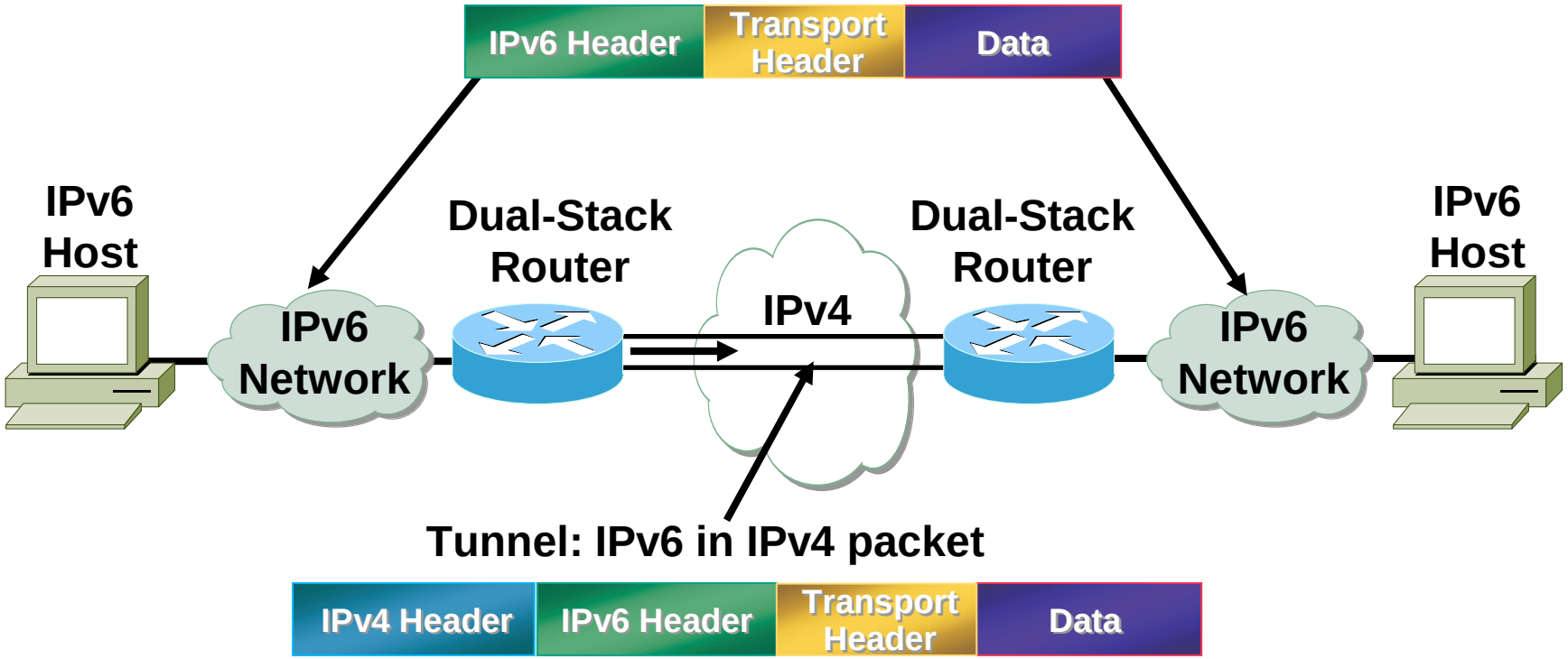
IPv4–to–IPv6 Transition Strategy (RFC 2893)

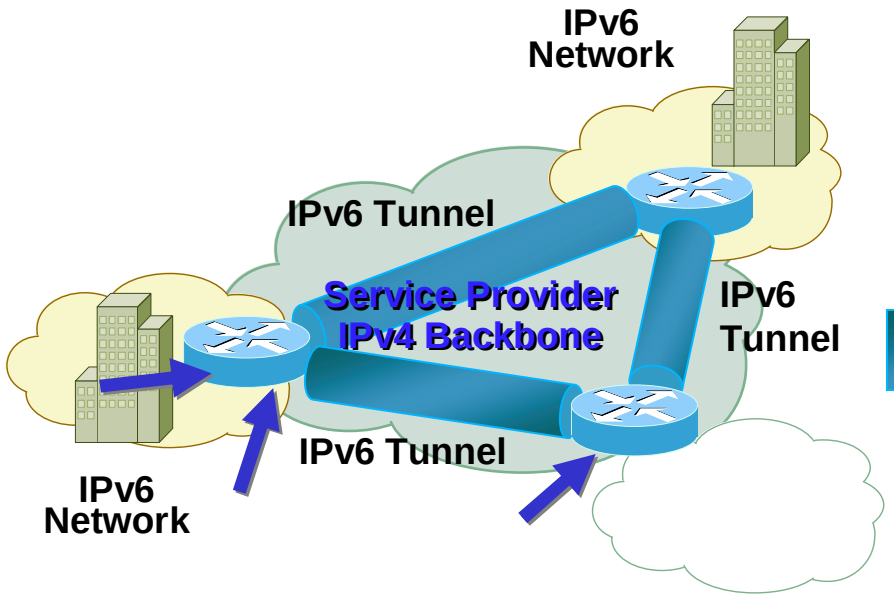
- Dual Stack
  - Reduce the cost invested in transition by running both IPv4/IPv6 protocols on the same machine .
- Tunneling
  - Reduce the cost in wiring by re-using current IPv4 routing infrastructures as a virtual link.
- Translation
  - Allow IPv6 realm to access the rich contents already developed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

Tunnels of IPv6 over IPv4

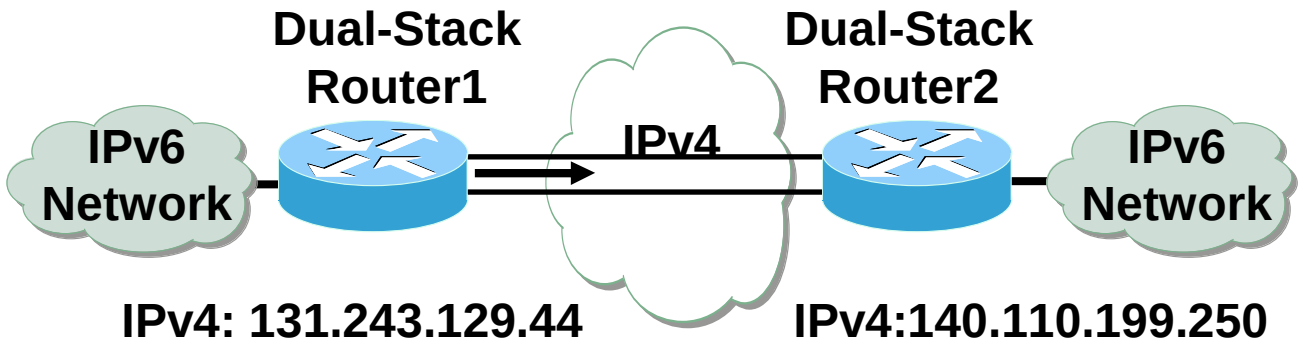


- Encapsulating the IPv6 packet in an IPv4 packet
- Tunneling can be used by routers and hosts

IPv6 Tunneling



Manually Configured Tunnel

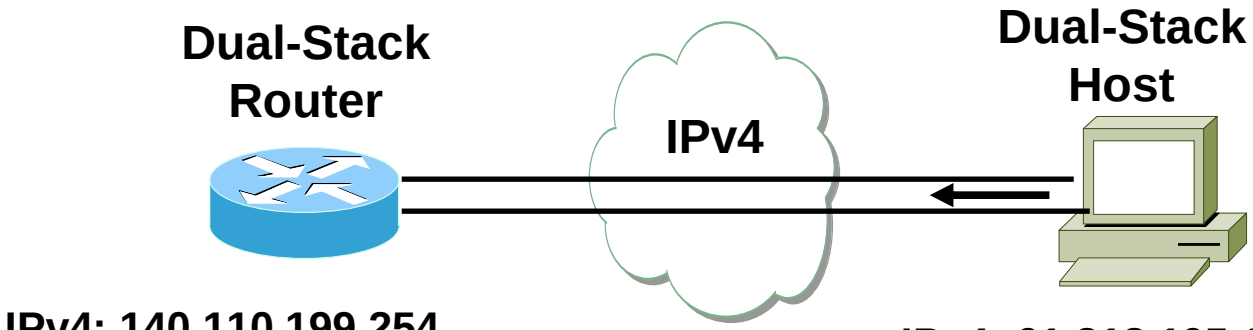



```
router1#

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



IPv4: 140.110.199.254

IPv6: 2001:288:03a1:210::3/127

IPv4: 61.218.105.10

IPv6: 2001:288:03a1:210::2/127

