DOI: 10.2298/CSIS120725024S

The Throughput Critical Condition Study for Reliable Multipath Transport

Fei Song[1,2], Huachun Zhou[1,2], Sidong Zhang[1,2], Hongke Zhang[1,2], and Ilsun You[3]

[1] School of Electronic and Information Engineering, Beijing Jiaotong University
100044 Beijing, P.R. China
{fsong, hchzhou, sdzhang, hkzhang}@bjtu.edu.cn
[2] National Engineering Lab for Next Generation Internet Interconnection Devices,
100044 Beijing, P.R. China
{fsong, hchzhou, sdzhang, hkzhang}@bjtu.edu.cn
[3] School of Information Science, Korean Bible University
Seoul, South Korea
isyou@bible.ac.kr

Abstract. Employing multiple paths for achieving high capability and robustness has some obvious benefits. New wireless technologies are giving more Internet access modes for notebooks and smart phones. However, most multipath protocols may not get acceptable throughput if reliable transmission is guaranteed. For some cases, the situation may even worse than using single path only. The main reason has strong relationship with the principal of multipath transmission. Motivated by these facts, specific analytical mechanisms are proposed to analyze the potential problems which may lead to serious performance decrease. Following that, we investigate how to use multiple paths legitimately when network environments are fluctuating. In our simulation, topologies for multiple paths and single path are set up for evaluating our analytical methods. Some distinguished scenarios are chosen from different perspectives. The results have revealed some throughput critical conditions and could be helpful in designing scheduling schemes for multipath protocols.

Keywords: reliable protocol, multipath transmission, critical conditions.

1. Introduction

The Internet terminal has been using one path for data transmission since the beginning. Expensive network interface and unitary access method lead to such circumstance. Single path protocols (like TCP and UDP in transport layer) are working very well even in complex network scenarios, such as Cloud computing [1], Optical networks [2], Wireless sensor network [3], etc. However, using one path may bring some insoluble problem. For example, when the path failure occurs, the users have to wait a long period for

recovery. Therefore, improving the transmission quality is always the significant target for protocol designers.

With the development of wireless technologies, people could use 802.11, 802.15, 802.16 and other methods to access the Internet, which provide the terminal a sense of possibility to adopt multiple paths for data transmission. As a promising solution for enhancing capability, reliability, security and mobility, multipath transport was widely discussed in both academic and industrial communities. New generation transport protocols, like SCTP [4] and DCCP [5], consider using multiple paths as backups. That means the data packets are still sent on one main path, only heartbeat packets are interchanging on other paths to keep them alive. This kind of mechanism could solve the problem suffered in single path transport (mentioned in previous paragraph). The users do not need to wait a long recovery time, but to use the pre-assigned path immediately. The throughput could be improved obviously. Although this scheme steps forward a little in multipath, the satisfaction is still far from enough. Concurrent multipath transfer is highly needed.

Any new technologies need an appropriate timing for expanding. The hypergrowth of notebooks and smart phones is giving multipath transport a golden opportunity to implement the academic ideas into industrialization. For example, the hardware platform and phone operation system (such as iOS or Android) could provide high speed computing and stable Internet access (via GPRS, 3G, WiFi, Bluetooth or Infrared). Many applications in smart phone are calling for high bandwidth capability based on multipath transport as well.

Using multiple paths simultaneously is quite helpful not only in aggregating the bandwidth, but also in obtaining the service from suitable Internet Service Providers (ISP) rapidly. Imaging a fancy service is located in the server of ISP A, the user who wants to get the service has two access authorities to both ISP A and ISP B. If the default setting is to connect with ISP B, all the data packets have to go through the Internet Exchange Point (IXP), which may cause evident increasing delay. Multipath transport could mitigate this by creating two connections from two ISPs, respectively. If only single path is mandatory for this service, the path created inside the same ISPs will be elected. If multiple paths are allowable, all the paths should be employed as soon as possible to boost the throughput.

The motivation of this paper is following current multipath research statues: Comparing with single path transmission, sending packets on multiple paths should have better throughput. However, that is not always true as expected in the realistic cases due to some potential problems [6]. Performance should be analyzed more carefully to figure out when the multipath should be adopted and when the single path could get higher performance. We only focus on the reliable multipath transmission here.

The main contributions of this paper are:

Proposed a specific mechanism for analyzing the potential problems which may lead to performance decline.

Studied some critical conditions by setting up suitable topologies and evaluating the analytical mechanism on both multiple paths and single path scenarios.

The ordinary principle of mainline multipath transport protocols will be discussed in Section 2. For the Internet terminals, the multipath protocols in transport layer have to consider many mechanisms to guarantee the reliable transmission. This leads to some potential problems which may influence the performance (detailed in Section 3). Some fundamental formulas are given in this Section as well. The evaluation part, given in Section 4, contains topologies description, parameters setting, performance analysis and comparison. In Section 5, related work is introduced and classified based on relevance. The conclusions and future works are explained in Section 6.

2. Principle of Multipath Transport

In multipath transport, data packets should be ejected and routed on more than one path. The end hosts may have several network interfaces and the multiple paths may have crossover point. Theoretically, no matter how complex the network scenario is, from the users' perspective, the terminals only need to care about the scheduling and reflection mechanism. In order to achieve reliable transmission, the sender needs to collect all the feedbacks giving by the network and receiver to adjust corresponding schemes.

As an indispensable part of reliable transport, schedule mechanism is responsible for controlling sending speed and sequences. For original TCP, there are some schemes to guarantee the performance of data transmission on single path. When multipath was introduced, all these schemes need to be investigated or modified.

Congestion Control: The multipath could be treated as independent individuals or a whole entity when facing the congestions. That means different congestion control schemes should be designed. The legacy of TCP could be used as reference. Some researchers deem that all the available paths should be optimal used to shift the congestion and obtain load balancing. The consensus of designing multipath protocols is to realize fairness and efficiency at the same time.

Flow Control: When TCP was just getting start running in the old days, the performance of receiver host was not very satisfactory. Especially if "high speed network" is connected, the arriving rate of data packets might be higher than the processing capability of receiver. Even for current modern equipments, establishing too many network connections (like P2P applications) may decrease the performance as well. So in multipath transport scenarios, flow control is also a critical problem. More than that, some recent works have expanded flow control to solve power consumption problems [7].

