ARP-Path Bridges: Implementación de Shortest Path Bridges Ethernet basados en ARP sobre Linux y Openflow/NetFPGA

Guillermo Ibáñez, Jad Naous, Elisa Rojas, Diego Rivera, Juan A. Carral, Juan Ramón Velasco

***Resumen* — En este artículo se describen los conmutadores ARP-Path (también denominados FastPath y sus implementaciones recientes en Linux y NetFPGA. ARP-Path es una evolución conceptual de los puentes transparentes con aprendizaje que no requiere protocolo auxiliar de encaminamiento (ni de árbol de expansión) en capa dos, a diferencia de las propuestas actualmente en estandarización RBridges y Shortest Path Bridges. Cada host establece, en los conmutadores ARP-Path, un camino mínimo al mismo tiempo que se envía el paquete estándar ARP_Request, pero inundado por *todos* los enlaces. El camino marcado por el paquete ARP Request que alcanza el destino se confirma aprovechando el paquete ARP_Reply de respuesta del host destino. Se han realizado implementaciones de puentes ARP-Path en Linux y en la plataforma Openflow con tarjetas NetFPGA (actualmente en pruebas), así como simulaciones sobre Omnet. Las prestaciones son similares o ligeramente superiores a las de enrutadores de camino mínimo y muy superiores a STP. Las pruebas con tráfico real y de reconfiguración muestran las robustez y rapidez del protocolo. ARP-Path no modifica la trama de Ethernet y es compatible con puentes estándar en modo núcleo-isla.**

I. Introducción

Las redes metropolitanas o redes campus basadas en Ethernet ofrecen ventajas tales como sencillez de configuración (incluso autoconfiguración completa), la de evitar la administración de direcciones IP en subredes, alto rendimiento y bajo coste. Actualmente se basan en los estándares establecidos de protocolos de árbol de expansión como el protocolo Rapid Spanning Tree (RSTP) [1] y Multiple Spanning Tree (MSTP) o en la utilización de topologías libres de bucles. Sin embargo, este tipo de redes no son escalables por las grandes limitaciones impuestas por el protocolo de árbol de expansión utilizado para evitar bucles y por la necesidad de segmentar la red en distintas subredes IP mediante enrutadores para limitar la zona de fallo ante una tormenta de tramas de difusión en capa dos. Para mejorar la escalabilidad de las redes campus Ethernet, hasta formar una subred IP única de forma fiable, existen propuestas en estandarización tales como Routing Bridges [3] en el grupo TRILL del IETF, y Shortest Path Bridges [4] en el grupo IEEE 802.1aq. En el presente documento se presenta una alternativa para implementar el encaminamiento Shortest Path sin emplear protocolos de encaminamiento en capa dos, como IS-IS. Entendemos que este método de encaminamiento es adaptable al entorno SPB [4].

II. El protocolo ARP-Path

A continuación describimos brevemente el funcionamiento de ARP-Path recientemente desarrollado por nosotros [2], junto con algunas de sus características.

A. Descubrimiento de camino (ARP Request)

Un "*ARP-Path*" es el camino más rápido (y, por tanto, único), creado por la primera copia (y única, como se verá más adelante) de la petición ARP que llega al destino solicitado, por carrera por todos los enlaces de la red. El proceso se describe en la figura 1. Básicamente, cuando el host S (fuente) envía un ARP Request en broadcast para resolver la dirección IP de un host D (destino), éste se esparce por toda la topología y cada puente la recibe por uno o más puertos.

