



| Title: | Document Version: |
|---|---|
| **IPv6 Tunnels through Routers with NAT** | 1.6 |

| Project Number: | Project Acronym: | Project Title: |
|---|---|---|
| IST-2001-32161 | Euro6IX | European IPv6 Internet Exchanges Backbone |

| Responsible and Editor/Author: | Organization: | Contributing WP: |
|---|---|---|
| Jordi Palet (jordi.palet@consulintel.es) | Consulintel | WP4 |

**Authors (organizations):**

César Olvera (cesar.olvera@consulintel.es, Consulintel)

David Fernández (david@dit.upm.es, UPM).

**Abstract:**

Some NAT boxes/routers allow the establishment of IPv6 tunnels from systems in the private LAN (using private IPv4 addresses) to routers or tunnel servers in the public Internet.

As far as we know this is not a common way of use IPv6 tunnels; the usual way is to finish the tunnel directly in a device with an IPv4 public address.

This behavior provides a big opportunity to rapidly deploy a huge number of IPv6 nodes and networks, w/o the need of new transition mechanism. So exploring this option is very important to facilitate the IPv6 deployment.

**Keywords:**

ADSL, IPv6, NAT, Router, Transition, Tunnel, Tunnel Broker.

# Table of Contents



# Table of Figures

# 1. INTRODUCTION

Some NAT boxes/routers allow the establishment of IPv6 tunnels from systems in the private LAN (using private IPv4 addresses) to routers or tunnel servers in the public Internet.

As far as we know this is not a common way of use IPv6 tunnels; the usual way is to finish the tunnel directly in a device with an IPv4 public address.

This behavior provides a big opportunity to rapidly deploy a huge number of IPv6 nodes and networks, w/o the need of new transition mechanism. So exploring this option is very important to facilitate the IPv6 deployment.

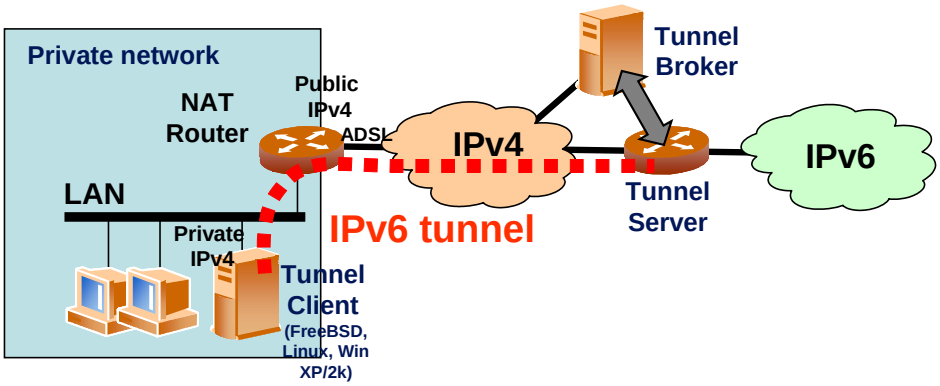


**Figure 1-1: IPv6 Tunnels through ADSL Routers with NAT**

We have tested the scenario shown in Figure 1-1, using a 3COM OfficeConnect 812 router (configured in routing mode with NAT) and we have successfully established IPv6 tunnels with tunnel servers from three well known Tunnel Broker implementations: BT, Freenet6 and TILAB, as well as with other routers in 6Bone, Consulintel, Euro6IX and UPM networks. Consulintel own tunnel broker implementation has been tested. Of course, this can be used also without a tunnel broker, with a manual configuration at the IPv6 router tunnel-end.

As described in detail in the next section, the router supports the establishment of IPv6 without any additional configuration. However, in some clients with certain operating systems, the tunnel configuration or the tunnel broker scripts have to be modified to reflect the private/public addressing conversion.

In the case of FreeBSD and Linux tunnel clients you just have to modify the configuration scripts received from the TB. As in FreeBSD and Linux scripts the client source address has to be configured, it is necessary to replace the public address configured in the script with the private address of that client.

In FreeBSD, for example, the script includes a line as follows:

*gifconfig gif0 [client IPv4] [server IPv4]*

You need to replace the client public address with the client private address.

In the case of Windows 2000/XP/.NET tunnel clients, as Windows scripts do not use the source address, no changes to the script are needed. The received script worked without modification in all the tests done:

*ipv6 rtu ::/0 2/::[server ipv4] pub* (for Windows 2000/XP)

*netsh interface ipv6 add route ::/0 interface=Automatic nexthop=::[server ipv4] publish=yes store=persistent* (for latest XP and Windows .NET)

Note: The "2" is the "interface index" corresponding to the "Tunnel Pseudo-Interface". May vary depending on your system setup. "Automatic" is the tunnel interface name, the "interface index" can also be used, but seems better to use the name, so it will work directly in different systems configurations.

