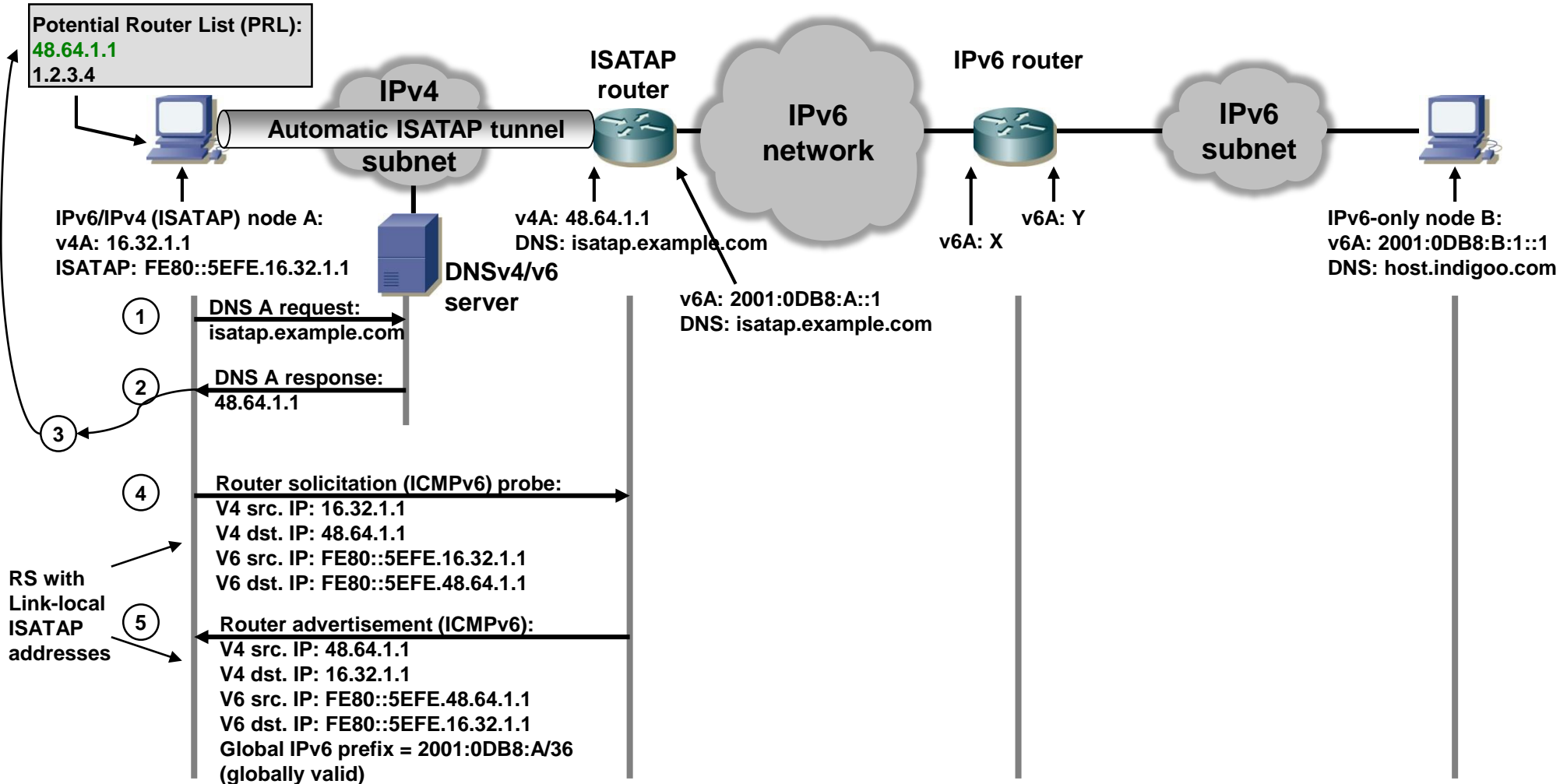
Critique of ISATAP:

☹ ISATAP requires several network resources to work in concert (DNS server, maybe DHCP server, ISATAP router). Configuring these consistently may not be easy.

12. Migration steps for transition from IPv4 to IPv6 (18/78)

B.1.4. ISATAP – Intra-Site Automatic Tunneling Addressing Protocol (RFC5214) (2/4):

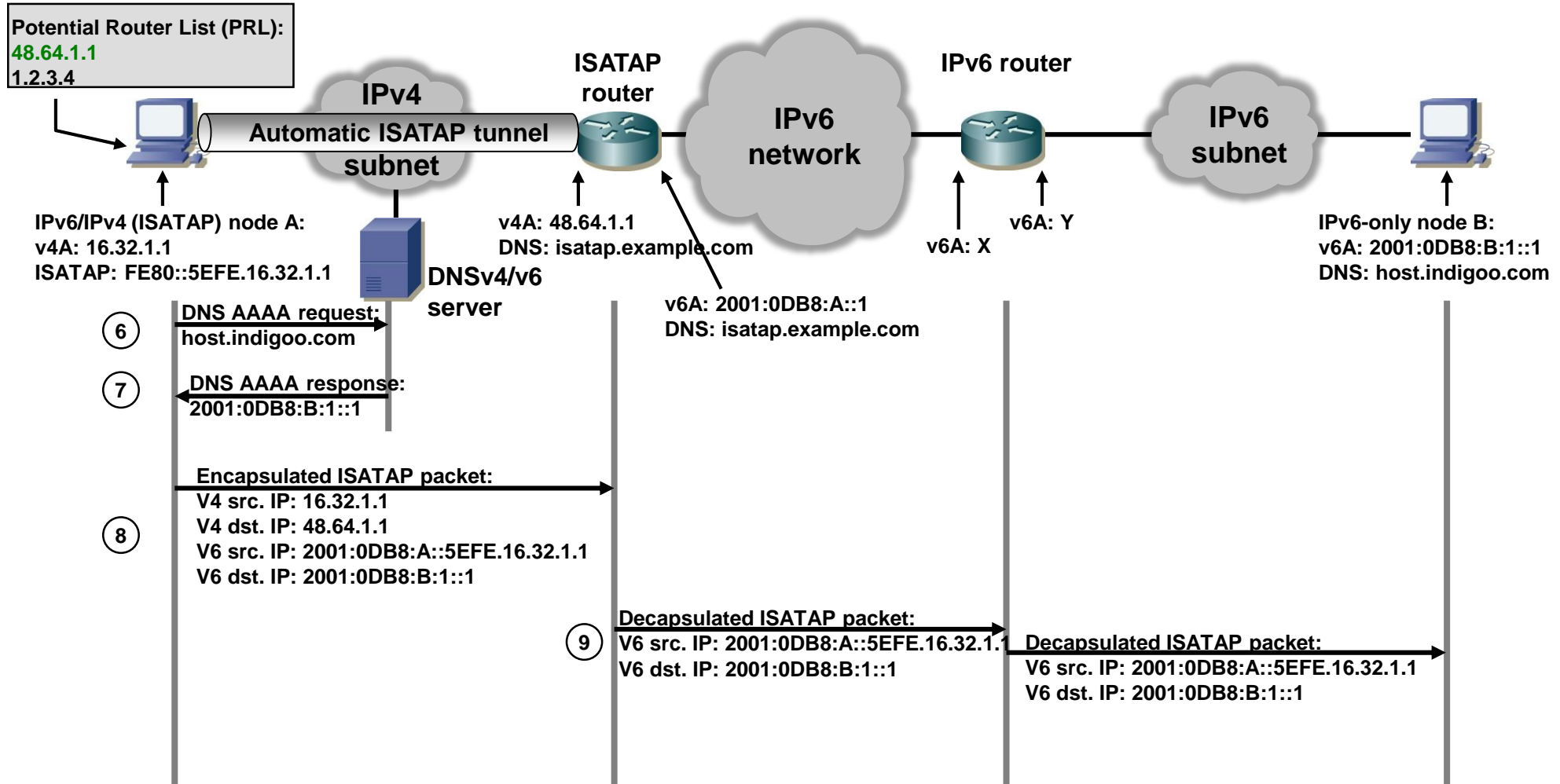
Example ISATAP scenario (1/2):



12. Migration steps for transition from IPv4 to IPv6 (19/78)

B.1.4. ISATAP – Intra-Site Automatic Tunneling Addressing Protocol (RFC5214) (3/4):

Example ISATAP scenario (2/2):



12. Migration steps for transition from IPv4 to IPv6 (20/78)

B.1.4. ISATAP – Intra-Site Automatic Tunneling Addressing Protocol (RFC5214) (4/4):

Step by step explanation of ISATAP interaction between ISATAP node and IPv6-only node:

1./2. ISATAP node ascertains ISATAP router:

The ISATAP node 'A' makes a normal IPv4 DNS query for isatap.example.com in order to find an ISATAP router. Instead of DNS (v4), the ISATAP node could use some other means such as DHCP options to find an ISATAP router. The IPv4 DNS server responds with the router's IPv4 address 48.64.1.1.

3. Add ISATAP router IPv4 address to PRL:

The ISATAP node 'A' adds the routers IPv4 address to its Potential Router List (PRL). This list contains the IPv4 address of available ISATAP router interfaces along with a time-to-live of this address (for redundancy reasons multiple ISATAP router interfaces may be available, so it is important that each ISATAP node know available and valid ISATAP interfaces).

4./5. Router solicitation to receive ISATAP support information:

The ISATAP node 'A' sends an ISATAP-encapsulated (link-local IPv6 addresses) router solicitation message (ICMPv6) to receive additional information, namely the global prefix to be used for the ISATAP addresses. The router responds with a router advertisement containing the global IPv6 prefix to be used for ISATAP addresses (needed so that the destination node 'B' can send back packets to the ISATAP node 'A').

6./7. DNS query for host.indigoo.com to obtain target IPv6 address:

The ISATAP node 'A' receives the IPv6 address of the node 'B' through a DNS query (DNSv4, one of the answers contains an AAAA entry).

8. ISATAP-encapsulation of packet:

Node 'A' encapsulates the IPv6 packet in an IPv4 packet (tunnel) using the router's IPv4 as destination address. The IPv6 source address is now 2001:0DB8:A::5EFE.16.32.1.1 so that the destination has a reachable IPv6 address where to send back packets.

9. Router decapsulates the ISATAP packet:

The ISATAP router decapsulates the packet (tunnel termination) and forwards it towards the destination using standard IPv6 routing.

12. Migration steps for transition from IPv4 to IPv6 (21/78)

B.1.5. Teredo (RFC4380) (1/6):

Teredo was developed mainly by Microsoft as tunneling a transition mechanism to pass through NATs. Simple 6in4 encapsulation as used in 6to4, 6over4, 6in4 or ISATAP makes it difficult or impossible to traverse NAT-firewalls.

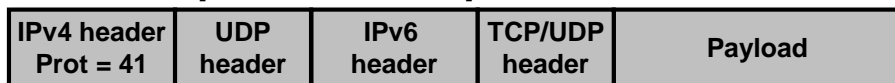
Teredo is a transition mechanism that will be replaced when more and more NATs support 6to4 tunneling (translate addresses also for proto=41 encapsulated packets).

Teredo requires an understanding of NAT-types as defined in RFC3489 (STUN).

Before communication with a peer starts, a Teredo client must determine the type of NAT it is behind (qualification procedure).

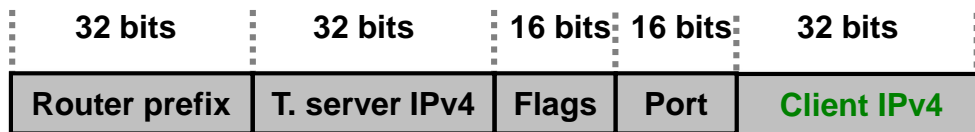
Teredo is a host to host (H2H) or host to router (H2R) tunneling protocol (using a Teredo relay).

Teredo encapsulates IPv6 packets in an additional UDP header for NAT-traversal:



Structure of Teredo address:

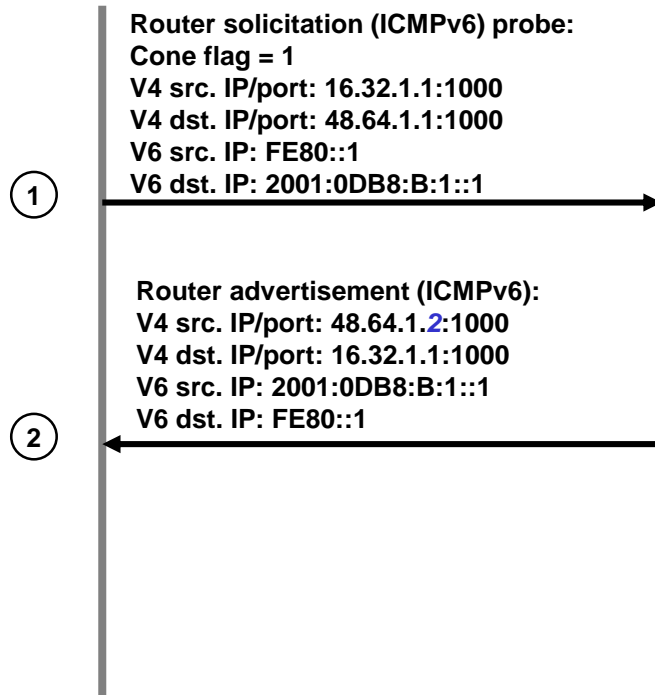
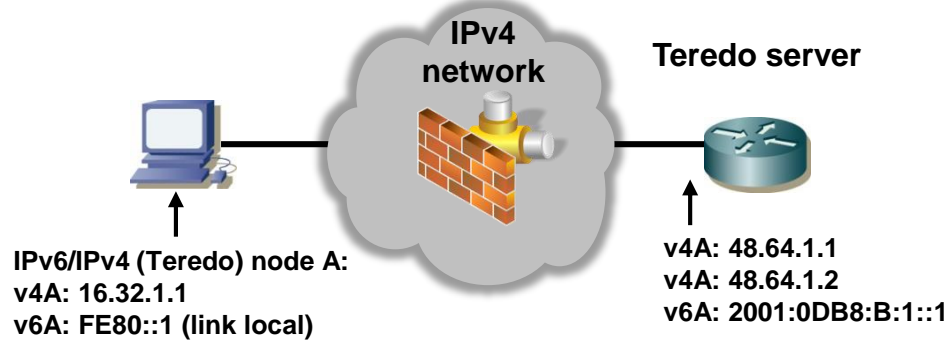
Teredo addresses are constructed from IPv4 addresses. They may be registered with DNS.



12. Migration steps for transition from IPv4 to IPv6 (22/78)

B.1.5. Teredo (RFC4380) (2/6):

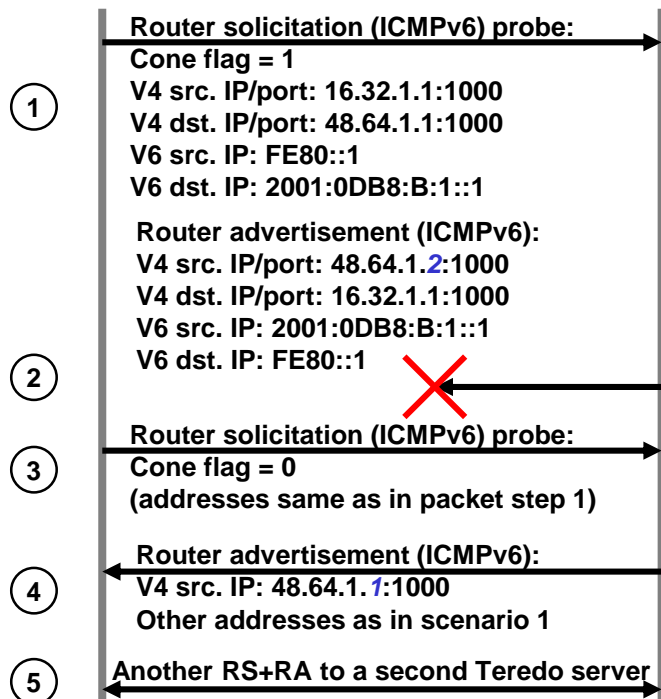
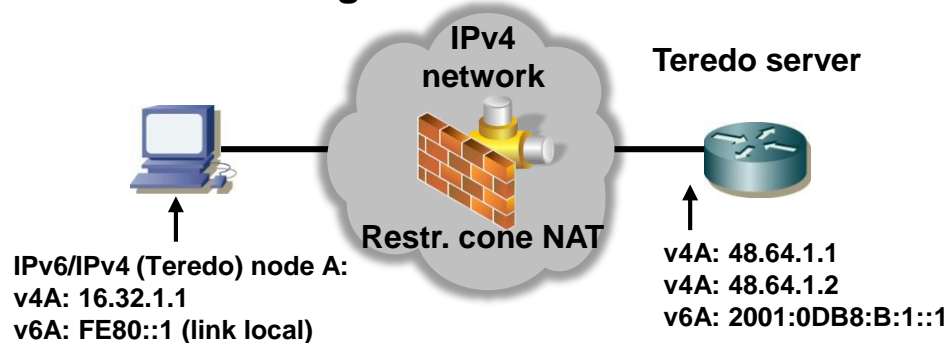
Initial client configuration scenario 1: Client behind cone NAT



12. Migration steps for transition from IPv4 to IPv6 (23/78)

B.1.5. Teredo (RFC4380) (3/6):

