Universidad Nacional de La Plata
Facultad de Informática

Maestría en redes de datos

Protocolo PGM – Pragmatic General Multicast

Su aplicación en servicio de transferencia de archivos

Tesis presentada para obtener el grado de Magíster en Redes de Datos

Director de Tesis: Ing Luis Marrone

Alumno: Lic María Claudia Abeledo

Junio de 2005

Agradecimientos

A Luis Marrone, por su paciencia y su guía

A Martín Spotorno y Omar Cuarterolo, que hicieron factible la implementación.

A Carmela Lancellotta, Fátima Mastroianni y Guido Vasallo, siempre a mi lado.

A Daniel, Carla y Guido, esposo e hijos, que me soportaron.

Índice

Motivación, objetivos y planificación del trabajo realizado

Motivación

Como se explica más adelante, el interés en protocolos multicast tiene antecedentes sumamente extensos. La razón radica en el hecho de que estos protocolos permiten una distribución eficiente de datos sobre un grupo dinámico de receptores.

La videoconferencia, el aprendizaje a distancia, la distribución de software, la recepción de noticias e informaciones de mercado; la recepción de conciertos en vivo; la actualización de bases de datos y otras innumerables aplicaciones son posibles a través de esta modalidad.

Pero los protocolos de capa de transporte utilizados en su mayoría se caracterizan por su calificación de “best effort” (el mejor esfuerzo). ¿Existen protocolos confiables como TCP para este tipo de modalidad?.

En algunos aspectos, este trabajo pretende dar respuesta a este interrogante.

A continuación, se presentan sucintamente los objetivos del presente trabajo

Objetivos del trabajo

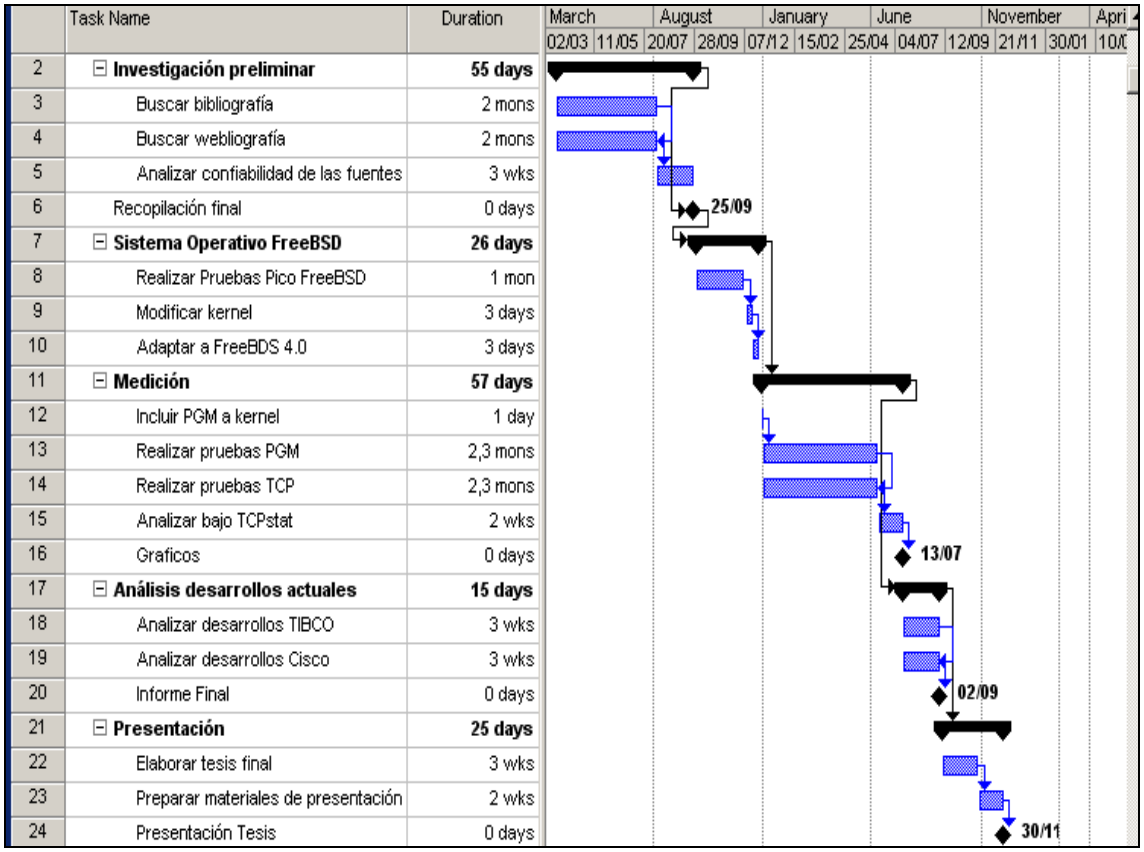
1. El primer punto a abordar fue el desarrollo de un estado del arte sobre el protocolo PGM (Pragmatic General Multicast) desde la publicación en Diciembre del 2001 del RFC 3208 con categoría de experimental, hasta la fecha. Esta parte del objetivo se centró en los desarrollos científicos y comerciales, aplicaciones y estudios comparativos con otros protocolos, sin perder de vista los principios que motivaron este trabajo.

2. Así se analizó, en particular, la factibilidad de implementación de un servicio de transferencia de archivos bajo PGM sobre una LAN. La experiencia de laboratorio se realizó en la Universidad CAECE.

3. Se confeccionó un resumen final de los resultados obtenidos sobre la experiencia, analizando ventajas, desventajas y se comparó con experiencias realizadas bajo otras plataformas y protocolos.

Planificación del trabajo realizado:

En el diagrama de Gantt se muestran el tiempo de dedicación a cada actividad o tarea del proyecto establecidas en la planificación del trabajo y que fueran cumplidas en su totalidad.



	Task Name	Duration
2	**Investigación preliminar**	**55 days**
3	Buscar bibliografía	2 mons
4	Buscar webliografía	2 mons
5	Analizar confiabilidad de las fuentes	3 wks
6	Recopilación final	0 days
7	**Sistema Operativo FreeBSD**	**26 days**
8	Realizar Pruebas Pico FreeBSD	1 mon
9	Modificar kernel	3 days
10	Adaptar a FreeBDS 4.0	3 days
11	**Medición**	**57 days**
12	Incluir PGM a kernel	1 day
13	Realizar pruebas PGM	2,3 mons
14	Realizar pruebas TCP	2,3 mons
15	Analizar bajo TCPstat	2 wks
16	Graficos	0 days
17	**Análisis desarrollos actuales**	**15 days**
18	Analizar desarrollos TIBCO	3 wks
19	Analizar desarrollos Cisco	3 wks
20	Informe Final	0 days
21	**Presentación**	**25 days**
22	Elaborar tesis final	3 wks
23	Preparar materiales de presentación	2 wks
24	Presentación Tesis	0 days

Capítulo 1: Introducción

¿Qué es multicast?

Existe una larga lista de antecedentes sobre el interés en protocolos multicast. La razón radica en el hecho de que estos protocolos permiten una distribución eficiente de datos sobre un grupo dinámico de receptores.

La tecnología multicast representa un servicio de red en el cual un único flujo de datos, proveniente de una determinada fuente, se puede enviar simultáneamente a diversos receptores interesados. Cabe a la infraestructura de red transportar este flujo de datos, replicándolo cuando sea necesario, para todos los receptores que registren interés en recibir estos datos.

El objetivo de la comunicación multicast es conectar a un grupo de usuarios o receptores. En la mayoría de los casos, el modelo de conexión de los usuarios o receptores se realiza mediante un "árbol" que interconecta a todos los miembros reenviando los mensajes a través de los enlaces correspondientes.[1]

En redes TCP/IP, estos receptores son representados por una dirección de grupo o dirección multicast. Esta dirección de grupo corresponde a una dirección IP que pertenece a la antigua clase D, es decir, en la franja entre 224.0.0.0 y 239.255.255.255. Cada fuente envía paquetes hacia una dirección de grupo (por ejemplo: 233.7.124.1), en el cual estarán asociados diversos receptores. Estos receptores, a su vez se pueden vincular y desvincular en forma dinámica.

Cabe a los dispositivos de la red y en particular a los enrutadores (routers), determinar cuáles de sus interfaces poseen receptores interesados en un grupo multicast y cuáles deberán recibir una copia de los paquetes enviados para ese grupo.

El multicast está orientado hacia aplicaciones del tipo "uno para muchos" y "muchos para muchos". En estos casos, presenta claras ventajas cuando se lo compara con los mecanismos de transmisión unicast y broadcast. En unicast, es necesario que la fuente replique varios flujos de datos idénticos con el objeto de transmitirlos a cada uno de los receptores, generando uso ineficaz del ancho de banda disponible en cada caso.

[1] **Protocolo Multicast de Congestión Mínima** – J. Alvarez-Hamelin- Luis Marrone – Depto de Electrónica – Facultad de Ingeniería – Universidad de Buenos Aires

Por otro lado, el sistema broadcast envía los datos a toda la red de forma indiscriminada. Esto también da como resultado el desperdicio de recursos, pues implica transporte de datos para todas las estaciones de la red, aunque el número de receptores deseosos de ese contenido, sea reducido. Con multicast, la fuente de tránsito envía una única copia de los paquetes hacia una dirección de grupo multicast. La infraestructura de red replica estos paquetes de forma inteligente, encaminando los datos de acuerdo con la topología de receptores interesados en esa información.

Entre las diversas aplicaciones que pueden obtener beneficios con el uso de multicast están: videoconferencia; aprendizaje a distancia; distribución de software, noticias e informaciones de mercado; conciertos en vivo; actualización de bases de datos; juegos distribuidos; procesamiento competidor; simulacros distribuidos etc...[2]

En el esquema multicast se realiza un aprovechamiento del ancho de banda del canal de comunicaciones mediante el envío de un único flujo que llega a un grupo final de usuarios independientemente del número de éstos. Las técnicas multicast precisan de nodos y protocolos capaces de organizar sesiones y distribuir los datos sólo a aquellos que son considerados integrantes del grupo.[3]

Descripción y estándares Multicast

La IETF (Internet Engineering Task Force) ha desarrollado estándares sobre los siguientes puntos:

- Esquema de direcciones multicast y concepto de grupo
- Registración dinámica
- Ruteo multicast

[2] http://www.rnp.br **– Rede Nacional de Ensino e Pesquisa**
La Rede Nacional de Ensino e Pesquisa (RNP – Red Nacional de Enseñanza e Investigación) es la infraestructura nacional de red avanzada para colaboración y comunicación en enseñanza e investigación. Además de interconectar todas las instituciones federales de enseñanza superior e investigación, esta red propicia un laboratorio para el desenvolvimiento experimental de nuevas aplicaciones y servicios de red para beneficio de sus organizaciones usuarias.

[3] **Análisis crítico de los sistemas de huella digital para multicast** – J.Campos, A. Martínez-Ballesté y J. Domingo-Ferrer – Departament d'Enginyería Informàtica y Matemàtiques -Universitat Rovira i Virgili

Esquema de direcciones multicast y concepto de grupo

Multicast está basado en el concepto de grupo. Un grupo arbitrario de receptores expresa interés en recibir un stream particular de datos. Este grupo no debe tener una localización física o geográfica específica. Los hosts pueden estar ubicados sobre cualquier lugar de la Internet. Los hosts interesados en recibir el flujo de datos deberán pertenecer a un grupo específico utilizando el protocolo IGMP (Internet Group Management Protocol). Los hosts deben ser miembros del grupo que recibirá el stream de datos.

Las direcciones Multicast especifican un grupo arbitrario de IP hosts que estarán unidos para recibir el tráfico.

El IANA (Internet Assigned Numbers Authority) controla las asignaciones de direcciones IP Multicast. IANA tiene asignado el espacio de direcciones de la vieja Clase D para ser usado por IP Multicast. Como fuera previamente indicado, las direcciones de IP Multicast caen dentro de este rango:

224. 0 . 0 . 0 - 239. 255. 255. 255

La utilización de este rango de direcciones es única para el grupo de direcciones de destino del tráfico IP Multicast. La fuente de direcciones para datagramas multicast es siempre la fuente de direcciones unicast.

IANA ha reservado direcciones desde 224.0.0.0 hasta 224.0.0.255 para ser usado a través de protocolos de red en un segmento de una red local. Los paquetes con estas direcciones nunca deben ser reenviados por un router. Se reduce su uso a un segmento particular de una LAN. Estos paquetes son siempre transmitidos con un TTL de 1.

Los protocolos de red usan estas direcciones para descubrir automáticamente y comunicar importante información de ruteo. Por ejemplo: OSPF (que se describirá más adelante) usa 224.0.0.5 y 224.0.0.6 para intercambiar links de estados de información. Abajo se detallan las direcciones más conocidas:

Dirección	Uso
224.0.0.1	Todos los sistemas en esta subred
224.0.0.2	Todos los routers en esta subred
224.0.0.5	Routers OSPF
224.0.0.6	Routers designados OSPF
224.0.0.12	Protocolo de configuración dinámica del host DHCP.

El rango de direcciones desde 224.0.1.0 hasta 238.255.255.255 se denomina direcciones de aplicación global. Pueden ser usadas para datos multicast entre organizaciones y a través de Internet.

Algunas de estas direcciones se han reservado para el uso de aplicaciones multicast a través de IANA. Por ejemplo: 224.0.1.1 ha sido reservada para el protocolo de tiempo de red (NTP).

El rango de direcciones desde 239.0.0.0 hasta 239.255.255.255 se denomina direcciones de campo limitado o direcciones de campo administrativo. Se definen en el RFC 2365 para limitarlas a un grupo local u organización. Los routers son típicamente configurados con filtros para prevenir tráfico multicast en el rango de direcciones en el flujo fuera de un Sistema Autónomo u otro dominio de usuario definido. Dentro del Sistema Autónomo o rango de dominio de la Dirección de Campo Limitado, éstas pueden ser subdivididas tal que los entornos del multicast local queden definidos. Esto permite reusar las direcciones entre dominios pequeños.

La RFC 2770 propone que el rango de direcciones 233.0.0.0/8 sea reservado para direcciones definidas estáticamente por organizaciones que ya tiene un número de Sistema Autónomo reservado. Esta práctica se denomina direccionamiento GLOP.
El número de Sistema Autónomo del Dominio es embebido dentro del segundo y tercer octeto del rango de direcciones 233.0.0.0/8.

Por ejemplo: El Sistema Autónomo 62010 es F23A en hexadecimal. Si separamos “F 2 “y “3 A” se obtienen respectivamente 242 y 58 en formato decimal. Estos valores se traducen en la subred 233.242.58.0 correspondiente al Sistema Autónomo 62010.

En el direccionamiento multicast para capa 2, normalmente, las NIC´s sobre un segmento de LAN solamente recibirán paquetes destinados a través de su dirección MAC o su dirección MAC de broadcast. Esto significa que algunas veces se podría recibir el mismo paquete y éste debería ser diferenciado entre los distintos grupos multicast.

Las especificaciones LAN IEEE prevén esta situación para la transmisión de paquetes broadcast y multicast. En el standard 802.3, el bit 0 del primer octeto se usa para indicar un frame broadcast y/o multicast. La figura 1 muestra la localización del bit Broadcast/Multicast en un frame Ethernet.

Figura 1

Este bit indica que el frame está destinado por un grupo de hosts arbitrario o todos los hosts sobre una red (en el caso de la dirección de broadcast 0xFFFF.FFFF.HF)

IP Multicast hace uso de su capacidad de transmisión de paquetes IP a un grupo de hosts sobre un segmento LAN.

Para el caso de mapeo de direcciones en Ethernet MAC, IANA posee un bloque de direcciones que comienzan con 01:00:5E en hexadecimal. La mitad del bloque es alojada para direcciones multicast. Esto crea 0100.5e00.0000 a través 0100.5e7F. HF como el rango disponible de direcciones MAC Ethernet.

Esta ubicación hace corresponder direcciones Ethernet de 23 bits con grupos de direcciones Multicast dentro de estos 23 bits disponibles para Ethernet. La Figura 2 es indicativa de estos conceptos:

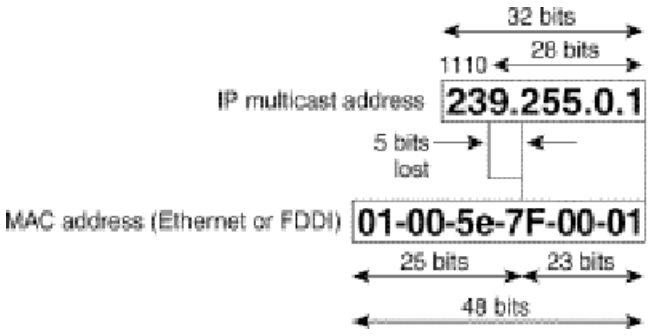


Figura 2

Ya que los cinco primeros bits de la dirección IP multicast están omitidos en este mapeo la dirección resultante no es única. Por lo tanto, 32 ID de grupos diferentes mapean a la misma dirección Ethernet como se sintetiza a continuación en la Figura 3

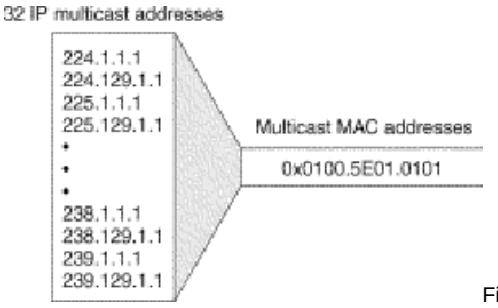


Figura 3

Registración dinámica

IGMP (Internet Group Management Protocol) es usado para registrar dinámicamente hosts individuales dentro de un grupo multicast en una LAN particular.. Los hosts se identifican como miembros de grupo enviando mensajes IGMP hacia su router local multicast.

Bajo IGMP, routers escuchan mensajes IGMP y periódicamente distribuyen preguntas para descubrir los grupos activos o inactivos de la subred particular.

El RFC 1112 define la especificación para IGMP Versión 1. Un diagrama del formato del paquete se muestra a continuación en la Figura 4:

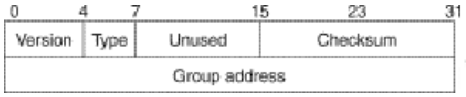


Figura 4

En la versión 1, solamente existen dos tipos de mensajes:

- Membership query
- Membership report

Los hosts distribuyen membership report IGMP hacia un grupo multicast particular para indicar que están interesados en unirse a ese grupo. El router (enrutador) periódicamente distribuye membership query IGMP para verificar que al menos un host en una subred está todavía interesado en recibir tráfico directamente de este grupo.

Cuando no hay respuesta a tres membership queries IGMP consecutivos el router determinará el time out del grupo y detiene el tráfico dirigido hacia ese grupo.

El RFC 2236 define la especificación IGMP versión 2. Un diagrama del formato del paquete se muestra a continuación en la Figura 5:

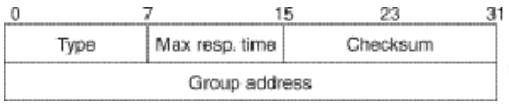


Figura 5

Existen 4 tipos de mensajes en esta versión:

- Membership query
- Version 1 membership report
- Version 2 membership report
- Leave group

La versión 2 de IGMP trabaja básicamente de la misma forma que la versión 1. La diferencia principal es que existe un mensaje de abandono del grupo (Leave group). El host puede comunicar activamente a un router multicast local su intención de dejar el grupo. El router entonces distribuye un grupo específico de queries y determina si algún host remanente tiene la intención de recibir el tráfico. Si no hay respuestas, el router dará el time out al grupo y detendrá el tráfico.

Esto puede reducir enormemente la latencia de abandono de la conexión comparado con la versión 1. El tráfico innecesario y no requerido es frenado mucho más rápido.

El protocolo IGMP versión 3 es el próximo escalón en la evolución de IGMP. La versión añade soporte para fuentes de filtrado, las cuales habilitan un host receptor multicast para señalar a un router el grupo desde el cual se recibirá el tráfico multicast y desde que fuentes este tráfico es esperado.

Esta información del grupo de miembros permite al software Cisco IOS responder al tráfico desde solamente aquellas fuentes donde los receptores requieren el tráfico.

Un diagrama del formato de un paquete query para un mensaje IGMPv3 se muestra a continuación en la Figura 6 con una tabla que contiene la descripción de los campos más importantes.

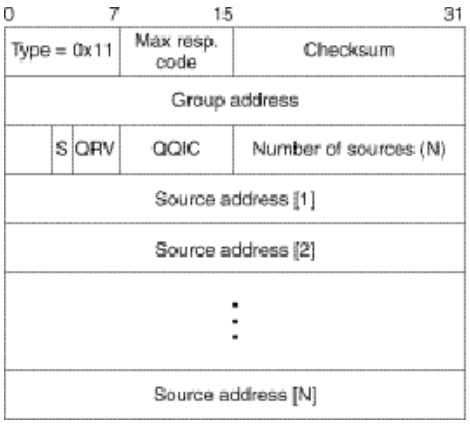


Figura 6

Campo	Descripción
Type=0x11	IGMP Query
Max resp. code	Código de respuesta máxima (en segundos). Si ese código es menor que 128, entonces este es igual al tiempo de respuesta máxima. Si ese código es mayor o igual que 128, éste representa un valor de punto flotante (en mantisa y formato exponencial).
Grupo de direcciones	Dirección de grupo multicast. Esta dirección es 0.0.0.0 para queries generales.
S	S flag. Este flag indica que los procesamientos de ruteo fueron suprimidos.
QRV	Querier Robustness Value. Este valor afecta los timers y el número de respuestas
QQIC	Código de intervalos entre queries. Si este código es menor que 128, entonces este es igual al intervalo entre queries. Si este código es más grande que o igual a 128, entonces este representa un valor de punto flotante (en mantisa y formato exponencial)
Number of sources [N]	Número de fuentes presentes en el query. Este número es distinto de cero para un query de fuente y grupo.
Source adress [1...N]	Dirección de la fuente

A continuación, en la Figura 7, se muestra el diagrama del formato de un paquete report con la tabla correspondiente a la descripción de los campos principales:

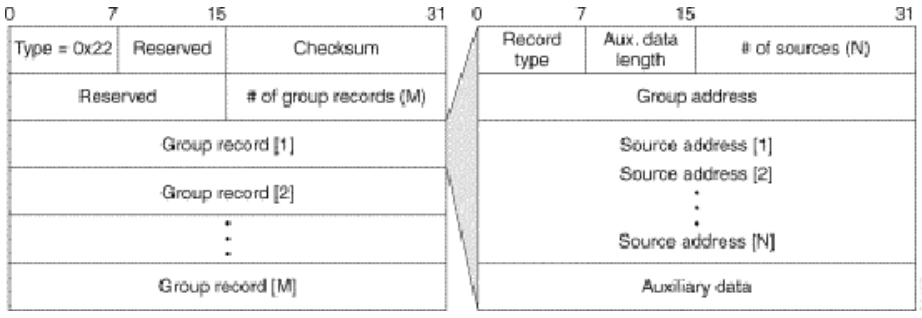


Figura 7

Field	Description
# de registros de grupo [M]	Número de registros de grupo presentes en el reporte.
Registro de grupo [1...M]	Bloque de campos conteniendo información concerniente al grupo de miembros emisores con un grupo multicast singular sobre la interfase desde la cual el reporte fue enviado.
Record Type	El tipo de registro de grupo
# of sources [N]	Número de fuentes presentes en el registro.
Source adress [1...N]	Dirección de la fuente

En IGMP v3 existen los siguientes tipos de mensajes:

- Versión 3 membership query
- Versión 3 membership report

IGMP V3 soporta aplicaciones que explícitamente señala fuentes desde las cuales se desea recibir el tráfico. Con IGMP v3, los miembros señalan como receptores a un grupo de host multicast de dos maneras:

1. INCLUDE MODE: El receptor anuncia a los miembros a un grupo de host y provee una lista de direcciones fuente (INCLUDE list) desde la cual se quiere recibir el tráfico.

