Development and experimentation towards a multicast-enabled Internet

Davide Careglio[1], Dimitri Papadimitriou[2], Fernando Agraz[1], Sahel Sahhaf[3], Jordi Perelló[1], Wouter Tavernier[3], Salvatore Spadaro[1], Didier Colle[3]

[1] Universitat Politècnica de Catalunya, Barcelona, Spain
[2] Alcatel-Lucent, Antwerp, Belgium
[3] iMinds, Ghent, Belgium

Abstract.
Today, the Internet is a large distributed and dynamic system consisting of more than 50,000 Autonomous Systems and 500,000 IPv4 address prefixes. To convince industry and regulators, any new protocol/scheme/mechanism needs to be executed in a large-scale experimental platform before its deployment. In this paper, we report our development experience and experimentation studies of two multicast routing schemes for the Internet, namely, PIM-SSM and GCMR. We detail their implementation over the Quagga open source routing suite, as well as their experimentation tests over a large-scale topology that reproduces the Internet characteristics.

1. Introduction and motivation

The Internet is a large, dynamic, heterogeneous collection of interconnected systems that can be used for communicating between any interested parties connected to it, no matter their location. The Internet architecture relies on a few design principles: modularization by layering, connectionless forwarding (no virtual circuit), the internetworking principle (gateways), and the end-to-end principle. It results in a transparent network that ensures for applications to survive partial network failures, and provides for a general connectivity capable of supporting many different applications. It shall be mentioned that the Internet was not optimized for any single application, but aiming at generality and evolvability. The network merely forwards packets, whereas knowledge of the application is localized at the edges, where hosts are attached. This functional decomposition facilitates innovation and the deployment of new applications.

Although the current Internet does work and is still capable of fulfilling its current missions, it suffers from a relative *ossification* [1], a condition where any technological innovation meets natural resistance, as exemplified by the lack of wide-scale deployment of technologies such as multicast or IPv6. Recent

initiatives, like the Global Environment for Network Innovations[1] (GENI), a project sponsored by the National Science Foundation (NSF), and the Future Internet Research and Experimentation[2] (FIRE) in Europe, try to overcome this ossification problem by bridging the gap between visionary research and technology deployment. The rationale behind them is to promote the experimental validation of new proposals in large-scale realistic network environments, as a way to convince both industry and regulators that a new technology deserves to be adopted and thus enable a possible migration path for technological developments.

Multicast becomes an excellent example of the Internet ossification. Originally defined in the 90's, its potential benefits have been verified by studies several times since then. However, only intra-domain multicast has been partially adopted in the context of IPTV in some Internet Service Provider (ISP) networks, while inter-domain multicast is still pending to date. Among other reasons, this failure could be attributed to the scaling limitations and relative complexity of the standard multicast protocol architecture, based on overlaying multicast routing on top of the unicast routing topology [2]. For this very reason, we recently proposed the Greedy Compact Multicast Routing (GCMR) scheme [3]. GMCR is characterized by its independence from any underlying unicast routing topology; more specifically, the local knowledge of the cost to direct neighbor nodes is enough for the GCMR scheme to properly operate.

In this paper, we present a prototype of the GCMR multicast scheme and evaluate its functionality and performance on the iLab.t virtual wall[3] platform, which is a large-scale experimental Linux machine-based emulation testbed. The prototype of a GCMR routing engine has been developed using the libraries of the Quagga open source routing suite[4]. The success in our endeavor, which was presented during the Hands-on-FIRE! Demonstration (collocated in the 2013 FIA Week in Dublin, Ireland), suggests a feasible multicast-enabled Internet.

The rest of the paper is organized as follows. Section 2 presents a brief overview of the main issues of the current Internet routing system and explains the state of the art in multicast routing. Section 3 introduces the details of the routing software implementation while Section 4 describes the testbed platform. In Section 5, we provide and discuss the results. Finally, Section 6 concludes the paper.

---

[1] http://www.geni.net

[2] http://www.ict-fire.eu/

[3] http://www.iminds.be/en/develop-test/ilab-t/virtual-wall

[4] http://www.nongnu.org/quagga/

2. Brief overview of routing and multicast routing in Internet

The most fundamental issues faced by the Internet architecture are the scalability, convergence, and stability properties of its inter-domain routing system [4]. Solving them requires to address multiple dimensions together: i) the routing table size growth resulting from a larger number of routing entries, and ii) the routing system dynamics characterized by the routing information exchanges resulting from topological or policy changes. Worst-case projections predict that routing engines could have to process and maintain in the order of 1 million active routes within the next 5 years[5]. Thus, while the Internet routing system prevents from any host specific routing information processing and maintenance (routing state), storing an increasingly large amount of network states in the routing system is expensive and places undue cost burdens on network administrative units. For example, realizing the vision of the Internet of Things, with its corresponding increase on the number of routes, would require significantly more efficient and scalable routing schemes. Furthermore, those impacts on the routing system dynamics (robustness/stability and convergence) resulting from inconsistencies (software implementation errors, router misconfigurations, etc.), instabilities (interactions between routing policies), and topological changes/failures are progressively becoming key concerns for the Internet operational community.

