Journal of Zhejiang University SCIENCE
ISSN 1009-3095
http://www.zju.edu.cn/jzus
E-mail: jzus@zju.edu.cn

A scalable admission control scheme based on time label

YANG Song-an (杨松岸)†, YANG Hua (杨 华), YANG Yu-hang (杨宇航)

(*Department of Electronic Engineering*, *Shanghai Jiaotong University*, *Shanghai 200030*, *China*)

†E-mail: yangsongan@sjtu.edu.cn

Received Aug. 18, 2003; revision accepted Oct. 6, 2003

Abstract: Resource reservation protocols allow communicating hosts to reserve resources such as bandwidth to offer guaranteed service. However, current resource reservation architectures do not scale well for a large number of flows. In this paper, we present a simple reservation protocol and a scalable admission control algorithm, which can provide QoS guarantees to individual flows without per-flow management in the network core. By mapping each flow to a definite time, this scheme addresses the problems that limit the effectiveness of current endpoint admission control schemes. The overall admission control process is described. Analysis is used to explain the reasonability of our scheme and simulation validates its performance.

Key words: Resource reservation, Admission control, QoS, Stream media
doi:10.1631/jzus.2004.1424 **Document code:** A **CLC number:** TB321

INTRODUCTION

The current Internet only provides a simple best effort service where the network treats all data packets equally. In this manner, the network keeps simple so that it can scale. Best effort service is sufficient for traditional Internet applications, such as email and web browse. But, with the advent of voice/video stream media and other real-time applications over the Internet, there is great demand for enhanced service on the Internet.

IntServ (Braden *et al*., 1997) and DiffServ (Blake *et al*., 1998) were brought out successively to provide QoS for the Internet. IntServ uses RSVP to reserve resources for each flow and can provide fine granularity QoS. But it needs per flow management on IP routers, so it does not scale well. DiffServ addresses the problem of IntServ and can be used in large scale. But it provides either premium service with the low utilization of the network or assured service with statistical guarantee and uncertain QoS of individual flow.

Recent research (Almesberger *et al*., 1997; Bianchi *et al*., 2000; Cetinkaya and Knightly, 2000; Elek *et al*., 2000), tried to combine the benefits of IntServ and DiffServ QoS solutions, proposed endpoint admission control. In these schemes, end hosts probe the network at the rate of the flow requesting a reservation. The end host admits the flow only if the loss rate of the probe traffic is less than a given threshold. The endpoint schemes present a novel approach to providing a scalable architecture for IntServ-like guarantees, but as Hill and Kung (2001) pointed out, additional work is needed before these schemes can provide the hard guarantees of traditional IntServ schemes. Bandwidth stealing and probe crowding are two main problems. They also tried to address these problems, but the scheme proposed by them still encountered problems. First, the scheme had special requirement for the probing traffic rate. Second it needed to maintain a number

* Project (No. 2003AA414025) supported by the National Basic Research Program (863) of China

of flows' state information, which increased the overhead at the core router.

In this paper, we propose a simple reservation protocol and a scalable admission control algorithm, which can provide hard guarantees to individual flow without per flow management in the network core. By mapping each flow to a definite time, this scheme addresses the problems that limit the effectiveness of current endpoint admission control schemes. Moreover, the simplicity of the scheme makes it scalable.

In Section 2, we present an analytical model to describe the behavior of the network. In Section 3, we present the detailed design of our scheme. Section 4 presents the simulation result of our scheme. In Section 5, we discuss some aspects of practical consideration related to our scheme. Section 6 concludes the paper.

ANALYTICAL MODEL

To provide QoS guarantees, a network must have enough resources, such as bandwidth and buffers. Resource reservation protocols allow communicating hosts to reserve resources to offer guaranteed service. In cooperation with resource reservation protocols, admission control is employed to offer guaranteed service. The principle of admission control is that the network does not take on new flows if the available resources are insufficient for them. Therefore, the QoS of admitted flows are protected from degrading.

The aim of endpoint admission control is to provide an end-to-end QoS without the overhead of maintaining per flow state. However, current endpoint admission control schemes suffer from the problems pointed out in Section 1. In our opinion, these problems all result from the network lacks of a mechanism to distinguish a flow from the others without per flow management, which make it impossible to do flow-based control. Based on the above point of view, we present a time-based competition scheme, which can provide end-to-end QoS to individual flow. In fact, we provide a mechanism to distinguish flows from each other without per flow management.