2. EXCLUDE MODE: En este modo el receptor anuncia a los miembros a un grupo multicast y provee una lista de direcciones fuente (EXCLUDE list) desde la cual no se quiere recibir tráfico. El host recibirá tráfico sola mente desde las fuentes cuyas direcciones IP no figuren en la EXCLUDE list. Para recibir tráfico desde todas las fuentes, como la vieja conducta de IGMP v2, un host usa la designación de miembro en modo excluído con una lista de exclusión vacía.

En estos momentos, se puede pensar que el IP Multicast ha alcanzado un grado de madurez suficiente para su explotación comercial. Sin embargo, si realizamos un estudio más profundo, podemos observar que el IP Multicast tiene algunos aspectos, sobre todo relacionados con seguridad, que tienen que ser solucionados, de entre los cuales, cabe destacar el problema de la autenticación de usuarios.

Esto es así, porque la autenticación de usuarios es un elemento clave en cualquier campo en el que se piense utilizar IP Multicast: distribución de contenidos pay per view, tele-enseñanza,... de modo que se pueda controlar quién accede al sistema y se pueda evitar accesos no deseados.
La última versión de IGMP v3 permite filtrar con base en el origen de los datagramas, de modo que se puede utilizar para controlar quién se une a un grupo. Además, en mayo de 2000, B. Quinn propuso el Internet-Draft llamado Source Filters que describe cómo adaptar el protocolo Session Description Protocol (SDP) para expresar uno o más filtros en el origen de los datagramas en uno o varios destinos.

En los trabajos anteriores, la autenticación se realiza con base en la dirección IP de origen del datagrama. Este mecanismo de autenticación es muy poco fiable, debido a la facilidad de que una dirección IP se pueda falsificar (IP spoofing), de modo que proponemos un nuevo mecanismo de autenticación. Además, el hecho de que la dirección de origen no sea fiable implica que tampoco se pueda utilizar IPSec en nuestro propósito.

La idea clave es extender IGMPv3 para que transporte información de autenticación de usuario, de modo que un router de una red pueda decidir si considera los mensajes IGMPv3 Report o no. Para esto, se propone la utilización del Auxiliary Data Field del paquete IGMPv3 para transportar la información de autenticación, de modo que no es necesario introducir cambios en el protocolo IGMPv3.

No se definirá ni se tratará la autenticación de los usuarios, sino los mecanismos para que se pueda realizar esta autenticación. Así, si, por ejemplo, la organización se basa en una infraestructura de clave pública (PKI, Public Key Infraestructure), se puede pensar en utilizar este

campo para transportar la firma del paquete IGMP basada en la clave privada del usuario, de modo que el router tendría que consultar la clave pública del usuario en un servidor de directorio (LDAP, por ejemplo) para poder comprobar si la firma es correcta o no (y por tanto, comprobar si el usuario es quién dice ser) y tomar la decisión oportuna sobre ese mensaje.

Este Auxiliary Data Field debe transportar, no sólo la información de autenticación (por ejemplo, la firma basada en PKI), sino también información propia del usuario para que el router pueda saber el usuario al que pertenece el paquete. De modo que, los primeros bytes del Auxiliary Data Field se corresponderán con la información de autenticación y el resto será la información de usuario. La Figura 8, muestra cómo se añade la identificación de usuario y un valor de hash en el primer Group Record del Auxiliary Data Field cuando se utiliza MD5 para garantizar la autenticación de usuario.

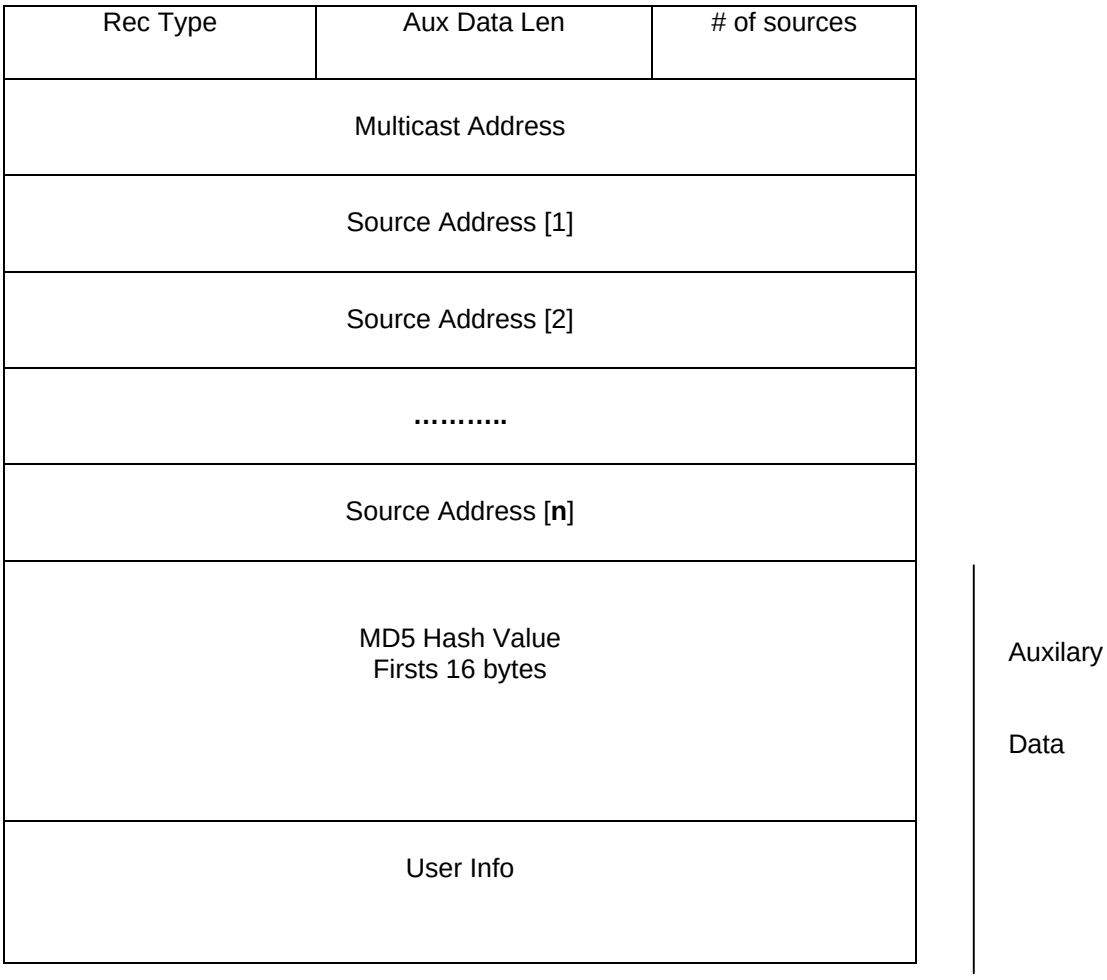


Figura 8

De esta manera, un router puede decidir si un usuario está autorizado o no para unirse a una sesión multicast. Por ejemplo, se puede utilizar este mecanismo para evitar ataques DoS en redes multicast, problema que no se puede resolver con ninguno de los métodos comentados anteriormente, en el caso de que por ejemplo, alguien de tu red decide conectarse a todos los grupos multicast activos. Ocurre el mismo problema cuando algún usuario de la red se pone de acuerdo con otra persona del exterior para inundar tu red cuando ambos se conectan a una misma sesión y el segundo inunda la red con un tráfico de gran ancho de banda. Nuestra propuesta evita este problema ya que el administrador puede decidir quién puede provocar qué flujos y quién no.

Este documento describe la utilización de IGMPv3 para proporcionar un cierto nivel de seguridad a redes multicast. Su principal objetivo es describir una extensión de IGMPv3, de modo que se pueda evitar el problema de ataques DoS y se pueda proporcionar un control de acceso de usuarios a la red multicast.

No obstante, no es más que el primer paso en el camino para conseguir una solución completa que ofrezca los servicios de seguridad típicos (autenticación, seguridad, integridad, etc.) en redes IP Multicast. De modo, que aún queda bastante trabajo por realizar. El siguiente paso consiste en implementar estas extensiones sobre alguna de las recientes implementaciones de IGMPv3, así como modificar el API de acceso a IGMPv3 para permitir el uso del Auxiliary Data Field.[4]

Ruteo en multicast

Las ventajas de la idea del IP multicast se hacen presentes cuando podemos extender el esquema de funcionamiento entre varias subredes, es decir, cuando los miembros de un determinado grupo multicast están distribuidos en varios segmentos de red distintos, interconectados a través de routers.

Para que el concepto multicast funcione, no basta con que los routers multicast conozcan, por medio del IGMP , qué equipos pertenecen a un determinado grupo multicast en los segmentos de red que este conecta, sino que deben saber tomar las decisiones necesarias para encaminar los

[4] **http://www.rediris.es/rediris/boletin/54-55/ponencia12.html - Uso de IGMPv3 para evitar ataques DoS en redes multicast- . F. Gómez Skarmeta,A. L. Mateo, P. M. Ruíz**

datagramas multicast entre dichas subredes, asegurando que los enviados por un determinado equipo lleguen a todos los miembros de cada grupo multicast, y procurar, por otro lado, que cada datagrama llegue a sus destinatarios sólo una vez (y, preferiblemente, por el camino más corto). Es decir, debe existir una determinada política de encaminamiento multicast, o dicho de otra forma, estos routers deben implementar un protocolo de encaminamiento (routing) multicast.

Un protocolo de encaminamiento multicast es el que se encarga de la construcción de los árboles de distribución (delivery trees) y habilitar la remisión (forwarding) de datagramas multicast.

La característica diferencial entre el encaminamiento unicast y el multicast, es que si un datagrama IP multicast es remitido hacia su origen, se podría producir un bucle de remisión, que podría dar lugar a una `avalancha' multicast.

Todos los protocolos de encaminamiento multicast hacen uso del protocolo IGMP para conocer la filiación de los equipos finales a cada determinado grupo multicast, pero difieren en la forma de intercambiar dicha información entre routers vecinos, así como en las técnicas empleadas en la construcción de los árboles de distribución.

En cuanto a los tipos de protocolos de encaminamiento podemos distinguir, del mismo modo que para el encaminamiento unicast, dos grandes familias:

- ✓ Protocolos **de distancia vectorial:** Basados en el algoritmo de "camino más corto" del Bellman-Ford, en el que cada nodo no conoce la arquitectura de la red sino que lo único que conoce son las rutas disponibles que puede alcanzar a través de sus vecinos. Cada uno de los nodos tiene una tabla que publicará para que los vecinos se enteren. Estas tablas muestran los nodos que pueden alcanzar y con qué costo. Cada nodo asigna un "peso" o métrica a cada ruta en función de los saltos necesarios para alcanzar a otro nodo. Su principal ventaja es su sencillez de operación y por ende, de implementación. Mientras que su mayor desventaja es su problema de escalabilidad. A medida que la red se hace mayor y más compleja, el algoritmo se vuelve menos eficiente y se produce un mayor consumo de ancho de banda en los enlaces por la diseminación de las tablas de encaminamiento. Por otro lado también es posible la formación de bucles de

encaminamiento (aunque existen implementaciones de este tipo de protocolo que evitan, en gran medida, este inconveniente).

- ✓ Protocolos **de estado del enlace:** Se basan en el concepto de un "mapa distribuido", es decir, que todos los nodos tienen una copia del mapa de la red, que se actualiza periódicamente. Se han desarrollado a partir de un algoritmo más eficiente que el de Bellman-Ford, propuesto por E.W. Dijkstra, llamado "el camino más corto primero" (shortest path first). Sin entrar en más detalles comentaremos que algunas de sus principales ventajas son: la rápida convergencia a la descripción real del estado de la red, la ausencia de creación de bucles, el soporte de métricas (costes asociados a un determinado enlace) múltiples, soporte de múltiples rutas a un mismo destino, etc.. Como contrapartida requieren mayor poder de procesamiento en los routers y son complejos de implementar y/o configurar.

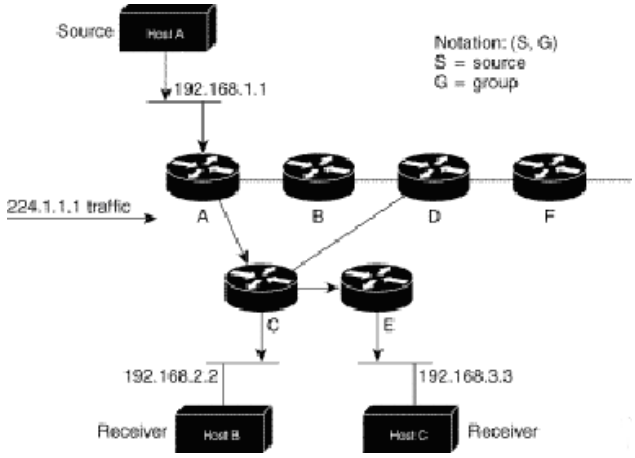
    - ❖ Árboles de distribución Multicast

Los routers con capacidad multicast crean árboles de distribución que controlan el camino que toma el tráfico IP multicast a través de la red, para distribuirlo a todos los receptores. Los dos tipos básicos de árboles de distribución multicast son: árboles fuente y árboles compartidos.

1. Árboles fuente

La forma más simple de árbol de distribución multicast es un árbol fuente con su raíz en la fuente y sus ramas formando un spanning tree a través de la red y hacia los receptores. Como estos árboles usan el camino más corto, son referenciados como shortest path tree (SPT)

La siguiente figura muestra un ejemplo de un SPT para el grupo 224.1.1.1 ruteado a la fuente, Host A y conectado hacia 2 receptores, Hosts B y C:



Fuente: Developing IP Multicast Networks – Volume I- Beau Williamson – 2000 Cisco Press – Página 36

La notación especial de (S,G) enumera un SPT cuando S es la dirección IP de la fuente y G la dirección del grupo multicast. Usando esta notación, el SPT para el ejemplo de la figura anterior puede ser (192.168.1.1 , 224.1.1.1)

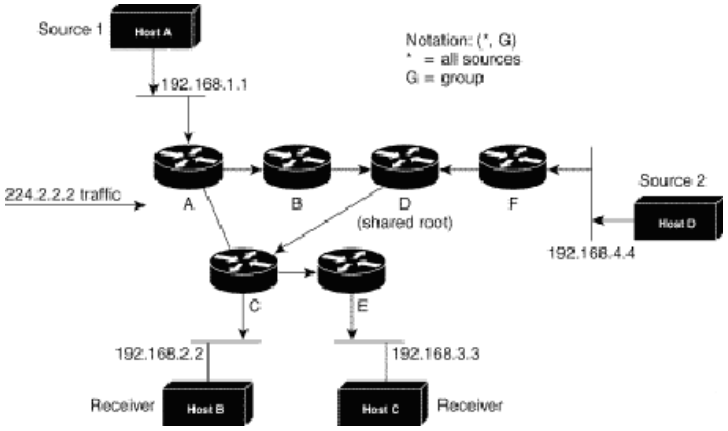
La notación (S,G) implica que existe un SPT separado para cada fuente individual y enviado a cada grupo. Por Ej. : Si el Host B está también enviando tráfico al grupo 224.1.1.1 y los hosts A y C son receptores, un SPT(S,G) separado existiría con una notación de (192.168.2.2 , 224.1.1.1)

2. Árboles compartidos

A diferencia de los árboles fuente, los árboles compartidos usan una raíz común localizada en algún punto de la red. Esta raíz compartida es llamada rendezvous point (RP).

La figura siguiente muestra un árbol compartido para el grupo 224.2.2.2 con la raíz localizada en el router D. Cuando se usa un árbol compartido, las fuentes deben enviar su tráfico hacia la raíz.

El tráfico es reenviado hacia abajo del árbol compartido extendiéndose hacia todos los receptores.



Fuente: Developing IP Multicast Networks – Volume I- Beau Williamson – 2000 Cisco Press – Página 37

En este ejemplo, el tráfico multicast de las fuentes, hosts A y D, viaja a la raíz (router D) y entonces baja por el árbol compartido hacia los dos receptores, Hosts B y C.

Como todas las fuentes en el grupo multicast usan un árbol compartido común, una notación comodín que se escribe (*,G) representa el árbol. En este caso, el * significa todas las fuentes y G representa el grupo multicast. Por lo tanto, el árbol compartido que se muestra arriba se escribe (*, 224.2.2.2).

3. Árboles fuentes vs. Árboles compartidos

Ambos árboles SPT(Shortest path trees) son loop-free. Los mensajes son replicados solamente hacia las ramas del árbol.

Los miembros de grupos multicast pueden unirse (join) o abandonar el mismo (leave) en cualquier momento. Cuando todos los receptores activos sobre una rama particular paran el requerimiento de tráfico para un grupo particular de multicast, los routers cortan las ramas de distribución del árbol y el reenvío de tráfico hacia abajo. Si un receptor sobre la rama se activa y requiere tráfico multicast, el router, dinámicamente modifica la distribución del árbol y comienza a reenviar tráfico nuevamente.

SPT tiene la ventaja de crear el camino óptimo entre la fuente y los receptores. Esta ventaja garantiza un monto mínimo de latencia de red para reenvío de tráfico multicast. Sin embargo, esta optimización tiene un costo: los routers deben mantener caminos de información para cada fuente. En una red con miles de fuentes y miles de grupos este overhead puede rápidamente volverse un recurso sobre los routers. La consumición de memoria desde el tamaño de la tabla de ruteo multicast es un factor que los diseñadores de redes deben considerar.

Los árboles compartidos tienen la ventaja de requerir el mínimo monto de estado en cada router. Esta ventaja disminuye los requerimientos de memoria innecesarios para una red que permite sólo árboles compartidos. La desventaja de los árboles compartidos es que, bajo ciertas circunstancias los caminos entre la fuente y los receptores pueden no ser los óptimos porque introducen alguna latencia en la distribución de paquetes. Los diseñadores de redes deben considerar cuidadosamente el lugar del RP cuando implementan un ambiente de árboles compartidos únicamente.

Existen varios estándares para ruteo de tráfico multicast:

El RFC 1075 define el protocolo DVMRP (Distance Vector Multicast Routing Protocol). DVMRP usa la técnica conocida como Reverse Path Forwarding. Emplea broadcast para identificar todas las estaciones que forman un grupo multicast. Si se conecta un router a un conjunto de LANs que no quieren recibir un grupo multicast particular, el equipo devuelve un mensaje al árbol de distribución para detener los paquetes en camino hacia donde no haya miembros del grupo. DVMRP usa cuenta de saltos (hop) para seleccionar la ruta. En general, la utilización de este protocolo se considera inadecuada para topologías de red que cambian rápidamente, puesto que la información de encaminamiento se suministra muy lentamente.

El RFC 1584 define el protocolo MOSPF(Multicast Open Shortest Path First). Es una extensión al protocolo de encaminamiento OSPF que permite soporte multicast (RFC 1583). Este último es un protocolo del tipo `estado del enlace', que permite un cálculo rápido de las rutas con un mínimo de intercambio de información entre routers. El MOSPF es una simple extensión al anterior, que incluye la posibilidad del encaminamiento del tráfico multicast. La ventaja de este esquema es que el protocolo de encaminamiento multicast se aprovecha del protocolo unicast, y no tiene que

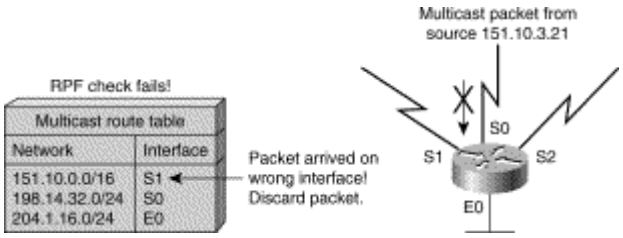
construir sus propias tablas de encaminamiento independientemente. El MOSPF sólo añade la información de origen y grupo multicast a los mensajes de estado del enlace, con los que el OSPF crea su mapa de la topología de red. El disponer de una descripción del estado del enlace con la información de filiación de miembros a los distintos grupos multicast, permite la construcción de los árboles de envío de camino más corto en la memoria de los mrouter. Vale aclarar que mrouter es un término utilizado habitualmente en reemplazo de Multicast Router, es decir un router que soporta Protocolos Multicasting.

De lo expresado anteriormente se evidencia que, a diferencia de DVMRP, MOSPF no necesita enviar un broadcast para identificar todas las estaciones que forman un grupo multicast. Por otro lado, la construcción del diagrama de distribución, se realiza "bajo demanda" cuando un mrouter recibe el primer datagrama para una determinada pareja {fuente, grupo}. Este esquema presenta la desventaja de que puede sobrecargar la CPU del router en los casos en los que varias parejas {fuente,grupo} aparecen al mismo tiempo, o cuando concurran muchas circunstancias que fuercen la reconstrucción de las cachés de remisión (forwarding caches).