```
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

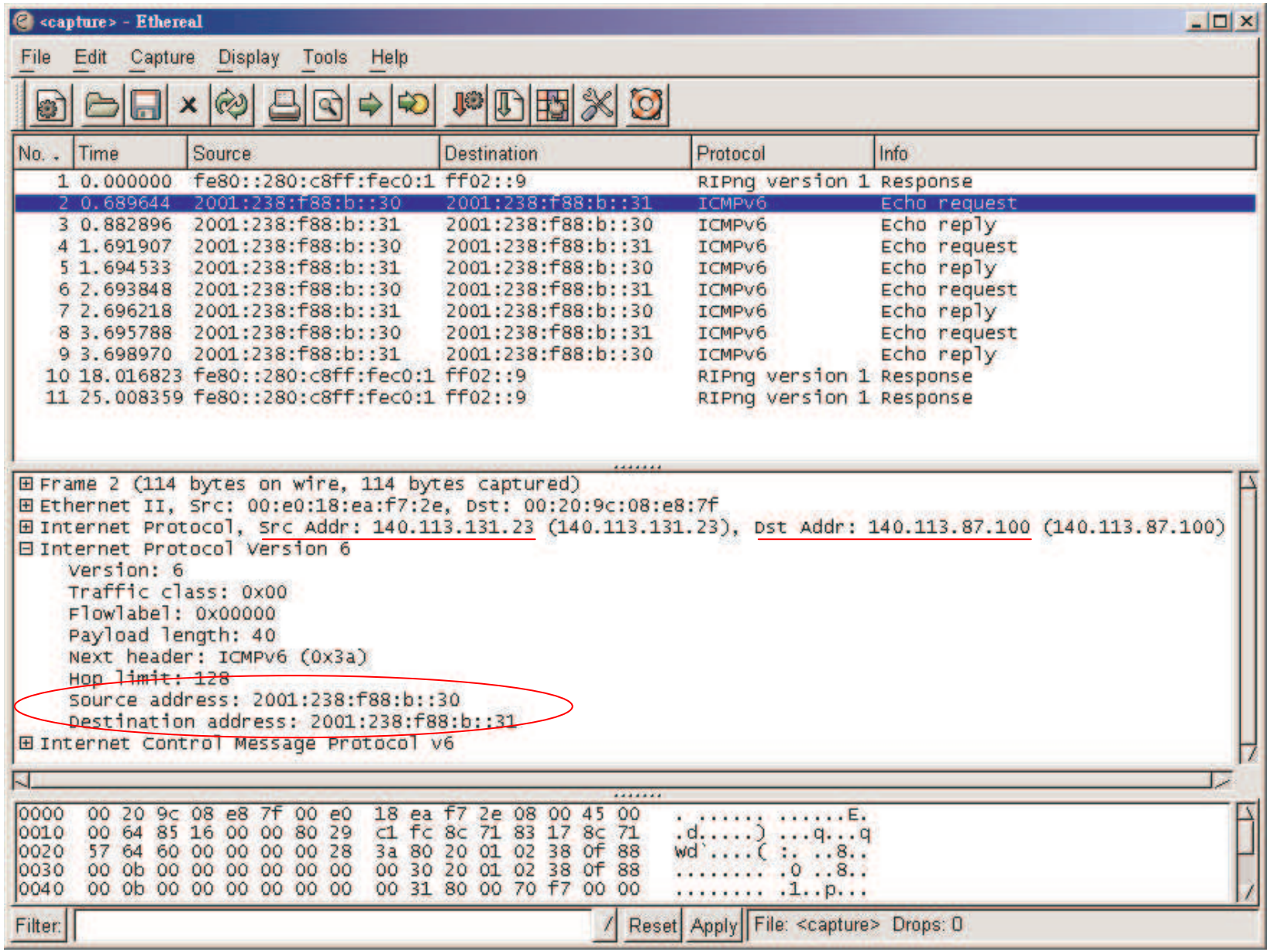
```
DEVICE=sit1
BOOTPROTO=none
ONBOOT=yes
IPV6INIT=yes
#Remote end-ISP IPv4 addr
IPV6TUNNELIPV4=140.110.199.250
#Yourself IPv6 tunnel addr from ISP
IPV6ADDR=2001:288:3A1:210::2/127
```

ifup sit1

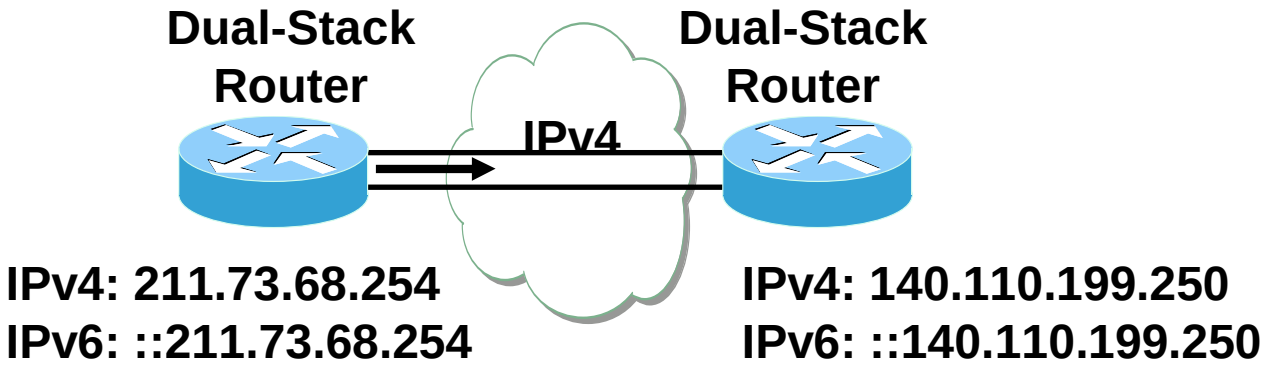
Windows XP Tunnel

- 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

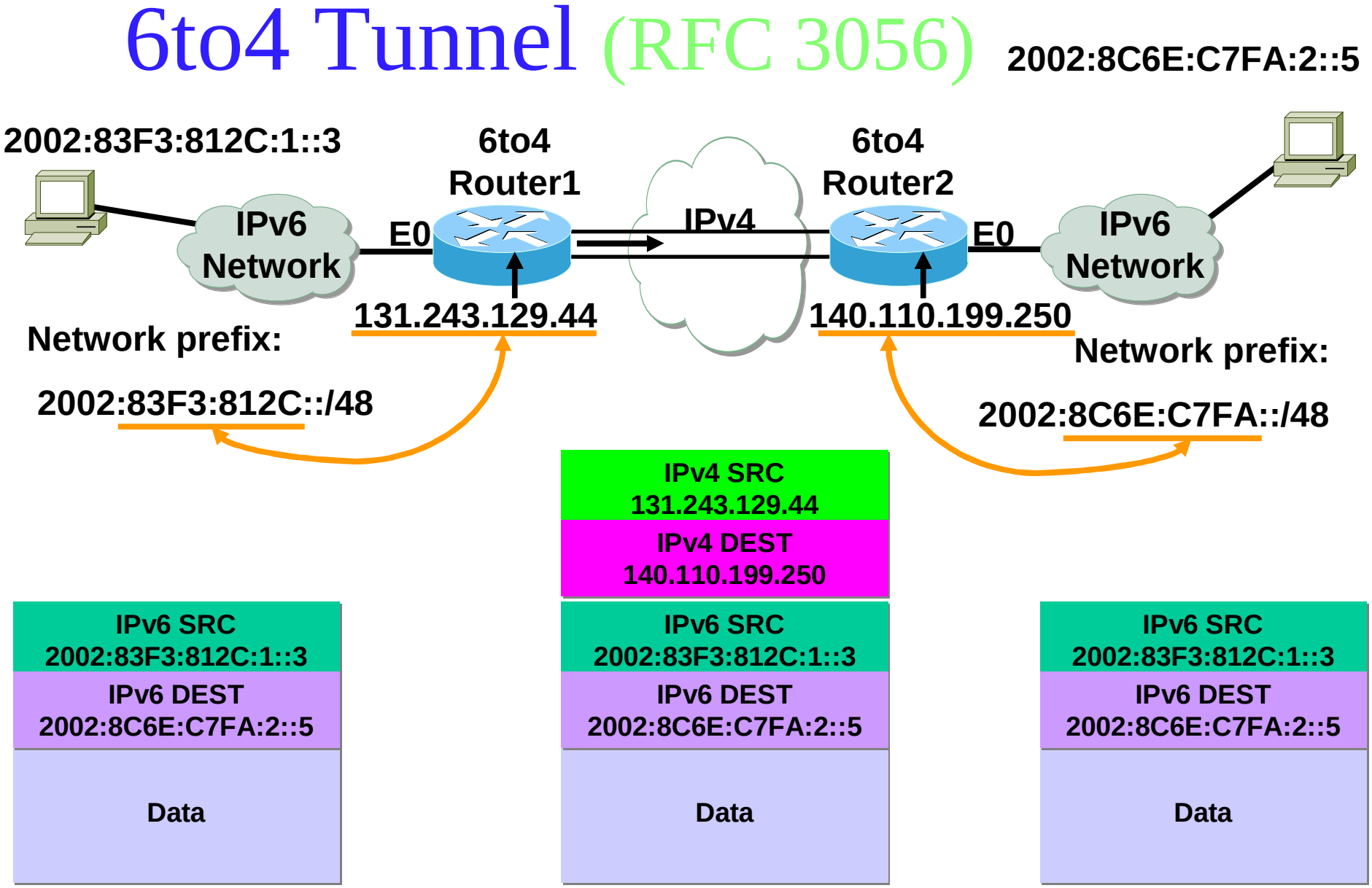


IPv4 Compatible Tunnel (RFC 2893)