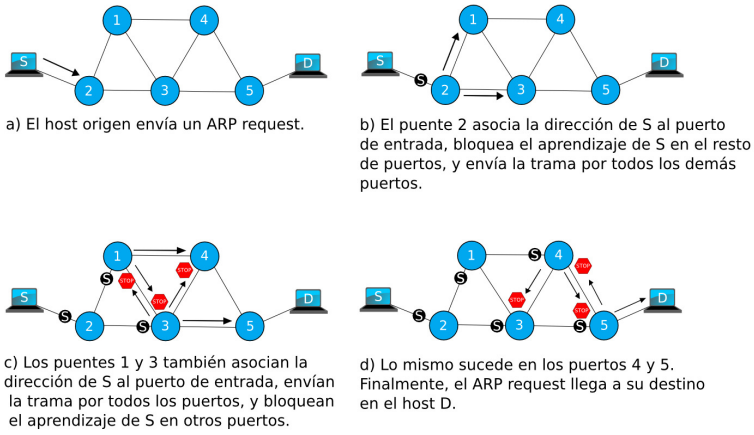


Figura 1: Descubrimiento del camino desde el host S al host D

Así pues, el camino "más rápido" se genera en base a que los bridges capturan (y reenvían en broadcast) la primera trama del *ARP Requests* que reciben, es decir, la que llegó primero, asocian el puerto por el que llegó a la dirección MAC origen de la trama y le asignan el estado "*Locked*" al par dirección-puerto, bloqueando el resto de puertos para las tramas que se reciban del mismo origen, por ser caminos "más lentas" y contener tramas duplicadas, redundantes en el árbol de inundación de ARP Request formado desde el host origen hasta el resto de los hosts.

Esto sucede en todos los puentes hasta que el ARP Request alcanza el host destino. De manera que los puentes de S a D se han quedado con uno de sus puertos en estado bloqueado para esa dirección (*locked*), y este estado posee un temporizador que establece un tiempo durante el que se podrá pasar el puerto a estado confirmado, o se liberará si vence la temporización de bloqueo sin haber sido confirmado el camino, lo que sucederá para todas las ramas del árbol de inundación creadas por el ARP Request excepto la rama que se confirme con la respuesta del host destino, como se indica a continuación.

B. Confirmación de camino (ARP Reply)

Como hemos comentado en el paso anterior, una serie de puentes quedan con uno de sus puertos en estado "Locked". Es un paso intermedio que se confirma (o no), mediante la segunda parte del protocolo, que se realiza en sentido contrario con la respuesta ARP, según se explica en la fig. 2:

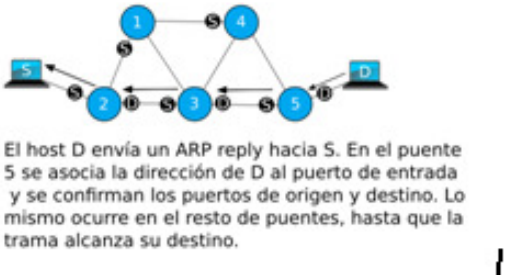
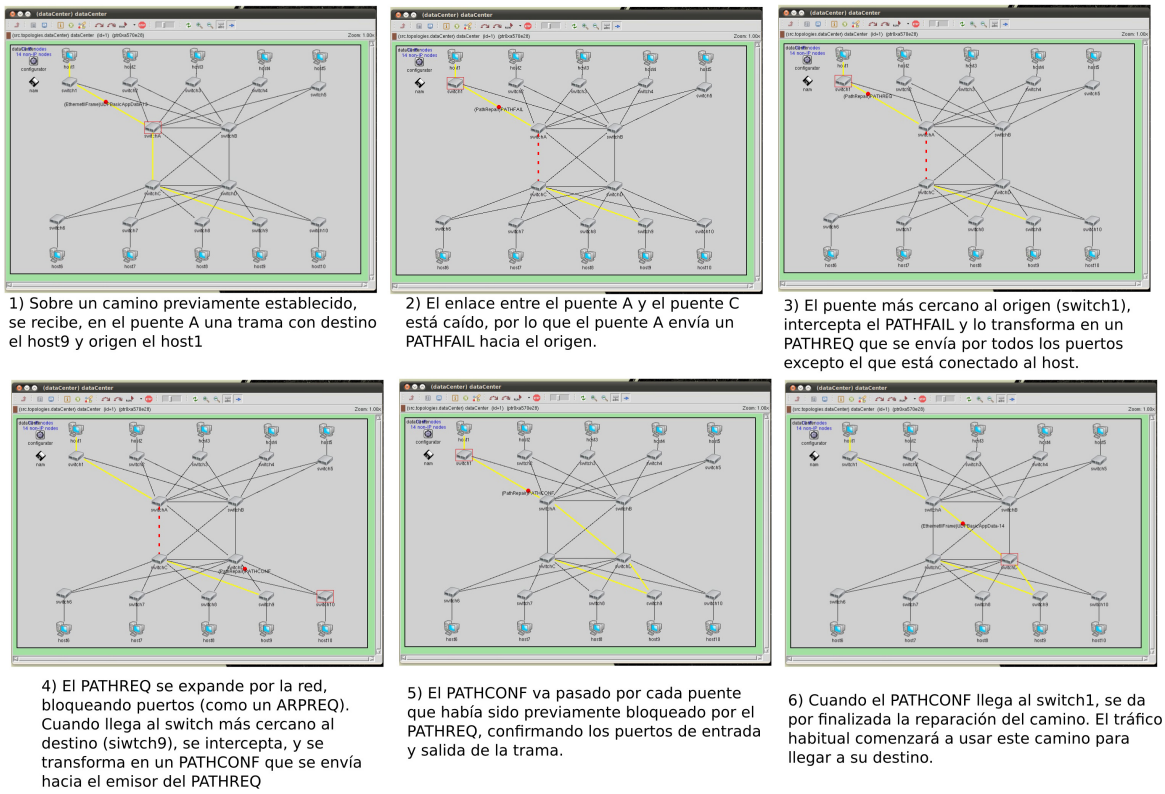


