INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

N° assigned by the library

THESIS

for obtaining the degree of

DOCTOR OF THE INPG

Speciality: ≪ Computer Network ≫

prepared at INRIA Rhône Alpes, Planete project-team
in the executive of **l'Ecole Doctorale ≪ MATHEMATIQUE, SCIENCES ET TECHNOLOGIE DE L'INFORMATION ≫**

presented and publicly discussed

by

Ayman El-Sayed Ahmed **EL-SAYED**

Defensed on March 8, 2004

Title:

Application-Level Multicast Transmission Techniques Over The Internet

commitee in charge

Prof.	Jacques MOSSIERE	President
Prof.	Eric FLEURY	Reporter
Prof.	Jean-Jacques PANSIOT	Reporter
Prof.	Andrzej DUDA	Director of thesis
Dr.	Vincent ROCA	Supervisor
Prof.	Stanislaw BUDKOWSKI	Examiner

INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

N° attribué par la bibliothèque

THÈSE

pour obtenir le grade de

DOCTEUR DE L'INPG

Spcialité: ≪Réseau Informatique≫

préparée à l'INRIA Rhône Alpes, projèt Planète

dans le cadre de **l'Ecole Doctorale ≪ MATHEMATIQUE, SCIENCES ET TECHNOLOGIE DE L'INFORMATION ≫**

présentée et soutenue publiquement

par

Ayman El-Sayed Ahmed **EL-SAYED**

le Mars 8, 2004

Titre:

Techniques de Transmission Multipoint de Niveau Applicatif pour l'Internet

jury

Prof.	Jacques MOSSIERE	Président
Prof.	Eric FLEURY	Rapporteur
Prof.	Jean-Jacques PANSIOT	Rapporteur
Prof.	Andrzej DUDA	Directeur de thèse
Dr.	Vincent ROCA	Encadrament scientifique
Prof.	Stanislaw BUDKOWSKI	Examinateur

Acknowledgment

I would like to express my sincere thanks to my supervisor Dr. Vincent ROCA for his invaluable help, advice, guidance and encouragement throughout the development of this work. I would like to thank Andrzej DUDA for his help ,and advice.
All my gratitude to Eric FLEURY and Jean-Jacques PANSIOT for the time they spent reading and commenting my thesis, and for the report they wrote. And I also would like to thank Stanislaw BUDKOWSKI and Jacques MOSSIERE for taking part to the committee.
I would also like to thank all those who helped, advised, and gave fruitful discussions during this work. Many thanks also are to all the member of the Planete project in Inria Rhone Alpes.
Finally, my sincere appreciation and gratitude to my parents, and my family (my wife, my sons, and my daughter).

Our Achievements: Published Papers and Implementations

Magazine Papers

1. Ayman El-Sayed, Vincent Roca, and Laurent Mathy, **A Survey of Proposals for an Alternative Group Communication Service**, IEEE Network, Special Issue on Multicasting: An Enabling Technology, January/Febrary, 2003.

Conference Papers

1. Laurent Mathy, Nick Blundell, Vincent Roca, and Ayman El-Sayed, **Impacts of Simple Cheating in Application-Level Multicast**, IEEE INFOCOM'04, Hong Kong, March 2004.

2. Lina Al-Chaal, Vincent Roca, Ayman El-Sayed, and Michel Habert, **A VPRN Solution for Fully Secure and Efficient Group Communications**, IEEE Symposium on Computers and Communications - ISCC'2003, Kemer-Antalya, Turkey,july, 2003. An extended version is available as INRIA Research Report number RR-4799, INRIA, Rhône Alpes, France, April, 2003.

3. Ayman El-Sayed and Vincent Roca, **Improving the Scalability of an Application Level Group Communication Protocol**, 10th IEEE International Conference on Telecommunications (ICT'03), Papeete, French Polynesia" February, 2003.

4. Vincent Roca and Ayman El-Sayed, **A Host-Based Multicast (HBM) Solution for Group Communications**, First IEEE International Conference on Networking (ICN'01), Colmar, France, July, 2001.

5. Ayman El-Sayed and Vincent Roca, **On Robustness in Application-Level Multicast: the case of HBM**, the ninth IEEE Symposium on Computer and Communications (ISCC'2004), Alexandria, Egypt, 29 June- 1^{st} July, 2004.

Published Papers about Qos and ATM (out of the scope of this thesis)

1. Ayman El-Sayed and Ehab A. Khalil, **Integration of Voice and Data in ATM Ring Networks**, Published in Telecommunication Information Management Journal Issue 8, Winter 2002.

2. Ayman El-Sayed and Ehab A. Khalil, **Performance Evaluation of Video/Voice/Data Integration Over VP-Based ATM Ring Network**, Published in Telecommunication Information Management Journal Issue 8, Winter 2002.

3. Ayman El-Sayed, Ehab A. Khalil, Nabil Ismail, and Ibrahim Z. Morsi, **Multimedia Traffic over VP-Based ATM Ring Network**, Published in 19th IEEE/IPCCC-2000, Arizona, USA, Feburary 20-22, 2000.

4. Ayman El-Sayed, Ehab A. Khalil, Nabil Ismail, and Ibrahim Z. Morsi, **Control Mechanism for Fairness Among Traffics on ATM Network**, Published in 18th IASTED Intal. Conf. AI2000, Austria,Febraury 14-17, 2000.

Software (C++/Linux)

1. **Group Communication Service Library (GCSL) implementing the HBM proposal**.

2. **Simulator for analyzing the overlay topology**.

3. **Simulator for the impacts of cheating members**.

Abstract

In this dissertation, we introduce a proposal for building an alternative group communication service that shifts the multicast support from core routers to end-systems. Our proposal, called Host Based Multicast (HBM), operates at application-level and provides an efficient multi-point data distribution service for one-to-many or many-to-many communications. With this approach end-hosts (running the application), dedicated servers and/or border routers automatically self-organize into an overlay distribution topology where data is disseminated. This overlay topology can be composed of both unicast connections and native multicast islands (e.g. within each site). Therefore it offers a group communication service to all hosts, even those located in a site that does not have access for any reason, to native multicast routing.
HBM is a centralized solution, where everything, including group membership management and overlay topology creation, is under the control of a single Rendez-vous Point (RP). This thesis focuses on three key aspects: scalability, robustness, and security. The scalability can be largely improved, with a few simple HBM protocol parameter adjustments like the frequency and size of the control messages exchanged. Robustness is another important practical issue, and we introduce packet loss reduction techniques, like the addition of redundant virtual links that avoid topology partition in case of transit node failures. Finally we investigate the use of HBM to build a fully secure but efficient group communication service between several sites using an IPSec VPN environment. We show that HBM and the IPSec VPN environment naturally fit with one-another and lead to the concept of Virtual Private Routed Network (VPRN).

Résumé

Dans cette thèse, nous présentons une proposition pour établir un service alternatif de communication de groupe qui déplace le support de multipoint depuis les routeurs vers les extrémités. Notre proposition, appelée HBM fonctionne au niveau applicatif et fournit un service efficace de distribution multipoint de données de type un-vers-plusieurs ou plusieurs-vers-plusieurs. Avec cette approche les machines terminelles, les serveurs et/ou des routeurs de bordure s'organisent automatiquement en une topologie de distribution de recouvrement grace à laquelle des données sont diffusées. Cette topologie de recouvrement peut se composer à la fois de connections point à point et de zones bénéficiant d'un routage multicast natif (par exemple dans chaque site). Par conséquent elle offre un service de communication de groupe à tous les hôtes, même ceux situés dans un site qui n'a pas accès, pour une quelconque raison, au routage multicast natif.
HBM est une solution centralisée, où tout, y compris la gestion d'adhésion au groupe et la création de topologie de recouvrement, est de la responsabilité d'un unique point de Rendez-vous (RP). Cette thése se concentre sur trois aspects principaux: scalabilité, robustesse et sécurité. La scalabilité peut être en grande partie améliorée, avec quelques ajustements simples de certains paramètres de HBM tels la fréquence et la taille des messages de contrôle. La robustesse est un aspect pratique important, et nous présentons des techniques de réduction de pertes de paquet, tels l'addition des liens virtuels redandants qui évitent la partition de topologie en cas de panne d'un noeud de transit. Enfin nous étudions l'utilisation de HBM afin d'établir un service entièrement sécurisé mais efficace de communication de groupe entre plusieurs sites au moyen de VPN IPSec. Nous montrons que HBM et un environnement de VPN IPSec sont naturellement complementaires et conduisent au concept de réseau privé virtuel routé (VPRN).

ملخص البحث

يعتبر علم شبكات الحاسبات من اهم العلوم المتعلقه بالحاسب فى العصر الحديث، لانها تتيح فرصه الاتصال و تبادل المعلومات بين مستخدمى الانظمه المتباعده بسهوله و يسر, و مع انتشار الوسائط المتعدده (Multimedia) واستخدامها فى العديد من المجالات و خاصه التعليم عن بعد و كثير من التطبيقات الحيويه.هذا فى حاله الاتصال من نقطه الى اخري (point-to-point Unicast) كما فى الشبكه العالميه (Internet). و زادت الحاجه ايضا لنقل المعلومه لاكثر من واحد فى نفس الوقت تسمى هذه التقنيه الانتقال المتعدد بين المجموعه الواحده (Multicast) و هذا يتم فى مستوى الشبكه (Network Layer).

و اصبح الانتقال المتعدد Multicast من الامور الهامه فى عصرنا هذا نظرا من الاحتياجات لنقل الوسائط المتعدده مثل الصوت و الفديو والمعلومات لاكثر من مستقبل فى ان واحد. و لكن الانتقال المتعدد فى مستوى الشبكه يقابل بعض المشاكل مثل: ليس كل الـ Routers فى الشبكه العالميه مدعم بامكانيه الانتقال المتعدد وهذه الامكانيه صعبه الانتشار فى الشبكه العالميه بسبب عوامل اقتصاديه و تسوقيه و لا يمكن تغير الشبكه العالميه مره واحده.

ولذلك عملنا معاينه (Survey) لمعظم المقترحات البديله وتصنيفهم الى خمس مجموعات:المجموعه الاولى خاصه بـ الـ Reflector and Punctual Tunneling و المجموعه الثانيه خاصه بـ Permanent Tunneling و كل من المجموعتان السابقتان يضبطان يدويا. اما المجموعه الثالثه فهى انشاء عده اتصالات خياليه (Unicast) بين الاعضاء فى مجموعه الاتصال Automatic Overlay Multicast. اما المجموعه الرابعه و الخامسه خاصه بـ الاتصال peer-to-peer و الاعتماد على Gossiping بالنسبه الى كل منهم.

وفى هذا البحث ركزنا على المقترحات التى تنفذ فى مستوى التطبيق وجدنا انهم ينقسموا الى مجموعتين: مركزيه معلومات التحكم - موزعه معلومات التحكم. بالنسبه الى المقترحات الخاصه بمركزيه معلومات التحكم فمنها كامل المركزيه مثل مقترحنا وجزئى المركزيه.

مقترحنا فى مستوى التطبيق يسمى Host Based Multicast (HBM) وهو عباره عن تطبيق فى مستوى التطبيق ولا يعتمد على اى تدعيم الانتقال المتعدد فى الشبكه العالميه.

مقترحنا من نوع مركزى معلومات التحكم، ويتكون من عنصر اساسى ويكون مسؤل عن كل شئ يسمى نقطه تلاقى Rendez-Vous Point (RP). يوجد نوعان من الاتصال بين الاعضاء فى المجموعه الواحده، اتصال بين RP والاعضاء فى المجموعه لتبادل رسائل التحكم و هذا الاتصال من نوع TCP/IP Unicast Connection. اما النوع الثانى فهو بين الاعضاء لتبادل المعلومات عن طريق اتصالات بينهم، من خلال Overlay الذى انشئ بواسطه RP، يكون هذا الاتصال من نوع UDP/IP Connection. كل عضويقوم دوريا باعاده تقيم الـ Metrics بينه وبين الاخرين فى نفس المجموعه للاخذ فى الاعتبار تغيرات الـ Metrics فى الشبكه العالميه لانشاء Overlay ينطبق مع الشروط

الحاليه للشبكه العالميه. ونظرا لهذا،الـ RP يقوم دوريا باعاده انشاء الـ Overlay لتتماشى مع الشروط الحاليه للشبكه العالميه.

هذا البحث يركز على اربعه امورمختلفه كما يلى:

- كيفيه انشاء الـ overlay التى تضم الاعضاء فى مجموعه واحده لتبادل المعلومات.
- اكبر عدد ممكن يكون فى المجموعه بدون مشاكل وتسمى Scalability.
- مدى متانه الـ Overlay من التهشم او الانقسام وتسمى Robustness.
- حمايه المجموعه من دخيل غير مصرح له بالانضمام لها و تسمى Security.

بالنسبه لكيفيه انشاء الـ Overlay، فى هذا البحث ناقشنا كيفيه انشاء الربط بين الاعضاء المتماثله والغير المتماثله مع الاخذ فى الاعتبار عده Metrics مثل RTT و نسبه الفقد فى الاتصال Loss، و يوجد ايضا عده Metrics اخرى ايضا يمكن اخذها فى الاعتبار مثل مدى استقرار العضو و كذلك ما هى طريقه اتصاله بـ الـ Internet اما عن طريق شبكه مباشرا او عن طريق Modem فيكون منخفض الـ Bandwidth. و من هؤلاء الـ Metrics يمكن معرفت العضويكون وسيط فى الـ Overlay او فى اخره. و باضافه امكانيه تحديد اقصى عدد للجيران (Fanout) لكل عضو يمكن انشاء الـ Overlay عاليه الكفائه.

نظرا لطبيعه مركزيه مقترحنا، يكون محدود فى عدد الاعضاء فى المجموعه وهذا لان تحديث الـ Metrics لتحسين الـ Overlay تهدر بعض الـ Bandwidth فى الشبكه وكذلك تهدر بعض الوقت والطاقه للاعضاء و كذلك لـRP ايضا. و بجانب هذا الاعضاء دوريا تحدث الـ Metrics. و لذلك فى هذا البحث بينا كيفه تحسين الـScalability بواسطه تحديد عدد و طول رسائل التحكم المتبادله بين الـ RP و الاعضاء، و يوجد ايضا فى هذا البحث اربعه استراتيجيات لتحسين الـ Scalability ووجدنا بعد دراستنا للاربع استراتيجيات ان الاستراتيجيه التى يكون فيها تثبيت طول رساله التحكم عند قيمه معينه بحيث ان معدل ارسال رسائل التحكم المتبادله لكلا من الـ RP و الاعضاء لا تذيد عن نسبه تساوى 5 % من مجموع معدل كلا من ارسال\استقبال رسائل التحكم وارسال المعلومات بين الاعضاء. و عندما تصل الى هذا الحد، هذه الاستراتيجيه تثبت طول رساله التحكم و تذيد من الفتره المتباينه بين كل رساله و هذا لتثبيت معدل ارسال\استقبال رسائل التحكم عند حد معين لا يذيد عنه.

نظرا لان مقترحنا يعمل فى مستوى التطبيق فلابد من وجود عضو من اعضاء المجموعه غير مستقر اى ان ممكن ان يقف عن العمل بسبأ ما, وبالتالى الـ Overlay ينقسم الى جزئين. بالتالى اضفنا عدد من الروابط بين الاعضاء الى الروابط الاساسيه داخل الـ Overlay تسمى Redundant Vitual Links (RVL). نتيجه لهذه الاضافه، ممكن حدوث دوران للمعلومه فى دائره مغلقه تسمى Looping وللذلك اضفنا تكنيك لتفادى هذا الدوران للمعلومه بواسطه:

- اذا العضو يستقبل المعلومه من الـ Overlay و ايضا من الـ RVL, هذا العضو يرسل رساله تحكم الى الجار المشترك فى RVL لايقاف الارسال فى هذا الرابطه لمده عدة ثوانى.

- اذا العضو يستقبل معلومه من خلال الـ RVL وليس من خلال الـ Overlay, اذن يوجد مشكله و لذلك يستمر العضو فى الاستقبال و توصيله الى باقى الجيران فى الـ Overlay و ايضا ينتظر بعض الوقت ويرسل رساله تحكم الى الـ RP ليبلغه انه يوجد مشكله فى الـ Overlay اذا كانت المشكله مستمره لفتره.

فى هذا البحث قدمنا خمس خورزميات Algorithms لاضافه روابط اضافيه الى الـ Overlay. و بعد دراستنا للخمس خورزميات وجدنا انهم يقلوا من احتماليه انقسام الـ Overlay بنسب مختلفه و لكن يوجد عدة عوامل للمقارنه و منها وجدنا ان واحد منهم ,الذى يضيف بعض الروابط بين الاعضاء الوسيطه فقط (اى الاعضاء التى تربط اعضاء ببعض و التى تملك قدره عاليه ونطاق واسع للارسال و الاستقبال), يقلل من احتماليه انقسام الـ Overlay. وجدنا ايضا انه يقلل احتماليه انقسام الـ Overlay عندما يوجد اثنان او ثلاثه اعضاء خرجوا من العمل بسبا ما.

نظرا لان مقترحنا يعمل على تحسين الـ Overlay دوريا و الذى يقوم به الـ RP ثم ابلاغ كلا من الاعضاء. و نظرا لاختلاف المسافات بين الـ RP و الاعضاء فليس من الممكن وصول رساله التحكم لتحديث الـ Overlay فى آن واحد لكل الاعضاء. نتيجه لذلك يوجد بعض المعلومات تفقد اثناء الفتره الانتقاليه وهى الفتره اللازمه لوصول رساله التحكم للتحديث الى كل الاعضاء و كذلك لتحديث الـ Overlay لكل الاعضاء. لذلك فى هذا البحث ناقشنا هذه المشكله (اى تقليل احتماليه فقد اى معلومه اثناء تغيراو تحديث الـ Overlay)، واقترحنا عدة استراتيجيات لتقليل نسبه فقد المعلومه. للتغلب على هذه المشكله لابد من استخدام رقم دوري لكلا من الـOverlay يسمى TSN يتغيرمع تغيرالـ Overlay وكذلك البكت المعلومه يسمىPSN .

و بعد دراستنا لكل الاستراتيجيات وجدنا ان استراتيجى رقم اثنان هى الافضل بالمقارنه بالاخرين باستخدام عده عوامل نوقشه فى هذا البحث. و فى الاستراتيجى الثانيه يحتفظ كل عضومن الاعضاء اثنين من الـ Overlay وهم السابق و الحالى معا، ثم يتم تحديثهم عند استلام رسالهالتحكم للتحديث فالحاليه تصبح السابق و كذلك الجديد يصبح الحالى و هكذا. المهم الان كيف ندير هذا؟ عندما العضو يستقبل بكت المعلومه اولاً يجب ان يعرف الـ PSN اذا كنت هذه البكت استقبلت من قبل، فيسقطها و بعدها لايفعل شئ، و اذا لم تستقبل من قبل فلابد من معرف الـ TSN فيوجد ثلاث احتمالات لـ TSN:

- اما لا تكون تساوى الـ TSN الحاليه و لا السابقه، اذن لابد من اسقاطها ايضا,
- اما تكون تساوى الـ TSN الحاليه او السابقه، اذن فتعاد ارسالها من خلال الـ Overlay الحالى او السابق على التوالى.

و باستخدام الاستراتيجيه الثانيه يمكن ان نقول ان معدل فقد بكت المعلومه يكاد ان يكون صفر وكذلك معدل ازدواج بكت المعلومه يكاد ان يكون صفرايضا.

امانه المعلومه او سريه المعلومه من الامور المهمه فى النقل المتعدد الانتقال. و هذه الخاصيه لم يتم اضافتها الى الـ Multicast ولذلك فى هذا البحث قدمنا مثال عن خدمات اتصال المجموعه كامل السريه اى حمايه المجموعه من دخيل عليها. و نظرا الى بيئه شبكه الـ VPN حيث ينشأ اتصال مؤمن باستخدام الـ IPSec بين جوانب الـ VPN المختلفه التى تحتاج الى الاتصال فيما بينهم. وفى تلك المقترح يستخدم عنصر مركزى للتحكم فى عمليه تأمين الاتصال يسمى VNOC ولكن كل جانب ينشأ اتصال مؤمن بينه و بين جميع الجوانب الاخرى اى ان يوجد مشكله الـScalability

ومن هذه النقطه وجدنا عنصر الشبه بينه و بين مقترحنا الـ HBM فى المركزيه. لذلك دمجنا كلاً من مقترحنا و الـ VPN ليكونوا مقترح واحد يسمى VPRN. دمج كلا من وظائف الـ RP و الـ VNOC ليكون عنصر واحد للتحكم فى عمليه الاتصال بين الجوانب المختلفه عن طريق الـ Overlay معين منشا باستخدام وظائف الـ RP و يكون الروابط فى الـ Overlay مؤمنه باستخدام وظائف الـ VNOC. ومن هنا يمكن ان نقول ان هذا الاندماج لتفادى كلا من مشكله الـ Security فى الـ HBM و كذلك مشكله الـ Scalability فى الـ VPN.

وقد تضمنت الرساله تسعه فصول: يشكل الفصل الاول مقدمه عامه للبحث، مع توضيح او اشاره الى اهميه النقل المتعدد فى حياتنا الحاليه و المستقبليه. و كذلك قدمنا شرح مختصر للنقل المتعدد البديهى IP Multicast. و كذلك وضحنا عده مشاكل التى تواجه النقل المتعدد البديهى من حيث صعوبه انتشاره فى الشبكه العالميه Internet. و فى نفس هذا الفصل ايضا وضحنا من الافكار البديله للحل مع مثال صغيرللمقترحات فى مستوى التطبيق و كذلك الهدف من هذا البحث وبينا محتوى الرساله من الفصول التاليه.

اما باقى الفصول تنقسم الى ثلاثه اجزاء. الجزء الاول خاص بالمقترحات البديله لـ IP Multicast ويمكن ان تكون مكمله له ايضاً و هذا الجزء يحتوى على فصلين الثانى والثالث. الفصل الثانى يحتوى على معاينه Survey للمقترحات البديله و التركيز على الحل البديل فى مستوى التطبيق Multicast Application-Level و ايضا يعرض بعض منهم و كذلك المقارنه بينهم. اما الفصل الثالث فيعرض بشكل تفصيلى مقترحنا HBM (من النوع المركزى فى مستوى التطبيق) . و كذلك ناقشنا بعض الامور المتعلقه بمقترحنا HBM مثل انشاء الـ Overlay و الـ Scalability والمتانه Robustness و كذلك امانه المعلومه Security. و كل هذه الامور ستناقش فى الجزء الثانى.

الجزء الثانى يحتوى على خمس فصول (من الفصل الرابع حتى الفصل الثامن) لتناقش بالتفصيل الامور السابق زكرها فى الفصل الثالث. اذن الفصل الرابع يناقش كيفيه انشاء الـ Overlay وكذلك كيفيه حساب كفائته بالمقارنه بالنقل الاحادى Multi-Unicast. الفصل الخامس يناقش الـ Scalability و فيه اقترحنا عده استراتيجيات و وضحنا عده عوامل لحساب كفائه الاستراتيجيات ومعرفت الافضل منهم. اما عن المتانه الـ Robustness فقد ناقشناها فى الفصلين السادس و السابع. بالنسبه لاضافه روابط اضافيه لتفادى الانقسام الـ Overlay عندما يوجد عطل فى احد الاعضاء، ناقشنا وقدمنا عده خورزميات لاضافه هذه الروابط الاضافيه بين الاعضاء فى الفصل السادس . واما عن تقليل احتماليه فقد المعلومه ناقشنا وقدمنا عده استراتيجيات فى الفصل السابع. اما الفصل الثامن يناقش امانه المعلومه فى الـ Overlay وكذلك ناقشنا كيفيه دمج مقترحنا HBM مع الـ VPN لتقديم مثال لمقترح كامل الامان.

اما الجزء الثالث يحتوى على الفصل التاسع و يناقش عده امور مهمه مع تقديم ماتم انجازه فى هذا البحث و كذلك بعض الامور المستقبليه فى هذا النطاق.

Acronyms

ACK	:	ACKnoledgment
AGCS	:	Alternative Group Communication Service
ALM	:	Application Level Multicast
ALMI	:	Application Level Multicast Infrastructure
AMRoute	:	Adhoc Multicast Routing protocol
API	:	Application Program Interface
ARP	:	Address Resolution Protocol
AS	:	Autonomous System
BGP	:	Border Gateway Protocol
BGMP	:	Border Gateway Multicast Protocol
BOOTP	:	Bootstrap Protocol
CAN	:	Content Addressable Networks
CBT	:	Core Based Tree
CM	:	Core Member
CPE	:	Customer Premises Equipment
DCM	:	Distributed Core Multicast
DCR	:	Distributed Core Router
DiffServ	:	Differentiated Services
DNS	:	Domain Name System
DVMRP	:	Distance Vector Multicast Routing Protocol
ED	:	Edge Device
FTP	:	File Transfer Protocol
GCS	:	Group Communication Services
GCSL	:	Group Communication Services Library
HBH	:	Hop By Hop multicast routing protocol
HBM	:	Host Based Multicast
HMTP	:	Host Multicast Tree Protocol
HTTP	:	Hypride Text Transport Protocol
ICMP	:	Internet Control Message Protocol
IDMaps	:	Internet Distance Map service
IGMP	:	Internet Group Management Protocol
IKE	:	Internet Key Exchange
IP	:	Internet Protocol
IPSec	:	Internet Protocol Security
IVGMP	:	Internet VPN Group Management Protocol
IPv4	:	Internet Protocol version 4
IPv6	:	Internet Protocol version 6
LAN	:	Local Area Network

L2TP	:	Layer 2 Tunneling Protocol
MAODV	:	Multicast operation of the Adhoc On Demand Distance Vector routing protocol
MBone	:	Multicast backBone
MIGP	:	Multicast Interior Gateway Protocol
ML	:	Member List
MLD	:	Multicast Listener Discovery
MOSPF	:	Multicast extension for Open Shortest Path First
MPLS	:	MultiProtocol Label Switching
MSDP	:	Multicast Source Discovery Protocol
MSEC	:	Multicast SECurity
MST	:	Minimum Spanning Tree
MU	:	Metric Update
NAT	:	Network Address Translation
NCap	:	Node Capability
nonCM	:	non Core Member
NTP	:	Network Time Protocol
ODMRP	:	On Demand Multicast Routing Protocol
OM	:	Overlay Multicast
OS	:	Operating System
OSI	:	Open Systems Interconnection
OSPF	:	Open Shortest Path First
PDA	:	Personal Digital Assistant
PIM	:	Protocol Independent Multicast
PIM-DM	:	Protocol Independent Multicast-Dense Mode
PIM-SM	:	Protocol Independent Multicast-Sparse Mode
PIM-SSM	:	Protocol Independent Multicast-Source Specific Mode
PLR	:	Packet Loss Reduction
PPTP	:	Point-to-Point Tunneling Protocol
PPVPN	:	Provider Provisioned Virtual Private Networks
PRM	:	Probabilistic Resilient Multicast
PSN	:	Packet Sequence Number
QoS	:	Quality of Service
RARP	:	Reverse Address Resolution Protocol
REUNITE	:	REcursive UNIcast TreE
RFC	:	Request for Comment
RP	:	Rendez-vous Point
RTCP	:	Real time Transport Control Protocol
RTP	:	Real Time Protocol
RTT	:	Round-Trip Time
RVL	:	Redundant Virtual Link
SA	:	Security Associations
SHDC	:	Scalable adaptive Hierarchical Clustering
SMTP	:	Simple Mail Transfer Protocol
SOAP	:	Simple Object Access Protocol
SP	:	Service Provider
SPT	:	Shortest Path Tree
SSL	:	Secure Socket Layer
TBCP	:	Tree Based Control Protocol

TCP	:	Transmission Control Protocol
TFTP	:	Trivial File Transfer Protocol
TSN	:	Topology Sequence Number
TTL	:	Time To Live
TU	:	Topology Update
UDP	:	User Datagram Protocol
UMTP	:	UDP Multicast Tunneling Protocol
VNOC	:	Virtual Network Operation Center
VPN	:	Virtual Private Network
VPRN	:	Virtual Private Routed Network
WAN	:	Wide Area Network
XCAST	:	eXplicit multiCAST
YOID	:	Your Own Internet Distribution
YTMP	:	YOID Topology Management Protocol

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Contents

In this chapter, we introduce the need for group communication in general, and IP multicast in particular and the deployment problems of multicast routing. These availability problems lead us to quickly introduce alternative group communication services. Finally we introduce the goals and organization of this thesis.

1.1 Introduction

As the Internet grows up, new communication needs arise. First, e-mail and FTP were enough for most people. Then the WWW arrived and people wanted to see graphics, not just plain text. Now, even static graphics are not enough; real-time video and audio are demanded. As communication needs evolve, communication paradigms originally designed to deal with e-mail and FTP need to evolve too. Multicasting is one of them.

Imagine that an event needs to be transmitted to several hosts dispersed over the Internet with audio and video streams. Of course, traffic should be sent as efficiently as possible - the less bandwidth used, the better.
With pre-multicast technology, two communication paradigms are available, both of which are inadequate. The first one is *Unicast*. TELNET, FTP, SMTP and HTTP are unicast-based protocols with *one* source and *one* destination. To send to multiple destinations, different communication paths are needed between the source and each of the destinations. Therefore, a copy of each audio and video stream needs to be sent separately to each receiver. This solution is often called multi-unicast. Clearly, this is not affordable. Even if you are quick enough in copying real-time audio and video streams, both your network and the Internet would collapse when the number of destinations increases.
The second choice is *Broadcast*. The broadcast paradigm saves a lot of bandwidth compared to unicast. If you want to send something to all computers on your LAN, you don't need a separate copy for each. On the contrary, only one copy is sent to the wire, and all computers connected to it receive the copy. This solution is better for our problem but is still insufficient, as we probably need to broadcast to only some of our computers, not all. Even worse, it is almost certain that many hosts interested in your conference will be outside of your LAN and broadcast packets are traditionally not forwarded by routers. Thus, broadcast is good for applications and protocols that don't need to cross LAN limits (such as ARP, BOOTP, DHCP and even routed), but it is not good enough for our problem.

1.2 IP Multicast

Multicast is one solution. After having looked at the problem described before, it is clear we need a solution that :

- allows data to be sent to multiple receivers in an efficient way, avoiding per-receiver copies.
- is not constrained by arbitrary network limits, so it can reach anyone, anywhere on the Internet.
- differentiates between multiple and unrelated transmissions, so that a host may select the ones that are of interest for the user.

The solution that meets all three requirements is **multicast**.

1.2.1 Standard IP Multicast Model

Stephen Deering is responsible for describing the standard multicast model for IP networks [25]:

- IP-style semantics. A source can send multicast packets at any time, with no need to register or to schedule transmission. IP multicast is based on UDP, so packets are delivered using a best-effort policy.
- Open groups. Sources only need to know a multicast address. They do not need to know group membership, and they do not need to be a member of the multicast group to which they are sending. A group can have any number of sources.
- Dynamic groups. Multicast group members can join or leave a multicast group at will. There is no need to register, synchronize, or negotiate with a centralized group management entity.

The standard IP multicast model is an end-system specification and does not discuss requirements on how the network should perform multicast routing. The model also does not specify any mechanisms for providing quality of service (QoS), security, or address allocation.

1.2.2 Multicast Routing Protocols

Multicasting [76] on the Internet is implemented by employing three types of protocols. The first type of protocol is employed by a host to join and leave a multicast group. An example of this type of protocol is the Internet Group Management Protocol (IGMP) [24] with IPv4, and Multicast Listener Discovery (MLD) [23] with IPv6. The second type of protocol is called a Multicast Interior Gateway Protocol (MIGP) and is employed by multicast routers to enable multicast communication within an Autonomous System (AS) which is a network of routers under the control of a single administrative domain. Some examples of MIGPs are Distance Vector Multicast Routing Protocol (DVMRP) [80], Multicast extensions for Open Shortest Path First (MOSPF) [71, 72], Protocol Independent Multicast (PIM) [33, 35] or Core-Based Tree (CBT) [8]. The third type of protocol is employed by border routers, that interconnect two ASes, to allow multicast communication across ASes. An example of this type of protocol is the Border Gateway Multicast Protocol (BGMP) [96].
At the early years of the MBone, the traditional solution was to *set up a tunnel* to a site connected to the MBone. Because of its limitations, it is now banned from new native PIM-SM/MSDP/MGBP deployments [6]. However, many of the existing routers on the Internet do not enable multicast routing (even if present in most operating system, this feature is by default tunned off).
Although a reflector [20] is a host connected to the multicast backbone and which creates point-to-point connections to all the remote hosts that do not enjoy inter-domain multicast routing. If this solution creates hot points within the network, on the other hand it is set up for a limited span of time – the session duration – and for a limited number of groups – those of the session – unlike tunnels. The reflector solution is not satisfying even if reflectors are frequently used.

1.3 Multicast Routing Deployment Problems

Group communication traditionally requires that each node at each site has access to a native multicast routing service. If intra-domain multicast (within a LAN or a site) is widely available, this is different for inter-domain multicast.
IP multicast has been a hot topic of research and development for more than one decade. However, there are still some open issues that make it difficult for IP multicast to be deployed in the global Internet. Today many ISPs are still reluctant to provide a wide-area multicast routing service because of technical or marketing reasons [27] as described in the following subsections.

1.3.1 Router Migration

Multicast deployment at a customer's premise is not a simple issue due to the legacy of existing network infrastructure. A long-term problem for multicast deployment is that it upsets the router migration model that ISPs follow, which is where routers are initially deployed in the backbone, and over time, pushed toward customer access points. As customer acquire higher-speed access lines, backbone routers are migrated toward customers access points to handle the higher speed access lines. Newer routers that support even higher bandwidth are added

to the backbone. In other word, routers are generally installed in the backbone and pushed toward customer access lines as technology moves forward. Multicast upsets this model because older hardware generally does not support multicast. When there are no offered software upgrades, the routers are forced into early retirement. However, removing hardware for upgrades prevents a normally available tax write-off of the depreciation. Furthermore, the natural cycle of cost of migration results in the use of equipment longer than a simple model of its value would predict. Hardware is typically removed when the cost to remove and replace it is less than or equivalent to the cost to maintain or upgrade vital components that would make the hardware support new features.
Router migration has another implication for multicast architecture designs. New routers that are deployed in the backbone are generally less intelligent routers, lacking complicated services such as congestion and admission control. Routers that are simple and unintelligent can handle higher capacity traffic more efficiently. Therefore, complex services, like multicast, would be better deployed in the edge routers, except that replacing such routers upsets the business model. Therefore, both backbone routers and edge routers resist multicast deployment. And despite frequent software updates, multicast will not be fully deployed in networks managed by carriers before a new generation of routers has been installed at all level of the network architecture.

1.3.2 Domain Independence

ISPs using PIM-SM, or other RP/core-based protocols face a number of problems regarding domain independence. Many problems are present when RPs and their associated sources are in distinct domains:

- Traffic sources in other domains potentially require traffic controls, such as rate or congestion control.
- An ISP that relies on a RP located in another domain has very little control over the service that its customers receive via the remote RP.
- ISPs do not want to be the core of a session for which they have no receivers or sources as it is a waste of their resources.
- Advertisement of the address of the RP or core must occur in a scalable fashion with low latency.

1.3.3 Complex Management

Problems common to multicast deployment today include firewalls and a lake of support for Network Address Translation (NAT). An Internet draft on NAT has been issued [38]. The main problem with most firewalls is that multicast (i.e. class D) addresses are not recognized. The only solution to this problem is to tunnel multicast packets through the firewall. This creates a serious security hole in the system where multicast is deployed. Until these problems can be fixed by vendors, the solution supported by vendors to solve firewall incompatibilities is to use static routes to all multicast enabled routers.
ISPs are having a difficult time managing multicast. While intra-domain multicast is relatively easier to deploy, providing inter-domain multicast is complicated. Inter-domain multicast precludes complete control of the network, which makes it difficult to debug problems. This is important with protocols like BGMP or Multicast Source Discovery Protocol (MSDP) [34], which involve contact with other domains.

1.3.4 Justifying the Cost of Multicast

Multicast services are currently significantly more expensive than the unicast service, in terms of deployment, installation at customer premises, and management. Consequently, multicast makes sense today for an ISP or corporate customer only when the bandwidth savings are higher than the deployment and management costs.

The cost of a network service can be defined as the sum of network related costs (router state, processing, and signaling, inter-domain routing scalability) and management costs (ease of deployment and maintenance; in terms of human resources and infrastructure). As described in [27], unicast cost is represented as an increasing straight line because each new receiver adds a new cost (mostly network cost). Multicast however has a high initial cost which is higher than that of unicast. But the cost of adding new receivers should not be as high as in the unicast case. Consequently, there is an incentive for ISPs and content providers for supporting small group sessions with unicast rather than multicast which is only valuable above a (yet to be determined) threshold in terms of group size.

1.3.5 Multicast Forwarding State Scalability

The multicast forwarding state scalability [103] is one of the critical issues that delay the deployment of IP multicast. With traditional Internet protocols, each router is required to maintain a forwarding entry for each multicast session whose distribution tree passes through the router. When there is a very large number of concurrent multicast sessions, the number of the corresponding multicast forwarding entries at routers is also very large. This could consume more router memory and might also result in slower packet forwarding as each packet forwarding involves a routing table lookup. This is the forwarding state scalability issue in providing scalable IP multicast.

1.3.6 Security

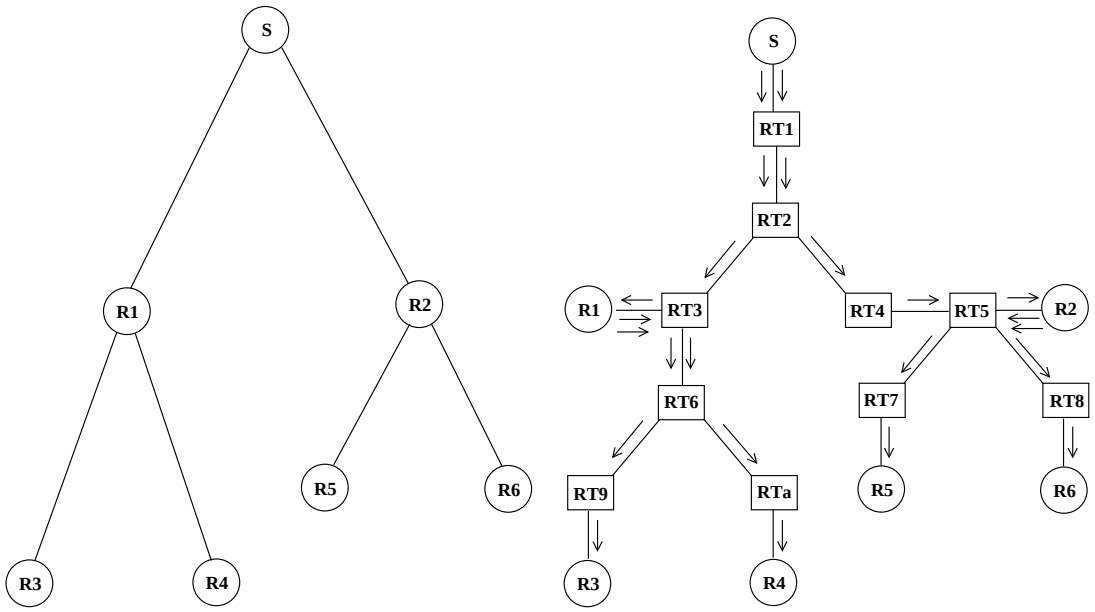
Providing security for multicast-based communication is inherently more complicated than for unicast-based communication because multiple entities participate, most of which will not have trusted relationships with each other. Future multicast security should provide four distinct mechanisms: authentication, authorization, encryption, and data integrity. Authentication is the process of forcing hosts to prove their identities so that they may be authorized to create, send data to, or receive data from a group. Authorization is the process of allowing authenticated hosts to perform specific tasks (i.e. it is the verification of whether a subject is allowed to perform an action on (e.g. access to or use of) an object [85]). Encryption ensures that eavesdroppers can not read data on the network. Data integrity mechanisms ensure that the datagram has not been altered in transit.