These issues are even more evident if we consider multicast routing system. By multicast routing, we refer to a distributed algorithm that, given a group identifier, allows any node to route multicast traffic to a group of destination nodes, usually called multicast group. To enable one-to-many traffic distribution, the multicast routing protocol configures the involved routers to build a (logical) delivery tree between the source and the multicast group, commonly referred to as the Multicast Distribution Tree (MDT). Multicast is (re-)gaining interest given the increasing popularity of multimedia streaming/content traffic and the explosion of cloud services, since it yields bandwidth savings competing with or complementing cached content distribution techniques. Nevertheless, the scaling problems faced in the 90's still remain mostly unaddressed. Although routing protocol independent routing schemes such as Protocol Independent Multicast Sparse Mode (PIM-SM) [5] and Core Base Trees (CBT) [6] have been standardized during last decade, only the Single Source Multicast (SSM) variant of PIM (PIM-SSM) [7] has been deployed in the context of IPTV systems for routing multicast streams between VLANs, subnets or access networks (intra-domain multicast). However, the adoption of inter-domain multicast has failed, as it relies on an overlay routing executing on top of the unicast routing topology, which suffers from the same scaling limits as unicast routing plus the following issues: i) the level of indirection added by the multicast routing as routers forward multicast datagrams to multicast group, and hosts have to subscribe to that multicast group; ii) the limits of shared trees between domains; iii) its

[5] http://bgp.potaroo.net/index-bgp.html

address space structure as firewalls have to be upgraded to recognize multicast addresses. A complete analysis of the deployment issues for the IP multicast routing architecture can be found in [2].

In this paper, we experimentally evaluate and compare the performance of the PIM-SSM protocol and the GCMR protocol initially proposed in [3] in the context of inter-domain multicast routing. In the following subsections, we describe PIM-SSM and highlight its limitations, which help us to motivate the proposal of GCMR.

2.1 The protocol independent multicast (PIM) scheme

PIM [5] is nowadays the most common multicast routing protocols for IP networks that provide one-to-many and many-to-many traffic data distribution. Its protocol-independence comes from the fact that PIM does not perform any network-wide topology discovery mechanism but instead uses routes learned from any unicast routing protocol to build the Multicast Routing Information Base (MRIB), perform the Reverse Path Forwarding (RPF) check, and forward the multicast packets that a router receives from a source. In the context of inter-domain routing, Border Gateway Protocol (BGP) has been enhanced with the Multiprotocol BGP (MBGP) extensions [8] to support and distribute IPv6 and multicast addresses.

In PIM-SSM [7], the delivery of traffic data is supported on (S,G) channels. A (S,G) channel supports data from the IP unicast source address S to the multicast group address G as the IP destination address. Receivers must subscribe to the (S,G) channel to receive traffic from the specific source S. In other terms, a (S,G) channel is the term used in PIM to indicate a MDT. However, applications are responsible of channel discovery. As the PIM scope is limited to routers, the Internet Group Management Protocol (IGMP) in IPv4 or Multicast Listener Discovery (MLD) in IPv6 needs to be used by hosts (receivers and source) to convey channel subscriptions to local routers. Once a router receives a subscription request from a receiver, it configures the local forwarding table accordingly, and sends the request upstream towards the source address based on its knowledge of the unicast topology. At each hop, routers configure the multicast forwarding table (usually referred as Tree Information Base, TIB) and become members of the MDT (i.e., configuring an entry in the MRIB). At the end of this process, the MDT is built from hosts towards the source, which is considered a shortest-path tree (SPT) from the perspective of the unicast routing tables.

2.2 The Greedy Compact Multicast Routing (GCMR) scheme

As part of the work conducted in the EULER FP7-project[6], we designed the GCMR scheme [3]. Its main objective is to minimize (i.e., to compact) the routing table size at each router by taking local (i.e., greedy) decisions at expenses of i) routing packets on paths with relative small deviation compared to the optimal tree; ii) increasing the number of messages required to create the MDT. In this way, GCMR can reduce the local storage of routing information by keeping only (direct) neighbor-related entries, rather than tree structures or network graph entries. In other terms, the novelty of this algorithm is on maintaining local topology information instead of global one, thus only providing the least cost to next hop during the MDT construction.

In the GCMR context, the information needed to reach a given multicast source S is acquired by means of a search mechanism that returns the upstream node along the least cost branching path to the MDT sourced at S. The algorithm search process is segmented in two different stages, namely, an initial local search covering the receiver neighborhood (defined by a path budget), and, if unsuccessful, a subsequent global search over the remaining topology. The rationale behind this approach is to put tighter limits and search locally first. Indeed, the likelihood of finding any router that belongs to the targeted MDT within a few hops distance from the joining receiver is high in large topologies (whose diameter is logarithmically proportional to its number of nodes) while also increasing with the size of the MDT. Such a search mechanism is triggered whenever a node decides to join a given multicast source address S as part of a multicast group G. Once a node becomes member of an MDT, a multicast routing entry is dynamically created and stored in the local MRIB. From these routing table entries, multicast forwarding entries are also derived and stored in the local TIB. Only two types of messages are needed to execute the search process, namely, request (R) and answer (A), whereas a couple of additional messages deal with the logical join (J) and detach (D) of the nodes to/from the MDT. A detailed description of the GCMR algorithm can be found in [9], theoretical performance analysis and simulation comparison with other major multicast routing paradigms are documented in [10].

As GCMR does not rely on any unicast routing protocol, it can work together with any addressing scheme like IPv4, IPv6 or even geometric coordinates over a Euclidian plane or hyperbolic plane [11]. In addition, GCMR can be implemented directly in any host, as its scope is not limited to routers, not requiring any host-router protocol like IGMP. Therefore, to the authors' knowledge, it is the first name-independent, receiver-initiated, dynamic, distributed, end-to-end multicast routing algorithm.

---

[6] http://www.euler-fire-project.eu

Figure 1a shows the different protocols involved in a typical PIM-SSM scenario and their scope, that is, MBGP for addresses discovery and IGMP for multicast membership subscription between hosts and local routers. In contrast, Figure 1.b highlights GCMR as a real end-to-end multicast protocol over the Internet.