Sequence Control: For some software inside application layer, data packets are expected to arrive at the destination in order. Due to the parameter variation or restriction policies of ISP network, packets might be

halted inside some routers and lead to disorder situation. It is quite widespread in multipath transport. Assigning Transmission Sequence Number (TSN) to each packet and using them to distinguish out of order are always necessary for reliable multipath transmission.

Retransmission Control: The packet loss is inevitable in the Internet due to current switching and forwarding rules of network infrastructure. It is not hard to perceive which packet is lost based on the information carried by acknowledgments (ACK). The multipath protocols could set reasonable timers for calculation. Path selection in retransmission is also important and might affect the transmission fairness. The core idea in this part is to let the receiver host obtain the lost packet as soon as possible.

It seems that most controlling work are assigned to the sender side, and the receiver is just responsible for collecting and submitting the data packets into application layer, then send the acknowledgements back to the sender. In fact, receiver provides a lot of information in the process of data transmission. Both of them have to deal with some potential problems for enhancing the throughput when using multipath.

3. Potential Problems Analysis

For each Internet end host, there are send buffer and receive buffer in charge of storing unconfirmed output packets and trimming the disorderly input packets, respectively. Both of them may encounter a serious blocking event, which is able to decrease the performance according to the principle of current reliable transport on multiple paths and single path.

3.1. Send buffer blocking

There are two types of data packet inside send buffer: New data packets and retransmission data packets. Each output packets will be added at the tail of retransmission queue and removed when it is delivered to the upper layer by the receiver. If the space in the send buffer is too narrow for sending new packets and the sending speed has been seriously affected, the situation could be called send buffer blocking.

Reliable single path protocols in transport layer, such as TCP and SCTP, need to gather the status information from ACKs. Multipath inherits this method when exchanging data packets. Original ACKs could only indicate the earliest data packets the receiver needs. Although some packets have arrived and stored at receiver's buffer, the sender may still consider these packets have been lost. This will lead to unnecessary degradation of sending speed.

Selective Acknowledgment (SACK) is used to inform the sender how many packets have been received successfully in both ordered and disordered status. It is quite helpful for the sender to release the send buffer. The TCP throughput could be improved obviously [8] [9]. Nowadays, more and more

web servers support SACK. However, based on the RFC2018 [10], the receiver has the right to SACK some out-of-order packets and then discard them for some particular reasons (such as the operating system needs to reuse previously allocated memory for other processes). This action is called "data receiver reneging". In order to tolerate the renege, the senders of reliable protocols have to store the copies of the SACKed data in its send buffer.

The receiver of TCP never deliver out-of-order packet to upper layer. However, In SCTP, multiple streaming is the default option which means the packets with continuous TSN may be assigned to different streams. Therefore, out-of-order packets with different TSN may belong to the same stream and have continuous Stream Sequence Number (SSN). These packets could be delivered to the application layer directly. If that happens, they become non-renegable. The problem is the sender could not distinguish the "real" disordering packets stored in the receive buffer from the "fake" disordering packets which has been delivered to the upper layer. That will lead to send buffer waste because only the cumulatively ACK could trigger the free up action for relevant packets. That means all the SACKed packets need to be reserved at send buffer even they have been delivered. Such situation could ascribe to the send buffer blocking.

Non-renegable selective acknowledgments (NR-SACKs) [11] is proposed to cope with above problems. The idea is to modify the original SACK packet by adding new field for reporting non-renegable packets. The performance of new method is tested both on multiple paths and single path scenarios [12]. Send buffer could be released efficiently by using NR-SACK based on the simulation results. This method is quite helpful for all the reliable protocol in transport layer that uses SACK and allow delivery of disordered data to upper layer. The authors deem that if data packet renege is rarely in the current Internet, NR-SACK would not bring too much burdens for both sender and receiver, which means that understanding the probability of renege is critical for deploying the NR-SACK.

3.2. Receive buffer blocking

The appearance of receive buffer blocking is quite common in reliable transmissions. In multipath transport scenario, all available paths have their own characteristics. Data packets sent on different paths may not keep in sequence when they arrive at the destination. These out-of-order packets have to be stored in the receive buffer. If there are a lot of these kinds of packets, the free space in the buffer will be quite small. Normally, the sending speed of reliable protocols is controlled by the size of local congestion window and free space of peer receive buffer, which will be detailed in the following. Therefore, the throughput of multipath transport might be limited by the receive buffer blocking.

For current Internet, there are three potential reasons may lead such phenomena. To facilitate the presentation, we employ two paths for illustration in this Section and below.

Buffer size related: In most implementations of transport layer protocols, there are a default size of receive buffer. For the single path transport, disordered packets are not quite prevalent (comparing with multiple paths environment). Even that, the relationship between receive buffer size and performance is also quite important. When sending the data packets via multipath, the setting of receive buffer size is hard to determine. The protocol could assign each path an independent receive buffer. However, the sum of buffer size on all paths is still finite and restrict packet deliver to the application layer. So this mechanism would not gain a lot. The general idea of the receive buffer size effect is that lager space could hold more out-of-order data packets. Meanwhile, larger space will bring more burdens for the operating system as well. The relevance between buffer size and transmission performance still needs more experiments.

Packets loss related: Basically, packets loss caused by intermediate routers or the links is quite common in the Internet. For the purpose of avoiding the heavy congestion, routers will drop some packet based on the different schemes (such as Drop Tail or RED). In wired network, link error loss is less than that in wireless network. Beside these two reasons, some loss event may appear when the handover occur in wireless environment. If there is a loss event on one path, packets sent on other paths could not reach the receiver side in order. For comparing the detail, we propose a mechanism for analyzing the whole process with “Non packets loss related” together in the following.

Non packets loss related: Receive buffer blocking may be triggered as well even if there is no packet loss in multipath transport. Different bottleneck bandwidth and transmission delay are able to generate out-of-order packets in the sending process. The value of bottleneck bandwidth determines forwarding speed at the intermediate routers. It will take long periods for a router which has low output bandwidth to send the data packets to the next hop. If the rate of ingress is higher than the rate of egress, some data packets have to wait in the queue of intermediate router.

Fig. 1 illustrates that multipath transport flows and other flows are sharing the same route. The packets of multipath were sent on two paths which have different bottleneck bandwidth. The value of bandwidth on path B is larger than another path. So for path A, there are more packets waiting inside the buffer of router. In Fig. 1, the number printed in each packet is TSN. Data packets sent from source are not able to arrive at destination in order. These packets will occupy much space in receive buffer.

