A Proxy-based Mobile Group Membership Protocol for Large Scale and Highly Dynamic Groups

Guojun Wang[1,2], Jiannong Cao[1,*], Keith C. C. Chan[1]
[1] Department of Computing, Hong Kong Polytechnic University,
Hung Hom, Kowloon, Hong Kong, {csgjwang,csjcao,cskcchan}@comp.polyu.edu.hk;
[2] School of Information Science and Engineering, Central South University,
Changsha, Hunan Province, P. R. China, 410083, csgjwang@mail.csu.edu.cn

***Abstract*— We propose a RingNet hierarchy of proxies for mobile group communications, which is a combination of logical rings and logical trees to take advantages of the simplicity of the former and the scalability of the latter. Based on the hierarchy, we propose a fault tolerant mobile group membership protocol for large scale and highly dynamic groups. Simulation studies show that the proposed protocol scales very well when the size of the network becomes large, and that it is highly resilient to failures when the node failure probability becomes large.**

***Index Terms*— Group communications, group membership, group multicast, fault tolerance**

I. Introduction

A Group Communication System (GCS) provides communication services among groups of processes. A group consists of a set of processes called members of the group. A process may voluntarily join or leave a group, or cease to be a member due to failure. The membership of a group is the list of currently operational processes in a group, which is usually called a view. A GCS is a powerful building block to facilitate the programming of fault tolerant distributed applications.

Many existing GCSs are designed in WANs without explicit consideration of Mobile Hosts (MHs) as group members. Therefore, there is no guarantee that they can also work well in the presence of MHs. However, the design of a GCS in next generation mobile networks is challenging. The intrinsic issues in WANs like high message latency, frequent connectivity change, and instability [5], still exist in next generation mobile networks. More difficult issues need to be addressed, such as frequent disconnection of MHs from their attached wireless networks, frequent handoff of MHs from one wireless network to another, and frequent failure occurrence of MHs and wireless communication links.

To maintain membership for large groups in mobile next generation networks, both computation and communication overheads become very large. Some hierarchical approaches are proposed to reduce the overheads. In [1], the Host-View protocol is proposed. A Host-View is a set of Mobile Support Stations (MSSs) representing the aggregate location information of a group. By tracking a set of MSSs other than individual MHs, membership management is greatly simplified. However, the global updates that are necessary with every "significant move" make it inefficient and could cause lengthy breaks in service to the MHs.

In [3], the RelM protocol is proposed to tackle the problem in the Host-View protocol. It groups the MSSs together to form the third tier. Each group of the MSSs is controlled by a Supervisor Host (SH). Since the SH is part of the wired network, it can handle most of protocol details such as maintaining connections for the MHs, and collecting the acknowledgement messages for reliable communications.

In [2], the RMP protocol with three-tiers of MHs, MSSs and Coordinators is proposed. In RMP, each MSS maintains a data structure called *local* that identifies the set of MHs in its cell, and the handoff of an MH does not imply the exchange of any message in the wired network.

We are motivated by some interesting problems concerned with large scale and highly dynamic groups in next generation mobile networks. Firstly, dynamic membership that involves a very large number of MHs to join or leave a group should be tackled. Secondly, dynamic locations of MHs add complexity to existing protocols that only deal with dynamic membership. Thirdly, dynamic networks due to node/link failures make the group membership information difficult to maintain.

We solve these problems by proposing a *RingNet hierarchy of proxies*, which is a combination of logical rings and logical trees to take advantages of the simplicity of logical rings and the scalability of logical trees. Such a combination makes this hierarchy more reliable than a tree-based hierarchy.

II. The System Model

We propose a multi-tier proxy-based next generation mobile networks architecture by placing multiple tiers of proxies in the network shown in Fig. 1. We call the multicast senders placed within the wired Internet *global senders*, while the multicast senders within the wireless networks *local senders*.

We differentiate two kinds of proxies: (1) a variety of Direct Proxies (DPs) that directly serve their attaching MHs, e.g., access points in WLANs, base stations in cellular networks, and satellites in satellite networks; and (2) some Intermediate Proxies (IPs) placed between DPs and multicast senders.

