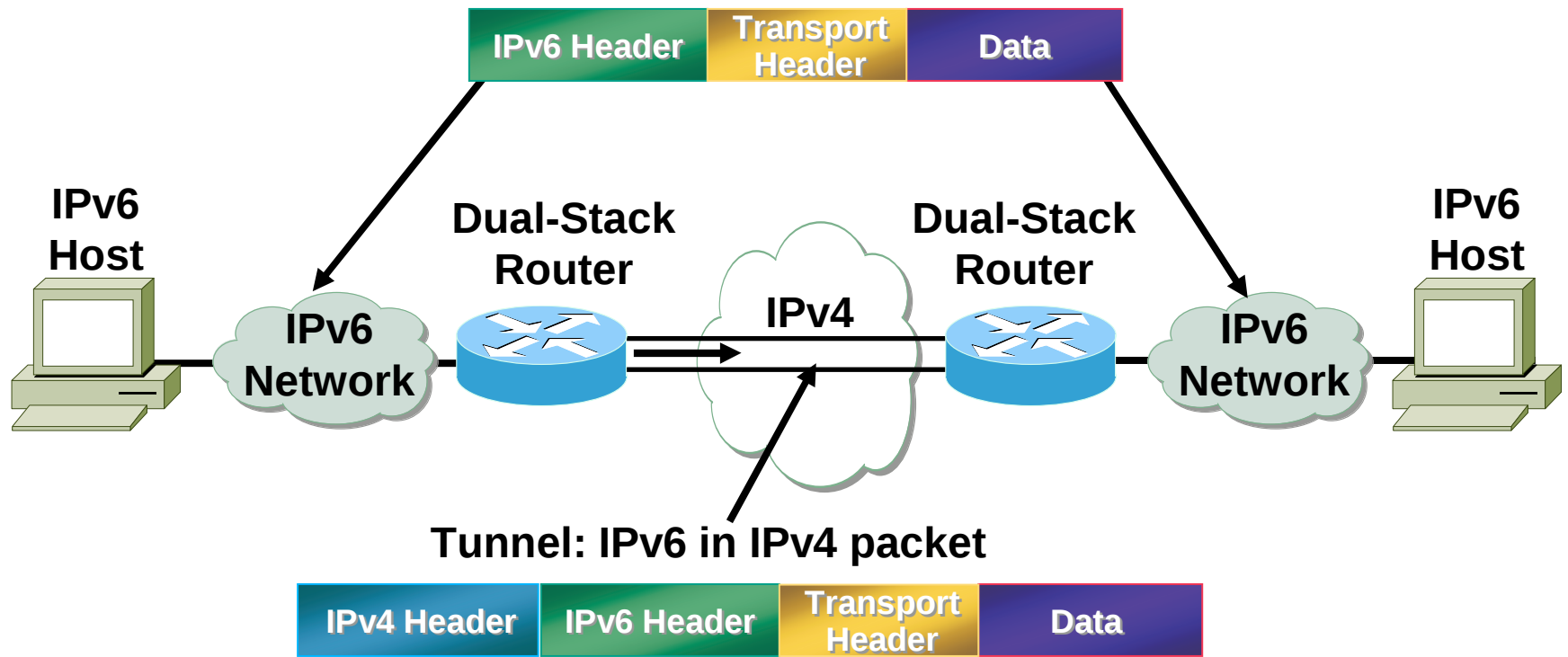


IPv4-to-IPv6 Transition Strategy (RFC 2893)

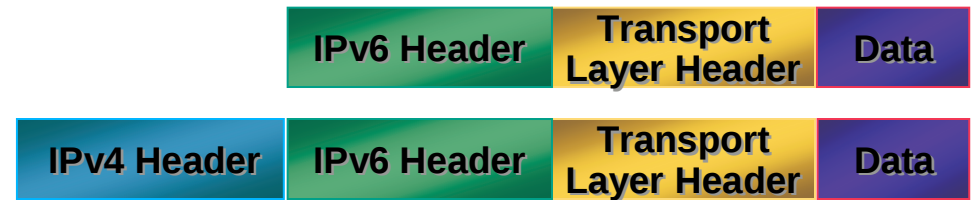
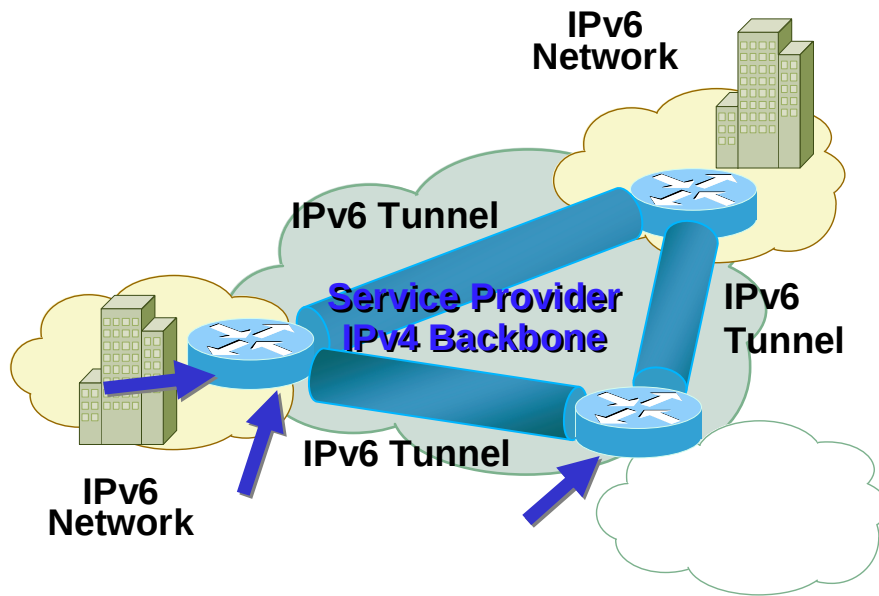
- Dual Stack
 - Reduce the cost invested in transition by running both IPv4/IPv6 protocols on the same machine .
- Tunneling
 - Reduce the cost in wiring by re-using current IPv4 routing infrastructures as a virtual link.
- Translation
 - Allow IPv6 realm to access the rich contents already developed on IPv4 applications

Tunnels of IPv6 over IPv4

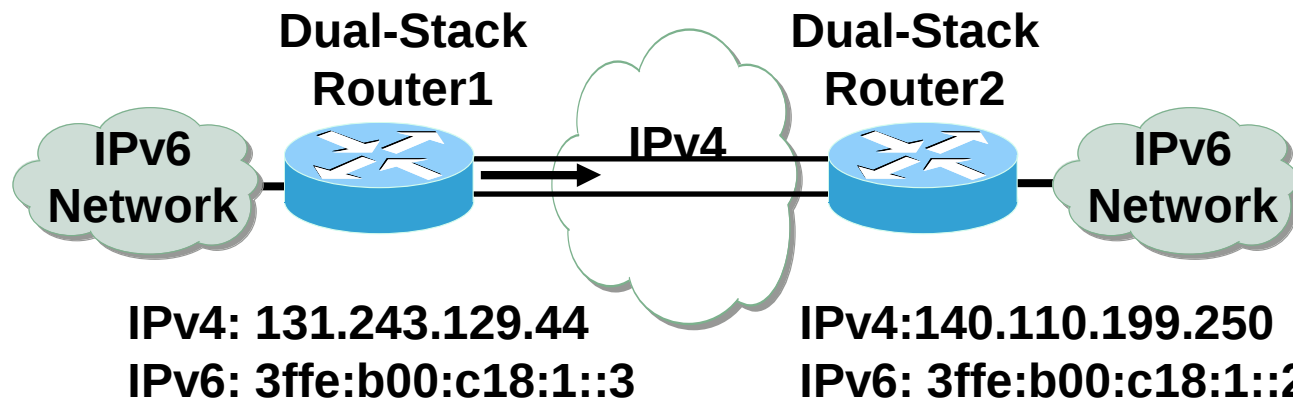


- Encapsulating the IPv6 packet in an IPv4 packet
- Tunneling can be used by routers and hosts

IPv6 Tunneling



Manually Configured Tunnel



router1#

```
interface Tunnel0
  ipv6 address 3ffe:b00:c18:1::3/64
  tunnel source 131.243.129.44
  tunnel destination 140.110.199.250
  tunnel mode ipv6ip
```

router2#

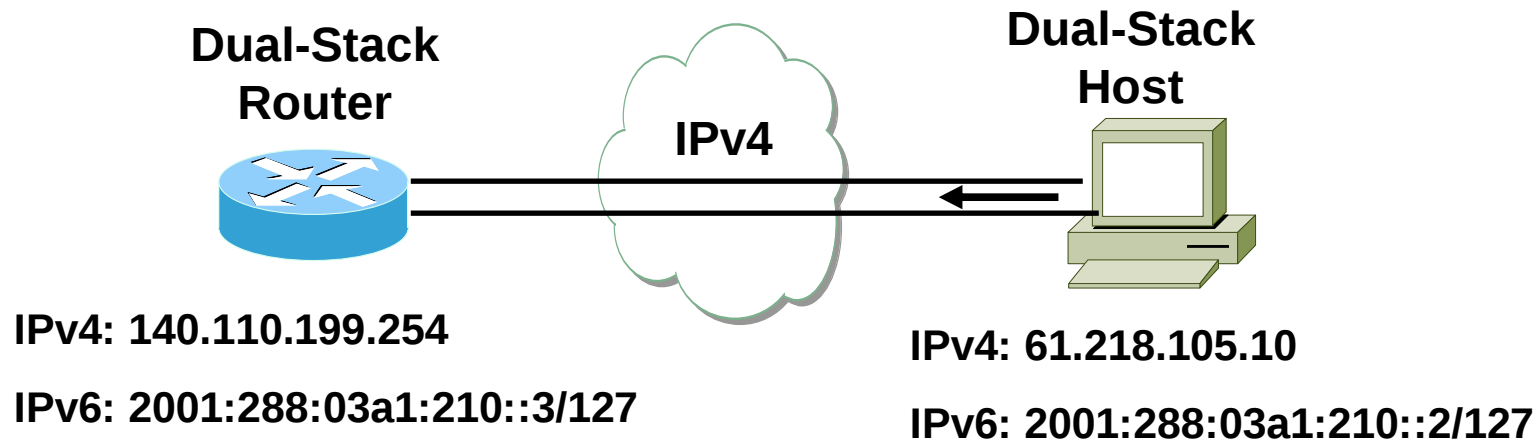
```
interface Tunnel0
  ipv6 address 3ffe:b00:c18:1::2/64
  tunnel source 140.110.199.250
  tunnel destination 131.243.129.44
  tunnel mode ipv6ip
```

