This paper was presented as part of the Mini-Conference at IEEE INFOCOM 2011

RDCM: Reliable Data Center Multicast

Dan Li*, Mingwei Xu*, Ming-chen Zhao†, Chuanxiong Guo‡, Yongguang Zhang‡, Min-you Wu†

*Tsinghua University, †Shanghai Jiaotong University, ‡Microsoft Research Asia

***Abstract*—Multicast benefits data center group communication in both saving network traffic and improving application throughput. The SLA (Service Level Agreement) of cloud service requires the computation correctness of distributed applications, translating to the requirement of reliable Multicast delivery. In this paper we present RDCM, a novel reliable Multicast approach for data center network. The key idea of RDCM is to minimize the impact of packet loss on the Multicast performance, by leveraging the rich link resource in data centers. A Multicast-tree-aware backup overlay is purposely built on group members for peer-to-peer packet repair. Riding on Unicast, packet repair not only achieves complete repair isolation, but also has high probability to bypass the pathological links in the Multicast tree where packet loss occurs. The backup overlay is organized in such a way that it causes little individual repair burden, control overhead, as well as overall repair traffic. We have implemented RDCM as a user-level library on Windows platform. The experiments on our test bed show that RDCM handles packet loss without obvious throughput degradation during high-speed data transmission.**

I. Introduction

Multicast benefits data center group communications in at least two aspects. By saving network traffic, it can increase the throughput of bandwidth-hungry computations such as Map-Reduce [1] and GFS [2]. By releasing the sender from sending multiple packet copies to different receivers, it can also reduce the task finish time of delay-sensitive applications such as on-line query index lookup. However, existing Multicast support built in data center switches/servers are based on the Multicast design for the Internet. Before the wide deployment of Multicast in data centers, we need to carefully investigate whether these Internet oriented Multicast protocols/technologies can well embrace the data center environment.

In this paper, we study a critical requirement of data center Multicast, i.e., reliable data transmission. Transmission reliability is important because it determines the computation correctness of upper-layer applications, and accordingly the SLA (Service Level Agreement) of cloud services. Packet loss in data center Multicast trees are supposed to be common due to several reasons. First, current data centers are built by commodity servers/switches for economical and scalability reasons [3], [4], [5], [8]. Packets can get lost from node/link failure in the fragile Multicast trees. Second, traffic is quite bursty and unpredictable in data center networks [7]. It is difficult to reactively schedule Multicast flows to low-utilized links. Hence, traffic congestion and consequent packet loss can occur anyplace in the Multicast tree. Third, the data center server population is usually very large. The larger the tree size is, the higher probability a packet gets lost with during transmission.

Previous reliable Multicast solutions for the Internet cannot readily encompass data center networks. Generally speaking, existing Internet reliable Multicast solutions can be divided into two categories, i.e., network-device assisted and end-host based. The former category requires assistance from network devices to achieve reliable data delivery. But the technical trend of data center design is to use low-end commodity switches, which are not supposed to bear much intelligence, for server interconnection [6]. Hence, the latter category of solutions, which put all reliable Multicast intelligence onto end hosts, seem to better accommodate data center networks. Yet, packet repair in these solutions either traverses along the Multicast tree, or does not achieve complete repair isolation. Considering the common packet loss in data centers, repairing the lost packets can cause severe Multicast throughput degradation, deferring the task finish time of cloud applications.

We design RDCM, an end-host based solution for reliable data center Multicast, by leveraging the characteristic of data center networks. Observing that current data centers are built with high link density, for each Multicast group we purposely construct a Multicast-tree-aware backup overlay upon group members for packet repair. In case of packet loss for a receiver, repair packets are transmitted in a peer-to-peer way on the backup overlay. Given the rich link resource and multiple equal-cost paths between any two servers, packet repair path in the backup overlay has high probability to be disjoint with the Multicast tree links. The advantage of this approach is multi-facet. First, repair isolation is completely achieved by packet repair riding on Unicast. No receiver will get the same packet twice, helping achieve high Multicast throughput in high-speed Multicast sessions. Second, when node/link failure happens, lost packets can still be repaired to the affected receivers via the backup overlay. Third, given packet loss from traffic congestion in the Multicast tree, backup-overlay based packet repair can alleviate the congestion in hot spots and thus enhance the Multicast throughput.