Figura 2. Confirmación de camino desde D a S

Este mecanismo asegura que el camino es simétrico. El camino en la dirección S->D se confirma hacia atrás mediante la respuesta unicast del destino sobre el mismo en dirección D->S, es decir, algunos de los puertos "Locked" anteriores pasarán a un estado confirmado "Confirmed" y otros caducarán (cuando expire su temporizador), mientras que a su vez el host D asociará nuevos puertos a su MAC, directamente confirmados, mediante la respuesta ARP. Finalmente los puertos "Confirmed" tendrán a su vez un temporizador, que se irá refrescando con las nuevas tramas que por ellos pasen con direcciones origen S y D respectivamente.

C. Reparación de camino

Los "ARP-Paths" pueden deshacerse debido a la expiración del temporizador (de algún puerto "Confirmed"), por fallos de algún enlace (esto a su vez provoca el borrado de las MACs asociadas a los dos puertos del enlace), etc. Cuando sucede, la asociación puerto-dirección pasa a estado "*Repairing*" o *en reparación*. Para la reconstrucción de un camino se procederá de la siguiente manera (ver fig. 3): El puente que recibe una trama y desconoce su destino, devuelve la trama unicast recibida encapsulada en un "*Path_Fail*" al host origen de la misma, enviándola en modo multicast a la dirección *All_ARP-Path_Bridges*. El puente con conexión directa a dicho host origen, intercepta la trama y reinicia la creación del "ARP-Path" mediante un paquete especial "*Path_Request*", que funciona de forma similar a un ARP request, inundando la red ARP-Path, reenviándose hasta encontrar el puente destino. Este "*Path_Request*" será interceptado por el puente conectado al host de destino el cual generará un paquete "*Path_Confirm*" que, recorriendo el camino bloqueado por el request, confirmará en cada uno de los puentes el origen y destino originales de la trama encapsulada. La interceptación de paquetes de reparación por parte de los puentes frontera (aquellos conectados directamente a los hosts implicados en el camino), se puede realizar mediante el aprendizaje por parte de estos nodos de la dirección MAC de estos hosts, asociándola al puerto por el que se realiza la conexión. Esto se consigue mediante una serie de paquetes *Hello* periódicos entre puentes ARP-Path. Los puentes ARP-Path que no reciben paquetes Hello por algún puerto asocian las direcciones MAC de los host aprendidas en esos puertos como hosts a su cargo, conectados directamente o a través de puentes estándar.



1) Sobre un camino previamente establecido, se recibe, en el puente A una trama con destino el host9 y origen el host1

2) El enlace entre el puente A y el puente C está caído, por lo que el puente A envía un PATHFAIL hacia el origen.

3) El puente más cercano al origen (switch1), intercepta el PATHFAIL y lo transforma en un PATHREQ que se envía por todos los puertos excepto el que está conectado al host.

4) El PATHREQ se expande por la red, bloqueando puertos (como un ARPREQ). Cuando llega al switch más cercano al destino (siwtch9), se intercepta, y se transforma en un PATHCONF que se envía hacia el emisor del PATHREQ

5) El PATHCONF va pasado por cada puente que había sido previamente bloqueado por el PATHREQ, confirmando los puertos de entrada y salida de la trama.

6) Cuando el PATHCONF llega al switch1, se da por finalizada la reparación del camino. El tráfico habitual comenzará a usar este camino para llegar a su destino.