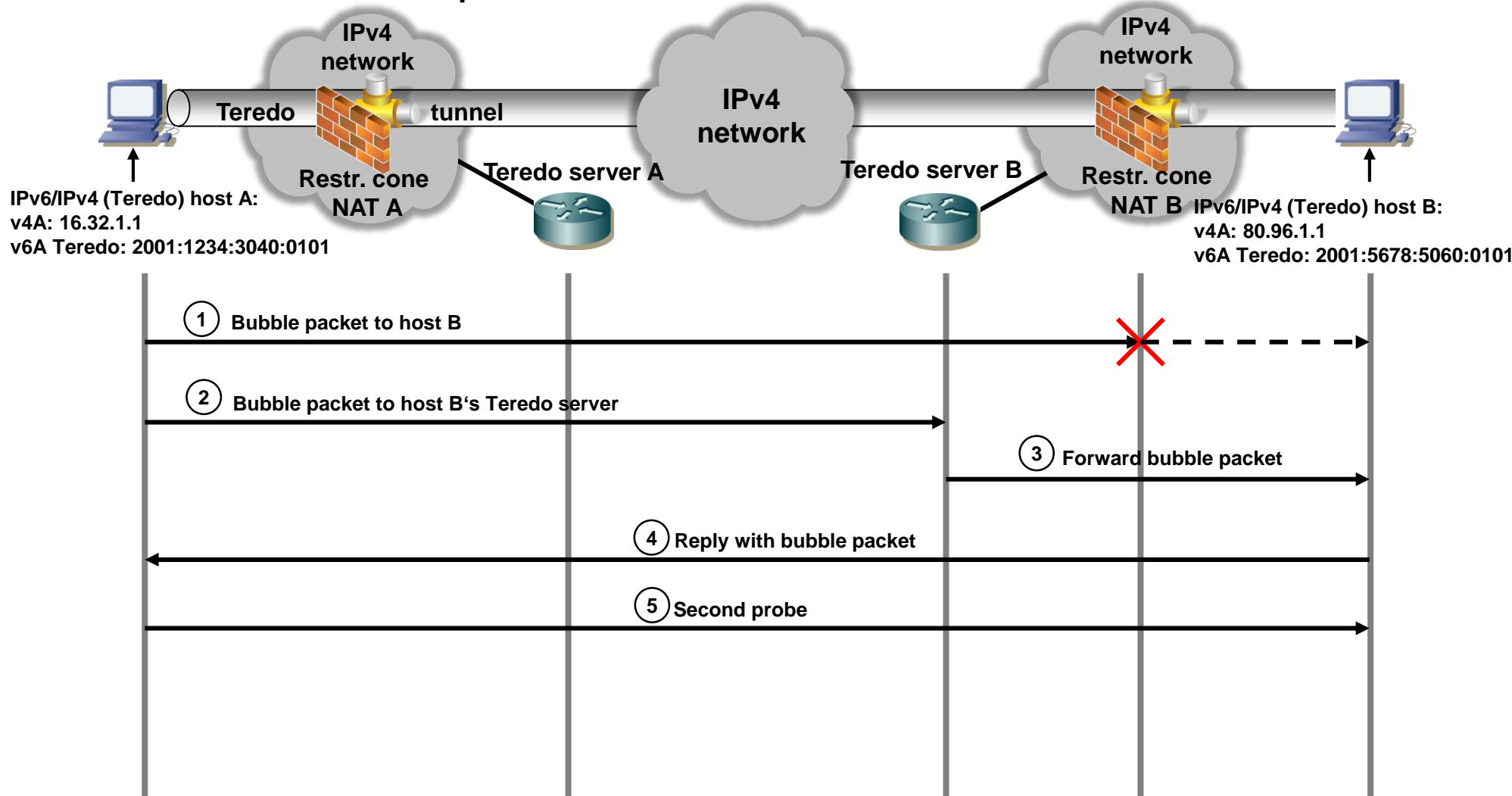
Initial client configuration scenario 2: Client behind restricted cone NAT



12. Migration steps for transition from IPv4 to IPv6 (24/78)

B.1.5. Teredo (RFC4380) (4/6):

Punch hole in NAT with bubble packets scenario 1: Restricted NAT



12. Migration steps for transition from IPv4 to IPv6 (25/78)

B.1.5. Teredo (RFC4380) (5/6):

Step by step explanation of initial client configuration of a Teredo session:

Scenario 1: Cone NAT

1. Client sends an RS probe with cone flag=1:

The client sends an ICMPv6 router solicitation (RS) message to the Teredo server. The cone flag in the probe is set to 1.

2. Teredo server RA response:

The Teredo server responds with a router advertisement message (RA). Because the cone flag in the RS message was set to 1, the server uses a different IPv4 address as source address (48.64.1.2 instead of 48.64.1.1).

If the client receives the RA message it knows that it is behind a cone NAT (different destination addresses use the same mapped address). The client now constructs its Teredo IPv6 address (structure see above).

Scenario 2: Restricted cone NAT

1. Client sends an RS probe with cone flag=1:

As in scenario 1 the client sends an RS probe packet.

2. Teredo server RA response:

As in scenario 1 the server responds with an RA packet. The restricted cone NAT, however, blocks the packet.

3. Client sends RS probe with cone flag=0:

Because the client has not received the RA packet it, re-sends the RS probe, but sets the cone flag to 0.

4. Teredo server RA response:

Because the cone flag in the probe packet was set to 0, the server sends the RA packet from the IPv4 address on which it received the RS probe packet (48.64.1.1).

5. Additional RS+RA to a different Teredo server:

The client sends an RS probe to the second Teredo server to check if it is behind a symmetric NAT.

If the client determines that it is behind a symmetric NAT communication stops. The client constructs its Teredo IPv6 address.

12. Migration steps for transition from IPv4 to IPv6 (26/78)

B.1.5. Teredo (RFC4380) (6/6):

Step by step explanation of punching holes into NATs:

1. Host A sends bubble packet:

Host A sends a bubble packet directly to host B. Host A's NAT will add an address mapping into its NAT table to allow packets from any outside host addressed to A's IPv4 address and port number to pass.

Host B's NAT blocks the bubble packet because there is no mapping in its table.

2. Host A sends a bubble packet to host B's Teredo server:

Host A determines host B's Teredo server (see address structure above) and sends a bubble packet to it.

3. Host B's Teredo server forwards bubble packet:

Host B's Teredo server determines that the packet is a Teredo packet and forwards it to host B. Host B's NAT lets the packet pass because it contains an address mapping for packets from Teredo server B (from the qualification procedure at the beginning).

4. Host B sends bubble packet to host A:

Host B sends a bubble packet back directly to host A. This adds a NAT entry for packets from host A in host B's NAT.

5. Tunneled application packet:

Host A now sends an application IPv6 packet encapsulated in an IPv4 packet to host B.

12. Migration steps for transition from IPv4 to IPv6 (27/78)

B.1.7. Tunnel broker / tunnel server (RFC3053) (1/3):

With dynamic (on-demand) tunnel creation no, configuration on the client is required.

Tunnel setup is similar to setting up a VPN connection (tunnel broker+server = VPN server).

As such a tunnel broker together with the tunnel server acts like a virtual IPv6 ISP.

Tunnel broker is not a protocol but a general architecture for connecting dual stack hosts to an IPv6 network.

The tunnel broker model can be used e.g. with 6to4 to automatically setup tunnels.

Tunnel broker is best suited to connect isolated nodes to an IPv6 network.

The main tunnel broker functions are:

- 1. Access control (e.g. through RADIUS)**
- 2. Register client DNS name in the IPv6 DNS space**
- 3. Assign one or multiple IPv6 prefixes to the client (default: 48 prefix)**

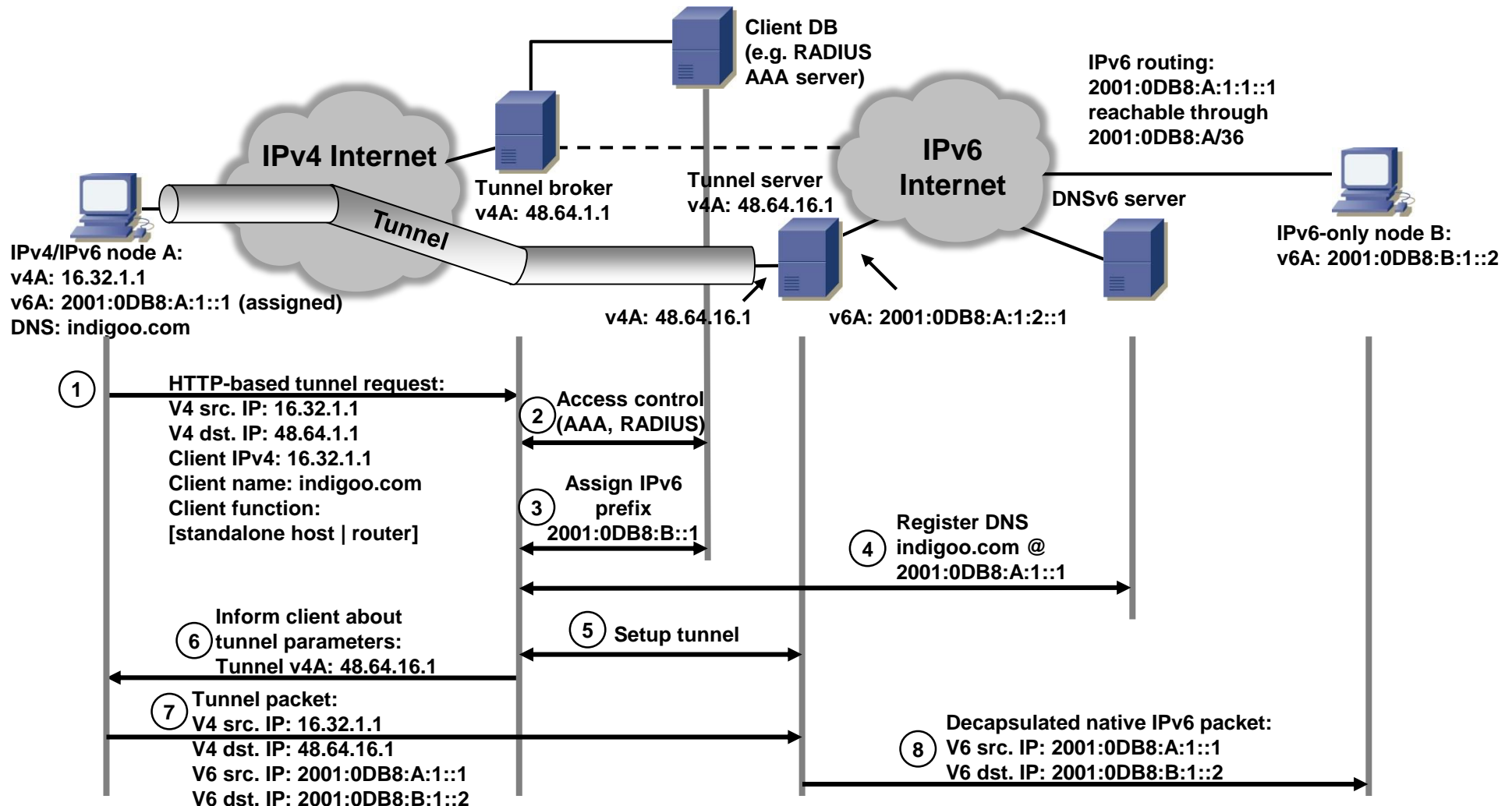
There exist commercial tunnel brokers.

List of tunnel brokers see http://en.wikipedia.org/wiki/List_of_IPv6_tunnel_brokers.

12. Migration steps for transition from IPv4 to IPv6 (28/78)

B.1.7. Tunnel broker / tunnel server (RFC3053) (2/3):

Tunnel brokers create tunnels on demand (act like an IPv6 Network Access Server, NAS).



12. Migration steps for transition from IPv4 to IPv6 (29/78)

B.1.7. Tunnel broker / tunnel server (RFC3053) (3/3):

Step by step explanation of a tunnel broker session:

1. Clients request:

The client sends a request to the tunnel broker (TB) to setup a tunnel. It is recommended to use HTTP as underlying protocol.

2. Access control:

The tunnel server may perform some access control functions such as authentication, authorization and possibly accounting through a protocol like RADIUS. This function is particularly interesting for ISPs to control who accesses their network.

3. IPv6 address allocation:

Based on the information given by the client (role: single node or router), the TB assigns and reserves an IPv6 address (range).

4. Client DNS name registration:

The TB registers the client's DNS name under the assigned IPv6 address in the global DNSv6 space.

5. Tunnel setup:

The TB sets up the tunnel on the tunnel server.

6. Tunnel parameters to client:

The TB informs the client about the tunnel parameters.

7. User packet sent by client:

The client application sends an IPv6 packet to the destination. The tunnel function in the client encapsulates the packet in an IPv4 packet.

8. Decapsulation + forward:

The tunnel server decapsulates the tunnel packet and forwards it to the next hop in the IPv6 network. The packet is forwarded based on standard IPv6 routing.

12. Migration steps for transition from IPv4 to IPv6 (30/78)

B.1.8. DSTM – Dual Stack Transition Mechanism (Internet draft) (1/3):

DSTM is intended for being used when IPv4 and IPv6 are in balance (communication between existing IPv4 and IPv6 hosts).

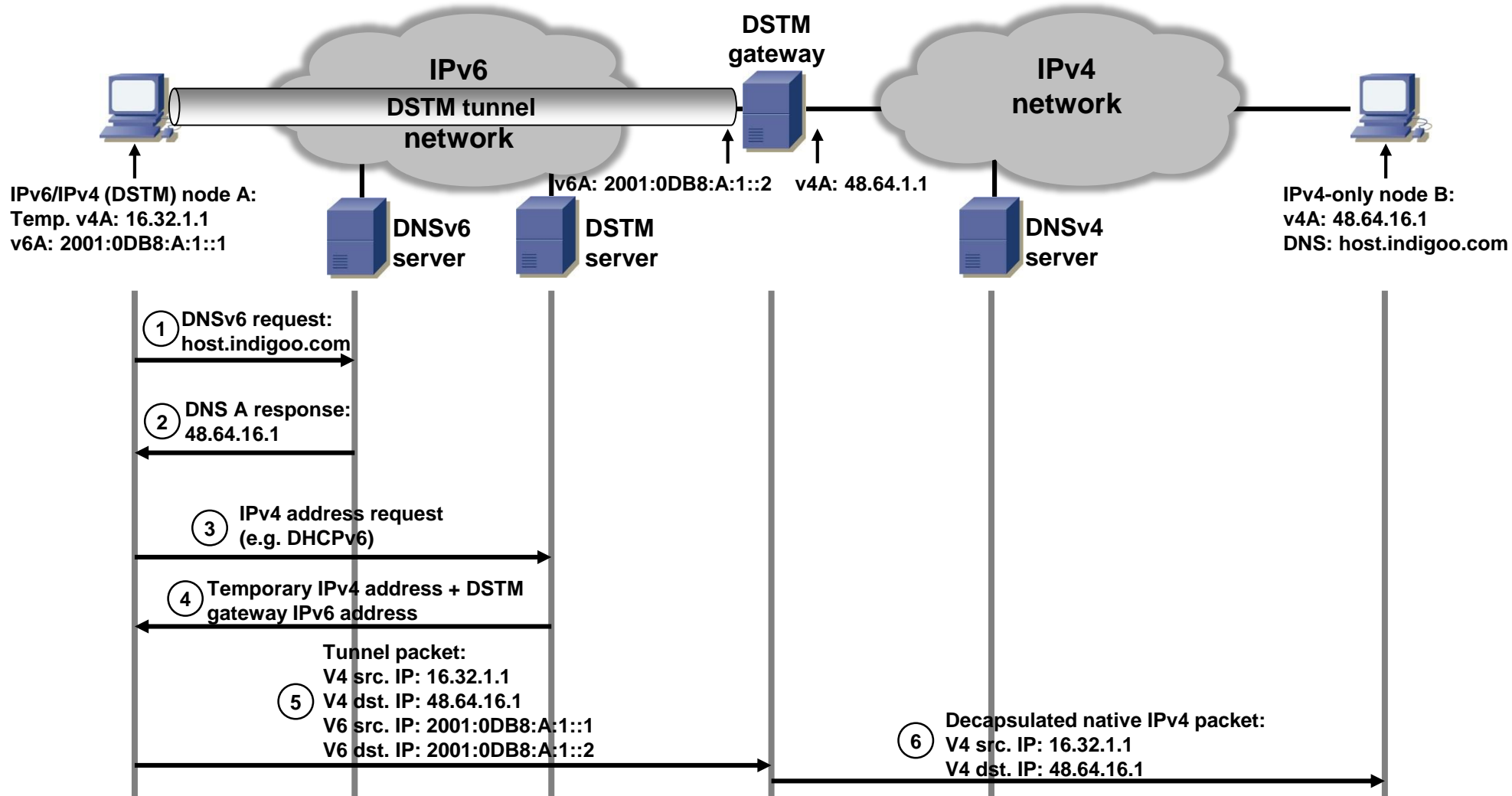
DSTM is very similar to the tunnel broker transition mechanism. Unlike tunnel broker, DSTM tunnels IPv4 packets over an IPv6 network (tunnel broker: IPv6 tunneled over IPv4).

DSTM is a component of the OS of a host and intercepts and tunnels packets as per the DSTM protocol. IPv6 applications are unaware of the presence of DSTM and work just like normal IPv6 applications using v6 sockets.

12. Migration steps for transition from IPv4 to IPv6 (31/78)

B.1.8. DSTM – Dual Stack Transition Mechanism (Internet draft) (2/3):

DSTM scenario 1: IPv6 → IPv4



12. Migration steps for transition from IPv4 to IPv6 (32/78)

B.1.8. DSTM – Dual Stack Transition Mechanism (Internet draft) (3/3):

Step by step explanation of DSTM scenario IPv6→IPv4:

1. DNSv6 request:

The DSTM component on host A intercepts a request by the DNS resolver. DSTM translates the request for host.indigoo.com into an A and AAAA DNS request for host.indigoo.com (v6 request).

2. DNS response:

As host B is an IPv4-only node, the DNS server has only an A record for host B and returns this to host A.

3.+4. Host A obtains temporary IPv4 address:

As DSTM only receives an A record, it contacts the DSTM server to obtain a temporary IPv4 address. This step may use existing protocols like DHCPv6. Along with the temporary IPv4 address DSTM also obtains the IPv6 address of the DSTM gateway.

5. Tunneling the application packet to the DSTM gateway:

The application sends a packet to the IPv4 host B. DSTM intercepts the packet, encapsulates it into an IPv6 packet and forwards it to the DSTM gateway.

6. Packet decapsulation:

The DSTM gateway decapsulates the packet and forwards it to the IPv4 destination host B.