We first apply such model to describe the behavior of network flows.

(a) With time lapse, new flow will come into the network continuously at a limited rate $r(t)$, where $r(t)$ is a stochastic process.

(b) Each flow lasts for a limited period of time and then exits.

For such a system, the arrival rate and exit rate of flows tend to be equal and the system will come into a stable state in the end. If the network bandwidth can carry all the traffic when the system is in the stable state, no admission control is needed. But in practice, there is always insufficient bandwidth on the network. And the arrival of flows is in a random way, which causes the number of flows on the network to vary over the time. So we must control the flows that come into the network and regulate them to come into the network at the rate that we desire, other than the rate at which the flows generate. At the same time, we should also limit the total number of flows in the range that the network can carry, other than the number of flows when the system described above is in the stable state.

Then, let us consider how many flows will come in a period of time ΔT. Based on Description (a), $r(t)$ has supremum. Denote $N(t)$ the number of flows that enter the network in time ΔT; we get

$$N(t)=\int_{t}^{t+\Delta T} r(\tau)\mathrm{d}\tau \le \int_{t}^{t+\Delta T} \sup(r(\tau))\mathrm{d}\tau = r_{\max}\Delta T \tag{1}$$

where sup() denotes the supremum and $r_{\max}=\sup(r(t))$. Eq.(1) shows that in time slot $[t, t+\Delta T]$, there come no more than $r_{\max}\Delta T$ flows. If each flow is mapped to the time at which the flow comes into the network, we can use $[t, t+\Delta T]$ to identify the flows belonging to the time slot $[t, t+\Delta T]$. Though flows come at random, by changing ΔT, we can identify the flows with the desirable statistical resolution, thus we accomplish the control based on flow without maintaining the information of individual flow. With this guideline in mind, we develop the scalable admission control scheme described below.

DETAILED DESIGN

Our proposal covers a reservation protocol and an admission control algorithm. In our scheme, the network consists of source end host, edge router, core router and destination end host. By tagging based on the type of service (TOS) field, packets are classified into best effort packets and guaranteed packets and then treated differently. For each guaranteed packet, additional time label and admission status field are carried within the packet header.

Source end host and edge router functionality

When the two sides of communication need a guaranteed service, the source end host sends a request packet containing the information such as the average sending rate, the peak rate, the maximum burst length and the destination address, to the edge router to make a reservation. The edge router disposes the request packet and decides whether to take on this flow according to its current available resources. If reservation succeeds, the information on the flow will be installed in the edge router. The information includes not only the content contained in the request packet, but also the other two items: the admission time of the flow and a probing timer. In addition, a packet is also sent to the destination end host by the edge router to help the destination end host to locate the edge router. The source end host then sends the successive packets at the desire rate and the edge router marks these packets as guaranteed packets and marks the status field as admitted, and then forwards them to the destination. At the same time, the admission time of the flow is put into every packet of the flow. By doing so, each flow corresponds to a definite time and this feature will be used by core routers.

The edge router also does the job of monitoring the traffic of each individual flow. It uses traffic shaping and even discards packets to ensure that the traffic of all the admitted flows conforms to their specification. Note that the edge router admission of a flow does not mean the end-to-end admission, but only means the edge router can take on this flow. Since the edge router has a relatively small number of flows, it is feasible for the edge router to maintain state information on the individual flow.

Core router functionality

Core routers receive packets that are marked either as best effort or guaranteed. Best effort packets are forwarded at the best effort service level. For the guaranteed packets, further check is needed before they are forwarded. All the core routers maintain two time state variables for each of their out ports. One is the current candidate time (CANT), which determines whether a flow can be selected as candidate flow, and the other is the current admitted time (ADT), which determines whether a flow can be admitted. All the core routers use an admission control algorithm, which we call core admission control algorithm (CACA), to adjust the value of the time state variables according to current network conditions and to decide the operation on each guaranteed packet. This manipulation will be described in detail in the next section. Core routers can discard and degrade the guaranteed packets to cope with the excessive guaranteed packets. However, we prefer the former. Because the applications need a service with QoS guarantee, degrading the service level of these packets can make these packets meaningless to the applications and be transmitted in vain. Furthermore, if the applications use UDP as the underlying protocol, the guaranteed packets that are degraded have negative impact on the TCP flows that are serviced at the best effort level. So in the later sections of the paper, we will use discarding policy to cope with the excessive guaranteed packets.