- Manually Configured tunnels require:

Dual stack end points

Both IPv4 and IPv6 addresses configured at each end

Manually Configured Tunnel



```
FreeBSD4.7#  
gifconfig gif0 61.218.105.10 140.110.199.254  
ifconfig gif0 inet6 2001:288:03a1:210::2 2001:288:3a1:210::3 prefixlen 128
```

Linux Tunnel

/etc/sysconfig/network-scripts/ifcfg-sit1

```
DEVICE=sit1
BOOTPROTO=none
ONBOOT=yes
IPV6INIT=yes
#Remote end-ISP IPv4 addr
IPV6TUNNELIPv4=140.110.199.250
#Yourself IPv6 tunnel addr from ISP
IPV6ADDR=2001:288:3A1:210::2/127
```

ifup sit1

Windows XP Tunnel

- netsh interface ipv6
 - add v6v4tunnel “T1” 140.113.131.23 140.113.87.100
 - Syntax: add v6v4tunnel [[interface=]*String*] localIPv4Address remoteIPv4Address
 - add address “T1” 2001:238:F88:B::30
 - add route 2001:238:F88:B::30/127 “T1”
- Now you can *ping* the remote tunnel endpoint 2001:238:F88:B::31
- Use Ethereal to capture packets with filter “ip host 140.113.87.100”.

Tunnel Packets

<capture> - Ethereal

File Edit Capture Display Tools Help

No. .	Time	Source	Destination	Protocol	Info
1	0.000000	fe80::280:c8ff:fec0:1	ff02::9	RIPng version 1	Response
2	0.689644	2001:238:f88:b::30	2001:238:f88:b::31	ICMPv6	Echo request
3	0.882896	2001:238:f88:b::31	2001:238:f88:b::30	ICMPv6	Echo reply
4	1.691907	2001:238:f88:b::30	2001:238:f88:b::31	ICMPv6	Echo request
5	1.694533	2001:238:f88:b::31	2001:238:f88:b::30	ICMPv6	Echo reply
6	2.693848	2001:238:f88:b::30	2001:238:f88:b::31	ICMPv6	Echo request
7	2.696218	2001:238:f88:b::31	2001:238:f88:b::30	ICMPv6	Echo reply
8	3.695788	2001:238:f88:b::30	2001:238:f88:b::31	ICMPv6	Echo request
9	3.698970	2001:238:f88:b::31	2001:238:f88:b::30	ICMPv6	Echo reply
10	18.016823	fe80::280:c8ff:fec0:1	ff02::9	RIPng version 1	Response
11	25.008359	fe80::280:c8ff:fec0:1	ff02::9	RIPng version 1	Response

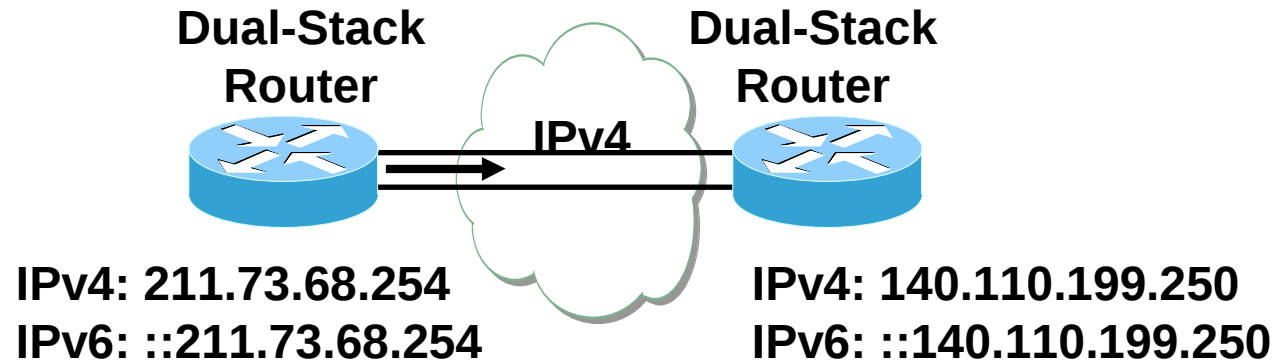
☒ Frame 2 (114 bytes on wire, 114 bytes captured)
☒ Ethernet II, Src: 00:e0:18:ea:f7:2e, Dst: 00:20:9c:08:e8:7f
☒ Internet Protocol, Src Addr: 140.113.131.23 (140.113.131.23), Dst Addr: 140.113.87.100 (140.113.87.100)
☒ Internet Protocol version 6
 version: 6
 Traffic class: 0x00
 Flowlabel: 0x000000
 Payload length: 40
 Next header: ICMPv6 (0x3a)
 Hop limit: 128
 Source address: 2001:238:f88:b::30
 Destination address: 2001:238:f88:b::31
☒ Internet Control Message Protocol v6

```

0000  00 20 9c 08 e8 7f 00 e0 18 ea f7 2e 08 00 45 00  . . . . .E.
0010  00 64 85 16 00 00 80 29 c1 fc 8c 71 83 17 8c 71  .d. . . . .) ...q...q
0020  57 64 60 00 00 00 00 28 3a 80 20 01 02 38 0f 88  wd` . . . . ( :. .8..
0030  00 0b 00 00 00 00 00 00 00 30 20 01 02 38 0f 88  . . . . .0 .8..
0040  00 0b 00 00 00 00 00 00 00 31 80 00 70 f7 00 00  . . . . .1..p...
  
```

Filter: / Reset Apply File: <capture> Drops: 0

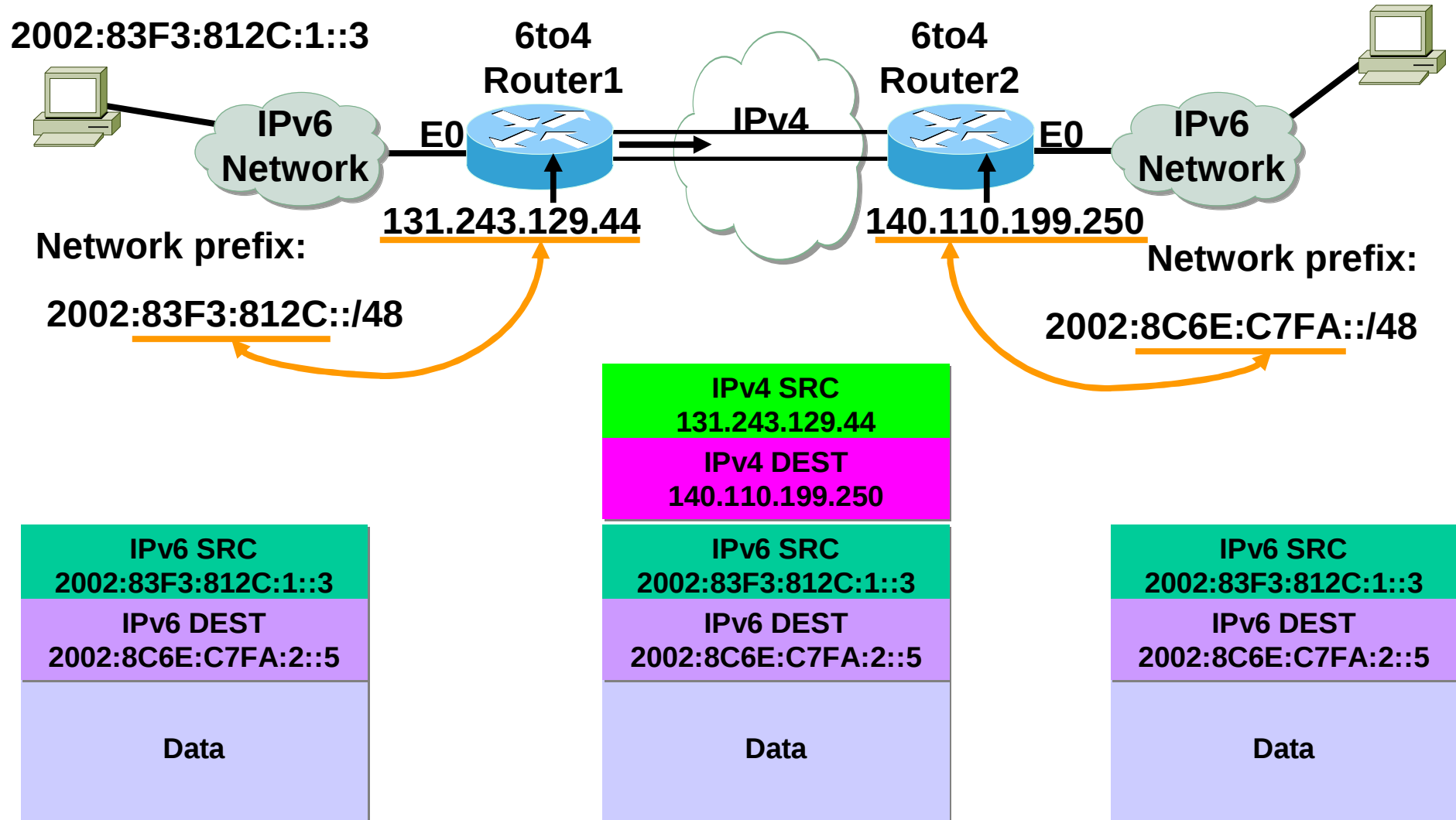
IPv4 Compatible Tunnel (RFC 2893)