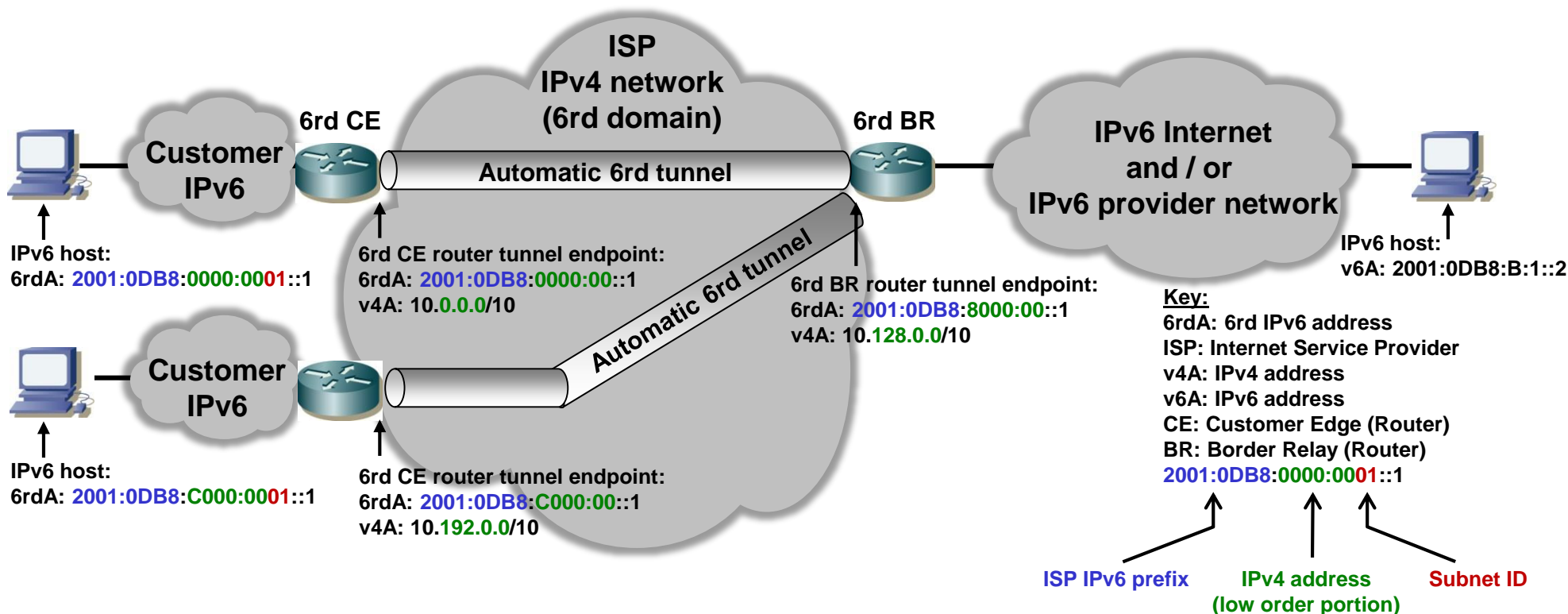
12. Migration steps for transition from IPv4 to IPv6 (33/78)

B.1.9. 6rd (RFC5969) (1/2):

6rd (IPv6 rapid deployment) is an extension or improvement of 6to4.

The key difference to 6to4 is that 6rd does not use 2002::/16 address prefixes but IPv6 addresses out of the ISP's IPv6 address space. Therefore 6rd service appears to the customer as a native IPv6 service.

In contrast to 6to4 where hosts may not be reachable from the IPv6 Internet, 6rd hosts are fully reachable because 6rd uses real IPv6 prefixes assigned to the ISP.



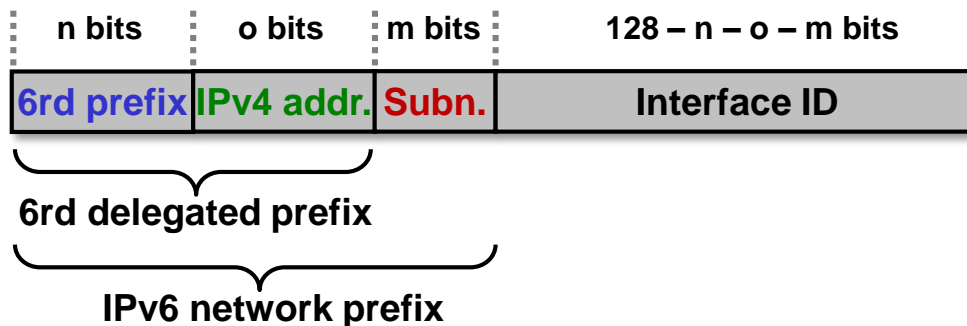
12. Migration steps for transition from IPv4 to IPv6 (34/78)

B.1.9. 6rd (RFC5969) (2/2):

Structure of the 6to4 IPv6 address:

6rd uses an ISP IPv6 prefix (e.g. **2001:0DB8**) plus the full IPv4 address assigned to the customer as 6rd prefix.

Within a 6rd domain (part of provider IPv4 network where one single IPv6 prefix is used for 6rd), multiple IPv4 addresses can be aggregated. In this case, only a portion of the IPv4 address with the relevant low order address bits are used by the CE router to automatically create a 6rd address.



Example:

6rd prefix:

2001:0DB8/16

IPv4 addresses in 6rd domain: **10.192.0.0/10** (hex notation: **0A.C0.00.00**)

6rd delegated prefix: **2001:0DB8:C000:00/56**

IPv6 network prefix: **2001:0DB8:C000:0001/64**

In case all IPv4 addresses can be aggregated to 10.0.0.0/8, only the low order 24 bits of the IPv4 address are used by the CE router to create 6rd delegated prefixes. This frees some bits for use as subnet ID.

12. Migration steps for transition from IPv4 to IPv6 (35/78)

B.1.10. Carrier Grade NAT (CGN) (1/3):

Carrier Grade NAT (also called Large Scale NAT - LSN) is technically the same as customer NAT. In CGN, the NAT function is moved to the provider (ISP) network.

The ISP provides Internet service based on RFC1918 private IPv4 addresses, i.e. IPv4 addresses out of the ranges 10.0.0./8, 172.16.0.0/12 and 192.168.0.0/16 as defined in RFC1918.

As such CGN is not a transition technology. Instead, it simply extends the provider's usable IPv4 address range.

Critique of CGN:

CGN is a simple mechanism to extend the lifetime of IPv4 addresses.

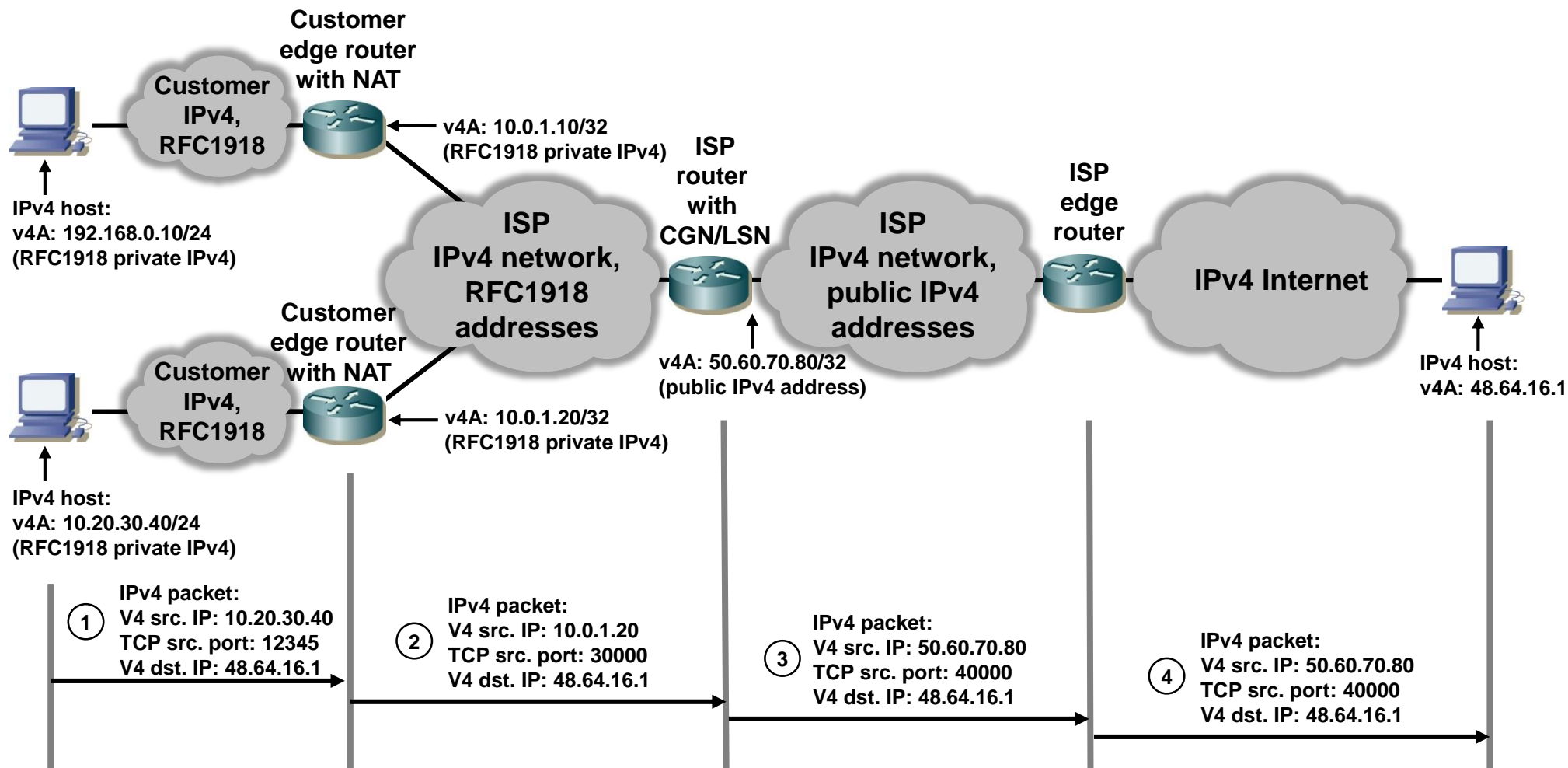
However, CGN has some serious drawbacks:

- ☹️ CGN uses NAT, a technique that was meant to be obsoleted by IPv6.
- ☹️ CGN is stateful, i.e. mapping tables need to be maintained in the CGN router, potentially leading to scalability problems.
- ☹️ CGN breaks the end-to-end principle (application specific functionality moved to the network rather than end-systems).
- ☹️ Customer networks are unreachable from the Internet, i.e. only outbound connections from customer network hosts to the Internet are possible.

12. Migration steps for transition from IPv4 to IPv6 (36/78)

B.1.10. Carrier Grade NAT (CGN) (2/3):

Scenario with customer NAT and CGN (NAT444):



12. Migration steps for transition from IPv4 to IPv6 (37/78)

B.1.10. Carrier Grade NAT (CGN) (3/3):

Step by step explanation of packets passing through the different IPv4 networks:

In this scenarios, customer IPv4 packets pass through 3 different IPv4 domains, thus the term NAT444.

1. Customer edge NAT:

A first NAT44 (mapping from IPv4 to another IPv4 address) function is run by a customer edge router.

The NAT router exchanges the source IPv4 address 10.20.30.40 by the ISPs provider IPv4 address 10.0.1.20. As part of the NAT mapping function, the TCP source port number is mapped as well (12345→30000) so that packets in reverse direction find their way back to the source.

2. Second NAT in provider network:

A second NAT44 function sits at the boundary between the provider's private and public IPv4 networks and maps from private to public IPv4 addresses.

Source IP address and TCP port number are mapped as 10.0.1.20→50.60.70.80 and 30000→40000.

3. Packet forwarded through IPS's public IPv4 network:

The packet is forwarded through the ISP's public IPv4 network towards the Internet edge router.

4. Packet forwarded through public Internet:

Finally, the packet finds its way through the public IPv4 Internet and reaches the end destination.

12. Migration steps for transition from IPv4 to IPv6 (38/78)

B.1.11. Dual-Stack Lite (DS-Lite) (1/3):

DS-Lite is similar to DSTM, but in DS-Lite the customer runs IPv4 with private RFC1918 IP addresses.

The customer edge router tunnels IPv4 packets over the ISP's IPv6 network to a CGN device that serves as tunnel endpoint and NAT device.

DS-Lite allows the ISP to hook up IPv4 customers through IPv6 only provider networks with the following advantages:

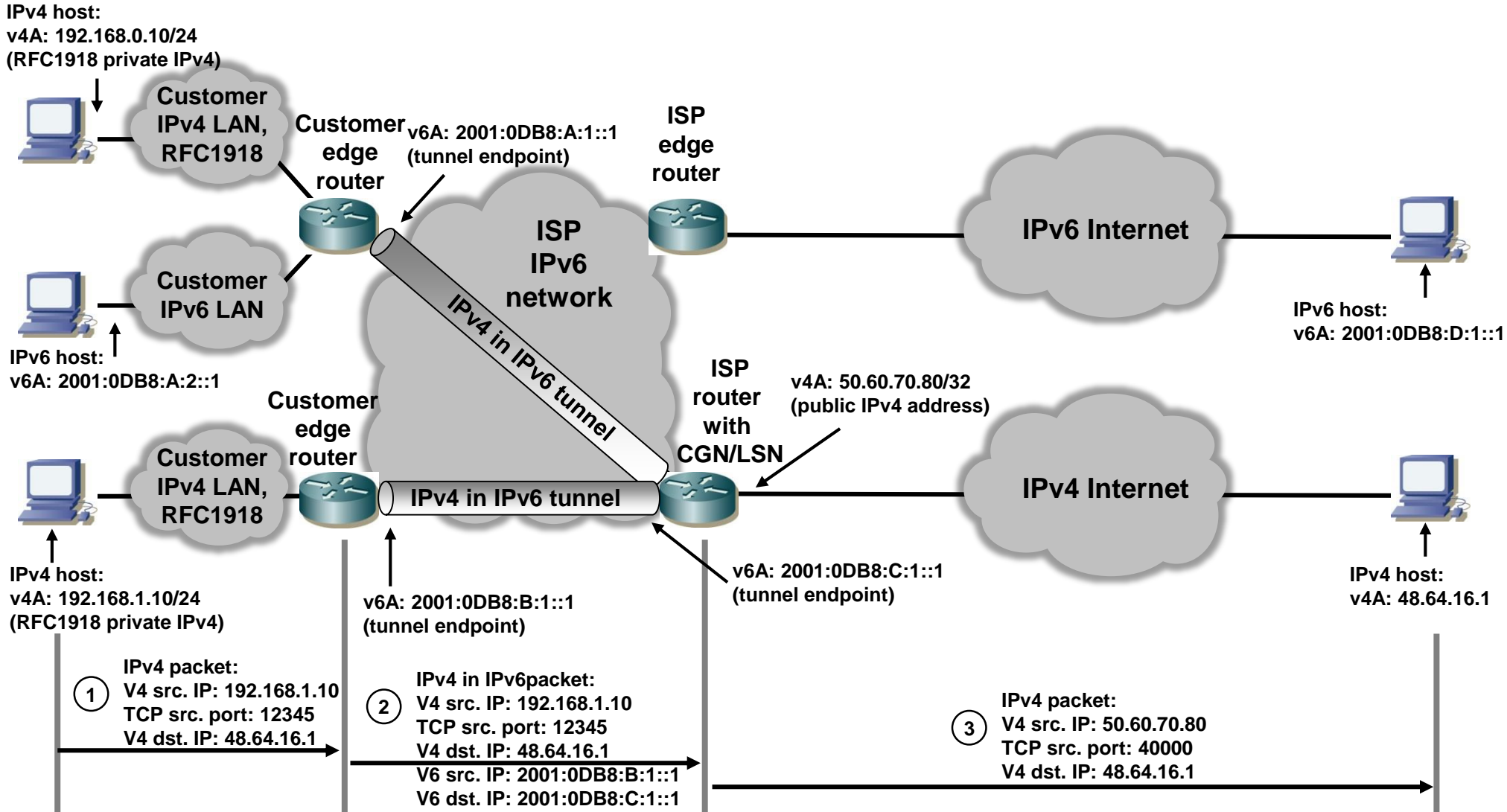
- 😊 Decoupling of IPv6 deployment in provider network from global IPv6 deployment
- 😊 Provider can offer dual service (IPv4 and IPv6)
- 😊 No protocol translation involved (IPv4→IPv6 and vice versa)

In the scenario below, customer A receives both IPv4 and IPv6 service. IPv4 traffic is forwarded through an IPv4 in IPv6 tunnel to the ISP's CGN device where the customer's private IPv4 addresses are mapped to public IPv4 address. IPv6 traffic is routed directly through the ISP's IPv6 network towards the IPv6 Internet.

Customer B gets IPv4 only service. Its traffic is tunneled to the CGN as well.

12. Migration steps for transition from IPv4 to IPv6 (39/78)

B.1.11. Dual-Stack Lite (DS-Lite) (2/3):



12. Migration steps for transition from IPv4 to IPv6 (40/78)

B.1.11. Dual-Stack Lite (DS-Lite) (3/3):

Step by step explanation of packets passing through the different IPv4 networks:

1. Customer edge router:

The customer edge router (CPE) receives an IPv4 packet and encapsulates it without any address mapping into an IPv6 packet (tunneling).

2. Tunnel termination at CGN router:

The CGN router first terminates the tunnel by decapsulating the IPv4 in IPv6 packet.

3. NAT function at CGN router:

Afterwards the NAT function translates source IP address (192.168.1.10) and TCP port number (12345) to a public IPv4 address (50.60.70.80) and the source TCP port number 40000.

Finally the packet is sent to the public IPv4 Internet.

12. Migration steps for transition from IPv4 to IPv6 (41/78)

B.1.12. 6bed4 (1/6):

6bed4 is an approach for connecting IPv6 stacks on embedded devices with restricted computational resources over an existing IPv4 network.

6bed4 uses static tunnels between a 6bed4 router and 6bed4-enabled peers.

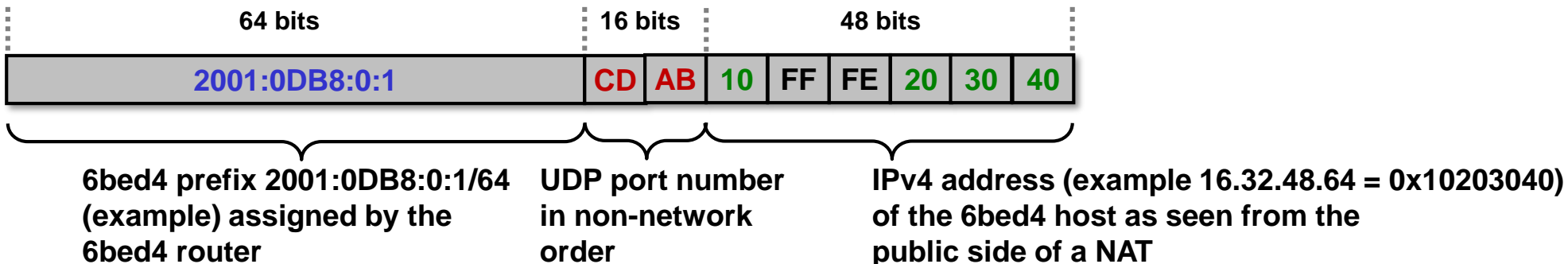
The UDP/IPv4 transport layer is used by the IPv6 stack like a link layer, akin to a native Ethernet layer 2.

6bed4 peers may be placed behind NAT devices like firewalls. Connectivity is provided by a 6bed4 router which inter-connects the 6bed4 peers through tunnels.

