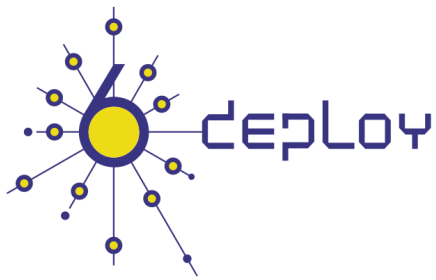
IPv6 Startup

APNIC Training
Christchurch, NZ
August, 2008

Jordi Palet (jordi.palet@consulintel.es)



Agenda

1. IPv6 setup in several Platforms (Windows XP/2003/Vista/W2K, Linux, BSD)
2. Basic Configuration, Stateless/Stateful Autoconfiguration, Privacy, Static Routes
3. Transition Mechanisms Configuration
4. Examples of Applications
5. Basic configuration in routers

Part 1

IPv6 Setup in several Platforms (Windows XP/2003/Vista/W2K, Linux, BSD)

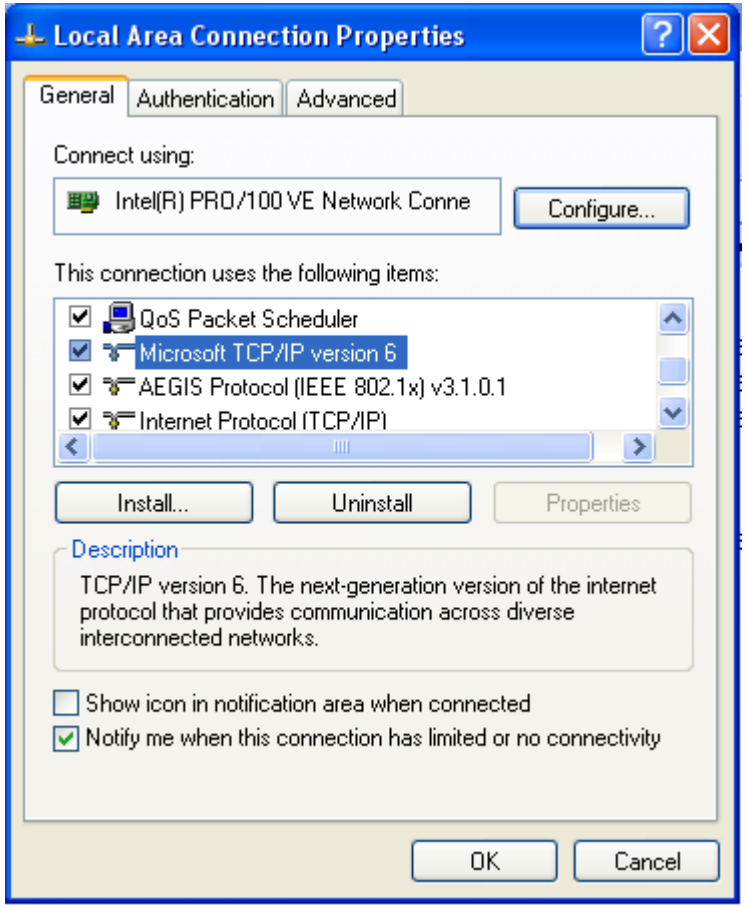
IPv6 Setup: XP/2003 (1)

- In a DOS Prompt:
  - **ipv6 install** to install IPv6 as Network Protocol

    - **ipconfig** or **ipv6 if** to check if IPv6 was installed

IPv6 Setup: XP/2003 (2)

- Another option to check if IPv6 was installed
  - Network Connections > Local Area Connection > Properties
- Also it is possible to install/uninstall IPv6 from here



IPv6 Setup: XP/2003 (3)

In a Command Prompt:

- **ipv6 uninstall** to delete IPv6 as Network Protocol

- **ipconfig** or **ipv6 if** to check if IPv6 was uninstalled

IPv6 Setup: Windows Vista (1)

- IPv6 is enabled by default
  - Includes GUI configuration
- New features:
  - Complete IPsec support
  - MLDv2
  - Link-Local Multicast Name Resolution (LLMNR)
    - No need for a DNS server. IPv6 nodes in a segment solicit the name to an IPv6 multicast address (similar to NetBIOS)
  - Literal IPv6 addresses in URLs
  - IPv6 over PPP
  - DHCPv6
  - Interface IDs are random by default (similar to privacy extensions)
  - Teredo with symmetric NAT support

IPv6 Setup: Windows Vista (2)

- Can't be uninstalled
- Can be disabled for a given interface
- Registry allows some tricks:
  - DWORD HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\tcpip6\Parameters\DisabledComponents
  - Each bit refers to a component. Cero means enable (default value)
    - Bit 0 ➔ all IPv6 tunnel-based interfaces, including ISATAP, 6to4, and Teredo
    - Bit 1 ➔ all 6to4 interfaces
    - Bit 2 ➔ all ISATAP interfaces
    - Bit 3 ➔ all Teredo interfaces
    - Bit 4 ➔ all IPv6 interfaces not-tunnel-based, including LAN and PPP
    - Bit 5 ➔ modify default policy table to prefer IPv4 instead of IPv6
  - Hex values for **DisabledComponents**

Disable all tunnel interfaces	0x1
Disable 6to4	0x2
Disable ISATAP	0x4
Disable Teredo	0x8
Disable Teredo and 6to4	0xA
Disable all LAN and PPP interfaces	0x10
Disable all LAN, PPP, and tunnel interfaces	0x11
Prefer IPv4 instead of IPv6	0x20
Disable IPv6 in all interfaces and prefer IPv4 instead of IPv6	0xFF

Address Selection Configuration

- In IPv6, each interface can have multiple addresses assigned to network and tunneling interfaces intended for different purposes. RFC3484 provides a standardized method to choose source and destination IPv6 addresses with which to attempt connections. It defines:
  - A destination address selection algorithm to sort the list of possible destination addresses in order of preference
  - A source address selection algorithm to choose the best source address to use with a destination address
- This is implemented by the Operating System so applications do not need to include their own address selection algorithms, reducing the development burden on IPv6-capable applications. However, the algorithm can be overridden by applications if either the source or destination address is used rather its full qualified domain name (FQDN)

Address Selection in Windows

- netsh interface ipv6 show prefixpolicy --> show the current local prefix policy table
- netsh interface ipv6 add prefixpolicy --> add new entries in the local prefix policy table
- netsh interface ipv6 set prefixpolicy --> set entries in the local prefix policy table
- netsh interface ipv6 delete prefixpolicy --> delete entries in the local prefix policy table
- Example:

```
C:\>netsh interface ipv6 show prefixpolicy
Precedence          Label          Prefix
  5         5       2001::/32
 10         4       ::ffff:0:0/96
 20         3       ::/96
 30         2       2002::/16
 40         1       ::/0
 50         0       ::1/128
```

- To change the precedence of one prefix, for example Teredo prefix over IPv4 addresses:

```
C:\>netsh interface ipv6 set prefixpolicy prefix=2001::/32 precedence=15 label=5
```

IPv6 Setup: W2K (1)

- Non-production stack available (originally developed by Microsoft Research)
- Download the “Microsoft IPv6 Technology Preview for Windows 2000”:
  - `Available at http://www.ipv6tf.org/using/connectivity/guides.php ?cid=1`
  - Note that Windows 2000 IPv6 isn't supported anymore by Microsoft
- Install Procedure:
  - Log on to the Windows 2000 with local administrator privileges
  - Extract IPv6 Technology Preview files, for example in C:\IPv6Kit
  - Follow the procedure in SPn & IE6 fixed.txt in order to change /setup/hotfix.ini file
  - Run the Setup.exe or hotfix.exe
  - From the Windows 2000 desktop, click Start, point to Settings, and then click Network and Dial-up Connections. As an alternative, you can right-click My Network Places, and then click Properties
  - Right-click the Ethernet-based connection to which you want to add the IPv6 protocol, and then click Properties (typically, this connection is named Local Area Connection
  - Click Install)
  - In the Select Network Component Type dialog box, click Protocol, and then click Add
  - In the Select Network Protocol dialog box, click Microsoft IPv6 Protocol and then click OK
  - Click Close to close the Local Area Connection Properties dialog box
- In a DOS Prompt:
  - **ipv6 if** to check if IPv6 has been installed

IPv6 Setup: W2K (2)

- Uninstall Procedure:
  - Log on to the Windows 2000 with local administrator privileges
  - From the Windows 2000 desktop, click Start, point to Settings, and then click Network and Dial-up Connections. As an alternative, you can right-click My Network Places, and then click Properties
  - Right-click the connection to which you want to remove the Microsoft Research IPv6 protocol, and then click Properties (typically, this connection is named Local Area Connection)
  - Click MSR IPv6 Protocol and then click Uninstall
  - In the Uninstall MSR IPv6 Protocol dialog box, click Yes
  - In the Local Network dialog box, click Yes to restart your computer
- In a DOS Prompt:
  - **ipv6 if** to check if IPv6 was uninstalled

IPv6 Setup: Linux (1)

- To check if IPv6 is installed:

  #test -f /proc/net/if_inet6 && echo “Current Kernel supports IPv6“

- Module Installation:

  #modprobe ipv6

- Module check:

  #lsmod |grep -w 'ipv6' && echo “IPv6 module loaded"

- Automatic Load/Unload of Module (/etc /modules.conf o /etc/conf.modules):

  alias net-pf-10 ipv6 #enables load on demand

  alias net-pf-10 off #disables load on demand

IPv6 Setup: Linux (2)

 # ifconfig to check

```
eth0  Link encap:Ethernet  HWaddr 00:E0:81:05:46:57
      inet addr:10.0.0.3  Bcast:10.0.0.255  Mask:255.255.255.0
      inet6 addr: fe80::2e0:81ff:fe05:4657/64 Scope:Link
      inet6 addr: 2001:800:40:2a05::3/64 Scope:Global
      UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
      RX packets:2010563 errors:0 dropped:0 overruns:0 frame:0
      TX packets:1700527 errors:0 dropped:0 overruns:2 carrier:0
      collisions:0 txqueuelen:100
      RX bytes:205094215 (195.5 Mb)  TX bytes:247063610 (235.6Mb)
      Interrupt:11 Base address:0xe000 Memory:f8201000-f8201038
lo    Link encap:Local Loopback
      inet addr:127.0.0.1  Mask:255.0.0.0
      inet6 addr: ::1/128 Scope:Host
      UP LOOPBACK RUNNING  MTU:16436  Metric:1
      RX packets:1675838 errors:0 dropped:0 overruns:0 frame:0
      TX packets:1675838 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:0
      RX bytes:659846244 (629.2 Mb)  TX bytes:659846244 (629.2 Mb)
```

IPv6 Setup: Linux (3)

Persistent Configuration

- Red Hat (from 7.1) and similar “distros”:

  Add in /etc/sysconfig/network:

  NETWORKING_IPV6=yes

  Network Restart:

  # service network restart

  Or

  #/etc/init.d/network restart

- SUSE:

  Add in /etc/sysconfig/network/ifcfg-<Interface-Name>:

  SUSE 8.0: IP6ADDR="<ipv6-address>/<prefix>"

  SUSE 8.1: IPADDR="<ipv6-address>/<prefix>"

IPv6 Setup: Linux (4)

Persistent Configuration

- Debian:

  Once the IPv6 module is loaded, then edit /etc/network/interfaces, for example:

```
iface eth0 inet6 static
        pre-up modprobe ipv6
        address 3ffe:ffff:1234:5::1:1
        # unable autoconfiguration:
        # up echo 0 > /proc/sys/net/ipv6/conf/all/autoconf
        netmask 64
        # router is autoconfigured and doesn't have static address
        # it finds it because of
        # (/proc/sys/net/ipv6/conf/all/accept_ra).
        # if not, gateway must be configured:
        # gateway 3ffe:ffff:1234:5::1
```

  – Reboot or:

```
# ifup --force eth0
```

IPv6 Setup: Linux (5)

- Tools:

1. net-tools package

```
# /sbin/ifconfig -? 2>& 1|grep -qw 'inet6' && echo "'ifconfig supports IPv6“
# /sbin/route -? 2>& 1|grep -qw 'inet6' && echo "'route supports IPv6"
```

2. iproute package

```
# /sbin/ip 2>&1 |grep -qw 'inet6' && echo "'ip supports IPv6“
```

3. iputils package contains ping6, traceroute6 and tracepath6

IPv6 Setup: BSD (1)

- To install the Stack (Versions 4.5+)
- Good IPv6 support

Part 2

Basic Configuration Stateless/Stateful Autoconfiguration, Privacy, Static Routes

Basic Configuration: XP/2003 (1)

- Basic Commands in XP/2003
- Useful to obtain information about the status and to configure interfaces, addresses, caches, routes, and so on
- Two groups of commands:
  - **ipv6.exe** (covers up to Windows XP SP2)
    - Some changes are not persistent (values lost with each reboot). It is possible to execute a configuration in a script in each boot.
  - **netsh interface ipv6** (starting on Windows XP SP2 and Server 2003)
    - Option **store=active|persistent** to save changes
- Equivalences at: http://www.microsoft.com /windowsserver2003/technologies/ipv6 /ipv62netshtable.mspx

Basic Configuration: XP/2003 (2)

- **"ipv6" Commands**
  - ipv6 [-p] [-v] if [ifindex]
  - ipv6 [-p] ifcr v6v4 v4src v4dst [nd] [pmld]
  - ipv6 [-p] ifcr 6over4 v4src
  - ipv6 [-p] ifc ifindex [forwards] [-forwards] [advertises] [-advertises] [mtu #bytes] [site site-identifier] [preference P]
  - ipv6 rlu ifindex v4dst
  - ipv6 [-p] ifd ifindex
  - ipv6 [-p] adu ifindex/address [life validlifetime[/preflifetime]] [anycast] [unicast]
  - ipv6 nc [ifindex [address]]
  - ipv6 ncf [ifindex [address]]
  - ipv6 rc [ifindex address]
  - ipv6 rcf [ifindex [address]]
  - ipv6 bc
  - ipv6 [-p] [-v] rt
  - ipv6 [-p] rtu prefix ifindex[/address] [life valid[/pref]] [preference P] [publish] [age] [spl SitePrefixLength]
  - ipv6 spt
  - ipv6 spu prefix ifindex [life L]
  - ipv6 [-p] gp
  - ipv6 [-p] gpu [parameter value] ... (try -?)
  - ipv6 renew [ifindex]
  - ipv6 [-p] ppt
  - ipv6 [-p] ppu prefix precedence P srclabel SL [dstlabel DL]
  - ipv6 [-p] ppd prefix
  - ipv6 [-p] reset
  - ipv6 install
  - ipv6 uninstall

Basic Configuration: XP/2003 (3)

- **“netsh interface ipv6” Commands**
  - 6to4 - Changes to the ‘netsh interface ipv6 6to4’ context
  - ? - Displays a list of commands
  - add - Adds a configuration entry to a table
  - delete - Deletes a configuration entry from a table
  - dump - Displays a configuration script
  - help - Displays a list of commands
  - install - Installs IPv6
  - isatap - Changes to the ‘netsh interface ipv6 isatap’ context
  - renew - Restarts IPv6 interfaces
  - reset - Resets IPv6 configuration state
  - set - Sets configuration information
  - show - Displays information
  - uninstall - Uninstalls IPv6

Basic Configuration: XP/2003 (4)

- **"netsh interface ipv6 add" Commands**
  - add 6over4tunnel - Creates a 6over4 interface.
  - add address - Adds an IPv6 address on an interface.
  - add dns - Adds a static DNS server address.
  - add prefixpolicy - Adds a prefix policy entry.
  - add route - Adds an IPv6 route over an interface.
  - add v6v4tunnel - Creates an IPv6-in-IPv4 point-to-point tunnel.
- **"netsh interface ipv6 set" Commands**
  - set address - Modifies IPv6 address information.
  - set global - Modifies global configuration general parameters.
  - set interface - Modifies interface configuration parameters.
  - set mobility - Modifies mobility configuration parameters.
  - set prefixpolicy - Modifies prefix policy information.
  - set privacy - Modifies privacy configuration parameters.
  - set route - Modifies route parameters.
  - set state - Sets the state of deprecated functionality.
  - set teredo - Sets Teredo state.
- **"netsh interface ipv6 show" Commands**
  - show address - Shows IPv6 addresses.
  - show bindingcacheentries - Shows binding cache entries.
  - show destinationcache - Shows destination cache entries.
  - show dns - Displays the DNS server addresses.
  - show global - Shows global configuration parameters.
  - show interface - Shows interface parameters.
  - show joins - Shows IPv6 multicast addresses.
  - show mobility - Shows mobility configuration parameters.
  - show neighbors - Shows neighbor cache entries.
  - show prefixpolicy - Shows prefix policy entries.
  - show privacy - Shows privacy configuration parameters.
  - show routes - Shows route table entries.
  - show siteprefixes - Shows site prefix table entries.
  - show state - Shows the state of deprecated functionality.
  - show teredo - Shows Teredo service state.