The primary challenges in RDCM design include how to control the individual burden for packet repair, the overall repair traffic and the protocol overhead. We carefully organize the backup overlay in such a way that each receiver is only responsible for repairing packets to at most *two* other receivers, no matter how large the group size is. The Multicast sender retransmits a packet only when all receivers lose it.

We have implemented RDCM as a user-level library on

The work in this paper is supported by the National Basic Research Program of China (973 Program) under Grant 2011CB302900 and 2009CB320501, the National Natural Science Foundation of China (No. 61073166), the National High-Tech Research and Development Program of China (863 Program) under Grants 2009AA01Z251, and the National Science & Technology Pillar Program of China under Grant 2008BAH37B03.

Windows platform for legacy IP Multicast applications. The experiments in a 16-server test bed show that packet loss can be gracefully handled during high-speed data transmission without obvious Multicast throughput degradation.

II. Design Challenges and Related Work

A. Design Challenges

We need to address the following challenges for reliable Multicast design in data center environment.

Fragile Multicast Trees: For considerations of economical cost and scalability, current data centers are built upon a large number of commodity switches and servers. Failure is common instead of exceptional in such networks [4], and the Multicast tree is quite fragile. Any node/link failure in the Multicast tree can pause packet delivery to downstream receivers. Reliable Multicast requires gracefully handling node/link failure during packet transmission.

Traffic Bursty in Data Centers: It has been shown that traffic is quite bursty and unpredictable in data center networks [7]. When the group size is large, traffic congestion can occur anywhere in the Multicast tree, resulting in frequent packet loss. Packet repair in existing Internet-oriented reliable Multicast solutions either traverses along the Multicast tree, or does not achieve complete repair isolation. Both can cause considerable throughput degradation for the Multicast session. This problem should be addressed for designing reliable Multicast in data center networks.

Design Intelligence: The low-end commodity switches used in current data centers usually have quite limited routing states, small buffer space as well as low programmability. These switches are not supposed to bear much Multicast intelligence, except the basic Multicast packet forwarding. Hence, network-device assisted reliable Multicast solutions are not suitable for data center environment.

B. Related Work

During the past two decades, many reliable Multicast solutions are proposed for the Internet. These proposals can be divided into two categories, namely, end-host based and network-equipment assisted. The former category is represented by PGM [13] and ARM [14], while the latter category includes SRM [9], RMTP [10], TMTP [11] and LBRM [12] and etc. Since low-end commodity switches in data centers are not supposed to bear much intelligence, end-host based solutions can better accommodate data center network.

SRM is a reliable Multicast framework for light-weight sessions in the Internet, especially for distributed whiteboard application. LBRM is designed for distributed interactive simulation in the Internet, characterized by low data rate and requirement on realtime packet loss recovery. TMTP targets for Internet collaborative multimedia applications, while RMTP focuses on bulk data transfer in the Internet. Our solution, RDCM, differs from them in that it is especially designed to support data intensive group computations in data centers. Packet repair in RDCM is highlighted by *complete repair isolation and the capability to bypass the congested or failed tree link where packet loss occurs*. Hence, the impact of packet loss on the Multicast throughput is minimized.

III. RDCM Design

We design RDCM, a novel solution to achieve reliable Multicast delivery in data center networks.

A. Basic Idea

RDCM provides reliable Multicast service to data center applications by addressing the technical challenges presented in Section II. First, RDCM avoids transmission pause when node/link failure happens in the fragile Multicast tree. Second, RDCM prevents Multicast throughput from severe degradation in case of traffic congestion in hot spots of the Multicast tree. Third, RDCM puts all design intelligence onto data center servers, without switch assistance.

