« Host-Based Multicast »
an alternative group communication service

Ecole d’été Réseaux Haut Débit et Multimédia (RHDM’00), Marrakech, July 9-16th, 2000

vincent.roca@inrialpes.fr
http://www.inrialpes.fr/planete/
Projet Planète

Université P&M. Curie, LIP6 / INRIA Rhône-Alpes, Planète project

Outline of the presentation

● Part 1 - Motivations and basic concepts
● Part 2 - HBM specificities
● Part 3 - The various proposals:
  ○ 3.1 - Void
  ○ 3.2 - Narada
  ○ 3.3 - AMRoute
  ○ 3.4 - our HBM approach
● Part 3 - Conclusions
Part 1:

The motivations for host based multicast...

An alternative group communication service

Not everybody has access to multicast routing...

See this very very good paper ;-)  
[Diot00] “Deployment issues for the IP multicast service and architecture”

- multicast adds no value to the receiver
- many deployment issues for ISPs:
  - offering wide area multicast is technically complex
  - multicast routing protocols are still under development/research
  - brings router migration problems
  - requires complex management
  - when is multicast more interesting than unicast?
  - What billing model?
- many functionalities are still not available: security, group management, address allocation...
- consequences: only Sprint/UUnet have a commercial multicast offer
Using a unicast/multicast reflector

- A reflector offers multicast connectivity to hosts restricted to unicast only transmissions
- can be as simple as this… (two sockets + a few lines of C)

- or more complex… (eg. www.ifi.uio.no/~meccano/reflectorn)

Using a unicast/multicast reflector... (cont’)

- Pros
  - simple
  - already available

- Cons
  - requires a manual setup
  - how many reflectors are needed?
  - where to place them?
  - limited and uncontrolled efficiency
    (depends on the placement of reflectors, on the number of unicast clients, etc.)

- HBM tries to solve these problems...
The goals of HBM

● Create automatically and easily a group communication service, using efficiently the underlying unicast/multicast routing services
  ○ no complex user implication
  ○ supports routing protocol heterogeneity
  ○ goes beyond traditional multicast which requires a unified deployment

● Many different names…
  Host Based Multicast (HBM)
  End Host Multicast
  Application-level Multicast, etc.

Using HBM for efficient unicast/multicast integration

● HBM builds a group interconnection topology between the participants, using unicast or multicast routing where they are the most efficient

● HBM can include multicast areas for improved:
  ○ scalability (all the nodes are collapsed)
  ○ efficiency (avoids several point-to-point connections)
Using HBM for unicast/multicast integration... (cont’)

● Pros:
  ❍ automatic setup
  ❍ more efficient than reflectors
  ❍ dynamic adaptation to network conditions
  ❍ more security

● Cons:
  ❍ can turn out to be rather complex
  ❍ hosts may be unstable (much more than routers/links are !)
  ❍ hbm is neither as efficient nor as scalable as native multicast routing

● two key points: robustness and efficiency

Can HBM do more ?

● Ok, so HBM can be used for unicast/multicast integration...
  But can it do much more ?

● Some people say that WAN multicast is of limited interest and suggest using HBM instead (e.g. Yoid)...

● well... HBM will also raise scalability and bad network resource usage problems...
  ❍ example: host connected through a low speed modem

● I don’t believe this is THE group communication technology
Part 2:

Host based multicast specificities...

Things that make HBM different from multicast

- A HBM node can be
  - a host (general assumption)
  - a (dedicated) server within the site
  - a (dedicated) server within the ISP
  - ... but no assumption is made on routers
    (unlike multicast where the routers are supposed to implement a given routing protocol)
Things that make HBM different from multicast... (cont’)

● HBM can only rely on end-hosts...
  ○ easy deployment, flexible
  ○ but not very efficient with bandwidth limited sites

● but it can also include dedicated servers at each site...
  ○ a server is certainly more stable than hosts
  ○ no processing power problems

● or even dedicated servers within ISPs
  ○ certainly the most efficient solution from a networking point of view
  ○ end-hosts are in “leaf-only” mode, so packets cross the ISP/site link only once

Things that make HBM different from multicast... (cont’)

● HBM is an overlay... (cont’)
  ○ different flavors of multicast/unicast routing protocols can be integrated
  ○ the physical topology is (almost completely) hidden at HBM level
  ○ create a complete virtual graph with all nodes/distances between them
  ○ several possible metrics (unidirectional delay, RTT, number of hops)

From physical topology... ...to complete virtual graph
Things that make HBM different from multicast... (cont')

● In traditional multicast, knowledge (1) is distributed and (2) routers only know that a given interface leads to a receiver

● In HBM, group members are known...
  ♦ either by a RP (Rendez-vous Point) (e.g. Yoid),
  ♦ or by the source,
  ♦ or by everybody (e.g. Narada)

- reliability is more limited
  ♦ nodes are far less reliable than routers/links are
  ♦ if HBM is implemented in a library, the application may be stopped/crash/etc.