In 6bed4, the IPv6 address is composed of a 6bed4 prefix and a 6bed4 interface ID.

The 6bed4 prefix is IPv6-routable and identifies the 6bed4 network. It is assigned by a 6bed4 router to the 6bed4 hosts (peers) through ICMPv6 Router Solicitation / Advertisement.

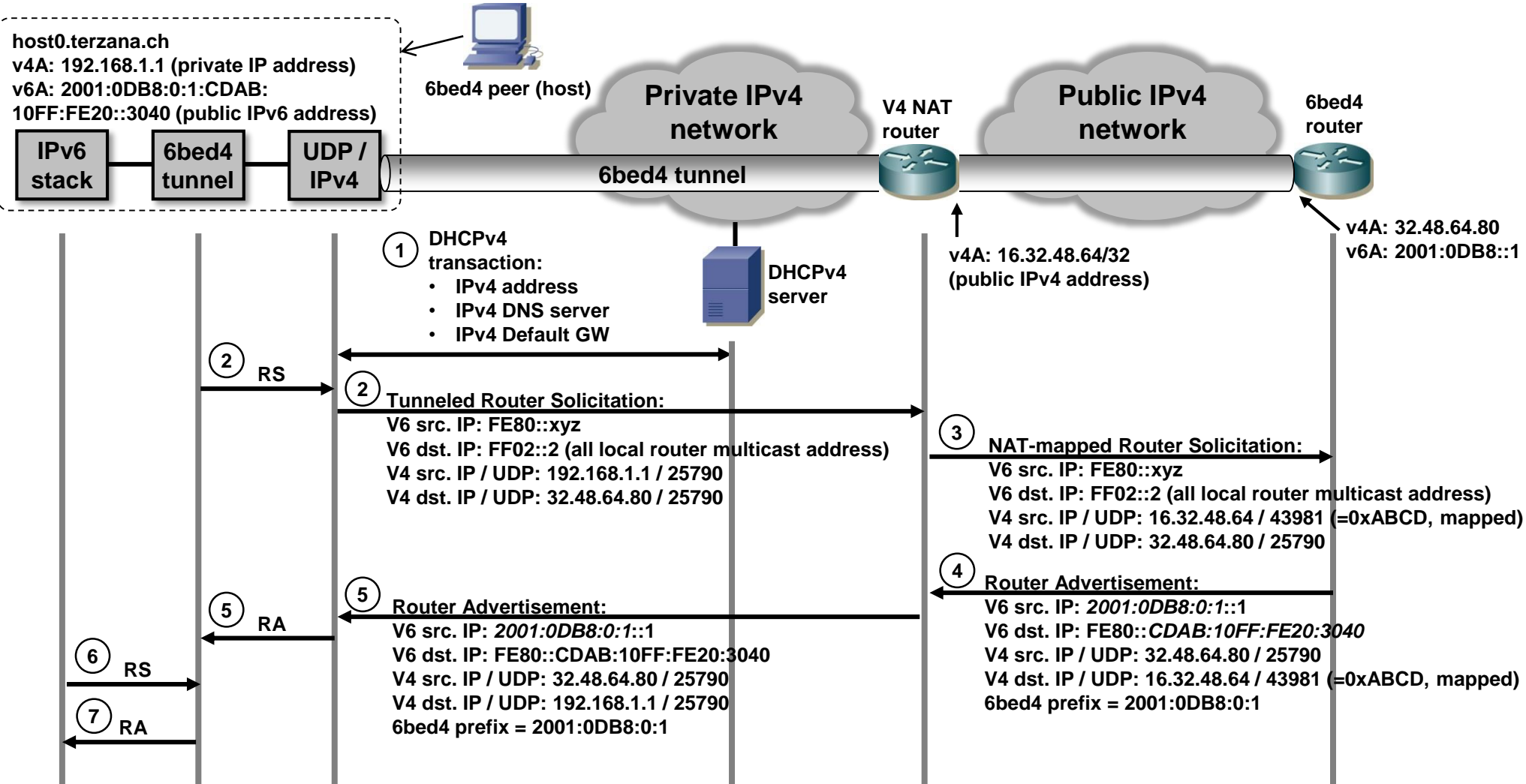
The interface ID contains the IPv4 address and UDP port under which a 6bed4-host is reachable in IPv4 (public side of NAT if present between 6bed4 peer and 6bed4 router).



12. Migration steps for transition from IPv4 to IPv6 (42/78)

B.1.12. 6bed4 (2/6):

6bed4 scenario 1: 6bed4 peer obtains 6bed4 prefix and interface ID from 6bed4 router:



12. Migration steps for transition from IPv4 to IPv6 (43/78)

B.1.12. 6bed4 (3/6):

Step by step explanation of 6bed4 scenario 1 (Router Solicitation (RS) and 6bed4 prefix assignment to 6bed4 peer):

1. DHCPv4 transaction:

The IPv4 stack of the 6bed4 peer performs an IPv4 DHCP transaction and obtains an IPv4 address (192.168.1.1), an IPv4 DNS server address and an IPv4 default gateway address.

2. Router Solicitation by 6bed4 tunnel:

In order to obtain a routable IPv6 prefix and interface ID, the 6bed4 tunnel on host0.terzana.ch sends an IPv6 Router Solicitation packet through the static tunnel to the 6bed4 router.

This RS packet is important to create a NAT-mapping in the NAT-router.

The IPv4 address and UDP port number of the 6bed4 tunnel endpoint on the 6bed4 router may be obtained through different ways outside the scope of 6bed4 such as:

- a. Hardcoded in 6bed4 tunnel (not recommended due to inflexibility)
- b. Anycast address (also hardcoded in tunnel, but needs not to be changed when the 6bed4 host is moved to a different network)
- c. DHCP option
- d. DNS query

The IPv6 source address of the RS packet is some link-local address that is irrelevant for the operation of 6bed4.

The IPv6 RS packet is encapsulated in a UDP/IPv4 packet to be sent to the 6bed4 router.

The default UDP port number used for the tunnel encapsulation is 25790 which reads as 0xBE64 in hexadecimal (IANA approval pending) in a 6bed4 created IPv6 address.

3. NAT-mapping of IPv4 address and UDP port number:

The NAT router translates the private source IPv4 address (192.168.1.1) and UDP port number (25790) to the NAT's public IPv4 address (16.32.48.64) and a mapped UDP port number (43981 = 0xABCD). The NAT-mapping is added to the NAT-mapping-table and ensures that return packets to host0.terzana.ch are passed by the NAT router.

12. Migration steps for transition from IPv4 to IPv6 (44/78)

B.1.12. 6bed4 (4/6):

Step by step explanation of 6bed4 scenario 1 (Router Solicitation (RS) and 6bed4 prefix assignment to 6bed4 peer):

4. Router Advertisement with 6bed4 prefix and interface ID:

The 6bed4 router selects a 6bed4 prefix (2001:0DB8:0:1) for host0.terzana.ch. This prefix identifies the 6bed4 network through which 6bed4 peers can communicate on IPv6 level.

Additionally, the 6bed4 router constructs an IPv6 interface ID (lower 64 bits of IPv6 address) for host0.terzana.ch from the UDP port number and IPv4 address as seen in the packet received from the NAT router (CDAB:10FF:FE20:3040).

This way, the 6bed4 host host0.terzana.ch learns its public IPv4 address and NAT-mapped UDP port number.

The 6bed4 router sends the 6bed4 prefix and interface ID back to host0.terzana.ch in a Router Advertisement. The 6bed4 prefix is contained in the ICMPv6 Router Advertisement as a type 3 option while the interface ID is used in the interface portion of the destination IPv6 address.

The 6bed4 router sends the RA back to the IPv4 address and UDP port number where the RA came from (16.32.48.64 / 43981).

5. Router Advertisement forwarded to host0.terzana.ch, creation of link layer address:

The NAT router changes the destination IPv4 address and UDP port number back to 192.168.1.1 / 25790 and forwards the RA.

The 6bed4 tunnel endpoint on host0.terzana.ch receives the RA, extracts the interface ID and prefix and constructs a link layer address (FE80::CDAB:10FF:FE20:3040).

6. Router Solicitation from IPv6 stack:

The IPv6 stack sends a Router Solicitation to obtain an IPv6 prefix. If the IPv6 stack sends an RS before steps 1 through 5 are completed, the 6bed4 tunnel holds back the answer (Router Advertisement) until steps 1 through 5 are finished.

7. Router Advertisement with 6bed4 prefix from step 1 through 5:

The 6bed4 tunnel sends back a Router Advertisement with the 6bed4 prefix obtained in step 1 through 5.

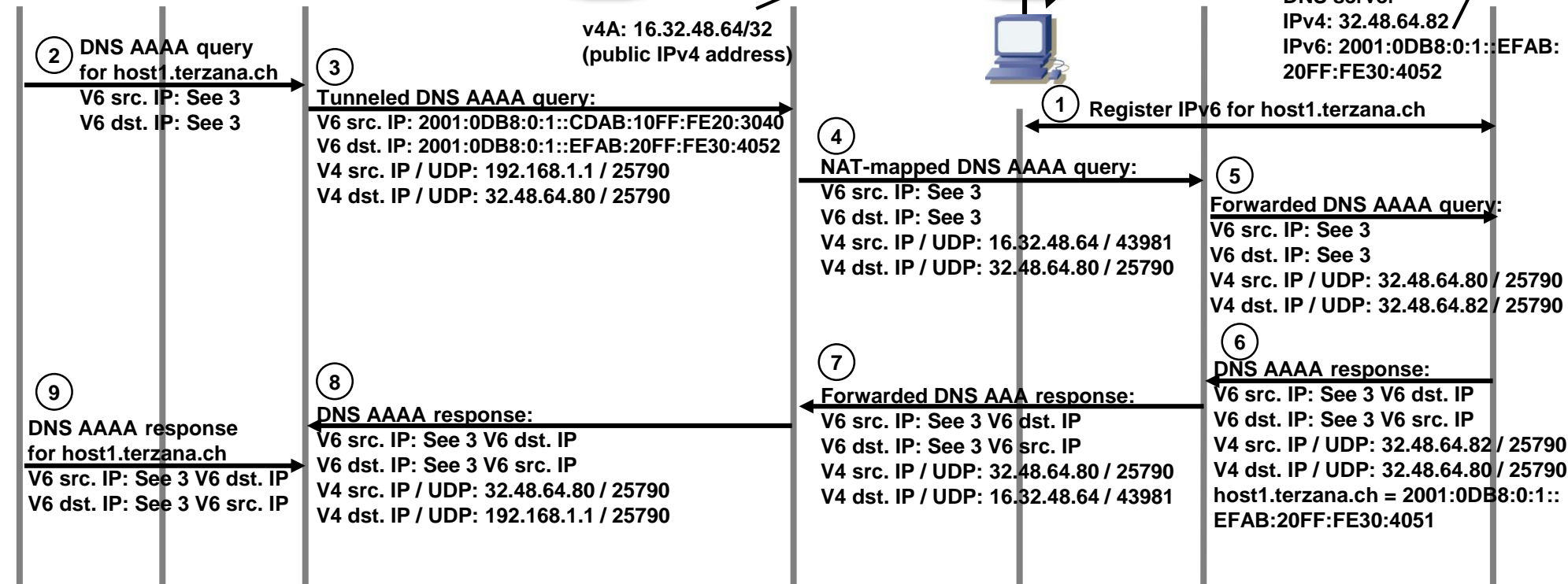
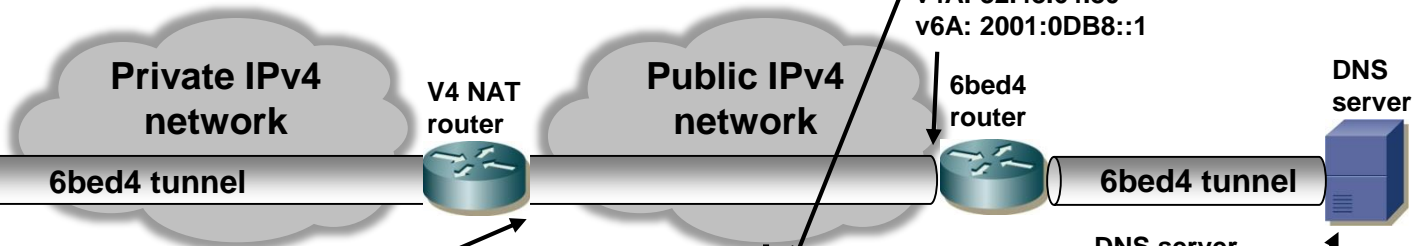
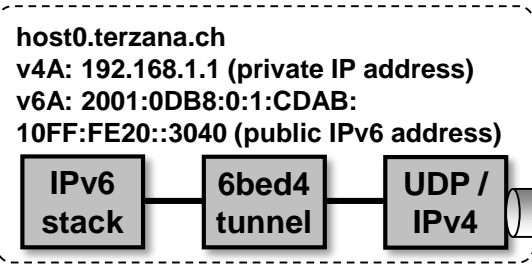
The IPv6 stack reconstructs its interface ID from the link layer address of the tunnel (FE80::CDAB:10FF:FE20:3040) and prefixes it with the 6bed4 prefix obtained in the Router Advertisement (2001:0DB8:0:1). Finally, the IPv6 stack assigns the full IPv6 unicast address (2001:0DB8:0:1::CDAB:10FF:FE20:3040) to the 6bed4 link and adds a default route through the 6bed4 tunnel.

12. Migration steps for transition from IPv4 to IPv6 (45/78)

B.1.12. 6bed4 (5/6):

6bed4 scenario 2: 6bed4 peer performs an IPv6 address lookup:

host1.terzana.ch
 v4A: 32.48.64.81 (private IP address)
 v6A: 2001:0DB8:0:1:EFAB:
 20FF:FE30::4051 (public IPv6 address)



12. Migration steps for transition from IPv4 to IPv6 (46/78)

B.1.12. 6bed4 (6/6):

Step by step explanation of 6bed4 scenario 2 (DNS lookup). N.B.: host0.terzanal.ch sending a packet to host1.terzana.ch on the 6bed4 network is very similar.

1. Remote 6bed4 peer registers its IPv6 address:

host1.terzana.ch registers its 6bed4 IPv6 address (6bed4 prefix and interface ID as seen on public side of NAT) on a DNS-server through some means outside the scope of 6bed4, e.g. through dynamic DNS (RFC2136 or through a web interface).

2. DNS AAAA query:

The DNS resolver on host0.terzana.ch sends an AAAA DNS query for host1.terzana.ch on request of an application process. Again, the IPv6 address (may be a 6bed4 address as well) is obtained through some means outside the scope of 6bed4 such as static configuration or DHCPv6.

3. Tunneling of DNS AAAA query:

The 6bed4 tunnel on host0.terzana.ch tunnels the DNS AAAA query by encapsulating it in UDP/IPv4 and forwards the packet to the 6bed4 router.

4. NAT-mapping of IPv4 address and UDP port number:

The NAT router maps the source UDP port number and IPv4 address and forwards the packet to the 6bed4 router.

5. 6bed4 router forwards packet to DNS server:

The 6bed4 router extracts the IPv6 DNS AAAA query packet from the UDP/IPv4 encapsulated packet (tunnel termination) and forwards it through the 6bed4 tunnel to the DNS server by encapsulating it again in UDP/IPv4.

6. DNS AAAA response:

The DNS server sends back a DNS AAAA response with the answer *host1.terzana.ch = 2001:0DB8:0:1::EFAB:20FF:FE30:4051*. The DNS AAAA response is tunneled back to the 6bed4 router.

7. – 9. Return DNS AAAA response:

Finally, the DNS AAAA response traverses the NAT router and is returned to the DNS resolver.

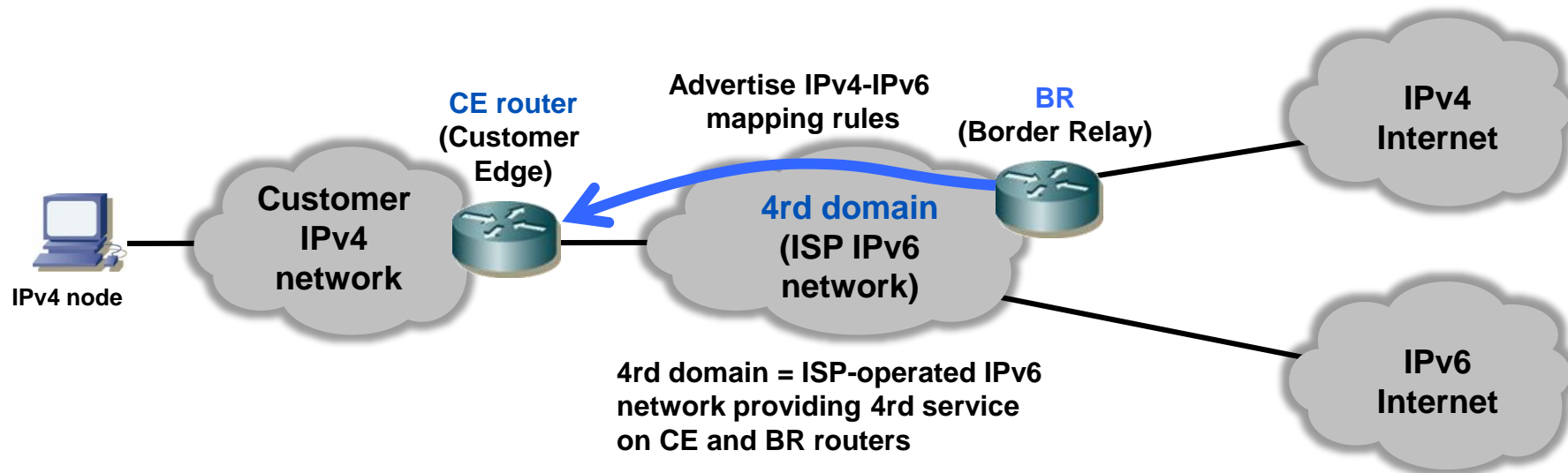
12. Migration steps for transition from IPv4 to IPv6 (47/78)

B.1.13. 4rd (RFC7600) (1/8):

4rd (IPv4 Residual Deployment) is a mechanism for providing IP service to isolated IPv4 islands over a provider's IPv6 network. Thus, 4rd is the opposite of 6rd where IPv6 service is provided over an IPv4 network.

4rd becomes relevant when there will be networks that no longer support IPv4 but only IPv6.

4rd classifies as a tunneling mechanism even though it does not encapsulate full IPv4 packets in a IPv6 packet. 4rd translates the IPv4 header into the IPv6 header, i.e. the IPv4 addresses are mapped into IPv6 addresses (as opposed to 6rd which fully encapsulates IPv6 in IPv4).