Based on the architecture, we propose a *RingNet hierarchy of proxies*. Fig. 2 shows an example hierarchy with four tiers, namely, Intermediate Proxy Tier 2 (IPT2), Intermediate Proxy

* Professor Jiannong Cao is the corresponding author.

1-4244-0355-3/06/$20.00 (c) 2006 IEEE

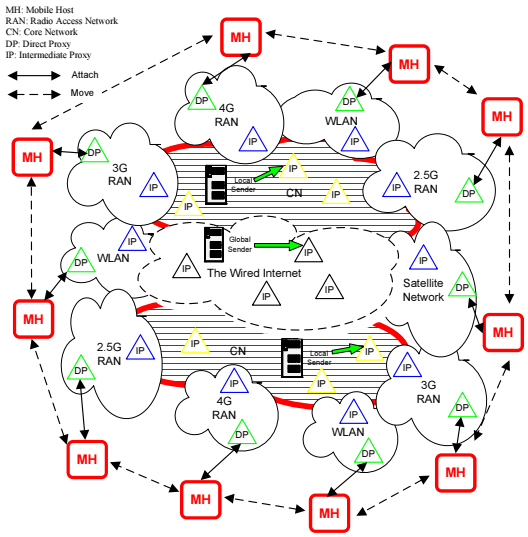

Fig. 1. The Proxy-based Next Generation Mobile Networks Architecture

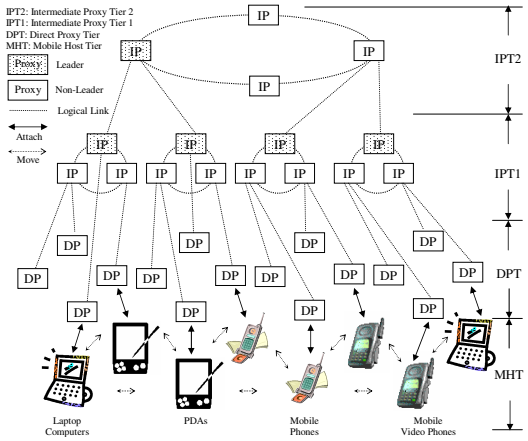

Fig. 2. The RingNet Hierarchy of Proxies

Tier 1 (IPT1), Direct Proxy Tier (DPT), and Mobile Host Tier (MHT), with the higher two tiers consisting of logically organized rings and with one *leader* in each logical ring. Besides functioning as normal proxies in a logical ring, the leader is also responsible for communicating with its *parent* in the upper-tier logical ring (if exists), and the leader is thus one of the *children* of its parent. For each proxy in a logical ring, there exists one proxy called its *previous* node, and another proxy called its *next* node.

Each proxy is initiated with the information about several *candidate sibling proxies* through which it can merge with a same-tier logical ring, and several *candidate parent proxies* through which it can attach to an upper-tier logical ring.

As for mobility detection, each DP periodically broadcasts heartbeat messages within its coverage area; upon receiving such a message, an MH as a group member announces its presence to the DP by sending back a greeting message containing its host identities. The MH refreshes its presence by sending Member-Update messages to the DP periodically.

III. The Basic RingNet Protocol

A. The Data Structures of MHs and Proxies

The data structure of an MH is as follows.

- GID: GroupID. Group identity, e.g. Class D address in IP multicast [4].
- DP: NodeID. Node identity of the attached DP, e.g. its IP address.
- GUID: GloballyUniqueID. Globally unique identity of the MH, e.g. Mobile IP Home Address (HA) [7].
- LUID: LocallyUniqueID. Locally unique identity of the MH, e.g. Mobile IP Care-of Address (CoA) [7].
- Status: Integer. Typical status like *operational*, *disconnected*, and *faulty*.

The data structure of a proxy is as follows.