Figura 3: Reparación de camino (simulador Omnet)

E. Etherproxy para mejora de escalabilidad

El uso de ARP en redes grandes es criticado frecuentemente por el aumento de la carga de proceso en los hosts para resolver los ARP Requests al crecer el número de hosts de la red, porque todos los hosts de la red deben procesar los ARP Request recibidos. Para limitar esa carga existe la posibilidad de insertar *proxys* ARP o Etherproxies [5] en la red, sea dentro de los mismos puentes ARP-Path o como elementos independientes, cerca de los puentes frontera de la red. Para ARP-Path, la implementación en los puentes presenta ventajas de sinergia, dado que capacitar a los conmutadores ARP-Path para actuar como proxy ARP requiere básicamente añadir una columna de dirección IP a las tablas de MAC aprendidas por el conmutador.

La tasa de ARP Requests efectiva, con el uso de cachés queda reducida por la tasa de fallo de caché (miss rate $m = 1 - h$, hit rate, tasa de aciertos de la cache)

$$Tasa\ efectiva\ de\ ARP\ \ Requests = \ ARPRequest\_rate \cdot (1-h)$$

Como se muestra en [5] la tasa de aciertos puede llegar al 100% si se utilize refresco proactive por los switches de las direcciones de hosts proximas a expirar. This refreshing consists of an ARP Request unicast sent by the proxy located at FastPath bridge to the expiring host MAC address, the ARP Reply from the host refreshes the cache. Sin refresco, la tas de exito es de 0,6-0,8 segun las redes evaluadas [5]. Actualmente estamos incorporando la funcionalidad Etherproxy simplificada a las implementaciones ARP-Path.

III. Implementaciones del protocolo

A. Implementación sobre Linux

Se ha implementado una prueba de concepto del protocolo ARP-Path sobre el kernel Linux 2.6 utilizando *ebtables*, que funciona fundamentalmente en espacio de usuario. Esta implementación se ha orientado a la verificación funcional del protocolo y no a la maximización o evaluación del rendimiento, lo que requeriría una implementación completa en espacio del kernel.

Para verificar la compatibilidad básica con la red campus y de servicios estándar que utilizan difusión tales como DHCP, se han conectado dos hosts a la red del campus a través de un triángulo compuesto de tres switches ARP-Path implementados en PCs estándar con Linux, incluyendo la instalación de la funcionalidad de puente ARP-Path. Los resultados muestran que los hosts obtienen sus direcciones via DHCP igual que antes de imponer la red de conmutadores . Igualmente el acceso a Internet, páginas Web, transferencia de ficheros o reproducción de vídeo a través de la red. No aparecen bucles en el tráfico broadcast incluso conectando entre sí dos puertos del mismo puente. Sí aparecen bucles si el puente estándar (no ARP-Path) situado en la frontera de la red campus se conecta mediante dos o más enlaces a la red ARP-Path. Este escenario de red ARP-Path en la parte de acceso no se corresponde con el escenario típico de uso de ARP-Path ni los mecanismos de compatibilidad descritos más arriba. Se han medido y comparado los retardos de ping entre dos hosts separados por un puente ARP-Path basado en Linux y entre dos hosts separados por un switch Ethernet estándar D-Link 10/100 Mbps. Se activó el modo de auto negociación

en ambos casos. Cuando el host no conoce la MAC del destino en su caché, envía un ARP. El procesado del ARP Request/Reply y del primer ping por tanto, puede tardar hasta 440 ms en el ARP-Path Linux, y sólo 2,43ms en el switch estándar (hardware). Esto es debido a la transferencia de tráfico entre el kernel y el espacio de usuario en el establecimiento de caminos en la implementación de Linux. Una vez se ha establecido el camino, con el primer ARP request/reply, la respuesta del ping tarda 450 microsegundos de media en el puente ARP-Path Linux, y 200 microsegundos para el puente estándar hardware. La razón para la relativa velocidad del puente Linux cuando una dirección se ha aprendido, es que en este caso no es necesaria la comunicación entre kernel y espacio de usuario, ya que el reenvío se realiza directamente por el kernel Linux.

La reparación de caminos se ha evaluado también en la red de la fig. 4. Mientras se envían pings continuamente de un host a otro a través del camino establecido entre los dos hosts H a través del enlace directo que los une, se desconecta dicho cable. Una vez se detecta el fallo del enlace Ethernet, el tiempo de restablecimiento del camino es 672 mseg desde el ultimo ping enviado.Estableciendo un árbol de expansión en la red, los hosts H quedan a una distancia de dos saltos debido al bloqueo del enlace redundante directo que los une. El retardo con ARP-Path en este caso se duplica hasta 880 microsegundos,como era de esperar. La tasa máxima de ARPs que soporta el sistema en Linux actualmente es de 100 ARP Request/segundo.

