Mecanismos de Transición hacia redes IPv6

Taffernaberry, Juan Carlos

Director: Ing. Marrone, Luis

Trabajo Final presentado para obtener el grado de Especialista en Redes y Seguridad

Facultad de Informática – Universidad Nacional de la Plata

Junio 2011

Índice de contenido

1 Descripción general del Trabajo

1.1 Introducción:

Debido a que el Protocolo de red de Internet actual, llamado IPv4, está alcanzando actualmente sus propios límites de diseño y se muestra incapaz de proveer una respuesta adecuada a las nuevas características deseables para Internet, en 1995 la Internet Engineering Task Force (IETF) comenzó a desarrollar un nuevo protocolo, llamado IPv6, para reemplazar al anterior. Contempla mejoras fundamentalmente en el espacio de direccionamiento y nuevas características como servicios de tiempo real, calidad de servicio, seguridad intrínseca, etc.

El crecimiento de Internet ha originado que cada vez más computadoras necesiten conectarse a ella. Hay una enorme cantidad de dispositivos como teléfonos celulares, cámaras de vigilancia, dispositivos inalámbricos, etc, que necesitarán, en el mediano plazo, sus propias direcciones IP para conectarse a Internet, incluso algunos necesitarán varias direcciones. Ésta es la principal causa que lo está llevando a sus límites de diseño, pues en la versión actual del protocolo, no existen suficientes direcciones disponibles.[1]

El protocolo IPv6 presenta un nuevo desafío que es su despliegue para ponerlo en producción. En la actualidad millones de computadores están interconectados al backbone de Internet usando IPv4 y es imposible cambiar a la nueva versión, IPv6, en forma simultánea cada uno de ellos para que sigan trabajando normalmente, fundamentalmente por la imposibilidad de actualizar a IPv6 sistemas operativos de routers intermedios, servidores web (HTTP), o de correo (SMTP), etc sin soporte IPv6; también se presentan problemas en servidores de nombre (DNS) sin registros AAAA o A6 para direcciones IPv6, etc.

El protocolo IPv6 es un protocolo “disruptivo”. El término disruptivo tiene sus orígenes en el libro “El dilema de Innovador” de Clayton Christensen, donde trata como los desarrollos tecnológicos pueden tener un impacto económico. Se basa en un estudio de la industria de Discos Rígidos, a través de varios años y varios cambios de tecnologías. Para nuestro caso, no se trata de quitar o deshabilitar IPv4 para usar, habilitar o instalar IPv6. Tampoco es una una migración, pues no es un día, mes o año (como el Y2K) para realizar la migración. Esto es una actualización necesaria de IP, permitiendo que ambas versiones convivan al mismo tiempo y/o independientemente.

Por tal motivo la IETF ha definido una serie de mecanismos para hacer una suave transición [2] donde convivan por un largo tiempo ambos protocolos. El presente trabajo ayudará al lector a lograr una transición controlada hacia el nuevo protocolo.

El objetivo del presente trabajo fue realizar un Análisis, Evaluación y Comparación de Métodos de Transición del protocolo IPv4 al protocolo IPv6 . Las comparaciones se hicieron usando un Test Bed llamado CODAREC6 [3], permitiendo colaborar en el lento, pero inexorable camino hacia la internet sobre IPv6.

1.2 Objetivos y Metodología de trabajo

1.2.1 Objetivos Generales

Analizar, evaluar y comparar los distintos Métodos de Transición del protocolo IPv4 al protocolo IPv6.

1.2.2 Objetivos Particulares:

La implementación de IPv6 en una red Lan o una red de Campus, puede ser una tarea difícil y compleja, pero si es planeada para hacerse en etapas y de manera controlada, se puede llegar a un buen destino. Para las redes hay muchos requerimientos diferentes dependiendo de cada caso en particular, lo que hacen necesario utilizar distintos mecanismos de transición, como por ejemplo si la red hace uso de Wireless, movilidad, o tecnologías de dial-up, etc. Como primer objetivo se determinaron los distintos escenarios que se pueden presentar en la transición de IPv4 a IPv6.

Como segundo objetivo se recrearon los escenarios determinados anteriormente sobre el Test Bed.

Una vez recreados los distintos escenarios, se estudiaron los métodos de transición más utilizados.

Se implementaron los métodos de transición estudiados, determinados en el objetivo anterior, sobre cada uno de los escenarios seleccionados en el primer objetivo.

Se compararon las soluciones obtenidas para la transición de cada uno de los escenarios previamente encontrados, y se determinó cual de los métodos es el más adecuado o correcto para cada caso.

1.3 Metodología

El desafío de las instituciones de investigación, desarrollo y educación frente a las nuevas tecnologías se basa en la necesidad primaria del conocimiento de la tecnología para luego el desarrollo de productos y la capacitación del medio circundante.

La tecnología IPv6 además produce, como consecuencia de su implementación, efectos en todo Internet, es decir a todas las redes conectadas mundialmente.

Una de las tareas más importantes es la referida a la transición del viejo protocolo IPv4 al nuevo y que afecta especialmente a las organizaciones debido al cambio interno de clientes, servicios y servidores, además de aprovechar las novedosas características del nuevo protocolo.

En este sentido, del abordaje de las nuevas tecnologías, este trabajo pretendió ser un ambiente de trabajo y desarrollo que permitió:
La DETERMINACIÓN de los distintos escenarios que pueden presentarse en la transición de IPv4 a IPv6, el MONTAJE de los distintos escenarios detectados, el ESTUDIO y EVALUACIÓN de las metodologías de transición para organizaciones de distinta envergadura, la IMPLEMENTACIÓN

de los distintos métodos de transición sobre cada uno de los escenarios determinados, la COMPARACIÓN de las soluciones obtenidas para la transición, y la DETERMINACIÓN de cual de las metodologías es la más adecuada o correcta para cada uno de los escenarios implementados.

Para cumplir con esta metas, como se nombró anteriormente, fue utilizado el IPv6 test bed, como instrumento de simulación y prueba de distintas tecnologías de transición hacia IPv6. A continuación es detallada la metodología aplicada para llevar a cabo los objetivos nombrados.

1.3.1 Determinación distintos Escenarios:

Se realizó el relevamiento de distintas redes, como la red de campus de la FRM, redes de área local y amplia de dependencias de Gobierno, y algunas redes de área local en empresas privadas.
De lo relevado, se analizó el tipo de acceso a Internet que cada lugar posee, aplicaciones que utiliza, tipo de ruteadores, servicios utilizados, etc.

Con lo analizado fueron determinados distintos escenarios típicos, para los que posteriormente se ensayaron las metodologías de transición.

1.3.2 Montaje escenarios usando un Test Bed:

Partiendo de la topología ofrecida por el Test Bed, fue realizada una adecuación de infraestructura, para reflejar cada uno de los distintos escenarios encontrados en el punto anterior.

En algunos casos fue necesario crear máquinas y redes virtuales para completar cada uno de los escenarios a ensayar.

Una vez concluido el montaje de cada uno de los escenarios, se realizó una verificación del funcionamiento de los mismos.

1.3.3 Estudio métodos de transición:

Se estudió el método de transición Dual Stack y los requerimientos para realizarlo.

Fueron estudiados los métodos de Túneles, como Túneles configurados, Túnel Broker, Túnel automático, Túnel 6to4, Isatap, Teredo, etc.

Se estudiaron los métodos de Traslación como el BIS, BIA, Transport Relay, SOCKS, Aplication Layer Gateway, etc.

Fue determinado el ambiente de aplicación y alcance para cada método estudiado.

1.3.4 Implementación de distintos métodos:

Sobre cada escenario fueron configurados, uno por uno, los distintos métodos de transición determinados en el punto anterior.

Se ejecutó un set de programas de prueba, fundamentalmente evaluando el desempeño de ruteo para los protocolos IPv4 e IPv6.

1.3.5 Comparación distintas metodologías:

Fue realizada una comparación de performance de cada técnica de transición aplicada a cada escenario en particular, obteniendo gráficos comparativos.

Se evaluó la flexibilidad de cada técnica de transición aplicada a cada escenario en particular, en lo que respecta a escalabilidad de la solución.

Finalmente, fue comparado el costo y mantenimiento de cada técnica en cada escenario en particular, teniendo en cuenta posibles cambios de proveedor, rango de direcciones, etc.

1.3.6 Difusión y Capacitación:

El IETF a través de su grupo de operaciones, v6ops, recomienda realizar una constante tarea de capacitación de recursos humanos. Por lo tanto un objetivo secundario del presente trabajo fue la capacitación y difusión de la temática de IPv6. Se realizaron las siguientes actividades durante todo el periodo en el cual se desarrolló la presente monografía:

- Dictado de clases teóricas y prácticas para la Cátedra Teleinformática del Departamento de Electrónica referidas a Protocolo IPv6 y Mecanismos de Transición- Abril-Mayo 2008.
- Autor del artículo “Codarec6: Transición, Implementación de Tunnel Broker” en el “XIV Congreso Argentino de Ciencia de la Computación”. ISBN 978-987-24611-0-2. Agosto 2008.
- Expositor en “XIV Congreso Argentino de Ciencia de la Computación. Chilecito” de “Codarec6: Transición, Implementación de Tunnel Broker” – La Rioja. Octubre 2008.
- Autor del artículo “Codarec6 Transición: Implementación de Tunnel 6to4” en el “IV Encuentro de Investigadores y Docentes de Ingeniería EnIDI”. ISBN 978-95042-0104-5. Setiembre 2008.
- Expositor en el “IV Encuentro de Investigadores y Docentes de Ingeniería EnIDI” de “Codarec6 Transición: Implementación de Tunnel 6to4”. Noviembre 2008.
- Expositor en en “Workshop de Actualización den telecomunicaciones” de “UTN Mendoza IPv6: Diseño e implementación de una Intranet IPv6” - Universidad Nacional de San Juan . Facultad de Ciencias Exactas y Naturales- Departamento de Informática. San Juan . Noviembre 2008.
- Dictado de clases teóricas y prácticas para la Cátedra Teleinformática del Departamento de Electrónica referidas a Protocolo IPv6 y Mecanismos de Transición. Abril-Mayo 2009.
- Autor de artículo “Prox6: Implementación de un Application Layer Gateway (ALG) para transición hacia redes IPv6” en el “V Encuentro de Investigadores y Docentes de Ingeniería EnIDI”. ISBN 978-950-42-0121-2. Noviembre 2009.

2 Descripción de Escenarios Detallados

2.1 Escenario A

Red de campus de la UTN Facultad Regional Mendoza

La topología LAN de la UTN-FRM es una estrella extendida según se detalla en el gráfico 1. El núcleo es un switch capa 3 3Com modelo SS3 4800. Este equipo se conecta mediante fibra óptica a switchs de 2do nivel (3Com 4500) y a la granja de servidores. Finalmente, los switchs de borde que proveen acceso a la red a las estaciones de trabajo, son 3Com SSII 3300/1100 y SSIII 4500. Existen 2 accesos a internet, uno a través de la RUT y otro a través del ISP Global Crossing.

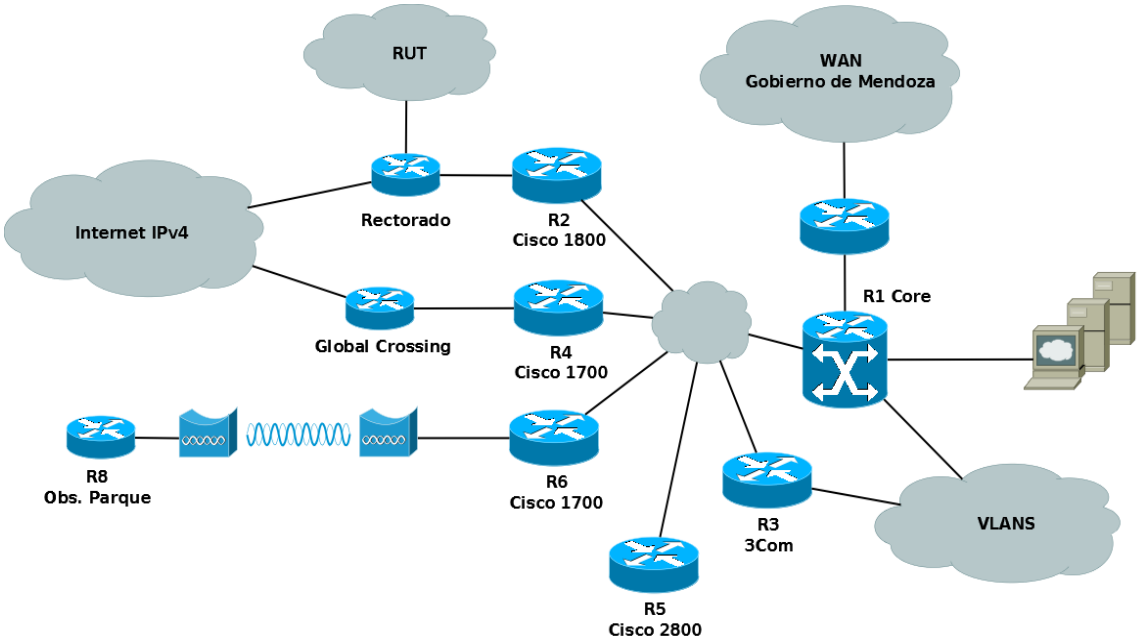

Gráfico 1 – Esquema de la red de la FRM-UTN

2.2 Escenario B:

Instituto Nacional de Vitivinicultura

Esta entidad, posee en Sede Central dos accesos a internet a través de ISPs solo IPv4 como se indica en el Gráfico 2. Las delegaciones se conectan mediante OpenVPN a Sede Central, utilizando para ello conexiones de banda ancha ADSL a través de proveedores de internet solo IPv4. Los Routers son basados en GNU/Linux. El switch de núcleo es capa 3, modelo Cisco Catalyst 3560. Los hosts de Sede Central y sucursales que dan servicios están basados en sistemas operativos Unix y GNU/Linux. El resto de los hosts ejecutan principalmente sistemas operativos Microsoft.

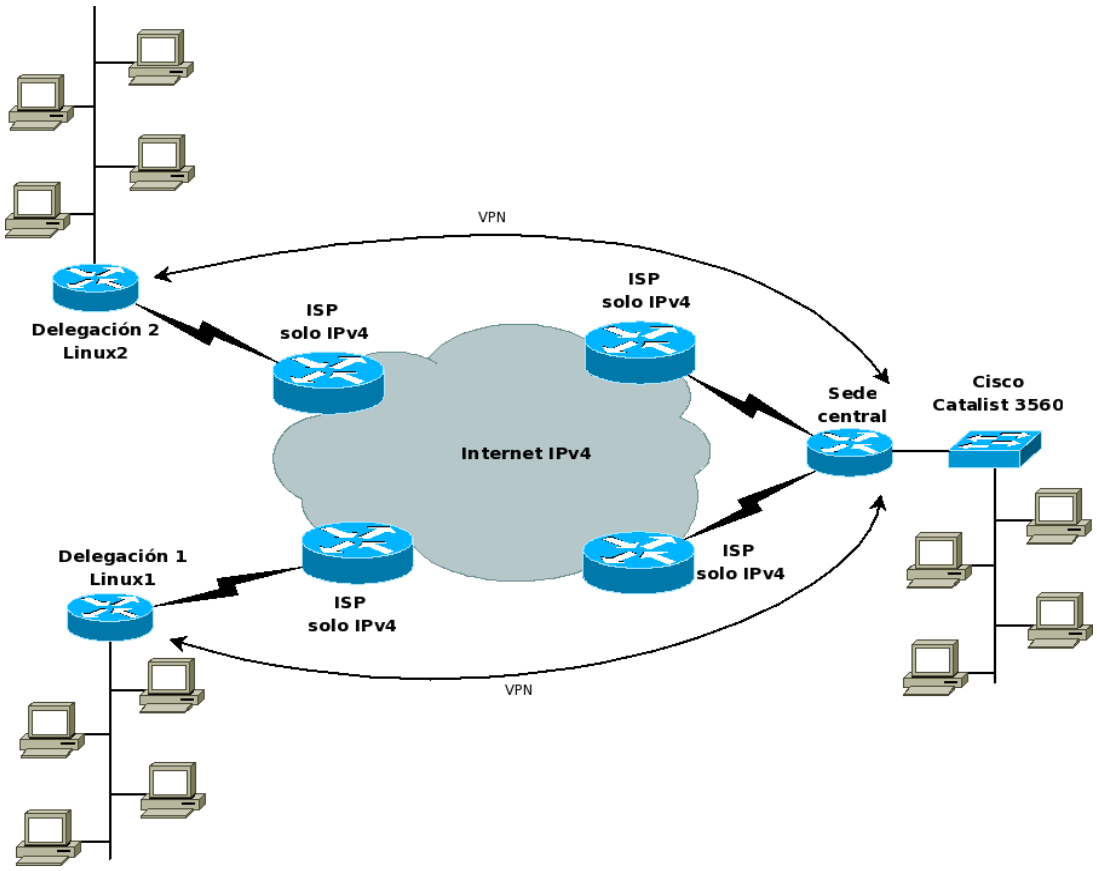

Gráfico 2: Esquema de enlaces de INV

2.3 Escenario C:

Enlace de banda ancha hogareño

En el Gráfico 3 se muestra una configuración típica de un enlace de banda ancha hogareño. En general, ni el router del cliente ni el ISP tienen soporte para IPv6 a diferencia del host del cliente, que probablemente utilizará como SO Windows XP / Vista / Seven o GNU/Linux y tendrá la posibilidad de tener doble pila. La conexión a la red IPv4 es mediante una IP pública dinámica, lo que representa un inconveniente a la hora de establecer algún tipo de túnel.

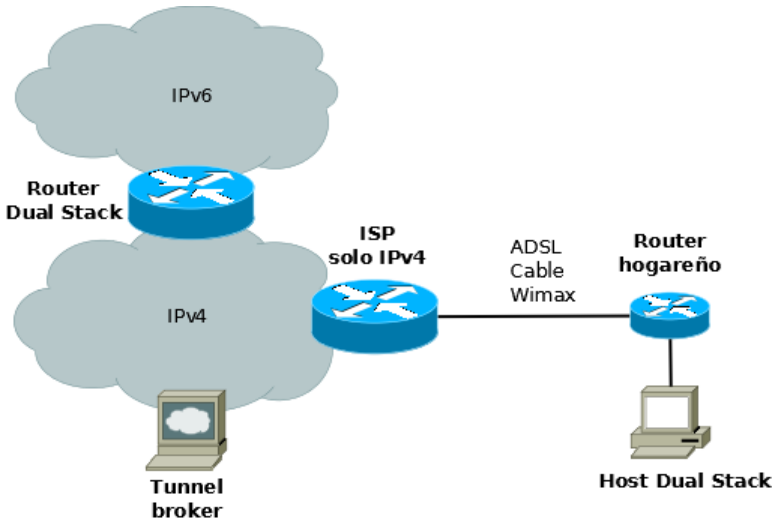

Gráfico 3: Enlace Hogareño

2.4 Escenario D:

Acceso a internet de red solo IPv6 a través de ISP solo IPv4

En este escenario se plantea una red de área local con hosts que soportan únicamente IPv6, como se indica en el Gráfico 4. El problema radica en cómo lograr que estos hosts accedan a servicios brindados por servidores solo IPv4. Una posible solución es utilizar el ALG Prox6 [4]. De esta manera, los requerimientos IPv6 de la LAN se realizarán a la aplicación Prox6, mientras que ésta, resolverá el requerimiento IPv4 correspondiente.