The diversity of hops on available paths which are set up at the initial stage of connecting may bring different transmission delay in multipath transport. It is possible that the values of transmission time on different paths are not similar even if the paths have the same hops. In Fig. 2, there are two access modes (GPRS and 802.15x) for the sender to connect to the Internet. The receiver has two access modes (CDMA and 802.11x) as well. The data

packets which were sent from different interfaces may not be able to arrive at destination simultaneously.

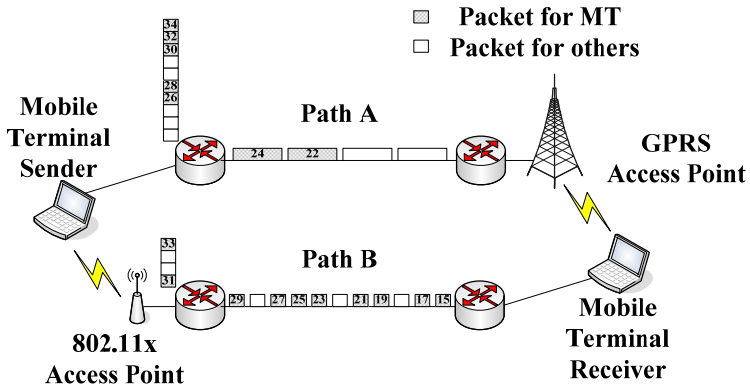

Fig. 1. Bottleneck bandwidth variety (with other flow)

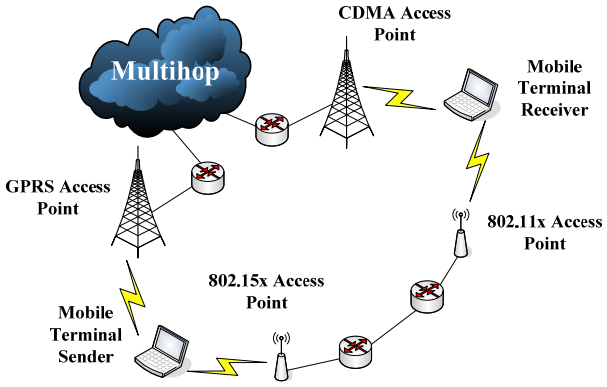

Fig. 2. Transmission delay variety

The reasons which may cause receive buffer blocking can be illustrated in Fig. 3 and Fig. 4. They are very useful for understanding the whole blocking process. CMT-SCTP proposed in [13] is used as an example in this Section.

To make it more simplify, we assume that delay ACK is not adopted. However, the following mechanism is also serviceable for delay ACK scenarios. When payload size of one packet is 1468 Bytes (It is according to the definition of CMT-SCTP. In TCP, the maximum size of payload is 1460 Bytes), the receiver is able to contain only 11 packets if the size of receive buffer is set to 16KB. Here are some definitions:

C*x*: the size of congestion window (*Cwnd*) on path *x*.

O*x*: the number of unacknowledged packets (i.e. Outstanding packets) which was sent on path *x*.

A*x* and B*x*: the interfaces of sender and receiver, respectively. Two interfaces are connected if the value of *x* is the same.

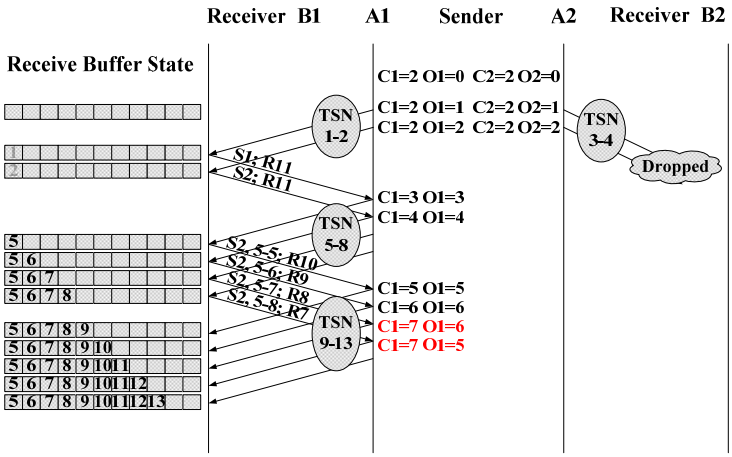

Fig. 3. Receive buffer blocking caused by packets loss

<S*a*, *b*-*c*; R*d*>: *a* indicates accumulative TSN. *b*-*c* means the packets with TSN between *b* and *c* have been acknowledged by gap. *d* means the size of free space in the receive buffer (*Rwnd*). The unit of all the parameters is packet.

Arrow lines indicate the process of sending and receiving packets.

The TSN of each packet is printed at the beginning of arrow lines.

There are 11 squares at the left side in the figures to show the state of receive buffer. It will be changed whenever the data packets arrive at receiver side. Only the latest state is shown when two packets which were sent on different paths reach the destination almost at the same time.

Inside receive buffer state squares, the transparent number indicate the TSN of packets which have been submitted to application layer. These squares with transparent number are just used as an identifier for submission process and will not take up any space in receive buffer. For instance, in Fig. 4, when the receive buffer state is showing: 13, 9 (written in transparent style), 14, there are only two packets in the buffer actually. And the sequence of them is 13 and 14.

Based on above preparations, we could describe the details about how the receive buffer blocking happens in packet loss situation (shown in Fig. 3). After hand shake stage, the initial congestion window on both paths is set to 2. The number of outstanding packets is 0. Then, the packets whose TSN is 1, 2 and 3, 4 will be sent from path 1 and 2, respectively. If the packets which were sent on path 2 are lost on the forward direction, the packets with TSN 1-2 will be submitted to application layer as soon as they arrive. The corresponding acknowledged packets will be generated and sent from receiver side. The accumulative TSN is 2 and the size of free space in the receive buffer is still 11.

The sender increases the value of *Cwnd* to 3 when the acknowledgement of packet with TSN 1 was received. Outstanding number decrease to 1(not

shown in Fig. 3). The number of packet which is permitted to send can be calculated by the formula (1) and (2):

$$SendWnd = \min\left(Cwnd,\ PeerRwnd\right) \quad (1)$$

$$PeerRwnd = Rwnd - \sum_{i=1}^{n} Outstanding_i \quad (2)$$

where n is the number of path which used for sending data packets. $Outstanding_i$ means the total number of unacknowledged packets on path i. In this case, the value of outstanding packets on path 1 and path 2 are 1 and 2, respectively. R11 indicates the free space in *Rwnd* is 11. Based on formula (2), the *PeerRwnd* is 8. The value of *Cwnd* on path 1 is 3. Therefore, the *SendWnd* is 3 according to the formula (1). Due to the packet with TSN 2 has not been acknowledged, the sender can only send 2 packets. After sending these packets, the number of outstanding packets is changed to 3. These figures only illustrate the final value of outstanding packets when all the permitted packets have been sent on one path. Transition states discussed here are not shown in the figures.