The current IP multicast service and architecture do not mandate any authentication. Source authentication and data integrity are possible through the services provided in IPSec, but not receiver authentication. Furthermore, IPSec does not prevent sources from sending; it just allows receivers to drop unauthenticated packets after they are received, IPSec is not widely deployed and is currently under study by the IETF.

1.4 Application-Level Multicast, A.K.A. Alternative Group Communication Service

1.4.1 Main Ideas

The deployment of IP Multicast (i.e. at the network layer) has been limited and sparse due to a variety of technical and non-technical reasons. Therefore some researchers have revisited the issue whether the network layer is necessarily the best layer for implementing multicast functionality and have proposed application-level multicast (i.e. at the application layer) as an alternate technique for multicasting [32]. They enable every host to participate in group communication sessions efficiently, no matter whether it has access to native multicast routing or not. Since data is sent via unicast, flow control, congestion control, and reliable delivery services available for unicast transmission can be exploited, perhaps with minor modifications.



(a) Logical Overlay Multicast Toplogy

(b) Mapping of the Overlay on the physical topology

Figure 1.1. Application-Level Multicast

1.4.2 Short Example

Figure 1.1 shows physical topology on top of which the overlay is built. When considering only the number of packets in all the physical links, we found 33, 23, and 16 copies of packets for multi-unicast, Application-level multicast, and IP multicast respectively. So, with respect to multi-unicast, we found that IP multicast reduce the used resources by 52 % but the application-level by 30 %.

1.5 Goals of the Thesis

As we have seen, the deployment of multicast routing in the Internet is still far behind expectations. Therefore a first motivation for an alternative group communication service

is to bypass the lack of native IP multicast routing. One proposal of an alternative group communication service is overlay Multicast.
In particular, we consider a model in which multicast related features, such as group membership, multicast routing, and packet duplication, are implemented at end systems, assuming only unicast IP services. We call the scheme either End System (i.e End Host) Multicast, Overlay Multicast, Application-level multicast, or Host Based Multicast (HBM).
The main goal is to propose an application-level multicast (HBM) that is simple, easy deployment and no need to routers that support native multicast. Our proposal HBM is a centralized approach and every thing is under control by a single node, or Rendez-vous point (RP). The hot topic of our proposal are studied. In this approach, it has to:

- create not bad the overlay topology,
- improve the scalability,
- impact the robustness in front of node failures and overlay topology modification,
- build a secure group communication service.

In order to achieve these issues, we implemented a group communication services library (GCSL) for our proposal HBM. This library has been implemented in a dedicated C++ under Linux.

1.6 Organization of the Document

The rest of the document is organized into as follows:
Chapter 2 introduces and discusses several proposals for building an Alternative Group Communication Service. In this chapter, We classify the proposals into several categories: based on a reflector approach, relying on permanent tunneling, creating an automatic overlay topology, or relying on a specific routing service.
Chapter 3 describes our Host Based Multicast (HBM) proposal which is a centralized solution, where everything, including group membership management and overlay topology creation, is under the control of a single Rendez-vous Point (RP).
In **chapter 4** we describe how to create and adapt the overlay topology, taking into account many metrics such as RTT, losses, node stability, and the access network features.
In **chapter 5**, we discuss the scalability of HBM. This chapter explains how the scalability can be largely improved, with a few simple protocol parameter adjustments: the number of records in a metric update message, the metric update generation period, and the topology update period.
In **chapter 6** we describe a strategy that adds Redundant Virtual Link (RVL) in order to avoid topology partition after a node failure.
Chapter 7 focuses on the problem of packet losses generated by transient incoherences in the overlay topology. In this chapter, we only focus on incoherences that are the result of a topology update (e.g. to improving the overlay multicast topology).
In **chapter 8** we investigate the use of HBM to build a fully secure and efficient group communication service between several sites based on Virtual Private Network (VPN). We showed that HBM and VPN naturally fit with one-another and lead to the concept of Virtual Private Routing Network (VPRN).
Finally, we discuss some key points and conclude this thesis in **chapter 9**.

Part I

Alternative Group Communication Services

Chapter 2

A Survey of Proposals for an Alternative Group Communication Service

Contents

In this chapter, we introduce a survey of proposals that build an alternative group communication service. We first describe the proposals, and then we compare them.

2.1 Introduction: Motivations for an Alternative Group Communication Service (AGCS)

A Group Communication Service refers to the ability to send information to several receivers at the same time, using either a one-to-many or many-to-many model. The any-source and source specific multicast routing approaches provide such a service. Yet other solutions are possible and this chapter proposes a survey of such alternatives.
Although this survey aims to give a complete overview of AGCS techniques, we do not claim to be exhaustive. Besides we only consider the routing service (i.e. as a replacement of, or complement to, IP-multicast) and ignore any upper-level service like reliability or congestion control. If some of the solutions we introduce largely impact these upper-level services, this is a by-product that will not be discussed in this chapter. Likewise, we do not cover DiffServ multicasting or, more generally, QoS-based multicast routing [94].
An AGCS can be used as a way to *bypass the multicast routing deployment problems.* Indeed, group communication traditionally requires that each node at each site has access to a native multicast routing service. If intra-domain multicast (within a LAN or a site) is widely available, this is different for inter-domain multicast. Today many ISPs are still reluctant to provide a wide-area multicast routing service [27].
But other Motivations exist. For instance an AGCS can be used to go beyond the limitations of traditional multicast routing. An AGCS can offer a bridging service between several multicast capable areas running different multicast routing protocols, for instance between IPv4 and IPv6 multicast islands.
An AGCS can also be used along with PIM-SSM [50]. Since only the source S is allowed to send traffic to an (S, G) channel, G being the group addresses, no multicast back-channel is available for a receiver to provide feedback to the group. If the feedback rate is sufficiently low (e.g. with RTCP), this feedback can be unicast to the source and echoed back onto the channel. If not, such an approach quickly results in source implosion and this is where an AGCS can be of some help.
Finally an AGCS can be used in working environments where traditional multicast routing is completely inappropriate. This is the case of ad-hoc networks where there is no fixed infrastructure. Multicast routing, designed for a fixed hierarchical routing infrastructure with well identified multicast routers, is completely defeated. This is also the case when there is a very high number of small dynamic groups. The signaling load required by traditional multicast routing for each group prevents the whole system to scale in terms of number of concurrent groups.

2.2 Performance Metrics

Several performance metrics have been defined to characterize AGCS performance and impacts on the network. Some of them focus on the *data path:*

Stress: [52] defines the stress of a physical link as the number of identical packets it carries. The optimal value, achieved with native multicast routing, is of course 1.

Resource Usage: [52] defines this metric as the sum of the $delay * stress$ over all the links that participate in data transmissions. This metric gives an idea of network resources used by the transmission process, assuming that links with high delays are more costly.

Stretch: also called "Relative Delay Penalty" in [52], the stretch metric between a source and a member is the ratio of the delay between them along the overlay distribution topology, to the delay of the direct unicast path.

Another set of metrics focuses on *end-host performance:*

Losses after Failures: This metric counts the average number of packet losses after an ungraceful failure of a single node [26, 86]. It highlights robustness in the occurrence of unpredicted events.

Time to First Packet: [26] defines the time required for a new member to start receiving a data flow when joining an on-going session.

Finally some metrics focus on the *control part:*

Control Overhead: maintaining the AGCS topology has a cost, in terms of control information exchanged (number of messages processed and bandwidth) [31].

The large diversity of performance metrics shows there is no single answer to the question: "what is the best solution". Some proposals can deliberately favor some of these metrics at the expense of others (e.g. the multi-unicast approach used by reflectors (section 2.3.1) offers a high robustness to member failures (other than the reflector itself) at the cost of a high link stress near the reflector).

2.3 A Taxonomy of AGCS Proposals

In this section we introduce several proposals that build an alternative group communication service and briefly discuss their advantages and limitations.

2.3.1 Unicast/Multicast Reflector and Punctual Tunneling Proposals

2.3.1.1 Principles

In this category we find solutions whereby a host having only access to unicast routing contacts a reflector, which is a user-level gateway between a multicast enabled network (e.g. the MBONE) and the set of unicast hosts. Each multicast packet coming from the multicast network (respectively from a unicast host) is forwarded to each unicast host (respectively to the multicast group and the other unicast hosts). These solutions are often called tunneling approaches too since they create tunnels between the reflector and the end-hosts. Yet they are completely different from the *permanent* tunneling approaches of section 2.3.2.
The first key aspect is its *application level feature*. The communication between a host and the reflector can be more or less elaborated: multicast packets can be captured by a BPF [53] packet filtering tool and encapsulated in unicast datagrams. A simpler solution consists in opening a UDP socket and forwarding only the payload, without the initial packet headers. In that case the source address and port are lost but upper protocols (e.g. RTCP) may recover the source identity.

Secondly this service is usually *set up for a limited time and for a limited number of groups* (usually there is one reflector per group).
The UMTP [39] and Mtunnel [75] proposals fall in this category.

2.3.1.2 Discussion

This approach is clearly not the most efficient one since it creates hot spots in the network, near the reflector. Yet it is easily set up and the reflector has a full control on the service, its duration, the multicast groups forwarded and the set of unicast authorized hosts. Therefore its global impact on the network on a longer period is limited.
A straightforward extension that largely improves scalability consists in having a topology of reflectors, each of them controlling a multicast capable area. All the receivers of a domain are thus hidden behind their local reflector.

2.3.2 Permanent Tunneling Proposals

2.3.2.1 Principles

Permanent Tunneling proposals differ from the reflector proposals from several points of view. First of all, tunneling is performed at routing level and uses IP encapsulation. Its creation requires privileges and is usually not set up by a end-host.
Secondly, if a reflector answers a punctual need within a well identified group of people, tunneling solutions offer permanent connectivity for a whole site.
Thirdly, tunnels are fully integrated in the multicast routing protocols and offer connectivity to all possible multicast groups.
The Mrouted DVMRP implementation [37] is undoubtedly the most popular tunneling solution and has long been used in the MBONE. AMT [97] is midway between the reflector and permanent tunneling categories. It manages the multicast traffic exchange for any groups between isolated multicast-enabled sites, yet is does not include a routing protocol, unlike DVMRP/Mrouted.

2.3.2.2 Discussion

This class has long been the only way to connect isolated multicast islands. Because of performance problems that tunnels creates (potential for a high physical link stress and loops), it is now banned from modern multicast routing protocols. Yet multicast tunneling is still used in some situations, for instance when crossing IPSec VPN tunnels that do not support multicast packets [4].

2.3.3 Automatic Overlay Multicast Proposals, A.K.A. Application-Level Multicast

2.3.3.1 Principles

This broad class of proposals shifts the multicast support from core routers to end systems. End Systems now implement all the group communication functionalities, including membership management, packet replication and distribution. For instance, in Narada [52] group members communicate via an overlay structure built on top of unicast paths between various pairs of hosts as shown in the Figure 2.1. The physical topology is abstracted as a complete virtual graph on which Narada constructs a spanning tree using a Reverse Path Forwarding algorithm. Since the group is dynamic, a mechanism is defined to add or drop links, repair partitioned topologies , and incrementally improve the virtual topology.

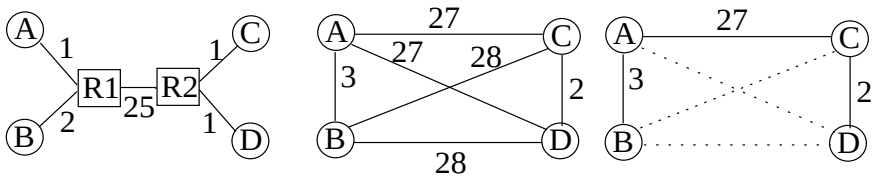


Figure 2.1. The physical and overlay topologies.

The Overlay Multicast (OM) approach differs in many respects from traditional multicast routing:

- A forwarding node in the overlay topology can be either *a end-host* (i.e. running the application), *a dedicated server* within the site, or *a border router*. On the opposite traditional multicast trees only include core routers.
- With an overlay topology, *the underlying physical topology is completely hidden*. A directed virtual graph is created between all the nodes. Undirected graphs can also be used if the possibility of having asymmetric routes is overlooked. Such graphs are built and optimized according to some form of metric measurements taken between some or all nodes.
- In traditional multicast, the membership knowledge is distributed in the multicast routers. With an OM *group members are known* either by a Rendez-vous Point [86], by the source, by everybody, or is distributed among members [52, 69].
- The *overlay topology is potentially under complete control*. For instance [86] takes advantage of the additional knowledge centralized at the RP (the node/link specificities, and their stability) during the topology creation process.

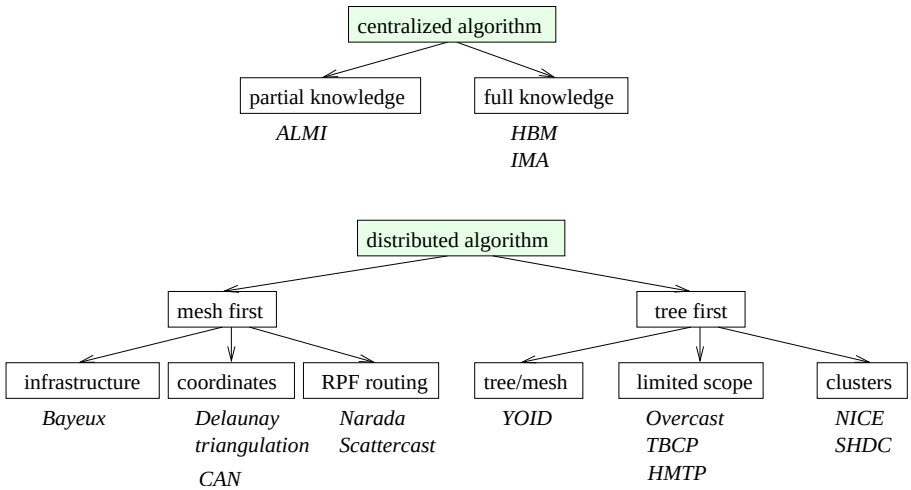


Figure 2.2. A taxonomy of overlay multicast proposals.

Figure 2.2 classifies the proposals according to the centralized or distributed topology building algorithm. The centralized approaches are further classified according to the full (HBM [86]) or partial (ALMI [77]) membership knowledge.

The distributed approaches further differ in the way they create the overlay topology: some of them first create the tree topology while others first create a mesh topology.
The "mesh first" approaches are Narada [52], Scattercast [19], the proposals that assign an arbitrary coordinate to each member and then performs Delaunary triangulation [64, 65], Content-Addressable Networks (CAN) [83] and Bayeux [81].
The "tree first" approaches include YOID [40, 41, 42, 43], Overcast [55], TBCP [69, 95], HMTP [104], NICE [1, 11, 62], SHDC [70], ZIGZAG [98], and Tiers [12]. Some of them (TBCP, HMTP) rely on a recursive algorithm to build the tree: a newcomer first contacts the tree root, chooses the best node among the root's children, and repeats this top-down process until it finds an appropriate parent. The clustering solutions (NICE, SHDC, ZIGZAG) create a hierarchy of clusters, i.e. sets of nodes "close" to each other. Newcomers recursively cross this hierarchy to find the appropriate cluster.

2.3.3.2 Discussion

These proposals form undoubtedly a rich family that reflects the large diversity of objectives: high performance thanks to an optimized communication topology, adaptability and per-host profiling of the topology to take into account their features, robustness in the event of member departures and failures, and high scalability. Having a good level of congruence between the physical topology and the overlay is rather challenging and is often the key for good performances. Yet, being "end-to-end", overall paths along the overlay can get rather long in terms of delays, and data can be replicated several times over some physical links. The bottom line here is that the right compromise should be achieved between growing an overlay multicast tree in length or in width, to suit both application requirements and network conditions. Anyway, because of its very own nature, an overlay will never match the efficiency of native IP multicast.
A major advantage is that most proposals do not require any special support from network routers and can therefore be deployed universally. As a result, they can be made available as libraries or built in application code, which reduces the need for standardization.

2.3.4 Proposals Based on Gossiping for Peer-to-Peer Communications

2.3.4.1 Principles

Gossiping is widely used for state and data distribution in distributed systems. Within the context of overlay networks, a gossiping technique like Scribe [18, 88] can be used for *state distribution*. Each group member periodically sends an announcement message containing its list of neighbors to these neighbors themselves. Hence, each node increases the "group membership knowledge horizon" by one hop. Furthermore, each node adds to the list any nodes it has heard of to propagate the knowledge about nodes not directly connected. Therefore a node gains complete knowledge of the group just by connecting to any node already in the group. The identity of such a node can be learned from a Rendez-vous Point (RP) that only maintains a list of a few members. Once a node has learn the identity of other nodes, it can periodically measure its distance to a randomly chosen set of these peers and replace its farthest neighbors with newly found closer ones to improve the overlay infrastructure efficiency.
Alternatively, gossiping can also be used to create a highly robust *data distribution service*. The difficulty is to estimate when to remove any given data item from the gossiping process (i.e. it is difficult to estimate when all group members have seen the corresponding data item,

especially within a dynamic group). This scheme is therefore usually limited to small data transfers or to static group environments.

2.3.4.2 Discussion

Gossiping is a very robust state distribution mechanism. Indeed, the loss of any single node does not result in any knowledge loss. Furthermore, because a RP is only needed to "bootstrap" new nodes and because the RP only has a limited membership knowledge, it does not represent a potential "hot spot".
For very large groups, however, the periodic announcements result in a large overhead. The classical solution is to reduce the announcement frequency which, in turn, reduces the system responsiveness. Tuning such a system is therefore non-trivial.
Finally, nodes of a large group only have a reasonably accurate view of their vicinity. This is due to the time required to advertise new members (an announcement period per hop) and to remove those who left (detected only after several silent announcement periods). The convergence of application-level multicast techniques based on gossiping can therefore be rather slow.

2.3.5 Proposals Based on a Specific Group Communication Routing Service

2.3.5.1 Principles

Several proposals rely on a dedicated new group communication routing service within routers. Therefore they cannot be deployed on demand by the end users. Yet, they may be one day standardized and deployed (e.g. the XCAST community tries to create a new IETF working group). Some proposals address several limitations of traditional multicast routing protocols:

- the low scalability in terms of the number of concurrent groups, and
- the need for a stable networking infrastructure.

Two proposals try to solve the *scalability issue*. For instance XCAST [5, 16] adds an explicit list of destinations in each packet using either a new XCAST header (IPv4) or a new routing extension header (IPv6). Each router along the way parses the IP header and in case of branching, creates and forwards a new packet with the appropriate sub-set of destinations reachable from each interface. When a single destination is left, the XCAST packet is turned into a normal unicast packet. XCAST derives its high scalability from the fact that no state information is kept within the backbone.
The Distributed Core Multicast (DCM) [14, 15] proposal has similar goals but follows a different method. DCM uses several Distributed Core Routers (DCRs), located at the edge of the backbone and synchronized with a dedicated membership distribution protocol. Each site contains one or more DCRs that forward traffic to/from other sites. The scalability asset of DCM derives from the fact that group state information is kept in the DCR routers rather than being disseminated across the backbone routers.
The second issue is typical in *Mobile Ad-hoc Networks* characterized by rapidly-changing, multi-hop topologies composed of several wireless links, with no fixed infrastructure. Traditional multicast routing protocols cannot be deployed then, as they rely on well identified multicast routers. Therefore a number of proposals have been proposed:

- AMRoute [66], a protocol that first creates an overlay mesh and then a shared multicast tree on top of it. It shares many similarities with the protocols of section 2.3.3.

- ODMRP [7, 61] is a protocol where a source creates on demand and for a limited lapse of time a mesh of hosts in which data is flooded.
- MAODV [89, 90], a multicast routing protocol building a shared tree. This process is purely on-demand and follows a route request/route reply discovery cycle, where the request is broadcast to neighborhood and forwarded until it reaches the destination or a node having a route to the destination.

These protocols largely differ in the way the distribution topology is created and maintained, some of them leading to mesh-based distribution, others to tree-based distribution. In each case the topology is regularly updated to take into account the possible topology changes.
Finally the REUNITE [93] and HBH [22] proposals follow a recursive unicast approach to solve the multicast deployment issue. The idea is to have some REUNITE/HBH capable routers that act as branching nodes and create copies with modified unicast destination address between two hops. It is similar to XCAST except that packets do not carry the list of destinations. Branching nodes thus need to keep some state for each group.

2.3.5.2 Discussion

The efficiency of many of these proposals (e.g. XCAST, REUNITE, HBH), not surprisingly, depends on the number and location of routers offering the service, even if a partial deployment is still possible. If routers at the natural branching points (usually within the backbone) support it, then efficiency can be high. If only the routers close to end-nodes support it, then the link stress is significantly higher.
Concerning the proposals dedicated to ad-hoc networks, performance largely differs and depends on the host mobility (how many of them are mobiles and how fast they are moving). For instance [63] shows that in highly mobile environments, mesh-based protocols outperform tree-based protocols, essentially because of the presence of alternate routes.

2.4 Description of some Application-Level Multicast Proposals

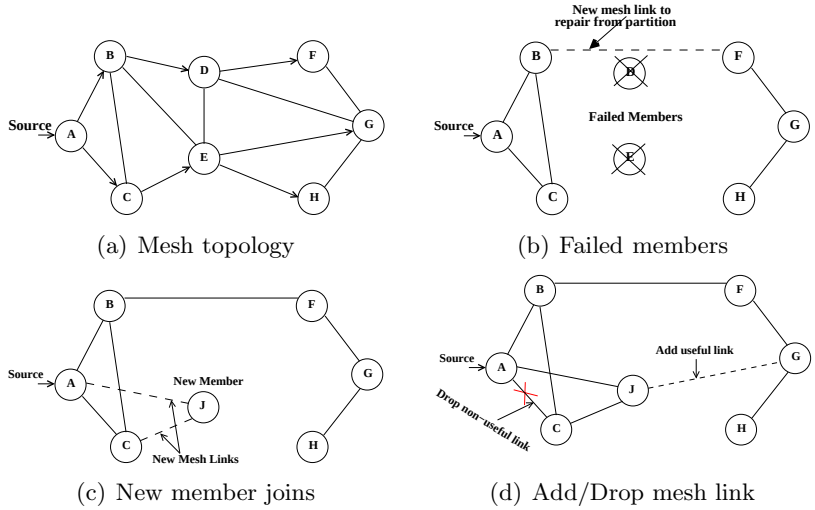
In this section, which is essentially extracted from [9], we describe with more details several Application-level multicast proposals, that we quickly introduced in section 2.3.3: ALMI, NARADA, CAN, HMTP, NICE, Bayeux, and YOID. Our proposal, HBM, that also belongs to this category, will be described in chapter 3.

2.4.1 ALMI

Application-Level Multicast Infrastructure (ALMI) [77, 92] consists of a session controller and multiple session members. A session controller is a program instance, located at a place that is is easily accessible by all members (e.g. within a dedicated server). Session members are organized into a shared-tree using bidirectional links. Session data is disseminated along this tree, while control messages are unicast between each member and the controller. The controller calculates a minimum spanning tree based on the measurement updates received from all members. To collect measurements the controller essentially instructs each member to monitor a set of other members.
Discussion: ALMI is very close to our own proposal, HBM, proposed simultaneously. The controller (ALMI) has be informed by the metrics about some member pairs while RP (HBM) has be informed by the metrics about all member pairs. Then RP can create a better multicast tree than that of the controller (ALMI).

2.4.2 NARADA

The Narada protocol [51, 52] was one of the first application layer multicast protocols that demonstrated the feasibility of implementing multicast functionality at the application-layer. Narada defines a special designated host, called the Rendezvous Point (RP), that is used to boot-strap the join procedure of a new member. In fact, most application layer multicast protocols use an entity equivalent to the RP in Narada to initiate the join mechanism.
Mesh construction: When a new member wants to join the multicast group, it first obtains all group members that are already joined to the mesh. This information can typically be obtained from the RP, which maintains state about all members joined to the multicast group. The new member then randomly selects a subset of these members and attempts to join the mesh as neighbors of these members. The join procedure succeeds when at least one of these members accepts the new member as its mesh neighbor.
After joining the mesh, the new member starts exchanging periodic refresh messages with its mesh-neighbors. Whenever a new member joins or an existing member leaves the group (and the mesh), this group change information is propagated through the mesh to all the other members. Thus, each member in the group keeps state about all other members that are part of the group. This information is also periodically refreshed. Distribution of such state information about each member to all other members leads to relatively high control overhead ($\bigcirc$ (N^2) aggregate control overhead, where N is the group size). Therefore, the Narada protocol is effective only when the multicast group size is small. However, this was an explicit design choice for the Narada protocol, where the authors traded off high control overheads for greater robustness in recovery from member failures in some specific cases. This is described in the following example in the Figure 2.3.



(a) Mesh topology (b) Failed members

(c) New member joins (d) Add/Drop mesh link

Figure 2.3. An example of Narada

Figure 2.3 describes control and data paths in Narada. Neighbors on the control path (mesh) are connected by edges. In Figure 2.3-a, the thicker edges (marked with arrows) indicate the multicast data path when A is the source of data. These edges are also part of the mesh. In Figure 2.3-b, two members, D and E, leave the group which leads to a mesh partition. The remaining members repair this partition. In Figure 2.3-c, a new member joins the mesh and sets up mesh-neighbor links with two randomly chosen members. Periodically members evaluate the utility of different links on the mesh. In Figure 2.3-d, anon-useful link is deleted from the mesh, and a potentially useful link is added to it.

When members D and E simultaneously fail, the mesh partitions into two parts. As a consequence, members A, B and C stop receiving state refresh messages from members F, G and H, and vice versa. Each member then probabilistically probes the members from which it has stopped receiving refresh messages to establish new links and repair the partition (Figure 2.3-b). Note that the recovery does not require the intervention of the RP and, therefore, works even when the RP fails.

Data Delivery Path: The members of the group run a routing protocol to compute unicast paths between all pair of members on the mesh. The multicast data delivery path with any specific member as the source can then be computed using the well-known Reverse Path Forwarding check employed by IP multicast protocols (e.g. DVMRP [80]). The specific links on the data path from source A is highlighted with thicker directed edges.

Mesh Refinement: The data delivery paths in Narada are spanning trees of the mesh. Therefore, the quality of the data delivery path (i.e. the stress and stretch properties for example) depends on the quality of links that are part of the mesh. When new members join, or when the mesh recovers from partitions, a random set of edges are added to the mesh. Thus, periodic refinements are made to mesh edges to improve the quality of data delivery paths.

In Figure 2.3-d, we show such refinements. Adding the edge <J, G> is considered useful because a large number of shorter unicast paths can be created on the mesh using this new edge (for example between member sets {A, C, J} and {G, H}, and between member sets {C, J} and {F}). The edge <A, C> is removed from the mesh since was being used to create a single shortest path (that between members A and C). These decisions to add or drop edges from the mesh are made locally by the two end-points of the edge, through simple heuristics that compute a specific "utility" for the edge.

2.4.3 CAN-Multicast

Content-Addressable Network (CAN) [82] is an application-level infrastructure where a set of end-hosts implement a distributed hash table on an Internet-wide scale. The constituent members of the CAN form a virtual d-dimensional Cartesian coordinate space, and each member "owns" a portion of this space. For example, Figure 2.4 shows a 2-dimensional coordinate space partitioned into zones by 34 CAN members of which 6 (members A , F) are marked in their respective zones. In [83], the authors propose an application layer multicast scheme based on the CAN architecture.

Control and data topologies: In the control topology two members peer with each other if their corresponding regions in the d-dimensional space about each other. For example, Figure 2.4-a, member A has 5 neighbors on the control topology, B, C, D, E and F. The data topology is implicitly defined by performing directed flooding on the control topology (e.g. Figure 2.4-b). In [83], one such data topology is defined by the following forwarding rule: *The source forwards a data packet to all its control topology neighbors. Consider a member, h, that receives a data packet from another member, p. Members h and p are neighbors on the control topology. Member h will forward this packet to a neighbor, n, on the control topology if and only if (a) $n \neq p$ and (b) the packet has not traversed half of the coordinate space toward the dimension along with h and n about. The latter condition ensures that packets do not loop around the CAN. Each member also maintains a packet cache to identify and discard any duplicate packets.*

Join Procedure: When a new member, Z wants to join the CAN, it queries the RP to find at least one existing member, X, that is already joined to the CAN. Z picks a random point in the coordinate space (say a point which is owned by member Y). The goal of the joining

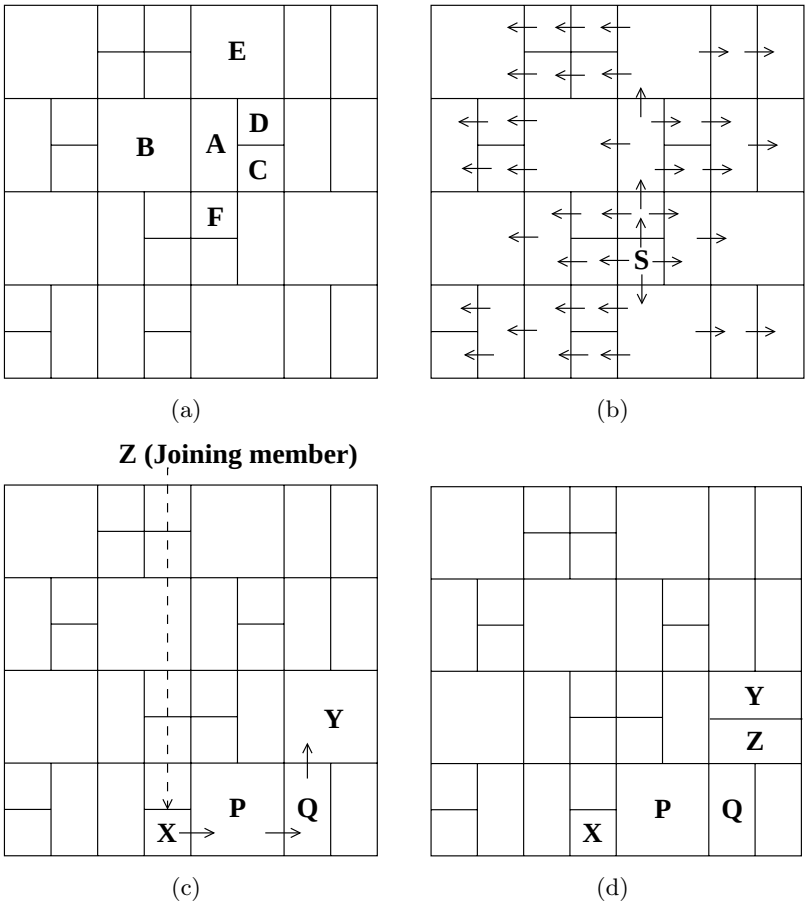


Figure 2.4. An example of Content-Addressable Network (CAN)

member is to find the member Y which owns this randomly chosen point. This is done by routing through the CAN, as shown in Figure 2.4-c. The protocol then splits the zone owned by Y into two, and the ownership of one of the halves is transferred to Z (Figure 2.4-d).
The assignment procedure of zones of the coordinate space to members of the CAN, as described, ignores the relative distances between the members in constructing the overlay. As a consequence, neighbors on the CAN may be far apart and thus, the multicast overlay paths can have high stretch. To remedy this situation, the authors in [83] suggest the use of a "distributed binning" scheme by which members that are close to each other are assigned nearby zones in the coordinate space.

2.4.4 HMTP

Host Multicast Tree Protocol (HMTP) [104] is another application layer multicast protocol that uses the tree-first approach and has some similarities with the Yoid protocol. **Tree Construction:** Like in Yoid, members in HMTP are responsible for finding parents on the shared tree. A joining member, h, finds its parent using the following heuristic: It first discovers the root of the shared tree by querying the RP. Starting from the root, at each level of the tree h tries to find a member, x, close to itself. If the number of children is less than its degree bound, h joins as a child of x. Or else it proceeds to the next level and tries to find a potential parent among the children of x. This is shown using an example in Figure 2.5.
Figure 2.5 describes tree join procedure in HMTP. The new member, N, discovers the root,

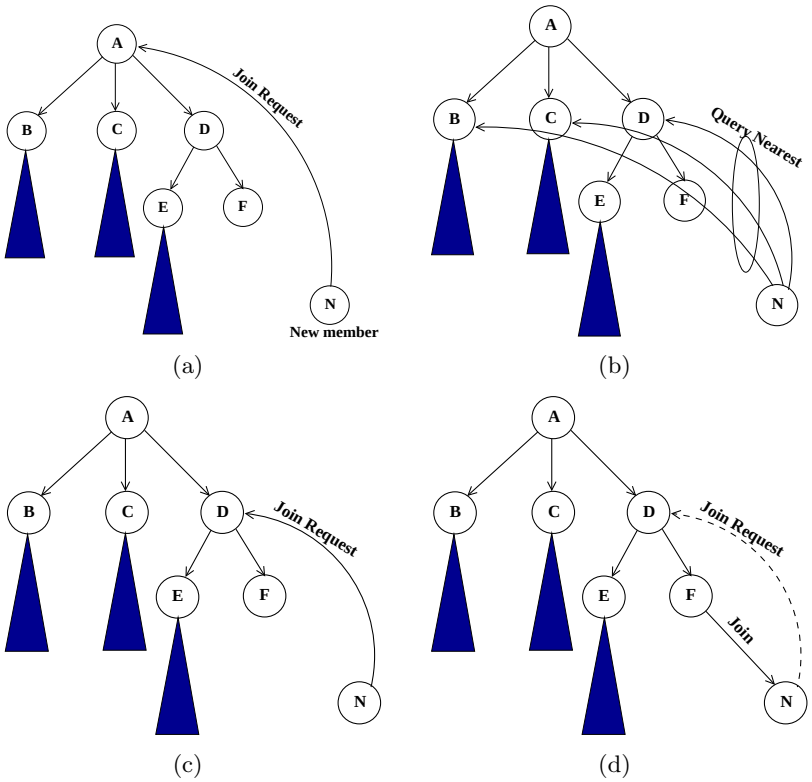


Figure 2.5. An example of HMTP

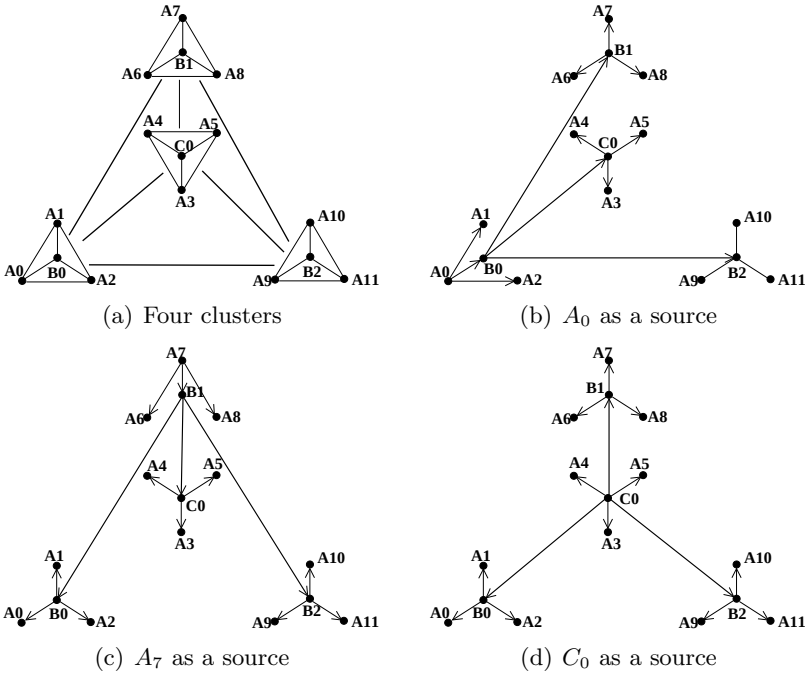
A, by querying the RP. For this example we assume that members A and D have a maximum degree of 3 (and therefore cannot support any other member as children). N recursively finds a nearby potential parent by querying down the tree. The new member, N probes the root,A. Since all its degree is filled, it then probes to find, D, the child of A which is closest to itself. However, since D also has no available degree, N proceeds to the next level and finds, F, the nearest child of D. F has available degree and so N joins as a child of F.
Members in HMTP maintain information about all members on its path to the root. Periodically, each member tries to find a better (i.e. closer) parent on the tree, by re-initiating the join process from some random member on its root path. Knowing the entire root path allows members to detect loops. HMTP employs a loop detection and resolution mechanism, instead of loop avoidance.
Unlike Yoid, HMTP does not explicitly create a mesh. However, each member periodically discovers and caches information about a few other members that are part of the tree. In the specific case when the RP is unavailable, the knowledge of such members is used to recover the tree from partitions.

2.4.5 NICE

The NICE protocol [11] arranges the set of members into a hierarchical control topology. As new members join and existing members leave the group, the basic operation of the protocol is to create and maintain the hierarchy. The hierarchy implicitly defines the multicast overlay data paths and is crucial for scalability of this protocol to large groups. The members at the bottom of the hierarchy maintain (soft) state about a constant number of other members, while the members at the top maintain such state for about $\bigcirc(\log N)$ other members.

The NICE hierarchy is created by assigning members to different levels (or layers). Layers are numbered sequentially with the lowest layer of the hierarchy being layer zero (denoted by L0). Members in each layer are partitioned into a set of clusters. Each cluster is of size between K and 3k-1, where k is a constant, and consists of a set of members that are close to each other. Further, each cluster has a cluster leader. The protocol distributively chooses the (graph-theoretic) center of the cluster to be its leader, i.e. the cluster leader has the minimum maximum distance to all other members in the cluster. This choice of the cluster leader is important in guaranteeing that a new joining member is quickly able to find its appropriate position in the hierarchy using a very small number of queries to other members. The members are assigned to the different layers as follows: All members are part of the lowest layer, L0. A distributed clustering protocol at each layer partitions these members into a set of clusters with the specified size bounds. The protocol also chooses the member which is the graph theoretic center of the cluster, to be the leader of the cluster. The cluster leaders of all the clusters in layer L_i join layer L_{i-1}.

Control and Data Topologies: The member hierarchy is used to define both the control and data overlay topologies. In the control topology, all members of each cluster peer with each other and exchange periodic refreshes between them. The data topology is defined by the following forwarding rule on the control topology: *The source member sends a data packet to all its peers on the control topology. Consider an intermediate member, h that belongs to layers $L_0 \ldots L_j$ that receives the data packet from an other member, say p. Then p and h belong to the same cluster in some layer, say L_i. Member h will forward the data packet to all other members of cluster $C_k, k \neq i$ (where C_k corresponds to its cluster in layer L_k) if and only if h is the cluster leader of C_k.*



(a) Four clusters (b) A_0 as a source

(c) A_7 as a source (d) C_0 as a source

Figure 2.6. An example of Nice

Figure 2.6 shows an example, for different sources and Nice hierarchy and control and data topologies for a two-layer hierarchy. All A_i hosts are members of only L_0 clusters. All B_i hosts are members of both layers L_0 and L_1. The only C host is the leader of theL_1 cluster comprising of itself and all the B hosts.