Destination end host functionality

Destination end hosts monitor the packets status field of the guaranteed flows they receive and make decision on whether to send a report packet to the edge router in an observing period (OP). Once the destination end host receives one packet whose admission status field is marked as admitted by the monitored flow, which indicates that the flow is admitted by all the routers along the path the flow travels, the destination end host will send a report packet to the edge router to tell that the flow has been end to end admitted. Receiving the report packet, the edge router will disable the probing timer and keep maintaining the related information on the flow. If no packet marked as admitted is received, the destination end host sends no report packets to the edge router. This will cause the probing timer to timeout and the edge router will delete the information on the flow and inform the source end host to give up. The initial value of the probing timer (T_{timeout}) should satisfy the inequality below:

$$T_{\text{timeout}} > RTT_{\max} + OP \tag{2}$$

where $RTT_{\max}$ is the maximum round trip time.

Core admission control algorithm

Core admission control algorithm (CACA) is applied in all core routers to deal with the guaranteed packets. Because core routers do not maintain the information on individual flow, we use measurement to estimate the network traffic. How accurate can the measurement reflect the network traffic mainly depends on the selection of measurement parameters and the characteristics of the network traffic. In our scheme, we use a simple method that uses the average arrival rate and loss rate in a statistical period as the measurement parameter. In the next section, we will see that though the method is simple, we can achieve good performance. Besides the measurement module, CACA contains two other parts: a queue management module and a packet scheduler.

Measurement is done by taking the statistics of the network traffic periodically. In CACA, the guaranteed packets are classified into admitted packets, candidate packets and request packets at each node of the network and the flows these packets belong to are classified into admitted flows, candidate flows and request flows accordingly. Let *SP* denote the statistical period. In each *SP*, we count the number of the arrival packets, the volume of bits contained in the arrival packets and the number of dropped packets due to lack of buffers. At the end of each *SP*, we calculate the measurement parameters. We use *AD_PKT_CNT* and *AD_BIT_VOL* to denote the number and the bit volume of the admitted packets that arrive at one of the core router's out port in *SP* respectively. Correspondingly, we use *CAN_PKT_CNT* and *CAN_BIT_VOL* to denote the number and the bit volume of the candidate packets. We also use *AD_PKT_LOSS* and *CAN_PKT_LOSS* to denote the number of admitted packets and candidate packets that are dropped in *SP* respectively. Then we can get:

$$ADR = \frac{AD\_BIT\_VOL}{SP} \tag{3}$$

$$CANR = \frac{CAN\_BIT\_VOL}{SP} \tag{4}$$

$$ADLR = \frac{AD\_PKT\_LOSS}{AD\_PKT\_CNT} \tag{5}$$

$$CANLR = \frac{CAN\_PKT\_LOSS}{CAN\_PKT\_CNT} \tag{6}$$

where *ADR*, *CANR*, *ADLR* and *CANLR* represent the arrival rate of admitted flows, the arrival rate of candidate flow, the loss rate of admitted flows and the loss rate of candidate flows, respectively. We choose the four measurement parameters as the decision criteria for adjusting the two time state variables: *ADT* and *CANT*. Let *C* represent the capacity of the output link, then our algorithm can be described as follows.

Measurement module:

(a) At the end of each *SP*, measurement module calculates *ADR*, *CANR*, *ADLR* and *CANLR*, then resets the value of *AD_PKT_CNT*, *AD_BIT_VOL*, *AD_PKT_LOSS*, *CAN_PKT_CNT*, *CAN_BIT_VOL* and *CAN_PKT_LOSS* to zero.

(b) If *ADR* and *ADLR* are less than their given threshold, it is feasible to permit candidate flows to be admitted or to choose some request flows as candidates by increasing *ADT* and *CANT*. Otherwise, no candidate flows are permitted to be admitted and no request flows are permitted to be candidates.

(c) If *CANLR* is less than the given threshold, the measurement module lets these candidate flows be admitted flows by increasing the value of *ADT* to the value of *CANT*. Otherwise the measurement module will keep *ADT* unchanged and continue measuring the *CANLR* for the next *SP*.