The *Cwnd* will be changed to 4 when the acknowledgment of TSN 2 was received. Outstanding value is changed to 2. According to the previous method, 2 packets are allowed to be sent. Then the value of outstanding packet is increased to 4.

When packets with TSN 5-8 arrive at the destination, the receiver can not submit them to the application layer at once because the TSN 3 and 4 have been lost. Therefore, the accumulative TSN at the receiver side is still 2 and packets with TSN 5-8 can only be acknowledged by gap in SACK. Meanwhile, the free space in receive buffer has been changed from 11 to 7.

When the acknowledgments of packets with TSN 5 and 6 have been received, the previous analysis method remains correct because *Cwnd* is smaller than *PeerRwnd*. However, the value of *PeerRwnd* will be 1 when the acknowledgment of packet with TSN 7 was received. That means only 1 packet can be sent. After sending this packet, the outstanding number will be changed to 6. We use red character to identify the sender has been influenced by receive buffer blocking. When the acknowledgment of packet with TSN 8 was received, the number of outstanding packets on path 1 is changed to 5. The *PeerRwnd* is 0 because both the *Rwnd* and outstanding packets on two paths are 7. Therefore, no packets can be sent. Only five packets were sent after the acknowledgments of packets with TSN 5-8 were received.

In non packet loss situation (shown in Fig. 4), the packets whose TSN is 1, 2 and 3, 4 will be sent from path 1 and 2, respectively. All these four packets can be submitted to application layer because they arrive at destination in order. The sender change the value of *Cwnd* and outstanding packets value when the acknowledgments of the packets with TSN 1-2 were received, then the sender will calculate *SendWnd* based on (1) and (2). After that, the packets with TSN 5-8 are sent via path 1.

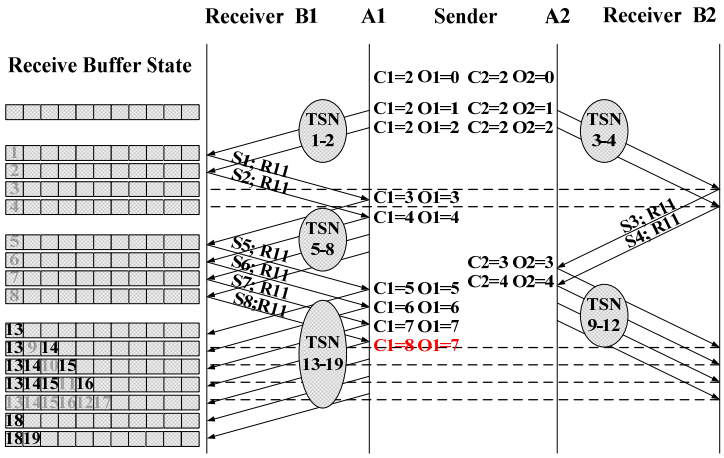

Fig. 4 Receive buffer blocking caused by non packets loss

For the path 2, when the SACKs of packets with TSN 3-4 are received, the sender will update the *Cwnd* and outstanding packets value again, and then transmit the packets with TSN 9-12 to the receiver. The C2 and O2 are equal to 4 when this transmission round is finished.

The *Cwnd* is always smaller than *PeerRwnd* when the SACKs of packets with TSN 5-7 are received. We can use similar method for analysis. However, when the acknowledgment of packet with TSN 8 arrives at the sender side, only 1 packet can be sent. Then the value of outstanding packet will not increase. So 7 packets were sent after the acknowledgements of packets with TSN 5-8 arrived. Receive buffer blocking happens again.

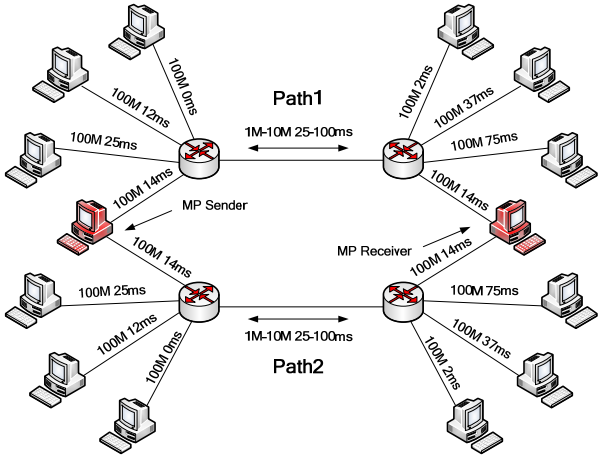

Fig. 5 Topology for multiple paths transport

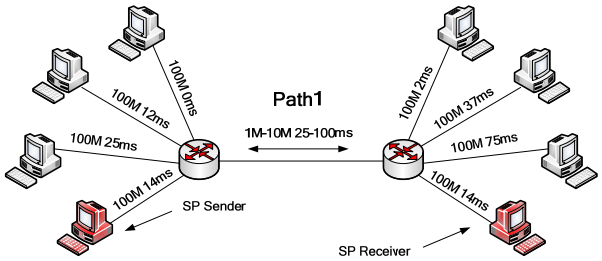

Fig. 6 Topology for single path transport

4. Performance Simulation

In Section 3, the main potential problems which may lead to performance decrease in reliable multipath transport were analyzed. We argue that if these problems emerge frequently, using single path might be a better choice to get higher throughput. To evaluate this idea, we set up two topologies following the guidelines proposed in [14] and adopt the CMT-SCTP and original SCTP as the object reliable multipath and single path protocols, respectively. All the simulations are running in NS2-2.35 [15]. The basic structures of multiple paths (MP) and single path (SP) topologies are showed in Fig. 5 and Fig. 6.