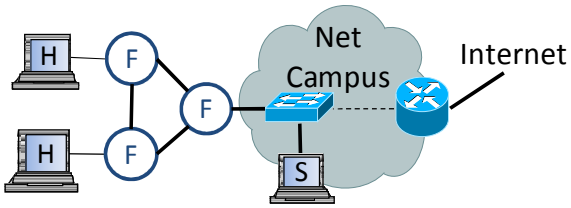


Figura 4. Red de evaluación de Conmutadores ARP-Path (F) en Linux

Figura 5. Demostración de Conmutadores ARP-Path en Linux

B Implementación sobre Openflow / NetFPGA

Openflow [8] es una plataforma abierta promovida por la Universidad de Stanford basada en tratamiento de tráficos por flujos, orientada a posibilitar a los investigadores la prueba de protocolos nuevos sobre hardware comercial instalado en redes (routers, switches, etc) corporativas que soporte Openflow, sin afectar al tráfico normal. NetFPGA [9] es una plataforma para implementar rápidamente nuevos protocolos sobre equipos de red de altas prestaciones mediante FPGAs a velocidades de 1Gbps por puerto. NETFPGA es una tarjeta PCI que contiene una FPGA de Xilinx, 4 puertos Gigabit Ethernet, 4.5MB de SRAM, 64MB de DDR2 DRAM. El diseño de la misma es de código abierto. En nuestro caso utilizamos la posibilidad que nos da NetFPGA de actuar como switch OpenFlow, en el que posee una alta capacidad de procesado de paquetes a nivel hardware en sus 4 interfaces Gigabit Ethernet. Instalando una o más tarjetas NetFPGA en un host, configurando una versión de OpenFlow sobre ellas y conectándolas a un controlador común (podría ser múltiples controladores) podremos construír diferentes topologías. La implementación ARP-Path sobre Openflow/NetFPGA hace posible dos escenarios de prueba: Usar tarjetas NetFPGA o bien switches comerciales que soporten Openflow como NEC IP8800, HP Procurve y Toroki Lightswitch. En ambos casos con un controlador NOX externo para la lógica de flujos.

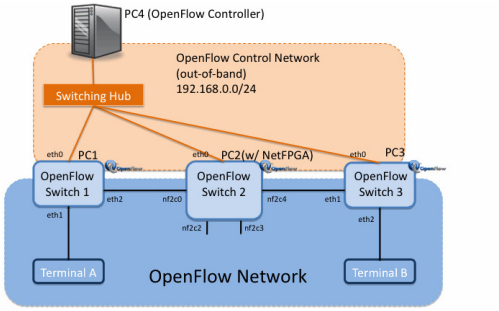


Figura 6 Implementación mediante OpenFlow y NetFPGA [9]

. Ya se ha desarrollado una implementación de ARP-Path en Openflow y se ha probado sobre diferentes topologías en máquinas virtuales y actualmente se prueba sobre una red de cuatro conmutadores NetFPGA, con tráfico real. El funcionamiento de un switch OpenFlow separan el data Path del control path, dejando el "Data Path" donde está y externalizando el "Control Path", implementándolo a partir de un controlador externo centralizado. Dicho controlador establecerá un canal seguro con cada switch OpenFlow (ver fig. 6) de comunicación sobre el que intercambiarán mensajes OpenFlow (tales como packet-received, send-packet-out, modify-forwarding-table, get-stats,..). Mientras, en el "Data Path" existe una tabla que contiene diferentes campos de tipos de paquetes (cuyas decisiones ya fueron realizadas por el controlador en ocasiones anteriores) y la acción a realizar con ellos (como ocurre en los switches estándar). Cuando el switch OpenFlow recibe un paquete que no encaja en ninguna de las entradas de la tabla, lo envía al controlador y éste decide qué hacer con él.

