Multicast & Group Communication Services

Lecture 3

Guide to Reliable Distributed Systems (Birman).
Also: slides and resources can be found at:
http://www.dsn.jhu.edu/courses/cs437/

IP Multicast is documented in IETF RFC’s and Internet-Drafts
which can be found at: http://www.ietf.org/
The Multicast Paradigm

- Ordering (Unordered, FIFO, Causal, Agreed).
- Delivery guarantees (Unreliable, Reliable, Safe/Stable).
- Open groups versus close groups.
- Failure model (Omission, Fail-stop, Crash & Recovery, Network Partitions).
- Multiple groups.

Using Traditional Transport Protocols for Multicast

Point to point (TCP/IP)

- Automatic flow control ✓
- Reliable delivery ✓
- Connection service ✓
- Complexity (n^2)
- Linear (?) degradation in performance
Using Traditional Transport Protocols for Multicast (cont.)

Unreliable broadcast/multicast (UDP, IP-Multicast)

- Employs hardware support for broadcast and multicast. ✓
- Message losses: 0.01% at normal load, 10%, 20%, 30% or more at high load.
  - Buffers overflow (in the network and in the OS).
  - Interrupt misses.
- Not a connection-oriented service.

IP Multicast

- Multicast extension to IP.
- Best effort multicast service.
- No accurate membership.
- Class D addresses are reserved for multicast: 224.0.0.0 to 239.255.255.255 and are used as group addresses.
- The standard defines how hardware Ethernet multicast addresses can be used if these are possible.
IP Multicast (cont.)

Extensions to IP inside a host:

- A host may send IP multicast by using a multicast address as the destination address.
- A host manages a table of groups and local application processes that belong to this group.
- When a multicast message arrives at the host, it delivers copies of it to all of the local processes that belong to that group.
- A host acts as a member of a group only if it has at least one active process that joined that group.
IP Multicast Group Management

Extensions to IP within one local area network

The Internet Group Management Protocol (IGMP)

• A host that joins a group transmits a report message to IP multicast address 224.0.0.1 (all hosts group)
• A multicast router sends periodic general query messages to discover IP multicast groups with local hosts to 224.0.0.1
• A host replies after setting a random timer for each group it is a member of
  -- The host sends a report message for that group only if no other host replied by the random timer expiration.

Extensions to IP within one local area network

The Internet Group Management Protocol (IGMP) – cont.

• When the host that replied last leaves the group, it sends a Leave Group message on IP multicast address 224.0.0.2 (all routers group).
• The multicast router then sends a group specific query to check whether there are additional members in the group
• After a timeout with no positive host responses for a certain group, the IP Multicast router stops participating in that group (beyond the local area network)
IP-Multicast Routing

Extensions to IP between routers in one network

Distance Vector Multicast Routing Protocol (DVMRP, PIM-DM)

- Messages ABOUT groups are sent on the special all hosts group 224.0.0.1
- Time to live: limits the distance messages travel.
- Dense method: Flood & Prune. All routers get packets initially, then prune out parts of the network that do not have group member hosts.
- Tunneling: encapsulates multicast packets in regular packets in order to pass through routers that do not support IP Multicast.

IP-Multicast Routing (cont.)

IP Multicast between routers in one network

PIM-SM

- Sparse Method for better scalability
  - only routers that participate, or are on the way to routers that participate, get IP multicast messages
  - In contrast to Dense Method that employs Flood and Prune
- Utilizes rendezvous points for each group
  - Rendezvous point router is determined via hashing the group address into a list of possible RP routers in the network (maintained by a bootstrap router)
PIM-SM (Sparse Method) Join Operation

- Join Request
- Join Confirm

IP-Multicast Routing (cont)

• Extensions to Open Shortest Path First – the link state routing protocol common in the Internet
• Group membership of local areas in the network is based on IGMP and is flooded between the routers on the network.
• Shortest path trees are calculated on demand for each source to each group (of destinations) it participates on
• Can work with inter-AS Multicast Routing (MBGP+) to support IP Multicast operation beyond a single network (beyond an AS Autonomous System)
IP Multicast Challenges

• Scalability with the number of applications / groups.
  – How many groups are needed on a world-wide basis?
  – What happens to the core routers with many global groups?
• Turned off by ISPs.
  – Can you think why?
• What can be done about that?