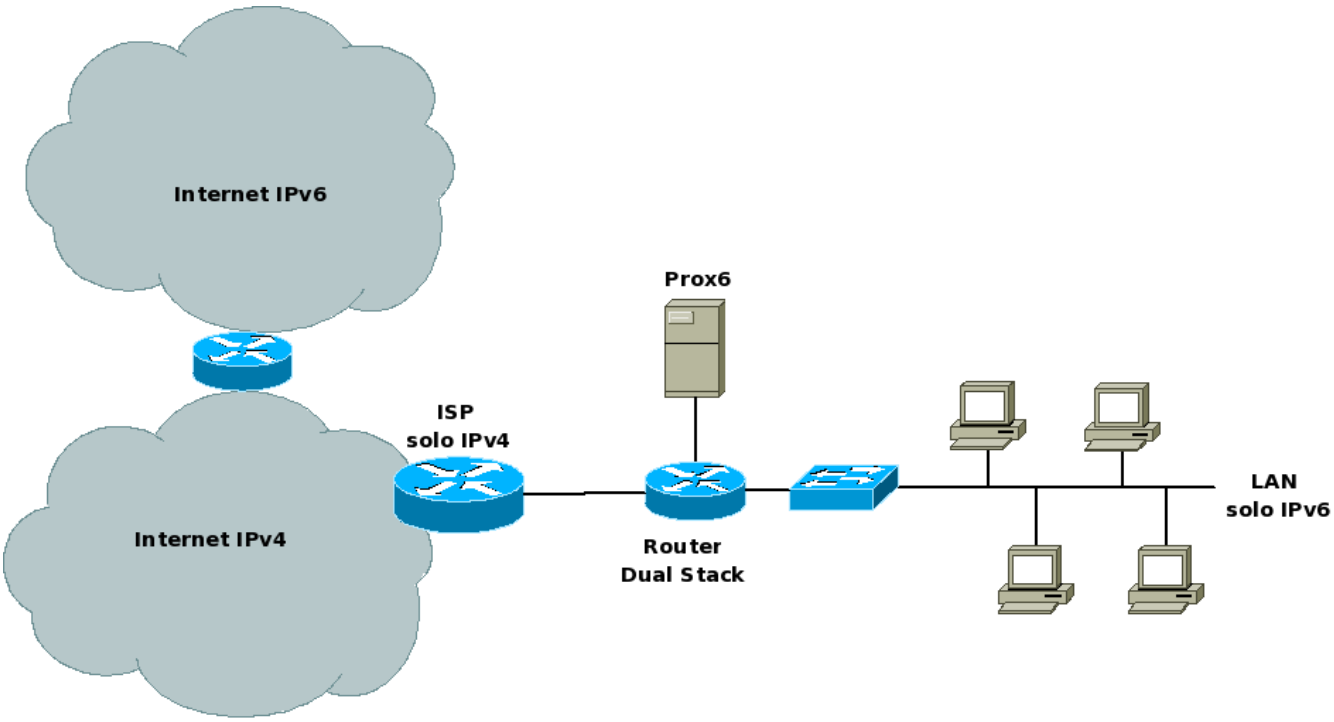

Gráfico 4: Red Local IPv6 Only

3 Descripción de Mecanismos de Transición

El soporte de IPv6 está ahora extensamente disponible tanto para la mayoría de los hosts como para routers. Actualmente hay métodos y procedimientos para configurar y ejecutar IPv6 en un host sencillo, en una red hogareña o en un gran sitio empresarial.

Si se desea tener comunicación con otros sistemas de IPv6, es vital obtener una comunicación con la Internet global IPv6 configurando convenientemente los sistemas locales.

Hoy, ésto puede realizarse de forma nativa, en caso de tener un proveedor ISP que lo soporte, o más normalmente, con alguna técnica de túnel IPv6-en-IPv4 [5].

Los despliegues de redes “IPv6-only” son raros, pero hay cada vez más redes experimentales, que básicamente se utilizan para poder evaluar las características y ventajas que tendrá IPv6 una vez establecido definitivamente.

Sin embargo, la realidad nos muestra que los sitios en producción que despliegan IPv6 no hacen la transición directamente a IPv6, sino primero hacen una transición a un estado intermedio donde coexisten IPv4 e IPv6 (mecanismo de pila dual) [6]. El ambiente dual-stack permite introducir gradualmente nodos solo IPv6 a medida que se retiran paulatinamente nodos solo IPv4.

La expansión de la funcionalidad de IPv6 desde una pequeña infraestructura a una red grande puede ser una aventura compleja y difícil. Pero si se desea llegar a buen puerto, el despliegue debe hacerse en una manera escalonada y progresiva, comenzando por la transición de los propios nodos, luego la transición de los accesos a Internet y finalmente los servicios. Por ejemplo, para un sitio grande hay muchos requisitos y condiciones diferentes, lo que hace necesario emplear varios mecanismos de transición según las peculiaridades de, por ejemplo, una subred dada, ambiente inalámbrico o móvil, una tecnología dial-in, etc. [7].

3.1 Pila Doble (“Dual Stack”)

Una de las maneras conceptualmente más fáciles de introducir IPv6 en una red, es el denominado "mecanismo de pila doble” que es describe en el RFC 2893 [2]. Por este método un host o un router tendrán ambas pilas de protocolos, IPv4 e IPv6, provistas directamente como un componente del sistema operativo. Cada nodo, denominado "nodo IPv4/IPv6”, se configura con ambas direcciones IPv4 e IPv6. Por consiguiente las dos pilas envían y reciben datagramas que pertenecen a ambos protocolos y así podrán comunicarse con cada nodo IPv4 e IPv6 en la red. Ésta es la manera más simple y más deseable de coexistencia para IPv4 e IPv6 y es, en general, el próximo paso en la evolución, antes de una transición más profunda, hacia una Internet mundial solo IPv6 (en el futuro a largo plazo).

La convivencia se logra debido a que el campo que indica el tipo de “payload” para la capa de acceso al medio es distinta para ambos protocolos (0x0800 y 0x86dd , para IPv4 e IPv6 respectivamente), de esta manera los paquetes que llegan al host dual stack son desencapsulados y entregados al stack

correspondiente dependiendo de dicho valor, dentro del sistema operativo.

Un desafío en el despliegue de una red IPv6/IPv4 con pila doble, es la configuración del ruteo tanto externo como interno para ambos protocolos. Si se ha estado usando OSPFv2, por ejemplo, para el ruteo entre sitios antes de agregar IPv6 a la Capa 3 de la red, se debería hacer la transición a un protocolo que sea capaz de encaminar ambos protocolos IPv4 e IPv6 como IS-IS u OSPFv3 en vez de OSPFv2.

Otro de los desafíos es la interacción entre estos dos protocolos, y como manejarla; suponiendo que una red dual stack generalmente se vinculará con redes solo IPv4 externas. Un ejemplo de estas interacciones lo podemos encontrar en el desarrollo de un servidor de correo SMTP, y como los registros MX del sistema de DNS proveerán ambos tipos de direcciones IP, e incluso como se comportará en el manejo de fallas entre los protocolos.

No existe un mecanismo de transición real usado en el escenario dual stack, debido a que “Dual Stack” integra en si mismo el soporte IPv6. Para construir un nodo de pila dual, solo es necesario habilitar en el sistema operativo el soporte IPv6 [8].

De esta manera el nodo se convierte en un nodo "híbrido" y dependiendo de la resolución de nombres será el protocolo que usará para cada requerimiento en particular. De acuerdo a lo mostrado en el Gráfico 5.

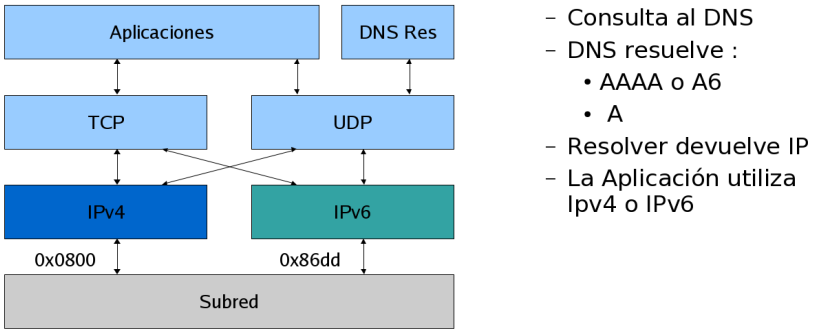

Gráfico 5: Detalle de funcionamiento de Pila Dual

Un backbone se dice que es de pila dual si todos los routers y switchs capa 3 pueden manejar tanto IPv4 como IPv6.

3.2 Túneles

Para crear redes IPv6, las técnicas de Túneles operan sobre la infraestructura disponible de IPv4 sin tener que hacer cualquier cambio en el ruteo IPv4 ni en los routers. Este método se usa a menudo donde la infraestructura completa, o parte de ella, no es todavía capaz de ofrecer la funcionalidad de IPv6 nativa. Por consiguiente el tráfico de IPv6 tiene que cruzar la red IPv4 existente, lo cual es posible por medio de varias técnicas del “tunnelling”. Estas técnicas son a menudo escogidas como primer paso

para probar el nuevo protocolo IPv6.

“Tunnelling”, que es también llamado encapsulamiento, es un proceso por el que todo un protocolo se encapsula dentro del área de datos de otro protocolo, permitiendo llevar así los datos originales sobre el segundo protocolo. Este mecanismo puede usarse cuando dos nodos o redes que usan el mismo protocolo quieren comunicarse sobre una red que usa otro protocolo.

En el Gráfico 6 se observa en detalle un ejemplo de como se realiza el encapsulamiento de IPv6 sobre IPv4.

El proceso de túnel involucra tres pasos: encapsulamiento, desencapsulamiento y administración del túnel. Se requiere dos extremos del túnel, los cuales son generalmente nodos que tienen implementada pila dual IPv4/IPv6 (normalmente routers), y son los que se ocupan del encapsulamiento y el descapsulamiento. Existen problemas del desempeño asociados con el “tunnelling”, como son la latencia debido a que deben realizar los procesos de encapsulamiento y desencapsulamiento. Hay un inconveniente más de desempeño debido al uso de ancho de banda adicional (payload overhead), aunque este último es normalmente marginal.

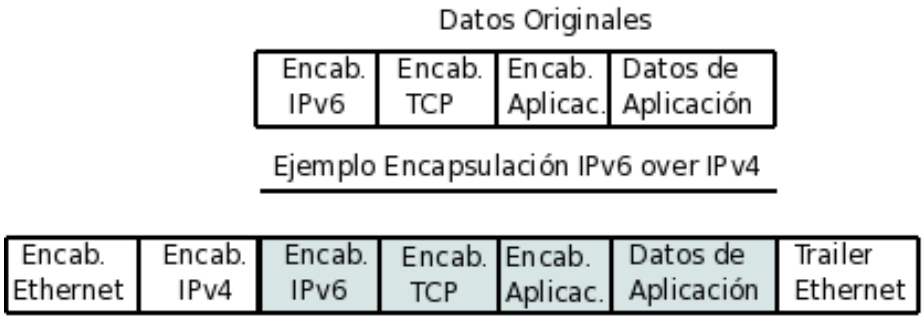

Gráfico 6: Encapsulamiento IPv6 sobre IPv4

Un túnel puede configurarse de cuatro maneras diferentes:

1. De router a router, que se aplica a un segmento de la ruta de extremo a extremo entre dos host. Éste probablemente es el método más común.
2. Host a router, que se aplica al primer segmento de la ruta de extremo a extremo entre dos host, tal como puede encontrarse en un tunnel broker.
3. Host to Host, se aplica a la ruta completa extremo a extremo entre dos host.
4. Router a host, que se aplica al último segmento de la ruta de extremo a extremo entre dos host.

Dependiendo de qué tipo se use, un túnel podría ser “configurado” (ambos lados necesitan ser configurados), "semi-configurado" (sólo un extremo tiene que ser configurado, los otros lados actúan como pasarelas) o "automático", donde no es necesario hacer casi nada para que los dos host puedan para comunicarse vía un túnel.

3.2.1 Implementación de Túneles

Esta sección describe los métodos para transportar IPv6 sobre redes IPv4 existentes, lo que trae aparejado distintos tipos de mecanismos de túneles.

Túneles Configurados o Manuales

Los túneles configurados se actualizaron en el RFC2893 [2] como túneles IPv6-sobre-IPv4. Éste tipo de túneles son punto a punto y deben ser configurados manualmente en los dos extremos, debido a que las direcciones IPv4 de los extremos dependen de la configuración de dichos nodos.

Si bien es tediosa la configuración, pues deben intervenir los administradores de ambos extremos, a la hora de controlar las rutas y reducir ataques de denegación de servicio son más convenientes que los túneles automáticos. Los túneles configurados se emplean mayormente para proveer conectividad IPv6 al exterior para un red completa.

Debido a que en la actualidad no hay demasiados proveedores que ofrezcan conectividad IPv6, si se quiere conectividad hacia sitios nativos IPv6, una forma muy estable y segura es que el tráfico IPv6 sea encaminado mediante un túnel IPv6-sobre-IPv4. Normalmente la configuración de rutas estáticas permitirá encaminar todo el tráfico IPv6, pero igualmente se puede configurar sobre el túnel el protocolo de encaminamiento BGP.

Como este tipo de túneles deben ser configurados manualmente, es recomendable no usar demasiados para dar conectividad IPv6 a un sitio. Hay otros métodos específicamente diseñados para este propósito, como tunnel brokers, 6to4, Teredo o ISATAP (los cuales de detallan a continuación).

A modo de ejemplo, se detalla en el Gráfico 7 el acceso a IPv6 que tenía toda la red “Codarec6” hacia la red 6Bone en los primeros tiempos del TestBed (2001-2002), a través de un túnel manual configurado entre la UTN regional Mendoza y la UTN regional La Plata. [3]

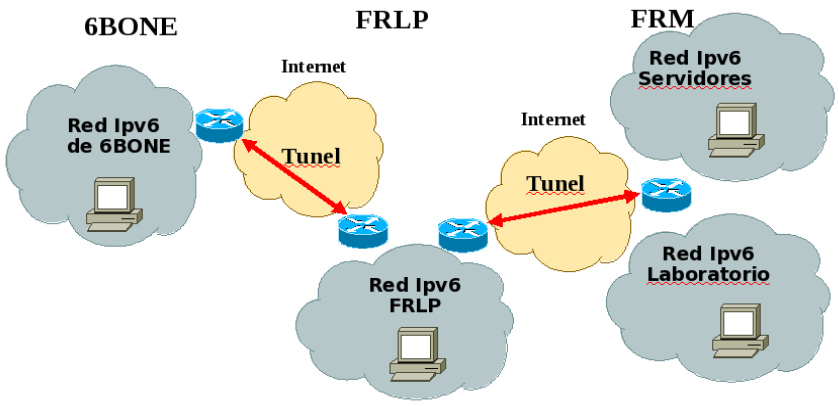

Gráfico 7: Acceso a 6Bone de Codarec6

Túnel Broker

En lugar de configurar manualmente cada extremo de los túneles, es posible correr scripts para automatizar la tarea, sobre todo si las direcciones IPv4 globales son dinámicas. Una alternativa "automática" se denomina “tunnel broker” y fue introducida en el RFC3053 [9].

Al igual que los túneles manuales, el tunnel broker es útil cuando un host quiere conectarse a una red IPv6 y su sistema operativo tiene dual Stack.

La filosofía básica de los tunnel broker es permitir al usuario que acceda a una página web, opcionalmente que ingrese datos de autentificación a continuación recibe un script que deberá ejecutar de manera local para establecer un túnel IPv4-IPv6 contra el servidor del tunnel broker.

El proveedor del servicio de tunnel broker debe brindar el servicio HTTP para IPv4 y contar con un router dual stack, capaz de aceptar comandos de configuración automáticos para crear los nuevos túneles hacia los host remotos de los clientes.

Los tunnel broker se pueden implementar de distintas maneras, no estando limitados a túneles IPv6-sobre-IPv4; se pueden usar túneles Layer 2 o GRE (Generic Routing Encapsulation), por ejemplo. El requerimiento principal para este servicio es saber que túnel pertenece a cual usuario. También debería tener alguna autenticación para acceder al servicio.

Los Tunnel brokers, como se mencionó anteriormente, se usan para conectar hosts de pila dual aislados, pero también se pueden usar para conectar redes enteras. En este último caso, el host que obtenga el túnel debería ser en realidad un Router con soporte IPv6.

Un tunnel broker es una importante ayuda para la transición; habilita fácilmente el acceso a redes IPv6, y en la actualidad existen varios sitios que ofrecen acceso gratuito a estos servicios. A la hora de elegir un servicio de tunnel broker, es importante la cantidad de saltos o hops hasta el mismo, pues todos los paquetes traficados con IPv6 tendrán adicionada esa demora; por lo que es recomendable seleccionar como proveedores de tunnel brokers, sitios nacionales.