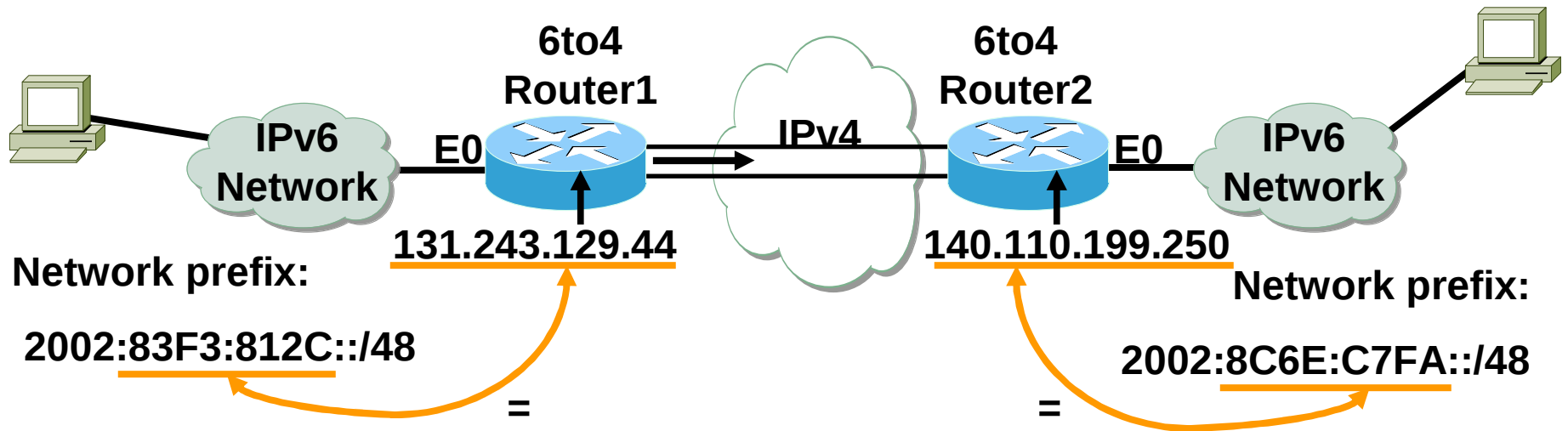
- IPv4-compatible addresses are easy way to autotunnel, but it:
 - May be deprecated soon
 - Consumes IPv4 addresses

6to4 Tunnel (RFC 3056)

2002:8C6E:C7FA:2::5



6to4 Tunnel



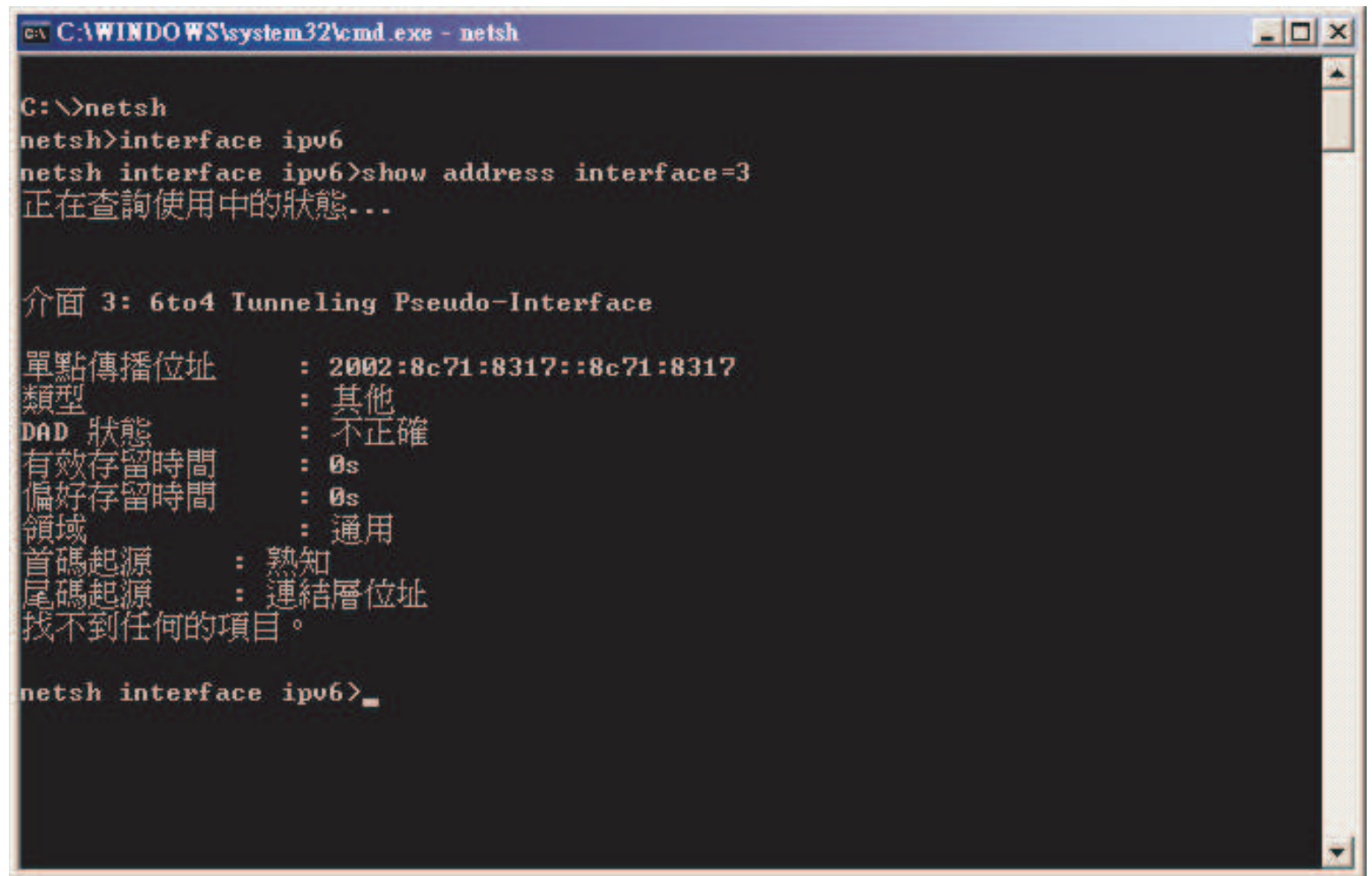
6to4 Tunnel:

- Is an automatic tunnel method
- Gives a prefix to the attached IPv6 network
- 2002::/16 assigned to 6to4
- Requires one global IPv4 address on each site

```
router2#  
interface Ethernet0  
 ip address 140.110.199.250 255.255.255.0  
 ipv6 address 2002:8C6E:C7FA:1::/64 eui-64  
interface Tunnel0  
 no ip address  
 ipv6 unnumbered Ethernet0  
 tunnel source Ethernet0  
 tunnel mode ipv6ip 6to4  
  
ipv6 route 2002::/16 Tunnel0
```

6to4 Tunnel in Windows XP

- 6to4 Tunnel is enabled in Windows XP by default.



```
C:\WINDOWS\system32\cmd.exe - netsh

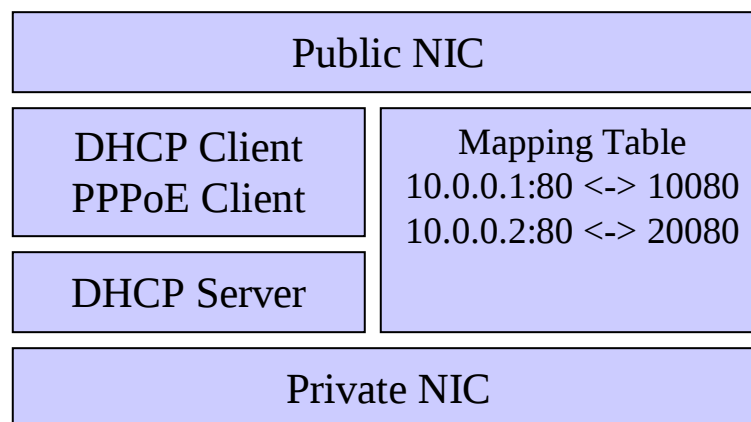
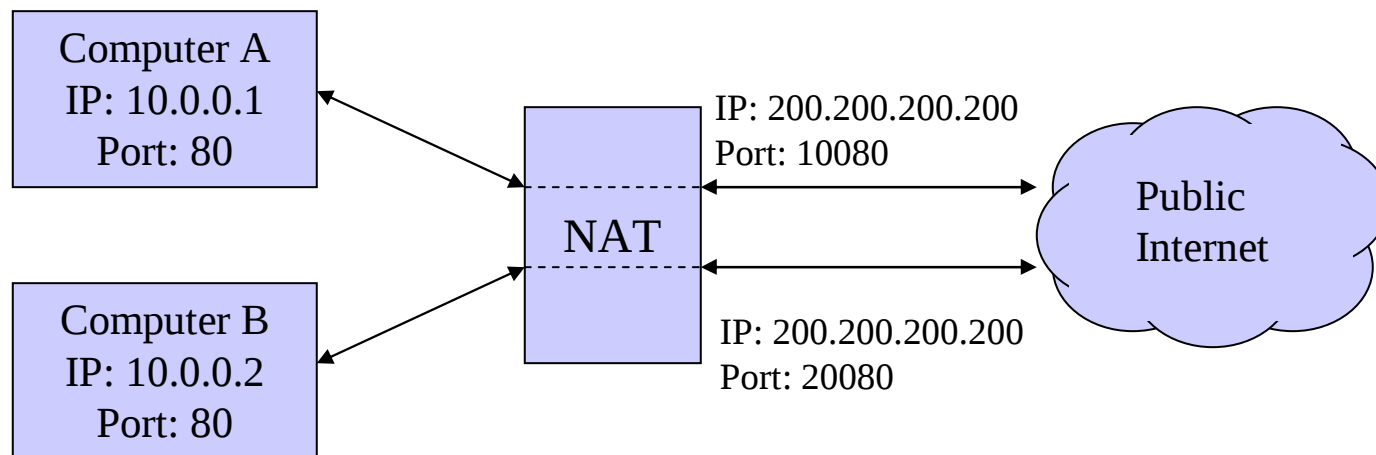
C:\>netsh
netsh>interface ipv6
netsh interface ipv6>show address interface=3
正在查詢使用中的狀態...

介面 3: 6to4 Tunneling Pseudo-Interface

單點傳播位址      : 2002:8c71:8317::8c71:8317
類型              : 其他
DAD 狀態          : 不正確
有效存留時間      : 0s
偏好存留時間      : 0s
領域              : 通用
首碼起源          : 熟知
尾碼起源          : 連結層位址
找不到任何的項目。

netsh interface ipv6>
```