– Private networks – using IP multicast – e.g. for IPTV
– Overlay networks – using unicast.
The Overlay Networks Approach

- Application-level routers working on top of a physical network.
- Overlay links consist of multiple “physical” links.
- Incurs overhead.
- Placement of overlay routers not optimal.
- Flexible use of peer-protocols.
- Provides added value.

Multicast Using Overlay Networks

- Routing is not optimal. But functional and does not require state at intermediate routers – just at overlay routers.
- Multiple overlay networks can coexist in the Internet without overhead to Internet routers.
- All the multicast traffic is seen as unicast packets at the network level. No need for hardware support.
- Group names space extends only to the scope of the application (no longer global).
Reliable Multicast Services

Service-Type

- SAFE
- AGREED
- CAUSAL
- FIFO
- RELIABLE

Cost (latency)

Reliable Multicast Services (cont.)

Fifo Order

\[ m \xrightarrow{\text{cause}} m' \] if \( q(m) \rightarrow q(m') \)

Causal Order

\[ m \xrightarrow{\text{cause}} m' \] if \( q(m) \rightarrow q(m') \)

Agreed Order

- Total order
- Consistent with Causal order and overlapping groups

Safe Delivery

* Not ordering

- Consistent with Agreed order
- Message is delivered after received by all processors
Multicast Protocols Outline

- Vector Timestamps (ISIS System)
- Trans Protocol (used by Transis)
- Lamport Timestamps
- Single Ring Protocol (Totem)
- Accelerated Ring Protocol (Spread)

Vector Time Stamp: Reliability and Causal Ordering (ISIS system)

- Each process maintains a time vector of size n.
- Initially \( VT[i] = 0 \).
- When process \( p \) sends a new message \( m \): \( VT[p]++ \)
- Each message is stamped with \( VTm \) which is the current \( VT \) of the sender.
- When process \( p \) delivers a message, \( p \) updates its vector: for \( k \) in 1..n:
  \[
  VTp[k] = \max\{ VTp[k], VTm[k] \}.
  \]
Isis Causal Order (Cont)

Comparing messages:

\[ VT_1 < VT_2 \iff \text{ for } k = 1..n \ VT_1[k] \leq VT_2[k] \]
\[ \text{and } \exists k \ VT_1[k] < VT_2[k] \]

Determining causality:

\[ m_1 \rightarrow m_2 \iff VT_1 < VT_2 \]

Determining whether a message sent by q can be delivered:

for any \( k \) in 1..n:

\[ VT_m[k] = VT[k] + 1 \text{ if } k = q. \]
\[ VT_m[k] \leq VT[k] \text{ otherwise.} \]

Example 1

Messages \{1,0,0\} and \{0,1,0\} are not causally related, so they can be delivered in any order.
Example 2

Time

<table>
<thead>
<tr>
<th></th>
<th>p</th>
<th>q</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>t1</td>
<td>1,0,0</td>
<td>1,0,0</td>
<td>1,1,0</td>
</tr>
<tr>
<td>t2</td>
<td>1,1,0</td>
<td>1,1,0</td>
<td>1,0,0</td>
</tr>
</tbody>
</table>

Message \{1,0,0\} **causally precedes** \{1,1,0\}, so \{1,0,0\} must be delivered **before** \{1,1,0\}.

Example 3

Time

<table>
<thead>
<tr>
<th></th>
<th>p</th>
<th>q</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>t1</td>
<td>1,0,0</td>
<td>1,0,0</td>
<td>1,1,0</td>
</tr>
<tr>
<td>t2</td>
<td>1,1,0</td>
<td>1,1,0</td>
<td>0,0,1</td>
</tr>
<tr>
<td>t3</td>
<td>0,0,1</td>
<td>0,0,1</td>
<td>1,0,0</td>
</tr>
</tbody>
</table>

A process does not update its timestamp until it **delivers** a message, so r sends \{0,0,1\}, even though it **received** \{1,1,0\}.
Isis Agreed (Total) Order

- Preserves causality.
- From time to time, the token holder sends an “ordering” message for all the previous Agreed-order messages it knows that are not yet ordered.
- Non-token holders cannot deliver Causal messages that are causally after an Agreed message that is not yet ordered.
- A new token holder may be determined after a membership change.

Multicast Protocols Outline

- Vector Timestamps (ISIS System)
- Trans Protocol (used by Transis)
- Lamport Timestamps
- Single Ring Protocol (Totem)
- Accelerated Ring Protocol (Spread)
The Trans Protocol

A Emits: A₁ A₂ A₃ ...