Join Procedure: A new member joins a L_0 cluster that is closest to itself with respect to the distance metric. Locating this L_0 cluster is approximated by a sequence of refinement steps, where the joining member starts with the topmost layer and sequentially probes one cluster in each layer to find the "closest" member in that layer.

2.4.6 Scribe

Scribe [18, 88] is a large-scale event notification system that uses application layer multicast to disseminate data on topic-based publish-subscribe groups. Scribe is built on top of Pastry [87] which is a peer-to-peer object location and routing substrate overlaid on the Internet. Each member in Pastry is assigned a random node identifier, which may be generated by computing the cryptographic hash of the member's public key. Pastry organizes the members into an overlay in which messages can be routed from a member to any other member by knowing the node identifier of the latter. This organization is shown in Figure 2.7. The members are represented by rectangular boxes and their node identifiers are marked inside the box. The routing table has 4 rows. Each member in the routing table share a common prefix with the member 2313. Row 2 in the routing table, for example, has members (2021, 2130 and 2201) that share the first digit (2) of their node identifiers. Additionally each of these members in the same row have a different digit at the second position (i.e. 0, 1 and 2 respectively). The fourth member in this row should have the same 1-digit prefix and the digit 3 in the second position. It is the member 2313 itself (and so it is not added to the routing table).
The node identifiers are thought of as a sequence of digits with base 2^b, where b is a small constant. In the example in Figure 2.7, b = 2. In this section we refer to members by their node identifiers.

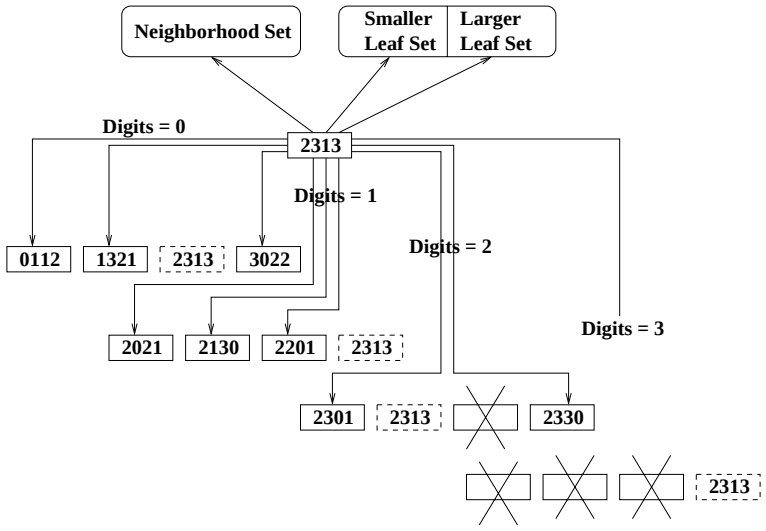


Figure 2.7. Neighborhood of a Pastry member, with identifier 2313. All numbers are in Base-4.

Each member has a routing table, a neighborhood set and a leaf set. The routing table for a member, h, contains information about a set of members in the overlay with which the member, h, shares a common prefix. All members in row i of the routing table of h have the same i-1 digits in their node identifier prefix (as shown in Figure xx6). There are $2^b - 1$ such entries in each row. If N is the size of the node identifier space, the total number of rows in

the routing table is $log_2^b N$, which corresponds to the number of digits in the node identifier. If this member is not aware of any member with a matching prefix of some given size, the corresponding entry in the routing table is empty. The neighborhood set of the member, h, has members that are close to h based on the distance metric. The leaf set of h contains members for which the node identifiers are numerically close to the node identifier of h. It is partitioned into two equal-sized sets with one corresponding to numerically smaller node identifiers and the other corresponding to numerically larger ones.

Control and Data Topologies: The Scribe application layer multicast protocol uses Pastry as the underlying routing substrate to provide multicast services. Therefore the control topology in Scribe is the same as the control topology in Pastry. The neighbors of any member on the control path include all its routing table, neighborhood set and leaf set entries.

Unicast paths to specific destination identifiers in Pastry are defined by the following rule:

> A message with destination identifier, y, is routed toward a member which has y as its node identifier. If no such member exists on the overlay, the message is routed to a member, z, whose node identifier is numerically closest to y. This routing is performed as follows – each member forwards the message to a member in its routing table that shares a longer common prefix with the destination identifier than its own identifier. When no such member can be found in the routing table, the message is forwarded to a member in the leaf set that is numerically closer to the destination identifier than its own identifier.

A multicast group in Scribe typically consists of a subset of the members that have already joined the Pastry overlay. Each multicast group in Scribe has its own identifier (and is called the topic identifier in [18]). The member whose node identifier is numerically closest to the multicast group identifier becomes the RP for that group. The data topology for a multicast group in Scribe is the union of the Pastry unicast paths from the different group members to the RP. The state for this data path is set up during the join procedure as described next.

Join Procedure: In Scribe, when a member on the overlay joins a multicast group, it routes a join message using the multicast group identifier as the destination identifier. This message gets routed by the Pastry substrate toward the RP. All members on this unicast path that are not already a part of the multicast data delivery tree for the group add themselves to the tree. A member needs to be joined to the Pastry group to be able to join a Scribe multicast group. The join procedure for Pastry is discussed in detail in [87].

2.4.7 Bayeux

Bayeux [81] is an application layer multicast scheme. It is built on top of a peer-to-peer object location system called Tapestry [105]. The Tapestry overlay structure is similar to Pastry [87]. Although their underlying routing substrates are similar, Bayeux and Scribe differ in the way the multicast data delivery paths are created. Both Tapestry and Pastry are complex because they depends on the identifier or member's public Key respectively. Tapestry uses local routing maps at each node, called neighbor maps, to incrementally route overlay messages to destination identifier digit by digit (e.g. ***8$\Longrightarrow$**98$\Longrightarrow$*598$\Longrightarrow$4598).

2.4.8 YOID

Yoid, along with Narada, is one of the first application layer multicast protocols. Since Yoid directly creates the data delivery tree, it has a direct control over various aspects of the

tree structure, e.g. the out-degree at members, the choice of tree neighbors, etc. This is in contrast to the mesh-first approach which has an indirect control on the tree structure and therefore, the quality of the data delivery tree depends on the quality of the mesh. Yoid has essentially on historical interest. Yet since this proposal has never been published anywhere and since no Research Report details it, we do not detail it.

2.5 Comparative Study of Overlay Multicast protocols

Different application layer multicast protocols have various properties that make them suitable for different applications. In this section, we compare them and comment on their suitability to specific applications.

Table 2.1. A Comparison of different Automatic Overlay Multicast Schemes

Scheme	Type	Tree-type	Max. path length	Max. Tree degree
HBM	centralized	Shared	Unbounded	bounded
ALMI	centralized	Shared	Unbounded	bounded
NARADA	Mesh-first	Source-specific	Unbounded	Approx. bounded
Bayeux/Scribe	Mesh-first	Source-specific	O(log N)	O(log N)
CAN-Multicast	Mesh-first	Source-specific	O($dN_{1/d}$)	constant
HMTP/Yoid	Tree-first	Shared	Unbounded	O(max degree)
NICE	Tree-first	Source-specific	O(log N)	O(log N)

Table 2.1 compares different aspects of the protocols. The tree-first protocols (Yoid and HMTP) and centralized protocols (HBM and ALMI) create shared trees. All the other remaining protocols create source-specific trees. However, this source-specific tree is not necessarily the "best" possible tree for a given source. This is because the flexibility of choosing a specific tree for each source is limited by the structure of the mesh topology.
In general, it is difficult to analytically compute either the stretch or stress metrics for most of the protocols. In particular, an analysis of the stress metric significantly depends on the characteristics of the underlying physical topology. Analysis of Bayeux [78] and NICE [10] have shown that both protocols can guarantee a constant stretch for a "dense distribution" of members. Simulations have shown reasonable to good stretch performance of all the other protocols.
In the table we show the path length measured by the number of application-level hops and the maximum degree of a member on the data delivery tree. These metrics are indirectly related to the stress and stretch metrics and are easily analyzed for the different protocols. The number of application-level hops is unbounded for both mesh-first and tree-first approaches. CAN-multicast also has a large bound for the number of application-level hops on the data path.
In Yoid and HMTP, members themselves choose their degree, and hence provide an upper bound for the tree degree. Although Narada defines a notion of maximum degree on the mesh, sometimes it is required to relax this constraint to allow new members to join the mesh. Otherwise in some cases new members will suffer from a long latency before they find an existing mesh member with a suitable degree. Therefore, we say that the tree degree for Narada is approximately bounded.
The average control overhead is highest for the mesh first protocols, since each member exchanges state information with each other member. The average overhead at members is constant for both NICE and CAN-multicast.

Based on these observations, we found that, Centralized protocols are simple and easy to use, even on lightweight hosts (i.e. with limited resources) like PDAs and phones. On the apposite in distributed protocols, each node runs various functions for group maintenance, and since nobody has a global knowledge of the mesh, every decision (e.g. add/drop a mesh link) is based on limited local information.

2.6 Conclusion

This chapter has introduced and discussed several proposals for building an Alternative Group Communication Service. The motivation is usually to offer an alternative to the lack of deployment of inter-domain multicast routing. Another motivation is sometimes to go beyond the limitations of multicast routing protocols: some proposals try to improve the scalability in terms of concurrent number of groups; others are designed for group communications in ad-hoc networks; finally some are used to create a robust communication system in large dynamic communities, for instance for peer-to-peer applications.
We classified the proposals into several categories: based on a reflector approach, relying on permanent tunneling, creating an automatic overlay topology, or relying on a specific routing service. We have shown that these proposals widely differ, and most of them are still the subject of important research efforts. We expect that this trend will continue, since AGCSes fulfill many important needs.
Finally, it should be noted that many of the techniques discussed in this chapter can complement each other, as well as IP multicast.

Chapter 3

Our proposal: Host Based Multicast (HBM)

Contents

One of the issues raised by many AGCS is the set up of an efficient and robust overlay topology. In this chapter we introduce a centralized approach, HBM, that addresses these two key aspects, and we quickly discuss its strong points and limitations.

3.1 The HBM Proposal

3.1.1 Protocol Description

The HBM protocol automatically creates a virtual overlay topology between the various group members (sources and receivers), using point-to-point UDP tunnels between them. Everything is under the control of a single host, the *Rendez-vous Point* (or RP). This RP knows the members, their features, and the communication costs between them. He is responsible of the overlay topology calculation and its setup at each member. *This proposal therefore follows a centralized approach.*
Figure 3.1 describes the control messages exchanged by the RP and each group member. Each group member evaluates the metrics between itself and either all the other group members or a subset of them (e.g. host N1 evaluates the metrics between itself and hosts N2, N3,

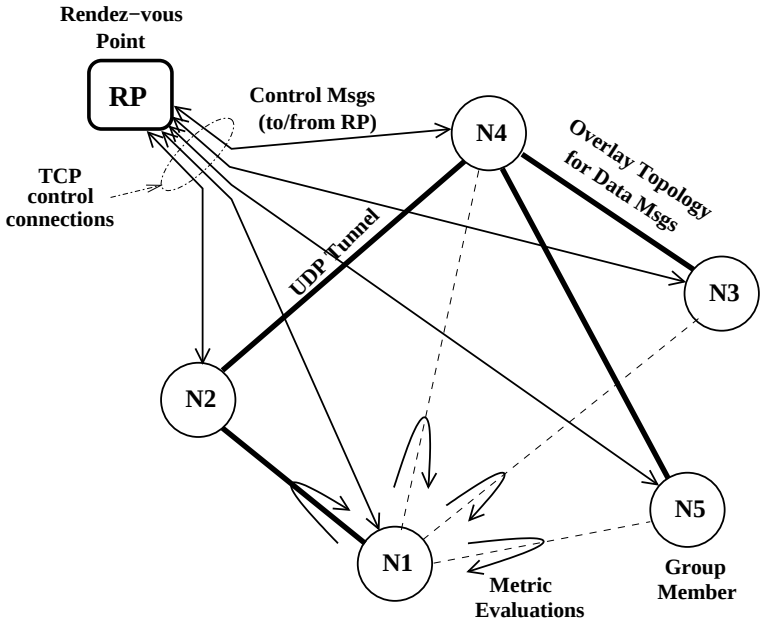


Figure 3.1. An example: HBM Connections

N4, and N5 by using the *ping* command), and informs the RP of these metrics via its TCP connection to the RP. Note that several kinds of metrics are possible, as will be explained later, even if for commodity reasons we essentially use those given by the *ping* command.
The RP then calculates an overlay topology (several possibilities exist here as will be explained later) and successively informs each group member of its neighbors.
A dynamic adaptation of the overlay topology is required for several reasons:

- to reflect the changing networking conditions,
- because new members joining an existing group are initially grafted on the existing topology in a sub-optimal way,
- after the departure of members, which can be a deliberate departure, or caused by a host, or network failure,
- because recovery actions taken by the RP after a partition lead to a sub-optimal overlay topology.

Therefore (1) all the members *periodically* evaluate the new communication costs between them (or a subset of them), and inform the RP, and (2) the RP *periodically* calculates a new topology and informs each member. These two mechanisms are *fully asynchronous*, meaning that a topology is created based on the current metric database of the RP, independently of the metric update process.
The HBM control messages are carried out in dedicated member $\rightleftharpoons$ RP TCP connections. Several such messages are defined:

- *A join message* is sent by a new member to the RP, and contains: his IP address and port to send or receive date packets, his connection kind (if known), and some special member parameters.
- *A Member List (or ML) message* is sent be the RP to a new member, and contains the current list of all members.

- *A Member Update message* is sent by the RP to all the members to update the member list when a member joins or leaves a group. This message avoids the need to list all the current members, since only the delta is sent.
- *A Leave message* is sent by a group member to leave the group.
- *A LeaveOk message* is a small message that informs the leaving member that the "leave" process is finished.
- *A Failure message* is sent by a member who detects a member failure.
- *A Metric Update (or MU) message* is sent by a member to the RP in order to update the RP's communication cost database. It contains a list of {member-metrics} pair.
- *A Topology Update (or TU) message* is sent by the RP to the members in order to inform them of the new topology. Since a TU message only contains the direct neighborhood, a different message is sent to each member.

3.1.2 HBM Group Management

We now describe the three main group management mechanisms of HBM. The presence of a central RP facilitates the allocation of a session-wide unique node identifier. This identifier is communicated to the members during all the group management operations.

3.1.2.1 Joining a Group

The RP address is supposed to be known by all potential members thanks to an out-of-band mechanism that is out of the scope of this work. To join a session, a new member contacts the RP who grafts him on the existing topology and indicates who are the current members. If not optimal, the new topology will be improved at the next topology calculation round.

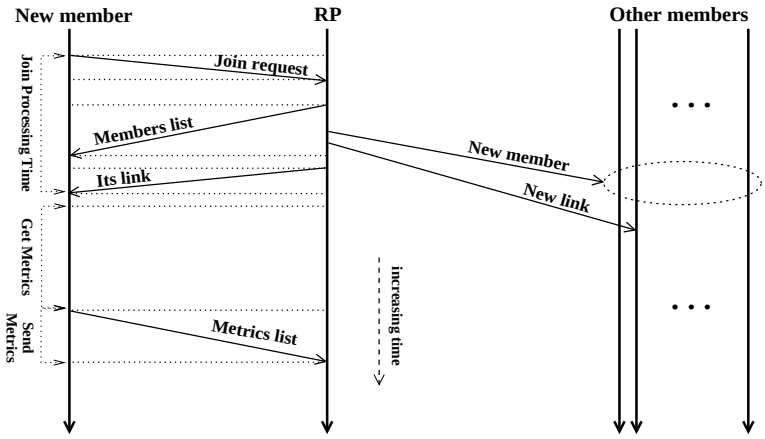


Figure 3.2. Joining a Session

The join scenario between the RP and a new member is shown in Figure 3.2. A new member sends a *Join message* to the RP. The RP replies with a *ML message* that contains the list of group members, and informs the other group members that a new member has joined the session. The RP attach randomly this new member to an existing member, and informs everybody. Later on, the new member evaluates the metrics to the other members and informs the RP using the regular metric update mechanism.

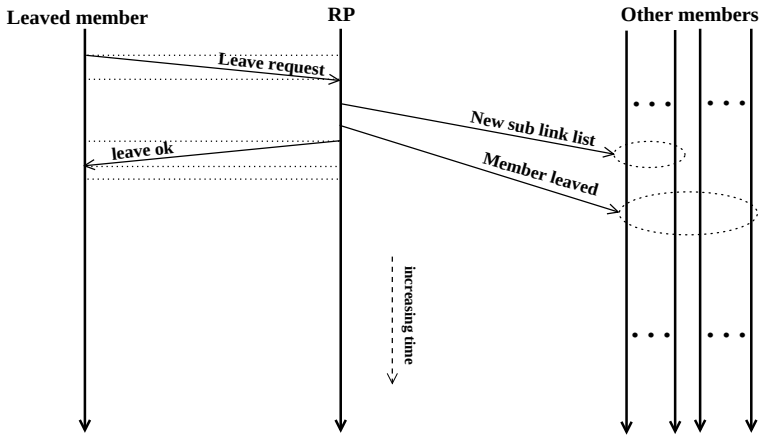


Figure 3.3. Leaving gracefully a Session

3.1.2.2 Leaving a Group Gracefully

A group member leaving the session gracefully informs the RP by sending a *Leave message*, and then waits until it receives a *LeaveOk message*. In the meantime the RP creates a new sub-topology among the neighbors of the leaving member, and informs them. The RP then informs everybody that a member has left the session, and finally it issues the *LeaveOk acknowledgment message* as shown in Figure 3.3.

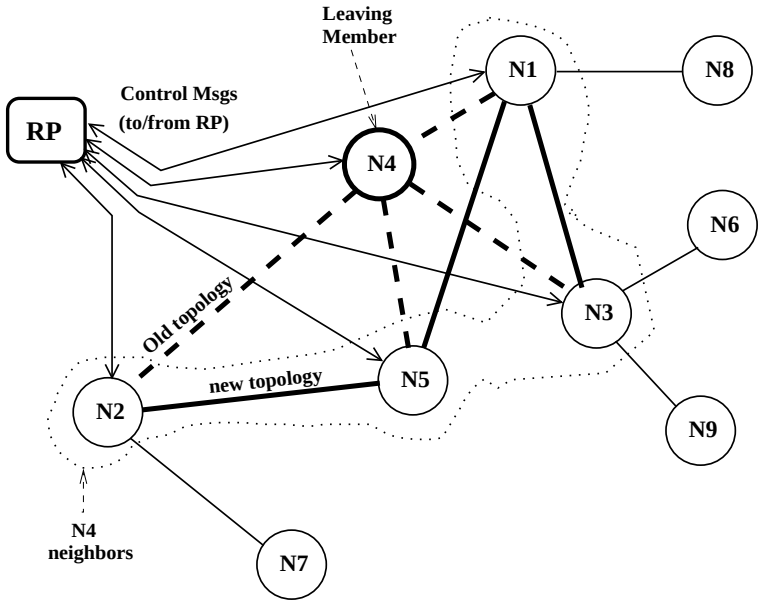


Figure 3.4. An example: N4 leaves the group gracefully.

This mechanism is illustrated in Figure 3.4 where N4 leaves the group. Before the RP sends *LeaveOk message* to N4, it creates a sub-topology between the nodes N1, N2, N3, and N5, and informs them.

3.1.2.3 Managing Member Failures

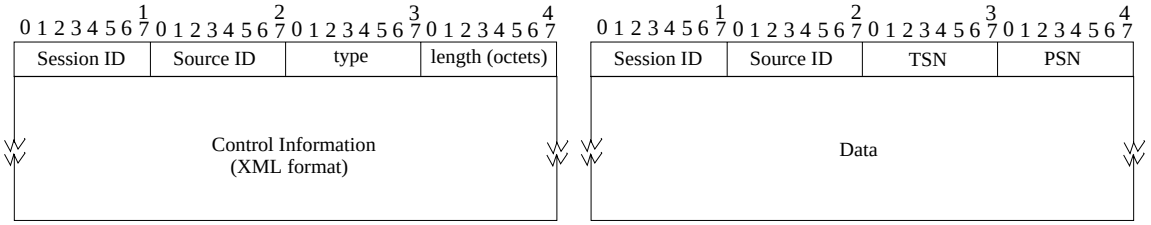
Of course the previous mechanism only applies to graceful departures. In case of a failure, a member can not inform the RP and another mechanism must be applied. One of its

neighbors, after detecting the failure, informs the RP. In case of topology partition (i.e. when this member was acting as a transit node in the topology), the RP takes an immediate recovery action, for instance by creating a new sub-topology among the direct neighbors and informs them. The RP then informs all the other members that a member of the member that failed left the session.

3.1.3 HBM Message/Packet Format

Two packet formats are defined:

- A message format used by control messages (Figure 3.5-a) between the RP and members.
- A packet format, used to forward data traffic among the members (Figure 3.5-b).



(a) TCP/IP segment

(b) Forwarded data packet format (UDP/IP datagram)

Figure 3.5. HBM packet formats.

3.1.3.1 Control Messages

All the control messages are textual and use an XML syntax. Using XML offers major advantages in terms of simplicity, extensibility, debugging and logging capabilities. This is fully compliant with the general trend where control information is textual (e.g. RTSP, SDP, HTTP, SMIL, etc), and data communications in binary format (e.g. RTP, TCP/IP, etc). For example Figure 3.6-a describes a MU message sent by N2 to the RP (shown in the first line of the message). It contains a list of RTT/loss rate metrics from N2 to the other four members. Figure 3.6-b describes a TU message sent by the RP to N2 (shown in the first line of the message). It contains the list of N2 neighbors.
Each control message contains a Session Identifier and a Source Identifier. Both of them are generated by the RP. In this work we assume that each session consists of a single group, but having multiple groups per session is also feasible.
The control information field contains the XML message that is shown in Figure 3.6. We also specify a message length which is the number of octets in the control information field.

3.1.3.2 Forwarded Data Packets

Forwarded data packets contain a specific HBM header. It defines a Topology Sequence Number (TSN) which identified the current topology, and a Packet Sequence Number (PSN). The precise goal of these two fields will be explained in chapter 7. The data field contains the multicast data packet. In other words, HBM encapsulates the multicast data packets into HBM data packets.
Since HBM is implemented as a user-level library, it simply uses unicast UDP/IP "tunnels" and TCP/IP for data packet and control messages respectively.

```
<metricupdate>2                    # MU report form node id=2
records=4                          # Number of metrics
<record>2, 1                       # metric between 2 and 1
<metric>2.50, 0.12</metric>        # RTT=10.10ms, loss=0.12%
</record>
<record>2, 3
<metric>30.10, 0.010</metric>
</record>
<record>2, 4
<metric>1.60, 0.195</metric>
</record>
<record>2, 5
<metric>10.10, 0.001</metric>
</record>
</metricupdate>                    # Message end
```

(a) Metric Update (MU) Control Information

```
<topologyupdate>2                  # Message start for node id =2
records=2                          # Number of links =2
<record>2, 4                       # link between 2 and 4
type=1                             # type of link = 1: tree link
groups=1                           # Number of groups in this link =1
<group>16843232, 1111</group>      # group ip=224.1.1.1, port = 1111
</record>
<record>2, 1
type=1
groups=2
<group>16843232, 1111</group>      # group ip=224.1.1.1, port = 1111
</record>
</topologyupdate>                  # Message end
```

(b) Topology Update (TU) Control Information

Figure 3.6. An example of MU and TU control messages.

3.2 Discussion of HBM

From the previous introduction to HBM, we can already highlight some key aspects of our proposal. The HBM proposal:

- *has a limited scalability...*
  More generally any overlay multicast solution based on point-to-point communications has scalability problems (even if having a central RP in HBM adds some more limitations). Yet many collaborative work sessions only include a limited number of hosts/sites and scalability is not a problem then. Besides a single HBM node can easily serve many local participants using the locally available multicast routing service. In chapter 5, we will analyze the control aspects of HBM and we will show that its scalability can very easily be improved by making sure the associated overhead does not exceed a given threshold.

- *greatly relies on the RP reliability...*
  The RP is a potential point of failure. If this RP is collocated with the primary source (if any), then this is not an issue as any failure of the source host would anyway compromise the session. We can also define one or more secondary RPs that will be used in the case of primary RP failure. To keep the group information up to date, the primary RP periodically informs the secondary RPs of the current group information.

- *and is intrinsically fragile...*
  Members are a potential points of failure. And if a transit node of the overlay topology fails, a partition may be created. In order to avoid this partition, in chapter 6, we describe created a strategy that adds a certain number of Redundant Virtual Links (RVL) to the overlay topology.

  Packet losses can also be caused by transient incoherences that are the result of a topology update decided by the RP in order to improve the overlay. Since the topology update is not synchronous, at some point of time a subset of the members will be informed of (and probably will use) the new topology, while others will still use the previous old one. This routing incoherency can easily lead packets in transit to be lost. In chapter 7 we discuss several solutions to the problem.

On the other hand:

- *this is a simple solution...*
  As all the information is centralized in the RP, there is no coherency problem and it does not create too much load on the nodes (an asset in case of lightweight hosts like PDAs). This is completely different in distributed solutions like Narada [52] where each node runs various algorithms for group maintenance and incremental mesh quality improvement.

- *which can easily create a "not too bad" topology...*
  The topology is optimal with respect to the metric database at the time of its creation. So the overlay topology quality depends on:

  - the database freshness: how often are these metrics updated?
  - the kind of topology created
  - the metric quality: which metrics are used, and what is their accuracy?

  The update frequency of the metric database depends on various criteria like the group size (the larger the group, the lower the frequency) and node specificities (a powerful workstation can update the database more frequently than a PDA).

  Several potential overlay topologies exist [86] each of them having distinctive features: *bus*, *tree*, *ring*, *star*, *sun*, and *hybrid*, even so far overlay multicast work essentially focused on trees (this is also true for the work carried out in this thesis). Finally, in chapter 4, we describe a tree creation strategy which considers a digest of several metrics.

3.3 Conclusion

This chapter has discussed the issues raised by the creation and the management of an overlay topology. We argue that a centralized solution where the group membership is known by a RP, has many benefits over distributed solutions. Having a centralized topology management is *simple* (no coherency problem), *efficient* (the RP can create an optimal topology with respect to the metric database accuracy) and *takes advantage of known node features* during the topology creation process (e.g. to have stable transit nodes).
The following chapters focus on several key aspects of HBM:

- *Topology creation:* the overlay topology creation process is essential to any application-level multicast solution. This topic is addressed in chapter 4.

- *Scalability:* a centralized approach has scalability limitations. Chapter 5 studies the possible solutions to improve scalability.

- *Robustness:* overlay multicast solutions are by nature fragile. Chapter 6 introduces a strategy that adds redundant virtual links to avoid partitions after a node failure, and chapter 7 discusses possible solutions to avoid packet looses during a topology update process.

- *Security:* chapter 8 introduces an example which explains how HBM can help to design a secure group communication service called Virtual Private Routed Network (VPRN).

Part II

Evaluation and Improvements

Chapter 4

Creating an Overlay Multicast Topology

Contents

We have seen in chapter 3 that the RP creates a virtual overlay topology between the group members. The questions that this chapter answers are:

- How to create this overlay?
- How can we evaluate the overlay performance?

We show in particular that an overlay topology can very easily be created, using a new algorithm which takes into account several metrics such as RTT and losses.

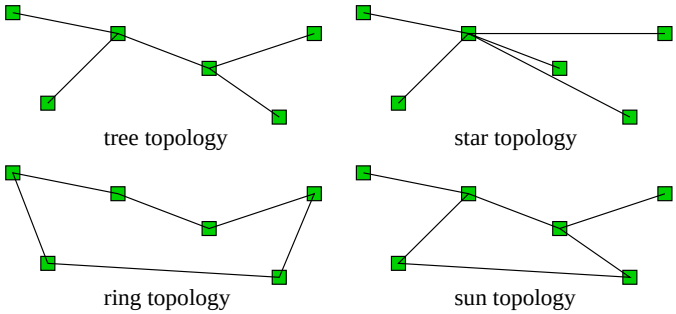


Figure 4.1. Some possible overlay topologies

4.1 Possible Topologies

Several overlay topologies (Figure 4.1) are possible and are briefly described:

bus: this is a serial connection of all the nodes.

tree: several kinds of trees are possible, like the Shortest Path Trees (SPT) and Minimum Spanning Trees (MST). A SPT is per-source. In case of different sources, several SPT must be created which turns out to be costly but efficient from the transmission point of view. On the opposite a MST is source independent which is an asset from a management point of view when several sources exist, but suboptimal from the transmission point of view.

ring: this is the solution of the "Traveling Salesman Problem" (TSP). This problem can be summarized as: "Given a finite number of "cities" along with the travel costs between them, find the cheapest way of visiting all the cities and returning to the starting point". This is known to be a NP-complete problem, but heuristics exist. Reference [99] gives more information on how to solve large-scale instances of the TSP. This topology is source independent.

star: all the nodes are connected around a central node. This node can be the source (in case of a single source session), in which case the transmission delay is minimal, but this is not mandatory. This is typically the kind of overlay topology created with a reflector (section 2.3.1).

sun: a "sun" is a "star" with a non null diameter. It is therefore composed of an internal "ring" with peripheral "solar beams" (i.e. a leaf directly connected to a node of the central ring).

hybrid: hybrid topologies mix the previous solutions. For instance, several sub-trees can be connected to a central ring.

Choosing one of these topologies has serious impacts on robustness and performances. In [86] we have compared the tree and ring topologies through simulations. It takes in input a randomly and homogeneously distributed[1] set of N nodes. A topology solver is run, creating MST and Ring topologies. Each topology is then analyzed, failures introduced, and statistics gathered.

[1] In case of non-homogeneous node distributions, e.g. to simulate the impacts of a transmission line, the Traveling Salesman solver that creates the ring topology must be modified to take it into account. [79] page 445 gives such a algorithm.

When we focus on average group-shared delays [91], MST has a lower average delay than a ring but if we consider robustness then this is the contrary (chapter 6).
In the remaining of this document *we only focus on MST shared trees.*

4.2 Metrics for Topology Creation

4.2.1 Metric Evaluation

Using the ping command to evaluate the RTT and losses metrics creates an important overhead, (section 5.2.3). Other metrics such as one-way delay, bandwidth, and bottleneck bandwidth are possible.
For instance, we can estimate the one-way delay metric from cyclic-path delay measurements [48] that does not require any kind of synchronization among the nodes. This approach is taking into account the asymmetric nature of the network. In [44, 45], the authors suggest a scalable Internet-wide architecture, called Internet Distance Map Service (IDMaps), which measures and disseminates distance information on the global Internet. Higher-level services can collect such delay information to build a virtual distance map of the Internet and estimate the delay between any pair of IP addresses.
In [58], the authors distinguish between the available bandwidth and bottleneck bandwidth of a route. The available bandwidth of a route is the maximum bandwidth at which a host can transmit at any point in time along that route. The available bandwidth is limited by other traffic along that route. The bottleneck bandwidth of a route is the bandwidth of bottleneck link on that route. In most networks, as long as the route between the two hosts remains the same, the bottleneck bandwidth remains the same (it is not affected by other traffic).
The current tools to measure bandwidths (1) either measure all link bandwidths instead of just the bottleneck, (2) or only measure the bandwidth in one direction, and/or 3) actively send probe packets. The pathchar [54], clink [28], pchar [67], and tailgater [59] tools measure all the link bandwidths along a path through the Internet, which can be time-consuming and unnecessary for applications that only want to know the bottleneck bandwidth. Nettimer [60] measures the bottleneck link bandwidths in real time in the Internet.
For commodity reasons, in this work we only consider the RTT and loss metrics given by a *ping* command.

4.2.2 Synthetic Metrics

The topology creation algorithm depends on the synthetic metric (*synthetic_metric*) that is the mix of many metrics, such as RTT or one-way delay, packet losses, stability of nodes, etc. In this work, we suppose that there are two metrics only: RTT and loss. The *synthetic_metric* is then given by:

$$synthetic\_metric = \frac{100}{(100 - loss)} \times RTT$$

where loss $\in [0\% \ldots 100\%]$.

4.2.3 Nodes Classification According to their Capability

Some of the nodes can turn out to be unstable (e.g. a mobile node with a bad wireless connection). Even if HBM includes redundancy and failure discovery mechanisms, instability must be taken into account when creating the topology. The idea is to have stable transit

nodes while unstable ones are moved to the leaves of the topology. Of course the stability of a node is unknown when he first joins a session. A default (conservative) value is first assigned to the *node_stability* variable and this latter is regularly updated as time goes by. In order to make adaptation possible, we associate a "capability" to each node. This capability has three possible values: disconnected, leaf_only, or transit_possible. We call the leaf-only nodes *non-Core Member* (nonCM) because they have a marginal role in the overlay: they can not forward traffic. On the opposite, nodes that can be transit nodes are called *Core Member* (CM). We first calculate a normalized capability, $NCap$:

$$NCap(node) = f(user\_desires, node\_stability, RP\_param)$$

where RP_param is a parameter specified by the RP to influence the capability of a node (e.g. if all the users choose to be `leaf_only`, then the RP can oblige some them to be transit node). Then $NCap(node)$ is compared to predefined thresholds in order to determine the exact capability of the node:

if ($NCap(node) \in [0; \alpha[$), then the node is disconnected (exceptional if α is small);
if ($NCap(node) \in [\alpha, \beta]$), then the node has capability "leaf_only" (nonCM);
if ($NCap(node) \in]\beta, 1]$), then the node has capability "transit_possible" (CM);

This is a lightweight mechanism as the RP already keeps per-node state information. It only adds four variables: the *user_desires*, the *node_stability* (dynamically updated), the RP_param and the node capability.

4.3 Overlay Topology Creation

From the previous section, we can use both the *synthetic metric* and the node capability ($NCap$) to create the overlay topology. In order to adapt the overlay multicast networks, there are two points of view: (1) the delay (e.g. the tree diameter), and (2) the bandwidth usage (e.g. the link stress). However, the objective of limiting delay can conflict with the objective of optimizing the interface bandwidth usage, so one must strike an appropriate balance between these objectives. To that purpose is first created among Core Members based on *synthetic_metric* only. Then, non-Core Members are grafted int the tree topology; taking into account both the *synthetic_metric* and the $NCap$.

4.3.1 Tree Construction Algorithm among Core Members

A tree is first created between core members. At any stage of the algorithm, two sets of members are defined:

- S_{in} contains the members already added to the current overlay topology. Usually S_{in} contains several subsets (C_i) that contain the various sets of already connected to each other in the overlay topology.
- S_{out} contains the members not yet added to the overlay topology.

The algorithm consists of N-1 steps. First of all, it chooses the two closest members in S_{out} and creates the first subset (C_0) in S_{in} (i.e. it creates a link between them in the overlay). Secondly, it chooses the next two closest members that are either:

- two members in S_{out}, and in that case it creates a new subset for them,

- one in S_{in} and another one in S_{out}, and in that case it adds the member of S_{out} to the subset of the member of S_{in}, or

- two members in S_{in} but in different subsets, and in that case it merges these two subsets.

It repeats the previous process until both there is no member in S_{out} any more, and there is a single subset in S_{in}.

Example:

Let's consider a group with 6 nodes. The overlay is created as follows (Figure 4.2):

- **Step 1**: S_{in} = [NULL] and S_{out} = [1, 2, 3, 4, 5, 6], Min ($S_{out} \rightleftharpoons S_{out}$) is $1 \rightleftharpoons 3$, so a new subset C_0 is created in S_{in}.

- **Step 2**: S_{in} = [C_0={1, 3}] and S_{out} = [2, 4, 5, 6], Min ($S_{in} \rightleftharpoons S_{out}$ and $S_{out} \rightleftharpoons S_{out}$) is nodes $4 \rightleftharpoons 5$, so a new subset C_1 is created in S_{in}.

- **Step 3**: S_{in} =[$C_0 = \{1, 3\}$, $C_1 = \{4, 5\}$] and S_{out} = [2, 6], Min ($S_{in} \rightleftharpoons S_{out}$, $S_{out} \rightleftharpoons S_{out}$ and $C_0 \rightleftharpoons C_1$) is $3 \rightleftharpoons 6$, so a node 6 is added to subset C_0.

- **Step 4**: S_{in} =[$C_0 = \{1, 3, 6\}$, $C_1 = \{4, 5\}$] and S_{out} = [2], Min ($S_{in} \rightleftharpoons S_{out}$, $G_{out} \rightleftharpoons G_{out}$ and $C_0 \rightleftharpoons C_1$) is $3 \rightleftharpoons 4$, so the subsets C_0 and C_1 are merged into the subset C_0.

- **Step 5**: S_{in} =[$C_0 = \{1, 3, 4, 5, 6\}$] and S_{out} = [2], Min ($S_{in} \rightleftharpoons S_{out}$ and $S_{out} \rightleftharpoons S_{out}$) is $2 \rightleftharpoons 4$, so a node 2 is added to subset C_0. because there is a single subset C_0 and S_{out} = [NULL], the algorithm is terminated.

Briefly, this algorithm groupes the closest members (i.e. subsets in S_{in}) and joins the subsets to each other.

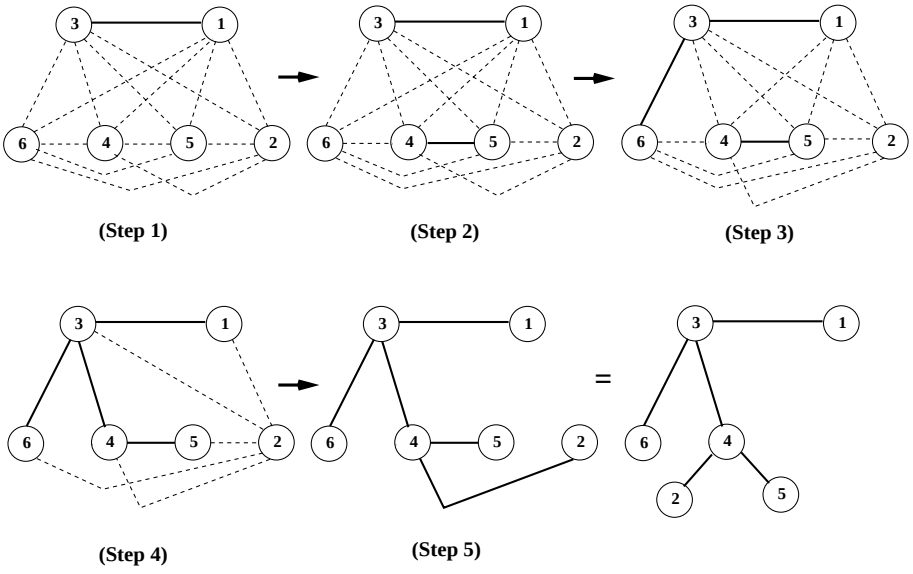


Figure 4.2. An Example of Tree Construction of 6 Core Members

4.3.2 Tree Construction Algorithm among non-Core/Core Members

We can modify the previous overlay topology creation algorithm to include nonCore Member (that are necessarily leaves). We use the same mechanism except that when the member in S_{in} is a nonCore Member, this member will not be connected to any current members either in S_{out} or in another subset of S_{in} (i.e. it means that when a nonCore Member is connected to the overlay topology, this member is deleted from the sets S_{in} and S_{out}).