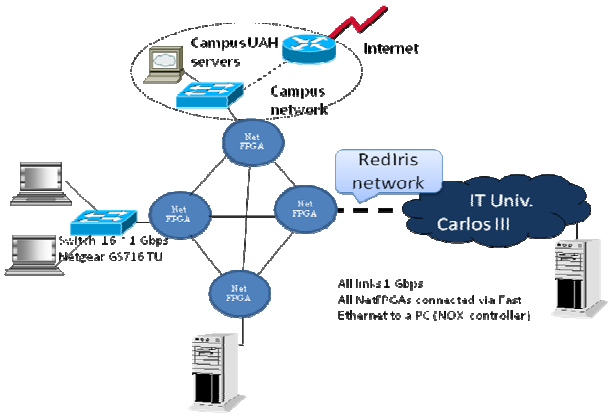


Figura 7: Red de demostración y evaluación de ARP-Path con OpenFlow/NetFPGA

Se han realizado medidas de reparación de caminos en la topología de la figura. Se envían pings entre los dos NetFPGA conectados por el enlace horizontal. El retardo de ida y vuelta del primer ping es de 28 mseg de promedio e incluye el primer ARP Request y Reply para resolver la dirección MAC destino y los mensajes ICMP de ida y de vuelta. Los siguientes ping, una vez establecido el camino y sin requerir ARP Request/Reply, presentan un retardo de 630 microsegundos de media. El tiempo total desde fallo de enlace hasta reparación de un camino es la suma del tiempo de detección del fallo hardware del enlace más el tiempo de reparación del camino de nuevo, arriba indicado. La tasa máxima que soporta de ARP Request el sistema es de 1000 ARP Requests/segundo.

Está prevista la implementación completa y directa de ARP-Path sobre NetFPGA, sin Openflow ni controlador, una vez estabilizado y optimizado el diseño del protocolo, dado que Openflow tiene gran facilidad de modificación de la lógica del protocolo, por su esquema de flujos, muy superior a NetFPGA.

IV. Evaluación

A. Complejidad

Si comparamos ARP-Path con protocolos como Spanning Tree o protocolos de "routing" como IS-IS, se puede ver que estos últimos son protocolos proactivos: En STP, cada nodo emite periódicamente su mejor BPDU (la ruta de menor coste hacia el puente raíz) hacia sus vecinos, procesando por tanto *d* (siendo *d* el grado medio del nodo) BPDUs recibidas por cada puerto para seleccionar el vecino que ofrece la distancia más corta hasta el puente raíz. Esto significa que la complejidad en mensajes es $\Theta(d)$. Los puentes que utilicen protocolos basados en el estado del enlace (como el algoritmo del camino más corto de Djikstra), tienen, para una red con N puentes tienen una complejidad N^2 (como mínimo $N*logN$). Además, necesitan un mecanismo adicional de sincronización para prevenir bucles causados por inconsistencias temporales en las rutas. Aunque N no sea un número muy grande, cada puente debe mantener informados a los otros de los hosts asociados a él. Esto significa que las tablas de reenvío pueden crecer hasta ser bastante grandes, y la cantidad de tráfico de control para mantener actualizada la lista de hosts activos puede ser significativo.

Sin embargo, ARP-Path es un protocolo reactivo, ya que los puentes no intercambian información de rutas de forma periódica. Al usarse paquetes estándar (ARP) de los que ya se intercambian entre los hosts, no se añade ningún coste adicional de mensajes, excepto las tramas usadas cuando expira una ruta (Path Fail, Path Request y Path Conf), que generan algunos mensajes extra para reconstruir el camino, y de los paquetes locales Hello, que identifican en qué puentes están conectados los hosts. Sobre la información almacenada en cada caso, los puentes ARP-Path contienen una cantidad de información sobre el estado similar a los puentes estándar, es decir, una asociación entre puertos del puente y direcciones MAC. La única diferencia está en el uso de temporizadores (para los diferentes estados de las asociaciones: confirmada, bloqueada y en reparación).