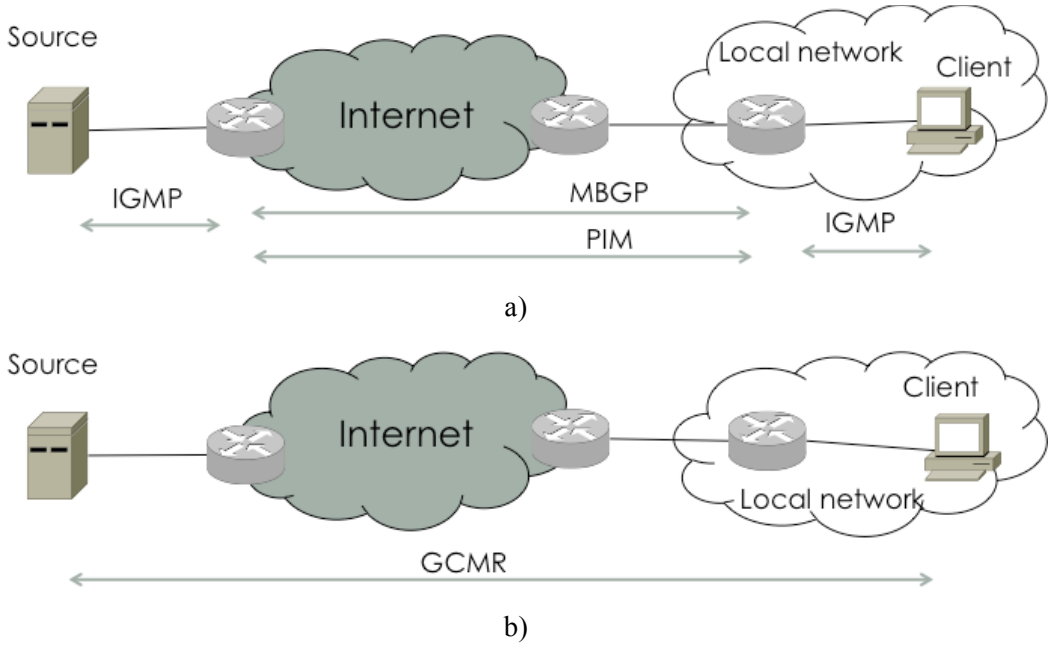


Figure 1. Protocols architecture a) using PIM-SSM, b) using GCMR

3. Routing platform

As the aim of this work is to experimentally validate GCMR against PIM-SSM, both protocols have been implemented and executed in a prototype. This prototype runs on top of an existing Quagga routing platform (described in this section) in the iLab.t virtual wall emulation testbed platform (Section 4).

3.1 Quagga routing suite

Few open source routing platforms exist that allow rapid introduction of new protocols, features, and functionality. Most popular ones are Quagga, XORP[7] and Bird[8], which can run on standard PC hardware. Among them, we have used Quagga given its widespread adoption and maturity. Quagga benefits from a large developer community including independent code committers, service provides, and academic institutions, while Bird still presents limitations regarding the inter-domain routing support and XORP is

[7] http://www.xorp.org

[8] http://bird.network.cz

neither mature enough nor widely used for research purposes. Moreover, Quagga starts becoming the common reference platform for software-defined routers running in production environments.

More specifically, Quagga is an open-source routing protocol suite providing implementations of different IP protocols such as OSPF and BGP. The software is developed in standard C programming language and is available for UNIX platforms like Linux, Solaris and BSD.

The Quagga architecture consists of a core module (i.e., the Zebra daemon), which acts as an abstraction layer to the UNIX kernel packet forwarding functionality. The Zebra daemon provides a set of client modules, called Zserv, implementing each a specific routing protocol. Furthermore, Zebra provides an Application Programming Interface (API), called ZAPI, through which routing protocol modules can access and communicate routing updates to the kernel routing table and network interfaces. During bootstrap, the Zebra daemon must be started first, followed by the routing protocol daemons that should operate in the network. In this way, Zebra is able to interface the communication between the routing daemons and the kernel. For configuration, all daemons in Quagga are equipped with a command-line interface (called VTY), which follows a CISCO OS-like syntax. Alternatively, a pre-defined configuration file can also be used.

It is worth mentioning that neither Quagga nor Bird or even XORP completely emulate a router. They only provide the route engine (algorithms and protocols), thus still requiring a forwarding engine to transmit datagrams. In the following sections, we describe the implementation of GCMR and PIM-SSM as well as the forwarding engines used in our prototype.

3.2 GCMR implementation

The different modules composing the architecture of a GCMR-capable node are depicted in Figure 2a. The system architecture can be logically divided into two main parts: the Routing Core (RC) and the Routing communication Protocol (RP) module. The former aggregates the set of procedures that a GCMR-capable node must implement in order to carry out its functionalities. The latter module is composed of a set of objects and messages that enables the communication between GCMR nodes and, thus, their collaborative operation.