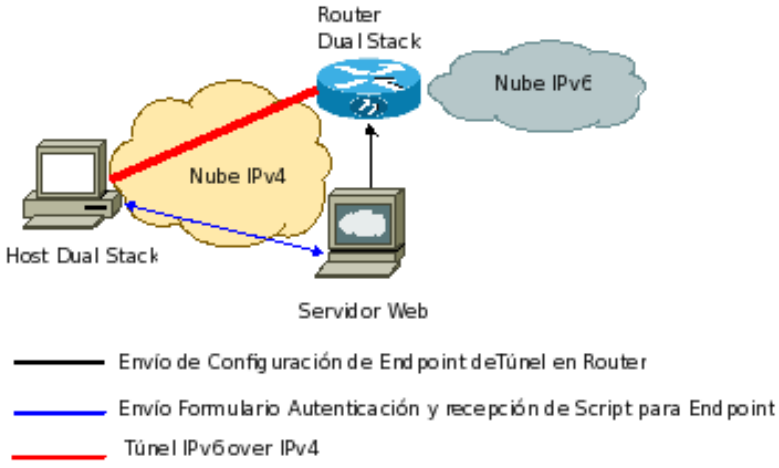

Gráfico 8: Tunnel Broker

En el Gráfico 8 se muestra el esquema de un Tunnel Broker. En primer lugar el host se contacta como cliente a un servidor HTTP donde se autentica y responde con el script que el host debe ejecutar. A continuación el servidor HTTP envía al extremo local del Tunnel Broker la configuración que debe relizar el router local. Finalmente queda establecido el Tunnel IPv6 over IPv4 entre el Host Dual Stack y el router del Tunnel Broker, asi de esta manera el Host accede a la nube IPv6. Cabe destacar que no es obligatorio que el servidor HTTP y el Router sean nodos distintos, se pueden implementar los dos servicios en el mismo host.

Túneles Automáticos

Este tipo de mecanismo de túnel fue uno de los primeros en desarrollarse, pero también uno de los primeros en ser reemplazado por otros métodos más sofisticados. Usa como extremos del túnel direcciones IPv6 del tipo IPv4-compatible. [10]

La dirección del nodo destino está especificada en el paquete que está siendo encapsulado por el túnel. Debido a eso, este método solo puede ser usado en comunicaciones host-a-host o router-a-host, pues son los únicos esquemas en los que el nodo destino también es el extremo del túnel. Esta es la causa de que solo funcione para túneles IPv6 over IPv4 y no en sentido inverso.

Si bien el mecanismo no es obsoleto, se recomienda seleccionar otras soluciones como ISATAP o 6to4. Una de las causas se basa en que la conectividad que resulta de estos túneles carece de una estructura en el dominio IPv6.

Túneles 6to4

Este mecanismo de transición está presentado en el año 2000 [11], como una forma de túneles automáticos router-a-router que utiliza un prefijo asignado por IANA (2002::/16) para designar los

sitios participantes en la técnica 6to4.

Permite que dominios IPv6 aislados se comuniquen con otros dominios IPv6 con una mínima configuración. Un sitio IPv6 aislado se asignará a si mismo una dirección global con prefijo 2002:ADDR_IPv4::/48, donde ADDR_IPv4 es la dirección global IPv4 configurada en la interfaz de salida del router de acceso a Internet IPv4. Esta dirección IPv4 debe ser única y estática, por lo que no es recomendable establecer un túnel 6to4 con direcciones de internet IPV4 dinámicas.

Este prefijo tiene exactamente el mismo formato que un prefijo normal /48, y por ello permite a un dominio IPv6 usarlo como cualquier otro prefijo /48 válido. En un escenario en donde dominios 6to4 quieren comunicarse entre si, no es necesaria la configuración explícita de los túneles. Las direcciones IPv4 de los extremos del túnel son determinados al extraerlos del prefijo global IPv6 de la dirección destino del paquete IPv6 a transmitir. En este escenario se pueden comunicar entre sí un número arbitrario de dominios sin necesidad de configurar ningún túnel, a diferencia del aumento exponencial de túneles manuales que se debían configurar para el mismo escenario. Adicionalmente, los routers 6to4 no necesitan correr ningún protocolo de enrutamiento IPv6, pues el enrutamiento IPv4 es el encargado de realizar la tarea. Se puede ver en el Gráfico 9 los aspectos descriptos.

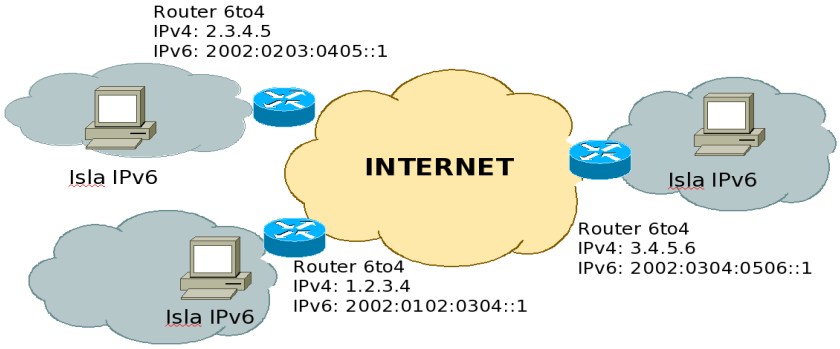

Gráfico 9: Direcciones IPv6 de routers 6to4

Para permitir que los host y las redes que utilizan este tipo de túnel puedan intercambiar tráfico con redes IPv6 nativas, se definieron Routers Relay, de acuerdo al RFC 3068. Un Router Relay interconecta redes IPv4 con redes IPv6. Los paquetes 6to4 que arriban a un Router Relay desde una interfaz IPv4, se desencapsularán y continuarán su transito hacia la red IPv6 nativa, mientras los paquetes IPv6 que arriben y tengan como destino una dirección IPv6 con prefijo 2002::/16, serán encapsulados en un paquete IPv4 hacia la red IPv4.

Para evitar la configuración manual del default gateway IPv6, que debe tener el formato 2002:ADDR_IPv4_Router_Relay:/128 , se definió la dirección “Equivalent IPv4 unicast address” que es una dirección IPv4 unicast (192.88.99.1). Los paquetes enviados a esa dirección son tratados como si hubieran sido enviados a la dirección anycast del Router Relay. De esta manera quedará siempre el default gateway hacia IPv6 como 2002:c058:6301:: .

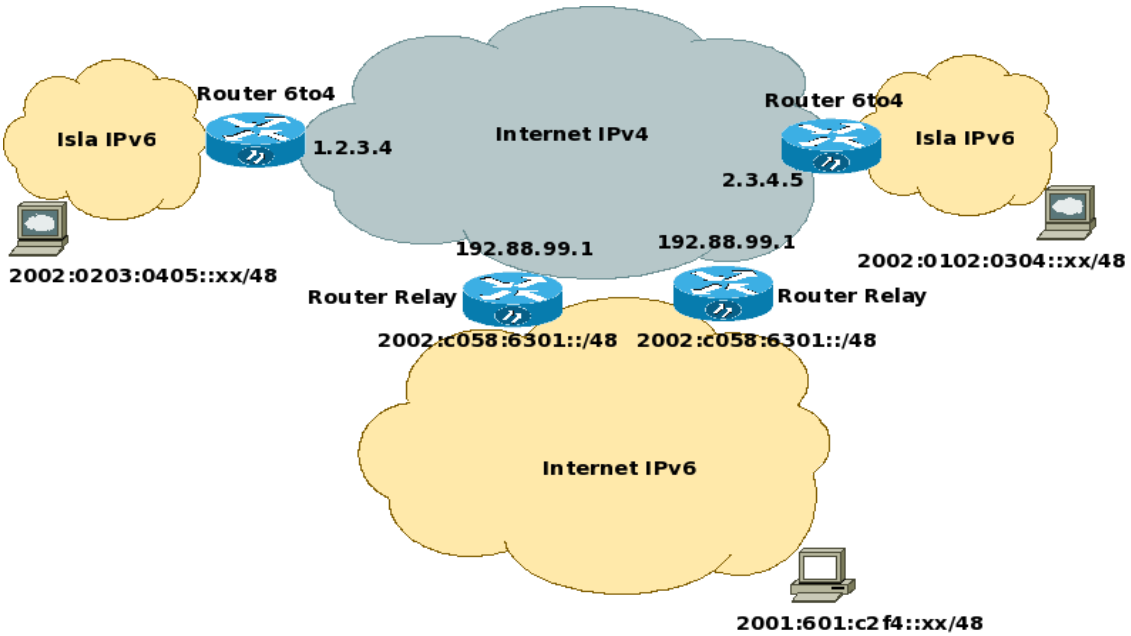

Gráfico 10: Tunnel 6to4 y acceso a IPv6 Nativa

Se puede observar en el Gráfico 10 el funcionamiento de los Routers Relays.

Túnel ISATAP

Una alternativa a los túneles 6to4 son los túneles ISATAP (Intra-Site Automatic Tunnel Addressing Protocol) [12].

ISATAP también utiliza la infraestructura IPv4 como enlace virtual, pero no hace uso de multidifusión, por lo que el enlace es NBMA (Non-Broadcast Multiple Access).

ISATAP, al igual que 6to4, crea un identificador de interfaz basado en la dirección IPv4 de la interfaz. Las direcciones en ISATAP pueden ser configuradas manual o automáticamente, pero la dirección IPv4 de la interfaz debe estar embebida en los últimos 32 bits de la nueva dirección IPv6.

Al igual que 6to4, la dirección IPv4 debe ser única, y si es usada para acceder a Internet debe ser global.

La multidifusión se usa normalmente para operaciones de “neighbour discovery”, como resolución de direcciones y solicitud de router. Como la dirección IPv4 siempre está embebida en la dirección IPv6, la resolución de direcciones es trivial. Se debe tener en cuenta que para que funcionen las solicitudes de router, el host debe haber aprendido de alguna manera las direcciones IPv4 de los posibles routers ISATAP (por DHCP, DNS, TEP, configuración manual etc), y enviará entonces solicitudes de manera unicast. El router siempre envía “advertisements” de manera unicast y solo como respuesta a la

solicitud del host. Cada host ISATAP enviará regularmente solicitudes a los routers ISATAP que conozca. ISATAP se ha implementado en algunas plataformas como Windows XP y Cisco IOS [13], mientras que no para GNU/Linux ni BSD, pues se sacó de la pila dual del kernel USAGI para Linux.

Túnel Teredo

Teredo, también conocido como un traslado de direcciones de red (NAT) para IPv6, se diseñó para que hosts IPv4 obtengan direcciones IPv6 a través una o más capas de NAT [14] creando túneles sobre el protocolo UDP. Éste es un mecanismo de túneles automáticos de host a host que provee conectividad IPv6, mientras que los hosts dual stack se ubican detrás de uno o más NATs, por encapsulamiento de paquetes IPv6 en mensajes UDP de IPv4. Teredo usa dos entidades: un Server Teredo y un Relay Teredo. El Server escucha requerimiento de los clientes en el puerto 3544 del protocolo UDP, respondiendo con una dirección IPv6 para que la usen. Las direcciones Teredo tienen la siguiente estructura: Prefijo Teredo (32 bits) : Dirección IPv4 del Servidor Teredo : Flags (16 bits) : Puerto externo (16 bits) : Dirección externa (32bit).

El método reenvía paquetes IPv6 (con IPv4 encapsulado) enviados desde el cliente al Teredo Relay, y también redirige los paquetes recibidos desde el Teredo Relay. De hecho el Relay actúa como un router IPv6. Esta técnica es por lejos una herramienta de “último recurso”, solo diseñada para cuando ningún otro método funcione. El método usado por Teredo es complejo, y no se puede garantizar que trabaje correctamente debido a la gran cantidad de distintas implementaciones de NAT existentes.

3.3 Traslación de Protocolos:

Los métodos de traslación fueron desarrollados para lograr la comunicación entre hosts solo IPv4 y hosts solo IPv6. Es común que algunas aplicaciones o sistemas operativos no estén portados todavía a IPv6, o lo que es peor, no se cuente con el código fuente para realizar las modificaciones correspondientes. A raíz de esta necesidad es que se idearon las traslaciones de protocolos, entre ellas encontramos:

3.3.1 SIIT y NAT-PT:

SIIT (Stateless Ip/Icmp Traslator) [15] y NAT-PT (Network Address Traslation – Protocol Traslation) [16] son mecanismos, a diferencia de los túneles, que traducen encabezados de IPv4 a IPv6 y viceversa.

Estas técnicas además de compartir los problemas normales de NAT deben lidiar con la semántica de convertir correctamente los campos de ambos protocolos, por lo que su implementación es bastante compleja. En algunos casos en el proceso de conversión se pierde información de encabezado. Por esta razón el IETF recomienda estos métodos solo como último recurso.

3.3.2 BIS:

BIS (Bump In the stack) [17] es una aproximación similar a la anterior SIIT, pero implementada directamente en el sistema operativo de cada host (entre el módulo de TCP/IP y el driver de placa de red), aunque solo disponible para aplicaciones IPv4 y redes IPv6. Es complejo implementarlo y muy poco utilizado. Es necesario contar con el código fuente del sistema operativo.

3.3.3 BIA:

BIA (Bump in the API) [18] agrega una API de traslación entre la API de Socket y el stack TCP/IP, permitiendo una mejora al método BIS en cuanto a la dependencia del driver de red, pero tiene las mismas limitaciones que BIS.

3.3.4 TRT:

TRT (Transport Relay Traslator) [19] es una conversión de protocolos en la capa de transporte que usa como pieza fundamental un DNS proxy. Éste recibe consultas de los hosts IPv6 y si el nombre requerido está asociado IPv4, devuelve una dirección IPv6 con el formato *prefijoIPv6 (64 bits) + ceros (32 bits) + direcciónIPv4 (32 bits)*.

La información de ruteo debe estar configurada de manera que los paquetes con destino *prefijoIPv6::/64* sean enviados hacia el servidor TRT que recibe las conexiones TCP/IPv6 y obtiene de la parte más baja de la dirección destino la dirección IPv4 a la que realmente se desea acceder. Con esta información inicia una conexión TCP/IPv4 y luego reenvía el tráfico entre las dos conexiones.

3.3.5 ALG:

ALG (Application Layer Gateway) es una traslación realizada en la capa de aplicación. No hay RFCs específicas a seguir, pues su implementación depende del protocolo de capa aplicación al que se dará soporte.

4 Implementación de distintos Métodos en el Test Bed

4.1 Túnel Manual

4.1.1 Requisitos para la Implementación

Para la implementación se necesitaron los siguientes elementos:

- 2 Router con sistema operativo GNU/Linux , distribución Fedora C10 o superior :
- Verificación de IPv6 habilitado como módulo en el kernel.

- Instalación el paquete initscripts versión 6.02 o superior.
- Instalación el paquete iproute.
- Disponibilidad de una dirección IPv4 global y única, preferentemente estática para ambos Routers.
- No estar bloqueado el protocolo 41(IPv6-in-IPv4) por parte del ISP.

4.1.2 Configuración

Se realizaron los siguientes pasos en ambos Routers:
Habilitación IPv6 en el arranque y especificación del túnel manual, según se muestra en el Gráfico 11.

```
#cat /etc/redhat-release
Fedora Core release 11 (Leonidas)
lsmod | egrep ipv6
ipv6                  275969  35 sit
# rpm -q initscripts
                              initscripts-8.31.6-1
# rpm -q iproute
iproute-2.6.16-1.fc5
#
# /sbin/ip tunnel add <nombre_tunel> mode sit ttl <ttldefault> remote <ipv4_remota> local <ipv4_local>
# /sbin/ip link set dev <nombre_tunel> up
# /sbin/ip addr add <dirección_ipv6> dev <nombre_tunel>
# /sbin/ip -6 route add <prefijo> dev <nombre_tunel> metric 1
```

Gráfico 11: validación de requisitos

4.1.3 Pruebas

Para nuestro caso se asignó en el Codarec6 Test Bed direcciones IPv4 Globales para el router1 la 1.1.1.1 y para el router2 la 2.2.2.2. Por su lado se asignaron direcciones de Red para IPv6 a la red del router1 1001:1::1/64 y para el router2 1001:2::1/64, validando la conectividad según el Gráfico 12.

```
#ping6 1001:2::1
PING 1001:2::1 56 data bytes
64 bytes from 1001:2::1: icmp_seq=1 ttl=64 time=0.037 ms
64 bytes from 1001:2::1: icmp_seq=2 ttl=64 time=0.048 ms
64 bytes from 1001:2::1: icmp_seq=3 ttl=64 time=0.041 ms
64 bytes from 1001:2::1: icmp_seq=4 ttl=64 time=0.046 ms
64 bytes from 1001:2::1: icmp_seq=5 ttl=64 time=0.041 ms
^C
--- 1001:2::1 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4311ms
rtt min/avg/max/mdev = 0.037/0.042/0.048/0.008 ms
#
```

Gráfico 12: Ensayo de conectividad desde el router1 al router2

4.2 Tunnel 6to4

4.2.1 Requisitos para la Implementación

Para la implementación se necesitaron los siguientes elementos:

- Router con sistema operativo GNU/Linux , distribución Fedora C10 o superior:

- Verificación de IPv6 habilitado como módulo en el kernel.
- Instalación el paquete initscripts versión 6.02 o superior.
- Instalación el paquete iproute.
- Instalación de paquete radvd.
- Disponer de una dirección IPv4 global y única, preferentemente estática.
- No estar bloqueado el protocolo 41(IPv6-in-IPv4) por parte del ISP.

Se pudo verificar en el Gráfico 13 que el sistema operativo cumplió con los requisitos enumerados

```
#cat /etc/redhat-release
Fedora Core release 11 (Leonidas)
lsmod | egrep ipv6
ipv6                  275969  35 sit
# rpm -q initscripts
initscripts-8.31.6-1
# rpm -q iproute
iproute-2.6.16-1.fc5
# rpm -q radvd
radvd-0.9.1-1.1.1
```

Gráfico13: Comandos para verificar los requisitos

4.2.2 Configuración

Se realizaron los siguientes pasos en el Router:
Habilitación IPv6 en el arranque y especificación de que la pseudo interfaz 6to4 sea el gateway IPv6 por defecto. Para ello se adicionaron las siguientes directivas al archivo de configuración general de redes (/etc/sysconfig/network), como indica el Gráfico 14.

```
#echo “NETWORKING_IPV6=yes” >> /etc/sysconfig/network
#echo “IPV6INIT=yes” >> /etc/sysconfig/network
#echo“"IPV6_DEFAULTDEV=tun6to4">>/etc/sysconfig/network
```

Gráfico 14: Habilitación IPv6

Se configuró la interfaz conectada a la dirección IPv4 global, que en nuestro caso fue la eth1, para que al momento de iniciarse, automáticamente iniciara el túnel 6to4, de acuerdo al Gráfico 15.

```
#echo ”IPV6INIT=yes” >> /etc/sysconfig/network-scripts/ifcfg-eth1
#echo ” IPV6TO4INIT=yes” >> /etc/sysconfig/network-scripts/ifcfg-eth1
```

Gráfico 15: Interfaz tun6to4

Se detuvo e inició nuevamente la interfaz real, verificando el inicio de la pseudo Interfaz tun6to4. Cabe destacar que, de acuerdo a lo detallado en el punto 3.2.3, el formato de la dirección global IPv6 cumplió con lo especificado [20]. El Gráfico 16 mostró que los bits del 17 al 48 de la dirección global IPv6 (aad2:ee19) se correspondieron con la dirección pública IPv4 (170.210.238.25), a excepción de la notación en la que se presentaron. Queda como tarea para el lector la conversión de representaciones.

```
#  ifdown eth1 ; ifup eth1
#  ip address show tun6to4
8: tun6to4@NONE: <NOARP,UP,10000> mtu 1480
    qdisc noqueue
    link/sit 170.210.238.25 brd 0.0.0.0
```

```
   inet6 2002:aad2:ee19::1/16 scope global
     valid_lft forever preferred_lft forever
```

Gráfico 16: Dirección IPv6 Global

A partir de ese momento quedó configurado y operativo el Router con soporte IPv6 y una dirección IPv6 global.