B. Simulaciones en Omnet

Se ha realizado una implementación de ARP-Path sobre la plataforma de simulación Omnet [7] para redes (INET), modificando la implementación de switches Ethernet de INET, adaptándolos a la funcionalidad ARP-Path. Al ser ARP-Path una evolución de los puentes transparentes, la implementación básica es sencilla y puede enriquecerse de forma progresiva con funcionalidades adicionales. Esta implementación se ha usado tanto para la verificación funcional del protocolo como para la realización de pruebas preliminares de evaluación. Recientemente, mediante los interfaces implementados en INET para la interconexión de redes simuladas mediante esta plataforma y módulos de red reales, se ha conectado una topología simple de estos switches a hosts reales, que se han conectado a través de la red simulada. En la implementación realizada sobre Omnet, se ha comparado el rendimiento del protocolo ARP-Path, el enrutado por camino más corto, y el Spanning Tree Protocol. Se han realizado simulaciones sobre un núcleo de red pan Europea, consistente en una malla plana de 16 nodos como se muestra en la figura 6 [7]. Se ha enviado tráfico UDP con longitudes de mensaje de 1 a 60KB, con una distribución exponencial de de media 75ms como tiempo entre mensajes. Los retardos ente enlaces son relativos a la distancia real en el mapa, y están entre 1 y algo más de 3ms. Todos los enlaces tienen la misma capacidad. El tráfico se origina en los hosts situados al oeste en el mapa, y se dirige hacia los nodos del este, facilitando así la saturación de los enlaces. La activación del envío de tráfico en los hosts se ha secuenciado de forma aleatoria con una media de 0,5 s entre las activaciones. Los enlaces se saturan ligeramente, como se muestra en la figura 7, más tarde con ARP-Path que con los routers usando el camino mínimo, y mucho más tarde que usando el Spanning Tree Protocol.

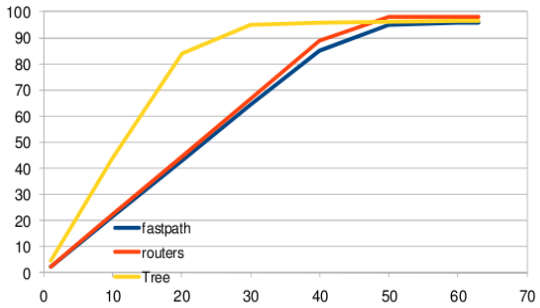


Figura 7. Comparación del trhoughput de la red Pan europea en % del enlace más cargado contra % de tráfico medio aplicado por el host

IV. Conclusiones

En este artículo se han presentado implementaciones de puentes ARP-Path Ethernet que reutilizan las tramas estándar ARP para el establecimiento del camino más rápido entre dos hosts utilizando cualquier enlace. Dada la sencillez del protocolo, la implementación en varias plataformas ha sido sencilla. Se ha comprobado la funcionalidad del protocolo con tráfico real en una red de la universidad. Las prestaciones son comparables a los routers de camino mínimo con una complejidad mucho menor. La implementacion de Etherproxy permitirá optimizar el tráfico ARP y mantener las prestaciones.

Agradecimientos

El presente trabajo ha sido parcialmente apoyado por la Comunidad de Madrid a través del proyecto MEDIANET-CM (S-2009/TIC-1468) y por la Comunidad de Castilla-La Mancha a través del proyecto EMARECE (PII1I09-0204-4319). Gracias a Bart de Schuymer por la rápida y efectiva implementación en Linux.

Referencias

[1] IEEE 802.ID-2004 *IEEE standard for local and metropolitan area netowrks-Media access control (MAC) Bridges*, http://standars.ieee.org/getieee802/802.1.html

[2] G. Ibáñez, J. A. Carral, A. Garcia-Martinez, J. M. Arco, D.Rivera, and A. Azcorra, "*Fast Path Ethernet Switching: On-demand, Efficient Transparent Bridges for Data Center and Campus Networks*", 17º IEEE Workshop on Local and Metropolitan Area Networks (LANMAN), New Jersey, USA, May 2010.

[3] *Transparent interconnection of lots of links (TRILL)* WG. Available on line at: http://www.ietf.org/html.charters/trill-charter.html

[4] M. Seaman. *Shortest Path Bridging*. http://www.ieee802.org/1/files/public/docs2005/new-seamanshoretestpath-par-0405-02.htm

[5] Elmeleegy, Khaled and Cox, Alan L. *EtherProxy: Scaling The Ethernet By Suppressing Broadcast Traffic*. Proceedings of IEEE INFOCOM 2009, Rio de Janeiro, Brazil.

[6] Omnet Simulatior. Available on line: http://www.omnetpp.or

[7] NRS Reference Networks http://www.ibcn.intec.ugent.be/INTERNAL/NRS/index.html

[8] Protocolo Openflow y ejemplos de implementaciones: http://www.openflowswitch.org/

[9] NetFPGA http://netfpga.org/