Basic Configuration: XP/2003 (5)

- Interface Information
- **ipconfig [/all]**
- **ipv6 [-v] if [IfIndex]**
- Example: ipv6 if 5

```
Interface 5: Ethernet: Local Area Connection
  Guid {F5149413-6E54-4FDA-87BD-24067735E363}
  uses Neighbor Discovery
  uses Router Discovery
  link-layer address: 00-01-4a-18-26-c7
    preferred global 2001:db8::2, life infinite (manual)
    preferred global 2001:db8::4, life infinite (manual)
    preferred global 2001:db8::fde7:a76f:62d5:3bb9, life 6d21h3m20s/21h33s (temporary)
    preferred global 2001:db8::201:4aff:fe18:26c7, life 29d23h51m39s/6d23h51m39s (public)
    preferred link-local fe80::201:4aff:fe18:26c7, life infinite
    multicast interface-local ff01::1, 1 refs, not reportable
    multicast link-local ff02::1, 1 refs, not reportable
    multicast link-local ff02::1:ff18:26c7, 2 refs, last reporter
    multicast link-local ff02::1:ffd5:3bb9, 1 refs, last reporter
    multicast link-local ff02::1:ff00:4, 1 refs, last reporter
    multicast link-local ff02::1:ff00:2, 1 refs, last reporter
  link MTU 1500 (true link MTU 1500)
  current hop limit 64
  reachable time 29000ms (base 30000ms)
  retransmission interval 1000ms
  DAD transmits 1
  default site prefix length 48
```

Basic Configuration: XP/2003 (6)

- Ping in XP/2003
- **ping6 [-t] [-a] [-n count] [-l size] [-w timeout] [-s srcaddr] [-r] dest**
  - t Ping the specified host until interrupted
  - a Resolve addresses to hostnames
  - n count Number of echo requests to send
  - l size Send buffer size
  - w timeout Timeout in milliseconds to wait for each reply
  - s srcaddr Source address to use
  - r Use routing header to test reverse route also
- **ping** command default to IPv6 if available

Basic Configuration: XP/2003 (7)

- **Examples of Ping in XP/2003**
- **ping6 www.ipv6tf.org**

Pinging www.ipv6tf.org [2001:800:40:2a03::3]
from 2001:800:40:2a05:9c4d:b1cd:98d5:5a32 with 32 bytes of data:
Reply from 2001:800:40:2a03::3: bytes=32 time<1ms
Reply from 2001:800:40:2a03::3: bytes=32 time<1ms
Reply from 2001:800:40:2a03::3: bytes=32 time<1ms
Reply from 2001:800:40:2a03::3: bytes=32 time<1ms
Ping statistics for 2001:800:40:2a03::3:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 0ms, Average = 0ms

Basic Configuration: XP/2003 (8)

- **Examples of Ping in XP/2003**
- **ping ::1**

  Pinging ::1 from ::1 with 32 bytes of data:
  Reply from ::1: bytes=32 time<1ms
  Reply from ::1: bytes=32 time<1ms
  Reply from ::1: bytes=32 time<1ms
  Reply from ::1: bytes=32 time<1ms
  Ping statistics for ::1:
  Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
  Approximate round trip times in milli-seconds:
  Minimum = 0ms, Maximum = 0ms, Average = 0ms
- **ping6 fe80::201:4aff:fe18:26c7 (own link-local)**

  Pinging fe80::201:4aff:fe18:26c7 from fe80::201:4aff:fe18:26c7%5 with 32 bytes of data:
  Reply from fe80::201:4aff:fe18:26c7%5: bytes=32 time<1ms
  Reply from fe80::201:4aff:fe18:26c7%5: bytes=32 time<1ms
  Reply from fe80::201:4aff:fe18:26c7%5: bytes=32 time<1ms
  Reply from fe80::201:4aff:fe18:26c7%5: bytes=32 time<1ms
  Ping statistics for fe80::201:4aff:fe18:26c7:
  Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
  Approximate round trip times in milli-seconds:
  Minimum = 0ms, Maximum = 0ms, Average = 0ms

Basic Configuration: XP/2003 (9)

- Which are my neighbors?
  - **netsh interface ipv6 show neighbors**

```
...
Interface 5: Local Area Connection
Internet Address                          Physical Address   Type
----------------------------------------  -----------------  -----------
fe80::201:4aff:fe18:26c7                  00-01-4a-18-26-c7  Permanent
fe80::200:87ff:fe28:a0e0                  00-00-87-28-a0-e0  Stale (router)
2001:db8::201:4aff:fe18:26c7              00-01-4a-18-26-c7  Permanent
2001:db8::fde7:a76f:62d5:3bb9             00-01-4a-18-26-c7  Permanent
2001:db8::2a03::3                         00-e0-81-05-46-57  Stale
2001:db8::1                               00-00-87-28-a0-e0  Stale
2001:db8::2                               00-01-4a-18-26-c7  Permanent
2001:db8::4                               00-01-4a-18-26-c7  Permanent
```

- The reference to specific interface is done with “%”
  - **%5 is about interface 5**

Basic Configuration: XP/2003 (10)

- **Examples of Ping in XP/2003**
- **ping fe80::200:87ff:fe28:a0e0%5 (link-local neighbor in interface 5)**

Pinging fe80::200:87ff:fe28:a0e0%5 from fe80::201:4aff:fe18:26c7%5 with 32 bytes of data:

Reply from fe80::200:87ff:fe28:a0e0%5: bytes=32 time<1ms

Reply from fe80::200:87ff:fe28:a0e0%5: bytes=32 time<1ms

Reply from fe80::200:87ff:fe28:a0e0%5: bytes=32 time<1ms

Reply from fe80::200:87ff:fe28:a0e0%5: bytes=32 time<1ms

Ping statistics for fe80::200:87ff:fe28:a0e0%5:

Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),

Approximate round trip times in milli-seconds:

Minimum = 0ms, Maximum = 0ms, Average = 0ms

Basic Configuration: XP/2003 (11)

- Traceroute in XP/2003
- **tracert6 [-d] [-h maximum_hops] [-w timeout] [-s srcaddr] target_name**
  - d Do not resolve addresses to hostnames
  - h max_hops Maximum number of hops to search for target
  - w timeout Wait timeout milliseconds for each reply
  - s srcaddr Source address to use
  - r Use routing header to test reverse route also
- **tracert** command defaults to IPv6 when available

Basic Configuration: XP/2003 (12)

- **Examples of traceroute in XP/2003:**
- **tracert www.lacnic.net**

```
Tracing route to lacnic.net [2001:12ff:0:2::15] over a maximum of 30 hops:
1    1 ms    <1 ms    <1 ms  gr2000-00.consulintel.euro6ix.org [2001:800:40:2a05::1]
2   <1 ms     *        1 ms  2001:800:40:2f02::1
3    4 ms     1 ms     1 ms  2001:800:40:2f01::2
4   10 ms     4 ms     4 ms  data-to-tid.tid.euro6ix.org [2001:800:40:2f1a::2]
5  200 ms   189 ms   189 ms  3ffe:80a::1
6  388 ms   390 ms   388 ms  v6gw.isc.registro.br [2001:4f8:0:1::10:2]
7  396 ms   396 ms   387 ms  lacnic.net [2001:12ff:0:2::15]
Trace complete.
```

Basic Configuration: XP/2003 (13)

- Adding an Address:
- **netsh interface ipv6 add address InterfaceNameOrIndex IPv6Address [[type=]unicast|anycast] [[validlifetime=]Minutes|infinite] [[preferredlifetime=]Minutes|infinite] [[store=]active|persistent]**
- Example: netsh interface ipv6 add address 5 2001:db8::2 type=unicast validlifetime=infinite preferredlifetime=10m store=active
- Check the configuration using **ipv6 if 5**

Basic Configuration: XP/2003 (14)

- Modifying the options in an already configured address:
- **netsh interface ipv6 set address** [interface=]<string> [address=]<IPv6 address> [[type=]unicast|anycast] [[validlifetime=]<integer>|infinite] [[preferredlifetime=]<integer>|infinite] [[store=]active|persistent]
- Example: netsh interface ipv6 set address 5 2001:db8::2 preferredlifetime=infinite
- Check the configuration using **ipv6 if 5**

Basic Configuration: XP/2003 (15)

- Deleting an Address:
- **netsh interface ipv6 delete address** [interface=]<string> [address=]<IPv6 address> [[store=]active|persistent]
- Example: netsh interface ipv6 delete address 5 2001:db8::2 store=persistent
- To check the configuration using **ipv6 if 5**

Basic Configuration: XP/2003 (16)

- Adding a Static Route:
- **netsh interface ipv6 add route [prefix=]IPv6Address/Integer [[interface=]String] [[nexthop=]IPv6Address] [[siteprefixlength=]Integer] [[metric=]Integer] [[publish=]{no | yes | immortal}] [[validlifetime=]{Integer | infinite}] [[preferredlifetime=]{Integer | infinite}] [[store=]{active | persistent}]**
- Example: netsh interface ipv6 add route 2002::/16 5 fe80::200:87ff:fe28:a0e0 store=persistent
- Above, fe80::200:87ff:fe28:a0e0 is the default gateway

Basic Configuration: XP/2003 (17)

- Showing Routes:
- **netsh interface ipv6 show routes [[level=]{normal | verbose}] [[store=]{active | persistent}]**
- Example: netsh interface ipv6 show routes

```
Querying active state...
Publish  Type      Met  Prefix                    Idx  Gateway/Interface Name
-------  --------  ---  --------------------------  ---  --------------------
no       Manual      0  2002::/16                   5  fe80::200:87ff:fe28:a0e0
no       Autoconf    8  2001:db8::/64               5  Local Area Connection
no       Autoconf  256  ::/0                        5  fe80::200:87ff:fe28:a0e0
```

Basic Configuration: XP/2003 (18)

- Deleting a Static Route:
- **netsh interface ipv6 delete route [prefix=]<IPv6 address>/<integer> [interface=]<string> [[nexthop=]<IPv6 address>] [[store=]active|persistent]**
- Example: netsh interface ipv6 delete route 2002::/16 5 fe80::200:87ff:fe28:a0e0 store=persistent
- Check using **netsh interface ipv6 show routes**

Basic Configuration: XP/2003 (19)

- Adding a Static DNS Server:
- **netsh interface ipv6 add dns [[interface=]String] [[address=]IPv6Address] [[index=]Integer]**

- Example: netsh interface ipv6 add dns “Local area network” 2001:7f9:1000:1::947c 1
- The index represent the position of the DNS server just configured in the DNS servers lists

Basic Configuration: XP/2003 (20)

- Showing DNS servers:
- **netsh interface ipv6 show dns [[interface=]string]**

- Example: netsh interface ipv6 show dns

```
DNS servers in LAN interface
Index       DNS server
------  ----------------------------------------
1           2001:7f9:1000:1::947c
2           2001:7f9:1000:1::947c
```

Basic Configuration: XP/2003 (21)

- Deleting a Static DNS server:
- **netsh interface ipv6 delete dns [interface=]<string> [[address=]<IPv6 address>|all]**

- Example: netsh interface ipv6 delete dns “Local area network” all
- Check using **netsh interface ipv6 show dns**

Basic Configuration: W2K (1)

- Basic Commands in W2K
- Useful to obtain information about the status and to configure interfaces, addresses, caches, routes, and so on
- Two groups of commands:
  - **Net.exe**
    - Can be used to stop and start the IPv6 protocol
    - Restarting the IPv6 protocol causes it to reinitialize as if the computer were rebooting, which might change interface numbers
  - **ipv6.exe** (covers up to Windows XP SP2)
    - All Microsoft IPv6 protocol configuration is done with the ipv6.exe tool
    - Some changes are not persistent (values lost with each reboot). It is possible to execute a configuration in a .cmd script in each boot

Basic Configuration: W2K (2)

- "Net" Commands
  - Net.exe has many subcommands, each with its own set of arguments and options. Only the following commands are directly relevant to IPv6:
    - net stop tcpip6: Stops the IPv6 protocol and unloads it from memory. This command fails if there are any open IPv6 sockets
    - net start tcpip6: Starts the IPv6 protocol if it was stopped. If a new Tcpip6.sys driver file is present in the %systemroot%\System32\Drivers directory, it is loaded
- "ipv6" Commands
  - ipv6.exe has many subcommands, each with its own set of arguments and options:
    - ipv6 if [if#]
    - ipv6 ifc if# [forwards] [advertises] [-forwards] [-advertises] [mtu #bytes] [site site-identifier]
    - ipv6 ifd if#
    - ipv6 nc [if# [address]]
    - ipv6 ncf [if# [address]]
    - ipv6 rc [if# address]
    - ipv6 rcf [if# [address]]
    - ipv6 bc
    - ipv6 adu if#/address [lifetime VL[/PL]] [anycast] [unicast]
    - ipv6 spt
    - ipv6 spu prefix if# [lifetime L]
    - ipv6 rt
    - ipv6 rtu prefix if#[/nexthop] [lifetime L] [preference P] [publish] [age] [spl site-prefix-length]
- Further information at: http://msdn.microsoft.com/downloads/sdks/platform/tpipv6/start.asp

Basic Configuration: W2K (3)

- Ping in W2K
  - **ping6** destination-address
- Traceroute in W2K
  - **tracert6** destination-address

Basic Configuration: W2K (4)

- Adding an Address:
- **ipv6 adu IfIndex/Address [life ValidLifetime[/PrefLifetime]] [anycast] [unicast]**
- Example: ipv6 adu 2/2001:db8::1

- Check the configuration using **ipv6 if 2**

Basic Configuration: W2K (5)

- Deleting an Address:
- **ipv6 adu IfIndex/Address [life ValidLifetime[/PrefLifetime]] [anycast] [unicast]**
- Example: ipv6 adu 2/2001:db8::1 life 0

- Check the configuration using **ipv6 if 2**

Basic Configuration: W2K (6)

- Adding a Static Route:
- **ipv6 rtu Prefix IfIndex[/Address] [lifetimeValid[/Preferred]] [preference P] [publish] [age] splSitePrefixLength]**
- Example: ipv6 rtu ::/0 2/::192.168.0.102
  - Above, ::192.168.0.102 is the default gateway

Basic Configuration: W2K (7)

- Showing Routes:
- **ipv6 [-v] rt**

Basic Configuration: W2K (8)

- Deleting a Static Route:
- **ipv6 rtu Prefix IfIndex[/Address] [lifetimeValid[/Preferred]] [preference P] [publish] [age] SpISitePrefixLength]**
- Example: ipv6 rtu ::/0 2/::192.168.0.102 pub life 0
  - Above, ::192.168.0.102 is the default gateway

- Check using **ipv6 rt**

Basic Configuration: W2K (9)

- **Manual Tunnel**
- Use **ipv6 adu** and **ipv6 rtu**
- Example:
- ipv6 rtu ::/0 2/::200.20.20.20
- ipv6 adu 2/2001:db8:0a20:0011::2
  - 200.20.20.20 is the remote endpoint address
  - 2001:db8:0a20:0011::2 is the local address

- Check using **ipv6 if 2** and **ipv6 rt**

Basic Configuration: Linux (1)

- **Basic Commands (1)**
  - ifconfig
  - ping6 <hostcondirIPv6>|<dirIPv6>|[-I <interface>] <link-local -ipv6address>
  - traceroute6 <hostcondirIPv6>|<dirIPv6>
  - tracepath6 <hostcondirIPv6>|<dirIPv6>
  - tcpdump

Basic Configuration: Linux (2)

 # ping6 ::1
PING ::1(::1) 56 data bytes
64 bytes from ::1: icmp_seq=1 ttl=64 time=0.047 ms
64 bytes from ::1: icmp_seq=2 ttl=64 time=0.039 ms
64 bytes from ::1: icmp_seq=3 ttl=64 time=0.042 ms
64 bytes from ::1: icmp_seq=4 ttl=64 time=0.020 ms
--- ::1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2999ms
rtt min/avg/max/mdev = 0.020/0.037/0.047/0.010 ms

 # ping6 -I eth0 fe80::2e0:81ff:fe05:4657
PING fe80::2e0:81ff:fe05:4657(fe80::2e0:81ff:fe05:4657) from ::1 eth0: 56 data bytes
64 bytes from fe80::2e0:81ff:fe05:4657: icmp_seq=1 ttl=64 time=0.056 ms
64 bytes from fe80::2e0:81ff:fe05:4657: icmp_seq=2 ttl=64 time=0.055 ms
64 bytes from fe80::2e0:81ff:fe05:4657: icmp_seq=3 ttl=64 time=0.048 ms
64 bytes from fe80::2e0:81ff:fe05:4657: icmp_seq=4 ttl=64 time=0.128 ms
--- fe80::2e0:81ff:fe05:4657 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2997ms
rtt min/avg/max/mdev = 0.048/0.071/0.128/0.034 ms

Basic Configuration: Linux (3)

- **Basic Commands (2)**
- Adding an Address:

/sbin/ip -6 addr add <ipv6address>/<prefixlength> dev <interface>

/sbin/ifconfig <interface> inet6 add <ipv6address>/<prefixlength>

- Deleting an Address:

/sbin/ip -6 addr del <ipv6address>/<prefixlength> dev <interface>

/sbin/ifconfig <interface> inet6 del <ipv6address>/<prefixlength>

Basic Configuration: Linux (4)

- **Static Routes**
- Showing Routes:

/sbin/ip -6 route show [dev <device>]

/sbin/route -A inet6

- Adding a Default Route via a Gateway:

/sbin/ip -6 route add <ipv6network>/<prefixlength> via <ipv6address>
[dev <device>]

#/sbin/route -A inet6 add <ipv6network>/<prefixlength> gw <ipv6address>
[dev <device>]

Basic Configuration: Linux (5)

- Deleting a Default Route via a Gateway:

/sbin/ip -6 route del <ipv6network>/<prefixlength> via <ipv6address>
[dev <device>]

/sbin/route -A inet6 del <network>/<prefixlength> [dev <device>]

- Adding a Route via an interface:

/sbin/ip -6 route add <ipv6network>/<prefixlength> dev <device> metric 1

/sbin/route -A inet6 add <network>/<prefixlength> dev <device>

- Deleting a Route via an interface:

/sbin/ip -6 route del <ipv6network>/<prefixlength> dev <device>

/sbin/route -A inet6 del <network>/<prefixlength> dev <device>

Basic Configuration: Linux (6)

- Showing Neighbors Table

ip -6 neigh show [dev <device>]

- Adding a Neighbor

ip -6 neigh add <IPv6 address> lladdr <link-layer address> dev <device>

- Deleting a Neighbor

ip -6 neigh del <IPv6 address> lladdr <link-layer address> dev <device>

Basic Configuration: BSD (1)

- **Basic Commands**
- Adding an Address

#>ifconfig <interface> inet6 add <dir. IPv6>

- Deleting an Address

#>ifconfig <interface> inet6 del <dir. IPv6>

Basic Configuration: BSD (2)

- **Persistent Configuration:**

  Edit file /etc/rc.conf:

  ipv6_enable=”YES”

  ipv6_ifconfig_rl0=”2001:618:10:4::4 prefixlen 64”

  In /etc/defaults/rc.conf you can find the different parameters to configure and the defaults values

- To make apply changes in rc.conf you must reboot

Basic Configuration: BSD (3)

- **Static Routes**
- Adding a Default Route:

#>route –n add -inet6 default <dir. IPv6>

- Deleting a Default Route:

#>route –n del -inet6 default

Basic Configuration: Exercise 1

- **ping6** to link-local Address of a Neighbor
- At the same time, capture packets using tcpdump:

  # tcpdump -t -n -i eth0 -s 512 -vv ip6 or proto ipv6

- Another way to show addresses:

  # /sbin/ip -6 addr show dev eth2

  # ifconfig eth0

- Add and delete the address:

  2001:800:40:2a09:1:2:3:4 in the eth0 interface

Basic Configuration: Exercise 2

Linux

- Add and delete a route through a gateway
- Add and delete a route through an interface
- Show neighbors table
- Add and delete a neighbor

BSD

- Add and delete a route through a gateway

Stateless Autoconfiguration (1)

- RFC 2462: IPv6 Stateless Address Autoconfiguration
- [STATELESS] Provides information about:
  - Network Prefix
  - Routing
- Global Addresses are built by two elements
  - Interface Identifier (64 bits based on EUI-64, and usually obtained from IEEE 48 bit MAC Address)
  - Prefix obtained from de Prefix Information Options contain in the Router Advertisements
- Easing the Configuration
  - The user does not need to configure any network parameter in order to obtain native IPv6 connectivity