Queue management module:

(a) Queue management module maintains two queues: one for admitted packets and the other for candidate packets.

(b) On receiving a packet, the queue management module checks its admission status field and reads its time label and then compares it to *ADT* and *CANT*. If the admission status field is "admitted" and the time label is earlier than *ADT*, the queue management module classifies the packet as admitted packet, and then puts it into the queue for admitted packets. If the admission status field is "admitted" and the time label is later than *ADT* and earlier than *CANT*, the queue management classifies the packet as candidate packet and marks its admission status field as "not admitted", and then puts it into the queue for candidate packets. If the admission status field is "not admitted" and its time label is early than *CANT*, the queue management module classifies the packet as candidate, and then puts it to the queue for candidate packet. If the time label is later than *CANT*, this packet is classified as request and will be dropped.

(c) According to the classification result, the queue management module increases the value of *AD_PKT_CNT* and *AD_BIT_VOL* or the value of *CAN_PKT_CNT* and *CAN_BIT_VOL* accordingly. If an admitted or candidate packet loss occurs due to lack of buffer space, the queue management module increments the value of *AD_PKT_LOSS* or *CAN_PKT_LOSS* accordingly. Especially for the candidate packet loss, the queue management module also decreases the value of CANT to force some of the candidate flows to become request flows.

Packet scheduler:

(a) If the queue for admitted packets is not empty, the scheduler sends the packets in this queue to the next hop.

(b) If the queue for admitted packets is empty and the queue for candidate packets is not empty, the scheduler sends the candidate packets to the next hop.

SIMULATIONS

We implemented our algorithm in the network simulator ns2 and simulated our scheme using the topology shown in Fig.1, where S, ER, CR and D represent the source end host, edge router, core router and destination end host respectively. Each S has a 2 Mbps connection with the edge router. The link between ER and CR is 30 Mbps and the link between CR and D is 10 Mbps. It was assumed that some bandwidth had already been reserved for the best effort packets, so no best effort packets appeared in the simulation.

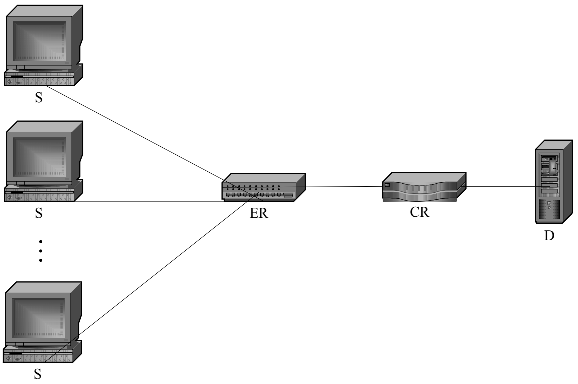

Fig.1 Simulation topology

We simulated two scenarios for CBR and VBR sources respectively. For both scenarios, the source end hosts each generated a flow directed to the same destination, which resulted in a total of 20 flows generated. These flows started uniformly at random within 50 milliseconds of the start of a simulation. And the average rate of each flow was 0.98 Mbps. We also forced the source end hosts not to stop sending packets during the simulation time.

For CBR source, the parameters were set as follows. The threshold for *ADR* was 0.98*C*, where *C* represents the capacity of the output link. The threshold for *ADLR* and *CANLR* were both set to 1%. *SP*=100 ms. The queue size for admitted packets and the queue size for candidate packets were both set to 10 packets.

Fig.2 shows the throughput of the 20 flows. We can see 10 flows' throughput is close to their sending rate, but the other 10 flows' throughput was nearly zero. This showed that with our scheme, we could treat different flows differently without maintaining the information on individual flow at core routers.

Table 1 shows the admission status of each flow and the packet loss probability of the admitted flows. We can see that only ten flows are end to end admitted, which is relative to the throughput of each flow shown in Fig.2. We can also see that the admitted flows suffer very low loss probability. This proved that the admitted flows were well protected though there were other flows trying to share the bandwidth.