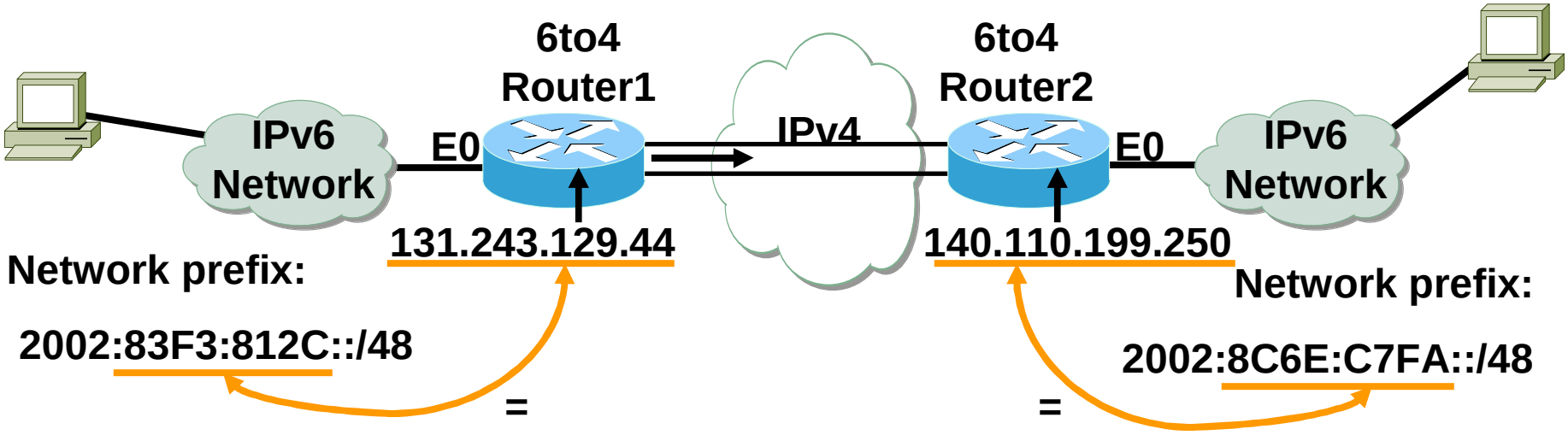


- IPv4-compatible addresses are easy way to autotunnel, but it:
  - May be deprecated soon
  - Consumes IPv4 addresses

6to4 Tunnel (RFC 3056)



6to4 Tunnel



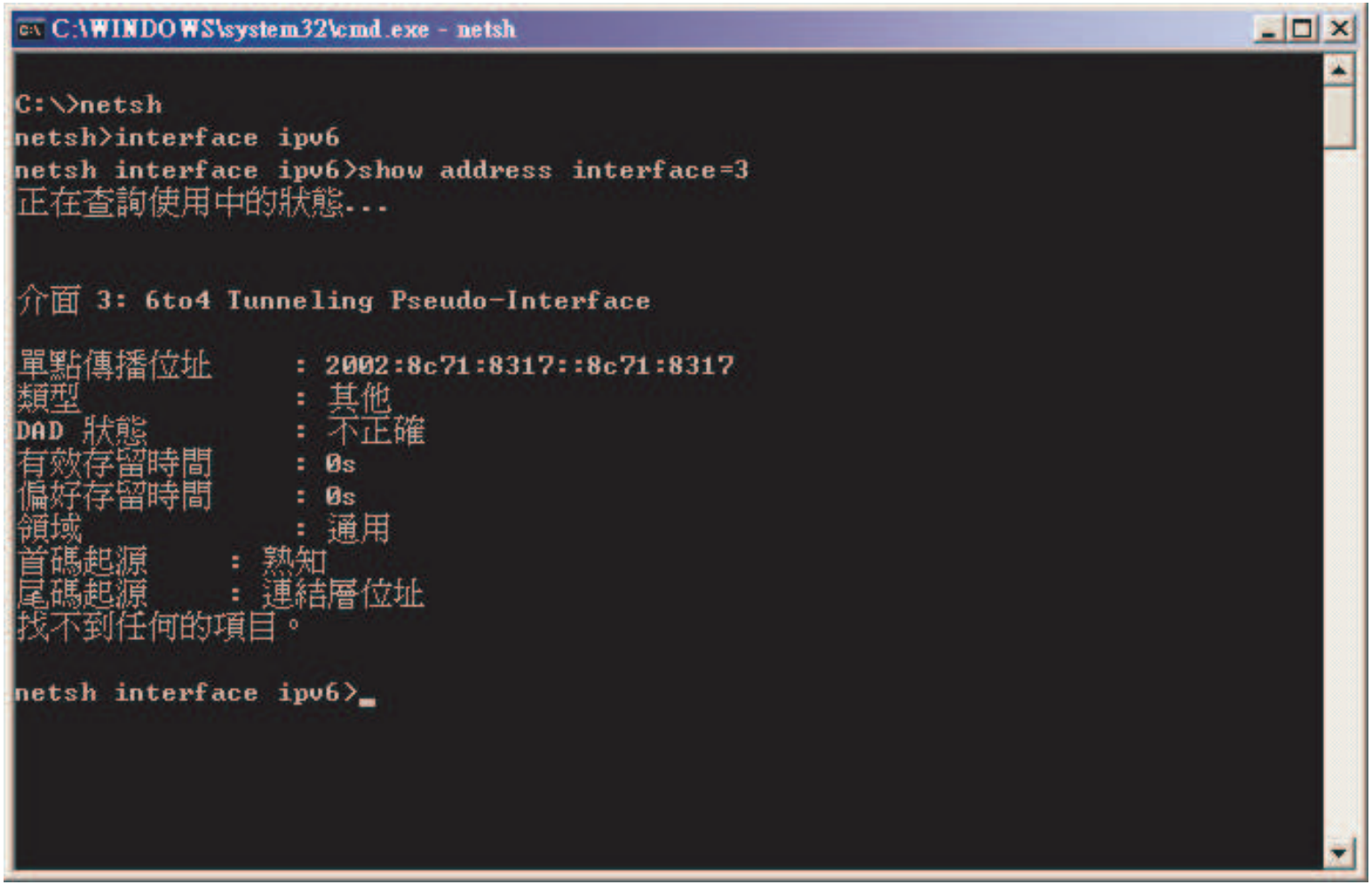
6to4 Tunnel:

- 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

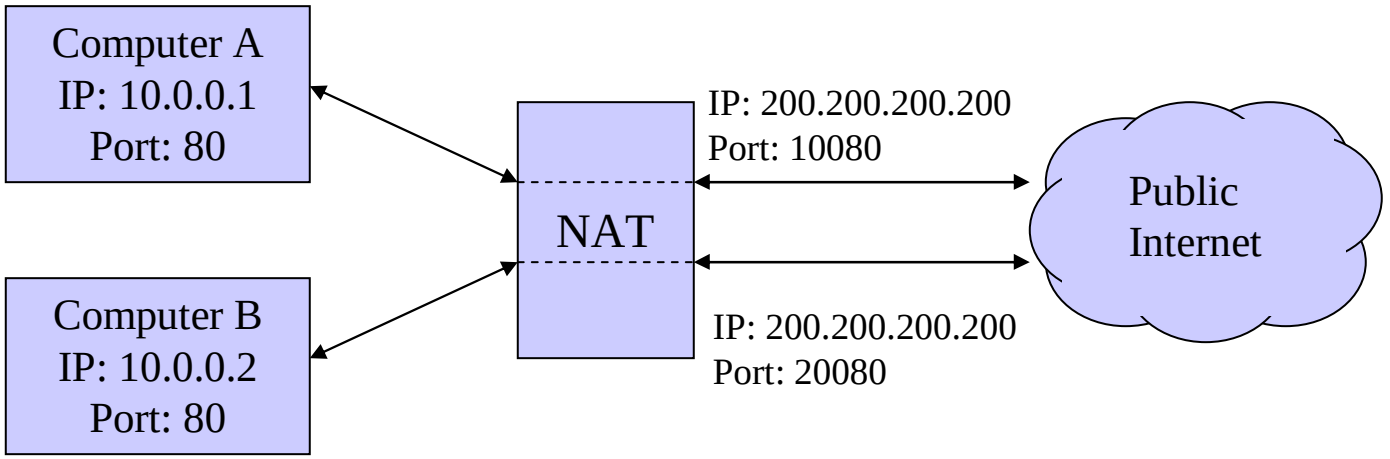
```
router2#
interface Ethernet0
 ip address 140.110.199.250 255.255.255.0
 ipv6 address 2002:8C6E:C7FA:1::/64 eui-64
interface Tunnel0
 no ip address
 ipv6 unnumbered Ethernet0
 tunnel source Ethernet0
 tunnel mode ipv6ip 6to4

ipv6 route 2002::/16 Tunnel0
```