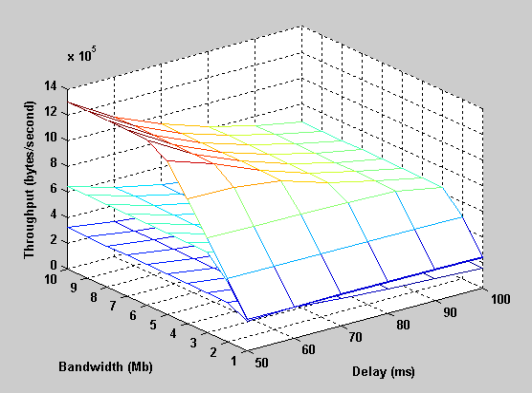

Fig. 7 Multipath throughput comparison for Scenario One

Two multipath transport nodes shown in red are adopted and other 12 nodes are used to generate application layer traffic. FTP, HTTP and UDP flows are sent randomly from left nodes to the right nodes. The value of bandwidth and delay between terminals and routers are fixed. The network parameters of bottleneck are set dynamically in different scenarios. The drop tail routers are used in our simulations. For the CMT-SCTP, three algorithms

(CUC, SFR and DAC) introduced in [16] are employed to ensure the transport quality. Chunk size is set to 1468 bytes. The send buffer of CMT-SCTP and SCTP terminals is set to the default size. The receive buffer size are changed in various experiments. Random seeds are used to run each experiment for 30 times.

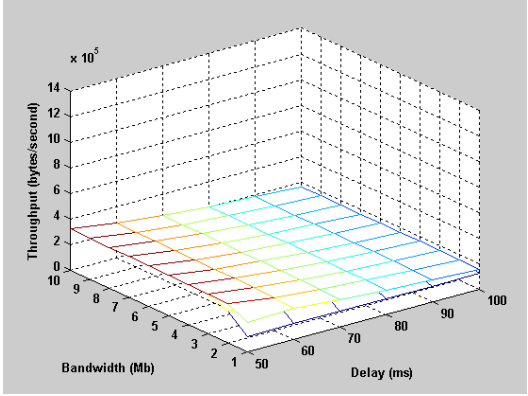

Fig. 8 Multipath vs Single path (buffer size = 32K) for Scenario One

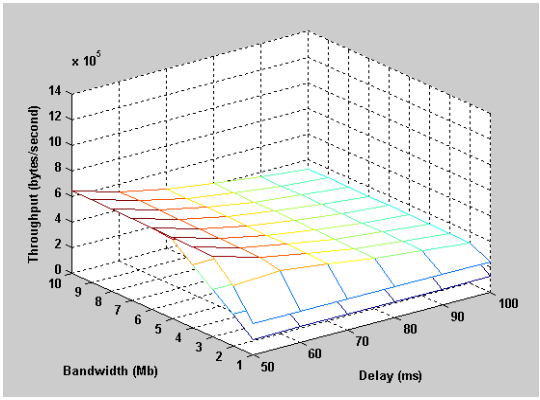

Fig. 9 Multipath vs Single path (buffer size = 64K) for Scenario One

In order to compare the transport performance between multiple paths and single path, several pretreatments are needed. Based on the previous analysis, NR-SACK could be used to enhance the utilization rate of send buffer. We enable the NR-SACK in all simulations. However, due to the restriction of receiving and delivering procedure, the influence of receive buffer blocking needs more investigation.

There are three possibilities which may aggravate receive buffer blocking and decrease the throughput. We set up three different scenarios for testing the performance.

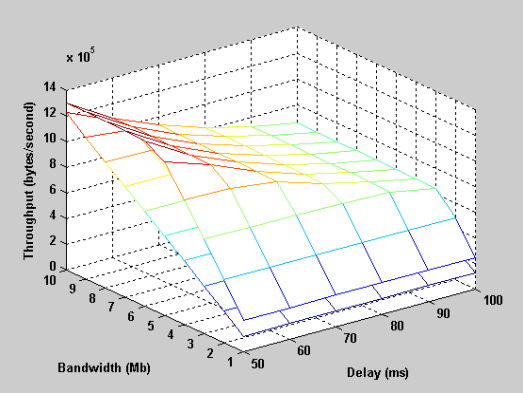

Fig. 10 Multipath vs Single path (buffer size = 128K) for Scenario One

4.1. Scenario One: influence of receive buffer size

The values of bandwidth and Round Trip Time (RTT) on the bottleneck links are modified. The value of bandwidth is changing from 1 M to 10 M (step size is 1M). The RTTs fluctuate from 50ms to 100ms (step size is 10ms). In order to clearly specify the relationship among bandwidth, RTTs and throughput, we did not add any packets loss in this scenario.

Three-dimensional figures could be used to illustrate the results. In Fig. 7, when the receive buffer size is set to 32K, 64K and 128K, the throughput is getting higher and higher. Based on analyzing the trace files, sending window is limited by the *PeerRwnd* if small buffer size is set. With the increasing of receive buffer, the limitation for the sender becomes weakening. There are not too much out-of-order data packets because the parameters of each bottleneck link are modified simultaneously.

We could find that the variation of RTT bring little changing in larger receive buffer size case if the bandwidth is low, which means the fluctuation of throughput is smaller than 1%. With the growth of bandwidth, the disadvantages brought by large RTT value are gradually obvious. In the 128K case, the throughput is reduced 49.8%. In other cases, the decrement rate is also around 50%.

However, the bandwidth plays an important role if larger receive buffer size are assigned. In 128K case, when the RTT is set to 50ms, the throughput will increase 424.2% if the bandwidth changed from 1M to 10M. The growth rate is getting low if smaller buffer size is set. In the case of 64K and 32K (RTTs are equal to 50ms as well) the performance will be increased 163.9% and 39.6%, respectively.

For analyzing the benefits given by larger receive buffer size, we fix the bandwidth and RTT to the best case and worst case to check the performance. When double and quadruple the 32K, the throughput increase

99.7% and 303.6% if 10M and 50ms are adopted. That means the growth rate is in proportion to the receive buffer size in the best case. However, the performance only increase 54.2% and 60.6% when amplifying buffer size to 64K and 128K in the worst case, i.e. bandwidth and RTT are equal to 1M and 100ms.

We compare the multiple paths and single path performance in Fig. 8, Fig. 9 and Fig. 10 scenario. The general view is that the disadvantage of multipath is not shown up. All results indicate single path could not get higher performance. We also find two main features: One is the throughput enhancement is quite obvious (86%, 96% and 98% in 32K, 64K and 128K, respectively) when low bandwidth and high RTT are set. The other feature is that marginal effect for multiple paths transport throughput is more serious than that in single path transmission when the bandwidth is varying from 1M to 10M.

4.2. Scenario Two: influence of packets loss factor

The packets drop will trigger the receive buffer blocking quickly. To simulate such process and test the performance, we run some experiments which fix RTTs (50ms on both paths) and change other parameters. The value of bandwidth is set from 1M to 10M and the loss rate is increased from 0.01 to 0.08. Receive buffer size are set to 32K, 64K and 128K.