Example:

Let's consider the example of section 4.3.1, but assume that a member 4 is a non-Core Member (e.g. it uses low speed modem), Figure (4.3). The algorithm follows these steps:

- **Step 1**: S_{in} = [NULL] and S_{out} = [1, 2, 3, 4, 5, 6], Min ($S_{out} \rightleftharpoons S_{out}$) is $1 \rightleftharpoons 3$, so a new subset C_0 is created in S_{in}.
- **Step 2**: S_{in} = [C_0={1, 3}] and S_{out} = [2, 4, 5, 6], Min ($S_{in} \rightleftharpoons S_{out}$ and $S_{out} \rightleftharpoons S_{out}$) is $4 \rightleftharpoons 5$, so a new subset C_1 is created in S_{in} and contains {5} only because the node 4 is nonCore Member.
- **Step 3**: S_{in} =[C_0 = {1, 3}, C_1 = {5}] and S_{out} = [2, 6], Min ($S_{in} \rightleftharpoons S_{out}$, $S_{out} \rightleftharpoons S_{out}$ and $C_0 \rightleftharpoons C_1$) is $3 \rightleftharpoons 6$, so a node 6 is added to subset C_0.
- **Step 4**: S_{in} =[C_0 = {1, 3, 6}, C_1 = {5}] and S_{out} = [2], Min ($S_{in} \rightleftharpoons S_{out}$, $G_{out} \rightleftharpoons G_{out}$ and $C_0 \rightleftharpoons C_1$) is $1 \rightleftharpoons 5$, so the subsets C_0 and C_1 are merged into the subset C_0.
- **Step 5**: S_{in} =[C_0 = {1, 3, 5, 6}] and S_{out} = [2], Min ($S_{in} \rightleftharpoons S_{out}$ and $S_{out} \rightleftharpoons S_{out}$) is nodes $2 \rightleftharpoons 5$, so a node 2 is added to subset C_0. because there is a single subset C_0 and S_{out} = [NULL], the algorithm is terminated.

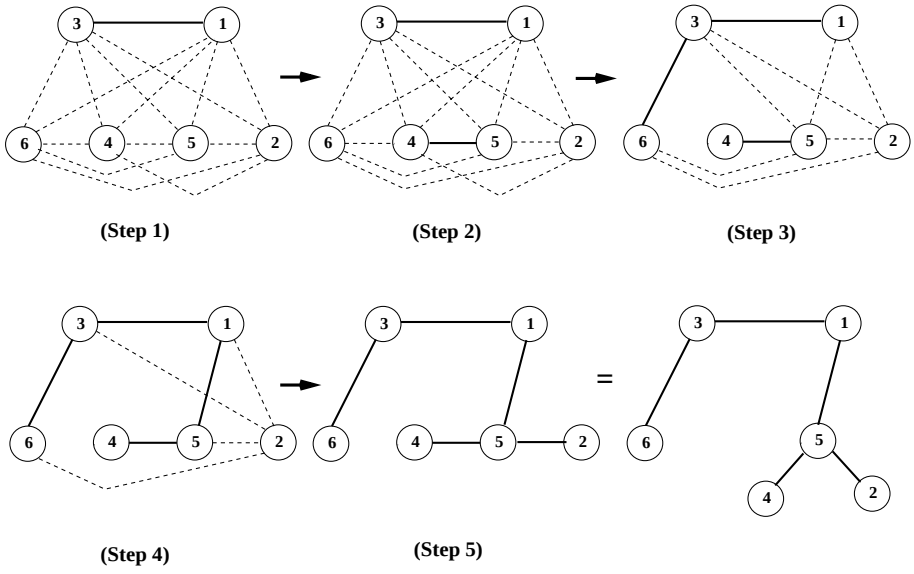


Figure 4.3. An Example of Tree Construction of 6 non-Core/Core Members

4.3.3 Constrained Tree Construction Algorithm

In any case, we can limit the number of neighbors a member has in the overlay topology. In that purpose we can reuse the algorithm of section 4.3.2 (where Core members have unlimited number of neighbors and nonCore Members have only one neighbor). Since core Members too do not enjoy an infinite bandwidth, we can additionally limit their number of neighbors to *max_neighbors*. A node having already *max_neighbors* neighbors is removed from both S_{in} and S_{out} and therefore can no longer be implied in the topology creation process.

4.4 Performance Evaluation

4.4.1 Performance Evaluation Parameters

The quality of the topology is judged by the following metrics as described in [52, 104]:

- **Tree Cost (T_{cost})** is the sum of delays on the tree links. Tree cost is a convenient, though somewhat simplified, metric to capture the total network resources consumption of a tree.

  The ratio between the tree cost to the average star cost that of a corresponding multi-unicast (i.e. star topologies centered around all possible sources) is called the **cost ratio(r_{cost})**. Let $L_{delay}(i)$ be the delay of the i^{th} link. Then:

$$T_{cost} = \sum_{i=1}^{N-1} L_{delay}(i)$$

$$r_{cost} = \frac{T_{cost}(overlay)}{T_{cost}(unicaststar)}$$

- **Tree Delay ($T_{delay-overlay}(x, y)$)** is the delay from a member (i.e. x) to another (i.e. y) along the tree. Similarly $T_{delay-unicast}(x, y)$ is the delay from member x to y using a unicast shortest path connection.

$$T_{delay-overlay} = \frac{\sum_{i=1}^{N-1}(\frac{\sum_{j=1,i\neq j}^{N-1} T_{delay-overlay}(i,j)}{N-1})}{N}$$

$$T_{delay-unicast} = \frac{\sum_{i=1}^{N-1}(\frac{\sum_{j=1,i\neq j}^{N-1} T_{delay-unicast}(i,j)}{N-1})}{N}$$

  The ratio between tree delay and unicast delay is called **delay ratio (r_{delay})**:

$$r_{delay} = \frac{T_{delay-overlay}}{T_{delay-unicast}}$$

- **Group diameter ($G_{diameter}$)** is the maximum delay over the tree between any pair of members. It represents the time after which a packet is assured to reach every member no matter who the source is.

$$G_{diameter-overlay} = \max_{i \neq j}(T_{delay-overlay}(1,2), ..., T_{delay-overlay}(i,j), ..., T_{delay-overlay}())$$

The ratio of group diameter to the diameter of any multi-unicast (i.e. all the star topology centered around any possible sources) is the **group diameter inflation**.

$$G_{diameter-unicast} = \max_{i \neq j}(T_{delay-unicast}(1,2), ..., T_{delay-unicast}(i,j), ..., T_{delay-unicast}())$$

$$G_{diameter-inflation} = \frac{G_{diameter-overlay}}{T_{diameter-unicast}}$$

- **Link Load or stress(S_i):** is the number of identical copies of a packet carried by a physical link. In order to capture the stress, we introduce the notation of **resource usage** ($resource\_usage$) that is another notation of tree cost:

$$resource\_usage = \sum_{i=1}^{L} S_i \times L_{delay}(i)$$

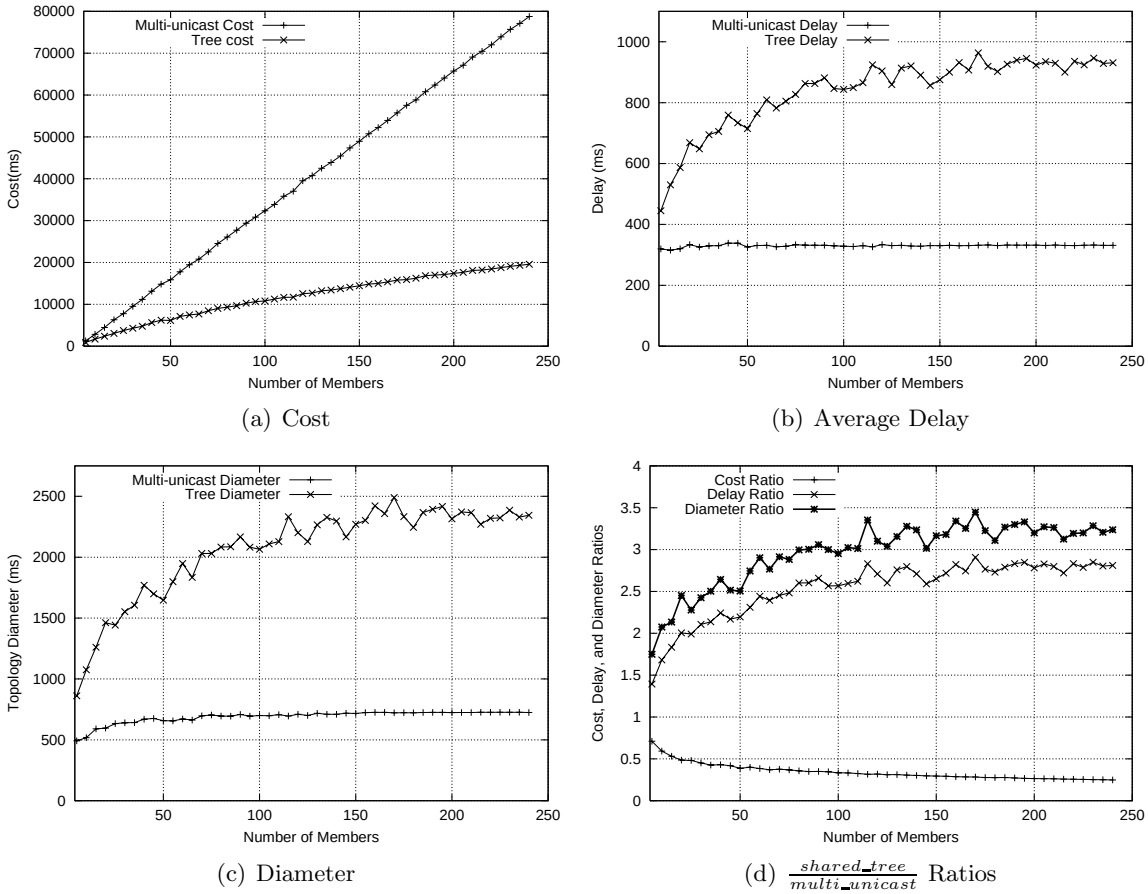
4.4.2 Performance Evaluation Results

4.4.2.1 Experimental Conditions

We implemented the constructed tree creation algorithm of section 4.3.3 in C++ on PC/Linux machines. The experiments reported in this chapter are based on a large interconnection transit-stub network, composed of 600 core routers, and generated by the Georgia Tech Model (GT-ITM) [102]. Some of these routers are interconnection routers, others, at the leaf of the topology, are access routers connecting the client members. We then choose N members randomly among the 243 possible leaves and consider a single member within each of these members. We compared (1) the multi-unicast solution, where a sending member unicasts a copy of each packet to the (N-1) members, and (2) our shared tree creation algorithm. This tree is constrained so that the maximum degree of each node is 6. Each point in the figures (except for the stress distribution figure) is the average (10 times) over N members chosen randomly among the 243 possibilities. Note that during simulations, the physical links are only limited by their unidirectional delay (assumed constant), not by their bandwidth, which leads to underestimate the effect of the link stress.

4.4.2.2 Results and Discussion

Figures 4.4 and 4.5 compare performance metrics (average delay and diameter) and resource consumption metrics (cost and link stress) with the multi-unicast and constrained shared tree topologies. Figure 4.5-c is a zoom that emphasizes the tail distribution, but that does not show that most links have a stress at most equal to 5, and 325 links a stress equal to 1. The multi-unicast topology is quickly limited by the fan-out of the sending site (equal to N -1) which creates a high stress on the first few links. This is visible on Figures 4.5-a (look at



(a) Cost (b) Average Delay

(c) Diameter (d) $\frac{shared\_tree}{multi\_unicast}$ Ratios

Figure 4.4. Multi-unicast versus constrained shared tree comparison

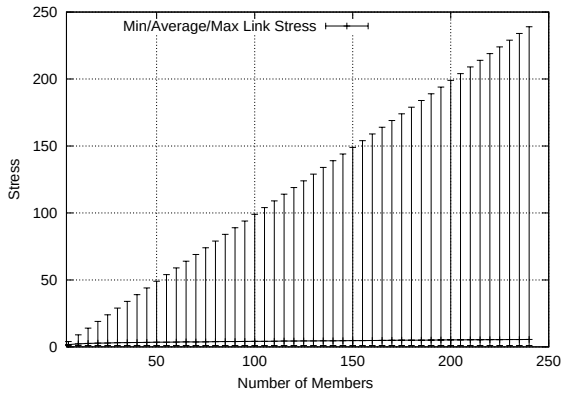
the maximum stress) and 4.5-d (stress distribution, showing a tail distribution at x = N - 1). Consequently, the multi-unicast approach is definitely not a reasonable solution when there are more than a few tens sites (especially as these tests underestimates the effects of the link stress).

On the opposite the constrained tree, by construction, limits this maximum stress (at most 6 neighbors in these experiments), no matter what is the number of sites, N. The price to pay is a higher maximum delay/diameter of the topology, but experiments show that the stretch factor when compared to the unicast case remains inferior to 3.5 which is fairly reasonable.

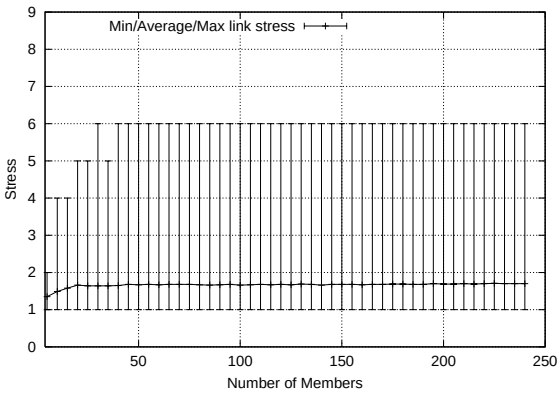
Note that in case of a single source application, the distribution topology can be optimized using a per-source tree. This is made possible by the total control over the topology at RP. This will of course be the preferred solution each time the application make it possible.

4.5 Conclusion

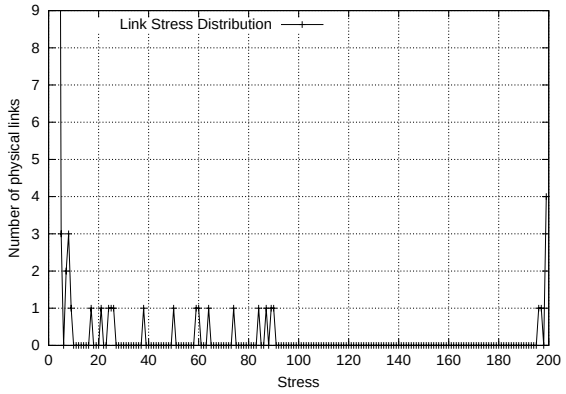
In this chapter we explained how to create the overlay topology by using a synthetic metric that contains many metrics such as the RTT (or one-way delay), loss, and node capability. We evaluate the performance of the overlay topology and compare it with that of multi-unicast connections.



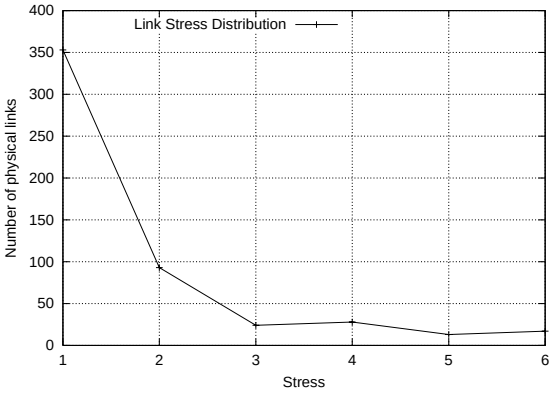
(a) Multi-unicast min/aver/max Link Stress



(b) Shared tree min/aver/max Link Stress



(c) multi-unicast link stress distribution with N = 200 sites (ZOOM)



(d) Shared tree link stress distribution with N = 200 sites

Figure 4.5. Multi-unicast versus constrained shared tree link stress comparison

Chapter 5

Improving the Scalability

Contents

From the chapter 3, we have introduced a centralized proposal, HBM, which typically has scalability problems. Yet we show in this chapter that its scalability can very easily be improved. In particular the control information being exchanged between the group members and the RP can be rate limited not to exceed a given threshold.

5.1 Introduction

The centralized nature of HBM naturally limits the scalability in terms of the number of simultaneous members in a session. The number and size of the metric update and topology update messages can quickly waste a significant amount of bandwidth on the network and

processing power at the RP. Besides, the necessity for each member to know other members and to periodically evaluate the new communication costs with them also limits the scalability of the solution.
More generally we believe that any overlay multicast solution based on point-to-point communications and that tries to create "not too bad" topologies has scalability limitations, even if this is more acute in a centralized solution.
HBM and many other overlay multicast solutions target applications, like collaborative working environments, that only need a reasonable scalability, in the order of a few hundreds of members at most. Therefore our tests in this chapter are limited to 200 participants. Anyway, a single HBM member per site is sufficient when intra-domain multicast is possible in this site, since this HBM member will hide all the internal members and behave as a reflector for them. Therefore, the true number of members in a session, as seen by the application, can be largely higher than the number of members known by HBM.

5.2 Improving the Scalability

After a detailed modeling of the control traffic overhead, this section introduces four strategies to increase the scalability of the HBM protocol.

5.2.1 Mathematical Model

5.2.1.1 Metric Update (MU) Message Incoming Rate

Let N be the number of members in the session, $T_{mu}(N)$ the metric update period at a member, $s_{mu}(N)$ the size of a single metric update message, s_{mu_header} the fixed size of a message header, $n_{rmu}(N)$ the number of records in each metric update message, each record being s_{rmu} bits long (assumed to be a constant). We assume that these parameters are the same for all members. The incoming rate, from the RP point of view, for all metric update messages, $R_{mu}(N)$, is given by:

$$\begin{aligned} R_{mu}(N) &= \frac{N * s_{mu}(N)}{T_{mu}(N)} \\ with: \; s_{mu}(N) &= s_{mu\_header} + n_{rmu}(N) * s_{rmu} \end{aligned} \tag{5.1}$$

5.2.1.2 Topology Update (TU) Message Outgoing Rate

Let $T_{tu}(N)$ be the topology update period at the RP. Let n_l be the total number of links in the virtual topology. Since each member needs a link to get connected, it follows that $n_l = (N-1)$. Having more links would create loops which is avoided in the present work (i.e. we do not take into account the possibility of having additional links for improved robustness as in Chapter 6. Since each link is common to two members, a record for a link is sent twice, in two different topology update messages.
Let $s_{all\_tu}(N)$ be the size of all topology update messages sent after a topology update, s_{tu_header} the fixed size of a message header, s_{rtu} the size (assumed to be a constant) in bits of each record in each message. $s_{all\_tu}(N)$ is given by:

$$s_{all\_tu}(N) = N * s_{tu\_header} + 2 * n_l * s_{rtu}$$

The outgoing rate, from the RP point of view, for all topology update messages, $R_{tu}(N)$, is given by:

$$R_{tu}(N) = \frac{s_{all\_tu}(N)}{T_{tu}(N)} \tag{5.2}$$

5.2.1.3 Total Rate of Control Messages

It follows that the total rate, $R_{ctrl}(N)$, for all HBM control messages is the sum of $R_{mu}(N)$ and $R_{tu}(N)$:

$$\begin{aligned} R_{ctrl}(N) &= R_{mu}(N) + R_{tu}(N) \\ R_{ctrl}(N) &= \frac{N * s_{mu\_header} + N * n_{rmu}(N) * s_{rmu}}{T_{mu}(N)} + \\ &\quad \frac{N * s_{tu\_header} + 2 * (N-1) * s_{rtu}}{T_{tu}(N)} \end{aligned} \tag{5.3}$$

5.2.2 Strategies to Reduce the Control Overhead

5.2.2.1 General Ideas

In order to limit the control traffic overhead, $R_{ctrl}(N)$ must not be greater than a given threshold. This threshold can be either a fixed predefined value, or be expressed as a given percentage, α, of the sum of both the average data traffic rate, R_{data} and the control traffic rate, $R_{ctrl}(N)$. Let's consider this second case (RTP/RTCP do the same). Therefore:

$$\begin{aligned} R_{ctrl}(N) &\leq \alpha * (R_{ctrl}(N) + R_{data}) \\ R_{ctrl}(N) &\leq (\frac{\alpha}{1-\alpha}) * R_{data} = R_{ctrl\_max} \end{aligned} \tag{5.4}$$

For a given $T_{mu}(N)$ and $T_{tu}(N)$, let $N_{n\_rmu\_max}$ be the maximum N such that having $n_{rmu} = (N-1)$ records, i.e. the maximum, in each metric update message satisfy this constraint. $N_{n\_rmu\_max}$ is the solution of a second order equation: $R_{ctrl}(N) = R_{ctrl\_max}$ (see equation 5.3).
This value of N is a major threshold, since for groups having less than $N_{n\_rmu\_max}$ members, an optimal solution is feasible, each member generating messages containing the new communication cost to all other members, while the total control message rate remains below a maximum threshold.
On the contrary, *for groups having more than $N_{n\_rmu\_max}$ members, specific measures must be taken in order to satisfy equation 5.4.* Several strategies are possible, depending on the protocol parameters we play upon:

Strategy 1 (none) is the reference, and *does not* include any optimization. Therefore, as N grows, the constraint of equation 5.4 is not necessarily fulfilled.

strategy 2 limits the number of records, $n_{rmu}(N)$, in a metric update message. When $N > N_{n\_rmu\_max}$, this number of records is progressively reduced until a lower predefined bound is reached, n_{rmu_min}, at $N = N_{n\_rmu\_min}$. Then, for $N > N_{n\_rmu\_min}$ the number of records remains constant and equal to n_{rmu_min}.

strategy 3 limits the number of records too in a metric update message. The difference with strategy 2 is that when $N > N_{n\_rmu\_max}$, the number of records remains constant, n_{rmu_max}, rather than being reduced. In order to keep the total control message rate below the upper bound, the $T_{mu}(N)$ period is progressively increased when $N > N_{n\_rmu\_max}$.

strategy 4 also limits the number of records in each metric update message, but less aggressively than the previous two strategies. To compensate, the $T_{mu}(N)$ period is aggressively increased when $N > N_{n\_rmu\_max}$.

Therefore these strategies try to find a balance between the various protocol parameters, that are shown in the Figures 5.1, 5.2, 5.3, 5.4, 5.5, and 5.6, by playing with the frequency and the size of the control messages. In some cases the messages are more frequent but contain less information, and vice-versa. Each of these strategies is now introduced with more details.

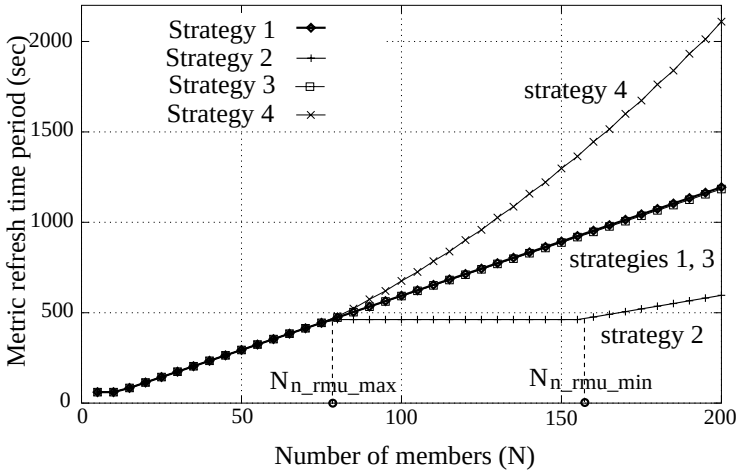


Figure 5.1. Metric update period, $T_{mu}(N)$, at a member.

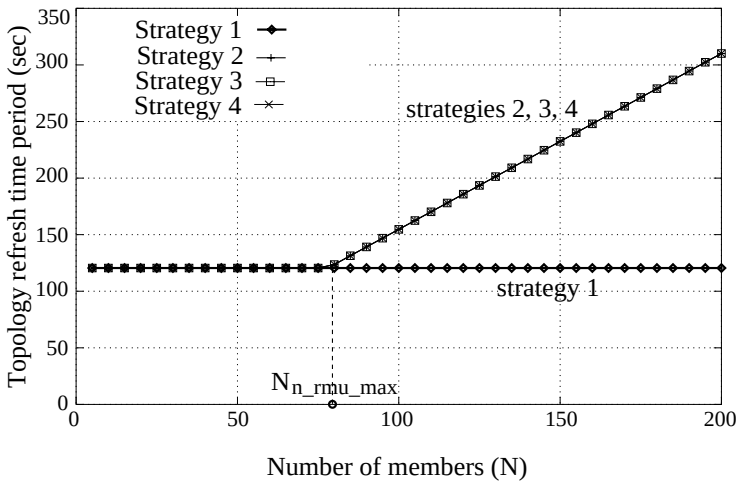


Figure 5.2. Topology update period, $T_{tu}(N)$, at the RP.

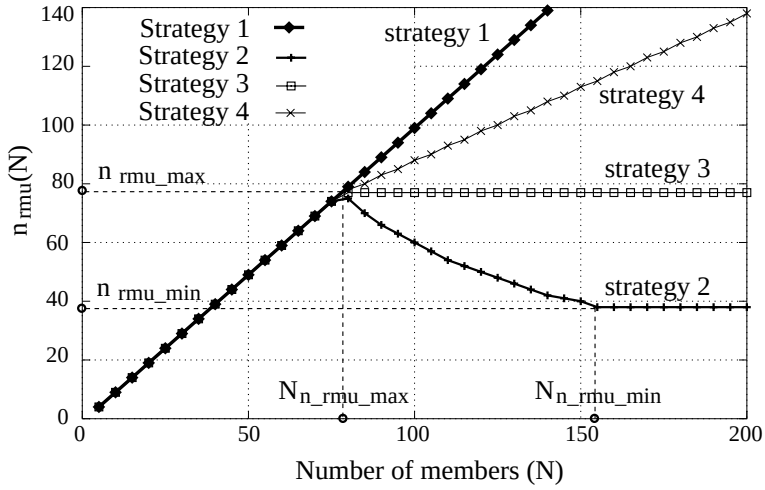


Figure 5.3. Number of records, $n_{rmu}(N)$, in a metric update message.

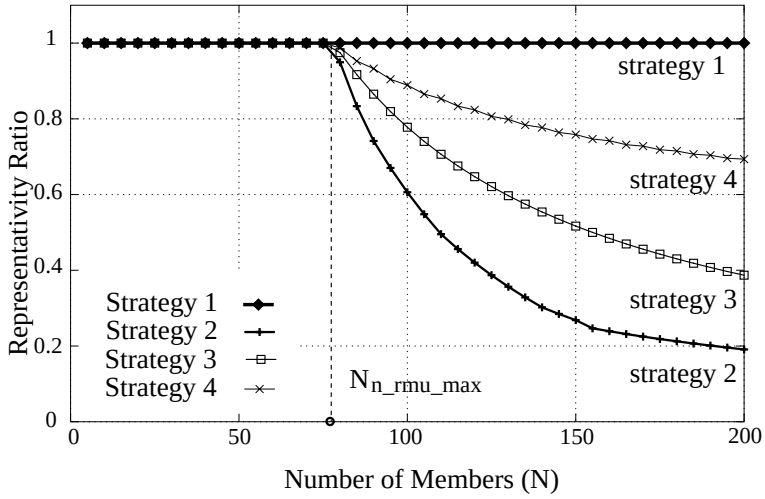


Figure 5.4. $\frac{n_{rrm}(N)}{(N-1)}$ ratio, i.e. representativity of a MU message.

5.2.2.2 Strategy 1: Non Optimized

The first strategy, not optimized, is controlled by the following equation set:

$$\begin{aligned} n_{rmu}(N) &= N-1 \\ T_{mu}(N) &= max(T_{mu}, T_{m\_eval} * (N-1)) \\ T_{tu}(N) &= T_{tu} \end{aligned}$$

where T_{m_eval} is the time (assumed constant) required to evaluate the communication cost between two members (section 5.2.3). Since the evaluation of $(N-1)$ metrics grows linearly with N, the $T_{mu}(N)$ period is only constant (T_{mu}) when $T_{m\_eval} * (N-1) \leq T_{mu}$, and then equal to $T_{m\_eval} * (N-1)$. Note that the same constraint applies to the three other strategies.

5.2.2.3 Strategy 2

The second strategy is controlled by the following equation set:

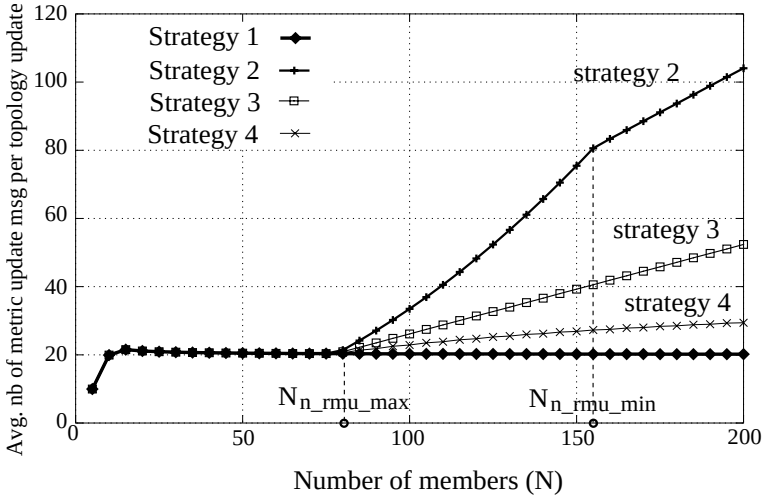


Figure 5.5. Average number of metric update message per topology update.

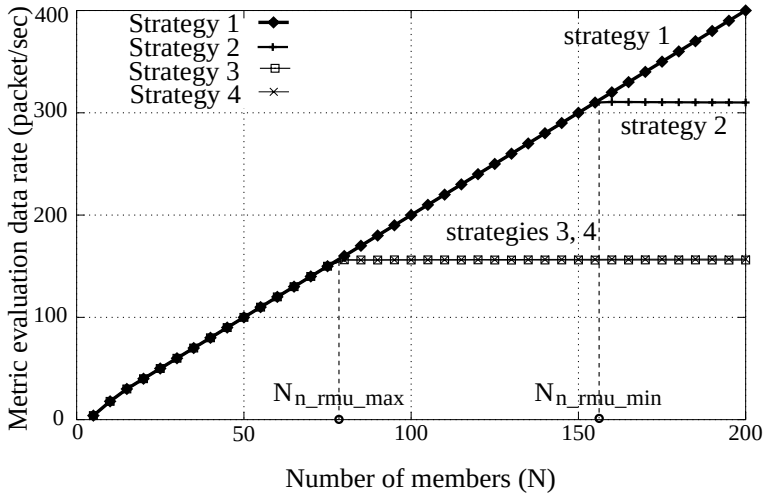


Figure 5.6. Metric evaluation overhead (with the ping command).

$$n_{rmu}(N) = \begin{cases} N-1 & \text{if } N \leq N_{n\_rmu\_max} \\ n_{rmu\_max} * \frac{N_{n\_rmu\_max}}{N} & \text{if } N_{n\_rmu\_max} < N \leq N_{n\_rmu\_min} \\ n_{rmu\_min} & \text{if } N > N_{n\_rmu\_min} \end{cases}$$

$$T_{mu}(N) = \begin{cases} max(T_{mu}, T_{m\_eval} * (N-1)) & \text{if } N \leq N_{n\_rmu\_max} \\ T_{m\_eval} * n_{rmu\_max} & \text{if } N_{n\_rmu\_max} < N \leq N_{n\_rmu\_min} \\ T_{m\_eval} * n_{rmu\_max} * \frac{N}{N_{n\_rrm\_min}} & \text{if } N > N_{n\_rmu\_min} \end{cases}$$

$$T_{tu}(N) = \begin{cases} T_{tu} & \text{if } N \leq N_{n\_rmu\_max} \\ T_{tu} * \frac{s_{tu}(N)}{s_{tu}(N_{n\_rmu\_max})}) & \text{if } N > N_{n\_rmu\_max} \end{cases}$$

This strategy essentially consists in a $n_{rmu}(N)$ adaptation. The $T_{mu}(N)$ period is adapted for the reasons explained in section 5.2.2. The $T_{tu}(N)$ period also needs an adaptation. Indeed, as N grows, more topology update messages are required (one per member), and this

is not caught with a $n_{rmu}(N)$ adaptation. Therefore the topology update period is slightly increased for groups larger than $N_{n\_rmu\_max}$ members.
An important question is: "what is an appropriate minimum number of records in a metric update message, n_{rmu_min}?". This parameter obviously depends on the maximum topology fanout (i.e. the maximum number of neighbors, which in turns depends on the topology creation algorithm). n_{rmu_min} must be at least equal to this maximum fanout to discover potential congestion problems on the topology. Some extra records with a random subset of the remaining members are then added to enable the RP to improve the topology.

5.2.2.4 Strategy 3

The third strategy is controlled by the following equation set:

$$n_{rmu}(N) = \begin{cases} N-1 & \text{if } N \leq N_{n\_rmu\_max} \\ n_{rmu\_max} & \text{if } N > N_{n\_rmu\_max} \end{cases}$$

$$T_{mu}(N) = \begin{cases} max(T_{mu}, T_{m\_eval} * (N-1)) & \text{if } N \leq N_{n\_rmu\_max} \\ T_{m\_eval} * n_{rmu\_max} * \frac{N}{N_{n\_rmu\_max}} & \text{if } N > N_{n\_rmu\_max} \end{cases}$$

$$T_{tu}(N) = \begin{cases} T_{tu} & \text{if } N \leq N_{n\_rmu\_max} \\ T_{tu} * \frac{s_{tu}(N)}{s_{tu}(N_{n\_rmu\_max}}) & \text{if } N > N_{n\_rmu\_max} \end{cases}$$

The main difference with strategy 2 concerns the $n_{rmu}(N)$ adaptation. This parameter remains constant above $N_{n\_rmu\_max}$ members, while the $T_{mu}(N)$ is immediately increased. It means that this strategy progressively reduces the number of metric update messages but each of them refreshes a higher number of communication costs with other members.

5.2.2.5 Strategy 4

Finally the fourth strategy is controlled by the following equation set:

$$n_{rmu}(N) = \begin{cases} N-1 & \text{if } N \leq N_{n\_rmu\_max} \\ n_{rmu\_max} + \frac{N-(n_{rmu\_max}+1)}{\beta} & \text{if } N > N_{n\_rmu\_max} \end{cases}$$

$$T_{mu}(N) = \begin{cases} max(T_{mu}, T_{meval} * (N-1)) & \text{if } N \leq N_{n\_rmu\_max} \\ T_{meval} * n_{rmu\_max} * \frac{s_{mu}(N)}{s_{mu}(N_{n\_rmu\_max})} & \text{if } N > N_{n\_rmu\_max} \end{cases}$$

$$T_{tu}(N) = \begin{cases} T_{tu} & \text{if } N \leq N_{n\_rmu\_max} \\ T_{tu} * \frac{s_{tu}(N)}{s_{tu}(N_{n\_rmu\_max}}) & \text{if } N > N_{n\_rmu\_max} \end{cases}$$

In this strategy, the number of records in each metric update message keeps increasing, even with large values of N, but in that case more slowly (we define a $1/\beta < 1$ slope). Consequently, the $T_{mu}(N)$ period increases much faster than with the previous solutions. This strategy goes further in the idea of strategy 3, i.e. have larger but less frequent metric update messages.

5.2.3 Metric Evaluation Overhead

HBM uses the `ping` command (by lack of a better tool) to evaluate the communication costs (i.e. RTT and losses). It keeps the default `ping` parameters (56 byte payload, one request

per second), but limits each evaluation to 6 `echo requests`. The corresponding bit rate overhead (including ICMP/IP), for N members, is:

$$ping\_rate(N) = N * \frac{6 * 2 * (20 + 8 + 56) * 8 * n_{rmu}(N)}{T_{mu}(N)} \quad (5.5)$$

where the $n_{rmu}(N)$ and $T_{mu}(N)$ parameters depend on the chosen strategy. Therefore $T_{m\_eval} \simeq 6$ seconds. Figure 5.6 shows the metric evaluation overhead (in packet/s) and proves that strategies 3 and 4, and to a lesser extent 2, also limit this overhead.

5.3 Experimental Evaluation

5.3.1 Experimental Setup

The HBM protocol and all the strategies defined previously have been implemented in a dedicated C++ Group Communication Service Library (GCSL) [30]. Experiments have been carried out with two hosts (PIII-1 GHz/Linux), one of them running the RP and the other one the N members. Since we only focus on the control traffic overhead, having all the members on the same host is not a problem. The parameters are set as follows: $T_{mu} = 60.5$ seconds, $T_{tu} = 120.5$ seconds, $T_{m\_eval} = 6$ seconds, $R_{data} = 128$ kbps, $R_{ctrl\_max} = 6.74$ kbps, i.e. $\alpha = 5\%$ of $(R_{ctrl}(N) + R_{data})$. The various control messages exchanged are sent as uncompressed, plain text. Each point in the figures is the average rate obtained after 30 minutes. In a second step we carried the same experiments with higher R_{data}/R_{ctrl_max} parameters in order to evaluate their impacts.

5.3.2 Experimental Results with $R_{ctrl\_max} = 6.74$ kbps

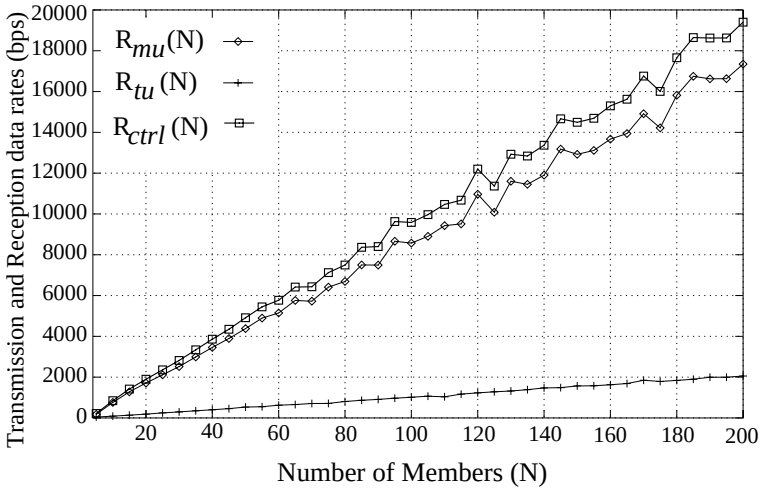


Figure 5.7. Experimental total control overhead for the strategy 1

Figure 5.7 shows the control overhead for strategy 1 and serves as a reference. It shows that $R_{ctrl}(N)$ increases linearly with N (and not in $O(N^2)$ because of the reason explained in section 5.2.2). Figure 5.8, 5.9, and 5.10 are for strategies 2, 3 and 4 respectively. $R_{mu}(N)$, $R_{tu}(N)$ and $R_{ctrl}(N)$ (the sum) increase progressively until $N = N_{n_{rrm-max}} = 79$ members are present. Above $N_{n_{rrm-max}}$, $R_{mu}(N)$, $R_{tu}(N)$ and $R_{ctrl}(N)$ are constant, around 6.75 kbps, 0.75 kbps, and 7.5 kbps respectively.