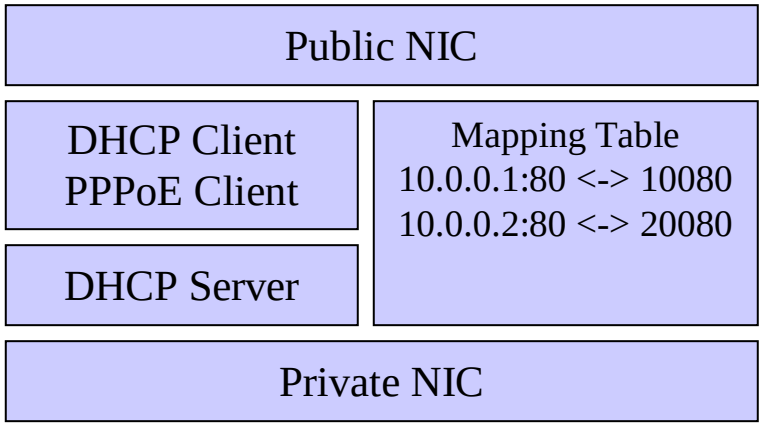
6to4 Tunnel in Windows XP

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



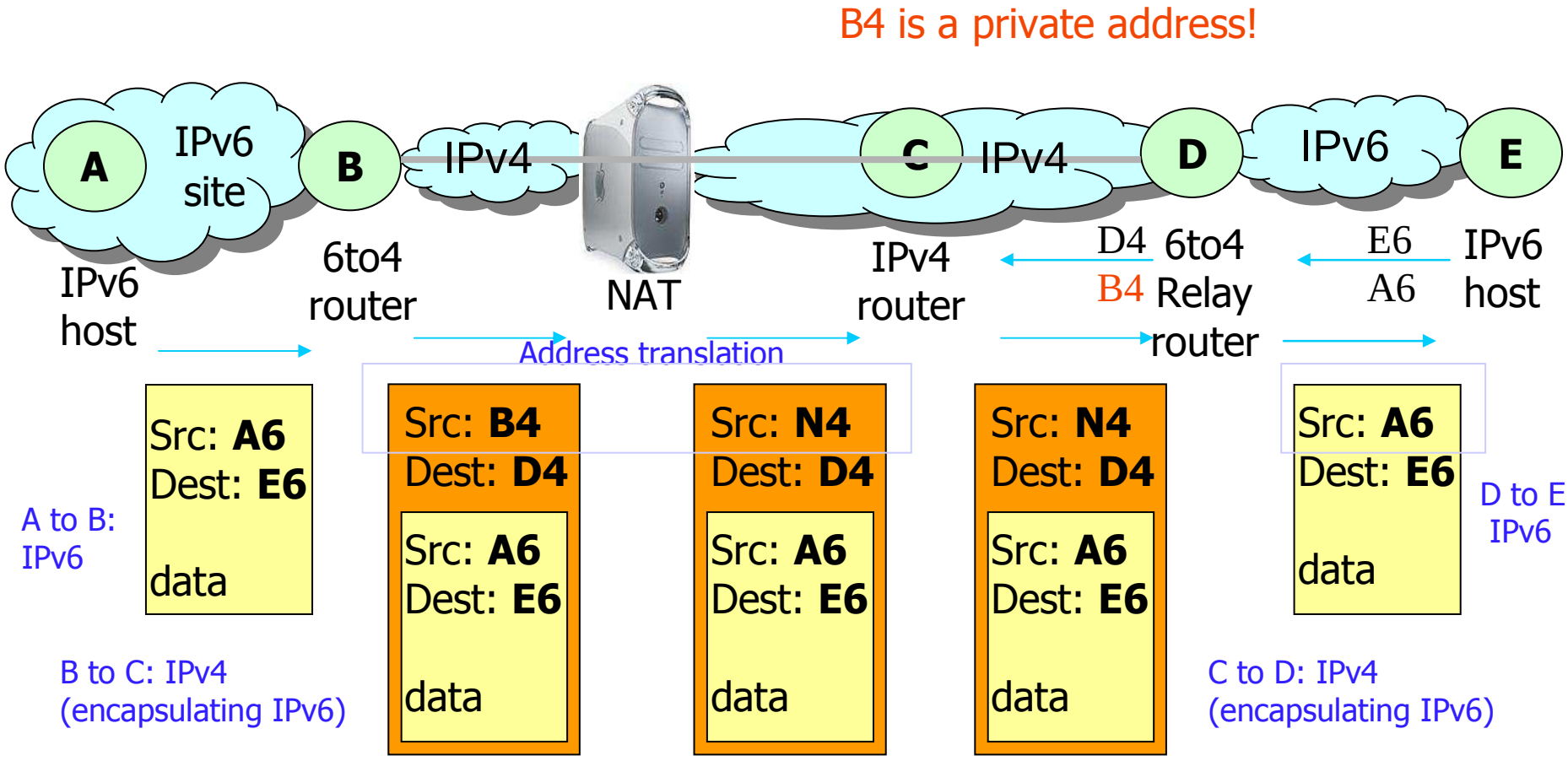
Network Address Translator





IPv6 tunneling problem

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

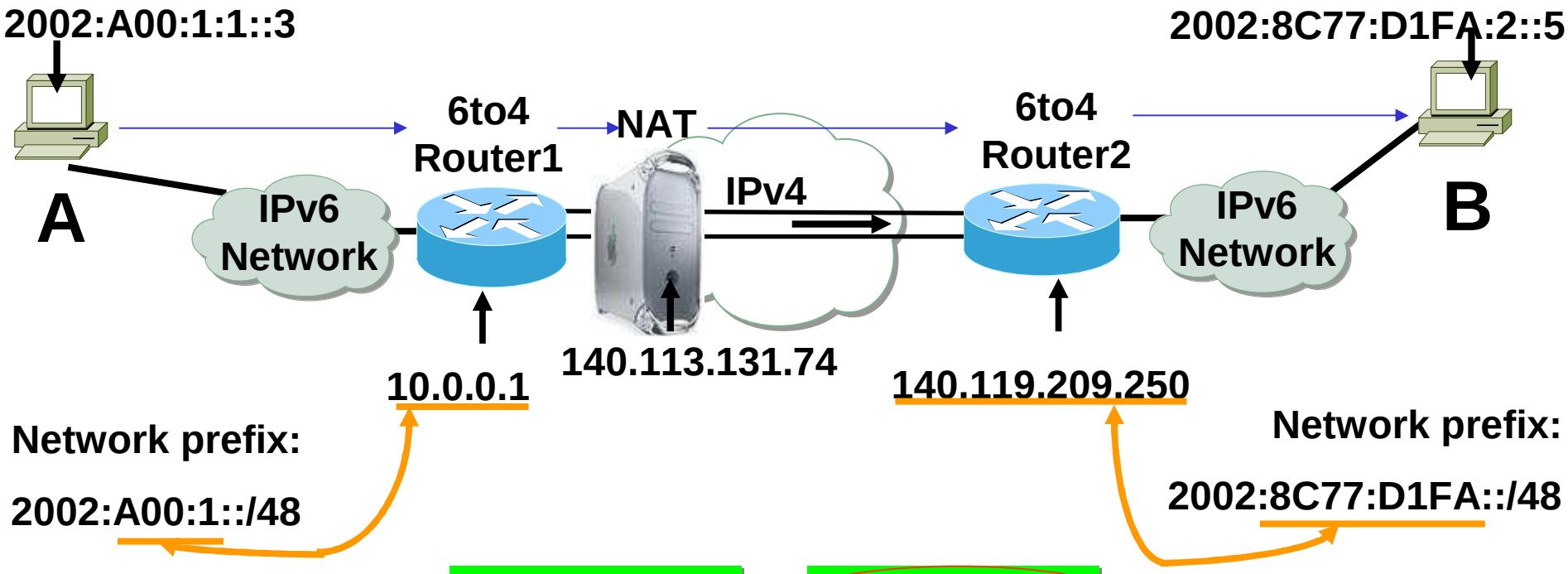


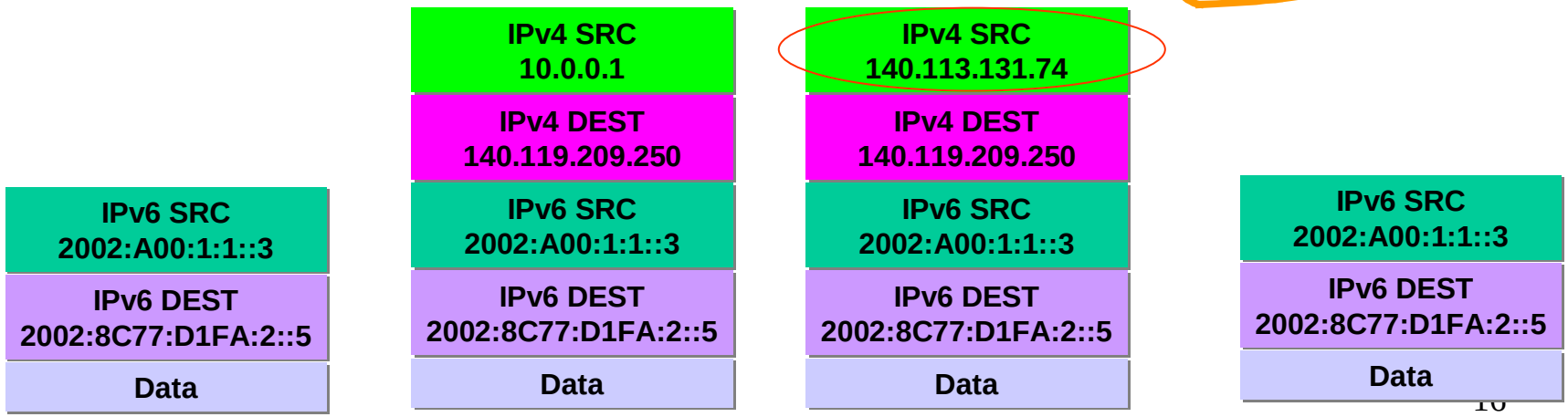
A v6 IP: 2002:A00:1:1::3/48 (A6)
B v6 IP: 2002:A00:1:1::1/48 (B6)
B v4 IP: 10.0.0.1 (B4)

E v6 IP: 2001:238:f88:4::2/64 (E6)
D v6 IP: 2001:238:f88:4::1/64 (D6)
D v4 IP: 140.114.1.254 (D4)