PIM (Protocol Independent Multicast) obtiene su nombre de los hechos: es un protocolo de ruteo IP independiente. Aunque es nombrado como un protocolo de ruteo multicast, utliza la tabla de ruteo unicast para implementar la función Reverse Path Forwarding. RPF es un concepto fundamental que permite a los routers reenviar correctamente el tráfico multicast hacia la fronda del árbol de distribución (downstream). RPF hace uso de la tabla de ruteo unicast existente para determinar los downstreams vecinos. Un router reenviará un paquete multicast solamente si éste es recibido sobre la interfaz “upstream”. Este chequeo RPF ayuda a garantizar que la distribución del árbol esté libre de “loops” o bucles. Cuando un paquete multicast llega a un router, éste desarrolla un chequeo RPF. Si este chequeo es exitoso, el paquete es reenviado. De lo contrario se descarta. El procedimiento es el siguiente:

- ✓ El router busca la dirección de la fuente en la tabla de ruteo unicast para determinar si el paquete ha llegado a la interfaz, esto es, sobre el camino de reversa a la fuente.
- ✓ Si el paquete ha llegado a la interfaz volviendo desde la fuente, el chequeo RPF es exitoso y el paquete es reenviado.

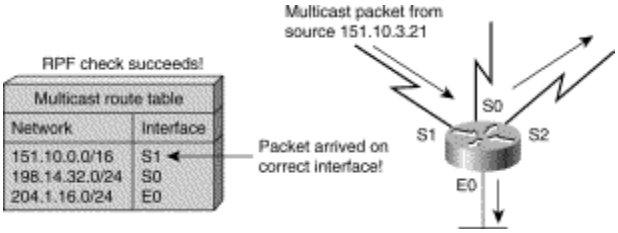
A continuación se muestra un chequeo RPF fallido:



Fuente: Developing IP Multicast Networks – Volume I- Beau Williamson – 2000 Cisco Press

Un paquete multicast desde la fuente 151.10.3.21 es recibido sobre la interfaz serial 0 (S0). Un chequeo de la tabla de ruteo unicast muestra que S1 es la interfaz del router que podría ser usada para reenviar información unicast a la dirección 151.10.3.21. Como el paquete ha llegado sobre la interfaz S0, el paquete es descartado.

A continuación se muestra un chequeo RPF exitoso:



Fuente: Developing IP Multicast Networks – Volume I- Beau Williamson – 2000 Cisco Press

En este caso, el paquete multicast ha llegado a la interfaz S1. El router referencia la tabla de ruteo unicast y encuentra que S1 es la interfaz correcta. El chequeo RPF es exitoso y el paquete es reenviado.

A diferencia de otros protocolos de ruteo, PIM no envía y recibe actualizaciones de ruteo multicast entre routers. Los modos de reenvío son descriptos a continuación:

- PIM Dense Mode (PIM-DM)

PIM-Dense Mode usa un modelo de “empuje” para flujo multicast hacia cada esquina de la red. Este modelo de empuje es un método de fuerza bruta para distribuir información hacia los

receptores pero en ciertas aplicaciones puede ser un mecanismo eficiente si los receptores activos están sobre cada subred dentro de la red.

PIM-Dense Mode inicialmente insufla tráfico multicast a través de la red. Los routers que no tienen vecinos “downstream” cortan el tráfico innecesario. Este proceso se repite a intervalos de tiempo regulares.

El flujo y el mecanismo de corte es la forma mediante la cual los routers acumulan estados de información recibiendo streams. Estos streams contienen la fuente y el grupo de información, por lo tanto los routers downstream pueden construir su tabla de reenvío multicast.

PIM-Dense Mode soporta solamente árboles fuente o entradas (S,G). No puede ser usado para construir un árbol de distribución compartido.

- PIM Sparse Mode (PIM-SM)

PIM Sparse Mode usa un modelo que “tira” para lograr la distribución de tráfico multicast. Solamente segmentos de red con receptores activos que han explícitamente requerido la información reenviarán el tráfico. PIM Sparse Mode fue originariamente descripto en el RFC 2362.

PIM Sparse Mode usa un árbol compartido para distribuir información sobre fuentes activas. Dependiendo de las opciones de configuración, una parte del tráfico puede quedar remanente en el árbol compartido o switcheado sobre un árbol de distribución optimizado.

PIM Sparse Mode requiere el uso de rendezvous point (RP). Los RP deben ser administrativamente configurados en la red.

Las fuentes se registran con los RP y los datos son enviados hacia la parte baja del árbol compartido, es decir, hacia los receptores. Si el árbol compartido no es un camino óptimo entre la fuente y el receptor, los routers dinámicamente, crean un árbol fuente y paran el tráfico desde el flujo descendente del árbol compartido.

Esta conducta es la conducta por default en routers Cisco IOS: Los administradores de red pueden forzar tráfico sobre le árbol compartido utilizando la opción de configuración específica.

PIM Sparse Mode se ajusta a redes de cualquier tamaño, incluyendo algunas con links WAN. El mecanismo de unión prevendrá el tráfico indeseado.[5]

Cuadro comparativo para Dense y Sparse Mode:

Dense Mode	**Sparse Mode**
Arbol basado en el origen	Árbol compartido llamado RP (Rendezvous point)
Hay un árbol (TBT) por cada origen	Hay un RP (SPT) por cada grupo multicast
Menor delay porque existe un árbol por cada origen	Delay más alto por tener un árbol compartido, lo que no asegura la mejor ruta a los destinos.
Mayor uso de la memoria del router porque se especifíca una ruta por cada destino	Mejor uso de la memoria del router porque solo registra un RP por cada árbol compartido

- Sparse-Dense Mode

La Firma Cisco System ha implementado una alternativa para elegir entre dense mode y sparse mode en la interfaz del router. Esta alternativa surge de la necesidad del cambio de paradigma para reenvío de tráfico multicast a través de PIM durante su desarrollo.

Así se determinó que cambiando el modo sparse o dense sobre la base de un grupo era más eficiente que sobre una base de interfaz de router.

La configuración puede ser realizada por los administradores y permite a grupos individuales correr tanto en modo sparse como dense, dependiendo de la información disponible sobre el RP del grupo. Si el router “aprende” la información RP para un grupo en particular, éste es tratado como un grupo en modo sparse. De lo contrario, es tratado en modo dense.

[5] http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/ipmulti.htm#xtocid25
Cisco Systems se ha convertido en el líder mundial en redes para Internet.

- Bidirectional PIM (Bidir-PIM)

Bidirectional PIM (Bidir-PIM) es un avance del protocolo PIM que fue diseñado para comunicaciones muchos a muchos dentro de un dominio PIM individual. Los grupos multicast en modo bidireccional pueden acceder a un número arbitrario de fuentes sin ningún overhead adicional.

En modo bidireccional, el tráfico es ruteado solamente a lo largo del un árbol compartido bidireccional que es fijado (su raíz) en el RP del grupo. En Bidir –PIM la dirección IP del RP actúa como una llave para tener todos los routers bajo una topología libre de “loops” o bucles con spanning tree, referenciada a esa dirección IP. Esta dirección IP no necesita ser una dirección de router pero puede ser cualquier dirección no asignada sobre una red.

Bidir-PIM es una derivación que surge desde mecanismos de PIM en modo sparse (PIM-Sparse Mode) y comparte algunas de las operaciones de los árboles compartidos. Bidir-PIM tiene incondicional reenvío del tráfico de la fuente hacia el RP “upstream” en el árbol compartido pero no registra procesos para fuentes como PIM-Sparse Mode.

Estas modificaciones son necesarias y suficientes para permitir reenvío de tráfico en todos los routers basados sobre las entradas de ruteo multicast (*,G). Este esquema elimina estados de fuentes específicos y permite acceder a mayor capacidad, hacia un número arbitrario de fuentes.

Capítulo 2: Arquitectura

¿Qué es PGM?

Pragmatic General Multicast es un protocolo de transporte multicast para aplicaciones que requieren datos provistos por una sola fuente a muchos receptores.

PGM trabaja sobre un servicio de datagrama best effort como IP multicast.

El RFC correspondiente a este protocolo es el número 3208 y se encuentra categorizado como experimental.

PGM garantiza que el receptor perteneciente a un grupo puede recibir todos los paquetes de transmisión o reparación o bien puede detectar paquetes perdidos irrecuperables. Obtiene escalabilidad a través de una vía jerárquica, Forward error correction (FEC), NAK elimination y NAK supresión (política de acknowledge negativos para eliminación y supresión).

En el curso normal de transferencia de datos una fuente envía bajo modalidad multicast paquetes de datos secuenciados (ODATA) y los receptores envían bajo modalidad unicast, ACK negativos selectivos (NAKs) por cada paquete que haya sido detectado como perdido dentro de la secuencia esperada. Los elementos de red reenvían estos NAKs salto a salto hasta la fuente y confirman enviando bajo modalidad multicast una confirmación del NAK recibido (NCF). Los elementos de red envían el NAK en sentido ascendente del camino de distribución de la fuente la cual originó el paquete hasta su potencial recepción. Los datos reparados (RDATA) pueden ser provistos tanto por la fuente misma como por un Reparador Local Desigando (DLR)en respuesta a un NAK. Analizaremos, a continuación, la confiabilidad en el transporte.

Llamamos transporte confiable como una propuesta end-to-end donde las fallas en los escenarios intermedios deben ser solucionadas. El protocolo de transporte más conocido de este tipo es TCP (Transport Control Protocol). La implementación de un escenario multicast confiable, debería enviar mensajes utilizando IP multicast y luego simular el comportamiento de TCP en la forma en que éste recibe los ACK de los correspondientes mensajes.

Sin embargo, sería imposible conseguir un protocolo confiable multicast basado en ACK end-to-end ya que no permitiría escalabilidad. La fuente recibiría una enorme cantidad de mensajes, producto de las confirmaciones a cada una de las recepciones realizadas. Esto provocaría una implosión de ACK. Explicaremos este concepto a continuación:

Mientras el número de receptores crece, el tráfico de mensajes de retorno a la fuente saturará eventualmente tanto la fuente como los links intermedios, produciendo un colapso. Otro inconveniente, aunque de menor importancia que el referido anteriormente, es que la memoria debe ser proporcional al número de receptores para mantener el almacenamiento del estado de los mismos.

En cambio, mediante la política de supresión de NAK's, no se repetirán NAK's correspondientes a paquetes iguales no recibidos. Aunque no sea enviado este NAK, los receptores actuarán como que sí lo fue. Para la fuente, es indistinto la llegada de uno o varios NAK's referidos al mismo paquete. Los receptores conocen de los otros NAK´s enviados mediante la confirmación de la recepción multicast del NAK que se efectúa a través de un paquete NCF(NAK Confirmation) que es enviado en modalidad multicast por la fuente, como se explicara arriba. Existe un retardo variable para el envío de los NAK´s que, conjuntamente con la supresión de los NAK's previene el suceso de implosión.

Sin embargo, para asegurar que la implosión no ocurra, los retardos anteriores al NAK deberían incrementarse en la misma medida que los receptores. Es importante hacer notar que retardos excesivamente largos podrían provocar muchos problemas como una enorme ventana de transmisión y la posibilidad de que un receptor se vea imposibilitado a enviar un NAK antes de que la sesión haya terminado.

Otra forma para mejorar la escalibilidad es la vìa jerárquica. Se construye un árbol para una sesión de multicast confiable integrada tanto por receptores como por nodos intermedios especiales. Los mensajes NAK o ACK son solamente enviados desde un nodo hacia el inmediato superior en el árbol jerárquico. Dicho nodo agrega o elimina la información (mensaje correspondiente) antes de enviarla hacia las ramas superiores del árbol. Si muchos nodos inferiores perdieron un determinado paquete de información, sólo un NAK será enviado hacia la parte superior del árbol. Ésta es la supresión de NAK's correspondiente a un mismo paquete.

Otra posibilidad en el uso de la vía jerárquica es limitar el envío de reparaciones de tal forma que sólo sean enviados a aquellos subárboles que contengan receptores necesitados de la reparación.

Una dificultad que se presenta con el crecimiento de la estructura, es que la pérdida de paquetes será proporcional al número de receptores. Por ejemplo: con 1.000.000 de receptores y una probabilidad de pérdida aleatoria del 0,01%, las posibilidades de todos los nodos de recibir un paquete enviado, es menor a 10^{-43} [6] . Esto constituye, virtualmente, la certeza de que algún nodo de la red perderá el paquete. De la misma manera, para que todos los receptores reciban todos los paquetes, cada paquete debe ser enviado por lo menos dos veces, disminuyendo el uso del ancho de banda a la mitad.

Para evitar esto, la política Forward Error Correction (FEC) debería permitir la corrección de las diferentes pérdidas a través de los diferentes receptores. Por ejemplo: si un receptor pierde su paquete 1 y otro pierde su paquete 2, una sola reparación de paquetes conteniendo la paridad de paquetes de 1 a 7, va a brindar la posibilidad de ambos receptores para reparar su pérdida.

PGM utiliza un esquema híbrido incluyendo:

- ◊ Supresión
- ◊ NAK elimination
- ◊ Constraint Forwarding: forzar el flujo ascendente de información en el árbol
- ◊ FEC (Forward Error Correction) para alcanzar la escalabilidad.

La jerarquía se construye con base en elementos de red capaces de transmitir a través de este protocolo. Estos elementos son routers típicos diseñados para soportar PGM sobre IP multicast. PGM está diseñado para operar con menos eficacia sobre algunos elementos de red que no están construidos para soportar este protocolo.

Cuando los elementos de red diseñados para soportar PGM son escasos, crece el árbol de distribución PGM, otorgándole la función de supresión y Forward Error Correction con un rol más crítico en cuanto a escalabilidad se refiere.

[6] The PGM Reliable Multicast Protocol – Jim Gemmell-Todd Mongotmery-Tony Speakman-Nidhi Bhaskar-Jon Crowcroft

PGM no requiere que el receptor realice transmisiones multicast. Sólo requiere transmisiones multicast del emisor al receptor. PGM también torna eficiente el uso del back channel del ancho de banda, produciendo mejores resultados para redes asimétricas, que tiene alta capacidad de canal del emisor hacia los receptores pero tienen un back channel restringido de los receptores al emisor.

PGM no se utiliza en aplicaciones que dependan de la política de envío de ACK a grupos conocidos de receptores. Tampoco se utiliza entre diferentes fuentes.

Por otra parte, PGM permite que los receptores puedan unirse al grupo o dejarlo cuando deseen en cualquier momento, brindándoles confiabilidad solamente en la actual ventana de transmisión. Se utiliza más eficazmente para aplicaciones que soportan algunos niveles de recuperación de aplicaciones en el evento correspondiente a pérdidas irrecuperables.

Esto no significa que los paquetes perdidos irrecuperables se van a recuperar con PGM. La transmisión confiable se espera solamente si el emisor no avanza en la ventana de transmisión muy agresivamente. Es una característica de PGM multicast que una aplicación puede elegir:

- En vez de numerar, continuar con el envío de nueva información si se determina que la información anterior ya fue enviada con éxito.
- Determinar la importancia del envío de información para satisfacer a los nuevos receptores que se han unido al grupo.

Arquitectura

El árbol de PGM se encuentra acotado utilizando los elementos de red existentes que soportan este protocolo en dicho árbol. Una fuente de PGM multicast determina una secuencia de paquetes de datos hacia los receptores (ODATA-Original Data). Cuando los receptores detectan paquetes que fueron perdidos en la secuencia, envían en modalidad unicast un NAK a su nodo inmediatamente superior en el árbol constituido. Dicho nodo confirma la recepción del NAK proveniente de todos sus receptores y/o nodos asociados con una confirmación de NAK a través de modalidad de envío multicast.

Las reparaciones son generadas tanto por una fuente como por un reparador local designado (DLR – Designated Local Repairer) en respuesta a un NAK. Esto se debe a que los elementos de red de PGM no almacenan paquetes ODATA o proveen reparaciones. Una reparación consiste en un paquete reenviado o bien en un paquete FEC, dependiendo de los parámetros de la sesión. Antes de enviar un NAK los receptores implementan un back-off aleatorio y suprimen el NAK si ya se ha recibido el NCF correspondiente, los datos o los datos reparados.

Jerarquía

Para establecer la jerarquía de PGM, los emisores emiten periódicamente un SPM (Source Path Message) que también sirve para otros propósitos que discutiremos más adelante.

El paquete SPM contiene la dirección del nodo inmediato superior del que proviene. Los elementos de red reemplazan o reubican esta dirección con la suya propia cuando envían un SPM de tal forma que los subnodos conozcan a su nodo inmediatamente superior.

Esta técnica le permite a los elementos de red que no soportan el protocolo PGM, operar transparentemente entre los nodos PGM. Si el ruteo multicast cambia o si, los elementos de red de PGM, comienzan a realizar operaciones fuera del entorno PGM, entonces SPM va a contribuir a que el árbol de PGM sea actualizado.

Por lo tanto, el árbol actualizado de PGM se superpondrá al árbol primitivo. Si todos los elementos de red soportan PGM, los árboles serán idénticos. Todos los paquetes multicast serán enviados con la dirección de fuente PGM aunque sean originados por un elemento de red. Esto asegura que viajarán a través del mismo camino sin importar que los paquetes provengan de la fuente PGM.

Si utilizamos el mismo árbol, evitaremos crear elementos de red adicionales y por lo tanto eliminaremos la posiblidad de que existan problemas tales como el encuentro de diferentes características de pérdida entre dos árboles. Para minimizar las pérdidas de NAK, PGM definirá una capa de red hop-by-hop producidas por el envío de NAK.

Después de detectar paquetes perdidos, un receptor enviará bajo modalidad unicast un paquete NAK repetidamente a su nodo inmediato superior (que soporte PGM) hasta que reciba un NCF por su paquete.

El elemento de red no eliminará un NAK, lo reenviará en modalidad unicast a su nodo inmediato superior y esta operación se repetirá hasta que un NCF sea recibido.

Sea o no satisfactoriamente enviado, el NAK será descartado por el elemento de red. Pero la confiabilidad en el envío del NAK no puede ser garantizada. La responsabilidad de regenerar el NAK sin confirmar recae en el receptor, quien debe ajustarse al principio end-to-end .

Los elementos de red también implementan una política de anticipación de NAK's. Existirán recepciones de NCF que no corresponderán a un NAK saliente. Si el NCF proviene de un elemento de red inmediato superior, este recordará el NCF con anticipación a que su nodo inmediatamente inferior pueda enviar un nuevo NAK sobre el mismo paquete. Si esto sucede, el NAK será confirmado sin otra acción.

La acción de anticipación de NAK, conjuntamente con la de eliminación de NAK´s, permiten implementar debidamente la política de que sólo un NAK debe enviarse hacia el nodo de red inmediatamente superior a los receptores, cumpliendo así con las especificaciones marcadas en la política de NAK.

Los mensajes NCF son enviados en modalidad multicast. Sin embargo, no son propagados por los elementos de red PGM, por que actúan bajo política de confirmación hop-by-hop.

Los nodos PGM inmediatamente superiores generan un NCF para cada NAK sin importar que el NAK haya sido previamente confirmado.

La figura que se encuentra a continuación muestra la secuencia de un escenario NAK/NCF donde ambos PGM y non PGM routers están presentes. Todos los envíos multicast originados por los elementos de red PGM son generados como si fueran emitidos por la fuente misma. El router non PGM los reenviará al mismo árbol de multicast utilizado para la data de multicast proveniente del emisor. Este escenario comienza cuando el receptor detecta una pérdida en la esquina más alejada del árbol:

(1) El receptor envía bajo modalidad unicast un NAK a su nodo inmediato superior de PGM.

(2) Dicho nodo envía bajo modalidad multicast un mensaje de confirmación al NAK correspondiente, suprimiendo cualquier NAK similar proveniente de otro receptor.

(3) El router unicastea un NAK a su nodo superior

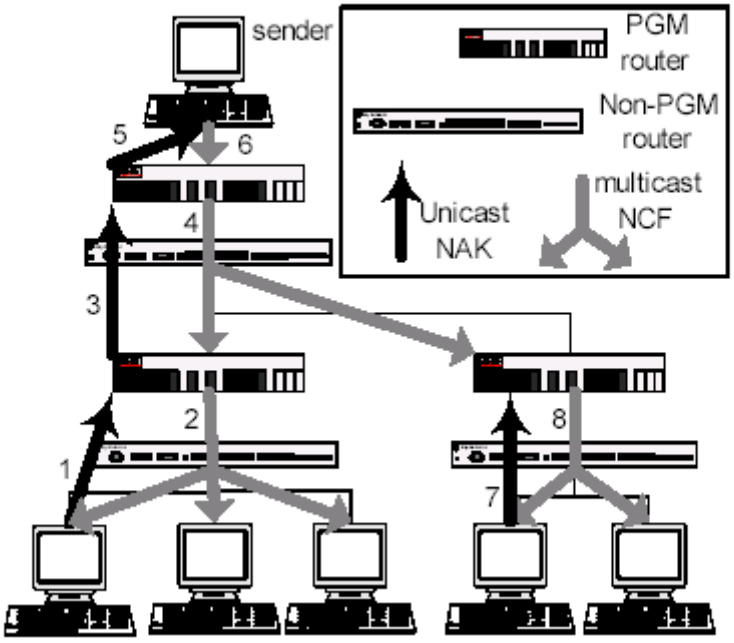
(4) El nodo superior enviará bajo modalidad multicast un NCF.

(5) El router PGM sobre la derecha almacenará el NCF para futuras referencias.

(6) Un NAK unicast es entonces enviado al emisor el cual responderá con un NCF multicast

(7) Luego de un tiempo otro receptor detecta la pérdida del mismo paquete y enviará bajo modalidad unicast un NAK hacia el primer nodo PGM.

(8) Este nodo, a su vez, enviará bajo modalidad multicast un NCF.



Los elementos de red ni siquiera intentan asegurar que los mensajes RDATA (Repair Data) son recibidos en respuesta a un NAK. Queda a consideración de los receptores reexpedir el NAK siguiendo el principio end-to-end. Los elementos de red, simplemente, descartan el estado del NAK después de un determinado intervalo de tiempo (time out) que ha sido establecido para

beneficiar el mantenimiento del estado de reparación en la red respondiendo a los cambios en el ruteo.

Para poder enviar las reparaciones, los elementos de red utilizarán interfaces por las cuales hayan recibido un NAK correspondiente a esa reparación. Esto procedimiento se denomina Constraint Repair Forwarding. Las reparaciones son enviadas a subárboles que contengan receptores que necesiten dicha reparación. Debe notarse que el esquema de PGM de propagación de NAKs hacia la parte superior del árbol, sigue la misma secuencia que las redes PGM en las cuales la ODATA ha sido enviada, pero en reversa. Esto le permite al correcto estado del NAK, ser establecido en el elemento de red para soportar el proceso de Constraint Repair Forwarding.