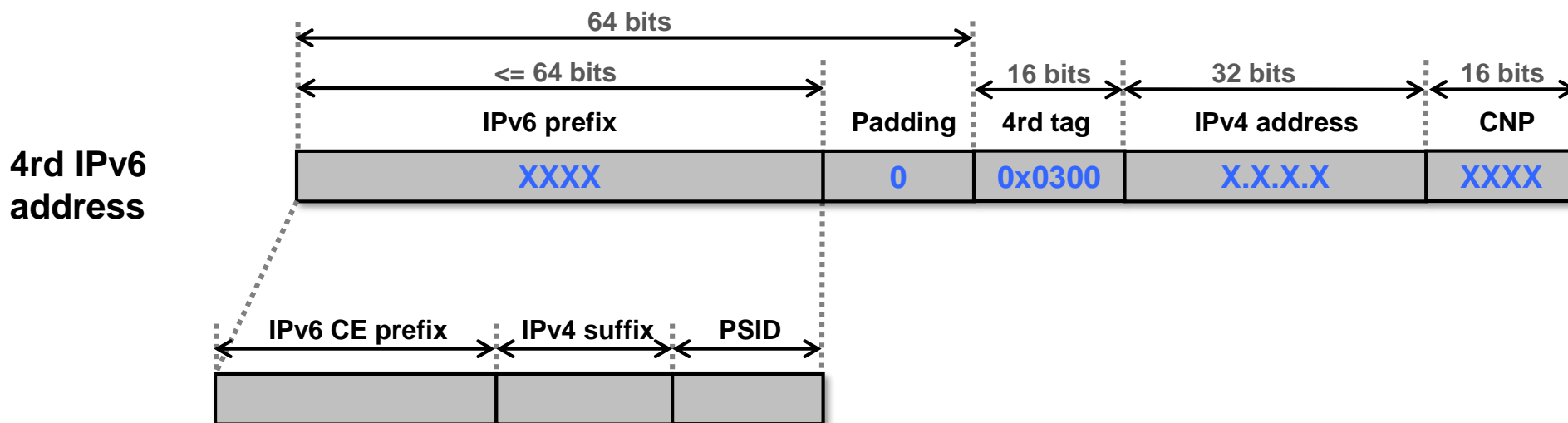
12. Migration steps for transition from IPv4 to IPv6 (48/78)

B.1.13. 4rd (RFC7600) (2/8):

In 4rd, IPv4 addresses are mapped into the IPv6 address. This is possible since the IPv4 address range is smaller than the IPv6 address range.

4rd allows sharing of IPv4 addresses by multiple customers.

To differentiate different customers, a (transport) port range is assigned to each customer (PSID). The port is mapped into the IPv6 address as well.



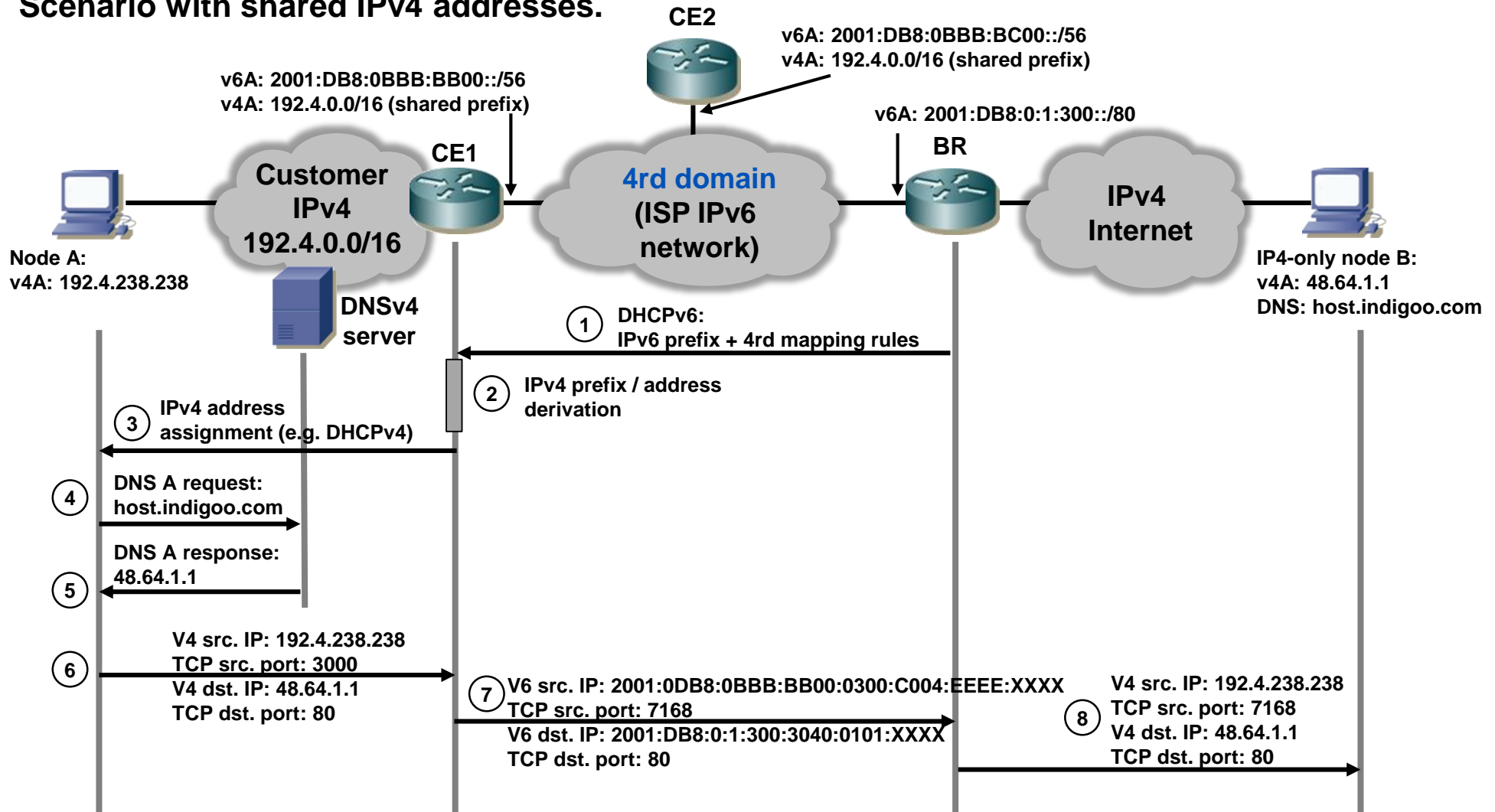
Key:

- PSID Port Set Identifier (port range for CE)
- 4rd tag Tag for differentiating non-4rd-IPv6 and 4rd-IPv6 addresses
- CNP Checksum Neutrality Provider (correction term for TCP/UDP checksum due to changes in IP address)

12. Migration steps for transition from IPv4 to IPv6 (49/78)

B.1.13. 4rd (RFC7600) (3/8):

Scenario with shared IPv4 addresses.



12. Migration steps for transition from IPv4 to IPv6 (50/78)

B.1.13. 4rd (RFC7600) (4/8):

Step by step explanation of 4rd scenario:

1. BR assigns mapping rules to CE1 (DHCPv6 transaction):

After booting, the BR assigns an IPv6 prefix, an IPv4 prefix and 4rd mapping rules to CEs (CE1, CE2 and further CEs) as part of DHCPv6 transactions through DHCPv6 options (DHCPv6 options OPTION_4RD with 4RD_MAP_RULE, see updates to [RFC3315](#)). In this example BR assigns the IPv6 prefix 2001:DB8:0BBB:BB00::/56 to CE1.

The assigned 4rd mapping rules are as follows (CE mapping rule table):

Rule #	Rule IPv4 prefix	Rule IPv4 prefix length	Rule IPv6 prefix	Rule IPv6 prefix length	EA-bits length	Comment
1	0.0.0.0	0	2001:DB8:0:1:300::	80	32	BR mapping rule ("default route") for off-domain IPv4 addresses (IPv4 Internet)
2	192.4.0.0	16	2001:0DB8:0800::	38	18	CE mapping rule
3	192.2.0.0	16	2001:0DB8:0C00::	38	18	CE mapping rule

EA means Embedded Address and consists of the IPv4 suffix and PSID (Port Set Identifier).

Both CE1 and CE2 (and possible further CEs) are assigned the same IPv4 prefix / address (address sharing scenario).

All CEs receive all mapping rules which results in a mesh-topology, i.e. CEs are directly reachable from other CEs.

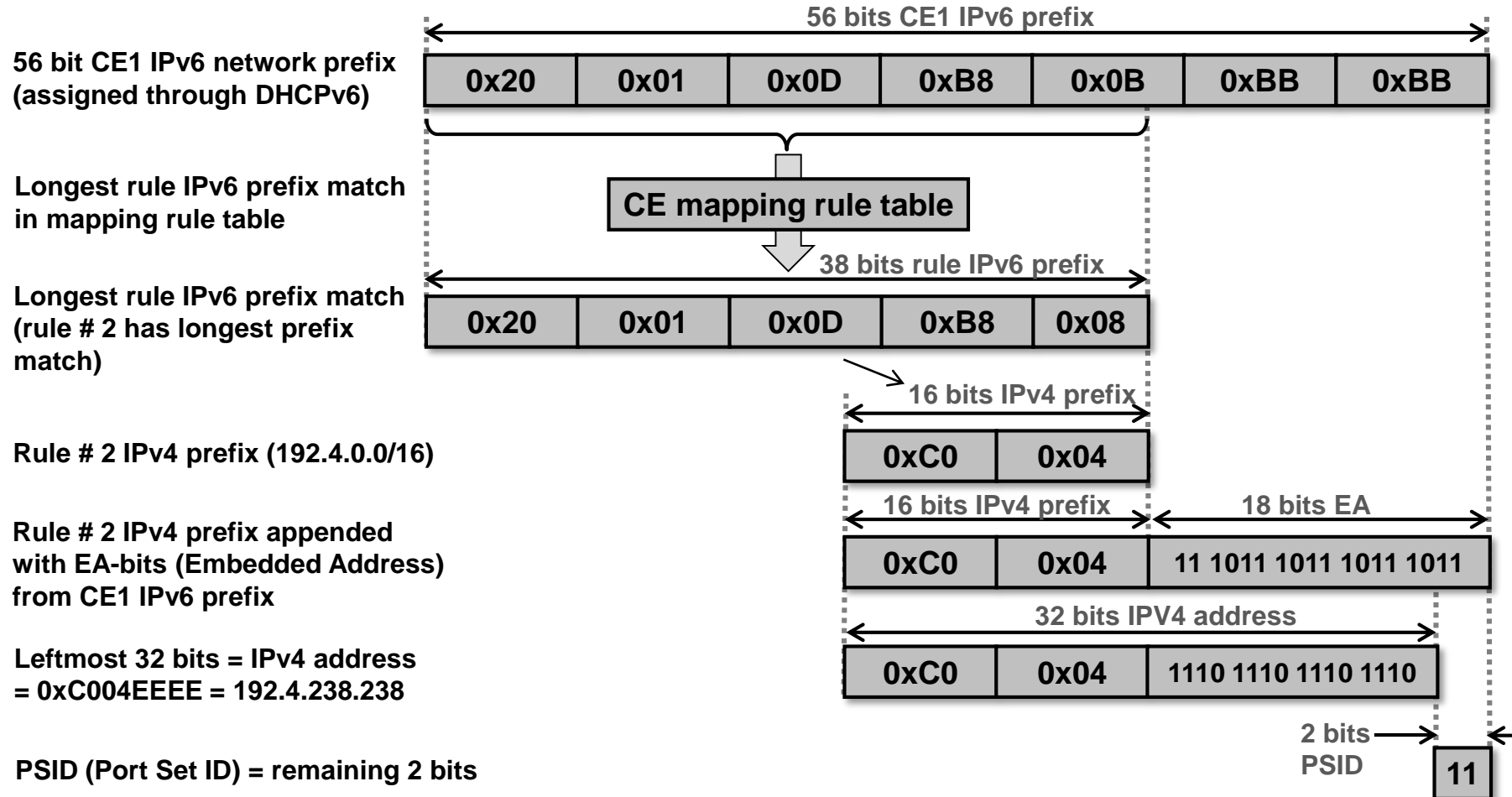
Alternatively, CEs in the 4rd domain may be reachable exclusively through the BR in a hub-spoke-topology.

12. Migration steps for transition from IPv4 to IPv6 (51/78)

B.1.13. 4rd (RFC7600) (5/8):

2. IPv4 address and PSID derivation (1/2):

CE1 derives its IPv4 address and port range as follows:



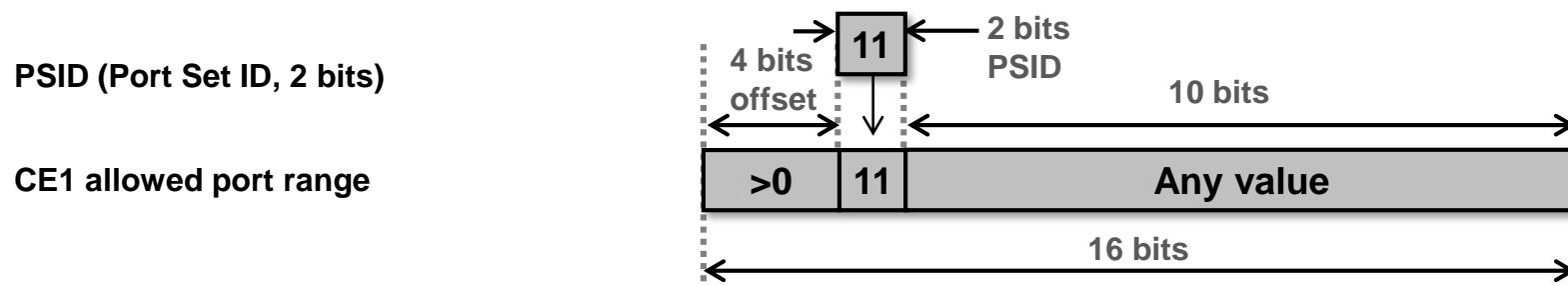
12. Migration steps for transition from IPv4 to IPv6 (52/78)

B.1.13. 4rd (RFC7600) (6/8):

2. IPv4 address and PSID derivation (2/2):

The obtained IPv4 address is 192.4.238.238.

The assigned port range to be used by CE1 for packets sent to the 4rd domain is [4096+0xC00, 4096+0xFFFF] defined as follows:



The port offset 4096 is used so that the reserved port range 0...1023 is not used in the 4rd domain, i.e. port ranges assigned to CEs always start at 4096 (4 bits are reserved to nicely align with nibbles).

3. IPv4 address assignment:

CE1 assigns the IPv4 address to host A, e.g. through DHCPv4.

4. & 5. DNS request and response:

Host A sends a DNS request for host.indigoo.com to the DNS server which responds with an A record containing 48.64.1.1.

6. Host A IPv4 packet:

Host A sends an ordinary IPv4 packet to its default gateway which is CE1.

CE1 contains a NAT44 function that maps the source port number 3000 (0xBB8) to 7168 (0x1C00) which is in the assigned port range in step 2.

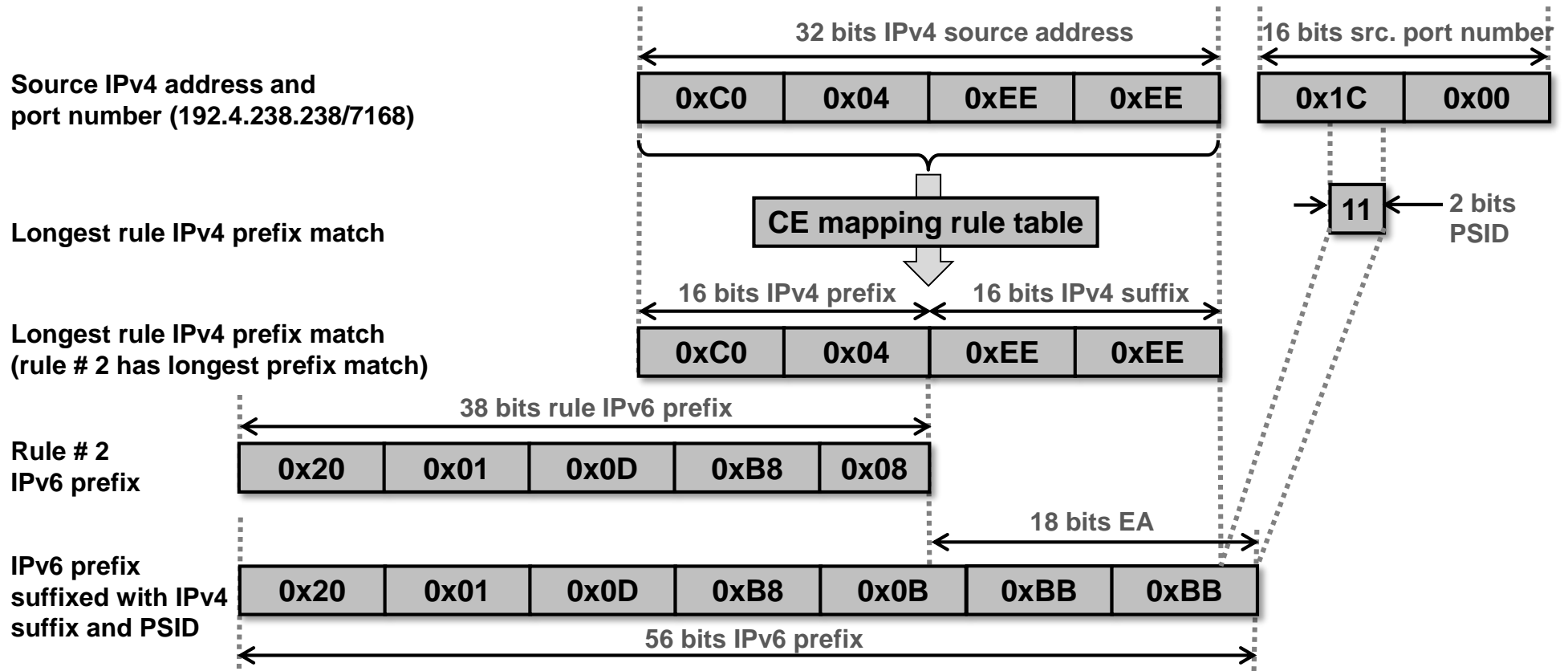
CE1 now forwards the packet through the 4rd domain to BR.