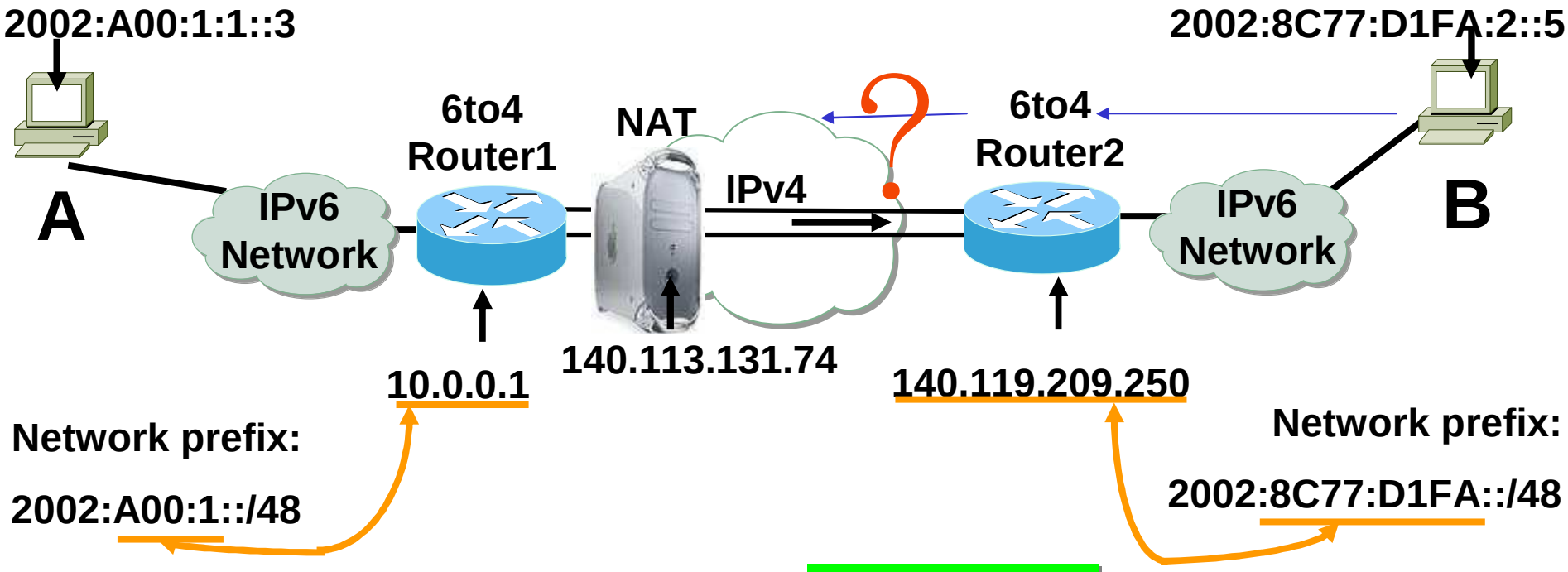
NAT address: 140.113.131.74 (N4)

IPv6 Tunneling Problem [1/2]

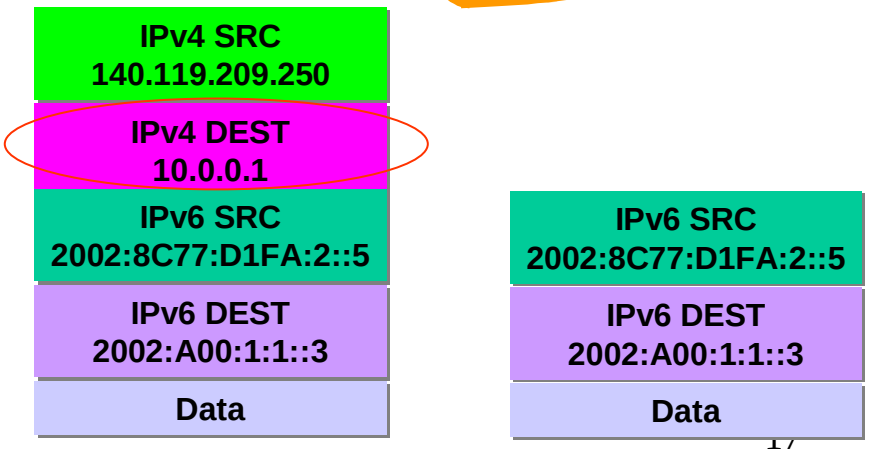




IPv6 Tunneling Problem [2/2]



Connection can't be established!



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.

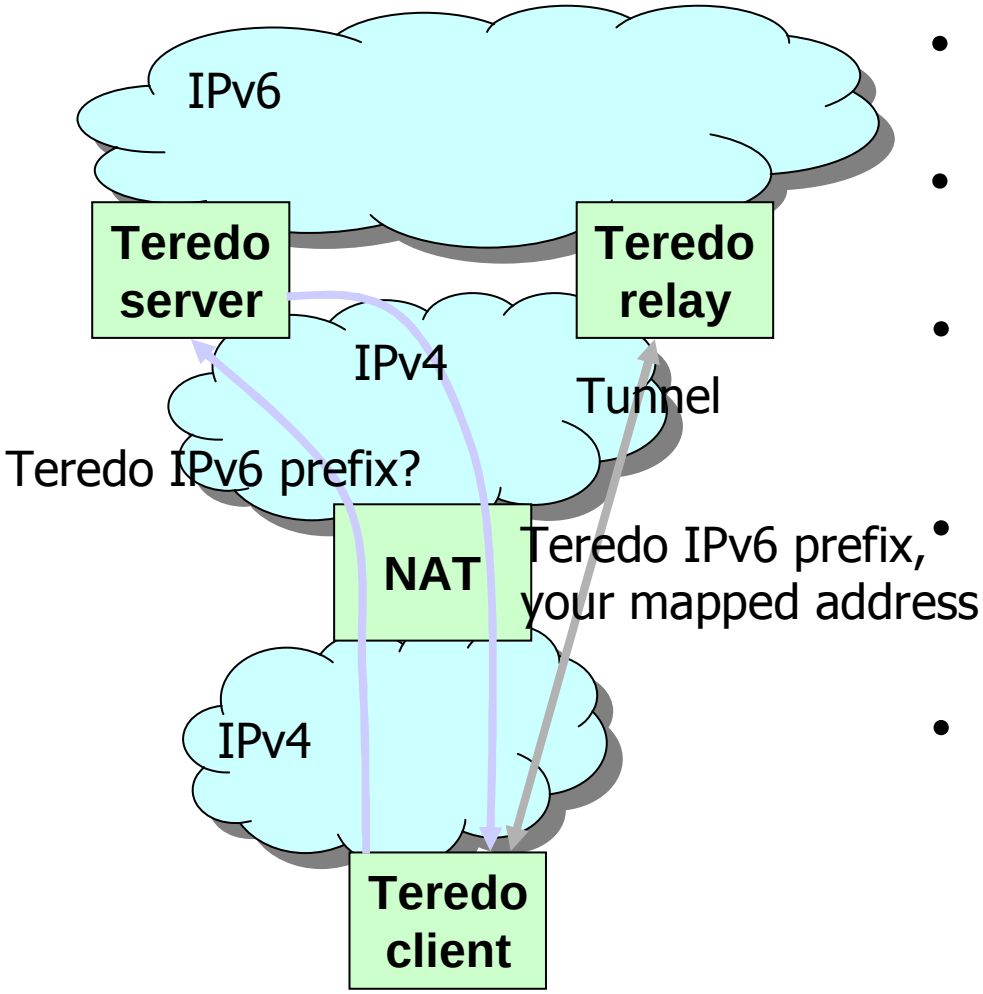
Teredo service

- 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.