SPM, NCF y RDATA necesitan especial tratamiento por los elementos de red PGM. Una obvia forma de descubrir estos paquetes, es examinar cada uno en la red exclusiva para transporte del protocolo PGM. Sin embargo, la carga de examinar cada paquete es imposible. En su lugar, SPM, NCF y RDATA son transmitidos con la opción de router IP. Dicha opción le permite a los elementos de la red una indicación de nivel de red en dónde un paquete debería ser extraído desde el switcheo IP para un procesamiento más detallado.

Los paquetes de información original son enviados como cualquier otro paquete multicast sin ningún tipo de atención especial del router.

Supresión de NAK's

Los receptores agregan una demora correspondiente a un intervalo de tiempo aleatorio antes de enviar un NAK. Si se produce el encuentro entre un NAK y su correspondiente NCF, el RDATA u ODATA es recibido durante el proceso de retardo. El receptor suprime el NAK. Después de enviar o suprimir un NAK, el receptor espera por un NCF hasta que exista un time out. Luego de recibir un NCF, dicho receptor espera por el ODATA o el RDATA para reparar su pérdida.

Si el receptor se encuentra durante el time out, esperando por un NCF o bien por un paquete reparado luego de un NCF, comienza el envío de un nuevo NAK luego de un retardo aleatorio con supresión. El intervalo de retardo base es especificado en el SPM, lo cual le permite al

nodo superior PGM ajustar los retardos en base a la experiencia en dicha sesión y es estimado según el número de nodos inferiores. El intervalo de retardo actual utilizado por el receptor es derivado de un intervalo base, pero debería ser incrementado basándonos en un número de tiempo fuera para las reparaciones de NCF que ya han sucedido. Este intervalo también está incrementado si el NAK se utiliza para más de un paquete de paridad.

PGM soporta también las opciones de multicast de un NAK con TTL=1 (Time to live) limitando de esta forma la modalidad multicast a la red local LAN. Conjuntamente se envía un paquete unicast NAK hacia un nodo superior. Otros nodos reciben el NAK enviado bajo modalidad multicast y suprimen sus NAK's similares. Esta opción tiene lugar para lograr mejorar y acelerar el proceso de supresión en la LAN local en el caso de que los nodos inmediatos superiores de PGM estén ubicados en niveles superiores alejados en el árbol.

Forward Error Correction

PGM soporta level-packet Reed-Solomon. Los códigos de bloque lineal son utilizados para generar los paquetes de paridad h provenientes de los paquetes de datos originales k de tal forma que cualquier k del total de paquetes (h+k) deberían ser decodificados para adquirir los paquetes k originales. Dichos paquetes (originales k) son referenciados a un grupo de transmisión.

FEC debería ser proactivo, esto es, que los paquetes FEC deberían ser enviados en anticipación a posibles pérdidas sin esperar ninguna respuesta de los receptores.

FEC también debería ser un paquete bajo demanda (on demand) generado por el pedido de los receptores. Para obtener FEC bajo demanda, un receptor genera una paridad NAK, la cual indica el número de paquetes de paridad solicitados por cualquier grupo de transmisión.

El FEC proactivo y el FEC "on demand " son opcionales y pueden ser combinados a consideración del emisor. Una paridad NAK suprime otro NAK para el mismo grupo de transmisión si éste solicita un mayor o igual número de paquetes de paridad.

El uso de FEC provee un importante número de ventajas:

1. Retransmisión eficiente: un mensaje de paridad puede reparar una importante cantidad de mensajes perdidos.

2. Una más eficiente supresión: un NAK resultante de la pérdida de un paquete puede suprimir el NAK para otros paquetes, si ellos están en el mismo grupo de transmisión.

3. Una utilización eficiente del ancho de banda por parte de los NAK: por ejemplo, es más simple representar “30 paquetes perdidos del grupo 999” que listar esos 30 paquetes.

Ventana de Transmisión

En una transferencia unicast es obvio que el desplazamiento de la ventana de transferencia del emisor avanzaría hasta el punto en el cual todos los paquetes anteriores hayan sido confirmados mediante un ACK. Para PGM, sin ACK’s, esto no es viable.

PGM permite que el desplazamiento de la ventana de transmisión sea arbitrario y es administrado por el emisor. Esto a su vez permite a la aplicación determinar que la información no se haya viciado y los paquetes informados hayan sido efectivamente entregados y confirmada su entrega.

También permite decidir si ha tenido suficiente confidencialidad por un período de silencio NAK. Por otra parte asegura el correcto avance de la ventana de transmisión.

Sin embargo, mientras algunas aplicaciones desearían esta completa flexibilidad en la ventana de transmisión descripta más arriba, otras no desearían incrementar la carga del manejo de la ventana.

Algunas estrategias simples y fácilmente aplicables soportadas en desarrollos API incluyen las ventanas deslizables en tiempo o tamaño. Mientras el tiempo avanza, mantiene el paquete en la ventana de transmisión por un intervalo fijo de tiempo. La ventana contiene cualquier paquete enviado en los últimos 5 segundos. Si la técnica se orienta al tamaño, la ventana de transmisión aumentará el suyo y avanzará deslizándose cuanto sea necesario.

Reparadores locales: DLRs (Designated Local Repairer)

Además de tener la fuente proveedora de reparaciones, PGM soporta reparadores designados locales (DLRs) para reparar pérdidas. DLRs son nodos del tipo hosts y no elementos de red. Un DLR notifica su presencia con un mensaje multicast causando subsiguientes NAKs para ser redireccionados al DLR más que a la fuente. Los DLRs reciben y alojan en un espacio cache determinado todos los paquetes de la ventana de transmisión. Luego emiten NCFs y reparan acorde al envío primitivo del emisor.

Es obvio como un DLR que está en niveles superiores del árbol se auto inserta en el lugar del emisor.

Un escenario menos obvio que también es soportado por PGM es la ubicación de los DLRs fuera de la estructura del árbol Ello son denominados “off-tree” porque eventualmente se encuentran en un nivel más bajo del árbol desde el punto de pérdida. Ellos pueden pertenecer a un sub-árbol que no fue afectado por la pérdida. Un DLR “of-tree” responde a un NAK por envío multicast de un NCF y los datos reparados a su nivel superior. Este nivel superior los envía hacia los niveles inferiores.

Los árboles DLRs of-tree son descubiertos vía el mecanismo de polling PGM que se tratará en la próxima sección.

Polling

El retardo aleatorio precedente al NAK debería ser proporcional al número de integrantes del grupo PGM. Esto previene la implosión a medida que crece la proporción. A menudo una visión general es tan importante como analizar el decrecimiento de la proporción mencionada arriba. Esto es, reduciendo las tardanzas a medida que los integrantes del grupo decrecen. Entonces PGM no tiene innecesariamente una perfomance inactiva para los pequeños grupos. PGM

soporta Polling para permitir a un nivel superior del árbol, estimar el número de sus nodos inferiores.

Basados en los resultados de la aplicación del polling, y experimentando con la sesión, por ejemplo cuando algún nivel superior ha sido agobiado por una sucesión de NAK, los niveles superiores utilizan un espacio en los mensajes SPM para indicarle a los niveles inferiores el intervalo de retardo aleatorio.

Un poll es ejecutado enviando un paquete multicast POLL. Estos paquetes POLL no son reenviados por elementos de red PGM. Al igual que los paquetes NCF, los POLL siempre tienen una fuente de dirección IP de la fuente PGM de tal forma que usarán el mismo árbol multicast.

Cada paquete POLL indica una probabilidad con la cual un nodo inferior debería responder al POLL con un mensaje de respuesta de POLL unicast. También contiene un intervalo back-off aleatorio en el cual el subnodo debe observar antes de responder. Además cada POLL se procesa en un número de vueltas. Solamente un subnodo que fue parte de la primera vuelta de un POLL puede responder a las siguientes vueltas del POLL.

La probabilidad de respuesta combinada a unas vueltas numeradas puede ser usada para evitar la implosión en caso de un rápido y masivo crecimiento en la población de los sub-nodos. La primer vuelta de un POLL puede ser estipulada con una muy baja probabilidad de respuesta. Por lo tanto, sin importar que existan millones de subnodos, la implosión no ocurrirá.

Las subsecuentes vueltas pueden incrementar la probabilidad hasta que la suficiente cantidad de respuestas sea obtenida para llegar a un estimativo y razonable número de subnodos. Si muchos subnodos se unen a la sesión luego de que la probabilidad de respuesta haya crecido, ellos no responderán al POLL (sin presencia en la primera ronda) y la implosión será evitada.

Las sesiones que deseen incrementar su protección contra implosiones pueden utilizar un paquete opcional (NAK_BO_IVL_SQN) para asegurar que solamente los receptores que han formado parte del mensaje POLL están capacitados para emitir NAK.

Esto le brinda a los nodos superiores la seguridad de conocer que el intervalo back-off del NAK es reflejado por el número de sub-nodos. Sin embargo, ya que el mensaje POLL en las vueltas de respuestas back-off puede tomar un tiempo significativo, hay un “precio que debe ser estipulado

para el tiempo de reparación cuando un nuevo receptor se suma. Los emisores que pueden asumir de forma segura el número de receptores sin un dramático y precipitado crecimiento le permitirán a los nuevos receptores realizar NAKs inmediatamente sin esperar a tomar parte en un POLL (pero siempre estableciendo NAK_BO_IVL_SQN en cero).

Control de Congestión

EL modo básico de operación PGM simplemente limita la tasa de transferencia de los emisores. Para realizar control de congestión, la fuente debe recibir información sobre el estado de la transmisión. Para lograr dicho objetivo, PGM soporta 3 tipos de respuesta de la fuente:

1. Los peores vínculos cargados como medida por los elementos de red
2. Los peores caminos end-to-end cargados como medida por los elementos de red.
3. Los peores caminos cargados como reporte por los receptores.

La fuente utiliza esta devolución para ajustar su tasa de envío. Sin embargo PGM no especifica como la tasa de envío debe ser ajustada. El esquema específico de control de congestión es derivado al implementador.

Capítulo 3: Performance

Introducción

En esta sección discutiremos la performance de PGM (Pragmatic General Multicast). Cubriremos:

- los requerimientos de memoria para los elementos de red
- la reducción del ancho de banda del “back-channel”
- la utilización eficiente de la red
- la transición de recursos a alta seguridad

Consideraremos también las vulnerabilidades de seguridad en PGM

Requerimiento de memoria de los elementos de red

PGM requiere elementos de red para almacenar información sobre su estado. El estado del camino hacia la fuente en sencillo. Ello alcanza y es suficiente para almacenar la información necesaria de la dirección del camino hacia la fuente desde un SPM e indica qué sesión multicast aplica para ello.

Adicionalmente y más significativamente, un elemento de red debería almacenar información del número de secuencia para cada NAK.

Si muchas sesiones PGM se encuentran utilizando elementos de red, éste podría tener memoria insuficiente para todos los NAK de todas las sesiones.

En este caso podría, simplemente, terminar operando transparentemente con un elemento de red non PGM para alguna de las sesiones, reduciendo los requerimientos de memoria hasta un nivel que puedan acomodarse.

Este escenario es más factible que ocurra en routers ubicados en los backbone de Internet, donde el número de sesiones pueden volverse extremos.

No es deseable que un backbone de Internet soporte PGM. Sin embargo como hemos desarrollado más arriba, PGM trabaja hasta a través de rutas que no son PGM y los beneficios de escalar en la jerarquía pueden ser realizados por los routers de PGM que están fuera del backbone.

Uso del back channel

PGM (Pragmatic General Multicast) incluye un número de opciones para hacer eficiente el uso del back channel en transmisiones de NAK.

Soporta la opción OPT_NAK_LIST para incluir la suficiente cantidad de NAK's individuales en el mismo paquete NAK.

Un mismo paquete NAK, para una sola pérdida, es de 56 bytes, mientras que un paquete con la opción OPT_NAK_LIST es de 64 bytes a los que se agregan 4 bytes por cada secuencia de números indicados. Igualmente, el tráfico del back channel puede ser significamente reducido como se muestra a continuación:

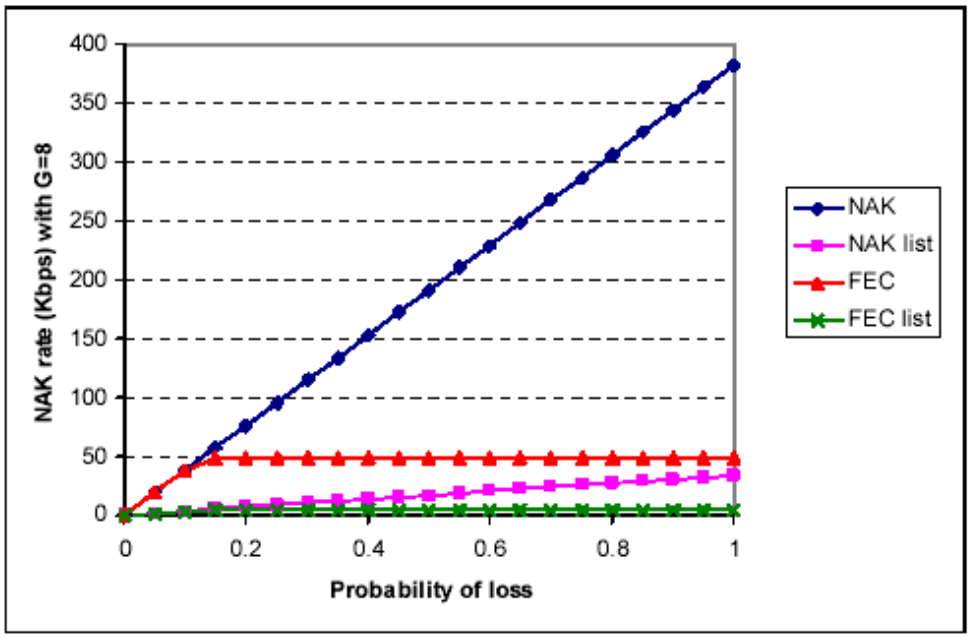


Figura 1

En la figura 1: Tasa de NAK para un receptor unitario desde un emisor unitario vs. Tasa de un simple receptor vs. probabilidad de pérdida. Se enviaron 1500 bytes en 874 veces por segundo para una tasa de transferencia de datos de 10 Mbps. Para FEC se han utilizado un tamaño de grupo de 8.

El uso de FEC puede producir gran cantidad de mejoras. Si G paquetes están en un grupo de transmisión, la tasa de pérdida será de 1/G, en el peor de los casos. Puede derivar en un paquete perdido de cada grupo, requiriendo un NAK para dicho paquete. Ninguna tasa de transferencia mayor incrementará el número de NAKs, mientras cada grupo habrá reportado haber tenido una pérdida. La contabilización de las pérdidas se incrementará.

Ejemplo: con G = 8, el máximo número de NAKs será del 12.5%. (Los efectos secundarios de la orden de pérdida de un paquete NAK y la pérdida de paridad podrían incrementar el ancho de banda del NAK. Pero a nosotros sólo nos conciernen los efectos primarios para los propósitos de ésta discusión)

En la figura 1 anterior se ilustra el uso de FEC. En este ejemplo, usando OPT_NAK _LIST sin FEC es mejor que usar FEC sin OPT_NAK *LIST*. Sin embargo, en un grupo más grande, como por ejemplo G = 64 como muestra la figura 2 a continuación, utilizando las opciones de OPT_NAK_LIST y FEC siempre minimizará el tráfico en reversa de NAK.

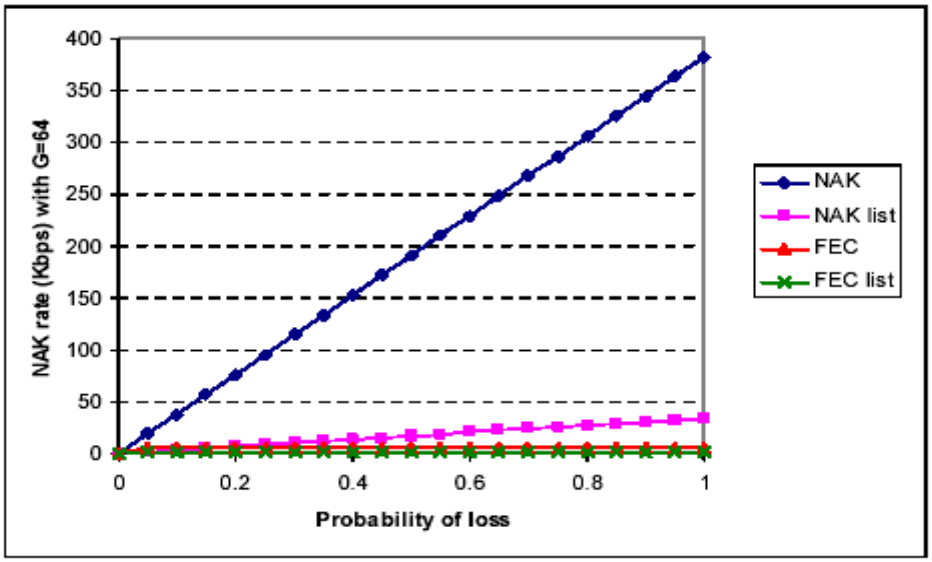


Figura 2

En la figura 2: Tasa de NAK desde un único receptor vs. la probabilidad de pérdida. Se enviaron 1500 bytes en 874 veces por segundo para una tasa de transferencia de datos de 10 Mbps. Para FEC se han utilizado un tamaño de grupo de 64.

El uso de FEC también reduce el tráfico del back channel y esto lo hace mejorando el método de supresión. Esto puede ser observado cuando consideramos el tráfico NAK que vuelve al emisor. Brindando un mayor número de receptores, con una muy baja posibilidad de pérdida, la realidad es que todo paquete es perdido por algún receptor y necesita ser "nakeado". Por ejemplo: una aleatoria e independiente pérdida del 0.1% sobre un 1.000.000 de receptores implica que la posibilidad de que todos los nodos reciban un paquete es menor a 10^{-43}. Como la pérdida de cualquier receptor en este ejemplo es pequeña, los NAKs serán indviduales (no será suficiente para un solo receptor en un mismo instante de tiempo, utlizar OPT_NAK _LIST). Por lo tanto, con una tasa de transferencia de datos de 10 Mbps, 1500 bytes en paquetes sin utilización de FEC, se tendrá la expectativa de un tráfico de NAK de 382 Kbps en el emisor.

En contraposición, usando FEC en un grupo de tamaño 8, se podría reducir potencialmente el tráfico de NAK llegando al emisor a 47,8 Kbps (Este stream de paridad NAKs es suficiente para suprimir todos los NAKs perdidos)

Utilización de la red

PGM y el overhead correspondiente al encabezado IP limitan la utilización de PGM al 97.1% en la red [7]. Otro factor de utilización de la red es los mensajes SPM. Estos son mayormente enviados con una tasa de transferencia de 1 por segundo. Cuando la tasa de transferencia de datos es mas alta que 10 Mbps el impacto en la utilización de la red es insignificante.

Hasta en las transferencias de datos en el dial up y el incremento de SPM's a 5 por segundo, la utilización de la red está todavía sobre el 90%. La siguiente tabla muestra la utilización de red para las diferentes tasas de transferencia de datos y los SPM's

[7] The PGM Reliable Multicast Protocol – Jim Gemmel – Todd Montgomery- Jon Crowcroft

Tasa de transferencia	SPM/seg	Utilización de la red
Any	0	97.1%
10 Mb/s	1	97.1%
10 Mb/s	5	97.0%
47 Kb/s	1	95.9%
47 Kb/s	5	91.7%

Impacto del SPM's en la utilización de la red

Las pérdidas independientes y FEC's también pueden impactar en la utilización de la red. Consideremos el ejemplo de 1.000.000 de receptores con una pérdida independiente de 0.1% donde cada paquete es plausible de perderse por algún receptor:

Sin la utilización de FEC, cada paquete debería ser enviado dos veces (para simplificar deberíamos asumir que los paquetes reparados no se han perdido)

Si la tasa de transferencia de datos es de 10 Mb/s con 1500 bytes/paquetes, la utilización de la red es sólo del 48.5%.

Utilizando FEC y un grupo G de tamaño 8, el tráfico a reparar sería solamente 1/8 del tráfico original, incrementando la utilización a un 86.3%.

Recursos de transmisión de alta velocidad

Fijaremos nuestra atención en la transmisión de alta velocidad utilizando PGM. Una implementación PGM no puede ser enviada a altas tasas de transferencia si el sistema de host

es incapaz de enviar paquetes IP en crudo a la misma tasa de transferencia. Esto requiere NIC's adecuados, establecimientos de buffers de red y ajustes en el kernel.

Adicionalmente los emisores de buffers para las ventanas de transmisión PGM no son preocupantes a bajas tasas de transferencia. Una ventana de transmisión de 30 segundos a una tasa de transmisión de 10 Mbps requiere más de 375 Mb de espacio en el buffer.

Si el buffer de transmisión no está restringido enteramente a la memoria física, los errores en la página pueden degradar la perfomance severamente. Para asegurar operaciones a alta velocidad es necesario requerir suficiente memoria RAM para sostener la ventana de transmisión entera.

Cuando el FEC es empleado, el receptor podría tener requerimientos de memoria similares. Si consideramos el caso donde un paquete es perdido por cada grupo en la ventana de transmisión, todos los paquetes k en cada grupo serán necesarios para decodificar. Esto trae como consecuencia que los mismos deberán ser guardados hasta que la recepción del grupo sea completa. Por lo tanto, el receptor necesitará mantener un (k – 1) / k de la ventana de transmisión. Una vez más, si estos no están en la memoria, la perfomance se verá seriamente degradada.

Un receptor PGM debe prevenir la pérdida en el buffer siempre que esto sea posible. Esto se debe a que PGM utiliza el IP raíz y no tiene ninguna construcción para control de flujo.

Si un emisor se encuentra realizando envíos con alta rapidez, los buffers receptores se desbordarán o se producirá la congestión de la red.