4.2.3 Entrega de IPv6 a la red interna

Para proveer IPv6 a la red interna (conectada en nuestro caso a la eth0), se habilitó el encaminamiento de paquetes IPv6 como se indica en el Gráfico 17.

```
# echo “IPV6FORWARDING=yes” >> /etc/sysconfig/network
```

Gráfico 17: Habilitación de ruteo IPv6

Adicionalmente se configuró y arrancó el servicio de router advertisement [21], llamado radvd. Se detalla también en el Gráfico 18.

```
# cat /etc/radvd.conf
              interface eth0
              {
    AdvSendAdvert on;
                  MinRtrAdvInterval 3;
    MaxRtrAdvInterval 10;
    prefix 0:0:0:c0da::/64
                  {
          Base6to4Interface eth1;
                      AdvPreferredLifetime 120;
                      AdvValidLifetime 300;
                  };
              };
#
# /etc/initd.d/radvd start
```

Gráfico 18: Configuración. Radvd

Se pudo observar que al configurar el prefix, se configuraron los 16 bits de la subred, de acuerdo a lo especificado por [20].

4.2.4 Ensayos

Se ejecutaron las pruebas de conectividad mostradas en el Gráfico 19 desde el Router IPv6 hacia sitios IPv6 nativos.

```
# host -t AAAA  whatismyv6.com
 whatismyv6.com has IPv6 address 2001:4810::110
# ping6 whatismyv6.com
PING whatismyv6.com(www.whatismyv6.com) 56 data bytes
64 bytes from www.whatismyv6.com: icmpseq=0 ttl=58 time=317 ms
64 bytes from www.whatismyv6.com: icmpseq=1 ttl=58 time=315 ms
64 bytes from www.whatismyv6.com: icmpseq=2 ttl=58 time=316 ms
64 bytes from www.whatismyv6.com: icmpseq=3 ttl=58 time=354 ms
64 bytes from www.whatismyv6.com: icmpseq=4 ttl=58 time=317 ms
--- whatismyv6.com ping statistics ---
5 packets transmitted, 5 received,
0% packet loss, time 4007ms
rtt min/avg/max/mdev
= 315.759/324.454/354.514/15.061 ms, pipe 2
```

Gráfico 19: Verificaciones para Router

4.3 Tunnel Broker

Se desarrolló una aplicación, dividida en dos componentes, Servidor Web y Router Dual Stack que luego fue implementada en el Test Bed.

4.3.1 Servidor Web:

Se utilizó para el intercambio de información entre el usuario en el Host Dual Stack y el servicio de Broker. La función se realizó con un servidor HTTP, implementado sobre sistema operativo GNU/Linux y aplicación Apache [22] con módulos de extensión de lenguaje PHP.

Las tareas que realiza son:

- Autenticación de usuarios o alta de nuevos usuarios, almacenando la información en el sistema de archivos.
- Visualización de túneles existentes para el usuario previamente autenticado.
- Creación de nuevos túneles, permitiendo seleccionar al usuario el tipo de sistema operativo de su extremo del Túnel y tipo de Túnel, Túnel Host-to Host o Host-to-Net.
- Envío de script a Router Dual Stack via ssh para creación o baja de túnel.
- Envío de script al cliente, para que éste lo ejecute (el script variará en función de la previa elección de su sistema operativo).

4.3.2 Router Dual Stack:

Es el encargado de crear, modificar o borrar el endpoint en el router del Túnel IPv6 over IPv4 contra el host dual stack del cliente.

El router fue implementado sobre sistema operativo GNU/Linux, y los scripts realizados en bash scripting [23].

En la Gráfico 20 se muestra el código principal para el manejo de los túneles.

```
#!/bin/sh

unset PATH
IPTUNNEL=/sbin/iptunnel
IFCONFIG=/sbin/ifconfig
ROUTE=/sbin/route
IP=/sbin/ip
BASENAME=/bin/basename
NAME=$2
IPV4_REMOTE=$3
IPV6_LOCAL=$4
IPV6_REMOTE=$5

case $1 in
        "up")
        (
```

```
                        $IPTUNNEL add $NAME mode sit remote $IPV4_REMOTE &&
                        $IFCONFIG $NAME up &&
                        $IFCONFIG $NAME add $IPV6_LOCAL/128 &&
                        $ROUTE -A inet6 add $IPV6_REMOTE/128 dev $NAME
                        )2>&1 && exit 0 || exit 1
                        ;;

        "down")
        (
                        $ROUTE -A inet6 del $IPV6_REMOTE/128 &&
                        $IFCONFIG $NAME del $IPV6_LOCAL/128 &&
                        $IFCONFIG $NAME down &&
                        $IPTUNNEL del $NAME
                        )2>&1 && exit 0 || exit 1

                        ;;

        "status")
                        $IFCONFIG $NAME &> /dev/null && exit $?
                        ;;

        "stats")
                        $IP -s tun lis $NAME && exit $?
                        ;;

        *)

                        echo "USO: $($BASENAME $0) (up|down|status|stats) NAME IPV4_REMOTE
IPV6_LOCAL IPV6_REMOTE"
                        exit 1

                        ;;

esac
```

Gráfico 20: rutina principal de manejo de túneles

4.3.3 Ensayos

Para realizar el ensayo, se necesitó un host dual Stack con conectividad a Internet y un cliente Web.

Creación de usuario

En el cliente Web se escribió la URL http://codarec.frm.utn.edu.ar/tunnel/ . Se seleccionó “Si es un nuevo usuario haga Click Aqui”, y se llenó el formulario de registración , como muestra la Gráfico 21.

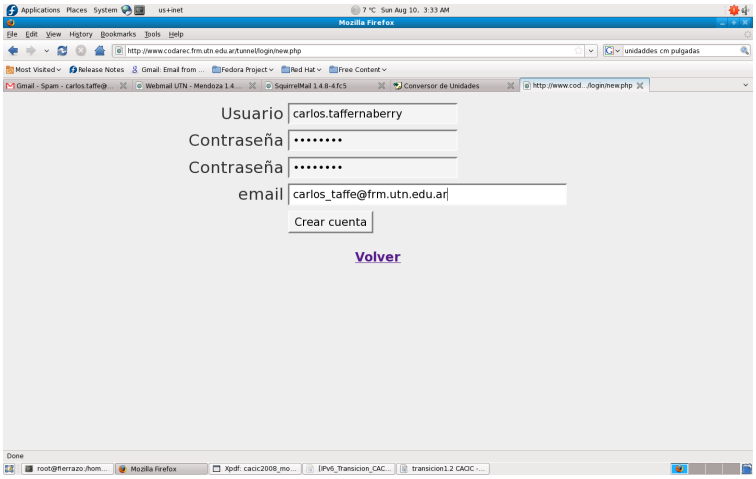

Gráfico 21: Creación de Usuario Nuevo

Creación de Túnel

Una vez que accedió, se seleccionó “Nuevo Túnel”, desplegando un formulario con los datos solicitados, como lo muestra el Gráfico 22.

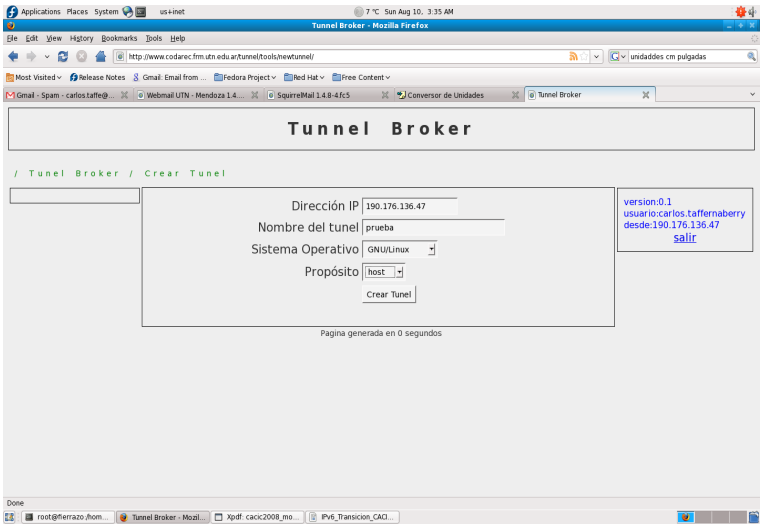

Gráfico 22: Creación de nuevo Túnel.

Ejecución de script de arranque de endpoint local

El punto anterior creó el endpoint del Túnel en el router del Codarec6. Solo restaba crear el endpoint local, siguiendo las instrucciones que aparecen al hacer click en "*Enviar Script*", como muestra el Gráfico 23.

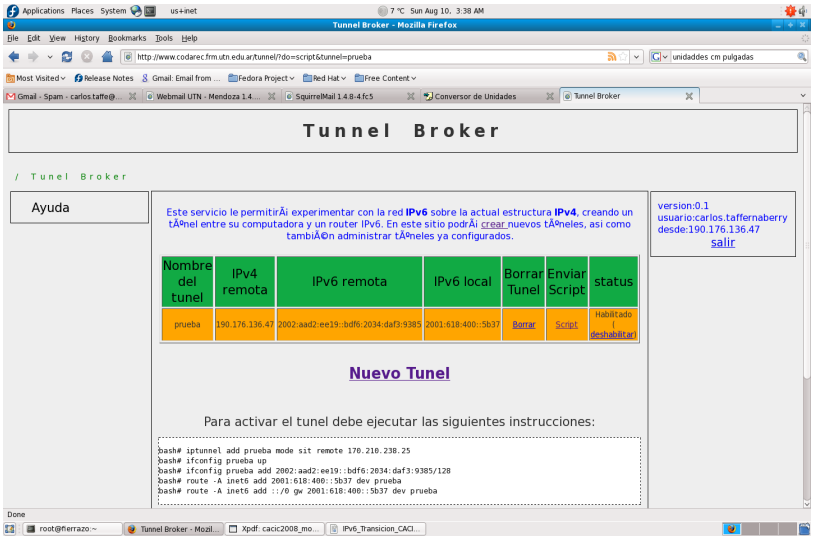

Gráfico 23: Instrucciones para ejecutar localmente

Acceso a la nube IPv6

Se verificó la creación de la nueva interfaz y la asignación de la dirección con prefijo global. Finalmente se verificó el acceso a IPv6 por medio del Túnel Broker, como lo demuestra el Gráfico 24.

```
$ ifconfig prueba

prueba    Link encap:IPv6-in-IPv4
          inet6 addr: fe80::ac04:2/64 Scope:Link
          inet6 addr: 2002:aad2:ee19:0:bdf6:2034:daf3:9385/128 Scope:Global
          inet6 addr: fe80::beb0:882f/64 Scope:Link
          UP POINTOPOINT RUNNING NOARP  MTU:1480  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)

$ ping6 www.deepspace6.net
PING www.deepspace6.net(2001:760:2e01:1::dead:beef) 56 data bytes
64 bytes from 2001:760:2e01:1::dead:beef:  icmp_seq=0 ttl=56 time=314 ms
64 bytes from 2001:760:2e01:1::dead:beef:   icmp_seq=1 ttl=56 time=313 ms
64 bytes from 2001:760:2e01:1::dead:beef:   icmp_seq=2 ttl=56 time=313 ms
64 bytes from 2001:760:2e01:1::dead:beef:   icmp_seq=3 ttl=56 time=313 ms
64 bytes from 2001:760:2e01:1::dead:beef:   icmp_seq=4 ttl=56 time=315 ms
www.deepspace6.net ping statistics ---
5 packets transmitted, 5 received, 0%
    packet loss, time 18636ms
rtt min/avg/max/mdev = 313.162/313.875/
    315.087/0.856 ms, pipe 2
```

Gráfico 24: Verificación de acceso

4.4 Prox6, Aplication Layer Gateway

Como resultado de lo evaluado para las alternativas del punto anterior, se decidió desarrollar e implementar un ALG para protocolo HTTP/HTTPS. Justifica esta decisión la facilidad de implementación y el no ser necesarios elementos adicionales, como un DNS Proxy o código fuente del sistema operativo.

Debido a que casi no existe diferencia entre un ALG y un proxy de aplicación, se intentó inicialmente utilizar un proxy HTTP/HTTPS para hacer la traslación de protocolos. Debido a que Squid [24], el proxy HTTP más difundido y utilizado, no tiene soporte IPv6, en sus versiones estables, por lo que finalmente se decidió realizar una aplicación propia para cumplir el objetivo.

4.4.1 Propuesta de funcionamiento

Se propone implementar la alternativa de los métodos anteriores como se muestra en el Gráfico 25.

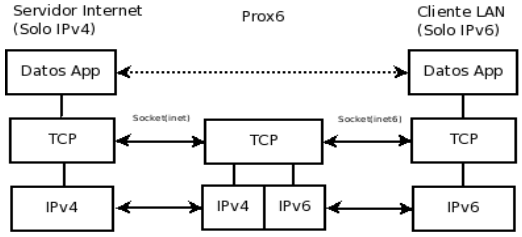

Gráfico 25: Diagrama que detalla la técnica de ALG utilizada por Prox6.

La idea básica de la aplicación Prox6 es permitir que el router R1 sea el encargado de intercambiar la información entre los dos extremos. Para ello es necesario que R1 tenga dual stack y ejecute la aplicación Prox6.

Los clientes locales iniciarán la comunicación utilizando un socket INET6, y hacen una solicitud a Prox6, que será almacenada en un buffer. Prox6, utilizando un socket INET, iniciará una nueva conexión como cliente al servicio InternetV4 solicitado por el cliente local y se reenvían los datos del requerimiento que almacenó previamente. La respuesta del servicio solicitado, será reenviado por Prox6 al cliente local, utilizando IPv6.

La aplicación prox6 debe resolver el nombre del dominio solicitado, antes de enviar el requerimiento hacia InternetV4.

4.4.2 Implementación

Se realizó un prototipo para poder evaluar el correcto funcionamiento de este mecanismo. La programación fue hecha en Python [25]. Se muestran a continuación las porciones de código más relevantes:

```
#Bucle principal
 def listen (self):
     escucha = socket(AF_INET6,SOCK_STREAM)#IPv6 Only
     escucha.bind(self.ADDR6,self.PORT)
     escucha.listen(10)        #hasta 10 a la espera
     while True:
         interno,cliente = escucha.accept()
```

```
        pid = os.fork()
        if pid != 0 :          #proceso hijo
            self.servicio()
        else:                  #proceso padre
            interno.close()

def servicio (self):
    PedidoProxy = interno.recv(self.buffer)
    externo = socket(AF_INET,SOCK_STREAM)# a InternetV4
    externo.connect(res[0][4][:2])
    externo.send(PedidoProxy) #reenvio requerimiento
    RespInternet = ''
    while RespInternet <> ''  #lee IPv4 -> escribe IPv6
        RespInternet = externo.recv(self.buffer)
        interno.send(RespInternet)
    interno.close()           #Termino el envio de IPV4
    externo.close()
    sys.exit()
```

La función “listen”, utilizando la API de socket, crea un socket de la familia AF_INET6 (IPv6) y espera que un cliente de la red Lan se conecte mediante el método escucha.accept(). Una vez conectado, usando la llamada a sistema os.fork() crea un hijo que se encarga de atender a cada uno de los clientes, usando el método self.servicio(). Esto permite concurrencia en el servidor.

La función “servicio” guarda en una variable local, PedidoProxy, el requerimiento original del cliente. Luego, usando la API de socket, crea un socket de la familia AF_INET (IPv4) y se conecta como cliente con el servidor al que tiene que hacer el requerimiento. Con externo.send(PedidoProxy) hace el requerimiento. Luego obtiene la respuesta con el método externo.recv(self.buffer) y lo reenvía usando el socket IPv6 al cliente original en la red Lan.

El router R1, donde se ejecutó la aplicación desarrollada fue host con GNU/Linux distribución Ubuntu 9.04.

El host de la red local con soporte solo IPv6 fue Windows Xp, simplemente por ser el sistema operativo más difundido, sin embargo se puede usar otro sistema operativo como GNU/Linux, Solaris, Mac Os o Windows Vista. También se hicieron pruebas exitosas con un teléfono celular Nokia modelo N95 con sistema operativo Symbian.