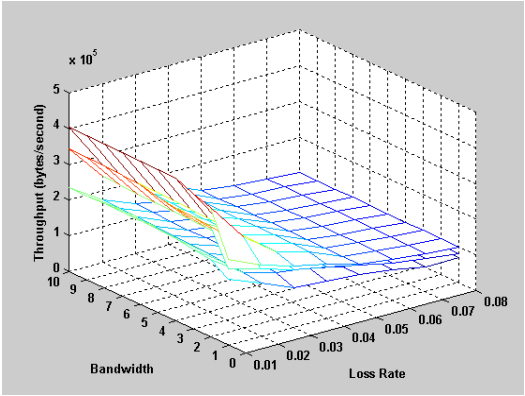

Fig. 11 Multipath throughput comparison for Scenario Two

In Fig. 11, three curved surfaces are used to illustrate and compare the performance of multipath transport. Generally, the larger receive buffer size is, the more throughput one could get. The difference among three curved surfaces is not significant. If the loss rate is getting higher (larger than 0.04), the variation tendency of them are quite similar. With the increase of bandwidth, the influence given by packet lost could be remitted, which enhance the multipath throughput. One interesting phenomenon, like in

scenario one, is that the marginal effect are highlighted when the bottleneck bandwidth is larger than 2M. We could find the point of inflection is smaller than that in scenario one.

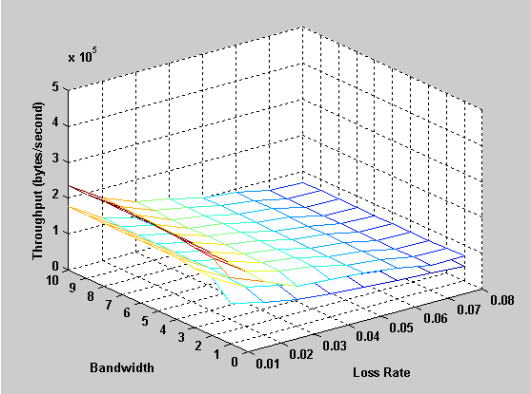

Fig. 12 Multipath vs Single path (buffer size = 32K) for Scenario Two

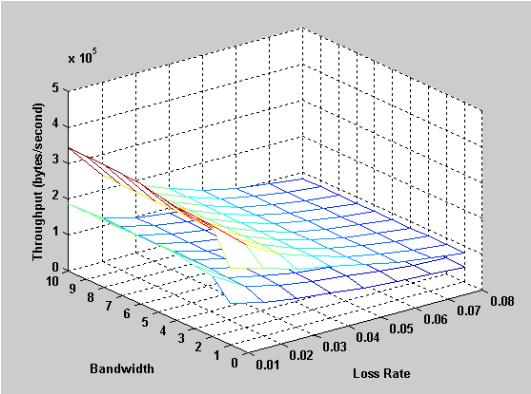

Fig. 13 Multipath vs Single path (buffer size = 64K) for Scenario Two

In Fig. 12, Fig. 13 and Fig. 14, single path transport results are added for comparison when the buffer size is set to 32K, 64K and 128K. Although, the bandwidth and loss rate are assigned to different value in the experiments, which may aggravate the receive buffer blocking, the throughput of multipath is always higher than the values of single path transport. The results show the performance is enhanced 36.3%, 84.1% and 114.5% in the best case.

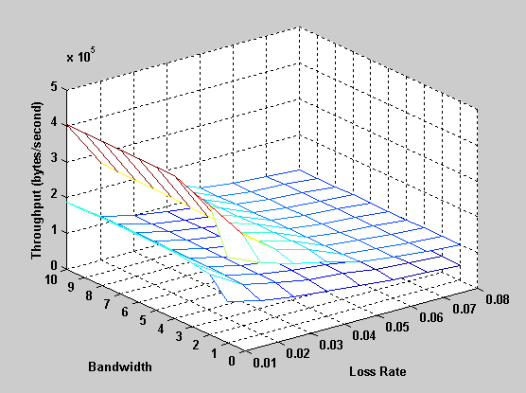

Fig. 14 Multipath vs Single path (buffer size = 128K) for Scenario Two

4.3. Scenario Three: influence of non packets loss factor

In this scenario, we fix the bandwidth of bottleneck link to 10M and vary the receive buffer size and RTT. Based on the analysis in Section 3, different RTT on two paths may block the receive buffer. The first experiment, shown in Fig. 15, is to change the RTT (from 50ms to 100ms) of the alternate path and leave main path RTT (50ms) untouched. 32K, 64K and 128K receive buffer size are still used here. When two paths have same RTT, the throughput of multiple paths and single path are quite similar if the buffer size is set to 32K or 64K, which means the performance is restricted mainly by the size of buffer. Such situation is released when the buffer size is 128K. With the increase of RTT on alternate path, the throughput of multipath is getting down. The decrease rate in 32K, 64K and 128K are 47.3%, 41.5% and 48.7%, respectively. The multipath (shown in solid line) is defeated by single path (shown in dash line) when the RTT is higher than 55ms.

The second experiment shown in Fig. 16 explores the performance variation when changing two paths' RTT simultaneously and maintaining the sum RTT of two paths (equals to 100ms). Due to the RTT of main path is getting smaller, the throughput of single path will increase. Although the curve of multipath has fluctuation, it is lower than the curve of single path in most cases. When the receive buffer size is set to 64K, the decrease rate will reach 64.1%. We could find the variation tendency of multipath and single path are quite different with the changing of RTTs.