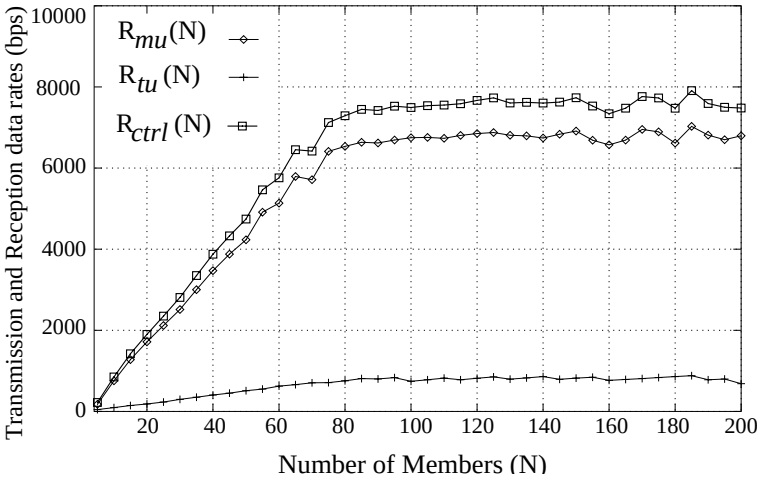


Figure 5.8. Experimental total control overhead for the strategy 2

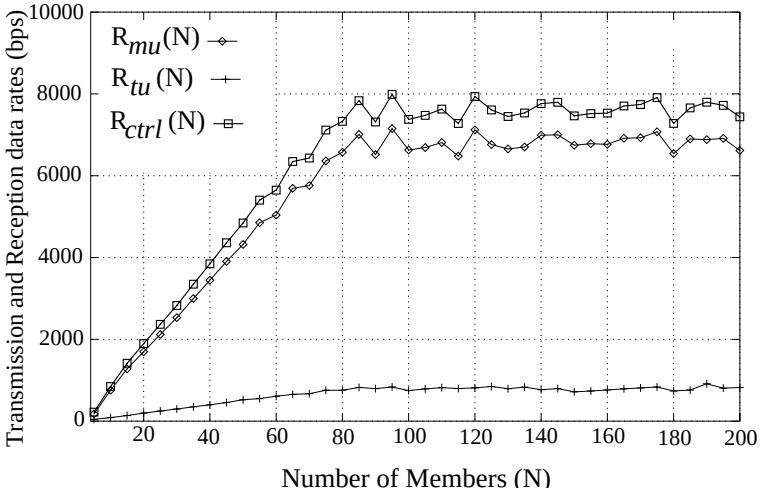


Figure 5.9. Experimental total control overhead for the strategy 3

We can say that *all strategies do achieve their goal of limiting the total control overhead rather well.* The total rate is in practice slightly larger than expected because of simplifications hypothesis taken during the theoretical analysis (e.g. the fixed size of the various messages fields of Figure 3.6).

5.3.3 Results with higher R_{ctrl_max} values

The previous results largely depend on the R_{data}/R_{ctrl_max} parameters, since they determine the maximum possible control overhead. We performed the same experiments with higher values and summarized the results for N=200 members in Figures 5.11 and 5.12. Non-surprisingly the HBM responsiveness is improved by increasing the bandwidth allocated to control messages up to a minimum of $T_{m\_eval} * (N-1) = 1194$ seconds time required to evaluate the metrics for T_{mu}). An exception is the T_{mu} period for strategy 2 which increases with R_{ctrl_max}. At the same time the number of records in a MU message, n_{rmu}, also increases more rapidly than with other strategies (from 38 records to 199 records, not shown) and MU messages become more representative.

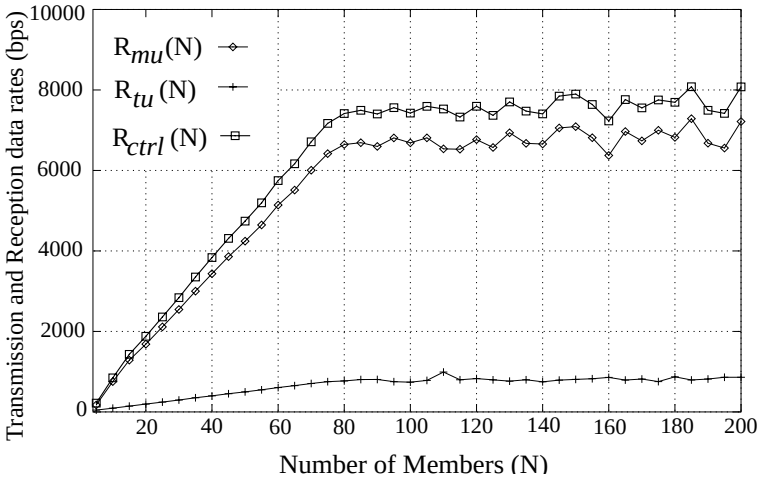


Figure 5.10. Experimental total control overhead for the strategy 4

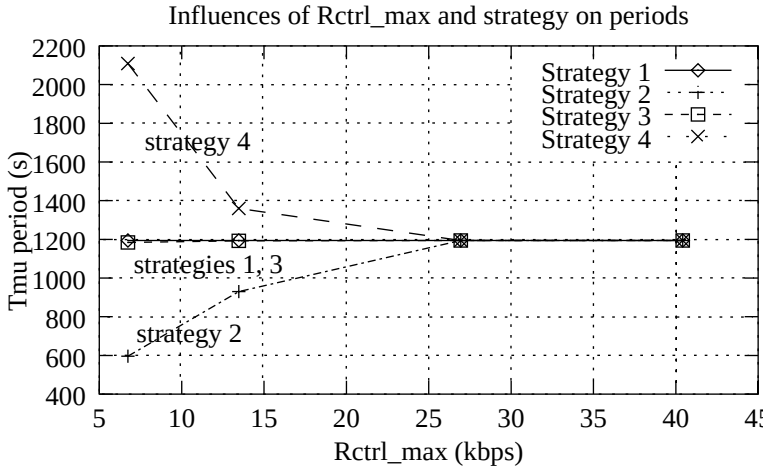


Figure 5.11. Influences of R_{ctrl_max} on the T_{mu} period for $N = 200$ members.

5.4 Discussion

If the three strategies do achieve their goal, they do it differently. The question is thus: "what is the best solution to achieve that goal"?
With strategy 2 the $T_{mu}(N)$ period slowly increases. This is an advantage since the metrics with the direct neighbors are frequently updated as shown in the Figure 5.1, which enables a fast discovery of congestion problems. But they are also less representative as shown in the Figure 5.4, which reduces the probability of finding better neighbors in the overlay topology. Another drawback is the higher metric evaluation overhead (section 5.2.3).
In our opinion *Strategy 3 offers a better compromise*. The metric evaluation overhead is reduced compared to strategy 2 and many members are probed during the metric update processes as shown in the Figure 5.3, thereby offering a better opportunity to improve the overlay topology. The price to pay is a lower metric update frequency.
Finally strategy 4 goes to far in this direction, and the $T_{mu}(N)$ period for very large values of N is by far too high, leading to a bad adaptability.
Note that results are largely impacted by the various protocol parameters (parameter $max(T_{mu}, T_{m\_eval} * (N-1))$ in the $T_{mu}(N)$ equations). For instance this is the reason why the T_{mu}

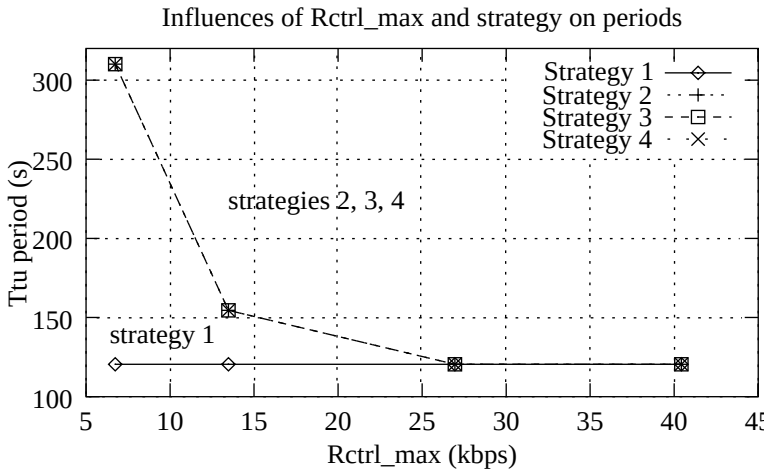


Figure 5.12. Influences of R_{ctrl_max} on the T_{tu} period for $N = 200$ members.

period is only constant for $N << (\frac{T_{mu}}{T_{m\_eval}}) + 1$ and keeps increasing afterward (at least up to $N_{n\_rmu\_max}$, and sometimes after this value depending on the strategy chosen). Having fast (i.e. not in $O(N)$) yet reliable metric evaluation techniques (section 4.2.1) between the various members would largely modify our results.

5.5 Conclusion

In this chapter we showed that the scalability of HBM can easily be improved by limiting both the number and the length of control messages that are exchanged between the RP and members. More precisely this can be controlled by adjusting: the number of records in a metric update message, the metric update generation period, and the topology update period. Finally an appropriate solution, strategy 3, that in our opinion offers a good compromise between the various aspects, has been identified. This chapter also highlights the practical limitations raised by the metric evaluation process which largely impacts our results.

Chapter 6

Improving the Robustness in front of Node Failures with Redundant Virtual Links (RVL)

Contents

In this chapter, we show that adding several Redundant Virtual Links (RVL) to the overlay topology helps to reduce the probability of overlay topology partition. The questions are:

- How many RVL links should be added to the overlay topology?,
- Between which nodes should RVL links be added?,
- Should it be source dependent or not?

In this chapter, we answer these questions by introducing and comparing five different strategies.

6.1 Introduction

In a tree-based protocol, because a tree overlay topology is an acyclic graph, if any non-leaf node leaves the multicast group or crashes, the tree is partitioned. Tree-based approaches thus require partition detection and recovery mechanisms.
Many application level multicast proposals, comparable to HBM, merely content themselves with a fast detection and repair mechanism, for instance to identify partition problems and take counter measures [52]. In our opinion this reactive approach is definitively insufficient. For instance, some applications may require that partitions be avoided altogether (e.g. co-operative work or high quality multimedia-on-demand session) and reactive solutions are not acceptable.
An approach that shares some similarities with our work is the Probabilistic Resilient Multicast (PRM) scheme [13]. Here a subset of the overlay tree nodes *randomly* "jump" data to other nodes of the tree, thereby creating redundant paths. Data coming from these random jumps is then flooded on sub-trees, unless it has already been received. Yet this solution is limited by the random nature of the jumping process which is the only feasible approach when no node has a consistent view of the topology (unlike HBM). This is the main difference with our own solution.
The TMesh proposal [100] also deliberately adds redundant links (called shortcuts) but with the goal of reducing latencies between members. A side effect is an increased robustness since shortcuts also provide redundant connections between members. But here also, since no single node has a consistent view of the topology, shortcuts are added in a random way (unlike HBM).
In this chapter we deliberately follow a voluntary approach by adding explicit redundancy in the overlay topology as well as a learning mechanism whereby less reliable hosts are identified and the topology created by taking it into account,

6.2 Adding Redundant Virtual Links (RVL)

6.2.1 General Ideas

In order to reduce the probability of overlay topology partition in front of one or more member failures, we add Redundant Virtual Links (or RVL) [86] to the overlay topology created by HBM. Since the RP has a full knowledge of group members and creates/manages this topology, it can easily add a certain number of RVLs. The RVLs are *strategically placed* so as to provide some level of robustness guaranties (e.g. a resilience to any single node failure, or to any two simultaneous failures, etc.). In this work we only assume a "robustness to a single transit node failure" (and evaluate experimentally the probabilistic robustness to several simultaneous node failures). This solution is not source dependent and therefore the robustness is the same no matter how many and where sources are.
To provide these guaranties we introduce and compare five different flavors, that differ on the way RVLs are added. For instance with the first flavor no difference is made between leaf and transit nodes, whereas the remaining four flavors clearly limit the number of RVLs that can be attached to any leaf node. This distinction makes sense since nodes with limited processing or communication power are always moved towards the leaves of the overlay shared tree topology.

Strategy I: any number of RVLs can be attached to any node. No difference is made between leaf and transit nodes.

Strategy II: there is no limit on the number of RVLs attached to a transit node, but at most one RVL can be attached to a leaf. A RVL can be attached to any kind of node.

Strategy III: any number of RVLs can be attached to any node. A RVL cannot be attached to two leaves. Only {leaf node; transit node} and {transit node; transit node} RVLs are possible.

Strategy IV: there is no limit on the number of RVLs attached to a transit node, but no RVL can be attached to a leaf node. Only {transit node; transit node} RVLs are possible.

Strategy V: there is no limit on the number of RVLs attached to a transit node, but at most one RVL can be attached to a leaf. Only {leaf node; transit node} RVLs are possible.

These strategies follow a recursive approach. First find the two farthest nodes in the set, add a RVL between them, and split the set into two sub-groups, depending on their closeness to the two elected nodes. For each sub-group, do the same process, recursively, until the sub-group contains at most two nodes. The way the two farthest nodes are chosen depends on the strategy flavor mentioned above. The detailed strategy is the following:

```
// initialization
AddRedundantLinks(virtual_topo, all_nodes_in_virtual_topo);


// recursive solver for strategies I, II, III, and IV
AddRedundantLinks (topology, G)
{
    if (number of nodes in G <= 2)
        return;
    find the two farthest nodes (N0, N1) in G, such that:
        Strategy I  : no condition
        Strategy II : if N0 is a leaf it must not already take part in a RVL,
                        and the same must be true for N1
        Strategy III: at least one of N0 and N1 must be a transit node
        Strategy IV : both N0 and N1 must be transit nodes
    add the RVL {N0; N1};
    split G into two sub-groups, SG0 and SG1, such that:
        SG0 includes N0 and all nodes of G that are closer to N0 than N1
            in the physical topology;
        SG1 includes N1 and all nodes of G that are closer to N1 than N0
            in the physical topology;
    if (number of nodes in SG0 > 2)
        AddRedundantLinks(topology, SG0);  // continue with SG0
    if (number of nodes in SG1 > 2)
        AddRedundantLinks(topology, SG1);  // continue with SG1
}


// recursive solver for strategy V
AddRedundantLinks (topology, G)
{
    split G into two sub-groups, SG0 and SG1, such that:
        SG0 includes all the leaf nodes of G, and
        SG1 includes all the transit nodes of G;
```

```
    for each leaf node N0 of SG0 {
        find the farthest transit node N1 of SG1;
        add the RVL {N0; N1};
    }
}
```

6.2.2 Examples

Let's consider a group of 10 members, and let's imagine the overlay created is given in Figure 6.1-a. Table 6.1 gives set of metrics between nodes in the physical topology.

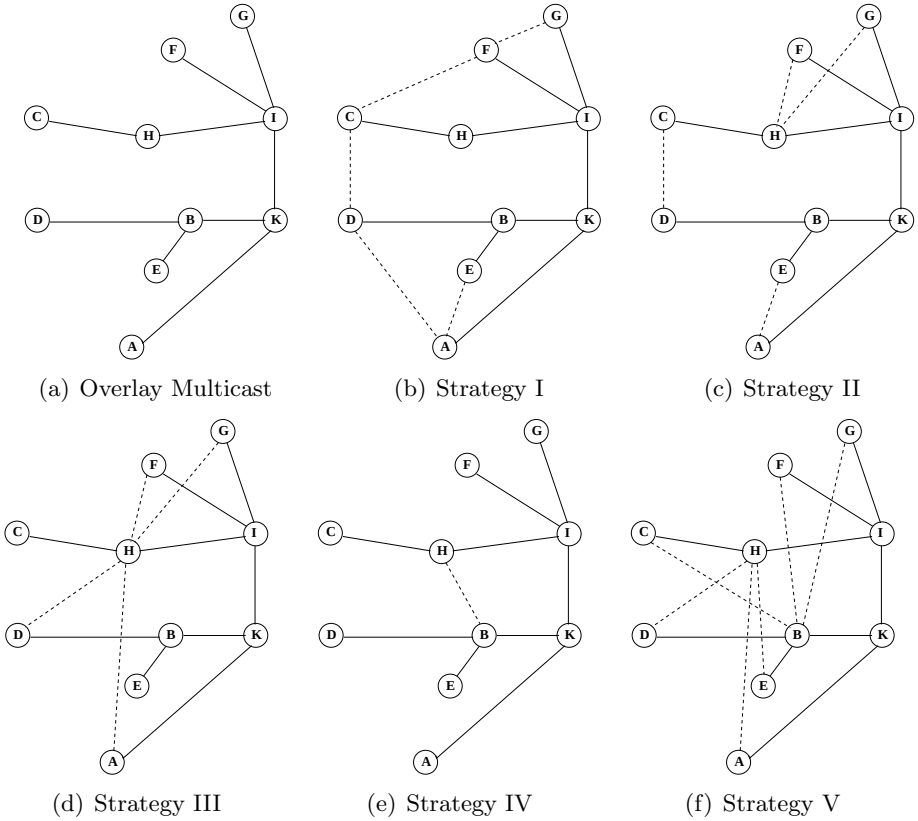


Figure 6.1. RVL addition (dashed lines), an example.

Table 6.1. The physical unicast delay (RTT/2)

	A	B	C	D	E	F	G	H	I	K
A	-	152.94	190.75	186.60	157.57	174.90	163.36	164.71	159.10	124.79
B	152.94	-	208.21	172.05	51.01	192.34	180.81	182.15	176.55	142.24
C	190.75	208.21	-	241.86	212.83	182.16	170.62	65.97	166.36	180.05
D	186.60	172.05	241.86	-	176.67	226.00	214.47	215.81	210.21	175.89
E	157.57	51.01	212.83	176.67	-	196.97	185.43	186.78	181.17	146.86
F	174.90	192.34	182.16	226.00	196.97	-	154.76	156.11	150.50	164.19
G	163.36	180.81	170.62	214.47	185.43	154.76	-	144.57	138.97	152.66
H	164.71	182.15	65.97	215.81	186.78	156.11	144.57	-	140.31	154.00
I	159.10	176.55	166.36	210.21	181.17	150.50	138.97	140.31	-	148.39
K	124.79	142.24	180.05	175.89	146.86	164.19	152.66	154.00	148.39	-

Strategy I: Let's assume the two farthest nodes are D and C. The D⇌C RVL is first created. Sub-group SG0 for D is {D, B, K, A, and E} and sub-group SG1 for C is {C, I, F, H, and G}. For SG0, the two farthest nodes are D and A, and the RVL D⇌A is added. The sub-group SG00 for D contains only node {D}, and the process finishes. The sub-group SG01 for node A contains {A, B, K, and E}. The RVL addition strategy continues recursively for sub-groups SG01 and SG1, and we finally get the topology of Figure 6.1-b.
Strategy II: The beginning is similar to strategy I, and here also the D⇌C RVL is first created. But the sub-group SG0 for D is now {B, K, A, and E} and the sub-group SG1 for C is {I, F, H, and G}. Nodes D and C are not added to SG0 and SG1 because D and C are leaves and both of them already take part in a RVL. The process then continues recursively and leads to the topology of Figure 6.1-c.
Strategy III: Because two leaf nodes cannot be part of the same RVL (i.e. D⇌C is not permitted), the farthest two nodes selected are now D and H, and the D⇌H RVL is created. The subgroup (sub1) for D is {B and E} and the subgroup (sub2) for H is {H, I, F, K, G, A, and C}. So the process continues with sub-group sub2, and we finally get the topology of Figure 6.1-d.
Strategy IV: Since RVLs can only be created between two transit nodes, we only consider nodes {B, H, I, and K}. The farthest nodes are B and H, and the RVL B⇌H is added. The sub-group for B is {B and K}, and the sub-group for H is {H, and I}. Therefore, the process is finished. The resulting topology is given in Figure 6.1-e.
Strategy V: For each leaf node, we first find the transit node which is the farthest, and we create the corresponding RVL. In this example we create RVLs D⇌H, F⇌B, G⇌B, A⇌H, C⇌B, and E⇌H, as shown in Figure 6.1-f.
From this example we see that the number and the location of RVLs largely differ for each strategy considered. For instance with strategy I we note that (1) the vast majority of RVLs are among leaf nodes, and (2) some leaf nodes have several RVL attached. Therefore, strategy I is devoted to cases where all group members have similar processing and communication capabilities, which is very restrictive. On the opposite strategy IV leads to the creation of a single RVL, and leaf nodes (who can be lightweight nodes, since the node features can be considered during the topology creation process) are never concerned by RVL.

6.2.3 Performance Evaluation Parameters

Let N be the total number of nodes. The following parameters are considered:

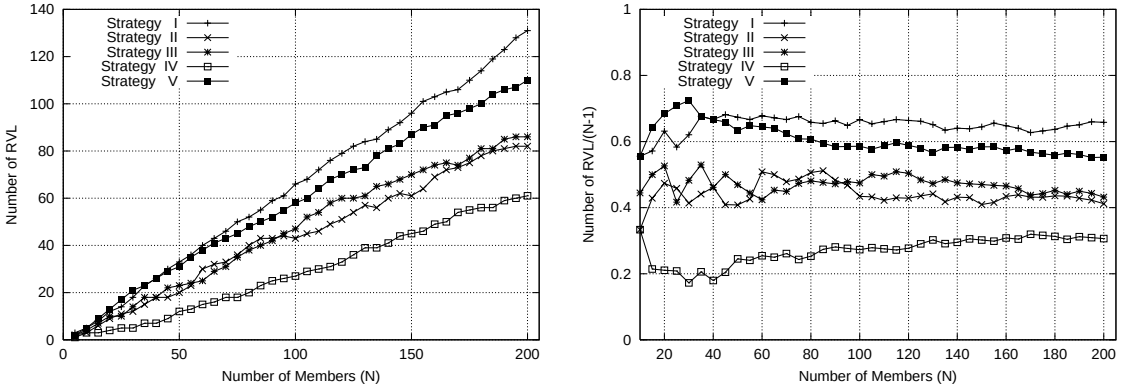
- the number of RVLs: N_{RVL}
- the ratio of the number of RVLs to the initial number of (non-RVL) links in the overlay: $R_{RVL} = \frac{N_{RVL}}{N-1}$. Note that with N nodes, without RVLs, there are always $N-1$ links in the shared tree overlay.
- the number of connected nodes after i node failures, respectively without and with RVLs: $R_{conn-without}(i)$ and $R_{conn-with}(i)$.
- the ratio of the number of connected nodes after i node failures to the total number of nodes, without and with RVLs: $R_{conn-without/with}(i) = \frac{N_{conn-without/with}(i)}{N}$. This ratio is an average over all possible sources (i.e. each node can be a source in a shared tree). Ideally i node failures should leave $N-i$ connected nodes, so the maximum ratio is: $R_{conn-ideal}(i) = \frac{N-i}{N}$.
- the relative increase (or gain) in the number of connected nodes by adding RVLs, in front of i failures: $G_{conn}(i) = \frac{N_{conn-with}(i) - N_{conn-without}(i)}{N_{conn-without}(i)}$

- link stress: this is the number of identical copies of a packet carried by a physical link. This stress is evaluated with and without RVLs.

6.3 Performance Evaluation Results

6.3.1 Experimental Conditions

The HBM and all the strategies defined previously have been implemented in a dedicated C++ Group Communication Service Library Library. The experiments reported here are simulations based on a large interconnection transit-stub network, composed of 600 core routers, and generated by the Georgia Tech Model (GT-ITM) [102]. Some of these routers are interconnection routers, others, at the leaf of the topology, are access routers connecting the client sites. We then choose N sites randomly among the 243 possible leaves and compare each strategy.

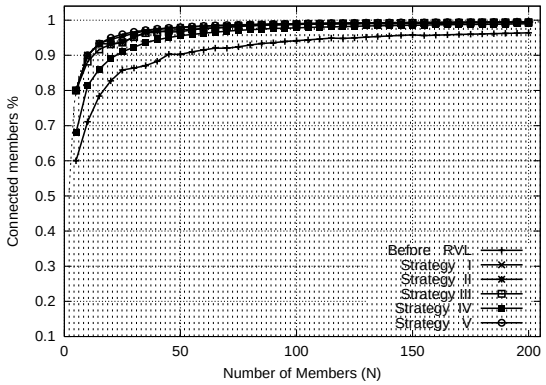


(a) Number of RVLs (b) R_{RVL} ratio

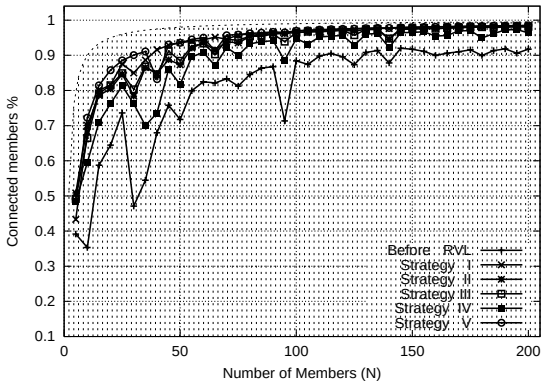
Figure 6.2. Addition of RVLs.

6.3.2 Results and discussion

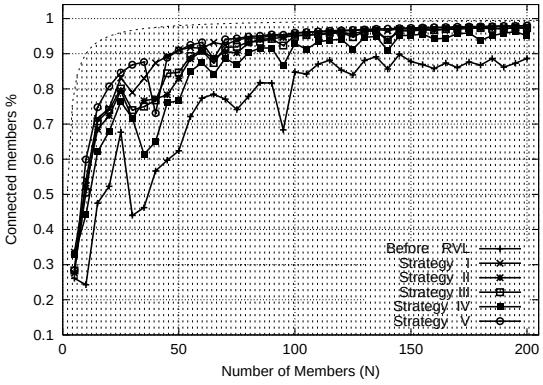
Figure 6.2-a depicts the number of RVLs versus the number of nodes N. We note that the number of RVLs increases with N for all strategies, but with a different slope. Figure 6.2-b depicts the associated ratio, R_{RVL}. We see that strategy IV adds the smallest number of RVLs, whereas strategy I adds the highest number of RVLs.
Figures 6.3-a/b/c depict the ratio of connected nodes when respectively one, two and three nodes fail. The upper edge of the dashed area represents the optimal ratio, $R_{conn-ideal}(i) = \frac{N-i}{N}$, where i is the number of failures. If all strategies that add RVLs improve the connectivity after a certain number of failures, we see that differences exist. Without any RVL, the ratio of connected nodes amounts to 60% for 5 nodes and 96% for 200 nodes after a single node failure. With strategy IV, the associated ratios are respectively 68% and 99%. With strategies I, II, III and V, the associated ratios are now 80% and 99% that must be compared to the 80% and 99.5% ideal values. We also note that for both two and three node failures, all strategies bring some benefits compared to the initial topology, even if the results are farther than the ideal.
Figures 6.4-a/b/c show the relative increase in the number of connected nodes, $G_{conn}(i)$. We note (1) that this gain decreases when N increases (larger groups), and (2) that this gain increases with the number of members failures.



(a) Single node failure



(b) Two node failure



(c) Three node failure

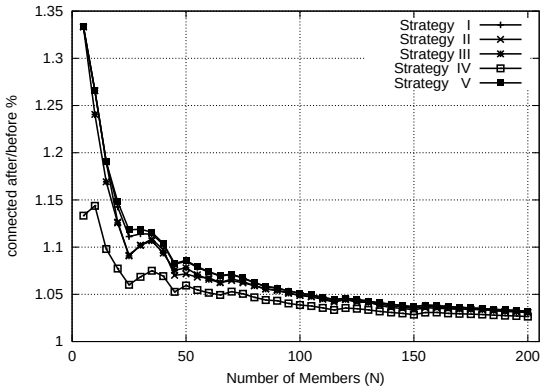
Figure 6.3. Ratio of connected nodes after 1, 2 or 3 failures, according to the RVL addition strategy.

Finally figure 6.5 depicts the stress before and after adding RVLs. As expected, the stress increases when RVLs are added. Yet strategy IV has the smallest stress (because it has the smallest number of RVL) whereas strategy V adds a prohibitive amount of traffic. (it is the strategy which adds the highest number of RVLs).
To conclude we can say that *strategy IV offers a good balance between the robustness in front of non-graceful node failures, and additional traffic within the network. Besides the additional traffic is managed by transit nodes, never by leaves who can have lower processing or networking capabilities since this feature is taken into account by the RP during the topology creation process.*

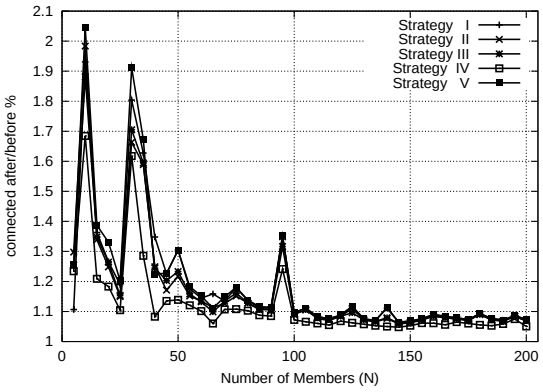
6.4 Loop Suppression Strategy when partial robustness is sufficient

The additional traffic generated by RVL links is sometimes prohibitive. In such a case we can choose to follow an intermediate approach, where RVL links are added but traffic forwarding on them is regularly suspended. The stress is therefore reduced, but the price to pay is a possibility of packet losses when a failure occurs while traffic is suspended. The detailed strategy is the following:

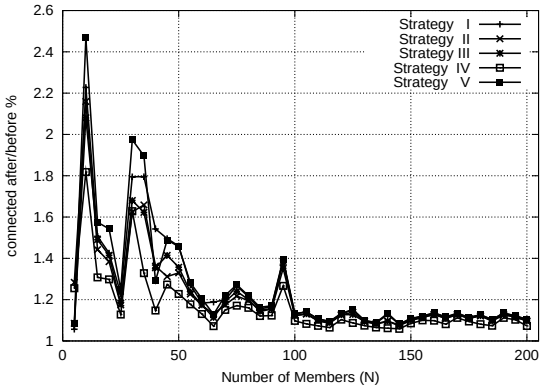
```
if (node N receives traffic both on the overlay multicast link and RVL)
    send a SUSPEND message on the RVL
```



(a) Single node failure



(b) Two node failure



(c) Three node failure

Figure 6.4. Relative increase in the number of connected nodes after 1, 2 or 3 failures, according to the RVL addition strategy.

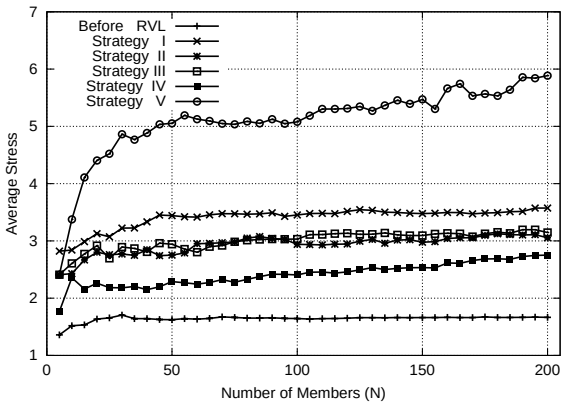
```
    // on receiving a SUSPEND, the peer stops forwarding packets on the RVL
    // during one second

if (node N receives traffic on the RVL but not overlay multicast link)
    // there is a problem, yet N still receives new traffic and can
    // forward them on the OM
    wait some time and send a failure report to RP if situation persists
```

6.5 Use with Other Application-Level Multicast Proposals

Our proposal can easily be applied to any centralized application-level multicast proposal, for instance ALMI [77, 92].

But this is not to case with distributed proposals because: (1) no node knows the whole overlay topology, and (2) the overlay topology management is a fully distributed process.

This is the reason why proposals the TMesh, PRM add redundant links randomly. The results are expected to be lower than ours (where we determine the *ideal* placement of these RVL links) both in terms of robustness and additional traffic, but the exact performance comparison is left to future works.



(a) Stress

Figure 6.5. stress

6.6 Conclusion

In this chapter we discussed the robustness in case of the partition of the overlay topology when a single node failures, and also two nodes and three nodes failure. We found that Adding a certain number of redundant virtual links is suitable for reducing the probability of overlay topology partition. We introduced five strategies for adding RVL links. We noted that RVL links has not to be both among two non core nodes and between non core node and core node to avoid the bottleneck problem. strategy IV adds RVL links among core nodes only. So in our opinion, strategy IV is the best. We note that when two or three nodes are failed, the probability of overlay topology partition is reduced too.

Chapter 7

Reducing the Packet Losses Experienced During a Topology Update

Contents

In this chapter we study the problem of packet losses generated by transient incoherences in the overlay topology. Solving this problem is another complementary way of improving the global robustness of HBM. More precisely, we focus on transient incoherences that are the result of a topology update, which can be triggered either (1) by a partition in the overlay topology, for instance after the failure of a host acting as a transit node in the overlay, or (2) by a periodic topology update in order to improve the overlay.

7.1 Introduction

Because of the dynamic nature of the group, because the partition recovery strategy may lead to sub-optimal topologies, because the networking conditions keep changing, it can be required to update the current overlay topology. This update decision can be periodic, or done when we realize the congruence between the physical topology and the overlay is too

bad, or when the number of partial modifications in the overlay exceeds a given threshold. The update decision can be taken locally by a member (distributed process), or globally by a super-member ([77, 86] define a dedicated Rendez-Vous Point host that is responsible of topology calculation and distribution).
The problem is that the topology update process is usually not synchronous, and at some point in time a subset of the members will be informed of (and probably will use) the new topology, while others will still use the previous old one. This routing incoherency can easily lead packets in transit to be lost. Our goal is to discuss and compare several solutions meant to largely reduce the probability of packet losses when such a topology update occurs. We believe this is of high practical importance while this aspect is usually overlooked. In this work we do not make any assumption on the reason which can trigger the topology update. Also, we do not consider here packet losses caused by other problems like node failures that are addressed by other mechanisms (see chapter 6).

7.2 Packet Loss Reduction (PLR) Techniques

7.2.1 Parameters Affecting Losses

Several parameters affect the number of packets that can be lost during a topology update:

- the importance of changes: more precisely this is the number of links of the overlay that are modified and the number of nodes that are concerned by these modifications;
- the time required to inform all nodes concerned about the new overlay topology: this parameter defines the period during which transient routing incoherences can occur;
- the number of packets in transit during this instability period: these are the packets that are potentially affected, i.e. that may be either lost (partition) or duplicated (loop).

Other parameters are not considered in this chapter: a node failure in chapter 6 and the congestion either in the Internet or in the connection to the Internet are not considered in case of packet losses.
Therefore routing problems will be all the more acute as the transmission rate is high, the changes numerous, and the topology update process long.
In the following, we assume that the RP who decides that the topology must be updated and calculates the new topology informs *sequentially* each member concerned. It is the role of the RP to decides that the topology must be updated, to calculate the new topology, and to inform each node concerned. If n nodes out of the N present in the group are concerned, then the instability period lasts approximately:

$$T\_instability(n) \approx \frac{n}{2} * call\_to\_send + Max_{i \in 1..n}(delay\ to\ node\ i)$$

Indeed, even if the `send()` TCP socket syscall is called sequentially for all n nodes (hence the first part of the formula), transmissions take place in parallel, and the topology update message is available at the receiving application before the TCP segment has been acknowledged (hence the one way delay in the formula).

7.2.2 Possible PLR strategies

The PLR strategies assume that each topology be identified by a dedicated *globally unique and monotonically increasing Topology Sequence Number, or TSN*. This TSN is incremented

each time the topology is updated. The globally unicity constraint requires that a dedicated node manages this identifier which is not an issue with a centralized approach like the HBM. *This TSN is present in all packets sent in the overlay topology, and each member remembers the current TSN in use along with its list of neighbors.* Of course, because of the instability period ($T\_instability(n)$), different members can have a different view of the current TSN in use. In this work we analyze the following PLR strategies:

Strategy 1: Each time a member needs to update its topology (e.g. a member received a control message that informs him of its neighbors in the new topology) he drops the current topology information (list of neighbors) and registers the new topology one. Each time a transit node receives a packet having a TSN different from its own current TSN, three possibilities exist:

1(a): this packet is not forwarded. This default behavior is used as a reference in our experiments. It is a *conservative approach* that tries to limit the risk of creating loops at the cost of a higher packet loss rate.

1(b): if this packet has never been received before, the transit node forwards it over the current topology (except to the node from which it was received). This behavior is the opposite of approach (a) and tries to reduce the packet loss rate but increases the risk of creating loops.

1(c): if this packet has been received on a link that belongs to the current topology, this packet is forwarded, otherwise it is dropped. This behavior tries to intelligently distinguish between situations where there is a little risk of creating loops (a packet is received on a link of the new topology but with a wrong TSN) and other situations where this risk is high, and the forwarding decision is only taken if the risk is considered limited.

Strategy 2: Each member keeps information for two topologies: the current one and the previous one. Each time a transit node receives a new packet, this latter is forwarded on the previous or current topology if its TSN is equal respectively to the previous or current TSN known by the node, and is dropped otherwise.

In all cases, each members keeps track of what packets have been received and drops duplicated packets in case a routing loop has been created.

7.3 Experimental Evaluations

In this section we analyze and compare the four PLR proposals in order to find the one which has the best balance between improved robustness and packet duplication avoidance.

7.3.1 Experimental Setup

The evaluation of the proposals is based on simulations carried out with a C++ implementation of the HBM. The importance of topology changes (section 7.2.1) is controlled by the number of communication metrics that have been changed (in this chapter we give them new random values) since these changes trigger topology changes. In the tests we use 25%, 50%, 75% and 100% metric changes. Figure 7.1 shows that the percentage of links of the overlay topology that are changed (i.e. the number of links changed divided by (N-1), the total number of links) is in line with the percentage of metrics changed.

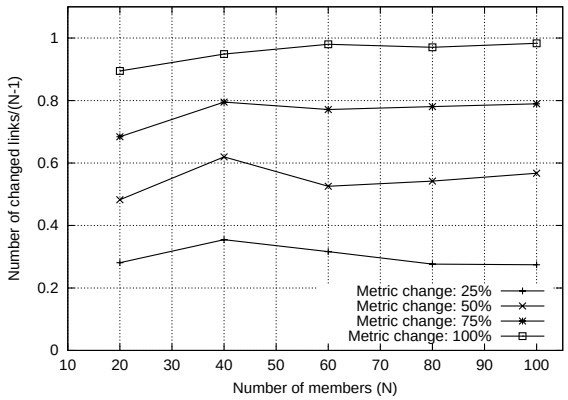
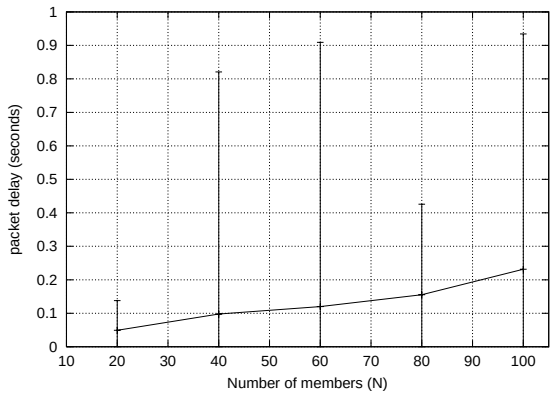


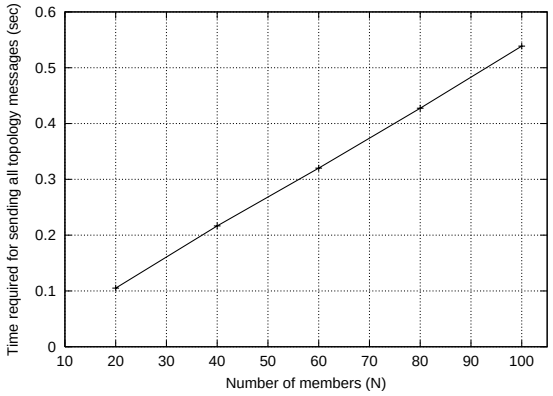
Figure 7.1. Percentage of links changed according to the percentage of metrics changed.