By observing that current data centers are built with high link density, RDCM repairs lost packets in a peer-to-peer way among receivers when packet loss occurs during Multicast delivery. Given the rich link resource and multiple equal-cost paths between any two servers in data center networks, the packet repair paths riding on Unicast has high probability to be disjoint with the Multicast tree. The advantage lies in several aspects. First, complete repair isolation is achieved. Only receivers missing a packet will receive the repair packet, and thus no receiver gets the same packet twice. In this way, the packet receiving rate at end hosts can be well utilized to improve Multicast throughput in high-speed data transmission. Second, when node/link failure happens in the Multicast tree, lost packets can still be repaired to the affected receivers by Unicast paths bypassing the Multicast tree, avoiding the transmission pause on downstream receivers of a failed tree node/link. Third, given packet loss from traffic congestion in the Multicast tree, our packet repair mechanism can alleviate the congestion in hot spots and thus enhance the Multicast throughput. By the reasons above, the impact of packet loss on the global Multicast throughput is minimized.

But we need to tackle some challenges to realize the peer-to-peer packet repair mechanism in RDCM. First, any individual receiver is not supposed to bear too much packet repair burden, since the Multicast session size can be very large. Second, the control overhead of the RDCM protocol put at a receiver should not be high. Third, the overall repair traffic needs to be restricted, since Unicast based packet repair is used. Our solution is to design a purposely-built Multicast-tree-aware *backup overlay* upon group members to help packet repair..

B. Backup Overlay

The backup overlay for a group is constructed upon the Multicast tree, composed of a number of *overlay rings*. Each branching tree node, i.e., the one with more than 1 children in the tree, has a corresponding overlay ring. The overlay ring for a level-l (l starts from 0 at the lowest level) branching node i, denoted as O_i, is called a *level-l overlay ring*. If node i has c children nodes in the tree, O_i is composed of c receivers, each from the subtree rooted from one of node i's child. Assume j is one of i's children in the tree, the downstream receiver of j chosen to join O_i is called the *overlay proxy* for j, denoted as p_j. Hence, the length of an overlay ring is bounded by

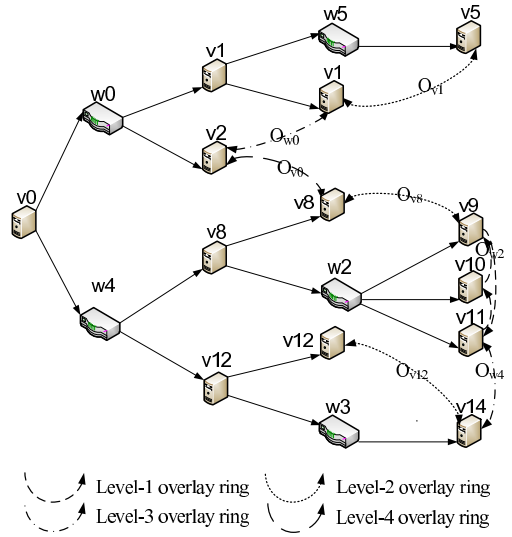


Fig. 1. An exemplified Multicast tree and backup overlay.

the number of switch ports in data center network. Within an overlay ring, each receiver has an *overlay successor* and an *overlay predecessor*. In addition, i is called the *tree parent* for O_i, p_i is called the *overlay parent* for O_i, and the receivers in O_i are called the *overlay children* for p_i.

Specifically, the sender is the overlay proxy for itself. It joins no overlay ring but it is both the tree parent and overlay parent for the highest-level overlay ring in the backup overlay.

In practice, the backup overlay for a group can be computed by a centralized controller or the source node of a Multicast group, which holds the tree information for the group. In what follows, we simply refer to this computing entity as the *Tree Manager*.