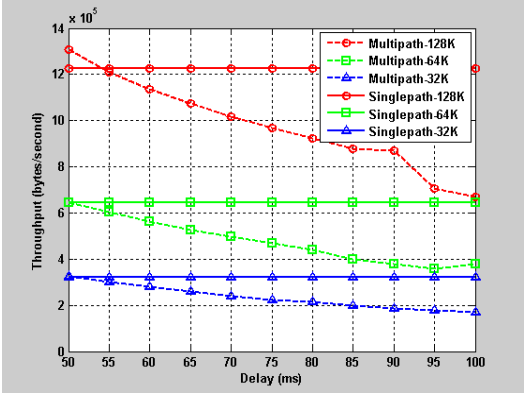

Fig. 15 Multipath vs Single path (adjust RTT in one path) for Scenario three

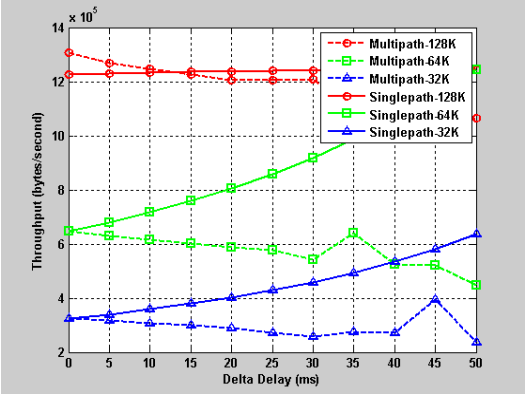

Fig. 16 Multipath vs Single path (adjust RTT in two paths) for Scenario Three

5. Related Work

Researches related with reliable transport on multiple paths have different emphases. Multipath TCP or SCTP should be discussed firstly due to correlation of our work.

Based on the principle of resource pooling [17], some new mechanisms were introduced and evaluated. The main goal of designing MPTCP [18] [19] is to be deployable and usable without significant changes to existing Internet infrastructure. In [20], an effective proposal for large-scale data centers is presented to enhance the throughput and fairness. Raiciu et al. [21] proposed an opportunistic-based mobility method by using multipath TCP. In order to settle the congestion problem in MPTCP, Wischik et al. [22] designed a multipath congestion control algorithm and implemented it into Linux system. The adoption of MPTCP in current network was analyzed from business

perspective in [23], which might be quite helpful in understanding the critical issues in the deployment process of multipath TCP.

In send buffer blocking part, although NR-SACK or other renege solutions are quite helpful in mitigating the blocking problem, investigations for occurrence frequency of data renege in current Internet are highly needed. A reasonable way is trying to process the dataset gathering from the real network environment. In [24], TCP traces obtained from Cooperative Association for Internet Data Analysis (CAIDA) [25] was analyzed to infer the state of receive buffer. The first step results show that data reneging appears constantly. However, the further work [26] point out that the generation of SACKs in many TCP implementations was unreasonable comparing with the RFC2018. Some necessary SACKs were wrongly sent or even not sent. Seven misbehaviors were demonstrated and discussed to show the risks. A series of extensions mechanisms which could be added into TBIT [27] were proposed for detecting these misbehaviors.

For the research of receive buffer blocking, Iyengar et al. [28] compare different retransmission schemes in finite receive buffer size to see which one is more suitable for reducing receive buffer blocking in concurrent multipath transfer (CMT). More details can be found in [16]. Natarajan et al. [29] added a new state called “Potentially-failed” into original CMT mechanism to reduce the receive buffer blocking. It can provide acceptable throughput when the path failure occurs. However, these works are not able to improve the performance in non packet loss related situation. Increasing the size of receive buffer can relieve this problem in all scenarios obviously, but it makes no sense for these hosts which do not have enough resource. The receive buffer blocking is still an open issue.

6. Conclusions and Future Work

The multipath transport has been attracting the research attention for some time. It is a reasonable solution for improving the capability, reliability, security, mobility and other property in the Internet. Normally, comparing with single path transport, people consider that sending data packets on multipath should have better performance. We argue such an opinion would not always hold water for the reliable multipath transport. In this paper, we proposed a specific mechanism for analyzing the potential problems which may lead to performance reduction. Then some suitable topologies for evaluating the analytical mechanism on both multiple paths and single path scenarios were set up.

The evaluations are divided into three scenarios. For the first scenario, when changing the size of receive buffer, all experiments are affected. Using multipath shows better throughput than adopting single path. An interesting finding is the marginal effect for multipath is more significant than single path. It is also confirmed via scenario two. In addition, the performance contrast of the second scenario is shown when the bandwidth and drop rate are adjusted

simultaneously. In scenarios three, the insufficient of multipath are revealed when fix the bandwidth and adjust the receive buffer size and RTTs, which explain the transmission on single path could beat multipath in some specific environments. More details, including increasing and decreasing rate for each experiments, could be found in the performance simulation part.

This paper is focusing on the end to end perspective. A more challenging question is trying to enable multipath in the whole network and validate the performance in different scenarios, which should be studied in the future work.

Acknowledgment. This work was supported in part by the SRFDP under Grant No. 20120009120005, in part by the Natural Science Foundation of China under Grant No. 61271202, in part by the MIIT of China under Grant No. 2012ZX03005003-04, in part by the Beijing Natural Science Foundation under Grant No. 4122060.

References

1. T. Lee, H. Kim, K. H. Rhee, and S. U. Shin. A Study on Design and Implementation of E-Discovery Service based on Cloud Computing, Journal of Internet Services and Information Security, 2(4), pp. 65-76, Nov. 2012.
2. V. Q. Bien, R. V. Prasad, and I. Niemegeers, Handoff in Radio over Fiber Indoor Networks at 60 GHz, Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications, 1(3), pp. 71-82, Sep. 2010.
3. Ž. Živanov, P. Rakić, and M. Hajduković, Wireless Sensor Network Application Programming and Simulation System. Computer Science and Information Systems, Vol. 5, No. 1, 109-126. 2008.
4. R. Stewart, Q. Xie, K. Morneault, C. Sharp, H. Schwarzbauer, T. Taylor, I. Rytina, M. Kalla, L. Zhang, and V. Paxson, Stream Control Transmission Protocol, RFC 2960, IETF, Oct. 2000.
5. E. Kohler, M. Handley, and S. Floyd, Datagram Congestion Control Protocol (DCCP), RFC 4340, IETF, Mar. 2006.
6. F. Song, H. Zhou, S. Zhang, H. Zhang, and I. You, Performance Analysis of Reliable Transmission on Multiple Paths and Single Path, IMIS Innovative Mobile and Internet Services in Ubiquitous Computing, 2012.
7. T. Enokido, A. Aikebaier, and M. Takizawa, Computation and Transmission Rate Based Algorithm for Reducing the Total Power Consumption, Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications, 2(2), pp. 1-18, Jun 2011.
8. R. Bruyeron, B. Hemon, and L. Zhang, Experimentations with TCP Selective Acknowledgment, ACM Computer Communication Review, 28(2), pp. 54-77, 1998.
9. M. Allman, C. Hayes, H. Kruse, and S. Ostermann, TCP Performance over Satellite Links, 5th International Conference on Telecommunications Systems, 1997.
10. M. Mathis, J. Mahdavi, and S. Floyd, A. Romanow, TCP Selective Acknowledgment Options, RFC2018, IETF, Oct. 1996.