La reacción a esta congestión debe ser manejada en forma diferente que a la pérdida provocada por el buffer desbordado. De esta manera se ajustará la tasa de transferencia del emisor. Para reducir el desborde de buffers en los receptores, las aplicaciones de éstos necesitarán conocer el proceso de adquisición para la lectura de paquetes PGM desde la red. También deberán permitir que PGM realice este proceso con alta prioridad comparado con otros niveles de tareas en las aplicaciones correspondientes,

Además, el sistema operativo conectado a los buffers normalmente requiere modificar los tamaños para manejar las capacidades que el tráfico PGM puede alcanzar a través de un emisor bien ajustado.

Consideraciones sobre la seguridad

Sumado a los problemas usuales de una autenticación end-to-end, PGM es vulnerable a un alto número de riesgos de seguridad. Sin total autentificación, todos los recursos del entorno (vecinos), los receptores, los DLR's y los elementos de red, el protocolo podría ser abusado por SPM's, NAK's, NCF's y mensajes RDATA. Por ejemplo:

1. Los falsos SPM's podrían provocar que los elementos PGM de red direccionen falsamente los NAK's previniendo reparaciones que deberían ser realizadas.

2. Los falsos NAK's podrían provocar que los elementos PGM de red establecieran espurios estados de reparación que caducarán solamente en el time out y podrán provocar, además, una saturación de la memoria en dicho tiempo.

3. Los falsos NCF´s podrán provocar que los elementos PGM de red suspendieran el envío de NAK`s prematuramente, provocando una pérdida de reparaciones.

4. Los falsos RDATA (falsos paquetes reparados) podrían provocar que los elementos PGM de red destruyeran el estado legítimo de reparación resultando en una eventual pérdida de la reparación legítima.

Algunas protecciones pueden ser enumeradas:

1. Amortiguar cambios en la dirección del emisor y el nodo superior PGM en los mensajes SPM (la dirección del emisor debería hacerlo cambiar muy infrecuentemente y el nodo superior PGM debería cambiar ocasionalmente así como cambia el ruteo multicast)

2. Los elementos de red pueden protegerse de sesiones generando un número excesivo de NAK´s por abandono de sesión.

3. Un handshake de tres vías entre los elementos de red y los DLR's podrían permitir a un elemento de red, acertar con gran confianza que un DLR allegado sea identificado por la dirección de la fuente cabecera de la red.

Algunas conclusiones previas

La especificación PGM es una de las propuestas para la ampliación de la pila TCP/IP con una extensión que soporte la difusión múltiple de contenidos, de forma que sea la red y no el servidor, el elemento encargado del envío de paquetes de datos desde una única fuente a un grupo preseleccionado de receptores ya sea sobre infraestructuras LAN o WAN.

Esta propuesta difiere considerablemente de la alternativa de la difusión indiscriminada de información sobre la red, Internet o IP empresarial, contando con la ventaja de racionalizar y por tanto disminuir la circulación innecesaria de tráfico sobre las mismas.[8]

Esta observación, desde una mirada experta, expresa el aumento de las factibles ventajas en la implementación de un servicio de transferencia de archivos bajo PGM sobre una LAN

La mejor perfomance de PGM es alcanzada cuando todos los routers soportan PGM. También cuenta con un excelente soporte asimétrico con el uso de FEC, lista de y NAKs unicast. Experiencias muestran que es posible desarrollar PGM en operaciones de alta velocidad superiores a 100 Mbps.

PGM ha sido implementado tanto en ambientes académicos como en ambientes comerciales. Las implementaciones cliente/servidor fueron incluidas en productos como Windows XP de Microsoft Co. Cisco System ha añadido soporte PGM a sus routers.

[8] Comunicaciones World – Marta Cabanillas –10/02/2000

Capítulo 4: Desarrollos científicos y comerciales sobre PGM y derivados

Este capítulo tiene por objetivo desarrollar un estado del arte con relación a lo explicitado en el primer objetivo del presente trabajo. A saber: desarrollos científicos y comerciales, aplicaciones y estudios comparativos con otros protocolos.

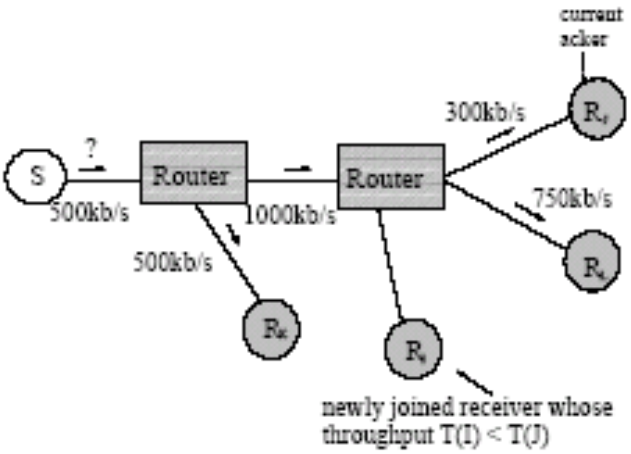
Desarrollos y aportes científicos y/o académicos

PGM Congestion Control

Uno de los trabajos más recientes se detalla a continuación: En el Internet-draft “PGMCC single rate multicast congestion control” se describe PGMCC como esquema del control de la congestión de procedimientos multicast, amigable con TCP (Transfer Control Protocol) y que alcanza escalabilidad, estabilidad y respuesta rápida a las variaciones de las condiciones de la red.

PGMCC se diseña para las sesiones del multicast con un emisor o fuente y uno o más receptores. Utiliza los reconocimientos negativos (NAKs) para recoger la regeneración de datos de los receptores. El esquema del control de la congestión puesto en ejecución por PGMCC se acerca al comportamiento del control de congestión TCP, pues utiliza un lazo de control basado en la ventana de transmisión que funciona entre el emisor y un receptor seleccionado llamado el ACKER. El papel del ACKER es proporcionar la regeneración oportuna a través de la modalidad TCP. Además, la selección del ACKER se realizará dinámicamente entre los receptores pues será el que experimente el más bajo throughput de acuerdo a las sesiones TCP que estuviesen corriendo entre el emisor y los receptores.

En la siguiente figura se muestra en forma esquemática, la selección del “acker”:



¿Qué significa esto? El procedimiento es el siguiente: El emisor mantiene el estado de dos variables W (window) y T (token count) respectivamente. W representa el número de paquetes en línea mientras que T es utilizada para regular la generación de paquetes de datos (data packets). Un "token" se necesita y se consume para transmitir un paquete de datos. Inicialmente ambas variables, W y T, están inicializadas en 1. Los valores de W y T son actualizados con cada ACK, NAK, time-out, y transmisión de paquetes.[9]

La escalabilidad en PGMCC está dado por el uso de los reconocimientos negativos (NAKs) para recoger la regeneración de receptores con excepción del ACKER. Por consiguiente, las técnicas generalmente para la supresión del NAK se pueden utilizar para reducir la cantidad de regeneración a la fuente y para mejorar la escalabilidad del esquema.

PGMCC se diseña para separar totalmente el control de la congestión de la integridad de los datos. Por consiguiente, el esquema puede trabajar con transferencia de datos confiable y protocolos de comunicación no fiables.

Mientras que PGMCC está diseñado para procesamiento multicast, se puede utilizar igualmente como reemplazo para TCP en las sesiones bajo modalidad unicast que requieren un grado más bajo de confiabilidad que aquel ofrecido por TCP.[10]

[9] Dynamics of the "pgmcc" Multicast Congestion Control Protocol
Chin-ying Wang and Sonia Fahmy - Department of Computer Sciences
Purdue University

[10] Draft-ietf-rmt-bb-pgmcc-03.txt
Luigi Rizzo/U. Pisa - Gianluca Iannaccone/Intel - Lorenzo Vicisano/Cisco - 12 July 2004.
Fecha de expiración: January 2005

PGM Congestion Control Negative-Acknowledgment (NACK)-Oriented Reliable Multicast (NORM) - Building Blocks

El transporte confiable multicast es una tecnología deseable para la distribución eficiente y confiable de datos a un grupo en el Internet. Las complejidades de los paradigmas de la comunicación del grupo hacen necesario diversos tipos e instancias del protocolo para resolver la gama de los requisitos del funcionamiento y de la escalabilidad de diversos usos confiables potenciales y de los usuarios de multicast. Este documento trata la creación del reconocimiento negativo (NACK) para protocolos confiables orientados multicast (NORM). Mientras que diversas instancias de este protocolo se pueden requerir para resolver el uso específico y arquitectura de red exigidos, hay un número de componentes fundamentales que pueden ser comunes a esas instancias y otras. Este documento describe el marco y los componentes comunes de los "Building-blocks" relevantes a los protocolos del multicast basados sobre todo en la operación de NACK para el transporte confiable.

También se discute un gran sistema de componentes confiables del multicast y trata detalladamente los "building blocks" que no se tratan en otros documentos del IETF:

1) estrategias de la transmisión del emisor de NORM

2) proceso de la reparación de NACK-oriented con la supresión basada en el tiempo de la regeneración.

3) sincronización ida-vuelta para adaptar contadores de tiempo de NORM. [11]

---

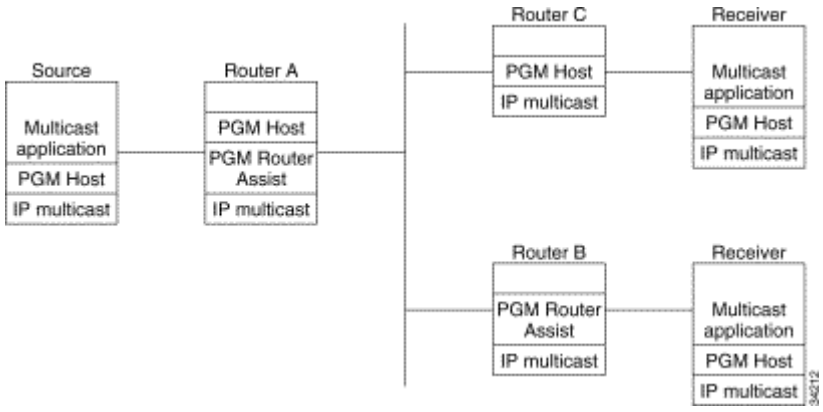
[11] Network Working Group: Request for Comments: 3941, Category: Experimental
B. Adamson- NRL, C. Bormann- Universitaet Bremen TZI, M. Handley- UCL, J. Macker- NRL
Noviembre 2004

Desarrollos comerciales

PGM según Cisco Systems

Otro nuevo servicio que presenta nuevos requerimientos para redes, tanto de empresas como del proveedor de servicios abarca la capacitación y las conferencias vía Internet. Para que las redes que soportan dichas aplicaciones resulten costo efectivas, se requieren IP Multicast y protocolos tales como Pragmatic General Multicast (PGM), Multicast Source Discovery Protocol (MSDP), y multicast BGP (mBGP).

PGM simplifica la operación de la red, al mismo tiempo que permite la entrega de información multicast en forma ordenada y libre de duplicaciones de una o múltiples fuentes a múltiples clientes. MSDP permite que los dominios de ruteo multicast descubran fuentes de otros dominios, de modo de reducir los avisos de membresía grupales (y asimismo los gastos generales de la red).[12]



Topología de red mostrando los hosts y routers PGM

Cuando el router no está funcionando como un elemento de router y las características del host PGM son configuradas (router C), el router puede recibir y enviar paquetes PGM en cualquier interfaz del router simultáneamente como lo especifica el protocolo PGM.

[12] http://www.cisco.com/global/LA/assets/pdfs/beneficios_tecnicos.pdf
Cisco Systems

Aunque esta configuración es soportada, no es recomendable en una red PGM porque este protocolo trabaja con routers que tienen la característica de asistencia PGM router configurada.

En el software Cisco IOS, el host PGM y la característica de asistencia PGM router soportan PGM sobre IP. Ambos utilizan una única sesión de identificación de transporte (TSI) que identifica cada sesión PGM individual.

Las características del host PGM habilitan a los routers de Cisco a soportar aplicaciones multicast que operan en la capa de transporte PGM de dicho protocolo.

Documentos relacionados

- *Cisco IOS IP and IP Routing Configuration Guide*, Release 12.1
- *Cisco IOS IP and IP Routing Command Reference*, Release 12.1

Plataformas soportadas

- Cisco 1003
- Cisco 1004
- Cisco 1005
- Cisco 1600 series
- Cisco 2500 series
- Cisco 2600 series
- Cisco 3600 series
- Cisco 4000 series (Cisco 4000, 4000-M, 4500, 4500-M, 4700, 4700-M)
- Cisco 7000 series
- Cisco 7200 series
- Cisco uBR7200 series
- Cisco 7500 series

TIBCO SmartPGM y TIBCO SmartPGM FX

TIBCO SmartPGM le asegura confiabilidad a las transmisiones multicast y es ideal para transmisiones que requieren órden o desorden, duplicaciones free para envíos multicast desde muchas fuentes a muchos receptores.

TIBCO SmartPGM FX es un producto poderoso de distribución de contenidos realizado sobre Smart PGM. Es por esta razón que TIBCO SmartPGM FX ofrece un conjunto de herramientas que le asegura a un archivo su cualidad multicast con la misma confiabilidad y ancho de banda utilizados en una operación de copia simple.

TIBCO SmartPGM

La habilidad de transmitir y actualizar grandes cantidades de información en tiempo real es especialmente crucial para aplicaciones críticas que impactan en la línea.

Las compañías financieras y de servicios ad hoc deben encontrar la satisfacción a sus requerimientos. La demanda de información financiera debe ser cumplida en forma inmediata, Miles de complicados instrumentos financieros deben ser valuados diariamente. Vínculos punto a punto redundantes no pueden manejar el alto volumen de la información eficientemente. Es por ello que debe ser distribuida y modificada.

Los capitales de 500 empresas se encuentran en movimiento hacia los lugares globales de mercado en donde se encuentran las empresas y están requiriendo aplicaciones distribuidas que no pueden ser afrontadas con la tecnología unicast.

Las características de TIBCO Smart PGM son:

- El control de congestión basado en el corriente ietf –draft para PGMCC
- La prioridad basada en el manejo del ancho de banda permite la ubicación y el compartir del ancho de banda de la red disponible, basado en la prioridad de trabajo.
- Integración de aplicaciones simples utilizando C y C++ APIs
- Completamente configurable para cualquier tipo de topología de red incluyendo terrestre, satelital o híbrida.
- Compatibilidad con Cisco SSM (Source Specific Multicast)

TIBCO ha desarrollado, trabajando conjuntamente con CISCO, un producto que actúa como un elemento de red PGM para redes sin routers PGM disponibles. El demonio SmartPGM soporta:

- PGM RFC 3208
- Operaciones básicas PGM de emisión y recepción

- Cisco IOS
- FEC (Forward error correction)
- Entorno de transferencia de archivos con o sin un “back-channel”
- Agregado de NAKs sin routers
- Soporte DLR

TIBCO Smart PGM FX

Este es un poderoso producto para distribución de contenidos creado sobre TIBO Smart PGM, optimizado para LAN, WAN y ambientes satelitales.

Provee eficiencia y confiabilidad en el envío a miles de receptores simultáneos sin importar lo que los datos sean o bien el tipo de red.

Ya que está creado sobre SmartPGM, TIBCO Smart PGM FX ofrece un conjunto de herramientas que le permiten a un archivo ser enviado bajo modalidad multicast con la misma tranquilidad y ancho de banda como si fuera una operación de copia simple.

Las características de TIBCO Smart PGM FX son:

- Línea de comando y una interfaz easy to use basada en GUI
- La transferencia de archivos debería ser catalogada al principio a un tiempo y fecha determinados y de acuerdo a un determinado ancho de banda y a restricciones prioritarias.
- Monitoreo de GUI que provee capacidad para monitorear archivos característicos de la transferencia del trabajo, incluyendo efectiva y no efectiva recepción por todos los receptores.
- Provee capacidad para transferir un archivo mediante un simple drug and drop en un directorio preconfigurado.

- Capacidad de ubicar las prioridades de transferencia yel suo del ancho de banda directamente desde el GUI.
- Capacidad para permitir o no la inclusión de receptores individuales o grupos de receptores.
- Definición de canales para cada archivo o grupo de archivos permitiéndole a los receptores determinar a cual archivo ellos deberían suscribirse.
- Transferencia de archivos C++ API
- Manejo basado en SNMP
- Alarma y heartbit basados en SMTP
- Soporte para múltiples archivos de compresión y descompresión en una sola transferencia.
- Integración de la transferencia de archivos de software con PGM C++ API

Capítulo 5: Descripción del Sistema Operativo

Para la realización del trabajo en cuestión, se eligió el sistema operativo que se detalla a continuación con ciertas modificaciones a su kernel. Por lo tanto, se describe las características del mismo para luego interpretar:

a) Los motivos de su elección

b) La adaptación basada en el trabajo de Luigi Rizzo

¿Qué es FreeBSD?

FreeBSD es un sistema operativo UNIX indirectamente basado en el port de Net/2 para i386 de Berkeley (conocido como 386BSD) realizado por William Jolitz's y ademas es un avanzado sistema operativo BSD UNIX para ordenadores "PC-compatibles", desarrollado y mantenido por un gran equipo de personas que son quienes toman las decisiones claves concernientes al proyecto FreeBSD como la dirección a seguir y quién está autorizado a añadir código a la distribución original, la cual se hace a través de un grupo de unas 17 personas llamado core team.

Existe también un grupo de unas 150 personas o committers que también están autorizadas a realizar cambios directamente sobre la distribución original.

La primera versión de FreeBSD apareció en 1993, basada en el código Net/2 (4.3BSD). En noviembre de 1994 apareció la versión 2.0 basada ya en el código de BSD 4.4, lo que provocó un aumento muy sustancial en las características, posibilidades y estabilidad de este sistema, a partir de un nuevo sistema de gestión de memoria virtual y un nuevo sistema de gestión de ficheros. En estos momentos, la última versión es la 2.2.6 (abril de este año 1998).

FreeBSD es un sistema operativo totalmente libre, es decir, no hay que pagar por usarlo además, se dispone de **todo** el código fuente del kernel (módulo principal del sistema), lo que permite poder realizar cualquier tipo de modificación o desarrollo sobre él, compilarlo, y comprobar los resultados. Una de las grandes ventajas de FreeBSD sobre otros sistemas como Linux (que cuenta con diferentes distribuciones y cada una con sus propias características, versiones de kernel, aplicaciones, etc), es que existe una sola distribución coordinada por un grupo de trabajo dedicado a ello, haciendo el sistema más homogéneo,

controlado y estándar. Esto no quiere decir que sea un sistema cerrado, ya que los grupos son totalmente abiertos, aceptándose la colaboración de todo el mundo. Además de la versión (también conocida como -stable) existe una rama en continuo desarrollo llamada current sobre la que se realizan todas las modificaciones y actualizaciones para la próxima versión. Tenemos la posibilidad de tener nuestra máquina totalmente actualizada en la rama current mediante un sistema de actualización online llamado cvsup (hay que tener en cuenta que la rama current se basa en el continuo desarrollo del código fuente del kernel y sus aplicaciones, por lo que solo debería trabajarse con esta versión en sistemas de pruebas).

BSD es un acrónimo de "Berkeley Software Distribution", el cual es el nombre que el CSRG de Berkeley (Computer Systems Research Group) escogió para sus distribuciones de Unix.

Características de FreeBSD

Algunas de las características principales de FreeBSD son:

- Sistema desarrollado totalmente en 32 bits.
- "Preemptive multitasking" con ajuste dinámico de prioridades para asegurar un buen reparto de recursos entra aplicaciones y usuarios.
- Multiusuario. Diferentes usuarios pueden usar un mismo sistema FreeBSD simultáneamente. El sistema comparte periféricos como impresoras, disco, cintas, etc.
- Sistema TCP/IP completo, incluyendo SLIP, PPP, NFS, NIS, etc, que nos permite usar FreeBSD como servidor de ficheros, servidor de red, servidor de comunicaciones (http, ftp, nntp, smtp, pop3, imap, dns, routing, firewall, etc) o estación de trabajo.
- Protección de memoria que evita que las aplicaciones o usuarios pueden interferir entre ellas. Si una aplicación falla, no afecta al resto de aplicaciones del sistema.
- X Window System (X11R6), como interface gráfico de usuario (GUI).

- Compatibilidad de binarios con otros sistemas operativos como SCO, BSD/OS, NetBSD, 386BSD, Linux, BSDi.
- Librerías compartidas.
- El sistema base incluye compiladores de C, C++ (cc y gcc), Fortran, etc.
- Disposición de todo el código fuente tanto del kernel como de las aplicaciones incluidas en la instalación base.

Todas esta características (y otras aquí no mencionadas), hacen de FreeBSD uno de los más completos sistemas operativos Unix libres que existen hoy en día.

Sistema de Archivos de FreeBSD

FreeBSD dispone de una serie de aplicaciones englobadas en dos sistemas (packages y ports) que actualmente cuenta con mas de 1000 programas en cada uno. Los packages son aplicaciones ya compiladas y "ready-to-run" en FreeBSD; se instalan mediante el comando pkg_add o mediante un interface de gestión de packages que permite la instalación y desinstalación de manera sencilla. Los ports son ficheros que incluyen el código (o patchs) necesarios para que una aplicación compile sin problemas en FreeBSD. Algunas de las secciones incluidas en los ports y packages son astro, audio, biología, cad, comunicaciones, conversores, desarrollo, bases de datos, editores, emuladores, gráficos, lenguajes de programación, matemáticas, redes, seguridad, herramientas de sistema, www, X11, etc.

FreeBSD es un sistema operativo ideal para usar en entornos de:

- Servicio Internet o de redes:
- Servidor ftp
- Servidor web
- Servidor gopher
- Servidor wais

- Servidor mail (sendmail o cualquier otro MTA)
- Servidor nntp (news)
- BBS
- Router
- DNS
- Firewall
- Educación:
- No hay mejor manera de aprender sobre sistemas operativos, arquitectura de computadoras, redes, etc, que poder poner las manos en ello. Existe un gran número de aplicaciones libres sobre CAD, matemáticas, diseño gráfico, etc disponibles y preparadas para instalar y usar de manera sencilla sobre FreeBSD.
- Investigación:
- Desarrollo de software:
- La inclusión de los diferentes compiladores y debuggers bajo licencia GNU en la distribución base, hace que el desarrollo de software sea mucho más sencillo.
- Posibilidad de desarrollo en diferentes lenguajes C, C++, Fortran, Pascal, Modula3, Perl, Shell script, etc.

Es importante destacar que, con el código fuente completo de todo el sistema a nuestra disposición, FreeBSD es una excelente plataforma de investigación en sistemas operativos, programación, etc. La naturaleza de libre disposición de FreeBSD hace posible la colaboración en ideas o desarrollos de grupos remotos sin tener que preocuparse sobre licencias de uso especiales, limitaciones de uso, etc.