12. Migration steps for transition from IPv4 to IPv6 (53/78)

B.1.13. 4rd (RFC7600) (7/8):

7. Tunneling of IPv4 packet by CE1 (1/2):

CE1 encapsulates the received IPv4 packet, i.e. maps the source and destination IPv4 addresses and port numbers to the corresponding IPv6 addresses.

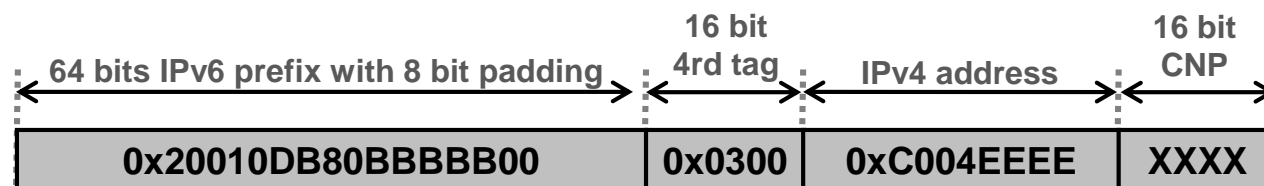


12. Migration steps for transition from IPv4 to IPv6 (54/78)

B.1.13. 4rd (RFC7600) (8/8):

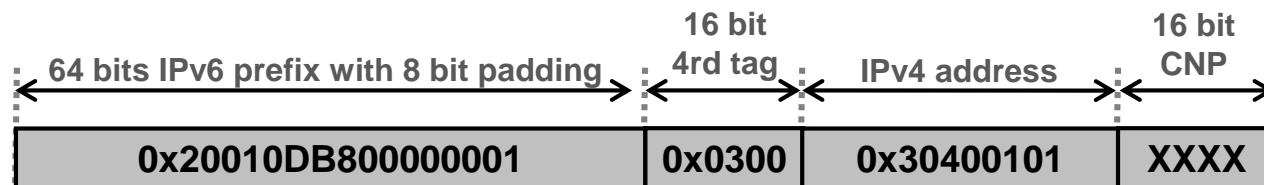
7. Tunneling of IPv4 packet by CE1 (2/2):

IPv6 prefix padded and suffixed with 4rd tag, IPv4 source address and CNP



The same procedure is applied for the target IPv4 address.

The longest IPv4 prefix matching rule is rule #1 which is the rule that matches off-4rd-domain IP addresses such as 48.64.1.1.



8. Packet forwarding through IPv4:

Finally, BR unpacks the packet and maps the IP addresses back the same way as shown in step 2.

12. Migration steps for transition from IPv4 to IPv6 (55/78)

C.1. NAT-PT – Network Address Translation, Port Transl. (RFC2766, obsoleted by RFC4966):

NAT-PT combines address translation together with protocol translation as defined in RFC2765.

NAT-PT maintains a pool of unique IPv4 addresses that are dynamically assigned to IPv6 hosts (stateful translation as the mapping of IPv6 to IPv4 must be maintained in tables).

NAT-PT comes in 2 flavors:

a. Basic NAT-PT:

Translation of IP addresses only.

Maps 1 IPv6 address to 1 IPv4 address (1:1 mapping). Problem: IPv4 address depletion.

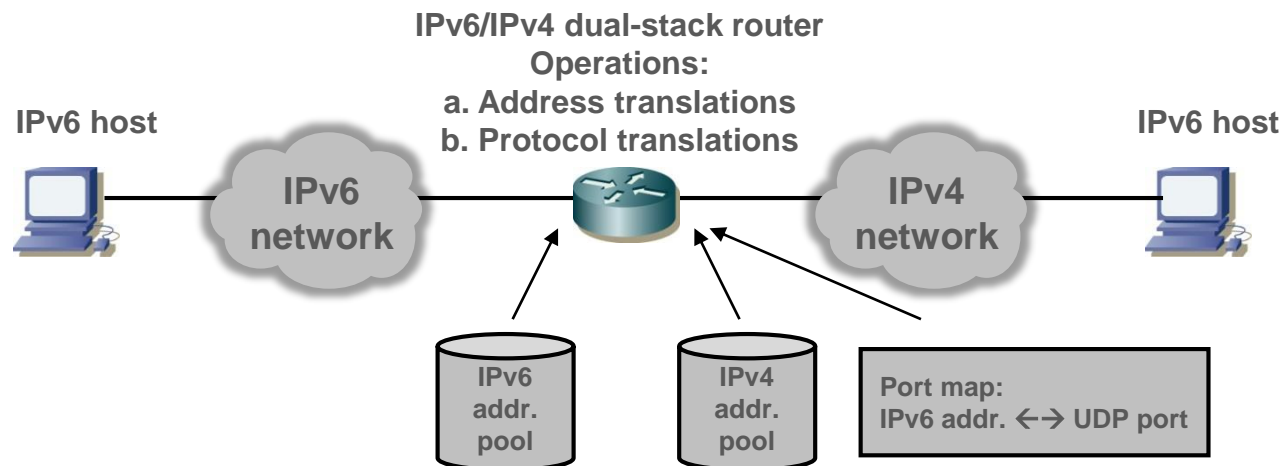
b. NAPT-PT:

Address (IP) and port translation.

Multiple IPv6 address are mapped to 1 common IPv4 address (conserves IPv4 addresses).

1 IPv6 address is mapped to a TCP/UDP port number.

NAT-PT is obsoleted by RFC4966 due to various technical problems that hamper the deployment of IPv6.



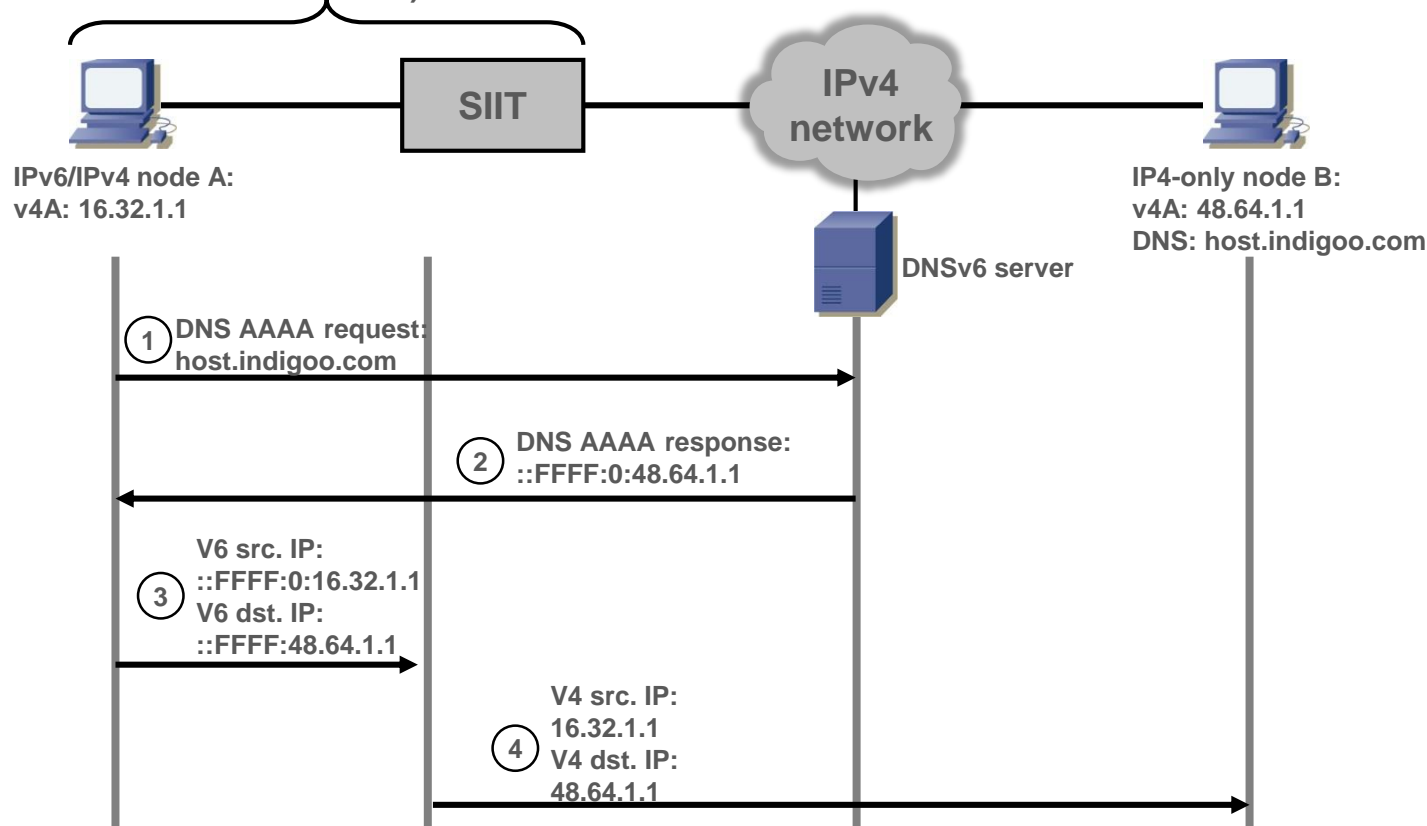
12. Migration steps for transition from IPv4 to IPv6 (56/78)

C.2. SIIT - Stateless IP/ICMP Translation (RFC2765) (1/3):

SIIT is similar to NAT-PT but does not translate port numbers. The translation is stateless (no address pools with stored mappings between IPv4 and IPv6 addresses).

SIIT has been obsoleted by RFC6145.

IPv6 layer and SIIT may be on the same host (dual stack with SIIT, see below)



12. Migration steps for transition from IPv4 to IPv6 (57/78)

C.2. SIIT - Stateless IP/ICMP Translation (RFC2765) (2/3):

Step by step explanation of SIIT transaction:

1.+2. DNSv4 request:

SIIT does not define how a SIIT host obtains a SIIT destination IPv6 address. This may be through standard IPv6 DNS lookup or some other mechanism.

The obtained destination address is the IPv4-mapped address.

3. IPv6 packet:

The IPv6 IP layer constructs a packet where the IPv6 destination address is the destination's IPv4-mapped address and the IPv6 source address is the source's IPv4-translated address.

4. SIIT packet interception and header translation:

The SIIT layer in the stack intercepts the packet. Because the IPv6 dest. address is an IPv4-mapped address (=trigger), SIIT translates the IP header from V6 to V4 with the following mappings (6→4 direction):

Protocol = IPv6 next header field value
Src. IP addr. = Low order 32 bits of IPv6 src. addr.
Dst. IP addr. = Low order 32 bits of IPv6 dst. addr.

In the reverse direction (4→6) the mappings are:

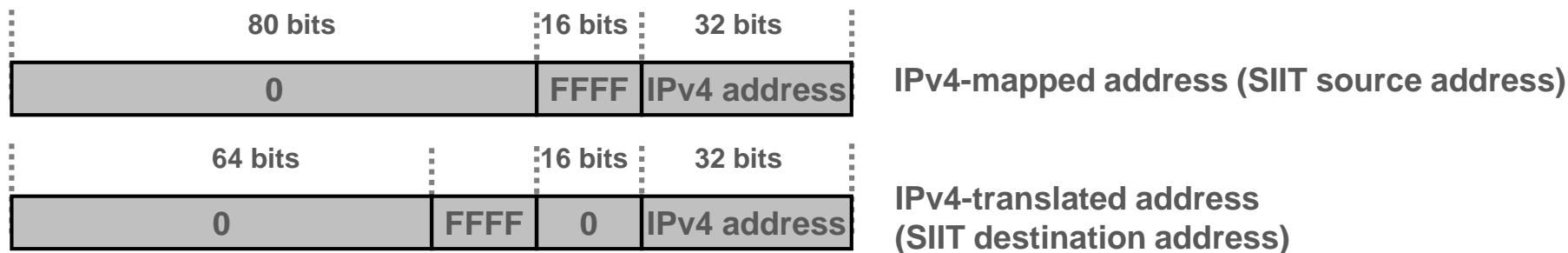
Next header = IPv4 protocol field value
Src. IP addr. = ::FFFF:0:A.B.C.D (IPv4-translated addr.)
Dst. IP addr. = ::FFFF:A.B.C.D (IPv4-mapped addr.)

12. Migration steps for transition from IPv4 to IPv6 (58/78)

C.2. SIIT - Stateless IP/ICMP Translation (RFC2765) (3/3):

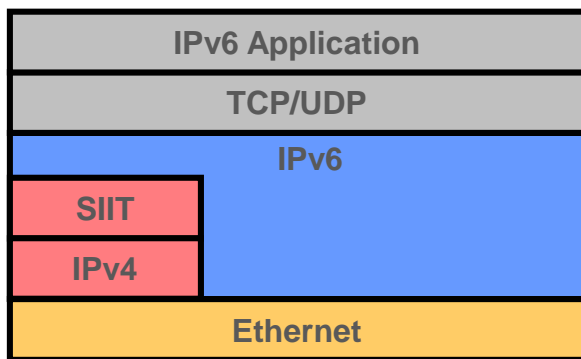
Structure of SIIT IPv6 addresses:

SIIT uses an IPv4-translated address as source IPv6 address to allow tunnels between the IPv6 layer and the SIIT layer (SIIT layer may reside in a separate box or on the same machine as the IPv6 layer).



SIIT stack:

When colocated with the IPv6 layer the SIIT layer intercepts IPv6 packets and translates them.



12. Migration steps for transition from IPv4 to IPv6 (59/78)

C.3. BIS - Bump In the Stack (RFC2767) (1/4):

BIS is similar to SIIT, but the purpose is to provide connectivity for IPv4 hosts over an IPv6 network. Basically BIS is NAT-PT and SIIT functionality combined and moved into the host OS between the IPv4 and IPv6 stack.

Possible scenarios:

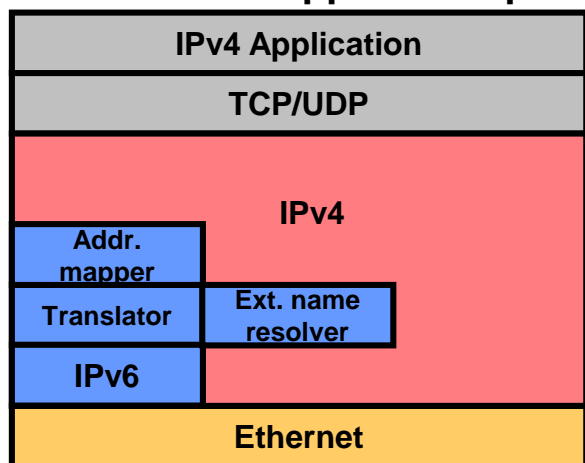
1. Remote host is an IPv4/IPv6 host. DNS provides an A and AAAA DNS mapping.
2. Remote host is an IPv6-only host. DNS provides only an AAAA mapping.
3. Remote host is an IPv4-only host. DNS provides only an A mapping.

BIS stack:

The extension name resolver intercepts IPv4 DNS queries (A queries) and creates an additional query for A (IPv4) and AAAA (IPv6) queries.

The translator component translates the IPv4 header into an IPv6 header according to SIIT (see above).

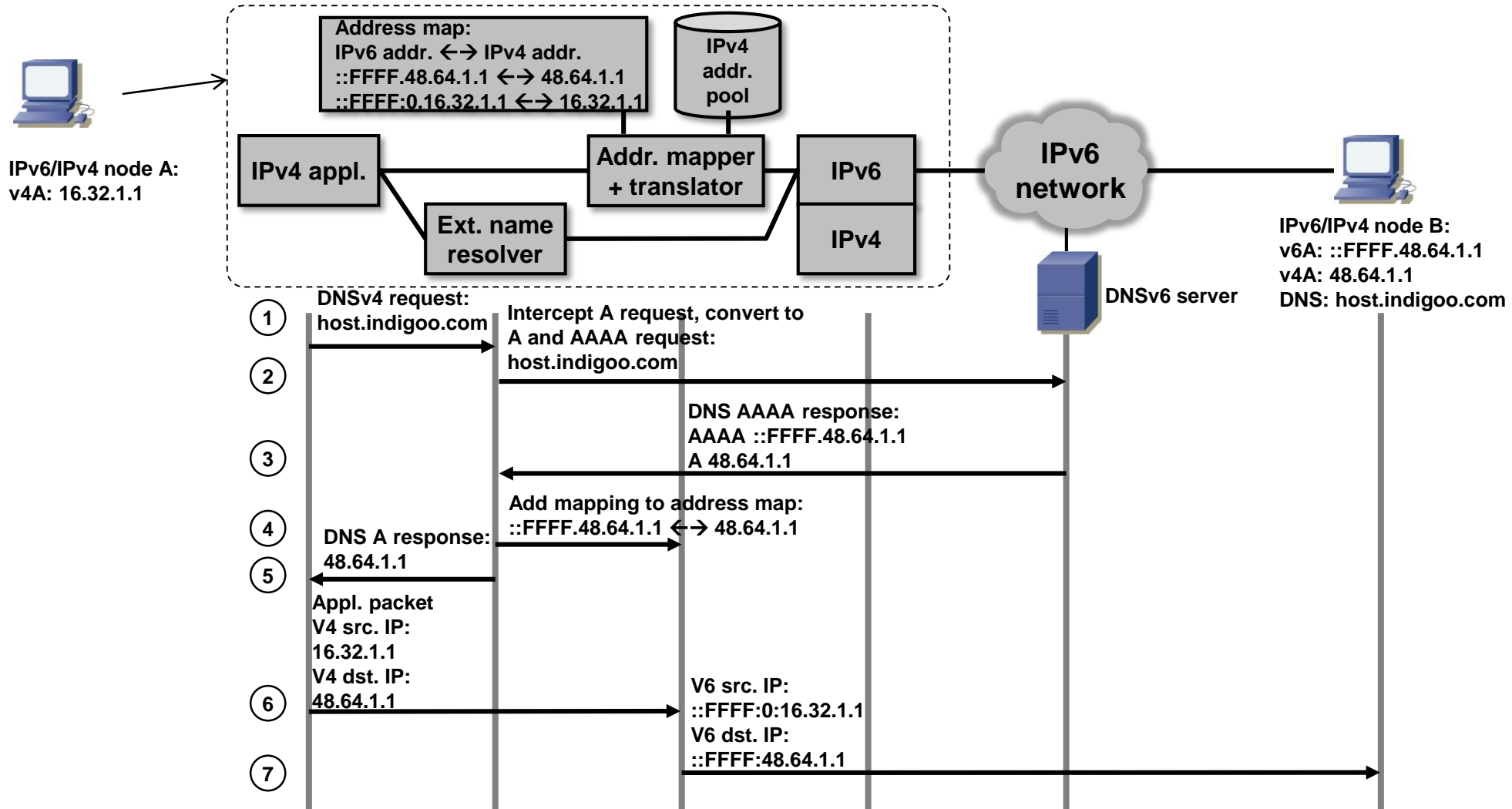
The address mapper is responsible for storing the IPv4 to IPv6 address pairs.