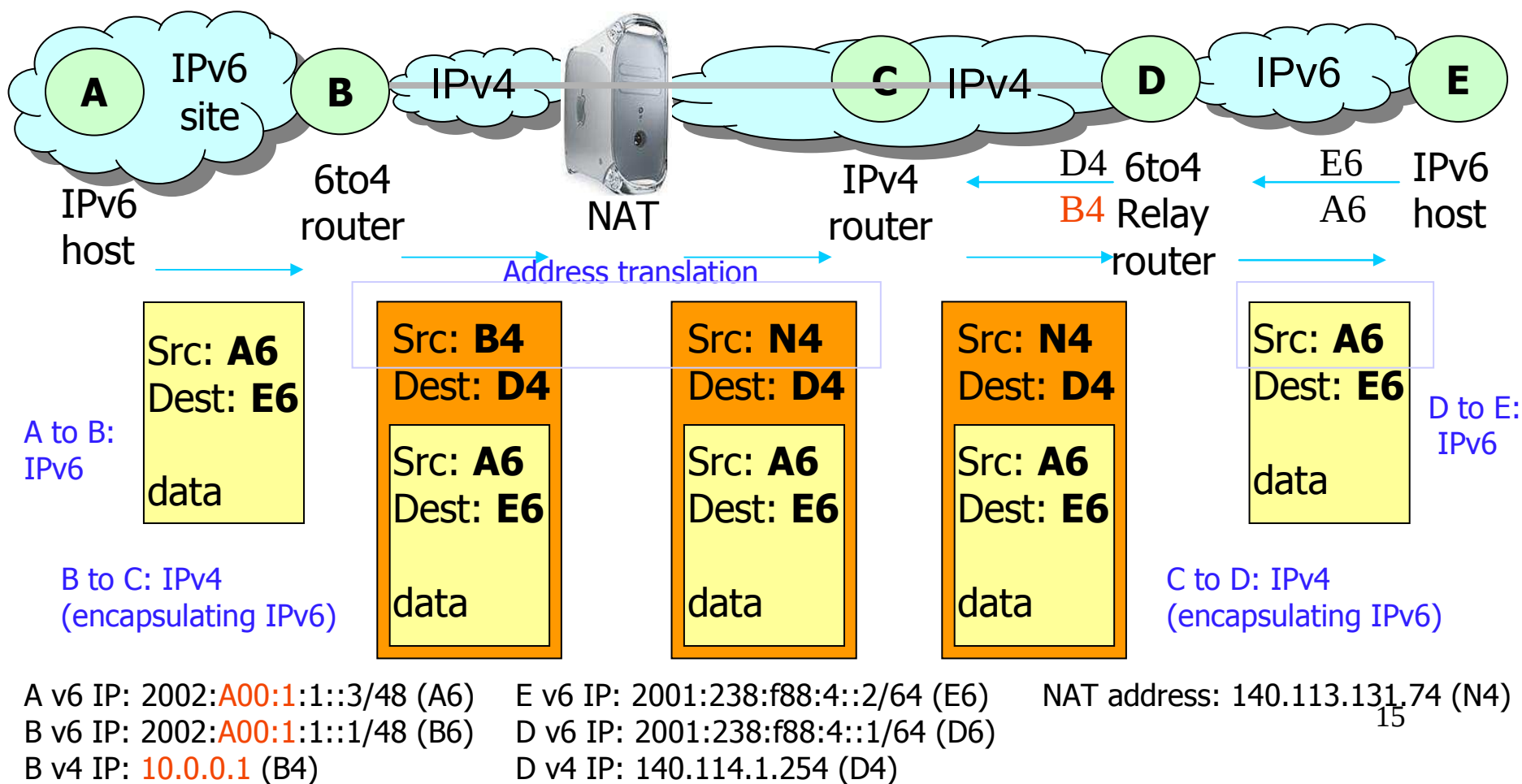
Network Address Translator



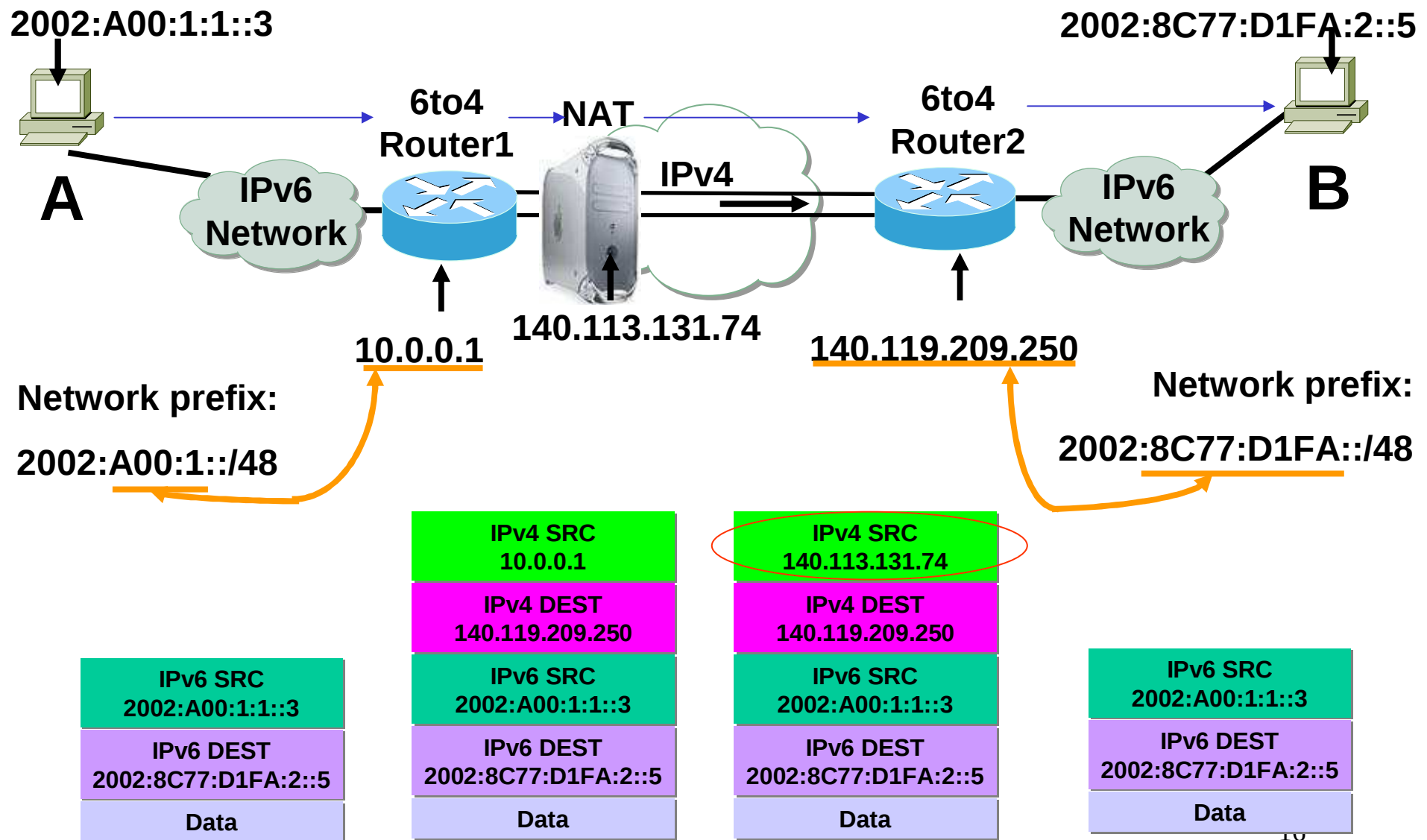
IPv6 tunneling problem

- It does not work when the IPv4 address is not globally routable

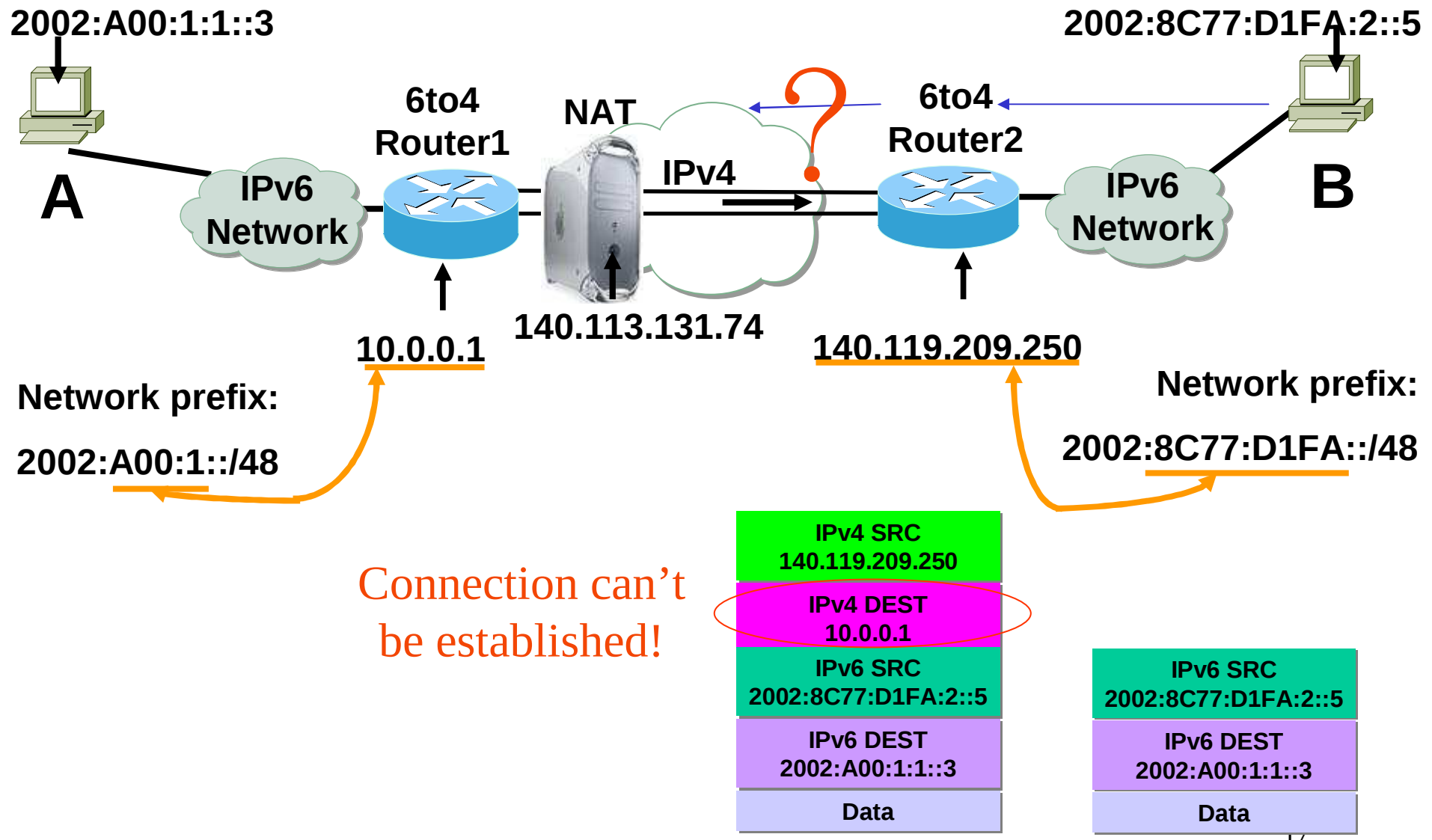
B4 is a private address!



IPv6 Tunneling Problem [1/2]



IPv6 Tunneling Problem [2/2]



Teredo Service (RFC 4380)

- Allow hosts behind NAT to access IPv6 without modifying NAT. It contains three basic components:
 - Teredo Client
 - a node wants to gain access to the IPv6 Internet.
 - Teredo Server
 - helper to provide IPv6 connectivity to Teredo clients.
 - Teredo Relay
 - an IPv6 router that can receive traffic from IPv6 realm to Teredo clients and vice versa.

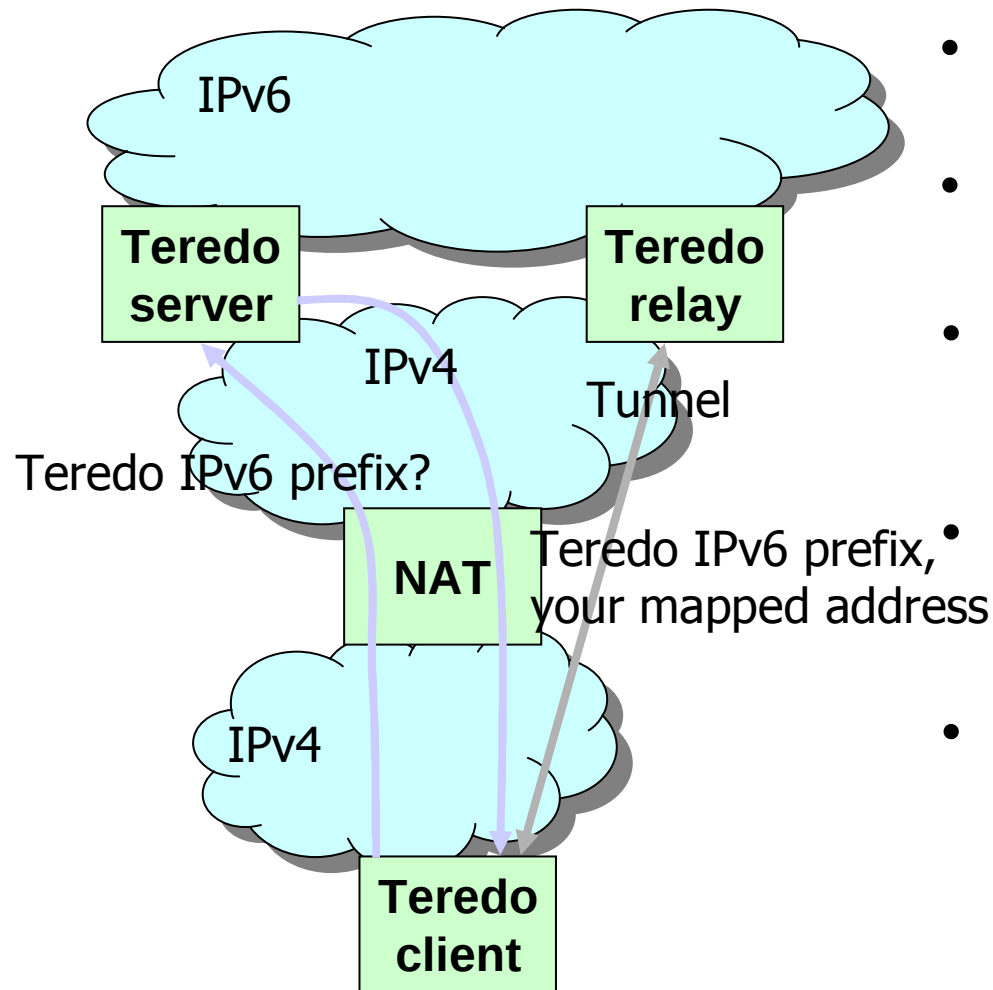
Teredo service

- To allow hosts behind NAT to access IPv6, without modifying NAT.
 - Teredo is not a long term solution