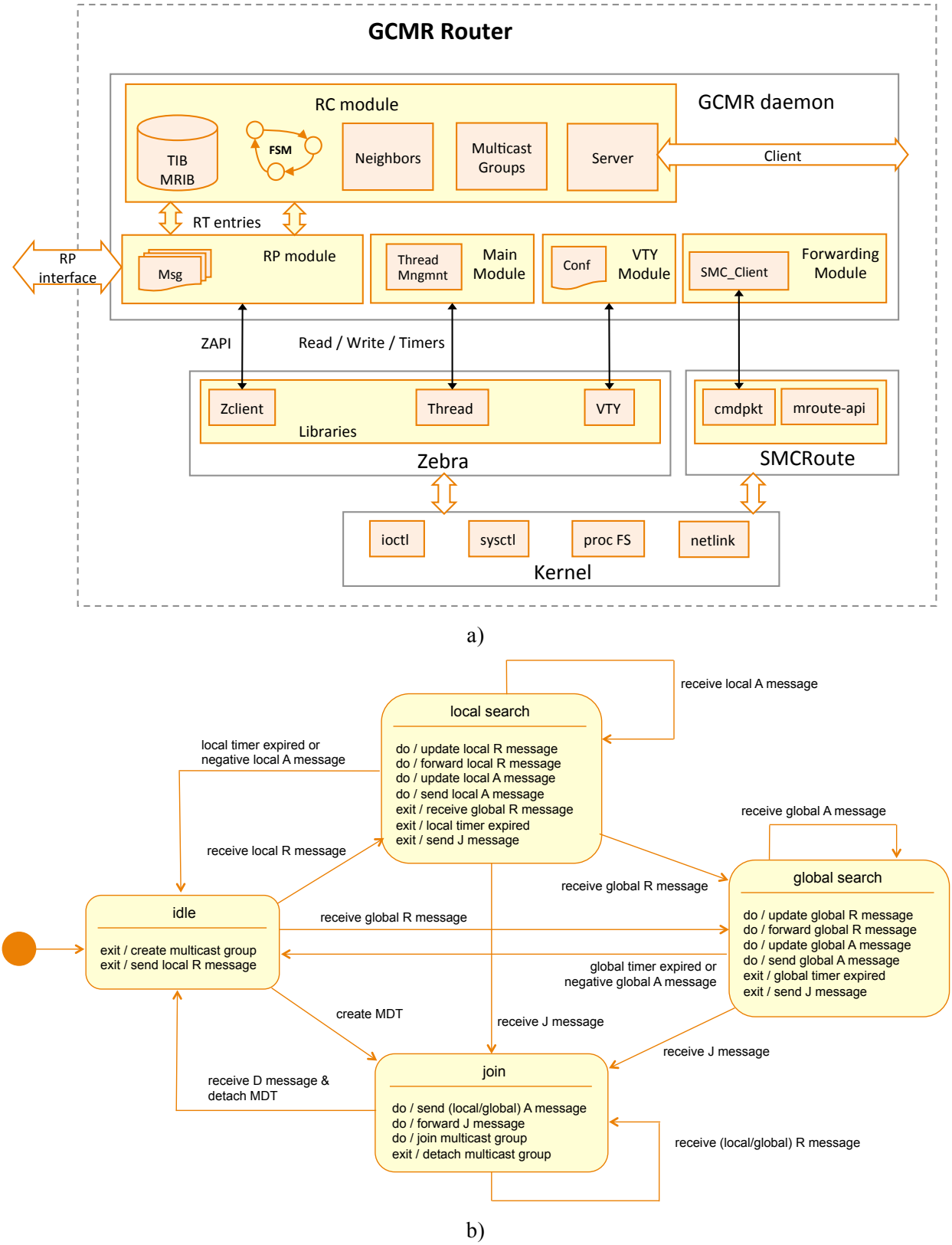


Figure 2. a) GCMR node architecture, b) GMCR state machine

The RC module contains the main objects that are involved in the GCMR operation. The database contains the information related to the TIB and the MRIB. This module has also two interfaces: one to the RP module and the other enables the communication between the node and an external client (e.g.,

the application) to trigger the GCMR functionalities, namely create a new multicast group, join the node to a new multicast group and detach the node from an existing multicast group.

The RP module implements the GCMR protocol primitives enabling communication with neighboring nodes and routing operation. Specifically, the protocol relies on two search message types (request R and answer A), which enable the collaborative MDT computation. It also implements two other messages (J and D) aimed at the physical join/detach of the nodes to/from the MDT. Through the ZAPI interface, RP communicates with the Zebra daemon in order to configure the interfaces when the node is added or removed to/from a multicast group.

As previously documented, Quagga does not provide any native forwarding engine. For this reason, we have developed an SMCRouteClient that automatically generates multicast route add/remove commands to the SMCRoute daemon[9]. SMCRoute is a command line tool that manipulates dynamic multicast routes directly in the Linux kernel.

Finally, the Finite State Machine (FSM) implementation defines the sequence of actions to be followed by the GCMR engine during each MDT computation. Its structure is depicted in Figure 2b. Four states are defined:

1. *idle* where GCMR initializes all resources and structures;
2. *local search* when a node sends a local R message to its neighbors or receives it from a neighbor;
3. *global search* when local search fails, a node moves to the global search;
4. *join* when a node receives a J message and remains in this state as long as a D message is received. In this case, the detach procedure is executed and the node is removed from MDT and returns to the idle state.

3.3 PIM-SSM implementation

An implementation of the PIM-SSM routing protocol, called qPIM[10], is available as an external daemon for Quagga routing suite. The architectural view of this PIM-capable node is depicted in Figure 3. The qPIM daemon consists of two modules, namely the Objects module that constructs and maintains the set of data structures and the objects and the Protocols module that implements the communication protocols.