⇒ redundancy, adaptation and fast failure discovery/tree update are required

● tree setup is entirely under control
  ♦ HBM topology can be tailored on a per-link basis
    ⇒ e.g. a specific tunnel can be setup on a lossy, congested path
  ♦ HBM can be tailored for application needs (ILP), unlike general purpose multicast routing
A few metrics to evaluate the benefits/costs of HBM

- **Physical link stress**: number of times a packet crosses a link
  - For link L1, HBM: stress = 2, multicast: 1; unicast: 4

- **Resource usage**: sum of delay * stress for all the links
  - HBM: resource usage = 2 * 1 + 2 * 1 + 20 + 2 * 2 + 1 = 31;
    multicast: 27; unicast: 50

- **Relative delay penalty**: ratio of HBM delay versus unicast delay between two hosts
  - For N1-N5, HBM: RDP = 26/22 = 1.18; multicast: 1; unicast: 1

PART 3:
The various proposals
The proposals and their fields of application

- HBM for general purpose Internet
  - Yoid
  - Narada
  - ours

- HBM for Ad’hoc networks
  - AMRoute

- Providing improved group communication services (e.g. more reliability) with HBM
  - RMX, see [Chawathe00]

3.1- The Yoid proposal (Your Own Internet Distribution)

- Status of the proposal
  - Proposed by Paul Francis (ACIRI)
  - Announced in July 1999
  - Described in a white paper [Francis99], September 1999
  - Detailed protocols descriptions, December 1999
  - The author works on a public implementation, due date: end of 2000
  - Previously called Yallcast
  - http://www.yoid.com/
The Yoid proposal... (cont’)

**Acronyms**
- **YTMP**  
  Yoid Tree Management Protocol  
  (tree/mesh creation and management)
- **YDP**  
  Yoid Distribution Protocol  
  ((reliable) transmission over the tree/mesh)
- **YIDP**  
  Yoid Identification Protocol  
  (packet, sender and receiver identification)
- **yTCP**, **yRTP**, **yMTCPP**, **yMRTP**  
  Yoid Transport Layers  
  (Yoid uni/multicast equivalents of TCP/RTP)

**Very ambitious project !**
- addresses ALL the aspects of multi-peer transmissions: connectivity, flow-control, reliability, etc.
- **we only focus on the YTMP protocol**
The YTMP (Yoid Tree Management) protocol

- Basic concepts:
  - a group is identified by the tuple: `<@RP, n°port RV, group name>`
  - RP: Rendez-vous Point; contact point used by newcomers; in charge of the tree/mesh management
  - tree: single shared (i.e. by all sources/receivers) loopless tree; used for data delivery
  - mesh: redundant interconnection; used for robust delivery (e.g. for tree partition/unreachable nodes discovery)
  - YTMP first creates a tree, then a mesh (tree-first)

The YTMP protocol... (cont’)

- Tree creation process:
  - choose a root (e.g. the more active source)
  - each child must choose one and only one parent
  - a parent can accept or refuse a child
  - create a “root-path” list at each node N as the list of nodes from N to the root; used to correct loops during a tree update
  - in spite of this “root-path”, a loop can be created during simultaneous topology updates; a specific algorithm is run in that case...
Personal appraisal of the Yoid proposal

● Pros:
  ❍ one of the first proposal
  ❍ created an awareness in the community
  ❍ using a RP seems good in many situations
  ❍ considers unicast/(LAN restricted) multicast integration

● Cons:
  ❍ too ambitious, should only focus on the tree creation process
  ❍ the tree-first approach leads to complex mechanisms (e.g. for loop avoidance)

● I don’t believe that Yoid will be the HBM solution

3.2- The NARADA proposal

● Status of the proposal
  ❍ Y-H Chu, S. Rao, H. Zhang (CMU) [Chu00]
  ❍ simulated, implementation under progress
  ❍ for standard Internet nodes

● Sketch of the protocol
  ❍ creates a self-organizing overlay that incrementally self-improves
  ❍ this is a mesh-first protocol (unlike Yoid)
  - first create a bidirectional mesh
  - then use a Reverse Path Forwarding algo (like DVMRP) on the mesh to create the tree

● Consequences:
  ⇒ the quality of the tree depends on the quality of the mesh
  ⇒ no centralized tree construction process
  ⇒ leads to per-source unidirectional trees (unlike Yoid)
The NARADA proposal... (cont’)

- Example of mesh/tree construction:

  ![Diagram of mesh/tree construction]

  From physical topology...