We take an exemplified Multicast tree from a BCube network, which is shown by solid lines in Fig. 1. Since the BCube servers also participate in packet forwarding, if a receiver appears as a relaying node in the tree, it is regarded as a switch as well as connecting another child receiver of itself, such as $v1$, $v8$ and $v12$ in Fig. 1. The logical links in the backup overlay for this group are shown as dashed lines in the same figure. There are 7 overlay rings, each corresponding to a branching node in the Multicast tree. For instance, the overlay ring O_{w4} is composed of two receivers, $v11$ and $v14$, which are the overlay proxies for $v8$ (switch) and $v12$ (switch) respectively. $w4$ is the tree parent for O_{w4}. $v8$ (receiver) is the overlay proxy for $w4$ and thus the overlay parent for O_{w4}. Within O_{w4}, $v11$ and $v14$ are both the overlay predecessor and overlay successor for each other.

Fig. 2 illustrates the whole procedure to build the backup overlay for a Multicast tree t. It takes a bottom-up way, i.e., from lowest-level branching tree nodes up to the sender. The overlay ring for a tree node i is constructed only after the overlay rings for all its children nodes are constructed. Then, if i has only one child, no overlay ring is built for it. Otherwise, for each of i's children, say j, its overlay proxy p_j is selected from its downstream receivers which has joined the least overlay rings; and all overlay proxies for i's children form the overlay ring for i, i.e., O_i.

Denotations:
C_i - The set of i's children nodes in the Multicast tree;
D_i - The set of receivers in the subtree rooted from i.

Algorithm:

```
1 void OverlayRingBuild(i){
2    if (|C_i| == 0) /*i is a leaf receiver*/
3       return;
4    for (j ∈ C_i)
5       OverlayRingBuild(j);
6    if (|C_i| == 1) /*i has only one child*/
7       return;
8    for (j ∈ C_i)
9       p_j = the receiver in D_j joining the least overlay rings;
10   Construct O_i on all p_j, j ∈ C_i;
11   return;
12}/*OverlayRingBuild*/
```

```
1 void BackupOverlayBuild(TREE t){
2    s = root of t;
3    OverlayRingBuild(s);
4    return;
5}/*BackupOverlayBuild*/
```

Fig. 2. Algorithm of constructing the backup overlay for a Multicast tree t.

We make definitions on the overlay rings a receiver r joins.

Leaf Overlay Ring: If receiver r joins such an overlay ring for a tree node i, in which r is the only one receiver in the subtree rooted from one of i's children, the overlay ring is called r's *leaf overlay ring*.

Proxy Overlay Ring: If r joins an overlay ring which is not its leaf overlay ring, the overlay ring is called r's *proxy overlay ring*.

Based on the definition, we can easily find that a receiver has no overlay children in its leaf overlay ring. But if it joins a proxy overlay ring for tree node i, it must have overlay children, because in this case it is the overlay parent for the overlay ring for one of i's children in the tree.

We can easily prove that RDCM holds the following two properties due to the construction rule of its backup overlay. First, every receiver joins exactly one leaf overlay ring if the group has more than one receivers, and joins at most one proxy overlay ring. Second, for each node i in the Multicast tree, any two downstream receivers of i can reach each other on the backup overlay by traversing the overlay rings for downstream tree nodes of i. The rigorous proof is omitted here for space limitation.

C. Packet Repair Scheme

Packet repair in RDCM is conducted in a peer-to-peer way upon the backup overlay.

Packet Acknowledgement: Each packet is assigned with a sequence number. A receiver r acknowledges a packet k in the following way. First, when r receives packet k either from the Multicast tree or from the backup overlay, it sends ACK for k to both overlay predecessor and overlay parent in its leaf overlay ring. Second, when r receives the ACK for k from its overlay children for the first time, it sends ACK for k to both overlay predecessor and overlay parent in its proxy overlay ring. All ACK packets are Unicast.