As mentioned, detailed tests have been done with a 3Com OfficeConnect 812 router, although it should work with any other routers implementing basic NAT functionality (a few other NAT router models have been successfully tested, including ISDN versions). Besides, if there is a firewall between tunnel client and server, it must be configured to allow IPv6 over IPv4 tunnels to pass.

Other routers from several manufacturers are know to work, but as this had been tested in public scenarios (conferences), without access to the router, we don't know the exact router models/manufacturers and firmware versions. This worked for example at IST2002, Eurescom and some IEEE conferences, among others.

# 2. TECHNICAL DESCRIPTION

As described in RFC 2663, IP Network Address Translator (NAT) Terminology and Considerations:

*"Address translations performed by NAT are session based and would include translation of incoming as well as outgoing packets belonging to that session…*

*…a session is defined as the set of traffic that is managed as a unit for translation. TCP/UDP sessions are uniquely identified by the tuple of (source IP address, source TCP/UDP port, target IP address, target TCP/UDP port). ICMP query sessions are identified by the tuple of (source IP address, ICMP query ID, target IP address). All other sessions are characterized by the tuple of (source IP address, target IP address, IP protocol)."*

Basically, what the router does in this case is a NAT for protocol identifier 41 (the one used for IPv6 over IPv4 tunnels). The router considers each tuple of the form [source IP address, target IP address, IP protocol (41)] a different session. This fact can be seen issuing the "list nat vc" command on our example router:

```
3Com-DSL>list nat vc internet port

TCP connections:
Private Address    Port   Remote Address   Port   Public Address      Port     Timer
192.168.011.015    3313   138.004.XXX.XXX  1723   213.004.YYY.YYY     3313     14370

UDP connections:
Private Address  Port   Remote Address   Port   Public Address      Port       Timer
192.168.011.015  3004   193.152.063.197   53     213.004.YYY.YYY     3004       30

GRE connections:
Private Address    ID   Remote Address   ID       Public Address      ID     Timer
192.168.011.015     0   138.004.XXX.XXX   12632   213.004.YYY.YYY     0      14280

Other PID connections:
Private Address   PID Remote Address   Public Address      Timer
192.168.011.013    41  213.096.ZZZ.ZZZ   213.004.YYY.YYY     150
3Com-DSL>
```

**Figure 2-1: Dump of NAT Sessions**

As shown on the latest part, the router maintains a session for an IPv6 tunnel established from a system in the private network (192.168.11.13) to a router in the public Internet (213.096.ZZZ.ZZZ). Of course, the router that is the external endpoint of the tunnel (213.096.ZZZ.ZZZ) must be configured with the public address assigned to the NAT router (213.004.YYY.YYY).

One of the main inconveniences is that, as IPv6 tunnels are treated as any other NAT dynamic session, the tunnel entries are only added to the table whenever an IPv6 packet is sent from inside, but not with packets coming from the external tunnel endpoint (this is the basic behavior of an unidirectional NAT: It only allows outgoing sessions). By default, a 180 seconds inactivity timer is started when the entry is created, so the tunnel is deleted in no packets are sent for that time.

Although it is possible in the router tested to configure static entries on the conversion table for TCP and UDP sessions, it is not possible (at least not documented) to do that for protocol based sessions. So, the only way to maintain the session permanently is to constantly send traffic (for example, with a periodic ping from inside).

This fact is only a problem when IPv6 servers or services inside the private network are needed to be accessible from outside. If the traffic is client initiated, the session is created normally as soon as the first packet is sent, allowing native IPv6 communication. Note that, in this case, the restrictions to applications due to NAT traversing do not apply, because NAT is made to IPv4 packets that transport IPv6 ones, not to IPv6 packets. Besides, the protection derived from the unidirectional nature of NAT disappears for IPv6, so some security mechanism (network or personal firewalls) could be necessary to protect IPv6 systems in the private network.

The workaround is to activate a reverse NAT mechanism, sometimes called NATP, or PAT (as is the case for this router), so all the incoming traffic (not statically forwarded to other private IPs by the rest of the router configuration) goes to a specific node behind the NAT.

In our case, we configured some TCP and UDP ports to be forwarded to some nodes in the internal network (HTTP, SMTP and DNS servers), but the rest to be forwarded to the one that is being used as the “IPv6 router”.

See section 4 for specific configuration scripts for this “IPv6 router”. In this case, is heavily recommended that this “IPv6 router” have security mechanism to protect itself and the rest of the network for external attacks. Latest versions of some operating systems already include some kind of personal IPv6 filter-set or firewall.

Finally, as the tunnel session is externally identified by the public IP address, the IP address of the remote tunnel endpoint and the protocol ID (41), only one tunnel is allowed from systems in the private network to each external router. We also expect no need for several tunnels, specially considering the option of the “IPv6 router”.