- GID: GroupID. See the data structure of an MH.
- Current, Leader, Previous, Next, Parent, Children[]: NodeID. Node identities of the current, leader, previous, next, parent and children[] nodes in the hierarchy, respectively, e.g. their IP addresses. Each item of the Children[] stands for one of the children nodes.
- PreviousOK, NextOK, ParentOK, ChildrenOK[]: Boolean. Status of the current node's previous, next, parent and children nodes in the hierarchy, respectively, with TRUE for *non-faulty* and FALSE for *faulty*.
- CandidateSiblings[], CandidateParents[]: NodeID. Lists of node identities of candidate siblings and candidate parents, respectively.
- CandidateSiblingsOK[], CandidateParentsOK[]: Boolean. Lists of status of candidate siblings and parents, respectively, with TRUE for *non-faulty* and FALSE for *faulty*.
- ListOfMembers[]: MemberInfo. List of operational members within the coverage area of a sub-RingNet hierarchy.

B. The Basic Membership Algorithms

1) The Basic Membership-Propagation Algorithm: We design two categories of signaling messages: Membership-Change (MC) and Membership-Update (MU) messages.

We first discuss MC messages. Member-Join/Leave/Handoff messages are generated when an MH joins, leaves, or hands-off in a group. Each MH as a member periodically sends Member-Update messages to its attached DP to refresh its status. Each DP periodically checks its ListOfMembers[] to determine whether any MH has been inactive for too long or not. If yes, it removes the MH from the list as a faulty member.

We then discuss MU messages, which are periodically generated by each proxy according to its ListOfMembers[]. If it's a non-leader in a logical ring, then it sends the message to its leader in the logical ring. If it's a DP or a leader in a logical ring, then it sends the message to its parent (if exists).

2) The Basic Topology-Maintenance Algorithm: The RingNet hierarchy changes when MHs dynamically join, leave, fail in the group, or move around from one DP to another. There are two types of topology change events. One is Proxy-Join/Leave event for a proxy to join or leave a logical ring at the same tier of the hierarchy. Another one is Proxy-Attach/Detach event for establishing or finishing a parent-child relationship between nodes at neighboring tiers.

In order to avoid frequent reconstruction of the hierarchy triggered by the Proxy-Leave/Detach events, we adopt a *lazy-leave/detach* mechanism. For each DP, only when the DP does not contain any operational members after a timeout interval, it leaves/detaches from the hierarchy if all its adjacent DPs do not contain any operational members. For each IP, only when it does not contain any children nodes after a timeout interval, it leaves/detaches from the hierarchy.

A topology maintenance procedure is treated as a transaction by adopting a two phase commit technique. In each Proxy-Attach/Detach procedure, only two proxies are involved, i.e., the INITIATOR and its PARENT that is one of the candidate parents for Proxy-Attach or its current parent for Proxy-Detach. At the first phase, INITIATOR sends Proxy-Attach/Detach request to PARENT, and PARENT responds positively or negatively. At the second phase, INITIATOR commits/rollbacks, and notifies PARENT to do so accordingly.

In Proxy-Join/Leave, three proxies are involved, namely, INITIATOR, PREVIOUS that is one of the candidate siblings for Proxy-Join or the previous node of INITIATOR for Proxy-Leave, and NEXT that is the next node of PREVIOUS for Proxy-Join or the next node of INITIATOR for Proxy-Leave. At the first phase, INITIATOR sends Proxy-Join/Leave request to PREVIOUS, and PREVIOUS responds positively or negatively; then possibly INITIATOR sends Proxy-Join/Leave request to NEXT, and NEXT responds positively or negatively. At the second phase, INITIATOR commits/rollbacks, and notifies PREVIOUS and NEXT to do so accordingly.

IV. The Fault Tolerant RingNet Protocol

A. The Failure Detection Mechanism

Each proxy running our protocol monitors the status of all of the proxy's current and candidate neighbors: (1) As for the current neighbors, each proxy periodically sends heartbeat messages to all of its current neighbors, and a neighbor suspects the proxy to be faulty if it cannot receive such a message after a timeout interval. (2) As for the candidate neighbors, each proxy periodically sends polling messages to all of its candidate neighbors; when a neighbor received a polling message, it sends the message back to the proxy; the proxy suspects a neighbor to be faulty if it cannot receive the returned polling message after a timeout interval.

If a proxy failure breaks a previous-next relationship in a logical ring, then a Ring-Repair procedure runs to exclude the failure. If a proxy failure finishes a parent-child relationship in a hierarchy, then a Hierarchy-Repair procedure runs instead.