 - If NAT also supports IPv6 routing, the problem of NAT traversal will disappear.

Teredo definitions

- Teredo client
 - A node wants to gain access to the IPv6 Internet.
- Teredo server
 - helper to provide IPv6 connectivity to Teredo clients.
- Teredo relay
 - An IPv6 router that can receive traffic destined to Teredo clients and forward it to Teredo client.
- Teredo bubble
 - minimal IPv6 packet, made of an IPv6 header and null payload, no Next Header.
- Teredo service
 - The transmission of IPv6 packets over UDP.

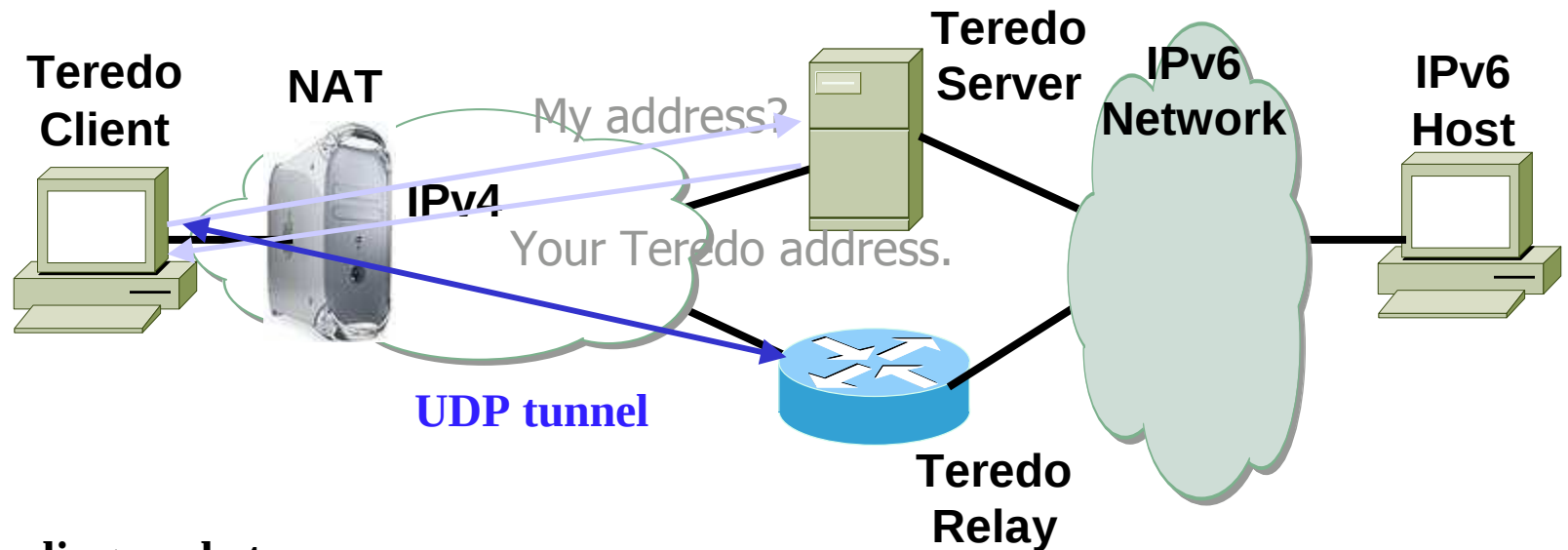
Operation model



- A client has pre-configured server location.
- A client gets IPv6 prefix from the Teredo server.
- Teredo server is stateless. Traffic goes directly between the relay router and the client.
- Teredo Relay announces reachability of Teredo prefix on IPv6 realm.
- Relay and Client maintain peer list to avoid sending Teredo message too often.