Stateless Autoconfiguration (2)

- In Windows XP/2003 hosts, it is enabled by default
- **ipconfig** o **ipv6 if** to check which is the autoconfigured address
- Example: **2001:db8:10:10**:201:4aff:fe18:26c7
  - Interface Identifier EUI-64 obtained from this MAC address: 4aff:fe18:26c7
  - Prefix provided by the router: **2001:db8:10:10**

Stateless: Exercise 1 (1)

- Configure a Linux router to send RA packets to the network
- Get a ‘radvd’ daemon for the used Linux distribution
  - http://www.rpmfind.net/linux/rpm2html/search.php?query=radvd&submit=Search+...
- Install it
- Enable routing capabilities
  - echo 1 > /proc/sys/net/ipv6/conf/all/forwarding
- Edit /etc/radvd.conf file with the following content:

Stateless: Exercise 1 (2)

```
interface eth00
{
        AdvSendAdvert on;


        MinRtrAdvInterval 3;
        MaxRtrAdvInterval 5;

        AdvHomeAgentFlag off;

        prefix 2001:8500:40:2a30::/64
        {
          AdvOnLink off;
          AdvAutonomous on;
          AdvRouterAddr off;
        };
};
```

Stateless: Exercise 1 (3)

- Launch radvd daemon
  - Radvd
- Check that other computers in the network are autoconfigured thanks to our radvd daemon

Stateful Autoconfiguration (1)

- [STATEFUL] Similar to DHCP in IPv4
- An IPv6 address is provided. This address can be different each time a node gets connected
- Provides information complementary to the stateless one
  - DNS Server (could be IPv6)
  - domain name
  - NTP server (could be IPv6)
  - SIP server (could be IPv6)
  - SIP domain name
  - Prefix delegation
  - Etc.
- DHCPv6 implementations are still not available in the most common OSs
  - An specific installation of a DHCPv6 application is needed (server and /or client)
    - http://klub.com.pl/dhcpv6/
    - http://sourceforge.net/projects/dhcpv6-linux/

Stateful: Exercise 1 (1)

- Configuring a DHCPv6 server on Linux
  - Obtain the DHCPv6 implementation for Linux from: http://klub.com.pl/dhcpv6/dibbler/dibbler-0.4.0-linux.tar.gz
  - Untar the file
    - tar –xvzf dibbler-0.4.0-linux.tar.gz
  - Make these directories
    - /var/lib/dibbler
    - /etc/dibbler

Stateful: Exercise 1 (2)

- Edit the content of file server.conf
  - log-level 7
  - log-mode short
  - iface eth0 {
  - T1 1000
  - T2 2000
  - class {
  - pool 2001:3820:40:2a03::10-2001:3820:40:2a03:ffff:ffff:ffff:ffff
  - }
  - option dns-server 2001:800:40:2a03::2, 2001:800:40:2a04::2
  - option domain example.com, test1.example.com
  - }
- The given addresses will be in the prefix 2001:3820:40:2a03::/64 starting from 2001:3820:40:2a03::10
- Copy the file server.conf in the directory /etc/dibbler
- Launch dhcpv6 server
  - dhcpv6-server run

Stateful: Exercise 2 (1)

- Configure DHCPv6 client in Linux
  - Get a DHCPv6 implementation for Linux from: http://klub.com.pl/dhcpv6/dibbler/dibbler-0.4.0-linux.tar.gz
  - Untar the file
    - tar –xvzf dibbler-0.4.0-linux.tar.gz
  - Create the directories
    - /var/lib/dibbler
    - /etc/dibbler

Stateful: Exercise 2 (2)

- Edit the content of file server.conf
  - log-mode short
  - iface eth0
  - {
  - IA
  - option dns-server
  - option domain
  - }
- With this configuration you get
  - An IPv6 address
  - DNS servers
  - Domain name
- Copy client.conf file in the directory /etc/dibbler
- Launch dhcpv6 client
  - dhcpv6-client run
- With ‘ifconfig eth0’ you can check if you have got an IPv6 address
- In /etc/resolv file you can check the DNS servers obtained
- Note that you don’t get routing information, so you can’t make ping6
  - The routing information is obtained by means of stateless autoconfiguration (RA)

Privacy (1)

- RFC 3041: Privacy Extensions for Stateless Address Autoconfiguration in IPv6
- Extension of Stateless Autoconfiguration
- It generates a global address that changes over time
- It makes more difficult to identify when different addresses used in different transactions actually correspond to the same node

Privacy (2)

- In Windows XP/2003 hosts, it is enabled by default
- **ipconfig** o **ipv6 if** to check which is the autoconfigured address
- There are two ways to disable it:
  1. **netsh interface ipv6 set privacy state=disabled store=persistent**
  2. **ipv6 [-p] gpu UseTemporaryAddresses no**
- To check the change: “disable” and “enable” the physical interface on Windows Network Connection, then **ipconfig** o **ipv6 if**

Privacy (3)

- **Additional options with netsh command:**
- netsh interface ipv6 set privacy
  [[state=]enabled|disabled]
  [[maxdadattempts=]<integer>]
  [[maxvalidlifetime=]<integer>]
  [[maxpreferredlifetime=]<integer>]
  [[regeneratetime=]<integer>]
  [[maxrandomtime=]<integer>]
  [[randomtime=]<integer>] [[store=]active
  |persistent]

Part 3

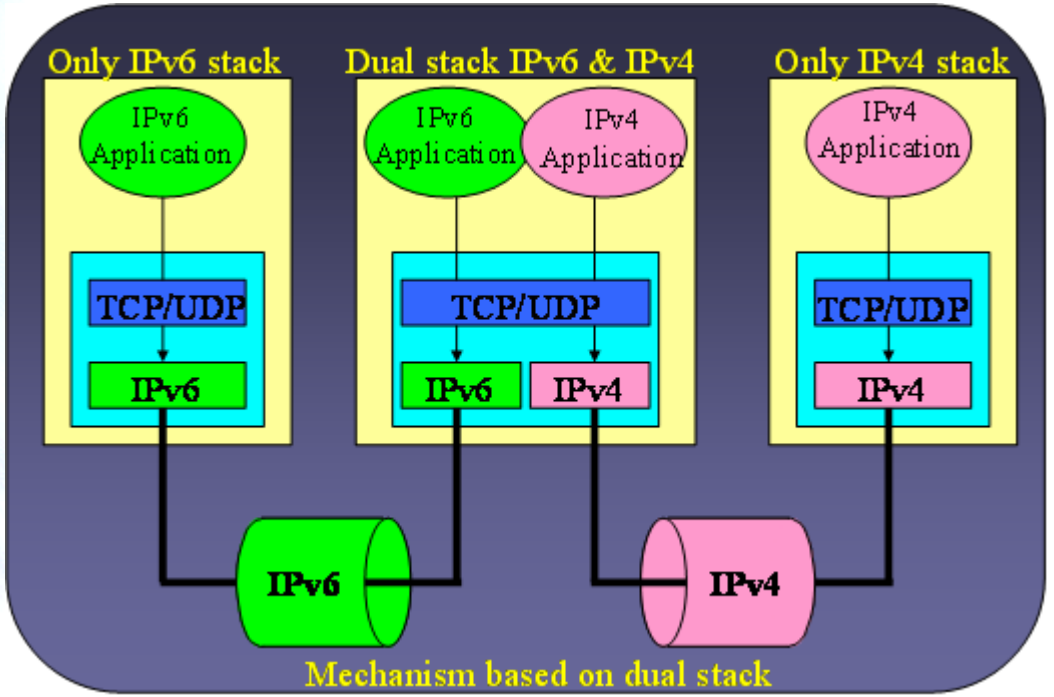
Transition Mechanisms Configuration

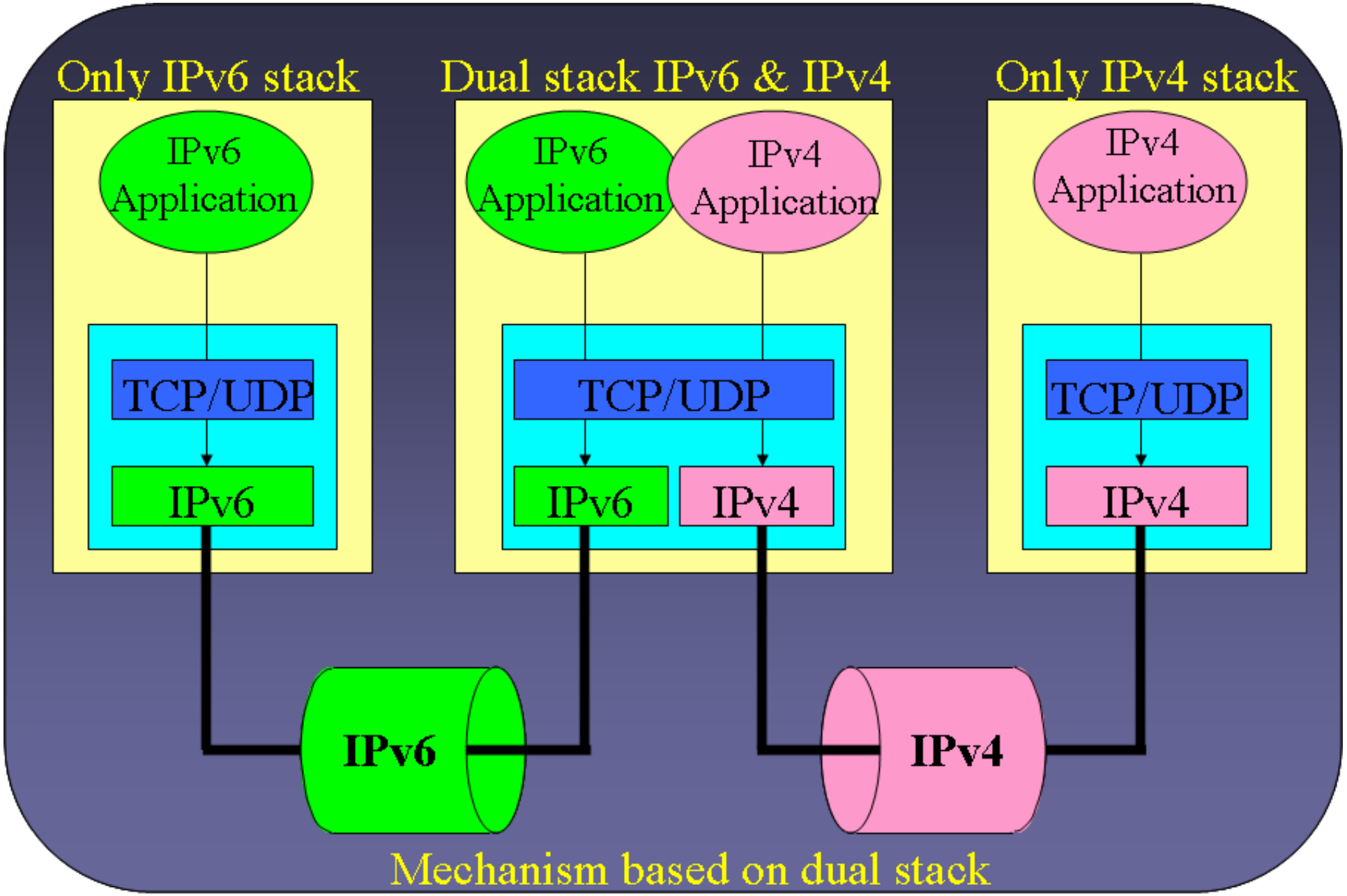
Transition Mechanisms

- IPv6 has been designed for easing the transition and coexistence with IPv4
- Several strategies have been designed for coexisting with IPv4 hosts
  - Dual stack: Simultaneous support for both IPv4 and IPv6 stacks
  - Tunnels: IPv6 packets encapsulated in IPv4 ones
    - This is the commonest choice
  - Translation:This should be the last choice because it isn't perfect

Dual Stack

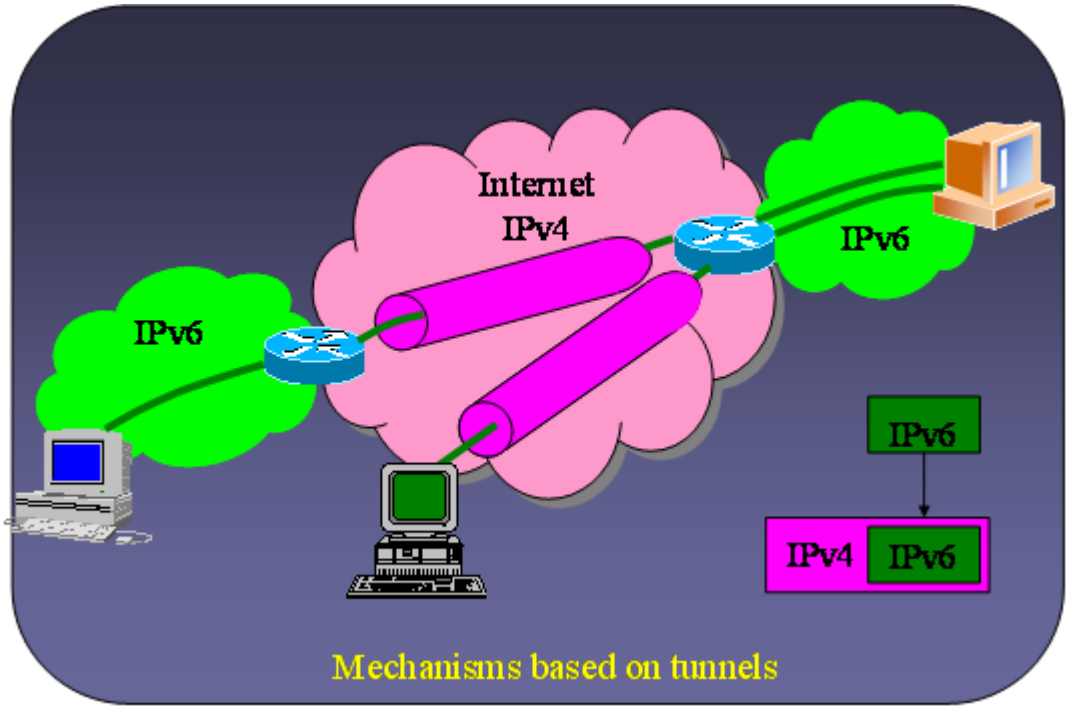
- All the hosts have both stacks IPv4 & IPv6
- IPv6-only communications ==> IPv6 stack, assuming IPv6 network support
- IPv4-only communications ==> IPv4 stack

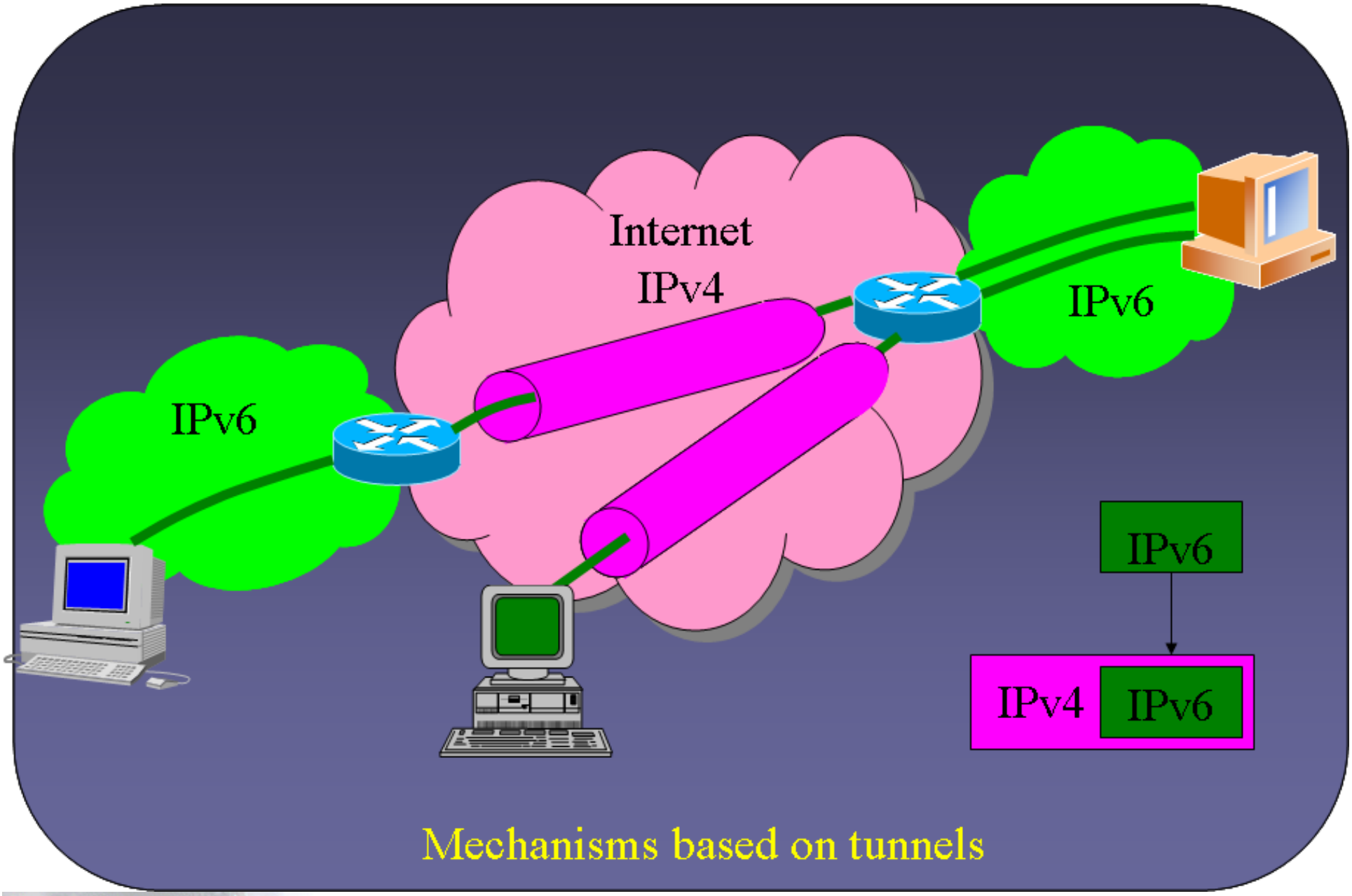



Only IPv6 stack
Dual stack IPv6 & IPv4
Only IPv4 stack
IPv6 Application
IPv6 Application
IPv4 Application
IPv4 Application
TCP/UDP
TCP/UDP
TCP/UDP
IPv6
IPv6
IPv4
IPv4
IPv6
IPv4
Mechanism based on dual stack