B. The Ring-Repair Procedure

For each proxy running the proposed protocol, if it detects its previous or next proxy as *faulty*, then it sets its PreviousOK or NextOK to FALSE accordingly, and starts to run a Fast-Repair sub-procedure to repair the ring. In order to exclude a single failure from a logical ring, the proxy that is aware of the failure issues a Fast-Repair message to a DESTINATION node to establish a new previous-next relationship between them. The DESTINATION is the proxy's *previous to previous* or *next to next* node in the logical ring where the proxy resides, the information of which is got from the heartbeat messages for detecting failures.

If Fast-Repair cannot make progress after a timeout interval, then it assumes consecutive failures occur in the ring, and starts to run a Slow-Repair sub-procedure to repair the ring by issuing a Slow-Repair message forwarded along the ring to find a DESTINATION node and then establish a new previous-next relationship between them. The DESTINATION is the node that cannot reliably forward the Slow-Repair message along the logical ring any longer.

C. The Hierarchy-Repair Procedure

For each proxy running the protocol, if it detects its parent proxy or one of its children proxies as *faulty*, it sets its ParentOK or a certain item in ChildrenOK[] to FALSE accordingly.

When a leader's ParentOK becomes FALSE, it joins the hierarchy either by a Proxy-Attach sub-procedure in subsection III-B.2 through one of its candidate parents, or by a Ring-Merge sub-procedure through one of its candidate siblings. If the logical ring contains the leader only, then Ring-Merge here is the same as Proxy-Join in subsection III-B.2. However, if the logical ring contains more than one node, then Ring-Merge can be regarded as an extension to Proxy-Join as follows.

The leader first issues a Ring-Merge request to some candidate siblings. If more than one responds positively, then the leader chooses one among them. A two phase commit technique is then adopted to merge the two rings into one. In each Ring-Merge transaction, four proxies are involved, namely, the LEADER, the LEADER's next node called NEXT, the CANDIDATE, and the CANDIDATE's next node called CANDIDATE-NEXT. The information of the last one is got from the returned polling messages when detecting failures. At the first phase, the LEADER sends a Leader-Merge message to the other three proxies, and the other three respond positively or negatively. At the second phase, if the three responses that the LEADER received are all positive, then the LEADER commits and notifies the other three to commit; otherwise, the LEADER rollbacks and notifies the other three to rollback. When the two rings are merged into one, the LEADER notifies the proxies in the former ring where the LEADER resides to change the new leader information.

V. Simulation Studies

A. Performance Metrics

Join-Delay is defined as the difference between the time at which an MH received the first multicast message and the time at which the MH issued its willing to join the group.

Handoff-Delay is defined as the difference between the time at which an MH received the first multicast message from a new DP and the time at which the MH was aware that it has handed-off to the new DP.

Service-Speed is defined as the difference between the time at which an MH's membership information is successfully registered with the leader of the top logical ring and the time at which the MH issued its willing to join the group.

Signaling-Overhead is defined as the total number/size of signaling messages received by all the proxies during

the simulation divided by the total number of proxies and then divided by the total simulated time. We call such a metric normalized number/size of signaling messages, or *Norm.Num.Msgs*/*Norm.Size.Msgs* for short.

Norm.Num.Events, the normalized number of Member-Join/Leave/Handoff events, is defined as the total number of such events divided by the total number of DPs in each time unit such as each minute.

B. Simulation Scenarios

We simulate our proposed protocol in ns-2 [6] by organizing the proxies into a RingNet hierarchy, and we simplify the simulation of tree-based protocols by organizing the proxies into a tree hierarchy. For each network topology, the DPs are configured into an *m*n* mesh. In the RingNet protocol, $\frac{m*n}{4}$ IPs are used at IPT1, $\frac{m*n}{4}$ IPs are used at IPT2, and $\frac{3m*n}{2}$ other proxies acting as intermediators are used among DPs and IPs, one of which also serves as a multicast sender. While in the tree-based protocols, $\frac{m*n}{4}$ IPs are used at IPT1, $\frac{m*n}{8}$ IPs are used at IPT2, $\frac{m*n}{8}-1$ IPs are used at IPT3, and only one IP is used at IPT4.