Los requerimientos a nivel de hardware de FreeBSD son mínimos ya que los podemos instalar en cualquier máquina con microprocesador 386 o superior y con un mínimo de 5Mb de memoria RAM. Acepta cualquier disco duro de tipo IDE o SCSI y diferentes tarjetas de red de diferentes fabricantes.

FreeBSD no usa más espacio de swap que Linux

Forma en que se personaliza el kernel

Primero, se necesita la distribución completa de fuentes o, por lo menos, la distribución de fuentes del kernel. De esta manera se tienen las fuentes necesarias para crear un nuevo kernel. Al contrario que muchos Unix comerciales, la política es la de NO vender el kernel en formato binario.

La instalación de las fuentes ocupa un poco más de espacio, pero permite consultar las fuentes del kernel en caso de dificultad o entender que está ocurriendo realmente en la ejecución del sistema.

Una vez instalada la distribución completa de fuentes, o por lo menos la del kernel, se hace lo siguiente como root:.

1. cd /usr/src/sys/i386/conf
2. cp GENERIC MYKERNEL
3. vi MYKERNEL
4. config MYKERNEL
5. cd ../../compile/MYKERNEL
6. make depend
7. make all
8. make install
9. reboot

El paso 2 no es necesario si todavía se tiene un fichero de configuración del kernel de una release anterior de FreeBSD 2.X. - simplemente, se copia el fichero antiguo y se examina

cuidadosamente para asegurar que no haya cambiado la sintaxis de algún driver, o haya alguno anticuado.

Un buen fichero de configuración para consultar es LINT, el cual contiene ejemplos documentados para todas las posibles opciones del kernel. El fichero de configuración GENERIC se usa para crear el kernel "por defecto" que es el que se estará usando si no se ha creado ninguno nuevo.

Si no se necesita hacer ningún cambio al fichero GENERIC, se puede saltar al paso 3, donde se personaliza el kernel para el sistema. El paso 8 solo debe ejecutarse si los pasos 6 y 7 se han realizado de manera satisfactoria. Esto copiará una imagen del nuevo kernel a /kernel y realizará una copia del antiguo kernel en /kernel.old. Es muy importante recordar esto por si el nuevo kernel falla en algún momento – se puede seleccionar kernel.old en el prompt de arranque. Al hacer un reboot, por defecto se cargará el nuevo kernel.

Si la compilación en el paso 7 falla por alguna razón, es recomendable que se empiece desde el paso 4 substituyendo GENERIC por MYKERNEL. Si se puede generar el kernel GENERIC, significa que algo en el fichero de configuración es incorrecto (o se ha descubierto un bug). Si la compilación del kernel GENERIC falla, posiblemente se tenga los archivos fuentes corruptos.

Finalmente, si se necesita ver los mensajes originales de arranque del sistema para compilar un nuevo kernel, se ejecuta el comando dmesg. Este comando debe imprimir en pantalla todos los mensajes producidos por el kernel al arrancar, los cuales pueden servir en la configuración del nuevo kernel.

Se recomienda hacer un historial fechado de los kernel que se vayan creando, de la manera kernel.YYMMDD una vez que estén funcionando correctamente. De esta manera, si la próxima vez que se modifique el kernel algo no funciona, se puede arrancar desde el último kernel correcto. Esto es especialmente importante si ahora sé este arrancando desde una controladora no soportada por el kernel GENERIC.

FreeBSD ofrece él más alto rendimiento, gran compatibilidad con otros sistemas operativos y una menor administración del sistema.

Los desarrolladores de FreeBSD se han enfrentado a algunos de los problemas más difíciles en el diseño de sistemas operativos para poder ofrecerte estas avanzadas características:

Bounce buffering trata sobre la limitación en la arquitectura ISA de los PC's que limita el acceso directo a memoria en los primeros 16 megabytes.

Resultado: sistemas con más de 16 megabytes operan más eficientemente con periféricos DMA en el bus ISA.

Un buffer de caché conjunto de memoria virtual y sistema de ficheros continuamente ajusta la cantidad de memoria usada por los programas y el cache de disco.

Resultado: los programas reciben una excelente gestión de memoria y un alto rendimiento en los accesos a disco, liberando al administrador del sistema del trabajo de ajustar los tamaños de los cachés.

Módulos de compatibilidad que permiten la ejecución de programas de otros sistemas operativos en FreeBSD, incluyendo programas para Linux, SCO, NetBSD y BSDI.

Resultado: los usuarios no tendrán que recompilar programas ya compilados para algunos de los sistemas compatibles, teniendo acceso a programas como las extensiones para BSDI de Microsoft FrontPage Server o WordPerfect para SCO y Linux.

Módulos de kernel de carga dinámica que permiten tener acceso a nuevos sistemas de ficheros, protocolos de red o emuladores de binarios en tiempo de ejecución sin necesidad de generar un nuevo kernel.

Resultado: Se puede ganar mucho tiempo y desarrolladores de terceras partes pueden ofrecer subsistemas completos como módulos de kernel sin necesidad de distribuir el código fuente o complejos procedimientos de instalación.

Librerías compartidas reducen el tamaño de los programas, ahorrando espacio de disco y memoria. FreeBSD usa un avanzado esquema de librerías compartidas que ofrecen muchas de las ventajas de ELF, ofreciendo la versión actual compatibilidad ELF con programas de Linux y nativos de FreeBSD.

Naturalmente, como FreeBSD es un esfuerzo en constante evolución, se pueden esperar nuevas características y niveles más altos de estabilidad con cada release.

El objetivo de FreeBSD

El objetivo del Proyecto FreeBSD es proveer software que pueda ser usado en todos los ámbitos sin ningún tipo de atadura. El grupo de FreeBSD está significativamente involucrado en el desarrollo del código (y del proyecto) y no sería cierto decir que no espera algún tipo de financiación, pero definitivamente no está preparado para insistir en ello. El objetivo primordial es que la primera y principal "misión" sea proveer el código libremente, y en cualquier ámbito, para que el código sea lo más expandido posible y produzca los mayores beneficios. Esto es, se sostiene uno de los objetivos fundamentales del Software Libre y se lo apoya de manera incondicional.

El código fuente de los programas registrados bajo GNU General Public License (GPL) o GNU Library General Public License (LGPL), se provee bajo las condiciones fijadas por esas licencias. Debido a complicaciones adicionales en el uso comercial de Software GPL, se intenta reemplazar ese software por otros registrados bajo el copyright BSD, menos estricto y más permisivo.

De esta manera, se alcanzará en la evaluación e implementación mencionada en los objetivos, un nivel de independencia importante, teniendo en cuenta que este software seguirá creciendo sin condicionamientos comerciales.

Metas Del Proyecto De FreeBSD

“Las metas del proyecto de FreeBSD son proporcionar al software, lógica que se puede utilizar para cualquier propósito[...]. Muchos de nosotros tienen una inversión significativa en el código (y el proyecto) y ahora y entonces no dependerían ciertamente de una poca remuneración financiera, [...]. Creemos que nuestro primer ``mission'' debe proporcionar código a cualesquiera y a todos los comers, y para cualquier propósito, de modo que el

código consiga el uso posible más amplio y proporcione a la mayor ventaja posible. Esto es, yo creo, una de las metas más fundamentales del software libre [...]".[13]

Consideraciones Finales

1. FreeBSD es usado por compañías, proveedores de Internet, profesionales de la informática, estudiantes y usuarios particulares de todo el mundo en su trabajo, educación y ocio.

2. FreeBSD ofrece muy altas prestaciones en networking, rendimiento, seguridad y compatibilidad, las cuales no están aún presentes en otros sistemas operativos, incluyendo los comerciales de mayor renombre.

3. La calidad de FreeBSD combinada con el hoy en día bajo coste del hardware de alta velocidad para PC's hace de este sistema una alternativa muy económica sobre las estaciones de trabajo UNIX comerciales. Existe gran cantidad de aplicaciones tanto a nivel servidor como usuario.

4. FreeBSD puede ser instalado desde una gran variedad de soportes incluyendo CD-ROM, floppies, cintas magnéticas, una partición MS-DOS o si se tiene una conexión de red, se puede instalar directamente sobre FTP anónimo o NFS. Todo lo que se necesita son dos discos de 1.44MB de arranque y las instrucciones adecuadas.

5. Por ultimo FreeBSD es uno de los más completos sistemas operativos Unix libres que existen hoy en día.

13

http://www.freebsd.org (Site Oficial)
http://www.es.freebsd.org/es/ (Documentación en castellanos)
http://www.es.freebsd.org/handbook.html (manual de Free BSD)
http://www.es.freebsd.org/es/FAQ/FAQ.html (Las FAQ de FreeBSD en castellano)
http://www.es.freebsd.org/es/FAQ/FAQ.html (Las FAQ de FreeBSD)
http://www.es.freebsd.org/cgi/man.cgi (Los man de FreeBSD)

Capítulo 6: Modificaciones al Sistema Operativo

La primera aproximación

En el sitio www.iet.unipi.it/~luigi/pgm.html obtuvimos el código fuente y la documentación para una puesta en práctica del host PGM para los varios lanzamientos de FreeBSD.

En el sitio se asegura la estabilidad del código. También se agrega:

- Puesta en práctica completa del host de los mecanismos básicos de PGM (FEC y reparadores locales no están implementados todavía).
- Opción experimental de las opciones FIN.
- Demostración disponible en una versión bajo diskette de FreeBSD.

Estas características permitieron utilizar los sockets de PGM como TCP. Una aplicación simple de la prueba, pgmcat está también disponible la cual permite una transferencia de archivo confiable simple en modalidad multicast, con control completo de la congestión.

Una versión de la demo

En un principio, recurrimos a una instalación precaria de FreeBSD.

Del sitio www.iet.unipi.it/~luigi/pgm.html obtuvimos la imagen de un sistema FreeBSD, PGM, el archivo pgmcat y las herramientas de la prueba tales como ipfw, dummynet, tcpdump.

Para instalar este sistema, descargamos 1.44MB correspondiente a la imagen, picopgm.000216.bin y utilizando “dd” bajo FreeBSD copiamos estos archivos a un diskette.

Luego colocamos el diskette en una máquina con interfaz de Ethernet y realizamos la instalación. Después de acceder a sistema como “root”, dispusimos de los códigos correspondientes a pgmcat, dummynet etc.

Prueba posterior

Posteriormente se realizó la implementación sobre Free BSD 4.0 –stable.

Para ello se modificó el kernel agregando los siguientes archivos:

```
/usr/src/sys/netinet/pgm.h
/usr/src/sys/netinet/pgm_var.h
/usr/src/sys/netinet/pgm_timer.c
/usr/src/sys/netinet/pgm_usrreq.c
/usr/src/sys/netinet/pgm.4
```

Los parches de actualización fueron los siguientes:

```
/usr/src/sys/netinet/in.h
/usr/src/sys/netinet/in_proto.c
/usr/src/sys/conf/files
/usr/src/sys/conf/options
```

A continuación se muestra las indicaciones dadas en el archivo README correspondiente:

PGM - FreeBSD (UCL MSc DCNDS project)

This zip file includes the following 11 files:

-in.h
-in_proto.c
-pgm.h
-pgm_rcv_subr.c
-pgm_snd_subr.c
-pgm_timer.c
-pgm_timer.h
-pgm_var.h
-pgm_usrreq.c

-socketvar.h
-files

Installation:

File 'socketvar.h' should be placed in usr/src/sys/sys/
File 'files' should be placed in /usr/src/sys/conf
The rest of the files should be placed in /usr/src/sys/netinet

---
Compiling:

1. cd /usr/src/sys/i386/conf
2. cp GENERIC PGMKERNEL

note: GENERIC config file is what is already there as a kernel.
All the above must be executed under the root account otherwise you'll get
permission denied errors.

When you're done type the following to compile and install your kernel:

4. /usr/sbin/config PGMKERNEL
5. cd ../../compile/PGMKERNEL
6. make depend
7. make (all)

The new kernel will be copied to the root directory as /kernel and the old kernel will be moved to /kernel.old
Now shutdown and reboot to use the new kernel.

Una vez recompilado el Kernel con estas modificaciones se utilizaron las instrucciones indicadas para la aplicación de sockets PGM

```
pgmcat -s -ttl TTL group/port < source_file   # on the sender

pgmcat group/port > dest_file   # on the receivers.
```

TCP DUMP

Tcpdump (y su port a Windows, Windump) son programas cuya utilidad principal es analizar el tráfico que circula por la red. Se apoya en la librería de captura pcap, la cual presenta una interfaz uniforme y que esconde las peculiaridades de cada sistema operativo a la hora de capturar tramas de red. Para seguir el manual es necesario unos conocimientos básicos del protocolo TCP/IP, remitiéndome al TCP/IP Illustrated, Volumen 1 de Stevens, para quien esté interesado.

Aunque viene incluido con la mayoría de las distribuciones de Linux, sus fuentes pueden encontrarse en www.tcpdump.org

El port completo para Windows, tanto de las librerías como del tcpdump puede encontrarse en la web de la Politécnica de Torino. Es un simple binario, que necesita tener instalado el port de las pcap para windows para funcionar.

Para su uso básico, lo primero que debemos averiguar cuando estamos usando el tcpdump, son las interfaces que queremos escuchar. Por defecto cuando se ejecuta sin parámetros, en

los Linux se pone a escuchar en la eth0, mientras que en Windows hay que especificarla la interfaz donde quiere escuchar.

Para averiguar las interfaces en cualquier Unix recurrimos al comando ifconfig -a el cual nos da una lista de las interfaces que tenemos, así como sus parámetros de configuración.

Este es el parche a esta aplicación para que reconozca y decodifique paquetes PGM:

```
--- interface.h.orig   Sun Nov 21 11:51:25 1999
+++ interface.hFri Jan  7 12:07:19 2000
@@ -67,7 +67,7 @@
  * In particular, it allows for an ethernet header, tcp/ip header, and
  * 14 bytes of data (assuming no ip options).
  */
-#define DEFAULT_SNAPLEN 68
+#define DEFAULT_SNAPLEN 80 /* for pgm, need at least 74 */

 #ifndef BIG_ENDIAN
 #define BIG_ENDIAN 4321
--- print-ip.c.orig    Tue Sep 15 21:46:59 1998
+++ print-ip.c Fri Jan  7 12:07:19 2000
@@ -417,6 +417,13 @@
                       icmp_print(cp, (const u_char *)ip);
                       break;

+#ifndef IPPROTO_PGM
+#define       IPPROTO_PGM  113
+#endif
+               case IPPROTO_PGM:
+                       pgm_print(cp, len, (const u_char *)ip);
+                       break;
+
 #ifndef IPPROTO_IGRP
 #define IPPROTO_IGRP 9
 #endif
--- print-udp.c.orig   Tue Sep 15 21:46:59 1998
+++ print-udp.cFri Jan  7 16:05:39 2000
@@ -446,3 +446,287 @@
        } else
                (void)printf(" udp %u", (u_int32_t)(ulen - sizeof(*up)));
 }
+
+#if 1
+/*
+ * this code does the printing of pgm packets
+ * Nov. 1999 - Universita` di Pisa
+ */
+typedef u_int32_t pgm_seq ;
+
+struct pgmhdr {
+        u_int16_t ph_sport ;
+        u_int16_t ph_dport ;
+        u_int8_t  type ;
+#define PGM_SPM_TYPE    0
+#define PGM_OD_TYPE     4
```

```
+#define PGM_RD_TYPE     5
+#define PGM_NAK_TYPE    8
+#define PGM_NNAK_TYPE   9
+#define PGM_NCF_TYPE    10
+#define PGM_ACK_TYPE    11      /* for cong.control */
+
+       u_int8_t options ;
+#define PGM_OPT_PRESENT         1       /* there are option extentions
*/
+#define PGM_OPT_NE_PRESENT      2       /* there are NE-significant
opt.ext. */
+
+       u_int16_t checksum ;
+       u_int32_t gsid_low ;
+       u_int16_t gsid_high ;
+       u_int16_t tsdu_len ;
+       /*
+        * all packets have a couple of sequence numbers here, but
+        * their meaning differs.
+        */
+       pgm_seq _seq1;
+#define od_txw_trail    _seq1   /* this is in ODATA pkts */
+#define spm_seq         _seq1   /* this is in SPM pkts */
+#define nak_req_seq     _seq1   /* this is in NAK pkts */
+#define ack_req_seq     _seq1   /* this is in ACK pkts */
+
+       pgm_seq _seq2;
+#define od_dp_seq       _seq2   /* this is in ODATA pkts */
+#define spm_txw_trail   _seq2   /* this is in SPM pkts */
+#define ack_rxw_lead    _seq2   /* this is in ACK pkts */
+} ;
+
+/*
+ * Source Path Message (SPM) packets
+ */
+
+struct pgm_spm_body {
+       u_int32_t spm_le_seq ;
+       u_int16_t nla_afi ;
+       u_int16_t rsvd ;
+       struct in_addr path_nla ;
+       u_char options[0];
+} ;
+
+struct pgm_spm {
+       struct pgmhdr pgmhdr ;
+       struct pgm_spm_body body ;
+} ;
+
+/*
+ * (N)ACK packets (from receivers/DLR)
+ */
+struct pgm_nack_body {
+       /*
+       u_int32_t req_seq ;
+       u_int16_t nla_afi ;
+       u_int16_t rsvd1 ;
+       */
+       struct in_addr src_nla ;
+       u_int16_t nla_afi2 ;
+       u_int16_t rsvd2 ;
```

```
+        struct in_addr mc_nla ;
+        u_char options[0] ;
+} ;
+
+/*
+ * ACK packets (from elected receivers)
+ */
+struct pgm_ack_body {
+        /*
+        u_int32_t req_seq ;
+        u_int16_t nla_afi ;
+        u_int16_t rsvd1 ;
+        */
+        u_int32_t ack_bitmask ;
+} ;
+#if 1
+/*
+ * options (similar to IP options)
+ */
+#define OPT_HLEN sizeof(struct pgm_option)
+
+struct pgm_option {
+        u_int8_t type;
+        u_int8_t len;
+        u_int16_t tot_len;
+};
+/*
+ * PGM options. The same names, in some cases, are also used
+ * as parameter names in set/getsockopt.
+ * Most options are unimplemented as of 990928
+ */
+#define PGM_OPT_LENGTH          0x00
+#define PGM_OPT_FRAGMENT        0x01
+#define PGM_OPT_JOIN            0x03
+#define PGM_OPT_TIME            0x04
+#define PGM_OPT_RXQ             0x05
+#define PGM_OPT_DROP            0x06
+#define PGM_OPT_REDIRECT        0x07
+
+#define PGM_OPT_LOSSRATE        0x10
+#define PGM_OPT_SEND_NAK        0x11    /* receivers, please send a NAK
*/
+#define PGM_OPT_ELECT           0x12    /* node X is elected as acker
*/
+#define PGM_OPT_SYN             0x21
+#define PGM_OPT_FIN             0x22
+
+#define PGM_OPT_MASK         0x7f
+
+#define PGM_OPT_END             0x80    /* end of options marker */
+
+/* experimental options */
+#define PGM_OPT_LOSSRATE        0x10
+#define PGM_OPT_DESC            0x20
+#define PGM_OPT_SYN             0x21
+#define PGM_OPT_FIN             0x22
+
+#endif
+
+void
```

```
+pgm_print(register const u_char *bp, u_int length, register const
u_char *bp2)
+{
+    const struct pgmhdr *pp; /* pgm header */
+    const struct ip *ip;
+    const u_char *cp;
+    const u_char *ep = bp + length;
+    u_short sport, dport, ulen;
+    int slen, hlen = 0 ;
+    char *s, buf[16];
+
+    hlen = sizeof(struct pgmhdr) ; /* header at least this long */
+    slen = snapend - bp ; /* have this many bytes in the snap */
+    slen -= hlen ; /* less the header... */
+    if (slen < 0) { /* not enough data to print the header */
+       printf("[|pgm]");
+       return;
+    }
+    if (ep > snapend)
+       ep = snapend;
+    pp = (struct pgmhdr *)bp;
+    ip = (struct ip *)bp2;
+    cp = (u_char *)(pp + 1 );
+
+    sport = ntohs(pp->ph_sport);
+    dport = ntohs(pp->ph_dport);
+    ulen = ntohs(pp->tsdu_len);
+    (void)printf("%s.%s > %s.%s: GSI 0x%04x%02x len %d ",
+       ipaddr_string(&ip->ip_src),
+       udpport_string(sport),
+       ipaddr_string(&ip->ip_dst),
+       udpport_string(dport),
+       ntohl(pp->gsid_low),
+       ntohs(pp->gsid_high),
+       ulen );
+    /* now process pieces after fixed header */
+    switch( pp->type ) {
+    case PGM_SPM_TYPE: {
+       struct pgm_spm_body *b = (struct pgm_spm_body *)(cp) ;
+       if (slen < sizeof (*b) ) {
+           printf("[|spm-body]");
+           return ;
+       }
+       printf( "SPM %u path %s",
+              ntohl(pp->spm_seq),
+              ipaddr_string( &(b->path_nla.s_addr) ) );
+       cp = (u_char *)(b+1);
+       }
+       break ;
+
+    case PGM_OD_TYPE:
+    case PGM_RD_TYPE:
+       printf( "%s %u trail %u",
+              pp->type == 4 ? "ODATA" : "RDATA" ,
+              ntohl(pp->od_dp_seq),
+              ntohl(pp->od_txw_trail) );
+       break ;
+
+    case PGM_NAK_TYPE:
+    case PGM_NCF_TYPE: {
+       struct pgm_nack_body *b= (struct pgm_nack_body *)(cp) ;
```

```
+       if (slen < sizeof (*b) ) {
+           printf("[|nak/ncf-body]");
+           return ;
+       }
+       printf( "%s %u src %s mc %s",
+               pp->type == 8 ? "NAK" : "NCF" ,
+               ntohl(pp->nak_req_seq),
+               ipaddr_string( &(b->src_nla.s_addr) ),
+               ipaddr_string( &(b->mc_nla.s_addr) )
+       );
+       cp = (u_char *)(b+1);
+       }
+       break ;
+
+   case PGM_ACK_TYPE: {
+       struct pgm_ack_body *b= (struct pgm_ack_body *)(cp) ;
+       if (slen < sizeof (*b) ) {
+           printf("[|ack-body]");
+           return ;
+       }
+       printf( "ACK %u lead %u mask %08x",
+               ntohl(pp->ack_req_seq),
+               ntohl(pp->ack_rxw_lead),
+               ntohl(b->ack_bitmask)
+       );
+       cp = (u_char *)(b+1);
+       }
+       break ;
+
+    default:
+       printf("UNRECOGNIZED TYPE 0x%02x\n", pp->type);
+       return ;
+    }
+    if (pp->options & PGM_OPT_PRESENT) {
+       struct pgm_option *o = (struct pgm_option *)cp ;
+       if (slen < 4) { /* minimal option len */
+           printf("[Ioptions]");
+           return ;
+       }
+       while (cp+4 <= ep) {
+           o = (struct pgm_option *)cp ;
+           switch (o->type & PGM_OPT_MASK) {
+           case PGM_OPT_LENGTH:
+               printf(" OPT[%d] %d ", o->len, ntohs(o->tot_len) );
+               break ;
+           case PGM_OPT_JOIN:
+               if (o->len != 8 || cp+8 > ep) {
+                   printf("[|join]");
+               } else {
+                   u_int32_t *p = (u_int32_t *)(cp+4);
+                   printf("join %d ", ntohl(*p) );
+               }
+               break ;
+           case PGM_OPT_LOSSRATE:
+               if (o->len != 16 || cp+16 > ep) {
+                   printf("[|lossrate]");
+               } else {
+                   u_int32_t *p = (u_int32_t *)(cp+4);
+                   printf("loss %0.6f ", (double)(1.0*ntohl(*p)/65536)
);
+                   printf("lead %d ", ntohs(p[1]) );
```

```
+                    printf("recv %s ", ipaddr_string( p+2 ) );
+                }
+                break ;
+            case PGM_OPT_SEND_NAK:
+                if (o->len != 8 || cp+8 > ep) {
+                    printf("[|send_nak]");
+                } else {
+                    u_int32_t *p = (u_int32_t *)(cp+4);
+                    printf("send_nak %s ", ipaddr_string( p ) );
+                }
+                break ;
+            case PGM_OPT_ELECT:
+                if (o->len != 8 || cp+8 > ep) {
+                    printf("[|elect]");
+                } else {
+                    u_int32_t *p = (u_int32_t *)(cp+4);
+                    printf("elect %s ", ipaddr_string( p ) );
+                }
+                break ;
+            default:
+                printf("OPT_%d [%d] ", o->type, o->len);
+                break ;
+            }
+            if (o->len < 4)
+                cp += 4 ; /* malformed opt, avoid infinite loop */
+            else
+                cp += o->len ;
+            if (o->type & PGM_OPT_END)
+                return ;
+        }
+    }
+}
+#endif
```

Capítulo 7: Medición sobre transferencia de archivos

En este capítulo se describirá y detallará las instancias previas a la prueba de transferencia sobre PGM, especificando las características específicas del marco que rodeó a la prueba realizada y los procedimientos llevados a cabo para tal fin. Por otra parte, se describirá la prueba propiamente dicha y las conclusiones obtenidas

Descripción del trabajo realizado

FTP (File transfer protocol) es una aplicación comúnmente usada. Aquí se tomó el concepto de transferencia de archivos provisto por FTP dentro de un esquema cliente-servidor a través del puerto 21. Se utilizó FTP con un servidor que permitió anonymous FTP.