  To logical mesh topology (2 random neighbors)...

  NB:
  - the mesh is bidirectional
  - one different tree for each source
  - N2 does not send to N5 as the shortest path from N5 to N1 (source) is through N4

...and to tree (using the RPF algorithm)

Group management with NARADA

- based on the mesh for simplicity/robustness
- Joining a session:

  1. (incomplete) list of contacts obtained out-of-band (e.g. mail, www, etc.):
     - N1, N4, N9
  2. choose some random neighbors: here N1, N9
  3. add me as a neighbor
  4. OK
  5. new mesh link

  ◦ the new mesh can be far from optimal as the newcomer’s main goal is only to get connected to the mesh
  ◦ will be improved in a second step...
Group management with NARADA... (cont')

- Leaving a session:
  - a member leaving the session must first inform its neighbors
  - Ok for graceful departures, but in case the host crashes...

- Identifying silent members
  - reachability is continuously tested...
  - each node \( N_i \) keeps the following information for node \( N_k \):
    - @ \( N_k \)
    - last sequence number \( s_k \) that \( i \) knows \( k \) has issued
    - local time when \( N_i \) first received information \( N_k \) has issued \( s_k \)

  if (\( N_i \) didn't receive an update from \( N_k \) for \( T_m \) time) then
    // \( N_k \) is either dead or the mesh partitioned
    add \( N_k \) in list of silent nodes;
    start mesh_repair algorithm;

Group management with NARADA... (cont')

- Repairing mesh partitions

// each member has a list of silent members (i.e. at least for \( T_m \))
periodically and probabilistically remove a node \( N_k \) from this list;
if (cannot contact node \( N_k \) directly) then
delete \( N_k \) completely;
else
  add a new link in the mesh to node \( N_k \);
Improving a mesh

- the mesh can have many inefficiencies due to:
  - new neighbors who attach the first responding member
  - partition repairs (does not consider topological efficiency)
  - evolution of group membership
  - changing network conditions

- an incremental mesh improvement is required for good tree quality

- to add a link:
  - first compute the utility for node $N_i$ to add a link to node $N_k$
    - $utility = 0$;
    - for (each member $N_m$ that $N_i$ knows) do
      - $CL = \text{current latency between } N_i \text{ and } N_m \text{ along the mesh};$
      - $NL = \text{new latency between } N_i \text{ and } N_m \text{ with link } N_i-N_k;$
      - if ($NL < CL$)
        - $utility += (CL - NL) / CL$;
    - then, if utility is above a given threshold, add link $N_i-N_k$

- to drop a link
  - compute the utilization of each link from node $N_i$
  - where utilization is
    - $\text{util}_{i,n} = \text{number of members for which } i \text{ uses } k \text{ as next hop};$
    - $\text{util}_{i,k} = \text{number of members for which } k \text{ uses } i \text{ as next hop};$
    - $util = \max (\text{util}_{i,n}; \text{util}_{i,k});$
  - drop the link $N_i-N_k$ with lowest utilization if below a given threshold
  - but this is not very clear in their paper...
Personal appraisal of the NARADA proposal

● Pros:
  ❍ group management over the mesh is kept simple
  ❍ robust distributed algorithm
  ❍ creates shortest path trees (for a given mesh)
  ❍ mesh adaptation and improvement possible

● Cons:
  ❍ node failure is only detected after a silent period of Tm seconds...
  ❍ nobody has a global knowledge of the mesh and therefore every
    decision (add/drop a mesh link) is based on limited local information
  ❍ the distributed approach of NARADA can be a problem for limited
    resource nodes (like mobile PDA, phones...)

● having a central (possibly replicated) node doing mesh/tree calculation
  is another possible approach...