En ambos casos se desactivó el stack IPv4. Se configuró la aplicación cliente Http (navegador) para que utilice como proxy la dirección IPv6 local del router R1 donde se ejecutó Prox6.
Finalmente en el Gráfico 26 se muestran las trazas capturadas tanto en la red LAN IPv6 como en el acceso a InternetV4 que demuestran la validez del método.

```
1  fe80::16fc:eeff:fe7f:a2ff -> ff02::1      ICMPv6 Router advertisement
2  2001:1938:110:23:213:d3ff:fe78:c33d -> 2001:1938:110:23:21b:9eff:fe2d:668 TCP 1093 > 8080 [SYN] Seq=0 Win=16384 Len=0 MSS=1440
3  2001:1938:110:23:21b:9eff:fe2d:668 -> ff02::1:ff78:c33d ICMPv6 Neighbor solicitation
4  2001:1938:110:23:213:d3ff:fe78:c33d -> 2001:1938:110:23:21b:9eff:fe2d:668 ICMPv6 Neighbor advertisement
5  2001:1938:110:23:21b:9eff:fe2d:668 -> 2001:1938:110:23:213:d3ff:fe78:c33d TCP 8080 > 1093 [SYN, ACK] Seq=0 Ack=1 Win=5760 Len=0
6  2001:1938:110:23:213:d3ff:fe78:c33d -> 2001:1938:110:23:21b:9eff:fe2d:668 TCP 1093 > 8080 [ACK] Seq=1 Ack=1 Win=17280 Len=0
7  2001:1938:110:23:213:d3ff:fe78:c33d -> 2001:1938:110:23:21b:9eff:fe2d:668 HTTP GET http://sitecheck2.opera.com/?host=www.altavista.com&hdn=nubrKnkzLB7qxAS86abtMw== HTTP/1.0
8  2001:1938:110:23:21b:9eff:fe2d:668 -> 2001:1938:110:23:213:d3ff:fe78:c33d TCP 8080 > 1093 [ACK] Seq=1 Ack=498 Win=6432 Len=0
9  2001:1938:110:23:213:d3ff:fe78:c33d -> 2001:1938:110:23:21b:9eff:fe2d:668 HTTP GET http://www.altavista.com/ HTTP/1.0
```

```
10  2001:1938:110:23:21b:9eff:fe2d:668 -> 2001:1938:110:23:213:d3ff:fe78:c33d TCP 8080 > 1094 [ACK] Seq=1 Ack=539 Win=6456 Len=0
11  192.168.1.223 -> 192.168.1.1  DNS Standard query AAAA www.altavista.com
12  192.168.1.223 -> 192.168.1.1  DNS Standard query AAAA sitecheck2.opera.com
13  192.168.1.1 -> 192.168.1.223 DNS Standard query response CNAME avatw.search.a00.yahoodns.net
14  192.168.1.223 -> 192.168.1.1  DNS Standard query A www.altavista.com
15  192.168.1.1 -> 192.168.1.223 DNS Standard query response CNAME avatw.search.a00.yahoodns.net A 72.30.186.25
16  192.168.1.223 -> 72.30.186.25 TCP 43019 > 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=2203659 TSER=0 WS=6
17  1.731010 72.30.186.25 -> 192.168.1.223 TCP 80 > 43019 [SYN, ACK] Seq=0 Ack=1 Win=8712 Len=0 MSS=1452 WS=0 TSER=2203659
18  1.731031 192.168.1.223 -> 72.30.186.25 TCP 43019 > 80 [ACK] Seq=1 Ack=1 Win=5888 Len=0 TSV=2203661 TSER=3240479415
19  1.731092 192.168.1.223 -> 72.30.186.25 HTTP GET http://www.altavista.com/ HTTP/1.0
20  1.749107 72.30.186.25 -> 192.168.1.223 TCP 80 > 43019 [ACK] Seq=1 Ack=539 Win=15846 Len=0 TSV=3240479417 TSER=2203661
21  2.188310 72.30.186.25 -> 192.168.1.223 HTTP HTTP/1.0 200 OK  (text/html)
22  2.188401 192.168.1.223 -> 72.30.186.25 TCP 43019 > 80 [ACK] Seq=539 Ack=1441 Win=8768 Len=0 TSV=2203775 TSER=3240479460
23  2.188462 2001:1938:110:23:21b:9eff:fe2d:668 -> 2001:1938:110:23:213:d3ff:fe78:c33d HTTP HTTP/1.0 200 OK  (text/html)
24  2.198668 2001:1938:110:23:213:d3ff:fe78:c33d -> 2001:1938:110:23:21b:9eff:fe2d:668 TCP 1096 > 8080 [SYN] Seq=0 Win=16384 Len=0
25  2.198693 2001:1938:110:23:21b:9eff:fe2d:668 -> 2001:1938:110:23:213:d3ff:fe78:c33d TCP 8080 > 1096 [SYN, ACK] Seq=0 Ack=1 Win=5760
```

Gráfico 26: Captura de tráfico existente

Se puede observar en las tramas 1-6 el inicio de sesión TCP entre el cliente local y prox6. El requerimiento a www.altavista.com se solicita en la trama 7. El pedido de resolución de nombres (DNS) para IPv4 se hace en las tramas 11-15. La conexión a www.altavista.com desde prox6 se lleva a cabo en las tramas 16-18. El reenvío del requerimiento se hace en la trama 19. La respuesta de www.altavista.com se realiza en la trama 20 y finalmente el reenvío de la respuesta al cliente local tiene lugar en la trama 23.

4.4.3 Soporte de acceso al Backbone IPv6

Debido a que esta alternativa de transición solo permitió acceder a servidores de Internet IPv4, se realizó una mejora para poder, con la misma aplicación, dar acceso también a servidores de Internet IPv6 de manera transparente para el usuario final.

La modificación debe evaluar la respuesta a la consulta DNS. Dependiendo de la respuesta, Prox6 debe hacer el requerimiento a un servidor IPv4 o IPv6, creando un socket INET o INET6 respectivamente.

Para que los requerimientos a InternetV6 lleguen a destino, se debe conectar el router R1 a un ISP IPv6. Como los ISP de la región no proveen conectividad nativa a IPv6, se debe implementar algún otro método de transición que permita interconectar Islas IPv6 sobre océanos IPv4. Concretamente se utilizó, para el router R1, la técnica de túnel llamada “tunnel 6to4”, implementada en el apartado 4.2. El esquema que detalla el funcionamiento de la aplicación se muestra a continuación en el Gráfico 27.

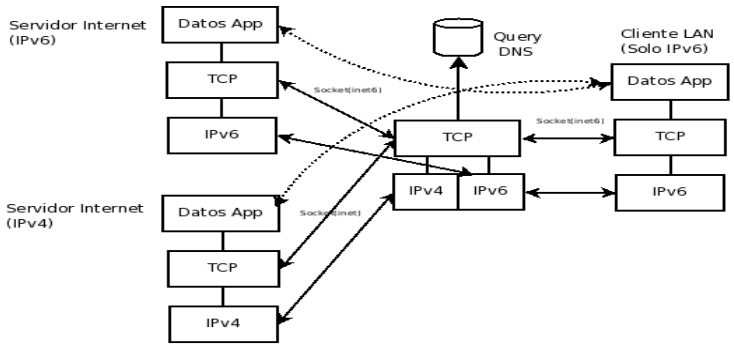

Gráfico 27: Diagrama de funcionamiento de la mejora de Prox6

Para determinar si la respuesta DNS a la dirección que necesitaba acceder el cliente final era IPv4 o IPv6, se utilizó la syscall “getaddrinfo”, que retorna una estructura con los valores obtenidos en la consulta. A partir de esos valores, se pudo determinar a que familia de protocolos pertenecía cada requerimiento.

Una vez determinado esto se realizó una segunda conexión, esta vez como cliente, hacia un servidor de Internet IPv4 o IPv6, según correspondiera. La respuesta, como en el caso anterior se envió por medio de un socket tipo INET6 al cliente local. Cabe aclarar que si la resolución de nombres retorna ambos tipos de direcciones, se da prioridad a las direcciones IPv6. A continuación se muestran las partes del código que realizan la tarea detallada en las mejoras:

```
try:
     res = getaddrinfo(host,80,AF_INET6,SOCK_STREAM)
     externo = soket(AF_INET6,SOCK_STREAM)
     if debug >= 1 print “Conectado a backbone de IPv6”
except socket.gaierror:
     try:
       res = getaddrinfo(host,80,AF_INET,SOCK_STREAM)
       externo = socket(AF_INET,SOCK_STREAM)
     except:
         interno.send(“no se resolvio el nombre”)
         interno.close()
         externo,close()
         sys.exit()
```

Gráfico 28: Mejoras en el código

La versión original de Prox6 se conecta directamente al Backbone IPv4. En esta modificación se puede ver que usando el método getaddrinfo(..AF_INET6..) intenta resolver, desde el nombre del servidor a consultar, hacia una dirección IPv6. En caso exitoso hace el requerimiento usando un socket de la familia AF_INET6 (IPv6).

En caso que el servidor no tenga una dirección IPv6, intenta resolver su nombre a una dirección IPv4 con getaddrinfo(..AF_INET..) y usando un socket de la familia AF_INET (IPv4) contacta el destino.
Para los casos que no pueda resolver ninguno de los dos tipos de direcciones, informa de un error al cliente y termina.

En el Gráfico 29 se detallan trazas capturadas tanto en la red LAN IPv6 como en el acceso a InternetV6 que demuestran la validez del método.

1 2001:1938:110:23:213:d3ff:fe78:c33d -> 2001:1938:110:23:21b:9eff:fe2d:668 TCP 1112 > 8080 [SYN] Seq=0 Win=16384 Len=0 MSS=1440
2 2001:1938:110:23:21b:9eff:fe2d:668 -> ff02::1:ff78:c33d ICMPv6 Neighbor solicitation
3 2001:1938:110:23:213:d3ff:fe78:c33d -> 2001:1938:110:23:21b:9eff:fe2d:668 ICMPv6 Neighbor advertisement
4 2001:1938:110:23:21b:9eff:fe2d:668 -> 2001:1938:110:23:213:d3ff:fe78:c33d TCP 8080 > 1112 [SYN, ACK] Seq=0 Ack=1 Win=5760 Len=0
5 2001:1938:110:23:213:d3ff:fe78:c33d -> 2001:1938:110:23:21b:9eff:fe2d:668 TCP 1112 > 8080 [ACK] Seq=1 Ack=1 Win=17280 Len=0
6 2001:1938:110:23:213:d3ff:fe78:c33d -> 2001:1938:110:23:21b:9eff:fe2d:668 HTTP GET http://ipv6.google.com/ HTTP/1.0
7 2001:1938:110:23:21b:9eff:fe2d:668 -> 2001:1938:110:23:213:d3ff:fe78:c33d TCP 8080 > 1112 [ACK] Seq=1 Ack=697 Win=6960 Len=0
8 192.168.1.223 -> 192.168.1.1 DNS Standard query AAAA ipv6.google.com
9 00:0f:66:8e:72:2d -> 01:80:c2:00:00:00 STP Conf. Root = 32768/00:0f:66:8e:72:2b Cost = 0 Port = 0x8002
10 192.168.1.1 -> 192.168.1.223 DNS Standard query response CNAME ipv6.google.com AAAA 2001:4860:a003::68
11 2001:1938:110:23:21b:9eff:fe2d:668 -> 2001:4860:a003::68 TCP 51334 > 80 [SYN] Seq=0 Win=5760 Len=0 MSS=1440 TSV=2237265 TSER=0
12 2001:4860:a003::68 -> 2001:1938:110:23:21b:9eff:fe2d:668 TCP 80 > 51334 [SYN, ACK] Seq=0 Ack=1 Win=8190 Len=0 MSS=1212
13 2001:1938:110:23:21b:9eff:fe2d:668 -> 2001:4860:a003::68 TCP 51334 > 80 [ACK] Seq=1 Ack=1 Win=5760 Len=0
14 2001:1938:110:23:21b:9eff:fe2d:668 -> 2001:4860:a003::68 HTTP GET http://ipv6.google.com/ HTTP/1.0
15 2001:4860:a003::68 -> 2001:1938:110:23:21b:9eff:fe2d:668 TCP 80 > 51334 [ACK] Seq=1 Ack=697 Win=6687 Len=0
16 2001:4860:a003::68 -> 2001:1938:110:23:21b:9eff:fe2d:668 HTTP HTTP/1.0 200 OK (text/html)
17 2001:1938:110:23:21b:9eff:fe2d:668 -> 2001:4860:a003::68 TCP 51334 > 80 [ACK] Seq=697 Ack=3051 Win=15756 Len=0
18 2001:1938:110:23:21b:9eff:fe2d:668 -> 2001:1938:110:23:213:d3ff:fe78:c33d HTTP HTTP/1.0 200 OK (text/html)
19 2001:1938:110:23:213:d3ff:fe78:c33d -> 2001:1938:110:23:21b:9eff:fe2d:668 TCP 1112 > 8080 [ACK] Seq=697 Ack=1668 Win=17280 Len=0

Gráfico 29: Captura de tráfico existente en la verificación

Se puede observar en las tramas 1-5 el inicio de sesión TCP entre el cliente local y prox6. El requerimiento a ipv6.google.com se hace en la trama 6. El pedido de resolución de nombres (DNS) para IPv6 se hace en las tramas 8-10. La conexión a ipv6.google.com desde prox6 se lleva a cabo en las tramas 11-13. El reenvío del requerimiento se hace en la trama 14. La respuesta de ipv6.google.com se muestra en la trama 16 y finalmente el reenvío de la respuesta al cliente local tiene lugar en la trama 18.

5 Pruebas de performance

Como se mencionó anteriormente, fue utilizado el Codarec6 Test Bed, según el Gráfico 30, para simular los escenarios determinados en el punto 2. Un gráfico del mismo muestra sus características.

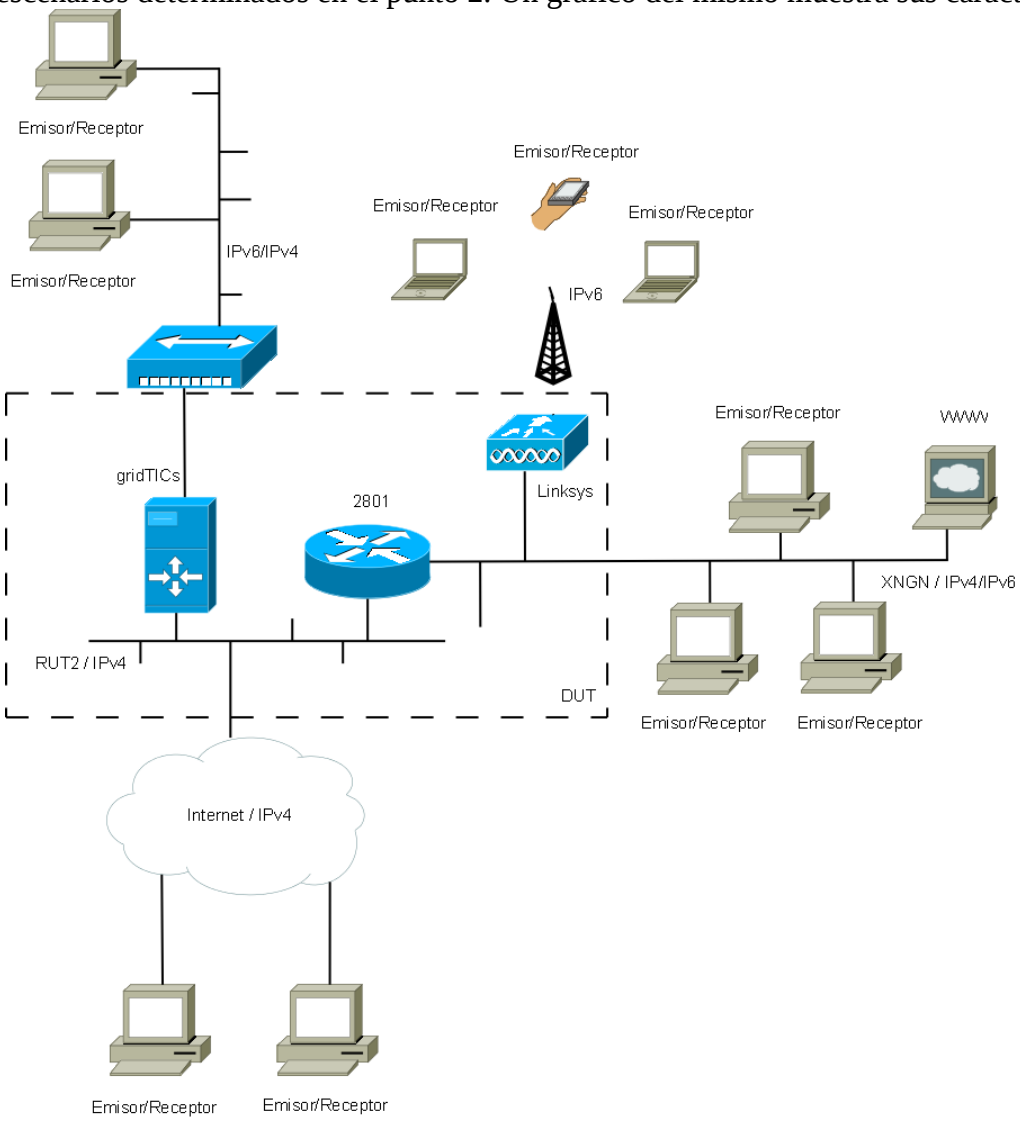

Gráfico 30: Esquema completo del Codarec6 Test Bed

5.1 Herramientas utilizadas para las pruebas

5.1.1 Netperf

Netperf es una herramienta de benchmarking utilizada para medir varios aspectos de la performance de una red [26]. Está diseñado utilizando el modelo clásico cliente-servidor. Existen 2 ejecutables, netperf y netserver. En general, netserver se ejecuta en el sistema remoto y netperf localmente. Cuando se ejecuta netperf, se establece una conexión de control con el sistema remoto, esta conexión es utilizada

para transmitir y recibir información y configuraciones de los tests. Una vez que la conexión está establecida y se ha pasado toda la información de control necesaria, se abre una conexión de datos para las mediciones.

Descripción de tests:

TCP_STREAM
Transfiere una cierta cantidad de datos desde el host que corre netperf hacia el host que corre netserver y calcula el throughput. El tiempo para establecer la conexión no es considerado.

TCP_MAERTS
Transfiere una cierta cantidad de datos desde el host que corre netserver hacia el host que corre netperf y calcula el throughput. El tiempo para establecer la conexión no es considerado.

UDP_STREAM
Transfiere una cierta cantidad de datos desde el host que corre netperf hacia el host que corre netserver y calcula el throughput.

TCP_RR
Es un test de requerimiento/respuesta. La tasa de transacciones es el número de transacciones completas sobre el tiempo que tomó realizar las transacciones.

TCP_CC

Mide que tán rápido los sistemas pueden abrir y cerrar conexiones, de manera sincrónica (una a la vez).

TCP_CRR
Mide el tiempo para establecer una conexión, realizar una transacción requerimiento/respuesta y cerrar la conexión.

5.1.2 Ping y Ping6

La utilidad ping comprueba el estado de la conexión con uno o varios equipos remotos por medio de los paquetes de ECHO_REQUEST y ECHO_RESPONSE, definidos en el protocolo ICMP [27]. Principalmente se utiliza para determinar si un host específico es accesible en una red.

También es usado frecuentemente, como en nuestro caso, para para medir la latencia o tiempo que tardan en comunicarse dos puntos (RTT)[1]. Adicionalmente brinda información de máximos, mínimos y promedios del RTT, indicando también si hay perdidas de paquetes ICMP. Fue utilizado con distintos modificadores para cambiar el comportamiento de las mediciones de RTT.

El modificador “-f” hace un envío masivo de paquetes ECHO_REQUEST. Envía paquetes a la máxima velocidad que trabaja la interfaz local. Con cada ECHO_REQUEST enviado se escribe un ".", mientras que por cada ECHO_RESPONSE recibido se escribe un backspace. Esto proporciona una muestra rápida de cuántos paquetes se están perdiendo.

El modificador “-s tamaño” especifica la cantidad de bytes de datos que se enviarán. La cantidad por defecto es 56 Bytes, que con el agregado del encabezado ICMP se convierten en 64 Bytes.
El modificador “'-c cantidad” le indica a ping la cantidad de paquetes que enviará la utilidad ping antes de terminar.

