A Transport-Level Protocol Suite for Multimedia Multicast Communication over an SSM-enabled Internet

Roberto Canossa1, Laurent Maty, David Hutchisson

Lancaster University - Computing Department - Lancaster, LAI 4YR, UK
1University of Naples “Federico II” - Dipartimento di Informatica e Sistemistica - Napoli, Italy

PIM-SM is for multicasting over the Internet: implications for end-to-end protocols

- PIM-SM introduces a new model of multicasting over the Internet, suitable for applications with few well-known sources
- PIM-SM is currently defined by I-D draft-ietf-multicast-pim-sm
- IANA has allocated the 222 / 8 address block for exclusive use of Source-Specific Multicast
- PIM-SM realizes components of the EIGP/SMP multicast service model based on source-specific channels

When PIM-SM is adopted, there is no way for a receiver to multicast a packet to all other receivers of (S,G) PIM-SM channel

- This, indeed, is one of the benefits of the SM model, since it prevents receivers from unwanted traffic
- However, many transport-protocol-related only on a “multicast-bad-channel”
  - E.g., incorrectly use and control messages “promiscuous” in the source and other groups receivers

Our solution: to build a “parallel” infrastructure for control purposes by using Application Level Multicasting

This infrastructure allows us to insert a prune to prevent connections among end-systems to form a spanning tree at the channel source

Our constraint:

- Since we work in end-systems, we assume no knowledge of network topology and routing
- Since we want to limit the complexity of our algorithms, we assume only “partial” knowledge of group membership

The Control Tree Building algorithm

- Goal: to build a spanning tree rooted at the channel sender S and connecting all receivers R, exploiting only “partial” info on the group membership and network topology
- Characteristics: as a support infrastructure for all these control protocols that need a “multicast-bad-channel”
- In all cases, messages will flow optimum from source towards the receiver S
- In some cases, messages could be aggregated at intermediate nodes, to avoid source implosion
- Typical applications: receivers counting, group integrity management, multicast transmissions, local repairs, ...
- Basic algorithm: the spanning tree is built as receivers join the channel through a distributed algorithm that starts at the tree root
  - Each node in the tree aggregates info to fewer children (load 1)
  - A receiver R finds info to the tree by contacting a candidate parent P, starting from the root node R (message HELLO)
  - P sends back to the list of its own children C (message HELLO, ACH)
  - C maintains its distance from P and C to the info to P (message HELLO, OCH)
  - P finds a place for R by evaluating all possible configurations having A as parent and [C] as children
  - One of the following three cases can occur:
    - P accepts R as child and it does not need to move one of the previous children C2 message HELLO to C2
    - P accepts R as child but it needs to move one of the previous children C2 message HELLO to R
    - P sends R to child C2 message HELLO to R and the algorithm continues until it receives a message HELLO from R
  - To evaluate all the possible configurations, a score function is used to measure how “good” a configuration is
  - Let NC = “no children” and NC “one child”
  - No. of configurations to be evaluated = NC(NC-1)/2 = if (distom(J) = NC NC = NC - 1 = if (distom(J) NC)

- Variant: hierarchical organization of receivers in “domains”
  - Motivation: organization is naturally “clanized”
  - Goal: the tree is built in a way that receivers of the same domain are all children of the same node called “domain root”
  - Assumption: each receiver knows its own domain (if P = 2/members in domain)
  - Recovery distance (own domain)/P by contacting one candidate parent
  - The tree root maintains a table of all domain roots
  - New receivers are indicated to existing domain parents
  - If a domain root does not exist for a domain, the next closest is elected domain root of the domain

Notes on the score function

- The score function describes the shape of the tree
- What is the “best” shape for a control tree?
  - We also impose one constraint that must be satisfied for a configuration to be “acceptable”
  - Rule 1: all the nodes of the domain should remain “clanized”
  - Rule 2: the tree should grow only from the tree root (or the domain root)
  - Rule 3: domain root should be as high as possible in the tree (i.e., as close as possible to the tree root)

Notes in the pictures below: colors are used to identify the domain each node belongs to.

Evaluation

- The Tree Building Protocol has been tested with the on-2-sim simulator, by generating a random topology consisting of:
  - 5 network “core”, composed of 25 nodes
  - 15-50 network, each composed of 30 nodes
  - 100 protocol entries were placed in end-systems connected to stub networks
  - 108 stimuli were executed in each condition, to calculate average execution time and performance values
  - To characterize the tree built by our algorithm, we plot the following performance indices against graph size (i.e., n of nodes):
    - mean and maximum distance of a node from its parent
    - mean and maximum distance of a node from the domain root
    - mean and maximum link stress: no. of copies of the same packet on a link
    - delay, latency, i.e., the ratio between the delay along the tree and the delay along a direct connection to the sender

Observations:
  - the tree built by our algorithm depends on the order of arrivals
  - when the “clan” of receiver increases, the average distance among groups decreases

However, to slightly favor a growth of the tree in breadth, rather than in depth, we apply two corrective actions:

1) We discard configurations of the type B when its score is no less than a given fraction of the score of configuration of type A
2) When more configurations exhibit a similar score, the configuration is chosen randomly among them

- A typical tree built by our algorithm (108 trials) reveals the characteristics of the system
- Colors are used to identify domains

Graph produced with the other simulation tool available at http://www.caida.org/tools/i2/treechamber/

Protocol overview in the GCAP protocol suite

Overview of the GCAP Project

- The work presented here has been developed within the European Project GCAP, in collaboration with the LBN Laboratory of the University “Parsis of Maastricht” of Public GCAP is funded by the European Commission under the Framework Y/ST Programme.
- GCAP’s goal is to design, implement and test a suite of transport-essential protocols to support multimedia multicast applications
- GCAP assumes an underlying network layer service the suitability provided by an IP network enabled with multicast PIM-SM services
- GCAP Protocol Entities will be deployed in end-systems and at active edge devices
- GCAP Protocol Entities deployed in edge devices could behave as domain roots for all the receivers in their domains

The GCAP Protocol Suite

- The GCAP Protocol Suite can be divided into Data Plane Protocols and Control Plane Protocols
- Data Plane Protocols provide end-to-end transport of multimedia data units, with Partial Order / Partial Reliability, according to a fully Programmable Transport Protocol model
- Control Plane Protocols provide support functionality for multimedia multiparty applications, such as Receivers Counting, Group Management, QoS monitoring, Reliability, ...
- All GCAP protocols are implemented in Java as net-level libraries

Further investigation and future work

- Testing of different cost functions for intra-domain placement and domain roots placement
- Design of end-to-end control protocols on top of the Control Tree
- Refinement of local heuristics
- Definition of tree building mechanism, triggered by maximum
- Usage of PIM-SM in IPv6 networks - Extension of the Multicast Listener Discovery protocol (MLD) to support IPv6 source-specific options
- Less implementation by the end of the year