---

[9] http://www.cschill.de/smcroute/

[10] http://www.nongnu.org/qpimd/

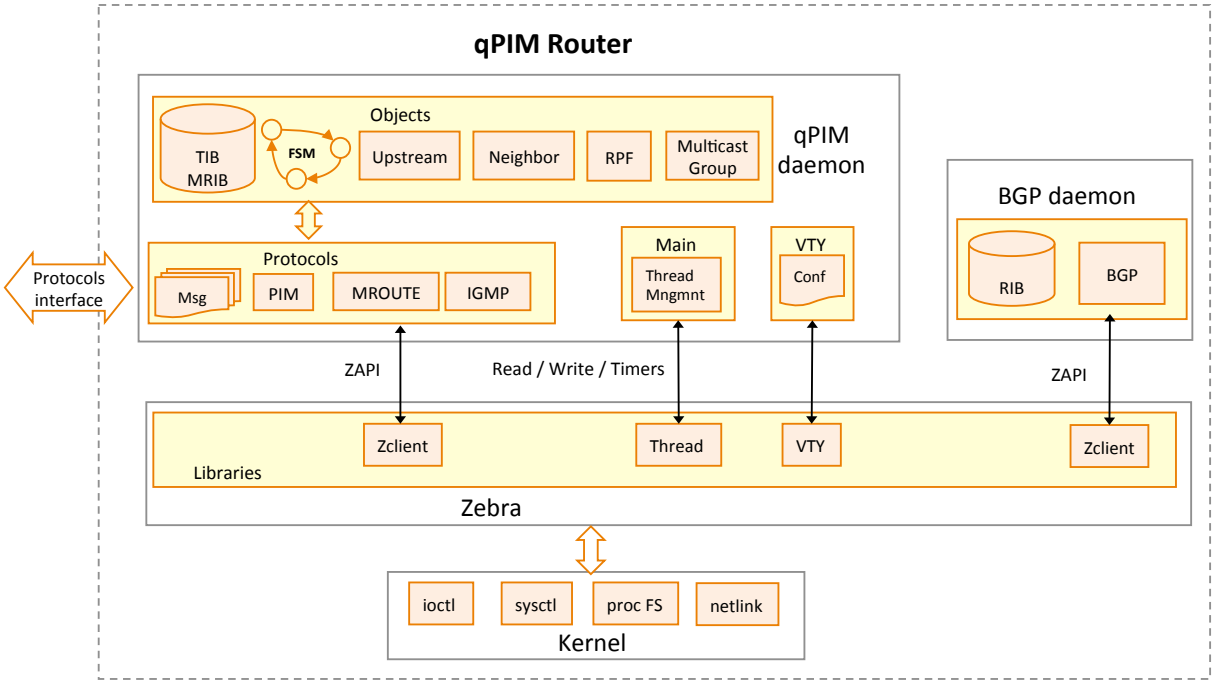


Figure 3. qPIM node architecture

As described in Section 2.1, PIM-SSM requires two additional protocols to operate. BGP daemon (with multiprotocol extensions) of Quagga is used as unicast routing protocol to populate the routing table, from which PIM-SSM derives the MRIB entries during the MDT construction. The IGMP protocol is used to interact hosts with routers. As a forwarding engine, qPIM uses the conventional ipmroute daemon of the Linux kernel to derive the TIB entries from the MRIB and generate the add/remove commands to configure the network interfaces accordingly.

4. iLab.t virtual wall platform

4.1 Overview

The iLab.t virtual wall is a generic test environment, which provides computing hardware and different software and hardware tools to researchers to validate and evaluate the performance of innovative network software prototypes. Each of the 3 virtual wall facilities of iLab.t consists of 100 server blades interconnected by a non-blocking 1.5 Tb/s VLAN Ethernet switch (Force 10 E1200). The specification of each node is as follows: dual processor, dual core server with 4GB RAM and 4x 80GB hard disk with 6x1 Gb/s or 4x1 Gb/s interfaces. Moreover, a control interface is provided in each node, which enables researchers to login.

The Emulab[11] control software is run on the virtual wall facilities and enables researchers to create arbitrary network topologies where each network node can be configured with a custom operating system and network software prototype. Realistic network scenarios can be evaluated using network traffic generator software such as iperf[12] or D-ITG[13].

4.2 Tools

Traditional mapping between Linux machines and nodes is one-to-one thus the emulation of network topologies is limited to the number of physical machines. On the iLab.t virtual wall, virtualization technique using OpenVZ[14] Linux containers allows for multiple virtual nodes to run on one machine enabling large-scale experiments (10-20 times the number of physical machines in the iLab.t). These virtual nodes run on Fedora15 Linux distribution. Using OpenVZ, isolation of the filesystem, process, network, and account namespaces are provided which result in to separate filesystem, process hierarchy, network interfaces and IP addresses for each virtual node. In addition, arbitrary number of virtual network links is provided by using virtual network interfaces. These links may be individually shaped and may be multiplexed over physical links or can be used to connect to virtual nodes within one physical node.

The virtual nodes behave like normal physical nodes in Emulab, and therefore can act as: end node, router or traffic generator. Virtual nodes can be accessed using ssh. The remote access through the control system enables building large dedicated experimental setup very fast. These setups are repeatable and different experimental parameters can be controlled and evaluated. Many performance metrics such computational complexity and convergence time of the routing algorithm can be measured and compared with the resulting values of a reference routing scheme. Arbitrary topologies can be generated and even a mixture of virtual and physical nodes is possible. Figure 4 illustrates the virtual wall testbed and emulation platform setup.

Different factors affect the number of virtual nodes that can be mapped on a physical machine. The two most important ones are i) the resource requirement of the application and ii) the bandwidth of the links which are emulated. Although, the Emulab resource mapper maps the virtual nodes onto physical nodes in a way to have the best overall use of physical resources, it is possible to determine the number of virtual nodes on a single physical machine and perform the mapping as desired.