3.3- The AMRoute proposal

● Status of the proposal
  ❍ M. Liu, R. Talpade, A. McAuley, E. Bommaiah [Liu99]
  ❍ covered by an old Internet Draft [Liu98]
  ❍ dedicated to Mobile Adhoc Networks (MANET IETF group)
    – dynamic multihop network
    – rapidly and randomly changing
    – wireless communications
    – usually bandwidth constrained
    – can have limited power range
    – no fixed infrastructure (no fixed server, no fixed router, etc.)

  ❍ example of application: rescue operations, battlefield
The AMRoute proposal... (cont’)

- Everything (network and membership) is very dynamic...
  - AMRoute handles group dynamicity
  - Adhoc unicast routing prot. handles network dynamicity

- Sketch of the protocol
  - creates a self-organizing overlay (like NARADA)
  - this is a mesh-first protocol (like NARADA)
    - first create a bi-directional mesh, using an Expending Ring Search (ERS) algorithm
    - then create the tree, subset of the mesh

- Consequences:
  ⇒ with ERS, the mesh creation exploits locality and therefore is not too bad (unlike NARADA) (at least at creation...)
  ⇒ no centralized tree construction process (like NARADA)
  ⇒ leads to a single shared bidirectional tree (unlike NARADA)
  ⇒ the tree is memorized and refreshed periodically (unlike NARADA)

Group management with AMRoute

- based on the mesh for simplicity/robustness

- distinguishes:
  - logical core members
  - non-core members

- the goals of a core is to initiate:
  - mesh joins
  - tree creation

- a core is not a central point in data distribution and changes dynamically
  ≠ RP of PIM-SM

- at least one core per group, but there can be several cores (e.g. after group partition, or whenever a newcomer arrives)

- controlled by a core-election algorithm
Mesh management

- Joining a session:
  - newcomer declares itself as a logical core
  - broadcasts JOIN_REQ messages, with increasing TTL, until it receives a JOIN_ACK from a member

- Leaving a session:
  - send a JOIN_NAK on all the mesh links

Tree management

- Creating a tree
  - the core sends periodic TREE_CREATE along the mesh
  - a member receiving a non-duplicated TREE_CREATE forwards it on all outgoing links and marks incoming/outgoing links as belonging to the tree
  - on receiving a duplicated TREE_CREATE (or for any other reason), a member returns a TREE_CREATE_NAK

...which leads to this shared tree
Repairing mesh partitions

- Repairing mesh partitions
  - can happen after a member leaves
  - a member who no longer receives any TREE_CREATE declares itself as a core after some random time
  - it then initiates new mesh and tree create

- Core resolution
  - there can be multiple active cores in a mesh (e.g. after merging a partitioned mesh)
  - detected when receiving TREE_CREATE from several different cores
  - elect one using a well-known deterministic algorithm and forward only the TREE_CREATE from this core

Personal appraisal of the AMRoute proposal

- Pros:
  - one of the first HBM proposal
  - simple protocol both for mesh and tree
  - robust distributed algorithm
  - the tree is regularly updated, taking into account the network dynamics

- Cons:
  - mesh quality degrades with the time and no mechanism is available to incrementally improve it
  - experiments [Lee00] have shown that mesh/tree management cannot cope with highly dynamical Adhoc networks...
    Simpler mesh-only protocols are more efficient.
3.4- Our HBM proposal

- Status of the proposal
  - work in progress
  - (partially) simulated

- Sketch of the protocol
  - creates a self-organizing overlay that periodically self-improves
  - this is a RP-based protocol
    - does not rely on any mesh
    - uses a centralized RP to calculate the shared tree topology
  - made possible by the complete knowledge of group membership/communication costs

Sketch of the protocol

- distinguish:
  - core-members (CM), that are part of the core distribution tree
  - non-core members (nonCM), that graft on the existing distribution tree
  - classification based on various criteria... (see later)

- everything is under the control of a central RP who:
  - knows CM and nonCM
  - knows distances between them (several possible metrics)
  - is responsible of the distribution topology calculation and dissemination

- requires that CM periodically evaluate distances between them and inform the RP
- likewise nonCM evaluate their distances with (a subset of) CM and inform the RP
Sketch of the protocol... (cont’)

● an example:

(1) evaluates inter-node distances
(2) send this info to the RP
(3) calculate new distribution topology
(4) distribute new topology information
    (either in pt-to-pt or along the new tree)

CM1: dist
CM2: dist
CM3: dist

nonCM5

CM1
CM2
CM3

RP

CM1: dist
core tree
CM2
CM3: dist
nonCM graft on the core tree
(e.g. closest CM)

Sketch of the protocol... (cont’)

● OK, that’s not scalable...