12. Migration steps for transition from IPv4 to IPv6 (60/78)

C.3. BIS - Bump In the Stack (RFC2767) (2/3):

BIS scenario 1: Remote host B is an IPv4 / IPv6 node



12. Migration steps for transition from IPv4 to IPv6 (61/78)

C.3. BIS - Bump In the Stack (RFC2767) (3/4):

Step by step explanation of BIS scenarios 1-3 (1/2):

1. DNSv4 request:

The v4 application makes a DNS A request for host.indigoo.com (v4 request).

2. Extension name resolver DNS request interception:

The extension name resolver (part of the BIS stack) intercepts the DNSv4 request and converts it into a V4/v6 request (A and AAAA request). The DNS server may be an IPv4 or IPv6 host.

3. DNS AAAA response:

Scenario 1 (IPv4/IPv6):

The DNS server responds with an A and AAAA response.

Scenario 2 (IPv6-only):

The DNS server responds with an AAAA response only.

Scenario 3 (IPv4-only):

The DNS server responds with an A response only.

4. Add mapping between IPv4 and IPv6 address:

Scenario 1 (IPv4/IPv6):

The extension name resolver instructs the address mapper and translator to add a mapping between the received IPv4 and IPv6 addresses to the mapping table (::FFFF.48.64.1.1 ↔ 48.64.1.1).

Scenario 2 (IPv6-only):

The extension name resolver instructs the address mapper and translator to allocate a free IPv4 address from the address pool and to add the mapping between the IPv4 and IPv6 address to the mapping table (2001:0DB8:B:1::1 ↔ 48.64.1.1).

Scenario 3 (IPv4-only):

As there is no IPv6 address there is no mapping between IPv4 and IPv6. The transaction continues with IPv4 only.

12. Migration steps for transition from IPv4 to IPv6 (62/78)

C.3. BIS - Bump In the Stack (RFC2767) (4/4):

Step by step explanation of BIS scenarios 1-3 (2/2):

5. Report IPv4 address to the IPv4 application:

The extension name resolver reports the queried IPv4 address to the calling application.

6. Send packet with IPv4 addresses:

The IPv4 application sends a packet with IPv4 addresses.

7. Packet interception and header translation:

The address mapper and translator intercepts the IPv4 packet, translates the header to IPv6 and inserts the IPv6 addresses as defined in the address map. This step is identical to what SIIT does. Finally the packet is forwarded towards the IPv6 destination.

12. Migration steps for transition from IPv4 to IPv6 (63/78)

C.4. BIA - Bump In the API (RFC3338):

Similar to BIS, but the purpose of BIA is to allow IPv4 applications to communicate over an IPv6 network.

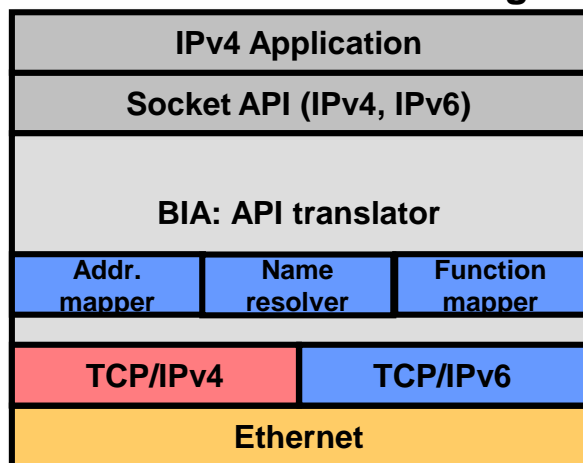
Unlike BIS, BIA translates between IPv4 and IPv6 APIs (socket layer). BIA is implemented as a layer between the application and the transport layer (TCP, UDP).

The BIA scenarios are very similar to BIS (also see scenarios above). Instead of translating IPv4 headers (translator), BIA translates the socket API calls, so there is no need to translate IPv4 headers.

BIA stack:

The extension name resolver intercepts IPv4 DNS queries (A queries) and creates an additional query for A (IPv4) and AAAA (IPv6) queries.

The function mapper component translates the IPv4 socket calls into corresponding IPv6 socket calls. The address mapper is responsible for storing the IPv4 to IPv6 address pairs (allocates IPv4 addresses from the unassigned range 0.0.0.0/24).



12. Migration steps for transition from IPv4 to IPv6 (64/78)

C.5. BIH – Bump In the Host:

BIH (RFC6535) combines the translation techniques BIS (RFC2767) and BIA (RFC3338) in one single RFC.

As such, RFC6535 obsoletes the BIS and BIA RFCs (RFC2767 and RFC3338).

BIH introduces the following main changes with respect to BIS (RFC2767) and BIA (RFC3338):

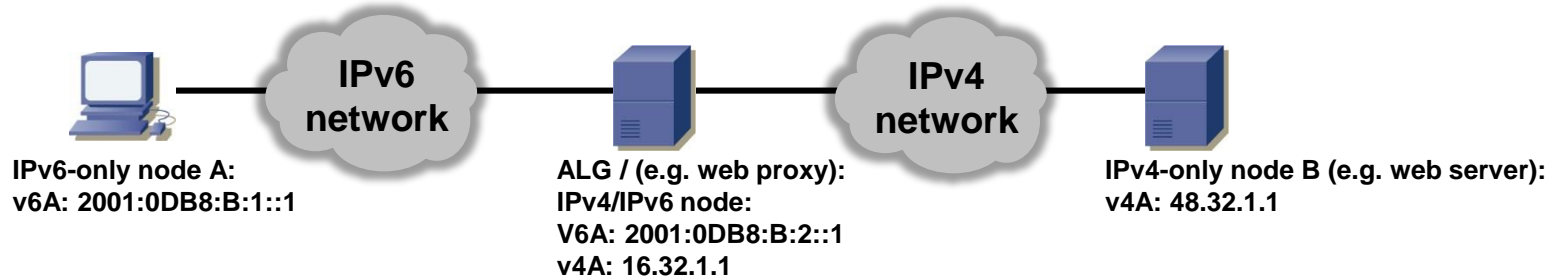
1. Use of RFC1918 addresses for address synthesis instead of unassigned IP addresses like 0.0.0.1
2. Support for DNS pointer queries (PTR)
3. DNSSEC support
4. RFC6535 recommends to use BIS (RFC2767) over BIA (RFC3338)
5. RFC6535 is on the standards track (RFC2767 was informational and RFC3338 was experimental)

12. Migration steps for transition from IPv4 to IPv6 (65/78)

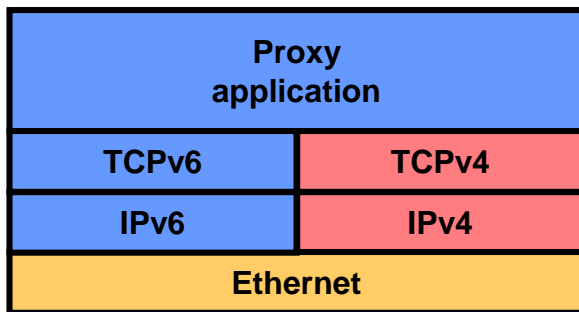
C.6. ALG - Application Layer Gateway:

ALGs are simple dual-stack application layer proxies that perform translation on application layer. Unlike SOCKS64 (see below), the translation includes the application layer protocol (e.g. HTTP), e.g. the translation of URLs (IPv4 addresses to IPv6 address).

ALGs can either connect existing IPv4 servers to the IPv6 Internet (picture below) or make new IPv6 servers available on the existing IPv4 Internet.



ALG stack:



12. Migration steps for transition from IPv4 to IPv6 (66/78)

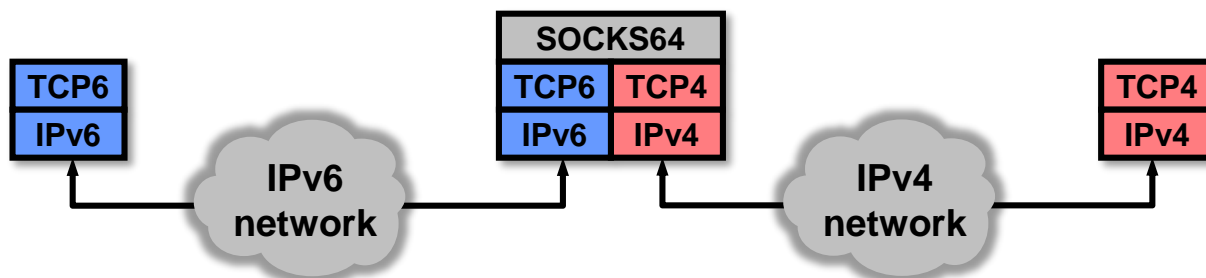
C.7. SOCKS64 (RFC1928 / RFC3089):

RFC1928 defines the SOCKS protocol for IPv4 and IPv6. It allows hosts to traverse firewalls similar to ALG. Unlike ALGs, SOCKS gateways perform only TCP / UDP protocol termination and address translation.

RFC3089 makes use of the SOCKS protocol to provide circuit layer translation between IPv4 and IPv6. Unlike ALG, the SOCKS64 translation is application-protocol agnostic (e.g. no URL translation for HTTP).

In SOCKS64, DNS name resolution is delegated to the SOCKS gateway.

SOCKS64 protocol stack:



12. Migration steps for transition from IPv4 to IPv6 (67/78)

C.8. TRT – Transport Relay Translator (RFC3142) (1/3):

TRT allows to connect IPv6 hosts with IPv4 servers (e.g. web servers).

TRT is very similar to SOCKS64. But unlike SOCKS64, TRT does not require modifications on the IPv6 or IPv4 hosts.

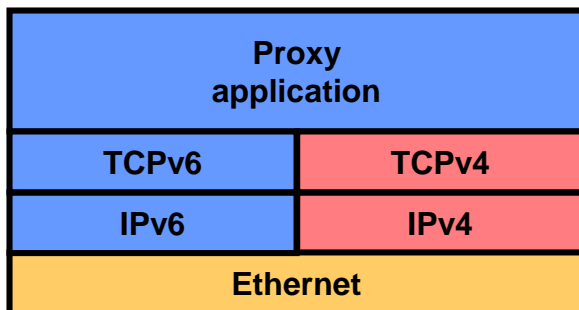
Like SOCKS64 TRT terminates transport protocols (TCP, UDP) but does not filter or manipulate the application PDU (APDU).

Similar to BIS/BIA, TRT is based on "spoofing" DNS responses. But unlike BIS/BIA, TRT uses an application layer DNS proxy on a separate machine. Compared to BIS/BIA this has the advantage that the IPv6 or IPv4 hosts do not need to be modified.

Possible scenarios:

1. Remote host is an IPv4-only host. DNS only provides an A mapping.
2. Remote host is an IPv4/IPv6 host. DNS provides an A and AAAA DNS mapping.
3. Remote host is an IPv6-only host. DNS only provides an AAAA mapping.

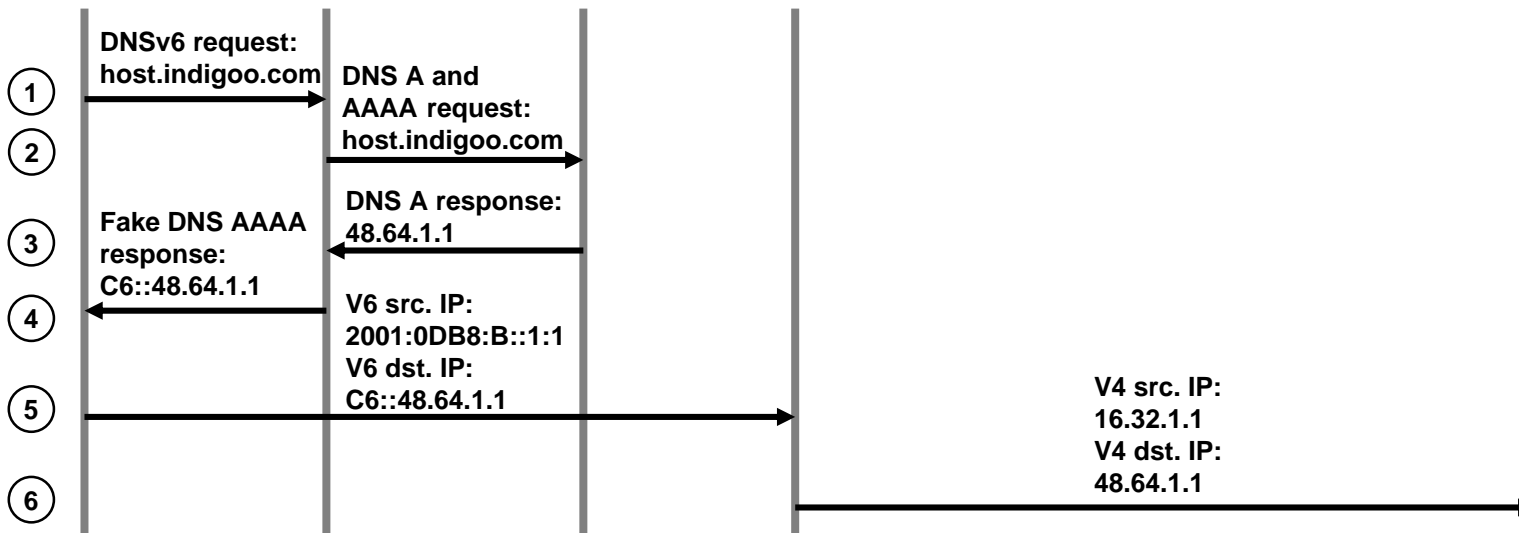
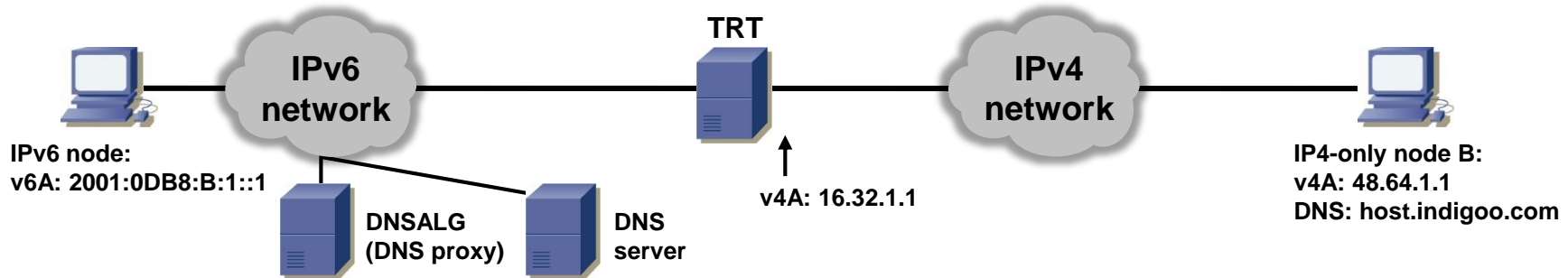
TRT stack:



12. Migration steps for transition from IPv4 to IPv6 (68/78)

C.8. TRT – Transport Relay Translator (RFC3142) (2/3):

TRT scenario 1: Remote host B is an IPv4-only node



12. Migration steps for transition from IPv4 to IPv6 (69/78)

C.8. TRT – Transport Relay Translator (RFC3142) (3/3):

Step by step explanation of TRT scenarios:

1. DNSv6 request:

The v6 application on host A makes a DNS AAAA request for host.indigoo.com (v6 request). Host A sends the DNS request to the DNS proxy (DNSALG).

2. DNS request:

The DNS proxy sends an A and AAAA query to the actual DNS server.

3. DNS response:

Scenario 1 (remote host B is IPv4-only):

The DNS server responds with an A response.

Scenario 2 (remote host B is IPv4/IPv6):

The DNS server responds with an A and AAAA response.

Scenario 3 (remote host is an IPv6-only node):

The DNS server responds with an AAAA response only.

4. Add mapping between IPv4 and IPv6 address:

Scenario 1 (IPv4-only):

The DNS proxy constructs a TRT address from the received IPv4 address with the prefix C6::/8.

Scenario 2+3 (IPv4/IPv6 and IPv6-only):

Obviously the remote host B is connected to IPv6 as well. Thus the DNS proxy returns the IPv6 address to host A. The communication continues with IPv6.

5.+6. Packet translation by TRT:

The TRT converts the IPv6 packet into an IPv4 packet (terminates TCP or UDP v6, creates a new packet). It uses the low order 4 bytes of the destination IPv6 address as destination IPv4 address.

The TRT uses its own IPv4 address as source address.

12. Migration steps for transition from IPv4 to IPv6 (70/78)

C.9. NAT64 (RFC6052 and RFC6144):

NAT64 is a framework for connecting IPv6 hosts with IPv4 hosts.

It allows IPv6 hosts to communicate with IPv4 servers by mapping the IPv4 address range into the IPv6 address range.

NAT64 is similar to SIIT (RFC2765) and NAT-PT (RFC2766) but supersedes both due to technical limitations of these early transition approaches.

NAT64 is a framework defined in different RFCs:

<u>RFC6144</u>	Framework for IPv4/IPv6 Translation, applies to both stateless and stateful NAT64.
<u>RFC6145</u>	Stateless IP/ICMP Translation Algorithm (SIIT – replacing <u>RFC2765</u>)
<u>RFC6146</u>	Stateful NAT64
<u>RFC6147</u>	DNS64

NAT64 comes in 2 flavors:

a. Stateless NAT64 (RFC6145) using IP address mapping in both IPv6→IPv4 and IPv4→IPv6 initiated sessions.

b. Stateful NAT64 (RFC6146) using IP address and port number mapping in IPv6→IPv4 initiated sessions only.