Each DP is attached by one MH for Sparse Mode (SM) simulation and two MHs for Dense Mode (DM) simulation. At any time, around one group member appears within a set of eight DPs in SM, and around one group member within each DP in DM. In the RingNet protocol in SM or DM, we call it RN-SM or RN-DM for short. While in tree-based protocols in SM or DM, we call it T-SM or T-DM for short.

In order to deal with fault tolerance, each DP in the RingNet protocol is configured with four candidate IPs at IPT1 for Proxy-Attach/Detach procedures, each IP at IPT1 is configured with four candidate IPs at IPT1 for Proxy-Join/Leave and four candidate IPs at IPT2 for Proxy-Attach/Detach, and each IP at IPT2 is configured with four candidate IPs at IPT2 for Proxy-Join/Leave. While in the case of tree-based protocols, each DP or IP is configured with four candidate IPs at its immediate upper tier for Proxy-Attach/Detach procedures.

In the case of the RingNet protocol, we simulate 8*4, 12*9, 16*16, 20*25, 24*36 DP configurations, with each IP at IPT1 initially being attached by a set of 4 DPs, with each IP at IPT2 initially being attached by 2, 3, 4, 5, 6 IPs at IPT1 respectively, and with each logical ring initially consisting of 2, 3, 4, 5, 6 IPs respectively. For the smallest topology in RN-SM, it consists of 32 MHs, 32 DPs, 8 IPs at IPT1, 8 IPs at IPT2, and 48 other proxies, with the whole coverage area of 5,360m*2,680m when the coverage area of each DP is 670m*670m. Therefore, the total number of nodes in the smallest topology in RN-SM is 128. For the largest topology in RN-DM, it consists of 1,728 MHs, 864 DPs, 216 IPs at IPT1, 216 IPs at IPT2, and 1,296 other proxies, i.e., totally 4,320 nodes, with the whole coverage area of 16,080m*24,120m when the coverage area of each DP is 670m*670m. Similarly, in the case of tree-based protocols, we also simulate 8*4, 12*9, 16*16, 20*25, 24*36 DP configurations, with each IP at IPT1 initially being attached by a set of 4 DPs, with some of the IPs at the ith IPT (here $2 \leq i \leq 4$) initially being attached by 2, 3, 4, 5, 6 IPs at the $(i-1)$th IPT, respectively.

In all the scenarios, the total simulated time is fixed to 600s. We fix the link bandwidth to 10Mbps, link delay to 10ms, message loss rate to 1.0% for all the wired links, and the equivalent link bandwidth to 2Mbps, link delay to 20ms, message loss rate to 2.0% for all the wireless links between DPs and MHs. Each MC message is assumed to be 10 Bytes in size, and MU messages generated from DPs and IPs are 20, 30 and 40 Bytes in size, respectively. All other signaling messages are assumed to be 10 Bytes in size. The Constant Bit Rate (CBR) multicast traffic is with message size of 512 Bytes and with message rate of one message every 100ms.

In order to detect whether a proxy's current/candidate neighbors are faulty or not, the timeout interval for sending heartbeat/polling messages and that for suspecting a node is set to 50ms/50ms and 200ms/250ms, respectively. The timeout interval at each DP for mobility detection is fixed to 1s, the Member-Update timeout interval at each MH is set to 1s, and the Membership-Update timeout interval at each proxy is set to 1s. The timeout interval for signaling message retransmission is set to 100ms and the maximal retransmission times is set to 3. The Lazy-Leave/Detach timeout interval for any proxy to really leave/detach from the hierarchy is set to 3s. The timeout interval for each DP to check whether any member has been inactive for too long or not is set to 3s. The timeout interval for Fast-Repair to start Slow-Repair is set to 1s.

To simulate dynamic locations of MHs, we adopt the CMU mobility model [6] at maximal speed of 15m/s and pause time of 5s. To simulate dynamic membership, we design a Member-Join/Leave pattern with two parameters: Minimal/Maximal Interval defined as the minimal/maximal time interval between any two consecutive Member-Join/Leave events for the same MH, which is set to {50s, 70s}. The start time for an MH to trigger its Member-Join event is defined as a random variable ranging from 1.3s to 20.0s.