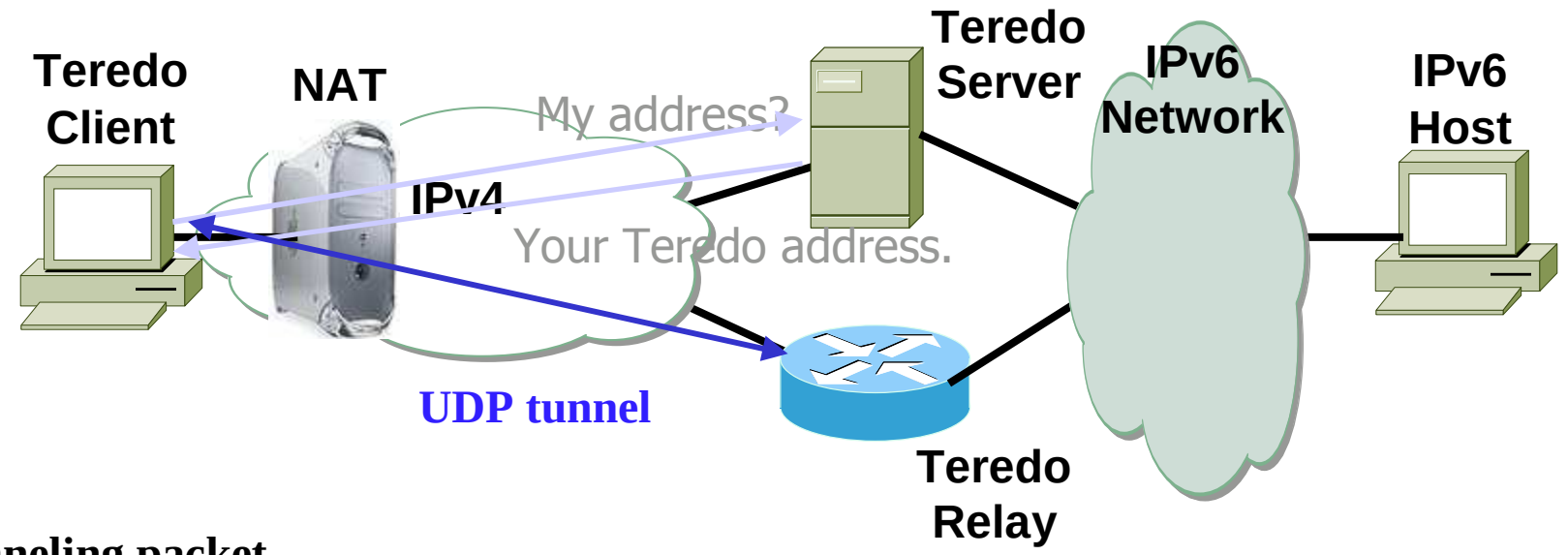
Operation model



- 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

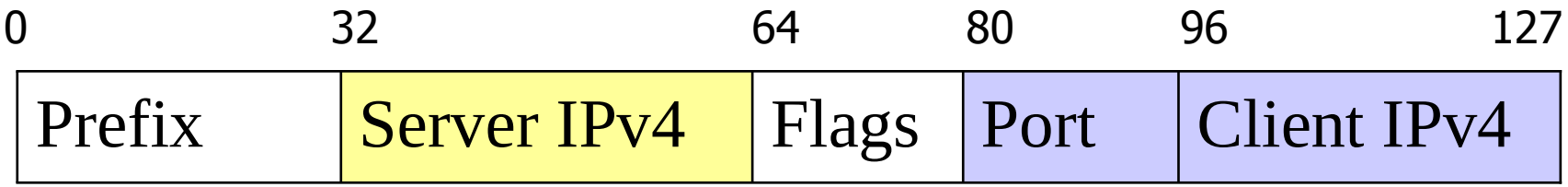
- Teredo Client gets its Teredo IPv6 address from Teredo Server.
- Use Teredo Relay as relay router.



Tunneling packet

IPv4 Header	UDP Header	Teredo Header	IPv6 packet

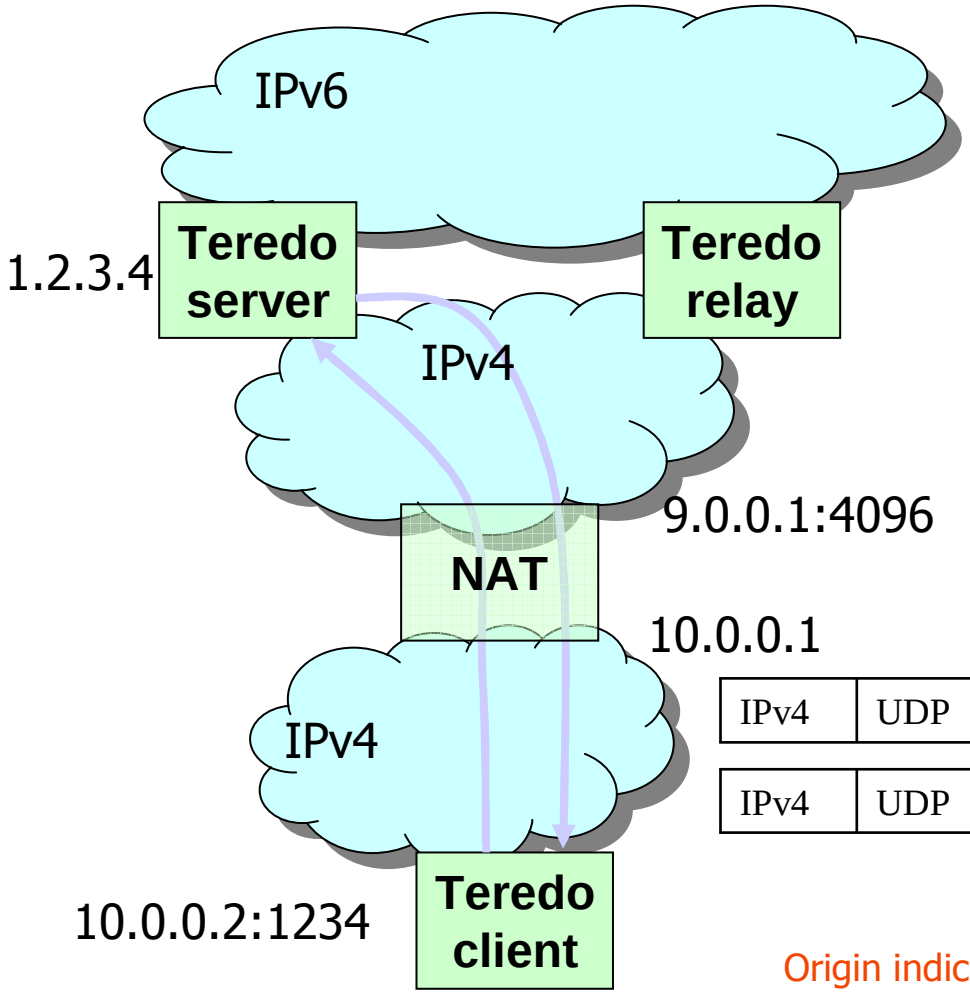
Teredo address encoding



- 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: “C00000UG00000000”
  - 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

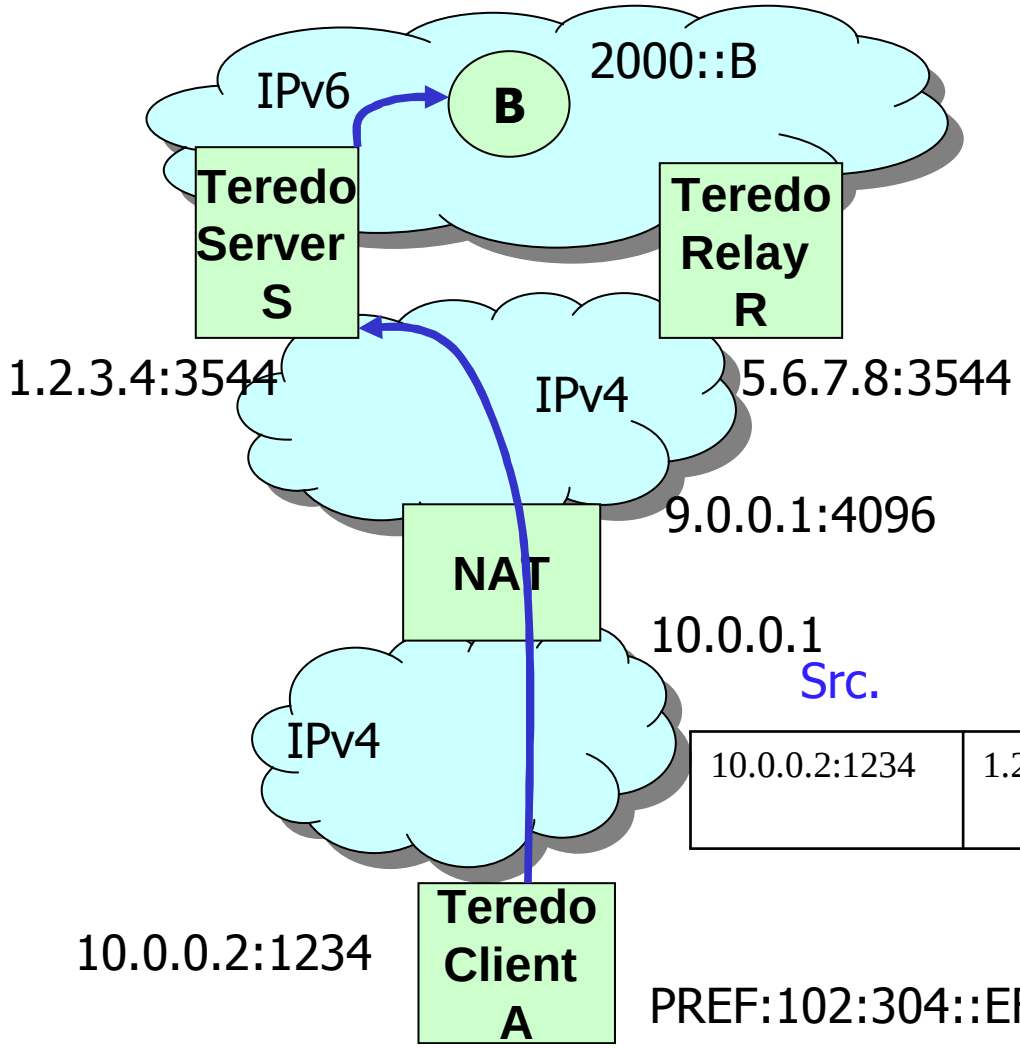
Origin indication format

0x00	0x00	mapped port #
mapped IPv4 address		

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⊕4096)
  - Obfuscated Address: 0xF6FF:FFFE (=0xFFFF:FFFF⊕ 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.