In these experiments, we assume that topology updates are sufficiently spread over the time and that all group members have the opportunity to update their topology information before a new topology is deployed. This is a reasonable assumption because of the intrinsic cost implied in a general topology update process (see [31] for an overview of control overhead). Experiments are made on two powerful hosts (PIV-2.4 GHz/Linux), one of them running the RP and the other one the N members. We do not simulate propagation delays between the various members and the RP (we merely run N instances of the programs), but we check that:

- the effective communication delays are in line with typical values: figure 7.2(a) illustrates the minimum/average/maximum communication delay between a traffic source and all the members with a data rate equal to 512 kbps. It shows that end-to-end delays range from 1 ms to 1 second.
- the sent topology period: Figure 7.2-b shows a linearly increasing sent topology period from 100 ms to 540 ms depending on the group size.



(a) Min/avg/max delay between a source and all members (512 kbps).



(b) Sent topology period.

Figure 7.2. Communication delay and sent topology period.

Tests are performed with a fixed packet size equal to 1024 bytes, and two transmission rates: 128 kbps and 512 kbps. In terms of packets per second (which determines the number of

packets in transit during the instability period), it corresponds to 15.6 and 78.1 packets/s. Each point in the figures is an average over 5 topology changes, each of them being sufficiently spaced over the time for the members to be synchronized in the meantime. Packet losses values are the average number of losses experienced by a member (i.e. the total number of packet losses divided by the number of members).
The quantitative results obtained are for sure highly dependent on the parameters chosen (section 7.2.1). We do not claim to have fully analyzed the problem space (in particular transmission delays between the various group members could be more realistically simulated using topology models). But we believe that the qualitative results obtained are realistic, which was our main goal.

7.3.2 Results with a Data Rate Equal to 512 Kbps

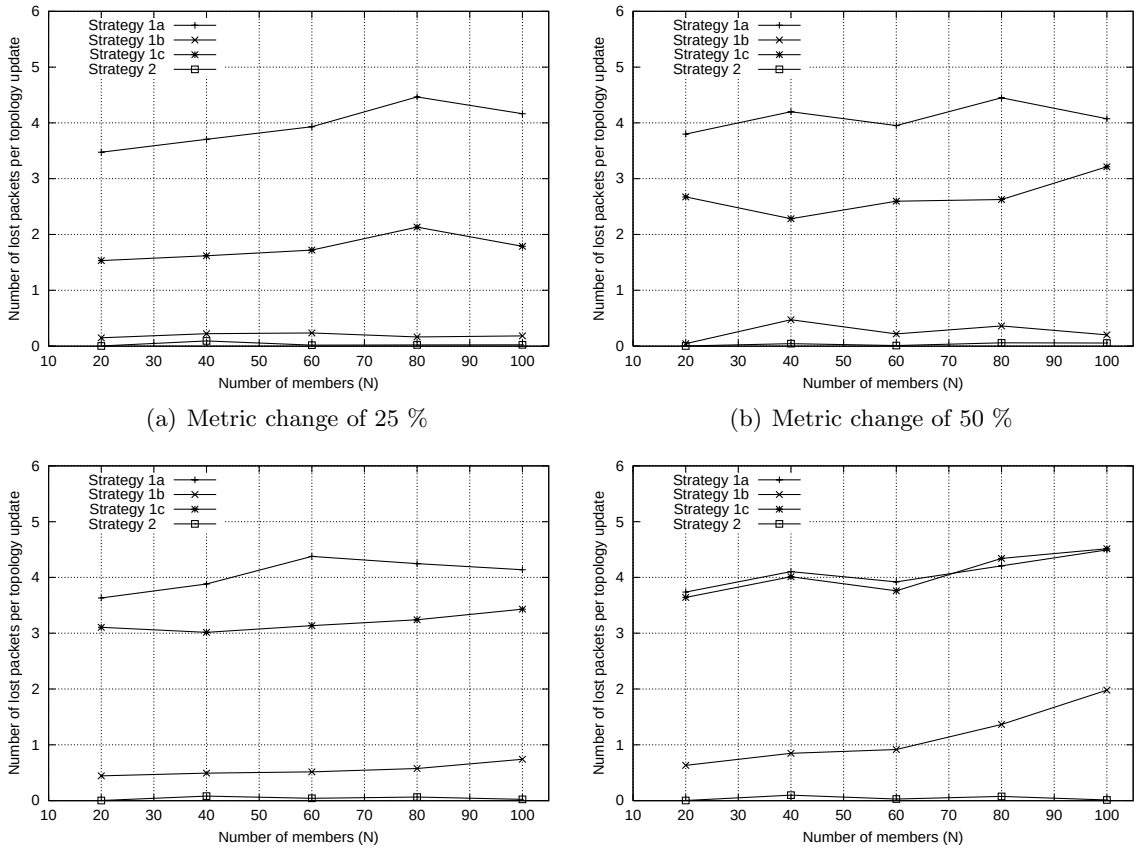
In this section we compare the four strategies when the data rate is equal to 512 kbps. Figures 7.3 show the average number of packet lost. We see that strategy 1(a) is the one which performs the worst in all cases, with around 4 packet losses. It does not depend on how much of the topology changed because a transit node that receives a packet with a bad TSN drops this latter no matter whether its own neighbors actually changed or not. A single link change in the overlay trigger a TSN modification and all packets in transit with a wrong TSN are dropped. *strategy 1(a), far too conservative, is definitely not appropriate.*
Strategy 1(b) tries to improve robustness which is visible in the figures. In particular, if the whole topology is changed, it leads to an average number of losses of approximately 1 packet only. The price to pay is non-surprisingly a very high packet duplication ratio compared to other strategies. This is visible in Figure 7.4 which exhibits between 26% to 37% of duplicated packets with strategy 1(b), whereas all other strategies never exceed 4%.
Strategy 1(c) yields a better robustness than strategy 1(a) when the topology changes are small. Yet both strategies tend to be equivalent in case of important topology modifications (e.g. there is no difference if the topology is completely changed). Strategy 1(c) has a low probability of packet duplications.
Strategy 2 has excellent performances, both in terms of robustness and packet duplication. Moreover these performances depend neither on the group size nor on the importance of the topology update. It largely outperforms all strategies where nodes remember a single topology at a time. Therefore we can conclude that strategy 2 is probably the best solution and should be used systematically.

7.3.3 Results with Different Data Rates

In this section we quickly show the influence of the packet bit rate on the final loss rate experienced. Since losses are extremely low with strategy 2, we show the impacts on strategy 1(c) which is the best alternative. Figures 7.5-a and 7.5-b show the high importance of this factor. Technical problems prevented us to test the influence of the instability period duration on the packet loss experienced. Nonetheless, since the key aspect is the number of packets in transit during the instability period, the same behavior could be observed when varying the duration of this period. Both parameters are therefore dual.

7.4 Use with other Application-Level Multicast Proposals

In this section we explain how the PLR schemes can be applied to various application-level multicast proposals. As described in chapter 2, the application-level proposals are classified into two classes: centralized and distributed.



(a) Metric change of 25 %

(b) Metric change of 50 %

(c) Metric change of 75 %

(d) Metric change of 100 %

Figure 7.3. Number of lost packets per topology update with a data rate of 512 kbps.

7.4.1 Centralized Proposals

In these proposals, a single node controls all group membership and topology management aspects. HBM [86] calls this node RP while ALMI [77] calls it controller. The two proposals are otherwise similar. In both cases the central node can assign a monotonically increasing TSN to each newly generated multicast topology, and informs group members of the new TSN and topology. In ALMI, each node already maintains a small cache of recent multicast tree incarnations. When receiving a packet with a known tree version (cached), the packet is forwarded on the corresponding tree, and if the tree version is too old, the packet is discarded. When the tree version is newer, the member re-registers itself with the controller to receive the new tree information. So ALMI already includes a scheme similar to our PLR proposal, but does not evaluate its benefits, nor compares it to other solutions, nor specify how many tree incarnations each member should keep.

Our PLR proposals fit well with centralized solutions. Note that the goal is to apply the PLR scheme during large topology updates, not during the graft process that enables a *rapid recovery* of partition problems after a transit node failure or departure (which can be frequently needed depending on the group size). This is not an issue since a limited topology modification can be managed very rapidly (few nodes are concerned).

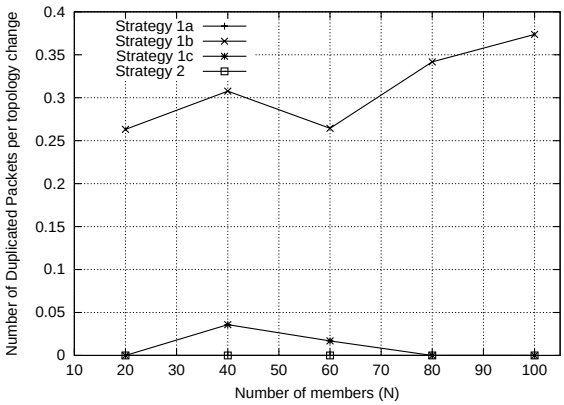
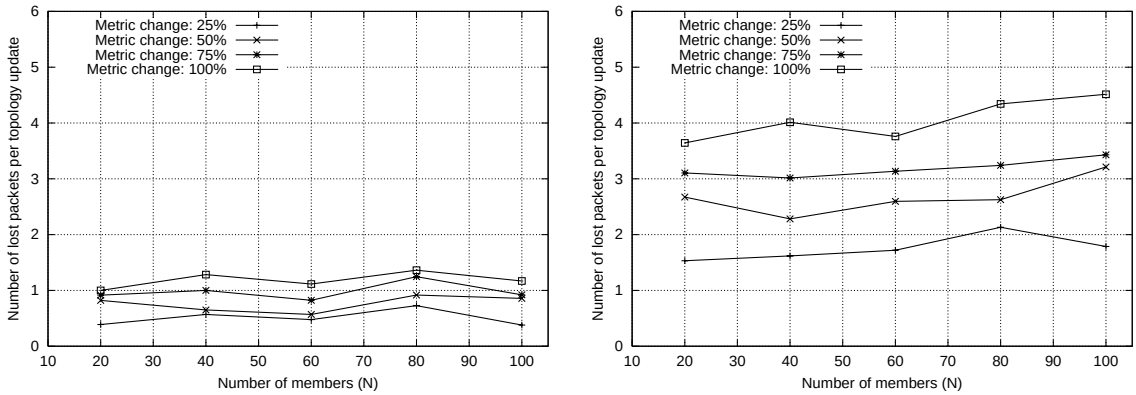


Figure 7.4. Number of duplicated packets per topology update (512 kbps, 100% metrics changed).



(a) Data rate 128 Kbps, strategy 1c

(b) Data rate 512 Kbps, strategy 1c

Figure 7.5. Number of lost packets per topology change with strategy 1(c).

7.4.2 Distributed Proposals

Our PLR scheme cannot be easily used with this class of approach because: (1) no node can reliably say when an instability period starts and finishes; and (2) topology management is a fully distributed process that can be initiated at any time, by any member, and even simultaneously. This situation is too different from the scenario for which PLR has been designed.

7.5 Conclusion

In this chapter we discussed the problem of packet losses during a topology update. Since the topology update is usually not synchronous, at some point in time a subset of the members will be informed of (and probably will use) the new topology, while others will still use the previous old one. We introduced and compared four proposals, that differ on the number of topologies that each member needs to remember. We have shown that the best solution is undoubtedly the "two topologies" approach (strategy 2), which yields a high robustness and does not incur packet duplications, while being simple to implement.

Chapter 8

Using HBM to Build a Secure but Efficient Group Communication Service

Contents

In this chapter we show how to build a fully secure but efficient group communication service between several sites. This service is built on top of a VPN environment where IPSec tunnels are created, on-demand, between the various sites that need to communicate. This chapter is a follow-up of previous work on group communications in a VPN environment and on application-level multicast. We show that these proposals naturally fit with one-another and lead to the concept of Virtual Private Routed Network, or VPRN [2, 3]. This concept enables us to largely improve the data distribution efficiency, and in particular reduces the physical link stress. This section can therefore be regarded as an additional field of application for HBM.

8.1 Offering a VPN Based Secure Group Communication Service

Before describing our solution, we first introduce the VPN specificities, how to build a group communication service on top of it, and how this approach departs from the work carried out in related IETF groups.

8.1.1 Definition of an IP VPN

An IP VPN [47, 57] is an extension of a private network that encompasses links across a shared or public network like the Internet. A secure VPN uses a combination of tunneling and data encryption to securely connect remote users and remote offices. Thus VPNs can replace troublesome remote-access systems and costly leased lines. There are currently three major tunneling protocols for VPNs [57]: Point-to-Point Tunneling Protocol (PPTP), Layer 2 Tunneling Protocol (L2TP), and Internet Protocol Security (IPSec). IPSec [56] has the advantage of offering advanced cryptographic services and proves to be the best security protocol for LAN-to-LAN VPNs (which is what we need), while other security protocols work better for Host-to-Host connections. Besides IPSec is now well known and integrated in many operating systems (e.g. FreeS/Wan for the Linux OS [74]). Therefore IP VPN relies on IPSec.

8.1.2 A Centralized Approach that Meets Secure Group Communication Needs

A VPN service provider (or VPN SP) is responsible of the IPSec/VPN deployment and management between the various sites, and that this service provider controls a VPN Edge Device (or ED), a small router with IPSec support, in each site. The Virtual Network Operation Center (or VNOC) is the central point of the service provider that collects all the configuration and policy information and that remotely configures the ED of each site during IPSec tunnel establishment.
This centralized but dynamic approach, which was designed for unicast transmissions, is well suited to our needs. The VPN SP can easily take in charge the group security management aspects (authentication and access control of the sites that want to join a VPN) on behalf of the communication group. The dynamic aspect of the VPN topology (since a site can join or leave a VPN at any time) fits well with the dynamic nature of a multicast group. Therefore taking advantage of the VPN infrastructure to offer a fully secure group communication service seems reasonable. It must be noticed that *security is managed on a per-sites basis, not on a per-node basis, which is reasonable when threats arise from the Internet.*

8.1.3 Security Versus Scalability

We believe that this approach meets many needs, in particular for the deployment of services in commercial and competitive environments requiring a high level of security. A typical example is a headquarter that needs to distribute a large confidential database to its remote offices. In this case the number of sites concerned is limited (a few tens), but communications must be fully secure (i.e. the source must be authenticated, the content encrypted and the integrity verified).
In this example, typical of the problem we address, security is the primary concern, not scalability. The number of sites that take part in the VPN is limited, at most a few hundreds and usually only a few tens. Therefore having a centralized approach for VPN management (and also for overlay multicast management as we will see later on) is by no means an issue.

Besides, within each site, the number of nodes, senders or receivers, is not limited, which largely increase the effective scalability (in terms of nodes). Finally the scalability in terms of the number of VPNs (rather than the number of sites for each VPN) is a different issue that can easily be addressed by having several VNOC.

8.1.4 The IVGMP Architecture

We now give an overview of the group communication service introduced in [4]. Each VPN ED must implement the Internet VPN Group Management Protocol (or IVGMP). The first goal of IVGMP is to discover group members and sources local to this site. To that goal it relies on the Query/Report mechanism of IGMP. When a local host needs to join a new group not already received by this site, the ED informs the VNOC which performs some policy checking. If the site is authorized to join the group, the VNOC then sends back the new IPSec/VPN configuration to this ED and to all other EDs concerned by the VPN. Thus only authorized sites can join a group. Interested readers are invited to refer to [4] for further details.
A limitation though is that traffic replication is performed by the ED attached to the source. Therefore when the number of remote sites increases, the performances quickly degrade. This is the reason why this chapter introduces the VPRN concept along with our HBM proposal to improve this efficiency.

8.1.5 An Non-Conventional Approach

The approach in [4] departs from the work carried out in the MSEC IETF working group where the problem is to add security to an already existing multicast routing infrastructure. The authors goal is to add a group communication service in a fully secure environment based on IPSec point-to-point tunnels. Scalability aspects, addressed by the MSEC group, is not the primary goal.
This approach also departs from that carried out in the PPVPN IETF working group where the VPN Service Provider, in addition to providing a VPN solution, also masters the core network and is an Internet Service Provider (ISP). Group communication solutions developed by PPVPN providers can easily take advantage of their own provider equipments (e.g. IP routers or MPLS-enabled infrastructure) to offer multicast-capable VPNs [73]. In this case the two entities, the VPN SP and the ISP, are different entities. It avoids ISP dependencies and enables to set up a VPN across sites connected to the Internet via different ISPs, without requiring preliminary ISP agreements and mutual confidence.
The priorities and assumptions made (of VPN SP) are competely different from that of the MSEC and PPVPN working groups. Non-surprisingly, the approaches considered differ and in fact complement each other, by addressing different needs.

8.2 The Traditional VPRN Concept Versus our View of a VPRN

Many aspects introduced by VPNs have a direct analogue with those of physical networks. One of them is the way in which VPN sites are connected together and traffic is forwarded. If a fully meshed topology between the various sites is feasible, it is not the only possibility and creating a non-fully connected topology can be highly beneficial in some situations. It naturally leads to the concept of Virtual Private Routed Network, or VPRN, which emulates a multi-site wide area routed network using IP facilities.
The traditional VPRN model described in RFC 2764 [47] (Figure 8.1-left) considers a provider network as an opaque IP cloud where only nodes on cloud border are part of VPN description;

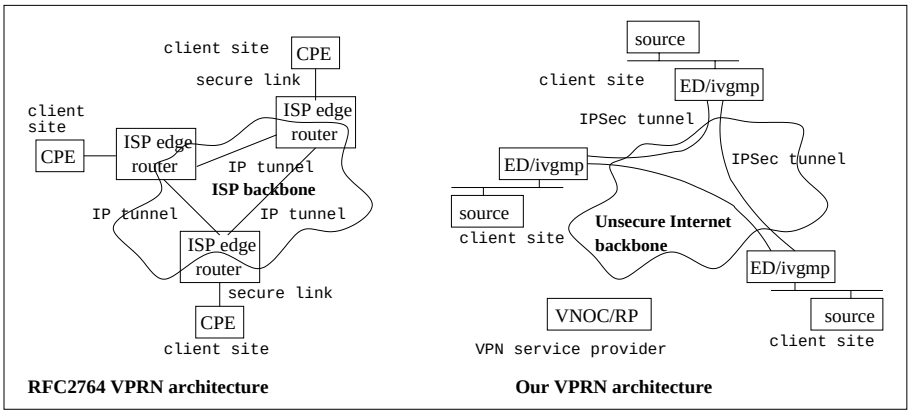


Figure 8.1. RFC 2764 versus ours VPRN architecture for group communications.

nodes within the cloud are transparent. Users access the network via a Customer Premises Equipment (CPE) router, which is a router connecting the customer internal network to the provider's edge router, using a non-shared secure link. The provider is responsible for establishing a mesh of tunnels between the provider's edge routers that have at least one attached CPE belonging to a given VPN. This mesh represents a new dedicated network that virtualizes the physical one. Conceptually, there is a dedicated mesh per VPN, and the mesh topology is arbitrary (partially or fully meshed, depending on customer needs). The main benefit of this approach is that it moves the complexity and the configuration tasks from the CPE router to the provider's edge router. Besides the mechanisms proposed intrinsically rely on the features provided by the underlying physical infrastructure (most of the time an MPLS network).
The VPRN model discussed in this chapter differs quite a lot from the previous model. In our case (Figure 8.1-right) the Edge Device (ED) located in each customer's site behaves as a VPRN node. The complexity and configuration tasks remain hidden to the customer since these ED are remotely managed by the VNOC. Another difference is that the ED/VPRN nodes have a more dynamic nature (compared to an ISP edge router) and are concerned by group management (e.g. by discovering local sources and receivers with IVGMP). Besides, our VPRN architecture is built on top of a generic IP network, without making any assumption on the underlying physical infrastructure. Likewise it does not need any ISP agreement when sites are connected through different ISPs, which is a big asset.

8.3 The IVGMP/HBM Architecture

We have described so far the various concepts and protocols. In this section we describe how they nicely fit with one another.

8.3.1 General Architecture

The VPN approach considered so far is centralized around the VNOC. Thanks to the IVGMP protocol running on each ED, the VNOC is also responsible of collecting and distributing configuration policies and membership information (in terms of sites) for each multicast group. The HBM protocol also assumes the presence of a central RP which collects membership and distance information, and performs topology creation. Therefore *it is natural to merge the various features and add a RP functionality to the VNOC* (Figure 8.2).

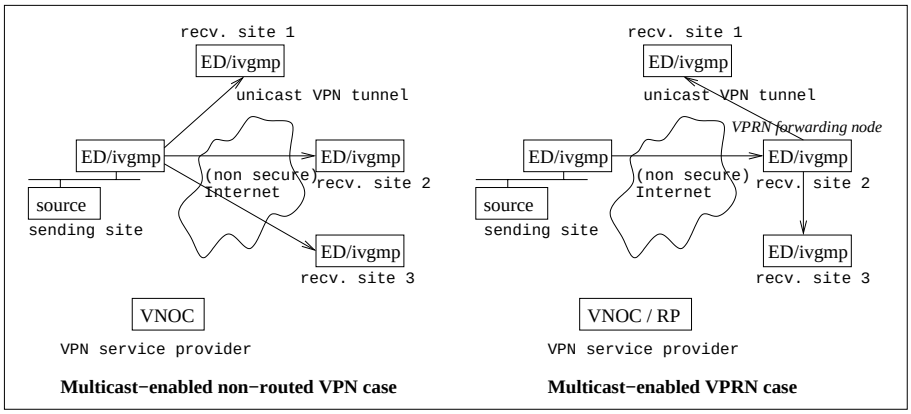


Figure 8.2. The non-routed versus VPRN approaches for group communications in a VPN environment.

8.3.1.1 Improved Scalability

Each VPN site can now act as a VPRN node and can forward traffic to its neighbors in the topology. Doing so reduces the fan-out of the site where the source settles, and because it removes a hot spot in the network, the scalability (in terms of number of sites) is significantly improved.

8.3.1.2 Dynamic Aspects

The group membership dynamic triggers both VPN updates (e.g. by removing tunnels set-up to/from sites that no longer participate in the group) and VPRN/distribution topology updates (e.g. to avoid forwarding traffic to the site that left the group). There is a risk of topology partition (and of packet losses) when a forwarding site leaves the group, until the topology is updated (see chapter 7). Yet, and this is a major difference with the overlay multicast general case, nodes considered here are well administered routers (the ED), instead of end hosts that are far less stable. Therefore the ED departures are almost always negotiated, and appropriate measures can easily be taken.

8.3.1.3 ED-to-VNOC/RP Security

Our architecture gives security the priority over other services. If site-to-site security is addressed by IPSec, site-to-VNOC (or RP since the VNOC acts as a RP too) communications must also be secure. To that goal, each ED establishes a secure communication channel with the VNOC based on SSL and certificates. The ED and the VNOC first authenticate one another, and the establish a secure SSL connection. Remote configuration and other control operations can then take place, using the SOAP approach [17].
The VNOC must be able to deploy all the IPSec features and ensure that key exchange can be handled properly on the VPN EDs. In order to enable secure communications between two EDs, the VNOC supports a framework for automatic key management, IKE [49]. The IPSec Security Associations (or SAs) generated dynamically by the VNOC, are created between two VPN sites to exchange keys as well as any details on the cryptographic algorithms that will be used during a session.

8.3.2 Detailed Description

A more detailed analysis of the VPRN/IVGMP/HBM integration exhibits several slight differences with the initial HBM proposal. In this section we highlight the specificities related to its integration in a VPRN environment.

8.3.2.1 ED Functionalities

In addition to its VPN and IVGMP functionalities, an ED now needs to participate in the metric evaluation with the other EDs of the group. The list of such EDs is necessarily present on each ED since IPSec tunnels are created between them. This is in line with the full membership knowledge assumption of HBM where each node potentially knows all other nodes.
By default, metric evaluation consists in issuing `ping ECHO_REQ/ECHO_REPLY` messages within the IPSec tunnels. It assumes the presence of a fully meshed VPN, otherwise some destinations could not be reached. This is a reasonable assumption since each tunnel between two EDs is in fact shared by all the unicast or multicast traffic between them, and there is a high probability that both sites have already exchanged some packets before.
The metrics are periodically and asynchronously collected by the ED, and sent to the VNOC/RP using the secure SSL channel. Once again, using SOAP is in line with the XML approach used by HBM for control messages.
An ED is rather different from the end-host assumed in HBM:

- an ED is rather stable when compared to a traditional end-host (usually a PC). An ED is a well administered router, that rarely reboots or crashes (at least in theory). This feature greatly improves the overlay multicast solution, since node stability, especially in case of a forwarding node, is of high importance.

- an ED is a small embedded PC, usually running a dedicated Linux OS, and has less processing power than a traditional end-host. The VPRN approach adds some processing on the ED (metric evaluation, packet forwarding), yet most of the work (topology creation, database management and configuration distribution) is performed by the VNOC, not the ED.

8.3.2.2 VNOC/RP Functionalities

Periodically the VNOC/RP calculates a new topology, taking into account new networking conditions (e.g. a congested path between two EDs can lead the VNOC/RP to find an alternate path). This topology update is also performed in case of membership modification (e.g. when a new site joins the group). The new topology is then communicated to the concerned ED. Each topology update message consists in an updated VPRN configuration, instead of the new list of neighbors of a node as in HBM.
A major difference with the initial HBM proposal is the fact that group departure is by default implicit, since a site always subscribes to a VPN for a limited span of time. Because of this soft-state approach, each ED must periodically subscribe to the group, sending a new `JOIN_GROUP` message to the VNOC, otherwise the site is automatically removed. This is different from the original HBM proposal which follows an explicit leave model, plus a partition recovery mechanism in case of ungraceful departures (e.g. after a crash). Having a soft-state model enables the VNOC/RP to asks an ED that implicitly leaves a group to keep on forwarding packets until the new topology has been updated.

8.4 Conclusion

In this chapter we have shown how to build a fully secure but efficient group communication service between several sites. It details both the underlying motivations and the architecture proposed. It is a follow-up of work performed in [4] on offering a group communication service in an IPSec VPN environment and on the HBM application-level multicast protocol. We have shown that these proposals, that both follow a centralized approach, naturally fit with one-another and lead to the concept of Virtual Private Routed Network.

Part III

Discussion, Conclusions, and Future Work

Chapter 9

Discussion, Conclusions and Future Works

Contents

We have so far described and compared a large variety of alternative group communication services proposals, and we have also introduced and analyzed our own proposal, HBM. We now discuss several key points that were raised and classify them in decreasing order of importance (which can be modified according to the exact target application requirements). For each point, we identity our contributions and, when applicable, future works.

9.1 Ease of Deployment

9.1.1 Discussion

An AGCS should be easy to deploy to offer a viable alternative to native multicast routing. Manual deployment is only realistic if the procedure is straightforward, for instance to bootstrap the AGCS system (e.g. to specify a reflector or a rendez-vous point address).
Several proposals [46, 101] suggest using an active networking approach to provide an AGCS. It is clear that if an active networking service is available in each potential node, then, because of the flexibility it offers, deploying an AGCS becomes an easy task. Yet this is largely dependent on the availability of the active networking service in a sufficiently large number of nodes.
Also, Network Address Translation (NAT) [29, 84] devices and firewalls present non-trivial issues within the context of some of the AGCS techniques discussed in chapter 2. Some of these issues require the use of *application-level gateways*, for instance to ensure the correct translation of addresses/port numbers.

9.1.2 Contributions in this Thesis

HBM has been implemented as a library (GCSL) [30], which can be either integrated in a legacy application requiring group communication services, or used as a standalone application to interconnect multicast islands. The only information needed to start the HBM services are the IP address/port number of the RP and the IP address/port number of the group you need to join.

9.2 Robustness

9.2.1 Discussion

Inter-domain multicast routing is often said to be fragile. If an AGCS offers a way to alleviate this problem, it also creates other instability problems. For instance, a solution based on end-hosts (usually PCs or workstations) is intrinsically less robust than one based on dedicated and well administered commercial routers. There is a high risk, as the group size increases, that the topology be partitioned after a single node failure. Some proposals address this robustness aspect proactively by using some level of flooding (e.g. gossiping approaches).
Robustness also requires that node failures be rapidly discovered. This feature depends on which distribution topology (i.e. tree, ring , etc.) is used: for instance, with a shortest path tree, ACK messages can be generated by the leaves and aggregated by the transit nodes as they are sent back to the source; a transit node that does not receive an ACK from one of its downstream neighbors can easily conclude that there is failure.
Many proposals only content themselves with a detection and repair mechanism. Having a fast detection and repair mechanism is required, but in our opinion, is not sufficient. For instance, some applications may require that partitions be avoided altogether (e.g. a cooperative work session requires that all members be present during the whole session)

9.2.2 Contributions in this Thesis

This thesis focuses on (1) the partition problem due to a single node failure and (2) on packet losses generated by transient incoherences in the overlay topology.

To solve the first problem, HBM adds redundant virtual links (RVL) to the overlay topology. In this thesis, we introduce five strategies that differ on the number of added RVL links and these links between which nodes they are added. We find that adding RVL links only between core nodes is a good solution. We also find that not only it is suitable for a single node failure but it also decreases the partition probability when two or three nodes fail simultaneously.
Another benefit of adding redundant links is significantly (1) to detect some failures rapidly (there is a failure if a node receives data packets via only one of the two posibilities, the normal and RVL links) or (2) to reduce the end-to-end delay as in TMesh [100].
Since the overlay topology update is usually not synchronous, at some point in time a subset of the nodes will be informed of (and probably will use) the new overlay topology, while others will still use the previous old one. This routing incoherency can easily lead packets in transit to be lost. In this thesis we introduce and compare four strategies, that differ on the number of topologies that each node needs to remember (just the current one versus the current and the previous one), and the way to behave when a packet is received with a topology identifier which does not correspond to the one(s) a node is aware of.
The best solution is undoubtedly the "two topologies" approach which yields a high robustness and does not incur packet duplications, and is simple to implement, the necessity to manage a Topology Sequence Number (TSN) that is both globally unique and monotonically increasing.

9.2.3 Future Work

Future work will consider the use of multiple mirrored RPs as a complementary way of increasing the HBM robustness, and will study when this is beneficial and what are the associated limitations.

9.3 Impacts of Cheats

9.3.1 Discussion

While not disrupting the flow of data on the overlay tree, there is an opportunity for receivers (i.e. cheats) to try and improve their position on the overlay tree by "manipulating" distance measurements, to be positioned closer to the data source while limiting to a minimum their replication burden.
If we consider the very popular round-trip time (RTT) distance measurements used in many application-level multicast protocols, a cheat could delay a probe received from another receiver to artificially increase their measured distance in order to try and reduce its replication burden (since the further away another receiver is, the more likely that receiver will be connected to another closer node). A cheat can also either lie outright about its distance to the data source or even change the time stamp in probes received from the source to a value in the future, in order to try and be positioned closer to the source in the tree and therefore improve its observed delays.
It is important to note that cheats do not attempt to disrupt the flow of data along the overlay tree or even to break the protocol used to build the tree, they simply try and improve their position on the tree. In other words, we are not interested in disruptive behavior such as denial of service: once in the tree, although the cheats can keep lying about measurements, they otherwise follow all other protocol rules.
Although the problems of cheats in application-level structures (and overlay trees in particular) has often been mentioned, we are not aware of any study of their effects on application-level multicast protocols, as well as on the underlying network. We believe that understanding

such effects is critical if the benefits offered by application-level multicast are to be reaped in application domains where the receivers do not pertain to the same administrative domain (e.g. corporation), as is the case in gaming, video distribution/webcasting, etc.

9.3.2 Contributions in this Thesis

A HBM cheat always reports a minimal distance to the source and a huge distance to the rest of the group. Because other nodes, including the source, periodically probe their distance to the cheats, for instance with RTT measurements, the cheats also try to alter these measurements accordingly, in order to achieve their goal. This is mandatory since otherwise the RP could easily detect cheats by comparing the $A \hookleftarrow B$ and $B \hookleftarrow A$ RTT measurements. If they differ significantly, the RP could easily conclude that one of A and B has a suspect behavior. Then, after cross-checking with other metrics evaluations where A and B are implicated, the RP could easily determine which node is cheating.
With this simple cheating method, a single cheat is guaranteed to be directly linked to the source node, while having no sibling at all. Said differently a *single* cheat is a leaf in the shared tree created by the RP. The topology update mechanism of HBM has no effect on this situation, since the set of metrics where this cheat is implicated remains the same. But depending on the tree fanout and the number of cheats, the situation can be quite different when the total number of cheats increases. [68] discusses the impacts of cheats in various application-level multicast protocols, and in particular HBM. We show that the presence of an increasingly high number of cheats has after strange consequences on the overlay topology created.

9.3.3 Future Work

As future work, we will study the impacts of cheats on our HBM proposal and on other Application level multicast proposals, also study if and how HBM can be made more robust to cheating.

9.4 Security

9.4.1 Discussion

A point that is usually neglected in the AGCS proposals is security. The AGCS service provided does not offer any additional security (e.g. there is no authentication of the nodes), nor is itself secure (e.g. control mechanisms are not secured). This is paradoxical since most proposals are based on unicast communications, either among group members and/or members and a rendez-vous point. Indeed, offering security for point-to-point communications (or when there is a central rendez-vous point collecting information on group members) is much simpler than in the general case of multicast communications where many additional hard problems must be solved (see the MSEC multicast security IETF working group charter).

9.4.2 Contributions in this Thesis

In [2] we explain how a group communication service can be set up in a fully secure virtual private network (VPN as in [21]) environment. Here, point-to-point IPSec tunnels are dynamically created between the sites that host group members, and removed when all members have left. It is a follow-up of work on group communication service in an IPSec VPN

environment [4] and on our HBM multicast protocol [86]. We show that these proposals, that both follow a centralized approach, naturally fit with one-another and lead to the concept of Virtual Private Routed Network, of VPRN, which largely improves distribution efficiency. It is worth noting that centralized overlay multicast approaches, often criticized, find a perfect field of application in VPN environments, and the security benefits brought by the whole architecture largely compensate any possible scalability limitation.

9.4.3 Future Work

The following general goals should be considered when designing and implementing a secure multicast system: authorization, authentication, encryption, data integrity, data confidentiality, access control, non-repudiation, and robustness against denial-of-service attacks. Future work will study how these services can be provided by HBM.

9.5 Performance

9.5.1 Discussion

The performance of most AGCS is non-surprisingly lower than that of native multicast routing protocols because doing traffic forwarding at the end-host or a limited number of hosts (e.g. with reflectors) is necessarily less efficient than using multicast routers in the backbone.
But performance is not necessarily the primary target of these proposals. For instance, solutions dedicated to ad-hoc networks and gossiping techniques used in large dynamic peer-to-peer communities, deliberately lay more importance on robustness(and scalability with gossiping). Consequently the topology created deliberately includes many redundant paths that affect the final performance. Likewise reflector-based solutions lay more importance on the ease of deployment aspect than on performance.
On the contrary, proposals creating an automatic overlay topology are more concerned with creating good data delivery topologies. Many aspects will affect performance:

- the type of topology created: a per-source shortest path tree is undoubtedly more efficient than a single shared tree used by many different sources. But managing several trees, one per source, also has a higher cost.
- the possibility of dynamic topology adaptation: the topology must reflect the dynamic networking conditions, so network monitoring is required. Passive monitoring is sometimes possible, taking advantage of on-going data flow reception statistics, otherwise active monitoring is required, adding some overhead. Since a topology update has a high signaling cost, this adaptation is necessarily done with a low frequency.
- the performance metrics considered: many works only consider communication delays, assuming that all paths are symmetric (which enables the use of simple tools like `ping`). [22] reminds that performance can be affected by the presence of asymmetric unicast routing, which is not so uncommon in the Internet. Finally [51] argues that the delay and bandwidth metrics should both be considered.
- per-host profiling: nodes can largely differ and network-related metrics cannot catch all of their specificities. For instance lightweight nodes (e.g. PDAs), with limited processing power and battery, should not become transit nodes, even if they benefit from small communication delays. Likewise a history of all nodes should be kept so that the unstable nodes (e.g. because of a wireless connection) be moved to the leaves

of a tree. Overlay multicast proposals using a centralized algorithm like HBM, because of their node database [86], can easily take it into account during the topology creation process.

The price to pay for higher performance are: (1) more complex topology maintenance algorithms and (2) a higher signaling load to perform network monitoring and dynamic topology adaptation. There is clearly a trade-off to find between performance and management costs.

9.5.2 Contributions in this Thesis

Performance has not been actively studied in this thesis. Yet we have introduced the idea of dynamic topology adaptation and per-host profiting at the RP mentioned above.

9.6 Scalability

9.6.1 Discussion

If scalability in terms of the group size is an explicit target of traditional multicast routing protocols, it is not required in all situations. Consequently some of the AGCS discussed in chapter 2 (e.g. reflectors or HBM) are specifically designed to handle small groups, which is sufficient in many situations (e.g. collaborative work). Better control mechanisms (e.g. to adapt more frequently to networking conditions) or a simpler architecture compensate for the lack of scalability.
Many other proposals (e.g. scalable adaptive hierarchical clustering [70]), on the contrary, target a high scalability, which is required for instance with large peer-to-peer applications. This scalability is usually achieved with a hierarchical overlay topology (e.g. based on clustering) and a distributed partial membership knowledge.
Since intra-domain multicast routing is often available, a frequent assumption is that the AGCS is only used between sites, not within a site. A representative in each site multicasts locally the traffic received. Doing so increases the global scalability since all the local members are hidden behind their representative.
In some cases, the scalability is not in terms of number of members but number of concurrent groups, which is a totally different issue, as discussed in section 2.3.5. This kind of scalability can be important for deploying new services and protocols over the Internet (e.g. there are proposals to improve Mobile-IP hand-off thanks to XCAST, which can be valuable in a Cellular-IP environment with a very high number of mobile nodes).
Finally the idea of aggregated multicast with inter-group tree sharing [36] can easily be applied to AGCS. For instance any collaborative work session is composed of several audio/video/white-board tools with approximately the same set of end-users. Sharing a single overlay topology would help reducing the global control overhead.

9.6.2 Contributions in this Thesis

In this thesis, we discuss the scalability of our HBM proposal, even if we do not regard it as a key requirement. After a detailed modelization of the HBM protocol behavior, we explain how the scalability can be improved by limiting the size and the number of the control messages exchanged between the RP and members, out of the four algorithms we introduced, we find that one of them offers a good compromise between the control traffic overhead and the number of metrics periodically updated.