To simulate dynamic networks, a proxy failure is emulated by breaking all its incident links simultaneously. We use the deterministic model in ns-2 [6] with four parameters: Start-Time that denotes the time for a proxy to start to be faulty, Up-Interval and Down-Interval that denote the proxy is up (i.e., non-faulty) and down (i.e., faulty) during that period of time, and Ratio that is the ratio of the number of proxies which may become faulty to the total number of proxies in the simulation. Start-Time is set by a random value ranging from 0.0s to 100.0s. The three-tuple of {Up-Time, Down-Time, Ratio} represents node failure probability, which is set to {95.0s, 5.0s, 0.2}, {95.0s, 5.0s, 1.0}, and {90.0, 10.0, 1.0} for 1.0%, 5.0%, and 10.0%, respectively.

C. Simulation Results

Simulation results are obtained as an average over 10 independent simulation runs. Fig. 3 shows the scalability property. The network sizes appear as pairs of numbers, which stand for the total number of simulated nodes in SM and DM, respectively. We do not report the number/size of heartbeat/polling messages for failure detection because they are localized to neighbors only and their numbers are relatively stable for different network settings with the same timeout intervals.

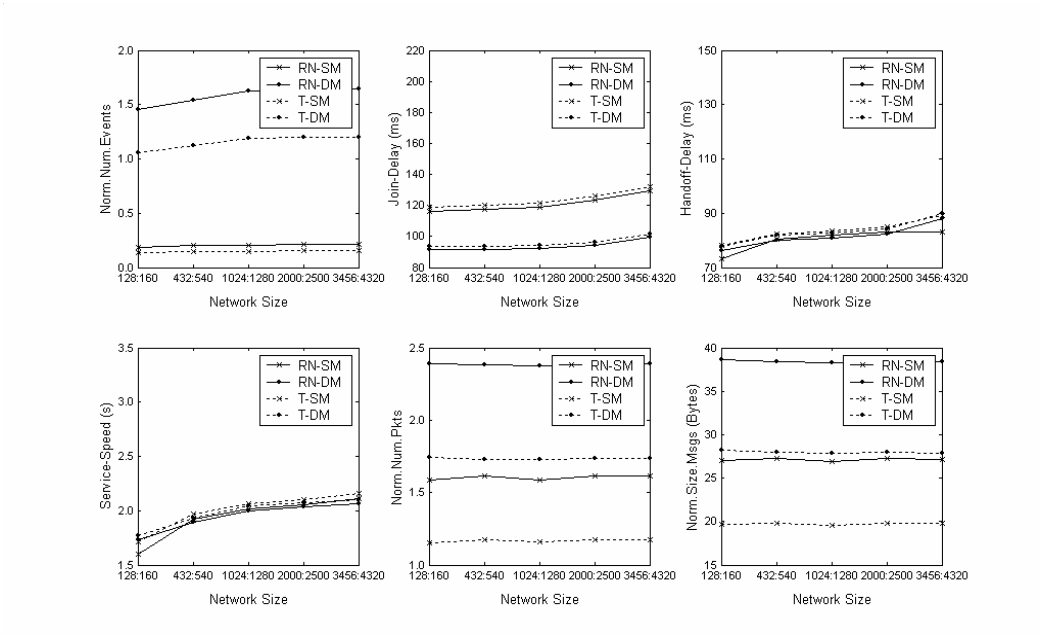

Fig. 3. The Simulation Results for the Scalability Property

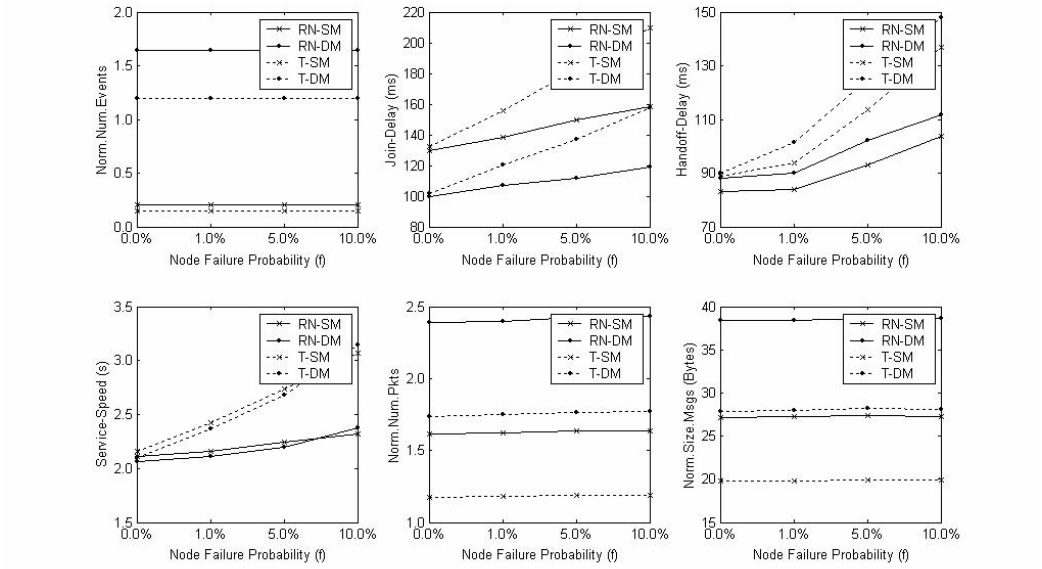

Fig. 4. The Simulation Results for the Fault Tolerance (Reliability) Property

Since our simulations show similar trends for fault tolerance property in all the topology settings, we reported here only with the largest topology setting in Fig. 4.

From the simulation results we observed that:

(1) Fig. 3 shows the proposed RingNet protocol scales very well. When the size of the network becomes large, the performance of our protocol, i.e., Join-Delay, Handofff-Delay and Service-Speed, keeps very high and varies in a small range, while the signaling overhead, i.e., Norm.Num.Msgs/Norm.Size.Msgs, is low and keeps almost at the same level. For example, the Join-Delay metrics for the five network sizes in RN-DM are 91.5ms, 91.7ms, 92.2ms, 94.2ms and 99.6ms, with the largest variance of only 99.6-91.5=8.1ms, and the maximal Norm.Num/Size.Msgs in RN-DM are 2.39 signaling messages and 38.6 Bytes per proxy per second, respectively.