In this way, in the leaf overlay ring a receiver joins, it receives ACK for each packet only from its overlay successor in the ring. If a receiver joins a proxy overlay ring, it also receives ACK for each packet from its overlay successor in the ring as well as its overlay children. The sender receives ACK only from its overlay children. Note that the number of overlay children for a receiver is bounded by the number of ports in a switch, say, n, which is usually a small constant. As a result, a receiver/sender receives at most $n + 2$ ACKs from other receivers, independent with the group size. ACK implosion is thus effectively avoided in RDCM. In practice, to further reduce ACK messages exchanged, a receiver can choose to send one ACK for multiple packets, instead of acknowledging each one in a separate packet.

Repair Window: Every receiver maintains a repair window for packets received from both the Multicast tree and the backup overlay. Packets within the repair window are buffered. The upper bound of the repair window is the highest sequence number among the packets received. The lower bound of the repair window at a receiver r is moved up if all the following conditions are satisfied for the corresponding packet k. First, r has received k. Second, r has received ACK for k from its overlay successors of all the overlay rings it joins. Third, r has received an ACK for k from one overlay child if it joins a proxy overlay ring. Hence, each receiver only needs to wait for at most 3 ACKs before moving the window up and releasing the buffer for the corresponding packet.

Specifically, the sender also has a repair window. Since it is the overlay parent for the highest-level overlay ring, the repair window is moved up when receiving an ACK for the corresponding packet from any one overlay child.

Packet Repair: In RDCM, repair packet is either Unicast within an overlay ring or Multicast by the sender. We first discuss packet repair within overlay rings. Each receiver is responsible for packet repair to its overlay successors in all the overlay rings it joins. If a receiver detects that its overlay successor has lost a packet, it immediately transmits the repair packet. From Theorem 1, every receiver is responsible for repairing packets to at most two other receivers, no matter how large the group size is.

When a packet gets lost in the incoming link of a level-l node i in the Multicast tree, all downstream receivers of i will lose the packet. In this case, p_i will receive the repair packet from the level-(l+1) overlay ring it joins. After that, each downstream receiver of i transmits the repair packet to overlay successors after receiving it. Based on Theorem 2, the repair packet can be distributed to all downstream receivers of i.

In the example of Fig. 1, assume a packet is lost in the incoming link to $w4$, then all downstream receivers of $w4$ will miss the packet. when this happens, $v8$, the overlay proxy for $w4$, will receive repair packet from $v2$ via the level-4 overlay ring it joins, O_{v0}. Note that $v2$ buffers the packet because it does not receive ACK for the packet from $v8$. The repair packet is Unicast and can bypass the incoming link to $w4$ in the Multicast tree. After $v8$ receives the packet, it repairs it to $v9$ in the level-2 overlay ring of O_{v8}. Then $v9$ repairs to $v11$. Next, $v11$ sends the packet simultaneously to $v10$ and $v14$. Finally, $v14$ repairs to $v12$ in the level-2 overlay ring of O_{v12}, and all downstream receivers of $w4$ receive the packet.

When the sender receives no ACK for a packet from its overlay children, all receivers should lose this packet based on our design. Then the sender Multicasts the repair packet to the whole tree.

Repair Traffic: In typical cases, dominant repair traffic for a lost packet occurs in the lowest-level overlay rings. Note that the lowest-level overlay rings are usually composed of receivers connected to the same switch, so neighboring receivers in the overlay ring are only two-hop away. Compared with the most traffic-saving packet repair method using Multicast (or scoped Multicast), our repair scheme only doubles the overall repair traffic for receivers experiencing packet loss. But we require no intelligence from switches to realize complete repair isolation.

Repair Latency: The repair latency in RDCM is not high for two reasons. First, when a receiver joining two overlay rings receives a repair packet, it simultaneously sends out the packet to both its overlay successors in the two overlay rings, and thus the repair packet is transmitted in parallel on the backup overlay. Second, dominant hop-by-hop repairs occur in the lowest-level overlay rings, crossing only two physical links.