Alternatively, being A and B systems inside the private networks and R1 and R2 routers in the public Internet, the following scenarios are possible:

- One tunnel between A and R1 and another between B and R2.
- One tunnel between A and R1 and another between A and R2.

What it’s not possible is to have two simultaneous tunnels from A and B to R1, because the NAT can’t distinguish between incoming packets belonging to each tunnel (they all have the same IPv4 source and destination addresses).

# 3. SUMMARY AND CONCLUSIONS

In summary, the basic NAT mechanism for translation of non-TCP/UDP sessions allows the easy establishment of IPv6 over IPv4 tunnels from a private network through a NAT, as tested with the 3COM 812 router.

With small and easy modifications, the drawbacks mentioned –the lack of static entries in the NAT conversion table and the lack of support in tunnel brokers for NAT scenarios– could be eliminated, allowing easy access to IPv6 services to residential and SOHO users connected through NAT routers and ISPs non supporting IPv6.

The modification suggested to the tunnel brokers could be as simple as including in the generated scripts the private IP address, so the user doesn't need to modify the script.

The support for Proto-41 forwarding to specific nodes behind the NAT, will also allow the bidirectional communication without the need to forward by default all the (non-statically forwarded) traffic to a given node.

Tests with others routers and operating systems are underway. If you have made similar tests we would like to hear about it.

Assuming that hundreds of routers already support this Proto-41 forwarding, the IPv6 deployment can be facilitated.

Alternatively, we suggest to router manufacturers, through an I-D under preparation, to incorporate this functionality for those routers or NAT boxes that support the firmware upgrade.

# 4. CONFIGURATION SCRIPTS FOR AN INTERNAL "IPV6 ROUTER" (BEHIND THE NAT)

## 4.1 Windows 2000 and earlier XP:

*Rem Tunnel from aaa.aaa.aaa.aaa (public tunnel broker IPv4) to bbb.bbb.bbb.bbb (NAT public IPv4)*
*Rem Tunnel Setup on the NAT box side*
*ipv6 rtu ::/0 2/::aaa.aaa.aaa.aaa pub*

*Rem Client "IPv6 Router" address setup*
*ipv6 adu 2/xxxx:xxxx:xxxx:xxxx::xxxx*

*Rem forward on ALL the interfaces (is needed or the RA will not be forwarded, seems to be a bug)*
*ipv6 ifc 1 forw*
*ipv6 ifc 2 forw*
*ipv6 ifc 3 forw*
*ipv6 ifc 4 forw*
*…*
*ipv6 ifc n forw*

*Rem enable forward and RA on the "router interface"*
*ipv6 ifc 3 forw adv*

*Rem and assign the prefix to advertise*
*ipv6 rtu yyyy:yyyy:yyyy:yyyy::/64 3 pub*

Note: The "2" is the "interface index" corresponding to the "Tunnel Pseudo-Interface". The "3" corresponds to the LAN interface. May vary depending on your system setup. In this case we are using the same LAN card for both sides of the "Windows router" (the one pointing to the NAT, and the one to the rest of the local network). This script need to be reloaded every time the system is restarted, but there is a procedure to convert a bat file into a service so its automatically started each time the system restarts.

## 4.2 Latest XP and .NET (Windows 2003):

*Rem Tunnel from aaa.aaa.aaa.aaa (public tunnel broker IPv4) to bbb.bbb.bbb.bbb (NAT public IPv4)*
*Rem Tunnel Setup on the NAT box side*
*netsh interface ipv6 add route ::/0 interface=Automatic nexthop=::aaa.aaa.aaa.aaa publish=yes store=persistent*

*Rem Client "IPv6 Router" address setup*
*netsh interface ipv6 add address interface=Automatic address=xxxx:xxxx:xxxx:xxxx::xxxx store=persistent*

*Rem forward on the tunnel interface (is needed or the RA will not be forwarded)*
*netsh interface ipv6 set interface interface=Automatic forwarding=enabled store=persistent*

*Rem enable forward and RA on the "router interface"*
*netsh interface ipv6 set interface interface=Public forwarding=enabled advertise=enabled store=persistent*

*Rem and assign the prefix to advertise*
*netsh interface ipv6 add route prefix=yyyy:yyyy:yyyy:yyyy::/64 interface=Private publish=yes store=persistent*

Note: "Automatic" is the interface name for the tunnel, the "interface index" can also be used, but seems better to use the name, so it will work directly in different systems configurations. Same for "Public" and "Private", referring to the interface to the NAT, and to the internal network interface respectively. If the same LAN card is used for the public and private interfaces, use "Public" in both cases. In this case, the scrip is stored in the OS configuration, so isn't needed again in a restart (see "netsh interface ipv6 dump").