11. P. Natarajan, N. Ekiz, E. Yilmaz, P. Amer, J. Iyengar, and R. Stewart, Nonrenegable Selective Acks (NR-SACKs) for SCTP, International Conference on Network Protocols (ICNP), Orlando, 2008.
12. E. Yilmaz, N. Ekiz, P. Natarajan, P. Amer, J. T. Leighton, F. Baker, and R. Stewart, Throughput Analysis of Non-renegable Selective Acknowledgments (NR-SACKs) for SCTP, Computer Communications, 33(16), 2010.
13. J. Iyengar, P. Amer, and R. Stewart. Concurrent Multipath Transfer Using SCTP Multihoming over Independent End-to-end Paths. IEEE/ACM Transactions on Networking, 14(5):951-964, 2006.
14. L. Andrew, C Marcondes, S. Floyd, L. Dunn, R. Guillier, W. Gang, L. Eggert, S. Ha, and I. Rhee, Towards a Common TCP Evaluation Suite. In PFLDnet, 2008.
15. The Network Simulator NS-2, version 2.35. www.isi.edu/nsnam/ns/.
16. J. Iyengar, P. Amer, and R. Stewart, Performance Implications of a Bounded Receive Buffer in Concurrent Multipath Transfer, Computer Communications 30 (2007) 818-829.
17. D. Wischik, M. Handley and M.B. Braun, The Resource Pooling Principle, ACM SIGCOMM Computer Communication Review, 38(5), Oct. 2008.
18. Ford, C. Raiciu, M. Handley, S. Barre, and J. Iyengar, Architectural Guidelines for Multipath TCP Development, RFC 6182, IETF, Mar. 2011
19. M. Bagnulo, Threat Analysis for TCP Extensions for Multipath Operation with Multiple Addresses, RFC 6181, IETF, Mar. 2011.
20. Raiciu, S. Barre, C. Pluntke, A. Greenhalgh, D. Wischik, and M. Handley, Improving Datacenter Performance and Robustness with Multipath TCP, ACM SIGCOMM, 2011.
21. Raiciu, D. Niculescu, M. Bagnulo, and M. Handley, Opportunistic Mobility with Multipath TCP, ACM MobiArch, 2011.
22. Wischik, C. Raiciu, A. Greenhalgh, and M. Handley, Design, Implementation and Evaluation of Congestion Control for Multipath TCP, Usenix NSDI, 2010.
23. T. Levä, H. Warma, A. Ford, A. Kostopoulos, B. Heinrich, R. Widera, and P. Eardley, Business Aspects of Multipath TCP Adoption, In Towards the Future Internet-Emerging Trends from European Research, IOS Press, 2010.
24. N. Ekiz, P. Amer, A Model for Detecting Transport Layer Data Reneging, PFLDNeT, Lancaster, PA, 2010.
25. http://www.caida.org/data/passive/
26. N. Ekiz, A. Rahman, and P. Amer, Misbehaviors in TCP SACK Generation, ACM Computer Communications Review, 41(2), 2011.
27. The TCP Behavior Inference Tool, www.icir.org/tbit/
28. J. Iyengar, P. Amer, and R. Stewart, Receive Buffer Blocking in Concurrent Multipath Transport, IEEE GLOBECOM, St. Louis, MO, Nov. 2005.
29. P. Natarajan, J. Iyengar, P. Amer, and R. Stewart, Concurrent Multipath Transfer using SCTP Multihoming: Introducing Potentially-failed Destination State, IFIP International Conference on Networking, Singapore, May. 2008.

Fei Song received his Ph.D. degree from Beijing Jiaotong University. He is now a Lecturer in National Engineering Laboratory for Next Generation Internet Interconnection Devices, School of Electronic and Information Engineering, Beijing Jiaotong University. His current research interests are Protocols Optimization, Wireless Communications and Cloud Computing.

Huachun Zhou received his B.S. degree from People's Police Officer University of China in 1986. He also received his M.S. and Ph.D degree from Beijing Jiaotong University, Beijing, China in 1989 and 2008, respectively. He is now a professor in the National Engineering Laboratory for Next Generation Internet Interconnection Devices at Beijing Jiaotong University. His research interests include mobility management, mobile and secure computing, routing protocols, network management technologies and database applications.

Sidong Zhang received his M.S. degree in communication and information engineering from Beijing Jiaotong University in 1982. He is now a professor in Beijing Jiaotong University. His research interests are computer networks, wireless communications technology, ad-hoc networks, sensor networks and information theory.

Hongke Zhang received his M.S. and Ph.D. degrees in Electrical and Communication Systems from University of Electronic Science and Technology of China in 1988 and 1992, respectively. From Sep. 1992 to June 1994, he was a post-doc research associate at Beijing Jiaotong University. In July 1994, he joined the School of Electronic and Information Engineering, Beijing Jiaotong University, where he is a professor. He has published more than 100 research papers in the areas of communications, computer networks and information theory. He is the holder of more than 50 Chinese patents and is the Chief Scientist of a National Basic Research Program of China ("973 Program"). Dr. Zhang is the recipient of various awards such as the Zhan Tianyou Technical Innovation Award and the Mao Yisheng Technical Innovation Award.

Ilsun You received his M.S. and Ph.D. degrees in Computer Science from Dankook University, Seoul, Korea in 1997 and 2002, respectively. From 1997 to 2004, he worked for the THINmultimedia Inc., Internet Security Co., Ltd. and Hanjo Engineering Co., Ltd. as a Research Engineer. Since March 2005, he has been an Assistant Professor in the School of Information Science at the Korean Bible University, South Korea. Prof. You has served or is currently serving on the organizing or program committees of international conferences and workshops such as IMIS, MobiWorld, MIST 2009, MUE, UASS, TRUST, PCASS-06, WPS-08, FGCN-07, IPC-07, SAMNet-08 and so forth. He is in the editorial board for International Journal of Ad Hoc and Ubiquitous Computing (IJAHUC), Computing and Informatics (CAI), International Journal of Smart Home (IJSH) and Journal of Korean Society for Internet Information (KSII). Also, he has served as a guest editor of several journals such as CAI, MIS, AutoSoft and WCMC. His main research interests include?internet security, authentication, access control, MIPv6 and ubiquitous computing. He is a member of the IEICE, KIISC, KSII, KIPS, and IEEK.

Received: July 25, 2012; Accepted: January 05, 2013