Teredo Operation Model

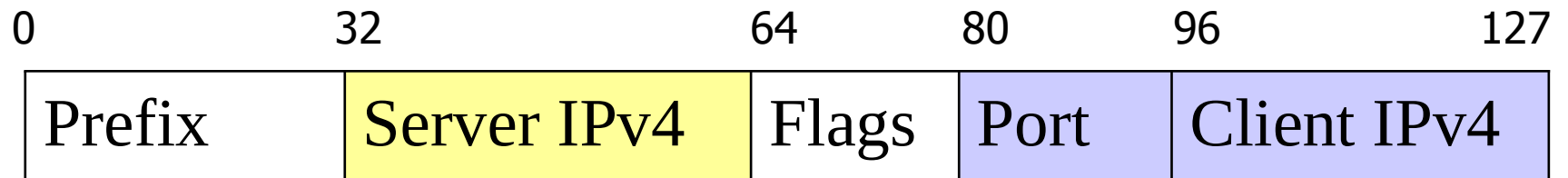
- Teredo Client gets its Teredo IPv6 address from Teredo Server.
- Use Teredo Relay as relay router.



Tunneling packet

IPv4 Header	UDP Header	Teredo Header	IPv6 packet
-------------	------------	---------------	-------------

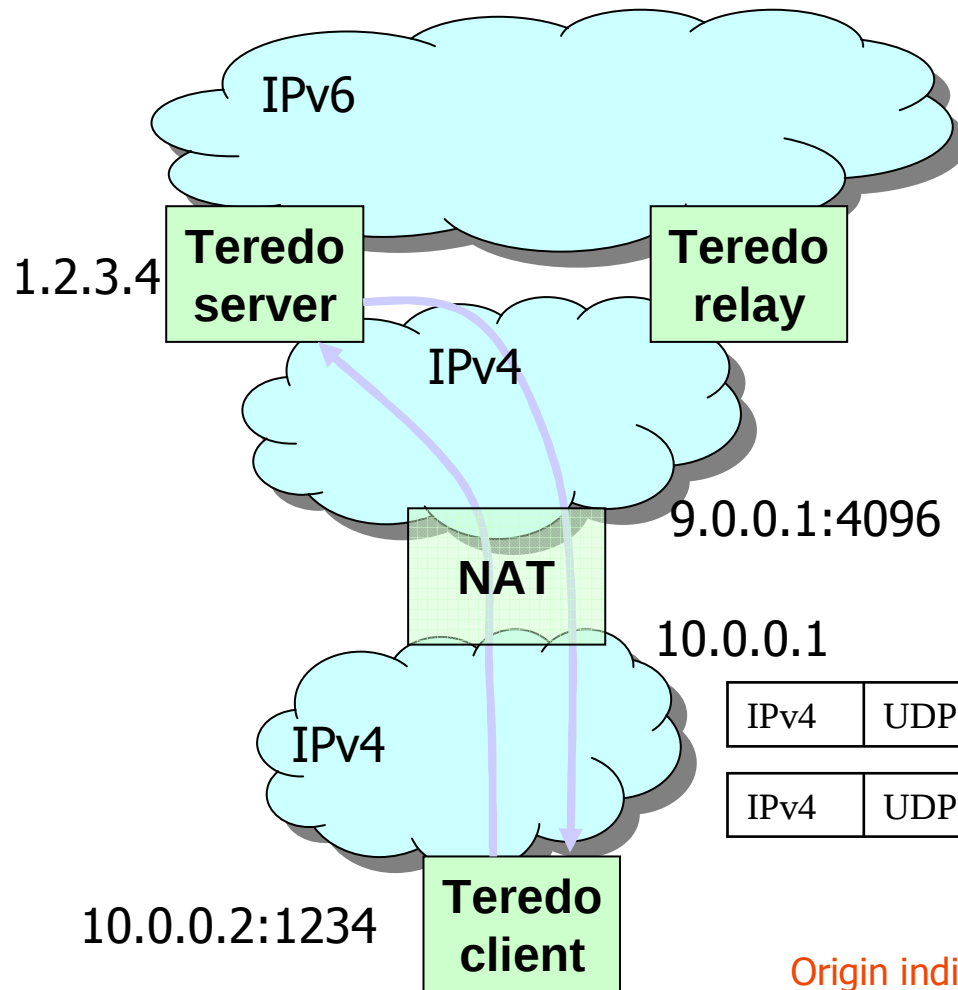
Teredo address encoding



- Prefix: the 32 bit Teredo service prefix.
 - 2001:0000::/32
- Server IPv4: the IPv4 address of a Teredo server.
- Flags: a set of 16 bits that document type of address and NAT.
 - 16 bits flag: “C000000UG000000000”
 - C=1 if NAT is cone.
 - UG should set to “00”.
- Port: the obfuscated "mapped UDP port" of the client
- Client IPv4: the obfuscated "mapped IPv4 address" of a client

Obfuscated: XOR every bits in the field with 1, prevent over-genius NAT's translation.

Obtaining an address(1/2)



- Teredo client sends a UDPv4 tunneled IPv6 Router Solicitation to the Teredo server.
- Teredo server replies UDPv4 tunneled IPv6 Router Advertisement with origin indication.

IPv4	UDP	IPv6 RS
------	-----	---------

IPv4	UDP	Origin indication	IPv6 RA
------	-----	-------------------	---------

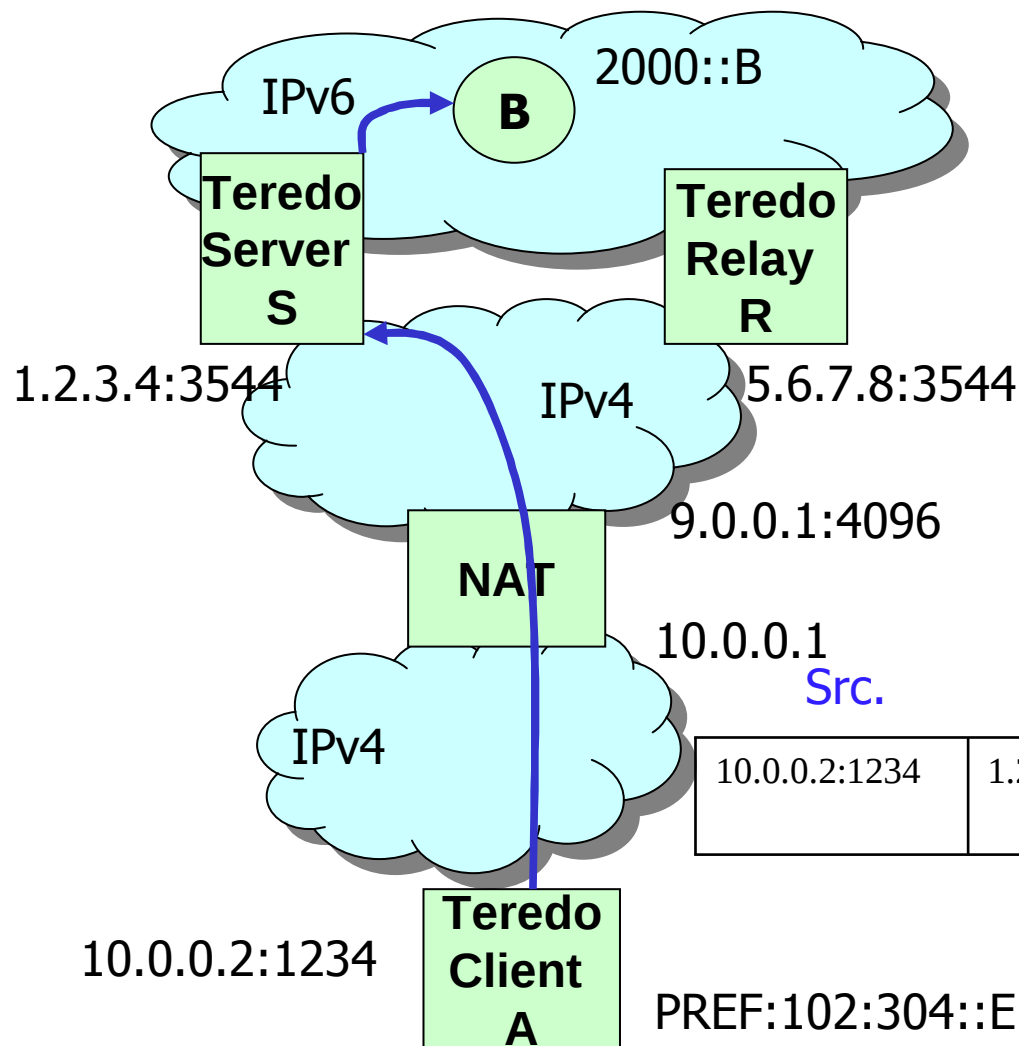
0x00	0x00	mapped port #
mapped IPv4 address		

Origin indication
format

Obtaining an address(2/2)

- Client get mapped address/port from origin indication
 - Mapped address: 9.0.0.1:4096
 - Already known server IP: 1.2.3.4
- Generated Teredo IPv6 address
 - Prefix: 2001:0000::/32
 - Server: 0x0102:0304 (Teredo server IP address: 1.2.3.4)
 - Flags: 0x8000 (cone NAT)
 - Obfuscated Port: 0xEFFF ($=0xFFFF \oplus 4096$)
 - Obfuscated Address: 0xF6FF:FFFE ($=0xFFFF:FFFF \oplus 9.0.0.1$)
 - Teredo IPv6 Address: 2001:0000:102:304:8000:EFFF:F6FF:FFFE
- Must keep alive address mapping on NAT
 - Default refresh interval: 30 seconds.