---

[11] http://www.emulab.net

[12] http://code.google.com/p/iperf/

[13] http://traffic.comics.unina.it/software/ITG/

[14] http://openvz.org

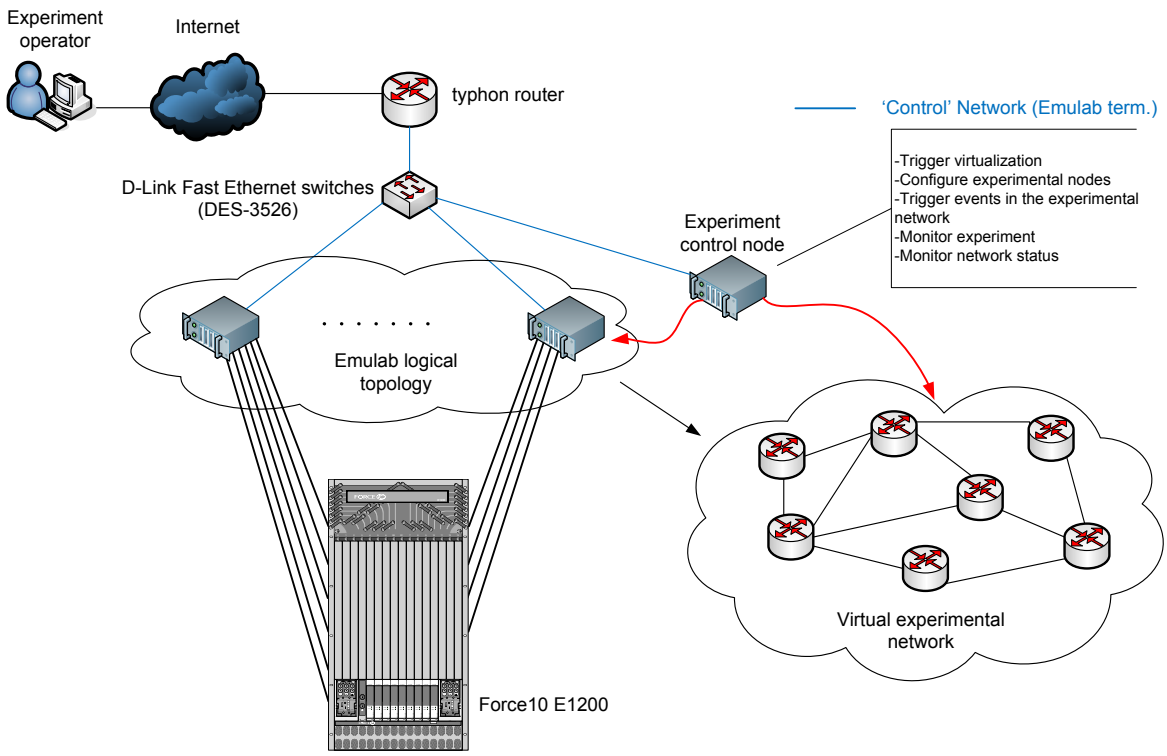


Figure 4. Emulation platform setup on virtual wall testbed

4.3 Implementation

To experimentally validate and compare PIM-SSM and GCMR, we used around 45 server blades of one of the iLab.t virtual wall facility. Each blade has been virtualized into approximately 5 virtual machines in order to reach a topology of more than 200 nodes. In each node, we installed a Debian 6 Linux distribution (kernel 2.6.32) and Quagga 0.99.17. Both PIM-SSM and GCMR were then automatically deployed in the nodes and properly configured using Phyton and bash scripts. Information of the nodes status was continuously stored in log files and processed afterward to obtain the performance results.

5. Experimentation and discussion of the results

5.1 Experimentation setup and objectives

The main objective of these tests is to demonstrate the successful operation of GCMR in the context of inter-domain routing over a large-scale network topology compared to the standard PIM-SSM protocol. In particular, the following performance metrics are evaluated:

- *stretch*, defined as the sum of the weights of edges used in multicasting from the source to all receivers divided by the optimal such tree. Intuitively, the stretch of a routing scheme provides a

quality measure of the path cost increase it produces compared to the optimal tree (which has clearly a stretch of 1). The optimal MDT (so-called Steiner Tree [3]) is computed offline knowing the server and all receivers beforehand.

- *routing table (RT) size*, defined as the maximum number of memory-bits required to locally store the RT entries. Thus, the RT size is computed using the bit-size of a single entry and the number of entries it comprises. The storage required by the algorithm is directly related to routing system scalability because the less memory a router needs to store its entries, the more scalable the routing system would be.
- *recovery time*, defined, in this case, as the maximum time needed to receive back a multicast transmission at the receivers once a failure occurs in a link of the MDT.

The experimentation tests of PIM-SSM and GCMR were performed in a network consisting of 207 Autonomous Systems (ASes). As we were interested in the inter-domain aspects, we represented the behavior of each AS with a single router with multiple interfaces (Figure 5). This 207 ASes emulates a portion of the Internet where one AS provides a multicast service to the rest of 206 ASes. In particular, we executed ten runs of the same experiment: one multicast server located in one AS is firstly selected and then ten receivers, located in ten different ASes, joined the MDT sequentially. Both the server and the receivers were randomly chosen.