Tunnels: IPv6 in IPv4 (1)

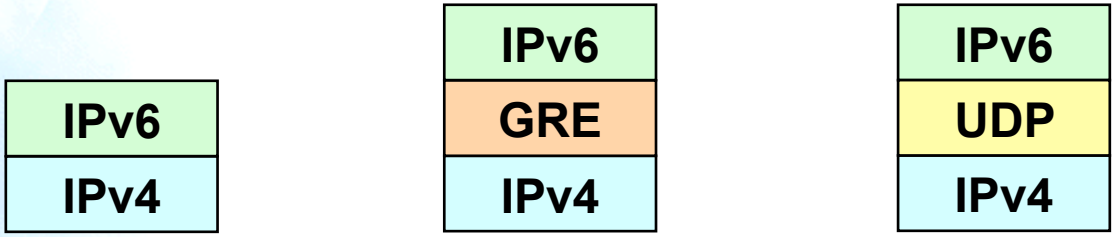
- It is used to provide IPv6 connectivity in IPv4-only networks
- The IPv6 packets are encapsulated into IPv4 packets
- There are different ways to make the encapsulation
  - 6in4, 6to4, 6over4, UDP, etc.
- The resulting packets flow through IPv4 networks towards the tunnel end point (TEP)




Internet
IPv4
IPv6
IPv6
IPv6
IPv4
IPv6
Mechanisms based on tunnels

Tunnels IPv6 in IPv4 (2)

- There are different ways for encapsulating the IPv6 packets into IPv4 ones



- Same for IPv4 being used in IPv6-only networks

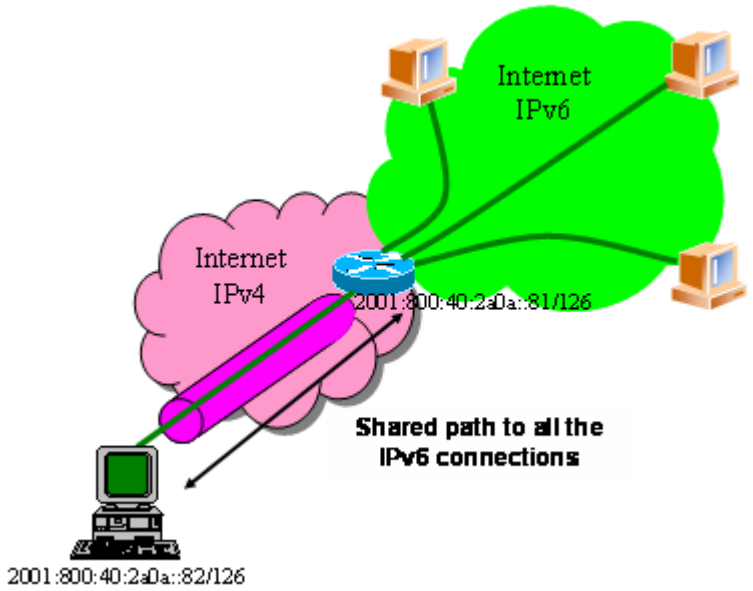
Tunnels IPv6 in IPv4 (3)

- Some transition mechanism based on tunnels:
  - 6in4 (*) [6in4]
  - TB (*) [TB]
  - TSP [TSP]
  - 6to4 (*) [6to4]
  - Teredo (*) [TEREDO], [TEREDOC]
  - Automatic tunnels[TunAut]
  - ISATAP [ISATAP]
  - 6over4 [6over4]
  - AYIYA [AYIYA]
  - Silkroad [SILKROAD]
  - DSTM [DSTM]
  - Softwires (*)

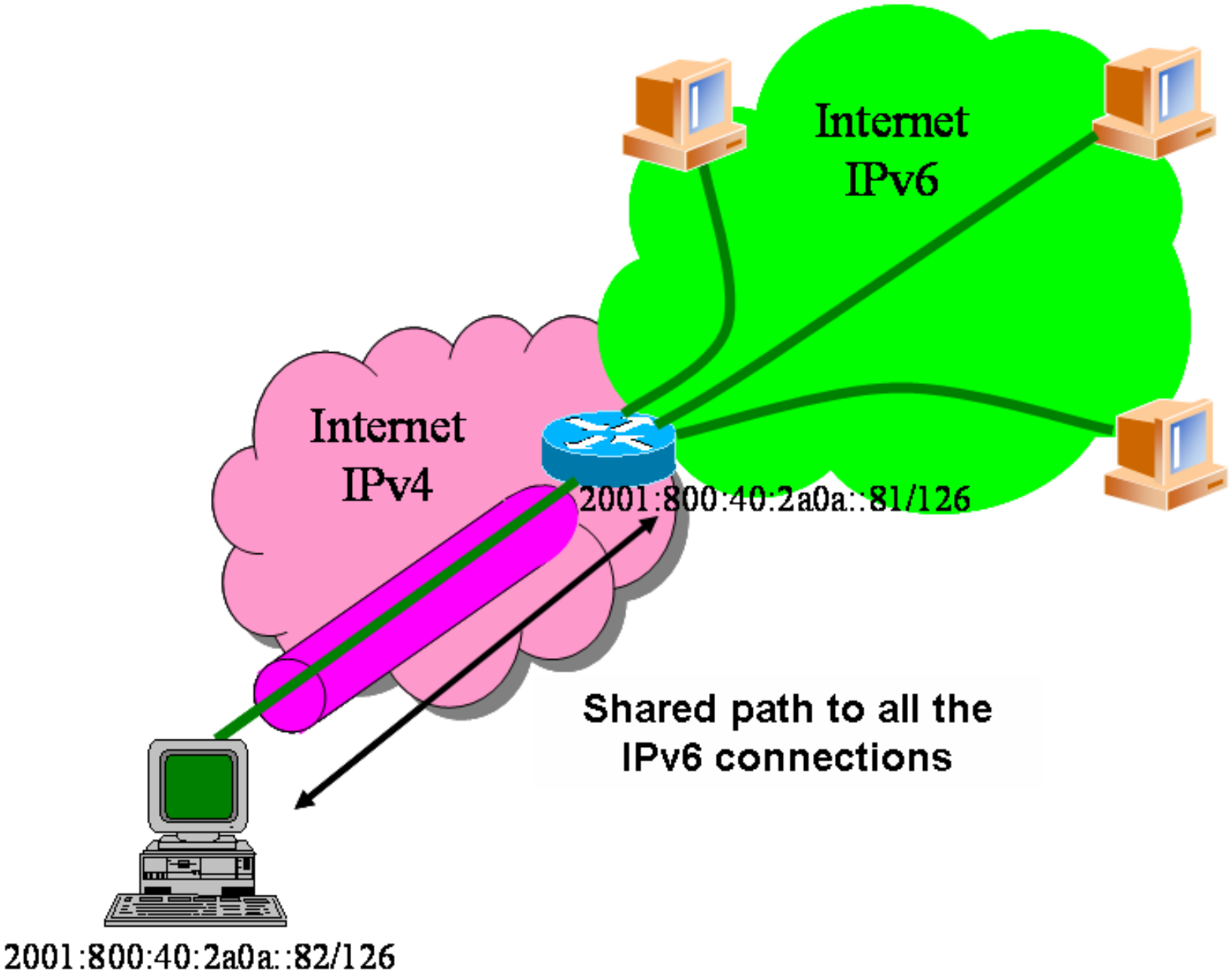
- (*) Commoner mechanisms and explained in depth in the following slides

6in4 Tunnels

- It encapsulates directly the IPv6 packet into the IPv4 packet
- It is usually used between:
  - end host ==> router
  - router ==> router



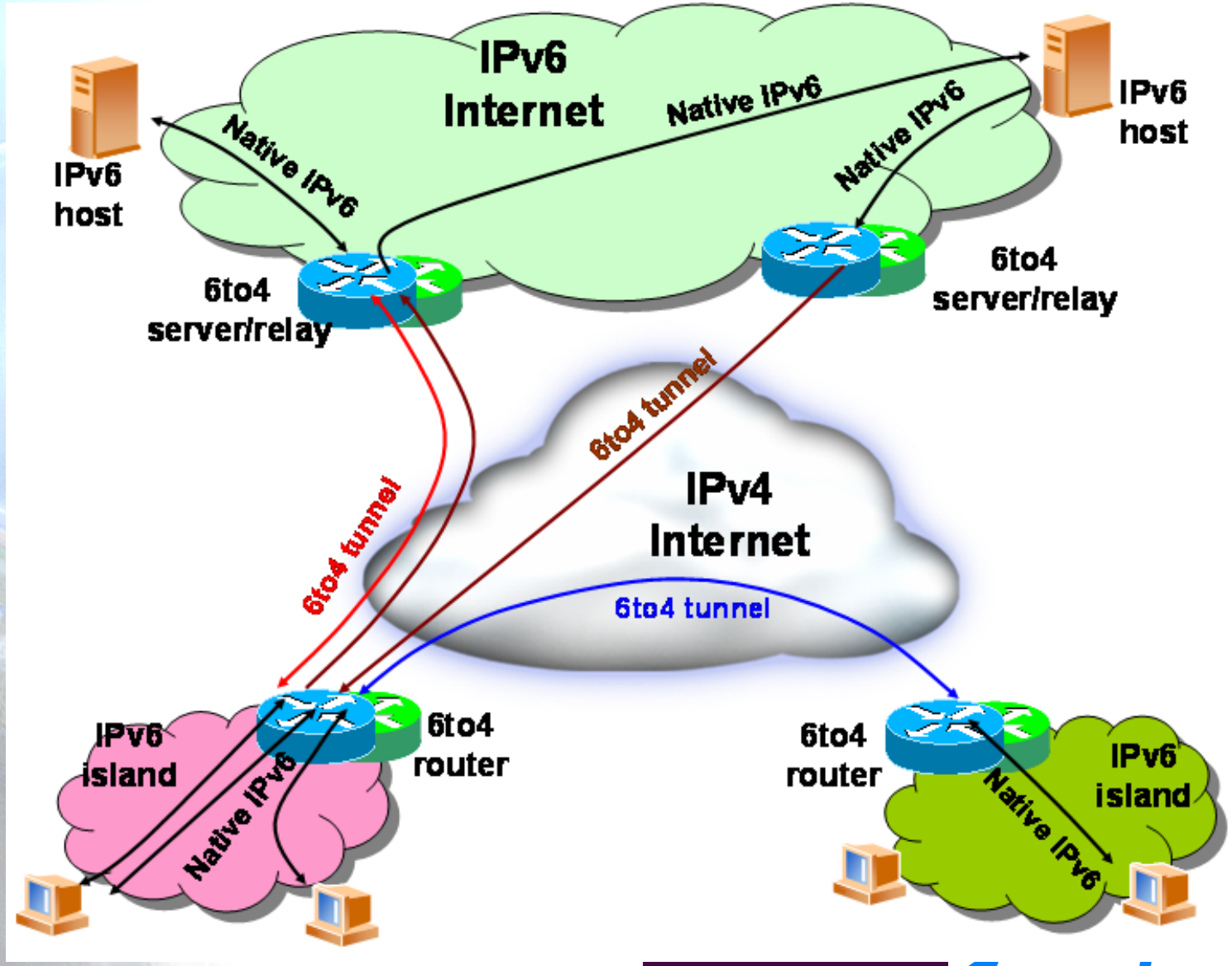
- However, it is also possible for
  - end host ==> end host
- From the point of view of IPv6 the tunnel is considered as a point-to-point link
  - Only an IPv6 network-hop although several IPv4-hops exist in the path
- The IPv6 addresses of both tunnel-ends belong to the same prefix
- All the IPv6 connections of the end-host flow always through the router located at the tunnel-end-point
- The 6in4 tunnels can be built from end-hosts located behind a NAT box
  - It is essential that the NAT implementation supports “proto-41 forwarding” [PROTO41] to let the IPv6-encasulated packets traverse the NAT box


Internet
IPv6
Internet
IPv4
2001:800:40:2a0a::81/126
Shared path to all the
IPv6 connections
2001:800:40:2a0a::82/126

Tunnel Broker

- The 6in4 tunnels require the manual configuration of the devices involved in the tunnel creation
- To easy the address assignment and the IPv6 tunnel creation, the Tunnel Broker (TB) concept has been developed
  - It is a intermediate host which the end user is connected, usually by using a web browser
- The user asks to the TB the creation of an IPv6 tunnel. The TB assigns to the user an IPv6 address and gives to the user instructions for building the tunnel in the user's side
- The TB also configures the router, which is the TEP for the end user
- In http://www.ipv6tf.org/using/connectivity/test.php exists a list of available TBs
- TSP [TSP] is a special case of TB because it is based on an application installed in the user's host which contacts to the TSP server to built the IPv6 tunnel. However, the concept is similar to the one previously enounced

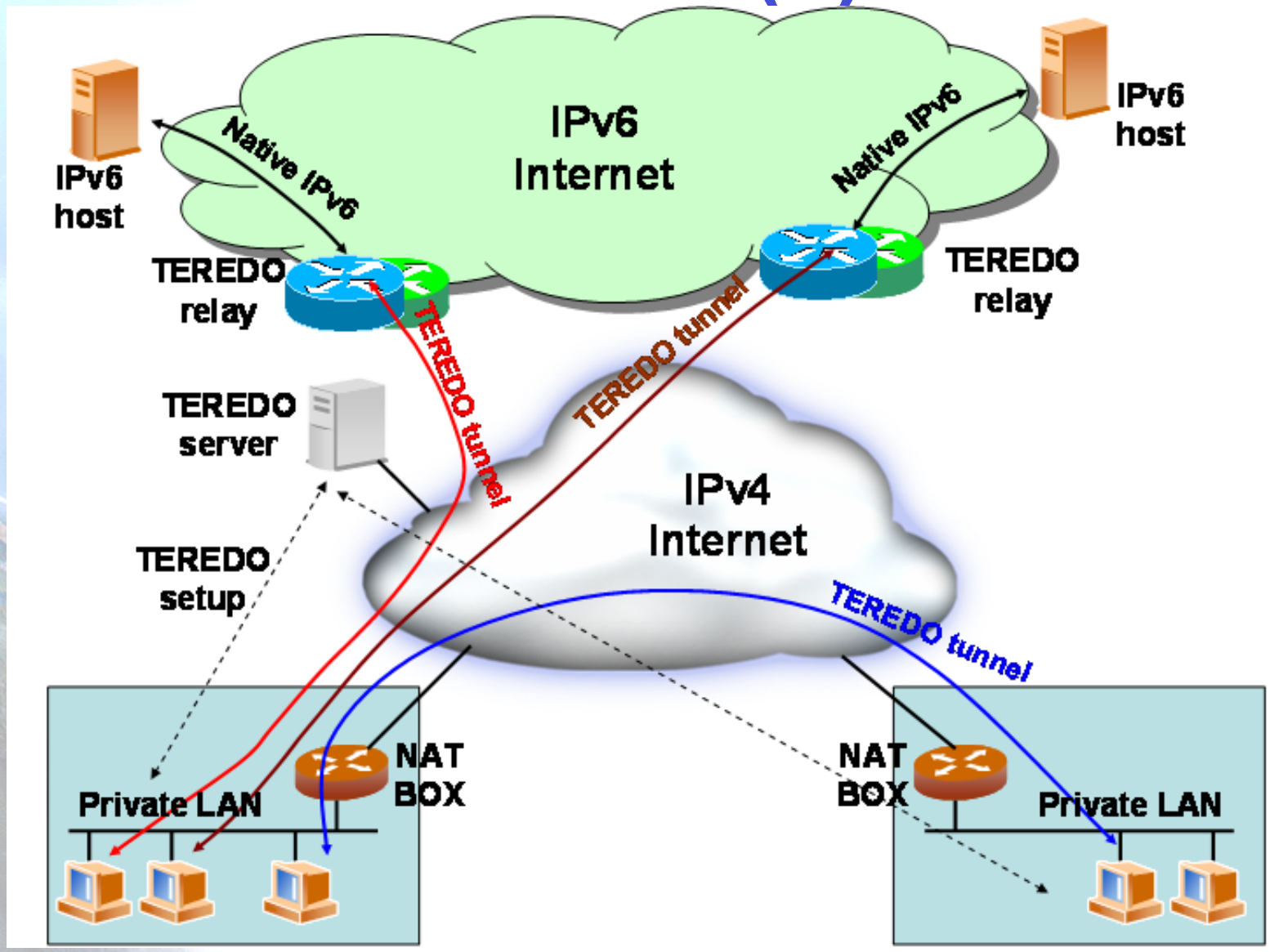
6to4 Tunnels (1)



6to4 Tunnels (2)

- IPv6 packets are encapsulated into IPv4 ones, in a similar way than the 6in4 tunnels
- Differences:
  - The user's IPv6 address does not depend on the router used to get IPv6 connected but on the public IPv4 used by the user
    - Prefix 2002::/16
  - All the user's outgoing IPv6 packets are always sent to the same "6to4 relay". However the user's incoming IPv6 packets could come from different "6to4 relays"
- IPv4 anycast prefix:
  - 192.88.99.1

Teredo (1)

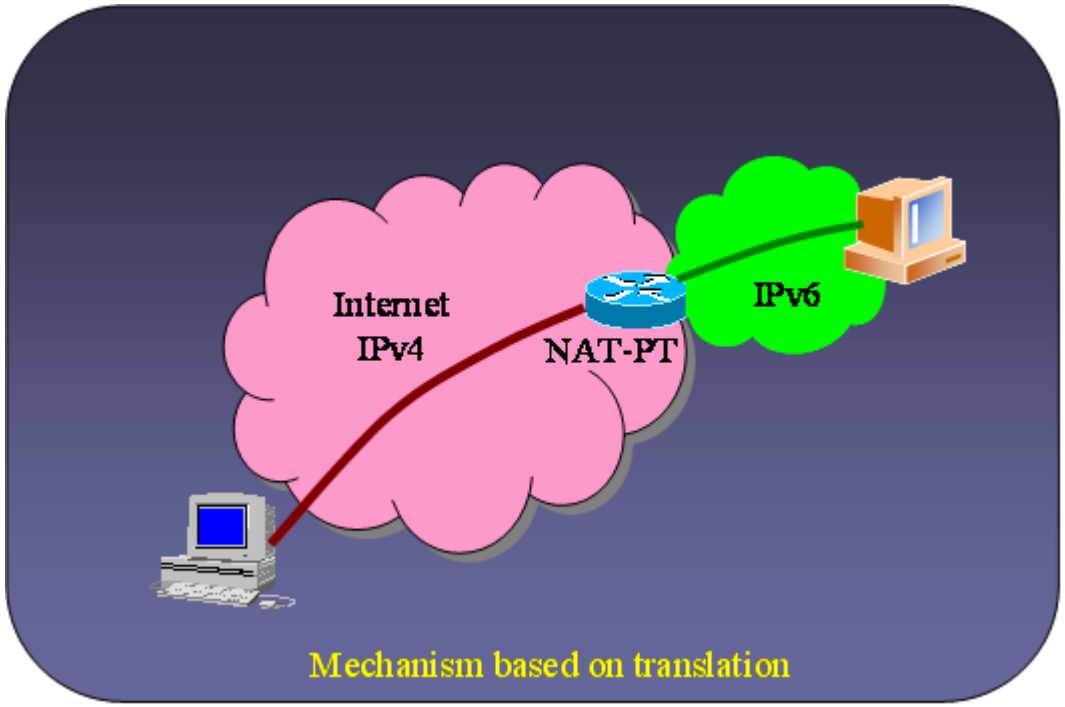

IPv6
host
Native IPv6
IPv6
Internet
Native IPv6
IPv6
host
TEREDO
relay
TEREDO tunnel
TEREDO tunnel
TEREDO
relay
TEREDO
server
IPv4
Internet
TEREDO
setup
TEREDO tunnel
Private LAN
NAT
BOX
NAT
BOX
Private LAN