9.6.3 Future Work

We have addressed the HBM scalability in terms of control traffic overhead. But other aspects must be considered, and in particular the RP load and the number of TCP connections managed by each of them. To that goal, future work will consider the possibility of having multiple RPs in a session, one primary RP and one or more secondary RPs (not for backup) that will manage different subsets of the members. The primary RP is the responsible of overlay topology creation. The first contact to the group is via the primary RP, and then the primary RP directs a new member to the closest secondary RP. Each secondary RP informs the primary RP of all the information related to its subset of members.
Another aspect is the use of single overlay structure (with minor variations) for several closely related groups. This is typically the case in a collaborative work session, since most participants will share the same audio, video application sharing, white board tools. Only low end receivers (if any) will avoid the use of for instance video tools. There is clearly the possibility here of using a single overlay topology for all four groups, therefore serving a lot of control overhead. Yet the presence of low-end receivers, if any, needs to be taken into account by the RP when creating the topology so that they be leaves. How this is done, the associated trade-offs, and the evaluation of the gains are left to future work.

9.7 Dynamic Discovery of Sources and Receivers

An AGCS system must be informed of the presence of sources and receivers. Many AGCS solve this problem by using a static configuration where the local administrator/user decides beforehand what group(s) to distribute. This can lead to useless traffic distribution, for instance after the last interested host has left from a site. A more elaborate solution is thus required.
In the simplest case, the AGCS is implemented as a library linked to the application. A direct communication is then possible (e.g. through a dedicated API) and source/receiver discovery is immediate.
But when the AGCS tool runs on a different host than applications, two cases are possible: (1) either the AGCS tool is on the same LAN as group members (sources or receivers), or (2) on a different LAN. Case (1) is easily addressed, by listening to IGMP traffic and multicast traffic. But this solution no longer works with case (2) since the top multicast router by default isolates the various LANs. [4] gives some elements on how to address the source and receiver discovery problems that arise in such a case.
This discussion shows that local source and receiver discovery is not so simple and is rarely considered. Yet this is the price to pay for the AGCS to follow the dynamic behavior of group members and to limit useless traffic.

9.8 A Few More Words ...

Application level multicasting has triggered a lot of work. Four years after its birth, and in spite of a huge amount of proposals, several key aspects that can deter its deployment are still open points. Recent work on cheats in application level proposals is, from this point of view, significant... And to the best of our knowledge, nobody has yet addressed the problem of Denial of Service attacks on application-level multicast which will sooner or later take place. More generally these proposals, which at the beginning were considered by many as a quick and dirty alternative to multicast routing, tend to be considered as realistic group communication solutions. Our work, during this thesis, and the previous

remarks lead us to be more cautious concerning their future. A lot of work remains to be done on the subject, but realistic fields of applications already exist, and the HBM/VPN integration we have proposed is one such example. It is highly likely, when considering the huge $application requirements * class of problems$ matrix that many different proposals will continue to come out. We have addressed in this work only a small subset of them. Our proposal is neither better nor worse than others, but addresses a different set of requirements and problems in the best possible manner.

Quatrième partie

Résumé en Français

Résumé étendu français

Avertissement : ce résumé vise à permettre à un lecteur de comprendre l'essentiel du travail effectué dans cette thèse. En revanche il ne couvre pas la totalité du travail. Ainsi il ne comporte qu'une introduction au domaine, la présentation de notre proposition, l'étude d'un aspect qui nous a semblé important, à savoir la robustesse de la topologie créée face à des défaillances, et une discussion synthétique finale.
Pour tous les autres aspects, à savoir l'état de l'art du domaine, les techniques de création de la topologie de recouvrement, le passage à l'échelle, la réduction des pertes de paquets lors d'une mise à jour de la topologie, et l'utilisation de HBM au sein d'un environnement de VPNs, le lecteur est invité à se référer à la version anglaise de ce document.

Chapitre 10

Introduction

Contents

10.1 Problèmes de déploiement du routage multicast

Les communications de groupe requièrent traditionnellement que chaque noeud, sur chaque site, accède à un service de routage multicast natif. Si le routage intra-domaine (interne à un site ou à un LAN) est largement disponible, ce n'est pas le cas du routage inter-domain, ou au sein d'un FAI (Fournisseur d'Accès Internet, ou encore ISP) privé. Le multicast IP a été un sujet chaud de recherche et développement pendant de nombreuses années. Malgré cela il reste nombre de points ouverts qui rendent son déploiement difficile. De nos jours, de nombreux FAIs sont peu enclins à fournir un service de routage multicast en raison de problèmes techniques ou marketing, ainsi que nous allons le voir [27].

10.1.1 Migration des routeurs

Un frein majeur au déploiement du routage multicast est qu'il chamboule le modèle de migration des FAIs, qui veut qu'un routeur neuf soit d'abord déployé dans le réseau de coeur, et au cours du temps, vers les points d'accès clients, en périphérie. Ces anciens routeurs sont alors conservés jusqu'à ce que le coût nécessaire à leur remplacement soit inférieur au coût de maintenance et de mise à jour nécessaire à leur utilisation.
Le service de routage multicast bouscule ce modèle car les matériels anciens supportent mal le multicast. En effet, les versions anciennes des systèmes d'exploitation de ces routeurs ne sont pas suffisamment matures quant au support des protocoles relatifs au multicast. S'il n'existe pas de possibilité de mise à jour logiciels de ces routeurs, ces derniers sont alors mis de coté prématurément, alors qu'ils auraient pu sinon rendre de nombreux services.

10.1.2 Indépendance des domaines

Les FAIs ne voulant pas dépendre d'autres FAIs, chacun d'eux crée un service de routage multicast localement, indépendant. Cela conduit naturellement, dans le cas de PIM-SM, à la multiplication des RPs (points de rendez-vous au sens PIM-SM du terme), un par FAI, afin que les sources et récepteurs locaux puissent communiquer efficacement. Dès lors plusieurs problèmes surviennent, tels:

- un FAI dépendant d'un RP distant car la source est externe à son domaine n'a que peu de contrôle sur le service rendu à ses clients récepteurs,
- les annonces de sources entre domaines, doivent avoir lieu de façon efficace et scalable, ce que ne permet pas le protocole MSDP actuel,

L'arrivée de PIM-SSM simplifiera indéniablement ces problèmes (MSDP n'a plus alors de sens), sans pour autant totalement les faire disparaître.

10.1.3 Gestion complexe

Des problèmes classiques lors du déploiement du routage multicast sont la traversée des firewall et la présence de NAT (les adresses de classes D ne sont pas reconnues). Ces problèmes conduisent souvent à tuneller les paquets, afin qu'ils traversent les équipements, mais cela crée aussi des failles de sécurité.
Quant au routage multicast inter-domaines, nous avons déjà souligné les problèmes techniques majeurs soulevés. Cela est d'autant plus vrai lorsque l'on considère la gestion et le débug des configurations multicast à l'inter-domaine.

10.1.4 Justification des coûts

Fournir un service de routage multicast a aujourd'hui un coût qui est supérieur à celui d'un simple routage unicast, en terme de déploiement et de gestion. De ce fait ce service n'a de sens que si les gains liés aux économies de trafic sont supérieurs aux coûts du service. Ceci n'aura lieu probablement que si le nombre de récepteurs excède un certain seuil, à définir, les coûts fixes étant relativement élevés.
D'autre part le business modèle, et en particulier la question fondamentale de "qui doit supporter le coût d'un trafic multicast", restent ouverts. Ce service doit il être inclus dans les frais globaux d'accès facturés par le FAI, ou bien à la charge de la source (qui a un bénéfice immédiat à l'utilisation d'un routage multicast, une seule copie de chaque paquet passant sur son lien montant), ou de chaque récepteur (qui chacun bénéficient du flux de données) ? Ce point reste ouvert.

10.1.5 Sécurité

Cet aspect des choses est certainement l'un des plus critiques. Outres la sécurité applicative, qui reste à la charge des participants (source et récepteur), le FAI qui offre un service de routage multicast s'expose à de nombreuses attaques (parfois triviales à monter).
Les attaques sur MSDP en sont un exemple typique... Suite à une infection de machines clientes, celles-ci ont scanné les ports d'un grand nombre d'adresses IP, parmi lesquelles des adresses de classe D. Si scanner des adresses multicast n'était certainement pas l'objectif du pirate mais plutôt le résultat d'une erreur de programmation, le résultat en revanche a été

une attaque par déni de service (DoS) sur le protocole MSDP qui a du annoncer un très grand nombre de sources à ses pairs situés dans les autres domaines.
Cet exemple d'attaque (ici à moitié volontaire) montre la grande fragilité du service de communication de groupe tel qu'il est défini en ce moment, et les risques auxquels s'expose un FAI entreprenant. Non seulement le service multicast est fragile, mais une attaque dans le plan de données ou de contrôle des protocoles multicast peut fort bien impacter le service unicast de base, et donc tous les utilisateurs du FAI. Raison de plus pour ne pas faire trop zèle...

10.2 Buts de ce travail et organisation du document

Face à cette situation, nombre de propositions alternatives ont été faites durant ces dernières années, afin de fournir un service de communication de groupe même en l'absence de routage multicast natif.
Dans cette thèse, nous présentons une proposition pour établir un service alternatif de communication de groupe qui déplace le support de multipoint depuis les routeurs vers les extrémités. Notre proposition, appelée HBM fonctionne au niveau applicatif et fournit un service efficace de distribution multipoint de données de type un-vers-plusieurs ou plusiers-vers-plusieurs. Avec cette approche les machines terminales, les serveurs et/ou des routeurs de bordure s'organisent automatiquement en une topologie de distribution de recouvrement grce à laquelle des données sont diffusées. Cette topologie de recouvrement peut se composer à la fois de connections point à point et de zones bénéficiant d'un routage multicast natif (par exemple dans chaque site). Par conséquent elle offre un service de communication de groupe à tous les hôtes, même ceux situés dans un site qui n'a pas accès, pour une quelconque raison, au routage multicast natif.
HBM est une solution centralisée, où tout, y compris la gestion d'adhésion au groupe et la création de la topologie de recouvrement Cette thése se concentre sur trois aspects principaux: scalabilité, robustesse et sécurité. La scalabilité peut être en grande partie améliorée, avec quelques ajustements simples de certains paramètres de HBM tels la fréquence et la taille des messages de contrôle. La robustesse est un aspect pratique important, et nous présentons des techniques de réduction de pertes de paquet, tels l'addition des liens virtuels redandants qui évitent la partition de topologie en cas de panne d'un noeud de transit. Enfin nous étudions l'utilisation de HBM afin d'établir un service entièrement sécurisé mais efficace de communication de groupe entre plusieurs sites au moyen de VPN IPSec. Nous montrons que HBM et un environnement de VPN IPSec sont naturellement complémentaires et conduisent au concept de réseau privé virtuel routé (ou VPRN).

Chapitre 11

Notre proposition: HBM

Contents

HBM est une solution centralisée où tout, y compris la gestion d'adhésion au groupe et la création de la topologie de recouvrement (ou "overlay"), est de la responsabilité d'un unique point de Rendez-vous (ou RP).

11.1 Description de notre proposition

HBM (Host Based Multicast) est une proposition qui crée automatiquement une topologie de recouvrement entre les différents membres du groupe, source ou récepteur, en utilisant des tunnels UDP point à point entre eux. Tout est contrôlé par un unique hôte, le "Rendez-vous Point", ou RP (à ne pas confondre avec la notion de RP de PIM-SM). Ce RP:

- connaît les membres du groupe,
- connaît leurs caractéristiques,
- connaît les coûts de communication entre eux,
- calcule périodiquement une topologie de recouvrement,
- diffuse cette topologie au sein des membres concernés,
- enfin gère tous les aspect de gestion de groupe (adhésion, retrait, recouvrement après panne).

De leur coté les membres:

- dialoguent avec le RP, et en obtiennent le sous-ensemble de la topologie de recouvrement les concernant,
- évaluent périodiquement les distances avec les autres membres (ou un sous ensemble de ces derniers) et communiquent ces métriques au RP, et enfin

- échangent du trafic au sein de la topologie de recouvrement (le but premier !).

Si le trafic de données véhiculé au sein de la topologie de recouvrement utilise des tunnels UDP, en revanche le trafic de contrôle entre membres et RP utilise des connections TCP (ce qui est nécessaire afin de garantir la fiabilité des transmissions).

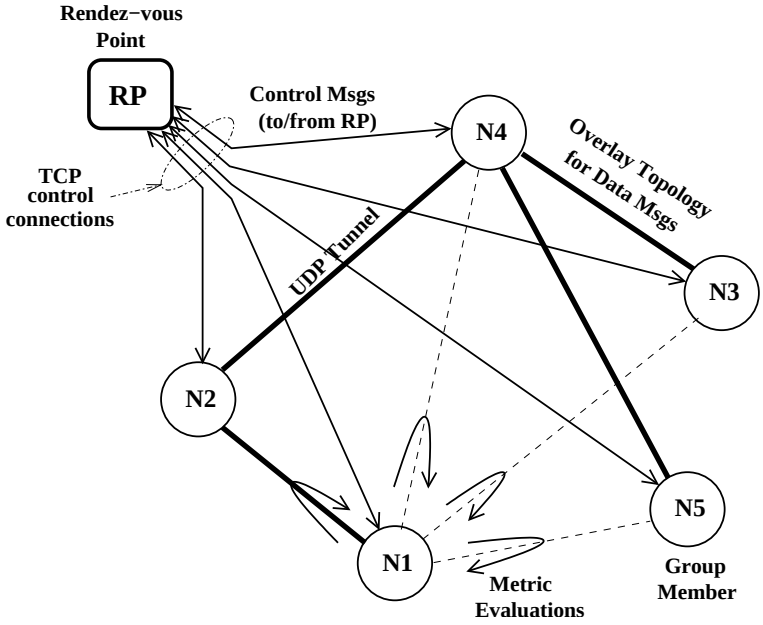


Figure 11.1. Les connexions mises en oeuvre dans HBM.

La figure 11.1 décrit les différentes connexions existant entre membres et RP.
La topologie de recouvrement créée n'est pas figée, mais au contraire se doit d'être périodiquement mise à jour, ceci afin:

- de refléter les conditions réseaux changeantes (mais comme nous le verrons plus tard il ne faut pas compter sur cette dynamicité pour résoudre d'éventuels problèmes de congestion, la fréquence des mises à jour étant faible),

- parce que de nouveaux membres joignant le groupe sont initialement ajoutés de façon non optimale à la topologie existante,

- parce que, suite à des départs volontaires ou brutaux (crash) de membres, la topologie doit être réparée,

- enfin parce que les mises à jour de la topologie effectuées suite à des départs ou pannes sont faites localement, de façon non optimale, dans un soucis de rapidité.

Par conséquent un membre du groupe évalue périodiquement les coûts de communication qu'il a avec les autres membres (ou un sous ensemble de ceux ci, pour des raisons de passage à l'échelle comme nous verrons plus tard), et informent le RP. Le RP de son coté calcule périodiquement de nouvelles topologies et en informe tous les membres concernés (seul un noeud ayant des modifications de voisinage reçoit un message de modification lui indiquant ses nouveaux voisins). Ces deux mécanismes sont totalement *asynchrones*, et le calcul de la topologie se fait en se basant sur la base de données de métriques du moment, indépendamment du processus de mise à jour de ces métriques.

Notons que pour des raisons de commodités nous avons considéré uniquement les temps d'aller-retour (RTT) et taux de pertes, ainsi que donnés par la commande `ping`. D'autres choix de métriques sont possibles, et suivant l'application envisagée, peuvent être plus pertinents.

11.2 Messages échangés par HBM

Les messages de contrôle échangés entre RP et membres sont:

- JOIN: envoyé au RP par un nouveau membre qui se joint à la session
- MEMBER LIST: envoyé par le RP à un nouveau membre afin de l'informer de la liste des autres membres du groupe (afin qu'il puisse ensuite évaluer sa distance avec eux), ou un sous ensemble des membres du groupe si une connaissance totale n'est pas souhaitable (ou possible).
- MEMBER UPDATE: envoyé par le RP à tous les membres afin de leur signaler l'arrivée ou le départ d'un autre membre
- LEAVE: envoyé par un membre qui désire quitter une session (évidemment ce ne sera pas possible en cas de crash)
- LEAVE OK: envoyé par le RP au noeud ayant émis un LEAVE afin d'achever le processus de départ négocié. Entre temps le RP aura calculé et diffusé aux noeuds voisins concernés une nouvelle topologie locale, afin d'éviter des pertes de paquets
- FAILURE: envoyé au RP par un membre qui détecte qu'un noeud voisin n'est plus accessible,
- METRIC UPDATE (MU): envoyé par un membre avec la liste des métriques mises à jour,
- TOPOLOGY UPDATE (TU): envoyé par le RP à chaque membre concerné par une mise à jour de la topologie, avec la liste de ses nouveaux voisins.

```
<metricupdate>2                          # MU report form node id=2
 records=4                               # Number of metrics
<record>2, 1                             # metric between 2 and 1
<metric>2.50, 0.12</metric>              # RTT=10.10ms, loss=0.12%
</record>
<record>2, 3
<metric>30.10, 0.010</metric>
</record>
<record>2, 4
<metric>1.60, 0.195</metric>
</record>
<record>2, 5
<metric>10.10, 0.001</metric>
</record>
</metricupdate>                          # Message end
```

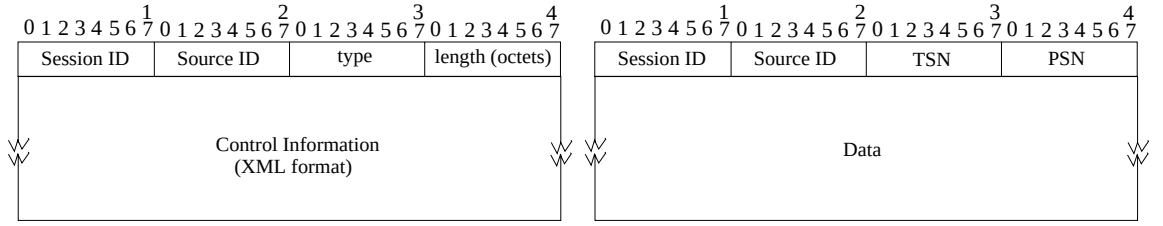
(a) Message de contrôle de type Metric Update (MU)

```
<topologyupdate>2                        # Message start for node id =2
 records=2                               # Number of links =2
<record>2, 4                             # link between 2 and 4
 type=1                                  # type of link = 1: tree link
groups=1                                 # Number of groups in this link =1
<group>16843232, 1111</group>            # group ip=224.1.1.1, port = 1111
</record>
<record>2, 1
 type=1
 groups=2
<group>16843232, 1111</group>            # group ip=224.1.1.1, port = 1111
</record>
</topologyupdate>                        # Message end
```

(b) Message de contrôle de type Topology Update (TU)

Figure 11.2. Un exemple de messages de contrôle MU et TU.

Les messages de contrôle suivent un formattage XML (figure 11.2). Ce sont donc des messages textuels, aisément lisibles et interprêtables par les membres et le RP. Un inconvénient est bien sûr une taille supérieure comparé à des formats binaires, mais ceci est conforme à la tendance générale qui privilégie la flexibilité pour les opérations de contrôle (penser à RTSP, SDP, SDPng, SOAP, SMIL). Et rien n'empêche une compression de ces messages...



(a) TCP/IP segment

(b) Format des paquets de données UDP/IP diffusés sur la topologie de recouvrement

Figure 11.3. Format des paquets de contrôle HBM.

Les messages de contrôle et les paquets de données sont ensuite envoyés au sein de connexions TCP (contrôle) ou tunnels UDP (données), en suivant le format donné figure 11.3. Notons en particulier que les paquets de données incluent des champs TSN (Topology Sequence Number) et PSN (Packet Sequence Number) qui ont un rôle majeur pour la robustification des transmissions, même en présence de mises à jour de la topologie (chapitre 7 en anglaise).

11.3 Discussion de HBM

De part cette introduction à HBM, il est clair que:

- *HBM offre un passage à l'échelle limité :*
  Plus généralement toute solution créant une topologie de recouvrement basée sur des connexions point à point a des problèmes de passage à l'échelle, même si HBM en ajoute d'autres du fait qu'il suit une approche centralisée. Cependant un bon nombre de sessions de travail collaboratif n'incluent qu'un nombre raisonnable de participants. En outre, un membre peut aussi servir de relais ("multicast reflector") à un nombre plus important de membres locaux au site ou LAN de ce dernier. Le nombre réel de participants peut donc largement excéder le nombre de membres connus et gérés par HBM. Nous avons vu chapitre 5 (en anglais) comment contrôler le flux de contrôle qui est l'un des verrous pour un passage à l'échelle.

- *dépend largement de la fiabilité du RP :*
  Le RP est typiquement un noeud sensible, et la fiabilité de l'ensemble repose sur ses épaules. Cependant, si le RP est co-localisé avec le source principale de la session (si elle existe), alors la fiabilité de l'ensemble n'est pas particulièrement impactée par le RP, le destin de celui ci étant de toute façon lié à celui de la source.

- *est intrinsèquement fragile :*
  C'est un problème majeur de toute solution de multicast applicatif. Un noeud terminal est intrinsèquement plus fragile qu'un routeur bien administré, sur lequel tourne un nombre limité de logiciels, et administré par des équipes compétentes. Et si un noeud de transit de la topologie de recouvrement vient à tomber, alors une partition est créée. Nous verrons chapitre 12 comment gérer ce problème au moyen de liens redondants.

D'un autre coté:

- *c'est une solution simple :*
  C'est sont avantage premier. Les informations étant centralisées sur le RP, qui est aussi en charge de la création de la topologie, aucune incohérence ou aléa n'est à craindre. En outre, des membres un peu légers (type noeuds mobiles) sont déchargés de taches trop complexes, prises en charge par le RP, ce qui est un avantage pratique important.

- *qui crée aisément une topologie de recouvrement pas trop mauvaise :*
  La topologie créée est optimale vis à vis de la base de métriques maintenue par le RP. Au final la qualité de la topologie de recouvrement dépend :

  – de la fraîcheur de la base de données de métriques : plus la fréquence de rafraîchissement est élevée, meilleure est la précision de la base,

  – du type de topologie créée : un arbre par source est indéniablement meilleur qu'un arbre partagé (ou qu'un anneau), mais au pris d'une plus grande complexité (s'il y a plusieurs sources), voir fragilité (vis à vis d'un anneau).

Chapitre 12

Amélioration de la robustesse face à des défaillances de noeuds par ajout de liens virtuels redondants

Contents

Dans cet chapitre nous montrons qu'ajouter des liens redondants (ou RVL, Redundant Virtual Links) à la topologie de recouvrement aide à réduire la probabilité de partition de cette topologie. Les questions auxquelles nous nous efforcerons de répondre sont donc:

- Combien de liens RVLs doivent être ajoutés à la topologie de recouvrement ?
- Entre quels noeuds ces liens RVLs doivent ils être ajoutés ?
- Ces ajouts doivent-ils se faire en tenant compte de la position de la source ou non ?

12.1 Introduction

De nombreuses propositions de multicast applicatif se contentent, afin de faire face aux problèmes de partition de la topologie de recouvrement, de mécanismes de détection et de réparation rapides de cette topologie. De notre point de vue cette approche réactive est notoirement insuffisante. Ainsi certaines applications peuvent nécessiter que la probabilité de partition soit quasi nulle, sous peine de pénaliser grandement l'application et les utilisateurs, auquel cas réagir n'est pas suffisant.
Une approche qui partage certaines similitudes avec notre travail est le "Probabilistic Resilient Multicast (PRM)" [13]. Ici un sous ensemble des noeuds de l'arbre de recouvrement

transfert aléatoirement des paquets à d'autres noeuds, créant ainsi des chemins redondants. Les paquets issus de ces "sauts" sont ensuite retransmis sur les sous-arbres, à moins qu'ils n'aient déjà été reçus. Si cette solution offre une garantie probabiliste de bonne réception, en revanche elle est limitée par l'approche aléatoire des sauts, qui est la seule possibilité lorsqu'aucun noeud n'a de vue cohérente de la topologie (à l'inverse de HBM). C'est en ce sens que nos solutions diffèrent.
La proposition TMESH [100] ajoute elle aussi des liens redondants (appelés court-circuits) mais l'objectif est ici surtout de réduire les latences de communication entre membres, un effet de bord étant bien sûr une augmentation de la robustesse. Mais ici aussi l'ajout de court-circuits se fait de façon aléatoire, aucun noeud n'ayant de vue cohérente de la topologie.
Dans ce chapitre nous suivons délibérément une approche déterministe, qui ajoute explicitement de la redondance à la topologie, de façon contrôlée et intelligente, ainsi qu'un mécanisme d'apprentissage par lequel les noeuds les moins fiables sont progressivement identifiés et la topologie créée adaptée à ces noeuds.

12.2 Ajout de liens virtuels redondants

12.2.1 Idées générales

Afin de garantir les propriétés de robustesse désirées, nous introduisons et comparons dans ce chapitre cinq algorithmes d'ajout de liens redondants (ou RVL). Ces algorithmes diffèrent sur la façon dont les liens RVLs sont ajoutés. Par exemple le premier algorithme ne fait aucune différence entre noeud de transit et feuille dans la topologie de recouvrement. A l'inverse les quatre autres algorithmes limitent d'une façon ou d'une autre le nombre de liens RVLs pouvant être attachés à un noeud feuille. Cette distinction est importante car les noeuds légers en terme de capacité de traitement (par exemple de type assistant personnel) ou connectivité réseau (par exemple des noeuds mobiles) sont toujours positionnés en périphérie de la topologie de recouvrement par le RP.
Les algorithmes sont les suivants :

Algorithme I: un nombre quelconque de liens RVL peut être attaché à tout noeud. Aucune différence n'est faite entre noeud de transit et feuille.

Algorithme II: il n'y a aucune limite sur le nombre de liens RVL pouvant être attachés à un noeud de transit, mais un lien RVL au plus peut être attaché à une feuille donnée. Un lien RVL peut lui être attaché à n'importe quel type de noeud.

Algorithme III: un nombre quelconque de liens RVL peut être attaché à tout noeud. Un lien RVL ne peut être attaché à deux feuilles. Seuls les liens RVL {feuille; noeud de transit} et {noeud de transit; noeud de transit} sont possibles.

Algorithme IV: il n'y a aucune limite sur le nombre de liens RVL pouvant être attachés à un noeud de transit, mais aucun lien RVL ne peut être attaché à une feuille. Seuls les liens RVL {noeud de transit; noeud de transit} sont possibles.

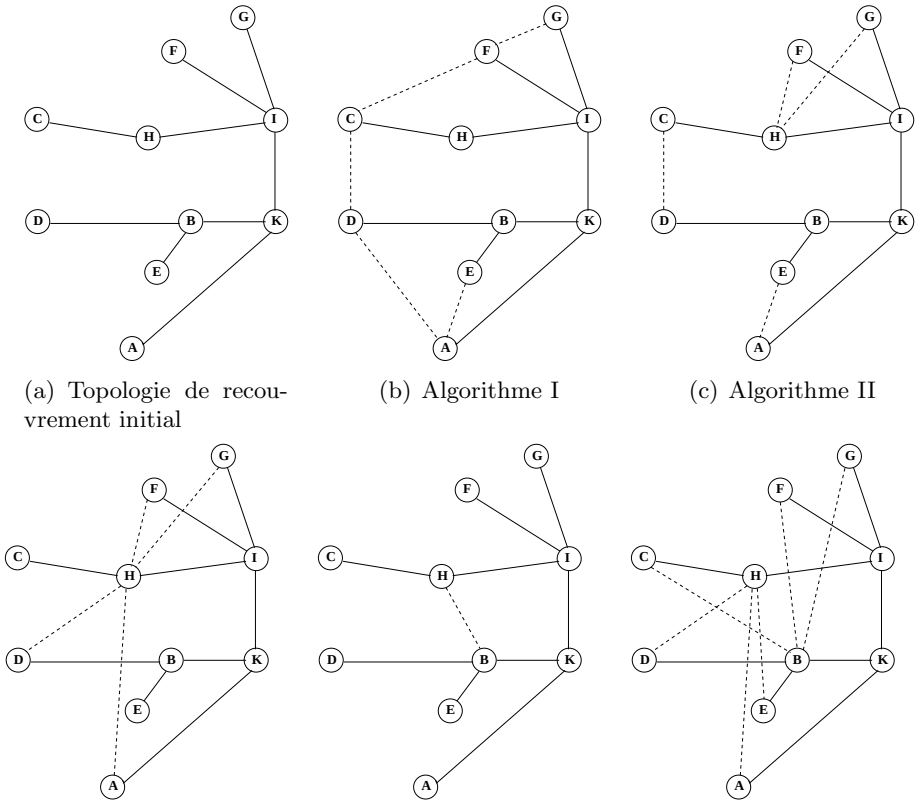
Algorithme V: il n'y a aucune limite sur le nombre de liens RVL pouvant être attachés à un noeud de transit, mais au plus un lien RVL peut être attaché à une feuille. Seuls les liens RVL {feuille; noeud de transit} sont possibles.

Ces stratégies suivent une approche récursive. Tous d'abord, nous cherchons les noeuds les plus éloignés de l'ensemble de noeuds et conformes à des conditions dépendant de l'algorithme.

Un lien RVL est alors créé entre eux, et l'ensemble est coupé en deux sous ensembles de noeuds, en fonction de leur proximité avec les deux noeuds choisis précédemment.
Le détail de cet algorithme, en pseudo-code, est donné dans la version anglaise de ce document.

12.2.2 Exemple

Considérons un groupe de dix noeuds, et supposons que la topologie de recouvrement initialement créée est celle de la figure 12.1-a.



(a) Topologie de recouvrement initial (b) Algorithme I (c) Algorithme II

(d) Algorithme III (e) Algorithme IV (f) Algorithme V

Figure 12.1. Ajout de liens RVL (lignes pointillées) sur un exemple.

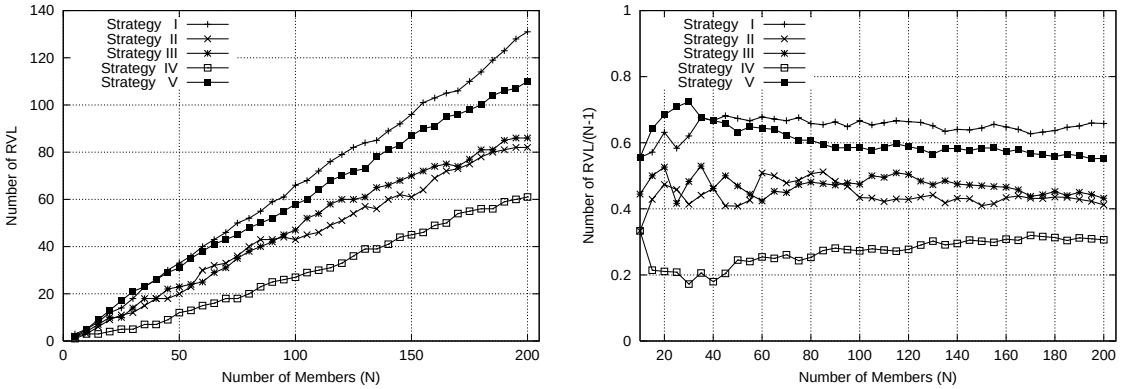
De cet exemple nous voyons que le nombre et la localisation des liens RVL différent largement suivant l'algorithme considéré. Par exemple, avec l'algorithme I, nous pouvons noter que (1) la vaste majorité des liens RVL sont créés entre des feuilles, et (2) à certaines feuilles sont attachés plusieurs liens RVL. Par conséquent l'algorithme I est typiquement réservé au cas où tous les noeuds sont similaires en termes de capacité de traitement et de connectivité réseau. A l'opposé l'algorithme IV conduit à la création d'un seul lien RVL, et les feuilles ne sont jamais concernées.

12.3 Evaluation de performances

12.3.1 Métriques considérées

Soit N le nombre total de noeuds. Nous considérons les métriques suivantes:

- nombre de liens RVL : N_{RVL}
- ratio du nombre de liens RVL sur le nombre initial de liens non-RVL de la topologie : $R_{RVL} = \frac{N_{RVL}}{N-1}$.
- nombre de noeuds connectés après i défaillances de noeuds, respectivement avec et sans liens RVL : $R_{conn-without}(i)$ and $R_{conn-with}(i)$.
- ratio du nombre de noeuds connectés après i défaillances de noeuds, sur le nombre total de noeuds, respectivement avec et sans liens RVL : $R_{conn-without/with}(i) = \frac{N_{conn-without/with}(i)}{N}$. Ce ratio est une moyenne sur toutes les sources possibles (en d'autres termes chaque noeud peut être source au sein de l'arbre partagé). Idéalement, i défaillances devraient laisser $N - i$ noeuds connectés, donc le ratio maximum est : $R_{conn-ideal}(i) = \frac{N-i}{N}$.
- l'augmentation relative (ou gain) du nombre de noeuds connectés après ajout de liens RVL, face à i défaillances : $G_{conn}(i) = \frac{N_{conn-with}(i) - N_{conn-without}(i)}{N_{conn-without}(i)}$
- stress des liens : c'est le nombre de copies identiques d'un paquet traversant un lien donné. Ce stress est évalué avec et sans liens RVL.

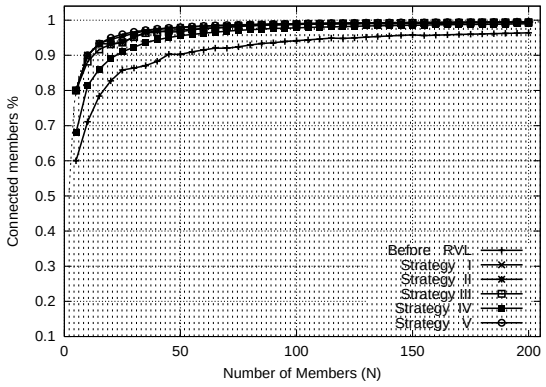


(a) Nombre de liens RVLs (b) Ration R_{RVL}

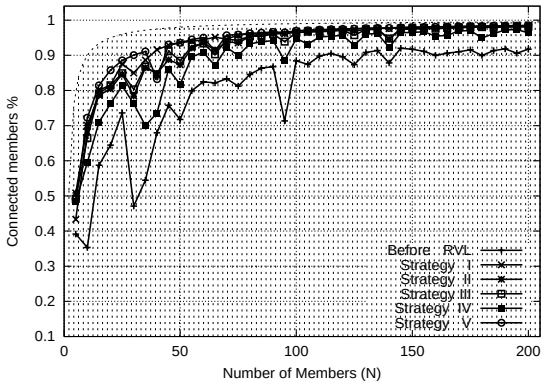
Figure 12.2. Addition de RVLs.

12.3.2 Résultats et discussion

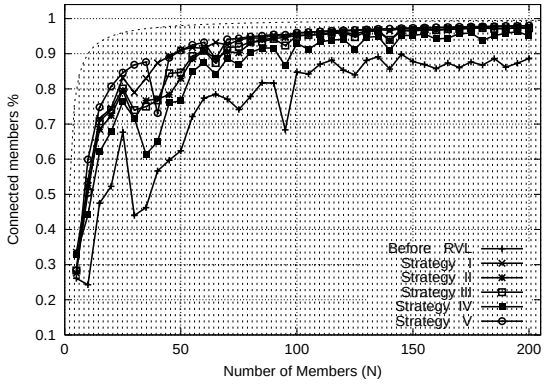
Les expériences rapportées ici sont des simulations basées sur un réseau d'interconnection composé de 600 routeurs de coeur, généré par l'outil GT-ITM [102]. Certains de ces routeurs sont considérés comme routeurs d'interconnection, d'autres, en périphérie de la topologie physique, sont des routeurs d'accès connectés aux sites clients. Nous choisissons donc N sites aléatoirement parmi les 243 feuilles possibles, et comparons chaque stratégie.
La figure 12.2-a représente le nombre de liens RVLs en fonction du nombre de noeuds N. Nous notons que le nombre de liens RVL augmente avec N pour toutes les stratégies, mais avec une pente différente.
La figure 12.2-b représente le ratio R_{RVL} associé. Nous voyons que l'algorithme IV ajoute le plus petit nombre de liens RVL, alors que l'algorithme I ajoute le plus grand nombre de liens RVL.



(a) Défaillance d'un seul noeud



(b) Défaillance de deux noeuds



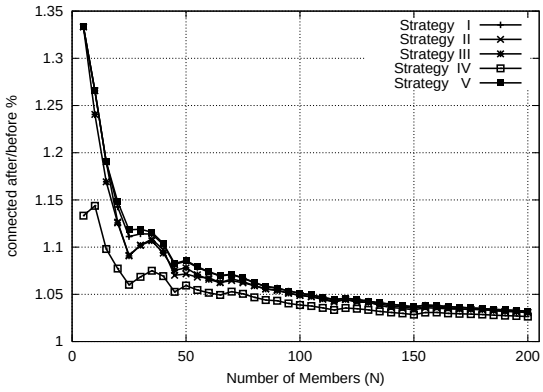
(c) Défaillance de trois noeuds

Figure 12.3. Ratio du nombre de noeuds connectés après 1, 2 ou 3 défaillances de noeuds, en fonction de l'algorithme.

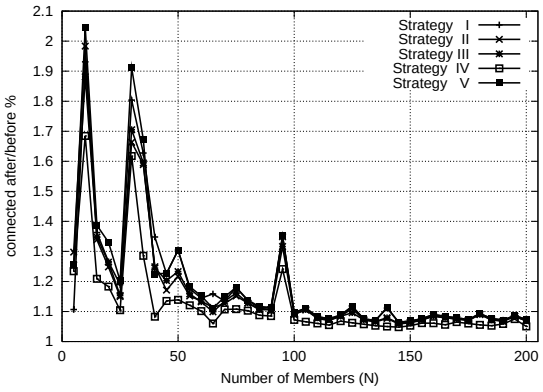
Les figures 12.3-a/b/c représentent le ratio du nombre de noeuds restant connectés après 1, 2 ou 3 défaillances de noeuds. La partie supérieure de la zone hachurée représente le ratio optimal $R_{conn-ideal}(i) = \frac{N-i}{N}$. Si tous les algorithmes d'ajout de liens RVL améliorent la connectivité, nous voyons que des différences existent. Sans lien RVL le ratio de noeuds connectés s'élève à 60% pour 5 noeuds et 96% pour 200 noeuds après une simple défaillance. Avec l'algorithme IV ces ratios sont respectivement de 68% et 99%. Avec les stratégies I, II, III et V, les ratios sont de 80% et 99% , ce qui est proche de l'idéal (80% et 99.5%). Nous notons aussi que pour deux et trois défaillances, tous les algorithmes offrent un bénéfice, même si les résultats sont maintenant plus éloignés de l'idéal.

Les figures 12.4-a/b/c montrent l'augmentation relative du nombre de noeuds connectés, $G_{conn}(i)$. Nous notons que (1) ce gain diminue lorsque N augmente (grands groupes), et (2) ce gain augmente avec le nombre de défaillances.