❖ but HBM (and other proposals) are not scalable either
❖ the true solution to scalability is native multicast routing...
  ...except if you use DM protocols, or MSDP, or any other non scalable piece of protocol
❖ anyway, many collaborative work sessions include a limited number of non multicast capable sites!
❖ and a single HBM node in a site can serve many local participants using native multicast!

● OK, reliability greatly depends on the RP reliability...

❖ If the RP is a fixed host collocated with the primary source, this is not an issue
❖ you can also setup secondary RPs (like secondary DNS/NIS...servers)
Sketch of the protocol... (cont’)

- ...but this is simple
  - limited coherency problems as everything is centralized
  - limited burden on the hosts
    (an asset in case of PDAs, etc.)

- ...and it creates a “not too bad” distribution topology
  - the distribution tree created is optimal with respect to the known distances at that time
  - this tree is regularly updated
    (periodic update, depending on the group size/stability/etc., or triggered by some event)

Offering a robust group communication service...

- redundancy is required (for data transmissions too)
  - how many redundant links?
  - where to place them?
  - fixed redundant links or source-dependant links?
  - ...under investigation...

- fast failure discovery mechanism is required
  - easily done with ACK Aggregation in case of a tree topology
  - easily done with ring topologies (the opposite node recvs two copies)

- adaptation is required
  - unstable nodes should be leaves rather than transit nodes...
  - node stability is continuously monitored

- unreliable hosts

- reliable host
Offering a robust group communication service... (cont)

● each node has a “capability”
  ● “transit_possible”, “leaf_only”, or “disconnected” (e.g. mobile with limited power/slow network/unstability => “leaf_only”)
  ● a CM is “transit_possible”
  ● a nonCM is “leaf_only”
  ● the user can say if he wants to be transit node
  ● the RP can modify node capabilities if required (e.g. if all the users choose to be “leaf_only”)

  ● capability(node) = f(user_desires, node_stability, group_req, ...)
    [0; alpha] disconnected
    [alpha, beta] leaf_only => nonCM
    [beta;1] transit_possible => CM

  ● enables adaptation...

Offering a robust group communication service... (cont)

● Several *topologies* are possible:
  ● bus: no fault-tolerance

  ● tree: medium fault-tolerance
    optimal perf. requires per-source tree
    minimum global cost tree possible

  ● ring: 1-fault tolerance if bi-directional
    perf. does not depend on source position
    balanced load on all links

  ● star: good fault-tolerance except for the core
    source must be core for optimal perf.
    very high traffic load close close to core
Offering a robust group communication service... (cont)

- **sun**: balance between ring and star topologies
- good fault tolerance if unreliable hosts are moved at the end of sun beams

![Diagram showing sun topology]

- the host connectivity on the ring can be checked with bi-directional transmissions...

![Diagram showing packet transmission]

Offering an “efficient” group communication service

- sometimes a multicast cluster should act as a transit area
  - requires a discovery mechanism within the collapsed WAN multicast cluster

![Diagram showing multicast cluster]

- update the topology periodically
  - frequency may depend on group size, stability...
  - ...under investigations...
Conclusions

- Host-based multicast: an alternative group communication service
- IMHO it cannot replace native multicast routing (limited scalability)...
- ... but it can be of great benefits for hosts limited to unicast routing (more efficient than tunnels, reflectors...)
- Offers interesting additional properties (e.g. on lossy, congested links...)
- Several proposals exist but many open points remain and no large scale experiment has been done...
Open points... HBM scalability

- so far we assumed only one session (i.e. multicast group)
- we said that scalability w.r.t. number of members is not a problem...
- but...
  - how does HBM scale with the number of sessions ?
  - how does HBM handle multi-layer sessions (e.g. ALC, asynchronous layered coding) ?
    - one HBM tree per layer is rather inefficient
  - how does HBM handle closely related sessions (e.g. one audio + one video + one wb channels)
    - one HBM tree per tool is rather inefficient
    - use shared trees ?

Selected bibliography

- Multicast Deployment

- Yoid Approach (http://www.yoid.com)

- Narada Approach (http://www.cs.cmu.edu/~hzhang/multicast/)
Selected bibliography... (cont')

- **AMRoute Approach** ([http://www.isr.umd.edu/~daphnel/](http://www.isr.umd.edu/~daphnel/))

- **RMX Approach** ([http://www.cs.berkeley.edu/~yatin/](http://www.cs.berkeley.edu/~yatin/))