Packet from Teredo node to IPv6 node (1/3)

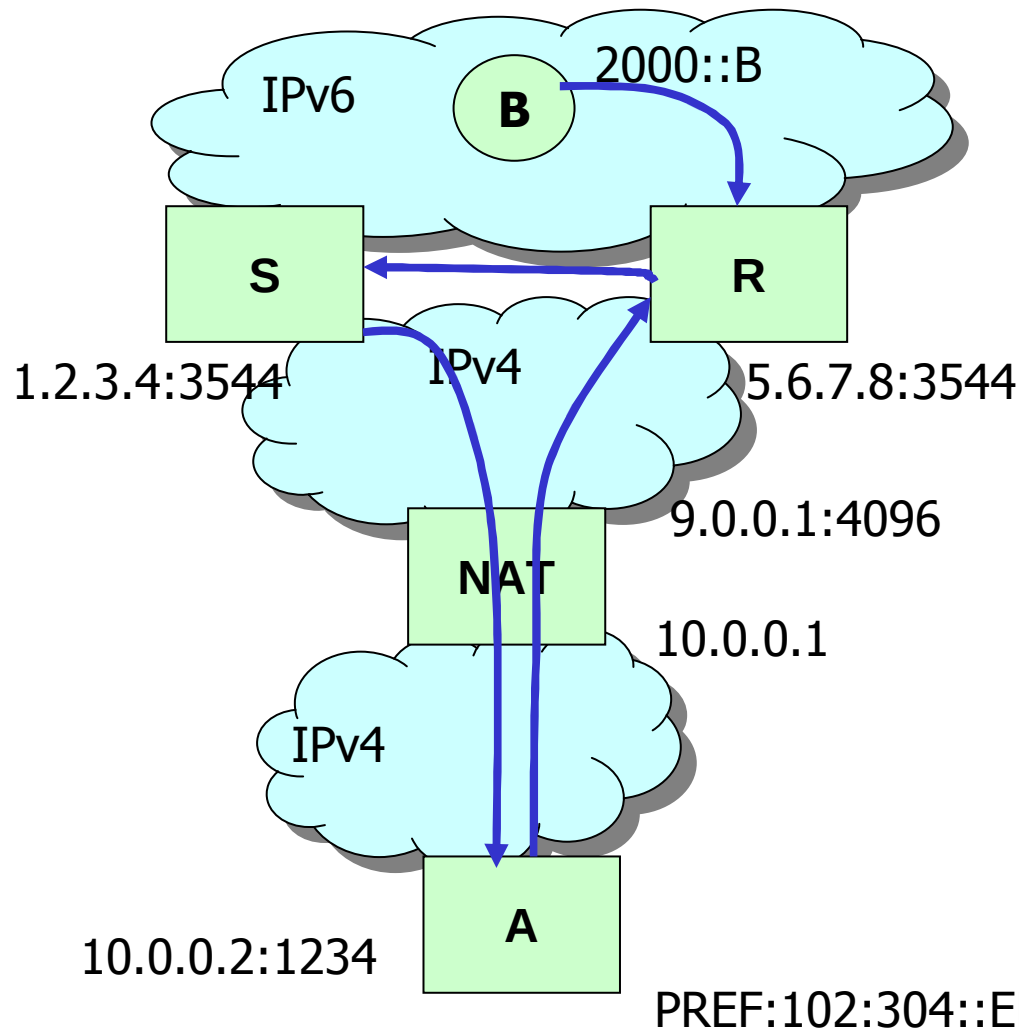


- A does not know which relay will be chosen by B.
- A sends ICMPv6 “echo request” toward B.
- S forwards “echo request” to IPv6 realm.

		IPv6 Src.	IPv6 dest.
10.0.0.2:1234	1.2.3.4:3544	PREF:102:304::E FFF:F6FF:FFFE	2000::B
		PREF:102:304::E FFF:F6FF:FFFE	2000::B

PREF:102:304::EFFF:F6FF:FFFE

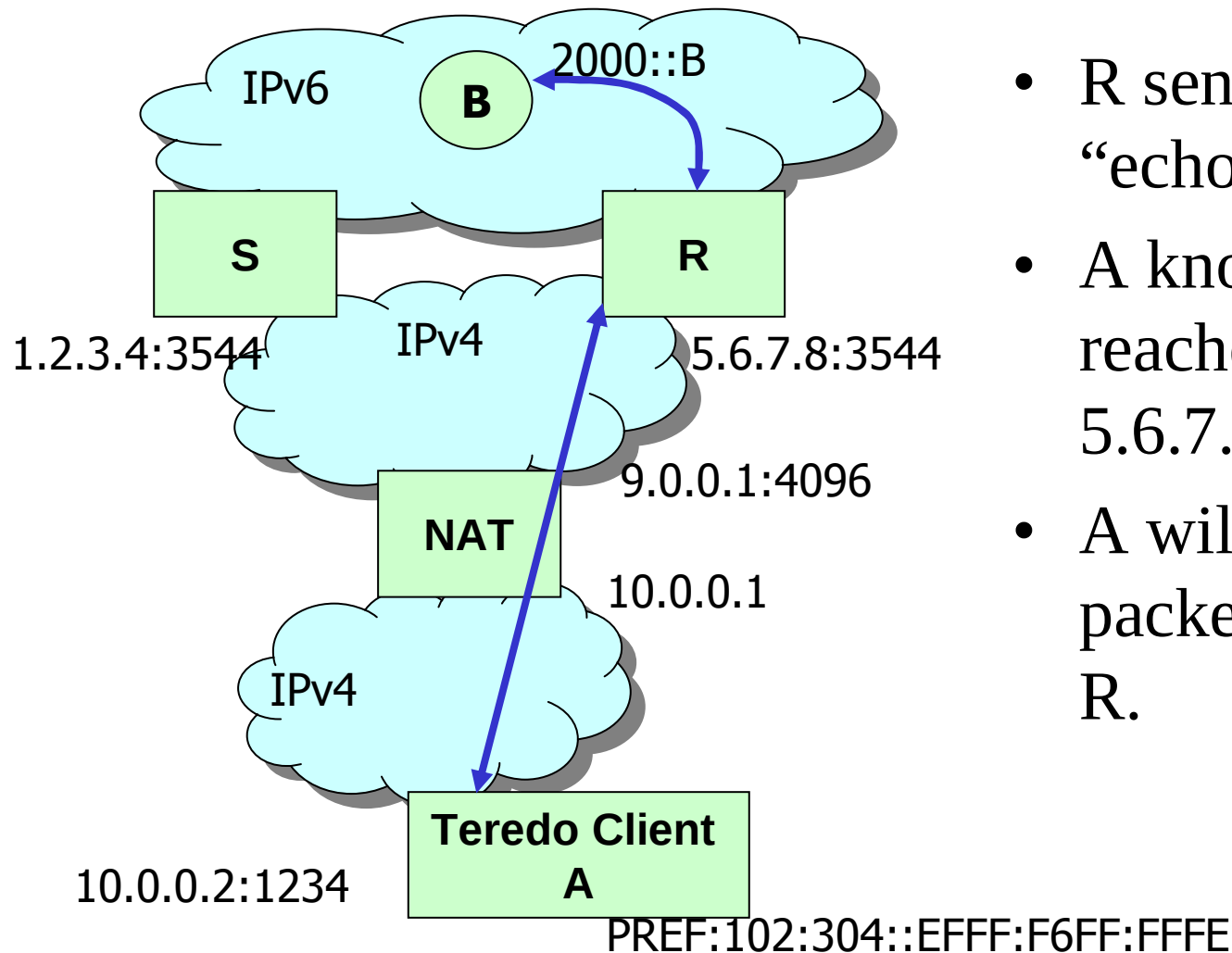
Packet from Teredo node to IPv6 node (2/3)



- B sends the “echo reply” back to Teredo Client.
- The IPv6 packet will be queued by Teredo Relay.
- If Teredo Client is behind a restricted NAT, a bubble must be sent to Teredo Server.

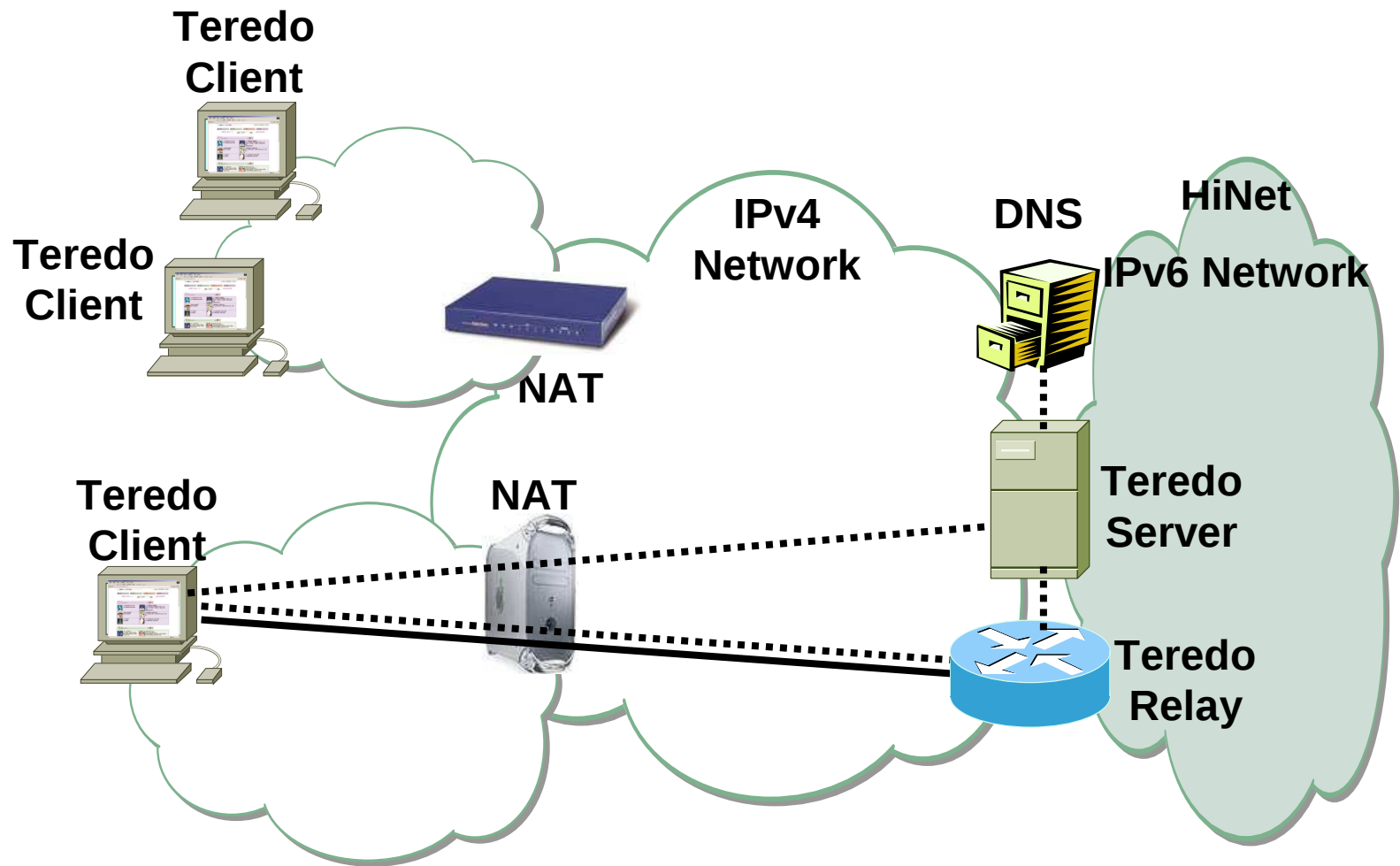
IPv6 Src.	IPv6 dest.
2000::B	PREF:102:304:: EFFF:F6FF:FFF
E	