La utilidad ping6 se utiliza para los mismos ensayos que ping, con la excepción que utiliza el stack de protocolos IPv6, al generar un paquete ICMPv6. ICMPv6 es el protocolo de mensajes de control y error usado por IPv6 y la familia de protocolos IPv6.

Tipos de ICMPv6

Los mensajes ICMPv6 son clasificados de acuerdo a los campos tipo y código presentes en el encabezado ICMPv6. Los siguientes tipos son definidos:

Núm.	**Abrev.**	**Descripción**
1	unreach	Destino inalcanzable
2	toobig	Paquete muy grande
3	timex	Tiempo excedido
4	paramprob	Encabezado IPv6 inválido
128	echoreq	Requerimiento de eco
129	echorep	Respuesta a eco
130	groupqry	Petición de membresía de grupo
130	listqry	Petición de oyente multicast
131	grouprep	Reporte de membresía de grupo
131	listenrep	Reporte de oyente multicast
132	groupterm	Finalización de membresía de grupo
132	listendone	Finalización de oyente multicast
133	routersol	Router solicitation
134	routeradv	Router advertisement
135	neighbrsol	Neighbor solicitation
136	neighbradv	Neighbor advertisement
137	redir	Existe ruta más corta
138	routrrenum	Renunmeración de ruta
139	fqdnreq	Pedido FQDN
139	niqry	Pedido de información de nodo
139	wrureq	Requerimiento Who-are-you
140	fqdnrep	Respuesta FQDN
140	nirep	Respuesta de información de nodo
140	wrurep	Respuesta Who-are-you
200	mtraceresp	Respuesta mtrace
201	mtrace	Mensajes mtrace

Los siguientes códigos también son definidos:

Num	**Abbrev.**	**Type**	**Description**
0	noroute-unr	unreach	Sin ruta hacia el destino
1	admin-unr	unreach	Prohibido administrativamente
2	beyond-unr	unreach	Fuera de alcance de la dirección fuente

2	notnbr-unr	unreach	(obsoleto)
3	addr-unr	unreach	Dirección inalcanzable
4	port-unr	unreach	Puerto inalcanzable
0	transit	timex	Tiempo excedido en tránsito
1	reassemb	timex	Tiempo excedido en reensamble
0	badhead	paramprob	Campo de encabezado erróneo
1	nxthdr	paramprob	Siguiente encabezado no reconocido
2		redir	Opción no reconocida
0	redironlink	redir	Redirección a nodo en el mismo enlace
1	redirrouter	redir	Redirección a mejor router

5.2 Escenario A:

5.2.1 Dual Stack:

Se configuraron los routers R1 y R2, para vincular 3 redes internas ubicadas en el CODAREC6 test bed, de acuerdo a la topología del Gráfico 31.

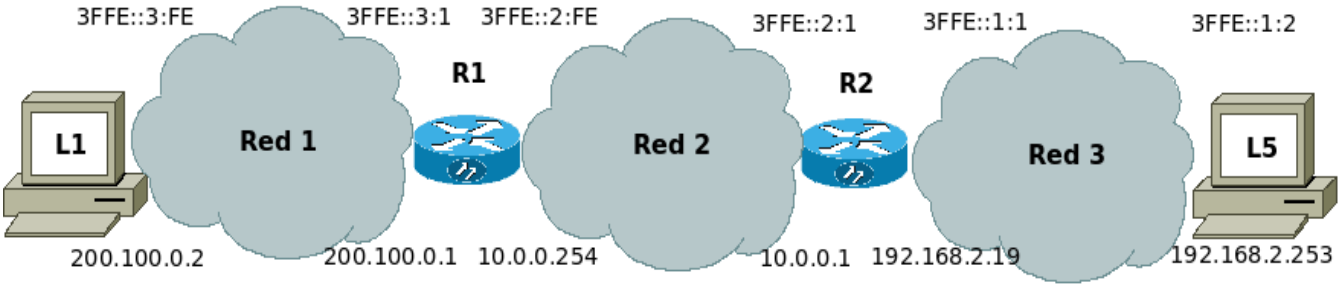

Gráfico 31: Topología de red

Todos los nodos se implementaron con sistema operativo GNU/Linux. Se configuraron en forma idéntica con soporte para doble stacks IPv4 e IPv6. Las direcciones se asignaron manualmente, tomando los valores que indica el Gráfico 31. En R1 y R2 adicionalmente se habilitó el encaminamiento para ambos protocolos.

Para estudiar el comportamiento se realizó un ensayo de tráfico, enviando paquetes ICMP tan rápido como la interfaz de red pueda despacharlos, sin esperar la respuesta correspondiente. Con esta medida se exigió a los routers, aumentando sustancialmente el tráfico. Se tomaron los tiempos de RTT y se repitió el ensayo variando la carga útil entre 54 bytes y 10000 bytes. La cantidad con las que se obtuvo el promedio fue de 5000 paquetes. En la Tabla 1 podemos ver el resultado de los ensayos realizados.

Carga útil	IPv4(miliseg)			IPv6 (miliseg)		
	Mínimo	Promedio	Máximo	Mínimo	Promedio	Máximo
56 bytes	0.963	1.165	8.113	1.071	1.193	2.836
500 bytes	3.606	3.823	4.612	3.740	3.947	5.318
2000 bytes	10.423	10.770	11.894	10.392	10.974	14.080
5000 bytes	15.671	16.730	24.836	15.761	17.735	30.079
8000 bytes	23.228	1146.369	1912.195	26.691	846.590	1571.177
10000bytes *	48.145	2777.031	4402.791	55.085	2598.171	4122.365

Tabla 1: RTT (Round-Trip Time)

* Se pierden paquetes tanto para IPV4 como para IPV6. El porcentaje fue significativamente superior para el stack IPV4.

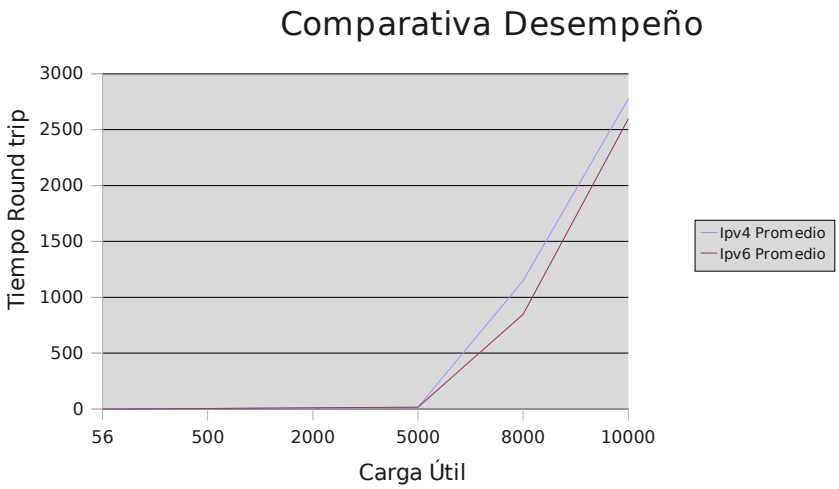

Gráfico 32: Comparación desempeño

El Gráfico 32 demostró que para tamaños de carga útil bajos, el tiempo de Roud-Trip es menor para IPv4. Se justifica por que prevalece el aumento de tamaño del encabezado IPv6 sobre la optimización en los algoritmos de ruteo. A medida que el tamaño de carga útil aumenta, los routers IPv4 se ven más exigidos. Esta demora prevalece sobre un aumento del encabezado IPv6.

Comportamiento frente a variaciones de MTU

Se comprobó el comportamiento de ambos protocolos cuando el MTU del camino era menor que el tamaño del datagrama enviado.

Utilizando la misma topología del Gráfico 31, se inundó con paquetes ICMP para la versión IPv4 con carga útil de 1472 bytes. Como las tres redes subyacentes eran ethernet, los paquetes ICMP no sufrieron fragmentación, como era de esperar.

Posteriormente se disminuyó el MTU en las interfaces que vinculan R1 y R2. Dejando su valor en 1280 [28]. Se tomaron los tiempos Round-Trip. La tabla 2 muestra la comparación, y un factor de degradación de performance. La cantidad con las que se obtuvo el promedio fue de 5000 paquetes.

Round-Trip(ms)	MTU 1500	MTU 1280	Degradación %
Mínimo	9.376	9.907	5.66
Promedio	9.606	13.566	41.22
Máximo	10.694	28.393	165.5

Tabla 2: Degradación por fragmentación

Se realizó el mismo ensayo sobre el stack IPv6 para ver la degradación en el Round-Trip cuando el MTU del canal es menor al tamaño de los paquetes enviados.

Se inundó de paquetes ICMPv6 con una carga útil de 1440 bytes. Como era de esperar los paquetes ICMPv6 no sufrieron fragmentación en todo el camino. Se colocaron los tiempos involucrados en una

tabla. Se disminuyó el MTU en las interfaces que vinculan R1 y R2. Dejando su valor en 1280.

Se inundó de paquetes ICMPv6, con una carga útil de 1440 bytes. Se logró verificar el mecanismo por el cual el origen fragmenta [5], como se observa en el Gráfico 33.

```
17:57:20.237204 IP6 3ffe::1:2 > 3ffe::3:fe: ICMP6, echo request, seq 1, length 1448
17:57:20.240848 IP6 3ffe::1:1 > ff02::1:ff01:2: ICMP6, neighbor solicitation, who has 3ffe::1:2, length 32
17:57:20.241282 IP6 3ffe::1:2 > 3ffe::1:1: ICMP6, neighbor advertisement, tgt is 3ffe::1:2, length 32
17:57:20.241342 IP6 3ffe::1:1 > 3ffe::1:2: ICMP6, packet too big, mtu 1280, length 1240
17:57:21.237980 IP6 3ffe::1:2 > 3ffe::3:fe: frag (0|1232) ICMP6, echo request, seq 2, length 1232
17:57:21.238054 IP6 3ffe::1:2 > 3ffe::3:fe: frag (1232|216)
17:57:22.237829 IP6 3ffe::1:2 > 3ffe::3:fe: frag (0|1232) ICMP6, echo request, seq 3, length 1232
17:57:22.237904 IP6 3ffe::1:2 > 3ffe::3:fe: frag (1232|216)
17:57:22.244294 IP6 3ffe::3:fe > 3ffe::1:2: frag (0|1232) ICMP6, echo reply, seq 3, length 1232
17:57:22.244554 IP6 3ffe::3:fe > 3ffe::1:2: frag (1232|216)
```

Gráfico 33: Fragmentación en origen

La tabla 3 muestra los tiempos Round-Trip Time y un factor de degradación de performance. La cantidad con las que se obtuvo el promedio fue de 5000 paquetes.

Round-Trip(ms)	MTU 1500	MTU 1280	Degradación%
Mínimo	9.286	9.395	1.17
Promedio	9.527	9.571	0.46
Máximo	10.782	17.936	66.35

Tabla 3

De la tabla 3 pudimos concluir que ante un MTU del canal menor que el tamaño del paquete, la degradación para el protocolo IPv6 es casi inexistente a diferencia de lo experimentado para IPv4.

Comparación ruteo IPv4 e IPv6

Se comparó el comportamiento del protocolo IPv6 para iguales estímulos. Se muestra en el Gráfico 34 el resultado del envío de paquetes ICMPv6 tipo ECHO_REQUEST desde la red 1 a la 3 y la respuesta de paquetes ICMPv6 ECHO_RESPONSE desde la red 3.

```
14:30:05.581351 IP6 3ffe::1:2 > 3ffe::3:fe: ICMP6, echo request, seq 11, length 64
14:30:05.582360 IP6 3ffe::3:fe > 3ffe::1:2: ICMP6, echo reply, seq 11, length 64
14:30:06.582001 IP6 3ffe::1:2 > 3ffe::3:fe: ICMP6, echo request, seq 12, length 64
14:30:06.583258 IP6 3ffe::3:fe > 3ffe::1:2: ICMP6, echo reply, seq 12, length 64
14:30:07.582847 IP6 3ffe::1:2 > 3ffe::3:fe: ICMP6, echo request, seq 13, length 64
14:30:07.584098 IP6 3ffe::3:fe > 3ffe::1:2: ICMP6, echo reply, seq 13, length 64
```

Gráfico 34: Envío de paquetes ICMPv6

Se aumentó la carga útil del paquete ICMPv6 a 2000 bytes. Se representa el tráfico generado en el Gráfico 35.

```
14:31:01.296304 IP6 3ffe::1:2 > 3ffe::3:fe: frag (0|1448) ICMP6, echo request, seq 10, length 1448
14:31:01.296354 IP6 3ffe::1:2 > 3ffe::3:fe: frag (1448|560)
14:31:01.296371 IP6 3ffe::3:fe > 3ffe::1:2: frag (0|1448) ICMP6, echo reply, seq 10, length 1448
14:31:01.296388 IP6 3ffe::3:fe > 3ffe::1:2: frag (1448|560)
14:31:02.283468 IP6 3ffe::1:2 > 3ffe::3:fe: frag (0|1448) ICMP6, echo request, seq 11, length 1448
14:31:02.283943 IP6 3ffe::1:2 > 3ffe::3:fe: frag (1448|560)
14:31:02.290691 IP6 3ffe::3:fe > 3ffe::1:2: frag (0|1448) ICMP6, echo reply, seq 11, length 1448
```

```
14:31:02.291116 IP6 3ffe::3:fe > 3ffe::1:2: frag (1448|560)
14:31:03.284336 IP6 3ffe::1:2 > 3ffe::3:fe: frag (0|1448) ICMP6, echo request, seq 12, length 1448
14:31:03.284783 IP6 3ffe::1:2 > 3ffe::3:fe: frag (1448|560)
14:31:03.291119 IP6 3ffe::3:fe > 3ffe::1:2: frag (0|1448) ICMP6, echo reply, seq 12, length 1448
14:31:03.291552 IP6 3ffe::3:fe > 3ffe::1:2: frag (1448|560)
```

Gráfico 35: Paquetes ICMPv6 con carga útil de 2Kbytes

Se verificó que los paquetes IPv6 se enviaron fragmentados desde el host origen, constatándose que en cada cabecera se agregó un Fragmentation header .

Se concluye, al comparar los timestamps de los Gráficos 34 y 35, que hubo más retardo utilizando la versión 6 del protocolo IP que la 4, a diferencia de lo que se suponía que ocurriría.

5.2.2 Túnel Manual

Para implementar este método, cada host que desea acceder a la red IPv6 debe crear un túnel al router del Test Bed, que es el que posee conexión al backone IPv6, como se indica en el gráfico 36. Ésta es una tarea tediosa ya que se debe configurar manualmente cada túnel del lado del router destino (r5-gridtics) y del lado del host. Es además, un método muy poco flexible frente a cambios en al red.

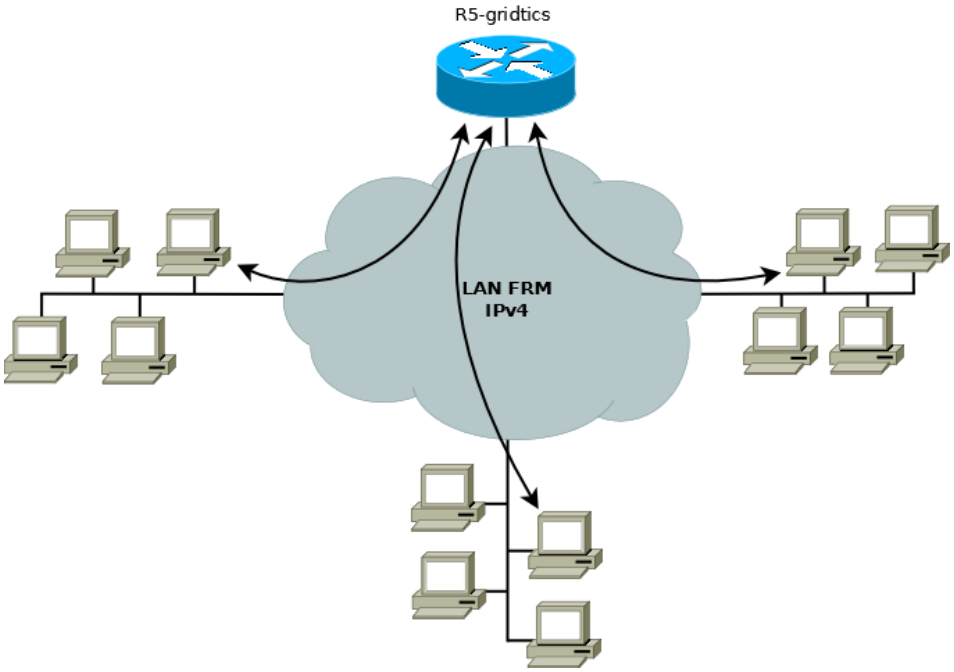

Gráfico 36: Escenario simplificado

En la gráfica 37 se observan las mediciones realizadas para túneles manuales en el ambiente de pruebas detallado anteriormente.