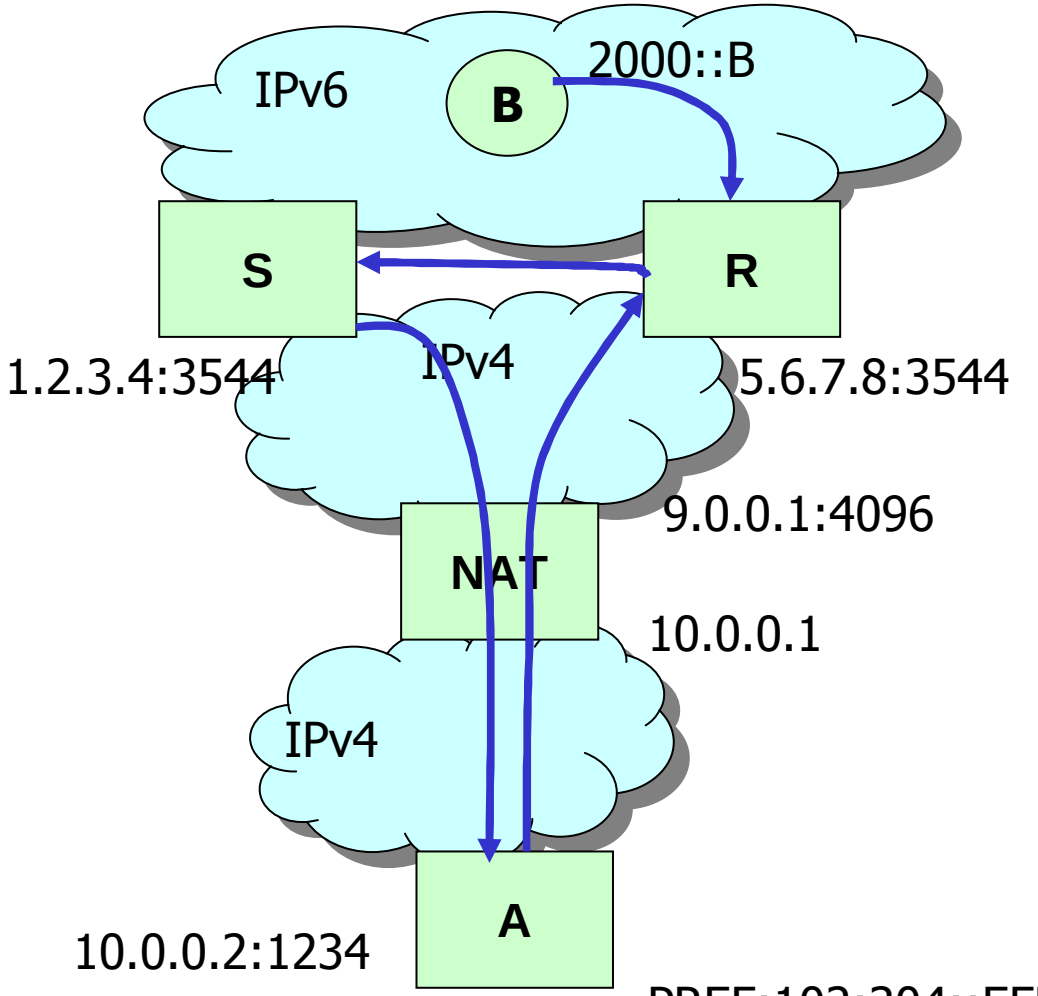
Packet from Teredo node to IPv6 node (1/3)



- 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.

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

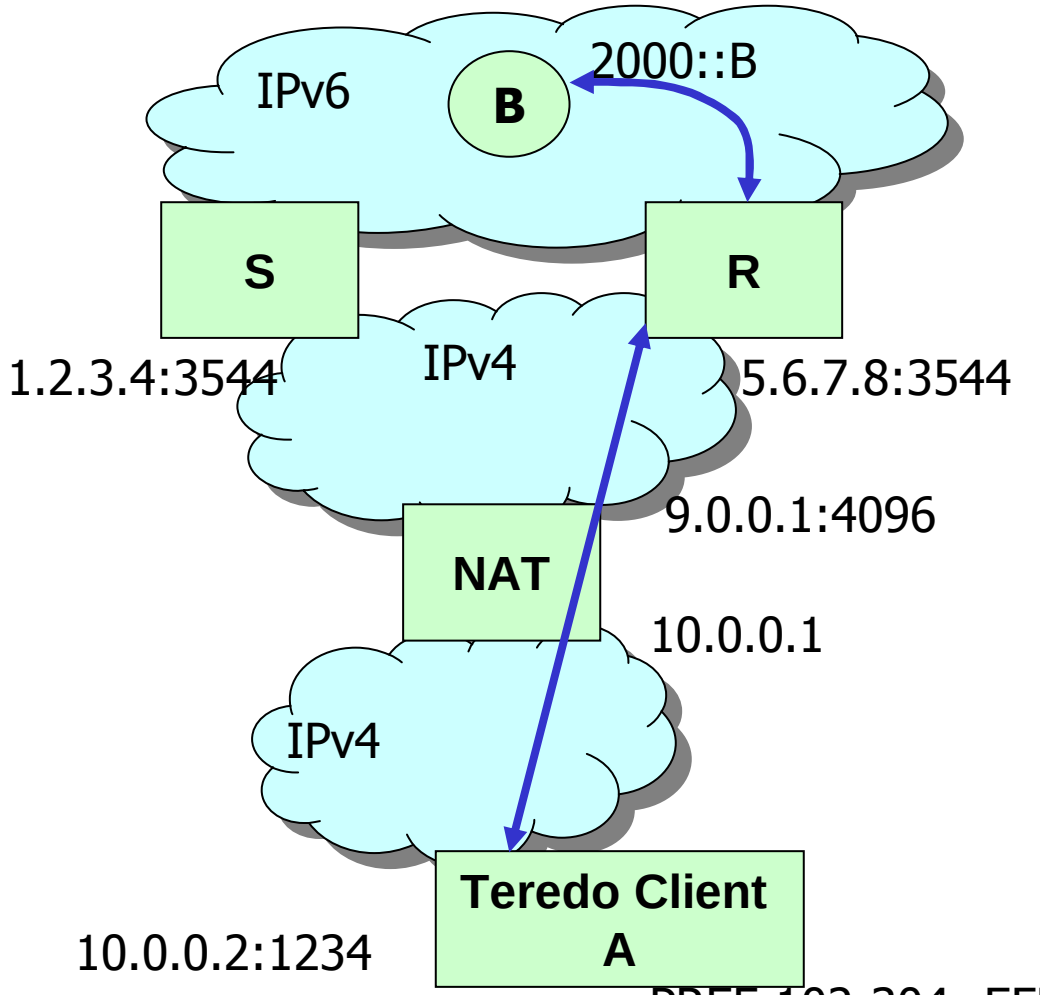
Packet from Teredo node to IPv6 node (2/3)



- 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.

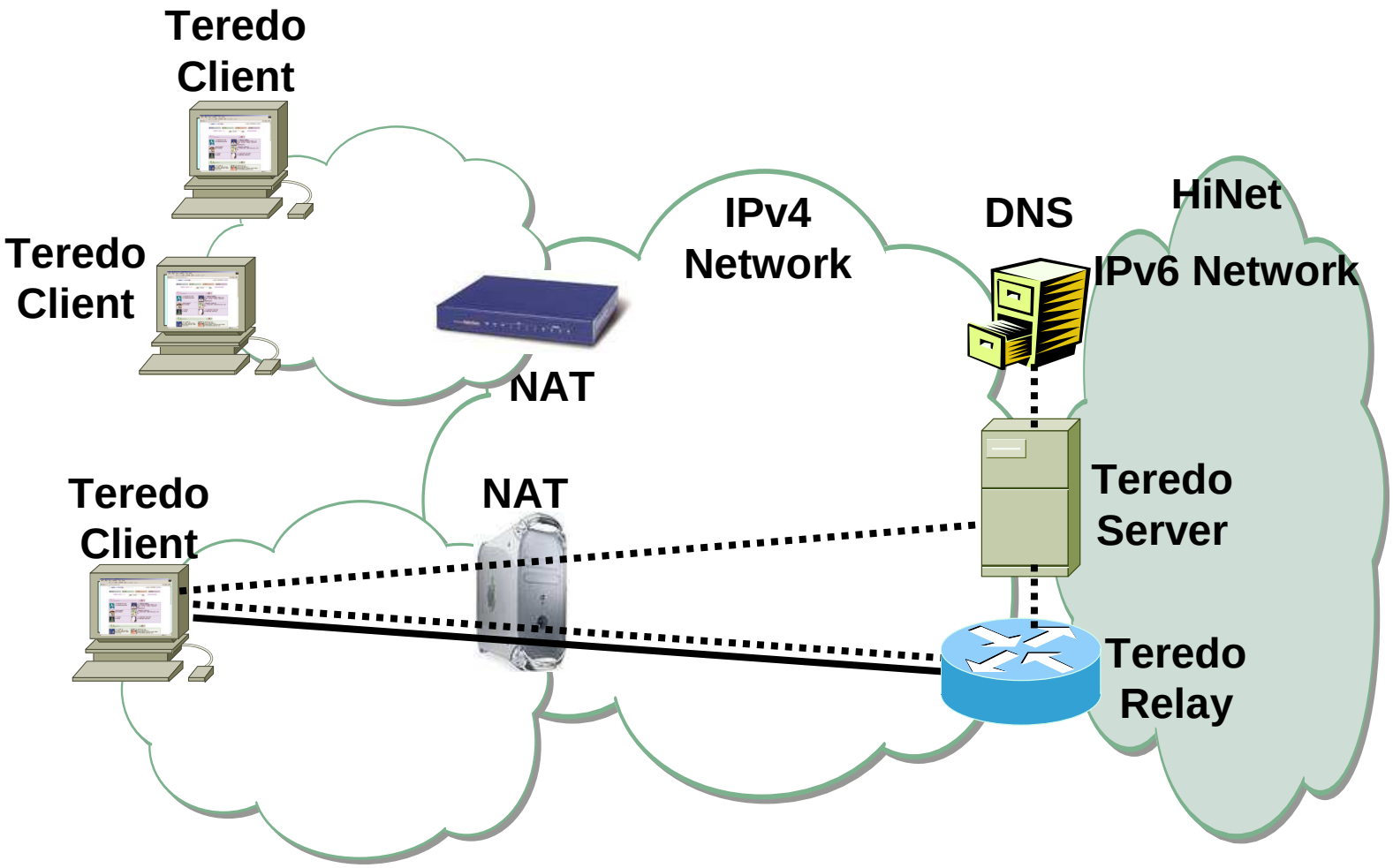
IPv6 Src.	IPv6 dest.
2000::B	PREF:102:304::EFFF:F6FF:FFFE