Packet from Teredo node to IPv6 node (3/3)

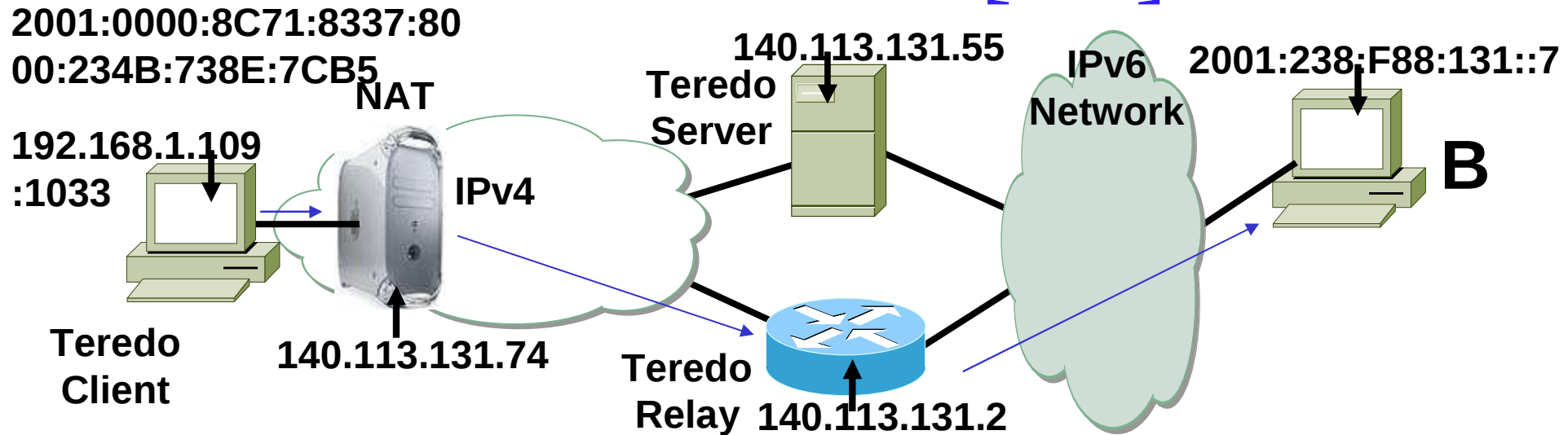


- R sends the queued “echo reply” to A.
- A knows B can be reached through address 5.6.7.8:3544.
- A will send all further packets directly through R.

Trial of Teredo in NCTU



Teredo Tunnel [1/2]

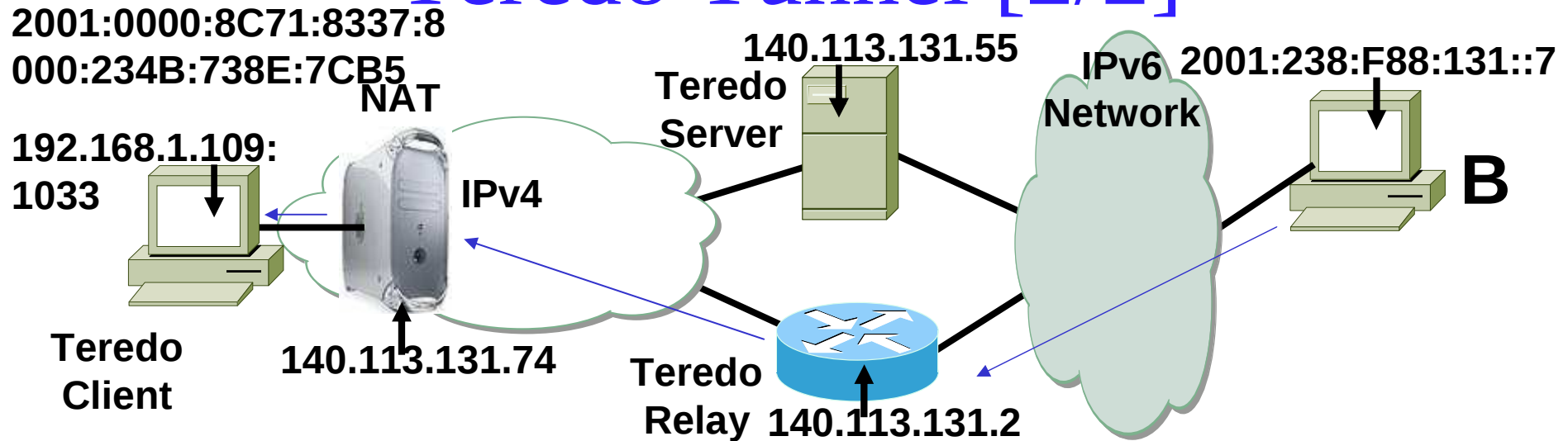


IPv4 SRC 10.0.0.2
IPv4 DEST 140.113.131.2
UDP SRC 1033
UDP DEST 3544
Teredo Header
IPv6 SRC 2001:0000:8C71:8337:80 00:234B:738E:7CB5
IPv6 DEST 2001:238:F88:131::7
Data

IPv4 SRC 140.113.131.74
IPv4 DEST 140.113.131.2
UDP SRC 56500
UDP DEST 3544
Teredo Header
IPv6 SRC 2001:0000:8C71:8337:80 00:234B:738E:7CB5
IPv6 DEST 2001:238:F88:131::7
Data

IPv6 SRC 2001:0000:8C71:8337:80 00:234B:738E:7CB5
IPv6 DEST 2001:238:F88:131::7
Data

Teredo Tunnel [2/2]



IPv4 SRC 140.113.131.2
IPv4 DEST 192.168.1.109
UDP SRC 3544
UDP DEST 1033
Teredo Header
IPv6 SRC 2001:238:F88:131::7
IPv6 DEST 2001:0000:8C71:8337:80 00:234E:738E:7CB5
Data

IPv4 SRC 140.113.131.2
IPv4 DEST 140.113.131.74
UDP SRC 3544
UDP DEST 56500
Teredo Header
IPv6 SRC 2001:238:F88:131::7
IPv6 DEST 2001:0000:8C71:8337:80 00:234E:738E:7CB5
Data

IPv6 SRC 2001:238:F88:131::7
IPv6 DEST 2001:0000:8C71:8337:80 00:234B:738E:7CB5
Data

Protocol Decoder in Ethereal

The screenshot shows the Ethereal (Wireshark) interface with a packet capture file named 'teredo.cap'. The packet list shows three ICMPv6 Echo requests and one Echo reply. The packet details pane for the selected Echo reply (Frame 9) shows the encapsulation details, including the source and destination addresses. The destination address is highlighted with a red circle and annotated with '= 140.113.131.74' and 'Port: 56500'.

No.	Time	Source	Destination	Protocol	Info
8	49.6	3ffe:831f:cf2e:e420:	2001:238:f88:131:	ICMPv6	Echo request
9	50.0	2001:238:f88:131:200	3ffe:831f:cf2e:e4	ICMPv6	Echo reply
10	50.0	3ffe:831f:cf2e:e420:	2001:238:f88:131:	ICMPv6	Echo request

Frame 9 (94 bytes on wire, 94 bytes captured)

- Ethernet II, Src: 00:90:cc:4f:d0:80, Dst: 08:00:46:47:1f:ad
- Internet Protocol, Src Addr: 210.242.96.200 (210.242.96.200), Dst Addr: 192.168.1.109
- User Datagram Protocol, Src Port: 1033 (1033), Dst Port: 3544 (3544)
- TEREDO simple encapsulation
- Internet Protocol version 6
 - Version: 6
 - Traffic class: 0x00
 - Flowlabel: 0x00000
 - Payload length: 12
 - Next header: ICMPv6 (0x3a)
 - Hop limit: 58
 - Source address: 2001:238:f88:131:200:b4ff:fe70:6a15
 - Destination address: 3ffe:831f:cf2e:e420:8000:234b:738e:7cb5 = 140.113.131.74
- Internet Control Message Protocol v6

Port: 56500

Offset	Hex	ASCII
0000	08 00 46 47 1f ad 00 90 cc 4f d0 80 08 00 45 00	..FG.... .O....E.
0010	00 50 86 69 00 00 76 11 c8 63 d2 f2 60 c8 c0 a8	.P.i..v. .c..`...
0020	01 6d 04 09 0d d8 00 3c 5d c4 60 00 00 00 00 0c	.m.....<].`.....
0030	3a 3a 20 01 02 38 0f 88 01 31 02 00 b4 ff fe 70	:: ..8.. .1....p
0040	6a 15 3f fe 83 1f cf 2e e4 20 80 00 23 4b 73 8e	j.?..... . ..#Ks.

Filter: / Reset Apply File: teredo.cap

Conclusions

- *Tunneling* is a useful technique to establish connectivity between IPv6 sites even though they don't have direct links between each other.
- Many users get private IPv4 address from their service providers, such as WLAN and GPRS. These users have difficulty in creating IPv6 tunnels.
- Before all NAT devices can be upgraded to support IPv6, Teredo service is useful for ISPs to provide IPv6 access to their users behind NAT.