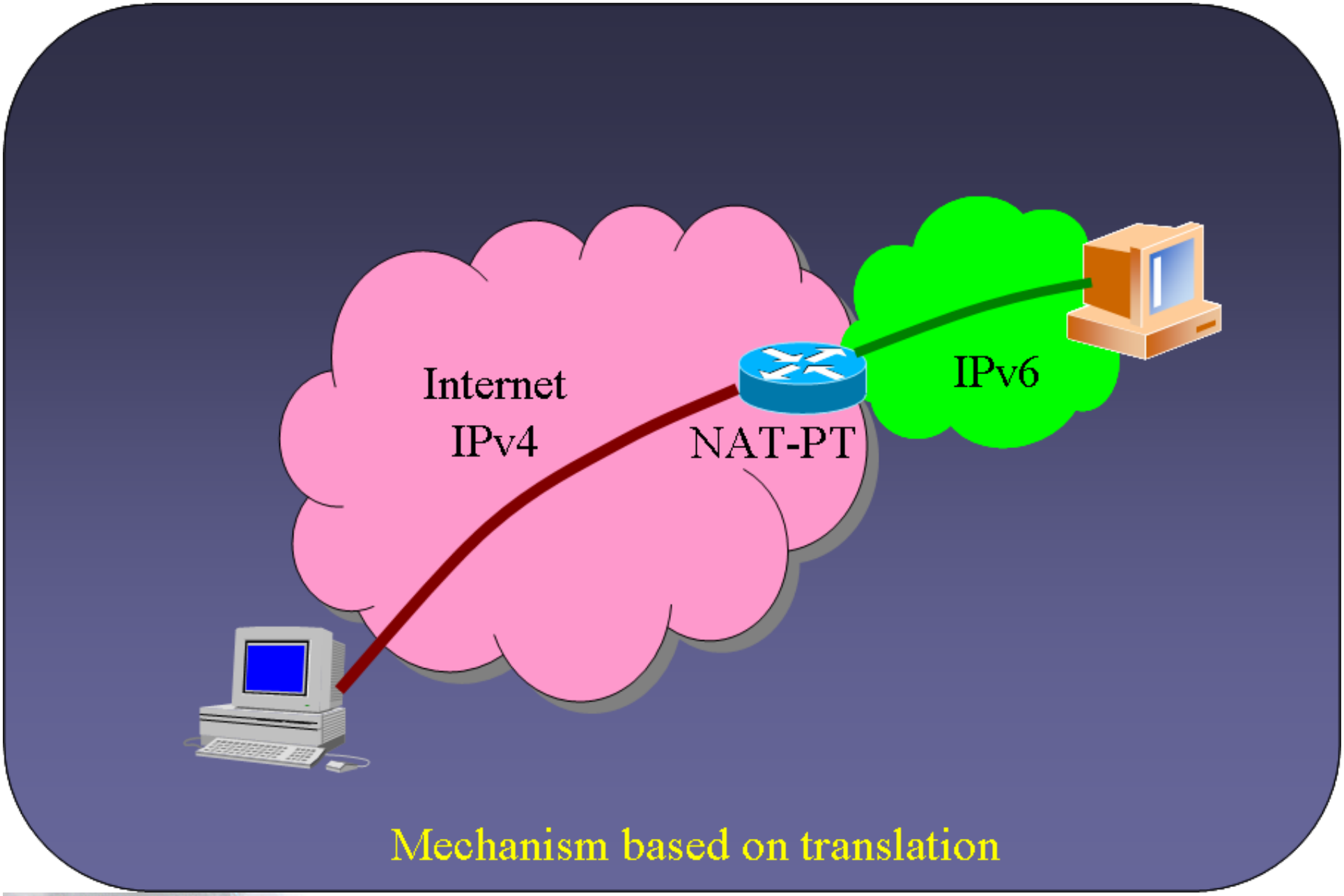
Teredo (2)

- Teredo [TEREDO] [TEREDOC] is thought for providing IPv6 to hosts that are located behind a NAT box that is not "proto-41 forwarding"
  - It encapsulates the IPv6 packets into UDP/IPv4 packets
- It only works in the following NAT types [STUN]:
  - Full Cone
  - Restricted Cone
- It does not work in the following NAT type:
  - Symmetric
- Teredo uses different agents to work:
  - Teredo Server
  - Teredo Relay
  - Teredo Client
- The user configures in its host a Teredo Server which provides an IPv6 address from the 2001:0000::/32 prefix and such an address is based on the user's public IPv4 address and used UDP port
  - If the Teredo Server is also a Teredo Relay, the user has also IPv6 connectivity with any IPv6 hosts
  - Otherwise, the user only has IPv6 connectivity with other Teredo users
- Microsoft currently provides public Teredo Servers for free, but not Teredo Relays

Translation

- There are several solutions, but all of them try to translate IPv4 packets into IPv6 and vice-versa
  - [SIT], [BIS], [TRT], [SOCKSv64]
- The commonest is NAT-PT [NATPT], [NATPTIMPL]
  - An intermediate node (router) modifies the IPv4 headers to convert them into IPv6 headers
  - The treatment of the packets is complex
- It is the worst solution because the translation is not perfect and it requires ALGs support, in the same way that IPv4-NATs
  - DNS, FTP, VoIP, etc.





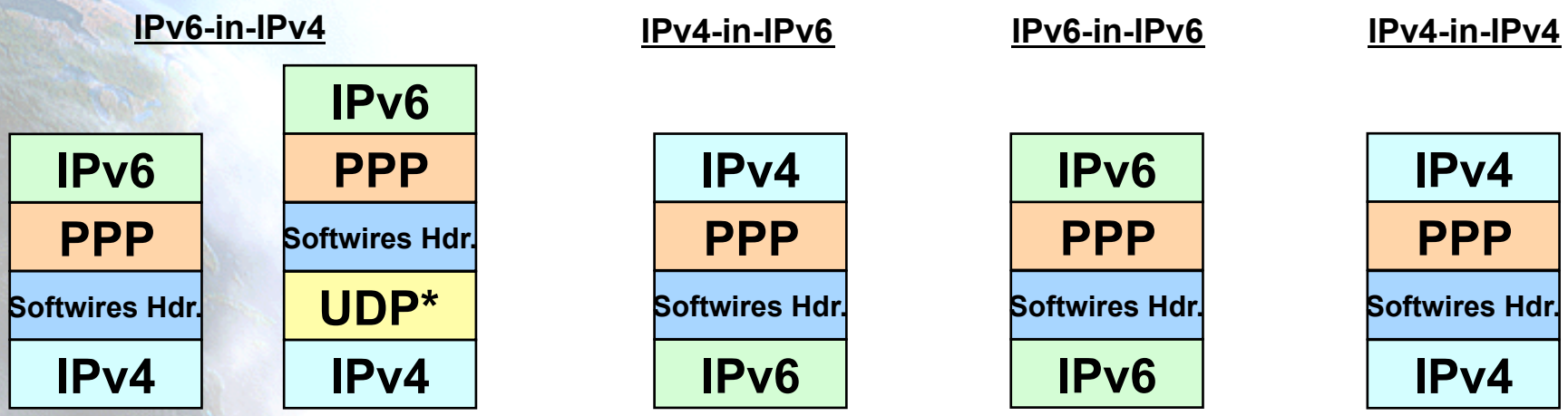
Mechanism based on translation

Softwires

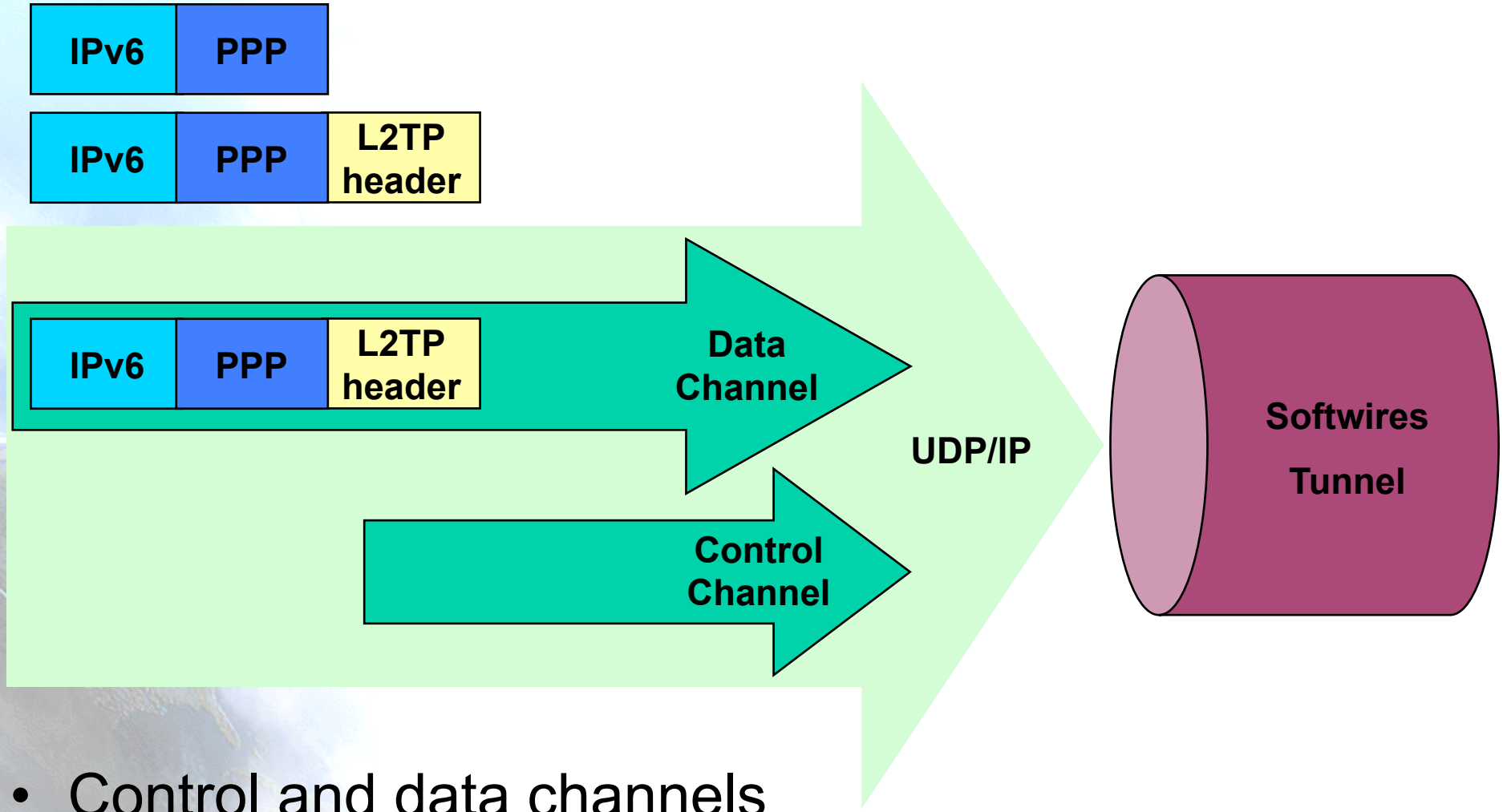
- Protocol introduced by the Softwires IETF WG
  - "Universal" transition mechanism based on the setup of tunnels
    - IPv6-in-IPv4, IPv6-in-IPv6, IPv4-in-IPv6, IPv4-in-IPv4
    - Allow NAT traversal
    - Prefix delegation in IPv6 (/48, /64, etc.)
    - User authentication for the tunnel setup interacting with AAA infrastructure
    - Choice of secure tunnels
    - Low overhead in the transport of IPv6 in the tunnels
    - Easy to code for mobile devices and/or devices with low hardware resources
  - Allows provisioning IPv6 connectivity in devices such as ADSL routers, cellular phones, PDAs, etc., when native IPv6 is not available in the access network
  - Also enables provisioning IPv4 connectivity in devices which only have native IPv6 connectivity
- In fact, it is not a new protocol, but a definition of how to use existing protocols in a different way in order to provide IPv6 connectivity in IPv4 networks and vice versa
- Based on:
  - L2TPv2 (RFC2661)
  - L2TPv3 (RFC3991)

Softwires L2TPv2 Encapsulation

- Specified in draft-ietf-softwire-hs-framework-l2tpv2
- Two entities:
  - Softwires Initiator (SI): Agent responsible for the tunnel request
  - Softwires Concentrator (SC): Agent responsible for the tunnel setup (tunnel end point)
- PPP is used to transport the IPvx (x=4, 6) packets in IPvy (y=4, 6) ones
  - PPP can be encapsulated in UDP in order to traverse NATs



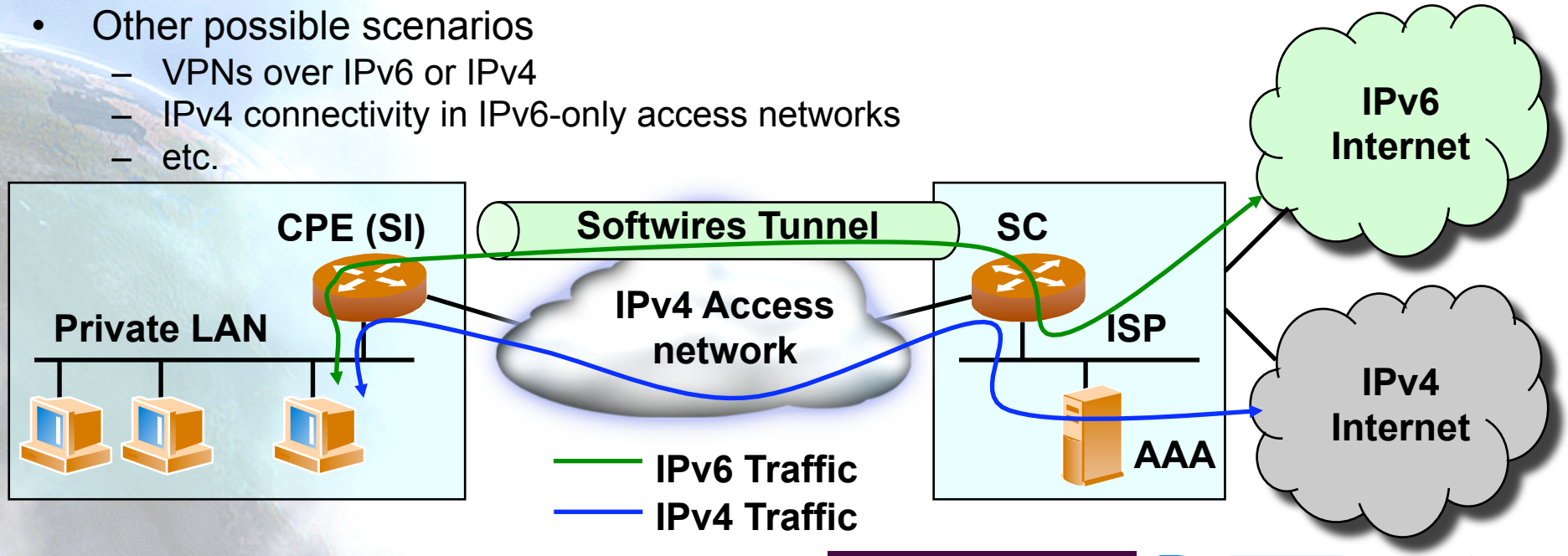
Softwires with L2TPv2



- Control and data channels
- PPP used as encapsulation protocol

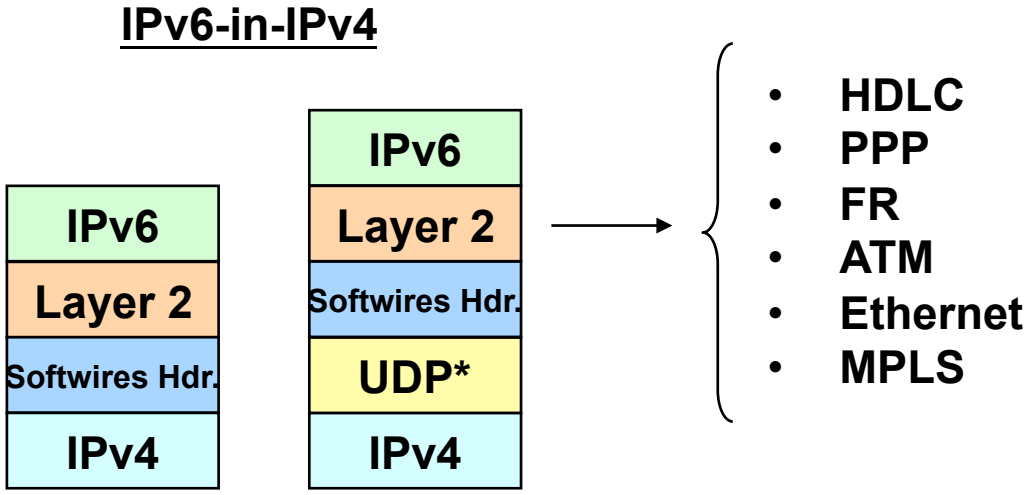
Softwires usage example

- A typical usage of softwires is the provision of IPv6 connectivity to residential users in IPv4-only access networks
  - The SC is deployed in the ISP network
    - DSLAM, aggregation router or other device
  - The SI is deployed in the user network
    - Typically the CPE. Also it may be another device in the user network, for example behind the NAT
  - The SC provides IPv6 connectivity to the SI, and the SI becomes the IPv6 router for the rest of the user network
  - IPv6 prefix delegation is used from the SC to the SI to provide a valid prefix (typically a /48) for the user network
    - DHCPv6 PD
- Other possible scenarios
  - VPNs over IPv6 or IPv4
  - IPv4 connectivity in IPv6-only access networks
  - etc.