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

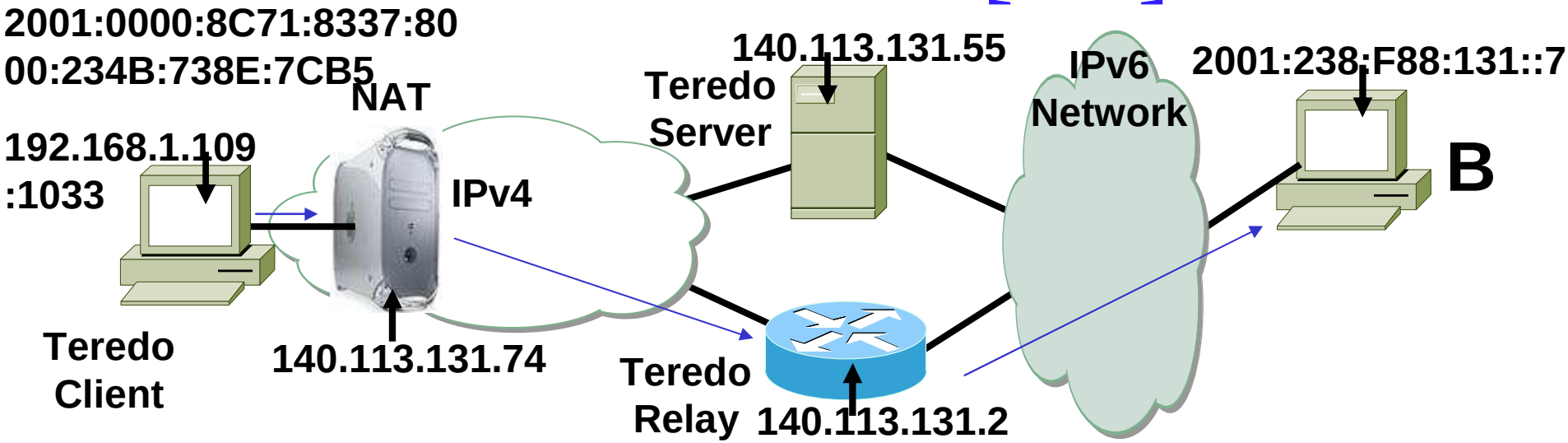


- R sends the queued “echo reply” to A.
- A knows B can be reached through address 5.6.7.8:3544.
- A will send all further packets directly through R.

Trial of Teredo in NCTU



Teredo Tunnel [1/2]

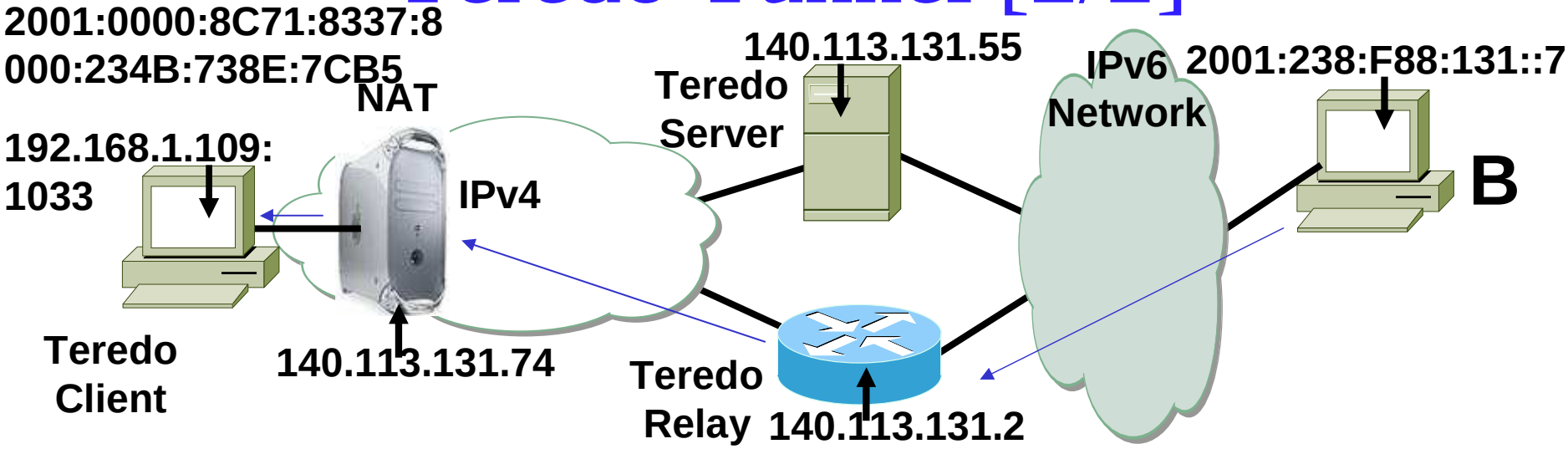


IPv4 SRC 10.0.0.2
IPv4 DEST 140.113.131.2
UDP SRC 1033
UDP DEST 3544
Teredo Header
IPv6 SRC 2001:0000:8C71:8337:80 00:234B:738E:7CB5
IPv6 DEST 2001:238:F88:131::7
Data

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]

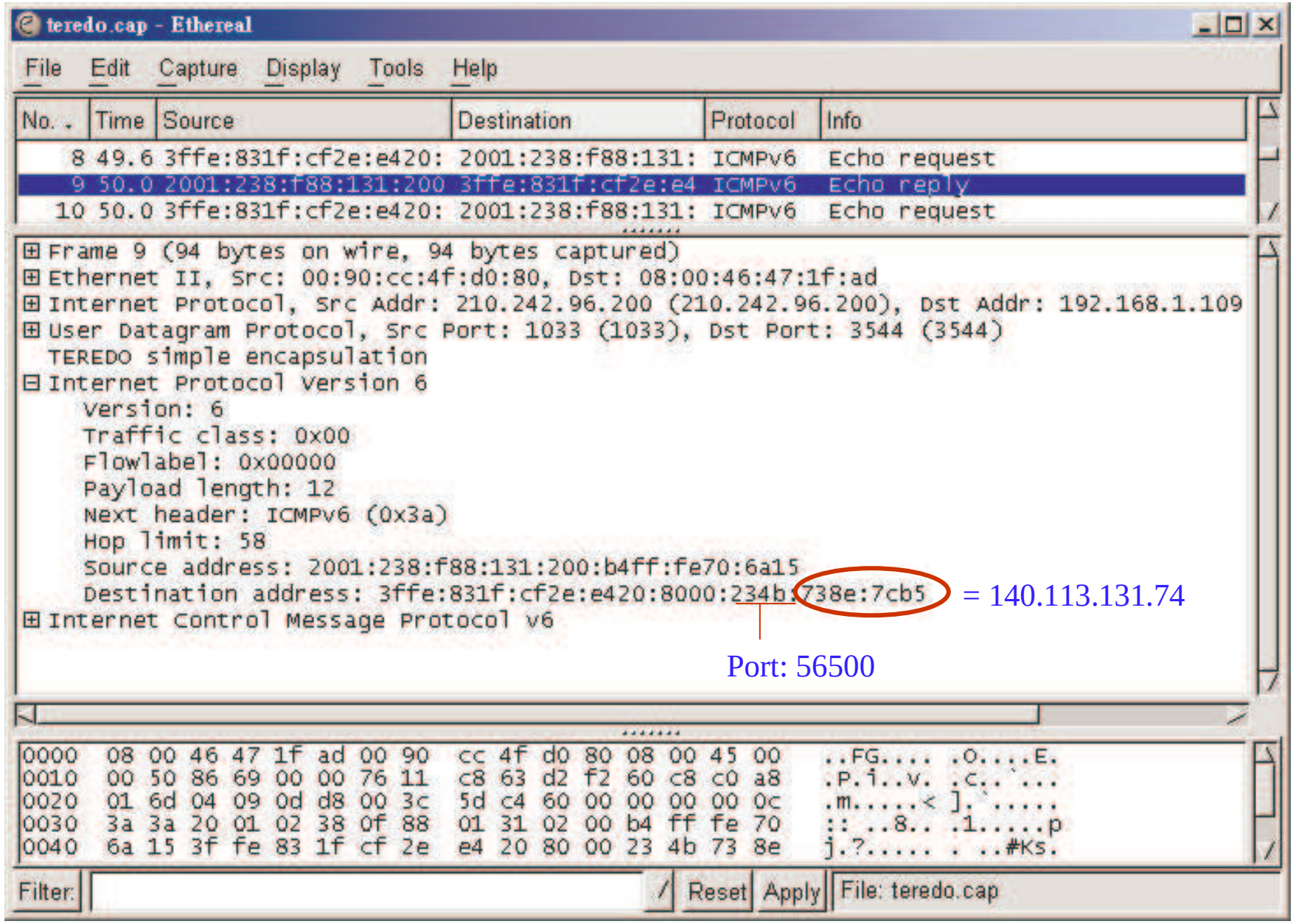


IPv4 SRC 140.113.131.2
IPv4 DEST 192.168.1.109
UDP SRC 3544
UDP DEST 1033
Teredo Header
IPv6 SRC 2001:238:F88:131::7
IPv6 DEST 2001:0000:8C71:8337:80 00:234E:738E:7CB5
Data

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



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.