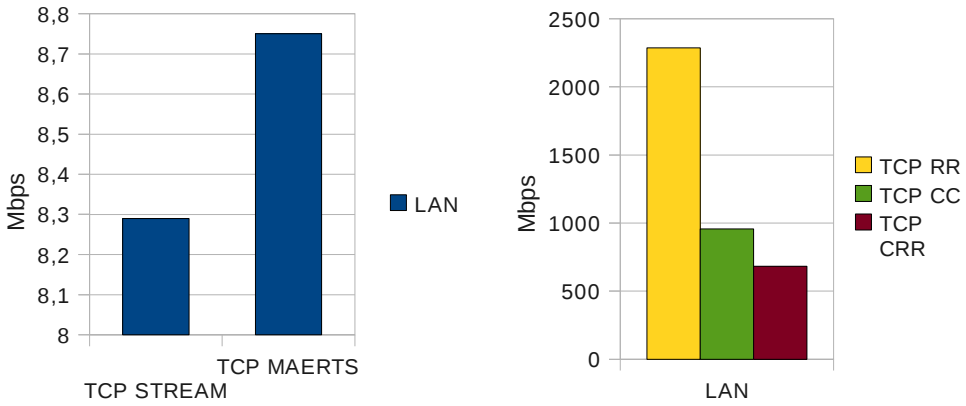

Gráfico 37: Ensayos

5.2.3 Tunnel Broker y Tunnel 6to4:

Se probaron 2 métodos de transición, tunnel broker y 6to4. Para el primero de ellos se utilizó el servicio de Hurricane Electric [29].
Descripción de los equipos:

Equipo	transición	r5-gridtics	Linux	prueba
Tipo	Desktop	Router Cisco	Servidor	Laptop
SO	CentOS 5	IOS 12.4	Debian 4.0	Kubuntu 10.04

Tabla 4: Equipamiento

Para las mediciones se utilizó el escenario según el Gráfico 38:

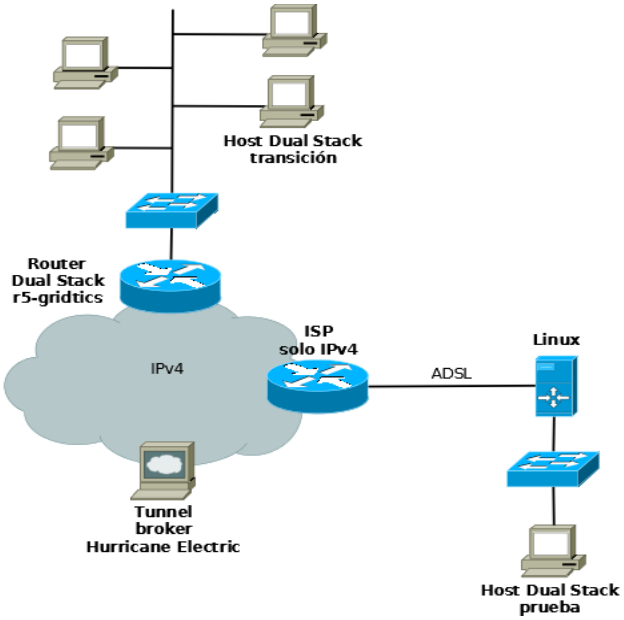

Gráfico 38: Esquemas de Túnel

Test TCP. Comparación Tunnel broker vs 6to4:

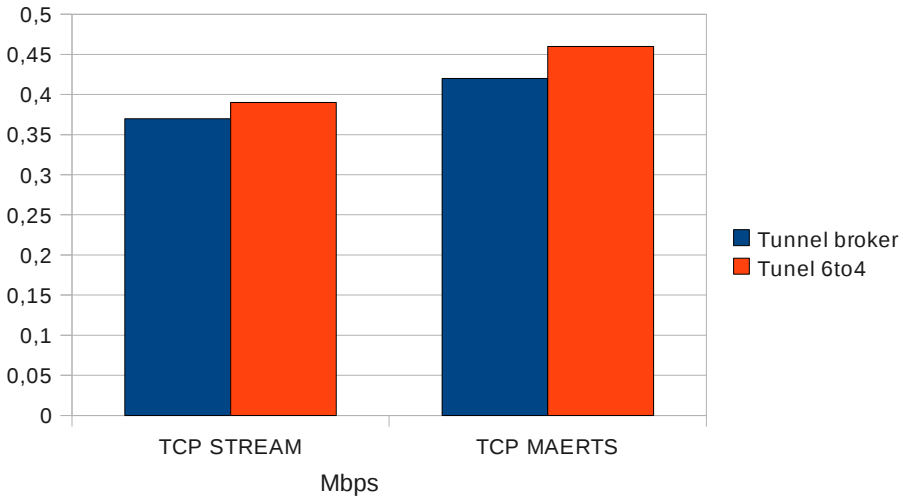

Gráfico 39: Comparativo Tráfico

Tests requerimiento/respuesta. Comparación Tunnel broker vs 6to4:

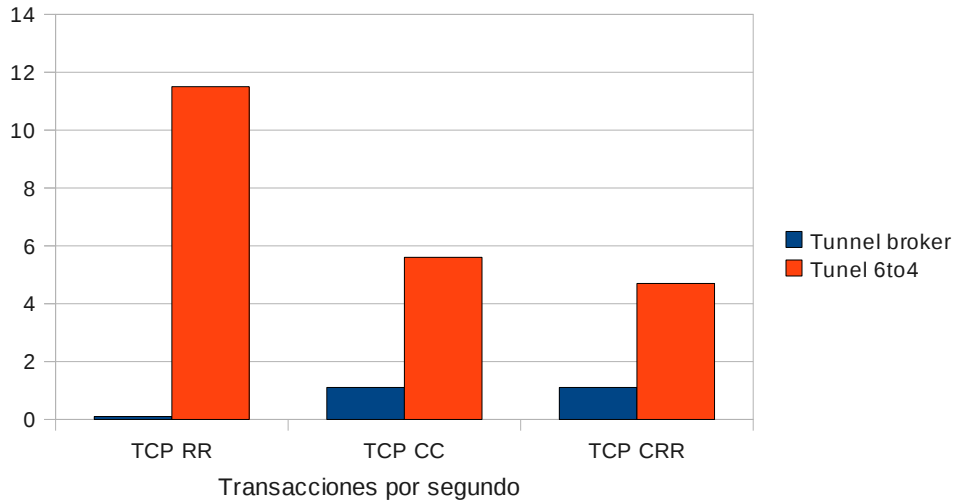

Gráfico 40: Comparativo Tráfico

5.2.4 Traducción ALG

Este método si bien es aplicable, no es recomendable utilizarlo a no ser que no existan alternativas. Por esta razón no se realizaron pruebas. Como primer inconveniente genera más overhead que el resto de los métodos que se ensayaron en el escenario. Por otro lado, se debe realizar una aplicación para soportar cada una de las utilidades a ensayar, como por ejemplo para ping y ping6 se debe hacer un ALG que traduzca ICMP a ICMPv6 y viceversa, para la utilidad Netperf se deben desarrollar al menos

dos ALGs que tengan soporte para TCP y UDP respectivamente.

5.3 Escenario B

5.3.1 Dual Stack

En el gráfico 41 se observa el escenario ensayado. Es posible utilizar este método de transición gracias a un patch para OpenVPN [30] que permite utilizar endpoints IPv6.

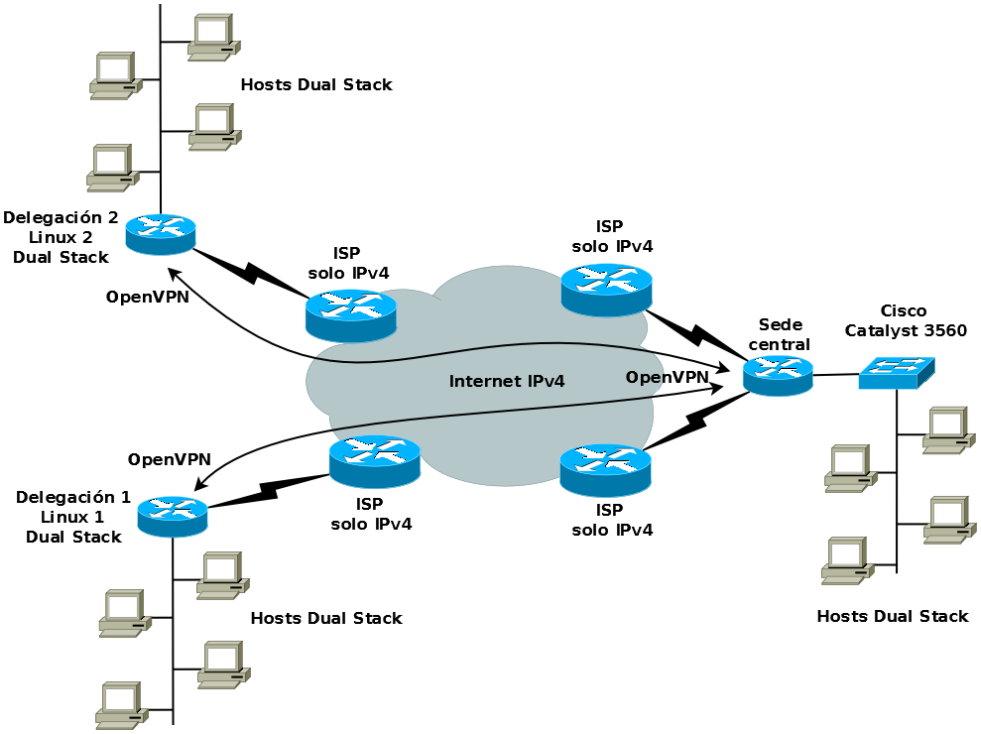

Gráfico 41: Escenario INV

Se realizaron mediciones STREAM y MAERTS entre las delegaciones y sede central y entre delegaciones. Los resultados se aprecian en el gráfico 42.

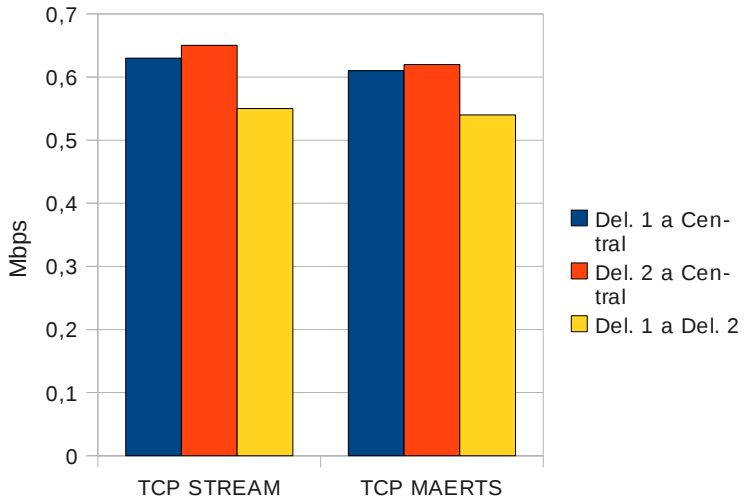

Gráfico 42: Comparativo Mbps

Se debe aclarar que si bien el método Dual Stack se ensayó de la misma manera que en el apartado 5.1.1, no se obtuvieron los mismos valores que en esa oportunidad. Esto es debido a que ahora el enlace que vincula los distintos nodos está encapsulado sobre tramas UDP/IPv4, obtenidas al usar OpenVPN.

5.3.2 Túnel Manual, Tunnel 6TO4 y Tunnel Broker

Para este escenario, se implementaron estos tres tipos de túneles y se ensayaron los enlaces entre las delegaciones y la Sede Central.

	Tunnel broker	Tunel ISATAP	Tunel 6to4	Tunel Manual
TCP STREAM	0,57	0,58	0,63	0,63
TCP MAERTS	0,70	0,80	0,84	0,84
TCP RR	0,10	0,30	11,50	11,50
TCP CC	1,10	2,10	5,60	5,60
TCP CRR	1,10	2,00	4,70	4,70

Tabla 5: Valores para los distintos túneles

En la tabla 5 se observan los valores medidos, mientras están graficados a continuación.

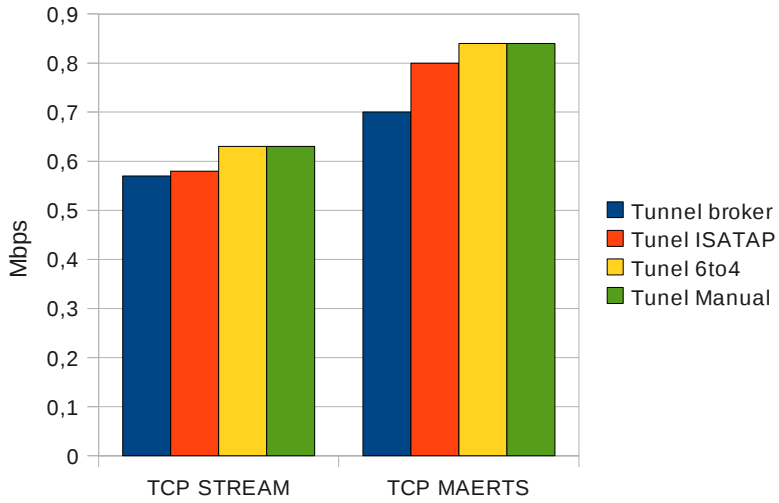

Gráfico 43: Comparativo Mbps

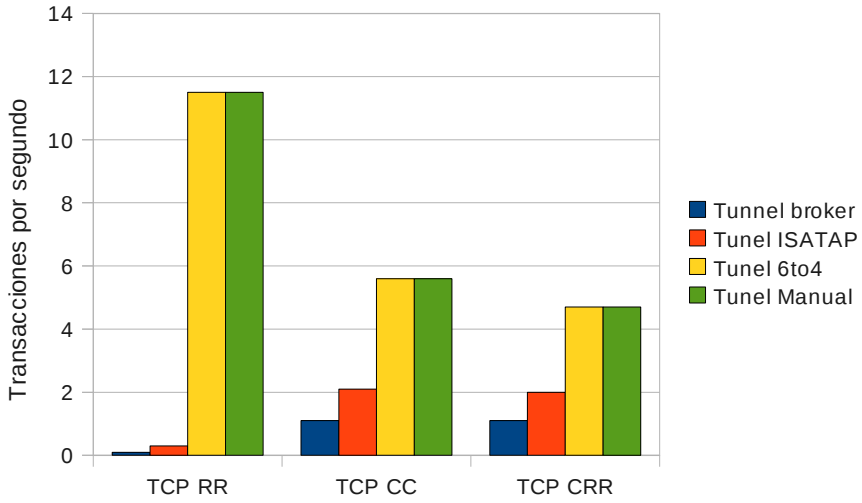

Gráfico 44: Comparativo Transacciones por Segundo

5.3.3 Traducción ALG

Este método si bien es aplicable, no es recomendable utilizarlo a no ser que no existan alternativas. Se aplican la mismas consideraciones que en el apartado 5.2.4 .

5.4 Escenario C:

El escenario ensayado es el que se muestra en el Gráfico 45. El host corre Windows Vista y tiene una conexión ADSL de 1Mbps a un proveedor solo IPv4.

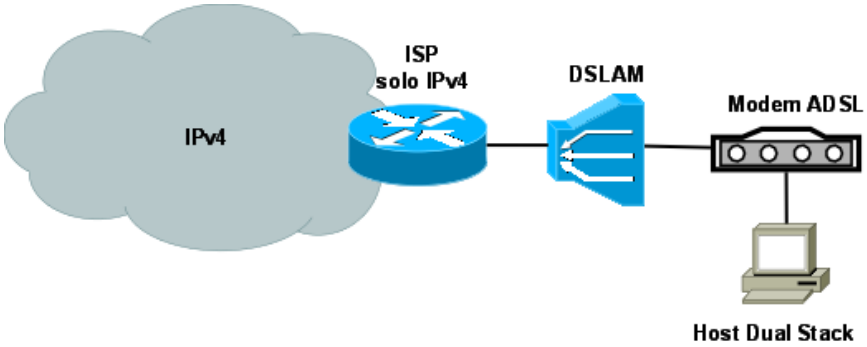

Gráfico 45: Esquema enlace hogareño

5.4.1 Dual Stack

El método Dual Stack, para la red local, se comporta exactamente igual a los ensayado en el apartado 5.2.1, por lo que no se detallan los resultados obtenidos.
No es aplicable esta técnica hacia el acceso a internet, por no contar el Modem del ISP soporte para IPv6.

5.4.2 Túnel Manual, Tunnel 6to4, ISATAP

Los túneles manual, 6to4 e ISATAP no aplican en este escenario debido a que las conexiones hogareñas poseen direcciones IPv4 dinámicas. El cambio de la dirección IPv4 provocaría, en el caso del túnel manual, tener que reconfigurar el túnel en ambos extremos y en el caso del túnel 6to4 e ISATAP, la modificación de la dirección IPv6 global.

5.4.3 Tunnel Broker

Se ensayó el escenario indicado en el gráfico 46.

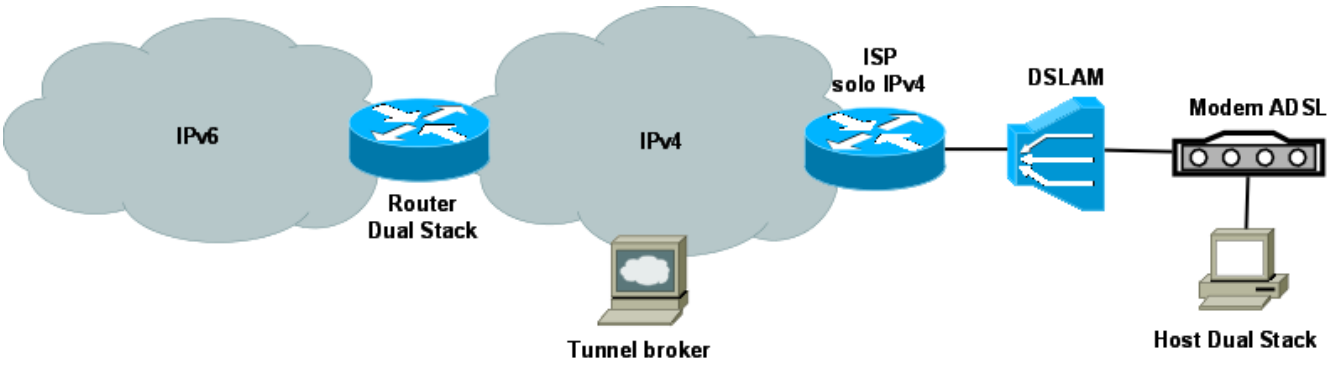

Gráfico 46: Esquema Enlace Hogareño con Tunnel Broker

En el gráfico 47 se observa una comparativa de los resultados obtenidos mediante los ensayos STREAM y MAERTS a través de los distintos Tunnel Broker`s.

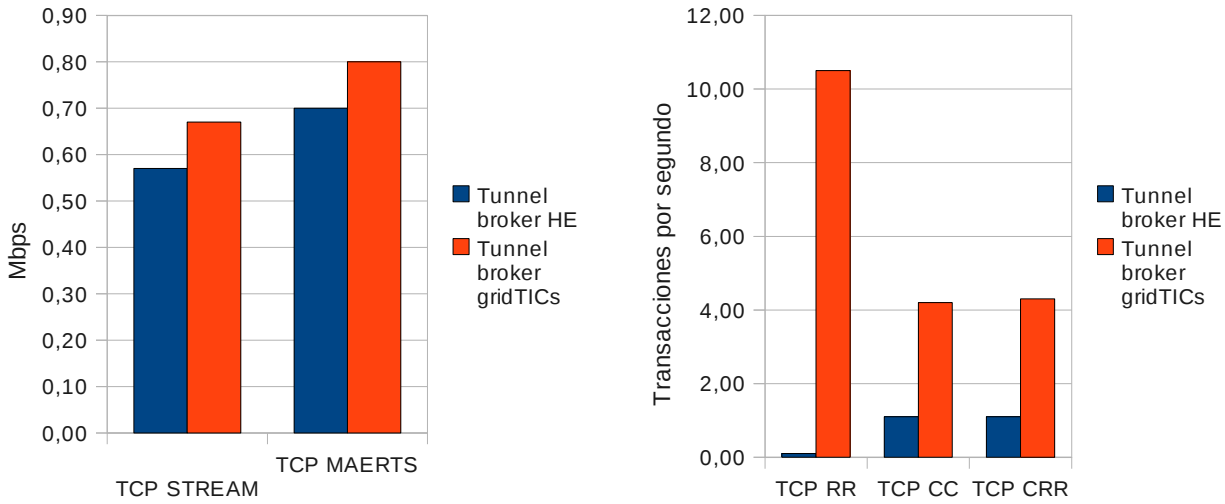

Gráfico 47: Datos obtenidos en el ensayo

5.4.4 Traducción BIAS/BIS

Windows Vista no soporta este método de transición, debido a que no se cuenta con el código fuente del sistema Operativo, ni de los Drivers de Red.