Softwires L2TPv3 Encapsulation

- Same philosophy and components as with L2TPv2, but with L2TPv3 features
  - Transport over IP/UDP of other layer 2 protocols different than PPP
    - HDLC, PPP, FR, ATM, Ethernet, MPLS, IP
  - Header formats enhanced to allow a faster process in the SC
    - Allows speeds such as T1/E1, T3/E3, OC48
  - Minimum overhead in the encapsulated packets (only 4 o 12 extra bytes)
  - Other authentication mechanisms different than CHAP and PAP
    - EAP



*** Optional**

Configuration of Transition Mechanisms: Exercises

- E1: Setup a 6in4 tunnel between two alumni's hosts
- E2: Delete the 6in4 tunnel
- E3: Get IPv6 connectivity by means of a 6in4 tunnel by using a TB
  - See the path to different IPv6 web sites
  - See the path to the provided IPv6 address from a looking glass
- E4: Get IPv6 connectivity by means of a 6to4 tunnel
  - See the path to different IPv6 web sites
  - See the path to the provided IPv6 address from a looking glass
- E5: Setup a 6to4 relay (Windows 2003)
- E6: Setup a Teredo Client (Windows XP/2003)
- E7: Usage of IPv4/IPv6 proxies
  - 46Bouncer
  - Windows XP/2003

E1: 6in4 Tunnel Setup (1)

1. Exercise to be made with partners (*)
   - Alumni A ==> ADD_IPv4_A
   - Alumni B ==> ADD_IPv4_B
2. Alumni A sets up the tunnel in his side by using the following data:
   - Local IPv6 address ==> ADD_IPv4_A
   - Remote IPv4 address ==> ADD_IPv4_B
   - IPv6 address ==> 2001:10:20:30::12/126
   - IPv6 gateway address ==> 2001:10:20:30::11/126
3. Alumni B sets up the tunnel in his side by using the following data:
   - Local IPv4 address ==> ADD_IPv4_B
   - Remote IPv4 address ==> ADD_IPv4_A
   - IPv6 address ==> 2001:10:20:30::11/126
   - IPv6 gateway address ==> 2001:10:20:30::12/126
4. Check IPv6 connectivity between both alumni
   - Alumni A ==> ping6 IPv6_Address_Alumna_B
   - Alumni B ==> ping6 IPv6_Address_Alumna_A
5. Enable forwarding
   - Alumni A ==> enable forwarding in both tunnel and LAN interfaces
   - Alumni B ==> enable forwarding in both tunnel and LAN interfaces

- (*) This exercise does not provide global IPv6 connectivity, just IPv6 connectivity between alumni A and alumni B

E1: 6in4 Tunnel Setup (2)

- Scripts for setting up 6in4 tunnels
  - Windows XP/2003 (from the command line window)
    - netsh interface ipv6 add v6v4tunnel “Tunnel01" Address_IPv4_local Address_IPv4_remote
    - netsh interface ipv6 add address “Tunnel01" Address_IPv6
    - netsh interface ipv6 add route ::/0 “Tunnel01" Address_gateway_IPv6 publish=yes
    - netsh interface ipv6 set interface “Tunnel01” forwarding=enable
    - netsh interface ipv6 set interface “LAN” forwarding=enable
  - Linux/UNIX (from the shell)
    - modprobe ipv6
    - ip tunnel add Tunnel01 mode sit remote Address_IPv4_remote local Address_IPv4_local ttl 255
    - ip link set Tunnel01 up
    - ip addr add Address_IPv6/126 dev Tunnel01
    - ip route add 2000::/3 dev Tunnel01
  - FreeBSD
    - gifconfig gif0 Address_IPv4_local Address_IPv4_remote
    - ifconfig gif0 inet6 Address_IPv6 Address_gateway_IPv6 prefixlen 128
    - route -n add -inet6 default Address_gateway_IPv6

E1: 6in4 Tunnel Setup (3)

- Scripts for setting up 6in4 tunnels
  - FreeBSD >= 4.4
    - ifconfig gif0 create
    - ifconfig gif0 tunnel Address_IPv4_local Address_IPv4_remote
    - ifconfig gif0 inet6 Address_IPv6 Address_gateway_IPv6 prefixlen 128
    - route add -inet6 default Address_gateway_IPv6
  - NetBSD
    - ifconfig gif0 Address_IPv4_local Address_IPv4_remote
    - ifconfig gif0 inet6 Address_IPv6 Address_gateway_IPv6 prefixlen 128
    - route -n add -inet6 default Address_gateway_IPv6
  - OpenBSD
    - ifconfig gif0 giftunnel Address_IPv4_local Address_IPv4_remote
    - ifconfig gif0 inet6 Address_IPv6 Address_gateway_IPv6 prefixlen 128
    - route -n add -inet6 default Address_gateway_IPv6

E2: Deleting 6in4 tunnels (1)

- Exercise to be done by each alumni (individually)
- The alumni deletes the tunnel configured previously according to the configuration script of its Operating System
- The alumni has to check that the tunnel has been deleted by using:
  - ipconfig on Windows XP/2003
  - ifconfig on Unix/Linux/*BSD

E2: Deleting 6in4 Tunnels (2)

- Scripts for deleting 6in4 tunnels
  - Windows XP/2003 (from the command line window)
    - netsh interface ipv6 del route ::/0 “Tunnel01" Address_gateway_IPv6
    - netsh interface ipv6 del address “Tunnel01" Address_IPv6
    - netsh interface ipv6 del int “Tunnel01"
  - Linux/UNIX (from the shell)
    - ip route del 2000::/3 dev Tunnel01
    - ip addr del Address_IPv6/126 dev Tunnel01
    - ip link set Tunnel01 down
    - ip tunnel del Tunnel01 mode sit remote Address_IPv4_remote local Address_IPv4_local ttl 255
  - FreeBSD
    - route delete -inet6 default
    - ifconfig gif0 inet6 delete Address_IPv6
    - ifconfig gif0 down

E2: Deleting 6in4 Tunnels (3)

- Scripts for deleting 6in4 tunnels
  - FreeBSD >= 4.4
    - route delete -inet6 default Address_gateway_IPv6
    - ifconfig gif0 inet6 Address_IPv6 prefixlen 128 delete
    - ifconfig gif0 delete
  - NetBSD
    - route delete -inet6 default
    - ifconfig gif0 inet6 delete Address_IPv6
    - ifconfig gif0 down
  - OpenBSD
    - ifconfig gif0 inet6 delete Address_IPv6
    - ifconfig gif0 deletetunnel
    - ifconfig gif0 down
    - route delete -inet6 default

E3: IPv6 Connectivity via a TB

1. Choose a TB from http://www.ipv6tf.org/using/connectivity/test.php
2. Follow the steps provided by the TB
3. Check that the IPv6 connectivity is available
   - ping6, traceroute6 (ping & tracert on windows)
     - www.kame.net, www.6power.org, www.ipv6.org
   - Browsing to the same web sites
4. Check the path to the assigned IPv6 address from an external looking glass
   - http://www.ipv6tf.org/using/connectivity/looking_glass.php
   - http://www.ipv6.udg.mx/lg.php
   - http://www.v6.dren.net/lg/

E4: IPv6 Connectivity with 6to4 (1)

1. Choose a 6to4 relay from http://www.ipv6tf.org /using/connectivity/6to4.php
2. Follow the configuration script according to the proper Operating System
3. Check that the IPv6 connectivity is available
   - ping6, traceroute6 (ping & tracert en windows)
     - www.kame.net, www.6power.org, www.ipv6.org
   - Browsing to the same web sites
4. Check the path to the assigned IPv6 address from an external looking glass
   - http://www.ipv6tf.org/using/connectivity /looking_glass.php
   - http://www.ipv6.udg.mx/lg.php
   - http://www.v6.dren.net/lg/

E4: IPv6 Connectivity with 6to4 (2)

- Scripts for deleting the 6to4 tunnels
  - Windows XP/2003 (from the command line window)
    - netsh int ipv6 6to4 set relay Address_6TO4_RELAY enabled 1440
  - Linux/UNIX (from the shell)
    - ip tunnel add tun6to4 mode sit ttl 80 remote any local Address_public_IPv4_local
    - ip link set dev tun6to4 up
    - ip -6 addr add 2002:XXYY:ZZUU::1/16 dev tun6to4
    - ip -6 route add 2000::/3 via ::192.88.99.1 dev tun6to4 metric 1
    - Note that XXYY:ZZUU is the hexadecimal notation for Address_public_IPv4_local (the public IPv4 address) according to the following:
    - Address_public_IPv4_local = 60.172.21.22 -> 60 -> 3C
    - 172 -> AC
    - 21 -> 15
    - 222 -> DE
    - 60.172.21.22 -> XXYY:ZZUU = 3CAC:15DE

E4: IPv6 Connectivity with 6to4 (3)

- Scripts for deleting 6to4 tunnels
  - *BSD
    - Be sure that there is at least one stf(4) interface configured in the kernel
      - In http://www.netbsd.org/Documentation/kernel/ information about that can be found
    - ifconfig stf0 inet6 2002:XXYY:ZZUU::1 prefixlen 16 alias
    - route add -inet6 default 2002:c058:6301::1
    - Note that XXYY:ZZUU is the hexadecimal notation for Address_public_IPv4_local (the public IPv4 address) according to the following:
    - Address_public_IPv4_local = 60.172.21.22 -> 60 -> 3C
    - 172 -> AC
    - 21 -> 15
    - 222 -> DE
    - 60.172.21.22 -> XXYY:ZZUU = 3CAC:15DE

E5: Setting-Up a 6to4 Relay (Windows 2003)

- The 6to4 Relay configuration is very ease in case of Windows 2003
  - netsh interface ipv6 set interface interface="Local area connection" forwarding=enabled
  - netsh interface ipv6 set state state=enabled undoonstop=disabled
  - netsh interface ipv6 set relay name=192.88.99.1 state=enabled interval=1440
  - netsh interface ipv6 set routing routing=enabled sitelocals=enabled
- Every 6to4 packet received by the "Local area connection" interface will be forwarded to the proper IPv6 destination
- In order to check the 6to4 relay configuration, a 6to4 tunnel can be configured in other host (following the instructions of previous slides) and the 6to4 server in such a new host will be the 6to4 relay just configured
  - Doing ping6 and traceroute6 (ping and tracert on Windows XP/2003) to check IPv6 connectivity

E6: Setting-Up a Teredo Client (Windows XP/2003)

- There are other Teredo implementations for other Operating Systems such as:
  - Linux: http://www.simphalempin.com/dev/miredo/
  - FreeBSD: http://www-rp.lip6.fr/teredo/
- Windows XP/2003 presents an implementation of Teredo Client
- From a DOS window type the following:
  - set teredo client teredo.ipv6.microsoft.com. 60 34567
  - a public Teredo Server by Microsoft is used
    - teredo.ipv6.microsoft.com
- There exist other experimental Teredo Server/Relays (without guarantied service)
  - teredo.ipv6.vol.cz
  - teredo.ipv6.wind.com
  - teredo.via.ecp.fr
- Check the provided IPv6 address
  - ipconfig
- Check the data of the Teredo interface
  - netsh int ipv6 show teredo
  - netsh int ipv6 show int teredo
- Global IPv6 connectivity is not provided because Microsoft does not provide any Teredo Relay
- IPv6 connectivity with other Teredo clients is available
  - Check by pinging to the IPv6 address of other alumni's Teredo Client

E7: Use of IPv4/IPv6 Proxies (1)

- An IPv4/IPv6 proxy is not the same that a transition mechanism based on translation (NAT-PT)
- The proxy is an intermediate host working on the application level
  - It receives TCP connections over a protocol (IPv4 or IPv6) and it extracts all the data from the application level
  - Then it establishes TCP connection (IPv6 or IPv4) with the destination host and it put in the new connection the application data extracted in the previous step
- So, it allows connections between:
  - Client IPv4 ==> Proxy IPv4/IPv6 ==> Server IPv6
  - Client IPv6 ==> Proxy IPv6/IPv4 ==> Server IPv4
- There are two well-known proxies:
  - 46Bouncer (Windows y Linux)
  - Windows XP/2003

E7: Use of IPv4/IPv6 Proxies (2)

- Implement a IPv4/IPv6 Proxy on Windows XP/2003
  - Forward the TCP/ IPv4 8220 port to the TCP/IPv6 80 port of www.kame.net (2001:200:0:8002:203:47ff:fea5:3085)
  - netsh int port set v4tov6 Port_v4_TCP_local Address_IPv6_remote Port_v6_TCP_remote Address_IPv4_local
  - netsh int port set v4tov6 8220 2001:200:0:8002:203:47ff:fea5:3085 80 Address_IPv4_local
  - Check with http://address_IPv4_local
- Implement a IPv6/IPv4 Proxy on Windows XP/2003
  - Forward the TCP/IPv6 8330 port to the TCP/IPv4 80 port of www.kame.net (203.178.141.194)
  - netsh int port set v6tov4 8330 203.178.141.194 80 Address_IPv6_local

Part 4

Examples of Applications

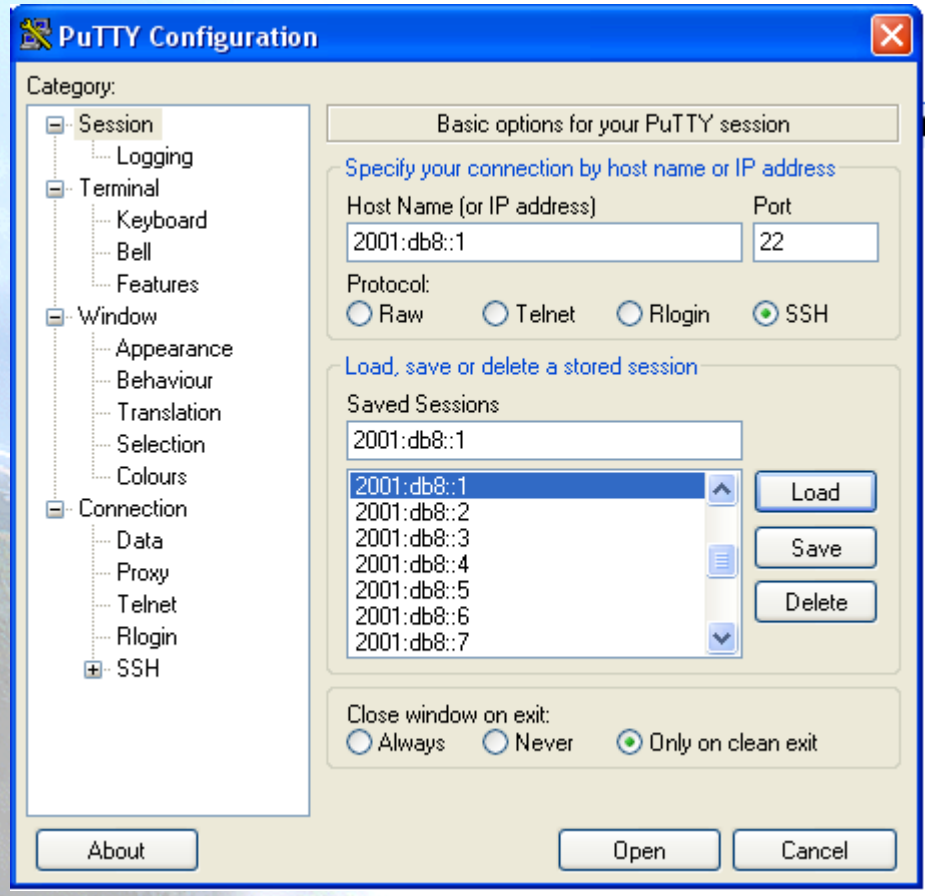
IPv6 Applications (1)

- Client-Server model implies that it is possible to have Client/Server applications working:
  - IPv4 Only
  - IPv6 Only
  - IPv4 + IPv6
- Thus provides a set of combinations that is needed to consider jointly with the availability or unavailability of IPv4/IPv6 connectivity

IPv6 Applications (2)

- **DNS lookups** are used to make or differentiate an available service through IPv4 and/or IPv6
- If a clients wants to connect to service.example.com, when resolving the domain name he/she can get an IPv4, IPv6 or both addresses
- In the case of getting both (v4 and v6) it is up to the client which protocol (v4/v6) to choose. The common practice is to choose v6 as the first option by default

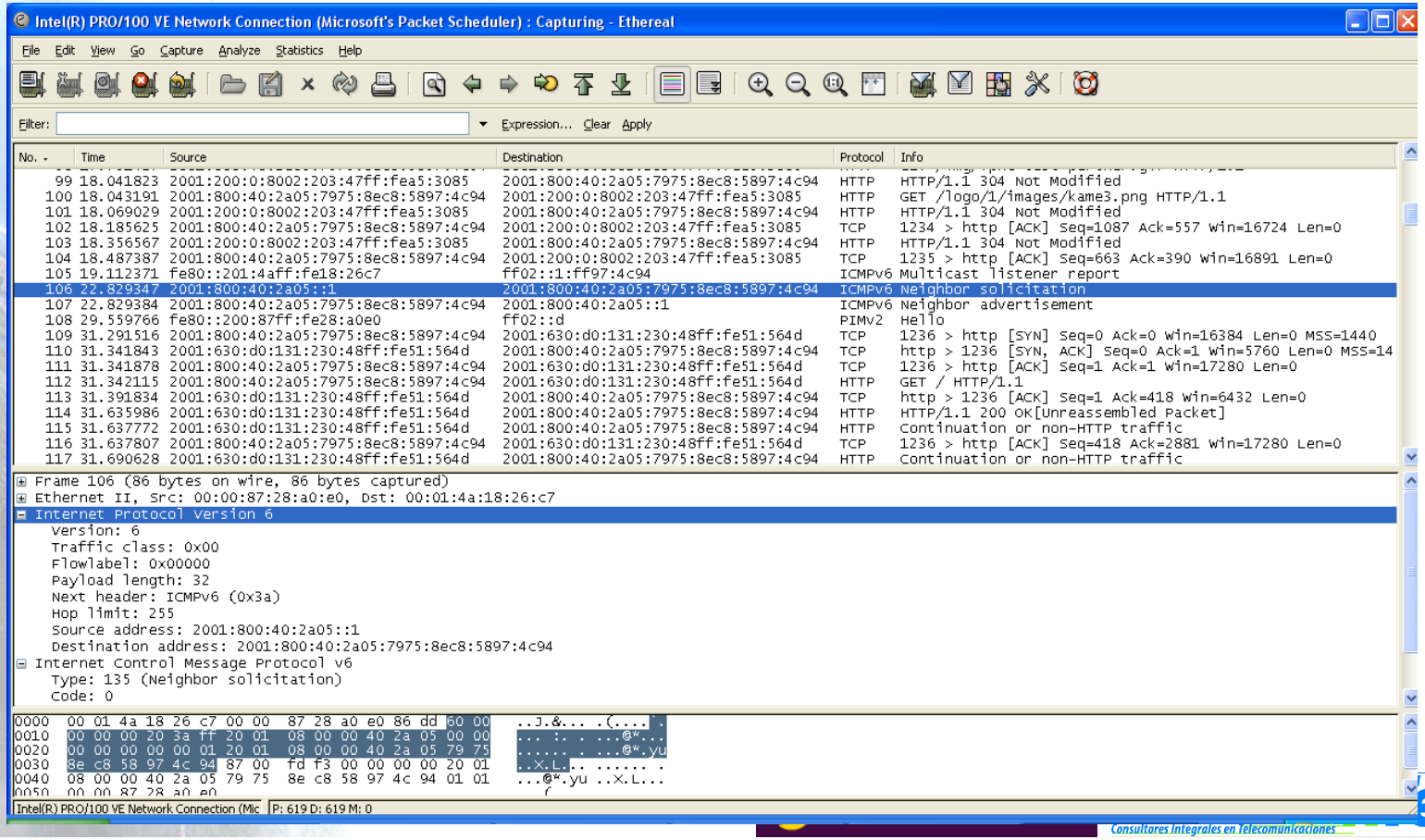
IPv6 Applications (3)