Una vez instalada la versión de FreeBSD 4.0 con las modificaciones para el soporte de PGM, se realizaron transferencias de archivos dentro de la LAN interna, en los laboratorios de la Universidad CAECE sede Piedras. :

Los sockets se crearon de acuerdo a las recomendaciones realizadas en el trabajo de Luiggi Rizzo utilizando la aplicación pgmcat, descriptas en en capítulo anterior

Las referencias son las siguientes:

avg: tamaño promedio de cada paquete

stddev: desvío estandar del tamaño de cada paquete

bps: ancho de banda

n: número de paquetes que atravesaron la interfaz

```
Pragamatic General Multicast

Desde el servidor

Transferencia de archivos:

Cliente 1

      Time:1089209588   n=2   avg=46.00         stddev=0.00 bps=147.20
      Time:1089209593   n=0   avg=0.00          stddev=0.00 bps=0.00
      Time:1089209598   n=0   avg=0.00          stddev=0.00 bps=0.00
      Time:1089209603   n=0   avg=0.00          stddev=0.00 bps=0.00
      Time:1089209608   n=0   avg=0.00          stddev=0.00 bps=0.00
      Time:1089209613   n=0   avg=0.00          stddev=0.00 bps=0.00
```

```
Time:1089209618   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209623   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209628   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209633   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209638   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209643   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209648   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209653   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209658   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209663   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209668   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209673   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209678   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209683   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209688   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209693   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209698   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209703   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209708   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209713   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209718   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209723   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209728   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209733   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209738   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209743   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209748   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209753   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209758   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209763   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209768   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209773   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209778   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209783   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209788   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209793   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209798   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209803   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209808   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209813   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209818   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209823   n=1   avg=46.00         stddev=0.00 bps=73.60
Time:1089209828   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209833   n=1   avg=46.00         stddev=0.00 bps=73.60
Time:1089209838   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209843   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209848   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209853   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209858   n=0   avg=0.00          stddev=0.00 bps=0.00
Time:1089209863   n=1   avg=46.00         stddev=0.00 bps=73.60
Time:1089209868   n=303 avg=67.88         stddev=1.84 bps=32908.80
Time:1089209873   n=500 avg=68.00         stddev=0.00 bps=54400.00
Time:1089209878   n=500 avg=68.00         stddev=0.00 bps=54400.00
Time:1089209883   n=500 avg=68.00         stddev=0.00 bps=54400.00
Time:1089209888   n=500 avg=68.00         stddev=0.00 bps=54400.00
Time:1089209893   n=499 avg=68.00         stddev=0.00 bps=54291.20
Time:1089209898   n=500 avg=68.00         stddev=0.00 bps=54400.00
Time:1089209903   n=500 avg=68.00         stddev=0.00 bps=54400.00
Time:1089209908   n=500 avg=68.00         stddev=0.00 bps=54400.00
Time:1089209913   n=500 avg=68.00         stddev=0.00 bps=54400.00
Time:1089209918   n=500 avg=68.00         stddev=0.00 bps=54400.00
```

```
Time:1089209923   n=500 avg=68.00         stddev=0.00 bps=54400.00
Time:1089209928   n=500 avg=68.00         stddev=0.00 bps=54400.00
Time:1089209933   n=500 avg=68.00         stddev=0.00 bps=54400.00
Time:1089209938   n=500 avg=68.00         stddev=0.00 bps=54400.00
Time:1089209943   n=500 avg=68.00         stddev=0.00 bps=54400.00
Time:1089209948   n=500 avg=68.00         stddev=0.00 bps=54400.00
Time:1089209953   n=500 avg=68.00         stddev=0.00 bps=54400.00
Time:1089209958   n=500 avg=68.00         stddev=0.00 bps=54400.00
Time:1089209963   n=500 avg=68.00         stddev=0.00 bps=54400.00
Time:1089209968   n=500 avg=68.00         stddev=0.00 bps=54400.00
Time:1089209973   n=500 avg=68.00         stddev=0.00 bps=54400.00
Time:1089209978   n=500 avg=68.00         stddev=0.00 bps=54400.00
Time:1089209983   n=500 avg=68.00         stddev=0.00 bps=54400.00
Time:1089209988   n=500 avg=68.00         stddev=0.00 bps=54400.00
Time:1089209993   n=366 avg=68.00         stddev=0.00 bps=39820.80


Comienzo de la transferencia: Wed, 7 Jul 2004 14:17:48 UTC

Fin de la transferencia: Wed, 7 Jul 2004 14:19:53 UTC


Duración:  2 min 5 segundos



Cliente 2


Time:1089209788   n=1   avg=46.00   stddev=0.00 bps=73.60
Time:1089209793   n=2   avg=46.00   stddev=0.00 bps=147.20
Time:1089209798   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089209803   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089209808   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089209813   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089209818   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089209823   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089209828   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089209833   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089209838   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089209843   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089209848   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089209853   n=1   avg=46.00   stddev=0.00 bps=73.60
Time:1089209858   n=1   avg=46.00   stddev=0.00 bps=73.60
Time:1089209863   n=2   avg=46.00   stddev=0.00 bps=147.20
Time:1089209868   n=3   avg=56.00   stddev=14.14      bps=268.80


Comienzo de la transferencia: Wed, 7 Jul 2004 14:17:48 UTC

Fin de la transferencia: Wed, 7 Jul 2004 14:19:53 UTC


Duración:  2 min 5 segundos


Recepción Cliente 1

Time:1089209866   n=3         avg=533.33  stddev=652.42     bps=2560.00
```

```
Time:1089209871   n=1146      avg=757.08  stddev=694.06 bps=1388188.80
Time:1089209876   n=1001      avg=761.31  stddev=694.00 bps=1219315.20
Time:1089209881   n=1002      avg=760.62  stddev=693.99 bps=1219430.40
Time:1089209886   n=1001      avg=761.31  stddev=694.00 bps=1219315.20
Time:1089209891   n=1002      avg=760.62  stddev=693.99 bps=1219430.40
Time:1089209896   n=1001      avg=761.31  stddev=694.00 bps=1219315.20
Time:1089209901   n=1002      avg=760.62  stddev=693.99 bps=1219430.40
Time:1089209906   n=1002      avg=760.62  stddev=693.99 bps=1219430.40
Time:1089209911   n=1001      avg=761.31  stddev=694.00 bps=1219315.20
Time:1089209916   n=1002      avg=760.62  stddev=693.99 bps=1219430.40
Time:1089209921   n=1001      avg=761.31  stddev=694.00 bps=1219315.20
Time:1089209926   n=1002      avg=760.62  stddev=693.99 bps=1219430.40
Time:1089209931   n=1001      avg=761.31  stddev=694.00 bps=1219315.20
Time:1089209936   n=1002      avg=760.62  stddev=693.99 bps=1219430.40
Time:1089209941   n=1001      avg=761.31  stddev=694.00 bps=1219315.20
Time:1089209946   n=1002      avg=760.62  stddev=693.99 bps=1219430.40
Time:1089209951   n=1002      avg=760.62  stddev=693.99 bps=1219430.40
Time:1089209956   n=1001      avg=761.31  stddev=694.00 bps=1219315.20
Time:1089209961   n=1002      avg=760.62  stddev=693.99 bps=1219430.40
Time:1089209966   n=1001      avg=761.31  stddev=694.00 bps=1219315.20
Time:1089209971   n=1002      avg=760.62  stddev=693.99 bps=1219430.40
Time:1089209976   n=1001      avg=761.31  stddev=694.00 bps=1219315.20
Time:1089209981   n=1002      avg=760.62  stddev=693.99 bps=1219430.40
Time:1089209986   n=1002      avg=760.62  stddev=693.99 bps=1219430.40
Time:1089209991   n=1001      avg=761.31  stddev=694.00 bps=1219315.20
Time:1089209996   n=195       avg=747.51  stddev=692.02 bps=233222.40
Time:1089210001   n=1         avg=72.00   stddev=0.00   bps=115.20
Time:1089210006   n=2         avg=72.00   stddev=0.00   bps=230.40


Comienzo del envío: Wed, 7 Jul 2004 14:17:46 UTC

Fin del envío: Wed, 7 Jul 2004 14:20:06 UTC

Duracion 2 min 20 seg



Recepción Cliente 2


Time:1089221283   n=3         avg=533.33  stddev=652.42 bps=2560.00
Time:1089221288   n=1146      avg=757.08  stddev=694.06 bps=1388188.80
Time:1089221293   n=1001      avg=761.31  stddev=694.00 bps=1219315.20
Time:1089221298   n=1002      avg=760.62  stddev=693.99 bps=1219430.40
Time:1089221303   n=1001      avg=761.31  stddev=694.00 bps=1219315.20
Time:1089221308   n=1002      avg=760.62  stddev=693.99 bps=1219430.40
Time:1089221313   n=1001      avg=761.31  stddev=694.00 bps=1219315.20
Time:1089221318   n=1002      avg=760.62  stddev=693.99 bps=1219430.40
Time:1089221323   n=1002      avg=760.62  stddev=693.99 bps=1219430.40
Time:1089221328   n=1001      avg=761.31  stddev=694.00 bps=1219315.20
Time:1089221333   n=1002      avg=760.62  stddev=693.99 bps=1219430.40
Time:1089221338   n=1001      avg=761.31  stddev=694.00 bps=1219315.20
Time:1089221343   n=1002      avg=760.62  stddev=693.99 bps=1219430.40
Time:1089221348   n=1001      avg=761.31  stddev=694.00 bps=1219315.20
Time:1089221353   n=1002      avg=760.62  stddev=693.99 bps=1219430.40
Time:1089221358   n=1001      avg=761.31  stddev=694.00 bps=1219315.20
Time:1089221363   n=1002      avg=760.62  stddev=693.99 bps=1219430.40
Time:1089221368   n=1002      avg=760.62  stddev=693.99 bps=1219430.40
Time:1089221373   n=1001      avg=761.31  stddev=694.00 bps=1219315.20
```

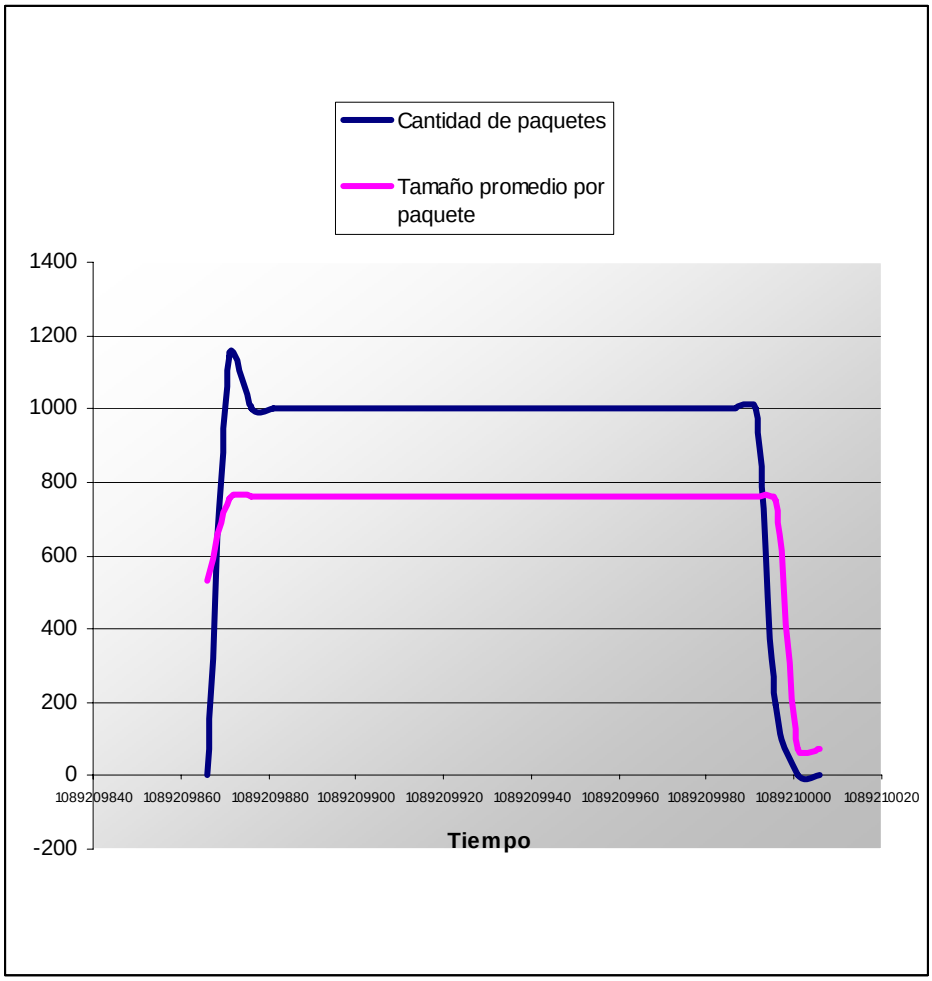
```
Time:1089221378   n=1002      avg=760.62  stddev=693.99 bps=1219430.40
Time:1089221383   n=1001      avg=761.31  stddev=694.00 bps=1219315.20
Time:1089221388   n=1002      avg=760.62  stddev=693.99 bps=1219430.40
Time:1089221393   n=1001      avg=761.31  stddev=694.00 bps=1219315.20
Time:1089221398   n=1002      avg=760.62  stddev=693.99 bps=1219430.40
Time:1089221403   n=1002      avg=760.62  stddev=693.99 bps=1219430.40
Time:1089221408   n=1001      avg=761.31  stddev=694.00 bps=1219315.20
Time:1089221413   n=195       avg=747.51  stddev=692.02 bps=233222.40
Time:1089221418   n=1         avg=72.00   stddev=0.00   bps=115.20
Time:1089221423   n=2         avg=72.00   stddev=0.00   bps=230.40
```

Comienzo de la recepción: Wed, 7 Jul 2004 17:28:03 UTC

Fin del envío: Wed, 7 Jul 2004 17:30:23 UTC

Duración: 2 min 20 segundos

Tiempo	Cantidad de paquetes	Tamaño promedio por paquete
1089209866	3	533,33
1089209871	1146	757,08
1089209876	1001	761,31
1089209881	1002	760,62
1089209886	1001	761,31
1089209891	1002	760,62
1089209896	1001	761,31
1089209901	1002	760,62
1089209906	1002	760,62
1089209911	1001	761,31
1089209916	1002	760,62
1089209921	1001	761,31
1089209926	1002	760,62
1089209931	1001	761,31
1089209936	1002	760,62
1089209941	1001	761,31
1089209946	1002	760,62
1089209951	1002	760,62
1089209956	1001	761,31
1089209961	1002	760,62
1089209966	1001	761,31
1089209971	1002	760,62
1089209976	1001	761,31
1089209981	1002	760,62
1089209986	1002	760,62
1089209991	1001	761,31
1089209996	195	747,51
1089210001	1	72
1089210006	2	72



Para el caso de transmisión utilizando TCP (Tranfer Control Protocol) los resultado obtenidos fueron totalmente distintos:

Desde el servidor (escuchando a los clientes)

Transferencia de archivos:

Cliente 1

```
Time:1089212993   n=16549 avg=1035.86     stddev=673.47bps=27427961.60
Time:1089212998   n=1294  avg=1032.72     stddev=674.24bps=2138137.60
Time:1089213003   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213008   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213013   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213018   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213023   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213028   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213033   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213038   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213043   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213048   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213053   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213058   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213063   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213068   n=8   avg=44.75   stddev=4.47 bps=572.80


Comienzo de la transmisión: Wed, 7 Jul 2004 15:09:53 UTC

Fin de la transferencia:    Wed, 7 Jul 2004 15:11:08 UTC


Tiempo: 1 minuto 15 segundos


Cliente 2:

Time:1089212992   n=17808 avg=1037.58 stddev=672.55  bps=29563638.40
Time:1089212997   n=1   avg=46.00   stddev=0.00 bps=73.60
Time:1089213002   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213007   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213012   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213017   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213022   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213027   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213032   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213037   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213042   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213047   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213052   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213057   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213062   n=8   avg=44.75   stddev=4.47 bps=572.80
```

```
Comienzo de la transmisión: Wed, 7 Jul 2004 15:09:52 UTC

Fin de la transferencia:    Wed, 7 Jul 2004 15:11:02 UTC

Tiempo: 1 minuto 10 segundos
```

Desde el cliente escuchando al servidor

Cliente 1:

```
Time:1089212782   n=23  avg=71.30   stddev=30.48      bps=2624.00
Time:1089212787   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212792   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212797   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212802   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212807   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212812   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212817   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212822   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212827   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212832   n=21  avg=74.38   stddev=30.16      bps=2499.20
Time:1089212837   n=3   avg=54.67   stddev=10.21      bps=262.40
Time:1089212842   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212847   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212852   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212857   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212862   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212867   n=3   avg=52.67   stddev=12.12      bps=252.80
Time:1089212872   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212877   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212882   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212887   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212892   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212897   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212902   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212907   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212912   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212917   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212922   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212927   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212932   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212937   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212942   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212947   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212952   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212957   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212962   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212967   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212972   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212977   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212982   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212987   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089212992   n=25947 avg=1036.34     stddev=674.06 bps=43023715.20
Time:1089212997   n=9705  avg=1033.82     stddev=675.37 bps=16053091.20
Time:1089213002   n=0   avg=0.00    stddev=0.00 bps=0.00
```

```
Time:1089213007   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213012   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213017   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213022   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213027   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213032   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213037   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213042   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213047   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213052   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213057   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213062   n=8   avg=47.00   stddev=2.65 bps=601.60
Time:1089213067   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089213072   n=8   avg=44.75   stddev=4.47 bps=572.80
```

Comienzo de la transmisión: Wed, 7 Jul 2004 15:06:22 UTC

Fin de la transferencia: Wed, 7 Jul 2004 15:11:12 UTC

Duración: 4 minutos 50 segundos

Cliente 2:

```
Time:1089224252   n=24  avg=71.08   stddev=29.98      bps=2729.60
Time:1089224257   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224262   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224267   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224272   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224277   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224282   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224287   n=3   avg=54.67   stddev=10.21      bps=262.40
Time:1089224292   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224297   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224302   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224307   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224312   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224317   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224322   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224327   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224332   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224337   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224342   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224347   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224352   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224357   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224362   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224367   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224372   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224377   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224382   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224387   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224392   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224397   n=0   avg=0.00    stddev=0.00 bps=0.00
```

```
Time:1089224402   n=0   avg=0.00    stddev=0.00 bps=0.00
Time:1089224407   n=19416     avg=1036.42 stddev=674.29 bps=32197120.00
Time:1089224412   n=16237     avg=1034.67 stddev=674.59 bps=26879961.60
```

Comienzo de la transmisión: Wed, 7 Jul 2004 18:17:32 UTC

Fin de la transferencia: Wed, 7 Jul 2004 18:20:12 UTC

Duración: 2 minutos 40 segundos

Cabe aclarar que los tiempos efectivos de transferencia son muy inferiores para TCP que para PGM. Se añade un trabajo adicional al respecto para aclarar este punto.

El siguiente trabajo muestra que, a través de TCP bajo plataforma Windows, la transferencia es mucho más rápida. Las desventajas se analizarán en las conclusiones.