Fig. 3 also shows that our protocol outperforms a little better than the tree-based protocols in terms of Join-Delay, Handoff-Delay and Service-Speed, due to the fact that we configured 2 tiers of IP proxies for the RingNet hierarchy and 4 tiers of IP proxies for the tree-based hierarchy, in order to accommodate the same DP configurations in both kinds of hierarchies. However, the cost of our protocol is that it needs to trigger more events, i.e., Norm.Num.Events, thus to incur more signaling overhead, i.e., Norm.Num.Msgs/Norm.Size.Msgs, than the tree-based protocols do. In this sense, our protocol has comparable scalability with the tree-based protocols.

(2) Fig. 4 shows the performance of our protocol is highly resilient to proxy failures in the RingNet hierarchy. With the node failure probability increasing from 0.0% to 10.0%, the performance of our protocol degrades gracefully, e.g., the Join-Delay metrics for the four node failure probabilities in RN-DM are 99.6ms, 107.2ms, 111.8ms and 119.2ms, with the largest variance of only 119.2-99.6=19.6ms between the largest fault case and the fault-free case.

Fig. 3 also shows that our protocol outperforms much better than the tree-based protocols in terms of Join-Delay, Handoff-Delay and Service-Speed when the node failure probability becomes large. The major reason is that a proxy in our protocol has two ways, i.e., either through its candidate sibling proxies or through its candidate parent proxies, to reconnect to the RingNet hierarchy when failures occur in the hierarchy; a proxy in the tree-based protocols, however, can only reconnect to the tree-based hierarchy through its candidate parent proxies when failures occur in the hierarchy.

(3) Fig. 3 and Fig. 4 show that some metrics are affected by the density of group members in both the RingNet protocol and the tree-based protocols.