Scenario: A₁ a₁B₁ b₁B₂ b₂C₁ ...

Direct Ack: a₁B₁ → A₁

Indirect Ack: b₂C₁ → A₁, B₁

Example

Scenario: A₁ B₁ a₁b₁B₂ a₁C₁ c₁b₂C₂ ...
Example

Scenario: $A_1$ $B_1$ $a_1b_1B_2$ $a_1C_1$ $c_1b_2C_2$ ...
Example

Scenario: $A_1 \ B_1 \ a_1 b_1 B_2 \ a_1 C_1 \ c_1 b_2 C_2 \ ...$

![Diagram of interconnected nodes A, B, C with edges]

Example

Scenario: $A_1 \ B_1 \ a_1 b_1 B_2 \ a_1 C_1 \ c_1 b_2 C_2 \ ...$

![Diagram of interconnected nodes A, B, C with edges]
Example (Cont.)

D received: $A_1 \ B_1 \ a_1C_1 \ c_1b_2C_2 \ ...

Nack:: $c_1b_1b_2D_1$

The DAG may be revealed in a different way, but its structure will be identical at all the processors!!
The DAG

The DAG may be revealed in a different way, but its structure will be identical at all the processors!!

Total order can be based on the structure of the graph.

Vector Timestamps vs DAG

- DAG is a compaction of a vector timestamp
- The DAG method is more efficient network-wise and can scale better with the number of participants
- However, the DAG requires maintaining a more sophisticated data structure
Flow Control

P-2-P:

Last Ack

Stop

Multicast?

BOOM
Flow Control for Trans (from the Transis system)

Multicast:

Network Sliding Window

- Last All Ack
- Stop

Multicast Protocols Outline

- Vector Timestamps (ISIS System)
- Trans Protocol (used by Transis)
- Lamport Timestamps
- Single Ring Protocol (Totem)
- Accelerated Ring Protocol (Spread)
A Lamport Time Stamp Approach

- A Lamport Time Stamp (LTS) contains two fields:
  - Counter.
  - Process id.
- When sending a message:
  - Increment your counter.
  - Stamp your message.
  - Send your message.
- When receiving a message:
  - Adopt the counter on the message if it is bigger than your local counter.
- Unique for every message.

- It is useful to add an index next to the LTS, such that the index is incremented only when sending new messages.
  - The index helps track how many messages were sent by a process as well as how many were missed from that process.
A Lamport Time Stamp Approach

• A Lamport Time Stamp (LTS) contains two fields:
  – Counter.
  – Process id.
• When sending a message.
  – Increment your counter.
  – Stamp your message.
  – Send your message.
• When receiving a message
  – Adopt the counter on the message if it is bigger than your local counter.
• Unique for every message.
• It is useful to add an index next to the LTS, such that the index is incremented only when sending new messages.
  – The index helps track how many messages were sent by a process as well as how many were missed from that process.
• Agreed order of messages can be achieved by comparing (counter, process id) on message.
• FIFO and Causal order as a by-product.

Multicast Protocols Outline

• Vector Timestamps (ISIS System)
• Trans Protocol (used by Transis)
• Lamport Timestamps
• Single Ring Protocol (Totem)
• Accelerated Ring Protocol (Spread)
The Single Ring Protocol (Totem)

- The communication is multicast (UDP/IP).
- Services: Agreed (which is also FIFO and Causal), Safe.
- supports message omissions, network partitions, crashes and recoveries.

The Ring Ordering Scheme

Token fields

- type - {regular, form}.
- seq - of last message.
- aru - replaces acks.
- rtr - retrans. requests
- fcc - flow control.
The Ring Ordering Scheme (cont)

How to update the token aru?

- If token.aru = token.seq and have all the messages then should raise aru together with the seq (when sending new messages).
- If the token.aru is higher than the highest in-order message (local aru), lower the token.aru to the local aru.
- If is the one that lowered the aru, and the token.aru is still the same, should set token.aru to its local aru.