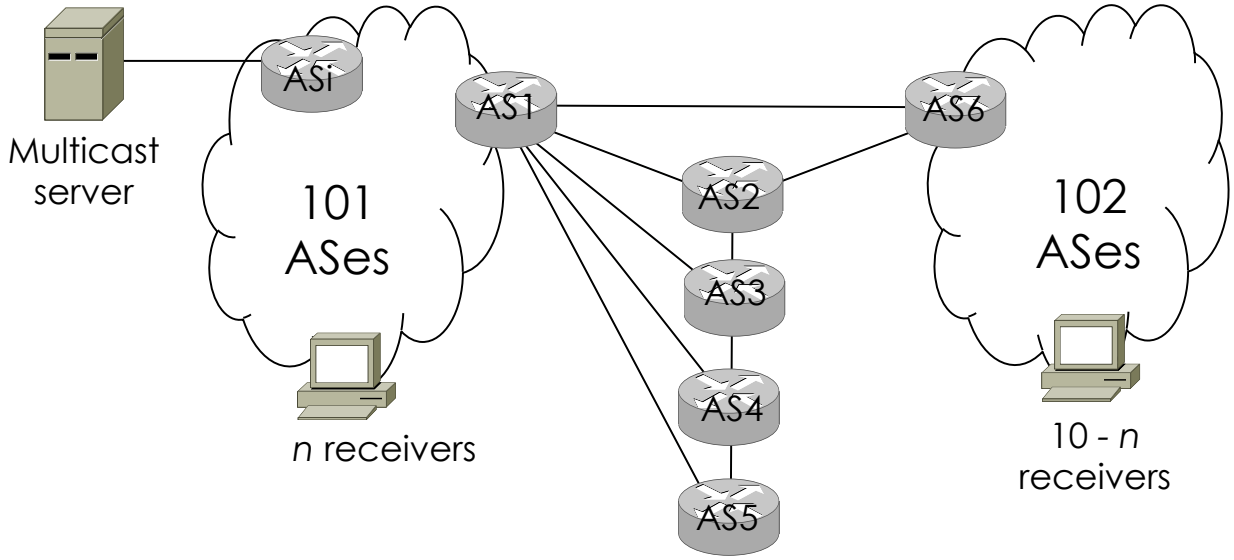


Figure 5. Topology for the experiments

The considered topology presents a particular structure consisting of AS1-AS6 that interconnects two blocks of ASes. The reason of this AS1-AS6 connectivity is to enforce the so-called *path exploration problem* of BGP [12]. In fact, path exploration suggests that, in response to path failures or routing policy changes, some BGP routers may try a number of transient paths before selecting a new best path

or declaring unreachability to a destination. Consequently, a long time period may elapse before the whole network eventually converges to the final decision, resulting in slow routing convergence. We will analyze this phenomenon when discussing the results.

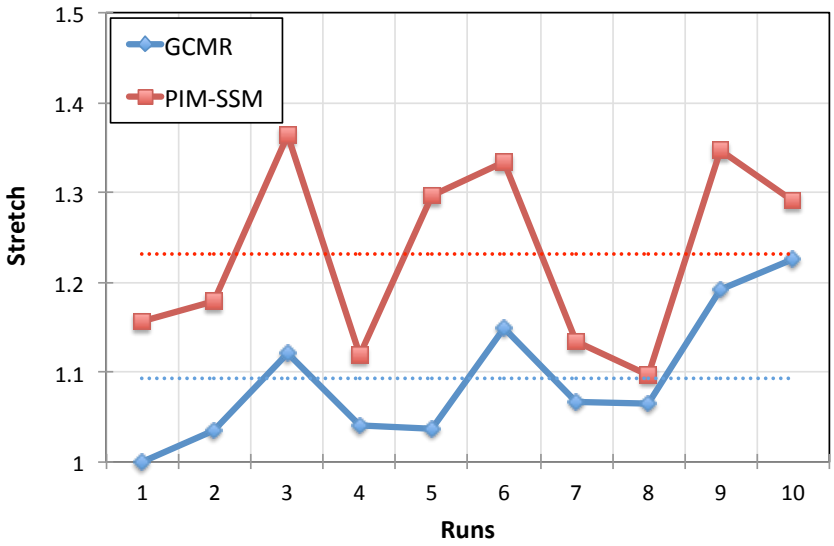
5.2 Results and discussions

Figure 6a shows the stretch of GCMR and PIM-SSM in ten different executions. GCMR presents in all runs lower stretch than PIM-SSM (0.14 better on average) and some deterioration (0.095 on average) against the optimal MDT (remind that stretch-1 is the reference). Using the information obtained from BGP, PIM-SSM establishes a shortest path tree (SPT) between the receivers and the source, and it is known that the SPT is not optimal from the stretch point of view. This result is consistent with the simulation results provided in [3], where, in much more larger simulation scenarios (32k nodes) and high number of receivers (500-4000), GCMR obtains approximately 0.1 better stretch than SPT.

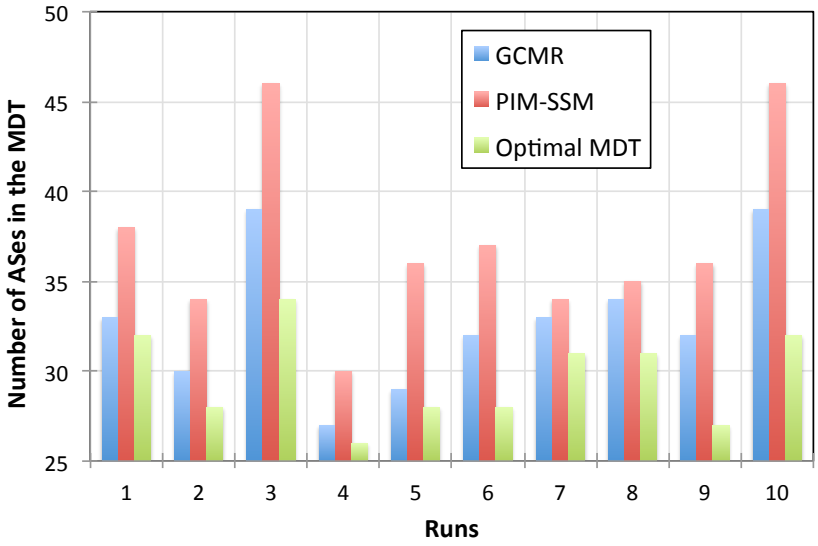
In Figure 6b, we show the number of ASes involved in the MDT. As expected from the previous results, GCMR is much more closer to the optimal MDT than PIM-SSM. This is an important outcome of GCMR from the cost point of view as providers charge customers based on the proportion of their resources that are used in multicast. Thus, as GCMR uses less transit ASes, the cost for the customers should be less using GCMR than PIM-SSM.