- **Putty**
- IPv4/IPv6 Client for Telnet and SSH
- Very useful for Administration and Management of devices
- Available at http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html

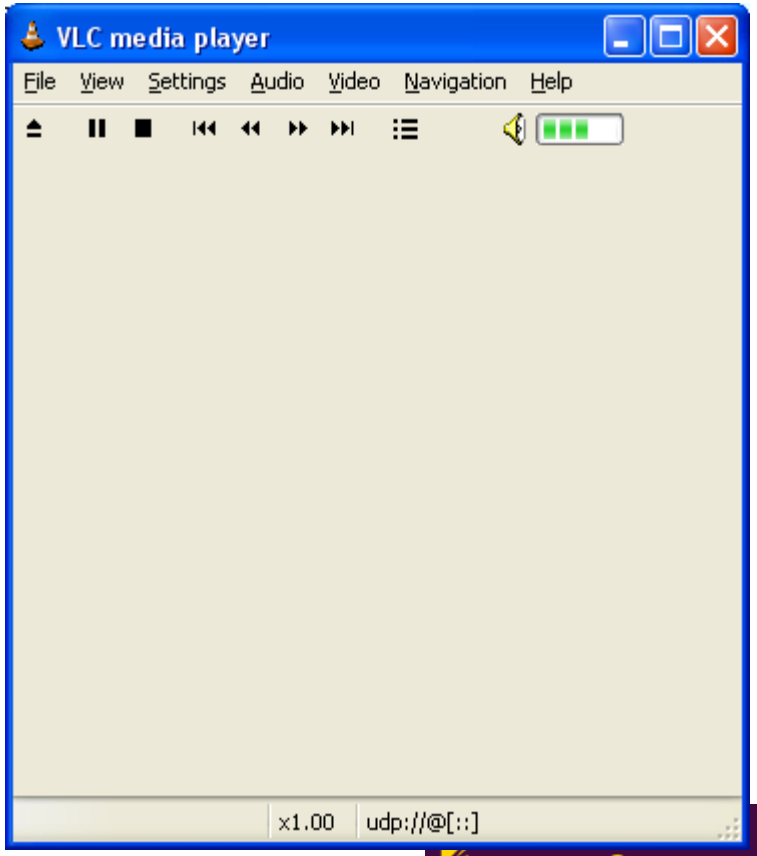
IPv6 Applications (4)

- **Ethereal**
- Captures y Decodes IPv4/IPv6 Traffic
- Very useful for connectivity validation and troubleshooting
- Available at http://www.ethereal.com/download.html



IPv6 Applications (5)

- **VLC**
- Multimedia Client and Server
- Unicast y Multicast Support
- Available at http://www.videolan.org/vlc/



IPv6 Applications (6)

- **VNC**
  - Remote Access to a PC using IPv6
  - Graphic Environment
- Client/server Model
  - Server installed in remote PC which is the target
  - Client installed in local PC for remote access
- Supported for
  - Windows XP
  - Linux
- Available at
  - http://jungla.dit.upm.es/~acosta/paginas/vncIPv6.html

IPv6 Applications (7)

- **Web**
- The most common Clients: Firefox, IE, Konqueror, Opera, Safari
- Servers: Apache 2, IIS

IPv6 Applications (8)

- **FreeBSD**
- You can use FreeBSD ports:

  #>cd /usr/ports

  #>make search key="ipv6"
- A list of available IPv6 applications with IPv6 support will appear. Among the information of each application you can find the *path*, which is the folder where we will go and from where we can install the application:

  #>cd path

  #>make install
- This starts a search over different source code servers, from where the application will be downloaded, compiled and installed
- You can also download just the source code, that will be in /usr/ports/distfiles, using instead of make install, make fetch

IPv6 Applications: Exercise 1 (1)

- **Windows**

  C:\>nslookup

  >set type=a

  >www.ipv6tf.org

  Name: www.ipv6tf.org

  Address: 213.172.48.141

  >set type=aaaa

  >www.ipv6tf.org

  www.ipv6tf.org AAAA IPv6 address =
  2001:800:40:2a03::3

IPv6 Applications: Exercise 1 (2)

- **Linux:**

  # dig a www.ipv6tf.org

  ;; QUESTION SECTION:

  ;www.ipv6tf.org. IN A

  ;; ANSWER SECTION:

  www.ipv6tf.org. 172800 IN A 213.172.48.141

- **# dig aaaa www.ipv6tf.org**

  ;; QUESTION SECTION:

  ;www.ipv6tf.org. IN AAAA

  ;; ANSWER SECTION:

  www.ipv6tf.org. 172800 IN AAAA 2001:800:40:2a03::3

IPv6 Applications: Exercise 1 (3)

- **Linux:**

```
#dig aaaa www.kame.net @2001:800:40:2a03::3
;; QUESTION SECTION:
;www.kame.net.          IN      AAAA
;; ANSWER SECTION:
www.kame.net. 86400 IN  AAAA
    2001:200:0:8002:203:47ff:fea5:3085
;; Query time: 400 msec
;; SERVER:
    2001:800:40:2a03::3#53(2001:800:40:2a03::3)
;; WHEN: Fri Jun 24 13:49:41 2005
;; MSG SIZE  rcvd: 107
```

IPv6 Applications: Exercise 2

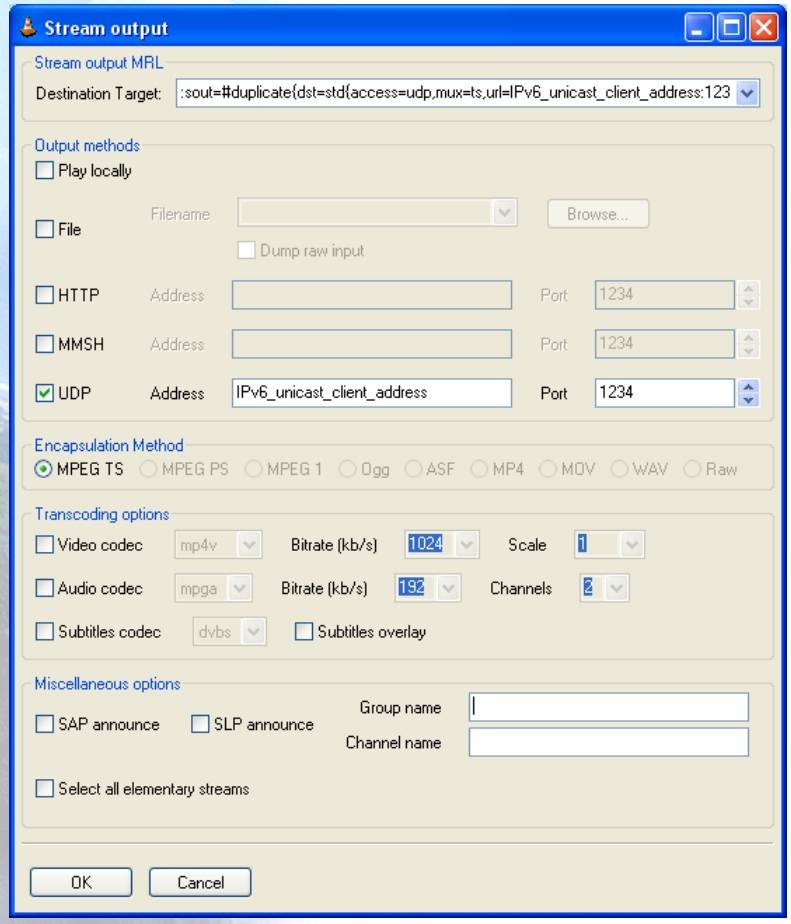
- To install (in case those are not already installed):
  - SSH Client with IPv6 support (Putty)
  - FTP Client (Command line on BSD, Linux, Windows)
  - Web Browser (Firefox, IE)
  - Ethereal
  - VLC
  - VNC

IPv6 Applications: Exercise 3

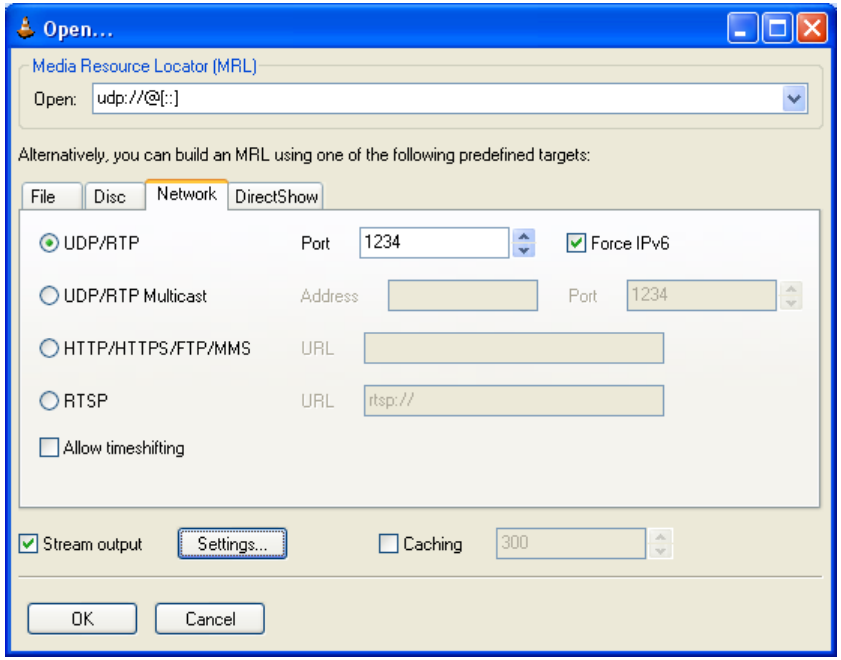
- To use the different services while Ethereal (or tcpdump) is used, in order to capture packets
- To use the SSH client to access by v4 or v6 choosing by means of DNS resolution
- To use the SSH client to access by v4 or v6 choosing by means of an application parameter (linux: #ssh -6|-4)(XP: ping -6|-4)

IPv6 Applications: Exercise 4 (1)

- VLC with Unicast



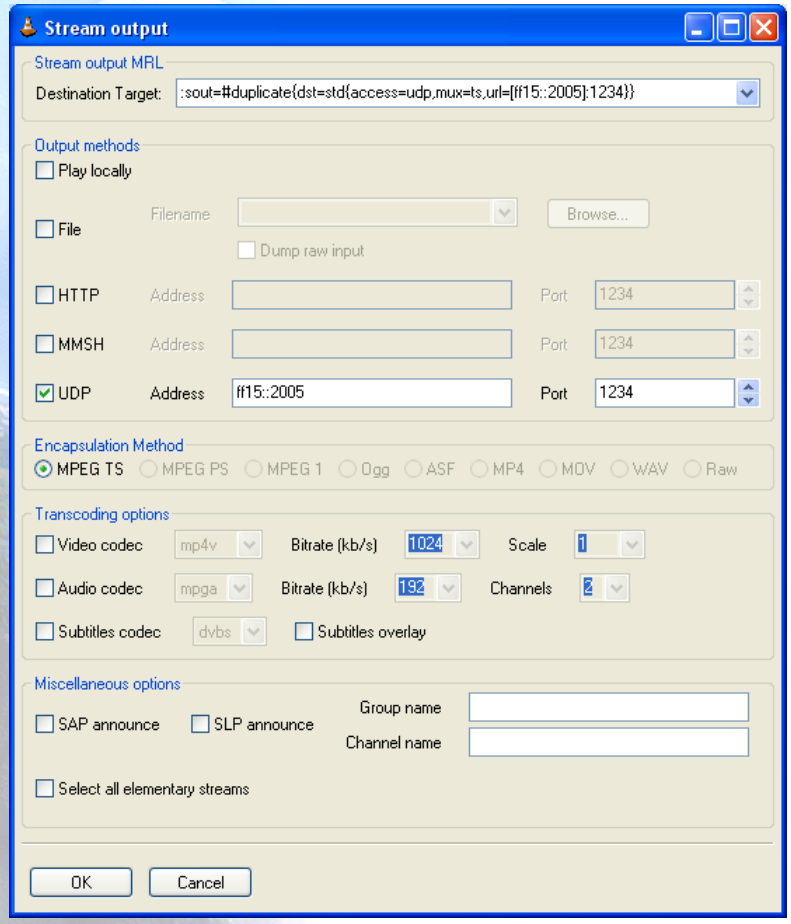
Server



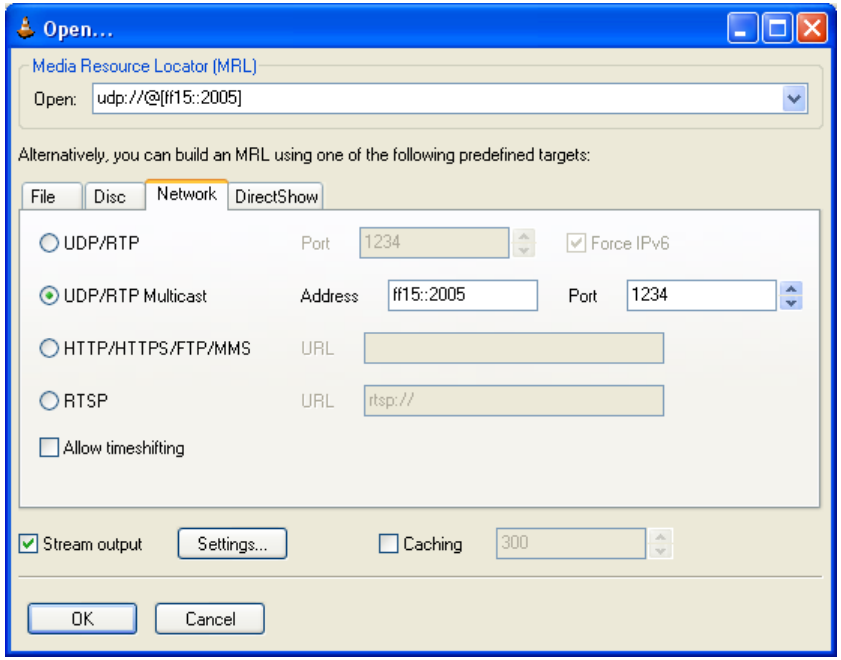
Client

IPv6 Applications: Exercise 4 (2)

- VLC with Multicast

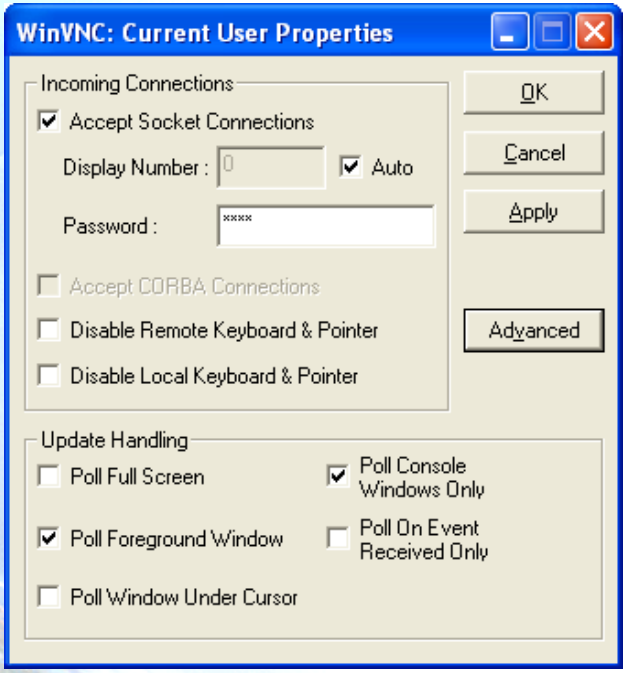


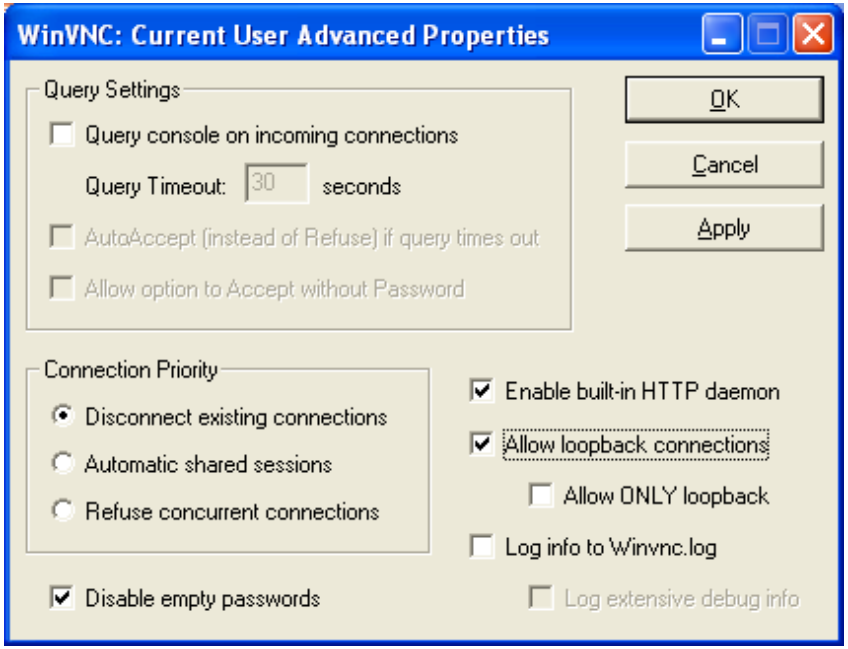
Server



Client

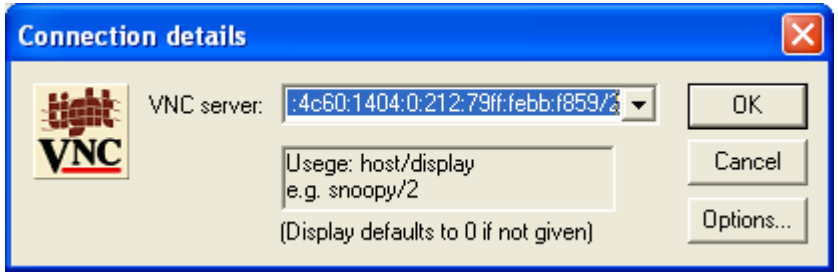
IPv6 Applications: Exercise 5 (1)