The trick: Everyone has all the messages up to: \(\text{min}(\text{token.aru}, \text{previous token.aru})\)

Multicast Protocols Outline

- Vector Timestamps (ISIS System)
- Trans Protocol (used by Transis)
- Lamport Timestamps
- Single Ring Protocol (Totem)
- Accelerated Ring Protocol (Spread)
Accelerated Ring Protocol

- **Original Ring Protocol**
  - Token is passed around a ring of participants
  - A participant multicasts while it holds the token, then passes the token to the next participant

- **Accelerated Ring Protocol**
  - Participants pass the token while multicasting
  - Circulates the token faster, allowing more rounds of sending per second
  - Allows controlled parallelism, while maintaining semantics
  - Designed for modern data centers

Yair Amir & Amy Babay  Fall 2016/Week 3

---

**Diagram: Accelerated Ring Protocol**

Original Protocol

Accelerated Protocol
Accelerated Ring Protocol

Updating token fields

• seq
  – Original: sequence number of last message sent
  – Accelerated: last sequence number claimed (message will be sent by the time the next token is processed)

• rtr - how do you decide what to request?
  – Original: request any missing messages with sequence numbers less than seq
  – Accelerated: request any missing messages with sequence numbers less than the value of seq on the token received in the previous round
    • seq may reflect messages that are still on their way or even not yet sent; you don’t want to request them unless they are really lost

1G Network Results

Simultaneously improves throughput by 60% and latency by 45%

500 Mbps, 1.3 ms latency

800 Mbps, 0.72 ms latency
10G Network Results

Simultaneously improves throughput by 20% and latency by 20%

Spread Original: 1 Gbps, 0.385 ms latency

Spread Accelerated: 1.2 Gbps, 0.31 ms latency

Simultaneously improves throughput by 40% and latency by 30%

Daemon-based Original: 2 Gbps, 0.39 ms latency

Daemon-based Accelerated: 2.8 Gbps, 0.265 ms latency
10G Network Results

![Network Results Diagram](image)

Simultaneously improves throughput by 35% and latency by 30%

Library-based
Original: 2.57 Gbps, 0.338 ms latency

Library-based Accelerated: 3.5 Gbps, 0.23 ms latency

Agreed Delivery Throughput (Mbps)

Latency (ms)
10G Network Results

Daemon-based: 6 Gbps (8850-byte msgs)

Library-based: 7.3 Gbps (8850-byte msgs)

10G Network Results with Loss (480 Mbps)

Accelerated Protocol
Original Protocol
* Lower line = better latency

Safe Delivery
Worst 5%
Safe Delivery
Agreed Delivery
Worst 5%
Agreed Delivery

Average Latency (ms)
Loss Rate at Each Daemon (%)
Failure Models

Possible faults:

- Message omissions and delays
- Processor crashes and recoveries
- Network partitions and re-merges

Most of the time it is assumed that:

- Message corruption is detected
- There are no malicious faults

Transis Membership Algorithm

- Utilizes hardware broadcast
- Ordering and Reliability optimized by DAG
- Handles crashes and recoveries.
- Handles network partitions and merges.
- Terminates in a bounded time (to do that, it allows the extraction of live but “inactive” processors).
- Guarantees extended virtual synchrony (relationship between messages and membership events).
Transis Membership (cont.)

- Partitioning / crashes detection
  - Timeout: invoked by timeout.
- Merging
  - Symmetric: no joining-side, accepting-side
  - Spontaneous: invoked after receiving Join messages or “foreign” messages.
- Faults may occur at any time (even while merging).

Faults & Partitions

When detecting a processor from which we did not hear for a certain timeout: we issue a fault message.

When we get a fault message, we adopt it (and issue our copy).

Problem: maybe p is only slow.
The Problem

When a partition occurs, we cannot always completely determine who received which messages.

It is proven that there is no solution to this problem (no common knowledge)

Merges

join (A,B)
join (A,B,C)
join (B,C)
The Single Ring Protocol

- Membership has several stages:
  - Detect that old membership is lost.
  - Gather together all alive members.
  - Form a new ring and send old state.
  - Transfer missing messages.
  - Install new membership.
- Supports message omissions, network partitions, crashes and recoveries.

Membership

Events

- Foreign Message
- Attempt join/ Join
- Gather timeout
- Commit timeout
- Form token
- Token loss timeout

States

- Operational state
- Gather state
- Commit state
- Form state
- Recover state
Membership (cont)

- Gather
  - Token loss timeout
  - Attempt Join OR Join
  - Gather timeout

- Commit
  - Join message AND Consensus AND Representative
  - Form token AND NOT representative

- Form
  - Extended Virtual Synchrony
  - Token loss timeout
  - Form token

- Recover
  - Token loss timeout
  - Form token AND representative

- Operational
  - Foreign Message
  - Token loss timeout
  - Commit timeout

Membership (cont..)

Representatives are shown shaded