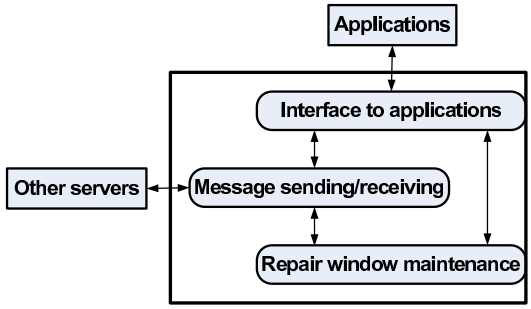
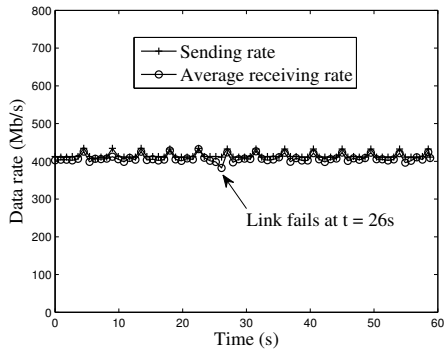


Fig. 3. Implementation architecture of RDCM.

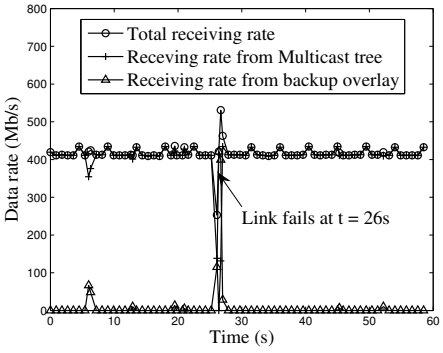
IV. Implementation and Experiments

We have implemented RDCM as a user-level library on Windows platform. Applications use legacy UDP/IP sockets for Multicast communication, which are intercepted by our library.

Fig. 3 shows the implementation architecture. The key components include repair window maintenance, interface to applications and message sending/receiving part interacting with Tree Manager as well as other data center servers. When the sender starts sending Multicast packets to a group, the first packet is intercepted and a message is sent to the Tree Manager. Data packets are sent out after the backup overlay is configured on all servers. Each packet is inserted with a sequence number. Receivers deliver received packets to upper-layer applications after removing the sequence number. The repair window is updated when receiving Multicast packets from either the Multicast tree or the backup overlay, or when receiving ACKs. In the implementation, we combine the ACKs for several packets into a single packet to reduce the processing overhead on servers. A receiver sends repair packets on the



(a)



(b)

Fig. 4. Packet repair in RDCM. The link of $w5 \rightarrow v5$ fails at t = 26s. (a) The data sending rate and average receiving rate of the Multicast session; (b) The data receiving rate at $v5$.

backup overlay when packet loss in its overlay successor is detected.

We conduct experiments on a 16-server testbed. Each server has one Intel 2.33GHz dualcore CPU, 2GB DRAM, and an Intel Pro/1000 PT quad-port Ethernet NIC. The OS installed on all servers is Windows Server 2003 Enterprise x64 Edition. The 16 servers are interconnected as a BCube(4,1) topology [5]. Switches support Gigabit Multicast forwarding. The Multicast tree is shown in Fig.ẽffig:retran. IP routing is used instead of BCube routing because currently BCube routing does not support Multicast. The MTU is set as 9000 bytes.

Packet Repair: We set the sender to generate packets at about 410Mb/s and observe the effectiveness of packet repair in RDCM. The initial Multicast tree is established as Fig. 1. At t = 26s, we shutdown the link of $w5 \rightarrow v5$. Fig. 4(a) shows the data sending rate at the sender and the average receiving rate on all receivers. We find that the receivers can receive Multicast data at the same speed of the sender throughout the whole session. The link failure has almost no impact on the Multicast throughput. It is because lost Multicast packets are gracefully repaired bypassing the failed link. In our experiment, after we shutdown the link, link failure is detected and the Multicast tree is adjusted to a new one.