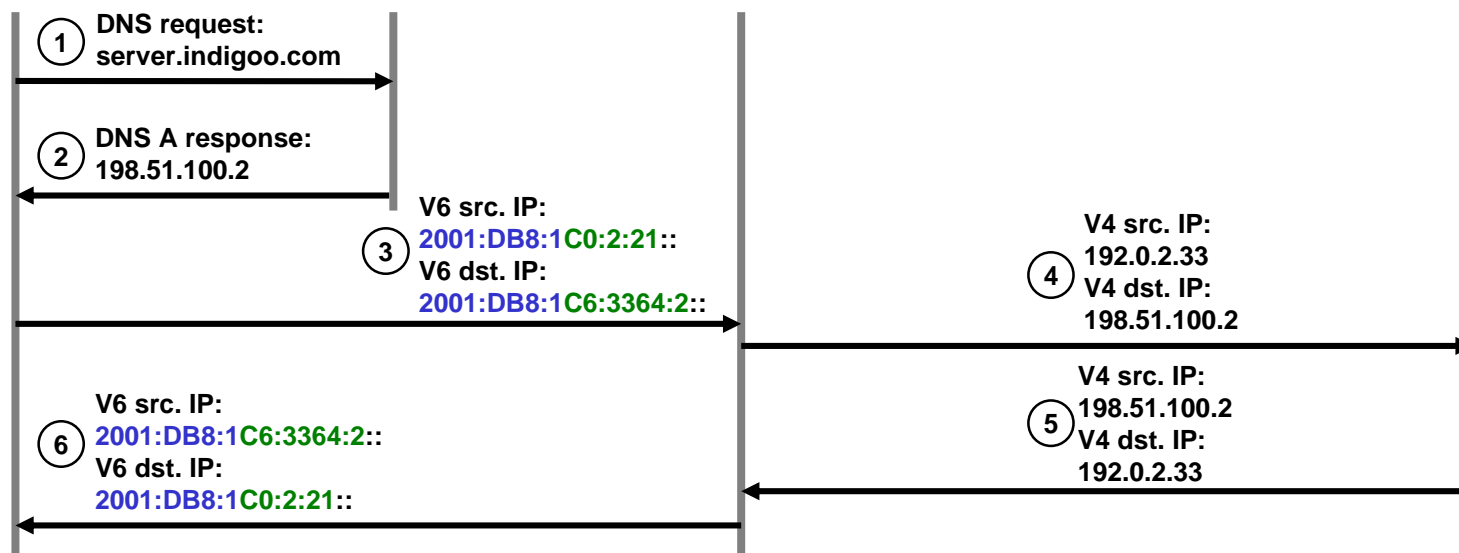
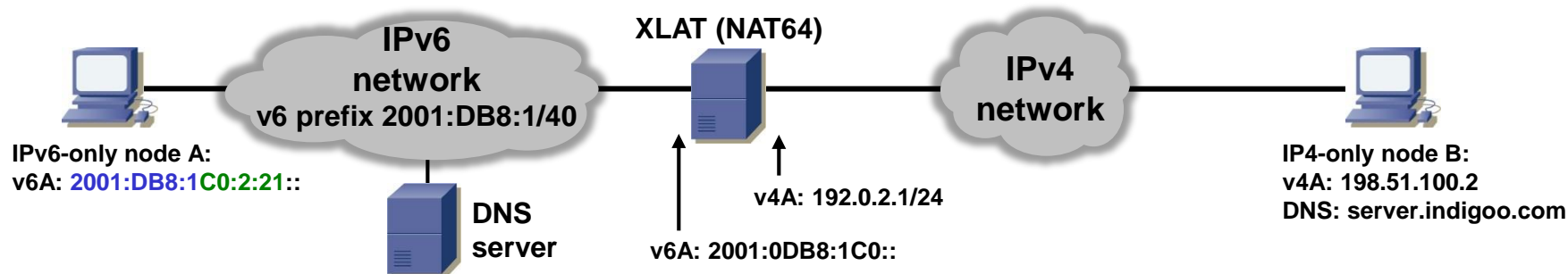
12. Migration steps for transition from IPv4 to IPv6 (71/78)

C.9. Stateless NAT64 (RFC6052 and RFC6145) (1/5):

Stateless NAT64 allows IPv6 nodes to reach IPv4 servers and vice versa.

Stateless NAT64 is possible only if all nodes in the IPv6 network use the same IPv6 prefix.

Stateless NAT 64 Scenario 1: IPv6 node reaches IPv4 node



12. Migration steps for transition from IPv4 to IPv6 (72/78)

C.9. Stateless NAT64 (RFC6052 and RFC6145) (2/5):

Step by step explanation of stateless NAT64 scenario 1 (1/2):

1. DNS request:

The IPv6-only node A makes a DNS request for server.indigoo.com.

2. DNS A record server response:

The DNS server returns an A record with the IPv4 address 198.51.100.2.

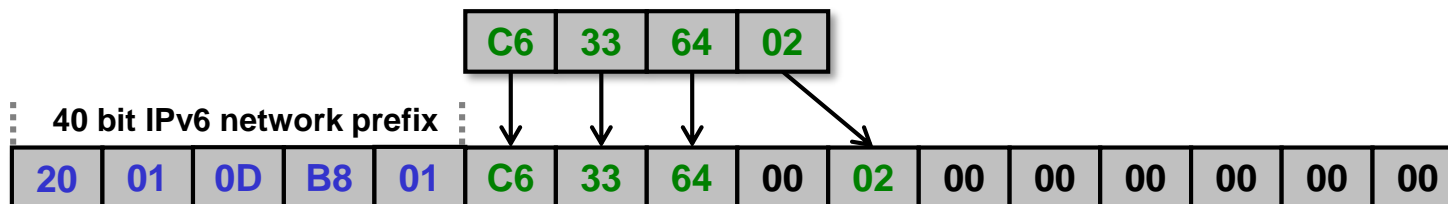
N.B.: Here the DNS server could itself perform the mapping done in step 3 by IPv6 node A and return an already mapped IPv6 address as a AAAA record (DNS64, see below).

3. Destination address mapping, forwarding of packet to XLAT:

Node A performs the IPv6 destination address mapping as defined in RFC6052.

The IPv6 destination address 2001:DB8:1C6:3364:2:: contains both the IPv6 network specific prefix 2001:DB8:1/40 and the IPv4 address 198.51.100.2 which is mapped to byte positions 5, 6, 7 and 9 as show below.

N.B.: Other mappings to different byte positions are defined in RFC6052.



The routing in the IPv6 network forwards all packets with the 2001:DB8:1/40 prefix to the IPv6 interface of the XLAT.

12. Migration steps for transition from IPv4 to IPv6 (73/78)

C.9. Stateless NAT64 (RFC6052 and RFC6145) (3/5):

Step by step explanation of stateless NAT64 scenario 1 (2/2):

4. Translation to IPv4 packet:

The XLAT receives the packet, translates the IPv6 header to IPv4 according to RFC6145 and forwards the IPv4 packet with source address 192.0.2.33 and destination address 198.51.100.2 towards IPv4-only node B.

5. IPv4 return packet:

IPv4 node B sends back an IPv4 packet. The routing in the IPv4 network forwards the packets with IPv4 prefix 192.0.2.0/24 to the IPv4 interface of the XLAT.

6. Address mapping, forwarding to IPv6 node A:

The XLAT performs the reverse mapping by mapping the IPv4 source and destination addresses to the IPv6 source and destination addresses according to RFC6145.

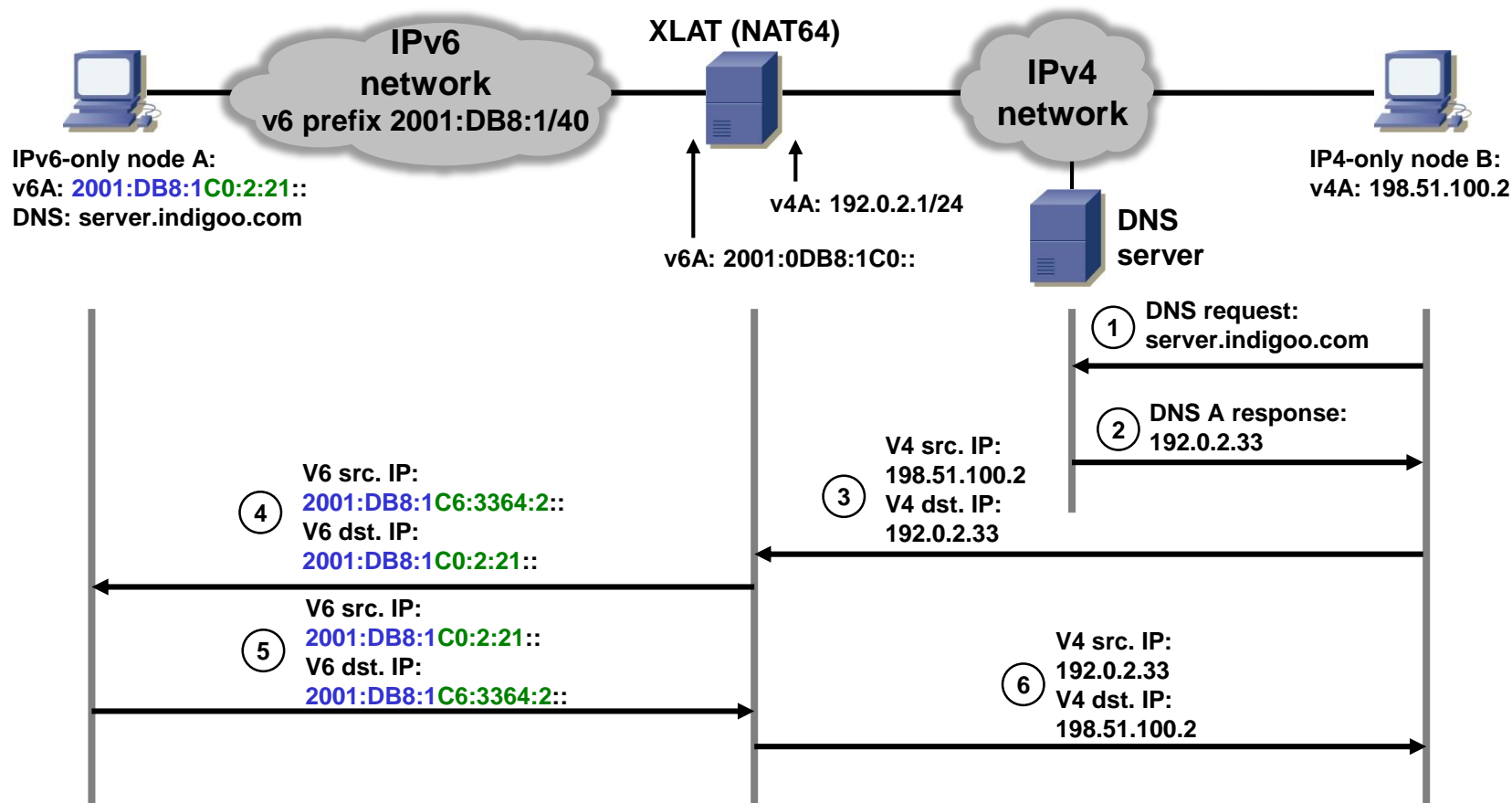
Because all nodes in the IPv6 network use the same IPv6 prefix, the XLAT just needs to compose the IPv6 prefix and the IPv6 addresses in the IPv4 packet into IPv6 addresses.

The IPv6 prefix may be administratively configured on the XLAT.

12. Migration steps for transition from IPv4 to IPv6 (74/78)

C.9. Stateless NAT64 (RFC6052 and RFC6145) (4/5):

Stateless NAT 64 Scenario 2: IPv4 node reaches IPv6 node



12. Migration steps for transition from IPv4 to IPv6 (75/78)

C.9. Stateless NAT64 (RFC6052 and RFC6145) (5/5):

Step by step explanation of stateless NAT64 scenario 2:

1. DNS request:

The IPv4-only node B makes a DNS request for server.indigoo.com.

2. DNS A record server response:

The DNS server returns an A record with the IPv4 address 192.0.2.33.

3. IPv4 packet forwarding of packet to XLAT:

Node B sends the IPv4 packet through the IPv4 network to the IPv4 interface of the XLAT.

4. Address mapping, forwarding to IPv6 node A:

The XLAT performs the address mapping as defined in RFC6052.

The IPv6 destination address **2001:DB8:1C0:2:21::** contains both the IPv6 network specific prefix **2001:DB8:1/40** and the IPv4 address **192.0.2.21** which is mapped to byte positions 5, 6, 7 and 9 in the IPv6 address.

The routing in the IPv6 network forwards the packet to node B.

5. & 6. Return packet mapping:

The return packet sent by node A to node B is treated as described in scenario 1.

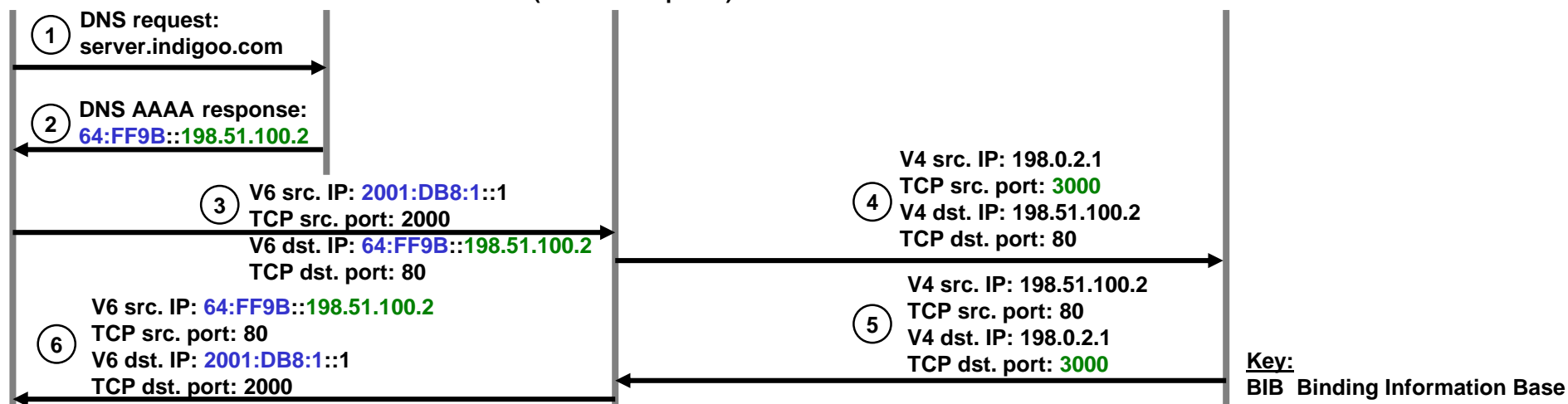
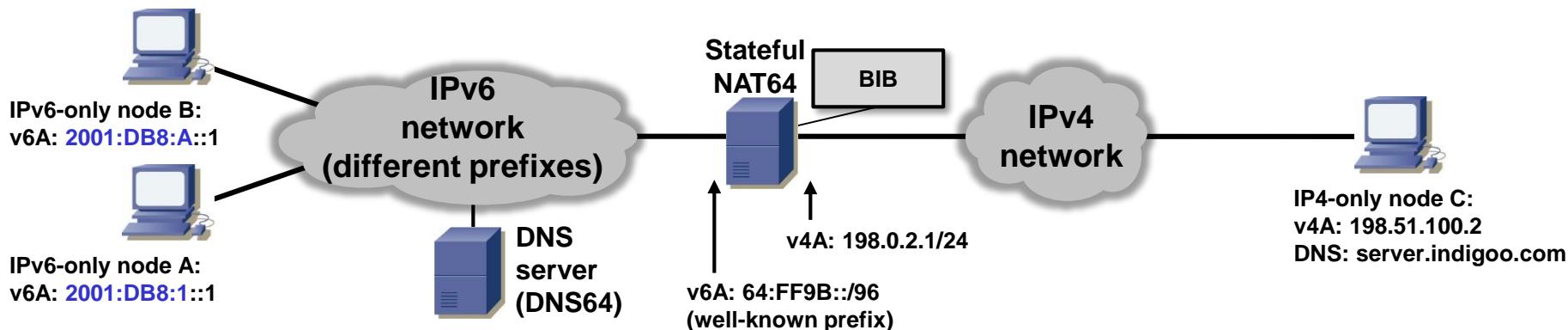
12. Migration steps for transition from IPv4 to IPv6 (76/78)

C.9. Stateful NAT64 (RFC6052 and RFC6146) (1/3):

In stateful NAT64, the IPv6 nodes may have different prefixes.

Just like in NAT44, the NAT64 function translates IP addresses and maps port numbers so that packets in IPv4→IPv6 direction find their target.

Unlike in stateless NAT64, IPv6 hosts are not reachable from the IPv4 network.



12. Migration steps for transition from IPv4 to IPv6 (77/78)

C.9. Stateful NAT64 ([RFC6052](#) and [RFC6146](#)) (2/3):

Step by step explanation of stateful NAT64 scenario (1/2):

1. DNS request:

The IPv6-only node A makes a DNS request for server.indigoo.com.

2. DNS AAAA record server response:

The DNS server operates as a DNS64 enabled server and synthesizes a mapped IPv6 address consisting of the well-known prefix **64:FF9B::** and the IPv4 address **198.51.100.2** of node C (**64:FF9B:: 198.51.100.2**) as defined in [RFC6052](#).

3. IPv6 packet forwarding of packet to XLAT:

Node A sends the IPv6 packet through the IPv6 network to the IPv6 interface of the NAT64.

The IPv6 network is configured such that packets with the IPv6 destination address 64:FF9B are routed to the stateful NAT64 box.

4. Address mapping, forwarding to IPv4 node C:

The stateful NAT64 box receives the packet and maps the IPv6 header to an IPv4 header according to [RFC6145](#).

Since the IPv6 source address does not contain an IPv4 address in the prefix that could be used by the NAT64 to identify the source when it receives a packet from the IPv4 node C, an additional mapping for the source TCP or UDP port number is necessary.

Thus the NAT64 box creates an entry in the BIB (Binding Information Base) consisting of the following elements:

IPv6 prefix	TCP source port	IPv4 source addr.	Mapped TCP source port	Session lifetime
2001:DB8:1/40	2000	198.0.2.1	3000	2345 seconds
2001:DB8:A/40	4000	198.0.2.1	5000	7200 seconds

Existing entry for node B

New entry for node A

Finally, the NAT64 box forwards the packet to node C.

12. Migration steps for transition from IPv4 to IPv6 (78/78)

C.9. Stateful NAT64 (RFC6052 and RFC6146) (3/3):

Step by step explanation of stateful NAT64 scenario (2/2):

5. Return packet:

Node C returns a packet with swapped source and destination IPv4 addresses and TCP port numbers. The routing in the IPv4 network forwards the packet to the NAT64's IPv4 interface.

6. BIB lookup, reverse mapping and forwarding to source:

The NAT64 box searches the BIB table for an entry with IPv4-dest-address = 192.0.2.1 and destination TCP port number = 3000. If an entry is found, the NAT64 box performs the reverse mapping, i.e. it creates an IPv6 packet with the BIB entry IPv6 address and TCP port number as destination addresses (2001:DB8:1::1 and TCP port number 2000) and 64:FF9B::198.51.100.2 as source IPv6 address along with the source TCP port number extracted from the IPv4 packet.

Finally the NAT64 box forwards the packet to node A.