- **VNC Server Properties**
  - It is needed to configure the “Display Number” so as to receive the connections
    - Default value is 0
  - It is needed to define a password
- VNC Server Properties = => Advanced
  - Also enable “allow loopback connections”

IPv6 Applications: Exercise 5 (2)



- **VNC client**
  - VNC server is specified trough
    - An IPv6 address
    - Or a DNS name
  - Then, the “Display” is added after the VNC server
    - It is specified by a number separate from VNC server with a ‘/’

References (1)

- [6in4] RFC1933
- [TunAut] RFC1933
- [6to4] RFC3056
- [6over4] RFC2529
- [TB] RFC3053
- [TSP] draft-vg-ngtrans-tsp-01, http://www.hexago.com/index.php?pgID=step1
- [TEREDO] RFC4380
- [TEREDOC] http://www.microsoft.com/technet/prodtechnol/winxppro/maintain/teredo.mspx
- [ISATAP] draft-ietf-ngtrans-isatap-24
- [AYIYA] draft-massar-v6ops-ayiya-02
- [SILKROAD] draft-liumin-v6ops-silkroad-02
- [DSTM] draft-ietf-ngtrans-dstm-10
- [SIIT] RFC2765
- [NATPT] RFC2767
- [BIS] RFC2767
- [TRT] RFC3142
- [SOCKSv64] RFC3089

References (2)

- [PROTO41] draft-palet-v6ops-proto41-nat-04
- [STUN] RFC3489
- [NATPTIMPL]
  - http://www.ipv6.or.kr/english/download.htm ==> Linux 2.4.0
  - http://www.ispras.ru/~ipv6/index_en.html ==> Linux y FreeBSD
  - http://research.microsoft.com/msripv6/napt.htm Microsoft
  - ftp://ftp.kame.net/pub/kame/snap/kame-20020722-freebsd46 -snap.tgz ==> KAME snapshot (22.7.2002)
  - http://ultima.ipv6.bt.com/
- [STATELESS] RFC2462
- [STATEFULL] RFC3315
- [PRIVACY] RFC3041
- Windows IPv6
  - http://www.microsoft.com/resources/documentation/windows/xp/all /proddocs/en-us/sag_ip_v6_add_utils.mspx
  - http://www.microsoft.com/technet/community/columns/cableguy /cg0902.mspx.

Basic Configuration in Routers

Enable IPv6 (1)

- Cisco IOS
  - conf t interface if number ipv6 enable
  - ipv6 unicast-routing
- Juniper JUNOS
  - set interfaces if unit no family inet6 address addr
- Hitachi OS
  - by default in IPv6 images

Add an IPv6 Address

- Cisco IOS
  - config t interface type number
  - ipv6 address IPv6-address/prefix-length
  - ipv6 enable
- Juniper JUNOS
  - config interface type number
  - ipv6 address IPv6-address/prefix-length
- Hitachi OS
  - config ip interface-name IPv6-address/prefix -length

Delete an IPv6 Address

- Cisco IOS
  - config t interface type number
  - no ipv6 address IPv6-address
- Juniper JUNOS
  - config interface type number
  - no ipv6 address IPv6-address
- Hitachi OS
  - config delete ip interface-name IPv6-address

Router Advertisements

- Cisco IOS
  - After enable ipv6 unicast-routing and configure an IPv6 address on an interface, the router will begin to send advertisements and reply to solicitations. More control of router advertisements is available with the ipv6 nd commands, which control neighbor discovery in general
- Juniper JUNOS
  - config interface type number
  - ipv6 nd
  - This command is redundant over Ethernet, because router advertisements are automatically sent on Ethernet interfaces. However, unless explicitly enabled, IPv6 RA are not sent on other types of interfaces. More control with ipv6 nd commands
- Hitachi OS
  - config ra yes
  - ra interface interface-name
  - More control with options of ra commands

Cisco Routers

IPv6 Support

- The IPv6 features are supported in the following Cisco IOS Release trains:
  - 12.0S, 12.2T, 12.2S, 12.3, 12.3T, 12.4, and 12.4T
- Following tables identifies the earliest release for each IOS release train in which the feature became available. Unless noted otherwise subsequent releases of that Cisco IOS release train also support that feature
- Further information:
  - http://www.cisco.com/univercd/cc/td/doc/product/software/ios123/123cgcr/ipv6_c/ftipv6s.pdf

Cisco: Basic IPv6 Features

Feature	12.0S Release	12.2T Release	12.2S Release	12.2SB Release	12.3 Release	12.3T Release	12.4 Release	12.4T Release
IPv6	12.0(22)S	12.2(2)T	12.2(14)S	12.2(27) SBC	12.3	12.3(2)T	12.4	12.4(2)T
IPv6 address types: Unicast	12.0(22)S	12.2(2)T	12.2(14)S	12.2(27) SBC	12.3	12.3(2)T	12.4	12.4(2)T
IPv6: ICMPv6	12.0(22)S	12.2(2)T	12.2(14)S	12.2(27) SBC	12.3	12.3(2)T	12.4	12.4(2)T
IPv6: IPv6 neighbor discovery	12.0(22)S	12.2(2)T	12.2(14)S	12.2(27) SBC	12.3	12.3(2)T	12.4	12.4(2)T
IPv6: IPv6 stateless autoconfiguration	12.0(22)S	12.2(2)T	12.2(14)S	12.2(27) SBC	12.3	12.3(2)T	12.4	12.4(2)T
IPv6: IPv6 MTU path discovery	12.0(22)S	12.2(2)T	12.2(14)S	12.2(27) SBC	12.3	12.3(2)T	12.4	12.4(2)T
IPv6: ICMPv6 redirect	12.0(22)S	12.2(4)T	12.2(14)S	12.2(27) SBC	12.3	12.3(2)T	12.4	12.4(2)T
IPv6: neighbor discovery duplicate address detection	12.0(22)S	12.2(4)T	12.2(14)S	12.2(27) SBC	12.3	12.3(2)T	12.4	12.4(2)T
IPv6: IPv6 static cache entry for neighbor discovery	12.0(22)S	12.2(8)T	12.2(14)S	12.2(27) SBC	12.3	12.3(2)T	12.4	12.4(2)T
IPv6 address types: Anycast	—	—	12.2(25)S	12.2(27) SBC	—	12.3(4)T	12.4	12.4(2)T
IPv6: NetFlow for IPv6	—	—	—	—	—	12.3(7)T	12.4	12.4(2)T
IPv6: Mobile IPv6 home agent	—	—	—	—	—	12.3(14)T	12.4	12.4(2)T
IPv6: IPv6 default router preference	—	—	—	—	—	—	—	12.4(2)T
IPv6: IPv6 ACL extensions for Mobile IPv6	—	—	—	—	—	—	—	12.4(2)T
IPv6: HSRP for IPv6	—	—	—	—	—	—	—	12.4(4)T
IPv6: syslog over IPv6	—	—	—	—	—	—	—	12.4(4)T

Cisco: IPv6 Tunnels

Feature	12.0S Release	12.2T Release	12.2S Release	12.2SB Release	12.3 Release	12.3T Release	12.4 Release	12.4T Release
IPv6 tunneling: manually configured IPv6 over IPv4 tunnels	12.0(23)S	12.2(2)T	12.2(14)S	12.2(27) SBC	12.3	12.3(2)T	12.4	12.4(2)T
IPv6 tunneling: automatic 6to4 tunnels	12.0(22)S	12.2(2)T	12.2(14)S	12.2(27) SBC	12.3	12.3(2)T	12.4	12.4(2)T
IPv6 tunneling: automatic IPv4 -compatible tunnels	12.0(22)S	12.2(2)T	12.2(14)S	12.2(27) SBC	12.3	12.3(2)T	12.4	12.4(2)T
IPv6 tunneling: IPv6 over IPv4 GRE tunnels	12.0(22)S	12.2(4)T	12.2(14)S	12.2(27) SBC	12.3	12.3(2)T	12.4	12.4(2)T
IPv6 tunneling: IPv6 over UTI using a tunnel line card	12.0(23)S	—	—	—	—	—	—	—
IPv6 tunneling: ISATAP tunnel support	—	12.2(15)T	12.2(14)S	12.2(27) SBC	12.3	12.3(2)T	12.4	12.4(2)T
IPv6 tunneling: IPv4 over IPv6 tunnels	—	—	—	—	—	12.3(7)T	12.4	12.4(2)T
IPv6 tunneling: IPv6 over IPv6 tunnels	—	—	—	—	—	12.3(7)T	12.4	12.4(2)T
IPv6 tunneling: IP over IPv6 GRE tunnels	—	—	—	—	—	12.3(7)T	12.4	12.4(2)T
IPv6 tunneling: IPv6 GRE tunnels in CLNS networks	—	—	12.2(25)S	12.2(27) SBC	—	12.3(7)T	12.4	12.4(2)T

Cisco: IPv6 Routing

Feature	12.0S Release	12.2T Release	12.2S Release	12.2SB Release	12.3 Release	12.3T Release	12.4 Release	12.4T Release
IPv6 routing: RIP for IPv6 (RIPng)	12.0(22)S	12.2(2)T	12.2(14)S	12.2(27) SBC	12.3	12.3(2)T	12.4	12.4(2)T
IPv6 routing: static routing	12.0(22)S	12.2(2)T	12.2(14)S	12.2(27) SBC	12.3	12.3(2)T	12.4	12.4(2)T
IPv6 routing: route redistribution	12.0(22)S	12.2(2)T	12.2(14)S	12.2(27) SBC	12.3	12.3(2)T	12.4	12.4(2)T
IPv6 routing: multiprotocol BGP extensions for IPv6	12.0(22)S	12.2(2)T	12.2(14)S	12.2(27) SBC	12.3	12.3(2)T5	12.4	12.4(2)T
IPv6 routing: multiprotocol BGP link-local address peering	12.0(22)S	12.2(4)T	12.2(14)S	12.2(27) SBC	12.3	12.3(2)T	12.4	12.4(2)T
IPv6 routing: IS-IS support for IPv6	12.0(22)S	12.2(8)T	12.2(14)S	12.2(27) SBC	12.3	12.3(2)T	12.4	12.4(2)T
IPv6 routing: IS-IS Multitopology support for IPv6	12.0(26)S	12.2(15)T	12.2(18)S	12.2(27) SBC	12.3	12.3(2)T	12.4	12.4(2)T
IPv6 routing: OSPF for IPv6 (OSPFv3)	12.0(24)S	12.2(15)T	12.2(18)S	12.2(27) SBC	12.3	12.3(2)T	12.4	12.4(2)T
IPv6 routing: OSPF for IPv6 Authentication support with IPSec	—	—	—	—	—	12.3(4)T	12.4	12.4(2)T
IPv6 routing: IPv6 policy-based routing	—	—	—	—	—	12.3(7)T	12.4	12.4(2)T

Configure Manual IPv6 Tunnels

- Router(config)# interface tunnel tunnel-number
  - Example: Router(config)# interface tunnel 0
- Router(config-if)# ipv6 address ipv6-prefix/ prefix-length [eui-64] (Specifies the IPv6 network assigned to the interface and enables IPv6 processing on the interface)
  - Example: Router(config-if)# ipv6 address 2001:DB8:1:1::1/126
- Router(config-if)# tunnel source { ip-address | type number} (Source IPv4 address or the source interface type and number for the tunnel interface)
  - Example: Router(config-if)# tunnel source ethernet 0
- Router(config-if)# tunnel destination ip-address (Destination IPv4 address or host name for the tunnel interface)
  - Example: Router(config-if)# tunnel destination 192.168.30.1
- Router(config-if)# tunnel mode {aurp | cayman | dvmrp | eon | gre | gre multipoint | gre ipv6 | ipip [decapsulate-any] | iptalk | ipv6ip |mpls | nos}
  - Example1: Router(config-if)# tunnel mode ipv6ip (Túnel 6in4)
  - Example2: Router(config-if)# tunnel [mode gre ip] (Tunel IPv6 sobre IPv4 Generic Route Encapsulation (GRE)
- Router(config-if)# exit
- Router(config)# ipv6 route ipv6-prefix/ prefix-length tunnel tunnel-number (Configure a static default route via the tunnel)
  - Example: Router(config)# ipv6 route ::/0 tunnel 0

Configure 6to4 Tunnels

- Router(config)# interface tunnel tunnel-number
  - Example: Router(config)# interface tunnel 0
- Router(config-if)# ipv6 address ipv6-prefix/ prefix-length [eui-64] (Specifies the IPv6 address assigned to the interface and enables IPv6 processing on the interface. The 32 bits following the initial 2002::/16 prefix correspond to an IPv4 address assigned to the tunnel source)
  - Example: Router(config-if)# ipv6 address 2002:c0a8:6301:1::1/64
- Router(config-if)# tunnel source { ip-address | type number} (Source IPv4 address or the source interface type and number for the tunnel interface)
  - Example: Router(config-if)# tunnel source ethernet 0
- Router(config-if)# tunnel mode ipv6ip 6to4 (Túnel 6to4)
- Router(config-if)# exit
- Router(config)# ipv6 route ipv6-prefix/ prefix-length tunnel tunnel-number (Configures a static route for the IPv6 6to4 prefix 2002::/16 to the specified tunnel interface)
  - Example: Router(config)# ipv6 route 2002::/16 tunnel 0

Steps for Activate IPv6 BGP

- Strategy or plan of IPv6 BGP network
- Enable IPv6 unicast-routing
- Configure IPv6 BGP Routing Process
- Configure BGP Router ID
- Configure Peerings
- Adequate the configuration to the particular network

Configure one IPv6 BGP Process and the BGP Router ID

- Router(config)# router bgp autonomous-system-number (Configure the IPv6 BGP Process)
  - Example: Router(config)# router bgp 65000
- Router(config-router)# no bgp default ipv4-unicast (Disable the IPv4 unicast address family for the above BGP process)
  - Example: Router(config-router)# no bgp default ipv4-unicast
- Router(config-router)# bgp router-id ip-address (Optional: Configure a router ID of 32-bits. It is needed if there not exist IPv4 addresses in the router)
  - Example: Router(config-router)# bgp router-id 192.168.99.70

Configure one IPv6 BGP Peer

- Router(config)# router bgp autonomous-system-number
  - Example: Router(config)# router bgp 65000
- Router(config-router)# neighbor ipv6-address remote-as autonomous -system-number (by default only the IPv4 prefixes are announced)
  - Example: Router(config-router)# neighbour 2001:0DB8:0:CC00::1 remote-as 64600
- Router(config-router)# address-family ipv6 [unicast | multicast]
  - Example: Router(config-router)# address-family ipv6
- Router(config-router-af)# neighbor ipv6-address activate (This actives the announce of IPv6 prefixes in address-family ipv6)
  - Example: Router(config-router-af)# neighbour 2001:0DB8:0:CC00::1 activate

Configure one Route Map for IPv6 BGP

- Router(config)# router bgp 65000
- Router(config-router)# neighbour 2001:DB8:0:cc00::1 remote-as 64600
- Router(config-router)# address-family ipv6
- Router(config-router-af)# neighbour 2001:DB8:0:cc00::1 activate
- Router(config-router-af)# neighbor ipv6-address route-map map-name {in | out} (Apply one route map to inbound or ourbound routes)
  - Example: Router(config-router-af)# neighbour 2001:0DB8:0:cc00::1 route-map rtp in
- Router(config-router-af)# exit
- Router(config-router)# exit
- Router(config)# route-map map-name [permit | deny] [sequence-number]
  - Example: Router(config)# route-map rtp permit 10
- Router(config-route-map)# match ipv6 address prefix-list prefix-list-name
  - Example: Router(config-route-map)# match ipv6 address prefix-list cisco

Announce Routes into IPv6 BGP

- Router(config)# router bgp 65000
- Router(config-router)# address-family ipv6 unicast
- Router(config-router-af)# network ipv6-address/ prefix-length (Announces or injects the specified IPv6 prefix into IPv6 BGP. The route must be already in the unicast IPv6 routing table)
  - Example: Router(config-router-af)# network 2001:DB8::/32

Juniper Routers

IPv6 Support

– The IPv6 features are supported in the following JUNOS Releases:
  - 5.x, 6.x and 7.x

– Further information:
  - http://www.juniper.net/techpubs/software/erx/junose701/swconfig-routing-vol1/html/ipv6-config9.html

Configure Manual IPv6 Tunnels

- [edit]
- interfaces {
- gr-1/0/0 {
- unit 0 {
- tunnel {
- source local-ipv4-address;
- destination remote-ipv4-address;
- }
- family inet6 {
- address local-ipv6-address/126;
- }
- }
- }
- }

- Configure a static default route via the tunnel
  - **ipv6 route** 0::/0 tunnel-remote-ipv6-address

Hitachi Routers

IPv6 Support

– The IPv6 features are supported in the following Hitachi OS releases:
  - 06-xx, 07-xx, and 08-xx

– Further information:
  - http://www.hitachi.us/Apps/hitachicom/content.jsp?page=serviceandsupport/technicaldocumentation/details/GR2000%20Series.html&level=2&section=serviceandsupport&parent=technicaldocumentation&nav=left&path=jsp/hitachi/forbus/internetworking/&nId=iD

Configure Manual IPv6 Tunnels

- tunnel tunnel-name local-ipv4-address remote remote-ipv4-address
- ip tunnel-name local-ipv6-address/126 destination_ip_address remote-ipv6-address connect_type point

- Configure a static default route via the tunnel
  - Static 0::/0 gateway tunnel-remote-ipv6-address

Thanks !

Contact:

- Jordi Palet Martínez (Consulintel): jordi.palet@consulintel.es

The IPv6 Portal:

- http://www.ipv6tf.org