(3.1) *Join-Delay*: If a group is densely populated, e.g., in RN-DM or T-DM, then it is highly probable for an MH to receive messages immediately when the MH joins a DP. On the contrary, if a group is sparsely populated, e.g., in RN-SM or T-SM, then an MH may have to wait for some time to receive messages when the MH joins a DP because the DP may have to join the hierarchy.

(3.2) *Signaling-Overhead*: Since both the size of the group and the number of proxies in RN-SM or T-SM are smaller than those in RN-DM or T-DM with the same network size, the number/size of signaling messages in RN-SM or T-SM is naturally less than that in RN-DM or T-DM.

(4) Fig. 3 and Fig. 4 show that some metrics are not affected by the density of group members in both the RingNet protocol and the tree-based protocols.

(4.1) *Handoff-Delay*: The gap between the two (four) curves in the Handoff-Delay sub-figures is not apparent because we assume our protocol and the tree-based protocols use a reservation mechanism to make adjacent DPs to join the hierarchy in advance whenever necessary, which weakens the difference of Handoff-Delay metrics between SM and DM.

(4.2) *Service-Speed*: This metric is relatively stable in all the simulated scenarios because we fixed the height of the RingNet hierarchy to 4, the height of the tree-based hierarchy to 6, and the timeout interval for generating Member-Update and Membership-Update messages to 1s.

VI. Conclusions

We proposed a novel group membership protocol based on a RingNet hierarchy. This hierarchy is more general than the ring-based hierarchy in [8] in the sense that each proxy within this hierarchy may have multiple children nodes, while each proxy within the ring-based hierarchy has at most one child node. Besides the work on the group membership problem using the RingNet hierarchy, we have also done some preliminary work on reliable and secure multicast communications [9], [10]. Our current work shows that solutions based on the RingNet hierarchy have comparable scalability but better reliability than those based on the tree-based hierarchy.

Acknowledgment

This work is supported by the Hong Kong PolyU research fund A-PF77, the University Grant Council of Hong Kong under the CERG Grant PolyU 5170/03E, and the National Natural Science Foundation of China under Grant No. 60503007.

References

[1] A. Acharya and B.R. Badrinath, "A framework for delivering multicast messages in networks with mobile hosts," *ACM/Kluwer Mobile Networks and Applications*, 1(2):199-219, Oct. 1996.

[2] G. Anastasi, A. Bartoli, and F. Spadoni, "A reliable multicast protocol for distributed mobile systems: Design and evaluation," *IEEE Transactions on Parallel and Distributed Systems*, 12(10):1009-1022, Oct. 2001.

[3] K. Brown and S. Singh, "RelM: Reliable Multicast for mobile networks," *Computer Communications (Elsevier)*, 21(16):1379-1400, Oct. 1998.

[4] S. Deering, "Host extensions for IP Multicasting," *IETF RFC 1112*, Aug. 1989.

[5] I. Keidar, J. Sussman, K. Marzullo, and D. Dolev, "Moshe: A group membership service for WANs," *ACM Transactions on Computer Systems*, 20(3):191-238, Aug. 2002.

[6] http://www.isi.edu/nsnam/ns/.

[7] C. Perkins, "IP mobility support," *IETF RFC 2002*, Oct. 1996.

[8] G. Wang, J. Cao, and Keith C.C. Chan, "RGB: A scalable and reliable group membership protocol in mobile Internet," *Proc. IEEE 33rd International Conference on Parallel Processing (ICPP 2004)*, Montreal, Quebec, Canada, pp. 326-333, Aug. 2004.

[9] G. Wang, J. Cao, and K.C.C. Chan, "A reliable totally-ordered group multicast protocol for mobile Internet," *Proc. IEEE 33rd International Conference on Parallel Processing Workshops (ICPP 2004 Workshops)*, Montreal, Quebec, Canada, pp. 108-115, Aug. 2004.

[10] G. Wang, L. Liao, J. Cao, and K.C.C. Chan, "Key management for secure multicast using the RingNet hierarchy," *Proc. 2004 International Symposium on Computational and Information Sciences (CIS 2004)*, Shanghai, P.R. China, pp. 77-84, Dec. 2004.