Trabajo realizado sobre datos tomados desde el servidor FTP (SERVER.DUMP) y desde su cliente (CLIENTE.DUMP).

EL SERVIDOR TIENE DIRECCION DE IP: 192.168.4.160 (windows 2000 server)
EL CLIENTE TIENE DIRECCION DE IP: 192.168.4.161 (windows 2000 professional)

EL comando con el que se obtuvieron los datos de las maquinas bajo plataforma Microsoft Windows fue

c:\WINDUMP -i 2 -w [archivo en el que guardo la info] -p tcp

Luego se utilizaron los datos del archivo usando NETTRACE (una aplicación de linux) con el siguiente comando:

nettrace -lr [nombre del archivo]

SERVER.DUMP

Ostermann's tcptrace -- version 6.6.1 -- Wed Nov 19, 2003

16622 packets seen, 16622 TCP packets traced
elapsed wallclock time: 0:00:00.079313, 209574 pkts/sec analyzed
trace file elapsed time: 0:00:09.872585
TCP connection info:
2 TCP connections traced:
TCP connection 1:

host a: 192.168.4.161:1082
host b: 192.168.4.160:21
complete conn: no (SYNs: 0) (FINs: 0)

first packet: Fri Nov 12 19:13:26.174958 2004
last packet: Fri Nov 12 19:13:36.047543 2004

elapsed time: 0:00:09.872585

total packets: 7

filename: server.dump

a->b:		b->a:	
total packets:	4	total packets:	3
ack pkts sent:	4	ack pkts sent:	3
pure acks sent:	2	pure acks sent:	0
sack pkts sent:	0	sack pkts sent:	0
dsack pkts sent:	0	dsack pkts sent:	0
max sack blks/ack:	0	max sack blks/ack:	0
unique bytes sent:	43	unique bytes sent:	79
actual data pkts:	2	actual data pkts:	3
actual data bytes:	43	actual data bytes:	79
rexmt data pkts:	0	rexmt data pkts:	0
rexmt data bytes:	0	rexmt data bytes:	0
zwnd probe pkts:	0	zwnd probe pkts:	0
zwnd probe bytes:	0	zwnd probe bytes:	0
outoforder pkts:	0	outoforder pkts:	0
pushed data pkts:	2	pushed data pkts:	3
SYN/FIN pkts sent:	0/0	SYN/FIN pkts sent:	0/0
urgent data pkts:	0 pkts	urgent data pkts:	0 pkts
urgent data bytes:	0 bytes	urgent data bytes:	0 bytes
mss requested:	0 bytes	mss requested:	0 bytes
max segm size:	25 bytes	max segm size:	29 bytes
min segm size:	18 bytes	min segm size:	23 bytes
avg segm size:	21 bytes	avg segm size:	26 bytes
max win adv:	17221 bytes	max win adv:	64148 bytes
min win adv:	17142 bytes	min win adv:	64130 bytes
zero win adv:	0 times	zero win adv:	0 times
avg win adv:	17180 bytes	avg win adv:	64136 bytes
initial window:	25 bytes	initial window:	0 bytes
initial window:	1 pkts	initial window:	0 pkts
ttl stream length:	NA	ttl stream length:	NA
missed data:	NA	missed data:	NA
truncated data:	0 bytes	truncated data:	0 bytes
truncated packets:	0 pkts	truncated packets:	0 pkts
data xmit time:	0.006 secs	data xmit time:	9.684 secs
idletime max:	9714.0 ms	idletime max:	9671.9 ms
throughput:	4 Bps	throughput:	8 Bps
RTT samples:	2	RTT samples:	3
RTT min:	3.8 ms	RTT min:	2.5 ms
RTT max:	9.8 ms	RTT max:	184.6 ms
RTT avg:	6.8 ms	RTT avg:	109.9 ms
RTT stdev:	0.0 ms	RTT stdev:	95.3 ms
RTT from 3WHS:	0.0 ms	RTT from 3WHS:	0.0 ms
RTT full_sz smpls:	1	RTT full_sz smpls:	1
RTT full_sz min:	3.8 ms	RTT full_sz min:	142.5 ms
RTT full_sz max:	3.8 ms	RTT full_sz max:	142.5 ms

RTT full_sz avg:	3.8 ms	RTT full_sz avg:	142.5 ms
RTT full_sz stdev:	0.0 ms	RTT full_sz stdev:	0.0 ms
post-loss acks:	0	post-loss acks:	0
segs cum acked:	0	segs cum acked:	0
duplicate acks:	0	duplicate acks:	0
triple dupacks:	0	triple dupacks:	0
max # retrans:	0	max # retrans:	0
min retr time:	0.0 ms	min retr time:	0.0 ms
max retr time:	0.0 ms	max retr time:	0.0 ms
avg retr time:	0.0 ms	avg retr time:	0.0 ms
sdv retr time:	0.0 ms	sdv retr time:	0.0 ms

==========

TCP connection 2:

host c: 192.168.4.160:20
host d: 192.168.4.161:1084
complete conn: yes

first packet: Fri Nov 12 19:13:26.182005 2004
last packet: Fri Nov 12 19:13:35.859631 2004
elapsed time: 0:00:09.677626
total packets: 16615
filename: server.dump

c->d:		d->c:	
total packets:	10742	total packets:	5873
ack pkts sent:	10741	ack pkts sent:	5873
pure acks sent:	2	pure acks sent:	5871
sack pkts sent:	0	sack pkts sent:	0
dsack pkts sent:	0	dsack pkts sent:	0
max sack blks/ack:	0	max sack blks/ack:	0
unique bytes sent:	14660178	unique bytes sent:	0
actual data pkts:	10738	actual data pkts:	0
actual data bytes:	14660178	actual data bytes:	0
rexmt data pkts:	0	rexmt data pkts:	0
rexmt data bytes:	0	rexmt data bytes:	0
zwnd probe pkts:	0	zwnd probe pkts:	0
zwnd probe bytes:	0	zwnd probe bytes:	0
outoforder pkts:	0	outoforder pkts:	0
pushed data pkts:	3580	pushed data pkts:	0
SYN/FIN pkts sent:	1/1	SYN/FIN pkts sent:	1/1
req sack:	Y	req sack:	Y
sacks sent:	0	sacks sent:	0
urgent data pkts:	0 pkts	urgent data pkts:	0 pkts
urgent data bytes:	0 bytes	urgent data bytes:	0 bytes
mss requested:	1460 bytes	mss requested:	1460 bytes
max segm size:	1460 bytes	max segm size:	0 bytes
min segm size:	594 bytes	min segm size:	0 bytes
avg segm size:	1365 bytes	avg segm size:	0 bytes
max win adv:	64240 bytes	max win adv:	17520 bytes
min win adv:	64240 bytes	min win adv:	776 bytes
zero win adv:	0 times	zero win adv:	0 times
avg win adv:	64240 bytes	avg win adv:	15701 bytes
initial window:	2920 bytes	initial window:	0 bytes

initial window:	2 pkts	initial window:	0 pkts
ttl stream length:	14660178 bytes	ttl stream length:	0 bytes
missed data:	0 bytes	missed data:	0 bytes
truncated data:	14209182 bytes	truncated data:	0 bytes
truncated packets:	10738 pkts	truncated packets:	0 pkts
data xmit time:	9.651 secs	data xmit time:	0.000 secs
idletime max:	371.0 ms	idletime max:	267.6 ms
throughput:	1514853 Bps	throughput:	0 Bps
RTT samples:	5608	RTT samples:	2
RTT min:	0.1 ms	RTT min:	0.1 ms
RTT max:	200.0 ms	RTT max:	0.1 ms
RTT avg:	1.4 ms	RTT avg:	0.1 ms
RTT stdev:	12.7 ms	RTT stdev:	0.0 ms
RTT from 3WHS:	0.2 ms	RTT from 3WHS:	0.1 ms
RTT full_sz smpls:	3766	RTT full_sz smpls:	1
RTT full_sz min:	0.2 ms	RTT full_sz min:	0.1 ms
RTT full_sz max:	103.3 ms	RTT full_sz max:	0.1 ms
RTT full_sz avg:	0.4 ms	RTT full_sz avg:	0.1 ms
RTT full_sz stdev:	1.7 ms	RTT full_sz stdev:	0.0 ms
post-loss acks:	0	post-loss acks:	0
segs cum acked:	5132	segs cum acked:	0
duplicate acks:	47	duplicate acks:	1
triple dupacks:	0	triple dupacks:	0
max # retrans:	0	max # retrans:	0
min retr time:	0.0 ms	min retr time:	0.0 ms
max retr time:	0.0 ms	max retr time:	0.0 ms
avg retr time:	0.0 ms	avg retr time:	0.0 ms
sdv retr time:	0.0 ms	sdv retr time:	0.0 ms

CLIENTE.DUMP

Ostermann's tcptrace -- version 6.6.1 -- Wed Nov 19, 2003

16627 packets seen, 16627 TCP packets traced
elapsed wallclock time: 0:00:00.296161, 56141 pkts/sec analyzed
trace file elapsed time: 0:00:09.872539

TCP connection info:

3 TCP connections traced:

TCP connection 1:
host a: 192.168.4.161:1082
host b: 192.168.4.160:21
complete conn: no (SYNs: 0) (FINs: 0)
first packet: Fri Nov 12 19:15:11.108221 2004
last packet: Fri Nov 12 19:15:20.980760 2004
elapsed time: 0:00:09.872539
total packets: 7
filename: cliente.dump

a->b:		b->a:	
total packets:	4	total packets:	3
ack pkts sent:	4	ack pkts sent:	3
pure acks sent:	2	pure acks sent:	0
sack pkts sent:	0	sack pkts sent:	0
dsack pkts sent:	0	dsack pkts sent:	0
max sack blks/ack:	0	max sack blks/ack:	0
unique bytes sent:	43	unique bytes sent:	79
actual data pkts:	2	actual data pkts:	3
actual data bytes:	43	actual data bytes:	79
rexmt data pkts:	0	rexmt data pkts:	0
rexmt data bytes:	0	rexmt data bytes:	0
zwnd probe pkts:	0	zwnd probe pkts:	0
zwnd probe bytes:	0	zwnd probe bytes:	0
outoforder pkts:	0	outoforder pkts:	0
pushed data pkts:	2	pushed data pkts:	3
SYN/FIN pkts sent:	0/0	SYN/FIN pkts sent:	0/0
urgent data pkts:	0 pkts	urgent data pkts:	0 pkts
urgent data bytes:	0 bytes	urgent data bytes:	0 bytes
mss requested:	0 bytes	mss requested:	0 bytes
max segm size:	25 bytes	max segm size:	29 bytes
min segm size:	18 bytes	min segm size:	23 bytes
avg segm size:	21 bytes	avg segm size:	26 bytes
max win adv:	17221 bytes	max win adv:	64148 bytes
min win adv:	17142 bytes	min win adv:	64130 bytes
zero win adv:	0 times	zero win adv:	0 times
avg win adv:	17180 bytes	avg win adv:	64136 bytes
initial window:	25 bytes	initial window:	0 bytes
initial window:	1 pkts	initial window:	0 pkts
ttl stream length:	NA	ttl stream length:	NA
missed data:	NA	missed data:	NA
truncated data:	0 bytes	truncated data:	0 bytes
truncated packets:	0 pkts	truncated packets:	0 pkts
data xmit time:	0.006 secs	data xmit time:	9.684 secs
idletime max:	9714.0 ms	idletime max:	9671.9 ms
throughput:	4 Bps	throughput:	8 Bps

RTT samples:	2	RTT samples:	3
RTT min:	3.9 ms	RTT min:	2.4 ms
RTT max:	9.9 ms	RTT max:	184.5 ms
RTT avg:	6.9 ms	RTT avg:	109.7 ms
RTT stdev:	0.0 ms	RTT stdev:	95.3 ms
RTT from 3WHS:	0.0 ms	RTT from 3WHS:	0.0 ms
RTT full_sz smpls:	1	RTT full_sz smpls:	1
RTT full_sz min:	3.9 ms	RTT full_sz min:	142.4 ms
RTT full_sz max:	3.9 ms	RTT full_sz max:	142.4 ms
RTT full_sz avg:	3.9 ms	RTT full_sz avg:	142.4 ms
RTT full_sz stdev:	0.0 ms	RTT full_sz stdev:	0.0 ms
post-loss acks:	0	post-loss acks:	0
segs cum acked:	0	segs cum acked:	0
duplicate acks:	0	duplicate acks:	0
triple dupacks:	0	triple dupacks:	0
max # retrans:	0	max # retrans:	0
min retr time:	0.0 ms	min retr time:	0.0 ms
max retr time:	0.0 ms	max retr time:	0.0 ms
avg retr time:	0.0 ms	avg retr time:	0.0 ms
sdv retr time:	0.0 ms	sdv retr time:	0.0 ms

==========

TCP connection 2:

host c: 192.168.4.160:20
host d: 192.168.4.161:1084
complete conn: yes
first packet: Fri Nov 12 19:15:11.115387 2004
last packet: Fri Nov 12 19:15:20.792951 2004
elapsed time: 0:00:09.677563
total packets: 16615
filename: cliente.dump

c->d:		d->c:	
total packets:	10742	total packets:	5873
ack pkts sent:	10741	ack pkts sent:	5873
pure acks sent:	2	pure acks sent:	5871
sack pkts sent:	0	sack pkts sent:	0
dsack pkts sent:	0	dsack pkts sent:	0
max sack blks/ack:	0	max sack blks/ack:	0
unique bytes sent:	14660178	unique bytes sent:	0
actual data pkts:	10738	actual data pkts:	0
actual data bytes:	14660178	actual data bytes:	0
rexmt data pkts:	0	rexmt data pkts:	0
rexmt data bytes:	0	rexmt data bytes:	0
zwnd probe pkts:	0	zwnd probe pkts:	0
zwnd probe bytes:	0	zwnd probe bytes:	0
outoforder pkts:	0	outoforder pkts:	0
pushed data pkts:	3580	pushed data pkts:	0
SYN/FIN pkts sent:	1/1	SYN/FIN pkts sent:	1/1
req sack:	Y	req sack:	Y
sacks sent:	0	sacks sent:	0
urgent data pkts:	0 pkts	urgent data pkts:	0 pkts
urgent data bytes:	0 bytes	urgent data bytes:	0 bytes
mss requested:	1460 bytes	mss requested:	1460 bytes
max segm size:	1460 bytes	max segm size:	0 bytes

min segm size:	594 bytes	min segm size:	0 bytes
avg segm size:	1365 bytes	avg segm size:	0 bytes
max win adv:	64240 bytes	max win adv:	17520 bytes
min win adv:	64240 bytes	min win adv:	776 bytes
zero win adv:	0 times	zero win adv:	0 times
avg win adv:	64240 bytes	avg win adv:	15701 bytes
initial window:	2920 bytes	initial window:	0 bytes
initial window:	2 pkts	initial window:	0 pkts
ttl stream length:	14660178 bytes	ttl stream length:	0 bytes
missed data:	0 bytes	missed data:	0 bytes
truncated data:	14209182 bytes	truncated data:	0 bytes
truncated packets:	10738 pkts	truncated packets:	0 pkts
data xmit time:	9.651 secs	data xmit time:	0.000 secs
idletime max:	371.0 ms	idletime max:	267.5 ms
throughput:	1514862 Bps	throughput:	0 Bps
RTT samples:	5608	RTT samples:	2
RTT min:	0.0 ms	RTT min:	0.1 ms
RTT max:	199.7 ms	RTT max:	0.2 ms
RTT avg:	1.0 ms	RTT avg:	0.2 ms
RTT stdev:	12.7 ms	RTT stdev:	0.0 ms
RTT from 3WHS:	0.1 ms	RTT from 3WHS:	0.1 ms
RTT full_sz smpls:	3766	RTT full_sz smpls:	1
RTT full_sz min:	0.0 ms	RTT full_sz min:	0.2 ms
RTT full_sz max:	103.1 ms	RTT full_sz max:	0.2 ms
RTT full_sz avg:	0.1 ms	RTT full_sz avg:	0.2 ms
RTT full_sz stdev:	1.7 ms	RTT full_sz stdev:	0.0 ms
post-loss acks:	0	post-loss acks:	0
segs cum acked:	5132	segs cum acked:	0
duplicate acks:	47	duplicate acks:	1
triple dupacks:	0	triple dupacks:	0
max # retrans:	0	max # retrans:	0
min retr time:	0.0 ms	min retr time:	0.0 ms
max retr time:	0.0 ms	max retr time:	0.0 ms
avg retr time:	0.0 ms	avg retr time:	0.0 ms
sdv retr time:	0.0 ms	sdv retr time:	0.0 ms

Hasta aquí, los resultados de las experiencias realizadas y sus descripciones. En el próximo capítulo se exponen las conclusiones finales.

Capítulo 8: Conclusiones

Conclusiones finales

Una vez finalizadas las pruebas de laboratorio y de acuerdo al marco teórico presentado previamente, se concluye:

1. PGM utiliza racionalmente el ancho de banda efectivo en la transferencia de archivos y lo mantiene constante a lo lago de todo el proceso de transferencia. Según se observa, cada cliente recibió a una tasa constante, sin saltos, los archivos transferidos.

2. PGM asegura que los archivos sean transferidos **simultáneamente** a todos los clientes integrados al grupo. Cada uno de ellos recepcionó **en el mismo intervalo de tiempo** la información transferida. Considerando el punto 1, todos los clientes recibieron la transferencia **simultáneamente, a tasa constante**.

3. No es posible lograr estas características de transferencias de archivos bajo otros protocolos de transporte, como se observa en la experiencia de transferencia realizada a través de TCP:

4. A pesar de todo, es importante aclarar que PGM se encuentra en estado experimental aún para muchas aplicaciones de uso masivo y tampoco puede ser utilizado en múltiples plataformas de la misma manera que otros protocolos.

5. La transferencia realizada sobre TCP muestra que: se disminuye el tiempo de transferencia pero, al no contar con un mecanismo explícito para el control de congestionamiento, puede suceder el evento de sobrecarga de las máquinas intermedias (esta es una característica intrínseca de TCP). Para ello, la medida inmediata a la que recurre es reducir la velocidad de transmisión, dependiendo de la implementación.

6. En el momento del envío, la transmisión no es simultánea a todos los usuarios bajo TCP. De esta manera, los recursos son utilizados monopólicamente por cada conexión.

7. PGM, al asegurar una recepción simultánea de todos los receptores, especifica un servicio de transferencia de archivos que no puede ser comparado con TCP ya que no soporta el retardo producido por los ACK.

8. La adopción de PGM para la transferencia de archivos es una de las posibilidades que se evalúan hoy en los laboratorios donde fuera realizado el trabajo y que poseen FreeBSD como plataforma. Este sería el paso inicial para una implementación de PGM más ambiciosa que incluiría la transmisión de voz y video.

"...Otro nuevo servicio que presenta nuevos requerimientos para redes, tanto de empresas como del proveedor de servicios abarca la capacitación y las conferencias vía Internet. Para que las redes que soportan dichas aplicaciones resulten costo efectivas, se requieren IP Multicast y protocolos tales como Pragmatic General Multicast (PGM), Multicast Source Discovery Protocol (MSDP), y multicast BGP (mBGP). PGM simplifica la operación de la red, al mismo tiempo que permite la entrega de información multicast en forma ordenada y libre de duplicaciones de una o múltiples fuentes a múltiples clientes..."

Fuente: Cisco Systems – White paper -www.cisco.com/global/LA/ assets/pdfs/beneficios_tecnicos.pdf

Bibliografía y Webliografía

Distributed Systems – Concepts and Design – G. Colouris, J. Dollmore, T. Kindberg.- Second Edition -Addison Wesley

Developing IP Multicast Networks – Cisco Systems – Beau Williamson -Cisco press.

Manual de FreeBSD – FreeBSD Documentation Project.

RFC 3208 – PGM Reliable Transport Protocol Specification – T.Spakman, J. Crowcroft y otros.

RFC 959 - File Transfer Protocol -J. Postel, J. Reynolds

RFC 793 - TRANSMISSION CONTROL PROTOCOL - DARPA INTERNET PROGRAM

RFC 2236 - Internet Group Management Protocol - W. Fenner - Xerox PARC

L. Rizzo, "PGMCC: A TCP-friendly Single-Rate Multicast Congestion Control Scheme", Proc. of ACM SIGCOMM -2000.

L. Rizzo, "A PGM Host Implementation for FreeBSD", http://www.iet.unipi.it/~luigi/pgm.html

M. Psaltaki, R. Araujo, G. Aldabbagh, P. Kouniakis, and A. Giannopoulos, "Pragmatic General Multicast (PGM) host implementation for FreeBSD.", http://www.cs.ucl.ac.uk/research/darpa/pgm/PGM_FINAL.html

The PGM Reliable Multicast Protocol – J. Gemmell, T.Speakman, J. Crowcroft y otros.

Implementation and Evaluation of Pragmatic general Multicast – Derek Chen-Becker, Manj Singla.- 2002

TIBCO® SmartPGM and TIBCO® SmartPGM FX - http://www.tibco.com/software/enterprise_backbone/smartpgm.jsp - www.tibco.com

CISCO IOS SOFTWARE RELEASES 12.0 T- PGM Router Assist - http://www.cisco.com/en/US/products/sw/iosswrel/ps1830/products_feature_guide09186a00800879a2.html#wp5038

http://www.rediris.es/rediris/boletin/54-55/ponencia12.html - Uso de IGMPv3 para evitar ataques DoS en redes multicast- . F. Gómez Skarmeta,A. L. Mateo, P. M. Ruíz

N. Ishikawa, N. Yamanouchi, O. Takahashi. "IGMP Extension for Authentication of IP Multicast Senders and Receivers". Internet-Draft. Agosto 1998.
W. Fenner. "Internet Group Management Protocol. Version 2". RFC 2236. Noviembre 1997.

A. Gómez Skarmeta, A. L. Mateo Martínez, P. M. Ruiz Martínez. "Access Control in Multicast Environments: an Approach to Senders Authentication". Proceedings of the IEEE LANOMS'99, pp 1-13. 1999

B. Cain, S. Deering, A. Thyagarajan. "Internet Group Management Protocol. Version 3". Internet-Draft. Noviembre 1999.

B. Quinn. "SDP Source-Filters". Internet-Draft. Mayo 2000.
M. Handley, V. Jacobson. "SDP: Session Description Protocol". RFC 2327. Abril 1998.

S. Kent, R. Atkinson. "Security Architecture for the Internet Protocol". RFC 2401. Noviembre 1998.