The comparison in terms of Routing Table (RT) size is presented in Figure 6c. To determine the size of the RT (in bits), we only consider the nodes involved in the MDT and use the RT formats defined in [9]: while GCMR only needs MRIB and TIB information, PIM-SSM, besides MRIB and TIB, also needs some unicast information from BGP to determine the shortest path towards the multicast source. Figure 6c shows that GCMR requires around 44% less bits than PIM-SSM.

a)



b)



c)

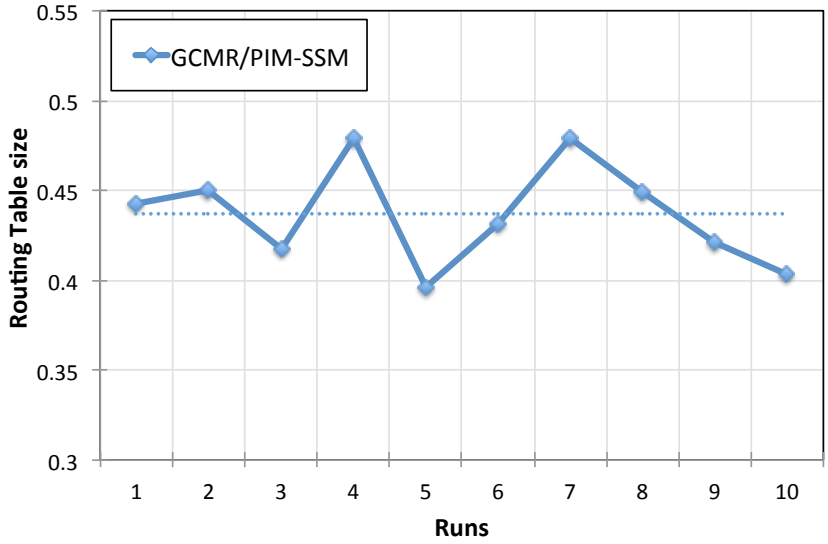


Figure 6. Comparison between GCMR and PIM-SSM in terms of a) Stretch, b) Number of AS in the MDT, c) Routing Table size (in bits) as GCMR/PIM-SSM ratio obtained in ten different runs

Finally, we emulated a failure in the link connecting AS1 and AS6 and counted the maximum time elapsed to receive back the multicast transmission at the affected receivers.

In the case of PIM-SSM, it equals the number of Minimum Route Advertisement Interval (MRAI) -in seconds- elapsing between the withdraw of the routing state corresponding to the initial route towards the multicast source and the next stable state where BGP routing state convergence can be declared [8] plus the time required for PIM Hello adjacency and Join message exchange with the new next hop router along the route to the multicast source node [7]. Due to the particular structure of AS-AS6 interconnection, BGP tends to explore all alternatives (problem known as path exploration [12]) before reaching a stable state and, as a consequence, the obtained traffic interruption result quite high in the experiments, 2 minutes and half approximately.

In the case of GCMR, recovery time comprises the time for the failure-detecting node to initiate a search and receive answers from its neighbors that point to the least cost branching path plus the time to initiate a Join message. In the experiments, the recovery time is of the order of one second.

6. Conclusions

This paper experimentally validates and compares the performance of standardized PIM-SSM multicast routing algorithm, which uses BGP for path discovery, and the novel GCMR compact multicast routing scheme recently proposed in [3]. In a large-scale experimental emulation platform, the GCMR scheme provides a better performance compared to PIM-SSM in terms of the memory space it requires to locally store the routing information and the stretch factor increase multicast routing paths it produces. Moreover, the adaptive property of the GCMR scheme induces a limited number of re-routing events in case of failure compared to the recovery strategy applied by the combination of BGP and PIM-SSM. Future work aims at validating GCMR in even larger topology (e.g., O(1k) ASes) with multiple concurrent multicast sessions.

References

[1] J.S.Turner, D.E.Taylor, “Diversifying the Internet”, *Globecom 2005*, St. Louis, MO, USA, December 2005.

[2] C.Diot, et al., “Deployment Issues for the IP Multicast Service and Architecture”, *IEEE Network*, vol. 4, no. 1, pp. 78-88, January 2000.

[3] P.Pedroso, D.Papadimitriou, D.Careglio “Dynamic compact multicast routing on power-law graphs”, *Globecom 2011*, Houston, TX, USA, December 2011.

[4] T.Li (ed.), “Design Goals for Scalable Internet Routing”, *IETF RFC 6227*, May 2011.

[5] B.Fenner et.al., “Protocol Independent Multicast - Sparse Mode (PIM-SM)”, *IETF RFC 4601*, August 2006.

[6] T.Ballardie, P.Francis, J.Crowcroft, “Core Based Trees (CBT): An Architecture for Scalable Multicast Routing”, *ACM Sigcomm 1995*, Cambridge, MA, USA, August 1995.

[7] H.Holbrook, B.Cain, “Source-Specific Multicast for IP”, *IETF RFC 4607*, August 2006.

[8] T.Bates, et al., “Multiprotocol Extensions for BGP-4”, *IETF RFC 4760*, January 2007.

[9] P.Pedroso, D.Papadimitriou, D.Careglio, “A name-independent compact multicast routing algorithm”, *Technical Report*, UPC-DAC-RR-CBA-2011-15, March 2011

[10] D.Papadimitriou, D.Careglio, P.Demeester, “Performance analysis of multicast routing algorithms”, *Technical Report*, UPC-DAC-RR-CBA-2013-4, August 2013.

[11] F. Papadopoulos, D. Krioukov, M. Boguñá, A. Vahdat, “Greedy Forwarding in Dynamic Scale-Free Networks Embedded in Hyperbolic Metric Spaces”, *Infocom 2010*, San Diego, CA, USA, March 2010.

[12] R.Oliveira, Beichuan Zhang, Dan Pei, Lixia Zhang, “Quantifying Path Exploration in the Internet”, *IEEE/ACM Trans. Networking*, vol. 17, no. 2, pp. 445-458, April 2009.