5.4.5 Traducción ALG

Este método si bien es aplicable, no es recomendable utilizarlo a no ser que no existan alternativas. Se aplican la mismas consideraciones que en el apartado 5.2.4 .

5.5 Escenario D:

Se muestra en el gráfico 48 el escenario creado para evaluar las alternativas de transición disponibles.

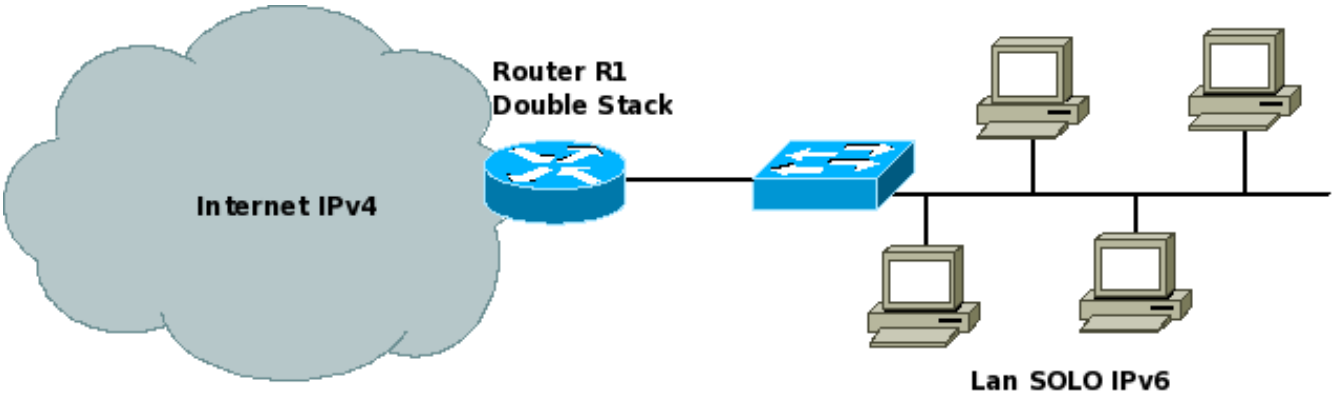

Gráfico 48: Escenario creado para evaluar métodos de transición

La topología propuesta está compuesta por hosts que constituyen una Lan IPv6 nativa. El router R1 se conecta a un ISP para tener acceso a Internet.

El objetivo deseado es permitir que los hosts de la LAN puedan acceder a servidores y servicios disponibles en InternetV4 sin que sea necesario modificar los clientes, ya sea instalando pila dual, configurando túneles o haciendo traslación de protocolos.

5.5.1 Dual Stack

Este mecanismo no se puede implementar porque el escenario es con hosts SOLO IPv6 en la red local.

5.5.2 Túnel Manual, Tunnel 6to4, Tunnel Broker, Tunnel ISATAP

Ninguno de estos métodos son aplicables debido que ningún host de la red local cuenta con Dual Stack para poder hacer algún tipo de túnel, por eso no hay más alternativas que utilizar métodos de Traslación de Protocolos.

5.5.3 Traslación de Protocolos

Para alcanzar el objetivo deseado se debe implementar alguna de las técnicas enumeradas en el apartado 2.3 pues la comunicación es exclusivamente entre hosts solo IPv4 y hosts solo IPv6, descartándose por esta causa las técnicas de pila dual o cualquier método de “tunneling”.

A continuación se analizan las alternativas existentes para realizar Traslación de protocolos:

- La aplicación de SIIT y NAT-PT también es descartada, siguiendo la recomendación de IETF, debido los problemas normales de NAT y la posible pérdida de información de encabezado [31].
- Si bien BIS o BIA pueden ser implementados, es necesario modificar los sistemas operativos de todos los hosts de la red Lan. Se encontrarán problemas para los sistemas operativos que no dispongan del código fuente.
- La alternativa de un TRT es factible y bastante usada, encontrando inclusive alguna implementación libre disponible para su ensayo. Como inconveniente requiere un DNS Proxy especialmente configurado para funcionar correctamente, lo cual no está previsto en el Escenario planteado.
- La implementación de un ALG también es viable, si no se tiene en cuenta la disminución de performance por hacer toda la conversión en la capa de aplicación.

6 Conclusiones de cada método de Transición.

6.1 Escenario A:

De lo ensayado en el punto anterior, se puede concluir que todos los hosts que están en la red de la FRM deben tener habilitado el stack IPv6 antes de poder implementar servicios sobre ese protocolo, pues de otra forma, al resolver una dirección IPv6 el servicio DNS, el host no podría interpretarla. Por ello se descarta la utilización de sistemas operativos como Windows 98 o anteriores.

Los proveedores de acceso a Internet (ISP) no ofrecen conectividad nativa a Ipv6 en la actualidad. Como los hosts internos poseen Dual Stack, se prefiere el método de Túnel al de traslación, por ser el último menos escalable y más costoso de implementar.

De las variadas alternativas de túneles, se enumeran los pros y contras de cada uno de ellos:
Para el túnel manual, es necesario que una entidad externa que ya tenga un prefijo global de IPv6, nos ceda una subred de éste. La configuración del túnel se hará solo una vez en cada extremo del túnel. Por otro lado, todo el tráfico generado en la FRM pasará por el vínculo de la entidad externa.

Generalmente estas entidades acceden a configurar túneles manuales solo con fines de pruebas, no para entornos de producción.

Debido a que la dirección pública de IPv4 que ofrece el proveedor es estática, se puede optar por la configuración de un Túnel 6to4 y no es necesario ninguna entidad externa para hacerlo. Un problema que se puede presentar de manera esporádica con este tipo de túnel está referido a los Routers Relay que vinculan las redes IPv4 e IPv6, pues tienen una dirección IPv4 (192.88.99.1) que está asociada siempre al router más cercano. En caso de salir de operación ese último, la FRM perderá todo acceso a la IPv6 nativa.

Para el método Tunnel Broker no se presenta el problema del Router Relay, pero la desventaja fundamental es el mayor retardo que se puede obtener y el SPOF, que de salir de operación el proveedor del servicio, la FRM no tendrá conectividad alguna hacia ninguna red IPV6.

Para el método Teredo la limitación fundamental es que está implementado en sistemas operativos de la familia Microsoft, lo cual es una limitación en la FRM, ya que los routers no son de ese tipo.

Quizás la alternativa más adecuada para un escenario como el “A” es tener más de un tipo de túnel configurado y operativo, como un Tunnel Broker y un Tunnel 6to4.

6.2 Escenario B:

De lo ensayado en el punto anterior, se puede concluir que todos los hosts que están en la red de INV, incluyendo las delegaciones, deben tener habilitado el stack IPv6 antes de poder implementar servicios sobre ese protocolo, pues de otra forma, al resolver una dirección IPv6 el servicio DNS, el host no podría interpretarla. Esto excluye hosts con sistemas operativos Windows 98 o anteriores.

Para lograr la conectividad IPv6 de las delegaciones se puede configurar nativamente IPv6 sobre la VPN instalada, sin necesidad de ningún método de transición.

Con respecto a los proveedores de acceso a Internet (ISP) ninguno ofrece conectividad a IPv6. Como los hosts internos poseen Dual Stack, se prefiere el método de Túnel al de traslación de protocolos, por ser el último menos escalable y más costoso de implementar.

De las variadas alternativas de túneles, se enumeran los pros y contras de cada uno de ellos:
Para el túnel manual, es necesario que una entidad externa que ya tenga un prefijo global de IPv6, nos ceda una subred de ese prefijo. La configuración del túnel se hará solo una vez en cada extremo del túnel. Por otro lado, todo el tráfico generado en el INV pasará por el vínculo de la entidad externa, lo que compromete el esquema de seguridad de la red.

Generalmente estas entidades acceden a configurar túneles manuales solo con fines de pruebas, por lo que no es conveniente este método para el escenario “B”.

Debido a que la dirección pública de IPv4 que ofrecen los proveedores es estática, se puede optar por la configuración de un Túnel 6to4 y no es necesario ninguna entidad externa para hacerlo. Un problema que se puede presentar de manera esporádica está referido a los Routers Relay que vinculan las redes IPv4 e IPv6, pues tienen una dirección IPv4 (192.88.99.1) que está asociada siempre al router más cercano. En caso de salir de operación el último, el INV perderá todo acceso a la IPv6 nativa. Como los dos ISP con los que se cuenta son distintos, el Router Relay es probable que también lo sea, por lo que, si se configura un Tunnel 6to4 sobre cada enlace, se puede minimizar este problema.

Para el método Tunnel Broker, la desventaja fundamental es el mayor retardo que se puede obtener y el SPOF, que de salir de operación el proveedor del servicio, el INV no tendrá conectividad alguna hacia ninguna red IPv6.

Para el método Teredo la limitación fundamental es que está implementado nativamente solo para sistemas operativos de la familia Microsoft. Para el INV, cuyos routers son GNU/Linux, se deben instalar aplicaciones adicionales para poder dar soporte a este método.

Quizás la alternativa más adecuada para el presente escenario es tener dos túneles del tipo 6to4 (uno porcada ISP), y como enlace secundario un Tunnel Broker hacia un proveedor regional, mientras que los proveedores de IPS no tengan soporte nativo a IPv6.

6.3 Escenario C:

Para un escenario de este tipo se puede utilizar, en principio, cualquiera de los métodos ensayados anteriormente de túneles, con algunas restricciones.

La mayoría de los proveedores de acceso a Internet (ISP) brindan una dirección IPv4 dinámica, que cambia cada vez que se establece el enlace a Internet, lo que dificulta un tanto la utilización del método de Túnel Configurado y Túnel 6to4.

El problema que se presenta con el Túnel Configurado es a la hora de su definición, pues hay que configurar en cada extremo la dirección IPv4 local y remota. Si una de ellas es dinámica se deben modificar o redefinir los dos extremos del túnel cada vez que se reactiva el enlace a internet.

Por su parte, recordemos el formato de dirección IPv6 en un túnel 6to4 es 2002:ADDR_IPv4::/48 y debido a que la dirección IPv6 de la red depende de la dirección global IPv4, la primera cambiará cada vez que se conecte a a Internet, debiendo modificar los registros A6 o AAAA en los servidores DNS para que se puedan acceder a los servicios.

Utilizar un servicio Tunnel Broker sería lo más adecuado para una conexión con IPv4 dinámica, ya que permite al usuario poseer una IPv6 estática. Esto evita el inconveniente de actualizar registros A6 o AAAA cada vez que se modifica la IPv4, modificando solo la configuración del túnel. Como se puede observar en los gráficos comparativos, la performance frente a un 6to4 es inferior, sin embargo, esto depende de la ubicación física del proveedor de tunnel broker, pudiendo mejorar el rendimiento si se utilizara uno local.

El túnel ISATAP no se implentó debido a que está orientado a conectividad intra-sitio, además de generar la dirección IPv6 a partir de la dirección IPv4. Adicionalmente limita el sistema operativo del Host a Windows XP o superior.

Con respecto a la Traslación de protocolos es aplicable, pero es el más costoso y menos escalable de los métodos.

De lo anterior se desprende que para accesos a Internet con direcciones IPv4 dinámicas es más conveniente usar un Tunnel Broker hasta tanto los proveedores IPS provean IPv6 nativa.

6.4 Escenario D:

Como se describió anteriormente, el escenario “D” está formado por una red de hosts SOLO IPv6. De acuerdo con lo detallado en los métodos de transición, es imposible ensayar aquí transiciones tipo Dual Stack o cualquier alternativa de túneles debido a que no se cuenta con el stack IPv4 correspondiente. Por lo tanto, solo son útiles los métodos de traslación de protocolos.

Solo en los casos que se cuente con el código fuente del sistema operativo de TODOS los hosts, es posible pensar en un método de traslación del tipo BIA o BIS. En la mayoría de las redes, esta alternativa no es válida, pues los sistemas operativos Microsoft de la familia Windows, no cuentan con ello.

Tampoco son aconsejables los métodos SIIT y NAT-PT , de acuerdo a las recomendaciones del IETF, debido los problemas normales de NAT y la posible pérdida de información de encabezado [31].

La alternativa de un TRT es factible y bastante usada, encontrando inclusive alguna implementación libre disponible para su ensayo. Como inconveniente requiere un DNS Proxy especialmente configurado para funcionar correctamente. Este DNS debería estar accesible desde la red IPv6, lo cual no está previsto para el Escenario planteado.

La implementación más viable para este tipo de escenarios es desarrollar un ALG para los protocolos específicos que se necesiten, como es el caso del Prox6. Como ventajas se pueden nombrar la facilidad de implementar ACLs, controles de recursos, controles de usuarios,etc,etc. Por otro lado, la desventaja principal está asociada a performance, debido a que la conversión se realiza en capa de aplicación.

El problema que presentan los métodos de traslación es su poca escalabilidad, pues para cada aplicación o protocolo se debe realizar una nueva aplicación, esto impacta directamente en el costo de la solución.

7 Agradecimientos

La presente monografía fue realizada con la colaboración de la UTN-FRM y el grupo de investigación GridTICS, permitiéndome utilizar la infraestructura del "CODAREC6 Test Bed".

A su vez, la Secretaría de Políticas Universitarias (SPU) ha contribuido con equipamiento para el laboratorio de IPv6 mediante el proyecto Promei "Fortalecimiento de la Capacitación e Investigación en Tecnologías de la Información y las Comunicaciones(TICs)".

Finalmente se agradece a los alumnos investigadores y graduados del grupo CridTICS que colaboraron permanentemente y de manera desinteresada con las tareas de ensayos.

8 Referencias

[1] Douglas E. Comer. *Redes Globales de Información con Internet y TCP/IP* . Pearson PH, tercera edición,1996, ISBN 0-13-219687.
[2] R. Gilligan. *Transition Mechanisms for IPv6 Hosts and Routers*. RFC 2893, August 2000.
[3] C. Taffernaberry, A. Dantiacq Picolella, G. Mercado y A. Francisconi. *CODAREC6: AN IPv6 TEST BED*, XII CACIC, Octubre 2006, Potrero de los Funes, San Luís.
[4] Carlos Taffernaberry, Gustavo Mercado, Cristian Perez, Raul Moralejo y Sebastián Tobar. *Prox6: Implementación de un Application Layer Gateway (ALG) para transición hacia redes IPv6.* V EnIDI. Noviembre 2009, Los Reyunos, Mendoza.
[5] Pete Loshin. *IPv6: Theory, Protocol and Practice,* Morgan Kaufmann, Segunda Edición, 2004, ISBN 1-55860-810-9.
[6] Silvia Hagen, *IPv6 Essentials.* Segunda Edición, O`Reilly, 2006, ISBN 0-596-10058-2.
[7] T. Chown. *IPv6 Campus Transition Scenario Description and Analysis.* Internet-Draft, March 2007.
[8] S. Roy, J. Paugh, A. Durand, *Issues with Dual Stack IPv6 on by Default*. Internet-Draft, July 2004.
[9] A. Durand, P. Fasano. *IPv6 Tunnel Broker.* RFC 3053, January 2001.
[10] C. Huitema, "An Anycast Prefix for 6to4 Relay Routers", RFC 3068, June 2001.
[11] B. Carpenter, K. Moor, B. Fink, *Connecting IPv6 Routing Domains Over the IPv4 Internet*, IPJ, March 2000 Volume 3, Number 1.
[12] F. Tremplin, T. Gleeson, M. Talwar, D. Thaler, *Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)*, RFC 4214, Octubre 2005.
[13] JOIN - The IPv6 project at the Center for Information Processing, http://www.join.uni-muenster.de/Dokumente/Howtos/Howto_ISATAP.

[14] C. Huitema. *Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs)*, RFC 4380, February 2006.
[15] E. Nordmark Stateless IP/ICMP Translation Algorithm (SIIT) RFC 2765, February 2000.
[16] G. Tsirtsis, P. Srisuresh, *Network Address Translation - Protocol Translation (NAT-PT)*, RFC 2766, February 2000.
[17] K. Tsuchiya, H. Higuchi, Y. Atarashi, *Dual Stack Hosts using the "Bump-In-the-Stack" Technique (BIS)*, RFC 2767, February 2000.
[18] S. Lee, M-K. Shin, Y-J. Kim, E. Nordmark, A. Duran, *Dual Stack Hosts Using "Bump-in-the-API" (BIA)*, RFC 3338, October 2002.
[19] J. Hagino, K. Yamamoto, *An IPv6-to-IPv4 Transport Relay Translator*, RFC 3142,June 2001.
[20] B. Carpenter, K. Moore. *Connection of IPv6 Domains via IPv4 Clouds*, RFC 3056, Febrary 2001.
[21] S. Thomson, Bellcore, T. Narten. *IPv6 Stateless Address Autoconfiguration*, RFC 2462, Dicember 1998.
[22] Nick Kew. *The Apache Modules*, 1 edition . Prentise Hall, Febrary 2007.
[23] Arnold Robbins and Nelson Beebe. *Classic Shell Scripting*, 1 edition, O'Reilly - ISBN: 9780596005955. May 2005.
[24] Squid – http://www.squid-cache.org/.
[25] Python Language Program – http://www.python.org.
[26] Benchmark Tool. http://www.netperf.org/.
[27] A. Conta. *Internet Control Message Protocol (ICMPv6)* , RFC 2463, December 1998.
[28] J. Reynolds, J. Postel, *Assigned Numbers*, RFC 1700, Oct 1994.
[29] Hurricane Electric Internet Services. http://www.he.net/.
[30] Open Virtual Private Network. *http://openvpn.net/*.
[31] C. Aoun, E. Davies, *Reasons to Move the Network Address Translator - Protocol Translator (NAT-PT) to Historic Status*, RFC 4966, July 2007.