The data receiving rate at $v5$ is further inspected and demonstrated in Fig. 4(b). We separate the receiving rates from the Multicast tree and from the backup overlay, the sum of which is the total receiving rate. We have the following observations from this figure. First, in the whole Multicast session, there is no obvious degradation of the total receiving rate on $v5$, even when congestion or link failure occurs. Second, during the interval between link failure at t = 26s and establishment of the new tree, $v5$ is receiving all packets from the backup overlay. But after $v5$ joins the new tree, it recovers receiving data from the Multicast tree. Third, there are always sporadic packet loss on $v5$, in either the old Multicast tree or the new Multicast tree. $v5$ gracefully handles this kind of light congestion by receiving lost packets from the backup overlay, without triggering tree adjustment.

V. Conclusion

We presented RDCM, a dedicated reliable Multicast proposal for data center networks. RDCM leverages the high link density in data centers to minimize the impact of packet loss on the throughput degradation of the Multicast session. A Multicast tree aware backup overlay is constructed upon group members for packet repair. When packet loss occurs, repair packet is transmitted in a peer-to-peer way on the backup overlay, which has high probability to bypass the congested/failed link in the Multicast tree where packet gets lost. Riding on Unicast, packet repair achieves complete repair isolation, which is critical for throughput enhancement. The backup overlay is carefully designed to control individual repair burden, control overhead as well as overall repair traffic. The experiments on a 16-server testbed show that RDCM can effectively achieve reliable data delivery by gracefully handling packet loss, while introducing low additional overhead.

References

[1] J. Dean and S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters", In *Proceedings of OSDI'04*, Dec 2004

[2] S. Ghemawat, H. Gobioff and S. Leung, "The Google File System", In *Proceedings of SOSP'03*, Oct 2003

[3] R. Mysore, A. Pamboris, N. Farrington and etc., "PortLand: A Scalable Fault-Tolerant Layer 2 Data Center Network Fabric", In *Proceedings of ACM SIGCOMM'09*, Aug 2009

[4] A.Greenberg, J. Hamilton, N. Jain and etc., "VL2: A Scalable and Flexible Data Center Network", In *Proceedings of ACM SIGCOMM'09*, Aug 2009

[5] C. Guo, G. Lu, D. Li and etc., "BCube: A High Performance, Server-centric Network Architecture for Modular Data Centers", In *Proceedings of ACM SIGCOMM'09*, Aug 2009

[6] A. Greenberg, J. Hamilton, D. Maltz and etc., "The Cost of a Cloud: Research Problems in Data Center Networks", *SIGCOMM CCR* 2009

[7] S. Kandula, S. Sengupta, A. Greenberg and etc., "The Nature of Datacenter Traffic: Measurements & Analysis", In *Proceedings of ACM SIGCOMM IMC'09*, Nov 2009

[8] D. Li, C. Guo, H. Wu and etc., "Scalable and Cost-effective Interconnection of Data Center Servers using Dual Server Ports", to appear in *IEEE/ACM Transactions on Networking*.

[9] S. Floyd, V. Jacobson, S. McCanne and etc., "Reliable Multicast Framework for Light-weight Sessions and Application Level Framing", In *Proceedings of ACM SIGCOMM'05*, Oct 1995

[10] S. Paul, K. Sabnani, J. Lin and etc., "Reliable Multicast Transport Protocol (RMTP)", In *IEEE Journal on Selected Areas in Communications*, 15(3):1414-1424, 1997

[11] J. Griffioen, and M. Sudan, "A Reliable Dissemination Protocol for Interactive Collaborative Applications", In *Proceedings of ACM Multimedia'95*, Nov 1995

[12] H. Holbrook, S. Singhal and D. Cheriton, "Log-Based Receiver-Reliable Multicast for Distributed Interactive Simulation", In *Proceedings of ACM SIGCOMM'05*, Oct 1995

[13] T. Speakman, J. Crowcroft, J. Gemmell and etc., "PGM Reliable Transport Protocol Specification", RFC3208, Dec 2001

[14] L. Lehman, S. Garland, and D. Tennenhouse, "Active Reliable Multicast", In *Proceedings of IEEE INFOCOM'98*, Mar 1998