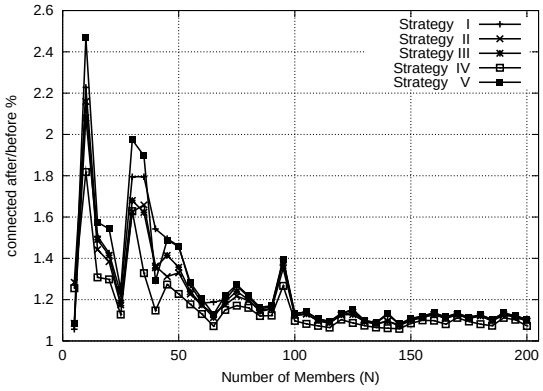
Finalement la figure 12.5 montre le stress avant et après l'ajout de liens RVL. Ainsi que nous nous y attendons, le stress augmente avec l'ajout des liens RVL. Cependant l'algorithme IV a le plus petit stress (car il a aussi le plus petit nombre de liens RVL ajoutés), alors que la stratégie V ajoute un trafic prohibitif (c'est aussi l'algorithme qui ajoute le plus grand nombre de liens RVL).



(a) Défaillance d'un seul noeud



(b) Défaillance de deux noeuds



(c) Défaillance de trois noeuds

Figure 12.4. Augmentation relative du nombre de noeuds connectés après 1, 2 ou 3 défaillances, en fonction de l'algorithme.

12.4 Conclusions

Pour conclure nous pouvons dire que *l'algorithme IV offre un bon compromis entre la robustesse face à des défaillances brutales de noeuds, et l'ajout de trafic au sein du réseau. En outres le trafic supplémentaire est géré par les noeuds de transit, jamais par les feuilles, ces dernières pouvant avoir de plus faibles capacités de traitement ou connectivité réseau puisque ce critère est considéré lors de la création de la topologie.*

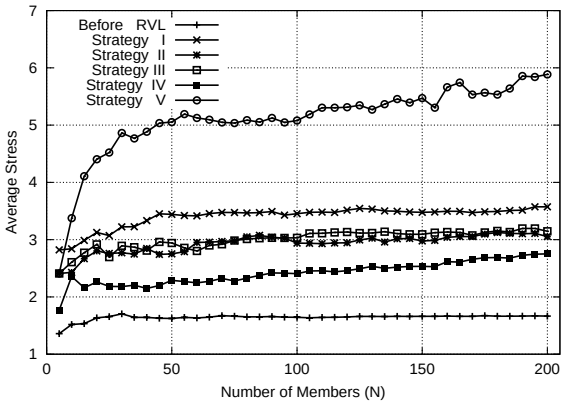


Figure 12.5. Stress moyen des liens physiques.

Chapitre 13

Discussion, conclusions, et travaux futurs

Contents

Nous avons dans ce travail décrit une grande variété de propositions offrant un service de communication de groupe alternatif (chapitre d'état de l'art), et analysé plusieurs facettes de notre proposition, HBM. Nous discutons maintenant plusieurs aspects clef qui ont été soulevés (implicitement ou explicitement), et les classons en ordre d'importance décroissant

(cet ordre pouvant être revu en fonction des objectifs précis poursuivis par l'application). Pour chaque point nous identifions nos contributions et les travaux futurs possibles lorsque ceci est pertinent.

13.1 Facilité de déploiement

13.1.1 Discussion

Une technique de multicast applicatif doit être aisément déployable afin d'offrir une alternative viable au service de routage multicast natif défaillant. Bien souvent il sera suffisant de spécifier les coordonnées d'un serveur pour initier la session. Notons que la présence d'équipements faisant de la translation d'adresse (NAT) ou de pare-feux peut présenter des problèmes non triviaux.

13.1.2 Contributions

HBM a été implanté sous forme d'une librairie qui peut soit être intégrée à une application, soit utilisée de façon indépendante afin de créer une passerelle locale unicast/multicast. Les seules informations nécessaires afin d'initier la sessions sont les coordonnées (port/adresse) du RP et du groupe.

13.2 Robustesse

13.2.1 Discussion

Le routage multicast inter-domaines est souvent vu comme étant fragile. Si les approches multicast applicatif offrent un moyen de contourner ces problèmes, en revanche elles induisent d'autres problèmes d'instabilité. Ainsi une solution basée sur des machines terminales (PC bien souvent) est intrinsèquement plus fragile qu'une solution basée sur des équipements dédiés de routage, soigneusement administrés. Il y a un fort risque de partition de la topologie de recouvrement à mesure que la taille du groupe augmente. Certaines propositions adressent ce problème de façon pro-active, en utilisant un certain niveau d'inondation (ainsi les approches pair-à-pair de type bavardage).
Offrir une robustesse nécessite aussi que les noeuds défaillant soient rapidement découverts. Cette propriété dépend de la topologie utilisée (arbre, anneau, etc.) : ainsi sur une arbre de type "plus court chemin", des messages d'acquittement peuvent être générés par les feuilles et agrégés par les noeuds de transit lors de leur parcourt en direction de la source. Un noeud de transit ne recevant pas d'acquittement de l'un de ses voisins en dessous de lui dans l'arbre peut facilement en déduire qu'un problème est survenu.
Plusieurs propositions se contentent de proposer des mécanismes de découverte et de réparation. Disposer de telles fonctions est certes nécessaire, mais à notre avis ce n'est pas suffisant. Par exemple certaines applications peuvent nécessiter que le maximum soit fait pour éviter tout risque de partition.

13.2.2 Contributions

Cette thèse s'est focalisée sur (1) le problème de partition de la topologie de recouvrement suite à des défaillances (brutales) de noeuds, et (2) au problème de pertes de paquets du fait des incohérences transitoires lorsque cette topologie est mise à jour.

Pour résoudre le premier problème nous avons proposé et évalué plusieurs algorithmes d'ajout de liens redondants, ou RVLs. Nous en avons conclu qu'ajouter des liens entre des noeuds de transit seuls (algorithme IV) est une bonne solution. Nous avons également montré que cette solution apportait certains bénéfices face à plusieurs défaillances simultanées de noeuds. Ajouter des liens RVL a aussi l'avantage de (1) détecter plus rapidement des défaillances (ainsi si un noeud reçoit un paquet de l'un seulement de ses liens normaux et RVL), et (2) de réduire la latence de communication sur la topologie de recouvrement.
Puisque la mise à jour de la topologie de recouvrement n'est pas synchrone, à certains moments un sous ensemble des noeuds peuvent être informés de la nouvelle topologie (et l'appliquer) tandis que les autres en sont encore à l'ancienne topologie. Cette incohérence transitoire conduit bien sûr à des pertes de paquets en transit. Nous les évitons avec HBM en permettant aux noeuds de mémoriser deux topologies, la courante et la précédente, et en routant chaque paquet en fonction de la topologie sur laquelle il a été diffusé (pour cela un identifiant de topologie a été ajouté aux en-têtes de paquets). Cette solution, qui a été comparée avec trois autres algorithmes, s'est révélée terriblement efficace puisqu'elle évite toute perte (ou presque). De plus elle est triviale à mettre en oeuvre, et la gestion centralisée d'HBM permet aisément de gérer ces identifiants globaux de topologie.

13.2.3 Travaux futurs

Les travaux futurs peuvent considérer l'utilisation de plusieurs RPs miroirs, comme une approche complémentaire de robustification d'HBM. Cependant reste à voir à quel moment cela est bénéfique et quelles en sont les limites.

13.3 Impacts des tricheurs

13.3.1 Discussion

Les approches multicast applicatif offrent la possibilité à certains noeuds (les tricheurs) d'améliorer leur positionnement sur la topologie de recouvrement, en manipulant par exemple les mesures de distance. Ils pourront ainsi chercher à être positionnés plus près de la source (ce qui améliore les latences de communication) et avoir moins de noeuds fils (afin de limiter le trafic qu'ils génèrent).
Si nous considérons l'utilisation du RTT en tant que métrique, une solution simple utilisée dans de nombreuses propositions (y compris HBM), un tricheur peut aisément retarder sa réponse à un message ICMP Echo Request généré par un noeud autre que la source, pour augmenter artificiellement sa distance avec lui. Cela pourra avoir comme effet de réduire la probabilité d'avoir ces noeuds comme voisins, et donc le nombre de fils sous le tricheur dans la topologie (du moins c'est son intention). Un tricheur va également annoncer une distance extrêmement faible à la source, voir changer l'estampille temporelle contenue dans le message ICMP Echo Request de la source avec une valeur dans le futur. Le but recherché est ici de donner l'illusion qu'il est très proche de la source, et donc qu'il mérite d'être positionné très près de celle-ci. Bien évidemment, des variantes plus élaborées de la façon de mentir sur sa distance avec autrui sont possibles, afin peut être d'éviter d'être trop facilement repérable (annoncer une distance nulle à la source est d'emblée suspect!).
Il est important de noter que ces tricheurs n'essaient pas de perturber autrement les flux de données (ce ne sont pas des attaques par déni de service), ils s'efforcent simplement d'améliorer leur position dans l'arbre. En d'autres termes, les tricheurs suivent tous les mécanismes imposés par le protocole de multicast applicatif, hormis l'évaluation des distances.

Ces problèmes de comportements de triche, liés à l'égoisme de certains membres, sont critiques car ils peuvent totalement perturber le bon fonctionnement des approches multicast applicatif. De surcroît, un utilisateur est incité à tricher, du fait de l'espoir d'en obtenir un bénéfice (même si nous verrons que c'est loin d'être gagné).

13.3.2 Contributions

Dans nos tests, un tricheur HBM annonce toujours au RP une distance minimale à la source et une grande distance au reste du groupe. Parce que les autres noeuds, y compris la source, évaluent périodiquement leur distance aux tricheur, par exemple au moyen de mesures de RTT, les tricheurs essaient également de modifier ces mesures de façon appropriée. Ceci est obligatoire, sinon le RP pourrait facilement identifier les tricheurs en comparant les mesures $A \hookleftarrow B$ et $B \hookleftarrow A$. Si elles diffèrent significativement, le RP pourrait alors en conclure que l'un des deux noeuds, A ou B, est suspect, et une vérification croisée des métriques impliquant ces deux noeuds pourrait alors permettre de déterminer le tricheur.
Au moyen de cette technique de triche triviale, un unique tricheur a la garantie de toujours être attaché directement à la source et de n'avoir aucun fils. Ce tricheur a donc une position de feuille, avec la source comme source, au sein de la topologie de recouvrement créée par le RP.
Cependant, lorsque le nombre de tricheurs augmente et si le degré maximum de chaque noeud de l'arbre est borne, alors la situation devient très différente. En fait, avec HBM, seul un petit nombre d'entre eux va bénéficier d'un avantage (en étant attaché directement à la source), alors que les autres vont se trouver à l'extrême périphérie de la topologie, après les noeuds honnêtes ! Cela montre que, pour une fois, tricher n'a pas nécessairement l'effet recherché. Dans [68], nous discutons des impactes des tricheurs sur différentes approches multicast applicatif, dont HBM. Nous montrons que la présence d'un nombre croissant de tricheurs a un effet toujours assez étrange sur la topologie créée.

13.3.3 Travaux futurs

Les travaux futurs sont nombreux, la notion de triche n'ayant jamais été analysée avec les approches multicast applicatif. Il s'agira donc d'étudier comment rendre HBM (et d'autres approches) robustes à la triche.

13.4 Sécurité

13.4.1 Discussion

Un aspect souvent oublié dans les approches multicast applicatif, est la sécurité. Le service fourni par HBM n'offre aucun mécanisme de sécurité (ainsi aucune authentification des noeud n'est gérée), ni n'est lui même sécurisé (ainsi les mécanismes de contrôle ne sont pas sécurisés). Ceci est paradoxale puisque la plupart des propositions sont basées sur des communications point à point, soit entre deux noeuds, soit entre un noeud et le RP. Or sécuriser des connexions point à point est bien plus simple que de sécuriser des communications multicast.

13.4.2 Contributions

Dans [2] nous avons expliqué comment fournir un service de communication de groupe au sein d'un environnement de réseaux privés virtuels (ou VPN, Virtual Private Network) totalement

sécurisé. Dans cette approche des tunnels IPSec sont créés dynamiquement entre les sites qui hébergent des membres du groupe, et supprimés lorsque ceux-ci quittent le groupe. Ce travail montre que les approches de VPN IPSec et HBM sont parfaitement complémentaires. Nous montrons que ces deux approches, qui suivent chacune une approche centralisée, conduisent naturellement au concept de VPRN, ou réseau privé virtuel routé, qui améliore largement l'efficacité du service de diffusion. Il est intéressant de noter que les approches multicast applicatives centralisées, telle HBM, souvent critiquées du fait de leur faible passage à l'échelle, trouvent un parfait champ d'application dans notre environnement de VPNs. Les bénéfices de sécurité apportée par l'architecture compensent alors largement les limites pouvant exister en terme de passage à l'échelle notamment.

13.4.3 Travaux futurs

Les services d'authentification, autorisation, chiffrement, intégrité, confidentialité, contrôle d'accès, non répudiation, robustesse face à des attaques par déni de service, sont des points clef qui devraient systématiquement être considérés. Nos travaux futurs étudierons ces aspects.

13.5 Performances

13.5.1 Discussion

Les performances de la plupart des approches de multicast applicatif sont sans surprise moindre que celles des protocoles de routage multicast natifs. Effectuer le relayage de trafic sur des noeuds terminaux, ou dans un nombre limité d'hôtes (par exemple dans le cas d'un réflecteur multicast), est nécessairement moins efficace que d'utiliser des routeurs idéalement placés dans le réseau de coeur.
Mais les performances ne sont pas nécessairement l'objectif premier de ces techniques multicast applicatif. Ainsi créer des liens redondants conduit nécessairement à des performances moindres, du fait du trafic supplémentaires, mais ajoute de la robustesse ce qui dans certains environnements est plus important. De même les solutions basées sur des réflecteurs attachent plus d'importance à l'aisance avec laquelle on peut déployer la solution que sur ses performances.
A l'opposé, les propositions créant des topologies de recouvrement sont d'avantage concernées avec la création de "bonnes" topologies. Plusieurs aspects vont alors déterminer les performances :

- le type de topologie créée : un arbre "de plus court chemin" par source est indéniablement plus efficace qu'un arbre unique partagé par différentes sources. Mais gérer plusieurs arbres a également un coût.

- possibilité d'adaptation dynamique de la topologie : la topologie doit refléter au mieux les conditions réseaux, aussi est-il nécessaire de monitorer le réseau. Un surveillance passive est parfois possible, en tirant profit des statistiques de réception de données, sinon des techniques de surveillance actives sont requises, ce qui ajoute un overhead. Enfin, puisque la mise à jour de la topologie a un coût important, sa fréquence de mise à jour est nécessairement limitée.

- métriques considérées : plusieurs travaux ont considérés les délais de communication en supposant les chemins symétriques, ce qui autorise alors l'utilisation d'outils simples tels `ping`. Or [22] rappelle que les situations de routage asymétriques ne sont pas rares

sur l'Internet. Finalement [51] prétend que les métriques délai de communication et bande passante doivent toutes deux être considérées.

- personnalisation par noeud : les noeuds peuvent largement différer et les simples métriques réseaux ne peuvent à elles seules capturer tous ces aspects. Ainsi les noeuds les plus légers ne devraient pas devenir noeuds de transit. De la même façon un historique de tous les noeuds devrait être conservé afin que les plus instables d'entre eux (par exemple du fait d'une connection sans fil) soient des feuilles. Les approches multicast applicatives centralisées telle HBM, du fait de leur base de données, peuvent facilement en tenir compte lors du processus de création de la topologie.

Le prix à payer pour de meilleures performances sont donc : (1) une plus grande complexité algorithmique, et (2) un coût de signalisation plus élevé. Il y a clairement un compromis à trouver entre les bénéfices de performance attendu et les coûts associés.

13.5.2 Contributions

Les aspects performance brute n'ont pas été activement étudiés dans cette thèse. Cependant nous avons introduit des idées d'adaptation dynamique de la topologie, ainsi que d'une personnalisation de cette topologie aux caractéristique de chaque noeud qui peuvent largement contribuer à obtenir de meilleures performances.

13.6 Passage à l'échelle

13.6.1 Discussion

Si le passage à l'échelle en terme de taille de groupe est typiquement un objectif majeur des protocoles de routage multicast traditionnels, ce n'est pas une nécessité dans toutes les situations. En conséquence, certaines des approches multicast applicatif (telles les réflecteurs et HBM) sont délibérément conçues pour gérer des groupes de taille "raisonnable", ce qui est suffisant dans nombre de situations. De meilleurs mécanismes de contrôle et une architecture simplifiée compensent alors largement le manque de passage à l'échelle.
D'autres propositions en revanche ciblent nettement un fort passage à l'échelle, ce qui peut être requis pour des systèmes pair à pair en particulier. Ce passage à l'échelle est habituellement obtenu au moyen d'une hiérarchisation de la topologie et d'une connaissance distribuée et partielle du groupe et de la topologie.
Puisque le service multicast est souvent disponible au sein d'un LAN ou d'un site, une hypothèse souvent faite est que le multicast applicatif n'est utilisé qu'entre sites et non au sein d'un site. Un représentant situé dans chaque site peut alors rediffuser via le multicast le trafic reçu. Le passage à l'échelle global est alors largement supérieur puisque de nombreux clients finaux peuvent se cacher derrière chaque représentant.
Dans certains cas, le passage à l'échelle n'est pas en terme de nombre de membres du groupe, mais en terme de nombre de groupes gérés pas le système, ce qui est un problème tout autre. Ce type de passage à l'échelle peut être critique pour déployer certains services et protocoles: ainsi il existe des propositions pour améliorer le hand-off de Mobile IP au moyen de l'approche XCAST, ce qui a un sens dans des environnements de types "IP cellulaire" où un très grand nombre de noeuds mobiles existeraient.
Finalement l'idée de partager des arbres entre plusieurs groupes fortement corrélés [36] peut elle aussi être appliquée aux approches multicast applicatif. Par exemple une session de travail coopératif est typiquement composée de plusieurs outils de type audio, vidéo, discussion, et

tableau blanc, chacun d'eux rassemblant les mêmes membres, ou presque. Partager alors une unique topologie de recouvrement permettrait alors de significativement réduire les coûts de contrôle.

13.6.2 Contributions dans cette thèse

Dans cette thèse nous avons étudié le passage à l'échelle de notre proposition, même si nous ne considérons pas ce point comme essentiel. Après une modélisation détaillée de notre protocole HBM, nous avons expliqué comment en améliorer le passage à l'échelle en limitant la taille et le nombre de messages de contrôle échangés entre le RP et le membres. Des divers algorithmes étudiés nous en avons identifié un qui semblait offrir un meilleur compromis.

13.6.3 Travaux futurs

Nous avons considéré uniquement le critère de la charge liée au trafic de contrôle lorsque nous avons étudié le passage à l'échelle. C'est notoirement insuffisant et la charge (CPU) du RP, et le nombre de connexions TCP gérées par le RP devraient également être pris en compte. Il peut être intéressant de ce point de vue de considérer la possibilité d'avoir plusieurs RPs, les différents membres étant répartis entre eux. Le premier contact se ferait alors via le RP principal, celui-ci redirigeant le nouveau membre vers l'un des RPs secondaires. Des échanges entre RPs secondaires et RP principal permettraient alors une synchronisation indispensable par exemple à la création de la topologie.
Un autre aspect qui nous intéresse particulièrement est l'utilisation d'une unique topologie de recouvrement, avec quelques variations mineures, pour gérer plusieurs groupes dont les membres sont essentiellement les mêmes. Cela a un sens lors de sessions de travail coopératif, l'ensemble des membres joignant souvent les mêmes outils (ou un sous-ensemble de ceux-ci, par exemple si un membre se passe de vidéo du fait d'un connectivité réseau insuffisante).

13.7 Quelques mots pour finir...

Le multicast applicatif a suscité un fort intérêt durant ces dernières années. En dépit du grand nombre de propositions ayant été faites, plusieurs aspects essentiels, de nature à en empêcher le déploiement, n'ont toujours pas trouvé de réponse satisfaisante. Nos travaux récents sur la triche sont de ce point de vue révélateurs. Et sauf erreur, personne n'a étudié les aspects dénis de service qui inévitablement verront le jour. Plus généralement ces propositions, qui ont été considérées au départ comme des rustines rapidement applicables afin de palier aux défectuosités du service de routage multicast natif, tendent à être considérées de plus en plus comme des techniques de communication de groupe à part entière. Nos travaux dans cette thèse, et les remarques que nous avons faites dans cette discussion, nous conduisent à être plus réservés en ce qui concerne leur futur. Un travail énorme reste à faire sur le sujet. Mais des exemples d'application réalistes existent, telle notre intégration de HBM dans un environnement de VPN. Il est hautement probable, lorsque nous considérons la matrice $besoins\ applicatifs * classes\ de\ problemes$, que de nombreuses propositions continueront d'être proposées. Nous avons adressé dans ce travail seulement un sous ensemble de ces dernières. Notre proposition n'est ni meilleure, ni pire que les autres, elle adresse seulement un ensemble différent de besoins applicatifs et de problèmes de la meilleure façon qui soit.

Bibliography

[1] *NICE working group home page.* http://www.math.princeton.edu/tsp/.

[2] Lina Al-Chaal, Vincent Roca, Ayman El-Sayed, and Michel Habert. A vprn solution for fully secure and efficient group communications. In *IEEE Symposium on Computers and Communications - ISCC'2003, Kemer-Antalya, Turkey*, July 2003.

[3] Lina Al-Chaal, Vincent Roca, Ayman El-Sayed, and Michel Habert. A vprn solution for fully secure and efficient group communications. In *Research Report number RR-4799, INRIA, Rhône Alpes, France*, April 2003.

[4] Lina Al-Chaal, Vincent Roca, and Michel Habert. Offering a multicast delivery service in a programmable secure ip vpn environment. In *Fourth International Workshop on Networked Group Communication (NGC 2002), Boston, USA*, October 2002.

[5] Alcatel. *XCAST working group home page.* http://www.alcatel.com/xcast.

[6] Kevin C. Almeroth. The evolution of multicast: From the mbone to interdomain multicast to internet2 deployment. *IEEE Network*, January 2000.

[7] Sang Ho Bae, Sung-Ju Lee, William Su, and Mario Gerla. The design, implementation, and performance evaluation of the on-demand multicast routing protocol in multihop wireless networks. *IEEE Network*, January 2000.

[8] T. Ballardie. Core based trees (cbt version 2) multicast routing. *RFC 2189*, September 1997.

[9] S. Banerjee and B. Bhattacharjee. A comparative study of application layer multicast protocols, 2002.

[10] Suman Banerjee and Bobby Bhattacharjee. Analysis of the nice application layer multicast protocol. In *Technical report UMIACS-TR2002-60 and CS-TR 4380, Departement of Computer Science, University of Maryland, College Park*, June 2002.

[11] Suman Banerjee, Bobby Bhattacharjee, and Christopher Kommareddy. Scalable application layer multicast. In *Proceedings of ACM SIGCOMM'02, Pittsburgh, USA*, August 2002.

[12] Suman Banerjee, Bobby Bhattacharjee, and Christopher Kommareddy. Scalable peer finding on the internet. In *In IEEE Global Internet Symposium (Globecom'02), Taipei International Convention Center, Taipei, Taiwan,*, November 2002.

[13] Suman Banerjee, Seungjoon Lee, Bobby Bhattacharjee, and Aravind Srinivasan. Resilient multicast using overlays. In *IEEE/ACM SIGMETRICS Performance Evaluation Review*, volume 31, pages 102–113, June 2003.

[14] Ljubica Blazevic and Jean-Yves Le Boudec. Distributed core multicast (dcm): a multicast routing protocol for many groups with few receivers. *ACM SIGCOMM Computer Communication Review*, 29(5), October 1999.

[15] Ljubica Blazevic and Jean-Yves Le Boudec. Distributed core multicast (dcm): a routing protocol for many small groups with application to mobile ip telephony. *Work in Progress; <draft-blazevic-dcm-mobility-00.txt>*, June 1999.

[16] R. Boivie, N. Feldman, Y. Imai Fujitsu, W. Livens, Dirk Ooms, and Olivier Paridaens. Explicit multicast (xcast) basic specification. *Work in Progress; <draft-ooms-xcast-basic-spec-03.txt>*, June 2002.

[17] D. Box, D. Ehnebuske, G. Kakivaya, A. Layman, N. Mendelsohn, H. F. Nielsen, S. Thatte, and D. Winer. *Simple Object Access Protocol (SOAP) 1.1, May 2000. W3C Note*. http://www.w3.org/TR/SOAP/.

[18] Miguel Castro, Peter Druschel, Anne-Marie Kermarrec, and Antony Rowstron. Scribe: a large-scale and decentralised application-level multicast infrastructure. *IEEE Journal on Selected Areas in Communications (JSAC)(Special issue on Network Support for Multicast Communications)*, 20(8), October 2002.

[19] Yatin Chawathe. Scattercast: An architecture for internet broadcast distribution as an infrastructure service. *Ph.D thesis at University of California at Berkeley*, September 2000.

[20] Tarik Cicic and Haakon Bryhni. *Multicast-unicast reflector*, January 2000.

[21] Jeremy De Clercq and Olivier Paridaens. Scalability implications of virtual private networks. *IEEE Communications Magazine*, 40(5), May 2002.

[22] Luis Henrique M. K. Costa, Serge Fdida, and Otto Carlos M. B. Duarte. Hop by hop multicast routing protocol. In *ACM SIGCOMM'01, San Diego, USA*, August 2001.

[23] S. Deering, W. Fenner, and B. Haberman. Multicast listener discovery (mld) for ipv6. *RFC 2710*, October 1999.

[24] Stephen Deering. Host extensions for ip multicasting. *RFC 1112*, August 1989.

[25] Stephen Deering. Multicasting routing in a datagram internetwork. In *Ph.D. dissertation*, 1991.

[26] Hrishikesh Deshpande, Mayank Bawa, and Hector Garcia-Molina. Streaming live media over peers. In *Stanford University, Database Group, Submitted for publication*, 2002.

[27] Christophe Diot, Brian Neil Levine, Bryan Lyles, Hassan Kassem, and Doug Balensiefen. Deployment issues for the ip multicast service and architecture. *IEEE Network*, pages 78–88, January 2000.

[28] Allen B. Downey. Using pathchar to estimate internet link characteristics. *Proceedings of ACM SIGCOMM*, March 1999.

[29] K. Egevang and P. Francis. The ip network address translator (nat). *RFC 1631*, May 1994.

[30] Ayman El-Sayed. Group communication service library (gcsl) implementation. *Implemented in C++ under linux, INRIA Rhone Alpes - FRANCE*, October 2001.

[31] Ayman El-Sayed and Vincent Roca. Improving the scalability of an application-level group communication protocol. In *10th Int. Conference on Telecommunications (ICT'03), Papeete, French Polynesia*, February 2003.

[32] Ayman El-Sayed, Vincent Roca, and Laurent Mathy. A survey of proposals for an alternative group communication service. *IEEE Network, special issue on Multicasting: an enableing technology*, January/February 2003.

[33] D. Estrin et al. Protocol independent multicast-sprse mode(pim-sm):protocol specification. *RFC 2362*, June 1998.

[34] D. Farinacci et al. Multicast source discovery protocol (msdp). *Internet Draft, Internet Engineering Task Force*, June 1998.

[35] Stephen Deering et al. The pim architecture for wide-area multicast routing. In *IEEE/ACM Transaction on Network*, volume 4, pages 153–162, April 1996.

[36] A. Fei, J. Cui, M. Gerla, and M. Faloutsos. Aggregated multicast with inter-group tree sharing. In *Third International Workshop on Networked Group Communication (NGC 2001), London, UK*, November 2001.

[37] B. Fenner. *Mrouted home page*. http://www.research.att.com/ fenner/.

[38] R. Finlayson. Ip multicast and firewalls. *RFC 2588*, May 1999.

[39] Ross Finlayson. The udp multicast tunneling protocol. *Work in Progress: <draft-finlayson-umtp-07.txt>*, September 2002.

[40] Paul. Francis. Yoid distribution protocol (ydp) specification. *Unrefered Report; http://www.yallcast.com*, April 2000.

[41] Paul Francis. Yoid: extending the internet multicast architecture. *Unrefered Report; http://www.yallcast.com*, April 2000.

[42] Paul Francis. Yoid identification protocol (yidp) specification. *Unrefered Report; http://www.yallcast.com*, April 2000.

[43] Paul Francis. Yoid tree management protocol (ytmp) specification. *Unrefered Report; http://www.yallcast.com*, April 2000.

[44] Paul Francis, Sugih Jamin, Cheng Jin, Yixin Jin, Danny Raz, Yuval Shavitt, and Lixia Zhang. Idmaps: A global internet host distance estimation service. *ACM/IEEE Transaction on Networking*, 9(5), October 2001.

[45] Paul Francis, Sugih Jamin, Vern Paxson, Lixia Zhang, Daniel F. Gryniewicz, and Yixin Jin. An architecture for a global internet host distance estimation service. *Proceedings of IEEE INFOCOM'99*, March 1999.

[46] Atanu Ghosh, Michael Fry, and Jon Crowcroft. An architecture for application layer routing. In *2nd Int. Working Conf. on Active Networks (IWAN2000)*, October 2000.

[47] B. Gleeson, A. Lin, J. Heinanen, G. Armitage, and A. Malis. A framework for ip based virtual private networks. In *IETF Request for Comments, RFC 2764,*, February 2000.

[48] Omer Gurewitz and Moshe Sidi. Estimating one-way delays from cyclic-path delay measurements. *Proceedings of IEEE INFOCOM'01*, April 2001.

[49] D. Harkins and D.Carrel. The internet key exchange (ike). In *IETF Request for Comments, RFC 2409,*, November 1998.

[50] H. Holbrook and B. Cain. Source-specific multicast for ip. *Work in Progress: <draft-holbrook-ssm-arch-02.txt>*, March 2001.

[51] Yang hua Chawathe, Sanjoy G. Rao, Srinirasan Sechan, and Hui Zhang. Enabling conferencing applications on the internet using an overlay multicast architecture. In *ACM SIGCOMM'01, San Diego, USA*, August 2001.

[52] Yang hua Chawathe, Sanjoy G. Rao, and Hui Zhang. A case for end system multicast. In *ACM SIGMETRICS*, June 2000.

[53] V. Jacobson and S. McCanne. A bsd packet filter: a new architecture for user-level packet capture. In *Usenix Winter Conference, San Diego, USA*, January 1993.

[54] Van Jacobson. *Pathchar*. ftp://ftp.ee.lbl.gov/pathchar/, 1997.

[55] J. Jannotti, D. Gifford, K. Johnson, F. Kaashoek, and J. O'Toole. Overcast: Reliable multicasting with an overlay network. In *USENIX OSDI*, October 2000.

[56] S. Kent and R. Atkinson. The internet key vexchange (iksecurity architecture for the internet protocol. In *IETF Request for Comments, RFC 2401,*, November 1998.

[57] D. Kosiur. *Building and Managing Virtual Private Networks*. John Wiley and Sons Inc., ISBN 0-471-29526-4, 1998.

[58] Kevin Lai and Mary Baker. Measuring bandwidth. *Proceedings of IEEE INFOCOM'99*, March 1999.

[59] Kevin Lai and Mary Baker. Measuring link bandwidths using a deterministic model of packet delay. In *Proceedings of ACM SIGCOMM 2000*, August 2000.

[60] Kevin Lai and Mary Baker. Nettimer: A tool for measuring bottleneck link bandwidth. In *Proceedings of the USENIX Symposium on Internet Technologies and Systems*, March 2001.

[61] S-J. Lee, W. Su, and M Gerla. On-demand multicast routing protocol (odmrp) for ad hoc networks. *Work in Progress: <draft-ietf-manet-odmrp-01.txt>*, June 1999.

[62] Seungjoon Lee, Rob Sherwood, and Bobby Bhattacharjee. Cooperative peer groups in nice. *IEEE INFOCOM'03*, April 2003.

[63] Sung-Ju Lee, William Su, Julian Hsu, Mario Gerla, and Rajive Bagrodia. A performance comparison study of ad hoc wireless multicast protocols. *IEEE INFOCOM'00*, March 2000.

[64] J. Liebeherr, M. Nahas, and W. Si. Application-layer multicast with delaunay triangulations. In *IEEE Global Internet Symposium (Globecom'01), also Technical Report CS-2001-26*, November 2001.

[65] Jorg Liebeherr, Michael Nahas, and Weisheng Si. Application-layer multicasting with delaunay triangulation overlays. *IEEE Journal on Selected Areas in Communications*, 20(8), October 2002.

[66] Mingyan Liu, Rajesh Talpade, Anthony McAuley, and Ethendranath Bommaiah. Amroute: Adhoc multicast routing protocol. Technical Report TR 99-8, University of Maryland and the Institute for Systems Research, Department of Defense (DOD), CSHCN, 1999.

[67] Bruce A. Mah. *Pchar*. http://www.ca.sandia.gov/ bmah/Software/pchar/, 2000.

[68] Laurent Mathy, Nick Blundell, Vincent Roca, and Ayman El-Sayed. On cheats in application-level multicast. *IEEE INFOCOM'04, Hong Kong*, March 2004.

[69] Laurent Mathy, Roberto Canonico, and David Hutchison. An overlay tree building control protocol. In *Third International Workshop on Networked Group Communication (NGC 2001), London, UK*, November 2001.

[70] Laurent Mathy, Roberto Canonico, Steven Simpson, and David Hutchison. Scalable adaptive hierarchical clustering. *IEEE Communication Lettters*, 6(3):117–119, March 2002.

[71] J. Moy. Multicast routing extensions for ospf. In *Communication ACM*, volume 37, August 1994.

[72] J. Moy. Ospf version 2. *RFC 2328*, April 1998.

[73] Dirk Ooms and Jeremy De Clercq. *Overview of Multicast in VPNs*, February 2002. Work in Progress: <draft-ooms-ppvpn-mcast-overview-00.txt>.

[74] FreeS/Wan org. *FreeS/Wan project home page: an open-source implementation of IPSEC and IKE for Linux*. http://www.freeswan.org.

[75] P. Parnes, K. Synnes, and D. Schefstrom. Lightweight application level multicast tunneling using mtunnel. *Computer Communications*, 21(15):1295–1301, April 1998.

[76] Sanjoy Paul. *Multicasting on the Internet and its Applications*. Baston, MA: Kluwer, 1998.

[77] Dimitrios Pendarakis, Sherlia Shi, Dinesh Verma, and Marcel Waldvogel. ALMI: An application level multicast infrastructure. In *Third USNIX Symposium on Internet Technologies and Systems (USITS '01)*, March 2001.

[78] C. G. Plaxton, R. Rajaraman, and A. W. Richa. Accessing nearby copies of replicated objects in a distributed environment. In *Proceedings of ACM Symposium on Parallel Algorithms and Architectures*, June 1997.

[79] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Second Edition, Cambridge University Press, 1992.

[80] T. Pusateri. Distance vector multicast routing protocol. *Work in Progress: <draft-ietf-idmr-dvmrp-v3-10.txt>*, February 2001.

[81] Shelley Q.Zhuang, Ben Y.Zhao, Anthony D. Joseph, Randy H. Katz, and John D. Kubiatowicz. Bayeux: An architecture for scalable and fault-tolerant wide-area data dissemination. In *11th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV 2001)*, June 2001.

[82] Sylvia Ratnasamy, P. Francis, Mark Handley, Richard Karp, and Scott Shenker. A scalable content-addressable network. In *Proceedings of ACM SIGCOMM 2001*, August 2001.

[83] Sylvia Ratnasamy, Mark Handley, R ichardKarp, and Scott Shenker. Application-level multicast using content-addressable networks. In *Third International Workshop on Networked Group Communication (NGC 2001), London, UK*, November 2001.

[84] Y. Rekhter, B. Moskowitz, D. Karrenberg, and E. Lear. Address allocation for private internets. *RFC 1918*, February 1996.

[85] Christoph Rensing, Martin Karsten, and Burkhard Stiller. Aaa: A survey and a policy-based architecture and framework. *IEEE Network*, November/December 2002.

[86] Vincent Roca and Ayman El-Sayed. A host-based multicast (hbm) solution for group communications. In *First IEEE International Conference on Networking (ICN'01), Colmar, France*, pages 610–619, July 2001.

[87] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *In IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, November 2001.

[88] Antony Rowstron, Anne-Marie Kermarrec, Miguel Castro, and Peter Druschel. Scribe: The desgin of a large-scale event notification infrastracture. In *In proceeding of the third International COST264 Workshop, Networked Group Communication (NGC 2001), London, UK*, November 2001.

[89] E. Royer and C. Perkins. Multicast operation of the ad-hoc on-demand distance vector routing protocol. In *MobiCom'99, Seattle, USA*, August 1999.

[90] E. Royer and C. Perkins. Multicast ad hoc on-demand distance vector (maodv) routing. *Work in Progress: <draft-ietf-manet-maodv-00.txt>*, February 2001.

[91] Laxman H. Sahasrabuddhe and Biswanath Mukherjee. Multicast routing algorithms and protocols: A tutorial. *IEEE Network*, January 2000.

[92] Yunxi Sherlia Shi. Design of overlay networks for internet multicast. *Ph.D Thesis, Departtement of Computer Science, Sever Institute of Technology, Washington University*, August 2002.

[93] Ion Stoica, Tze Sing Eugene Ng, and Hui Zhang. Reunite: A recursive unicast approach to multicast. *IEEE INFOCOM'00*, March 2000.

[94] Aaron Striegel and G. Manimaran. A survey of qos multicasting issues. In *IEEE Communications Magazine*, pages 82–87, June 2002.

[95] Tan Su-Wei and Gill Waters. Building low delay application layer multicast trees. *IEEE INFOCOM'03*, April 2003.

[96] D. Thaler, D. Estrin, and D Meyer. Border gateway multicast protocol (bgmp): Protocol specification. *Internet Draft, Internet Engineering Task Force*, August 1998.

[97] Dave Thaler, Mohit Talwar, Lorenzo Vicisano, and Dirk Ooms. Ipv4 automatic multicast without explicit tunnels (amt). *Work in Progress: <draft-ietf-mboned-auto-multicast-01.txt>*, April 2002.

[98] Duc A. Tran, Kien A. Hua, and Tai T. Do. Zigzag: An efficient peer-to-peer scheme for media streaming. *IEEE INFOCOM'03*, April 2003.

[99] TSP. *TSP working group home page*. http://www.cs.umd.edu/projects/nice/.

[100] Wenjie Wang, David Helder, Sugih Jamin, and Lixia Zhang. Overlay optimizations for end-host multicast. In *In proceeding of the fourth International Workshop on Networked Group Communication (NGC 2002), Boston, USA*, October 2002.

[101] Lidia Yamamoto and Guy Leduc. Autonomous multicast reflectors over active networks. In *Symposium on Software mobility and adaptive behaviour (AISB'01 Convention)*, March 2001.

[102] E. Zegura, K. Calvert, and S. Bhattacharjee. How to model an internetwork. In *Proceedings of IEEE INFOCOM'96*, March 1996.

[103] Baoxian Zhang and Hussein T. Mouftah. Forwarding state scalability for multicast provisioning in ip networks. In *IEEE Communications Magazine*, pages 46–51, June 2003.

[104] Beichuan Zhang, Sugih Jamin, and Lixia Zhang. Host multicast: a framework for delivering multicast to end users. *IEEE INFOCOM'02, New York, USA*, June 2002.

[105] B. Y. Zhao, J. D. Kubiatowicz, and A. D. Joseph. Tapestry: An infrastructure for fault-tolerant wide-area location and routing. In *Technical report UCB/CSD-01-1141, University of California at Berkeley, Computer Science Division*, April 2001.