Table 1 Admission status and packet loss probability

Flow ID	Admission status	Packet loss rate
0	Not admitted	–
1	Admitted	0
2	Not admitted	–
3	Admitted	0
4	Admitted	0
5	Admitted	0
6	Not admitted	–
7	Not admitted	–
8	Admitted	0
9	Not admitted	–
10	Admitted	0.04%
11	Not admitted	–
12	Admitted	0.06%
13	Not admitted	–
14	Admitted	0.01%
15	Admitted	0.06%
16	Not admitted	–
17	Not admitted	–
18	Not admitted	–
19	Admitted	0.05%

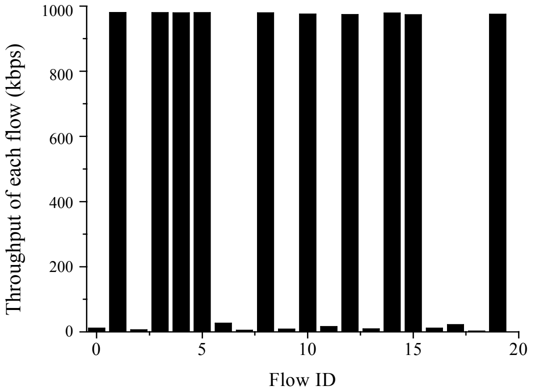

Fig.2 Throughput of each flow

Fig.3 shows the total arrival rate of the admitted flows, which is measured by the measurement module in each SP. We can see that the admitted traffic is consistent with the bottleneck bandwidth 10 Mbps.

For the VBR source, we assumed that the packet interval time distributes exponentially. We used large queue size to overcome the burstness of the VBR sources. The queue size for the admitted packets and candidate packets were both set to 30 packets. The other simulation parameters were set the same as that for CBR sources.

Fig.4 shows the throughput of the 20 flows. Table 2 shows the admission status of each flow and the packet loss probability of the admitted flows. Similar to the CBR sources, we can see 10 flows are admitted, and that the admitted flows have throughput that is close to their sending rate, while the other 10 flows' throughput is nearly zero. However, from Table 2 we can see that the loss probability of admitted flows is higher than that of CBR sources. There are two explanations for this situation. One hand, for the VBR source, the measurement of the mean packet arrival rate may deviate from the real value, so it is possible to admit excessive flows. On the other hand, even if the average packet arrival rate of the admitted flows does not exceed the link bandwidth, the high peak rate burst of VBR traffic can cause buffer overflow, which result in packet loss. Decreasing the threshold for *ADR* and the threshold for *ADLR* and *CANLR* is helpful for coping with this problem and produces better performance. However, this is achi-

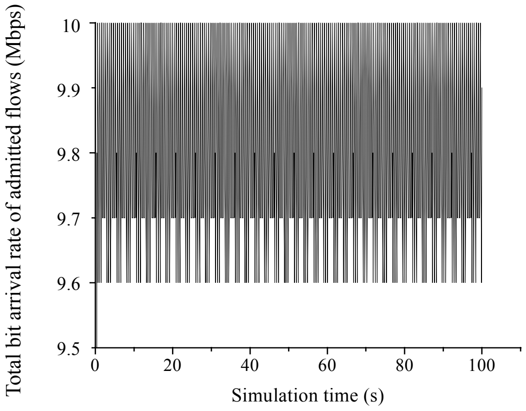

Fig.3 Total bit arrival rate of admitted flows

eved at the cost of low utilization of link bandwidth. More complex algorithm, which selects the threshold for *ADR*, *ADLR* and *CANLR* adaptively according the traffic characteristics, can be developed to make maximum use of the link bandwidth on condition that QoS is guaranteed.

Fig.5 shows the total arrival rate of the admitted flows. We can see that the average rate of the admitted traffic is also consistent with the bottleneck bandwidth. However, the measurement value in each *SP* fluctuates around the threshold for *ADR* over time, which causes more packets loss.

PRATICAL CONSIDERATION

Our scheme faces two challenges. One is that our scheme requires the clocks of all the routers (including edge routers and core routers) be synchronous. But the fact is that clocks on the Internet are not synchronous. However, as far as technology is concerned, it is no problem to synchronize the clocks on the Internet. One way is to rely on the network itself. Using network time protocol NTP (Mills, 1992), we can synchronize the clocks on the Internet to millisecond accuracy. Another way is using the time service of GPS. By this means we can achieve more accurate synchronization.

Another challenge to our scheme is how to carry the time label and the admission status field with packets. Because routers dispose packets according to the information in network layer header, the time label should be put into the network layer header for the sake of efficiency. Unfortunately, there is no such field in IP header to contain the time label. However, in the IP header, there is a time stamp optional field. In this field, three kinds of time stamp had already been defined. We can add the fourth kind of time stamp to this field for our purpose to make it possible for the packets to carry the time label. For the status field, we can use the unused bit of TOS field.

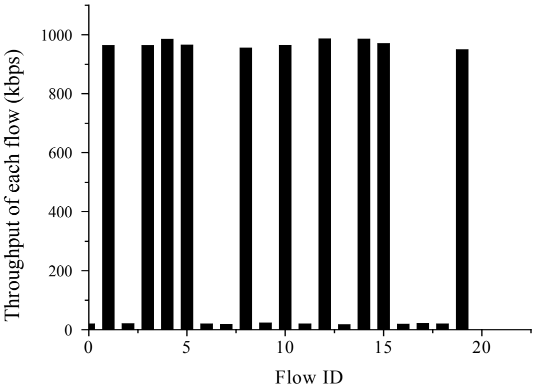

Fig.4 Throughput of each flow

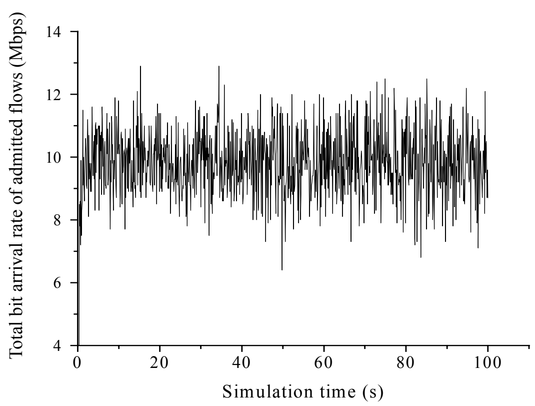

Fig.5 Total bit arrival rate of admitted flows

Table 2 Admission status and packet loss probability

Flow ID	Admission status	Packet loss rate
0	Not admitted	–
1	Admitted	1.50%
2	Not admitted	–
3	Admitted	1.36%
4	Admitted	1.37%
5	Admitted	1.64%
6	Not admitted	–
7	Not admitted	–
8	Admitted	1.44%
9	Not admitted	–
10	Admitted	1.97%
11	Not admitted	–
12	Admitted	1.19%
13	Not admitted	–
14	Admitted	1.26%
15	Admitted	1.46%
16	Not admitted	–
17	Not admitted	–
18	Not admitted	–
19	Admitted	1.42%

CONCLUSION

In this paper, we present a scalable resource reservation protocol and an admission control algorithm, which are able to provide hard guarantees to individual flows without per-flow management in the network core. By using analysis, we proved the reasonability of our scheme. We also used simulation to validate its performance. Our scheme eliminates the problem of probe crowding because the admission control algorithm can selectively choose the candidate flows and can drop the excessive candidate flows at any time. By letting the candidate flows transmitted accompany the admitted flows and become admitted flow only when the candidate flows receive the same service level as admitted flows for some time, we also ensure that the problem of bandwidth stealing seldom occurs. For the CBR flows, our scheme provides good QoS. For the VBR flows, though, they have higher packet loss probability compared to CBR flows, because of the burst of traffic, we can still keep the packet loss probability at a very low level.

References

Almesberger, W., Le Boudec, J.Y., Ferrari, T., 1997. Scalable Resource Reservation for the Internet. Proceedings of IWQOS'97.

Bianchi, G., Capone, A., Petrioli, C., 2000. Throughput Analysis of End-to-End Measurement-Based Admission Control in IP. Proceedings of INFOCOM .

Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., Weiss, W., 1998. An Architecture for Differentiated Service. IETF RFC 2475.

Braden, R., Zhang, L., berson, S., Herzog, S., Jasmin, S., 1997. Resource Reservation Protocol (RSVP) – Version 1 Functional Specification. IETF RFC 2205.

Cetinkaya, C., Knightly, E., 2000. Egress Admission Control. Proceedings of INFOCOM 2000.

Elek, V., Karlsson, G., Ronngren, R., 2000. Admission Control Based on End-to-End Measurements. Proceedings of INFOCOM 2000.

Hill, R., Kung, H.T., 2001. A Diff-Serv Enhanced Admission Control Scheme. Proceedings of IEEE Globecom 2001.

Mills, D.L., 1992. Network Time Protocol. RFC1305.